

## TP 1

### Objectifs :

- Utiliser des commandes en mode connecte

### Guide de travail :

- ✓ Rédaction d'un rapport.
- ✓ Technologie ADO.NET, Fournisseur d'accès OLEDB

## AFFICHAGE D'INFORMATIONS RELATIVES A DES CLIENTS :

On veut consulter les données contenues dans une table **Client** dans une base de données **Vente** :

- Client (CodeCl, Nom, Ville).

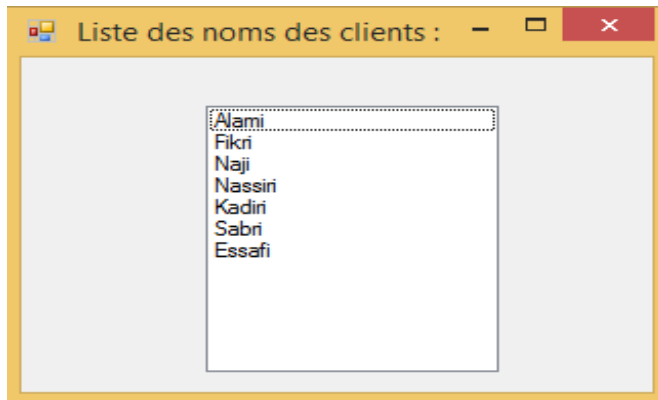
### Travail à réaliser :

a. Créer la base de données **Vente** et la table **Client** sous Sql Server.

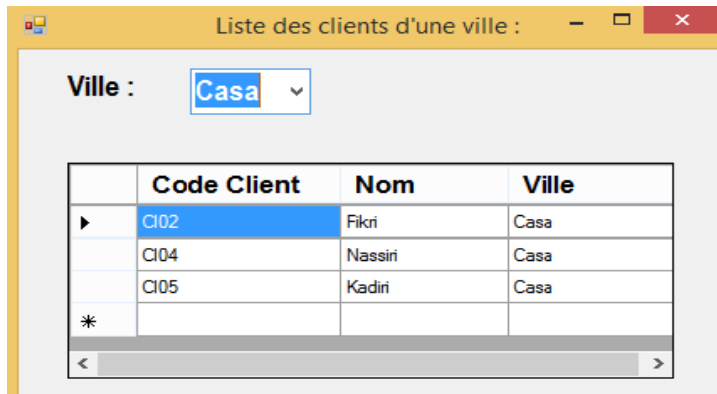
b. Pour accéder à cette base de données, en mode connecté, créer les objets suivants :

- une connexion con.  
`SqlConnection con = new SqlConnection();`
- une commande CmdSelect.  
`SqlCommand CmdSelect = new SqlCommand();`
- une variable dr de type datareader.  
`SqlDataReader dr;`

c. Créer un projet VS et le formulaire suivant permettant d'afficher les noms de tous les clients dans un ListBox :

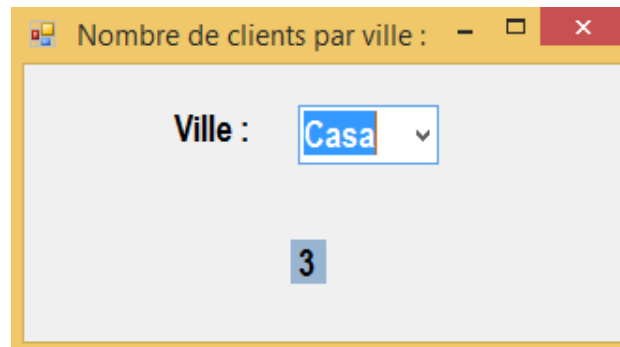


d. Dans le même projet ajouter un deuxième formulaire permettant d'afficher dans un DataGridView les clients d'une ville choisie dans un Combo :



Exemple d'une commande qui renvoie une valeur :

Dans le même projet ajouter un troisième formulaire permettant d'afficher dans un Label le nombre de clients d'une ville choisie dans un Combo :



## GESTION COMMERCIALE « PARTIE 1 »

Dans le cadre du développement d'une application de gestion commerciale, on veut commencer par faire la gestion de nos clients.

On utilisera la table **Client** de la base de données **Vente** :

- Client (CodeCl, Nom, Ville).

Pour réaliser la mise à jour (Ajout, modification et suppression) de cette table, en mode connecté, créer les objets suivants :

- une connexion con.

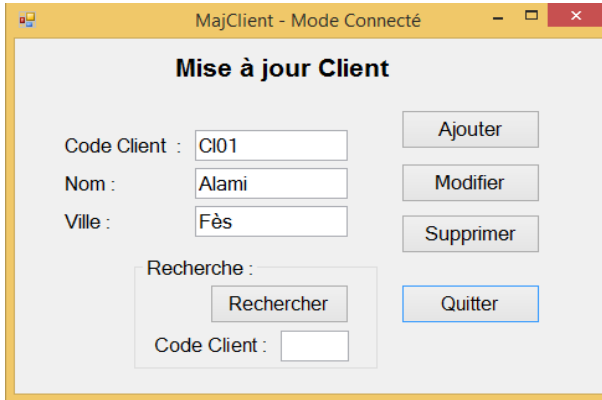
- une commande CmdMaj.

Pour modifier ou supprimer un client, il faudra le chercher. Créer aussi :

- une variable dr de type datareader.

