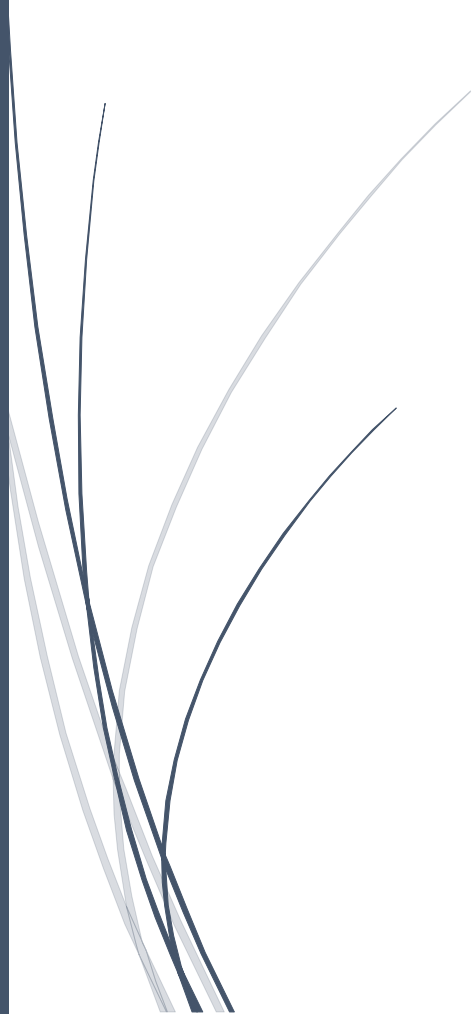
16/11/2021

Projet BDD

Partie 1



Saba Woldearegay

Tables des matières

Introduction

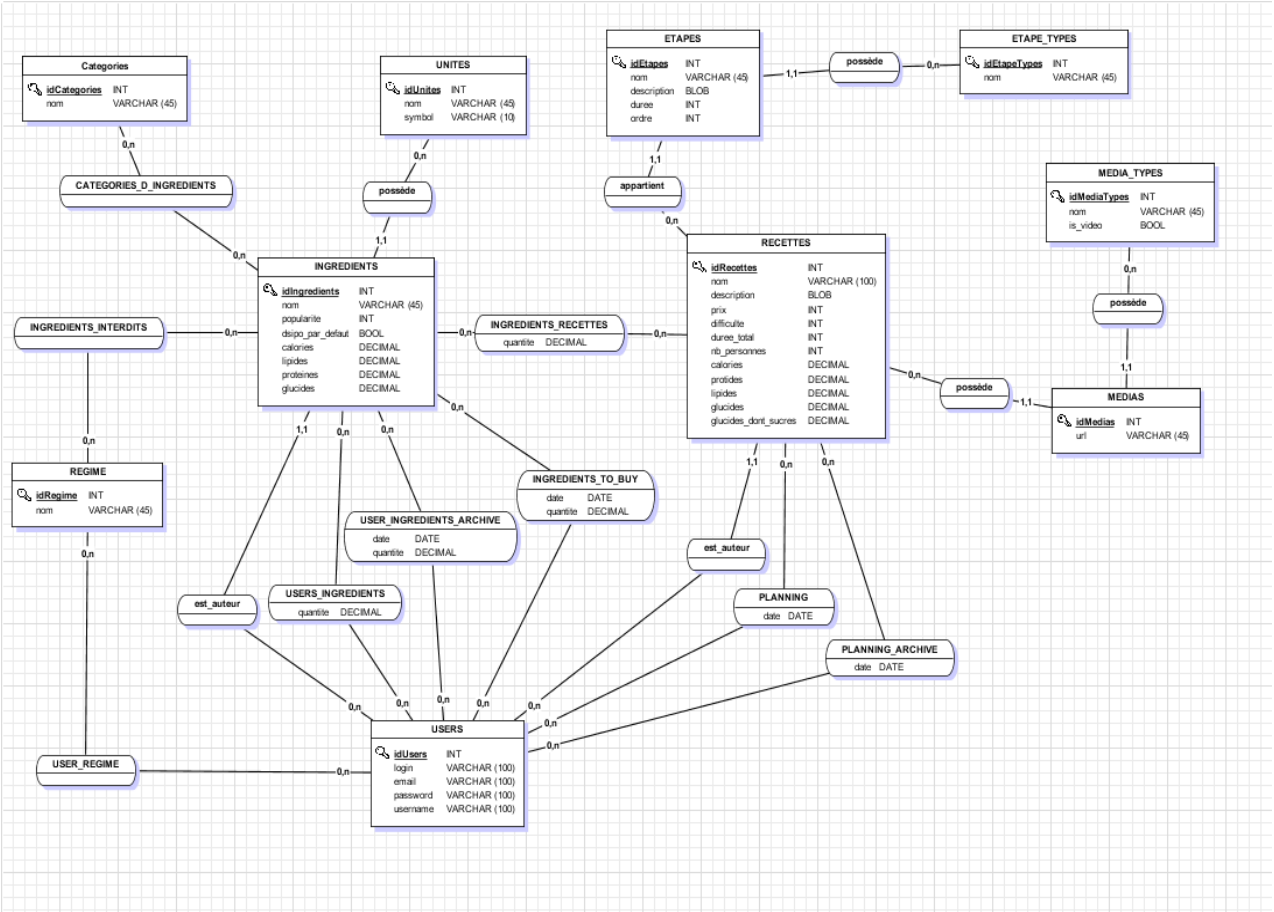
Dans le cadre de l'UE de base de données de la 3ème année de licence Informatique à l'Université de Strasbourg, il nous a été demander de réaliser la conception de la base de données pour gérer des recettes de cuisines.

