C#Corner

.Net Interview
Question & Answer

Nitin Pandit

Including Technologies

C#, ADO.NET, ASP.NET, WCF, WPF, MVC
SQL Server

Total Pages : 400

.NET Interview Questions and answer:

Practical Implementation

This free book is provided by courtesy of <u>C# Corner</u> *and Mindcracker Network and its authors. Feel free to share this book with your friends and co-workers.* *Please do not reproduce, republish, edit or copy this book.*

Nitin Pandit

Software Engineer,
Microsoft.Net Consultant Noida
C# corner Delhi Chapter Lead
C# corner MVP

INDEX

About Author

He is a Microsoft Professional having Master's degree in Computer Science. With over 5 years of experience his rick skill set includes designing, integrating, implementing, ORM with LINQ, WCF, MVC & managing IT/Web applications. He is an expert in C#.NET ADO.NET, LINQ to SQL, LINQ to EF, ASP.NET 2.0/3x/4.0, WCF, MVC 5.0 (Razor), WPF and Silverlight, including client side technologies jQuery and AngularJS.

He has trained more than one lakh students and professionals as a speaker for workshops and AppFests conducted in more than 25 universities of North India.

Nitin Pandit

Software Engineer,
Microsoft.Net Consultant Noida
C# corner Delhi Chapter Lead
C# corner MVP

Chapter 1
.NET CLR Interview Questions and Answers

Question 1: What is the .NET Framework?

Answer: The .NET is a Framework, which is a collection of classes of reusable libraries given by Microsoft to be used in other .NET applications and to develop, build and deploy many types of applications on the Windows platform including the following:

- Console Applications
- Windows Forms Applications
- Windows Presentation Foundation (WPF) Applications
- Web Applications
- Web Services
- Windows Services
- Services-oriented applications using Windows Communications Foundation (WCF)
- Workflow-enabled applications using Windows Workflow Foundation (WF)

that primarily runs on the Microsoft Windows Operating System.
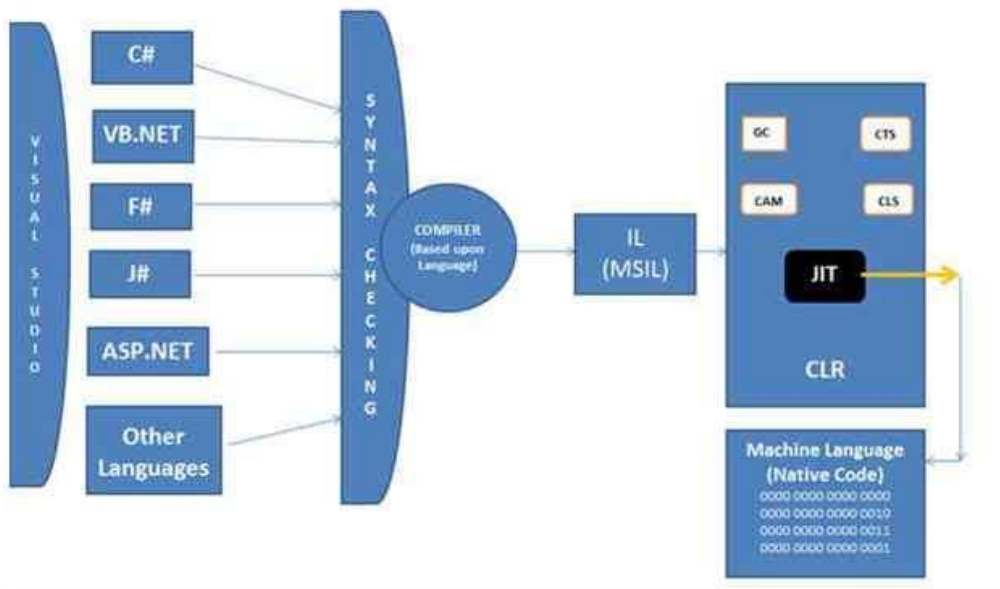
Question 2: What is CLR?

Answer: The CLR stands for Common Language Runtime and it is an Execution Environment. It works as a layer between Operating Systems and the applications written in .NET languages that conforms to the Common Language Specification (CLS). The main function of Common Language Runtime (CLR) is to convert the Managed Code into native code and then execute the program. The Managed Code compiled only when it is needed, that is it converts the appropriate instructions when each function is called. The Common Language Runtime (CLR)'s just in time (JIT) compilation converts Intermediate Language (MSIL) to native code on demand at application run time.

When a .NET application is executed at that time the control will go to Operating System, then Operating System create a process to load **CLR.**

The program used by the operating system for loading CLR is called runtime host, which are different depending upon the type of application that is desktop or web based application i.e.

The runtime host for **desktop applications** is API function called **CorbinToRuntime**.

The runtime host for **web based** applications is ASP.NET worker process **(aspnet-wp.exe)**.
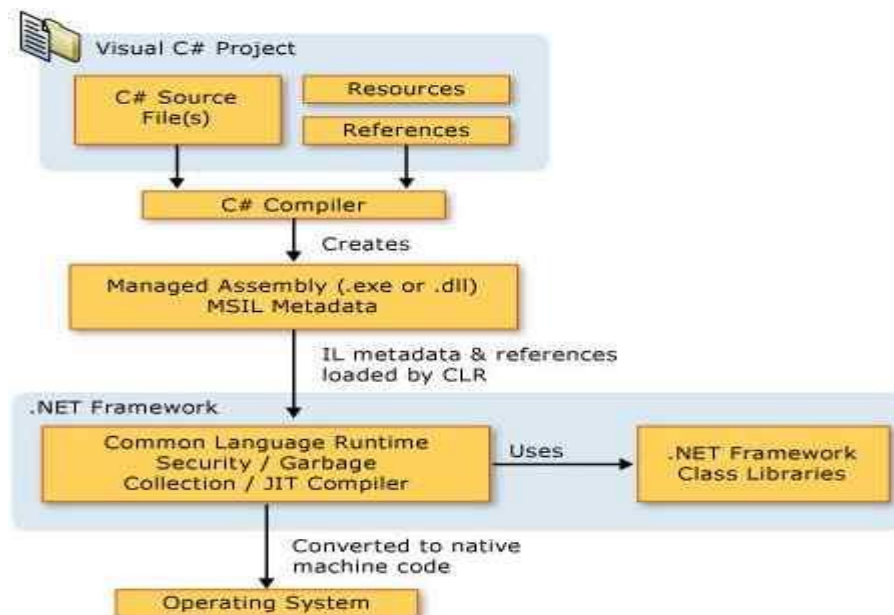
CLR runtime engine comes with set of services, which are classified as follows

CLR services

- Assembly Resolver
- Assembly Loader
- Type Checker
- COM marshalled
- Debug Manager
- Thread Support
- IL to Native compiler
- Exception Manager
- Garbage Collector

Question 3: What is CTS?

Answer: The Common Type System (CTS) standardizes the data types of all programming languages using .NET under the umbrella of .NET to a common data type for easy and smooth communication among these .NET languages.
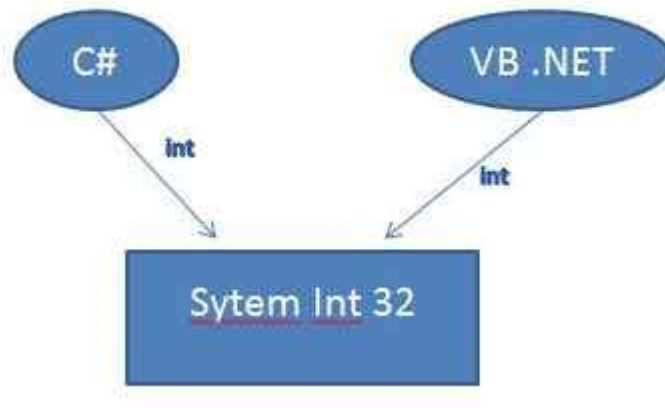
To implement or see how CTS is converting the data type to a common data type, for example, when we declare an int type data type in C# and VB.NET, then they are converted to int32. In other words, now both will have a common data type that provides flexible communication between these two languages.

Question 4: What is CLS?

Answer: One of the important goals of .NET Framework is to support Multiple Languages. This is achieved by CLS. For multiple languages to interoperate, it is necessary that they should go on in common in certain features such as Types that are used. For example, every language has its own size and range for different data types. Thus CLS is the agreement among language designers and class library designers concerning these usage conventions.
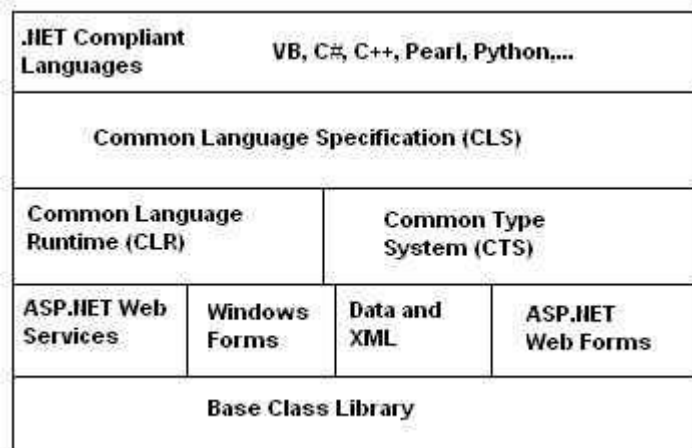
Question 5: What is managed code?

Answer: The resource, which is within your application domain is, managed code. The resources that are within domain are faster.

The code, which is developed in .NET framework, is known as managed code. This code is directly executed by CLR with help of managed code execution. Any language that is written in .NET Framework is managed code.

Managed code uses CLR which in turn looks after your applications by managing memory, handling security, allowing cross - language debugging, and so on.

Unmanaged Code Execution Process

Source Code

Any language written code of .NET Framework

CCW
(COM callable wrapper)

Native Compiler (Langauge Specific)

MSIL Code

Langauge independent within .NET Framework

JIT (Just-in-Time) Compiler

Executable Code

CPU

Output
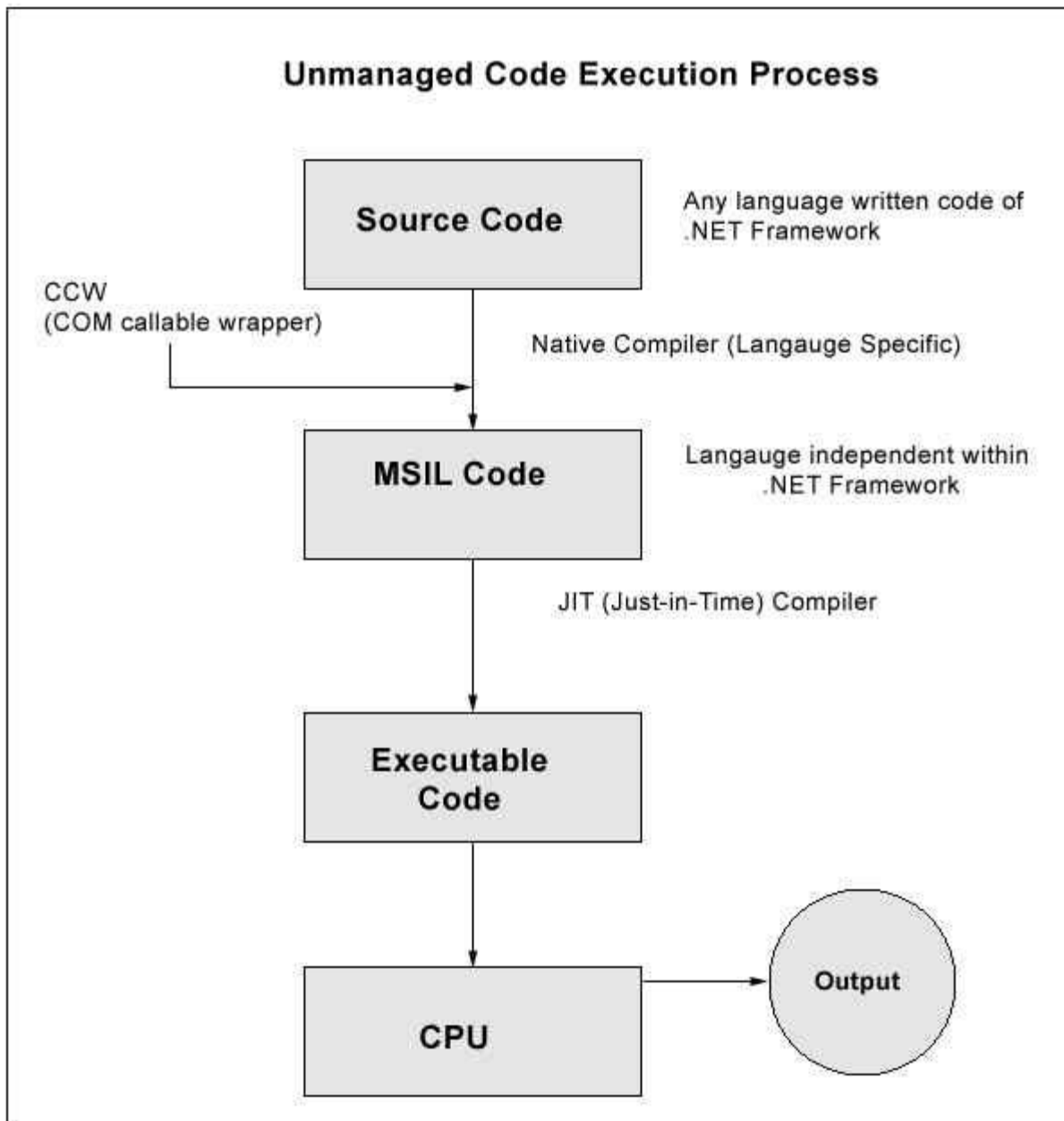
Question 6: What is MSIL?

Answer: When we compile our .NET code then it is not directly converted to native/binary code; it is first converted into intermediate code known as MSIL code which is then interpreted by the CLR. MSIL is independent of hardware and the operating system. Cross language relationships are possible since MSIL is the same for all .NET languages. MSIL is further converted into native code.

Code in any .Net Language	→ Language Compiler *compile time* →	MSIL Code	→ Just In Time Compiler *run time* →	Executable Native Code
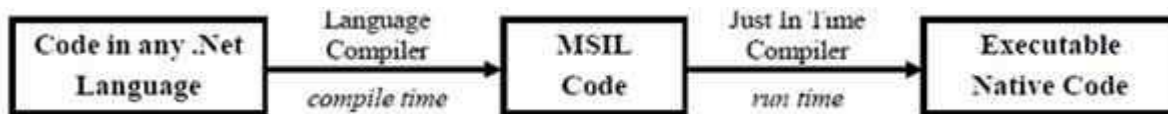
Question 7: What is JIT?

Answer: A Web Service or Web Forms file must be compiled to run within the CLR. Compilation can be implicit or explicit. Although you could explicitly call the appropriate compiler to compile your Web Service or Web Forms files, it is easier to allow the file to be complied implicitly. Implicit compilation occurs when you request the .asmx via HTTP-SOAP, HTTP-GET, or HTTP-POST. The parser (xsp.exe) determines whether a current version of the assembly resides in memory or in the disk. If it cannot use an existing version, the parser makes the appropriate call to the respective compiler (as you designated in the **Class** property of the .asmx page).

When the Web Service (or Web Forms page) is implicitly compiled, it is actually compiled twice. On the first pass, it is compiled into IL. On the second pass, the Web Service (now an assembly in IL) is compiled into machine language. This process is called Just-In-Time JIT compilation because it does not occur until the assembly is on the target machine.

JIT Types:
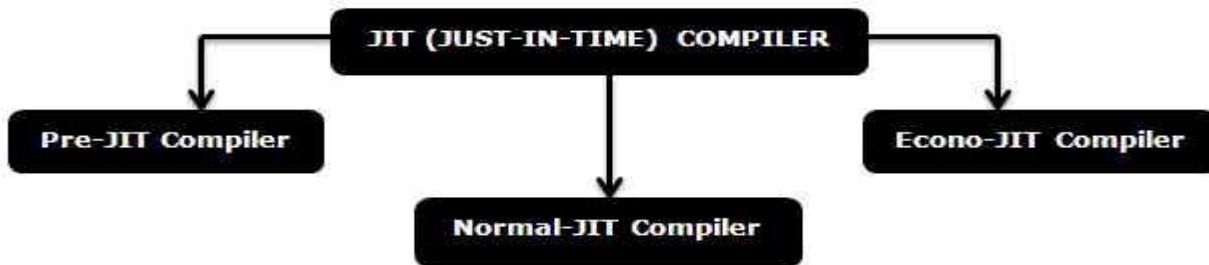
Question 8: What is portable executable (PE)?

Answer: Every .NET program first compiles with an appropriate compiler like if we write a program in C# language then it gets compiled by C# compiler (i.e. csc.exe).

In .NET framework every program executes (communicate) in an operating system by using CLR (Common Language Runtime).
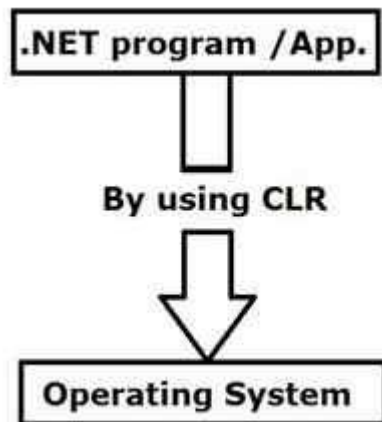
Managed module is standard windows Portable Executable (PE) file which contains the following parts.

- **PE Header-**

  It is similar to common object file format.

- **CLR Header-**

  This contains CLR version required to run this managed module, location & metadata. This also contains entry point of function i.e. the address of entry point of function.

- **Metadata-**

  This contains table information means variable with its data types and default values, functions / methods which are declared & defined in our program.

Question 9: What is an application domain?

Answer: An Application Domain is a logical container for a set of assemblies in which an executable is hosted. As you have seen, a single process may contain multiple Application Domains, each of which is hosting a .NET executable. The first appdomain created when the CLR is initialized is called the default AppDomain and this default one is destroyed when the Windows process is terminated.

- An AppDomain can be independently secured.
- An AppDomain can be unloaded.
- Independently configured.
- No mutual intervention by multiple appdomains.
- Performance
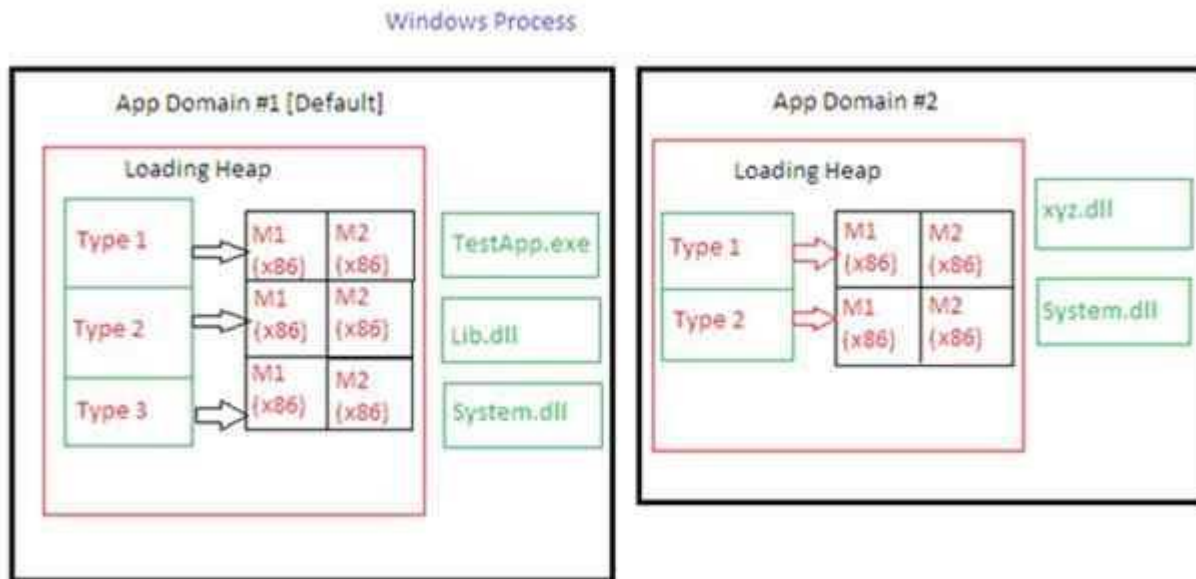
How does an AppDomain get created-

The AppDomain class is used to create and terminate Application Domains, load and unload assemblies and types and enumerates assemblies and threads in a domain. The following table shows some useful methods of the AppDomain class:

Methods	Description
CreateDomain()	It allows us to create a new Application Domain.
CreateInstance()	Creates an instance of type in an external assembly.
ExecuteAssembly()	It executes an *.exe assembly in the Application Domain.
Load()	This method dynamically loads an assembly into the current app domain.
UnLoad()	It allows us to unload a specified AppDomain within a given process.
GetCurrentThread()	Returns the ID of the active thread in the current Application Domain.

In addition, the AppDomain class also defined as a set of properties that can be useful when you

wish to monitor the activity of a given Application Domain.

Properties	Description
CurrentDomain	Gets the Application Domain for the currently executing thread.
FriendlyName	Gets the friendly name of the current Application Domain.
SetupInformation	Get the configuration details for a given Application Domain.
BaseDirectory	Gets the directory path that the assembly resolver uses to probe for assemblies.

Question 10: What is an assembly?

Answer: An Assembly is a basic building block of .NET Framework applications. It is basically compiled code that can be executed by the CLR. An assembly is a collection of types and resources that are built to work together and form a logical unit of functionality. An Assembly can be a DLL or exe depending upon the project that we choose.

Assemblies are basically the following two types:

1. Private Assembly
2. Shared Assembly

Question 11: What are the contents of assembly?

Answer: Assembly

- An Assembly is a basic unit of application deployment and versioning.
- An Assembly is also called the building block of a .NET application.
- An Assembly is either an .exe or .dll file.

An Assembly structure consists of the following parts:

- Assemblies manifest (name, language and version).
- CIL code (logic part).
- Type information (Datatype).
- Resources.

Question 12: What are the different types of assembly?

Answer: An Assembly contains metadata and manifest information. The reason for the emergence of assembly concept was to overcome the common "**DLL Hell**" problem in COM. The assembly contains all the code, resources, metadata and even version information. Metadata contains the details of every "type" inside the assembly. In addition to metadata, assemblies also have a special file called Manifest. It contains information about the current version of the assembly, culture information, public key token if available and other related information.

There are in all 3 different types of assemblies:

1. Private Assembly

2. Shared or Strong named assembly

3. Satellite assembly

Question 13: What is a dynamic assembly?

Answer: Technically, the act of loading external assemblies on demand is known as Dynamic Loading. Using the Assembly class, we can dynamically load both private and shared assemblies from the local location to a remote location as well as, explore its properties.

To illustrate dynamic loading, we are creating a console based application that loads an external TestLib.dll assembly. During the execution, the application asks the user to specify the dynamic loading assembly name and that reference is passed to the helper method that is responsible for loading the assembly.

Question 14: What is GAC?

Answer: The GAC is a shared location of computer where we can put an assembly so that it will be accessible from many locations; I mean it is accessible from another project or application. It's always a good practice to provide a strong name to a public assembly, I mean the assembly to be registered in the GAC, and otherwise the DLL hell problem may occur.

Problems that occurred-

I have seen DLLs added to the GAC that you can't remove - very frustrating. I have seen registered DLLs into the cache - verified everything is there ok using ILDASM only to find the DLLs are no longer in the GAC

Strongly naming the assembly-

When doing this make sure you get the directory slashes \\ correct within the assembly file (assembly.cs). - If not, you will get errors whilst the code is looking for the .snk file. If you get errors which leave you scratching your head - best bet is to remove the .snk file and start over.

Project References-

Also be careful and watch where you build projects as the referenced DLLs can easily be built to the development instead of the release folder - sometimes even when you specify the release folder. This can be very, very frustrating.

Conclusion-

My conclusion on using the GAC was only use it if you really need to as it isn't the 'end of DLL hell' as first thought. Also only use it if you are using a DLL that is shared by other projects. Don't put it in the GAC if you don't have to.
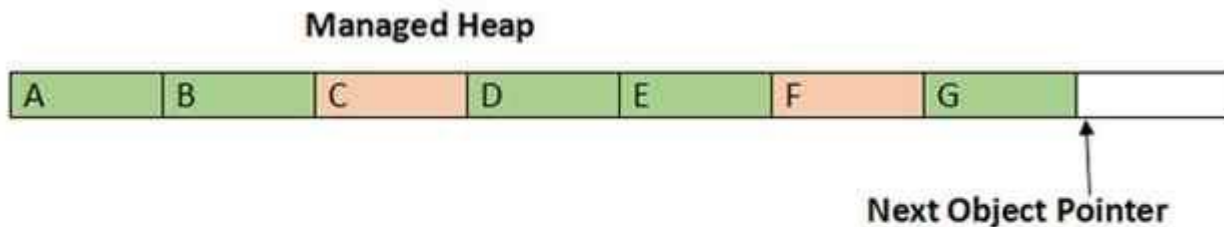
Question 15: What is a garbage collector?

Answer: The Garbage Collector (GC) is the part of the .NET Framework that allocates and releases memory for your .NET applications. The Common Language Runtime (CLR) manages allocation and deallocation of a managed object in memory. C# programmers never do this directly; there is no delete keyword in the C# language. It relies on the garbage collector.

Example:

Assume the managed heap contains a set of objects named A, B, C, D, E, F and G. During garbage collection, these objects are examined for active roots. After the graph has been constructed, unreachable objects (that we will assume are objects C and F) are marked as garbage in reddish color in the following diagram.


Question 16: What are generations and how are they used by the garbage collector?

Answer: Basically the generation of Garbage Collection (GC) shows the life of objects, it means it defines how long an object will stay in the memory. It's categorized into the following three generations:

- Generation 0
- Generation 1
- Generation 2

Generation of Garbage Collector in .NET Framework
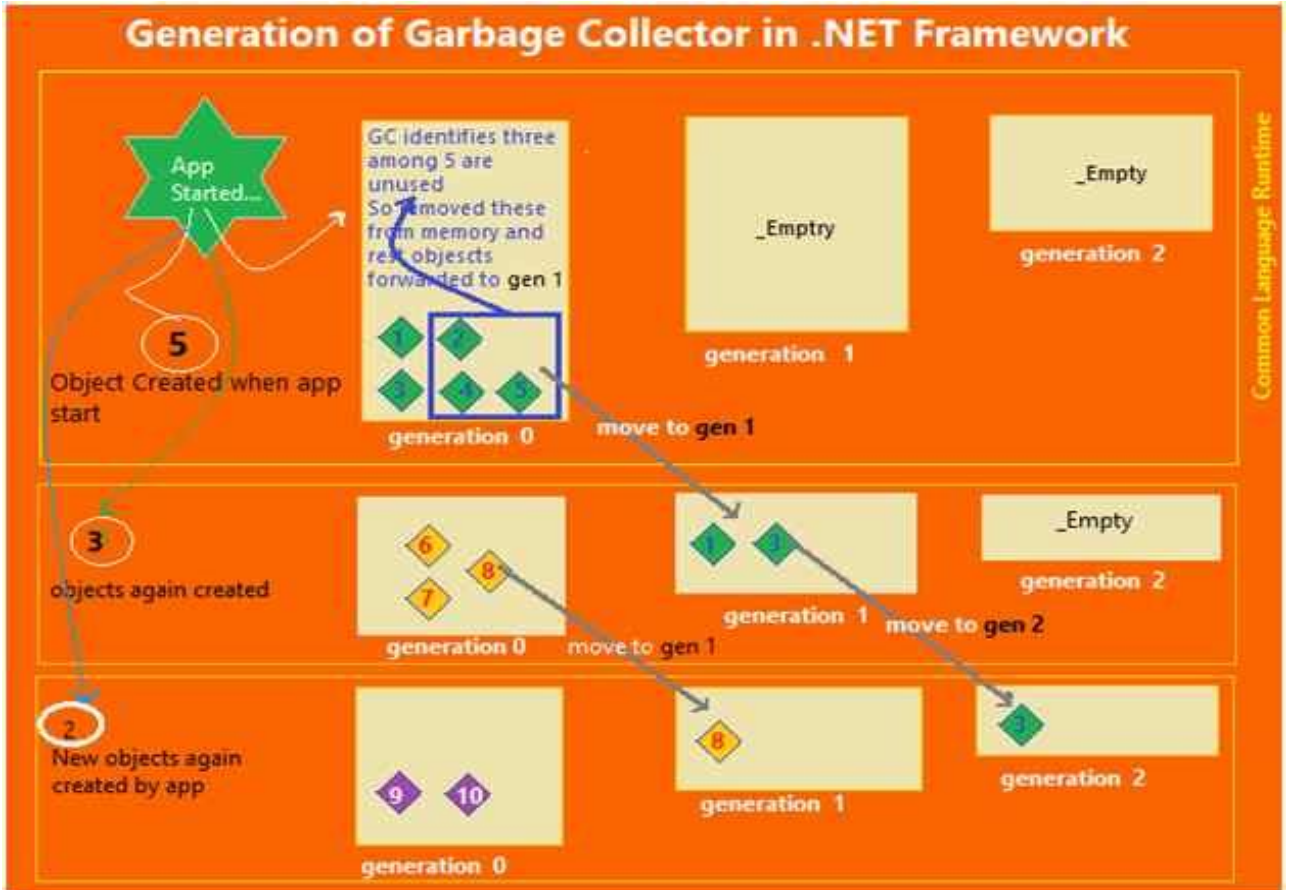
Chapter 2

C# Interview Questions and Answers

Question 1: What is C#?

Answer: C# is the best language for writing Microsoft .NET applications. C# provides the rapid application development found in Visual Basic with the power of C++. Its syntax is similar to C++ syntax and meets 100% of the requirements of OOPs like the following:

- Abstraction
- Encapsulation
- Polymorphism
- Inheritance

Question 2: What is an Object?

Answer: According to MSDN, "*a class or struct definition is like a blueprint that specifies what the type can do. An object is basically a block of memory that has been allocated and configured according to the blueprint. A program may create many objects of the same class. Objects are also called instances, and they can be stored in either a named variable or in an array or collection. Client code is the code that uses these variables to call the methods and access the public properties of the object. In an object-oriented language such as C#, a typical program consists of multiple objects interacting dynamically*".

Question 3: What is Managed or Unmanaged Code?

Answer: Managed Code- "The code, which is developed in .NET framework, is known as managed code. This code is directly executed by CLR with the help of managed code execution. Any language that is written in .NET Framework is managed code".

Unmanaged Code- The code, which is developed outside .NET framework, is known as unmanaged code.

"Applications that do not run under the control of the CLR are said to be unmanaged, and certain languages such as C++ can be used to write such applications, which, for example, access low - level functions of the operating system. Background compatibility with the code of VB, ASP and COM are examples of unmanaged code".

Unmanaged code can be unmanaged source code and unmanaged compile code. Unmanaged code is executed with the help of wrapper classes.

Wrapper classes are of two types:

- CCW (COM Callable Wrapper).
- RCW (Runtime Callable Wrapper).

Question 4: What is Boxing and Unboxing?

Answer: Boxing and Unboxing both are used for type conversion but have some difference:

Boxing: Boxing is the process of converting a value type data type to the object or to any interface data type which is implemented by this value type. When the CLR boxes a value means when CLR is converting a value type to Object Type, it wraps the value inside a System.Object and stores it on the heap area in application domain.

Example:

```
public void Function1()
{
    int i = 111;
    object o = i;//implicit Boxing
    Console.WriteLine(o);
}
```

Unboxing: Unboxing is also a process which is used to extract the value type from the object or any implemented interface type. Boxing may be done implicitly, but unboxing have to be explicit by code.

Example:

```
public void Function1()
{
    object o = 111;
    int i = (int)o;//explicit Unboxing
    Console.WriteLine(i);
}
```

The concept of boxing and unboxing underlines the C# unified view of the type system in which a value of any type can be treated as an object.

Question 5: What is the difference between a struct and a class in C#?

Answer: Class and Struct both are the user defined data type but have some major difference:

Struct-

- The struct is value type in C# and it inherits from System.Value Type.
- Struct is usually used for smaller amounts of data.
- Struct can't be inherited to other type.

- A structure can't be abstract.
- No need to create object by new keyword.
- Do not have permission to create any default constructor.

Class-

- The class is reference type in C# and it inherits from the System.Object Type.
- Classes are usually used for large amounts of data.
- Classes can be inherited to other class.
- A class can be abstract type.
- We can't use an object of a class with using new keyword.
- We can create a default constructor.

Question 6: What is the difference between Interface and Abstract Class?

Answer: Theoretically there are some differences between Abstract Class and Interface which are listed below:

- A class can implement any number of interfaces but a subclass can at most use only one abstract class.
- An abstract class can have non-abstract methods (concrete methods) while in case of interface all the methods has to be abstract.
- An abstract class can declare or use any variables while an interface is not allowed to do so.
- In an abstract class all data member or functions are private by default while in interface all are public, we can't change them manually.
- In an abstract class we need to use abstract keyword to declare abstract methods while in an interface we don't need to use that.
- An abstract class can't be used for multiple inheritance while interface can be used as multiple inheritance.
- An abstract class use constructor while in an interface we don't have any type of constructor.

Question 7: What is enum in C#?

Answer:

- An enum is a value type with a set of related named constants often referred to as an enumerator list. The enum keyword is used to declare an enumeration. It is a primitive data type, which is user defined.
- An enum type can be an integer (float, int, byte, double etc.). But if you used beside int it has to be cast.
- An enum is used to create numeric constants in .NET framework. All the members of enum are of enum type. There must be a numeric value for each enum type.

The default underlying type of the enumeration element is int. By default, the first enumerator has the value 0, and the value of each successive enumerator is increased by 1.

1. enum Dow {Sat, Sun, Mon, Tue, Wed, Thu, Fri};

Some points about enum-

- Enums are enumerated data type in c#.
- Enums are not for end-user, they are meant for developers.
- Enums are strongly typed constant. They are strongly typed, i.e. an enum of one type may not be implicitly assigned to an enum of another type even though the underlying value of their members is the same.
- Enumerations (enums) make your code much more readable and understandable.
- Enum values are fixed. Enum can be displayed as a string and processed as an integer.
- The default type is int, and the approved types are byte, sbyte, short, ushort, uint, long, and ulong.
- Every enum type automatically derives from System.Enum and thus we can use System.Enum methods on enums.
- Enums are value types and are created on the stack and not on the heap.

Question 8: What is the difference between "continue" and "break" statements in C#?

Answer: Using break statement, you can 'jump out of a loop' whereas by using continue statement, you can 'jump over one iteration' and then resume your loop execution.

Break Statement Example-

```
1.  using System;
2.  using System.Collections;
3.  using System.Linq;
4.  using System.Text;
5.   namespace break_example {
6.     {
7.        Class brk_stmt {
8.           public static void main(String[] args) {
9.              for (int i = 0; i <= 5; i++) {
10.                 if (i == 4) {
11.                    continue;  }
12.                 Console.ReadLine("The number is" + i); }} }  }
```

Output:

The number is 0;
The number is 1;
The number is 2;
The number is 3;

Continue Statement Example-

```
1.  using System;
2.  using System.Collections;
3.  using System.Linq;
4.  using System.Text;
5.   namespace continue_example
6.  {
7.     Class cntnu_stmt
8.     {
9.        public static void main(String[]
10.       {
```

```
11.        for (int i = 0; i <= 5; i++)
12.        {
13.           if (i == 4)
14.           {
15.              continue; }
16.           Console.ReadLine("The number is" +i); } } } }
```

Output:

The number is 1;
The number is 2;
The number is 3;
The number is 5;

|

Question 9: What is the difference between constant and read only in c#?

Answer: Constant (const) and **Readonly** (readonly) both looks like same as per the uses but they have some differences:

Constant is known as "const" keyword in C# which is also known immutable values which are known at compile time and do not change their values at run time like in any function or constructor for the life of application till the application is running.

Readonly is known as "readonly" keyword in C# which is also known immutable values and are known at compile and run time and do not change their values at run time like in any function for the life of application till the application is running. You can assay their value by constructor when we call constructor with "new" keyword.

See the example-

We have a Test Class in which we have two variables one is readonly and another is constant.

```
1.  class Test {
2.     readonly int read = 10;
3.     const int cons = 10;
4.     public Test() {
5.        read = 100;
6.        cons = 100;
7.     }
```

```
8.      public void Check() {
9.          Console.WriteLine("Read only : {0}", read);
10.         Console.WriteLine("const : {0}", cons);
11.     }
12. }
```

Here I was trying to change the value of both the variables in constructor but when I am trying to change the constant it gives an error to change their value in that block which have to call at run time.
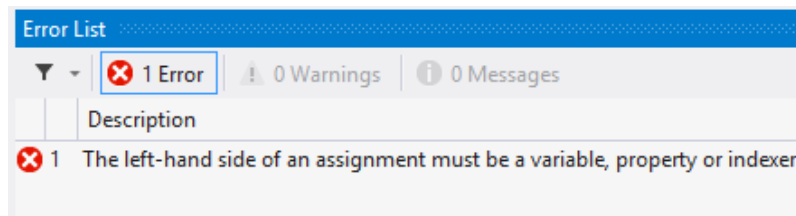
So finally remove that line of code from class and call this Check() function like the following code snippet:

```
1.   class Program {
2.      static void Main(string[] args) {
3.          Test obj = new Test();
4.          obj.Check();
5.          Console.ReadLine();
6.      }
7.   }
8.   class Test {
9.      readonly int read = 10;
10.     const int cons = 10;
11.     public Test() {
12.         read = 100;
13.     }
14.     public void Check() {
15.         Console.WriteLine("Read only : {0}", read);
16.         Console.WriteLine("const : {0}", cons);
17.     }
18. }
```

Output:

Question 10: What is the difference between ref and out keywords?

Answer: In C Sharp (C#) we can have three types of parameters in a function. The parameters can be in parameter (which is not returned back to the caller of the function), out parameter and ref parameter. We have lots of differences in both of them.

Ref	Out
The parameter or argument must be initialized first before it is passed to ref.	It is not compulsory to initialize a parameter or argument before it is passed to an out.
It is not required to assign or initialize the value of a parameter (which is passed by ref) before returning to the calling method.	A called method is required to assign or initialize a value of a parameter (which is passed to an out) before returning to the calling method.
Passing a parameter value by Ref is useful when the called method is also needed to modify the pass parameter.	Declaring a parameter to an out method is useful when multiple values need to be returned from a function or method.
It is not compulsory to initialize a parameter value before using it in a calling method.	A parameter value must be initialized within the calling method before its use.
When we use REF, data can be passed bi-directionally.	When we use OUT data is passed only in a unidirectional way (from the called method to the caller method).
Both ref and out are treated differently at run time and they are treated the same at compile time.	
Properties are not variables, therefore it cannot be passed as an out or ref parameter.	

Question 11: Can "this" be used within a static method?

Answer: We can't use this in static method because keyword 'this' returns a reference to the current instance of the class containing it. Static methods (or any static member) do not belong to a particular instance. They exist without creating an instance of the class and call with the name of a class not by instance so we can't use this keyword in the body of static Methods, but in case of Extension Methods we can use it the functions parameters. Let's have a look on "this" keyword.

The "this" keyword is a special type of reference variable that is implicitly defined within each constructor and non-static method as a first parameter of the type class in which it is defined. For example, consider the following class written in C#.

Question 12: Define Property in C# .net?

Answer: Properties are members that provide a flexible mechanism to read, write or compute the values of private fields, in other words by the property we can access private fields. In other words we can say that a property is a return type function/method with one parameter or without a parameter. These are always public data members. It uses methods to access and assign values to private fields called accessors.

Now question is what are accessors?

The get and set portions or blocks of a property are called accessors. These are useful to restrict the accessibility of a property, the set accessor specifies that we can assign a value to a private field in a property and without the set accessor property it is like a read-only field. By the get accessor we can access the value of the private field, in other words it returns a single value. A Get accessor specifies that we can access the value of a field publically.

We have the three types of properties

- Read/Write.
- ReadOnly.
- WriteOnly

Question 13: What is extension method in c# and how to use them?

Answer: Extension methods enable you to add methods to existing types without creating a new derived type, recompiling, or otherwise modifying the original type. An extension method is a special kind of static method, but they are called as if they were instance methods on the extended type.

How to use extension methods?

An extension method is a static method of a static class, where the "this" modifier is applied to the first parameter. The type of the first parameter will be the type that is extended.

Extension methods are only in scope when you explicitly import the namespace into your source code with a using directive.

Like: suppose we have a class like bellow:

```
1.  public class Class1 {
2.      public string Display() {
3.          return ("I m in Display");
4.      }
5.      public string Print() {
6.          return ("I m in Print");
7.      }
8.  }
```

Now we need to extend the definition of this class so m going to create a static class to create an extinction method like:

```
1.  public static class XX {
2.      public static void NewMethod(this Class1 ob) {
3.          Console.WriteLine("Hello I m extended method");
4.      }
5.  }
```
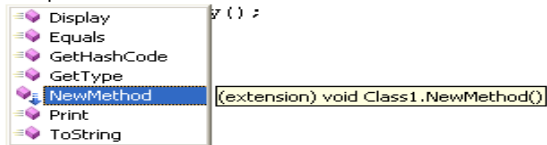
Here I just create a method that name is NewMethod with a parameter using this to define which type of data I need to be extend, now let's see how to use this function.

```
using System;
using System.Text;
using ClassLibExtMethod;

namespace ExtensionMethod1
{
    public static class XX
    {
        public static void NewMethod(this Class1 ob)
        {
            Console.WriteLine("Hello I m extended method");
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            Class1 ob = new Class1();
            ob.Display();
            ob.Print();
            ob.|                  y();

              Display
              Equals
              GetHashCode
              GetType
              NewMethod        (extension) void Class1.NewMethod()
              Print
              ToString
        }
    }
}
```
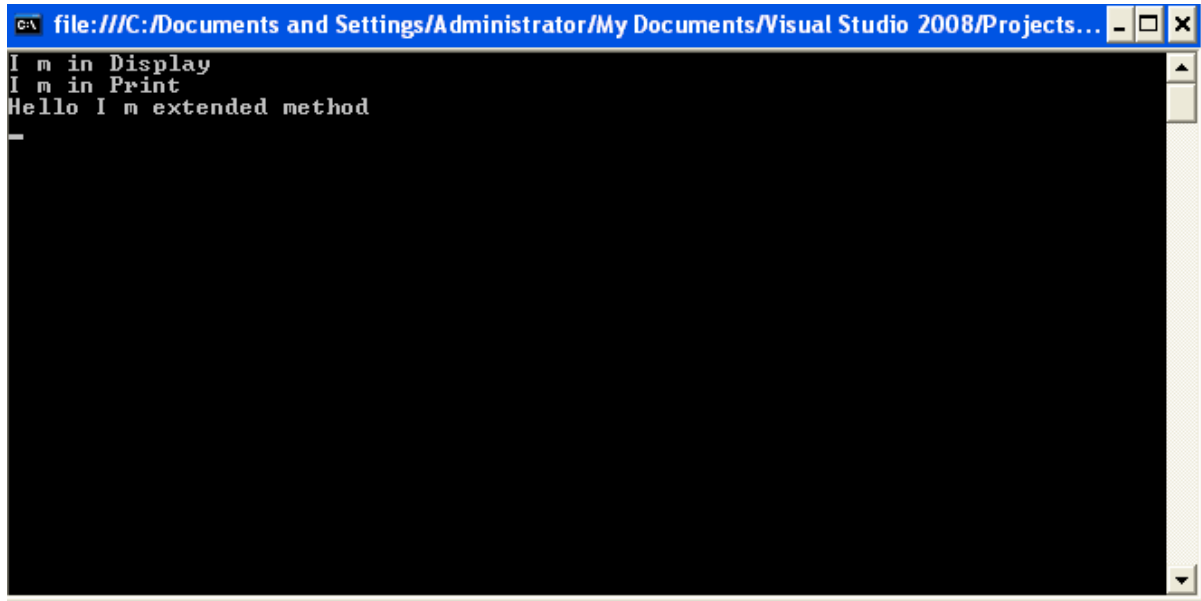
Code will look like that:

1. class Program {
2.   static void Main(string[] args) {
3.   Class1 ob = new Class1();
4.   ob.Display();
5.   ob.Print();
6.   ob.NewMethod();
7.   Console.ReadKey();
8.   }
9. }

Output will be:

```
file:///C:/Documents and Settings/Administrator/My Documents/Visual Studio 2008/Projects...
I m in Display
I m in Print
Hello I m extended method
```

Question 14: What is the difference between dispose and finalize methods in c#?

Answer: Finalize and dispose both are used for same task like to free unmanaged resources but have some differences see.

Finalize:

- Finalize used to free unmanaged resources those are not in use like files, database connections in application domain and more, held by an object before that object is destroyed.
- In the Internal process it is called by Garbage Collector and can't called manual by user code or any service.
- Finalize belongs to System.Object class.
- Implement it when you have unmanaged resources in your code, and make sure that these resources are freed when the Garbage collection happens.

Dispose:

- Dispose is also used to free unmanaged resources those are not in use like files, database connections in Application domain at any time.
- Dispose explicitly it is called by manual user code.
- If we need to dispose method so must implement that class by IDisposable interface.
- It belongs to IDisposable interface.
- Implement this when you are writing a custom class that will be used by other users.

Question 15: What is the difference between string and StringBuilder in c#?

Answer: StringBuilder and string both use to store string value but both have many differences on the bases of instance creation and also for performance:

String: String is an immutable object. Immutable like when we create string object in code so we cannot modify or change that object in any operations like insert new value, replace or append any value with existing value in string object, when we have to do some operations to change string simply it will dispose the old value of string object and it will create new instance in memory for hold the new value in string object like:

```
class Program
{
    static void Main(string[] args)
    {
        string val = "Hello";
        // create a new string instance instead of changing the old one
        val += "am ";
        val += "Nitin Pandit";
        Console.WriteLine(val);
    }
}
```

Note:

- It's an immutable object that holds string value.

- Performance wise string is slow because its' create a new instance to override or change the previous value.

- String belongs to System namespace.

StringBuilder:

System.Text.Stringbuilder is mutable object which also hold the string value, mutable means once we create a System.Text.Stringbuilder object we can use this object for any operation like insert value in existing string with insert functions also replace or append without creating new instance of System.Text.Stringbuilder for every time so it's use the previous object so it's work fast as compare than System.String. Let's have an example to understand System.Text.Stringbuilder like:

```
class Program
{
    static void Main(string[] args)
    {
        StringBuilder val = new StringBuilder("");
        val.Append("hello");
        val.Append(" am Nitin Pandit :)");
        Console.WriteLine(val);
    }
}
```

Note:

- StringBuilder is a mutable object.

- Performance wise StringBuilder is very fast because it will use same instance of

  StringBuilder object to perform any operation like insert value in existing string.

- StringBuilder belongs to System.Text.Stringbuilder namespace.

Question 16: What are delegates in C# and uses of delegates?

Answer: C# delegates are same as pointers to functions, in C or C++. A delegate Object is a reference type variable that use to holds the reference to a method. The reference can be changed at runtime which is hold by an object of delegate, a delegate object can hold many functions reference which is also known as Invocation List that refers functions in a sequence FIFO, we can new functions ref in this list at run time by += operator and can remove by -= operator.

Delegates are especially used for implementing events and the call-back methods. All delegates are implicitly derived from the System.Delegate class.

Let's see how to use Delegate with Example:

```csharp
class Program
{
    static void Main(string[] args)
    {
        TestDelegate obj = new TestDelegate();
        obj.delObject("Nitin");
    }
}
delegate void Del(string UserName);
class TestDelegate
{
    public Del delObject;
    public TestDelegate()
    {
        delObject = new Del(this.SayHello);
    }
    public void SayHello(string UserName)
    {
        Console.WriteLine("Hello.." + UserName);
    }
}
```

Question 17: What is sealed class in c#?

Answer: Sealed classes are used to restrict the inheritance feature of object oriented programming. Once a class is defined as a sealed class, the class cannot be inherited.

In C#, the sealed modifier is used to define a class as sealed. In Visual Basic .NET the Not Inheritable keyword serves the purpose of sealed. If a class is derived from a sealed class then the compiler throws an error.

If you have ever noticed, structs are sealed. You cannot derive a class from a struct.

The following class definition defines a sealed class in C#:

1. // Sealed class
2. sealed class SealedClass
3. {
4.
5. }

Question 18: What are partial classes?

Answer: A partial class is only use to splits the definition of a class in two or more classes in a same source code file or more than one source files. You can create a class definition in multiple files but it will be compiled as one class at run time and also when you'll create an instance of this class so you can access all the methods from all source file with a same object.

Partial Classes can be create in the same namespace it's doesn't allowed to create a partial class in different namespace. So use "partial" keyword with the entire class name which you want to bind together with the same name of class in same namespace, let's have an example:
Example:

```
partial class Class1
 {
     public void Function1()
     {
         Console.WriteLine("Function 1 ");
     }
 }
partial class Class1
{
     public void Function2()
     {
         Console.WriteLine("Function 2 ");
     }
}
 class Program
 {
     static void Main(string[] args)
     {
         Class1 obj = new Class1();
         obj.Function1();
         obj.Function2();
         Console.ReadLine();
     }
 }
```

Question 19: What is IEnumerable<> in c#?

Answer: IEnumerable is the parent interface for all non-generic collections in System.Collections namespace like ArrayList, HastTable etc. that can be enumerated. For the generic version of this interface as IEnumerable<T> which a parent interface of all generic collections class in System.Collections.Generic namespace like List<> and more.

In System.Collections.Generic.IEnumerable<T> have only a single method which is GetEnumerator() that returns an IEnumerator. IEnumerator provides the power to iterate through the collection by exposing a Current property and Move Next and Reset methods, if we doesn't have this interface as a parent so we can't use iteration by foreach loop or can't use that class object in our LINQ query.

```
•O System.Collections.Generic.IEnumerable<out T>                                    ▼

  ⊞ Assembly mscorlib.dll, v4.0.0.0

   using System.Collections;

 ⊟namespace System.Collections.Generic
  {
 ⊟     // Summary:
       //     Exposes the enumerator, which supports a simple iteration over a collection
       //     of a specified type.To browse the .NET Framework source code for this type,
       //     see the Reference Source.
       //
       // Type parameters:
       //   T:
       //     The type of objects to enumerate.This type parameter is covariant. That is,
       //     you can use either the type you specified or any type that is more derived.
       //     For more information about covariance and contravariance, see Covariance
       //     and Contravariance in Generics.
 ⊟     public interface IEnumerable<out T> : IEnumerable
       {
 ⊟         // Summary:
           //     Returns an enumerator that iterates through the collection.
           //
           // Returns:
           //     An enumerator that can be used to iterate through the collection.
           IEnumerator<T> GetEnumerator();
       }
  }
```

Question 20: What is difference between late binding and early binding in c#?

Answer: Early Binding and Late Binding concepts belongs to polymorphism so let's see first about polymorphism:

Polymorphism is an ability to take more than one form of a function means with a same name we can write multiple functions code in a same class or any derived class.

Polymorphism we have 2 different types to achieve that:

- Compile Time also known as Early Binding or Overloading.
- Run Time also known as Late Binding or Overriding.

Compile Time Polymorphism or Early Binding: In Compile time polymorphism or Early Binding we will use multiple methods with same name but different type of parameter or may be the number or parameter because of this we can perform different-different tasks with same method name in the same class which is also known as Method overloading.

See how we can do that by the following example:

```
class MyMath
{
    public int Sum(int val1, int val2)
    {
        return val1 + val2;
    }
    public string Sum(string val1, string val2)
    {
        return val1 + " " + val2;
    }
}
```

Run Time Polymorphism or Late Binding: Run time polymorphism also known as late binding, in Run Time polymorphism or Late Binding we can do use same method names with same signatures means same type or same number of parameters but not in same class because compiler doesn't allowed that at compile time so we can use in derived class that bind at run time when a child class or derived class object will instantiated that's way we says that Late Binding. For that we have to create my parent class functions as partial and in driver or child class as override functions with override keyword.

Like as following example:

```
class Class1
{
    public virtual string TestFunction()
    {
        return "Hello";
    }
}
class Class2 : Class1
{
    public override string TestFunction()
    {
        return "Bye Bye";
    }
}
class Program
{
    static void Main(string[] args)
    {
        Class2 obj = new Class2();
        Console.WriteLine(obj.TestFunction());
        Console.ReadLine();
    }
}
```

Question 21: What are the differences between IEnumerable and IQueryable?

Answer: Before the differences learn what is IEnumerable and IQueryable.

IEnumerable:

Is the parent interface for all non-generic collections in System.Collections namespace like ArrayList, HastTable etc. that can be enumerated. For the generic version of this interface as IEnumerable<T> which a parent interface of all generic collections class in System.Collections.Generic namespace like List<> and more.

IQueryable:

As per MSDN IQueryable interface is intended for implementation by query providers. It is only supposed to be implemented by providers that also implement IQueryable<T>. If the provider does not also implement IQueryable<T>, the standard query operators cannot be used on the provider's data source.

The IQueryable interface inherits the IEnumerable interface so that if it represents a query, the results of that query can be enumerated. Enumeration causes the expression tree associated with an IQueryable object to be executed. The definition of "executing an expression tree" is specific to a query provider. For example, it may involve translating the expression tree to an appropriate query language for the underlying data source. Queries that do not return enumerable results are executed when the Execute method is called.

IEnumerable	IQueryable
IEnumerable belongs to **System.Collections** namespace.	IQueryable belongs to **System.Linq** namespace.
IEnumerable is the best way to write query on collections data type like List, Array etc.	IQueryable is the best way to write query data like remote database, service collections.
IEnumerable is the return type for LINQ to Object and LINQ to XML queries.	IQueryable is the return type of LINQ to SQL queries.
IEnumerable doesn't support lazy loading. So it's not a recommended approach for paging kind of scenarios.	IQueryable support lazy loading so we can also use in paging kind of scenarios.
Extension methods are supports by IEnumerable takes functional objects for LINQ Query's.	IQueryable implements IEnumerable so indirectly it's also supports Extensions methods.

Question 22: What happens if the inherited interfaces have conflicting method names?
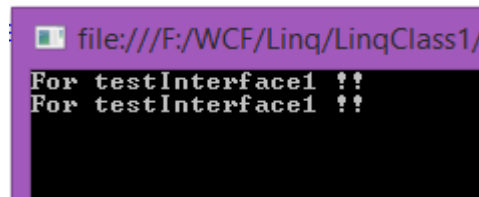
Answer: If we implement multipole interface in the same class with conflict method name so we don't need to define all or in other words we can say if we have conflict methods in same class so we can't implement their body independently in the same class coz of same name and same signature so we have to use interface name before method name to remove this method confiscation let's see an example:

```
1.  interface testInterface1 {
2.      void Show();  }
3.  interface testInterface2 {
4.      void Show();  }
5.  class Abc: testInterface1,
6.  testInterface2 {
7.     void testInterface1.Show() {
8.         Console.WriteLine("For testInterface1 !!");   }
9.     void testInterface2.Show() {
10.        Console.WriteLine("For testInterface2 !!");
11.    }
12. }
```

Now see how to use those in a class:

```
1.  class Program {
2.     static void Main(string[] args) {
3.         testInterface1 obj1 = new Abc();
4.         testInterface1 obj2 = new Abc();
5.         obj1.Show();
6.         obj2.Show();
7.      Console.ReadLine();   }  }
```



```
file:///F:/WCF/Linq/LinqClass1/
For testInterface1 !!
For testInterface1 !!
```

Question 23: What are the Arrays in C#.Net?

Answer: Arrays are powerful data structures for solving many programming problems. You saw during the creation of variables of many types that they have one thing in common; they hold information about a single item, for instance an integer, float and string type and so on. So what is the solution if you need to manipulate sets of items? One solution would be to create a variable for each item in the set but again this leads to a different problem. How many variables do you need?

So in this situation Arrays provide mechanisms that solves problem posed by these questions. An array is a collection of related items, either value or reference type. In C# arrays are immutable such that the number of dimensions and size of the array are fixed.

Arrays Overview-

An array contains zero or more items called elements. An array is an unordered sequence of elements. All the elements in an array are of the same type (unlike fields in a class that can be of different types). The elements of an array accessed using an integer index that always starts from zero. C# supports single-dimensional (vectors), multidimensional and jagged arrays.

Elements are identified by indexes relative to the beginning of the arrays. Indexes are also commonly called indices or subscripts and are placed inside the indexing operator ([]). Access to array elements is by their index value that ranges from 0 to (length-1).

Array Properties

- The length cannot be changed once created.
- Elements are initialized to default values.
- Arrays are reference types and are instances of System.Array.
- Their number of dimensions or ranks can be determined by the Rank property.
- An array length can be determined by the GetLength() method or Length property.

Question 24: What is the Constructor Chaining in C#?

Answer: Constructor Chaining is a way to connect two or more classes in a relationship as Inheritance, in Constructor Chaining every child class constructor is mapped to parent class Constructor implicitly by base keyword so when you create an instance of child class to it'll call parent's class Constructor without it inheritance is not possible.

Question 25: What's the difference between the System.Array.CopyTo() and System.Array.Clone()?

Answer:

Clone - Method creates a shallow copy of an array. A shallow copy of an Array copies only the elements of the Array, whether they are reference types or value types, but it does not copy the objects that the references refer to. The references in the new Array point to the same objects that the references in the original Array point to.

CopyTo - The Copy static method of the Array class copies a section of an array to another array. The CopyTo method copies all the elements of an array to another one-dimension array. The code listed in Listing 9 copies contents of an integer array to an array of object types.

Question 26: Can Multiple Catch Blocks executed in c#?

Answer: We can use multiple Catches block with every try but when any Exceptions is throw by debugger so every catches match this exception type with their signature and catch the exception by any single catch block so that means we can use multiple catches blocks but only one can executed at once like:

```
1.  using System;
2.  class MyClient {
3.    public static void Main() {
4.      int x = 0;
5.      int div = 0;
6.      try {
```

```
7.          div = 100 / x;
8.          Console.WriteLine("Not executed line");
9.      } catch (DivideByZeroException de) {
10.         Console.WriteLine("DivideByZeroException");
11.     } catch (Exception ee) {
12.         Console.WriteLine("Exception");
13.     } finally {
14.         Console.WriteLine("Finally Block");
15.     }
16.     Console.WriteLine("Result is {0}", div);
17.   }
18. }
```

Question 27: What is Singleton Design Patterns and How to implement in C#?

Answer: Singleton Design Pattern-

1. Ensures a class has only one instance and provides a global point of access to it.

2. A singleton is a class that only allows a single instance of itself to be created, and usually gives simple access to that instance.

3. Most commonly, singletons don't allow any parameters to be specified when creating the instance, since a second request of an instance with a different parameter could be problematic! (If the same instance should be accessed for all requests with the same parameter then the factory pattern is more appropriate.)

4. There are various ways to implement the Singleton Pattern in C#. The following are the common characteristics of a Singleton Pattern.

Some key points:-

- A single constructor, that is private and parameterless.

- The class is sealed.

- A static variable that holds a reference to the single created instance, if any.

- A public static means of getting the reference to the single created instance, creating one if necessary.

This is the example how to write the code with Singleton:

```
1.   namespace Singleton {
2.      class Program {
3.        static void Main(string[] args) {
4.          Calculate.Instance.ValueOne = 10.5;
5.          Calculate.Instance.ValueTwo = 5.5;
6.          Console.WriteLine("Addition : " + Calculate.Instance.Addition());
7.          Console.WriteLine("Subtraction : " + Calculate.Instance.Subtraction());
8.          Console.WriteLine("Multiplication : " + Calculate.Instance.Multiplication());
9.          Console.WriteLine("Division : " + Calculate.Instance.Division());
10.         Console.WriteLine("\n---------------------\n");
11.         Calculate.Instance.ValueTwo = 10.5;
12.         Console.WriteLine("Addition : " + Calculate.Instance.Addition());
13.         Console.WriteLine("Subtraction : " + Calculate.Instance.Subtraction());
14.         Console.WriteLine("Multiplication : " + Calculate.Instance.Multiplication());
15.         Console.WriteLine("Division : " + Calculate.Instance.Division());
16.         Console.ReadLine();
17.       }
18.     }
19.     public sealed class Calculate {
20.       private Calculate() {}
21.       private static Calculate instance = null;
22.       public static Calculate Instance {
23.         get {
24.           if (instance == null) {
25.             instance = new Calculate();
26.           }
27.           return instance;
```

```
28.          }
29.        }
30.      public double ValueOne {
31.          get;
32.          set;
33.        }
34.      public double ValueTwo {
35.          get;
36.          set;
37.        }
38.      public double Addition() {
39.          return ValueOne + ValueTwo;
40.        }
41.      public double Subtraction() {
42.          return ValueOne - ValueTwo;
43.        }
44.      public double Multiplication() {
45.          return ValueOne * ValueTwo;
46.        }
47.      public double Division() {
48.          return ValueOne / ValueTwo;
49.        }
50.    }
51. }
```

Question 28: Difference between Throw Exception and Throw Clause.

Answer: The basic difference is that the Throw exception overwrites the stack trace and this makes it hard to find the original code line number that has thrown the exception.

Throw basically retains the stack information and adds to the stack information in the exception that it is thrown.

Let us see what it means rather speaking so many words to better understand the differences. I am using a console application to easily test and see how the usage of the two differs in their functionality.

```
1.  using System;
2.  using System.Collections.Generic;
```

```
3.   using System.Linq;
4.   using System.Text;
5.   namespace TestingThrowExceptions {
6.      class Program {
7.         public void ExceptionMethod() {
8.            throw new Exception("Original Exception occurred in ExceptionMethod");
9.         }
10.        static void Main(string[] args) {
11.           Program p = new Program();
12.           try {
13.              p.ExceptionMethod();
14.           } catch (Exception ex) {
15.              throw ex;
16.           }
17.        }
18.     }
19. }
```

Now run the code by pressing the F5 key of the keyboard and see what happens. It returns an exception and look at the stack trace:

Question 29: What is Indexer in C# .Net?

Answer: Indexer allows classes to be used in more intuitive manner. C# introduces a new concept known as Indexers which are used for treating an object as an array. The indexers are usually known as smart arrays in C#. They are not essential part of object-oriented programming.

An indexer, also called an indexed property, is a class property that allows you to access a member variable of a class using the features of an array.

Defining an indexer allows you to create classes that act like virtual arrays. Instances of that class can be accessed using the [] array access operator.

Creating an Indexer:

```
1.  < modifier > <
2.  return type > this[argument list] {
3.      get {
4.          // your get block code
5.      }
6.      set {
7.          // your set block code
8.      }
9.  }
```

In the above code:

<modifier> - can be private, public, protected or internal.

<return type> - can be any valid C# types.

Question 30: What is multicast delegate in c#?

Answer: Delegate can invoke only one method reference has been encapsulated into the delegate.it is possible for certain delegate to hold and invoke multiple methods such delegate called multicast delegates.multicast delegates also know as combinable delegates, must satisfy the following conditions:

- The return type of the delegate must be void. None of the parameters of the delegate type can be delegate type can be declared as output parameters using out keywords.
- Multicast delegate instance that created by combining two delegates, the invocation list is formed by concatenating the invocation list of two operand of the addition operation. Delegates are invoked in the order they are added.

Implement Multicast Delegates Example:

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Linq;
4.  using System.Text;
5.  delegate void MDelegate();
6.  class DM {
7.      static public void Display() {
```

```
8.        Console.WriteLine("Meerut")    }
9.     static public void print() {
10.       Console.WriteLine("Roorkee"); }
11. }
12. class MTest {
13.    public static void Main() {
14.       MDelegate m1 = new MDelegate(DM.Display);
15.       MDelegate m2 = new MDelegate(DM.print);
16.       MDelegate m3 = m1 + m2;
17.       MDelegate m4 = m2 + m1;
18.       MDelegate m5 = m3 - m2;
19.       m3();
20.       m4();
21.       m5();
22.    }
23. }
```

Question 31: Difference between Equality Operator (==) and Equals() Method in C#.

Answer: Both the == Operator and the Equals() method are used to compare two value type data items or reference type data items. The Equality Operator (==) is the comparison operator and the Equals() method compares the contents of a string. The == Operator compares the reference identity while the Equals() method compares only contents. Let's see with some examples.

In this example we assigned a string variable to another variable. A string is a reference type and in the following example, a string variable is assigned to another string variable so they are referring to the same identity in the heap and both have the same content so you get True output for both the == Operator and the Equals() method.

```
1.  using System;

2.  namespace ComparisionExample {

3.     class Program {

4.        static void Main(string[] args) {

5.           string name = "sandeep";
```

```
6.          string myName = name;
7.          Console.WriteLine("== operator result is {0}", name == myName);
8.          Console.WriteLine("Equals method result is {0}", name.Equals(myName));
9.          Console.ReadKey();
10.     }
11.   }
12. }
```

Question 32: Difference between "is" and "as" operator in C#.

Answer: "is" operator-

In the C# language, we use the "is" operator to check the object type. If the two objects are of the same type, it returns true and false if not.

Let's understand the preceding from a small program. We defined the following two classes:

```
1.  class Speaker {
2.     public string Name {
3.        get;
4.        set;
5.     }
6.  }
7.  class Author {
8.     public string Name {
9.        get;
10.       set;
11.    }
12. }
```

Now, let's try to check the preceding types as:

```
1.  var speaker = new Speaker { Name="Gaurav Kumar Arora"};
```

We declared an object of Speaker as in the following:

1. var isTrue = speaker is Speaker;

In the preceding, we are just checking the matching type. Yes, our speaker is an object of Speaker type.

1. Console.WriteLine("speaker is of Speaker type:{0}", isTrue);

So, the results as true.

But, here we get false:

1. var author = new Author { Name = "Gaurav Kumar Arora" };
2. var isTrue = speaker is Author;
3. Console.WriteLine("speaker is of Author type:{0}", isTrue);

Because our speaker is not an object of Author type.

"as" operator-

The "as" operator behaves similar to the "is" operator. The only difference is it returns the object if both are compatible to that type else it returns null.

Let's understand the preceding with a small snippet as in the following:

1. public static string GetAuthorName(dynamic obj)

2. {

3. Author authorObj = obj as Author;

4. return (authorObj != null) ? authorObj.Name : string.Empty;

5. }

We have a method that accepts dynamic objects and returns the object name property if the object is of the Author type.

Here, we declared two objects:

1. var speaker = new Speaker { Name="Gaurav Kumar Arora"};

2. var author = new Author { Name = "Gaurav Kumar Arora" };

The following returns the "Name" property:

1. var authorName = GetAuthorName(author);
2. Console.WriteLine("Author name is:{0}", authorName);

It returns an empty string:

1. authorName = GetAuthorName(speaker);
2. Console.WriteLine("Author name is:{0}", authorName);

Question 33: How to use Nullable<> Types in .Net?

Answer: A nullable Type is a data type is that contain the defined data type or the value of null. You should note here that here variable datatype has been given and then only it can be used.

This nullable type concept is not comaptible with "var".

I will explain this with syntax in next section.

Declaration:

Any DataType can be declared nullable type with the help of operator "?".

Example of the syntax is as Follows:-

1. int? i = null;

As discussed in previous section "var" is not compatible with this Nullable Type. So we will have Compile Time error if we are declaring something like: -

1. var? i = null;

Though following syntax is completely fine:-

1. var i = 4;

Question 34: Different Ways of Method can be overloaded.

Answer: Method overloading is a way to achieve compile time Polymorphism where we can use a method with the same name but different signature, Method overloading is done at compile time and we have multiple way to do that but in all way method name should be same.

- Number of parameter can be different.
- Types of parameter can be different.
- Order of parameters can be different.

Example:

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Linq;
4.  using System.Text;
5.  namespace Hello_Word {
6.     class overloding {
7.        public static void Main() {
8.           Console.WriteLine(volume(10));
9.           Console.WriteLine(volume(2.5F, 8));
10.          Console.WriteLine(volume(100L, 75, 15));
11.          Console.ReadLine();
12.       }
13.       static int volume(int x) {
14.          return (x * x * x);
15.       }
16.       static double volume(float r, int h) {
17.          return (3.14 * r * r * h);
18.       }
19.       static long volume(long l, int b, int h) {
20.          return (l * b * h);
21.       }
22.    }
23. }
```

Note:

If we have a method that have two parameter object type and have a same name method with two integer parameter so when we call that method with int value so it'll call that method have integer parameter instead of object type parameters method.

Question 35: What is an Object Pool in .Net?

Answer: Object Pooling is something that tries to keep a pool of objects in memory to be re-used later and hence it will reduce the load of object creation to a great extent. This article will try to explain this in detail. The example is for an Employee object, but you can make it general by using Object base class.

What does it mean?

Object Pool is nothing but a container of objects that are ready for use. Whenever there is a request for a new object, the pool manager will take the request and it will be served by allocating an object from the pool.

How it works?

We are going to use Factory pattern for this purpose. We will have a factory method, which will take care about the creation of objects. Whenever there is a request for a new object, the factory method will look into the object pool (we use Queue object). If there is any object available within the allowed limit, it will return the object (value object), otherwise a new object will be created and give you back.

Question 36: What are generics in c#.net?

Answer: Generics allow you to delay the specification of the data type of programming elements in a class or a method, until it is actually used in the program. In other words, generics allow you to write a class or method that can work with any data type.

You write the specifications for the class or the method, with substitute parameters for data types. When the compiler encounters a constructor for the class or a function call for the method, it generates code to handle the specific data type.
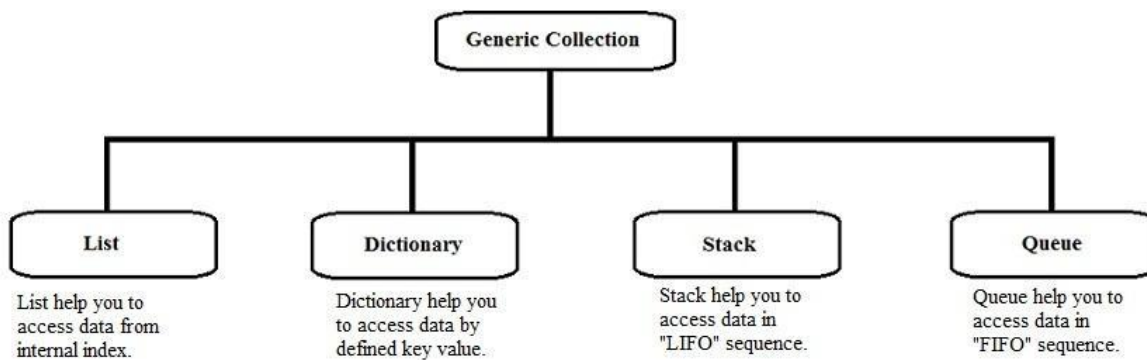
Generic classes and methods combine reusability, type safety and efficiency in a way that their non-generic counterparts cannot. Generics are most frequently used with collections and the methods that operate on them. Version 2.0 of the .NET Framework class library provides a new namespace, System.Collections.Generic, which contains several new generic-based collection classes. It is recommended that all applications that target the .NET Framework 2.0 and later use the new generic collection classes instead of the older non-generic counterparts such as ArrayList.

Features of Generics:

Generics is a technique that enriches your programs in the following ways:

- It helps you to maximize code reuse, type safety and performance.

- You can create generic collection classes. The .NET Framework class library contains several new generic collection classes in the System.Collections.Generic namespace. You may use these generic collection classes instead of the collection classes in the System.Collections namespace.

