LEAD KINDLY LIGHT

UIS

GEMS

United Indian School

ABU DHABI

PRACTICAL RECORD 2022-2023

Roll No.____________________

Name: ________________________________

Class:________________

Sec:_______________

Subject:_______________________________

GEMS

EDUCATION

Certificate

Roll No. ___________________

This is to certify that

Miss/Master________________________ of grade _____________________ section ____________ has carried out practical work in Assignment prescribed by the Central Board of Secondary Education, New Delhi during the academic year 2022-2023.

Teacher-in charge: Dr.Harini Priyadharsini

Date: ___________________

_____________________ External Examiner

_____________________ Internal Examiner

INDEX

- **PYTHON PROGRAMS**

1. Calculating factorial of a number
2. Checking if a number is prime number or not
3. Search any word in a given string
4. Bubble sort
5. Display file content line by line
6. Display the total number of consonants, uppercase, vowels and lowercase characters in a text file
7. Create a binary file and display its contents
8. Create a binary file and update its contents
9. Copy a file to another file excluding lines containing ‘a’
10. Create and display a CSV file
11. Simulate a dice
12. Implement Stack in Python using list
13. Linear and Binary Search
14. Count most common words in phishing email
15. Store and display data using MySQL connectivity
16. Search data using MySQL connectivity
17. Update data using MySQL connectivity
18. Delete data using MySQL connectivity
19. Create a CSV file with given details
20. Copy all contents from one CSV file to another

- **MySQL**

PYTHON PROGRAMS

Experiment No: 1

Aim: Input any number from user and calculate factorial of a number

CODE:

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)


n = int(input("Enter a number: "))
print(factorial(n))
```

OUTPUT

```
#Run 1
Enter a number: 0
1

#Run 2
Enter a number: 69
171122452428141311372468338881272839092270544893520369393648040923257279754
140647424000000000000000

#Run 3
Enter a number: 420
1179832395293178259148587778443982767423908163629667689799210969550884231351169347
8047667995005102940503883496965320847293740875 333842040193228929611788194646981212
6353301268533527300429478938265247732446542700170132623014591146631602964471437174
8823861128004214806081770714277374544632880180009063325310867611466814559562175609
41434017741747858029098129266158670076807554478836024205 34368994391860098591471476
5387864406466779970942769373120803592028405220313102208368842580526563153497848176
1954009800546844281261649619610291306374918025956972209823833523561696079181976208
783662818235613615149296343931089295234402130043253489826928097199 2110743409299161
6162585470522759556509074096211379330874264959860396374796094106383547466430697189
2700806057422478626083960243385932102946293048920279760860198799159782580284293120
0000000000000000000000000000000000000000000000000000000000000000000000000000 000000
00000000000000000000
```

Experiment No: 2

Aim: Input any number from user and check it is Prime number or not

CODE:

```
def check_prime():
    num = int(input("Enter a number: "))
    if num > 1:
        for i in range(2, num):
            if (num % i) == 0:
                print(num, "is not a prime number")
                break
        else:
            print(num, "is a prime number")
    else:
        print(num, "is not a prime number")


check_prime()
```

OUTPT

```
Enter a number: 22
22 is not a prime number

Enter a number: 2
2 is a prime number

Enter a number: 0
0 is not a prime number
```

Experiment No: 3

Aim : Program to search any word in given string/sentence

CODE:

```
def search_in_string():
    s = input("Enter a string: ")
    t = input("Enter a substring to search in string: ")
    if t in s:
        print("Substring found")
        print("The number of times the substring is present in the string is", s.count(t))
    else:
        print("Substring not found")


search_in_string()
```

OUTPUT:

```
Enter a string: Computer Science
Enter a substring to search in string: computer
Substring not found


Enter a string: Computer Science
Enter a substring to search in string: Science
Substring found
The number of times the substring is present in the string is 1
```

Experiment No: 4

Aim : Program bubble sort using a user-defined function

CODE:

```
def BubbleSort():
    L = [eval(i) for i in input("Enter the list items : ").split()]
    for i in range(len(L)):
        for j in range(len(L)-1):
            if L[j] > L[j+1]:
                L[j], L[j+1] = L[j+1], L[j]
    return L


print(BubbleSort())
```

OUTPUT:

```
Enter the list items : 3 7 9 5 2 8 4 1 6 2 10

[1, 2, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Enter the list items : 'banana' 'apple' 'orange'
'mango'

['apple', 'banana', 'mango', 'orange']
```

Experiment No: 5

Aim : Program to read and display file content line by line with each word separated by “#”

CODE:

```
def split():
    f = open("data.txt", "r")
    for line in f.readlines():
        print(line.replace(' ', '#'), end='')
    f.close()


split()
```

File Content

```
Python Software
This is the new beginning
```

OUTPUT

```
Python#Software
This#is#the#new#beginning
```

Experiment No: 6

Aim : Program to read the content of file and display the total number of consonants, uppercase, vowels and lower case characters"

CODE:

```
def count_vowels():
    vowels = 0
    for line in L:
        for char in line:
            if char.lower() in 'aeiou':
                vowels += 1
    print("The number of vowels in the file is",
vowels)


def count_consonants():
    consonants = 0
    for line in L:
        for char in line:
            if char.isalpha() and char.lower() not
in 'aeiou':
                consonants += 1
    print("The number of consonants in the file
is", consonants)


def count_lowercase():
    lowercase = 0
    for line in L:
        for char in line:
            if char.islower():
                lowercase += 1
    print("The number of lowercase letters in
the file is", lowercase)


def count_uppercase():
    uppercase = 0
    for line in L:
        for char in line:
            if char.isupper():
                uppercase += 1
    print("The number of uppercase letters in
the file is", uppercase)
```

```
f = open("data.txt", "r")
L = f.readlines()
f.close()
count_vowels()
count_consonants()
count_lowercase()
count_uppercase()
```

File Content

There was koruna on the street. A girl was walking by and saw it. She bent down to pick it up and was on her way back home

OUTPUT

```
The number of vowels in the file is 33
The number of consonants in the file is 61
The number of lowercase letters in the file is
90
The number of uppercase letters in the file is
3
```

Experiment No: 7

Aim : Program to create binary file to store Rollno and Name, Search any Rollno and display name if Rollno found otherwise "Rollno not found"

CODE:

```
import pickle
L = []


def add_student():
    f = open("student.dat", "wb+")
    while True:
        roll = int(input("Enter roll number: "))
        name = input("Enter name: ")
        L.append([roll, name])
        if input("Would you like to add another student? (y/n): ") != "y":
            break
        else:
            continue
    print()
    pickle.dump(L, f)
    f.close()


def search_student():
    roll = int(input("Enter roll number to search: "))
    f = open("student.dat", "rb")
    c = pickle.load(f)
    for i in c:
        if roll == i[0]:
            print("Roll number:", i[0], "Name:", i[1], '\n')
            break
    else:
        print("Roll number not found")
    f.close()


while True:
    print("Menu")
    print("1. Add student\n2. Search student\n3. Exit")
    choice = int(input("Enter your choice: "))
    if choice == 1:
        add_student()
    elif choice == 2:
        search_student()
    elif choice == 3:
        exit()
```

OUTPUT

