NAME: SAHIL RAJESH SHRIWARDHANKAR

PRN: 2130331245504

BRANCH: COMPUTER ENGINEERING

SUBJECT: PYTHON PROGRAMMING

INDEX

|  |  |  |
| --- | --- | --- |
| Sr. No. | Title | Page No. |
| 1 | Executive Summary | 3 |
| 2 | Description | 4 |
| 3 | Code | 8 |
| 4 | Conclusion | 10 |

* Executive Summary

In this case study, we will see how to perform python database access.

We also will see how to create a database in Python 3 and perform operation like insert, read, update, delete, commit, rollback, and disconnect.

We will also see how to deal with errors in python database access. Moreover, we will discuss the PyMySQL and Installation and the benefits of Python Database access.

* Description

What is Python Database Access?

A database is a collection of tables related to each other via columns. For most real-world projects, a database is a must.

We can use SQL (Structured Query Language) to create, access, and manipulate data.

We can also make use of normalization to avoid redundancy of data.

For database programming, Python supports many database servers-

MySQL, Oracle, PostgreSQL, SQLite, Sybase, Microsoft SQL Server, mSQL, Microsoft Access, and many more.

It also supports Data Query Statements, Data Definition Language (DDL), and Data Manipulation Language (DML).

The standard database interface for Python is Python DB-API. For that, we have the module MySQLdb for MySQL.

This is independent of database engines; so we can write Python scripts to access any database engine. However, this isn’t compatible with Python 3.

So, in this Python Database Case Study, we use the module PyMySQL.

Advantages of Database Programming with Python

With Python, we have the following benefits:

Platform-independent

Faster and more efficient

Portable

Support for relational database systems

Easy to migrate and port database application interfaces

Support for SQL cursors

It handles open and closed connections

PyMySQL and Installation

PyMySQL implements the Python Database API 2.0. In this Python Database tutorial, we will use it to connect to a MySQL database server from Python.

We have the following requirements to install PyMySQL-

1. Python (any of)

CPython>=2.6 or >=3.3

PyPy>=4.0

IronPython 2.7

2. MySQL(any of)

MySQL>=4.1

MariaDB>=5.1

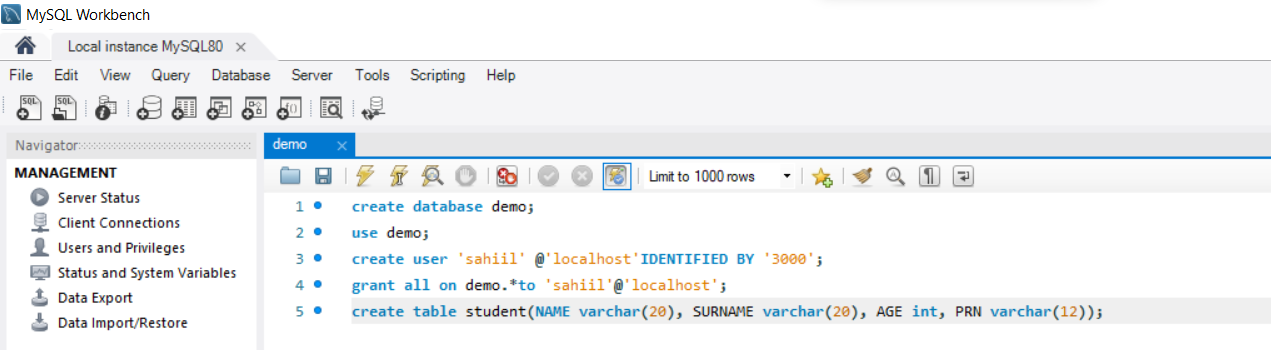
To install it, run the following command in the command prompt:-

C:\Users\sahiil>pip install PyMySQL

Also, make sure to install a database server on your machine. In this study, we use MySQL. We download it from here-

dev.mysql.com/downloads/mysql

1. Connecting Python Database



1. Connecting Database in Python

import mysql.connector

db=mysql.connector.connect(host="localhost",user="sahiil",passwd="3000",database=

"demo")

1. Creating Table in Python Database

cursor.execute("DROP TABLE IF EXISTS student")

query="""CREATE TABLE student(

NAME VARCHAR(20), Surname VARCHAR(20),

AGE INT, PRN VARCHAR(30))"""

cursor.execute(query)

Errors in Transactions

When holding a transaction, you may come across ten different kinds of errors:

1. Error

This is the base class for errors and a subclass to StandardError.

2. InterfaceError

This is a subclass to Error and Python uses it for errors relating to the module for database access.

3. DatabaseError

This is a subclass to Error and Python uses it for database errors.

4. OperationalError

This is a subclass of DatabaseError. When Python loses connection to a database, it throws this error.

