ONLINE RESTAURANT MANAGEMENT SYSTEM

DBMS PROJECT

TEAM MEMBERS:
121910313002 SIRI CHANDANA
121910313005 SHIFA MEHREEN
121910313044 JASWANTH SAI
121910313049 ADITHYA
121910313055 SATYANARAYANA

INTRODUCTION

Definition :

Online food Management is a website to order food online. The system provides access to the customer to order online , reserve tables, and cancel tables from the website.

Description :

The Online Ordering System can be defined as a simple and convenient way for customers to purchase food online, without having to go to the restaurant. This system is enabled by the internet – it is the internet that connects the restaurant or the food company on one hand, and the customer on the other hand.

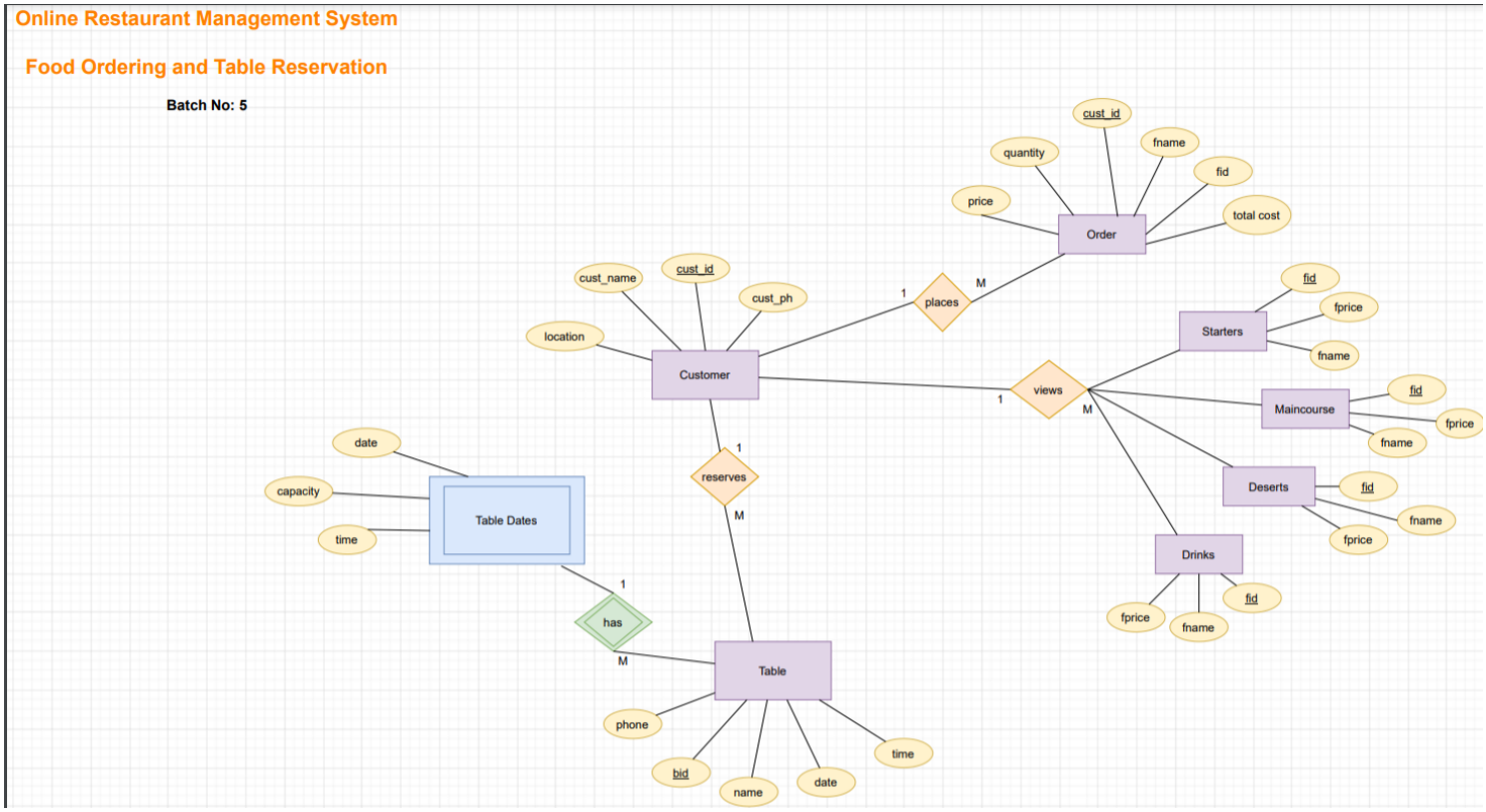
Abstract :

An Online Food Ordering System is proposed here which simplifies the food ordering process. The proposed system shows a user interface and updates the menu with all available options so that it eases the customer work. Customers can choose more than one item to make an order and can view order details. The order confirmation is sent to the customer. The order is placed in the queue and updated in the database and returned in real time. This system assists the staff to go through the orders in real time and process it efficiently with minimal errors.

Our Application Use:

It mainly allows the users to be able to view our food menu, reserve tables, basing on their convenient time and dates, cancel their reservation by giving their generated booking id, and order for food, by selecting any items they want from the following categories: starters, main course, desserts and drinks, at the end of the order, they will be able to view their bill.

ER diagram:



Link to ER diagram:
https://drive.google.com/file/d/1mCqxKHAOFgiihhEJolsc4RwFQxNk55Ay/view?usp=sharing

Software requirements:

In order to run our file we require:

- Python
- Mysql-connector-python
- Mysql server

Online food Database Conceptual schema:

CUSTOMER:

1. Cust_id: int - primary key
2. cust_name: varchar(30)
3. cust_ph: varchar(10)
4. location: varchar(30)

ORDERS:

1. cust_id: varchar(10)
2. fid: varchar(10)
3. fname: varchar(30)
4. price: int
5. quantity: int
6. cost: int

TABLE RESERVATION:

1. date: varchar(10)
2. time: varchar(10)
3. name: varchar(30)
4. phone: varchar(10)
5. b_id: int - PRIMARY KEY

TABLE DATES:

1. date: varchar(10)
2. time: varchar(10)
3. capacity: int

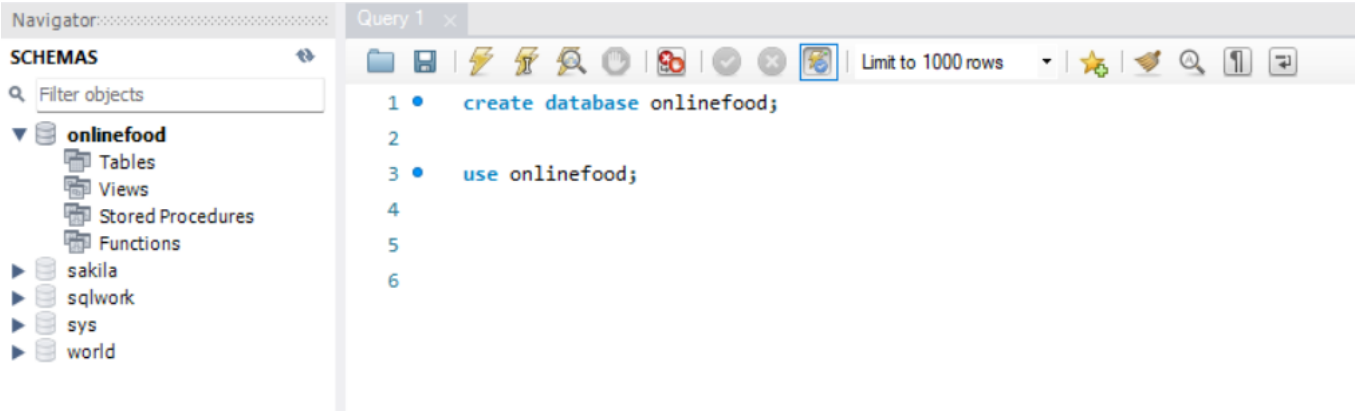
STARTERS, MAINCOURSE, DESSERTS, DRINKS:

1. Fid: int - primary key
2. Fname - varchar(30)
3. Fprice - int

---------- START OF PROJECT----------

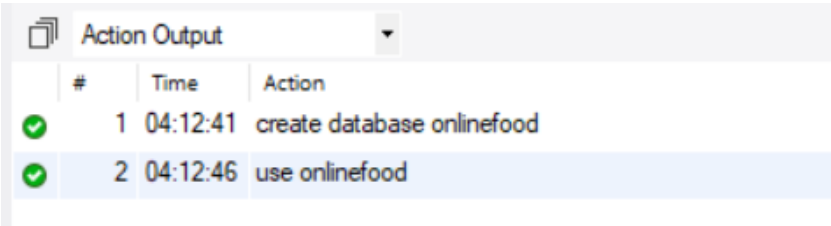
We used MySQL Workbench to run the backend commands as an admin. Our customer or the online user, will interact with our Python-written program.

MySQL:



We first created a database and named it “onlinefood”.
Using the command“ create database database_name”.

We then make our “onlinefood” database, as the default/current database for our further statements.
Using the “use” keyword we use our “onlinefood” database.



FOOD MENU:

We now create tables for our food menu. Since we can't create tables with the same name every time a user interacts with the program, we do the creation of tables and insertion of items, in the backend (as an admin), using MySQL Workbench.

Our restaurant menu has the following categories:

1. Starters
2. Main Course
3. Desserts
4. Drinks

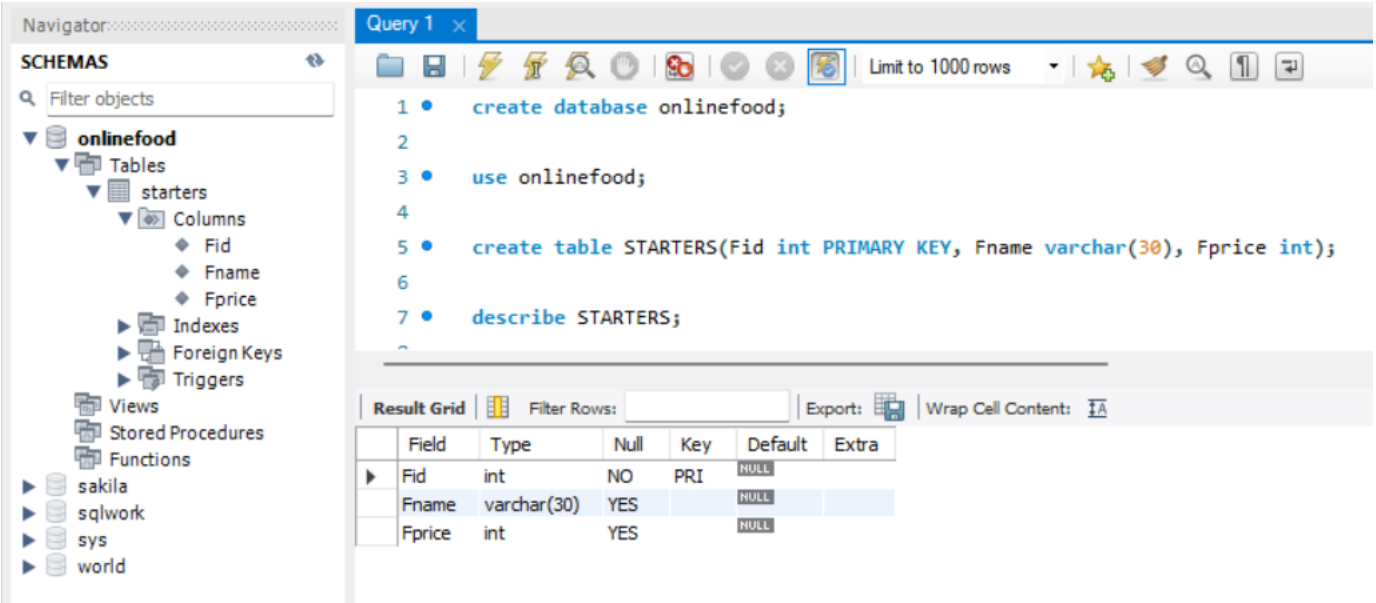
Each of these categories has its own table, containing the following attributes and datatypes:

1. Fid : int (primary key)
2. Fname: varchar(30)
3. Fprice: int

Fid → Food Id. It is a primary key, hence a unique and not null attribute.
Our user has to enter the particular food id for ordering that particular food item.
Fname → Food Name. It is to display the particular food item name.
Fprice → Food Price. It is to display the particular food item's price.

We then insert values for each column, including id, name, and price, and keep adding rows, for different food items, containing different values.

STARTERS Table:



Then we inserted some 10 items(rows) into the starters table.
As follows:

```
insert into STARTERS values
(1,'Manchuria', 120),
(2,'Crispy Corn',110),
(3,'Chicken 65', 150),
(4,'Panner Tikka',80),
(5,'Fish Chips', 90),
(6,'Aloo Tikka', 80),
(7,'Veg Spring Rolls', 120),
(8, 'Majestic Chicken', 130),
(9, 'Pepper Chicken', 110),
(10,'Kebab', 150);
```

We used the select command to show the starters table.

```
21 •  select * from starters;
```

Result Grid | Filter Rows:

Fid	Fname	Fprice
1	Manchuria	120
2	Crispy Corn	110
3	Chicken 65	150
4	Panner Tikka	80
5	Fish Chips	90
6	Aloo Tikka	80
7	Veg Spring Rolls	120
8	Majestic Chicken	130
9	Pepper Chicken	110
10	Kebab	150

MAINCOURSE Table:

Navigator

SCHEMAS

Filter objects

- onlinefood
  - Tables
    - maincourse
      - Columns
      - Indexes
      - Foreign Keys
      - Triggers
    - starters
  - Views
  - Stored Procedures
  - Functions
- sakila
- sqlwork
- sys
- world

Administration | Schemas

Information

Table: maincourse

Columns:

Fid	int PK
Fname	varchar(30)
Fprice	int

Query 1

Limit to 1000 rows

```
14    (5,'Fish Chips', 90),
15    (6,'Aloo Tikka', 80),
16    (7,'Veg Spring Rolls', 120),
17    (8, 'Majestic Chicken', 130),
18    (9, 'Pepper Chicken', 110),
19    (10,'Kebab', 150);
20
21 •  select * from starters;
22
23
24 •  create table MAINCOURSE(Fid int PRIMARY KEY, Fname varchar(30), Fprice int);
25
26 •  describe MAINCOURSE;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Fid	int	NO	PRI	NULL	
Fname	varchar(30)	YES		NULL	
Fprice	int	YES		NULL	

Then we inserted some 10 items(rows) into the main course table.
As follows:

```
insert into MAINCOURSE values
(1,'Butter Chicken', 170),
(2,'Veg Biryani',120),
(3,'Mughlai Biryani', 160),
(4,'Veg Fried Rice',80),
(5,'Chicken Fried Rice', 90),
(6,'Chana Dal', 80),
(7,'Cashew Curry', 110),
(8, 'Palak Paneer', 130),
(9, 'Chicken Mandi', 150),
(10,'Lamb Curry', 180);
```

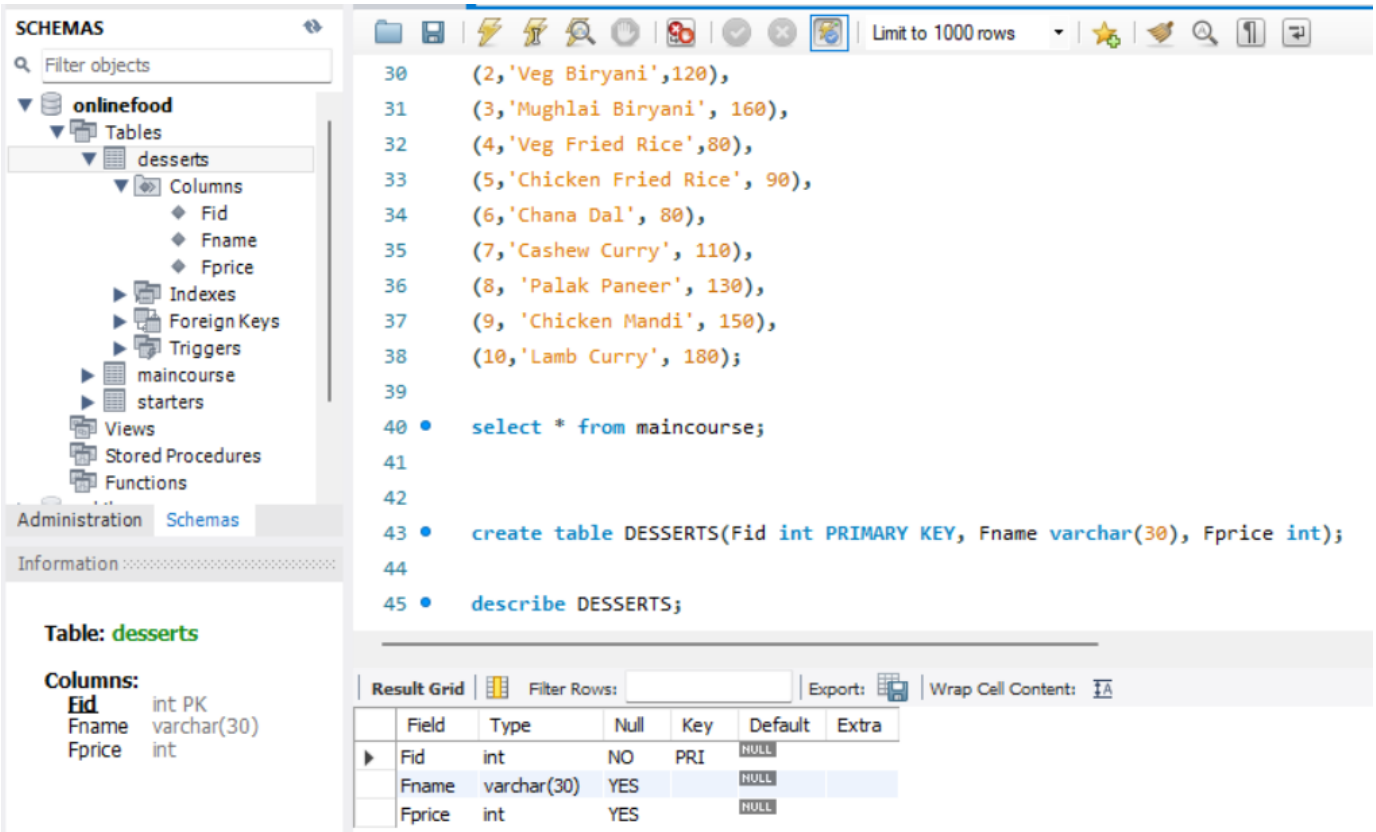
Showing the maincourse table:

```
select * from maincourse;
```

Result Grid | Filter Rows:

Fid	Fname	Fprice
1	Butter Chicken	170
2	Veg Biryani	120
3	Mughlai Biryani	160
4	Veg Fried Rice	80
5	Chicken Fried Rice	90
6	Chana Dal	80
7	Cashew Curry	110
8	Palak Paneer	130
9	Chicken Mandi	150
10	Lamb Curry	180
NULL	NULL	NULL

DESSERTS Table:



Then we inserted some 5 items(rows) into the desserts table.
As follows:

```
insert into DESSERTS values
(1,'Apricort Delight', 110),
(2,'Gajar ka halwa',80),
(3,'Gulab Jamun', 90),
(4,'Chocolate Fudge',150),
(5,'Choco Lava Cake', 130);
```

Showing the desserts table:

```
54 •  select * from desserts;
```

Result Grid | Filter Rows:

Fid	Fname	Fprice
1	Apricort Delight	110
2	Gajar ka halwa	80
3	Gulab Jamun	90
4	Chocolate Fudge	150
5	Choco Lava Cake	130

DRINKS Table:

Navigator

SCHEMAS

Filter objects

- onlinefood
  - Tables
    - desserts
    - drinks
      - Columns
        - Fid
        - Fname
        - Fprice
      - Indexes
      - Foreign Keys
      - Triggers
    - maincourse
    - starters
  - Views
  - Stored Procedures

Administration | Schemas

Information

Table: drinks

Columns:

Fid	int PK
Fname	varchar(30)
Fprice	int

Query 1

Limit to 1000 rows

```
45 •  describe DESSERTS;
46
47 •  insert into DESSERTS values
48    (1,'Apricort Delight', 110),
49    (2,'Gajar ka halwa',80),
50    (3,'Gulab Jamun', 90),
51    (4,'Chocolate Fudge',150),
52    (5,'Choco Lava Cake', 130);
53
54 •  select * from desserts;
55
56 •  create table DRINKS(Fid int PRIMARY KEY, Fname varchar(30), Fprice int);
57
58 •  describe DRINKS;
59
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Fid	int	NO	PRI	NULL	
Fname	varchar(30)	YES		NULL	
Fprice	int	YES		NULL	

Then we inserted some 7 items(rows) into the drinks table.
As follows:

```
insert into DRINKS values
(1,'Lemonade', 50),
(2,'Coke', 60),
(3,'Chocolate Milkshake', 130),
(4,'Strawberry Juice',110),
(5,'Iced Berry Tea', 120),
(6, 'Iced Matcha Latte', 140),
(7, 'Mineral Water', 45);
```

Showing the drinks table:

```
select * from drinks;
```

Result Grid Filter Rows:

Fid	Fname	Fprice
1	Lemonade	50
2	Coke	60
3	Chocolate Milkshake	130
4	Strawberry Juice	110
5	Iced Berry Tea	120
6	Iced Matcha Latte	140
7	Mineral Water	45
NULL	NULL	NULL

PYTHON:

We install MySQL Python Connector, to access the MySQL database.

