Specification of Features

Individual Features

FEATURE 1

Input: customer name, address, state, zip, email.
Calling: exec feature('Riya','Paradise House','MD','21227','riya122@gmail.com')
Output: screen output of 'the client already exists'

Otherwise, a new row inserted into the customer table and a screen output of the newly assigned customer ID.

FEATURE 2

Input: Customer Email ID
Calling : exec feature('eric@gmail.com')
Output: No such a customer with the email ID

Otherwise print out the profile of the customer, including name, address, state, zip code, email, credit, total number of orders with status 2 (delivered) in the last six months and total amount spent (sum of total cost for orders with status 2) in the last six months.

FEATURE 3

Search for a restaurant by category. Input is a part of category name (e.g., for fast food the input could be just 'fast'). Please print out the name, average review score, average wait time, and zip code for restaurants that are open and match the input category name.

Scenario 1:
This is the scenario when the input (category substring) does not exist or if there is a spelling error1.

Input: vegetarian
Calling: exec find_restaurant_by_category('vegetarian');
Output:
No such category
Over here the output is 'No such category' because 'vegetarian' is neither a string nor a substring in the category table.

Scenario 2:
This is the scenario when the input exists.

Input: vegan
Calling: exec find_restaurant_by_category('vegan');
Output:
ihop | zip code: 21228 | average wait time: 30 minutes | average review score: 4.5
Database consists of a category with substring vegan. The output of the query is restaurants that serve the vegan category food along with its zip code, average wait time, average review score.

FEATURE 4

Show dishes offered by a restaurant. Input is a restaurant ID. The procedure first checks whether this is a valid restaurant ID. If not, please print a message 'no such restaurant'. Otherwise print out all dishes in this restaurant, along with dish name and price

Scenario 1:
This is the scenario when the input (Restaurant ID) does not exist.

Input: 10
Calling: exec show_dishes_by_restaurant(10);
Output:
No such Restaurant
As there is no restaurant with ID 10, output is 'No such Restaurant'

Scenario 2:
This is the scenario when the input (Restaurant ID) exists.

Input :1
Calling: exec show_dishes_by_restaurant(1);
Output:
Dish Name: Coffee and Hot Chocolate at 5.99
Dish Name: Eggs Benedict at 11.79
Dish Name: Turkey Sausage Links at 3.29
Dish Name: Pot Roast Dinner at 12.49

The output is all the dishes served at restaurant with ID 1.
Format of the output is dishes at price

FEATURE 5

Scenario 1: If given Cart ID is invalid.

Input : Cart ID

Example of calling this feature : Exec showDishes(1149);

Output : Screen output of 'Invalid Cart Id'

Scenario 2: If cart ID is valid then.

Example of calling this feature : Exec showDishes(114);

Output : screen output of dishes names, price and quantity for all the rows

Example Output:

1: Dish Name: Impossible Whooper, Dish Price: $11.69, Dish Quantity: 2

2: Dish Name: Double Whooper, Dish Price: $12.73, Dish Quantity: 1

3: Dish Name: Big King XL, Dish Price: $11.18, Dish Quantity: 2

FEATURE 6

Scenario 1: If given Cart ID is invalid.

Input : dish ID, cart ID and quantity of dish

Example of calling this feature : Exec removeDish(311, 112, 2);

Output : screen output of 'Invalid Input' if there is no cart id with the given dish id.

Scenario 2: If cart ID is valid and quantity of dish is greater than 1.

Input : dish ID, cart ID and quantity of dish

Example of calling this feature : Exec removeDish(311, 112, 2);

Output : Find the quantity of a dish and if it's greater than 1, update the quantity of that dish from the cart table to quantity -1. As well as output the message saying, 'quantity reduced'.

Scenario 3: If cart ID is valid and quantity of dish is 1

Input : dish ID, cart ID and quantity of dish

Example of calling this feature : Exec removeDish(311, 112, 2);

Output : If the quantity is 1 then delete the row from the cart table where the combination of dish id with 1 quantity and cart id exists and screen output of 'dish removed'.

FEATURE 7

Input: order_id, new_status, input_time

Scenario 1: Update order status with **invalid order id**.
Calling: exec update_status('W023345', 1, timestamp '2022-10-30 09.09.09.00');
Output: A message will be displayed as 'Invalid Order ID'.

Scenario 2: Update order status with status id 1 i.e. '**is in progress**'.
Calling: exec update_status('W123345', 1, timestamp '2022-10-30 09.09.09.00');
Output: Order status will be updated and the same will be displayed on the screen. For example - 'Order W123345 updated. Order Status - 1'.

