Lab 09.02 Python and Databases

Using databases

### Overview

Calling the sql commands in python

### Before the lab

Install the python package

pip install mysql-connector

You will need to have your mysql server up and running,

I would usually create the database and tables on the server and not through python.

In this lab I show you how to create the table and then perform the CRUD operations.

NOTE: The username and password for your database in WAMP the default is root and blank,

You should make a new file for each of these tasks.

## Not usually done (create database and tables)

1. Create a database called datarepresentation using a python script

import mysql.connector

connection = mysql.connector.connect(

  host="localhost",

  user="root",

  password=""

)

mycursor = connection.cursor()

mycursor.execute("CREATE database datarepresentation ")

mycursor.close()

connection.close()

1. Create the table in the database with the python script

import mysql.connector

mydb = mysql.connector.connect(

  host="localhost",

  user="root",

  password="",

  database="datarepresentation"

)

mycursor = mydb.cursor()

sql="CREATE TABLE student (id INT AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(255), age INT)"

mycursor.execute(sql)

mycursor.close()

connection.close()

## CRUD operations on a table, this is what you would normally do from an application

1. Insert data

import mysql.connector

db = mysql.connector.connect(

  host="localhost",

  user="root",

  password="",

  database="datarepresentation"

)

cursor = db.cursor()

sql="insert into student (name, age) values (%s,%s)"

values = ("Mary",21)

cursor.execute(sql, values)

db.commit()

print("1 record inserted, ID:", cursor.lastrowid)

mycursor.close()

connection.close()

1. View data

import mysql.connector

db = mysql.connector.connect(

  host="localhost",

  user="root",

  password="",

  database="datarepresentation"

)

cursor = db.cursor()

sql="select \* from student where id = %s"

values = (1,)

cursor.execute(sql, values)

result = cursor.fetchall()

for x in result:

  print(x)

mycursor.close()

connection.close()

1. Update data

import mysql.connector

db = mysql.connector.connect(

  host="localhost",

  user="root",

  password="",

  #user="datarep",  # this is the user name on my mac

  #passwd="password" # for my mac

  database="datarepresentation"

)

cursor = db.cursor()

sql="update student set name= %s, age=%s  where id = %s"

values = ("Joe",33, 1)

cursor.execute(sql, values)

db.commit()

print("update done")

mycursor.close()

connection.close()

1. Delete

import mysql.connector

db = mysql.connector.connect(

  host="localhost",

  user="root",

  password="",

  #user="datarep",  # this is the user name on my mac

  #passwd="password" # for my mac

  database="datarepresentation"

)

cursor = db.cursor()

sql="delete from student where id = %s"

values = (1,)

cursor.execute(sql, values)

db.commit()

print("delete done")

mycursor.close()

connection.close()

Delete from student where id = 1;

## Put it into a file that can be used from another file (eg from your flask app)

import mysql.connector

class StudentDAO:

    host =""

    user = ""

    password =""

    database =""

    connection = ""

    cursor =""

    def \_\_init\_\_(self):

        #these should be read from a config file

        self.host="localhost"

        self.user="root"

        self.password=""

        self.database="datarepresentation"

    def getCursor(self):

        self.connection = mysql.connector.connect(

            host=self.host,

            user=self.user,

            password=self.password,

            database=self.database

        )

        self.cursor = self.connection.cursor()

        return self.cursor

    def closeAll(self):

        self.connection.close()

        self.cursor.close()

    def create(self, values):

        cursor = self.getCursor()

        sql="insert into student (name, age) values (%s,%s)"

        cursor.execute(sql, values)

        self.connection.commit()

        newid = cursor.lastrowid

        self.closeAll()

        return newid

    def getAll(self):

        # your code here

    def findByID(self, id):

#your code here

    def update(self, values):

#your code here

    def delete(self, id):

# your code here

studentDAO = StudentDAO()

## Test it

from zstudentDAO import studentDAO

#create

latestid = studentDAO.create(('mark', 45))

# find by id

result = studentDAO.findByID(latestid);

print (result)

#update

studentDAO.update(('Fred',21,latestid))

result = studentDAO.findByID(latestid);

print (result)

# get all

allStudents = studentDAO.getAll()

for student in allStudents:

  print(student)

# delete

studentDAO.delete(latestid)