**1) What is Entity Framework?**

Entity Framework is an ORM from Microsoft that will enable the developers to work with domain specific objects, which eliminates the extra code being written in the data access layer.

**2) Why to use Entity Framework?**

Writing ADO.NET code and managing it is a tedious job. To avoid this, Microsoft has provided a solution - Entity Framework. Entity Framework reduces a great deal of code by enabling to work with relational data as domain specific objects.

**3) What is the difference between Entity Framework and ADO.NET?**

| **ADO.NET** | **Entity Framework** |
| --- | --- |
| ADO.NET is faster. | "Entity Framework will be around the ADO.NET, which means ADO.NET is faster than Entity Framework." |
| We need to write so much code to talk to database. | Easy to use. As an Entity Framework will talk to database without much code involved. |
| Performance is better than Entity Framework. | Performance is not good compared to ADO.NET. |

**4) What is the difference between Entity Framework and LINQ to SQL?**

| **LINQ to SQL** | **Entity Framework** |
| --- | --- |
| Supports only for SQL database. | "Supports databases like SQL, MySQL, DB2 etc." |
| Allows only one-to-one mapping between the entity classes and the relational tables. | Allows one-to-one, one-to-many & many-to-many mappings between the Entity classes and the relational tables. |
| .dbml file is generated for maintaining the relationships. | Relationships are maintained in 3 different files - .csdl, .msl and .ssdl |
| No Support for Complex Types | Supports Complex Types |
| Supports Rapid application development with SQL Server. | Supports Rapid application development with SQL Server, DB2, MySQL etc. |

**5) What are the components of Entity Framework Architecture?**

Below are the components of Entity Framework –

* Entity Data Model (EDM)
* LINQ to Entities
* Entity SQL
* Object Service
* Entity Client Data Provider
* ADO.Net Data Provider

**6) What are the parts of Entity Data Model (EDM)?**

Below are the parts of Entity Data Model –

* Conceptual Model
* Mapping
* Storage Model

**7) How to create Entity Data Model (EDM)?**

Add New Item -> ADO.NET Entity Data Model, which generates file with .edmx extension.

**8) What does .edmx consists of?**

.edmx file is a XML file and it has Conceptual Model, Storage Model and Mapping details i.e,

* SSDL (Store schema definition language)
* CSDL (Conceptual schema definition language)
* MSL (Mapping specification language)

**9) What is Conceptual Model?**

Conceptual Models are the model classes which contain the relationships. These are independent of the database design.

**10) What is Storage Model?**

Storage Models are our database design models, which contains database tables, views, stored procs and keys with relationships.

**11) What is mapping?**

The Mapping will have the information on how the Conceptual Models are mapped to Storage Models.

**12) What is LINQ to Entities?**

LINQ to Entities is a query language which we used to write queries against the object models and the query result will return the entities defined in the Conceptual Model.

**13) What is Entity SQL?**

Entity SQL is a query language is like LINQ to Entities. This is a bit complex compared to LINQ to Entities. A developer who is using this should learn this separately.

**14) What is the role of Entity Client Data Provider?**

Responsibility of Entity Client Data Provider is to convert the LINQ to Entities or Entity SQL queries to a SQL query, which is understood by the underlying database. This finally communicates with ADO.NET Data Provider which in turn used to talk to the database.

**15) What is the meaning of Pluralize and Singularize in Entity Framework?**

Pluralize and Singularize gives the meaningful naming conventions for objects.

We will get this option while adding an edmx file. On selecting this option Entity Framework will adhere to Singular or Plural coding conventions.

**16) What is DB Context?**

When we create an edmx file, it will have the list of entities and context class which will be derived from DB Context class.

**17) What is the role of DB Context?**

DBContext will be responsible for Insert, Update and Delete functionalities of the entities. It acts like a bridge between the database and entity classes.

**18) What is Object Context?**

Object Context manages all the database operations, like database connection, and manages various entities of the Entity Model. DB Context is a wrapper around Object Context.

**19) What is the difference between ObjectContext and DbContext?**

Conceptually, both these are here for the same reason.

* DBContext is wrapper around Object Context.
* Prior to EF 4.1, EDM used to use Object Context as base class for Context classes.
* There were some difficulties been observed in Object Context so now Db Context is introduced in EF 6.0 and this is used for all the development models –
* Database First, Model First and Code First.
* Object Context supports complied queries, but DB Context not.
* Object Context supports self-tracking entities, but DB Context does not.
* DB Context is thread safe, but Object Context not.

**20) What is Entity Set?**

Entity Set holds the Entity Types and this most of the times compared with a database table.

**21) What is Association Set?**

Association Set is used to define the relationship between Entity Sets.

**22) What is Entity Container?**

It is a wrapper for Association Sets and Entity Sets and this is a critical to query a model.

**23) What is Entity Graph?**

Entity Graph is when one entity has a relation with other entity.

**24) What is T4 Templates?**

T4 Template (Text Template Transformation Toolkit) will generate the C# code based on edmx XML file. (.tt extension)

**25) What are the types of Entities in Entity Framework?**

* POCO Entity
* Dynamic Proxy

**26) What is POCO Entity?**

POCO Entity is Plain old CLR objects. These can be used as domain entities with our models.

