Manual Técnico - Sistema de gestión de librería:

[proyecto 1 : grupo 1]

Escrito por: Agustin Ortiz Cavallero 4°2

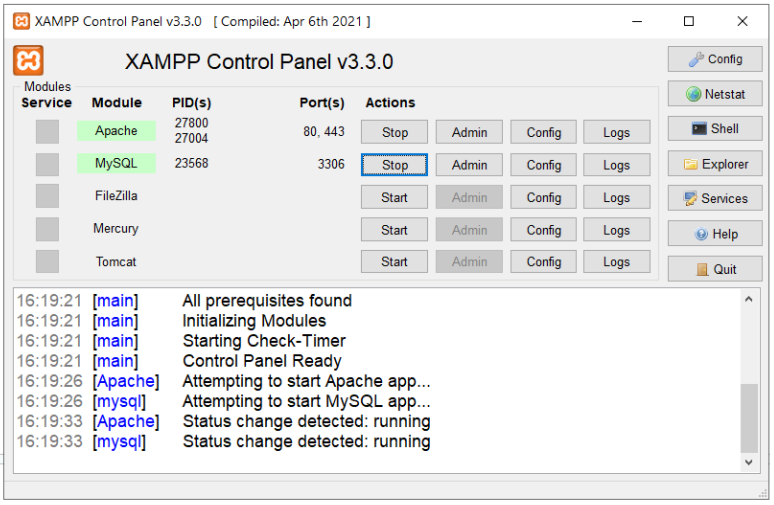
Este manual contiene la información necesaria para poder comenzar a utilizar el programa de gestión de librería SQL, además, también sobre cómo modificar el código base de Python y SQL ordenadamente para sus necesidades.

Fundamentalmente, se necesita tener instalado lo siguiente:

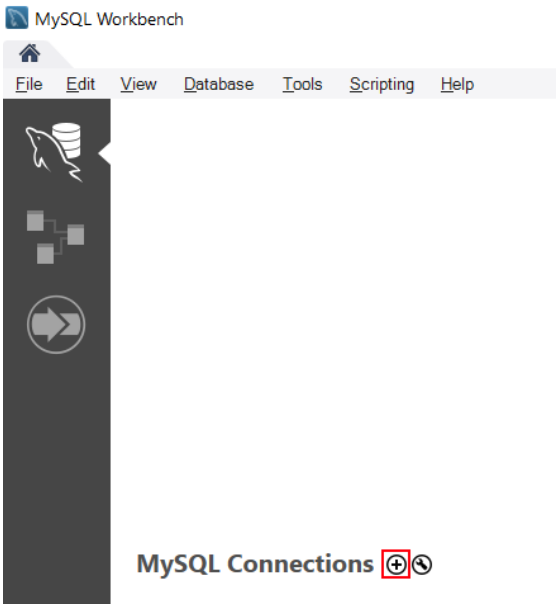
-Python (version 3.9 o mayor) + el driver llamado “MySQL Connector”
-Pycharm (o otra IDE para correr el programa)
-MySQL Workbench
-XAMPP

MySQL Workbench + XAMPP:

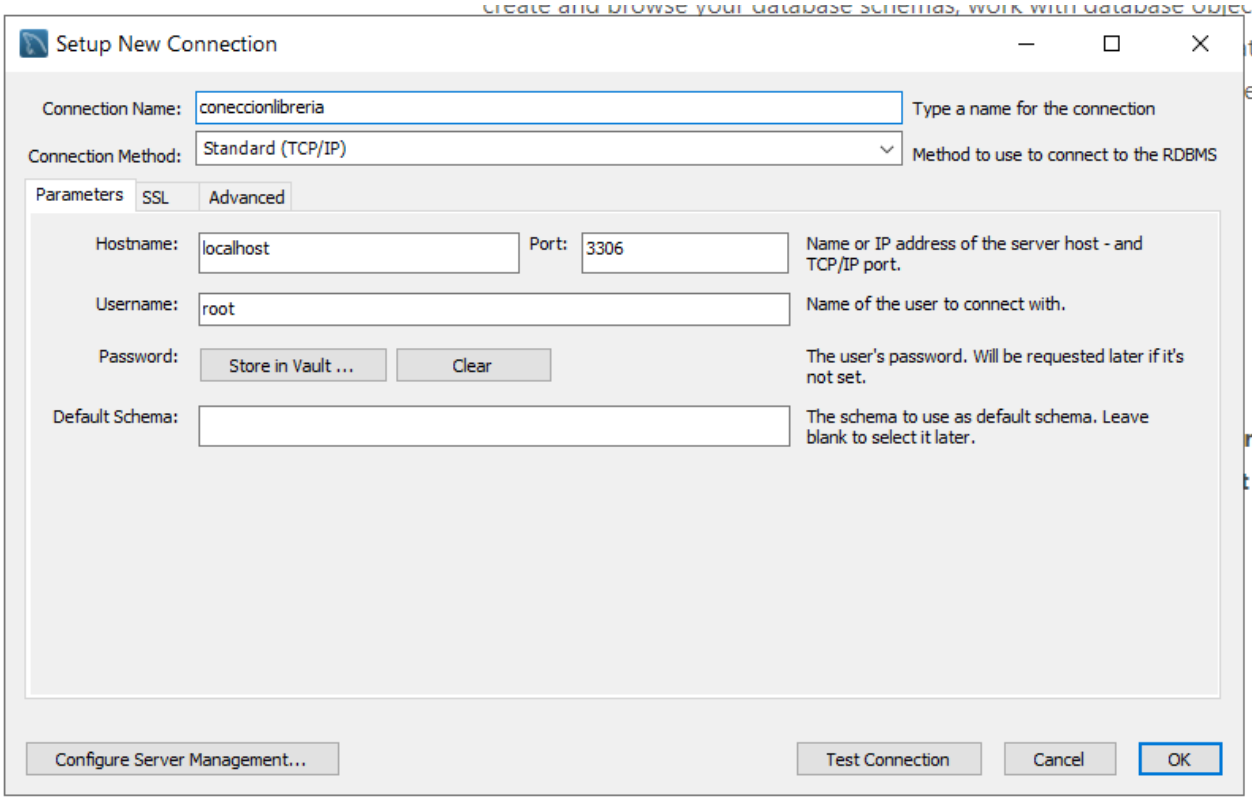
Si no tenes instalados estos programas, se pueden instalar de sus respectivos links oficiales, adjuntados a continuación:
[XAMPP] : https://www.apachefriends.org/es/index.html
[MySQL Workbench] : https://dev.mysql.com/downloads/workbench/

