



LAUREATE INTERNATIONAL UNIVERSITIES®

CDMX, 29 de Abril de 2025.

*Actividad 8. Proyecto integrador. Etapa 3
Base de datos relacionales.*

Miranda Velazquez Mariles
Jose Emiliano Jauregui Guzman
Alfonso Roberto Gómez Rosales



Índice

ETAPA 1: PROYECTO INTEGRADOR

I. DBMS Sistema manejador de base de datos

- 1.1 Instalación
- 1.2 Creación de base de datos de prueba

II. Modelo Entidad – Relación

- 2.1 Identificación de entidades y atributos
- 2.2 Generación de diagrama Entidad-Relación

ETAPA 2: PROYECTO INTEGRADOR

III. Creación y llenado de la base de datos

- 3.1 Creación de la estructura de la base de datos
- 3.2 Llenado de la base de datos

ETAPA 3: PROYECTO INTEGRADOR

IV. Generación de consultas, procedimientos almacenados y vistas

- 4.1 Generación de consultas
- 4.2 Generación de procedimientos almacenados
- 4.3 Generación de vistas



ETAPA 1: PROYECTO INTEGRADOR

Introducción

Esta actividad consiste en aplicar los conocimientos adquiridos a lo largo del curso y retomar lo aprendido en cada una de las actividades realizadas, lo que garantiza la transversalidad de los contenidos revisados para fortalecer el desarrollo de competencias y lograr el fin de formación planteado.

Objetivo

El objetivo del Proyecto integrador es aplicar los conceptos de bases de datos en la generación de la estructura y manipulación de la información en una base de datos relacional, utilizando un software especializado para ello.

Planteamiento

Cierta institución educativa ha solicitado el desarrollo de un prototipo de sistema de control escolar para la gestión de alumnos, profesores y asignaturas. Tu objetivo es desarrollar la estructura de base de datos necesaria para cubrir la solicitud.

Inicialmente se plantea el siguiente Modelo entidad - relación.



I. DBMS Sistema manejador de base de datos

1.1 Instalación

④ MySQL Product Archives

« MySQL Workbench (Archived Versions)

⚠ Please note that these are old versions. New releases will have recent bug fixes and features!
To download the latest release of MySQL Workbench, please visit [MySQL Downloads](#).

Product Version: 8.0.40
Operating System: macOS
OS Version: All

Packages for Sonoma (14) are compatible with Sequoia (15)			
macOS (ARM, 64-bit), DMG Archive (mysql-workbench-community-8.0.40-macos-arm64.dmg)	Oct 30, 2024	119.0M	Download
macOS (x86, 64-bit), DMG Archive (mysql-workbench-community-8.0.40-macos-x86_64.dmg)	Oct 30, 2024	121.4M	Download
We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.			

MySQL open source software is provided under the [GPL License](#).

④ MySQL Product Archives

« MySQL Community Server (Archived Versions)

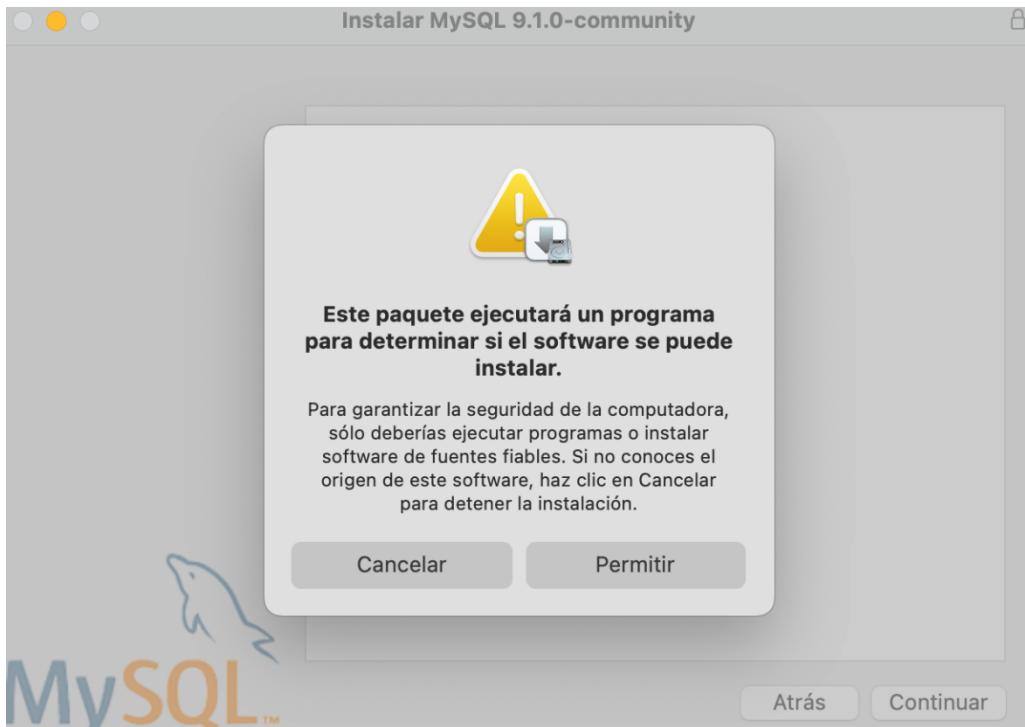
⚠ Please note that these are old versions. New releases will have recent bug fixes and features!
To download the latest release of MySQL Community Server, please visit [MySQL Downloads](#).

Product Version: 9.1.0
Operating System: macOS
OS Version: All

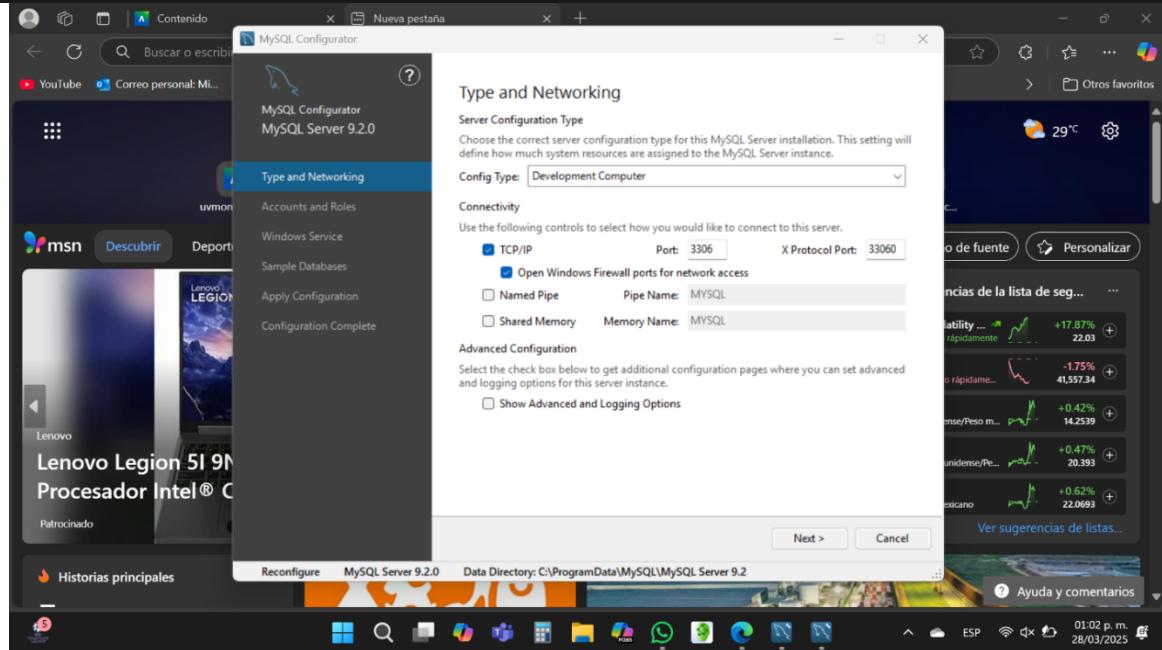
Packages for Sonoma (14) are compatible with Sequoia (15)			
macOS 14 (ARM, 64-bit), DMG Archive (mysql-9.1.0-macos14-arm64.dmg)	Sep 24, 2024	584.7M	Download
macOS 14 (x86, 64-bit), DMG Archive (mysql-9.1.0-macos14-x86_64.dmg)	Sep 24, 2024	589.5M	Download
macOS 14 (ARM, 64-bit), Compressed TAR Archive (mysql-9.1.0-macos14-arm64.tar.gz)	Sep 24, 2024	159.2M	Download
macOS 14 (x86, 64-bit), Compressed TAR Archive (mysql-9.1.0-macos14-x86_64.tar.gz)	Sep 24, 2024	163.3M	Download

