# Esercitazione di Fine Settimana – Week 4

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Debora |
|  |  | Cognome | Spano |
|  |  | Data | 10/8/2021 |

Leggete attentamente ogni domanda e argomentare quanto più possibile fornendo anche degli esempi.

1. **Descrivere le modalità di utilizzo di ADO.NET**

ADO.NET ha due modalità di utilizzo

* Connected Mode: lavora con la connesione sempre aperta. Permette di modificare direttamente i dati nel db senza passare da una struttura di supporto.

Per le operazioni di lettura si ottengono dati forward-only (leggibili solo dall’inizio alla fine)

* + Disconnected Mode: permette di creare una struttura di supporto che è il DataSet. In questo modo è possibile modificare i dati con la connessione chiusa e solo alla fine mandare le modifiche al database.

In questo caso si apre la connessione solo nel momento della lettura del DB. I dati del db vengono trasferiti nel dataset e da quel momento la connesione potrà essere chiusa.

Qualunque modifica apportata al DataSet sarà persistente solo nel in cui si esegue il comando Update che manda tutte le modifiche al db.

La struttura del dataset simula quella di un database, infatti è formato da una collezione di DataTable e DataRelations e le DataTable a loro volta contengono una collezione di DataColumn e di DataRow e così via.

1. **Quali sono i metodi di esecuzione della classe DbCommand disponibili e in quali casi vanno utilizzati?**

I comandi possono essere eseguiti con tre metodi: ExecuteReader, ExecuteNonQuery, ExecuteScalar

ExecuteReader è usata, per esempio, per le operazioni di select e restituisce un reader che contiene il risultato dell’operazione eseguita. Il reader può essere letto solo in un verso, dall’inizio alla fine.

ExecuteNonQuery è usata per tutte le operazioni “non select”, come le insert o le procedure.

ExecuteScalar è simile alla ExecuteReader ma restitusce un solo valore. Usata per quelle query che restituiscono sempre un solo valore (tipo count(), max())

1. **Descrivere l’utilizzo della classe DataAdapter**

Il dataAdapter è una classe utilizzata in Disconnected Mode. E’ l’oggetto da cui partono i principali comandi per caricare il database sul DataSet (Fill) e per aggiornare il database con le modifiche del dataset (update).

Per poter funzionare, bisogna assegnare all’adapter i comandi che si vogliono utilizzare, per esempio per eseguire la Fill l’adapter deve per forza avere un comando di select.

1. **Come si definisce il modello dati in Entity Framework?**

In entity framework un modello dati si crea partendo da una classe context (che eredita da DbContext). Questa classe conterrà due costruttori (costruttore no-args e un costruttore che prende le opzioni di configurazione), un metodo che permetta di configurarlo (con la stringa di connessione per esempio) e una o più proprietà DBSet.

Le prorpietà DbSet rappresenteranno le tabelle del nostro Db.

Ogni Tabella avrà una classe associata con gli stessi dati e un DbSet<Classe> nel context.

Ogni volta che si vuole accedere al db per leggere o modificare bisogna creare un’istanza di context. Dentro l’oggetto ontext sarà possibile accedere ad ogni dato di ogni tabella senza aver bisogno di utilizzare comandi SQL

1. **Descrivere l’utilizzo delle Migration e i vantaggi che ne derivano**

Le migration permettono di inizializzare delle tabelle e/o modificarle partendo da codice con il vantaggio che è possibile creare tabelle e vincoli senza mai mettere mano a sql, ma solo tramite codice.

**Esercitazione Pratica**

Realizzare un sistema di gestione delle spese che si basi su:

* Un database **GestioneSpese** (SQL Server), costituito dalle tabelle
  + **Spese**
    - *Id* (int, PK, auto-incrementale)
    - *Data* (datetime)
    - *CategoriaId* (int, FK)
    - *Descrizione* (varchar(500))
    - *Utente* (varchar(100))
    - *Importo* (decimal)
    - *Approvato* (bit)
  + **Categorie**
    - *Id* (int, PK, auto-incrementale)
    - *Categoria* (varchar(100))
* Una **Console app** che consenta di:
  + Inserire nuove Spese
  + Approvare le Spese esistenti (impostare il campo *Approvato*)
  + Cancellare le Spese esistenti
  + Mostri
    - l'elenco delle Spese Approvate
    - L'elenco delle Spese di uno specifico Utente
    - Il totale delle Spese per Categoria

VINCOLI TECNICI

* Utilizzare Entity Framework
* Utilizzare l'approccio Code-First e attivare le Migrations

**OPZIONALE**: Implementare una o più delle funzionalità utilizzando ADO.NET (Connected o Disconnected mode, a scelta)

*Es. Inserire una nuova spesa*