Ahsanullah University of Science and Technology (AUST)
Department of Computer Science and Engineering

Project Proposal: Restaurant Management System

Course No.: CSE4126

Course Title: Distributed Database Systems Lab

Semester: Fall 2022

Date of Submission - [17.08.2023]

Submitted To-
Ms. Zarin Tasnim Shejuti
&
Ms. Sanzana Karim Lora

Submitted By-
Member 1:
190204004: Md Abu Hanjala
Member 2:
190204005: M. Saymon Islam Iftikar
Lab Group: A1
Year: 4th
Semester: 1st
Department: CSE

Restaurant Management System

Objective: The restaurant management system is a distributed database management system where people easily find their daily food items. Here, customers can see the food list, check the tables availability, reserved the table, give orders and the admin can update the food list, accept order and place the orders. Finally, customers can give their valuable feedback to the items.

Database Schema:

Global Schema:

- **Menu**: (item_id,name, description, price, category)
- **Customers**: (customer_id,name, contact , address, phone ,Gender).
- **Orders**: (customer ID, menu items ordered, and the total cost).
- **Feedback(**fid,item_id,msg,points**)**
- **Tables**(table_id,seats,status)
- **Reservation (**Rid,reserve_date,customer_id,table_id,guests,**)**
- **Placed_Order(**place_id,total_amount,customer_id,address,email**)**

Fragmentation Schema:

Menu1 = SL $_{\text{cat='Bangalee'}}$ (Menu1)

Menu2 = SL $_{\text{cat='Chinese'}}$ (Menu2)

Customer1 = PJ $_{\text{customer\_id,name,address,gender}}$ (Customer1)

Customer2 = PJ $_{\text{customer\_id,phone,email,gender}}$ (Customer2)

Order1 = SL $_{\text{total<=500}}$(Order1)

Order2 = SL $_{\text{total>500}}$ (Order2)

Table1 = SL $_{\text{status='Available'}}$(Table1)

Table2 = SL $_{\text{status='Unavailable'}}$(Table1)

Feedback1 = PJ $_{\text{fid,item\_id}}$(Feedback1)

Feedback2 = PJ $_{\text{fid,msg,points}}$(Feedback2)

Allocation Schema:

Site:1

- ❖ Customer1
- ❖ MeuItems1
- ❖ Feedback1
- ❖ Tables1
- ❖ Orders1
- ❖ Reservation

Site:2

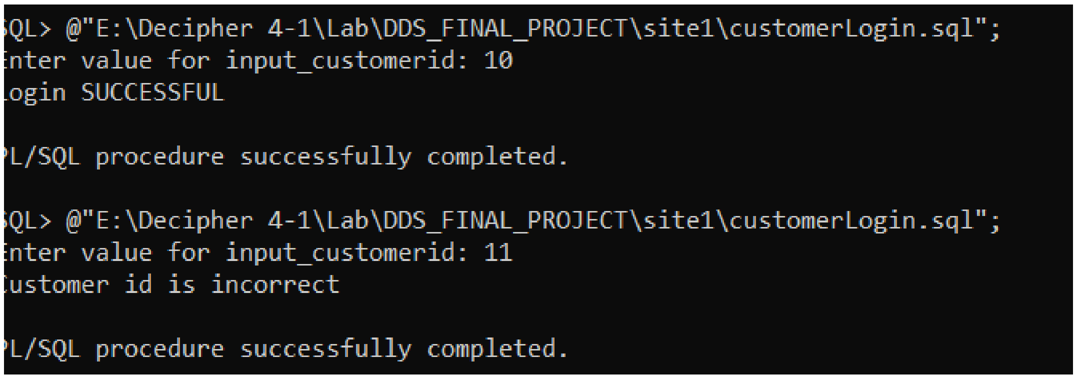
- ❖ Customer2
- ❖ MeuItems2
- ❖ Feedback2
- ❖ Tables2
- ❖ Orders2
- ❖ Placed_Order

Functionalities and Outputs:

- Admin can insert, update, and delete data from tables;
- User can login if him or her id is available in the table
- All related data is updated or deleted from every table when update or delete operation is performed by calling a trigger function.
- Only registered customers can order and only admin can register them.
- Customer can check if tables are available or not.
- Customer can reserve their desired items.
- Customers ordered items are inserted into order table.
- Order will be placed by confirming admin.
- If a user wants, he or she can give feedback.