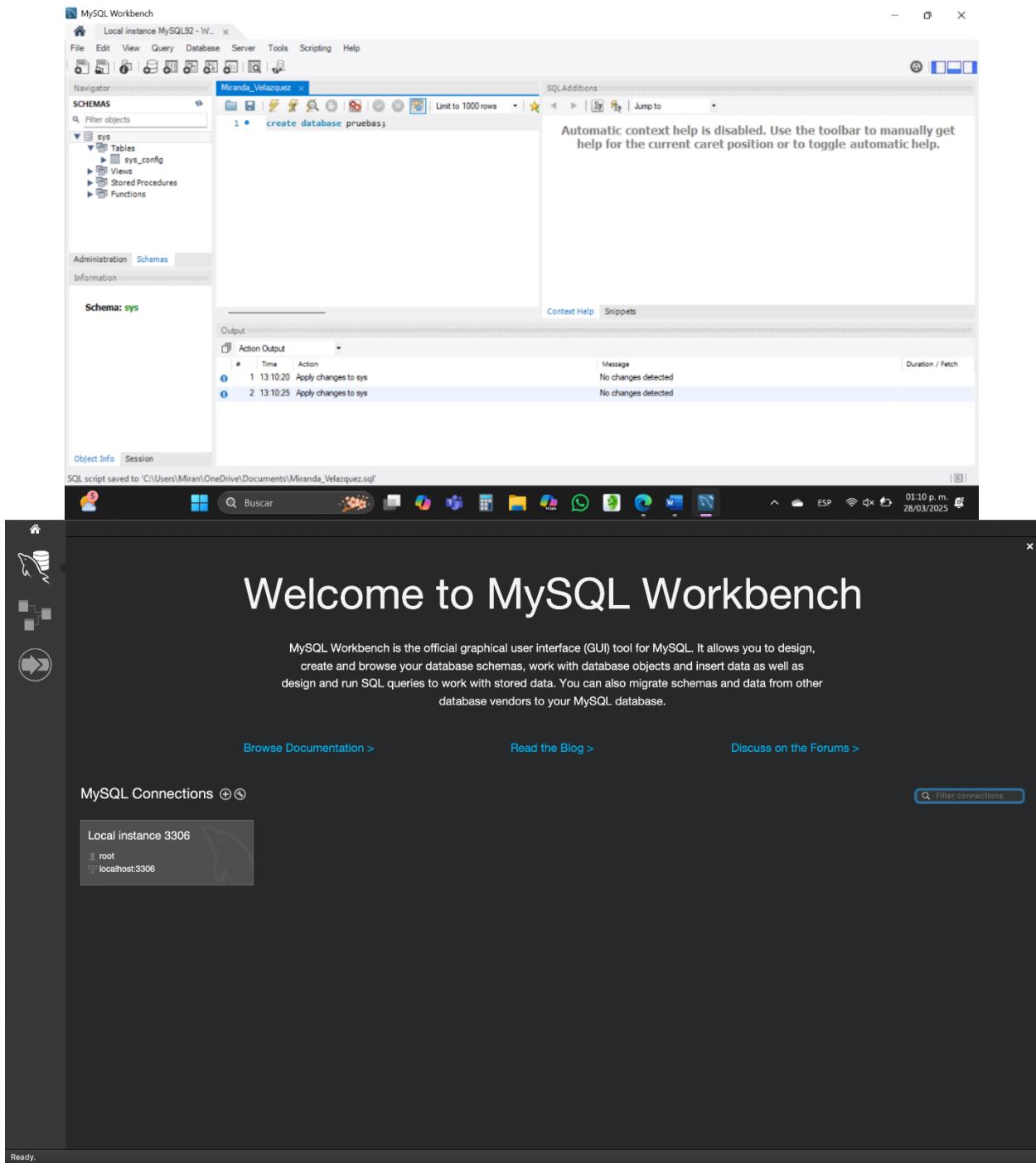






mysql-9.2.0-winx64	28/03/2025 12:06 p. m.	Paquete de Windows...
mysql-workbench-community-8.0.40-winx64	28/03/2025 12:38 p. m.	Paquete de Windows...
VC_redist.x64	28/03/2025 12:11 p. m.	Aplicación





1.2 Creación de base de datos de prueba

The screenshot shows the MySQL Workbench interface. In the top left, under 'Administration' > 'Schemas', there is a list of databases: 'Jose_Jauregui' and 'sys'. The main area is a query editor with the following command:

```
1 create database Jose_Jauregui;
```

To the right of the query, a note says: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

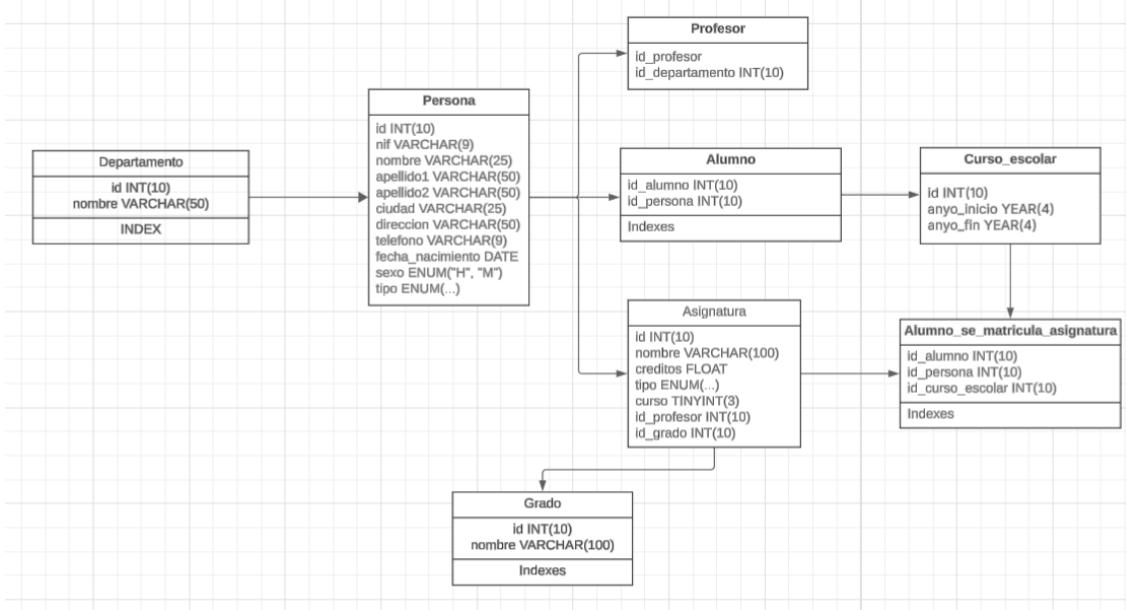
Below the query editor, the 'Action Output' pane shows the execution details:

Action	Time	Response	Duration / Fetch Time
create database Jose_Jauregui	16:05:19	1 row(s) affected	0.0031 sec

At the bottom, it says 'Query Completed'.

II. Modelo Entidad - Relación

2.1 Identificación de entidades y atributos



2.2 Generación de diagrama Entidad-Relación

El diagrama representa una base de datos para gestionar información de una institución educativa, donde se contemplan personas, alumnos, profesores, departamentos,



asignaturas, grados y cursos escolares. A continuación, se describen las tablas y sus relaciones principales:

1. Departamento

- a. **id** (INT): identificador único del departamento.
- b. **nombre** (VARCHAR(100)): nombre del departamento.
- c. Contiene un índice (INDEX) para optimizar búsquedas basadas en el nombre.
- d. Relación con **Profesor**: un departamento puede tener varios profesores (relación 1 a muchos).

