1. Create ‘Sales’ database.

.open sales.db

1. Create tables and insert records in ‘Sales’ database and perform SQLite commands given in Practical Assignment 1

Salesman (sid, name, city, commission)

Order (oid, pur\_amt, ord\_date, sid)

Customer(cid,cust\_name,grade,sid)

create table salesman(sid int primary key,name text,city text,commission float);

insert into salesman values(5001,'mohan patel','anand',0.15),(5002,'nail shah','surat',0.13),(5005,'preet vyas','ahmedabad',0.11),(5006,'jeevan mehat','navsari',0.14),(5003,'pual adam','nadaid',0.12),(5007,'ramesh patel','surat',0.13);

create table order411(oid int,pur\_amt int,ord\_date date,sid int references salesman(sid));

insert into order411 values(1,12000,'05-dec-2020',5001),(2,42000,'20-dec-2020',5002),(3,2000,'02-feb-2021',5005),(4,14000,'23-mar-2021',5005),(5,23000,'15-apr-2021',5003),(6,33000,'20-may-2021',5001),(7,32000,'22-jun-2021',5003),(8,23500,'01-jul-2021',5003),(9,43000,'05-jul-2021',5001),(10,12000,'15-jul-2021',5002);

create table customer(cid text,cust\_name text,grade int,sid int references salesman(sid));

insert into customer values('c1','mohit patel',100,5001),('c2','geeta vyas',200,5003),('c3','jaya patel',100,5005),('c4','vishal gohel',300,5005),('c5','kartik goyenka',200,5002),('c6','meera prajapati',100,5001),('c7','veer vyas',300,5002),('c8','maya mehta',200,5003);

1. Create database trigger as follow Before inserting new record in Salesman table check whether commission is in between 0 or 1 or not. If not display error message.

create trigger sales before insert on salesman

begin

select

case

when new.commission not between 0 and 1 then

raise(ABORT,'commission must be between 0 or 1 ')

end;

end;

1. Check whether inserted or update dorder\_date in Order table is in ‘yyyy-mm-dd’ format or not. If not generate error message.

create trigger ord1 before insert update on order411

begin

select

case when new.ord\_date not like'\_\_\_\_-\_\_-\_\_' then

raise(ABORT,'order date must be in yyyy-mm-dd format ')

end;

end;

create trigger ord1\_1 before update on order411

begin

select

case when new.ord\_date not like'\_\_\_\_-\_\_-\_\_' then

raise(ABORT,'order date must be in yyyy-mm-dd format ')

end;

end;

1. Create one TRANSACTION\_LOG table (ID, table\_name, current\_date) when new record is inserted into Salesman or Customer tables. (2 triggers)

create table trans\_log(id,t\_name,curr\_date);

create trigger log1 after insert on salesman

begin

insert into trans\_log(id,t\_name,curr\_date)

values(new.sid,'salesman',datetime('now'));

end;

create trigger log2 after insert on customer

begin

insert into trans\_log(id,t\_name,curr\_date) values(new.cid,'customer',datetime('now'));

end;

1. Example: TRANSACTION\_LOG

Select\*from trans\_log;

1. Create one AUDIT table ( cid, old\_name, new\_name, old\_sid, new\_sid, operation, date) for Customer table’s modified operations (For Update and delete). (2 triggers)

create table audit(cid,old\_name,new\_name,old\_sid,new\_sid,operation,date);

create trigger audit1 after update on customer

begin

insert into audit values(new.cid,old.name,new.name,old.sid,new.sid,'update',datetime('now'))

end;

create trigger audit2 before delete on customer

begin

insert into audit values(new.cid,old.name,new.name,old.sid,new.sid,'delete',datetime('now'))

end;

1. If any record is deleted from Salesman table then first check whether related records are available in child table (customer). If available then generate error message and don’t allow to delete record

create trigger sales2 before delete on salesman

begin

select raise(abort,'cannot delete table')

from customer where sid=old.sid;

end;

1. Perform operation for dumping data into files

.open sales.db

.output sales.txt

.dump

.quit

1. Create “Calculator.py” module and write mathematical functions

def addnum(n1,n2):

    return n1+n2

def subnum(n1,n2):

    return n1-n2

def multinum(n1,n2):

    return n1\*n2

def div(n1,n2):

    return n1/n2

def maxnum(n1,n2):

    if(n1>n2):

        return n1

    else:

        return n2

def smnum(n1,n2):

    if(n1<n2):

        return n1

    else:

        return n2

def prnum(n1,n2):

    return n1\*n2/100

1. Create “calculation.py” file. Take input of required numbers and call each function of “Calculator.py” module and display appropriate message.

import calcualtor

n1=eval(input('enter first number:'))

n2=eval(input('enter second number'))

print('''chose the following:

        1.addition

        2.subtraction

        3.multiplication

        4.division

        5.largest number

        6.smallest number

        7.percentage''')

choice=int(input('enter the choice'))

match(choice):

    case 1:

        a=calcualtor.addnum(n1,n2)

        print(a)

    case 2:

        a=calcualtor.subnum(n1,n2)

        print(a)

    case 3:

        a=calcualtor.multinum(n1,n2)

        print(a)

    case 4:

        a=calcualtor.div(n1,n2)

        print(a)

    case 5:

        a=calcualtor.maxnum(n1,n2)

        print(a)

    case 6:

        a=calcualtor.smnum(n1,n2)

        print(a)

    case 7:

        a=calcualtor.prnum(n1,n2)

        print(a)

1. 1.1. Create “Sales.db” database and perform queries given in Assignment 2 Question 3.4 to 3.17 (CREATE BY USING SQLITE3 MODULE)

Create following table and insert records.

Salesman (sid, name, city, commission) (primary key = SID)

import sqlite3

conn=sqlite3.connect(“sales.db”)

print(conn)

conn.execute(“create table salesman(sid int primary key,name text,city text,commission float”)

print("table created")

conn.execute(“insert into salesman values(5001,'mohan patel','anand',0.15),(5002,'nail shah','surat',0.13),(5005,'preet vyas','ahmedabad',0.11),(5006,'jeevan mehat','navsari',0.14),(5003,'pual adam','nadiad',0.12),(5007,'ramesh patel','surat',0.13)”)

print("record inserted")

1. Display 1st and 2nd record using fetchone() method

row= conn.execute(“select\*from salesman”)

r1=row.fetchone()

r2=row.fetchone()

print(r1)

print(r2)

1. Display data using fechall() method

row= conn.execute(“select\*from salesman”)

r1=row.fetchall()

for i in r1:

print(i)

1. Display 4th and 5th record using fechall() method

row= conn.execute(“select\*from salesman by limit 2 offset 3”)

r1=row.fetchall()

for i in r1:

print(i)

1. Display data without using fecthone() and fetchall() methods

R1=conn.execute(“select\*from salesman”)

for I in R1:

print(i)

1. Display the records of salesman who are leaving in surat city

row= conn.execute(“select\*from salesman where city=’surat’ ”)

r1=row.fetchall()

for i in r1:

print(i)

1. Display records of salesman whose name consists letter “m”

row= conn.execute(“select\*from salesman where name like ‘%m%’”)

r1=row.fetchall()

for i in r1:

print(i)

1. Display records of salesman whose commission is in range of 0.11 to 0.138.

row= conn.execute(“select\*from salesman where commission between 0.11 and 0.13”)

r1=row.fetchall()

for i in r1:

print(i)

1. Display records of salesman of Anand and Navsari city

row= conn.execute(“select\*from salesman where city in(‘anand’,’navsari’) ”)

r1=row.fetchall()

for i in r1:

print(i)

1. Take salesman’s SID from user and display his detail

S=input(‘input salesman’s SID number:- ’)

row=conn.execute(f“select\*from salesman where sid={S}”)

r1=row.fetchall()

for i in r1:

print(i)

1. Modify the commission of 5006 to 0.15

conn.execute(“update salesman set commission=0.15 where sid=5006”)

print(‘updated’)

1. Modify the city to Bharuch whose commission is 0.13

conn.execute(“update salesman set city = ’Bharuch’ where commission=0.13”)

print(‘updated’)

1. Delete the records of 5007

conn.execute(“delete from salesman where sid=5007”)

print(‘deleted’)

1. Delete the records of salesman of Bharuch city

conn.execute(“delete from salesman where city = ’Bharuch’ ”)

print(‘deleted’)

1. Insert three records into salesman table as per user’s input

for i in range(3):

s=input(“Enter salesman’s SID:- ”)

n=input(“Enter salesman’s name:- ”)

city=input(“Enter salesman’s city:- ”)

c=input(“Enter salesman’s commission:- ”)

conn.execute(f“insert into salesman values{s,n,city,c}”)

print(“inserted”)

1. Write examples of user defined package and import package in python file

D:\package\stud.py

Tot=0

Per=0

def stud\_total(m1,m2,m3):

print(“total marks”)

Tot = m1+m2+m3

return=Tot

def stud\_per (total):

Per=total/3

return = Per

from package import stud

m1=int(input(“Enter marks1:- ”))

m2=int(input(“Enter marks2:- ”))

m3=int(input(“Enter marks3:- ”))

total=stud.stud\_total(m1,m2,m3)

print(f”total is:- {total}”)

p=stud.stud\_per(total)

print(f”percentage is:-{p}”)

1. Create ‘stud.csv’ file which includes ‘rno, name, sub1, sub2, sub3, sub4, sub5’ fields. Retrieve data from above created tables.

