ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ

Στέλιος Κάτσης - 03120139

Γιάννης Μερτζιώτης - 03120016

Γιώργος Σερετάκος - 03120084

Σύνδεσμος για το gitrepo: https://github.com/GeorgeSeretakos/DB-Project

Υποθέσεις - Παραδοχές

Για τα queries, τα οποία μπορεί να εκτελέσει κάποιος χρήστης, είτε αυτός είναι chef είτε admininstrator, μέσω της αντίστοιχης ιστοσελίδας, έχουμε κάνει τις ακόλουθες παραδοχές:

1. Οι συνταγές στις οποίες έχει πρόσβαση ένας Chef είναι δύο ειδών:
   - Συνταγές που έχουν προστεθεί από αυτόν
   - Συνταγές που του έχουν ανατεθεί σε κάποιον διαγωνισμό

2. Για τη χρήση των queries, προσθέτουμε μια επιπρόσθετη παράμετρο p_ID_User, η οποία έχει σχέση με τον χρήστη και τον επαληθεύει. Ειδικότερα, εάν ο χρήστης είναι ο Administrator, τότε έχει πρόσβαση σε όλες τις συνταγές και χρήστες και μπορεί να διαγράφει και να τροποποιεί κάθε συνταγή και χρήστη. Αντίθετα, εάν ο χρήστης είναι ένας Chef, μπορεί να έχει πρόσβαση, να διαγράφει και να τροποποιεί μόνο τα δικά του στοιχεία και τις συνταγές στις οποίες έχει ο ίδιος πρόσβαση.

3. Όσον αφορά τα queries, τα οποία αφορούν συνταγές, απαιτείται το id του user και αν ο user αυτός έχει πρόσβαση στη συγκεκριμένη συνταγή, μπορεί να εκτελέσει το query

4. Όσον αφορά τα queries, τα οποία αφορούν μάγειρες, απαιτείται το id του user όσο και του chef και αν το chef id επαληθεύεται μέσω του user, μπορεί να εκτελέσει το query

5. Όσον αφορά τις σχέσεις μεταξύ tables, όπως tags, meals κλπ, τα στοιχεία δεν μπορούν να τροποποιηθούν, αλλά μόνο να εισαχθούν νέα και να διαγραφούν τα παλιά. Για τροποποίηση μίας σχέσης, μπορεί να διαγραφεί και να προστεθεί εκ νέου.

