Consultas Modelo Relacional



Guillermo de Carlos Muñoz

23/24 - IABD - Big Data Aplicado

# 

[**Paso 1: Crear la BBDD, sin tablas. 3**](#_lstyi9pt1cxr)

[**Paso 2: Generar las tablas de la BBDD 5**](#_7xshqhsip0wo)

[Crear Base de Datos 5](#_5p9fvc1iq1p9)

[Crear Tablas 6](#_dx7cq5i7vccb)

[**Paso 3: Insertar/modificar/borrar datos en las tablas 9**](#_4d8g2lpr0lna)

[Insertar 10 registros por tabla 9](#_64yhm9glqmz2)

[Tabla departamento 9](#_htfcvq63k6d)

[Tabla estudiantes 9](#_wk5a5dkgaqoe)

[Tabla Cursos 10](#_7sq6bizudub)

[Tabla Profesores 10](#_janrp6k2pmdu)

[Tabla Estudiantes\_Cursos 11](#_iiwex3lfavdz)

[Tabla Profesores\_Cursos 11](#_c4li5k5uln9e)

[Cambiar valor de un atributo 12](#_4qnpaih0dowb)

[Borrar un registro 12](#_7fl76xg9yovc)

[**Paso 4: Generar consultas 13**](#_hysr78ufz8x)

[1. Muestra todos los estudiantes y sus nombres completos. 13](#_2chfu1nhaor3)

[2. Lista los cursos ofrecidos por un departamento específico, por ejemplo, el Departamento de Matemáticas. 14](#_9usfwcxbrpr3)

[3. Obtén todos los profesores asociados a un departamento en particular, como el Departamento de Informática. 15](#_vbjcnl8tbxqm)

[4. Muestra los estudiantes matriculados en un curso específico, por ejemplo, el curso de Matemáticas Avanzadas. 16](#_wgqiqj49eom4)

[5. Encuentra los cursos que no tienen ningún estudiante matriculado. 17](#_owr89rbbzvmk)

[6. Identifica a los profesores que imparten más de un curso. 18](#_k0x8hpfns410)

[7. Muestra todos los estudiantes que nacieron en un mes determinado, como en enero. 19](#_pn4rp2mtohox)

[8. Encuentra los cursos con el mayor número de créditos. 20](#_tw5t4aie5nd4)

[9. Encuentra a los estudiantes que no se han matriculado en ningún curso 21](#_e601xzxf8sff)

[10. Identifica los departamentos que no tienen ningún curso asociado. 22](#_pwtl9essaro5)

# 

# Paso 1: Crear la BBDD, sin tablas.

Una vez borrados los containers y volúmenes anteriores de MySQL, vamos a empezar todo desde cero.

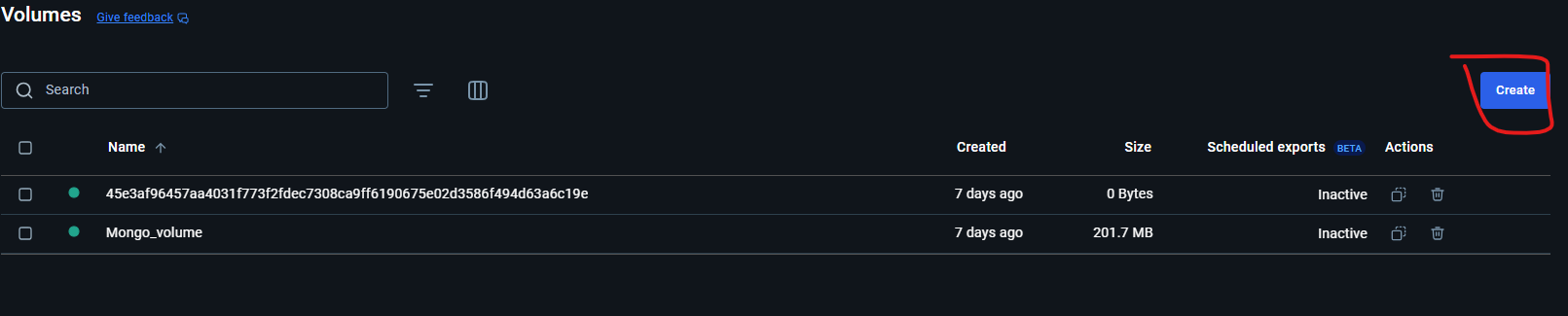
Para ello primero iremos a la imagen oficial de MySQL en [docker hub](https://hub.docker.com/_/mysql).

