BLOOMING MINDS CENTRAL SCHOOL & JR COLLEGE (+1 & +2)

Class XII MPC Holidays Work

ENGLISH:

- Elucidate the poverty of Mukesh in lost spring and write possible measures to overcome poverty.

PHYSICS:

- Write Physics Record and Practice JEE Mains Q/A from 1 to 5 lessons.

CHEMISTRY:

- Complete the chemistry record work and solve Mains Questions from Material (Solutions, Electrochemistry).

MATHEMATICS:

- Write all formulas first and second year till where we complete and also read the formulae up to integration chapter.

PHYSICAL EDUCATION:

- Write Physical Education Record & 4th lesson Classwork.

COMPUTER SCIENCE:

- Complete the Record work.

BLOOMING MINDS CENTRAL SCHOOL & JR COLLEGE (+1 & +2)

Class XII BiPC Holidays Work

ENGLISH:

- Elucidate the poverty of Mukesh in lost spring and write possible measures to overcome poverty.

PHYSICS:

- Write Physics Record and Practice NEET Q/A from 1 to 5 lessons.

CHEMISTRY:

- Complete the chemistry record work and solve NEET Questions from Material (Solutions, Electrochemistry).

BOTANY:

- Collect the information about different kinds of levels and prepare herbarium.

ZOOLOGY:

- Write some applications of Biotechnology in Medicines in A4 Sheets.

PHYSICAL EDUCATION:

- Write Physical Education Record & 4^{th} lesson Classwork.

COMPUTER SCIENCE:

- Complete the Record work.

CONTENTS

<table>
<tr><th>SNo.</th><th>Practical</th><th>Date</th><th>Signature</th></tr>
<tr><td>8</td><td>Write a program to store customer data into a binary file cust.dat using a dictionary and print them on screen after reading them. The customer data contains ID as key, and name, city as values.</td><td></td><td></td></tr>
<tr><td>9</td><td>Write a program to update a record from student.dat file by its rollno. Display the updated record on screen.</td><td></td><td></td></tr>
<tr><td>10</td><td>Write a program to write data into binary file marks.dat and display the records of students who scored more than 95 marks.</td><td></td><td></td></tr>
<tr><td>11</td><td>Read a CSV file top5.csv and print the contents in a proper format. The data for top5.csv file are as following:
<table>
<tr><th>SNo</th><th>Batsman</th><th>Team</th><th>Runs</th><th>Highest</th></tr>
<tr><td>1</td><td>K L Rahul</td><td>KXI</td><td>670</td><td>132*</td></tr>
<tr><td>2</td><td>S Dhawan</td><td>DC</td><td>618</td><td>106*</td></tr>
<tr><td>3</td><td>David Warner</td><td>SRH</td><td>548</td><td>85*</td></tr>
<tr><td>4</td><td>Shreyas Iyer</td><td>DC</td><td>519</td><td>88*</td></tr>
<tr><td>5</td><td>Ishan Kishan</td><td>MI</td><td>516</td><td>99</td></tr>
</table></td><td></td><td></td></tr>
<tr><td>12</td><td>Read a CSV file top5.csv and print them with tab delimiter. Ignore first row header to print in tabular form.</td><td></td><td></td></tr>
<tr><td>13</td><td>Read a CSV file students.csv and print them with tab delimiter. Ignore first row header to print in tabular form.
<table>
<tr><th>Field 1</th><th>Data Type</th></tr>
<tr><td>StudentID</td><td>Integer</td></tr>
<tr><td>StudentName</td><td>String</td></tr>
<tr><td>Score</td><td>Integer</td></tr>
</table></td><td></td><td></td></tr>
</table>

SNo.	Practical	Date	Signature
3	**Data Structure - Stack**		
14	Write a program to implement a stack for the employee details (empno, name).		
15	Write a python program to check whether a string is a palindrome or not using stack.		
4	**Database management (MySQL Queries)**		
16	Queries Set 1 (Fetching records)		
17	Queries Set 2 (Based on Aggregate Functions)		
18	Queries Set 3 (DDL Commands)		
19	Queries set 4 (Based on Two Tables)		
20	Queries Set 5 (Group by , Order By)		
5	**Python-MySQL Connectivity**		

21	Write a MySQL connectivity program in Python to
	• Create a database school
	• Create a table students with the specifications - ROLLNO integer, STNAME character(10) in MySQL and perform the following operations:
	  ○ Insert two records in it
	  ○ Display the contents of the table		

22	Perform all the operations with reference to table 'students' through MySQL-Python connectivity.		
23	Write a menu-driven program to store data into a MySQL database named shop and table customer as following: 1. Add customer details 2. Update customer details 3. Delete customer details 4. Display all customer details		
24	Modify the above program and display the customer details based on the following menu: 1. Display customer details by city 2. Display customer details by bill amount 3. Display customer details by name 4. Display customer details by category		

PART 1 WORKING WITH FUNCTIONS

1. **Write a python program using a function to print factorial number series from n to m numbers.**

```
def facto():
  n=int(input("Enter the number:"))
  f=1
  for i in range(1,n+1):
    f*=i
  print("Factorial of ",n, "is: ",f, end=" ")
facto()
```

```
========== RESTART:
Enter the number:5
Factorial of  5 is:  120
>>>
```

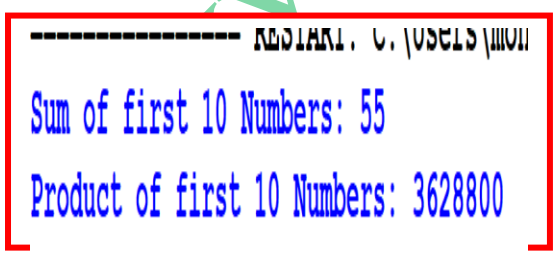
2. **Write a python program to accept username "Admin" as default argument and password 123 entered by user to allow login into the system.**

```
def user_pass(password,username="Admin"):
  if password=='123':
    print("You have logged into system")
  else:
    print("Password is incorrect!!!!!!")
password=input("Enter the password:")
user_pass(password)
```

```
Enter the password:123
You have logged into system
>>>
========== RESTART: C:
Enter the password:hello123
Password is incorrect!!!!!!
```

3. Write a python program to demonstrate the concept of variable length argument to calculate product and power of the first 10 numbers.

```
def sum10(*n):
  total=0
  for i in n:
    total=total + i
  print("Sum of first 10 Numbers:",total)
sum10(1,2,3,4,5,6,7,8,9,10)
def product10(*n):
  pr=1
  for i in n:
    pr=pr * i
  print("Product of first 10 Numbers:",pr)
product10(1,2,3,4,5,6,7,8,9,10)
```

