--> Get Dishname, Rating and RecipeURL

CREATE VIEW ShowAvailableRecipes AS

SELECT r.DishName, r.Rating, r.TotalCookingTime AS TotalCookingTime, i.ImageURL

FROM Recipes r

LEFT JOIN Images i ON r.RecipeID = i.RecipeID;

GO

A screenshot of a computer

Description automatically generated

This view combines data from the ‘Recipes’ and ‘Images’ tables. It prints each recipe’s name, rating, total cooking time and image URL. A LEFT JOIN is used to include all the recipes (even those without images).

--> View to show all recipe instructions

--NOTE: this view prints out multiple lines for the results, I’m still yet to figure out how to make it work:

CREATE VIEW RecipeInstructions AS

SELECT R.DishName AS RecipeName,

R.Description,

R.Rating,

I.IngredientName,

I.Quantity AS IngredientQuantity,

I.MeasurementUnit AS IngredientMeasurementUnit,

IM.ImageURL,

I2.StepNumber,

I2.InstructionDescription

FROM Recipes R

INNER JOIN Ingredients I ON R.RecipeID = I.RecipeID

LEFT JOIN (SELECT RecipeID, StepNumber, InstructionDescription

FROM Instructions) I2 ON R.RecipeID = I2.RecipeID

LEFT JOIN (SELECT RecipeID, ImageURL

FROM Images) IM ON R.RecipeID = IM.RecipeID;

GO

This view combines data from the ‘Recipes’, ‘Ingredients’, ‘Instructions’, and ‘Images’ tables to provide a general overview of a recipe. It prints the recipe name, description, rating, ingredient details (name, quantity, unit), image URL, and cooking instructions (step number and description). The view uses INNER JOIN to link recipes with their ingredients and LEFT JOIN to include optional images and instructions.

--View Recipe Instructions

SELECT \*

FROM RecipeInstructions

WHERE RecipeName = 'Fried Rice';

GO

The query above utilizes the ‘RecipeInstructions’ view to print the general details of the recipe ‘Fried Rice’.

--> Insert new User

CREATE PROCEDURE InsertUser(

@Username NVARCHAR(50),

@Password NVARCHAR(50),

@DateOfBirth DATE,

@Email NVARCHAR(100),

@UserRole NVARCHAR(50)

)

AS

BEGIN

INSERT INTO Users (Username, Password, DateOfBirth, Email, UserRole)

VALUES (@Username, @Password, @DateOfBirth, @Email, @UserRole);

END;

GO

EXEC InsertUser 'user4', 'passsostrong', '1994-01-04', 'user4@example.com', 'contributor';

GO

A screenshot of a computer

Description automatically generated

**NOTE:** The insert procedures take the same format as the one used for the ‘InsertUsers’ above so we’ll will just show the queries to save space in the document, and avoid it being too clunky.

The ‘InsertUser’ procedure streamlines the process of inserting new users into the ‘Users’ table; it takes all the necessary data and passes it into the table.

--> Insert new category

CREATE PROCEDURE InsertCategory(

@CategoryName VARCHAR(255)

) AS

BEGIN

INSERT INTO Categories (CategoryName) VALUES (@CategoryName)

END;

GO

EXEC InsertCategory 'Seafood';

GO

Similar to the ‘InsertUser’ procedure, the ‘InsertCategory’ simplifies the process of inserting new data into the Categories table.

--> Insert new Recipes

CREATE PROCEDURE InsertRecipe(

@CategoryID INT,

@UserID INT,

@DishName VARCHAR(100),

@TotalCookingTime VARCHAR(50),

@Description VARCHAR(255),

@Rating VARCHAR(255)

) AS

BEGIN

INSERT INTO Recipes (CategoryID, UserID, DishName, TotalCookingTime, Description, Rating)

VALUES (@CategoryID, @UserID, @DishName, @TotalCookingTime, @Description, @Rating)

END;

GO

EXEC InsertRecipe '4', '7', 'Shrimp Fried Rice', '55 mins',

'a symphony of flavors and textures that tantalizes the taste buds with each bite', '4.9';

GO

The ‘InsertRecipe’ procedure optimises the process of inserting new data into the Recipes table.

