**CO5-PYTHON SQL CONNECTIVITY ASSIGNMENT**

**CODE:**

import sys;

import sqlite3;

import re;

conn=sqlite3.connect("freefire.db")

cur=conn.cursor()

class freefire:

def \_\_init\_\_(self,version,lastupdated,noofusers,gamesize,ftype):

self.\_\_version=version

self.\_\_lastupdated=lastupdated

self.\_\_noofusers=noofusers

self.\_\_gamesize=gamesize

self.\_\_type=ftype

def getversion(self):

return self.\_\_version

def getlastupdated(self):

return self.\_\_lastupdated

def getnoofusers(self):

return self.\_\_noofusers

def getgamesize(self):

return self.\_\_gamesize

def gettype(self):

return self.\_\_type

def setversion(self, value):

self.\_\_version = value

def setlastupdated(self, value):

self.\_\_lastupdated = value

def setnoofusers(self, value):

self.\_\_noofusers = value

def setgamesize(self, value):

self.\_\_gamesize = value

def settype(self, value):

self.\_\_ftype = value

def checkgamesize(self):

if(int(self.\_\_gamesize)<2):

return True

else:

print("GAME SIZE EXCEEDED")

return False

class player(freefire):

def \_\_init\_\_(self,version,lastupdated,noofusers,gamesize,ftype,pname,pid,age,level):

self.\_\_pname=pname

self.\_\_pid=pid

self.\_\_age=age

self.\_\_level=level

super(player, self).\_\_init\_\_(version,lastupdated,noofusers,gamesize,ftype)

def getpname(self):

return self.\_\_pname

def getpid(self):

return self.\_\_pid

def getage(self):

return self.\_\_age

def getlevel(self):

return self.\_\_level

def setpname(self, value):

self.\_\_pname = value

def setpid(self, value):

self.\_\_pid = value

def setage(self, value):

self.\_\_age = value

def setlevel(self, value):

self.\_\_level = value

def checkage(self):

if(int(self.\_\_age)>18):

return True

else:

print("NOT ELIGIBLE")

return False

class maintenance(freefire):

def \_\_init\_\_(self,version,lastupdated,noofusers,gamesize,ftype,mid,mname,time,salary):

self.\_\_mid=mid

self.\_\_mname=mname

self.\_\_time=time

self.\_\_salary=salary

super(maintenance, self).\_\_init\_\_(version,lastupdated,noofusers,gamesize,ftype) # calling base class constructor

def getmid(self):

return self.\_\_mid

def getmname(self):

return self.\_\_mname

def gettime(self):

return self.\_\_time

def getsalary(self):

return self.\_\_salary

def setmid(self, value):

self.\_\_mid = value

def setmname(self, value):

self.\_\_mname = value

def settime(self, value):

self.\_\_time = value

def setsalary(self, value):

self.\_\_salary = value

def incrementsalary(self):

if(int(self.\_\_time)<60):

self.\_\_salary=self.\_\_salary+25000

def displayincrementsalary(self):

print('Salary:',self.\_\_salary)

class gamedetails(player):

def \_\_init\_\_(self,version,lastupdated,noofusers,gamesize,ftype,pname,pid,age,level,gid,noofkills,noofdeaths,noofvictory,noofloses):

self.\_\_gid=gid

self.\_\_noofkills=noofkills

self.\_\_noofdeaths=noofdeaths

self.\_\_noofvictory=noofvictory

self.\_\_noofloses=noofloses

super(gamedetails, self).\_\_init\_\_(version,lastupdated,noofusers,gamesize,ftype,pname,pid,age,level)

def getgid(self):

return self.\_\_gid

def getnoofkills(self):

return self.\_\_noofkills

def getnoofdeaths(self):

return self.\_\_noofdeaths

def getnoofvictory(self):

return self.\_\_noofvictory

def getnoofloses(self):

return self.\_\_noofloses

def setgid(self, value):

self.\_\_gid = value

def setnoofkills(self, value):

self.\_\_noofkills = value

def setnoofdeaths(self, value):

self.\_\_noofdeaths = value

def setnoofloses(self, value):

self.\_\_age = value

def setnoofvictory(self, value):

self.\_\_noofvictory = value

def calculatekbydratio(self):

print('%.2f'%(int(self.\_\_noofkills)/int(self.\_\_noofdeaths)))

class hello:

n=0

ch='n'

def options(self):

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("GARENA GAMING DETAILS")

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("1. Add New details")

print("2. View Player Details")

print("3. View Maintenance Details")

print("4. View freefire Details")

print("5. Exit")

n=int(input("Select your choice"))

if n==1:

self.insertrecord()

elif n==2:

self.viewallplayer()

elif n==3:

self.viewallmaintenance()

elif n==4:

self.viewfreefiredetails()

elif n==5:

print("LET'S CHILL WITH GAMING EXPERIENCE")

sys.exit

def insertrecord(self):

try:

#conn.execute('''drop table freefire;''')

#conn.execute('''drop table player;''')

#conn.execute('''drop table maintenance;''')

