Outline

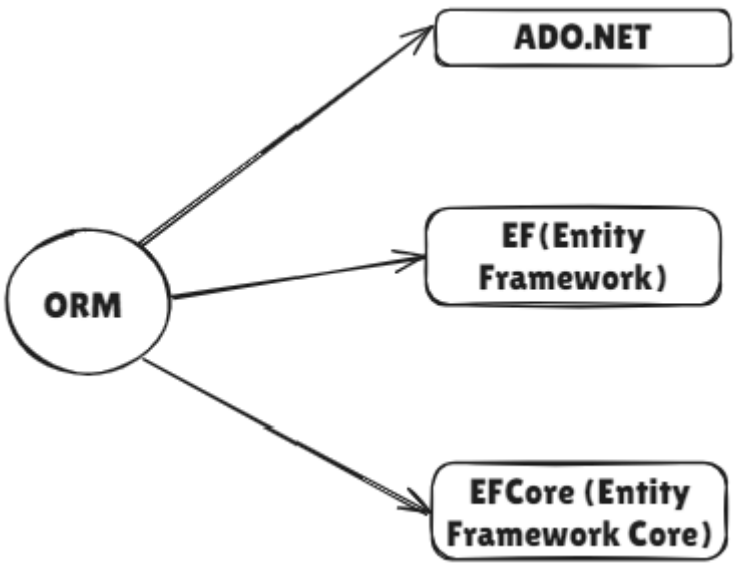
1. Introduction to ORM and EFCore
2. What is ADO.NET?
3. EF & EFCore & ADO.NET
4. Nuget Packages
5. Database Modeling with EF Core
6. Database First
7. Scaffold & .NET CLI
8. EF Core Power Tool
9. DbContext & DbSet
10. IEnumerable vs IQueryable(Expression Tree)
11. Update changes
12. Creating Models and DbContext
13. Configuring Relationships
    1. One-to-One
    2. One-to-Many
    3. Many-to-Many

EFCore

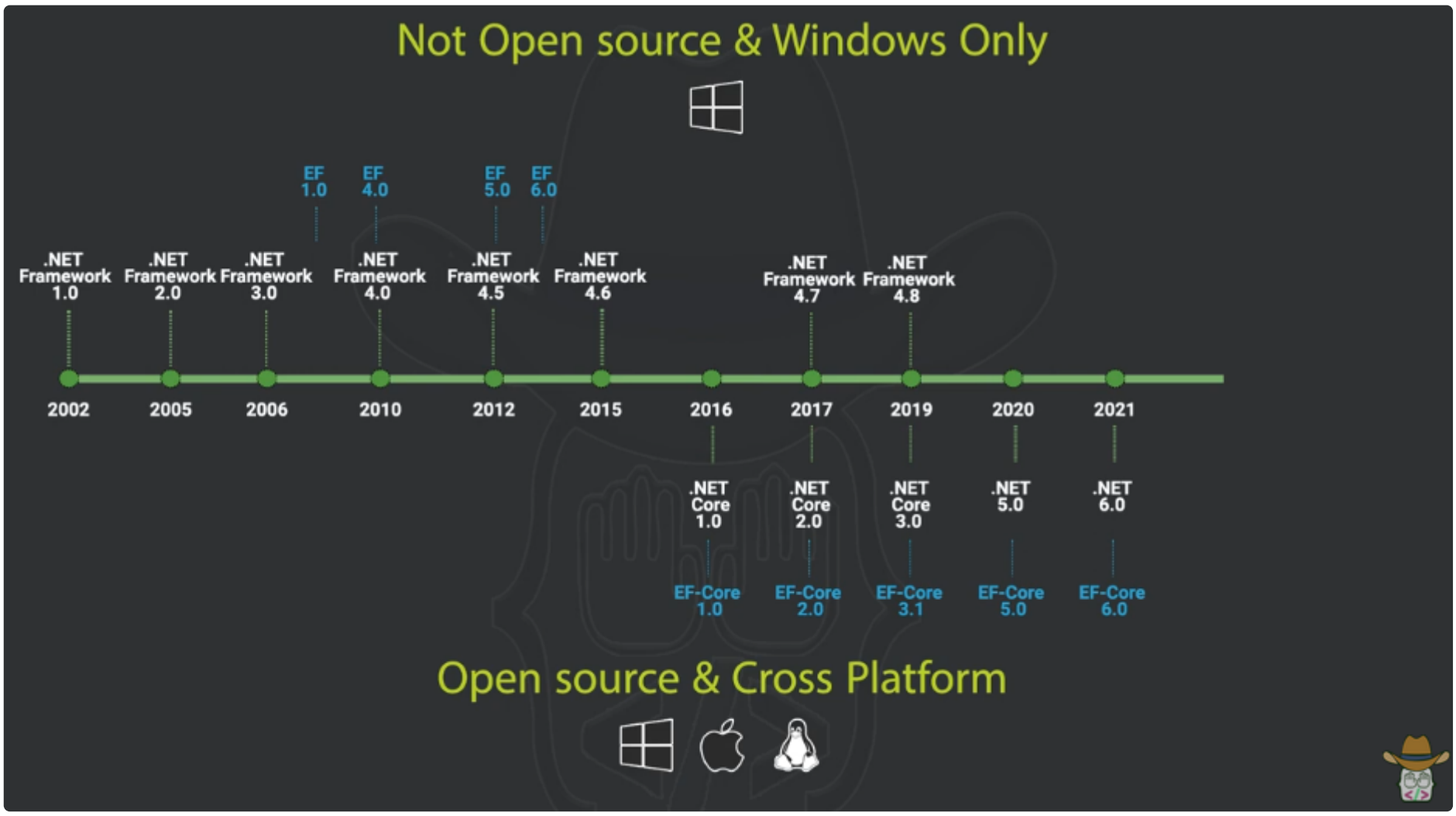
⇒ Entity Framework (EF) Core is a lightweight, extensible, open source and cross-platform version of the popular Entity Framework data access technology. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite, and other databases through a provider plugin API.

