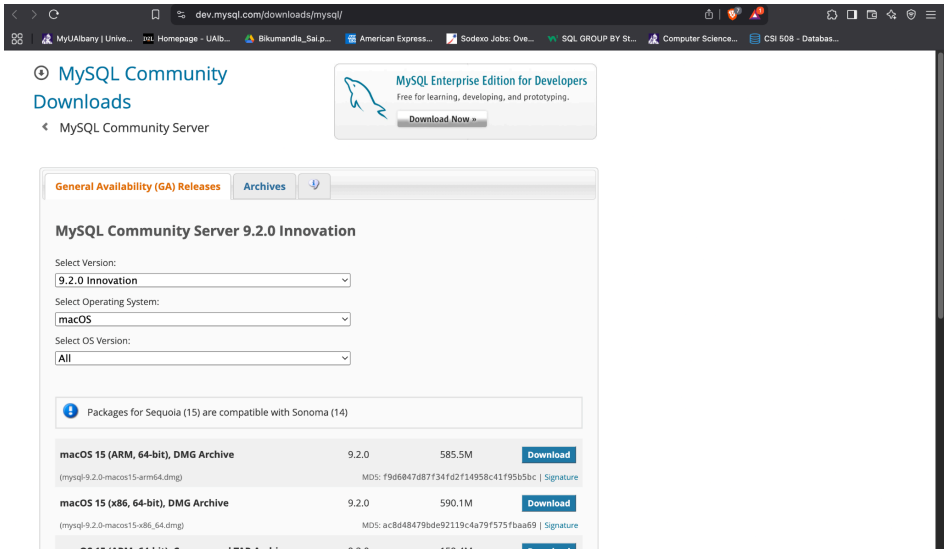
Milestone 2: DB setup and SQL

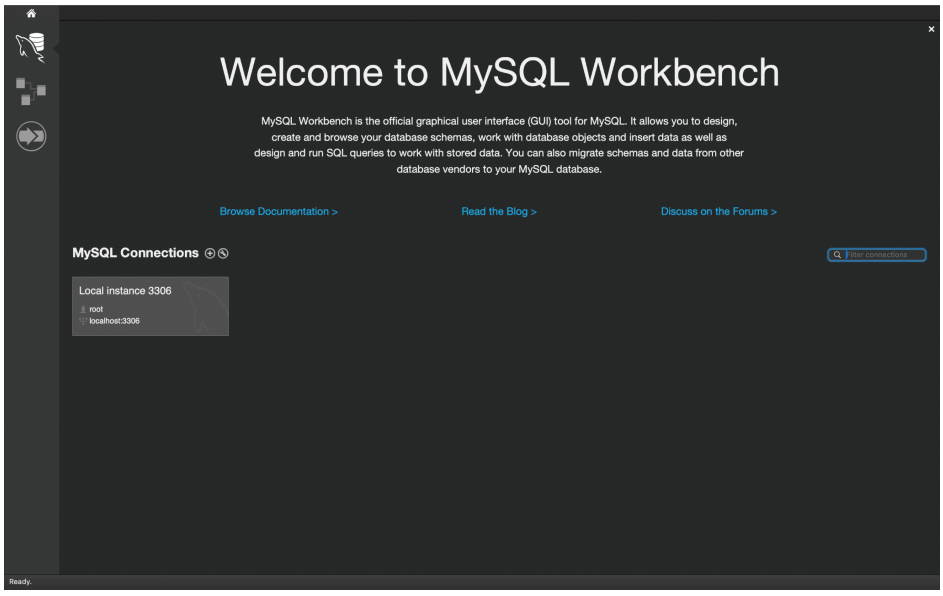
Setting Up Database Server:

1. Downloading MySQL:

   First, we downloaded the MySQL community version from https://dev.mysql.com/downloads/mysql/

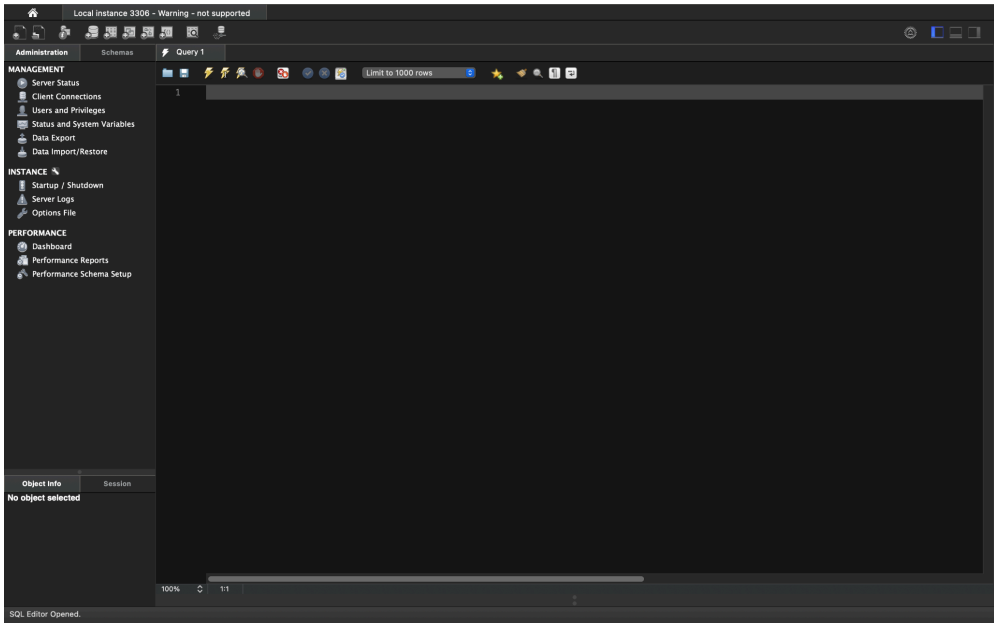


2. Downloading MySQL workbench: To visually manage the Database, we downloaded MySQL workbench. Now, we are ready to host the database locally.

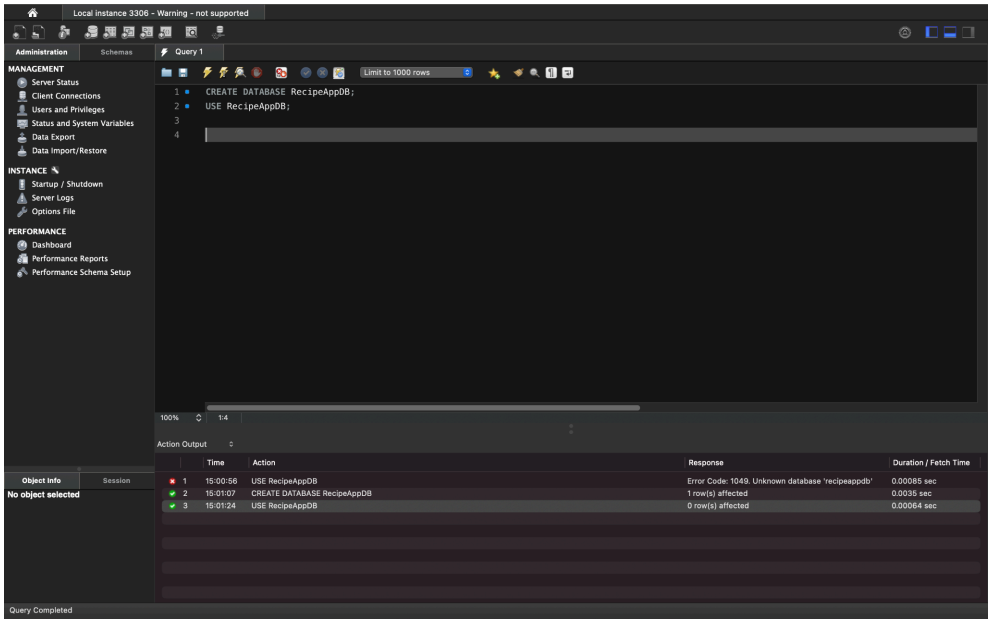


Creating Database and Working with it:

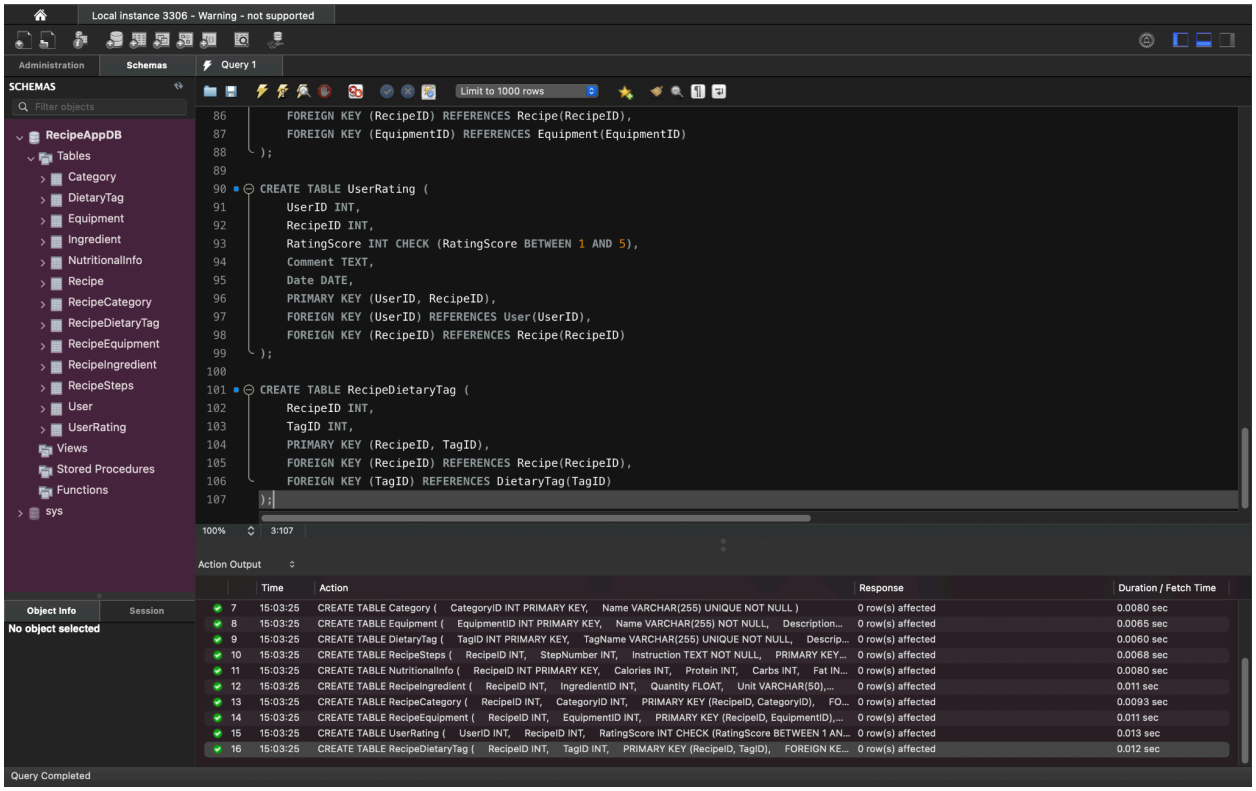
Our SQL Workbench is ready to be used now at this point:



Database Created:

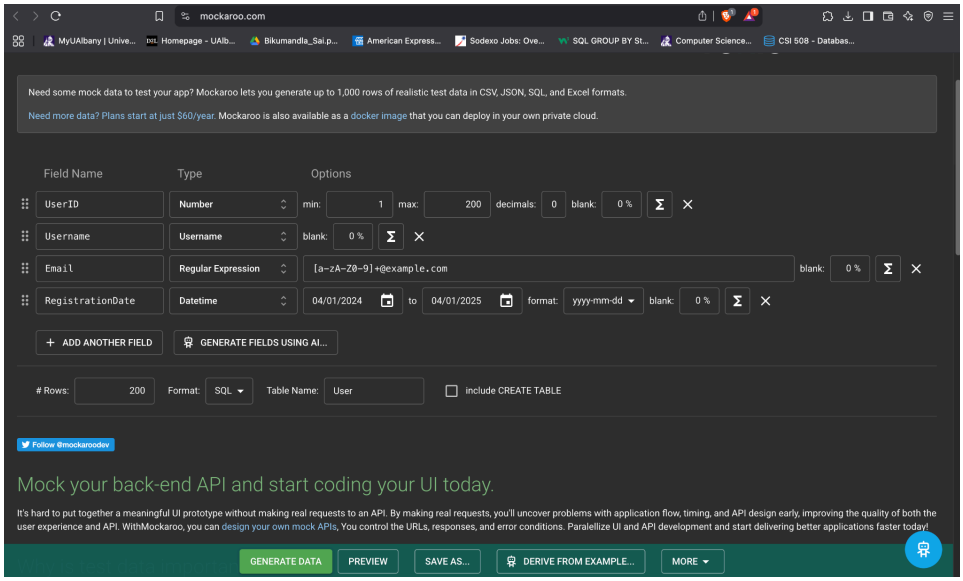


Creating Tables:



In the screenshot, You can see that we have created 13 tables.

Note: We are using https://mockaroo.com/ to procure sample user data to fill the table along with LLM models like ChatGPT. (We already mentioned that we are going to use LLMs for sample data in our Project Proposal, now we have added mockaroo.com to improve quality of sample data)



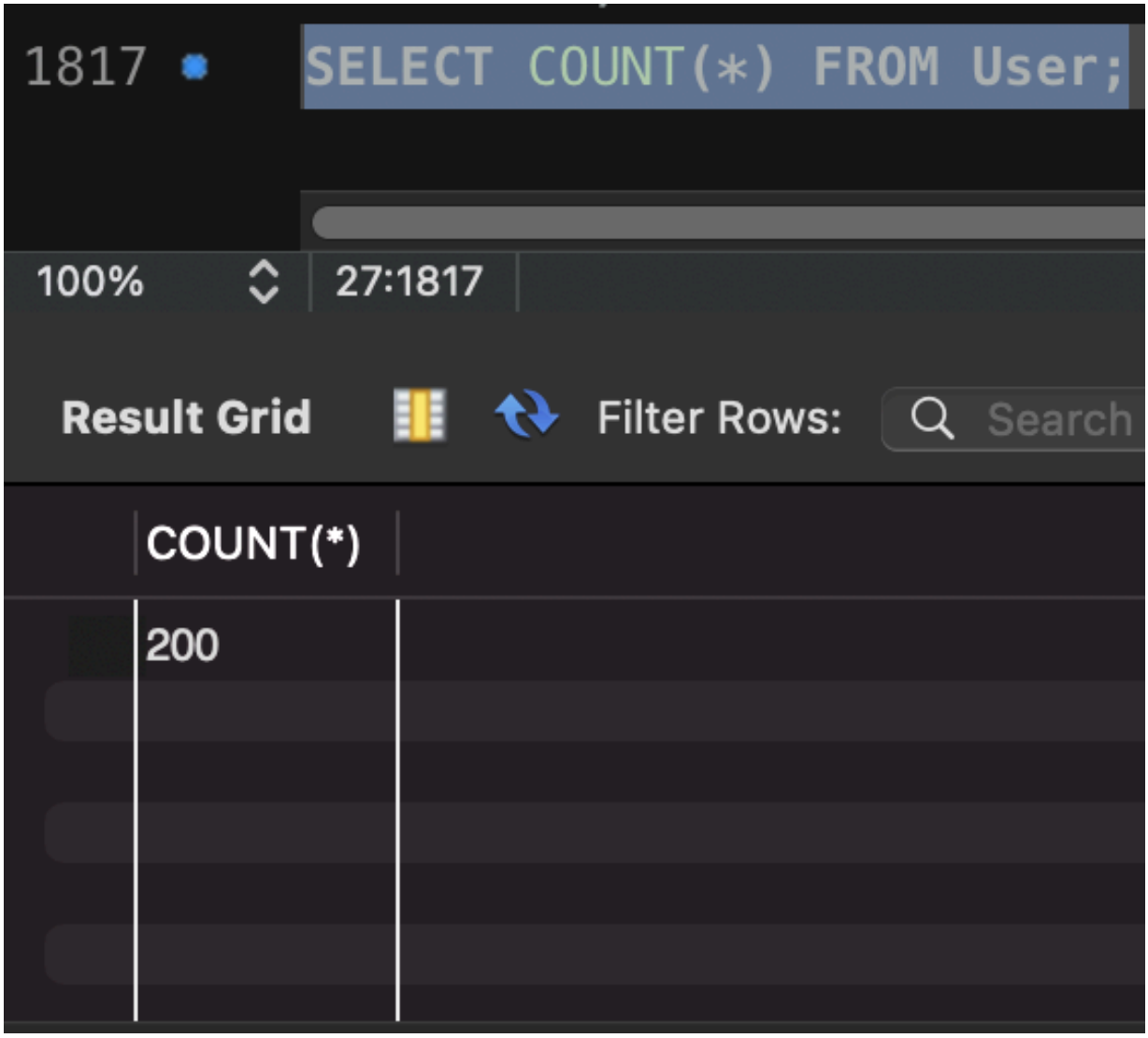
After populating each table with an adequate number of rows of data, we are now ready to show table schemas and total number of rows of data in each table.

Table Schemas:

1.DESC User;

Field	Type	Null	Key	Default	Extra
UserID	int	NO	PRI	NULL	
Username	varchar(255)	NO	UNI	NULL	
Email	varchar(255)	NO	UNI	NULL	
RegistrationDate	date	YES		NULL	

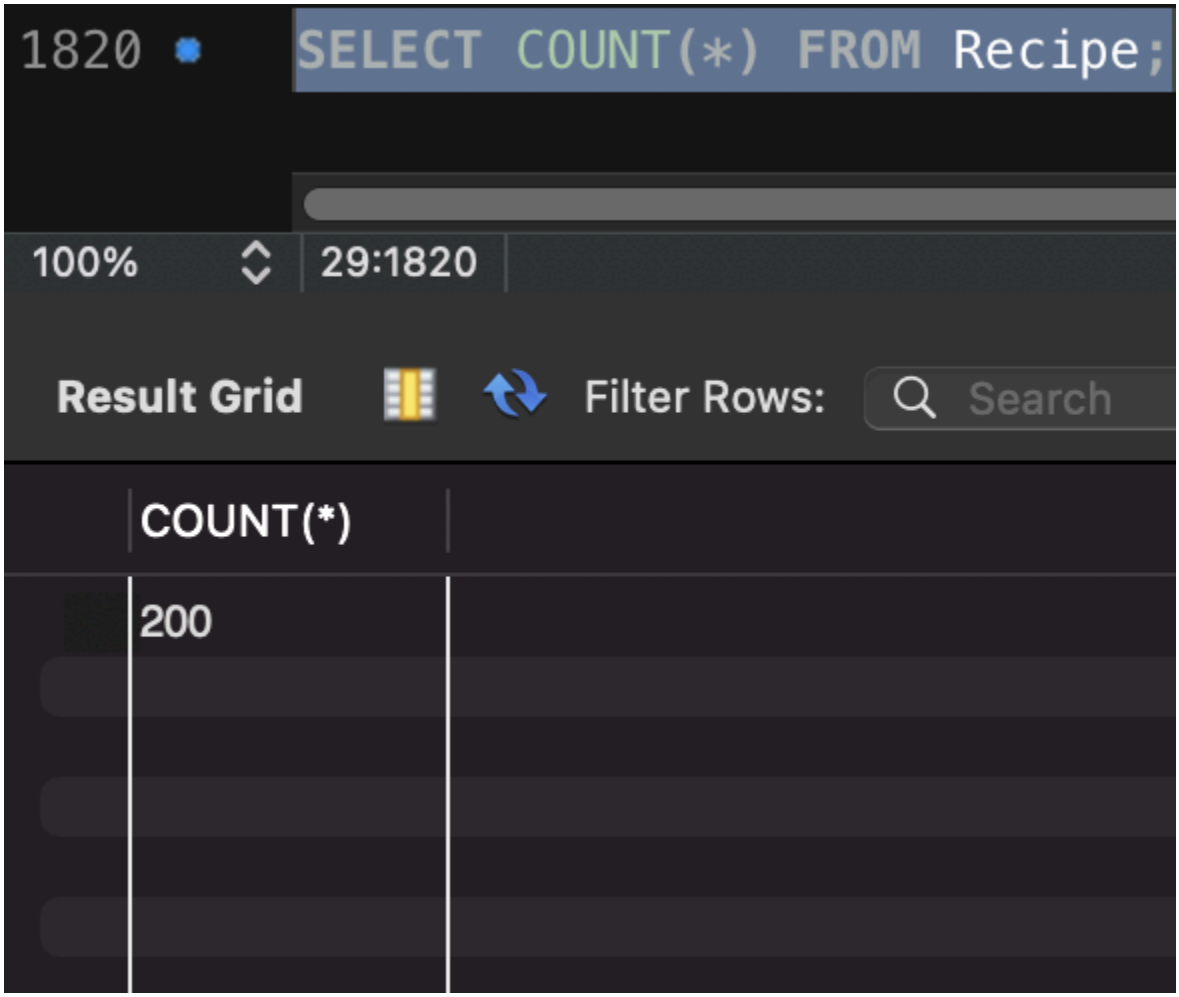
SELECT COUNT(*) FROM User;



