----------MySQL----------

Q1. Describe with example how MySQL is used in python.

→

1.MySQL is a popular relational database management system, and Python provides a convenient way to interact with MySQL databases using the mysql-connector module.

2. Installation:

Before using MySQL in Python, you need to install the mysql-connector module. You can install it using pip:

pip install mysql-connector-python

3. Connecting to MySQL:

To connect to a MySQL database, use the mysql.connector.connect() method. Provide the necessary connection details like host, user, password, and database.

```
import mysql.connector

cn = mysql.connector.connect(
    host="localhost",
    user="username",
    password="password",
    database="database_name"
)
```

4. Creating a Cursor:

After establishing a connection, create a cursor object using cursor () method. The cursor is used to execute SQL queries.

```
cur = cn.cursor()
```

5. Executing Queries:

Use the execute() method of the cursor to run SQL queries. Fetch data using methods like fetchone() or fetchall().

```
cur.execute("SELECT * FROM table_name")

rows = cur.fetchall()

for row in rows:
    print(row)
```

6. Inserting Data:

Insert data into a table using the INSERT statement.

query = "INSERT INTO table_name (column1, column2) VALUES (%s, %s)"

values = ("value1", "value2")

cur.execute(query, values)

7.a. Updating Data:

Use the UPDATE statement to modify existing records.

query = "UPDATE table_name SET column1 = %s WHERE column2 = %s"

values = ("new_value", "condition_value")

cur.execute(query, values)

b.Committing Changes:

After executing queries that modify data, commit the changes to the database.

cn.commit()

8.Closing Connection:

Always close the database connection after performing operations.

cn.close()

Example:

1. Considering “Course” table with attribute (cno, cname, duration, fees). Read value from keyboard and insert into table using python programming.

→1.Code:

```
# Importing the MySQL connector module
import mysql.connector
# Establishing a database connection
cn = mysql.connector.connect(
   host="localhost",
   user="Kedar",
   password="1234567",
   database="School"
)
# Checking if the database connection is successful
```

```
if cn:
    print("Database is connected")
    cur = cn.cursor()  # The cursor is used to execute SQL queries.
    # Displaying data from the Course table before insertion
    print("Course table before inserting new record")
    cur.execute("SELECT * FROM Course")
    for row in cur:
        print(row)
    # Taking input from the user to insert a new record
    cno = input("Enter Course no")
    cname = input("Enter Course Name")
    duration = input("Enter Course Duration")
    fees = input("Enter Course fees")

    # SQL query to insert values into the Course table
    query = "INSERT INTO Course VALUES (%s, %s, %s, %s)"
    values = (cno, cname, duration, fees)
    cur.execute(query, values)
    # Displaying data from the Course table after insertion
    print("Course table After inserting new record")
    cur.execute("SELECT * FROM Course")
    for row in cur:
        print(row)
    cn.commit()  # Committing the changes to the database
else:
    print("Not connected")
# Closing the database connection
cn.close()
```

B. output:

```
Database is connected
Course table before inserting new record
(1, 'Btech Computer Science', 4, 60000)
(2, 'BCA', 3, 33000)
(3, 'Btech food technology', 2, 66000)
(4, 'Btech Agriculture', 4, 70000)
Enter Course no 5
Enter Course Name MCA
Enter Course Duration 2
Enter Course fees 40000
Course table After inserting new record
(1, 'Btech Computer Science', 4, 60000)
(2, 'BCA', 3, 33000)
(3, 'Btech food technology', 2, 66000)
(4, 'Btech Agriculture', 4, 70000)
(5, ' MCA', 2, 40000)
PS K:\program\PYTHON\classpro\database>
```

2. Considering “Studentfees” table with attributes (recno, recdate, rno, recamt). read value for “rno” from keyboard and display total fees and remaining fees using python programming.

   →code:

```
# Importing the MySQL connector module
import mysql.connector
# Establishing a database connection
cn = mysql.connector.connect(
    host="localhost",
    user="Kedar",
    password="1234567",
    database="School"
)
# Checking if the database connection is successful
if cn:
    print("Database is connected")
    cur = cn.cursor()  # The cursor is used to execute SQL queries.
    # Taking input from the user for roll number
    vrno = input("Enter roll no")

    # Fetching total fees for the given roll number from the Course table
    cur.execute("SELECT fees FROM course WHERE cno=(SELECT cno FROM student
WHERE rno=%s)", (vrno,))
    vcfees = cur.fetchone()[0]  # Store total fees for the Course
```

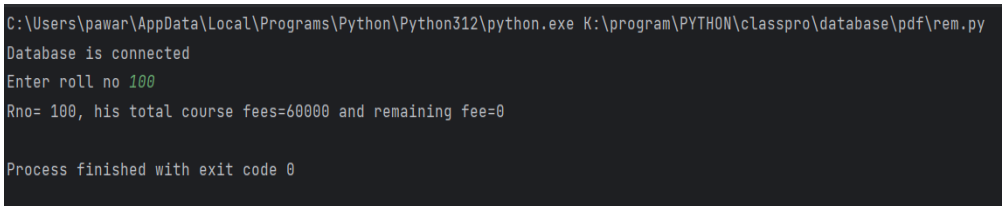
```
    # Fetching the sum of paid fees for the given roll number from the Studentfees
table
    cur.execute("SELECT SUM(recamt) FROM studentfees WHERE rno=%s", (vrno,))
    vpfees = cur.fetchone()[0]  # Store total paid fees

    # Calculating remaining fees
    vrfees = vcfees – vpfees

    # Displaying the results
    print(f"Rno={vrno}, his total course fees={vcfees} and remaining fee={vrfees}")
    cn.commit()  # Committing the changes to the database

else:
    print("Not connected")
# Closing the database connection
cn.close()
```

output:



Q2. Describe, with a proper example, how the procedure of MySQL is used in Python.
→
1.A stored procedure is a set of SQL statements stored in the database and executed as a single unit. To use a MySQL stored procedure in Python:

2.Establish a Database Connection:
Use the mysql.connector.connect method to establish a connection to the MySQL database.

