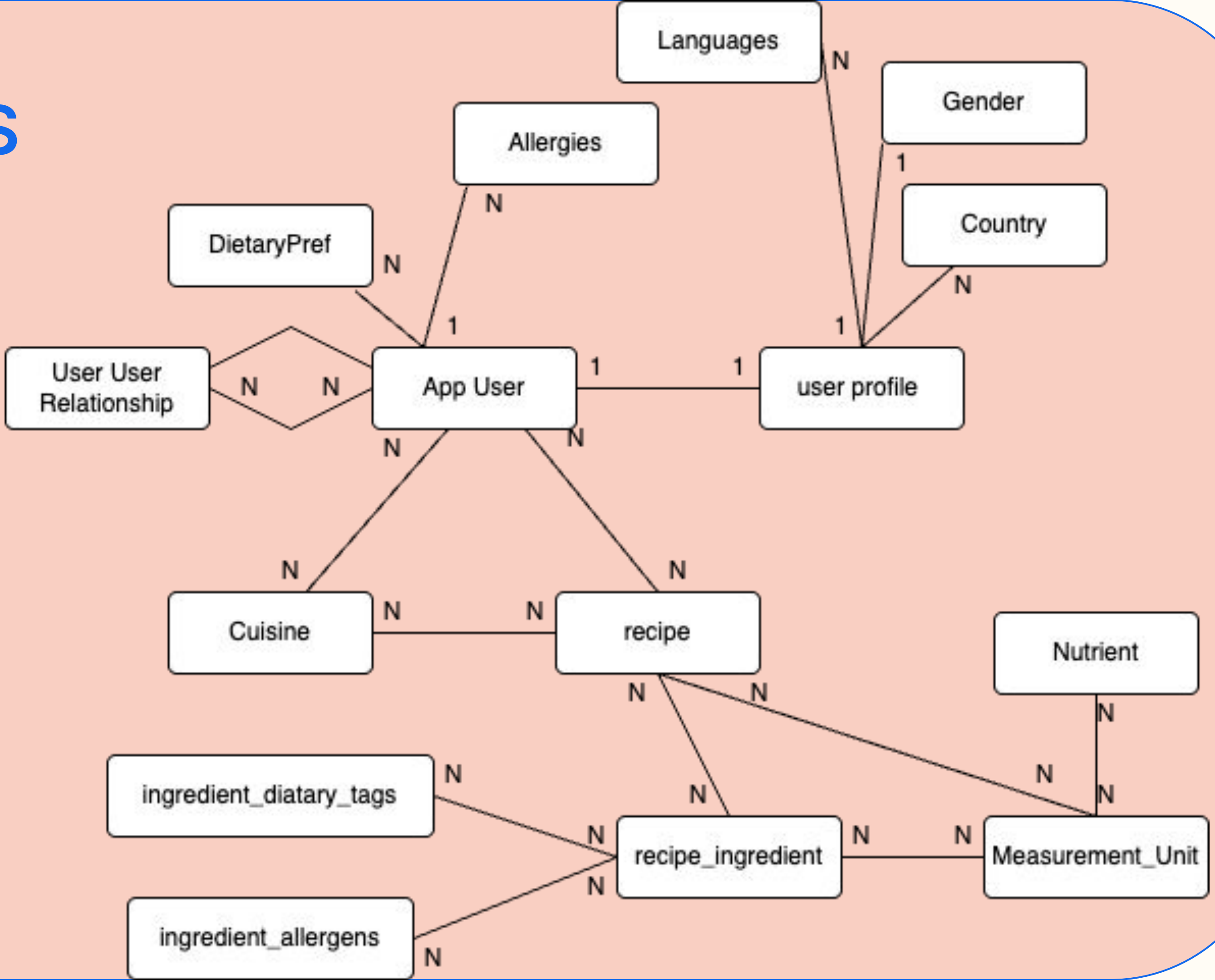
# DATA TEAM

# POSTGRES DATABASE

Zoha Hassan, Fiona Nicdao, Yvette Diaz,
Ololade Akinsanola, Aidan Klug,
Md Sharuf Hossain

