CS4347 Database Systems Final Project Deliverable 2

Healthy Recipes Generation/Meal Planning/Nutrition System

Group Members: Muhammad Siddique, Mudit Upadhyay, Anirudh Nemmani, Syed Kabir, Vaishnavi Karaguppi, Senthen Velmurugan, Arvindh Kumar Kalainathan

0. Description:

Project Title

Healthy Recipes Generation/Meal Planning/Nutrition System

GitHub Repo

https://github.com/msiddique12/Meal-Planning-Nutrition-System

The repository currently contains the implementation of our design including SQL statements for database and table creation, along with the setup of primary and foreign keys. It also houses insert, update, delete, and query operations on the database.

Team members

- Muhammad Siddique
- Mudit Upadhyay
- Anirudh Nemmani
- Syed Kabir
- Vaishnavi Karaguppi
- Senthen Velmurugan
- Arvindh Kumar Kalainathan

Delegation of Tasks for Deliverable 1 + 2

- **Muhammad Siddique** - Involved in creating the conceptual design of the database and implementing the design using SQL, creating a functional schema to manage users, dietary preferences, meal plans, and cuisine options. Involved in creating the node backend web server and integrating it with the react frontend.
- **Mudit Upadhyay -** Involved in researching background and finding related work as well as database design. Involved in frontend design.
- **Anirudh Nemmani -** Involved in creating the conceptual design of the database and implementing the design using SQL and designing the connection between the frontend and backend.
- **Syed Kabir -** Involved in creating the conceptual design of the database and the Enhanced Entity Relational model designs of the project. Worked on database normalization and justifying that it was already normalized.
- **Vaishnavi Karaguppi -** Involved in creating the conceptual design of the database and creating a functional schema of the project, and refining the react native frontend.
- **Senthen Velmurugan-** Involved in implementing the design using SQL, and Test all execution types, and built the react native frontend.

- **Arvindh Kumar Kalainathan -** Involved in creating the relational database schema, the Enhanced Entity Relational model designs of the project and the normalization of it.

Project Motivation

The motivation for this project stems from our team's collective interest in leading a healthier lifestyle without sacrificing the enjoyment of food. We all face challenges in balancing our busy lives with proper nutrition, and finding the right recipes that fit both our dietary restrictions and taste preferences can be difficult. We believe that a tool like this can simplify meal planning, not only for ourselves but for anyone looking to eat healthier while maintaining variety and cultural tastes in their meals.

Additionally, This tool could be useful for fitness enthusiasts who want to keep track of their exact calories and macronutrients, people with specific dietary restrictions, or even healthcare providers looking to offer personalized meal plans for patients. It could also be extended for use in meal prep services, fitness apps, or nutrition platforms that want to incorporate more personalized dietary options.

Project Timeline

Phase 1

1. Conceptual Design
    - Held extensive team discussion on how we wanted the design to look and what exact use cases our system could serve.
    - Created an EER (Enhanced Entity-Relationship) diagram to illustrate the design we came up with.
    - The EER diagram was continuously updated throughout the design time of this deliverable to reflect our most current plans for the system.
2. Logical Design
    - Performed ER/EER to relational mapping to create the relational schema of our system. This involved mapping our entities to relational tables, converting some of the relationships to relational tables, ETC.
3. Physical Design (SQL Implementation)
    - Created the database schema using SQL based on the relational model with referential integrity constraints enforced
    - Populated the database with initial data and performed query, update, and delete operations on the database state.

Phase 2

1. Backend Development
    - Set up Node.js server environment
    - Implemented RESTful API endpoints
    - Created database connection middleware
2. Frontend Development
    - Built React Native mobile application
    - Implemented user interface components:
        - Querying interface
        - Insertion interface
        - Deletion interface
        - Update interface
        - Close application interface
3. Integration & Testing
    - Connected frontend to backend API
    - Implemented data flow between layers
    - Performed end-to-end testing

Current Status

- Functional full-stack application
- Complete database integration
- Operational CRUD functionality
- Ready for demonstration and future enhancements

1. Introduction:

Our project, a meal planning app, focuses on simplifying daily meal preparation by generating personalized meal plans. We chose this topic to address the growing need for individuals to maintain a healthy diet despite their busy schedules. With the rise of diverse dietary requirements and preferences, people often struggle to plan meals that cater to their unique needs. By developing this app, we aim to contribute to the field of security, particularly regarding data privacy and dietary data protection, ensuring that users' personal information, such as their calorie intake, dietary restrictions, and meal preferences, is kept secure.

Our app's database is designed to store user profiles, meal plans, recipes, and nutritional information securely. The reason we chose this particular database design is to create a comprehensive, scalable solution capable of handling personalized meal

data while maintaining strict data privacy. Each user profile includes details such as daily calorie goals, dietary preferences, and restrictions, allowing the app to generate non-repetitive, customized meal plans.

The real-world problem this design helps solve is the increasing demand for personalized, accessible, and secure nutritional planning. Many apps store data but do not use it effectively to cater to individual preferences or to protect it sufficiently from breaches. By focusing on data protection, we ensure that users can trust the app with their sensitive health-related information while enjoying tailored meal plans that help them meet their nutritional goals.

2. Background and Related Work:

After looking over several other sources, we found 2 other applications that are working towards the same problem statement. **HealthifyMe** is a comprehensive health and fitness app that empowers users to opt for healthy food choices. It offers a vast database of healthy recipes catering to various dietary restrictions, allowing users to customize their meal schedules by day or week. This all-in-one approach emphasizes holistic wellness but may lack the depth of automated meal planning. **MealBoard** is focusing on streamlining meal prep by combining recipe storage, meal scheduling, and shopping management in one app. Unlike HealthifyMe, which leans heavily on nutritional guidance, MealBoard provides a straightforward meal planning experience. Our app, **MealPlanning** differentiates itself by automatically generating personalized meal plans for breakfast, lunch, dinner, and snacks based on user-inputted calorie goals and dietary preferences, ensuring a balanced and varied diet while minimizing repetitive meals.

3. Design & Implementation (Phase II):

3.1. Normalization of EER Conceptual Data Model:

1NF Verification:

In the User table, we have Cuisine preferences as a multi-valued attribute. But from the previous rules which we applied 8A, the multivalued attributes are separated from the base table. This follows 1NF form since there are no other multivalued or composite attributes or nested relations.

2NF Verification:

For 2NF, we need to look for partial functional dependencies. In the User table, there are no partial dependencies since all the non-prime attributes depend on the primary key. In the Nutrients table, since it is a weak entity, we have recipe_ID as a foreign key paired along with the partial key to make our primary key combination. All the non-prime

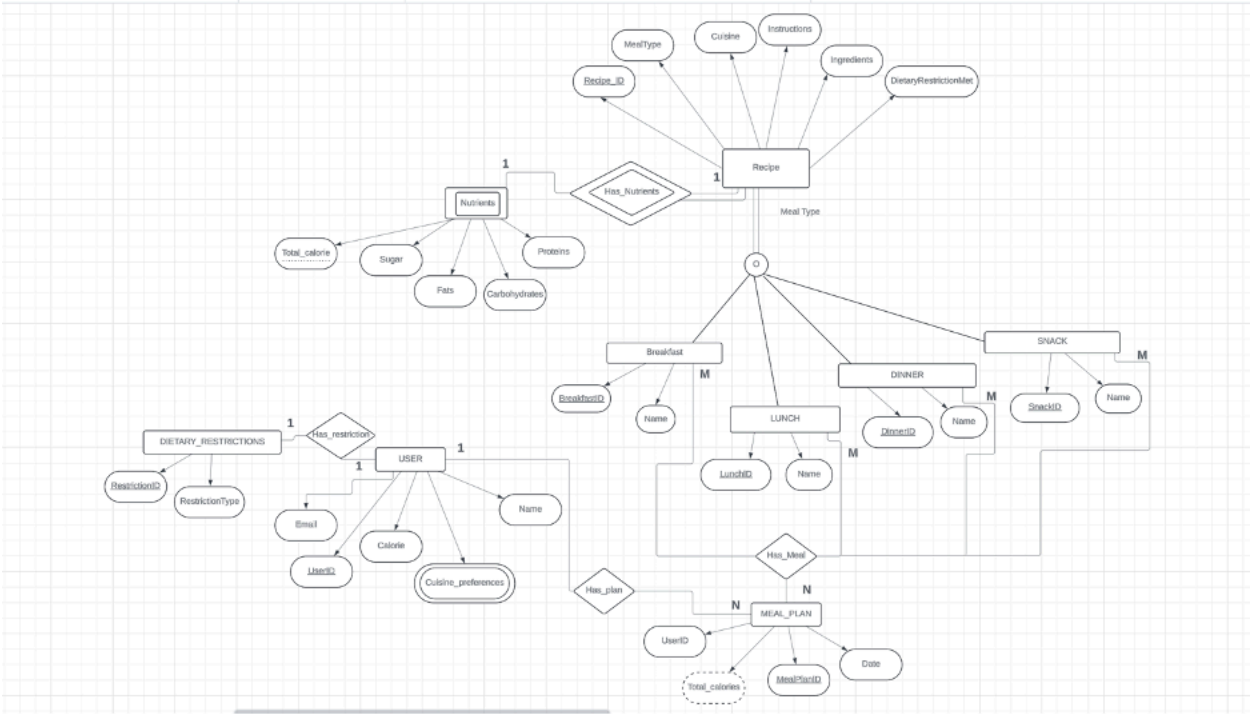
attributes are dependent on this combination of primary keys. In Recipe table, all the non-prime keys are functionally dependent on the primary key (Recipe_ID). The subclass tables (Breakfast, Lunch, Dinner and Snack) follow 2NF as they are derived from recipes and don't have any partial functional dependencies. This is the reason why we split them into subclasses on last deliverable to follow the rules. Meal plan table follows 2NF (no partial FDs). Has_meal is our final answer or where we get the meal plan from. We made a wide combination of primary key as they would make it unique for each person and by doing that also satisfies the 2NF protocol of that table as there are no partial dependencies. Finally, the Dietary_Restrictions table has no partial FDs since there are only 2 attributes and one of them is the primary key (Restriction_ID).

Therefore, our relational schema follows 2NF.

3NF Verification:

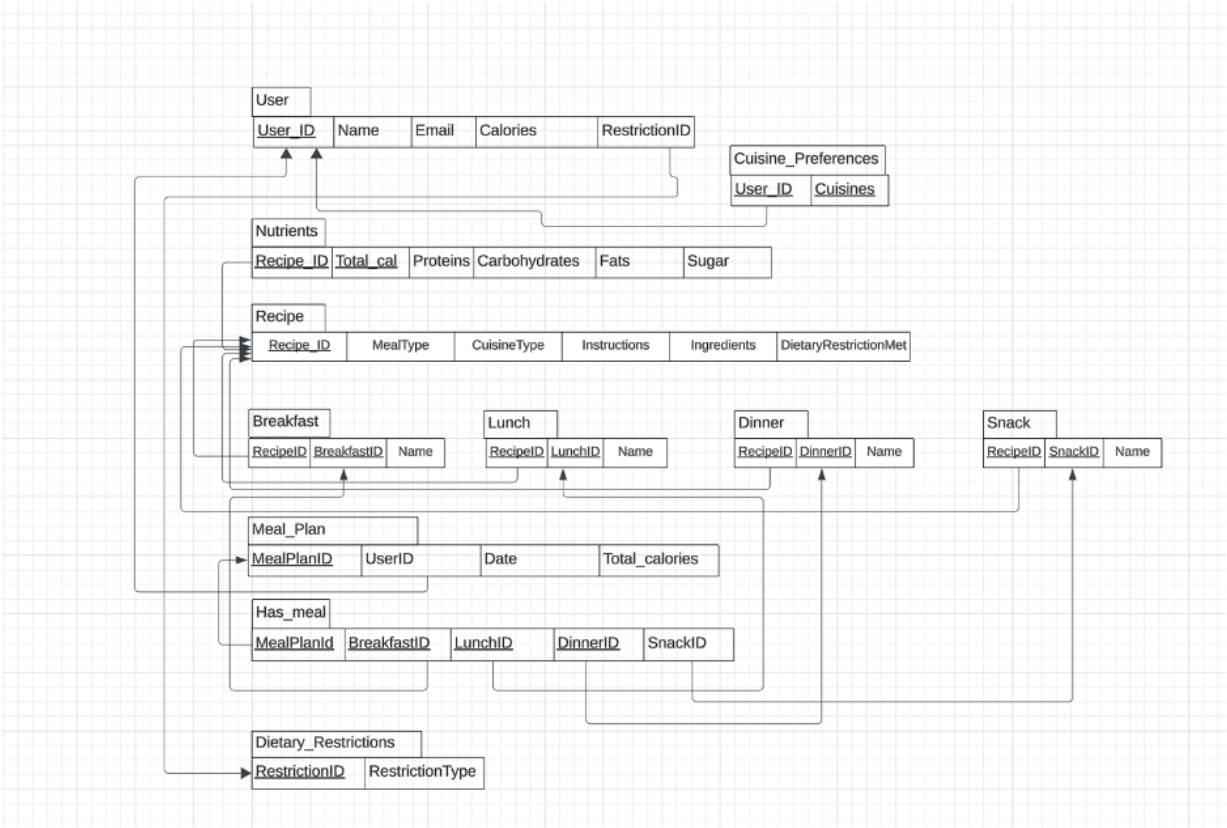
For 3NF, we need to look for transitive dependencies. There was a transitive dependency which was solved in the last deliverable, where the Dietary_Restriction was included in the User table. This would cause transitive FD since User_ID -> Restriction ID, Restriction_ID -> RestrictionType and therefore User_ID -> RestrictionType. This is why a new Dietary_Restriction table was made separate from User table with RestrictionID as our primary key. Looking at the other tables, since each of them have only one functional dependency (From primary key combination to non-prime attributes), this removes any chance of transitive FDs as that requires 2 FDs.

Therefore, our relational schema follows 3NF.



3.2. Relational Data Model Design Using Normalized EER Diagram:

There is no change in our relational model since the EER diagram was already normalized in deliverable 1 as shown in section 3.1 and so no change in the EER to relational mapping.



3.3. Create your Normalized Database and Populate:

There is no change in our database since our design was already normalized in deliverable 1 as shown in section 3.1 and so no change in the SQL database.

DIETARY RESTRICTIONS:

```sql
-- Create table for Dietary Restrictions
CREATE TABLE IF NOT EXISTS Dietary_Restrictions (
    RestrictionID INT PRIMARY KEY,
    RestrictionType VARCHAR(255) NOT NULL
);
```

```sql
-- Populating Dietary_Restrictions table
INSERT INTO Dietary_Restrictions (RestrictionID, RestrictionType) VALUES
(1, 'Vegetarian'),
(2, 'Vegan'),
(3, 'Halal'),
(4, 'Kosher'),
(5, 'Gluten-Free'),
(6, 'Dairy-Free'),
(7, 'Nut-Free'),
(8, 'Pescatarian'),
(9, 'Low-Carb'),
(10, 'Paleo');
```

RestrictionID	RestrictionType
1	Vegetarian
2	Vegan
3	Halal
4	Kosher
5	Gluten-Free
6	Dairy-Free
7	Nut-Free
8	Pescatarian
9	Low-Carb
10	Paleo
NULL	NULL

USERS:

```sql
-- Create table for Users
CREATE TABLE IF NOT EXISTS User (
    User_ID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL UNIQUE,
    Calories INT,
    RestrictionID INT,
    FOREIGN KEY (RestrictionID) REFERENCES Dietary_Restrictions(RestrictionID)
);
```

```sql
-- Populating User table
INSERT INTO User (User_ID, Name, Email, Calories, RestrictionID) VALUES
(1, 'John Doe', 'john.doe@example.com', 2000, 1),
(2, 'Jane Smith', 'jane.smith@example.com', 1500, 2),
(3, 'Ahmed Ali', 'ahmed.ali@example.com', 1800, 3),
(4, 'Sarah Cohen', 'sarah.cohen@example.com', 1700, 4),
(5, 'Emily Johnson', 'emily.johnson@example.com', 1600, 5),
(6, 'David Lee', 'david.lee@example.com', 2200, 6),
(7, 'Anna Gupta', 'anna.gupta@example.com', 1900, 7),
(8, 'Paul Martin', 'paul.martin@example.com', 2100, 8),
(9, 'Olivia Taylor', 'olivia.taylor@example.com', 1400, 9),
(10, 'Carlos Ramirez', 'carlos.ramirez@example.com', 2000, 10);
```

User_ID	Name	Email	Calories	RestrictionID
1	John Doe	john.doe@example.com	2000	1
2	Jane Smith	jane.smith@example.com	1500	2
3	Ahmed Ali	ahmed.ali@example.com	1800	3
4	Sarah Cohen	sarah.cohen@example.com	1700	4
5	Emily Johnson	emily.johnson@example.com	1600	5
6	David Lee	david.lee@example.com	2200	6
7	Anna Gupta	anna.gupta@example.com	1900	7
8	Paul Martin	paul.martin@example.com	2100	8
9	Olivia Taylor	olivia.taylor@example.com	1400	9
10	Carlos Ramirez	carlos.ramirez@example.com	2000	10
NULL	NULL	NULL	NULL	NULL

Cuisine Preferences:

```sql
-- Create table for Cuisine Preferences
CREATE TABLE IF NOT EXISTS Cuisine_Preferences(
    User_ID INT,
    Cuisine VARCHAR(255),
    PRIMARY KEY (User_ID, Cuisine),
    FOREIGN KEY (User_ID) REFERENCES User(User_ID)
);
```

```sql
-- Populating Cuisine_Preferences table
INSERT INTO Cuisine_Preferences (User_ID, Cuisine) VALUES
(1, 'Indian'),
(1, 'Italian'),
(2, 'Mediterranean'),
(2, 'Mexican'),
(3, 'Middle Eastern'),
(3, 'South Asian'),
(4, 'Jewish'),
(5, 'French'),
(6, 'Chinese'),
(7, 'Thai'),
(8, 'Japanese'),
(9, 'Keto'),
(10, 'Paleo');
```

User_ID	Cuisine
1	Indian
1	Italian
2	Mediterranean
2	Mexican
3	Middle Eastern
3	South Asian
4	Jewish
5	French
6	Chinese
7	Thai
8	Japanese
9	Keto
10	Paleo
NULL	NULL

Recipe:

```sql
-- Create table for Recipe
CREATE TABLE IF NOT EXISTS Recipe (
    Recipe_ID INT PRIMARY KEY,
    MealType VARCHAR(255) NOT NULL,
    CuisineType VARCHAR(255),
    Instructions TEXT,
    Ingredients TEXT,
    DietaryRestrictionMet BOOLEAN
);
```

```sql
-- Populating Recipe table
INSERT INTO Recipe (Recipe_ID, MealType, CuisineType, Instructions, Ingredients, DietaryRestrictionMet) VALUES
(1, 'Lunch', 'Indian', 'Cook roti and curry', 'wheat flour, vegetables, spices', TRUE),
(2, 'Dinner', 'Italian', 'Bake pizza with cheese and sauce', 'wheat flour, cheese, tomato sauce', FALSE),
(3, 'Lunch', 'Indian', 'Cook biryani', 'rice, chicken, spices', FALSE),
(4, 'Breakfast', 'American', 'Prepare scrambled eggs', 'eggs, butter, salt', FALSE),
(5, 'Snack', 'American', 'Prepare french fries', 'potatoes, oil, salt', FALSE),
(6, 'Dinner', 'Indian', 'Cook dal with spices', 'lentils, turmeric, cumin', TRUE),
(7, 'Lunch', 'Mexican', 'Make tacos', 'corn tortillas, meat, cheese, lettuce', FALSE),
(8, 'Snack', 'Italian', 'Bake garlic bread', 'bread, garlic, butter', FALSE),
(9, 'Breakfast', 'French', 'Make croissants', 'flour, butter, sugar', FALSE),
(10, 'Dinner', 'Mediterranean', 'Grill fish with veggies', 'fish, olive oil, vegetables', TRUE);
```

Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryRestrictionMet
1	Lunch	Indian	Cook roti and curry	wheat flour, vegetables, spices	1
2	Dinner	Italian	Bake pizza with cheese and sauce	wheat flour, cheese, tomato sauce	0
3	Lunch	Indian	Cook biryani	rice, chicken, spices	0
4	Breakfast	American	Prepare scrambled eggs	eggs, butter, salt	0
5	Snack	American	Prepare french fries	potatoes, oil, salt	0
6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
7	Lunch	Mexican	Make tacos	corn tortillas, meat, cheese, lettuce	0
8	Snack	Italian	Bake garlic bread	bread, garlic, butter	0
9	Breakfast	French	Make croissants	flour, butter, sugar	0
10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1
NULL	NULL	NULL	NULL	NULL	NULL

Nutrients:

```sql
-- Create table for Nutrients
CREATE TABLE IF NOT EXISTS Nutrients (
    Recipe_ID INT,
    Total_cal INT,
    Proteins DECIMAL(5,2),
    Carbohydrates DECIMAL(5,2),
    Fats DECIMAL(5,2),
    Sugar DECIMAL(5,2),
    PRIMARY KEY (Recipe_ID, Total_cal),
    FOREIGN KEY (Recipe_ID) REFERENCES Recipe(Recipe_ID)
);
```

```sql
-- Populating Nutrients table
INSERT INTO Nutrients (Recipe_ID, Total_cal, Proteins, Carbohydrates, Fats, Sugar) VALUES
(1, 400, 10.5, 60.3, 8.2, 2.1),
(2, 800, 20.3, 90.1, 30.5, 5.2),
(3, 650, 25.2, 70.5, 15.8, 3.4),
(4, 300, 20.0, 2.0, 25.0, 1.0),
(5, 500, 5.2, 50.5, 25.3, 0.5),
(6, 450, 12.5, 60.0, 10.0, 2.5),
(7, 550, 15.8, 65.3, 20.2, 3.0),
(8, 350, 8.5, 40.0, 10.5, 2.5),
(9, 600, 12.3, 55.5, 35.0, 6.0),
(10, 480, 25.0, 20.0, 18.0, 0.5);
```

Recipe_ID	Total_cal	Proteins	Carbohydrates	Fats	Sugar
1	400	10.50	60.30	8.20	2.10
2	800	20.30	90.10	30.50	5.20
3	650	25.20	70.50	15.80	3.40
4	300	20.00	2.00	25.00	1.00
5	500	5.20	50.50	25.30	0.50
6	450	12.50	60.00	10.00	2.50
7	550	15.80	65.30	20.20	3.00
8	350	8.50	40.00	10.50	2.50
9	600	12.30	55.50	35.00	6.00
10	480	25.00	20.00	18.00	0.50
NULL	NULL	NULL	NULL	NULL	NULL

Breakfast:

```sql
-- Create table for Breakfast
CREATE TABLE IF NOT EXISTS Breakfast (
    RecipeID INT,
    BreakfastID INT PRIMARY KEY,  -- Make BreakfastID a primary key
    Name VARCHAR(255),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(Recipe_ID)
);
```

```sql
-- Populating Breakfast table
INSERT INTO Breakfast (RecipeID, BreakfastID, Name) VALUES
(4, 1, 'Scrambled Eggs'),
(9, 2, 'Croissants'),
(10, 3, 'Fish with veggies'),
(6, 4, 'Dal with spices'),
(1, 5, 'Roti and curry'),
(2, 6, 'Pizza'),
(5, 7, 'French fries'),
(8, 8, 'Garlic Bread'),
(7, 9, 'Tacos'),
(3, 10, 'Biryani');
```

RecipeID	BreakfastID	Name
4	1	Scrambled Eggs
9	2	Croissants
10	3	Fish with veggies
6	4	Dal with spices
1	5	Roti and curry
2	6	Pizza
5	7	French fries
8	8	Garlic Bread
7	9	Tacos
3	10	Biryani
NULL	NULL	NULL

Lunch:

```sql
-- Create table for Lunch
CREATE TABLE IF NOT EXISTS Lunch (
    RecipeID INT,
    LunchID INT PRIMARY KEY,  -- Make LunchID a primary key
    Name VARCHAR(255),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(Recipe_ID)
);
```

```sql
-- Populating Lunch table
INSERT INTO Lunch (RecipeID, LunchID, Name) VALUES
(1, 1, 'Roti with Curry'),
(3, 2, 'Chicken Biryani'),
(7, 3, 'Tacos'),
(5, 4, 'French Fries'),
(6, 5, 'Dal with rice'),
(9, 6, 'Croissants with jam'),
(8, 7, 'Garlic bread with dipping'),
(10, 8, 'Grilled fish'),
(2, 9, 'Cheese Pizza'),
(4, 10, 'Scrambled eggs');
```

RecipeID	LunchID	Name
1	1	Roti with Curry
3	2	Chicken Biryani
7	3	Tacos
5	4	French Fries
6	5	Dal with rice
9	6	Croissants with jam
8	7	Garlic bread with dipping
10	8	Grilled fish
2	9	Cheese Pizza
4	10	Scrambled eggs
NULL	NULL	NULL

Dinner:

```sql
-- Create table for Dinner
CREATE TABLE IF NOT EXISTS Dinner (
    RecipeID INT,
    DinnerID INT PRIMARY KEY,  -- Make DinnerID a primary key
    Name VARCHAR(255),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(Recipe_ID)
);
```

```sql
-- Populating Dinner table
INSERT INTO Dinner (RecipeID, DinnerID, Name) VALUES
(3, 1, 'Mutton Biryani'),
(1, 2, 'Roti with Curry'),
(4, 3, 'Scrambled Eggs'),
(2, 4, 'Cheese Pizza'),
(6, 5, 'Dal with spices'),
(7, 6, 'Tacos'),
(5, 7, 'French fries'),
(9, 8, 'Croissants'),
(8, 9, 'Garlic bread'),
(10, 10, 'Fish with veggies');
```

RecipeID	DinnerID	Name
3	1	Mutton Biryani
1	2	Roti with Curry
4	3	Scrambled Eggs
2	4	Cheese Pizza
6	5	Dal with spices
7	6	Tacos
5	7	French fries
9	8	Croissants
8	9	Garlic bread
10	10	Fish with veggies
NULL	NULL	NULL

Snack:

```sql
-- Create table for Snack
CREATE TABLE IF NOT EXISTS Snack (
    RecipeID INT,
    SnackID INT PRIMARY KEY,  -- Make SnackID a primary key
    Name VARCHAR(255),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(Recipe_ID)
);
```

```sql
-- Populating Snack table
INSERT INTO Snack (RecipeID, SnackID, Name) VALUES
(5, 1, 'French Fries'),
(8, 2, 'Garlic Bread'),
(7, 3, 'Tacos'),
(9, 4, 'Croissants'),
(1, 5, 'Roti and Curry'),
(3, 6, 'Biryani'),
(10, 7, 'Grilled Fish'),
(2, 8, 'Cheese Pizza'),
(6, 9, 'Dal with spices'),
(4, 10, 'Scrambled Eggs');
```

RecipeID	SnackID	Name
5	1	French Fries
8	2	Garlic Bread
7	3	Tacos
9	4	Croissants
1	5	Roti and Curry
3	6	Biryani
10	7	Grilled Fish
2	8	Cheese Pizza
6	9	Dal with spices
4	10	Scrambled Eggs
NULL	NULL	NULL

Meal_Plan:

```sql
-- Create table for Meal_Plan
CREATE TABLE IF NOT EXISTS Meal_Plan (
    MealPlanID INT PRIMARY KEY,
    Date DATE NOT NULL,
    UserID INT,
    Total_calories INT,
    FOREIGN KEY (UserID) REFERENCES User(User_ID)
);
```

```sql
-- Populating Meal_Plan table
INSERT INTO Meal_Plan (MealPlanID, Date, UserID, Total_calories) VALUES
(1, '2024-10-15', 1, 2000),
(2, '2024-10-16', 2, 1500),
(3, '2024-10-17', 3, 1800),
(4, '2024-10-18', 4, 1700),
(5, '2024-10-19', 5, 1600),
(6, '2024-10-20', 6, 2200),
(7, '2024-10-21', 7, 1900),
(8, '2024-10-22', 8, 2100),
(9, '2024-10-23', 9, 1400),
(10, '2024-10-24', 10, 2000);
```

MealPlanID	Date	UserID	Total_calories
1	2024-10-15	1	2000
2	2024-10-16	2	1500
3	2024-10-17	3	1800
4	2024-10-18	4	1700
5	2024-10-19	5	1600
6	2024-10-20	6	2200
7	2024-10-21	7	1900
8	2024-10-22	8	2100
9	2024-10-23	9	1400
10	2024-10-24	10	2000
NULL	NULL	NULL	NULL

Has_Meal:

```sql
-- Create table for Has_Meal (to associate Meal_Plan with meals)
CREATE TABLE IF NOT EXISTS Has_meal (
    MealPlanId INT,
    BreakfastID INT,
    LunchID INT,
    DinnerID INT,
    SnackID INT,
    PRIMARY KEY (MealPlanId, BreakfastID, LunchID, DinnerID, SnackID),
    FOREIGN KEY (MealPlanId) REFERENCES Meal_Plan(MealPlanID),
    FOREIGN KEY (BreakfastID) REFERENCES Breakfast(BreakfastID),
    FOREIGN KEY (LunchID) REFERENCES Lunch(LunchID),
    FOREIGN KEY (DinnerID) REFERENCES Dinner(DinnerID),
    FOREIGN KEY (SnackID) REFERENCES Snack(SnackID)
);
```

```sql
-- Populating Has_Meal table
INSERT INTO Has_meal (MealPlanId, BreakfastID, LunchID, DinnerID, SnackID) VALUES
(1, 1, 1, 1, 1),
(2, 2, 2, 2, 2),
(3, 3, 3, 3, 3),
(4, 4, 4, 4, 4),
(5, 5, 5, 5, 5),
(6, 6, 6, 6, 6),
(7, 7, 7, 7, 7),
(8, 8, 8, 8, 8),
(9, 9, 9, 9, 9),
(10, 10, 10, 10, 10);
```

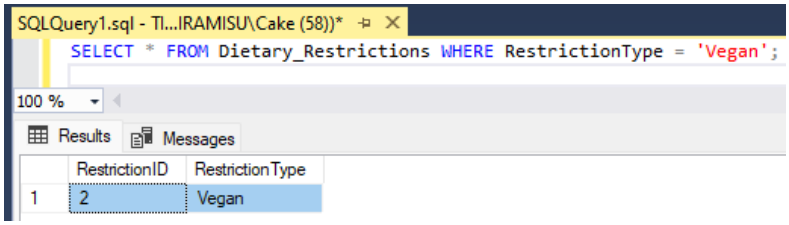
MealPlanId	BreakfastID	LunchID	DinnerID	SnackID
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4
5	5	5	5	5
6	6	6	6	6
7	7	7	7	7
8	8	8	8	8
9	9	9	9	9
10	10	10	10	10
NULL	NULL	NULL	NULL	NULL

3.4. Database Query Execution on your Normalized Database (from inside your SQL client):

1. **Dietary_Restrictions**

Query:

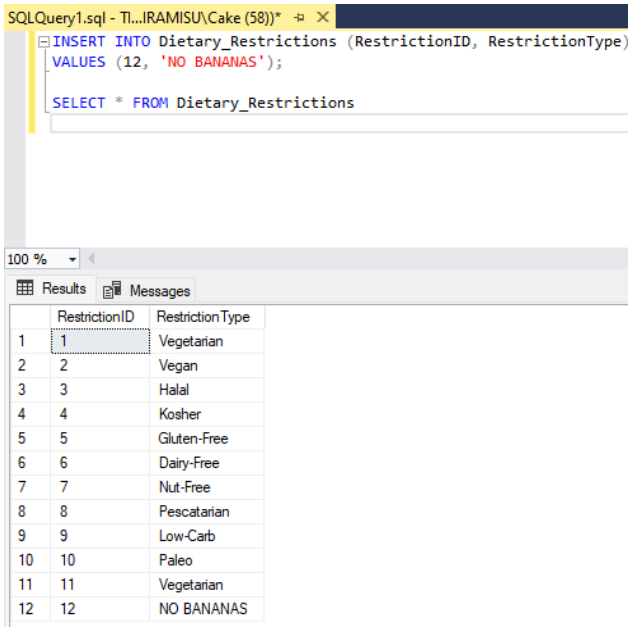
SELECT * FROM Dietary_Restrictions WHERE RestrictionType = 'Vegan';



Insert:

INSERT INTO Dietary_Restrictions (RestrictionID, RestrictionType)
VALUES (12, 'NO BANANAS');

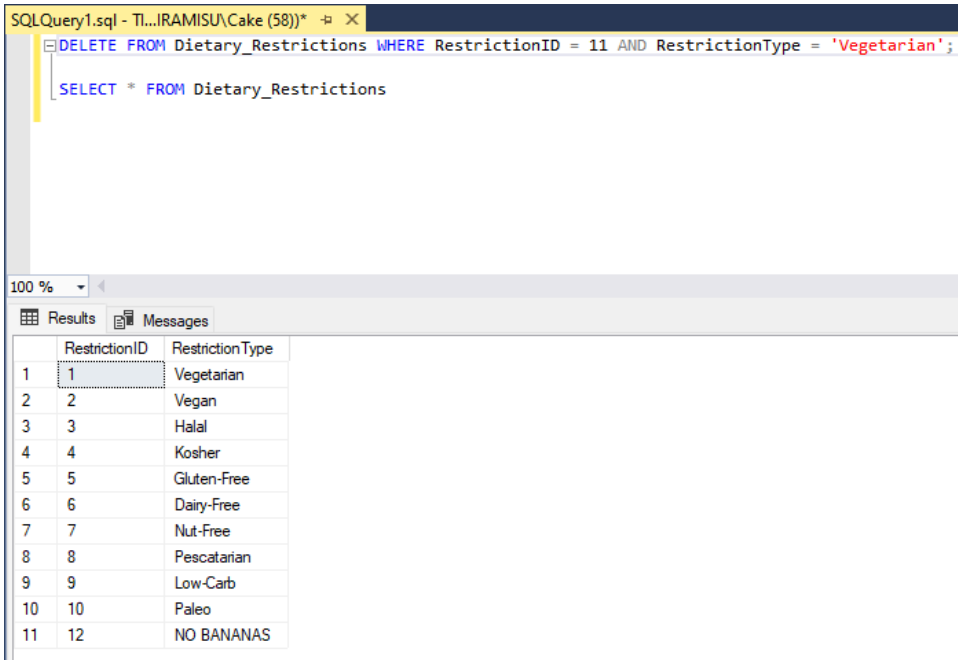
SELECT * FROM Dietary_Restrictions



	RestrictionID	RestrictionType
1	1	Vegetarian
2	2	Vegan
3	3	Halal
4	4	Kosher
5	5	Gluten-Free
6	6	Dairy-Free
7	7	Nut-Free
8	8	Pescatarian
9	9	Low-Carb
10	10	Paleo
11	11	Vegetarian
12	12	NO BANANAS

Delete:

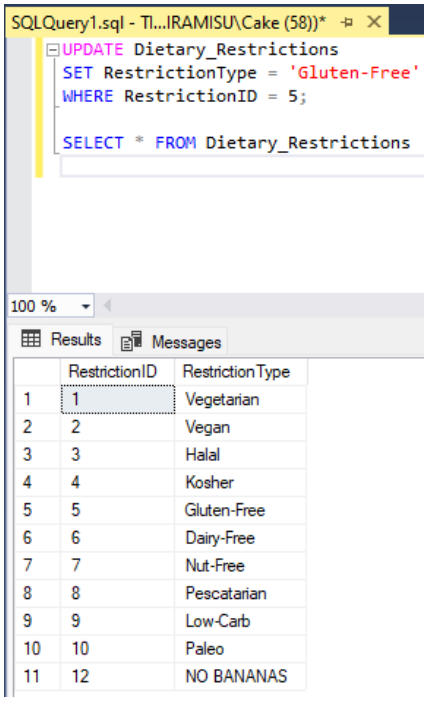
DELETE FROM Dietary_Restrictions WHERE RestrictionID = 11 AND RestrictionType = 'Vegetarian';



	RestrictionID	RestrictionType
1	1	Vegetarian
2	2	Vegan
3	3	Halal
4	4	Kosher
5	5	Gluten-Free
6	6	Dairy-Free
7	7	Nut-Free
8	8	Pescatarian
9	9	Low-Carb
10	10	Paleo
11	12	NO BANANAS

Update:

UPDATE Dietary_Restrictions
SET RestrictionType = 'Gluten-Free'
WHERE RestrictionID = 5;

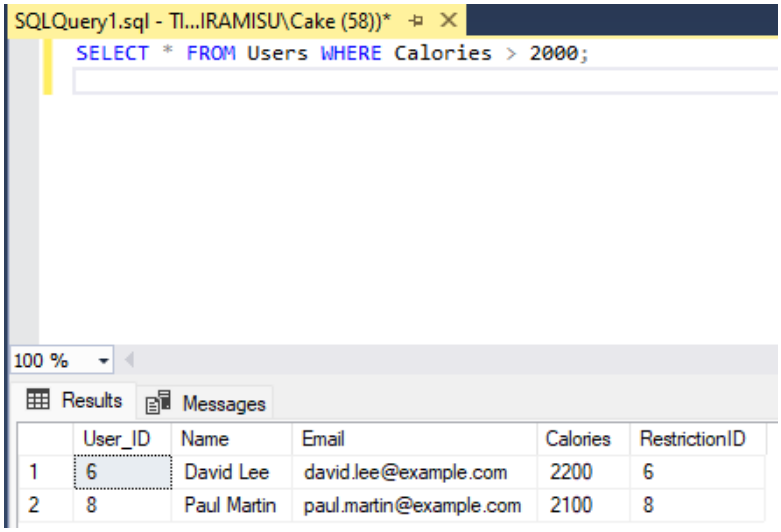


	RestrictionID	RestrictionType
1	1	Vegetarian
2	2	Vegan
3	3	Halal
4	4	Kosher
5	5	Gluten-Free
6	6	Dairy-Free
7	7	Nut-Free
8	8	Pescatarian
9	9	Low-Carb
10	10	Paleo
11	12	NO BANANAS

2. Users

Query:

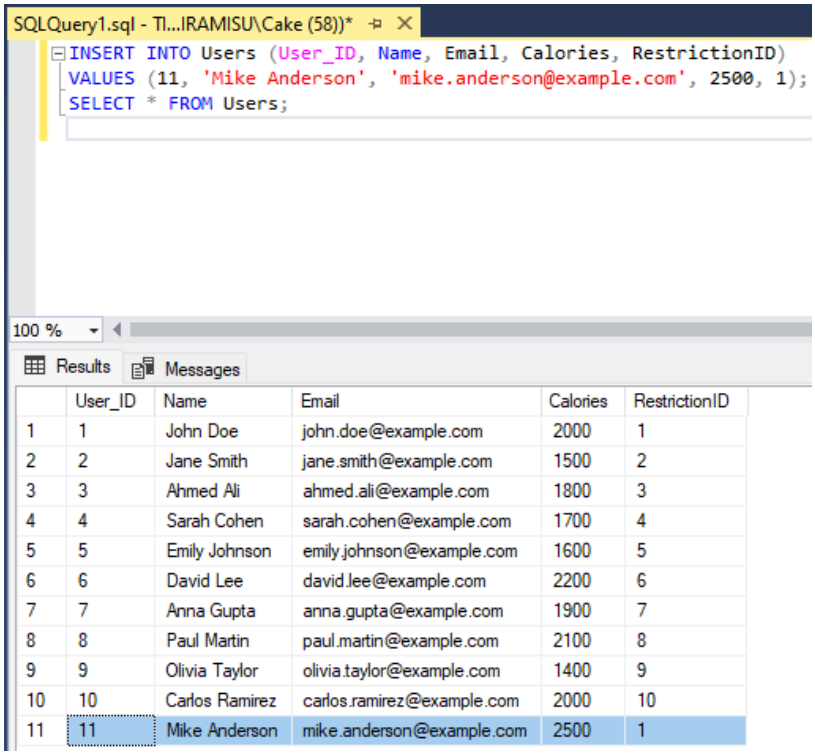
SELECT * FROM Users WHERE Calories > 2000;



	User_ID	Name	Email	Calories	RestrictionID
1	6	David Lee	david.lee@example.com	2200	6
2	8	Paul Martin	paul.martin@example.com	2100	8

Insert:

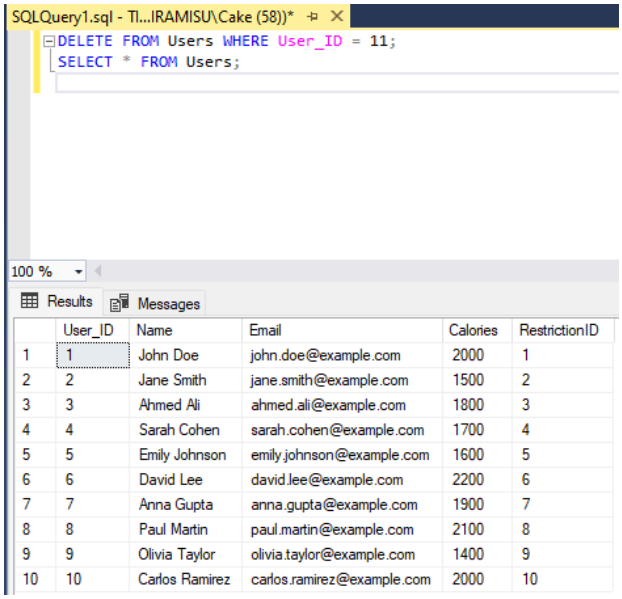
INSERT INTO Users (User_ID, Name, Email, Calories, RestrictionID)
VALUES (11, 'Mike Anderson', 'mike.anderson@example.com', 2500, 1);
SELECT * FROM Users;



	User_ID	Name	Email	Calories	RestrictionID
1	1	John Doe	john.doe@example.com	2000	1
2	2	Jane Smith	jane.smith@example.com	1500	2
3	3	Ahmed Ali	ahmed.ali@example.com	1800	3
4	4	Sarah Cohen	sarah.cohen@example.com	1700	4
5	5	Emily Johnson	emily.johnson@example.com	1600	5
6	6	David Lee	david.lee@example.com	2200	6
7	7	Anna Gupta	anna.gupta@example.com	1900	7
8	8	Paul Martin	paul.martin@example.com	2100	8
9	9	Olivia Taylor	olivia.taylor@example.com	1400	9
10	10	Carlos Ramirez	carlos.ramirez@example.com	2000	10
11	11	Mike Anderson	mike.anderson@example.com	2500	1

Delete:

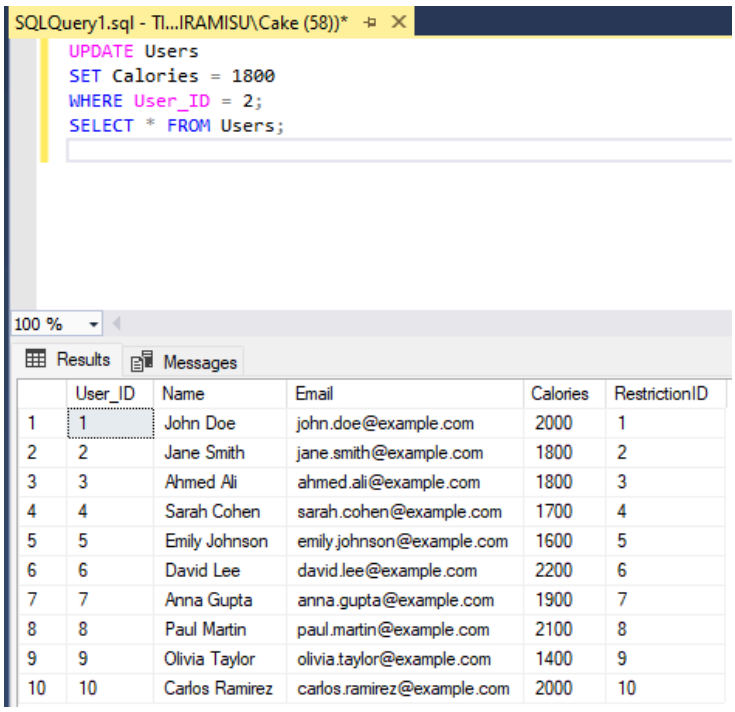
DELETE FROM Users WHERE User_ID = 11;
SELECT * FROM Users;



	User_ID	Name	Email	Calories	RestrictionID
1	1	John Doe	john.doe@example.com	2000	1
2	2	Jane Smith	jane.smith@example.com	1500	2
3	3	Ahmed Ali	ahmed.ali@example.com	1800	3
4	4	Sarah Cohen	sarah.cohen@example.com	1700	4
5	5	Emily Johnson	emily.johnson@example.com	1600	5
6	6	David Lee	david.lee@example.com	2200	6
7	7	Anna Gupta	anna.gupta@example.com	1900	7
8	8	Paul Martin	paul.martin@example.com	2100	8
9	9	Olivia Taylor	olivia.taylor@example.com	1400	9
10	10	Carlos Ramirez	carlos.ramirez@example.com	2000	10

Update:

UPDATE Users
SET Calories = 1800
WHERE User_ID = 2;



	User_ID	Name	Email	Calories	RestrictionID
1	1	John Doe	john.doe@example.com	2000	1
2	2	Jane Smith	jane.smith@example.com	1800	2
3	3	Ahmed Ali	ahmed.ali@example.com	1800	3
4	4	Sarah Cohen	sarah.cohen@example.com	1700	4
5	5	Emily Johnson	emily.johnson@example.com	1600	5
6	6	David Lee	david.lee@example.com	2200	6
7	7	Anna Gupta	anna.gupta@example.com	1900	7
8	8	Paul Martin	paul.martin@example.com	2100	8
9	9	Olivia Taylor	olivia.taylor@example.com	1400	9
10	10	Carlos Ramirez	carlos.ramirez@example.com	2000	10

3. Cuisine_Preferences

Query:

SELECT * FROM Cuisine_Preferences WHERE Cuisine = 'Indian';

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
SELECT * FROM Cuisine_Preferences WHERE Cuisine = 'Indian';
```

100 %

Results | Messages

	User_ID	Cuisine
1	1	Indian

Insert:

INSERT INTO Cuisine_Preferences (User_ID, Cuisine)
VALUES (2, 'Japanese');
SELECT * FROM Cuisine_Preferences;

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
INSERT INTO Cuisine_Preferences (User_ID, Cuisine)
VALUES (2, 'Japanese');
SELECT * FROM Cuisine_Preferences;
```

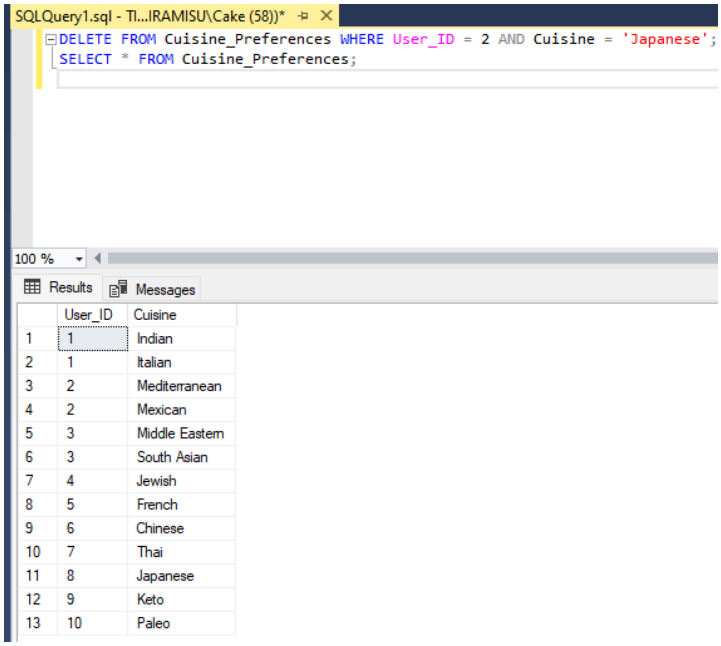
100 %

Results | Messages

	User_ID	Cuisine
1	1	Indian
2	1	Italian
3	2	Japanese
4	2	Mediterranean
5	2	Mexican
6	3	Middle Eastern
7	3	South Asian
8	4	Jewish
9	5	French
10	6	Chinese
11	7	Thai
12	8	Japanese
13	9	Keto
14	10	Paleo

Delete:

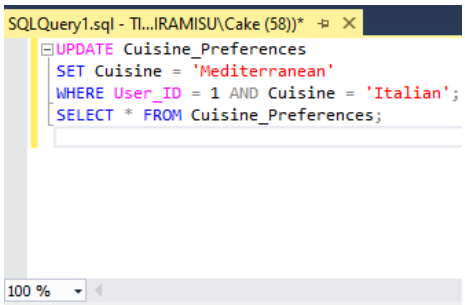
DELETE FROM Cuisine_Preferences WHERE User_ID = 2 AND Cuisine = 'Japanese';
SELECT * FROM Cuisine_Preferences;



	User_ID	Cuisine
1	1	Indian
2	1	Italian
3	2	Mediterranean
4	2	Mexican
5	3	Middle Eastern
6	3	South Asian
7	4	Jewish
8	5	French
9	6	Chinese
10	7	Thai
11	8	Japanese
12	9	Keto
13	10	Paleo

Update:

UPDATE Cuisine_Preferences
SET Cuisine = 'Mediterranean'
WHERE User_ID = 1 AND Cuisine = 'Italian';
SELECT * FROM Cuisine_Preferences;



	User_ID	Cuisine
1	1	Indian
2	1	Mediterranean
3	2	Mediterranean
4	2	Mexican
5	3	Middle Eastern
6	3	South Asian
7	4	Jewish
8	5	French
9	6	Chinese
10	7	Thai
11	8	Japanese
12	9	Keto
13	10	Paleo

1. **Recipe**

Query:

SELECT * FROM Cuisine_Preferences WHERE Cuisine = 'Indian';

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
SELECT * FROM Recipe WHERE MealType = 'Dinner' AND DietaryRestrictionMet = 1;
```

100 %

Results | Messages

	Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryRestrictionMet
1	6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
2	10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1

Insert:

INSERT INTO Recipe (Recipe_ID, MealType, CuisineType, Instructions, Ingredients, DietaryRestrictionMet)
VALUES (11, 'Snack', 'Greek', 'Mix yogurt with honey', 'yogurt, honey', 1);
SELECT * FROM Recipe;

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
INSERT INTO Recipe (Recipe_ID, MealType, CuisineType, Instructions, Ingredients, DietaryRestrictionMet)
VALUES (11, 'Snack', 'Greek', 'Mix yogurt with honey', 'yogurt, honey', 1);
SELECT * FROM Recipe;
```

100 %

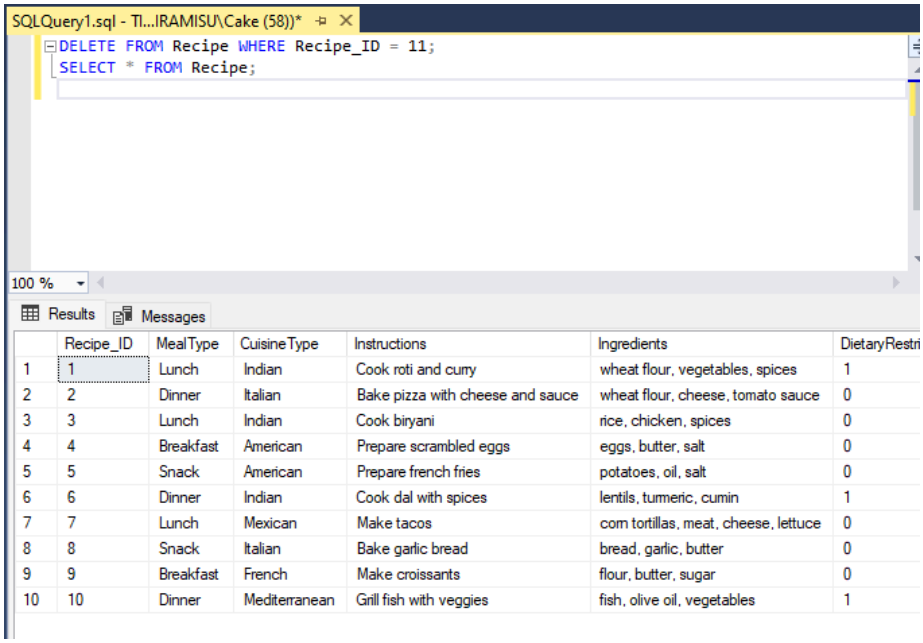
Results | Messages

	Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryRestrictionMet
1	1	Lunch	Indian	Cook roti and curry	wheat flour, vegetables, spices	1
2	2	Dinner	Italian	Bake pizza with cheese and sauce	wheat flour, cheese, tomato sauce	0
3	3	Lunch	Indian	Cook biryani	rice, chicken, spices	0
4	4	Breakfast	American	Prepare scrambled eggs	eggs, butter, salt	0
5	5	Snack	American	Prepare french fries	potatoes, oil, salt	0
6	6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
7	7	Lunch	Mexican	Make tacos	corn tortillas, meat, cheese, lettuce	0
8	8	Snack	Italian	Bake garlic bread	bread, garlic, butter	0
9	9	Breakfast	French	Make croissants	flour, butter, sugar	0
10	10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1
11	11	Snack	Greek	Mix yogurt with honey	yogurt, honey	1

Delete:

DELETE FROM Recipe WHERE Recipe_ID = 11;

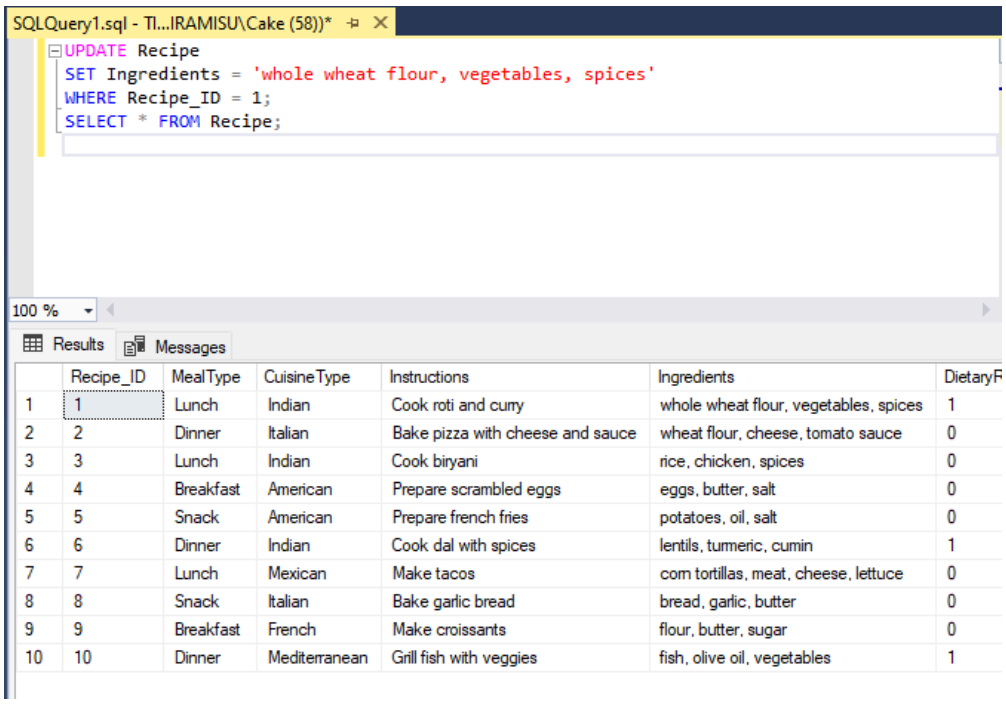
SELECT * FROM Recipe;



	Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryRestri
1	1	Lunch	Indian	Cook roti and curry	wheat flour, vegetables, spices	1
2	2	Dinner	Italian	Bake pizza with cheese and sauce	wheat flour, cheese, tomato sauce	0
3	3	Lunch	Indian	Cook biryani	rice, chicken, spices	0
4	4	Breakfast	American	Prepare scrambled eggs	eggs, butter, salt	0
5	5	Snack	American	Prepare french fries	potatoes, oil, salt	0
6	6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
7	7	Lunch	Mexican	Make tacos	corn tortillas, meat, cheese, lettuce	0
8	8	Snack	Italian	Bake garlic bread	bread, garlic, butter	0
9	9	Breakfast	French	Make croissants	flour, butter, sugar	0
10	10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1

Update:

UPDATE Recipe
SET Ingredients = 'whole wheat flour, vegetables, spices'
WHERE Recipe_ID = 1;
SELECT * FROM Recipe;

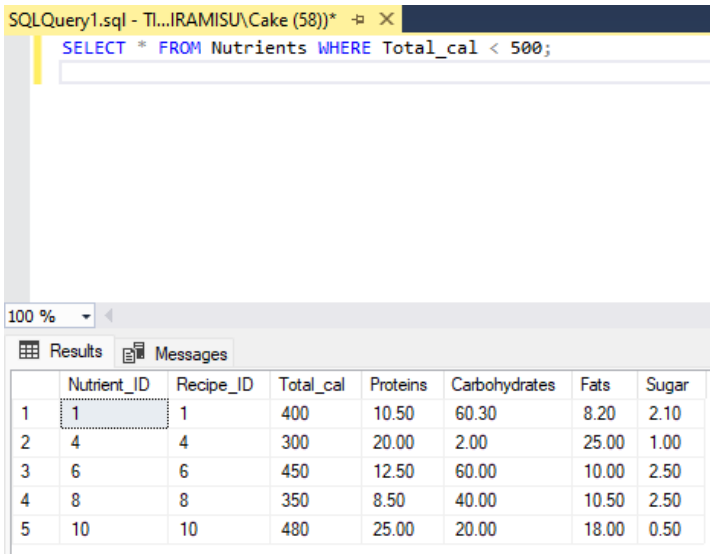


	Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryR
1	1	Lunch	Indian	Cook roti and curry	whole wheat flour, vegetables, spices	1
2	2	Dinner	Italian	Bake pizza with cheese and sauce	wheat flour, cheese, tomato sauce	0
3	3	Lunch	Indian	Cook biryani	rice, chicken, spices	0
4	4	Breakfast	American	Prepare scrambled eggs	eggs, butter, salt	0
5	5	Snack	American	Prepare french fries	potatoes, oil, salt	0
6	6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
7	7	Lunch	Mexican	Make tacos	corn tortillas, meat, cheese, lettuce	0
8	8	Snack	Italian	Bake garlic bread	bread, garlic, butter	0
9	9	Breakfast	French	Make croissants	flour, butter, sugar	0
10	10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1