- You can create your own generic interfaces, classes, methods, events and delegates.

- You may create generic classes constrained to enable access to methods on specific data types.

- You may get information on the types used in a generic data type at run-time using reflection.

Question 37: Describe the accessibility modifiers in c#.Net.

Answer: Access modifiers are keywords used to specify the declared accessibility of a member or a type.

Why to use access modifiers?

Access modifiers are an integral part of object-oriented programming. They support the concept of encapsulation, which promotes the idea of hiding functionality. Access modifiers allow you to define who does or doesn't have access to certain features. In C# there are 5 different types of Access Modifiers:

Modifier	Description
public	There are no restrictions on accessing public members.
private	Access is limited to within the class definition. This is the default access modifier type if none is formally specified
protected	Access is limited to within the class definition and any class that inherits from the class
internal	Access is limited exclusively to classes defined within the current project assembly
protected internal	Access is limited to the current assembly and

Question 38: What is Virtual Method in C#?

Answer: A virtual method is a method that can be redefined in derived classes. A virtual method has an implementation in a base class as well as derived the class. It is used when a method's basic functionality is the same but sometimes more functionality is needed in the derived class. A virtual method is created in the base class that can be overridden in the derived class. We create a virtual method in the base class using the virtual keyword and that method is overridden in the derived class using the override keyword.

When a method is declared as a virtual method in a base class then that method can be defined in a base class and it is optional for the derived class to override that method. The overriding method

also provides more than one form for a method. Hence it is also an example for polymorphism.

When a method is declared as a virtual method in a base class and that method has the same definition in a derived class then there is no need to override it in the derived class. But when a virtual method has a different definition in the base class and the derived class then there is a need to override it in the derived class.

When a virtual method is invoked, the run-time type of the object is checked for an overriding member. The overriding member in the most derived class is called, which might be the original member, if no derived class has overridden the member.

Virtual Method:

1. By default, methods are non-virtual. We can't override a non-virtual method.
2. We can't use the virtual modifier with the static, abstract, private or override modifiers.

Question 39: What is the Difference between Array and ArrayList in C#.Net?

Answer: Difference between Array and ArrayList:

Array	ArrayList
Array uses the Vector array to store the elements	ArrayList uses the Linked List to store the elements.
Size of the Array must be defined until redeem used(vb)	No need to specify the storage size.
Array is a specific data type storage	ArrayList can be stored everything as object.
No need to do the type casting	Every time type casting has to do.
It will not lead to Runtime exception	It leads to the Run time error exception.
Element cannot be inserted or deleted in between.	Elements can be inserted and deleted.
There is no built in members to do ascending or descending.	ArrayList has many methods to do operation like Sort, Insert, Remove, Binary Search, etc..,

Question 40: What you understand by Value types and Reference types in C#.Net?

Answer: In C# data types can be of two types: Value Types and Reference Types. Value type variables contain their object (or data) directly. If we copy one value type variable to another then we are actually making a copy of the object for the second variable. Both of them will independently operate on their values, Value Type member will locate into Stack and reference member will locate in Heap always.

Let consider each case briefly:

1. **Pure Value Type -** Here I used a structure as a value type. It has an integer member. I created two instances of this structure. After wards I assigned second instance to the first one. Then I changed the state of second instance, but it hasn't effect the first one, as whole items are value type and assignments on those types will copy only values not references i.e. in a Value Type assignment, all instances have its own local copy of members.

2. **Pure Reference Type -** I created a class and added a "DataTable" as a Reference Type member for this class. Then I performed the assignments just like below. But the difference is that on changing the state of second instance, the state of first instance will automatically alter. So in a Reference Type assignment both Value and Reference will be assigned i.e. all instances will point to the single object.

3. **Value Type with Reference Type -** This case and the last case to come are more interesting. I used a structure in this particular scenario also. But this time it includes a Reference Type (A Custom Class Object) Member besides a Value Type (An Integer) Member. When you performing the assignments, it seems like a swallow copy, as Value Type member of first instance won't effected, but the Reference Type member will alter according to the second instance. So in this particular scenario, assignment of Reference Type member produced a reference to a single object and assignment of Value Type member produced a local copy of that member.

4. **Reference Type With Value Type -** Contrary to the above case, in this scenario, both Reference & Value Types will be affected. I.e. a Value Type member in a Reference Type will be shared among its instances.

Question 41: What is Serialization?

Answer: Serialization means saving the state of your object to secondary memory, such as a file. Suppose you have a business layer where you have many classes to perform your business data. Now suppose you want to test whether your business classes give the correct data out without verifying the result from the UI or from a database. Because it will take some time to process.

Here comes Serialization. You will serialize all your necessary business classes and save them into a text or XML file, on your hard disk. So you can easily test your desired result by comparing your serialized saved data with, your desired output data. You can say it is a little bit of autonomic unit testing performed by the developer.

There are three types of serialization:

1. Binary serialization (Save your object data into binary format).
2. Soap Serialization (Save your object data into binary format; mainly used in network related communication).
3. XmlSerialization (Save your object data into an XML file).

Question 42: What is the use of using statement in C#?

Answer: The .Net Framework provides resource management for managed objects through the garbage collector - You do not have to explicitly allocate and release memory for managed objects. Clean-up operations for any unmanaged resources should perform in the destructor in C#. To allow the programmer to explicitly perform these clean-up activities, objects can provide a Dispose method that can be invoked when the object is no longer needed. The using statement in C# defines a boundary for the object outside of which, the object is automatically destroyed. The using statement is excited when the end of the "using" statement block or the execution exits the "using" statement block indirectly, for example - an exception is thrown. The "using" statement allows you to specify multiple resources in a single statement. The object could also be created outside the "using" statement. The objects specified within the using block must implement the IDisposable interface. The framework invokes the Dispose method of objects specified within the "using" statement when the block is exited.

Question 43: What is jagged array in C#.Net?

Answer: A jagged array is an array whose elements are arrays. The elements of a jagged array can be of different dimensions and sizes. A jagged array is sometimes called an "array of arrays." A special type of array is introduced in C#. A Jagged Array is an array of an array in which the length of each array index can differ.

Example:

int[][] jagArray = new int[5][];

In the above declaration the rows are fixed in size. But columns are not specified as they can vary.

Declaring and initializing jagged array:

1. int[][] jaggedArray = new int[5][];
2. jaggedArray[0] = new int[3];
3. jaggedArray[1] = new int[5];
4. jaggedArray[2] = new int[2];
5. jaggedArray[3] = new int[8];
6. jaggedArray[4] = new int[10];
7. jaggedArray[0] = new int[] { 3, 5, 7, };
8. jaggedArray[1] = new int[] { 1, 0, 2, 4, 6 };
9. jaggedArray[2] = new int[] { 1, 6 };
10. jaggedArray[3] = new int[] { 1, 0, 2, 4, 6, 45, 67, 78 };
11. jaggedArray[4] = new int[] { 1, 0, 2, 4, 6, 34, 54, 67, 87, 78 };

Question 44: What is Multithreading with .NET?

Answer: The real usage of a thread is not about a single sequential thread, but rather using multiple threads in a single program. Multiple threads running at the same time and performing various tasks is referred as Multithreading. A thread is considered to be a lightweight process because it runs within the context of a program and takes advantage of resources allocated for that program.
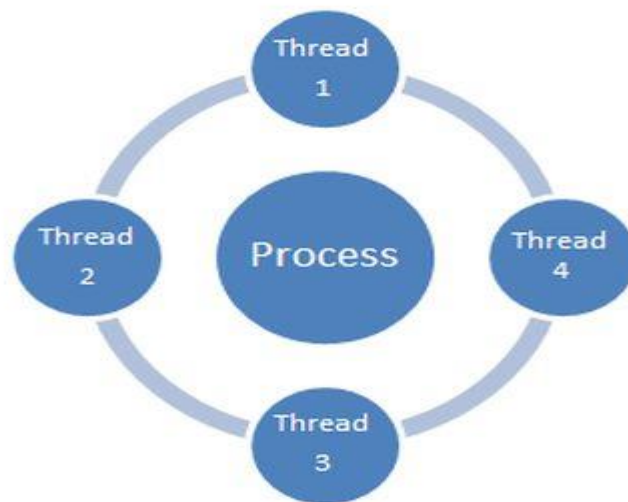
A single-threaded process contains only one thread while a multithreaded process contains more than one thread for execution.
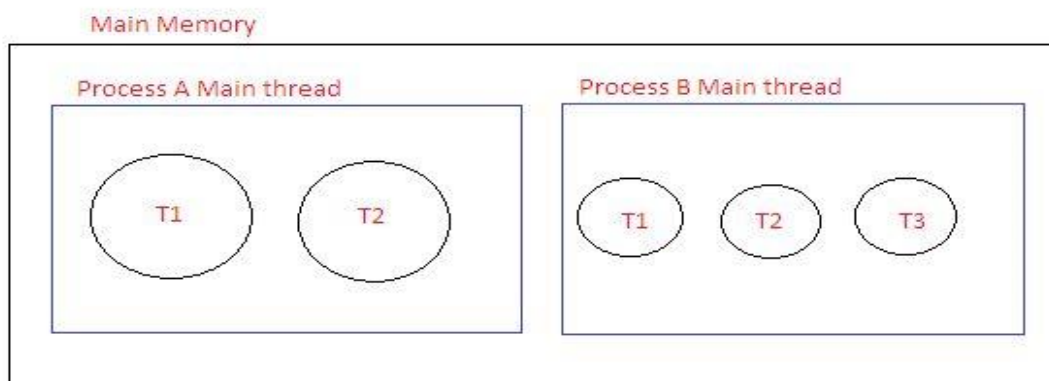
System.Threading Namespace-

Like many other features, in .NET, System.Threading is the namespace that provides various types to help in construction of multithreaded applications.

Type	Description
Thread	It represents a thread that executes within the CLR. Using this, we can produce additional threads in an application domain.
Mutex	It is used for synchronization between application domains.
Monitor	It implements synchronization of objects using Locks and Wait.
Smaphore	It allows limiting the number of threads that can access a resource concurrently.
Interlock	It provides atomic operations for variables that are shared by multiple threads.
ThreadPool	It allows you to interact with the CLR maintained thread pool.
ThreadPriority	This represents the priority level such as High, Normal, and Low.

Question 45: Explain Anonymous type in C#?

Answer: Anonymous types allow us to create new type without defining them. This is way to defining read only properties into a single object without having to define type explicitly. Here Type is generating by the compiler and it is accessible only for the current block of code. The type of properties is also inferred by the compiler.

We can create anonymous types by using "new" keyword together with the object initializer.

Example:

```
1.  var anonymousData = new
2.  {
3.      ForeName = "Jignesh",
4.      SurName = "Trivedi"
```

```
5.  };
6.  Console.WriteLine("First Name : " + anonymousData.ForeName);
```

Anonymous Types with LINQ Example:

Anonymous types are also used with the "Select" clause of LINQ query expression to return subset of properties.

Example:

If Any object collection having properties called FirstName , LastName, DOB etc. and you want only FirstName and LastName after the Querying the data then.

```
1.  class MyData {
2.     public string FirstName {
3.        get;
4.        set;
5.     }
6.     public string LastName {
7.        get;
8.        set;
9.     }
10.    public DateTime DOB {
11.       get;
12.       set;
13.    }
14.    public string MiddleName {
15.       get;
16.       set;
17.    }
18. }
19. static void Main(string[] args) {
20.    // Create Dummy Data to fill Collection.
21.    List < MyData > data = new List < MyData > ();
22.    data.Add(new MyData {
23.       FirstName = "Jignesh", LastName = "Trivedi", MiddleName = "G", DOB = new Dat
       eTime(1990, 12, 30)
24.    });
25.    data.Add(new MyData {
```

```
26.        FirstName = "Tejas", LastName = "Trivedi", MiddleName = "G", DOB = new DateT
       ime(1995, 11, 6)
27.    });
28.    data.Add(new MyData {
29.        FirstName = "Rakesh", LastName = "Trivedi", MiddleName = "G", DOB = new Date
       Time(1993, 10, 8)
30.    });
31.    data.Add(new MyData {
32.        FirstName = "Amit", LastName = "Vyas", MiddleName = "P", DOB = newDateTime
       (1983, 6, 15)
33.    });
34.    data.Add(new MyData {
35.        FirstName = "Yash", LastName = "Pandiya", MiddleName = "K", DOB = newDateTi
       me(1988, 7, 20)
36.    });
37. }
38. var anonymousData = from pl in data
39. select new {
40.    pl.FirstName, pl.LastName
41. };
42. foreach(var m in anonymousData) {
43.    Console.WriteLine("Name : " + m.FirstName + " " + m.LastName);  }
44. }
```

Question 46: Explain Hashtable in C#?

Answer: A Hashtable is a collection that stores (Keys, Values) pairs. Here, the Keys are used to find the storage location and are immutable and cannot have duplicate entries in the Hashtable. The .Net Framework has provided a Hash Table class that contains all the functionality required to implement a hash table without any additional development. The hash table is a general-purpose dictionary collection. Each item within the collection is a DictionaryEntry object with two properties: a key object and a value object. These are known as Key/Value. When items are added to a hash table, a hash code is generated automatically. This code is hidden from the developer. All access to the table's values is achieved using the key object for identification. As

the items in the collection are sorted according to the hidden hash code, the items should be considered to be randomly ordered.

The Hashtable Collection: The Base Class libraries offers a Hashtable Class that is defined in the System.Collections namespace, so you don't have to code your own hash tables. It processes each key of the hash that you add every time and then uses the hash code to look up the element very quickly. The capacity of a hash table is the number of elements the hash table can hold. As elements are added to a hash table, the capacity is automatically increased as required through reallocation. It is an older .Net Framework type.

Declaring a Hashtable: The Hashtable class is generally found in the namespace called System.Collections. So to execute any of the examples, we have to add using System.Collections; to the source code. The declaration for the Hashtable is:

Hashtable HT = new Hashtable ();

Question 47: What is LINQ in C#?

Answer: LINQ stands for Language Integrated Query. LINQ is a data querying methodology which provides querying capabilities to .NET languages with syntax similar to a SQL query

LINQ has a great power of querying on any source of data. The data source could be collections of objects, database or XML files. We can easily retrieve data from any object that implements the IEnumerable<T> interface.

Advantages of LINQ:

1. LINQ offers an object-based, language-integrated way to query over data no matter where that data came from. So through LINQ we can query database, XML as well as collections.

2. Compile time syntax checking.

3. It allows you to query collections like arrays, enumerable classes etc. in the native language of your application, and like VB or C# in much the same way as you would query a database using SQL.
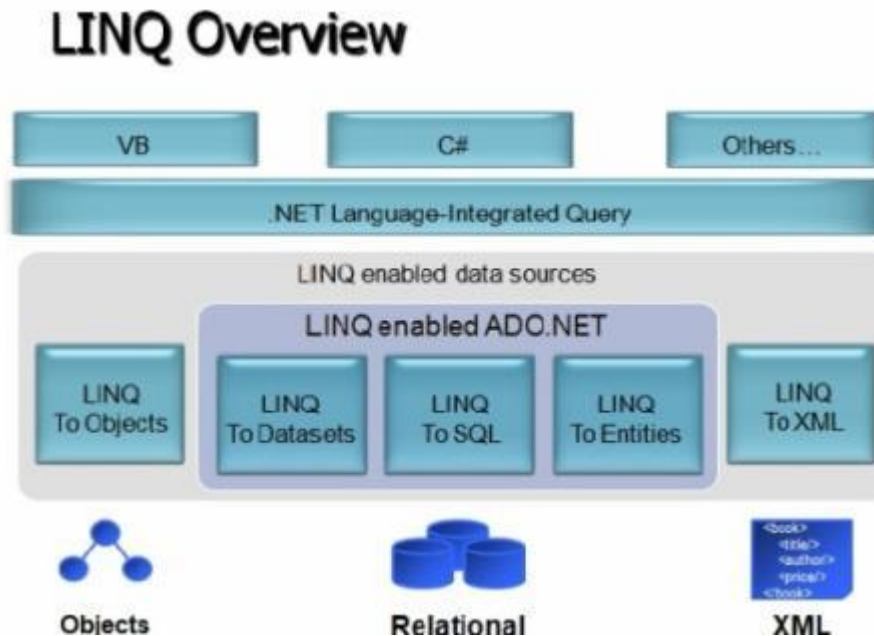
Question 48: What is File Handling in C#.Net?

Answer: The System.IO namespace provides four classes that allow you to manipulate individual files, as well as interact with a machine directory structure. The Directory and File directly extends System.Object and supports the creation, copying, moving and deletion of files using various static methods. They only contain static methods and are never instantiated. The FileInfo and DirecotryInfo types are derived from the abstract class FileSystemInfo type and they are typically, employed for obtaining the full details of a file or directory because their members tend to return strongly typed objects. They implement roughly the same public methods as a Directory and a File but they are stateful and the members of these classes are not static.
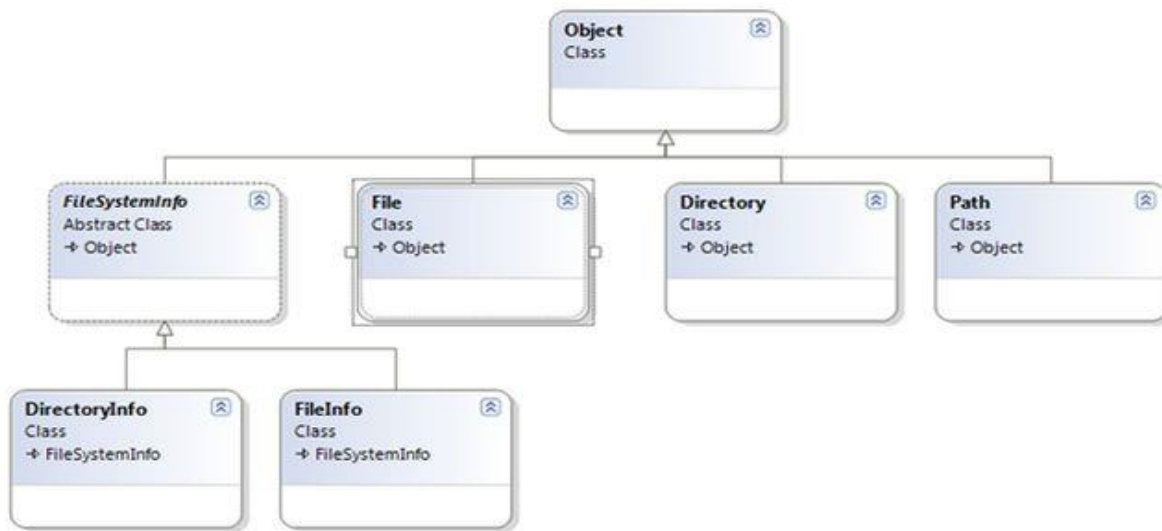
Question 49: What is Reflection in C#.Net?

Answer: Reflection typically is the process of runtime type discovery to inspect metadata, CIL code, late binding and self-generating code. At run time by using reflection, we can access the same "type" information as displayed by the ildasm utility at design time. The reflection is analogous to reverse engineering in which we can break an existing *.exe or *.dll assembly to explore defined significant contents information, including methods, fields, events and properties.

You can dynamically discover the set of interfaces supported by a given type using the System.Reflection namespace. This namespace contains numerous related types as follows:

Types	Description
Assembly	This static class allows you to load, investigate and manipulate an assembly.
AssemblyName	Allows to exploration of abundant details behind an assembly.
EventInfo	Information about a given event.
PropertyInfo	Holds information of a specified property.
MethodInfo	Contains information about a specified method.

Reflection typically is used to dump out the loaded assemblies list, their reference to inspect methods, properties etcetera. Reflection is also used in the external disassembling tools such

Reflector, Fxcop and NUnit because .NET tools don't need to parse the source code similar to C++.

Metadata Investigation:

The following program depicts the process of reflection by creating a console based application. This program will display the details of the fields, methods, properties and interfaces for any type within the mscorlib.dll assembly. Before proceeeding, it is mandatory to import "System.Reflection".

Here, we are defining a number of static methods in the program class to enumerate fields, methods and interfaces in the specified type. The static method takes a single "System.Type" parameter and returns void.

```
1.  static void FieldInvestigation(Type t) {
2.      Console.WriteLine("*********Fields*********");
3.      FieldInfo[] fld = t.GetFields();
4.      foreach(FieldInfo f in fld) {
5.          Console.WriteLine("-->{0}", f.Name);
6.      }
7.  }
8.  static void MethodInvestigation(Type t) {
9.      Console.WriteLine("*********Methods*********");
10.     MethodInfo[] mth = t.GetMethods();
11.     foreach(MethodInfo m in mth) {
12.         Console.WriteLine("-->{0}", m.Name);
13.     }
14. }
```

Question 50: What is Expression Trees In C#?

Answer: Expression and Expression<> are basically classes that can represent the CSharp code as Data. Unlike Func<> or Action<> Expressions are non-compiled Data about the code. Most of LINQ Providers has been built using Expressions.

Walkthrough of a sample expression: Expression can be parsed, analyzed in the program. Let's create a simple Expression:

```
// A simple delegated operation which perform string join.
Func<string, string, string> StringJoin = (str1, str2) => string.Concat(str1, str2);
```

Now I want to parse it analyze it or may be doing some more but for that I need to treat this code as Data. Now let's write an expression for the above lambda expression:

```
Expression<Func<string, string, string>> StringJoinExpr = (str1, str2) => string.Concat(str1, str2);
```

The Expression Tree can be visualized as with its major properties:
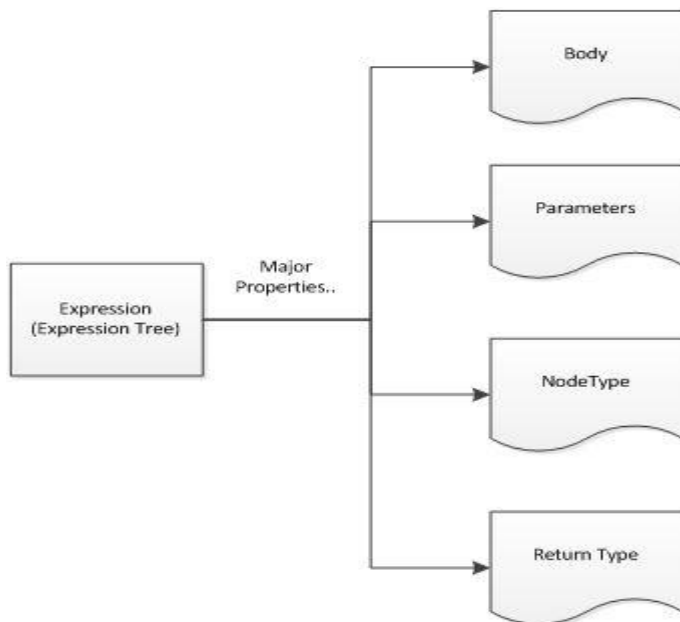
Let's analyze our Expression according to above figure, and see what values are populated to understand the break of Expression statement:
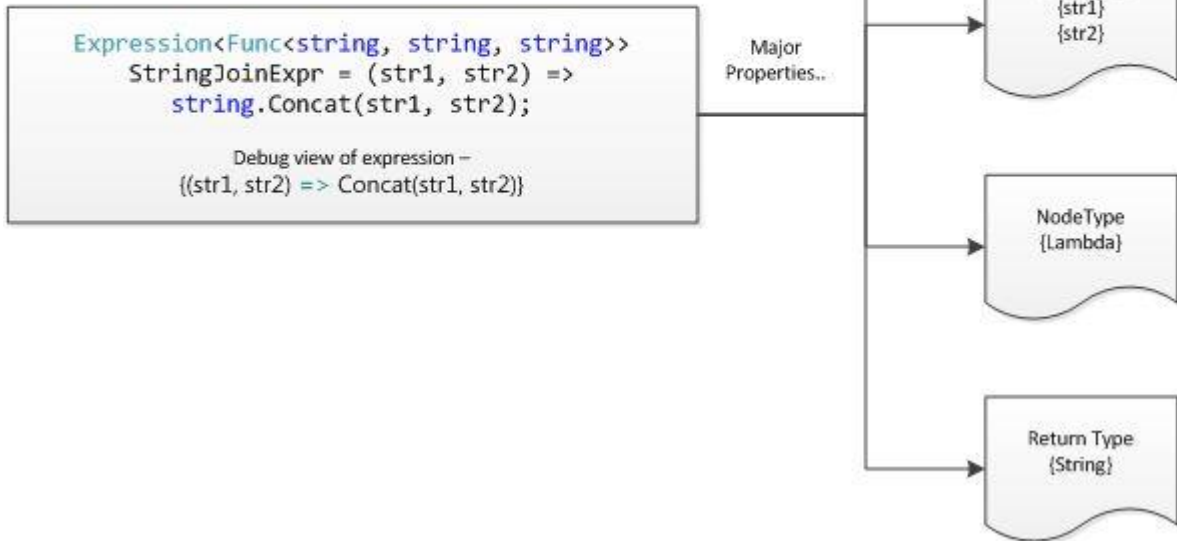
Chapter 3

Most Asked ADO.NET Interview Questions and Answers

Question 1: What is ADO.NET?

Answer: ADO stands for Active Data Object and ADO.NET is a set of .NET libraries for ADO. ADO.NET is a collection of managed libraries used by .NET applications for data source communication using a driver or provider:

- Enterprise applications handle a large amount of data. This data is primarily stored in relational databases, such as Oracle, SQL Server, and Access and so on. These databases use Structured Query Language (SQL) for retrieval of data.

- To access enterprise data from a .NET application, an interface was needed. This interface acts as a bridge between an RDBMS system and a .NET application. ADO.NET is such an interface that is created to connect .NET applications to RDBMS systems.

- In the .NET framework, Microsoft introduced a new version of Active X Data Objects (ADO) called ADO.NET. Any .NET application, either Windows based or web based, can interact with the database using a rich set of classes of the ADO.NET library. Data can be accessed from any database using connected or disconnected architecture.

ADO.NET provides mainly the following two types of architectures:

- Connected Architecture.
- Disconnected Architecture.

ADO.NET Namespaces

| Namespaces	Description
System.Data	Contains the definition for columns, relations, tables, database, rows, views and constraints.
System.Data.SqlClient	Contains the classes that are used to connect to a Microsoft SQL Server database such as SqlCommand, SqlConnection, SqlDataAdapter.
System.Data.Odbc	Contains classes required to connect to most ODBC drivers. These classes include OdbcCommand,OdbcConnection.
System.Data.OracleClient	Contains classes such as OracleConnection,OracleCommand required to connect to an Oracle database.

Question 2: What are the ADO.NET components?

Answer: ADO.NET components categorized in three modes: disconnected, common or shared and the .NET data providers.

The disconnected components build the basic ADO.NET architecture. You can use these components (or classes) with or without data providers. For example, you can use a DataTable object with or without providers and shared or common components are the base classes for data providers. Shared or common components are the base classes for data providers and shared by all data providers. The data provider components are specifically designed to work with different kinds of data sources. For example, ODBC data providers work with ODBC data sources and OleDb data providers work with OLE-DB data sources.

Figure represents the ADO.NET components model and how they work together:
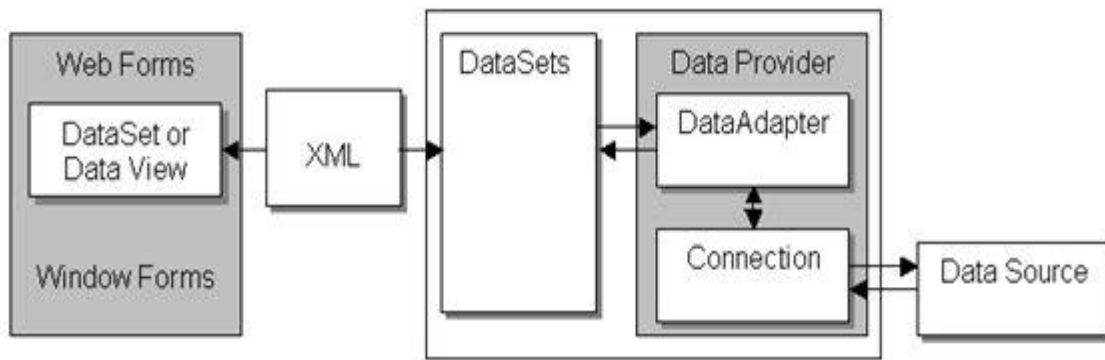
Question 3: How can you define the DataSet structure?

Answer: A DataSet object falls in disconnected components series. The DataSet consists of a collection of tables, rows, columns and relationships.

DataSet contains a collection of DataTables and the DataTable contains a collection of DataRows, DataRelations, and DataColumns. A DataTable maps to a table in the database. The previous DataSet contains a DataTable that maps to the Orders table because you filled it with a SELECT query executed on the Order table.
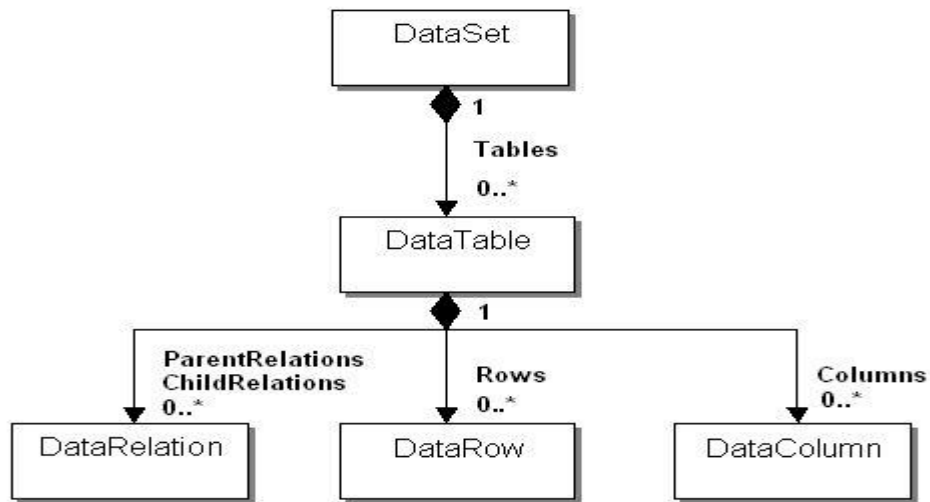
Question 4: What is Connection Pooling in ADO.NET?

Answer: Connection pooling is the ability of reusing your connection to the database. This means if you enable Connection pooling in the connection object, actually you enable the re-use of the connection to more than one user.

ADO.NET uses a technique called connection pooling, which minimizes the cost of repeatedly opening and closing connections. Connection pooling reuses existing active connections with the same connection string instead of creating new connections when a request is made to the database. It involves the use of a connection manager that is responsible for maintaining a list, or pool, of available connections for a given connection string. Several pools exist if different connection strings ask for connection pooling.

Example of Pooling:

1. connection.ConnectionString = sqlConnectString + "Connection Timeout=30;Connection Lifetime=0;Min Pool Size=0;Max Pool Size=100;Pooling=true;";
2. //Open connection

A Connection String in the Web.Config file with connection pooling option:

1. <connectionStrings>
2.
3.   <add name="sqlConnectionString" connectionString="Data Source=mySQLServer;Initial Catalog=myDatabase;Integrated Security=True;Connection Timeout=15;Connection Lifetime=0;Min Pool Size=0;Max Pool Size=100;Pooling=true;" />
4. </connectionStrings>

SQL Server connection string pooling attributes:

- **Connection Lifetime:** Length of time in seconds after creation after which a connection is destroyed. The default is 0, indicating that connection will have the maximum timeout.

- **Connection Reset:** Specifies whether the connection is reset when removed from the pool. The default is true.

- **Enlist:** Specifies whether the connection is automatically enlisted in the current transaction context of the creation thread if that transaction context exists. The default is true.

- **Load Balance Timeout:** Length of time in seconds that a connection can remain idle in a connection pool before being removed.

- **Max Pool Size:** Maximum number of connections allowed in the pool. The default is 100.

- **Min Pool Size:** Minimum number of connections maintained in the pool. The default is 0.

- **Pooling:** When true, the connection is drawn from the appropriate pool, or if necessary, created and added to the appropriate pool. The default is true.

Question 5: What are the differences Between DataReader and DataSet?

Answer:

No	Data Reader	DataSet
1	Used in a connected architecture	Used in a disconnected architecture.
2	Provides better performance	Provides lower performance.
3	DataReader object has read-only access	A DataSet object has read/write access
4	DataReader object supports a single table based on a single SQL query of one database	A DataSet object supports multiple tables from various databases.
5	A DataReader object is bound to a single control.	A DataSet object is bound to multiple controls.
6	A DataReader object has faster access to data.	A DataSet object has slower access to data.
7	A DataReader object must be manually coded.	A DataSet object is supported by Visual Studio tools.
8	We can't create a relation in a data reader.	We can create relations in a dataset.
9	Whereas a DataReader doesn't support data reader communicates with the command object.	A Dataset supports integration with XML Dataset communicates with the Data Adapter only.
10	DataReader cannot modify data.	A DataSet can modify data.

Question 6: What is SqlCommand Object?

Answer: The SqlCommand carries the SQL statement that needs to be executed on the database. SqlCommand carries the command in the CommandText property and this property will be used when the SqlCommand calls any of it's execute methods.

- The Command Object uses the connection object to execute SQL queries.
- The queries can be in the form of Inline text, Stored Procedures or direct Table access.
- An important feature of Command object is that it can be used to execute queries and Stored Procedures with Parameters.
- If a select query is issued, the result set it returns is usually stored in either a DataSet or a DataReader object.

The three important methods exposed by the SqlCommand object is shown below:

- ExecuteScalar
- ExecuteNonQuery
- ExecuteReader

ExecuteScalar – This is useful for returning a single value from the database. For example, using this method we can retrieve a sum of sales made by a specific product, total number of records in the employee table, unique id by supplying filtering conditions and so on. Since this method performs faster we do not need to go for the Reader method just to retrieve a single scalar value.

ExecuteNonQuery - This is useful for performing data manipulation on the database. Simply, the ExecuteNonQuery is for executing the DML statements. The return value of the ExecuteNonQuery is an integral value that represents the number of rows affected by the Operation.

ExecuteReader – This is used when we need to retrieve rows and columns of data using the SQL select statements. As the data retrieved is a table of data, ExecuteReader returns SqlDataReader. We should iterate through this object to get the required values.

Question 7: What is the difference between ADO and ADO.NET?

Answer: Difference between ADO and ADO.NET is in following table:

ADO	ADO.NET
ADO has one main object that is used to reference data, called the RecordSet object.	ADO.NET provides objects that allow you to access data in various ways. The DataSetobject allows you to store the relational model of your database. MARS (Multiple Active Result Sets) is implemented in ADO.NET
You can only work on connected manner. This means that when you access data, such as viewing and updating data, it is real-time, with a connection being used all the time. This is barring, of course, you programming special routines to pull all your data into temporary tables. In connected model you always get refreshed data.	ADO.NET uses data in a disconnected fashion. When you access data, ADO.NET makes a copy of the data using XML. ADO.NET only holds the connection open long enough to either pull down the data or to make any requested updates. This makes ADO.NET efficient to use for Web applications. It's also decent for desktop applications. You can work on connected and disconnected manner.br> In disconnected model you will get old data as you are editing it. Outlook is an example of disconnected model. We work on offline object model and when connection is required it is connected. Connected object can be used on disconnected object.
Whereas ADO allows you to persist records in XML format.	ADO.NET allows you to manipulate your data using XML as the primary means. This is nice when you are working with other business applications and also helps when you are working with firewalls because data is passed as HTML and XML.
ADO allows you to create client-side cursors only.	ADO.NET gives you the choice of either using client-side or server-side cursors. In ADO.NET, classes actually handle the work of cursors. The developer has the freedom of choice in internet development, for creating efficient applications.

Question 9: What is the use of DataSet object in ADO.NET?

Answer: A DataSet is a collection of DataTable and DataRelations. Each DataTable is a collection of DataColumn, DataRows and Constraints.

- It is used in a disconnected architecture.
- Provides lower performance. A DataSet object has read/write access.
- A DataSet object supports multiple tables from various databases.
- A DataSet object is bound to multiple controls.
- A DataSet object has slower access to data.
- A DataSet object is supported by Visual Studio tools.
- We can create relations in a dataset.
- A Dataset supports integration with XML.
- A DataSet communicates with the Data Adapter only.
- A DataSet can modify data.

Example:

1. DataTable dt = new DataTable();
2. DataColumn col =new DataColumn();
3. Dt.columns.Add(col2);
4. DataRow row = dt.newRow();

Question 10: Describe the System.Data Namespace Class?

Answer: The three general namespaces are System.Data, System.Data.Common and System.Data.SqlTypes. Some of the provider-specific namespaces are System.Data.OleDb, Microsoft.Data.Odbc and System.Data.SqlClient.

The System.Data namespace defines classes that you can use with all the data providers or without data providers at all. This namespace also defines interfaces that are base classes for the data provider classes. The following figure shows the System.Data namespace class hierarchy.

```
Object
    Marshal ByValueComponent
        DataColumn
        DataSet
        DataTable
        DataView
        DataViewManager
    Constraint
        ForeignKeyConstraint
        UniqueConstraint
    DataRelation
    DataRow
    DataRowView
    DataViewSetting
    DataViewSettingCollection
    InternalDataCollectionBase
        ConstraintCollection
        DataColumnCollection
        DataRelationCollection
        DataRowCollection
        DataTableCollection
    TypedDataSetGenerator
    Exception
        SystemException
            DataException
            DBConcurrencyException
```

```
Object
    Delegate
        MulticastDelegate
            DataColumnChangeEventHandler
            DataRowEventChangeHandler
            FillErrorEventHandler
            MergeFailedEventHandler
            StateChangeEventHandler
    EventArgs
        DataColumnChangeEventArgs
        DataRowEventChangeArgs
        FillErrorEventArgs
        MergeFailedEventArgs
        StateChangeEventArgs
```

Question 11: What is DataTable in ADO.NET?

Answer:

- DataTable represents a single table in a database.

- In this show row and column.

- DataSet is a collection of data tables.

- In this store data record.

DataTable representation in .aspx.cs code,

Example:

```
1.   protected void BinddataTable()
2.   {
3.       SqlConnection con = new SqlConnection("your database connection string");
4.       con.Open();
5.       SqlCommand cmd = new SqlCommand("Write your query or procedure", con);
6.       SqlDataAdapter da = new SqlDataAdapter(cmd);
7.       DataTable dt = new DataTable();
8.       da.Fill(dt);
9.       grid.DataSource = dt;
10.      grid.DataBind();
11.  }
```

Question 12: What is the DataReader in ADO.Net?

Answer:

- DataReader holds only one table at a time.
- It only provides read only access mode and cannot write data.
- It is not required local storage to data store.
- Holds one row at a time.
- Uses less memory.
- DataReader do not maintain relation.

DataReader representation in .aspx.cs code,

```
1.  protected void Bind()
2.  {
3.      SqlConnection con = new SqlConnection("your database connection string ");
4.      con.Open();
5.      SqlCommand cmd = new SqlCommand("Write your query or procedure ", con);
6.      SqlDataReader dr = cmd.ExecuteReader();
7.      grid.DataSource = dr;
8.      grid.DataBind();
9.  }
```

The DataReader properties:

Property	Description
Depth	Indicates the depth of nesting for row
FieldCount	Returns number of columns in a row
IsClosed	Indicates whether a data reader is closed
Item	Gets the value of a column in native format
RecordsAffected	Number of row affected after a transaction

The DataReader methods:

Property	Description
Close	Closes a DataRaeder object.
Read	Reads next record in the data reader.
NextResult	Advances the data reader to the next result during batch transactions.
Getxxx	There are dozens of Getxxx methods. These methods read a specific data type value from a column. For example. GetChar will return a column value as a character and GetString as a string.

Question 13: What is the SqlCommandBuilder?

Answer: CommandBuilder helps you to generate update, delete, and insert commands on a single database table for a data adapter. Similar to other objects, each data provider has a command builder class. The OleDbCommandBuilder, SqlCommonBuilder, and OdbcCommandBuilder classes represent the CommonBuilder object in the OleDb, Sql, and ODBC data providers.

Creating a SQLCommandBuilder Object:

Creating a CommonedBuider object is pretty simply. You pass a DataAdapter as an argument of the CommandBuilder constructor. For example,

1. // Create a command builder object
2. SqlCommandBuilder builder = new SqlCommandBuilder(adapter);

Question 14: What is the Connection object in ADO.NET?

Answer: A Connection object sits between a data source and a DataAdapter (via Command). You need to define a data provider and a data source when you create a connection. With these two, you can also specify the user ID and password depending on the type of data source. Figure shows the relationship between a connection, a data source, and a data adapter.
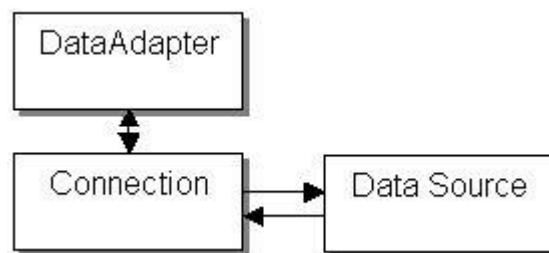
Connection can also be connected to a Command object to execute SQL queries, which can be used to retrieve, add, update and delete data to a data source. Figure 2 shows the relationship between the Command and Connection objects.
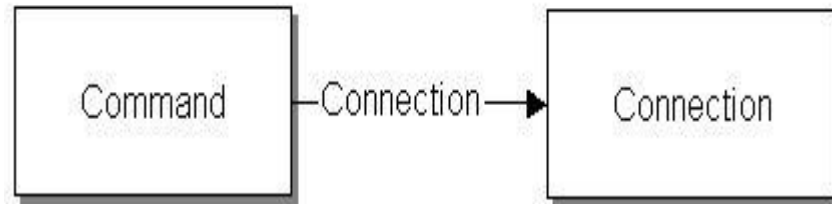
Data provider connection classes

Data provider	Connection classes
OleDb	OleDbConnection
Sql	SqlConnection
ODBC	OdbcConnection

Question 15: Describe the DataView in ADO.NET?

Answer: A DataView enables you to create different views of the data stored in a DataTable, a capability that is often used in data binding applications. Using a DataView, you can expose the data in a table with different sort orders, and you can filter the data by row state or based on a filter expression. A DataView provides a dynamic view of data whose content, ordering, and membership reflect changes to the underlying DataTable as they occur. This is different from the Select method of the DataTable, which returns a DataRow array from a table per particular filter and/or sort order and whose content reflects changes to the underlying table, but whose membership and ordering remain static. The dynamic capabilities of the DataView make it ideal for data-binding applications.

How we can create a DataView?

There are two ways to create a DataView. You can use the DataView constructor, or you can create a reference to the DefaultView property of the DataTable. The DataView constructor can

be empty, or will also take either a DataTable as a single argument, or a DataTable along with filter criteria, sort criteria, and a row state filter.

```
1.  DataView custDV = new DataView(customerDS.Tables["Customers"],
2.  "Country = 'USA'",
3.  "ContactName",
4.  DataViewRowState.CurrentRows);
5.  DataView custDV = customerDS.Tables["Customers"].DefaultView;
```

Question 16: What is ExecuteScalar method in ADO.NET?

Answer: The ExecuteScalar method of the SqlCommand object is useful for retrieving a single value from the database. In our example, we need to retrieve the total number of records in the Titles table of the Pubs database. Since the total number of records is a single scalar value, the Execute Scalar method is used. The following is the code and its explanation:

```
1.  private void frmSqlCommand_Load(object sender, EventArgs e)
2.  {
3.      //Sample 03: Open Database Connection
4.      String con_string = Properties.Settings.Default.ConStrPubs;
5.      pubs_db_connection = new SqlConnection(con_string);
6.      pubs_db_connection.Open();
7.      //Sample 04: Form the Command Object
8.      SqlCommand cmd = new SqlCommand();
9.      cmd.CommandText = "Select Count(*) as Count from Titles";
10.     cmd.Connection = pubs_db_connection;
11.     //Sample 05: Execute the Command & retrive scalar value
12.     lblTotal.Text = System.Convert.ToString(cmd.ExecuteScalar());
13. }
```

Question 17: What are the methods of DataSet?

Answer: It is used in disconnected architecture. It represents records in the form of Database table (Row and Column) format. It stores record of one or more tables.

Example:

```
1.  SqlDataAdapter da;
2.  DataSet ds;
3.  string strconn = "Data Source=YourServerName;Initial Catalog=EMP;Integrated Security
    =True";
4.  private void Form1_Load(object sender, EventArgs e)
5.  {
6.      da = new SqlDataAdapter("select * from userdet", strconn);
7.      ds = new System.Data.DataSet();
8.      da.Fill(ds);
9.      dataGridView1.DataSource = ds.Tables[0];
10. }
```

Methods of DataSet:

- **AcceptChanges():** This method saves changes which are made with records in a DataSet.

- **Clear():** This method clears (removes) all rows from DataSet.

- **Clone():** The clone method copy the structure of DataSet. Means it copy only schema not full records of DataSet.

- **Copy():** It copies the whole records with structure of DataSet.

- **RejectChanges():** This method discard changes which is made with DataSet and set the DataSet to previous stage (which was at first).

- **HasChanges():** This method return boolean value to show whether record of DataSet has changed or not. It returns true if any changes has made and false if no other changes made.

- **GetChanges():** This method keep copy of those record, which is changed or modified.

Question 18: What are Parameters in ADO.NET?

Answer: Parameters in conjunction with SelectCommand to help you Select data for the DataSet.

The OleDbType describes the type information for the parameter. It consists of everything from strings to Global Unique Identifiers (GUIDs). SQL data provider has SqlDbType, and ODBC data provider has an ODBC type. These type names and definitions differ depending upon the provider you're using. For example, the Money type is the same in ODBC and Sqldata providers, but is called Currency in OleDb data providers.

The OLEDB Parameter Class properties:

Property	Description
DbType	Represents the DbType of the parameter.
Direction	Represents the direction of a parameter. A parameter can be input-only, output-only, bi-directional, or a stored procedure.
IsNullable	Represents whether a parameter accepts null values.
OleDbType	Represents the OleDbType of the parameter.
ParameterName	Represents the name of the parameter.
Precision	Represents the maximum number of digits used to represent the Value property.
Scale	Represents the decimal places to which Value is resolved.
Size	Represents the maximum size in bytes a column can store.
SourceColumn	Represents the source column mapped to the DataSet.
SourceVersion	Represents the DataRowversion.
Value	Represents the Value of the parameter.

Creating a parameter:

this.oleDbDeleteCommand2.Parameters.Add(newSystem.Data.OleDb.OleDbParameter("Contact Name", System.Data.OleDb.OleDbType.char, 30, System.Data.ParameterDirection.Input, false, ((system.Byte)(0)),((System.Byte)(0)), "Contact Name", System.Data.DataRowVersion.Original, null));

Question 19: What is the difference between DataSet and DataReader?

Answer: DataReader-

1. The ADO.NET DataReader is used to retrieve read-only (cannot update data back to datasource) and forward-only (cannot read backward/random) data from a database.

2. Using the DataReader increases application performance and reduces system overheads. This is due to one row at a time is stored in memory.

3. You create a DataReader by calling Command.ExecuteReader after creating an instance of the Command object.

4. This is a connected architecture: The data is available as long as the connection with database exists.

5. You need to open and close the connection manually in code.

DataSet-

1. The DataSet is a in-memory representation of data.

2. It can be used with multiple data sources. That is a single DataSet and can hold the data from different data sources holding data from different databases/tables.

3. The DataSet represents a complete set of data including related tables, constraints, and relationships among the tables.

4. The DataSet can also persist and reload its contents as XML and its schema as XML Schema definition language (XSD) schema.

5. The DataAdapter acts as a bridge between a DataSet and a data source for retrieving and saving data.

6. The DataAdapter helps mapping the data in the DataSet to match the data in the data source.

7. Also, upon an update of dataset, it allows changing the data in the data source to match the data in the DataSet.

8. No need to manually open and close connection in code.

9. Hence, point (8) says that it is a disconnected architecture. Fill the data in DataSet and that's it. No connection existence required.

Question 20: What you understand by ExecuteNonQuery Method?

Answer: The ExecuteNonQuery method is used to execute the command and return the number of rows affected.The ExecuteNonQuery method cannot be used to return the result set.

Snippets working with ExecuteNonQuery

```
1.  public void CallExecuteNonQuery()
2.  {
3.      SqlConnection conn = new SqlConnection();
4.      conn.ConnectionString = ConfigurationManager.ConnectionStrings["connString"].ConnectionString;
5.      try
6.      {
7.          SqlCommand cmd = new SqlCommand();
8.          cmd.Connection = conn;
9.          cmd.CommandText = "DELETE FROM EMP WHERE DEPTNO = 40";
10.         cmd.CommandType = CommandType.Text;
11.         conn.Open();
12.         Int32 RowsAffected = cmd.ExecuteNonQuery();
13.         MessageBox.Show(RowsAffected + " rows affected", "Message");
14.         cmd.Dispose();
15.         conn.Dispose();
16.     }
17.     catch(Exception ex)
18.     {
19.         MessageBox.Show(ex.Message);
20.     }
21. }
```

Question 21: What do you understand by DataRelation class?

Answer: The DataRelation is a class of disconnected architecture in the .NET framework. It is found in the System.Data namespace. It represents a relationship between database tables and correlates tables on the basis of matching column.

Example:

1. DataRelation drel;
2. drel = new DataRelation("All", ds.Tables[0].Columns[0], ds.Tables[1].Columns[0]);

Question 22: How can create a SqlConnection?

Answer: SqlConnection class is used to establish the connection between front end and back end.

Syntax:
SqlConnection obj=new SqlConnection("Integrated Security=true;Initial Catalog=Table_Name;Data Source=.");-- for Windows authentication

SqlConnection obj=new SqlConnection("user id= sa ; Password=sa123;server=.;database=name"); --Sql server Authentication

Question 23: What are the important classes in ADO.NET?

Answer: ADO.NET is a set of classes (a framework) to interact with data sources such as databases and XML files. ADO is the acronym for ActiveX Data Object. It allows us to connect to underlying data or databases. It has classes and methods to retrieve and manipulate data.

The following are a few of the .NET applications that use ADO.NET to connect to a database, execute commands and retrieve data from the database.

- ASP.NET Web Applications
- Console Applications
- Windows Applications.

Important classes in ADO.NET:

1. Connection Class
2. Command Class
3. DataReader Class
4. DataAdaptor Class
5. DataSet.Class

How to Connect to a Database using ADO.NET:

To create a connection, you must be familiar with connection strings. A connection string is required as a parameter to SQLConnection. A ConnectionString is a string variable (not case sensitive).

This contains key and value pairs, like provider, server, database, userid and word as in the following:

Server="nameof the server or IP Address of the server"

Database="name of the database"

userid="user name who has permission to work with database"

word="the word of userid"

Question 24: What do you understand by SqlTransaction class in ADO.NET?

Answer: The SqlTransaction class is an important class of .NET Framework. It ensures that a body of code will affect a Database or kept the same as previous (Rollback).

At first we should know about it's two most important method which will be used here. They are given below.

- **Commit():** It commits the transaction. It saves changes made in Database during transaction. In simple term we can also say that it shows the end of transaction at that time.

- **Rollback():** It is used to rollback the transaction. It set the database in previous stage which was, before the begin of transaction.

Question 25: What are the two fundamental objects in ADO.NET?

Answer: There are the two fundamental objects in ADO.NET:

- Data reader
- Data set

DataReader Class:

The DataReader class object allows you to read the data returned by a SELECT command by a simple forward-only and read-only cursor. It requires a live connection with the data source and provides a very efficient way of looping and consuming all parts of the result set. The object of the DataReader cannot be directly instantiated. Instead you must call the ExecuteReader method of the Command object and close the connection when you are done using the DataReader otherwise the connection remains alive until it is explicitly closed.

DataReader with ExecuteReader() Method:

```
1.  //Open connection
2.  Conn.Open();
3.  sdr = sc.ExecuteReader(CommandBehavior.CloseConnection);
4.  //Get all records
5.  while(sdr.Read())
6.  {
7.     textBox1.AppendText(sdr.GetValue(0) + "\t" + sdr.GetValue(1));
8.     textBox1.AppendText("\n");
9.  }
```

DataSet class:

A DataSet is a disconnected architecture technology. It contains zero or more tables and relationships. When you work with a dataset, the data in the data source is not touched at all. Instead all the changes are made locally to the dataset in memory. In the following example you will see how to retrieve data from a SQL Server table and use it to fill in a DataTable object in the DataSet.

Example:

```
1.   private void Form1_Load(object sender, EventArgs e)
2.   {
3.      //Connection String
4.      string conString = "Data Source=localhost;Database=AdventureWorksLT2008;Integrat
        ed Security=SSPI";
5.      // Add Connection string to SqlConnection
6.      SqlConnection Conn = new SqlConnection(conString);
7.      string query = "select * from SalesLT.Customer";
8.      //Command Class definition
9.      SqlCommand sc = new SqlCommand(query, Conn);
10.     // Data Adapter definition
11.     SqlDataAdapter sda = new SqlDataAdapter();
12.     sda.SelectCommand = sc;
13.     //data Set definition
14.     DataSet ds = new DataSet();
15.     // filling the result set in data table
16.     sda.Fill(ds, "SalesLT.Customer");
17.     //output in data grid
18.     dataGridView1.DataSource = ds.Tables["SalesLT.Customer"];
19.  }
```

Question 26: What is DataAdapter and its property?

Answer: A DataAdapter bridges the gap between the disconnected DataTable objects and the physical data source. The SqlDataAdapter is capable of executing a SELECT, DELETE and UPDATE statement on a data source as well as extracting input from the result set into a DataTable object. The SqlDataAdapter class provides a method called Fill() to copy the result set into the DataTable.

Example:

1. // Data Adapter definition
2. SqlDataAdapter sda = new SqlDataAdapter(sc);

These are the commonly used properties offered by the SqlDataAdapter class as in the following:

Property	Description
SelectCommand	This command executed to fill in a Data Table with the result set.
InsertCommand	Executed to insert a new row to the SQL database.
UpdateCommand	Executed to update an existing record on the SQL database.
DeleteCommand	Executed to delete an existing record on the SQL database.

Question 27: Which namespaces are used for data access?

Answer: ADO.NET is a collection of managed libraries used by .NET applications for data source communication using a driver or provider.

ADO.NET provides libraries for the data source communication under the following namespaces.

1. system.Data
2. system.Data.OleDb
3. system.Data.SqlClient
4. system.Data.OracleClient
5. system.Data.Odbc

1. **System.Data:** This namespace is used for holding and managing data on a client machine.

2. **System.Data.OleDb:** This namespace can communicate with any data source like files, databases, indexing servers and so on using the "**OleDb**" Provider.

3. **System.Data.SqlClient:** This namespace can communicate with "**SQL Server**" database only using SqlClient Providers.

4. **System.Data.OracleClient:** This namespace can communicate with an "**Oracle**" database only using OracleClient Providers.

5. **System.Data.ODBC**: This namespace contains the same set of classes as the following:

   o Connection

   o Command

   o DataReader

   o DataAdaptar

   o CommandBuilder

   o Parameter

Question 28: Explain the properties and methods of Command Object.

Answer: The command object is one of the basic components of ADO .NET.

1. The Command Object uses the connection object to execute SQL queries.

2. The queries can be in the form of Inline text, Stored Procedures or direct Table access.

3. An important feature of Command object is that it can be used to execute queries and Stored Procedures with Parameters.

4. If a select query is issued, the result set it returns is usually stored in either a DataSet or a DataReader object.

Associated Properties of SqlCommand class:

Property	Type of Access	Description
Connection	Read/Write	The SqlConnection object that is used by the command object to execute SQL queries or Stored Procedure.
CommandText	Read/Write	Represents the T-SQL Statement or the name of the Stored Procedure.
CommandType	Read/Write	This property indicates how the CommandText property should be interpreted. The possible values are: 1. Text (T-SQL Statement) 2. StoredProcedure (Stored Procedure Name) 3. TableDirect
CommandTimeout	Read/Write	This property indicates the time to wait when executing a particular command. **Default Time for Execution of Command is 30 Seconds.** The Command is aborted after it times out and an exception is thrown.

Now, let us have a look at various execute methods that can be called from a Command Object.

Property	Description
ExecuteNonQuery	This method executes the command specifies and returns the number of rows affected.
ExecuteReader	The ExecuteReader method executes the command specified and returns an instance of instance of SqlDataReader class.
ExecuteScalar	This method executes the command specified and returns the first column of first row of the result set. The remaining rows and column are ignored
ExecuteXMLReader	This method executes the command specified and returns an instance of XmlReader class. This method can be used to return the result set in the form of an XML document

Question 29: Explain the ExecuteReader() method.

1. The DataReader object is a forward-only and read-only cursor.
2. It requires a live connection to the data source.
3. The DataReader object cannot be directly instantiated. Instead, we must call the ExecuteReader() method of the command object to obtain a valid DataReader object.

Example:

1. SqlDataReader reader = cmd.ExecuteReader(CommandBehavior.CloseConnection);

Question 30: Explain the ExecuteScalar method in ADO.NET?

Answer: The ExecuteScalar Method in SqlCommandObject returns the first column of the first row after executing the query against the Data Source.

1. If the result set contain more than one column or rows, it takes only the first column of the first row. All other values are ignored.
2. If the result set is empty it will return null.

Example:

1. Int32 TotalSalary = Convert.ToInt32(cmd.ExecuteScalar());

Question 31: Explain the ExecuteXmlReader?

Answer: The execute reader method is flexible when we need the result set in the form of an XML doucment. The ExecuteXmlReader method returns an instance of XmlReader class.

Example: *XmlReader xmlreader = cmd.ExecuteXmlReader();*
XmlDocument xdoc = new XmlDocument();

Using the XmlDocument class we load the XmlReader object and save it to the File System using the Save method.

Question 32: What method in the oleDbDataAdapter class populates a dataset with records?

Answer: The Fill() method populates a dataset with records.

DataSet: At Client (In Memory copy) ... SqlDataAdapter as bridge Fill(DataSet) ... Data Source: SqlServer
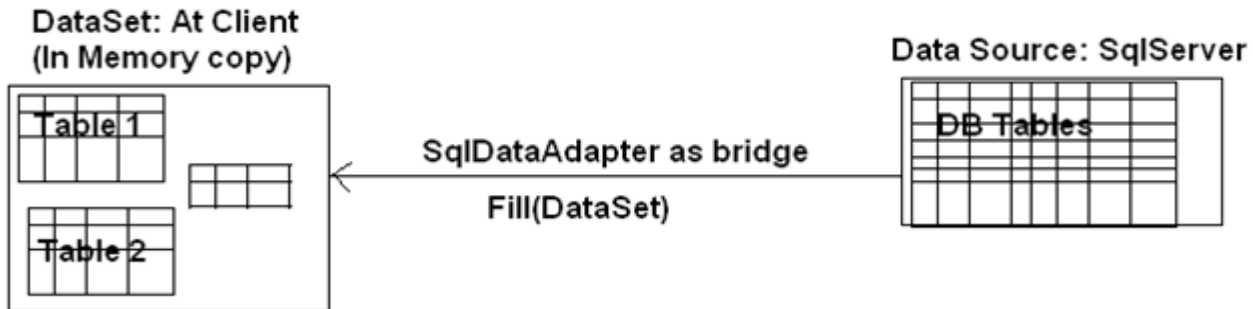
Example:

```
1.   private void button1_Click(object sender, EventArgs e)
2.   {
3.       SqlConnection con;
4.       SqlDataAdapter adapter;
5.       DataSet ds = new DataSet();
6.       try
7.       {
8.          //create connection object
9.          con = new SqlConnection("connetionString");
10.         //create query string(SELECT QUERY)
11.         String query = "SELECT * FROM SOMETABLE WHERE...";
12.         con.Open();
13.         //Adapter bind to query and connection object
14.         adapter = new SqlDataAdapter(query, con);
15.         //fill the dataset
16.         adapter.Fill(ds);
17.      }
18.      catch(Exception ex)
19.      {
20.         con.Close();
21.      }
22. }
```

Your DataSet contains the result table of SELECT query. You can play around the table and its data.

Question 33: Explain the OleDbDataAdapter Command Properties with Examples?

Answer:

Property	Example
SelectCommand	cmd.SelectCommand.CommandText = "SELECT * FROM Orders ORDER BY Price"
DeleteCommand	TheDataSetCommand.DeleteCommand.CommandText = "DELETE FROM orders WHERE LastName = 'Smith' ";
InsertCommand	TheDataSetCommand.InsertCommand.CommandText = "INSERT INTO Orders VALUE (25, 'Widget1', 'smith')";
UpdateCommand	TheDataSetCommand.UpdateCommand.CommandText = "UPDATE Orders SET ZipCode = '34956' WHERE OrderNum = 14";

The OLEDb Data Adapter methods:

Method	Description
Fill	This method fills data records from a DataAdapter to a DataSet object.
FillSchema	This method adds a DataTable to a DataSet.
GetFillParameters	This method retrieves parameters that are used when a SELECT statement is executed.
Update	This method stores data from a data set to the data source.

Question 34: Explain the clear() method of DataSet?

Answer: This method clears (removes) all rows from DataSet.

Create a button, set it's text as Clear and write the following code in the click event.

1. private void btnclear_Click(object sender, EventArgs e)
2. {
3.   ds.Clear();
4. }
5. Run the application.

Output:
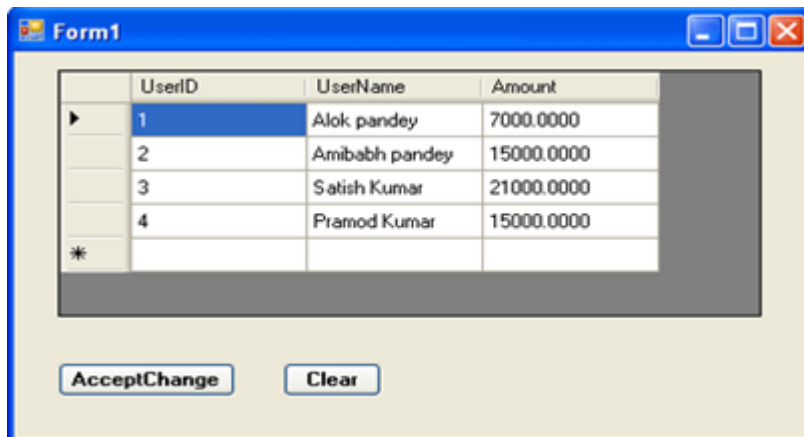
Click the "clear" button. You will note that all rows are clear as in the following screenshot,
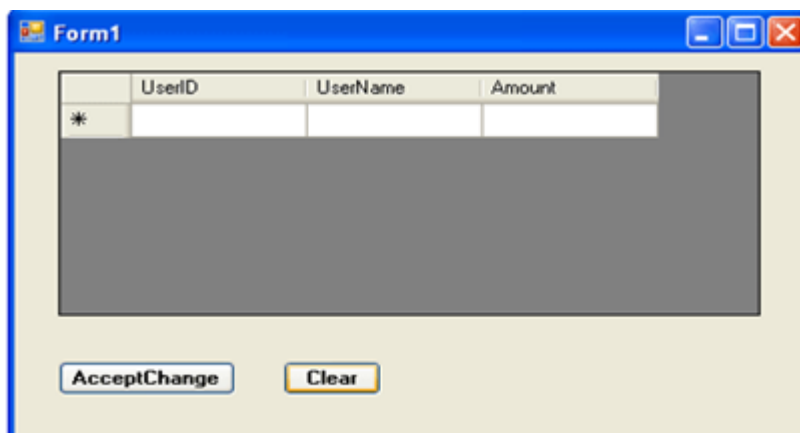
Question 35: What is clone() method of DataSet?

Answer: The clone() method, copy the structure of DataSet. Means it copy only schema not full records of DataSet.

Take another DataGridView and one button in your project and write the following code on button click event.

1. private void btnclone_Click(object sender, EventArgs e)
2. {
3.   DataSet daset = ds.Clone();
4.   dataGridView2.DataSource = daset.Tables[0];
5. }

Now run the application.

Output:
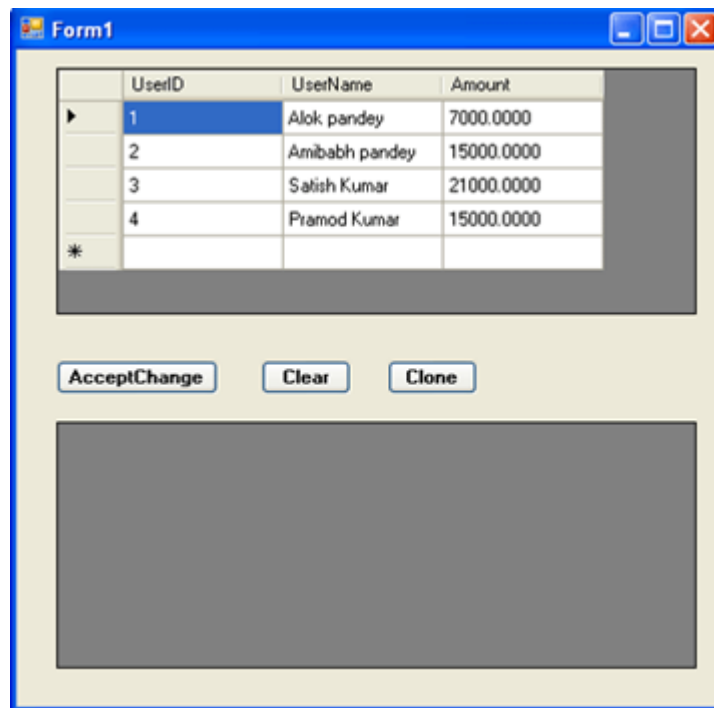
Click the "**Clone**" button. The following figure shows that only structure has copied.
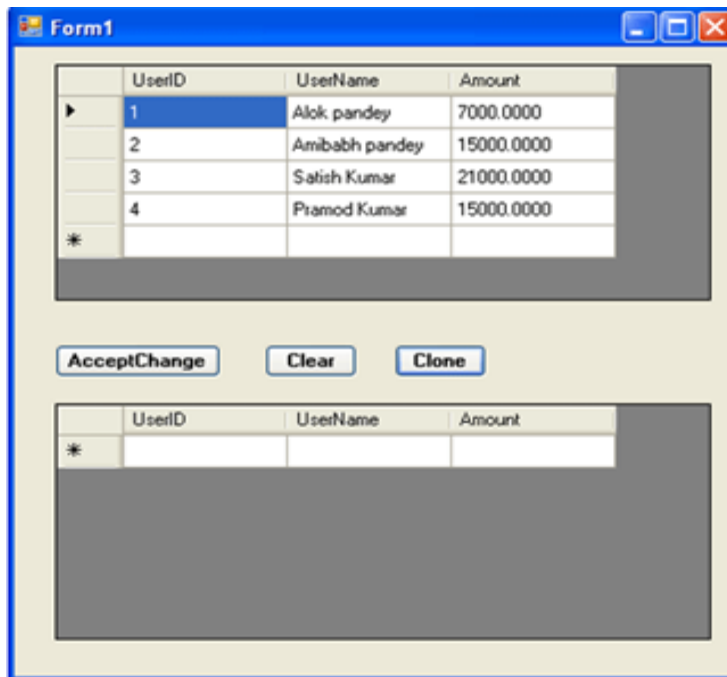
Question 36: What is the Copy() method of DataSet?
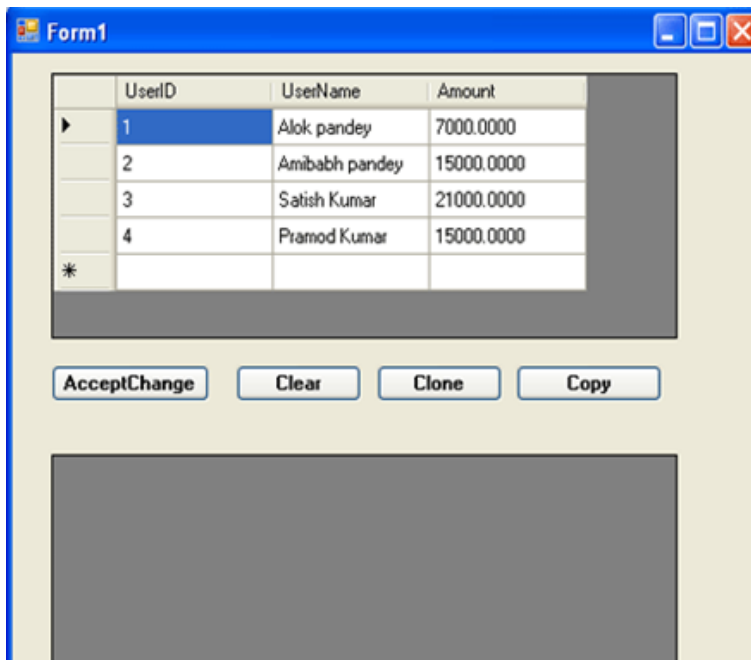
Answer: Copy the whole records with structure of DataSet.

Take a button and set it's text as copy and write the following code on click event.

```
1.  private void btncopy_Click(object sender, EventArgs e)
2.  {
3.      DataSet daset = ds.Copy();
4.      dataGridView2.DataSource = daset.Tables[0];
5.  }
```
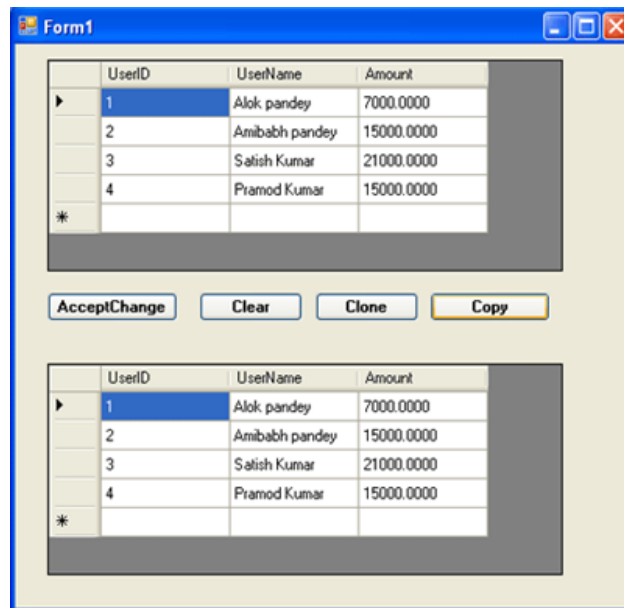
Now run the application.

Output:

Click the copy button.

Question 37: What is the HasChanges() method of DataSet?

Answer: This method return Boolean value to show whether record of Dataset has changed or not. It returns true if any changes made and false if no changes performed.

Take a button and set it's text as "HasChanges" and write the following code on button click.