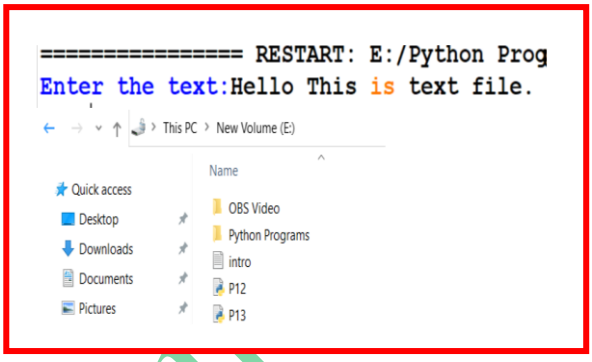


PART 2 DATA FILE HANDLING

4. Create a text file "intro.txt" in python and ask the user to write a single line text by user input.

```
def program4():
  f = open("intro.txt","w")
  text=input("Enter the text:")
  f.write(text)
  f.close(
program4()
```

```
================ RESTART: E:/Python Prog
Enter the text:Hello This is text file.
```



5. Write a program to count a total number of lines and count the total number of lines starting with 'A', 'B', and 'C' from the file myfile.txt.

```
def program5():
  with open("MyFile.txt","r") as f1:
    data=f1.readlines()
    cnt_lines=0
    cnt_A=0
    cnt_B=0
    cnt_C=0
    for lines in data:
      cnt_lines+=1
      if lines[0]=='A':
        cnt_A+=1
      if lines[0]=='B':
        cnt_B+=1
      if lines[0]=='C':
        cnt_C+=1
  print("Total Number of lines are:",cnt_lines)
  print("Total Number of lines strating with A are:",cnt_A)
  print("Total Number of lines strating with B are:",cnt_B)
  print("Total Number of lines strating with C are:",cnt_C)
program5()
```

```
File Contents:
Python is super and trending language.
Allows to store the output in the files.
A text file stores textual data.
Binary files can handle binary data.
Binary files use pickle module to store data.
CSV files can handle tabular data.
CSV files can be read easily using CSV reader object.
Total Number of lines are: 7
Total Number of lines strating with A are: 2
Total Number of lines strating with B are: 2
Total Number of lines strating with C are: 2
```

6. Write a program to replace all spaces from text with - (dash) from the file intro.txt.

```
def program6():
  cnt = 0
  with open("intro.txt","r") as f1:
    data = f1.read()
    data=data.replace(' ','-')
  with open("intro.txt","w") as f1:
    f1.write(data)
  with open("intro.txt","r") as f1:
    print(f1.read())
program6()
```

```
City,State
Ahmedabad,Gujarat
Mumbai,Maharashtra
Indore,Madhya-Pradesh
```

7. Write a program to know the cursor position and print the text according to the below-given specifications:

a. Print the initial position

b. Move the cursor to 4th position

c. Display next 5 characters

d. Move cursor to next 10 characters

e. Print the current cursor position

f. Print next 10 characters from current position

```
def program7():
  f = open("intro.txt","r")
  print("Cusror initial position.")
  print(f.tell())
  f.seek(4,0)
  print("Displaying values from 5th position.")
  print(f.read(5))
  f.seek(10,0)
  print(f.tell())
  print("Print cursor's current postion")
  print(f.seek(7,0))
  print("Displaying next 10 characters from cursor's current postion.")
  print(f.read(10))
program7()
```

```
Cusror initial position.
0
Displaying values from 5th position.
,Stat
10
Print cursor's current postion
7
Displaying next 10 characters from cursor's current postion.
ate
Ahmeda
```

8. Create a binary file client.dat to hold records like ClientID, Client name, and Address using the dictionary. Write functions to write data, read them, and print on the screen.

```
import pickle
rec={}
def file_create():
  f=open("client.dat","wb")
  cno = int(input("Enter Client ID:"))
  cname = input("Enter Client Name:")
  address = input("Enter Address:")
  rec={cno:[cname,address]}
  pickle.dump(rec,f)
def read_data():
  f = open("client.dat","rb")
  print("*"*78)
  print("Data stored in File....")
  rec=pickle.load(f)
  for i in rec:
    print(rec[i])
file_create()
read_data()
```

```
Enter Client ID:101
Enter Client Name:Shailesh
Enter Address:Ahmedabad
******************************
Data stored in File....
['Shailesh', 'Ahmedabad']
```

9. **Write a program to update a record from student.dat file by its rollno. Display the updated record on screen.**

```
import pickle as p
rec=[]
found=0
f=open("student.dat","rb+")
r=p.load(f)
print("Existing Record:",r)
ro=int(input("Enter roll no to update:"))
for i in r:
  if ro==i[0]:
    i[1]=input("Enter new name:")
    found=1
    break
if found==0:
  print("record not found..")
else:
  f.seek(0)
  p.dump(r,f)
  print('Record Updated...')
f.close()

f1=open('student.dat','rb+')
r=p.load(f1)
print("Updated Record:",r)
f1.close()
```

```
Existing Record: [[101, 'Hemu']]
Enter roll no to update:101
Enter new name:Vikas
Record Updated...
Updated Record: [[101, 'Vikas']]
```

10. Write a program to write data into binary file marks.dat and display the records of students who scored more than 95 marks.

```
import pickle
def search_95plus():
 f = open("marks.dat","ab")
 while True:
  rn=int(input("Enter the rollno:"))
  sname=input("Enter the name:")
  marks=int(input("Enter the marks:"))
  rec=[]
  data=[rn,sname,marks]
  rec.append(data)
  pickle.dump(rec,f)
  ch=input("Wnat more records?Yes:")
  if ch.lower() not in 'yes':
   break
 f.close()
 f = open("marks.dat","rb")
 cnt=0
 try:
  while True:
   data = pickle.load(f)
   for s in data:
    if s[2]>95:
     cnt+=1
     print("Record:",cnt)
     print("RollNO:",s[0])
     print("Name:",s[1])
     print("Marks:",s[2])
 except Exception:
  f.close()
search_95plus()
```

```
Enter the rollno:1
Enter the name:Aryan
Enter the marks:85
Wnat more records?Yes:y
Enter the rollno:2
Enter the name:Debanjan
Enter the marks:96
Wnat more records?Yes:y
Enter the rollno:3
Enter the name:Prakash
Enter the marks:98
Wnat more records?Yes:y
Enter the rollno:4
Enter the name:Priya
Enter the marks:92
Wnat more records?Yes:n
Record: 1
RollNO: 2
Name: Debanjan
Marks: 96
Record: 2
RollNO: 3
Name: Prakash
Marks: 98
```

11. Read a CSV file top5.csv and print the contents in a proper format. The data for top5.csv file are as following:

SNo	Batsman	Team	Runs	Highest
1	K L Rahul	KXI	670	132*
2	S Dhawan	DC	618	106*
3	David Warner	SRH	548	85*
4	Shreyas Iyer	DC	519	88*
5	Ishan Kishan	MI	516	99

```
from csv import reader
def pro1():
  with open("top5.csv","r") as f:
    d = reader(f)
    data=list(d)
    for i in data:
      print(i)
pro1()
```

```
========== RESTART:
['SNo', 'Batsman', 'Team', 'Runs', 'Highest']
['1', 'K L Rahul', 'KXI', '670', '132*']
['2', 'S Dhawan', 'DC', '618', '106*']
['3', 'David Warner', 'SRH', '548', '85*']
['4', 'Shreyas Iyer', 'DC', '519', '88*']
['5', 'Ishan Kishan', 'MI', '516', '99']
```

