DENIZOT Roland TD-D
DANGLEJAN Thomas

Rapport de projet Base de données et Interopérabilité

1- Résumé du Projet

Dans le cadre du cours de Base de données et interopérabilité, nous avons dû créer une base de données concernant Monsieur Legrand, un vendeur de vélos, afin de l'aider à mieux gérer ses commandes, ses programmes de fidélité, etc. De plus, nous devions lier cette base de données avec C# afin de pouvoir interagir plus facilement avec et gérer les données en temps réel avec une application console.

Nous sommes Thomas DANGLEJAN et Roland DENIZOT et nous allons expliquer dans ce rapport la structure et nos décisions concernant la base de données que nous avons créé.

2- Organisation de notre base de données

Notre base de données est constituée de 12 tables nommées : Bicyclette, Liste_Assemblage, Constitution, Piece, Approvisionnement, Fournisseur, Commande_vélo, Commande, Commande_pièce, Boutique_entreprise, Client_particulier et Programme_fidélio. Certaines tables telles que « Constitution » ne servent que de lien entre deux autres tables pour les requêtes SQL en raison des cardinalités sur le schéma Entité/Association.

Les tables « Commande_vélo » et « Commande_pièce » sont directement liées à la table « Commande » mais sont essentielles au bon fonctionnement de notre base de données puisqu'elles indiquent les nombres de chaque item à commander, de même pour la table « Approvisionnement » qui donne des informations essentielles concernant les fournisseurs et les pièces.

Comme nous ne pouvons pas avoir plusieurs fois le même nom de clé étrangère, nous avons dû dans certains cas ajouter un chiffre à la fin de cette clé afin que toute notre base de données fonctionne correctement, mais celles-ci contiennent les mêmes types d'informations que les clés sans chiffre (exemple : « numero_commande » et « numero_commande2 »).

Les différentes tables de la base données ont été remplies à la main avec les informations contenues dans le sujet en plus d'informations provenant de notre imagination. La base de données est donc constituée de :

- 15 Commandes
- 20 Commandes de pièces
- 15 Commandes de vélos
- 66 Pièces
- 16 Fournisseurs
- 15 Clients particuliers
- 10 Boutiques entreprises
- 15 Listes d'assemblage

- 15 Bicyclettes
- 4 Programmes Fidélio
- 78 Approvisionnements
- 147 Constitutions

3- Options de codage

Tout d'abord, l'essentiel des requêtes et des affichages ont été fait selon ce que nous avons vu en cours, notamment via les supports de cours avec en plus de cela une interaction avec l'utilisateur pour chaque fonctionnalité au lieu d'une insertion en dur dans le programme.

De plus, notre programme est verrouillé par un identifiant et un mot de passe caché comme pour l'entrée d'un code confidentiel sur un automate de banque par exemple. L'utilisateur ayant tous les droits a pour identifiant : root, mot de passe : root et l'utilisateur ne pouvant pas modifier la base de données mais juste consulter les données a pour identifiant : bozo, mot de passe : bozo.

Par la suite, nous n'avons pas créé d'approvisionnement de vélos mais juste de pièces car nous considérons que Monsieur Legrand construit les vélos à partir des pièces car nous avons compris dans le sujet que les fournisseurs ne fournissaient que des pièces.

Dans le cas où un client tente d'acheter plus de pièce que la quantité de cette dernière en stock, la date de livraison de la commande sera actualisée en fonction du temps d'approvisionnement de la pièce de la part du fournisseur.

Concernant la quantité en stock des pièces et des vélos, nous considérons que cette quantité sera retirée du stock lors de l'expédition de la commande, c'est pourquoi la quantité n'est pas retirée immédiatement lors d'une commande, car à tout moment le client peut annuler sa commande avant l'expédition, ce qui implique que la pièce n'est pas retirée du stock.

Nous avons incorporé quelques fonctionnalités bonus dans le programme comme par exemple la possibilité de revenir en arrière lors de l'affichage du module statistique concernant les différents stocks de Monsieur Legrand, ce système fonctionne comme un livre où nous pouvons tourner les pages comme bon nous semble grâce à l'utilisation de LinkedList.

Certaines parties étaient plus difficiles que d'autres, notamment lorsqu'une pièce manquait dans une livraison, il a été particulièrement difficile pour nous de recalculer la nouvelle date de livraison. Nous avons aussi dû changer plusieurs fois notre base de données initiale car certaines clés manquaient ou nous empêchaient de faire ce que nous voulions. Au contraire, les parties concernant les ajouts, modifications et suppressions de données n'ont pas été trop difficiles pour nous.

Nous n'avons pas implémenté de WPF car nous étions à l'étranger au premier semestre et le délai entre le TD concernant le WPF et le rendu final ne nous permettait pas de faire une implémentation correcte de ce dernier.

Nous avons codé le premier mode démo demandé, l'utilisateur peut donc visualiser le nombre de clients, le nom des clients avec le cumul de ses commandes en cours, la liste des produits ayant une quantité en stock inférieure ou égale à 2, le nombre de pièces fournies par fournisseur et l'export en XML et JSON. Le tout à la suite en appuyant sur n'importe quelle touche.

De plus, il est important de noter que le projet utilise des références à ajouter manuellement telles que MySql.data et des packages tels que Newtonsoft.Json. Sans cela, il ne pourra pas se lancer.

En ce qui concerne l'export en XML et Json, le programme exporte tout d'abord la table en XML sur le modèle présenté en cours et dans les TD. Par la suite, le fichier XML est converti en Json via une fonction JsonConvert qui transforme les informations en string puis en document Json.

Finalement, nous avons réussi à réaliser presque tout ce qui était demandé dans le cahier des charges, hormis le WPF.

4- Compétences acquises

Durant ce projet, nous avons pu acquérir de nombreuses compétences. Nous avons expérimenté une consigne réaliste qui peut réellement arriver dans le monde du travail, donc nous avons développé la compétence d'adaptation d'un projet selon la volonté exacte d'une personne extérieure au projet. La deuxième compétence principale que nous avons développée est l'autonomie, c'est-à-dire que nous avons appris à aller chercher nous-même les informations qui permettent de résoudre la majorité des problèmes que nous rencontrions en C# ou en SQL.