Example:

```
1.  private void btnHasChanges_Click(object sender, EventArgs e)
2.  {
3.      if(ds.HasChanges())
4.      {
5.          MessageBox.Show("Changes Has Made");
6.      }
7.      if(!ds.HasChanges())
8.      {
9.          MessageBox.Show("No Change");
10.     }
11. }
```
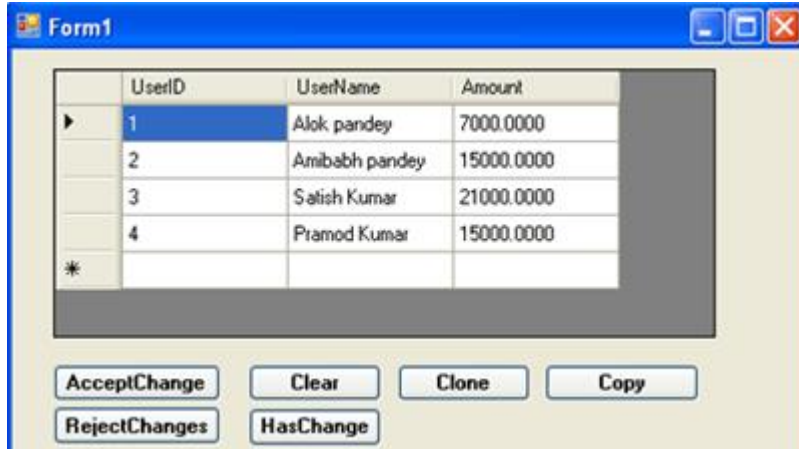
Run the application.
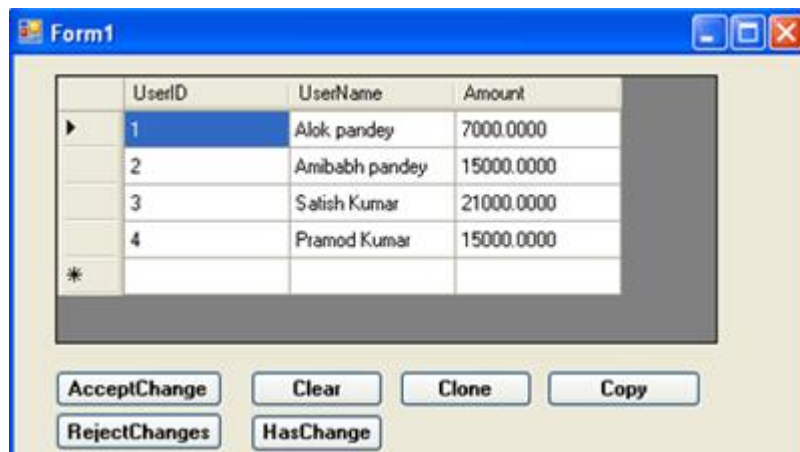
Output:

Now click at "**HasChanges**" button after doing some changes in dataset record.

Output:

Question 38: What are the Connection object properties and Connection class members?

Answer: The Connection class has a connection string that opens a connection to the database. The connection string will vary depending upon the provider used. The connection strings typically contain a group of property-value pair to describe how to connect to a database. For an OleDbConnection, you have properties such as Provider and DataSource.

Property	Description
ConnectionString	Represent the connection string.
ConnectionTimeOut	Waiting time while establishing a connection.
DataBase	Name of the current database.
DataSource	Location of the file name of the data source.
Provider	Name of the OLE DB provider. This property is not available for Sql and ODBC data providers.
State	Current state of the connection of type ConnectionState. (Table 5-17 describes the ConnectionState).
PacketSize	Size of network packets. Available to only Sql data providers.
ServerVersion	SQL server version. Available to only Sql data providers.
WorkStationId	Database client ID. Available to only Sql data providers.

The connection can have different states such as open, closed, connecting, and so on. The ConnectionType enumeration defines the members of the ConnectionState.

The connection Class Members:

Method	Description
BeginTransaction	Begins database transaction.
ChangeDatabase	Changes databases for an open connection.
Close	Closes an opened connection.
CreateCommand	Creates and return a Command object depends on the data providers. For example, OleDb Connection returns OleDbCommand, and SqlConnection returns SqlCommand.
Open	Open a new connection.
ReleaseObjectPool	Represents that the connection pooling can be cleared when the provider is released. Available only for Ole Db data providers.

Question 39: What is the preferred method for executing SQL commands that contain parameters?

Answer: The preferred method is to use the SqlParameter and oleDbParameter objects, as detailed in the section "Using Parameters with the command Object."

The SqlParameter class is found in the "**System.Data.SqlClient**" namespace. It is a class of a connected architecture of .NET framework. It represents parameters. To work with the SqlParameter class we should have a database.

Question 40: How DataSet objects in ADO.NET replace the ADO Recordset object?

Answer: DataSet is good for ADO.NET objects to replace the ADO Recordset object-

- DataSet can hold multiple tables at a time.
- It allows data access to easily read or write data operations in / from the database.
- DataSet data stores in local system.
- It holds multiple rows at a time.
- It uses more memory.
- DataSet maintain relation.
- It bind data from the database.

DataSet representation in .aspx.cs code-

```
1.  protected void BindDataSet()
2.  {
3.     SqlConnection con = new SqlConnection("your database connection string ");
4.     con.Open();
5.     SqlCommand cmd = new SqlCommand("Write your query or procedure ", con);
6.     SqlDataAdapter da = new SqlDataAdapter(cmd);
7.     DataSet ds = new DataSet();
8.     da.Fill(ds);
9.     grid.DataSource = ds;
10.    grid.DataBind();
11. }
```

Question 41: What is Transactions and Concurrency in ADO.NET?

Answer: Transactions: ADO.NET providers a transaction class that represents a transaction. All data providers provide their own version of the transaction class. The IDbTransaction interface implements the basic functionality of the transaction class. All data provider-specific classes implement this namespace.

Figure shows some of the classes that implement IDbTransaction.
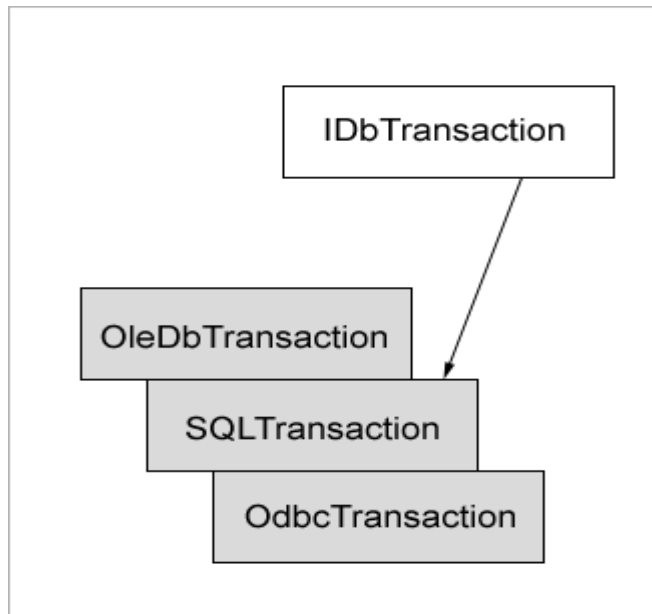
Methods of the Transaction Class:

Method	Description
Commit	Commits the transaction to the database
Rollback	Rollbacks a transaction to the previous database state
Begin(IsolationLevel)	Begins a nested database transaction passing the isolation level

Concurrency in ADO.NET:

The ADO.NET model assumes that the optimistic concurrency is the default concurrency because of its disconnected nature of data. A user reads data in a data through a data adapter, and data is available to user as a local copy of the data. The server database is available to all other users.

Another way of handling optimistic concurrency that you may be familiar with is by checking to see if a timestamp on the data source row has changed or the row version number has changed on the row being updated.

Pessimistic locking on the database isn't really supported by the data providers because the connection to the database is not kept open, so you must perform all locking with business logic on the DataSet.

Question 42: What is the ADO.NET Data provider?

Answer: There are four .NET data providers available.

1. **SQL Server:** It is used to work specifically with Microsoft SQL Server. It exists in a namespace within the System.Data.SqlClient.

2. **OLE DB:** It is used to work with the OLEDB provider. The System.Data.dll assembly implements the OLEDB .NET framework data provider in the System.Data.OleDb namespace.

3. **ODBC:** To use this type of provider, you must use an ODBC driver. The System.Data.ODBC.dll assembly implements the ODBC .NET framework data provider. This assembly is not part of the Visual Studio .NET installation.

4. **Oracle:** The System.Data.OracleClient.dll assembly implements the Oracle .NET framework data provider in the System.Data.OracleClient namespace. The Oracle client software must be installed on the system before you can use the provider to connect to an Oracle data source.

Question 43: How can we Create and Manage Connections In ADO.NET?

Answer: Creating a Connection object-

The connection component of a dataprovider establishes a connection with a database. To connect to Microsoft SQL Server, you use the SQL connection class. The following are the commonly used properties and methods of the SqlConnection class.

ConnectionString: It provides information, such as database name and user credentials for database access and so on.

Open(): Opens the connection for accessing the database.

Close(): Closes the connection to the database.

For Example:

1. // Creating object of SqlConnection Class.
2. SqlConnection cn = new SqlConnection();
3. //Creating connection string to sample database.
4. cn.ConnectionString = "Data source=.; Initial Catalog=Sample; User Id=sa; Password=fac ulty";

The connection string provides the information that defines the connection to the database.

- **Data Source:** Specifies the provider name or your server name.
- **Initial Catalog:** Specifies the name of the database.
- **User Id and Password:** Provide the username and password of your database server.

Open the Connection:

1.   // Creating object of SqlConnection Class.
2. SqlConnection cn = new SqlConnection();
3. //Creating connection string to sample database.
4. cn.ConnectionString = "Data source=.; Initial Catalog=Sample; User Id=sa; Password=fa culty";
5. cn.Open(); // it open the connection to database server..

Close the Connection:

1. // Creating object of SqlConnection Class.
2. SqlConnection cn = new SqlConnection();
3. //Creating connection string to sample database.
4. cn.ConnectionString = "Data source=.; Initial Catalog=Sample; User Id=sa; Password=fac ulty";
5. cn.Open(); // it open the connection to database server..
6. //Creating sqlcommand class object
7. SqlCommand cmd = new SqlCommand("Select * from tblEmployees", cn);
8. SqlDataReader dr = cmd.ExecuteReader();//Executing query
9. cn.Close();//Closing the connection

Question 44: What is disconnected data?

Answer: A data representation, such a DataSet, that doesn't require a continuous database connection. Working with disconnected data:

The data in DataSet is disconnected from database. Once you fetch the results of a query into a DataSet using a DataAdapter object, there is no longer a connection between DataSet and database. Changes you make to the contents of the DataSet will not affect the database. If other users modify data in database that corresponds to the data in DataSet, you will not see those changes in your DataSet.

Working with disconnected data structures definitely has its benefits. The first major benefit of working with disconnected data is that it does not require a live connection to your database. Once you've fetched the results of your query into a DataSet object, you can close the connection to your database and continue to work with the data in your DataSet.

Disconnected data structures such as DataSets are also helpful when you build multi-tiered applications. If your application uses business objects running on a middle-tier server to access database, business object needs to pass disconnected data structures to client application. The DataSet object is designed for use in such situations. You can pass the contents of a DataSet from one component to another. The component that receives the data can work with the information as a DataSet (if the component is built using the Microsoft .NET Framework) or as an XML document.

Question 45: Explain the DataTable and Relationship between the DataTable, the DataRow, and the DataColumn.

Answer: A DataTable object represents a database table. A data table is a collection of columns and rows. The DataRow object represents a table row, and the DataColumn object represents a column of the table.

The Columns property of the DataTable object represents the DataColumnCollection, which is a collection of DataColumn objects in a DataTable. You use a DataRow object to add data to a data table. TheDataRowCollection object represents a collection of rows of a DataTable object, which can be accessed by its Rows property.

Figure shows the relationship between the DataTable, DataRow, and DataColumn.
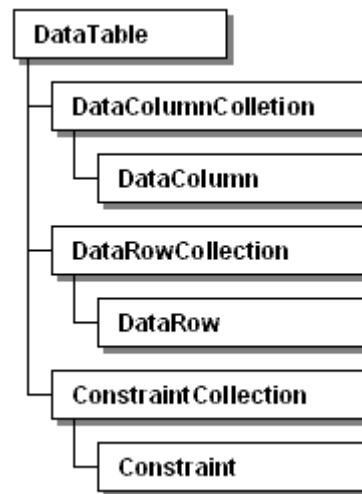
Question 46: Explain about Data Access Object or DAO?

Answer: Data Access Object (DAO) enabled programmers to access local databases in the Microsoft Jet Database Engine format, which were primarily Indexed Sequential Access Method (ISAM) files. After DAO came RDO and then ActiveX Data Objects (ADO). These data access technologies were designed for a client / server paradigm. However the tendency of distributed computing forced the development of a new technology to solve the problems of data manipulation on a n-tier architecture. ADO.NET is the evolution of ADO and its components have been designed to function properly on n-tier architecture.

Summary of the evolution of the database objects to access data from Microsoft:

Name	Brief	Description
DAO	Data Access Objects	The first object-oriented interface that exposed the Microsoft Jet database engine that allowed developers using Visual Basic to directly connect to Access tables and other databases using ODBC. It is ideal for small databases in local deployments and single-system applications.
RDO	Remote Data Objects	An object-oriented data access interface to ODBC combined with the easy functionality of DAO allowing an interface to almost all of ODBC's low power and flexibility. RDO can't access Jet of ISAM databases in an efficient way. RDO provides the objects, properties, and methods needed to access the more complex aspects of stored procedures and complex resultsets.
ADO	Microsoft ActiveX Data Objects	ADO is the successor to DAO/RDO. ADO is the consolidation of almost all the functionality of DAO and RDO. ADO mostly includes RDO-style functionality to interact with OLE DB data sources, plus remoting and DHTML technology.
ADO MD	Microsoft ActiveX Data Objects Multidimensional	Provides easy access to multidimensional data from languages such as Microsoft Visual Basic and Microsoft Visual C++. ADO MD extends Microsoft ActiveX Data Objects (ADO) to include objects specific to multidimensional data, such as the CubeDef and Cellset objects. To work with ADO MD, the provider must be a multidimensional data provider (MDP) as defined by the OLE DB for OLAP specification. MDPs present data in multidimensional views as opposed to tabular data providers (TDPs) that present data in tabular views. With ADO MD you can browse multidimensional schema, query a cube, and retrieve the results.
ADOX	Microsoft ActiveX Data Objects	Extensions for Data Definition Language and Security Is an extension to the ADO objects and programming model. ADOX includes objects for schema creation and modification, as well as security. Because it is an object-based approach to schema manipulation, you can write code that will work against various data sources regardless of differences in their native syntaxes.ADOX is a companion library to the core ADO objects. It exposes additional objects for creating, modifying, and deleting schema objects, such as tables and procedures. It also includes security objects to maintain users and groups and to grant and revoke permissions on objects.
RDS	Remote Data Service	You can move data from a server to a client application or Web page, manipulate the data on the client, and return updates to the server in a single round trip.
ADO.NET	Microsoft ActiveX Data Objects	.NET ADO.NET is entirely based on XML. ADO.NET provides consistent access to data sources such as Microsoft SQL Server, as well as data sources exposed via OLE DB and XML. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, manipulate, and update data. ADO.NET cleanly factors data access from data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, or placed in an ADO.NET DataSet object in order to be exposed to the user in an ad-hoc manner, combined with data from multiple sources, or remoted between tiers. The ADO.NET DataSet object can also be used independently of a .NET data provider to manage data local to the application or sourced from XML. The ADO.NET classes are found in System.Data.dll, and are integrated with the XML classes found in System.Xml.dll. When compiling code that uses the System.Data namespace, reference both System.Data.dll and System.Xml.dll. Compared with ADO there is no Recordset object. In ADO.NET there are four classes that read and write data from data sources: 1. Connection .- Connect to data source 2. Command .- Execute stored procedures 3. DataAdapter .- Connects DataSet to database 4. DataReader .- Forward/only, read/only cursor

Question 47: What are the advantages of ADO.NET?

Answer: ADO.NET offers several advantages over previous Microsoft data access technologies, including ADO. Few advantages are listed below:

Single Object-oriented API-

ADO.NET provides a single object-oriented set of classes. There are different data providers to work with different data sources but the programming model for all these data providers work in the same way.

Managed Code-

The ADO.NET classes are managed classes. CLR takes care of language independency and automatic resource management.

Deployment-

Microsoft uses MDAC (Microsoft Data Access Component), which is used as ActiveX component in .NET Framework (X is extensible component, when X is written after a term means extensible). .NET components takes care of deployment which was difficult than the previous technologies used in deployment.

XML Support-

ADO.NET data is cached and transferred in XML (EXtensible Markup Language) format. XML provide fast access of data for desktop and distributed applications.

Performance and scalability-

Performance and scalability are two major factors while developing web-based application and services. Disconnected cached data in XML help in performance and scalability.

Question 48: What is GetChanges() method in ADO.NET?

Answer: The GetChanges method of DataSet can be used to retrieve the rows that have been modified since the last time DataSet was filled, saved or updated. The GetChanges method returns a DataSet object with modified rows.

The GetChanges method can take either no argument or one argument of type DataRowState. The DataRowState enumeration defiles the DataRow state, which can be used to filter a DataSet based on the types of rows.

DataRowState members:

Member	Description
Added	Add added rows to a DataRowCollection of a DataSet and AcceptChanges has not been called.
Deleted	All the deleted rows.
Detached	Rows were created but not added to the row collection. Either waiting for the addition or have removed from the collection.
Modified	Modified rows and AcceptChanges has not been called.
Unchanged	Unchanged rows since last AcceptChanges was called.

Question 49: How can you access the data from DataReader?

Answer: DataReader is a class that holds data as rows and coloums to access data from the DataReader.

It provides the following methods:

1. **GetName(int ColIndex) -** The return type of this method is a string, it returns the name of the column for the given index position.

2. **Read() -** Moves the Record Pointer from the current location to the next row and returns a Boolean status that tells whether the row to which we have moved contains data in it or not, that will be true if present or false if not present.

3. **GetValue(int Colindex) -** Returns a column's value from the row to which the pointer was pointing by specifying the column index position.

4. **NextResult () -** Moves the record pointer from the current table to the next table if a table exists and returns true else returns false.

Features of Data Reader:

1. Provides faster access to data from a Data Source, since it is connection oriented.

2. It can hold multiple tables at a time. To load multiple tables into a DataReader pass multiple select statements as the argument to the command separated by a colon (;).

For example:

1. Command cmd=new Command("Select * From Student ; Select * From Mark ", Con);
2. Data Reader dr= cmd.ExecuteReader();

Question 50: What is BindingSource class in ADO.NET?

Answer: The BindingSource class is used to simplify data binding as well as various operations on records. It has different methods like AddNew( ), MoveFirst( ), MovePrevious( ), MoveNext( ), etc which provide easier way for adding new row, moving to first record, moving to previous record, moving to next record and many other operations without writing code for them.

Chapter 4

ASP.NET Interview Questions and Answers

Question 1: What is ASP.NET?

Answer: ASP.NET was developed in direct response to the problems that developers had with classic ASP. Since ASP is in such wide use, however, Microsoft ensured that ASP scripts execute without modification on a machine with the .NET Framework (the ASP engine, ASP.DLL, is not modified when installing the .NET Framework). Thus, IIS can house both ASP and ASP.NET scripts on the same machine.

Advantages of ASP.NET:

Separation of Code from HTML:

To make a clean sweep, with ASP.NET you have the ability to completely separate layout and business logic. This makes it much easier for teams of programmers and designers to collaborate efficiently.

Support for compiled languages:

Developer can use VB.NET and access features such as strong typing and object-oriented programming. Using compiled languages also means that ASP.NET pages do not suffer the performance penalties associated with interpreted code. ASP.NET pages are precompiled to byte-code and Just In Time (JIT) compiled when first requested. Subsequent requests are directed to the fully compiled code, which is cached until the source changes.

Use services provided by the .NET Framework:

The .NET Framework provides class libraries that can be used by your application. Some of the key classes help you with input/output, access to operating system services, data access, or even debugging. We will go into more detail on some of them in this module.

Graphical Development Environment:

Visual Studio .NET provides a very rich development environment for web developers. You can drag and drop controls and set properties the way you do in Visual Basic 6. And you have full IntelliSense support, not only for your code, but also for HTML and XML.

State management:

To refer to the problems mentioned before, ASP.NET provides solutions for session and application state management. State information can, for example, be kept in memory or stored in a database. It can be shared across web farms, and state information can be recovered, even if the server fails or the connection breaks down.

Update files while the server is running:

Components of your application can be updated while the server is online and clients are connected. The framework will use the new files as soon as they are copied to the application. Removed or old files that are still in use are kept in memory until the clients have finished.

XML-Based Configuration Files:

Configuration settings in ASP.NET are stored in XML files that you can easily read and edit. You can also easily copy these to another server, along with the other files that comprise your application.

ASP.NET Overview: Here are some points that give the quick overview of ASP.NET.

- ASP.NET provides services to allow the creation, deployment, and execution of Web Applications and Web Services.

- Like ASP, ASP.NET is a server-side technology.

- Web Applications are built using Web Forms. ASP.NET comes with built-in Web Forms controls, which are responsible for generating the user interface. They mirror typical HTML widgets like text boxes or buttons. If these controls do not fit your needs, you are free to create your own user controls.

- Web Forms are designed to make building web-based applications as easy as building Visual Basic applications.

Question 2: What are the different validators in ASP.NET?

Answer: ASP.NET validation controls define an important role in validating the user input data. Whenever the user gives the input, it must always be validated before sending it across to various layers of an application. If we get the user input with validation, then chances are that we are sending the wrong data. So, validation is a good idea to do whenever we are taking input from the user.

There are the following two types of validation in ASP.NET:

- Client-Side Validation
- Server-Side Validation

Client-Side Validation:

When validation is done on the client browser, then it is known as Client-Side Validation. We use JavaScript to do the Client-Side Validation.

Server-Side Validation:

When validation occurs on the server, then it is known as Server-Side Validation. Server-Side Validation is a secure form of validation. The main advantage of Server-Side Validation is if the user somehow bypasses the Client-Side Validation, we can still catch the problem on server-side.

The following are the Validation Controls in ASP.NET:

- RequiredFieldValidator Control

- CompareValidator Control

- RangeValidator Control

- RegularExpressionValidator Control

- CustomFieldValidator Control

- ValidationSummary

Question 3: What is View State?

Answer: View State is the method to preserve the Value of the Page and Controls between round trips. It is a Page-Level State Management technique. View State is turned on by default and normally serializes the data in every control on the page regardless of whether it is actually used during a post-back.

A web application is stateless. That means that a new instance of a page is created every time when we make a request to the server to get the page and after the round trip our page has been lost immediately.

Features of View State:

These are the main features of view state-

1. Retains the value of the Control after post-back without using a session.
2. Stores the value of Pages and Control Properties defined in the page.
3. Creates a custom View State Provider that lets you store View State Information in a SQL Server Database or in another data store.

Advantages of View State:

These are the main advantages of view state-

1. Easy to Implement.
2. No server resources are required: The View State is contained in a structure within the page load.
3. Enhanced security features: It can be encoded and compressed or Unicode implementation.

Question 4: What are the different Session state management options available in ASP.NET?

Answer: State Management in ASP.NET-

- A new instance of the Web page class is created each time the page is posted to the server.

- In traditional Web programming, all information that is associated with the page, along with the controls on the page, would be lost with each roundtrip.

- The Microsoft ASP.NET framework includes several options to help you preserve data on both a per-page basis and an application-wide basis.
  These options can be broadly divided into the following two categories:

  - Client-Side State Management Options
  - Server-Side State Management Options

Client-Side State Management:

- Client-based options involve storing information either in the page or on the client computer.

- Some client-based state management options are:

  - Hidden fields
  - View state
  - Cookies
  - Query strings

Server-Side State Management:

- There are situations where you need to store the state information on the server side.

- Server-side state management enables you to manage application-related and session-related information on the server.

- ASP.NET provides the following options to manage state at the server side:

  - Application state
  - Session state
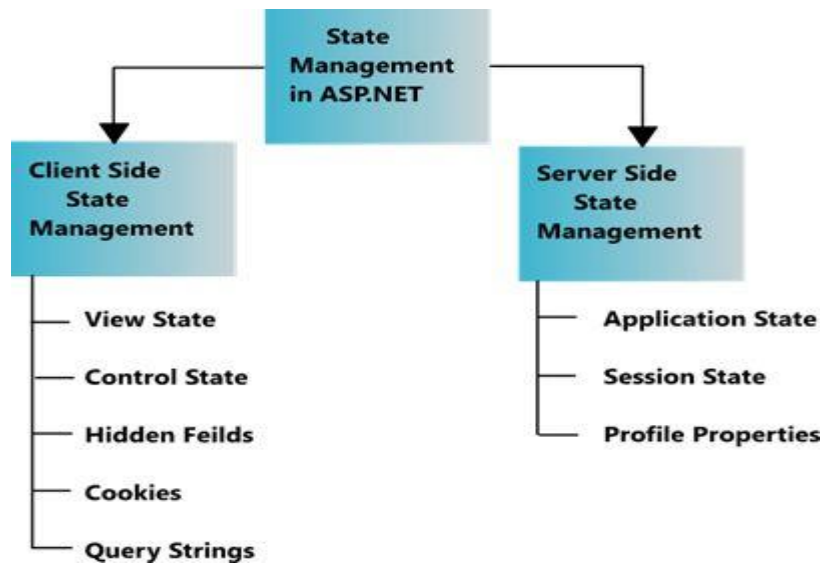
Question 5: What is caching in ASP.NET?

Answer: Caching is one of the most interesting concept and operation in ASP.NET. If you can handle it, you can run any web application by applying the caching concept depending on the requirements.

Caching is for providing solutions or the results to the users depending on their request, admin needs to recreate the pages often depending on user requests…STOP!!! "A cache simply stores the output generated by a page in the memory and this saved output (cache) will serve us (users) in the future."

Types

Page caching

Fragment Caching

Data Caching

Question 6: How can we apply themes in ASP.NET application?

Answer: A theme is a collection of settings that define the look of controls and web pages. These themes are applied across all the pages in a web application to maintain a consistent appearance. Themes are included images and skin files; the skin files set the visual properties of ASP.NET controls. Themes are of two types:

Page Theme:

A Page theme contains the control skins, style sheets, graphic files, and other resources inside the subfolder of the App_Theme folder in the Solution Explorer window. A page theme is applied to a single page of the web site.

Global Theme:

A Global theme is a theme that is applied to all the web sites on a web server and includes property settings, and graphics. This theme allows us to maintain all the websites on the same web server and define the same style for all the web pages of the web sites.
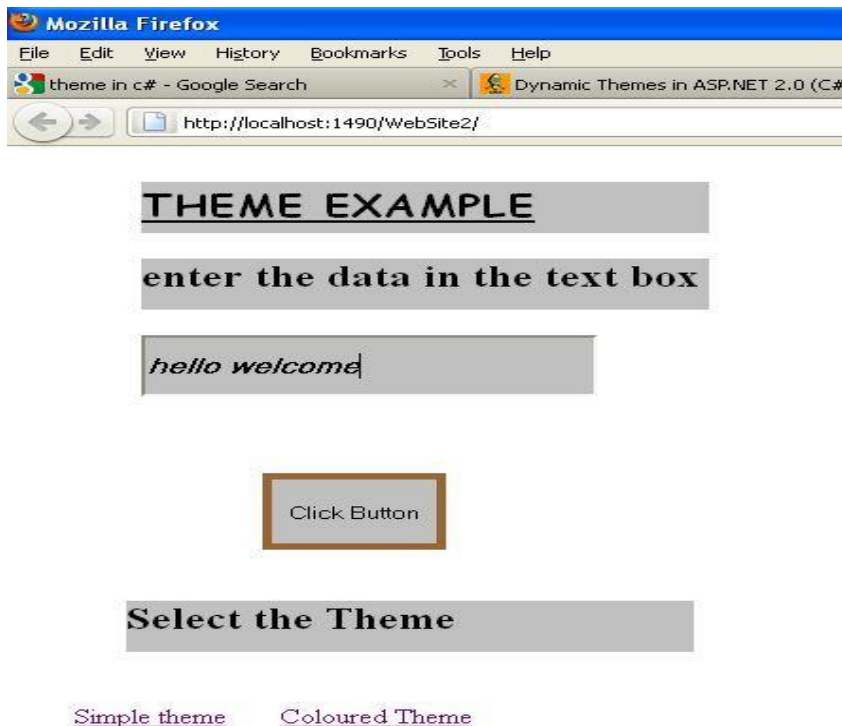
Question 7: What is MVC?

Answer: Model-View-Controller (MVC) is a pattern to separate an application into the following three main components:

1. Model
2. View
3. Controller

The ASP.NET MVC framework provides an alternative to the ASP.NET Web Forms pattern for creating web applications. The ASP.NET MVC Framework is a lightweight, highly testable presentation framework that (as with Web Forms-based applications) is integrated with existing ASP.NET features, such as master pages and membership-based authentications. The MVC framework is defined in the **System.Web.Mvc** assembly. It provides full control over HTML, JavaScript and CSS. It's the better as well as a recommended approach for large-scale applications where various teams are working together.
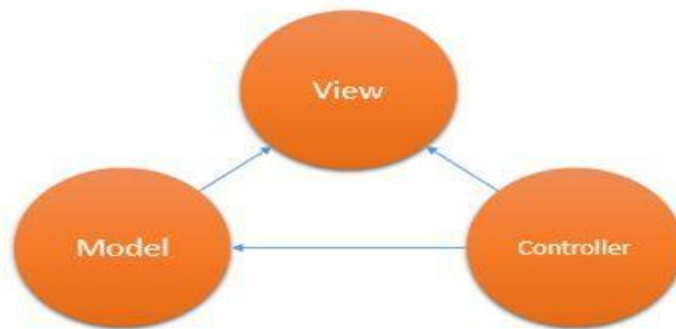
The ASP.NET MVC framework offers the following advantages:

- It makes it very easy to manage complexity by dividing an application into the Model, View and Controller.
- It does not use view state or server-based forms.
- Full control over HTML, JavaScript and CSS.
- It provides better support for Test-Driven Development (TDD).
- It works well for Web applications that are supported by large teams of developers and for web designers who need a high degree of control over the application behaviour.
- By default support of Facebook and Google Authentication.
- It easy to manage a large application to divide in multiple areas.

Question 8: What are Cookies in ASP.NET?

Answer: Cookies are a State Management Technique that can store the values of control after a post-back. Cookies can store user-specific Information on the client's machine like when the user last visited your site. Cookies are also known by many names, such as HTTP Cookies, Browser Cookies, Web Cookies, Session Cookies and so on. Basically cookies are a small text file sent by the web server and saved by the Web Browser on the client's machine.

List of properties containing the HttpCookies Class:

1. **Domain:** Using these properties we can set the domain of the cookie.
2. **Expires:** This property sets the Expiration time of the cookies.
3. **HasKeys:** If the cookies have a subkey then it returns True.
4. **Name:** Contains the name of the Key.
5. **Path:** Contains the Virtual Path to be submitted with the Cookies.
6. **Secured:** If the cookies are to be passed in a secure connection then it only returns True.
7. **Value:** Contains the value of the cookies.

Limitation of the Cookies:

1. The size of cookies is limited to 4096 bytes.
2. A total of 20 cookies can be used in a single website.

Question 9: What is Ajax in ASP.NET?

Answer: Ajax stands for Asynchronous JavaScript and XML; in other words Ajax is the combination of various technologies such as a JavaScript, CSS, XHTML, DOM, etc.

AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the entire page.

We can also define Ajax is a combination of client side technologies that provides asynchronous communication between the user interface and the web server so that partial page rendering occurs instead of complete page post back.

Ajax is platform-independent; in other words AJAX is a cross-platform technology that can be used on any Operating System since it is based on XML & JavaScript. It also supports open source implementation of other technology. It partially renders the page to the server instead of complete page post back. We use AJAX for developing faster, better and more interactive web applications. AJAX uses a HTTP request between web server & browser.

- With AJAX, when a user clicks a button, you can use JavaScript and DHTML to immediately update the UI, and spawn an asynchronous request to the server to fetch results.

- When the response is generated, you can then use JavaScript and CSS to update your UI accordingly without refreshing the entire page. While this is happening, the form on the users screen doesn't flash, blink, disappear, or stall.

- The power of AJAX lies in its ability to communicate with the server asynchronously, using an XMLHttpRequest object without requiring a browser refresh.

- Ajax essentially puts JavaScript technology and the XMLHttpRequest object between your Web form and the server.

Question 10: What are Web Services in ASP.NET?

Answer: A Web Service is a software program that uses XML to exchange information with other software via common internet protocols. In a simple sense, Web Services are a way for interacting with objects over the Internet.

A web service is:

- Language Independent.
- Protocol Independent.
- Platform Independent.
- It assumes stateless service architecture.
- Scalable (e.g. multiplying two numbers together to an entire customer-relationship management system).
- Programmable (encapsulates a task).
- Based on XML (open, text-based standard).

- Self-describing (metadata for access and use).
- Discoverable (search and locate in registries)- ability of applications and developers to search for and locate desired Web services through registries. This is based on UDDI.

Key Web Service Technologies:

- **XML-** Describes only data. So, any application that understands XML-regardless of the application's programming language or platform-has the ability to format XML in a variety of ways (well-formed or valid).
- **SOAP-** Provides a communication mechanism between services and applications.
- **WSDL-** Offers a uniform method of describing web services to other programs.
- **UDDI-** Enables the creation of searchable Web services registries.
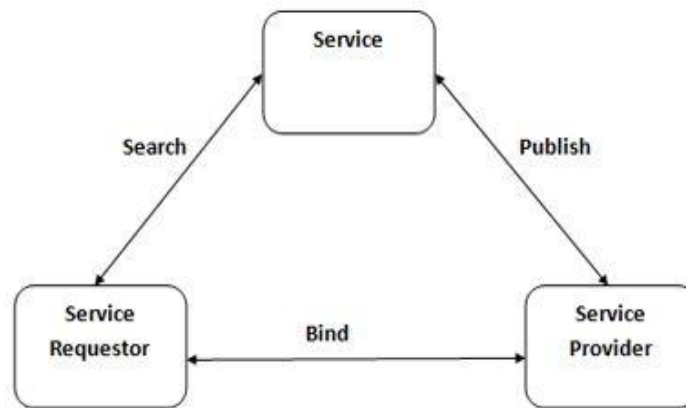
Question 11: What are the Advantages of ASP.NET?

Answer: ASP.NET provides services to allow the creation, deployment, and execution of Web Applications and Web Services like ASP, ASP.NET is a server-side technology. Web Applications are built using Web Forms. ASP.NET comes with built-in Web Form controls, which are responsible for generating the user interface. They mirror typical HTML widgets such as text boxes or buttons. If these controls do not fit your needs, you are free to create your own user controls.

Advantages of ASP.NET:

- Separation of Code from HTML
- Support for compiled languages
- Use services provided by the .NET Framework
- Graphical Development Environment
- Update files while the server is running
- XML-Based Configuration Files

Question 12: What is the concept of Globalization and Localization in .NET?

Answer: Localization means "**process of translating resources for a specific culture**", and Globalization means "**process of designing applications that can adapt to different cultures**".

- **Proper Globalization:** Your application should be able to Accept, Verify, and Display all global kind of data. It should well also be able to operate over this data, accordingly. We will discuss more about this "**Accordingly operations over diff. culture data**".

- **Localizability and Localization:** Localizability stands for clearly separating the components of culture based operations regarding the user interface, and other operations from the executable code.

.NET framework has greatly simplified the task of creating the applications targeting the clients of multiple cultures. The namespaces involved in creation of globalize, localizing applications are:

- System.Globalization
- System.Resources
- System.Text

Question 13: What is the Web.config file in ASP?

Answer: Configuration file is used to manage various settings that define a website. The settings are stored in XML files that are separate from your application code. In this way you can configure settings independently from your code. Generally a website contains a single Web.config file stored inside the application root directory. However there can be many configuration files that manage settings at various levels within an application.

Usage of configuration file:

ASP.NET Configuration system is used to describe the properties and behaviors of various aspects of ASP.NET applications. Configuration files help you to manage the settings related to your website. Each file is an XML file (with the extension .config) that contains a set of configuration elements. Configuration information is stored in XML-based text files.

Benefits of XML-based Configuration files:

- ASP.NET Configuration system is extensible and application specific information can be stored and retrieved easily. It is human readable.
- You need not restart the web server when the settings are changed in configuration file. ASP.NET automatically detects the changes and applies them to the running ASP.NET application.
- You can use any standard text editor or XML parser to create and edit ASP.NET configuration files.

Question 14: What is the App Domain Concept in ASP.NET?

Answer: ASP.NET introduces the concept of an Application Domain which is shortly known as **AppDomain**. It can be considered as a Lightweight process which is both a container and boundary. The .NET runtime uses an **AppDomain** as a container for code and data, just like the operating system uses a process as a container for code and data. As the operating system uses a process to isolate misbehaving code, the .NET runtime uses an **AppDomain** to isolate code inside a secure boundary.

The CLR can allow the multiple .NET applications to run in a single AppDomain. Mulitple Appdomains can exist in Win32 process.

How to create AppDomain: AppDomains are created using the CreateDomain method. AppDomain instances are used to load and execute assemblies (Assembly). When an AppDomain is no longer in use, it can be unloaded.
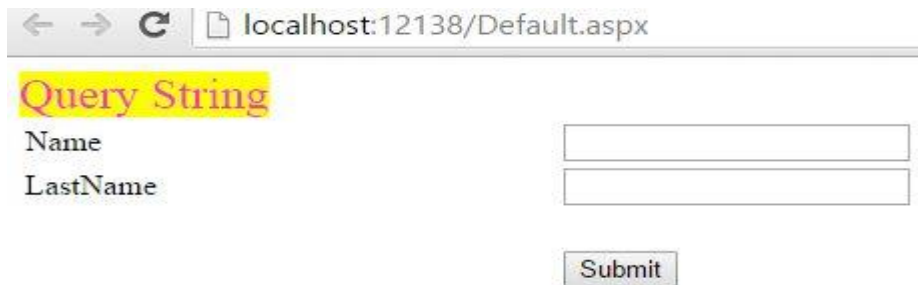
```
1.  public class MyAppDomain: MarshalByRefObject
2.  {
3.     public string GetInfo()
4.     {
5.        return AppDomain.CurrentDomain.FriendlyName;
6.     }
7.  }
8.  public class MyApp
9.  {
10.    public static void Main()
11.    {
12.       AppDomain apd = AppDomain.CreateDomain("Rajendrs Domain");
13.       MyAppDomain apdinfo = (MyAppDomain) apd.CreateInstanceAndUnwrap(Assembly.GetCallingAssembly()
14.          .GetName()
15.          .Name, "MyAppDomain");
16.       Console.WriteLine("Application Name = " + apdinfo.GetInfo());
17.    }
18. }
```

Question 15: What is Query String in ASP?

Answer: A QueryString is a collection of characters input to a computer or web browser. A Query String is helpful when we want to transfer a value from one page to another. When we need to pass content between the HTML pages or aspx Web Forms in the context of ASP.NET, a Query String is Easy to use and the Query String follows a separating character, usually a Question Mark (?). It is basically used for identifying data appearing after this separating symbol. A Query String Collection is used to retrieve the variable values in the HTTP query string. If we want to transfer a large amount of data then we can't use the Request.QueryString. Query Strings are also generated by form submission or can be used by a user typing a query into the address bar of the browsers.

Syntax of Query String:

Request.QueryString(variable)[(index).count]

Advantages:

- Simple to Implement
- Easy to get information from Query string.
- Used to send or read cross domain (from different domain).

Disadvantages:

- Human Readable
- Client browser limit on URL length
- Cross paging functionality makes it redundant
- Easily modified by end user

Question 16: What is master page in ASP.NET?

Answer: The extension of MasterPage is '**.master**'. MasterPage cannot be directly accessed from the client because it just acts as a template for the other Content Pages. In a MasterPage we can have content either inside ContentPlaceHolder or outside it. Only content inside the **ContentPlaceHolder** can be customized in the Content Page. We can have multiple masters in one web application.A MasterPage can have another MasterPage as Master to it. The MasterPageFile property of a webform can be set dynamically and it should be done either in or before the Page_PreInit event of the WebForm. **Page.MasterPageFile = "MasterPage.master".** The dynamically set Master Page must have the ContentPlaceHolder whose content has been customized in the WebForm.

A master page is defined using the following code:

<%@ master language="C#" %>

Adding a MasterPage to the Project:

1. Add a new MasterPage file (MainMaster.master) to the Web Application.
2. Change the Id of ContentPlaceHolder in <Head> to "**cphHead**" and the Id "**ContentPlaceHolder1**" to "**cphFirst**".
3. Add one more ContentPlaceHolder (cphSecond) to Master page.
4. To the master page add some header, footer and some default content for both the content place holders.
    1. <form id="form1" runat="server"> Header...
    2.
    3.   <asp:ContentPlaceHolder id="cphFirst" runat="server"> This is First Content Place Holder (Default) </asp: ContentPlaceHolder>
    4.
    5.   <asp:ContentPlaceHolder ID="cphSecond" runat="server">

This is Second Content Place Holder (Default).

1. </asp:ContentPlaceHolder>
2.
 Footer...
3. </form>

Question 17: What is tracing in .NET?

Answer: Tracing helps to see the information of issues at the runtime of the application. By default Tracing is disabled.

Tracing has the following important features:

1. We can see the execution path of the page and application using the debug statement.
2. We can access and manipulate trace messages programmatically.
3. We can see the most recent tracing of the data.

Tracing can be done with the following 2 types.

1. **Page Level:** When the trace output is displayed on the page and for the page-level tracing we need to set the property of tracing at the page level.

   <%@ Page Trace="true" Language="C#"

2. **Application:** Level: In Application-Level tracing the information is stored for each request of the application. The default number of requests to store is 10. But if you want to increase the number of requests and discard the older request and display a recent request then you need to set the property in the web.config file.

   <trace enabled="true"/>

Question 18: What are the data controls available in ASP.NET?

Answer: The Controls having DataSource Property are called Data Controls in ASP.NET. ASP.NET allows powerful feature of data binding, you can bind any server control to simple properties, collections, expressions and/or methods. When you use data binding, you have more flexibility when you use data from a database or other means. Data Bind controls are container controls.

Controls -> Child Control

Data Binding is binding controls to data from databases. With data binding we can bind a control to a particular column in a table from the database or we can bind the whole table to the data grid. Data binding provides simple, convenient, and powerful way to create a read/write link between the controls on a form and the data in their application.

Data binding allows you to take the results of properties, collection, method calls, and database queries and integrate them with your ASP.NET code. You can combine data binding with Web control rendering to relieve much of the programming burden surrounding Web control creation. You can also use data binding with ADO.NET and Web controls to populate control contents from SQL select statements or stored procedures.

Data binding uses a special syntax:

<%# %>

The *<%#,* which instructs ASP.NET to evaluate the expression. The difference between a data binding tags and a regular code insertion tags <% and %> becomes apparent when the expression is evaluated. Expressions within the data binding tags are evaluated only when the DataBind method in the Page objects or Web control is called.

Data Bind Control can display data in connected and disconnected model.

Following are data bind controls in ASP.NET:

- Repeater Control
- DataGrid Control
- DataList Control
- GridView Control
- DetailsView
- FormView
- DropDownList
- ListBox
- RadioButtonList
- CheckBoxList
- BulletList etc.

Question 19: What are the major events in global.aspx?

Answer: The Global.asax file, which is derived from the HttpApplication class, maintains a pool of HttpApplication objects, and assigns them to applications as needed. The Global.asax file contains the following events:

- *Application_Init*
- *Application_Disposed*
- *Application_Error*
- *Application_Start*
- *Application_End*
- *Application_BeginReques*

Question 20: Use of CheckBox in .NET?

Answer: The CheckBox control is a very common control of HTML, unlike radio buttons it can select multiple items on a webpage. The CheckBox control in ASP.NET has many properties and some of them are listed below.

Property	Description
AutoPostBack	Specifies whether the form should be posted immediately after the Checked property has changed or not. The default is false.
CausesValidation	Specifies if a page is validated when a Button control is clicked.
Checked	Specifies whether the check box is checked or not.
InputAttributes	Attribute names and values used for the Input element for the CheckBox control.
LabelAttributes	Attribute names and values used for the Label element for the CheckBox control.
runat	Specifies that the control is a server control. Must be set to "server".
Text	The text next to the check box.
TextAlign	On which side of the check box the text should appear (right or left).
ValidationGroup	Group of controls for which the Checkbox control causes validation when it posts back to the server.
OnCheckedChanged	The name of the function to be executed when the Checked property has changed.

Question 21: What is the authentication and authorization in ASP.NET?

Answer:

- **Authentication:** Prove genuineness
- **Authorization:** process of granting approval or permission on resources.

In ASP.NET authentication means to identify the user or in other words its nothing but to validate that he exists in your database and he is the proper user.

Authorization means does he have access to a particular resource on the IIS website. A resource can be an ASP.NET web page, media files (MP4, GIF, JPEG etc), compressed file (ZIP, RAR) etc.
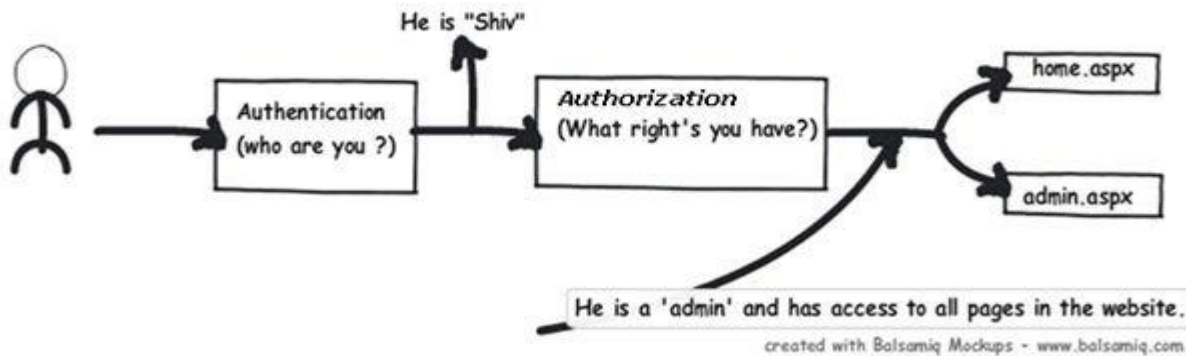
Types of authentication and authorization in ASP.NET: There are three ways of doing authentication and authorization in ASP.NET:

- **Windows authentication:** In this methodology ASP.NET web pages will use local windows users and groups to authenticate and authorize resources.

- **Forms Authentication:** This is a cookie based authentication where username and password are stored on client machines as cookie files or they are sent through URL for every request. Form-based authentication presents the user with an HTML-based Web page that prompts the user for credentials.

- **Passport authentication:** Passport authentication is based on the passport website provided by the Microsoft .So when user logins with credentials it will be reached to the passport website (i.e. hotmail,devhood,windows live etc) where authentication will happen. If Authentication is successful it will return a token to your website.

- **Anonymous access:** If you do not want any kind of authentication then you will go for Anonymous access.

In 'web.config' file set the authentication mode to 'Windows' as shown in the below code snippets.

1. <authentication mode="Windows"/>

We also need to ensure that all users are denied except authorized users. The below code snippet inside the authorization tag that all users are denied. '?' indicates any unknown user.

1. <authorization>
2.   <deny users="?"/>
3. </authorization>

Question 22: What are the HTML server controls in ASP.NET?

Answer: The Microsoft.NET Framework provides a rich set of server-side controls for developing Web applications. You can add these controls to WebForms pages just as you add Windows controls to a form. Server-side controls are often called server controls or Web Forms controls. There are four types of Server controls: HTML server controls. Web server controls, validation control, and user controls.

HTML Server controls-

HTML developers must be familiar with old HTML controls, which they use to write GUI applications in HTML. These controls are the same HTML controls; you can run these controls on the server by defining the runat ="server" attribute. These control names start with Html.

Controls	Description
HtmlForm	Create an HTML form control, used as a place holder of other controls.
HtmlInputText	Creates an input text box control used to get input from user.
HtmltextArea	Creates multiline text box control.
HtmlAnchor	Creates a Web navigation.
HtmlButton	Creates a button control.
HtmlImage	Creates an image control, which is used to display an image.
HtmlInputCheckBox	Creates a check box control.
HtmlInputRadioButton	Creates a radio button control.
HtmlTable	Creates a table control.
HtmlTableRow	Creates a row within a table.
HtmlTableCell	Creates a cell with in a row.

- o Web Server Controls
- o Validation Controls
- o User Controls

Question 23: What are the authentication modes in ASP.NET for security?

Answer: When you begin a program for a customer using ASP.NET, you should consider about security. Security is one of the most important components of any application. Security is even more important when you are making a web application which is exposed to million of users. ASP.NET provides classes and methods that ensure that the application is secure from outside attacks. In this article we will investigate the different types of authentication provided by ASP.NET. In web.config file you can set authentication mode value 'windows' or 'forms'. What's about difference and how to you use them? (Authentication have some other values to, this article does not consider them.).

How to use mode "Windows"?

Windows Authentication mode provides the developer to authenticate a user based on Windows user accounts. This is the default authentication mode provider by ASP.NET. This will return the computer name along with the user name.

1. <authentication mode="Windows">
2.   <forms name=" AuthenticationDemo" loginUrl="logon.aspx" protection="All" path="/"t imeout="30" />
3. </authentication>

How to use mode "Forms"?

Insert the *<Forms>* tag, and fill the appropriate attributes.

1. <authentication mode="Forms">
2.   <forms name=" AuthenticationDemo" loginUrl="logon.aspx" protection="All" path="/"t imeout="30" />
3. </authentication>

Question 24: What is the web API in ASP.NET?

Answer: It is a framework provided by Microsoft for writing HTTP services. There are many frameworks available to build HTTP based services. They follow a common guideline of international standardization but with different flavors.

For example, all frameworks must adhere to these status codes-

- **1xx** - Informational Message
- **2xx** - Successful
- **3xx** - Redirection
- **4xx** - Client Error
- **5xx** - Server Error

Features:

- It is light weight and thus good for small devices also like tablets, smart phones.
- No tedious & extensive configuration like WCF REST is required.
- MediaTypeFormatter makes easy to configure your APIs response type in single line (JSON, XML and so on).
- IIS Hosting dependency is no more and it can be hosted in application too.
- Easy and simple control with HTTP features such as Caching, Versioning, request/response headers and its various content formats.
- It support content-negotiation (deciding the best response data format that client can accept).

Question 25: Describe application state management in ASP.NET?

Answer: Application Level State Management is used to maintain the state of all the users accessing the web forms present within the website. The value assigned for an application is considered as an object. Application object will not have any default expiration period.

Whenever the webserver has been restarted or stopped then the information maintained by the application object will be lost.

If any data is stored on the application object then that information will be shared upon all the users accessing the webserver.

Since the information is shared among all the users, it is advisable to lock and unlock the application object as per requirement.

Global Application Class(Global.asax):

It is a Class which consists of event handlers which executes the code implicitly whenever a relevant task has been performed on the web server.**Design**:

```
1.   <%@ Application Language="C#" %>
2.    <script runat="server">
3.    void Application_Start(object sender, EventArgs e)
4.    {
5.       // Code that runs on application startup
6.    }
7.    void Application_End(object sender, EventArgs e)
8.    {
9.       // Code that runs on application shutdown
10.   }
11.   void Application_Error(object sender, EventArgs e)
12.   {
13.      // Code that runs when an unhandled error occurs
14.   }
15.   void Session_Start(object sender, EventArgs e)
16.   {
17.      // Code that runs when a new session is started
18.   }
19.   void Session_End(object sender, EventArgs e)
20.   {
21.      // Code that runs when a session ends.
22.   }
23.   </script>
```

Question 26: What is the code behind and Inline Code?

Answer: Code Behind-

Code Behind refers to the code for an ASP.NET Web page that is written in a separate class file that can have the extension of .aspx.cs or .aspx.vb depending on the language used. Here the code is compiled into a separate class from which the .aspx file derives. You can write the code in a separate .cs or .vb code file for each .aspx page. One major point of Code Behind is that the code for all the Web pages is compiled into a DLL file that allows the web pages to be hosted free from any Inline Server Code.

Inline Code-

Inline Code refers to the code that is written inside an ASP.NET Web Page that has an extension of .aspx. It allows the code to be written along with the HTML source code using a <Script> tag. Its major point is that since it's physically in the .aspx file it's deployed with the Web Form page whenever the Web Page is deployed.

Question 27: What is the ASP.NET page life Cycle?

Answer: When a page is requested by the user from the browser, the request goes through a series of steps and many things happen in the background to produce the output or send the response back to the client. The periods between the request and response of a page is called the "Page Life Cycle".

- **Request:** Start of the life cycle (sent by the user).
- **Response:** End of the life cycle (sent by the server).

There are four stages that occur during the Page Life Cycle before the HTML Response is returned to the client. Later in this article we"ll study all these stages and their sub events.

1. Initialization
2. Loading
3. Rendering
4. Unloading

Initialization	During this stage the IsPostback property is set. The page determines whether the request is a Postback (old request) or if this is the first time the page is being processed (new request). Controls on the page are available and each control's UniqueID property is set. Now if the current request is a postback then the data has not been loaded and the value of the controls have not yet been restored from the view state.
Loading	At this stage if the request is a Postback then it loads the data from the view state.
Rendering	Before rendering, the View State is saved for the page and its controls. During this phase, the page calls the render method for each control, providing a text writer that writes its output to the OutputStream of the page's Response property.
Unloading	Unload is called after the page has been fully rendered, sent to the client and is ready to be discarded. At this point also the page properties such as Response and Request are unloaded.

Question 28: What are the ASP.NET page life cycle events?

Answer: We have many events in ASP.NET page life cycle let's see some most important events:

- **Page request** - When ASP.NET gets a page request, it decides whether to parse and compile the page or there would be a cached version of the page; accordingly the response is sent,

- **Starting of page life cycle** - At this stage, the Request and Response objects are set. If the request is an old request or post back, the IsPostBack property of the page is set to true. The UICulture property of the page is also set.

- **Page initialization -** At this stage, the controls on the page are assigned unique ID by setting the UniqueID property and themes are applied. For a new request postback data is loaded and the control properties are restored to the view-state values.

- **Page load** - At this stage, control properties are set using the view state and control state values.

- **Validation** - Validate method of the validation control is called and if it runs successfully, the IsValid property of the page is set to true.

- **Postback event handling** - If the request is a postback (old request), the related event handler is called.

- **Page rendering** - At this stage, view state for the page and all controls are saved. The page calls the Render method for each control and the output of rendering is written to the OutputStream class of the Page's Response property.

- **Unload -** The rendered page is sent to the client and page properties, such as Response and Request are unloaded and all cleanup done.

ASP.NET Page Life Cycle Events: Following are the page life cycle events:

- **PreInit** - PreInit is the first event in page life cycle. It checks the IsPostBack property and determines whether the page is a postback. It sets the themes and master pages, creates dynamic controls and gets and sets profile property values. This event can be handled by overloading the OnPreInit method or creating a Page_PreInit handler.

- **Init -** Init event initializes the control property and the control tree is built. This event can be handled by overloading the OnInit method or creating a Page_Init handler.

- **InitComplete -** InitComplete event allows tracking of view state. All the controls turn on view-state tracking.

- **LoadViewState -** LoadViewState event allows loading view state information into the controls.

- **LoadPostData -** During this phase, the contents of all the input fields defined with the <form> tag are processed.

- **PreLoad -** PreLoad occurs before the post back data is loaded in the controls. This event can be handled by overloading the OnPreLoad method or creating a Page_PreLoad handler.

- **Load -** The Load event is raised for the page first and then recursively for all child controls. The controls in the control tree are created. This event can be handled by overloading the OnLoad method or creating a Page_Load handler.

- **LoadComplete -** The loading process is completed, control event handlers are run and page validation takes place. This event can be handled by overloading the OnLoadComplete method or creating a Page_LoadComplete handler.

- **PreRender -** The PreRender event occurs just before the output is rendered. By handling this event, pages and controls can perform any updates before the output is rendered.

- **PreRenderComplete -** As the PreRender event is recursively fired for all child controls, this event ensures the completion of the pre-rendering phase.

- **SaveStateComplete** - State of control on the page is saved. Personalization, control state and view state information is saved. The HTML markup is generated. This stage can be handled by overriding the Render method or creating a Page_Render handler.

- **UnLoad** - The UnLoad phase is the last phase of the page life cycle. It raises the UnLoad event for all controls recursively and lastly for the page itself. Final cleanup is done and all resources and references, such as database connections, are freed. This event can be handled by modifying the OnUnLoad method or creating a Page_UnLoad handler.

Question 29: Describe login Controls in ASP?

Answer: The Login control provides the user interface to log a user into a web site. The Login control uses the Membership service to authenticate the user in your membership system. The default Membership service from your configuration file will be used automatically, however you can also set the Membership provider that you would like used as a property on the control.

The Login Control consists of:

- **Username Label and Textbox:** Collects the string used to identify the user in the membership system.
- **Password Label and Textbox:** Collects the password for the specified user. The textbox text is always obscured.
- **LoginButton:** The button to submit the users request for authentication.
- **RememberMe:** Configurable to display a checkbox giving the user the option to store a persistent cookie on the user's machine.
- **Title and Instruction:** Text to orient and guide the user through the process.
- **Links:** Configurable links to help, password recovery and user registration information.
- **Validators:** Required field Validators for the username and password textboxes.

For Example:

```
<asp:Login ID="Login1" runat="server" BackColor="#FFE0C0" BorderColor="Red" ></asp:Login>
```

Question 30: How to use repeater control in ASP.NET?

Answer: A Repeater is a Data-bound control. Data-bound controls are container controls. It creates a link between the Data Source and the presentation UI to display the data. The repeater control is used to display a repeated list of items.

The main use of Repeater Control is for displaying a repeated list of items bound to the control. A Repeater Control is faster and lightweight for displaying data compared to a GridView or DataGrid. With the Repeater control we can display data in a custom format. The main drawback of a Repeater Control is that it doesn't support paging and sorting.

The Repeater Control has the following types of template fields:

- Item Template
- AlternatingItem Template
- Header Template
- Footer Template
- Separator Template

Write connection code and select command in code bihaind file like:
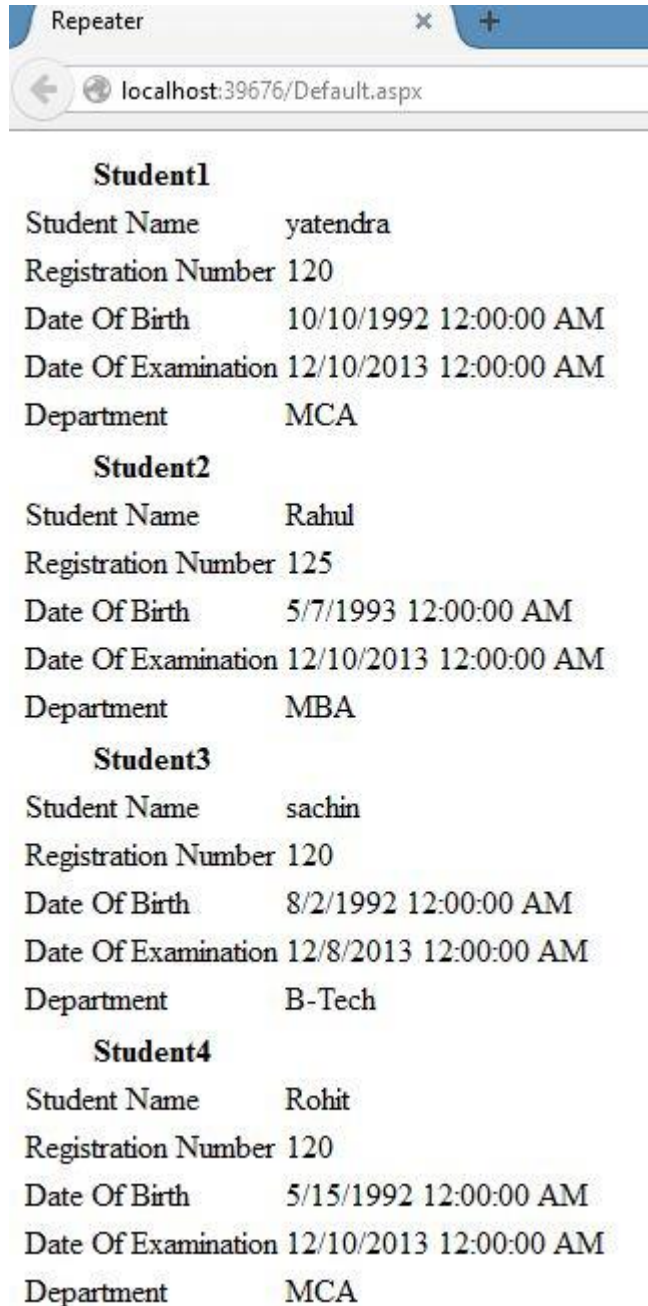
1. protected void Page_Load(object sender, EventArgs e)
2. {
3.   SqlConnection con = new SqlConnection("Data Source=MCNDESKTOP34;Initial Cata log=yatendra;Persist Security Info=True;User ID=sa;
4.   Password = Password$2 ");
5.   SqlDataAdapter sda = new SqlDataAdapter("select * from Student_Details1", con); DataTable dt = new DataTable(); sda.Fill(dt); Repeater1.DataSource = dt; Repeater1.Data Bind();
6. }

Now use Repeater control object in .aspx file like:

1. <asp:Repeater ID="Repeater1" runat="server">
2.   <ItemTemplate>
3.   <div>
4.   <table>
5.   <tr>

```
6.              <th>Student
7.                  <%#Eval("S_ID")%>
8.              </th>
9.          </tr>
10.         <tr>
11.           <td>Student Name</td>
12.           <td>
13.               <%#Eval("Student_Name")%>
14.           </td>
15.         </tr>
16.         <tr>
17.           <td>Registration Number</td>
18.           <td>
19.               <%#Eval("Register_No")%>
20.           </td>
21.         </tr>
22.         <tr>
23.           <td>Date Of Birth</td>
24.           <td>
25.               <%#Eval("D_O_B")%>
26.           </td>
27.         </tr>
28.         <tr>
29.           <td>Date Of Examination</td>
30.           <td>
31.               <%#Eval("D_O_E")%>
32.           </td>
33.         </tr>
34.         <tr>
35.           <td>Department</td>
36.           <td>
37.               <%#Eval("Department")%>
38.           </td>
39.         </tr>
40.         </table>
41.       </div>
42.    </ItemTemplate>
43. </asp:Repeater>
```

When you run this page so output will look like as:

Question 31: What are different methods of session maintenance in ASP.NET?

Answer: Session is a State Management Technique. A Session can store the value on the Server. It can support any type of object to be stored along with our own custom objects. A session is one of the best techniques for State Management because it stores the data as client-based, in other words the data is stored for every user separately and the data is secured also because it is on the server.

We can set the session on one of the following 2 types of configuration files:

1. **Machine Configuration file:** Machine Configuration is applied for all application.

2. **Application Configuration file:** It's applied for only application by application basis.
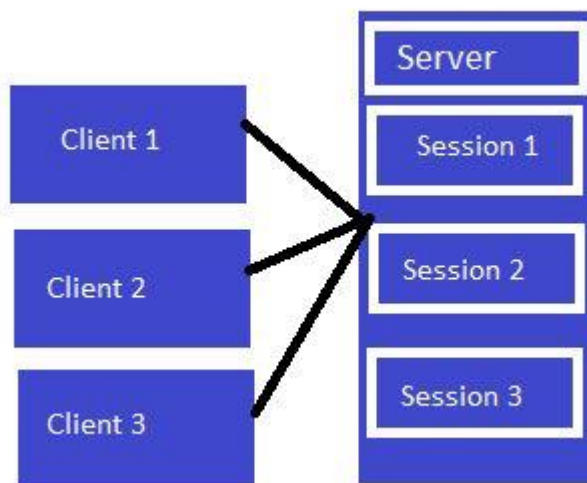
Session Mode:

In ASP.NET there are 4 types of Session Mode.

Off: We can disable the session mode for the entire application using the off mode.
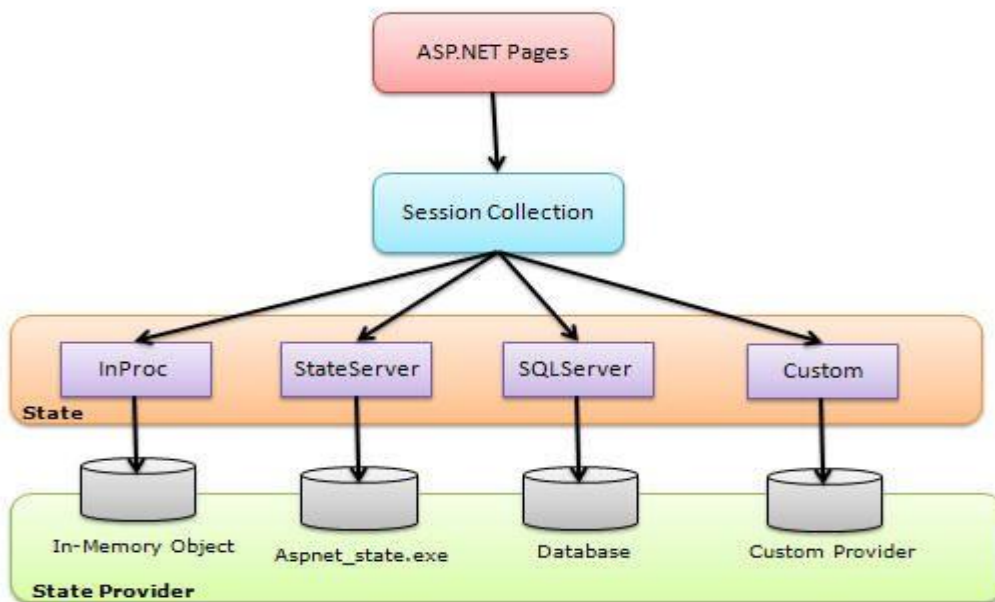
According to performance and durability the difference between InProc,State Server and SQL Server is:

Session mode	Performance Durability
InProc	More(1 processor and 1 server) less.
State Server	Medium(n processor and 1 server) Medium
SQL Server	Less More

Question 32: What is the Difference between session and caching?

Answer: The first main difference between session and caching is: a session is per-user based but caching is not per-user based, so what does that mean? Session data is stored at the user level but caching data is stored at the application level and shared by all the users. It means that it is simply session data that will be different for the various users for all the various users, session memory will be allocated differently on the server but for the caching only one memory will be allocated on the server and if one user modifies the data of the cache for all, the user data will be modified.

Question 33: What is the difference between HttpContext.Current.Items and HttpContext.Current.Session in ASP.NET?

Answer: Session state is one of the popular state management techniques in ASP.NET environment. We developer people play with session storage every now and then. It's pretty simple to manage session if you understand the basic concept. Here is the syntax to do that.

1. Session[index] = "Value";
2. Let's a have an example: using System;
3. using System.Collections.Generic;
4. using System.Linq;
5. using System.Web;
6. using System.Web.UI;
7. using System.Web.UI.WebControls;
8. namespace WebApp
9. {
10. public partial class WebForm1: System.Web.UI.Page
11. {
12. protected void Page_Load(object sender, EventArgs e)
13. {
14. if(!IsPostBack)
15. {
16. HttpContext.Current.Items["Value"] = "Sourav Kayal in ITEM";
17. HttpContext.Current.Session["Value"] = "Sourav Kayal in SESSION";
18. Response.Write((string)(HttpContext.Current.Items["Value"]) + "<br>");
19. Response.Write((string)(HttpContext.Current.Session["Value"])); } }
20. protected void Button1_Click(object sender, EventArgs e)
21. {
22. Response.Write((string)(HttpContext.Current.Items["Value"]) + "<br>");
23. Response.Write((string)(HttpContext.Current.Session["Value"]));
24. } }
25. }

Question 34: What is the difference between Server.Transfer and Response.redirect?

Answer: Both Response.Redirect and Server.Transfer methods are used to transfer a user from one web page to another web page. Both methods are used for the same purpose but still there are some differences as follows.

The Response.Redirect method redirects a request to a new URL and specifies the new URL while the Server.Transfer method for the current request, terminates execution of the current page and starts execution of a new page using the specified URL path of the page.

Both Response.Redirect and Server.Transfer has same syntax like:

1. Response.Redirect("UserDetail.aspx");
2. Server.Transfer("UserDetail.aspx");

Before touching on more points I want to explain some HTTP status codes, these are important for the understanding of the basic differences between these two. The HTTP status codes are the codes that the Web server uses to communicate with the Web browser or user agent.
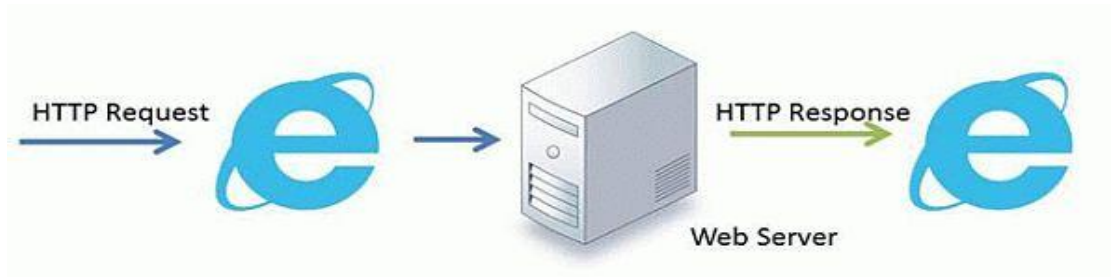
Question 35: What is page directives in ASP.NET?

Answer: Basically Page Directives are commands. These commands are used by the compiler when the page is compiled.

How to use the directives in an ASP.NET page:

It is not difficult to add a directive to an ASP.NET page. It is simple to add directives to an ASP.NET page. You can write directives in the following format:

<%@[Directive][Attributes]%>

See the directive format, it starts with "<%@" and ends with "%>". The best way is to put the directive at the top of your page. But you can put a directive anywhere in a page. One more thing, you can put more than one attribute in a single directive.

Here is the full list of directives:

- @Page
- @Master
- @Control
- @Import
- @Implements
- @Register
- @Assembly
- @MasterType
- @Output Cache
- @PreviousPageType
- @Reference

Question 36: What is HTTP Handler?

Answer: Every request into an ASP.NET application is handled by a specialized component known as an HTTP handler. The HTTP handler is the most important ingredient while handling ASP.NET requests.

Examples: ASP.NET uses different HTTP handlers to serve different file types. For example, the handler for web Page creates the page and control objects, runs your code, and renders the final HTML.

ASP.NET default handlers:

1. Page Handler (.aspx) - Handles Web pages.
2. User Control Handler (.ascx) - Handles Web user control pages.
3. Web Service Handler (.asmx) - Handles Web service pages.
4. Trace Handler (trace.axd) - Handles trace functionality.

Why we need to create our own HTTP Handler: Sometime we need to avoid ASP.NET full page processing model, which saves lot of overheads, as ASP.NET web form model has to go through many steps such as creating web page objects, persisting view state etc. What we are interested into is to develop some low level interface that provides access to objects like Request and Response but doesn't use the full control based web form model discussed above.

Examples:

1. Dynamic image creator - Use the System.Drawing classes to draw and size your own images.
2. RSS - Create a handler that responds with RSS-formatted XML. This would allow you to add RSS feed capabilities to your sites.
3. Render a custom image,
4. Perform an ad hoc database query,
5. Return some binary data.

All HTTP handlers are defined in the <httpHandlers> section of a configuration file which is nested in the <system.web> element.