Then we import the connector in our code, using the statement:

```
import mysql.connector          #importing connector package
```

We also import random to generate random id numbers for our customers (cust-ids) and our booking ids(b_id) for table reservations, using:

```
import random       #importing random package
```

To establish a connection we use the connect() function from the mysql.connector package and give arguments like, host, user, and password

host → local host (default)
user → root (default user)
password → Password123# (it's my local MySQL database password)

Statement used:

```
db =mysql.connector.connect(host="localhost",user="root",password="Password123#")
 #to establish connection
```

db → it is a variable used here as an object to help with the connection

The cursor() method creates a cursor object that is used to execute a SQL query by using the execute() method.

We have created an object, with mycursor as a variable that acts as our cursor in the database.
mycursor=db.cursor()

To execute the statements, we use the execute() method.
Here we use,
mycursor.execute()

```
import mysql.connector
import random


db =mysql.connector.connect(host="localhost",user="root",password="Password123#", database = "OnlineFood")

mycursor=db.cursor()
```

PROGRAM:

about()
main()

```
about()
main()
```

about() method:

We first call our about() method.

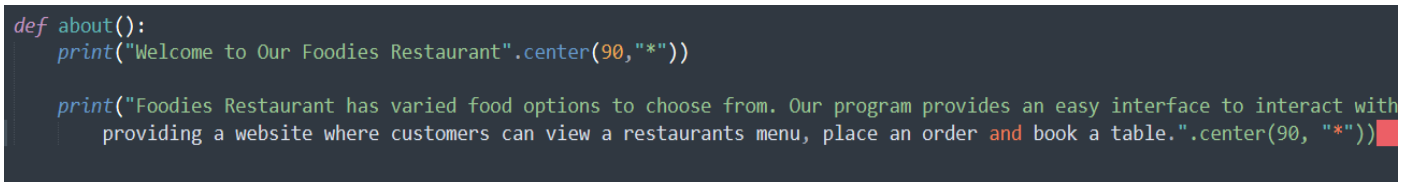
In about(), we welcome the users, to use our program and talk a little about the program applications.

We named our restaurant, “Foodies Restaurant”.

CODE:

```
def about():
    print("Welcome to Our Foodies Restaurant".center(90,"*"))

    print("Foodies Restaurant has varied food options to choose from. Our program provides an easy interface to interact with providing a website where customers can view a restaurant's menu, place an order and book a table.".center(90, "*"))
```



Output:

```
============== RESTART: C:\Users\shifa\OneDrive\Documents\food.py ==============
****************************Welcome to Our Foodies Restaurant****************************
Foodies Restaurant has varied food options to choose from. Our program provides an easy interface
to interact with providing a website where customers can view a restaurants menu, place an order
and book a table.
```

(pre-defined function in python:

print() → is to print the given string in the quotations.
center() → is to format the code and align it to the center)

main() method:

After about() we call our main() method, where we give the user a few options to select from. If the user enters the option number, we call the function corresponding to the number, and then, the code moves to that part where a particular function is written, in the program.

Options are as follows:

1. Order food
2. Reserve Table
3. Cancel Table
4. View Menu
5. Exit

```
def main():
    print("Our Online System provides the following options:  ".center(90,"*"))


    print("""
        (1) ORDER FOOD
        (2) RESERVE TABLE
        (3) CANCEL TABLE
        (4) VIEW MENU
        (5) EXIT
            """)

    print("Select one from the above options".center(90,"."))

    option = input("Enter Your Choice No.: ")

    if option == "1": orderFood()
    elif option == "2": reserveTable()
    elif option == "3": cancelTable()
    elif option == '4': viewMenu()
    elif option == "5": exit()
    else:
        print("Try Again! Please, chose from the given options ")
        main()
```

```python
def main():
    print("Our Online System provides the following options:  ".center(90,"*"))


    print("""
            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT
                    """)

    print("Select one from the above options".center(90,"."))

    option = input("Enter Your Choice No.: ")

    if option == "1": orderFood()
    elif option == "2": reserveTable()
    elif option == "3": cancelTable()
    elif option == '4': viewMenu()
    elif option == "5": exit()
    else:
        print("Try Again! Please, chose from the given options ")
        main()
```

Output:

```
********************Our Online System provides the following options:  ********************

            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT

...........................Select one from the above options...........................
Enter Your Choice No.:
```

exit() method:

If the user enters option 5, we call the “exit()” method, and the program prints a thank you statement and then terminates the program, using the quit() method.

```python
def exit():
    print("Thank You! Do Visit Us again!".center(90,"."))
    print("Closing The Application!".center(90,"*"))
    quit()
```

(pre-defined function in python:

quit() → When it encounters the quit() function in the system, it terminates the execution of the program completely.)

```
def exit():
    print("Thank You! Do Visit Us again!".center(90,"."))
    print("Closing The Application!".center(90,"*"))
    quit()
```

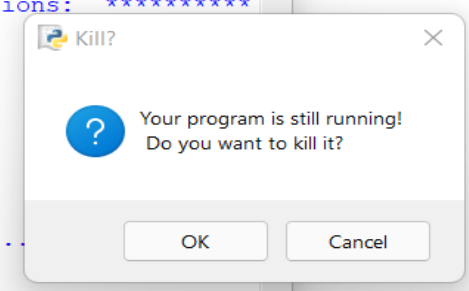
Output:

```
============== RESTART: C:\Users\shifa\OneDrive\Documents\food.py ==============
****************************Welcome to Our Foodies Restaurant*******************
**********
Foodies Restaurant has varied food options to choose from. Our program provides
an easy interface to interact with providing a website where customers can view
a restaurants menu, place an order and book a table.


********************Our Online System provides the following options:  **********
**********

            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT

..............................Select one from the above options....
..........
Enter Your Choice No.: 5
...............................Thank You! Do Visit Us again!......................
..........
*********************************Closing The Application!************************
**********
```



viewMenu() method:

When the user enters option 4, we call the “viewMenu” method. In this method, we display our whole Food Menu to our user, in an order. We show the user the category and the items in that particular category.

```
def viewMenu():
    print("Our Online System provides the following categories and food items in our FoodMenu: ".center(90,"*"))
    print("\n")
    lis=['starters','maincourse','desserts','drinks']
    for i in lis:
        print("\n")
        print(i.upper().center(90,"."))
        print("|NO|".center(30," ")+"|FOOD NAME|".center(30," ")+"|PRICE|".center(30," "))
        mycursor.execute("select * from "+i)
        for x in mycursor:
            no = x[0]
            name = x[1]
            price = x[2]
            print(str(no).center(30," ") + str(name).center(30," ")  + str(price).center(30," "))
    print("\n")
    main()
```

In the end, we call the main() function, in order to show the user the other available options he can choose from.