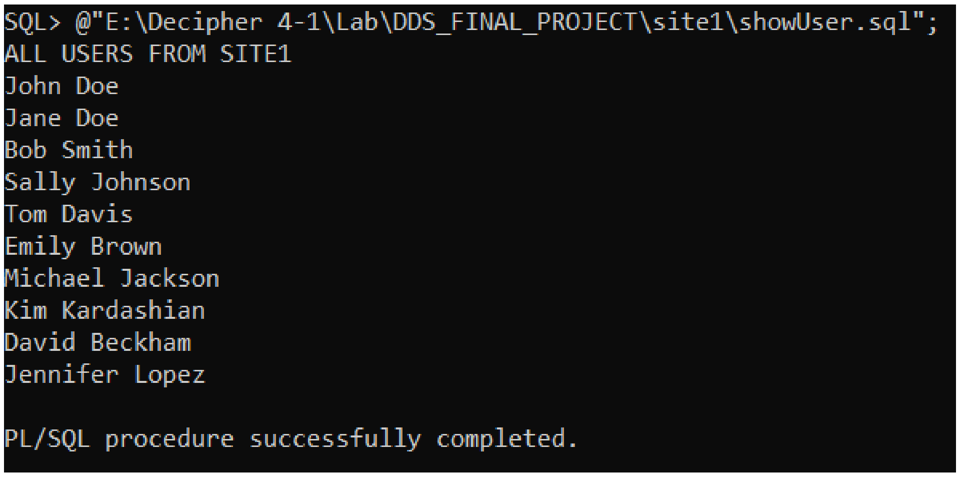
Output:

1. **Customer login**



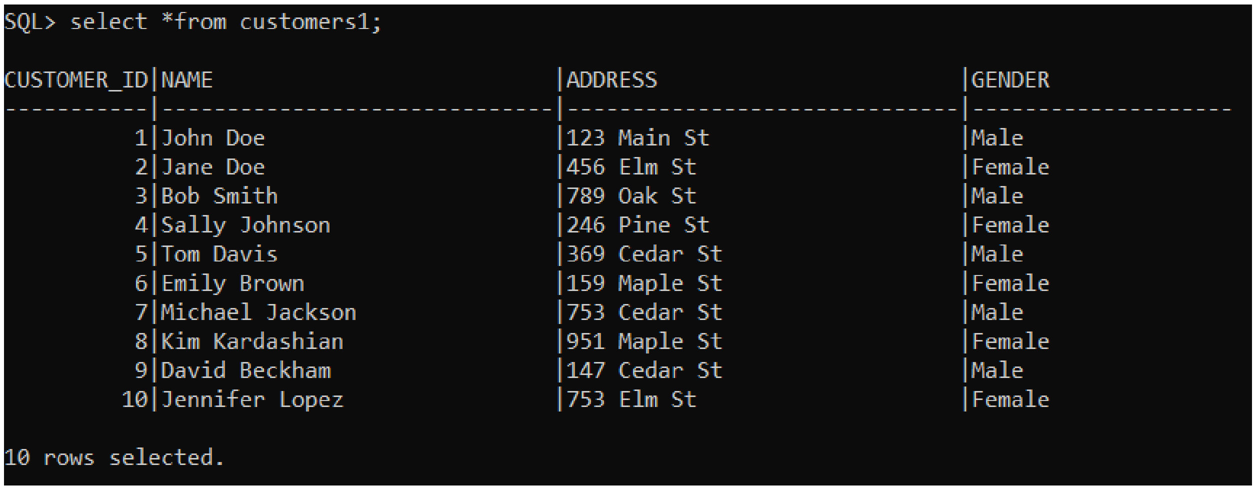
```
QL> @"E:\Decipher 4-1\Lab\DDS_FINAL_PROJECT\site1\customerLogin.sql";
nter value for input_customerid: 10
ogin SUCCESSFUL

L/SQL procedure successfully completed.

QL> @"E:\Decipher 4-1\Lab\DDS_FINAL_PROJECT\site1\customerLogin.sql";
nter value for input_customerid: 11
ustomer id is incorrect

L/SQL procedure successfully completed.
```