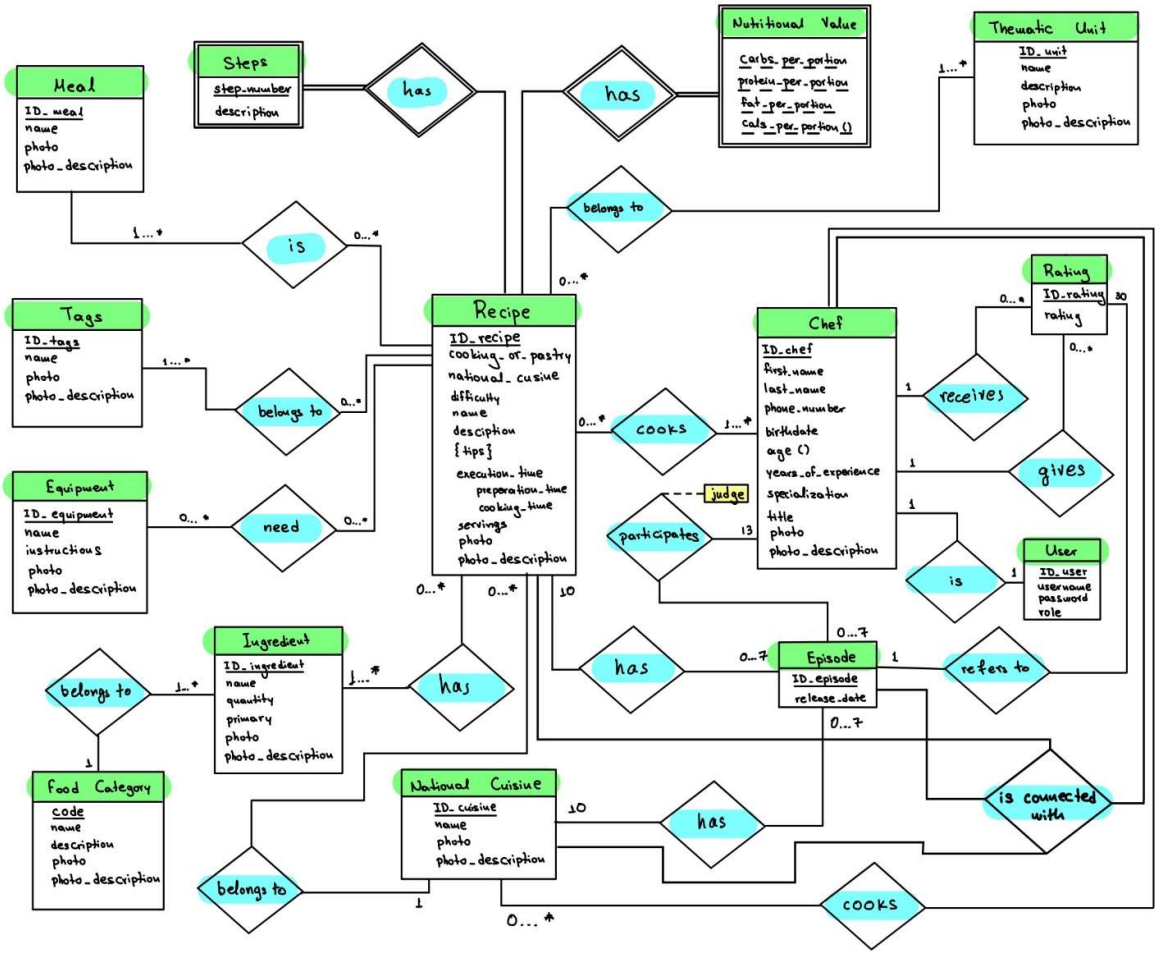
6. Όσον αφορά τα steps, μπορούν να εισαχθούν και να τροποποιηθούν μεμονωμένα βήματα, αλλά μόνο να διαγραφούν όλα τα βήματα για μια συνταγή, καθώς διαφορετικά θα μπορεί να υπήρχαν κενά στα βήματα (1, 2, 4, ...)

7. Ένας chef δεν μπορεί να διαγράψει μία συνταγή, ακόμα και αν έχει πρόσβαση σε αυτή, καθώς η συνταγή μπορεί να συνδέεται και με άλλες έννοιες στη βάση (με επεισόδια, άλλους chef κλπ), οπότε σβήνοντάς την για αυτόν θα διαγραφόταν και από τους άλλους. Μπορεί μόνο να διαγράψει, να τροποποιήσει ή να προσθέσει άλλα στοιχεία στη συνταγή.

8. Η εκχώρηση ενός νέου χρήστη στο σύστημα, είτε αυτός είναι admin είτε chef, γίνεται μέσω του administrator. Επίσης κάθε νέος χρήστης είναι αρχικά chef και μπορεί μετά ο administrator να τον αναβαθμίσει σε admin μέσω της διαδικασίας UpdateUserStatus

9. Η μόνη δυνατότητα που υπάρχει για την τροποποίηση των επεισοδίων είναι η δυνατότητα διαγραφής ενός επεισοδίου και μόνο από τον administrator, και αυτό γιατί η δημιουργία των επεισοδίων γίνεται αυτόματα μέσω του back-end, οπότε η οποιαδήποτε προσπάθεια εισαγωγής, τροποποίησης ή διαγραφής μεμονωμένων επεισοδίων μπορεί να οδηγούσε σε λειτουργικά σφάλματα. Η διαγραφή ενός επεισοδίου επιτρέπεται μόνο, γιατί έτσι διαγράφονται μαζί και οποιεσδήποτε άλλες πληροφορίες σχετικές με το συγκεκριμένο επεισόδιο, δηλαδή διασφαλίζεται κατά μία έννοια το η συνέπεια της βάσης.