2. **Nutrients**

Query:

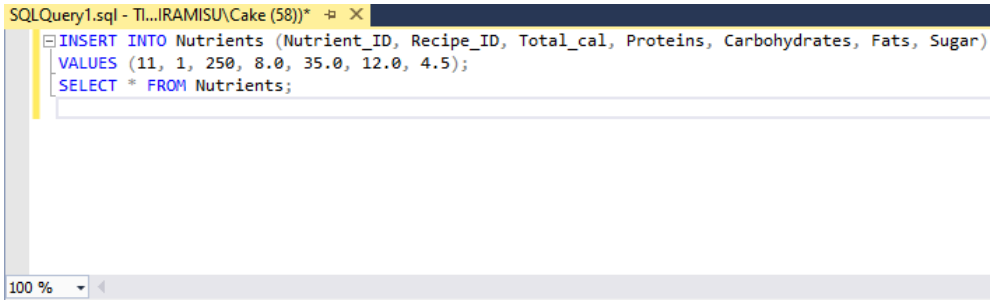
SELECT * FROM Nutrients WHERE Total_cal < 500;



	Nutrient_ID	Recipe_ID	Total_cal	Proteins	Carbohydrates	Fats	Sugar
1	1	1	400	10.50	60.30	8.20	2.10
2	4	4	300	20.00	2.00	25.00	1.00
3	6	6	450	12.50	60.00	10.00	2.50
4	8	8	350	8.50	40.00	10.50	2.50
5	10	10	480	25.00	20.00	18.00	0.50

Insert:

INSERT INTO Nutrients (Nutrient_ID, Recipe_ID, Total_cal, Proteins, Carbohydrates, Fats, Sugar)
VALUES (11, 1, 250, 8.0, 35.0, 12.0, 4.5);
SELECT * FROM Nutrients;



	Nutrient_ID	Recipe_ID	Total_cal	Proteins	Carbohydrates	Fats	Sugar
1	1	1	400	10.50	60.30	8.20	2.10
2	2	2	800	20.30	90.10	30.50	5.20
3	3	3	650	25.20	70.50	15.80	3.40
4	4	4	300	20.00	2.00	25.00	1.00
5	5	5	500	5.20	50.50	25.30	0.50
6	6	6	450	12.50	60.00	10.00	2.50
7	7	7	550	15.80	65.30	20.20	3.00
8	8	8	350	8.50	40.00	10.50	2.50
9	9	9	600	12.30	55.50	35.00	6.00
10	10	10	480	25.00	20.00	18.00	0.50
11	11	1	250	8.00	35.00	12.00	4.50

Delete:

DELETE FROM Nutrients WHERE Nutrient_ID = 11;
SELECT * FROM Nutrients;

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
DELETE FROM Nutrients WHERE Nutrient_ID = 11;
SELECT * FROM Nutrients;
```

100 %

Results | Messages

	Nutrient_ID	Recipe_ID	Total_cal	Proteins	Carbohydrates	Fats	Sugar
1	1	1	400	10.50	60.30	8.20	2.10
2	2	2	800	20.30	90.10	30.50	5.20
3	3	3	650	25.20	70.50	15.80	3.40
4	4	4	300	20.00	2.00	25.00	1.00
5	5	5	500	5.20	50.50	25.30	0.50
6	6	6	450	12.50	60.00	10.00	2.50
7	7	7	550	15.80	65.30	20.20	3.00
8	8	8	350	8.50	40.00	10.50	2.50
9	9	9	600	12.30	55.50	35.00	6.00
10	10	10	480	25.00	20.00	18.00	0.50

Update:

UPDATE Nutrients
SET Total_cal = 700
WHERE Recipe_ID = 3;
SELECT * FROM Nutrients;

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
UPDATE Nutrients
SET Total_cal = 700
WHERE Recipe_ID = 3;
SELECT * FROM Nutrients;
```

100 %

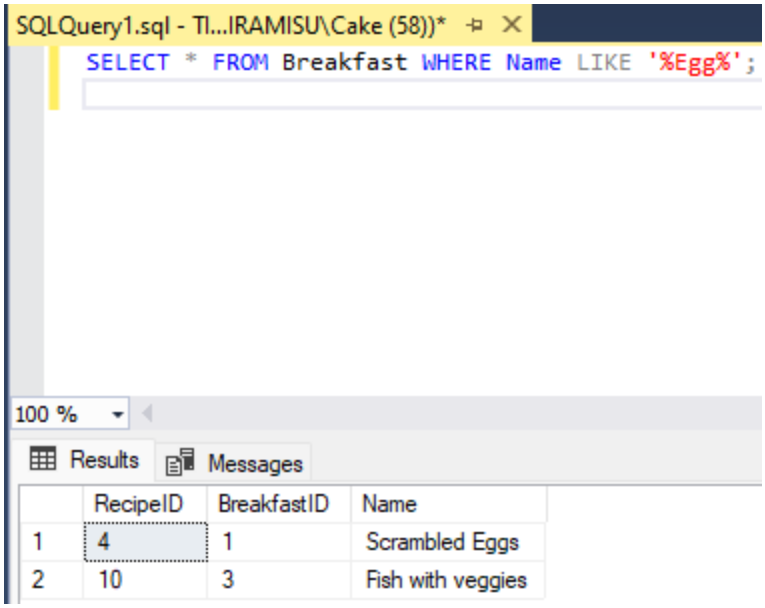
Results | Messages

	Nutrient_ID	Recipe_ID	Total_cal	Proteins	Carbohydrates	Fats	Sugar
1	1	1	400	10.50	60.30	8.20	2.10
2	2	2	800	20.30	90.10	30.50	5.20
3	3	3	700	25.20	70.50	15.80	3.40
4	4	4	300	20.00	2.00	25.00	1.00
5	5	5	500	5.20	50.50	25.30	0.50
6	6	6	450	12.50	60.00	10.00	2.50
7	7	7	550	15.80	65.30	20.20	3.00
8	8	8	350	8.50	40.00	10.50	2.50
9	9	9	600	12.30	55.50	35.00	6.00
10	10	10	480	25.00	20.00	18.00	0.50

3. **Breakfast**

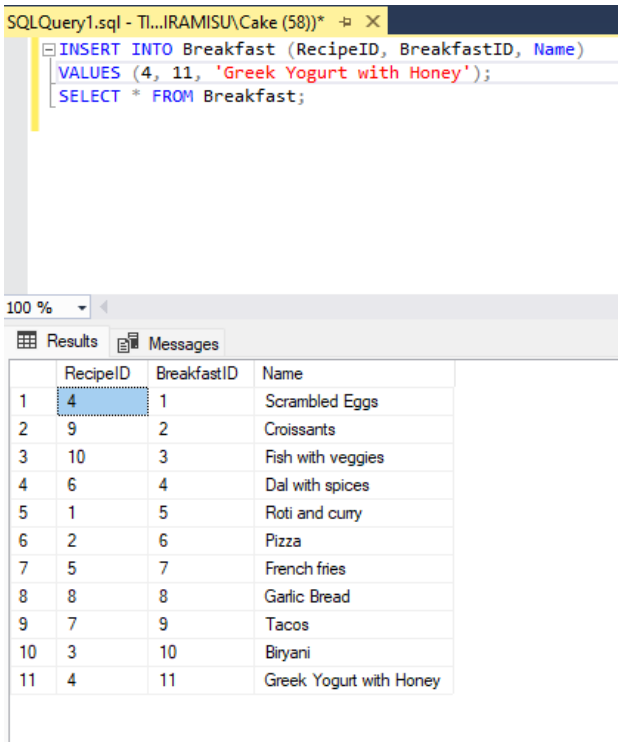
Query:

SELECT * FROM Breakfast WHERE Name LIKE '%Egg%';



Insert:

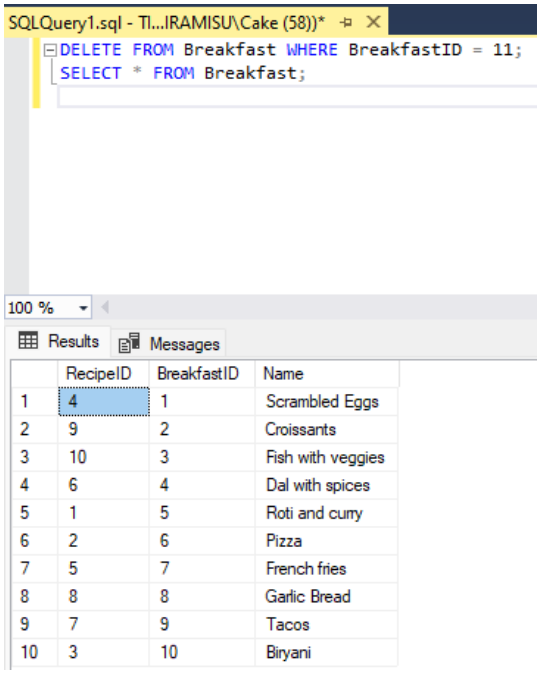
```
INSERT INTO Breakfast (RecipeID, BreakfastID, Name)
VALUES (4, 11, 'Greek Yogurt with Honey');
SELECT * FROM Breakfast;
```



	RecipeID	BreakfastID	Name
1	4	1	Scrambled Eggs
2	9	2	Croissants
3	10	3	Fish with veggies
4	6	4	Dal with spices
5	1	5	Roti and curry
6	2	6	Pizza
7	5	7	French fries
8	8	8	Garlic Bread
9	7	9	Tacos
10	3	10	Biryani
11	4	11	Greek Yogurt with Honey

Delete:

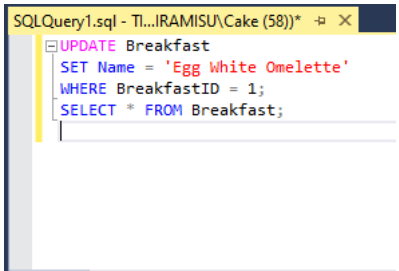
```
DELETE FROM Breakfast WHERE BreakfastID = 5;
SELECT * FROM Breakfast;
```



	RecipeID	BreakfastID	Name
1	4	1	Scrambled Eggs
2	9	2	Croissants
3	10	3	Fish with veggies
4	6	4	Dal with spices
5	1	5	Roti and curry
6	2	6	Pizza
7	5	7	French fries
8	8	8	Garlic Bread
9	7	9	Tacos
10	3	10	Biryani

Update:

UPDATE Breakfast
SET Name = 'Egg White Omelette'
WHERE BreakfastID = 1;
SELECT * FROM Breakfast;

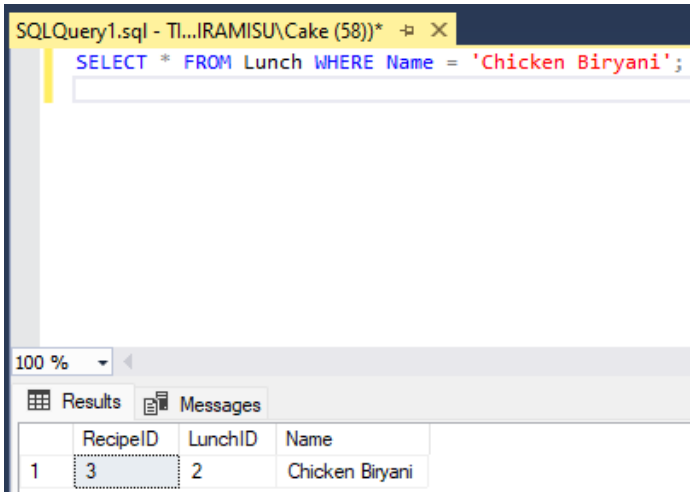


	RecipeID	BreakfastID	Name
1	4	1	Egg White Omelette
2	9	2	Croissants
3	10	3	Fish with veggies
4	6	4	Dal with spices
5	1	5	Roti and curry
6	2	6	Pizza
7	5	7	French fries
8	8	8	Garlic Bread
9	7	9	Tacos
10	3	10	Biryani

4. **Lunch**

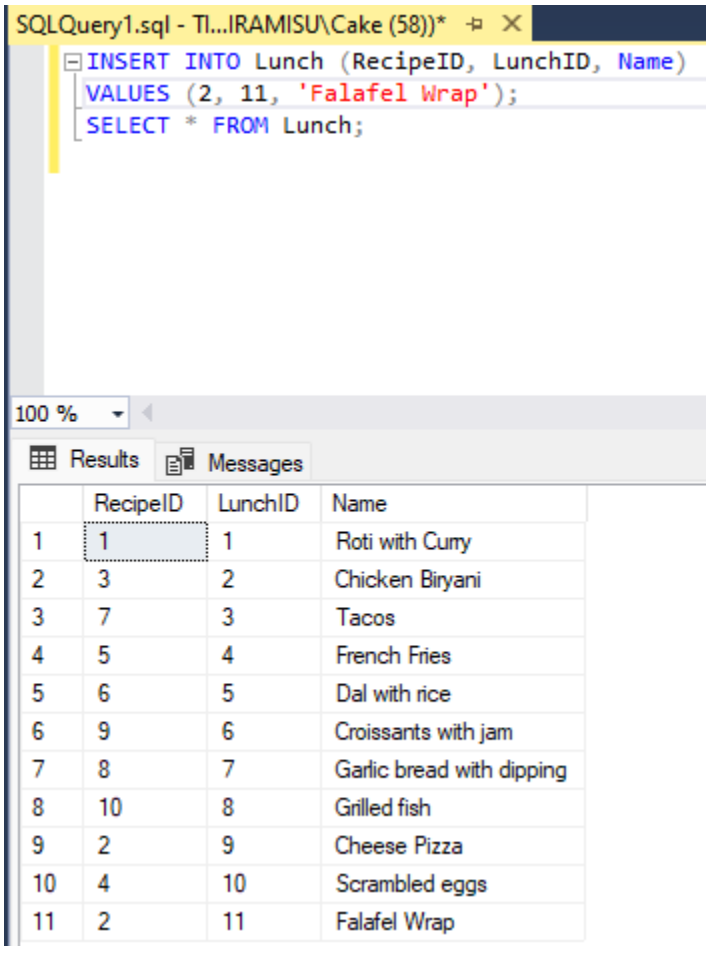
Query:

SELECT * FROM Lunch WHERE Name = 'Chicken Biryani';



Insert:

INSERT INTO Lunch (RecipeID, LunchID, Name)
VALUES (2,11, 'Falafel Wrap');
SELECT * FROM Lunch;

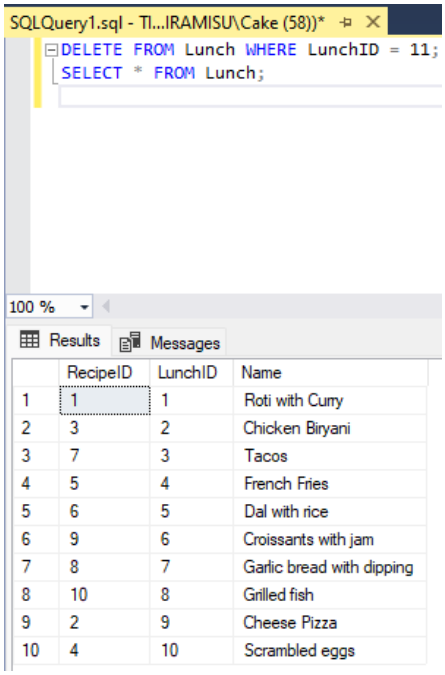


	RecipeID	LunchID	Name
1	1	1	Roti with Curry
2	3	2	Chicken Biryani
3	7	3	Tacos
4	5	4	French Fries
5	6	5	Dal with rice
6	9	6	Croissants with jam
7	8	7	Garlic bread with dipping
8	10	8	Grilled fish
9	2	9	Cheese Pizza
10	4	10	Scrambled eggs
11	2	11	Falafel Wrap

Delete:

DELETE FROM Lunch WHERE LunchID = 11;

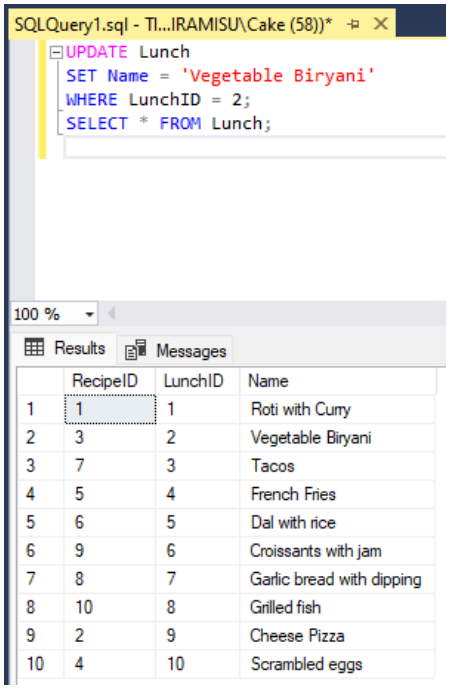
SELECT * FROM Lunch;



	RecipeID	LunchID	Name
1	1	1	Roti with Curry
2	3	2	Chicken Biryani
3	7	3	Tacos
4	5	4	French Fries
5	6	5	Dal with rice
6	9	6	Croissants with jam
7	8	7	Garlic bread with dipping
8	10	8	Grilled fish
9	2	9	Cheese Pizza
10	4	10	Scrambled eggs

Update:

UPDATE Lunch
SET Name = 'Vegetable Biryani'
WHERE LunchID = 2;
SELECT * FROM Lunch;

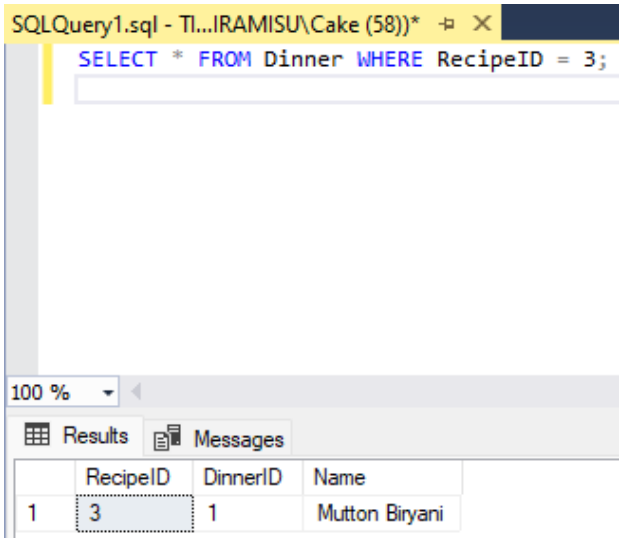


	RecipeID	LunchID	Name
1	1	1	Roti with Curry
2	3	2	Vegetable Biryani
3	7	3	Tacos
4	5	4	French Fries
5	6	5	Dal with rice
6	9	6	Croissants with jam
7	8	7	Garlic bread with dipping
8	10	8	Grilled fish
9	2	9	Cheese Pizza
10	4	10	Scrambled eggs