2. **All Users**



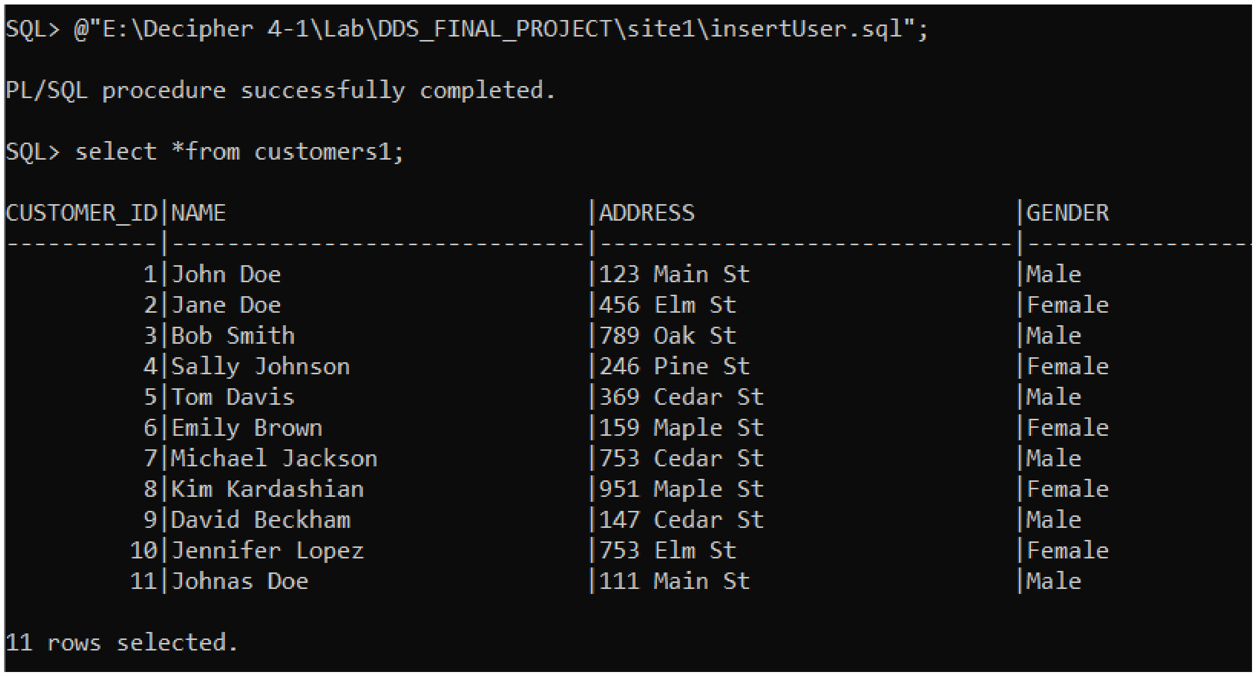
```
SQL> @"E:\Decipher 4-1\Lab\DDS_FINAL_PROJECT\site1\showUser.sql";
ALL USERS FROM SITE1
John Doe
Jane Doe
Bob Smith
Sally Johnson
Tom Davis
Emily Brown
Michael Jackson
Kim Kardashian
David Beckham
Jennifer Lopez

PL/SQL procedure successfully completed.
```

