

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ
ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ



NATIONAL TECHNICAL UNIVERSITY OF ATHENS

SCHOOL OF APPLIED MATHEMATICS AND PHYSICAL
SCIENCE

Βάσεις Δεδομένων

Project 27

Αντωνία Γιαννουλέα
ge19028
antoniagian5@gmail.com
ge19028@mail.ntua.gr
10^ο εξάμηνο

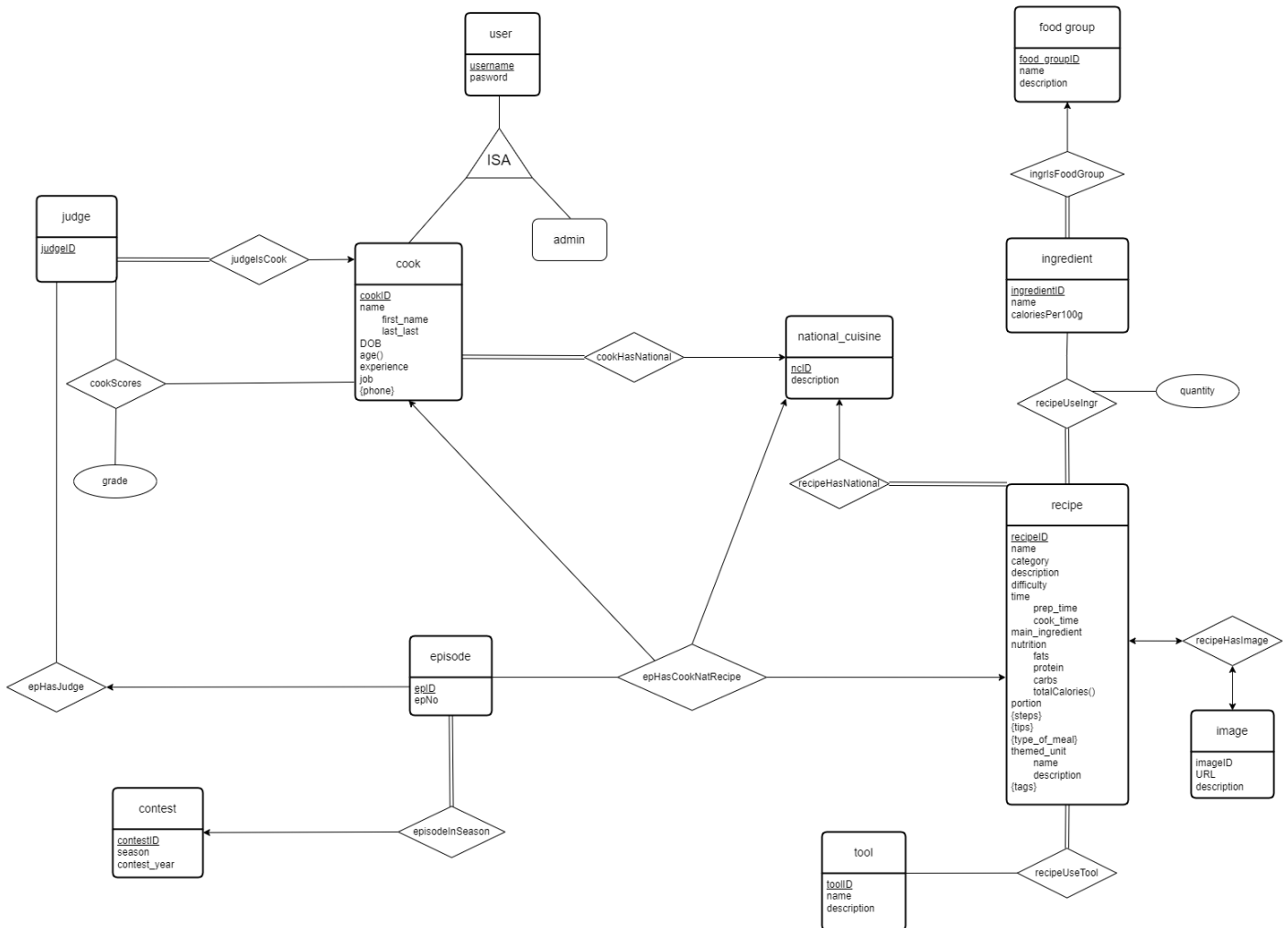
Καμενίδου Μαρίνα Ανθούλα
ge19057
marianthikamenidoy@gmail.com
ge19057@mail.ntua.gr
10^ο εξάμηνο

ΠΕΡΙΕΧΟΜΕΝΑ

1. Σχεδίαση της Βάσης Δεδομένων	
1.1. ER (Entity Relationship Diagram)	
1.2. Σχεσιακό Διάγραμμα (Relational Diagram)	
1.3. Υποθέσεις και Παραδοχές	
2. DDL και DML Scripts	
2.1. DDL Script	
2.2. DML Script	
2.3. Views	
2.4. Triggers	
2.5. Queries	
3. Οδηγίες εγκατάσταση της εφαρμογής	

1. Διαγράμματα της Βάσης Δεδομένων

1.1. ER (Entity Relationship Diagram)

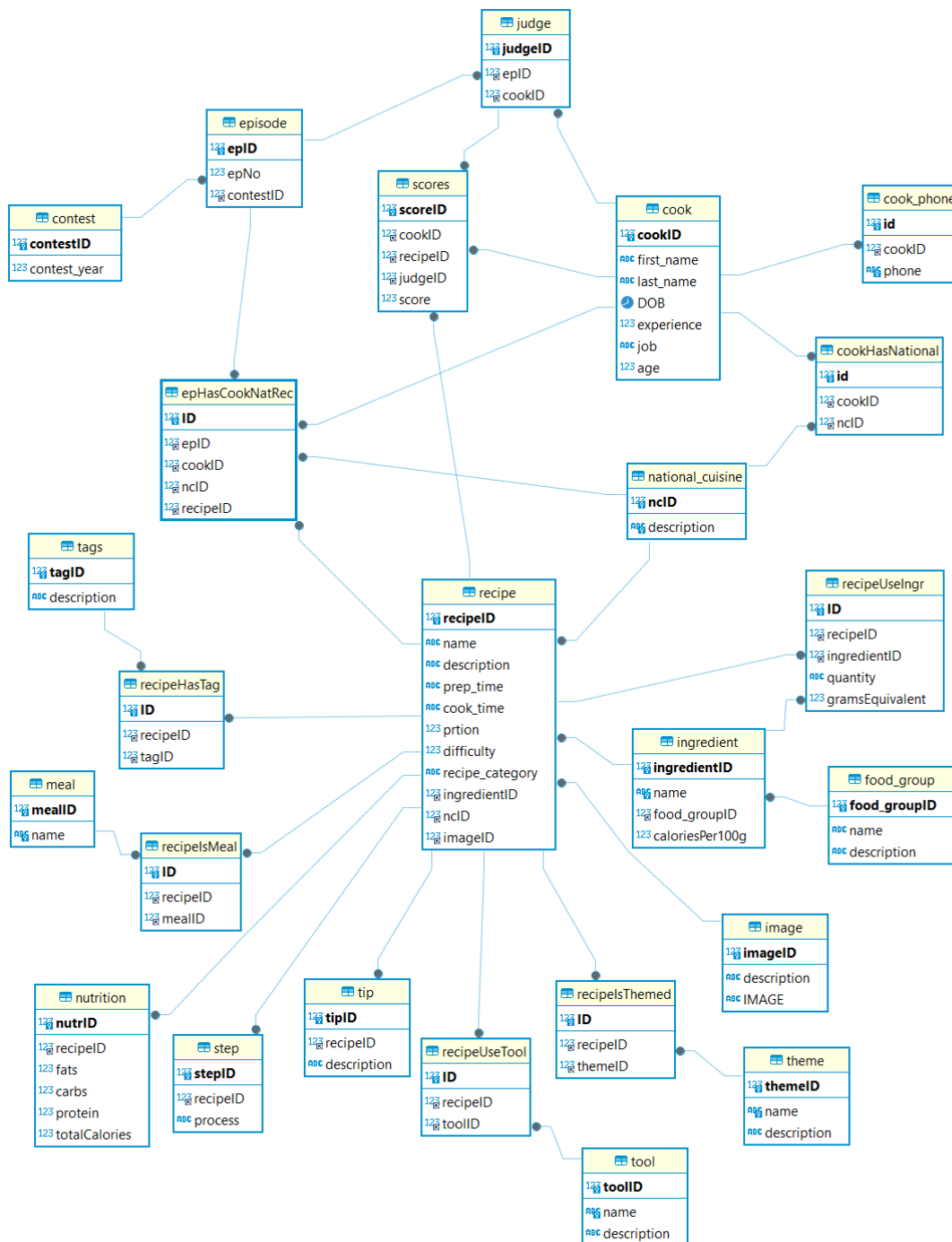


Οι σχέσεις που χρησιμοποιούνται στο διάγραμμα, περιγράφονται ως εξής:

- **recipeHasImage**: Αντιστοιχίζει την εικόνα (image) της συνταγής με την συνταγή (recipe).
- **recipeUseIngr**: Αντιστοιχίζει την συνταγή (recipe) με τα υλικά (ingredient) που χρειάζεται.
- **recipeUseTool**: Αντιστοιχίζει την συνταγή (recipe) με τον εξοπλισμό (tool) που χρειάζεται.
- **recipeHasNational**: Αντιστοιχίζει την κάθε συνταγή (recipe) με την εθνική κουζίνα (national_cuisine) που ανήκει.
- **ingrIsFoodGroup**: Ορίζει στο κάθε υλικό (ingredient) την ομάδα τροφίμων (food_group) στην οποία ανήκει.
- **cookHasNational**: Αντιστοιχίζει τον κάθε μάγειρα (cook) με τις εθνικές κουζίνες (national_cuisine) που εξειδικεύεται.
- **judgelsCook**: Ορίζει τους μάγειρες (cook) που έχουν πάρει την αρμοδιότητα κριτή (judge)
- **episodeInSeason**: Αντιστοιχίζει τα επεισόδια (episode) με τον διαγωνισμό (contest) και κατά συνέπεια το έτος και season.
- **epHasJudge**: Αντιστοιχίζει τους μάγειρες (cook) με τα επεισόδια (episode) στα οποία έχουν την αρμοδιότητα του κριτή (judge).

- cookScores: Αντιστοιχίζει τον βαθμό (score) που δίνει ο κάθε κριτής (judge) σε κάποιον μάγειρα (cook).
- epHasCookNatRecipe: Ορίζει σε ένα επεισόδιο (episode), μια εθνική κουζίνα (national_cuisine), έναν μάγειρα (cook) αντιπρόσωπο της εθνικής κουζίνας αυτής και μια συνταγή (recipe) της.

1.2. Σχεσιακό Διάγραμμα (Relational Diagram)



Στο διάγραμμα αυτό φαίνεται η υλοποίηση του ER διαγράμματος. Οι πίνακες αντιστοιχούν στα προηγούμενα entities με τα attributes να αποτελούν πλέον τις στήλες των πινάκων. Πίνακες, επίσης, αποτελούν οι σχέσεις του ER διαγράμματος.

Κατά τον σχεδιασμό του relational schema, γίνεται βελτιστοποίηση των πινάκων. Συγκεκριμένα, οι σχέσεις recipeHasImage, recipeHasNational, judgeIsCook, epHasJudge, ingrlsFoodGroup δεν υπάρχουν στο σχήμα αφού είτε έχουν συγχωνευθεί μεταξύ τους για την αποφυγή διπλής αποθήκευσης ίδιων δεδομένων, είτε έχουν εκφραστεί μέσω Foreign Key (αν για παράδειγμα μιλάμε για 1:1 σχέση).

Για την υλοποίηση της βάσης θα κάνουμε χρήση της MariaDB η οποία δεν υποστηρίζει multivalued attributes, επομένως, χρειάστηκε τα attributes: cook_phone, tips, steps, meal, tag, theme να μεταφραστούν σε πίνακες.

Ακόμα, ορίζοντας ένα attribute ως Primary Key και χρησιμοποιώντας ένα ως Foreign Key, αυτόματα η MariaDB τα ορίζει ως indexes.

1.3. Υποθέσεις και Παραδοχές

Κατά τον σχεδιασμό της βάσης χρειάστηκε να θεωρήσουμε ή διευκρινίσουμε τα εξής, σχετικά με το σενάριο που δόθηκε:

- Ο κάθε μάγειρας μπορεί να εκτελέσει μόνο συνταγές οι οποίες ανήκουν στις εθνικές κουζίνες που είναι εξειδικευμένος.
- Σε κάθε επεισόδιο επιλέγονται αυτόματα, με τυχαίο τρόπο, 10 εθνικές κουζίνες. Για κάθε μια από αυτές επιλέγεται 1 από τους μάγειρες που εξειδικεύεται σε αυτή. Για κάθε μια από αυτές επιλέγεται μια συνταγή με αυτή την εθνικότητα και ανατίθεται στον αντίστοιχο μάγειρα για την εθνική αυτή κουζίνα.
- Δεν μπορεί κάποιος μάγειρας/κριτής/ εθνική κουζίνα/ συνταγή να συμμετέχει συνεχόμενα σε περισσότερα από 3 επεισόδια ανά έτος/ season. Με το πέρας των 10 επεισοδίων, δηλαδή μιας season, η κλήρωση μηδενίζει, δεν έχει χρησιμοποιηθεί καμία εθνική κουζίνα, κανένας μάγειρας, καμία συνταγή.
- Οι μάγειρες έχουν τη δυνατότητα να επεξεργαστούν όλα τα στοιχεία των συνταγών που τους έχουν ανατεθεί και επίσης να προσθέσουν νέα συνταγή, υλικά (ingredients), tools (εξοπλισμό), tags (ετικέτες), themes (θεματικές ενότητες), meals (γεύματα).