12.Read a CSV file top5.csv and print them with tab delimiter. Ignore first row header to print in tabular form.

```
from csv import reader
def pro12():
  f = open("e:\\top5.csv","r")
  dt = reader(f,delimiter=',')
  headr_row=next(dt)
  data = list(dt)
  f.close()
  for i in data:
    for j in i:
      print(j,"\t",end=" ")
    print()
pro12()
```

```
1       K L Rahul       KXI     670     132*
2       S Dhawan        DC      618     106*
3       David Warner    SRH     548     85*
4       Shreyas Iyer    DC      519     88*
5       Ishan Kishan    MI      516     99
```

13. Read a CSV file students.csv and print them with tab delimiter. Ignore first row header to print in tabular form.

Field 1	*Data Type*
StudentID	Integer
StudentName	String
Score	Integer

```
from csv import writer
def pro13():
  #Create Header First
  f = open("result.csv","w",newline='\n')
  dt = writer(f)
  dt.writerow(['Student_ID','StudentName','Score'])
  f.close()
  #Insert Data
  f = open("result.csv","a",newline='\n')
  while True:
    st_id= int(input("Enter Student ID:"))
    st_name = input("Enter Student name:")
    st_score = input("Enter score:")
    dt = writer(f)
    dt.writerow([st_id,st_name,st_score])
    ch=input("Want to insert More records?(y or Y):")
    ch=ch.lower()
    if ch !='y':
      break
  print("Record has been added.")
  f.close()
pro13()
```

```
Enter Student ID:1
Enter Student name:Shekhar
Enter score:52
Want to insert More records?(y or Y):y
Enter Student ID:2
Enter Student name:Manoj
Enter score:85
Want to insert More records?(y or Y):y
Enter Student ID:2
Enter Student name:Swadesh
Enter score:58
Want to insert More records?(y or Y):n
Record has been added.
>>>
```

PART 3 DATA STRUCTURE

14. Write a menu-driven python program to implement stack operation.

```
def check_stack_isEmpty(stk):
    if stk==[]:
        return True
    else:
        return False
# An empty list to store stack elements, initially empty
s=[]
top = None # This is top pointer for push and pop

def main_menu():
    while True:
        print("Stack Implementation")
        print("1 - Push")
        print("2 - Pop")
        print("3 - Peek")
        print("4 - Display")
        print("5 - Exit")
        ch = int(input("Enter the your choice:"))
        if ch==1:
            el = int(input("Enter the value to push an element:"))
            push(s,el)
        elif ch==2:
            e=pop_stack(s)
            if e=="UnderFlow":
                print("Stack is underflow!")
            else:
```

```
            print("Element popped:",e)
        elif ch==3:
            e=pop_stack(s)
            if e=="UnderFlow":
                print("Stack is underflow!")
            else:
                print("The element on top is:",e)
        elif ch==4:
            display(s)
        elif ch==5:
            break
        else:
            print("Sorry,invalid option")

def push(stk,e):
    stk.append(e)
    top = len(stk)-1

def display(stk):
    if check_stack_isEmpty(stk):
        print("Stack is Empty")
    else:
        top = len(stk)-1
        print(stk[top],"-Top")
        for i in range(top-1,-1,-1):
            print(stk[i])

def pop_stack(stk):
    if check_stack_isEmpty(stk):
        return "UnderFlow"
```

```
Stack Implementation
1 - Push
2 - Pop
3 - Peek
4 - Display
5 - Exit
Enter the your choice:1
Enter the value to push an element:74
Stack Implementation
1 - Push
2 - Pop
3 - Peek
4 - Display
5 - Exit
Enter the your choice:1
Enter the value to push an element:83
Stack Implementation
1 - Push
2 - Pop
3 - Peek
4 - Display
5 - Exit
Enter the your choice:4
83 -Top
74
Stack Implementation
1 - Push
2 - Pop
3 - Peek
4 - Display
5 - Exit
Enter the your choice:2
Element popped: 83
Stack Implementation
1 - Push
2 - Pop
3 - Peek
4 - Display
5 - Exit
Enter the your choice:3
The element on top is: 74
Stack Implementation
1 - Push
2 - Pop
3 - Peek
4 - Display
5 - Exit
Enter the your choice:5
```

```
    else:
        e = stk.pop()
        if len(stk)==0:
            top = None
        else:
            top = len(stk)-1
        return e
def peek(stk):
    if check_stack_isEmpty(stk):
        return "UnderFlow"
    else:
        top = len(stk)-1
        return stk[top]
```

15.Write a program to implement a stack for the employee details (empno, name).

```
stk=[]
top=-1
def line():
  print('~'*100)
def isEmpty():
  global stk
  if stk==[]:
    print("Stack is empty!!!")
  else:
    None
def push():
  global stk
  global top
```

```
  empno=int(input("Enter the employee number to push:"))
  ename=input("Enter the employee name to push:")
  stk.append([empno,ename])
  top=len(stk)-1
def display():
  global stk
  global top
  if top==-1:
    isEmpty()
  else:
    top=len(stk)-1
    print(stk[top],"<-top")
    for i in range(top-1,-1,-1):
      print(stk[i])
def pop_ele():
  global stk
  global top
  if top==-1:
    isEmpty()
  else:
    stk.pop()
    top=top-1
def main():
  while True:
    line()
    print("1. Push")
    print("2. Pop")
    print("3. Display")
```

```
**********
1. Push
2. Pop
3. Display
4. Exit
Enter your choice:1
Enter the employee number to push:32
Enter the employee name to push:Sagar
Element Pushed
**********
1. Push
2. Pop
3. Display
4. Exit
Enter your choice:1
Enter the employee number to push:23
Enter the employee name to push:Minesh
Element Pushed
**********
1. Push
2. Pop
3. Display
4. Exit
Enter your choice:3
[23, 'Minesh'] <-top
[32, 'Sagar']
**********
1. Push
2. Pop
3. Display
4. Exit
Enter your choice:2
**********
1. Push
2. Pop
3. Display
4. Exit
Enter your choice:4
```

```
print("4. Exit")
ch=int(input("Enter your choice:"))
if ch==1:nm
  push()
  print("Element Pushed")
elif ch==2:
  pop_ele()
elif ch==3:
  display()
elif ch==4:
  break
else:
  print("Invalid Choice")
```

