Le fonctionnement et la Structure de notre base de données est la suivante :  
  
Table : custommer, cuisinier et utilisateurs

Puisqu’un cuisinier peut aussi être client nous avons choisi de les regrouper dans une table utilisateurs et d’ajouter des Héritages.

Nous savons **qu’il y à une relation 1,1-1,1 entre les tables utilisateurs et custommer** et idem pour cuisinier, **mais pour des raisons de clarité** du MCD nous avons choisi de **ne pas mettre de relations réflexive** de la table utilisateur vers elle-même comme vu en cours.   
  
Les tables custommer et cuisinier héritent donc de tous les attributs de la table utilisateur, elles ont uniquement un attribut, **id** chacune pour identifier chaque client et cuisinier.   
  
Table avis  et Commande:  
  
La table avis à un liens avec la table commande puisqu’un avis porte sur une commande. Elle a tous les attributs nécessaires à un avis. Le client et le cuisinier font un retour et le client donne un nombre d’étoile aux plats reçu.

La table commande est centrale dans notre base de donnée. Le client passe une commande préparer par le cuisinier. Cette commande comporte des lignes.

table ligne de commande et livraison :

Chaque ligne de commande à 1 seule livraison et chaque livraisons à au moins 1 ligne de commande.

Les livraisons ont un id, une date et heure de livraison enregistré au moment de la livraisons

Tables plat, pays et ingrédients :

Chaque plat contient n ingrédient et chaque ingrédient est contenu dans n plats.

De plus dans l’association contient nous avons ajouter une colonne quantité, comme cela chaque plat à une quantité précise pour chaque ingrédients de sa recette.   
  
La table Pays sert à éviter de dupliquer des données récurrentes, puisque les pays sont connus à l’avance.   
  
Les plats ont une nationalité/origine et les ingrédients une provenance d’importation/fabrication , d’où leurs liens avec la table pays.   
  
L’attribut prix\_Kg sera utile plus tard pour calculer le cout en matière première de chaque plats et ainsi fixé le prix de vente du plat.

Partie c sharp :

Nous avons divisé notre code en 4 classes différentes : la classe Lien, la classe noeud la classe Graphe et la classe affichagegraph.

* La classe nœud : elle représente un sommet et ses différentes connexions.
* La classe Lien : la classe Lien modélise une arête reliant deux sommets différents
* La classe Graphe : Cette classe permet de gérer l’ensemble du graphe avec ses différentes structures de données et algorithmes la composant.
* La classe affichagegraph : Cette classe va permettre la modélisation et la conception visuelle du graphe créer.

Le graphe va être représenté de deux façons différentes :

D’abord sous forme de listesAdjacentes : Une structure sous forme de tableau de listes, où chaque sommet est relié à une liste de ses voisins directs. Cette représentation va être celle que nous utiliserons principalement notamment dans l’exploration du graphique car elle est très efficace dans l’exploration des voisins de chaque sommets ainsi que l’ajout de nouvelles connexions entre eux (les liens).

On représente également le graphique sous forme de MatriceAdjacentes : Une matrice carrée où une valeur 1 indique une connexion entre deux sommets et 0 une absence de lien. Ainsi cette représentation permet un accès rapide et simplifié aux relations entre différents sommets.

Nous avons également déclaré une variable Nombredesommets du graphe qui permettra de construire le graphe en fonction du nombre de sommets demandé

Pour ce qui est de l’instanciation du graphique , nous l’avons fait ici à partir d’un fichier texte, où chaque ligne contient une relation entre deux sommets.

Ainsi nous avons créer la méthode ChargerDepuisFichier(string chemin) qui lit ces données et construit le graphe sous les deux formes mentionnées.