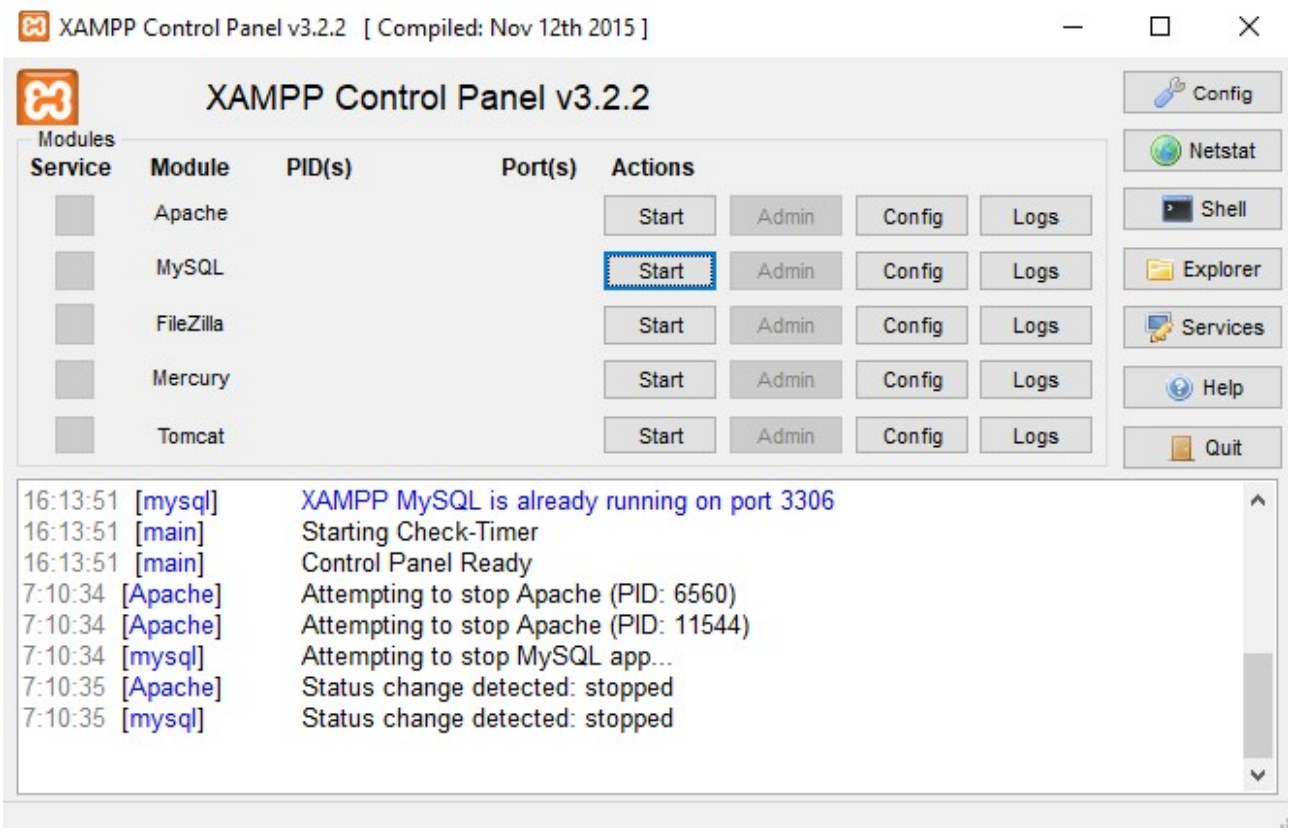
Paso 1: Instalación de MySQL

Podemos contar con varias opciones de mercado de servidores locales que ya traen incorporada la base de datos MySQL, una opción que ya trae configurada la base de datos y un servidor del tipo Apache es xampp. Las versiones de los lenguajes que utiliza se actualizan todo el tiempo y tenemos la posibilidad de elegir el sistema operativo. Yo en particular he descargado la versión para Windows del siguiente sitio:

https://www.apachefriends.org/es/index.html

Para instalarlo es muy fácil, simplemente lo descargamos y damos “next” hasta finalizar la instalación eligiendo en el proceso el directorio en el cual se va a instalar, el cual por defecto en Windows es el disco raíz. De esta forma nos aparece una vez finalizada la instalación un directorio llamado xampp en la carpeta raíz de nuestro equipo. Dentro de este directorio nos vamos a encontrar un sistema de carpetas y un ejecutable con el nombre de “xamppControl”. Si hacemos doble click sobre el mismo se nos va a desplegar la siguiente ventana:



En la misma podemos ver varios botones “Start” que son los que encienden los servicios. Dado que vamos a utilizar también una interfaz gráfica para manipular la base de datos de forma gráfica, iniciaremos los servicios de Apache y MySQL. Cada servicio funciona como un semáforo el cual si todo está bien debe presentar un color verde e indicar el puerto de trabajo. Por defecto el puerto del servidor es el 80 y el de MySQL el 3306.

Paso 2 – Instalamos conector para python - MySQL

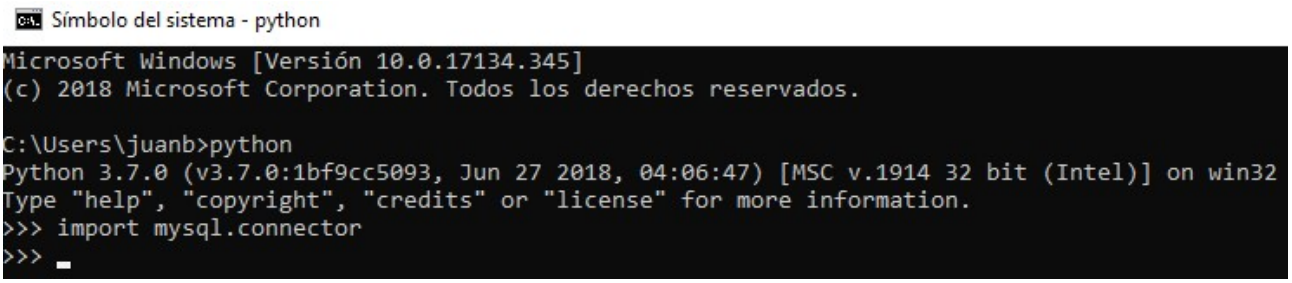
Ingresamos al cmd con permisos de administrador y ejecutamos:

```
pip install mysql-connector
```

Si todos está bien podemos ingresar al intérprete de python desde el cmd (escribiendo: python) y ejecutar:

```
import mysql.connector
```

No debería retornarnos ningún error como se muestra en la siguiente imagen.

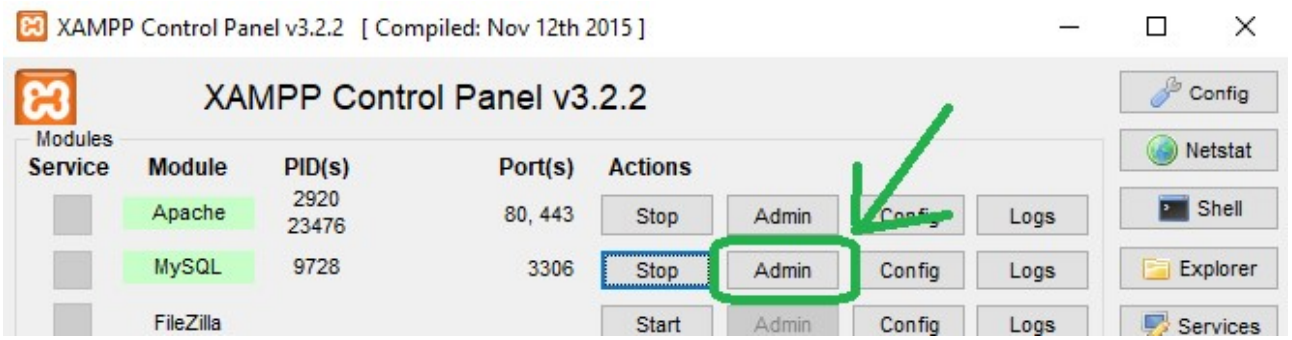


Paso 3 – Creamos una base de datos.