3. **All Users from customer1 table which is located in site1**



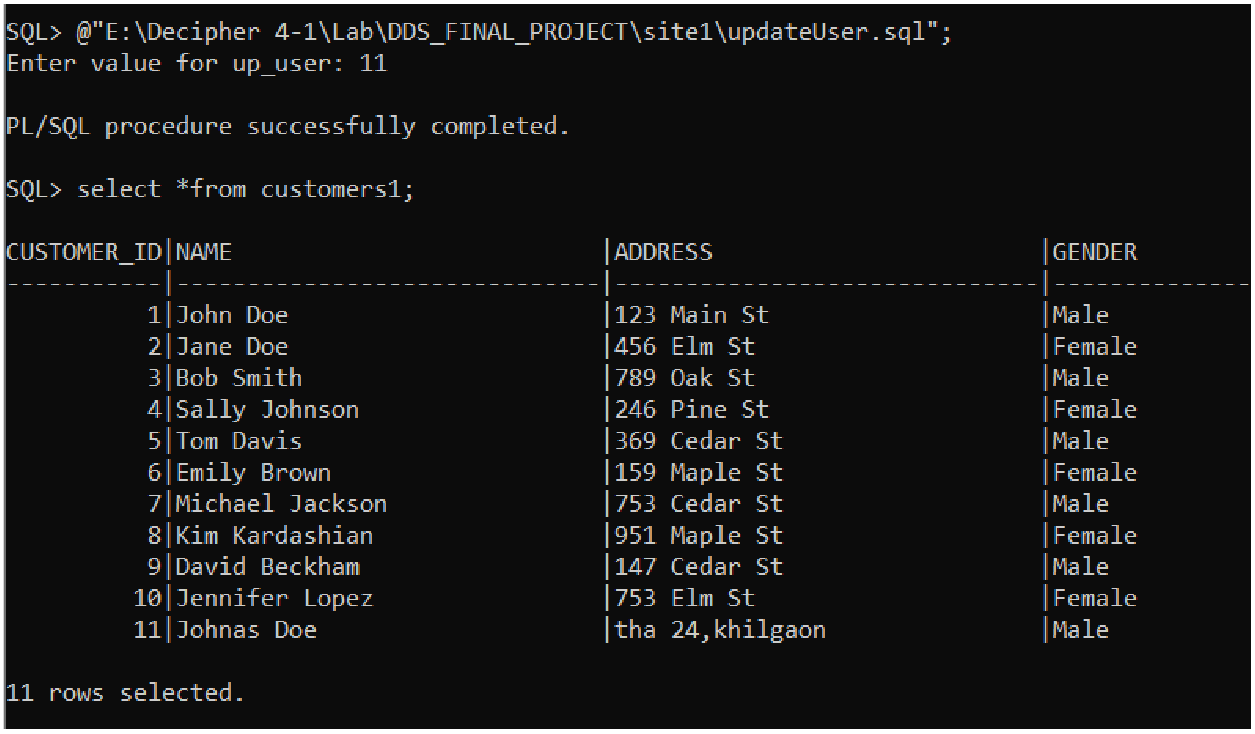
```
SQL> select *from customers1;

CUSTOMER_ID|NAME                          |ADDRESS                       |GENDER
-----------|------------------------------|------------------------------|--------------------
          1|John Doe                      |123 Main St                   |Male
          2|Jane Doe                      |456 Elm St                    |Female
          3|Bob Smith                     |789 Oak St                    |Male
          4|Sally Johnson                 |246 Pine St                   |Female
          5|Tom Davis                     |369 Cedar St                  |Male
          6|Emily Brown                   |159 Maple St                  |Female
          7|Michael Jackson               |753 Cedar St                  |Male
          8|Kim Kardashian                |951 Maple St                  |Female
          9|David Beckham                 |147 Cedar St                  |Male
         10|Jennifer Lopez                |753 Elm St                    |Female

10 rows selected.
```

4. Insert operation for customers



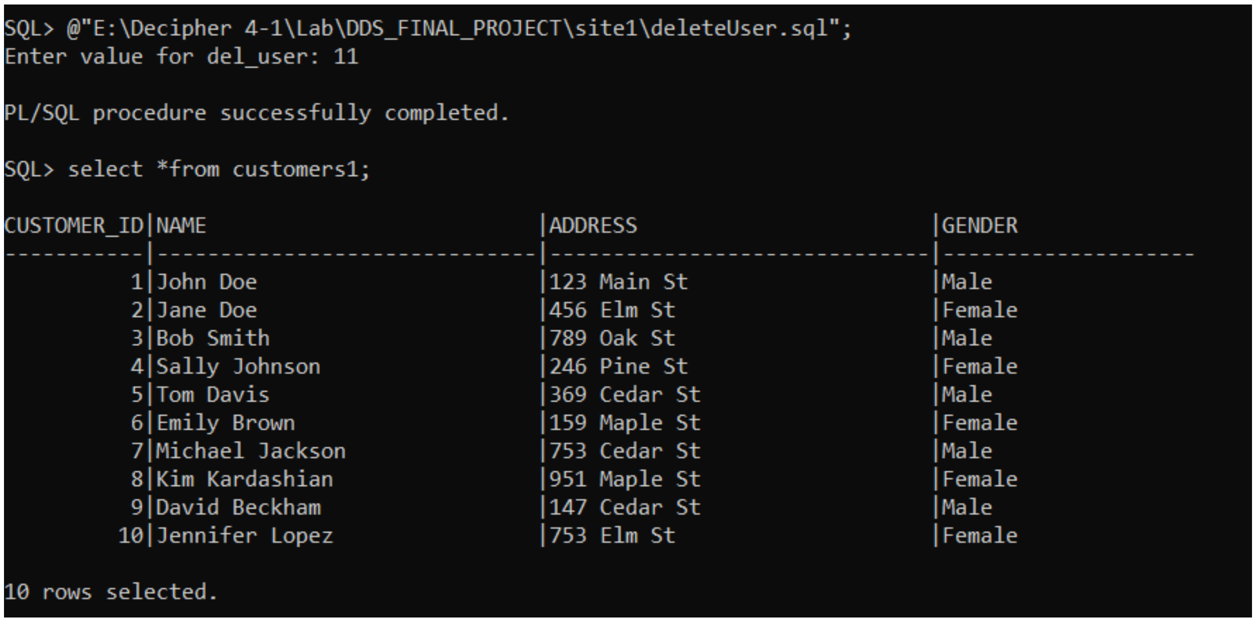
```
SQL> @"E:\Decipher 4-1\Lab\DDS_FINAL_PROJECT\site1\insertUser.sql";

PL/SQL procedure successfully completed.

SQL> select *from customers1;

CUSTOMER_ID|NAME                          |ADDRESS                       |GENDER
-----------|------------------------------|------------------------------|----------------
          1|John Doe                      |123 Main St                   |Male
          2|Jane Doe                      |456 Elm St                    |Female
          3|Bob Smith                     |789 Oak St                    |Male
          4|Sally Johnson                 |246 Pine St                   |Female
          5|Tom Davis                     |369 Cedar St                  |Male
          6|Emily Brown                   |159 Maple St                  |Female
          7|Michael Jackson               |753 Cedar St                  |Male
          8|Kim Kardashian                |951 Maple St                  |Female
          9|David Beckham                 |147 Cedar St                  |Male
         10|Jennifer Lopez                |753 Elm St                    |Female
         11|Johnas Doe                    |111 Main St                   |Male

11 rows selected.
```