```
Menu
1. Add student
2. Search student
3. Exit
Enter your choice: 1
Enter roll number: 01
Enter name: Shysie
Would you like to add another student? (y/n): y
Enter roll number: 02
Enter name: Fathima
Would you like to add another student? (y/n): n

Menu
1. Add student
2. Search student
3. Exit
Enter your choice: 2
Enter roll number to search: 02
Roll number: 2 Name: Fathima

Menu
1. Add student
2. Search student
3. Exit
Enter your choice: 2
Enter roll number to search: 03
Roll number not found

Menu
1. Add student
2. Search student
3. Exit
Enter your choice: 3
```

Experiment No: 8

Aim : Program to create binary file to store Rollno,Name and Marks and update marks of entered Rollno

CODE:

```
import pickle
L = []


def add_student():
    f = open("student.dat", "wb+")
    while True:
        roll = int(input("Enter roll number: "))
        name = input("Enter name: ")
        marks = int(input("Enter marks: "))
        L.append([roll, name, marks])
        if input("Would you like to add another student? (y/n): ") != "y":
            break
        else:
            continue
    print()
    pickle.dump(L, f)
    f.close()


def search_student():
    roll = int(input("Enter roll number to search: "))
    f = open("student.dat", "rb")
    c = pickle.load(f)
    for i in c:
        if roll == i[0]:
            print("Roll number:", i[0], "Name:", i[1], "Marks: ", i[2], '\n')
            break
    else:
        print("Roll number not found")
    f.close()


def edit_student():
    roll = int(input("Enter roll number to edit: "))
    mark = int(input("Enter new marks: "))
    f = open("student.dat", "rb")
    c = pickle.load(f)
    f.close()
    for i in c:
        if roll == i[0]:
            i[2] = mark
            f = open("student.dat", "wb")
            pickle.dump(c, f)
            f.close()
            break
    else:
        print("Roll number not found")
    print("Marks updated")


while True:
    print("Menu")
    print("1. Add student\n2. Search student\n3. Edit student\n4. Exit")
    choice = int(input("Enter your choice: "))
    if choice == 1:
        add_student()
    elif choice == 2:
        search_student()
    elif choice == 3:
        edit_student()
    elif choice == 4:
        exit()
```

OUTPUT

```
Menu
1. Add student
2. Search student
3. Edit student
4. Exit
Enter your choice: 1
Enter roll number: 01
Enter name: Shysie
Enter marks: 91
Would you like to add another student? (y/n): y
Enter roll number: 02
Enter name: Fathima
Enter marks: 83
Would you like to add another student? (y/n): n

Menu
1. Add student
2. Search student
3. Edit student
4. Exit
Enter your choice: 2
Enter roll number to search: 01
Roll number: 1 Name: Shysie Marks:  91
```

```
Menu
1. Add student
2. Search student
3. Edit student
4. Exit
Enter your choice: 3
Enter roll number to edit: 02
Enter new marks: 85
Marks updated
Menu
1. Add student
2. Search student
3. Edit student
4. Exit
Enter your choice: 4
```

Experiment No: 9

Aim : Program to read the content of file line by line and write it to another file except for the lines contains „a‟ letter in it.

CODE:

```
f1 = open("file1.txt", "r")
f2 = open("file2.txt", "w")
c = f1.readlines()
L = []
for i in c:
    if 'a' not in i.lower():
        L.append(i)
f2.writelines(L)
f1.close()
f2.close()
```

File Content

```
I have a dog
The dog is cute
We have a dog house
It is a good dog
```

OUTPUT

```
The dog is cute
```

Experiment No: 10

Aim : Program to create CSV file and store empno, name, salary and search any empno and display name, salary and if not found appropriate message.

CODE:

```
import csv
L = []
f = open('data.csv', 'w', newline='')
writer = csv.writer(f, delimiter=',')
writer.writerow(['Employee ID', 'Employee Name', 'Employee Salary'])
f.close()


def create():
    f = open('data.csv', 'a+', newline='')
    writer = csv.writer(f, delimiter=',')
    while True:
        idno = input("Enter Employee ID: ")
        name = input("Enter Employee Name: ")
        salr = input("Enter Employee Salary: ")
        L.append([idno, name, salr])
        if input("Do you want to continue? (y/n): ").lower() != 'y':
            break
    writer.writerows(L)
    f.close()


def search():
    f = open('data.csv', 'r')
    reader = csv.reader(f, delimiter=',')
    idno = input("Enter Employee ID: ")
    for row in reader:
        if idno == row[0]:
            print("Employee ID: ", row[0], "\nEmployee Name: ", row[1], "\nEmployee Salary: ",
row[2])
            break
    f.close()


while True:
    print("Menu")
    print("1. Create\n2. Search\n3. Exit")

    ch = int(input("Enter your choice: "))
    if ch == 1:
        create()
    elif ch == 2:
        search()
    elif ch == 3:
        exit()
    else:
        print("Invalid choice")
```

OUTPUT

```
Menu
1. Create
2. Search
3. Exit
Enter your choice: 1
Enter Employee ID: 001
Enter Employee Name: Tom
Enter Employee Salary: 2000
Do you want to continue? (y/n): y
Enter Employee ID: 002
Enter Employee Name: Alex
Enter Employee Salary: 1500
Do you want to continue? (y/n): n
Menu
1. Create
2. Search
3. Exit
Enter your choice: 2
Enter Employee ID: 001
Employee ID:  001
Employee Name:  Tom
Employee Salary:  2000
Menu
1. Create
2. Search
3. Exit
Enter your choice: 3
```

Experiment No: 11

Aim : Program to generate random number 1-6, simulating a dice

CODE:

```
import random
import time


def simulate_dice():
    try:
        print("Press Ctrl+C to stop the dice")
        while True:
            n = random.randint(1, 6)
            print(n, end=' ')  # This statement can be commented out to keep the terminal clean
            time.sleep(0.001)
    except KeyboardInterrupt:
        print("\nDice stopped")
        print("Your number is", n)