--> Insert new Ingredients

CREATE PROCEDURE InsertIngredient(

@RecipeID INT,

@IngredientName VARCHAR(255),

@Quantity VARCHAR(255),

@MeasurementUnit VARCHAR(255)

) AS

BEGIN

INSERT INTO Ingredients (RecipeID, IngredientName, Quantity, MeasurementUnit)

VALUES (@RecipeID, @IngredientName, @Quantity, @MeasurementUnit)

END;

GO

EXEC InsertIngredient '13', 'Rice', '2 cups', 'cups';

EXEC InsertIngredient '13', 'Vegetable Oil', '2 tablespoons', 'tablespoons';

EXEC InsertIngredient '13', 'Onions', '1 large', 'large';

EXEC InsertIngredient '13', 'Carrots', '2 medium', 'medium';

EXEC InsertIngredient '13', 'Green Peas', '1 cup', 'cup';

EXEC InsertIngredient '13', 'Bell Peppers', '2', '';

EXEC InsertIngredient '13', 'Salt', 'to taste', '';

EXEC InsertIngredient '13', 'Seasoning Cubes', '2', '';

GO

--> Insert new OptionalIngredients

CREATE PROCEDURE InsertOptionalIngredient(

@RecipeID INT,

@IngredientName VARCHAR(255),

@Quantity VARCHAR(255)

) AS

BEGIN

INSERT INTO OptionalIngredient (RecipeID, IngredientName, Quantity)

VALUES (@RecipeID, @IngredientName, @Quantity)

END;

GO

EXEC InsertOptionalIngredient @RecipeID = 13, @IngredientName = 'Soy Sauce', @Quantity = '2 tablespoons';

EXEC InsertOptionalIngredient @RecipeID = 13, @IngredientName = 'Sesame Oil', @Quantity = '1 teaspoon';

EXEC InsertOptionalIngredient @RecipeID = 13, @IngredientName = 'Scallions', @Quantity = '2 stalks';

GO

--> Insert new CookingHardware

CREATE PROCEDURE InsertCookingHardware(

@RecipeID INT,

@HardwareName VARCHAR(255)

) AS

BEGIN

INSERT INTO CookingHardware (RecipeID, HardwareName)

VALUES (@RecipeID, @HardwareName)

END;

EXEC InsertCookingHardware @RecipeID = 13, @HardwareName= 'Large pan or wok';

EXEC InsertCookingHardware @RecipeID = 13, @HardwareName= 'Stove or cooker';

EXEC InsertCookingHardware @RecipeID = 13, @HardwareName= 'Measuring cups and spoons';

EXEC InsertCookingHardware @RecipeID = 13, @HardwareName= 'Knife';

EXEC InsertCookingHardware @RecipeID = 13, @HardwareName= 'Cutting board';

GO

The above procedure inserts the provided data into the ‘CookingHardware’ table.

--> Insert new Image

CREATE PROCEDURE InsertImage(

@RecipeID INT,

@ImageURL VARCHAR(255)

) AS

BEGIN

INSERT INTO Images (RecipeID, ImageURL)

VALUES (@RecipeID, @ImageURL)

END;

GO

EXEC InsertImage '13', 'https://www.lecremedelacrumb.com/wp-content/uploads/2019/05/one-pan-spanish-shrimp-rice-1.jpg';

GO

This procedure functions like other insert procedures, it inserts image URLs into the Images table and associates them with the corresponding RecipeID.

--> Delete Recipe data from the recipe database

CREATE PROCEDURE DeleteRecipe @RecipeID INT

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRY

BEGIN TRANSACTION;

DELETE FROM RecipeIngredients WHERE RecipeID = @RecipeID;

DELETE FROM CookingHardware WHERE RecipeID = @RecipeID;

DELETE FROM Images WHERE RecipeID = @RecipeID;

DELETE FROM Instructions WHERE RecipeID = @RecipeID;

DELETE FROM OptionalIngredient WHERE RecipeID = @RecipeID;

DELETE FROM Ingredients WHERE RecipeID = @RecipeID;

DELETE FROM Recipes WHERE RecipeID = @RecipeID;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

-- errors

PRINT ERROR\_MESSAGE();

END CATCH;

END;

GO

EXEC DeleteRecipe '13';

GO

The DeleteRecipe procedure streamlines the process of deleting an entire recipe's data from the database.