5. Update operation for customers



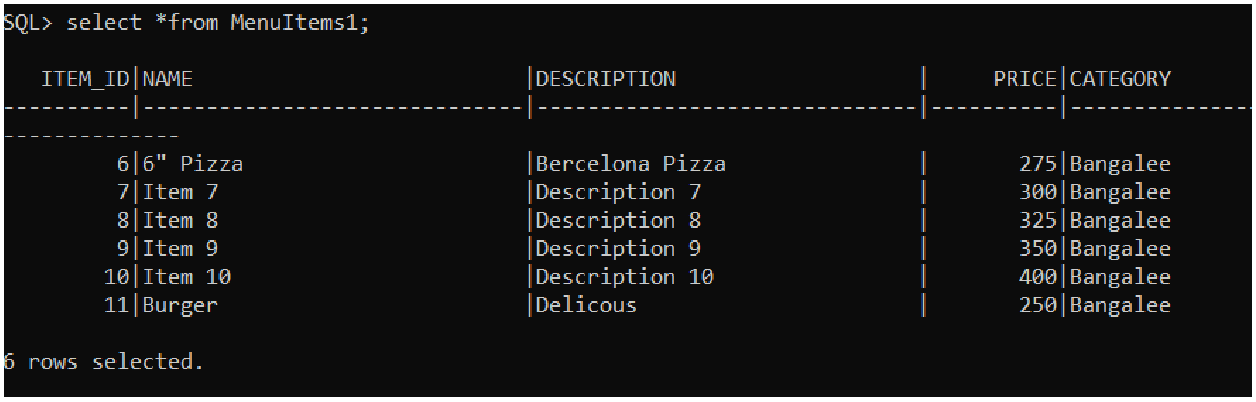
```
SQL> @"E:\Decipher 4-1\Lab\DDS_FINAL_PROJECT\site1\updateUser.sql";
Enter value for up_user: 11

PL/SQL procedure successfully completed.

SQL> select *from customers1;

CUSTOMER_ID|NAME                          |ADDRESS                       |GENDER
-----------|------------------------------|------------------------------|--------------
          1|John Doe                      |123 Main St                   |Male
          2|Jane Doe                      |456 Elm St                    |Female
          3|Bob Smith                     |789 Oak St                    |Male
          4|Sally Johnson                 |246 Pine St                   |Female
          5|Tom Davis                     |369 Cedar St                  |Male
          6|Emily Brown                   |159 Maple St                  |Female
          7|Michael Jackson               |753 Cedar St                  |Male
          8|Kim Kardashian                |951 Maple St                  |Female
          9|David Beckham                 |147 Cedar St                  |Male
         10|Jennifer Lopez                |753 Elm St                    |Female
         11|Johnas Doe                    |tha 24,khilgaon               |Male

11 rows selected.
```

