# Esercitazione di Fine Settimana – Week 4

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Deborah |
|  |  | Cognome | Tucci |
|  |  | Data | 09/04/2021 |

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

1. Descrivere le modalità di utilizzo di ADO.NET

Ado.Net è un insieme di tecnologie che consente l’accesso ai dati in due modalità.

Connected Mode fornendo l’accesso di sola lettura ai dati direttamente nell’origine dati e permette di eseguire le operazioni direttamente sull’origine dati.

Disconnected Mode consente di recuperare e manipolare i dati in locale e successivamente riconciliarli con l’origine dati.

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

I metodi di esecuzione della classe DbCommand disponibili sono il DataReader, per la lettura dei record, l’ExecuteNonQuery per l’esecuzioni di Insert ad esempio, e il command Scalar per il recupero di un solo valore dalla select da eseguire.

1. Descrivere l’utilizzo della classe DataAdapter

La classe DataAdapter viene utilizzata nella modalità disconnessa e si pone da ponte tra l’oggetto DataSet (che raccoglie il DB in locale) e l’origine dei dati.

Ed è la classe che esegue i comandi SQL sull’origine dei dati e riconcilia le modifiche eseguite nel DataSet.

1. Come è possibile andare a definire il modello dati in Entity Framework?

È possibile definire il modello dati in Entity Framework perché esso è un ORM, cioè una tecnica che converte i dati di un database in oggetti di un’applicazione object-oriented, e viceversa, mediante mapping e fetching.

Esistono diversi approcci:

Database-First nel quale il modello viene importato direttamente dal database.

Model-First nel quale il modello viene create dal designer di Visual Studio.

Code-First nel quale il modello viene creato dal codice.

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

Le Migration traducono il codice in sql per realizzare il Database sul server scelto. Inoltre memorizzano i vari aggiornamenti apportati alla struttura del database in specifiche classi facilmente consultabili.

**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*