16. Write a python program to check whether a string is a palindrome or not using stack.

```
stack = []
top = -1
# push function
def push(ele):
        global top
        top += 1
        stack[top] = ele
# pop function
def pop():
        global top
        ele = stack[top]
        top -= 1
        return ele
```

```
# Function that returns 1 if string is a palindrome
def isPalindrome(string):
        global stack
        length = len(string)
        # Allocating the memory for the stack
        stack = ['0'] * (length + 1)
        # Finding the mid
        mid = length // 2
        i = 0
        while i < mid:
                push(string[i])
                i += 1
        # Checking if the length of the string is odd then neglect the middle character
        if length % 2 != 0:
                i += 1
        # While not the end of the string
        while i < length:
                ele = pop()
                # If the characters differ then the given string is not a palindrome
                if ele != string[i]:
                        return False
                i += 1
        return True
string = input("Enter string to check:")
if isPalindrome(string):
      print("Yes, the string is a palindrome")
else:
   print("No, the string is not a palindrome")
```

```
>>>
========== RESTART: E:/Python Programs/stack.py ==========
Enter string to check:nitin
Yes, the string is a palindrome
>>>
========== RESTART: E:/Python Programs/stack.py ==========
Enter string to check:Sanjay
No, the string is not a palindrome
```

PART 4 SQL QUERIES

1. Consider the following MOVIE table and write the SQL queries based on it.

Movie_ID	MovieName	Type	ReleaseDate	ProductionCost	BusinessCost
M001	Dahek	Action	2022/01/26	1245000	1300000
M002	Attack	Action	2022/01/28	1120000	1250000
M003	Looop Lapeta	Thriller	2022/02/01	250000	300000
M004	Badhai Do	Drama	2022/02/04	720000	68000
M005	Shabaash Mithu	Biography	2022/02/04	1000000	800000
M006	Gehraiyaan	Romance	2022/02/11	150000	120000

a) Display all information from movie.

b) Display the type of movies.

c) Display movieid, moviename, total_eraning by showing the business done by the movies. Claculate the business done by movie using the sum of productioncost and businesscost.

d) Display movieid, moviename and productioncost for all movies with productioncost greater thatn 150000 and less than 1000000.

e) Display the movie of type action and romance.

f) Display the list of movies which are going to release in February, 2022.

Answers:

a) `select * from movie;`

```
mysql> select * from movie;
+----------+------------------+-----------+-------------+----------------+--------------+
| Movie_id | moviename        | type      | releasedate | productioncost | businesscost |
+----------+------------------+-----------+-------------+----------------+--------------+
| M001     | The Kashmir Files | Action    | 2022-01-26  |        1245000 |      1300000 |
| M002     | Attack           | Action    | 2022-01-28  |        1120000 |      1250000 |
| M003     | Loop Lapeta      | Thriller  | 2022-02-01  |         250000 |       300000 |
| M004     | Bdhai Do         | Drama     | 2022-02-04  |         720000 |      4800000 |
| M005     | Shabaash Mothu   | Biography | 2022-02-04  |        1000000 |       800000 |
| M006     | Gehriyaan        | Romance   | 2022-02-11  |         150000 |       120000 |
+----------+------------------+-----------+-------------+----------------+--------------+
6 rows in set (0.01 sec)
```

b) select distinct from a movie;

```
mysql> select distinct type from movie;
+-------------+
| type        |
+-------------+
| Action      |
| Thriller    |
| Drama       |
| Biography   |
| Romance     |
+-------------+
5 rows in set (0.00 sec)
```

c) select movieid, moviename, productioncost + businesscost "total earning" from movie;

```
mysql> select movie_id, moviename, productioncost + businesscost "total earning" from movie;
+----------+------------------+--------------+
| movie_id | moviename        | total earning |
+----------+------------------+--------------+
| M001     | The Kashmir Files |      2545000 |
| M002     | Attack           |      2370000 |
| M003     | Loop Lapeta      |       550000 |
| M004     | Bdhai Do         |      5520000 |
| M005     | Shabaash Mothu   |      1800000 |
| M006     | Gehriyaan        |       270000 |
+----------+------------------+--------------+
6 rows in set (0.03 sec)
```

d) select movie_id,moviename, productioncost from movie where producst is >150000 and <1000000;

```
mysql> select movie_id,moviename, productioncost from movie where productioncost >150000 and productioncost<100000
+----------+-----------+--------------+
| movie_id | moviename | productioncost |
+----------+-----------+--------------+
| M003     | Loop Lapeta |       250000 |
| M004     | Bdhai Do  |       720000 |
+----------+-----------+--------------+
2 rows in set (0.03 sec)
```

e) select moviename from movie where type ='action' or type='romance';

```
mysql>  select moviename from movie where type ='action' or type='romance'
+------------------+
| moviename        |
+------------------+
| The Kashmir Files |
| Attack           |
| Gehriyaan        |
+------------------+
3 rows in set (0.00 sec)
```

f) `select moviename from movie where month(releasedate)=2;`

```
mysql> select moviename from movie where month(releasedate)=2;
+----------------+
| moviename      |
+----------------+
| Loop Lapeta    |
| Bdhai Do       |
| Shabaash Mothu |
| Gehriyaan      |
+----------------+
4 rows in set (0.00 sec)
```

2. Consider the given table patient and Write following queries:

pid	pname	age	department	dateofadm	charges	gender
1	Dipak	34	Cardiac	2021-12-02	550.00	M
2	Divya	15	Women	2021-11-20	2640.00	F
3	Harshil	22	Men	2021-10-19	2750.00	M
4	Suman	15	Women	2021-09-11	3080.00	F
5	Anurag	34	Orthopadic	2021-11-11	880.00	M
6	Usha	41	Cardiac	2021-05-20	2090.00	F
7	Manish	42	Men	2021-06-18	1430.00	M
8	Vishruti	15	Women	2021-07-06	3190.00	F
9	Sharad	32	Orthopadic	NULL	NULL	NULL
10	Aksh	25	cardiac	2022-05-20	3500.00	M

a) Display the total charges of patient admitted in the month of November.

b) Display the eldest patient with name and age.

c) Count the unique departments.

d) Display an average charges.

Answers:

a) `select sum(charges) from patient`
`where dateofadm like '%-11-%';`

```
+--------------+
| sum(charges) |
+--------------+
|      3520.00 |
+--------------+
1 row in set (0.00 sec)
```

b) `select pname,max(age) from patient;`

```
+-------+----------+
| pname | max(age) |
+-------+----------+
| Dipak |       42 |
+-------+----------+
1 row in set (0.01 sec)
```

c) `Select count(distinct department) from patient;`

```
+----------------------------+
| count(distinct department) |
+----------------------------+
|                          4 |
+----------------------------+
1 row in set (0.00 sec)
```

d) `Select avg(charges) from patient;`

```
+--------------+
| avg(charges) |
+--------------+
|  2234.444444 |
+--------------+
1 row in set (0.00 sec)
```

3. Suppose your school management has decided to conduct cricket matches between students of Class XI and Class XII. Students of each class are asked to join any one of the four teams – Team Titan, Team Rockers, Team Magnet and Team Hurricane. During summer vacations, various matches will be conducted between these teams. Help your sports teacher to do the following:

a) Create a database "Sports".

b) Create a table "TEAM" with following considerations:

   a. It should have a column TeamID for storing an integer value between 1 to 9, which refers to unique identification of a team.

   b. Each TeamID should have its associated name (TeamName), which should be a string of length not less than 10 characters.

   c. Using table level constraint, make TeamID as the primary key.

c) Show the structure of the table TEAM using a SQL statement.

d) As per the preferences of the students four teams were formed as given below. Insert these four rows in TEAM table:

   a. Row 1: (1, Tehlka)
   b. Row 2: (2, Toofan)
   c. Row 3: (3, Aandhi)
   d. Row 3: (4, Shailab)

e) Show the contents of the table TEAM using a DML statement.

f) Now create another table MATCH_DETAILS and insert data as shown below. Choose appropriate data types and constraints for each attribute.

