Logo

Description automatically generated



**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ**

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

**Εξαμηνιαία Εργασία**

**Βάσεων Δεδομένων 6ου Εξαμήνου 2023-2024**

Ρέιντι Πασάι ΑΜ: 03121801

Νταβέας Στασινός ΑΜ: 03121076

Παπανδρικόπουλος Γρηγόρης Παναγιώτης ΑΜ: 03121136

Περιεχόμενα

**1 Βάση δεδομένων**

* 1. Διάγραμμα Οντοτήτων-Συσχετίσεων (Entity-Reletionship Diagram)
  2. Σχεσιακό σχήμα (Reletional Schema)
  3. Ευρετήρια (Indexes)
  4. DDL και DML script
  5. Τriggers, Views, Procedures and Autharazition

**2 Αναλυτικά βήματα εγκατάστασης της εφαρμογής**

**3 Σύνδεσμος στο github**

**4 Ανάλυση των queries**

1. **ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ**

1.1 Διάγραμμα Οντοτήτων-Συσχετίσεων (Entity-Reletionship Diagram)

A diagram of a computer

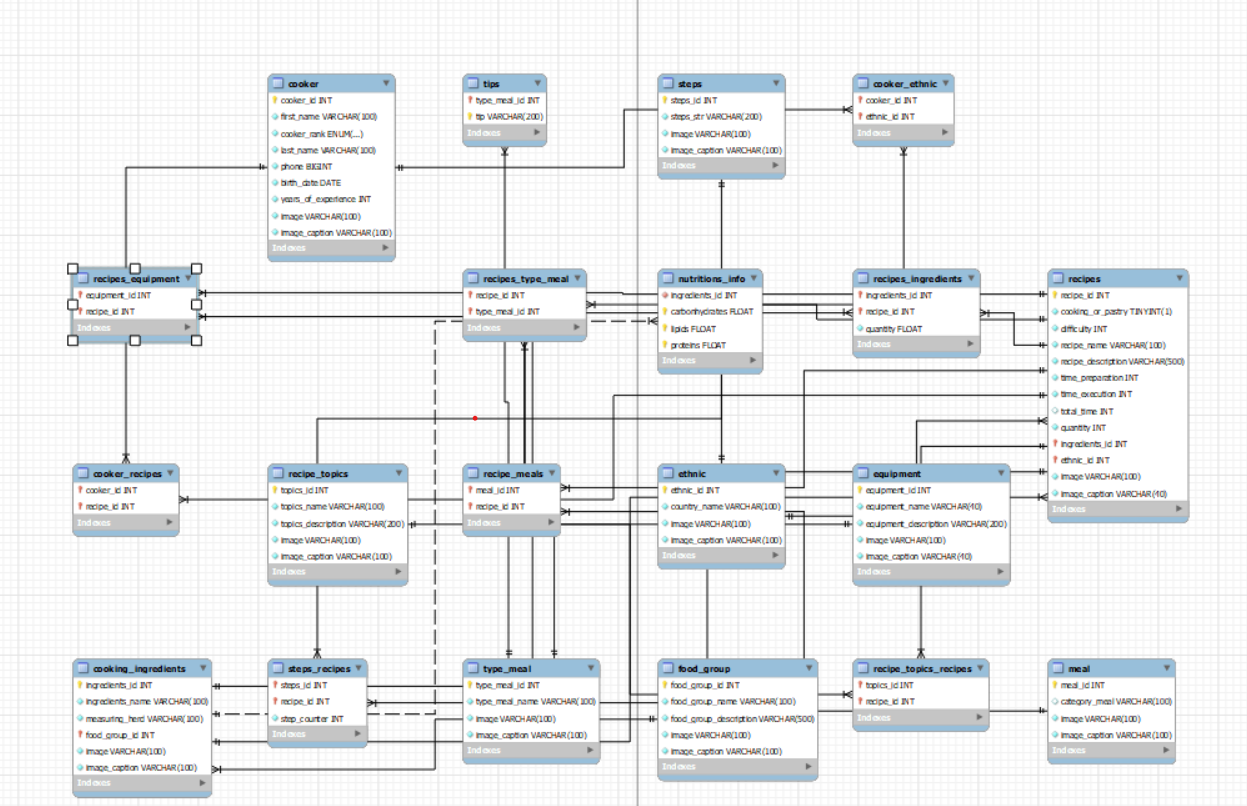
Description automatically generated with medium confidenceΑρχικά σχεδιάσαμε το διάγραμμα οντοτήτων συσχετίσεων το οποίο παραθέτουμε παρακάτω :

**(To παραπάνω διάγραμμα σχεδιάστηκε μέσω της εφαρμογής lucid.app)**

Αρχικά, υλοποιήσαμε το entity recipes το οποίο περιέχει τις συνταγές μαγειρικής. Ως primary key χρησιμοποιήσαμε το recipe\_id (για αυτό και το υπογραμμίζουμε) και τα εξής υπόλοιπα attributes: Το pastry\_or\_cooking το οποίο δηλώνει αν είναι συνταγή μαγειρικής ή ζαχαροπλαστικής, το difficulty το οποίο δηλώνει την δυσκολία εκτέλεσης της συνταγής το recipe\_name και το recipe\_discription που αποτελούν το όνομα και την περιγραφή περιγραφή της συνταγής αντίστοιχα. Το time\_preparation και το time\_execution είναι οι χρόνοι προετοιμασίας και εκτέλεσης της συνταγής ενώ το total\_time() είναι dirived attribute ως άθροισμα των άλλων δύο χρόνων .Τέλος το quantity είναι πόσες μερίδες παράγονται από κάθε συνταγή. Έπειτα δημιουργούμε το entity type\_meal το οποίο δείχνει σε ποια μορφή γεύματος ανήκει η συνταγή ενώ εμπεριέχει και μέχρι τρία tips για κάθε συνταγή (περικλείονται σε αγκύλες γιατί είναι multivalued data).Επίσης η σχέση που συνδέει το entity recipes με το entity type\_meal είναι μία σχέση many to many καθώς κάθε συνταγή αντιστοιχίζεται σε μία ή περισσότερες μορφές γεύματος και κάθε συνταγή αντιστοιχίζεται σε μία ή περισσότερες συνταγές. Επιπλέον το reletion recipes\_type\_meal με το recipes συνδέεται με διπλή γραμμή καθώς κάθε συνταγή πρέπει να ανήκει σε ένα τύπο γεύματος ενώ το relation με μονή καθώς μία μορφή γεύματος δεν είναι απαραίτητο να αντιστοιχίζεται σε κάποια συνταγή. Παρόμοια δημιουργούμε το entity meal το οποίο έχει ως attributes το primary key καθώς και σε ποια κατηγορία φαγητού ανήκει και εντελώς αντίστοιχα συνδέεται με το recipes με μία σχέση many to many με κάθε(για αυτό το λόγο η διπλή γραμμή) συνταγή να αντιστοιχίζεται τουλάχιστον σε ένα meal και ένα meal αντιστοιχίζεται τουλάχιστον σε μία καμία συνταγή. Στην συνέχεια ,δημιουργήσαμε το attribute equipment που περιέχει όπως και τα υπόλοιπα ένα id ως primary key ,όνομα και περιγραφή και συνδέεται όπως και τα υπόλοιπα με μία σχέση many to many. Παρόμοια, δημιουργούμε και τα βήματα steps με αντίστοιχο primary key και ένα attribute steps\_str που περίχει την περιγραφή του βήματος .Το relation που συνδέει συνταγές με βήματα εντελώς αντίστοιχα είναι many to many με τις αντίστοιχες γραμμές όπως και στις προηγούμενες σχέσεις ( μία διπλή γραμμή στη μεριά του reletion με το recipes γιατί όπως υποδεικνύει και η εκφώνηση της άσκησης κάθε συνταγή έχει τουλάχιστον 1 βήμα ).Μόνη διαφορά με τα προηγούμενα είναι ότι στο steps\_recipes με διακεκομμένες γραμμές βάζουμε ώς έξτρα attribute στο relation το step\_counter που καθορίζει πιο βήμα της συνταγής είναι (τα βήματα εξάλλου εκτελούνται σειριακά).Έπειτα, δημιουργούμε ένα πίνακα για τα υλικά cooking\_ingredients που περιλαμβάνει ένα id που είναι το primary key , το όνομα του συστατικού καθώς και την περιγραφή του. Συνδέεται με την σχέση recipes μέσω του reletion recipes\_ingredients πρόκειται για μία many to many relation καθώς κάθε συνταγή μπορεί να αντιστοιχίζεται με ένα ή και περισσότερα υλικά και κάθε υλικό με μία ή περισσότερες συνταγές. Επίσης, συνδέεται με διπλή γραμμή το relation με τα recipes γιατί κάθε γραμμή έχει τουλάχιστον ένα υλικό και το relation έχει ως επιπλέον attribute το quantity που είναι τα πόσα υλικά χρειάζονται για κάθε συνταγή (τα οποία θα υπολογιστούν δυναμικά).Ακόμη τα cooking\_ingredients συνδέονται με την σχέση basic\_ingredients σχέση many to one .Για κάθε συνταγή (διπλή γραμμή) αντιστοιχίζεται ακριβώς ένα βασικό υλικό. Επιπλέον, δημιουργούμε το entity food group το οποίο συνδέεται με τα cooking\_ingredients μέσω μίας σχέσης one to many (για αυτό το λόγο και το βελάκι στη μεριά του food\_group. Η σχέση many to one προγράφει ότι κάθε cooking\_ingedient(διπλή γραμμή) αντιστοιχίζεται σε μία ομάδα τροφίμων. Τέλος συνδέουμε τα cooking\_ingedients με το nutritions\_info. Πρόκειται για ένα weak entity καθώς δεν έχει primary key και εξαρτάται από το strong enitity nutritions\_info και χρησιμοποιεί ως identifier το primary key των cooking\_ingredients σε συνδυασμό με όλα τα στοιχεία των nutritions\_info που είναι τα discriminators και υπογραμμίζονται με διακεκομμένες γραμμές(είναι τα λιπίδια οι υδατάνθρακες και οι πρωτείνες) .Το relation που συνδέει τους δύο πίνακες είναι one to one (μία διατροφική πληροφορία αντιστοιχίζεται σε ένα υλικό και λόγω του ότι το nutrition\_info είναι weak τόσο το entity όσο και το relation απεικονίζονται με διπλή γραμμή. Επίσης οι συνταγές ομαδοποιούνται βάσει θεματικές ενότητες για αυτό συνδέονται με μία σχέση many to one με το entity ethnic το relation εξυπηρετεί την σχέση κάθε συνταγή ανήκει σε μία εθνική κουζίνα κάθε κουζίνα .Έπειτα, δημιουργούμε τους μάγειρες με attributes το id(primary key) το name που είναι composite attribute και παράγεται από τα simple attributes first\_name και last\_name .Συνδέεται με το recipes με μία σχέση many to many καθώς και με το ethnic με μία εξίσου σχέση many to many. Δημιουργούμε το attribute episode και μέσω του relation episode\_expansion το συνδέουμε με τα attributes recipes cooker και ethnic . Το episode\_expansion παίρνει ως extra attributes τα evaluation1 ,evaluation2 kαι evaluation3 τα οποία αντιστοιχίζονται στις βαθμολογήσεις που κάθε κριτής-μάγειρας θα βάλει σε κάθε διαγωνιζόμενο-μάγειρα.