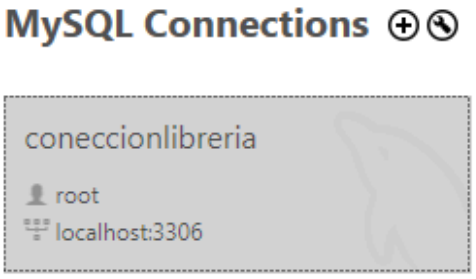


Antes de abrir el MySQL Workbench, se debe ir al panel de control de XAMPP y activar los módulos “Apache” y “MySQL”; Luego de ello se puede iniciar el Workbench.

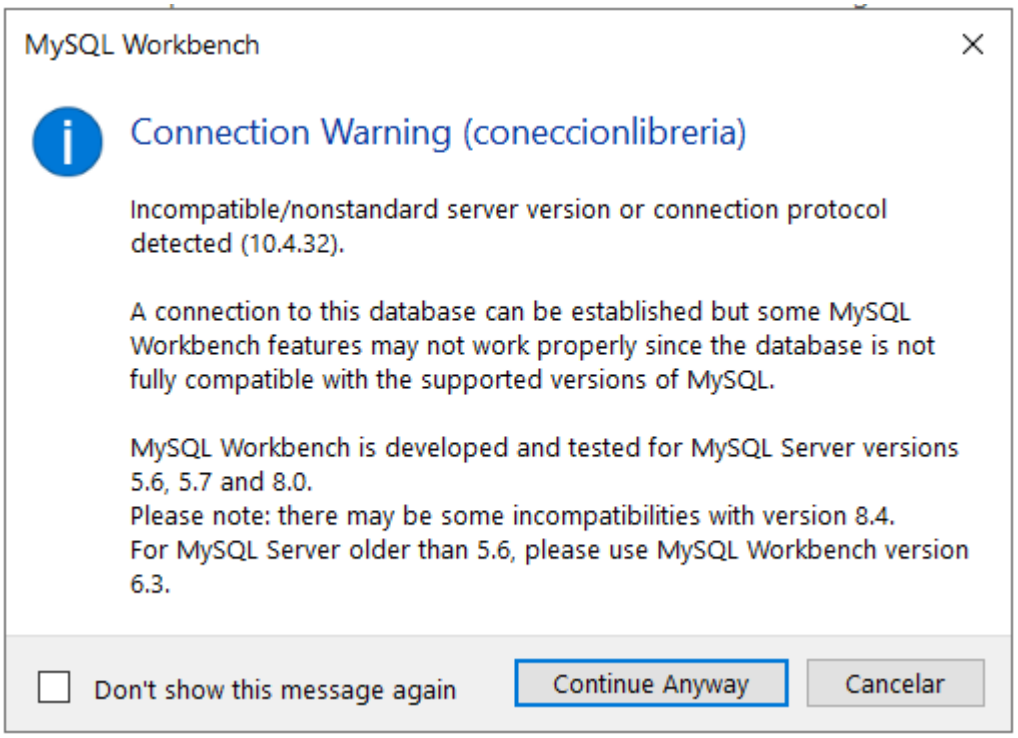


Haciendo click en el símbolo de añadir (+) dentro del Workbench va a aparecer un apartado para setear una nueva coneccion: Las opciones que se deben cambiar son únicamente el nombre de la conexión (Connection Name) el cual puede ser elegido por el usuario ya que no afecta en el funcionamiento de la base de datos, y el HostName, que si o si debe estar puesto como “localhost”, demostrado así en la siguiente imagen. Teniendo las opciones listas, se debe clickear en “Ok”, que finalizaría la creación de esta misma conexión.