MatchID	MatchDate	FirstTeamID	SecondTeamID	FirstTeamScore	SecondTeamScore
M1	2021/12/20	1	2	107	93
M2	2021/12/21	3	4	156	158
M3	2021/12/22	1	3	86	81
M4	2021/12/23	2	4	65	67
M5	2021/12/24	1	4	52	88
M6	2021/12/25	2	3	97	68

Answers:

a) `create database sports;`

```
mysql> create database sports
    -> ;
Query OK, 1 row affected (0.01 sec)

mysql> use sports;
Database changed
mysql>
```

b) Creating table with the given specification

```
create table team
(teamid int(1),
teamname varchar(10), primary key(teamid));
```

c) desc team;

```
mysql> desc team;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| teamid   | int         | NO   | PRI | NULL    |       |
| teamname | varchar(10) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

- **Inserting data:**

mqsql> insert into team values(1,'Tehlka');

```
mysql> insert into team
    -> values(1,'Tehlka');
Query OK, 1 row affected (0.01 sec)

mysql> insert into team
    -> values(2,'Toofan');
Query OK, 1 row affected (0.01 sec)

mysql> insert into team
    -> values(3,'Aandhi');
Query OK, 1 row affected (0.03 sec)

mysql> insert into team
    -> values(4,'Shailab');
Query OK, 1 row affected (0.03 sec)
```

- **Show the content of table - team:**

select * from team;

```
mysql> select * from team;
+--------+----------+
| teamid | teamname |
+--------+----------+
|      1 | Tehlka   |
|      2 | Toofan   |
|      3 | Aandhi   |
|      4 | Shailab  |
+--------+----------+
4 rows in set (0.00 sec)
```

- **Creating another table:**

```
create table match_details
-> (matchid varchar(2) primary key,
-> matchdate date,
-> firstteamid int(1) references team(teamid),
-> secondteamid int(1) references team(teamid),
-> firstteamscore int(3),
-> secondteamscore int(3));
```

```
mysql> create table match_details
    -> (matchid varchar(2) primary key,
    -> matchdate date,
    -> firstteamid int(1) references team(teamid),
    -> secondteamid int(1) references team(teamid),
    -> firstteamscore int(3),
    -> secondteamscore int(3));
Query OK, 0 rows affected, 4 warnings (0.03 sec)
```

```
mysql> select * from match_details;
+---------+------------+-------------+--------------+----------------+-----------------+
| matchid | matchdate  | firstteamid | secondteamid | firstteamscore | secondteamscore |
+---------+------------+-------------+--------------+----------------+-----------------+
| M1      | 2021-12-20 |           1 |            2 |            107 |              93 |
| M2      | 2021-12-21 |           3 |            4 |            156 |             158 |
| M3      | 2021-12-22 |           1 |            3 |             86 |              81 |
| M4      | 2021-12-23 |           2 |            4 |             65 |              67 |
| M5      | 2021-12-24 |           1 |            4 |             52 |              88 |
| M6      | 2021-12-25 |           2 |            3 |             97 |              68 |
+---------+------------+-------------+--------------+----------------+-----------------+
6 rows in set (0.01 sec)
```

4. Write following queries:
   a) Display the matchid, teamid, teamscore whoscored more than 70 in first ining along with team name.
   b) Display matchid, teamname and secondteamscore between 100 to 160.
   c) Display matchid, teamnames along with matchdates.
   d) Display unique team names
   e) Display matchid and matchdate played by Anadhi and Shailab.

Answers:

a) `select match_details.matchid, match_details.firstteamid, team.teamname,match_details.firstteamscore from match_details, team where match_details.firstteamid = team.teamid and match_details.first`

```
mysql> select * from match_details;
+---------+------------+-------------+--------------+---------------+-----------------+
| matchid | matchdate  | firstteamid | secondteamid | firstteamscore | secondteamscore |
+---------+------------+-------------+--------------+---------------+-----------------+
| M1      | 2021-12-20 |           1 |            2 |           107 |              93 |
| M2      | 2021-12-21 |           3 |            4 |           156 |             158 |
| M3      | 2021-12-22 |           1 |            3 |            86 |              81 |
| M4      | 2021-12-23 |           2 |            4 |            65 |              67 |
| M5      | 2021-12-24 |           1 |            4 |            52 |              88 |
| M6      | 2021-12-25 |           2 |            3 |            97 |              68 |
+---------+------------+-------------+--------------+---------------+-----------------+
6 rows in set (0.01 sec)
```

b) `select match_details.matchid, match_details.firstteamid, team.teamname,match_details.firstteamscore from match_details, team where match_details.firstteamid = team.teamid and match_details.firstteamscore>70;`

```
mysql> select matchid, teamname, secondteamscore from match_details, team where match_details.secondteamid=team.teamid
 and match_details.secondteamscore between 100 and 160;
+---------+----------+-----------------+
| matchid | teamname | secondteamscore |
+---------+----------+-----------------+
| M2      | Shailab  |             158 |
+---------+----------+-----------------+
1 row in set (0.00 sec)
```

c) select matchid, teamname, firstteamid, secondteamid, matchdate from match_details, team where match_details.firstteamid = team.teamid;

```
mysql> select matchid,teamname,firstteamid,secondteamid,matchdate from match_details, team where match_details.firstte
amid=team.teamid;
+---------+----------+------------+-------------+------------+
| matchid | teamname | firstteamid | secondteamid | matchdate  |
+---------+----------+------------+-------------+------------+
| M1      | Tehlka   |          1 |           2 | 2021-12-20 |
| M2      | Aandhi   |          3 |           4 | 2021-12-21 |
| M3      | Tehlka   |          1 |           3 | 2021-12-22 |
| M4      | Toofan   |          2 |           4 | 2021-12-23 |
| M5      | Tehlka   |          1 |           4 | 2021-12-24 |
| M6      | Toofan   |          2 |           3 | 2021-12-25 |
+---------+----------+------------+-------------+------------+
6 rows in set (0.00 sec)
```

d) select distinct(teamname) from match_details, team where match_details.firstteamid = team.teamid;

```
mysql> select distinct(teamname) from match_details, team where match_details.firstteamid=team.teamid;
+----------+
| teamname |
+----------+
| Tehlka   |
| Aandhi   |
| Shailab  |
| Toofan   |
+----------+
4 rows in set (0.03 sec)
```

e) select matchid,matchdate from match_details, team where match_details.firstteamid = team.teamid and team.teamname in ('Aandhi','Shailab');