Όσον αφορά τους περιορισμούς πλήθους για το episode\_expansion με το 13 έως 13 για τα επεισόδια, ισχύει καθώς ξέρουμε ότι κάθε επεισόδιο θα εμφανιστεί ακριβώς 13 φορές στον πίνακα αυτό (10 για τους μαγείρους και 3 για τους κριτές). Μετά, για τις συνταγές, τις κουζίνες και τους μαγείρους, ξέρουμε ότι δεν μπορούν να εμφανιστούν 3 συνεχόμενες φορές σε τρία συνεχόμενα επεισόδια, άρα προφανώς δεν γίνεται και να εμφανιστούν πάνω από 8 φορές.

1.2 Σχεσιακό Σχήμα (Relational Schema)



**(παραπάνω διάγραμμα δημιουργήθηκε από τo αρχείο ddl.sql και η απεικόνιση γίνεται χάρη στο mysql workbench)**

Περιγραφή του ddl

Γενικά, σε όλα τα primary keys γίνονται indexes by default από την βάση για αυτό έχουμε συμπεριλάβει στον κώδικα σε μορφή σχολίων την δημιουργία indexes για όλα τα primary keys. Για τα foreign keys πάλι δημιουργούνται by default από το μysql workbench indexes για αυτό και σε αυτά την δημιουργία indexes την έχουμε συμπεριλάβει σε μορφή σχολίων εκτός από κάποια τα οποία δεν γίνονται και τα κάνουμε εμείς.Τα υπόλοιπα indexes pου χρησιμοποιήσουμε για να αυξήσουμε την ταχύτητα των queries θα αναλυθούν εκτενώς σε άλλη παράγραφο.

Δημιουργία βάσης

Αρχικά, κάνουμε drop δηλαδή διαγράφουμε το schema σε περίπτωση που έχει δημιουργηθεί. Έπειτα, μέσω της create δημιουργούμε το σχήμα και το ονομάζουμε cooking\_contest\_ntua και στην συνέχεια μέσω του use θέτουμε default την βάση μας.

Δημιουργία πινάκων

Αρχικά δημιουργούμε το food\_group με το primary key να αυξάνεται ανα 1 μέσω της auto increment και να είναι μη προσημασμένος ακέραιος .Ολα τα υπόλοιπα στοιχεία είναι varchar και ΝΟΤ NULL ενώ στην image μέσω του check θέτουμε τον περιορισμό να ξεκινά με το https://% λόγω του ότι πρόκειται για λινκ σε εικόνα.

Έπειτα, δημιουργούμε το cooking\_ingredients primary key το ingredients\_id είναι unsigned not null και auto increment όπως και όλα τα primary keys που δημιουργούμε. Όλα τα υπόλοιπα attributes είναι varchar kαι not null και αντίστοιχα όπως όλες οι εικόνες ξεκινούν με το https://% .Επίσης, έχει foreign key το food\_group\_id δηλαδή το primary key του food\_group καθώς food\_group και cooking\_ingredients συνδέονται με μία σχέση many to one και το primary key του οne δηλαδή του food\_group γίνεται foreign key στο many Παράλληλα, σε περίπτωση που κάνουμε προσπάθεια ενημέρωσης του foreign key δεν έχω κανένα περιορισμό και σε περίπτωση διαγραφής του κλειδιού δεν το επιτρέπω. Επίσης, σαν primary key βάζουμε και το food\_group\_id γιατί πρόκειται για σύνδεση μεταξύ strong entities.

Δημιουργούμε, το table nutritions\_info και έχει ώς primary keys τα discriminators attributes carbonhydrates, lipids, proteins τα οποία τα ορίζω και ως primary keys και ως foreign key έχει το primary key των cooking\_ingredients το ingredients\_id το οποίο δεν το βάζουμε ώς primary key γιατί η σχέση που συνδεέι τα δύο tables είναι weak.

Επειτα, δημιουργούμε το type meal ένα type\_meal\_id είναι unsigned not null και auto increment όπως και όλα τα primary keys που δημιουργούμε όλα τα υπόλοιπα ναι είναι είτε varchar είτε int και όλα not null

Ομοίως και το table ethnic.

Το table recipes έχει primary key το recipe\_id INT UNSIGNED, AUTO\_INCREMENT, NOT NULL το cooking\_or\_pastry είναι boolean και not null τα υπόλοιπα είναι not null. Το total\_time είναι το derived ως άθροισμα των time\_preparation και των time\_execution .Επίσης για foreign key έχουμε το ethnic\_id και το ingredients\_id από το ethnic kαι cookιng\_ingedients αντίστοιχα καθώς ο πίνακας recipes συνδέεται με αυτοίς τους πίνακες με μία σχέση many to one(basic\_ingredient και recipe\_country αντίστοιχα) και το private key του οne γίνεται foreign key στον πίνακα many.

Δημιουργούμε, τον πίνακα recipes\_ingredients που είναι το relation που προκύπτει από την σύνδεση many to many των πινάκων cooking\_ingredients και recipes kαι έχει ως attributes ως foreign keys τα οποία γίνονται και primary key του επειδή πρόκειται για strong relation τα primary keys των δύο tables pου συνέει και ως extra attribute έχει το quantity το οποίο τσεκάρουμε να είναι και μεγαλύτερο του μηδενός

Ομοίως δημιουργούμε τον πίνακα recipes\_type\_meal pου είναι το relation pου συνδέει το table recipes με το table type\_meal kαι έχει για foreign keys τα primary keys των tables που ενώνει τα οποία τα θέτουμε και ως primary διότι πρόκειται για strong σύνδεση

Δημιουργούμε επίσης έναν πίνακα για το multivalued attribute tips mε attribute το tip το οποίο είναι και primary key του ενώ έχει και ως foreign key το primary του type\_meal καθώς είναι attribute αυτού του πίνακα

Έπειτα δημιουργούμε το table meal mε παρόμοιο primary key με τα υπόλοιπα και τα απραίτητα attributes που έχουμε ορίσει στο er diagram μας

Εν συνεχεία, δημιουργούμε τον πίνακα που αποτελεί το relation many to many των tables recipes kαι meals έχοντας ως attributes foreign keys τα primary keys των tables που συνδέει και γιατί προκειται για strong relation τα θέτουμε και ως private

Όμοια με τα προηγούμενα tables δημιουργούμε τον πίνακα steps.