1. <httpHandlers>

2. <add verb="*" path="trace.axd" validate="true" type="System.Web.Handlers.TraceHandler" />

3. <add verb="*" path="*.config" validate="true" type="System.Web.HttpForbiddenHandler" />

4. <add verb="*" path="*.cs" validate="true" type="System.Web.HttpForbiddenHandler" />

5. <add verb="*" path="*.aspx" validate="true" type="System.Web.UI.PageHandlerFactory" /> </httpHandlers>

Question 37: What are Differences between ASP.NET HttpHandler and HttpModule?

Answer: The user requests for a resource on web server. The web server examines the file name extension of the requested file, and determines which ISAPI extension should handle the request. Then the request is passed to the appropriate ISAPI extension. For example when an .aspx page is requested it is passed to ASP.NET page handler. Then Application domain is created and after that different ASP.NET objects like Httpcontext, HttpRequest, HttpResponse are created. Then instance of HttpApplication is created and also instance of any configured modules. One can register different events of HttpApplication class like BeginRequest, AuthenticateRequest, AuthorizeRequest, ProcessRequest etc.

HTTP Handler:

HTTP Handler is the process which runs in response to a HTTP request. So whenever user requests a file it is processed by the handler based on the extension. So, custom http handlers are created when you need to special handling based on the file name extension. Let's consider an example to create RSS for a site. So, create a handler that generates RSS-formatted XML. Now bind the .rss extension to the custom handler.

HTTP Modules:

HTTP Modules are plugged into the life cycle of a request. So when a request is processed it is passed through all the modules in the pipeline of the request. So generally http modules are used for:

- **Security:** For authenticating a request before the request is handled.
- **Statistics and Logging:** Since modules are called for every request they can be used for gathering statistics and for logging information.
- **Custom header:** Since response can be modified, one can add custom header information to the response.

Question 38: Explain the AdRotator Control?

Answer: AdRotator control is used to create dynamic ads. The AdRotator Control presents ad images each time a user enters or refreshes a webpage. When the ads are clicked, it will navigate to a new Web location. The AdRotator control is used to display a sequence of ad images.The AdRotator control to work we need an Advertisement file (XML file) and some sample images.

Adding the AdRotator web server control to your web application. first, select the AdRotator and drag and drop the control to your web form. Map the XML file which contains the details about each and every ad.

The advertisement file is an XML file. The following are some of the elements of this XML file.

1. **<imageUrl>:** Optional. The path to the image file.
2. **<NavigateUrl>:** Optional. The URL to link to if the user clicks the ad.
3. **<AlternateText>:** Optional. An alternate text for the image.
4. **<Impressions>:** Optional. The display rates in percent of the hits.

XML code that has the details about the ads. The file Ads.xml looks like the code below:

```
1.  <Advertisements>
2.     <Ad>
3.        <ImageUrl>adimages/2.jpg</ImageUrl>
4.        <NavigateUrl>http://cat2.com</NavigateUrl>
5.        <AlternateText>Cat 2</AlternateText>
6.        <Impressions>30</Impressions>
7.     </Ad>
8.     <Ad>
9.        <ImageUrl>adimages/3.jpg</ImageUrl>
10.       <NavigateUrl>http://cat3.com</NavigateUrl>
11.       <AlternateText>Cat 3</AlternateText>
12.       <Impressions>20</Impressions>
13.    </Ad>
14.    <Ad>
15.       <ImageUrl>adimages/4.jpg</ImageUrl>
16.       <NavigateUrl>http://cat4.com</NavigateUrl>
17.       <AlternateText>Cat 4</AlternateText>
18.       <Impressions>10</Impressions>
19.    </Ad>
20. </Advertisements>
```

Question 39: What is cross-page posting in ASP.NET?

Answer: ASP.NET 1.1 provides for web forms posting back only to themselves. In many situations, the solution requires posting to a different web page. The traditional workaround alternatives were to use Response.Redirect and/or Server.Transfer to move to a different page and simulate cross page post-back behavior.

ASP.NET 2.0 provides a feature known as Cross Page PostBack for a web form to post-back to a different web form (other than itself).

How to post to a different page?

To set a web form to post back to a different web form, in the source web form, set the PostBackURL property of a control that implements IButtonControl (eg. Button, ImageButton, LinkButton) to the target web form. When the user clicks on this button control, the web form is cross-posted to the target web form. No other settings or code is required in the source web form.

Access source page info within the posted page: FindControl Method()

The target web form resulting from the cross-page postback provides a non-null PreviousPage property. This property represents the source page and provides reference to the source web form and its controls.

The controls on the source page can be accessed via the FindControl method on the object returned by the PreviousPage property of the target page.

```
1.  protected void Page_Load(object sender, EventArgs e)
2.  {
3.    ...
4.    TextBox txtStartDate = (TextBox) PreviousPage.FindControl("txtStartDate ");
5.    ...
6.  }
```

At this point the target page does not have any knowledge of the source page. The PreviousPage property is of the type Page. For accessing controls using FindControl, the developer has to presume a certain structure in the source web form. This approach using FindControl has a few limitations. FindControl is dependent on the developer to provide the ids of the controls to access. The code will stop working if the control id is changed in the source web form. The FindControl method can retrieve controls only within the current container. If you need to access a control within another control, you need to first get a reference to the parent control.

Access source page info within the posted page: @PreviousPageType Directive

There is another more direct option to get access to the source page controls if the source page is pre-determined. The @PreviousPageType directive can be used in the target page to strongly type the source page. The directive specifies the source page using either the VirtualPath attribute or the TypeName attribute. The PreviousPage property then returns a strongly typed reference to the source page. It allows access to the public properties of the source page.

SourcePage.aspx:

1. <form runat="server"> ...
2.   <asp:textbox runat="server" id="txtFirstName" />
3.   <asp:textbox runat="server" id="txtLastName" />
4.   <asp:button runat="server" id="btnViewReport" Text="View Report" PostbackURL="~/targetpage.aspx" /> ... public string FirstName { get { return txtFirstName.Text; } } ...

TargetPage.aspx:

1. <%@ PreviousPageType VirtualPath="sourcepage.aspx" %>
2. string strFirstName;
3. strFirstName = PreviousPage.FirstName //Strongly Typed PreviousPage allows direct access to the public properties of the source page

Question 40: Explain GridView control in ASP.NET?

Answer: The GridView control displays the values of a data source in a table. Each column represents a field, while each row represents a record. The GridView control supports the following features:

- Binding to data source controls, such as SqlDataSource.
- Built-in sort capabilities.
- Built-in update and delete capabilities.
- Built-in paging capabilities.
- Built-in row selection capabilities.
- Programmatic access to the GridView object model to dynamically set properties, handle events, and so on.
- Multiple key fields.
- Multiple data fields for the hyperlink columns.
- Customizable appearance through themes and styles.

Creating a GridView:

1. <asp:GridView ID="gridService" runat="server">
2. </asp:GridView>

Question 41: What is the difference between ASP.NET Web API and WCF?

Answer: The ASP. NET Web API is a framework that uses the HTTP services and makes it easy to provide the response to the client request. The response depends on the request of the clients. The Web API builds the HTTP services, and handles the request using the HTTP protocols. The request may be GET, POST, DELETE, PUT. We can also say that the ASP. NET Web API:

- Is an HTTP service.

- Is designed for reaching the broad range of clients.

- Uses the HTTP application.

We use the ASP. NET Web API for creating the REST ful (Representational State Transfer) services.

The following are some important points of the ASP. NET Web API:

- The ASP. NET Web API supports the MVC application features that are controller, media formatters, routing etcetera.

- It is a platform for creating the REST services.

- It is a framework for creating the HTTP services.

- Responses can be formatted by the APIs MediaTypeFormatter into the Java Script Object Notation (JSON) and Extencible Markup Language (XML) formats.

Question 42: What is the PostBack property in ASP.NET?

Answer: If we create a web Page, which consists of one or more Web Controls that are configured to use AutoPostBack (Every Web controls will have their own AutoPostBack property), the ASP.NET adds a special JavaScipt function to the rendered HTML Page. This function is named _doPostBack() . When Called, it triggers a PostBack, sending data back to the web Server.

ASP.NET also adds two additional hidden input fields that are used to pass information back to the server. This information consists of ID of the Control that raised the event and any additional information if needed. These fields will empty initially as shown below,

1. <input type="hidden" name="__EVENTTARGET" id="__EVENTTARGET" value="" />

2. <input type="hidden" name="__EVENTARGUMENT" id="__EVENTARGUMENT" value="" />

The following actions will be taken place when a user changes a control that has the AutoPostBack property set to true:

1. On the client side, the JavaScript _doPostBack function is invoked, and the page is resubmitted to the server.
2. ASP.NET re-creates the Page object using the .aspx file.
3. ASP.NET retrieves state information from the hidden view state field and updates the controls accordingly.
4. The Page.Load event is fired.
5. The appropriate change event is fired for the control. (If more than one control has been changed, the order of change events is undetermined.)
6. The Page.PreRender event fires, and the page is rendered (transformed from a set of objects to an HTML page).
7. Finally, the Page.Unload event is fired.
8. The new page is sent to the client.

Question 43: Explain Cookie-less Session in ASP.NET.

Answer: By default a session uses a cookie in the background. To enable a cookie-less session, we need to change some configuration in the Web.Config file. Follow these steps:

1. Open Web.Config file.
2. Add a <sessionState> tag under <system.web> tag.
3. Add an attribute "cookieless" in the <sessionState> tag and set its value to "AutoDetect" like below:

   <sessionState cookieless="AutoDetect" regenerateExpiredSessionId="true"/>

The possible values for "**cookieless**" attribute are:

- **AutoDetect:** Session uses background cookie if cookies are enabled. If cookies are disabled, then the URL is used to store session information.

- **UseCookie:** Session always use background cookie. This is default.

- **UseDeviceProfile:** Session uses background cookie if browser supports cookies else URL is used.

- **UseUri:** Session always use URL.

"**RegenerateExpiredSessionId**" is used to ensure that if a cookieless url is expired a new new url is created with a new session. And if the same cookieless url is being used by multiple users an the same time, they all get a new regenerated session url.

Question 44: What is Themes in ASP.NET?

Answer: A theme decides the look and feel of the website. It is a collection of files that define the looks of a page. It can include skin files, CSS files & images.

We define themes in a special App_Themes folder. Inside this folder is one or more subfolders named Theme1, Theme2 etc. that define the actual themes. The theme property is applied late in the page's life cycle, effectively overriding any customization you may have for individual controls on your page.

How to apply themes?

There are 3 different options to apply themes to our website:

1. Setting the theme at the page level: the Theme attribute is added to the page directive of the page.

   <%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"Inherits="Default" Theme="Theme1"%>

2. Setting the theme at the site level: to set the theme for the entire website you can set the theme in the web.config of the website. Open the web.config file and locate the <pages> element and add the theme attribute to it:

   1. **<pages theme="Theme1">**
   2. **....**
   3. **....**
   4. **</pages>**

3. Setting the theme programmatically at runtime: here the theme is set at runtime through coding. It should be applied earlier in the page's life cycle ie. Page_PreInit event should be handled for setting the theme. The better option is to apply this to the Base page class of the site as every page in the site inherits from this class.

   Page.Theme = Theme1;

Uses of Themes:

1. Since themes can contain CSS files, images and skins, you can change colors, fonts, positioning and images simply by applying the desired themes.

2. You can have as many themes as you want and you can switch between them by setting a single attribute in the web.config file or an individual aspx page. Also you can switch between themes programmatically.

3. Setting the themes programmatically, you are offering your users a quick and easy way to change the page to their likings.

4. Themes allow you to improve the usability of your site by giving users with vision problems the option to select a high contrast theme with a large font size.

Question 45: What are the Navigations techniques in ASP.NET?

Answer: Navigation can cause data loss if it not properly handled. We do have many techniques to transfer data from one page to another but every technique has its own importance and benefits.

We will discuss the following techniques in this article.

- Response.Redirect
- Server.Transfer
- Server.Exceute
- Cross page posting

Question 46: What is WebParts in ASP.NET?

Answer: ASP.NET 2.0 incorporates the concept of **WEB PARTS** in itself and we can code and explore that as easily as we have done with the other controls in the previous sessions.

We can compose web parts pages from "web parts", which can be web controls, user controls.

Component of Web Parts:

The web parts consist of different components like:

- Web Part Manager
- Web Part Zone
- CatLog Part
- CatLog Zone
- Connections Zone
- Editor Part
- Editor Zone

Web Part Zone:

- Web Part Zone can contain one or more Web Part controls.
- This provides the layout for the Controls it contains. A single ASPX page can contain one or more Web Part Zones.
- A Web Part Control can be any of the controls in the toolbox or even the customized user controls.

Question 47: What are master pages?

Answer: Some points about Master Pages:

1. The extension of MasterPage is '.master'.

2. MasterPage cannot be directly accessed from the client because it just acts as a template for the other Content Pages.

3. In a MasterPage we can have content either inside ContentPlaceHolder or outside it. Only content inside the ContentPlaceHolder can be customized in the Content Page.

4. We can have multiple masters in one web application.

5. A MasterPage can have another MasterPage as Master to it.

6. The content page content can be placed only inside the content tag.

7. Controls of MasterPage can be programmed in the MasterPage and content page but a content page control will never be programmed in MasterPage.

8. A master page of one web application cannot be used in another web application.

9. The **MasterPageFile** property of a webform can be set dynamically and it should be done either in or before the Page_PreInit event of the WebForm. **Page.MasterPageFile = "MasterPage.master"**. The dynamically set Master Page must have the ContentPlaceHolder whose content has been customized in the WebForm.

10. The order in which events are raised: Load (Page) a Load (Master) a LoadComplete (Page) i.e. if we want to overwrite something already done in Load event handler of Master then it should be coded in the **LoadComplete** event of the page.

11. Page_Load is the name of method for event handler for Load event of Master. (it's not **Master_Load**).

Question 48: What is Data Cache in ASP.NET and how to use?

Answer: Data Cache is used to store frequently used data in the Cache memory. It's much efficient to retrieve data from the data cache instead of database or other sources. We need to use System.Web.Caching namespace. The scope of the data caching is within the application domain unlike "session". Every user is able to access this object.

When client request to the server, server execute the stored procedure or function or select statements on the Sql Server database then it returns the response to the browser. If we run again same process will happen on the web server with sql server.

How to create data cache?

Cache ["Employee"] = "DataSet Name"

We can create data caching use Cache Keyword. It's located in the System.Web.Caching namespace. It's just like assigning value to the variable.

How to remove a Data Cache?

We can remove Data Cache manually.

1. //We need to specify the cache name
2. Cache.Remove(String key);

Question 49: Enterprise Library in ASP.NET?

Answer: Enterprise Library: It is a collection of application blocks and core infrastructure. Enterprise library is the reusable software component designed for assisting the software developers.

We use the Enterprise Library when we want to build application blocks intended for the use of developers who create complex enterprise level application.

Enterprise Library Application Blocks:

1. **Security Application Block** - Security Application Block provide developers to incorporate security functionality in the application. This application can use various blocks such as authenticating and authorizing users against the database.

2. **Exception Handling Application Block -** This block provides the developers to create consistency for processing the error that occur throughout the layers of Enterprise Application.

3. **Cryptography Application Block -** Cryptography application blocks provides developers to add encryption and hashing functionality in the applications.

4. **Caching Application Block -** Caching Application Block allows developers to incorporate local cache in the applications.

Question 50: How can we improve the Performance of an ASP.NET Web Page?

Answer: This is the most common question from ASP.NET forum to any interview. In this post I'm going to point out some of the important points that may help to improve the performance.

Here I used the word "improve performance" in the sense to decrease the loading time of the page. There are various reasons behind. Some of them we look into from the "backend side" (Database side) and rest of them we need to take care in "front-end" ((UI) side.

For illustrative purpose, you have an ASP.NET Web site, one of the aspx page take much time to load. Throughout this article, we are going to see how to decrease the loading time.

Back End (DB)

1. Try to check the Query performance that is how much time the query will take to execute and pull the records from DB. Then use SQL Server Profiler and Execution plan for that query so that you can come to a conclusion in which part it took much time.

2. Check in every table (who are all part of the query) Index is created properly.

3. If your query involves a complex stored procedure, which in turn use lot of joins, then you should focus on every table. In some cases, sub-query perform better than the joins.

4. If your web page involves paging concepts, try to move the paging concepts to SQL Server. I meant that based on the page count the SP will return the records, instead of bringing the records as a whole.

Chapter 5
WCF Interview Questions and Answers

Question 1: What is WCF?

Answer: WCF is a platform for building distributed businesses and deploying services among various endpoints in Windows. WCF was initially called "Indigo" and we can build service-oriented applications and provide interoperability.

WCF or Windows Communication Foundation is a programming model to create service oriented applications. It is used to create and deploy the service that is accessible to lots of different clients. It provides an environment where you can create a service which can be accessible to Windows clients as well as Linux clients or any others. It provides more features compared to web services.

WCF is a Microsoft technology to create service oriented application. Before the WCF, the Web Service was used to create services but that type of service is only accessible to Windows client hosted on HTTP protocol. But WCF services are accessible with different protocols like http, tcp, msmq, etc. A few sample scenarios include:
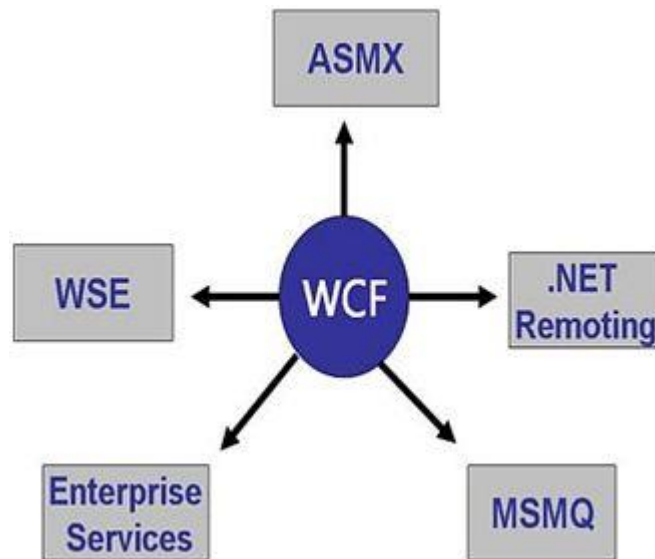
Question 2: Explain WCF Architecture and also explain its Fundamentals?

Answer: Windows Communication Foundation (WCF) is a framework for building service-oriented applications by which we can send asynchronous message/data from one service endpoint to another service endpoint.

WCF is a runtime and a set of APIs for creating systems that sends messages between services and clients. The same infrastructure and APIs are used to create applications that communicate with other applications on the same computer system or on a system that resides in another company and is accessed over the Internet.

The WCF fundamentals are as follows:

- Unification

    - COM+ Services
    - Web Services
    - .NET Remoting
    - Microsoft Message Queuing

- Interoperability
- Service Orientation

WCF Architecture

There is four major layers that provide developers with a new service-oriented programming model. The WCF architecture consists of the following layers.
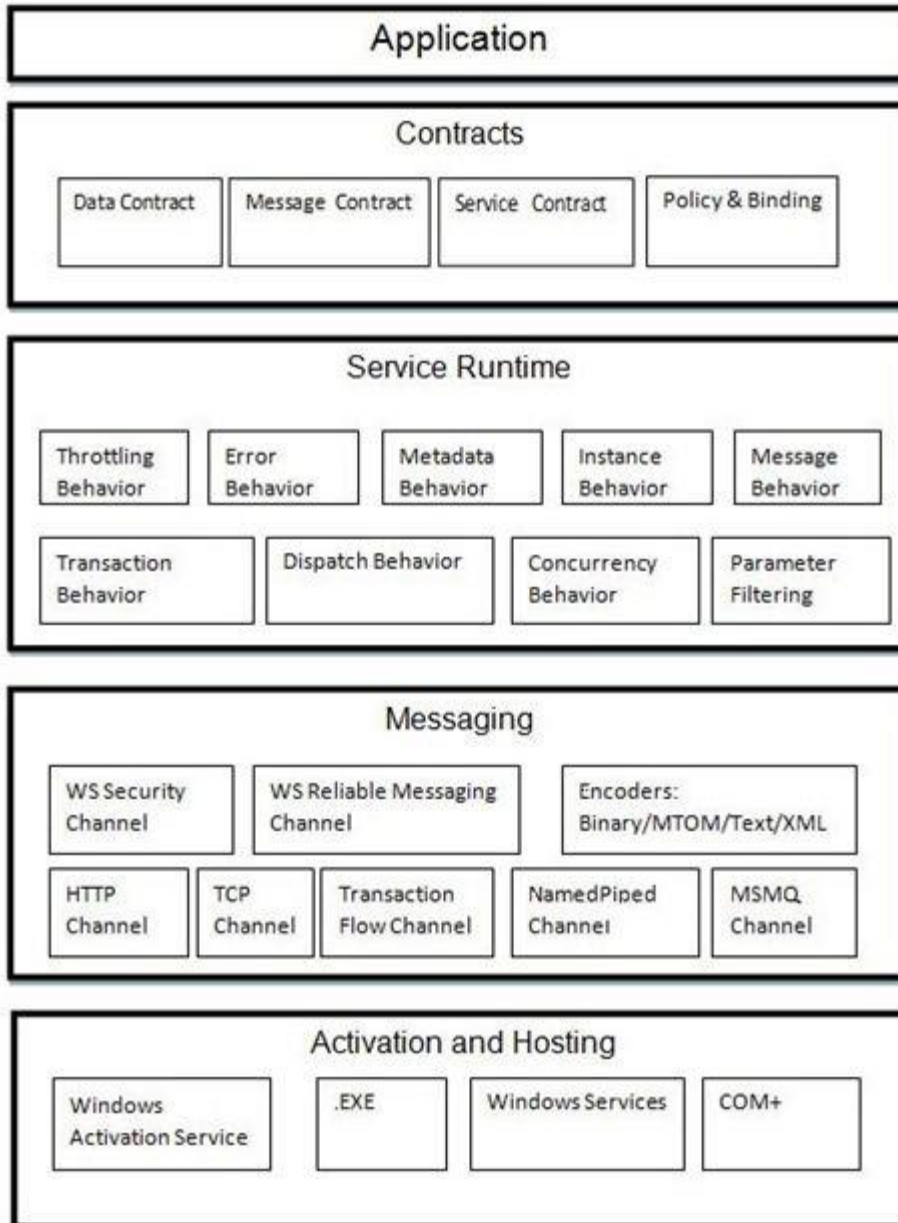
Question 3: Why Should We Use WCF Service?

Answer: Windows Communication Foundation (WCF) is a framework for building service-oriented applications. Most of WCF functionality is included in a single assembly called System.ServiceModel.dll, located in the System.ServiceModel namespace.
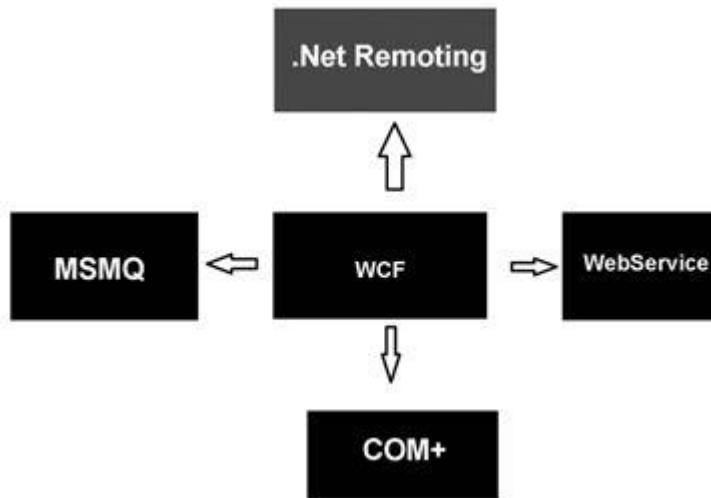
Why WCF?

1. A web service to exchange messages in XML format using HTTP protocol for interoperability.

2. A remoting service to exchange messages in binary format using TCP protocol for performance.

   - A secure service to process business transactions.

   - A service that supplies current data to others, such as a traffic report or other monitoring service.

   - A chat service that allows two people to communicate or exchange data in real time.

   - A dashboard application that polls one or more services for data and presents it in a logical presentation.

   - A Silverlight application to poll a service for the latest data feeds.

Question 4: What are the features and advantage of WCF?

Answer: Features of WCF-

Windows Communication Foundation (WCF) is a secure, reliable, and scalable messaging platform for the .NET Framework 3.0,

- Service Orientation
- Interoperability
- Multiple Message Patterns
- Service Metadata
- Data Contracts
- Security
- Multiple Transports and Encodings
- Reliable and Queued Messages
- Durable Messages
- Transactions
- AJAX and REST Support
- Extensibility

Advantages of WCF-

1. Service Oriented
2. Location Independent
3. Language Independent
4. Platform Independent
5. Support Multiple operation
6. WCF can maintain transaction like COM+ Does
7. It can maintain state
8. It can control concurrency
9. It can be hosted on IIS, WAS, Self hosting, Windows services.

It has AJAX Integration and JSON (JavaScript object notation) support.

- WCF can be configured to work independently of SOAP and use RSS instead.
- WCF is one of the fastest communication technologies and offers excellent performance compared to other Microsoft specifications.
- To improve communication, transmission speed needs to be optimized. This is done by transmitting binary-coded XML data instead of plain text to decrease latency.
- Object life-cycle management and distributed transaction management are applicable to any application developed using WCF.

Question 5: What is the difference between WCF and Web services?

Answer: WCF - Windows Communication Foundation (WCF) is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another.

Web Services: A Web Service is programmable application logic accessible via standard web protocols. One of these web protocols is the Simple Object Access Protocol (SOAP). SOAP is a W3C submitted note (as of May 2000) that uses standards based technologies (XML for data description and HTTP for transport) to encode and transmit application data.

Features	Web Service	WCF
Hosting	It can be hosted in IIS	It can be hosted in IIS, windows activation service, Self-hosting, Windows service
Programming	[WebService] attribute has to be added to the class	[ServiceContraact] attribute has to be added to the class
Model	[WebMethod] attribute represents the method exposed to client	[OperationContract] attribute represents the method exposed to client
Operation	One-way, Request- Response are the different operations supported in web service	One-Way, Request-Response, Duplex are different type of operations supported in WCF
XML	System.Xml.serialization name space is used for serialization	System.Runtime.Serialization namespace is used for serialization
Encoding	XML 1.0, MTOM(Message Transmission Optimization Mechanism), DIME, Custom	XML 1.0, MTOM, Binary, Custom
Transports	Can be accessed through HTTP, TCP, Custom	Can be accessed through HTTP, TCP, Named pipes, MSMQ,P2P, Custom
Protocols	Security	Security, Reliable messaging, Transactions

Question 6: Explain what is SOA? Are web-services SOA?

Answer: SOA stands for service oriented architecture. Service Oriented Architecture is an architectural approach in software development where the application is organized as "**Services**". Services are a group of methods that contain the business logic to connect a DB or other services. For instance you go to a hotel and order food. Your order first goes to the counter and then it goes to the kitchen where the food is prepared and finally the waiter serves the food.

Some important characteristics of Service Oriented Architecture

- SOA services should be **independent** of other services.

- Altering a service should not affect the client calling the service.

- Services should be **self-contained.**

- Services should be able to **define themselves** (in the Web Service Description Language (WSDL)). Services should be able to describe what they do. It should be able to tell the client what all of the operations it does, what are all the data types it uses and what kind of value it will return.

SOA Overview: A Service Oriented Architecture is based on four key abstractions.

- An Application Front End
- A Service
- A Service Repository
- A Service Bus

Application Front End: The application front end is decoupled from the services. Each service has a contract that defines what it will do and one or more interfaces to implement the contract.

A Service: It has the methods with the defined contracts and the implementation of the business logic to connect the DB or other service.

A Service Repository: The service repository provides a home for the services and the service bus provides an industry-standard mechanism for connecting to and interacting with the services.

Are web-services SOA?

SOA is thinking, it's an architectural concept and web service is one of the technical approach to complete it. Web services are the preferred standards to achieve SOA.

- In SOA we need the services to be loosely coupled. A web service communicates using SOAP protocol which is XML based which is very loosely coupled. SOA services should be able to describe themselves. WSDL describes how we can access the service.

- SOA services are located in a directory. UDDI describes where we can get the web service. This nothing but implementation of SOA registry.

Question 7: What is Service Contract in WCF?

Answer: Service contract means the collective mechanisms by which a service's capabilities and requirements are specified for its consumers. We must say that it defines the operations that a service will perform when executed. It tells more things about a service, like message data types, operation locations, the protocols the client will need in order to communicate with the service.

To create a service contract you define an interface with related methods representative of a collection of service operations, and then decorate the interface/class with the **ServiceContract** Attribute to indicate it is a service contract. Methods in the interface that should be included in the service contract are decorated with the **OperationContract** Attribute.

How to define a Service Contract:

```csharp
[ServiceContract (Namespace = "http://Rameshkartik/WCFSamples/OnlineShoppingService",
    Name = "OnlineShoppingService")]
public interface IOnlineShoppingService
{
    [OperationContract(Name = "StockAvailability")]
    bool IsStockAvailable(string sModelNumber);

    [OperationContract(Name = "CalculatePrice",
        ProtectionLevel = System.Net.Security.ProtectionLevel.None)]
    double CalculatePrice(string sModelNumber, int iQuantity,
        string sDeliveryLocation);



}
```

ServiceContractAttribute: It is used to declare the type as a Service Contract. It can be declared without any parameters but it can also take named parameters.

1. [ServiceContract(Name="MyService", Namespace="http://tempuri.org")]
2. public interface IMyService
3. {
4.   [OperationContract]
5.   int AddNum(string numdesc, string assignedTo);
6. }

OperationContractAttribute: It can only be applied on methods. It is used to declare methods which belong to a Service Contract. It controls the service description and message formats.

Question 8: What are Contracts in WCF? How many Contracts are in WPF?

Answer: The main use of contracts is to allow the client and services agree as to the types of operations and structures they will use during the communication process. It also shows the formal agreements between client and service to define a platform-neutral and standard for describing that what the service does.
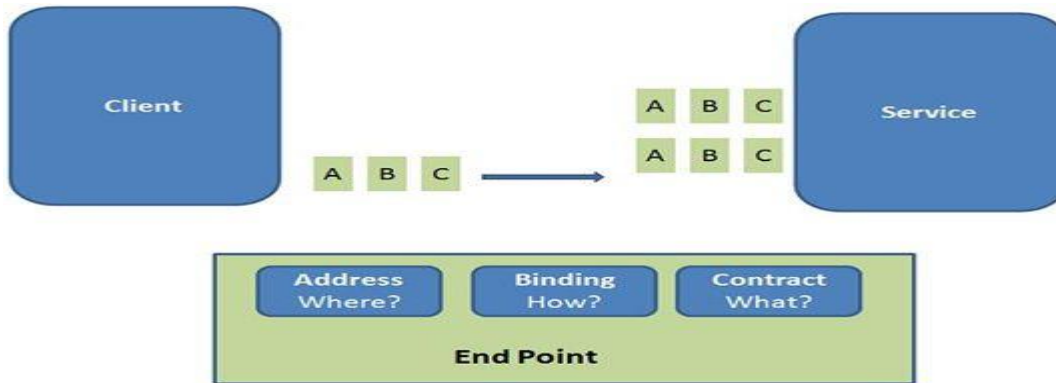
Service Contract: A service contract defines the operations which are exposed by the service to the outside world. A service contract is the interface of the WCF service and it tells the outside world what the service can do. It may have service-level settings, such as the name of the service and namespace for the service.

Operation Contract: An Operation Contract defines the method exposed to the client to exchange the information between the client and server. An Operation Contract describes what functionality is to be given to the client, such as addition, subtraction and so on.

It can be defined as in the following:

```
1.  public interface IService1
2.  {
3.      [OperationContract]
4.      string GetData(int value);
5.      [OperationContract]
6.      CompositeType GetDataUsingDataContract(CompositeType composite);
7.  }
```

Data Contract: Data Contracts define the data type for variables that are the same as get and set properties but the difference is that a Data Contract in WCF is used to serialize and deserialize the complex data. It defines how data types are serialized and deserialized. Using serialization, you can convert an object into a sequence of bytes that can be transmitted over a network. Using de-serialization, you reassemble an object from a sequence of bytes that you receive from a calling application.

Example:

```
1.  public class Student
2.  {
3.      private string _Name;
4.      private string _City;
5.      [DataMember]
6.      public string Name
7.      {
8.         get
9.         {
10.            return _Name;
11.        }
12.        set
13.        {
14.            _Name = value;
15.        }
16.    }
17. }
```

Message Contract: When an operation contract required passing a message as a parameter or returning value as a message, the type of this message will be defined as message contract. A message contract defines the elements of the message (like as Message Header, Message Body), as well as the message-related settings, such as the level of message security.

Message contracts give you complete control over the content of the SOAP header, as well as the structure of the SOAP body.

Fault Contract: A fault contract defines errors raised by the service, and how the service handles and propagates errors to its clients. An operation contract can have zero or more fault contracts associated with it.

The following is the syntax to raise the custom error in WCF:

```
1.  [ServiceContract]
2.  public interface IGetDetailsService
3.  {
4.      [OperationContract]
5.      [FaultContract(typeof (Student))]
6.      Student GetDetails(string Name);
7.  }
8.  [DataContract]
9.  public class Student
10. {
11.     private string _Name;
12.     private string _City;
13.     [DataMember]
14.     public string Name
15.     {
16.        get
17.        {
18.           return _Name;
19.        }
20.        set
21.        {
22.           _Name = value;
23.        }
24.     }
25.     [DataMember]
26.     public string City
27.     {
28.        get
29.        {
30.           return _City;
31.        }
32.        set
33.        {
34.           _City = value;
35.     } } }
```

Question 9: What is Data Contract in WCF?

Answer: A data contract is a formal agreement between a service and a client that abstractly describes the data to be exchanged.

Data contract can be explicit or implicit. Simple type such as int, string etc has an implicit data contract. User defined object are explicit or Complex type, for which you have to define a Data contract using [DataContract] and [DataMember] attribute.

A data contract can be defined as follows:

- It describes the external format of data passed to and from service operations.

- It defines the structure and types of data exchanged in service messages.

- It maps a CLR type to an XML Schema.

- It defines how data types are serialized and deserialized. Through serialization, you convert an object into a sequence of bytes that can be transmitted over a network. Through deserialization, you reassemble an object from a sequence of bytes that you receive from a calling application.

- It is a versioning system that allows you to manage changes to structured data.

Data Contract is not inheritable, so WCF requires every level in the class hierarchy explicitly opt in for the given Data Contract.

1. [DataContract]
2. public class CuboidInfo
3. {
4. }

Question 10: What is Message Contract in WCF?

Answer: A Message Contract is used to control the structure of a message body and serialization process. It is also used to send / access information in SOAP headers. By default WCF takes care of creating SOAP messages according to service DataContracts and OperationContracts.

A MessageContract controls the structure of a message body and serialization process. A MessageContract can be typed or untyped and are useful in interoperability cases and when there is an existing message format we must comply with.

The SOAP header is implemented in the namespace system.web.services.protocol.

1. [MessageContract]
2. public class AutherRequest
3. {
4.   public string AutherId;
5. }

Message Header: A Message Header is applied to a member of a message contract to declare the member within the message header.

1. [MessageContract]
2. public class AutherRequest
3. {
4.   [MessageHeader]
5.   public string AutherId;
6. }

Message Body Member: A Message Body Member is applied to the member of a message contract to declare the members within the message body.

1. [MessageContract]
2. public class AuthorResponse
3. {
4.   [MessageBodyMember]
5.   public Auther Obj;
6. }

Question 11: What is Fault Contract in WCF?

Answer: Fault Contract provides documented view for error accorded in the service to client. This help as to easy identity the what error has occurred, and where. By default when we throw any exception from service, it will not reach the client side. The less the client knows about what happened on the server side, the more dissociated the interaction will be, this phenomenon (not allowing the actual cause of error to reach client) is known as error masking. By default all exceptions thrown on the service side always reach the client as FaultException, as by having all service exceptions indistinguishable from one another, WCF decouples the client from service.

Syntax:

[OperationContract]
[FaultContract(typeof(MyException))]
string getDetails(int value);

```
Assembly System.ServiceModel.dll, v4.0.30319

using System;
using System.Net.Security;

namespace System.ServiceModel
{
    ...public sealed class FaultContractAttribute : Attribute
    {
        ...public FaultContractAttribute(Type detailType);

        ...public string Action { get; set; }
        ...public Type DetailType { get; }
        ...public bool HasProtectionLevel { get; }
        ...public string Name { get; set; }
        ...public string Namespace { get; set; }
        ...public ProtectionLevel ProtectionLevel { get; set; }
    }
}
```

Question 12: What is End Points and how many types of End points?

Answer: Endpoints provide the configuration required for the communication and create the complete WCF service application.

An Endpoint is a piece of information that tells WCF how to build the runtime communication channels to send and receive messages. An endpoint consists of the three things address, binding and contract.
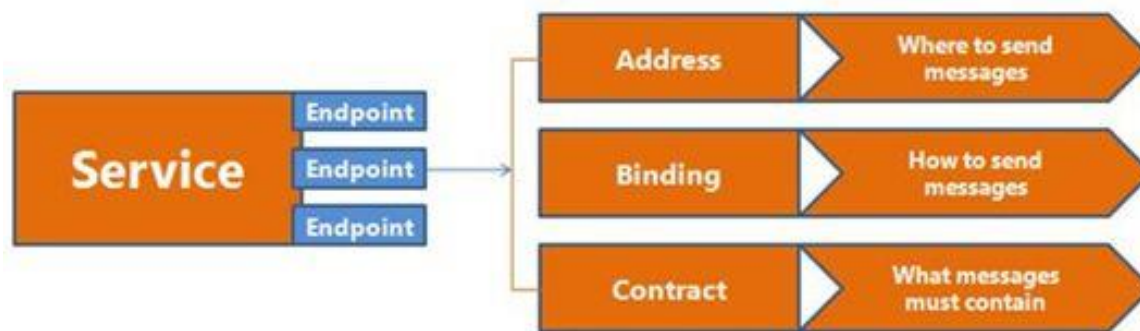
All communication with a Windows Communication Foundation (WCF) service occurs through the endpoints of the service. Endpoints provide clients access to the functionality offered by a WCF service.

Each endpoint consists of four properties:

- An address that indicates where the endpoint can be found.
- A binding that specifies how a client can communicate with the endpoint.
- A contract that identifies the operations available.

The Structure of an Endpoint: Each endpoint consists of the following-

- A set of behaviors that specify local implementation details of the endpoint.

- **Address:** The address uniquely identifies the endpoint and tells potential consumers of the service where it is located. It is represented in the WCF object model by the EndpointAddress class. An EndpointAddress class contains:

  - An Uri property, which represents the address of the service.

- o   An Identity property, which represents the security identity of the service and a collection of optional message headers. The optional message headers are used to provide additional and more detailed addressing information to identify or interact with the endpoint.

- **Binding:** The binding specifies how to communicate with the endpoint. This includes:

  - o   The transport protocol to use (for example, TCP or HTTP).
  - o   The encoding to use for the messages (for example, text or binary).
  - o   The necessary security requirements (for example, SSL or SOAP message security).

- **Contracts:** The contract outlines what functionality the endpoint exposes to the client. A contract specifies:

  - o   What operations can be called by a client?
  - o   The form of the message.
  - o   The type of input parameters or data required to call the operation.
  - o   What type of processing or response message the client can expect.
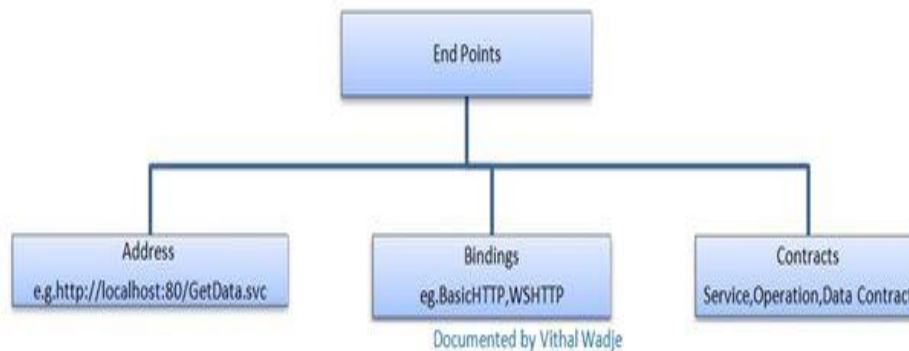
Question 13: What is Address in WCF?

Answer: Address - Where - Where to send messages. An Address is a unique Uniform Resource Locator (URI) that identifies the location of the service. It defines the network address for sending and receiving the messages. It is divided into four parts:
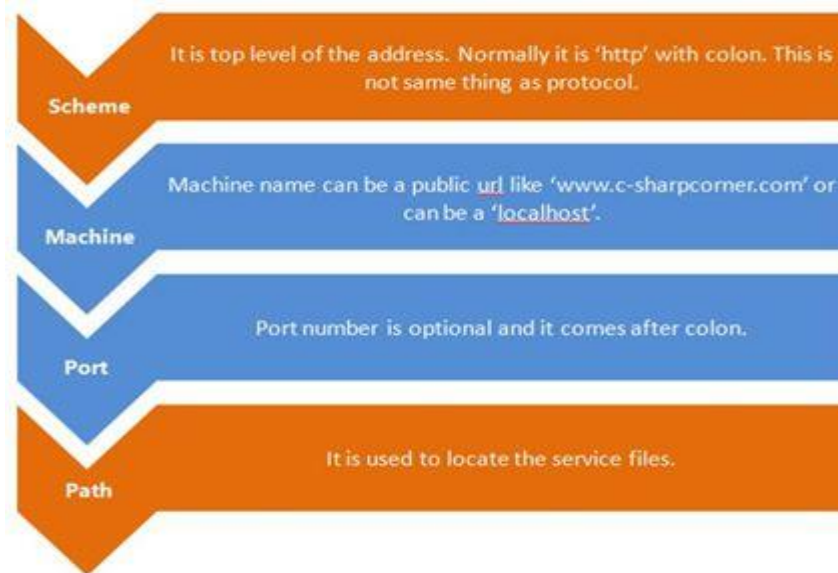
In WCF every service are associated with unique address. The address provided two imported elements the location of services and the transport protocol, which will help to communicate with the services .WCF support the following transport schemes. Here is the answer for what are the supported transport protocols in WCF.

- HTTP/HTTPS
- TCP
- IPC
- Peer Network
- MSMQ
- Service BUS

Format of an address in WCF

Address always have the following format:

[transport]://[machine or domain][:optional port].
ex: https://localhost:82/MyService

©2016 C# CORNER.

Question 14: What is Binding in WCF?What are the types of binding?

Answer: Binding describes how client will communicate with service. There are different protocols available for the WCF to communicate to the Client. You can mention the protocol type based on your requirements.

A binding has several characteristics, including the following:

- **Transport -** Defines the base protocol to be used like HTTP, Named Pipes, TCP, and MSMQ are some type of protocols.

- **Encoding (Optional) -** Three types of encoding are available-Text, Binary, or Message Transmission Optimization Mechanism (MTOM). MTOM is an interoperable message format that allows the effective transmission of attachments or large messages (greater than 64K).

- **Protocol(Optional) -** Defines information to be used in the binding such as Security, transaction or reliable messaging capability.

The following table gives some list of protocols supported by WCF binding:

Binding	Description
BasicHttpBinding	Basic Web service communication. No security by default.
WSHttpBinding	Web services with WS-* support. Supports transactions.
WSDualHttpBinding	Web services with duplex contract and transaction support.
WSFederationHttpBinding	Web services with federated security. Supports transactions.
MsmqIntegrationBinding	Communication directly with MSMQ applications. Supports transactions.
NetMsmqBinding	Communication between WCF applications by using queuing. Supports transactions.
NetNamedPipeBinding	Communication between WCF applications on same computer. Supports duplex contracts and transactions
NetPeerTcpBinding	Communication between computers across peer-to-peer services. Supports duplex contracts
NetTcpBinding	Communication between WCF applications across computers. Supports duplex contracts and transactions

Question 15: Explain transactions in WCF?

Answer: A transaction is a logical unit of work consisting of multiple activities that must either succeed or fail together. For example, you are trying to make an online purchase for a dress; your amount is debited from the bank, but your ordered dress was not delivered to you. So what did you get from the preceding example? Bank operations hosted in a separate service and the online purchase service (Ordering the goods) hosted as another separate service. For the preceding example the online purchase service did not work well as expected due to the database query error, without knowing it the Bank operations were completed perfectly. For this we really need transactions to ensure that either of the operations should succeed or fail. Before we go to the WCF transactions, let us discuss what exactly the basics behind the transactions are:

The following are the types of the Transactions:

1. Atomic
2. Long Running

Phases of WCF Transaction:

There are two phases in a WCF transaction as shown in the following figure.
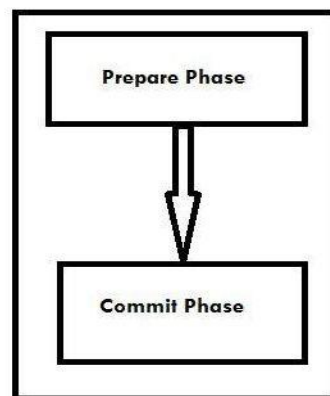
- **Prepare Phase -** In this phase, the transaction manager checks whether all the entities are ready to commit for the transaction or not.
- **Commit Phase -** In this phase, the commitment of entities get started in reality.

Question 16: What are the possible ways of hosting a WCF service?

Answer: A WCF service can be self-hosting using a console application or Windows Forms applications. Hosting a WCF service in any managed .Net application is called self-hosting. Now the following is the procedure for creating a WCF service and hosting it in a console application. Hosting Environment Requirements:

- **Availability:** When do you want to be able to reach your service?
- **Reliability:** What happens when your service somehow breaks? How does this affect other consumers?
- **Manageability:** Do you need easy access to information about what is happening on the host where WCF services live?
- **Versioning:** Do you need to support older versions of the service? Do you know who is consuming your services?
- **Deployment:** What is your deployment model? Are you installing through the Microsoft Installer process and Visual Studio deployment packages, or is xcopy sufficient?
- **State:** Are your services stateless? Do you need sessions?

The following are the various options available for hosting a WCF Service:

1. Self-Hosting
2. Windows Service
3. Internet Information Services (IIS)
4. Windows Activation Services (WAS)

Question 17: What is Self Hosting in WCF?

Answer: We have our project (it may be Windows or a web application or something else) and we want to host one WCF service within this solution locally. This type of hosting is called Self-Hosting. To implement self-hosting we need to include System.Service.Model.ServiceHost namespace.

The following are the advantages of self-hosting:

- **Is easy to use:** With only a few lines of code you have your service running.

- **Is flexible:** You can easily control the lifetime of your services through the **Open()** and **Close()** methods of **ServiceHost<T>**.

- **Is easy to debug:** Debugging WCF services that are hosted in a self-hosted environment provides a familiar way of debugging, without having to attach to separate applications that activate your service.

- **Is easy to deploy:** In general, deploying simple Windows applications is as easy as xcopy. You don't need any complex deployment scenarios on server farms, and the like, to deploy a simple Windows application that serves as a WCF **ServiceHost**.

- **Supports all bindings and transports:** Self-hosting doesn't limit you to out-of-the-box bindings and transports whatsoever. On Windows XP and Windows Server 2003, IIS limits you to HTTP only.

The following are the disadvantages of self-hosting:

- **Limited availability:** The service is reachable only when the application is running.

- **Limited features:** Self-hosted applications have limited support for high availability, easy manageability, robustness, recoverability, versioning, and deployment scenarios. At least, out-of-the-box WCF doesn't provide these, so in a self-hosted scenario you have to implement these features yourself; IIS, for example, comes with several of these features by default.

Question 18: What are the main components of WCF Service?

Answer: Service - A Service is a program or application that is used by other applications. In other words, a service that is directly consumed by the end user to do their work and is something they ask for and recognize Or It's just self-contained business functionality.

WCF Service is - Windows Communication Foundation (WCF) was introduced in .NET 3.0. This is a great network distributed system developed by Microsoft for communication between applications. WCF is meant for designing and deploying distributed applications under a Service-Oriented Architecture (SOA) implementation. From MSDN: Windows Communication Foundation is a part of the .NET Framework that provides a unified programming model for

rapidly building service-oriented applications that communicate across the web and the enterprise. WCF is a combined feature of Web Service, Remoting, MSMQ and COM+.

There are the following three main components of a WCF application.

1. **WCF Service class:** A WCF Service class implements some service as a set of methods.

2. **WCF Service host:** WCF supports four types of hosting, IIS, Windows Process Activation Services (WAS), self-hosting and Windows Services.

3. **WCF Service endpoints:** All communication with a Windows Communication Foundation (WCF) service occurs through the endpoints of the service. Usually the name of the Interface will be specified in the contract, so the client application will be aware of the operations that are exposed to the client.

Each endpoint contains:

- **Address:** An address that indicates where to find the service.
- **Binding:** A binding that specifies how a client can communicate with the service.
- **Contract:** A Contract that specifies what can the service do for us.

Question 19: How do we host a WCF service in IIS?

Answer: IIS hosting - If we are hosting a WCF Service in IIS (version 5 or 6), only the HTTP option is available as the transport protocol. In later versions of Internet Information Services (IIS), we can use any other protocol (TCP/IP, MSMQ, NamedPipes and so on.) as transport protocol.

If we host our service under IIS then we have all the features of IIS like process recycling, ideal shutdown etc. Visual Studio provides two different ways to host our WCF service in a local IIS in all are familiar with Publishing Wizard. Here we will discuss the second way using Visual Studio only in a simple manner to host our WCF service in local IIS. To host the WCF services follow the steps below.

The updated configuration for System.ServiceModel in the web.config will be as follows:

```
1.    <system.serviceModel>
2.       <behaviors>
3.          <serviceBehaviors>
4.             <behavior name="StudentServiceBehavior">
5.                <serviceMetadata httpGetEnabled="true"/>
6.                <serviceDebug includeExceptionDetailInFaults="false"/> </behavior>
7.          </serviceBehaviors>
8.       </behaviors>
9.       <services>
10.         <service behaviorConfiguration="StudentServiceBehavior" name="StudentService.StudentService">
11.            <endpoint address="http://localhost/StudentIISHost/MyStudentHost.svc" binding="wsHttpBinding" contract="StudentService.IStudentService">
12.               <identity>
13.                  <dns value="localhost"/> </identity>
14.            </endpoint>
15.            <endpoint address="mex" binding="mexHttpBinding" contract="IMetadataExchange"/> </service>
16.      </services>
17. </system.serviceModel>
```

Question 20: What is one-way operation?

Answer: When an operation has no return value and the client does not care about the success or failure of the invocation. WCF offers one-way operations to support this sort of fire-and-forget invocation, once the client issues the call, WCF generates a request message, but no correlated reply message will ever return to the client. The one-way message exchange pattern is useful when a client needs to send information to a service but doesn't receive a response.

Use the one-way design pattern:

- When the client must call operations and is not affected by the result of the operation at the operation level.

- When using the NetMsmqBinding or the MsmqIntegrationBinding class. (For more information about this scenario, see Queues in Windows Communication Foundation.)

To create a one-way service contract, define your service contract, apply the OperationContractAttribute class to each operation, and set the IsOneWay property to true, as shown in the following sample code.

```
1.  [ServiceContract(Namespace = "http://Microsoft.ServiceModel.Samples")]
2.  public interface IOneWayCalculator
3.  {
4.      [OperationContract(IsOneWay = true)]
5.      void Add(double n1, double n2);
6.      [OperationContract(IsOneWay = true)]
7.      void Subtract(double n1, double n2);
8.      [OperationContract(IsOneWay = true)]
9.      void Multiply(double n1, double n2);
10.     [OperationContract(IsOneWay = true)]
11.     void Divide(double n1, double n2);
12. }
```

Question 21: Why we need One Way Service?

Answer: There might be requirements where Service's Operation is not going to return any value. And at the same time client also does not bother about success or failure of service. Only for the client does calling the service matter. Client will call the operation and forget and continue with the operations.

One Way Operations: This is the way in WCF to achieve Fire and forget invocation of the service. Once the Client issues the call, WCF generates the request message and there is no corresponding response message for that. No Service side Exception will make their way to client.

Configuring One Way Operation: Boolean **ISOneWay** property of **OperationContract** class is used to configure service as one way service. We need to set this property to true. Setting true to this property turns a service operation as one way operation.

```
1.   using System;
2.   using System.Net.Security;
3.   namespace System.ServiceModel
4.   {
5.       [AttributeUsage(AttributeTargets.Method)]
6.       public sealed class OperationContractAttribute: Attribute
7.       {
8.           public OperationContractAttribute();
9.           public string Action
10.          {
11.              get;
12.              set;
13.          }
14.          public bool AsyncPattern
15.          {
16.              get;
17.              set;
18.          }
19.          public bool HasProtectionLevel
20.          {
21.              get;
22.          }
23.          public bool IsInitiating
24.          {
25.              get;
26.              set;
27.          }
28.          public bool IsOneWay
29.          {
30.              get;
31.              set;
32.          }
33.          public bool IsTerminating
34.          {
35.              get;
36.              set;
37.          }
38.          public string Name
39.          {
40.              get;
41.              set;
42.          }
43.          public ProtectionLevel ProtectionLevel
```

```
44.    {
45.       get;
46.       set;
47.    }
48.    public string ReplyAction
49.    {
50.       get;
51.       set;
52.    }
53.  }
54. }
```

The client has nothing to do anything specific to invoke one way operation.

Question 22: Explain what is DataContractSerializer?

Answer: WCF provides a message-oriented programming framework. We use WCF to transmit messages between applications. Internally WCF represents all the messages by a Message class. When WCF transmits a message it takes a logical message object and encodes it into a sequence of bytes. After that WCF reads those bytes and decodes them in a logical object. The process forms a sequence of bytes into a logical object; this is called an encoding process. At runtime when WCF receives the logical message, it transforms them back into corresponding .Net objects. This process is called serialization.

WCF supports three basic serializers:

* XMLSerializer
* NetDataContractSerializer
* DataContractSerializer

WCF deserializes WCF messages into .Net objects and serializes .Net objects into WCF messages. WCF provides DataContractSerializer by default with a servicecontract. We can change this default serializer to a custom serializer like XMLSerializer.

```
1.  [XmlSerializerFormat]
2.  [ServiceContract]
3.  public interface IService1
4.  {
5.      [OperationContract]
6.      void AddAuthor(Author author);
7.  }
```

The XmlSerializerFormat attribute above the ServiceContract means this serializer is for all operation contracts. You can also set a separate contract for each operation.

XMLSerializer-

We can find XMLSerializer in the System.Xml.Serialization namespace. WCF supports this serialization from .Net 1.0. It is used by default with the ASP.Net webservices (ASMX).

NetDataContractSerializer-

NetDataContractSerializer is analogous to .Net Remoting Formatters. It implements IFormatter and it is compatible with [Serializable] types. It is not recommended for service oriented design.

DataContractSerializer-

DataContractSerializer is a default serializer for WCF. We need not to mention DataContractSerializer attribute above the service contract.

Question 23: What is Transaction Propagation? And how WCF support it?

Answer: Transaction propagation is the ability to propagate a transaction across the boundaries of a single service. Or in other words, we can say that a service can participate in a transaction that is initiated by a client.

In a SOA environment, transaction propagation becomes a key requirement. As we know, WCF supports SOA, so it provides support for transaction propagation as well.

To enable transaction propagation, we need to set the value of the TransactionFlow property of the binding being used. This can be done programmatically as follows:

```
1.  WSHttpBinding bindingBeingUsed = new WSHttpBinding();
2.  bindingBeingUsed.TransactionFlow = "true";
```

Or it can be done decoratively by updating the configuration file as follows:

1. <bindings>
2.   <wsHttpBinding>
3.   <binding name="binding1" transactionFlow="true" />
4.   </wsHttpBinding>
5. </bindings>

The default value for the TransactionFlow property is "**False**".

Do all WCF bindings support Transaction Propagation?

No. Not all WCF bindings support transaction propagation. Only the following list of bindings support it:

- wsHttpBinding
- netTcpBinding
- netNamedPipeBinding
- wsDualHttpBinding
- wsFederationHttpBinding

Question 24: What is WCF throttling?

Answer: WCF throttling provides some properties that you can use to limit how many instances or sessions are created at the application level. Performance of the WCF service can be improved by creating proper instance.

The throttling of services is another key element for WCF performance tuning. WCF throttling provides the prosperities maxConcurrentCalls, maxConcurrentInstances, and maxConcurrentSessions, which can help us to limit the number of instances or sessions are created at the application level.

Attribute / Property	Description
maxConcurrentCalls	This specifies the maximum number of messages processed across the service host. The default value for this property is 16 (WCF 4.0 is improved to default is 16 * Processor Count).
maxConcurrentInstances	This specifies the maximum number of instances of a context object that executes at one time with the service. The default is Int32.MaxValue.
maxConcurrentSessions	This specifies the maximum number of sessions at one time within the service host object. The default value is 10 (WCF 4.0 increases that to 100 * Processor Count).

```
1.  <configuration>
2.    <system.serviceModel>
3.      <services>
4.        <service .... .... .... </service>
5.      </services>
6.      <behaviors>
7.        <serviceBehaviors>
8.          <behavior name="ServiceBehavior">
9.            <serviceThrottling maxConcurrentCalls="16" maxConcurrentSessions="100"
   maxConcurrentInstances="10" />
10.           <serviceMetadata httpGetEnabled="true" /> </behavior>
11.        </serviceBehaviors>
12.      </behaviors>
13.    </system.serviceModel>
14. </configuration>
```

Question 25: What is a Service Proxy in WCF?

Answer: A service proxy or simply proxy in WCF enables application(s) to interact with a WCF Service by sending and receiving messages. It's basically a class that encapsulates service details i.e. service path, service implementation technology, platform and communication protocol etc. It contains all the methods of a service contract (signature only, not the implementation).

A client application uses the **WCF** client **proxy** to communicate with the service. Client applications usually import a **service's** metadata to generate **WCF** client code that can be used to invoke the **service**.

The basic steps for creating a **WCF** client include the following: Compile the service code. The proxy provides the same operation as service contract, but also provides additional methods for managing proxy life cycle and the connection to the service. Proxy contains all the information of the service like Address, Binding and Configuration so that client knows how to communicate with the service. What security policies he has to follow and all whatever is dictated by the service to work properly.

Question 26: What is WCF Concurrency and How many modes are of Concurrency in WCF?

Answer: WCF Concurrency means "Several computations are executing simultaneously. WCF concurrency helps us configure how WCF service instances can serve multiple requests at the same time. You will need WCF concurrency for the below prime reasons; there can be other reasons as well but these stand out as important reasons:

We can use the concurrency feature in the following three ways:

- Single
- Multiple
- Reentrant

Single: A single request will be processed by a single thread on a server at any point of time. The client proxy sends the request to the server, it process the request and takes another request.

The following is the declaration of Concurrency.Single:

```
[ServiceBehavior(ConcurrencyMode = ConcurrencyMode.Single)]
public class CafeShopServiceImpl : ICafeShopService
{
```

Multiple: Multiple requests will be processed by multiple threads on the server at any point of time. The client proxy sends multiple requests to the server. Requests are processed by the server by spawning multiple threads on the server object.

Reentrant: The reentrant concurrency mode is nearly like the single concurrency mode. It is a single-threaded service instance that receives requests from the client proxy and it unlocks the thread only after the reentrant service object calls the other service or can also call a WCF client through call back.

1. [ServiceBehavior(ConcurrencyMode = ConcurrencyMode.Multiple]
2. Public class MyService : IMyService
3. {
4. }

Question 27: What is REST and how to create a WCF RESTful Service?

Answer: REST - Representational State Transfer (REST) is a protocol for exchanging data over a distributed environment. The main idea behind REST is that we should treat our distributed services as a resource and we should be able to use simple HTTP protocols to perform various operations on that resource.

JSON - The **JavaScript Object Notation (JSON)** data format, or JSON for short, is derived from the literals of the JavaScript programming language. **JSON** helps us to present and exchange data in a self-descriptive, independent and light way. This data can then be easily consumed and transformed into JavaScript objects.

In most browser-based applications, WCF can be consumed using JavaScript, jQuery, AngularJS and so on. When a client makes a call to WCF, JSON or XML is used as the format of the communication. WCF has the option to send the response in JSON object. This can be configured with the WebGet orWebInvoke attribute and the WebHttpBinding. This allows you to expose a ServiceContract as a REST service that can be invoked using either JSON or plain XML.

- GET: Retrieves the information.
- PUT: Replaces the entire collection with another collection.
- POST: Creates a new entry in the collection.
- DELETE: Deletes the entire collection.

Question 28: What is Exception Handling in WCF? What are the ways for WCF Exception Handling?

Answer: Exception handling is critical to all applications for troubleshooting the unexpected problems in applications. The Windows Communication Framework (WCF) provides several options for handling exceptions in WCF services. This article discusses these approaches and describes the advantages and disadvantages of each. The following options are provided to handle exceptions in WCF:

1. Using returnUnknownExceptionsAsFaults: **Debugging Mode**
2. Using FaultException: **Best Option.**
3. Using IErrorHandler: **Only when the exception can't be handled by Fault**

Exception handling in WCF:
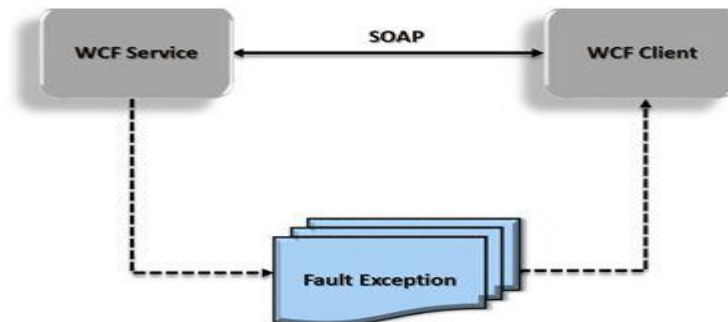
Exceptions inside a WCF Service - Before describing the details of exception handling in WCF, let's explore what happens if we do not handle an exception inside the service. Consider a service with the CreateUser method as shown in the following:

```
1.  public void CreateUser(User user)
2.  {
3.     if(user.isValid())
4.     {
5.         //Create User
6.     }
7.     else
8.     {
9.         throw new ApplicationException("User Inavalid.");
10.    }
11. }
```

Example:

```
1.  public class DBManagerService: IDBManagerService
2.    void Save(Employee emp)
3.  {
4.     try
5.     {
6.         //Code to store an employee object to the database
7.     }
8.     catch (Exception ex
9.        {
10.           throw new Exception("Error occurred while saving data….");
11.       }
12.    }
13. }
```

Question 29: What is Tracing in WCF?

Answer: The tracing mechanism in the Windows Communication Foundation is based on the classes that reside in the System.Diagnostic namespace. The important classes are Trace, TraceSource and TraceListener.

Configuring WCF to emit tracing information/Define Trace Source, we have the following options:

- System.ServiceModel
- System.ServiceModel.MessageLogging
- System.ServiceModel.IdentityModel
- System.ServiceModel.Activation
- System.Runtime.Serialization
- System.IO.Log
- Cardspace

Example

```
1.  configuration>
2.     <system.diagnostics>
3.        <sources>
4.           <source name="System.ServiceModel"
5.                  switchValue="Information, ActivityTracing"
6.                  propagateActivity="true">
7.              <listeners>
8.                 <add name="traceListener"
9.                    type="System.Diagnostics.XmlWriterTraceListener"
10.                   initializeData= "c:\log\Traces.svclog" />
11.             </listeners>
12.          </source>
13.       </sources>
14.    </system.diagnostics>
15. </configuration>
```

There are essentially four steps involved in WCF Tracing:

Set the Trace Level:

```
<system.diagnostics>
  <sources>
    <source name="System.ServiceModel"
            switchValue="Information, ActivityTracing">

    </source>
  </sources>
```

Question 30: What are the Various Types of Bindings WCF Supports?

Answer: Binding describes how a client is going to communicate to WCF service. Binding is used as per client need. It supports different types of protocol to communicate with client and different types of encoding to transfer the data over the internet. So, basically binding is nothing but it is a way of communication between client and service as per client need.

Basic binding - This binding is provided by the BasicHttpBinding class. It is designed to expose a WCF service as an ASMX web service, so that old clients (that are still using an ASMX web service) can consume the new service. By default, it uses the HTTP protocol for transport and encodes the message in UTF-8 text format. You can also use HTTPS with this binding.

Web binding - This binding is provided by the WebHttpBinding class. It is designed to expose WCF services as HTTP requests using HTTP-GET and HTTP-POST. It is used with REST based services that may provide output in XML or JSON format. This is very much used with social networks for implementing a syndication feed.

Web Service (WS) binding - This binding is provided by the WSHttpBinding class. It is like a basic binding and uses HTTP or HTTPS protocols for transport. But this is designed to offer various WS - * specifications such as WS – Reliable Messaging, WS - Transactions, WS - Security and so on which are not supported by Basic binding.

wsHttpBinding= basicHttpBinding + WS-* specification

WS Dual binding - This binding is provided by the WsDualHttpBinding class. It is like a WsHttpBinding except it supports bi-directional communication which means both clients and services can send and receive messages.

TCP binding - This binding is provided by the NetTcpBinding class. It uses TCP protocol for communication between two machines within intranet (means same network). It encodes the message in binary format. This is a faster and more reliable binding compared to the HTTP protocol bindings. It is only used when the communication is WCF-to-WCF which means both client and service should have WCF.

IPC binding - This binding is provided by the NetNamedPipeBinding class. It uses named pipe for communication between two services on the same machine. This is the most secure and fastest binding among all the bindings.

MSMQ binding - This binding is provided by the NetMsmqBinding class. It uses MSMQ for transport and offers support to a disconnected message queued. It provides solutions for disconnected scenarios in which the service processes the message at a different time than the client sending the messages.

Federated WS binding - This binding is provided by the WSFederationHttpBinding class. It is a specialized form of WS binding and provides support to federated security.

Question 31: What is Instance Context Mode in WCF?

Answer: An Instance Context mode defines how long a service instance remains on the server.

Request Response Cycle:
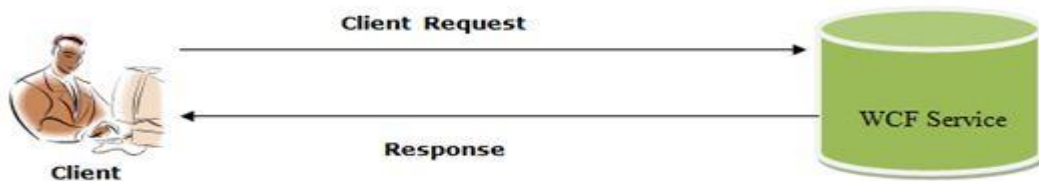
Whenever the client sends a request to a WCF service, what exactly happens behind the scenes? Basically after making a client request the WCF service will create a service class instance at the service that will do the operations involved and then it will return the response back to the client. In this request and response process the service instance object has been created in the process.

The WCF framework has defined the following three Instance Context modes:

PerCall: A new instance of the service will be created for the request from the same client or a different client, meaning every request is a new request. In this mode no state is maintained.In per-call service, every client request achieves a new dedicated service instance and its memory consumption is less as compared to other types of instance activation.

1. [ServiceContract]
2. interfaceIMyContract
3. {...}
4. [ServiceBehavior(InstanceContextMode=InstanceContextMode.PerCall)]
5. classMyService:IMyContract
6. {...}

Per-Session Service: A new Instance will be created for every new client session and the scope of that object will be the scope of that session.

1. [ServiceBehavior(InstanceContextMode=InstanceContextMode.PerSession)]
2. classMyService:IMyContract
3. {...}

Singleton Service: A single instance will be created for the service object that will take care of all the requests coming from the same client or a different one.

By decorating the service with a service behavior, an Instance Context mode can be set.

[ServiceBehavior(InstanceContextMode=InstanceContextMode.PerCall)]

Question 32: What is the KnownType Attribute in WCF?

Answer: The KnownTypeAttribute class allows you to specify, in advance, the types that should be included for consideration during deserialization. The WCF service generally accepts and returns the base type. If you expect the service to accept and return an inherited type then we use the knowntype attribute.

When passing parameters and return values between a client and a service, both endpoints share all of the data contracts of the data to be transmitted.

```
1.  [DataContract]
2.  [KnownType(typeof (Student))]
3.  [KnownType(typeof (Teacher))]
4.  publicabstractclass Person
5.  {
6.      [DataMember]
7.      publicint Code
8.      {
9.          get;
10.         set;
11.     }
12.     [DataMember]
13.     publicstring Name
14.     {
15.         get;
16.         set;
17.     }
18. }
```

Question 33: How to create Basic HTTP Binding in WCF?

Answer: The **basicHttpBinding** binding is used in order to provide support for older clients that expect a legacy ASMX Web service. The **TransportCredentialOnly** security mode option passes the user credentials without encrypting or signing the messages.

BasicHttpBinding is suitable for communicating with ASP.NET Web Service (ASMX) based services that conform to the WS-Basic Profile that conforms with Web Services.

- This binding uses HTTP as the transport and text/XML as the default message encoding.
- Security is disabled by default.
- This binding does not support WS-* functionalities like WS- Addressing, WS-Security, WS-ReliableMessaging.
- It is fairly weak on interoperability.

Use the following procedure:

- Create a WCF basicHttpBinding project.
- Start Visual Studio and select New Project from the Start page or from the File menu, select New and then Project.
- After selecting, a dialog will pop up showing all the templates provided by Visual Studio.
- From the Installed Templates select C# then inside that select WCF and inside that you will select WCF Service Application.

Question 34: What is the difference between BasicHttpBinding and WsHttpBinding?

Answer: Below is a detailed comparison table between both the entities from security, compatibility, reliability, and SOAP version perspectives.

Criteria	BasicHttpBinding	WsHttpBinding
Security support	This supports the old ASMX style, i.e., WS-BasicProfile 1.1.	This exposes web services using WS-* specifications.
Compatibility	This is aimed for clients who do not have .NET 3.0 installed and it supports wider ranges of clients. Many of the clients like Windows 2000 still do not run .NET 3.0. So an older version of .NET can consume this service.	As it is built using WS-* specifications, it does not support wider ranges of clients and it cannot be consumed by older .NET versions less than 3 version.
SOAP version	SOAP 1.1	SOAP 1.2 and WS-Addressing specification.
Reliable messaging	Not supported. In other words, if a client fires two or three calls you really do not know if they will return back in the same order.	Supported as it supports WS-* specifications.
Default security options	By default, there is no security provided for messages when the client calls happen. In other words, data is sent as plain text.	As WsHttBinding supports WS-*, it has WS-Security enabled by default. So the data is not sent in plain text.
Security options	• None • Windows – default authentication • Basic • Certificate	• None • Transport • Message • Transport with message credentials

One of the biggest differences you must have noticed is the security aspect. Bydefault, BasicHttpBinding sends data in plain text while **WsHttpBinding** sends it in an encrypted and secured manner. To demonstrate the same, let's make two services, one using **BasicHttpBinding** and the other using WsHttpBinding and then let's see the security aspect in a more detailed manner.

Question 35: What is Security Implementation in WCF? How many are there?

Answer: As WCF supports various protocols i.e. TCP, HTTP, and MSMQ, user must be sure enough to take necessary steps to guard your message and also must establish security policies for protecting messages and for authenticating and authorizing calls. WCF provide a very easy and rich configurable environment to implement security.WCF supports following securities:

- Message
- Transport
- TransportWithMessageCredential

Message Security - Message security uses the WS-Security specification to secure messages. The message is encrypted using the certificate and can now safely travel over any port using plain http. It provides end-to-end security.