2. DESC Recipe;

Field	Type	Null	Key	Default	Extra
RecipeID	int	NO	PRI	NULL	
Name	varchar(255)	NO		NULL	
Description	text	YES		NULL	
PrepTime	int	YES		NULL	
CookTime	int	YES		NULL	
Servings	int	YES		NULL	
UserID	int	YES	MUL	NULL	

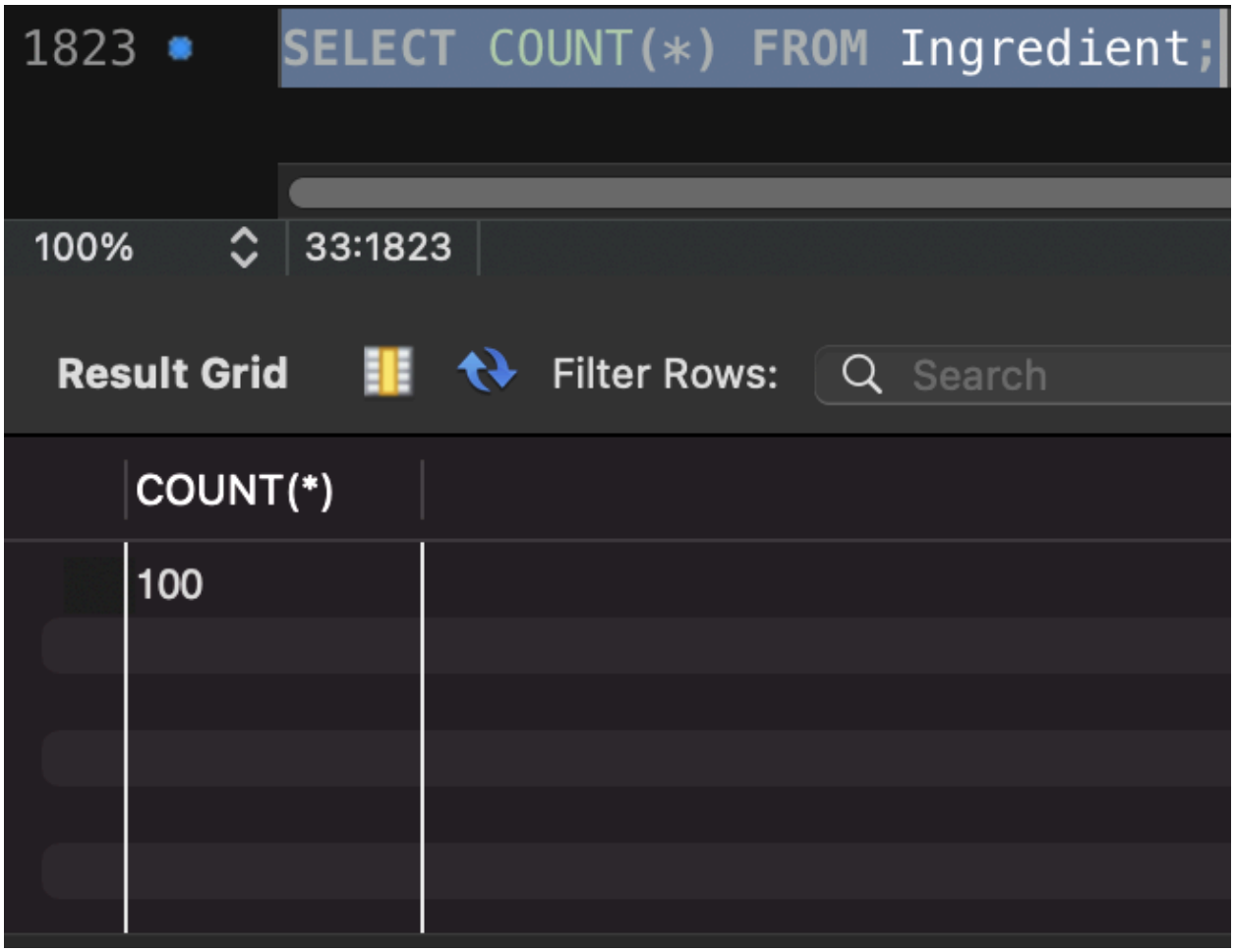
SELECT COUNT(*) FROM Recipe;



3. DESC Ingredient;

Field	Type	Null	Key	Default	Extra
IngredientID	int	NO	PRI	NULL	
Name	varchar(255)	NO		NULL	
UnitType	varchar(50)	YES		NULL	

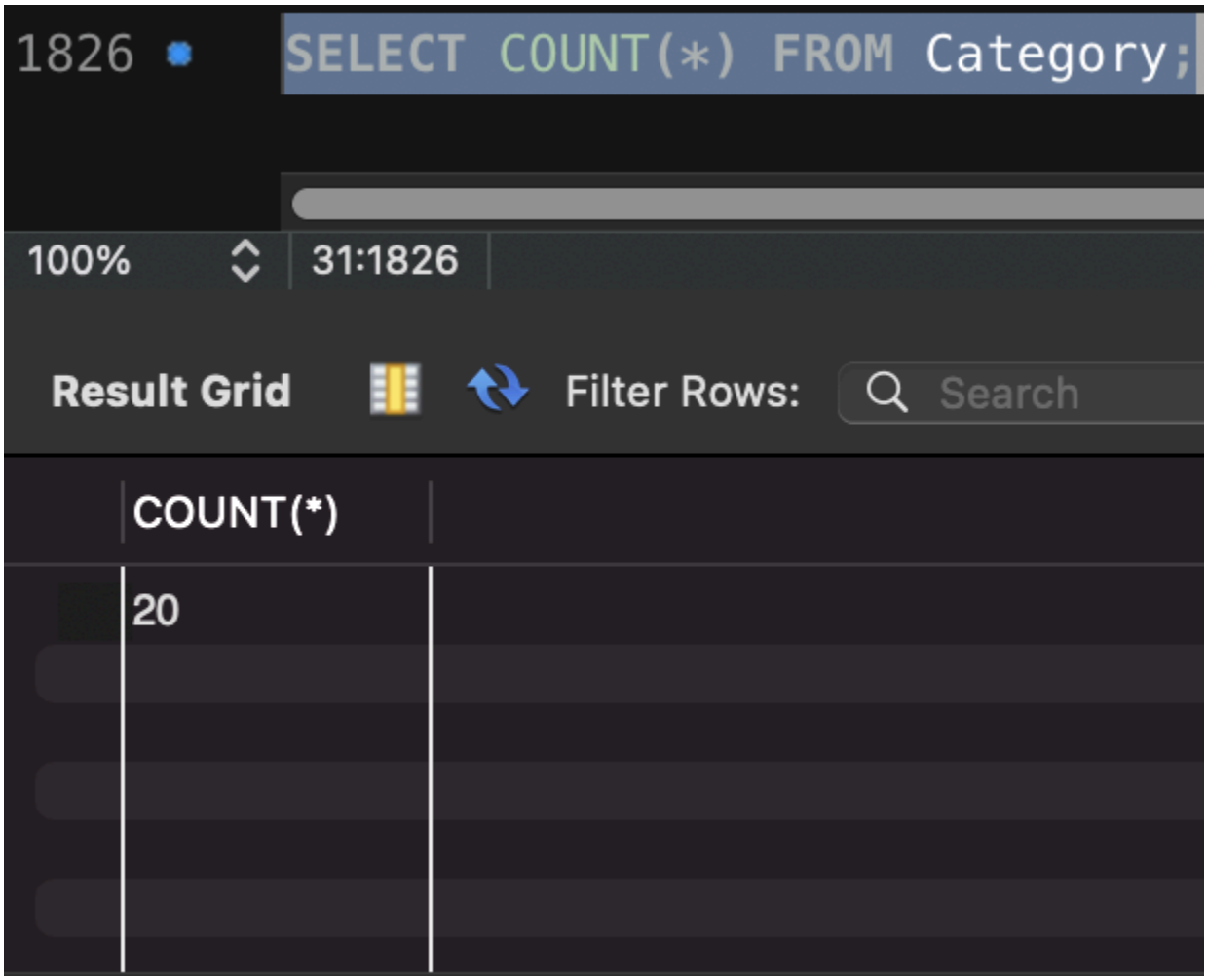
SELECT COUNT(*) FROM Ingredient;



4. DESC Category;

Field	Type	Null	Key	Default	Extra
CategoryID	int	NO	PRI	NULL	
Name	varchar(255)	NO	UNI	NULL	

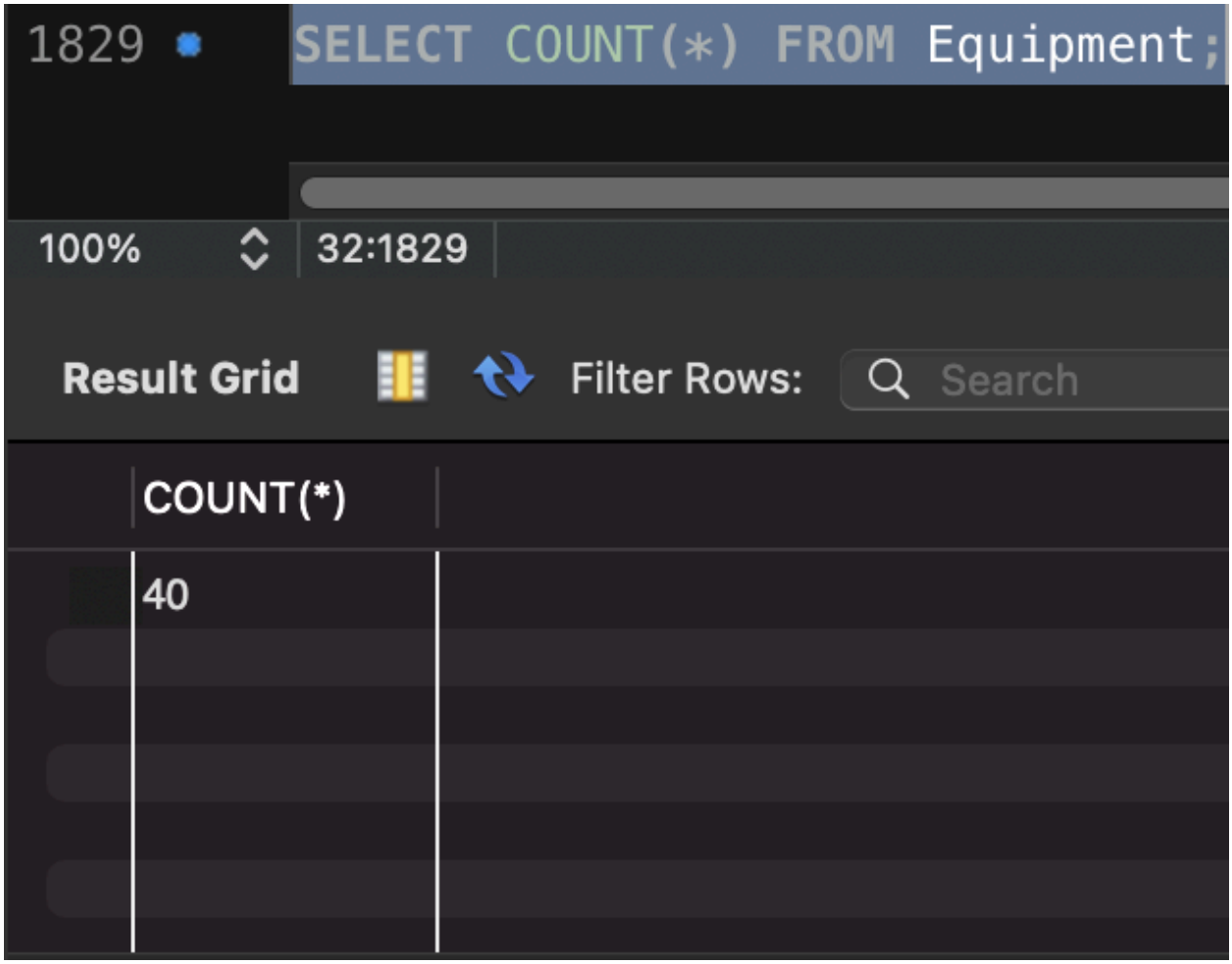
SELECT COUNT(*) FROM Category;