```
mysql> select matchid,matchdate from match_details, team where match_details.firstteamid=team.teamid and team.teamname
 in ('Aandhi','Shailab');
+---------+------------+
| matchid | matchdate  |
+---------+------------+
| M2      | 2021-12-21 |
| M4      | 2021-12-23 |
+---------+------------+
2 rows in set (0.00 sec)
```

5. Consider the following table and write the queries:

itemno	Item	dcode	qty	unitprice	stockdate
S005	Ballpen	102	100	10	2018/04/22
S003	Gel Pen	101	150	15	2018/03/18
S002	Pencil	102	125	5	2018/02/25
S006	Eraser	101	200	3	2018/01/12
S001	Sharpner	103	210	5	2018/06/11
S004	Compass	102	60	35	2018/05/10
S009	A4 Papers	102	160	5	2018/07/17

a) Display all the items in the ascending order of stockdate.

b) Display maximum price of items for each dealer individually as per dcode from stock.

c) Display all the items in descending orders of itemnames.

d) Display average price of items for each dealer individually as per doce from stock which avergae price is more than 5.

e) Diisplay the sum of quantity for each dcode.

Answers:

a) `select * from stock order by stockdate;`

```
mysql> select * from stock order by stockdate;
+--------+-----------+-------+------+-----------+------------+
| itemno | item      | dcode | qty  | unitprice | stockdate  |
+--------+-----------+-------+------+-----------+------------+
| S006   | Eraser    |   101 |  200 |         3 | 2018-01-12 |
| S002   | Pencil    |   102 |  125 |         5 | 2018-02-25 |
| S003   | Gel Pen   |   101 |  150 |        15 | 2018-03-18 |
| S005   | BallPen   |   102 |  100 |        10 | 2018-04-22 |
| S004   | Compass   |   102 |   60 |        35 | 2018-05-10 |
| S009   | A4 Papers |   102 |  160 |         5 | 2018-07-17 |
| S001   | Sharpner  |   103 |  210 |         5 | 2018-11-06 |
+--------+-----------+-------+------+-----------+------------+
7 rows in set (0.00 sec)
```

b) select dcode,max(unitprice) from stock group by code;

```
mysql> select dcode,max(unitprice) from stock group by dcode;
+-------+----------------+
| dcode | max(unitprice) |
+-------+----------------+
|   103 |              5 |
|   102 |             35 |
|   101 |             15 |
+-------+----------------+
3 rows in set (0.01 sec)
```

c) select * from stock order by item desc;

```
mysql> select * from stock order by item desc;
+--------+-----------+-------+------+-----------+------------+
| itemno | item      | dcode | qty  | unitprice | stockdate  |
+--------+-----------+-------+------+-----------+------------+
| S001   | Sharpner  |   103 |  210 |         5 | 2018-11-06 |
| S002   | Pencil    |   102 |  125 |         5 | 2018-02-25 |
| S003   | Gel Pen   |   101 |  150 |        15 | 2018-03-18 |
| S006   | Eraser    |   101 |  200 |         3 | 2018-01-12 |
| S004   | Compass   |   102 |   60 |        35 | 2018-05-10 |
| S005   | BallPen   |   102 |  100 |        10 | 2018-04-22 |
| S009   | A4 Papers |   102 |  160 |         5 | 2018-07-17 |
+--------+-----------+-------+------+-----------+------------+
7 rows in set (0.01 sec)
```

d) select dcode,avg(unitprice) from stock group by dcode having avg(unitprice)>5;

```
mysql> select dcode,avg(unitprice) from stock group by dcode having avg(unitprice)>5;
+-------+---------------+
| dcode | avg(unitprice) |
+-------+---------------+
|   102 |       13.7500 |
|   101 |        9.0000 |
+-------+---------------+
2 rows in set (0.00 sec)
```

e) select dcode,sum(qty) from stock group by dcode;

```
mysql> select dcode,sum(qty) from stock group by dcode;
+-------+----------+
| dcode | sum(qty) |
+-------+----------+
|   103 |      210 |
|   102 |      445 |
|   101 |      350 |
+-------+----------+
3 rows in set (0.03 sec)
```

PART 5 PYTHON DATABASE CONNECTIVITY

1. **Write a MySQL connectivity program in Python to**
   - **Create a database school**
   - **Create a table students with the specifications - ROLLNO integer, STNAME character(10) in MySQL and perform the following operations:**
     - **Insert two records in it**
     - **Display the contents of the table**
2. **Perform all the operations with reference to table 'students' through MySQL-Python connectivity.**

Answers:

1. Using pymysql - Code:

```
import pymysql as ms

#Function to create Database as per users choice

def c_database():

 try:

  dn=input("Enter Database Name=")

  c.execute("create database {}".format(dn))

  c.execute("use {}".format(dn))

  print("Database created successfully")

 except Exception as a:

  print("Database Error",a)

#Function to Drop Database as per users choice

def d_database():
```

```
 try:
  dn=input("Enter Database Name to be dropped=")
  c.execute("drop database {}".format(dn))
  print("Database deleted sucessfully")
 except Exception as a:
  print("Database Drop Error",a)
#Function to create Table
def c_table():
 try:
  c.execute('''create table students
  (
  rollno int(3),
  stname varchar(20)
  );
  ''')
  print("Table created successfully")
 except Exception as a:
  print("Create Table Error",a)

#Function to Insert Data
def e_data():
 try:
  while True:
   rno=int(input("Enter student rollno="))
   name=input("Enter student name=")
   c.execute("use {}".format('school'))
```

```
    c.execute("insert into students values({},'{}');".format(rno,name))
    db.commit()
    choice=input("Do you want to add more record<y/n>=")
    if choice in "Nn":
     break
  except Exception as a:
   print("Insert Record Error",a)

#Function to Display Data
def d_data():
  try:
   c.execute("select * from students")
   data=c.fetchall()
   for i in data:
    print(i)
  except Exception as a:
   print("Display Record Error",a)
db=ms.connect(host="localhost",user="root",password="root")
c=db.cursor()
while True:
  print("MENU\n1. Create Database\n2. Drop Database \n3. Create Table\n4.
Insert Record \n5. Display Entire Data\n6. Exit")
  choice=int(input("Enter your choice<1-6>="))
  if choice==1:
   c_database()
  elif choice==2:
   d_database()
  elif choice==3:
```

```
  c_table()
elif choice==4:
  e_data()
elif choice==5:
  d_data()
elif choice==6:
  break
else:
  print("Wrong option selected")
```

Output:

```
MENU
1. Create Database
2. Drop Database
3. Create Table
4. Insert Record
5. Display Entire Data
6. Exit
Enter your choice<1-6>=1
Enter Database Name=school
Database created successfully
MENU
1. Create Database
2. Drop Database
3. Create Table
4. Insert Record
5. Display Entire Data
6. Exit
Enter your choice<1-6>=3
Table created successfully
```

```
atabase
abase
able
ecord
Entire Data

choice<1-6>=4
nt rollno=11
nt name=Raj
 to add more record<y/n>=n

atabase
abase
able
ecord
Entire Data

Enter your choice<1-6>=5
(11, 'Raj')
MENU
1. Create Database
2. Drop Database
3. Create Table
4. Insert Record
5. Display Entire Data
6. Exit
Enter your choice<1-6>=6
```

2 . using mysqlconnector

