

Note technique explicative

Projet BDD – Joyce LAPILUS, TD B A3 Promo 2025

Organisation de la base de données

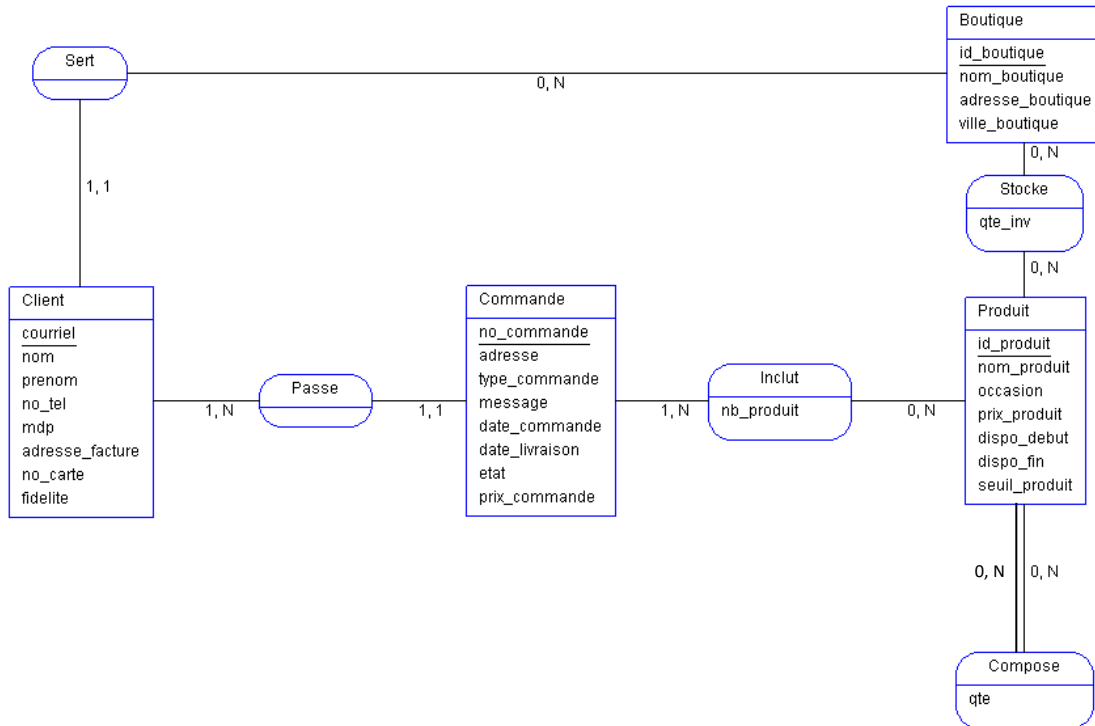


Figure 1 : Modèle E/A du projet

J'ai choisi qu'un client ne pouvait être inscrit qu'auprès d'une seule boutique à des fins de simplifications de la conception de la base de données ; étant donné que je me suis occupée de la création des fichiers csv à la main. Cette décision aide également à la simplification de la gestion des commandes (quel stock décrémenter lors de la complétion d'une commande).

On a également une association réflexive au niveau de l'entité « Produit » car, dans mon projet, les fleurs, accessoires et les bouquets sont tous considérés comme des produits, étant donné que les clients peuvent tous les acheter. Or, un bouquet standard est composé de fleurs, donc il a fallu trouver une solution pour stocker ces compositions florales standards. Cette solution ne pose pas de problème car les produits ont des identificateurs différents selon leur catégorie ; il commence par un « A » pour un accessoire, un « F » pour les fleurs, et un « B » pour les bouquets. Ce modèle E/A donne lieu au modèle relationnel suivant :

Boutique (id_boutique : int, nom_boutique : varchar, adresse_boutique : varchar, ville_boutique : varchar)

Client (courriel : varchar, nom : varchar, prenom : varchar, no_tel : varchar, mdp : varchar, adresse_facture : varchar, no_carte : varchar, fidelite : varchar, #id_boutique : int)

Commande (no_commande : varchar, adresse : varchar, type_commande : varchar, message : varchar, date_commande : date, date_livraison : date, etat : varchar, prix_sans_reduc : decimal, prix_commande : decimal, #courriel : varchar)

Produit (id_produit : varchar, nom_produit : varchar, occasion : varchar, prix_produit : decimal, dispo_debut : int, dispo_fin : int, seuil_produit : int)

Inclut (#no_commande : varchar, #id_produit : varchar, nb_produit : int)

Compose (#id_bouquet : varchar, #id_composant : varchar, qte : int)

Stoque (#id_boutique : int, #id_produit : varchar, qte_inv : int)

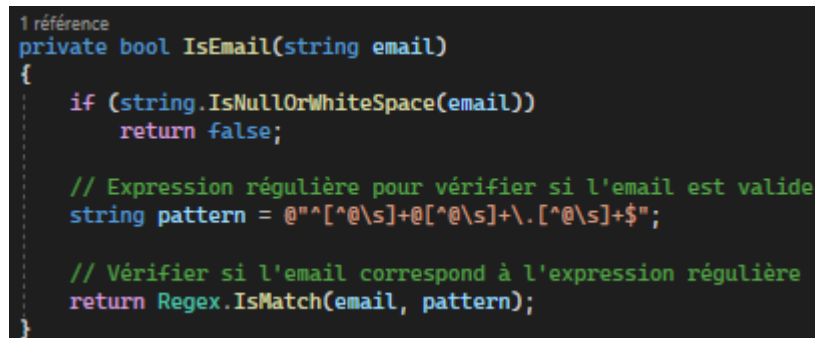
Options de codage

Innovations apportées, etc.