Και όμοια δημιουργούμε το table recipes\_steps για να δημιουργήσουμε το relation many to many που ενώνει τους πίνακες steps kai recipes έχοντας ως attributes τα primary keys των σχέσεων που συνδέει τα οποία είναι foreign key στον πινακα και επειδή πρόκειται για strong relation τα κάνουμε να έιναι και private .Ως extra attributeορίζουμε και το step\_counter που αποτελεί πιο είναι το βήμα της κάθε συνταγης (πχ 2ο ή 3ο)

Όμοια με τα προηγούμενα tables δημιουργούμε τον πίνακα equipments και γιατί συνδέεται με το table recipes με μία many to many releationship ομοίως με τα παραπάνω δημιουργούμε ένα table recipes\_equipment που συνδέει αυτά τα δύο entitities

Παρόμοια με τα παραπάνω δημιουργούμε τον πίνακα recipe\_topics και γιατί συνδέεται με το table recipes με μία many to many releationship ομοίως με τα παραπάνω δημιουργούμε ένα table recipes\_topics\_recipes που συνδέει αυτά τα δύο entitities.

Δημιουργούμε τον πίνακα cooker .Τα attributes είναι παρόμοιων τύπων με τα παραπάνω απλά στο phone που έιναι οι αριθμοί τηλεφώνου των μαγείρων είναι τύπου bigint γιατί ο αριθμός τηλεφώνου είναι αρκετά μεγάλος και παίρνει τιμές για να είναι έγκυρος αριθμός μεταξύ του 6900000000 και του 6999999999.Η ημερομηνία γέννησης τους είναι τύπος data και έχουμε ορίσει να είναι τέτοιος ώστε η χρονολογία γεννησής του να είμαι μικρότερη του 2005 δηλαδή οι συμμετέχοντες να είναθ μεγαλύτεροι των 18 ετών. Ακόμη, έχουμε θέσει ως περιορισμό τα χρόνια εμπειρίας του κάθε μάγειρα μείον την ηλικία του να ξεπερνάνε τα 15 έτη καθώς η εργασία κάτων των 15 ετών απαγορεύεται.Ακόμη έχει και ως attribute ένα rank που μέσω enum απαριθμούνται όλα τα δυνατά rank του cooker.

Όπως και με τα παραπάνω όμοια tables τα tables cooker\_ethnic και cooker\_recipes συνδέουν την many to many σχέση που συνδέει τον cooker με το ethnic και τον cooker mε τα recipes αντίστοιχα .Οι δύο αυτοι πίνακες έχουν ως attributes foreign keys τα primary keys των tables που συνδέουν τα οποία τα θέτουν και primary key γιατί η σχέσεις που συνδέουν είναι strong

Δημιουργούμε έπειτα το table episode με περιορισμούς το episode\_id ναι είναι ανάμεσα σε 0 και 10 καθώς κάθε κύκλος έχει 10 σειρές και επειδη έχουμε πάρει ως παραδοχή ότι ο διαγωνισμός έχει διεξαχθεί για 3 συνεχομενες σεζον 2022,2023,2024 το seazon\_year παίρνει τιμές από 2022 έως 2024.

Ακόμα δημιουργούμε και το episode\_expansion το οποίο είναι ένα table στο οποίο συνδέουμε πολλά attributes πολλών tables μαζί .Γιατί συνδέει many to many relations (κάθε reletion είναι το episode mε έναν άλλο πίνακα) παίρνει ως attributes τα foreign keys των πινάκων αυτών και τα κάνει και primary keys γιατί οι πίνακες συνδέονται με strong relationships μεταξύ τους επίσης έχει ως εξτρά attributes τα eval1,eval2,eval3 που παίρνουν τιμές 1 έως 5 και είναι οι βαθμολογίες του κάθε κριτή για την προσπαθεια κάθε μάγειρα.

Indexes που προκύπτουν από τα queries

Προσθέσαμε ένα index στο attribute is\_judge καθώς παρατηρήσαμε ότι είναι αρκετά χρήσιμο να μπορούμε γρηγορότερα να διαχωρίζουμε τους κριτές από τους διαγωνιζόμενους σε διάφορα queries. Ακόμη, προσθέσαμε index στην ημερομηνία γέννησης των μαγείρων, καθώς χρειάζεται στο query 3.3 να φιλτράρουμε τους μάγειρες ηλικίας κάτω των 30, άρα θα ήταν χρήσιμο να υπάρχει ένα index για να γίνεται αυτή η διαδικασία γρηγορότερα. Τέλος, προσθέτουμε και ένα index στα ονόματα των εθνικοτήτων καθώς στο query 3.2 τα ταξινομούμε ως προς τα ονόματά τους για να είναι πιο κομψή η απάντηση.

1.5 Τriggers, Views, Procedures and Autharazition

**Triggers**

Τα triggers στην SQL ειδικά είδη αποθηκευμένων διαδικασιών που ενεργοποιούνται αυτόματα ως απόκριση σε συγκεκριμένα μεταβολές. Χρησιμοποιούνται για να καθορίσουν συγκεκριμένους κανόνες σύμφωνα με τους οποίους γίνονται ή δεν γίνονται δεκτές οι αλλαγές των δεδομένων (insertions, deletions, modifications) από τη βάση δεδομένων μας.

Στον κώδικά μας, συγκεκριμένα στο αρχείο [procedures.sql](https://github.com/stasinosntaveas/cooking-contest-ntua/blob/main/procedures.sql), έχουμε triggers της μορφής:

DELIMITER //

CREATE TRIGGER trigger\_name

BEFORE INSERT ON table

FOR EACH ROW

BEGIN

    (table operations)

    IF condition THEN

        SIGNAL SQLSTATE '45000’

        SET MESSAGE\_TEXT = '(error\_message)';

    END IF;

END//

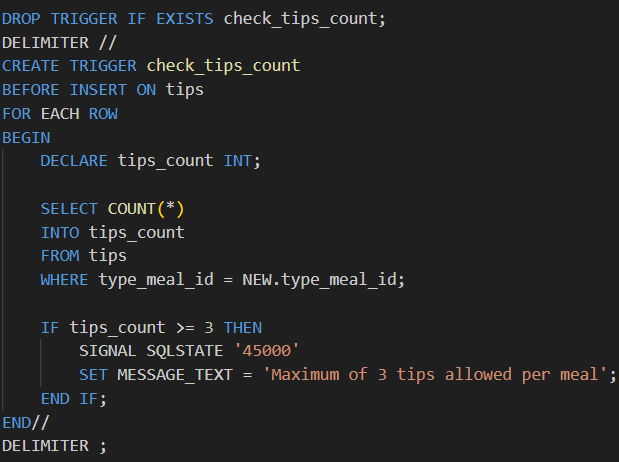
DELIMITER ;

Αρχικά, αλλάζουμε το delimeter σε “//”. Το delimeter οριοθετεί τα statements της SQL και είναι by default καθορισμένο ως “;”. Αλλάζοντάς το προσωρινά σε “//”, μπορούμε να διαχειριστούμε τα statements που θα έχει μέσα το trigger μας ως ένα “ενιαίο μεγαλύτερο statement”.

