এখানে **Lead Software Engineer** পদের জন্য **Entity Framework (EF)** সম্পর্কিত গুরুত্বপূর্ণ ইন্টারভিউ প্রশ্ন ও উত্তরগুলোর বাংলা অনুবাদ দেওয়া হলো। এগুলো মূলত আপনার অভিজ্ঞতা যাচাই, আর্কিটেকচার ডিজিশন এবং পারফরম্যান্স অপ্টিমাইজেশনের উপর ভিত্তি করে তৈরি করা হয়েছে।

**🔹 প্রশ্ন ১: Entity Framework কি?**

**উত্তর:**  
Entity Framework (EF) হচ্ছে .NET এর জন্য একটি ORM (Object Relational Mapper), যেটি ডেভেলপারদেরকে ডাটাবেজ অপারেশন করতে দেয় C# বা VB.NET কোড ব্যবহার করে SQL কোয়েরি না লিখেই। এটি ডাটাবেজ টেবিল ও C# ক্লাসগুলোর মাঝে ম্যাপিং করে।

**🔹 প্রশ্ন ২: Entity Framework এর কোন কোন ওয়ার্কফ্লো রয়েছে?**

**উত্তর:**  
EF-এর ৩টি ওয়ার্কফ্লো রয়েছে:

1. **Database-First:** প্রথমে ডাটাবেজ তৈরি করা হয়, পরে EF এর মাধ্যমে মডেল জেনারেট করা হয়।
2. **Model-First:** প্রথমে মডেল ডিজাইন করা হয়, পরে EF ডাটাবেজ তৈরি করে।
3. **Code-First:** C# কোড দিয়ে মডেল তৈরি করা হয়, তারপর সেই মডেল থেকেই ডাটাবেজ তৈরি হয়।

**🔹 প্রশ্ন ৩: DbContext ও DbSet কী?**

**উত্তর:**

* DbContext: এটি EF এর মূল ক্লাস, যা ডাটাবেজের সাথে কমিউনিকেশন করে।
* DbSet<T>: এটি ডাটাবেজের একটি টেবিলের প্রতিনিধিত্ব করে। উদাহরণ:
* public class AppDbContext : DbContext
* {
* public DbSet<Customer> Customers { get; set; }
* }

**🔹 প্রশ্ন ৪: Entity Framework এ Lazy Loading, Eager Loading ও Explicit Loading এর পার্থক্য কী?**

**উত্তর:**

| **ধরণ** | **বর্ণনা** |
| --- | --- |
| Lazy Loading | প্রয়োজন হলে তখন ডাটাগুলো লোড হয়। |
| Eager Loading | একসাথে মূল ডেটা ও সম্পর্কিত ডেটা লোড হয় (Include() ব্যবহার করে)। |
| Explicit Loading | Lazy Loading এর মতো, তবে প্রোগ্রামার নিজে নির্ধারণ করে কখন Load() করবে। |

**🔹 প্রশ্ন ৫: Entity Framework এ Migrations কী?**

**উত্তর:**  
Migrations ব্যবহার করে কোড-ফার্স্ট মডেলে ডাটাবেজ স্কিমা পরিবর্তন ট্র্যাক করা যায়। এতে করে ডাটাবেজ ড্রপ না করেই পরিবর্তন করা যায়।

**কম্যান্ড উদাহরণ:**

Add-Migration AddCustomerTable

Update-Database

**🔹 প্রশ্ন ৬: Entity Framework Performance Optimization কিভাবে করবেন?**

**উত্তর:**

* .AsNoTracking() ব্যবহার করলে ট্র্যাকিং বন্ধ হয়, যা পারফরম্যান্স বাড়ায়।
* প্রয়োজন অনুযায়ী Include() দিয়ে Eager Loading করুন।
* Bulk Operation এর জন্য EF Core Bulk Extensions ব্যবহার করুন।
* কমপ্লেক্স কোয়েরি SQL View বা Stored Procedure দিয়ে হ্যান্ডেল করুন।

**🔹 প্রশ্ন ৭: EF Core vs EF 6 এর মধ্যে পার্থক্য কী?**

**উত্তর:**

| **বৈশিষ্ট্য** | **EF Core** | **EF 6** |
| --- | --- | --- |
| Cross-platform | হ্যাঁ (.NET Core) | না |
| Performance | বেশি দ্রুত | তুলনামূলক ধীর |
| LINQ অপশন | আধুনিক ও উন্নত | পুরোনো |
| Future Support | সক্রিয় | সীমিত |

**🔹 প্রশ্ন ৮: Entity Framework এ Transaction কিভাবে হ্যান্ডেল করবেন?**

**উত্তর:**

using var transaction = context.Database.BeginTransaction();

try

{

context.SaveChanges();

transaction.Commit();

}

catch

{

transaction.Rollback();