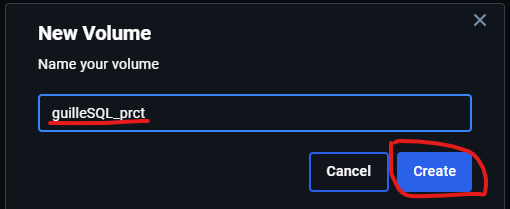


$ docker run --name some-mysql -v /my/own/datadir:/var/lib/mysql -e MYSQL\_ROOT\_PASSWORD=my-secret-pw -d mysql:tag

Este comando lo usaremos para lanzar un nuevo container con un nombre del container , una dirección de un volumen que tengamos ya creado, una contraseña a elegir y la versión de imagen de mysql.

Empezaremos por crear el volumen, para tener un volumen al que asociar el container:





Le he puesto este nombre haciendo referencia a que es mi volumen de sql de una práctica.



Por el momento nos saldrá sin uso.

Una vez tengamos el volumen ya podremos crear el contenedor y asociarlo:

docker run --name guilleSQL -v guilleSQL\_prct:/var/lib/mysql -e MYSQL\_ROOT\_PASSWORD=my-secret-pw -d mysql:latest



PS C:\Users\Vespertino> docker run --name guilleSQL -v guilleSQL\_prct:/var/lib/mysql -e MYSQL\_ROOT\_PASSWORD=my-secret-pw -d mysql:latest

d502afb2978431989642c287a288c0e562bfc11a246a71ff2f389496049dda28

Esta serie alfanumérica “random”, es el hash/id del contenedor, con el o con el nombre podremos manipular el container.

Para acceder al container:

PS C:\Users\Vespertino> docker exec -it guilleSQL /bin/bash

Aquí estamos indicando que, ejecutamos el container de manera iterativa, el nombre del container que queremos acceder y /bin/bash para acceder a la consola, de manera iterativa ya que no queremos sólo hacer un comando y que se deje de ejecutar el container.



Nos pedirá acceso



Accedemos con el usuario root y la contraseña la ponemos solo con la -p para que no salga directamente en el comando y venga un gracioso a robarla.

****Ya dentro la command line se cambiará a algo de este estilo, indicando que estamos dentro de mysql de nuestro container.

# Paso 2: Generar las tablas de la BBDD

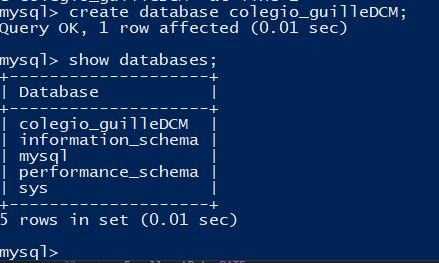
## Crear Base de Datos

Lo primero creamos la base de datos, donde crearemos las tablas, relaciones y meteremos los datos.

**create database colegio\_guilleDCM;**

Hacemos este comando siguiente para mostrar las bases de datos que existen en nuestro container para ver si esta la que acabamos de crear.

**show databases;**



Vemos como nuestra base de datos colegio\_guilleDCM esta.

Para usarla haremos el siguiente comando:

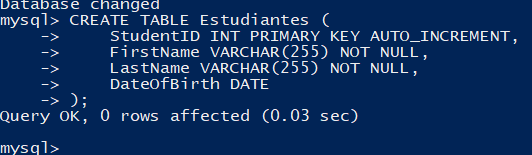


**use colegio\_guilleDCM;**

Vemos que nos avisa de que hemos cambiado a nuestra base de datos.

