Bake n' Take: Database Systems Project

Amani Merhi
202304693

Lorence Khalifeh
202304550

Sandi El Zein
202301685

Project Idea

Our absolute favorite websites are those of bakeries where we can order a variety of delicious freshly baked goods, satisfying our savory and sweet cravings. This is why we decided to develop a website for a bakery and now base our Database Project on building a database for our online bakery website “Bake n’ take.” Our database will be used to store important data related to the food items and customers in a structured and organized manner. In this way, managing customer orders and payments will become so much easier, while handling customer information, messages, and reviews will be so much more efficient. It will also provide valuable insights that will help improve our pastries and services. Therefore, by building a competent and robust bakery management system we will ensure the successful running of the online bakery business and customers’ satisfaction.

Project Objectives

Many bakeries struggle to manage customer orders and deliver their requested baked goods freshly on time. Thus, for the bakery to optimize customer ordering and ensure exceptional quality of food and rapid delivery, a database system is of paramount importance. Furthermore, to understand customer preferences, keep track of the purchasing trends, and handle high demands a database system will certainly be necessary. Not to forget, the essential role that it plays in improving the food item assortment, tailoring weekly offers, enhancing overall user experience, and increasing customer contentment. And finally, our bakery can improve their goods and services based on customer feedback and reviews.

Overview of Bake n’ take Database

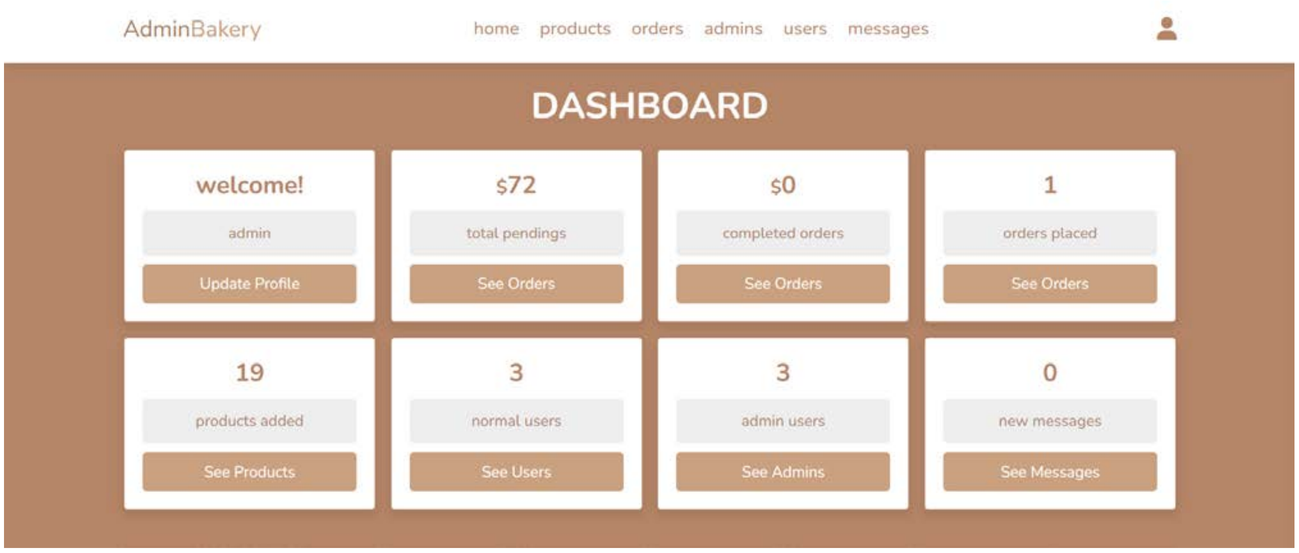
In our project, upon customers registering on the “Bake n’ take” website their information will be added to the Users table and will be assigned a RoleId 2 (in the Role table). While all administrators will already be added to the Users table and have RoleId 1 assigned to them. Once the customer logs in they will be able to browse the baked goods included in the FoodItem table, place orders that will be added to Orders table, provide reviews and ratings that will be inserted in Reviews table, send messages that will be added to Messages table, and add food items to their cart or Wishlist (Cart table and Wishlist tables). Meanwhile, administrators can manage user accounts and customer orders (in Users and Orders tables), update orders status (in Orders table), add food items to FoodItem table and sort them to their corresponding categories in Category table, and view customers’ messages and reviews (in Reviews and Messages tables respectively). Yet for us to elevate the user experience throughout the process of them logging into the website, browsing the pastries, adding to their cart or Wishlist some of them, placing an order, and finally sending a message or leaving a review, a database is extremely crucial. Thus, we decided to utilize Microsoft SQL as our DBMS where we can implement SQL queries that will help in enhancing the customers’ journey; while also, simplifying administrative tasks and providing beneficial data insights.

Planning the project

As we had our online bakery website “Bake n’ take” already developed, the first thing we did was decide upon the project’s entities, their respective attributes, and the relationships among the entities. Then we built the ER Diagram to be able to visualize the database structure, which we then converted into a Relational Model that helped us in writing the DDL for developing the relations and the relationships among them. And after generating the Database Diagram, we formulated a list of diverse questions for data retrieval. These questions were then translated into SQL queries that include update and query operations, nested queries, arithmetic and comparison operators, and diverse types of joins for comprehensive data extraction. Executing these queries will highlight the effectiveness of our database and will help in managing the bakery shop more efficiently. By that our bakery database is ready to make a worthy contribution to the online shop.

Website

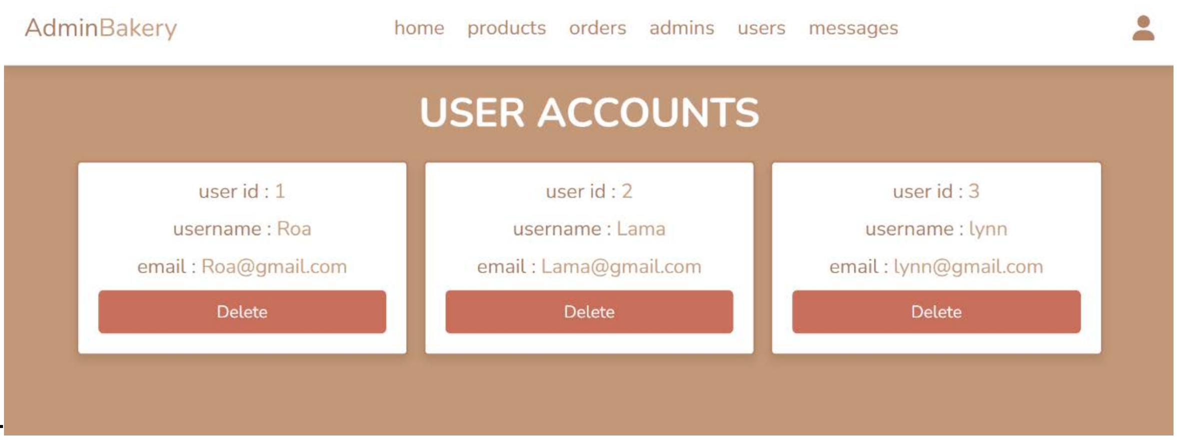
- Admin Dashboard


AdminBakery home products orders admins users messages

DASHBOARD

welcome!	$72	$0	1
admin	total pendings	completed orders	orders placed
Update Profile	See Orders	See Orders	See Orders
19	3	3	0
products added	normal users	admin users	new messages
See Products	See Users	See Admins	See Messages

- Users account display for admin

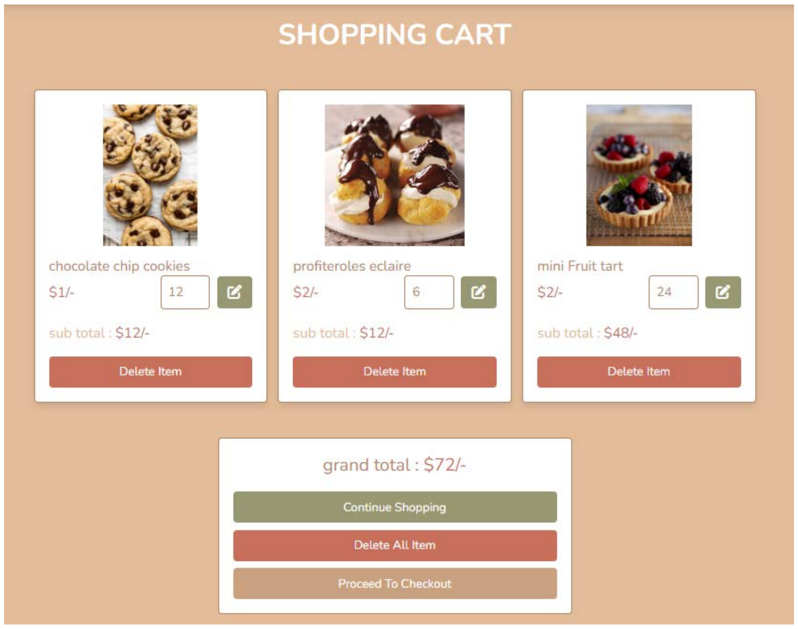

AdminBakery home products orders admins users messages

USER ACCOUNTS

user id : 1	user id : 2	user id : 3
username : Roa	username : Lama	username : lynn
email : Roa@gmail.com	email : Lama@gmail.com	email : lynn@gmail.com
Delete	Delete	Delete

-

- Cart



SHOPPING CART

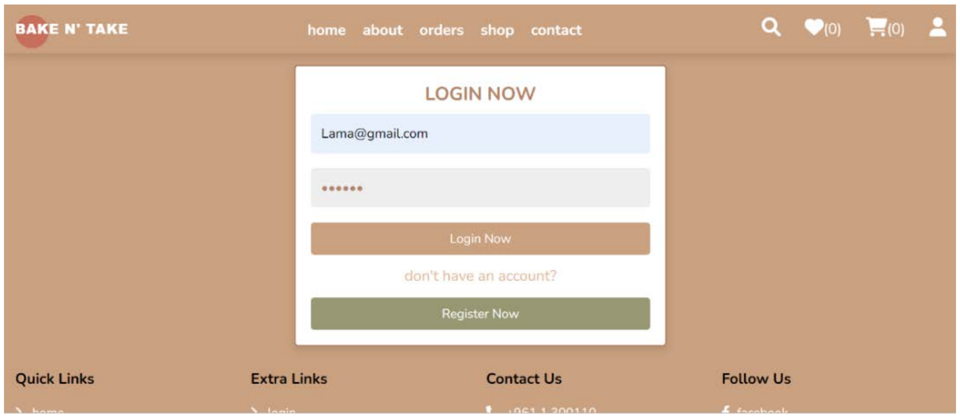
chocolate chip cookies
$1/- 12
sub total : $12/-
Delete Item

profiteroles eclaire
$2/- 6
sub total : $12/-
Delete Item

mini Fruit tart
$2/- 24
sub total : $48/-
Delete Item

grand total : $72/-
Continue Shopping
Delete All Item
Proceed To Checkout

- Login for users

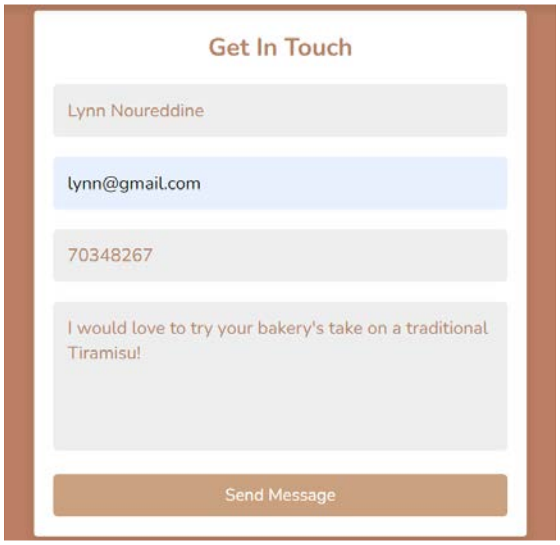


BAKE N' TAKE home about orders shop contact (0) (0)

LOGIN NOW
Lama@gmail.com
••••••
Login Now
don't have an account?
Register Now

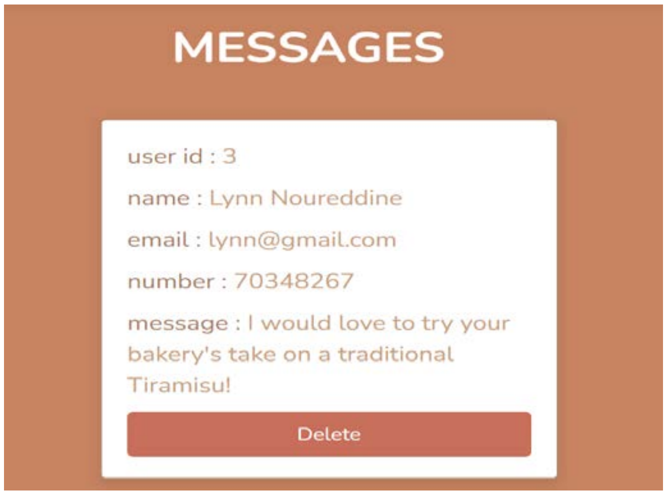
Quick Links Extra Links Contact Us Follow Us

- Contact Us



Get In Touch
Lynn Noureddine
lynn@gmail.com
70348267
I would love to try your bakery's take on a traditional Tiramisu!
Send Message

- Messages display for admin



MESSAGES

user id : 3

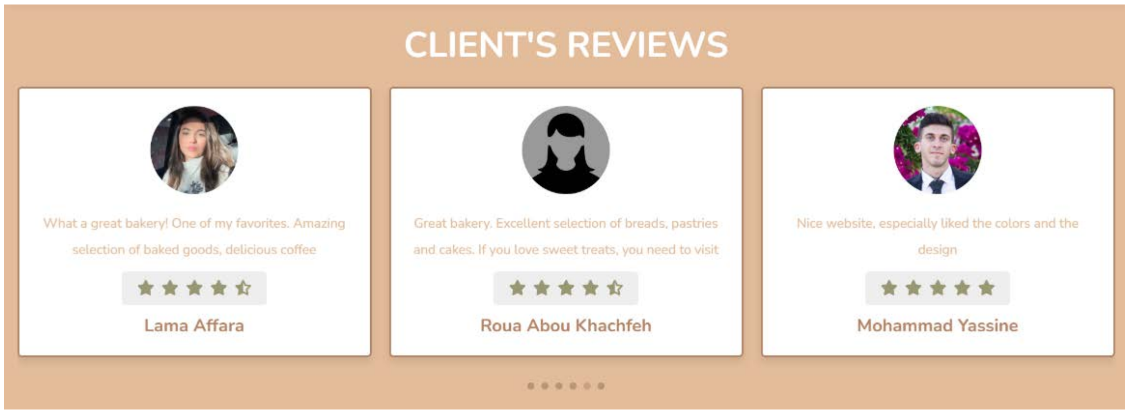
name : Lynn Noureddine

email : lynn@gmail.com

number : 70348267

message : I would love to try your bakery's take on a traditional Tiramisu!

- Reviews



CLIENT'S REVIEWS

What a great bakery! One of my favorites. Amazing selection of baked goods, delicious coffee

Lama Affara

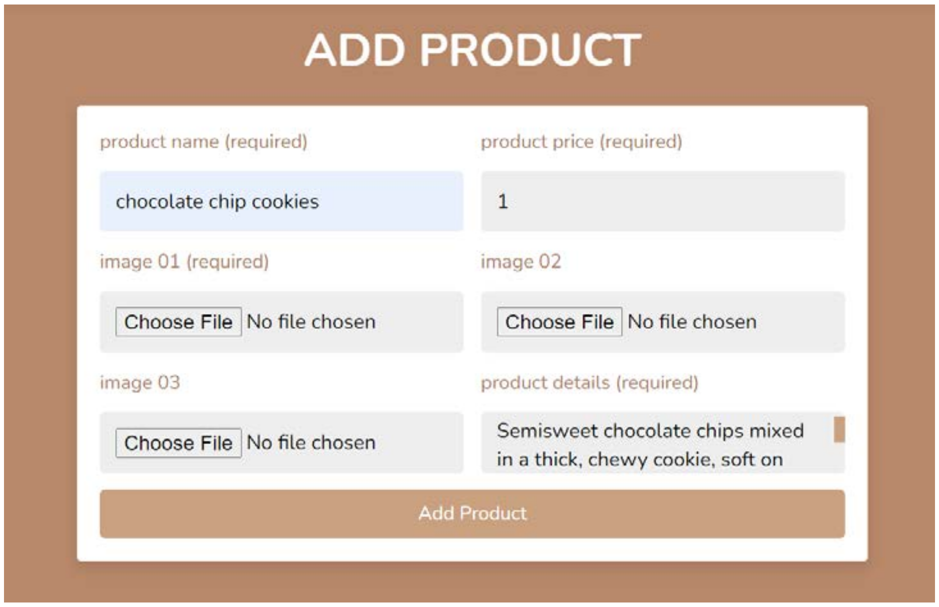
Great bakery. Excellent selection of breads, pastries and cakes. If you love sweet treats, you need to visit

Roua Abou Khachfeh

Nice website, especially liked the colors and the design

Mohammad Yassine

- Adding products by admin



ADD PRODUCT

product name (required): chocolate chip cookies

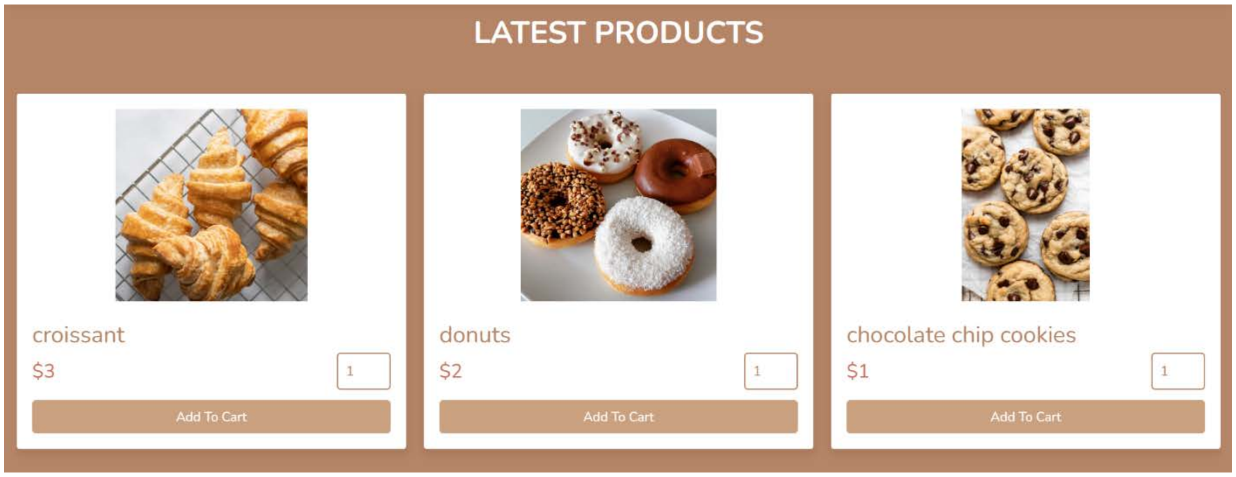
product price (required): 1

image 01 (required): Choose File No file chosen

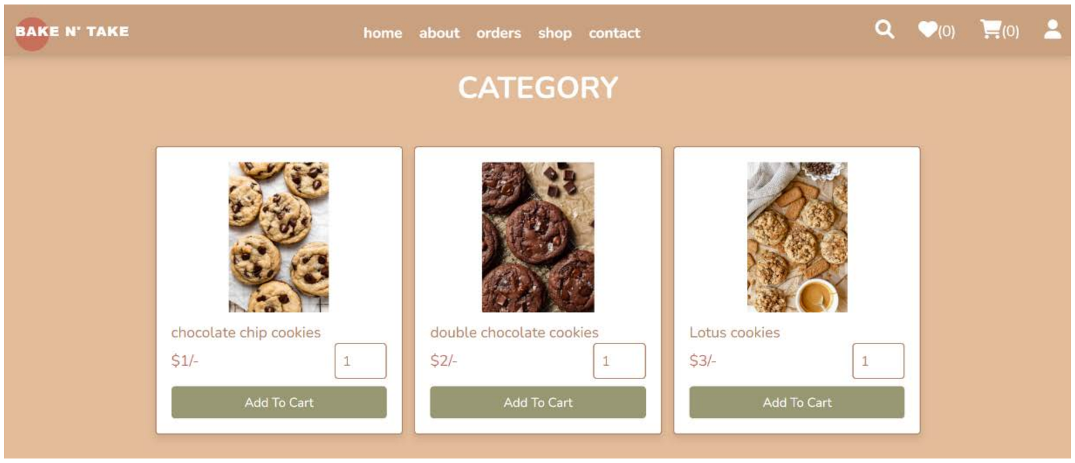
image 02: Choose File No file chosen

image 03: Choose File No file chosen

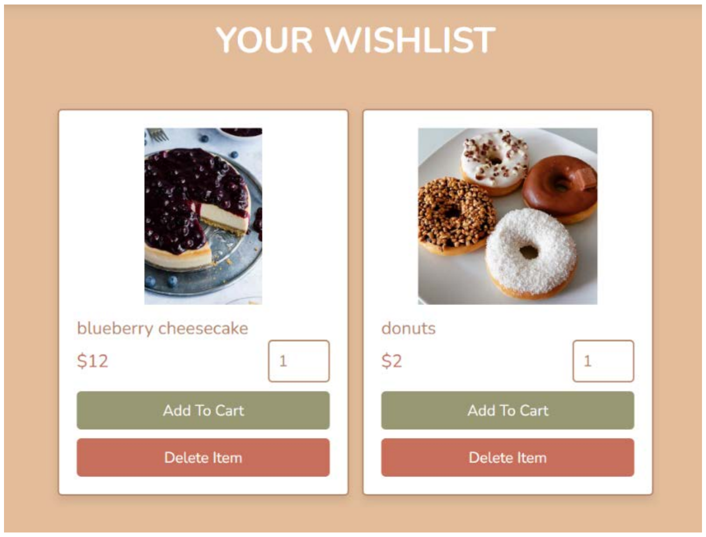
product details (required): Semisweet chocolate chips mixed in a thick, chewy cookie, soft on o Latest Products display



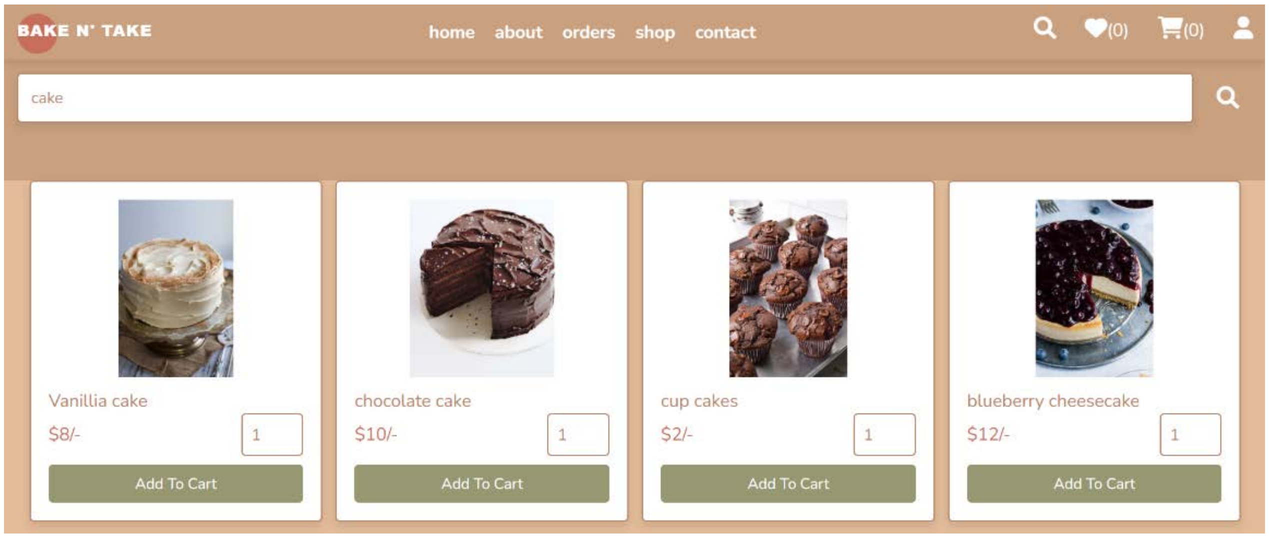
o Products display by Category



o Wishlist:



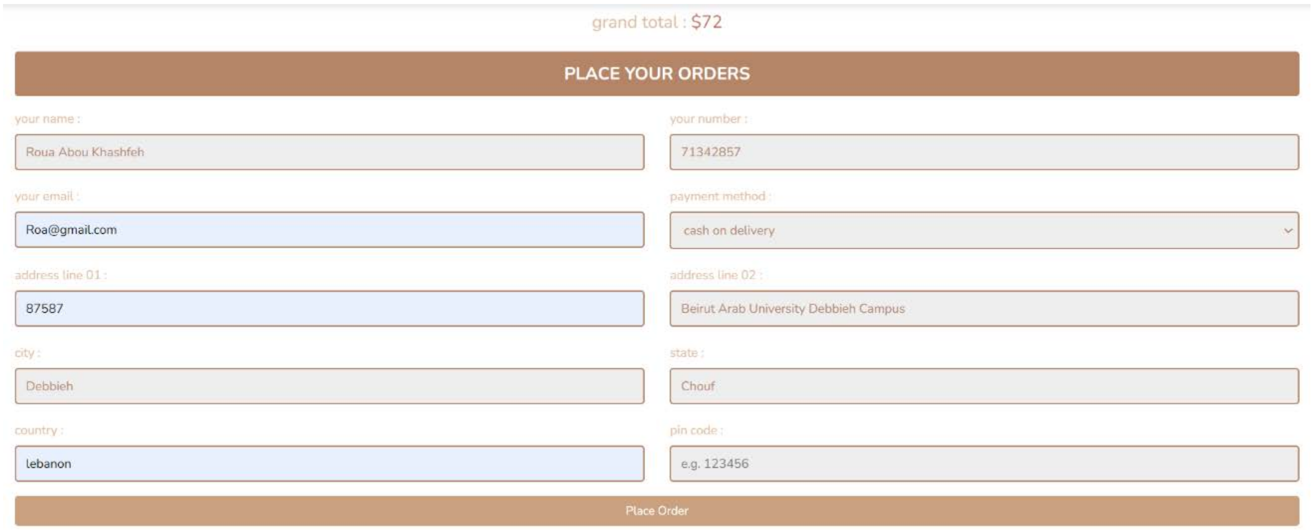
o Search a Product by Name



cake

Vanillia cake	chocolate cake	cup cakes	blueberry cheesecake
$8/-	$10/-	$2/-	$12/-
1	1	1	1
Add To Cart	Add To Cart	Add To Cart	Add To Cart

o Insert info to place an Order

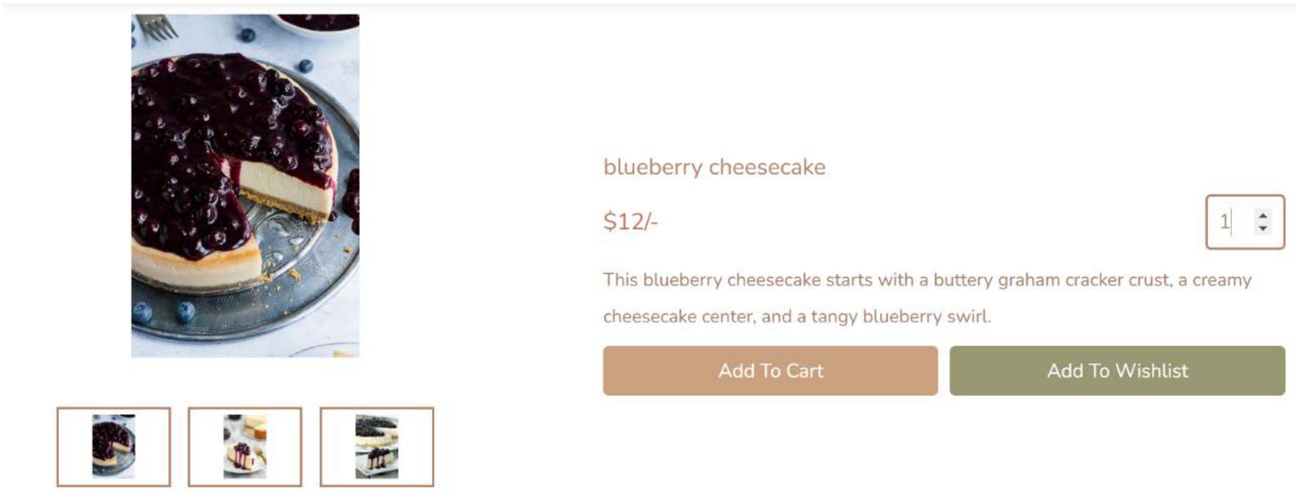


grand total : $72

PLACE YOUR ORDERS

your name : Roua Abou Khashfeh	your number : 71342857
your email : Roa@gmail.com	payment method : cash on delivery
address line 01 : 87587	address line 02 : Beirut Arab University Debbieh Campus
city : Debbieh	state : Chouf
country : lebanon	pin code : e.g. 123456

o Display information of a foodItem



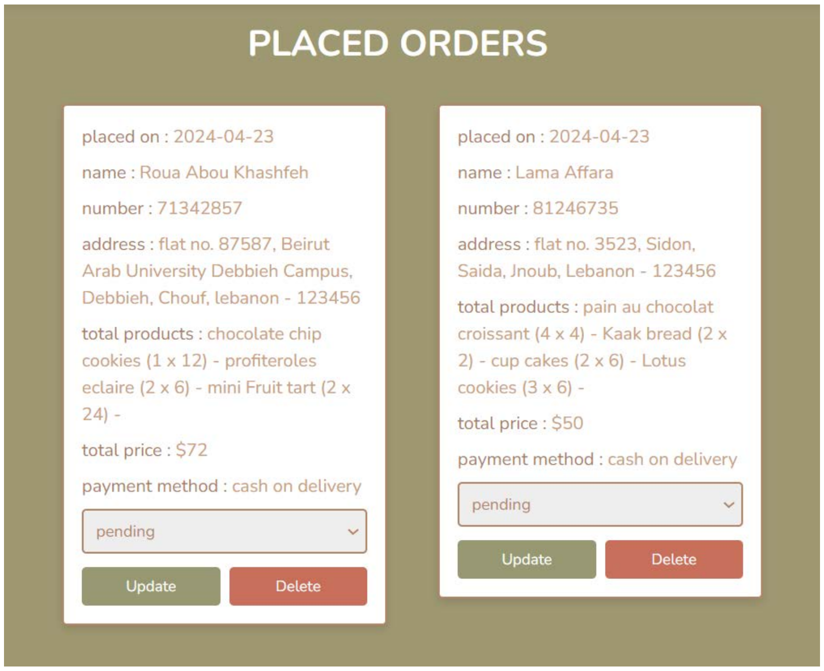
blueberry cheesecake

$12/-

1

This blueberry cheesecake starts with a buttery graham cracker crust, a creamy cheesecake center, and a tangy blueberry swirl.

- View Pending Orders



PLACED ORDERS

placed on : 2024-04-23

name : Roua Abou Khashfeh

number : 71342857

address : flat no. 87587, Beirut Arab University Debbieh Campus, Debbieh, Chouf, lebanon - 123456

total products : chocolate chip cookies (1 x 12) - profiteroles eclaire (2 x 6) - mini Fruit tart (2 x 24) -

total price : $72

payment method : cash on delivery

pending

Update Delete

placed on : 2024-04-23

name : Lama Affara

number : 81246735

address : flat no. 3523, Sidon, Saida, Jnoub, Lebanon - 123456

total products : pain au chocolat croissant (4 x 4) - Kaak bread (2 x 2) - cup cakes (2 x 6) - Lotus cookies (3 x 6) -

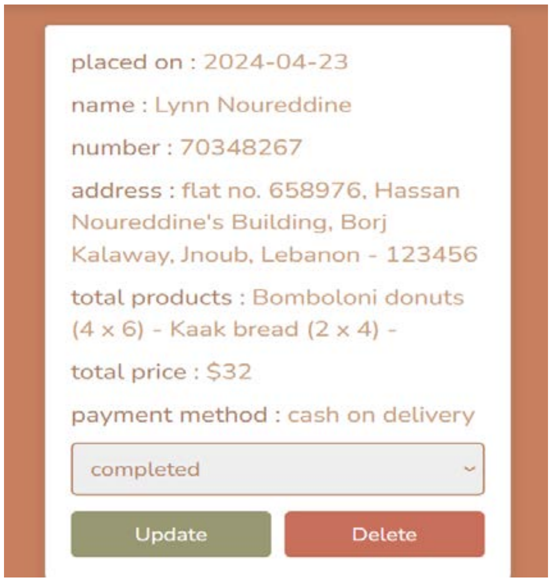
total price : $50

payment method : cash on delivery

pending

Update Delete

- View Completed Orders



placed on : 2024-04-23

name : Lynn Noureddine

number : 70348267

address : flat no. 658976, Hassan Noureddine's Building, Borj Kalaway, Jnoub, Lebanon - 123456

total products : Bomboloni donuts (4 x 6) - Kaak bread (2 x 4) -

total price : $32

payment method : cash on delivery

completed

Update Delete

Functional Requirements

Registration	**R1:** The customers can register an account providing their name, email address, and password.
Login	**R2:** Once they register customers can login to the bakery website and access their accounts, while administrators log in directly without registering beforehand.
Browsing the Pastries	**R3:** Customers can browse all the available food items and search by category.
Placing Orders	**R4:** Customers can place orders by choosing their desired food items and the quantity of each item, while also providing the first name and last name of the order's owner, the address (address number, city, and street), phone number, and the payment method.
Viewing & Cancelling Placed Orders	**R5:** Customers can view the orders they have already placed, and in case of an issue, the customers can cancel their order.
Payment Processing	**R6:** The customers will be able to make payments online using their credit cards once they order or pay cash on delivery.
Adding to Cart	**R7:** Customers can add the food items they are planning to purchase to their cart.
Adding to Wishlist	**R8:** The Customers can add their preferred food items to their Wishlist.
Sending messages	**R9:** If the customers intend to interact with the bakery owners and administrators, they can do so through their messages.
Selecting a Rating	**R10:** According to their experience, customers can rate our services (food and delivery).
Writing a Review	**R11:** Customers can write a review describing their overall experience.
Adding Food Items	**R12:** Admins can add and remove food items from the available food items and categorize each added food item.
View, Update, & Delete Orders	**R13:** Admins can view completed and pending orders & update and delete certain orders. Also, they can put a discount on an order, and change the order status.
Managing User Accounts	**R14:** The admins can manage the customer accounts and update their own profiles.
Viewing Messages	**R15:** Admins receive the messages sent by the customers.
Viewing Ratings & Reviews	**R16:** Admins can view the rating that the customers left and the feedback from the customers' experience.

Non-Functional Requirements

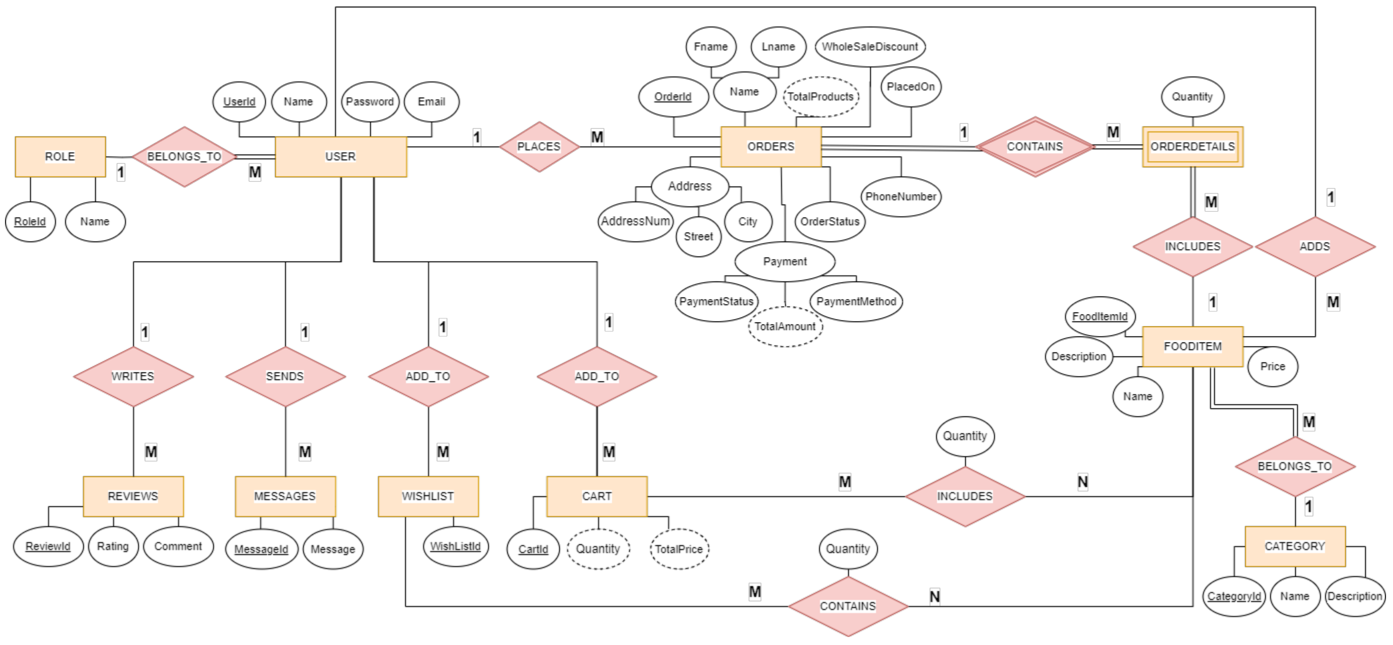
Performance	**R1:** Optimized database queries help in quick system responses to user interactions with minimal latency.
Accurateness	**R2:** Our queries should be able to generate correct results based on the user's requests.
Reliability	**R3:** Ensure data integrity and consistency across the relations guaranteeing the prevention of errors or data loss.
Scalability	**R4:** Our system design handles a massive number of users, orders, messages, reviews, and food items.

Implementation

General Idea of EER Diagram

As a take-off step, we first decided upon the project's entities, their respective attributes, and the relationships among the entities. And then built the ER Diagram to be able to visualize the database structure. We thoroughly chose our entities, that served as real-world objects, Role, User, Orders, Orderdetails, Fooditem, Category, Reviews, Messages, Wishlist, Cart. Also, we made sure to specify the correct cardinality ratios and participation constraints among those entities. In addition, for each entity, we had to specify a set of attributes describing characteristic or property that defines an entity. Some attributes were composite while others were derived and while some entities were strong, others were weak entities dependent on the strong ones. Not to forget, the primary keys that we added for each entity are to ensure that the key constraint and entity integrity constraints are not violated. And the correct choice of the domain for all attributes to guarantee that the domain constraint is not breached. Finally, we made sure that the foreign keys that reference the primary keys in different relations are correctly connected and do not violate the referential integrity constraint.

EER Diagram



Entities

- USER: contains UserId as a primary key, Name, Password, and Email to store user information.
- ROLE: contains RoleId and name. It is used to classify users as customers or admins for website interaction.
- ORDERS: contains OrderId as a primary key, also this entity contains 3 composite attributes like Name (Fname, Lname), Address (AddressNum, Street, City), Payment (PaymentStatus, PaymentMethod, TotalAmount as a derived attribute), OrderStatus, PhoneNumber, placedOn WholeSaleDiscount ,and TotalProducts as a derived attribute to manage order details and information for delivery.
- ORDERDETAILS: a weak entity, it contains Quantity attribute that represents the quantity of each food item in the order.
- FOODITEM: contains FoodItemId as a primary key, Name, Description, and Price to store information of bakery items with their prices.
- CATEGORY: contains CategoryId as a primary key, Name, and Description to categorize bakery items.
- CART: contains CartId as a primary key, Quantity and TotalPrice as derived attributes to track quantities and total prices of items in the shopping cart.
- WISHLIST: which contains WishListId as a primary key to store items for future purchase consideration.
- MESSAGES: contains MessageId as a primary key and Message to record customer messages or inquiries.
- REVIEWS: which contains ReviewId as a primary key, Rating, and comment to collect customer ratings and comments on bakery items.

Customer Relationships

- CUSTOMER PLACES ORDERS: This One-To-Many relationship is used to allow a customer to place an order based on his chosen bakery items. A customer can place many orders, but each order refers to only one customer.
- CUSTOMER ADD_TO CART: This One-To-Many relationship is used to allow customers to add the items they chose from the shop to their cart. A customer can have many carts with his/her name, but each cart belongs to one customer only.
- CUSTOMER ADD_TO WISHLIST: This One-To-Many relationship is used to allow customers to add their favorite items to the Wishlist to be able to order them later. A customer can have many carts with his/her name, but each cart belongs to one customer only.
- CUSTOMER SENDS MESSAGES: This One-To-Many relationship is used to allow customers to send messages to the bakery shop owners. While each customer can send as many messages as they want, each message refers to only one customer.
- CUSTOMER WRITES REVIEWS: This One-To-Many relationship is used to allow customers to write and send reviews to the bakery shop owners. Each customer can send as many reviews as they want but each review refers to only one customer.

- CUSTOMER BELONGS_TO ROLE: This One-To-Many relationship is used to classify users according to RoleId into customers or admins. Where the customers have RoleId = 2. It is an especially important relationship since it will change the way the user has access to specific features of the website or how he/she will interact with the website. A user has one role, and a role can belong to many users.

Admin Relationships

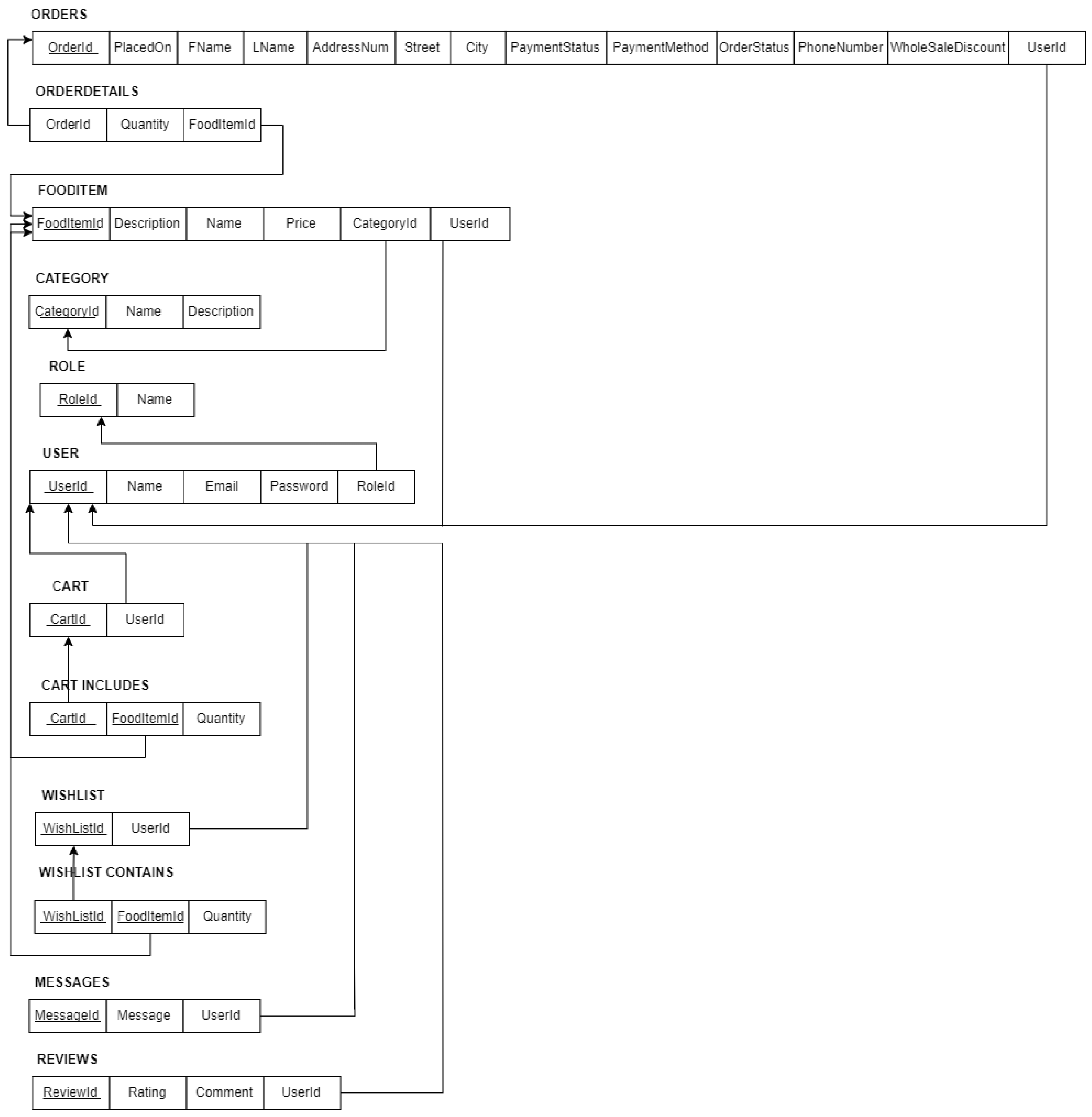
- ADMIN UPDATES ORDERS: This One-To-Many relationship is used to allow the admin to view and update the orders being made by the customers. An admin can view or update many orders, but an order can be viewed by one admin at a time.
- ADMIN ADDS FOODITEM: This One-To-Many relationship is used to allow the admins to add new food items to the website with their names and prices and update the list of old and new bakery items in the shop.
- ADMIN VIEWS MESSAGES: This One-To-Many relationship is used to allow admin to view all the messages being sent through the website by the customers.
- ADMIN VIEWS REVIEWS: This One-To-Many relationship is used to allow admin to view all the reviews being sent through the website by the customers to know their feedback and improve the bakery shop more.
- ADMIN BELONGS_TO ROLE: This One-To-Many relationship is used to classify users according to RoleId into customers or admins. Where the customers have RoleId = 1. It is an important relationship since it will change the way the user has access to specific features of the website or how he/she will interact with the website. A user has one role, and a role can belong to many users.

Other Relationships

- ORDERS CONTAINS ORDERDETAILS: This One-To-Many relationship is a weak relationship since the entity ORDERDETAILS cannot exist without an order existing beforehand. An order contains many Order details (the quantities of the bakery items being placed in that order), but the order details refer to one order.
- ORDERDEATILS INCLUDES FOODITEM: This One-To-Many relationship is used to show the quantity of the food items being placed in the order; this information needed to be present for each order.
- FOODITEM BELONGS_TO CATEGORY: This One-To-Many relationship is used to categorize food items based on Id. Many food items belong to one category and one category can have many food items.
- CART INCLUDES FOODITEM: This Many-To-Many relationship includes the food items and their quantity being added to the cart.
- WISHLIST CONTAINS FOODITEM: This Many-To-Many relationship is used to include the food items and their quantity in the wish list since customers can add their favorite items to the wish list to be able to order them when they want.