5. **Dinner**

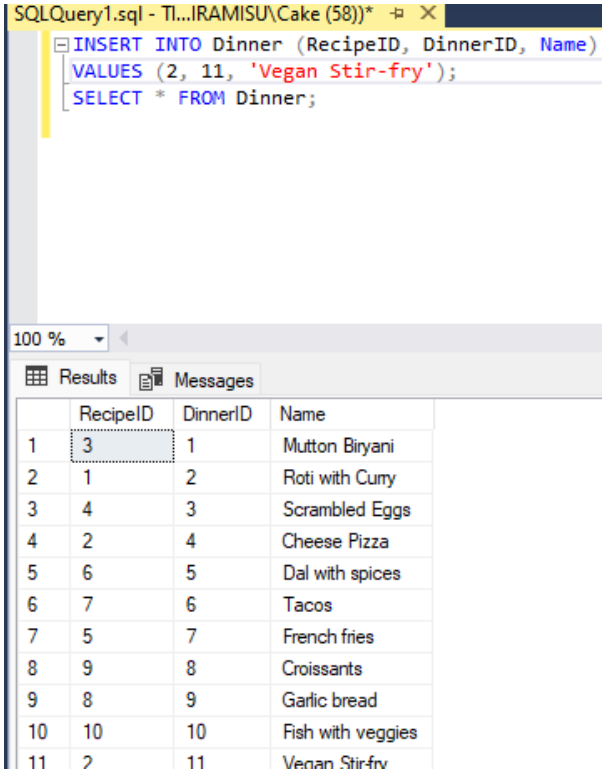
Query:

SELECT * FROM Dinner WHERE RecipeID = 3;



Insert:

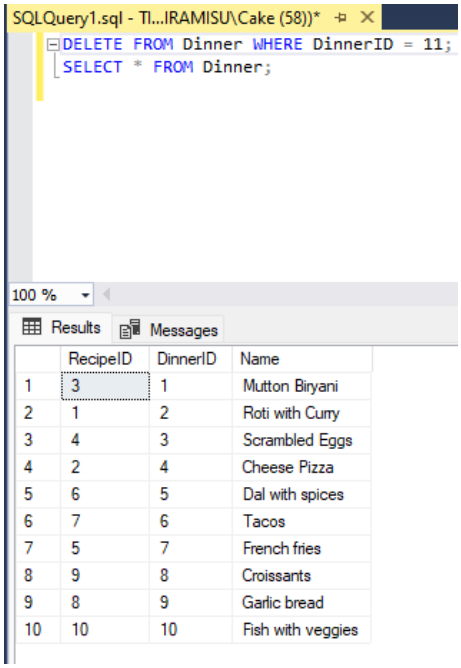
INSERT INTO Dinner (RecipeID, DinnerID, Name)
VALUES (2, 11, 'Vegan Stir-fry');
SELECT * FROM Dinner;



	RecipeID	DinnerID	Name
1	3	1	Mutton Biryani
2	1	2	Roti with Curry
3	4	3	Scrambled Eggs
4	2	4	Cheese Pizza
5	6	5	Dal with spices
6	7	6	Tacos
7	5	7	French fries
8	9	8	Croissants
9	8	9	Garlic bread
10	10	10	Fish with veggies
11	2	11	Vegan Stir-fry

Delete:

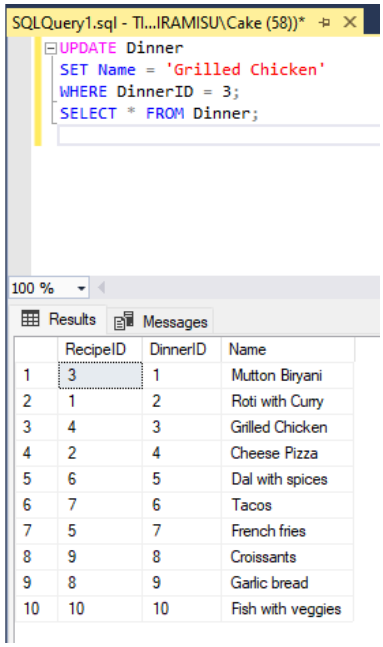
DELETE FROM Dinner WHERE DinnerID = 11;
SELECT * FROM Dinner;



	RecipeID	DinnerID	Name
1	3	1	Mutton Biryani
2	1	2	Roti with Curry
3	4	3	Scrambled Eggs
4	2	4	Cheese Pizza
5	6	5	Dal with spices
6	7	6	Tacos
7	5	7	French fries
8	9	8	Croissants
9	8	9	Garlic bread
10	10	10	Fish with veggies

Update:

UPDATE Dinner
SET Name = 'Grilled Chicken'
WHERE DinnerID = 3;
SELECT * FROM Dinner;

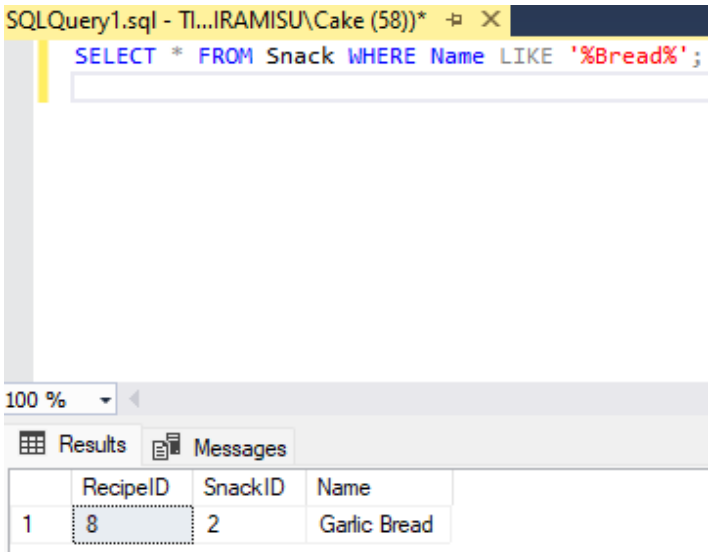


	RecipeID	DinnerID	Name
1	3	1	Mutton Biryani
2	1	2	Roti with Curry
3	4	3	Grilled Chicken
4	2	4	Cheese Pizza
5	6	5	Dal with spices
6	7	6	Tacos
7	5	7	French fries
8	9	8	Croissants
9	8	9	Garlic bread
10	10	10	Fish with veggies

6. **Snack**

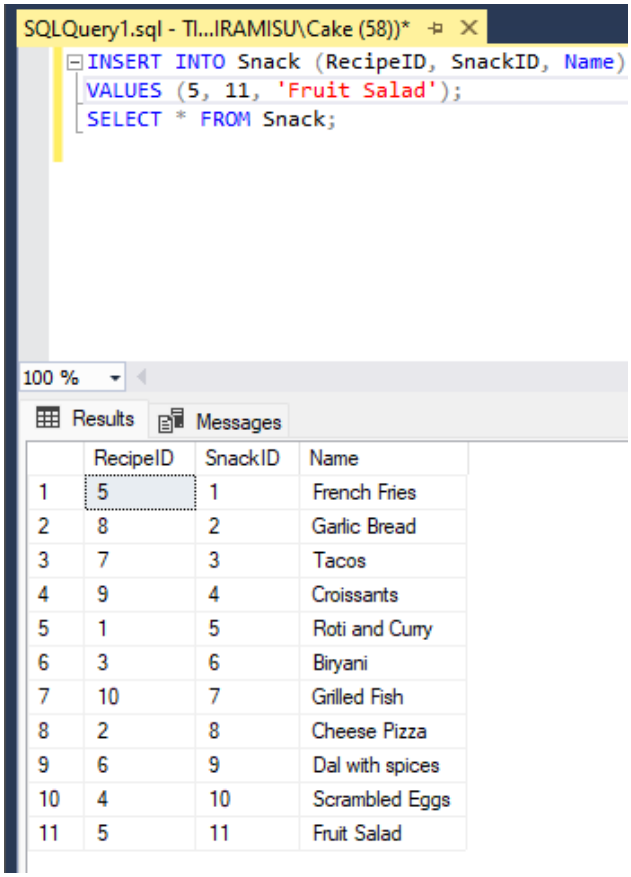
Query:

SELECT * FROM Snack WHERE Name LIKE '%Bread%';



Insert:

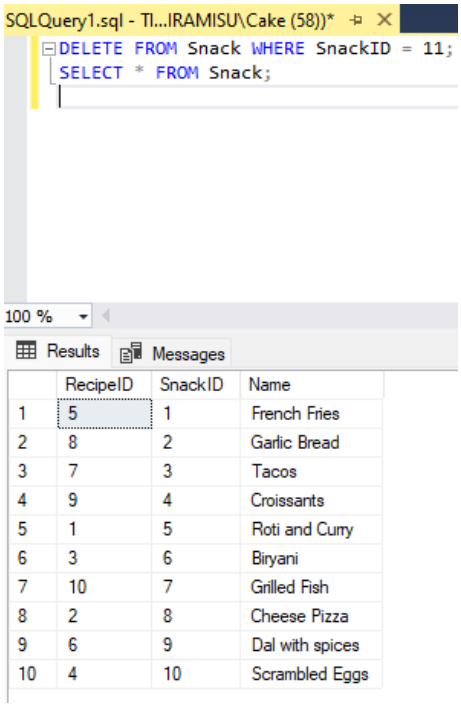
INSERT INTO Snack (RecipeID, SnackID, Name)
VALUES (5, 11, 'Fruit Salad');
SELECT * FROM Snack;



Delete:

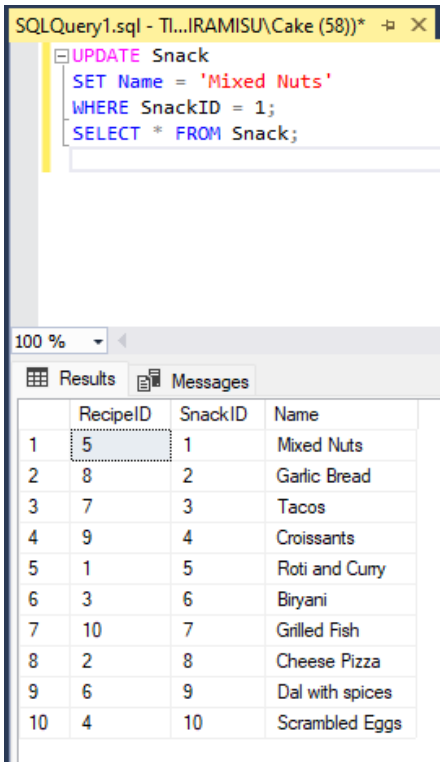
DELETE FROM Snack WHERE SnackID = 11;

SELECT * FROM Snack;



	RecipeID	SnackID	Name
1	5	1	French Fries
2	8	2	Garlic Bread
3	7	3	Tacos
4	9	4	Croissants
5	1	5	Roti and Curry
6	3	6	Biryani
7	10	7	Grilled Fish
8	2	8	Cheese Pizza
9	6	9	Dal with spices
10	4	10	Scrambled Eggs

Update:

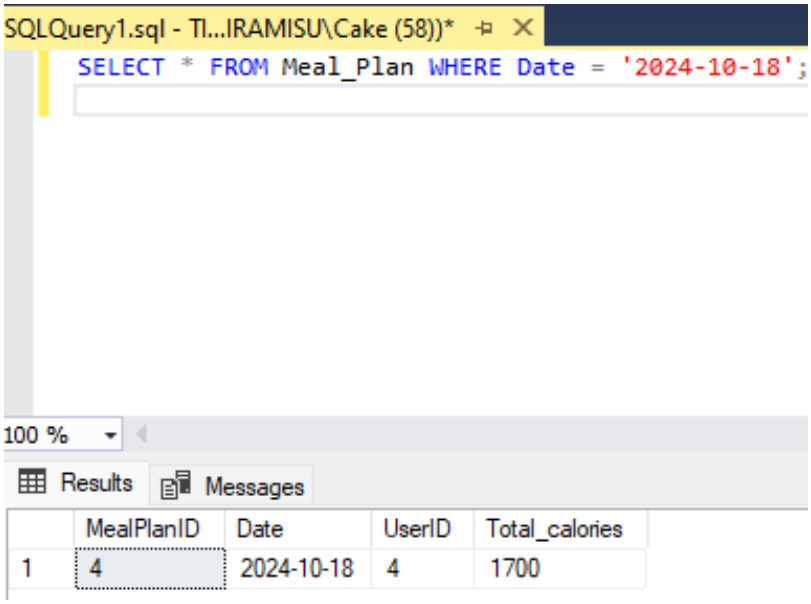


	RecipeID	SnackID	Name
1	5	1	Mixed Nuts
2	8	2	Garlic Bread
3	7	3	Tacos
4	9	4	Croissants
5	1	5	Roti and Curry
6	3	6	Biryani
7	10	7	Grilled Fish
8	2	8	Cheese Pizza
9	6	9	Dal with spices
10	4	10	Scrambled Eggs

7. **Meal_Plan**

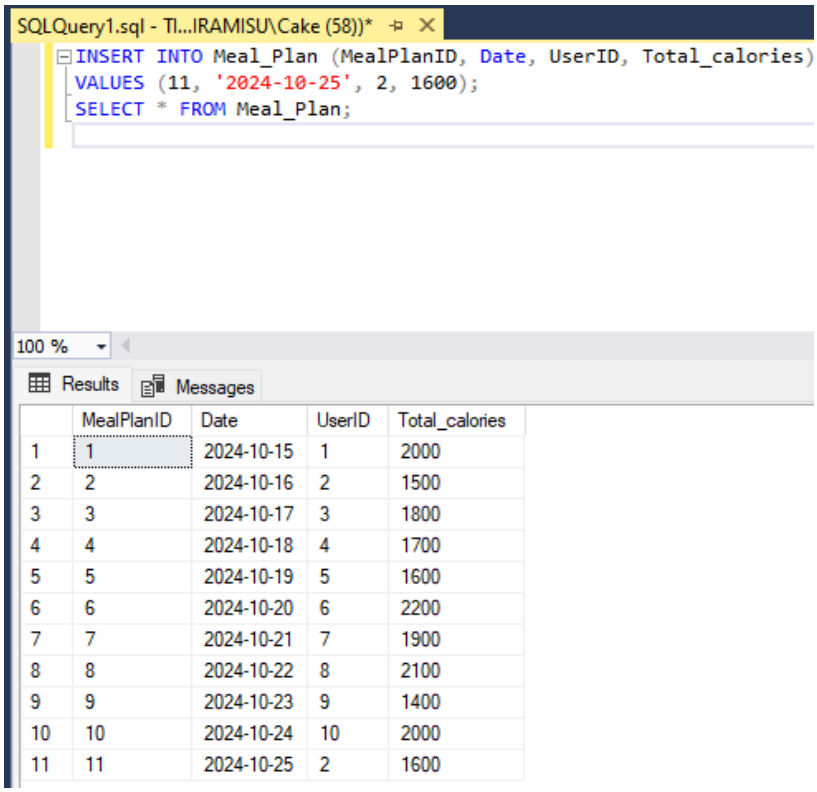
Query:

SELECT * FROM Meal_Plan WHERE Date = '2024-10-18';



Insert:

INSERT INTO Meal_Plan (MealPlanID, Date, UserID, Total_calories)
VALUES (11, '2024-10-25', 2, 1600);
SELECT * FROM Meal_Plan;



	MealPlanID	Date	UserID	Total_calories
1	1	2024-10-15	1	2000
2	2	2024-10-16	2	1500
3	3	2024-10-17	3	1800
4	4	2024-10-18	4	1700
5	5	2024-10-19	5	1600
6	6	2024-10-20	6	2200
7	7	2024-10-21	7	1900
8	8	2024-10-22	8	2100
9	9	2024-10-23	9	1400
10	10	2024-10-24	10	2000
11	11	2024-10-25	2	1600

Delete:

DELETE FROM Meal_Plan WHERE MealPlanID = 11;
SELECT * FROM Meal_Plan;

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
DELETE FROM Meal_Plan WHERE MealPlanID = 11;
SELECT * FROM Meal_Plan;
```

100 %

Results | Messages

	MealPlanID	Date	UserID	Total_calories
1	1	2024-10-15	1	2000
2	2	2024-10-16	2	1500
3	3	2024-10-17	3	1800
4	4	2024-10-18	4	1700
5	5	2024-10-19	5	1600
6	6	2024-10-20	6	2200
7	7	2024-10-21	7	1900
8	8	2024-10-22	8	2100
9	9	2024-10-23	9	1400
10	10	2024-10-24	10	2000

Update:

```
UPDATE Meal_Plan
SET Total_calories = 1750
WHERE MealPlanID = 2;
SELECT * FROM Meal_Plan;
```

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
UPDATE Meal_Plan
SET Total_calories = 1750
WHERE MealPlanID = 2;
SELECT * FROM Meal_Plan;
```

100 %

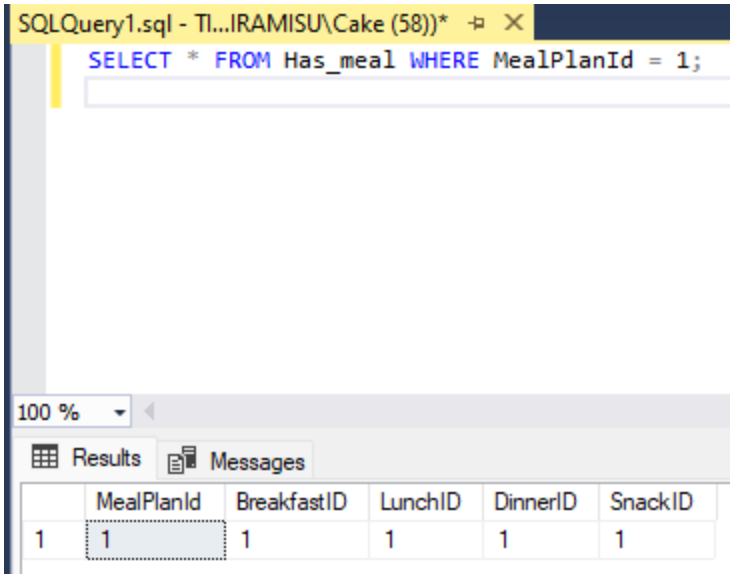
Results | Messages

	MealPlanID	Date	UserID	Total_calories
1	1	2024-10-15	1	2000
2	2	2024-10-16	2	1750
3	3	2024-10-17	3	1800
4	4	2024-10-18	4	1700
5	5	2024-10-19	5	1600
6	6	2024-10-20	6	2200
7	7	2024-10-21	7	1900
8	8	2024-10-22	8	2100
9	9	2024-10-23	9	1400
10	10	2024-10-24	10	2000