**27) What are the different approaches supported in the Entity Framework to create Entity Model?**

Below are the different approaches supported in the Entity Framework to create Entity Model –

* Database First
* Model First
* Code First

**28) What is the Database First Approach?**

This approach is suitable, if we have a database already created and ready to use it. Using the existing database, we can create the Entity Models.

**29) What is the Model First Approach?**

This approach is suitable, when we prefer to create the Entity Models first and derive the database from the Entity Models.

**30) What is the Code First Approach?**

This approach is suitable, when we prefer to create the Domain classes first and derive the database from the Domain classes.

**31) What are the advantages of Database First Approach?**

Below are the advantages of Database First Approach –

* Easy to create entity models if there is an existing database.
* Preferred approach for data intensive applications.

**32) What are the disadvantages of Database First Approach?**

Below are the disadvantages of Database First Approach –

* Once we create a edmx file from an existing database, huge pile of code is generated.
* If we want to add the additional functionality to the models generated, we need to extend the models.

**33) What are the advantages of Model First Approach?**

Below are the advantages of Model First Approach –

* Model first approach gives the flexibility to design the Entity Models independently and gives an option to improve at later stages.
* Model classes can be created by drawing it in the edmx designer, so no much of database is required.

**34) What are the advantages of Code First Approach?**

Below are the advantages of Code First Approach –

* Based on business objects we can decide the database structure.
* We can decide which classes need to be serialized and can specify the collection to eager load.
* Good for smaller applications.

**35) What are the disadvantages of Code First Approach?**

Below are the disadvantages of Code First Approach –

* All database related stuffs should be included in the code.
* For Stored Procs, we need to use the Fluent APIs to write it in code.
* Not good for data intensive applications.

**36) What are the Entity States supported in Entity Framework?**

Below are the States supported in Entities during lifetime –

* Added
* Deleted
* Modified
* Un Changed
* Detached

**37) What is Connected Scenario in Entity Framework?**

Connected Scenario is when the Entity is retrieved from the database and some operations like modification is done in the same context.

**38) What is Disconnected Scenario in Entity Framework?**

Disconnected Scenario is when the Entity is retrieved from the database and some operations like modification is done in the different context. In this scenario context does not understand the changes done so we have to specify the changes done outside of context.

**39) How we can increase the performance of Entity Framework?**

Below are the points we can consider increasing the performance –

* Disable the Change Tracking if it’s not required.
* Use the compiled query whenever required.
* Avoid using Views
* Fetch the required data from database.

**40) What are the ways we can load the related entities in Entity Framework?**

Below are the ways to load the entities in Entity Framework –

* Lazy Loading
* Eager Loading
* Explicit Loading

**41) What is Lazy Loading in Entity Framework?**

Lazy Loading is the default behavior of Entity Framework, wherein the dependent/related entities are loaded once they are accessed for the first time.

**42) Give an example for Lazy Loading?**

Consider a scenario of Student and Courses. Student can enroll to multiple Courses.

Filter the Student based on Student Id.

Student s = dbContext.Students.FirstOrDefault(a => a. StudentId == sId);

Load all the Courses related to Student.

List<Course> courses = s.Courses;

**43) What is Eager Loading in Entity Framework?**

Eager Loading will load the dependent/related entities at once. Unlike Lazy loading, Eager loading will do only one database call and get all the dependent entities.

**44) Give an example for Eager Loading?**

Consider a same Student scenario -

Student s = dbContext. Students.Include(s => s. Courses).FirstOrDefault(s => s. StudentId == sId);

**45) What is Explicit Loading in Entity Framework?**

Be default Entity Framework supports Lazy loading, but we can disable the Lazy loading and we can still load the dependent/related entities by calling “Load” method.

**46) Give an example for Explicit Loading?**

Continuing Student Scenario -

Student s = dbContext. Students.FirstOrDefault(a => a. StudentId == sId);

dbContext.Entry(s).Reference(s => s. Courses).Load();

**47) Which type of loading is good in which scenario?**

* Use Eager Loading, when the relations are not too much so that we can get the data in one database query.
* Use Eager Loading, if you are sure that we are going to use the dependent/related entities.
* If there are one-to-many relations use Lazy loading and if the dependent/related entities are not required down the line.
* When Lazy Loading is turned off, use explicit loading when you are not sure whether dependent/related entities are not required down the line.

**48) Can we use Stored Procedures in Entity Framework?**

Yes.

**49) Do Entity Framework handle the Change Tracking?**

Yes.

**50) How Entity Framework handles the Change Tracking?**

Entity Framework supports automatic change tracking of all the loaded entities through context class.

**51) What is the minimum requirement for Change Tracking in Entity Framework?**

If Entity Framework need to handle Change Tracking, each entity should have Entity Key (Primary Key).

**52) Mention what is CSDL, SSDL and MSL sections in an EDMX file?**

* **CSDL:** It stands for **Conceptual Schema Definition Language,**it is the conceptual abstraction which is exposed to the application
* **SSDL:** It stands for **Storage Schema Definition Language**, it defines the mapping with our RDBMS data structure
* **MSL:**It stands for **Mapping Schema Language**, it connects the SSDL and CSDL