Gestion d'une librairie d'éditeur

I/ Expression du besoin

Le but de l'application à développer est de permettre à un éditeur de gérer facilement l'affichage de ses auteurs, les livres qu'ils ont publiés avec leurs dates de publication ainsi que le prix des livres dans des devises différentes.

L'ensemble de cette étude est donc basé sur une base de données MySQL nommée "***BddEditeur***" contenant :

- Une table de livres "BookList",
- Une table des auteurs des livres "BookAuthors"
- Et une table des prix des livres "BookPrices".

La table "**BookList**" contient la description des livres de l'éditeur avec pour chaque "tuple" le numéro ISBN, le titre et la date de publication.

La table "**BookPrices**" contient la description des prix des livres avec pour chaque "tuple" le numéro ISBN, la monnaie pour laquelle le prix est spécifié (un livre peut être commercialisé dans plusieurs états donc avec une monnaie différente), et le prix.

La table "**BookAuthors**" contient la liste des auteurs de l'éditeur avec pour chaque "tuple" le numéro ISBN, le prénom (Firstname) et le nom (LastName) de l'auteur.

La table "**Users**" contient la liste des utilisateurs de l'application autorisés à mettre à jour la base de données.

Configuration du système informatique

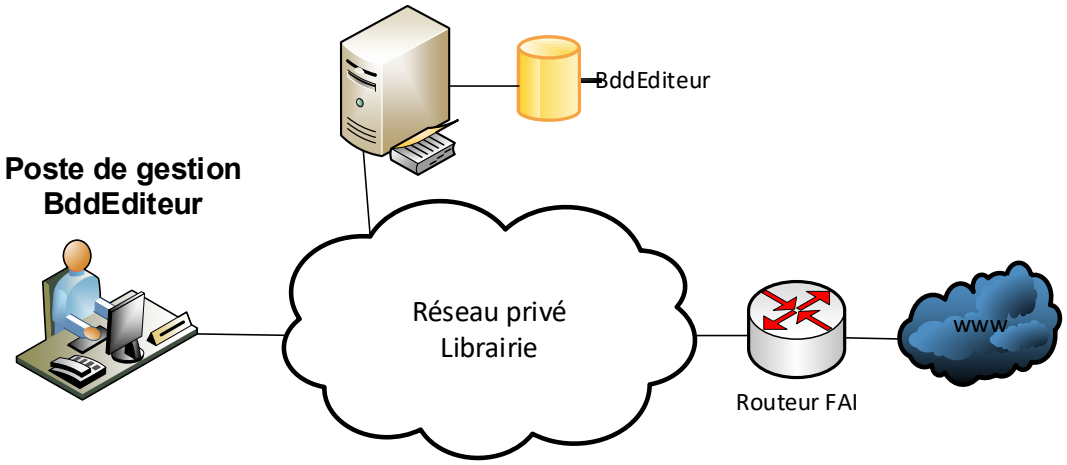
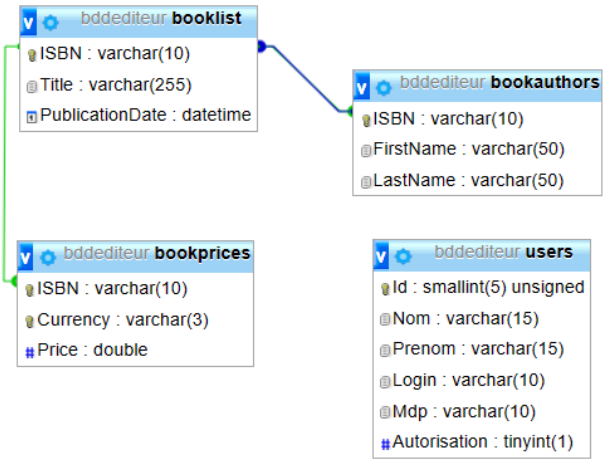
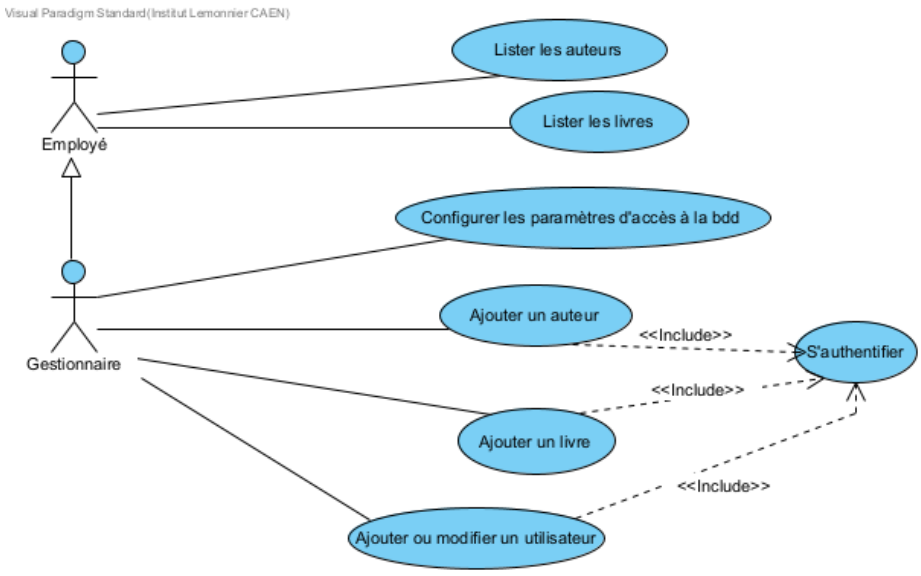


