# LINQ to Entities – Exercices

## Objectifs

- ✓ Créer un modèle à partir d'une base de données existante
- ✓ Générer une base de données à partir d'un modèle
- ✓ Manipuler la base de données au travers de Linq to Entities ( ajout, suppression, recherche)
- ✓ Utiliser les propriétés de navigation

## A – Création de la base de données à partir d'un schéma existant

1. Récupérez sur ecampus le fichier « instnwnd.sql»
2. Dans Visual Studio, vous pouvez manipuler les bases de données SQL SERVER
    a. Affichage -> Explorateur d'objets SQL SERVER -> Ajouter SQL SERVER -> Local -> sélectionner votre instance SQL SERVER
    b. Ex instances SQL SERVER : MSSqlLocalDb, V11.0
    c. Retenez le nom de l'instance, vous en aurez besoin !
3. Créer un nouveau query
    a. Outils –> SQL SERVER -> Nouvelle requête
    b. Copier-coller le code du fichier «instnwnd.sql»
4. Exécutez le query -> vous avez maintenant une base de données nommée « Northwind »
5. Créer un nouveau projet console
6. Créer une connexion à la base de données « Northwind »
    a. Outil -> connexion à la base de données
    b. Entrez le nom de l'instance du serveur (voir point 2) (**ne faites pas actualiser -> cela va être long car un scan du réseau va s'effectuer à la recherche des moteurs de base de données SQL SERVER**)
    c. Sélectionner la base de données « Northwind »
7. Créer le modèle entités associations à partir de ce schéma :
    a. Click droit sur le projet -> ajouter nouvel élément.
    b. Sélectionnez « ADO.NET Entity Data Model ».
    c. Sélectionner « EF Designer à partir de la base de données ».
8. Regardez le schéma edmx en vue graphique
9. Regardez également les fichier c# générés.
10. Dans votre M*ain()* créez un objet contexte à partir de la base de données.

## B – Queries

1. Lister tous les *Customers* habitants dans une ville saisie au clavier.
2. Saisissez au clavier un nom de pays et donner le nom de tous les produits disponibles dans ce pays. Utilisez **le lazy loading** ! Le résultat sera trié par Pays.
3. Saisissez au clavier un nom de pays et donner le nom de tous les produits disponibles dans ce pays. Utilisez **l'eager loading** ! Le résultat sera trié par Pays.
4. Donnez pour un client donné ( saisi au clavier) la liste de ces 3 dernières commandes par ordre décroissant de date de livraison. Les commandes ne sont affichées que si elles ont été livrées c'est-à-dire que le champ «ShippedDate » est rempli. Les champs renvoyés par ce

query sont le ID du client «CustomerID », la date de la commande « OrderDate » et la date de livraison « ShippedDate ».

5. Afficher le total des ventes par produit (ID produit -> Total ).

## C – Updates

1. Mettez en majuscule le nom de tous les clients.
2. Vérifiez que l'update a bien été réalisé en faisant un query en rechargeant la DB.

