**Rapport du projet *Enjoy!****Projet de base de données*

Réalisé par :  
 DUCOUDRÉ Max  
 GUILLET Elsa  
 GIBOZ Alexandre

Table des matières

[Présentation du projet : 3](#_Toc125552220)

[Entité-association : 4](#_Toc125552221)

[Diagramme d’entité-association 4](#_Toc125552222)

[Éclaircissements : 4](#_Toc125552223)

[Limites du diagramme : 4](#_Toc125552224)

[Modèle relationnel : 6](#_Toc125552225)

[Tables : 6](#_Toc125552226)

[Clés étrangères : 6](#_Toc125552227)

[Contraintes : 7](#_Toc125552228)

[Maquette du site WEB 8](#_Toc125552229)

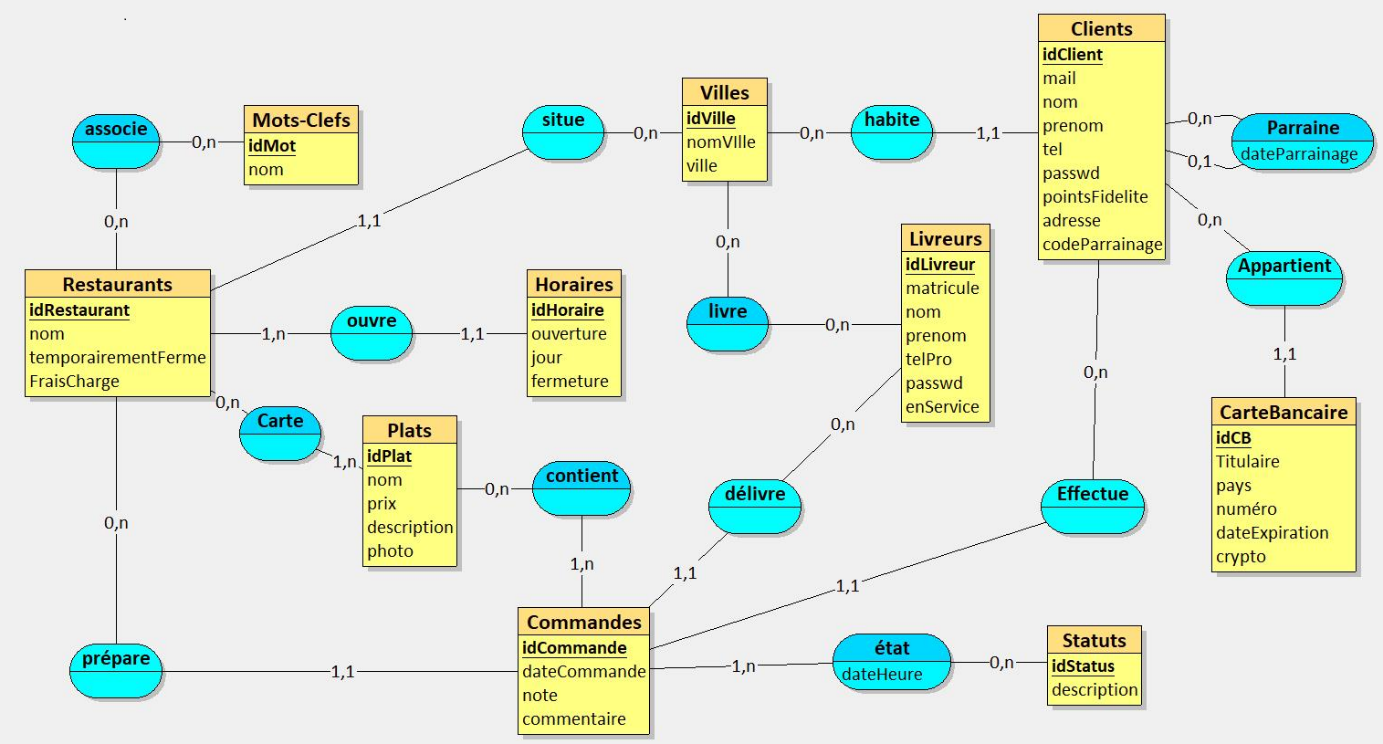
# Présentation du projet :

Enjoy ! est une plateforme de livraison de plats à domicile, dont le but est d’offrir à ses utilisateurs un vaste choix de restaurants et aux restaurants partenaires de ne pas avoir à gérer leur propre service de livraison. Ainsi, Enjoy ! ne vend pas directement de nourriture, mais se contente de transmettre aux restaurants les commandes passées sur la plateforme, et met ensuite à disposition des restaurants sa flotte de livreurs afin d’acheminer les plats préparés jusqu’au domicile des clients.

*Plus de détails sur « consignes\_enjoy.pdf »*

# Entité-association :

## Diagramme d’entité-association



*mcd.jpg*

## Éclaircissements :

Un « horaires » représente une « tranche » de journée d’ouverture pour un restaurant.  
Un « statuts » représente l’état d’une commande, il peut être : « en attente », « en préparation », « en cours de livraison »,  « livré » et « annulé ».   
L’attribut « codeParrainage » est un code unique que le client peut partager a ses futurs filleuls.

