# Esercitazione di Fine Settimana – Week 5

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Michael |
|  |  | Cognome | Locci |
|  |  | Data | 10/08/2021 |

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

1. Descrivere le modalità di utilizzo di ADO.NET

Le modalità di utilizzo di ADO.NET sono due:

* Connected Mode che fornisce accesso di sola lettura ai dati nell’origine dei dati e la possibilità di eseguire i comandi su di essa. Le classi principali sono Connection per la connessione sql, Command per la creazione dei comandi sql, DataReader per la gestione dei dati in lettura e Parameter per la gestione dei parametri sql. La modalità Connected segue i seguenti passi:
  + Creazione ed apertura della connessione
  + Creazione del comando e cui seguono la creazione dei parametri se occorrono
  + Esecuzione del comando
  + Lettra dei dati
  + Chiusura connessione
* Disconnected Mode che consente di manipolare i dati recuperati dall’origine dei dati e dopo averli manipolati in locale e possibile fare una riconciliazione con l’origine dei dati, infatti la connessione viene aperta al momento della lettura e chiusa dopo ver letto i dati. Le principali classi utilizzate sono DataSet, DataTable, DataColumn, DataRow, DataRelation e Constraint. Il DataSet è una raccolta di uno o più oggetti DataTable e viene popolato da un oggetto di tipo DataAdapter, infatti contiene tabelle, relazione tra le tabelle ed i vincoli di integrità dei dati. La DataTable è l’effettivo contenitore delle informazioni in memoria e presenta una struttura composta da una collezione di DataColumn e da una di Constraint (PK, FK, UNIQUE).

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

I metodi della classe Dbcommand sono la creazione del comando INSERT per le operazioni di aggiunta, creazione del comando DELETE per le operazioni di eliminazione, creazione del comando Update per le operazioni di update, creazione del comando READ per le operazioni di lettura, creazione del comando SCALAR che ritorna il primo valore della prima riga della tabello nel database.

1. Descrivere l’utilizzo della classe DataAdapter

Il DataAdapter fa da collegamento fra l’oggetto DataSet e l’origine dei dati. Utilizza oggetti command per eseguire i comandi SQL sull’origine dei dati per caricare sia DataSet con dati sia per riconciliare le modifiche apportate ai dati nel DataSet con l’origine dei dati. Inoltre permette di collegare l’origine dei dati e le classe disconnesse tramite le classi connesse. Sull’oggetto DataAdapter vengono eseguite le chiamate dei metodi Fill, per le operazioni di query ed apertura della connessione, ed Update, per l’operazione di aggiornamento dei dati.

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

Il modello dei dati può essere definito con 3 approcci:

* Database-First ovvero il modello viene importato da un DB esistente e se modifico il DB posso sempre aggiornare il modello.
* Model-First ovvero il modello del SB viene creato dal designer di Visual Studio, per cui l’implementazione fisica è basata sul modello creato. Inoltre e modifiche al DB andranno perse in quanto il modello definisce il DB.
* Code-First ovvero il modello viene creato dal nostro codice come l’implementazione fisica che si basa sul nostro codice

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

Le Migrations sono un meccanismo che consente , utilizzando l’approcio code-first, l’aggionamento del database a fronte di modifiche al modello. Possono avvenire in due modi ovvero in automatico oppure manuali, basate sul codice. Per farle funzionare si devono eseguire due comandi dalla Package Manager Console:

* Add-migration “nome della migrazione”, ovvero crea una nuova classe con tutte le modifiche rispetto allo stato precedente del DB, caratterizzata da:
  + Timestamp + nome migrazione come nome
  + La classe contenuta eredita da migration
  + Contiene due metodi: UP (aggiorna con le nuove modifiche) e DOWN (riporta allo stato precedente)
* Update-database, ovvero aggiorna il DB sulla base del modello

Ed è possible fare anche il rollback delle modifiche fatte. Le migration ci fornisco più controllo sulle modifiche.

**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*