5. DESC Equipment;

Field	Type	Null	Key	Default	Extra
EquipmentID	int	NO	PRI	NULL	
Name	varchar(255)	NO		NULL	
Description	text	YES		NULL	

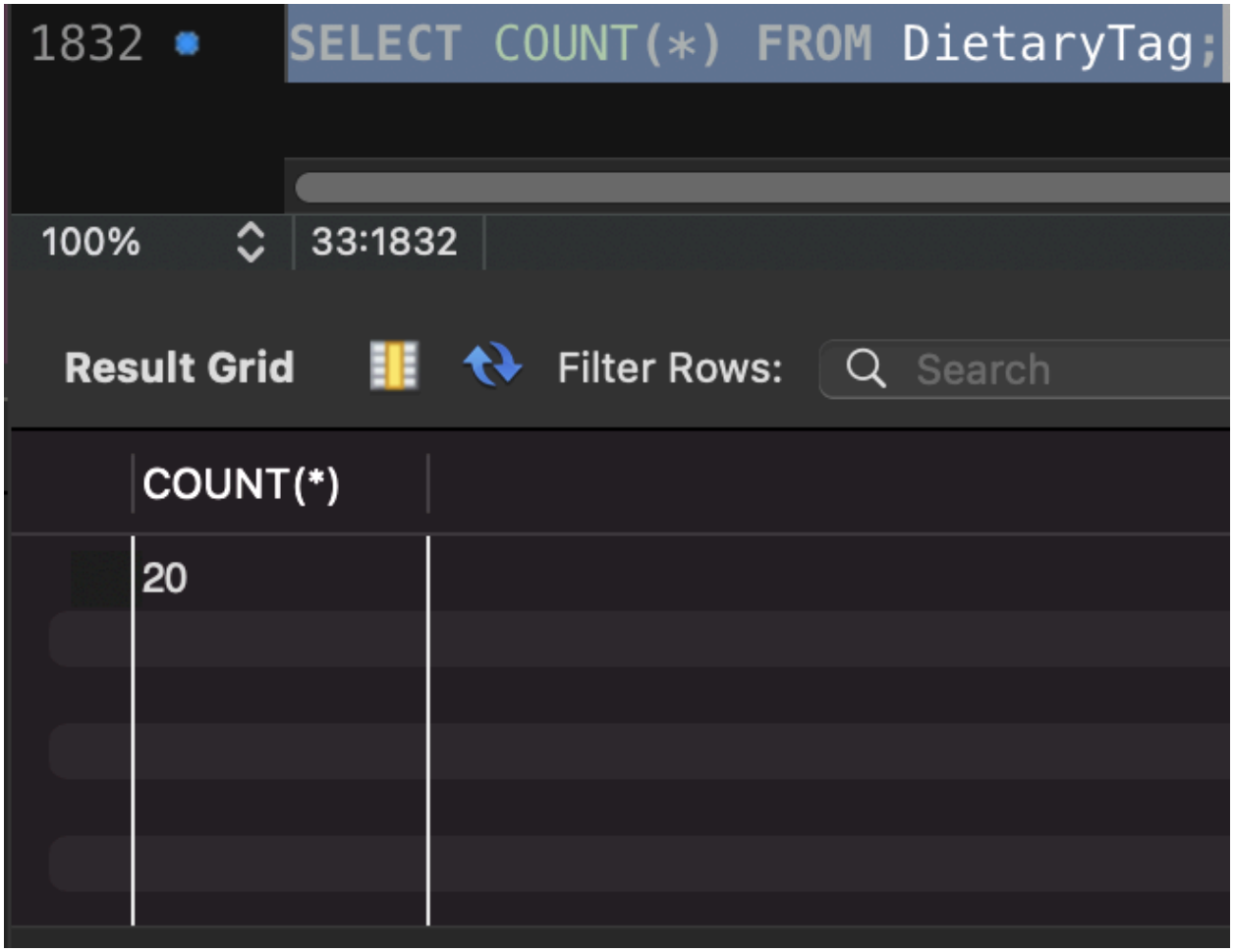
SELECT COUNT(*) FROM Equipment;



6. DESC DietaryTag;

Field	Type	Null	Key	Default	Extra
TagID	int	NO	PRI	NULL	
TagName	varchar(255)	NO	UNI	NULL	
Description	text	YES		NULL	

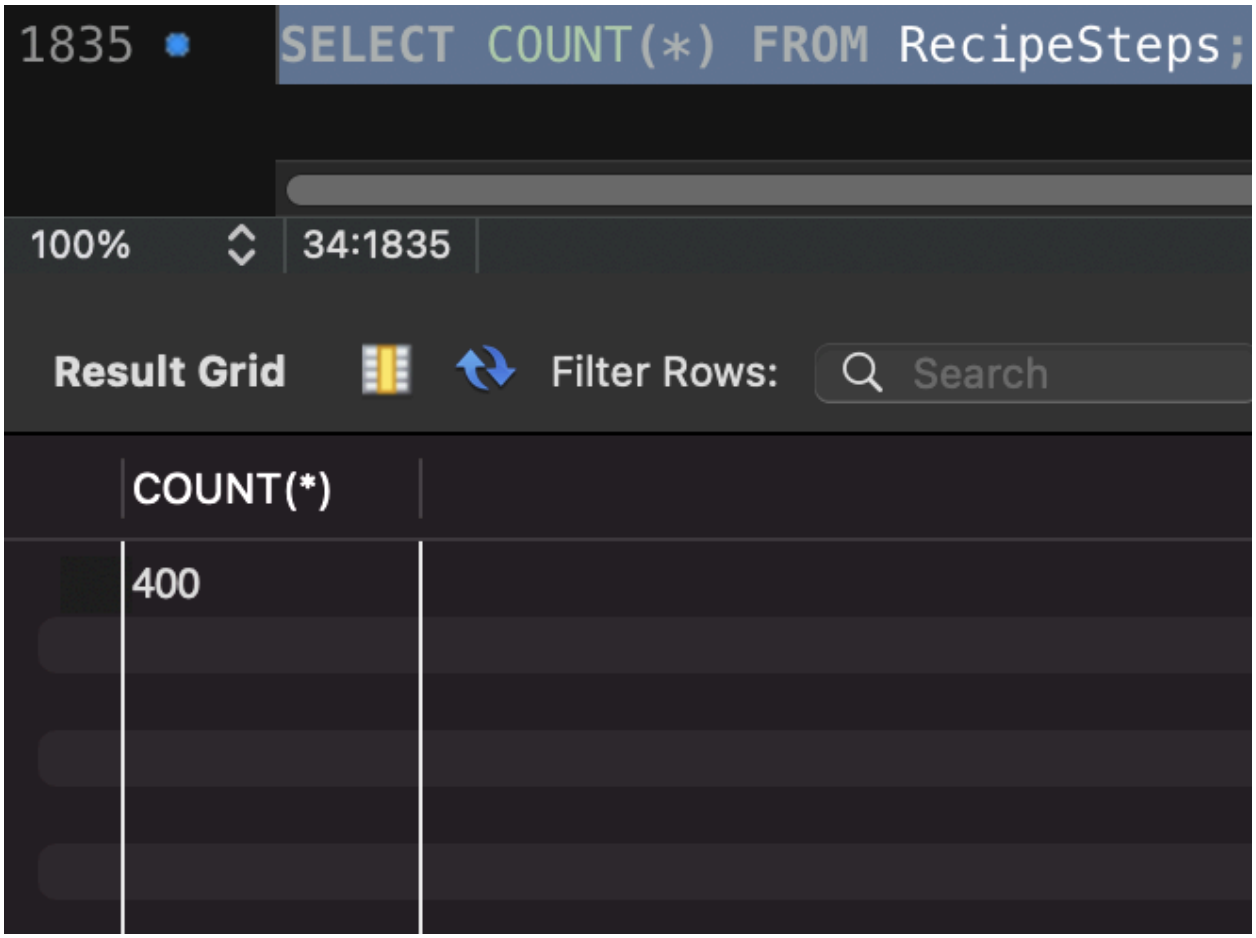
SELECT COUNT(*) FROM DietaryTag;



7. DESC RecipeSteps;

Field	Type	Null	Key	Default	Extra
RecipeID	int	NO	PRI	NULL	
StepNumber	int	NO	PRI	NULL	
Instruction	text	NO		NULL	