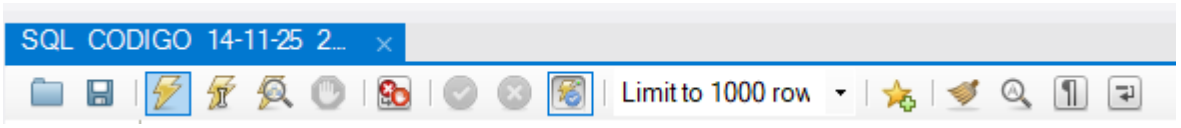




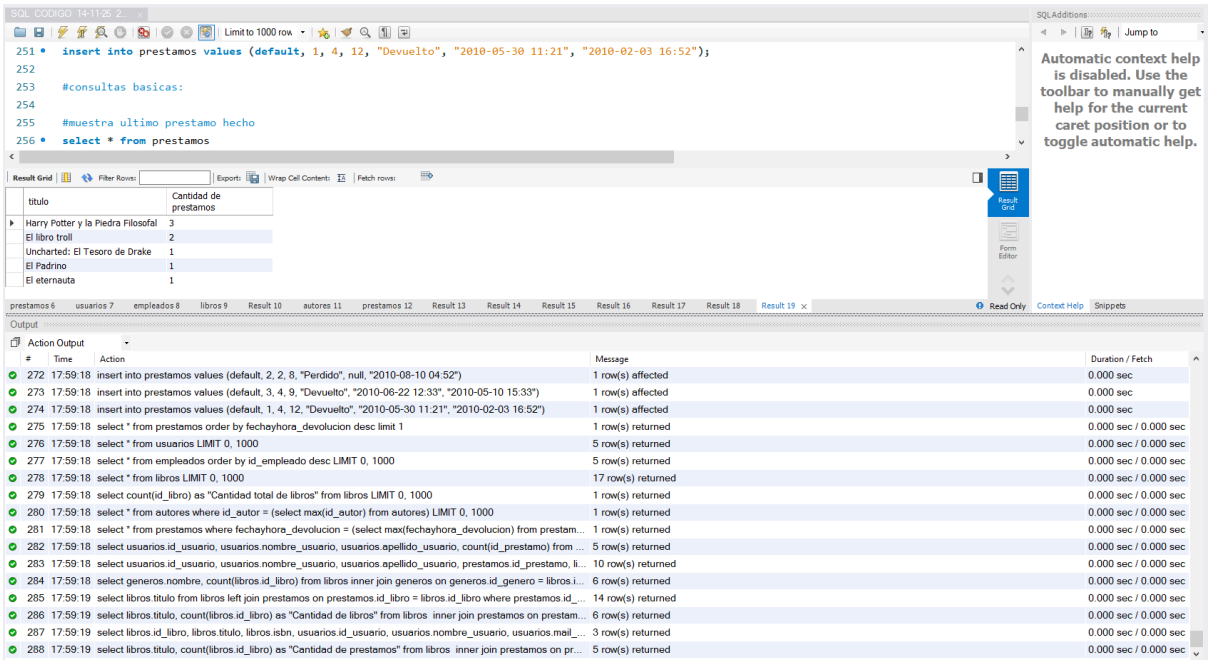
Luego se puede entrar a la conexión, clickeando de la manera que aparece en la foto demostrativa de la izquierda.
Un error común que suele aparecer al ingresar en la conexión es el siguiente, pero solamente hay que clickear en “Continue Anyway” y no va a afectar en la base de datos.



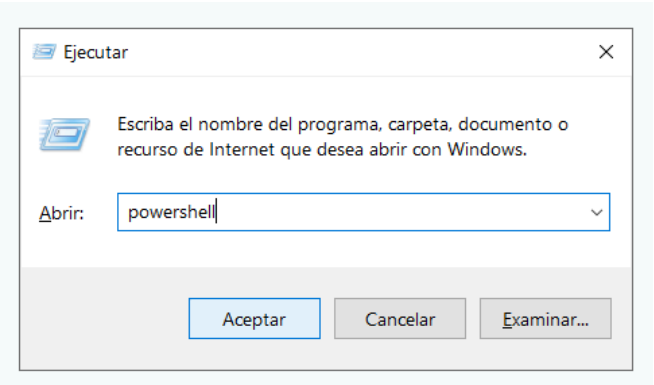
Al haber seguido estos pasos, verá el editor de código de WorkBench; Simplemente debe abrir el archivo .sql adjuntado dentro del Github seleccionando la opción a continuación, y buscar la dirección en la que se encuentra en tu computadora este archivo.



Al haber abierto el código sql, debes clickear en el icono de relámpago sin seleccionar ninguna parte del código.
Al hacer esto se va a instanciar todo el código, no debería tirar errores en la consola ni signos de error potencial (⚠️).