## Crear Tablas

**CREATE** **TABLE** Estudiantes ( StudentID **INT** PRIMARY **KEY** AUTO\_INCREMENT, FirstName **VARCHAR**(255) **NOT** NULL, LastName **VARCHAR**(255) **NOT** NULL, DateOfBirth **DATE** );

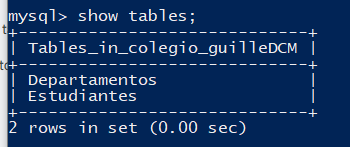


La siguiente tabla que crearemos será la de departamento ya que es la otra que tampoco depende de una relación propia en sus atributos.

**CREATE** **TABLE** Departamentos ( DepartmentID **INT** PRIMARY **KEY** AUTO\_INCREMENT, DepartmentName **VARCHAR**(255) **NOT** NULL );

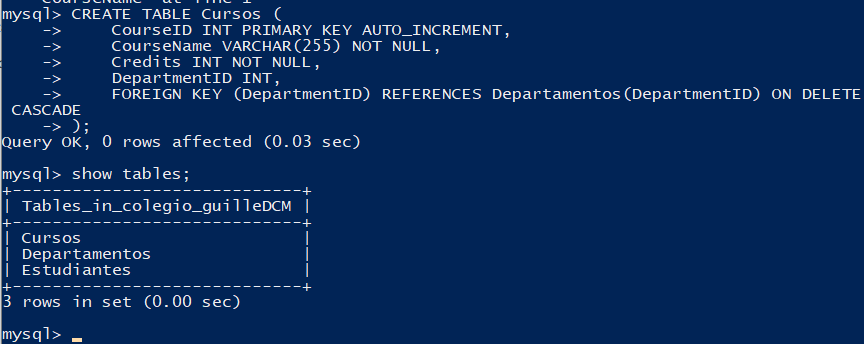


Con el **show** **tables**; podemos ir viendo las tablas que vamos creando.

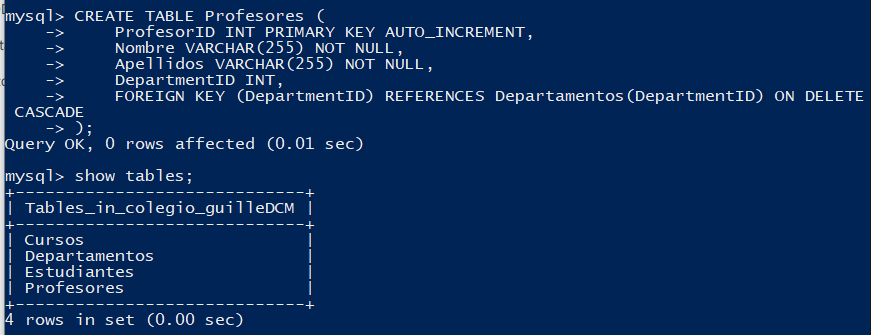


De momento todo bien

**CREATE** **TABLE** Cursos ( CourseID **INT** PRIMARY **KEY** AUTO\_INCREMENT, CourseName **VARCHAR**(255) **NOT** NULL, Credits **INT** **NOT** NULL, DepartmentID **INT**, FOREIGN **KEY** (DepartmentID) **REFERENCES** Departamentos(DepartmentID) **ON** **DELETE** **CASCADE** );



**CREATE** **TABLE** Profesores ( ProfesorID **INT** PRIMARY **KEY** AUTO\_INCREMENT, Nombre **VARCHAR**(255) **NOT** NULL, Apellidos **VARCHAR**(255) **NOT** NULL, DepartmentID **INT**, FOREIGN **KEY** (DepartmentID) **REFERENCES** Departamentos(DepartmentID)**ON** **DELETE** **CASCADE**);



Ya tenemos todas las tablas bases con las relaciones de 1 a N y N a 1, pero las de NaA,

necesitamos crear nuevas tablas.