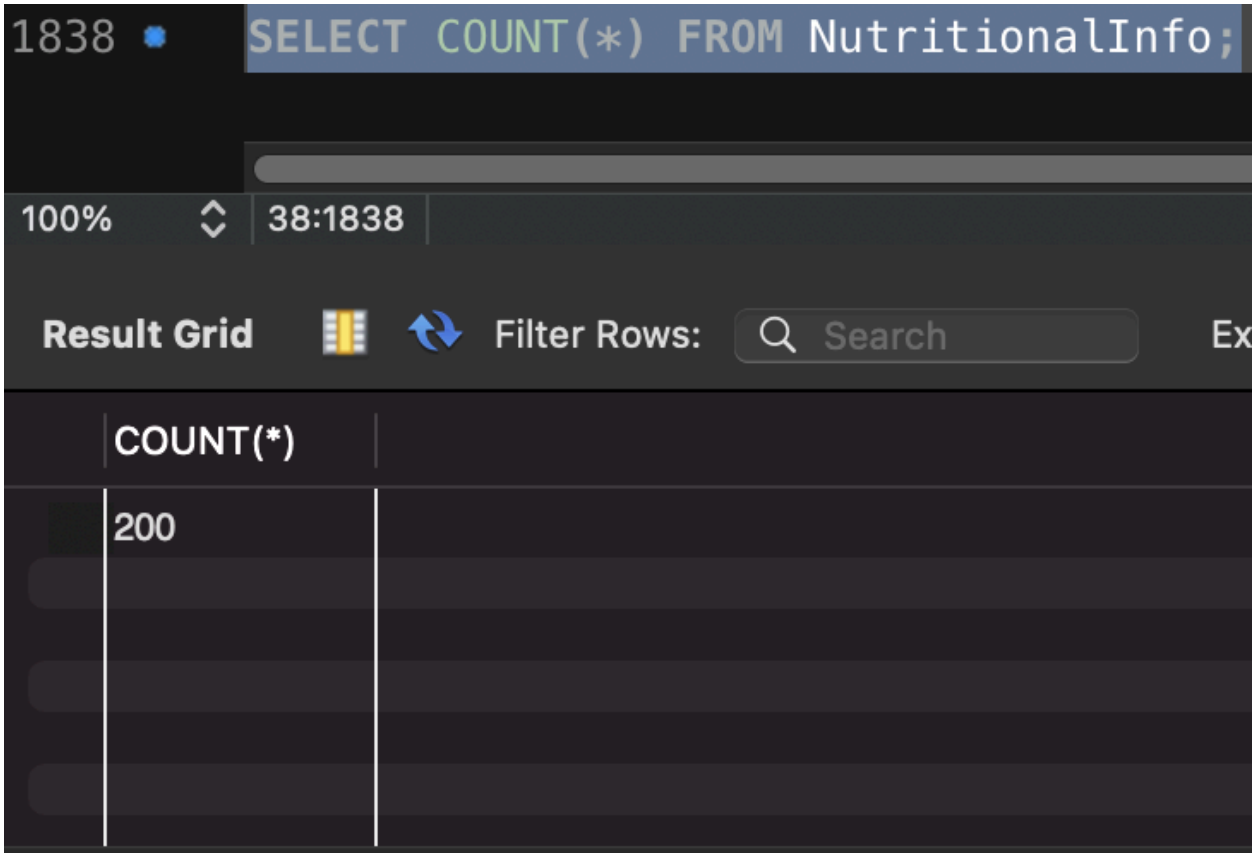
SELECT COUNT(*) FROM RecipeSteps;



8. DESC NutritionalInfo;

Field	Type	Null	Key	Default	Extra
RecipeID	int	NO	PRI	NULL	
Calories	int	YES		NULL	
Protein	int	YES		NULL	
Carbs	int	YES		NULL	
Fat	int	YES		NULL	

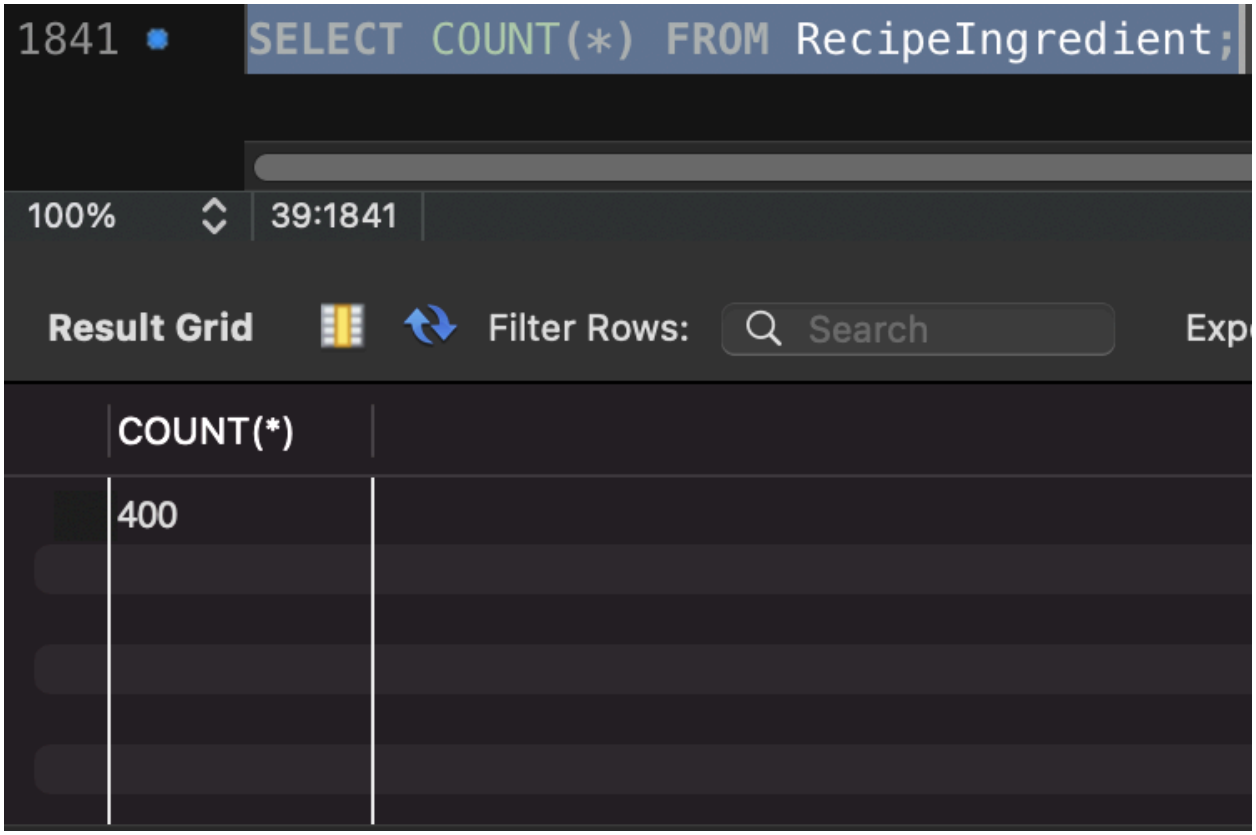
SELECT COUNT(*) FROM NutritionalInfo;



9. DESC RecipeIngredient;

Field	Type	Null	Key	Default	Extra
RecipeID	int	NO	PRI	NULL	
IngredientID	int	NO	PRI	NULL	
Quantity	float	YES		NULL	
Unit	varchar(50)	YES		NULL	

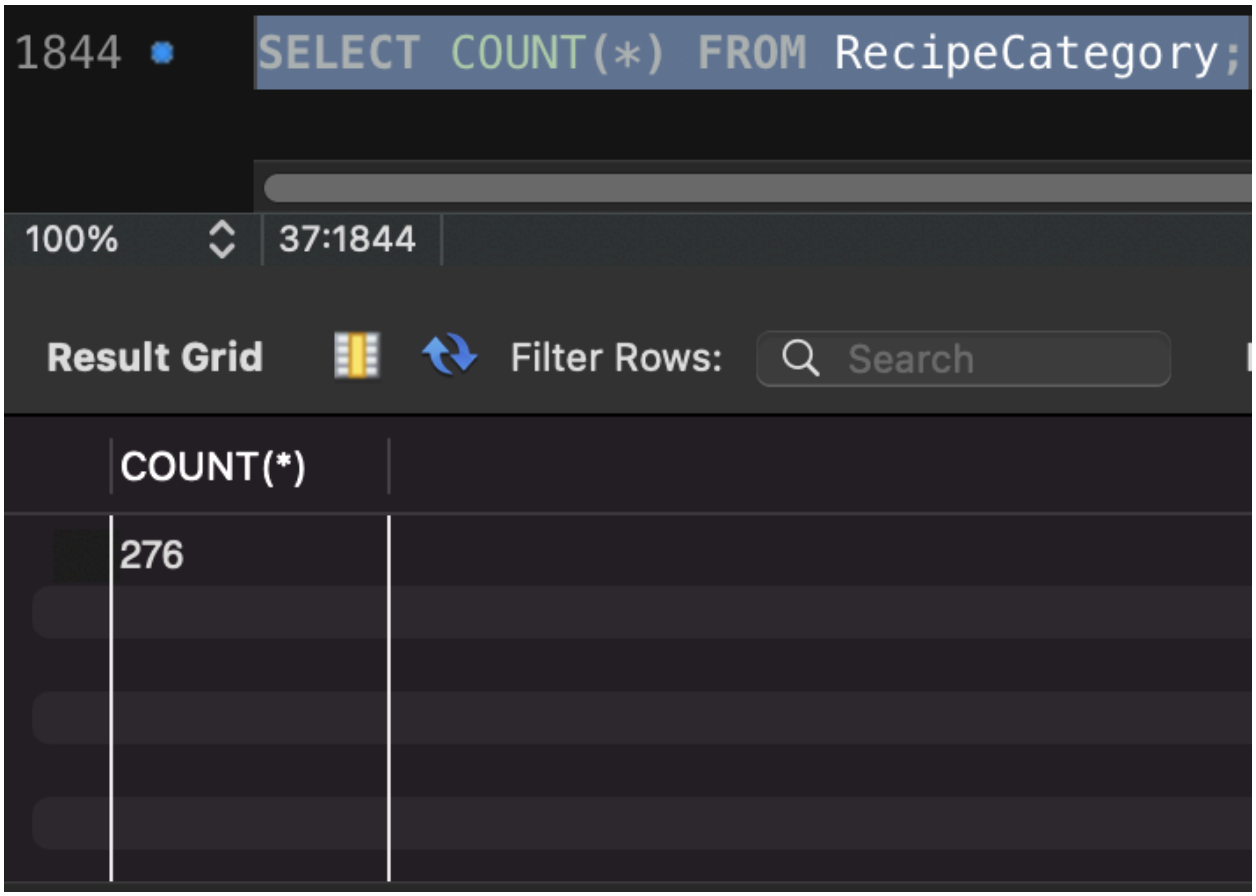
SELECT COUNT(*) FROM RecipeIngredient;