#conn.execute('''drop table gamedetails;''')

#conn.execute('''CREATE TABLE player (PNAME TEXT NOT NULL,PID INT NOT NULL,AGE INT NOT NULL,LEVEL INT NOT NULL);''')

#conn.execute('''CREATE TABLE maintenance (MID INT NOT NULL,MNAME TEXT NOT NULL,TIME INT NOT NULL,SALARY INT NOT NULL);''')

#conn.execute('''CREATE TABLE freefire (VERSION INT NOT NULL,LASTUPDATED TEXT NOT NULL,NOOFUSERS INT NOT NULL,GAMESIZE INT NOT NULL,TYPE TEXT NOT NULL);''')

#conn.execute('''CREATE TABLE gamedetails (GID INT NOT NULL,NOOFKILLS INT NOT NULL,NOOFDEATHS INT NOT NULL,NOOFVICTORY INT NOT NULL, NOOFLOSES INT NOT NULL);''')

version=input("Enter version")

lastupdated = input("Enter lastupdated")

noofusers=input("Enter noofusers")

gamesize=input("Enter gamesize")

ftype=input("Enter type (player/maintenance)")

obj=freefire(version,lastupdated,noofusers,gamesize,ftype)

conn.execute("""insert into freefire(version,lastupdated,noofusers,gamesize,type)

values (?,?,?,?,?)""",(obj.getversion(),obj.getlastupdated(),obj.getnoofusers(),obj.getgamesize(),obj.gettype()))

#to check gamesize

if(obj.checkgamesize()):

print("GAME SIZE NOT EXCEEDED")

else:

print("GAME SIZE EXCEEDED")

hello.options(self)

if(obj.gettype() =="player"):

pname=input("Enter name")

pid=input("Enter pid")

age=int(input("Enter age"))

level=int(input("Enter level"))

pobj=player(version,lastupdated,noofusers,gamesize,ftype,pname,pid,age,level)

conn.execute("insert into player(pname,pid,age,level) values(?,?,?,?)",(pobj.getpname(),pobj.getpid(),pobj.getage(),pobj.getlevel()))

if(pobj.checkage()):

print("ELIGIBLE")

gid = input("Enter gameid")

noofkills = input("Enter noofkills")

noofdeaths = input("Enter noofdeaths")

noofvictory = input("Enter noofvictory")

noofloses = input("Enter noofloses")

gobj = gamedetails(version,lastupdated,noofusers,gamesize,ftype,pname,pid,age,level,gid,noofkills,noofdeaths,noofvictory,noofloses)

conn.execute("""insert into gamedetails(gid,noofkills,noofdeaths,noofvictory,noofloses)

values (?,?,?,?,?)""",(gobj.getgid(),gobj.getnoofkills(),gobj.getnoofdeaths(),gobj.getnoofvictory(),gobj.getnoofloses()))

else:

print("NOT ELIGIBLE")

hello.options(self)

choice = input("Do you want game details of player(y/n)")

if(choice=='y'):

g = int(input("Enter game id"))

cur.execute("select \* from gamedetails where gid=?",(g,))

for n in cur.fetchall():

print(n)

gobj.calculatekbydratio()

elif(obj.gettype()=="maintenance"):

mid=input("Enter mid ")

mname=input("Enter mname")

time=(input("Enter time"))

salary=int(input("Enter salary"))

mobj=maintenance(version,lastupdated,noofusers,gamesize,ftype,mid,mname,time,salary)

conn.execute("""insert into maintenance(mid,mname,time,salary)

values (?,?,?,?)""",(mobj.getmid(),mobj.getmname(),mobj.gettime(),mobj.getsalary()))

mobj.incrementsalary()

mobj.displayincrementsalary()

oobj.gooptions()

except sqlite3.DatabaseError as e:

print("Error details: ",e)

finally:

conn.commit()

def viewallplayer(self):

cur.execute("select \* from player")

for n in cur.fetchall():

print(n)

oobj.gooptions()

def viewallmaintenance(self):

cur.execute("select \* from maintenance")

for n in cur.fetchall():

print(n)

oobj.gooptions()

def viewfreefiredetails(self):

cur.execute("select \* from freefire")

for n in cur.fetchall():

print(n)

def gooptions(self):

ch=input("Do you wish to continue(y/n)")

if(ch=='y' or ch=='Y'):