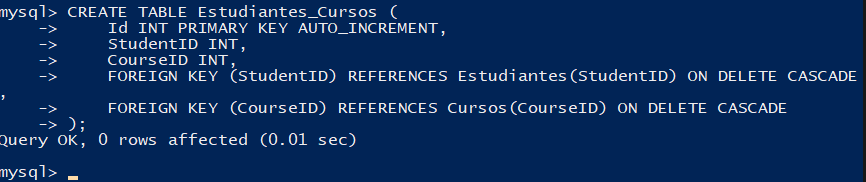
**Los estudiantes pueden matricularse en Cursos**

Un estudiante en muchos cursos y un curso tiene muchos estudiantes

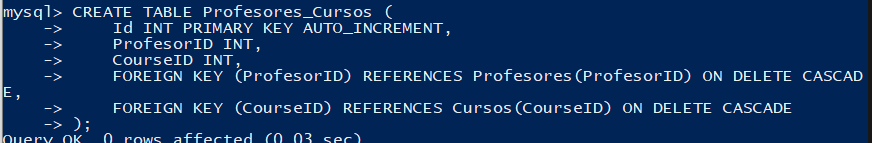
**Los profesores pueden impartir cursos.**

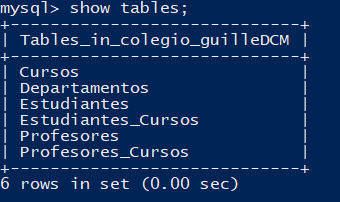
Un profe da en un curso y un curso tiene varios profesores

**CREATE** **TABLE** Estudiantes\_Cursos ( **Id** **INT** PRIMARY **KEY** AUTO\_INCREMENT, StudentID **INT**, CourseID **INT**, FOREIGN **KEY** (StudentID) **REFERENCES** Estudiantes(StudentID)**ON** **DELETE** **CASCADE**,FOREIGN **KEY** (CourseID) **REFERENCES** Cursos(CourseID)**ON** **DELETE** **CASCADE** );



**CREATE** **TABLE** Profesores\_Cursos ( **Id** **INT** PRIMARY **KEY** AUTO\_INCREMENT, ProfesorID **INT**, CourseID **INT**, FOREIGN **KEY** (ProfesorID) **REFERENCES** Profesores(ProfesorID)**ON** **DELETE** **CASCADE**,FOREIGN **KEY** (CourseID) **REFERENCES** Cursos(CourseID)**ON** **DELETE** **CASCADE** );





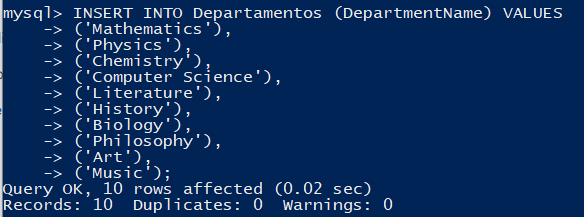
Ya tendríamos todas las tablas creadas.

# Paso 3: Insertar/modificar/borrar datos en las tablas

## Insertar 10 registros por tabla

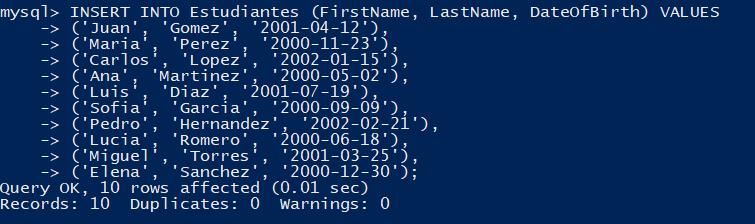
### Tabla departamento

**INSERT INTO** Departamentos(DepartmentName)**VALUES**('Mathematics'),('Physics'),('Chemistry'),('Computer Science'),('Literature'),('History'),('Biology'),('Philosophy'),('Art'),('Music');