E-R DIAGRAM



Αναλύουμε τα relationships ανάμεσα στα tables ως εξής:

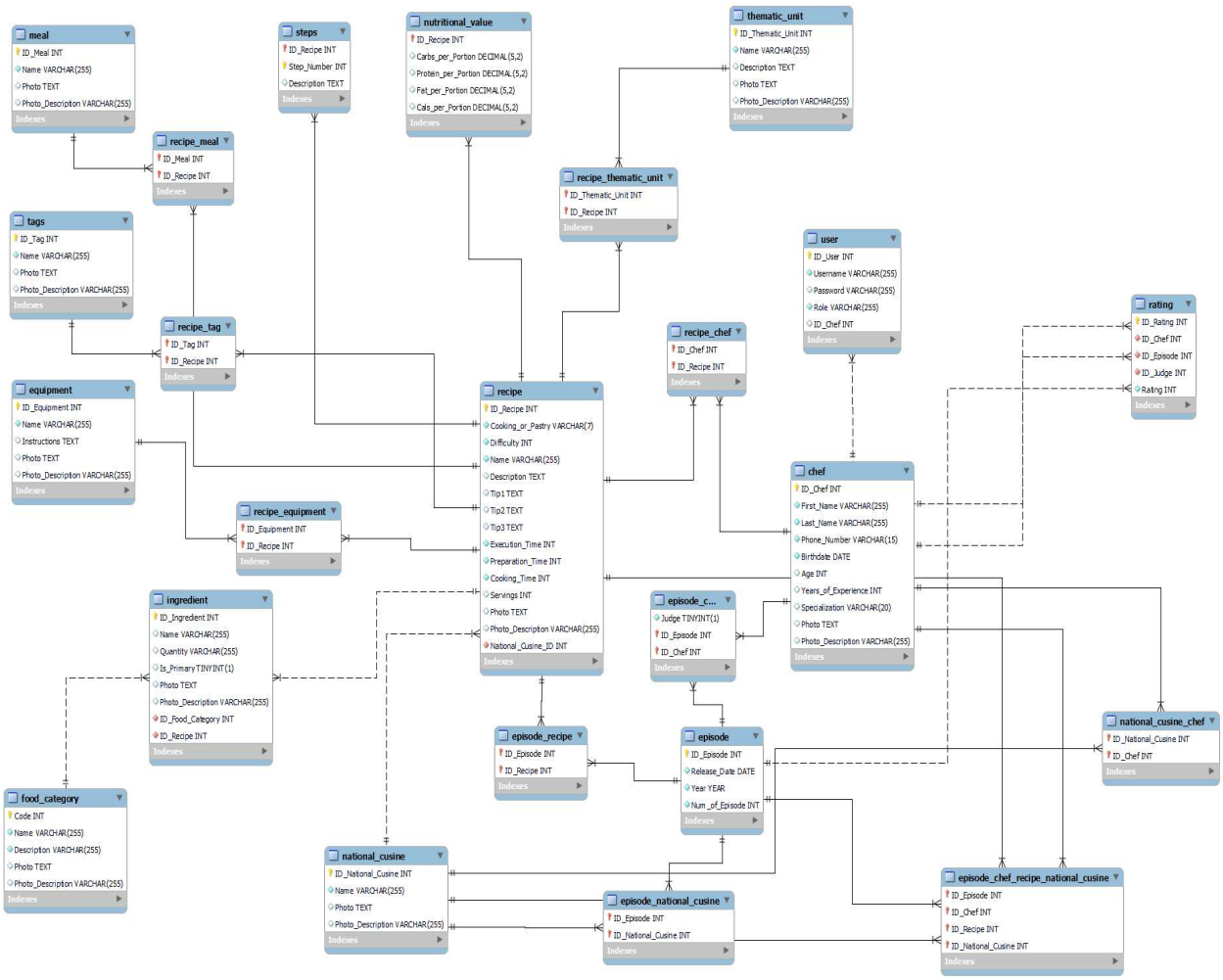
1. Meal - Recipe:
    - Relationship: One-to-Many
    - Επεξήγηση: Κάθε είδος γεύματος σχετίζεται με πολλαπλές συνταγές , κάθε συνταγή μπορεί να ανήκει μόνο σε ένα είδος γεύματος.
2. Recipe - Steps:
    - Relationship: One-to-Many
    - Επεξήγηση: Κάθε συνταγή έχει ένα η παραπάνω βήματα. Κάθε βήμα ανήκει σε μια συνταγή .
3. Recipe - Nutritional Value:
    - Relationship: One-to-One
    - Επεξήγηση: Κάθε συνταγή έχει μια συγκεκριμένη θρεπτική αξία, και κάθε θρεπτική αξία ανήκει σε μια συνταγή.