10. DESC RecipeCategory;

Field	Type	Null	Key	Default	Extra
RecipeID	int	NO	PRI	NULL	
CategoryID	int	NO	PRI	NULL	

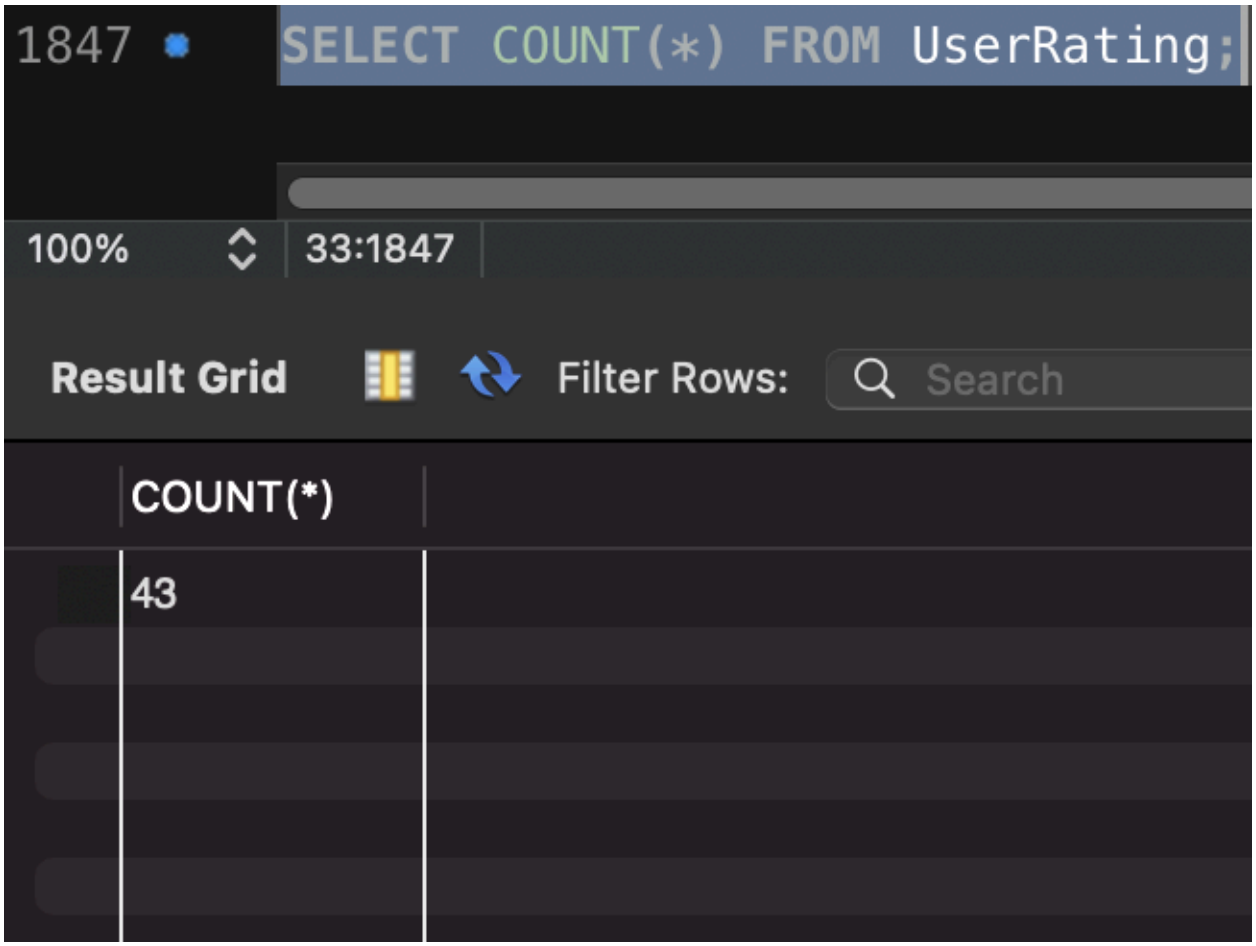
SELECT COUNT(*) FROM RecipeCategory;



11. DESC UserRating;

Field	Type	Null	Key	Default	Extra
UserID	int	NO	PRI	NULL	
RecipeID	int	NO	PRI	NULL	
RatingScore	int	YES		NULL	
Comment	text	YES		NULL	
Date	date	YES		NULL	

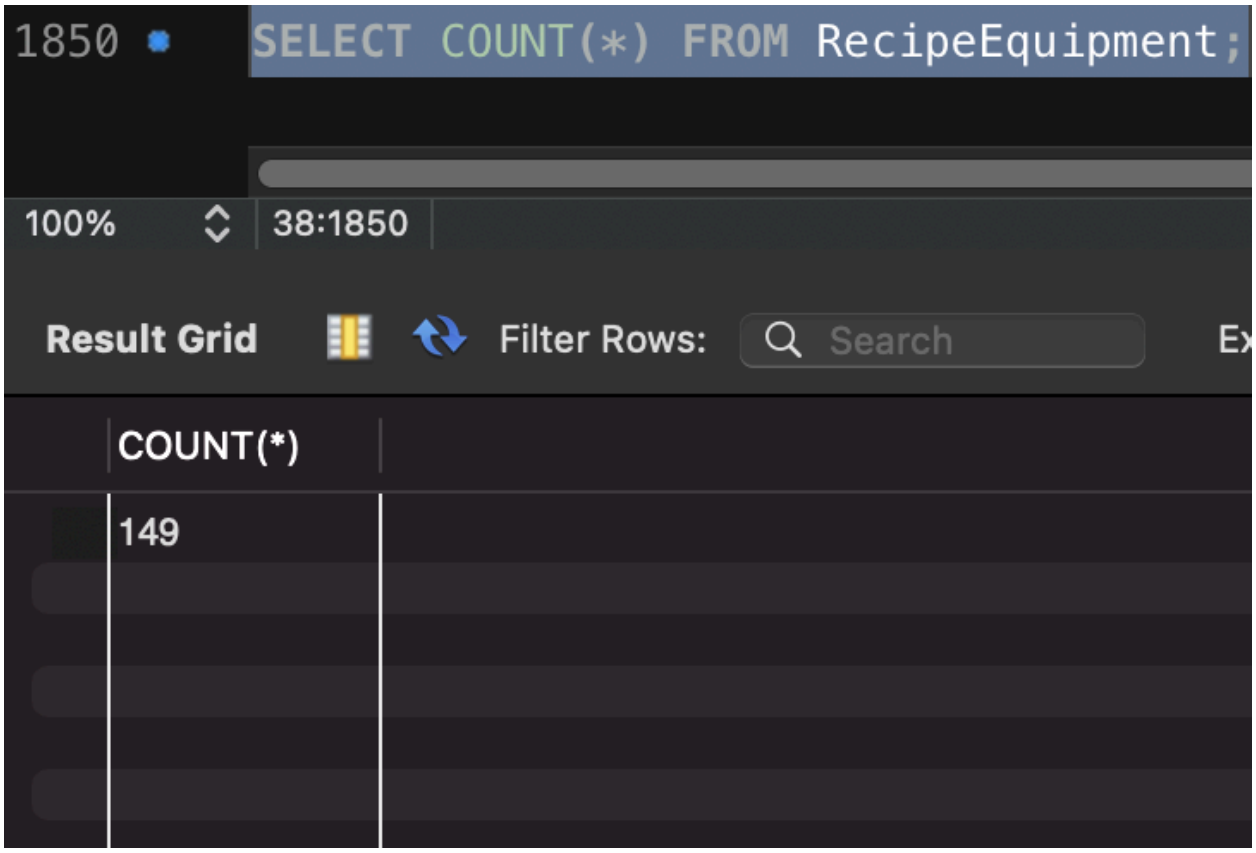
SELECT COUNT(*) FROM UserRating;



12. DESC RecipeEquipment;

Field	Type	Null	Key	Default	Extra
RecipeID	int	NO	PRI	NULL	
EquipmentID	int	NO	PRI	NULL	

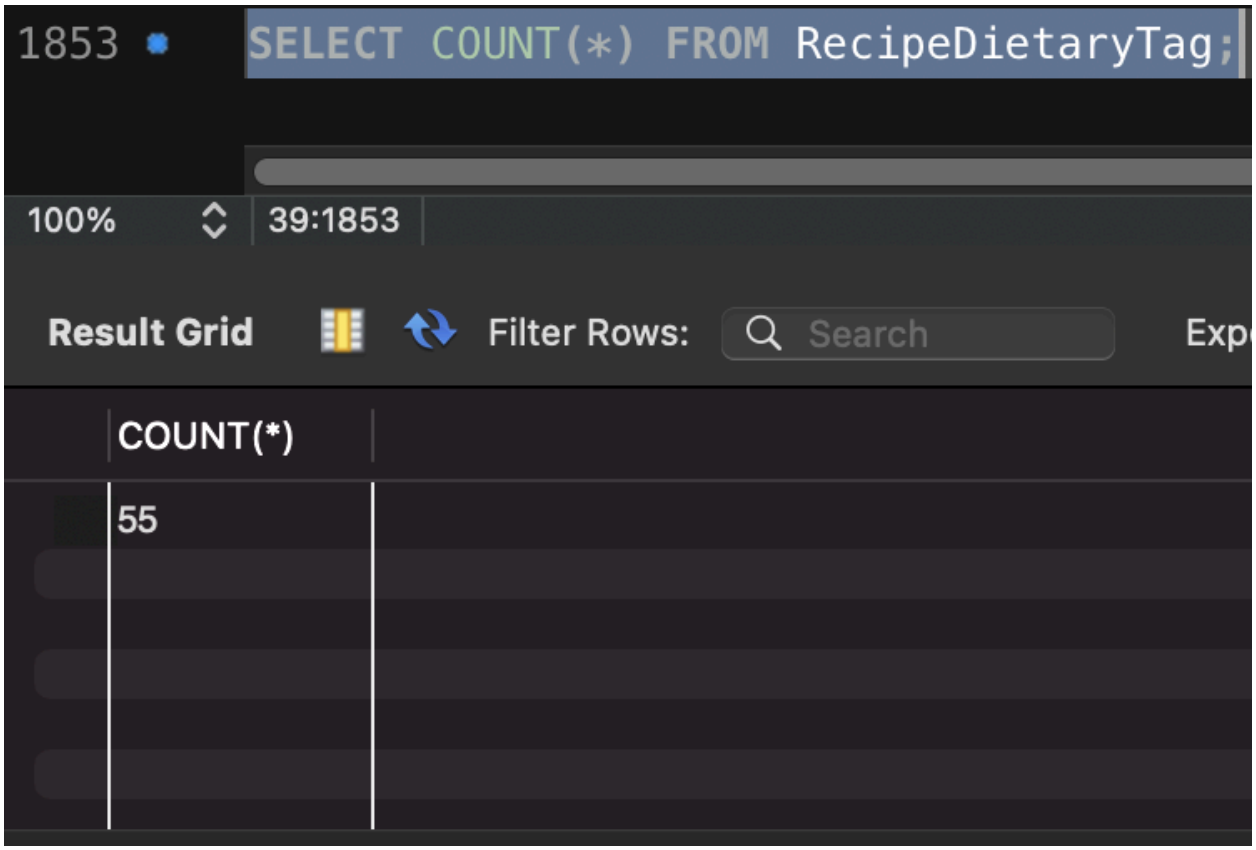
SELECT COUNT(*) FROM RecipeEquipment;