Relational Model

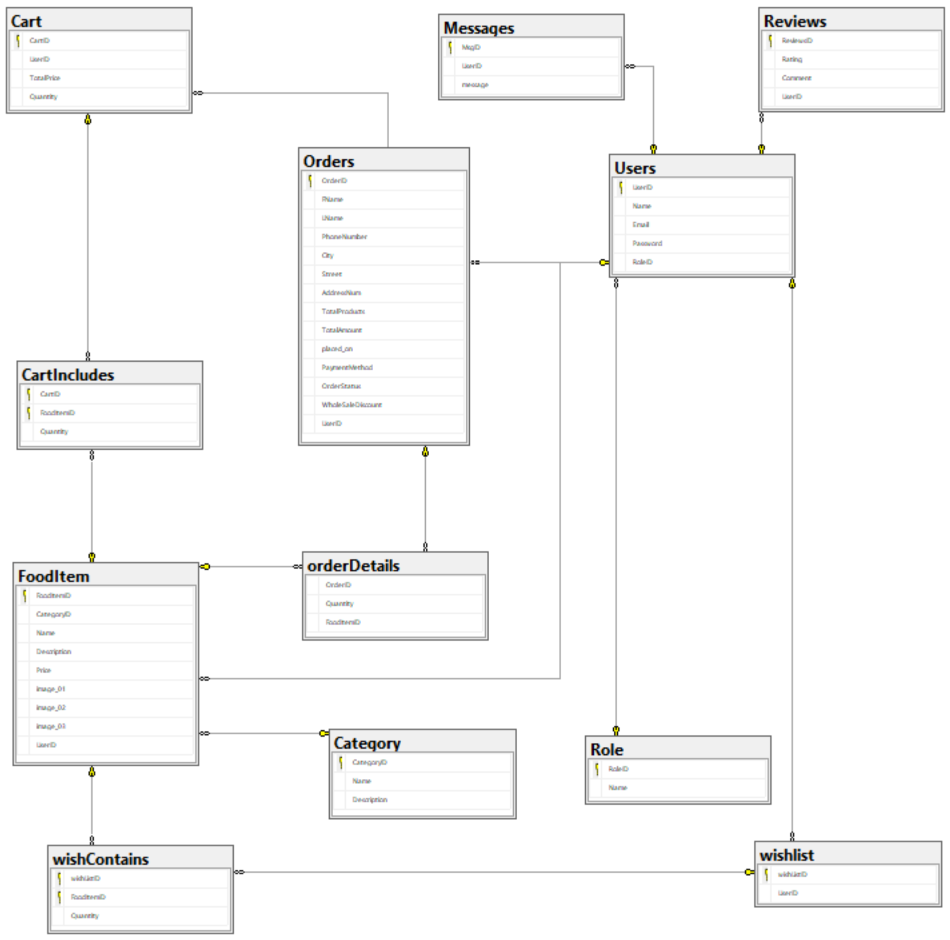
The next step we took was mapping our EER model into a relational model. We drew a schema diagram to clearly define each relation's primary keys, foreign keys, attributes, and relationships; before we started writing the queries to build our database. The relational schema diagram is shown below:



Transforming an Enhanced Entity-Relationship (EER) diagram into a relational model involves several steps to ensure that all entities, attributes, relationships, and constraints are appropriately represented in a set of relational tables and to ensure that no information will be lost. We first identified all entities and their respective attributes, each entity became a table in the relational model, and each attribute became a column in the corresponding table. We started with the strong entities and then the weak ones that depended on another entity for their existence (ex: ORDERDETAILS). Secondly, we translated the relationships between entities into foreign key references in the relational model; for one-to-many relationships we added the foreign key to the 'many' table, while for many-to-many relationships we developed a new table to represent the relationship. This new table included foreign keys referencing the 2 primary keys of the 2 'many' tables as 1 composite primary key of the newly generated table. Thirdly, we made sure to normalize the relational model to eliminate redundancy, ensure data integrity, and maintain consistency within the database. By following these steps, we transformed our EER diagram into a relational model that accurately represented the database of our project.

Database Diagram

After utilizing the DDL (shown below) to write the code of our database on Microsoft SQL (our chosen DBMS) we generated the database diagram to ensure that our tables with their respective attributes and relationships with other tables are all correct. Below is our database diagram:

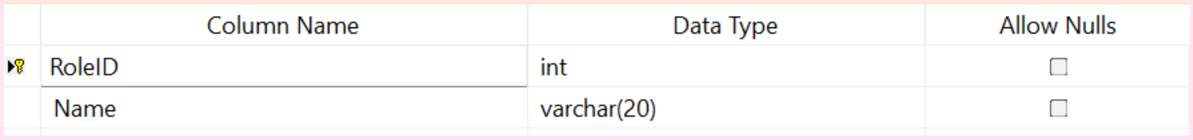


SQL – Data Definition Language

The final step for implementing our database is writing queries. To do so, we used the DDL to define our database for generating our relations. Therefore, for each relation, we specified each attribute's datatype and format (domain) as well as their constraints (ex: NOT NULL), the primary key(s), and the foreign key(s) that reference other relation's primary keys.

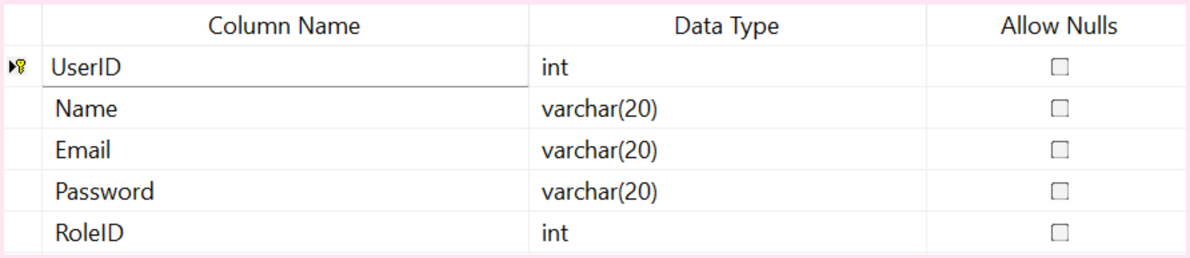
t is said:

```
CREATE TABLE Role (
  RoleID int NOT NULL default(2),
  Name varchar(20)  NOT NULL default('Customer'),
  PRIMARY KEY (RoleID)
);
```



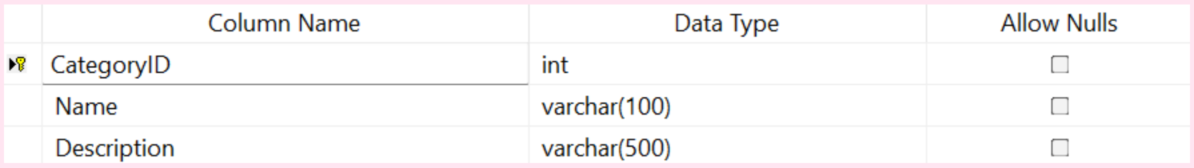
	Column Name	Data Type	Allow Nulls
	RoleID	int	☐
	Name	varchar(20)	☐

```
CREATE TABLE Users(
  UserID int identity(1, 1) NOT NULL,
  Name varchar(20) NOT NULL,
  Email varchar(20) NOT NULL,
  Password varchar(20) NOT NULL,
  RoleID int NOT NULL,
  PRIMARY KEY (UserID),
  FOREIGN KEY (RoleID) REFERENCES Role (RoleID) ON Delete Cascade ON Update Cascade
);
```



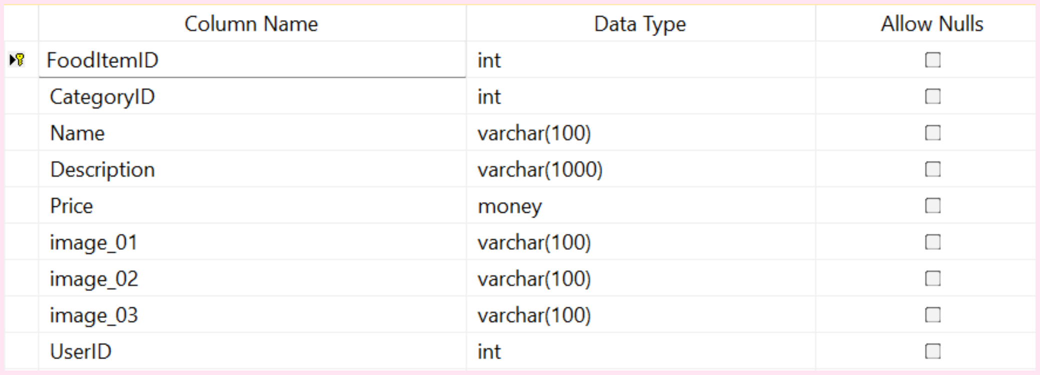
	Column Name	Data Type	Allow Nulls
	UserID	int	☐
	Name	varchar(20)	☐
	Email	varchar(20)	☐
	Password	varchar(20)	☐
	RoleID	int	☐

```
CREATE TABLE Category (
  CategoryID int identity(1, 1)NOT NULL,
  Name varchar(100) NOT NULL,
  Description varchar(500) NOT NULL,
  PRIMARY KEY (CategoryID)
);
```



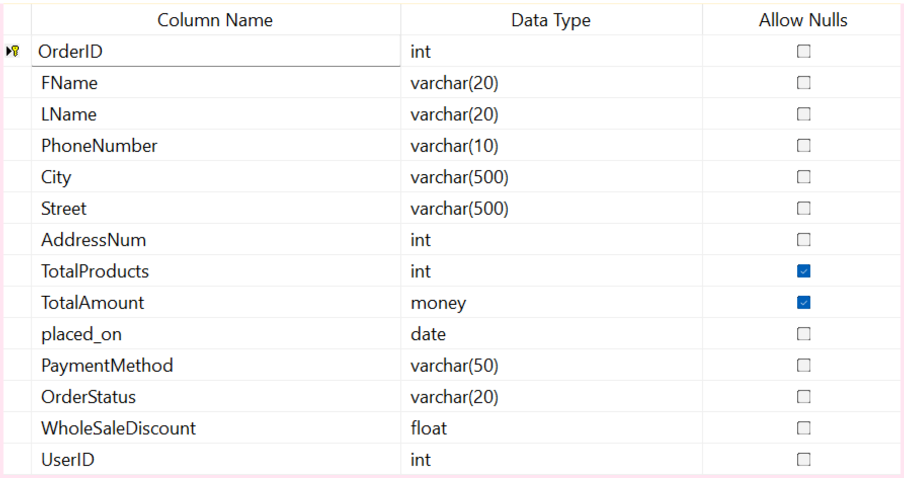
	Column Name	Data Type	Allow Nulls
	CategoryID	int	☐
	Name	varchar(100)	☐
	Description	varchar(500)	☐

```
CREATE TABLE FoodItem (
  FoodItemID int identity(1, 1),
  CategoryID int NOT NULL,
  Name varchar(100) NOT NULL,
  Description varchar(1000) NOT NULL,
  Price money NOT NULL default(0),
  image_01 varchar(100) NOT NULL,
  image_02 varchar(100) NOT NULL,
  image_03 varchar(100) NOT NULL,
  UserID int NOT NULL,
  PRIMARY KEY (FoodItemID),
  FOREIGN KEY (CategoryID) REFERENCES Category (CategoryID) ON Delete Cascade ON Update
Cascade,
  FOREIGN KEY (UserID) REFERENCES Users (UserID) ON Delete Cascade ON Update Cascade
);
```



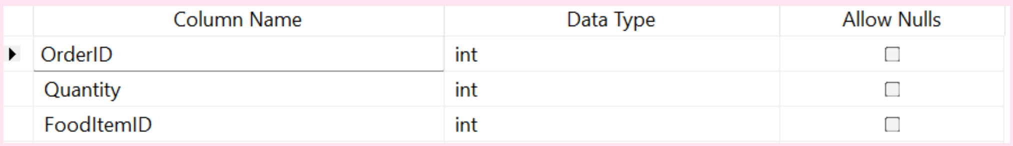
Column Name	Data Type	Allow Nulls
FoodItemID	int	☐
CategoryID	int	☐
Name	varchar(100)	☐
Description	varchar(1000)	☐
Price	money	☐
image_01	varchar(100)	☐
image_02	varchar(100)	☐
image_03	varchar(100)	☐
UserID	int	☐

```
CREATE TABLE Orders (
  OrderID int identity(1, 1),
  FName varchar(20) NOT NULL,
  LName varchar(20) NOT NULL,
  PhoneNumber varchar(10) NOT NULL,
  City varchar(500) NOT NULL,
  Street varchar(500) NOT NULL,
  AddressNum int NOT NULL,
  TotalProducts int default(0),
  TotalAmount money default(0),
  placed_on date NOT NULL,
  PaymentMethod varchar(50) NOT NULL,
  OrderStatus varchar(20) NOT NULL DEFAULT 'pending',
  WholeSaleDiscount float NOT NULL,
  UserID int NOT NULL,
  PRIMARY KEY (OrderID),
  FOREIGN KEY (UserID) REFERENCES Users (UserID) ON Delete Cascade ON Update Cascade
);
```



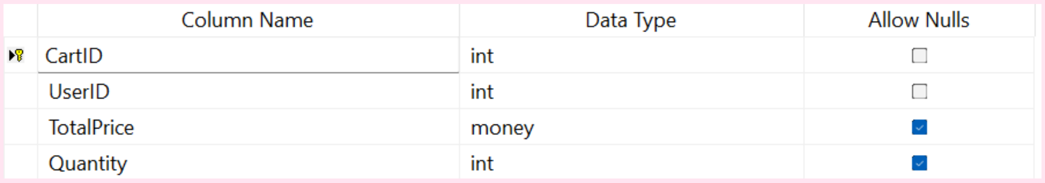
	Column Name	Data Type	Allow Nulls
🔑	OrderID	int	☐
	FName	varchar(20)	☐
	LName	varchar(20)	☐
	PhoneNumber	varchar(10)	☐
	City	varchar(500)	☐
	Street	varchar(500)	☐
	AddressNum	int	☐
	TotalProducts	int	✓
	TotalAmount	money	✓
	placed_on	date	☐
	PaymentMethod	varchar(50)	☐
	OrderStatus	varchar(20)	☐
	WholeSaleDiscount	float	☐
	UserID	int	☐

```sql
CREATE TABLE orderDetails (
  OrderID int NOT NULL,
  Quantity int NOT NULL default(1),
  FoodItemID int NOT NULL,
  FOREIGN KEY (OrderID) REFERENCES Orders (OrderID) ON Delete Cascade ON Update Cascade,
  FOREIGN KEY (FoodItemID) REFERENCES FoodItem (FoodItemID));
```



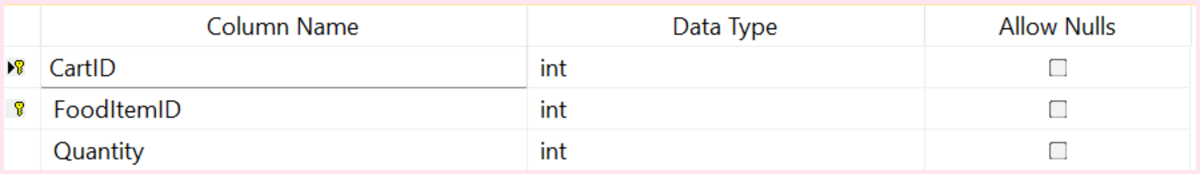
	Column Name	Data Type	Allow Nulls
▶	OrderID	int	☐
	Quantity	int	☐
	FoodItemID	int	☐

```sql
CREATE TABLE Cart (
  CartID int identity(1, 1),
  UserID int NOT NULL,
  TotalPrice money default(0),
  Quantity int default(0),
  PRIMARY KEY (CartID),
  FOREIGN KEY (UserID) REFERENCES Users (UserID) ON Delete Cascade ON Update Cascade);
```



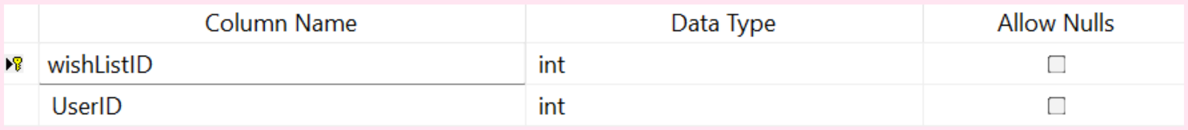
	Column Name	Data Type	Allow Nulls
🔑	CartID	int	☐
	UserID	int	☐
	TotalPrice	money	✓
	Quantity	int	✓

```sql
CREATE TABLE CartIncludes (
  CartID int NOT NULL,
  FoodItemID int NOT NULL,
  Quantity int NOT NULL default(0),
  PRIMARY KEY (CartID, FoodItemID),
  FOREIGN KEY (CartID) REFERENCES Cart (CartID),
  FOREIGN KEY (FoodItemID) REFERENCES FoodItem (FoodItemID)
);
```



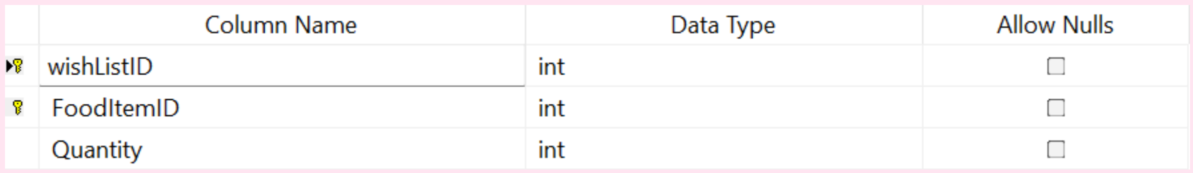
	Column Name	Data Type	Allow Nulls
🔑	CartID	int	☐
🔑	FoodItemID	int	☐
	Quantity	int	☐

```sql
CREATE TABLE wishlist (
  wishListID int identity(1, 1),
  UserID int NOT NULL,
  PRIMARY KEY (wishListID),
  FOREIGN KEY (UserID) REFERENCES Users (UserID) ON Delete Cascade ON Update Cascade
);
```



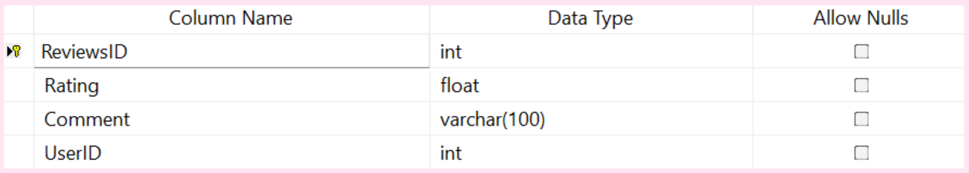
	Column Name	Data Type	Allow Nulls
▶🔑	wishListID	int	☐
	UserID	int	☐

```sql
CREATE TABLE wishContains (
  wishListID int NOT NULL,
  FoodItemID int NOT NULL,
  Quantity int NOT NULL default(0),
  PRIMARY KEY (wishListID, FoodItemID),
  FOREIGN KEY (wishListID) REFERENCES wishlist (wishListID),
  FOREIGN KEY (FoodItemID) REFERENCES FoodItem (FoodItemID)
);
```



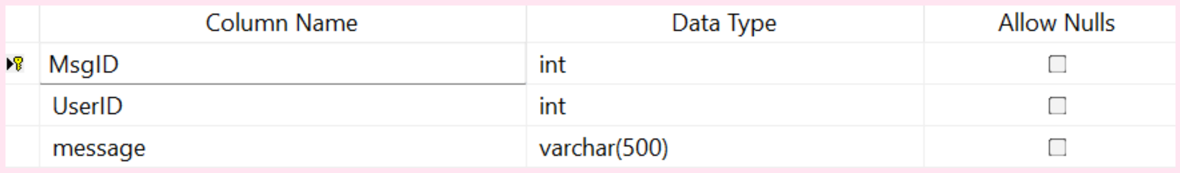
	Column Name	Data Type	Allow Nulls
▶🔑	wishListID	int	☐
🔑	FoodItemID	int	☐
	Quantity	int	☐

```sql
CREATE TABLE Reviews (
  ReviewsID int identity(1, 1),
  Rating float NOT NULL check(Rating >0 and Rating <=5),
  Comment varchar(100) NOT NULL,
  UserID int NOT NULL,
  PRIMARY KEY (ReviewsID),
  FOREIGN KEY (UserID) REFERENCES Users (UserID) ON Delete Cascade ON Update Cascade
);
```



	Column Name	Data Type	Allow Nulls
▶🔑	ReviewsID	int	☐
	Rating	float	☐
	Comment	varchar(100)	☐
	UserID	int	☐

```sql
CREATE TABLE Messages (
  MsgID int identity(1, 1),
  UserID int NOT NULL,
  message varchar(500) NOT NULL,
  PRIMARY KEY (msgID),
  FOREIGN KEY (UserID) REFERENCES Users (UserID) ON Delete Cascade ON Update Cascade
);
```



	Column Name	Data Type	Allow Nulls
▶🔑	MsgID	int	☐
	UserID	int	☐
	message	varchar(500)	☐

List of Information Retrieval Questions

1. Retrieve the name, description, and price of all baked goods available on the website.
2. List all categories of baked goods available.
3. List the number of food items in each category of baked goods.
4. Display all baked goods along with their corresponding categories.
5. Retrieve all baked goods that belong to the 'Cookies' category.
6. Display the category with the least amount of food items.
7. Retrieve all baked goods that belong to the category with the least amount of food items.
8. Search for the available cheesecakes in the bakery.
9. Retrieve all baked goods sorted by their price from cheapest to the most expensive.
10. Get the details of the top 5 food items that are the most ordered by customers.
11. Display the name and price of the food items having the word 'tart' in their name and having a price higher than or equal to $2.
12. Retrieve the name and price of the least expensive baked good.
13. Display all baked goods with a price between $10 and $15.
14. Get the details of the top 5 food items that are the least ordered by customers.
15. Show the most popular category based on the number of orders.
16. Display the least popular category based on the number of orders.
17. Show the average price of baked goods in each category.
18. Display the last 3 food items added by the admin to the list of food items.
19. List all customers who have registered on the website.
20. Display all the administrators of the bakery.
21. Logging in by email 'Lama@gmail.com' and Password = '432109'.
22. Find the number of orders placed by each customer.
23. Display all orders placed by the customers.
24. Show the total revenue generated from completed orders.
25. Show the total revenue generated from pending orders.
26. Show the total revenue generated from orders placed.
27. Display the top 3 most valuable customers with the highest number of orders.
28. Mark all the orders placed before 2024-04-22 as completed.
29. Display all the orders that include more than 4 items.
30. Calculate the number of orders that have food items belonging to the 'Croissant' Category.
31. Display the total revenue generated by the bakery in April.
32. Show the top 3 best-selling baked goods.
33. Retrieve all orders that were placed by admins and not by customers.
34. A customer whose orderID is 5 wants to cancel their order.
35. Retrieve all orders made by the customer Farah.
36. Retrieve all orders placed on the 14th and 21st of April.
37. Retrieve the total number of orders placed from the first day the bakery opened till today.
38. List all orders that are pending and yet to be delivered from the oldest order to the most recent one.
39. List all orders completed and ready for delivery.
40. Display the number of orders placed and the total revenue generated from the order in each city.
41. List all customers who have placed orders from a specific city, 'Beirut'.
42. Display the total revenue generated from orders placed by a customer with UserId = 4.
43. Display the customer and total price of the most expensive order.

44. Find all users who have placed orders containing 'chocolate chip cookies' and provide details of those orders.
45. List all customers who have placed orders worth more than $20.
46. Show the number of orders and total revenue generated on '2024-04-21'.
47. Retrieve the average order price.
48. Show the total quantity of food items over all the order ordered by each customer.
49. Show the number of customers who have registered but have not placed any orders.
50. List all customers who have placed orders above the average order price.
51. Retrieve all customers along with the total number of orders they have placed.
52. Retrieve all customers who have placed more than 2 orders.
53. Retrieve all customers along with their total spending.
54. Retrieve all customers who have spent more than $50.
55. Retrieve the top 10 customers by total spending.
56. List the names of customers, along with their total order price, who have placed orders with a price greater than the average price of all food items.
57. List the names of customers along with their number of orders that have food items from more than one category.
58. Display the customers who have made at least 1 order with a total price above $20 and have submitted a rating of 5 stars and wrote a review.
59. Display all the customers who have not bought in the past month.
60. List all customers who have added items to their cart.
61. Retrieve all customers who have not yet checked out their orders.
62. List all customers who have items in their Wishlist.
63. Display all items with their quantities in the Wishlist of a user with userid = 14.
64. Show all items with their quantities in the cart of a user with userid = 2.
65. Show all users that have placed orders for food items that are also present in their Wishlists.
66. Display the number of customers who have registered but not added any items to their Wishlist.
67. Delete food item named 'double chocolate cookies' from cartId = 8.
68. Display the top 1 food item in the carts and in the Wishlists of all customers.
69. Display the total number of messages sent by each customer.
70. Retrieve all customers who have sent messages to the admin, along with their sent messages.
71. Retrieve all messages sent by customers to the admin.
72. Retrieve all reviews left by customers.
73. Retrieve the average rating of all reviews.
74. Show the details of customers who have given the highest rating.
75. List all customers who have given ratings below 3 stars.
76. List all customers who have registered but have not sent any messages to the admin.
77. Retrieve all customers who have not yet written a review.
78. Retrieve the total number of messages sent to the admin.

SQL – Data Manipulation Language

We used DML to be able to retrieve, insert, update, and delete data within the database. We first inserted values into all our relations so that when we run our queries, we have the data needed to validate their correctness. Then we generated the views of the derived attributes and updated their values in the corresponding relations. After that, we started answering the above-listed questions by writing the queries. We utilized the DML to help search for specific information, modify existing records, manage data, while also allowing users to perform essential tasks and allowing dynamic user interactions. Thus, it was necessary for us to use DML for effective data management and decision-making.