3.Create a Cursor:
Create a cursor using cursor() to execute SQL queries.

Call the Stored Procedure:
Use the callproc method of the cursor to call the MySQL stored procedure, passing any required input parameters.

4.Fetch the Result:
Use the fetchall() or similar methods to retrieve the result of the stored procedure execution.

Commit Changes (Optional):
If modifications are made to the database (e.g., insertion, update), use commit() to save the changes.

5.Close the Connection:
Close the database connection using close().

6.Here's a concise example demonstrating the steps:

```
import mysql.connector
# Establishing a database connection
cn = mysql.connector.connect(
   host="localhost",
   user="your_username",
   password="your_password",
   database="your_database"
)
# Creating a cursor
cur = cn.cursor()
# Calling the stored procedure with input parameter
cur.callproc("YourStoredProcedureName", [param1, param2, ...])

 # Fetching the result from the stored procedure
 result = cur.fetchall()
 # Process the result as needed
```

```
# Committing changes and closing the connection
cn.commit()
cn.close()
```

example:

1.Example on how procedure can return value to python program.

→A. code:

```
#Example on how procedure can return value to python program.

#DELIMITER $$

#drop procedure pro_rem_fees;

#CREATE PROCEDURE pro_rem_fees(

#    IN rno INT,

#    OUT vrfees INT

#)

#BEGIN

#    DECLARE vcfees INT;

#    DECLARE vpfees INT;

 #   -- Use LIMIT 1 to ensure that the subquery returns only one row

 #   SELECT SUM(recamt) INTO vcfees FROM student_fees WHERE rno = rno LIMIT 1;

#    -- Use LIMIT 1 to ensure that the subquery returns only one row

#    SELECT fees INTO vpfees FROM course WHERE cno = (SELECT cno FROM student
WHERE rno = rno LIMIT 1);

 #   SET vrfees = vcfees - vpfees;

#END$$

#DELIMITER ;

# Importing the MySQL connector module

import mysql.connector


# Establishing a database connection

cn = mysql.connector.connect(

   host="localhost",
```

```
    user="Kedar",
    password="1234567",
    database="School"
)
# Checking if the database connection is successful
if cn:
    print("Database is connected")
    cur = cn.cursor()  # The cursor is used to execute SQL queries.
    # Taking input from the user
    vrno = int(input("Enter remaining roll no"))
    # Defining a list to pass parameters to the stored procedure
    prs = [vrno, 0]  # vrno at index 0 (passing value), 0 at index 1 (returning value)
    # Calling the stored procedure with input and output parameters
    res = cur.callproc("pro_rem_fees", prs)
    # Displaying the result returned by the stored procedure
    print(f"Rno={vrno} and remaining Fees={res[1]}")
    # Closing the cursor
    cur.close()
else:
    print("The database is not connected successfully")
# Closing the database connection
cn.close()
```

Output:

```
Database is connected
Enter remaining roll no 99
Rno=99 and remaining Fees=283800
```

2. Example on how to call stored procedure of MySQL to insert record in course table from python:

→ code:

```
#MySQL code:
#DELIMITER $$
#create procedure
#pro_ins( in cno int, in cname varchar(20), in duration int, in fees int)
#Begin
#    insert into Course values(cno, cname, duration, fees);
#END$$
#DELIMITER;

# Importing the MySQL connector module
import mysql.connector

# Establishing a database connection
cn = mysql.connector.connect(
   host="localhost",
   user="Kedar",
   password="1234567",
   database="school"
)

# Checking if the database connection is successful
if cn:
   print("connected")
   # Creating a cursor to execute SQL queries
   cur = cn.cursor()
   # Displaying data from the Course table before insertion
   print("Course table before inserting new record")
   cur.execute("SELECT * FROM Course")
   # Printing each row in the Course table
   for row in cur:
      print(row)
   # Taking input for new course details
   cno = input("Enter Course number")
   cname = input("Enter Course name")
   duration = input("Enter Course Duration")
   fees = input("Enter Course fees")
   # Creating a tuple with input values
   prs = (cno, cname, duration, fees)
   # Calling the stored procedure "pro_ins" with input parameters
   record = cur.callproc("pro_ins", prs)
   # Displaying data from the Course table after insertion
```

```
        print("Course table after inserting new record")
        cur.execute("SELECT * FROM Course")
        # Printing each row in the Course table after insertion
        for row in cur:
            print(row)
    else:
        print("not connected")
    # Closing the database connection
    cn.close()
```

OUTPUT:

```
connected
Course table before inserting new record
(1, 'Btech Computer Science', 4, 60000)
(2, 'BCA', 3, 33000)
(3, 'Btech food technology', 2, 66000)
(4, 'Btech Agriculture', 4, 70000)
Enter Course number 5
Enter Course name MCA
Enter Course Duration 2
Enter Course fees 40000
Course table after inserting new record
(1, 'Btech Computer Science', 4, 60000)
(2, 'BCA', 3, 33000)
(3, 'Btech food technology', 2, 66000)
(4, 'Btech Agriculture', 4, 70000)
(5, ' MCA', 2, 40000)
```