--> Insert new entire Recipe

CREATE PROCEDURE InsertEntireRecipe

@RecipeName VARCHAR(255),

@TotalCookingTime VARCHAR(50),

@IngredientsList XML = NULL,

@InstructionsList XML,

@CookingHardwareList XML = NULL,

@ImageURL VARCHAR(255),

@CategoryName VARCHAR(255),

@Description VARCHAR(255),

@Rating VARCHAR(255),

@Username VARCHAR(50)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @RecipeID INT;

--> Insert into Recipes table

INSERT INTO Recipes (DishName, TotalCookingTime, CategoryID, Description, Rating, UserID)

VALUES (@RecipeName, @TotalCookingTime, (SELECT CategoryID FROM Categories WHERE CategoryName = @CategoryName), @Description, @Rating, (SELECT UserID FROM Users WHERE Username = @Username));

SET @RecipeID = SCOPE\_IDENTITY();

* Insert into Ingredients table

IF @IngredientsList IS NOT NULL

BEGIN

INSERT INTO Ingredients (RecipeID, IngredientName, Quantity, MeasurementUnit)

SELECT

@RecipeID,

ParamValues.x.value('(IngredientName)[1]', 'VARCHAR(255)'),

ParamValues.x.value('(Quantity)[1]', 'VARCHAR(255)'),

ParamValues.x.value('(MeasurementUnit)[1]', 'VARCHAR(255)')

FROM @IngredientsList.nodes('/Ingredients/Ingredient') AS ParamValues(x);

END;

-- Insert into Instructions table

INSERT INTO Instructions (RecipeID, StepNumber, InstructionDescription)

SELECT

@RecipeID,

ParamValues.x.value('(StepNumber)[1]', 'INT'),

ParamValues.x.value('(InstructionDescription)[1]', 'VARCHAR(255)')

FROM @InstructionsList.nodes('/Instructions/Instruction') AS ParamValues(x);

-- Insert into CookingHardware table

IF @CookingHardwareList IS NOT NULL

BEGIN

INSERT INTO CookingHardware (RecipeID, HardwareName)

SELECT

@RecipeID,

ParamValues.x.value('(HardwareName)[1]', 'VARCHAR(255)')

FROM @CookingHardwareList.nodes('/CookingHardware/Hardware') AS ParamValues(x);

END;

-- Insert into Images table

INSERT INTO Images (RecipeID, ImageURL)

VALUES (@RecipeID, @ImageURL);

END;

--> Insert into the procedure

EXEC InsertEntireRecipe

@RecipeName = 'Pasta Carbonara',

@TotalCookingTime = '30 mins',

@IngredientsList = '

<Ingredients>

<Ingredient>

<IngredientName>Pasta</IngredientName>

<Quantity>200g</Quantity>

<MeasurementUnit>grams</MeasurementUnit>

</Ingredient>

<Ingredient>

<IngredientName>Bacon</IngredientName>

<Quantity>150g</Quantity>

<MeasurementUnit>grams</MeasurementUnit>

</Ingredient>

<Ingredient>

<IngredientName>Eggs</IngredientName>

<Quantity>2</Quantity>

<MeasurementUnit></MeasurementUnit>

</Ingredient>

<Ingredient>

<IngredientName>Grated Parmesan Cheese</IngredientName>

<Quantity>50g</Quantity>

<MeasurementUnit>grams</MeasurementUnit>

</Ingredient>

<Ingredient>

<IngredientName>Black Pepper</IngredientName>

<Quantity>To taste</Quantity>

<MeasurementUnit></MeasurementUnit>

</Ingredient>

</Ingredients>',

@InstructionsList = '

<Instructions>

<Instruction>

<StepNumber>1</StepNumber>

<InstructionDescription>Cook pasta according to package instructions until al dente.</InstructionDescription>

</Instruction>

<Instruction>

<StepNumber>2</StepNumber>

<InstructionDescription>Cook bacon in a skillet until crispy, then chop into small pieces.</InstructionDescription>

</Instruction>

<Instruction>

<StepNumber>3</StepNumber>

<InstructionDescription>In a bowl, whisk together eggs and grated Parmesan cheese.</InstructionDescription>

</Instruction>

<Instruction>

<StepNumber>4</StepNumber>

<InstructionDescription>Drain cooked pasta and immediately add it to the egg and cheese mixture, tossing quickly to coat the pasta.</InstructionDescription>