⇒ EF Core serve as an object-relational mapper (ORM), which:

- Enables working with a database using .NET objects.



⇒ **ORM (Object-Relational Mapping)** is a **programming technique ,** not a specific tool or framework. It’s a concept that can be implemented in various ways. ORM is realized through frameworks or libraries like **Entity Framework Core (EF Core), Dapper, NHibernate,** or others, depending on the programming language and ecosystem.



Comparison Table:

Feature/Aspect	ADO.NET	Entity Framework (EF6)	Entity Framework Core (EF Core)
Type	Low-level data access	High-level ORM	High-level ORM
Open Source	No	No	Yes
Platform Support	.NET Framework, .NET Core/.NET 5+	Primarily .NET Framework	Cross-platform (.NET Core, .NET 5+)
Performance	High (no ORM overhead)	Moderate (ORM overhead)	High (optimized for performance)
Ease of Use	Low (requires manual SQL)	High (abstracts SQL)	High (abstracts SQL)
LINQ Support	No	Yes	Yes
Change Tracking	Manual	Automatic	Automatic
Migrations	Manual	Automatic	Automatic
Cross-Platform	Yes (via .NET Core/.NET 5+)	No	Yes
Development Status	Stable	Maintenance mode	Actively developed

⇒ to use EFCore you to download its Packages

NuGet

⇒ NuGet is the package manager for .NET. It enables developers to create, share, and consume useful .NET libraries.

What is Package?

⇒ Compiled Library + Descriptive Metadata

Database Modeling with EF Core

Connection String

⇒ database providers require a connection string to connect to the database.

In sql server: it contains

1. server name
2. database name

can stored in:

1. variable
2. JSON file
3. using Azure key vault
4. using the Secret Manager tool

Database First

Scaffold (Reverse Engineering)

⇒ Reverse engineering is the process of scaffolding entity type classes and a DbContext class based on a database schema. It can be performed using the `Scaffold-DbContext` command of the EF Core Package Manager Console (PMC) tools or the `dotnet ef dbcontext scaffold` command of the .NET Command-line Interface (CLI) tools.

prerequisite:

- to use in visual studio : install PMC Tools
- another platforms: install .NET CLI
- Install the NuGet packages
  - `Microsoft.EntityFrameworkCore.Design`.
  - `Microsoft.EntityFrameworkCore.Tools` in the project you are scaffolding to.
  - Install the NuGet package for the database provider.

Scaffold command

- PWC Tools : `Scaffold-DbContext 'connection-string' provider`
  - example: `Scaffold-DbContext 'Data Source=HADEER;Initial Catalog=CourseStudentDB; Integrated Security = SSPI; TrustServerCertificate = True' Microsoft.EntityFrameworkCore.SqlServer`
- .NET CLI : ``
  - example `dotnet ef dbcontext scaffold "Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=Chinook" Microsoft.EntityFrameworkCore.SqlServer`

EF Core Power Tools

→ provide a graphical tool which builds on top of the EF Core command line tools and offers additional workflow and customization options.
→ Check Tool in GitHub

DbContext

⇒ the connection between database and Entities

its Role:

- Database Connections
- Data operations
- Change Tracking
- Model building
- Data Mapping
- Object caching
- Transaction management

types of Configuration

1. Internally
    1. using `onConfiguring method`
2. Externally
    1. using options

DbSet

- Represents tables in the database
- Supports LINQ queries
- Implements `IEnumerable` and `IQueryable`

Eager vs Deferred Execution

Eager: The results are computed right away, and the system does not wait for an explicit request to execute the code.