2. DDL και DML Scripts

2.1. DDL Script

```
DROP DATABASE IF EXISTS project27;
```

```
CREATE DATABASE project27;  
USE project27;
```

```
DROP TABLE IF EXISTS contest;  
CREATE TABLE contest(  
  contestID INT NOT NULL AUTO_INCREMENT,  
  -- season INT NOT NULL,  
  contest_year INT(4) NOT NULL,  
  PRIMARY KEY (contestID)  
);
```

```
DROP TABLE IF EXISTS episode;  
CREATE TABLE episode(  
  epID INT NOT NULL AUTO_INCREMENT,  
  epNo INT NOT NULL ,  
  contestID INT NOT NULL,  
  PRIMARY KEY (epID),  
  FOREIGN KEY (contestID) REFERENCES contest(contestID)  
);
```

```
DROP TABLE IF EXISTS image;  
CREATE TABLE image(  
  imageID INT NOT NULL AUTO_INCREMENT,  
  description VARCHAR(200) NOT NULL,  
  IMAGE TEXT,  
  PRIMARY KEY(imageID)  
);
```

```
DROP TABLE IF EXISTS national_cuisine;  
CREATE TABLE national_cuisine(  
  ncID int NOT NULL AUTO_INCREMENT,  
  description VARCHAR(300) NOT NULL,  
  UNIQUE(description),  
  PRIMARY KEY (ncID)  
);
```

```
DROP TABLE IF EXISTS meal;  
CREATE TABLE meal(  
  mealID INT NOT NULL AUTO_INCREMENT,  
  name VARCHAR(40),  
  UNIQUE (name),  
  PRIMARY KEY (mealID)  
);
```

```
DROP TABLE IF EXISTS tags;
```

```
CREATE TABLE tags(  
tagID INT NOT NULL AUTO_INCREMENT,  
description VARCHAR(40),  
PRIMARY KEY (tagID)  
);
```

```
DROP TABLE IF EXISTS theme;  
CREATE TABLE theme(  
themeID INT NOT NULL AUTO_INCREMENT,  
name VARCHAR(80) NOT NULL,  
description VARCHAR(300) NOT NULL,  
UNIQUE(name),  
PRIMARY KEY (themeID)  
);
```

```
DROP TABLE IF EXISTS food_group;  
CREATE TABLE food_group(  
food_groupID INT NOT NULL AUTO_INCREMENT,  
name VARCHAR(40) NOT NULL,  
description LONGTEXT NOT NULL,  
PRIMARY KEY (food_groupID)  
);
```

```
DROP TABLE IF EXISTS tool;  
CREATE TABLE tool(  
toolID INT NOT NULL AUTO_INCREMENT,  
name VARCHAR(30) NOT NULL UNIQUE,  
description VARCHAR(200) NOT NULL,  
UNIQUE(name),  
PRIMARY KEY (toolID)  
);
```

```
DROP TABLE IF EXISTS ingredient;  
CREATE TABLE ingredient (  
ingredientID INT NOT NULL AUTO_INCREMENT,  
name VARCHAR(20) NOT NULL unique,  
food_groupID INT NOT NULL,  
caloriesPer100g INT NOT NULL,  
UNIQUE (name),  
PRIMARY KEY (ingredientID),  
FOREIGN KEY (food_groupID) REFERENCES food_group (food_groupID)  
);
```

```
DROP TABLE IF EXISTS recipe;  
CREATE TABLE recipe(  
recipeID INT NOT NULL AUTO_INCREMENT,  
name VARCHAR(255) NOT NULL,  
description VARCHAR(600) NOT NULL,  
prep_time VARCHAR(20) NOT NULL,  
cook_time VARCHAR(20) NOT NULL,
```

```

portion INT NOT NULL,
difficulty INT
CHECK (difficulty IN (1, 2, 3, 4, 5)),
recipe_category VARCHAR(20)
CHECK (recipe_category IN ('Desert Recipe', 'Cooking Recipe')),
ingredientID INT NOT NULL,
ncID INT NOT NULL,
imageID INT NOT NULL,
PRIMARY KEY (recipeID),
FOREIGN KEY (ingredientID) REFERENCES ingredient (ingredientID),
FOREIGN KEY (ncID) REFERENCES national_cuisine (ncID),
FOREIGN KEY (imageID) REFERENCES image(imageID)
);

```

```

DROP TABLE IF EXISTS tip;
CREATE TABLE tip(
tipID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
description LONGTEXT,
PRIMARY KEY (tipID),
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID)
);

```

```

DROP TABLE IF EXISTS step;
CREATE TABLE step(
stepID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
process LONGTEXT NOT NULL,
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
PRIMARY KEY (stepID)
);

```

```

DROP TABLE IF EXISTS nutrition;
CREATE TABLE nutrition(
nutrID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
fats FLOAT NOT NULL,
carbs FLOAT NOT NULL,
protein FLOAT NOT NULL,
totalCalories FLOAT,
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
PRIMARY KEY (nutrID)
);

```

```

DROP TABLE IF EXISTS recipeIsMeal;
CREATE TABLE recipeIsMeal(
ID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
mealID INT NOT NULL,
PRIMARY KEY (ID),

```



```
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
FOREIGN KEY (mealID) REFERENCES meal (mealID)
);
```

```
DROP TABLE IF EXISTS recipeHasTag;
CREATE TABLE recipeHasTag(
ID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
tagID INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
FOREIGN KEY (tagID) REFERENCES tags (tagID)
);
```

```
DROP TABLE IF EXISTS recipeUseTool;
CREATE TABLE recipeUseTool(
ID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
toolID INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
FOREIGN KEY (toolID) REFERENCES tool (toolID)
);
```

```
DROP TABLE IF EXISTS recipeUseIngr;
CREATE TABLE recipeUseIngr(
ID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
ingredientID INT NOT NULL,
quantity VARCHAR(100),
gramsEquivalent DECIMAL(10, 2),
PRIMARY KEY (ID),
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
FOREIGN KEY (ingredientID) REFERENCES ingredient (ingredientID)
);
```

```
DROP TABLE IF EXISTS recipeIsThemed;
CREATE TABLE recipeIsThemed(
ID INT NOT NULL AUTO_INCREMENT,
recipeID INT NOT NULL,
themeID INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
FOREIGN KEY (themeID) REFERENCES theme (themeID)
);
```

```
DROP TABLE IF EXISTS cook;
CREATE TABLE cook(
cookID INT NOT NULL AUTO_INCREMENT,
first_name VARCHAR(20) NOT NULL,
```

```
last_name VARCHAR(20) NOT NULL,  
DOB DATE NOT NULL,  
experience INT NOT NULL,  
job ENUM ('3rd cook' , '2nd cook' , '1st cook' , 'assistant cook' , 'chef'),  
age INT,  
PRIMARY KEY (cookID)  
);
```

```
DROP TABLE IF EXISTS cook_phone;  
CREATE TABLE cook_phone(  
id INT NOT NULL AUTO_INCREMENT,  
cookID INT NOT NULL,  
phone VARCHAR(20) NOT NULL UNIQUE,  
PRIMARY KEY (id),  
FOREIGN KEY (cookID) REFERENCES cook(cookID)  
);
```

```
DROP TABLE IF EXISTS cookHasNational;  
CREATE TABLE cookHasNational(  
id INT NOT NULL AUTO_INCREMENT,  
cookID INT NOT NULL,  
ncID INT NOT NULL,  
PRIMARY KEY (id),  
FOREIGN KEY (cookID) REFERENCES cook(cookID),  
FOREIGN KEY (ncID) REFERENCES national_cuisine (ncID)  
);
```

```
DROP TABLE IF EXISTS judge;  
CREATE TABLE judge(  
judgeID INT NOT NULL AUTO_INCREMENT,  
epID INT NOT NULL,  
cookID INT NOT NULL,  
PRIMARY KEY (judgeID) ,  
FOREIGN KEY (epID) REFERENCES episode(epID),  
FOREIGN KEY (cookID) REFERENCES cook(cookID)  
);
```

```
DROP TABLE IF EXISTS scores;  
CREATE TABLE scores (  
scoreID INT AUTO_INCREMENT,  
cookID INT NOT NULL,  
recipeID INT NOT NULL,  
judgeID INT NOT NULL,  
score INT NOT NULL,  
PRIMARY KEY (scoreID),  
FOREIGN KEY (cookID) REFERENCES cook (cookID),  
FOREIGN KEY (recipeID) REFERENCES recipe (recipeID),  
FOREIGN KEY (judgeID) REFERENCES judge (judgeID)  
);
```

```
DROP TABLE IF EXISTS epHasCookNatRec;
CREATE TABLE epHasCookNatRec(
ID INT NOT NULL AUTO_INCREMENT,
epID INT NOT NULL,
cookID INT NOT NULL,
ncID INT NOT NULL,
recipeID INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (ncID) REFERENCES national_cuisine(ncID),
FOREIGN KEY (recipeID) REFERENCES recipe(recipeID),
FOREIGN KEY (epID) REFERENCES episode(epID),
FOREIGN KEY (cookID) REFERENCES cook(cookID)
);
```

```
DROP TABLE IF EXISTS user;
CREATE TABLE user(
ID INT NOT NULL AUTO_INCREMENT,
username VARCHAR(30) NOT NULL UNIQUE,
password VARCHAR(10) NOT NULL,
userType VARCHAR(20)
CHECK (userType IN ('admin', 'cook')),
cookID INT,
PRIMARY KEY (ID)
);
```

2.2. DML Script

Το αρχείο αυτό είναι μεγάλης έκτασης. Για τον λόγο αυτό θα παρατεθεί η εισαγωγή κάποιων δεδομένων.

Το συνολικό αρχείο μπορείτε να το δείτε στο git repo, από τον εξής σύνδεσμο:

<https://github.com/marinanthi/Project27>

```
INSERT INTO image (imageID, description, IMAGE) VALUES
(1, 'Lentil Soup', 'https://www.recipetineats.com/wp-content/uploads/2017/03/Lentil-Soup-5.jpg?resize=900,643'),
(2, 'Fried Rice', 'https://www.tasteofhome.com/wp-content/uploads/2018/01/Mom-s-Fried-Rice_exps47879_TH1789928D70B_RMS-2.jpg?fit=696,696'),
(3, 'Tabbouleh', 'https://rawmanda.com/wp-content/uploads/2016/03/Vegan-Buckwheat-Mango-Tabbouleh-Lunch-Idea-Recipe.jpg'),
(4, 'Chicken Potato', 'https://letthebakingbegin.com/wp-content/uploads/2014/05/One-Pan-Chicken-and-Potatoes-by-Let-the-Baking-Begin_-300x300.jpg'),
...
```

```
INSERT INTO national_cuisine (ncID, description) VALUES
(1, 'Western'),
(2, 'Chinese'),
(3, 'Lebanon'),
(4, 'American'),
(5, 'Italian'),
(6, 'Brussels'),
(7, 'Mexican'),
...
```

```
INSERT INTO food_group(food_groupID, name, description) VALUES
(1,'Vegetables', 'All raw vegetables (e.g., lettuce, cabbage, carrot, tomato, cucumber, onion, etc.), all cooked vegetables (e.g., broccoli, cauliflower, zucchini, greens, beets, etc.), and starchy vegetables (e.g., peas, corn, squash) belong in the vegetable category. Potatoes and their varieties are not included. '),
(2,'Fruits', 'All raw fruits (e.g., orange, apple, pear, banana, peach, etc.), all dried fruits (e.g., prunes, raisins, apricots, etc.), and natural fruit juices (100% without added sugar) belong in the fruit category. '),
(3,'Cereals and Potatoes', 'Cereals include subcategories such as wheat, oats, barley, rye, and rice. They also include cereal products like flour, bread, simple bakery products (e.g., toast, rusks, breadsticks, crackers), complex bakery products (e.g., doughs, pies), pasta (e.g., spaghetti, barley pasta, noodle pasta), and various cereal products (e.g., bulgur, trahana) and breakfast cereals. This category includes potatoes and their varieties. '),
(4,'Milk and Dairy Products', 'The category includes milk, dairy products (e.g., yogurt, cheese, sour milk). Butter is not included (it is classified under fats and oils). '),
...
```

```
INSERT INTO ingredient (ingredientID, name, food_groupID, caloriesPer100g) VALUES
```

```
(1, 'olive oil', 10, 890),
(2, 'onion', 1, 44),
(3, 'garlic', 1, 44),
(4, 'carrot', 1, 41),
(5, 'celery', 1, 21),
(6, 'lentils', 5, 114),
(7, 'tomato', 1, 22),
...
```

```
INSERT INTO recipe (recipeID, name, description, prep_time, cook_time, prtion, difficulty,
recipe_category, ingredientID, ncID, imageID) VALUES
(1, 'Lentil Soup', 'A well made Lentil Soup, recipe is can't-stop-eating-it good. You'll
go back for seconds and thirds, then you'll be taking big tubs of it to work for lunch and
happily have it for dinner again. And - I'm going to say it (*head swell*) - plenty of
readers have said this is the best lentil soup they've ever had!', '10 min', '45 min', 6,
3, 'Cooking Recipe', 6, 1, 1),
(2, 'Fried Rice', 'A quick dish with simple ingredients that you will always come back
to.', '5 min', '20 min', 4, 1, 'Cooking Recipe', 15, 2, 2),
(3, 'Tabbouleh', 'The best way to ensure a healthy week of eating is being prepared ahead
of time.', '15 min', '20 min', 4, 1, 'Cooking Recipe', 18, 3, 3),
(4, 'Chicken Potato', 'Just toss chicken thighs, potatoes and carrots in the baking dish
with seasoning & roast!', '5 mins', '1 hour and 20 mins', 5, 3, 'Cooking Recipe', 20, 11,
4),
(5, 'Roasted Pork with Grapes and Couscous', 'Ready to serve in just 30 minutes, you
simply cannot go wrong with this juicy roasted pork', '5 min', '25 min', 4, 5, 'Cooking
Recipe', 25, 4, 5),
...
```

```
INSERT INTO recipeUseIngr (recipeID, ingredientID, quantity, gramsEquivalent) VALUES
(1, 1, '2 tbsp', 14),
(1, 2, '1 whole', 150),
(1, 3, '2 cloves', 5),
(1, 4, '1 large', 110),
(1, 5, '2 ribs', 40),
(1, 6, '400 gr', 400),
(1, 7, '400 gr', 400),
...
```

```
INSERT INTO step(stepID, recipeID, process) VALUES
(1, 1, 'Heat oil in a large pot over medium heat. Add garlic and onion, cook for 2 minutes.
Add celery and carrot. Cook for 7 - 10 minutes or until softened and the onion is sweet.
Don't rush this step, it is key to the flavour base of the soup. Add all remaining
ingredients except the lemon and salt. Stir. Increase heat and bring to simmer.
Scoop scum on the surface off and discard (do this again during cooking if required).
Place lid on and turn heat down to medium low. Simmer for 35 - 40 minutes or until lentils
are soft. Remove bay leaves. Thicken Soup: Using a stick blender, do 2 or 3 quick whizzes
to thicken the soup (see video below). Or transfer 2 cups to a blender, let it cool
slightly, then hold lid with tea towel and blend then transfer back into pot. Add a touch
```

of water if you want to adjust soup consistency. Season to taste with salt and pepper. Grate over the zest of the lemon then add a squeeze of lemon juice just before serving. Garnish with parsley if desired and serve with warm crusty bread slathered liberally with butter!'),

(2, 2, 'In a large skillet, heat oil over medium-high heat. Pour egg into the pan. As egg sets, lift edges, letting uncooked portion flow underneath. When egg is completely cooked, remove to a plate. Set aside. In the same skillet, cook bacon over medium heat until crisp. Using a slotted spoon, remove to paper towels; drain, reserving 2 tablespoons drippings. Saute mushrooms and onions in the drippings. Stir in the rice, bean sprouts, peas, soy sauce and bacon. Chop egg into small pieces; stir into the pan and heat through.'),

...