4. Recipe - Thematic Unit:
    - Relationship: Many-to-One
    - Επεξήγηση: Κάθε συνταγή ανήκει σε μια θεματική μονάδα, και κάθε θεματική μονάδα μπορεί να περιέχει πολλές συνταγές.
5. Recipe - Tags:
    - Relationship: Many-to-Many
    - Επεξήγηση: Κάθε συνταγή μπορεί να έχει πολλές ετικέτες και κάθε ετικέτα μπορεί να ανήκει σε πολλές συνταγές.
6. Recipe - Equipment:
    - Relationship: Many-to-Many
    - Επεξήγηση: Κάθε συνταγή μπορεί να χρειάζεται πολλαπλά είδη εξοπλισμού , και κάθε είδος εξοπλισμού μπορεί να χρειάζεται από πολλές συνταγές.
7. Recipe - Ingredient:
    - Relationship: Many-to-Many
    - Επεξήγηση: Κάθε συνταγή χρειάζεται πολλά υλικά , και κάθε υλικό μπορεί να ανήκει σε πολλές συνταγές.
8. Ingredient - Food Category:
    - Relationship: Many-to-One
    - Επεξήγηση: Κάθε υλικό ανήκει σε μια κατηγορία τροφίμων, και κάθε κατηγορία τροφίμων μπορεί να περιέχει πολλά υλικά .
9. Recipe - National Cuisine:
    - Relationship: Many-to-One
    - Επεξήγηση: Κάθε συνταγή σχετίζεται με μια εθνική κουζίνα και κάθε εθνική κουζίνα μπορεί να περιέχει πολλές συνταγές
10. Recipe - Episode:
    - Relationship: Many-to-Many
    - Επεξήγηση : Μία συνταγή μπορεί να βρίσκεται σε πολλαπλά επεισόδια , και κάθε επεισόδιο έχει 10 συνταγές.
11. Recipe – Chef:
    - Relationship: Many-to-Many
    - Επεξήγηση: Κάθε συνταγή μπορεί να μαγειρευτεί από πολλούς μάγειρες , και κάθε μάγειρας μπορεί να μαγειρέψει πολλές συνταγές.
12. Chef - Episode:
    - Relationship: Many-to-Many
    - Επεξήγηση: Κάθε μάγειρας μπορεί να συμμετέχει σε πολλαπλά επεισόδια , και κάθε επεισόδιο έχει 10 μάγειρες.
13. Chef – Rating(chef recieves):
    - Relationship: One-to-Many
    - Επεξήγηση: Κάθε μάγειρας μπορεί να λάβει πολλαπλές κριτικές , και κάθε κριτική αντιστοιχεί σε έναν μάγειρα.

- Αντίστοιχα συμβαίνει και όταν ένας σεφ είναι κριτής και βαθμολογεί όπως φαίνεται από το διάγραμμα.