Deferred: delays the evaluation of expressions or operations until their results are explicitly needed

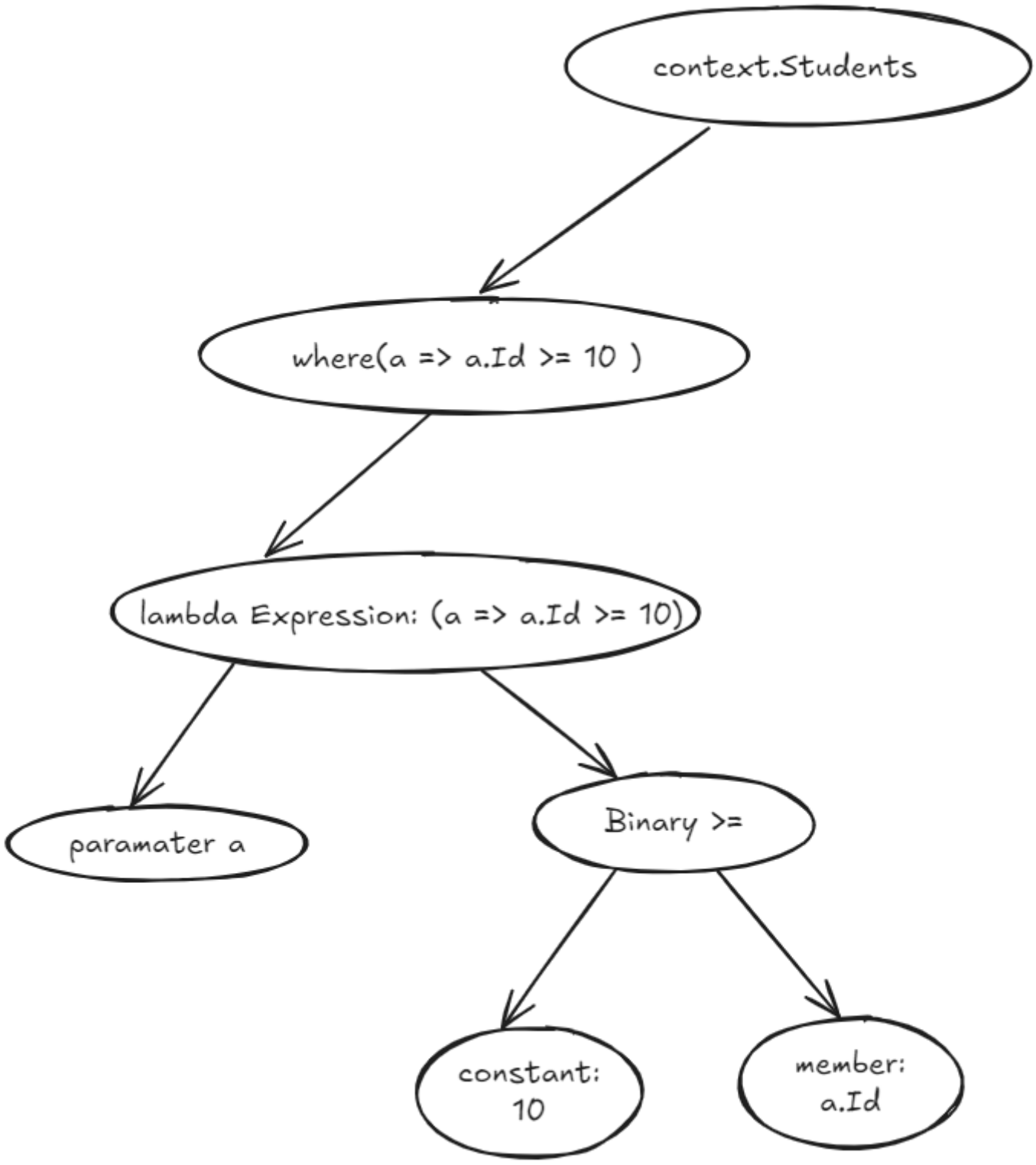
→ LINQ in .NET

IEnumerable vs IQueryable

Feature	**IEnumerable**	**IQueryable**
Execution	In-memory (C#)	Database Server
Query Processing	Fetches all data first	Translates to SQL
Performance	Less efficient for large datasets	More efficient
Use Case	Client-side filtering	Database-side filtering

Expression Tree

- Generate sql query
  `students = context.Students.Where(a=>a.Id>=10);`



Updating Data in EF Core

- Change tracking
- Object states: Detached, Modified, Unmodified
- Persisting changes to the database

Lets create our project and build model

Mapping Relation

1. one to one
    1. student and location (fk)
        1. in student class: 1 object of location
        2. in location class: 1 object of student
2. many to many
    1. student and courses: student have many coureses and courses have many students
3. one to many:

Resources

- [05] - Data Access & EF-Core - YouTube
- [02] - Databases (MS. SQL Server) | قواعد البيانات - YouTube
- dotnet official channel YouTube

- Connection Strings - EF Core | Microsoft Learn
- Learn Entity Framework Core - Getting Started EF Core Tutorial
- EF Core Documentation
- EF Core GitHub
- Power Tools Guide