Objectif : Intégrer les accès aux données dans le client en mode connecté.

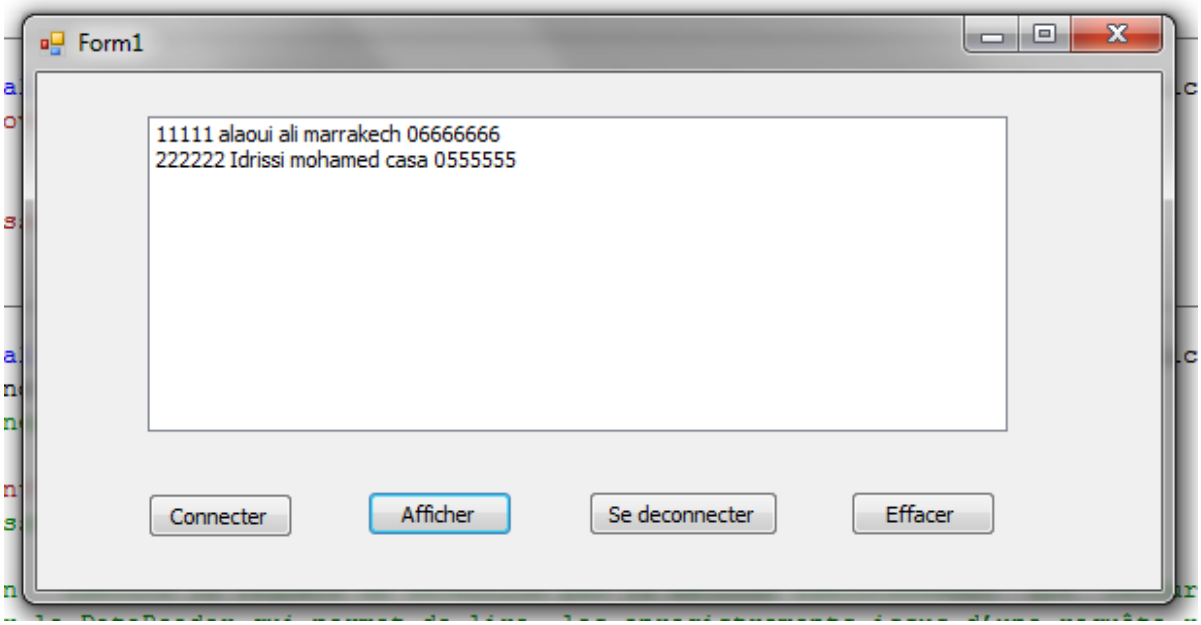
- Connexion à une base de données MS Access.
- Exécution des requêtes SQL.
- Exploitation du résultat d'une requête SELECT.

Créer une base de données MS Access ClientsDb comportant la table client :

CLIENT (CIN, NOM, PRENOM, VILLE, TEL)

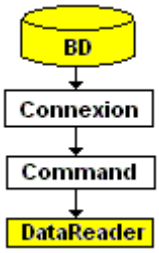
Question :

Soit à réaliser l'interface ci-dessous permettant d'afficher la liste de tous les clients dans un contrôle ListBox comme suit :



Solution :

Afin de réaliser une telle application, suivez les étapes suivantes :



1. Etape 1 : Déclaration du fournisseur d'accès aux données :

Puisque les données sont stockées dans le serveur de base de données MS ACCESS, il faut utiliser le fournisseur de données .Net Framework pour MS Access.

Pour ce faire il suffit d'importer l'espace de nom System.Data.OleDb

```
using System.Data.OleDb;
```

2. Etape 2 : Définir l'objet connexion :

L'objet connexion constitue la première étape dans l'accès aux données : elle permet de se connecter au serveur de base de données.

1. Déclarer un objet connexion global :

```
OleDbConnection cn = new OleDbConnection();
'déclaration d'un objet connection avec un constructeur par défaut
```

2. Définir la chaine de connexion en utilisant la propriété **ConnectionString** de l'objet **cn** (au chargement du formulaire)

```
cn.ConnectionString="Provider=Microsoft.ACE.OleDb.12.0;Data Source=ClientsDb.accdb";
```

La principale propriété est « connectionString ». C'est une chaîne de caractères définissant les principaux attributs nécessaires à la connexion. Parmi ces informations figurent :

- Le fournisseur d'accès
- L'emplacement de la base (le chemin vers la base de données)
- Informations d'authentification

Chaque fournisseur de données a une ConnectionString spécifique.

Remarque :

La chaine de connexion peut être le paramètre d'un constructeur de la classe OleDbConnection

```
OleDbConnection cn = new OleDbConnection("Provider=Microsoft.ACE.OleDb.12.0;Data
Source=ClientsDb.accdb");
```

Il faut mettre la base de données dans le dossier **bin\debug** ou bien préciser le chemin de la base de données dans le Data Source.