2. Persona

- a. **id** (INT): identificador único de la persona.
- b. **dni** (VARCHAR(9)): documento de identidad.
- c. **nombre, apellido1, apellido2** (VARCHAR(25)): datos de identificación personal.
- d. **ciudad_nacimiento, ciudad_residencia** (VARCHAR(50)): información de ubicación.
- e. **fecha_nacimiento** (DATE): fecha de nacimiento.
- f. **sexo** (ENUM('H','M')): género de la persona.
- g. **tipo** (ENUM): posible campo para distinguir roles o categorías de persona.

Esta tabla es la base para almacenar datos personales, pudiendo relacionarse con **Alumno** y **Profesor**.

3. Profesor

- a. **id_profesor** (INT): identificador único del profesor (probablemente foráneo de Persona o se relaciona con ella).
- b. **id_departamento** (INT): clave foránea que vincula al profesor con un departamento.

Un profesor hereda o se asocia a la información básica de **Persona** (normalmente mediante una relación 1 a 1 o 1 a muchos, dependiendo de la lógica de diseño).

4. Alumno

- a. **id_alumno** (INT): identificador único del alumno (relacionado con Persona).
- b. **id_persona** (INT): clave foránea que vincula al alumno con su información personal en la tabla **Persona**.

Similar a **Profesor**, el alumno hereda datos personales de **Persona** y se especializa en el rol de alumno.

5. Curso_escolar

- a. **id_curso_escolar** (INT, no se ve en el diagrama pero se asume como PK).
- b. **anyo_inicio** (YEAR(4)): año en que inicia el curso escolar.
- c. **anyo_fin** (YEAR(4)): año en que finaliza el curso escolar.

Permite identificar períodos académicos.



6. Asignatura

- a. **id_asignatura** (INT): identificador único de la asignatura.
- b. **nombre** (VARCHAR(100)): nombre de la asignatura.
- c. **creditos** (FLOAT): créditos de la asignatura.
- d. **tipo** (ENUM): campo que describe la naturaleza de la asignatura (troncal, optativa, etc.).
- e. **curso** (INT): puede indicar el año o nivel en el que se imparte la asignatura.
- f. **id_grado** (INT): clave foránea que indica a qué grado pertenece la asignatura.

7. Grado

- a. **id** (INT): identificador único del grado.
- b. **nombre** (VARCHAR(100)): nombre del grado (p.ej. “Ingeniería Informática”, “ADE”, etc.).

Un grado puede tener múltiples asignaturas asociadas (relación 1 a muchos).

8. Alumno_se_matricula_asignatura

- a. **id** (INT): identificador de la matrícula o registro de matrícula.
- b. **id_persona** (INT): clave foránea para identificar al alumno (a través de Persona o directamente relacionando con Alumno).
- c. **id_asignatura** (INT): clave foránea para la asignatura en la que se matricula el alumno.
- d. **id_curso_escolar** (INT): clave foránea para el curso escolar en el que se realiza la matrícula.

Esta tabla intermedia permite gestionar la relación de muchos a muchos entre alumnos y asignaturas en distintos cursos escolares.



ETAPA 2: PROYECTO INTEGRADOR

III. Creación y llenado de la base de datos.

3.1 Creación de la estructura de la base de datos.

- Genera el script.sql con las instrucciones necesarias para crear la estructura de base de datos con las tablas, campos y tipos de datos que definiste.

script.sql

```
CREATE DATABASE IF NOT EXISTS Universidad;
USE Universidad;
```

```
CREATE TABLE Departamento (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(100) NOT NULL,
    INDEX(nombre)
);
```

```
CREATE TABLE Persona (
    id INT PRIMARY KEY AUTO_INCREMENT,
    dni VARCHAR(9) UNIQUE NOT NULL,
    nombre VARCHAR(25) NOT NULL,
    apellido1 VARCHAR(50) NOT NULL,
    apellido2 VARCHAR(50),
    ciudad_nacimiento VARCHAR(50),
    ciudad_residencia VARCHAR(50),
    fecha_nacimiento DATE,
    sexo ENUM('H','M'),
    tipo ENUM('Profesor', 'Alumno') NOT NULL
);
```

```
CREATE TABLE Profesor (
    id_profesor INT PRIMARY KEY AUTO_INCREMENT,
    id_persona INT UNIQUE NOT NULL,
    id_departamento INT,
    FOREIGN KEY (id_persona) REFERENCES Persona(id) ON DELETE CASCADE,
    FOREIGN KEY (id_departamento) REFERENCES Departamento(id) ON DELETE
    SET NULL
);
```

```
CREATE TABLE Alumno (
    id_alumno INT PRIMARY KEY AUTO_INCREMENT,
    id_persona INT UNIQUE NOT NULL,
    FOREIGN KEY (id_persona) REFERENCES Persona(id) ON DELETE CASCADE
);
```

```
CREATE TABLE Curso_escolar (
```



```

id INT PRIMARY KEY AUTO_INCREMENT,
anyo_inicio YEAR(4) NOT NULL,
anyo_fin YEAR(4) NOT NULL
);

CREATE TABLE Grado (
  id INT PRIMARY KEY AUTO_INCREMENT,
  nombre VARCHAR(100) NOT NULL
);

CREATE TABLE Asignatura (
  id INT PRIMARY KEY AUTO_INCREMENT,
  nombre VARCHAR(100) NOT NULL,
  creditos FLOAT NOT NULL,
  tipo ENUM('Obligatoria', 'Optativa', 'Troncal') NOT NULL,
  curso TINYINT NOT NULL,
  id_profesor INT,
  id_grado INT NOT NULL,
  FOREIGN KEY (id_profesor) REFERENCES Profesor(id_profesor) ON DELETE SET NULL,
  FOREIGN KEY (id_grado) REFERENCES Grado(id) ON DELETE CASCADE
);

CREATE TABLE Alumno_se_matricula_asignatura (
  id INT PRIMARY KEY AUTO_INCREMENT,
  id_alumno INT NOT NULL,
  id_asignatura INT NOT NULL,
  id_curso_escolar INT NOT NULL,
  FOREIGN KEY (id_alumno) REFERENCES Alumno(id_alumno) ON DELETE CASCADE,
  FOREIGN KEY (id_asignatura) REFERENCES Asignatura(id) ON DELETE CASCADE,
  FOREIGN KEY (id_curso_escolar) REFERENCES Curso_escolar(id) ON DELETE CASCADE
);

```

- Ejecuta tu script y valida que tus tablas se hayan creado conforme a tu diagrama, si hiciste cambios, justifícalos. Utiliza las funciones DESCRIBE y SHOW TABLES.





Local Instance 3306 - Warning - not supported

Administration Schemas SQL File 3* SQL File 5*

SCHEMAS Filter objects

Jose_Jauregui Tables

- Alumno
- Asignatura
- Curso_escolar
- Departamento
- Grado
- Persona
- Profesor

Views Stored Procedures Functions

sys Universidad

Object Info Session Schema: Jose_Jauregui

```

13  dni VARCHAR(9) UNIQUE NOT NULL,
14  nombre VARCHAR(25) NOT NULL,
15  apellido1 VARCHAR(50) NOT NULL,
16  apellido2 VARCHAR(50),
17  ciudad_nacimiento VARCHAR(50),
18  ciudad_residencia VARCHAR(50),
19  fecha_nacimiento DATE,
20  sexo ENUM('H','M'),
21  tipo ENUM('Profesor', 'Alumno') NOT NULL
22  );
23
24  DESCRIBE Profesor;
25  
```

100% 19:24

Result Grid Filter Rows: Search Export: Result Grid

Field	Type	Null	Key	Default	Extra
id_profesor	int	NO	PRI	HULL	auto_increment
id_persona	int	NO	UNI	HULL	
id_departamento	int	YES	MUL	HULL	

Result 3 Read Only

Action Output

Time	Action	Response	Duration / Fetch Time
00:15:41	CREATE TABLE Curso_escolar (id INT PRIMARY KEY AUTO_INCREMENT,...	0 row(s) affected, 2 warning(s): 1287 'YEAR(4)' is d...	0.0060 sec
00:15:47	CREATE TABLE Grado (id INT PRIMARY KEY AUTO_INCREMENT,...	0 row(s) affected	0.0046 sec
00:15:51	CREATE TABLE Asignatura (id INT PRIMARY KEY AUTO_INCREMENT,...	0 row(s) affected	0.010 sec
00:16:01	CREATE TABLE Alumno_se_matricula_asignatura (id INT PRIMARY KEY A...	0 row(s) affected	0.011 sec
00:17:41	DESCRIBE Departamento	2 row(s) returned	0.0014 sec / 0.00001...
00:18:10	DESCRIBE Persona	10 row(s) returned	0.0023 sec / 0.00001...
00:18:40	DESCRIBE Profesor	3 row(s) returned	0.0022 sec / 0.00001...