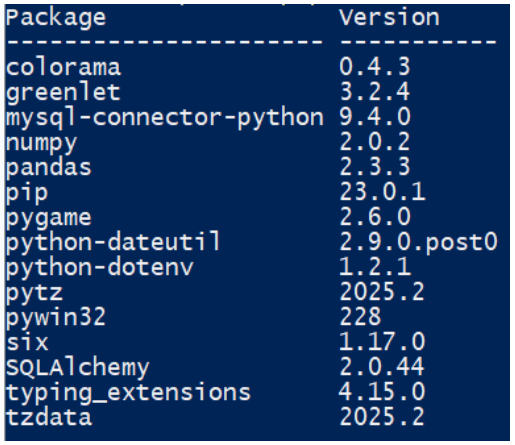
Python + MySQL Connector:



Para instalar Python, primero debes ingresar al PowerShell (el cmd tambien se podria utilizar, pero es preferencia del usuario);
Para ello debes utilizar la combinación de teclas Windows + R, y luego en el apartado “Ejecutar” escribir “powershell” y tocar en donde dice Aceptar.

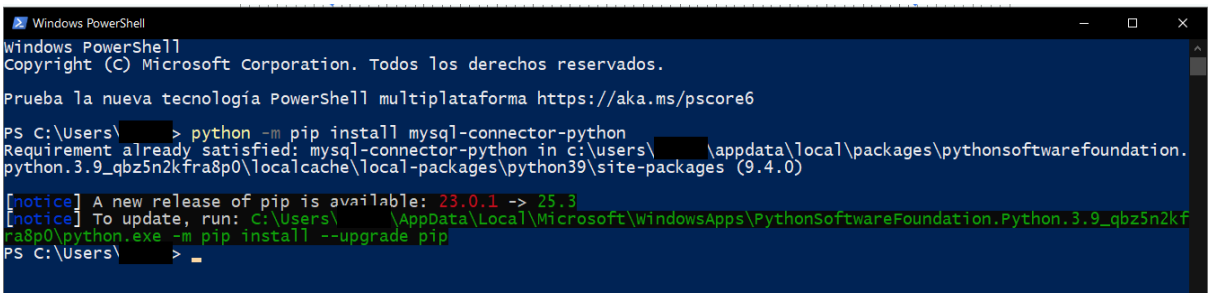
Al escribir “python” en el Powershell, si está instalado anteriormente se va a mostrar la versión y la fecha en la que se lanzó, pero si su computadora no la tiene instalada lo mandará a la página de descarga para la versión más reciente actualmente en la Microsoft Store.

Al instalarla correctamente, se debe cerrar y abrir de nuevo a la ventana de powershell e ingresar el siguiente comando: “python -m pip install mysql-connector-python”

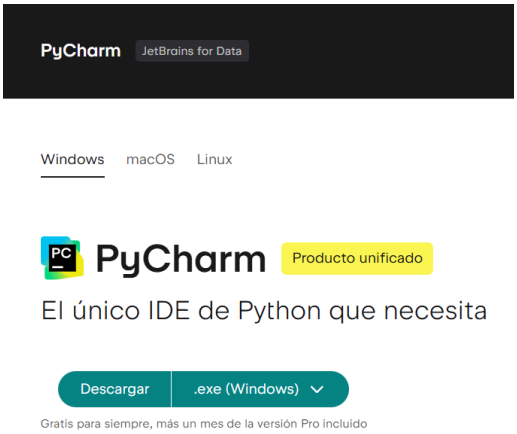


```
Package                 Version
----------------------- -----------
colorama                0.4.3
greenlet                3.2.4
mysql-connector-python  9.4.0
numpy                   2.0.2
pandas                  2.3.3
pip                     23.0.1
pygame                  2.6.0
python-dateutil         2.9.0.post0
python-dotenv           1.2.1
pytz                    2025.2
pywin32                 228
six                     1.17.0
SQLAlchemy              2.0.44
typing_extensions       4.15.0
tzdata                  2025.2
```

Si ya está instalado el MySQL Connector previamente te puede llegar a aparecer un error “Requirement already satisfied”, aunque antes de instalarlo podrias ingresar el comando “pip list”, que muestra en una lista de todos los agregados de Python que tenés ya instalados en tu computadora, asegurandote de no tenerlo:

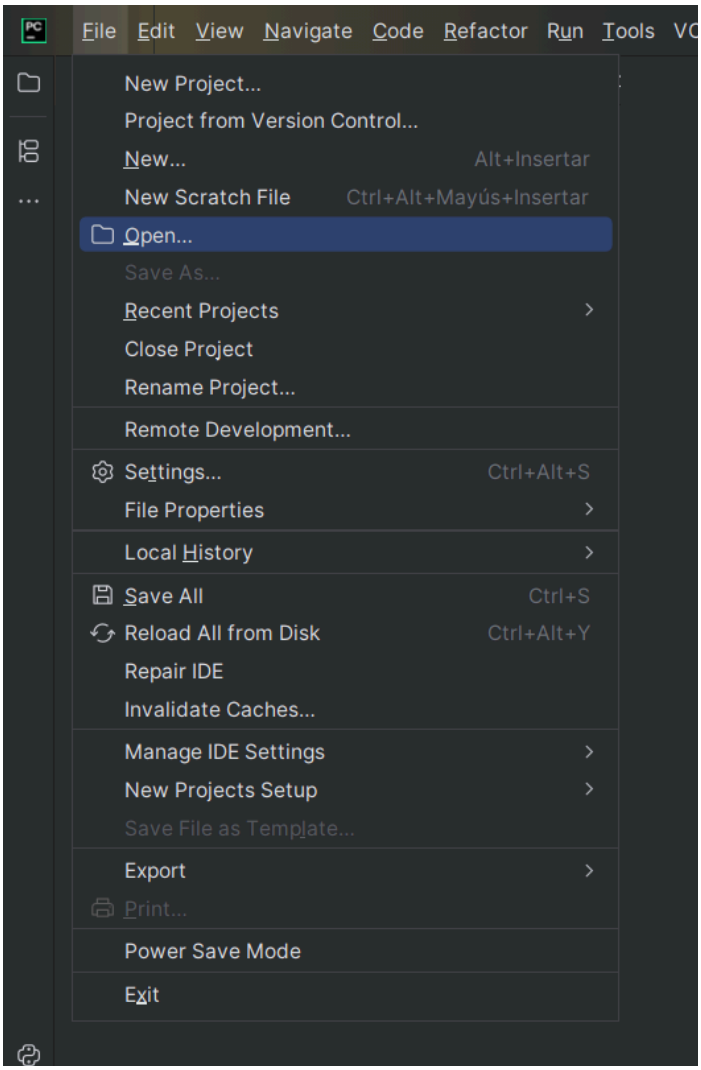


```
Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\[redacted]> python -m pip install mysql-connector-python
Requirement already satisfied: mysql-connector-python in c:\users\[redacted]\appdata\local\packages\pythonsoftwarefoundation.
python.3.9_qbz5n2kfra8p0\localcache\local-packages\python39\site-packages (9.4.0)

[notice] A new release of pip is available: 23.0.1 -> 25.3
[notice] To update, run: C:\Users\[redacted]\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.9_qbz5n2kf
ra8p0\python.exe -m pip install --upgrade pip
PS C:\Users\[redacted]>
```



Para utilizar nuestro programa, vas a necesitar un editor para correr / actualizar el codigo .py de la manera que usted desee; En el caso de nuestro programa utilizamos la IDE “PyCharm” para programarlo, a continuación se ve la pagina de donde se descarga el archivo del instalador:

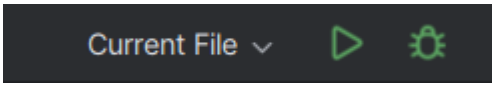
https://www.jetbrains.com/es-es/pycharm/download/?section=windows

