**SQL PROJECT**

**FOODIE CRUSH**

* ***ABSTRACT***

A Delicious Exploration

The recipe portal is a virtual tempting haven designed to inspire educate and delight food enthusiasts of all levels, our platform is dedicated to the art of cooking, offering a vast collection of recipes, tasty resources and a growing community of passionate cooks



* Key features of recipe portal

1. Diverse Recipe Collection:-

Access a collection of recipes spanning cuisine, testy preference and skill levels. From mouth-watering desserts to tangy entrees, this 0portal serves to all tastes.

2. User- Friendly Interface:-

Navigate this natural interface with ease. Discover recipes, save favorites and can share yummy creations effortlessly.

3. Interactive Cooking Guide:-

Step-by-step cooking instructions, along with detailed ingredient lists and cooking tips, empower both beginners and experienced chefs to succeed in the kitchen.

4. Personalization:-

Fit users tasty journey by creating a user's profile, saving recipe and receiving personalized recipe recommendations based on users preferences.

5. Community Engagement:-

Connect with fellow food enthusiasts, exchange idea and engage in lively discussions within our community forums.

6. Mobile Accessibility:-

Enjoy this recipe portal on the go with responsive design, ensuring a seamless experience on both desktop and mobile devices.

Whether an experienced chef looking to expand tasty repertoire or a beginner taking first steps in kitchen, the Recipe Portal is gateway to world yummy possibilities.

* ***Aim of Project***:-

**The main aim of this project is to build a management system which will help in Recipe Portal to function smoothly in order to provide desired results and that too on time and will also maintain records of the Portal's various entities which will help it to function in an organized manner.**

***❖ Objective of Project:-***

* The portal will be able to maintain the records of all the entities like Users, Recipe, Ingredient, Category, and Comment.
* It will create permanent organized relational records about the Users which have helped to obtain particular Recipe.
* It will give details about Recipes and Ingredient being used for particular Recipes.

***ER Diagram***

**USERS**

-ID

Provides

User\_Name

User\_ID (PK)

-id

Join\_Date

Email

City

Used For

**RECIPES**

Recipe\_ID (PK)

**COMMENTSSSS**

Comment\_ID (PK)

Recipe\_Name

Comment

Prep\_Time

Cook\_Time

**CATEGORY**

**INGREDIENT**

Recipe\_ID (FK)

Servings

User\_ID (FK)

Ingredient\_ID (PK)

Category\_ID (FK)

Ingredient\_Name

Works

Description

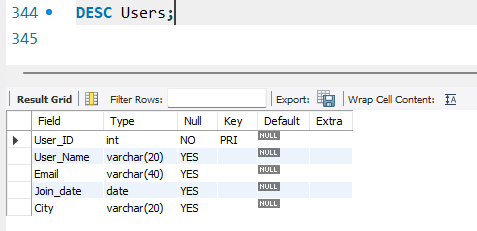
Recipe\_ID (FK)

Category\_ID (PK)