Nuestro siguiente paso es crear una base de datos con la cual interactuar, para ello contamos con dos opciones

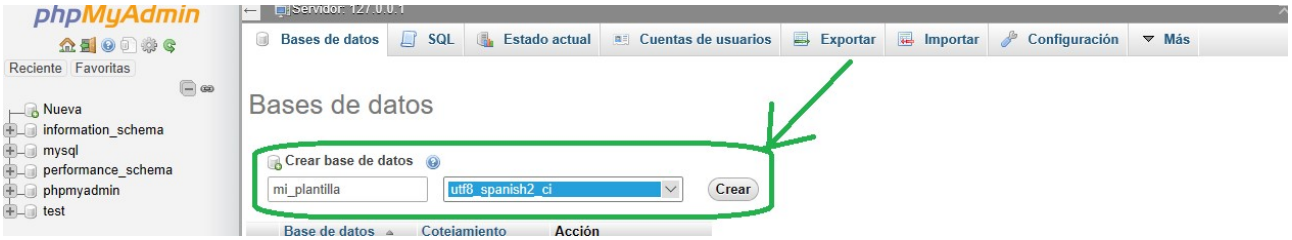
Opción 1 - desde phpmyadmin.

La primera y más fácil es presionar el botón “Admin”, el cual se encuentra en la línea del servicio MySQL para ingresar a la interface gráfica phpmyadmin. La aplicación se lanza en el explorador que tengamos configurado por defecto.

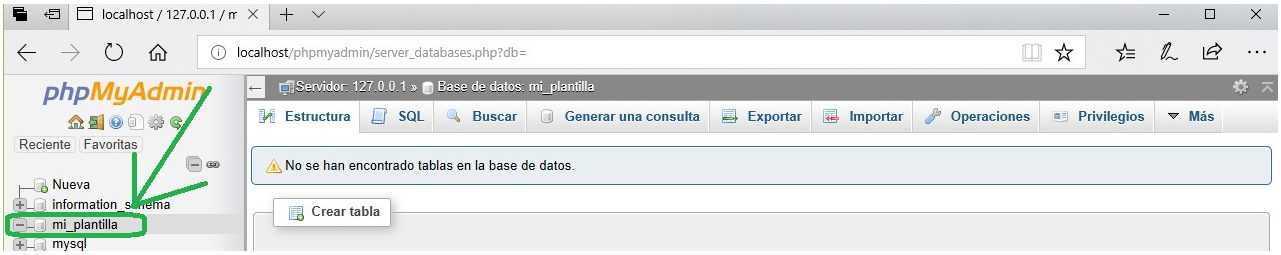


Una vez ahí vamos a “Bases de datos” en las solapas que se encuentran en la cabecera.

Solo nos resta en esta parte darle un nombre cualquiera (yo le he llamado “mi_plantilla”) y elegir el tipo de caracteres que utilizaremos en español (puede ser “utf8_spanish2_ci”). Finalmente presionamos en crear.



Nuestra base ya aparece listada en el menú de la izquierda.



Nota: Por defecto la conexión de la base de datos viene configurada como:

Servidor = localhost

usuario = root

password = “”

El usuario y el password puede ser modificado directamente desde la interfaz gráfica, sin embargo modificar el nombre del servidor requiere alterar algunas líneas en el archivo de configuración de apache.