14. User – Chef:
    - **Relationship**: One-to-One
    - **Επεξήγηση**:Κάθε χρήστης είναι μάγειρας , και κάθε μάγειρας είναι χρήστης.
15. Episode - National Cuisine:
    - **Relationship**: Many-to-Many
    - **Explanation**: Κάθε επεισόδιο έχει 10 εθνικές κουζίνες και κάθε εθνική κουζίνα μπορεί να συμμετέχει σε πολλαπλά επεισόδια.
16. Episode - Rating
    - **Relationship**: One-to-Many
    - Επεξήγηση: Κάθε βαθμολογία αντιστοιχεί σε ένα επεισόδιο. Κάθε επεισόδιο έχει 30 βαθμολογίες

RELATIONAL DIAGRAM



DDL Script (Tables, Constraints, Indices, Drops)

```
-- Primary Instructions
CREATE DATABASE IF NOT EXISTS MasterChef;
USE MasterChef;      -- Must be executed every time before using the db



-- Table creation
CREATE TABLE National_Cusine (
  ID_National_Cusine INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL UNIQUE,
  Photo TEXT,
  Photo_Description VARCHAR(255)
);

CREATE TABLE Recipe (
  ID_Recipe INT AUTO_INCREMENT PRIMARY KEY,
  Cooking_or_Pastry VARCHAR(7) NOT NULL CHECK (Cooking_or_Pastry IN ('Cooking', 'Pastry')),
  Difficulty INT NOT NULL CHECK (Difficulty BETWEEN 1 AND 5),
  Name VARCHAR(255) NOT NULL,
  Description TEXT,
  Tip1 TEXT,
  Tip2 TEXT,
  Tip3 TEXT,
  Execution_Time INT NOT NULL CHECK (Execution_Time > 0),
  Preparation_Time INT NOT NULL CHECK (Preparation_Time > 0),
  Cooking_Time INT NOT NULL CHECK (Cooking_Time > 0),
  Servings INT CHECK (Servings > 0),
  Photo TEXT,
  Photo_Description VARCHAR(255),

  National_Cusine_ID INT NOT NULL,
  FOREIGN KEY (National_Cusine_ID) REFERENCES National_Cusine(ID_National_Cusine) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE Meal (
```

```
  ID_Meal INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL UNIQUE,
  Photo TEXT,
  Photo_Description VARCHAR(255)
);

CREATE TABLE Recipe_Meal (
        ID_Meal INT NOT NULL,
  ID_Recipe INT NOT NULL,
  PRIMARY KEY (ID_Meal, ID_Recipe),
  FOREIGN KEY (ID_Meal) REFERENCES Meal(ID_Meal) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Tags (
  ID_Tag INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL UNIQUE,
  Photo TEXT,
  Photo_Description VARCHAR(255)
);

CREATE TABLE Recipe_Tag (
        ID_Tag INT NOT NULL,
  ID_Recipe INT NOT NULL,
  PRIMARY KEY (ID_Tag, ID_Recipe),
  FOREIGN KEY (ID_Tag) REFERENCES Tags(ID_Tag) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Equipment (
  ID_Equipment INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL UNIQUE,
  Instructions TEXT,
  Photo TEXT,
  Photo_Description VARCHAR(255)
);
```

```
CREATE TABLE Recipe_Equipment (
        ID_Equipment INT NOT NULL,
  ID_Recipe INT NOT NULL,
  PRIMARY KEY (ID_Equipment, ID_Recipe),
  FOREIGN KEY (ID_Equipment) REFERENCES Equipment(ID_Equipment) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Steps (
  ID_Recipe INT NOT NULL,
  Step_Number INT  CHECK (Step_Number > 0),
  Description TEXT,
  PRIMARY KEY (Step_Number, ID_Recipe),
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Nutritional_Value (
  ID_Recipe INT PRIMARY KEY,
  Carbs_per_Portion DECIMAL(5,2),
  Protein_per_Portion DECIMAL(5,2),
  Fat_per_Portion DECIMAL(5,2),
  Cals_per_Portion DECIMAL(5,2),
