COMPUTER SCIENCE PROJECT

AMUSEMENT PARK

DONE BY:

SHRADHA RAJ P S

INDEX

SNO	CONTENT	PAGE NO
1	Hardware and Software requirements	3
2	Theoretical background	3
3	Program description	8
4	Functions used	8
5	Program	10
6	Output	20
7	Advantages and Disadvantages	33
8	Future enhancement of the Project	34
9	Bibliography	35

HARDWARE REQUIREMENTS

- **Operating System- Windows 10**
- **Processor- Intel Core 15**

SOFTWARE REQUIREMENTS

- **Python [Version 3.7]**
- **MySQL [Version 5.7]**

THEORETICAL BACKGROUND

WHAT IS PYTHON?

Python is an interpreter. It is a high-level and general- purpose programming language which emphasizes code readability with its notable use of significant whitespace. Its language constructs an object-oriented approach which aims to help programmers write a clear, logical code for small and large-scale projects. Python is dynamically typed and it supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python interpreters are supported for mainstream operating systems and available for a few more. A global community of programmers develops and maintains C Python, a free and open-source reference implementation. A non- profit organization, the Python Software Foundation, manages and directs resources for Python and C Python development.

HISTORY OF PYTHON:

The programming language Python was conceived in the late 1980s, and its implementation was started in December 1989 by Guido van Rossum at CWI in the Netherlands as a successor to ABC Capable of exception handling and interfacing with the Amoeba operating system. Van Rossum is Python's principal author, and his continuing central role in deciding the direction of Python is reflected in the title given to him by the Python community, Benevolent Dictator for Life (BDFL). (However, van Rossum stepped down as leader on July 12, 2018. Python 2.0 was released on October 16, 2000, with many major new features for memory management and support for Unicode.

However, the most important change was to the development process itself, with a shift to a more transparent and community-backed process. Python 3.0, a major, backwards-incompatible release, was released on December 3, 2008 after a long period of testing. Many of its major features have also been back ported to the backwards-compatible, while by now unsupported, Python 2.6 and 2.7.

WHAT IS A DATABASE?

Database is a collection of information that is organised so as to access them easily and quickly. In a relational database, the digital information are arranged into rows, columns and tables which are indexed to access the relevant information. There are different kinds of databases ranging for the most approached relational database, to a distributed database, cloud database, graph database or NoSQL database.

RELATIONAL DATABASE:

A relational database, invented by E.F. Codd at IBM in 1970, is a tabular database in which data is defined so that it can be reorganized and accessed in a number of different ways. Relational databases are made up of a set of tables with data that fits into a predefined category. Each table has at least one data category in a column, and each row has at certain data instance for the categories which are defined in the columns. Relational databases are easy to extend, and a new data category can be added after the original database creation without requiring that you modify all the existing applications.

RELATIONAL DATABASE MANAGEMENT SYSTEM:

A relational database management system shortly called as RDBMS is a Database management system that is designed specifically for relational databases. It is the software that executes queries on the data, including adding, updating, and searching for values. An RDBMS may also provide a visual representation of the data. For example, it may display data in a tables like a spreadsheet, allowing you to view and even edit individual values in the table. Some RDMBS programs allow you to create forms that can streamline entering, editing, and deleting data. Most well known DBMS applications fall into the RDBMS category. Examples include Oracle Database, MySQL, Microsoft SQL Server, and IBM DB2. Some of these programs support non- relational databases, but they are primarily used for relational database management.

WHAT IS SQL?

SQL (Structured Query Language) is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It is particularly useful in handling structured data, i.e. data incorporating relations among entities and variables. SQL offers two main advantages over older read-write APIs. Firstly, it introduced the concept of accessing many records with one single command. Secondly, it eliminates the need to specify how to reach a record, e.g. with or without an index. HISTORY OF SQL: SQL was initially developed at IBM by Donald D. Chamberlin and Raymond F. Boyce after learning about the relational model from Edgar F. Codd in the early 1970s. This version, initially called SEQUEL (Structured English Query Language), was designed to manipulate and retrieve data stored in IBM's original quasi-relational database management system, System R, which a group at IBM San Jose Research Laboratory had developed during the 1970s. In the late 1970s, Relational Software, Inc. (now Oracle Corporation) saw the potential of the concepts described by Codd, Chamberlin, and Boyce, and developed their own SQL-based RDBMS with aspirations of selling it to the U.S. Navy, Central Intelligence Agency, and other U.S. government agencies. In June 1979, Relational Software, Inc. introduced the first com VAX computers. By 1986, ANSI and ISO standard groups officially adopted the standard "Database Language SQL" language definition. New versions of the standard were published in 1989, 1992, 1996, 1999, 2003, mercially available implementation of SQL, Oracle V2 (Version2) for 2006,2008, 2011 and, most recently, 2016.

PROGRAM DESCRIPTION

This program is made to create a bill for the customers in an amusement park named “The Magic Kingdom”. In this program, three tables can be created (one for details about the customer and the other two for details about the games available). There are various functions for the user to do a specific task of his/her choice. For example, choice 1 is to create the required tables, choice 2 is to insert values into one of the tables, etc … By entering the appropriate choice the particular function is called and is executed. If choice is out of range the loop is broken.

FUNCTIONS CREATED

- **createtables():**
  This function creates any of the three tables (Customers, Land_games, Water_games depending on the user’s choice. It also asks if the user wants to create another table and creates it if the user says yes.

- **insertcust():**
  This function is used to insert records into the table customers .The user is required to input the details of the customers.

- **insertwgame():**
  This function is used to insert records into the table Water_games.

- **insertlgame():**
  This function is used to insert records into the table Land_games.

- **display_customers():**
  This function displays all the records in the table Customers in a way that is easy to understand for the user.

- **display_Watergames():**
  This function displays all the records in the table Water_games.

- **display_Landgames():**
  This function displays all the records in the table Land_games.

- **join_tables():**
  This function is used to join the tables customers and Land_games or the tables customers and Water_games according to the user's choice.

- **bill():**
  This function creates the bill for a specific customer when his/her customer ID is given.

- **delete_tables():**
  This function deletes any of the three tables depending on the

user’s choice. It asks if the user wants to delete another table and then deletes it if the user says yes.

PROGRAM

```
import mysql.connector as sqltor
con=sqltor.connect(host="localhost",user="root",password="devi",d
atabase="Amusement_Park")
if con.is_connected():
	print("Connected with mysql database successfully")
else:
	 print("Connection Error. Please try again.")
cursor=con.cursor()

def createtables():
	ch="y"
	while ch.lower()=="y":
		print("Choose the table to be created from the menu")
		print("1.Customers")
		print("2.Water_Games")
		print("3.Land_Games")
		x=int(input("Enter your choice"))
		if x==1:
			query="Create table if not exists customers(sno integer
not null primary key ,custid char(4), custname
varchar(30),custgender varchar(10),custage integer, game_category
varchar(20), gameid1 char(4))"
			cursor.execute(query)
		if x==2:
```