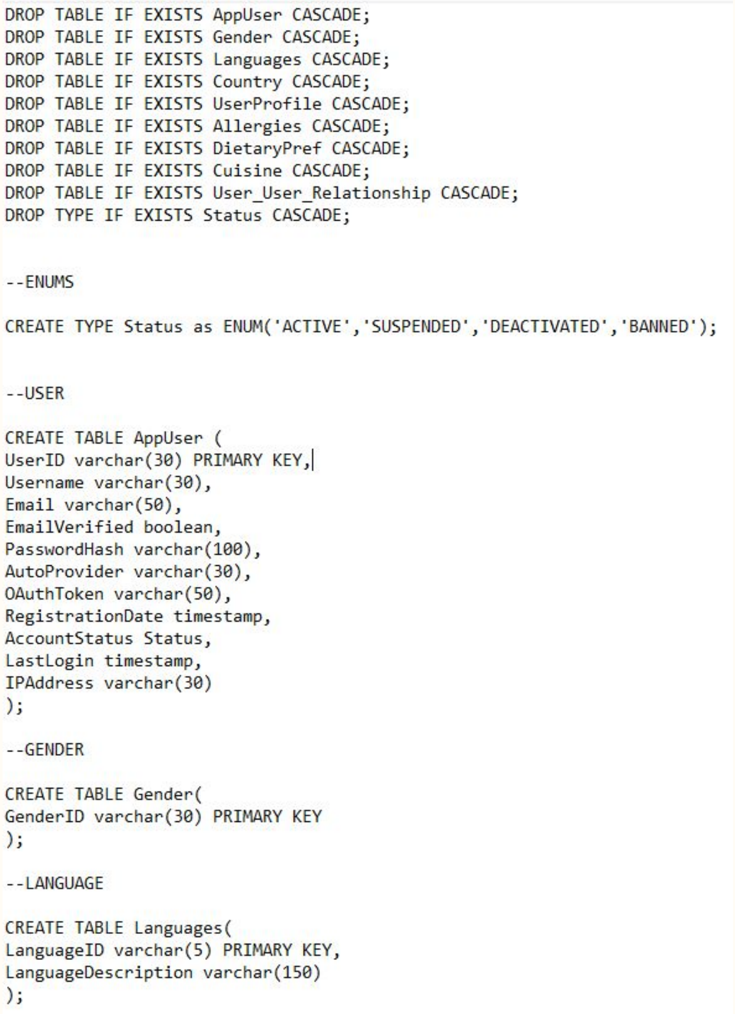
# TABLES


Languages
N
Gender
1
Allergies
N
DietaryPref
N
Country
N
1
1
User User Relationship
N
N
App User
1
1
user profile
N
N
N
N
Cuisine
N
N
recipe
Nutrient
N
N
N
ingredient_diatary_tags
N
N
N
N
recipe_ingredient
N
N
Measurement_Unit
N
ingredient_allergens
N

# TABLE CREATION

Files for the Scripts for creating tables and insert the data

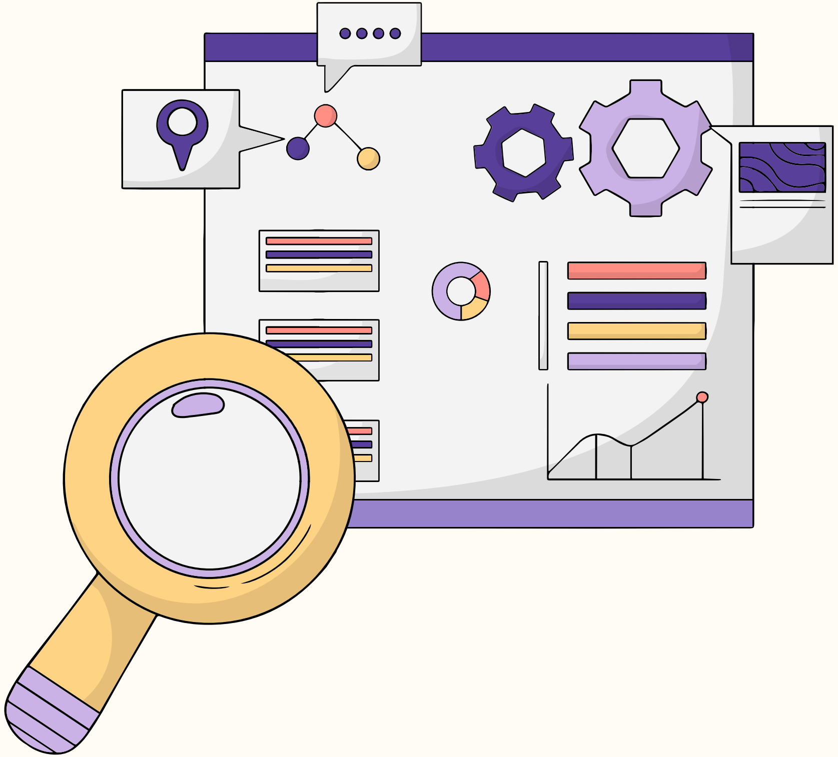
- Combined.txt
  - script to create the tables
- RANDOM_DATA_INSERTION.txt
  - script to insert data
- CLEAR_ALL_TABLES.txt
  - script to delete all tables