simulate_dice()
```

OUTPUT

```
Press Ctrl+C to stop the dice
2 4 1 5 1 2 2 4 5 6 1 5 5 2 2 4 1 3 4 2 5 3 6 4 3 6 1 5 2 6 4 4 6 3 3 5 1 2 3 4 2 6 3 3 5 6 4 3 6 5 2 1 4 4 4 2
2 4 4 2 3 3 5 1 2 5 3 4 2 3 4 5 2 4 3 3 6 3 3 6 6 3 5 3 2 5 5 5 4 6 5 3 3 4 3 4 1 6 5 3 4 5 6 1 2 5 2 1 6 2 5 3
2 4 4 2 5 4 1 4 6 1 6 6 6 6 4 1 4 4 6 2 4 3 3 1 4 1 5 5 1 1 2 4 5 6 5 3 3 1 4 6 5 1 3 6 3 4 5 1 1 2 4 4 5 2 2 4
4 1 6 1 2 5 1 3 6 4 5 4 2 2 5 1 2 6 4 3 6 4 3 3 6 4 6 1 5 2 3 6 3 2 5 2 6 3 5 1 1 3 5 2 5 1 4 1 6 4 5 3 2 5 4 5
2 4 5 5 1 2 6 6 4 5 5 5 2 4 2 4 1 4 6 4 5 4 5 6 2 3 4 4 1 4 3 4 6 2 3 1 6 2 6 6 5 3 5 4 4 2 5 5 6 2 1 1 4 3 5 2
1 2 2 3 4 3 6 6 3 1 6 3 6 5 3 4 4 1 2 4 5 4 3 2 4 2 4 5 5 5 2 4 3 5 6 5 1 1 4 4 2 4 1 6 6 2 6 6 4 1 1 1 5 6 1 5
4 6 6 3 5 2 5 2 5 3 3 4 4 6 3 6 5 1 3 4 1 2 6 6 6 2 3 1 2 6 5 3 4 2 1 5 2 6 3 1 5 3 6 3 2 2 3 5 5 2 6 6 2 1 4 6
2 5 3 2 5 5 3 5 6 1 2 6 4 3 6 4 4 3 6 3 2 6 4 5 3 3 5 3 6 1 6 3 1 2 1 1 5 5 3 3 4 6 2 4 5 3 5 4 5 6 6 4 6 6 5 5
4 2 5 5 2 1 4 2 2 2 5 6 5 1 3 1 4 5 5 5 4 1 1 5 3 5 5 3 6 2 6 5 5 1 3 2 4 1 4 4 6 1 1 2 6 3 6 6 4 3 5 4 6 3 1 6
3 2 6 1 6 6 2 1 3 2 2 1 1 1 6 2 6 4 6 6 5 5 3 4 2 5 6 1 6 5 3 2 6 5 6 3 5 5 1 5 6 5 5 4 5 6 2 5 2 3 6 6 1 3 2 3
4 4 3 5 2 1 6 1 1 4 4 5 4 2 4 3 1 4 2 3 2 6 6 1 5 1 2 1 2 5 4 5 4 4 1 5 2 2 1 5 6 2 3 6 4 2 2 3 3 2 3 1 5 3 1 1
2 1 1 3 6 6 6 6 2 4 3 5 3 1 1 3 6 5 3 3 2 2 1 4 1 3 5 1 6 4 6 4 1 1 1 1 4 2 1 1 2 3 3 2 4 4 4 3 5 4 6 2 6 6 3 3
2 5 5 2
Dice stopped
Your number is 2
```

Experiment No: 12

Aim : Program to implement Stack in Python using List

CODE:

```
S = []


def isEmpty(S):
    return True if len(S) == 0 else False


def push(S, x):
    S.append(x)
    top = len(S) - 1


def pop(S):
    return "Underflow" if isEmpty(S) else S.pop()


def peek(S):
    return "Underflow" if isEmpty(S) else S[-1]


def show(S):
    print("No elements in the stack") if isEmpty(S) else None
    t = len(S) - 1
    while t >= 0:
        print(S[t], end=' ')
        t -= 1
    print()


while True:
    print("Menu")
    print("1. Push\n2. Pop\n3. Peek\n4. Show\n5. Exit")
    c = int(input("Enter your choice: "))
    if c == 1:
        x = input("Enter the element to be pushed: ")
        push(S, x)
    elif c == 2:
        print(pop(S))
    elif c == 3:
        print(peek(S))
    elif c == 4:
        show(S)
    elif c == 5:
        exit()
    else:
        print("Invalid choice")
```

OUTPUT

```
Menu
1. Push
2. Pop
3. Peek
4. Show
5. Exit
Enter your choice: 1
Enter the element to be pushed: 1
Menu
1. Push
2. Pop
3. Peek
4. Show
5. Exit
Enter your choice: 1
Enter the element to be pushed: 2
Menu
1. Push
2. Pop
3. Peek
4. Show
5. Exit
Enter your choice: 1
Enter the element to be pushed: 3
Menu
1. Push
2. Pop
3. Peek
4. Show
5. Exit
Enter your choice: 2
3
Menu
1. Push
2. Pop
3. Peek
4. Show
5. Exit
Enter your choice: 3
2
```

```
Menu
1. Push
2. Pop
3. Peek
4. Show
5. Exit
Enter your choice: 4
2 1
```

Experiment No: 13

Aim : Linear and Binary Search using lists

CODE:

```
def LinearSearch():
    L = [eval(i) for i in input("Enter the list items : ").split()]
    c = eval(input("Enter element to search : "))
    for i in range(len(L)):
        if L[i] == c:
            print("Element found at index", i)
            break
    else:
        print("Element not found")


def BinarySearch():
    L = [eval(i) for i in input("Enter the list items : ").split()]
    c = eval(input("Enter element to search : "))
    L.sort()
    low = found = 0
    high = len(L) - 1
    while low <= high:
        mid = (low + high) // 2
        if L[mid] == c:
            found = 1
            break
        elif L[mid] > c:
            high = mid - 1
        else:
            low = mid + 1
    if found == 1:
        print("Element found")
    else:
        print("Element not found")


LinearSearch()
BinarySearch()
```

OUTPUT

```
Enter the list items : 3 6 7 2 5 1 8 6 9
Enter element to search : 4
Element not found

Enter the list items : 3 6 7 2 5 1 8 6 9
Enter element to search : 1
Element found
```

Experiment No: 14

Aim : Program to take 10 sample phishing email, and find the most common word occurring

CODE:

```
L = []
d = {}


def findmostoccuringdomain():
    while True:
        c = input("Enter phishing emails (enter q to exit loop): ")
        if c == "q":
            break
        else:
            L.append(c)
    for i in range(len(L)):
        L[i] = L[i].split("@")[1]

    for i in L:
        d[i] = d.get(i, 0) + 1

    max = 0
    for i in d:
        if d[i] > max:
            max = d[i]
            domain = i
    print("\nMost occuring domain is", domain, "with", max, "occurences")


findmostoccuringdomain()
```

OUTPUT

```
Enter phishing emails (enter q to exit loop): chrisworth@solo.com
Enter phishing emails (enter q to exit loop): millionwinner@semis.com
Enter phishing emails (enter q to exit loop): wheelspin@solo.com
Enter phishing emails (enter q to exit loop): lotterywin@finals.com
Enter phishing emails (enter q to exit loop): q

Most occuring domain is solo.com with 2 occurences
```

Experiment No: 15

Aim : Program to connect with database and store record of employee and display records.

CODE:

```
from mysql.connector import connect as c
cdb = c(user='root', password='root', host='localhost')
db = cdb.cursor()

db.execute("CREATE DATABASE IF NOT EXISTS company")
db.execute("USE company")
db.execute("CREATE TABLE IF NOT EXISTS employee (id INT PRIMARY KEY, name
VARCHAR(255), salary INT, department VARCHAR(255))")
cdb.commit()


def add_records():
    while True:
        id = int(input("Enter ID: "))
        name = input("Enter Name: ")
        salary = int(input("Enter Salary: "))
        department = input("Enter Department: ")
        db.execute("INSERT INTO employee VALUES (%s, %s, %s, %s)", (id, name, salary,
department))
        cdb.commit()
        print("Record Added Successfully")
        ch = input("Do you want to add more records? (y/n): ")
        if ch != 'y':
            break

def show_records():
    db.execute("SELECT * FROM employee")
    rs = db.fetchall()
    print("%10s" % "Employee ID", "%20s" % "Employee Name", "%10s" % "Salary", "%20s"
% "Department")
    for i in rs:
        print("%7s" % i[0], "%20s" % i[1], "%12s" % i[2], "%17s" % i[3])


while True:
    print("Menu")
    print("1. Add Records\n2. Show Records\n3. Exit")
    ch = int(input("Enter your choice: "))
    if ch == 1:
        add_records()
    elif ch == 2:
        show_records()
    elif ch == 3:
        exit()
```

OUTPUT

```
Menu
1.Add Records
2.Show Records
3.Exit
Enter your choice: 1
Enter ID: 001
Enter Name: Shysie
Enter Salary: 10000
Enter Department: HR
Record Added Successfully
Do you want to add more records? (y/n): y
Enter ID: 002
Enter Name: Fathima
Enter Salary: 10000
Enter Department: Finance
Record Added Successfully
Do you want to add more records? (y/n): n

Menu
1.Add Records
2.Show Records
3.Exit
Enter your choice: 2

Employee ID          Employee Name          Salary        Department
    001                   Shysie             10000            HR
    002                   Fathima            10000          Finance

Menu
1.Add Records
2.Show Records
3.Exit
Enter your choice: 3
```

Experiment No: 16

Aim: Program to connect with database and search employee number in table employee and display record, if empno not found display appropriate message.

CODE:

```
from mysql.connector import connect as c
cdb = c(user='root', password='root', host='localhost')
db = cdb.cursor()

db.execute("CREATE DATABASE IF NOT EXISTS company")
db.execute("USE company")
db.execute("CREATE TABLE IF NOT EXISTS employee (id INT PRIMARY KEY, name VARCHAR(255),
salary INT, department VARCHAR(255))")
cdb.commit()


def search_records():
    id = int(input("Enter ID: "))
    db.execute("SELECT * FROM employee WHERE id = %s", (id,))
    rs = db.fetchall()
    if len(rs) == 0:
        print("Employee not found")
    else:
        print("Employee ID: ", rs[0][0], "\nEmployee Name: ", rs[0][1], "\nSalary: ", rs[0][2], "\nDepartment: ",
rs[0][3], "\n")


print("Employee Search")
search_records()
```

OUTPUT

```
Employee Search
Enter ID: 101
Employee ID: 101
Employee Name: Shysie
Salary: 10000
Department: HR

Employee Search
Enter ID: 102
Employee ID: 102
Employee Name: Fathima
Salary: 10000
Department: Finance
```

Experiment No: 17

Aim : Program to connect with database and update the employee record of entered empno.

CODE:

```
from mysql.connector import connect as c
cdb = c(user='root', password='root', host='localhost')
db = cdb.cursor()

db.execute("CREATE DATABASE IF NOT EXISTS company")
db.execute("USE company")
db.execute("CREATE TABLE IF NOT EXISTS employee (id INT PRIMARY KEY, name
VARCHAR(255), salary INT, department VARCHAR(255))")
cdb.commit()


def edit_records():
    id = int(input("Enter ID: "))
    db.execute("SELECT * FROM employee WHERE id = %s", (id,))
    rs = db.fetchall()
    if len(rs) == 0:
        print("Employee not found")
        exit()
    else:
        print("Employee ID: ", rs[0][0], "\nEmployee Name: ", rs[0][1], "\nSalary: ", rs[0][2],
"\nDepartment: ", rs[0][3], "\n")

    name = input("Enter new name: ")
    salary = int(input("Enter new salary: "))
    department = input("Enter new department: ")
    db.execute("UPDATE employee SET name = %s, salary = %s, department = %s
WHERE id = %s", (name, salary, department, id))
    cdb.commit()
    print("Record updated successfully")


print("Employee Updation")
edit_records()
```

OUTPUT

```
Employee Updation
Enter ID: 101
Employee ID: 101
Employee Name: Shysie
Salary: 10000
Department: HR

Enter new name: Shysie Santhosh
Enter new salary: 10500
Enter new department: IT
Record updated successfully
```

Experiment No: 18

Aim : Program to connect with database and delete the record of entered employee number.

CODE:

```
from mysql.connector import connect as c
cdb = c(user='root', password='root', host='localhost')
db = cdb.cursor()

db.execute("CREATE DATABASE IF NOT EXISTS company")
db.execute("USE company")
db.execute("CREATE TABLE IF NOT EXISTS employee (id INT PRIMARY KEY, name
VARCHAR(255), salary INT, department VARCHAR(255))")
cdb.commit()


def delete_records():
    id = int(input("Enter ID: "))
    db.execute("SELECT * FROM employee WHERE id = %s", (id,))
    rs = db.fetchall()
    if len(rs) == 0:
        print("Employee not found")
        exit()
    else:
        print("Employee ID: ", rs[0][0], "\nEmployee Name: ", rs[0][1], "\nSalary: ", rs[0][2],
"\nDepartment: ", rs[0][3], "\n")

    ch = input("Do you want to delete this record? (y/n): ")
    if ch == "y":
        db.execute("DELETE FROM employee WHERE id = %s", (id,))
        cdb.commit()
        print("Record deleted successfully")
    else:
        print("Record not deleted")
        exit()


print("Employee Deletion")
delete_records()
```

OUTPUT

```
Employee Deletion
Enter ID: 102
Employee ID: 102
Employee Name: Fathima
Salary: 10000
Department: Finance

Do you want to delete this record? (y/n): y
Record deleted successfully
```

Experiment No: 19

Aim: Write a method CREATE() to create an EMP.csv file with the following details:

Id to store employee number of integer type,
name to store employee name of string type,
dept to store their respective department of string type,
basic to store basic salary of respective employee,
hra to be calculated from his/her basic salary which is 10% of basic ,
sal to be calculated as salary = basic_salary + hra

CODE:

```
        import csv


        def create():
            f = open('employee.csv', 'w', newline='')
            w = csv.writer(f)
            w.writerow(['ID', 'Name', 'Salary', 'Department'])
            f.flush()
            while True:
                id = int(input("Enter ID: "))
                name = input("Enter Name: ")
                dept = input("Enter Department: ")
                basic = int(input("Enter Basic Salary: "))
                hra = 0.1 * basic  # House Rent Allowance
                sal = basic + hra
                w.writerow([id, name, dept, str(basic), str(hra), str(sal)])
                f.flush()
                ch = input("Do you want to add more records? (y/n): ")
                if ch != 'y':
                    break
            f.close()


create()
```

OUTPUT

```
Enter ID: 101
Enter Name: Shysie
Enter Department: IT
Enter Basic Salary: 10000
Do you want to add more records? (y/n): y
Enter ID: 102
Enter Name: Fathima
Enter Department: Finance
Enter Basic Salary: 10000
Do you want to add more records? (y/n): n
```

File content