```
import mysql.connector as ms
db=ms.connect(host="localhost",user="root",passwd="root",database='school')
cn=db.cursor()
def insert_rec():
  try:
    while True:
      rn=int(input("Enter roll number:"))
      sname=input("Enter name:")
      marks=float(input("Enter marks:"))
      gr=input("Enter grade:")
            cn.execute("insert into students
values({},'{}',{},'{}')".format(rn,sname,marks,gr))
      db.commit()
      ch=input("Want more records? Press (N/n) to stop entry:")
      if ch in 'Nn':
        break
  except Exception as e:
    print("Error", e)
def update_rec():
  try:
      rn=int(input("Enter rollno to update:"))
      marks=float(input("Enter new marks:"))
      gr=input("Enter Grade:")
      cn.execute("update students set marks={},grade='{}' where
rno={}".format(marks,gr,rn))
      db.commit()
```

```
  except Exception as e:
    print("Error",e)
def delete_rec():
  try:
     rn=int(input("Enter rollno to delete:"))
     cn.execute("delete from students where rno={}".format(rn))
     db.commit()
  except Exception as e:
    print("Error",e)
def view_rec():
  try:
    cn.execute("select * from students")
import mysql.connector as ms
db=ms.connect(host="localhost",user="root",passwd="root",database='school')
cn=db.cursor()
def insert_rec():
  try:
    while True:
     rn=int(input("Enter roll number:"))
     sname=input("Enter name:")
     marks=float(input("Enter marks:"))
     gr=input("Enter grade:")
     cn.execute("insert into students values({},'{}',{},'{}')".format(rn,sname,marks,gr))
```

```
    db.commit()
    ch=input("Want more records? Press (N/n) to stop entry:")
    if ch in 'Nn':
      break
  except Exception as e:
    print("Error", e)
def update_rec():
  try:
    rn=int(input("Enter rollno to update:"))
    marks=float(input("Enter new marks:"))
    gr=input("Enter Grade:")
    cn.execute("update students set marks={},grade='{}' where
rno={}".format(marks,gr,rn))
    db.commit()
  except Exception as e:
    print("Error",e)
def delete_rec():
  try:
    rn=int(input("Enter rollno to delete:"))
    cn.execute("delete from students where rno={}".format(rn))
    db.commit()
```

```
  except Exception as e:
    print("Error",e)
def view_rec():
  try:
    cn.execute("select * from students")
   data=c.fetchall()
    for i in data:
      print(i)
  except Exception as e:
    print("Error",e)
while True:
  print("MENU\n1. Insert Record\n2. Update Record \n3. Delete Record\n4.
Display Record \n5. Exit")
  ch=int(input("Enter your choice<1-4>="))
  if ch==1:
    insert_rec()
  elif ch==2:
    update_rec()
  elif ch==3:
    delete_rec()
  elif ch==4:
```

```
    view_rec()
  elif ch==5:
    break
  else:
    print("Wrong option selected")
  except Exception as e:
    print("Error",e)
while True:
  print("MENU\n1. Insert Record\n2. Update Record \n3. Delete Record\n4.
Display Record \n5. Exit")
  ch=int(input("Enter your choice<1-4>="))
  if ch==1:
    insert_rec()
  elif ch==2:
    update_rec()
  elif ch==3:
    delete_rec()
  elif ch==4:
    view_rec()
  elif ch==5:
    break
  else:
    print("Wrong option selected")
```

Output:

```
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=1
Enter roll number:111
Enter name:Sagar
Enter marks:98
Enter grade:A1
Want more records? Press (N/n) to stop entry:y
Enter roll number:114
Enter name:Vipul
Enter marks:83
Enter grade:A2
Want more records? Press (N/n) to stop entry:n
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=4
(1, 'Amrita', Decimal('88.00'), 'A2')
(102, 'Jay', Decimal('98.00'), 'A1')
(111, 'Sagar', Decimal('98.00'), 'A1')
(114, 'Vipul', Decimal('83.00'), 'A2')
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=2
Enter rollno to update:1
Enter new marks:98
Enter Grade:A1
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=4
(1, 'Amrita', Decimal('98.00'), 'A1')
(102, 'Jay', Decimal('98.00'), 'A1')
(111, 'Sagar', Decimal('98.00'), 'A1')
(114, 'Vipul', Decimal('83.00'), 'A2')
Enter your choice<1-4>=3
Enter rollno to delete:1
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=4
(102, 'Jay', Decimal('98.00'), 'A1')
(111, 'Sagar', Decimal('98.00'), 'A1')
(114, 'Vipul', Decimal('83.00'), 'A2')
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=4
(102, 'Jay', Decimal('98.00'), 'A1')
(111, 'Sagar', Decimal('98.00'), 'A1')
(114, 'Vipul', Decimal('83.00'), 'A2')
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=5
>>>
```

3. Write a menu-driven program to store data into a MySQL database named shop and table customer as following:

1. Add customer details
2. Update customer details
3. Delete customer details
4. Display all customer details

```
import mysql.connector as ms
db=ms.connect(host="localhost",user="root",passwd="root",database='mydb')
cn=db.cursor()
def insert_rec():
  try:
    while True:
      cid=int(input("Enter customer id:"))
      cname=input("Enter name:")
      city=input("Enter city:")
      bill_amt=float(input("Enter bill amount:"))
      cat=input("Enter category:")
      cn.execute("insert into customer
values({},'{}','{}',{},'{}')".format(cid,cname,city,bill_amt,cat))
      db.commit()
      ch=input("Want more records? Press (N/n) to stop entry:")
      if ch in 'Nn':
        break
  except Exception as e:
    print("Error", e)


def update_rec():
```

```
try:
  cn.execute("select * from customer")
  data=cn.fetchall()
  for i in data:
    ci=i[0]
    cna=i[1]
    ct=i[2]
    b=i[3]
    c=i[4]
  cid=int(input("Enter customer id to update:"))
  if cid==ci:
    ch_cname=input("Want to update Name, Press 'Y':")
    if ch_cname.lower()=='y':
      cname=input("Enter new name:")
    else:
      cname=cna
    ch_city=input("Want to update city, Press 'Y':")
    if ch_city.lower()=='y':
      city=input("Enter new city:")
    else:
      city=ct
    ch=input("Want to update bill amount, Press 'Y':")
    if ch.lower()=='y':
      bill_amt=float(input("Enter new bill amount:"))
    else:
      bill_amt=b
    ch_cat=input("Want to update Category, Press 'Y':")
    if ch_cat.lower()=='y':
```