Opción 2 – Desde Python

La otra opción con la cual contamos es desde la consola de python estando el servidor de la base de datos encendido, ejecutar el siguiente script:

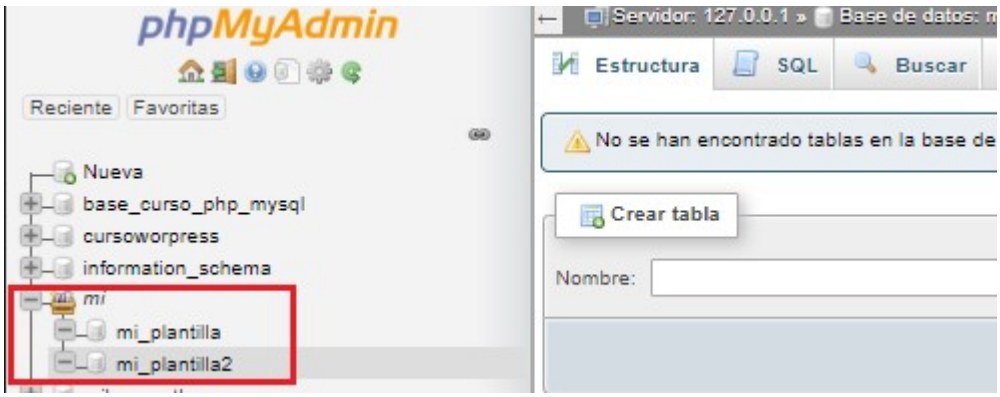
crearbd.py

```
import mysql.connector

mibase = mysql.connector.connect(
  host="localhost",
  user="root",
  passwd=""
)
micursor = mibase.cursor()

micursor.execute("CREATE DATABASE mi_plantilla2")
```

Aquí “mysql.connector” es un objeto que podemos utilizar para conectarnos a la base de datos y mediante notación de punto (agregando un punto al final y a

continuación el método que queremos aplicar) le aplicamos el método connect() el cual recibe los parámetros de servidor, usuario y password para poder abrir la conexión. Una vez abierta la conexión, podemos nuevamente con notación de punto asociarle el método execute(), con el cual le pasamos en lenguaje sql la instrucción para que cree la base de datos.

Luego de ejecutarlo podemos chequear desde la interfaz gráfica como se ha agregado al menú de la izquierda.

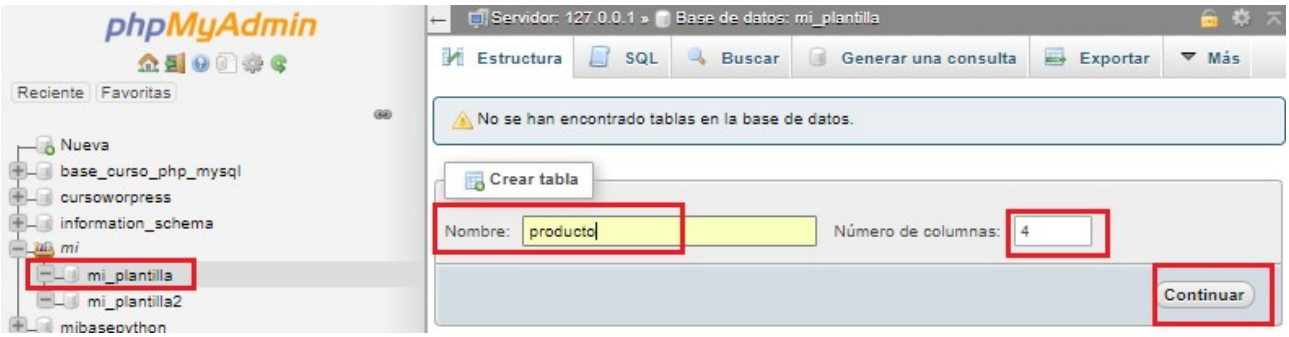


Paso 4 – Creamos una tabla

Las bases de datos relacionales poseen tablas para guardar los datos, y trabajar con ellas es similar a trabajar con hojas de Excel. Nuevamente podemos crear las tablas de forma gráfica o mediante código.