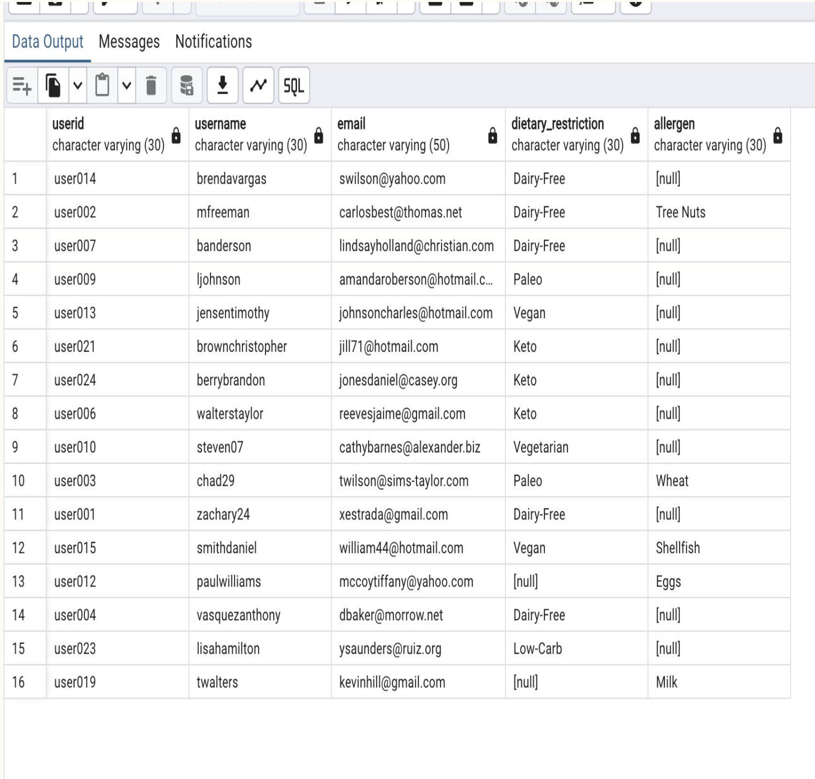


```
DROP TABLE IF EXISTS AppUser CASCADE;
DROP TABLE IF EXISTS Gender CASCADE;
DROP TABLE IF EXISTS Languages CASCADE;
DROP TABLE IF EXISTS Country CASCADE;
DROP TABLE IF EXISTS UserProfile CASCADE;
DROP TABLE IF EXISTS Allergies CASCADE;
DROP TABLE IF EXISTS DietaryPref CASCADE;
DROP TABLE IF EXISTS Cuisine CASCADE;
DROP TABLE IF EXISTS User_User_Relationship CASCADE;
DROP TYPE IF EXISTS Status CASCADE;


--ENUMS

CREATE TYPE Status as ENUM('ACTIVE','SUSPENDED','DEACTIVATED','BANNED');


--USER

CREATE TABLE AppUser (
UserID varchar(30) PRIMARY KEY,
Username varchar(30),
Email varchar(50),
EmailVerified boolean,
PasswordHash varchar(100),
AutoProvider varchar(30),
OAuthToken varchar(50),
RegistrationDate timestamp,
AccountStatus Status,
LastLogin timestamp,
IPAddress varchar(30)
);

--GENDER

CREATE TABLE Gender(
GenderID varchar(30) PRIMARY KEY
);

--LANGUAGE

CREATE TABLE Languages(
LanguageID varchar(5) PRIMARY KEY,
LanguageDescription varchar(150)
);
```

# Potential Queries

Here are some potential queries that can be used for the Machine Learning team or the Development Team to get the data they need to do their work.