Query Completed Local Instance 3306 - Warning - not supported

Administration Schemas SQL File 3* SQL File 5*

SCHEMAS Filter objects

Jose_Jauregui Tables

- Alumno
- Asignatura
- Curso_escolar
- Departamento
- Grado
- Persona
- Profesor

Views Stored Procedures Functions

sys Universidad

Object Info Session Schema: Jose_Jauregui

```

25  CREATE TABLE Profesor (
26    id_profesor INT PRIMARY KEY AUTO_INCREMENT,
27    id_persona INT UNIQUE NOT NULL,
28    id_departamento INT,
29    FOREIGN KEY (id_persona) REFERENCES Persona(id) ON DELETE CASCADE,
30    FOREIGN KEY (id_departamento) REFERENCES Departamento(id) ON DELETE SET NULL
31  );
32  DESCRIBE Alumno;
33  CREATE TABLE Alumno (
34    id_alumno INT PRIMARY KEY AUTO_INCREMENT,
35    id_persona INT UNIQUE NOT NULL,
36    FOREIGN KEY (id_persona) REFERENCES Persona(id) ON DELETE CASCADE
37  );
38
100% 17:32

```

Result Grid Filter Rows: Search Export: Result Grid

Field	Type	Null	Key	Default	Extra
id_alumno	int	NO	PRI	HULL	auto_increment
id_persona	int	NO	UNI	HULL	

Result 4 Read Only

Action Output

Time	Action	Response	Duration / Fetch Time
00:15:47	CREATE TABLE Grado (id INT PRIMARY KEY AUTO_INCREMENT,...	0 row(s) affected	0.0046 sec
00:15:51	CREATE TABLE Asignatura (id INT PRIMARY KEY AUTO_INCREMENT,...	0 row(s) affected	0.010 sec
00:16:01	CREATE TABLE Alumno_se_matricula_asignatura (id INT PRIMARY KEY A...	0 row(s) affected	0.011 sec
00:17:41	DESCRIBE Departamento	2 row(s) returned	0.0014 sec / 0.00001...
00:18:10	DESCRIBE Persona	10 row(s) returned	0.0023 sec / 0.00001...
00:18:40	DESCRIBE Profesor	3 row(s) returned	0.0022 sec / 0.00001...
00:19:03	DESCRIBE Alumno	2 row(s) returned	0.0027 sec / 0.00000...

Query Completed



Local Instance 3306 - Warning - not supported

Administration Schemas SQL File 3* SQL File 5*

SCHEMAS Filter objects

Jose_Jauregui Tables

- Alumno
- Asignatura
- Curso_escolar
- Departamento
- Grado
- Persona
- Profesor
- Views
- Stored Procedures
- Functions

sys Universidad

Object Info Session Schema: Jose_Jauregui

```

31   );
32   DESCRIBE Alumno;
33   CREATE TABLE Alumno (
34       id_alumno INT PRIMARY KEY AUTO_INCREMENT,
35       id_persona INT UNIQUE NOT NULL,
36       FOREIGN KEY (id_persona) REFERENCES Persona(id) ON DELETE CASCADE
37   );
38
39   DESCRIBE Curso_escolar;
40   CREATE TABLE Curso_escolar (
41       id INT PRIMARY KEY AUTO_INCREMENT,
42       anyo_inicio YEAR(4) NOT NULL,
43       anyo_fin YEAR(4) NOT NULL
44   );

```

100% 24:39

Result Grid Filter Rows: Search Export: Result Grid Form Editor

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
anyo_inicio	year	NO		NULL	
anyo_fin	year	NO		NULL	

Result 5 Read Only

Action Output

Action	Time	Response	Duration / Fetch Time
CREATE TABLE Asignatura (00:15:51	0 row(s) affected	0.010 sec
CREATE TABLE Alumno_se_matricula_asignatura (00:16:01	0 row(s) affected	0.011 sec
DESCRIBE Departamento	00:17:41	2 row(s) returned	0.0014 sec / 0.00001...
DESCRIBE Persona	00:18:10	10 row(s) returned	0.0023 sec / 0.00001...
DESCRIBE Profesor	00:18:40	3 row(s) returned	0.0022 sec / 0.00001...
DESCRIBE Alumno	00:19:03	2 row(s) returned	0.0027 sec / 0.00000...
DESCRIBE Curso_escolar	00:19:26	3 row(s) returned	0.0027 sec / 0.00000...

Query Completed Local Instance 3306 - Warning - not supported

Administration Schemas SQL File 3* SQL File 5*

SCHEMAS Filter objects

Jose_Jauregui Tables

- Alumno
- Asignatura
- Curso_escolar
- Departamento
- Grado
- Persona
- Profesor
- Views
- Stored Procedures
- Functions

sys Universidad

Object Info Session Schema: Jose_Jauregui

```

37   );
38
39   DESCRIBE Curso_escolar;
40   CREATE TABLE Curso_escolar (
41       id INT PRIMARY KEY AUTO_INCREMENT,
42       anyo_inicio YEAR(4) NOT NULL,
43       anyo_fin YEAR(4) NOT NULL
44   );
45
46   DESCRIBE Grado;
47   CREATE TABLE Grado (
48       id INT PRIMARY KEY AUTO_INCREMENT,
49       nombre VARCHAR(100) NOT NULL
50   );
51

```

100% 16:46

Result Grid Filter Rows: Search Export: Result Grid Form Editor

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
nombre	varchar(100)	NO		NULL	

Result 6 Read Only

Action Output

Action	Time	Response	Duration / Fetch Time
CREATE TABLE Alumno_se_matricula_asignatura (00:16:01	0 row(s) affected	0.011 sec
DESCRIBE Departamento	00:17:41	2 row(s) returned	0.0014 sec / 0.00001...
DESCRIBE Persona	00:18:10	10 row(s) returned	0.0023 sec / 0.00001...
DESCRIBE Profesor	00:18:40	3 row(s) returned	0.0022 sec / 0.00001...
DESCRIBE Alumno	00:19:03	2 row(s) returned	0.0027 sec / 0.00000...
DESCRIBE Curso_escolar	00:19:26	3 row(s) returned	0.0027 sec / 0.00000...
DESCRIBE Grado	00:19:51	2 row(s) returned	0.0021 sec / 0.00000...

Query Completed



The screenshot shows the MySQL Workbench interface with the following details:

- Administration** tab selected.
- Schemas** pane: Schema **Jose_Jauregui** selected.
- Tables** pane: Tables listed include **Alumno**, **Asignatura**, **Curso_escolar**, **Departamento**, **Grado**, **Persona**, **Profesor**, **Views**, **Stored Procedures**, and **Functions**.
- SQL Editor** pane:
 - SQL code:

```
46 • DESCRIBE Grado;
47 • ○ CREATE TABLE Grado (
48     id INT PRIMARY KEY AUTO_INCREMENT,
49     nombre VARCHAR(100) NOT NULL
50 );
51
52 • DESCRIBE Asignatura;
53 • ○ CREATE TABLE Asignatura (
54     id INT PRIMARY KEY AUTO_INCREMENT,
55     nombre VARCHAR(100) NOT NULL,
56     creditos FLOAT NOT NULL,
57 );
```
 - Result Grid (empty)
 - Action Output (empty)
- Context Help** and **Snippets** panes are visible on the right.