Opción 1 – Desde phpmyadmin

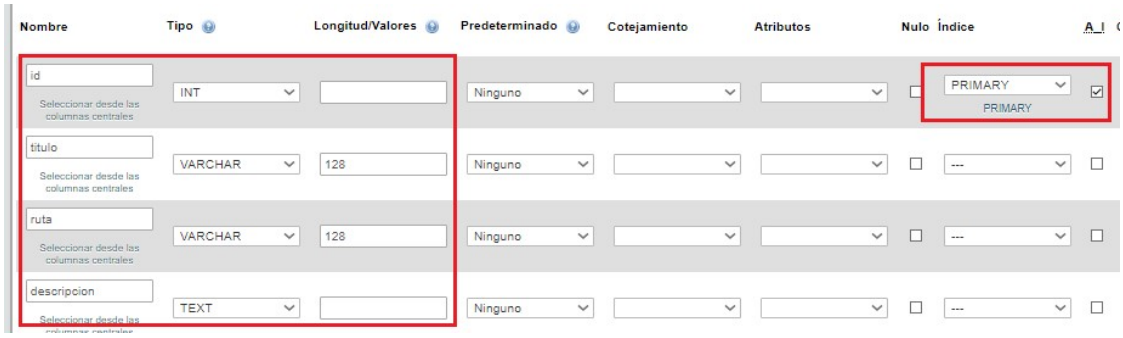
Para crear una tabla, nos paramos en la base que queremos adicionarla y le damos un nombre seleccionando además la cantidad de columnas que va a poseer, luego presionamos en “Continuar”.



En este caso he agregado una tabla de cuatro columnas:

- **id**: Campo de enteros (int) autoincremental y clave primaria (campo clave)
- **titulo**: Campo de tipo varchar (strings) de 128 caracteres.
- **ruta**: Campo de tipo varchar (strings) de 128 caracteres.
- **descripcion**: Campo de texto, ocupa más memoria.

Nota: no utilizar acentos ni caracteres en español para los campos.



Opción 2 – Desde el código

Esta opción requiere un poco más de conocimiento del lenguaje de la base de datos, sin embargo salvo que hemos indicado como parámetro de la conexión la base de datos dentro de la cual se va a agregar la tabla, la estructura es la misma que ya hemos utilizado al crear la base.