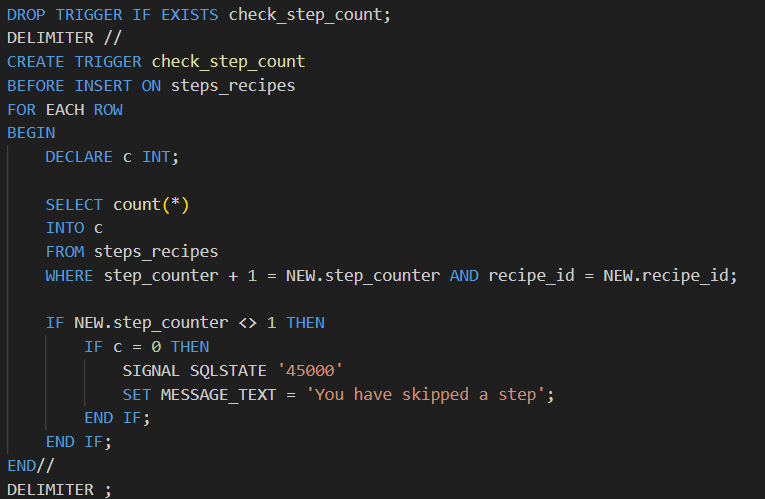
Έπειτα, δημιουργούμε boolean συνθήκες που ελέγχουν αν τα δεδομένα μας θα απορριφθούν λόγω μη τήρησης των constraints. Με το trigger μας, πριν κάνουμε insertion στον πίνακα table, ελέγχουμε αν ισχύουν οι συνθήκες αυτές (IF condition THEN …), ώστε να απορρίψουμε το insertion κάνοντας throw error. Τέλος, επαναφέρουμε το delimeter σε “;”.

Συγκεκριμένα, στον κώδικά μας έχουμε δύο trigger.

Το πρώτο αναφέρεται στην περίπτωση που κάποιος χρήστης προσπαθεί να προσθέσει περισσότερα από 3 tips στον πίνακα τύπου γεύματος. Για να το πραγματοποιήσουμε αυτό, δημιουργούμε ένα εσωτερικό query που επιστρέφει τον αριθμό των tips που υπάρχουν για κάθε τύπο γεύματος, και αποθηκεύουμε αυτόν τον αριθμό σε μια μεταβλητή που ορίζουμε τοπικά μέσα στο trigger. Τέλος, με ένα if statement ελέγχουμε αν έχουμε ξεπεράσει το όριο και τότε δηλώνουμε σφάλμα με το μήνυμα που επιθυμούμε.



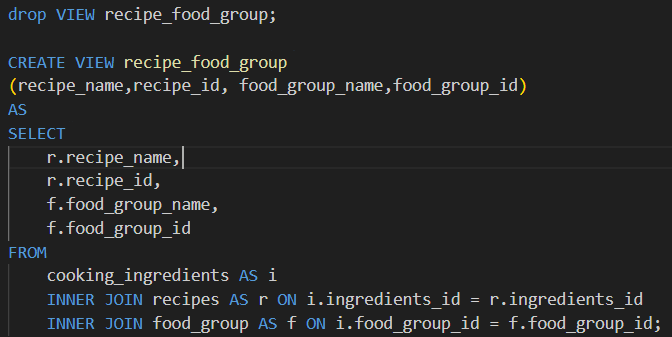
Το δεύτερο trigger αναφέρεται στην περίπτωση που προσπαθούμε να προσθέσουμε ένα βήμα σε μια συνταγή και στη μετρητή βημάτων προσπερνάμε κάποιο βήμα. Κάνουμε την υπόθεση ότι τα βήματα θα μπαίνουν με σειρά. Έτσι, στο trigger μας ελέγχουμε για το `recipe\_id` να είναι το ίδιο, αν το νέο βήμα είναι το επόμενο του προηγούμενου και αυτό είναι μοναδικό. Αν όχι, μέσω του if statement δηλώνουμε σφάλμα και εμφανίζουμε το μήνυμα που επιθυμούμε.



**Views**

Τα views είναι εικονικοί πίνακες της SQL που περιέχουν διάφορα δεδομένα, παρμένα από πραγματικούς πίνακες. Χρησιμοποιούνται για να δώσουμε στους χρήστες περιορισμένη πρόσβαση στην πληροφορία, με ασφάλεια, μη δίνοντας πρόσβαση απευθείας στους πίνακες με τα δεδομένα αλλά εμμέσως δυναμικά, και ευκολία, αφού μπορούν να αποθηκεύονται και να καλούνται ξανά, οποιαδήποτε στιγμή τα χρειαστούμε.

Εδώ έχουμε χρησιμοποιήσει το εξής view:



 Το παραπάνω view δημιουργεί δυναμικά έναν εικονικό πίνακα που συνδέει τους πίνακες cooking\_ingredients, recipes και food\_group μέσω inner join, για να παρουμε απεθιας την πληροφορια του πια συνταγη ανικη σε πια ομαδα τροφιμων, μεσω ενως πινακα.

Επιπλέον, έχουμε τα εξής views στο τέλος του κώδικα:

create view co\_re as

(

select rr.recipe\_id, rr.cooking\_or\_pastry, rr.recipe\_name, rr.recipe\_description, rr.time\_preparation, rr.time\_execution, rr.quantity, rr.ingredients\_id, rr.ethnic\_id, rr.image, rr.image\_caption

from cooker\_recipes as cr inner join recipes as rr

on cr.recipe\_id = rr.recipe\_id

);

create view my\_cook as

(

select \*

from cooker

where cooker\_id = 1

);

Τα συγκεκριμένα views χρησιμοποιούνται ακριβώς για τον λόγο που αναφέρθηκε παραπάνω: να δώσουν περιορισμένη πρόσβαση στον χρηστη μαγειρα. Σιγκεκριμενα, ο πινακας my\_cook διχνει της πληροφοριες για τον ιδιο τον μαγειρα, και μονο. Ενω ο co\_re, διχνει της πληροφοριες ολψν τον συνταγων που γνωριζει ενας σιγκεκριμενος μαγειρας. Τα δυο αυτα τελευτεα views χρησιμευουν για το autharazition.

**Procedures**

Τα procedures είναι κι αυτά, όπως και τα triggers, σύνολα από εντολές που αποθηκεύονται στη βάση δεδομένων και εκτελούνται ως μία ενιαία διαδικασία. Βοηθούν στην αυτοματοποίηση της βάσης μας για λειτουργίες που επαναλαμβάνονται. Τα procedures που χρησιμοποιήσαμε είναι της μορφής:

1. Trigger-like procedure

DELIMITER //

CREATE PROCEDURE procedure

BEGIN    select count(cooker\_id) into x

    (table operations)

    if condition then

        set flag = false;

    ELSE

        SET flag = TRUE;

    END IF;

END //

DELIMITER ;

Εδώ βλέπουμε ότι ελέγχεται η τήρηση των constraints από το procedure και τίθεται η αντίστοιχη τιμή στο flag. Η λειτουργία αυτού του είδους procedure θυμίζει την λειτουργία των triggers που είδαμε παραπάνω.

1. General UtilityProcedure

DELIMITER //

CREATE PROCEDURE procedure

BEGIN

   (table operations)

END //

DELIMITER ;

Εδώ βλέπουμε ότι γίνονται κοινές πράξεις μεταξύ πινάκων και επιστρέφονται ως “μία ενιαία εντολή”.

Το αρχείο “procedures.sql” το τρέξαμε στο xampp με την εντολή source C:/path/procedures.sql, έπειτα από το αρχείο “ddl.sql”(source C:/path/ddl.sql).

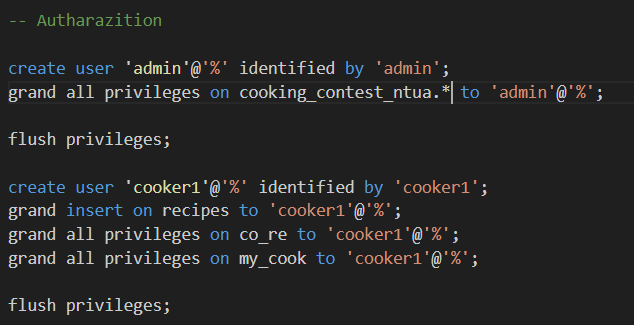
**Dummy Data**

Τα dummy data κατασκευάστηκαν με τη βοήθεια ενός script “create.py” γραμμένου σε Python. Το script αυτό, όπως θα δείτε και στο τέλος του κώδικα, δημιουργεί τυχαία δεδομένα και παράλληλα τα εισάγει σε μορφή “INSERT(data) ( );” σε νέο αρχείο “fake\_data.sql”.

Χρησιμοποίηθηκε η βιβλιοθήκη faker και με το όρισμα των κατάλληλων constraints στο πάνω μέρος του αρχείου, δημιουργήθηκαν και έτρεξαν τα functions μας.

Κάθε function τύπωσε τα επιθυμητά insertions για ακριβώς έναν πίνακα της βάσης μας. Αφού τρέξαμε το αρχείο “create.py” (python create.py όντας στο σωστό directory), δημιουργήσαμε το αρχείο “fake\_data.sql” το οποίο έπειτα εισάγαμε στη βάση τελευταίο, με την εντολή “source C:/path/fake\_data.sql”.