Query Completed

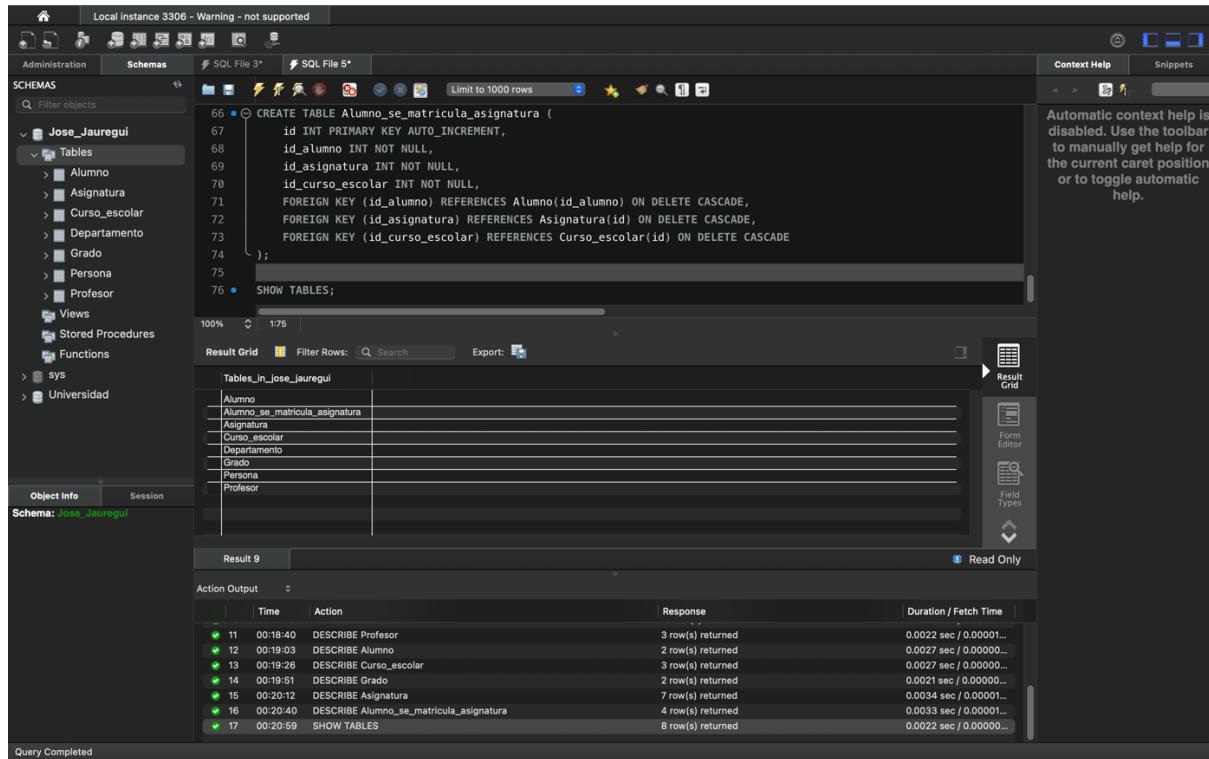
The screenshot shows the MySQL Workbench interface with the following details:

- Administration** tab selected.
- Schemas** pane: Schema **Jose_Jauregui** selected.
- Tables** pane: Tables listed include **Alumno**, **Asignatura**, **Curso_escolar**, **Departamento**, **Grado**, **Persona**, **Profesor**, **Views**, **Stored Procedures**, and **Functions**.
- SQL Editor** pane:
 - SQL code:

```
60
61     FOREIGN KEY (id_profesor) REFERENCES Profesor(id_profesor) ON DELETE SET NULL,
62     FOREIGN KEY (id_grado) REFERENCES Grado(id) ON DELETE CASCADE
63 );
64
65 • DESCRIBE Alumno_se_matricula_asignatura;
66 • ○ CREATE TABLE Alumno_se_matricula_asignatura (
67     id INT PRIMARY KEY AUTO_INCREMENT,
68     id_alumno INT NOT NULL,
69     id_asignatura INT NOT NULL,
70     id_curso_escolar INT NOT NULL,
71     FOREIGN KEY (id_alumno) REFERENCES Alumno(id_alumno) ON DELETE CASCADE,
72 );
```
 - Result Grid (empty)
 - Action Output (empty)
- Context Help** and **Snippets** panes are visible on the right.

Query Completed





3.2 Llenado de la base de datos.

- Una vez creada tu base de datos ingresa registros en ella. Para ello utiliza la función INSERT. En la referencia del punto anterior en el apartado 1.5.3 encontrarás datos de prueba para el diagrama propuesto, si te son de utilidad puedes usarlos.
- Genera un script sql con las instrucciones de inserción a tus tablas, ejecútalo y valida en Workbench que tus inserciones sean exitosas. Toma capturas de pantalla de tu proceso. Puedes ejecutar tu script por fragmentos.

Script crear tabla:

```

-- Tabla Departamento CREATE TABLE Departamento ( id INT PRIMARY KEY AUTO_INCREMENT, nombre VARCHAR(100) NOT NULL );

-- Tabla Persona CREATE TABLE Persona ( id INT PRIMARY KEY AUTO_INCREMENT, dni VARCHAR(9) UNIQUE NOT NULL, nombre VARCHAR(25) NOT NULL, apellido1 VARCHAR(50) NOT NULL, apellido2 VARCHAR(50), ciudad_nacimiento VARCHAR(50), ciudad_residencia VARCHAR(50), fecha_nacimiento DATE, sexo ENUM('H','M'), tipo ENUM('Profesor', 'Alumno') NOT NULL );

-- Tabla Profesor CREATE TABLE Profesor ( id_profesor INT PRIMARY KEY AUTO_INCREMENT, id_persona INT UNIQUE NOT NULL, id_departamento INT, FOREIGN KEY (id_persona) REFERENCES Persona(id) ON DELETE CASCADE,
  
```



FOREIGN KEY (id_departamento) REFERENCES Departamento(id) ON DELETE SET NULL);

-- Tabla Alumno CREATE TABLE Alumno (id_alumno INT PRIMARY KEY AUTO_INCREMENT, id_persona INT UNIQUE NOT NULL, FOREIGN KEY (id_persona) REFERENCES Persona(id) ON DELETE CASCADE);

-- Tabla Curso_escolar CREATE TABLE Curso_escolar (id INT PRIMARY KEY AUTO_INCREMENT, anyo_inicio YEAR(4) NOT NULL, anyo_fin YEAR(4) NOT NULL);

-- Tabla Grado CREATE TABLE Grado (id INT PRIMARY KEY AUTO_INCREMENT, nombre VARCHAR(100) NOT NULL);

-- Tabla Asignatura CREATE TABLE Asignatura (id INT PRIMARY KEY AUTO_INCREMENT, nombre VARCHAR(100) NOT NULL, creditos FLOAT NOT NULL, tipo ENUM('Obligatoria', 'Optativa', 'Troncal') NOT NULL, curso TINYINT NOT NULL, id_profesor INT, id_grado INT NOT NULL, FOREIGN KEY (id_profesor) REFERENCES Profesor(id_profesor) ON DELETE SET NULL, FOREIGN KEY (id_grado) REFERENCES Grado(id) ON DELETE CASCADE);

-- Tabla Alumno_se_matricula_asignatura CREATE TABLE Alumno_se_matricula_asignatura (id INT PRIMARY KEY AUTO_INCREMENT, id_alumno INT NOT NULL, id_asignatura INT NOT NULL, id_curso_escolar INT NOT NULL, FOREIGN KEY (id_alumno) REFERENCES Alumno(id_alumno) ON DELETE CASCADE, FOREIGN KEY (id_asignatura) REFERENCES Asignatura(id) ON DELETE CASCADE, FOREIGN KEY (id_curso_escolar) REFERENCES Curso_escolar(id) ON DELETE CASCADE);