Schéma de la base de données :



La connexion à cette base se fera selon l'authentification MySQl avec le compte "***AdminEditeur***" et le mot de passe "***@Password1234 !***".

Cette application met en évidence les acteurs "Gestionnaire" autorisé à modifier le contenu de la base après authentification, et l'acteur employé destiné uniquement à afficher la liste des auteurs ou des livres s'il veut faire une recherche. Ces acteurs interviennent dans les scénarios suivants :



<u>Cas d'utilisation " Configurer les paramètres d'accès à la base de données" :</u>

Description : le gestionnaire configure les paramètres d'accès à la base de données grâce à une fenêtre dédiée accessible depuis un bouton du menu de l'application. Les paramètres sont :

1. L'adresse IPv4 du serveur de base de données,
2. Le port d'écoute du service base de données,
3. Le login du compte d'accès à la base de données,
4. Le mot de passe du compte d'accès à la base de données,

Cas d'utilisation "S'authentifier" :

Description : le gestionnaire doit être préalablement authentifié afin de pouvoir accéder à la gestion de la base de données. Pour cela il utilisera un login et un mot de passe dédié correspondant à un compte d'accès enregistré dans la table "*Users"* Une fois l'authentification validée les menus ou boutons permettant l'accès à la gestion de la bdd seront activés.

Cas d'utilisation "Ajouter un auteur" :

Description : le gestionnaire une fois authentifié, peut ajouter un nouvel auteur à la liste des auteurs.

Cas d'utilisation "Ajouter un livre" :

Description : le gestionnaire une fois authentifié, peut ajouter un nouveau livre à la liste des livres pour un auteur sélectionné.

Cas d'utilisation "Ajouter ou modifier un utilisateur" :

Description : le gestionnaire une fois authentifié, peut ajouter un nouvel utilisateur de l'application ou modifier un utilisateur déjà enregistré pour lui attribuer les droits de mise à jour da la base de données.

Cas d'utilisation "lister les auteurs" :

Description : le gestionnaire, ou un employé de la maison d'édition, peut visualiser la liste des auteurs.

Cas d'utilisation "lister les livres" :

Description : le gestionnaire, ou un employé de la maison d'édition, peut visualiser la liste des livres d'un auteur sélectionné dans la liste des auteurs.

II/ Contraintes de réalisation :

Environnement de développement : Visual Studio 2019 Entreprise.

Langages : C#, XAML, SQL ou LINQ.

Ressources logicielles : Visual Studio 2019 Entreprise, LINQconnect Express (Devart Software).