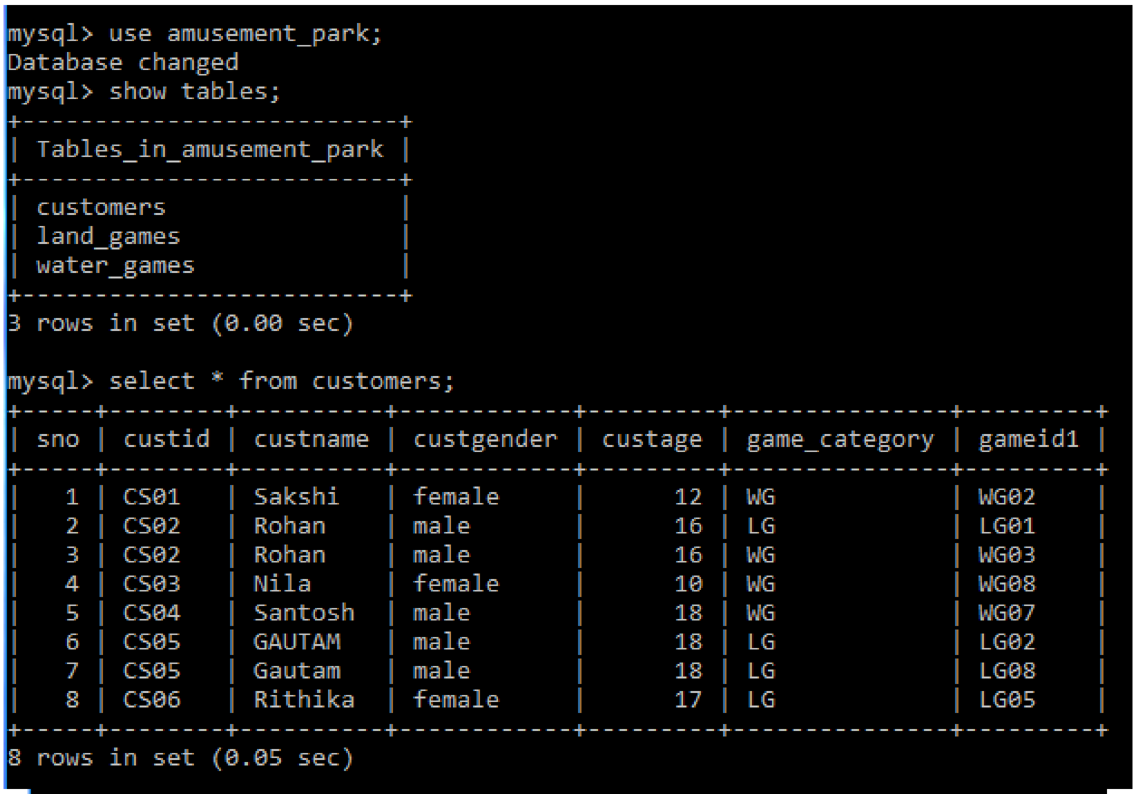
```
                    query="Create table if not exists Water_Games(gameid
char(4) not null primary key,gamename varchar(20),game_category
varchar(20), min_age integer, entryfees integer)"
                    cursor.execute(query)
              if x==3:
                    query="Create table if not exists Land_Games(gameid
char(4) not null primary key,gamename varchar(20),game_category
varchar(20), min_age integer , entryfees integer)"
                    cursor.execute(query)
              print("Table",x,"created")
              ch=input("Do you want to create another table?(y /n)")


def insertcust():
       ch="y"
       while ch.lower()=="y":
              sno=int(input("Enter the serial number"))
              custid=input("Enter customer ID")
              custname=input("Enter customer's name")
              custgender=input("Enter customer's gender")
              custage=int(input("Enter customer's age "))
              gamecat=input("Enter the category of game
chosen(LG/WG)")
              gameid=input("Enter game ID")
              query="insert into
customers(sno,custid,custname,custgender,custage,game_category,
gameid1)
values({},'{}','{}','{}',{},'{}','{}')".format(sno,custid,custname,custgender
,custage,gamecat,gameid)
              cursor.execute(query)
              con.commit()
              ch=input("Do you want to enter another record?(y/n)")


def insertwgame():
        query1="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG01","Water
Wars","WG",10,1200)
        cursor.execute(query1)
        con.commit()
        query2="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG02","Water
volcano","WG",10,1200)
        cursor.execute(query2)
        con.commit()
        query3="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG03","Boating","WG",12,1500)
        cursor.execute(query3)
        con.commit()
        query4="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG04","Frog slide","WG",10,1000)
        cursor.execute(query4)
        con.commit()
        query5="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG05","Rain dance","WG",6,800)
        cursor.execute(query5)
        con.commit()
        query6="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG06","Tornado
Coaster","WG",10,1200)
        cursor.execute(query6)
        con.commit()
        query7="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG07","3 Lane
slides","WG",10,1200)
        cursor.execute(query7)
        con.commit()
        query8="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG08","Swimming
pool","WG",6,1000)
        cursor.execute(query8)
        con.commit()
        query9="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG09","Dome
slide","WG",10,1200)
        cursor.execute(query9)
        con.commit()
        query10="insert into
Water_Games(gameid,gamename,game_category,min_age,entryfee
s)values('{}','{}','{}',{},{})".format("WG10","aqua race","WG",10,1200)
        cursor.execute(query10)
        con.commit()



def insertlgame():
        query1="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG01","Roller Coaster","LG",9,1300)
        cursor.execute(query1)
        con.commit()
        query2="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG02","Ferris Wheel","LG",12,1500)
        cursor.execute(query2)
        con.commit()
        query3="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG03","Haunted
house","LG",13,1000)
        cursor.execute(query3)
        con.commit()
        query4="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG04","Flat rides","LG",10,1200)
        cursor.execute(query4)
        con.commit()
        query5="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG05","Bumper cars","LG",8,800)
        cursor.execute(query5)
        con.commit()
        query6="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG06","Sonic colours","LG",5,1000)
        cursor.execute(query6)
        con.commit()
        query7="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG07","Merry Go Road","LG",8,1000)
        cursor.execute(query7)
        con.commit()
        query8="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG08","Free fall","LG",12,1500)
        cursor.execute(query8)
        con.commit()
        query9="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG09","Haunted train","LG",10,1200)
        cursor.execute(query9)
        con.commit()
        query10="insert into
Land_Games(gameid,gamename,game_category,min_age,entryfees)
values('{}','{}','{}',{},{})".format("LG10","Shoot and win","LG",9,1000)
        cursor.execute(query10)
        con.commit()



def display_customers():
    query="select * from customers"
    cursor.execute(query)
    x=cursor.fetchall()
    for i in x:
        print("Entry number:",i[0])
        print("Customer ID:",i[1])
        print("Customer name:",i[2])
        print("Customer gender:",i[3])
        print("Customer age:",i[4])
        print("Game category:",i[5])
        print("Game ID:",i[6])
        print()

def display_Watergames():
    query="select * from Water_Games"
    cursor.execute(query)
    x=cursor.fetchall()
    for i in x:
        print("Game ID:",i[0])
        print("Game name:",i[1])
        print("Game category:",i[2])
        print("Minimum age:",i[3])
        print("Entry fees:",i[4])
        print()
```

```
def display_Landgames():
    query="select * from Land_Games"
    cursor.execute(query)
    x=cursor.fetchall()
    for i in x:
        print("Game ID",i[0])
        print("Game name:",i[1])
        print("Game category:",i[2])
        print("Minimum age:",i[3])
        print("Entry fees:",i[4])
        print()

def join_tables():
    print("Choose the tables that you'd like to join from the menu")
    print("1.Customers and Water_Games")
    print("2.Customers and Land_Games")
    x=int(input("Enter your choice"))
    if x==1:
        query="select custid,custname, gameid,gamename from
Customers,Water_Games where
customers.gameid1=Water_Games.gameid and
Customers.custage>=Water_Games.min_age"
        cursor.execute(query)
        x=cursor.fetchall()
        for i in x:
            print(i)
    if x==2:
        query="select custid,custname,gameid,gamename from
Customers,Land_Games where
customers.gameid1=Land_Games.gameid and
Customers.custage>=Land_Games.min_age"
        cursor.execute(query)
        x=cursor.fetchall()
        for i in x:
            print(i)
```