13. DESC RecipeDietaryTag;

Field	Type	Null	Key	Default	Extra
RecipeID	int	NO	PRI	NULL	
TagID	int	NO	PRI	NULL	

SELECT COUNT(*) FROM RecipeDietaryTag;



Justifications & Explanations:

Our database consists of **100 users** with different profiles with various usernames like "steresi9" (joined Aug 2024), "gcottee4" (joined Dec 2024), and "alowrej" (joined Mar 2025) to provide authentic and varied user interaction. The **Recipe table** (Which is the core of our project) consists of **300 recipes** for categories like breakfast, lunch, dinner, and sweets, including examples like: "Spaghetti Carbonara" (RecipeID 102, prep 20 mins), "Miso Soup" (RecipeID 170, prep 10 mins), and "Chicken Tikka Masala" (RecipeID 175, prep 40 mins), showing a wide range of cooking times and cuisines.

The **Ingredient table** has **100 entries** such as "Olive Oil" (which is usually used in salads and sautéing), "Arborio Rice" (usually used in risotto), and "Coconut Milk" (usually used in curries), so that the recipes hold accurate ingredients. **Categories** such as "Appetizer", "Soup", and "International allows the user to browse by meal. **Diet tags** such as "Vegan", "Low-Carb", and "Pescatarian" allows filtering. The **Equipment table** is filled with appliances like "Dutch Oven", "Stand Mixer", and "Grill Pan".

The **RecipeIngredient table** refers to the recipes with their ingredient: For example, "Shrimp Scampi" (RecipeID 120) uses "Shrimp (278)" and "Garlic (209)," while "Vegetable Stir-Fry" (RecipeID 105) uses "Bell Pepper (230)" and "Soy Sauce (225)." User ratings include both positive ("Creamy and delicious!" for RecipeID 126) and negative reviews ("Underseasoned" for RecipeID 124), showing real user behavior.

Note: All examples used above are accurate with the existing database setup. They are mentioned here to show how the database is designed while demonstrating it with real-world data.

Here is the Sample Data stored in each table:

1. User Table

UserID	Username	Email	RegistrationDa...
1	kbadder4y	kbadder4y@example.com	2024-07-27
2	ebruhnicket	ebruhnicket@example.com	2024-12-06
3	lfothergill16	lfothergill16@example.com	2024-04-09
4	bshakespeare3e	bshakespeare3e@example.com	2024-05-26
6	idevere31	idevere31@example.com	2024-08-16
7	sgoodbairn2d	sgoodbairn2d@example.com	2025-01-25
9	sjannequin1m	sjannequin1m@example.com	2024-05-01

2. Recipe Table

RecipeID	Name	Description	PrepTime	CookTime	Servings	UserID
101	Classic Chocolate Cake	Rich and moist chocolate cake	30	60	8	96
102	Spaghetti Carbonara	Creamy Italian pasta with eggs and cheese	20	25	4	170
103	Garden Salad	Fresh mix of seasonal vegetables	15	0	4	92
104	Beef Tacos	Seasoned ground beef in crispy tortillas	25	20	6	62
105	Vegetable Stir-Fry	Sautéed veggies with soy sauce	20	15	4	199
106	Chicken Alfredo	Creamy pasta with grilled chicken	25	30	6	83
107	Blueberry Pancakes	Fluffy pancakes with fresh blueberries	15	20	4	184

3. Ingredient Table

IngredientID	Name	UnitType
201	Flour	Cup
202	Sugar	Cup
203	Eggs	Piece
204	Milk	Liter
205	Butter	Gram
206	Salt	Teaspoon
207	Black Pepper	Teaspoon

4. Category Table

CategoryID	Name
303	Appetizer
309	Beverage
310	Bread
307	Breakfast
308	Brunch
315	Casserole
301	Dessert

5. Equipment Table

EquipmentID	Name	Description
401	Oven	For baking, roasting, and heating dishes
402	Mixing Bowl	Stainless steel or glass bowl for combining ingre…
403	Chef Knife	Sharp multi-purpose knife for chopping and slicing
404	Blender	For blending smoothies, soups, and sauces
405	Microwave	For reheating and quick cooking tasks
406	Food Processor	For chopping, shredding, and mixing dough
407	Stand Mixer	Heavy-duty mixer for doughs and batters

6. DietaryTag Table:

TagID	TagName	Description
501	Vegetarian	No meat or fish ingredients
502	Vegan	No animal products (meat, dairy, eggs, honey)
503	Gluten-Free	No gluten-containing ingredients
504	Dairy-Free	No milk or dairy products
505	Nut-Free	No tree nuts or peanuts
506	Low-Carb	Reduced carbohydrate content
507	Keto	High-fat, very low-carb for ketosis

7. RecipeSteps Table:

RecipeID	StepNumber	Instruction
101	1	Prepare ingredients
101	2	Cook as directed
102	1	Chop vegetables
102	2	Cook until tender
103	1	Mix ingredients
103	2	Bake in oven
104	1	Boil water

8. NutritionalInfo Table:

RecipeID	Calories	Protein	Carbs	Fat
101	450	6	60	20
102	300	8	45	12
103	250	3	30	10
104	600	10	80	25
105	180	2	25	8
106	400	5	55	15
107	350	4	50	12

9. RecipeIngredient

RecipeID	IngredientID	Quantity	Unit
101	201	2.5	Cup
101	219	0.75	Cup
102	201	2.25	Cup
102	287	300	Gram
103	201	1.5	Cup
103	295	2	Cup
104	275	500	Gram

10. RecipeCategory Table

RecipeID	CategoryID
101	301
102	301
103	301
104	301
105	301
107	301
122	301

11. UserRating Table

UserID	RecipeID	RatingScore	Comment	Date
2	119	4	Spicy but good	2024-10-29
7	147	2	Broth was bland	2024-04-16
11	125	4	Quick and tasty	2024-05-30
14	101	5	Best chocolate cake ever!	2024-04-05
17	120	4	Easy weeknight meal	2024-11-05
19	143	5	Decadent dessert	2024-12-25
22	128	5	Best guacamole!	2024-09-18

12. RecipeEquipment Table

RecipeID	EquipmentID
101	401
102	401
104	401
105	401
107	401
109	401
114	401

13. RecipeDietaryTag Table

RecipeID	TagID
101	501
102	501
103	501
104	501
105	501
106	501
107	501

Here is the list on statements used to create all the above Normalized Tables:

```
CREATE TABLE User (
    UserID INT PRIMARY KEY,
    Username VARCHAR(255) UNIQUE NOT NULL,
    Email VARCHAR(255) UNIQUE NOT NULL,
    RegistrationDate DATE
);

CREATE TABLE Recipe (
    RecipeID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Description TEXT,
    PrepTime INT,
    CookTime INT,
    Servings INT,
    UserID INT,
    FOREIGN KEY (UserID) REFERENCES User(UserID)
);

CREATE TABLE Ingredient (
    IngredientID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    UnitType VARCHAR(50)
);
```

```
CREATE TABLE Category (
   CategoryID INT PRIMARY KEY,
   Name VARCHAR(255) UNIQUE NOT NULL
);

CREATE TABLE Equipment (
   EquipmentID INT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   Description TEXT
);

CREATE TABLE DietaryTag (
   TagID INT PRIMARY KEY,
   TagName VARCHAR(255) UNIQUE NOT NULL,
   Description TEXT
);

-- Weak Entities
CREATE TABLE RecipeSteps (
   RecipeID INT,
   StepNumber INT,
   Instruction TEXT NOT NULL,
   PRIMARY KEY (RecipeID, StepNumber),
   FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID)
);

CREATE TABLE NutritionalInfo (
   RecipeID INT PRIMARY KEY,
   Calories INT,
   Protein INT,
   Carbs INT,
   Fat INT,
   FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID)
);
-- Relationship Tables
CREATE TABLE RecipeIngredient (
   RecipeID INT,
   IngredientID INT,
   Quantity FLOAT,
   Unit VARCHAR(50),
   PRIMARY KEY (RecipeID, IngredientID),
   FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID),
   FOREIGN KEY (IngredientID) REFERENCES Ingredient(IngredientID)
);
```

```
CREATE TABLE RecipeCategory (
    RecipeID INT,
    CategoryID INT,
    PRIMARY KEY (RecipeID, CategoryID),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID),
    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID)
);

CREATE TABLE RecipeEquipment (
    RecipeID INT,
    EquipmentID INT,
    PRIMARY KEY (RecipeID, EquipmentID),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID)
);

CREATE TABLE UserRating (
    UserID INT,
    RecipeID INT,
    RatingScore INT CHECK (RatingScore BETWEEN 1 AND 5),
    Comment TEXT,
    Date DATE,
    PRIMARY KEY (UserID, RecipeID),
    FOREIGN KEY (UserID) REFERENCES User(UserID),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID)
);

CREATE TABLE RecipeDietaryTag (
    RecipeID INT,
    TagID INT,
    PRIMARY KEY (RecipeID, TagID),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID),
    FOREIGN KEY (TagID) REFERENCES DietaryTag(TagID)
);
```

Performing Different Actions on the Database that will be made by the App:

1. Retrieve Data:

-- Get all recipes with preparation time under 30 minutes

SELECT * FROM Recipe WHERE PrepTime < 30;

```
1857    -- Get all recipes with preparation time under 30 minutes
1858    SELECT * FROM Recipe WHERE PrepTime < 30;
```

100% 42:1858

Result Grid Filter Rows: Search Edit: Export/Import:

RecipeID	Name	Description	PrepTime	CookTime	Servings	UserID
102	Spaghetti Carbonara	Creamy Italian pasta with eggs and cheese	20	25	4	170
103	Garden Salad	Fresh mix of seasonal vegetables	15	0	4	92
104	Beef Tacos	Seasoned ground beef in crispy tortillas	25	20	6	62
105	Vegetable Stir-Fry	Sautéed veggies with soy sauce	20	15	4	199
106	Chicken Alfredo	Creamy pasta with grilled chicken	25	30	6	83
107	Blueberry Pancakes	Fluffy pancakes with fresh blueberries	15	20	4	184
108	Tomato Soup	Creamy soup with roasted tomatoes	15	30	4	44
109	Garlic Bread	Toasted bread with garlic butter	10	15	6	87
110	Lemon Herb Salmon	Baked salmon with lemon and herbs	15	25	2	14
112	Caprese Salad	Mozzarella, tomatoes, and basil	10	0	2	121
115	Banana Bread	Moist bread with ripe bananas	15	60	8	151
117	Chicken Noodle Soup	Comforting soup with shredded chicken	20	45	6	125
118	Greek Salad	Cucumbers, olives, and feta cheese	15	0	4	133
120	Shrimp Scampi	Garlic butter shrimp over pasta	20	20	4	200
121	Caesar Salad	Romaine lettuce with croutons and dressing	15	0	4	40
123	Grilled Cheese	Crispy bread with melted cheese	5	10	2	69
124	Beef Burger	Juicy patty with lettuce and tomato	20	15	4	148
125	Omelette	Fluffy eggs with cheese and veggies	10	10	2	80
127	French Toast	Golden-brown bread with syrup	10	15	4	41
128	Guacamole	Creamy avocado dip with lime	15	0	4	110
129	Baked Ziti	Pasta with ricotta and marinara	25	45	8	202
130	Cobb Salad	Mixed greens with chicken and bacon	20	0	4	2
131	Pumpkin Soup	Creamy soup with roasted pumpkin	20	30	6	203
136	Chocolate Chip Co…	Soft cookies with melted chocolate	15	12	24	204
139	Hummus	Chickpea dip with tahini	15	0	6	172
140	Lentil Soup	Hearty soup with vegetables	20	45	6	25
142	Pesto Pasta	Basil sauce with pine nuts	15	20	4	173
144	Beef Burrito	Spicy beef and beans in tortilla	25	15	2	145
148	Tuna Salad	Canned tuna with mayo and veggies	15	0	4	153
149	Chicken Wings	Crispy wings with hot sauce	20	45	6	66
150	Mashed Potatoes	Creamy potatoes with butter	20	30	6	159
156	Egg Fried Rice	Stir-fried rice with veggies	20	15	4	209

Recipe 44

-- Get vegetarian recipes
SELECT r.Name, r.Description
FROM Recipe r
JOIN RecipeDietaryTag rdt ON r.RecipeID = rdt.RecipeID
JOIN DietaryTag dt ON rdt.TagID = dt.TagID
WHERE dt.TagName = 'Vegetarian';

```
1860    -- Get vegetarian recipes
1861    SELECT r.Name, r.Description
1862    FROM Recipe r
1863    JOIN RecipeDietaryTag rdt ON r.RecipeID = rdt.RecipeID
1864    JOIN DietaryTag dt ON rdt.TagID = dt.TagID
1865    WHERE dt.TagName = 'Vegetarian';
```

100% 33:1865

Result Grid Filter Rows: Search Export:

Name	Description
Classic Chocolate Cake	Rich and moist chocolate cake
Spaghetti Carbonara	Creamy Italian pasta with eggs and cheese
Garden Salad	Fresh mix of seasonal vegetables
Beef Tacos	Seasoned ground beef in crispy tortillas
Vegetable Stir-Fry	Sautéed veggies with soy sauce
Chicken Alfredo	Creamy pasta with grilled chicken
Blueberry Pancakes	Fluffy pancakes with fresh blueberries
Tomato Soup	Creamy soup with roasted tomatoes
Garlic Bread	Toasted bread with garlic butter
Mushroom Risotto	Creamy Arborio rice with mushrooms
Caprese Salad	Mozzarella, tomatoes, and basil
Vegetable Lasagna	Layered pasta with veggies and cheese
Banana Bread	Moist bread with ripe bananas
Margherita Pizza	Classic pizza with tomatoes and basil
Chicken Noodle Soup	Comforting soup with shredded chicken
Greek Salad	Cucumbers, olives, and feta cheese
Caesar Salad	Romaine lettuce with croutons and dressing
Apple Pie	Flaky crust with cinnamon apples
Grilled Cheese	Crispy bread with melted cheese
Beef Burger	Juicy patty with lettuce and tomato
Omelette	Fluffy eggs with cheese and veggies
Chicken Curry	Spicy curry with coconut milk
French Toast	Golden-brown bread with syrup
Guacamole	Creamy avocado dip with lime
Baked Ziti	Pasta with ricotta and marinara
Cobb Salad	Mixed greens with chicken and bacon
Pumpkin Soup	Creamy soup with roasted pumpkin
Pad Thai	Stir-fried rice noodles with shrimp
Tiramisu	Coffee-flavored Italian dessert
Beef Stroganoff	Tender beef in creamy sauce
Falafel Wrap	Crispy chickpea patties in pita
Chocolate Chip Cookies	Soft cookies with melted chocolate

Result 45

2. Add Data

-- Add a new recipe

INSERT INTO Recipe (RecipeID, Name, Description, PrepTime, CookTime, Servings, UserID)
VALUES (301, 'Pumpkin Soup', 'Creamy autumn soup', 20, 40, 6, 96);

3. Update Data

-- Update recipe servings

UPDATE Recipe SET Servings = 8 WHERE RecipeID = 101;

-- Correct email address

UPDATE User SET Email = 'new_email@example.com' WHERE UserID = 14;

4. Delete Data

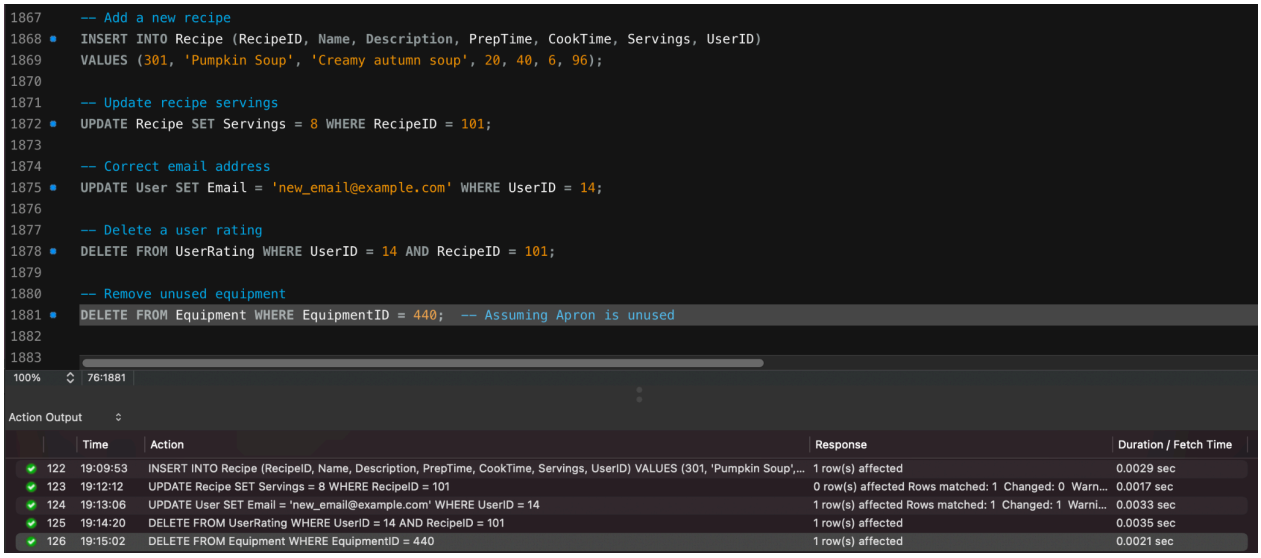
-- Delete a user rating

DELETE FROM UserRating WHERE UserID = 14 AND RecipeID = 101;

-- Remove unused equipment

DELETE FROM Equipment WHERE EquipmentID = 440; -- Assuming Apron is unused

Summary:



Advanced SQL Commands (To be used in the App):

1. CHECK Constraint
UserRating table already includes:

RatingScore INT CHECK (RatingScore BETWEEN 1 AND 5)

```
CREATE TABLE UserRating (
    UserID INT,
    RecipeID INT,
    RatingScore INT CHECK (RatingScore BETWEEN 1 AND 5),
    Comment TEXT,
    Date DATE,
    PRIMARY KEY (UserID, RecipeID),
    FOREIGN KEY (UserID) REFERENCES User(UserID),
    FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID)
);
```

2. Trigger: Enforce Valid Preparation Time

```
DELIMITER //
CREATE TRIGGER ValidatePrepTime
BEFORE INSERT ON Recipe
FOR EACH ROW
BEGIN
   IF NEW.PrepTime < 0 THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Preparation time cannot be negative.';
   END IF;
END //
DELIMITER ;
```

This ensures that the user does not add invalid Preparation time for a recipe. This will ultimately improve the quality of Data that is stored in the Database.

Conclusion:

This project delivers a robust, fully functional recipe management database system that meets all technical and functional specifications outlined. The database was well designed and deployed across a MySQL server, in accordance with relational database best practice and normalisation guidelines to ensure data integrity and performance.

Key Achievements:

Database Setup & Structure: All 14 normalized tables (User, Recipe, Ingredient, and relation tables) were properly set up in MySQL with appropriate primary/foreign keys and constraints. Tables were constructed to minimize redundancy (separate DietaryTag and RecipeDietaryTag tables for many-to-many relationships).

Data Population: Realistic, common data was inserted into each table, e.g., 282 users, 300 recipes, 100 ingredients, and over 1,000 relationship rows (e.g., RecipeIngredient, UserRating). The sample data replicates realistic cooking scenarios (e.g., recipe steps, nutritional facts, equipment mappings).

Non-Advanced SQL Commands: CRUD operations for efficiently used. For example: Retrieving vegetarian recipes with ingredients.

Advanced SQL Features: Used TRIGGER to validate preparation time. Used CHECK Constraint to validate ratings.

The database is running completely and is ready to be incorporated into a recipe management system. Its normalized design allows scalability, while the availability of constraints and triggers allows data integrity. Additional advanced features (e.g., full-text search on recipes) or incorporation of user authentication can be added in the future.

Report Submitted By Team Members:
Sai Satwik Bikumandla - 001655014
Hari Pavan Reddy Bojjam - 001605887
Rishikesh Sirisilla - 001661448
Akshitha Janapana - 001615320