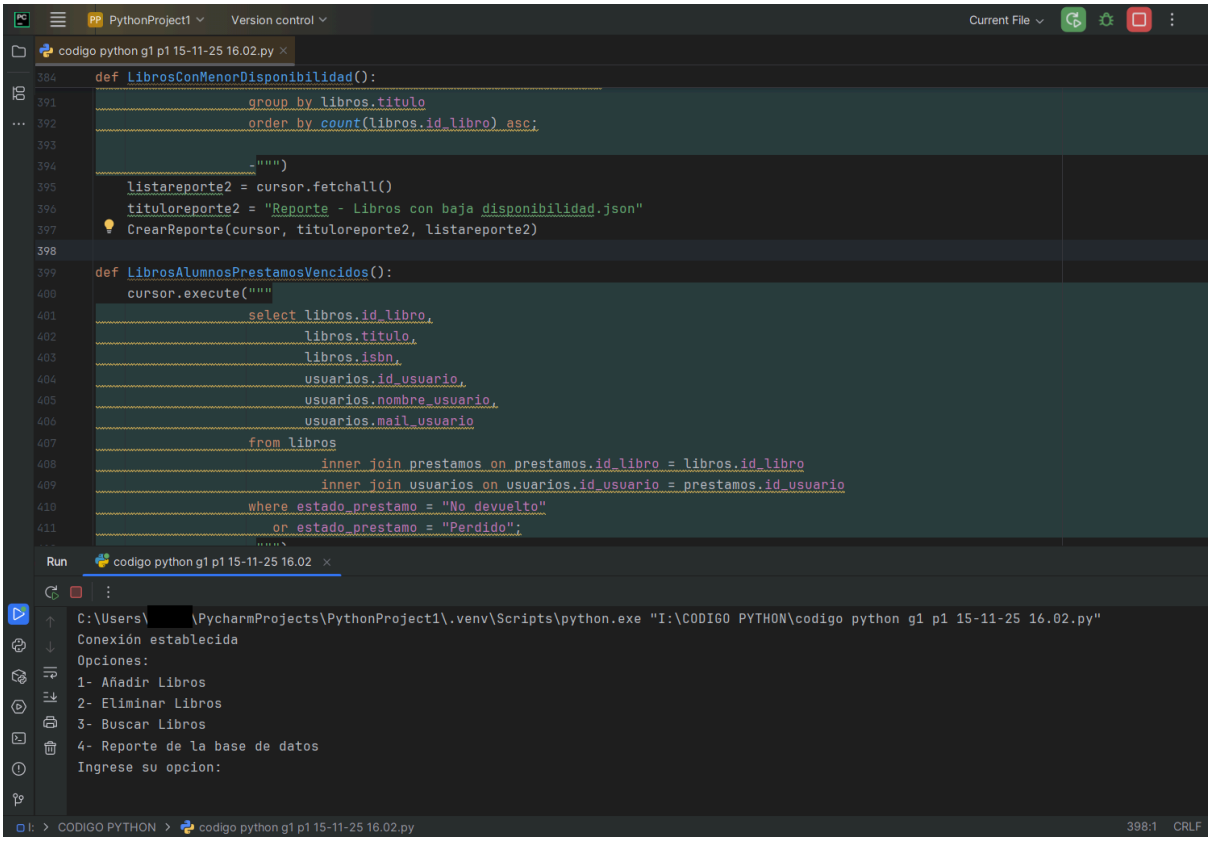


Al entrar al pycharm, deben ir a la barra de tareas que se encuentra en la parte de arriba de este programa y en la sección llamada “File” [Archivo] se debe clickear en “Open” [Abrir]

Luego, tienen que buscar el directorio del archivo .py que se encuentra en el GitHub

Para iniciar el programa, debes clickear en el botón verde, que se asemeja a una flecha apuntando hacia la derecha, que se encuentra a la derecha más cercana al centro, de la barra de tareas.





¡Ahora pueden comenzar a utilizar el programa! :D

Especificaciones del código (SQL + Python):

Para cambiar el código, sugerimos que utilices la estructura que utilizamos para programarlo, para mejor organización:

```sql
drop database if exists biblioteca;
create database biblioteca;
use biblioteca;

create table autores (
id_autor int auto_increment,
nombre_autor varchar (20),
apellido_autor varchar (20),
primary key(id_autor)
);
insert into autores
values (default, "J.K." , "Rowling");
insert into autores
values (default, "Ruben" , "Doblas");
insert into autores
values (default, "J. R. R. " , "Tolkien");
insert into autores
values (default, "Mario" , "Puzo");
insert into autores
values (default, "Tom" , "Clancy");
insert into autores
values (default, "Hector", "Oesterheld");
insert into autores
values (default, "Muneyuki", "Kaneshiro");
insert into autores
values (default, "Yoichi", "takahashi");
insert into autores
values (default, "Christopher", "Golden");
insert into autores
values (default, "Antoine", "Saint-Exupéry");
```

Para el código de SQL, lo separamos en 3 partes: La creación de la base de datos (primeras 3 líneas de código), la creación de las distintas tablas e ingreso de datos de las mismas (todos los “create table” son seguidos de “insert into” para no perder tiempo yendo entre dos partes distintas del código), y las consultas SQL (ej: select * from prestamos)

Para crear una tabla se deben tener en cuenta varias cosas:

Para iniciar una tabla se utiliza el codigo “create table *nombre* (“
y para finalizarla se utiliza “); ”
Dentro de las tablas hay atributos escritos en secuencia de nombre, tipo de dato (como ejemplo: varchar [caracteres variables], int [entero], date [fecha : AAAA/MM/DD], datetime [fecha y hora : AAAA/MM/DD HH:MM]), cantidad de caracteres límite del atributo (ejemplo: varchar(60) - significa que tiene un límite de 60 caracteres), y al final otros comandos que puede tener un atributo como primary key, que hace que la variable sirva para identificar de forma única a cada fila de una tabla; AUTO_INCREMENT, que hace que en cada registro se le sume un valor de 1 a un atributo entero, o tambien podria ser NOT NULL, que no deja que un campo este sin completar, entre otros comandos.

Para ingresar datos a una tabla se utiliza el codigo “insert into *nombre de la tabla* values (*valores*)

Para ser mas especifico y no insertar cada variable en orden de como estan creadas en la tabla original, se pueden definir cuáles variables se quieren únicamente llenar poniendo un paréntesis luego del nombre de la tabla y especificando dentro de él cada atributo separado por comas:

Ejemplo:
insert into usuarios(nombre_usuario, apellido_usuario, mail_usuario, telefono_usuario, direccion_usuario, dni_usuario)
values ("Franco", "Armani", "notengomanos@gmail.com", 1175820756, "paysandu 5432", 30945821);

Truco para facilitar primary keys! : El comando default, mientras que sirve al crear una tabla para ponerle un valor de defecto a un atributo:

[Ejemplo de tabla: pais varchar(100) DEFAULT 'Argentina']

También se puede utilizar en un insert into cuando el atributo que es primary key tiene un “auto_increment”, se van a definir automáticamente los valores de la primary key en cada registro.

Ejemplo de código:

```
create table localidades(
 id_localidad int primary key auto_increment,
 localidad_libro text not null
);

insert into localidades values (default, "Biblioteca 1 (Pared izquierda 1° estante de abajo para arriba)");
```

En Python, organizamos el código de la siguiente manera:

```
import mysql.connector
from mysql.connector import errorcode
import json
```

Comienza con la importación de librerías: en el caso de nuestro lado proyecto importamos la librería json, para poder imprimir nuestros reportes en archivos de tipo .json, y la librería MySQL Connector, la librería que funciona como un puente entre la base de datos en MySQL Workbench y el código ejecutado para la base de datos en Python.