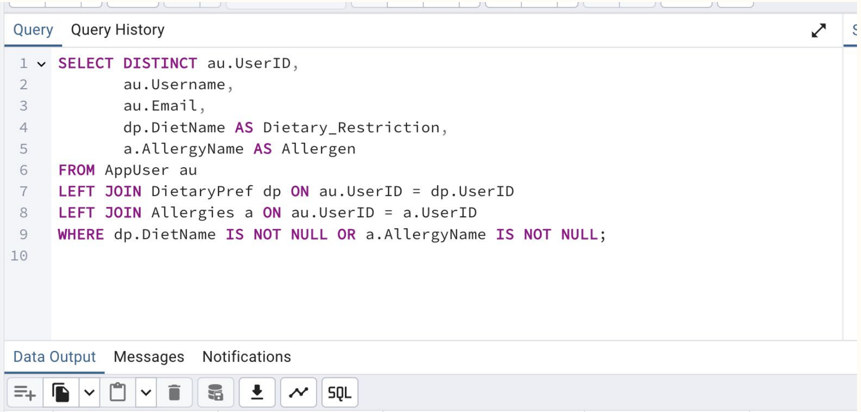
| | userid<br>character varying (30) | username<br>character varying (30) | email<br>character varying (50) | dietary_restriction<br>character varying (30) | allergen<br>character varying (30) |
|---|---|---|---|---|---|
| 1 | user014 | brendavargas | swilson@yahoo.com | Dairy-Free | [null] |
| 2 | user002 | mfreeman | carlosbest@thomas.net | Dairy-Free | Tree Nuts |
| 3 | user007 | banderson | lindsayholland@christian.com | Dairy-Free | [null] |
| 4 | user009 | ljohnson | amandaroberson@hotmail.c... | Paleo | [null] |
| 5 | user013 | jensentimothy | johnsoncharles@hotmail.com | Vegan | [null] |
| 6 | user021 | brownchristopher | jill71@hotmail.com | Keto | [null] |
| 7 | user024 | berrybrandon | jonesdaniel@casey.org | Keto | [null] |
| 8 | user006 | walterstaylor | reevesjaime@gmail.com | Keto | [null] |
| 9 | user010 | steven07 | cathybarnes@alexander.biz | Vegetarian | [null] |
| 10 | user003 | chad29 | twilson@sims-taylor.com | Paleo | Wheat |
| 11 | user001 | zachary24 | xestrada@gmail.com | Dairy-Free | [null] |
| 12 | user015 | smithdaniel | william44@hotmail.com | Vegan | Shellfish |
| 13 | user012 | paulwilliams | mccoytiffany@yahoo.com | [null] | Eggs |
| 14 | user004 | vasquezanthony | dbaker@morrow.net | Dairy-Free | [null] |
| 15 | user023 | lisahamilton | ysaunders@ruiz.org | Low-Carb | [null] |
| 16 | user019 | twalters | kevinhill@gmail.com | [null] | Milk |

# QUERY #1

This query gives a list of users who have provided either dietary preferences, allergy information, or both. It shows contact information along with the relevant dietary or allergen data

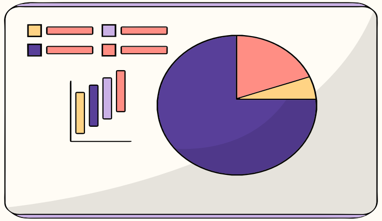
Tables

- App Users, DietaryPref, Allergies
- Where Diet name or AllergyName is not null

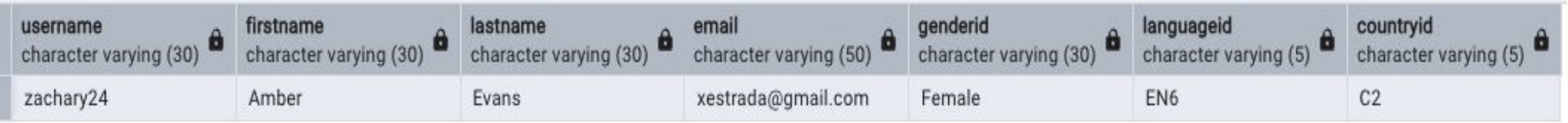


```
SELECT DISTINCT au.UserID,
        au.Username,
        au.Email,
        dp.DietName AS Dietary_Restriction,
        a.AllergyName AS Allergen
FROM AppUser au
LEFT JOIN DietaryPref dp ON au.UserID = dp.UserID
LEFT JOIN Allergies a ON au.UserID = a.UserID
WHERE dp.DietName IS NOT NULL OR a.AllergyName IS NOT NULL;
```

# QUERY

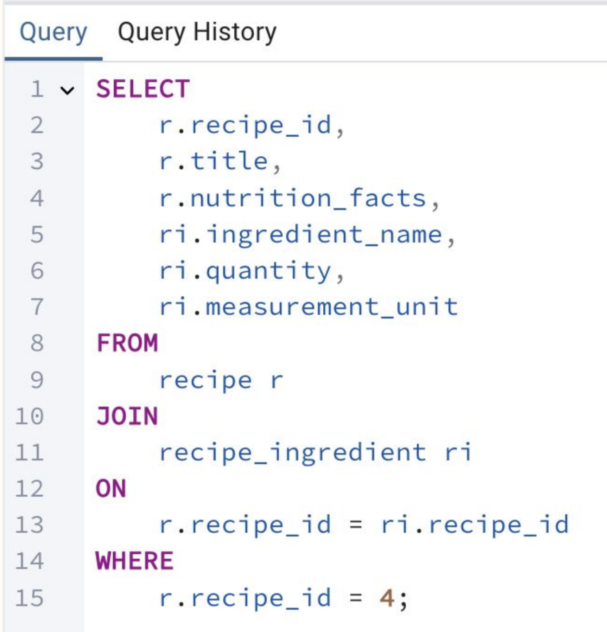


For the Development team, after the user logins in the development team can use this query to upload information about the user on their profile page or dashboard page.
Information:  username, first name, last name, email , gender, language, and country