Ressources pédagogiques : Cours et TDs C#, WPF,

Documentations : AllDotBlog-Tome-7-XAML.pdf, AllDotBlog-Tome-3-LinqEF.pdf,

III/ Réalisation :

1. Installer la base de données sur xamp à l'aide des scripts fournis,
2. Créer une solution vide et la nommer "***SolutionEditeur***",

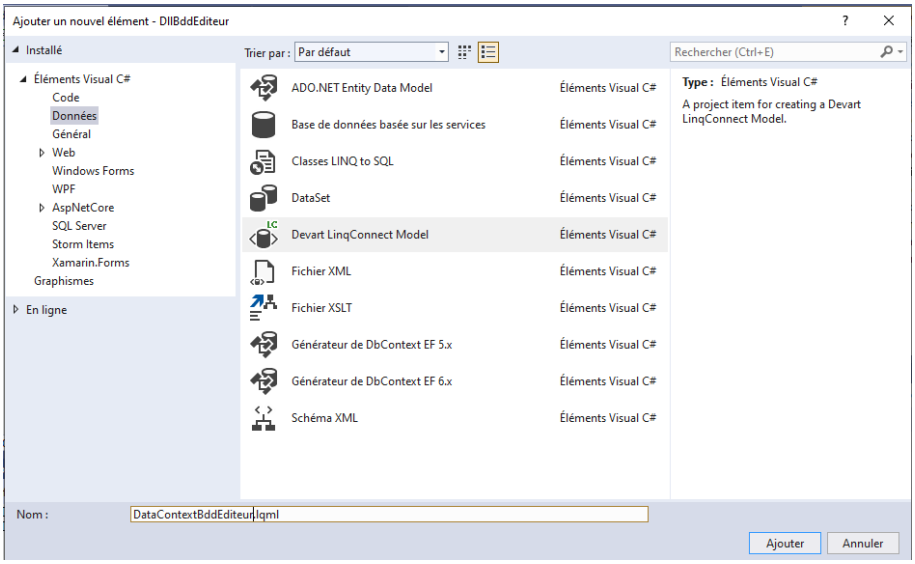
3. Ajouter à la solution "***SolutionEditeur***", un projet de type "Application WPF .Net Framework" et le nommer "***WpfAppEditeur***",
4. Ajouter à la solution "***SolutionEditeur***", un projet de type "Bibliothèque de classes .Net Framework" et le nommer "***DllBddEditeur***". Ce projet contiendra les classes crées automatiquement par l'outil ***Linq*** (conseillé) ou les classes crées explicitement pour accéder à la base de données.

LINQ to SQL Compatible ORM solution | LinqConnect (devart.com)

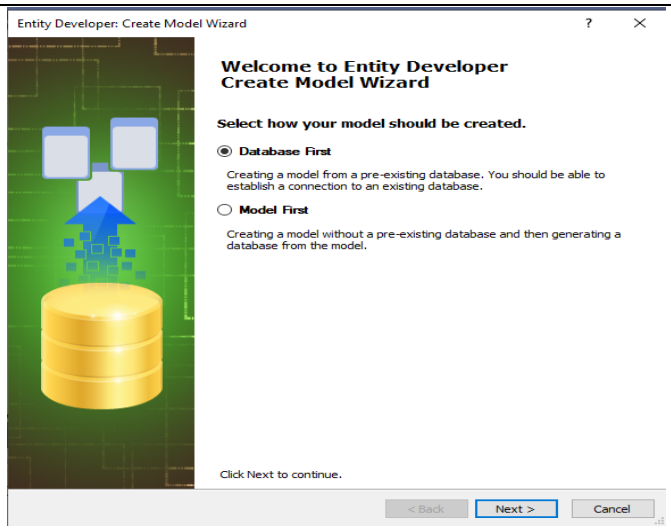
La version Linq Connect express est gratuite et prend en charge .Net Framework. Cette version est limitée à 10 entités cad 10 tables dans la base de données. (L'installation doit être faite avec Visual studio fermé). Choisir l'installation par défaut qui installe tous les outils nécessaires ainsi qu'une aide à l'utilisation de linq.

Pour installer "la relation" entre la base de données *BddEditeur* et *DllBddEditeur* il faut procéder comme suit :