Populating the Database

```sql
-- TotalProducts: Order
CREATE VIEW OrderTotalProductsView AS
Select o.UserID, O.OrderID, Sum(Od.Quantity) AS TotalProducts From Orders As O
Join orderDetails Od On O.OrderID = Od.OrderID
Group By O.UserID, O.OrderID;



UPDATE Orders SET TotalProducts = v.TotalProducts From OrderTotalProductsView As v
Where Orders.UserID = v.UserID AND Orders.OrderID = v.OrderID;



--TotalAmount: Order
CREATE VIEW OrderTotalPriceView AS
Select O.UserID, O.OrderID, Sum((Od.Quantity * F.Price) * (1 - O.WholeSaleDiscount)) As
TotalAmount From Orders AS O
Join orderDetails AS Od ON O.OrderID = Od.OrderID
Join FoodItem AS F ON Od.FoodItemID = F.FoodItemID
Group By O.UserID, O.OrderID, O.WholeSaleDiscount;



UPDATE Orders SET TotalAmount = v.TotalAmount From OrderTotalPriceView As v
Where Orders.UserID = v.UserID AND Orders.OrderID = v.OrderID;

SELECT * FROM Orders



-- Quantity: Cart
CREATE VIEW CartTotalQuantityView AS
Select c.UserID, Sum(Ci.Quantity) AS TotalQuantity From Cart AS C
Join CartIncludes AS Ci ON C.CartID = Ci.CartID
Group By c.UserID;

Update Cart Set Quantity = v.TotalQuantity From CartTotalQuantityView As v
Where Cart.UserID = v.UserID;



-- TotalPrice: Cart
```

```sql
CREATE VIEW CartTotalPriceView AS
Select C.UserID, Sum(Ci.Quantity * F.Price) AS TotalPrice From Cart AS C
Join CartIncludes As Ci ON C.CartID = Ci.CartID
Join FoodItem F On Ci.FoodItemID = F.FoodItemID
Group By C.UserID;

UPDATE Cart SET TotalPrice = v.TotalPrice FROM CartTotalPriceView As v
Where Cart.UserID = v.UserID;




INSERT INTO ROLE (ROLEID, Name) VALUES
(1, 'Admin'),
(2, 'Customer');

INSERT INTO Users (Name, Email, Password, RoleID) VALUES
('Admin', 'candy@gmail.com', '111', 1),
('Amani', 'many@gmail.com', '111', 1),
('Lorance', 'laury@gmail.com', '111', 1);

INSERT INTO Users (Name, Email, Password, RoleID) VALUES
('Lynn', 'Lynn@gmail.com', '654321', 2),
('Jad', 'Jad@gmail.com', '765432', 2),
('Moamen', 'Moamen@gmail.com', '876543', 2),
('Farah', 'Farah@gmail.com', '987654', 2),
('Darine', 'Darine@gmail.com', '098765', 2),
('AbdelAziz', 'AbdelAziz@gmail.com', '109876', 2),
('Ezzeldine', 'Ezzeldine@gmail.com', '210987', 2),
('Adnan', 'Adnan@gmail.com', '212002', 2),
('Dana', 'Dana@gmail.com', '321098', 2),
('Mhmd', 'Mhmd@gmail.com', '648643', 2),
('Sanaa', 'Sanaa@gmail.com' , '978445', 2),
('Lama', 'Lama@gmail.com', '432109', 2),
('Roa', 'Roa@gmail.com', '543210', 2);


INSERT INTO Category (Name, Description) VALUES
('Croissant', 'Butter croissant'),
('Cookies', 'Butter cookies'),
('Donuts', 'doughnut'),
('Bread', 'French & Lebanese bread'),
('Choux Pastry', 'French Choux Pastry'),
('Cakes', 'Cakes & Cupcakes'),
('Cheesecake', 'New York Cheesecakes');
```

```
INSERT INTO FoodItem (Name, description, price, CategoryID, UserID, image_01, image_02,
image_03) VALUES
('croissant', 'Classic butter croissant with soft, flaky layers and a golden-brown crust.',
3, 1, 1,'210228-GFJulesCroissants-R.MoraPhotography-6201-copy-720x720.webp',
'722aef4487a43a74e90ec6822a73a7aa.jpg', 'cf1771e4dfb7357420141f27bde281a8.jpg'),
('donuts', 'Old-fashioned doughnut with a variety of glazes and toppings', 2, 3, 1,
'e0c197059ac98f9ea24c4112a34fe38b.jpg', '8eb272c94b526844adbcace9ef454c27.jpg',
'e0544c2485237c140710f0c3bd28f6b0.jpg'),
('chocolate chip cookies', 'Semisweet chocolate chips mixed in a thick, chewy cookie, soft on
the inside and crunchy on the outside.', 1, 2, 1, 'Classic Toll House Cookies Recipe.jpg',
'3b265e60f4eb5e5551e153612139fdc9.jpg', 'c6d5b665ca6f36e2051bc4302bcf326a.jpg'),
('double chocolate cookies', 'Double Chocolate Chip Cookies that are soft and fudgy and
filled to the brim with chocolate. ', 2, 2, 1, 'b9587b0b4d728c235439d36dd6055441.jpg',
'f23416af24d110af4aa58380544eb92b.jpg', '85571b5f700dab925666032eac637306.jpg'),
('Lotus cookies', 'Biscoff butter cookies with a surprisingly crunchy bite and caramelized
flavor.', 3, 2, 1, 'e8bd50dfa7ee7b57d2e9fe2577d11f21.jpg',
'28caad976ad8ad37117e1e12d6a08bde.jpg', '693d91e0d51358865ea710810fef3fd0.jpg'),
('pain au chocolat croissant', 'Golden French Croissant with a rich dark chocolate filling.',
4, 1, 1, 'b899218bfb6112c3c37e108ec280c052.jpg', '7f9d871e53e6190a1b55c003777aa4e8.jpg',
'388c394cfaa0767924ad989b86255499.jpg'),
('baguette bread', 'French bread, hard and crusty on the outside, with a light and soft
crumb.', 1, 4, 1, 'bb6d61d14c0dd63708a1710aefbbe5c2.jpg',
'351a60075ce9f322bd5c53f7c3e0b33b.jpg', '9dc13c912ae476ea207d122e7b885aea.jpg'),
('Kaak bread', 'A ring shaped savory roll covered with sesame seeds, crispy on the outside
and chewy a bit when eating', 2, 4, 1, '93bdf31d146096cebef7778afca6c2bc.jpg',
'e17ff156547e71c23efa3a060543d801 (1).jpg', 'kaak.jpg'),
('Eclair', 'French choux pastry éclair,  filled with  pastry cream  and topped with chocolate
ganache.', 4, 5, 1, '62e7665cf57796b4753a83c202b8313c.jpg',
'8c4888473d32c7f7680d573f15dde947.jpg', '02308669acc3b2bd856b40f43231380b.jpg'),
('profiteroles eclaire', 'Choux pastries filled with crème pâtissière and coated in chocolate
sauce.\r\n', 2, 5, 1, '068b7215ba921bf31fee7cdad4fa5d89.jpg',
'626b16e0e22a7c17d4c87857ed7d5810.jpg', '53ef17e90ba6ee4ce08d7e91ed54159b.jpg'),
('Vanillia cake', 'Soft dreamy Vanilla Cake topped with vanilla buttercream', 8, 6, 1,
'91e3c174536f4d26f4edd971159f9528.jpg', 'f66206cb63de41ed7e217a423b514de9.jpg',
'a0173549d43278b4dcd177ead9ab6902.jpg'),
('chocolate cake', 'Moist chocolate cake with a rich butter cream filling, chocolate sauce
and Belgian chocolate ganache', 10, 6, 1, 'Salted-Caramel Six-Layer Chocolate Cake.jpg',
'80e21cf0f9e37c578512d29e48c90b32.jpg', 'Salted Caramel Chocolate Cake.jpg'),
('cup cakes', 'Double classic chocolate cupcake with lot of chocolate chips. ', 2, 6, 1,
'3494c2b1572fe41c357aa5672609574e.jpg', '6a822c3814ed6454598c4f4cd3bfebb3.jpg',
'a47e421586166916f39f8d7a10bf88a6.jpg'),
('blueberry cheesecake', 'This blueberry cheesecake starts with a buttery graham cracker
crust, a creamy cheesecake center, and a tangy blueberry swirl.', 12, 7, 1,
'5ce3668ff940d6b6d00ef1094da47f52.jpg', '28dc25fef89eae3f540813f53cf32837.jpg',
'5746b8f83f3106d949b381221164edc1.jpg'),
('lotus cheesecake', 'This Biscoff cheesecake uses Biscoff cookies for the crust, cookie
butter in the cheesecake and topped with a gorgeous cookie butter swirl.', 12, 7, 1,
'c886b39062f72760e6d15971ae8392fe.jpg', 'b8b9f68b71bb6be85584a14b3d199480.jpg',
'c0333a450a3be84f98494daf3da43521.jpg'),
('raspberry cheesecake', 'This raspberry cheesecake starts with a buttery graham cracker
crust, a creamy cheesecake center, and a tangy raspberry swirl.', 12, 7, 1,
'ef18cd8c1fb988b2235bb5085ce0ba57.jpg', '17262b5b92981f22fa0d958a4ce45a74.jpg',
'059f2f570464889d15e46b6c7d6545cb.jpg'),
('Bomboloni donuts', 'Light and delicious Italian doughnuts that are fried, coated in
granulated sugar, and traditionally stuffed with pastry cream.', 4, 3, 1,
'caa048f02b1936e883bca18749de729c.jpg', '365b6dc91d0d11be82cd409ef53ff170.jpg',
'151145e757e3b015e62ccb9fcab34cd9.jpg'),
('mini Fruit tart', 'Fresh fruit slices and a sweet custard filling on top of a bite-sized
cookie shell.', 2, 5, 1, 'fd409ad9246d93ca80e01cd043156057.jpg',
'660ec6685178a2547595c018778fe123.jpg', 'f010cf2f2ccaf635fe14805d86ec073d.jpg'),
('mini chocolate tart', 'mini chocolate Tartlets that start with a buttery shortbread crust
and filled with rich chocolate ganache', 2 ,5, 1, '99eb89ecfebbc929afbc4930321e2108.jpg',
'a5503cf464577718c391e0b6f3ea1990.jpg', 'cd15bea98556095802eedb4b4d6bcc4c.jpg');


INSERT INTO messages(UserID, message) VALUES (4, 'Hello, I have a suggestion for your menu.
Have you considered adding gluten-free options for those with dietary restrictions?'),
(5, 'Good afternoon! I wanted to express my appreciation as the pastries were absolutely
delightful.'),
(7, 'Hi, do you have any special deals for large orders? I am planning a birthday party and
need a lot of sweets.'),
(8, 'Hello! Can you please recommend some popular items from your bakery. I am having trouble
deciding.'),
(9, 'Hey, I wanted to leave a review for the cupcakes I ordered last week. They were
phenomenal!'),
(13, 'Hello, do you offer gift wrapping for orders? I would like to send some pastries as
presents to my friend.')



INSERT INTO wishlist(UserID) SELECT UserID FROM Users;
INSERT INTO wishContains (wishListID, FoodItemID, Quantity) VALUES
(1, 1, 2), (1, 2, 1), (1, 3, 3), (1, 4, 1), (1, 6, 2), (1, 10, 1),
(2, 1, 5), (2, 2, 1), (2, 4, 4),(2, 8, 1), (2, 12, 1),
(3, 1, 1),(3, 2, 1), (3, 3, 1),(3, 4, 1),(3, 5, 1),(3, 6, 1), (3, 7, 1),(3, 8, 1),(3, 9,
1),(3, 10, 1), (3, 11, 1),(3, 12, 1), (3, 13, 1), (3, 14, 1),
(4, 2, 1),(4, 6, 2),(4, 9, 2), (4, 10, 3),(4, 11, 1), (4, 15, 1),
(5, 1, 1), (5, 3, 1),(5, 4, 2), (5, 8, 1),(5, 12, 3),
(6, 2, 1),(6, 5, 2),(6, 9, 5),(6, 10, 2),(6, 11, 1),(6, 12, 3),(6, 13, 1),(6, 14, 1),(6, 15,
1),
(7, 6, 6), (7, 7, 4), (7, 10, 3), (7, 12, 2), (7, 13, 1), (7, 14, 1), (7, 15, 1), (7, 16, 1),
(7, 17, 1), (7, 18, 1), (7, 19, 1),
(8, 4, 5), (8, 7, 3), (8, 11, 2), (8, 15, 5),
(9, 1, 6), (9, 3, 1), (9, 8, 2), (9, 9, 1), (9, 12, 10), (9, 18, 4), (9, 19, 3),
(10, 2, 1), (10, 6, 2), (10, 9, 4), (10, 14, 1), (10, 16, 4), (10, 17, 5), (10, 18, 1),
(11, 4, 1), (11, 5, 2), (11, 10, 3), (11, 11, 1), (11, 15, 4), (11, 18, 2), (11, 19, 3),
(12, 2, 1), (12, 9, 1), (12, 10, 4), (12, 13, 3), (12, 14, 1),
(13, 5, 1), (13, 10, 2), (13, 13, 3), (13, 17, 4), (13, 19, 1),
(14, 2, 1), (14, 8, 2), (14, 14, 5),
(15, 1, 5), (15, 5, 2), (15, 10, 3), (15, 15, 1), (15, 17, 3),
(16, 3, 3), (16, 9, 1), (16, 12, 2), (16, 17, 4), (16, 18, 1);


INSERT INTO Cart (UserID) SELECT UserID FROM Users;
INSERT INTO CartIncludes (CartID, FoodItemID, Quantity) VALUES
(1, 1, 1), (1, 3, 2), (1, 5, 1), (1, 6, 2), (1, 12, 4), (1, 15, 1),
(2, 1, 2), (2, 4, 3),(2, 8, 4),(2, 10, 2), (2, 12, 3),
(3, 1, 1),(3, 2, 1), (3, 3, 1),(3, 4, 1),(3, 5, 1),(3, 6, 1), (3, 7, 1),(3, 8, 1),(3, 9,
1),(3, 10, 1), (3, 11, 1),(3, 12, 1), (3, 13, 1), (3, 14, 1),
(4, 3, 2),(4, 4, 1),(4, 8, 3), (4, 10, 2),(4, 12, 3), (4, 18, 4),
(5, 1, 1), (5, 3, 2), (5, 4, 2), (5, 7, 3), (5, 8, 6),(5, 10, 2) ,(5, 12, 1),(5, 15, 3),
(6, 5, 2),(6, 9, 3),(6, 10, 2), (6, 12, 2),(6, 13, 1), (6, 15, 1),
(7, 1, 2), (7, 3, 1), (7, 7, 3), (7, 11, 3), (7, 14, 2), (7, 15, 1), (7, 17, 1), (7, 19, 2),
(8, 4, 5), (8, 7, 4),(8, 8, 3), (8, 11, 2),(8, 15, 4), (8, 18, 6),
(9, 1, 3), (9, 3, 1), (9, 7, 2), (9, 8, 4), (9, 12, 3), (9, 15, 2), (9, 17, 1),
(10, 2, 4), (10, 5, 2), (10, 10, 1), (10, 14, 1), (10, 16, 2), (10, 19, 3),
(11, 8, 1), (11, 9, 2), (11, 10, 3), (11, 11, 1), (11, 15, 4), (11, 17, 2), (11, 19, 3),
(12, 2, 2), (12, 5, 2), (12, 8, 3), (12, 10, 3), (12, 15, 1),
(13, 3, 1), (13, 7, 2), (13, 10, 3), (13, 15, 2), (13, 19, 1),
(14, 2, 2), (14, 8, 1), (14, 14, 2),
(15, 1, 3), (15,6, 1), (15, 12, 2), (15, 15, 3), (15, 18, 2),
(16, 3, 2), (16, 10, 1), (16, 12, 2), (16, 16, 2), (16, 18, 1);


INSERT INTO
Orders(FName,LName,PhoneNumber,City,Street,AddressNum,placed_on,PaymentMethod,OrderStatus,
WholeSaleDiscount, UserID) VALUES
('Lynn', 'Noureddine', '03111222', 'Beirut', 'Hamra Street', 123, '2024-04-01', 'Credit
Card', 'pending', 0.1, 4),
('Jad', 'Moghrabi', '03444555', 'Saida', 'Al-Najmeh Square', 456, '2024-04-02', 'PayPal',
'pending', 0.15, 5),
('Farah', 'Hamzeh', '03123456', 'Saida', 'Al-Najmeh Square', 789, '2024-04-03', 'Cash',
'pending', 0.05, 7),
('Darine', 'Chames', '03789654', 'Beirut', 'Mar Mikhael Street', 1011, '2024-04-04', 'Credit
Card', 'pending', 0.1, 8),
('AbdelAziz', 'Mustapha', '03632541', 'Tripoli', 'Al-Mina Street', 1213, '2024-04-05',
'Credit Card', 'pending',0, 9),
('Roa', 'Bou Khashfi', '03211366', 'Tripoli', 'Al-Mina Street', 1415, '2024-04-06', 'PayPal',
'pending', 0, 13),
('David', 'Khalil', '03678954', 'Beirut', 'Mar Mikhael Street', 1617, '2024-04-07', 'Credit
Card', 'pending', 0.1, 12),
('Jennifer', 'Nour', '03447885', 'Tyre', 'Al-Qalaa Street', 1819, '2024-04-08', 'Cash',
'pending', 0.15, 12),
('Rawan', 'Wehbi', '70111546', 'Tyre', 'Al-Qalaa Street', 2021, '2024-04-09', 'Credit Card',
'pending', 0.05, 9),
('Sally', 'Hashem', '71421542', 'Beirut', 'Gemmayzeh Street', 2223, '2024-04-10', 'Credit
Card', 'pending', 0.05, 6),
('Sarah', 'Khalil', '03666321', 'Saida', 'Al-Najmeh Square', 2425, '2024-04-11', 'PayPal',
'pending', 0.1, 7),
('Mohammad', 'Khalil', '71845963', 'Beirut', 'Gemmayzeh Street', 123, '2024-04-12', 'Credit
Card', 'pending', 0.05, 8),
('Sara', 'Merhi', '81546987', 'Beirut', 'Gemmayzeh Street', 456, '2024-04-13', 'PayPal',
'pending', 0.15, 10),
('Sandy', 'Khalil', '81032023', 'Tyre', 'Al-Mina Street', 789, '2024-04-14', 'Cash',
'pending', 0.1, 11),
('Sandy', 'El Zein', '81032023', 'Zaarouriyeh', 'Al-Zaarouriyeh Street', 789, '2024-04-14',
'Cash', 'pending', 0.3, 1),
('Sara', 'Khalifeh', '03123962', 'Tyre', 'Al-Mina Street', 1011, '2024-04-15', 'Credit Card',
'pending', 0.05, 10),
('Amani', 'Khalifeh', '81546879', 'Saida ', 'Al-Zeitoun Street', 1213, '2024-04-16', 'Credit
Card', 'pending', 0.1, 4),
('Angelina', 'Hashem', '81555945', 'Beirut', 'Bliss Street', 1415, '2024-04-17', 'PayPal',
'pending', 0.05, 6),
('Loren', 'Hashem', '81050540', 'Beirut', 'Hamra Street', 1617, '2024-04-18', 'Credit Card',
'pending', 0.15, 5),
('Kate', 'Khalifeh', '81045500', 'Beirut', 'Hamra Street', 1819, '2024-04-19', 'Cash',
'pending', 0, 7),
('Zainab', 'Merhi', '71724241', 'Saida', 'Al-Najmeh Square', 2021, '2024-04-20', 'Credit
Card', 'pending', 0, 8),
('Lama', 'Affara', '81458851', 'Saida', 'Al-Najmeh Square', 2021, '2024-04-20', 'Credit
Card', 'pending', 0.72, 15),
('Roaa', 'Abou Khachfeh', '03147906', 'Barja', 'Barja Street', 2021, '2024-04-21', 'Credit
Card', 'pending', 0.82, 16),
('Sarah', 'Hashem', '03724241', 'Saida', 'Al-Najmeh Square', 2223, '2024-04-21', 'Credit
Card', 'pending', 0 , 9),
('Mohammad', 'Khalil', '03613002', 'Saida', 'Al-Najmeh Square', 2425, '2024-04-22', 'PayPal',
'pending', 0, 10),
('Diana', 'Said', '71819151', 'Tyre', 'Al-Mina Street', 123, '2024-04-23', 'Credit Card',
'pending', 0, 11),
('Sara', 'Zein', '71445664', 'Beirut', 'Hamra Street', 456, '2024-04-24', 'PayPal',
'pending', 0.5, 12),
('Roa', 'Zein', '032223223', 'Saida', 'Al-Najmeh Square', 789, '2024-04-25', 'Cash',
'pending', 0.15, 13),
('Ahmad', 'Khalil', '70544220', 'Saida', 'Al-Najmeh Square', 1011, '2024-04-26', 'Credit
Card', 'pending', 0.01, 5),
('Shaza', 'Mourad', '71645645', 'Beirut', 'Mar Mikhael Street', 1213, '2024-04-27', 'Credit
Card', 'pending', 0, 8),
('Khalil', 'Khalifeh', '71000333', 'Beirut', 'Mar Mikhael Street', 1415, '2024-04-28',
'PayPal', 'pending',0.15, 10),
('Hasan', 'Merhi', '70520520', 'Beirut', 'Mar Mikhael Street', 1617, '2024-04-29', 'Credit
Card', 'pending',0.20, 12);


INSERT INTO orderDetails(OrderID,Quantity,FoodItemID) VALUES
-- For OrderID 1
(1, 2, 1),
(1, 3, 2),
(1, 1, 3),
-- For OrderID 2
(2, 1, 4),
(2, 2, 5),
(2, 1, 6),
-- For OrderID 3
(3, 3, 7),
(3, 2, 8),
-- For OrderID 4
(4, 1, 9),
(4, 1, 10),
-- For OrderID 5
(5, 2, 11),
(5, 1, 12),
-- For OrderID 6
(6, 3, 13),
(6, 1, 14),
-- For OrderID 7
(7, 2, 15),
(7, 1, 16),
-- For OrderID 8
(8, 1, 17),
(8, 2, 18),
-- For OrderID 9
(9, 3, 19),
-- For OrderID 10
(10, 2, 1),
(10, 3, 2),
(10, 1, 3),
-- For OrderID 11
(11, 1, 4),
(11, 2, 5),
(11, 1, 6),
-- For OrderID 12
(12, 3, 7),
(12, 2, 8),
-- For OrderID 13
(13, 1, 9),
(13, 1, 10),
-- For OrderID 14
(14, 2, 11),
(14, 1, 12),
-- For OrderID 15
(15, 3, 13),
(15, 1, 14),
-- For OrderID 16
(16, 2, 15),
(16, 1, 16),
```

```sql
-- For OrderID 17
(17, 1, 17),
(17, 2, 18),
-- For OrderID 18
(18, 3, 19),
-- For OrderID 19
(19, 2, 1),
(19, 1, 2),
-- For OrderID 20
(20, 1, 3),
(20, 3, 4),
-- For OrderID 21
(21, 2, 5),
(21, 1, 6),
-- For OrderID 22
(22, 3, 7),
(22, 1, 8),
-- For OrderID 23
(23, 1, 9),
(23, 2, 10),
-- For OrderID 24
(24, 3, 11),
(24, 1, 12),
-- For OrderID 25
(25, 2, 13),
(25, 1, 14),
-- For OrderID 26
(26, 1, 15),
(26, 2, 16),
-- For OrderID 27
(27, 3, 17),
(27, 1, 18),
-- For OrderID 28
(28, 2, 19),
(28, 1, 2),
-- For OrderID 29
(29, 3, 1),
(29, 1, 2),
-- For OrderID 30
(30, 3, 1),
(30, 1, 2),
(30, 2, 13),
(30, 1, 14),
-- For OrderID 31
(31, 3, 11),
(31, 5, 12),
-- For OrderID 32
(32, 4, 17),
(32, 2, 18);
```

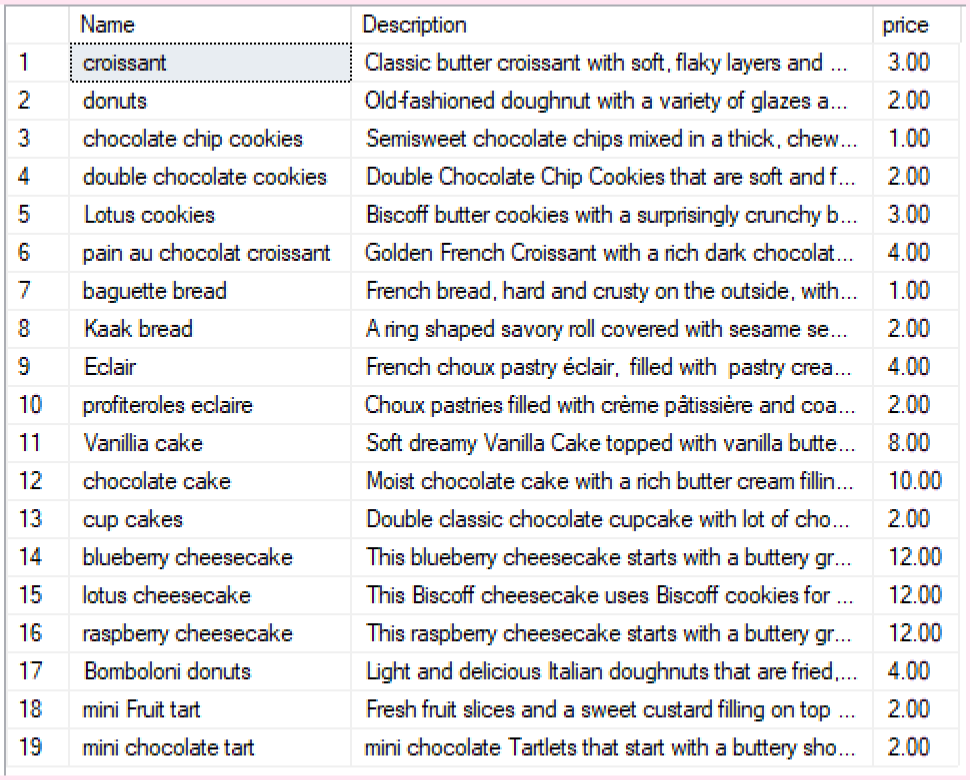
```sql
INSERT INTO Reviews(Rating, Comment, UserID) Values
(5, 'Absolutely loved the bakery items!', 12),
(5, 'Best bakery in town!', 13),
(4.5, 'Quick service and amazing taste!', 10),
(4.75, 'Wow! Super cute place with amazing pastries. The staff was also super nice and 
welcoming.', 8),
(5, 'The food is excellent, especially the glazed donuts. Also the delivery is fast', 7),
(4.75, 'Just got back from Europe and missing fresh baked French bread? This is the place to 
go', 5),
(5, 'What a great bakery! One of my favorites. Amazing selection of baked goods', 9),
(4.5, 'Great bakery.If you love sweet treats, you need to visit', 13);
```

Implementing operations & Displaying Data

After choosing a list of the most valuable, beneficial, and fun 78 questions for data retrieval, we then translated these questions into the proper SQL queries that answer them. We made sure the queries include insert, update, delete, select, diverse query operations, aliasing, generation of views, arithmetic operators, ordering, nested queries, comparison operators, exists functions, aggregate functions, grouping by, and diverse types of joins for comprehensive data extraction. Executing these queries will highlight the effectiveness of our database and will help in managing the bakery shop more efficiently. By that our bakery database is ready to make a worthy contribution to the online shop.