**Authorization**

Όσον αφορά τους χρήστες της εφαρμογής, προσθέτουμε τις εντολές CREATE USER, GRANT και FLUSH PRIVILEGES. Η πρώτη ορίζει έναν χρήστη με το πρώτο όρισμα να είναι το όνομα χρήστη (π.χ. admin). Έπειτα δηλώνουμε από ποιο host μπορεί να συνδεθεί ο συγκεκριμένος χρήστης (επιλέξαμε από οπουδήποτε και άρα βάλαμε %) και τέλος τον κωδικό του για την σύνδεσή του. Ύστερα, μετά τη δημιουργία του, ορίζουμε τα δικαιώματα που έχει μέσω της εντολής GRANT, όπου περνάμε ως όρισμα τα δικαιώματα (π.χ. all privileges για να τα έχει όλα ή insert για να του επιτρέπουμε να κάνει εισαγωγή δεδομένων στη βάση κ.λπ.). Γράφουμε σε ποιον πίνακα αναφέρεται (στην πρώτη περίπτωση βάζουμε όλους από τη βάση μας μέσω της εντολής .\* που δείχνει σε όλα τα tables, ενώ για τον μάγειρα μας βάζουμε τα views που φτιάξαμε παραπάνω και έχουν μόνο την πληροφορία που θέλουμε να έχει πρόσβαση ο μάγειρας). Τέλος, αναφέρουμε τον χρήστη που του δίνουμε τα δικαιώματα και κάνουμε FLUSH PRIVILEGES ώστε να ενσωματωθούν στη βάση μας.

**2 Ανάλυση βημάτων εγκατάστασεις εφαρμογής**

Για να εγκαταστήσουμε την εφαρμογή μας, πρέπει (έχοντας ενεργοποιήσει στο XAMPP το Apache και την MySQL και έχοντας οδηγηθεί στον φάκελο που έχουμε κατεβάσει την MySQL) να εκτελέσουμε την εντολή:

mysql -u root -p

όπου θεωρούμε ότι το username είναι by default το 'root'. Στη συνέχεια (αφού έχουμε κατεβάσει τα αρχεία ddl.sql, procedure.sql και fake\_data.sql) πρέπει να εκτελέσουμε την εντολή:

source path

όπου το path είναι το μονοπάτι όπου βρίσκονται τα αρχεία μας. Έτσι θα εκτελεστούν οι εντολές που περιέχονται σε αυτά και θα έχουμε έτοιμη τη βάση μας για να εκτελέσουμε τα queries.

**4 Ανάλυση των queries**

Qeury 1

Σκοπός του query 1 είναι να πάρουμε τον μέσο όρο των αξιολογήσεων ανά μάγειρα και εθνική κουζίνα.

Ως attributes που θα απεικονίσουμε παίρνουμε το last\_name ,το first\_name και το cooker\_id από τον πίνακα cooker , το country\_name από τον πίνακα ethnic καθώς και τον μέσο όρο των αξιολογήσεων για τον κάθε μάγειρα που υπολογίζεται από την συνάρτηση ((AVG(ee.eval1) + AVG(ee.eval2) + AVG(ee.eval3)) / 3.0) την οποία ονομάζουμε average\_score ,όπου eval1,eval2,eval3 είναι οι αξιολογήσεις του κάθε κριτή/μάγειρα για τον κάθε διαγονιζόμενο μάγειρα από το table episode\_expansion. Έπειτα, κάνουμε join τον πίνακα cooker με το episode\_expansion συσχετίζοντας τους πίνακες ως προς το cooker\_id προκειμένου να πάρουμε τα στοιχεία του μάγειρα (δηλαδή το ονοματεπώνυμο).Στην συνέχεια, κάνουμε join δηλαδή συσχέτιση του πίνακα που έχουμε ήδη δημιουργήσει πιο πάνω με το table recipes ως προς το recipe\_id προκειμένου στην συνέχεια να κάνουμε join τον πίνακα ethnic ως προς το ethnic\_id προκειμένου να πάρουμε το country\_name. Έπειτα, μέσω της where φιλτράρουμε τα δεδομένα θέτοντας is\_judge = 0 προκειμένου να συμπεριλάβουμε μόνο τις εγγραφές που αντιστοιχούν σε διαγωνιζόμενους μάγειρες και όχι σε μάγειρες/κριτές. Τέλος, ομαδοποιούμε τα αποτελέσματα με βάση το cooker\_id και το country\_name όπως υποδεικνύει και το ερώτημα.

Qeury 2

Για δεδομένη Εθνική κουζίνα και έτος, ποιοι μάγειρες ανήκουν σε αυτή και ποιοι μάγειρες συμμετείχαν σε επεισόδια

Επιλέγουμε να απεικονίσουμε το **first\_name**-**last\_name** ως **full\_name** μέσω της **concat**, τη **season\_year** (τη σεζόν της εκπομπής) που είναι attribute του **episode\_expansion**, καθώς και τη χώρα της εθνικής κουζίνας. Κάνουμε join τον πίνακα **episode\_expansion** με τον πίνακα **cooker** για να λάβουμε την πληροφορία του ονόματος του κάθε μάγειρα. Στη συνέχεια, κάνουμε join και με τον πίνακα **recipes** για να πάρουμε το attribute **ethnic\_id** που έχει ο πίνακας **recipes** ως foreign key, για να κάνουμε, τέλος, join με τον πίνακα **ethnic** ως προς το **ethnic\_id** και να λάβουμε το όνομα της χώρας που αναφερόμαστε. Για μια πιο κομψή απάντηση, κάνουμε και ταξινόμηση των απαντήσεων κατά χρόνια και εθνική κουζίνα.

Qeury 3

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

Επιλέγουμε να εμφανίσουμε το cooker\_id, το first.name, το last\_name του κάθε μάγειρα καθώς και την ηλικία του καθενός μετατρέποντας την ημερομηνία γέννησης του σε χρονιά και αφαιρώντας την από το τρέχον έτος 2024 και το ονομάζουμε age ,όλα αυτά τα attributes τα παίρνουμε από table cooker και χρησιμοποιούμε και την count(recipe\_id) καθώς υπολογίζουμε με αυτόν τον τρόπο τις συνολικές συνταγές που μπορεί να εκτελέσει ο κάθε μάγειρας(για αυτό τον λόγο εξάλλου ομαδοποιούμε όλα τα στοιχεία ανά cooker\_id(group by cooker\_id).Για να μπορούμε να επιλέξουμε τα παραπάνω attributes συσχετίζουμε τον πίνακα cooker mε τον cooker\_recipes(περιέχει το recipe\_id) ως προς το cooker\_id. Επίσης, αξίζει να σημειωθεί ότι όλοι οι μάγειρες είναι μικρότεροι των 30 ετών για αυτό στην where θέτω ως περιορισμό το birth\_data να είναι μικροτερο του 1994-05-26(30 χρόνια πριν την ημερομηνία παράδοσης της άσκησης .Τέλος ταξινομούμε τα αποτελέσματα κατά φθίνουσα σειρά (DESC) με βάσει το πόσες συνταγές μπορεί να εκτελέσει ο κάθε μάγειρας μέσω της order.

Qeury 4

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

Επιλέγουμε το `cooker\_id`, το `first\_name` και το `last\_name` από τον πίνακα `episode\_expansion` με την προϋπόθεση ότι το `cooker\_id` του δεν ανήκει στο subquery που έχουμε γράψει. Αυτό το subquery βρίσκει όλα τα id (μία φορά) όλων των μαγείρων που έχουν υπάρξει σαν κριτές στον διαγωνισμό. Άρα, από όλους τους μάγειρες που έχουν υπάρξει στον διαγωνισμό, παίρνουμε αυτούς που δεν ανήκουν στο σύνολο των κριτών, και άρα δεν έχουν υπάρξει ποτέ ως κριτές. Τέλος, κάνουμε `inner join` με τον πίνακα `cooker` για να λάβουμε την πληροφορία του ονοματεπωνύμου όλων αυτών των μαγείρων.

