GABOT Quentin
FERNANDES William

TD P

Rapport du projet

Table des matières

1) Relations entre SQL et C#

Notre projet est une réflexion sur une problématique de base de données et d'interopérabilité : nous avons donc dû traiter au cours de notre projet la gestion d'une base de données sur MySQL Workbench tout en implémentant un outil de gestion à ladite base de données en C#.

Après avoir établi un modèle Entité/Association et généré les tables correspondantes sur MySQL Workbench, la question la plus cruciale s'est portée sur la gestion des relations entre les tables et surtout de la manière d'établir un pont de communication entre la base de données (MySQL) et le centre des opérations (C#).

Nous avons alors décidé de procéder de la sorte : toutes les tables de MySQL sont versées dans des listes d'objets correspondants en C#. Exemple : la table 'Recette' de MySQL possède un équivalent en C#, la classe Recette. Au lancement du programme, les éléments de la table Recette sont tous insérés dans une liste d'objet Recette : on a ainsi fait le lien de SQL vers C#.

Il est également essentiel de faire le chemin retour, soit le versement des listes d'objets en C# vers SQL. En effet, les modifications de la base de données se font en C# mais le stockage se fait en SQL ; il est donc essentiel de pouvoir enregistrer les ajouts, retraits et altérations faites sur les éléments des listes en C#. Il est à noter que nous avons pris le parti d'exporter les fonctions d'opérabilités sur C#, nous ne sollicitons donc pas la base de données en dehors de l'ouverture et la fermeture de l'application.

Une fois les problématiques liées à l'interopérabilité évacuées, il est essentiel de se pencher sur les questionnements soulevés par la base de données et plus particulièrement les différences fondamentales entre les liaisons des tables en SQL et celles des classes en C#. Nous avons donc transformé la majorité des associations SQL en listes C# afin de simuler les liaisons entre les entités SQL devenues classes C#.

A noter que certaines associations dérogent à la règle, telles que les associations Composer et Commander qui se sont finalement traduites par des listes et des classes C#. Les listes servent les liaisons basiques entre les entités partageant une connexion alors que les classes permettent de réaliser des opérations plus pointues et surtout de pouvoir enregistrer ces données en SQL. En effet, si les listes sont enregistrées en tant qu'éléments de listes traduisant des entités SQL, certaines associations doivent conserver un suivi après la fermeture de l'application.

En définitive, le modèle Entité/Association propre à SQL ne se traduit pas tel quel en C#. Il est nécessaire d'y apporter des modifications afin de servir l'objectif premier de l'utilisation de C# : l'opérabilité. Il est donc évident que des ponts de communication entre C# et SQL sont nécessaires afin de jongler entre base de données et opérations.

2) Problèmes rencontrés

Le principal problème rencontré lors de la mise en place du projet fût la traduction des concepts du modèle Entité/Association en C#. En effet, SQL étant un langage d'interrogation des données et C# un langage de programmation orienté objet, le lien entre ces deux univers ne fût pas tout de suite évident ! Nous avons fait plusieurs tentatives au niveau de la structure de notre programme C# avant de trouver ce qui allait devenir la version finale de notre projet.

Un autre contretemps au sein du projet (bien que nous ne puissions décemment pas appeler ceci un problème) fût d'utiliser dans un premier temps des fichiers CSV pour modéliser la base de données ! Cette décision entraina l'utilisation de trois plateformes : la base de données sous MySQL Workbench, la base de données sous Excel et finalement le programme sous C#. Finalement, il est clair que toute ceci était loin d'être nécessaire puisqu'après réflexion, nous avons juste retirer les fichiers CSV de l'équation sans que notre application s'en trouve bouleversé.

En vérité, cette erreur est survenue d'une incompréhension quant à l'utilisation de MySQL Workbench : nous pensions que SQL servait de structure à la base de données qui devait être externalisée… Nous nous sommes rapidement

rendu compte de notre fourvoiement, et après rectifié le coche nous pouvons affirmer que la compréhension de SQL dans sa globalité nous semble simplifiée.

3) Axes d'amélioration et réussites

Globalement, les enjeux du sujet ont été compris, assimilé et remis en perspective des enjeux plus globaux de la gestion de base de données et de l'interopérabilité. Nous avons nettement gagné en compétence via la réalisation de ce projet : une problématique de même nature serait traitable très facilement au vu des enseignements tirés.

Plus encore, ce projet a éveillé un vrai intérêt chez nous en ce qui concerne l'interopérabilité puisque c'était bien la première fois que nous utilisions les atouts de plusieurs langages et logiciels en même temps et surtout ensemble afin de présenter un rendu se rapprochant d'un résultat professionnel.

En ce qui concerne les axes d'amélioration, nous avons ciblé quelques petits temps morts au sein du projet qui nous ont un peu ralenti et empêché de pouvoir nous pencher sérieusement sur le traitement WPF. La plus grosse piste d'amélioration en ce qui nous concerne est donc l'utilisation de WPF afin de créer une interface graphique de qualité.