Rapport Projet BDD

SEGUI Matthieu, SCHNEEGANS Hugo

Github du projet

TABLE DES MATIÈRES

I. INTRODUCTION

VéloMax est une compagnie qui vend des bicyclettes et des pièces de rechange pour les bicyclettes. Étant donné la croissance énorme du volume des ventes dans les dernières années, Max Legrand, l'unique propriétaire de VéloMax, désire avoir une nouvelle base de données afin de mieux gérer les ventes et les clients.

Nous, SEGUI Matthieu et SCHNEEGANS Hugo, sommes donc les élèves ingénieurs en charge de réaliser cette base de données regroupant toutes les fonctionnalités du cahier des charges pour le compte de M. Legrand.

II. SCHÉMA ENTITÉ / ASSOCIATION

Nous avons débuté notre travail en réalisant un schéma Entité / Association sur le logiciel Looping. Il permet de mieux visualiser la forme de la base de données que nous allions créer sur MySQL par la suite. Nous avons repris tous les éléments du cahier des charges, à savoir les ventes, les fournisseurs et les clients pour créer le schéma (Annexe 1 et 2).

III. LE CODE

A. *Script MySql*

Viens ensuite la traduction de ce schéma Entité / Association sur MySQL, pour créer la base de données. Nous avons utilisé les connaissances que nous avons développé lors des différents TD de Base de données et Interopérabilité. Lors de la création des différentes tables de la base, nous avons pris soin de bien inscrire les clés primaires et secondaires car cela sera utile pour nos requêtes par la suite. Nous avons aussi peuplé les tables avec les informations qui nous ont été donné dans le cahier des charges, avec notamment les vélos et les pièces qui les composent. Nous avons aussi créé différents clients, fournisseurs et commandes pour simuler l'utilisation de notre base de données en situation réelle.

B. *De MySql à C#/WPF*

Afin de créer une réelle « application » pour M. Legrand, nous avons utilisé les notions que nous avons apprises en Interopérabilité afin d'ouvrir une connexion entre notre base de données sur SQL, et avec Visual Studio 2022 pour nous permettre de faire une application WPF.

WPF (Windows Presentation Foundation) est un outil graphique de .NET utilisé pour créer des applications. Pour afficher nos tables, nous utilisons des Datagrid (tableaux) WPF. Ils nous permettent de modifier directement les cellules voulues. Nous avons rencontré un léger problème lors de la modification des cellules. En effet, lorsque l'évènement de fin de modification est déclenché, l'utilisateur est déjà sur une autre cellule. De ce fait, nous n'avons pas réussi à récupérer la valeur de la cellule qu'il a modifié. Pour pallier ce problème, l'utilisateur doit valider 2 fois sa valeur.

Nous avons aussi pour toutes les tables, un bouton d'ajout de ligne, de suppression et d'export en Json.

Nous avons aussi bloqué certaines tables en ReadOnly (par exemple la table d'affichage des abonnements Fidélio).

Les Datagrid peuvent aussi être triées pour faciliter la lecture des données.

IV. FONCTIONNALITÉS

Nous avons implémenté dans notre code C toutes les fonctionnalités demandées dans le cahier des charges. Cela regroupe :

- La création, suppression, et modification de pièces de vélo.
- La création, suppression, et modification de modèles de vélo.
- La création, suppression, et modification de fournisseurs.
- La création, suppression, et modification de clients professionnels.
- La création, suppression, et modification de clients particuliers.
- La création, suppression, et modification de commandes de pièces et de vélos, ainsi que la possibilité de voir les commandes et directement leur contenu.
- La gestion du stock, avec la possibilité d'afficher les stocks de pièces et de vélos en fonction du nom des pièces, des fournisseurs, des modèles de vélos, des catégories de vélos, et des types de vélos. M. Legrand est aussi informé d'une petite notification s'il manque n'importe quelle pièce ou n'importe quel modèle de vélo dans son stock.
- L'implémentation d'un module statistique, permettant de voir à tout moment la quantité de vélos et de pièces vendue, le nombre de membres des différents programmes d'adhésion Fidélio, les dates d'expiration de leur abonnement, les meilleurs clients rangés par le prix total de leurs commandes, le prix moyen des commandes, le

nombre moyen de pièces par commandes de pièces et le nombre moyen de vélos par commande de vélos.

Nous avons aussi ajouté 3 requêtes complexes SQL :

— Une requête union qui permet d'afficher toutes les pièces et modèles commandés, la commande associée et la quantité sans avoir à séparer les commandes de pièces et de vélos.

— Une requête d'auto-jointure qui permet d'afficher une pièce ainsi que tous les modèles de vélos que cette pièce sert à fabriquer.

— Une requête synchronisée qui permet d'afficher tous les noms et prénoms des clients individuels qui ont commandé une pièce en particulier.