INSERT INTO nutrition (nutrID, recipeID, fats, carbs, protein) VALUES

(1, 1, 5, 48, 18),
(2, 2, 15, 44, 14),
(3, 3, 10.5, 45.2, 7.4),
(4, 4, 38, 38, 36),
(5, 5, 40, 32.4, 41.1),
(6, 6, 22, 44, 7),
(7, 7, 7, 22, 1),
(8, 8, 17.1, 82, 23.6),
(9, 9, 4, 31, 5),
...

INSERT INTO meal (mealID, name) VALUES

(1, 'Breakfast'),
(2, 'Lunch'),
(3, 'Dinner'),
(4, 'Snack'),
(5, 'Dessert'),
(6, 'Beverages'),
(7, 'Apetizer');

INSERT INTO recipeIsMeal(recipeID, mealID) VALUES

(1, 3), (1, 2),
(2, 3), (2, 2),
(3, 2),
(4, 2), (4, 3),
(5, 2), (5, 3),
(6, 2),
(7, 4), (7, 5),
...

INSERT INTO tags (tagID, description) VALUES

(1, 'Brunch'),

```
(2,'quick-lunch'),
(3,'cold dish'),
(4,'Comfort Food'),
(5,'Low Calorie'),
(6,'5 or less ingredients'),
...
```

```
INSERT INTO recipeHasTag(recipeID, tagID) VALUES
```

```
(1, 4),
(2, 2), (2, 9),
(3, 2),
(4, 4), (4, 9),
(5, 8), (5, 10),
(9, 3), (9, 4), (9, 9),
...
```

```
INSERT INTO tip (tipID, recipeID, description) VALUES
```

```
(1, 1, 'This freezes extremely well! Or keeps in the fridge for 3 to 5 days. '),
(2, 1, 'Cook times vary slightly as well so just start checking if the lentils are done at around 30 minutes. '),
(3, 2, 'Feel free to add cubed chicken or steak to make this side a main dish. Pea pods also add some color and crunch. '),
(4, 3, 'Mix in hummus before serving'),
(5, 4, 'Garlic makes the chicken and potatoes very fragrant and delicious!'),
(6, 4, 'Eat while still warm!!!'),
...
```

```
INSERT INTO tool(toolID, name, description) VALUES
```

```
(1, 'spoon', 'A spoon is used for mixing or measuring ingredients'),
(2, 'dish', 'Dishes are used for serving foods, saving foods etc. '),
(3, 'frying pan', 'A frying pan also known as skillet is a flat bottomed pan used for frying, searing and browning foods'),
...
```

```
INSERT INTO recipeUseTool(recipeID, toolID) VALUES
```

```
(1, 1), (1, 2), (1, 6),
(2, 5), (2, 1),
(3, 1), (3, 3),
(4, 3), (4, 1), (4, 4),
(5, 7), (5, 1), (5, 4), (5, 2),
(6, 8), (6, 1), (6, 3),
(7, 7), (7, 5), (7, 9), (7, 1),
...
```

```
INSERT INTO theme (themeID, name, description) VALUES
```

```
(1,'Village-Inspired Recipes','Traditional dishes from various villages around the world, showcasing local ingredients and cooking methods that reflect the unique culinary heritage of each region. '),
(2,'Street Food Favorites','Compile recipes that celebrate popular street foods. '),
```

```
(3,'Seasonal Harvest','Recipes based on seasonal ingredients, offering spring, summer,
autumn, and winter collections that highlight the freshest produce available at the
time. '),
(4,'Gluten Free', ' A collection of recipes entirely free from gluten, featuring dishes
like gluten-free pastas, breads, and desserts, catering to those with celiac disease or
gluten sensitivity. '),
...
```

```
INSERT INTO recipeIsThemed(recipeID, themeID) VALUES
(1, 1), (1, 3),
(2, 7),
(3, 4),
(4, 7), (4, 8),
(5, 7), (5, 8),
(6, 8), (6, 1), (6, 3),
...
```

```
INSERT INTO cook(cookID, first_name, last_name, DOB, experience, job) VALUES
(1, 'Marina', 'Kamenidou', '2001-05-15', 7, 'chef'),
(2, 'Demetra', 'Overel', '1994-08-22', 9, 'chef'),
(3, 'Lutero', 'Elmhurst', '1996-07-04', 10, '3rd cook'),
(4, 'Geno', 'Ordish', '2001-07-28', 11, '3rd cook'),
...
```

```
INSERT INTO cook_phone(cookID, phone) VALUES
(1, '8016597918'),
(2, '8891639222'),
(3, '3089863345'),
(4, '1142969807'),
(5, '6411569724'),
(6, '6387253775'),
(7, '1475493698'),
(8, '3703448932'), (8, '5356182234'),
...
```

```
INSERT INTO user (username, password, userType, cookID) VALUES
('marina', '456', 'cook', 1),
('ociciotti0', 'mU6=', 'cook', 2),
('dboshell1', 'cD0dk', 'cook', 3),
('lfoxley2', 'hS', 'cook', 4),
('dwarry3', 'sZ0lu', 'cook', 5),
('wmannix4', 'pZ6P,>9', 'cook', 6),
...
```

```
INSERT INTO cookHasNational (cookID, ncID) VALUES
(1, 1), (1, 2), (1, 3) ,
(2, 4), (2, 5), (2, 6) ,
(3, 7), (3, 8), (3, 9) ,
(4, 10), (4, 11), (4, 12) ,
...
```



```
INSERT INTO contest(contestID, contest_year) VALUES
```

```
(1, 2020),  
(2, 2021),  
(3, 2022),  
(4, 2023),  
(5, 2024);
```

```
INSERT INTO episode (epID, epNo, contestID) VALUES
```

```
(1, 1, 1), (2, 2, 1), (3, 3, 1), (4, 4, 1), (5, 5, 1), (6, 6, 1), (7, 7, 1), (8, 8, 1),  
(9, 9, 1), (10, 10, 1),  
...
```

```
INSERT INTO epHasCookNatRec (epID,cookID,ncID,recipeID) VALUES
```

```
(1,27,23,43), (1,43,16,44), (1,17,22,42), (1,4,12,27), (1,54,11,22), (1,17,21,34),  
(1,31,9,17), (1,3,8,11), (1,42,13,29), (1,19,28,50),  
(2,11,3,3), (2,40,7,10), (2,49,6,8), (2,9,26,35), (2,18,24,36), (2,28,27,49),  
(2,22,10,20), (2,37,25,47), (2,5,15,32), (2,1,1,39),  
(3,41,9,17), (3,6,17,31), (3,30,4,24), (3,17,23,43), (3,26,20,41), (3,26,22,42),  
(3,14,14,30), (3,21,5,6), (3,44,19,26), (3,19,28,38),  
(4,8,24,36), (4,51,3,3), (4,51,1,21), (4,46,25,47), (4,40,6,8), (4,54,10,19),  
...
```

```
INSERT INTO judge (epID,cookID) VALUES
```

```
(1,9), (1,20), (1,50), (2,17), (2,17), (2,16), (3,24), (3,9), (3,2), (4,15),  
(4,7), (4,31), (5,41), (5,1), (5,1), (6,14), (6,5), (6,18), (7,33), (7,43),  
(7,30), (8,47), (8,9), (8,11), (9,29), (9,24), (9,14), (10,53), (10,37), (10,26),  
(11,10), (11,5), (11,34), (12,42), (12,23), (12,45), (13,26), (13,4), (13,32), (14,31),  
...
```

```
INSERT INTO scores (cookID,recipeID,judgeID,score) VALUES
```

```
(27,43,1,1),  
(43,44,1,4),  
(17,42,1,2),  
(4,27,1,4),  
(54,22,1,1),  
...
```

Το DML αρχείο, περιέχει όλους τους πίνακες ήδη γεμάτους. Για να γεμίσουμε, ωστόσο, με δεδομένα κάποιους από τους πίνακες της βάσης, λόγω μεγάλου όγκου που θα έπρεπε να εισαχθεί αλλά και πολυπλοκότητας, έγινε χρήση κώδικα και στη συνέχεια, export τα δεδομένα αυτά για να μπουν στο DML script.

Συγκεκριμένα :

- Ο πίνακας epHasCookNatRec αποθηκεύει το ID του επεισοδίου, τα ID από τις 10 εθνικές κουζίνες που επιλέγονται τυχαία, το ID του 1 μάγειρα από την κάθε εθνική κουζίνα και το ID της 1 συνταγής από την κάθε εθνική κουζίνα που αντιστοιχεί στον μάγειρα της εθνικής αυτής κουζίνας.

Ο παρακάτω κώδικας:

1. Ελέγχει αν ο χρήστης είναι συνδεδεμένος.
2. Ελέγχει ότι ο χρήστης είναι admin και όχι μάγειρας.
3. Δέχεται input την χρονιά που θα διαδραματιστεί ο νέος διαγωνισμός.
4. Κάνει INSERT στον πίνακα της βάσης contest την χρονιά που δόθηκε από τον χρήστη.
5. Ορίζει έναν πίνακα episodes[], ο οποίος αποθηκεύει ID επεισοδίου, εθνικής κουζίνας, μάγειρα, κριτή και συνταγής. Ένα επεισόδιο, αποτελεί και ένα object.
6. Κάνει INSERT στον πίνακα της βάσης episode.
7. Κάνει SELECT 10 εθνικές κουζίνες με τυχαίο τρόπο.
8. Για την κάθε κουζίνα επιλέγει έναν μάγειρα.
Από την δεύτερη επανάληψη και μετά εξαιρεί τους μάγειρες του προηγούμενου επεισοδίου.
9. Για την κάθε κουζίνα επιλέγει μια συνταγή.
Από την δεύτερη επανάληψη και μετά εξαιρεί τις συνταγές του προηγούμενου επεισοδίου.
10. Με τα ID που δημιουργούνται στον πίνακα episode, κάνει INSERT τα παραπάνω δεδομένα στον πίνακα epHasCookNatRec.
11. Επιλέγει 3 κριτές από όλους τους μάγειρες, εξαιρώντας αυτούς που συμμετέχουν στο επεισόδιο ως μάγειρες.
12. Κάνει INSERT τους κριτές στον πίνακα judges.
- 13.

```
app.get("/createcontest", async function (req, res) {
```

```
  if (req.session.loggedin !== true) {  
    return res.send("You are not logged in");  
  }  
  
  if (req.session.userType !== "admin") {  
    return res.send("You are not an admin user");  
  }  
}
```

```
let episodes = [];  
let contestID;  
const contest_year = req.query.contest_year;
```

```

if (!contest_year) {
  return res.send('You have to provide year.')
}

try {
  let [results, fields] = await connection.query(
    `SELECT * FROM contest WHERE contest_year = ? ;`,
    [
      contest_year
    ]
  );

  if (results.length > 0) {
    return res.send('This contest already exists.')
  }

  [results, fields] = await connection.query(
    `INSERT INTO contest(contest_year)
    VALUES (?) ;`,
    [
      contest_year
    ]
  );

  contestID = results.insertId;
} catch (err) {
  console.log(err);
}

try {
  for (let j = 0; j < 10; j++) {
    episodes.push({
      nationalCuisineIDs: [],
      cookIDs: [],
      judgeIDs: [],
      recipeIDs: [],
      epID: -1,
      epNo: j+1
    })

    const [results, fields] = await connection.query(
      `INSERT INTO episode (epNo, contestID) VALUES
      (?,?) ;`,
      [
        episodes[j].epNo,
        contestID
      ]
    );
  }
}

```

```

    episodes[j].epID = results.insertId;
}

for (let e = 0; e < 10; e++) {

    let excludeNationalCuisinesIDs = []
    if (e === 0) {
        excludeNationalCuisinesIDs = [-1]
    }
    else {
        excludeNationalCuisinesIDs = episodes[e - 1].nationalCuisineIDs
    }
    const [results, fields] = await connection.query(
        `SELECT ncID
        FROM national_cuisine nc
        WHERE ncID NOT IN ( ? )
        ORDER BY RAND()
        LIMIT 10 ;`,
        [
            excludeNationalCuisinesIDs
        ]
    );

    episodes[e].nationalCuisineIDs = results;

    // Converting objects to numbers
    for (let i = 0; i < episodes[e].nationalCuisineIDs.length; i++) {
        episodes[e].nationalCuisineIDs[i] = episodes[e].nationalCuisineIDs[i].ncID
    }

    let excludeCookIDs = []
    if (e === 0) {
        excludeCookIDs = [-1]
    }
    else {
        excludeCookIDs = episodes[e - 1].cookIDs
    }
    for (let i = 0; i < episodes[e].nationalCuisineIDs.length; i++) {

        const [results, fields] = await connection.query(
            `SELECT cookID
            FROM cookHasNational chn
            WHERE ncID = ? AND cookID NOT IN ( ? )
            ORDER BY RAND()
            LIMIT 1;`,
            [
                episodes[e].nationalCuisineIDs[i],
                excludeCookIDs
            ]
        );
    }
}