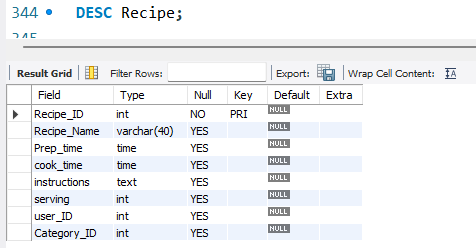
Category\_Name

**STRUCTURE OF TABLES**

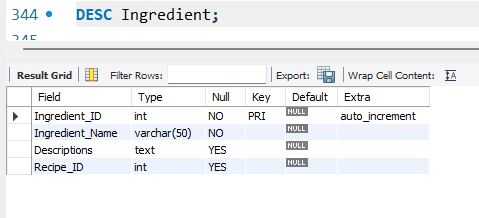
**Users:-**

****

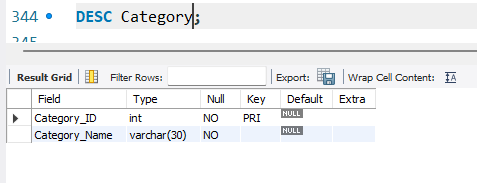
**Recipe:-**

****

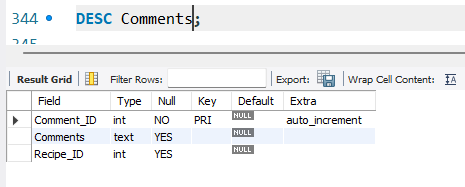
**Ingredients:-**

****

**Category:-**

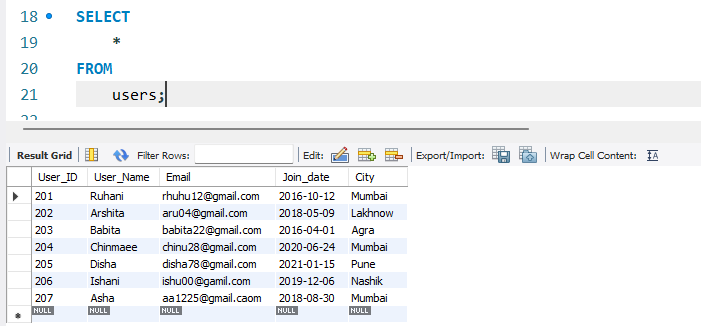
****

**Comments:-**

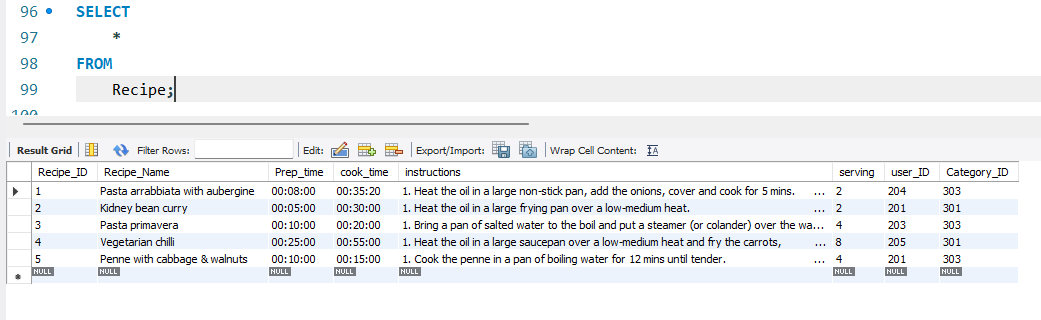
****

**Contents of Tables**

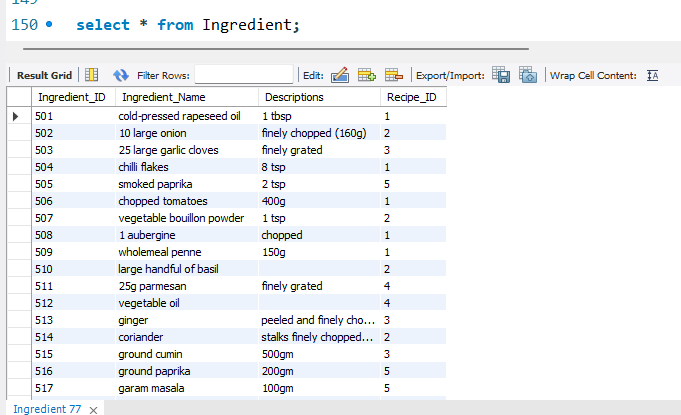
**Users:-**

****

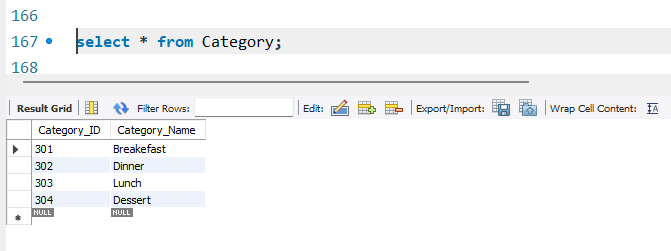
**Recipe:-**

****

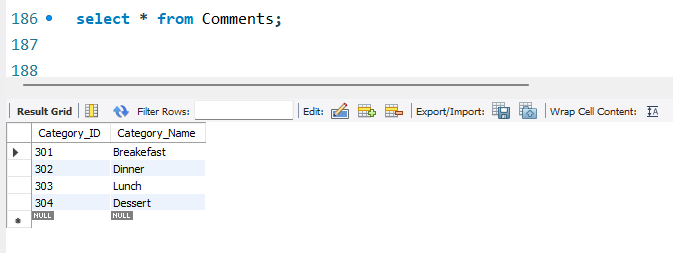