Transport Security - Transport security is a protocol implemented security so it works only point to point. As security is dependent on protocol, it has limited security support and is bounded to the protocol security limitations.

TransportWithMessageCredential - This we can call a mixture of both Message and Transport security implementation.

Question 36: What is Streaming in WCF?

Answer: In WCF any receiving message is delivered only once entire message has been received. What I mean here is that first message is buffered at the receiving side and once it is fully received it gets delivered to the receiving end. Main problem with this approach is that receiver end is unresponsive while message is getting buffered. So default way of message handling in WCF is ok for small size messages but for the large size messages this approach is not good. So to overcome this problem Streaming in WCF come into action.

Steaming and Binding:

1. TCP, IPC and HTTP bindings support streaming.
2. For all the Binding streaming is disabled by default.
3. Streaming of the message should be enabled in the binding to override the buffering and enable the streaming.
4. TransferMode property should be set according to the desired streaming mode in the bindings.
5. Type of TransferMode property is enum TransferMode

Configuring Streaming in Config file:

1. <system.serviceModel>
2.   <services>
3.   <service name="OneWayService.Service1" behaviorConfiguration="OneWayService.Service1Behavior">
4.   <!-- Service Endpoints -->
5.   <endpoint address="" binding="basicHttpBinding" bindingConfiguration="StreamedHttp" contract="OneWayService.IService1">
6.   <identity>
7.   <dns value="localhost" /> </identity>
8.   </endpoint>
9.   <endpoint address="mex" binding="mexHttpBinding" contract="IMetadataExchange" /> </service>
10.   </services>
11.   <bindings>
12.   <basicHttpBinding>
13.   <binding name="StreamedHttp" transferMode="Streamed" /> </basicHttpBinding>
14.   </bindings>
15.   <behaviors>
16.   <serviceBehaviors>
17.   <behavior name="OneWayService.Service1Behavior">
18.   <serviceMetadatahttpGetEnabled="true" />
19.   <serviceDebugincludeExceptionDetailInFaults="false" /> </behavior>
20.   </serviceBehaviors>
21.   </behaviors>
22. </system.serviceModel>

Question 37: What is SOA, and Messages in WCF?

Answer: SOA is nothing but an architectural style where two non-compatible applications can communicate using a common language and WCF is nothing but one example of the Service Oriented Architecture (SOA).Characteristics of SOA:

- In SOA, Services should be independent of other services. Altering a service should not affect calling service.

- Services should be able to define themselves. Services should be able to answer a question what is does? It should be able to tell client what all operations it does, what all data types it uses and what kind of responses it will return.

- Services should support reliable messaging. Means there should be a guarantee that request will be reached to correct destination and correct response will be obtained.

- Services should support secure communication.

Message Pattern: A message in WCF is very similar to messages in the real world.

It describes how the programs will exchange message each other. There are three way of communication between source and destination,

There is no need to call a Java class from a C# class. When they want to talk with each other they just send a message to each other. In the Next programming industry though it's fine and cool to communicate via messages but there should be some standardization of those messages. Then, when people try to exchange data between two programs they send the standardized message to get it done.
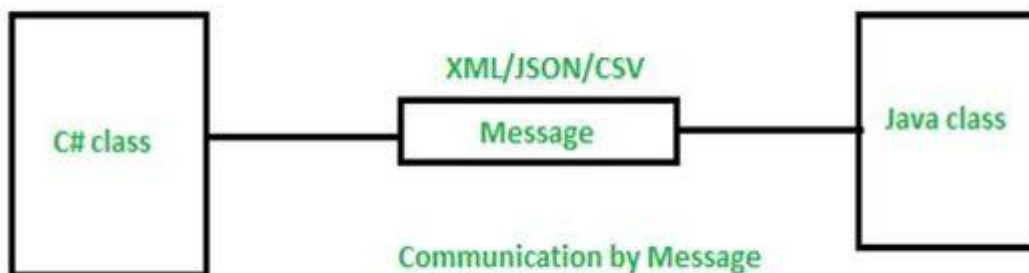
Question 38: What is Transport level security in WCF?

Answer: The goal of transport level security is to provide integrity, privacy and authentication.
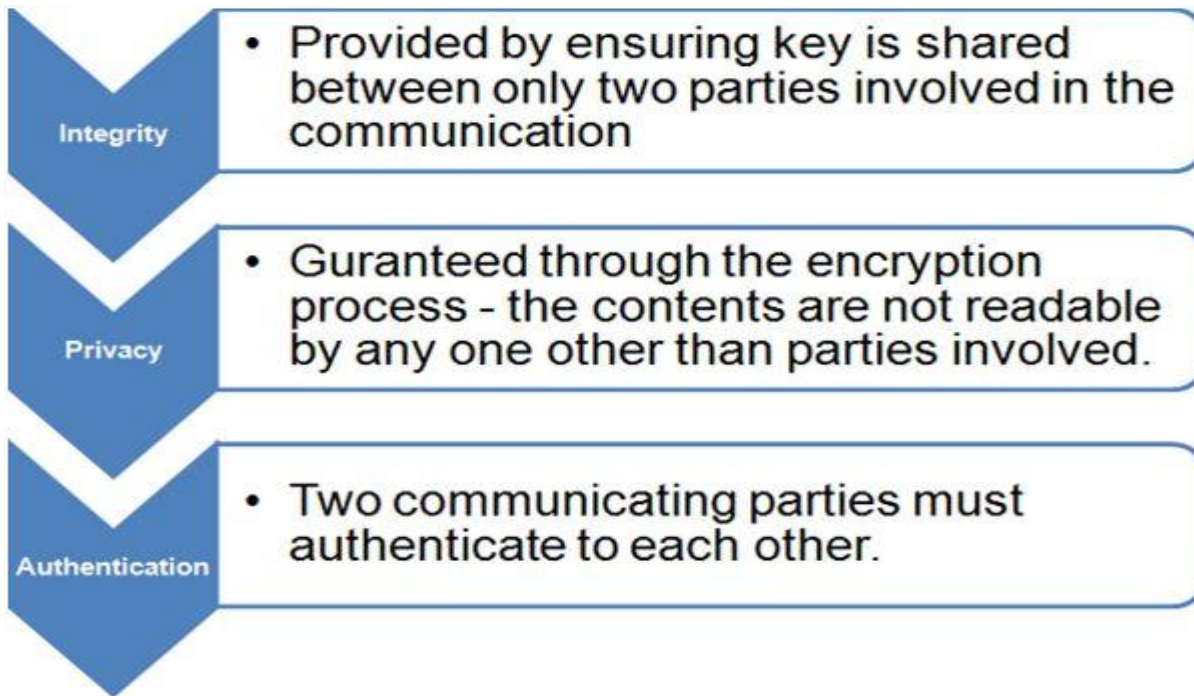
Advantage of using Transport security:

- Less chances of sniffing network.
- Less chances of Phishing network.
- Less chances of message alteration.
- Less chances of replay of message attack.

Regardless of the Binding used, Transport level security provides:

- Authentication of the sender.
- Authentication of the service.
- Message integrity
- Message confidentiality.
- Replay of message detection.

Question 39: What Message Exchange Patterns (MEPs) supported by WCF? Explain each of them briefly.

Answer: MEP stands for Message Exchange Pattern. In WCF we are creating services. What does a service do for us? A service does some task on behalf of us or sends a response back from our request. All such communications are done using messages. We send some message as a request and get a message as a response from the service. WCF supports three types of MEPs:

- Request / Response
- One-Way
- Duplex

Message Exchange Patterns describes the way of communication between Client and Server means how client and server would be exchange messages to each other. There are three types of message exchange patterns.

Request/Response - Request / Response is a very common MEP. To set this pattern we need to set the IsOneWay property of OperationContractAttribute as false. And the default value of the IsOneWay property is false. So ultimately we do not need to set this property. It is set by default. That is why it is very easy to implement.

Request-Reply Message Exchange Pattern

1. [ServiceContract]
2. publicvinterface RequestReplyService
3. {
4.   [OperationContract]
5.   string GetData(int value);
6.
7.   [OperationContract(IsOneWay = false)]
8.   voidSaveData(string value);
9. }

One-Way - Set the IsOneWay property of OperationContractAttribute to true for an OneWay message exchange pattern. Suppose you send a message to a service. This pattern is used when the service does some operation and you do not want a response back. For example you want to change the status of a transaction from pending to complete and you do not want to get a confirmation from the service that the status is changed. You can use an OneWay pattern.

1. [ServiceContract]
2. public interface IService1
3. {
4.   [OperationContract(IsOneWay=true)]
5.   void ChangeTransactionStatus(int value);
6. }

One-Way Message Exchange Pattern

Duplex - The duplex MEP is a two-way message channel. When the client sends a message to the service to instantiate some long-running processing and requires notification back from the service, a duplex MEP is applicable.

There are two types of contacts, ServiceContract and CallbackContract required implementing a duplex MEP. This MEP is declared by associating a CallbackContract with the ServiceContract. It is done through a CallbackContract property of ServiceContract.

Duplex (Request-Reply) Message Exchange Pattern

Duplex (One-Way) Message Exchange Pattern
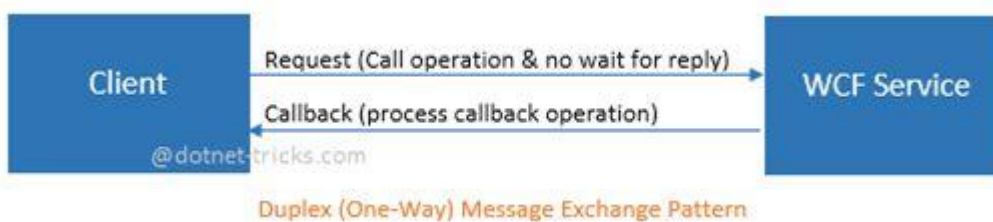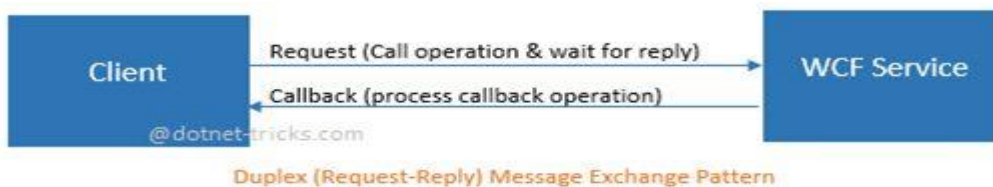
Question 40: Explain briefly different Instance Modes in WCF?

Answer: Instance mode is basically a service side implementation detail. It should not manifest itself on the client side. WCF defines the behaviors. Behavior is a local attribute of a service or client that does not affect communication. The client never knows about the service behavior and does not manifest service binding or published metadata.

Instance Modes available in WCF are:

1. PerCall
2. PerSession
3. Single

PerCall Services:

When a service is configured PerCall, every client gets a new service instance. The above diagram shows how a PerCall instance works:

1. Client calls the proxy and proxy forwards the call to service.
2. WCF creates an instance and calls the method call.
3. When method call returns, WCF calls IDisponsible if implemented.

PerSession Service - WCF maintains a logical session between client and service in Persession instance mode. A new instance of the service class is created for each client.

The following given diagram represents the request from the client using Persession service instance mode:
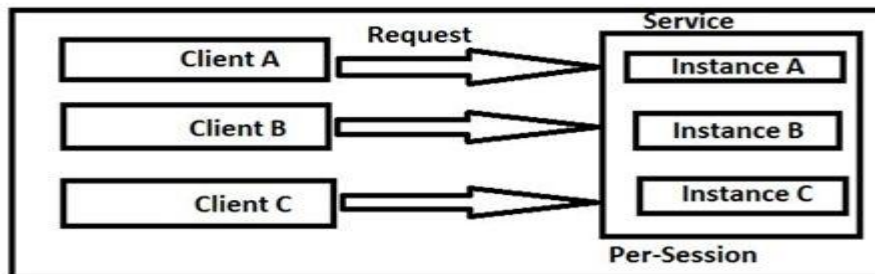
Single Instance Service - When we configure a service as a singleton, all clients are connected to a single instance context. We configure a singleton service by setting the InstanceContextMode property as single. Only one instance is created upon creation of a service host. This instance is forever for the service. Another service instance is created only when we reset the IIS or when the service host is shut down.

Diagrammatic representation of a singletion instance:
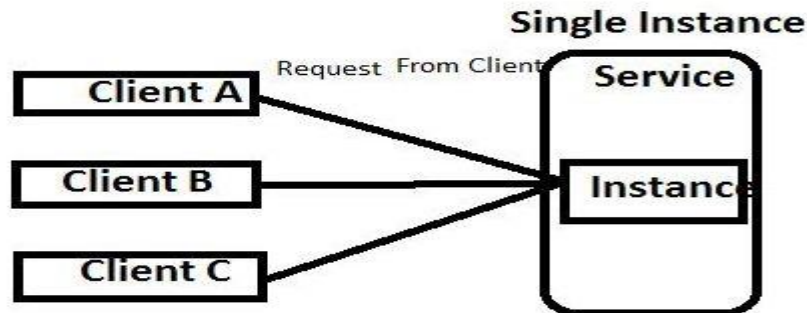
When should you use per call, per session, and single mode?

Per call:

- You want a stateless service.
- Your service holds intensive resources like connection objects and huge memory objects.
- Scalability is a prime requirement. You would like to have a scaled out architecture.
- Your WCF functions are called in a single threaded model.

Per session:

- You want to maintain states between WCF calls.
- You a scaled up architecture.
- Light resource references.

Single:

- You want share global data through your WCF service.
- Scalability is not a concern.

Question 41: Explain bindings in WCF with description?

Answer: It specifies how to access the hosted service. There are following characteristics of binding:

Transport defines the communication protocol to be used to communicate between service and client. It may be HTTP, TCP, MSMQ, NamedPipes etc. It is mandatory to define transport.

Encoding defines the technique used to encode the data before communicating it from one end to the other.

Protocol defines the configurations like reliability, security, transaction, timeouts, message size, etc.

The following table shows the types of bindings and their uses:

Binding Name	Description
BasicHttpBinding	Allows us to create and consume ASMX-style services within WCF. No Security provided for the messages. No Reilable Messaging, Responses will not come in order. Suitable for clients who do not have .Net 3.0 installed. Transport: HTTP.
webHTTPBinding	Allows us to create and expose your services as HTTP requests. Used for REST Based Services.
wsHttpBinding	Exposes web services using ws-* specifications. Security provided (Ws-Security) for messages by default. Uses for WS-Transactions, WS-BusinessActivity.
wsDualHttpBinding	This is like the preceding one, but it uses duplex contracts. This means the both client and server can both send and receive messages
wsFederationHttpBinding	Provides a mechanism for exposing a federated service. The client uses a security token issued by a security token service to authenticate.
netTCPBinding	Service to Service Communication in an intranet. Features include transactions and security. For WCF to WCF service communication netTCPBinding is best.
netPeerTCPBinding	Used when you require more security for Peer to Peer Communication as netTCPBinding.
netMsMqBinding	Binding for asynchronous communication using Microsoft Message Queue (MSMQ). Best when you need to execute service operations in a queued manner.

Question 42: What is MSMQ in WCF?

Answer: Microsoft Messaging Queue (MSMQ) technology is used for asynchronous communication using messages. MSMQ also can be considered to be an Inter- process communication capability. Whenever two processes want to communicate with each other in a "Fire and Forget" manner, MSMQ is very useful for that.

Example: For example what if Billing software needs to process 1000 bills at midnight and must send mail to all those users. Such as when the operator runs the software he wants immediate notification. The operator can't wait for all the bills to be processed and gets email.

Here MSMQ plays an important role for communication by billing software to send those 1000's of customer email information into the Queue and the Queue event handler will handle the requests. In this way the Operator gets instant notification of the processes instead of knowing the "Behind the Scene" process.

Question 43: What is Method overloading in WCF?

Answer: Method Overloading is a feature that allows creation of multiple methods with the same name but each method should differ from another in the context of input and output of the function. No two methods must have the same type parameters. Either they must differ in the number of parameters or in the type of the parameters.

You can also refer to Method Overloading as a compile-time polymorphism since the calling method knows the address of the called method and the method addresses are resolved at compile time. This is also called as Early Binding.

Example:

```
1.  [ServiceContract]
2.  public interface ICalenderService
3.  {
4.     [OperationContract(Name = "GetScheduledEventsByDate")]
5.     ScheduledEvent[] GetScheduledEvents(DateTime date);
6.     [OpeartionContract(Name = "GetScheduledEventsByDateRange")]
7.     ScheduledEvent[] GetScheduledEvents(DateTime start, DateTime end);
8.  }
```

Question 44: What is WCF Messaging Layer?

Answer: Messaging layer is composed of channels. A channel is a component that processes a message in some way. Channels are the core abstraction for sending messages to and receiving messages from an Endpoint.

Message Structure:

WCF uses messages to pass data or exchange information from one point to another. All messages are SOAP messages. The basic structures of a SOAP message are made up of three components.

SOAP Envelope:

The SOAP envelope is a container for the two most important pieces of a SOAP message, the SOAP header and the SOAP body. A SOAP envelope contains several pieces of key information in the form of elements.

SOAP Header:

Using a SOAP header, we can pass useful information about the services to the outer world if needed; it's just for information sharing. Any child elements of the header element are called "header blocks". This provides a mechanism for grouping logical data together.

SOAP Body:

This element contains the actual SOAP message for communication with the SOAP receiver; a message can contain zero or more bodies. Any information intended to be exchanged when the message reaches the intended destination goes in the message body.

SOAP Envelope

namespace

```
<school:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
         xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing">
```

SOAP Header `<school:Header>`

This section will contain school information's.

`</school:Header>`

SOAP Body `<school:Body>`

This part contains the response for the requests made by client.

`</school:Body>`
`</school:Envelope>`
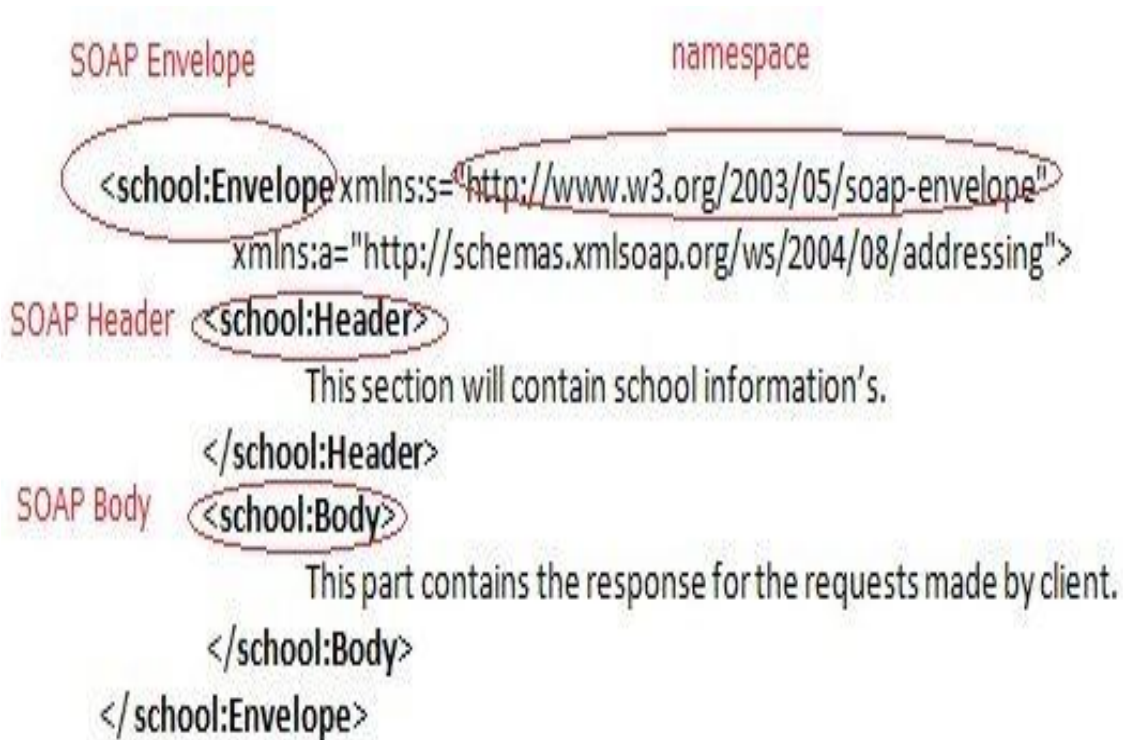
Question 46: What is Duplex in WCF? Explain also Message Exchange Pattern?

Answer: Duplex - Duplex is a Two-Way Communication Process. In which, a consumer send a message to the service and the service sends a notification that the request processing has been done. It is just like that we send some command about the printing of some papers to the Printer machine. The printer machine start communication by making the connection with our machine, and start printing the paper.

1. Duplex service allows calling back some operation (function) on the client.
2. Duplex service also knows as Call Backs.
3. All Binding does not support duplex service.
4. Duplex communication is not standard. They are absolute Microsoft feature.
5. wsDualHttpBinding supports duplex communication over HTTP binding.
6. Duplex communication is possible over netTcpBinding and netNamedPipeBinding

Example:

1. [ServiceContract(CallbackContract = typeof (DuplexServiceCallback))]
2. public interface IMyService
3. {
4.   [OperationContract(IsOneWay = true)] //one-way message
5.   void AddName(string name);
6.   [OperationContract] //request-response message
7.   string GetName(string name);
8. }

Message Exchange Patterns in WCF - The Message Exchange Pattern (MEP) provides a way of communication between client and server. MEP is a beauty of WCF. A services do some task depend on us or send a response to our request. The communication between client and server are done in form of messages. When we send a message as request and get a message as response from the service. The WCF Services support three types of Message Exchange Pattern:

- Request-Response
- One-Way
- Duplex

Why we need the Message Exchange Pattern:

1. When we want the calling of the function to not take more time.
2. We just want to call the function and not want wait for the response.
3. When we want that function call is not blocking and at the same time we want something out of the function (response) for our application.
4. When we want a two-way communication mechanism.

Question 47: What you understand by Fault Exception in WCF?

Answer: A Fault Contract is a way to handle an error/exception in WCF. In C# we can handle the error using try and catch blocks at the client side. The purpose of a Fault Contract is to handle an error by the service class and display in the client side. Whenever the client makes a call to a service class and an unexpected exception comes such as, SQL Server is not responding, or Divide by zero; in this case the service class throws an exception to the client using the Fault Contract.

By default when we throw an exception from a service, it will not reach the client. WCF provides the option to handle and convey the error message to the client from a service using a SOAP Fault Contract.

To support SOAP Faults, WCF provides the FaultException class. This class has two forms:

1. **FaultException:** to send an untyped fault back to the caller.
2. **FaultException<T>:** to send a typed fault data to the client. It is basically a Generic Type.

Syntax:
[OperationContract]
[FaultContract(typeof(MyException))]
string getDetails(int value);

The Fault Contract sample demonstrates how to communicate error information from a service to a client. The sample is based with some additional code added to the servie to convert an internal exception to a fault.

Question 48: What is the concept of tracelevel in trace listeners?

Answer: To enable the **wcf trace** we need to mention the trace source in configuration file like Web.config. Mostly and widely used trace source is **System.ServiceModel.** Another trace source which is mostly used for WCF Message logging is **System.ServiceModel.MessageLogging.**

WCF tracing provides us different **wcf tracing levels** as given below:

Sr. No	Trace Level	Event Category	Tracing Details	Traced Events
1	Off	-	– (No Tracing)	Ignores tracing.
2	Critical	Negative Events	Unexpected events or any error condition. When occurs, application may shut shortly or immediately.	All unexpected events OutOfMemoryException ThreadAbortException StackOverFlowException ConfigurationErrorsException SEHException Application start errors System Hang errors All other traces that cause application to terminate
3	Error	Negative Events	Unexpected events or any error condition. When occurs, application does not shut, it remains up & continues processing.	Traces all exceptions.
4	Warning	Negative Events	Indicates possibility of problems. Application remains up and continues the processing.	Traces all possible problems that are occurred or may occur. 1) TimeOut has exceeded 2) Rejected credentials 3) Receiving more request than expected. 4) Receiving Queue is nearby maximum capacity.
5	Information	Positive Events	Only informative traces that tells the successful milestone of application execution.	Traces all diagnostic data.
6	Verbose	Positive Events	Same as "Information", however only difference is – it allows low level tracing.	Mostly used for Debugging purpose.
7	ActivityTracing	-	The traces are related to communication between components and activities.	-
8	All	Positive & Negative	It includes all above events	It includes all above events

Question 49: What is Instance Management in WCF?

Answer: Instance Management is the name for a set of techniques used by WCF to bind client requests to service instances, governing which service instance handles which client request. This is a service side implementation detail.

In WCF, Instance Management is a technique to decide which service instance will serve a client request. It also decides when a client request will be served by a service instance. Instance Management is a very important factor when designing a service. Its affects:

- Scalability of Service
- Performance of Service
- Durability of Service
- Transactions
- Service Queues

Instance Management can be seen as a set of techniques to decide which service instance will serve a client request and when.

Basically there are three instance modes in WCF:

- Per-Call instance mode
- Per-Session instance mode
- Singleton Instance Mode

Why we need Instance Management?

We need Instance Management because applications differ too much in their needs for scalability, performance, throughput, transactions and queued calls. So one solution for all applications doesn't fit.
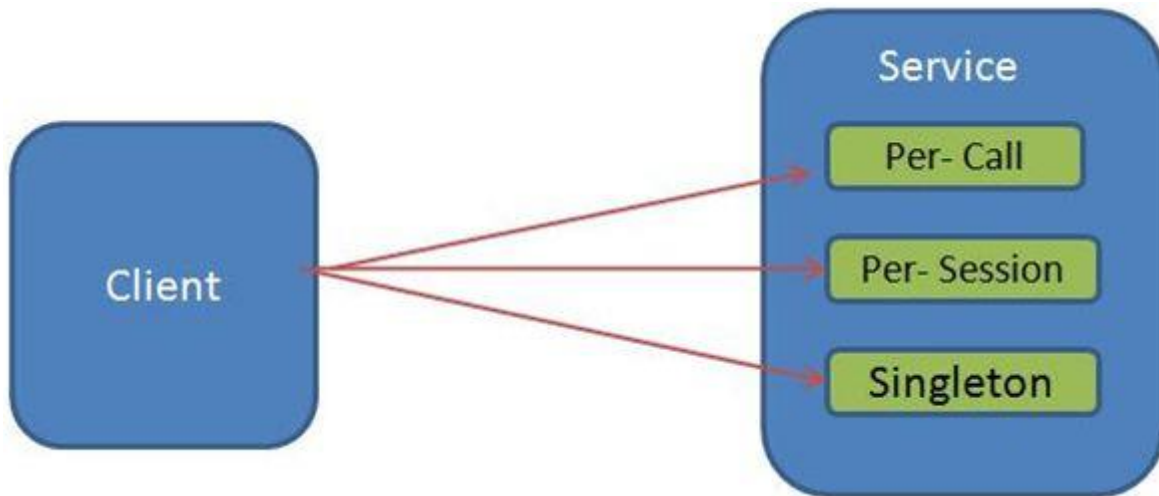
- Per-Call
  - Per Call service allocates and destory a new SERVICE INSTANCE for each client request.

- Per-Sesion
  - SessionFul service allocates a SERVICE INSTANCE for each client connection.

- Single
  - Single service shares same SERVICE INSTANCE for all clients across all connection and activation.

Question 50: What is Information cards in WCF?

Answer: As we are using DerivativesCalculator sample to enhance the WCF application hence we can use it in very interesting way to add an Information Card to a WCF application. It provides more benefits than CardSpace.

Benefits of adding Information Card:

- Authentication
- Authorization
- Reduce IT pain
- Granting trust to domains
- Helps in maintaining security

Information cards are virtual representations of a person's identity that are assured by a particular party. Information cards are analogous to real-world identity cards such as passports, driver's licenses, credit cards, and employee ID cards.

Sample Code: To show the usage of DerivativesCalculator which help in adding Information card which require no external reference to do so.

```
1.  namespace DerivativesCalculator
2.  {
3.      public class Calculator
4.      {
5.          public decimal CalculateDerivative(string[] symbols, string[] parameters, string[] functions)
6.          {
7.              return (decimal)(System.DateTime.Now > Millisecond);
8.          }
9.      }
10. }
```

- Information cards are more flexible than simple user names and passwords.
- Information cards employ strong cryptography, which makes their use more secure than passwords.
- Information cards can potentially present any type of identity claim that makes sense to all of the interacting parties and which users are willing to release.

Chapter 6

Most Asked ASP.NET MVC Interview Questions and Answers

Question 1: What is MVC (Model view controller)?

Answer: Model–view–controller (MVC) is a software architectural pattern for implementing user interfaces. It divides a given software application into three interconnected parts, so as to separate internal representation of information from the way that information is presented to or accepted from the user.

MVC is a framework for building web applications using a MVC (Model View Controller) design:

- The Model represents the application core (for instance a list of database records).
- The View displays the data (the database records).
- The Controller handles the input (to the database records).

The MVC model also provides full control over HTML, CSS, and JavaScript.
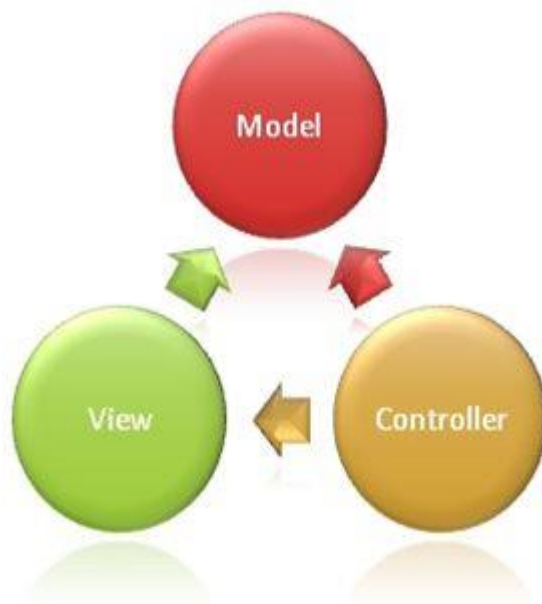
The MVC model defines web applications with 3 logic layers:

- The business layer (Model logic)
- The display layer (View logic)
- The input control (Controller logic)

The Model is the part of the application that handles the logic for the application data. Often model objects retrieve data (and store data) from a database.

The View is the part of the application that handles the display of the data. Most often the views are created from the model data.

The Controller is the part of the application that handles user interaction. Typically controllers read data from a view, control user input, and send input data to the model.

The MVC separation helps you manage complex applications, because you can focus on one aspect a time. For example, you can focus on the view without depending on the business logic. It also makes it easier to test an application.

Question 2: What are the advantages of MVC?

Answer: Advantages of MVC-

Multiple view support - Due to the separation of the model from the view, the user interface can display multiple views of the same data at the same time.

Change Accommodation - User interfaces tend to change more frequently than business rules (different colors, fonts, screen layouts, and levels of support for new devices such as cell phones or PDAs) because the model does not depend on the views, adding new a type of views to the system generally does not affect the model. As a result, the scope of change is confined to the view.

SoC – Separation of Concerns - Separation of Concerns is one of the core advantages of ASP.NET MVC. The MVC framework provides a clean separation of the UI, Business Logic, Model or Data.

More Control - The ASP.NET MVC framework provides more control over HTML, JavaScript and CSS than the traditional Web Forms.

Testability - ASP.NET MVC framework provides better testability of the Web Application and good support for the test driven development too.

Lightweight - ASP.NET MVC framework doesn't use View State and thus reduces the bandwidth of the requests to an extent.

Full features of ASP.NET - One of the key advantages of using ASP.NET MVC is that it is built on top of ASP.NET framework and hence most of the features of the ASP.NET like membership providers, roles, etc can still be used.

Other Code

MVC Code
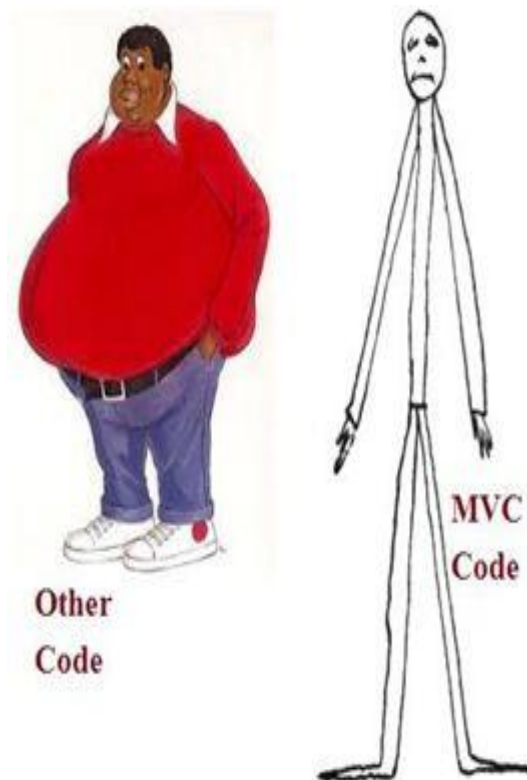
Question 3: Explain MVC application life cycle?

Answer: Any web application has two main execution steps, first understanding the request and depending on the type of the request sending out appropriate response. MVC application life cycle is not different it has two main phases, first creating the request object and second sending our response to the browser.

Creating the request object: The request object creation has four major steps. The following is the detailed explanation of the same:

Step 1: Fill route

MVC requests are mapped to route tables which in turn specify which controller and action to be invoked. So if the request is the first request the first thing is to fill the route table with routes collection. This filling of route table happens in the global.asax file.

Step 2: Fetch route

Depending on the URL sent "UrlRoutingModule" searches the route table to create "RouteData" object which has the details of which controller and action to invoke.

Step 3: Request context created

The "RouteData" object is used to create the "RequestContext" object.

Step 4: Controller instance created

This request object is sent to "MvcHandler" instance to create the controller class instance. Once the controller class object is created it calls the "Execute" method of the controller class.

Creating Response object: This phase has two steps executing the action and finally sending the response as a result to the view.
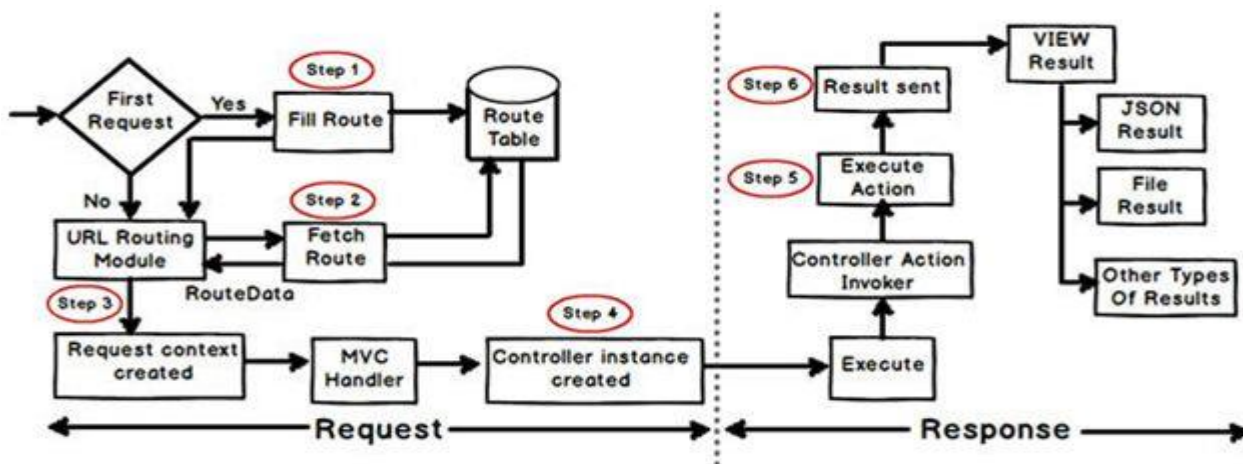
Question 4: List out different return types of a controller action method?

Answer: There are total nine return types we can use to return results from controller to view. The base type of all these result types is ActionResult.

1. **ViewResult (View)**: This return type is used to return a webpage from an action method.
2. **PartialviewResult (Partialview)**: This return type is used to send a part of a view which will be rendered in another view.
3. **RedirectResult (Redirect)**: This return type is used to redirect to any other controller and action method depending on the URL.
4. **RedirectToRouteResult (RedirectToAction, RedirectToRoute):** This return type is used when we want to redirect to any other action method.
5. **ContentResult (Content)**: This return type is used to return HTTP content type like text/plain as the result of the action.
6. **jsonResult (json):** This return type is used when we want to return a JSON message.
7. **javascriptResult (javascript):** This return type is used to return JavaScript code that will run in browser.
8. **FileResult (File):** This return type is used to send binary output in response.
9. **EmptyResult:** This return type is used to return nothing (void) in the result.

Question 5: What are Filters in MVC?

Answer: In MVC, controllers define action methods and these action methods generally have a one-to-one relationship with UI controls such as clicking a button or a link, etc. For example, in one of our previous examples, the UserController class contained methods UserAdd, UserDelete, etc. But many times we would like to perform some action before or after a particular operation. For achieving this functionality, ASP.NET MVC provides feature to add pre and post action behaviors on controller's action methods.

Types of Filters: ASP.NET MVC framework supports the following action filters:

- **Action Filters:** Action filters are used to implement logic that gets executed before and after a controller action executes. We will look at Action Filters in detail in this chapter.
- **Authorization Filters:** Authorization filters are used to implement authentication and authorization for controller actions.
- **Result Filters:** Result filters contain logic that is executed before and after a view result is executed. For example, you might want to modify a view result right before the view is rendered to the browser.
- **Exception Filters:** Exception filters are the last type of filter to run. You can use an exception filter to handle errors raised by either your controller actions or controller action results. You can also use exception filters to log errors.

Action filters are one of most commonly used filters to perform additional data processing, or manipulating the return values or cancelling the execution of action or modifying the view structure at run time.

Question 6: What are Action Filters in MVC?

Answer: Action Filters are additional attributes that can be applied to either a controller section or the entire controller to modify the way in which action is executed. These attributes are special .NET classes derived from System.Attribute which can be attached to classes, methods, properties and fields.

ASP.NET MVC provides the following action filters:

Output Cache: This action filter caches the output of a controller action for a specified amount of time.

Handle Error: This action filter handles errors raised when a controller action executes.

Authorize: This action filter enables you to restrict access to a particular user or role.

Now we will see the code example to apply these filters on an example controller ActionFilterDemoController. (ActionFilterDemoController is just used as an example. You can use these filters on any of your controllers.)

Output Cache:

E.g.: Specifies the return value to be cached for 10 seconds.

```
1.  publicclassActionFilterDemoController: Controller
2.  {
3.    [HttpGet]
4.    OutputCache(Duration = 10)]
5.  publicstringIndex()
6.  {
7.    returnDateTime.Now.ToString("T");
8.
9.  }
10. }
```

Question 7: Explain what is routing in MVC? What are the three segments for routing important?

Answer: Routing is a mechanism to process the incoming url that is more descriptive and give desired response. In this case, URL is not mapped to specific files or folder as was the case of earlier day's web sites.

There are two types of routing (after the introduction of ASP.NET MVC 5).

1. **Convention based routing:** to define this type of routing, we call MapRoute method and set its unique name, URL pattern and specify some default values.
2. **Attribute based routing:** to define this type of routing, we specify the Route attribute in the action method of the controller.

Routing is the URL pattern that is mappped together to a handler,rounting is responsible for incoming browser request for particular MVC controller. In other ways let us say routing help you to define a URL structure and map the URL with controller. There are three segments for routing that are important,

1. ControllerName
2. ActionMethodName
3. Parammeter

i.e: ControllerName/ActionMethodName/{ParamerName} and also route map coding written in a Global.asax file.

Question 8: What is Route in MVC? What is Default Route in MVC?

Answer: A route is a URL pattern that is mapped to a handler. The handler can be a physical file, such as an .aspx file in a Web Forms application. A handler can also be a class that processes the request, such as a controller in an MVC application. To define a route, you create an instance of the Route class by specifying the URL pattern, the handler, and optionally a name for the route.

You add the route to the application by adding the Route object to the static Routes property of the RouteTable class. The Routesproperty is a RouteCollection object that stores all the routes for

the application.

You typically do not have to write code to add routes in an MVC application. Visual Studio project templates for MVC include preconfigured URL routes. These are defined in the Mvc Application class, which is defined in the Global.asax file.

Route definition	Example of matching URL
{controller}/{action}/{id}	/Products/show/beverages
{table}/Details.aspx	/Products/Details.aspx
blog/{action}/{entry}	/blog/show/123
{reporttype}/{year}/{month}/{day}	/sales/2008/1/5
{locale}/{action}	/US/show
{language}-{country}/{action}	/en-US/show

Default Route:

The default ASP.NET MVC project templates add a generic route that uses the following URL convention to break the URL for a given request into three named segments.

URL: "{controller}/{action}/{id}"
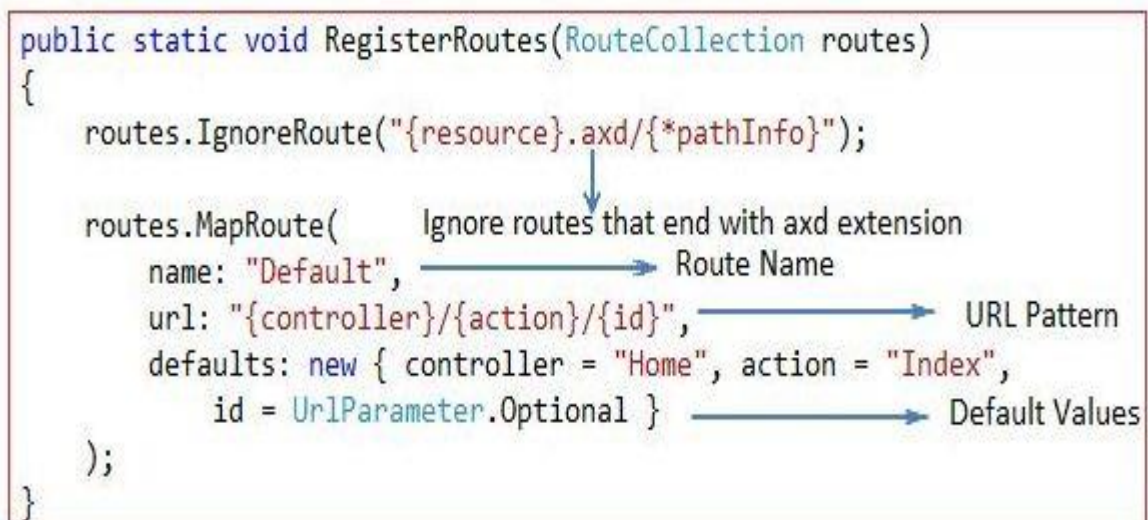
This route pattern is registered via call to the **MapRoute()** extension method of **RouteCollection**.

```
public static void RegisterRoutes(RouteCollection routes)
{
    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
                                    Ignore routes that end with axd extension
    routes.MapRoute(
        name: "Default",                        Route Name
        url: "{controller}/{action}/{id}",                  URL Pattern
        defaults: new { controller = "Home", action = "Index",
            id = UrlParameter.Optional }                Default Values
    );
}
```

Question 9: Mention what is the difference between Temp data, View, and View Bag?

Answer: In ASP.NET MVC there are three ways to pass/store data between the controllers and views.

ViewData:

1. ViewData is used to pass data from controller to view.
2. It is derived from ViewDataDictionary class.
3. It is available for the current request only.
4. Requires typecasting for complex data type and checks for null values to avoid error.
5. If redirection occurs, then its value becomes null.

ViewBag:

1. ViewBag is also used to pass data from the controller to the respective view.
2. ViewBag is a dynamic property that takes advantage of the new dynamic features in C# 4.0
3. It is also available for the current request only.
4. If redirection occurs, then its value becomes null.
5. Doesn't require typecasting for complex data type.

TempData:

1. TempData is derived from TempDataDictionary class
2. TempData is used to pass data from the current request to the next request
3. It keeps the information for the time of an HTTP Request. This means only from one page to another. It helps to maintain the data when we move from one controller to another controller or from one action to another action
4. It requires typecasting for complex data type and checks for null values to avoid error. Generally, it is used to store only one time messages like the error messages and validation messages

Question 10: What is Partial View in MVC?

Answer: A partial view is a chunk of HTML that can be safely inserted into an existing DOM. Most commonly, partial views are used to componentize Razor views and make them easier to build and update. Partial views can also be returned directly from controller methods. In this case, the browser still receives text/html content but not necessarily HTML content that makes up an entire page. As a result, if a URL that returns a partial view is directly invoked from the address bar of a browser, an incomplete page may be displayed. This may be something like a page that misses title, script and style sheets.

However, when the same URL is invoked via script, and the response is used to insert HTML within the existing DOM, then the net effect for the end user may be much better and nicer.

Partial view is a reusable view (like a user control) which can be embedded inside other view. For example, let's say all the pages of your site have a standard structure with left menu, header, and footer as in the following image,
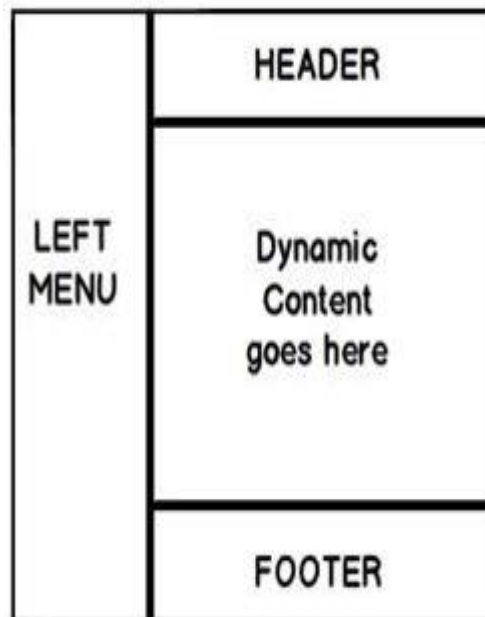
```
@model IEnumerable<MvcApplication8.Models.Product>

@{
    ViewBag.Title = "Index";
}

<h2>Products</h2>

<table>
    <tr>
        <th>
            Name
        </th>
        <th>
            Description
        </th>
        <th>
            Price
        </th>
        <th>
            EDD
        </th>
    </tr>

@foreach (var item in Model) {
    <tr>
        <td>
            @item.Name
        </td>
        <td>
            @item.Description
        </td>
        <td>
            @(item.Price/10)
        </td>
        <td>
            @Html.ActionLink("Edit", "Edit", new { id=item.ProductId }) |
            @Html.ActionLink("Details", "Details", new { id=item.ProductId }) |
            @Html.ActionLink("Delete", "Delete", new { id=item.ProductId })
        </td>
    </tr>
}
</table>
```

Let's make this piece of code reusable

Question 11: Explain what is the difference between View and Partial View?

Answer:

View:

- It contains the layout page.
- Before any view is rendered, viewstart page is rendered.
- View might have markup tags like body, html, head, title, meta etc.
- View is not lightweight as compare to Partial View.

Partial View:

- It does not contain the layout page.
- Partial view does not verify for a viewstart.cshtml.We cannot put common code for a partial view within the viewStart.cshtml.page.
- Partial view is designed specially to render within the view and just because of that it does not consist any mark up.
- We can pass a regular view to the RenderPartial method.

Question 12: What are HTML helpers in MVC?

Answer: With MVC, HTML helpers are much like traditional ASP.NET Web Form controls. Just like web form controls in ASP.NET, HTML helpers are used to modify HTML. But HTML helpers are more lightweight. Unlike Web Form controls, an HTML helper does not have an event model and a view state. In most cases, an HTML helper is just a method that returns a string. With MVC, you can create your own helpers, or use the built in HTML helpers.

Standard HTML Helpers:

HTML Links: The easiest way to render an HTML link in is to use the HTML.ActionLink() helper.With MVC, the Html.ActionLink() does not link to a view. It creates a link to a controller action.

ASP Syntax:

1. <%=Html.ActionLink("About this Website", "About")%>

The first parameter is the link text, and the second parameter is the name of the controller action. The Html.ActionLink() helper above, outputs the following HTML:

1. About this Website

The Html.ActionLink() helper has several properties:

Property	Description
.linkText	The link text (label)
.actionName	The target action
.routeValues	The values passed to the action
.controllerName	The target controller
.htmlAttributes	The set of attributes to the link
.protocol	The link protocol
.hostname	The host name for the link
.fragment	The anchor target for the link

HTML Form Elements:

There following HTML helpers can be used to render (modify and output) HTML form elements:

- BeginForm()
- EndForm()
- TextArea()
- TextBox()
- CheckBox()
- RadioButton()
- ListBox()
- DropDownList()
- Hidden()
- Password()

ASP.NET Syntax C#:

```
1.  <%= Html.ValidationSummary("Create was unsuccessful. Please correct the errors and try
    again.") %>
2.    <% using (Html.BeginForm()){%>
3.      <p>
4.        <label for="FirstName">First Name:</label>
5.      <%= Html.TextBox("FirstName") %>
6.        <%= Html.ValidationMessage("FirstName", "*") %>
7.      </p>
8.      <p>
9.        <label for="LastName">Last Name:</label>
10.     <%= Html.TextBox("LastName") %>
11.       <%= Html.ValidationMessage("LastName", "*") %>
12.     </p>
13.     <p>
```

```
14.        <label for="Password">Password:</label>
15.        <%= Html.Password("Password") %>
16.          <%= Html.ValidationMessage("Password", "*") %>
17.      </p>
18.      <p>
19.        <label for="Password">Confirm Password:</label>
20.        <%= Html.Password("ConfirmPassword") %>
21.          <%= Html.ValidationMessage("ConfirmPassword", "*") %>
22.      </p>
23.      <p>
24.        <label for="Profile">Profile:</label>
25.        <%= Html.TextArea("Profile", new {cols=60, rows=10})%>
26.      </p>
27.      <p>
28.        <%= Html.CheckBox("ReceiveNewsletter") %>
29.          <label for="ReceiveNewsletter" style="display:inline">Receive Newsletter?</la
    bel>
30.      </p>
31.      <p>
32.        <input type="submit" value="Register" />
33.      </p>
34.      <% }%>
```

Question 13: Explain attribute based routing in MVC?

Answer: In ASP.NET MVC 5.0 we have a new attribute route, By using the "Route" attribute we can define the URL structure. For example in the below code we have decorated the "GotoAbout" action with the route attribute. The route attribute says that the "GotoAbout" can be invoked using the URL structure "Users/about".

Code:

```
1.  public class HomeController: Controller
2.  {
3.    [Route("Users/about")]
4.    publicActionResultGotoAbout()
5.    {
6.      return View();
7.    }
8.  }
```

Question 14: What is TempData in MVC?

Answer: TempData is a dictionary object to store data temporarily. It is a TempDataDictionary class type and instance property of the Controller base class. TempData is able to keep data for the duration of a HTP request, in other words it can keep live data between two consecutive HTTP requests. It will help us to pass the state between action methods. TempData only works with the current and subsequent request. TempData uses a session variable to store the data. TempData Requires type casting when used to retrieve data.

TempDataDictionary is inherited from the IDictionary<string, object>, ICollection<KeyValuePair<string, object>>, IEnumerable<KeyValuePair<string, object>> and IEnumerable interfaces.

Example:

```
1.  public ActionResult FirstRequest()
2.  {
3.      List < string > TempDataTest = new List < string > ();
4.      TempDataTest.Add("Tejas");
5.      TempDataTest.Add("Jignesh");
6.      TempDataTest.Add("Rakesh");
7.      TempData["EmpName"] = TempDataTest;
8.      return View();
9.  }
10. public ActionResult ConsecutiveRequest()
11. {
12.     List < string > modelData = TempData["EmpName"] as List < string > ;
13.     TempData.Keep();
14.     return View(modelData);
15. }
```

Question 15: What is Razor in MVC?

Answer: ASP.NET MVC has always supported the concept of "view engines" - which are the pluggable modules that implement different template syntax options. The "default" view engine for ASP.NET MVC uses the same .aspx/.ascx/. master file templates as ASP.NET Web Forms. Other popular ASP.NET MVC view engines are Spart&Nhaml.

MVC 3 has introduced a new view engine called Razor.

Why is Razor?

1. Compact & Expressive.

2. Razor minimizes the number of characters and keystroke required in a file, and enables a fast coding workflow. Unlike most template syntaxes, you do not need to interrupt your coding to explicitly denote server blocks within your HTML. The parser is smart enough to infer this from your code. This enables a really compact and expressive syntax which is clean, fast and fun to type.

3. **Easy to Learn:** Razor is easy to learn and enables you to quickly be productive with a minimum of effort. We can use all your existing language and HTML skills.

4. Works with any Text Editor: Razor doesn't require a specific tool and enables you to be productive in any plain old text editor (notepad works great).

5. Has great Intellisense.

6. **Unit Testable:** The new view engine implementation will support the ability to unit test views (without requiring a controller or web-server, and can be hosted in any unit test project - no special app-domain required).

Question 16: Differences between Razor and ASPX View Engine in MVC?

Answer: Razor View Engine VS ASPX View Engine:

Razor View Engine	ASPX View Engine (Web form view engine)
The namespace used by the Razor View Engine is System.Web.Razor	The namespace used by the ASPX View Engine is System.Web.Mvc.WebFormViewEngine
The file extensions used by the Razor View Engine are different from a web form view engine. It uses cshtml with C# and vbhtml with vb for views, partial view, templates and layout pages.	The file extensions used by the Web Form View Engines are like ASP.Net web forms. It uses the ASPX extension to view the aspc extension for partial views or User Controls or templates and master extensions for layout/master pages.
The Razor View Engine is an advanced view engine that was introduced with MVC 3.0. This is not a new language but it is markup.	A web form view engine is the default view engine and available from the beginning of MVC
Razor has a syntax that is very compact and helps us to reduce typing.	The web form view engine has syntax that is the same as an ASP.Net forms application.
The Razor View Engine uses @ to render server-side content.	The ASPX/web form view engine uses "<%= %>" or "<%: %>" to render server-side content.
By default all text from an @ expression is HTML encoded.	There is a different syntax ("<%: %>") to make text HTML encoded.
Razor does not require the code block to be closed, the Razor View Engine parses itself and it is able to decide at runtime which is a content element and which is a code element.	A web form view engine requires the code block to be closed properly otherwise it throws a runtime exception.
The Razor View Engine prevents Cross Site Scripting (XSS) attacks by encoding the script or HTML tags before rendering to the view.	A web form View engine does not prevent Cross Site Scripting (XSS) attack.
The Razor Engine supports Test Driven Development (TDD).	Web Form view engine does not support Test Driven Development (TDD) because it depends on the System.Web.UI.Page class to make the testing complex.
Razor uses "@* â€¦ *@" for multiline comments.	The ASPX View Engine uses "<!--...-->" for markup and "/* â€¦ */" for C# code.
There are only three transition characters with the Razor View Engine.	There are only three transition characters with the Razor View Engine.

The Razor View Engine is bit slower than the ASPX View Engine.

Conclusion:

Razor provides a new view engine with streamlined code for focused templating. Razor's syntax is very compact and improves readability of the markup and code. By default MVC supports ASPX (web forms) and Razor View Engine. MVC also supports third-party view engines like Spark, Nhaml, NDjango, SharpDOM and so on. ASP.NET MVC is open source.

Question 17: What are the Main Razor Syntax Rules?

Answer: There are following types of Razor syntax-

- Razor code blocks are enclosed in @{ ... }
- Inline expressions (variables and functions) start with @
- Code statements end with semicolon
- Variables are declared with the var keyword
- Strings are enclosed with quotation marks
- C# code is case sensitive
- C# files have the extension .cshtml

C# Example:

```
1.  <!-- Single statement block -->
2.  @ {
3.      varmyMessage = "Hello World";  }
4.  <!-- Inline expression or variable -->
5.  < p > The value of myMessage is: @myMessage < /p>
6.      <!-- Multi-statement block -->
7.  @ {
8.      var greeting = "Welcome to our site!";
9.      varweekDay = DateTime.Now.DayOfWeek;
10.     vargreetingMessage = greeting + " Here in Huston it is: " + weekDay;
11. }
12. < p > The greeting is: @greetingMessage < /p>
```

Question18: How do you implement Forms authentication in MVC?

Answer: Authentication is giving access to the user for a specific service by verifying his/her identity using his/her credentials like username and password or email and password. It assures that the correct user is authenticated or logged in for a specific service and the right service has been provided to the specific user based on their role that is nothing but authorization.

ASP.NET forms authentication occurs after IIS authentication is completed. You can configure forms authentication by using forms element with in web.config file of your application. The default attribute values for forms authentication are shown below:

1. <system.web>
2.   <authenticationmode="Forms">
3.   <formsloginUrl="Login.aspx" protection="All" timeout="30" name=".ASPXAUTH" path="/" requireSSL="false" slidingExpiration="true" defaultUrl="default.aspx" cookieless="UseDeviceProfile" enableCrossAppRedirects="false" />
4.   </authentication>
5. </system.web>

The FormsAuthentication class creates the authentication cookie automatically when SetAuthCookie() or RedirectFromLoginPage() methods are called. The value of authentication cookie contains a string representation of the encrypted and signed FormsAuthenticationTicket object.

Question 19: Explain Areas in MVC?

Answer: From ASP.Net MVC 2.0 Microsoft provided a new feature in MVC applications, Areas. Areas are just a way to divide or "isolate" the modules of large applications in multiple or separated MVC.
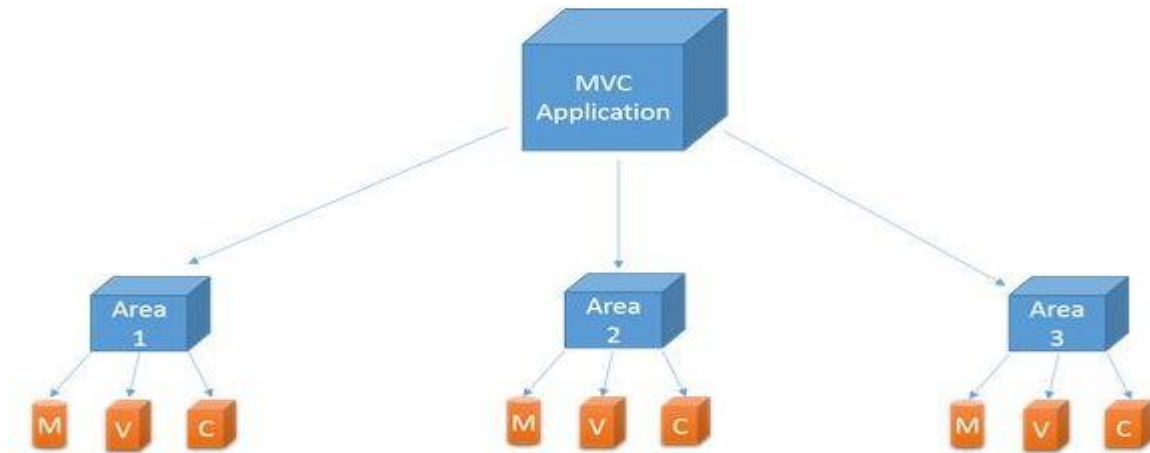
When you add an area to a project, a route for the area is defined in an AreaRegistration file. The route sends requests to the area based on the request URL. To register routes for areas, you add code to theGlobal.asax file that can automatically find the area routes in the AreaRegistration file.

AreaRegistration.RegisterAllAreas();

Benefits of Area in MVC:

1. Allows us to organize models, views and controllers into separate functional sections of the application, such as administration, billing, customer support and much more.
2. Easy to integrate with other Areas created by another.
3. Easy for unit testing.

Question 20: Explain the need of display mode in MVC?

Answer: DisplayModes give you another level of flexibility on top of the default capabilities we saw in the last section. DisplayModes can also be used along with the previous feature so we will simply build off of the site we just created.

Using display modes involves in 2 steps:

1. We should register Display Mode with a suffix for particular browser using "DefaultDisplayMode"e class inApplication_Start() method in the Global.asax file.
2. View name for particular browser should be appended with suffix mentioned in first step.

1. Desktop browsers (without any suffix. e.g.: Index.cshtml, _Layout.cshtml).
2. Mobile browsers (with a suffix "Mobile". e.g.:
   Index.Mobile.cshtml,Layout.Mobile.cshtml)

   If you want design different pages for different mobile device browsers (any different browsers) and render them depending on the browser requesting. To handle these requests you can register custom display modes. We can do that using DisplayModeProvider.Instance.Modes.Insert(int index, IDisplayMode item) method.

Question 21: Explain the concept of MVC Scaffolding?

Answer: ASP.NET Scaffolding is a code generation framework for ASP.NET Web applications. Visual Studio 2013 includes pre-installed code generators for MVC and Web API projects. You add scaffolding to your project when you want to quickly add code that interacts with data models. Using scaffolding can reduce the amount of time to develop standard data operations in your project.

Scaffolding consists of page templates, entity page templates, field page templates, and filter templates. These templates are called Scaffold templates and allow you to quickly build a functional data-driven Website.
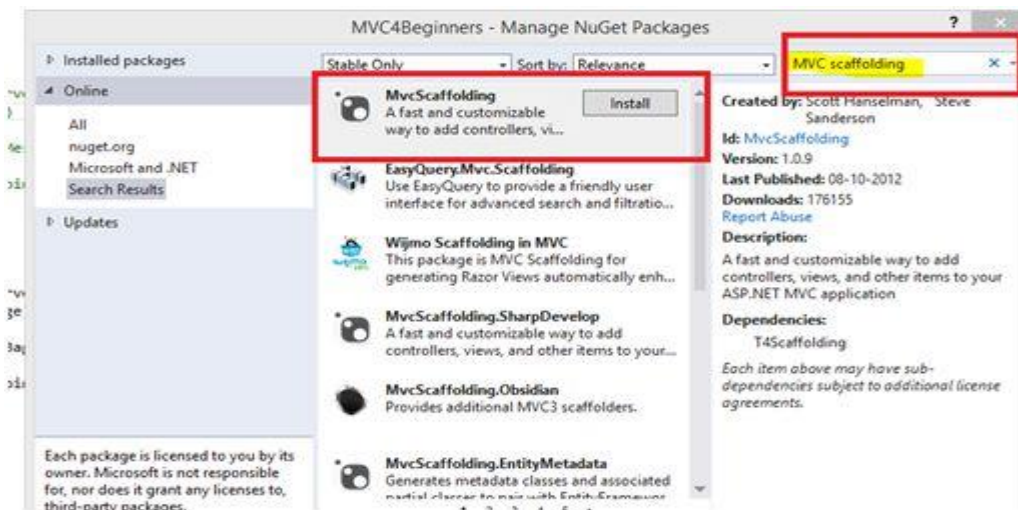
Scaffolding Templates:

Create: It creates a View that helps in creating a new record for the Model. It automatically generates a label and input field for each property in the Model.

Delete: It creates a list of records from the model collection along with the delete link with delete record.

Details: It generates a view that displays the label and an input field of the each property of the Model in the MVC framework.

Edit: It creates a View with a form that helps in editing the current Model. It also generates a form with label and field for each property of the model.

List: It generally creates a View with the help of a HTML table that lists the Models from the Model Collection. It also generates a HTML table column for each property of the Model.

Question 22: What is Route Constraints in MVC?

Answer: Routing is a great feature of MVC, it provides a REST based URL that is very easy to remember and improves page ranking in search engines.

This article is not an introduction to Routing in MVC, but we will learn a few features of routing and by implementing them we can develop a very flexible and user-friendly application. So, let's start without wasting valuable time.

Add constraint to URL:

This is very necessary for when we want to add a specific constraint to our URL. Say, for example we want a URL.

So, we want to set some constraint string after our host name. Fine, let's see how to implement it.

It's very simple to implement, just open the RouteConfig.cs file and you will find the routing definition in that. And modify the routing entry as in the following. We will see that we have added "abc" before.

```
17  |  --------------routes.MapRoute(
18  |  ............................."D.f..lt".
19  |  ..................url: "abc/{controller}/{action}/{id}",
20  |  ................uefauits: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
21  |  ..............);
22  |
23  |  .........}
```

Controller name, now when we browse we need to specify the string in the URL, as in the following:

localhost:2968/abc/person/

I am simple view.

Question 23: What is Razor View Engine in MVC?

Answer: ASP.NET MVC has always supported the concept of "view engines" that are the pluggable modules that implement various template syntax options. The "default" view engine for ASP.NET MVC uses the same .aspx/.ascx/.master file templates as ASP.NET Web Forms. In this article I go through the Razor View Engine to create a view of an application. "Razor" was in development beginning in June 2010 and was released for Microsoft Visual Studio in January 2011.

Razor is not a new programming language itself, but uses C# syntax for embedding code in a page without the ASP.NET delimiters: <%= %>. It is a simple-syntax view engine and was released as part of ASP.NET MVC 3. The Razor file extension is "cshtml" for the C# language. It supports TDD (Test Driven Development) because it does not depend on the System.Web.UI.Page class.

Question 24: What is Output Caching in MVC?

Answer: The main purpose of using Output Caching is to dramatically improve the performance of an ASP.NET MVC Application. It enables us to cache the content returned by any controller method so that the same content does not need to be generated each time the same controller method is invoked. Output Caching has huge advantages, such as it reduces server round trips, reduces database server round trips, reduces network traffic etc.

Keep the following in mind:

- Avoid caching contents that are unique per user.
- Avoid caching contents that are accessed rarely.
- Use caching for contents that are accessed frequently.

My MVC application displays a list of database records on the view page so by default each time the user invokes the controller method to see records, the application loops through the entire process and executes the database query. And this can actually decrease the application performance. So, we can advantage of the "Output Caching" that avoids executing database queries each time the user invokes the controller method. Here the view page is retrieved from the cache instead of invoking the controller method and doing redundant work.

Cached Content Locations:

In the above paragraph I said, in Output Caching the view page is retrieved from the cache, so where is the content cached/stored?

Please note, there is no guarantee that content will be cached for the amount of time that we specify. When memory resources become low, the cache starts evicting content automatically.

OutputCache label has a "Location" attribute and it is fully controllable. Its default value is "Any", however there are the following locations available; as of now, we can use any one.

1. Any
2. Client
3. Downstream
4. Server
5. None
6. ServerAndClient

With "Any", the output cache is stored on the server where the request was processed. The recommended store cache is always on the server very carefully. You will learn about some security related tips in the following "Don't use Output Cache".

Question 25: What is Bundling and Minification in MVC?

Answer: Bundling and minification are two new techniques introduced to improve request load time. It improves load time by reducing the number of requests to the server and reducing the size of requested assets (such as CSS and JavaScript).

Bundling: It lets us combine multiple JavaScript (.js) files or multiple cascading style sheet (.css) files so that they can be downloaded as a unit, rather than making individual HTTP requests.

Minification: It squeezes out whitespace and performs other types of compression to make the downloaded files as small as possible. At runtime, the process identifies the user agent, for example IE, Mozilla, etc. and then removes whatever is specific to Mozilla when the request comes from IE.

Question 26: What is Validation Summary in MVC?

Answer: The ValidationSummary helper method generates an unordered list (ul element) of validation messages that are in the ModelStateDictionary object.

The ValidationSummary can be used to display all the error messages for all the fields. It can also be used to display custom error messages. The following figure shows how ValidationSummary displays the error messages.

ValidationSummary() Signature:

MvcHtmlStringValidateMessage(bool excludePropertyErrors, string message, object htmlAttributes)

Display field level error messages using ValidationSummary:

By default, ValidationSummary filters out field level error messages. If you want to display field level error messages as a summary then specify *excludePropertyErrors = false.*

Example: ValidationSummary to display field errors:

@Html.ValidationSummary(false, "", new { @class = "text-danger" })

So now, the following Edit view will display error messages as a summary at the top. Please make sure that you don't have a ValidationMessageFor method for each of the fields.

Question 27: What is Database First Approach in MVC using Entity Framework?

Answer: Database First Approach is an alternative to the Code First and Model First approaches to the Entity Data Model which creates model codes (classes, properties, DbContextetc) from the database in the project and that class behaves as the link between database and controller.

There are the following approachs which is used to connect with database to application.

- Database First
- Model First
- Code First
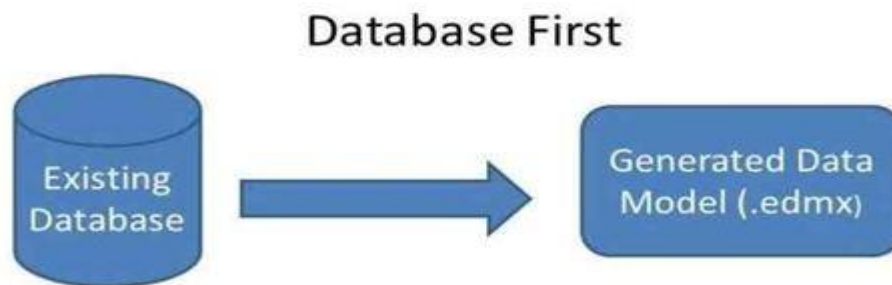
Database first is nothing but only an approach to create web application where database is available first and can interact with the database. In this database, database is created first and after that we manage the code. The Entity Framework is able to generate a business model based on the tables and columns in a relational database.

Question 28: What are the Folders in MVC application solutions?

Answer: Understanding the folders:

When you create a project a folder structure gets created by default under the name of your project which can be seen in solution explorer. Below i will give you a brief explanation of what these folders are for.

Model: This folder contains classes that are used to provide data. These classes can contain data that is retrived from the database or data inserted in the form by the user to update the database.

Controllers: These are the classes which will perform the action invoked by the user. These classes contain methods known as "Actions" which responds to the user action accordingly.

Views: These are simple pages which use the model class data to populate the HTML controls and render it to the client browser.

App_Start: Contains Classes such as FilterConfig, RoutesConfig, WebApiConfig. As of now we need to understand the RouteConfig class. This class contains the default format of the url that should be supplied in the browser to navigate to a specified page.

Question 29: What is difference between MVC and Web Forms?

Answer: ASP.Net MVC / Web Forms differences:

The following are some difference-

ASP.Net MVC	ASP.Net Web Forms
View and logic are separate; it has separation of concerns theory. MVC 3 onwards has .aspx page as .cshtml.	No separation of concerns; Views are tightly coupled with logic (.aspx.cs /.vb file).
Introduced concept of routing for route based URL. Routing is declared in Global.asax.	File-based routing .Redirection is based on pages.
Support Razor syntax as well as .aspx	Support web forms syntax only.
State management handled via Tempdata, ViewBag, and View Data. Since the controller and view are not dependent and also since there is no view state concept in ASP.NET, MVC keeps the pages lightweight.	State management handled via View State. Large viewstate, in other words increase in page size.
Partial Views	User Controls
HTML Helpers	Server Controls
Multiple pages can have the same controller to satisfy their requirements. A controller may have multiple Actions (method name inside the controller class).	Each page has its own code, in other words direct dependency on code. For example Sachin.aspx is dependent on Sachin.aspx.cs (code behind) file.
Unit Testing is quite easier than ASP.Net Web forms Since a web form and code are separate files.	Direct dependency, tight coupling raises issues in testing.
layouts	Master pages

Question 30: What are the methods of handling an Error in MVC?