6. **Delete Operation for customers**

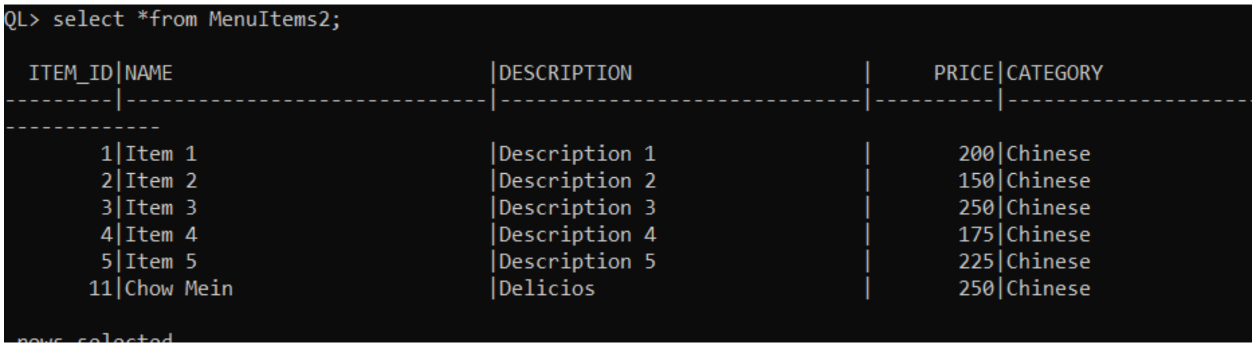


```
SQL> @"E:\Decipher 4-1\Lab\DDS_FINAL_PROJECT\site1\deleteUser.sql";
Enter value for del_user: 11

PL/SQL procedure successfully completed.

SQL> select *from customers1;

CUSTOMER_ID|NAME                          |ADDRESS                       |GENDER
-----------|------------------------------|------------------------------|--------------------
          1|John Doe                      |123 Main St                   |Male
          2|Jane Doe                      |456 Elm St                    |Female
          3|Bob Smith                     |789 Oak St                    |Male
          4|Sally Johnson                 |246 Pine St                   |Female
          5|Tom Davis                     |369 Cedar St                  |Male
          6|Emily Brown                   |159 Maple St                  |Female
          7|Michael Jackson               |753 Cedar St                  |Male
          8|Kim Kardashian                |951 Maple St                  |Female
          9|David Beckham                 |147 Cedar St                  |Male
         10|Jennifer Lopez                |753 Elm St                    |Female

10 rows selected.
```

7. **Menu Items from site1 and site2 (If category='Bangalee' then site 1 and if category='Chinese' then site2)**