Qeury 5

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

Αρχικά, μέσω της εντολής with δημιουργούμε έναν προσωρινό ονοματικό πίνακα με το όνομα temp. Στον πίνακα αυτό επιλέγουμε ως στοιχειά το cooker\_id, το seazon\_year και την συχνότητα freq που είναι το count(\*) το οποίο δείχνει πόσες εμφανίσεις έχει κάθε κριτής μάγειρας σε διάρκεια μίας σεζόν αφού ο πίνακας έχει ομαδοποιηθεί ανά μάγειρα και σεζόν (group by cooker\_id, season\_year).Τα attributes τα παίρνουμε από τον πίνακα episode\_expansion . Τέλος, αξίζει να σημειωθεί πως παίρνουμε τον περιορισμό οι μάγειρες που συμμετέχουν στα επεισόδια να είναι κριτές (where is\_judge = 1).Έπειτα, επιλέγουμε το cooker\_id δύο φορές από το temp (καθώς αποτελεί το tuple κριτών που θα απεικονίσουμε και έχουν κοινό ετήσιο αριθμό εμφανίσεων στον διαγωνισμό .Αυτό το επιτυγχάνουμε κάνοντάς μία εσωτερική (inner join) του πίνακα temp με τον εαυτό του χρησιμοποιώντας για κριτήριο σύνδεσης να έχουν ίδια freq (δηλαδή ίδιο αριθμό συμμετοχών ) ώστε να δημιουργείται ζευγάρι κριτών με ίδιες συμμετοχές ανά έτος και θέτουμε ως περιορισμό οι μάγειρές να έχουν εμφανιστεί πάνω από 3 φορές.

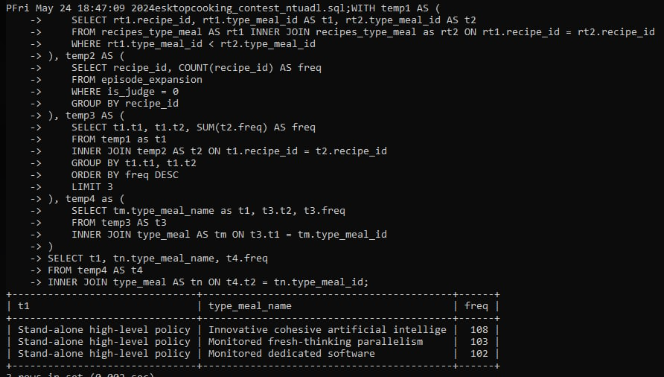
Qeury 6

Αρχικά, υπολογίζουμε τοπικά (μέσω της εντολής `WITH`) κάποιους πίνακες για να μας βοηθήσουν στην εύρεση του αποτελέσματος. Άρα, υπολογίζουμε τον `temp1` που βρίσκει όλα τα ζεύγη ετικετών μιας συνταγής. Ύστερα, υπολογίζουμε όλες τις συνταγές που δόθηκαν σε μάγειρες και τις ομαδοποιούμε ως προς το `recipe\_id` για να βρούμε το πλήθος εμφανίσεών τους (`count`). Στη συνέχεια, ο `temp3` κάνει `INNER JOIN` τους δύο προηγούμενους πίνακες ως προς τις συνταγές για να φιλτράρουμε αρχικά μόνο στις συνταγές που εμφανίστηκαν σε κάποιο επεισόδιο και δεύτερον για να λάβουμε την πληροφορία του πλήθους εμφανίσεων κάθε συνταγής. Έτσι, θα έχουμε έναν πίνακα με όλα τα ζεύγη τύπων γεύματος που εμφανίστηκαν στον διαγωνισμό και τη συχνότητά τους. Τέλος, κάνουμε `INNER JOIN` με τον πίνακα `type\_meal` για να λάβουμε την πληροφορία των ονομάτων των `type\_meal\_id` και τυπώνουμε το αποτέλεσμα.

* Εναλλακτικό Query Plan

Χρησιμοποιούμε το ίδιο ακριβώς query, με τη διαφορά ότι προσθέτουμε την εντολή `FORCE INDEX`, η οποία ουσιαστικά δηλώνει το index με το οποίο θέλουμε να υπολογίσουμε το ερώτημα αυτό, χωρίς να αφήνουμε από μόνη της τη βάση να βρει αυτό που θεωρεί εκείνη τον βέλτιστο τρόπο. Έτσι, προσθέτοντας `FORCE INDEX` στο πρώτο `INNER JOIN` (αρχικά με το `fk\_recipes\_type\_meal` για να προσδιορίσουμε να χρησιμοποιηθεί το index που έχει το `recipe\_id` στον πίνακα `recipe\_type\_meal`) και στο `episode\_expansion` στα index `fk\_episode\_expansion`, `idx\_is\_judge` που ήδη έχουμε, για να δηλώσουμε να κάνει το φιλτράρισμα του `WHERE is\_judge = 0` και το `GROUP BY recipe\_id`, τρέχουμε τα δύο αυτά queries και παρατηρούμε τα παρακάτω.

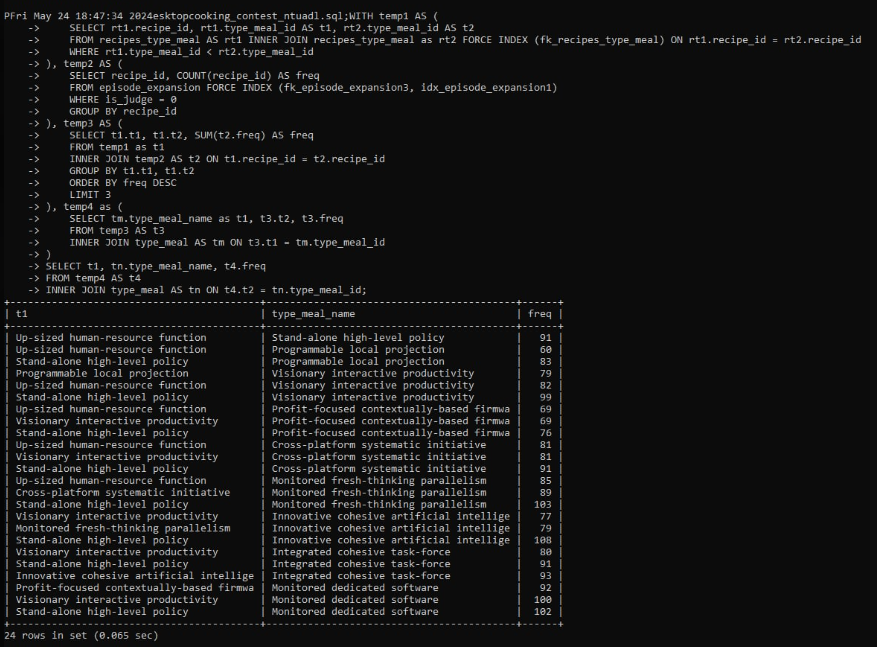
Εκτελυοεμ την εντολη και τρεχουμε το προτο query.



Στην συνεχεια μεσο τησ εντολης  βλεπουμε τον χρονο που χριαστικε να τρεξει το query



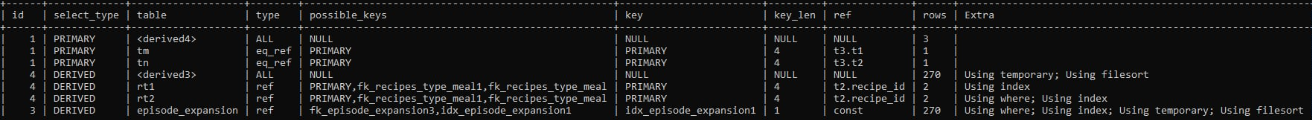
Ομια για το δευτερο (με force index)

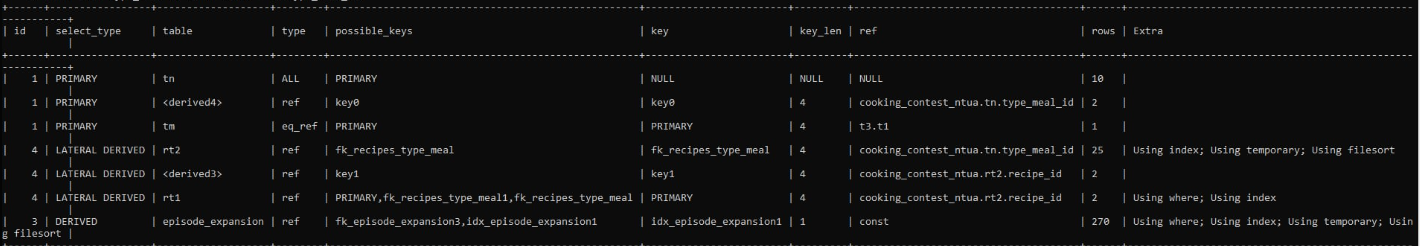




Παρατηρούμε ότι χρειάστηκε πολύ περισσότερος χρόνος για να εκτελεστεί (30 φορές περισσότερο).

