C# - FICHE PRATIQUE N° 3 – Solutions et compléments

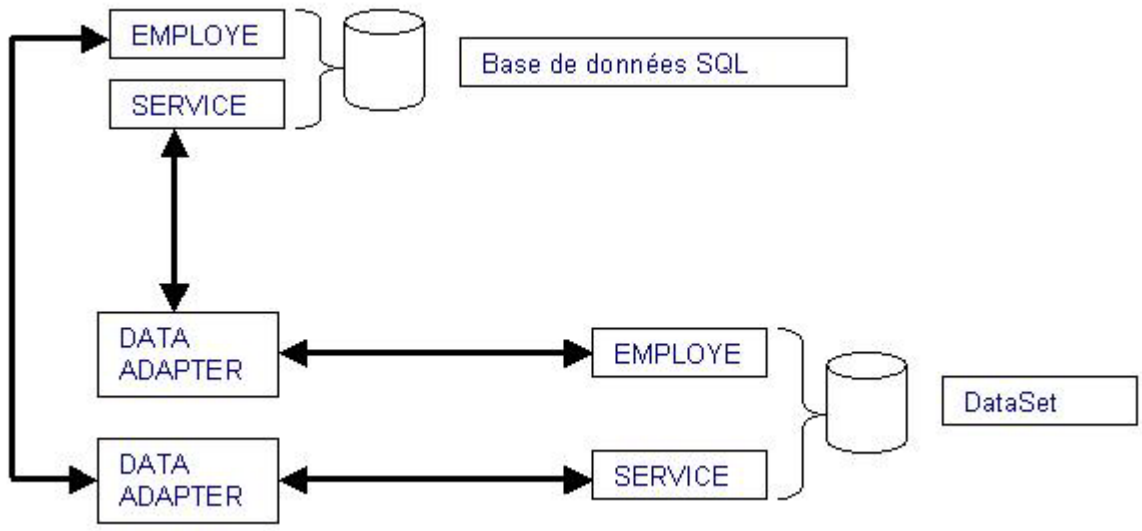
Une petite gestion de personnel
Les bases de l'accès aux données.

A/ Création de la base de données

Il est bien entendu possible de transposer les exemples proposés pour utiliser un autre SGBD. L'environnement fournit des middlewares spécialisés pour Access, SQL Server et Oracle, ainsi qu'un middleware ODBC.

B/ Première approche

> Le schéma de principe de l'accès aux données en mode déconnecté est le suivant :



- Chaque table de la base de données est gérée par un Data Adapter (un Data Adapter peut également gérer une requête plus complexe qu'un simple « select * from… »).
- Un groupe de données local (DataSet) contient une ou plusieurs tables locales, des relations entre ces tables, etc… Un DataSet ressemble donc à une base de données locale gérée par le framework. Il peut contenir des tables « identiques » à celle de la base de données distante, ou même des tables de structures différentes, alimentées par des Data Adapter correspondant à des requêtes plus personnelles (examinez la propriété « TableMappings » du Data Adapter…). Un DataSet est en fait une sorte de « vue » de la base de données propre à l'application à développer.
- Un Data Adapter gère plusieurs objets de type Command (requêtes SQL) :
  - SelectCommand : utilisé pour obtenir les données de la base
  - InsertCommand : utilisé pour insérer des données dans la base
  - UpdateCommand : utilisé pour modifier les données de la base
  - DeleteCommand : utilisé pour supprimer des données de la base
- La méthode « Fill » d'un Data Adapter permet de remplir le DataSet à partir de la base de données en utilisant son objet SelectCommand.

- Toutes les mises à jour s'effectuent ensuite localement, dans le DataSet. Ces mises à jour peuvent être faites par programme ou en utilisant des contrôles de saisie liés aux données du DataSet.
- La méthode « Update » d'un Data Adapter permet de mettre à jour la base de données en une seule fois, en utilisant ses objets InsertCommand, UpdateCommand et DeleteCommand.

On peut également accéder à une base de données en mode connecté, plus classique. Il est alors possible :

- D'exécuter une instruction SQL de type insert, update ou delete à l'aide d'un objet Command (SqlCommand dans le cas de SQL Server).
- D'exécuter une procédure stockée à l'aide d'un objet Command.
- D'obtenir un curseur correspondant à un instruction select et de le parcourir à l'aide d'un objet DataReader (SqlDataReader dans le cas de SQL Server).

> Affichage de la position courante et du nombre d'enregistrements

La programmation événementielle fournit une solution pour l'appel de la méthode « affichePosCpt ». En effet, un objet de type BindingContexte possède un événement PositionChanged. Il suffit donc de lier la méthode « affichePosCpt » à cet événement de la manière suivante :

```
public Fm_service()
{
    InitializeComponent();
    dbDs_service1.Clear();
    this.BindingContext[dbDs_service1,"tp1_service"].PositionChanged+=new EventHandler(affichePosCpt);
    dbAd_service.Fill(dbDs_service1);
}
```

La méthode « affichePosCpt » doit alors respecter le prototype d'une fonction de gestion d'événement :

```
private void affichePosCpt(object sender, EventArgs e)
{
    …
}
```

Remarques : il est bien entendu possible d'appeler explicitement « affichePosCpt » lors de chaque déplacement. Il est même souhaitable de laisser les étudiants commencer par cette solution simple.

> Mise à jour des données

Les méthodes AddNew et RemoveAt de l'objet BindingContexte permettent d'ajouter et de supprimer des services (la modification d'un service existant étant ici prise en charge par les contrôles liés aux données). Il est intéressant de montrer que ces modifications n'affectent absolument pas la base de données tant que la validation globale n'a pas été faite par l'intermédiaire du Data Adapter.

> Validation des données vers le SGBD

La solution de la validation automatique sera systématiquement employée par la suite, c'est un choix… On pourrait également faire traiter les deux événements « RowChanged » et « RowDeleted » par la même procédure, mais c'est sans grand intérêt.

Il est assez simple de modifier les fiches suivantes pour les adapter à l'autre possibilité : mise à jour « globale » de la base par un seul appel à la méthode « Update » du « DataAdapter ».

Remarque : la technique présentée nécessite l'inclusion des nameSpaces « System.data » et « Navigation ».

C/ Le composant DbNavigateur

Pour installer ce composant dans la boîte à outil, il suffit de :

- Choisir l'option « Ajouter / Supprimer des éléments de la boîte à outils » dans le menu « Outils ».
- Cliquer sur le bouton « parcourir » et sélectionner le fichier « navigation.dll » fourni.

Que fait ce composant :

- Il encapsule les problèmes de navigation et d'édition des données.
- Il permet d'appeler automatiquement une fonction lors de l'ajout d'un enregistrement ou du déplacement dans le jeu d'enregistrements.
- Il fournit un accès à l'enregistrement en cours (méthode Courant).
- Il facilite la mise à jour de la base de données distante en fournissant une méthode statique « GererRowAction ».
- Il met en place la gestion des éventuelles erreurs lors de la modification des données.