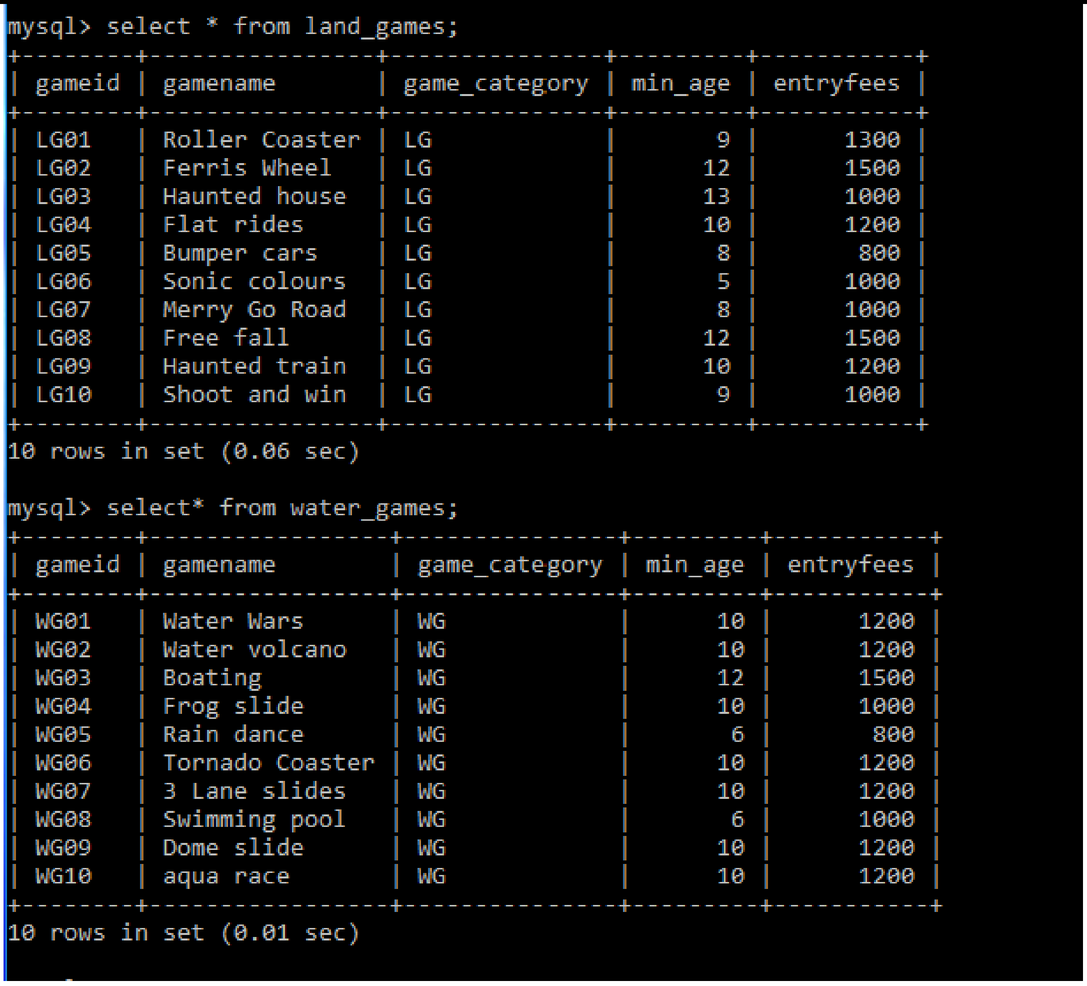
```
def bill():
        ch="y"
        while ch.lower()=="y":
                x=input("Enter the customer's ID whose bill is to be
prepared")
                y=input("Enter the Customer's name")
                query1="select custname,gamename,entryfees from
customers, Land_Games where
customers.gameid1=Land_Games.gameid and custid='{}'".format(x)
                cursor.execute(query1)
                a=cursor.fetchall()
                query2="select custname,gamename,entryfees from
customers, Water_Games where
customers.gameid1=Water_Games.gameid and custid='{}'".format(x)
                cursor.execute(query2)
                b=cursor.fetchall()
                print("≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈MAGIC KINGDOM PARK≈≈≈≈≈≈≈≈≈≈
≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈")
                print("Name:",y)
                s1=0
                s2=0
                for i in a:
                    print("Entryfees of the game",i[1],": ₹",i[2])
                    lsum=s1+i[2]
                    s1=lsum
                for j in b:
                    print("Entryfees of the game",j[1],": ₹",j[2])
                    wsum=s2+j[2]
                    s2=wsum
                total=lsum +wsum
                gst=total*0.18
                print("Actual price: ₹",total)
                print("GST amount: ₹",gst)
```

```
            print("Total amount : ₹",total+gst)
            print()
            print("Thank you for coming!")
            print("Have a great day!")
            print()
            print("☆｡:*•.────────  ❁ ❁ ────────.•*:｡☆")
            print()
            print()
            ch=input("Do you want to get the bill for another
customer?(y/n)")

def delete_tables():
        ch="y"
        while ch.lower()=="y":
                print("Choose the table that you'd like to delete from the
menu :")
                print("1.Customers")
                print("2.Water_Games")
                print("3.Land_Games")
                x=int(input("Enter your choice"))
                if x==1:
                        cursor.execute("drop table if exists Customers")
                        con.commit()
                if x==2:
                        cursor.execute("drop table if exists Water_Games")
                        con.commit()
                if x==3:
                        cursor.execute("drop table if exists Land_Games")
                        con.commit()
                print("Table ",x,"deleted")
                ch=input("Do you want to delete another table?(y/n)")

z="y"
while z.lower()=="y":
    print("Choose the task to be done from the menu:")
    print("1.To Create the required tables (Customers, Water_Games,
Land_Games)")
    print("2.To insert records into the table 'Customers' ")
    print("3.To insert records into the table 'Water_Games' ")
    print("4.To insert records into the table 'Land_Games' ")
    print("5.To display the records in the table 'Customers' ")
    print("6.To display the records in the table 'Water_Games' ")
    print("7.To display the records in the tabe 'Land_Games' ")
    print("8.To join two tables and display the content")
    print("9.To issue the bill for the customer")
    print("10.To delete a table from the database")
    c=int(input("Enter your choice"))
    if c==1:
        createtables()
    if c==2:
        insertcust()
    if c==3:
        insertwgame()
    if c==4:
        insertlgame()
    if c==5:
        display_customers()
    if c==6:
        display_Watergames()
    if c==7:
        display_Landgames()
    if c==8:
        join_tables()
    if c==9:
        bill()
    if c==10:
        delete_tables()
    print()
 z=input("Do you want to do another task?(y/n)"
con.close()
```