(print("\n") → takes us to a new line)

```
def viewMenu():
    print("Our Online System provides the following categories and food items in our FoodMenu: ".center(90,"*"))
    print("\n")
    lis=['starters','maincourse','desserts','drinks']
    for i in lis:
        print("\n")
        print(i.upper().center(90,"."))
        print("|NO|".center(30," ")+"|FOOD NAME|".center(30," ")+"|PRICE|".center(30," "))
        mycursor.execute("select * from "+i)
        for x in mycursor:
            no = x[0]
            name = x[1]
            price = x[2]
            print(str(no).center(30," ") + str(name).center(30," ")  + str(price).center(30," "))
    print("\n")
    main()
```

lis → is our list of names of our categories tables
i → variable to access one category at a time, from lis

For each category,
We use the query: "select * from "+i , to display all the elements or food items in that particular category table.
We use the execute() method to execute our query and mycursor() object along with it.

We print all the rows using a for loop and since our table has 3 columns, we take 3 variables, no, name and price, and print them together, in one line.

Output:

~~~~~~~~~~~~~~~~~~~~~~~~~~~~Welcome to Our Foodies Restaurant~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Foodies Restaurant has varied food options to choose from. Our program provides an easy interface to interact with providing a website where customers can view a restaurants menu, place an order and book a table.

**********Our Online System provides the following options: **********

(1) ORDER FOOD
(2) RESERVE TABLE
(3) CANCEL TABLE
(4) VIEW MENU
(5) EXIT

==========Select one from the above options==========

Enter Your Choice No.: 4

***Our Online System provides the following categories and food items in our FoodMenu: ***

.....STARTERS.....

NO	FOOD NAME	PRICE
1	Manchuria	120
2	Crispy Corn	110
3	Chicken 65	150
4	Panner Tikka	80
5	Fish Chips	90
6	Aloo Tikka	80
7	Veg Spring Rolls	120
8	Majestic Chicken	130
9	Pepper Chicken	110
10	Kebab	150

.....MAINCOURSE.....

NO	FOOD NAME	PRICE
1	Butter Chicken	170
2	Veg Biryani	120
3	Mughlai Biryani	160
4	Veg Fried Rice	80
5	Chicken Fried Rice	90
6	Chana Dal	80
7	Cashew Curry	110
8	Palak Paneer	130
9	Chicken Mandi	150
10	Lamb Curry	180

.....DESSERTS.....

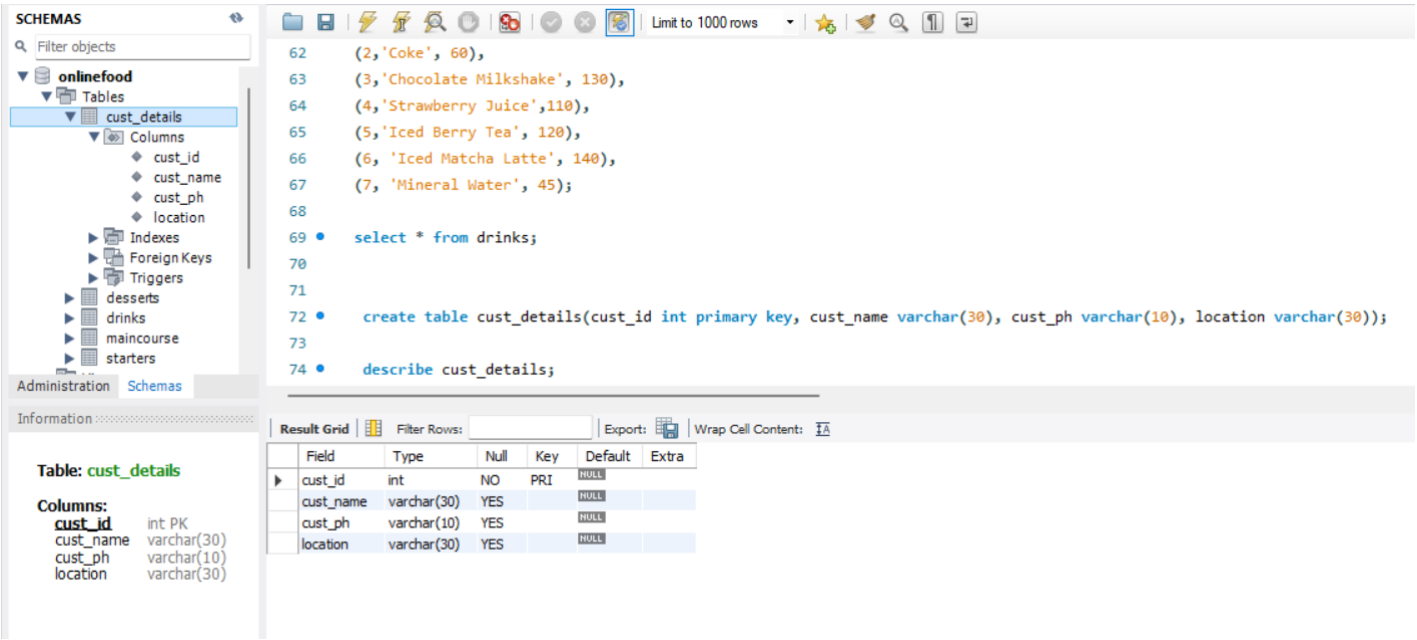
NO	FOOD NAME	PRICE
1	Apricort Delight	110
2	Gajar ka halwa	80
3	Gulab Jamun	90
4	Chocolate Fudge	150
5	Choco Lava Cake	130

.....DRINKS.....

NO	FOOD NAME	PRICE
1	Lemonade	50
2	Coke	60
3	Chocolate Milkshake	130
4	Strawberry Juice	110
5	Iced Berry Tea	120
6	Iced Matcha Latte	140
7	Mineral Water	45
~~~~~~~~~~~~~~~~~~~~~~~~~~~~

CUSTOMER DETAILS Table:

cust_details table:



We dynamically insert values of customer details, in the table, when the customer wants to order food, from our program.

orderFood() method:

When the customer enters option 1, we call the “orderFood” method.

- In this method, we ask the customer to enter his details, which include, name, phone number, and location.
- We then assign that customer, with a random integer as cust_id
- And insert all the values into our cut_details table, dynamically.

We call another method chooseMenu(), in our orderFood() method and pass the cust_id as an argument.

```
def orderFood():

    print("Let's Get You Some Food!".center(90,"."))
    name=input("Enter your name : ")
    phno = input("Enter your phone number: ")
    location  =  input("Enter the location : ")
    s='select cust_id from cust_details'
    mycursor.execute(s)
    data = mycursor.fetchall()
    id = random.randint(1,100000)
    sql='insert into cust_details values(%s,%s,%s,%s)'
    x=(id,name,phno,location)
    mycursor.execute(sql,x)
    db.commit()
    chooseMenu(id)
    print("\n")
    main()
```


```
def orderFood():


    print("Let's Get You Some Food!".center(90,"."))
    name=input("Enter your name : ")
    phno = input("Enter your phone number: ")
    location  =  input("Enter the location : ")
    s='select cust_id from cust_details'
    mycursor.execute(s)
    data = mycursor.fetchall()
    id = random.randint(1,100000)
    sql='insert into cust_details values(%s,%s,%s,%s)'
    x=(id,name,phno,location)
    mycursor.execute(sql,x)
    db.commit()
    chooseMenu(id)
    print("\n")
    main()
```

db.commit() → this method commits the changes to the database.
fetchall() → fetches all the rows of a query result.
randint() → returns an integer number selected element from the specified range.