```
ID, Name, Salary, Department
101, Shysie, IT, 10000, 100.0, 11000.0
101, Fathima, Finance, 10000, 100.0, 11000.0
```

Experiment No: 20

Aim : Write a Python program to copy file1.csv into file2.csv.

CODE:

```
import csv


def copy():
    f = open('csv1.csv', 'r')
    if not f:
        print("File not found!")
        exit()
    r = csv.reader(f)
    f1 = open('csv2.csv', 'w', newline='')
    w = csv.writer(f1)
    for row in r:
        w.writerow(row)
    f.close()
    f1.close()


copy()
```

OUTPUT:

csv1.csv

```
Name,Abbreviation,Numeric
January,Jan,1
Feburary,Feb,2
March,Mar,3
April,Apr,4
May,May,5
June,June,6
July,July,7
August,Aug,8
September,Sept,9
October,Oct,10
November,Nov,11
December,Dec,12
```

c sv 2 .csv

```
Name,Abbreviation,Numeric
January,Jan,1
Feburary,Feb,2
March,Mar,3
April,Apr,4
May,May,5
```

```
June,June,6
July,July,7
August,Aug,8
September,Sept,9
October,Oct,10
November,Nov,11
December,Dec,12
```

MySQL QUERIES

Create Database LOANS and use it

1.Create the database LOANS.
Answer:

```
mysql> create database LOANS;
Query OK, 1 row affected (0.25 sec)
```

2. Use the database LOANS.
Answer:

```
mysql> use LOANS;
Database changed
```

Create Table / Insert Into

3. Create the table Loan_Accounts and insert tuples in it.
Answer:

```
mysql>  create table Loan_Accounts(Accno int(5),Cust_Name varchar(35),Loan_amount
int(25),Instalments int(30),Int_Rate decimal(12,2)NULL,Start_date date,Interest int(5)NULL);
Query OK, 0 rows affected (0.03 sec)
mysql> insert into Loan_Accounts
values(1,"R.K.Gupta",300000,36,12.00,'2009/07/19',NULL);
Query OK, 1 row affected (0.02 sec)
mysql> insert into Loan_Accounts
values(2,"S.P.Sharma",500000,48,10.00,'2008/03/22',NULL);
Query OK, 1 row affected (0.00 sec)
mysql> insert into Loan_Accounts values(3,"K.P.Jain",300000,36,NULL,'2007/03/08',NULL);
Query OK, 1 row affected (0.00 sec)
mysql> insert into Loan_Accounts
values(4,"M.P.Yadav",800000,60,10.00,'2008/12/06',NULL);
Query OK, 1 row affected (0.00 sec)
mysql> insert into Loan_Accounts
values(5,"S.P.Sinha",200000,36,12.50,'2010/01/03',NULL);
Query OK, 1 row affected (0.01 sec)
mysql> insert into Loan_Accounts
values(6,"P.Sharma",700000,60,12.50,'2008/06/05',NULL);
Query OK, 1 row affected (0.01 sec)
mysql> insert into Loan_Accounts values(7,"K.S.Dhall",500000,48,NULL,'2008/03/05',NULL);
Query OK, 1 row affected (0.00 sec)
```

Simple Select

4. Display the details of all the loans.
Answer:

```
mysql> select * from Loan_Accounts;
+-------+-----------+------------+------------+----------+-----------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+------------+------------+----------+-----------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     3 | K.P.Jain   |      300000 |          36 |     NULL | 2007-03-08 |     NULL |
|     4 | M.P.Yadav  |      800000 |          60 |    10.00 | 2008-12-06 |     NULL |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
|     6 | P.Sharma   |      700000 |          60 |    12.50 | 2008-06-05 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |     NULL | 2008-03-05 |     NULL |
+-------+------------+-------------+-------------+----------+------------+----------+
8 rows in set (0.00 sec)
```

5. Display the AccNo, Cust_Name, and Loan_Amount of all the loans.
Answer:

```
mysql> select Accno,Cust_name,Loan_Amount from loan_accounts;
+-------+------------+-------------+
| Accno | Cust_name  | Loan_Amount |
+-------+------------+-------------+
|     1 | R.K.Gupta  |      300000 |
|     2 | S.P.Sharma |      500000 |
|     3 | K.P.Jain   |      300000 |
|     4 | M.P.Yadav  |      800000 |
|     5 | S.P.Sinha  |      200000 |
|     6 | P.Sharma   |      700000 |
|     7 | K.S.Dhall  |      500000 |
+-------+------------+-------------+
8 rows in set (0.00 sec)
```

Conditional Select using Where Clause

6. Display the details of all the loans with less than 40 instalments.
Answer:

```
mysql> select * from loan_accounts where instalments<40;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     3 | K.P.Jain  |      300000 |          36 |     NULL | 2007-03-08 |     NULL |
|     5 | S.P.Sinha |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
+-------+-----------+-------------+-------------+----------+------------+----------+
3 rows in set (0.00 sec)
```

7. Display the AccNo and Loan_Amount of all the loans started before 01-04-2009.
Answer:

```
mysql> select Accno,loan_amount from loan_accounts where start_date<'2009/04/01';
+-------+-------------+
| Accno | loan_amount |
+-------+-------------+
|     2 |      500000 |
|     3 |      300000 |
|     4 |      800000 |
|     6 |      700000 |
|     7 |      500000 |
+-------+-------------+
5 rows in set (0.00 sec)
```

8. Display the Int_Rate of all the loans started after 01-04-2009.
Answer:

```
mysql> select int_rate from loan_accounts where start_date>'2009/04/01';
+----------+
| int_rate |
+----------+
|    12.00 |
|    12.50 |
+----------+
2 rows in set (0.00 sec)
```

Using NULL

9. Display the details of all the loans whose rate of interest is NULL.
Answer:

```
mysql> select * from loan_accounts where int_rate is NULL;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     3 | K.P.Jain  |      300000 |          36 |     NULL | 2007-03-08 |     NULL |
|     7 | K.S.Dhall |      500000 |          48 |     NULL | 2008-03-05 |     NULL |
+-------+-----------+-------------+-------------+----------+------------+----------+
2 rows in set (0.00 sec)
```

10. Display the details of all the loans whose rate of interest is not NULL.
Answer:

```
mysql> select * from loan_accounts where int_rate is not NULL;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     4 | M.P.Yadav  |      800000 |          60 |    10.00 | 2008-12-06 |     NULL |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
|     6 | P.Sharma   |      700000 |          60 |    12.50 | 2008-06-05 |     NULL |
+-------+------------+-------------+-------------+----------+------------+----------+
5 rows in set (0.00 sec)
```

Using DISTINCT Clause

11. Display the amounts of various loans from the table Loan_Accounts. A loan amount should appear only once.
Answer:

```
mysql> select distinct(loan_amount) from loan_accounts;
+-------------+
| loan_amount |
+-------------+
|      300000 |
|      500000 |
|      800000 |
|      200000 |
|      700000 |
+-------------+
5 rows in set (0.02 sec)
```

12. Display the number of instalments of various loans from the table Loan_Accounts. An instalment should appear only once.
Answer:

```
mysql> select distinct(instalments) from loan_accounts;
+-------------+
| instalments |
+-------------+
|          36 |
|          48 |
|          60 |
+-------------+
3 rows in set (0.00 sec)
```

Using Logical Operators (NOT, AND, OR)

13. Display the details of all the loans started after 31-12-2008 for which the number of instalments are more than 36.
Answer:

```
mysql> select * from loan_accounts where start_date>'2008/12/31' AND instalments>36;
Empty set (0.00 sec)
```

14. Display the Cust_Name and Loan_Amount for all the loans which do not have number of instalments 36.