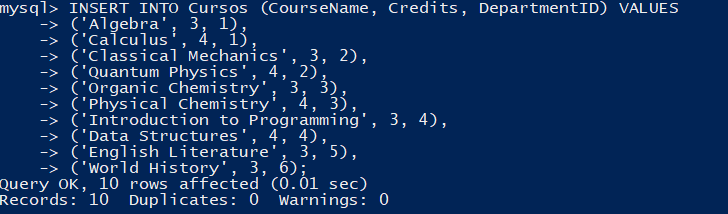
### Tabla estudiantes

**INSERT INTO** Estudiantes(FirstName, LastName, DateOfBirth)**VALUES**('Juan', 'Gomez', '2001-04-12'), ('Maria', 'Perez', '2000-11-23'), ('Carlos', 'Lopez', '2002-01-15'), ('Ana', 'Martinez', '2000-05-02'), ('Luis', 'Diaz', '2001-07-19'), ('Sofia', 'Garcia', '2000-09-09'), ('Pedro', 'Hernandez', '2002-02-21'), ('Lucia', 'Romero', '2000-06-18'), ('Miguel', 'Torres', '2001-03-25'), ('Elena', 'Sanchez', '2000-12-30');



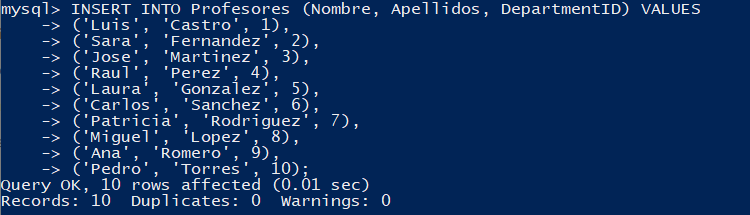
### Tabla Cursos

**INSERT INTO** Cursos(CourseName, Credits, DepartmentID)**VALUES**('Algebra', 3, 1), ('Calculus', 4, 1), ('Classical Mechanics', 3, 2), ('Quantum Physics', 4, 2), ('Organic Chemistry', 3, 3), ('Physical Chemistry', 4, 3), ('Introduction to Programming', 3, 4), ('Data Structures', 4, 4), ('English Literature', 3, 5), ('World History', 3, 6);



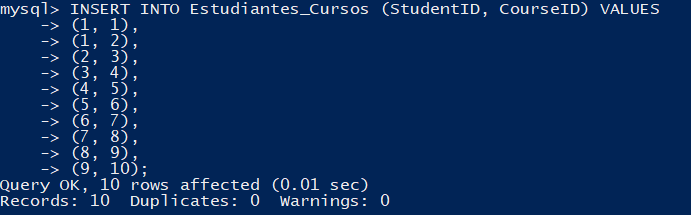
### Tabla Profesores

**INSERT INTO** Profesores (Nombre, Apellidos, DepartmentID)**VALUES**('Luis', 'Castro', 1), ('Sara', 'Fernandez', 2), ('Jose', 'Martinez', 3), ('Raul', 'Perez', 4), ('Laura', 'Gonzalez', 5), ('Carlos', 'Sanchez', 6), ('Patricia', 'Rodriguez', 7), ('Miguel', 'Lopez', 8), ('Ana', 'Romero', 9), ('Pedro', 'Torres', 10);



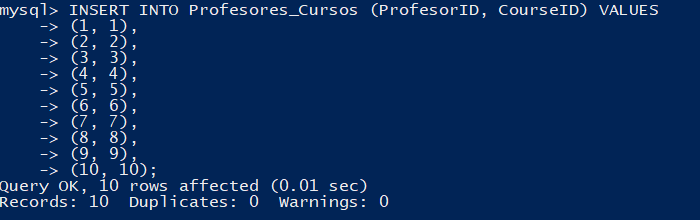
### Tabla Estudiantes\_Cursos

**INSERT INTO** Estudiantes\_Cursos (StudentID, CourseID) **VALUES**(1, 1), (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9), (9, 10 );