| username<br>character varying (30) 🔒 | firstname<br>character varying (30) 🔒 | lastname<br>character varying (30) 🔒 | email<br>character varying (50) 🔒 | genderid<br>character varying (30) 🔒 | languageid<br>character varying (5) 🔒 | countryid<br>character varying (5) 🔒 |
|---|---|---|---|---|---|---|
| zachary24 | Amber | Evans | xestrada@gmail.com | Female | EN6 | C2 |

```
SELECT a.username, cp.firstname, cp.lastname, a.email, g.genderid, lan.languageid,
          c.countryid
FROM AppUser a, Gender g, Languages lan, Country c, UserProfile cp
WHERE a.userid = 'user001' and a.userid = cp.userid and cp.gender = g.genderid
           and cp.languages = lan.languageid and cp.country = c.countryid;
```

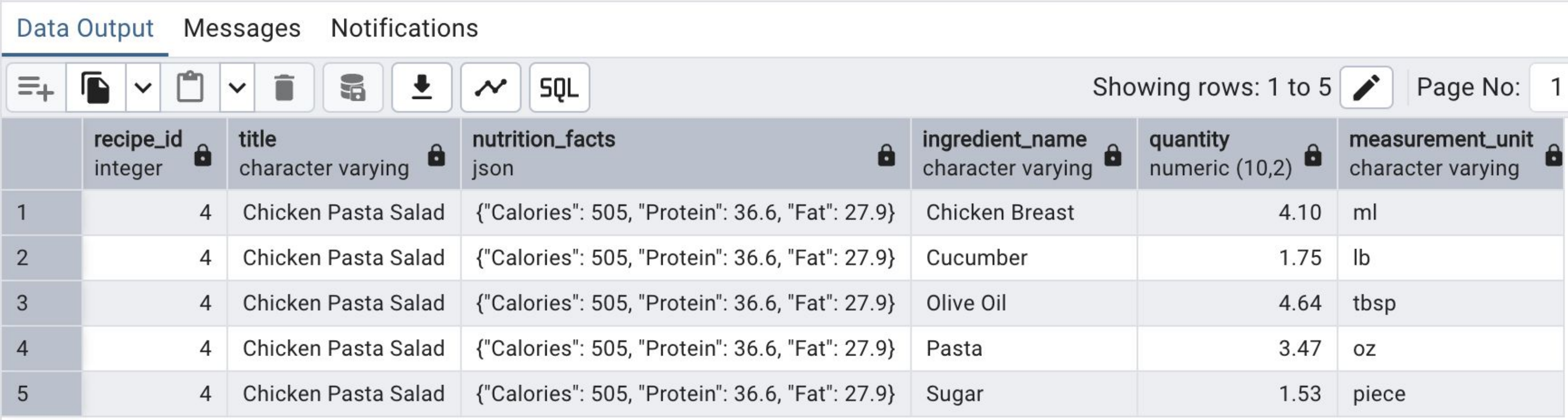


```sql
SELECT
    r.recipe_id,
    r.title,
    r.nutrition_facts,
    ri.ingredient_name,
    ri.quantity,
    ri.measurement_unit
FROM
    recipe r
JOIN
    recipe_ingredient ri
ON
    r.recipe_id = ri.recipe_id
WHERE
    r.recipe_id = 4;
```

# QUERY

Query to access a recipe using the recipe_id, title, ingredients, nutrition facts, and quantity of each ingredient