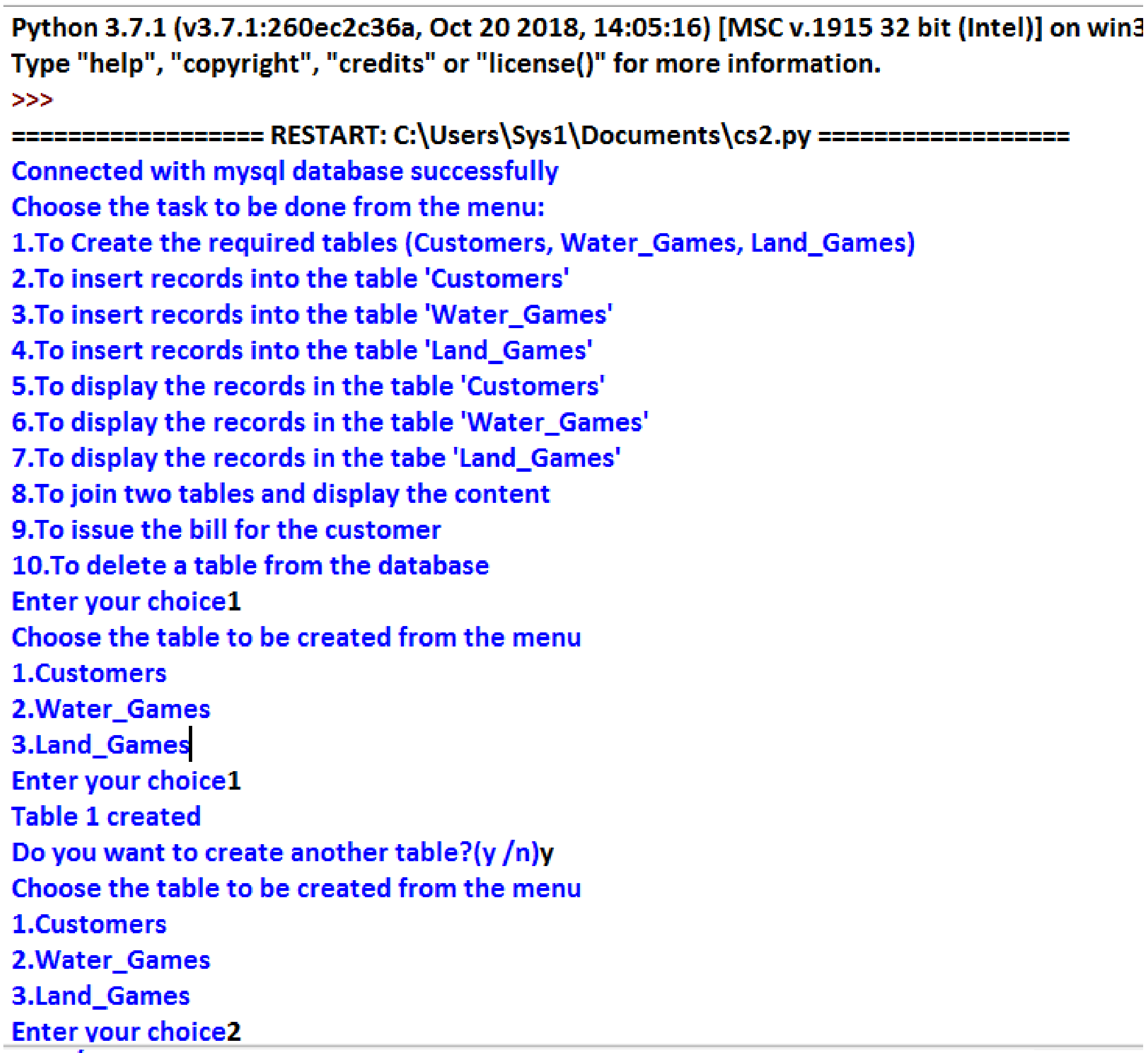
OUTPUT



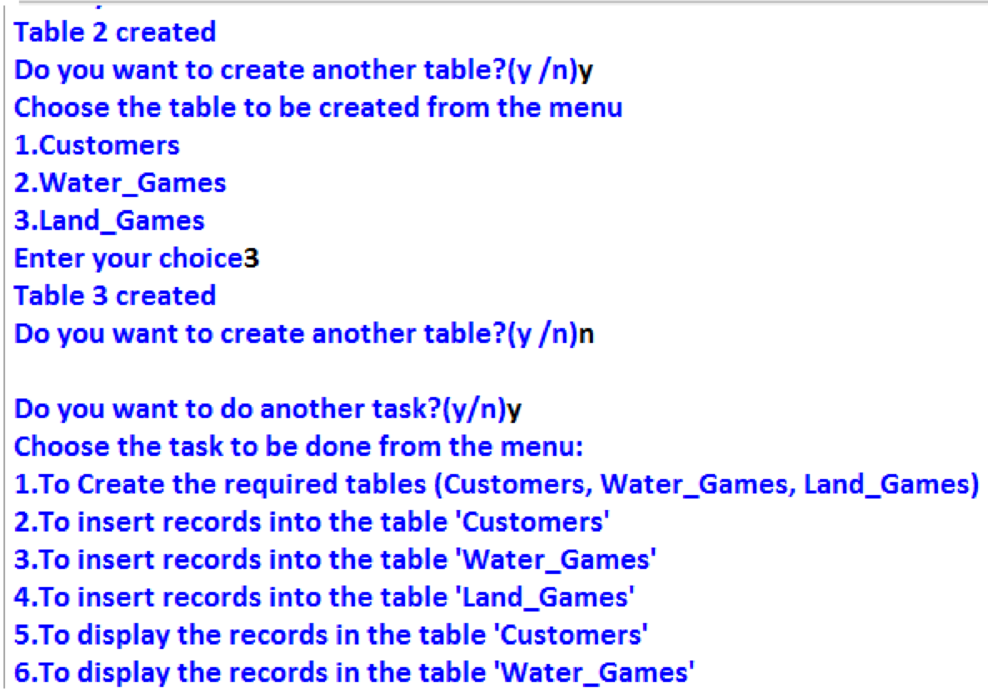
```
mysql> use amusement_park;
Database changed
mysql> show tables;
+--------------------------+
| Tables_in_amusement_park |
+--------------------------+
| customers                |
| land_games               |
| water_games              |
+--------------------------+
3 rows in set (0.00 sec)

mysql> select * from customers;
+-----+--------+----------+------------+---------+---------------+---------+
| sno | custid | custname | custgender | custage | game_category | gameid1 |
+-----+--------+----------+------------+---------+---------------+---------+
|   1 | CS01   | Sakshi   | female     |      12 | WG            | WG02    |
|   2 | CS02   | Rohan    | male       |      16 | LG            | LG01    |
|   3 | CS02   | Rohan    | male       |      16 | WG            | WG03    |
|   4 | CS03   | Nila     | female     |      10 | WG            | WG08    |
|   5 | CS04   | Santosh  | male       |      18 | WG            | WG07    |
|   6 | CS05   | GAUTAM   | male       |      18 | LG            | LG02    |
|   7 | CS05   | Gautam   | male       |      18 | LG            | LG08    |
|   8 | CS06   | Rithika  | female     |      17 | LG            | LG05    |
+-----+--------+----------+------------+---------+---------------+---------+
8 rows in set (0.05 sec)
```