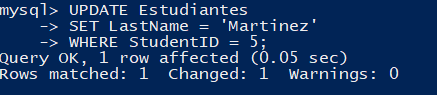
### Tabla Profesores\_Cursos

**INSERT INTO** Profesores\_Cursos (ProfesorID, CourseID)**VALUES**(1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9), (10, 10);



## Cambiar valor de un atributo

**UPDATE** Estudiantes **SET** LastName = 'Martinez' **WHERE** StudentID = 5;



## Borrar un registro

**DELETE FROM** Profesores **WHERE** ProfesorID = 7;



# Paso 4: Generar consultas

### 1. Muestra todos los estudiantes y sus nombres completos.

SELECT

StudentID,

CONCAT(FirstName, ' ', LastName) AS NombreCompleto

FROM

Estudiantes;



PD: La consola ha cambiado de color ya que esta ultima parte de la practica la estoy haciendo desde casa y mi ordenador personal.

### 

### 2. Lista los cursos ofrecidos por un departamento específico, por ejemplo, el Departamento de Matemáticas.

SELECT

Cursos.CourseID,

Cursos.CourseName,

Cursos.Credits

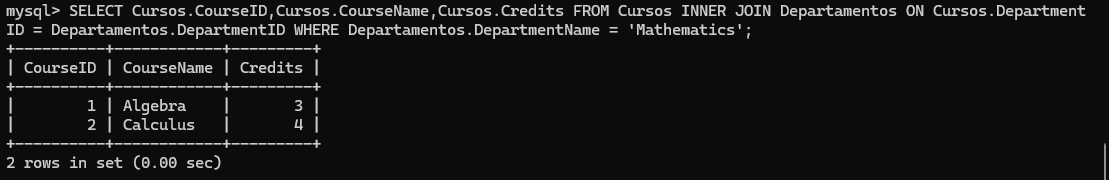
FROM

Cursos

INNER JOIN Departamentos ON Cursos.DepartmentID = Departamentos.DepartmentID

WHERE

Departamentos.DepartmentName = ‘ Mathematics ’;



### 

### 3. Obtén todos los profesores asociados a un departamento en particular, como el Departamento de Informática.

SELECT

Profesores.ProfesorID,

Profesores.Nombre,

Profesores.Apellidos

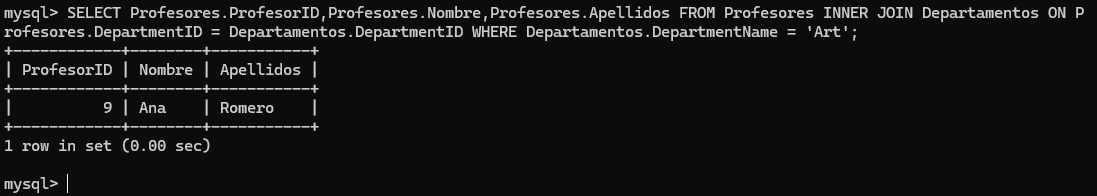
FROM

Profesores

INNER JOIN Departamentos ON Profesores.DepartmentID = Departamentos.DepartmentID

WHERE

Departamentos.DepartmentName = ‘ Art ’;



### 

### 4. Muestra los estudiantes matriculados en un curso específico, por ejemplo, el curso de Matemáticas Avanzadas.

SELECT

e.StudentID,

CONCAT(e.FirstName, ' ', e.LastName) AS NombreCompleto

FROM

Estudiantes e

INNER JOIN

Estudiantes\_Cursos ec ON e.StudentID = ec.StudentID

INNER JOIN

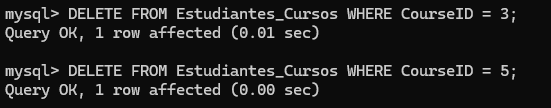
Cursos c ON ec.CourseID = c.CourseID

WHERE

c.CourseName = 'Algebra';

### 

### 5. Encuentra los cursos que no tienen ningún estudiante matriculado.

  
  
Para que salga algún resultado he borrado dos filas.