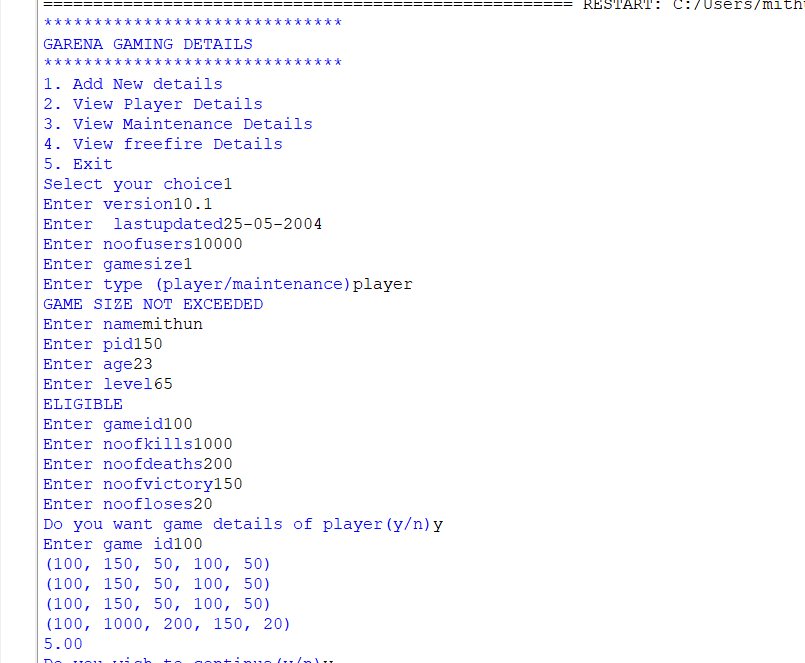
hello.options(self)

oobj=hello()

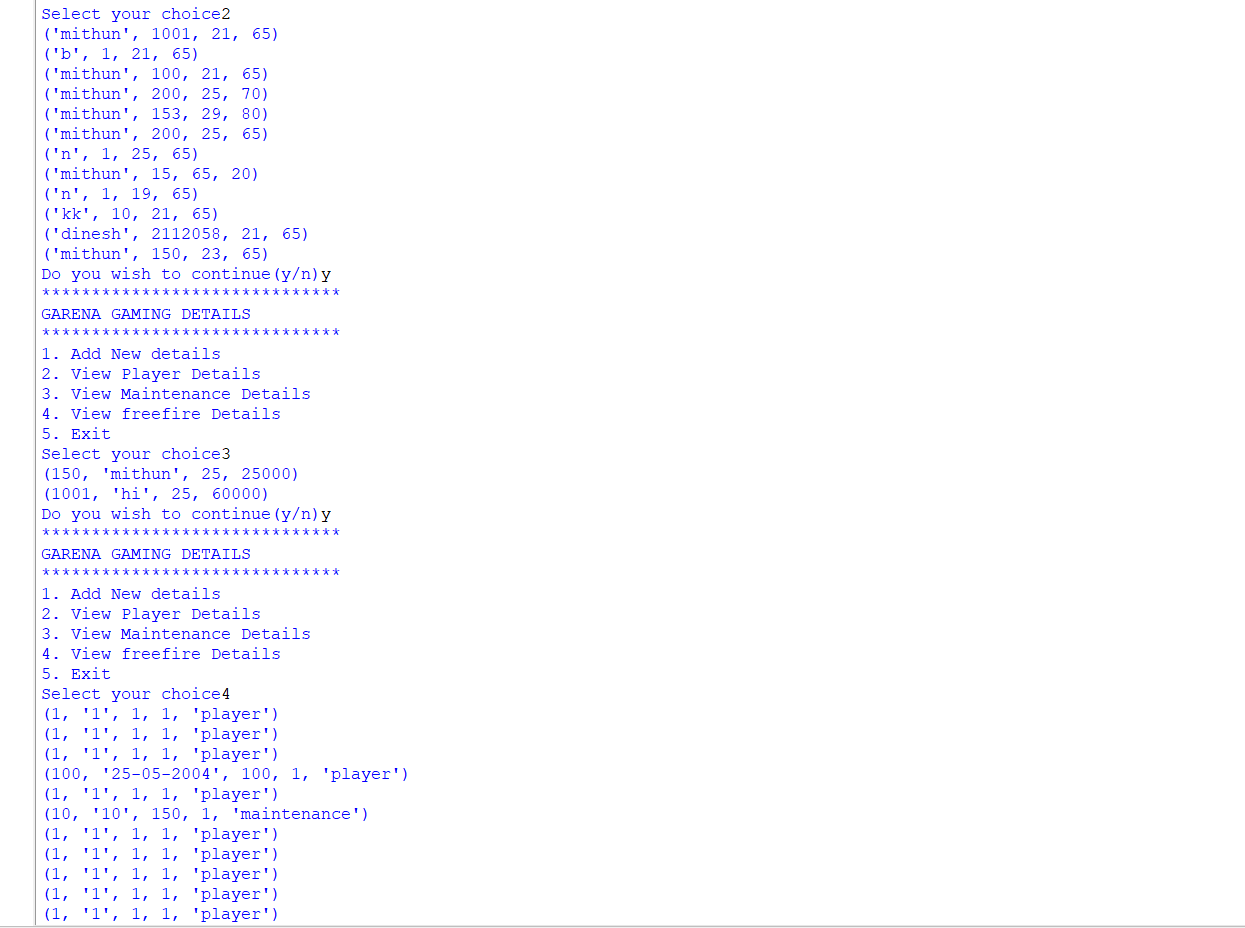
oobj.options()

oobj.gooptions()

**OUTPUT:**

****

****

****