```

```

);

let cookID = results[0].cookID;
episodes[e].cookIDs.push(cookID);
}

let excludeRecipeIDs = []
if (e === 0) {
  excludeRecipeIDs = [-1]
}
else {
  excludeRecipeIDs = episodes[e - 1].recipeIDs
}
for (let i = 0; i < episodes[e].cookIDs.length; i++) {

  const [results, fields] = await connection.query(
    `SELECT recipeID
    FROM recipe r
    WHERE ncID = ? AND recipeID NOT IN ( ? )
    ORDER BY RAND()
    LIMIT 1;`,
    [
      episodes[e].nationalCuisineIDs[i],
      excludeRecipeIDs
    ]
  );

  let recipeID = results[0].recipeID;
  episodes[e].recipeIDs.push(recipeID);
}

let excludeJudgeIDs = []
if (e === 0) {
  excludeJudgeIDs = [-1]
}
else {
  excludeJudgeIDs = episodes[e - 1].judgeIDs
}
for (let i = 0; i < 3; i++) {

  const [results, fields] = await connection.query(
    `SELECT cookID
    FROM cook
    WHERE cookID NOT IN ( ? ) AND cookID NOT IN ( ? )
    ORDER BY RAND()
    LIMIT 1;`,
    [
      excludeJudgeIDs,
      episodes[e].cookIDs
    ]
  );

```

```

    ]
  );

  let judgeID = results[0].cookID;
  episodes[e].judgeIDs.push(judgeID);
}

for (let i = 0; i < episodes[e].nationalCuisineIDs.length; i++) {

  const [results, fields] = await connection.query(
    `INSERT INTO epHasCookNatRec (epID, ncID, cookID, recipeID)
    VALUES(?,?,?,?)`,
    [
      episodes[e].epID,
      episodes[e].nationalCuisineIDs[i],
      episodes[e].cookIDs[i],
      episodes[e].recipeIDs[i]
    ]
  );
}

for (let i = 0; i < episodes[e].judgeIDs.length; i++) {

  const [results, fields] = await connection.query(
    `INSERT INTO judge(epID, cookID) VALUES ( ?, ? )`,
    [
      episodes[e].epID,
      episodes[e].judgeIDs[i]
    ]
  );
}

}

res.send('You created a contest successfully!!!');

} catch (err) {
  console.log(err);
}
});

```

- Ο πίνακας scores αποθηκεύει το ID του κριτή, του μάγειρα, της συνταγής και τον βαθμό που βάζει ο κάθε κριτής στον μάγειρα για την συνταγή που εκτέλεσε.
Ο παρακάτω κώδικας:
 1. Κάνει SELECT και αποθηκεύει σε τρεις πίνακες, τα ID των επεισοδίων με τα αντίστοιχα ID κριτών, συνταγών και μαγείρων.
 2. Ορίζει έναν πίνακα και για όλους τους συνδυασμούς, αποθηκεύει με την εντολή Math.random, τυχαίους αριθμούς ενώ με την εντολή Math.floor τους περιορίζει στο διάστημα [1,5].
 3. Κάνει INSERT τα ID και τα τυχαία score που έχει αποθηκεύσει στο αντίστοιχο πίνακα scores.

```
app.get("/runcontest", async function (req, res) {

  try {
    [results, fields] = await connection.query(
      `SELECT j.epID, judgeID, ehcnr.cookID, recipeID
      FROM epHasCookNatRec ehcnr
      JOIN judge j ON ehcnr.epID = j.epID;`, );

    let cookIDs= [];
    let recipeIDs= [];
    let judgeIDs= [];
    let scores=[];

    for (let i = 0; i < results.length; i++) {
      recipeIDs[i] = results[i].recipeID
      judgeIDs[i] = results[i].judgeID
      cookIDs[i] = results[i].cookID
      scores[i]= Math.floor(Math.random() * 5) + 1;
    }

    for (let i = 0; i < scores.length; i++) {
      const [results, fields] = await connection.query(
        `INSERT INTO scores(cookID, recipeID, judgeID, score)
        VALUES (?, ?, ?, ?);`,
        [
          cookIDs[i],
          recipeIDs[i],
          judgeIDs[i],
          scores[i]
        ]
      );
    }

    res.send('Scores added.');
```

```
  } catch (err) {
    console.log(err);
  });
});
```

- Τέλος, για τις συνταγές που εισάχθηκαν manually, χρησιμοποιήθηκε ο εξής κώδικας για τον δυναμικό υπολογισμό των συνολικών θερμίδων.

Ο παρακάτω κώδικας:

1. Κάνει SELECT όλα τα ID και τα αποθηκεύει σε έναν πίνακα ως objects.
2. Δημιουργεί ένα διάνυσμα με τις τιμές των ID.
3. Για κάθε συνταγή κάνει UPDATE την στήλη totalCalories.

```
app.get("/totalcal", async function (req, res) {

  try {

    let [results, fields] = await connection.query(
      `SELECT recipeID FROM recipe; `
    );

    const recipeIDs = results;

    for (let i = 0; i < recipeIDs.length; i++) {
      recipeIDs[i] = recipeIDs[i].recipeID
    }

    console.log(recipeIDs);

    for (let i = 0; i < recipeIDs.length; i++) {

      [results, fields] = await connection.query(
        `UPDATE nutrition n
          SET totalCalories = (SELECT ROUND(SUM( rui.gramsEquivalent *
            i.caloriesPer100g)/(100*r.prtion), 1) AS totalCalories_inGr
          FROM recipeUseIngr rui
            JOIN ingredient i ON rui.ingredientID = i.ingredientID
            JOIN recipe r ON rui.recipeID =r.recipeID
          WHERE r.recipeID=?)
          WHERE n.recipeID=?; ` ,
        [
          recipeIDs[i],
          recipeIDs[i]
        ]
      );
    }
    res.send(results);

  } catch (err) {
    console.log(err);
  }
});
```


2.3. Views

```
CREATE VIEW cook_and_recipes AS
SELECT cookID , r.recipeID, name
FROM epHasCookNatRec ehcnr
JOIN recipe r ON ehcnr.recipeID = r.recipeID
GROUP BY r.recipeID
ORDER BY cookID ASC;
```

```
CREATE VIEW episodeYearSeason AS
SELECT e.epID, e.epNo, e.contestID, c.contest_year
FROM episode e
JOIN contest c ON e.contestID = c.contestID;
```

```
-- [DONE] 3.1. Μέσος Όρος Αξιολογήσεων (σκορ) ανά μάγειρα και Εθνική κουζίνα.
CREATE VIEW score_per_cook_and_cuisine AS
SELECT chn.cookID, CONCAT(first_name, ' ', last_name) AS full_name, nc.description
FROM cook c
JOIN cookHasNational chn ON c.cookID = chn.cookID
JOIN national_cuisine nc ON chn.ncID = nc.ncID
ORDER BY cookID;
```

```
-- [DONE] 3.2. Για δεδομένη Εθνική κουζίνα και έτος, ποιοι μάγειρες ανήκουν σε αυτήν και
ποιοι μάγειρες
-- συμμετείχαν σε επεισόδια;
CREATE VIEW nationalCuisine_year_cook AS
SELECT eys.contest_year, nc.description, chn.cookID, CONCAT(first_name, ' ', last_name) AS
fullName
FROM cook c
JOIN cookHasNational chn ON c.cookID = chn.cookID
JOIN national_cuisine nc ON chn.ncID = nc.ncID
JOIN epHasCookNatRec ehcnr ON ehcnr.ncID = nc.ncID
JOIN episodeYearSeason eys ON ehcnr.epID = eys.epID
ORDER BY eys.contest_year ASC, nc.description ;
```

```
-- [DONE] 3.5. Ποιοι κριτές έχουν συμμετάσχει στον ίδιο αριθμό επεισοδίων σε διάστημα ενός
έτους με
-- περισσότερες από 3 εμφανίσεις;
CREATE VIEW judge_andEpisode_info AS
SELECT j.judgeID, j.cookID, e.epID, e.epNo, e.contestID, c.contest_year
FROM judge j
JOIN episode e ON j.epID = e.epID
JOIN contest c ON e.contestID = c.contestID ;
```

```
CREATE VIEW participation_per_yearAndJudge AS
SELECT contest_year, judgeID, cookID, COUNT(epID) AS participation
FROM judge_andEpisode_info
GROUP BY cookID, contest_year
ORDER BY contest_year, judgeID;
```

```
CREATE VIEW countOfmoreThan3 AS
```

```

SELECT contest_year, participation, COUNT(cookID) AS zeygoi
FROM participation_per_yearAndJudge
WHERE participation >= 3
GROUP BY contest_year , participation;

```

```

CREATE VIEW moreThan3 AS
SELECT p.cookID, CONCAT(first_name, ' ' , last_name) AS full_name, contest_year,
participation
FROM participation_per_yearAndJudge p
JOIN cook c ON p.cookID = c.cookID
WHERE participation >= 3;

```

-- [DONE] 3.6. Πολλές συνταγές καλύπτουν περισσότερες από μια ετικέτες. Ανάμεσα σε ζεύγη πεδίων (π.χ.

-- brunch και κρύο πιάτο) που είναι κοινά στις συνταγές, βρείτε τα 3 κορυφαία (top-3) ζεύγη που

-- εμφανίστηκαν σε επεισόδια.

```

CREATE VIEW recipes_and_tags AS
SELECT recipeID, rht.tagID, description
FROM recipeHasTag rht
JOIN tags t ON rht.tagID = t.tagID;

```

```

CREATE VIEW count_of_tagPairs AS
SELECT t1.tagID AS tagID1, t1.description AS tag1 , t2.tagID AS tagID2, t2.description AS
tag2, COUNT(*) AS pair_count
FROM recipes_and_tags t1
JOIN recipes_and_tags t2 ON t1.recipeID = t2.recipeID AND t1.tagID < t2.tagID
JOIN tags t ON t1.tagID = t.tagID
JOIN tags t3 ON t2.tagID = t3.tagID
GROUP BY t1.tagID, t2.tagID
ORDER BY pair_count DESC;

```

-- [DONE] 3.7. Βρείτε όλους τους μάγειρες που συμμετείχαν τουλάχιστον 5 λιγότερες φορές από τον μάγειρα

-- με τις περισσότερες συμμετοχές σε επεισόδια.

```

CREATE VIEW summetoxi_cook AS
SELECT cookID, COUNT(cookID) AS summetoxi
FROM epHasCookNatRec ehcnr
GROUP BY cookID ;

```

-- [DONE] 3.8. Σε ποιο επεισόδιο χρησιμοποιήθηκαν τα περισσότερα εξαρτήματα (εξοπλισμός); Ομοίως με

-- ερώτημα 3.6, η απάντησή σας θα πρέπει να περιλαμβάνει εκτός από το ερώτημα (query), -- εναλλακτικό Query Plan (πχ με force index), τα αντίστοιχα traces και τα συμπεράσματά σας από

-- την μελέτη αυτών.

```

CREATE VIEW episode_match_recipe AS
SELECT eys.epID, eys.epNO, eys.contest_year, ehcnr.recipeID, r.name
FROM epHasCookNatRec ehcnr

```

```
JOIN episodeYearSeason eys ON eys.epID = ehcnr.epID
JOIN recipe r ON ehcnr.recipeID = r.recipeID
ORDER BY ehcnr.epID;
```

```
CREATE VIEW toolCount_perRecipe AS
SELECT r.recipeID, COUNT(toolID) AS numberOfTools_perRecipe
FROM recipe r
JOIN recipeUseTool rut ON rut.recipeID = r.recipeID
GROUP BY rut.recipeID
ORDER BY rut.recipeID;
```

-- [DONE] 3.10. Ποιες Εθνικές κουζίνες έχουν τον ίδιο αριθμό συμμετοχών σε διαγωνισμούς, σε διάστημα δύο

-- συνεχόμενων ετών, με τουλάχιστον 3 συμμετοχές ετησίως

```
CREATE VIEW summetoxi_ethnikis_kouzinas AS
SELECT contest_year, ncID, COUNT(ncID) AS summetoxi
FROM epHasCookNatRec ehcnr
JOIN episode e ON ehcnr.epID = e.epID
JOIN contest c ON e.contestID = c.contestID
GROUP BY contest_year, ncID ;
```

```
CREATE VIEW pano_apo3_summetoxes AS
SELECT contest_year, ncID, summetoxi
FROM summetoxi_ethnikis_kouzinas
HAVING summetoxi > 3
ORDER BY ncID;
```

```
CREATE VIEW TwoYearParticipation AS
SELECT t1.contest_year AS year1,
       t2.contest_year AS year2,
       t1.ncID,
       t1.summetoxi AS summetoxi1,
       t2.summetoxi AS summetoxi2
FROM pano_apo3_summetoxes t1
JOIN pano_apo3_summetoxes t2 ON t1.ncID = t2.ncID
WHERE t2.contest_year = t1.contest_year + 1;
```

-- [DONE] 3.11. Βρείτε τους top-5 κριτές που έχουν δώσει συνολικά την υψηλότερη βαθμολόγηση σε ένα

-- μάγειρα. (όνομα κριτή, όνομα μάγειρα και συνολικό σκορ βαθμολόγησης)

```
CREATE VIEW judge_names AS
SELECT judgeID, j.cookID, first_name, last_name
FROM judge j
JOIN cook c ON j.cookID = c.cookID
GROUP BY judgeID;
```

```
CREATE VIEW sum_of_scores_perJudge_andCook AS
SELECT CONCAT(jn.first_name, ' ', jn.last_name ) AS judge_name,
       CONCAT(c.first_name, ' ', c.last_name) AS cook_name, c.cookID,
```

```

SUM(score) AS sunoliki_bathmologisi
FROM scores s
JOIN judge_names jn ON s.judgeID = jn.judgeID
JOIN cook c ON s.cookID = c.cookID
GROUP BY s.judgeID , s.cookID ;