Script Datos:

-- Insertar datos en la tabla Departamento

```
INSERT INTO Departamento (nombre) VALUES ('Matemáticas'), ('Física'), ('Informática');
```

-- Insertar datos en la tabla Persona

```
INSERT INTO Persona (dni, nombre, apellido1, apellido2, ciudad_nacimiento, ciudad_residencia, fecha_nacimiento, sexo, tipo) VALUES ('12345678A', 'Ana', 'López', 'Pérez', 'Madrid', 'Madrid', '1995-02-10', 'M', 'Alumno'), ('87654321B', 'Luis', 'Gómez', 'Sánchez', 'Sevilla', 'Sevilla', '1993-06-20', 'M', 'Profesor'), ('45678912C', 'María', 'Pérez', 'García', 'Valencia', 'Valencia', '1990-11-15', 'F', 'Profesor');
```

-- Insertar datos en la tabla Profesor INSERT INTO Profesor (id_persona, id_departamento) VALUES (2, 1), -- Luis Gómez, Profesor de Matemáticas (3, 2); -- María Pérez, Profesora de Física



-- Insertar datos en la tabla Alumno

```
INSERT INTO Alumno (id_persona) VALUES (1); -- Ana López, Alumna
```

-- Insertar datos en la tabla Curso_escolar

```
INSERT INTO Curso_escolar (anyo_inicio, anyo_fin) VALUES (2023, 2024);
```

-- Insertar datos en la tabla Grado

```
INSERT INTO Grado (nombre) VALUES ('Grado en Matemáticas'), ('Grado en Física'), ('Grado en Informática');
```

-- Insertar datos en la tabla Asignatura

```
INSERT INTO Asignatura (nombre, creditos, tipo, curso, id_profesor, id_grado) VALUES ('Cálculo I', 6, 'Obligatoria', 1, 1, 1), -- Asignatura de Cálculo en el Grado en Matemáticas ('Física I', 6, 'Obligatoria', 1, 2, 2), -- Asignatura de Física I en el Grado en Física ('Programación', 6, 'Obligatoria', 1,
```

Testigos:

Results

[Copy as Markdown](#)

Query #9 Execution time: 0.35ms

id	dni	nombre	apellido1	apellido2	ciudad_nacimiento	ciudad_residencia	fecha_nacimiento	sexo	tipo
1	12345678A	Ana	López	Martínez	Madrid	Barcelona	1995-06-15	H	Profesor
2	87654321B	Juan	Gómez	Pérez	Valencia	Madrid	1997-08-20	M	Alumno

Query #10 Execution time: 0.13ms

id_alumno	id_persona
1	2

Query #11 Execution time: 0.13ms

Query #11 Execution time: 0.13ms

id	nombre	creditos	tipo	curso	id_profesor	id_grado
1	Algoritmos	6	Obligatoria	1	1	1
2	Cálculo	5	Optativa	1	1	1

Query #12 Execution time: 0.1ms

id	id_alumno	id_asignatura	id_curso_escolar
1	1	1	1
2	1	2	1

Schema SQL

```
1 -- Tabla Departamento
2 CREATE TABLE Departamento (
3   id INT PRIMARY KEY AUTO_INCREMENT,
4   nombre VARCHAR(100) NOT NULL
5 );
6
7 -- Tabla Persona
8 CREATE TABLE Persona (
9   id INT PRIMARY KEY AUTO_INCREMENT,
10  dni VARCHAR(9) UNIQUE NOT NULL,
11  nombre VARCHAR(25) NOT NULL,
12  apellido1 VARCHAR(50) NOT NULL,
13  apellido2 VARCHAR(50),
14  ciudad_nacimiento VARCHAR(50)
```

Query SQL

```
17 (1, 2, 1) -- El alumno con id_alumno = 1 (Juan Gómez) se matricula en Cálculo en
18 el curso 2023-2024
19 -- Verificar los datos en la tabla Persona
20 SELECT * FROM Persona;
21
22
23 -- Verificar los datos en la tabla Alumno
24 SELECT * FROM Alumno;
25
26 -- Verificar los datos en la tabla Asignatura
27 SELECT * FROM Asignatura;
28
29 -- Verificar los datos en la tabla Alumno_se_matricula_asignatura
30 SELECT * FROM Alumno_se_matricula_asignatura;
31
```



ETAPA 3: PROYECTO INTEGRADOR

VI. Generación de consultas, procedimientos almacenados y vistas

4.1 Generación de consultas

- Retoma las base de datos creada en la etapa previa del proyecto y realiza las siguientes consultas utiliza tu ambiente de trabajo Workbench:

- Listado con nombre y apellidos de todos los alumnos.

SELECT

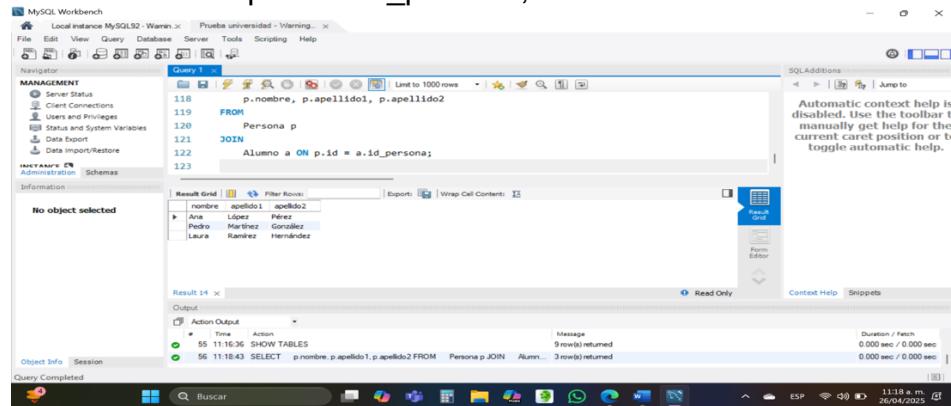
```
p.nombre, p.apellido1, p.apellido2
```

FROM

```
Persona p
```

JOIN

```
Alumno a ON p.id = a.id_persona;
```



The screenshot shows the MySQL Workbench interface with a query editor window titled 'Query 1'. The query is:

```

118   p.nombre, p.apellido1, p.apellido2
119   FROM
120     Persona p
121   JOIN
122     Alumno a ON p.id = a.id_persona;
123

```

The results grid displays three rows of data:

nombre	apellido1	apellido2
Alicia	Pérez	Ortega
Pedro	Martínez	González
Laura	Ramírez	Hernández

Below the results, the 'Output' pane shows the execution log:

```

55 11:16:36 SHOW TABLES
Message: 9 rows(s) returned
Duration / Fetch: 0.000 sec / 0.000 sec

56 11:18:43 SELECT p.nombre, p.apellido1, p.apellido2 FROM Persona p JOIN Alum... 3 row(s) returned
Message: Duration / Fetch: 0.000 sec / 0.000 sec

```

- Listado de los profesores junto con el nombre del departamento al que están vinculados. El listado debe devolver cuatro columnas, nombre, primer apellido, segundo apellido y nombre del departamento.