Scenario 3: Update order status with status id 2 i.e. '**delivered**'.
Calling: exec update_status('W123345', 2, timestamp '2022-10-30 09.09.09.00');
Output: Order status will be updated and the same will be printed on the screen. Also, a new entry in the message table will be created with the message body as 'Your order W123345 has been delivered!'.
For Example - 'Order W123345 updated. Order Status - 2
Message M901 created'.

Scenario 4: Update order status with status id 3 i.e. '**canceled**'.
Calling: exec update_status('W123345', 3, timestamp '2022-10-30 09.09.09.00');
Output: Order status will be updated and the same will be printed on the screen. Also, a new entry in the message table will be created with the message body as 'Your order W123345 has been canceled and refund issued!'. Additionally, a new row in the payment table will be inserted with the payment amount as negative of the original amount.
For Example - 'Order W123345 updated. Order Status - 3.
Message M901 created.
Payment T101 recorded'.

FEATURE 8

Input: customer_id, restaurant_id, review_date, review_score, review_comment, average_score

Scenario 1: Update review status with **invalid customer ID**
Calling: update_review(10,1,1, timestamp '2022-10-10 09:05:00.00',09, 'Excellent Food Mama Mia',5);
Output: A message will be displayed as ‘Invalid customer ID’.

Scenario 2: Update review status with **valid customer ID**
Calling: update_review(1,1,1, timestamp '2022-10-10 09:05:00.00',09, 'Excellent Food Mama Mia',5);
Output: A new row of the customer’s review will be added to the database.

Scenario 3: Update review status with **invalid restaurant ID**
Calling: update_review(1,10,1, timestamp '2022-10-10 09:05:00.00',09, 'Excellent Food Mama Mia',5);
Output: A message will be displayed as ‘Invalid restaurant ID’.

Scenario 4: Update review status with **valid restaurant ID**
Calling: update_review(1,1,1, timestamp '2022-10-10 09:05:00.00',09, 'Excellent Food Mama Mia',5);
Output: A new row of the customer’s review will be added to the database.

Scenario 5: Update new average score
Calling: update_review(1,1,1, timestamp '2022-10-10 09:05:00.00',09, 'Excellent Food Mama Mia',5);
Output: Take the old average and update it with the new average.

FEATURE 9

Input: restaurant_id

Scenario 1: Update review status with **invalid restaurant ID**
Calling: display_data(restaurant_id);
Output: A message will be displayed as ‘Invalid restaurant ID’.

Scenario 2: Update review status with **valid restaurant ID**
Calling: display_data(restaurant_id);
Output: (1,1,1, timestamp '2022-10-10 09:05:00.00',09, 'Excellent Food Mama Mia',5);

Group Features

FEATURE 10

Input: Customer ID, Restaurant ID, Dish ID

Output: First we will check if the customer ID is valid, if it is not valid then we will print a Customer ID invalid message. Then we will check if the restaurant ID is valid, if it is not we will print a invalid restaurant ID message. Then we will check if the dish, if it belongs to the input restaurant, if it does not then we will print invalid dish ID message.If no shopping cart exists then create a new shopping cart for the customer and restaurant. Then we will print the new cart ID. We can check if the dish is already in the cart, if yes increase the quantity by one or insert a new row in the table to keep the dish in the cart.

Scenario 1:

To check if customer ID is valid.

Input : Customer ID_101

Calling : exec feature10(101);

Output : Customer ID not valid, No such customer

Scenario 2:

Checking if restaurant ID is valid

Input : Restaurant ID_109

Calling: exec feature10(190);

Output: Restaurant ID not valid.

Restaurant, Restaurant Closed.

Scenario 3:

Checking if the dish belongs to the input restaurant

Input: Dish ID_18

Calling: exec feature10(18)

Output: Invalid Dish ID or dish does not belong to the input restaurant.

Scenario 4:

If Customer ID and the restaurant ID is valid but the dish ID correlates with the restaurant

Input :101,104

Calling: exec feature10(108,189)

Output: The dish ID is invalid

Scenario 5:

If no existing cart for a customer

Input: 102, 108

Calling: exec feature10(89,78);

Output:145

A new Cart ID has been created - 145, there is no existing cart ID for 102 at restaurant with the Restaurant ID 2.

Scenario 6:

If the dish is not in the cart then we need to insert a new row in the table to keep the dish in the cart.