Luego, se declaran cnx y cursor, ambos como None, la variable cursor sirve para ejecutar comandos dentro de python en la base de datos, y la variable cnx sirve para la parte de conexiones entre la base de datos y el código Python.

```
cursor = None
cnx = None
```

Luego se encuentra el código para realizar esta conexión entre el Workbench y el python, detecta si los datos que tiene python sobre la base de datos como su nombre, contraseña, host y nombre de base de datos concuerdan con los que están actualmente en el Workbench.

```
def ConectarBase():
    global cnx, cursor

    try:
        cnx = mysql.connector.connect(user="root", password="", host="Localhost", database="biblioteca")
        cursor = cnx.cursor(dictionary=True)
        print('Conexión establecida')
    except mysql.connector.Error as err:
        if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
            print('Usuario o contraseña incorrectos!')
        elif err.errno == errorcode.ER_BAD_DB_ERROR:
            print('La base de datos no existe!')
        else:
            print(err)
```

Luego de este código se encuentra el código base para el árbol binario, el cual dura 80 líneas, pero es necesario para la búsqueda de id y de ISBN.

A continuación de ese código, se encuentran definidas todas las funciones del menú, las cuales se mencionan a continuación:

(Nota de programador: Si vas a hacer una función personalizada dentro del código base no es obligatorio poner cursor como una variable definida en las funciones, al ser una variable global, que funciona en cualquier lugar del código, pero nos dimos cuenta muy tarde)

```
BuscarLibrosnombre(cursor)
```

Función utilizada para buscar libros por nombre, selecciona todos los libros con el nombre ingresado en una variable por el usuario, y luego imprime toda la lista.

```
AniadirLibros(cursor)
```

Función utilizada para añadir libros a la base de datos, pregunta titulo, isbn y numero de paginas del libro; Luego, muestra una tabla con todas las ids y nombres de las locaciones disponibles en el local para que vos le ingreses la id de la locación; Luego, pregunta el género del libro, también utilizando una tabla de géneros y pregunta su id; Después pregunta, también por método de tabla e id, la editorial del cuento, pero esta vez pregunta antes si está o no en la tabla; si no se encuentra en la tabla, te deja ingresar a la base de datos el nombre de la nueva editorial y le agrega automáticamente al libro su id.

```
BuscarLibrosID(cursor)
```

Función utilizada para buscar libros por id, pone en una lista todos los ids disponibles de los libros, y usa un árbol binario para verificar si realmente hay en la base de datos un libro con la misma id y luego lo muestra.

```
BuscarLibrosISBN(cursor)
```

Función utilizada para buscar libros por ISBN, utiliza el mismo proceso que la función anterior pero en este caso para el ISBN.

```
borrarporid(cursor)
```

Función utilizada para borrar un libro por ID, utiliza el mismo proceso de árbol binario que las funciones anteriores para buscar si un libro con la id ingresada existe y después lo borra de la base de datos, primero borrando las conexiones del libro (instancias de las tablas prestamos, libros_autores donde se encuentra conectada como foreign key el id_libros) y luego la instancia de libros que coincida con la id ingresada por el usuario.

```
EliminarLibrosPorTitulo(cursor):
```

Función utilizada para borrar libros, buscando los que tienen el mismo título ingresado y poniendo sus ids en una lista, para luego preguntarle al usuario la id del libro que quiere borrar; Luego de eso, sucede el mismo proceso que la función borrarporid().

```
CrearReporte(cursor,tituloreporte,listareporte)
```

Esta función es utilizada en la opción 4 del menú, para crear reportes: Utiliza como variables ingresadas tituloreporte: que contiene el título del reporte, ej: tituloreporte = “librosmasprestados.json”; y listareporte, que contiene la lista / tabla entera que fue conseguida al hacer la consulta SQL.

```
LibrosMasPrestados()
LibrosConMenorDisponibilidad()
LibrosAlumnosPrestamosVencidos()
```

Estas funciones tienen iguales procesos, pero diferentes consultas.

