PROJET BASE DE DONNÉES
Coquet Jean-Philippe

L'objectif de fin de semestre était de réaliser une application mono-utilisateur pour gérer une base de données en utilisant SQLite. Nous avions des exemples mis à notre disposition. Et le but était de rendre la base de données utilisable avec une interface graphique codée avec le langage de programmation de notre choix.
L'application permet de gérer la base de données d'un restaurant avec des tables gérant les clients ou encore le stock.

Pour rendre mon projet dans les temps et concevoir une interface graphique agréable au visuel et à utiliser, je suis parti sur une interface en html/css, mais n'ayant pas de connaissances en php,j'ai changé pour le langage python qui est natif au sql. Grâce aux différents exemples, j'ai eu l'idée de faire une base de données qui tourne autour de la restauration. J'ai donc rédigé mes premières tables, puis j'ai créé le modèle ENTITE_ASSOCATION. J'ai opté pour une orientation de mes views sur des tops pour les « clients » du restaurant ou encore qui tourne autour du nombre de plats vendus. J'ai articulé ce restaurant autour de stocks de chaque plat mais également avec des réservations de client ainsi que le montant que chaque client a dépensé dans un intervalle de temps donné. Pour l'interface graphique j'ai opté pour quelque chose de minimaliste mais totalement fonctionnel et épuré. Le but étant de relier le plus efficacement possible la base de données avec l'interface graphique.

La base de données étant créée avec les différentes tables, il fallait maintenant réussir à relier cette base à une interface. Le sql étant natif à python cela s'est avéré moins difficile que prévu. Or, il a fallu intégrer toutes les idées que j'avais pour gérer les bases de données ou encore différents « events ».

Schéma EA :

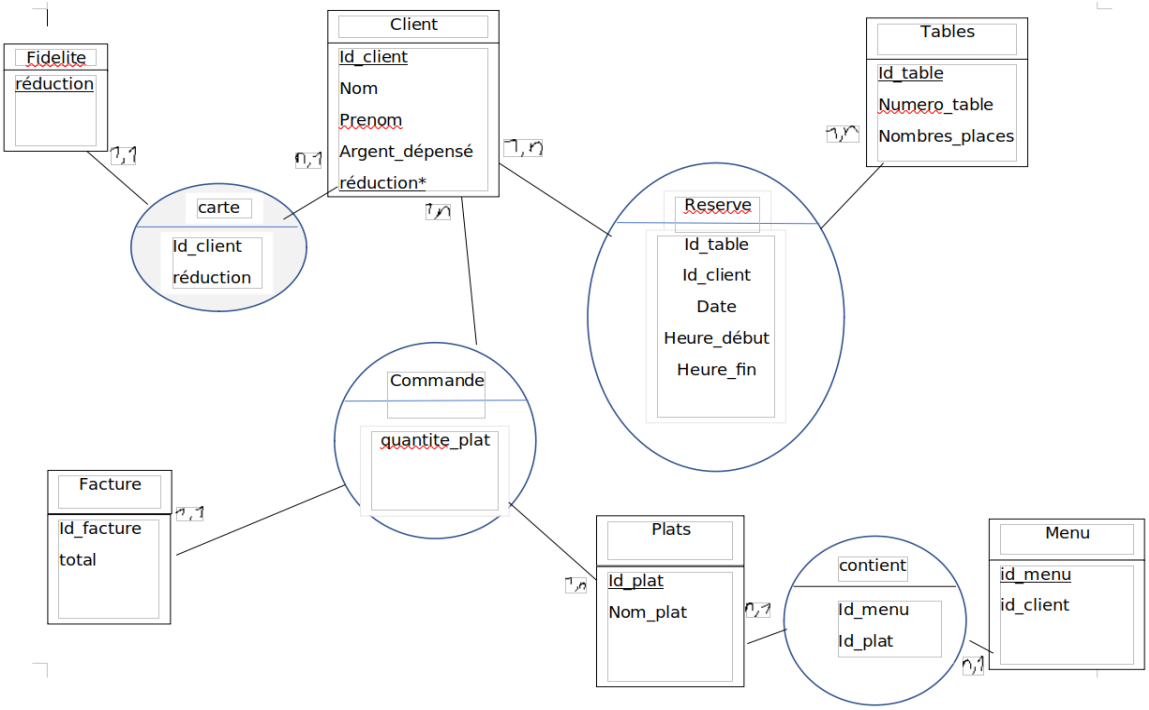

Schéma relationnel :

7 tables :
-stock
(id_plat, nom_plat, quantite_plat)
-réservation
(id_client*, id_table*, date, heure_debut, heure_fin)
-fidélité
(id_client*, réduction)
-facture
(id_facture, id_client*, total)
-client
(id_client, nom, prenom, argent_depense)
-tables
(id_table, numero_table)
-menu
(id_menu, id_plat*, prix_plat)
gras = clé primaire
* = clé étrangère
____ = contrainte

4 views :
-Afficher le plat qui a été le plus stocké.
-Table qui a le plus de place.
-Client ayant le plus dépensé.
- Plat le plus cher.

4 déclencheurs :

- Après avoir ajouté une réservation, les quantités de chaque plat dans le stock est augmenté de 100.
- Après avoir supprimé un client, sa réservation est également supprimée.
- Après avoir supprimé une réservation, le client est également supprimé.
- Après avoir modifié un client, son id_table est augmenté de 1.