### **Travail à réaliser :**

a. Créer un projet VS nommé « **Gestion Commerciale** » et le formulaire suivant permettant de faire la mise à jour de la table client :



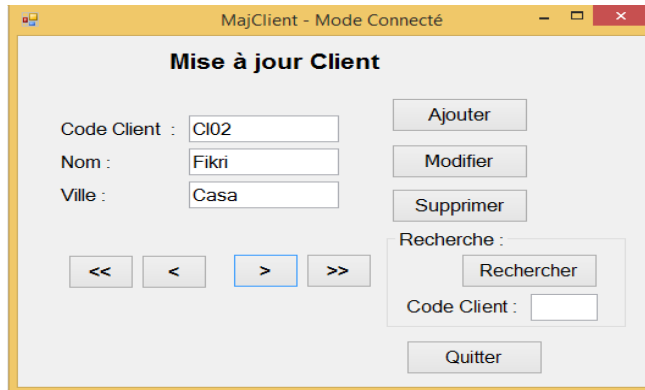
b. Écrire le code correspondant à chaque bouton réalisant le traitement attendu en utilisant les objets créés.

## **GESTION COMMERCIALE « PARTIE 2 »**

Pour finaliser le formulaire de gestion des clients, nous allons y ajouter de la navigation.

### **Travail à réaliser :**

a. Ajouter au formulaire de gestion des clients des boutons de navigation selon le modèle suivant :



Créer les objets suivants :

- un BindingSource :

```
BindingSource ClientBS = new BindingSource() ;
```

- un DataTable :

```
DataTable dt = new DataTable();
```

c. Écrire le code de chaque bouton de navigation en utilisant les méthodes Move du BindingSource.

Au démarrage le premier client doit être affiché.

## GESTION COMMERCIALE « PARTIE 3 »

Reprendre le travail fait dans les deux activités 2 et 3, pour réaliser la mise à jour de la table Article dont la structure est la suivante :

- Article (CodeArt, Désignation, PU, QStock).

### Travail à réaliser :

a. Dans la base de données **Vente**, ajouter la table Article.

b. Modèle du formulaire MajArticle :



Créer les objets nécessaires et le code réalisant le travail attendu.  
Au démarrage le premier article doit être affiché.

## GESTION COMMERCIALE « PARTIE 4 »

Il s'agit de créer un formulaire pour la saisie d'une commande d'un client en mode connecté utilisant, en plus des tables Client et Article, les deux tables suivantes :

- Commande ( NumCom, DateCom, #CodeCl).
- Détail ( NumCom, CodeArt, Qte)

### Travail à réaliser :

- a. Dans la base de données **Vente**, ajouter les deux tables Commande et Détail.
- b. Ajouter au projet « **Gestion Commerciale** » le formulaire suivant :

- c. Créer les objets nécessaires et le code réalisant le travail demandé.

### **Spécifications :**

- Au choix du code d'un client, ses informations sont affichées.
- Le combo au dessus du DataGridView sert à choisir les articles à ajouter à la commande. La quantité sera saisie.
- Un Clic sur le bouton « **Ajouter Ligne** » fait descendre la ligne dans le DataGridView.
- Le bouton « **Ajouter Ligne** » fait aussi ce qui suit :
  - ✓ calcule le Montant de la ligne et le Total de la commande à chaque ajout de ligne.
  - ✓ Verifie si le stock est suffisant (**Quantité commandée <= Quantité en stock**, c.à.d. **Qte <= QStock**) et affiche le message « **Stock insuffisant !** » si ce n'est pas le cas.
- Le bouton « **Supprimer Ligne** » permet de supprimer une ligne sélectionnée dans le DataGridView et recalcule le Total de la commande.

Lorsqu'il n'y a plus aucune modification à effectuer sur la commande, on peut l'enregistrer dans la base de données.

- Le bouton « **Enregistrer la commande** » permet d'enregistrer la commande et ses lignes (ses détails).
- Le bouton « **Enregistrer la commande** » fait aussi ce qui suit :

- ✓ Met à jour le stock (retranche la **quantité commandée** de chaque article de la commande de la **quantité en stock** de ce même article).
- ✓ Efface le formulaire pour préparer la saisie d'une autre commande.

## GESTION COMMERCIALE « PARTIE 5 »

Pour terminer l'application **Gestion Commerciale**, Il reste à créer un formulaire pour une consultation totale des commandes d'un client en mode connecté.

### Travail à réaliser :

a. Ajouter au projet « **Gestion Commerciale** » le formulaire suivant :

Créer les objets nécessaires et le code réalisant le travail demandé.

#### **Spécifications :**

- Au choix d'un client :
  - ✓ le Nom et la Ville s'affichent automatiquement.
  - ✓ La liste de toutes ses commandes s'affiche dans le 1<sup>er</sup> DataGridView.
  - ✓ Son Chiffre d'affaires est calculé et affiché dans un TextBox.  
(Chiffre d'affaires d'un client = Somme des totaux de ses commandes)
- Lorsqu'une commande est sélectionnée :
  - ✓ Ses détails son affichés dans le 2<sup>eme</sup> DataGridView.
  - ✓ Le Total HT de la commande, la TVA et le TTC sont calculés et affichés.

Le bouton « **Supprimer la commande** » permet de supprimer une commande sélectionnée dans le 1<sup>er</sup> DataGridView. Le Chiffre d'affaires du client est recalculé.