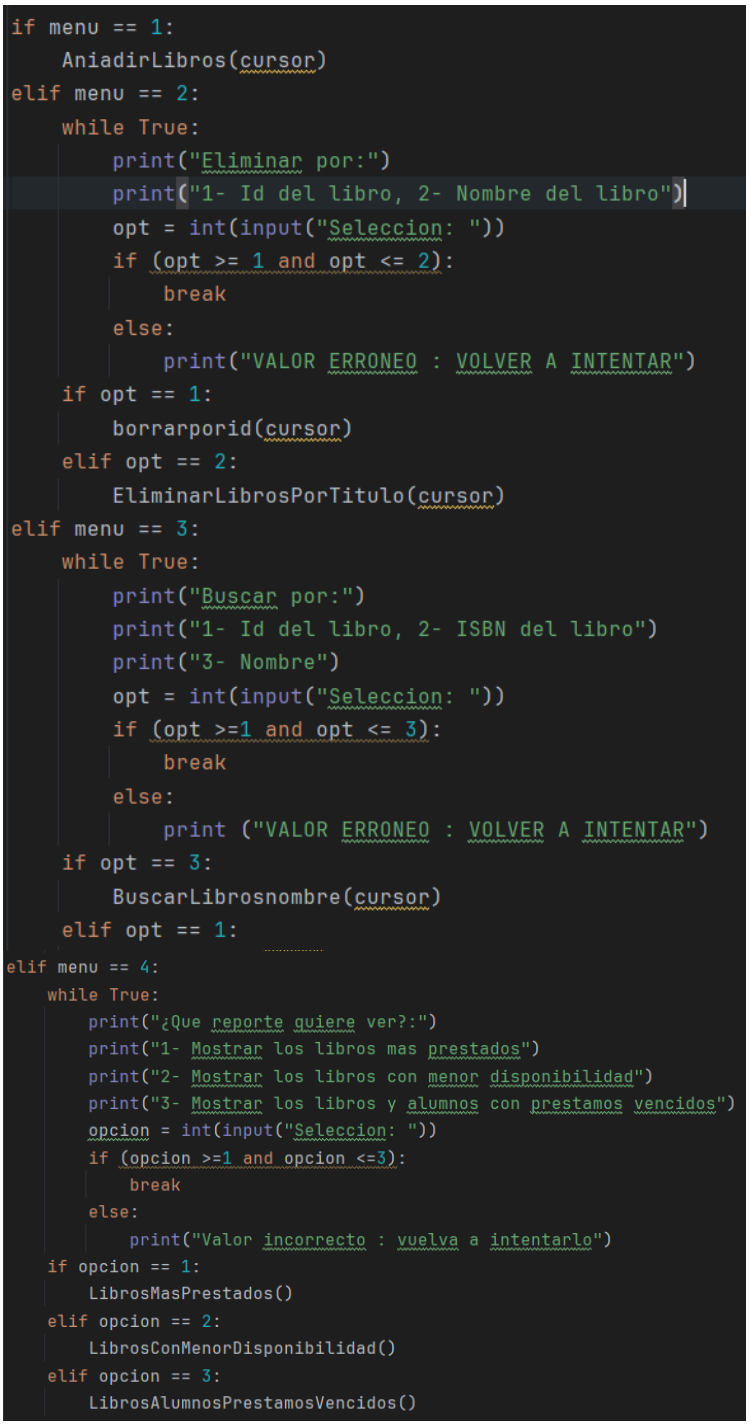
Después de las funciones fundamentales, viene el codigo del menu:

```
while True:
    print ("--Sistema Gestor de Libreria--")
    print ("Opciones:")
    print ("1- Añadir Libros")
    print ("2- Eliminar Libros")
    print ("3- Buscar Libros")
    print ("4- Reporte de la base de datos")
    menu = int(input("Ingrese su opcion: "))
    if menu >= 1 and menu <= 4:
        break
    else:
        print ("OPCION INVALIDA -- VUELVA A INTENTAR")
```

Muestra las opciones disponibles en el código y te permite ingresar la opción deseada, mientras esté entre el rango habilitado de opciones (hay 4 preguntas, entonces el rango válido de valores es entre el 1 y el 4), si no se encuentra dentro del rango, te pregunta que vuelvas a ingresar otro valor, esta opción se define en la variable “menu”.

En esta parte del código para el menú, se establecen las distintas opciones.
Esto también incluye las opciones que pueden haber dentro de las del menú principal (ejemplo: Buscar por Id del libro, ISBN del libro o Nombre); Para ellas se utiliza una segunda variable de menú que puede ser o opt o opción dependiendo de quién hizo cada parte del código.

Al final del proyecto, se encuentra código de muestra pero más básico que usuarios principiantes a el código SQL y el código python puedan entender más fácilmente y utilizar como base.



```
if menu == 1:
    AniadirLibros(cursor)
elif menu == 2:
    while True:
        print("Eliminar por:")
        print("1- Id del libro, 2- Nombre del libro")
        opt = int(input("Seleccion: "))
        if (opt >= 1 and opt <= 2):
            break
        else:
            print("VALOR ERRONEO : VOLVER A INTENTAR")
    if opt == 1:
        borrarporid(cursor)
    elif opt == 2:
        EliminarLibrosPorTitulo(cursor)
elif menu == 3:
    while True:
        print("Buscar por:")
        print("1- Id del libro, 2- ISBN del libro")
        print("3- Nombre")
        opt = int(input("Seleccion: "))
        if (opt >=1 and opt <= 3):
            break
        else:
            print ("VALOR ERRONEO : VOLVER A INTENTAR")
    if opt == 3:
        BuscarLibrosnombre(cursor)
    elif opt == 1:
elif menu == 4:
    while True:
        print("¿Que reporte quiere ver?:")
        print("1- Mostrar los libros mas prestados")
        print("2- Mostrar los libros con menor disponibilidad")
        print("3- Mostrar los libros y alumnos con prestamos vencidos")
        opcion = int(input("Seleccion: "))
        if (opcion >=1 and opcion <=3):
            break
        else:
            print("Valor incorrecto : vuelva a intentarlo")
    if opcion == 1:
        LibrosMasPrestados()
    elif opcion == 2:
        LibrosConMenorDisponibilidad()
    elif opcion == 3:
        LibrosAlumnosPrestamosVencidos()
```