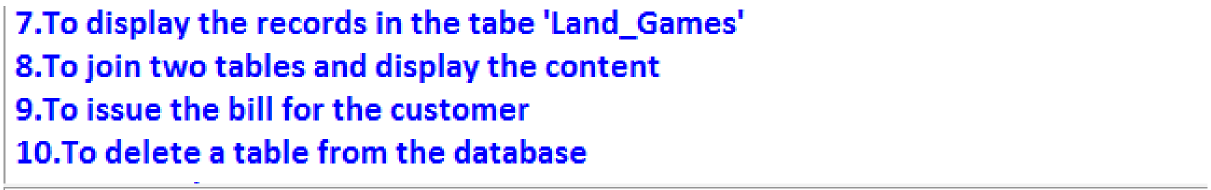
```
mysql> select * from land_games;
+--------+----------------+---------------+---------+-----------+
| gameid | gamename       | game_category | min_age | entryfees |
+--------+----------------+---------------+---------+-----------+
| LG01   | Roller Coaster | LG            |       9 |      1300 |
| LG02   | Ferris Wheel   | LG            |      12 |      1500 |
| LG03   | Haunted house  | LG            |      13 |      1000 |
| LG04   | Flat rides     | LG            |      10 |      1200 |
| LG05   | Bumper cars    | LG            |       8 |       800 |
| LG06   | Sonic colours  | LG            |       5 |      1000 |
| LG07   | Merry Go Road  | LG            |       8 |      1000 |
| LG08   | Free fall      | LG            |      12 |      1500 |
| LG09   | Haunted train  | LG            |      10 |      1200 |
| LG10   | Shoot and win  | LG            |       9 |      1000 |
+--------+----------------+---------------+---------+-----------+
10 rows in set (0.06 sec)

mysql> select* from water_games;
+--------+-----------------+---------------+---------+-----------+
| gameid | gamename        | game_category | min_age | entryfees |
+--------+-----------------+---------------+---------+-----------+
| WG01   | Water Wars      | WG            |      10 |      1200 |
| WG02   | Water volcano   | WG            |      10 |      1200 |
| WG03   | Boating         | WG            |      12 |      1500 |
| WG04   | Frog slide      | WG            |      10 |      1000 |
| WG05   | Rain dance      | WG            |       6 |       800 |
| WG06   | Tornado Coaster | WG            |      10 |      1200 |
| WG07   | 3 Lane slides   | WG            |      10 |      1200 |
| WG08   | Swimming pool   | WG            |       6 |      1000 |
| WG09   | Dome slide      | WG            |      10 |      1200 |
| WG10   | aqua race       | WG            |      10 |      1200 |
+--------+-----------------+---------------+---------+-----------+
10 rows in set (0.01 sec)
```



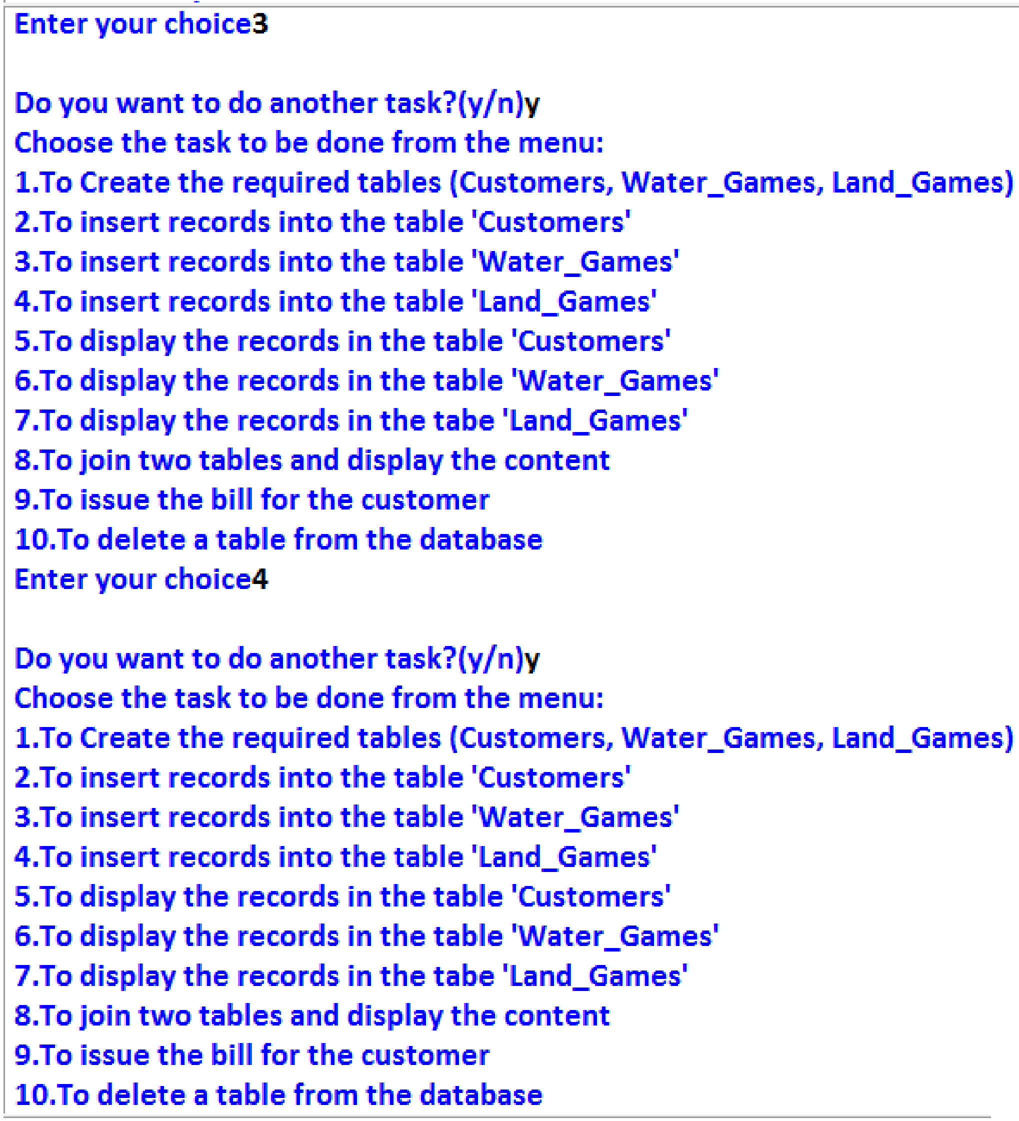
```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Intel)] on win3
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: C:\Users\Sys1\Documents\cs2.py ==================
Connected with mysql database successfully
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice1
Choose the table to be created from the menu
1.Customers
2.Water_Games
3.Land_Games
Enter your choice1
Table 1 created
Do you want to create another table?(y /n)y
Choose the table to be created from the menu
1.Customers
2.Water_Games
3.Land_Games
Enter your choice2
```



```
Table 2 created
Do you want to create another table?(y /n)y
Choose the table to be created from the menu
1.Customers
2.Water_Games
3.Land_Games
Enter your choice3
Table 3 created
Do you want to create another table?(y /n)n

Do you want to do another task?(y/n)y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
```