Input: 197,899

Calling:exec feature10(197,899)

Output: A new row in the table is added to keep the dish in carts.

FEATURE 11

Scenario 1: If given Cart ID is invalid.

Input : cart ID

Example of calling this feature : Exec totalAmtCart(1101);

Output : Screen output of 'Invalid Cart Id'

Scenario 2: If given Cart ID is Valid.

Input : cart ID, checkout time(Current time), flag(Delivery method), discount start time, discount end time, discount type, order delivery method, customer's zip ode and restaurant zip code

Example of calling this feature : Exec totalAmtCart(111);

Output : Sum of the price of each dish * quantity of dish in the cart

Initialize and update the totalAmount variable with the above sum.

Check if discount is valid at the checkout time by comparing it with discount start and end time from Customer_Discount table.

If discount is valid, then check discount type from discount table –

If its fixed percentage i.e 2 then assign totalAmount = (1- discount_amount from discount table) * totalAmount.

Else if fixed amount i.e 3 then assign totalAmount = totalAmount-amount.

Add the delivery fee, if it's not a pickup order (check using flag column in Orders table) –

If zip code of the customer's address from customer table is same as restaurant's zip code from restaurant table then delivery fee = 2 else delivery fee = 5

If discount type is free delivery i.e. 1, then delivery fee = 0

FEATURE 12

Input: cart_id, order_time, delivery_method, estimated_time, tip, payment_method

Scenario1: Generate an order with an **invalid cart id**.
Calling: exec generate_order(7381, timestamp '2022-10-30 09:09:09.00', 1,
timestamp '2022-10-30 09:09:09.00', 10, 3);
Output: A message will be displayed as 'Invalid Cart ID'.

Scenario 2: Generate an order with a **valid cart id**.
Calling: exec generate_order(111, timestamp '2022-10-30 09:09:09.00', 1,
timestamp '2022-10-30 09:09:09.00', 10, 3);
Output: An order time will be generated and the order ID will be printed on the screen. A new entry in a message table will be added with the message body as 'A new order W123345 is placed at restaurant Pizza Hut with estimated 120 mins and amount 500.12'. Additionally, a new row in a payment table will be added with the corresponding total amount. Also, dishes in the order will be added to the dish_order table deleting values from the dish_cart and cart table.
For Example - 'Order W123345 generated
Message M901 created
Payment T102 recorded
Cart 111 deleted.'

FEATURE 13

Advanced Search

The procedure first checks if the customer ID is valid. If not print a message “Invalid Customer ID”, else it returns all the restaurants that satisfy all of the conditions.

Input: customer ID, list of category names, minimal review score, and wait time
Output: Displays “invalid input” if customer ID is invalid. Otherwise displays a list of restaurants that satisfy all the conditions.

Example of calling this features:

exec search_resaurants(11,(‘paneer masala’,’north indian’), 5, 40);

*passing varray containing a list of categories to the procedure.

FEATURE 14

Feature 14: Restaurant recommendation. The input is a customer ID. This procedure does the following.
1) check whether the customer ID is valid. If not, print an error message and stop.
2) Find restaurants where customers have placed an order. Please exclude the same customer in the input. Please print out these restaurants' IDs.
3) Find customers who have placed orders in any restaurant in step 2). Please print out these customer IDs.
4) Find other restaurants these customers (in step 3) go to. Please exclude those restaurants in step 2) (i.e., the customer already visited).
5) Print out the id and names of these restaurants, their addresses, and average reviews.

Scenario 1:
When the input customer id is not valid

Input: 8
Calling: exec feature14(8);
Output:
No such Customer
The output is ‘No such customer’ because there exists no customer with ID 8.

Scenario 2:
When data exists partially

Input: 3

Calling: exec feature14(3);
Output:
restaurant 3
No such customer
No recommendation

This is the case when the input customer ID exists, but no other customer has ordered from the restaurant where Input customer ID has ordered from.

Scenario 3:
When the details exist completely.

Input: 2
Calling: exec feature14(2);
Output:
restaurant 1 restaurant 2
customer 1
restaurant 3

Our input in Customer with ID 2.
First part of output is all restaurant ID where customer 2 has placed orders at (which is restaurant 1 and restaurant 2 in our case). Second part is the customer ID of customers who have also purchased from the same restaurants (restaurant 1 and restaurant 2). This excludes the input customer id (customer 2). Third part is restaurant ID where following customer in second part(customer 1) has purchased from but input customer(customer 2) hasn't in order to recommend the restaurant to input customer ID (customer 2)