Ceci est le rapport pour la 1ère partie du projet : la modélisation et la conception de la base de données.

Modélisation

Modèle entité-association

Voici le modèle entité association de la base de données qui respecte les fonctionnalités indiquées dans l'énoncer :



Voyons les tables ci-dessus de plus proche.

Table USERS

- idUsers : la clé primaire qui est unique pour chaque utilisateur.
- Login : pour se connecter au site web, unique.
- Email : contiendra l'adresse mail.
- Password : mot de passe chiffré de l'utilisateur.
- username : pour que l'utilisateur puisse personnaliser son nom d'affichage.

Table RECETTES

- idRecettes : clé primaire et unique permettant de référer la recette.
- Nom : nom de la recette (pas unique car plusieurs façons de faire la même recette).
- Description : description de la recette, peut être vide.
- Prix : un entier entre 1 et 5 indiquant le prix :
  - 1 : gratuit
  - 2 : pas cher
  - 3 : prix correct
  - 4 : un peu cher
  - 5 : prix élevé
- Difficulte : un entier entre 1 et 5 indiquant la difficulté :
  - 1 : très facile
  - 2 : facile
  - 3 : intermédiaire
  - 4 : difficile
  - 5 : très difficile
- Duree_total : un entier représentant la durée totale en minutes pour effectuer cette recette.
- Nb_personnes : un entier indiquant le nombre de personnes pour qui a été prévue cette recette.
- Calories : un entier représentant le nombre de calories.
- Protides : un décimal représentant la quantité de protides.
- Lipides : un décimal représentant la quantité de lipides.
- Glucides : un décimal représentant la quantité de glucides.
- Glucides_dont_sucres : un décimal représentant la quantité de sucre.

Les valeurs nutritionnelles concernent l'ensemble de la recette et non pas celle de chaque ingrédient individuellement.

Table UNITES

- idUnites : clé primaire pour référencer l'unité dans la table INGREDIENTS.
- Nom : pour donner un nom à l'unité (ex : Litres).
- Symbol : pour donner le symbole correspondant à cette unité (ex : L pour litres), unique.

J’ai choisi de ne pas mettre un champ unité dans la table INGREDIENTS pour faciliter l’ajout/sélection de nouvelles unités pour les utilisateurs sans avoir à changer le code html manuellement.