```

-- [DONE] 3.12. Ποιο ήταν το πιο τεχνικά δύσκολο, από πλευράς συνταγών, επεισόδιο του διαγωνισμού ανά

-- έτος;

```

CREATE VIEW erotima_3_12 AS
SELECT contest_year, epNo, SUM(r.difficulty) AS episodesDifficulty
FROM epHasCookNatRec ehcnr
JOIN recipe r ON ehcnr.recipeID = r.recipeID
JOIN episode e ON ehcnr.epID = e.epID
JOIN contest c ON e.contestID = c.contestID
GROUP BY contest_year, epNo;

```

-- [DONE] 3.13. Ποιο επεισόδιο συγκέντρωσε τον χαμηλότερο βαθμό επαγγελματικής κατάρτισης (κριτές και

-- μάγειρες);

```

CREATE VIEW cooksProfessionalism AS
SELECT contest_year, e.epNo, ehcnr.epID, SUM(c.job) AS cookProfessionalismDegree
FROM epHasCookNatRec ehcnr
JOIN cook c ON ehcnr.cookID= c.cookID
JOIN episode e ON ehcnr.epID = e.epID
JOIN contest c2 ON e.contestID = c2.contestID
GROUP BY contest_year, epNo ;

```

```

CREATE VIEW judgesProfessionalism AS
SELECT contest_year, e.epNo, j.epID, SUM(c.job) AS judgeProfessionalismDegree
FROM judge j
JOIN cook c ON j.cookID= c.cookID
JOIN episode e ON j.epID = e.epID
JOIN contest c2 ON e.contestID = c2.contestID
GROUP BY contest_year, epNo ;

```

```

CREATE VIEW erotima_3_13 AS
SELECT cp.contest_year, cp.epNo, SUM(cookProfessionalismDegree +
judgeProfessionalismDegree) AS profDegree
FROM cooksProfessionalism cp
JOIN judgesProfessionalism jp ON cp.epID= jp.epID
GROUP BY contest_year, epNo;

```

-- [DONE] 3.14. Ποια θεματική ενότητα έχει εμφανιστεί τις περισσότερες φορές στο διαγωνισμό;

```

CREATE VIEW erotima_3_14 AS
SELECT t.themeID, t.name , COUNT(rit.themeID) AS timesUsed
FROM recipeIsThemed rit JOIN theme t
ON rit.themeID= t.themeID
GROUP BY themeID ;

```

2.4. Triggers

```
-- trigger for age calculation through DOB, after insert on cook
DELIMITER //
```

```
CREATE TRIGGER insert_age
BEFORE INSERT ON cook
FOR EACH ROW
BEGIN
    SET NEW.age = TIMESTAMPDIFF(YEAR, NEW.DOB, CURDATE());
END //
```

```
DELIMITER ;
```

```
-- trigger for updating the age, after updating DOB on cook
DELIMITER //
```

```
CREATE TRIGGER update_age_after_update
BEFORE UPDATE ON cook
FOR EACH ROW
BEGIN
    IF OLD.DOB != NEW.DOB THEN
        SET NEW.age = TIMESTAMPDIFF(YEAR, NEW.DOB, CURDATE());
    END IF;
END //
```

```
DELIMITER ;
```

```
-- trigger to make a constraint of 3 tips or less
DELIMITER //
```

```
CREATE TRIGGER 3_tips_or_less
BEFORE INSERT ON tip
FOR EACH ROW
BEGIN
    DECLARE tip_count INT;

    SELECT COUNT(*) INTO tip_count
    FROM tip
    WHERE recipeID = NEW.recipeID;
    -- Check if the count is more than 3, if yes then give ERROR
    IF tip_count >= 3 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'You have reached the maximum number of
tips for this recipe';
    END IF;
END //
```

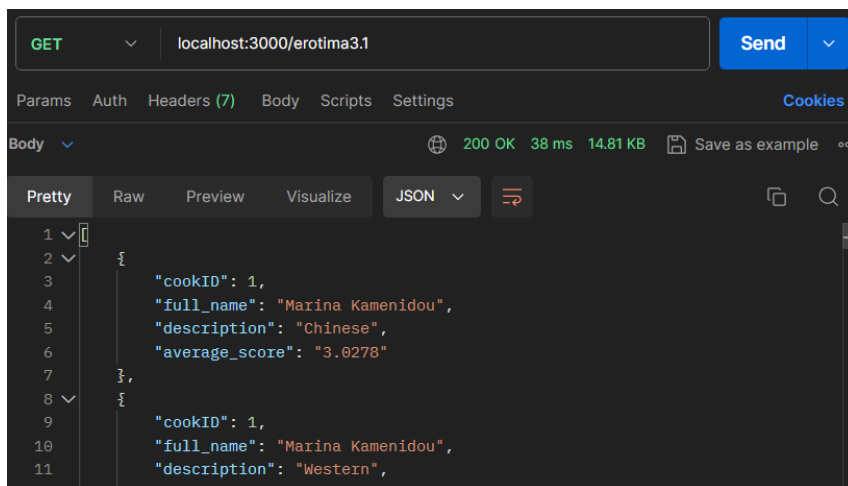
```
DELIMITER ;
```

2.5. Queries

Η API πλατφόρμα που χρησιμοποιήθηκε για να τρέξουμε τον server είναι το Postman.

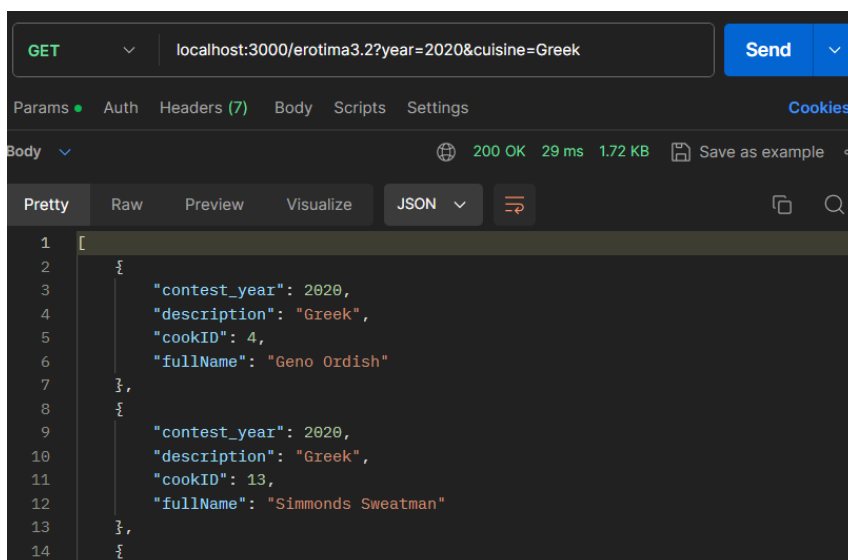
3.1. Μέσος Όρος Αξιολογήσεων (σκορ) ανά μάγειρα και Εθνική κουζίνα.

```
SELECT d.cookID, d.full_name, description, AVG(score) AS average_score
FROM denkseroposnatopo d
JOIN scores s ON d.cookID = s.cookID
GROUP BY d.cookID, description;
```



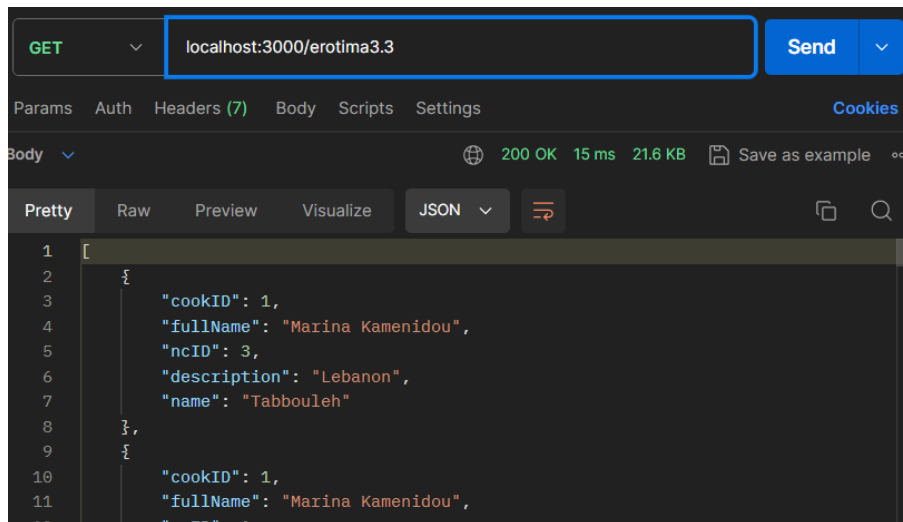
3.2. Για δεδομένη Εθνική κουζίνα και έτος, ποιοι μάγειρες ανήκουν σε αυτήν και ποιοι μάγειρες συμμετείχαν σε επεισόδια;

```
SELECT *
FROM nationalCuisine_year_cook
WHERE contest_year = 2020 AND description = 'Greek';
```



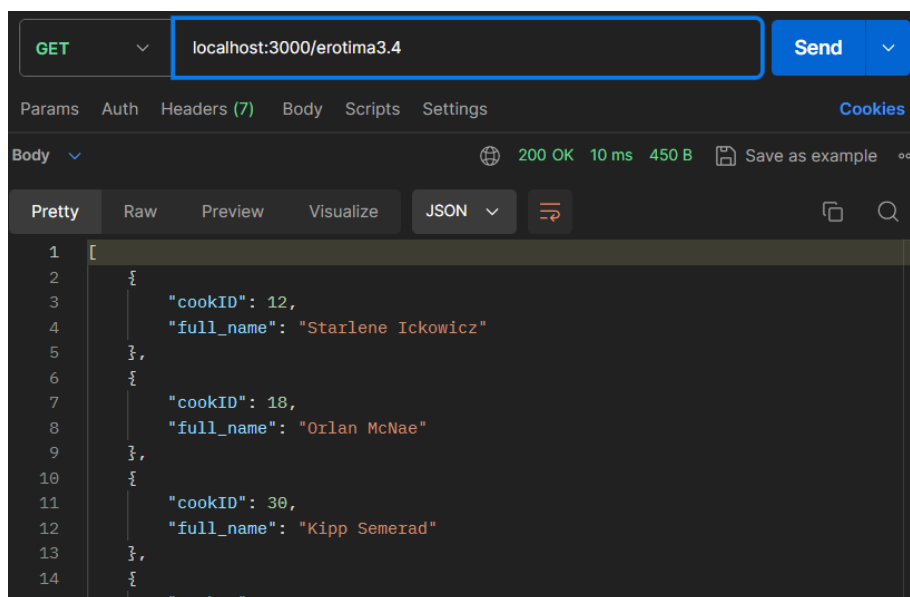
3.3. Βρείτε τους νέους μάγειρες (ηλικία < 30 ετών) που έχουν τις περισσότερες συνταγές.

```
SELECT chn.cookID, CONCAT(first_name, ' ', last_name) AS fullName, chn.ncID,  
nc.description, r.name  
FROM cook c  
JOIN cookHasNational chn ON c.cookID = chn.cookID  
JOIN national_cuisine nc ON chn.ncID = nc.ncID  
JOIN recipe r ON nc.ncID = r.ncID  
WHERE age<30  
ORDER BY cookID ;
```



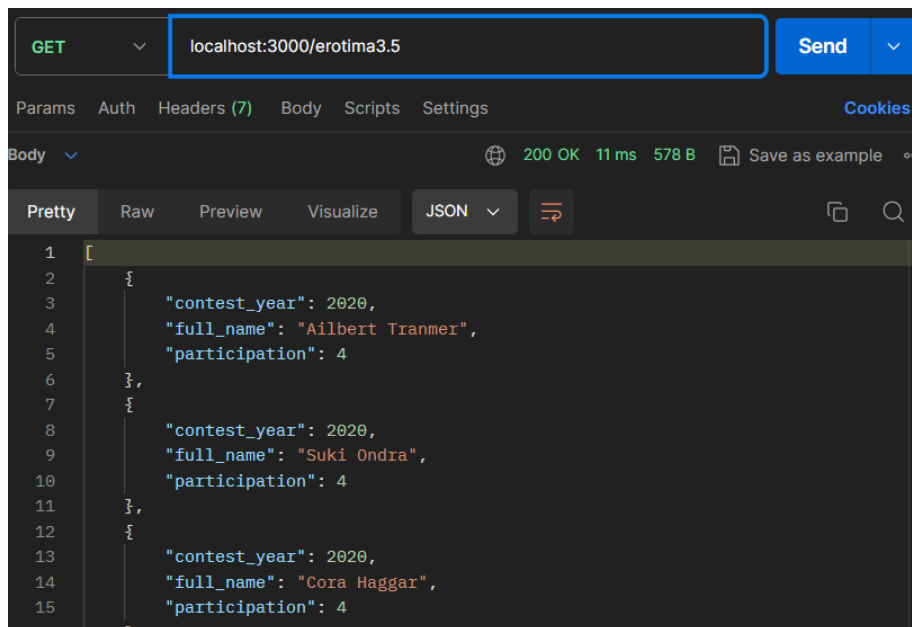
3.4. Βρείτε τους μάγειρες που δεν έχουν συμμετάσχει ποτέ σε ως κριτές σε κάποιο επεισόδιο.

```
SELECT cookID, CONCAT(first_name, ' ', last_name) AS full_name  
FROM cook c  
WHERE c.cookID NOT IN  
(SELECT cookID FROM judge j);
```