Import pandas as pd

a={ 'rno': [1, 2, 3, 4, 5],

'name': ['raj', 'jay', 'ravi', 'manoj', 'het'],

'sub1': [85, 78, 90, 88, 92],

'sub2': [82, 88, 84, 79, 91],

'sub3': [76, 95, 89, 85, 87],

'sub4': [90, 91, 93, 86, 80],

'sub5': [88, 85, 92, 90, 84] }

df=pd.DataFrame(a)

df.to\_csv('stud1.csv',index=False)

1. Extract data from ‘stud.csv’ file and store it as dataframe. Perform following tasks:

Import pandas as pd

Import matplotlib.pyplot as plt

df=pd.read\_csv(‘stud.csv’)

1. Display line chart which includes rollno on x axis and Sub1’s marks on y axis.

1.1.1.Give proper headings.

1.1.2.Display grid.

r=df.rno

s1=df.sub1

plt.plot(r,s1,’.b--’)

plt.xlabel(‘roll no’)

plt.ylabel(‘sub1’s marks’)

plt.grid()

plt.show()

2. Display comparison line chart for Sub1, Sub2, Sub3, Sub4 and Sub5 with respect to Rollno.

1.2.1.Give proper heading.

1.2.2.Display legend.

1.2.3.Display each line with different formatting.

r=df.rno

s1=df.sub1

s2=df.sub2

s3=df.sub3

s4=df.sub4

s5=df.sub5

plt.plot(r,s1, linestyle=’-', color='b',label=’Sub1’)

plt.plot(r,s2, linestyle=’-', color='r',label=’Sub2’)

plt.plot(r,s3, linestyle=’- -‘, color='g',label=’Sub3’)

plt.plot(r,s4, linestyle=’:’, color='k',label=’Sub4’)

plt.plot(r,s5, linestyle=’-.’, ', color='y',label=’Sub5’)

plt.title(‘subjectwise marks’)

plt.xlabel(‘roll no’)

plt.ylabel(‘marks’)

plt.legend()

plt.show()

3. Display above two charts in one figure (Use subplotcolumnwise)

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv('stud.csv')

plt.subplot(1,2,1)

plt.plot(df['rno'],df['sub1'],linewidth=3)

plt.xlabel('roll no')

plt.ylabel('sub1 marks')

plt.title('Result-1')

plt.subplot(1,2,2)

plt.plot(df['rno'],df['sub2'],linewidth=2,linestyle='--')

plt.xlabel('Roll no')

plt.ylabel('sub2 marks')

plt.title('Result-2')

plt.show()

1. Extract data of ‘rno, name, sub1, sub2, sub3, sub4, sub5’ directly from SQLite database and create dataframe using it (Use DataFrame()). And perform following tasks:

import sqlite3

import pandas as pd

import matplotlib.pyplot as plt

conn=sqlite3.connect('stud.db')

query = "SELECT rno, name, sub1, sub2, sub3, sub4, sub5 FROM student"

data = pd.read\_sql\_query(query, conn)

conn.close()

df = pd.DataFrame(data)

* 1. Draw bar chart for Sub5 with respect to Name 3.1.1.Give proper headings

plt.bar(df['name'], df['sub5'], color='blue')

plt.title('Bar Chart of Sub5 Marks with Respect to Name')

plt.xlabel('Name')

plt.ylabel('Marks in Sub5')

plt.show()

1.2. Draw scatter chart for Sub1, Sub2, Sub3, Sub4, Sub5 with different formatting.

3.2.1.Give proper headings.

plt.scatter(df['rno'], df['sub1'], color='blue', marker='o', label='Sub1')

plt.scatter(df['rno'], df['sub2'], color='black', marker='s', label='Sub2')

plt.scatter(df['rno'], df['sub3'], color='green', marker='^', label='Sub3')

plt.scatter(df['rno'], df['sub4'], color='red', marker='x', label='Sub4')

plt.scatter(df['rno'], df['sub5'], color='orange', marker='D', label='Sub5')

plt.title('Scatter Chart of Subjects Marks with Respect to Roll Numbers') plt.xlabel('Roll Numbers')

plt.ylabel('Marks')

plt.legend()

plt.show()

* 1. Draw histogram chart for Sub4’s marks with proper heading.

plt.hist(df['sub4'], bins=10, color='skyblue', edgecolor='black')

plt.title(' Histogram Chart of Sub5 Marks with Respect to Name')

plt.xlabel('Name')

plt.ylabel('Marks in Sub4')

plt.xticks(df[rno])

plt.show()

1.4. Draw histogram chart for Sub2 as per [0,5,10,15,20,25,30,35,40,45,50] intervals.

bins = [0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50]

plt.hist(df['sub2'], bins=bins, color='skyblue', edgecolor='black')

plt.title('Histogram of Sub2 Marks')

plt.xlabel('Marks')

plt.ylabel('Frequency')

plt.xticks(df[rno])

plt.show()