Table INGREDIENTS

- idIngredients : clé primaire unique pour identifier un ingrédient.
- Nom : nom de l’ingrédient.
- Popularite : à chaque recherche par ingrédient avec cet ingrédient, on incrémente sa valeur (au départ à 0). Cette valeur est réinitialisée chaque mois à 0.
- Dispo_par_defaut : par défaut vaut false (ou 0) permet de savoir si on peut considérer que cet ingrédient est toujours disponible chez un utilisateur (ex : le sel, le poivre, l’eau…), dans ce cas ce champ vaut true (ou 1).
- Calories : un entier représentant le nombre de calories.
- Lipides : un décimal représentant la quantité de lipides.
- Protéines : un décimal représentant la quantité de protéines.
- Glucides : un décimal représentant la quantité de glucides.

Lees valeurs nutritionnelles sont renseignées par unité et on a qu’a multiplier la quantité par les valeurs pour trouver les valeurs finales pour la recette.

Table REGIME

- idRegime : clé primaire unique représentant un régime particulier.
- Nom : le nom de cette régime (ex : vegan…).

Table ETAPES

- idEtapes : clé primaire unique de l’étape.
- Nom : nom de l’étape (ex : etape 1, préparer les légumes…).
- Description : décrit l’étape en détail, peut être vide.
- Duree : durée de l’étapes en minutes.
- Ordre : utile si l’on souhaite réorganiser l’ordre des étapes d’une recette, pour finir il n’y aura plus qu’à ordonner par ce champ pour avoir la recette dans le bon ordre

Table ETAPE_TYPES

- idEtapeTypes : clé primaire unique pour y faire référence depuis la table ETAPES.
- Nom : nom de l’étape (ex : Cuisson, Préparation, Dégustation…).

Table MEDIAS

- idMedias : clé primaire unique du media.
- url : l’url vers le média en question.

Cette table liste les différents médias disponibles pour les recettes (photo, vidéos…).

Table MEDIA_TYPES

- idMediaTypes : clé primaire unique du type de média.
- Nom : le nom du type de média (ex : photo…).
- Is_video : un booléen, si true (càd 1) alors il s'agit d'une vidéo sinon c'est une photo.

Cette table liste les différents types de média possible (photo, vidéo YouTube, vidéo Dailymotion, . . .). L'avantage est que l'on peut ajouter rapidement dans la liste un nouveau service de vidéos qui viendrait de sortir par exemple.

Table CATEGORIES

- idCategories : clé primaire unique pour la catégorie d'ingrédient.
- Nom : le nom de la catégorie.

Cette table permet de stocker les différentes catégories d'ingrédients dans le but de remplacer un ingrédient par un autre qui lui est équivalent si jamais un utilisateur n'a pas un certain ingrédient ou s'il il a une restriction diététique.

Contraintes d'intégrité

Voici la liste des contraintes d'intégrité que devront respecter le modèle :

Tous les id devront être non vides et uniques dans chacune des tables.

Table USERS

- Login doit être unique et non vide
- Email doit être unique et non vide ainsi que respecter le format d'un email : @ précédé et suivi d'au moins un caractère.
- Password doit être non vide

Table RECETTES

- Nom doit être non vide
- Prix est un entier compris entre 1 et 5
- Difficulte est un entier compris entre 1 et 5
- Durre_total est un entier strictement positif
- Nb_personnes est un entier strictement positif
- Calories, protides, lipides et glucides doivent être supérieure ou égal à 0
- Glucides_dont_sucres doit être supérieure ou égal à 0 et inférieure ou égal à la valeur du champ glucides

Table UNITES

- Nom doit être unique et non vide
- Symbol doit être unique

Table INGREDIENTS

- Nom doit être unique et non vide
- Popularite doit être supérieure ou égale à 0
- Calories, lipides, protéines, glucides et glucides_dont_sucres doivent être supérieure ou égale à 0

Table REGIME

- Nom doit être unique et non vide

Table ETAPES

- Nom doit être unique et non vide
- Duree doit être supérieure ou égale à 0

Table ETAPE_TYPES

- Nom doit être unique et non vide

Table MEDIAS

- Url doit être unique et non vide

Table MEDIA_TYPES

- Nom doit être unique et non vide

Table CATEGORIES

- Nom doit être unique et non vide

Ensemble des relations entre les tables

Dans cette partie nous verrons les différents liens entre les tables, les cardinalités, et les tables de jonctions qu'il faudra encore créer.