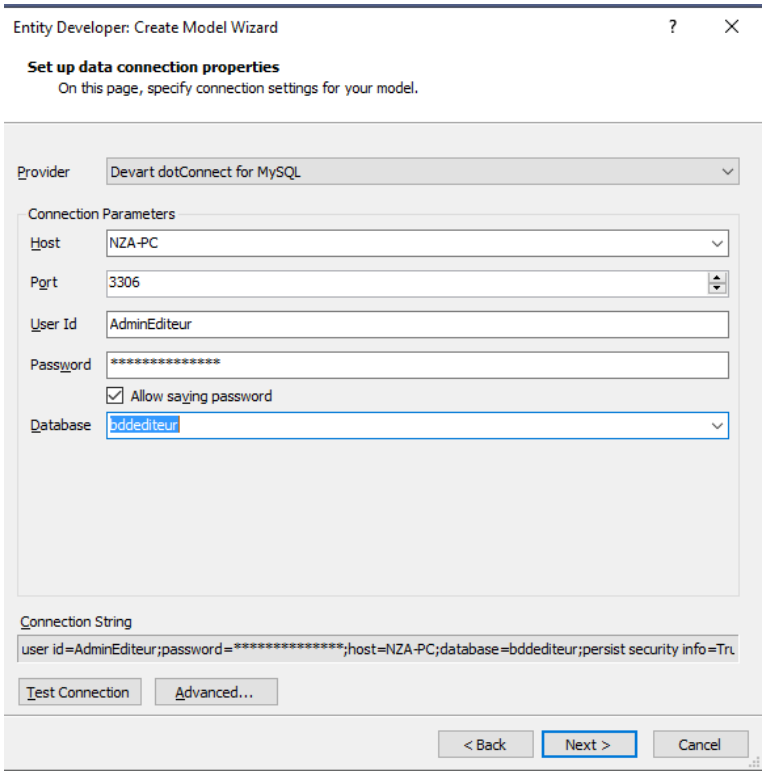
- Clic sur le projet DllBddEditeur → ajouter →nouvel element : choisir le modèle DevartLinqConnectModel et changer le nom en ***DataContextBddEditeur.lqml***



ensuite cliquer sur *Ajouter* :

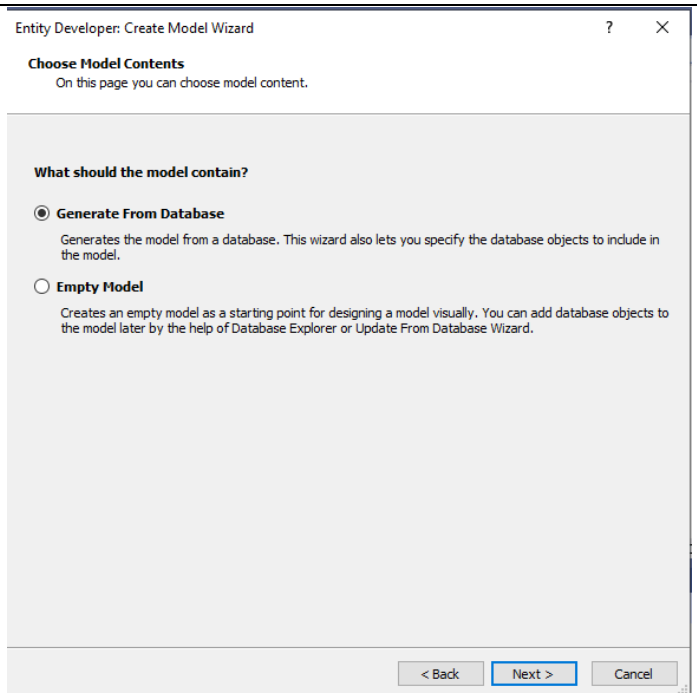


- Choisir *DataBase first* puisque la base *BddEditeur* est déjà créée.
- *Clic sur Next* et remplir les champs selon les paramètres du serveur et de la base de données :