Answer: Exception handling may be required in any application, whether it is a web application or a Windows Forms application. ASP.Net MVC has an attribute called "HandleError" that provides built-in exception filters. The HandleError attribute in ASP.NET MVC can be applied over the action method as well as Controller or at the global level. The HandleError attribute is the default implementation of IExceptionFilter. When we create a MVC application, the HandleError attribute is added within the Global.asax.cs file and registered in the Application_Start event.

```
1.  public static void RegisterGlobalFilters(GlobalFilterCollection filters)
2.  {
3.      filters.Add(new HandleErrorAttribute());
4.  }
5.  protected void Application_Start()
6.  {
7.      AreaRegistration.RegisterAllAreas();
8.      RegisterGlobalFilters(GlobalFilters.Filters);
9.      RegisterRoutes(RouteTable.Routes);
10. }
```

Important properties of HandleError attribute: The HandleError Error attribute has a couple for properties that are very useful in handling the exception.

ExceptionType: Type of exception to be catch. If this property is not specified then the HandleError filter handles all exceptions.

View: Name of the view page for displaying the exception information.

Master: Master View for displaying the exception.

Order: Order in which the action filters are executed. The Order property has an integer value and it specifies the priority from 1 to any positive integer value. 1 means highest priority and the greater the value of the integer is, the lower is the priority of the filter.

AllowMultiple: It indicates whether more than one instance of the error filter attribute can be specified.

Example:

```
1.  [HandleError(View = "Error")]
2.  public class HomeController: Controller
3.  {
4.     public ActionResult Index()
5.     {
6.        ViewBag.Message = "Welcome to ASP.NET MVC!";
7.        int u = Convert.ToInt32(""); // Error line
8.        return View();
9.     }
10. }
```

HandleError Attribute at Action Method Level,

```
1.  [HandleError(View = "Error")]
2.  public ActionResult Index()
3.  {
4.     ViewBag.Message = "Welcome to ASP.NET MVC!";
5.     int u = Convert.ToInt32(""); // Error line
6.     return View();
7.  }
```

Question 31: How can we pass the data From Controller to View in MVC?

Answer: There are three options in Model View Controller (MVC) for passing data from controller to view. This article attempts to explain the differences among ViewData, ViewBag and TempData with examples. ViewData and ViewBag are similar and TempData performs additional responsibility. The following are the key points on those three objects.

ViewData

- The ViewData is used to move data from controller to view.
- The ViewData is a dictionary of objects that are derived from the "ViewDataDictionary" class and it will be accessible using strings as keys.
- ViewData contains a null value when redirection occurs.
- ViewData requires typecasting for complex data types.

ViewBag

- ViewBag is just a dynamic wrapper around ViewData and exists only in ASP.NET MVC 3. ViewBag is a dynamic property that takes advantage of the new dynamic features in C# 4.0.
- ViewBag doesn't require typecasting for complex data types.
- ViewBag also contain a null value when redirection occurs.

TempData

- ViewData moves data from controller to view.
- Use TempData when you need data to be available for the next request, only. In the next request, it will be there but will be gone after that.
- TempData is used to pass data from the current request to the subsequent request, in other words in case of redirection. That means the value of TempData will not be null.

Question 32: What is Scaffolding in MVC?

Answer: Scaffolding is a code generation framework for ASP.NET Web applications. Visual Studio 2013 includes pre-installed code generators for MVC and Web API projects. You add scaffolding to your project when you want to quickly add code that interacts with data models. Using scaffolding can reduce the amount of time to develop standard data operations in your project.

Prerequisites: To use ASP.NET Scaffolding, you must have:

- Microsoft Visual Studio 2013
- Web Developer Tools (part of default Visual Studio 2013 installation)
- ASP.NET Web Frameworks and Tools 2013 (part of default Visual Studio 2013 installation)

Advantages of using Scaffolding:

- Minimal or no code to create a data-driven Web applications.
- Quick development time.
- Pages that are fully functional and include display, insert, edit, delete, sorting, and paging functionalities.
- Built-in data validation that is based on the database schema.
- Filters that are created for each foreign key or Boolean fields.

Question 33: What is ViewStart?

Answer: Razor View Engine introduced a new layout named _ViewStart which is applied on all view automatically. Razor View Engine firstly executes the _ViewStart and then start rendering the other view and merges them.

Example of Viewstart:

```
1.  @ {
2.    Layout = "~/Views/Shared/_v1.cshtml";
3.  } < !DOCTYPE html >
4.    < html >
5.    < head >
6.    < meta name = "viewport"
7.  content = "width=device-width" / >
8.    < title > ViewStart < /title> < /head> < body >
9.    < /body> < /html>
```

Question 34: What is JsonResultType in MVC?

Answer: Action methods on controllers return JsonResult (JavaScript Object Notation result) that can be used in an AJAX application. This class is inherited from the "ActionResult" abstract class. Here Json is provided one argument which must be serializable. The JSON result object that serializes the specified object to JSON format.

Example:

```
1.  public JsonResult JsonResultTest()
2.  {
3.    Return Json("Hello My Friend!");
4.  }
```

Question 35: What is TempData?

Answer: Tempdata-

- TempData is a dictionary object derived from the TempDataDictionary class.

- TempData is used to pass data from the current request to a subsequent request, in other words in the case of redirection.

- The life of a TempData is very short and it retains its value for a short period of time.

- It requires typecasting for complex data type as I've used in my example:

- @foreach (var item in (List<MVCSample.Models.EmpRegistration>)TempData["EmployeeRegistration"])

- You can retain its value using the Keep method for subsequent requests.
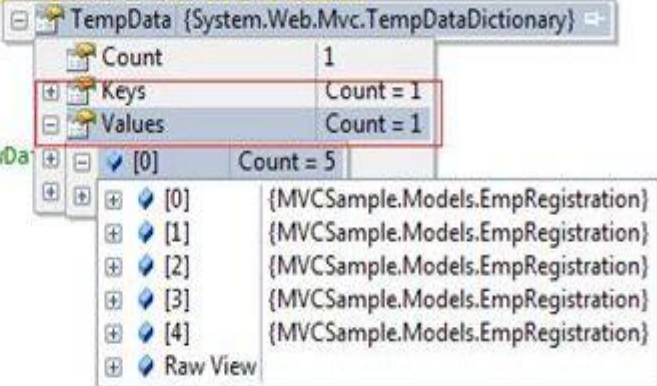
```
public ActionResult Verify()
{
    TempData["EmployeeRegistration"] = ObjEmp.GetEmpRegistrationsDetails();
    TempData.Keep("EmployeeRegistration");
    return View("Verify", TempData["EmployeeRegistration"]);



    #region Commented Code

    //return View("Verify", ViewDa
    #endregion Commented Code

}

//
// GET: /Register/Details/5
```

TempData {System.Web.Mvc.TempDataDictionary}	
Count	1
Keys	Count = 1
Values	Count = 1
[0]	Count = 5
[0]	{MVCSample.Models.EmpRegistration}
[1]	{MVCSample.Models.EmpRegistration}
[2]	{MVCSample.Models.EmpRegistration}
[3]	{MVCSample.Models.EmpRegistration}
[4]	{MVCSample.Models.EmpRegistration}
Raw View	

Question 36: How to use ViewBag?

Answer: ViewBag is dynamic property that takes advantage of new dynamic features in C# 4.0. It's also used to pass data from a controller to a view. In short, The ViewBag property is simply a wrapper around the ViewData that exposes the ViewData dictionary as a dynamic object. Now create an action method "StudentSummary" in the "DisplayDataController" controller that stores a Student class object in ViewBag.

```
1.   public ActionResult StudentSummary()
2.   {
3.      var student = new Student()
4.      {
5.         Name = "Sandeep Singh Shekhawat",
6.            Age = 24,
7.            City = "Jaipur"
8.      };
9.      ViewBag.Student = student;
10.     return View();
11. }
```

Thereafter create a view StudentSummary ("StudentSummary.cshtml") that shows student object data. ViewBag does not require typecasting for complex data type so you can directly access the data from ViewBag.

```
1.   @ {
2.      ViewBag.Title = "Student Summary";
3.      var student = ViewBag.Student;
4.   }
5.   < table >
6.      < tr >
7.      < th > Name < /th> < th > Age < /th> < th > City < /th> < /tr> < tr >
8.      < td > @student.Name < /td> < td > @student.Age < /td> < td > @student.City < /td> <
     /tr>
9.   < /table>
```

Here we used one more thing, "ViewBag.Title", that shows the title of the page.

Question 37: What is the Difference between ViewBag&ViewData?

Answer: Difference between ViewBag&ViewData-

- ViewData is a dictionary of objects that is derived from ViewDataDictionary class and accessible using strings as keys.
- ViewBag is a dynamic property that takes advantage of the new dynamic features in C# 4.0.
- ViewData requires typecasting for complex data type and check for null values to avoid error.
- ViewBag doesn't require typecasting for complex data type.
- Calling of ViewBag is:

  ViewBag.Name = "Yogesh";
  Calling of ViewDatais :
  ViewData["Name"] = "yogesh";

Question 38: What is Data Annotation Validator Attributes in MVC?

Answer: DataAnnotation plays a vital role in added validation to properties while designing the model itself. This validation can be added for both the client side and the server side.

You understand that decorating the properties in a model with an Attribute can make that property eligible for Validation.

Some of the DataAnnotation used for validation are given below:

Error List:

- Minimum char is 5 and maximum char is 10
- CustomerName contains invalid character.
- Customer Code is too small
- Invalid character
- Range should be between 1k & 10k

Customer Code | hkj | Customer Code is too small

Customer Name | hj | Minimum char is 5 and maximum char is 10

Amount | 788 | Range should be between 1k & 10k

Click

1. **Required**

   Specify a property as required.
   1. [Required(ErrorMessage="CustomerName is mandatory")]

2. **RegularExpression**

   Specifies the regular expression to validate the value of the property.
   1. [RegularExpression("[a-z]", ErrorMessage = "Invalid character")]

3. **Range**

   Specifies the Range of values between which the property values are checked.
   1. [Range(1000,10000,ErrorMessage="Range should be between 1k & 10k")]

4. **StringLength**

   Specifies the Min & Max length for a string property.
   1. [StringLength(50, MinimumLength = 5, ErrorMessage = "Minimum char is 5 and maximum char is 10")]

5. **MaxLength**

Specifies the Max length for the property value.
1. [MaxLength(10,ErrorMessage="Customer Code is exceeding")]

6. **MinLength**

It is used to check for minimum length.
1. [MinLength(5, ErrorMessage = "Customer Code is too small")]

Question 39: How can we done Custom Error Page in MVC?

Answer: The HandleErrorAttribute allows you to use a custom page for this error. First you need to update your web.config file to allow your application to handle custom errors.

1. <system.web>
2.   <customErrors mode="On">
3. </system.web>

Then, your action method needs to be marked with the atttribute.

1. [HandleError]
2. public class HomeController: Controller
3. {
4.   [HandleError]
5.   publicActionResultThrowException()
6.   {
7.   throw new ApplicationException();
8.   }
9. }

By calling the ThrowException action, this would then redirect the user to the default error page. In our case though, we want to use a custom error page and redirect the user there instead.So, let's create our new custom view page.

```
<%@ Page Language="C#" Inherits="System.Web.Mvc.ViewPage" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://r
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>CustomErrorView</title>
</head>
<body>
    <h2>
        Error</h2>
    <p>
        Controller:
        <%=((HandleErrorInfo)ViewData.Model).ControllerName %>
    </p>
    <p>
        Action:
        <%=((HandleErrorInfo)ViewData.Model).ActionName %>
    </p>
    <p>
        Message:
        <%=((HandleErrorInfo)ViewData.Model).Exception.Message %>
    </p>
    <p>
        Stack Trace:
        <%=((HandleErrorInfo)ViewData.Model).Exception.StackTrace %>
    </p>
</body>
</html>
```

Next, we simply need to update the HandleErrorAttribute on the action method.

```
1.  [HandleError]
2.  public class HomeController: Controller
3.  {
4.     [HandleError(View = "CustomErrorView")]
5.     publicActionResultThrowException()
6.     {
7.        throw new ApplicationException();
8.     }
9.  }
```

Question 40: Server Side Validation in MVC?

Answer: The ASP.NET MVC Framework validates any data passed to the controller action that is executing, it populates a ModelState object with any validation failures that it finds and passes that object to the controller. Then the controller actions can query the ModelState to discover whether the request is valid and react accordingly.

I will use two approaches in this article to validate a model data. One is to manually add an error to the ModelState object and another uses the Data Annotation API to validate the model data.

Approach 1: Manually Add Error to ModelState object-

I create a User class under the Models folder. The User class has two properties "Name" and "Email". The "Name" field has required field validations while the "Email" field has Email validation. So let's see the procedure to implement the validation. Create the User Model as in the following:

```
1.  namespace ServerValidation.Models
2.  {
3.     public class User
4.     {
5.        public string Name
6.        {
7.           get;
8.           set;
9.        }
10.       public string Email
11.       {
12.          get;
13.          set;
14.       }
15.    }
16. }
```

After that I create a controller action in User Controller (UserController.cs under Controllers folder). That action method has logic for the required validation for Name and Email validation on the Email field. I add an error message on ModelState with a key and that message will be shown on the view whenever the data is not to be validated in the model.

```
1.  using System.Text.RegularExpressions;
2.  using System.Web.Mvc;
3.  namespace ServerValidation.Controllers
4.  {
5.     public class UserController: Controller
6.     {
7.        public ActionResult Index()
8.        {
9.           return View();
10.       }
11.       [HttpPost]
12.       public ActionResult Index(ServerValidation.Models.User model)
13.       {
```

```
14.        if (string.IsNullOrEmpty(model.Name))
15.        {
16.            ModelState.AddModelError("Name", "Name is required");
17.        }
18.        if (!string.IsNullOrEmpty(model.Email))
19.        {
20.            string emailRegex = @ "^([a-zA-Z0-9_\-\.]+)@((\[[0-9]{1,3}" + @ "\.[0-
    9]{1,3}\.[0-9]{1,3}\.)|(([a-zA-Z0-9\-]+\" +  @ ".)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$";
21.            Regex re = new Regex(emailRegex);
22.            if (!re.IsMatch(model.Email))
23.            {
24.                ModelState.AddModelError("Email", "Email is not valid");
25.            }
26.        } else {
27.            ModelState.AddModelError("Email", "Email is required");
28.        }
29.        if (ModelState.IsValid)
30.        {
31.            ViewBag.Name = model.Name;
32.            ViewBag.Email = model.Email;
33.        }
34.        return View(model);
35.    }
36. }
37. }
```

Thereafter I create a view (Index.cshtml) for the user input under the User folder.

```
1.  @model ServerValidation.Models.User
2.  @ {
3.      ViewBag.Title = "Index";
4.  }
5.  @using(Html.BeginForm())
6.  {
7.    if (@ViewData.ModelState.IsValid)
8.        {
9.      if (@ViewBag.Name != null)
10.   { < b >
11.          Name: @ViewBag.Name < br / >
12.          Email: @ViewBag.Email < /b>       }
13.   } < fieldset >
14.      < legend > User < /legend>  < div class = "editor-label" >
15.      @Html.LabelFor(model => model.Name) < /div> < div class = "editor-field" >
```

16.   @Html.EditorFor(model => model.Name)
17.   @if(!ViewData.ModelState.IsValid)
18. {
19. < span class = "field-validation-
    error" > @ViewData.ModelState["Name"].Errors[0].ErrorMessage < /span>
20. }
21. < /div> < div class = "editor-label" >
22.   @Html.LabelFor(model => model.Email) < /div> < div class = "editor-field" >
23.   @Html.EditorFor(model => model.Email)
24.   @if(!ViewData.ModelState.IsValid)
25. {
26. < span class = "field-validation-
    error" > @ViewData.ModelState["Email"].Errors[0].ErrorMessage < /span>   }
27. < /div> < p >
28.   < input type = "submit"
29.   value = "Create" / >
30.   < /p> < /fieldset>
31. }

Question 41: What is the use of remote validation in MVC?

Answer: Remote validation is the process where we validate specific data posting data to a server without posting the entire form data to the server. Let's see an actual scenario, in one of my projects I had a requirement to validate an email address, whetehr it already exists in the database. Remote validation was useful for that; without posting all the data we can validate only the email address supplied by the user.

Practical Explanation

Let's create a MVC project and name it accordingly, for me its "TestingRemoteValidation". Once the project is created let's create a model named UserModel that will look like:

1. public class UserModel
2. {
3.   [Required]
4.   public string UserName
5.   {
6.   get;
7.   set;
8.   }

```
9.      [Remote("CheckExistingEmail", "Home", ErrorMessage = "Email already exists!")]
10.     public string UserEmailAddress
11.     {
12.        get;
13.        set;
14.     }
15. }
```

Let's get some understanding of the remote attribute used, so the very first parameter "CheckExistingEmail" is the the name of the action. The second parameter "Home" is referred to as controller so to validate the input for the UserEmailAddress the "CheckExistingEmail" action of the "Home" controller is called and the third parameter is the error message. Let's implement the "CheckExistingEmail" action result in our home controller.

```
1.   public ActionResult CheckExistingEmail(string UserEmailAddress)
2.   {
3.      bool ifEmailExist = false;
4.      try
5.      {
6.         ifEmailExist = UserEmailAddress.Equals("mukeshknayak@gmail.com") ? true : fals
     e;
7.         return Json(!ifEmailExist, JsonRequestBehavior.AllowGet);
8.      } catch (Exception ex)
9.      {
10.        return Json(false, JsonRequestBehavior.AllowGet);    }  }
```

Question 42: What are the Exception filters in MVC?

Answer: Exception is part and parcel of an application. They are a boon and a ban for an application too. Isn't it? This would be controversial, for developers it helps them track minor and major defects in an application and sometimes they are frustrating when it lets users land on the Yellow screen of death each time. This would make the users mundane to the application. Thus to avoid this, developers handle the exceptions. But still sometimes there are a few unhandled exceptions.

Now what is to be done for them? MVC provides us with built-in "Exception Filters" about which we will explain here.

Get Started:

Exception filters run when some of the exceptions are unhandled and thrown from an invoked action. The reason for the exception can be anything and so is the source of the exception.

Creating an Exception Filter:

Custom Exception Filters must implement the builtinIExceptionFilter interface. The interface looks as in the following:

1. public interface IExceptionFilter
2. {
3.   void OnException(ExceptionContext filterContext)
4. }

Whenever an unhandled exception is encountered, the OnException method gets invoked. The parameter as we can see, ExceptionContext is derived from the ControllerContext and has a number of built-in properties that can be used to get the information about the request causing the exception. Their property's ExceptionContextpassess are shown in the following table:

Name	Type	Detail
Result	ActionResult	The result returned by the action being invoked.
Exception	Exception	The unhandled exceptions caused from the actions in the applications.
ExceptionHandled	BOOL	This is a very handy property that returns a bool value (true/false) based on if the exception is handled by any of the filters in the applicaiton or not.

The exception being thrown from the action is detailed by the Exception property and once handled (if), and then the property ExceptionHandled can be toggled, so that the other filters would know if the exception has been already handled and cancel the other filter requests to handle. The problem is that if the exceptions are not handled, then the default MVC behavior shows the dreaded yellow screen of death. To the users, that makes a very impression on the users and more importantly, it exposes the application's handy and secure information to the outside world that may have hackers and then the application gets into the road to hell. Thus, the exceptions need to be dealt with very carefully. Let's show one small custom exception filter. This filter can be stored inside the Filters folder in the web project of the solution. Let's add a file/class called CustomExceptionFilter.cs.

```
1.  public class CustomExceptionFilter: FilterAttribute,
2.      IExceptionFilter
3.  {
4.      public void OnException(ExceptionContext filterContext)
5.      {
6.          if (!filterContext.ExceptionHandled && filterContext.Exception is NullReference
    Exception)
7.          {
8.              filterContext.Result = new RedirectResult("customErrorPage.html");
9.              filterContext.ExceptionHandled = true;
10.         }
11.     }
12. }
13.
```

Question 43: What is MVC HTML- Helpers and it's Methods?

Answer: Helper methods are used to render HTML in the view. Helper methods generate HTML output that is part of the view. They provide an advantage over using the HTML elements since they can be reused across the views and also requires less coding. There are several builtin helper methods that are used to generate the HTML for some commonly used HTML elements, like form, checkbox, dropdownlist etc. Also we can create our own helper methods to generate custom HTML. First we will see how to use the builtin helper methods and then we will see how to create custom helper methods.

Standard HtmlHelper methods: Some of the standard helper methods are:

- **ActionLink**: Renders an anchor.
- **BeginForm**: Renders HTML form tag
- **CheckBox**: Renders check box.
- **DropDownList**: Renders drop-down list.
- **Hidden**: Renders hidden field
- **ListBox**: Renders list box.
- **Password**: Renders TextBox for password input
- **RadioButton**: Renders radio button.
- **TextArea**: Renders text area.
- **TextBox**: Renders text box.

Question 44: Define Controller in MVC?

Answer: The controller provides model data to the view, and interprets user actions such as button clicks. The controller depends on the view and the model. In some cases, the controller and the view are the same object.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace DemoMVC.Controllers
{
    0 references
    public class EmployeeController : Controller
    {
        0 references
        public string EmployeeNames()
        {
            return @"<ul>
                        <li>Sourabh Somani</li>
                        <li>Shaili Dashora</li>
                        <li>Saloni Choudhry</li>
                        <li>Mahesh Chand</li>
                        <li>DJ</li>
                        <li>Dinesh Beniwal</li>
                    </ul>";
        }
    }
}
```

Controller

Action

The Controllers Folder: The Controllers Folder contains the controller classes responsible for handling user input and responses. MVC requires the name of all controllers to end with "Controller".

In our example, Visual Web Developer has created the following files: HomeController.cs (for the Home and about pages) and AccountController.cs (For the Log On pages):
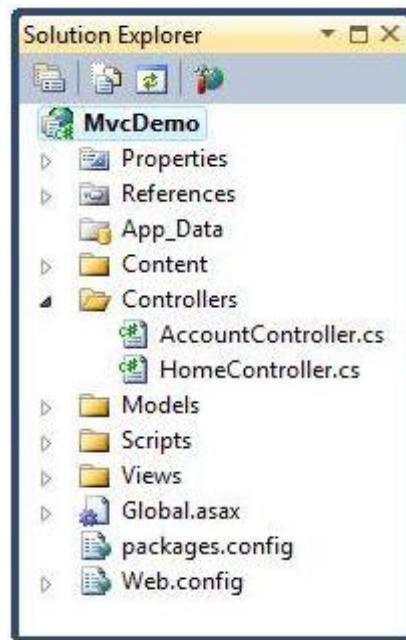
Question 45: Explain Model in MVC?

Answer: The model represents the data, and does nothing else. The model does NOT depend on the controller or the view. The MVC Model contains all application logic (business logic, validation logic, and data access logic), except pure view and controller logic. With MVC, models both hold and manipulate application data.

The Models Folder: The Models Folder contains the classes that represent the application model.

Visual Web Developer automatically creates an AccountModels.cs file that contains the models for application security.

Question 46: Explain View in MVC?

Answer: A view is responsible for displaying all of, or a portion of, data for users. In simple terms, whatever we see on the output screen is a view.

The Views Folder: The Views folder stores the files (HTML files) related to the display of the application (the user interfaces). These files may have the extensions html, asp, aspx, cshtml, and vbhtml, depending on the language content.

The Views folder contains one folder for each controller. Visual Web Developer has created an Account folder, a Home folder, and a Shared folder (inside the Views folder). The Account folder contains pages for registering and logging in to user accounts. The Home folder is used for storing application pages like the home page and the about page. The Shared folder is used to store views shared between controllers (master pages and layout pages).

Question 47: What is Attribute Routing in MVC?

Answer: A route attribute is defined on top of an action method. The following is the example of a Route Attribute in which routing is defined where the action method is defined.

In the following example, I am defining the route attribute on top of the action method:

```
1.  public class HomeController: Controller
2.  {
3.      //URL: /Mvctest
4.      [Route("Mvctest")]
5.      public ActionResult Index()
6.      ViewBag.Message = "Welcome to ASP.NET MVC!";
7.      return View();
8.  }
9.  }
```

Attribute Routing with Optional Parameter-

We can also define an optional parameter in the URL pattern by defining a question mark ("?") to the route parameter. We can also define the default value by using parameter=value.

```
1.  public class HomeController: Controller
2.  {
3.      // Optional URI Parameter
4.      // URL: /Mvctest/
5.      // URL: /Mvctest/0023654
6.      [Route("Mvctest /
7.      {
8.          customerName ?
9.      }")]
10.     public ActionResult OtherTest(string customerName)
11.     ViewBag.Message = "Welcome to ASP.NET MVC!";
12.     return View();
13. }
14. // Optional URI Parameter with default value
15. // URL: /Mvctest/
16. // URL: /Mvctest/0023654
```

```
17.    [Route("Mvctest /
18.    {
19.        customerName = 0036952
20.    }")]
21. public ActionResult OtherTest(string customerName)
22. {
23.    ViewBag.Message = "Welcome to ASP.NET MVC!";
24.    return View();
25. }
26. }
```

Question 48: Explain RenderSection in MVC?

Answer: RenderSection() is a method of the WebPageBase class. Scott wrote at one point, The first parameter to the "RenderSection()" helper method specifies the name of the section we want to render at that location in the layout template. The second parameter is optional, and allows us to define whether the section we are rendering is required or not. If a section is "required", then Razor will throw an error at runtime if that section is not implemented within a view template that is based on the layout file (that can make it easier to track down content errors). It returns the HTML content to render.

```
1.  <div id="body">
2.      @RenderSection("featured", required: false)
3.      <section class="content-wrapper main-content clear-fix">
4.          @RenderBody()
5.      </section>
6.  </div>
```

Question 49: What is GET and POST Actions Types?

Answer:

GET - GET is used to request data from a specified resource. With all the GET request we pass the URL which is compulsory, however it can take the following overloads.

.get(url [, data] [, success(data, textStatus, jqXHR)] [, dataType]).done/.fail

POST - POST is used to submit data to be processed to a specified resource. With all the POST requests we pass the URL which is compulsory and the data, however it can take the following overloads.

.post(url [, data] [, success(data, textStatus, jqXHR)] [, dataType])

Question 50: What's new in MVC 6?

Answer: In MVC 6 Microsoft removed the dependency of System.Web.Dll from MVC6 because it's so expensive that typically it consumes 30k of memory per request and response, whereas now MVC 6 only requires 2k of memory per request and the response is a very small memory consumtion.

The advantage of using the cloud-optimized framework is that we can include a copy of the mono CLR with your website. For the sake of one website we do not need to upgrade the .NET version on the entire machine. A different version of the CLR for a different website running side by side.

MVC 6 is a part of ASP.NET 5 that has been designed for cloud-optimized applications. The runtime automatically picks the correct version of the library when our MVC application is deployed to the cloud.
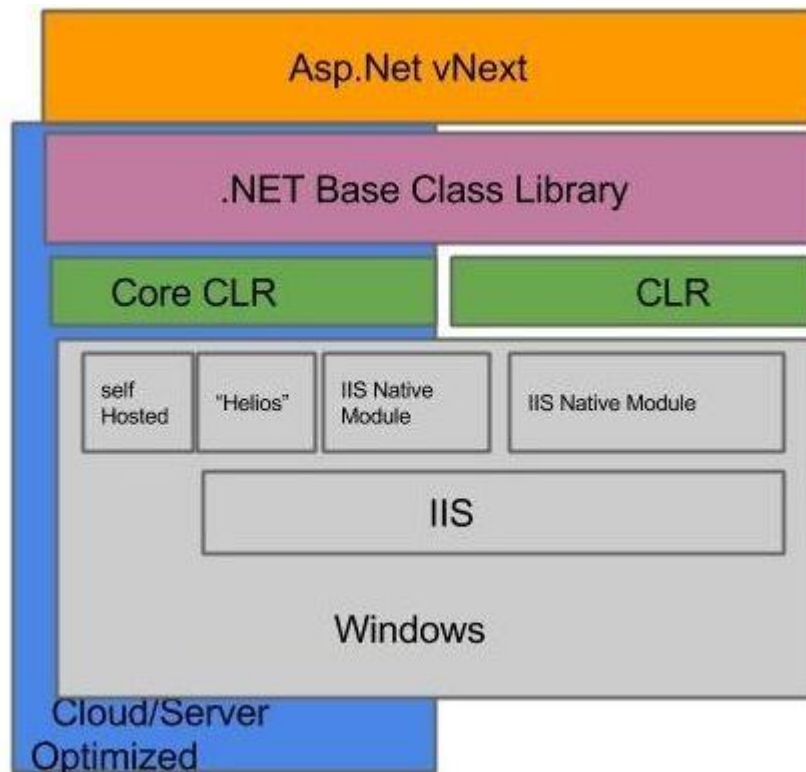
The Core CLR is also supposed to be tuned with a high resource-efficient optimization. Microsoft has made many MVC, Web API, WebPage and SignalLrpeices we call MVC 6.

Most of the problems are solved using the Roslyn Compiler. In ASP.NET vNext uses the Roslyn Compiler. Using the Roslyn Compiler we do not need to compile the application, it automatically compiles the application code. You will edit a code file and can then see the changes by refreshing the browser without stopping or rebuilding the project.

Run on hosts other than IIS:

Where we use MVC5 we can host it on an IIS server and we can also run it on top of an ASP. NET Pipeline, on the other hand MVC 6 has a feature that makes it better and that feature is itself hosted on an IIS server and a self-user pipeline.

Environment based configuration system:

The configuration system provides an environment to easily deploy the application on the cloud. Our application works just like a configuration provider. It helps to retrieve the value from the

various configuration sources like XML file.

MVC 6 includes a new environment-based configuration system. Unlike something else it depends on just the Web.Config file in the previous version.

Dependency injection:

Using the IServiceProvider interface we can easily add our own dependency injection container. We can replace the default implementation with our own container.

Supports OWIN:

We have complete control over the composable pipeline in MVC 6 applications. MVC 6 supports the OWIN abstraction.

Chapter 7

WPF Interview Questions and Answers

Question 1: What is WPF?

Answer: WPF stands for Windows Presentation Foundation. It's a re-invention of a UI for Desktop applications using WPF. Apart from dropping controls on "**Windows Forms**" just as developers have been doing for years, WPF provides an extra rapid boost to the application development including Rich User Interface, Animation and much more.

In a nutshell the following things can be done using WPF:

- Draw normal controls and graphics.
- Can easily load/play audio and video files.
- Can provide smooth graphical effects such as drop shadows and color gradients.
- Can use shared styles which can be used across the same controls to provide the same theme, skin and design.
- Transforming objects including shapes, controls and video.
- Can create and animate 3D graphics.
- Can easily draw vector graphics that scale without jagged aliasing.

Advantages:

- Tight multimedia integration
- Resolution independence
- Hardware acceleration

Question 2: What is Content Alignment in WPF?

Answer: The content of content controls in **WPF** is dealt using various properties. These two properties are **HorizontalContentAlignment** and **VerticalContentAlignment**. These properties are defined in the **System.Windows.Controls.Control** class that is the parent class of all controls in WPF.

If we create a UI with a Button and a TextBox control, the UI looks like the following figure where the default vertical and horizontal alignment of content of a Button is center. The default vertical and horizontal alignment of content of a TextBox is left and top.

You want to set the contents of the Button and TextBox to bottom and right.
The code sets **VerticalContentAlignment** and **HorizontalContentAlignmentproperties** to bottom and right.

1. <Grid Name="StackPanel1" Background="LightGray">
2. <Button Name="Button1" Background="LightBlue" Height="45" Content="Click Me!" Margin="23,12,159,0" VerticalAlignment="Top" FontSize="16" FontWeight="Bold" VerticalContentAlignment="Bottom" HorizontalContentAlignment="Right" />
3. <TextBox Height="50" Margin="26,72,74,0" Name="textBox1" VerticalAlignment="Top" Text="I am a TextBox" FontSize="16" VerticalContentAlignment="Bottom" HorizontalContentAlignment="Right" />
4. </Grid>

Output:

Question 3: What are Resources in WPF?

Answer: Windows Presentation Foundation (WPF) resources provide a simple way to reuse commonly defined objects and values. Resources in WPF allow you to set the properties of multiple controls at a time. For example, you can set the background property on several elements in a WPF application using a single resource.

The best way of defining the resources is on a Window or Page element level. Any resource that you define for an element also applies to their child elements of that element. For example, if you define a resource for a Window element that has a Grid as a child element, then the resources defined for the window elements can also be used by the grid element. However, if you define a resource for the grid element, then the resource applies only to the child elements of the grid element.

Syntax for resources in WPF,

<elementName propertyName="{markupExtension keyName}">
  <!-Content -->
</elementName>

Where,

- **elementName:** Name of the element that uses the resource.
- **propertyName:** Name of the property that takes its value from the resource.
- **markupExtension:** Define type of resource.
- **keyName:** key name of the resource, which is unique string to identify the resource.

There are two types of resource, namely,

- Static Resource
- Dynamic Resource

Question 4: What are static and dynamic resources?

Answer: There are two types of resource, namely,

- Static Resource
- Dynamic Resource

Let's see basics of both resources,

Static Resource - We should use the StaticResource markup extension to define the resource as a static resource. The value of StaticResource is determined at the time of loading.

Let's have a sample program, Add the below code snippet in Window1.xaml file inside the Grid.

```
1. <Grid.Resources>
2.    <SolidColorBrush x:Key="lblbgcolor" Color="Blue"/>
3. </Grid.Resources>
4. <Label Name="lbl" Margin="71,44,77,0" Background="{StaticResourcelblbgcolor}" Height="49" />
```

Above code, Grid control uses the Resources property (<Grid.Resources>) to define resource. SolidColorBrush resource named lblbgcolor defined. lblbgcolor resource is used to set the background property of lable.

Dynamic Resource - Dynamic Resource we use in a situation where we want to change the value of property at run time.

Let's have a sample program, Add the following code snippet in Window1.xaml file inside the Window element.

1. <Window.Resources>
2.   <SolidColorBrush x:Key="brush" Color="Red" />
3. </Window.Resources>
4. <Button x:Name="btn" Content="Click Me" Click="Button_Click" Background="{Dyna micResource brush}" Height="100" Width="100" />

Open code behind and add the following code snippet.

1. private void Button_Click(object sender, RoutedEventArgs e)
2. {
3.   this.btn.SetResourceReference(BackgroundProperty, "brush");
4. }

In the above code, Window control uses the Resources property (<Window.Resources>) to define resource. SolidColorBrush resource named brush defined. Brush resource is used to set the background property of button.

Question 5: What is value convertor in WPF?

Answer: A Value Converter functions as a bridge between a target and a source and it is necessary when a target is bound with one source, for instance you have a text box and a button control. You want to enable or disable the button control when the text of the text box is filled or null.

In this case you need to convert the string data to Boolean. This is possible using a Value Converter. To implement Value Converters, there is the requirement to inherit from **I Value Converter** in the **System.Windows.Data** namespace and implement the two methods **Convert** and **Convert Back.**

Note: In WPF, Binding helps to flow the data between the two WPF objects. The bound object that emits the data is called the Source and the other (that accepts the data) is called the Target.

Example:

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Linq;
4.  using System.Text;
5.  using System.Threading.Tasks;
6.  using System.Windows.Data;
7.  namespace ValueConverters
8.  {
9.      public class ValueConverter: IValueConverter
10.     {
11.         public object Convert(object value, Type targetType, object parameter, System.Globalization.CultureInfo culture)
12.         {
13.             bool isenable = true;
14.             if (string.IsNullOrEmpty(value.ToString()))
15.             {
16.                 isenable = false;
17.             }
18.             return isenable;
19.         }
20.         public object ConvertBack(object value, Type targetType, object parameter, System.Globalization.CultureInfo culture)
21.         {
22.             throw new NotImplementedException();
23.         }
24.     }
25. }
```

Question 6: What is MVVM?

Answer: MVVM (Model View ViewModel) is a framework for making applications in WPF. MVVM is the same as the MVC framework. It is 3-tier architecture plus one more layer. We can do loose coupling using MVVM.

MVVM was introduced by John Gossman in 2005 specifically for use with WPF as a concrete application of Martin Fowler's broader Presentation Model pattern. The implementation of an application, based on the MVVM patterns, uses various platform capabilities that are available in some form for WPF, Silverlight Desktop/web, and on Windows. Many commercial applications, including Microsoft Expression products, were built following MVVM.

Advantage of MVVM:

- Modularity

- Test driven approach.

- Separation UI and Business layer as view and view model.

- Code sharing between pages and forms.

- Easy to Maintain.

List of features of MVVM:

- It separates the business and presentation layers, like MVP and MVC.

- Improve Structure/separation of concerns (View, ViewModel and Model).

- Enable a better Design/Developer Workflow.

- Enhance simplicity and testability.

- Enabled by the robust data binding capability of XAML.

- No need to use a code behind file (minimalist code-behind file).

- Provides application development ability for multiple environments.

- Powerful Data Binding, command, validation and much more.

- The designer and developer can work together.

Question 7: How can you explain view and view model in MVVM?

Answer: The View is the client interface, input-output interface or the user interface. It collects all the user interface elements of the window, navigation page, user control, resource file, style and themes, custom tools and controls. The view is unaware of the ViewModel and the Model, and vice versa the ViewModel and Model is unaware of the View and control is tightly decoupled.

But the view model is aware of the needs of the view. They communicate by data binding and a dependency property or properties.

ViewModel in MVVM:

ViewModel is a non-visual class. The MVVM Design Pattern does not derive from any WPF or Silverlight based class. The ViewModel is unaware of the view directly. Communication between the View and ViewModel is through some property and binding. Models are connected directly to the ViewModel and invoke a method by the model class; it knows what the model has, like properties, methods etcetera and also is aware of what the view needs.

One View-Model can connect to multiple models, work like a one-to-many relationship and encapsulate business logic and data for the View. A ViewModel inherits some interface like **INotifyPropertyChanged**, **icommand INotifyCollectionChanged** etcetera.
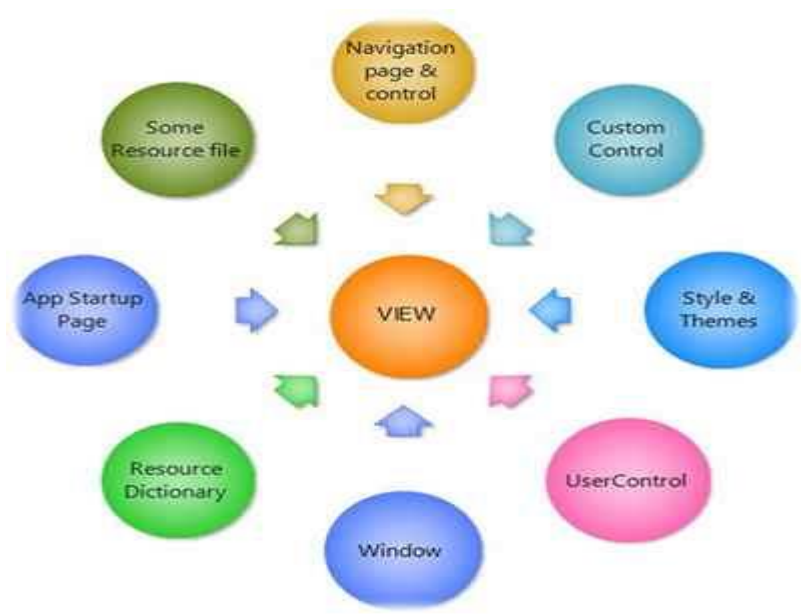
Question 8: What is Command Design Pattern and ICommand in WPF?

Answer: Command pattern is one of the most powerful design patterns in object oriented design patterns. This pattern allows you to decouple the request of an action from the object that actually performs an action, in other words a command pattern represents an action as an object. A Command object does not contain the functionality that is to be executed. This removes the direct link between the command definitions and the functionality and it's promoting a loose coupling. When you want to implement operations based on the user request then the command pattern is the best pattern to handle your object.

The following are the members of the Command Design Pattern:

- Client
- Invoker
- Command
- Concrete Command
- Receiver

Flow of Command Design Pattern:
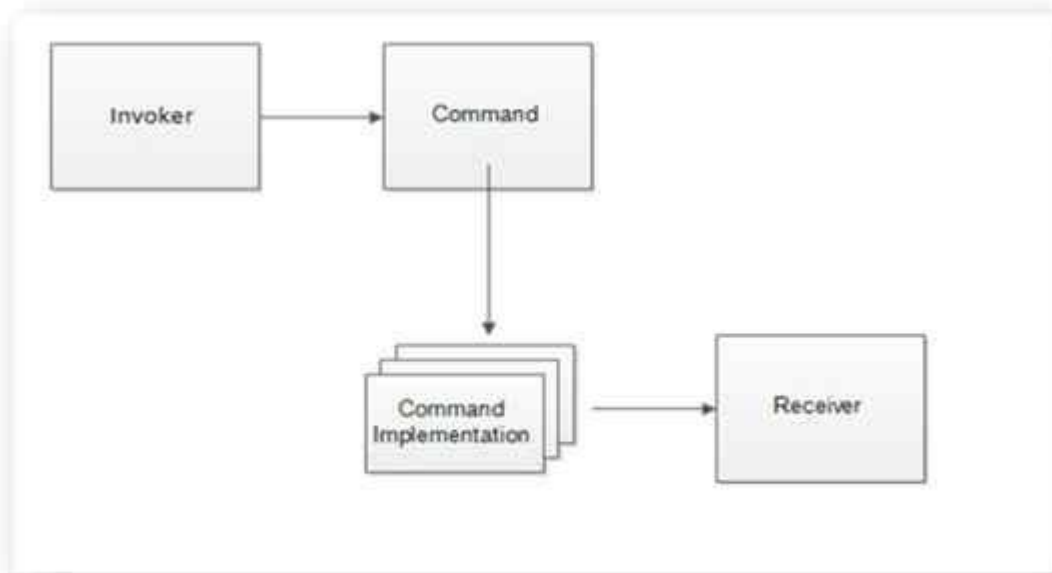
ICommand:

An ICommand is a core component of MVVM. ICommand is frequently used in MVVM, it provides separation logic between a View and View Model (User Interface and Business Logic). XAML offers a way to bind your GUI event better by ICommand. ICommand requires the user to define two methods, *bool CanExecute* and *void Execute*. The CanExecute method really just says to the user, can I execute this Action? This is useful for controlling the context in which you can perform GUI actions.

ICommand is very simple but it's more interesting and complicated when you are using it in an application. ICommand integrates your User Interface to business logic or it's making a direct communication between a View to a View Model. It also provides a mechanism for the view to update the model/view-model.

```
1   #Assembly System.dll, v2.0.5.0

5   using System;
6   using System.ComponentModel;

8   namespace System.Windows.Input
9   {
10          public interface ICommand
14          {
15              event EventHandler CanExecuteChanged;

19              bool CanExecute(object parameter);
31              void Execute(object parameter);
40          }
41  }
42
```

Question 9: What is the Data Binding concept and How Binding works in WPF?

Answer: Data Binding is one of the greatest and most powerful features of XAML (WPF, Silverlight, Windows Phone or Windows 8) compared to other traditional web and Windows app technology in .NET. There was simple data binding for displaying single values, and complex data binding for displaying and formatting a bunch of data. XAML (WPF, Silverlight, Windows Phone or Windows 8) however makes it easy to bind nearly any property to any element or object or data source. You take data from some property or object or dependency property and bind it to another dependency property or object or else directly to an element. In a single word you can say in XAML "*Data binding is the process of getting information from one object to another and displaying it in one or more elements in the user interface*".

How {Binding} works in WPF:

The Binding keyword looks as in the following image. Here is the binding of TextBox UI controls with some binding property source object.

```
<TextBox Grid.Column="1" Grid.Row="1" Text="{Binding UserName,Mode=TwoWay}" />
```

Binding to a WPF element from code behind:

```
private void DataBindingMode()
{
    Binding binding = new Binding();
    binding.ElementName = "Slider1";
    binding.Path = new PropertyPath("Value");
    binding.UpdateSourceTrigger = UpdateSourceTrigger.PropertyChanged;
    txtMyTextBox.SetBinding(TextBox.TextProperty, binding);
    Binding_string.Text = "Text ={Binding ElementName=" + binding.Elem
}
```

In the above sample a TextBox and slider from code behind using some binding property.

Some Very Useful Properties of the Binding Class:

Property Name	Description
Element Name	The name of the element that gets or sets the binding source object when binding to a XAML element.
FallbackValue	Set the value to use when the binding does not return values.
Converter	Set the converter for the UI element.
Mode	Set the Binding diection between the target and source objects.
Path	Get or Set the path to the source property of the Binding source.
RelativeSource	Gets or Sets the binding source by specifying it's location and relative to the postion of the binding target.
Source	Gets or Sets the binding source when not binding to a WPF element.
StringFormat	
UpdateSourceTrigger	
ValidationRules	
NotifyOnSourceUpdated	Gets or sets the value determining the direction of the dataflow in the binding.
NotifyOnTargetUpdated	Gets or sets a value that indicates whether to raise the source Update event when a value is transferred from the source to the target.

Question 10: What is Trigger and how many types of triggers in WPF?

Answer: A Trigger is typically used in a Style or Control Template. It triggers on properties of whatever is being templated, and sets other properties of the control (or of specific template elements). For example, you would use a Trigger on **IsMouseOver** to respond to the mouse being over the control, and the setters might update a brush to show a "**hot**" effect.

Why to use Trigger?

Triggers are used in styles to perform actions on a change of any property value or event fires. Triggers create visual effects on controls. By using Triggers we can change the appearance of Framework Elements.

There are **five** types of triggers supported by WPF; they are:

1. Property Trigger
2. Data Trigger
3. MultiTrigger
4. MultiDataTrigger
5. Event Trigger

Example: For example, let's say you have a rectangle control. You want to change the background color of that control when the mouse hovers over the rectangle control and revert back when mouse leaves. For this you need to code in the mouse hover event and mouse leave event of the rectangle control in the backend class for changing the color of rectangle as in the following code.

```
1. private void Rectangle_MouseMove_1(object sender, MouseEventArgs e)
2. {
3.     this.rctback.Fill = Brushes.Red;
4. }
5. private void Rectangle_MouseLeave(object sender, MouseEventArgs e)
6. {
7.     this.rctback.Fill = Brushes.Green;
8. }
```

In the preceding code you have filled the background color of the Rectangle control by writing code in two different events, but a trigger helps to overcome this problem by reducing the code.

Question 11: What is Prism in WPF?

Answer: Prism (Composite Application Guidance for WPF and Silverlight) is designed to build applications in WPF and Silverlight that have a single code base. It helps to develop the client application in a modular fashion so that complexity of a large application can be divided into simpler modules.

In other words "*Prism is developed by Microsoft Patterns and Practices and provides guidance designed to help you to more easily design and build rich, flexible and easy-to-maintain Windows Presentation Foundation (WPF) desktop applications.*"

Architecture: The following diagram shows basic architecture:
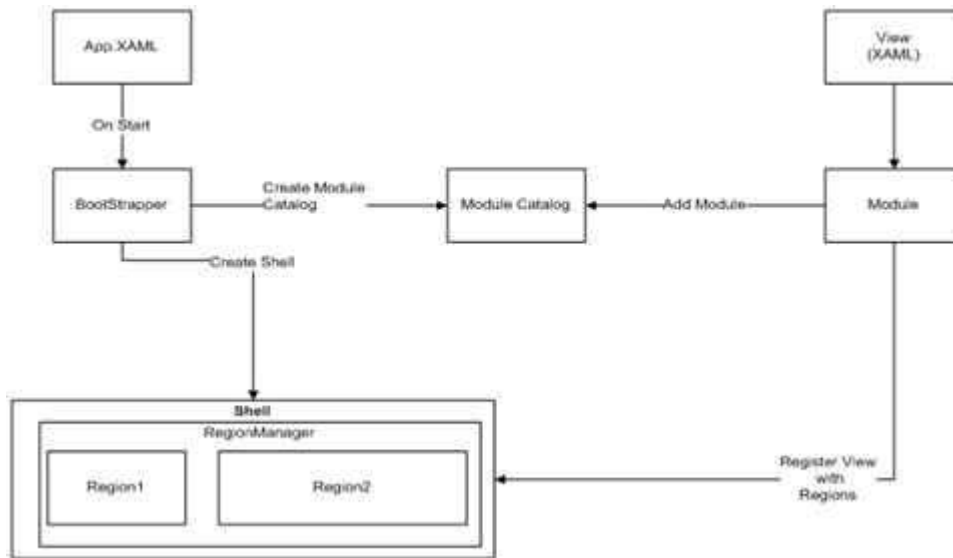
1. **App.XAML:** Call Boot Strapper on Application_Startup.
2. **BootStrapper:** This is a class file that calls Shell (Shell.XAML) and so creates catalogue of module.
3. **Shell:** This is like a Master Page having regions.
4. **Region:** It is like placeholders to register views.
5. **View:** This is XAML file having User Interface
6. **Module:** Each module can have one or more View(s) which are registered to Region (in the Shell) through Region Manager.

Question 12: What are the Binding Modes in XAML?

Answer: The DataBinding mode defines the communication direction to the source or the direction of data flow from the source. In XAML (WPF, Silverlight, WP or Win8 App) there are five ways you can bind a data target object to a source.
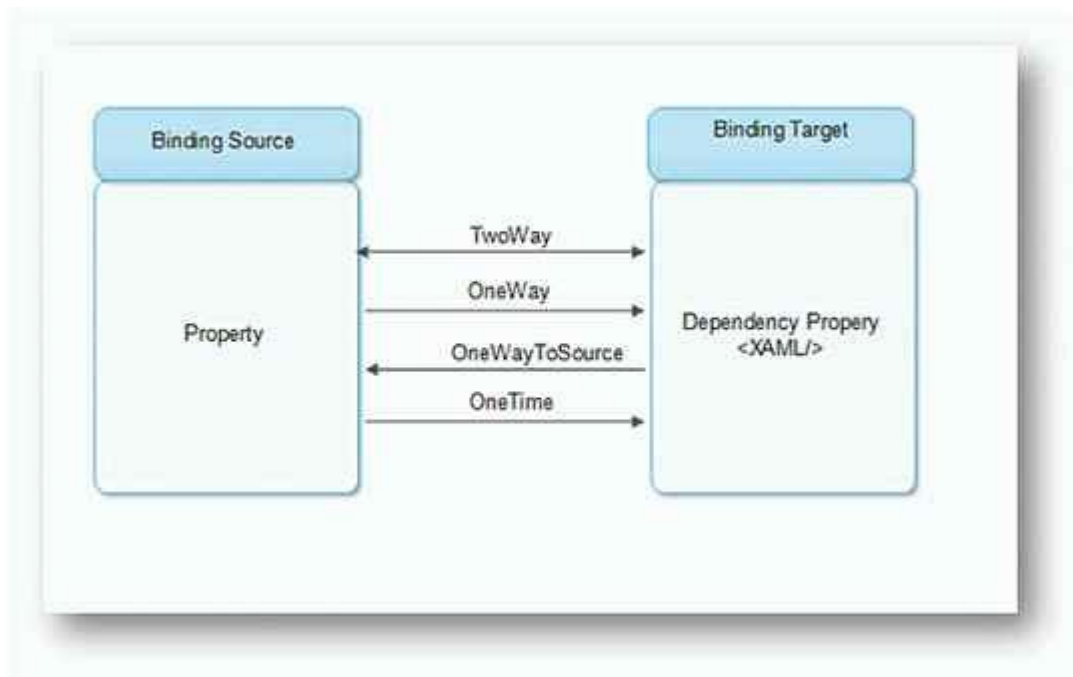
- **OneWay:** Data moves only one direction, the source property automatically updates the target property but the source is not changed.

- **TwoWay:** Data moves both directions, if you change it in the source or target it is automatically updated to the other.

- **OneWayToSource:** Data moves from the target to the source changes to the target property to automatically update the source property but the target is not changed.

- **OneTime:** Data is changed only one time and after that it is never set again, only the first time changes to the source property automatically update the target property but the source is not changed and subsequent changes do not affect the target property.

Question 13: What is the difference between MVP, MVC and MVVM?

Answer: MVP (Model-View-Presenter)-

In the MVP pattern the User sends the input to the view, the view forward it to presenter and presenter then modify the view or the model depending on the type of user action. The view and the presenter are tightly coupled through bi-directional relationship. The model does not know about the presenter. The view itself is passive, that's why it's called presenter pattern, since the presenter pushes the data into the view.

MVC (Model-View-Controller)-

In this pattern there is only one controller that gets all the inputs directly, it modifies the data in the model depending upon the type of the input. Both the model and the view are created by the controller. The view only knows about the model, but the model does not know about any other objects.
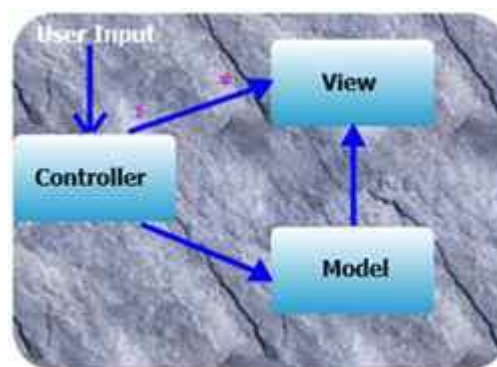
The Model View ViewModel (MVVM) is an architectural pattern used in software engineering that originated from Microsoft which is specialized in the Presentation Model design pattern. It is based on the Model-view-controller pattern (MVC), MVVM is a way of creating client applications that leverages core features of the WPF platform, allows for simple unit testing of application functionality, and helps developers and designers work together with less technical difficulties.
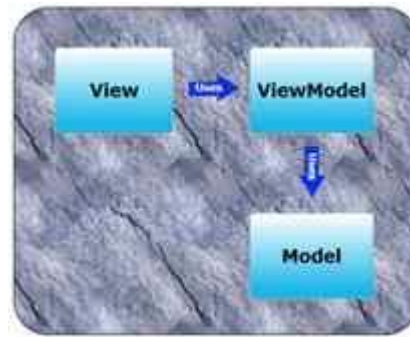
Question 14: What are the Templates in WPF?

Answer: Templates are an integral part of user interface design in WPF. WPF has the following three types of templates:

- Control Template
- Items Panel Template
- Data Template

Control Template - The ControlTemplate of a control defines the appearance of the control. We can change or define a new look and appearance of a control by simply changing the ControlTemplate of a control. ControlTemplates are even more useful when you write your own controls. Using ControlTemplates, you can build a custom button that has a circular layout and changes its color when you mouse over or press it.

The ControlTemplate element in XAML defines a ControlTemplate at design-time. Templates are usually defined as resources using a FrameworkElement's Resources property. The following code snippet is the syntax for defining a ControlTemplate for a Button element.

1. <Grid>
2.   <Grid.Resources>
3.   <ControlTemplate x:Key="RoundButtonTemplate" />
4.   </Grid.Resources>
5. </Grid>

We need to create a circular button where the outer circle of the button is of a different color than the inner circle and when you mouse over and press the button, it changes the background color.

Add a Grid as contents of the ControlTemplate. Add two Ellipse elements within a Grid with different radii and different color fills.

1. <Grid.Resources>
2.   <ControlTemplate x:Key="RoundButtonTemplate">
3.   <Grid>
4.   <Ellipse Width="100" Height="100" Name="ButtonBorder" Fill="OrangeRed" />

5.   <Ellipse Width="80" Height="80" Fill="Orange" /> </Grid>
6.   </ControlTemplate>
7. </Grid.Resources>

The following code snippet creates a Button element and sets its Template to the ControlTemplate that we created-

<Button Template="{StaticResource RoundButtonTemplate}" >OK</Button>

ItemsPanelTemplate - In the previous example, we saw how a Style element can be used within the resources to group multiple properties of elements and set them using the Style property of elements. However, Style functionality does not end here. Style can be used to group and share not only properties, but also resources and event handlers on any FrameworkElement or FrameworkContentElement.

Styles are resources and used as any other resource and can be applied to the current element, parent element, root element and even on the application level. The scope if styles are similar to any other resources. The resource lookup process first looks up for local styles and if not found, it traverses to the parent element in the logical tree and so on. In the end, the resource lookup process looks for styles in the application and themes.

The Style element in XAML represents a style. The typical definition of the Style element looks as in the following:

<Style>
  Setters
</Style>

As you can see from the definition of Style, a Style has one more Setter element. Each Setter consists of a property and a value. The property is the name of the property and the value is the actual value of that property of the element to that the style will be applied to.

Setters Property - The Setters property of Type represents a collection of Setter and EventSetter objects. Listing 4 uses the Setters property and adds a Setter and EventSetter object.

The code snippet in Listing 4 sets the Setters property of a Style by adding a few Setter elements and one EventSetter element using XAML at design-time.

```
1.  <Grid>
2.     <Grid.Resources>
3.        <Style TargetType="{x:Type Button}">
4.  <Setter Property="Width" Value="200"/> <Setter Property="Height" Value="30"/> <Setter Property="Foreground" Value="White"/> <Setter Property="Background" Value="DarkGreen"/> <Setter Property="BorderBrush" Value="Black"/> <EventSetter Event="Click" Handler="Button1_Click"/>
5.        </Style>
6.     </Grid.Resources>
7.     <Button>Click me</Button>
8.  </Grid>
```

Question 15: What are the various layout panels in WPF?

Answer: WPF comes with the following five built-in panels:

- Canvas
- DockPanel
- Grid
- StackPanel
- WrapPanel

The purpose and use of these panels is different. Each panel has a different way to position and reposition child controls placed within that panel. The following articles in this series will summarise these panels and their usages.

Similar to any other WPF control, a Panel control may be represented in two ways. First, at design-time using XAML elements and attributes, and second, at run-time, using a WPF class and its properties.

The code snippet creates a Grid panel at design-time using XAML.

```
1.   <Window x:Class="CanvasPanelSample.Window1"
2.     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3.     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4.     Title="Window1" Height="300" Width="300"
5.     Name="RootWindow">
6.     <Grid Name="GridPanel" Background="Blue"
7.       Width="250" Height="200"
8.       VerticalAlignment="Top"
9.       HorizontalAlignment="Left"
10.      FlowDirection="LeftToRight"
11. />
12. </Window>
```

Output:
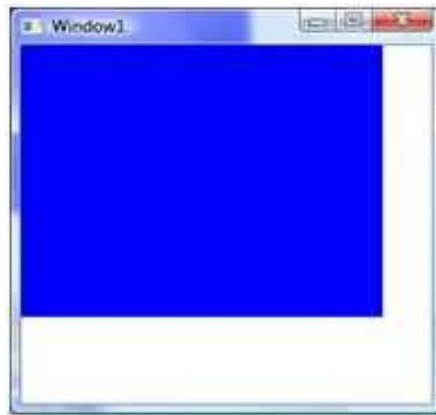
Question 16: What is Attached Properties in WPF?

Answer: Attached properties are basically Dependency Properties that allows the attachment of a value to any random object. Attached Properties (AP) are again a kind of Dependency Property (DP) in XAML. They can be used to receive a notification of a change of themself since they are a type of Dependency Property but one of the differences that these properties have is that they are not defined in the same class they used, unlike DPs.

- The type that defines the Attached Property is designed so that it can be the parent element of the elements that will set values for the Attached Property. The type then iterates its child objects using internal logic against some object's tree structure, obtains the values and acts on those values in some manner.

- The type that defines the Attached Property will be used as the child element for a variety of possible parent elements and content models.

- The type that defines the Attached Property represents a service. Other types set values for the Attached Property. Then, when the element that set the property is evaluated in the context of the service, the Attached Property values are obtained using internal logic of the service class.

APs are called "**attached**" properties because we can attach some behaviour to the control that is originally not expected from that control.

Example: Suppose I have a TextBox control and I want to extend its functionality to accept letters and special symbols but not numeric values.

The AP that I have defined in my class **TextBlockExtension.cs** is as in the following:

```
1.  public static bool GetAllowOnlyString(DependencyObject obj)
2.  {
3.      return (bool) obj.GetValue(AllowOnlyStringProperty);
4.  }
5.  public static void SetAllowOnlyString(DependencyObject obj, bool value)
6.  {
7.      obj.SetValue(AllowOnlyStringProperty, value);
8.  }
9.  // Using a DependencyProperty as the backing store for AllowOnlyString. This enables ani
    mation, styling, binding, etc...
10. public static readonly DependencyProperty AllowOnlyStringProperty = DependencyPrope
    rty.RegisterAttached("AllowOnlyString", typeof (bool), typeof (TextblockExtension), new
     PropertyMetadata(false, AllowOnlyString));
11. private static void AllowOnlyString(DependencyObject d, DependencyPropertyChangedE
    ventArgs e)
12. {
13.     if (d is TextBox)
14.     {
15.         TextBox txtObj = (TextBox) d;
16.         txtObj.TextChanged += (s, arg) =>
17.         {
18.             TextBox txt = s as TextBox;
19.             if (!Regex.IsMatch(txt.Text, "^[a-zA-Z]*$"))
20.             {
21.                 txtObj.BorderBrush = Brushes.Red;
22.                 MessageBox.Show("Only letter allowed!");
23.             }
24.         };
25.     }
26. }
```

In the code, as we can see, the AP has the default value of False, in other words we need to provide the APs the value true wherever I want this functionality to work for the TextBox.

In my **MainWindow.xaml.cs** I have defined my TextBox as in the following:

```
1.  <TextBox Width="200" Height="50" local:TextblockExtension.AllowOnlyString="True"
    ></TextBox>
```

Question 17: What is resource in WPF? How many types of resources in WPF?

Answer: Windows Presentation Foundation (WPF) resources provide a simple way to reuse commonly defined objects and values. Resources in WPF allow you to set the properties of multiple controls at a time. For example, you can set the background property on several elements in a WPF application using a single resource. The best way of defining the resources is on a Window or Page element level. Any resource that you define for an element also applies to their child elements of that element.

If you define a resource for the grid element, then the resource applies only to the child elements of the grid element.

Syntax for resources in WPF is as follows:

<elementName propertyName="{markupExtension keyName}">
<!-Content -->
</elementName>

There are two types of resource, namely,

- Static Resource
- Dynamic Resource

Static Resource: We should use the StaticResource markup extension to define the resource as a static resource. The value of StaticResource is determined at the time of loading.

1. <Grid.Resources>
2.   <SolidColorBrush x:Key="lblbgcolor" Color="Blue"/>
3. </Grid.Resources>
4. <Label Name="lbl" Margin="71,44,77,0" Background="{StaticResourcelblbgcolor}" Height="49" />

Dynamic Resource: Dynamic Resource we use in a situation where we want to change the value of property at run time.

1. <Window.Resources>
2.   <SolidColorBrush x:Key="brush" Color="Red" />
3. </Window.Resources>

4. `<Button x:Name="btn" Content="Click Me" Click="Button_Click" Background="{Dyna micResource brush}" Height="100" Width="100" />`

Open Code behind and add the following code snippet:

1. `private void Button_Click(object sender, RoutedEventArgs e)`
2. `{`
3. `    this.btn.SetResourceReference(BackgroundProperty, "brush");`
4. `}`

Question 18: What is the difference between Static and Dynamic resources?

Answer: The most basic difference is that StaticResource evaluates the resource one time only, but DynamicResource evaluates it every time the resource is required. And due to this reason, DyanamicResource is heavy on the system but it makes pages or windows load faster.

Static Resources - A Static Resource will be resolved and assigned to the property during the loading of the XAML that occurs before the application is actually run. It will only be assigned once and any changes to the resource dictionary are ignored.

Static resource references work best for the following circumstances:

- Your application design concentrates most of its resources into page or application level resource dictionaries. Static resource references are not re-evaluated based on runtime behaviours such as reloading a page, and therefore there can be some performance benefit to avoiding large numbers of dynamic resource references when they are not necessary per your resource and application design.

- You are setting the value of a property that is not on a Dependency Object or a freezable.

- You are creating a resource dictionary that will be compiled into a DLL, and packaged as part of the application or shared between applications.

Dynamic Resources - A Dynamic Resource assigns an Expression object to the property during loading but does not actually lookup the resource until runtime when the Expression object is asked for the value. This defers looking up the resource until it is needed at runtime.

Dynamic resources work best for the following circumstances:

- The value of the resource depends on conditions that are not known until runtime. This includes system resources, or resources that are otherwise user settable. For example, you can create setter values that refer to system properties, as exposed by System Colours, System Fonts, or System Parameters. These values are truly dynamic because they ultimately come from the runtime environment of the user and operating system. You might also have application-level themes that can change, where page-level resource access must also capture the change.

- You are creating or referencing theme styles for a custom control.

- You intend to adjust the contents of a Resource Dictionary during an application lifetime.

Example:

```
1.  <Window.Resources>
2.     <SolidColorBrush x:Key="brush" Color="Green" />
3.     <Style TargetType="Border" x:Key="PageBackground">
4.  <Setter Property="Background" Value="Gold"/>
5.     </Style>
6.  </Window.Resources>
7.  <Grid>
8.     <Border Style="{DynamicResource PageBackground}">
9.       <Button x:Name="btn" Content="Rajkumar Test" Click="Button_Click" Background="{DynamicResource brush}" Height="30" Margin="53,130,85,130" /> </Border>
10. </Grid>
```

You can apply on code behind like the following,

```
1.  private void Button_Click(object sender, RoutedEventArgs e)
2.  {
3.     this.btn.SetResourceReference(BackgroundProperty, "brush");
4.  }
```

Question 19: What is WPF Dependency Property and how can we use?

Answer: WPF has provided some extended services to the CLR property that we can collectively call Dependency Properties. A Dependency Property is a property whose value depends on the external sources, such as animation, data binding, styles, or visual tree inheritance. Not only this, but a Dependency Property also has the builtin feature of providing notification when the property has changed, data binding and styling.

Advantages of a Dependency Property:

- Less memory consumption
- Property value inheritance
- Change notification and Data Bindings
- Participation in animation, styles and templates
- CallBacks
- Resources
- Overriding Metadata

Code:

```
1. public class CarDependencyProperty: DependencyObject
2. {
3.     //Register Dependency Property
4.     public static readonly DependencyProperty CarDependency = DependencyProperty.Register("MyProperty", typeof (string), typeof (DependencyPropertySample));
5.     public string MyCar
6.     {
7.         get
```

```
8.      {
9.          return (string) GetValue(CarDependency);
10.     }
11.     set
12.     {
13.         SetValue(CarDependency, value);
14.     }
15.   }
16. }
17. public partial class CarDependencyPropertyDemo: Window
18. {
19.    public CarDependencyPropertyDemo()
20.    {
21.       InitializeComponent();
22.    }
23.    private void MyButton_Click(object sender, RoutedEventArgs e)
24.    {
25.       CarDependency dpSample = TryFindResource("CarDependency ") as CarDependency;
26.       MessageBox.Show(dpSample.MyCar);
27.    }
28. }
```

Question 20: What is Attached Properties and how to register it?

Answer: Attached Properties (AP) can be used to receive a notification of a change of themself since they are a type of Dependency Property but one of the differences that these properties have is that they are not defined in the same class they used, unlike DPs. One of the misconceptions that a WPF developer usually has is that these properties can only be defined in the parent control the control in which we want to use them.

AP can be defined in one of the following three contexts:

- The type that defines the Attached Property is designed so that it can be the parent element of the elements that will set values for the Attached Property. The type then iterates its child objects using internal logic against some object's tree structure, obtains the values and acts on those values in some manner.

- The type that defines the Attached Property will be used as the child element for a variety of possible parent elements and content models.

- The type that defines the Attached Property represents a service. Other types set values for the Attached Property. Then, when the element that set the property is evaluated in the context of the service, the Attached Property values are obtained using internal logic of the service class.

Example:

```
1.  public static bool GetAllowOnlyString(DependencyObject obj)
2.  {
3.      return (bool) obj.GetValue(AllowOnlyStringProperty);
4.  }
5.  public static void SetAllowOnlyString(DependencyObject obj, bool value)
6.  {
7.      obj.SetValue(AllowOnlyStringProperty, value);
8.  }
9.  // Using a DependencyProperty as the backing store for AllowOnlyString. This enables ani
    mation, styling, binding, etc...
10. public static readonly DependencyProperty AllowOnlyStringProperty = DependencyPrope
    rty.RegisterAttached("AllowOnlyString", typeof (bool), typeof (TextblockExtension), new
     PropertyMetadata(false, AllowOnlyString));
11. private static void AllowOnlyString(DependencyObject d, DependencyPropertyChangedE
    ventArgs e)
12. {
13.     if (d is TextBox)
14.     {
15.         TextBox txtObj = (TextBox) d;
16.         txtObj.TextChanged += (s, arg) =>
17.         {
18.             TextBox txt = s as TextBox;
19.             if (!Regex.IsMatch(txt.Text, "^[a-zA-Z]*$"))
20.             {
21.                 txtObj.BorderBrush = Brushes.Red;
22.                 MessageBox.Show("Only letter allowed!");
23.             }
24.         };
```

25. }
26. }

In the code, as we can see, the AP has the default value of False, in other words we need to provide the APs the value true wherever I want this functionality to work for the TextBox.

In my MainWindow.xaml.cs I have defined my TextBox as in the following:

> <TextBox Width="200" Height="50" local:TextblockExtension.AllowOnlyString="True" ></TextBox>

Question 21: What is a Routed event?

Answer: Routed Events is about the Hierarchy of the controls you are using in the Events. Routed Events are a new Infrastructure given by WPF that permit events to tunnel down the tree to the target elements or Bubble up to the Root element. Routed Events are just like normal events.

Types of Routed Events: Routed Events are of three types, which are as follows:

- Bubbling Events
- Tunneling Events
- Direct Events

Tunneling Events: Tunneling events are the reverse of the Bubbling events. Tunneling Events raised first in the controls hierarchy. These events are raised by the Root elements. This allows events to tunnel down the tree.
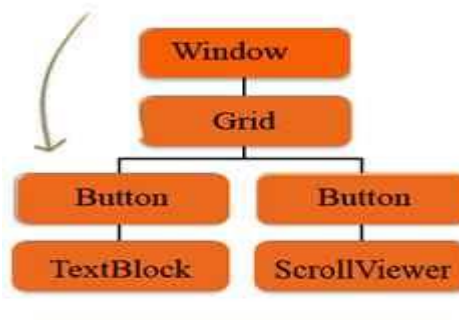
Bubbling Events: Bubbling Events are those Events which are first raised by the control than raised by the other controls in the control hierarchy. It allows Bubble up to the tree till the Root Element. First Tunneling events are raised then bubbling events raised.
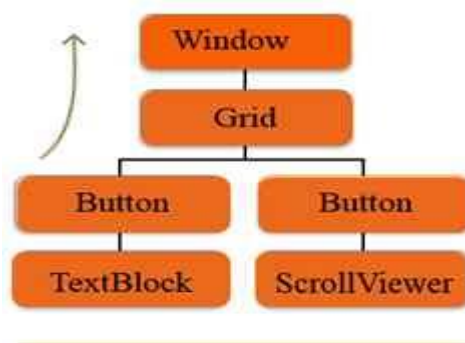
Direct Event: Direct Event is generally raised by the control itself. The behavior of this event is same as the .NET general event.

Question 22: What is Rotate transform in WPF?

Answer: RotateTransform rotates an element clockwise by a specified angle about the point. The RotateTransform object in WPF represents RotateTransform. The Angle property represents the angle in degrees to rotate clockwise. The CenterX and CenterY properties represent the X and Y coordinates of the center point. By default, a ScaleTransform is centered at the point (0,0), which corresponds to the upper-left corner of the rectangle.