3. Ouverture et fermeture de la connexion:

- Ouverture de la connexion

```
Cn.Open() ;  //Ouvrir la connexion
```

- Fermeture de la connexion

```
Cn.Close() ; //Ferme la connexion et libère les ressources
```

On peut faire appel à la propriété State à fin de savoir l'état de la connexion

Valeur	Description
ConnectionState.Open	La connexion est ouverte
ConnectionState.Closed	La connexion est fermée

Exemple d'utilisation :

Le bouton « Connecter » qui permet de se connecter à la base de données :

```
cn.Open();
MessageBox.Show("Connexion bien établie");
```

Le bouton « Déconnecter » qui permet de se déconnecter de la base de données :

```
 cn.Close();
 MessageBox.Show("Connexion bien Fermée");
```

A faire :

Reprendre et améloirer le code précédant pour tenir compte des cas suivants :

- Connexion déjà ouverte.
- Connexion déjà fermée.
- La Connexion n'a pas pu s'établir.

3. Etape 3 : Définir l'objet commande :

Une fois la connexion établie avec la source de données, vous devez communiquer avec cette dernière pour gérer vos traitements. Trois types de traitements peuvent être effectués :

- Requête de sélection pour extraire des informations (Récupérer les données)
- Requête d'exécution (insérer, modifier et supprimer des données)
- Procédures stockées (scripts stockés sur le serveur)

Lors de la création d'un objet commande, vous devez définir le type d'opération qu'il devra réaliser ainsi que la connexion à laquelle il est rattaché.

1. Déclarer un objet Commande :

```
OleDbCommand cmd = new OleDbCommand();
```

2. Associer une connexion ouverte à la commande :

```
cmd.Connection = cn;
```

3. Définir le type de la commande :

```
cmd.CommandType = CommandType.Text; (le type par défaut)
```

4. Associer une requête SQL à la commande :

```
string req = "select * from client";
cmd.CommandText = req;
```

Exemple :

La première partie pour Le bouton « Afficher » :

```
OleDbCommand cmd = new OleDbCommand();
cmd.Connection = cn;
cmd.CommandType = CommandType.Text;
cmd.CommandText = "select * from client";
```

Les principales propriétés et méthodes de la classe Commande :

Propriété	Description
Commandtext	Texte SQL de la requête ou nom de la procédure stockée
CommandType	Type de la commande (requête, table, procédure)
Connection	Connexion liée à la commande
Transaction	Objet transaction lié (voir plus bas)
CommandTimeOut	Nombre de seconde pour l'exécution de la commande
Parameters	Collection de paramètres à envoyer avec la commande

Méthode	Description
Cancel	Annule l'exécution de la commande
ExecuteNonQuery	Exécute la requête d'action et retourne le nombre de lignes affectées
ExecuteReader	Exécute la requête de sélection et retourne un objet de type DataReader
ExecuteScalar	Exécute la requête et retourne la valeur scalaire (1ère ligne, 1ère colonne)
CreateParameter	Crée un objet paramètre
ExecuteXMLReader	Exécute la requête de sélection et retourne un objet de type XmlReader

De manière générale, il existe deux types de résultat pour un objet command : soit il retourne un seul résultat (c'est le cas lorsque vous utilisez les méthodes ExecuteScalar ou ExecuteNonQuery), soit il retourne un ensemble d'enregistrements (méthode ExecuteReader).

4. Etape 4 : Définir l'objet DataReader :

L'objet DataReader permet de récupérer les enregistrements issus d'une requête renvoyés par l'objet Command après son exécution.

1. Déclarer un objet DataReader :

```
OleDbDataReader dr;
```

2. Exécuter la commande et récupérer son résultat dans l'objet DataReader :

```
dr = cmd.ExecuteReader();
```

3. Lire le contenu de l'objet DataReader :

```
while (dr.Read())
{
listBox1.Items.Add(dr[0] +" "+ dr[1] +" "+ dr[2] +" "+ dr[3] +" "+ dr[4]);
}
```

4. Fermer l'objet DataReader :

```
dr.Close();
```

Voici les principales méthodes de l'objet DataReader :

Méthode	Description
Close	Définie par l'interface ***IDataReader***, elle permet de fermer le ***DataReader*** afin de libérer la connexion à la base de données
Read	Elle permet la lecture des enregistrements de la table. Elle va déplacer le pointeur de l'objet ***DataReader*** vers l'enregistrement suivant et ensuite va lire les informations de la ligne courante. Elle renvoie ***False*** s'il n'y a pas d'enregistrement suivant à lire, sinon elle renvoie ***True***

Ce qui donne pour la suite du bouton « Afficher »

```
OleDbCommand cmd = new OleDbCommand();
cmd.Connection = cn;
cmd.CommandType = CommandType.Text;
cmd.CommandText = "select * from client";
OleDbDataReader dr;
dr = cmd.ExecuteReader();
while (dr.Read())
  {
   listBox1.Items.Add(dr[0] +" "+ dr[1] +" "+ dr[2] +" "+ dr[3] +" "+ dr[4]);
  }
dr.Close();
```

À faire :

- Améliorer le code précédant pour tenir compte du cas ou la connexion n'est pas disponible.
- Afficher la liste des clients dans une grille.