3.6 Interface of Python with an SQL database (2/2)

Python code with SQL queries

As we know, the method cursor () is used to create a cursor instance/ object, after establishing the connection between Python and MySQL database. The database cursor is a control structure, which enables us to make traversal over the records / rows in a database. A cursor can be treated a pointer to the first row / tuple of the set of the records stored in the database. Like a file pointer which is pointing to current record / object of the file, moves automatically to next record and finally reaches the end of the file, the cursor points to the first row, then automatically travels to other rows of the database and finally reaches end of the table. Thus, the cursor facilitates retrieve, addition, updating and deletion of database records.

Once we have created a cursor instance /object, we can execute various types of SQL queries using the method execute() to manipulate the records of the database linked with the Python.

Let us consider the following relation RESULT is already created in the database "AECS", which is connected with the Python.

RollNo	Name	Class	Subject	DOE	Marks
1011	Ramesh	XII-A	Computer	14-03-2020	98
1013	Harsha	XII-B	Physics	12-03-2020	95
1016	Yogesh	XII-B	Chemistry	09-03-2020	96
1018	Girija	XII-C	Accountancy	14-03-2020	100
1014	Jevan	XII-A	Mathematics	18-03-2020	99
1015	Anju	XII-C	Economics	12-03-2020	94
1016	Kishore	XII-B	Biology	16-03-2020	96
1017	Kiran	XII-C	English	06-03-2020	91
1020	Kirutika	XII-A	Computer	14-03-2020	100
1012	Arvind	XII-B	Mathematics	18-03-2020	96

3.6 Interface of Python with an SQL database (2/2)

Python Code – 1

```
#Python code to retrieve all the rows from the relation RESULT
#Import the package mysql.connector
import mysql.connector as SqlCon

#Interface MySQL database with the Python
MYCon = SqlCon.connect(host="localhost", user="root", passwd="gms",
        database = "AECS")
if MyCon.is_connected() :
        print("MySQL database is successfully connected")
else:
        print("Error in connecting to MySQL database")

#Create cursor instance and execute SQL query
CurObj = MyCon.cursor()
CurObj.execute("select * from result")

DataRows = CurObj.fetchall()    #Fetch all the rows from the database
NumRows = CurObj.rowcount    #Find number of rows
print("Number of records retrieved from the table : ", NumRows);
#To display all the records
for row in DataRows:
        print (row)  #Display each record
MyCon.close()   #Terminate the connection
```

OUTPUT

```
MySQL database is successfully connected
Number of records retrieved from the table : 10
Then all the rows will be displayed
```

3.6 Interface of Python with an SQL database (2/2)

Python Code – 2

```
#Alternate Python code to retrieve one record at a time
#from the relation RESULT
#Import the package mysql.connector
import mysql.connector as SqlCon
MYCon = SqlCon.connect(host="localhost", user="root", passwd="gms",
        database = "AECS")
if MyCon.is_connected() :
        print("MySQL database is successfully connected")
else:
        print("Error in connecting to MySQL database")

#Create cursor instance and execute SQL query
CurObj = MyCon.cursor()
CurObj.execute("select * from result")

DataRow = CurObj.fetchone() # fetch each record
while  DataRow is not None:
        print (DataRow)
        DataRow = CurObj.fetchone()
NumRows = CurObj.rowcount
print("Number of records retrieved from the table : ", NumRows);
MyCon.close()
```

OUTPUT

MySQL database is successfully connected
First, all the rows will be displayed
Then,
Number of records retrieved from the table : 10 will be displayed

3.6 Interface of Python with an SQL database (2/2)

Python Code – 3

```
#Python code to retrieve all the record and display few columns
#from the relation RESULT
#Import the package mysql.connector
import mysql.connector
conn = mysql.connector.connect(host="localhost", user="root", passwd="gms",
        database = "AECS")
if conn.is_connected() :
        print("MySQL database is successfully connected")
else:
        print("Error in connecting to MySQL database")

#Create cursor instance and execute SQL query
cob = conn.cursor()
cob.execute("select * from result")

dr = cob.fetchall()    #Fetch all the rows from the database
print("Number of records retrieved from the table : ", cob.rowcount);

#To display all the records
for row in dr:
        rno =   row[0]        #Assign RollNo
        nam = row[1]        #Assign Name
        sub =  row[3]        #Assign Subject
        mark= row[5]        #Assign Marks
        print ('%-6d %-15s% %-15s %-6d'%(rno, nam, sub, mark))  #Display
conn.close()   #Terminate the connection
```

3.6 Interface of Python with an SQL database (2/2)

OUTPUT

MySQL database is successfully connected
First,
Number of records retrieved from the table : 10 will be displayed
Then,
RollNo, Name, Subject and Marks of each student will be displayed

Deleting rows from a relation

The Roll number of a student can be supplied through the keyboard while running the Python code. Further, a function definition **Delete_Rows(rno)** can also be used to delete a record of the student whose Roll number is passed as an argument to the function. The SQL command delete can be used with format as given below.

delete from result where rollno = '%d'

Here, '%d' represents an integer(rollno of the student), which is supplied as an argument to the command. After deleting rows from the table, we can save the changes in the database by using commit method and if there is an error, we can un-save the changes by using rollback method as depicted below.

conn.commit()

conn.rollback()

Python Code – 4

```
#Python code to delete a record / row from the relation RESULT
import  mysql.connector as sqcon

#Function to delete a row from the relation
def Delete_Rows(rno):
    conn = sqcon.connector.connect(host="localhost", user="root",
                passwd="gms", database = "AECS")
```