Παρατηρούμε δηλαδή ότι το μονοπάτι που επιλέγει η ίδια η βάση δεδομένων μας για να απαντήσει στο ερώτημα είναι πολύ πιο αποδοτικό από αυτό που επιλέγουμε εμείς. Γίνονται δηλαδή διαφορές βελτιστοποιήσεις από πίσω που δεν γνωρίζουμε και συνεπώς δεν υπολογίζονται απλά με ένα index που έχουμε ορίσει (παρόλο που και αυτά βοηθάνε).

Μπορούμε να συμπεράνουμε τα ίδια και από τα παρακάτω, που προέκυψαν μέσω της εντολής EXPLAIN και ύστερα το query (πρώτα το αρχικό query και ύστερα αυτό με το force index). Όπου παρατηρούμε ότι τα μονοπάτια και οι πράξεις που έγιναν σε κάθε query είναι διαφορετικά.



Τελος παρατειρουμε οτι τα αποτελεσματα των δυο query εινια διαφωρετικα (το δευτερο ειανι υπερσυνολο του πρωτου). Αυτο σημβενει καθος λογο του διαφωρετικου μονοπατιου εκτελεσεις, μπορει να προκυψουν διαφωρετικα αποτελεσματα λογο διαφωρετικης σειρας εκτελεσεις εντολων.

Qeury 7

Βρείτε όλους τους μάγειρες που συμμετείχαν τουλάχιστον 5 λιγότερες φορές από τον μάγειρα με τις περισσότερες συμμετοχές σε επεισόδια

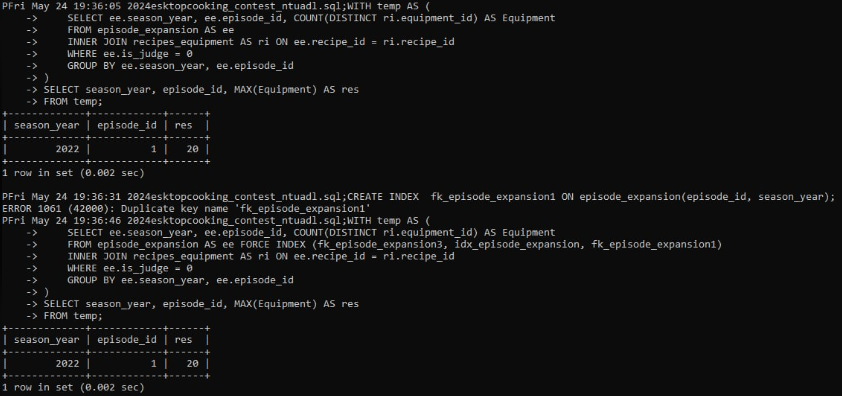
Δημιουργούμε, ένα προσωρινό ονοματικό πίνακα με την ονομασία with και επιλέγουμε ως στοιχεία το cooker\_id και την συχνότητα freq που είναι το count(\*) το οποίο δείχνει πόσες εμφανίσεις έχει κάθε μάγειρας-διαγωνιζόμενος σε όλα τα επεισόδια που έχει διεξαχθεί συνολικά ο διαγωνισμός αφού ο πίνακας έχει ομαδοποιηθεί ανά μάγειρα (group by cooker\_id). Επίσης, μας ενδιαφέρουν μόνο οι διαγωνιζόμενοι μάγειρες για αυτό και is\_judge =0 ως περιορισμός. Έπειτα, επιλέγουμε από τον πίνακα που δημιουργήσαμε το cooker\_id και το πόσες εμφανίσεις έχει και βάλαμε ως περιορισμό μέσω του where και ενός subquery το freq δηλαδή ο αριθμός των συμμετοχών του μάγειρα να είναι μικρότερος από το μέγιστο αριθμό συμμετοχών που έχει μάγειρας στο παιχνίδι – 4 συμμετοχές (δηλαδή τουλάχιστον 5 λιγότερες φορές από τον μάγειρα με τις περισσότερες συμμετοχές σε επεισόδια)

Qeury 8

Σε αυτό το query, κάνουμε join τον πίνακα episode\_expansion με τον πίνακα recipes\_equipment για να κάνουμε την αντιστοιχία όλων των συνταγών και των εξαρτημάτων τους, και τα ομαδοποιούμε ανά έτος και επεισόδιο, μετρώντας πόσες φορές εμφανίστηκαν (ξεχωριστά εκτιμήματα). Έπειτα, βρίσκουμε το MAX του πίνακα.

Για το εναλλακτικό query, βάζουμε force index στο index που έχει το recipe\_id (ως foreign key), στο is\_judge και στο index που έχουν το διπλό foreign key season\_year και episode\_id. Τα αποτελέσματα των δύο μετρήσεων είναι τα εξής:

Τρεχοντας τα παραπανω με ομιο τροπο με το Query 6 εχουμε τα παρακατω αποτελεσματα.

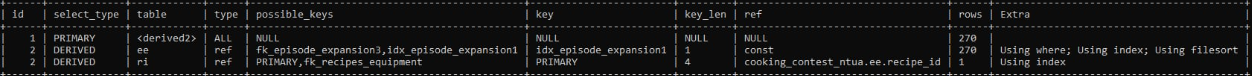


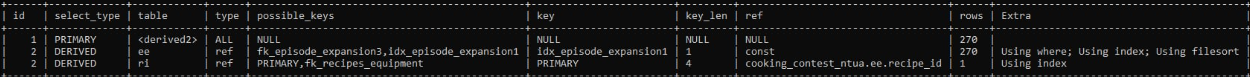
Όπου παρατηρούμε ότι τα δύο αποτελέσματα είναι τα ίδια (σε σύγκριση με προηγούμενο που είχαμε διαφορετική λογική διαφορετικής ροής εκτέλεσης εντολών).

Οι δύο χρόνοι είναι περίπου οι ίδιοι (με τον πρώτο που επιλέγει η βάση μας τον μονοπάτη) να είναι ελάχιστα πιο γρήγορη.

Τελος βλεπουμε και τα μονοπατια που ακολουθισαν μεσω της εντολης explain





Όπου παρατηρούμε ότι είναι ακριβώς τα ίδια.

Συνεπώς, συμπεραίνουμε ότι στη δεύτερη περίπτωση, ο τρόπος που αναδιαμορφώσαμε στη βάση για να εκτελέσει το query είναι αυτός που θεώρησε και η βάση από μόνη της ότι είναι ο βέλτιστος. Σε αντίθετη περίπτωση, προηγουμένως ήταν διαφορετικά, δημιουργώντας μεγαλύτερες καθυστερήσεις. Άρα η βάση μας γενικά βρίσκει βέλτιστα μονοπάτια, εκτός αν είμαστε σίγουροι ότι θέλουμε να επιλέξουμε έναν συγκεκριμένο μονοπάτη για οποιονδήποτε λόγο, οπότε τότε προτείνεται να κάνουμε force index.

Qeury 9

Λίστα με μέσο όρο αριθμού γραμμάριων υδατανθράκων στο διαγωνισμό ανά έτος;

Αρχικά, στο subquery μέσα στο inner join συσχετίζουμε τον πίνακα recipe\_expansion με τον πίνακα recipes\_ingredients μέσω του recipe\_id και παίρνουμε τις συνταγές μόνο των κριτών is\_judge = 0 δημιουργώντας έτσι ένα table με στήλες το ingredients\_id, την ποσότητα του κάθε υλικού και το έτος του διαγωνισμού. Στην συνέχεια, κάνουμε inner join τον πίνακα του subquery με το table nutritions\_info συσχετίζοντας τα ως προς το ingredients\_id και επιλέγουμε ως attributes που θα εμφανίσουμε το μέσο όρο αριθμού γραμμαρίων υδατανθράκων ανά έτος όπου είναι το άθροισμα των υδατανθράκων επί την ποσότητα δια 100 ,όπου 100 είναι όλες οι συνταγές της σεζόν 10 συνταγές από 10 μάγειρες ανά επεισόδιο και συνολικά 10 επεισόδια και επειδή εμφανίζει τον μέσο όρο ανά έτος κάνω group by season\_year