8. **Has_Meal**

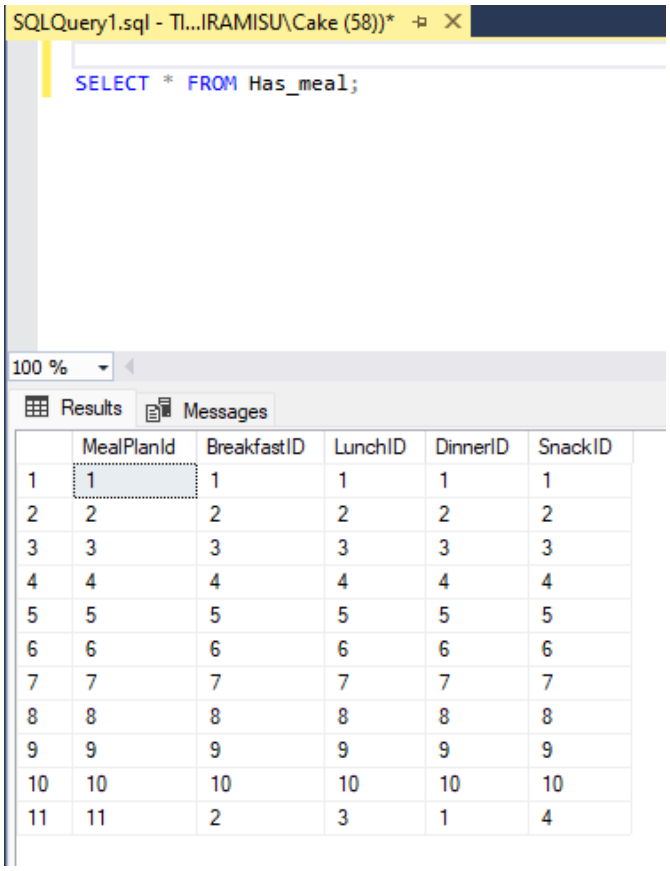
Query:

```
SELECT * FROM Has_meal WHERE MealPlanId = 1;
```



Insert:

INSERT INTO Has_meal (MealPlanId, BreakfastID, LunchID, DinnerID, SnackID)
VALUES (11, 2, 3, 1, 4);



	MealPlanId	BreakfastID	LunchID	DinnerID	SnackID
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	4	4	4	4	4
5	5	5	5	5	5
6	6	6	6	6	6
7	7	7	7	7	7
8	8	8	8	8	8
9	9	9	9	9	9
10	10	10	10	10	10
11	11	2	3	1	4

Delete:

DELETE FROM Has_meal WHERE MealPlanId = 11;
SELECT * FROM Has_meal;

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
DELETE FROM Has_meal WHERE MealPlanId = 11;
SELECT * FROM Has_meal;
```

100 %

Results | Messages

	MealPlanId	BreakfastID	LunchID	DinnerID	SnackID
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	4	4	4	4	4
5	5	5	5	5	5
6	6	6	6	6	6
7	7	7	7	7	7
8	8	8	8	8	8
9	9	9	9	9	9
10	10	10	10	10	10

Update:

UPDATE Has_meal
SET LunchID = 5
WHERE MealPlanId = 1;
SELECT * FROM Has_meal;

SQLQuery1.sql - TI...IRAMISU\Cake (58))*

```
UPDATE Has_meal
SET LunchID = 5
WHERE MealPlanId = 1;
SELECT * FROM Has_meal;
```

100 %

Results | Messages

	MealPlanId	BreakfastID	LunchID	DinnerID	SnackID
1	1	1	5	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	4	4	4	4	4
5	5	5	5	5	5
6	6	6	6	6	6
7	7	7	7	7	7
8	8	8	8	8	8
9	9	9	9	9	9
10	10	10	10	10	10

3.5. Create View:

Use a minimum one CREATE VIEW statement in your normalized database to implement a view based on your specific database design. Please indicate what this view is for. Provide a screenshot of your view(s), as well as each resulting table in your report exactly in this section.

View 1 - DailyMealPlan: This view provides a detailed view of the meal plan for each user on a given day including the different meals in the day and total calories.

```sql
CREATE VIEW DailyMealPlan
AS SELECT
    MP.MealPlanID,
    MP.Date,
    U.Name AS User_Name,
    U.Calories AS User_Calorie_Goal,
    B.Name AS Breakfast,
    L.Name AS Lunch,
    D.Name AS Dinner,
    S.Name AS Snack,
    MP.Total_calories AS Total_MealPlan_Calories
FROM
    Meal_Plan MP
JOIN
    Users U ON MP.UserID = U.User_ID
LEFT JOIN
    Has_meal HM ON MP.MealPlanID = HM.MealPlanId
LEFT JOIN
    Breakfast B ON HM.BreakfastID = B.BreakfastID
LEFT JOIN
    Lunch L ON HM.LunchID = L.LunchID
LEFT JOIN
    Dinner D ON HM.DinnerID = D.DinnerID
LEFT JOIN
    Snack S ON HM.SnackID = S.SnackID;
SELECT * FROM DailyMealPlan;
```

MealPlanID	Date	User_Name	User_Calorie_Goal	Breakfast	Lunch	Dinner	Snack	Total_MealPlan_Calori...
1	2024-10-15	John Doe	2000	Scrambled Eggs	Roti with Curry	Mutton Biryani	French Fries	2000
2	2024-10-16	Jane Smith	1500	Croissants	Chicken Biryani	Roti with Curry	Garlic Bread	1500
3	2024-10-17	Ahmed Ali	1800	Fish with veggies	Tacos	Scrambled Eggs	Tacos	1800
4	2024-10-18	Sarah Cohen	1700	Dal with spices	French Fries	Cheese Pizza	Croissants	1700
5	2024-10-19	Emily Johnson	1600	Roti and curry	Dal with rice	Dal with spices	Roti and Curry	1600
6	2024-10-20	David Lee	2200	Pizza	Croissants with jam	Tacos	Biryani	2200
7	2024-10-21	Anna Gupta	1900	French fries	Garlic bread with dipping	French fries	Grilled Fish	1900
8	2024-10-22	Paul Martin	2100	Garlic Bread	Grilled fish	Croissants	Cheese Pizza	2100
9	2024-10-23	Olivia Taylor	1400	Tacos	Cheese Pizza	Garlic bread	Dal with spices	1400
10	2024-10-24	Carlos Ramirez	2000	Biryani	Scrambled eggs	Fish with veggies	Scrambled Eggs	2000

DailyMealPlan 7 | RecipeNutritionalInfo 8

View 2 - RecipeNutritionalInfo: This view joins the recipe table with the nutrients tables to show each recipe with its nutrient breakdown