}

**🔹 প্রশ্ন ৯: Stored Procedure কিভাবে কল করবেন EF এ?**

**উত্তর:**

var result = context.Customers

.FromSqlRaw("EXEC GetAllCustomers")

.ToList();

**🔹 প্রশ্ন ১০: Entity State Management কী?**

**উত্তর:**  
EF Entity Object গুলোর অবস্থা ট্র্যাক করে:

| **অবস্থা** | **বর্ণনা** |
| --- | --- |
| Added | নতুন Entity |
| Modified | পরিবর্তিত Entity |
| Deleted | মুছে ফেলার জন্য চিহ্নিত |
| Unchanged | কোন পরিবর্তন নেই |
| Detached | ট্র্যাক করা হচ্ছে না |

নিচে **Lead Software Engineer** পদে ইন্টারভিউয়ের জন্য **Entity Framework (EF)** নিয়ে আরো **বিস্তারিত ব্যাখ্যা ও প্রশ্নোত্তর** দেওয়া হলো — একদম বাংলায় ও গভীরভাবে, যাতে আপনি বাস্তব জীবনের প্রজেক্টে এই জ্ঞান প্রয়োগ করতে পারেন।

**🔹 প্রশ্ন ১: Entity Framework (EF) কী এবং এটি কেন ব্যবহার করা হয়?**

**🧠 ব্যাখ্যা:**

Entity Framework হচ্ছে Microsoft-এর একটি ORM (Object Relational Mapping) ফ্রেমওয়ার্ক, যা ডাটাবেজ ও অবজেক্ট ওরিয়েন্টেড কোড (C# ক্লাস) এর মধ্যে ব্রিজ হিসেবে কাজ করে। এটি SQL লেখার পরিবর্তে LINQ ব্যবহার করে ডেটা অ্যাক্সেস করার সুবিধা দেয়।

**✅ আপনি কেন EF ব্যবহার করবেন?**

* ডাটাবেজে সরাসরি SQL লেখার দরকার পড়ে না।
* কোড থেকে সরাসরি ডাটাবেজ মডেল তৈরি করা যায় (Code First)।
* টাইপ-সেইফ কোয়েরি লেখা যায়, কমপাইল টাইমে ভুল ধরা যায়।
* Readability ও Maintainability বাড়ায়।

**🔹 প্রশ্ন ২: Entity Framework এর কোন কোন ওয়ার্কফ্লো রয়েছে এবং আপনি কোনটি ব্যবহার করবেন কেন?**

**🧠 ব্যাখ্যা:**

Entity Framework ৩টি উপায়ে ব্যবহার করা যায়:

| **পদ্ধতি** | **ব্যবহার** | **কখন ব্যবহার করবেন** |
| --- | --- | --- |
| Database First | ডাটাবেজ আগে তৈরি হয়, তারপর মডেল জেনারেট | ডাটাবেজ আগে থেকেই থাকলে |
| Model First | মডেল আগে ডিজাইন, পরে ডাটাবেজ তৈরি | কম ব্যবহার হয় |
| Code First | C# ক্লাস লিখে ডাটাবেজ তৈরি | মডার্ন ও অনেক কাস্টমাইজেশন দরকার হলে |

**✅ Interview Tips:**

একজন লিড ইঞ্জিনিয়ার হিসেবে, আপনি Code First বেশি পছন্দ করবেন কারণ এটি Source Control এ রাখা সহজ, Version Control Friendly, এবং Continuous Deployment-এর জন্য উপযোগী।

**🔹 প্রশ্ন ৩: DbContext ও DbSet কী?**

**🧠 ব্যাখ্যা:**

* DbContext: এটি ডাটাবেজের সাথে সংযোগ স্থাপন এবং ডেটা CRUD অপারেশন করার মূল API।
* DbSet<T>: ডাটাবেজের টেবিলের মতো কাজ করে। T হলো আপনার Entity ক্লাস।

public class AppDbContext : DbContext

{

public DbSet<Product> Products { get; set; }

}

এক কথায়, DbContext ডাটাবেজকে represent করে, আর DbSet একটি টেবিলকে।

**🔹 প্রশ্ন ৪: Lazy, Eager এবং Explicit Loading কী এবং কবে কোনটি ব্যবহার করবেন?**

**🧠 ব্যাখ্যা:**

| **Loading টাইপ** | **Description** | **Example** | **ব্যবহার সময়** |
| --- | --- | --- | --- |
| Lazy Loading | প্রয়োজন হলে তখনই ডেটা লোড হয় | navigation property অ্যাক্সেস করলে | পারফরম্যান্স ভালো রাখার জন্য |
| Eager Loading | শুরুতেই সব রিলেটেড ডেটা লোড হয় | .Include(x => x.Orders) | যখন সব রিলেশন দরকার হয় |
| Explicit Loading | Lazy এর মতো, কিন্তু নিজে থেকে .Load() করতে হয় | context.Entry(entity).Collection(e => e.Orders).Load(); | যখন শর্তসাপেক্ষে লোড করতে হয় |