**Ingredients:-**

****

**Category:-**

****

**Comments:-**

****

**DQL QUERY:**

**Like Operator**

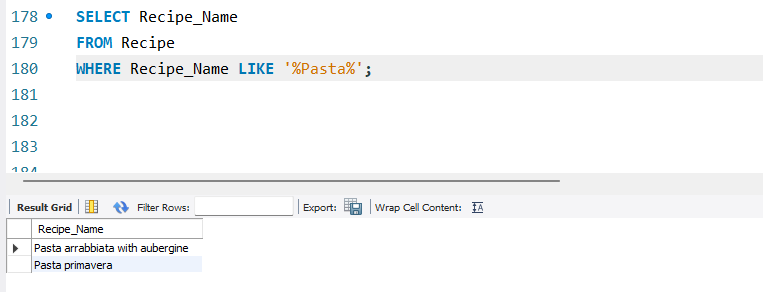
1. Find the recipe with “PASTA” in their title.

Query:

SELECT Recipe\_Name

FROM Recipe

WHERE Recipe\_Name LIKE '%Pasta%';

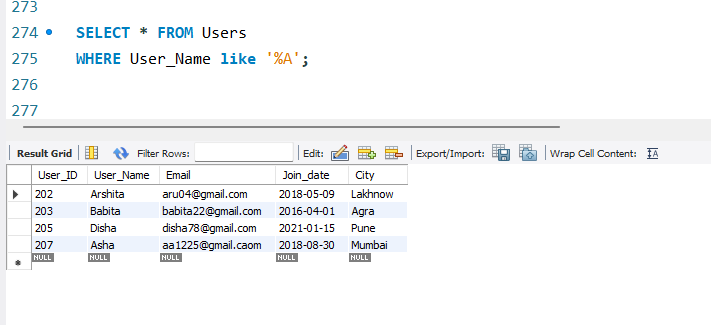


1. Retrieve users whose last letter is A

Query:

SELECT \* FROM Users

WHERE User\_Name like '%A';



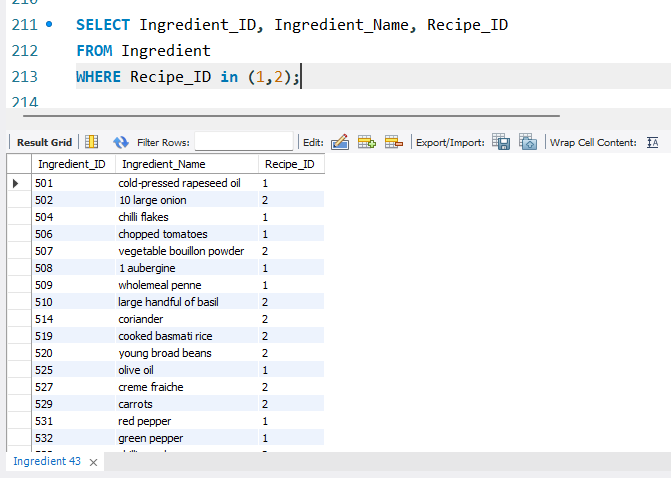
1. Display Ingredient\_ID and Ingredient\_Name whose Recipe\_ID is 1 and 2