SELECT

c.CourseID,

c.CourseName

FROM

Cursos c

LEFT JOIN

Estudiantes\_Cursos ec ON c.CourseID = ec.CourseID

WHERE

ec.CourseID IS NULL;

### 

### 6. Identifica a los profesores que imparten más de un curso.

SELECT

p.Nombre,

COUNT(pc.CourseID) AS CantidadCursos

FROM

Profesores p

LEFT JOIN

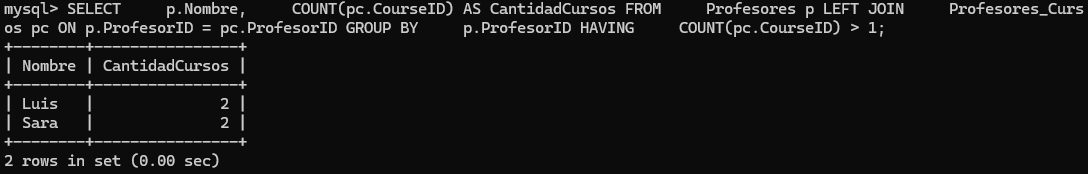
Profesores\_Cursos pc ON p.ProfesorID = pc.ProfesorID

GROUP BY

p.ProfesorID

HAVING

COUNT(pc.CourseID) > 1;



### 

### 7. Muestra todos los estudiantes que nacieron en un mes determinado, como en enero.

SELECT

FirstName,

DateOfBirth

FROM

Estudiantes

WHERE

MONTH(DateOfBirth) = 1;

### 

### 8. Encuentra los cursos con el mayor número de créditos.

SELECT

CourseID,

CourseName,

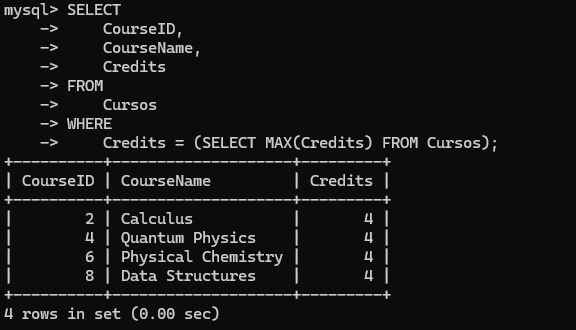
Credits

FROM

Cursos

WHERE

Credits = (SELECT MAX(Credits) FROM Cursos);



### 9. Encuentra a los estudiantes que no se han matriculado en ningún curso

SELECT

e.FirstName

FROM

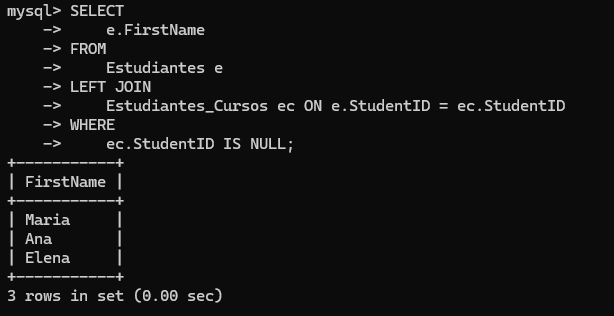
Estudiantes e

LEFT JOIN

Estudiantes\_Cursos ec ON e.StudentID = ec.StudentID

WHERE

ec.StudentID IS NULL;



### 10. Identifica los departamentos que no tienen ningún curso asociado.

SELECT

d.DepartmentID,

d.DepartmentName

FROM

Departamentos d

LEFT JOIN

Cursos c ON d.DepartmentID = c.DepartmentID

WHERE

c.CourseID IS NULL;