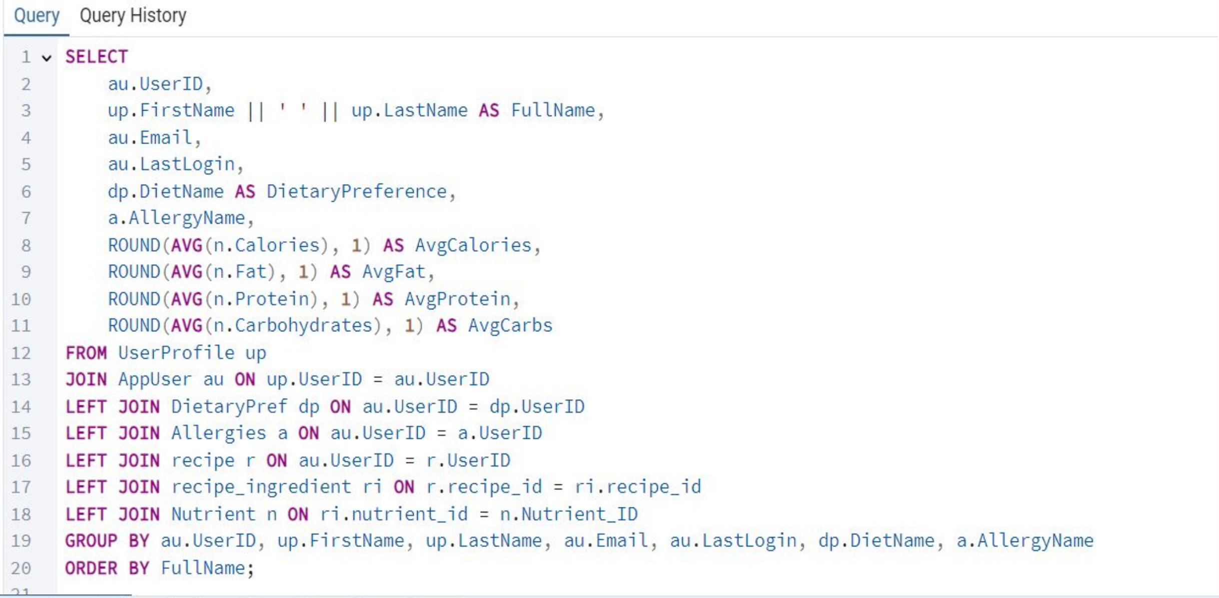
- Two tables: recipe and recipe_ingredient



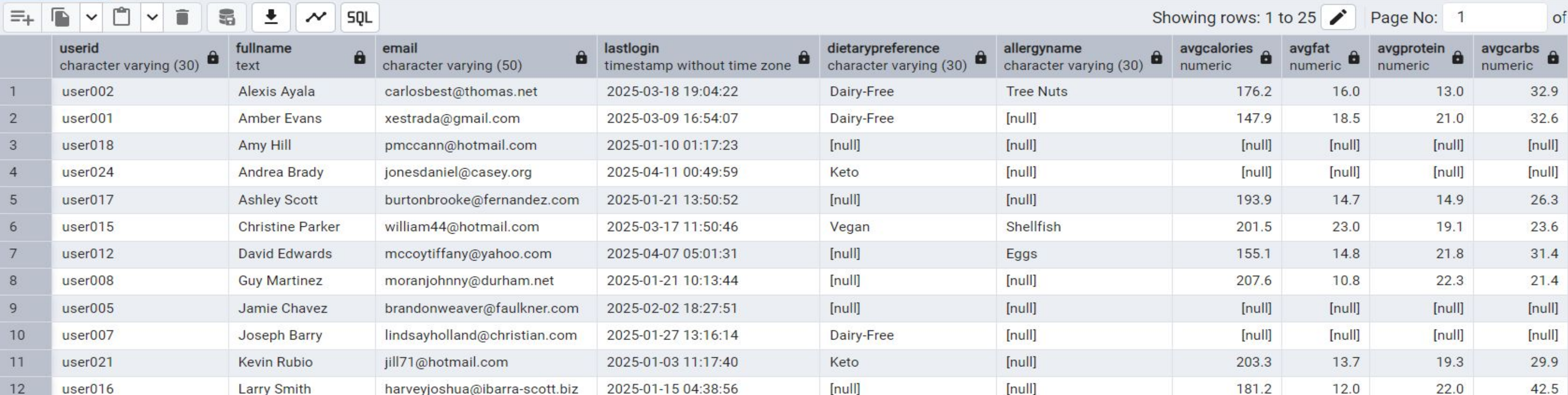
| | recipe_id<br>integer | title<br>character varying | nutrition_facts<br>json | ingredient_name<br>character varying | quantity<br>numeric (10,2) | measurement_unit<br>character varying |
|---|---|---|---|---|---|---|
| 1 | 4 | Chicken Pasta Salad | {"Calories": 505, "Protein": 36.6, "Fat": 27.9} | Chicken Breast | 4.10 | ml |
| 2 | 4 | Chicken Pasta Salad | {"Calories": 505, "Protein": 36.6, "Fat": 27.9} | Cucumber | 1.75 | lb |
| 3 | 4 | Chicken Pasta Salad | {"Calories": 505, "Protein": 36.6, "Fat": 27.9} | Olive Oil | 4.64 | tbsp |
| 4 | 4 | Chicken Pasta Salad | {"Calories": 505, "Protein": 36.6, "Fat": 27.9} | Pasta | 3.47 | oz |
| 5 | 4 | Chicken Pasta Salad | {"Calories": 505, "Protein": 36.6, "Fat": 27.9} | Sugar | 1.53 | piece |