```
      cat=input("Enter new category:")
    else:
      cat=c
    cn.execute("update customer set cname='{}', city='{}', bill_amt={},category='{}'
where cust_id={}".format(cname,city,bill_amt,cat,cid))
    db.commit()
  else:
    print("Record Not Found...")
 except Exception as e:
  print("Error",e)
def delete_rec():
 try:
    cid=int(input("Enter rollno to delete:"))
    cn.execute("delete from customer where cust_id={}".format(cid))
    db.commit()
 except Exception as e:
  print("Error",e)
def view_rec():
 try:
  cn.execute("select * from customer")
  data=cn.fetchall()
  cnt=0
  for i in data:
    cnt=cnt+1
    print("Record:",cnt)
    print('~'*50)
    print("Customer ID:",i[0])
    print("Customer Name:",i[1])
```

```
    print("City:",i[2])
    print("Bill Amount:",i[3])
    print("Category:",i[4])
    print('~'*50)
  except Exception as e:
   print("Error",e)
while True:
 print("MENU\n1. Insert Record\n2. Update Record \n3. Delete Record\n4.
Display Record \n5. Exit")
 ch=int(input("Enter your choice<1-4>="))
 if ch==1:
  insert_rec()
 elif ch==2:
  update_rec()
 elif ch==3:
  delete_rec()
 elif ch==4:
  view_rec()
 elif ch==5:
  break
 else:
  print("Wrong option selected")
```

Output:

```
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=1
Enter customer id:106
Enter name:Shashank
Enter city:Bhuj
Enter bill amount:2500
Enter category:Silver
Want more records? Press (N/n) to stop entry:n
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=4
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 1
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 102
Customer Name: Vishruti
City: Ahmedabad
Bill Amount: 1500.00
Category: Silver
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 2
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 104
Customer Name: Ankit
City: Ahmedabad
Bill Amount: 3200.00
Category: Platinum
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 3
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 106
Customer Name: Shashank
City: Bhuj
Bill Amount: 2500.00
Category: Silver
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 4
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 110
Customer Name: Heena
City: Baroda
Bill Amount: 1500.00
Category: Golden
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=2
Enter customer id to update:102
Want to update Name, Press 'Y':y
Enter new name:Sujata
Want to update city, Press 'Y':n
Want to update bill amount, Press 'Y':y
Enter new bill amount:3000
Want to update Category, Press 'Y':y
Enter new category:Golden
Data Updated successfully...
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=3
Enter customer id to delete:110
Record Deleted...
MENU
1. Insert Record
2. Update Record
3. Delete Record
4. Display Record
5. Exit
Enter your choice<1-4>=5
```

4. Modify the above program and display the customer details based on the following menu:

1. Display customer details by city
2. Display customer details by bill amount
3. Display customer details by name
4. Display customer details by category

```
import mysql.connector as ms
db=ms.connect(host="localhost",user="root",passwd="root",database='mydb')
cn=db.cursor()

def byCity():
  try:
    city=input("Enter city to search:")
    cn.execute("select * from customer where city='{}'".format(city))
    data=cn.fetchall()
    if data!=[]:
      cnt=0
      for i in data:
        cnt=cnt+1
        print('~'*100)
        print("Record:",cnt)
        print('~'*100)
        print("Customer ID:",i[0])
        print("Customer Name:",i[1])
        print("City:",i[2])
        print("Bill Amount:",i[3])
        print("Category:",i[4])
    else:
      print("No records found for city ", city)
  except Exception as e:
    print("Error",e)
```

```
def byBillAmt():
  try:
    ba=input("Enter the bill amount:")
    cn.execute("select * from customer where bill_amt={}".format(ba))
    data=cn.fetchall()
    if data!=[]:
      cnt=0
      for i in data:
        cnt=cnt+1
        print('~'*100)
        print("Record:",cnt)
        print('~'*100)
        print("Customer ID:",i[0])
        print("Customer Name:",i[1])
        print("City:",i[2])
        print("Bill Amount:",i[3])
        print("Category:",i[4])
    else:
      print("No records found for bill amount ", ba)
  except Exception as e:
    print("Error",e)

def byName():
  try:
    name=input("Enter the name:")
    cn.execute("select * from customer where cname='{}'".format(name))
    data=cn.fetchall()
    if data!=[]:
```

```
    cnt=0
    for i in data:
      cnt=cnt+1
      print('~'*100)
      print("Record:",cnt)
      print('~'*100)
      print("Customer ID:",i[0])
      print("Customer Name:",i[1])
      print("City:",i[2])
      print("Bill Amount:",i[3])
      print("Category:",i[4])
   else:
     print("No records found for ", name)
  except Exception as e:
   print("Error",e)

def byCat():
  try:
   cat=input("Enter the cat:")
   cn.execute("select * from customer where category='{}'".format(cat))
   data=cn.fetchall()
   if data!=[]:
    cnt=0
    for i in data:
      cnt=cnt+1
      print('~'*100)
      print("Record:",cnt)
      print('~'*100)
```

```
        print("Customer ID:",i[0])
        print("Customer Name:",i[1])
        print("City:",i[2])
        print("Bill Amount:",i[3])
        print("Category:",i[4])
    else:
      print("No records found for category ", cat)
  except Exception as e:
    print("Error",e)



while True:
  print('''
MENU
1.Display customer details by city
2.Display customer details by bill amount
3.Display customer details by name
4.Display customer details by category
5.Exit
''')
  ch=int(input("Enter your choice<1-4>="))
  if ch==1:
    byCity()
  elif ch==2:
    byBillAmt()
  elif ch==3:
    byName()
  elif ch==4:
```

```
   byCat()
  elif ch==5:
   break
  else:
   print("Wrong option selected")
```

Output:

```
MENU
1.Display customer details by city
2.Display customer details by bill amount
3.Display customer details by name
4.Display customer details by category
5.Exit

Enter your choice<1-4>=1
Enter city to search:Ahmedabad
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 1
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 102
Customer Name: Sujata
City: Ahmedabad
Bill Amount: 3000.00
Category: Golden
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 2
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 104
Customer Name: Ankit
City: Ahmedabad
Bill Amount: 3200.00
Category: Platinum

MENU
1.Display customer details by city
2.Display customer details by bill amount
3.Display customer details by name
4.Display customer details by category
5.Exit
```

```
Enter your choice<1-4>=2
Enter the bill amount:3000
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 1
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 102
Customer Name: Sujata
City: Ahmedabad
Bill Amount: 3000.00
Category: Golden

MENU
1.Display customer details by city
2.Display customer details by bill amount
3.Display customer details by name
4.Display customer details by category
5.Exit

Enter your choice<1-4>=3
Enter the name:Ankit
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 1
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 104
Customer Name: Ankit
City: Ahmedabad
Bill Amount: 3200.00
Category: Platinum

MENU
1.Display customer details by city
2.Display customer details by bill amount
3.Display customer details by name
4.Display customer details by category
5.Exit
```

```
Enter your choice<1-4>=4
Enter the cat:Golden
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Record: 1
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Customer ID: 102
Customer Name: Sujata
City: Ahmedabad
Bill Amount: 3000.00
Category: Golden

MENU
1.Display customer details by city
2.Display customer details by bill amount
3.Display customer details by name
4.Display customer details by category
5.Exit

Enter your choice<1-4>=5
```