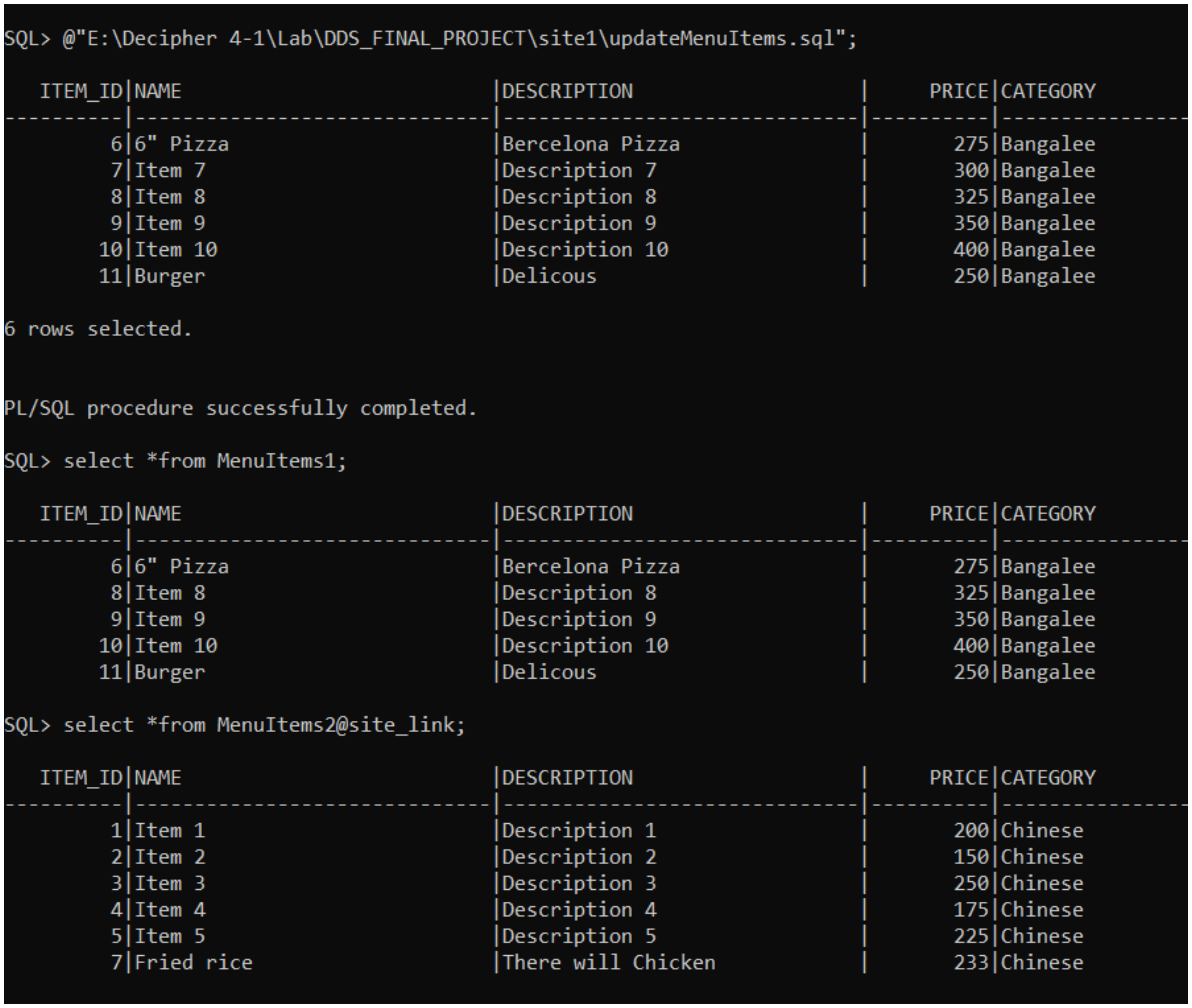


```
SQL> select *from MenuItems1;

   ITEM_ID|NAME                          |DESCRIPTION                   |     PRICE|CATEGORY
----------|------------------------------|------------------------------|----------|--------------
--------------
         6|6" Pizza                      |Bercelona Pizza               |       275|Bangalee
         7|Item 7                        |Description 7                 |       300|Bangalee
         8|Item 8                        |Description 8                 |       325|Bangalee
         9|Item 9                        |Description 9                 |       350|Bangalee
        10|Item 10                       |Description 10                |       400|Bangalee
        11|Burger                        |Delicous                      |       250|Bangalee

6 rows selected.
```



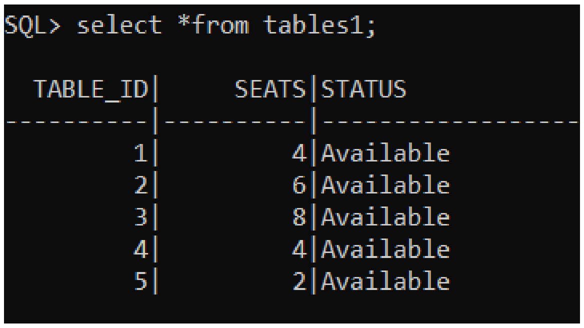
```
QL> select *from MenuItems2;

  ITEM_ID|NAME                          |DESCRIPTION                   |     PRICE|CATEGORY
---------|------------------------------|------------------------------|----------|--------------------
-------------
        1|Item 1                        |Description 1                 |       200|Chinese
        2|Item 2                        |Description 2                 |       150|Chinese
        3|Item 3                        |Description 3                 |       250|Chinese
        4|Item 4                        |Description 4                 |       175|Chinese
        5|Item 5                        |Description 5                 |       225|Chinese
       11|Chow Mein                     |Delicios                      |       250|Chinese

 rows selected
```