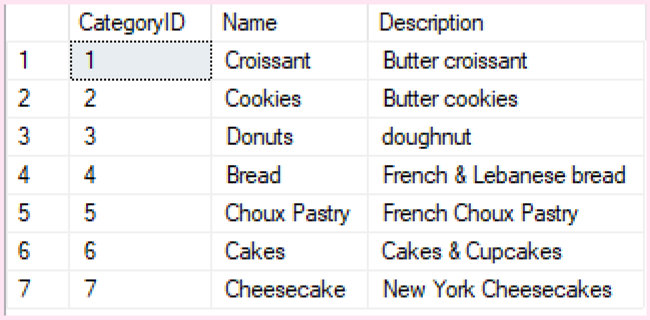
```sql
-- FOOD ITEM AND CATEGORY RELATED

--1. Retrieve the name, description, and price of all baked goods available on the website.
Select F.Name, F.Description, F.price From FoodItem as F;
```



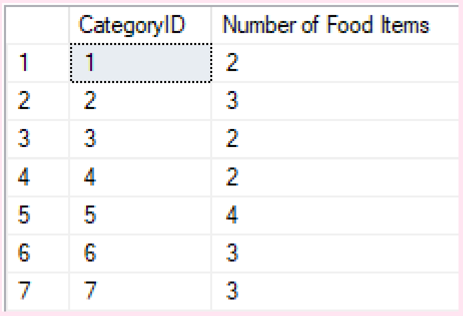
	Name	Description	price
1	croissant	Classic butter croissant with soft, flaky layers and ...	3.00
2	donuts	Old-fashioned doughnut with a variety of glazes a...	2.00
3	chocolate chip cookies	Semisweet chocolate chips mixed in a thick, chew...	1.00
4	double chocolate cookies	Double Chocolate Chip Cookies that are soft and f...	2.00
5	Lotus cookies	Biscoff butter cookies with a surprisingly crunchy b...	3.00
6	pain au chocolat croissant	Golden French Croissant with a rich dark chocolat...	4.00
7	baguette bread	French bread, hard and crusty on the outside, with...	1.00
8	Kaak bread	A ring shaped savory roll covered with sesame se...	2.00
9	Eclair	French choux pastry éclair, filled with pastry crea...	4.00
10	profiteroles eclaire	Choux pastries filled with crème pâtissière and coa...	2.00
11	Vanillia cake	Soft dreamy Vanilla Cake topped with vanilla butte...	8.00
12	chocolate cake	Moist chocolate cake with a rich butter cream fillin...	10.00
13	cup cakes	Double classic chocolate cupcake with lot of cho...	2.00
14	blueberry cheesecake	This blueberry cheesecake starts with a buttery gr...	12.00
15	lotus cheesecake	This Biscoff cheesecake uses Biscoff cookies for ...	12.00
16	raspberry cheesecake	This raspberry cheesecake starts with a buttery gr...	12.00
17	Bomboloni donuts	Light and delicious Italian doughnuts that are fried,...	4.00
18	mini Fruit tart	Fresh fruit slices and a sweet custard filling on top ...	2.00
19	mini chocolate tart	mini chocolate Tartlets that start with a buttery sho...	2.00

```sql
--2. List all categories of baked goods available.
Select * From Category;
```



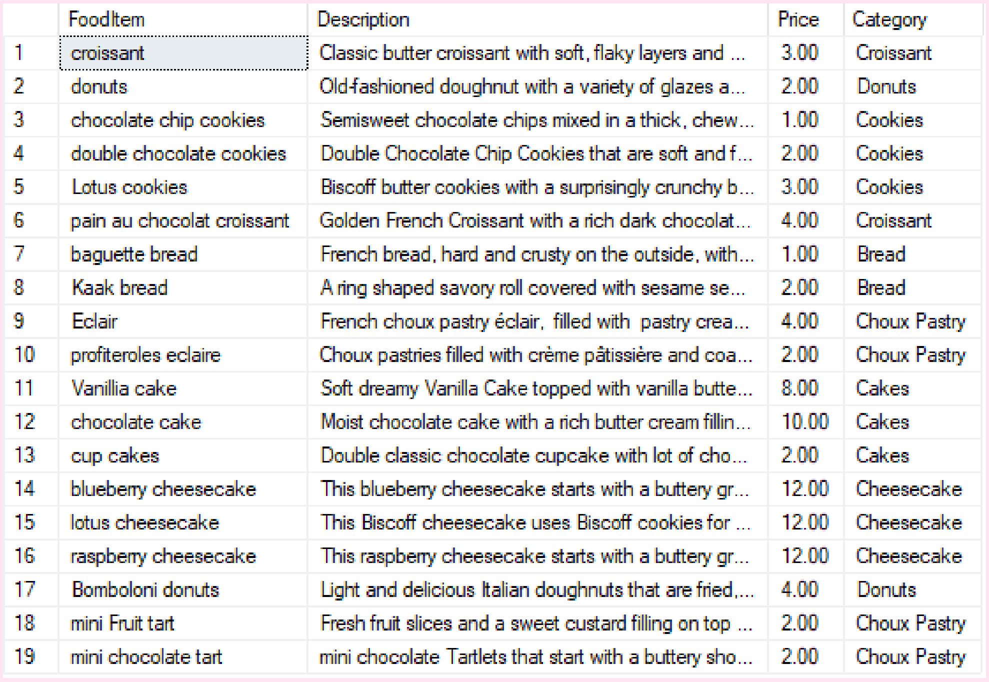
	CategoryID	Name	Description
1	1	Croissant	Butter croissant
2	2	Cookies	Butter cookies
3	3	Donuts	doughnut
4	4	Bread	French & Lebanese bread
5	5	Choux Pastry	French Choux Pastry
6	6	Cakes	Cakes & Cupcakes
7	7	Cheesecake	New York Cheesecakes

```sql
--3. List the number of food items in each category of baked goods.
Select C.CategoryID, count(F.FoodItemID) as 'Number of Food Items' From Category as C
Right Join FoodItem as F On F.CategoryID = C.CategoryID
Group By C.CategoryID;
```



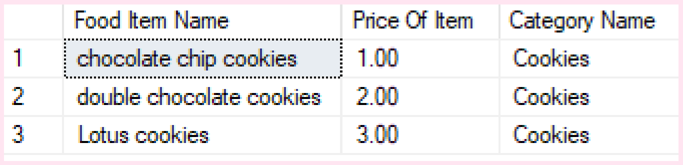
	CategoryID	Number of Food Items
1	1	2
2	2	3
3	3	2
4	4	2
5	5	4
6	6	3
7	7	3

```sql
--4. Display all baked goods along with their corresponding categories.
Select F.Name AS FoodItem, F.Description, F.Price, C.Name AS 'Category'
From FoodItem as F
Left Outer Join Category C On F.CategoryID = C.CategoryID;
```



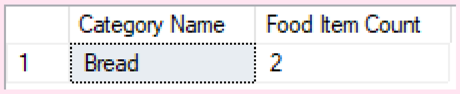
	FoodItem	Description	Price	Category
1	croissant	Classic butter croissant with soft, flaky layers and ...	3.00	Croissant
2	donuts	Old-fashioned doughnut with a variety of glazes a...	2.00	Donuts
3	chocolate chip cookies	Semisweet chocolate chips mixed in a thick, chew...	1.00	Cookies
4	double chocolate cookies	Double Chocolate Chip Cookies that are soft and f...	2.00	Cookies
5	Lotus cookies	Biscoff butter cookies with a surprisingly crunchy b...	3.00	Cookies
6	pain au chocolat croissant	Golden French Croissant with a rich dark chocolat...	4.00	Croissant
7	baguette bread	French bread, hard and crusty on the outside, with...	1.00	Bread
8	Kaak bread	A ring shaped savory roll covered with sesame se...	2.00	Bread
9	Eclair	French choux pastry éclair, filled with pastry crea...	4.00	Choux Pastry
10	profiteroles eclaire	Choux pastries filled with crème pâtissière and coa...	2.00	Choux Pastry
11	Vanillia cake	Soft dreamy Vanilla Cake topped with vanilla butte...	8.00	Cakes
12	chocolate cake	Moist chocolate cake with a rich butter cream fillin...	10.00	Cakes
13	cup cakes	Double classic chocolate cupcake with lot of cho...	2.00	Cakes
14	blueberry cheesecake	This blueberry cheesecake starts with a buttery gr...	12.00	Cheesecake
15	lotus cheesecake	This Biscoff cheesecake uses Biscoff cookies for ...	12.00	Cheesecake
16	raspberry cheesecake	This raspberry cheesecake starts with a buttery gr...	12.00	Cheesecake
17	Bomboloni donuts	Light and delicious Italian doughnuts that are fried,...	4.00	Donuts
18	mini Fruit tart	Fresh fruit slices and a sweet custard filling on top ...	2.00	Choux Pastry
19	mini chocolate tart	mini chocolate Tartlets that start with a buttery sho...	2.00	Choux Pastry

```sql
--5. Retrieve all baked goods that belong to the 'Cookies' category
Select F.Name As 'Food Item Name', F.Price As 'Price Of Item', C.Name as 'Category Name' From
FoodItem as F
Inner Join Category as C On F.CategoryID = C.CategoryID
WHERE C.Name = 'Cookies';
```



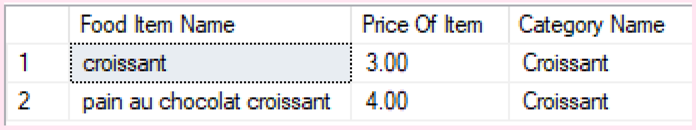
	Food Item Name	Price Of Item	Category Name
1	chocolate chip cookies	1.00	Cookies
2	double chocolate cookies	2.00	Cookies
3	Lotus cookies	3.00	Cookies

```sql
--6. Display the category with the least amount of food items
Select Top 1 C.Name As 'Category Name', Count(*) As 'Food Item Count' FROM Category As C
Left Join FoodItem AS F On C.CategoryID = F.CategoryID
Group By C.Name Order By Count(*) ASC;
```



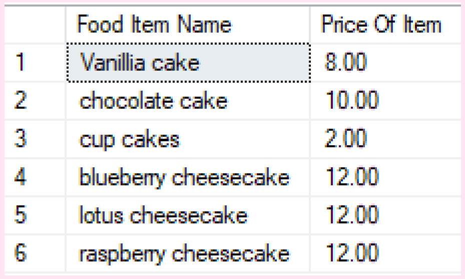
	Category Name	Food Item Count
1	Bread	2

```sql
--7. Retrieve all baked goods that belong to the category with the least amount of food items
Select F.Name As 'Food Item Name', F.Price As 'Price Of Item', C.Name as 'Category Name' From
FoodItem As F, Category as C
Left Outer Join (Select Top 1 CategoryID, COUNT(*) As 'FoodItemCount' From FoodItem
Group By CategoryID Order By FoodItemCount ASC) As LeastFoodCategory
On C.CategoryID = LeastFoodCategory.CategoryID
WHERE F.CategoryID = LeastFoodCategory.CategoryID;
```



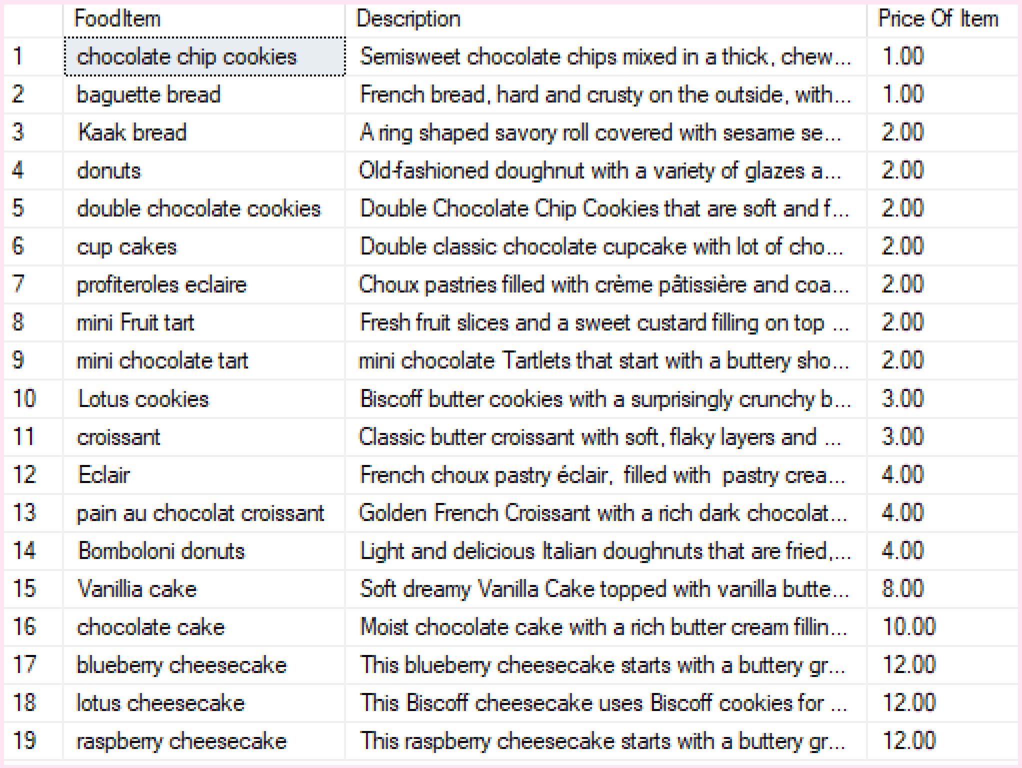
	Food Item Name	Price Of Item	Category Name
1	croissant	3.00	Croissant
2	pain au chocolat croissant	4.00	Croissant

```sql
--8. Search for the available cheesecakes in the bakery
Select F.Name As 'Food Item Name', F.Price As 'Price Of Item' From FoodItem as F
WHERE F.Name LIKE '%cake%';
```



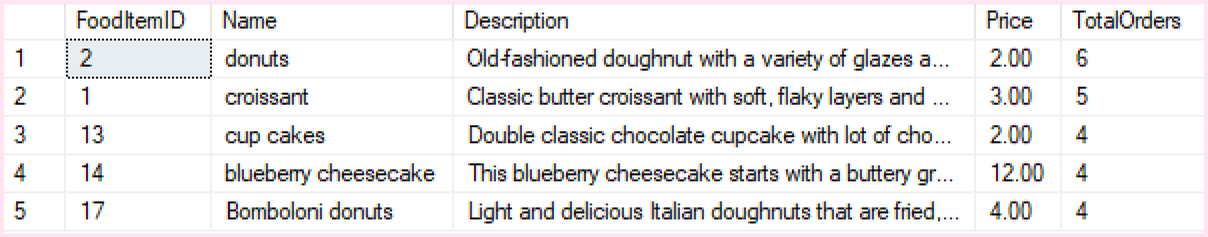
	Food Item Name	Price Of Item
1	Vanillia cake	8.00
2	chocolate cake	10.00
3	cup cakes	2.00
4	blueberry cheesecake	12.00
5	lotus cheesecake	12.00
6	raspberry cheesecake	12.00

```sql
--9. Retrieve all baked goods sorted by their price from cheapest to the most expensive.
Select F.Name AS FoodItem, F.Description, F.Price As 'Price Of Item' From FoodItem as F
Order By Price;
```



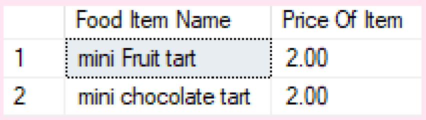
	FoodItem	Description	Price Of Item
1	chocolate chip cookies	Semisweet chocolate chips mixed in a thick, chew...	1.00
2	baguette bread	French bread, hard and crusty on the outside, with...	1.00
3	Kaak bread	A ring shaped savory roll covered with sesame se...	2.00
4	donuts	Old-fashioned doughnut with a variety of glazes a...	2.00
5	double chocolate cookies	Double Chocolate Chip Cookies that are soft and f...	2.00
6	cup cakes	Double classic chocolate cupcake with lot of cho...	2.00
7	profiteroles eclaire	Choux pastries filled with crème pâtissière and coa...	2.00
8	mini Fruit tart	Fresh fruit slices and a sweet custard filling on top ...	2.00
9	mini chocolate tart	mini chocolate Tartlets that start with a buttery sho...	2.00
10	Lotus cookies	Biscoff butter cookies with a surprisingly crunchy b...	3.00
11	croissant	Classic butter croissant with soft, flaky layers and ...	3.00
12	Eclair	French choux pastry éclair, filled with pastry crea...	4.00
13	pain au chocolat croissant	Golden French Croissant with a rich dark chocolat...	4.00
14	Bomboloni donuts	Light and delicious Italian doughnuts that are fried,...	4.00
15	Vanillia cake	Soft dreamy Vanilla Cake topped with vanilla butte...	8.00
16	chocolate cake	Moist chocolate cake with a rich butter cream fillin...	10.00
17	blueberry cheesecake	This blueberry cheesecake starts with a buttery gr...	12.00
18	lotus cheesecake	This Biscoff cheesecake uses Biscoff cookies for ...	12.00
19	raspberry cheesecake	This raspberry cheesecake starts with a buttery gr...	12.00

```sql
--10. Get the details of the top 5 food items that are the most ordered by customers
-- Note: based on the number of times they have been ordered
Select Top 5 F.FoodItemID, F.Name, F.Description, F.Price, COUNT(Od.FoodItemID) AS
'TotalOrders'
From FoodItem F
Join orderDetails as Od On F.FoodItemID = Od.FoodItemID
Group By F.FoodItemID, F.Name, F.Description, F.Price
Order By TotalOrders DESC;
```



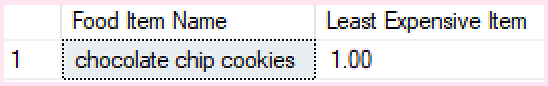
	FoodItemID	Name	Description	Price	TotalOrders
1	2	donuts	Old-fashioned doughnut with a variety of glazes a...	2.00	6
2	1	croissant	Classic butter croissant with soft, flaky layers and ...	3.00	5
3	13	cup cakes	Double classic chocolate cupcake with lot of cho...	2.00	4
4	14	blueberry cheesecake	This blueberry cheesecake starts with a buttery gr...	12.00	4
5	17	Bomboloni donuts	Light and delicious Italian doughnuts that are fried,...	4.00	4

```sql
--11. Display the name and price of the food items having the word 'tart' in their name and
having a price higher than or equal to $2.
Select F.Name As 'Food Item Name', F.Price As 'Price Of Item' From FoodItem as F
Where F.Name LIKE '%tart%' and F.Price >= 2;
```



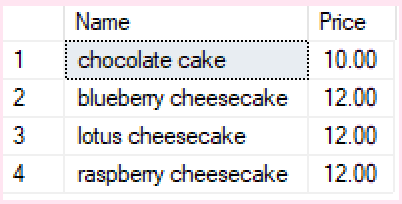
	Food Item Name	Price Of Item
1	mini Fruit tart	2.00
2	mini chocolate tart	2.00

```sql
--12. Retrieve the name and price of the least expensive baked good.
Select Top 1 F.Name As 'Food Item Name', F.Price As 'Least Expensive Item'
From FoodItem As F
Order by F.Price;
```



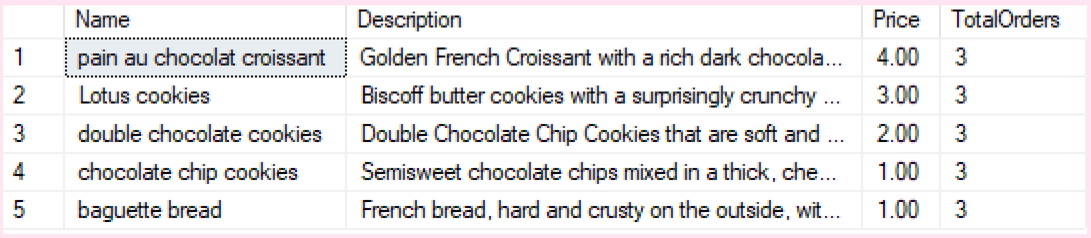
	Food Item Name	Least Expensive Item
1	chocolate chip cookies	1.00

```sql
--13. Display all baked goods with a price between $10 and $15.
Select F.Name, F.Price From FoodItem As F
Where Price BETWEEN 10 AND 15;
```



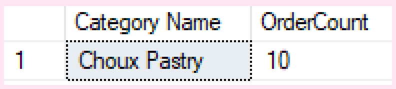
	Name	Price
1	chocolate cake	10.00
2	blueberry cheesecake	12.00
3	lotus cheesecake	12.00
4	raspberry cheesecake	12.00

```sql
--14. Get the details of the top 5 food items that are the least ordered by customers
Select Top 5 F.Name, F.Description, F.Price, COUNT(Od.FoodItemID) AS 'TotalOrders'
From FoodItem F
Left Join orderDetails as Od On F.FoodItemID = Od.FoodItemID
Group By F.FoodItemID, F.Name, F.Description, F.Price
Order By TotalOrders ASC;
```



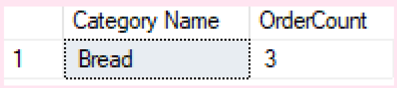
	Name	Description	Price	TotalOrders
1	pain au chocolat croissant	Golden French Croissant with a rich dark chocola...	4.00	3
2	Lotus cookies	Biscoff butter cookies with a surprisingly crunchy ...	3.00	3
3	double chocolate cookies	Double Chocolate Chip Cookies that are soft and ...	2.00	3
4	chocolate chip cookies	Semisweet chocolate chips mixed in a thick, che...	1.00	3
5	baguette bread	French bread, hard and crusty on the outside, wit...	1.00	3

```sql
--15. Show the most popular category based on the number of orders.
Select Top 1 C.Name As 'Category Name',Count(DISTINCT O.OrderID) As 'OrderCount' From
Category As C
Left Join FoodItem As F On C.CategoryID = F.CategoryID
Left Join orderDetails As Od On F.FoodItemID= Od.FoodItemID
Left Join Orders As O On Od.OrderID=O.OrderID
Group By C.Name Order By OrderCount DESC;
```