Creates two rectangles with same position and sizes accept the second rectangle is rotated at 45 degrees.

```
1.  <Grid>
2.    <!-- Original Rectangle -->
3.    <Rectangle Width="200" Height="50" Fill="Yellow" />
4.    <!-- Rectangle with 45 degrees rotation -->
5.    <Rectangle Width="200" Height="50" Fill="Blue" Opacity="0.5">
```

6.   <Rectangle.RenderTransform>
7.   <RotateTransform CenterX="0" CenterY="0" Angle="45" /> </Rectangle.Render Transform>
8.   </Rectangle>
9.   </Grid>
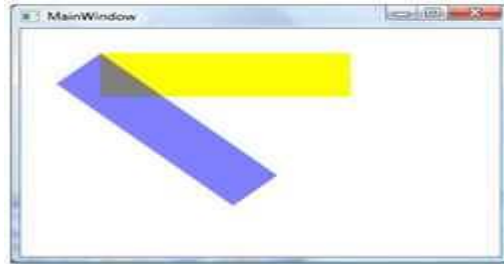
The following code snippet changes the values of CenterX and CenterY.

1.   <Rectangle Width="200" Height="50" Fill="Blue" Opacity="0.5" Margin="61,27,117,184">
2.   <Rectangle.RenderTransform>
3.   <RotateTransform CenterX="-50" CenterY="50" Angle="45" />
4.   </Rectangle.RenderTransform>
5.   </Rectangle>

Question 23: What is the Control Template in WPF?

Answer: The ControlTemplate contains the tree of elements that define the desired look. After you define a ControlTemplate you can attach it to any Control or Page by setting it's TemplateProperty.

1.   <Grid>
2.   <Grid.Resources>
3.   <ControlTemplate x:Key="buttonTemplate">
4.   <Grid>
5.   <Ellipse Width="160" Height="160" x:Name="outerCircle">
6.   <Ellipse.Fill>
7.   <LinearGradientBrush StartPoint="0,0" EndPoint="0,1">
8.   <GradientStop Offset="0" Color="Green"></GradientStop>

```
9.              <GradientStop Offset="1" Color="Purple"></GradientStop>
10.           </LinearGradientBrush>
11.         </Ellipse.Fill>
12.      </Ellipse>
13.      <Ellipse Width="120" Height="120">
14.         <Ellipse.Fill>
15.           <LinearGradientBrush StartPoint="0,0" EndPoint="0,1">
16.             <GradientStop Offset="0" Color="Gray"> </GradientStop>
17.             <GradientStop Offset="1" Color="Blue"> </GradientStop>
18.           </LinearGradientBrush>
19.         </Ellipse.Fill>
20.      </Ellipse>
21.   </Grid>
22.   <ControlTemplate.Triggers>
23.      <Trigger Property="Button.IsMouseOver" Value="True">
24.         <Setter TargetName="outerCircle" Property="Fill" Value="Black"> </Setter
     >
25.      </Trigger>
26.      <Trigger Property="Button.IsPressed" Value="True">
27.         <Setter Property="RenderTransform">
28.           <Setter.Value>
29.             <ScaleTransform ScaleX=".8" ScaleY=".8"> </ScaleTransform>
30.           </Setter.Value>
31.         </Setter>
32.         <Setter Property="RenderTransformOrigin" Value=".6,.6"> </Setter>
33.      </Trigger>
34.   </ControlTemplate.Triggers>
35.   </ControlTemplate>
36. </Grid.Resources>
37. <Button Template="{StaticResource buttonTemplate}">Click Me</Button>
38. </Grid>
```
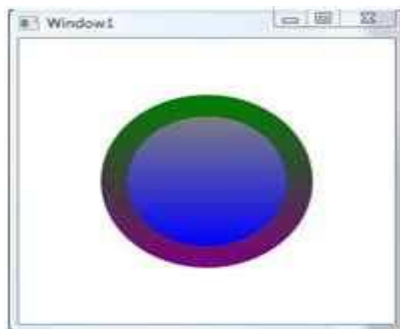
After MouseOver:
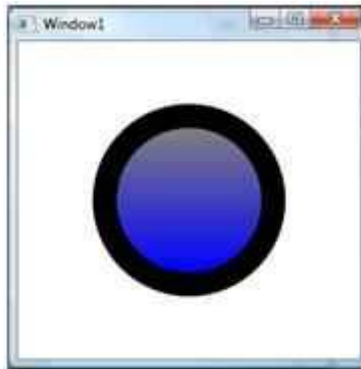
Question 24: How can we create Borderless Window in WPF?

Answer: We can create a borderless window in two ways.

Firstly, by writing a *WindowStyle property = None, SingleBorderWindow*, *ThreeDBorderWindow* or *ToolWindow* in the *<Window>* element.

1. <Window x:Class="WpfApp1.MainWindow" xmlns="http://schemas.microsoft.com/winf x/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" Titl e="MainWindow" Height="350" Width="525" WindowStyle="None">
2.   <Grid> </Grid>
3. </Window>

The second way is to open the Property Window, select the window style property to either None, SingleBorderWindow, ThreeDBorderWindow or ToolWindow.

Question 25: What is XAML in WPF and also explain the types of XAML?

Answer: Extensible Application Markup Language and pronounced "zammel" is a markup language used to instantiate .NET objects. Although XAML is a technology that can be applied to many different problem domains, its primary role in life is to construct WPF user interfaces.

Important task XAML performs as follows,

- **Wiring up an event handler:** Attaching event handler in the most cases, for example Click on Button is easy to do in Visual Studio. However once we understand how events are wired up in XAML, we'll be able create more sophisticated connections.

- **Defining resources:** Resources are the object which once we define in XAML can be re-used in the various places inside markup. Resources allow us to centralize and standardize formatting, and create nonvisual objects such as templates and animations.

- **Defining control template:** WPF controls are designed to be lookless, which means we can substitute our custom visuals in place of the standard appearance. To do so, we must create our own control template, which is nothing more than a block of XAML markup.

Type of XAML:

- **WPF XAML:** Encompasses the elements that describe WPF content, such as vector graphics, controls, and documents.

- **XPS XAML:** It is the part of WPF XAML that defines an XML representation for formatted electronic documents.

- **Silverlight XAML:** A subset of WPF XAML that's intended for Silverlight applications. Silverlight is a cross-platform browser plug-in that allows us to create rich web content.

- **WF XAML:** Encompasses the elements that describe Windows Workflow Foundation.

Question 26: How to set content of a NavigationWindow?

Answer: The Navigation Window class is derived from the Window class, so it inherits all the properties of Windows such as methods, properties and events. The navigation window provides backward and forward buttons for navigating to pages that we have visited before or have yet to visit.

Creating Navigation Paged Application: To create Navigation Window based applications, use a Navigation Window container instead of a Window container as shown in the following picture and the source is the property of the Navigation window and write the name of the page that you want to set as the home page as shown in the following picture:
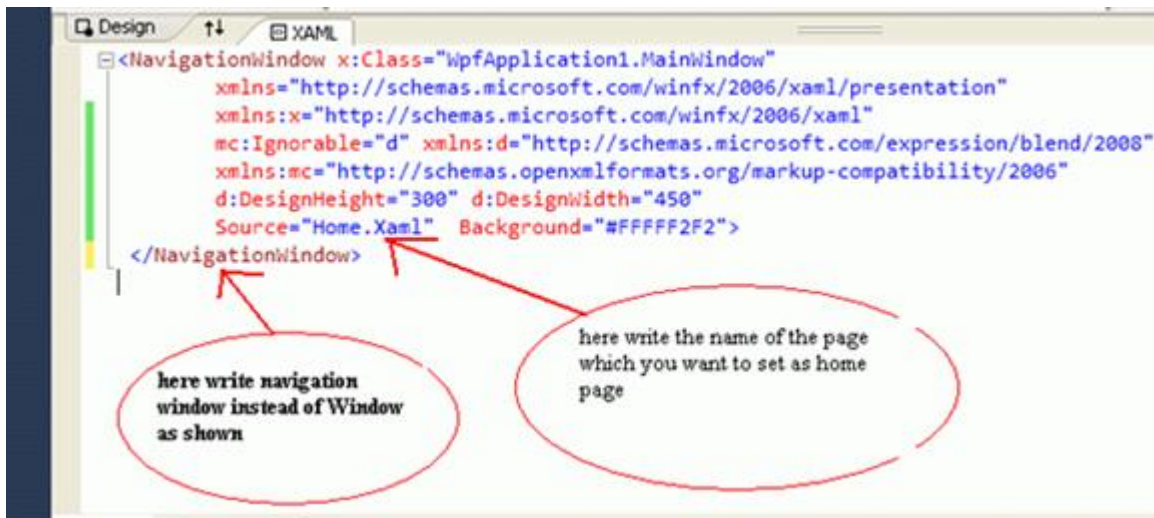
Question 27: What is a WPF Child Window?

Answer: A ChildWindow control is a light weight window that can be used as a child window or a popup control. The parent window is automatically disabled when a child window is active and modal. You can think of a ChildWindow as a custom modal or modeless dialog where you can place any child controls you want. However, the ChildWindow has several common Window properties.

Creating a ChildWindow:

The ChildWindow element represents a WPF ChildWindow control in XAML. The ChildWindow control is defined in the System.Windows.Controls namespace.

Creates a simple ChildWindow control with its Width, Height, Name, IsModal and Caption property.

1. `<wpfx:ChildWindow Name="PopupChildWindow" Caption="Child Window" Width="30 0" Height="200"IsModal="True" />`

We need to call the Show() method to make a Child Window visible.

1. `PopupChildWindow.Show();`

Question 28: What are the WPF Content Controls?

Answer: Content Controls are mainly parent containers to hold the content. It displays information to the user to be viewed but that generally won't be modified.

With content controls, the following aspects:

1. **Property:** A Property specifies the object's appearance or behavior. For example, the IsEnabled property specifies whether the user can interact with a control or not.

2. **Method:** A Method is a routine that a control executes to perform something. For example Coa unt Method counts the number of items available for the object or control.

3. **Event:** An event is related when a control raises one to let your application know that something has happened. For example TextBox raises a TextChanged event whenever its text changes.

Code Snippet:

<Button Name="btnAdd" Content="Add" Click="btnAdd_Click" />

Question 29: What is the Tab Control in WPF?

Answer: The Tab control is a common UI element that has been around for some time. It makes a convenient way to organize your window when there is more than could realistically fit and still be comprehensible. Tab Control is easier in Windows Presentation Foundation.

Two elements play main roles in building a tab control:

- TabControl and
- TabItem

TabControl is the container of one or more TabItem elements like as follows.

1. <TabControl>
2.   <TabItem Header="Tab 1">xyz</TabItem>
3.   <TabItem Header="Tab 2">abc</TabItem>
4. </TabControl>

In WPF, Tabs are very easy to implement. Create a new WPF Window, remove the default Grid tags, and add the following XAML:

1. <TabControl>
2.   <TabItem Header="Tab 1">xyz</TabItem>
3.   <TabItem Header="Tab 2">abc</TabItem>
4. </TabControl>

Question 30: How can I clip or crop an image?

Answer: Clipping a region is a process of displaying partial area of a region by setting the outline of a region. In WPF, the clipping has been extended to all elements that are inherited from UIElement that includes controls, images, panels, Windows, and Pages.

```
1.  Window.Clip>
2.     <EllipseGeometry Center="150,160" RadiusX="120" RadiusY="120" />
3.  </Window.Clip>
```

The following code snippet clips or crops an image to an ellipse.

```
1.  <Image Source="Garden.jpg">
2.     <Image.Clip>
3.        <EllipseGeometry Center="150,160" RadiusX="120" RadiusY="120" />
4.     </Image.Clip>
5.  </Image>
```



```
1.  private void ClipImage()
2.  {
3.     // Create a BitmapImage
4.     BitmapImage bmpImage = new BitmapImage();
5.     bmpImage.BeginInit();
6.     bmpImage.UriSource = new Uri(@ "C:\Images\Garden.jpg", UriKind.RelativeOrAbsol
       ute);
7.     bmpImage.EndInit();
8.     // Clipped Image
9.     Image clippedImage = new Image();
10.    clippedImage.Source = bmpImage;
11.    EllipseGeometry clipGeometry = new EllipseGeometry(new Point(150, 160), 120, 120)
       ;
12.    clippedImage.Clip = clipGeometry;
13.    LayoutRoot.Children.Add(clippedImage);
14. }
```

Question 31: What are Converters in WPF?

Answer: Converters provide substantial supremacy since they allow insertion of an object between a source and a target object. At a high level, converters are a chunk of custom code hooked up using the binding and the data will flow via that converter. So, whenever data is flown from a source to a target, one can change the value or can change the type of object that needs to be set on the target property.

So, whenever data travels from source to target, it can be transformed in two ways:

1. **Data value:** Here the transformation will be done with just the value by keeping the data type intact. For example, for number fields, you can transform a value from a floating point number to an integer by keeping the actual value as a float.

2. **Data type:** One can also transform the data type. For example, setting a style based on some Boolean flag.

Defining a converter:
Defining any converter requires implementation of an IValueConverter interface in a class. This interface has the following two methods:

1. **Convert:** Is called when data is flowing from a source to a target.

2. **ConvertBack:** Is called when data is flowing from a target to a source. It is basically useful in two-way binding scenarios.

How to use:
Once your class is part of a ResourceDictionary, you can point to the instance of the converter using the Converter property of Binding, along with a StaticResource markup extension.

Where to place converters:
To implement an IValueConverter one must create a class and then put the instance of that class in a ResourceDictionary within your UI.

Question 32: How does "UpdateSourceTrigger" affect bindings?

Answer: This is a property on a binding that controls the data flow from a target to a source and used for two-way data binding. The default mode is when the focus changes but there are many other options available.

Properties available with UpdateSourceTrigger:

- **Default:** This is the default value and it means a lost focus for most of the controls.
- **LostFocus:** Value update will be on hold until the focus moves out of the control.
- **PropertyChanged:** Value update will happen whenever a target property changes. It usually happens on every keystoke.
- **Explicit:** Used to defer source updates until the user does it forcibly by the click of a button or so.

Default vs LostFocus: Default and LostFocus means the same thing for most of the controls with the exception of DataGrid. For DataGrid:

- **Lost Focus**: Cell lost focus
- **Default:** Row lost focus

Code:

1. `<grid>`
2. `<Grid.ColumnDefinitions>`
3. `<ColumnDefinition Width="50*" />`
4. `<ColumnDefinition Width="50*" />`
5. `</Grid.ColumnDefinitions>`
6. `<TextBlock Text="Source:" Width="auto" />`
7. `<TextBox Name="SourceText" Width="160" Height="30" Margin="48,0,44,82" />`
8. `<TextBlock Text="Target:" Grid.Column="1" Width="auto" />`
9. `<TextBox Name="TargetText" Width="160" Height="30" Text="{Binding ElementName=SourceText, Path=Text,UpdateSourceTrigger=Default}" Grid.Column="1" Margin="44,0,47,82" />`
10. `</grid>`

Output:

When the user types into the source TextBox:
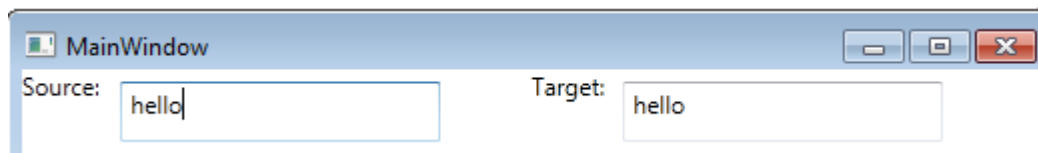
Question 33: What are various ways of doing alignment in WPF?

Answer: The FrameworkElement has two alignment properties: HorizontalAlignment and Vertical Alignment. The Horizontal Alignment property is a type of HorizontalAlignment enumeration and represents how a child element is positioned within a parent element horizontally.

The HorizontalAlignment enumeration has the four properties Left, Center, Right and Stretch. The Left, Center and Right properties sets a child element to left, center and right of the parent element. The Stretch property stretches a child element to fill the parent element's allocated layout space.

Example:

```
1.  <StackPanel Background="LightGray">
2.      <Button Name="Rect1" Background="LightBlue" Width="150" Height="50" Horizont
    alAlignment="Left" Content="Left Aligned" />
3.      <Button Name="Rect2" Background="LightGreen" Width="150" Height="50" Horizon
    talAlignment="Center" Content="Center Aligned" />
4.      <Button Name="Rect3" Background="LightCyan" Width="150" Height="50" Horizont
    alAlignment="Right" Content="Right Aligned" />
5.      <Button Name="Rect4" Background="LightPink" Height="50" HorizontalAlignment="
    Stretch" Content="Stretch Aligned" />
6.  </StackPanel>
```
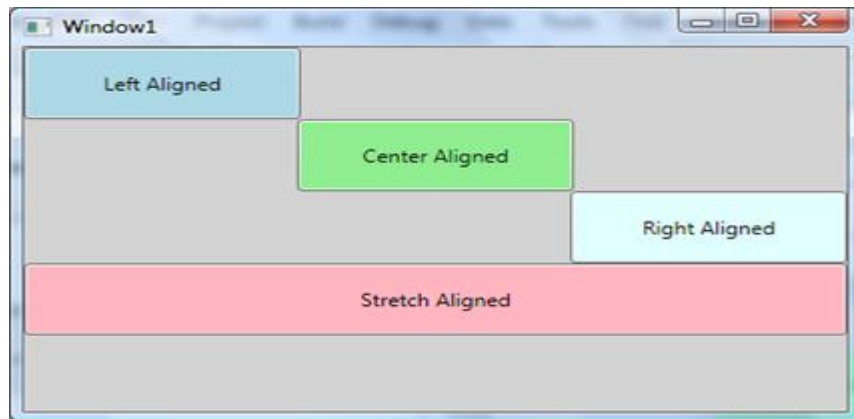
The **VerticalAlignment** property is a type of **HorizontalAlignment** enumeration and represents how a child element is positioned within a parent element vertically.

The **VerticalAlignment** enumeration has the four properties Top, Center, Bottom and Stretch. The Top, Center and Bottom properties set a child element to top, center or bottom of the parent element. The Stretch property stretches a child element to fill the parent element's allocated layout space vertically.

Example:

```
1.  <Grid>
2.    <Grid.ColumnDefinitions>
3.      <ColumnDefinition Width="100" />
4.      <ColumnDefinition Width="100" />
5.      <ColumnDefinition Width="100" />
6.      <ColumnDefinition Width="100" />
7.    </Grid.ColumnDefinitions>
8.    <Button Name="Button1" Background="LightBlue" Height="30" Width="100" VerticalAlignment="Top" Content="Left Aligned" />
9.    <Button Name="Button2" Background="LightGreen" Height="30" Width="100" Grid.Column="1" VerticalAlignment="Center" Content="Center Aligned" />
10.   <Button Name="Button3" Background="LightCyan" VerticalAlignment="Bottom" Height="30" Width="100" Grid.Column="2" HorizontalAlignment="Left" Content="Right Aligned" />
11.   <Button Name="Button4" Background="LightPink" Content="Stretch ligned" Width="100" Grid.Column="3" HorizontalAlignment="Stretch" />
12. </Grid>
```

Question 34: What is ToolTip and how we use it in WPF?

Answer: A ToolTip control shows some information or a hint about a control in a floating box when the mouse hovers over that control and it disappears when the mouse is moved away from that control.

Tool tips are used to help the users to learn the purpose of Controls in many Windows-based programs. Tooltips are nothing but small rectangles that appear when the user hovers the mouse pointer over the Control. The rectangle contains a short piece of text describing the control. Once displayed, the tips disappear when the user moves the mouse pointer away from the linked control or clicks a mouse button, or after a short delay.

ToolTip Example: Drag a Button control from the toolbox and drop it. Add a ToolTip property to the button with a message you want to show on mouse hover of the button. We can add a ToolTip in two ways. First the button shows the simplest way to display a ToolTip. The second method is preferred for creating a customized ToolTip.

1. <Button Content="Click Here" Margin="30" FontSize="16" ToolTip="Click Here"></Button>
2. <Button Content="Click Here" Margin="30" FontSize="16">
3.   <Button.ToolTip>
4.   <ToolTip> Click Here </ToolTip>
5.   </Button.ToolTip>
6. </Button>
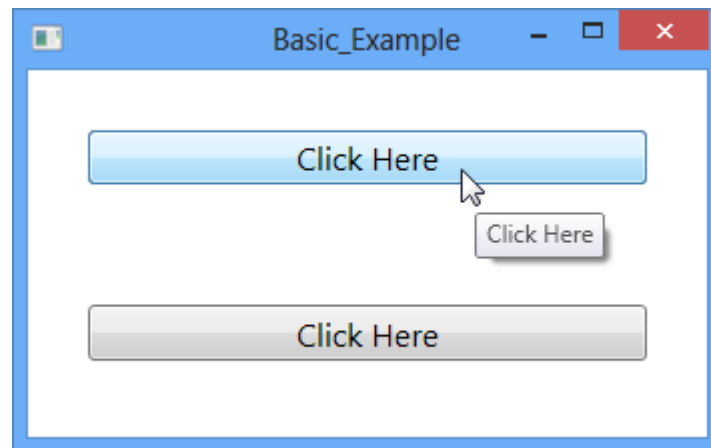
Question 35: How can ListBox are made to scroll smoothly?

Answer: A ListBox control is an items control that works as a ListBox control but only one item from the collection is visible at a time and clicking on the ListBox makes the collection visible and allows users to pick an item from the collection. Unlike a ListBox control, a ListBox does not have multiple item selection.

The ListBox element represents a ListBox control in XAML.

<ListBox></ListBox>

The Width and Height properties represent the width and the height of a ListBox. The x:Name property represents the name of the control, which is a unique identifier of a control. The Margin property sets the location of a ListBox on the parent control. The HorizontalAlignment and VerticalAlignment properties are used to set horizontal and vertical alignments.

The code sets the vertical and horizontal alignment of the ListBox and sets the margin.

1. <ListBox x:Name="ListBox1" Width="200" Height="200"
2.   VerticalAlignment="Top" HorizontalAlignment="Left"
3.   Margin="10,10,0,0">
4. </ListBox>

Question 36: What is a Popup window and how to open and close the popup window?

Answer: A popup window is a window that floats over a page or window providing functionality for some quick action. For example, a login control on a page or window or an animated popup tip of a control.

The Popup element of XAML represents a WPF Popup control.

<Popup></Popup>

The Width and the Height properties represent the width and the height of a Popup. The Name property represents the name of the control that is a unique identifier of a control. The Margin property is used to set the location of a Popup on the parent control. The HorizontalAlignment and VerticalAlignment properties are used to set horizontal and vertical alignments.

The following code snippet sets the name, height, and width of a Popup control. The code also sets horizontal alignment to left and vertical alignment to top.

1. <Popup Margin="10,10,0,13" Name="Popup1" HorizontalAlignment="Left"
2. VerticalAlignment="Top" Width="194" Height="200" />

Open a Popup: The Popup Control displays its contents when IsOpen set true.

1. privatevoid Popup_Ok_Click(object sender, RoutedEventArgs e)
2. {

3.   Popup1.IsOpen = true;
4.   }

Where popup1 is the name of the popup control.

Closing a Popup: The Popup Control displays its contents when IsOpen is set to false.

1.   privatevoid Popup_Ok_Click(object sender, RoutedEventArgs e)
2.   {
3.   Popup1.IsOpen = false;
4.   }

Question 37: What is a Decorators class in WCF?

Answer: Decorator class that is a simple base class of WPF layout controls. Decorators have a single child control to which they apply a single child control.

Add a click event to the button. Replace the Button's XAML with the following:

1.   <Border Name="MyBorder" BorderBrush="Black" BorderThickness="4" CornerRadius= "10">
2.   <Button Click="Button_Click">Hello, world!</Button>
3.   </Border>

When the program is executed the window that appears similar to the image below:

Question 38: What are the advantages and disadvantages of WPF?

Answer: Advantages of WPF-

- **Tight multimedia integration:** To use 3-D graphics, video, speech, and rich document viewing in Windows 32 or Windows Forms applications, you would need to learn several independent technologies and blend them together without much built-in support. WPF applications allow you to use all these features with a consistent programming model.

- **Resolution independence:** WPF lets you shrink or enlarge elements on the screen, independent of the screen's resolution. It uses vector graphics to make your applications resolution-independent.

- **Hardware acceleration:** WPF is built on top of Direct3D, which offloads work to graphics processing units (GPUs) instead of central processor units (CPUs). This provides WPF applications with the benefit of hardware acceleration, permitting smoother graphics and enhanced performance.

- **Declarative programming:** WPF uses Extensible Application Markup Language (XAML) declarative programming to define the layout of application objects and to represent 3-D models, among other things. This allows graphic designers to directly contribute to the look and feel of WPF applications.

- **Rich composition and customization:** WPF controls are easily customizable. You need not write any code to customize controls in very unique ways. WPF also lets you create skins for applications that have radically different looks.

- **Easy deployment:** WPF provides options for deploying traditional Windows applications (using Windows Installer or Click Once). This feature is not unique to WPF, but is still an important component of the technology.

- **Culturally aware controls:** Static text in controls and the return data for the String function are modified according to the culture and language specified by the end user's operating system.

Disadvantages of WPF:

- WPF's in-box control suite is far more limited than that of WinForms.

- There's greater support in the 3rd-party control space for WinForms. (That's changing, but for now by advantage of time, WinForms has greater support in the community).

- Most developers already know WinForms; WPF provides a new learning curve.

- WPF will not run on Windows 2000 or lower.

- No MDI child mode.

Question 39: What are the WPF assemblies and Namespace?

Answer:

WPF Assembly	Meaning in Life
WindowsBase.dll	Defines the base infrastructure of WPF, including dependency properties support. While this assembly contains types used within the WPF framework, the majority of these types can be used within other .NET applications.
PresentationCore.dll	This assembly defines numerous types that constitute the foundation of the WPF GUI layer.
PresentationFoundation.dll	This assembly—the 'meatiest' of the three—defines the WPF controls types, animation and multimedia support, data binding support, and other WPF services. For all practical purposes, this is the assembly you will spend most of your time working with directly.

Although these assemblies provide hundreds of types within numerous namespaces, consider this partial list of WPF namespaces:

- You will encounter other namespaces during the remainder of this class.
- Again, consult the .NET Framework SDK documentation for full details.

WPF Namespace	Meaning in Life
System.Windows	Here you will find core types such as Application and Window that are required by any WPF desktop project.
System.Windows.Controls	Here you will find all of the expected WPF widgets, including types to build menu systems, tool tips, and numerous layout managers.
System.Windows.Markup	This namespace defines a number of types that allow XAML markup and the equivalent binary format, BAML, to be parsed.
System.Windows.Media	Within these namespaces you will find types to work with animations, 3D rendering, text rendering, and other multimedia primitives.
System.Windows.Navigation	This namespace provides types to account for the navigation logic employed by XAML browser applications / desktop navigation apps.
System.Windows.Shapes	This namespace defines basic geometric shapes (Rectangle, Polygon, etc.) used by various aspects of the WPF framework.

Question 40: What is the difference between WPF and Silverlight?

Answer: Silverlight and Windows Presentation Foundation (WPF) are 2 different products from Microsoft, but have lot of overlap. Silverlight is a subset of WPF in terms of features and functionality. Silverlight was of course known as WPF/E where E means everywhere. Both use XAML, a form of XML to define controls but WPF is purely for windows while Silverlight runs in the browser on Windows and Macs. So, here are the differences between WPF and Silverlight:

1. Silverlight is meant to be used online, while WPF is for local use.
2. Silverlight lacks access to local resources, while WPF can utilize local resources.
3. Silverlight only has perspective 3D support, while WPF is capable of full 3D images.
4. Silverlight does not support some of the more advanced concepts of WPF such as controls and templating.
5. Silverlight integrates right into an HTML page whereas WPF XAML files have to be loaded via a frame if they want to mix with HTML content.

Question 41: What is WPF accesstext Control?

Answer: The AccessText control in WPF converts a character preceded by an underscore to an Access Key. The Access Key is registered and therefore raises an event when pressed.

Creating an AccessText: The AccessText element represents an AccessText control in XAML:

AccessText>_Click Me</AccessText>

The following code snippet adds an AccessText to a Button control. The button control click event will be raised when you select the ALT+C keys on the keyboard.

1. <Button Name="Button1" Width="120" Height="50" Margin="33,70,59,124" FontSize=" 16" Click="Button1_Click">
2.   <AccessText>_Click Me</AccessText>
3. </Button>

If there are multiple underscore characters, only the first one is converted into an AccessKey; the other underscores appear as normal text. If the underscore that you want converted to the access key is not the first underscore, use two consecutive underscores for any underscores that precede the one that you want to convert.

Question 42: What are the different types of brushes that WPF offers?

Answer: Brushes and pens are objects used to draw and fill graphics objects. WPF and XAML work together and there is an WPF class corresponding to each XAML control. For example, <SolidColorBrush> tag in XAML and SolidColorBrush class in WPF, both represent the solid color brush. You can create and use brushes in both ways seperately or mix them.

In XAML and WPF model provides the following brush objects:

1. SolidColorBrush
2. LinearGradientBrush
3. RadialGradientBrush
4. DrawingBrush
5. Visual Brush
6. ImageBrush

Question 43: How many types of Bitmap Effects in WPF?

Answer: Bitmap effects are simple pixel processing operations. A bitmap effect takes a BitmapSource as an input and produces a new BitmapSource after applying the effect.

Bitmap effects enable designers and developers to apply visual effects to rendered Microsoft Windows Presentation Foundation (WPF) content. For example, bitmap effects allow you to easily apply a DropShadowBitmapEffect effect or a blur effect to an image or a button. The unmanaged APIs provide an extensible framework for independent hardware vendors (IHVs) to develop custom effects to use in WPF applications.

The following are the available Bitmap Effects in WPF:

1. BlurBitmapEffect
2. OuterGlowBitmapEffect
3. DropShadowBitmapEffect
4. BevelBitmapEffect
5. EmbossBitmapEffect

Each bitmap effect has properties that can control the filtering properties, such as Radius of BlurBitmapEffect.

Question 44: What is tree view in WPF? How we can delete a tree view in WPF?

Answer: A TreeView represents data in a hierarchical view in a parent child relationship where a parent node can be expanded or collapse. The left side bar of Windows Explorer is an example of a TreeView.

The TreeView tag represents a WPF TreeView control in XAML.

<TreeView></TreeView>

Adding TreeView Items: A TreeView control hosts a collection of TreeViewItem. The Header property is the text of the item that is displayed on the view. The following code snippet adds a parent item and six child items to a TreeView control.

```
1. <TreeView Margin="10,10,0,13" Name="TreeView1" HorizontalAlignment="Left" VerticalAlignment="Top" Width="194" Height="200">
2.    <TreeViewItem Header="Cold Drinks">
3.      <TreeViewItem Header="Coke"></TreeViewItem>
4.      <TreeViewItem Header="Pepsi"></TreeViewItem>
5.      <TreeViewItem Header="Orange Juice"></TreeViewItem>
6.      <TreeViewItem Header="Milk"></TreeViewItem>
7.      <TreeViewItem Header="Iced Tea"></TreeViewItem>
8.      <TreeViewItem Header="Mango Shake"></TreeViewItem>
9.    </TreeViewItem>
10. </TreeView>
```
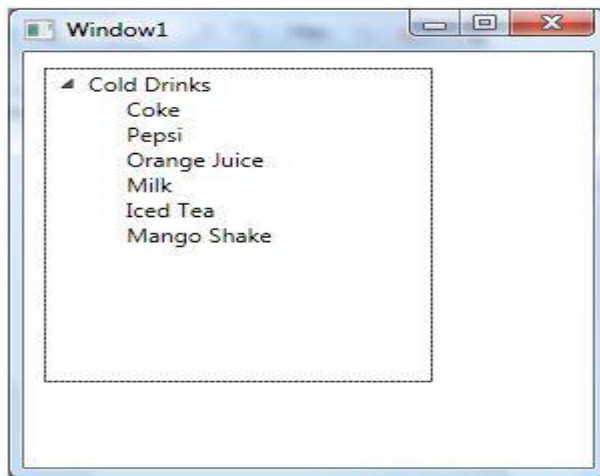
Deleting TreeView Items: We can use TreeView.Items.Remove or TreeView.Items.RemoveAt method to delete an item from the collection of items in the TreeView. The RemoveAt method takes the index of the item in the collection.

```
1.  <Button Height="23" Margin="226,14,124,0" Name="DeleteButton"
2.  VerticalAlignment="Top" Click="DeleteButton_Click">
3.  Delete Item</Button>
4.  private void DeleteButton_Click(object sender, RoutedEventArgs e)
5.  {
6.     TreeView1.Items.RemoveAt
7.     (TreeView1.Items.IndexOf(TreeView1.SelectedItem));
8.  }
```

The above code removes root items from the TreeView, not the subitems. To remove sub items, first we need to find the selected item and then we need to call TreeViewItem.Items.RemoveAt method.

Question 45: What are the Commands in WPF?

Answer: Commands have several purposes. The first purpose is to separate the semantics and the object that invokes a command from the logic that executes the command. This allows for multiple and disparate sources to invoke the same command logic and it allow the command logic to be customized for different targets. For example, the editing operations Copy, Cut, and Paste, which are found in many applications, can be invoked by using different user actions if they are implemented by using commands. An application might allow a user to cut selected objects or text by clicking a button, choosing an item in a menu, or using a key combination, such as CTRL+X. By using commands, you can bind each type of user action to the same logic.

Commands are used to share grouped actions within an application in different ways. Sometimes we need to perform the same activity; WPF provides us a feature called Command to make our work easier and faster.

There are basically four types of Commands:

- Application Commands
- Edit Commands
- Component Commands
- Media Commands

XAML:

```
1.  <StackPanel>
2.    <Menu>
3.      <MenuItem Command="ApplicationCommands.Paste" /> </Menu>
4.    <TextBox />
5.  </StackPanel>
```

C#:

```
1.  / Creating the UI objects
2.  StackPanel mainStackPanel = new StackPanel();
3.  TextBox pasteTextBox = new TextBox();
4.  Menu stackPanelMenu = new Menu();
5.  MenuItem pasteMenuItem = new MenuItem();
6.
7.  // Adding objects to the panel and the menu
8.  stackPanelMenu.Items.Add(pasteMenuItem);
9.  mainStackPanel.Children.Add(stackPanelMenu);
10. mainStackPanel.Children.Add(pasteTextBox);
11.
12. // Setting the command to the Paste command
13. pasteMenuItem.Command = ApplicationCommands.Paste;
14.
15. // Setting the command target to the TextBox
16. pasteMenuItem.CommandTarget = pasteTextBox;
```

Question 46: What are the properties of canvas panel in WPF?

Answer: The Canvas is the most basic layout panel in WPF. It's child elements are positioned by **explicit coordinates**. The coordinates can be specified **relative to any side** of the panel usind the Canvas.Left, Canvas.Top, Canvas.Bottom andCanvas.Right attached properties.

The panel is typically used to group 2D graphic elements together and not to layout user interface elements. This is important because specifing absolute coordinates brings you in trouble when you begin to resize, scale or localize your application. People coming from WinForms are familiar with this kind of layout - but it's not a good practice in WPF.

In WPF, a Canvas Panel has a very simple layout. We can position any object using its two properties:

- Canvas.Left
- Canvas.Top

Note: A Canvas panel does not resize automatically when our normal application resizes in the browser.

Here is an example of a Canvas panel:

1. <Canvas Background="Pink" Width="250" Height="100">
2. </Canvas>
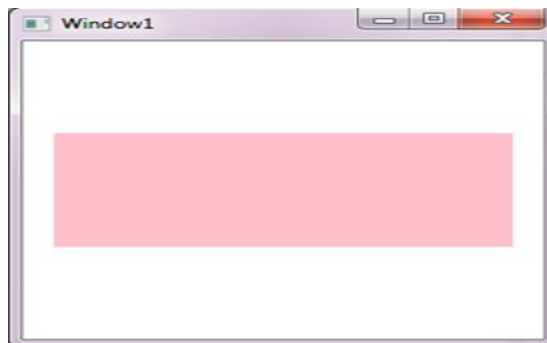
Question47: What is WPF TextBlock?

Answer: The TextBlock control is one of the most fundamental controls in WPF, yet it's very useful. It allows you to put text on the screen, much like a Label control does, but in a simpler and less resource demanding way. A common understanding is that a Label is for short, one-line texts (but may include e.g. an image), while the TextBlock works very well for multiline strings as well, but can only contain text (strings). Both the Label and the TextBlock offers their own unique advantages, so what you should use very much depends on the situation.

Creating a TextBlock: The TextBlock element represents a WPF TextBlock control in XAML.

<TextBlock/>

The Width and Height attributes of the TextBlock element represent the width and the height of a TextBlock. The Text property of the TextBlock element represents the content of a TextBlock. The Name attribute represents the name of the control, which is a unique identifier of a control. The Foreground property sets the foreground color of contents. This control does not have a Background property.

1. <TextBlock Name="TextBlock1" Height="30" Width="200"
2.   Text="Hello! I am a TextBlock." Foreground="Red">
3. </TextBlock>

The **output** looks like the following,

Hello! I am a TextBlock.

Question 48: What is XAML TabControl in WPF?

Answer: A Tab Control has tab items and each tab item represents a container that is used to host other controls.

The WPF TabControl allows you to split your interface up into different areas, each accessible by clicking on the tab header, usually positioned at the top of the control. Tab controls are commonly used in Windows applications and even within Windows' own interfaces, like the properties dialog for files/folders etc.

Just like with most other WPF controls, the TabControl is very easy to get started with. Here's a very basic example:

1. <Window x:Class="WpfTutorialSamples.Misc_controls.TabControlSample" <Window x: Class="WpfTutorialSamples.Misc_controls.TabControlSample" xmlns="http://schemas.m icrosoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winf x/2006/xaml" Title="TabControlSample" Height="200" Width="250">
2.   <Grid>
3.   <TabControl>
4.   <TabItem Header="General">
5.   <Label Content="Content goes here..." /> </TabItem>
6.   <TabItem Header="Security" />
7.   <TabItem Header="Details" /> </TabControl>
8.   </Grid>
9. </Window>
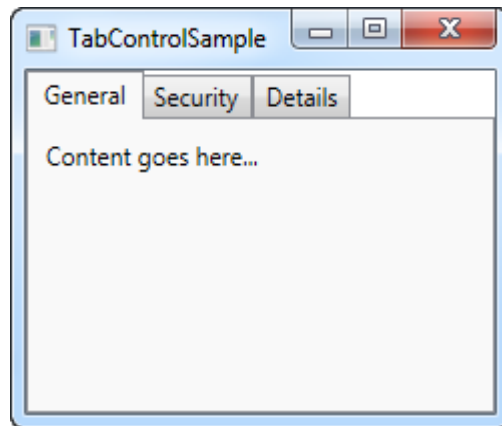
Question 49: What is Virtualization in WPF?

Answer: Virtualization technique in WPF improves the rendering performance of UI elements. By applying virtualization, the layout system ensures that only the visible items of a container are rendered on the screen. For example, a list control may have thousands of items but virtualization will reduce the rendering to the visible items only.

VirtualizingStackPanel: The VirtualizingStackPanel control in WPF is used to implement virtualization.The IsVirtualizing property of the VirtualizingStackPanel activates the virtualization. By default, the IsVirtualizing property is set to true. When IsVirtualizing is set to false, a VirtualizingStackPanel behaves the same as an ordinary StackPanel.

<VirtualizingStackPanel Width="300" Height="200" />

The VirtualizingStackPanel.VirtualizationMode property has two values, Standard and Recycling. The default value of VirtualizationMode is Standard and means that the VirtualizingStackPanel creates an item container for each visible item and discards it when it is no longer needed (such as when the item is scrolled out of view). When an ItemsControl contains many items, the process of creating and discarding item containers can degrade performance. In that case, using the Recycling reuses item containers instead of creating a new one each time.

Question 50: What is DockPanel Control in WPF?

Answer: A DockPanel is a panel where each child element docks to one of the four edges. DockPanel enables docking of child elements to an entire side of the panel streching it to fill the entire height or width.

The DockPanel makes it easy to dock content in all four directions (top, bottom, left and right). This makes it a great choice in many situations, where you want to divide the window into specific areas, especially because by default, the last element inside the DockPanel, unless this feature is specifically disabled, will automatically fill the rest of the space (center).

1. <Window x:Class="DockPanelExample1.MainWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" Title="DockPanel Example" Height="350" Width="525">
2.   <DockPanel>
3.   <Button Content="Button1"></Button>
4.   <Button Content="Button2"></Button>
5.   <Button Content="Button3"></Button>
6.   <Button Content="Button4"></Button>
7.   <Button Content="Button5"></Button>
8.   <Button Content="Button6"></Button>
9.   </DockPanel>
10. </Window>

Preview:
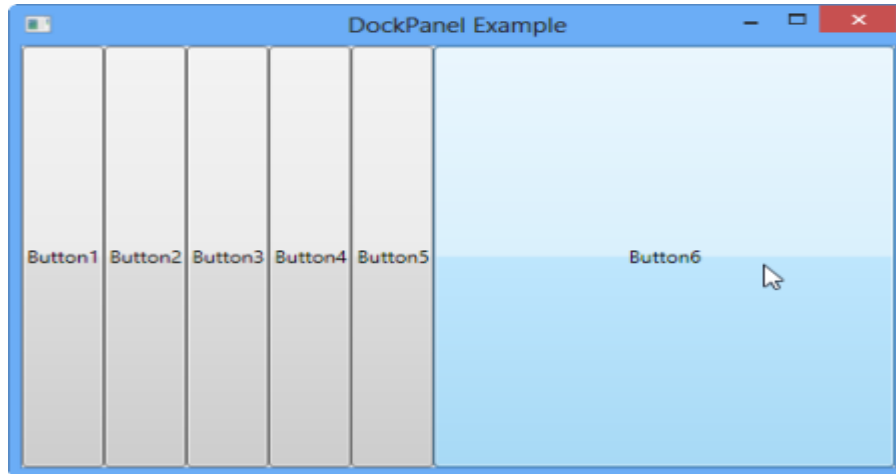
Question 51: What is XAML StatusBar?

Answer: XAML StatusBar represents a status bar. A StatusBar is a horizontal window that usually sits at the bottom of a window to display various kinds of status information of an application.

The **StatusBar** element in **XAML** represents a **WPF StatusBar** control.

<StatusBar></StatusBar>

The Width and Height properties represent the width and the height of a StatusBar. The Name property represents the name of the control that is a unique identifier of a control.

The following code snippet creates a StatusBar control and set its content to some text.

1. <StatusBar Name="McSBar" Height="30" VerticalAlignment="Bottom"
2.   Background="LightBlue" >

3. This is a status bar
4. </StatusBar>

The output looks as in the following figure:
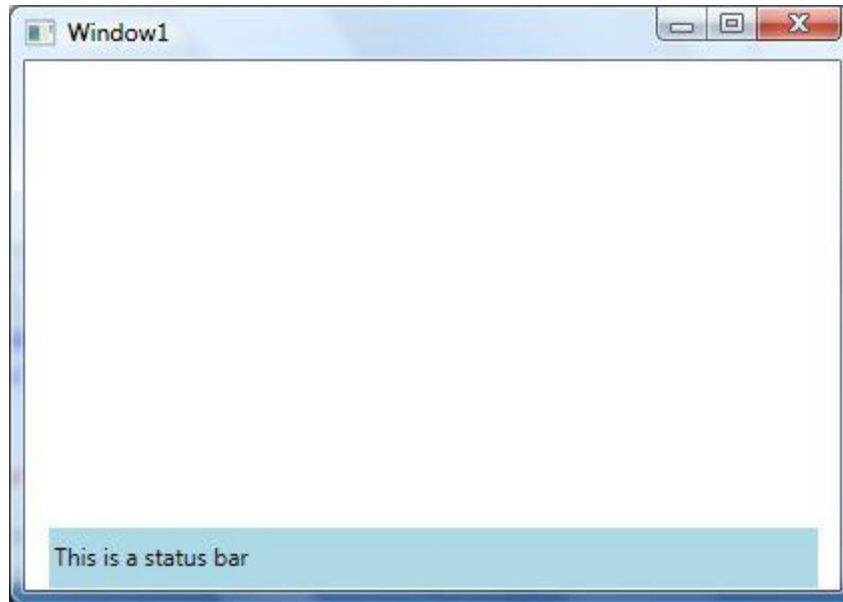
Question 52: Describe Polyline in WPF?

Answer: A polyline is an object in AutoCAD that consists of one or more line (or arc) segments. A rectangle is an example of a polyline that you are already familiar with. As you've seen, it is one object that can be modified and worked with easier than four separate lines.

In other words, a polyline is a collection of connected straight lines. The Polyline object represents a polyline shape and draws a polyline with the given points. The Points property represents the points in a polyline. The Stroke property sets the color and StrokeThickness represents the width of the line of a polyline.

Creating a Polyline: The Polyline element in XAML creates a polyline shape. The following code snippet creates a polyline by setting its Points property. The code also sets the black stroke of width 4.

```
1.  <Polyline
2.    Points="10,100 100,200 200,30 250,200 200,150"
3.    Stroke="Black"
4.  StrokeThickness="4" />
```

The Polyline element in XAML creates a polyline shape. The following code snippet creates a polyline by setting its Points property. The code also sets the black stroke of width 4.

```
1.  <Polyline
2.    Points="10,100 100,200 200,30 250,200 200,150"
3.    Stroke="Black"
4.  StrokeThickness="4" />
```

Output:
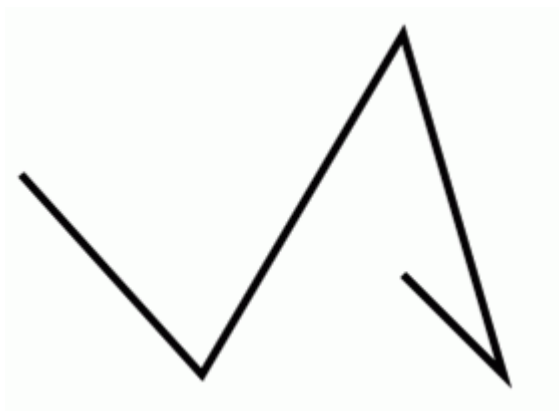
Chapter8
Most Asked SQL Server Interview Questions and Answers

Question 1: What is a database?

Answer: A database is described as an organized way of collection of DATA. It is the collection of schemes, tables, queries, reports, views and other objects.

Syntax: CREATEDATABASEDatabaseName

Example: CREATEDATABASE Student
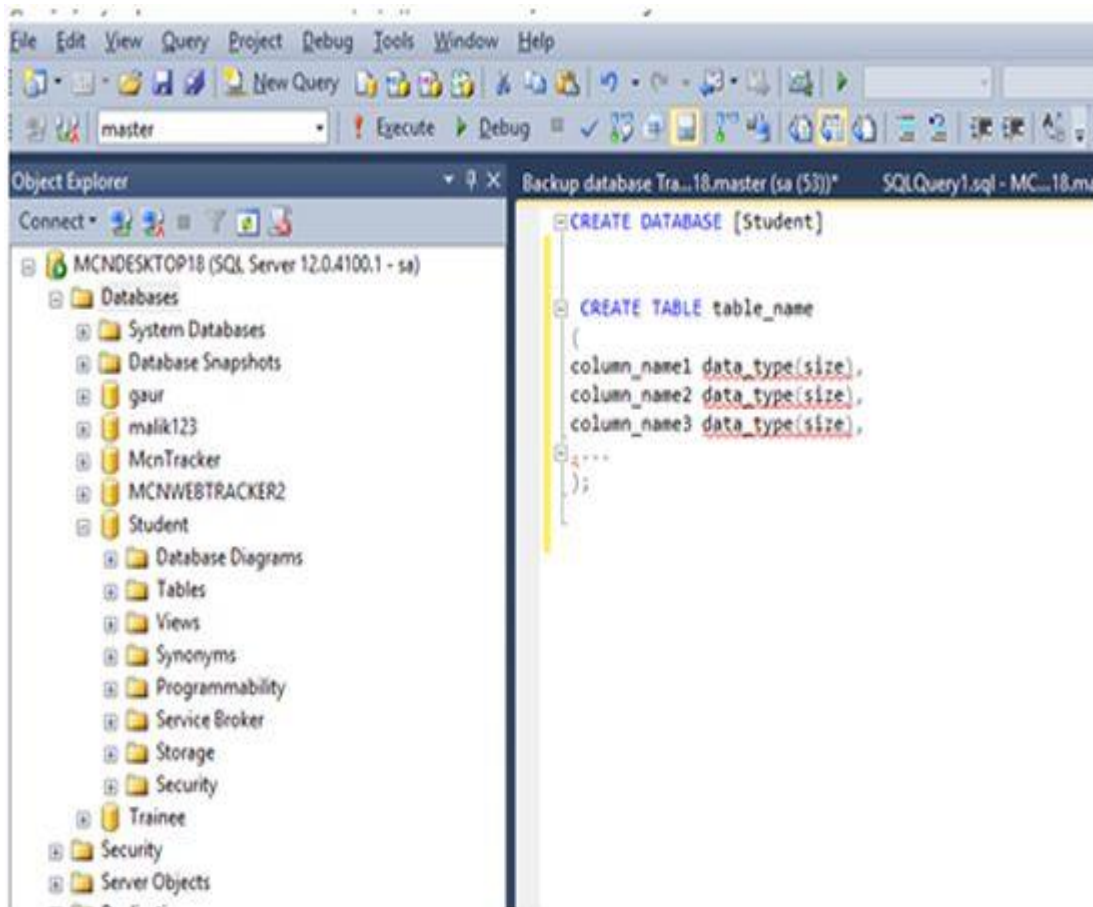
or you can Create Database through Design/ Wizard form by right clicking on DATABASE option- New Database.
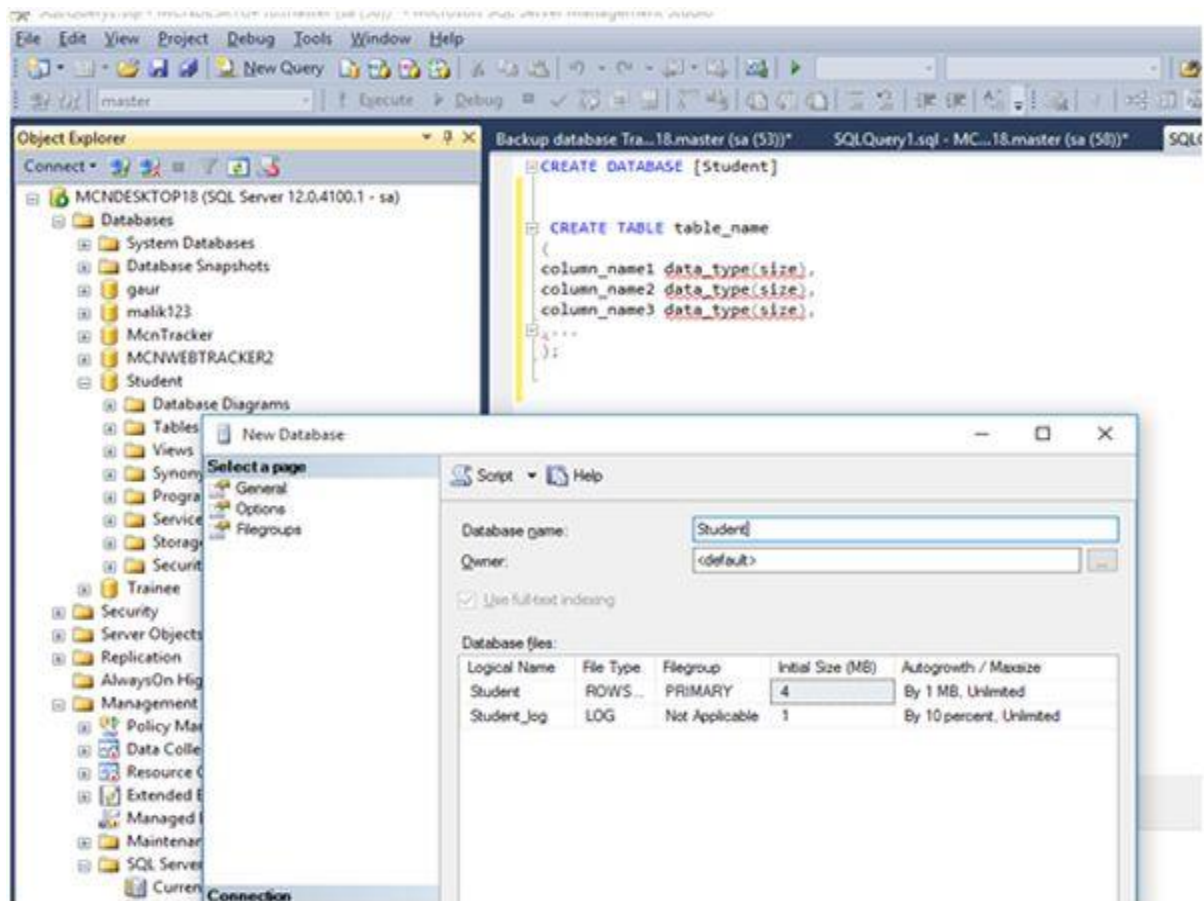
Question 2: What is SQL?

Answer: Structured Query Language, also known as SQL, is a programming language designed for managing Relational Database Management Systems (RDBMSs). SQL is an International Organization for Standardization (ISO) standard.

In RDBMS all the data is stored in tables with each table consisting of rows and columns.

Example of Sql Server 2014 SQL format:
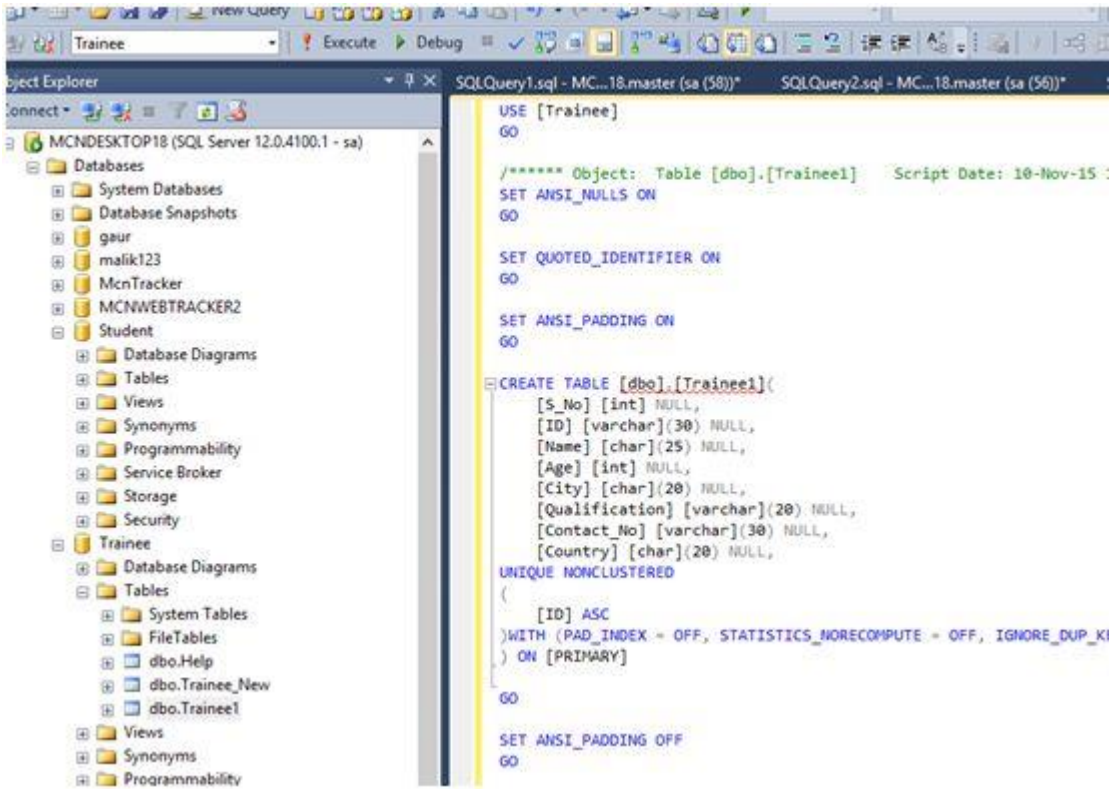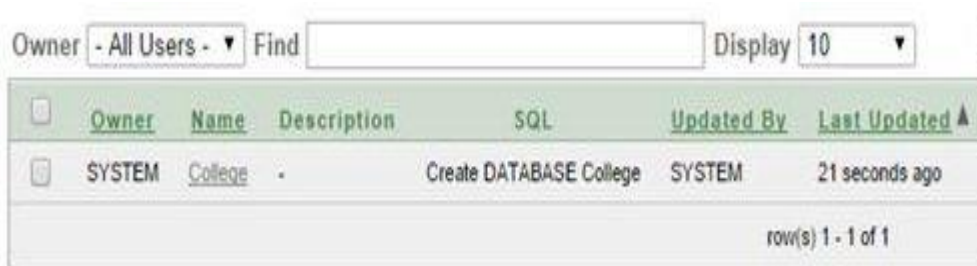
Example of Oracle SQL format below:

Create database:
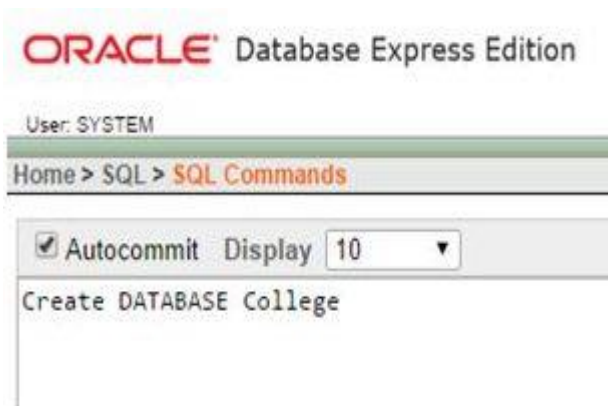
Output: Here we can see our database is created.

Question 3: What is PL/SQL?

Answer: PL/SQL Control Statements in Oracle.
Control Statements,

- Control statements are very important in PL/SQL.
- Control Statements are elements in a program that control the flow of program execution.
- The syntax of control statements are similar to regular English and are very similar to choices that we make every day.
- Branching statements are as follows:

  o If statement
  o If - THEN - ELSE
  o Nested IF
  o Branching with logical connectivity
  o While
  o For Loop

Question 4: What is the difference between SQL and PL/SQL?

Answer: SQL: It is referred as Structured Query Language.

- Only simple IF / Else statements.
- Through SQL you can interact with database through ADO.NET
- In SQL you can execute a line of code
- It can run only on windows
  PL/SQL: It is referred as Procedure Language / Structure Query Language:
- In PL/SQL you can execute a block of code not a single line of code.
- Deep control statements
- It can run in UNIX also.
- PL/SQL language includes object oriented programming techniques such as encapsulation, function overloading, and information hiding (all but inheritance).

Question 5: What is RDBMS?

Answer: RDBMS: It is referred as Relation Database Management Systems (RDBMS). RDBMS possesses a set of the below given characteristics:

- Write-intensive operations: The RDBMS is frequently written to and is often used in transaction-oriented applications.
- Data in flux or historical data: The RDBMS is designed to handle frequently changing data. Alternatively, RDBMS can also store vast amounts of historical data, which can later be analyzed or "mined".
- Application-specific schema: The RDBMS is configured on a per-application basis and a unique schema exists to support each application.
- Complex data models. The relational nature of the RDBMS makes it suitable for handling sophisticated, complex data models that require many tables, foreign key values, complex join operations, and so on.
- Data integrity: The RDBMS features many components designed to ensure data integrity. This includes rollback operations, referential integrity, and transaction-oriented operations.

Question 6: What is a database table?

Answer: Database table: Table contains records in the form of rows and columns. A permanent table is created in the database you specify and remains in the database permanently, until you delete it.

Syntax:

1. Create table TableName (ID INT, NAME VARCHAR(30))
2. Drop syntax: drop table TableName
3. Select Syntax: Select * from TableName

Question 7: How to create a table in SQL?

Answer: SQL provides an organized way for table creation.

Syntax:

1. Create table TableName (columnName1 datatype, columnName2 datatype)

The following is an example of creating a simple table-

1. create table Info
2. (
3.   Name varchar(20),
4.   BirthDate date,
5.   Phone nvarchar(12),
6.   City varchar(20)
7.)
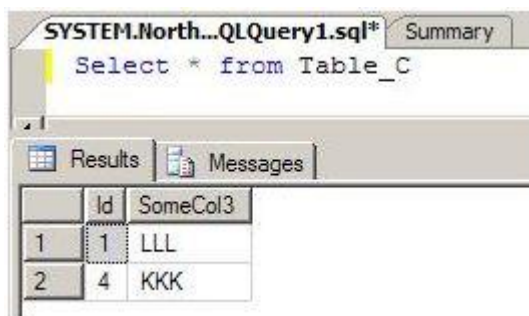
Question 8: How to delete a table in SQL Server?

Answer: Delete Data Record from Database Table and deleting an existing table by the following method:

Syntax: To delete all table records of a table:

1. **Delete TableName**
2. **DELETE info**
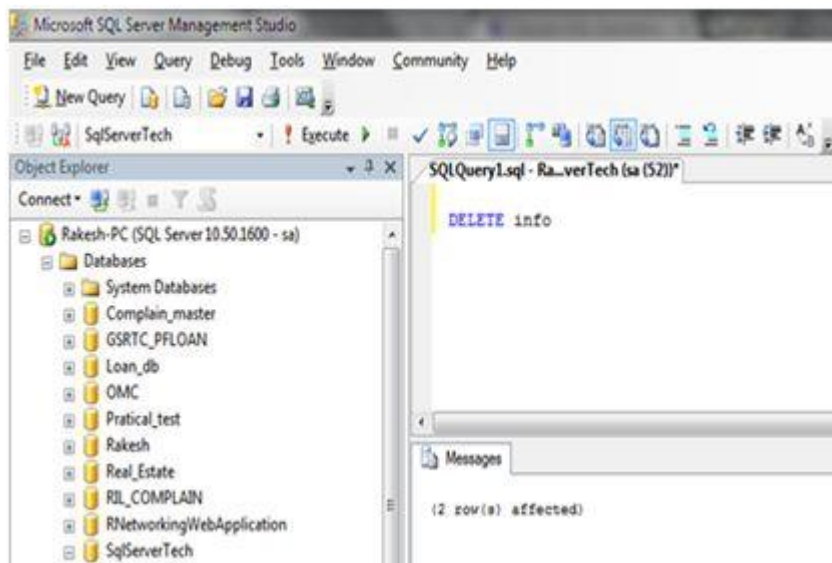
Question 9: How to update a database table using SQL?

Answer: To update an existing Table we use SQL Command UPDATE: It will update the records as per user defined query/need.

Syntax:

1. Update TableName SET ColumnName = NewData where Condition
2. Update info Set City = 'Baroda' where id = 2
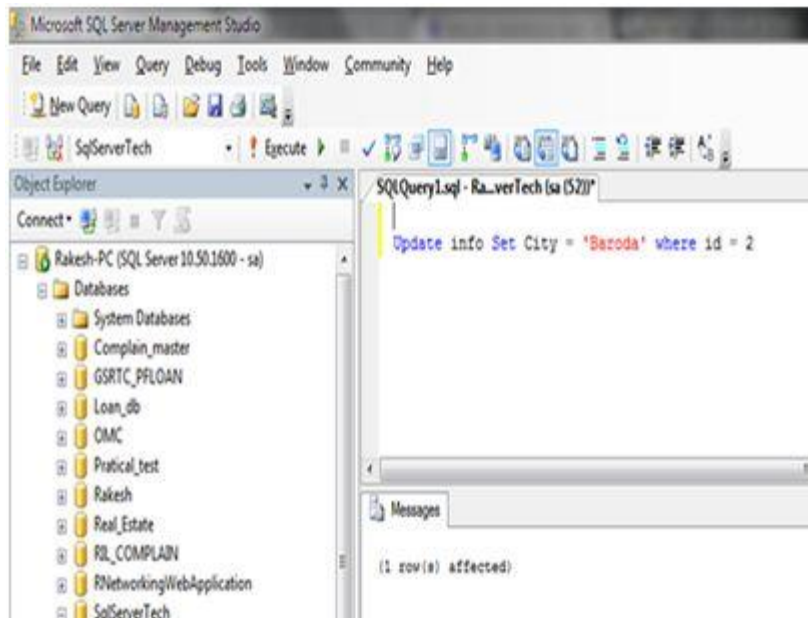
Question 10: What is a database relationship?

Answer: Relationships are created by linking the column in one table with the column in another table. There are four different types of relationship that can be created.

The relationships are listed below:

1. One to One Relationship
2. Many to One relationship
3. Many to Many relationship
4. One to One relationship

One to Many & Many to One Relationship:

- For a One to many relationships, a single column value in one table has one or more dependent column values in another table. Look at the following diagram:
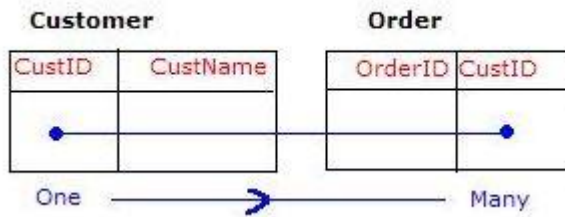
Many to Many Relationship:
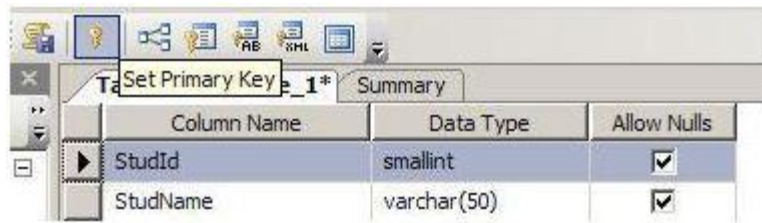
The third table acts as a bridge between the tables that want to establish a Many to Many relationship. The bridge table stores the common information between Many to Many relationship tables. Have a look at the following diagram:
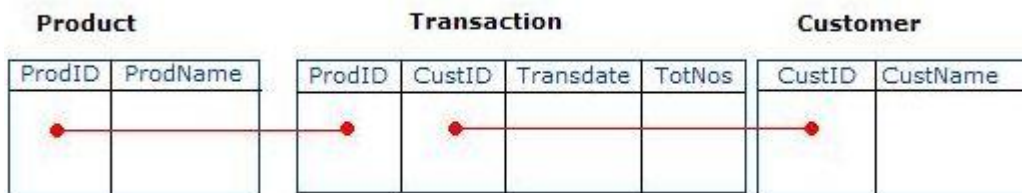
Question 11: What is a primary key of a database?

Answer: Primary key:-

A table column with this constraint is called the key column for the table. This constraint helps the table to make sure that the value is not repeated and also that there are no null entries.

Now this column does not allow null values and duplicate values. You can try inserting values to violate these conditions and see what happens. A table can have only one Primary key. Multiple columns can participate on the primary key.

Example:

Question 12: What is a foreign key of a database?

Answer: To define the relationship between two tables (one is called parent and the other one is the child table) connected by columns, a foreign key constraint is used. In this constraint the values of the child table must appear in the parent table, which means that for a foreign key, one table should point to a Primary Key in another table. A table can have multiple foreign keys and each foreign key can have a different referenced table.

Example: To understand the foreign key clearly let's assume the following two tables:

1. CUSTOMER {Cust_ID, Cust_Name, Age, ContactNo, Gender, Address}
2. VENDOR {Vend_ID, Vend_Name, Cust_ID}

Customer Table:

Cust_ID	Cust_Name	Age	ContactNo	Gender	Address
1	Joseph	54	9965234525	Male	Cross Road, 10
2	Sheena	42	7851356434	Female	Link Road, 15B
3	Mohan	36	8325576237	Male	Lodhi Colony, 111Z

Vendor Table:

Vend_ID	Vend_Name	Cust_ID
111	Ravish	3
222	Athena	3
333	Kumar Brothers	1

Example: Foreign Key Constraint while using CREATE TABLE statement.

Syntax:

1. CREATE TABLE table_name
2. (
3.   Col1 datatype NOT NULL,
4.   Col2 datatype NOT NULL,
5.   Col3 datatype NOT NULL,
6.   CONSTRAINT FK_Column FOREIGN KEY(Col1, Col2, Col3) REFERENCES parent _table(Col1, Col2, Col3)
7.);

AT SINGLE COLUMN LEVEL

Question 13: What is database normalization?

Answer: Database normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency. Normalization usually involves dividing large tables into smaller (and less redundant) tables and defining relationships among them. Normalization is a bottom-up technique for database design.

The evolution of Normalization theories is illustrated below:

- First Normal Form (1NF)
- Second Normal Form (2NF)
- Third Normal Form (3NF)
- Boyce-Codd Normal Form (BCNF)
- 4th Normal Form
- 5th Normal Form
- 6th Normal Form

Question 14: What are database normalization forms?

Answer: Normalization is the process of organizing data into a related table. it also eliminates redundancy and increases the integrity which improves performance of the query. To normalize a database, we divide the database into tables and establish relationships between the tables.
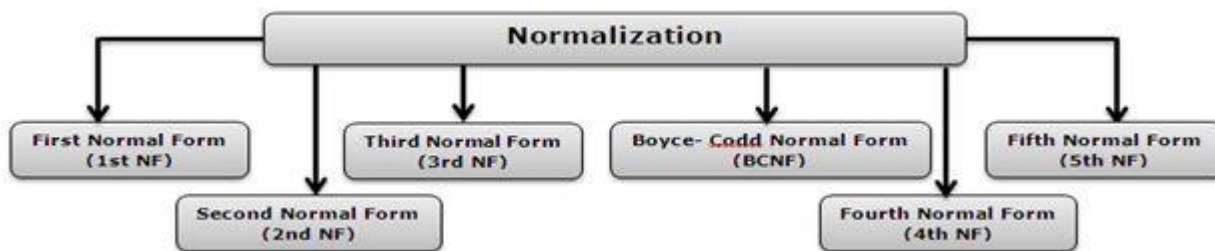
- First Normal Form (1st NF)
- Second Normal Form (2nd NF)
- Third Normal Form (3rd NF)
- Boyce-Codd Normal Form (BCNF)
- Fourth Normal Form (4th NF)
- Fifth Normal Form (5th NF)

Question15: What is a stored procedure?

Answer: A Stored Procedure is a collection or a group of T-SQL statements. Stored Procedures are a precompiled set of one or more statements that are stored together in the database. They reduce the network load because of the precompilation. We can create a Stored Procedure using the "Create proc" statement.

Why we use Stored Procedure

There are several reasons to use a Stored Procedure. They are a network load reducer and decrease execution time because they are precompiled. The most important use of a Stored Procedure is for security purposes. They can restrict SQL Injection. We can avoid SQL injection by use of a Stored Procedure.