Query:

SELECT Ingredient\_ID, Ingredient\_Name, Recipe\_ID

FROM Ingredient

WHERE Recipe\_ID in (1,2);



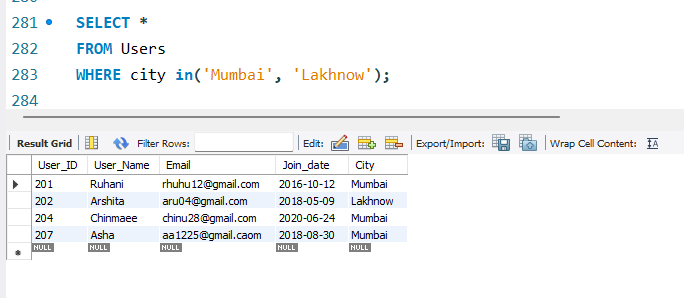
1. Retrieve users whose either from "Mumbai" or "Lakhnow"

Query:

SELECT \*

FROM Users

WHERE city in ('Mumbai', 'Lakhnow');



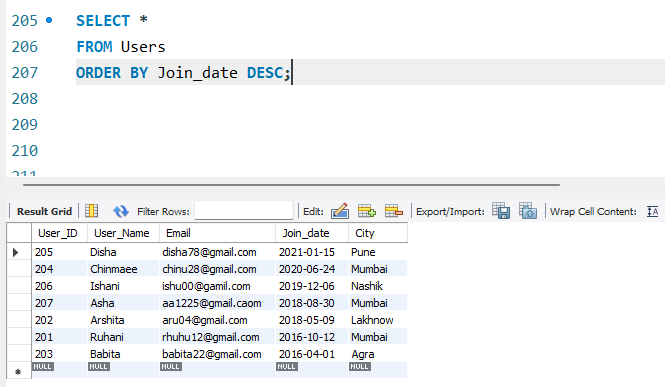
1. List the most recent user join to the portal

Query:

SELECT \*

FROM Users

ORDER BY Join\_date DESC;

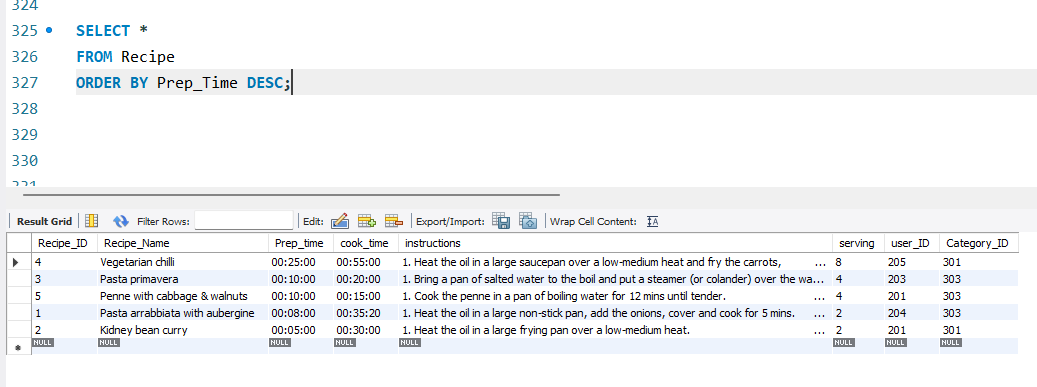


1. Retrieve Recipe according to preparation time in descending order

Query:

SELECT \*FROM Recipe

ORDER BY Prep\_Time DESC;



1. Retrieve the quickest recipe

Query:

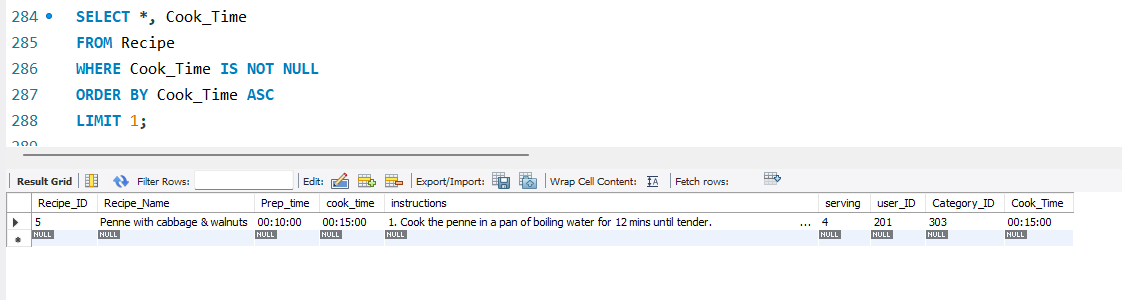
SELECT \*, Cook\_Time

FROM Recipe

WHERE Cook\_Time IS NOT NULL

ORDER BY Cook\_Time ASC

LIMIT 1;



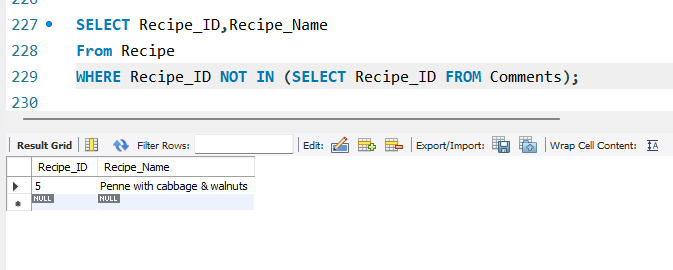
1. Retrieve the Recipe\_ID and Recipe\_Name whose recipe has not been commented

Query:

SELECT Recipe\_ID, Recipe\_Name

From Recipe

WHERE Recipe\_ID NOT IN (SELECT Recipe\_ID FROM Comments);



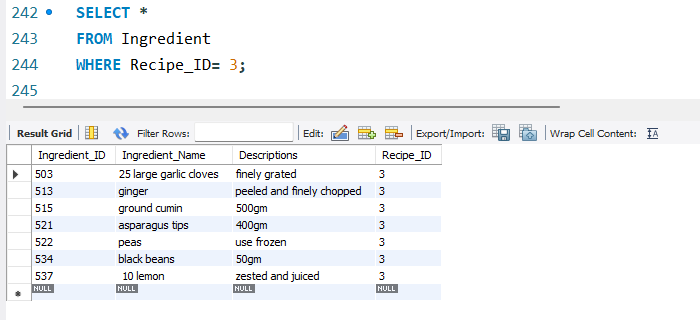
1. Retrieve the ingredients used in recipe\_ID "3"

Query:

SELECT \*

FROM Ingredient

WHERE Recipe\_ID= 3;

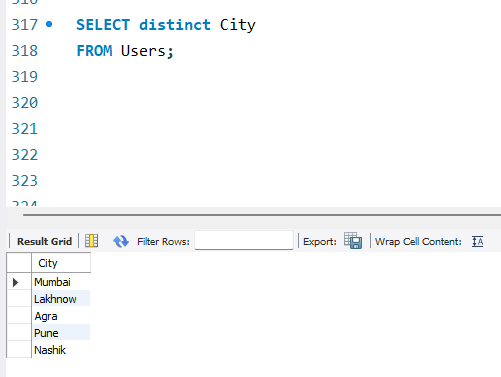


1. Retrieve users from distinct city's

Query:

SELECT distinct City

FROM Users;