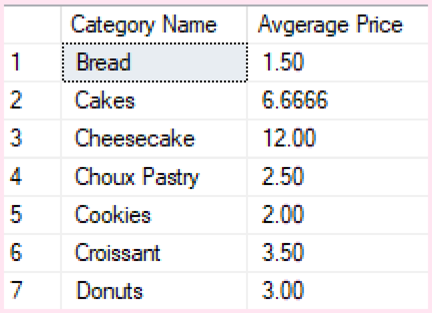
	Category Name	OrderCount
1	Choux Pastry	10

```sql
--16. Display the least popular category based on the number of orders.
Select Top 1 C.Name As 'Category Name',Count(DISTINCT O.OrderID) As 'OrderCount' From
Category As C
Left Join FoodItem As F On C.CategoryID = F.CategoryID
Left Join orderDetails As Od On F.FoodItemID= Od.FoodItemID
Left Join Orders As O On Od.OrderID=O.OrderID
Group By C.Name Order By OrderCount ASC;
```



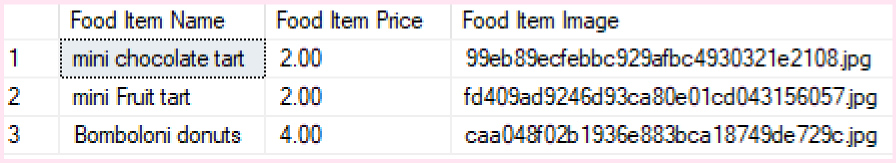
	Category Name	OrderCount
1	Bread	3

```sql
--17. Show the average price of baked goods in each category.
Select C.Name As 'Category Name',AVG(F.Price) As 'Average Price' From Category As C
Join FoodItem F On C.CategoryID =F.CategoryID
Group By C.Name;
```



	Category Name	Avgerage Price
1	Bread	1.50
2	Cakes	6.6666
3	Cheesecake	12.00
4	Choux Pastry	2.50
5	Cookies	2.00
6	Croissant	3.50
7	Donuts	3.00

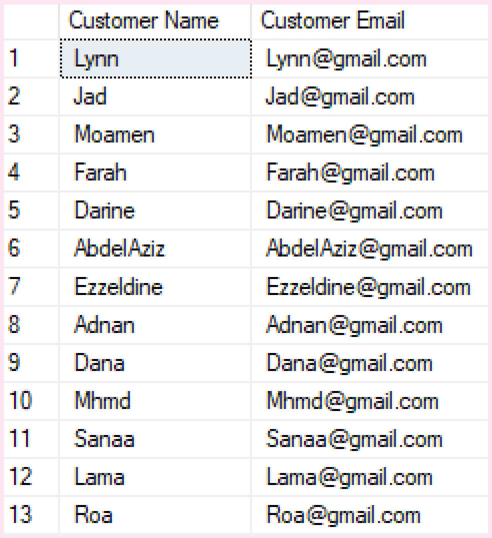
```sql
--18. Display the last 3 food items added by the admin to the list of food items
SELECT TOP 3 F.Name As 'Food Item Name', F.Price As 'Food Item Price', F.image_01 As 'Food
Item Image'
FROM FoodItem As F ORDER BY FoodItemID DESC;
```



	Food Item Name	Food Item Price	Food Item Image
1	mini chocolate tart	2.00	99eb89ecfebbc929afbc4930321e2108.jpg
2	mini Fruit tart	2.00	fd409ad9246d93ca80e01cd043156057.jpg
3	Bomboloni donuts	4.00	caa048f02b1936e883bca18749de729c.jpg

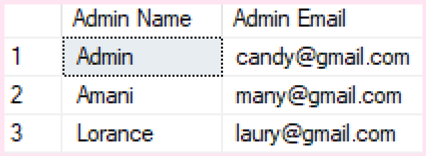
```sql
-- CUSTOMER AND ORDERS RELATED

--19. List all customers who have registered on the website.
Select Name As 'Customer Name', Email As 'Customer Email' From Users
Where RoleID = 2;
```



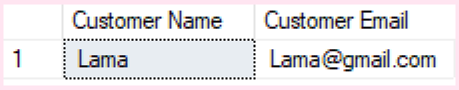
	Customer Name	Customer Email
1	Lynn	Lynn@gmail.com
2	Jad	Jad@gmail.com
3	Moamen	Moamen@gmail.com
4	Farah	Farah@gmail.com
5	Darine	Darine@gmail.com
6	AbdelAziz	AbdelAziz@gmail.com
7	Ezzeldine	Ezzeldine@gmail.com
8	Adnan	Adnan@gmail.com
9	Dana	Dana@gmail.com
10	Mhmd	Mhmd@gmail.com
11	Sanaa	Sanaa@gmail.com
12	Lama	Lama@gmail.com
13	Roa	Roa@gmail.com

```sql
--20. Display all the administrators of the bakery.
Select Name As 'Admin Name', Email As 'Admin Email' From Users
Where RoleID = 1;
```



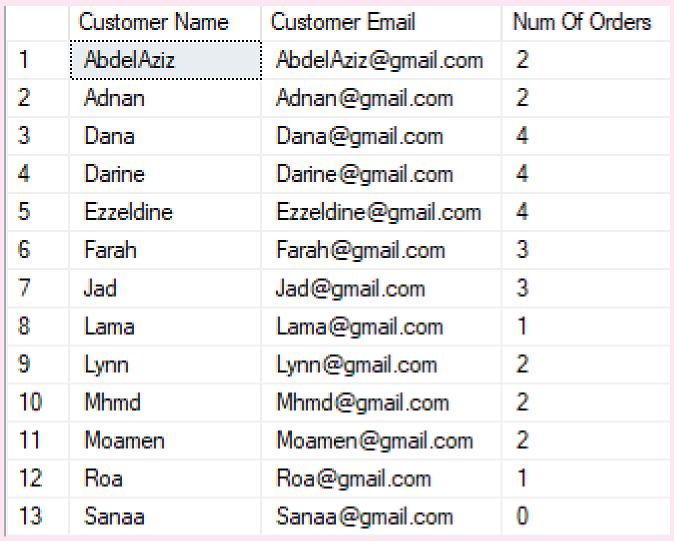
	Admin Name	Admin Email
1	Admin	candy@gmail.com
2	Amani	many@gmail.com
3	Lorance	laury@gmail.com

```sql
--21. Logging in With email 'Lama@gmail.com' and Password = '432109'
Select Name As 'Customer Name', Email As 'Customer Email' From Users
Where RoleID = 2 AND Email = 'Lama@gmail.com' AND Password = '432109';
```



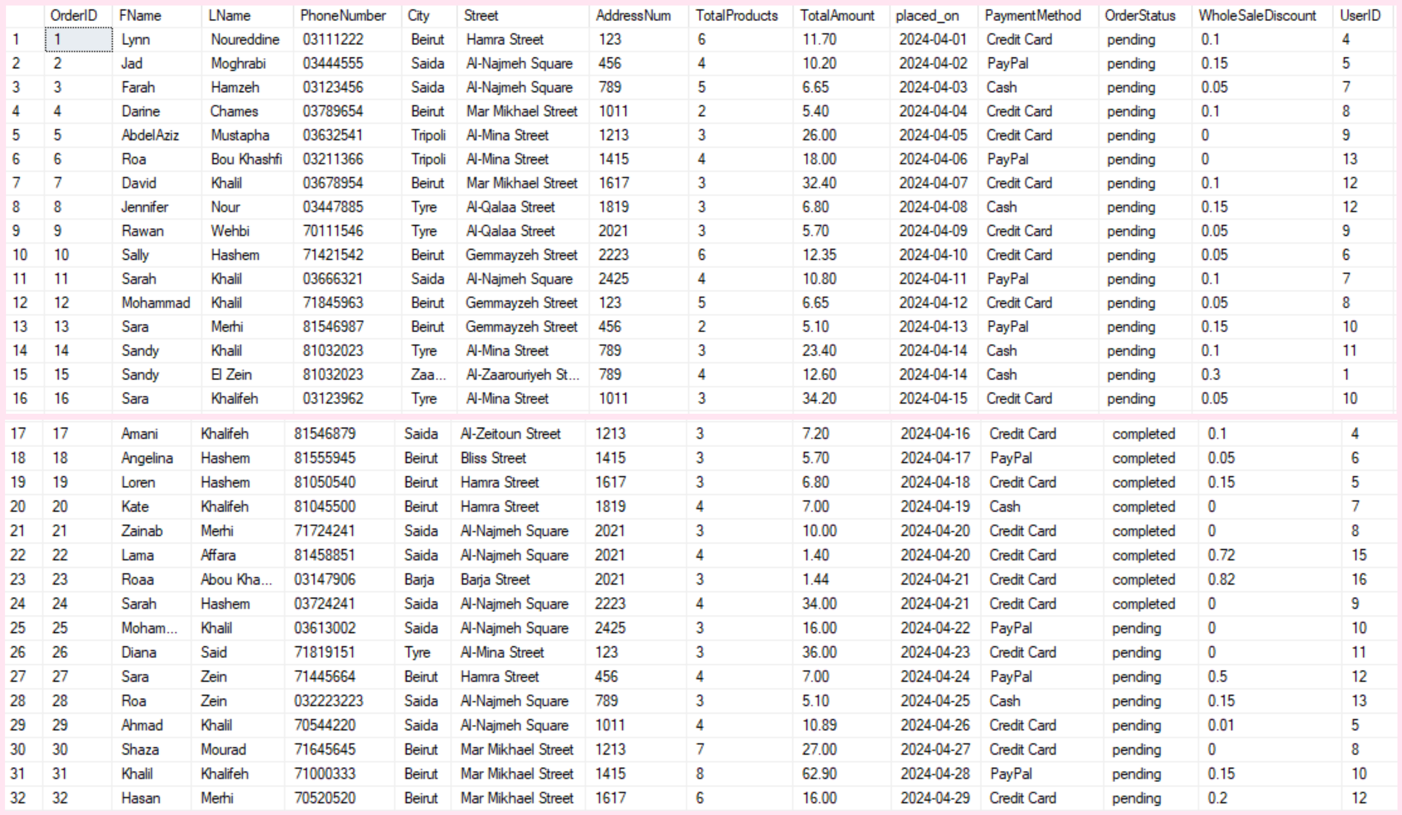
	Customer Name	Customer Email
1	Lama	Lama@gmail.com

```sql
--22. Find the number of orders placed by each customer.
Select U.Name As 'Customer Name', U.Email As 'Customer Email', Count(O.OrderID) As 'Num Of
Orders'
From Users As U
Left Join Orders As O On U.UserId = O.UserID
Where U.RoleID = 2
Group By U.Name, U.Email;
```



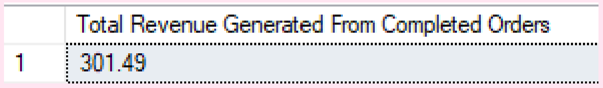
	Customer Name	Customer Email	Num Of Orders
1	AbdelAziz	AbdelAziz@gmail.com	2
2	Adnan	Adnan@gmail.com	2
3	Dana	Dana@gmail.com	4
4	Darine	Darine@gmail.com	4
5	Ezzeldine	Ezzeldine@gmail.com	4
6	Farah	Farah@gmail.com	3
7	Jad	Jad@gmail.com	3
8	Lama	Lama@gmail.com	1
9	Lynn	Lynn@gmail.com	2
10	Mhmd	Mhmd@gmail.com	2
11	Moamen	Moamen@gmail.com	2
12	Roa	Roa@gmail.com	1
13	Sanaa	Sanaa@gmail.com	0

```sql
--23. Display all orders placed by the customers.
Select * From Orders As O, Users As U Where U.roleID = 2;
```



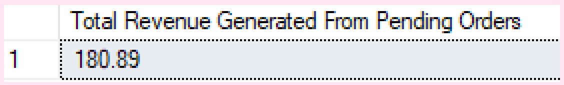
	OrderID	FName	LName	PhoneNumber	City	Street	AddressNum	TotalProducts	TotalAmount	placed_on	PaymentMethod	OrderStatus	WholeSaleDiscount	UserID
1	1	Lynn	Noureddine	03111222	Beirut	Hamra Street	123	6	11.70	2024-04-01	Credit Card	pending	0.1	4
2	2	Jad	Moghrabi	03444555	Saida	Al-Najmeh Square	456	4	10.20	2024-04-02	PayPal	pending	0.15	5
3	3	Farah	Hamzeh	03123456	Saida	Al-Najmeh Square	789	5	6.65	2024-04-03	Cash	pending	0.05	7
4	4	Darine	Chames	03789654	Beirut	Mar Mikhael Street	1011	2	5.40	2024-04-04	Credit Card	pending	0.1	8
5	5	AbdelAziz	Mustapha	03632541	Tripoli	Al-Mina Street	1213	3	26.00	2024-04-05	Credit Card	pending	0	9
6	6	Roa	Bou Khashfi	03211366	Tripoli	Al-Mina Street	1415	4	18.00	2024-04-06	PayPal	pending	0	13
7	7	David	Khalil	03678954	Beirut	Mar Mikhael Street	1617	3	32.40	2024-04-07	Credit Card	pending	0.1	12
8	8	Jennifer	Nour	03447885	Tyre	Al-Qalaa Street	1819	3	6.80	2024-04-08	Cash	pending	0.15	12
9	9	Rawan	Wehbi	70111546	Tyre	Al-Qalaa Street	2021	3	5.70	2024-04-09	Credit Card	pending	0.05	9
10	10	Sally	Hashem	71421542	Beirut	Gemmayzeh Street	2223	6	12.35	2024-04-10	Credit Card	pending	0.05	6
11	11	Sarah	Khalil	03666321	Saida	Al-Najmeh Square	2425	4	10.80	2024-04-11	PayPal	pending	0.1	7
12	12	Mohammad	Khalil	71845963	Beirut	Gemmayzeh Street	123	5	6.65	2024-04-12	Credit Card	pending	0.05	8
13	13	Sara	Merhi	81546987	Beirut	Gemmayzeh Street	456	2	5.10	2024-04-13	PayPal	pending	0.15	10
14	14	Sandy	Khalil	81032023	Tyre	Al-Mina Street	789	3	23.40	2024-04-14	Cash	pending	0.1	11
15	15	Sandy	El Zein	81032023	Zaa...	Al-Zaarouriyeh St...	789	4	12.60	2024-04-14	Cash	pending	0.3	1
16	16	Sara	Khalifeh	03123962	Tyre	Al-Mina Street	1011	3	34.20	2024-04-15	Credit Card	pending	0.05	10
17	17	Amani	Khalifeh	81546879	Saida	Al-Zeitoun Street	1213	3	7.20	2024-04-16	Credit Card	completed	0.1	4
18	18	Angelina	Hashem	81555945	Beirut	Bliss Street	1415	3	5.70	2024-04-17	PayPal	completed	0.05	6
19	19	Loren	Hashem	81050540	Beirut	Hamra Street	1617	3	6.80	2024-04-18	Credit Card	completed	0.15	5
20	20	Kate	Khalifeh	81045500	Beirut	Hamra Street	1819	4	7.00	2024-04-19	Cash	completed	0	7
21	21	Zainab	Merhi	71724241	Saida	Al-Najmeh Square	2021	3	10.00	2024-04-20	Credit Card	completed	0	8
22	22	Lama	Affara	81458851	Saida	Al-Najmeh Square	2021	4	1.40	2024-04-20	Credit Card	completed	0.72	15
23	23	Roaa	Abou Kha...	03147906	Barja	Barja Street	2021	3	1.44	2024-04-21	Credit Card	completed	0.82	16
24	24	Sarah	Hashem	03724241	Saida	Al-Najmeh Square	2223	4	34.00	2024-04-21	Credit Card	completed	0	9
25	25	Moham...	Khalil	03613002	Saida	Al-Najmeh Square	2425	3	16.00	2024-04-22	PayPal	pending	0	10
26	26	Diana	Said	71819151	Tyre	Al-Mina Street	123	3	36.00	2024-04-23	Credit Card	pending	0	11
27	27	Sara	Zein	71445664	Beirut	Hamra Street	456	4	7.00	2024-04-24	PayPal	pending	0.5	12
28	28	Roa	Zein	032223223	Saida	Al-Najmeh Square	789	3	5.10	2024-04-25	Cash	pending	0.15	13
29	29	Ahmad	Khalil	70544220	Saida	Al-Najmeh Square	1011	4	10.89	2024-04-26	Credit Card	pending	0.01	5
30	30	Shaza	Mourad	71645645	Beirut	Mar Mikhael Street	1213	7	27.00	2024-04-27	Credit Card	pending	0	8
31	31	Khalil	Khalifeh	71000333	Beirut	Mar Mikhael Street	1415	8	62.90	2024-04-28	PayPal	pending	0.15	10
32	32	Hasan	Merhi	70520520	Beirut	Mar Mikhael Street	1617	6	16.00	2024-04-29	Credit Card	pending	0.2	12

```sql
--24. Show the total revenue generated from completed orders.
Select Sum(TotalAmount) As 'Total Revenue Generated From Completed Orders' From Orders Where
orderStatus = 'completed';
```



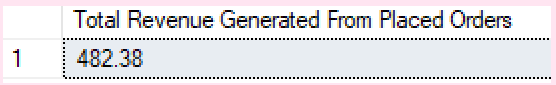
	Total Revenue Generated From Completed Orders
1	301.49

```sql
--25. Show the total revenue generated from pending orders.
Select Sum(TotalAmount) As 'Total Revenue Generated From Pending Orders' From Orders Where
orderStatus='pending';
```



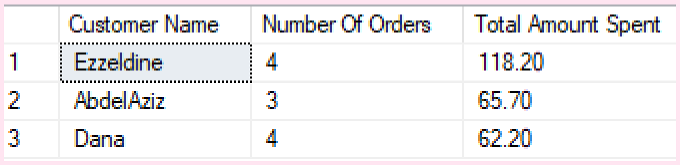
	Total Revenue Generated From Pending Orders
1	180.89

```sql
--26. Show the total revenue generated from placed orders.
Select Sum(TotalAmount) AS 'Total Revenue Generated From Placed Orders' From Orders;
```



	Total Revenue Generated From Placed Orders
1	482.38

```sql
--27. Display the top 3 most valuable customers with the highest number of orders.
Select Top 3 U.Name As 'Customer Name', Count(Distinct O.OrderID) As 'Number Of Orders',
Sum(O.TotalAmount) As 'Total Amount Spent' From Users As U
Inner Join Orders As O ON U.UserID = O.UserID
Group By U.UserID, U.Name Order By Sum(O.TotalAmount) DESC, Count(Distinct O.OrderID) DESC;
```

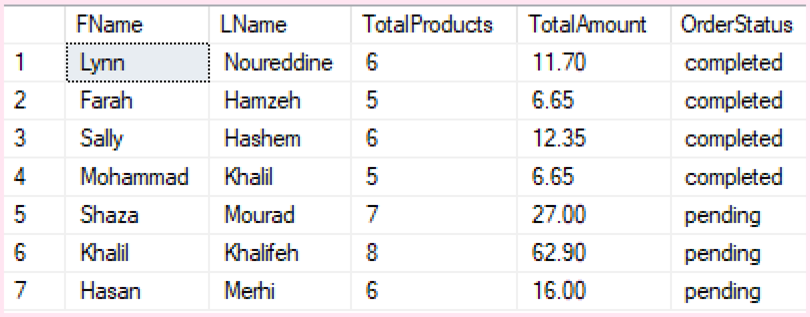


	Customer Name	Number Of Orders	Total Amount Spent
1	Ezzeldine	4	118.20
2	AbdelAziz	3	65.70
3	Dana	4	62.20

```sql
--28. Mark all the orders placed before 2024-04-22 as completed
UPDATE Orders Set OrderStatus = 'completed' Where placed_on < '2024-04-22';
```

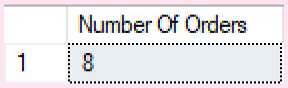
```
(24 row(s) affected)
```

```sql
--29. Display all the orders that include more than 4 items.
Select O.FName, O.LName, O.TotalProducts, O.TotalAmount, O.OrderStatus
From Orders as O Where O.TotalProducts > 4;
```



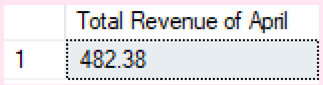
	FName	LName	TotalProducts	TotalAmount	OrderStatus
1	Lynn	Noureddine	6	11.70	completed
2	Farah	Hamzeh	5	6.65	completed
3	Sally	Hashem	6	12.35	completed
4	Mohammad	Khalil	5	6.65	completed
5	Shaza	Mourad	7	27.00	pending
6	Khalil	Khalifeh	8	62.90	pending
7	Hasan	Merhi	6	16.00	pending

```sql
--30. Calculate the number of orders that have food items belonging to the 'Croissant'
Category.
Select Count(Distinct O.OrderID) As 'Number Of Orders' From Orders As O
Join orderDetails As Od On O.OrderID = Od.OrderID
Join FoodItem  As F On Od.FoodItemID = F.FoodItemID
Join Category As C On C.CategoryID = F.CategoryID
Where C.Name = 'Croissant';
```



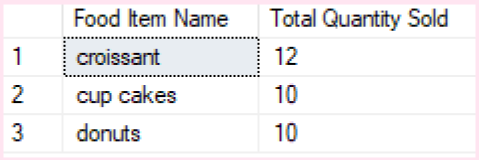
	Number Of Orders
1	8

```sql
--31. Display the total revenue generated by the bakery in the month of april
Select Sum(O.TotalAmount) As 'Total Revenue of April' From Orders As O
Where O.placed_on >= '2024-04-01' AND O.placed_on <= '2024-04-30';
```



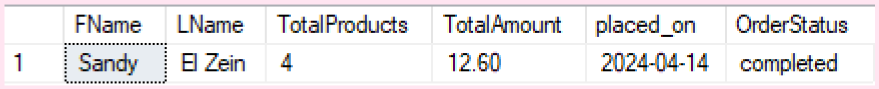
	Total Revenue of April
1	482.38

```sql
--32. Show the top 3 best-selling baked goods.
-- based on the total quantity sold
Select Top 3 F.Name As 'Food Item Name', SUM(Od.Quantity) As 'Total Quantity Sold' From
FoodItem As F
Join OrderDetails As Od On F.FoodItemID = Od.FoodItemID
Group By F.Name Order By 'Total Quantity Sold' DESC;
```



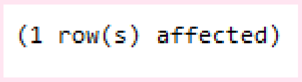
	Food Item Name	Total Quantity Sold
1	croissant	12
2	cup cakes	10
3	donuts	10

```sql
--33. Retrieve all orders that were placed by admins and not by customers.
Select O.FName, O.LName, O.TotalProducts, O.TotalAmount, O.placed_on, O.OrderStatus From
Orders As O
Left Outer Join Users As U On O.UserID = U.UserID
Left Outer Join Role As R On U.RoleID = R.RoleID
WHERE R.RoleID = 1;
```



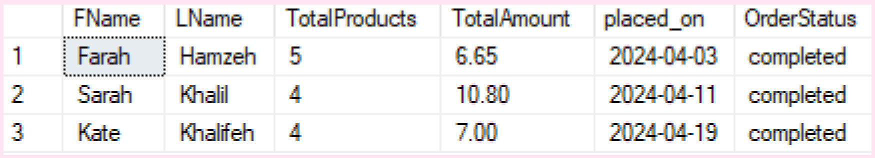
	FName	LName	TotalProducts	TotalAmount	placed_on	OrderStatus
1	Sandy	El Zein	4	12.60	2024-04-14	completed

```sql
--34. A customer whose orderID is 5 wants to cancel their order
Delete From Orders Where OrderID = 5;
```



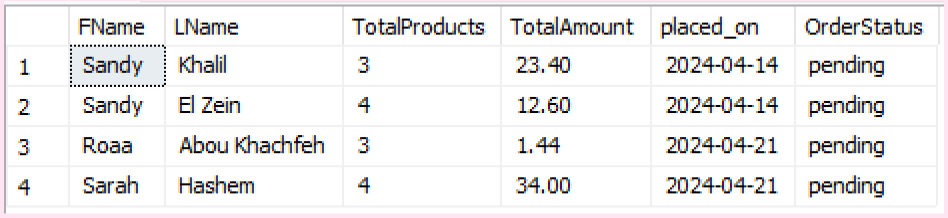
```
(1 row(s) affected)
```

```sql
--35. Retrieve all orders made by the customer Farah.
Select O.FName, O.LName, O.TotalProducts, O.TotalAmount, O.placed_on, O.OrderStatus From
Orders As O
Where UserID IN (Select U.UserID From Users As U Where Name LIKE 'Farah%' AND ROLEID = 2);
```