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Food_Category (
  Code INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL UNIQUE,
  Description VARCHAR(255) NOT NULL UNIQUE,
  Photo TEXT,
  Photo_Description VARCHAR(255)
);

CREATE TABLE Ingredient (
  ID_Ingredient INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255),
  Quantity VARCHAR(255),
```

```
  Is_Primary BOOL DEFAULT FALSE,
  Photo TEXT,
  Photo_Description VARCHAR(255),
  ID_Food_Category INT NOT NULL,
  ID_Recipe INT NOT NULL,
  FOREIGN KEY (ID_Food_Category) REFERENCES Food_Category(Code) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Thematic_Unit (
  ID_Thematic_Unit INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL UNIQUE,
  Description TEXT,
  Photo TEXT,
  Photo_Description VARCHAR(255)
);

CREATE TABLE Recipe_Thematic_Unit (
        ID_Thematic_Unit INT NOT NULL,
  ID_Recipe INT NOT NULL,
  PRIMARY KEY (ID_Thematic_Unit, ID_Recipe),
  FOREIGN KEY (ID_Thematic_Unit) REFERENCES Thematic_Unit(ID_Thematic_Unit) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Chef (
  ID_Chef INT AUTO_INCREMENT PRIMARY KEY,
  First_Name VARCHAR(255) NOT NULL,
  Last_Name VARCHAR(255) NOT NULL,
  Phone_Number VARCHAR(15) NOT NULL,
  Birthdate DATE NOT NULL,
  Age INT CHECK (Age > 0 AND Age <= 100),
  Years_of_Experience INT CHECK (Years_of_Experience >= 0),
  Specialization VARCHAR(20) CHECK (Specialization IN ('Third cook','Second cook', 'First cook','Assistant chef', 'Head chef')),
  Photo TEXT,
  Photo_Description VARCHAR(255)
);
```

```
CREATE TABLE Recipe_Chef (
        ID_Chef INT NOT NULL,
  ID_Recipe INT NOT NULL,
  PRIMARY KEY (ID_Chef, ID_Recipe),
  FOREIGN KEY (ID_Chef) REFERENCES Chef(ID_Chef) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE National_Cusine_Chef (
        ID_National_Cusine INT NOT NULL,
  ID_Chef INT NOT NULL,
  PRIMARY KEY (ID_National_Cusine, ID_Chef),
  FOREIGN KEY (ID_National_Cusine) REFERENCES National_Cusine(ID_National_Cusine) ON DELETE CASCADE ON UPDATE
CASCADE,
  FOREIGN KEY (ID_Chef) REFERENCES Chef(ID_Chef) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Episode (
  ID_Episode INT AUTO_INCREMENT PRIMARY KEY,
  Release_Date DATE NOT NULL,
  Year YEAR NOT NULL,
  Num_of_Episode INT NOT NULL
);

CREATE TABLE Episode_Recipe (
        ID_Episode INT NOT NULL,
  ID_Recipe INT NOT NULL,
  PRIMARY KEY (ID_Episode, ID_Recipe),
  FOREIGN KEY (ID_Episode) REFERENCES Episode(ID_Episode) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Episode_Chef (
        Judge BOOL NOT NULL,
        ID_Episode INT NOT NULL,
  ID_Chef INT NOT NULL,
  PRIMARY KEY (ID_Episode, ID_Chef),
```

```
  FOREIGN KEY (ID_Episode) REFERENCES Episode(ID_Episode) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Chef) REFERENCES Chef(ID_Chef) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Episode_National_Cusine (
        ID_Episode INT NOT NULL,
  ID_National_Cusine INT NOT NULL,
  PRIMARY KEY (ID_Episode, ID_National_Cusine),
  FOREIGN KEY (ID_Episode) REFERENCES Episode(ID_Episode) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_National_Cusine) REFERENCES National_Cusine(ID_National_Cusine) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE Rating (