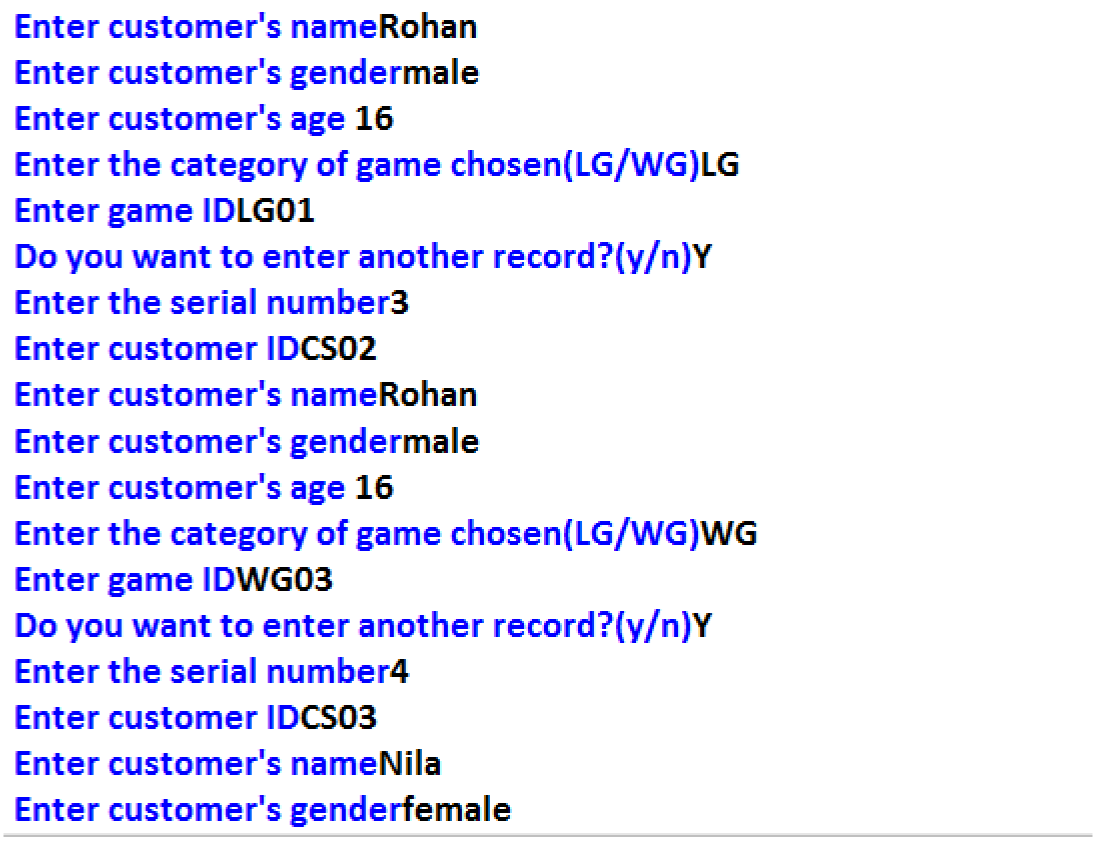


```
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
```

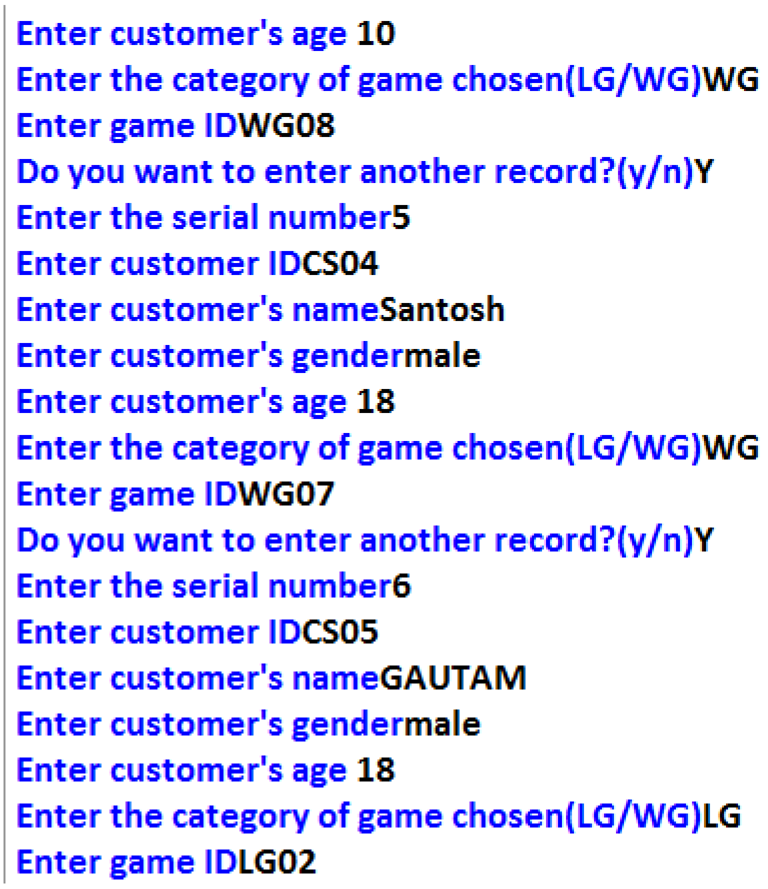


```
Enter your choice3

Do you want to do another task?(y/n)y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice4

Do you want to do another task?(y/n)y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
```

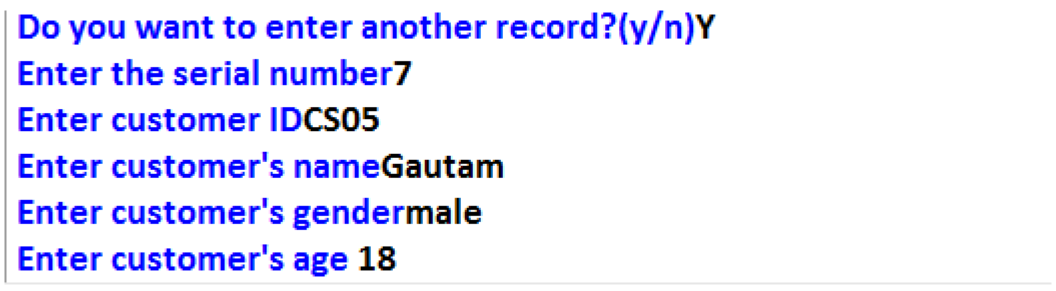
```
Enter your choice2
Enter the serial number1
Enter customer IDCS01
Enter customer's nameSakshi
Enter customer's genderfemale
Enter customer's age 12
Enter the category of game chosen(LG/WG)WG
Enter game IDWG02
Do you want to enter another record?(y/n)Y
Enter the serial number2
Enter customer IDCS02
```