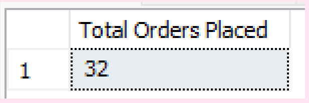
	FName	LName	TotalProducts	TotalAmount	placed_on	OrderStatus
1	Farah	Hamzeh	5	6.65	2024-04-03	completed
2	Sarah	Khalil	4	10.80	2024-04-11	completed
3	Kate	Khalifeh	4	7.00	2024-04-19	completed

```sql
--36. Retrieve all orders placed on the 14th and 21st of april
Select O.FName, O.LName, O.TotalProducts, O.TotalAmount, O.placed_on, O.OrderStatus From
Orders As O
Where placed_on IN ('2024-04-14', '2024-04-21');
```



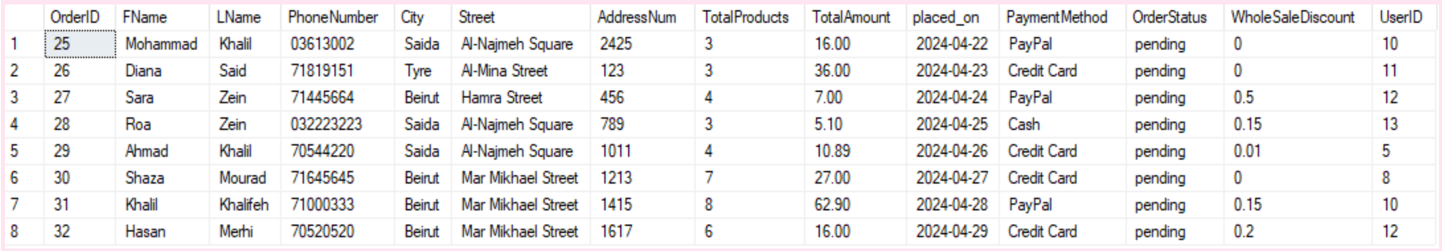
	FName	LName	TotalProducts	TotalAmount	placed_on	OrderStatus
1	Sandy	Khalil	3	23.40	2024-04-14	pending
2	Sandy	El Zein	4	12.60	2024-04-14	pending
3	Roaa	Abou Khachfeh	3	1.44	2024-04-21	pending
4	Sarah	Hashem	4	34.00	2024-04-21	pending

```
--37. Retrieve the total number of orders placed from the first day the bakery opened till today.
Select Count(*) AS 'Total Orders Placed' From Orders;
```



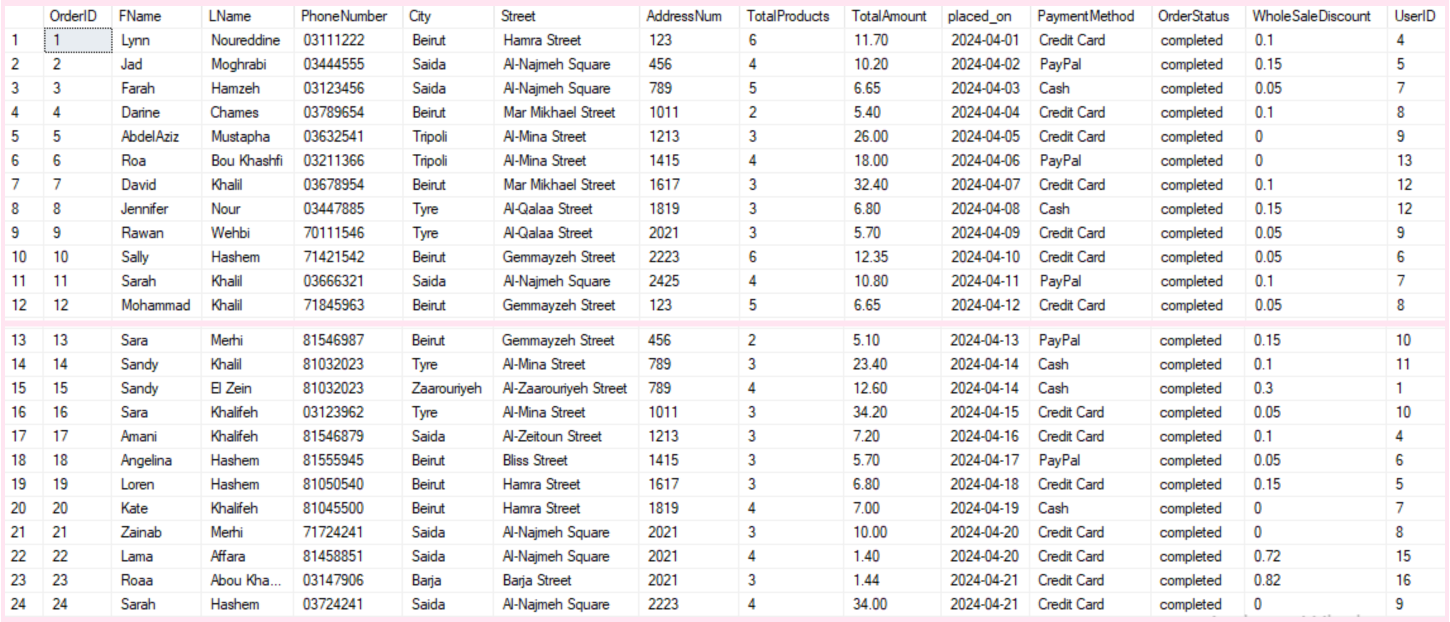
	Total Orders Placed
1	32

```
--38. List all orders that are pending and yet to be delivered from the oldest order to the most recent one.
Select * From Orders Where OrderStatus NOT IN (Select OrderStatus FROM Orders WHERE OrderStatus = 'completed')
ORDER BY placed_on ASC;
```



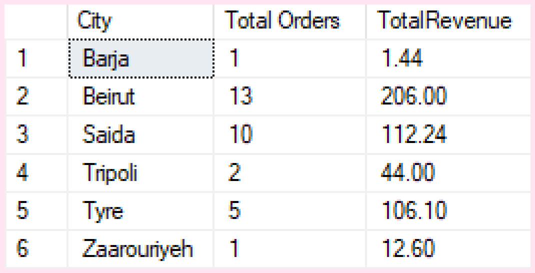
	OrderID	FName	LName	PhoneNumber	City	Street	AddressNum	TotalProducts	TotalAmount	placed_on	PaymentMethod	OrderStatus	WholeSaleDiscount	UserID
1	25	Mohammad	Khalil	03613002	Saida	Al-Najmeh Square	2425	3	16.00	2024-04-22	PayPal	pending	0	10
2	26	Diana	Said	71819151	Tyre	Al-Mina Street	123	3	36.00	2024-04-23	Credit Card	pending	0	11
3	27	Sara	Zein	71445664	Beirut	Hamra Street	456	4	7.00	2024-04-24	PayPal	pending	0.5	12
4	28	Roa	Zein	032223223	Saida	Al-Najmeh Square	789	3	5.10	2024-04-25	Cash	pending	0.15	13
5	29	Ahmad	Khalil	70544220	Saida	Al-Najmeh Square	1011	4	10.89	2024-04-26	Credit Card	pending	0.01	5
6	30	Shaza	Mourad	71645645	Beirut	Mar Mikhael Street	1213	7	27.00	2024-04-27	Credit Card	pending	0	8
7	31	Khalil	Khalifeh	71000333	Beirut	Mar Mikhael Street	1415	8	62.90	2024-04-28	PayPal	pending	0.15	10
8	32	Hasan	Merhi	70520520	Beirut	Mar Mikhael Street	1617	6	16.00	2024-04-29	Credit Card	pending	0.2	12

```
--39. List all orders completed and ready for delivery.
Select * From Orders Where orderStatus = 'completed';
```



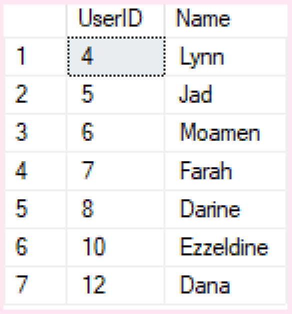
	OrderID	FName	LName	PhoneNumber	City	Street	AddressNum	TotalProducts	TotalAmount	placed_on	PaymentMethod	OrderStatus	WholeSaleDiscount	UserID
1	1	Lynn	Noureddine	03111222	Beirut	Hamra Street	123	6	11.70	2024-04-01	Credit Card	completed	0.1	4
2	2	Jad	Moghrabi	03444555	Saida	Al-Najmeh Square	456	4	10.20	2024-04-02	PayPal	completed	0.15	5
3	3	Farah	Hamzeh	03123456	Saida	Al-Najmeh Square	789	5	6.65	2024-04-03	Cash	completed	0.05	7
4	4	Darine	Chames	03789654	Beirut	Mar Mikhael Street	1011	2	5.40	2024-04-04	Credit Card	completed	0.1	8
5	5	AbdelAziz	Mustapha	03632541	Tripoli	Al-Mina Street	1213	3	26.00	2024-04-05	Credit Card	completed	0	9
6	6	Roa	Bou Khashfi	03211366	Tripoli	Al-Mina Street	1415	4	18.00	2024-04-06	PayPal	completed	0	13
7	7	David	Khalil	03678954	Beirut	Mar Mikhael Street	1617	3	32.40	2024-04-07	Credit Card	completed	0.1	12
8	8	Jennifer	Nour	03447885	Tyre	Al-Qalaa Street	1819	3	6.80	2024-04-08	Cash	completed	0.15	12
9	9	Rawan	Wehbi	70111546	Tyre	Al-Qalaa Street	2021	3	5.70	2024-04-09	Credit Card	completed	0.05	9
10	10	Sally	Hashem	71421542	Beirut	Gemmayzeh Street	2223	6	12.35	2024-04-10	Credit Card	completed	0.05	6
11	11	Sarah	Khalil	03666321	Saida	Al-Najmeh Square	2425	4	10.80	2024-04-11	PayPal	completed	0.1	7
12	12	Mohammad	Khalil	71845963	Beirut	Gemmayzeh Street	123	5	6.65	2024-04-12	Credit Card	completed	0.05	8
13	13	Sara	Merhi	81546987	Beirut	Gemmayzeh Street	456	2	5.10	2024-04-13	PayPal	completed	0.15	10
14	14	Sandy	Khalil	81032023	Tyre	Al-Mina Street	789	3	23.40	2024-04-14	Cash	completed	0.1	11
15	15	Sandy	El Zein	81032023	Zaarouriyeh	Al-Zaarouriyeh Street	789	4	12.60	2024-04-14	Cash	completed	0.3	1
16	16	Sara	Khalifeh	03123962	Tyre	Al-Mina Street	1011	3	34.20	2024-04-15	Credit Card	completed	0.05	10
17	17	Amani	Khalifeh	81546879	Saida	Al-Zeitoun Street	1213	3	7.20	2024-04-16	Credit Card	completed	0.1	4
18	18	Angelina	Hashem	81555945	Beirut	Bliss Street	1415	3	5.70	2024-04-17	PayPal	completed	0.05	6
19	19	Loren	Hashem	81050540	Beirut	Hamra Street	1617	3	6.80	2024-04-18	Credit Card	completed	0.15	5
20	20	Kate	Khalifeh	81045500	Beirut	Hamra Street	1819	4	7.00	2024-04-19	Cash	completed	0	7
21	21	Zainab	Merhi	71724241	Saida	Al-Najmeh Square	2021	3	10.00	2024-04-20	Credit Card	completed	0	8
22	22	Lama	Affara	81458851	Saida	Al-Najmeh Square	2021	4	1.40	2024-04-20	Credit Card	completed	0.72	15
23	23	Roaa	Abou Kha...	03147906	Barja	Barja Street	2021	3	1.44	2024-04-21	Credit Card	completed	0.82	16
24	24	Sarah	Hashem	03724241	Saida	Al-Najmeh Square	2223	4	34.00	2024-04-21	Credit Card	completed	0	9

```sql
--40. Display the number of orders placed and the total revenue generated from the order in
each city.
Select City, Count(OrderID) As 'Total Orders', Sum(O.TotalAmount) As TotalRevenue From Orders
As O
Group By O.City;
```



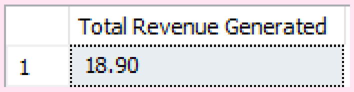
	City	Total Orders	TotalRevenue
1	Barja	1	1.44
2	Beirut	13	206.00
3	Saida	10	112.24
4	Tripoli	2	44.00
5	Tyre	5	106.10
6	Zaarouriyeh	1	12.60

```sql
--41. List all customers who have placed orders from a specific city 'Beirut'.
Select Distinct U.UserID, U.Name From Users U
Left Outer Join Orders As O On U.UserID = O.UserID
Where O.City='Beirut' AND U.RoleID=2;
```



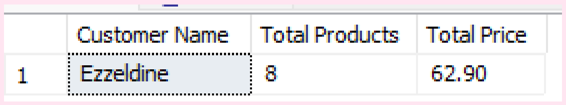
	UserID	Name
1	4	Lynn
2	5	Jad
3	6	Moamen
4	7	Farah
5	8	Darine
6	10	Ezzeldine
7	12	Dana

```sql
--42. Display the total revenue generated from orders placed by a customer with UserId = 4.
Select Sum(O.TotalAmount) As 'Total Revenue Generated' From Orders As O
Where O.UserID = 4;
```



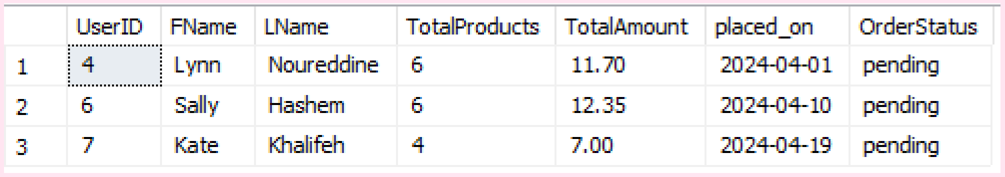
	Total Revenue Generated
1	18.90

```sql
--43. Display the customer and total price of the most expensive order
Select U.Name As 'Customer Name', O.TotalProducts As 'Total Products', O.TotalAmount As
'Total Price' From Orders As O
JOIN Users As U On O.UserID = U.UserID
Where O.TotalAmount = (Select Max(TotalAmount) From Orders);
```



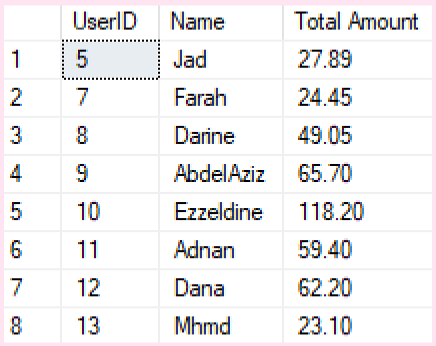
	Customer Name	Total Products	Total Price
1	Ezzeldine	8	62.90

```sql
--44. Find all users who have placed orders containing 'chocolate chip cookies' and provide
details of those orders.
Select U.UserID, O.FName, O.LName, O.TotalProducts, O.TotalAmount, O.placed_on, O.OrderStatus
From Users As U
Join Orders As O On U.UserID = O.UserID
Join orderDetails As Od On O.OrderID = Od.OrderID
Where EXISTS (Select 1 From FoodItem AS F
Where F.FoodItemID = Od.FoodItemID AND F.Name = 'chocolate chip cookies');
```



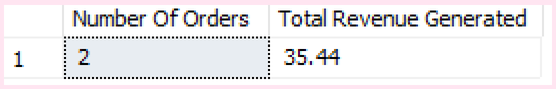
	UserID	FName	LName	TotalProducts	TotalAmount	placed_on	OrderStatus
1	4	Lynn	Noureddine	6	11.70	2024-04-01	pending
2	6	Sally	Hashem	6	12.35	2024-04-10	pending
3	7	Kate	Khalifeh	4	7.00	2024-04-19	pending

```sql
--45. List all customers who have placed orders worth more than $20.
Select U.UserID, U.Name, Sum(O.TotalAmount) as 'Total Amount' From Users As U, Orders As O
Where U.UserID = O.UserID
Group By U.UserID, U.Name Having Sum(O.TotalAmount) > 20;
```



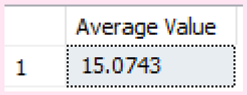
	UserID	Name	Total Amount
1	5	Jad	27.89
2	7	Farah	24.45
3	8	Darine	49.05
4	9	AbdelAziz	65.70
5	10	Ezzeldine	118.20
6	11	Adnan	59.40
7	12	Dana	62.20
8	13	Mhmd	23.10

```sql
--46. Show the number of orders and total revenue generated on '2024-04-21'
Select COUNT(*) As 'Number Of Orders', Sum(TotalAmount) As 'Total Revenue Generated'
From Orders As O Where O.placed_on = '2024-04-21';
```



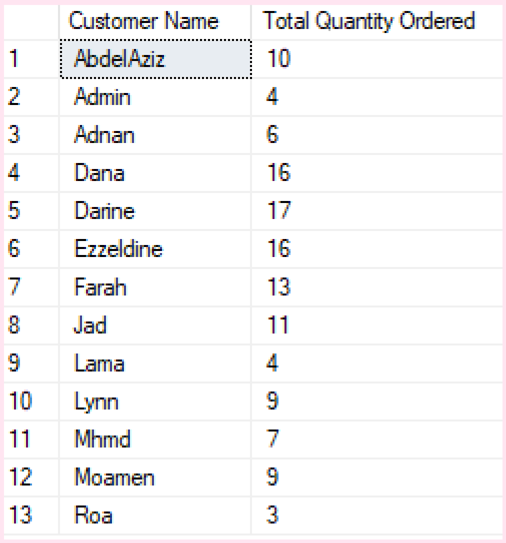
	Number Of Orders	Total Revenue Generated
1	2	35.44

```sql
--47. Retrieve the average order price.
Select AVG(TotalAmount) As 'Average Value' From Orders;
```



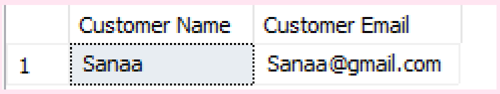
	Average Value
1	15.0743

```
--48. Show the total quantity of food items over all the order ordered by each customer
Select U.Name As 'Customer Name', Sum(Od.Quantity) As 'Total Quantity Ordered' From Users As
U
Join Orders As O On U.UserID =O.UserID
Join orderDetails As Od On O.OrderID= Od.OrderID
Join FoodItem As F On Od.FoodItemID = F.FoodItemID
GROUP BY U.Name;
```



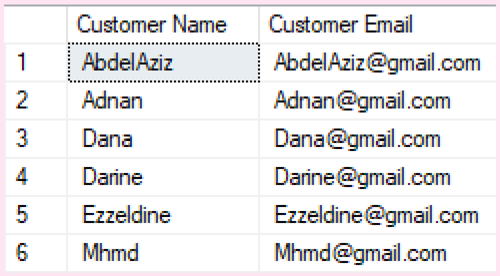
	Customer Name	Total Quantity Ordered
1	AbdelAziz	10
2	Admin	4
3	Adnan	6
4	Dana	16
5	Darine	17
6	Ezzeldine	16
7	Farah	13
8	Jad	11
9	Lama	4
10	Lynn	9
11	Mhmd	7
12	Moamen	9
13	Roa	3

```
--49. Show the number of customers who have registered but not placed any orders.
Select U.Name As 'Customer Name', U.Email As 'Customer Email' From Users As U
Where UserID NOT IN (Select Distinct UserID From Orders) AND U.RoleID = 2;
```



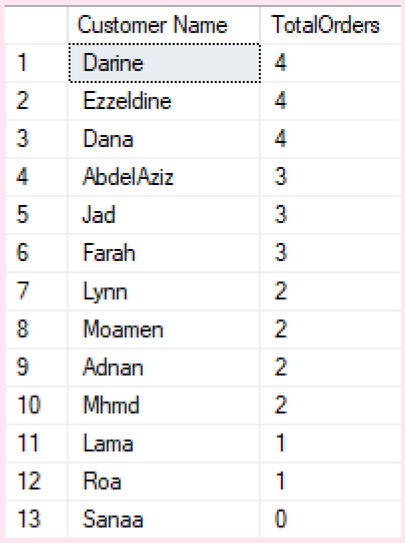
	Customer Name	Customer Email
1	Sanaa	Sanaa@gmail.com

```
--50. List all customers who have placed orders above the average order price.
Select Distinct U.Name as 'Customer Name', U.Email As 'Customer Email' From Users As U
Join Orders As O On U.UserID =O.UserID
Where O.TotalAmount > (Select AVG(TotalAmount) From Orders);
```



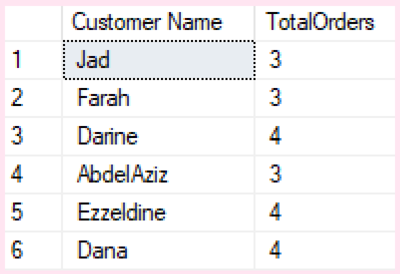
	Customer Name	Customer Email
1	AbdelAziz	AbdelAziz@gmail.com
2	Adnan	Adnan@gmail.com
3	Dana	Dana@gmail.com
4	Darine	Darine@gmail.com
5	Ezzeldine	Ezzeldine@gmail.com
6	Mhmd	Mhmd@gmail.com

```
--51. Retrieve all customers along with the total number of orders they've placed.
Select U.Name as 'Customer Name', Count(O.OrderID) As 'TotalOrders' From Users As U
Left Join Orders As O On U.UserID=O.UserID
Where U.RoleID=2 Group By U.UserID, U.Name Order By TotalOrders DESC;
```



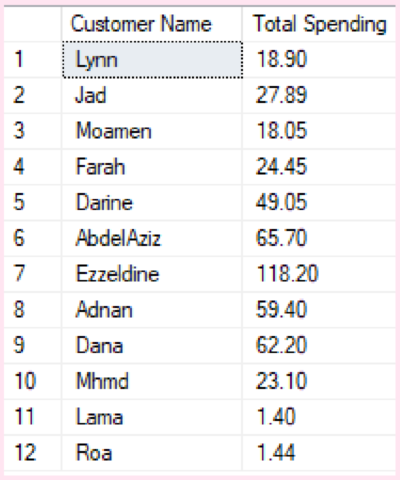
	Customer Name	TotalOrders
1	Darine	4
2	Ezzeldine	4
3	Dana	4
4	AbdelAziz	3
5	Jad	3
6	Farah	3
7	Lynn	2
8	Moamen	2
9	Adnan	2
10	Mhmd	2
11	Lama	1
12	Roa	1
13	Sanaa	0

```
--52. Retrieve all customers who have placed more than 2 orders.
Select U.Name as 'Customer Name', Count(O.OrderID) As 'TotalOrders' From Users As U
Left Join Orders O On U.UserID=O.UserID
Where U.RoleID=2 Group By U.UserID, U.Name HAVING Count(O.OrderID) > 2;
```



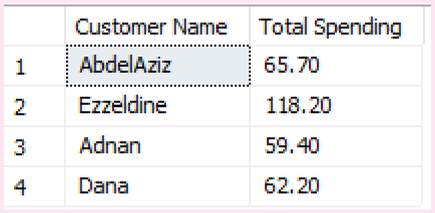
	Customer Name	TotalOrders
1	Jad	3
2	Farah	3
3	Darine	4
4	AbdelAziz	3
5	Ezzeldine	4
6	Dana	4

```
--53. Retrieve all customers along with their total spending.
Select U.Name as 'Customer Name',Sum(O.TotalAmount) As 'Total Spending' From Users As U
Join Orders As O On U.UserId=O.UserID
Where U.RoleID=2 Group By U.UserId,U.Name;
```



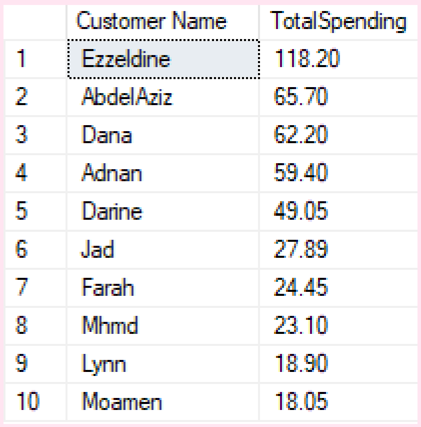
	Customer Name	Total Spending
1	Lynn	18.90
2	Jad	27.89
3	Moamen	18.05
4	Farah	24.45
5	Darine	49.05
6	AbdelAziz	65.70
7	Ezzeldine	118.20
8	Adnan	59.40
9	Dana	62.20
10	Mhmd	23.10
11	Lama	1.40
12	Roa	1.44

```
--54. Retrieve all customers who have spent more than $50.
Select  U.Name as 'Customer Name',Sum(O.TotalAmount) As 'Total Spending' From Users As U
Join Orders As O On U.UserId=O.UserID
Where U.RoleID=2 Group By U.UserId,U.Name Having Sum(O.TotalAmount)>50;
```