## Limites du diagramme :

A partir de ce diagramme, on peut déceler la structure d’une base de données relationnelle.  
Néanmoins, avec ce modèle de base de données, certaines contraintes définies dans le cahier des charges ne pourront pas être respectés. Les cas suivants sont possibles alors qu’ils ne devraient pas :

- Un restaurant peut avoir un horaire d’ouverture qui mord sur un horaire de fermeture.  
- Une commande peut avoir un avis & une note alors que son statut n’est pas « livré ».  
- Un client ayant une commande peut ne pas avoir de carte bancaire.  
- Un livreur peut avoir une commande qui n’appartient pas à sa ville.  
- Les dates de changement de statuts (état) peuvent ne pas être dans le bon ordre chronologique.  
- Un livreur peut ne pas être en service mais avoir une commande.  
- Si un restaurant est « temporairement fermé » ou qu’il est fermé due à ses « horaires », il peut avoir des commande

Néanmoins, quasiment toutes les limites citées ici peuvent être corrigées avec des triggers ou une bonne utilisation de la base de données depuis le frontend.

Note : Ici ne sont pas listé les soucis pouvant être corrigés avec des « constraints » et des « checks » lors de l’implémentation SQL.

Modèle relationnel :

Depuis le schéma d’entité-relation, on peut écrire à quoi ressemblera la structure de la base de données relationnelle.

Tables :  
Mots\_Clefs = (**idMot SERIAL**, nom *VARCHAR(50)*);  
**Plats = (idPlat SERIAL**, nom *VARCHAR(50)*, prix *DECIMAL(15,2)*, description *VARCHAR(50)*, photo *VARCHAR(50)***);**  
**Livreurs = (idLivreurSERIAL**, matricule *CHAR(8)*, nom *VARCHAR(50)*, prenom *VARCHAR(50)*, telPro *VARCHAR(50)*, passwd *VARCHAR)*, enService *BOOL***);**  
**Villes = (idVilleSERIAL**, nomVille, ville *CHAR(5)***);**  
**Clients = (mail *VARCHAR(50)***, idClient *INT*, #parrain INT, nom *VARCHAR(50)*, prenom *VARCHAR(50)*, tel *VARCHAR(50)*, passwd *VARCHAR(50)*, pointsFidelite *INT*, adresse *VARCHAR(50), #idVille, dateInscription DATE***);**  
**Statuts = (idStatusSERIAL**, description *VARCHAR(50)***);**  
**Restaurants = (idRestaurantSERIAL**, nom *VARCHAR(50)*, temporairementFerme *LOGICAL*, FraisCharge *DECIMAL(15,2), #idVille***);  
Horaires = (idHoraireSERIAL**, ouverture *TIME*, jour *INT*, fermeture *TIME, #idRestaurant***);**  
**Commandes = (idCommandeSERIAL**, dateCommande *DATE*, note *INT*, commentaire *VARCHAR(50), #idRestaurant, #idLivreur, #mail***);  
Carte = (*#idRestaurant, #idPlat*);  
associe = (*#idRestaurant, #idMot*);**  
**livre = (*#idLivreur, #idVille*);**  
**contient = (*#idPlat, #idCommande*);**  
**état = (*#idCommande, #idStatus***, dateHeure *DATETIME***);**

## Clés étrangères :

FK : Restaurants(idVille) référence Villes(idVille)FK : Clients(idVille références Villes(idVille

FK : Horaires(idRestaurant) référence Restaurants(idRestaurant)  
FK :Commandes(idRestaurant) réféfence Restaurants(idRestaurant)

FK : Commandes(idLivreur) référence Livreurs(idLivreur)  
FK : Commandes(mail) référence Client(mail)  
FK : Carte(idRestaurant) référence Restaurants(idRestaurant)  
FK : Carte(idPlat) référence Plats(idPlat)  
FK : associe (idRestaurant) référence Restaurants(idRestaurant)  
FK : associe (idMot) référence Mots\_clefs(idMot)  
FK : livre(idLivreur) référence Livreurs(idLivreur)  
FK : livre(idVille) référence Ville(idVille)  
FK : contient(idPlat) référence Plats(idPlat)  
FK : contient (idCommande) référence Commandes(idCommande)  
FK : état(idCommande) référence Commandes(idCommande)  
FK : état(idStatus) référence Statuts(idStatus)

FK : clients(parrain) référence Clients(mail)

## Contraintes :

Pour respecter le cahier des charges, des attributs de certaines tables doivent contenir des contraintes pour éviter que les données soient incohérentes :

- Dans la table « **commandes** » :  
 - Une note doit être comprise entre 0 et 5  
 - Une note et un commentaire peuvent être NULL si le client n’a pas laissé d’avis sur la commande

- Dans la table « **plats** »  
 - Un plat a forcément un nom (attribut « nom » en NOT NULL)  
 - Le prix d’un plat ne peut être inférieur à 0.

- Dans la table « **clients** »  
 - Les points de fidélités sont par défaut à 0 et ne peuvent pas être inférieur à 0.

- Dans la table « **carte**\_**bancaire** »  
 - Tous les attributs de la carte sont des char(50) car ils seront chiffrés.

- Dans la table « **horaires** »  
 - Un jour doit être un entier entre 0 et 6 pour les 7 jours de la semaine  
 - L’ouverture ne peut être inférieur à la fermeture pour des raisons de cohérence

- Dans la table « **restaurants** »   
 - L’attribut « fraisCharge » ne peut être inférieur à 0 car un frais ne peut être négatif.

Lors de la création des tables, ces contraintes sont spécifiées en SQL (voir le fichier pour implémenter la base de données .sql).

# Maquette du site WEB