Tout d'abord, un utilisateur peut s'inscrire. Il peut être l'auteur de 0 à n recettes, et une recette a été écrite par un seul et unique utilisateur. Une recette est composée de 0 (dans le cas où la recette viendrait d'être crée, et pas encore publiée) à n étapes, et une étape appartient à une seule et unique recette.

Une étape possède un seul et unique type d'étape, et un type d'étape peut appartenir à 0 ou n étapes différentes.

Une recette possède de 0 à n médias, et un média appartient impérativement à une seule et unique recette. Un média possède un et un seul type de média, et un type de média peut contenir de 0 à médias différents.

Une recette peut être dans 0 à n planning d'utilisateur, tout comme un planning d'utilisateur peut contenir de 0 à n recettes ; il faut donc créer une table de jonction, planning, dans laquelle en plus des références vers l'utilisateur et la recette, je mets un champ de type date que j'appelle date pour dire quand est-ce que la recette est programmée dans le planning de l'utilisateur.

Ce champ devra respecter la contrainte du fait que sa valeur ne doit pas être dans plus d'un mois, et que si la date est dépassée, on doit déplacer l'élément dans la table planning_archive qui est identique à celle-ci, sauf sans les contraintes sur le champ date.

Une recette est composée de 0 à n ingrédients (0 dans le cas d'une recette non publiée, qui vient d'être crée par exemple), et un ingrédient peut être utilisé par 0 à n recettes ; il faut donc une table de jonction, que j'appelle ingredients_recettes, dans laquelle j'ai, en plus des références vers les id de recette et ingrédient, un champ décimal quantite, qui me permet de stocker la quantité de cet ingrédient qu'il faut pour la recette.

Un ingrédient peut appartenir à 0 à n catégories, et une catégorie peut contenir 0 à n ingrédients ; je créer donc une table de jonction categories_d_ingredients qui contiendra les références vers les id des ingrédients et de leurs catégories.

Chaque ingrédient est exprimé selon une et une seule unité stockée dans la table unites, et une unité est utilisée par 0 à n ingrédients. Il faut donc créer une référence dans la table ingredients qui pointe vers le champ id de la table unites.