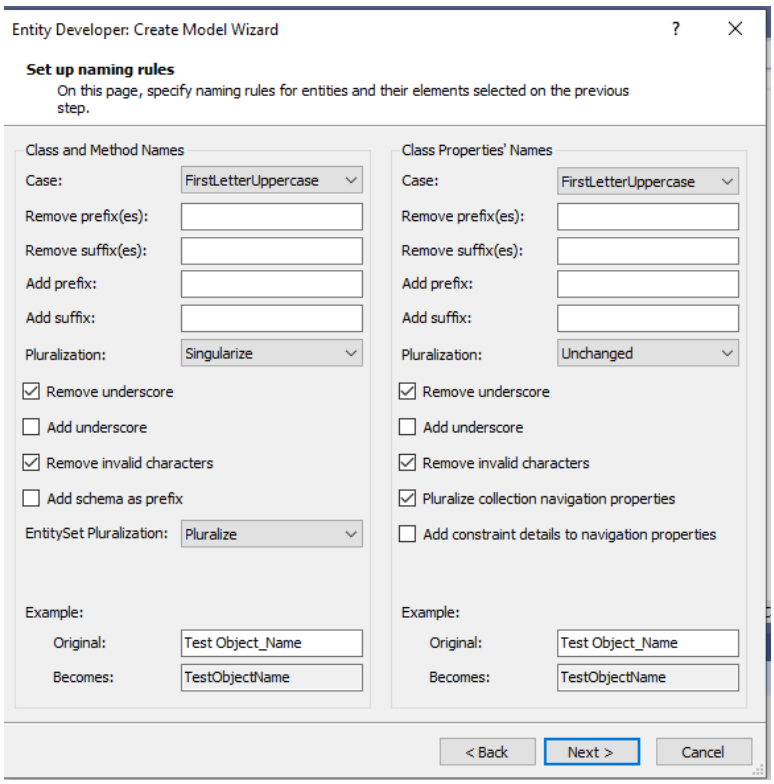
Le nom de l'hôte est celui de la machine *serveur*. On peut alors tester la connexion à la base de données avant de poursuivre.

Clic sur Next :



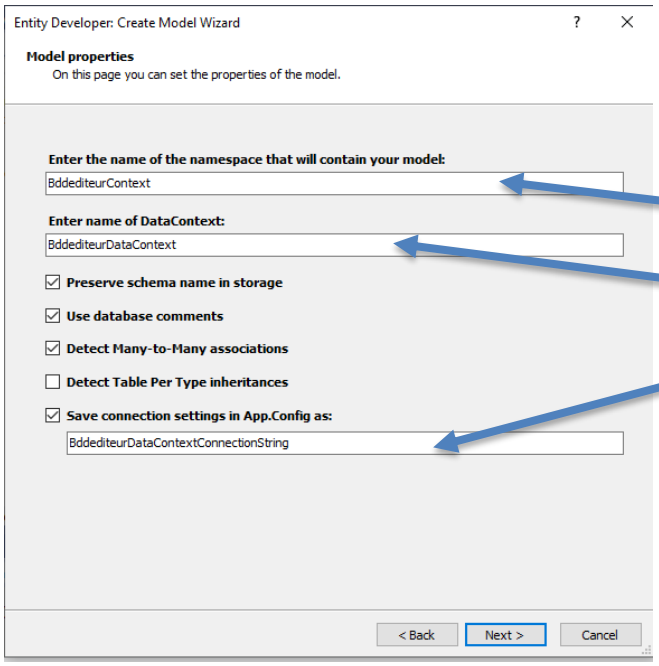
On peut alors choisir le modèle généré depuis la base de données ou choisir un modèle vide (non conseillé)

5. *Clic sur Next :* Linq recherche dans la BddEditeur les tables et permet de sélectionner celles qu'on souhaite intégrer au projet. On souhaite intégrer toutes les tables de la Bdd, il faut donc laisser la sélection telle et poursuivre.
6. *Clic sur Next :*



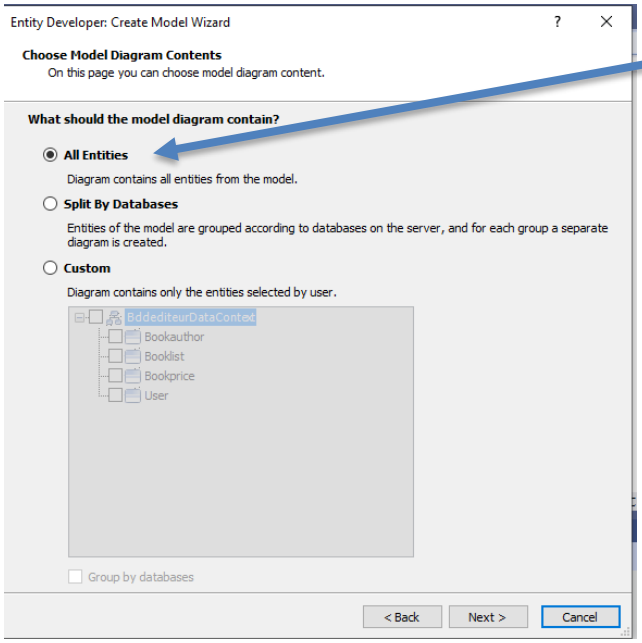
Sur cette fenêtre il est possible de modifier des noms, changer la casse etc. sans rien modifier pour reprendre les mêmes noms des tables de la Bdd, on poursuit.

