#### Part 1: Conceptual Planning

- Step 1: Brainstorming
  - Users
    - ID (Serial Primary Key)
    - Name
    - Email
    - Password
  - Recipe
    - ID (Serial Primary Key)
    - Recipe Name
    - Public or Private
  - Instructions
    - ID (Serial Primary Key)
    - Recipe ID (Foreign Key references Recipe(ID)
    - Ingredient\_id (Foreign Key references Ingredients(id)
  - Ingredients
    - ID (Serial Primary Key)
      - ex. Tomato ID
      - ex. Carrot ID
    - Recipe ID (Foreign Key references Recipe(ID)
    - Title
  - Reviews
    - ID (Serial Primary Key)
    - Comments
    - Rating
    - user\_ID (Foreign Key references User(id)
  - Occasion
    - ID (Serial Primary Key)
    - Recipe ID (Foreign Key references Recipe(id)
    - Title
  - Grocery Lists
    - ID (Serial Primary Key)
    - Ingredients ID (Foreign Key references Ingredients(id)
  - Images
    - ID (Serial Primary Key)
    - Image\_url
      - Image\_url
    - Recipe ID (foreign key references Recipe(ID)

# Step 2: Table Ideas

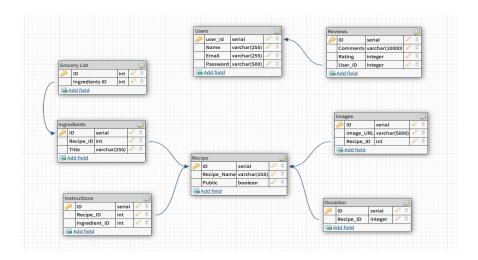
- Users
  - This table will house the information for each user.
- Recipe
  - o This table will have information about each specific recipe.
- Instructions
  - This table will hold the information on how to cook the meal.
- Ingredients
  - Will contain all ingredients needed for each recipe.
- Reviews
  - o This will hold the user comments as well as a rating.
- Occasion
  - Will list different occasions the user will need to prepare for.
- Grocery Lists
  - o This will list the ingredients that are called for in the recipe.
- Images
  - Images related to each recipe.

### **Step 3: Relationships**

- One to One
  - User to Image
  - User to Grocery List

0

- One to Many
  - Recipe to Ingredients
  - Recipe to Reviews
  - User to Reviews
- Many to Many
  - Recipes to Occasions



#### **Columns:**

- Users
  - ID (Serial Primary Key) INTEGER
    - So each recipe can be uniquely identified.
  - Name VARCHAR
    - So each user can input their name. Name will take in varying characters.
  - Email VARCHAR
    - So each user can input their email. Email will take in varying characters.
  - Password VARCHAR
    - So each user can create a password. Passwords can be alphanumeric.
- GroceryList
  - o ID (Serial Primary Key) INTEGER
    - So each recipe can be uniquely identified.
  - o Ingredients ID (Foreign Key) INTEGER
    - So the list can access the ingredients. Its ID will be a number.
- Ingredients
  - o ID (Serial Primary Key) INTEGER
    - So each recipe can be uniquely identified.
  - Recipe ID (Foreign Key) INTEGER
    - So the recipe list can be accessed. Recipe ID is a number.
  - Title TEXT
    - This will be the name of each ingredient and it only needs letters.
- Instructions
  - ID (Serial Primary Key) INTEGER
    - So each recipe can be uniquely identified.
  - Recipe ID (Foreign Key) INTEGER
    - Each set of instructions needs to reference a specific Recipe ID of type integer to know which set of instructions to render.
  - Ingredient ID (Foreign Key) INTEGER
    - Every set of instructions needs to access specific Ingredient IDs
- Recipe
  - ID (Serial Primary Key) INTEGER
    - So each recipe can be uniquely identified.
  - Recipe Name VARCHAR
    - Every recipe needs a name that can contain letters or numbers.
  - Public or Private BOOLEAN
    - Each recipe needs to specify whether or not it can be seen by other users. If public is false, it will be set to private.
- Occasion
  - ID (Serial Primary Key)
    - So each recipe can be uniquely identified.
  - Recipe ID INTEGER (Foreign Key)

- Each occasion needs a recipe ID associated with it and that will be a number.
- Title VARCHAR
  - Each occasion will need a title and that title could have letters or numbers.
- Images
  - ID (Serial Primary Key)
    - So each image can be uniquely identified.
  - Image\_URL VARCHAR (Serial Primary Key)
    - Each image needs a URL that will contain characters/numbers.
  - Recipe\_ID INTEGER (Foreign Key)
    - Each image needs to be linked to a Recipe ID which will be a number.
- Reviews
  - ID (Serial Primary Key)
    - So each recipe can be uniquely identified.
  - Comments VARCHAR
    - The user needs to be able to leave comments in character/number form.
  - Rating INTEGER
    - Each user can leave a number rating.
  - User\_ID INTEGER (Foreign Key)
    - We need to know which user posted each review which will be of type integer.

#### Part 3: Create Tables in SQL

Users

```
CREATE TABLE users(
user_id SERIAL PRIMARY KEY,
name VARCHAR(255),
email VARCHAR(255),
password VARCHAR(500)
);
```

Recipe

```
CREATE TABLE recipe (
recipe_id SERIAL PRIMARY KEY,
recipe_name VARCHAR(255),
recipe_privacy BOOLEAN
);
```

Instructions

```
CREATE TABLE instructions (
 instructions id SERIAL PRIMARY KEY,
 recipe_id INTEGER NOT NULL REFERENCES recipe(recipe_id),
 recipe_title VARCHAR(255)
);

    Ingredients

CREATE TABLE ingredients (
 ingredient_id SERIAL PRIMARY KEY,
 recipe id INTEGER NOT NULL REFERENCES recipe(recipe id),
title VARCHAR(255)
);

    Reviews

CREATE TABLE reviews (
 review id SERIAL PRIMARY KEY,
 comments VARCHAR(10000),
 rating INTEGER,
 user_id INTEGER NOT NULL REFERENCES users(user_id)
);

    Occasion

CREATE TABLE occasion (
 occasion_id SERIAL PRIMARY KEY,
 recipe_id INTEGER NOT NULL REFERENCES recipe(recipe_id),
title VARCHAR(255)
);

    Grocery Lists

CREATE TABLE groceryList (
grocery list id SERIAL PRIMARY KEY,
 ingredients_id INTEGER NOT NULL REFERENCES ingredients(ingredient_id)
);
```

Images

```
CREATE TABLE images (
image_id SERIAL PRIMARY KEY,
image_url VARCHAR (500),
recipe_id INTEGER NOT NULL REFERENCES recipe(recipe_id)
);
```

## Part 4: Inserting in to tables

```
Users:
```

```
INSERT INTO users (name, email, password)
VALUES ('Jane', '123@gmail.com', 123456);
```

```
INSERT INTO users (name, email, password)
VALUES ('John', '456@gmail.com', 105978654568);
```

### Recipes:

```
INSERT INTO recipe (recipe_name, recipe_privacy)
VALUES ('Chicken Noodle Soup', True);
```

#### Reviews:

```
INSERT INTO reviews (comments, rating, user_id) VALUES ('It was bad', 3, 2);
```

```
INSERT INTO reviews (comments, rating, user_id) VALUES ('It was good', 6, 1);
```