# Modulo 6

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Deborah |
|  |  | Cognome | Tucci |
|  |  | Data | 05/03/2021 |

Leggete attentamente ogni domanda e argomentare quanto più possibile **fornendo anche degli esempi**.  
ATTENZIONE: Le domande a risposta multipla possono contenere più risposte corrette.

1. *Quale/i delle seguenti affermazioni è esatta riguardo la connection string?*

La connection string è un oggetto che gestisce la connessione a un database

La connection string è una stringa che fornisce i parametri di inizializzazione per creare una connessione

La connection string è composta da valori in formato chiave-valore

Se un utente accede al db da diverse postazioni, la connection string cambia

I parametri di inizializzazione per creare una connessione con un database che contiene una stringa di connessione sono ad esempio il nome del server, le credenziali di accesso e il nome del database di riferimento.

1. *Quale/i degli oggetti seguenti sono forniti da un provider per gestire l’accesso ai dati?*

Connection

DataSet

Command

DataTable

L’oggetto Connection fornisce la connessione ad una sorgente dati tramite una Connection String.

L’oggetto Command consente l’accesso ai comandi del database per restituire dati, modificare dati, eseguire stored procedure o recuperare informazioni sui parametri.

1. *Quale/i delle seguenti affermazioni è vera riguardo ai DataSet?*

Il Dataset dipende da un’unica sorgente di dati

Il Dataset contiene righe e colonne di dati

Il Dataset può gestire sorgenti di dati multiple

Il DataSet è composto da una gerarchia ad “albero”

Il DataSet è progettato per l’accesso ai dati indipendentemente da qualsiasi origine dati. Il DataSet contiene la struttura gerarchica, le tabelle e i dati prelevati da uno o più Database relazionali e tutte le relative informazioni. Dunque il DataSet contiene una raccolta di uno o più oggetti DataTable costituiti da righe e colonne dati, informazioni relative a chiave primaria, chiavi esterne, vincoli e relazione sui dati.

1. *Spiegare come funziona una connection pool*

Una Connection Pool è un insieme di connessioni al database. Ogni volta che un utente chiama una connessione con open(), il pooler cerca una connessione libera nella connection pool. Se è libera una connessione in pool, la restituisce al chiamante che la occupa, invece di crearne una nuova. Quando invece l’applicazione chiude la connessione con close(), il pooler libera la connessione e la restituisce al set di connessioni attive in connection pool anzichè chiuderla. Una volta che la connessione viene restituita al pool, è pronta per essere richiamata e riutilizzata. Se invece un utente deve accedere al database ma le connessioni sono tutte occupate deve attendere che si liberi una per via del timeout o attendere che un altro ne liberi una.

1. *Spiegare le differenze tra le modalità di connessione al database*

ADO.NET consente l’accesso ai dati in due modalità distinte, quali modalità connessa o modalità disconnessa. Il Connected Mode permette un accesso di sola lettura ai dati nell’origine dati e di eseguire comandi direttamente sull’origine dei dati.

Mentre, il Disconnected Mode consente di manipolare in locale, i dati recuperati dall’origine dati, senza manomettere direttamente il database origine, e solo successivamente rinconciliarli con l’origine. Dunque ha una parte che sfutta la connessione, ma le classi disconnesse forniscono un modo di lavorare con i dati disconnessi, indipendentemente dall’ambiente di origine data.

1. *Spiegare cosa è un DataAdapter e specificare in quale modalità si usa*

Il DataAdapter è un oggetto utilizzato nella modalità disconnessa e funge da ponte tra l’oggetto DataSet in locale e il Database d’origine. DataAdapter utilizza oggetti Command per eseguire comandi SQL sia per caricare il DataSet con dati prelevati dal database sorgente sia per riconciliare le modifiche apportate ai dati nel DataSet con l’origine dati. Dunque la classe DataAdapter permette di collegare l’origine dati e le classi disconnesse tramite le classi connesse. Inoltre, Adapter confronta le versioni del Database per riconciliare i dati.