Interview Hint: Eager Loading বড় Query-তে Join দিয়ে Data আনতে পারে, পারফরম্যান্স degrade করতে পারে — তাই সচেতনভাবে ব্যবহার করতে হয়।

**🔹 প্রশ্ন ৫: EF Code First Migrations এর কাজ কী?**

**🧠 ব্যাখ্যা:**

Migration একটি version-control পদ্ধতি, যার মাধ্যমে আপনি মডেল ক্লাসের পরিবর্তনগুলো ডাটাবেজে প্রয়োগ করতে পারেন।

**✅ কমান্ডসমূহ:**

Add-Migration AddCustomerTable

Update-Database

**Interview Tips:**

Migration ব্যবহার করে ডাটাবেজ আপডেট করলে ডেটা লস হয় না এবং টিম-ভিত্তিক ডেভেলপমেন্টে সহজে track রাখা যায়।

**🔹 প্রশ্ন ৬: Entity Framework এ পারফরম্যান্স অপ্টিমাইজেশন কিভাবে করবেন?**

**🧠 ব্যাখ্যা:**

1. **AsNoTracking()** ব্যবহার করুন — যদি আপনি শুধু রিড অপারেশন করেন।
2. var customers = context.Customers.AsNoTracking().ToList();
3. **Selective Include()** ব্যবহার করুন — সব রিলেশন একসাথে না এনে প্রয়োজন অনুযায়ী আনুন।
4. **Pagination** প্রয়োগ করুন:
5. var pageData = context.Customers
6. .Skip((page-1)\*pageSize)
7. .Take(pageSize)
8. .ToList();
9. **Raw SQL বা Stored Procedure ব্যবহার করুন** যদি Complex Query থাকে।

**🔹 প্রশ্ন ৭: EF Core vs EF 6: কী কী পার্থক্য আছে?**

| **বৈশিষ্ট্য** | **EF Core** | **EF 6** |
| --- | --- | --- |
| .NET Version | .NET Core, .NET 5/6+ | .NET Framework |
| Performance | উন্নত | তুলনামূলক ধীর |
| Cross-Platform | হ্যাঁ | না |
| LINQ Improvements | বেশি | কম |
| Future Development | Active | Deprecated ধাপে |

Interview Hint: নতুন প্রজেক্ট হলে EF Core-ই ব্যবহারযোগ্য, বিশেষ করে Microservices বা .NET 6+ প্ল্যাটফর্মে।

**🔹 প্রশ্ন ৮: Entity Framework এ ট্রানজ্যাকশন কিভাবে হ্যান্ডেল করবেন?**

**🧠 ব্যাখ্যা:**

একাধিক ডেটাবেজ অপারেশন একই ট্রানজ্যাকশনে রাখতে চাইলে এইভাবে করা হয়:

using var transaction = context.Database.BeginTransaction();

try

{

context.SaveChanges();

transaction.Commit();

}

catch

{

transaction.Rollback();

}

Interview Hint: DDD বা Microservice এ টু-ফেজ কমিট দরকার হলে, ট্রানজ্যাকশন boundary কে খুব ভালভাবে ডিজাইন করতে হবে।

**🔹 প্রশ্ন ৯: Stored Procedure কিভাবে কল করবেন EF-এ?**

**🧠 উদাহরণ:**

var data = context.Customers

.FromSqlRaw("EXEC GetAllCustomers")

.ToList();

Complex Report বা Performance-sensitive কোয়েরির জন্য Stored Procedure ব্যবহার উপযোগী।

**🔹 প্রশ্ন ১০: Entity State Management কী এবং কেন গুরুত্বপূর্ণ?**

**🧠 ব্যাখ্যা:**

EF Entity Object গুলোর current state track করে:

| **EntityState** | **Description** |
| --- | --- |
| Added | নতুন Entity |
| Modified | Entity তে পরিবর্তন হয়েছে |
| Deleted | Entity মুছে ফেলার জন্য চিহ্নিত |
| Unchanged | কোন পরিবর্তন হয়নি |
| Detached | ট্র্যাকিং হচ্ছে না |

context.Entry(customer).State = EntityState.Modified;

Interview Hint: আপনি যদি Detached Entity ক্লায়েন্ট থেকে পান, তখন state সেট করে EF কে জানাতে হয় এটি আপডেট হবে না insert।

**🔚 উপসংহার:**

একজন Lead Software Engineer হিসেবে আপনাকে শুধু Entity Framework API জানলেই চলবে না — বরং এটি কিভাবে Clean Architecture, DDD, CQRS বা Microservice Design এ integrate করতে হয় তাও জানতে হবে।