En ce qui concerne la modification de plusieurs tables, notamment *produit*, *commande*, on ne peut pas changer la boutique d'appartenance. De même pour les commandes et les bouquets, pour lesquels il n'est pas possible de modifier le contenu de ces derniers une fois créés. D'un point de vue technique, retrouver les contenus de ces derniers, voir lesquels ont été supprimés comparés à ceux qui ont été ajoutés paraissait trop compliqué pour mon niveau en C#, voire également en SQL.

Le stock des différents produits étant géré par une table séparée et différent pour chaque boutique ouverte, j'ai décidé qu'il était plus facile, lors de la modification d'un produit, de séparer la gestion du stock (et que l'on générerait celui de la boutique sélectionnée) et la gestion des informations communes à toutes les boutiques (le nom du produit, son prix, etc.) en (dé)cochant une *CheckBox*.

Lors de la saisie d'informations pour les différents formulaires (surtout pour la table *client*), l'utilisation de *Regex* a été sollicitée pour vérifier les formats des numéros de téléphone, des numéros de carte bancaire et des adresses électroniques. *Regex* a été utilisé dans trois méthodes booléennes, si l'une d'elles retourne *false*, la modification ou l'ajout d'un client ne sera pas abouti(e).

A screenshot of a code editor showing a C# method named `IsEmail`. The code is as follows:

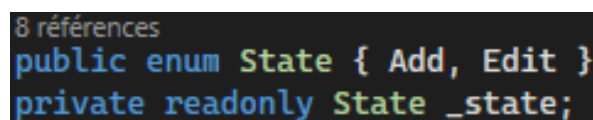
```
1 référence
private bool IsEmail(string email)
{
    if (string.IsNullOrEmpty(email))
        return false;

    // Expression régulière pour vérifier si l'email est valide
    string pattern = @"^[^@\s]+@[^@\s]+\.[^@\s]+$";

    // Vérifier si l'email correspond à l'expression régulière
    return Regex.IsMatch(email, pattern);
}
```

Figure 2 : Exemple de l'utilisation de *Regex* dans *CustomerForm.cs*

Également, pour ces formulaires, qui sont utilisés pour la modification ainsi que l'ajout, j'ai utilisé un *type enum* pour la plupart d'entre eux. Les *membres enum* sont *Add* (pour l'ajout d'une ligne d'une table) et *Edit* (pour la modification d'une ligne d'une table) ; ce qui facilite la gestion des contrôles dans un cas ou dans l'autre. Le même processus est utilisé pour les différents produits ; les bouquets et les autres (fleurs et accessoires).

A screenshot of a code editor showing a C# `enum` definition. The code is as follows:

```
8 références
public enum State { Add, Edit }
private readonly State _state;
```

Figure 3 : Exemple de l'utilisation du *type enum* dans *BouquetForm.cs*

Développement(s) complémentaire(s)

L'export des clients n'ayant pas commandé depuis plus de six mois en JSON, même si optionnel, a été effectué. Les attributs choisis dans la table *client* pour cet export sont seulement le nom, le prénom et l'adresse électronique du client. J'estime que nous n'avons besoin que d'identifier le client, et d'avoir un moyen de le contacter, dans ce cas les attributs mentionnés précédemment seraient suffisants.

```
1  [
2    {
3      "Courriel": "abc@gmail.com",
4      "Nom": "Kim",
5      "Prenom": "Hongjoong"
6    },
7    {
8      "Courriel": "contact@sment.com",
9      "Nom": "Jung",
10     "Prenom": "Jaehyun"
11   },
12   {
13     "Courriel": "ghi@gmail.com",
14     "Nom": "Jung",
15     "Prenom": "Yunho"
16   },
17 ]
```

Figure 4 : Aperçu de l'export en JSON

Lors de la création d'un bouquet personnalisé, seulement les produits disponibles sont affichés et peuvent donc être ajoutés au contenu de la commande.

Les statistiques sont accessibles par période ; il y a des périodes à disposition (les sept derniers jours, les trente derniers jours, le mois courant et les six derniers mois) mais si ces dernières ne correspondent pas aux besoins de l'utilisateur, il peut également personnaliser la période voulue. Par défaut, lors de l'ouverture du module « Statistiques », la période affichée est celle des trente derniers jours.

lundi 17 avril 2023	mercredi 17 mai 2023	Personnalisé	7 derniers jours	30 derniers jours	Ce mois-ci	6 derniers mois
---------------------	----------------------	--------------	------------------	-------------------	------------	-----------------

Figure 5 : Sélection des périodes pour les statistiques

Finalement, le formulaire principal ne possédant pas de bordures « natives », il a fallu trouver un autre moyen de pouvoir laisser l'utilisateur bouger le formulaire à sa guise (voir la région « Drag form without borders » dans *MainForm.cs*).