**Aggregate Function:**

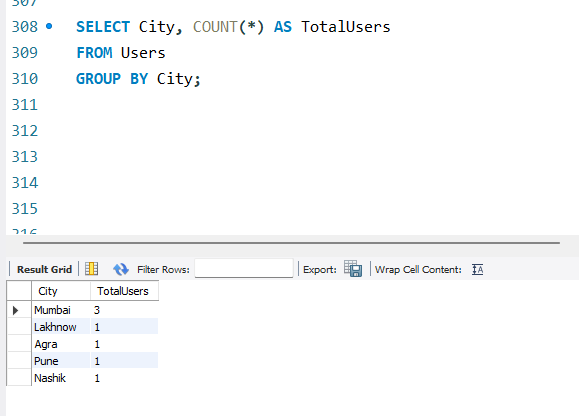
1. Fetch total number of users from various cities

Query:

SELECT City, COUNT (\*) AS TotalUsers

FROM Users

GROUP BY City;



**SUB –QUERY:**

1. Retrieve the third highest cooking time taken by recipe

Query:

SELECT max(Cook\_Time)

FROM Recipe

WHERE Cook\_Time < (select max(Cook\_Time) from Recipe where Cook\_Time < (select max(Cook\_Time) from Recipe));



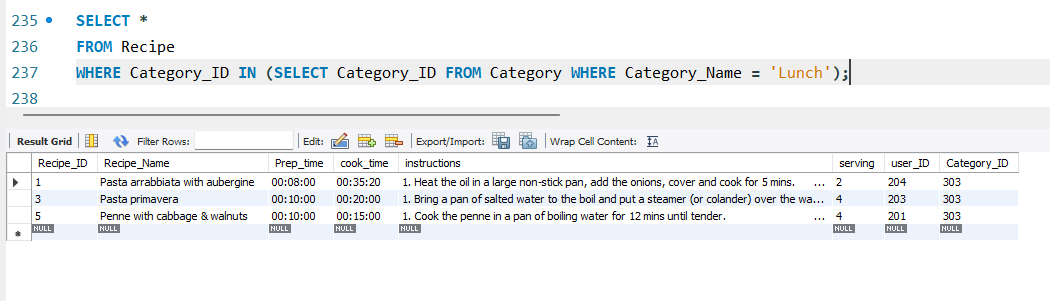
1. Find recipe in a category "Lunch"

Query:

SELECT \*

FROM Recipe

WHERE Category\_ID IN (SELECT Category\_ID FROM Category WHERE Category\_Name = 'Lunch');



**JOIN:**

1. Retrieve the number of comments for each recipe

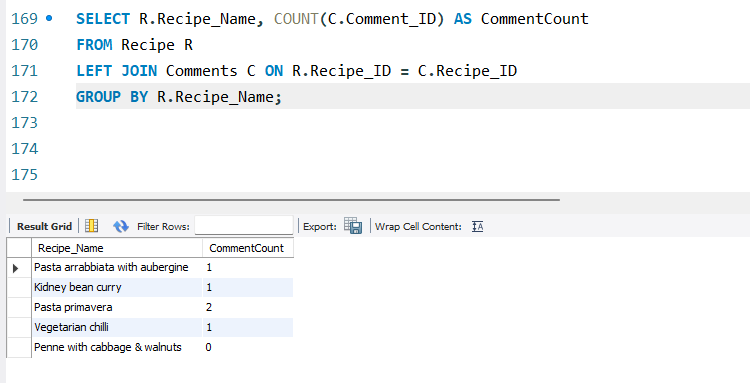
Query:

SELECT R.Recipe\_Name, COUNT(C.Comment\_ID) AS CommentCount

FROM Recipe R

LEFT JOIN Comments C ON R.Recipe\_ID = C.Recipe\_ID

GROUP BY R.Recipe\_Name;