7. *Clic sur Next :*



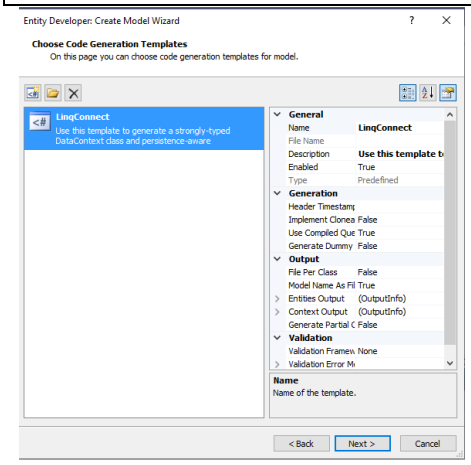
Linq crée un *namespace* ***BddediteurContext*** et crée dedans une classe spécifique ***BddediteurDataContext*** et enfin la chaine de connexion à la BddEditeur ***BddediteurDataContextConnectionString***

8. *Clic sur Next :*



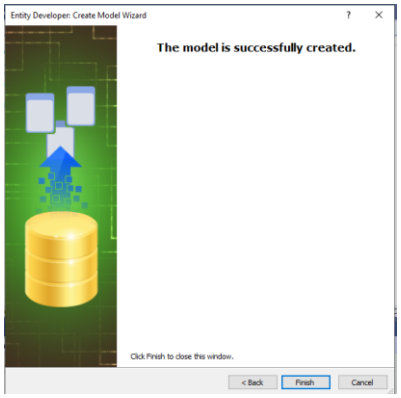
On choisit un diagramme de classes avec toutes les tables qui ont été choisies précédemment

9. *Clic sur Next :*



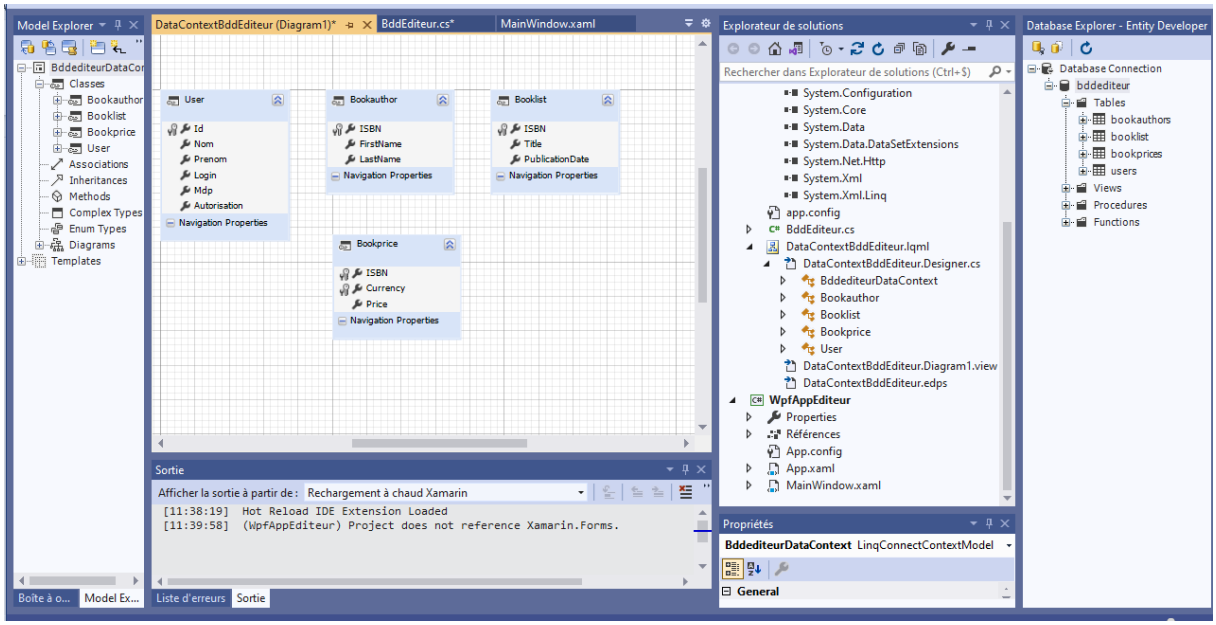
A cette étape, linq donne des informations sur le code qu'il va générer