This may happen when we haven’t selected a database.

5. DataError

This is a subclass of DatabaseError. Python uses this when there is an error in the data.

6. InternalError

This is a subclass of DatabaseError. Python uses this for errors internal to the module we use for the database access.

7. IntegrityError

Also a subclass of DatabaseError. Python uses this for cases where there can be damage to relational integrity.

This may happen when you try to enter duplicate records in the database.

8. ProgrammingError

This is a subclass of DatabaseError. Errors like bad table names cause this. This may happen when we try to create a duplicate database.

9. NotSupportedError

A subclass of DatabaseError. When we attempt to call functionality that it doesn’t support, Python raises this error.

10. Warning

This is a subclass of StandardError. Python uses this for non-fatal issues.

* Code

import mysql.connector

db=mysql.connector.connect(host="localhost",user="sahiil",passwd="3000",database=

"demo")

cursor=db.cursor()

cursor.execute("DROP TABLE IF EXISTS student") #This drops the table and replaces it

query="""CREATE TABLE student(

NAME VARCHAR(20), Surname VARCHAR(20),

AGE INT, PRN VARCHAR(30))"""

cursor.execute(query)

#----------------------------------------------------#

choice=input("ENTER OPERATION TO PERFORM, INSERT ,UPDATE ,DELETE ,EXIT----> ")

print()

while choice!="EXIT":

if choice=="INSERT":

#INSERTING VALUES

query="""INSERT INTO student(NAME, Surname, AGE, PRN)

VALUES(%s, %s, %s, %s )"""

val=int(input("PRESS 1 if you want to add more data or PRESS 0 to exit: "))

print()

while val!=0:

records\_to\_insert = [input("ENTER NAME: "),input("ENTER SURNAME: "),int(input("ENTER AGE: ")),input("ENTER PRN: ")]

print()

cursor.execute(query,records\_to\_insert)

val=int(input("PRESS 1 if you want to add more data or PRESS 0 to exit: "))

db.commit()

elif choice=="UPDATE":

#UPDATE

P=input("ENTER PRN of STUDENT---> ")

#changing Age

A=int(input("ENTER NEW AGE---> "))

query="update student set AGE={0} where PRN={1}".format(A,P)

cursor.execute(query)

db.commit()

elif choice=="DELETE":

#DELETE

P=int(input("ENTER PRN OF STUDENT TO BE DELETED: "))

query="delete from student where PRN={0}".format(P)

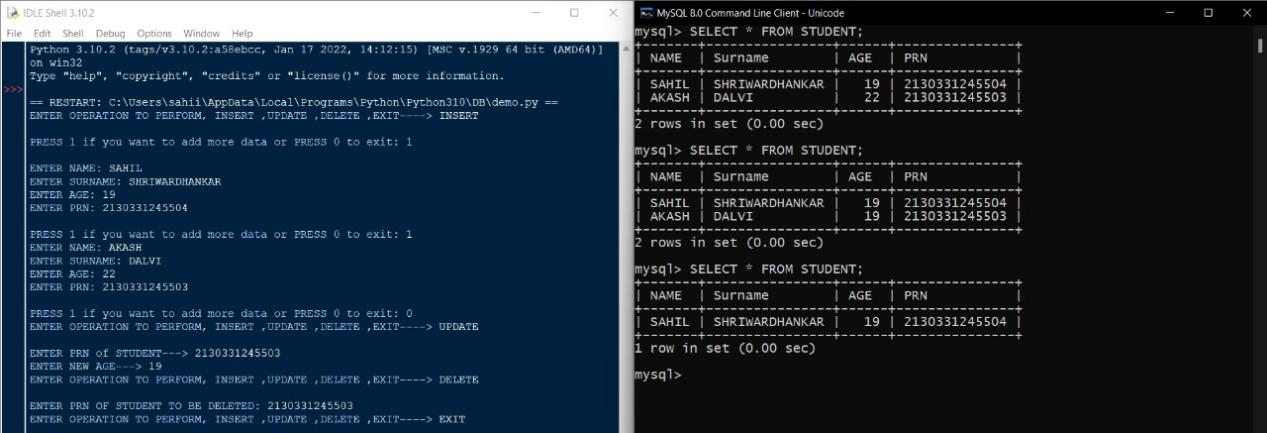
cursor.execute(query)

db.commit()

choice=input("ENTER OPERATION TO PERFORM, INSERT ,UPDATE ,DELETE ,EXIT----> ")

print()

db.close()

OutPut:-

* Conclusion

Hence, now you know how to access a database using Python with MySQL.

In addition, we saw how to perform Python database access and how to create a database in Python 3 and also perform operations like insert, update, delete, commit.

At last, we cover how to deal with errors in Python Database Access and PyMySQL and Installation and the benefits of Python database access.