Qeury 10

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

Στην αρχή δημιουργούμε μέσω της with ένα προσωρινό πίνακα με ονομασία temp. Επιλέγουμε το έτος της σεζον το ethnic\_id και μέσω της count(\*) as freq τον αριθμό συμμετοχών των εθνικών κουζινών σε διαγωνισμούς καθώς έχουμε κάνει group by ανά σεζόν και ethnic\_id. Για να μπορούμε να κάνουμε select τα παραπάνω στοιχεία κάνουμε inner join τον πίνακα recipes με τον episode\_expansion ως προς το recipe\_id και γιατί μας ενδιαφέρουν μόνο οι συνταγές των μαγείρων θέτουμε τον περιορισμό το ιs\_judge = 0. Έπειτα δημιουργούμε τον προσωρινό πίνακα tw0\_year\_freq στο οποίο το temp κάνει μία εσωτερική σύνδεση με τον εαυτό του στην οποία το seazon\_year του t1 είναι το προηγούμενο έτος του t2 και το ethnic\_id είναι το ίδιο και κάνουμε select την πρώτη χρονιά το εθνικό αναγνωριστικό της εθνικής κουζίνας καθώς και τις συνολικές συμμετοχές σε διαγωνισμούς σε διάστημα 2 συνεχόμενων ετών. Τέλος, μέσω της where απαιτούμε να εμφανίζονται μόνο οι εγγραφές όπου η συχνότητα των συμμετοχών τους και για τα 2 έτη είναι τουλάχιστον 3.Τέλος, στο τελικό ερώτημα επιλέγουμε τις δύο διαδοχικές σεζόν πχ 2023-2024 ως περίοδο με αυτή την εκτύπωση χάρη στην concat το αναγνωριστικό της εθνικής κουζίνας και τις συνολικες φορές που εμφανίζονται σε διάστημα 2 συνεχόμενων ετών. Τα παραπάνω τα λαμβάνουμε από την inner join του πίνακα two\_year με τον εαυτό του, ως προς την ίδια διετία, για να λάβουμε όλα τα ζευγάρια εθνικών κουζίνων με ίδια διετία, και τα φιλτράρουμε ώστε το ένα ethnic\_id να είναι μικρότερο του δεύτερου (για να μην έχουμε δύο φορές τα ίδια ζευγάρια με την ίδια κουζίνα) και βάζουμε περιορισμό να έχουν ίδια συχνότητα εμφάνισης στη διετία αυτή.

Qeury 11

Αρχικά, βρίσκουμε στο subquery όλα τα ζευγάρια κριτών και μαγείρων (με αυτήν τη σειρά) σε ολόκληρο τον διαγωνισμό και λαμβάνουμε την πληροφορία του βαθμού που έβαλε ο τάδε κριτής στον τάδε μάγειρα μέσω της εντολής case. Έχουμε κάνει την παραδοχή ότι στον πίνακα episode\_expansion οι τρεις κριτές έχουν τιμές null στη θέση του evalx όπου δεν ανήκουν και not null εκεί που ανήκουν, οπότε μπορούμε εύκολα να δούμε ποιο βαθμό έχουν βάλει σε κάθε διαγωνιζόμενο. Στη συνέχεια, από αυτόν τον πίνακα, τα ομαδοποιούμε ως προς τα id των μαγείρων κριτών και προσθέτουμε τα κοινά tuples σε ένα άθροισμα για το σκορ. Τέλος, ταξινομούμε τον τελικό πίνακα και παίρνουμε μόνο τα 5 μεγαλύτερα σκορ.

Qeury 12

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

Αρχικά δημιουργόυμε ένα μέσω της with έναν πρψσορινό πίνακα ranked\_recipes επιλέγοντας ως attributes το συνολικό βαθμό δυσκολίας των συνταγών για αυτό έχουμε κάνει sum το difficulty , την σεζόν και το επισόδιο ενώ προκειμένου να παρουσιάζουμε το αθροισμα της δυσκολίας των συνταγών ανα επεισόδιο και έτος ομαδοποιούμε τα στοιχεία ανα σεζόν και έτος.Επίσης προκειμένου να μπορούμε να χρησιμοποιήσουμε αυτά τα στοιχεία συσχετίζουμεμέσω inner join τα στοιχεία του recipes με το episode\_expansion ενώ γιατί μας ενδιαφέρει μόνο η δυσκολία των συνταγών των διαγονιζόμενων μαγείρων θέτουμε των περιορισμό is\_judge = 0.Ακόμη, δημιουργούμε το προσωρινό πίνακα max\_ranked\_recipes και επιλέγουμε από τον πίνακα ranked\_recipes την μεγαλύτερη συνολική δυσκολία από τον προηγούμενο προσωρινό πίνακα που δημιουργήσαμε .Τέλος, κάνουμε inner join τους δύο προσωρινούς πίνακες ως προς συσχετίζοντας τους ως προς το total\_difficulty kαι το max\_total\_difficulty προκειμένου να εμφανίσουμε το πιο τεχνικά δύσκολο, από πλευράς συνταγών, επεισόδιο του διαγωνισμού ανά έτος

Qeury 13

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

Ο προσωρινός πίνακας ranked\_episodes υπολογίζει το συνολικό βαθμό επαγγελματικής κατάρτισης (total\_rank) για κάθε επεισόδιο και σεζόν.Η κατάταξη κάθε μάγειρα και κριτή προστίθεται στον συνολικό βαθμό χρησιμοποιώντας μία case (σαν την switch στην c) και με αυτόν τον τρόπο αντιστοιχίζουμε κάθε επαγγελματικό βάθμό μάγειρα σε αριθμό. Τέλος, ομαδοποιούμε τις εγγραφές ανά σεζόν και επεισόδιο ώστε να υπολογίζεται ο συνολικός βαθμός επαγγελματικής κατάρτισης για κάθε επεισόδιο. Έπειτα, δημιουργούμε έναν επιπλέον προσωρινό πίνακα που υπολογίζει τον ελάχιστο συνολικό βαθμό κατάρτισης από τον πίνακα ranked\_episodes που δημιουργήσαμε προηγουμένως. Στο τελικό ερώτημα συσχετίζουμε με inner join τα αποτελέσματα από το ranked\_episodes με αυτά από το min\_ranked\_episodes ως προς την τελική βαθμολογία total\_rank. Το αποτέλεσμα είναι το επεισόδιο με τη χαμηλότερη συνολικά επαγγελματική κατάρτιση, λαμβάνοντας υπόψη τους βαθμούς όλων των συμμετεχόντων(κριτών και μαγείρων)

Qeury 14

Ποια θεματική ενότητα έχει εμφανιστεί τις περισσότερες φορές στο διαγωνισμό;

Αρχικά στο subquery επιλέγουμε τις θεματικές ενότητες και υπολογίζουμε τον αριθμό των εμφανίσεών τους.Για αυτό συσχετίζουμε τα στοιχεία του πίνακα episode\_expansion με τον πίνακα recipes\_topics\_recipes ως προς το recipe\_id πτοκειμένου να συνδέσουμε κάθε συνταγή με θεματικές ενότητες και έπειτα κάνουμε και join τον πίνακα recipes\_topics προκειμένου να προσθέσουμε και τα ονόματα των θεματικών ενοτήτων.Τέλος ομαδοποιούμε ανά topics\_id προκειμένου να υπολογίσουμε τον αριθμό των εμφανίσεων κάθε θεματικής ενοτητας.΄Στο εξωτερικό query επιλέγουμε το όνομα της θεματικής ενότητας και το μέγιστο αριθμό εμφανίσεων MAX(topic\_count) από το αποτέλεσμα του subquery

Qeury 15

Ποιες ομάδες τροφίμων δεν έχουν εμφανιστεί ποτέ στον διαγωνισμό;

Στο subquery επιλέγουμε το food\_group\_id δηλαδή τις ομάδες τροφίμων που έχουν εμφανιστεί στον διαγωνισμό.Συσχετίζουμε τους πίνακες episode\_expansion kαι recipes\_ingredients ως προς το recipe\_id προκειμένου να βρούμε ποια υλικά έχουν χρησιμοποιηθεί σε συνταγές των επεισοδίων.Επειτα, κάνουμε inner join mε το cooking\_ingredients για να συνδέσουμε τα υλικά με τις ομάδες τροφίμων.Τέλος τα ομαδοποιούμε να food-group\_id προκειμένου να πάρουμε τα αποτελέσματα με βάση τις ομάδες τροφίμων ώστε να συμπεριλάβουμε κάθε ομάδα τροφίμων μόνο μία φορά στο υποερώτημα. Στο εξωτερικό ερώτημα επιλέγουμε τα ονόματα των ομάδων τροφίμων από τον πίνακα food\_group και θέτουμε ως περιορισμό να μην εμφανίζονται οι ομάδες τροφίμων που έχουν εμανιστεί στον διαγωνισμό ,τις οποίες υπολογίσαμε μέσω του subquery

STASINE GRAPSE TA THS PYTHON TA TRIGGERS KAI TA PROCEDURES KAI BALE TO LINK GIA TA REPOSITORY KAI FTIAJE KAI TO GITHUB ΝΑ ΕΧΕΙ ΚΑΙ ΤΟ RELATIONAL SCHEMA