**RECURSOS**

**APUNTS MÒDUL SISTEMES DE GESTIÓ EMPRESARIAL**

**RA1: Identifica sistemes de gestió de dades** mitjançant la connexió a una base de dades MySQL i el maneig de les dades extretes de la mateixa.

**RA2: Implanta sistemes de gestió de bases de dades** utilitzant la configuració del connector MySQL amb Python i integració amb interfícies gràfiques.

**RA3: Interpreta la documentació tècnica** de MySQL i Python connector per solucionar problemes de connexió i consultar dades de manera correcta.

**Objectiu**

L'objectiu d'aquesta pràctica és aprendre a connectar-se a una base de dades MySQL des d'un script de Python i integrar-ho amb una interfície gràfica utilitzant la llibreria Tkinter. Això permetrà a l'estudiant comprendre com es gestionen les dades en una aplicació de múltiples capes, afavorint la comprensió del model client-servidor.

# **Guia de conexió de bases de dades Mysql Amb Python**

## Pas 1: Instal·lar la biblioteca si encara no ho has fet

pip install mysql-connector-python

Aquesta comanda instal·la un connector que permet que Python interactuï amb una base de dades MySQL. Ens facilitarà la connexió, l'execució de consultes SQL i la gestió de resultats.

## Pas 2: Conexió de MySQL desde Python

 import mysql.connector

from mysql.connector import Error

try:

connection = mysql.connector.connect(

host='el\_teu\_host', # Per exemple 'localhost'

database='la\_teva\_base\_de\_dades', # Nom de la base de dades

user='el\_teu\_usuari', # Usuari MySQL

password='la\_teva\_contrasenya' # Contrasenya de l'usuari MySQL

)

if connection.is\_connected():

print(f"Connectat a MySQL Server")

cursor = connection.cursor()

# Consulta SQL per obtenir dades

query = "SELECT \* FROM taula;"

cursor.execute(query)

resultats = cursor.fetchall()

if resultats:

for fila in resultats:

print(fila)

else:

print("No s'han trobat dades.")

except Error as e:

print(f"Error en connectar a MySQL: {e}")

finally:

if connection.is\_connected():

cursor.close()

connection.close()

print("Connexió MySQL tancada")

1. **Connexió a la base de dades**: Es fa servir mysql.connector.connect() per connectar-se a la base de dades.
2. **Consulta SQL**: La consulta SELECT \* FROM prueba; s'executa per extreure totes les dades de la taula prueba.
3. **Mostrar les dades**: Es comproven els resultats i es mostren per consola fila per fila.
4. **Gestió d'errors**: Es capturen i es mostren possibles errors en la connexió.
5. **Tancar la connexió**: La connexió es tanca en acabar l'execució.

Aquest codi intenta establir una connexió amb la base de dades MySQL utilitzant el connector que hem instal·lat. Si la connexió té èxit, recupera totes les files de la taula especificada i les mostra per pantalla. També es preveu el tractament d'errors per assegurar-se que el programa no es bloquegi en cas de fallar la connexió.

## Pas 3. Connexió de dades amb Tkinter

import tkinter as tk

from tkinter import ttk

import mysql.connector

def obtenir\_dades():

try:

connection = mysql.connector.connect(

host='el\_teu\_host',

database='la\_teva\_base\_de\_dades',

user='el\_teu\_usuari',

password='la\_teva\_contrasenya'

)

if connection.is\_connected():

cursor = connection.cursor()

query = "SELECT \* FROM taula;"

cursor.execute(query)

return cursor.fetchall()

except mysql.connector.Error as e:

print(f"Error: {e}")

return []

finally:

if connection.is\_connected():

cursor.close()

connection.close()

def omplir\_treeview(dades):

for fila in dades:

tree.insert("", "end", values=fila)

# Configuració de la finestra

root = tk.Tk()

root.title("Dades de la Base de Dades")

# Creació de la vista amb Treeview

tree = ttk.Treeview(root, columns=(1, 2, 3), show="headings", height="10")

tree.heading(1, text="ID")

tree.heading(2, text="Nom")

tree.heading(3, text="Cognom")

tree.pack()

# Obtenir les dades i omplir la interfície

dades = obtenir\_dades()

omplir\_treeview(dades)

# Iniciar la finestra

root.mainloop()

Aquest codi fa servir Tkinter per crear una interfície gràfica senzilla que mostra el contingut de la base de dades en un widget tipus Treeview. Aquesta interfície és dinàmica, ja que obté les dades directament de la base de dades i les carrega de manera interactiva.

## Treeview

El **Treeview** és un widget de la llibreria **Tkinter** de Python que es fa servir per mostrar informació en format de taula o estructura d'arbre (similar a un full de càlcul). És molt útil quan volem mostrar dades estructurades en files i columnes.

Visualització en taula: El Treeview permet mostrar dades en format tabular, és a dir, amb diverses columnes i files.

Capçaleres de columnes: El Treeview permet definir capçaleres per a cada columna, com ho faria una taula en un full de càlcul. Això és útil per identificar clarament què representa cada columna.

Visualització jeràrquica: A més de mostrar dades en format de taula, també pot mostrar informació de manera jeràrquica (per això s'anomena "Treeview"). Això vol dir que pots tenir "nodos" que continguin altres "nodos" a sota, com en un explorador d'arxius, on les carpetes poden tenir subcarpetes o arxius.

Funcionalitats addicionals:

* Selecció de files: Els usuaris poden seleccionar una o més files.
* Desplaçadors: Quan hi ha més files o columnes de les que caben en l'espai visible, pots afegir desplaçadors horitzontals o verticals.
* Interactivitat: Pots programar accions que s'executin quan un usuari fa clic en una fila o selecciona un element.

**Data actualització i autor/res:** Ignacio López-Aylagas 15/10/2024