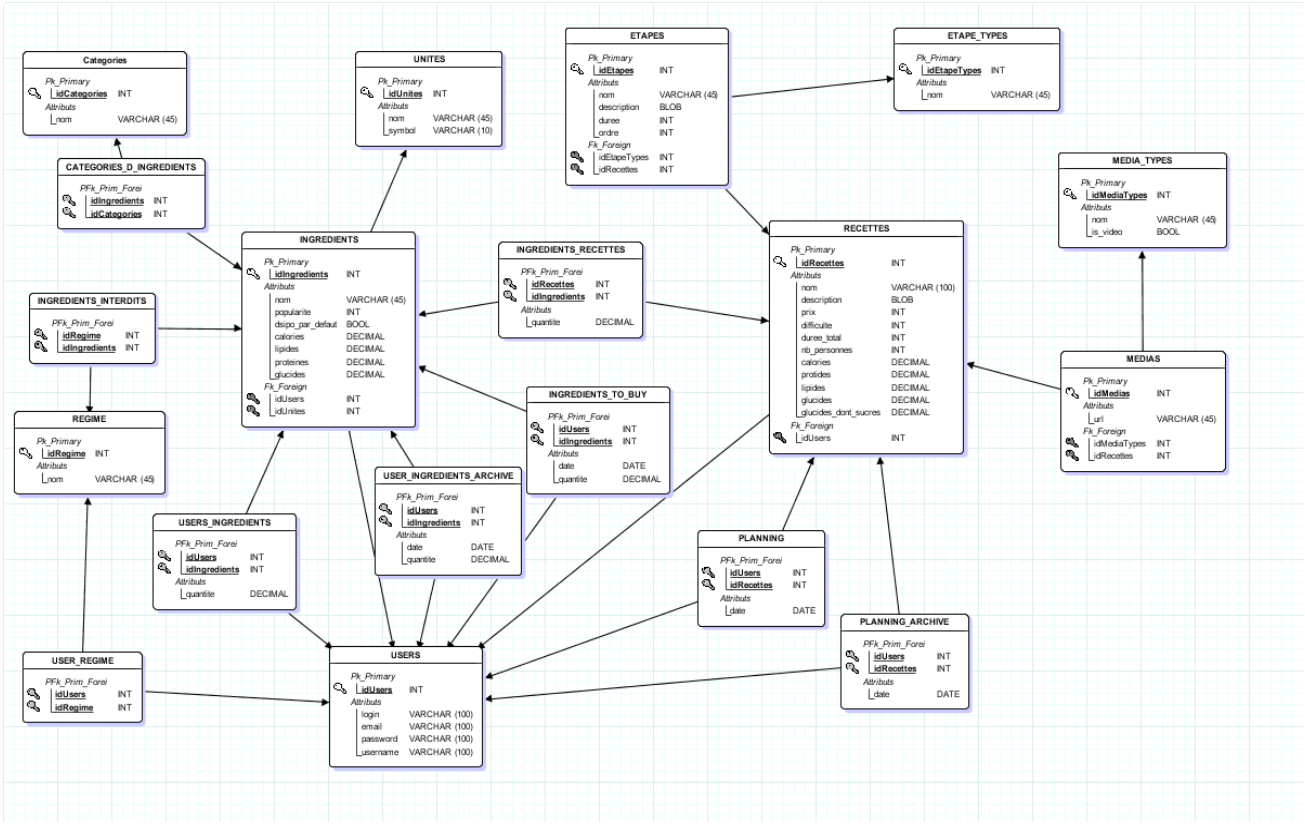
Un ingrédient est ajouté par un seul et unique utilisateur, et un utilisateur peut ajouter de 0 à n ingrédients ; c'est donc la raison pour laquelle j'ai ajouté un champ auteur à la table ingredients qui pointe vers le champ id de la table users, dans le but d'identifier l'auteur de l'ingrédient.

Chaque utilisateur peut avoir 0 à n ingrédients chez lui, et un ingrédient peut être possédé par 0 à n utilisateurs ; je créé donc une table de jonction users_ingredients, avec les références vers les id de l'utilisateur et de l'ingrédient, ainsi qu'un champ quantite qui servira à stocker la quantité de l'ingrédient en question que possède l'utilisateur.
Idem pour les listes d'achats, mais je mets en plus un champ qui contient la date à laquelle l'ingrédient a été ajouté à la liste d'achat ; cette date ne peut pas être dans plus d'un mois et inférieure à aujourd'hui. La même table pour les archives mais sans les contraintes sur la date.

Enfin, pour finir, un régime peut concerner de 0 à n utilisateurs, un utilisateur peut avoir 0 à n régimes ; il faut donc créer une table de jonction que j'appelle user_regime dans laquelle se trouve simplement les références vers les id de l'user et du régime.
Et un régime peut interdire un certain nombre d'ingrédients, tout comme un ingrédient peut être interdit par 0 à n régimes ; je créé donc une table de jonction ingredients_interdits, avec les références vers le régime et l'ingrédient.

Modèle logique relationnel

Voici le modèle logique relationnel de la base de données construit sur la base de modèle entité-association vu précédemment :



Précisions

Il se peut qu'il y soit quelques différences sur les types et les quantités des varchar entre le code et le rapport/modèle EA.