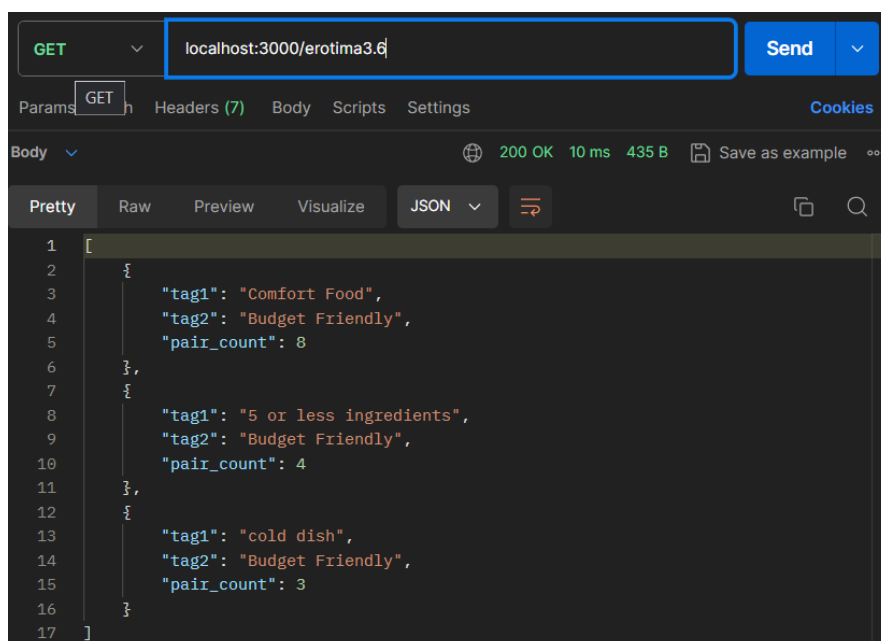
3.5. Ποιοι κριτές έχουν συμμετάσχει στον ίδιο αριθμό επεισοδίων σε διάστημα ενός έτους με περισσότερες από 3 εμφανίσεις;

```
SELECT co.contest_year, mt.full_name, co.participation
FROM countOfmoreThan3 co
JOIN moreThan3 mt ON co.participation = mt.participation
WHERE co.zeygoi >1;
```



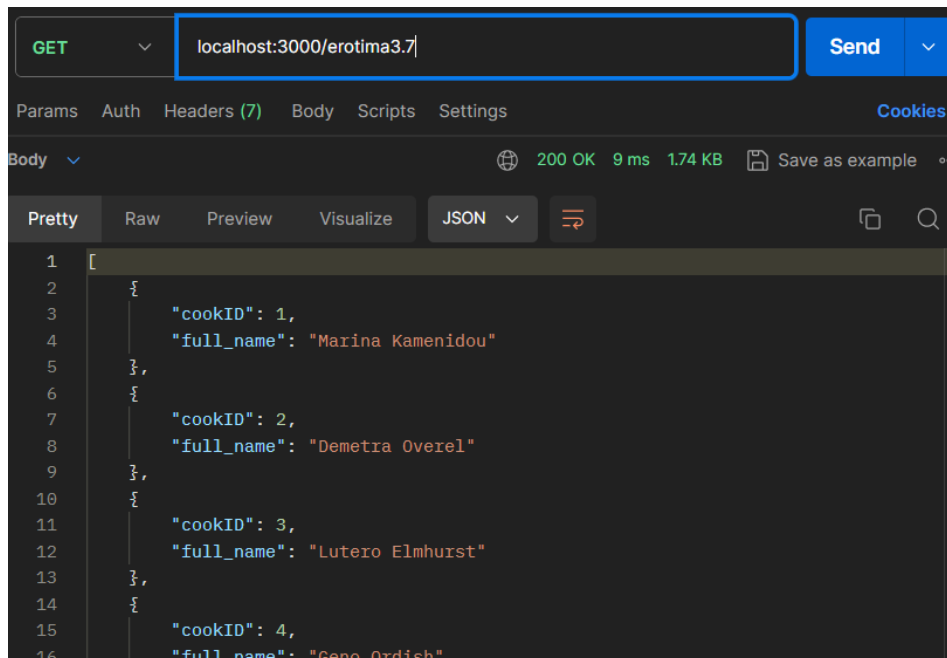
3.6. Πολλές συνταγές καλύπτουν περισσότερες από μια ετικέτες. Ανάμεσα σε ζεύγη πεδίων (π.χ. brunch και κρύο πιάτο) που είναι κοινά στις συνταγές, βρείτε τα 3 κορυφαία (top-3) ζεύγη που εμφανίστηκαν σε επεισόδια.

```
SELECT tag1 , tag2, pair_count
FROM count_of_tagPairs
LIMIT 3;
```



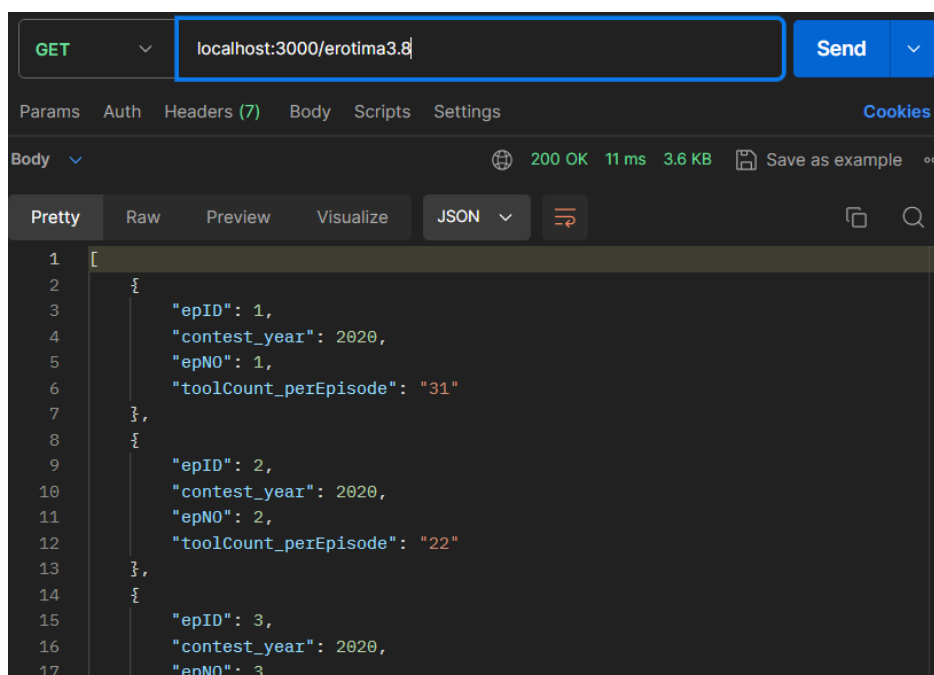
3.7. Βρείτε όλους τους μάγειρες που συμμετείχαν τουλάχιστον 5 λιγότερες φορές από τον μάγειρα

```
SELECT sc.cookID , CONCAT(first_name, ' ' , last_name) AS full_name
FROM summetoxi_cook sc
JOIN cook c ON sc.cookID = c.cookID
WHERE summetoxi < ( SELECT MAX(summetoxi) FROM summetoxi_cook sc) - 5 ;
```



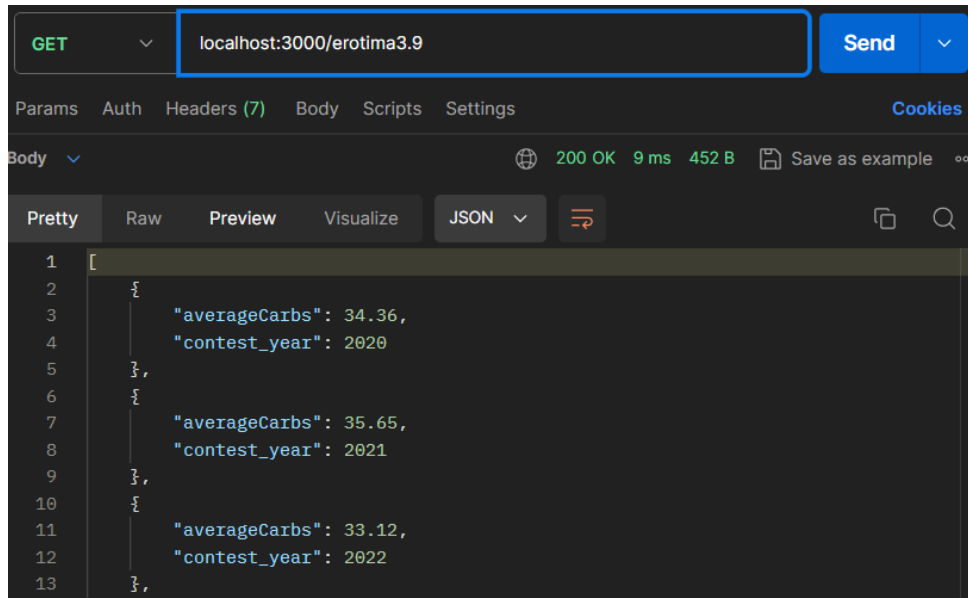
3.8. Σε ποιο επεισόδιο χρησιμοποιήθηκαν τα περισσότερα εξαρτήματα (εξοπλισμός);

```
SELECT emr.epID, emr.contest_year, emr.epNo,
       SUM(numberOfTools_perRecipe) AS toolCount_perEpisode
FROM toolCount_perRecipe tcpr
JOIN episode_match_recipe emr ON tcpr.recipeId= emr.recipeID
GROUP BY (epID);
```



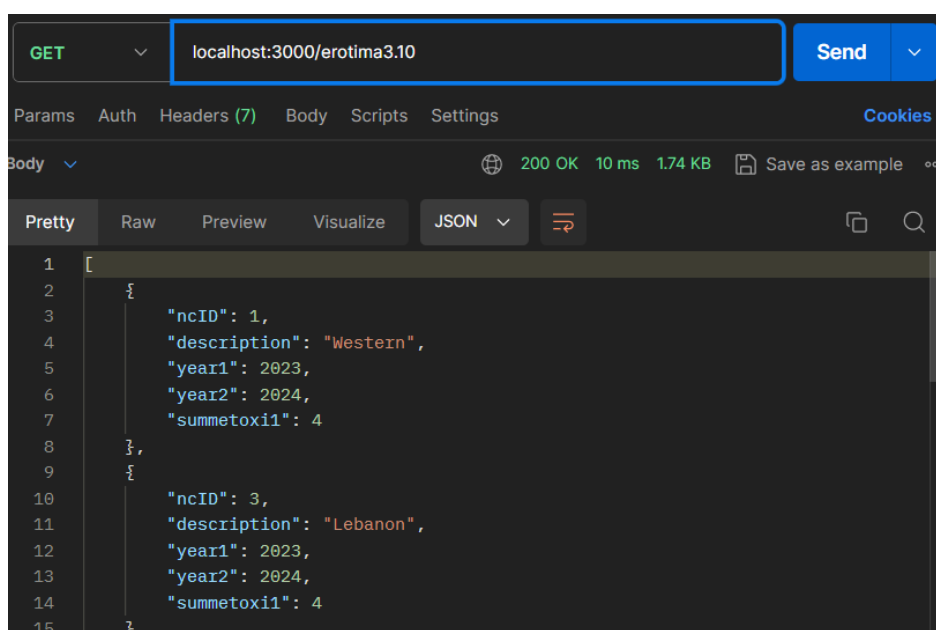
3.9. Λίστα με μέσο όρο αριθμού γραμμάρων υδατανθράκων στο διαγωνισμό ανά έτος.

```
FROM nutrition n
JOIN recipe r ON n.recipeID = r.recipeID
JOIN epHasCookNatRec ehcnr ON r.recipeID = ehcnr.recipeID
JOIN episode e ON ehcnr.epID = e.epID
JOIN contest c ON e.contestID = c.contestID
GROUP BY c.contestID ;
```



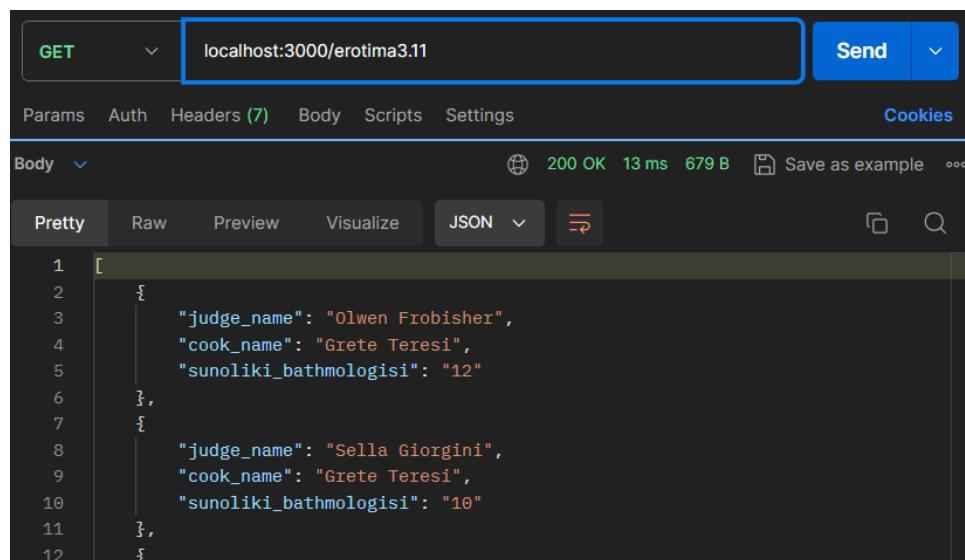
3.10. Ποιες Εθνικές κουζίνες έχουν τον ίδιο αριθμό συμμετοχών σε διαγωνισμούς, σε διάστημα δύο συνεχόμενων ετών, με τουλάχιστον 3 συμμετοχές ετησίως;

```
SELECT nc.ncID, nc.description, year1, year2, summetoxi1
FROM TwoYearParticipation typ JOIN national_cuisine nc ON typ.ncID = nc.ncID
WHERE summetoxi1 = summetoxi2
ORDER BY ncID, year1;
```