## D – Deletes

**Important suivant le modèle généré (dbContext -> fichier .tt, objectContect) la méthode de suppression est soit :**

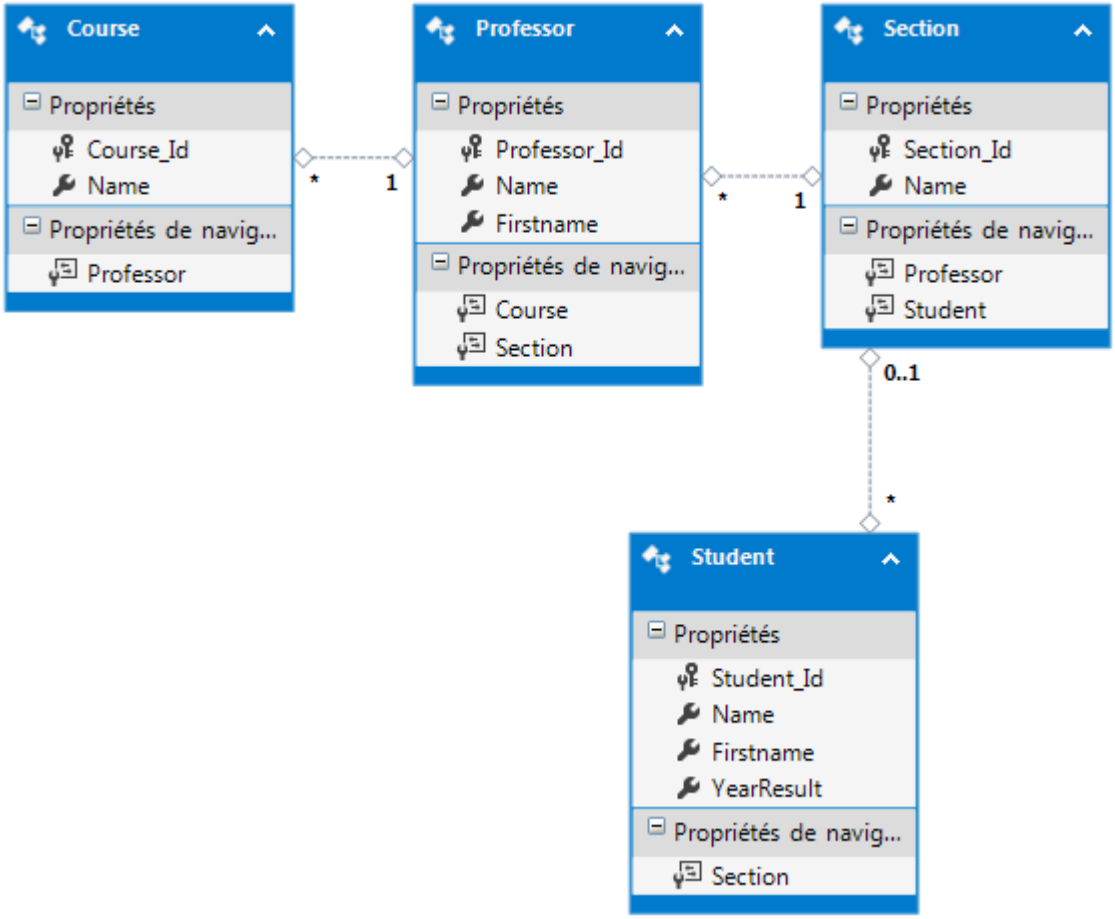
- **DeleteObject -> context.DeleteObject(objetCategorie) ;**
- **Remove -> context.Categories.Remove(objetCategorie) ;**

**Utilisez donc la méthode suivant le modèle que vous a généré Visual Studio. La dernière version du framework utilise Remove donc c'est normalement celle-ci que vous utiliserez.**

1. Supprimer une catégorie à partir d'un nom rentré en paramètre
2. Vérifiez que le delete à bien été réalisé en faisant un query en rechargeant la DB.
3. Entrez au clavier deux noms d'employés. Supprimez le premier employé et réassignez tous ses *orders* au deuxième.
4. Vérifier

## E – Création de la base de données à partir d'un modèle objet

1. A partir du modèle UML suivant -> créer un modèle entité avec associations
    a. https://msdn.microsoft.com/en-us/data/jj205424




2. Utiliser des propriétés de navigation comme sur le schéma
3. Créer une base de données « test ».
4. Créer une connexion de données vers cette base de données
5. Générer la base de données à partir de ce modèle
6. Injecter le sql pour créer les tables

7. Ajouter deux sections (sectInfo, sectDiet)
    a. Si la section existe déjà en DB elle ne doit pas être ajoutée -> vérification su r le nom
8. Faites un query LINQ pour vérifier que toutes les sections ont bien été créées.
9. Ajouter 3 étudiants
    a. studinfo1 dans la section informatique avec un « Year_Result » de 100.
    b. studdiet dans la section diététique avec un « Year_Result » de 120.
    c. studinfo2 dans la section informatique avec un « Year_Result » de 110.
    d. Si l'étudiant existe déjà en DB il ne doit pas être ajouté -> vérification su r le nom et prénom

10. Vérifiez le résultat en DB
11. Faites un query LINQ pour afficher les étudiants par section triée par ordre de résultat (les meilleurs en tête) -> utilisez les propriétés de navigation

12. Réfléchissons maintenant à améliorer le code produit ci-dessus
    a. L'accès aux données ( query LINQ ) ne devrait pas se trouver dans la classe Program
        i. Le pattern Repository pourra vous aider
    b. Essayer de rendre le code le plus générique possible afin d'éviter la réécriture de code similaire.