SELECT

```

p.nombre,
p.apellido1,
p.apellido2,
d.nombre AS nombre_departamento

```

FROM

```
Persona p
```

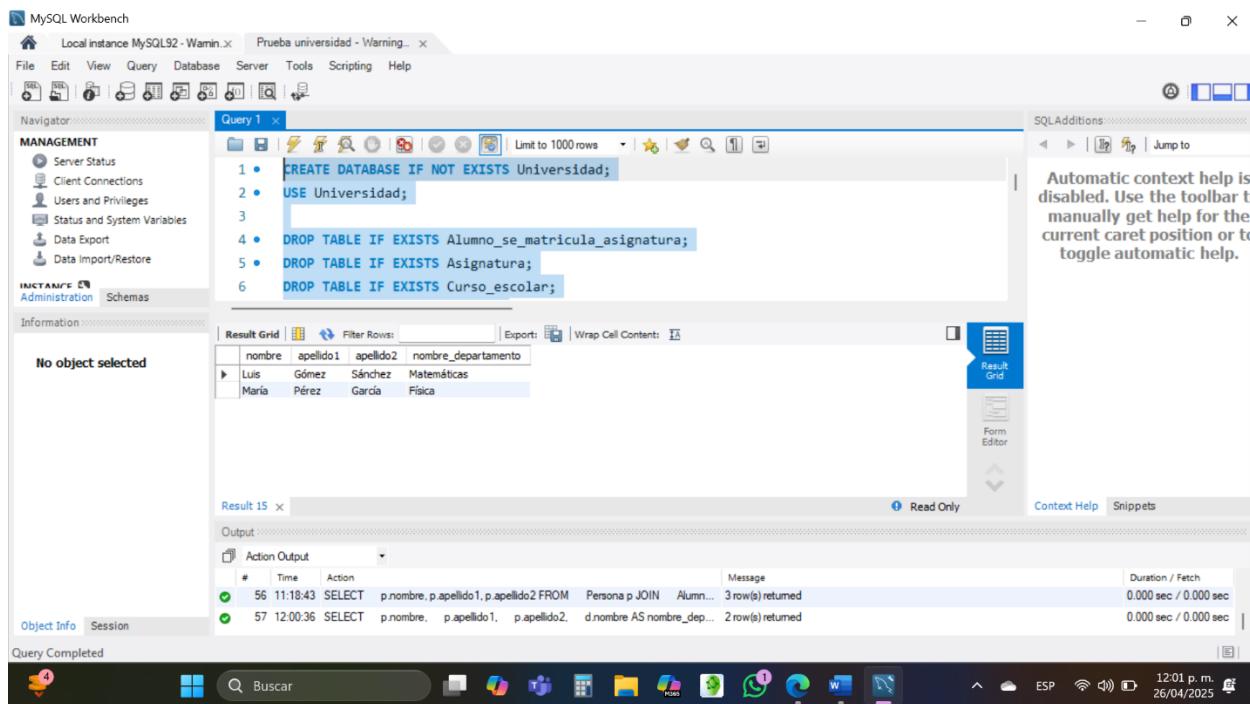
JOIN

```
Profesor prof ON p.id = prof.id_persona
```

LEFT JOIN

```
Departamento d ON prof.id_departamento = d.id;
```





The screenshot shows the MySQL Workbench interface. In the top-left, the title bar says "MySQL Workbench" and "Local instance MySQL82 - Wamin.x Prueba universidad - Warning...". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, Help. The left sidebar has sections for MANAGEMENT (Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, Data Import/Restore) and INSTANCE Administration (Schemas). The main area has a "Query 1" tab open with the following SQL script:

```

1 • CREATE DATABASE IF NOT EXISTS Universidad;
2 • USE Universidad;
3
4 • DROP TABLE IF EXISTS Alumno_se_matricula_asignatura;
5 • DROP TABLE IF EXISTS Asignatura;
6 • DROP TABLE IF EXISTS Curso_escolar;

```

Below the script, the "Result Grid" shows a table with three rows:

nombre	apellido1	apellido2	nombre_departamento
Luis	Gómez	Sánchez	Matemáticas
Maria	Pérez	García	Física

The "Output" pane at the bottom shows two recent actions:

- # 56 11:18:43 SELECT p.nombre, p.apellido1, p.apellido2 FROM Persona p JOIN Alumn... 3 row(s) returned Duration / Fetch 0.000 sec / 0.000 sec
- # 57 12:00:36 SELECT p.nombre, p.apellido1, p.apellido2, d.nombre AS nombre_dep... 2 row(s) returned Duration / Fetch 0.000 sec / 0.000 sec

c) Listado con los profesores que no imparten ninguna asignatura. El listado debe devolver Nombre, apellido y sexo.

```

SELECT
  p.nombre,
  p.apellido1,
  p.sexo
FROM Profesor prof
INNER JOIN Persona p ON prof.id_persona = p.id
LEFT JOIN Asignatura a ON prof.id_profesor = a.id_profesor
WHERE a.id IS NULL;

```

nombre	apellido1	sexo
Carlos	Ramírez	M
Lucía	Fernández	M

d) Devuelve el número total de alumnos.

```

SELECT COUNT(*) AS Total_Alumnos
FROM Alumno;

```



Total_Alumnos
3

e) Listado de todas las asignaturas de un alumno en particular.

```
CREATE VIEW Vista_Asignaturas_Alumno AS
SELECT
  p.id AS id_persona,
  CONCAT(p.nombre, ' ', p.apellido1, ' ', IFNULL(p.apellido2, '')) AS nombre_completo,
  a.nombre AS asignatura,
  a.creditos,
  a.tipo,
  a.curso,
  g.nombre AS grado,
  ce.anho_inicio,
  ce.anho_fin
FROM Persona p
JOIN Alumno al ON p.id = al.id_persona
JOIN Alumno_se_matricula_asignatura ama ON al.id_alumno = ama.id_alumno
JOIN Asignatura a ON ama.id_asignatura = a.id
JOIN Grado g ON a.id_grado = g.id
JOIN Curso_escolar ce ON ama.id_curso_escolar = ce.id;
```

```
SELECT * FROM Vista_Asignaturas_Alumno
WHERE nombre_completo LIKE 'Ana López Pérez%';
```

Result Grid  Filter Rows:   Search Export: 

id_persona	nombre_completo	asignatura	creditos	tipo	curso	grado	anho_inicio	anho_fin
10	Ana López Pérez	Cálculo I	6	Obligatoria	1	Grado en Matemáticas	2023	2024
10	Ana López Pérez	Programación	6	Obligatoria	1	Grado en Informática	2023	2024

f) Listado de asignaturas con el nombre y apellido del profesor que las imparte.

```
SELECT
  a.nombre AS Asignatura,
  p.nombre AS Profesor_Nombre,
```



```
p.apellido1 AS Profesor_Apellido
FROM Asignatura a
JOIN Profesor prof ON a.id_profesor = prof.id_profesor
JOIN Persona p ON prof.id_persona = p.id
ORDER BY a.nombre;
```

Asignatura	Profesor_Nombre	Profesor_Apellido
Cálculo I	Luis	Gómez
Física I	Maria	Pérez

g) Listado de alumnos, nombre y apellido de una asignatura en particular.

```
CREATE VIEW Vista_Alumnos_por_Asignatura AS
SELECT
  p.nombre AS nombre_alumno,
  p.apellido1,
  p.apellido2,
  a.nombre AS asignatura
FROM Alumno_se_matricula_asignatura ama
JOIN Alumno al ON ama.id_alumno = al.id_alumno
JOIN Persona p ON al.id_persona = p.id
JOIN Asignatura a ON ama.id_asignatura = a.id;

SELECT * FROM Vista_Alumnos_por_Asignatura
WHERE asignatura = 'Cálculo I';
```

nombre_alumno	apellido1	apellido2	asignatura
Ana	López	Pérez	Cálculo I
Pedro	Martínez	González	Cálculo I
Laura	Ramírez	Hernández	Cálculo I

- Ejecuta cada una de las consultas y toma evidencia del código ejecutado y los resultados obtenidos

4.2 Generación de procedimientos almacenados



- Retoma las consultas e) y g) y realízalas mediante un procedimiento almacenado, ingresando el alumno y asignatura como parámetros de entrada respectivamente.
- Toma evidencia de la creación del procedimiento, así como, de su invocación y resultados obtenidos.