Output:

```
*******************Our Online System provides the foll

            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT

===========================Select one from the above
Enter Your Choice No.: 1


.................................Let's Get You Some Fo
Enter your name : Meena
Enter your phone number: 8901675423
Enter the location : Vizag
...........................Choose What you Wish To Ea
*******************Our Online System provides the fol
             |NO|                        |FOOD NAME|
              1                           Manchuria
              2                          Crispy Corn
```

```
67    (7, 'Mineral Water', 45);
68
69 •  select * from drinks;
70
71
72 •   create table cust_details(cust_
73
74 •   describe cust_details;
75
76 •   select * from cust_details;
```

Result Grid | Filter Rows:

cust_id	cust_name	cust_ph	location
11903	Meena	8901675423	Vizag
NULL	NULL	NULL	NULL

Table: cust_details

Columns:

cust_id	int PK
cust_name	varchar(30)
cust_ph	varchar(10)

ORDERS Table:

```
67    (7, 'Mineral Water', 45);
68
69 •  select * from drinks;
70
71
72 •   create table cust_details(cust_id int primary key, cust_name varchar(30), cust_ph varchar(10), location varchar(30));
73
74 •   describe cust_details;
75
76 •   select * from cust_details;
77
78
79 •  create table orders(cust_id varchar(10), fid varchar(10), fname varchar(30), price int, quantity int, cost int);
80
81 •  describe orders;
82
83 •  select * from orders;
84
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
cust_id	varchar(10)	YES		NULL	
fid	varchar(10)	YES		NULL	
fname	varchar(30)	YES		NULL	
price	int	YES		NULL	
quantity	int	YES		NULL	
cost	int	YES		NULL	

Table: orders

Columns:

cust_id	varchar(10)
fid	varchar(10)
fname	varchar(30)
price	int
quantity	int
cost	int

We dynamically insert values of cust_id, in the table, when the customer wants to order food, from our program.

chooseMenu() method:

We call another method chooseMenu(), in our orderFood() method and pass the cust_id as argument.

Where we show the user each category's, food item
and ask if he wants anything from that category:
If yes: we ask him to enter the food item number and the quantity and in the end print a bill

Else and if later the user enters 0: we move to the next category

After the end of selection, total amount to be paid and also the order details are mentioned.

```
def chooseMenu(cid):
    print("Choose What you Wish To Eat/Drinks".center(90,"."))

    print("Our Online System provides the following options: ".center(90,"*"))

    lis=['starters','maincourse','desserts','drinks']
    total=0
    for i in lis:
        print("|NO|".center(30," ")+"|FOOD NAME|".center(30," ")+"|PRICE|".center(30," "))
        mycursor.execute("select * from "+i)
        for x in mycursor:
            no = x[0]
            name = x[1]
            price = x[2]
```

```
    print(str(no).center(30," ") + str(name).center(30," ")  + str(price).center(30," "))
    print("\n")
    k=input("Do you want "+i+"? press yes or no : ")

    while True:
        if k=='yes':
            while True:
                fid=input("Enter FID : ")
                q=int(input("Enter quantity : "))
                z='select fname,fprice from '+i+' where fid = %s'
                w=(fid,)
                mycursor.execute(z,w)
                l=mycursor.fetchone()
                fname,fprice=l[0],l[1]
                cost=q*fprice
                s=(cid,fid,fname,fprice,q,cost)
                x='insert into orders values(%s,%s,%s,%s,%s,%s)'
                mycursor.execute(x,s)
                db.commit()
                a=int(input("Enter 0 to quit "))
                if a==0:
                    k='no'
                    break
        elif k=='no':
            break
        else:
            k=input("enter a valid string yes/no : ")

print("\n")
```

```
#order payment amount
    total_bill=0
    print("Your bill is : ")
    sql='select fname,price,quantity,cost from orders where cust_id=%s'
    d=(cid,)
    mycursor.execute(sql,d)
    l=mycursor.fetchall()
    j=0
    print("s.no".center(15," ")+"food name".center(15," ")+"price".center(15,"
")+"quantity".center(15," ")+"cost".center(15," "))
    for i in l:
        print(str(j).center(15," ")+str(i[0]).center(15," ")+str(i[1]).center(15," ")+str(i[2]).center(15,"
")+str(i[3]).center(15," "))
        j+=1
        total_bill+=i[3]
    s="Total amount to be paid is : "+str(total_bill)
    print(s.center(90," "))
    print("\n")
```

```python
def chooseMenu(cid):
    print("Choose What you Wish To Eat/Drinks".center(90,"."))

    print("Our Online System provides the following options: ".center(90,"*"))

    lis=['starters','maincourse','desserts','drinks']
    total=0
    for i in lis:
        print("|NO|".center(30," ")+"|FOOD NAME|".center(30," ")+"|PRICE|".center(30," "))
        mycursor.execute("select * from "+i)
        for x in mycursor:
            no = x[0]
            name = x[1]
            price = x[2]
            print(str(no).center(30," ") + str(name).center(30," ")  + str(price).center(30," "))
        print("\n")
        k=input("Do you want "+i+"? press yes or no : ")
        while True:
            if k=='yes':
                while True:
                    fid=input("Enter FID : ")
                    q=int(input("Enter quantity : "))
                    z='select fname,fprice from '+i+' where fid = %s'
                    w=(fid,)
                    mycursor.execute(z,w)
                    l=mycursor.fetchone()
                    fname,fprice=l[0],l[1]
                    cost=q*fprice
                    s=(cid,fid,fname,fprice,q,cost)
                    x='insert into orders values(%s,%s,%s,%s,%s,%s)'
                    mycursor.execute(x,s)
                    db.commit()
                    a=int(input("Enter 0 to quit "))
                    if a==0:
                        k='no'
                        break
            elif k=='no':
                break
            else:
                k=input("enter a valid string yes/no : ")

    print("\n")
```

```python
total_bill=0
print("Your bill is : ")
sql='select fname,price,quantity,cost from orders where cust_id=%s'
d=(cid,)
mycursor.execute(sql,d)
l=mycursor.fetchall()
j=0
print("s.no".center(15," ")+"food name".center(15," ")+"price".center(15," ")+"quantity".center(15," ")+"cost".center(15," "))
for i in l:
    print(str(j).center(15," ")+str(i[0]).center(15," ")+str(i[1]).center(15," ")+str(i[2]).center(15," ")+str(i[3]).center(15," "))
    j+=1
    total_bill+=i[3]
s="Total amount to be paid is : "+str(total_bill)
print(s.center(90," "))
print("\n")
```

OUTPUT:

```
============================Select one from the above options============================
Enter Your Choice No.: 1