Nous avons enfin rajouté la possibilité pour M. Legrand d'exporter n'importe quelle table de sa base de données au format JSON, en ajoutant un bouton « Exportation » dans son application.

V. ANNEXES

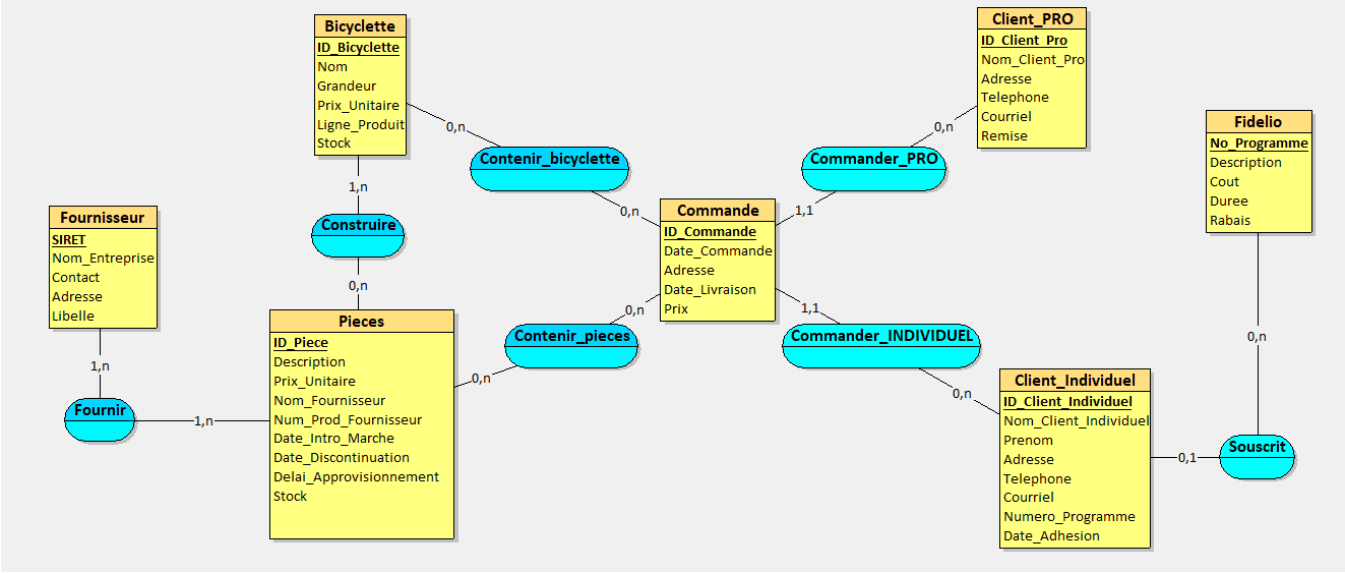

FIGURE 1 – Schema E/A

Bicyclette = (**ID_Bicyclette** INT, Nom VARCHAR(50), Grandeur VARCHAR(50), Prix_Unitaire DECIMAL(15,2), Ligne_Produit VARCHAR(50), Stock INT);
Pieces = (**ID_Piece** INT, Description VARCHAR(50), Prix_Unitaire DECIMAL(15,2), Nom_Fournisseur VARCHAR(50), Num_Prod_Fournisseur INT, Date_Intro_Marche DATE, Date_Discontinuation DATE, Delai_Approvisionnement INT, Stock INT);
Fournisseur = (**SIRET** INT, Nom_Entreprise VARCHAR(100), Contact VARCHAR(50), Adresse VARCHAR(50), Libelle INT);
Client_PRO = (**ID_Client_Pro** INT, Nom_Client_Pro VARCHAR(100), Adresse VARCHAR(150), Telephone VARCHAR(20), Courriel VARCHAR(100), Remise INT);
Fidelio = (**No_Programme** INT, Description VARCHAR(50), Cout INT, Duree INT, Rabais INT);
Client_Individuel = (**ID_Client_Individuel** INT, Nom_Client_Individuel VARCHAR(100), Prenom VARCHAR(100), Adresse VARCHAR(150), Telephone VARCHAR(20), Courriel VARCHAR(100), Numero_Programme INT, Date_Adhesion DATE, *#No_Programme*);
Commande = (**ID_Commande** INT, Date_Commande DATE, Adresse VARCHAR(50), Date_Livraison DATE, Prix REAL, *#ID_Client_Individuel, #ID_Client_Pro*);
Construire = (*#ID_Bicyclette, #ID_Piece*);
Contenir_bicyclette = (*#ID_Bicyclette, #ID_Commande*);
Contenir_pieces = (*#ID_Piece, #ID_Commande*);
Fournir = (*#ID_Piece, #SIRET*);

FIGURE 2 – Schema relationnel