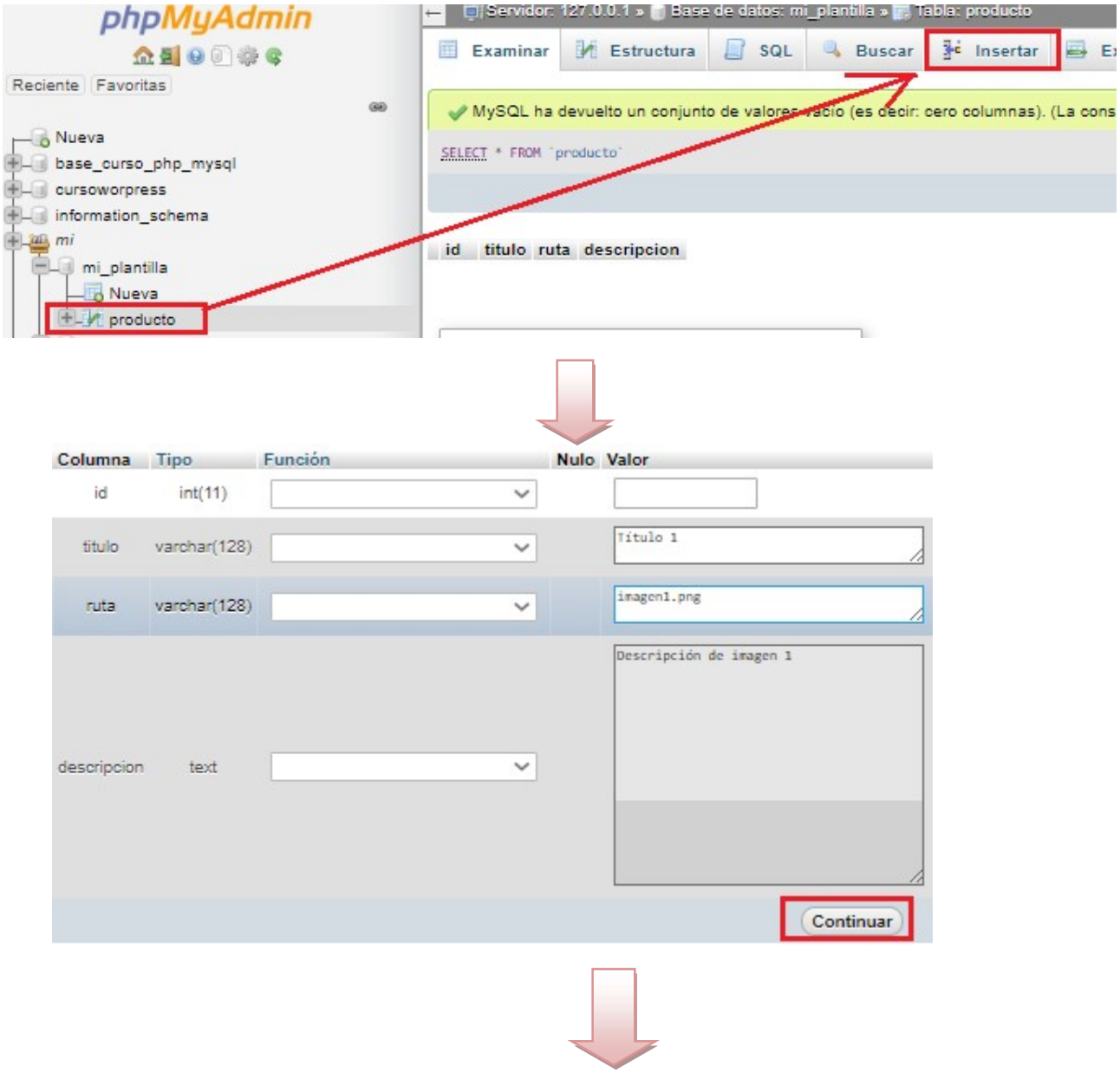
creartabla.py

```
import mysql.connector

mibase = mysql.connector.connect(
  host="localhost",
  user="root",
  passwd="",
  database="mi_plantilla2"
)
micursor = mibase.cursor()

micursor.execute("CREATE TABLE producto( id int(11) NOT NULL PRIMARY KEY AUTO_INCREMENT, titulo VARCHAR(128) COLLATE utf8_spanish2_ci NOT NULL, ruta varchar(128) COLLATE utf8_spanish2_ci NOT NULL, descripcion text COLLATE utf8_spanish2_ci NOT NULL )")
```

Paso 5 - Insertamos registros

Opción 1 – Desde phpmyadmin

Nuevamente la opción 1 es la más fácil de implementar para insertar registros, solamente tendríamos que pararnos en la tabla en el menú de la izquierda, y luego desde el menú superior ir a “Insert” para poder agregar los nuevos registros. Tener en cuenta que el campo “id” no debe ser completado, ya que al ser autoincremental el valor es agregado automáticamente.



Opción 2 – Desde el código

Desde el código podemos insertar registros de la siguiente forma, teniendo en cuenta que:

- El campo id como lo definimos auto incremental no debe ser agregado
- Para que se registre el cambio debemos realizar un commit.

insert.py

```
import mysql.connector

mibase = mysql.connector.connect(
  host="localhost",
  user="root",
  passwd="",
  database="mi_plantilla2"
)
micursor = mibase.cursor()

sql = "INSERT INTO producto (titulo, ruta, descripcion) VALUES (%s, %s, %s)"
datos = ("Tema 3", "ruta3", "descripción 3")

micursor.execute(sql, datos)

mibase.commit()

print(micursor.rowcount, "Cantidad de registros agregados.")
```

Si en lugar de utilizar “execute()”, utilizamos “executemany()”, podemos pasar una lista de tuplas.

```
datos = [("Tema 4", "ruta4", "descripción 4"),
("Tema 5", "ruta5", "descripción 5"),
("Tema 6", "ruta6", "descripción 6"),
]

micursor.executemany(sql, datos)
```

Paso 6 – Borrar registro

Opción 1 – Desde phpmyadmin

Borrar datos gráficamente es muy simple, solo tenemos que presionar en el ícono que se encuentra en la fila del registro que queremos borrar.