</Instruction>

<Instruction>

<StepNumber>5</StepNumber>

<InstructionDescription>Add crispy bacon pieces and black pepper, toss again to combine.</InstructionDescription>

</Instruction>

<Instruction>

<StepNumber>6</StepNumber>

<InstructionDescription>Serve immediately, garnished with extra grated Parmesan cheese and black pepper.</InstructionDescription>

</Instruction>

</Instructions>',

@CookingHardwareList = '

<CookingHardware>

<HardwareName>Pot</HardwareName>

<HardwareName>Skillet</HardwareName>

<HardwareName>Bowl</HardwareName>

<HardwareName>Whisk</HardwareName>

</CookingHardware>',

@ImageURL = 'https://www.example.com/images/pasta\_carbonara.jpg',

@CategoryName = 'Italian Dishes',

@Description = 'Classic Italian pasta dish with bacon, eggs, and Parmesan cheese.',

@Rating = '4.9',

@Username = 'user1';

The InsertEntireRecipe stored procedure inserts a complete recipe into the database, including details about the recipe, ingredients, optional ingredients, instructions, cooking hardware, and images. It checks if the specified category exists, inserts it if not, and then proceeds to insert the recipe details, followed by the related ingredients, optional ingredients, instructions, cooking hardware, and images. The example executions demonstrate inserting different recipes with their associated details into the database.

--> Testing alternative recipe recommendations based on ingredients

-- Search for recipes based on available ingredients

-- Create a table variable to hold available ingredients

DECLARE @AvailableIngredients TABLE (IngredientName VARCHAR(255));

-- Insert available ingredients into the table variable

INSERT INTO @AvailableIngredients (IngredientName) VALUES ('Eggs'), ('Butter');

-- Select recipes that can be made with the available ingredients

SELECT R.RecipeID, R.DishName, R.Description, R.Rating

FROM Recipes R

WHERE NOT EXISTS (

-- Subquery to check if a recipe requires any unavailable ingredients

SELECT RI.IngredientID

FROM RecipeIngredients RI

JOIN Ingredients I ON RI.IngredientID = I.IngredientID

WHERE RI.RecipeID = R.RecipeID

AND I.IngredientName NOT IN (SELECT IngredientName FROM @AvailableIngredients)

);

The query above first creates a temporary table to store available ingredient names and inserts ‘Eggs’ and ‘Butter’ into this table. It then selects recipes from the ‘Recipes’ table where all required ingredients are available. This is done by ensuring that no ingredient required by a recipe is missing from the list of available ingredients. If a recipe requires any ingredient that is not listed in the ‘@AvailableIngredients’ table, it is excluded from the results, filtering the recipes to those that can be made with the provided ingredients.

--> Testing alternative/similar recipe recommendation feature

-- Declare and Populate Available Ingredients Table Variable

DECLARE @AvailableIngredients TABLE (IngredientID INT);

INSERT INTO @AvailableIngredients (IngredientID)

SELECT IngredientID

FROM Ingredients

WHERE IngredientName IN ('Rice', 'Carrots', 'Shrimp'); -- Example of available ingredients

-- Search for Alternative Recipes

SELECT DISTINCT R.RecipeID, R.DishName, R.Description, R.Rating

FROM Recipes R

JOIN RecipeIngredients RI ON R.RecipeID = RI.RecipeID

JOIN Ingredients I ON RI.IngredientID = I.IngredientID

WHERE I.IngredientID IN (SELECT IngredientID FROM @AvailableIngredients)

AND R.RecipeID NOT IN (

SELECT DISTINCT R2.RecipeID

FROM Recipes R2

JOIN RecipeIngredients RI2 ON R2.RecipeID = RI2.RecipeID

JOIN Ingredients I2 ON RI2.IngredientID = I2.IngredientID

WHERE I2.IngredientID NOT IN (SELECT IngredientID FROM @AvailableIngredients)

);

The above query creates a temporary table to store available ingredient IDs and populates it with the ID’s of ‘Rice’, ’Carrots’, and ‘Shrimp’ from the ‘Ingredients’ table. It selects recipes from the ‘Recipes’ table that can be made with the available ingredients. This is done by ensuring that each selected recipe does not require any ingredients outside the list of available ingredients, filtering out any recipes that require unavailable ingredients to be prepared. The result is a list of recipes that can be prepared using only the specified ingredients.