# Esercitazione di Fine Settimana – Week 4

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Martina |
|  |  | Cognome | Libreri |
|  |  | Data | 9/04/2021 |

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

1. Descrivere le modalità di utilizzo di ADO.NET

Ado può essere utilizzato in due diverse modalità:

1. Modalità Connessa: permette un’accesso in lettura ai dati e la possibilità di eseguire comandi sull’origine dei dati. Per poter eseguire questi comandi vengono utilizzati degli elementi che sono la Connection, Command, Parameter e DataReader.

Il funzionamento consiste nel creare e aprire la connessione ad database creare i comandi ed eseguirli, e successivamente chiudere la connessione.

1. Modalità Disconnessa: utilizza oltre alla connection, Command, Parameter e Datareader anche il Dataset che consiste nel contenere una raccolta di tabelle e successivamente il DataAdapter fornirà una connessione tra il Dataset e l’origine dei dati.
2. Quali sono i metodi di esecuzione della classe DbCommand disponibili e in quali casi vanno utilizzati?

Il comando inserito in Ado è un comando che rappresenta un’istruzione di Sql; questo comando può essere una query che deve essere eseguita sull’origine dati a cui si collega Ado. Per eseguire I comandi ho tre possibilità di scelta: ExecuteNonQuery, che esegue l’istruzione dal comando, ExecuteReader, che restituisce un’oggetto DataReader, ExecuteScalar.

1. Descrivere l’utilizzo della classe DataAdapter

Il DataAdapter viene utilizzato in modaità disconnessa per fornire il ponter tra il DataSer e l’origine dei dati. Il data adapter consente di eseguire quindi i comandi in maniera disconnessa per modificare i dati nel dataset e successivamente riconciliarli con l’origine dati apportandone modifiche.

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

Entity framework essendo un ORM permette di essere uno strato che si interpone tra l’applicazione ad oggetti e i dati, fa da adapter interponendosi tra i due strati, e l’operaizone di mapping mi consente di trasformare gli oggetti instanziati nell’applicazione in dati leggibili dal database. Il mapping viene effettuato senza che il programmatore abbia controllo sul mapping stesso, non è del tutto automatico, ho delle convenzioni da usare ma ho meno elementi da dover inserire, con meno vincoli.

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

Con Entity Framework è stato introdotto il concetto di migration che permette di aggiornare lo schema di un database quando il modello dambia senza perdere i dati. Prò essere aggiornato in modo automatico o da 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*