```
31    -- This view gives each recipe with its nutrient breakdown
32    CREATE VIEW RecipeNutritionalInfo
33    AS SELECT
34        R.Recipe_ID,
35        R.MealType,
36        R.CuisineType,
37        R.Ingredients,
38        N.Total_cal AS Calories,
39        N.Proteins,
40        N.Carbohydrates,
41        N.Fats,
42        N.Sugar
43    FROM
44        Recipe R
45    LEFT JOIN
46        Nutrients N ON R.Recipe_ID = N.Recipe_ID;
47    SELECT * FROM RecipeNutritionalInfo;
48
```

Recipe_ID	MealType	CuisineType	Ingredients	Calories	Proteins	Carbohydrates	Fats	Sugar
1	Lunch	Indian	wheat flour, vegetables, spices	400	10.50	60.30	8.20	2.10
2	Dinner	Italian	wheat flour, cheese, tomato sauce	800	20.30	90.10	30.50	5.20
3	Lunch	Indian	rice, chicken, spices	650	25.20	70.50	15.80	3.40
4	Breakfast	American	eggs, butter, salt	300	20.00	2.00	25.00	1.00
5	Snack	American	potatoes, oil, salt	500	5.20	50.50	25.30	0.50
6	Dinner	Indian	lentils, turmeric, cumin	450	12.50	60.00	10.00	2.50
7	Lunch	Mexican	corn tortillas, meat, cheese, lettuce	550	15.80	65.30	20.20	3.00
8	Snack	Italian	bread, garlic, butter	350	8.50	40.00	10.50	2.50
9	Breakfast	French	flour, butter, sugar	600	12.30	55.50	35.00	6.00
10	Dinner	Mediterranean	fish, olive oil, vegetables	480	25.00	20.00	18.00	0.50

DailyMealPlan 7 | RecipeNutritionalInfo 8

4. Front End User Interface:

Query:

Some examples of regular querying:

HealthyMeals Dashboard

Successfully queried Users

Query | Insert | Delete | Update | Quit

Query Data

Regular Table Query

Users

User_ID	Name	Email	Calories	RestrictionID
1	John Doe	john.doe@example.com	2000	1
2	Jane Smith	jane.smith@example.com	1500	2
3	Ahmed Ali	ahmed.ali@example.com	1800	3
4	Sarah Cohen	sarah.cohen@example.com	1700	4
5	Emily Johnson	emily.johnson@example.com	1600	5
6	David Lee	david.lee@example.com	2200	6
7	Anna Gupta	anna.gupta@example.com	1900	7
8	Paul Martin	paul.martin@example.com	2100	8
9	Olivia Taylor	olivia.taylor@example.com	1400	9
10	Carlos Ramirez	carlos.ramirez@example.com	2000	10

HealthyMeals Dashboard

Successfully queried Recipe

Query | Insert | Delete | Update | Quit

Query Data

Regular Table Query

Recipe

Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryRestrictionMet
1	Lunch	Indian	Cook roti and curry	wheat flour, vegetables, spices	1
2	Dinner	Italian	Bake pizza with cheese and sauce	wheat flour, cheese, tomato sauce	0
3	Lunch	Indian	Cook biryani	rice, chicken, spices	0
4	Breakfast	American	Prepare scrambled eggs	eggs, butter, salt	0
5	Snack	American	Prepare french fries	potatoes, oil, salt	0
6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
7	Lunch	Mexican	Make tacos	corn tortillas, meat, cheese, lettuce	0
8	Snack	Italian	Bake garlic bread	bread, garlic, butter	0
9	Breakfast	French	Make croissants	flour, butter, sugar	0
10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1
12	Dinner	Meditteranean	Cook the pasta, add parmesan cheese on top	Pasta, Parmesan, Fried chicken	1

HealthyMeals Dashboard

Successfully queried Dietary_Restrictions

Query | Insert | Delete | Update | Quit

Query Data

Regular Table Query

Dietary_Restrictions

RestrictionID	RestrictionType
1	Vegetarian
2	Vegan
3	Halal
4	Kosher
5	Gluten-Free
6	Dairy-Free
7	Nut-Free
8	Pescatarian
9	Low-Carb
10	Paleo

Special Queries:

HealthyMeals Dashboard

Successfully executed meal-calories query

Query | Insert | Delete | Update | Quit

Query Data

Meal Calories Analysis

Execute Query

Recipe_ID	MealType	CuisineType	Calories	Proteins	Carbohydrates	Fats
2	Dinner	Italian	800	20.30	90.10	30.50
3	Lunch	Indian	650	25.20	70.50	15.80
9	Breakfast	French	600	12.30	55.50	35.00
7	Lunch	Mexican	550	15.80	65.30	20.20
5	Snack	American	500	5.20	50.50	25.30
6	Dinner	Indian	450	12.50	60.00	10.00
1	Lunch	Indian	400	10.50	60.30	8.20
8	Snack	Italian	350	8.50	40.00	10.50
4	Breakfast	American	300	20.00	2.00	25.00

HealthyMeals Dashboard

Successfully executed user-restrictions query

Query | Insert | Delete | Update | Quit

Query Data

User Dietary Profiles

Execute Query

Name	Email	RestrictionType	Cuisine_Preferences
John Doe	john.doe@example.com	Vegetarian	Indian,Italian
Jane Smith	jane.smith@example.com	Vegan	Mediterranean,Mexican
Ahmed Ali	ahmed.ali@example.com	Halal	Middle Eastern,South Asian
Sarah Cohen	sarah.cohen@example.com	Kosher	Jewish
Emily Johnson	emily.johnson@example.com	Gluten-Free	French
David Lee	david.lee@example.com	Dairy-Free	Chinese
Anna Gupta	anna.gupta@example.com	Nut-Free	Thai
Paul Martin	paul.martin@example.com	Pescatarian	Japanese
Olivia Taylor	olivia.taylor@example.com	Low-Carb	Keto
Carlos Ramirez	carlos.ramirez@example.com	Paleo	Paleo

HealthyMeals Dashboard

Successfully executed meal-plan-summary query

Query | Insert | Delete | Update | Quit

Query Data

Meal Plan Summary

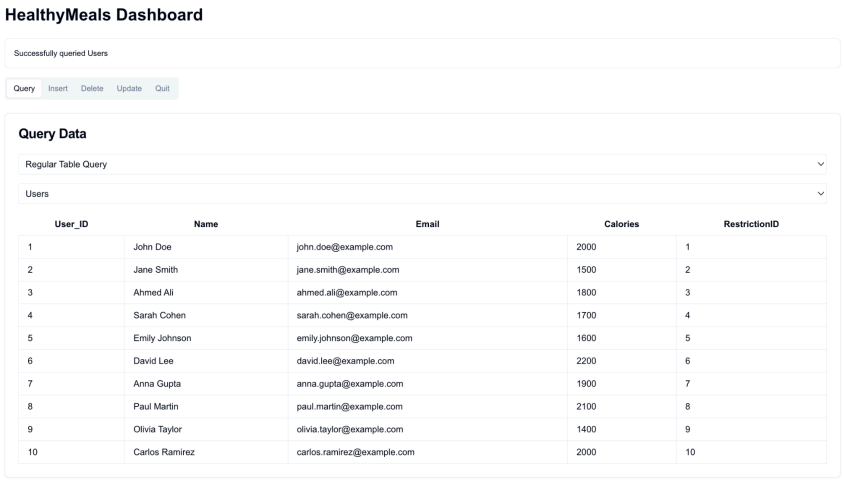
Execute Query

MealPlanID	Date	User_Name	Breakfast_Name	Lunch_Name	Dinner_Name	Snack_Name	Total_calories
10	2024-10-24T05:00:00.000Z	Carlos Ramirez	Biryani	Scrambled eggs	Fish with veggies	Scrambled Eggs	2000
9	2024-10-23T05:00:00.000Z	Olivia Taylor	Tacos	Cheese Pizza	Garlic bread	Dal with spices	1400
8	2024-10-22T05:00:00.000Z	Paul Martin	Garlic Bread	Grilled fish	Croissants	Cheese Pizza	2100
7	2024-10-21T05:00:00.000Z	Anna Gupta	French fries	Garlic bread with dipping	French fries	Grilled Fish	1900
6	2024-10-20T05:00:00.000Z	David Lee	Pizza	Croissants with jam	Tacos	Biryani	2200
5	2024-10-19T05:00:00.000Z	Emily Johnson	Roti and curry	Dal with rice	Dal with spices	Roti and Curry	1600
4	2024-10-18T05:00:00.000Z	Sarah Cohen	Dal with spices	French Fries	Cheese Pizza	Croissants	1700
3	2024-10-17T05:00:00.000Z	Ahmed Ali	Fish with veggies	Tacos	Scrambled Eggs	Tacos	1800
2	2024-10-16T05:00:00.000Z	Jane Smith	Croissants	Chicken Biryani	Roti with Curry	Garlic Bread	1500
1	2024-10-15T05:00:00.000Z	John Doe	Scrambled Eggs	Roti with Curry	Mutton Biryani	French Fries	2000

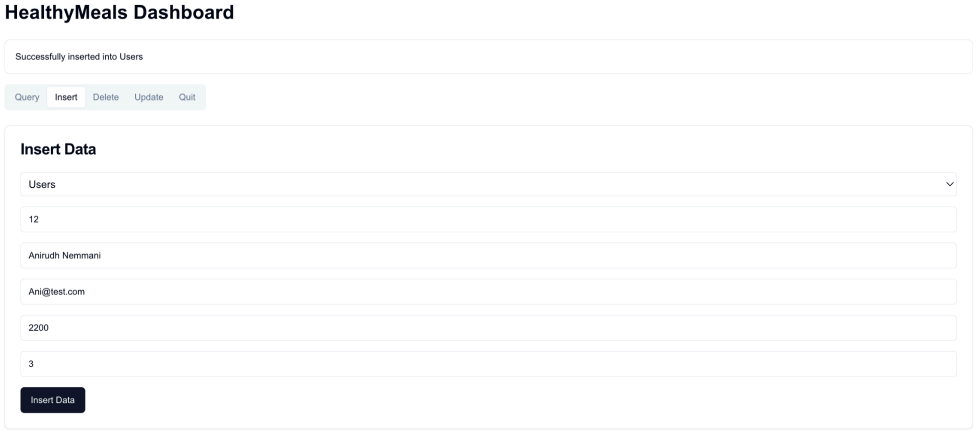
Insert:

Example 1:

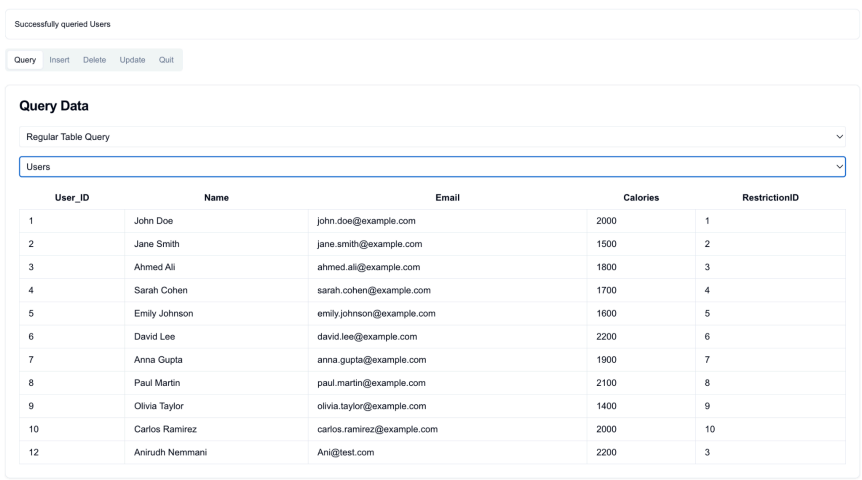
User table before:



Insert page:

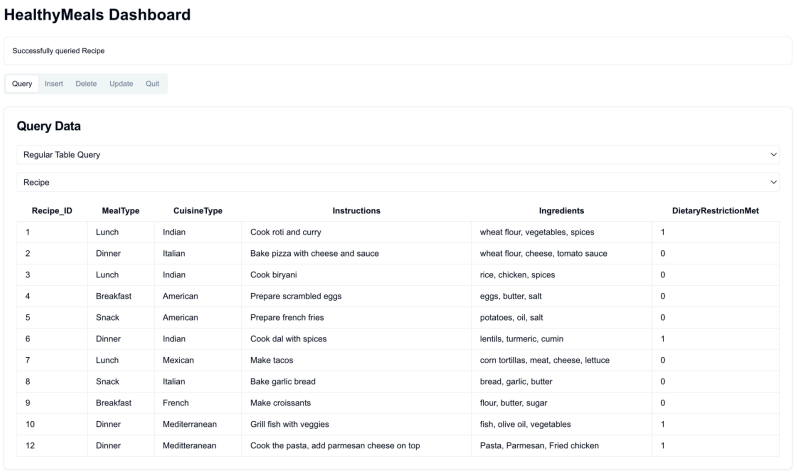


User table after:



Example 2:

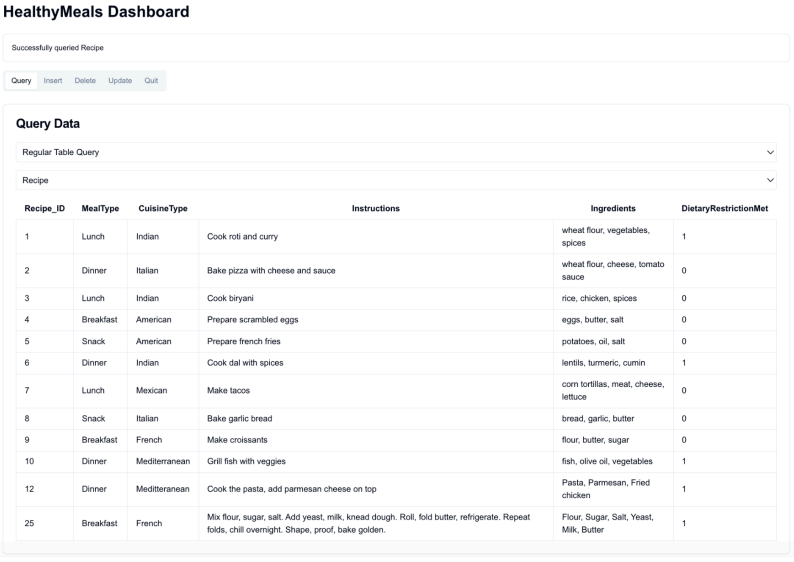
Recipe table before:



Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryRestrictionMet
1	Lunch	Indian	Cook roti and curry	wheat flour, vegetables, spices	1
2	Dinner	Italian	Bake pizza with cheese and sauce	wheat flour, cheese, tomato sauce	0
3	Lunch	Indian	Cook biryani	rice, chicken, spices	0
4	Breakfast	American	Prepare scrambled eggs	eggs, butter, salt	0
5	Snack	American	Prepare french fries	potatoes, oil, salt	0
6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
7	Lunch	Mexican	Make tacos	corn tortillas, meat, cheese, lettuce	0
8	Snack	Italian	Bake garlic bread	bread, garlic, butter	0
9	Breakfast	French	Make croissants	flour, butter, sugar	0
10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1
12	Dinner	Meditteranean	Cook the pasta, add parmesan cheese on top	Pasta, Parmesan, Fried chicken	1

Insert Page:

Recipe table after:

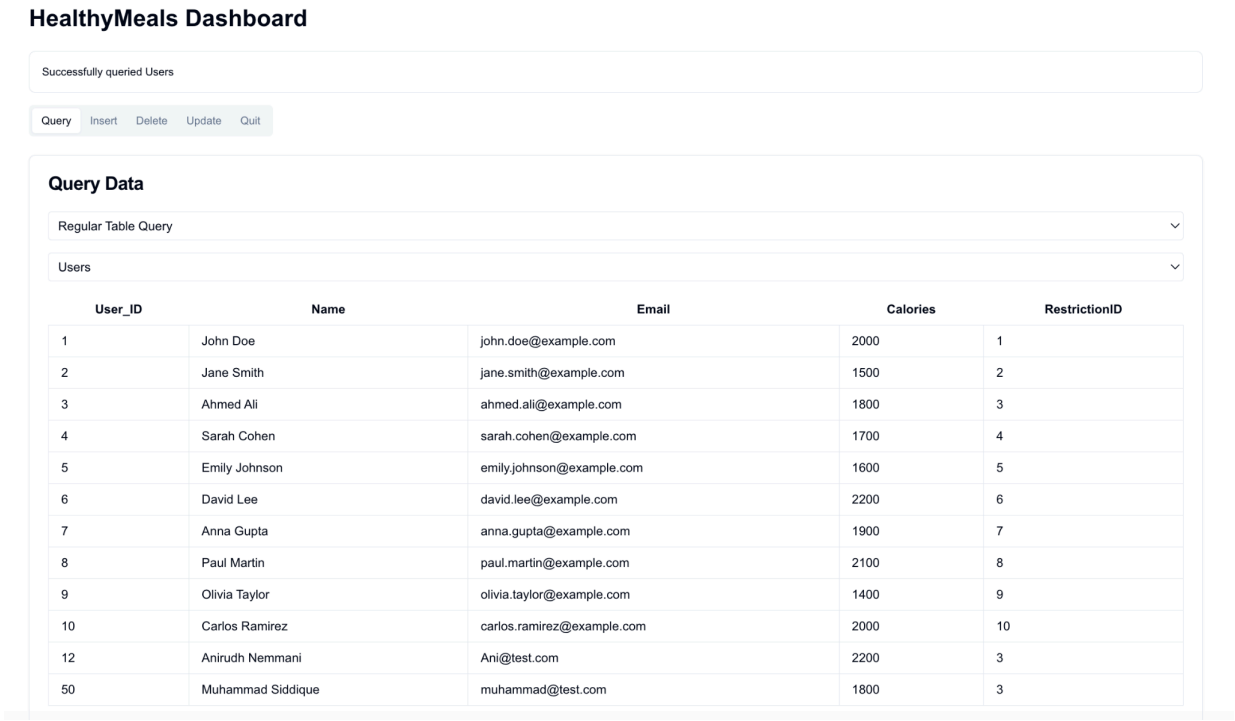


Recipe_ID	MealType	CuisineType	Instructions	Ingredients	DietaryRestrictionMet
1	Lunch	Indian	Cook roti and curry	wheat flour, vegetables, spices	1
