# Esercitazione di Fine Settimana – Week 5

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Gian mario |
|  |  | Cognome | Falchi |
|  |  | Data | 8/10/2021 |

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

1. Descrivere le modalità di utilizzo di ADO.NET

Connected mode: l’applicazione è sempre connessa al db quando deve fare operazioni. Le operazioni sono fatte tramite stringhe contenenti codice sql direttamente sul db.

disconnected mode: i dati vengono immagazzinati in dei data set tramite comandi select, e poi la connessione col db viene chiusa. Se devo elaborare dati lo faccio sul dataset; se devo modificare dati, inserirne nuovi, modificare tabelle colonne ecc. posso farlo sul dataset e poi devo sincronizzare il db col dataset (tramite comando Update())

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

DbCommand viene ereditata da SqlCommand. Nel Connected mode viene usato per inviare i comandi al db. I membri che abbiamo usato sono:

* Per impostare l’oggetto: CommandType, Connection, CommandText

I Metodi che abbiamo usato sono :

-new() o oggettoSqlConnection.CreateCommand() per istanziarlo

-ExecuteReader() per fare le select e ExecuteNonQuery per fare insert e delete.

Nel disconnected mode usiamo Connection, commandtype e commandtext, in più usiamo Parameters.Add() per fare il passaggio di parametri, per poi salvare i comandi negli adapter.

1. Descrivere l’utilizzo della classe DataAdapter

Nella disconnected mode il data adapter fa da tramite tra il dataset e il database disconnesso. Tramite oggetti di tipo command ci permette di fare le operazioni (SelectCommand, InsertCommand, DeleteCommand), più il metodo Fill() al quale passiamo il dataset per riempirlo e Update() per caricarlo nel db.

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

Con l’approccio database first il modello è importato da un db esistente

Con l’appr. Model first chi scrive il codice dell’applicazione fa anche il db e poi adatta il modello al db che ha creato

Con l’appr. Code first è sufficiente creare il db e nient’altro. Tramite la creazione del modello e in seguito la migrazione il db verrà creato in automatico.

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

Il vantaggio delle migration è che non scrivo una sola riga di sql e sono certo al 100% che il db e il mio modello combaciano. Col comando add-migration NomeMigrazione viene creato un file .cs (che volendo posso anche modificare a mano se non mi soddisfa a pieno) dove si vede chiaramente come vengono descritte tutte le tabelle coi campi già ben definiti (basati sul mio modello e in particolare sul context e sul metodo OnModelConfig()). Col comando update-database vengono eseguiti tutti i comandi del file della migrazione e in seguito ad esso il modello creato verrà tradotto sul database.

**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*