# QUERY

**For each user, what is their User ID, full name, email address, last login date, dietary preference, and any known allergies — along with the average calories, fat, protein, and carbohydrates across all the recipes they have created?**



```sql
SELECT
    au.UserID,
    up.FirstName || ' ' || up.LastName AS FullName,
    au.Email,
    au.LastLogin,
    dp.DietName AS DietaryPreference,
    a.AllergyName,
    ROUND(AVG(n.Calories), 1) AS AvgCalories,
    ROUND(AVG(n.Fat), 1) AS AvgFat,
    ROUND(AVG(n.Protein), 1) AS AvgProtein,
    ROUND(AVG(n.Carbohydrates), 1) AS AvgCarbs
FROM UserProfile up
JOIN AppUser au ON up.UserID = au.UserID
LEFT JOIN DietaryPref dp ON au.UserID = dp.UserID
LEFT JOIN Allergies a ON au.UserID = a.UserID
LEFT JOIN recipe r ON au.UserID = r.UserID
LEFT JOIN recipe_ingredient ri ON r.recipe_id = ri.recipe_id
LEFT JOIN Nutrient n ON ri.nutrient_id = n.Nutrient_ID
GROUP BY au.UserID, up.FirstName, up.LastName, au.Email, au.LastLogin, dp.DietName, a.AllergyName
ORDER BY FullName;
```