```
mysql> select cust_name,loan_amount from loan_accounts where instalments<>36;
+------------+-------------+
| cust_name  | loan_amount |
+------------+-------------+
| S.P.Sharma |      500000 |
| M.P.Yadav  |      800000 |
| P.Sharma   |      700000 |
| K.S.Dhall  |      500000 |
+------------+-------------+
4 rows in set (0.00 sec)
```

15. Display the Cust_Name and Loan_Amount for all the loans for which the loan amount is less than 500000 or int_rate is more than 12.

Answer:

```
mysql> select cust_name,loan_amount from loan_accounts where loan_amount<500000 OR int_rate>12;
+-----------+-------------+
| cust_name | loan_amount |
+-----------+-------------+
| R.K.Gupta |      300000 |
| K.P.Jain  |      300000 |
| S.P.Sinha |      200000 |
| P.Sharma  |      700000 |
+-----------+-------------+
4 rows in set (0.00 sec)
```

16. Display the details of all the loans which started in the year 2009.

Answer:

```
mysql> select * from loan_accounts where Year(start_date)=2009;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
+-------+-----------+-------------+-------------+----------+------------+----------+
1 row in set (0.00 sec)
```

17. Display the details of all the loans whose Loan_Amount is in the range 400000 to 500000.

Answer:

```
mysql> select * from loan_accounts where loan_amount between 400000 and 500000;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |     NULL | 2008-03-05 |     NULL |
+-------+------------+-------------+-------------+----------+------------+----------+
2 rows in set (0.00 sec)
```

18. Display the details of all the loans whose rate of interest is in the range 11% to 12%.

Answer:

```
mysql> select * from loan_accounts where int_rate between 11 and 12;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
+-------+-----------+-------------+-------------+----------+------------+----------+
1 row in set (0.02 sec)
```

Using IN Operator

19. Display the Cust_Name and Loan_Amount for all the loans for which the number of instalments are 24, 36, or 48. (Using IN operator)
Answer:

```
mysql> select cust_name,loan_amount from loan_accounts where instalments IN(24,36,48);
+------------+-------------+
| cust_name  | loan_amount |
+------------+-------------+
| R.K.Gupta  |      300000 |
| S.P.Sharma |      500000 |
| K.P.Jain   |      300000 |
| S.P.Sinha  |      200000 |
| K.S.Dhall  |      500000 |
+------------+-------------+
5 rows in set (0.00 sec)
```

Using BETWEEN Operator

20. Display the details of all the loans whose Loan_Amount is in the range 400000 to 500000. (Using BETWEEN operator)
Answer:

```
mysql> select * from loan_accounts where loan_amount between 400000 and 500000;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |     NULL | 2008-03-05 |     NULL |
+-------+------------+-------------+-------------+----------+------------+----------+
2 rows in set (0.00 sec)
```

21. Display the details of all the loans whose rate of interest is in the range 11% to 12%. (Using BETWEEN operator)
Answer:

```
mysql> select * from loan_accounts where int_rate between 11 and 12;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
+-------+-----------+-------------+-------------+----------+------------+----------+
1 row in set (0.00 sec)
```

Using LIKE Operator

22. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the

Cust_Name ends with 'Sharma'.
Answer:

```
mysql> select Accno,Cust_name,loan_amount from loan_accounts where cust_name like
'%Sharma';
+-------+------------+-------------+
| Accno | Cust_name  | loan_amount |
+-------+------------+-------------+
|     2 | S.P.Sharma |      500000 |
|     6 | P.Sharma   |      700000 |
+-------+------------+-------------+
2 rows in set (0.01 sec)
```

23. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the Cust_Name ends with 'a'.
Answer:

```
mysql> select accno,cust_name,loan_amount from loan_accounts where cust_name like
'%a';
+-------+------------+-------------+
| accno | cust_name  | loan_amount |
+-------+------------+-------------+
|     1 | R.K.Gupta  |      300000 |
|     2 | S.P.Sharma |      500000 |
|     5 | S.P.Sinha  |      200000 |
|     6 | P.Sharma   |      700000 |
+-------+------------+-------------+
4 rows in set (0.01 sec)
```

24. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the Cust_Name contains 'a'.
Answer:

```
mysql> select accno,cust_name,loan_amount from loan_accounts where cust_name like
'%a%';
+-------+------------+-------------+
| accno | cust_name  | loan_amount |
+-------+------------+-------------+
|     1 | R.K.Gupta  |      300000 |
|     2 | S.P.Sharma |      500000 |
|     3 | K.P.Jain   |      300000 |
|     4 | M.P.Yadav  |      800000 |
|     5 | S.P.Sinha  |      200000 |
|     6 | P.Sharma   |      700000 |
|     7 | K.S.Dhall  |      500000 |
+-------+------------+-------------+
7 rows in set (0.00 sec)
```

25. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the Cust_Name does not contain 'P'.
Answer:

```
mysql> select Accno,Cust_Name,Loan_amount from loan_accounts where cust_name not
like '%P%';
+-------+-----------+-------------+
| Accno | Cust_Name | Loan_amount |
+-------+-----------+-------------+
|     7 | K.S.Dhall |      500000 |
+-------+-----------+-------------+
1 row in set (0.00 sec)
```

26. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the Cust_Name contains 'a' as the second last character.
Answer:

```
mysql> select Accno,Cust_Name,Loan_Amount from loan_accounts where cust_name like
'%a_';
+-------+-----------+-------------+
| Accno | Cust_Name | Loan_Amount |
+-------+-----------+-------------+
|     4 | M.P.Yadav |      800000 |
+-------+-----------+-------------+
1 row in set (0.00 sec)
```

Using ORDER BY clause

27. Display the details of all the loans in the ascending order of their Loan_Amount.
Answer:

```
mysql> select * from loan_accounts order by Loan_Amount asc;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     3 | K.P.Jain   |      300000 |          36 |     NULL | 2007-03-08 |     NULL |
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |     NULL | 2008-03-05 |     NULL |
|     6 | P.Sharma   |      700000 |          60 |    12.50 | 2008-06-05 |     NULL |
|     4 | M.P.Yadav  |      800000 |          60 |    10.00 | 2008-12-06 |     NULL |
+-------+------------+-------------+-------------+----------+------------+----------+
7 rows in set (0.00 sec)
```

28. Display the details of all the loans in the descending order of their Start_Date.
Answer:

```
mysql> select * from loan_accounts order by Start_Date desc;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     4 | M.P.Yadav  |      800000 |          60 |    10.00 | 2008-12-06 |     NULL |
|     6 | P.Sharma   |      700000 |          60 |    12.50 | 2008-06-05 |     NULL |
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |     NULL | 2008-03-05 |     NULL |
|     3 | K.P.Jain   |      300000 |          36 |     NULL | 2007-03-08 |     NULL |
+-------+------------+-------------+-------------+----------+------------+----------+
7 rows in set (0.00 sec)
```

29. Display the details of all the loans in the ascending order of their Loan_Amount and within Loan_Amount in the descending order of their Start_Date.
Answer:

```
mysql> select * from loan_accounts order by Loan_Amount asc,Start_Date desc;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     3 | K.P.Jain   |      300000 |          36 |     NULL | 2007-03-08 |     NULL |
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |     NULL | 2008-03-05 |     NULL |
|     6 | P.Sharma   |      700000 |          60 |    12.50 | 2008-06-05 |     NULL |
|     4 | M.P.Yadav  |      800000 |          60 |    10.00 | 2008-12-06 |     NULL |
+-------+-----------+-------------+-------------+----------+------------+----------+
7 rows in set (0.00 sec)
```

Using UPDATE, DELETE, ALTER TABLE

30. Put the interest rate 11.50% for all the loans for which interest rate is NULL.
Answer:

```
mysql> update loan_accounts set Int_Rate=11.50 where Int_Rate is NULL;
Query OK, 2 rows affected (0.00 sec)
Rows matched: 2  Changed: 2  Warnings: 0
mysql> select * from loan_accounts;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     2 | S.P.Sharma |      500000 |          48 |    10.00 | 2008-03-22 |     NULL |
|     3 | K.P.Jain   |      300000 |          36 |    11.50 | 2007-03-08 |     NULL |
|     4 | M.P.Yadav  |      800000 |          60 |    10.00 | 2008-12-06 |     NULL |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
|     6 | P.Sharma   |      700000 |          60 |    12.50 | 2008-06-05 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |    11.50 | 2008-03-05 |     NULL |
+-------+-----------+-------------+-------------+----------+------------+----------+
7 rows in set (0.00 sec)
```

31. Increase the interest rate by 0.5% for all the loans for which the loan amount is more than 400000.
Answer:

```
mysql> update loan_accounts set Int_Rate=int_rate+0.5 where loan_amount>400000;
Query OK, 4 rows affected (0.01 sec)
Rows matched: 4  Changed: 4  Warnings: 0

mysql> select * from loan_accounts;
+-------+-----------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+-----------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |     NULL |
|     2 | S.P.Sharma |      500000 |          48 |    10.50 | 2008-03-22 |     NULL |
|     3 | K.P.Jain   |      300000 |          36 |    11.50 | 2007-03-08 |     NULL |
|     4 | M.P.Yadav  |      800000 |          60 |    10.50 | 2008-12-06 |     NULL |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |     NULL |
|     6 | P.Sharma   |      700000 |          60 |    13.00 | 2008-06-05 |     NULL |
|     7 | K.S.Dhall  |      500000 |          48 |    12.00 | 2008-03-05 |     NULL |
+-------+------------+-------------+-------------+----------+------------+----------+
7 rows in set (0.00 sec)
```
32. For each loan replace Interest with (Loan_Amount*Int_Rate*Instalments) 12*100.
Answer:
```
mysql> update loan_accounts set Interest=(Loan_amount*Int_Rate*Instalments)/1200;
Query OK, 7 rows affected (0.00 sec)
Rows matched: 7  Changed: 7  Warnings: 0

mysql> select * from loan_accounts;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |   108000 |
|     2 | S.P.Sharma |      500000 |          48 |    10.50 | 2008-03-22 |   210000 |
|     3 | K.P.Jain   |      300000 |          36 |    11.50 | 2007-03-08 |   103500 |
|     4 | M.P.Yadav  |      800000 |          60 |    10.50 | 2008-12-06 |   420000 |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |    75000 |
|     6 | P.Sharma   |      700000 |          60 |    13.00 | 2008-06-05 |   455000 |
|     7 | K.S.Dhall  |      500000 |          48 |    12.00 | 2008-03-05 |   240000 |
+-------+------------+-------------+-------------+----------+------------+----------+
7 rows in set (0.00 sec)
```
33. Delete the records of all the loans whose start date is before 2007.
Answer:
```
mysql> delete from loan_accounts where Start_date<2007;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> select *  from loan_accounts;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |   108000 |
|     2 | S.P.Sharma |      500000 |          48 |    10.50 | 2008-03-22 |   210000 |
|     3 | K.P.Jain   |      300000 |          36 |    11.50 | 2007-03-08 |   103500 |
|     4 | M.P.Yadav  |      800000 |          60 |    10.50 | 2008-12-06 |   420000 |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |    75000 |
|     6 | P.Sharma   |      700000 |          60 |    13.00 | 2008-06-05 |   455000 |
|     7 | K.S.Dhall  |      500000 |          48 |    12.00 | 2008-03-05 |   240000 |
+-------+------------+-------------+-------------+----------+------------+----------+
7 rows in set (0.00 sec)
```
34. Delete the records of all the loans of 'K.P. Jain'.
Answer:
```
mysql> delete from loan_accounts where cust_name="K.P.Jain";
```

```
Query OK, 1 row affected (0.02 sec)