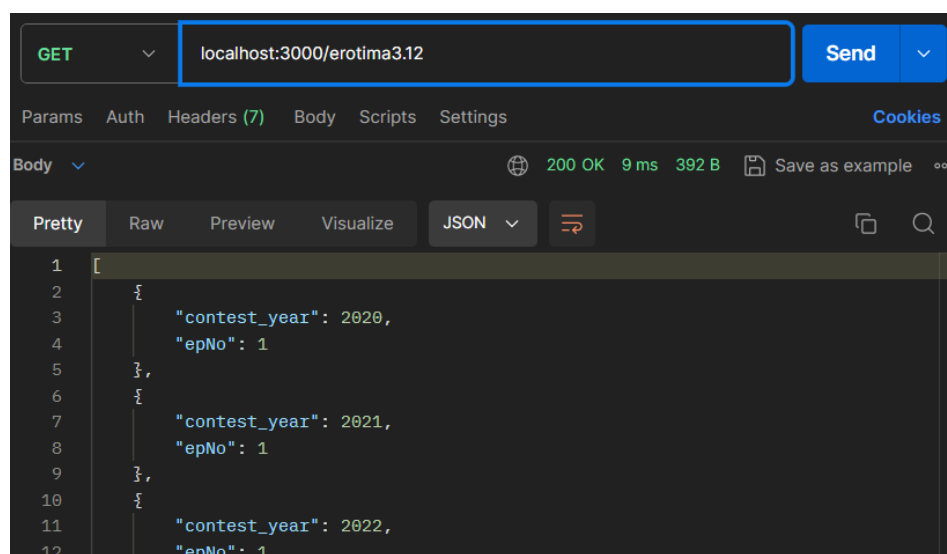
3.11. Βρείτε τους top-5 κριτές που έχουν δώσει συνολικά την υψηλότερη βαθμολόγηση σε ένα μάγειρα. (όνομα κριτή, όνομα μάγειρα και συνολικό σκορ βαθμολόγησης)

```
SELECT judge_name, cook_name, sunoliki_bathmologisi
FROM sum_of_scores_perJudge_andCook
WHERE cookID = 10
ORDER BY sunoliki_bathmologisi DESC
LIMIT 5;
```



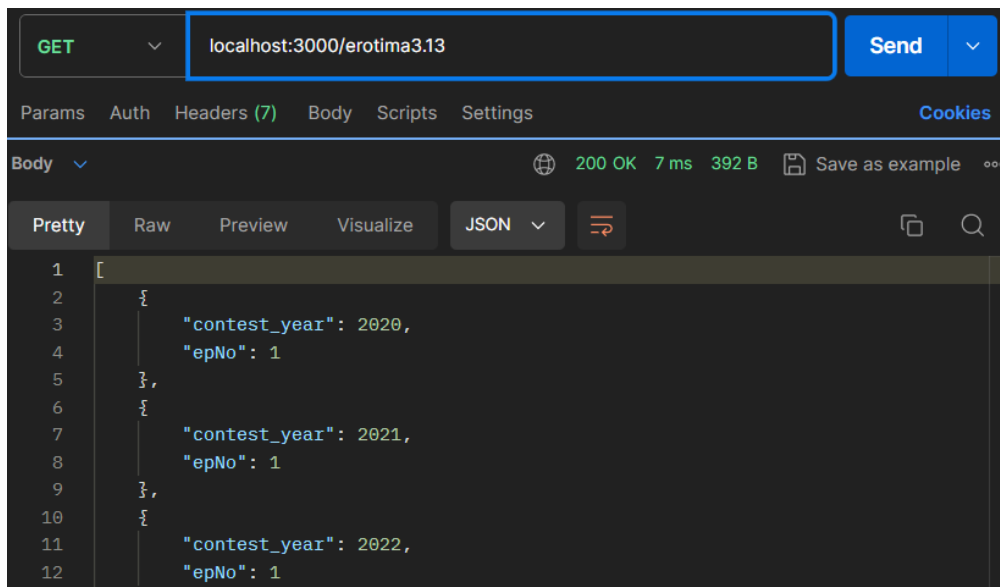
3.12. Ποιο ήταν το πιο τεχνικά δύσκολο, από πλευράς συνταγών, επεισόδιο του διαγωνισμού ανά έτος;

```
SELECT contest_year, epNo
FROM (SELECT contest_year, epNo, MAX (episodesDifficulty)
      FROM erotima_3_12 e312
      GROUP BY contest_year) as subquery;
```



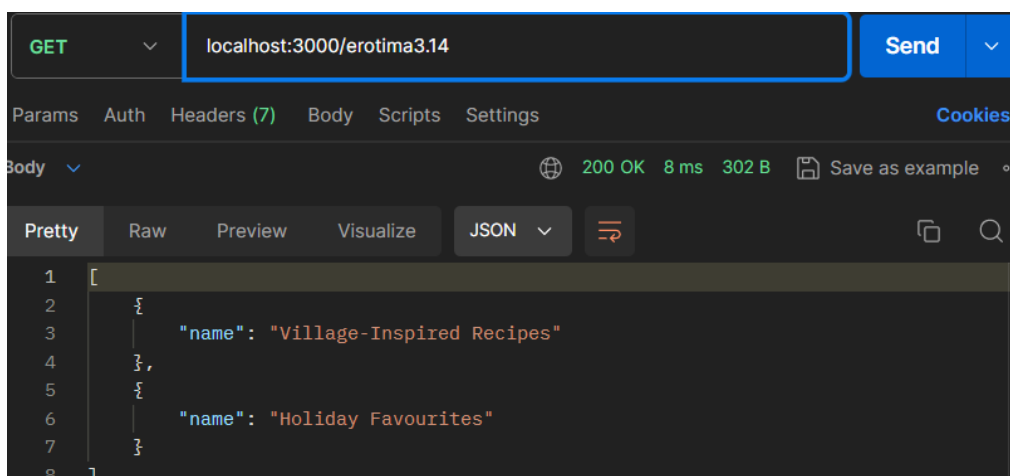
3.13. Ποιο επεισόδιο συγκέντρωσε τον χαμηλότερο βαθμό επαγγελματικής κατάρτισης (κριτές και μάγειρες);

```
SELECT contest_year, epNo
FROM (SELECT contest_year, epNo, MIN (profDegree)
      FROM erotima_3_13
      GROUP BY contest_year) as subquery;
```



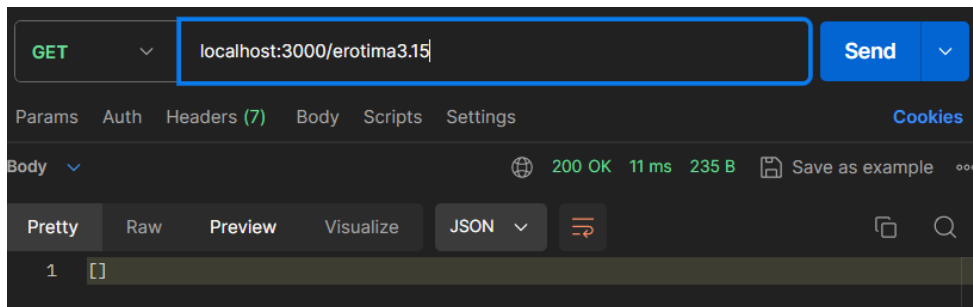
3.14. Ποια θεματική ενότητα έχει εμφανιστεί τις περισσότερες φορές στο διαγωνισμό;

```
SELECT name
FROM erotima_3_14
WHERE timesUsed =
      (SELECT MAX(timesUsed)
       FROM erotima_3_14);
```



3.15. Ποιες ομάδες τροφίμων δεν έχουν εμφανιστεί ποτέ στον διαγωνισμό;

```
SELECT fg.name
FROM food_group fg
WHERE food_groupID NOT IN
(SELECT fg.food_groupID
FROM food_group fg
JOIN ingredient i ON fg.food_groupID = i.food_groupID
JOIN recipeUseIngr rui ON i.ingredientID = rui.ingredientID
GROUP BY fg.name);
```



3. Οδηγίες εγκατάσταση της εφαρμογής

Προαπαιτούμενα:

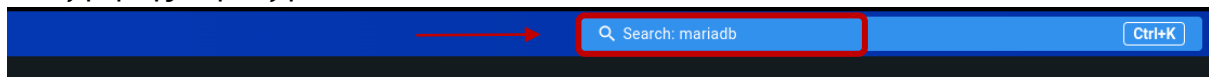
* Docker Desktop, αν δεν το έχεις εγκατεστημένο μπορείς να το εγκαταστήσεις από εδώ <https://docs.docker.com/get-docker/>

* Node.JS, αν δεν το έχεις εγκατεστημένο μπορείς να το εγκαταστήσεις από εδώ <https://nodejs.org/en/download>

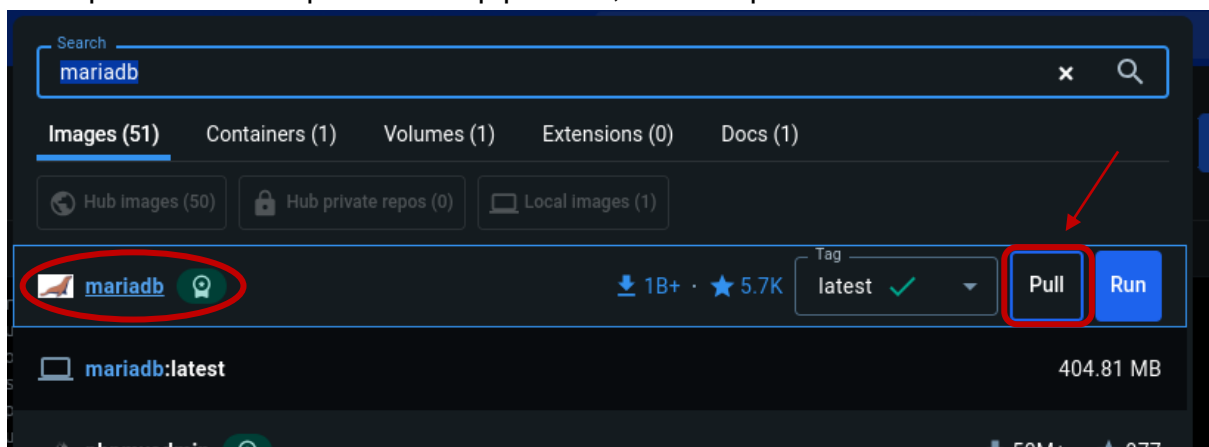
* DBeaver, αν δεν το έχεις εγκατεστημένο μπορείς να το εγκαταστήσεις από εδώ <https://dbeaver.io/download/>
(ή κάποιο άλλο DBMS Graphical Client)

A. Εγκατέστησε την βάση μέσω Docker

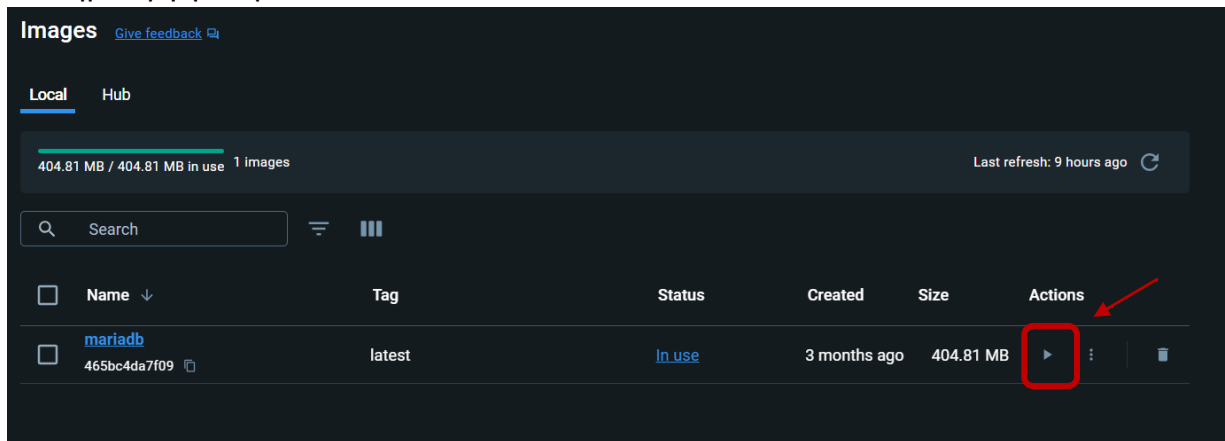
1. Κάνε pull την Mariadb image, μέσω του Docker Desktop. Πληκτρολόγησε στην μπάρα αναζήτησης, τη λέξη **mariadb**.