1. *Spiegare cosa è un’interfaccia ed eventuali differenze con una classe astratta. Descriverne il funzionamento con un esempio.*

Un'interfaccia è in simile a una classe base astratta pura, cioè con solo metodi astratti. Qualsiasi che implementa l'interfaccia deve implementarne tutti i membri. Dunque un’interfaccia è priva di qualsiasi implementazione. Un’interfaccia definisce un contratto che la classe che la implementa deve rispettare. Non è possibile creare direttamente un'istanza di un'interfaccia. Una classe può implementare più interfacce contemporaneamente. La classe derivata però non può essere abstract. L’utilizzo dell’interfaccia permette di creare dipendenze tra i tipi senza ricorrere all’ereditarietà, dato che .NET l’ereditarietà multipla non è ammessa. Infatti una classe può ereditare una sola classe base ma anche implementare una o più interfacce.

Una classe invece si definisce astratta (abstract) se uno o più metodi o proprietà di tale classe sono privi di implementazione. Le classi astratte possono svolgere la funzione di classi base per altre classi che forniscono l'implementazione mancante. Ma una classe derivata, a differenza delle interfacce, può derivare una solo classe base astratta.

Esempio di interfaccia

Interface IPerson {

void Work();

}

Public class Manager : IPerson {

public void Work() { //Implementazione }

}

1. *Dare una definizione di Extension Method fornendo un esempio di utilizzo.*

Un estension method è un metodo necessariamente marcato static che permette di estendere le funzionalità di tipi e classi esistenti. Il metodo di estensione viene agganciato al tipo con la keyword this. Il metodo si estenzione poi viene chiamato come se fosse un metodo di istanza sul tipo esteso.

Esempio di definizione: public static class SqlDataReaderExtensions {

public static Movie ConvertToMovie(this SqlDataReader reader)

{

return new Movie()

{

ID = (int)reader["ID"],

Titolo = reader["Titolo"].ToString(),

Genere = reader["Genere"].ToString(),

Durata = (int)reader["Durata"],

};

}

}

Esempio di chiamata:

SqlDataReader reader = command.ExecuteReader();

Movie movie = reader.ConvertToMovie();

1. *Definire una funzione virtuale e spiegare eventuali differenze con una funzione “normale” e una funzione astratta.*

Un funzione virtuale è un metodo che permette la realizzazione del polimorfismo. Un metodo marcato virtual è una funzione che a differenze di quelle normali, può essere polimorfo, cioè può avere una implementazione di default che può essere sovrascritta nelle classi derivate mediante la parola chiave override, ma non necessariamente. Invece, i metodi marcati abstract sono funzioni astratte che non hanno un’implementazione e che deve essere necessariamente definita dalle classi derivate concrete. Esempio:

nella classe base:

public virtual void GetPrezzo (var valore){

var prezzo = valore;

}

nella classe derivata:

public override void GetPrezzo(var valore){

var prezzo = valore + tax;

}

*Esercitazione pratica*

*Fare un’applicazione Console che implementi le seguenti funzionalità:*

*-Mostrare tutti gli agenti*

*-Mostrare gli agenti assegnati ad una determinata area data da input dell’utente.*

*-Mostrare gli agenti con anni di servizio maggiori o uguali rispetto ad un input dato dall’utente*

*-Dare la possibilità all’utente di inserire un nuovo record di agente. (Modalità disconnessa)*

*L’agente deve essere inteso come una derivazione di un’astrazione di Persona.*

*Persona ha le seguenti caratteristiche:*

* *Nome*
* *Cognome*
* *Codice Fiscale*

*Due Persone sono uguali se hanno lo stesso codice fiscale.*

*I dati relativi ad un agente devono essere mostrati a schermo tramite il seguente formato:*

*Codice Fiscale – Nome Cognome – AnnidiServizio anni di servizio*

*Opzionale:*

*Gestire l’input dell’utente tramite un menu iniziale*

*Mettere la prova pratica e teorica su Github.*