DELIMITER //

```
CREATE PROCEDURE sp_Alumnos_por_Asignatura(IN asignatura_nombre
VARCHAR(100))
BEGIN
SELECT
  p.nombre AS nombre_alumno,
  p.apellido1,
  p.apellido2,
  a.nombre AS asignatura
FROM Alumno_se_matricula_asignatura ama
JOIN Alumno al ON ama.id_alumno = al.id_alumno
JOIN Persona p ON al.id_persona = p.id
JOIN Asignatura a ON ama.id_asignatura = a.id
WHERE a.nombre = asignatura_nombre;
END //
```

DELIMITER ;

CALL sp_Alumnos_por_Asignatura('Cálculo I');

Result Grid					 Filter Rows:	 Search	 Export:
	nombre_alumno	apellido1	apellido2	asignatura			
	Ana	López	Pérez	Cálculo I			
	Pedro	Martínez	González	Cálculo I			
	Laura	Ramírez	Hernández	Cálculo I			

DELIMITER //

```
CREATE PROCEDURE sp_Asignaturas_Alumno(IN alumno_nombre_completo
VARCHAR(100))
BEGIN
SELECT
  p.id AS id_persona,
  CONCAT(p.nombre, ' ', p.apellido1, ' ', IFNULL(p.apellido2, '')) AS
  nombre_completo,
  a.nombre AS asignatura,
  a.creditos,
```



```

a.tipo,
a.curso,
g.nombre AS grado,
ce.anexo_inicio,
ce.anexo_fin
FROM Persona p
JOIN Alumno al ON p.id = al.id_persona
JOIN Alumno_se_matricula_asignatura ama ON al.id_alumno = ama.id_alumno
JOIN Asignatura a ON ama.id_asignatura = a.id
JOIN Grado g ON a.id_grado = g.id
JOIN Curso_escolar ce ON ama.id_curso_escolar = ce.id
WHERE CONCAT(p.nombre, ' ', p.apellido1, ' ', IFNULL(p.apellido2, '')) LIKE
alumno_nombre_completo;
END //
  
```

DELIMITER ;

CALL sp_Asignaturas_Alumno('Ana López Pérez');

Result Grid										 Filter Rows:	 Search	 Export:
	id_persona	nombre_completo	asignatura	creditos	tipo	curso	grado	anexo_inicio	anexo_fin			
	10	Ana López Pérez	Cálculo I	6	Obligatoria	1	Grado en Matemáticas	2023	2024			
	10	Ana López Pérez	Programación	6	Obligatoria	1	Grado en Informática	2023	2024			

4.3 Generación de vistas

- Retoma la consulta **a)** y crea una vista que arroje el mismo resultado.
- Toma evidencia de la creación de la vista así como de su consulta y resultados obtenidos.

CREATE VIEW Vista_Alumnos AS

SELECT

```

p.nombre,
p.apellido1,
p.apellido2
  
```

FROM

Persona p

JOIN

Alumno a ON p.id = a.id_persona;



MySQL Workbench

Local instance MySQL52 - Wamin.x Prueba universidad - Warning... x

File Edit View Query Database Server Tools Scripting Help

Navigator: MANAGEMENT Client Connections Users and Privileges Status and System Variables Data Export Data Import/Restore INSTANCE Administration Schemas Information

No object selected

Query 1 x

```

132 p.apellido2
133 FROM
134 Persona p
135 JOIN
136 Alumno a ON p.id = a.id_persona;
137
138 • SHOW TABLES;
139
  
```

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid | Filter Rows | Export | Wrap Cell Content | Result Grid | Read Only | Context Help | Snippets

nombre	apellido1	apellido2
Ana	López	Pérez
Pedro	Martínez	González
Laura	Ramírez	Hernández

Output: Action Output

#	Time	Action	Message	Duration / Fetch
30	12:10:21	CREATE VIEW Vista_Alumnos AS SELECT p.nombre, p.apellido1, p.apellido2	0 row(s) affected	0.000 sec
31	12:10:21	SHOW TABLES	9 row(s) returned	0.016 sec / 0.000 sec

Object Info Session

Query Completed

Buscar

12:11 p.m. 26/04/2025

Conclusión

Miranda Velázquez Mariles:

En este proyecto se diseñó y creó un sistema de base de datos relacional para una universidad estructurando cuidadosamente la información de departamentos, personas, profesores, alumnos, grados, asignaturas y cursos escolares; a través de esta base de datos se logró establecer relaciones claras y consistentes entre los distintos elementos que conforman el entorno académico, garantizando la integridad referencial mediante el uso de claves foráneas y restricciones adecuadas.

El modelo propuesto no solo permite almacenar de forma eficiente la información actual, sino que también está preparado para crecer permitiendo agregar nuevos grados, asignaturas, departamentos, alumnos y profesores de manera sencilla. En resumen, el trabajo realizado demuestra cómo un diseño cuidadoso de base de datos puede apoyar eficazmente las operaciones de una universidad mejorando el acceso a la información, la organización interna y la toma de decisiones basadas en datos confiables.



Alfonso Roberto Gómez Rosales:

A través de este ejercicio se aprendió a manejar de manera integral una base de datos relacional, comprendiendo la importancia de diseñar tablas correctamente, establecer relaciones mediante claves foráneas y asegurar la integridad referencial. Además, se profundizó en el uso de consultas SQL para extraer información precisa, como el conteo de registros, la obtención de listados específicos y la combinación de datos de varias tablas mediante sentencias JOIN. Se reforzó también el entendimiento sobre cómo interpretar y estructurar resultados en un formato legible de consola, simulando un entorno real de consulta de datos. Este proceso permitió ver claramente la utilidad de normalizar la información y trabajar con datos consistentes. En conjunto, la práctica fortaleció habilidades tanto técnicas como analíticas, necesarias para resolver problemas reales en bases de datos. Además, se evidenció cómo una buena organización y planeación del esquema de la base de datos facilita consultas más eficientes y comprensibles.

Jose Emiliano Jauregui Guzman:

A lo largo del desarrollo de esta última etapa del proyecto integrador, he logrado comprender de manera sólida los conocimientos adquiridos en la materia de bases de datos, específicamente en la generación de consultas SQL, la creación de vistas y el diseño e implementación de procedimientos almacenados. La práctica constante y la aplicación directa de estos conceptos en un modelo relacional completo me permitieron comprender no solo la sintaxis y funcionalidad de cada instrucción, sino también su utilidad real para optimizar la gestión de datos y mejorar la eficiencia en el acceso a la información. Me siento satisfecho con el aprendizaje obtenido, ya que ahora soy capaz de estructurar bases de datos relacionales con una visión más integral, implementar soluciones dinámicas mediante procedimientos, y construir vistas que faciliten la interpretación de la información para distintos tipos de usuarios. Esta experiencia me ha dado mayor confianza para las futuras materias de mi carrera y siempre poder perfeccionar lo aprendido ya que fue una materia muy completa que me agrado demasiado.



Referencias APA

- Hernández, J. J. S. (s. f.). Ejercicios. Realización de consultas SQL. <https://josejuansanchez.org/bd/ejercicios-consultas-sql/index.html>
- Sánchez, J. (2020). Ejercicios. Realización de consultas SQL [Sitio Web] Recuperado de sql/index.html#datos-1 <https://josejuansanchez.org/bd/ejercicios-consultas>
- López, M., Gallegos, F. (2017). *Programación de bases de datos relacionales*. <https://www.detodoprogramacion.org/2022/02/programacion-de-bases-de-datos-relacionales.html>
- Yizreel 09 de Febrero de 2015). MySql. Creación de Base de Datos y Tablas. (Workbench Código). Recuperado de https://www.youtube.com/watch?v=plfGB_U1auQ
- Pulido, E., Escobar, O., Núñez, J. (2019). Base de Datos. <https://learn-us-east-1-prod-fleet02-xythos.content.blackboardcdn.com/5fdc0d8108394/31747795?X-Bla>
- López, M., Gallegos, F. (2017). *Programación de bases de datos relacionales*. Programación de bases de datos relacionales. <https://www.detodoprogramacion.org/2022/02/programacion-de-bases-de-datos-relacionales.html>
- Códigos de Programación. (Productor). (09 de Diciembre de 2016). Curso MySQL 12: Vistas, Procedimientos y Triggers MySql. Recuperado de <https://www.youtube.com/watch?v=peFOijUfGO4>