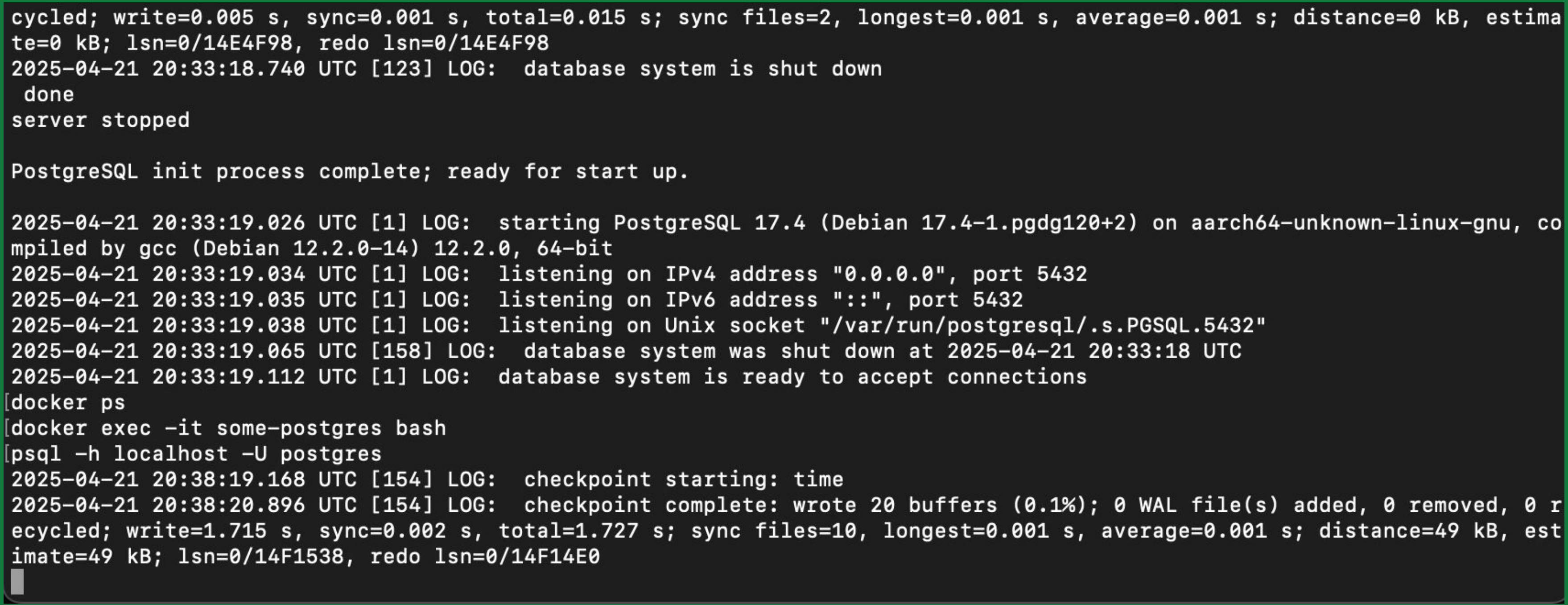


| | userid | fullname | email | lastlogin | dietarypreference | allergyname | avgcalories | avgfat | avgprotein | avgcarbs |
|---|---|---|---|---|---|---|---|---|---|---|
| | character varying (30) | text | character varying (50) | timestamp without time zone | character varying (30) | character varying (30) | numeric | numeric | numeric | numeric |
| 1 | user002 | Alexis Ayala | carlosbest@thomas.net | 2025-03-18 19:04:22 | Dairy-Free | Tree Nuts | 176.2 | 16.0 | 13.0 | 32.9 |
| 2 | user001 | Amber Evans | xestrada@gmail.com | 2025-03-09 16:54:07 | Dairy-Free | [null] | 147.9 | 18.5 | 21.0 | 32.6 |
| 3 | user018 | Amy Hill | pmccann@hotmail.com | 2025-01-10 01:17:23 | [null] | [null] | [null] | [null] | [null] | [null] |
| 4 | user024 | Andrea Brady | jonesdaniel@casey.org | 2025-04-11 00:49:59 | Keto | [null] | [null] | [null] | [null] | [null] |
| 5 | user017 | Ashley Scott | burtonbrooke@fernandez.com | 2025-01-21 13:50:52 | [null] | [null] | 193.9 | 14.7 | 14.9 | 26.3 |
| 6 | user015 | Christine Parker | william44@hotmail.com | 2025-03-17 11:50:46 | Vegan | Shellfish | 201.5 | 23.0 | 19.1 | 23.6 |
| 7 | user012 | David Edwards | mccoytiffany@yahoo.com | 2025-04-07 05:01:31 | [null] | Eggs | 155.1 | 14.8 | 21.8 | 31.4 |
| 8 | user008 | Guy Martinez | moranjohnny@durham.net | 2025-01-21 10:13:44 | [null] | [null] | 207.6 | 10.8 | 22.3 | 21.4 |
| 9 | user005 | Jamie Chavez | brandonweaver@faulkner.com | 2025-02-02 18:27:51 | [null] | [null] | [null] | [null] | [null] | [null] |
| 10 | user007 | Joseph Barry | lindsayholland@christian.com | 2025-01-27 13:16:14 | Dairy-Free | [null] | [null] | [null] | [null] | [null] |
| 11 | user021 | Kevin Rubio | jill71@hotmail.com | 2025-01-03 11:17:40 | Keto | [null] | 203.3 | 13.7 | 19.3 | 29.9 |
| 12 | user016 | Larry Smith | harveyjoshua@ibarra-scott.biz | 2025-01-15 04:38:56 | [null] | [null] | 181.2 | 12.0 | 22.0 | 42.5 |

# DEMO

LINK TO VIDEO DEMO

We are unable to add the image into the server, when running the final command "`docker run some-postgres`" it never finished running for the last 15 mins, this was as screenshot of our last progress



```
cycled; write=0.005 s, sync=0.001 s, total=0.015 s; sync files=2, longest=0.001 s, average=0.001 s; distance=0 kB, estima
te=0 kB; lsn=0/14E4F98, redo lsn=0/14E4F98
2025-04-21 20:33:18.740 UTC [123] LOG:  database system is shut down
 done
server stopped

PostgreSQL init process complete; ready for start up.

2025-04-21 20:33:19.026 UTC [1] LOG:  starting PostgreSQL 17.4 (Debian 17.4-1.pgdg120+2) on aarch64-unknown-linux-gnu, co
mpiled by gcc (Debian 12.2.0-14) 12.2.0, 64-bit
2025-04-21 20:33:19.034 UTC [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
2025-04-21 20:33:19.035 UTC [1] LOG:  listening on IPv6 address "::", port 5432
2025-04-21 20:33:19.038 UTC [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
2025-04-21 20:33:19.065 UTC [158] LOG:  database system was shut down at 2025-04-21 20:33:18 UTC
2025-04-21 20:33:19.112 UTC [1] LOG:  database system is ready to accept connections
[docker ps
[docker exec -it some-postgres bash
[psql -h localhost -U postgres
2025-04-21 20:38:19.168 UTC [154] LOG:  checkpoint starting: time
2025-04-21 20:38:20.896 UTC [154] LOG:  checkpoint complete: wrote 20 buffers (0.1%); 0 WAL file(s) added, 0 removed, 0 r
ecycled; write=1.715 s, sync=0.002 s, total=1.727 s; sync files=10, longest=0.001 s, average=0.001 s; distance=49 kB, est
imate=49 kB; lsn=0/14F1538, redo lsn=0/14F14E0
```

# Thank you!

Github Link