  ID_Rating INT AUTO_INCREMENT PRIMARY KEY,
  ID_Chef INT NOT NULL,
  ID_Episode INT NOT NULL,
  ID_Judge INT NOT NULL,
  Rating INT NOT NULL CHECK (Rating BETWEEN 1 AND 5),
        FOREIGN KEY (ID_Episode) REFERENCES Episode(ID_Episode) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Chef) REFERENCES Chef(ID_Chef) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Judge) REFERENCES Chef(ID_Chef) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Episode_Chef_Recipe_National_Cusine (
        ID_Episode INT NOT NULL,
  ID_Chef INT NOT NULL,
  ID_Recipe INT NOT NULL,
  ID_National_Cusine INT NOT NULL,
  PRIMARY KEY (ID_Episode, ID_Chef, ID_Recipe, ID_National_Cusine),
  FOREIGN KEY (ID_Episode) REFERENCES Episode(ID_Episode) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Chef) REFERENCES Chef(ID_Chef) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_Recipe) REFERENCES Recipe(ID_Recipe) ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (ID_National_Cusine) REFERENCES National_Cusine(ID_National_Cusine) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE User (
```

```
  ID_User INT AUTO_INCREMENT PRIMARY KEY,
  Username VARCHAR(255) NOT NULL UNIQUE,
  Password VARCHAR(255),
  Role VARCHAR(255) NOT NULL CHECK (Role IN ('Admin', 'Chef')),
        ID_Chef INT,
  FOREIGN KEY (ID_Chef) REFERENCES Chef(ID_Chef) ON DELETE CASCADE ON UPDATE CASCADE
);



-- Index Creation
CREATE INDEX idx_name ON National_Cusine (Name);
CREATE INDEX idx_release_date ON Episode (Release_Date);
CREATE INDEX idx_ID_Chef ON Episode_chef(ID_Chef);
CREATE INDEX idx_Age ON Chef(Age);
CREATE INDEX idx_ID_Chef ON Recipe_Chef(ID_Chef);
CREATE INDEX idx_Name ON Tags(Name);
CREATE INDEX idx_ID_Recipe ON Nutritional_Value(ID_Recipe);
CREATE INDEX idx_ID_National_Cusine ON Episode_National_Cusine(ID_National_Cusine);
CREATE INDEX idx_ID_Judge ON Rating(ID_Judge);
CREATE INDEX idx_ID_Chef ON Rating(ID_Chef);
CREATE INDEX idx_Years_of_Experience ON Chef(Years_of_Experience);
CREATE INDEX idx_ID_Thematic_Unit ON Recipe_Thematic_Unit(ID_Thematic_Unit);
CREATE INDEX idx_ID_Food_Category ON Ingredient(ID_Food_Category);



-- Table deletion (must be deleted is the following order)
DROP TABLE IF EXISTS National_Cusine;
DROP TABLE IF EXISTS Recipe;
DROP TABLE IF EXISTS Tags;
DROP TABLE IF EXISTS Recipe_Tag;
DROP TABLE IF EXISTS Meal;
DROP TABLE IF EXISTS Recipe_Meal;
DROP TABLE IF EXISTS Equipment;
DROP TABLE IF EXISTS Recipe_Equipment;
DROP TABLE IF EXISTS Steps;
DROP TABLE IF EXISTS Nutritional_Value;
DROP TABLE IF EXISTS Ingredient;
```

```
DROP TABLE IF EXISTS Food_Category;

DROP TABLE IF EXISTS Thematic_Unit;

DROP TABLE IF EXISTS Recipe_Thematic_Unit;

DROP TABLE IF EXISTS Chef;

DROP TABLE IF EXISTS Recipe_Chef;

DROP TABLE IF EXISTS National_Cusine_Chef;

DROP TABLE IF EXISTS Episode;

DROP TABLE IF EXISTS Episode_National_Cusine;

DROP TABLE IF EXISTS Episode_Chef;

DROP TABLE IF EXISTS Episode_Recipe;

DROP TABLE IF EXISTS Rating;

DROP TABLE IF EXISTS Episode_Chef_Recipe_National_Cusine;

DROP TABLE IF EXISTS User;


-- Database deletion (this action cannot be undone)

DROP DATABASE masterchef;
```