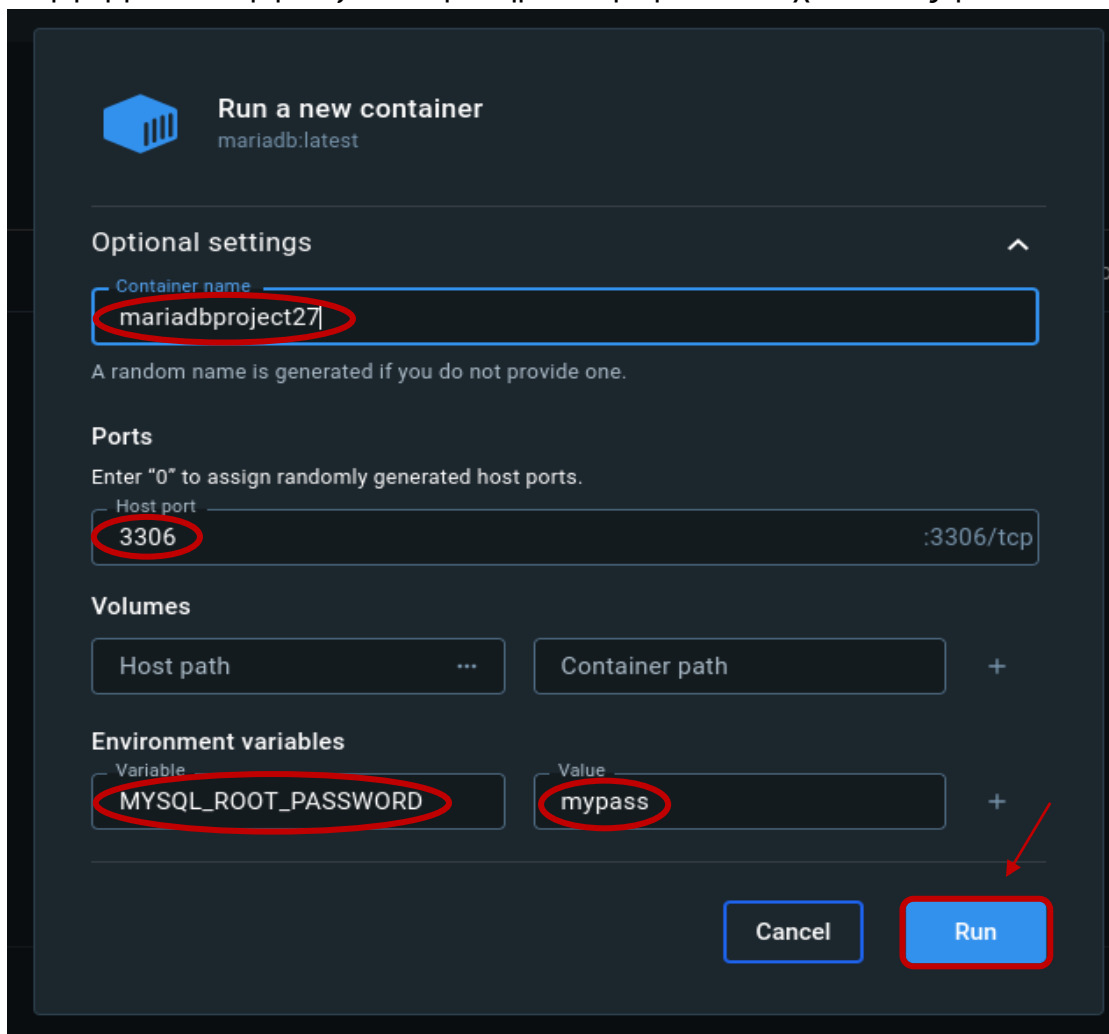
Στο πρώτο αποτέλεσμα που θα εμφανιστεί, πάτα το pull.



2. Στη συνέχεια, μεταφερόμαστε στα images. Πατάμε στο mariadb image το βελάκι (run) για να δημιουργήσουμε το container.



Στη φόρμα που εμφανίζεται συμπληρώνουμε με τα στοιχεία όπως φαίνονται παρακάτω.

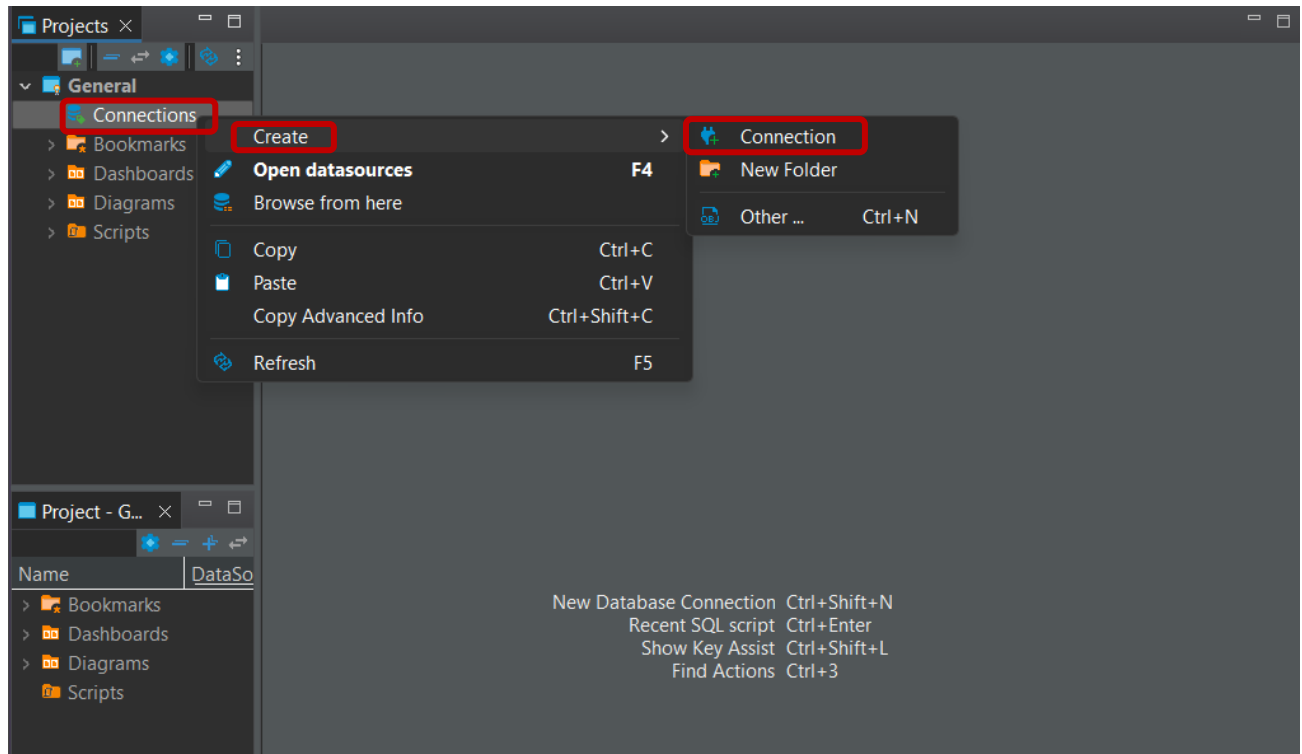


3. Η βάση είναι έτοιμη για χρήση.

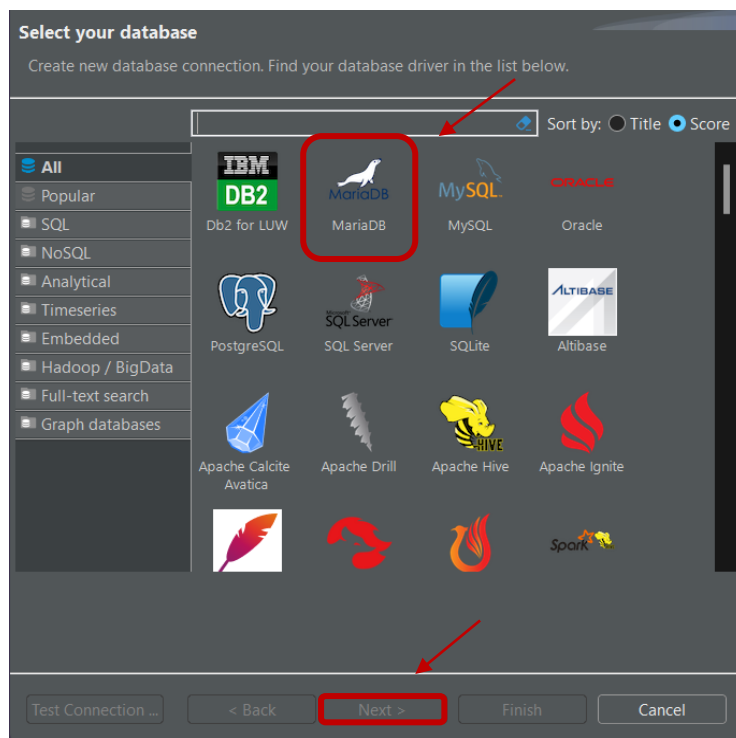


B. Εισαγωγή DDL αρχείου (DDLscript.sql)

1. Άνοιξε το DBeaver. Με δεξί κλικ στο Connections, επέλεξε Create και στη συνέχεια πάλι Connection.



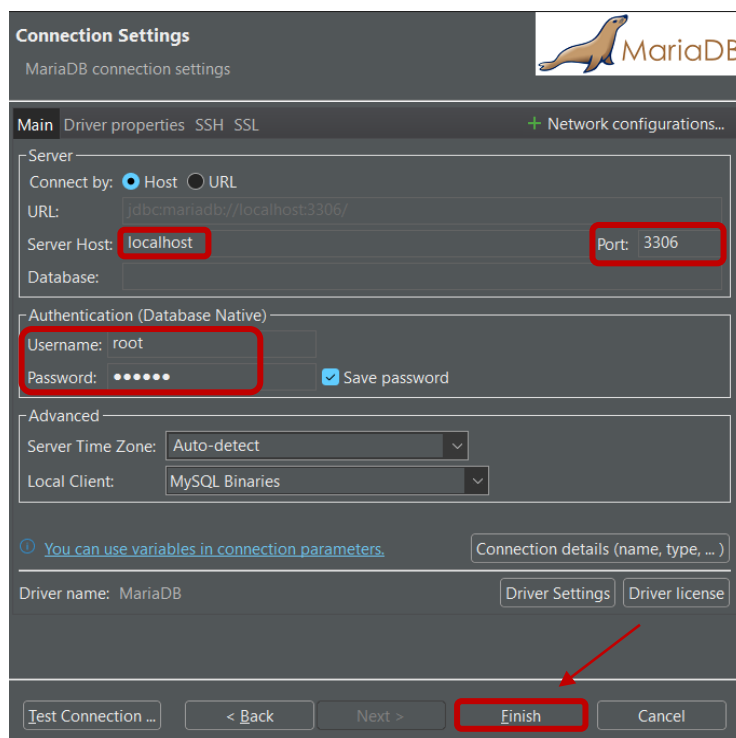
2. Επέλεξε την βάση MariaDB και πάτησε Next.



3. Συνδεθείτε στη βάση με τα παρακάτω στοιχεία

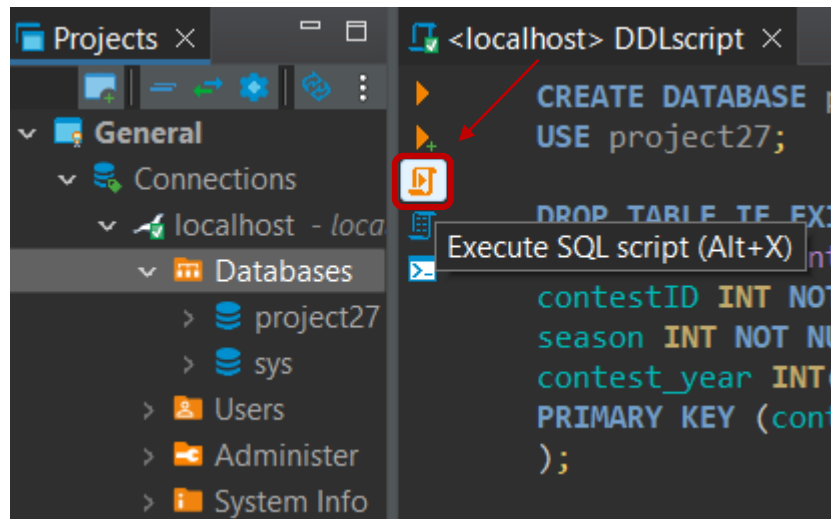
server host: localhost
port: 3306
username: root
password: mypass

Κάνε κλικ στο Finish.



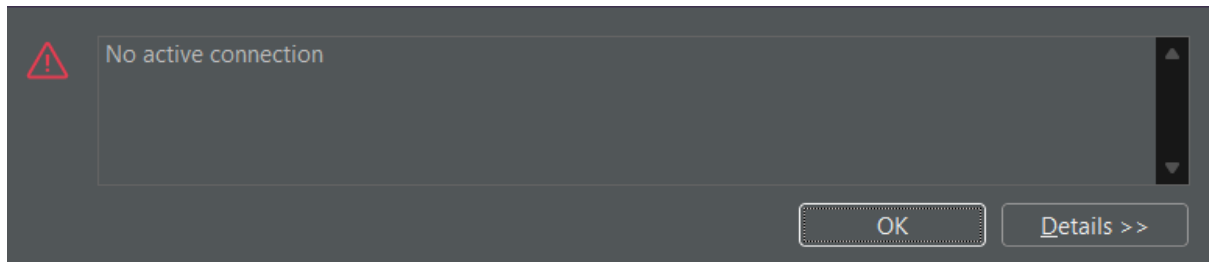
4. Χρησιμοποίησε το URL του git repo της εργασίας:
<https://github.com/marinanthi/Project27> , για να εισάγεις το DDL αρχείο
(DDLscript.sql).

- Αφού αντιγράψεις και επικολλήσεις το script, κάνε κλικ στο εικονίδιο Execute SQL Script.

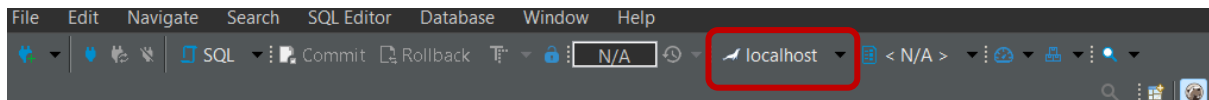


- Κάνοντας Refresh το πεδίο Database θα πρέπει να εμφανίζεται η βάση project 27, όπως παραπάνω.

6. Αν λαμβάνεις αυτό το μήνυμα κατά την υλοποίηση του script, βεβαιώσου πως είσαι συνδεδεμένος στο Connection που έφτιαξες.



Κάνε διπλό κλικ στο πεδίο localhost.



Επέλεξε το Connection localhost και στη συνέχεια κανε κλικ στο Select.



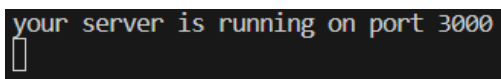
Γ. Εισαγωγή του DML αρχείου

Αφού έχετε συνδεθεί στη βάση με τα στοιχεία του προηγούμενου βήματος, τρέξε το DML script (DMLscript.sql) για να εισάγετε τα δεδομένα, με τον ίδιο τρόπο.

Δ. Τρέξε την εφαρμογή κάνοντας χρήση κάποιου Command Prompt.

1. Πληκτρολόγησε `npm install` για να κατεβάσεις τα απαραίτητα packages.
2. Πληκτρολόγησε `node index.js` για να τρέξεις την εφαρμογή.

Αν έχεις ακολουθήσει σωστά τις οδηγίες στην οθόνη σου θα εμφανιστεί το εξής:



```
your server is running on port 3000
█
```

3. Για να σταματήσεις το πρόγραμμα πάτησε (Ctrl + C).