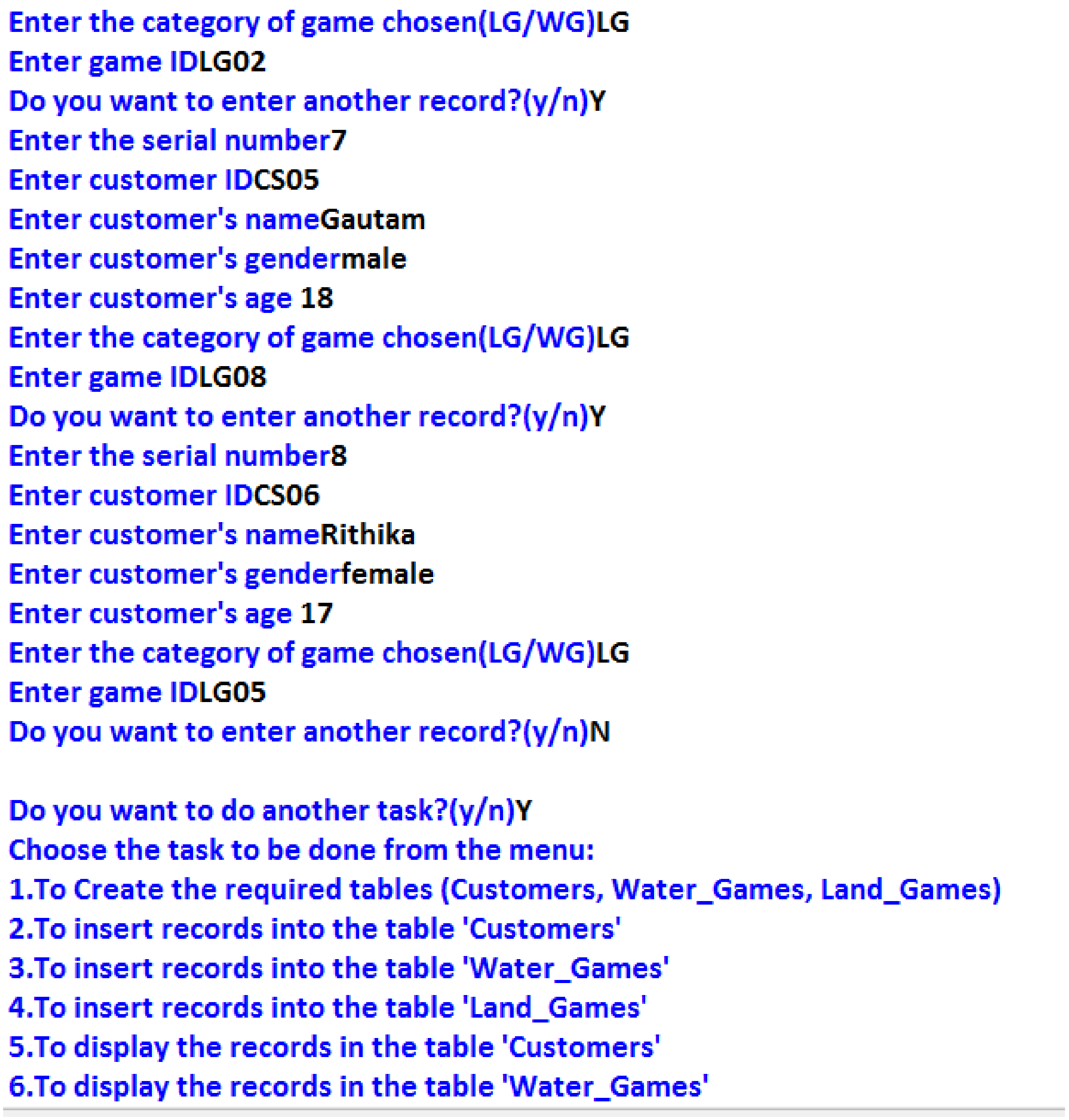
```
Enter customer's nameRohan
Enter customer's gendermale
Enter customer's age 16
Enter the category of game chosen(LG/WG)LG
Enter game IDLG01
Do you want to enter another record?(y/n)Y
Enter the serial number3
Enter customer IDCS02
Enter customer's nameRohan
Enter customer's gendermale
Enter customer's age 16
Enter the category of game chosen(LG/WG)WG
Enter game IDWG03
Do you want to enter another record?(y/n)Y
Enter the serial number4
Enter customer IDCS03
Enter customer's nameNila
Enter customer's genderfemale
```



```
Enter customer's age 10
Enter the category of game chosen(LG/WG)WG
Enter game IDWG08
Do you want to enter another record?(y/n)Y
Enter the serial number5
Enter customer IDCS04
Enter customer's nameSantosh
Enter customer's gendermale
Enter customer's age 18
Enter the category of game chosen(LG/WG)WG
Enter game IDWG07
Do you want to enter another record?(y/n)Y
Enter the serial number6
Enter customer IDCS05
Enter customer's nameGAUTAM
Enter customer's gendermale
Enter customer's age 18
Enter the category of game chosen(LG/WG)LG
Enter game IDLG02
```