.................................Let's Get You Some Food!.................................
Enter your name : Meena
Enter your phone number: 9876657191
Enter the location : Vizag
...........................Choose What you Wish To Eat/Drinks...........................
********************Our Online System provides the following options: ********************
            |NO|                      |FOOD NAME|                      |PRICE|
             1                         Manchuria                         120
             2                        Crispy Corn                        110
             3                         Chicken 65                        150
             4                        Panner Tikka                        80
             5                         Fish Chips                         90
             6                         Aloo Tikka                         80
             7                      Veg Spring Rolls                     120
             8                      Majestic Chicken                     130
             9                       Pepper Chicken                      110
             10                          Kebab                           150


Do you want starters? press yes or no : yes
Enter FID : 1
Enter quantity : 1
Enter 0 to quit /
Enter FID : 2
Enter quantity : 2
Enter 0 to quit 0
```

```
Enter 0 to quit 0
            |NO|                    |FOOD NAME|                    |PRICE|
             1                     Butter Chicken                    170
             2                      Veg Biryani                      120
             3                    Mughlai Biryani                    160
             4                     Veg Fried Rice                     80
             5                   Chicken Fried Rice                   90
             6                       Chana Dal                        80
             7                      Cashew Curry                     110
             8                      Palak Paneer                     130
             9                     Chicken Mandi                     150
             10                      Lamb Curry                      180


Do you want maincourse? press yes or no : yes
Enter FID : 3
Enter quantity : 1
Enter 0 to quit 0
            |NO|                    |FOOD NAME|                    |PRICE|
             1                    Apricort Delight                   110
             2                     Gajar ka halwa                     80
             3                      Gulab Jamun                       90
             4                    Chocolate Fudge                    150
             5                    Choco Lava Cake                    130


Do you want desserts? press yes or no : 3
enter a valid string yes/no : yes
Enter FID : 3
Enter quantity : 1
Enter 0 to quit 0
            |NO|                    |FOOD NAME|                    |PRICE|
             1                        Lemonade                        50
             2                          Coke                          60
             3                  Chocolate Milkshake                   130
             4                    Strawberry Juice                    110
             5                     Iced Berry Tea                     120
             6                   Iced Matcha Latte                    140
             7                     Mineral Water                       45


Do you want drinks? press yes or no : yes
Enter FID : 2
Enter quantity : 1
Enter 0 to quit 0
```

```
Your bill is :
      s.no        food name        price         quantity         cost
       0          Manchuria         120             1             120
       1         Crispy Corn        110             2             220
       2       Mughlai Biryani      160             1             160
       3         Gulab Jamun         90             1              90
       4             Coke            60             1              60
                 Total amount to be paid is : 650
```

Changes in Database's Cust_details and Order Table:

```
76 •    select * from cust_details;
77
```

Result Grid | Filter Rows: | Edit:

cust_id	cust_name	cust_ph	location
45194	Meena	9876657191	Vizag
NULL	NULL	NULL	NULL

```
81 •    select * from orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

cust_id	fid	fname	price	quantity	cost
45194	1	Manchuria	120	1	120
45194	2	Crispy Corn	110	2	220
45194	3	Mughlai Biryani	160	1	160
45194	3	Gulab Jamun	90	1	90
45194	2	Coke	60	1	60

TABLERESERVATION Table:

SCHEMAS

Filter objects

- desserts
- drinks
- maincourse
- orders
- starters
- tabledates
- tablereservation
  - Columns
  - Indexes
  - Foreign Keys
  - Triggers
- Views
- Stored Procedures
- Functions

Administration | Schemas

Information

Table: tabledates

Columns:
date varchar(10)
time varchar(10)
capacity int

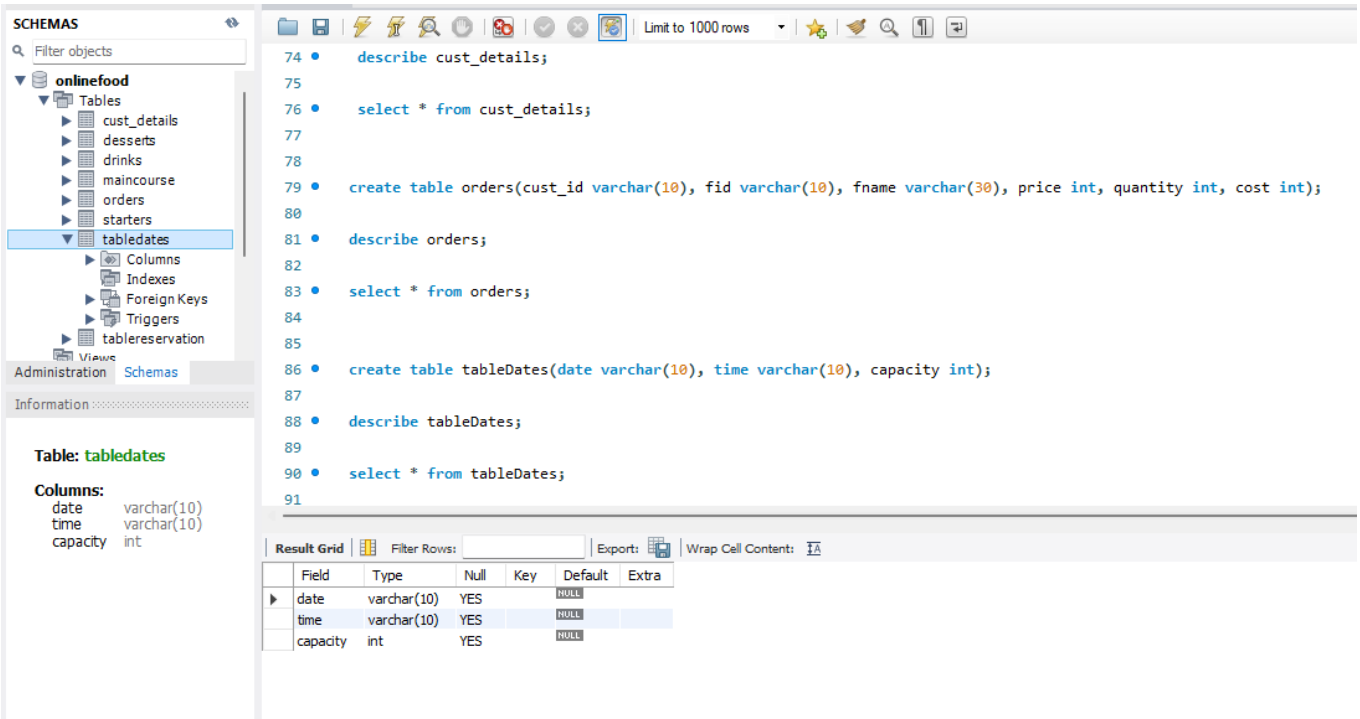
Limit to 1000 rows

```
81 •  describe orders;
82
83 •  select * from orders;
84
85
86 •  create table tableDates(date varchar(10), time varchar(10), capacity int);
87
88 •  describe tableDates;
89
90 •  select * from tableDates;
91
92 •  create table tableReservation(date varchar(10), time varchar(10), name varchar(30), phone varchar(10), b_id int);
93
94 •  describe tableDates;
95
96 •  select * from tableReservation;
97
98
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
date	varchar(10)	YES		NULL	
time	varchar(10)	YES		NULL	
capacity	int	YES		NULL	

TABLEDATES Table:



reserveTable() method:

When the user inputs option no - 2. We executed the reserve table function.

Here we first display that the restaurant is open on all days from 12pm to 11pm.
Then we show the user that he can only choose from the given time slots.

Everytime, a user reserves a table:

- We consider that we have 10 tables available for a given time slot and whenever one person books at that particular time slot and date, the capacity (i.e, the number of tables decreases by 1).
- If no one has booked any table yet at a particular date and time slot, we call the bookTable function and insert the initial capacity value in TableDate as 9.
- If someone else books a table at that same time slot and date we update the capacity value in the TableDate, i.e, the number of tables decreases by 1.
- Hence, if the tables are available we call the bookTable() function, and take user details and confirm the table booking.
- If all the 10 tables are occupied, we tell the user that “no tables are available”.