How to create a Stored Procedure

```
1.  CREATE PROCEDURE spEmployee
2.  AS
3.  BEGIN
4.
5.  SELECT EmployeeId, Name, Gender, DepartmentName
6.  FROM tblEmployees
7.  INNER JOIN tblDepartments
8.  ON tblEmployees.EmployeeDepartmentId = tblDepartments.DepartmentId
9.  END
```
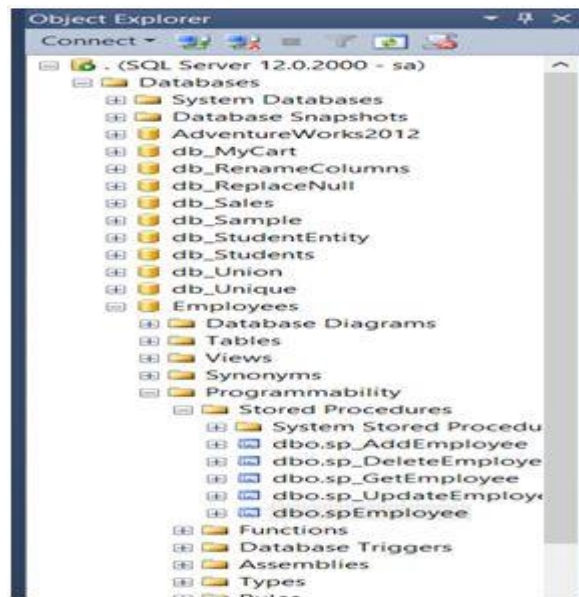
Advantages of using a Stored Procedure in SQL Server

- It is very easy to maintain a Stored Procedure and they are re-usable.
- The Stored Procedure retains the state of the execution plans.
- Stored Procedures can be encrypted and that also prevents SQL Injection Attacks

Question 16: What is a function in SQL Server?

Answer: A function is a sequence of statements that accepts input, processes them to perform a specific task and provides the output. Functions must have a name but the function name can never start with a special character such as @, $, #, and so on.

Types of function

- Pre-Defined Function
- User-Defined Function

User-defined Function:

In a user-defined function we write our logic according to our needs. The main advantage of a user-defined function is that we are not just limited to pre-defined functions. We can write our

own functions for our specific needs or to simplify complex SQL code. The return type of a SQL function is either a scalar value or a table.

Creation of a function

1. Create function ss(@id int)
2. returns table
3. as
4. return select * from item where itemId = @id

Execution of a Function

1. select * from ss(1)

Output:

Question 17: What are the different types of functions in SQL Server?

Answer: A function must return a result. So that is also called a function that returns a result or a value. When we create it a function must specify a value type that will return a value.

- Functions only work with select statements.
- Functions can be used anywhere in SQL, such as AVG, COUNT, SUM, MIN, DATE and so on with select statements.
- Functions compile every time.
- Functions must return a value or result.
- Functions only work with input parameters.
- Try and catch statements are not used in functions.

Function Types:

The following is the function list in SQL Server databases.
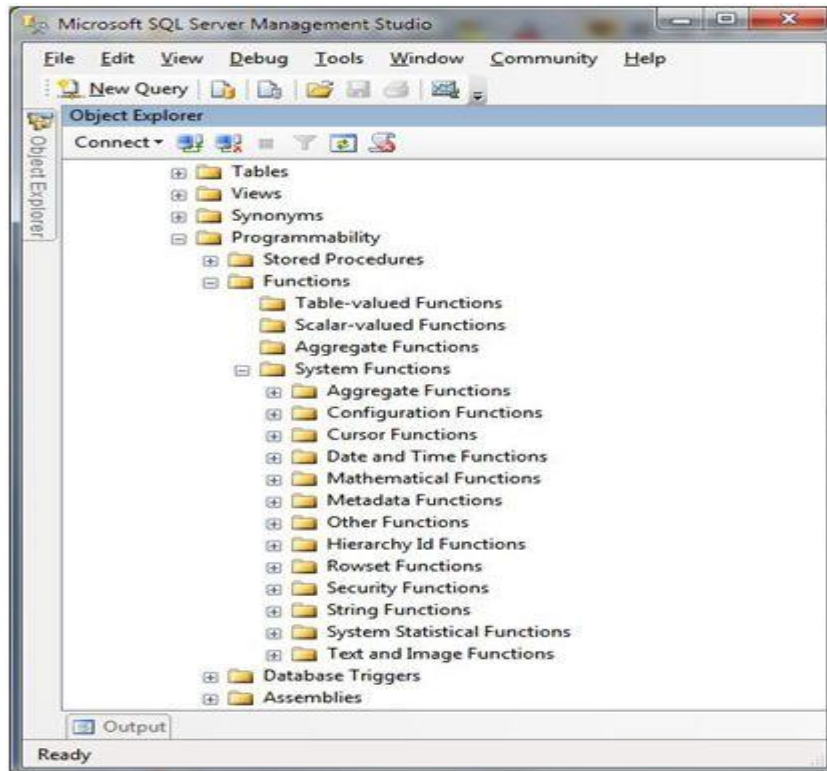
SQL Server contains the following aggregates functions:

Question 18: What is a trigger in SQL Server?

Answer: "A Trigger is a Database object just like a stored procedure or we can say it is a special kind of Stored Procedure which fires when an event occurs in a database.".

It is a database object that is bound to a table and is executed automatically. We cannot explicitly call any trigger. Triggers provide data integrity and used to access and check data before and after modification using DDL or DML query.

Type of Triggers:
There are two types of Triggers:
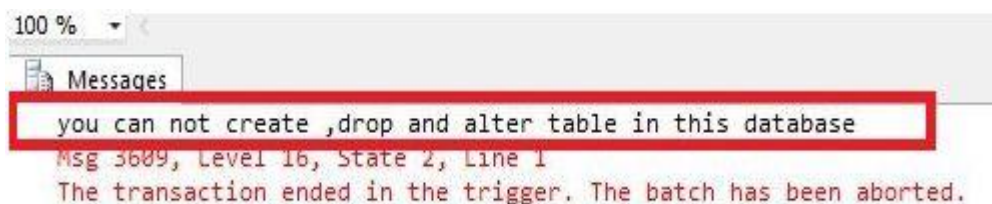
1. DDL Trigger
2. DML trigger

DDL Triggers: They fire in response to DDL (Data Definition Language) command events that start with Create, Alter and Drop like Create_table, Create_view, drop_table, Drop_view and Alter_table.

Code of DDL Triggers:

1. create trigger saftey
2. on database
3. for
4. create_table, alter_table, drop_table
5. as
6. print 'you can not create ,drop and alter table in this database'
7. rollback;

Output:

DML Triggers: They fire in response to DML (Data Manipulation Language) command events that
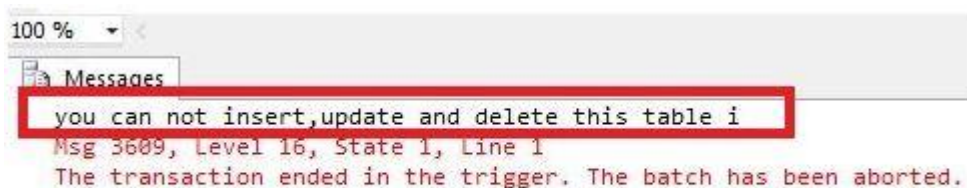
start with Insert, Update and Delete like insert_table, Update_view and Delete_table.

Code of DML Trigger:

1. create trigger deep
2. on emp
3. for
4. insert, update, delete
5. as
6. print 'you can notinsert,update and delete this table i'
7. rollback;

Output:

When we insert, update or delete in a table in a database then the following message appears:

Question 19: Why do we need triggers?

Answer: Why and when to use a trigger:

We use a trigger when we want some event to happen automatically on certain desirable scenarios.

You have a table that changes frequently, now you want to know how many times and when these changes take place. In that case you can create a trigger that will insert the desired data into another table whenever any change in the main table occurs.

In SQL Server we can create the following 3 types of triggers:

- Data Definition Language (DDL) triggers

- Data Manipulation Language (DML) triggers
- Logon triggers

Example:

1. CREATE TRIGGER trgAfterInsert ON[dbo].[Employee_Test]
2. FOR INSERT
3. AS
4. declare@ empid int;
5. declare@ empname varchar(100);
6. declare@ empsal decimal(10, 2);
7. declare@ audit_action varchar(100);
8. select@ empid = i.Emp_ID from inserted i;
9. select@ empname = i.Emp_Name from inserted i;
10. select@ empsal = i.Emp_Sal from inserted i;
11. set@ audit_action = 'Inserted Record -- After Insert Trigger.';
12. insert into Employee_Test_Audit
13. (Emp_ID, Emp_Name, Emp_Sal, Audit_Action, Audit_Timestamp)
14. values(@empid, @empname, @empsal, @audit_action, getdate());
15. PRINT 'AFTER INSERT trigger fired.'
16. GO

Question 20: What are the different types of triggers?

Answer: Triggers are a special type of stored procedure which is executed automatically based on the occurrence of a database event. These events can be categorized as:

- Data Manipulation Language (DML) and
- Data Definition Language (DDL) events.

The benefit derived from triggers is based in their events driven nature. Once created, the trigger automatically fires without user intervention based on an event in the database.

A. **Using DML Triggers:** DML triggers are invoked when any DML command such as INSERT, DELETE, and UPDATE occurs on the data of a table and/or view.

- o DML triggers are powerful objects for maintaining database integrity and consistency.
- o DML triggers evaluate data before it has been committed to the database.
- o During this evaluation the following actions are performed.

We cannot use the following commands in DML trigger:

- o ALTER DATABASE
- o CREATE DATABASE
- o DISK DATABASE
- o LOAD DATABASE
- o RESTORE DATABASE

B. **Using DDL Triggers:**

- o These triggers focus on changes to the definition of database objects as opposed to changes to the actual data.
- o This type of trigger is useful for controlling development and production database environments.

Let us create DDL trigger now-
The following is the syntax.

1. CREATE TRIGGER trigger_name
2. ON
3. {
4.   ALL SERVER | DATABASE
5. }
6. [WITH < ddl_trigger_option > [, ...n]]
7. {
8.   FOR | AFTER
9. }
10. {
11.   event_type | event_group
12.   }[, ...n]
13. AS
14. {
15.   sql_statement[;][...n] | EXTERNAL NAME < method specifier > [;]
16. }

17. CREATE TRIGGER tr_TableAudit
18. ON DATABASE
19. FOR CREATE_TABLE, ALTER_TABLE, DROP_TABLE
20. AS
21. PRINT 'You must disable the TableAudit trigger in order
22. to change any table in this database '
23. ROLLBACK
24. GO

Question 21: What is a view in the database?

Answer: A View is nothing but a select query with a name given to it or we can simply say a view is a Named Query. Ok! Why do we need a view? There can be many answers for this. Some of the important stuff I feel is:

1. A view can combine data from multiple tables using adequate joins and while bringing it may require complex filters and calculated data to form the required result set. From a user's point of view, all these complexities are hidden data queried from a single table.
2. Sometimes for security purposes, access to the table, table structures and table relationships are not given to the database user. All they have is access to a view not knowing what tables actually exist in the database.
3. Using the view, you can restrict the user update only portions of the records.
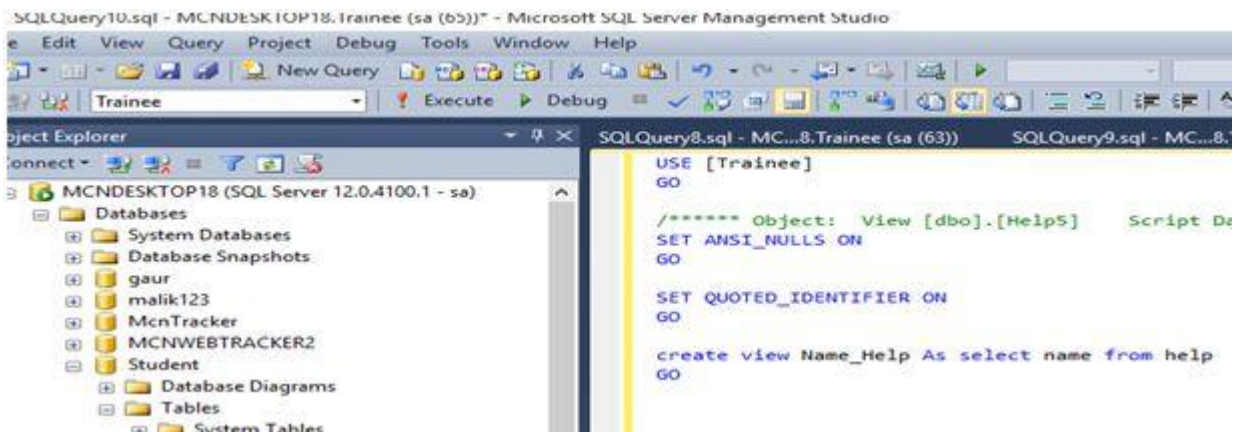
The following are the key points to be noted about views:

1. Multiple views can be created on one table.
2. Views can be defined as read-only or updatable.
3. Views can be indexed for better performance.
4. Insert, update, delete can be done on an updatable view.

Question 22: Why do I need views in a database?

Answer: There are a number of scenarios where we have to look for a view as a solution.
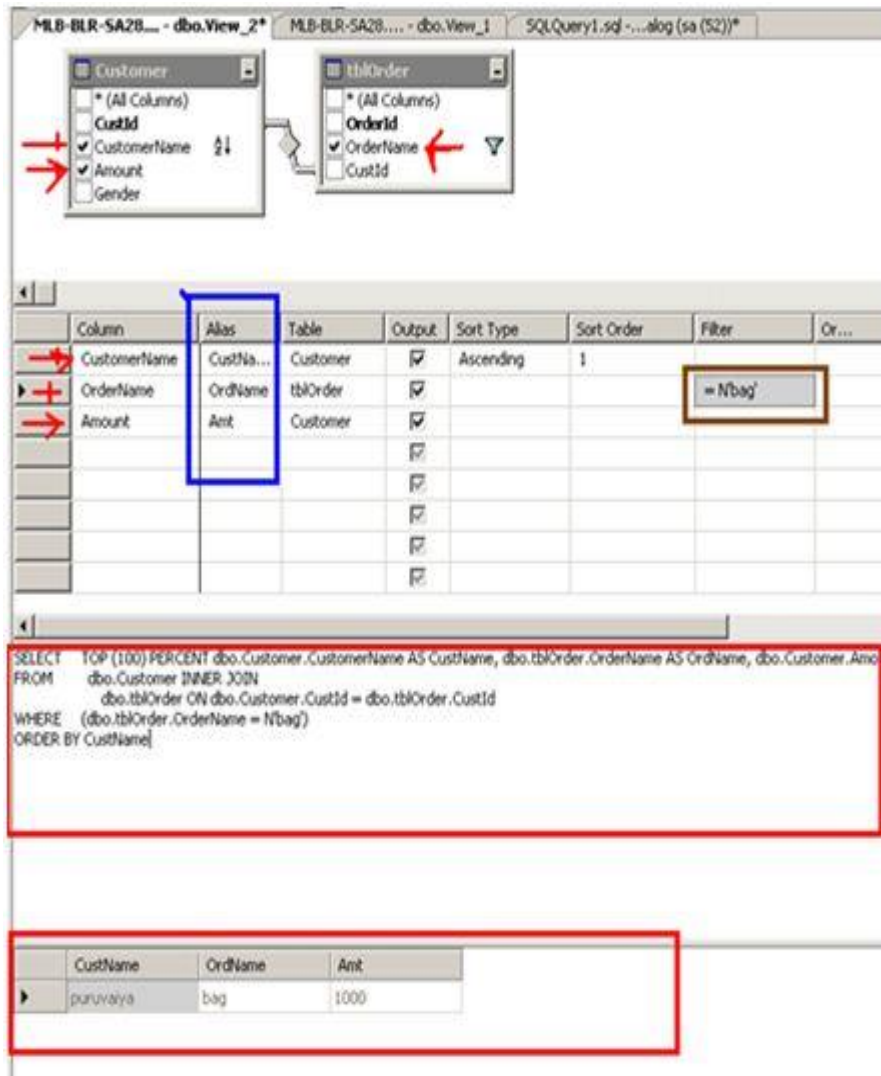
- To hide the complexity of the underlying database schema, or customize the data and schema for a set of users.
- To control access to rows and columns of data.
- To aggregate data for performance.

Views are used for security purposes because they provide encapsulation of the name of the table. Data is in the virtual table, not stored permanently. Views display only selected data.

Syntax of a View:

1. CREATE VIEW view_name AS
2. SELECT column_name(s)
3. FROM table_name
4. WHERE condition

There are two types of views.

- Simple View
- Complex View

Question 23: What is an index?

Answer: An Index is one of the most powerful techniques to work with this enormous information. Database tables are not enough for getting the data efficiently in case of a huge amount of data. In order to get the data quickly we need to index the column in a table.

An index is a database object that is created and maintained by the DBMS. Indexed columns are ordered or sorted so that data searching is extremely fast. An index can be applied on a column or a view. A table can have more than one index.

Types of Index: Microsoft SQL Server has two types of indexes. These are:

Clustered Index: A Clustered Index sorts and stores the data in the table based on keys. A Clustered Index can be defined only once per table in the SQL Server Database, because the data rows can be sorted in only one order. Text, nText and Image data are not allowed as a Clustered index.

1. SET STATISTICS IO ON
2. SELECT * FROM Employee WHERE EmpID = 20001

EmpID	EmpName	Cell	Dept
20001	Black Smith	12345678901	1
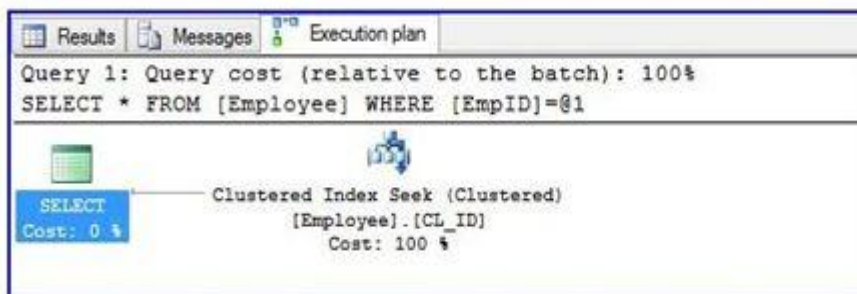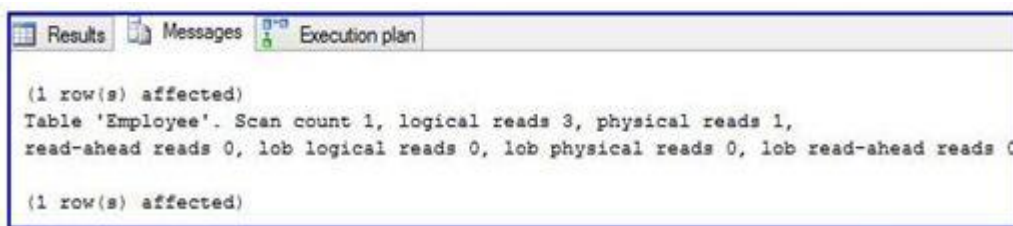
Non-Clustered Index: Non Clustered Indexes or simply indexes are created outside of the table. SQL Server supports 999 Non-Clustered per table and each Non-Clustered can have up to 1023 columns. A Non-Clustered Index does not support the Text, nText and Image data types.

1. CREATE NONCLUSTERED INDEX NCL_ID ON Employee(DeptID)
2. SET STATISTICS IO ON

1. SELECT * FROM Employee WHERE DeptID = 20001

EmpID	EmpName	Cell	Dept
40001	Black Smith	12345678901	20001
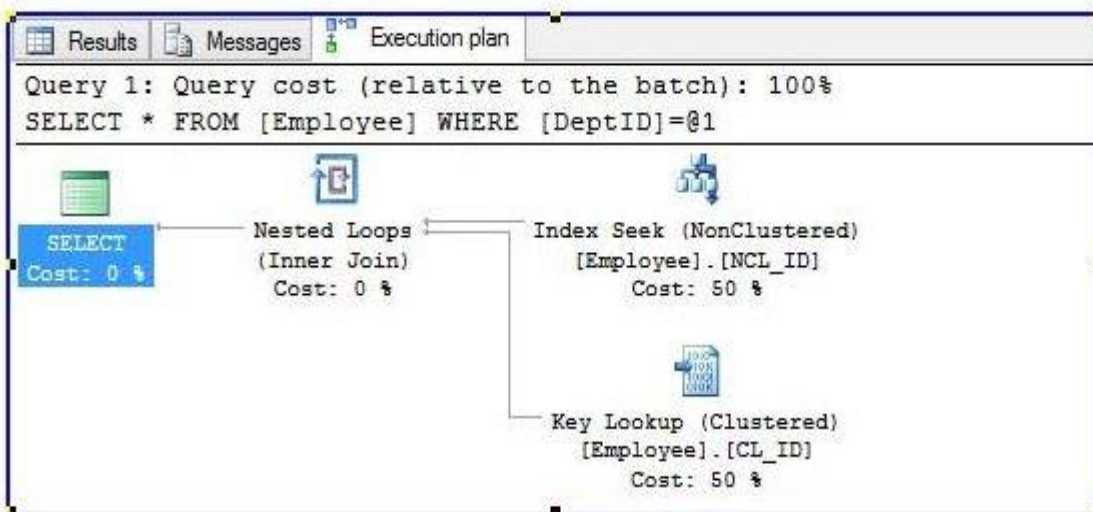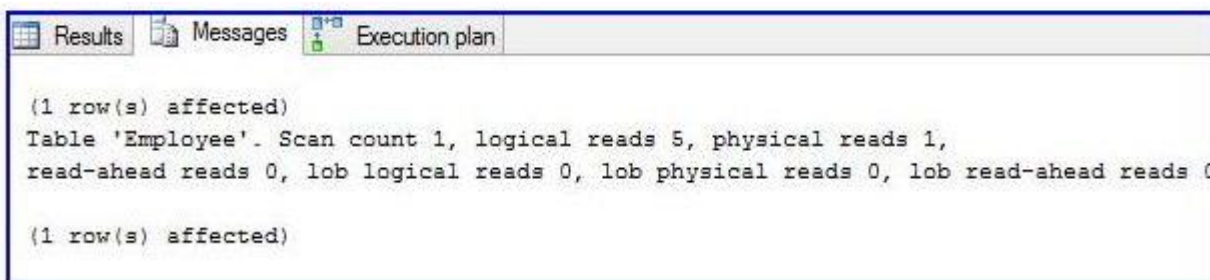
Question 24: Why do I need an index in a database?

Answer: An Index is a database object that can be created on one or more columns (16 max column combinations). When creating the index it will read the column(s) and forms a relevant data structure to minimize the number of data comparisons. The index will improve the performance of data retrieval and add some overhead to the data modification such as create, delete and modify. So it depends on how much data retrieval can be done on table versus how much of DML (Insert, Delete and Update) operations.

Need of Index in Database: An index is basically used for fast data retrieval from the database.

Syntax:

1. CREATE INDEX index_name ON table_name;
2. or
3. DROP INDEX index_name;

Type of Index:

In a SQL Server database there are mainly the following two types of indexes:

1. Clustered index and
2. Non Clustered index

Question 25: What is a query in a database?

Answer: SQL is a complete data manipulation language that is used not only for database queries, but also to modify and update data in the database. Compared to the complexity of the SELECT statement, which supports SQL queries, the SQL statements that modify and create database contents are somewhat simple.

However, database updates pose some challenges for a DBMS beyond those presented by database queries. The DBMS must protect the integrity of the stored data during changes, ensuring that only valid data is introduced into the database. The DBMS must also coordinate simultaneous updates by multiple users, ensuring that the users and their changes do not interfere with one another.

INSERT statement adds new rows of data to a table.

Syntax:

1. Insert into table_Name(column names) values(Values for column).
2. INSERT INTO employee(ID, SURNAME, FIRSTNAME, EMAIL, COUNTRY, PHONE)

3. VALUES(111, 'vithal', 'wadje', 'vithal.wadje@yahoo.com', 'India', ' +914545455454')

Question 26: What are query types in a database?

Answer: Types of Commands in SQL ServerThese commands are categorized into:

- DDL
- DCL
- DML
- TCL

Let's see these categories one-by-one.

DDL - Data Definition Language (DDL) commands are the category responsible for dealing with the structure of objects. I mean that with these commands we can modify our object/entity structure. For example if there's one table and you want to modify the structure of that table, you can use DDL commands.

The following are the commands in this category:

Command	Description
Create	Used to create objects.
Alter	Used to modify created object.
Drop	Used to delete object.

Using these commands you can create any objects like tables, views, databases, triggers, and so on.

For example:

1. CREATE DATABASE DB2
2. GO
3. CREATE TABLE tblDemo
4. (
5. Id int primary key,
6. Name char(20)
7.)
8. GO
9. DROP DATABASE DB2

DML - Data Manipulation Language (DML) commands manipulate data stored in objects like tables, views and so on. With the help these commands you can easily modify, insert and delete your data.

The following are the commands in this category:

Command	Description
Insert	Insert data into table.
Delete	Delete data from table.
Update	Update data into table.
Insert Into	Insert bulk data into table.

Using these commands you can manipulate any kind of data stored in entities.

For example:

1. INSERT INTO tblDemo VALUES(1, 'Abhishek')
2. GO
3. DELETE FROM tblDemo WHERE Id = 4
4. GO
5. UPDATE tblDemo
6. SET Name = 'Sunny'
7. WHERE Id = 6
8. GO

DCL - Data Control Language (DCL) commands are for security purposes. These commands are used to provide roles, permissions, access and so on.

The following are the commands in this category:

Command	Description
Grant	Provide user access to Database or any other object.
Revoke	Take back the access from user.

For example: we have the following data:

Database: CSharpCornerDB
Table:
User: CSharpCorner

currently we didn't provide any permission to this user.

Now we'll create a table in the CSharpCornerDB database:

1. CREATE table tblArticles
2. (
3. ArticleId int primary key identity,
4. ArticleName varchar(10),
5. Category varchar(10)
6.)

If we execute this command, we'll get an error message.

Msg 262, Level 14, State 1, Line 1

CREATE TABLE permission denied in database 'CSharpCornerDB'.

This is because this user doesn't have permission to create anything right now. We'll learn how to grant or revoke permission on objects in our next article.

TCL - Transaction Control Language (TCL) commands are for managing transactions in SQL Server. The following are the commands in this category:

Command	Description
Commit	Used to save any transaction permanently.
Rollback	This command Is used to restore database to its last committed state.
Save Tran	This command is used to save the transaction so that we can rollback that transaction to the point whenever necessary.

For example, we have a table named "tblStudent" with 3 records as shown below:

Now we'll begin our transaction and add another record and commit that transaction.

1. Begin Tran
2. Insert INTO tblStudents VALUES('Sumit')
3. COMMIT

Now we have 4 Records.

Now, we'll add three records, one by one with save point, but we don't commit our transaction.

1. Begin Tran
2. Insert INTO tblStudents VALUES('Kajal')
3. SAVE Tran A;
4. Insert INTO tblStudents VALUES('Rahul')
5. SAVE Tran B;
6. Insert INTO tblStudents VALUES('Ram')
7. SAVE Tran C;
8.
9. SELECT * from tblStudents

Now we have the following records in the table, from which the last three records are not yet committed.

Now we have 3 savepoints, in other words A, B and C. Since our transaction is not yet committed, we can roll it back to any savepoint. We'll roll it back to point B, in other words at "Rahul".

1. ROLLBACK TRAN B
2. COMMIT now when you fire the select query, you'll get records up to ID 6.

In this, we have seen the types of commands in SQL Server and done some overview of that. We have also seen how to commit transactions and how to roll back any transaction to any saved point.

Question 27: What is a join in SQL Server?

Answer: If you want to retrieve data from multiple tables then you need to use joins in SQL Server. Joins are used to get data from two or more tables based on the relationships among some of the columns in the tables.

Syntax:

The Inner join syntax is as follows:

1. SELECT < column list >
2. FROM < left joined table > [INNER] JOIN < right joined table >
3. ON < join condition >

The example is developed in SQL Server 2012 using the SQL Server Management Studio.

Creating Table in SQL Server:

Now to create 3 tables in the Master database named Table1, Table2 and Table3.

Table1-

1. CREATE TABLE Table1
2. (
3. ID INT, Name VARCHAR(20)
4.)

Table2-

1. CREATE TABLE Table2
2. (
3. ID INT, Name VARCHAR(30)
4.)

Table3-

1. CREATE TABLE Table3
2. (
3. ID INT, Name VARCHAR(40)
4.)

Question 28: What are different types of joins in SQL Server?

Answer: Joins are useful for bringing data together from different tables based on their database relations. First we will see how the join operates between tables. Then we will explore the Order of Execution when both a join and a where condition exist. Finally we will move our exploration to the importance of the Join order.
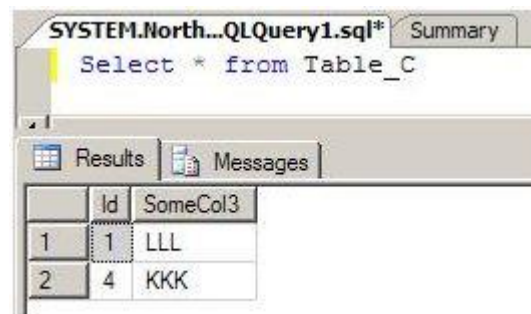
A Join condition defines a way two tables are related in a query by:

- Specifying the column to be used for the Join from each table. In joining foreign keys in a table and its associated key in the other table.
- To use the logical operator in comparing values from the columns.

There are three type of joins available based on the way we join columns of two different tables.

1. Full Join
2. Inner Join
3. Left outer Join
4. Right outer Join

Full Join - A full join is somewhat different from the Cartesian product. A Cartesian product will get all the possible row combinations of the two joining tables. A Full Join takes the matching columns plus all table rows from the left table that does not match the right and all table rows in the right that does not match the left. It applies null for unmatched rows on the other end when doing so. The following example shows the full join between Table_A and Table_C

Question 29: What is an inner join in SQL?

Answer: Inner or Self Join - This Join returns a row when there is at least one match in both tables.

Let's see an example:
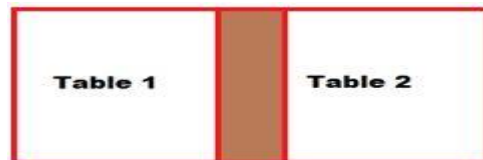
1. Select * From Table1
2. Inner Join Table2
3. on table1.ID = table2.ID

The following query displays the Employee Name and the corresponding Manager Name within the employee table.

1. SELECT e1.Employee_Name EmployeeName, e2.Employee_Name ManagerName
2. FROM employee e1(nolock), employee e2(nolock)
3. WHERE e1.EmployeeID = e2.ManagerID

Output:

An inner join (sometimes called a "simple join") is a join of two or more tables that returns only those rows that satisfy the join condition.

Question 30: What is an outer join in SQL?

Answer: There are three different types of outer joins; let's see 1 by 1.

- Left Outer Join
- Right Outer Join
- Full Outer Join

Left Outer Join - A LEFT OUTER JOIN is one of the JOIN operations that allows you to specify a join clause. It preserves the unmatched rows from the first (left) table, joining them with a NULL row in the shape of the second (right) table.
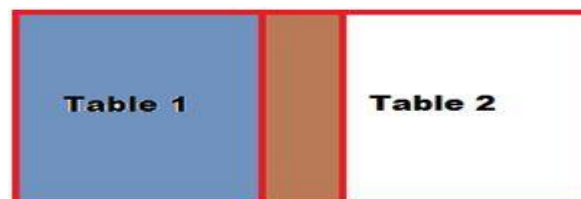
1. Select * From Table1
2. Left Outer Join
3. on table1.ID = table2.ID

Right Outer Join - A RIGHT OUTER JOIN is one of the JOIN operations that allows you to specify a JOIN clause. It preserves the unmatched rows from the Table2 (right) table, joining them with a NULL in the shape of the Table1 (left) table. A LEFT OUTER JOIN B is equivalent to B RIGHT OUTER JOIN A, with the columns in a different order.

1. Select * From Table1
2. Right Outer Join
3. on table1.ID = table2.ID

|Question 31: What is full join in SQL?

Answer: A Full Outer Join fetches all records of both tables; where the record does not match, it returns Null.

> select e.empId, e.empName, e1.empAdd from emp e full outer join emp_add e1 on e.empI
> d = e1.empId

Output:

Or

Full Outer Join - FULL OUTER JOIN: This JOIN is a combination of both. All records from both Left_Table and Right_Table are in the result set and matched when they can be on the Join_Condition; when no record is found in the opposite table, NULL values are used for the columns.

1. Select * From Table1
2. Full Outer Join
3. on table1.ID = table2.ID

Question 32: What is left join in SQL Server?

Answer: Left Join: A LEFT OUTER JOIN is one of the JOIN operations that allow you to specify a join clause.

It preserves the unmatched rows from the first (left) table, joining them with a NULL row in the shape of the second (right) table.
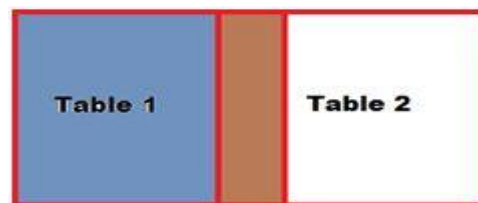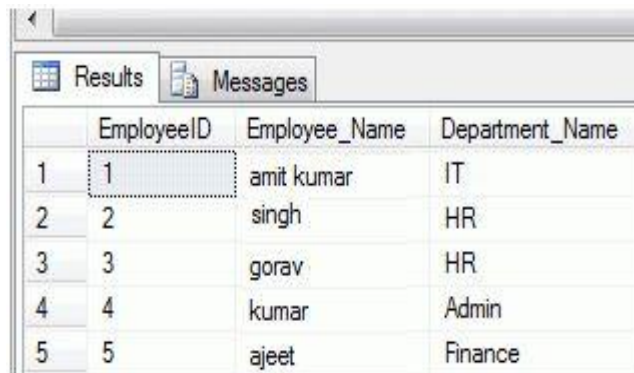
1. Select * From Table1
2. Left Outer Join
3. on table1.ID=table2.ID

A left outer join displays all the rows from the first table and the matched rows from the second table.

Example: The following query retrieves the employee name and the corresponding department he belongs to, whereas all the departments are displayed even if the employee is not assigned to any department.

1. SELECT e.EmployeeID, e.Employee_Name, d.Department_Name
2. FROM employee e(nolock) LEFT JOIN department d(nolock)
3. ON e.DepartmentID = d.DepartmentID

Output:

Question 33: What is a right join in SQL Server?

Answer: Right JOIN - A RIGHT OUTER JOIN is one of the JOIN operations that allows you to specify a JOIN clause. It preserves the unmatched rows from the Table2 (right) table, joining them with a NULL in the shape of the Table1 (left) table. A LEFT OUTER JOIN B is equivalent to B RIGHT OUTER JOIN A, with the columns in a different order.
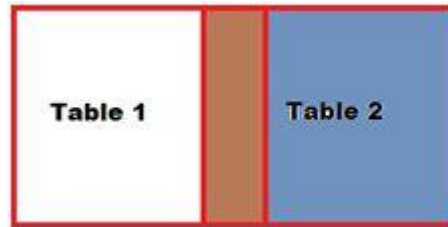
1. Select * From Table1
2. Right Outer Join
3. on table1.ID = table2.ID

The right outer join displays all the rows from the second table and matched rows from the first table.

Example:

1. SELECT e.EmployeeID, e.Employee_Name, d.Department_Name
2. FROM employee e(nolock) RIGHT JOIN department d(nolock)
3. ON e.DepartmentID = d.DepartmentID

Output:

Question 34: What is database engine in SQL Server?

Answer: The SQL Server Database Engine, SQL Server Agent, and several other SQL Server components run as services. These services typically are started when the operating system starts. This depends on what is specified during setup; some services are not started by default.

A service is a type of application (executable) that runs in the system background. Services usually provide core operating system features, such as Web serving, event logging, or file serving. Services can run without showing a user interface on the computer desktop. The SQL Server Database Engine, SQL Server Agent, and several other SQL Server components run as services. These services typically are started when the operating system starts. This depends on what is specified during setup; some services are not started by default.

This describes the management of the various SQL Server services on your machine. Before you log in to an instance of SQL Server, you need to know how to start, stop, pause, resume, and restart an instance of SQL Server. After you are logged in, you can perform tasks such as administering the server or querying a database.

Let's start now, select start/All Programs/Microsoft SQL Server2005/Configuration Tools/SQL Server Configuration Manager.
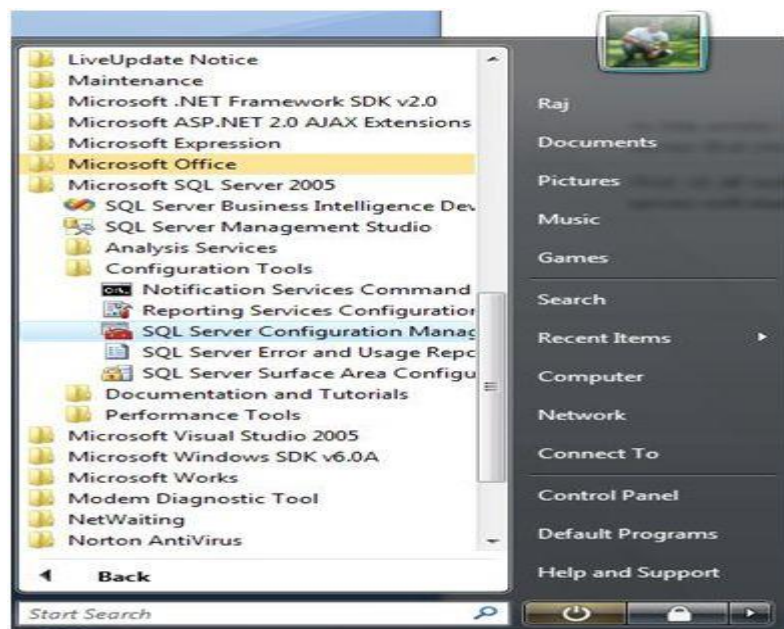
Question 35: What are the Analysis Services in SQL Server?

Answer: The purpose of analysis services is to turn data into information and to provide quick and easy access to that information for decision makers. SSAS provides OLAP by letting you design, create and manage multidimensional structures that contain data aggregated from other data sources, such as relational databases and provides many data mining algorithms for mining data from data sources. So for delivering OLAP and data mining it uses client and server technologies.

The main idea of SSAS is to provide fast results from data sources when we apply a query because in order to make a decision we need data of various dimensions.

Components of the Architecture in detail:

1. **Server Architecture:** This runs as a Windows service. The Msmdsrv.exe application is a server component. This application consists of security, XMLA listener, query processor and other components that perform the following tasks:

1	Parsing statements received from clients
2	Managing metadata
3	Handling transactions
4	Processing calculations
5	Storing dimension and cell data
6	Creating aggregations
7	Scheduling queries
8	Caching objects
9	Managing server resources

2. **Client Architecture:** SSAS has a thin client Component Architecture. All queries and calculations are resolved by the server only. So for each request a server to client connection is required. There are several providers with SSAS to support various programming languages. These providers communicate using SOAP packets. You can better understand this by the following diagram:
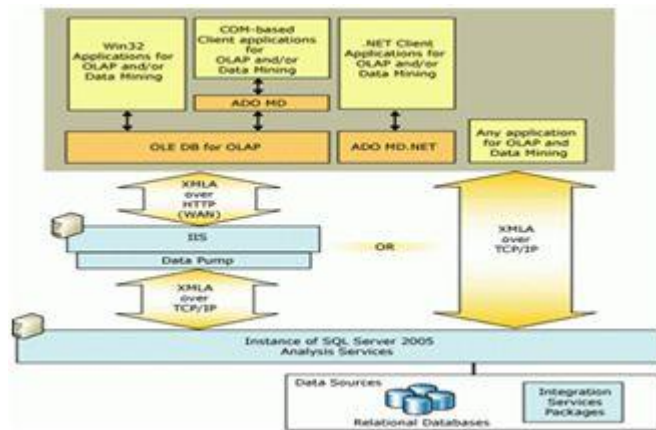
Question 36: What are the integration services in SQL Server?

Answer: Integration Services is a platform for building high performance data integration and workflow solutions, including extraction, transformation and loading (ETL) operations for data warehousing.
This includes graphical tools and wizards for building and debugging packages.

Uses of Integration Services:

One use of Integration Services is to merge data from multiple data stores and update the data to data warehouses and/or data marts. Create the Data Transformation process logic and automate the data loading process.

Architecture of Integration Services:
Some important components to using Integration Services:

- SSIS Designer
- Runtime engine
- Tasks and other executables
- Data Flow engine and Data Flow components
- API or object model
- Integration Services Service
- SQL Server Import and Export Wizard
- Other tools, wizards and command prompt utilities

Question 37: What are the data quality services in SQL Server?

Answer: SQL Server Data Quality Services - SQL Server 2012 Data Quality Services (DQS) is the data quality product from Microsoft SQL Server 2012. DQS enables you to perform a variety of critical data quality tasks, including correction, enrichment, standardization and de-duplication of your data.

DQS provides the following features to resolve data quality issues:

- Data Cleansing
- Matching
- Reference Data Services
- Profiling and Monitoring
- Knowledge Base
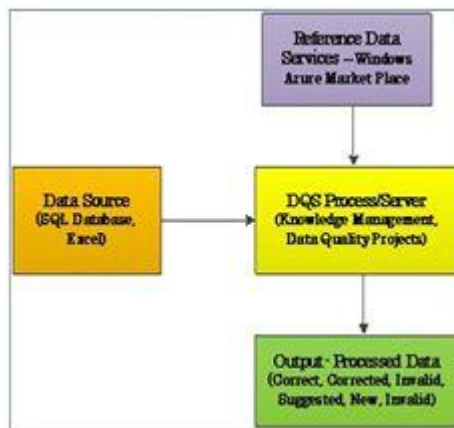
DQS enables you to perform data cleansing using cloud-based reference data services provided by reference data providers. DQS also provides profiling that is integrated into its data-quality tasks, enabling to analyze the integrity of the data.

Data Quality Server - Data Quality Server is implemented as three SQL Server catalogs DQS_MAIN, DQS_PROJECTS, and DQS_STAGING_DATA.

DQS_MAIN includes DQS Stored Procedures, DQS engine, and published Knowledge Bases.

DQS_PROJECTS includes data that is required for Knowledge Base management and DQS project activities.

DQS_STAGING_DATA provides an intermediate staging database where you can copy your source data to perform DQS operations, and then export your processed data.
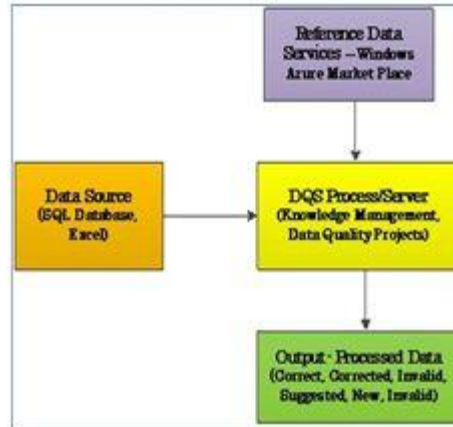
Question 38: What are the reporting services in SQL Server?

Answer: SQL Server Reporting Services is a comprehensive reporting platform that includes processing components. Processing components are the basis for the multilayered architecture of SQL Server Reporting Services. Processing components interact with each other to retrieve data and deliver a report.

SQL Server Reporting Services has the following two basic components:

- Processors
- Extensions

Tools and Components of SQL Server Reporting Services architecture: This architecture consists mainly of the following types of components and tools:

- Report Builder
- Report Designer
- Report Manager
- Report Server
- Report server database
- Data sources

Question 39: What are the master data services in SQL Server?

Answer: The goal of MDS is to address the challenges of both operational and analytical master data management by providing a master data hub to centrally organize, maintain, and manage your master data. This master data hub supports these capabilities with a scalable and extensible infrastructure built on SQL Server and the Windows Communication Foundation (WCF) APIs.

Master Data Services Components: The wizard installs Master Data Services Configuration Manager, installs the files necessary to run the Master Data Services Web service, and registers assemblies. After installation, you use the Master Data Services Configuration Manager to create and configure a Master Data Services database in a SQL Server instance that you specify, create the Master Data Services Web application, and enable the Web service.

Data Stewardship: Master Data Manager is the data stewardship portal in which authorized business users can perform all activities related to master data management. At minimum, a user can use this Web application to review the data in a master data model. Users with higher permissions can make changes to the master data and its structure, define business rules, review changes to master data, and reverse changes.

Model Objects: Most activities in MDS revolve around models and the objects they contain. A model is a container for all objects that define the structure of the master data. A model contains at least one entity, which is analogous to a table in a relational database. An entity contains members, which are like the rows in a table, as shown in Figure 7-1. Members (also known as leaf members) are the master data that you are managing in MDS. Each leaf member of the entity has multiple attributes, which correspond to table columns in the analogy.

Name	Code	ProductSubCategory	ProductLine	Country
Adjustable Race	AR-5381	38	NA	US
Bearing Ball	BA-8327	38	NA	US
LL Bottom Bracket	BB-7421	5	NA	US
ML Bottom Bracket	BB-8107	5	NA	US
HL Bottom Bracket	BB-9108	5	NA	US

Master Data Maintenance: Master Data Manager is more than a place to define model objects. It also allows you to create, edit, and update leaf members and consolidated members. When you add a leaf member, you initially provide values for only the Name and Code attributes, as shown in Figure 7-4. You can also use a search button to locate and select the parent consolidated

member in each hierarchy.

Question 40: What is replication in SQL Server?

Answer: Replication is a process or method to synchronize the data across multiple servers. Replication is done by a replica set. A replication maintains the same data set. Replica sets provide redundancy and high availability with multiple copies of data on different database servers.

Replication removes dependencies from a single server so replication protects a database from the loss of a single server. Replication provides a mechanism to recover from hardware failure and service interruptions.

Replication is also used to increase the read capacity. Replication provides choices for the client to select a different server for read and write operations. Replication maintains copies in different data centers to increase the locality and availability of data for distributed applications.

Example: Snapshot Replication

Step 1: Open the replication node in your database and choose the option Local Publications.

Step 2: Right-click on Local Publications and click on New publication.
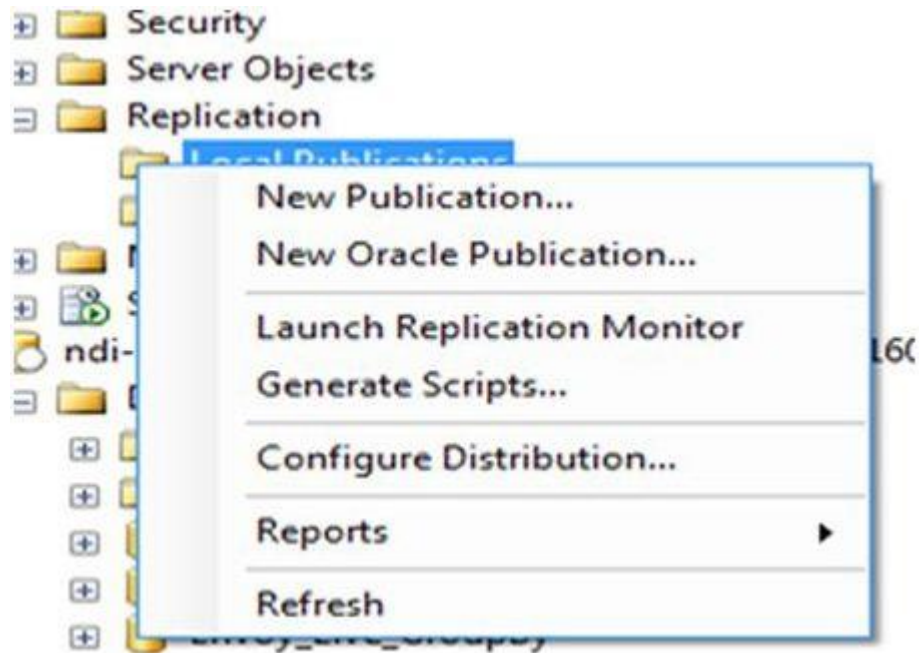
Step 3: After clicking on the new publication tab the following window will appear and click on the "Next" button.
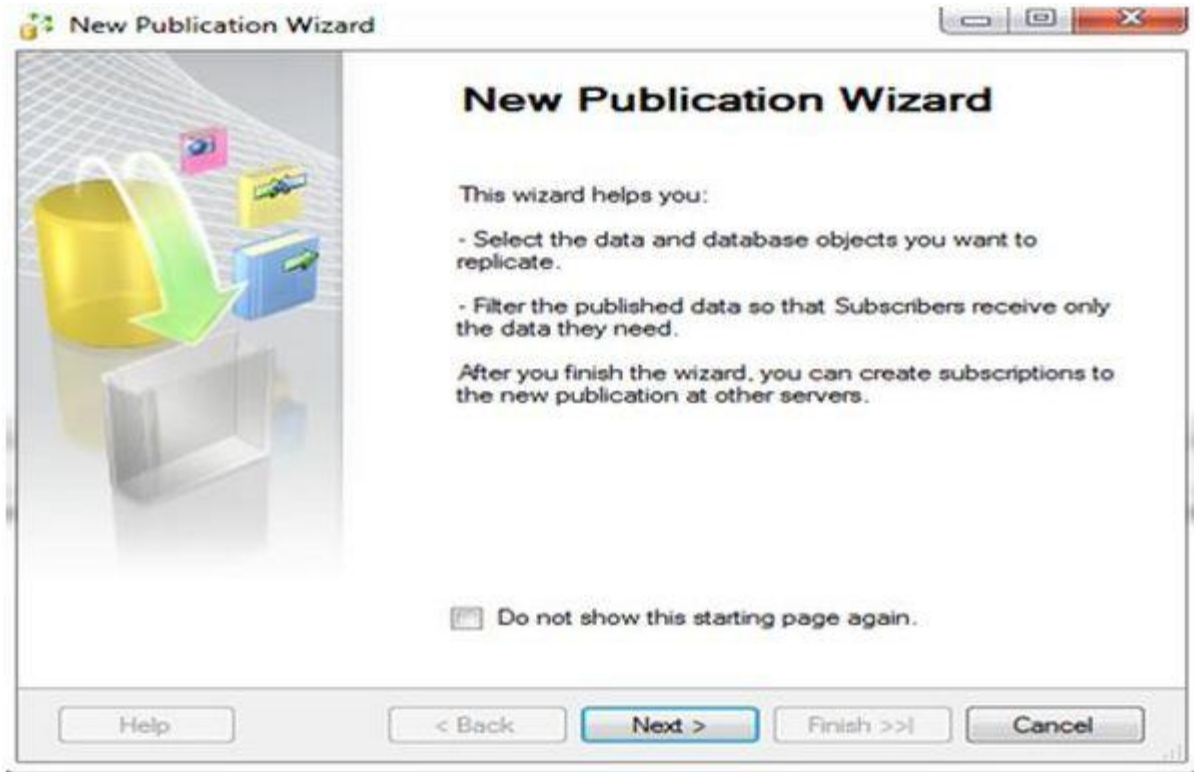
Question 41: How to I select data from an SQL Server table?

Answer: How to select specific rows or all columns, selecting distinct rows, filtering with where clause, sorting rows using orderby and so on.

We will be using the AdventureWorks2012 database for this demo.

1. To select all the rows and columns from a table, we use the following query:

SELECT * FROM HumanResources.Employee

Execute the query by pressing F5 or via the execute button.

Output:

There is another way to select all the columns from a table. Instead of using * we can specify the column names.

1. SELECT BusinessEntityID, NationalIDNumber, LoginID, OrganizationNode, Organizatio
    nLevel, JobTitle, BirthDate, MaritalStatus, Gender, HireDate, SalariedFlag, VacationHour
    s, SickLeaveHours, CurrentFlag, rowguid, ModifiedDate FROM HumanResources.Emplo
    yee

The output will be the same.

If you feel lazy in writing this long query given above then what you can do is go to the Object Explorer window, then expand adventureWorks2012 then select HumanResourcesEmployee table and right-click on it. After that "select script table as" then select "To", then you will see a New query editor window.
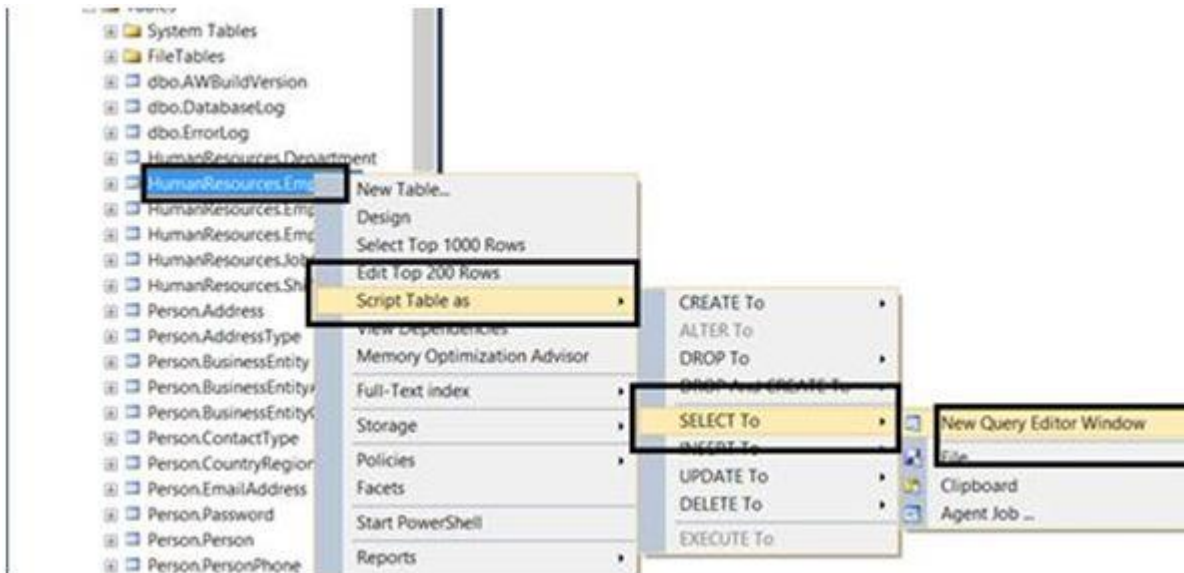
SQL Server will generate the SELECT query for us.

Question 42: What is a check in SQL?

Answer: A Check Constraint is a rule that identifies valid values for columns of data. A Check Constraint helps to enforce Domain Integrity. If the condition in a Check Constraint is not satisfied then it prevents the value from entering into the database.

Syntax:

Create table tableName(Column1 dataType Check(expression), Column2, columnN)

Example:

create table emp(empId int check(empId >10),empName varchar(15))

Output:
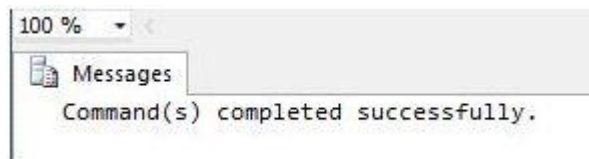
insert into emp values(8,'d')

Output:
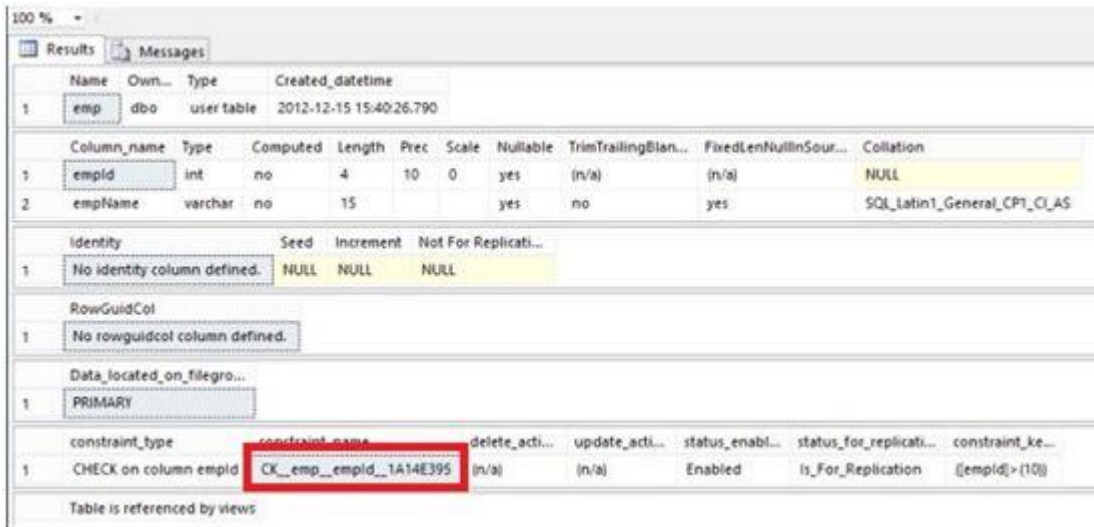
Dropping the Check Constraint:

Firstly, we can determine the name of the constraint using the following command:

1. exec sp_help emp

Output:

Question 43: What is a default in SQL?

Answer: Constraints are rules that decide what kind of data can enter into the database tables. SQL server has six types of constraints and we will explore all these constraints here with suitable examples. The constraints that we are going to explore are listed below:

1. Primary Key Constraint
2. Foreign Key Constraint
3. Not Null Constraint
4. Unique constraint
5. Default Constraint
6. Check Constraint

Default Constraint:

Default constraint allows you set a default value for the column. That means when a row is created for the first time, and there is no entry specified for the column that has a default constraint on it, then the default value is stored in the column.

Note that this not a Not Null constraint and do not confuse the default value constraint with disallowing the Null entries. Default value for the column is set only when the row is created for the first time and column value is ignored on the Insert. Modification to the column with NULL value or even the Insert operation specifying the Null value for the column is allowed.

Let us set the Default value of 1 for the Class. Here are the steps:

- Bring up the table designer
- Select the Class Row as you already did.
- At the bottom of the layout, you will see Column properties.

## Question 44: How to create a database using SQL?

Answer: A database is described as an organized way of collection of DATA. It is the collection of schemes, tables, queries, reports, views and other objects.

Syntax: *CREATEDATABASEDatabaseName*

Example: *CREATEDATABASE Student*
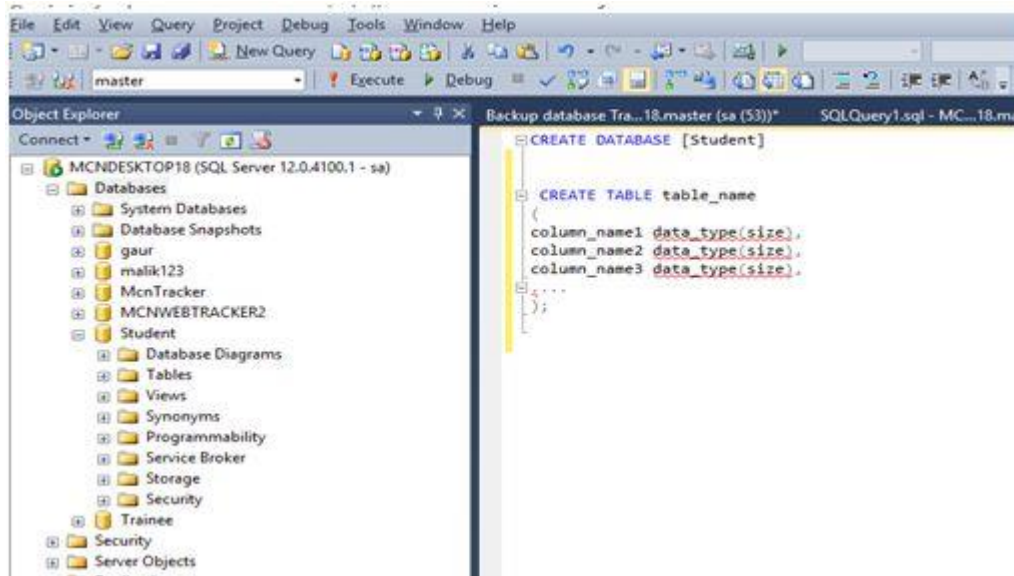
Or you can create Database through Design/ Wizard form by right clicking on DATABASE option - New Database
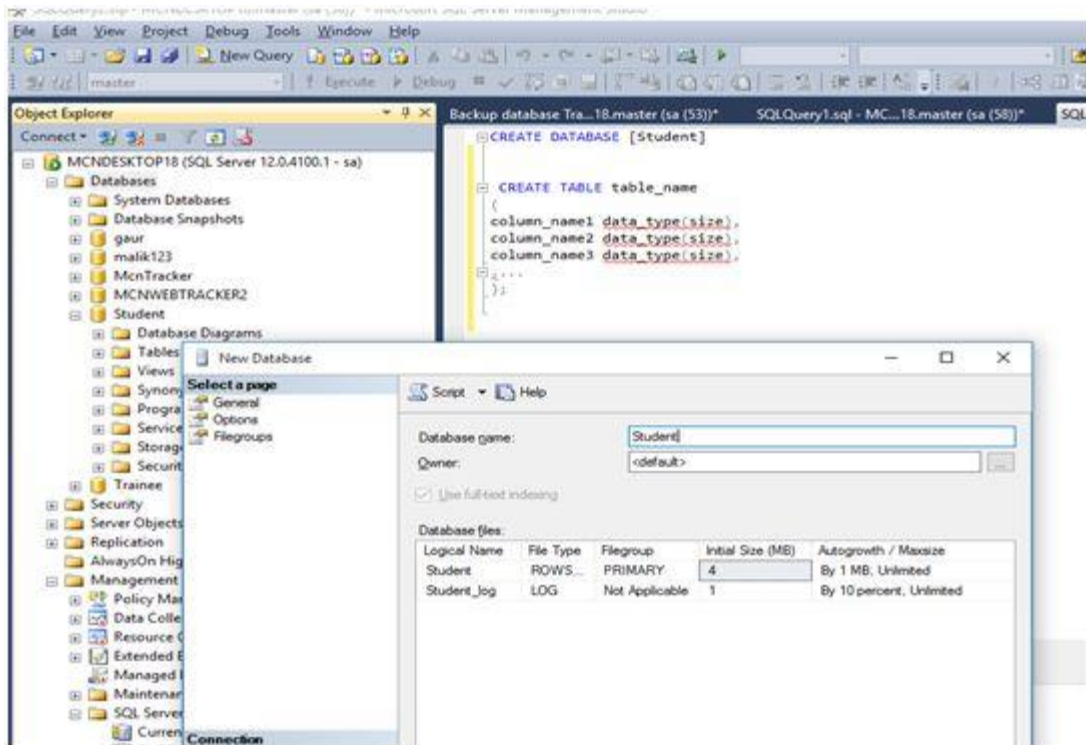
Question 45: What is a constraint in SQL?

Answer: Constraints are the rules that decide what kind of data can enter into the database tables. SQL server has six types of constraints and we will explore all these constraints here with suitable examples. The constraints that we are going to explore are listed below:

1. Primary Key Constraint
2. Foreign Key Constraint
3. Not Null Constraint
4. Unique constraint
5. Default Constraint
6. Check Constraint

To explain these constraints we need two tables. Firstly, let us create these tables. Run the scripts shown below to create the tables. Copy and paste the code into the new Query Editor window, then execute it.

1. CREATE TABLE Student(StudId smallint, StudName varchar(50), Class tinyint);
2. CREATE TABLE TotalMarks(StudentId smallint, TotalMarks smallint);
3. Go

Note that there are no constraints at present on these tables. We will add the constraints one by one.

Primary Key Constraint - A table column with this constraint is called the key column for the table. This constraint helps the table to make sure that the value is not repeated and also that there are no null entries. We will mark the StudId column of the Student table as the primary key. Follow these steps:

- Right click the student table and click on the modify button
- From the displayed layout select the StudId row by clicking the Small Square like button on the left side of the row.
- Click on the Set Primary Key toolbar button to set the StudId column as primary key column.
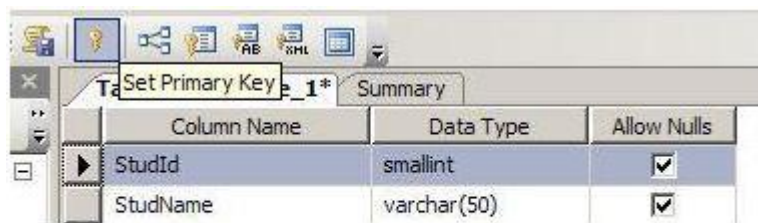
Now this column does not allow null values and duplicate values. You can try inserting values to violate these conditions and see what happens. A table can have only one Primary key. Multiple columns can participate on the primary key column. Then the uniqueness is considered among all the participant columns by combining their values.

Not Null Constraint - This constraint is useful to stop storing the null entries in the specified columns. We will mark student name column as not null column. This allows us to always having some entries in the student name column of the student table without having NULL. Follow the steps below:

- As you did previously, bring up the table design view by clicking the modify context menu for the table.
- Remove the check mark as shown in the picture below. This action will enable the Not Null constraint for the StudName column.

Default Constraint - Default constraint allows you set a default value for the column. That means when a row is created for the first time, and there is no entry specified for the column that has a default constraint on it, then the default value is stored in the column. Note that this not a Not Null constraint and do not confuse the default value constraint with disallowing the Null entries. Default value for the column is set only when the row is created for the first time and column value is ignored on the Insert. Modification to the column with NULL value or even the Insert operation specifying the Null value for the column is allowed.

Let us set the Default value of 1 for the Class. Here are the steps:

- Bring up the table designer
- Select the Class Row as you already did.
- At the bottom of the layout, you will see a Column properties as shown in the below picture. Set the default as shown below:

Question 46: How do I define constraints in SQL?

Answer: Constraints are rules and restrictions applied on a column or a table such that unwanted data can't be inserted into tables. This ensures the accuracy and reliability of the data in the database. We can create constraints on single or multiple columns of any table. Constraints maintain the data integrity and accuracy in the table.

Constraints can be classified into the following two types:

Column Types Constraints - Definitions of these types of constraints is given when the table is created.

1. Create Table My_Constraint
2. (
3. IID int NOT NULL,
4. Salary int CHECK(Salary > 5000)
5.)

Table Types Constraints - Definitions of these types of constraints is given after the creation of the table using the Alter Command.

1. Alter Table My_Cosntraint

2. Add constraint Check_Constraint Check(Age>50)

SQL Server contains the following six types of constraints:

- Not Null Constraint
- Check Constraint
- Default Constraint
- Unique Constraint
- Primary Constraint
- Foreign Constraint

Not Null Constraint - A Not Null constraint restrict the insertion of null values into a column. If we are using a Not Null Constraint for a column then we cannot ignore the value of this column during an insertion of data into the table.

Column Level-

Syntax:

1. CREATE TABLE Table_Name
2. (
3. Column_Name Datatype CONSTRAINT Constraint_Name NOT NULL,
4.);

Example:

1. Create Table My_Constraint
2. (
3. IID int NOT NULL,
4. Name nvarchar(50) CONSTRAINT Cons_NotNull not null,
5. Age int Not Null,
6.)

Table Level-

Syntax:

1. ALTER TABLE Table_Name
2. ALTER COLUMN Column_Name Datatype NOT NULL

Example:

1. Alter Table My_Constraint
2. Alter Column IId int Not Null

Without SQL Command - We can also create a Not Null constraint in Microsoft SQL Server without execution of a SQL query.
First right-click on the table and select and click on the design option. Now check all the columns in the "Allow Nulls" option that should have a Null Value.

Check Constraint - A Check constraint checks for a specific condition before inserting data into a table. If the data passes all the Check constraints then the data will be inserted into the table otherwise the data for insertion will be discarded. The CHECK constraint ensures that all values in a column satisfies certain conditions.

Question 47: What is the meaning of Not Null in SQL?

Answer: Constraints are rules that decide what kind of data can enter into the database tables. SQL server has six types of constraints and we will explore all these constraints here with suitable examples. The constraints that we are going to explore are listed below:

- Primary Key Constraint
- Foreign Key Constraint
- Not Null Constraint
- Unique constraint
- Default Constraint
- Check Constraint

This constraint is useful to stop storing the null entries in the specified columns. We will mark student name column as not null column. This allows us to always have some entries in the student name column of the student table without having NULL. Here are the steps:

1. As you did previously, bring up the table design view by clicking the modify context menu for the table.

2. Remove the check mark as shown in the picture below. This action will enable the Not Null constraint for the StudName column.

Example:

Column Name	Data Type	Allow Nulls
StudId	smallint	☐
StudName	varchar(50)	☑
Class	tinyint	☑
		☐

Question 48: How to alter a table schema in SQL Server?

Answer:

Altering Tables: It is used to modify an existing table.

1. CREATE TABLE Stock
2. (
3. ID SMALLINT
4.);
5. mysql > ALTER TABLE Stock -
   > ADD COLUMN Quantity SMALLINT UNSIGNED NOT NULL, -
   > MODIFY ID SMALLINT UNSIGNED NOT NULL, - > ADD PRIMARY KEY(ID);
6. mysql > Describe Stock;
7. mysql > ALTER TABLE Stock;

Example in Sql:

```
SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[Trainee1](
    [S_No] [int] NULL,
    [ID] [varchar](30) NULL,
    [Name] [char](25) NULL,
    [Age] [int] NULL,
    [City] [char](20) NULL,
    [Qualification] [varchar](20) NULL,
    [Contact_No] [varchar](30) NULL,
    [Country] [char](20) NULL,
UNIQUE NONCLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
) ON [PRIMARY]

GO

SET ANSI_PADDING OFF
GO

select * from Trainee1
alter table Trainee1 drop Column country
```

Question 49: How to create index in SQL Server?

Answer: Indexes are data structures that are used to improve the searching speed in a table. The user can not see the index directly. An index increases the performance of select statements and where clausees and slows down insert and update statements. So we create indexes only for those columns that are not frequently updated.

Creation of index:

Example:

```
create index i_select on emp(empName)
```

Creation of composite index: It is created on more than one column of the table using.

Example:

create index i_select on emp(empId,empName)

Creation of Unique index: Used for Data Integrity. A Unique index does not allow any duplicate values to be inserted into the table.

Example:

create index i_unique on emp(empId)

Question 50: How to get unique records in SQL?

Answer: Unique Constraint: It ensures that each row for a column must have a unique value. It is like a Primary key but it can accept only one null value. In a table one or more column can contain a Unique Constraint.

Column Level:
Syntax:

1. **Create Table** Table_Name
2. (
3. Column_Name Datatype **Constraint** Constraint_Name **Unique**
4.)

Example:

1. **Create Table** MY_Tab
2. (
3. IId **int constraint** Unique_Cons **Unique** ,
4. **Name** nvarchar(50)
5.)

Question 51: How to create a date column in SQL Server?

Answer: Datetime data type can store dates from January 1, 1753 to December 31, 9999 with a precision up to 0.003 fraction of a second. The smalldate data type can store dates from January 1, 1900 to June 6, 2079 with a precision of a second.

1. create table tbDate
2. (col datetime);
3. go
4. insert into tbDate values('8:00 AM');
5. go
6. insert into tbDate values('March 24,2008');
7. go

There are styles to format the input and output when converting from datetime into characters. We must use the convert function and the following list of styles.

Style ID	Style Type
0 or 100	mon dd yyyy hh:miAM (or PM)
101	mm/dd/yy
102	yy.mm.dd
103	dd/mm/yy
104	dd.mm.yy
105	dd-mm-yy
106	dd mon yy
107	Mon dd, yy
108	hh:mm:ss
9 or 109	mon dd yyyy hh:mi:ss:mmmAM (or PM)
110	mm-dd-yy
111	yy/mm/dd
112	Yymmdd
13 or 113	dd mon yyyy hh:mm:ss:mmm(24h)
114	hh:mi:ss:mmm(24h)
20 or 120	yyyy-mm-dd hh:mi:ss(24h)
21 or 121	yyyy-mm-dd hh:mi:ss.mmm(24h)
126	yyyy-mm-dd Thh:mm:ss.mmm(no spaces)
130	dd mon yyyy hh:mi:ss:mmmAM
131	dd/mm/yy hh:mi:ss:mmmAM