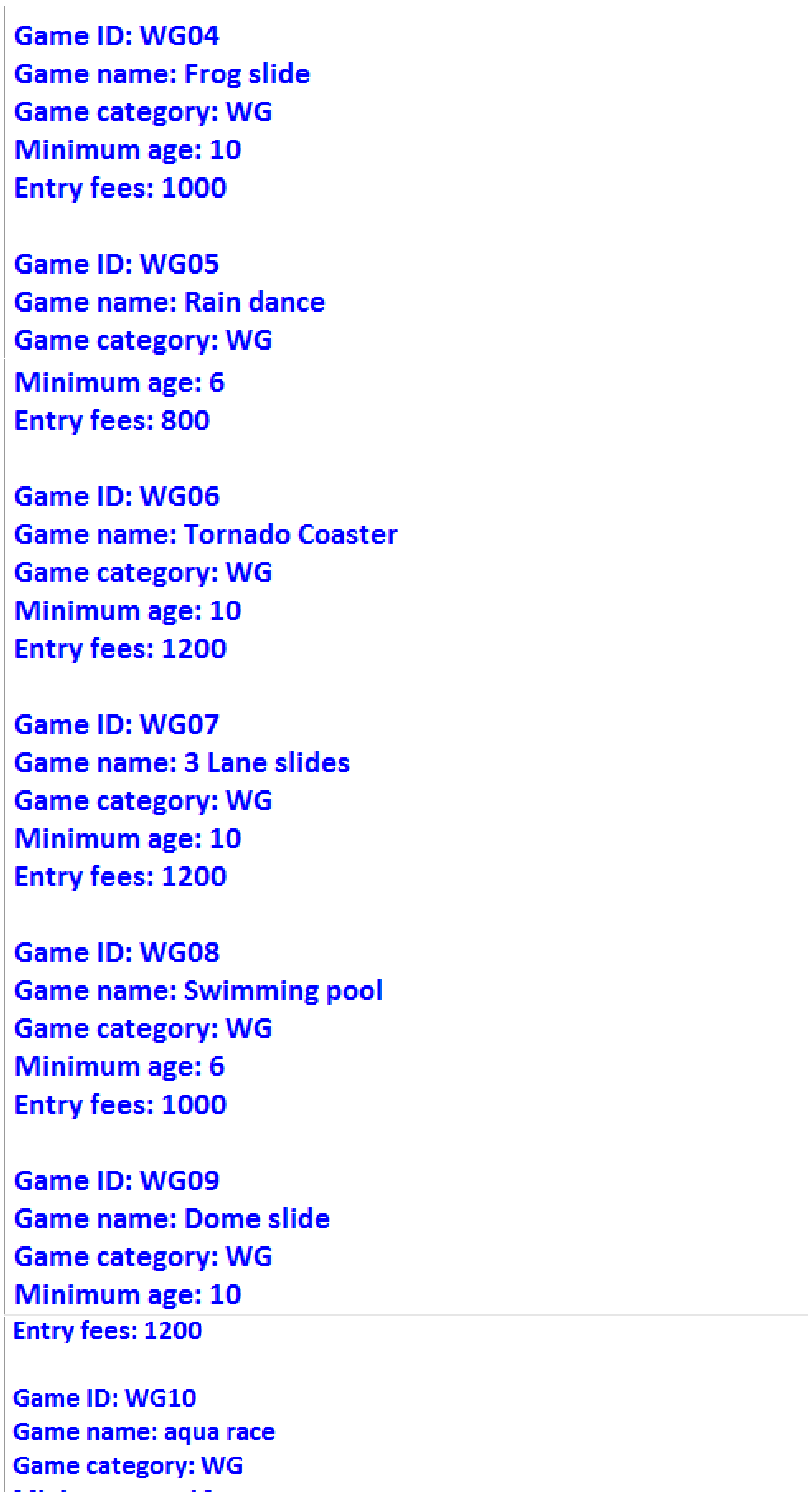
```
Do you want to enter another record?(y/n)Y
Enter the serial number7
Enter customer IDCS05
Enter customer's nameGautam
Enter customer's gendermale
Enter customer's age 18
```



```
Enter the category of game chosen(LG/WG)LG
Enter game IDLG02
Do you want to enter another record?(y/n)Y
Enter the serial number7
Enter customer IDCS05
Enter customer's nameGautam
Enter customer's gendermale
Enter customer's age 18
Enter the category of game chosen(LG/WG)LG
Enter game IDLG08
Do you want to enter another record?(y/n)Y
Enter the serial number8
Enter customer IDCS06
Enter customer's nameRithika
Enter customer's genderfemale
Enter customer's age 17
Enter the category of game chosen(LG/WG)LG
Enter game IDLG05
Do you want to enter another record?(y/n)N

Do you want to do another task?(y/n)Y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
```

```
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice5
Entry number: 1
Customer ID: CS01


 Customer name: Sakshi
 Customer gender: female
 Customer age: 12
 Game category: WG
 Game ID: WG02

 Entry number: 2
 Customer ID: CS02
 Customer name: Rohan
 Customer gender: male
 Customer age: 16
 Game category: LG
 Game ID: LG01

 Entry number: 3
 Customer ID: CS02
 Customer name: Rohan
 Customer gender: male
 Customer age: 16
```

```
Game ID: WG03

Entry number: 4
Customer ID: CS03
Customer name: Nila
Customer gender: female
Customer age: 10
Game category: WG
Game ID: WG08

Entry number: 5
Customer ID: CS04
Customer name: Santosh
Customer gender: male
Customer age: 18
Game category: WG
Game ID: WG07



Entry number: 6
Customer ID: CS05
Customer name: GAUTAM
Customer gender: male
Customer age: 18
Game category: LG
Game ID: LG02

Entry number: 7
Customer ID: CS05
Customer name: Gautam
Customer gender: male
Customer age: 18
Game category: LG
Game ID: LG08

Entry number: 8
Customer ID: CS06
Customer name: Rithika
Customer gender: female
Customer age: 17
Game category: LG
Game ID: LG05


Do you want to do another task?(y/n)Y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice6
Game ID: WG01
Game name: Water Wars
Game category: WG
Minimum age: 10
Entry fees: 1200

Game ID: WG02
Game name: Water volcano
Game category: WG
Minimum age: 10
Entry fees: 1200


Game ID: WG03
Game name: Boating
Game category: WG
Minimum age: 12
Entry fees: 1500
```



```
Game ID: WG04
Game name: Frog slide
Game category: WG
Minimum age: 10
Entry fees: 1000

Game ID: WG05
Game name: Rain dance
Game category: WG
Minimum age: 6
Entry fees: 800

Game ID: WG06
Game name: Tornado Coaster
Game category: WG
Minimum age: 10
Entry fees: 1200

Game ID: WG07
Game name: 3 Lane slides
Game category: WG
Minimum age: 10
Entry fees: 1200

Game ID: WG08
Game name: Swimming pool
Game category: WG
Minimum age: 6
Entry fees: 1000

Game ID: WG09
Game name: Dome slide
Game category: WG
Minimum age: 10
Entry fees: 1200

Game ID: WG10
Game name: aqua race
Game category: WG
```

```
Minimum age: 10
Entry fees: 1200


Do you want to do another task?(y/n)Y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice7
Game ID LG01
Game name: Roller Coaster
Game category: LG
Minimum age: 9
Entry fees: 1300

Game ID LG02
Game name: Ferris Wheel
Game category: LG
Minimum age: 12
Entry fees: 1500

Game ID LG03
Game name: Haunted house
Game category: LG
Minimum age: 13
Entry fees: 1000

Game ID LG04
Game name: Flat rides
Game category: LG
Minimum age: 10
Entry fees: 1200
```

```
Game ID LG05
Game name: Bumper cars
Game category: LG
Minimum age: 8
Entry fees: 800

Game ID LG06
Game name: Sonic colours
Game category: LG
Minimum age: 5
Entry fees: 1000

Game ID LG07
Game name: Merry Go Road
Game category: LG
Minimum age: 8
Entry fees: 1000

Game ID LG08
Game name: Free fall
Game category: LG
Minimum age: 12
Entry fees: 1500

Game ID LG09
Game name: Haunted train
Game category: LG
Minimum age: 10
Entry fees: 1200

Game ID LG10
Game name: Shoot and win
Game category: LG
Minimum age: 9
Entry fees: 1000
```

```
Do you want to do another task?(y/n)Y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice8
Choose the tables that you'd like to join from the menu
1.Customers and Water_Games
2.Customers and Land_Games
Enter your choice1
('CS01', 'Sakshi', 'WG02', 'Water volcano')
('CS02', 'Rohan', 'WG03', 'Boating')
('CS03', 'Nila', 'WG08', 'Swimming pool')
('CS04', 'Santosh', 'WG07', '3 Lane slides')

Do you want to do another task?(y/n)Y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice8
Choose the tables that you'd like to join from the menu
1.Customers and Water_Games
2.Customers and Land_Games
Enter your choice2
('CS02', 'Rohan', 'LG01', 'Roller Coaster')
('CS05', 'GAUTAM', 'LG02', 'Ferris Wheel')
('CS05', 'Gautam', 'LG08', 'Free fall')
('CS06', 'Rithika', 'LG05', 'Bumper cars')
```

```
Do you want to do another task?(y/n)Y
Choose the task to be done from the menu:
1.To Create the required tables (Customers, Water_Games, Land_Games)
2.To insert records into the table 'Customers'
3.To insert records into the table 'Water_Games'
4.To insert records into the table 'Land_Games'
5.To display the records in the table 'Customers'
6.To display the records in the table 'Water_Games'
7.To display the records in the tabe 'Land_Games'
8.To join two tables and display the content
9.To issue the bill for the customer
10.To delete a table from the database
Enter your choice9
Enter the customer's ID whose bill is to be preparedCS02
Enter the Customer's nameRohan
~~~~~~~~~~~~~~~~~~~~~~~~~~~~MAGIC KINGDOM PARK~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Name: Rohan
Entryfees of the game Roller Coaster : ₹ 1300
Entryfees of the game Boating : ₹ 1500
Actual price: ₹ 2800
GST amount: ₹ 504.0
Total amount : ₹ 3304.0

Thank you for coming!
Have a great day!

☆｡:*•.────── ❁ ❁ ──────.•*:｡☆

Do you want to get the bill for another customer?(y/n)y
Enter the customer's ID whose bill is to be preparedCS05
Enter the Customer's nameGautam
~~~~~~~~~~~~~~~~~~~~~~~~~~~~MAGIC KINGDOM PARK~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Name: Gautam
Entryfees of the game Ferris Wheel : ₹ 1500
Entryfees of the game Free fall : ₹ 1500
Actual price: ₹ 4500
GST amount: ₹ 810.0
Total amount : ₹ 5310.0

Thank you for coming!
Have a great day!

☆｡:*•.────── ❁ ❁ ──────.•*:｡☆
```

ADVANTAGES

- Eco-friendly paperwork can be avoided.
- Efficient control over customers’ data.
- Cost-efficient and user-friendly.
- Bill is prepared for any customer when his/her ID is given as input.
- Easy access to bill amounts.
- Easy to access the games available and their details
- Easy to access customers’ bio-data/information.

DISADVANTAGES

- Absence of proper internet network makes it difficult for a user to access information.
- Alignment error in output
- Inability to alter the structures of the tables ‘Customers’ ,‘land_games’ and ‘water_games’.

FUTURE ENHANCEMENT OF THE PROJECT

This project helps the users to have an effective record of all the data regarding this amusement park with the help of three tables alone. It provides an efficient billing system for the customers. It can help the user to do many tasks very easily without the need of manual labour and tiring human calculations. In the future, functions to change the structure of the tables can be added. Extra offers and discounts for specific age groups and specific games can be added so that more customers will be coming to this amusement park and the effective profit will increase.

BIBLIOGRAPHY

- **Class 12 NCERT Textbook**

- **Python class 11 and 12 Sumita Arora**

- **Websites:**
  1. **www.python.org**
  2. **www.mysql.org**