10. On poursuit : clic sur Next :

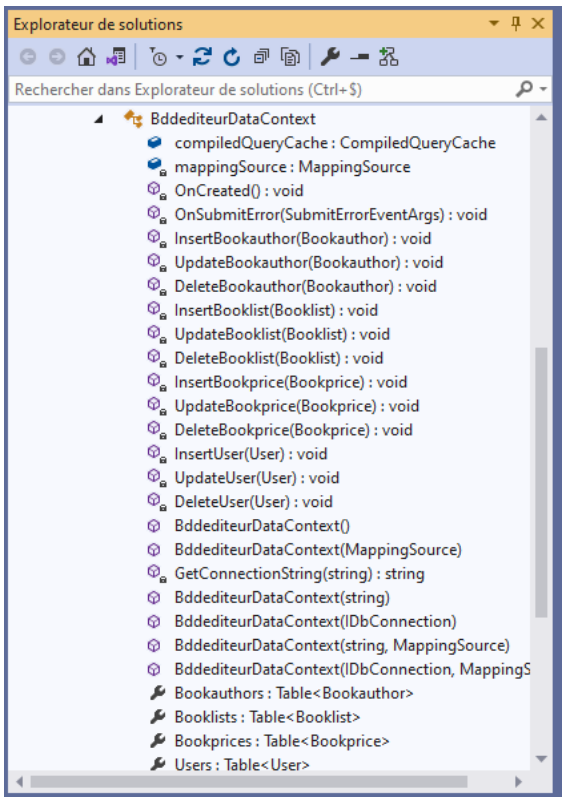


On poursuit en cliquant sur Finish

A cette étape, on peut voir toutes les classes qu'il a générées :

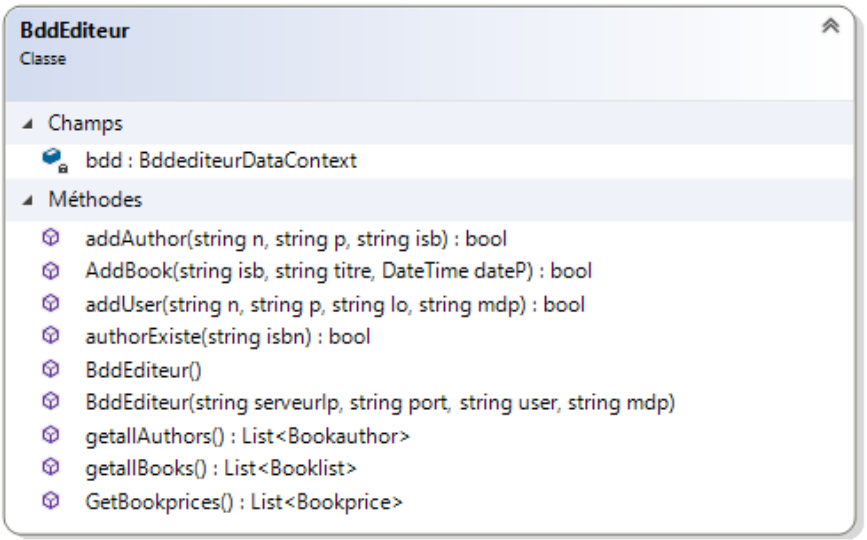


On retrouve les classes correspondant aux tables ainsi que les méthodes qui vont permettre de gérer la base de données. En déroulant "***BddEditeurDataContext***", on peut voir les méthodes générées :



Maintenant il est possible de commencer à coder les méthodes de la classe *BddEditeur.cs*. Pour utiliser toutes ces méthodes généreusement crées par Linq, il est possible de se documenter dans la rubrique documentation (extraction de données) dont le lien est enregistré à la suite de l'installation de linq. Un lien vers des exemples C# est également fourni.

11. Complétez la classe *BddEditeur.cs* selon le diagramme ci-dessus :



Remarque : les méthodes de la classe *BddEditeur* sont évidemment amenées à êtres modifiées selon les besoins de l'application. Certaines méthodes représentées dans le diagramme sont inutiles (données en exemple) et d'autres manquent peut-être.

12. Complétez le projet ***WpfAppEditeur*** pour répondre aux fonctionnalités de l'application selon la description et le diagramme de cas d'utilisation présentés ci-dessus.