8. Update Menu Items



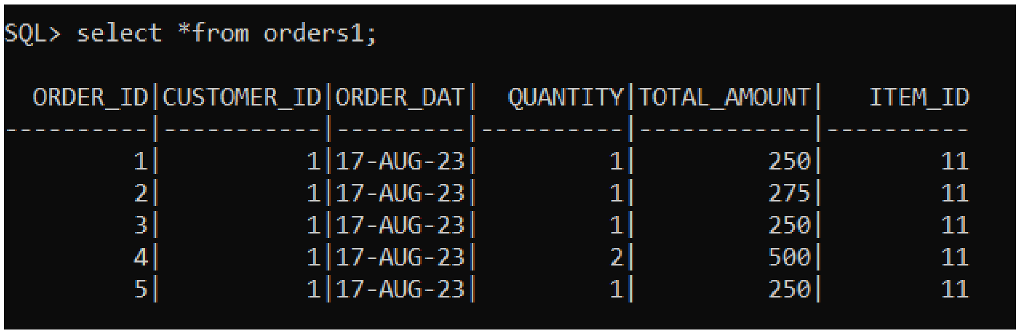
```
SQL> @"E:\Decipher 4-1\Lab\DDS_FINAL_PROJECT\site1\updateMenuItems.sql";

   ITEM_ID|NAME                          |DESCRIPTION                   |     PRICE|CATEGORY
----------|------------------------------|------------------------------|----------|---------------
         6|6" Pizza                      |Bercelona Pizza               |       275|Bangalee
         7|Item 7                        |Description 7                 |       300|Bangalee
         8|Item 8                        |Description 8                 |       325|Bangalee
         9|Item 9                        |Description 9                 |       350|Bangalee
        10|Item 10                       |Description 10                |       400|Bangalee
        11|Burger                        |Delicous                      |       250|Bangalee

6 rows selected.


PL/SQL procedure successfully completed.

SQL> select *from MenuItems1;

   ITEM_ID|NAME                          |DESCRIPTION                   |     PRICE|CATEGORY
----------|------------------------------|------------------------------|----------|---------------
         6|6" Pizza                      |Bercelona Pizza               |       275|Bangalee
         8|Item 8                        |Description 8                 |       325|Bangalee
         9|Item 9                        |Description 9                 |       350|Bangalee
        10|Item 10                       |Description 10                |       400|Bangalee
        11|Burger                        |Delicous                      |       250|Bangalee

SQL> select *from MenuItems2@site_link;

   ITEM_ID|NAME                          |DESCRIPTION                   |     PRICE|CATEGORY
----------|------------------------------|------------------------------|----------|---------------
         1|Item 1                        |Description 1                 |       200|Chinese
         2|Item 2                        |Description 2                 |       150|Chinese
         3|Item 3                        |Description 3                 |       250|Chinese
         4|Item 4                        |Description 4                 |       175|Chinese
         5|Item 5                        |Description 5                 |       225|Chinese
         7|Fried rice                    |There will Chicken            |       233|Chinese
```

9. **Check if table is available or not. (If reservation is happened then available will be unavailable then store in site 2)**



```
SQL> select *from tables1;

  TABLE_ID|     SEATS|STATUS
----------|----------|------------------
         1|         4|Available
         2|         6|Available
         3|         8|Available
         4|         4|Available
         5|         2|Available
```

10. Customer Order (If order is <=500 then orders1 otherwise orders2)



```
SQL> select *from orders1;

  ORDER_ID|CUSTOMER_ID|ORDER_DAT|  QUANTITY|TOTAL_AMOUNT|   ITEM_ID
----------|-----------|---------|----------|------------|----------
         1|          1|17-AUG-23|         1|         250|        11
         2|          1|17-AUG-23|         1|         275|        11
         3|          1|17-AUG-23|         1|         250|        11
         4|          1|17-AUG-23|         2|         500|        11
         5|          1|17-AUG-23|         1|         250|        11
```

We have another three table names placed_order that show if order is placed, reservation that show if anyone reserved any tables and feedback table that show the feedback of customers.

Contribution:

190204005 – M Saymon Islam Iftikar

- Create tables and customers login system.
- CRUD (Create, Read, Update, Delete) operations for Customers.
- CRUD operations for Menu Items.
- Customers Order manipulation
- Reservations manipulation
- Feedback manipulation
- Create package, function, procedure, trigger, view, and exception handling
- Report writing 50%

190204004 - Abu Hanjala

- Create tables for availability check and manipulate
- Placed Order and manipulate
- Report writing 50%

Conclusion:

We believe that using a solid distributed database system in conjunction with efficient management techniques can assist restaurant owners and managers in succeeding in the fiercely competitive market of today. During the implementation phase of this project, sometimes we may face some difficulties. But regardless of the difficulties, we offer a solid framework for handling the challenges of managing a restaurant.