3.6 Interface of Python with an SQL database (2/2)

```
    if conn.is_connected() :
        print("MySQL database is successfully connected")
    else:
        print("Error in connecting to MySQL database")
    #Create cursor instance and execute SQL query
    cob = conn.cursor()
    sq = "delete from result where rno = '%d'"
    arg = (rno)
    try:
        cob.execute(sq%arg)
        cob.commit()        #save the changes in the database
        print("1 row deleted…")
    except:
        cob.rollback()      #un-save the changes in the database
    finally:
        cob.close()         #close the connection
        conn.close()
#__main__
rn = int(input('Enter roll number ='))
Delete_Rows(rn)   #call the function
```

OUTPUT

```
MySQL database is successfully connected
Enter roll number = 1012
1 row deleted…
```

3.6 Interface of Python with an SQL database (2/2)

Updating rows from a relation

We can modify the value(s) in the row(s) by using UPDATE-SET command. For instance, we wish to increase the marks scored in the subject Physics by 2 for all the students. Then, a function definition

Update_Rows(sub) can be used to update the marks. The SQL command UPDATE can be written with format as given below.

update result set marks = marks +2 where subject = '%s'

Here, '%s' represents a string (subject) supplied as argument to the command.

Python Code – 5

```
#Python code to update a record / row from the relation RESULT
import  mysql.connector as sqcon

#Function to update a row from the relation
def Update_Rows(sub):
    conn = sqcon.connector.connect(host="localhost", user="root",
            passwd="gms", database = "AECS")
    if conn.is_connected() :
        print("MySQL database is successfully connected")
    else:
        print("Error in connecting to MySQL database")

    #Create cursor instance and execute SQL query
    cob = conn.cursor()
    sq = "update result set marks = marks +2 where subject = '%s'"
    arg = (sub)
```

3.6 Interface of Python with an SQL database (2/2)

```
        try:
                cob.execute(sq%arg)
                cob.commit()        #save the changes in the database
                print("Marks in Physics updated…")
        except:
                cob.rollback()      #un-save the changes in the database
        finally:
                cob.close()         #close the connection
                conn.close()
#__main__
st = input('Enter Subject : ')
Update_Rows(st)  #call the function
```

OUTPUT

```
MySQL database is successfully connected
Enter Subject : Physics
Marks in Physics updated…
```

Creating database tables through Python

We have, so far, used the relation "RESULT", which was already created in the MySQL database. We can also create any relation and insert rows in the relation by using SQL commands in the Python code as illustrated below.

```
drop table  if exists result   #drop the table result if already exists
create table result(rollno int(4), name varchar2(20), class char(5),
                    subject varchar(15), doe date, marks int(3))
```

3.6 Interface of Python with an SQL database (2/2)

Python Code – 6

```
#Python code to create a new table result in MySQL database
import  mysql.connector as sqcon
#Function to create a new table result
def Create_Table():
    conn = sqcon.connector.connect(host="localhost", user="root",
            passwd="gms", database = "AECS")

    if conn.is_connected() :
        print("MySQL database is successfully connected")

    else:
        print("Error in connecting to MySQL database")
    #Create cursor instance and execute SQL query
    cob = conn.cursor()
    cob.execute(drop table  if exists result)
    sq = "create table result(rollno int(4), name varchar2(20), class
            char(5), subject varchar(15), doe date, marks int(3))"
    cob.execute(sq)
    print('Table created …')
    cob.close()        #close the connection
    conn.close()
#__MAIN__
Create_Table()
```

OUTPUT

```
MySQL database is successfully connected
Table created...
```

3.6 Interface of Python with an SQL database (2/2)

Inserting rows into a table

By using the SQL command INSERT-INTO-VALUES in the Python code as given below, we can insert rows into the table result.

Str =”insert into result (rollno, name, class, subject, doe, marks)
values(‘%d’,’%s’,’%s’,’%s’,’%d’)”

Python Code – 7

```
#Python code to insert rows into the table result in MySQL database
import  mysql.connector as sqcon
#Function to create a new table result
def Insert_Rows(rollno, name, class, subject, doe, marks):
    conn = sqcon.connector.connect(host=”localhost”, user=”root”,
            passwd=”gms”, database = “AECS”)
    if conn.is_connected() :
        print(“MySQL database is successfully connected”)
    else:
        print(“Error in connecting to MySQL database”)
    cob = conn.cursor()
    Str =”insert into result (rollno, name, class, subject, doe, marks)
            values(‘%d’,’%s’,’%s’,’%s’,’%d’)”
    args = (rollno, name, class, subject, doe, marks)
    try:
        cob.execute(Str%args)
        cob.commit()        #save the changes in the database
        print(“1 row inserted…”)
    except:
        cob.rollback()      #un-save the changes in the database
```

3.6 Interface of Python with an SQL database (2/2)

```
        finally:
                cob.close()          #close the connection
                conn.close()
#__MAIN__
n = int(input("How many rows to be inserted ? "))
for i in range(n):
        rn    = int(input('Enter rollno : '))
        nam = input('Enter name : ')
        cl     = input('Enter class : ')
        sub     = input('Enter subject : ')
        dt     = input('Enter doe : ')
        mks    = int(input('Enter marks : '))
        Insert_Rows(rn, nam, cl, sub, dt, mks)
        print("_____")
```

OUTPUT

```
MySQL database is successfully connected
How many rows to be inserted ? 1
Enter rollno : 1025
Enter name : Dinesh
Enter class : XII-C
Enter subject : Accountancy
Enter doe : 14-03-2020
Enter marks : 99
_____
1 row inserted …
```

0o0o0o0o0o0o0o0