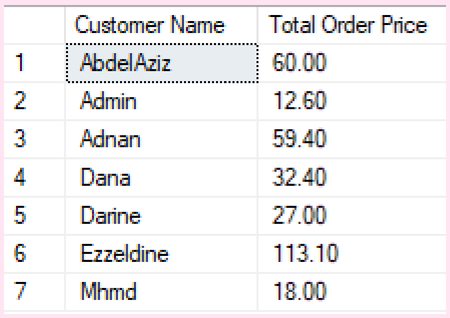
	Customer Name	Total Spending
1	AbdelAziz	65.70
2	Ezzeldine	118.20
3	Adnan	59.40
4	Dana	62.20

```
--55. Retrieve the top 10 customers by total spending.
Select Top 10 U.Name as 'Customer Name',Sum(O.TotalAmount) AS 'TotalSpending' From Users As U
Join Orders As O ON U.UserId=O.UserID
Where U.RoleID=2 Group By U.UserId,U.Name Order By TotalSpending DESC;
```



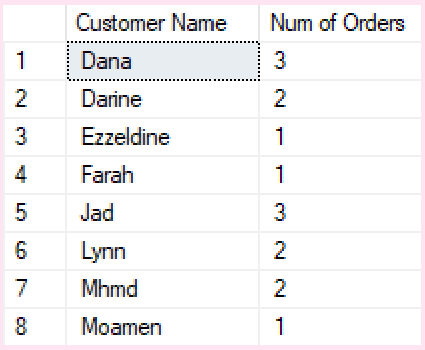
	Customer Name	TotalSpending
1	Ezzeldine	118.20
2	AbdelAziz	65.70
3	Dana	62.20
4	Adnan	59.40
5	Darine	49.05
6	Jad	27.89
7	Farah	24.45
8	Mhmd	23.10
9	Lynn	18.90
10	Moamen	18.05

```
--56. List the names of customers, along with their total order price who have placed orders
with a price greater than the average price of all food items.
Select U.Name As 'Customer Name', SUM(O.TotalAmount) AS 'Total Order Price' From Users As U
Join Orders As O On U.UserID = O.UserID
Where O.OrderID IN (Select Distinct OrderID From orderDetails
Where FoodItemID IN (Select FoodItemID From FoodItem
Where Price > (Select AVG(Price) From FoodItem)))
GROUP BY U.Name;
```



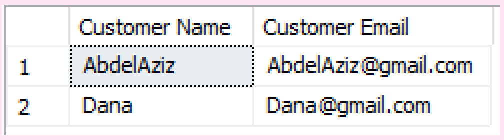
	Customer Name	Total Order Price
1	AbdelAziz	60.00
2	Admin	12.60
3	Adnan	59.40
4	Dana	32.40
5	Darine	27.00
6	Ezzeldine	113.10
7	Mhmd	18.00

```
--57. List the names of customers along with their number of orders that have food items from
more than one category
Select U.Name As 'Customer Name', Count(Distinct O.OrderID) AS 'Num of Orders' From Users As
U
Join Orders As O On U.UserID = O.UserID
Where U.RoleID = 2 AND O.OrderID IN (Select O.OrderID From Orders O
Join orderDetails As Od On O.OrderID = Od.OrderID
Join FoodItem As F On Od.FoodItemID = F.FoodItemID
Group By O.OrderID Having Count(Distinct F.CategoryID) > 1)
Group By U.Name;
```



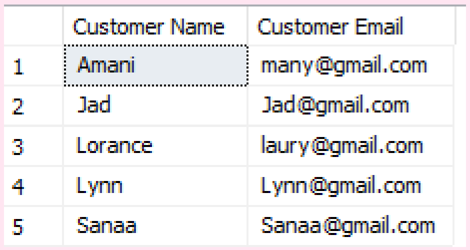
	Customer Name	Num of Orders
1	Dana	3
2	Darine	2
3	Ezzeldine	1
4	Farah	1
5	Jad	3
6	Lynn	2
7	Mhmd	2
8	Moamen	1

```
--58. Display the customers who have made at least 1 order with a total price above $20, and
have submitted a rating of 5 stars and wrote a review
Select Distinct U.Name as 'Customer Name', U.Email As 'Customer Email' From Users As U
Where U.UserID IN (Select O.UserID From Orders As O
Where O.TotalAmount > 20 AND O.UserID IN (Select R.UserID From Reviews As R
Where R.Rating = 5 AND R.UserID IN (Select O.UserID From Orders As o)));
```



	Customer Name	Customer Email
1	AbdelAziz	AbdelAziz@gmail.com
2	Dana	Dana@gmail.com

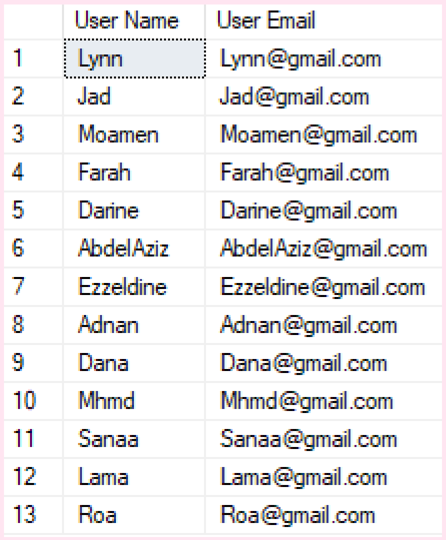
```
--59. Display all the customers who have not bought in the past month
Select Distinct U.Name as 'Customer Name', U.Email As 'Customer Email' From Users As U
Left Join Orders As O ON U.UserID =O.UserID
Where O.UserID IS NULL OR O.placed_on < DATEADD(month, -1, GETDATE());
```



	Customer Name	Customer Email
1	Amani	many@gmail.com
2	Jad	Jad@gmail.com
3	Lorance	laury@gmail.com
4	Lynn	Lynn@gmail.com
5	Sanaa	Sanaa@gmail.com

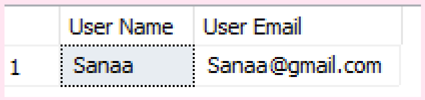
```sql
-- CART AND WISHLIST RELATED

--60. List all customers who have added items to their cart.
Select U.Name As 'User Name', U.Email As 'User Email' From Users As U
Full Outer Join Cart As C ON U.UserID =C.UserID
Where U.RoleID=2;
```



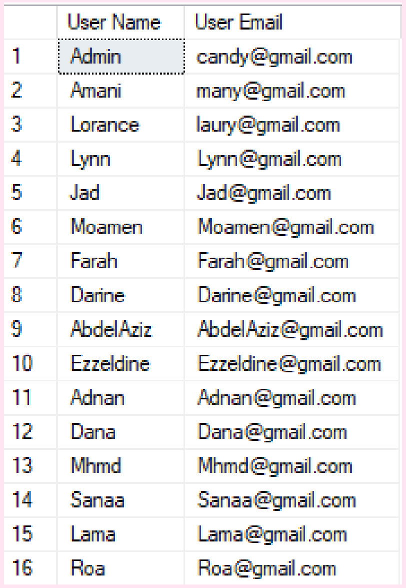
	User Name	User Email
1	Lynn	Lynn@gmail.com
2	Jad	Jad@gmail.com
3	Moamen	Moamen@gmail.com
4	Farah	Farah@gmail.com
5	Darine	Darine@gmail.com
6	AbdelAziz	AbdelAziz@gmail.com
7	Ezzeldine	Ezzeldine@gmail.com
8	Adnan	Adnan@gmail.com
9	Dana	Dana@gmail.com
10	Mhmd	Mhmd@gmail.com
11	Sanaa	Sanaa@gmail.com
12	Lama	Lama@gmail.com
13	Roa	Roa@gmail.com

```sql
--61. Retrieve all customers who have not yet checked out their orders.
Select U.Name As 'User Name', U.Email As 'User Email' From Users As U
Where RoleID NOT LIKE 1 AND UserID IN (SELECT UserID FROM Cart) AND UserID NOT IN(SELECT
UserID FROM Orders);
```



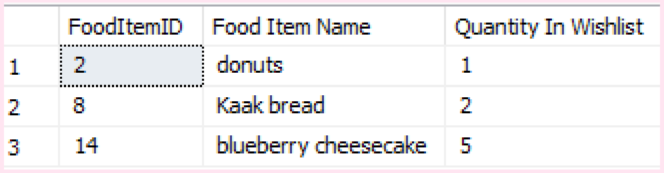
	User Name	User Email
1	Sanaa	Sanaa@gmail.com

```sql
--62. List all customers who have items in their wishlist.
Select U.Name As 'User Name', U.Email As 'User Email' From Users As U
Where UserID IN (Select Distinct UserID From Wishlist);
```



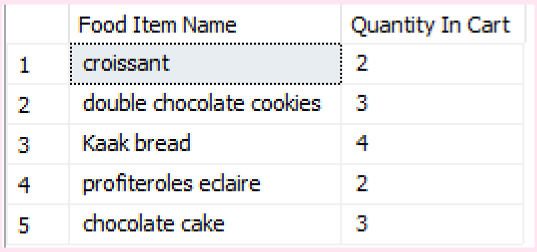
	User Name	User Email
1	Admin	candy@gmail.com
2	Amani	many@gmail.com
3	Lorance	laury@gmail.com
4	Lynn	Lynn@gmail.com
5	Jad	Jad@gmail.com
6	Moamen	Moamen@gmail.com
7	Farah	Farah@gmail.com
8	Darine	Darine@gmail.com
9	AbdelAziz	AbdelAziz@gmail.com
10	Ezzeldine	Ezzeldine@gmail.com
11	Adnan	Adnan@gmail.com
12	Dana	Dana@gmail.com
13	Mhmd	Mhmd@gmail.com
14	Sanaa	Sanaa@gmail.com
15	Lama	Lama@gmail.com
16	Roa	Roa@gmail.com

```sql
--63. Display all items with their quantities in the wishlist of a user with userid = 14
Select Wc.FoodItemID, F.Name As 'Food Item Name', Wc.Quantity As 'Quantity In Wishlist' From
wishContains As Wc
Join FoodItem As F On Wc.FoodItemID = F.FoodItemID
Where Wc.wishListID = (Select wishListID From wishlist Where UserID = 14);
```



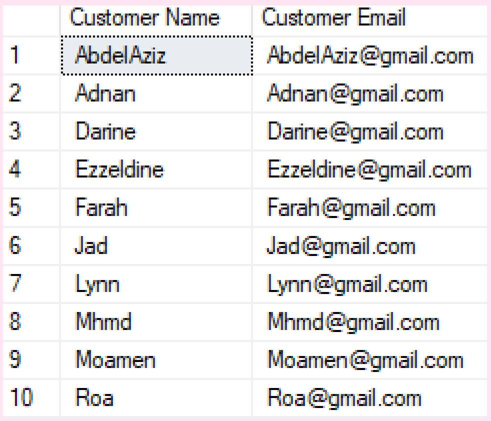
	FoodItemID	Food Item Name	Quantity In Wishlist
1	2	donuts	1
2	8	Kaak bread	2
3	14	blueberry cheesecake	5

```sql
--64. Show all items with their quantities in the cart of a user with userid = 2
Select F.Name As 'Food Item Name', Ci.Quantity As 'Quantity In Cart' From Cart As C
Join CartIncludes As Ci On C.CartID = Ci.CartID
Join FoodItem As F ON Ci.FoodItemID = F.FoodItemID
Where C.UserID = 2;
```



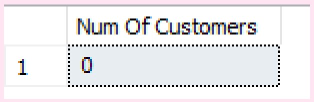
	Food Item Name	Quantity In Cart
1	croissant	2
2	double chocolate cookies	3
3	Kaak bread	4
4	profiteroles eclaire	2
5	chocolate cake	3

```sql
--65. Show all users that have placed orders for food items that are also present in their
wishlists
Select Distinct U.Name As 'Customer Name', U.email As 'Customer Email' From Users As U
Where U.UserID IN (Select Distinct W.UserID from wishlist As W
Join wishContains As Wc ON W.wishListID = Wc.wishListID
Where Wc.FoodItemID IN (
Select Distinct Od.FoodItemID From Orders As o
Join orderDetails As Od On O.OrderID = Od.OrderID
Where O.UserID = W.UserID));
```



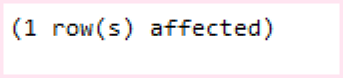
	Customer Name	Customer Email
1	AbdelAziz	AbdelAziz@gmail.com
2	Adnan	Adnan@gmail.com
3	Darine	Darine@gmail.com
4	Ezzeldine	Ezzeldine@gmail.com
5	Farah	Farah@gmail.com
6	Jad	Jad@gmail.com
7	Lynn	Lynn@gmail.com
8	Mhmd	Mhmd@gmail.com
9	Moamen	Moamen@gmail.com
10	Roa	Roa@gmail.com

```sql
--66. Display the number of customers who have registered but not added any items to their
wishlist.
Select Count(UserID) As 'Num Of Customers' From Users As U
Where U.UserID NOT IN (SELECT UserID FROM wishlist) AND U.RoleID = 2;
```



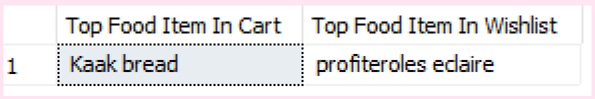
	Num Of Customers
1	0

```sql
--67. Delete food item named 'double chocolate cookies' from cartid = 8
Delete From CartIncludes
Where CartID = 4 AND
FoodItemID = (Select FoodItemID From FoodItem
Where Name = 'double chocolate cookies');
```



```
(1 row(s) affected)
```

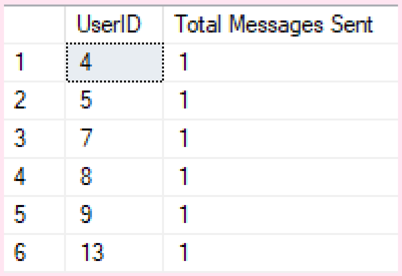
```sql
--68. Display the top 1 food item in the carts and in the wishlists of all customers
Select (Select Top 1 Name As 'Food Item Name' From FoodItem
Where FoodItemID = (Select Top 1 FoodItemID From CartIncludes
Group By FoodItemID Order By SUM(Quantity) DESC)) As 'Top Food Item In Cart',
(Select Top 1 Name As 'Food Item Name' From FoodItem
Where FoodItemID = (Select Top 1 FoodItemID From wishContains
Group By FoodItemID Order By SUM(Quantity) DESC)) AS 'Top Food Item In Wishlist';
```



	Top Food Item In Cart	Top Food Item In Wishlist
1	Kaak bread	profiteroles eclaire

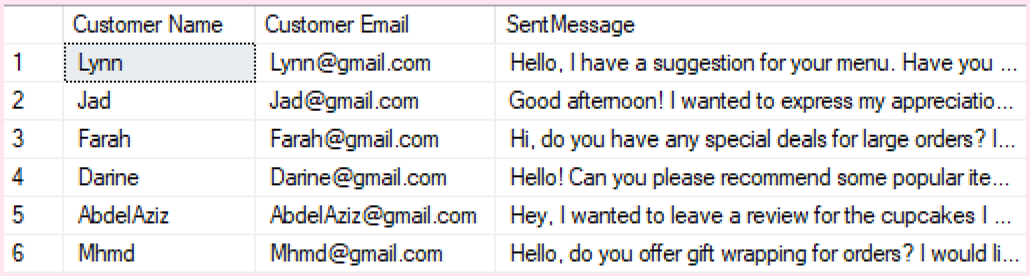
```sql
-- MESSAGES AND REVIEWS RELATED

--69. Display the total number of messages sent by each customer.
Select UserID, Count(*) AS 'Total Messages Sent' From Messages
Where UserID IN (Select UserID From Users Where RoleID=2)
Group By UserID;
```



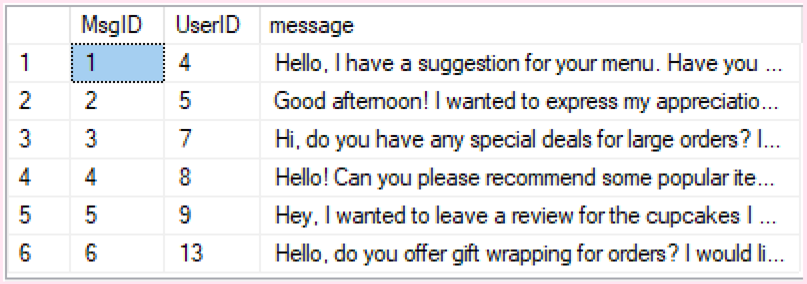
	UserID	Total Messages Sent
1	4	1
2	5	1
3	7	1
4	8	1
5	9	1
6	13	1

```sql
--70. Retrieve all customers who have sent messages to the admin, along with their sent
messages.
Select U.Name As 'Customer Name', U.email As 'Customer Email',  M.message AS SentMessage From
Users As U
Join Messages As M On U.UserID=M.UserID
Where U.RoleID=2;
```



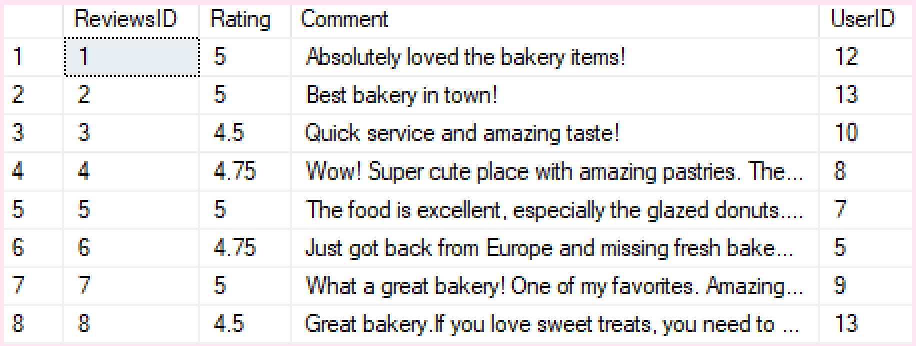
	Customer Name	Customer Email	SentMessage
1	Lynn	Lynn@gmail.com	Hello, I have a suggestion for your menu. Have you ...
2	Jad	Jad@gmail.com	Good afternoon! I wanted to express my appreciatio...
3	Farah	Farah@gmail.com	Hi, do you have any special deals for large orders? I...
4	Darine	Darine@gmail.com	Hello! Can you please recommend some popular ite...
5	AbdelAziz	AbdelAziz@gmail.com	Hey, I wanted to leave a review for the cupcakes I ...
6	Mhmd	Mhmd@gmail.com	Hello, do you offer gift wrapping for orders? I would li...

```sql
--71. Retrieve all messages sent by customers to the admin.
Select * From Messages;
```



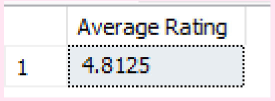
	MsgID	UserID	message
1	1	4	Hello, I have a suggestion for your menu. Have you ...
2	2	5	Good afternoon! I wanted to express my appreciatio...
3	3	7	Hi, do you have any special deals for large orders? I...
4	4	8	Hello! Can you please recommend some popular ite...
5	5	9	Hey, I wanted to leave a review for the cupcakes I ...
6	6	13	Hello, do you offer gift wrapping for orders? I would li...

```sql
--72. Retrieve all reviews left by customers.
Select * From Reviews;
```



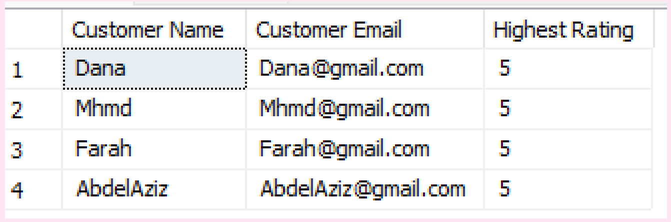
	ReviewsID	Rating	Comment	UserID
1	1	5	Absolutely loved the bakery items!	12
2	2	5	Best bakery in town!	13
3	3	4.5	Quick service and amazing taste!	10
4	4	4.75	Wow! Super cute place with amazing pastries. The...	8
5	5	5	The food is excellent, especially the glazed donuts....	7
6	6	4.75	Just got back from Europe and missing fresh bake...	5
7	7	5	What a great bakery! One of my favorites. Amazing...	9
8	8	4.5	Great bakery.If you love sweet treats, you need to ...	13

```sql
--73. Retrieve the average rating of all reviews.
Select AVG(Rating) As 'Average Rating' FROM Reviews;
```



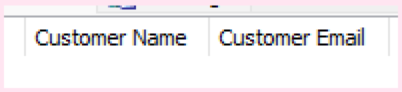
	Average Rating
1	4.8125

```sql
--74. Show the details of customers who have given the highest rating.
Select U.Name As 'Customer Name', U.Email As 'Customer Email', R.Rating As 'Highest Rating'
From Users As U
Join Reviews As R ON U.UserID = R.UserID
Where R.Rating = (Select MAX(Rating) From Reviews);
```



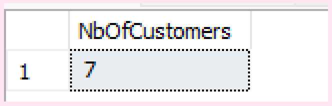
	Customer Name	Customer Email	Highest Rating
1	Dana	Dana@gmail.com	5
2	Mhmd	Mhmd@gmail.com	5
3	Farah	Farah@gmail.com	5
4	AbdelAziz	AbdelAziz@gmail.com	5

```sql
--75. List all customers who have given ratings below 3 stars.
Select U.Name As 'Customer Name', U.Email As 'Customer Email'
From Users As U, Reviews
Where U.UserID = Reviews.UserID AND Rating < 3;
```



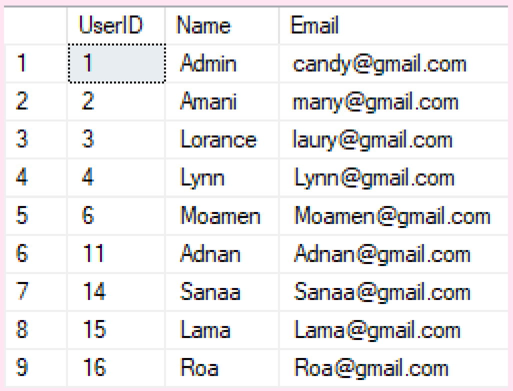
Customer Name	Customer Email

```sql
--76. List all customers who have registered but have not sent any messages to the admin.
Select Count(UserID) AS 'Number Of Customers'
From Users
WHERE Users.UserID NOT IN (SELECT UserID FROM Messages)
AND Users.RoleID = 2;
```



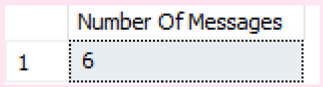
	NbOfCustomers
1	7

```sql
--77. Retrieve all customers who have not yet written a review.
Select * From Users As U
Where UserID NOT IN (Select Distinct UserID From Reviews);
```



	UserID	Name	Email
1	1	Admin	candy@gmail.com
2	2	Amani	many@gmail.com
3	3	Lorance	laury@gmail.com
4	4	Lynn	Lynn@gmail.com
5	6	Moamen	Moamen@gmail.com
6	11	Adnan	Adnan@gmail.com
7	14	Sanaa	Sanaa@gmail.com
8	15	Lama	Lama@gmail.com
9	16	Roa	Roa@gmail.com

```sql
--78. Retrieve the total number of messages sent to the admin.
Select COUNT(*) AS 'Number Of Messages' From Messages;
```



	Number Of Messages
1	6

Conclusion

To conclude, we wanted to base our Database Project on building an effective database for our online bakery website “Bake n’ take” to ensure that the business’ customers receive our delicious pastries freshly baked with minimal latency in the processing of their orders. By providing our customers with numerous functionalities we hope that we will elevate their shopping experience, while also equipping the administrators with priceless data insights that are of paramount importance for tailoring customer-specific offerings and thus boosting the bakery's sales. Not forgetting that the organized way the data will be stored in the database will indeed simplify the administrators' procedures and tasks tenfold and thus help to enhance the bakery’s overall performance. On the other hand, building the bakery’s database as a project has equipped us with a comprehensive understanding of database design and implementation processes. The journey from building an ER diagram, designing a relational schema, using DDL to develop relations and the relationships among them, then formulating and executing diverse SQL queries has certainly helped us to grasp all the needed knowledge from our Database course. Along the way we were presented with the opportunity to demonstrate the practical skills we have gained throughout each lecture and lab. And by effectively developing a functional and efficient database system we also gained experience when working on future database-related projects.