mysql> select * from loan_accounts;
+-------+------------+-------------+-------------+----------+------------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
+-------+------------+-------------+-------------+----------+------------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |   108000 |
|     2 | S.P.Sharma |      500000 |          48 |    10.50 | 2008-03-22 |   210000 |
|     4 | M.P.Yadav  |      800000 |          60 |    10.50 | 2008-12-06 |   420000 |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |    75000 |
|     6 | P.Sharma   |      700000 |          60 |    13.00 | 2008-06-05 |   455000 |
|     7 | K.S.Dhall  |      500000 |          48 |    12.00 | 2008-03-05 |   240000 |
+-------+------------+-------------+-------------+----------+------------+----------+
6 rows in set (0.00 sec)
```
35. Add another column Category of type CHAR(1) in the Loan table.
Answer:
```
mysql> Alter table loan_accounts ADD Category char(1);
Query OK, 6 rows affected (0.09 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select * from loan_accounts;
+-------+------------+-------------+-------------+----------+------------+----------+----------+
| Accno | Cust_Name  | Loan_amount | Instalments | Int_Rate | Start_date | Interest |
Category |
+-------+------------+-------------+-------------+----------+------------+----------+----------+
|     1 | R.K.Gupta  |      300000 |          36 |    12.00 | 2009-07-19 |   108000 | NULL     |
|     2 | S.P.Sharma |      500000 |          48 |    10.50 | 2008-03-22 |   210000 | NULL     |
|     4 | M.P.Yadav  |      800000 |          60 |    10.50 | 2008-12-06 |   420000 | NULL     |
|     5 | S.P.Sinha  |      200000 |          36 |    12.50 | 2010-01-03 |    75000 | NULL     |
|     6 | P.Sharma   |      700000 |          60 |    13.00 | 2008-06-05 |   455000 | NULL     |
|     7 | K.S.Dhall  |      500000 |          48 |    12.00 | 2008-03-05 |   240000 | NULL     |
+-------+------------+-------------+-------------+----------+------------+----------+----------+
6 rows in set (0.00 sec)
```

Find the Output of the following queries

36. SELECT cust_name, LENGTH(Cust_Name),LCASE(Cust_Name),UCASE(Cust_Name) FROM Loan_Accounts WHERE Int_Rate < 11.00;
```
+------------+-------------------+------------------+------------------+
| cust_name  | LENGTH(Cust_Name) | LCASE(Cust_Name) | UCASE(Cust_Name) |
+------------+-------------------+------------------+------------------+
| S.P.Sharma |                10 | s.p.sharma       | S.P.SHARMA       |
| M.P.Yadav  |                 9 | m.p.yadav        | M.P.YADAV        |
+------------+-------------------+------------------+------------------+
2 rows in set (0.00 sec)
```

37.SELECT LEFT(Cust_Name, 3), Right(Cust_Name, 3), SUBSTR(Cust_Name, 1, 3) FROM Loan_Accounts WHERE Int_Rate > 10.00;

```
+--------------------+---------------------+-------------------------+
| LEFT(Cust_Name, 3) | Right(Cust_Name, 3) | SUBSTR(Cust_Name, 1, 3) |
+--------------------+---------------------+-------------------------+
| R.K                | pta                 | R.K                     |
| S.P                | rma                 | S.P                     |
| M.P                | dav                 | M.P                     |
| S.P                | nha                 | S.P                     |
| P.S                | rma                 | P.S                     |
| K.S                | all                 | K.S                     |
+--------------------+---------------------+-------------------------+
6 rows in set (0.00 sec)
```

38. SELECT RIGHT(Cust_Name, 3), SUBSTR(Cust_Name, 5) FROM Loan_Accounts;

```
+---------------------+----------------------+
| RIGHT(Cust_Name, 3) | SUBSTR(Cust_Name, 5) |
+---------------------+----------------------+
| pta                 | Gupta                |
| rma                 | Sharma               |
| dav                 | Yadav                |
| nha                 | Sinha                |
| rma                 | arma                 |
| all                 | Dhall                |
+---------------------+----------------------+
6 rows in set (0.00 sec)
```

39. SELECT DAYNAME(Start_Date) FROM Loan_Accounts;

```
+---------------------+
| DAYNAME(Start_Date) |
+---------------------+
| Sunday              |
| Saturday            |
| Saturday            |
| Sunday              |
| Thursday            |
| Wednesday           |
+---------------------+
6 rows in set (0.00 sec)
```

40. SELECT ROUND(Int_Rate*110/100, 2) FROM Loan_Accounts WHERE Int_Rate > 10;

```
+----------------------------+
| ROUND(Int_Rate*110/100, 2) |
+----------------------------+
|                      13.20 |
|                      11.55 |
|                      11.55 |
|                      13.75 |
|                      14.30 |
|                      13.20 |
+----------------------------+
```

6 rows in set (0.00 sec)

Write the output produced by the following SQL commands:

41. SELECT POW(4,3), POW(3,4);
```
+----------+----------+
| POW(4,3) | POW(3,4) |
+----------+----------+
|       64 |       81 |
+----------+----------+
1 row in set (0.00 sec)
```

42. SELECT ROUND(543.5694,2), ROUND(543.5694), ROUND(543.5694,-1);
```
+-------------------+-----------------+--------------------+
| ROUND(543.5694,2) | ROUND(543.5694) | ROUND(543.5694,-1) |
+-------------------+-----------------+--------------------+
|            543.57 |             544 |                540 |
+-------------------+-----------------+--------------------+
1 row in set (0.00 sec)
```

43. SELECT TRUNCATE(543.5694,2), TRUNCATE(543.5694,-1);
```
+----------------------+-------------------v---+
| TRUNCATE(543.5694,2) | TRUNCATE(543.5694,-1) |
+----------------------+-----------------------+
|               543.56 |                   540 |
+----------------------+-----------------------+
1 row in set (0.00 sec)
```

44. SELECT LENGTH("Prof. M. L. Sharma");
```
+-----------------------------+
| LENGTH("Prof. M. L. Sharma") |
+-----------------------------+
|                          18 |
+-----------------------------+
1 row in set (0.00 sec)
```

45. SELECT CONCAT("SHEIKH", " HAROON") "FULL NAME";
```
+---------------+
| FULL NAME     |
+---------------+
| SHEIKH HAROON |
+---------------+
1 row in set (0.00 sec)
```

SELECT YEAR(‘2000/09/08’);

46. SELECT YEAR(CURDATE()), MONTH(CURDATE()), DAY(CURDATE());

```
+------------------+------------------+-----------------+
| YEAR(CURDATE()) | MONTH(CURDATE()) | DAY(CURDATE()) |
+------------------+------------------+-----------------+
|            2018 |               10 |               9 |
+------------------+------------------+-----------------+
1 row in set (0.00 sec)
```

```
47. SELECT DAYOFYEAR(CURDATE()), DAYOFMONTH(CURDATE()),
DAYNAME(CURDATE());
+----------------------+-----------------------+---------------------+
| DAYOFYEAR(CURDATE()) | DAYOFMONTH(CURDATE()) | DAYNAME(CURDATE()) |
+----------------------+-----------------------+---------------------+
|                  282 |                     9 | Tuesday             |
+----------------------+-----------------------+---------------------+
1 row in set (0.00 sec)
```

```
48. SELECT LEFT("Unicode",3), RIGHT("Unicode",4);
+-------------------+--------------------+
| LEFT("Unicode",3) | RIGHT("Unicode",4) |
+-------------------+--------------------+
| Uni               | code               |
+-------------------+--------------------+
1 row in set (0.00 sec)
```

```
49. SELECT INSTR("UNICODE","CO"), INSTR("UNICODE","CD");
+-----------------------+-----------------------+
| INSTR("UNICODE","CO") | INSTR("UNICODE","CD") |
+-----------------------+-----------------------+
|                     4 |                     0 |
+-----------------------+-----------------------+
1 row in set (0.00 sec)
```

```
50. SELECT MID("Informatics",3,4), SUBSTR("Practices",3);
+------------------------+-----------------------+
| MID("Informatics",3,4) | SUBSTR("Practices",3) |
+------------------------+-----------------------+
| form                   | actices               |
+------------------------+-----------------------+
1 row in set (0.00 sec)
```