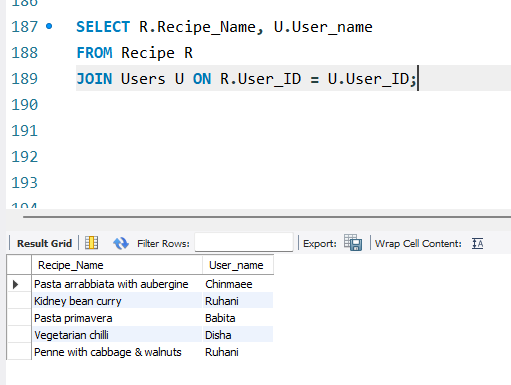
1. Retrieve all the recipe titles and their author’s usernames

Query:

SELECT R.Recipe\_Name, U.User\_name

FROM Recipe R

JOIN Users U ON R.User\_ID = U.User\_ID;



1. Retrieve recipes with a "onion" ingredient in it\

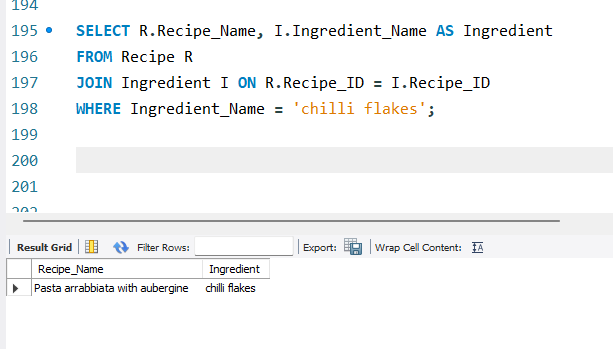
Query:

SELECT R.Recipe\_Name, I.Ingredient\_Name AS Ingredient

FROM Recipe R

JOIN Ingredient I ON R.Recipe\_ID = I.Recipe\_ID

WHERE Ingredient\_Name = 'chili flakes';



1. Retrieve users which didn't posted recipe yet

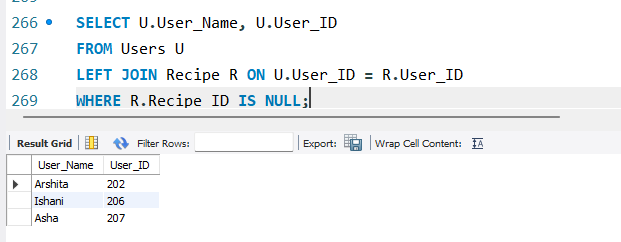
Query:

SELECT U.User\_Name, U.User\_ID

FROM Users U

LEFT JOIN Recipe R ON U.User\_ID = R.User\_ID

WHERE R.Recipe\_ID IS NULL;



1. Retrieve User\_ID, User\_Name and Recipe\_ID, Recipe\_Name

Query:

(Select U.User\_ID, U.User\_Name,

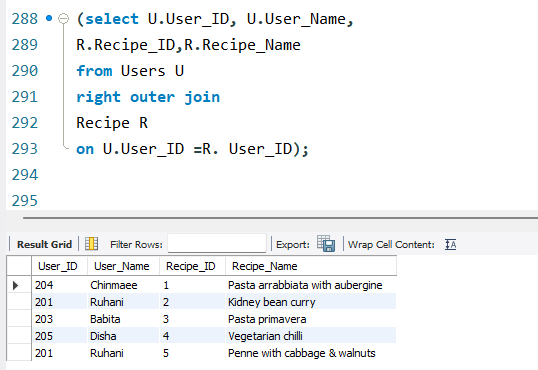
R.Recipe\_ID, R.Recipe\_Name

From Users U

Right outer join

Recipe R

On U.User\_ID =R. User\_ID);



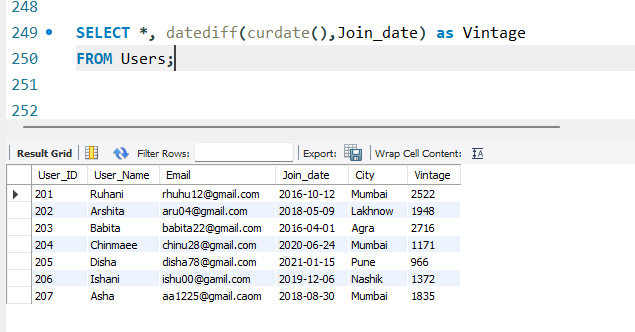
**DATE AND DAY FUNCTION:**

1. Retrieve all users vintage till date

Query:

SELECT \*, datediff (curdate (), Join\_date) as Vintage

FROM Users;

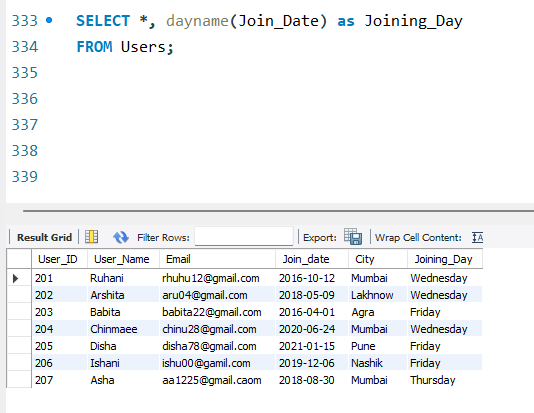


1. Retrieve the Joining day of Users as Joining\_Day

Query:

SELECT \*, dayname (Join\_Date) as Joining\_Day

FROM Users;



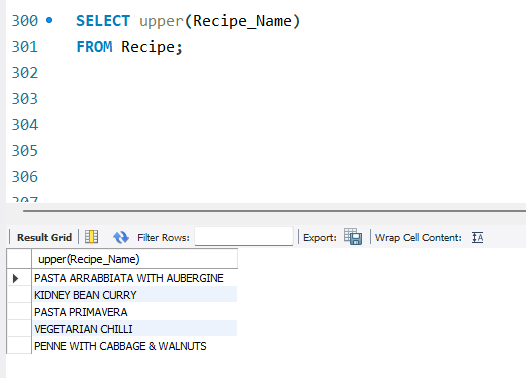
**Built in SQL – String Functions:**

1. Retrieve all Recipe\_name in Capital letter

Query:

SELECT upper (Recipe\_Name)

FROM Recipe;

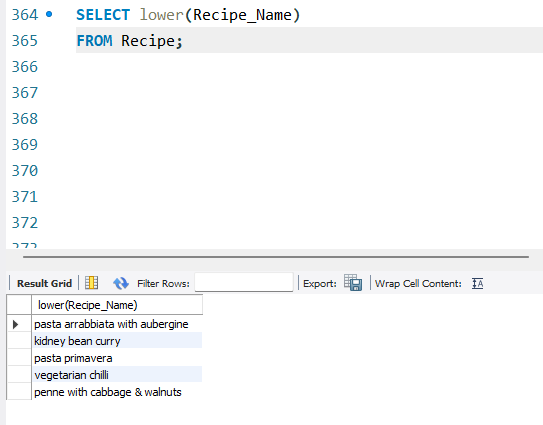
****

1. Retrieve all User\_Name in small letter

Query:

SELECT lower (Recipe\_Name)

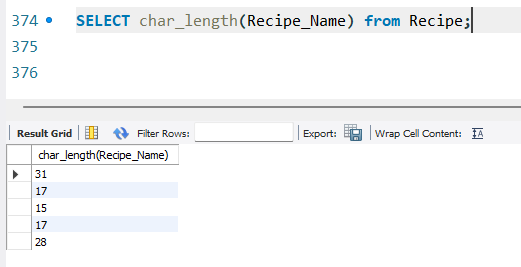
FROM Recipe;



1. Retrieve all character length of Recipe\_Name

Query:

SELECT char\_length (Recipe\_Name) from Recipe;



**DDL QUERY:**

1. Add column "Ratings" in comment table

Query:

ALTER TABLE Comments

Add column Rating int;