```
def reserveTable():

    print("Restaurant is open on all days from 12:00 PM to 11:00 PM")

    print("Tables can be booked in given time slots, 12:00 PM, 01:00 PM ,02:00 PM, 03:00PM,
05:00PM, 06:00PM,07:00PM,08:00PM,09:00PM,10:00PM,11:00PM")

    date = input("Enter date of reservation in DD-MM-YYYY format : ")
    time = int(input("Select time slot between 12 PM to 11PM, Enter in hours only:"))

    while time<1 and time>12:
        time=int(input("Enter valid time : "))
    s='select capacity,time from tableDates where date=%s'
    d=(date,)
    mycursor.execute(s,d)
    l=mycursor.fetchall()

    if l==[]:
        s='insert into tableDates values(%s,%s,%s)'
        d=(date,time,9)
        mycursor.execute(s,d)
        db.commit()
        bookTable(date,time)
    else:
        if int(l[0][1])==time:
            if l[0][0]>=1:
                s='update tableDates set capacity=%s where date=%s and time=%s'
                d=(l[0][0]-1,date,time)
                mycursor.execute(s,d)
                db.commit()
                bookTable(date,time)
            else:
                print("Tables not available at the given time ")
        else:
            s='insert into tableDates values(%s,%s,%s)'
```

```
            d=(date,time,9)
            mycursor.execute(s,d)
            db.commit()
            bookTable(date,time)
        print("\n")
        main()
```

bookTable() method:

In this function, we ask for customer's details, like name and phone number and insert them into a TableReservation Table in our database and generate a random booking id.

```
def bookTable(d,t):

    print("Enter your details!".center(90,"."))

    name=input("Enter your name : ")
    phone=input("Enter your phone number : ")
    s='select b_id from tableReservation'
    mycursor.execute(s)
    l=mycursor.fetchall()
    id=random.randint(1,100000)
    while id in l:
        id=random.randint(1,100000)
    s='insert into tableReservation values(%s,%s,%s,%s,%s)'
    d=(d,t,name,phone,id)
    mycursor.execute(s,d)
    print("Booking successful")
    print("Your booking id is ",id)
    print("\n")
    db.commit()
```

OUTPUT:

```
*******************Our Online System provides the following options:  ********************

            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT

==========================Select one from the above options============================
Enter Your Choice No.: 2


Restaurant is open on all days from 12:00 PM to 11:00 PM
Tables can be booked in given time slots, 12:00 PM, 01:00 PM ,02:00 PM, 03:00PM, 05:00PM, 06:00PM,07:00PM,08:00PM,09:00PM,10:00PM,11:00PM
Enter date of reservation in DD-MM-YYYY format : 22-10-2021
Select time slot between 12 PM to 11PM, Enter in hours only:01
.................................Enter your details!.................................
Enter your name : Shifa
Enter your phone number : 6543278210
Booking successful
Your booking id is  97327
```

Database Outputs:

```
96 •   select * from tableReservation;
97
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

date	time	name	phone	b_id
22-10-2021	1	Shifa	6543278210	97327

```
90 •   select * from tableDates;
91
```

Result Grid | Filter Rows: | Export:

date	time	capacity
22-10-2021	1	9

cancelTable() method:

If the customer selects the option no-3. We call the function cancelTable().

- Here, we ask the user to input his booking id.
- If the booking id is valid, we send a message that cancellation was successful. And we delete that customer's details from the TableReservation Table and update the capacity of tables in the TableDate table (i.e, increment the capacity by 1).
- Else, if the booking id id invalid, we send a message that the booking id is not valid.

```
def cancelTable():

    i=int(input("Enter booking id : "))
    try:
        b='select date,time from tableReservation where b_id=%s'
        q=(i,)
        mycursor.execute(b,q)
        l=mycursor.fetchone()
        date=l[0]
        time=l[1]
        sql1='delete from tableReservation where b_id=%s'
        d1=(i,)
        mycursor.execute(sql1,d1)
        a='select capacity from tableDates where date=%s and time=%s'
        d2=(date,time)
        mycursor.execute(a,d2)
        l=mycursor.fetchone()
        sql2='update tableDates set capacity=%s where date=%s and time=%s'
        d2=(l[0]+1,date,time)
        mycursor.execute(sql2,d2)
        db.commit()
        print("Cancellation successful!")
        print("\n")
        main()
```

```
except:
    print("No booking found! ")
    print("\n")
    main()
```

OUTPUT:

Table Booked:

```
*******************Our Online System provides the following options:  *******************

            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT

===========================Select one from the above options============================
Enter Your Choice No.: 2


Restaurant is open on all days from 12:00 PM to 11:00 PM
Tables can be booked in given time slots, 12:00 PM, 01:00 PM ,02:00 PM, 03:00PM, 05:00PM, 06:00PM,07:00PM,08:00PM,09:00PM,10:00PM,11:00PM
Enter date of reservation in DD-MM-YYYY format : 22-10-2021
Select time slot between 12 PM to 11PM, Enter in hours only:02
....................................Enter your details!....................................
Enter your name : Siri
Enter your phone number : 8765290119
Booking successful
Your booking id is  39730
```

Database output:

```
95
96 •    select * from tableReservation;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

date	time	name	phone	b_id
22-10-2021	1	Shifa	6543278210	97327
22-10-2021	2	Siri	8765290119	39730

Cancelling the booked table:

```
*******************Our Online System provides the following options:  *******************

            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT

==========================Select one from the above options==========================
Enter Your Choice No.: 3


Enter booking id : 3970
No booking found!
```

Since, we entered the wrong booking id

Else:

```
*******************Our Online System provides the following options:  *******************

            (1) ORDER FOOD
            (2) RESERVE TABLE
            (3) CANCEL TABLE
            (4) VIEW MENU
            (5) EXIT

==========================Select one from the above options==========================
Enter Your Choice No.: 3


Enter booking id : 39730
Cancellation successful!
```

----------END OF PROJECT----------

Conclusion:

This was how we implemented the project. This was a rough draft of our project idea, where we could connect to the database and perform basic CRUD operations. We were able to help the customer order food, reserve and cancel tables and view our food menu through our program.

We plan on improving our project in the future.

THANK YOU!