Opción 2 – Desde el código

Para borrar el registro desde el código debemos ejecutar “DELETE FROM tabla WHERE registro”

insert.py

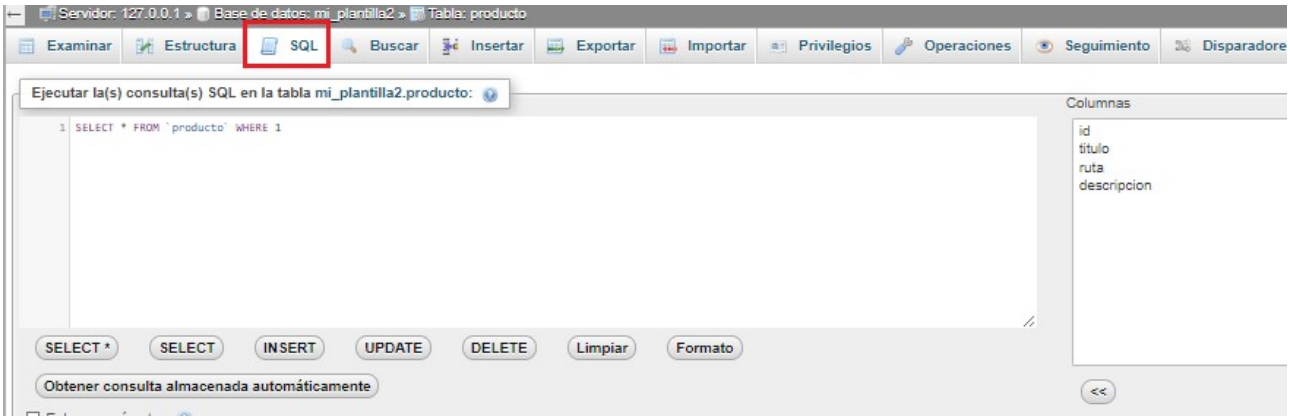
```
import mysql.connector

mibase = mysql.connector.connect(
  host="localhost",
  user="root",
  passwd="",
  database="mi_plantilla2"
)
micursor = mibase.cursor()

sql = "DELETE FROM producto WHERE id = %s"
dato = ('2',)

micursor.execute(sql, dato)

mibase.commit()

print(micursor.rowcount, "Registro borrado")
```

Paso 7 – Seleccionar registros

En el caso de seleccionar registros en ambos casos debemos introducir el código sql de los datos a seleccionar.

Opción 1 – Desde phpmyadmin

Desde phpmyadmin, podemos ir a la solapa “SQL” en donde contamos con algunas herramientas para armar la consulta específica que queremos realizar.



Opción 2 – Desde el código

Podemos recuperar los datos mediante un “SELECT”, si queremos traer todos los campos simplemente ponemos un asterisco, en caso contrario seleccionamos los campos separados por coma. Para darle formato a los datos recuperados utilizamos “fetchall()”

resultado.py

```
import mysql.connector

mibase = mysql.connector.connect(
  host="localhost",
  user="root",
  passwd="",
  database="mi_plantilla2"
)
micursor = mibase.cursor()

sql = "SELECT * FROM producto"


micursor.execute(sql)
resultado = micursor.fetchall()

for x in resultado:
    print(x)
```

Nos retorna:

```
(1, 'Tema 3', 'ruta3', 'descripción 3')
(4, 'Tema 5', 'ruta5', 'descripción 5')
```