2	Dinner	Italian	Bake pizza with cheese and sauce	wheat flour, cheese, tomato sauce	0
3	Lunch	Indian	Cook biryani	rice, chicken, spices	0
4	Breakfast	American	Prepare scrambled eggs	eggs, butter, salt	0
5	Snack	American	Prepare french fries	potatoes, oil, salt	0
6	Dinner	Indian	Cook dal with spices	lentils, turmeric, cumin	1
7	Lunch	Mexican	Make tacos	corn tortillas, meat, cheese, lettuce	0
8	Snack	Italian	Bake garlic bread	bread, garlic, butter	0
9	Breakfast	French	Make croissants	flour, butter, sugar	0
10	Dinner	Mediterranean	Grill fish with veggies	fish, olive oil, vegetables	1
12	Dinner	Meditteranean	Cook the pasta, add parmesan cheese on top	Pasta, Parmesan, Fried chicken	1
25	Breakfast	French	Mix flour, sugar, salt. Add yeast, milk, knead dough. Roll, fold butter, refrigerate. Repeat folds, chill overnight. Shape, proof, bake golden.	Flour, Sugar, Salt, Yeast, Milk, Butter	1

Delete:

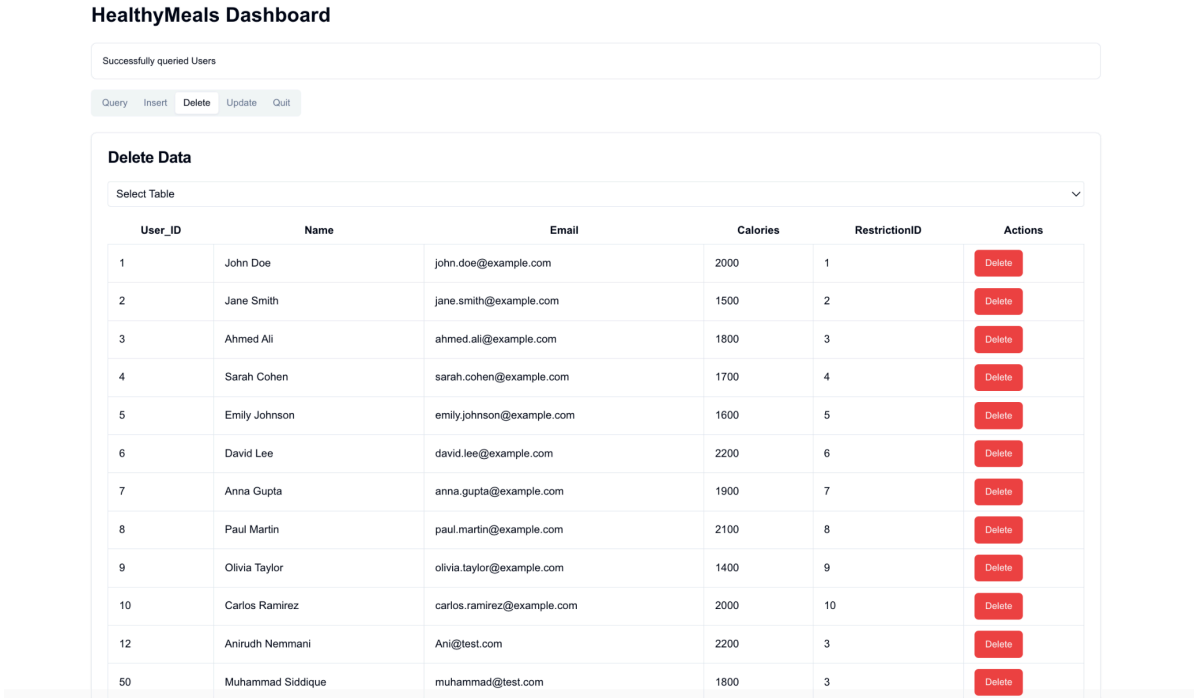
Example 1:

User table before:



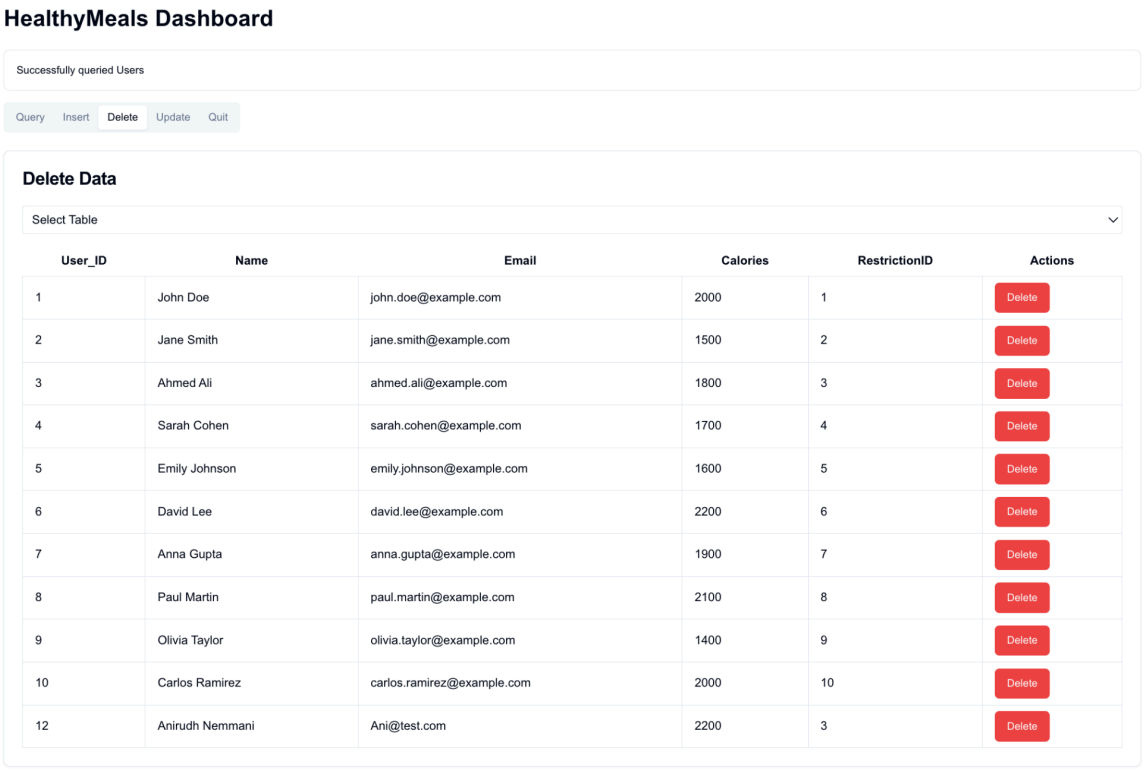
User_ID	Name	Email	Calories	RestrictionID
1	John Doe	john.doe@example.com	2000	1
2	Jane Smith	jane.smith@example.com	1500	2
3	Ahmed Ali	ahmed.ali@example.com	1800	3
4	Sarah Cohen	sarah.cohen@example.com	1700	4
5	Emily Johnson	emily.johnson@example.com	1600	5
6	David Lee	david.lee@example.com	2200	6
7	Anna Gupta	anna.gupta@example.com	1900	7
8	Paul Martin	paul.martin@example.com	2100	8
9	Olivia Taylor	olivia.taylor@example.com	1400	9
10	Carlos Ramirez	carlos.ramirez@example.com	2000	10
12	Anirudh Nemmani	Ani@test.com	2200	3
50	Muhammad Siddique	muhammad@test.com	1800	3

Deletion page:



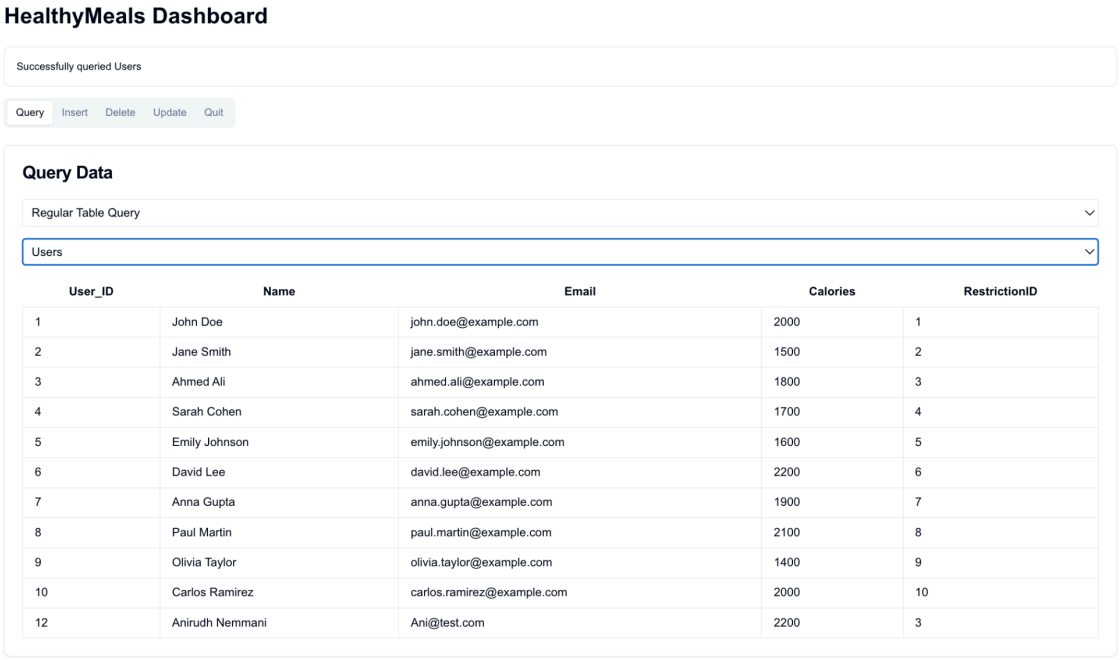
User_ID	Name	Email	Calories	RestrictionID	Actions
1	John Doe	john.doe@example.com	2000	1	Delete
2	Jane Smith	jane.smith@example.com	1500	2	Delete
3	Ahmed Ali	ahmed.ali@example.com	1800	3	Delete
4	Sarah Cohen	sarah.cohen@example.com	1700	4	Delete
5	Emily Johnson	emily.johnson@example.com	1600	5	Delete
6	David Lee	david.lee@example.com	2200	6	Delete
7	Anna Gupta	anna.gupta@example.com	1900	7	Delete
8	Paul Martin	paul.martin@example.com	2100	8	Delete
9	Olivia Taylor	olivia.taylor@example.com	1400	9	Delete
10	Carlos Ramirez	carlos.ramirez@example.com	2000	10	Delete
12	Anirudh Nemmani	Ani@test.com	2200	3	Delete
50	Muhammad Siddique	muhammad@test.com	1800	3	Delete

After deleting:



User_ID	Name	Email	Calories	RestrictionID	Actions
1	John Doe	john.doe@example.com	2000	1	Delete
2	Jane Smith	jane.smith@example.com	1500	2	Delete
3	Ahmed Ali	ahmed.ali@example.com	1800	3	Delete
4	Sarah Cohen	sarah.cohen@example.com	1700	4	Delete
5	Emily Johnson	emily.johnson@example.com	1600	5	Delete
6	David Lee	david.lee@example.com	2200	6	Delete
7	Anna Gupta	anna.gupta@example.com	1900	7	Delete
8	Paul Martin	paul.martin@example.com	2100	8	Delete
9	Olivia Taylor	olivia.taylor@example.com	1400	9	Delete
10	Carlos Ramirez	carlos.ramirez@example.com	2000	10	Delete
12	Anirudh Nemmani	Ani@test.com	2200	3	Delete

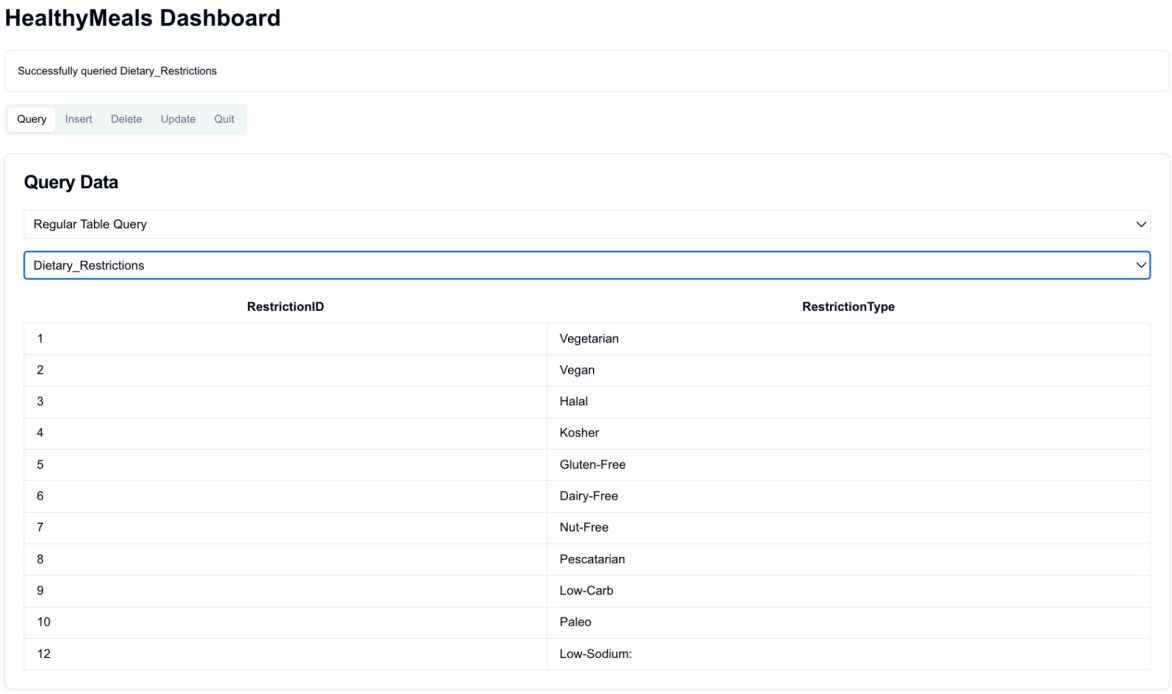
User table after:



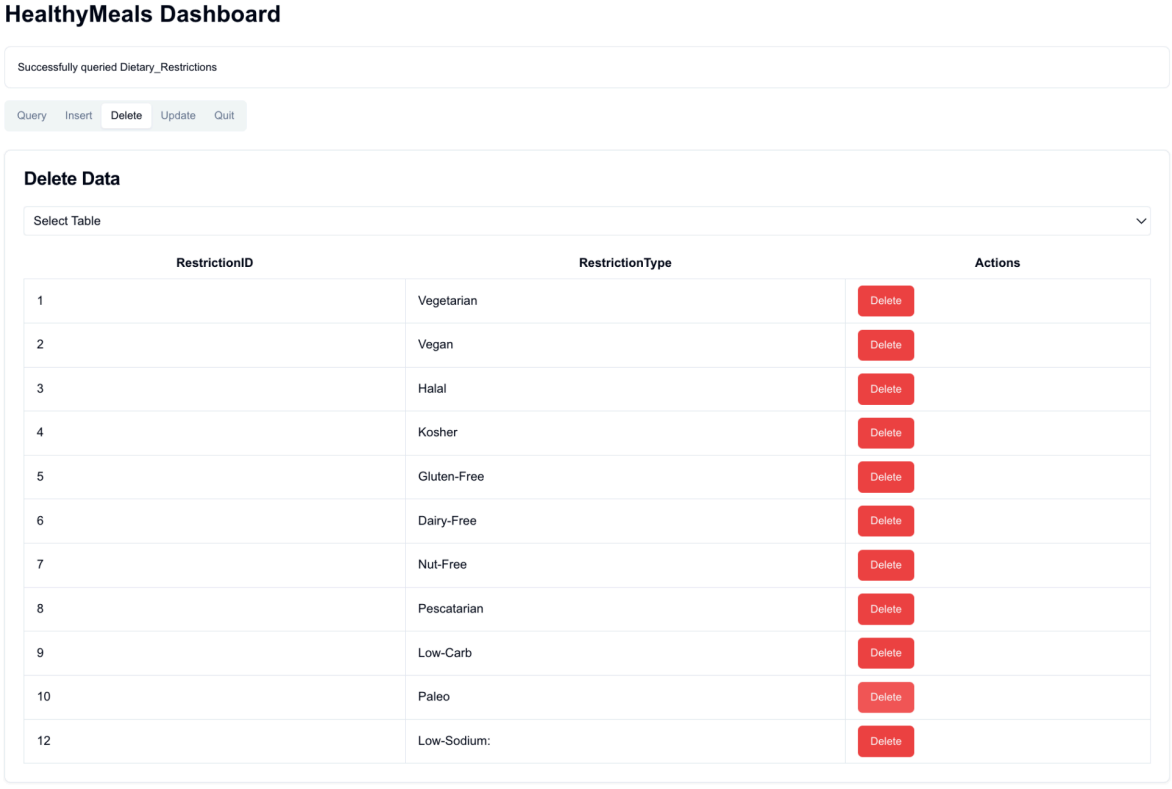
User_ID	Name	Email	Calories	RestrictionID
1	John Doe	john.doe@example.com	2000	1
2	Jane Smith	jane.smith@example.com	1500	2
3	Ahmed Ali	ahmed.ali@example.com	1800	3
4	Sarah Cohen	sarah.cohen@example.com	1700	4
5	Emily Johnson	emily.johnson@example.com	1600	5
6	David Lee	david.lee@example.com	2200	6
7	Anna Gupta	anna.gupta@example.com	1900	7
8	Paul Martin	paul.martin@example.com	2100	8
9	Olivia Taylor	olivia.taylor@example.com	1400	9
10	Carlos Ramirez	carlos.ramirez@example.com	2000	10
12	Anirudh Nemmani	Ani@test.com	2200	3

Example 2:

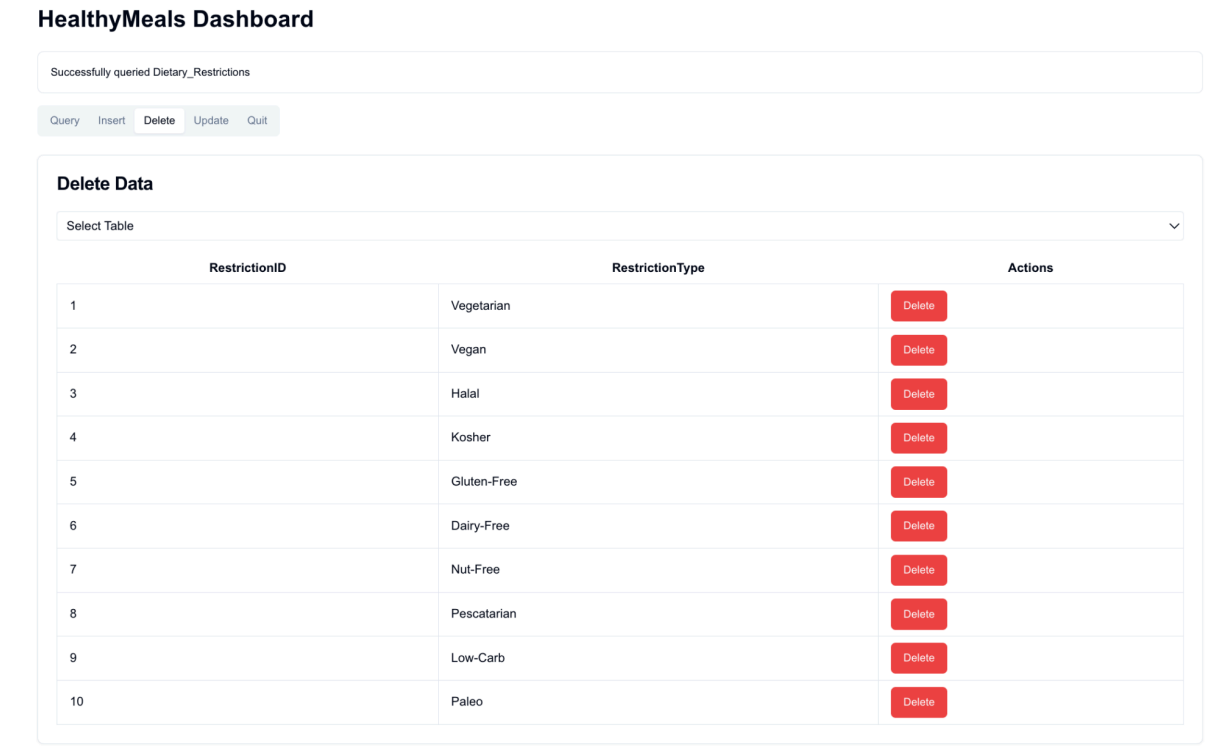
Dietary restrictions table before:



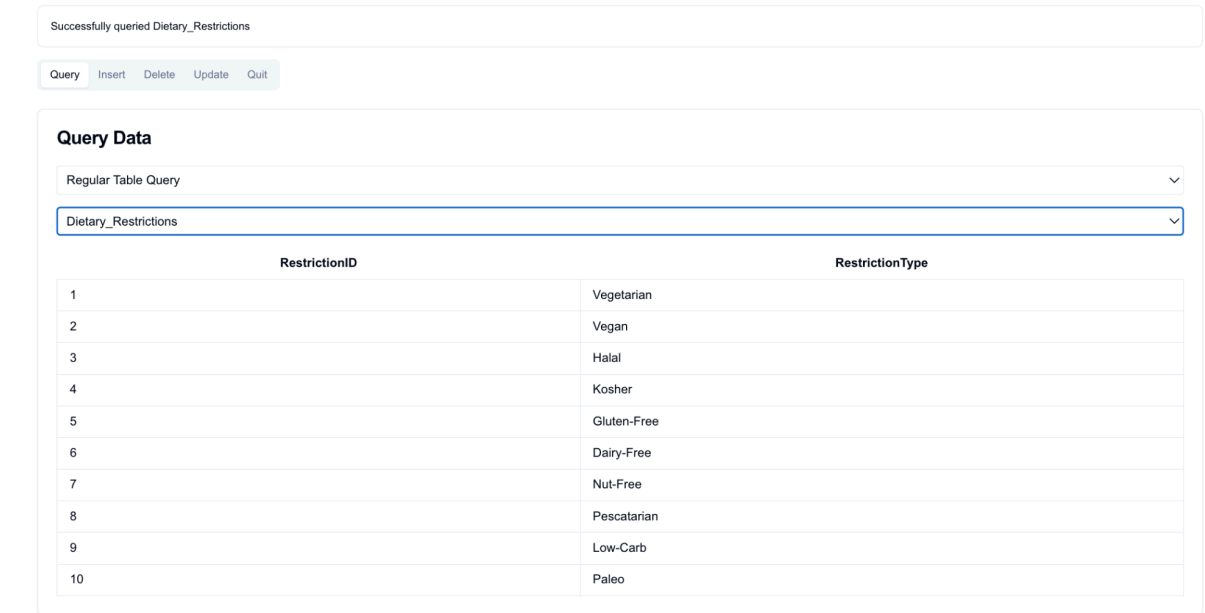
Deletion page:



After deletion:



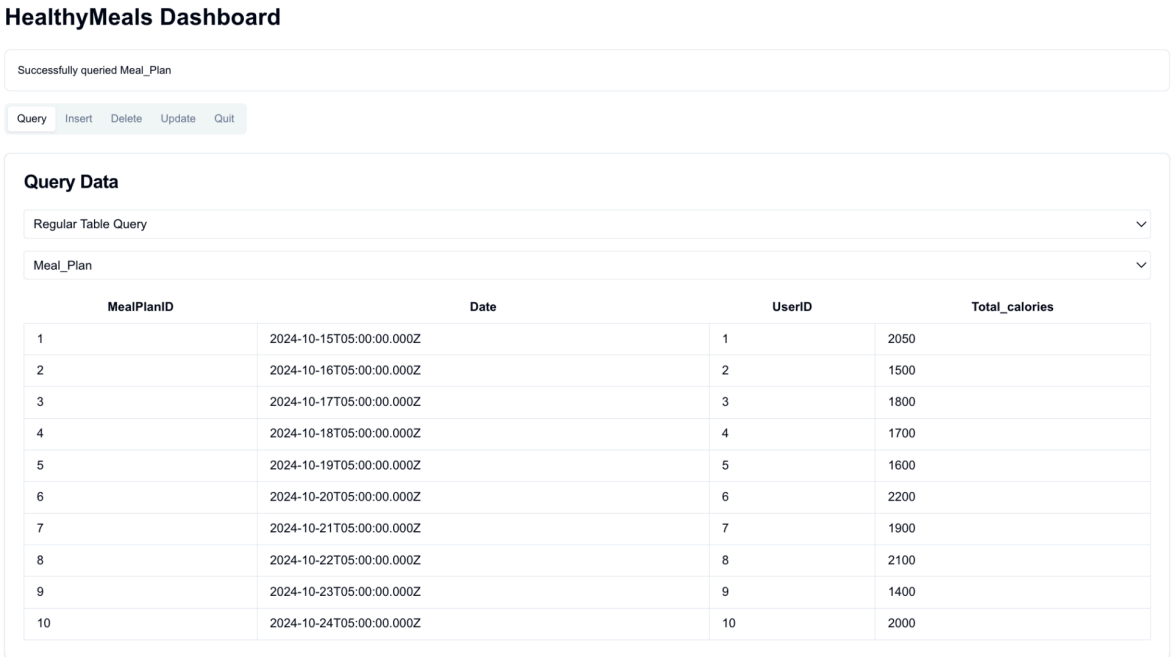
Dietary restrictions table after:



Update:

Example 1:

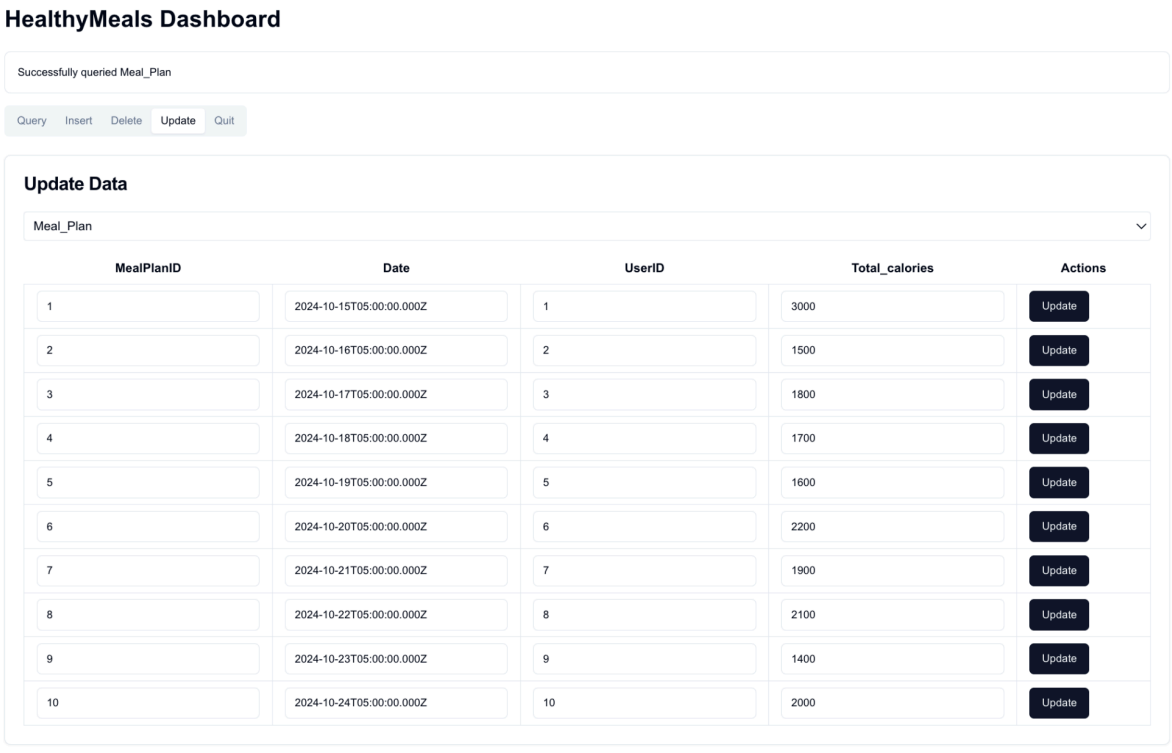
Meal_Plan table before:



MealPlanID	Date	UserID	Total_calories
1	2024-10-15T05:00:00.000Z	1	2050
2	2024-10-16T05:00:00.000Z	2	1500
3	2024-10-17T05:00:00.000Z	3	1800
4	2024-10-18T05:00:00.000Z	4	1700
5	2024-10-19T05:00:00.000Z	5	1600
6	2024-10-20T05:00:00.000Z	6	2200
7	2024-10-21T05:00:00.000Z	7	1900
8	2024-10-22T05:00:00.000Z	8	2100
9	2024-10-23T05:00:00.000Z	9	1400
10	2024-10-24T05:00:00.000Z	10	2000

Update Page:

*Updated first meal plan from 2050 total cals to 3000 total cals



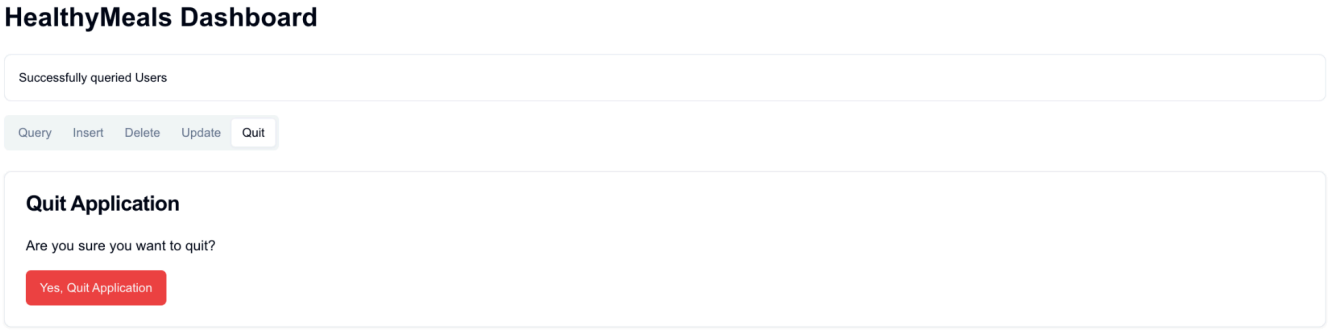
MealPlanID	Date	UserID	Total_calories	Actions
1	2024-10-15T05:00:00.000Z	1	3000	Update
2	2024-10-16T05:00:00.000Z	2	1500	Update
3	2024-10-17T05:00:00.000Z	3	1800	Update
4	2024-10-18T05:00:00.000Z	4	1700	Update
5	2024-10-19T05:00:00.000Z	5	1600	Update
6	2024-10-20T05:00:00.000Z	6	2200	Update
7	2024-10-21T05:00:00.000Z	7	1900	Update
8	2024-10-22T05:00:00.000Z	8	2100	Update
9	2024-10-23T05:00:00.000Z	9	1400	Update
10	2024-10-24T05:00:00.000Z	10	2000	Update

Meal_Plan table after:

MealPlanID	Date	UserID	Total_calories
1	2024-10-15T05:00:00.000Z	1	3000
2	2024-10-16T05:00:00.000Z	2	1500
3	2024-10-17T05:00:00.000Z	3	1800
4	2024-10-18T05:00:00.000Z	4	1700
5	2024-10-19T05:00:00.000Z	5	1600
6	2024-10-20T05:00:00.000Z	6	2200
7	2024-10-21T05:00:00.000Z	7	1900
8	2024-10-22T05:00:00.000Z	8	2100
9	2024-10-23T05:00:00.000Z	9	1400
10	2024-10-24T05:00:00.000Z	10	2000

Quit:

*If red button is pressed, the application is closed in the browser



5. Conclusion and Future Work:

Project Summary

Our Meal Planning and Nutrition System has successfully evolved from a database-focused application into a comprehensive full-stack solution. Through two development phases, we have achieved our primary objectives of creating a secure, scalable, and user-friendly platform for personalized meal planning and nutrition tracking.

Key Achievements

1. **Database Implementation**
    - Developed a robust relational database structure
    - Implemented comprehensive data validation and integrity constraints
    - Created efficient queries for meal plan generation and nutrition tracking
    - Successfully integrated user dietary preferences and restrictions
2. **Full-Stack Development**
    - Built a responsive React Native frontend interface
    - Implemented secure Node.js backend with RESTful API endpoints
    - Established reliable database connectivity

Future Work

In the immediate future, we plan to enhance our system with advanced recipe recommendations powered by machine learning algorithms, which will provide personalized suggestions based on user interaction history and dietary preferences. Additionally, we want to integrate practical features such as grocery delivery service connections, barcode scanning for nutrition information, and a comprehensive meal prep tracking system. The platform will also expand to include social features, allowing users to share recipes and meal plans while participating in supportive community forums.

6. References:

[**1**] HealthifyMe, “HealthifyMe Model- How it Works and Makes Money?,” 2024 [Online]. Available: //oyelabs.com/healthifyme-business-model/age (oyelabs.com) [Accessed: Oct 2024]

[**2**] MealBoard, “MealBoard app review: plan in advance-2020,” 2020 [Online]. Available: MealBoard - Meal and Grocery Planner app review - appPicker [Accessed: Oct 2024]