

C# Data Access and Entity Framework Core

Rasmus Lystrøm
Associate Professor
ITU



Agenda

Databases

Old school SQL in C#

Package managers

Secrets

The IDisposable interface

SQL Injection

Object Relational Mapping

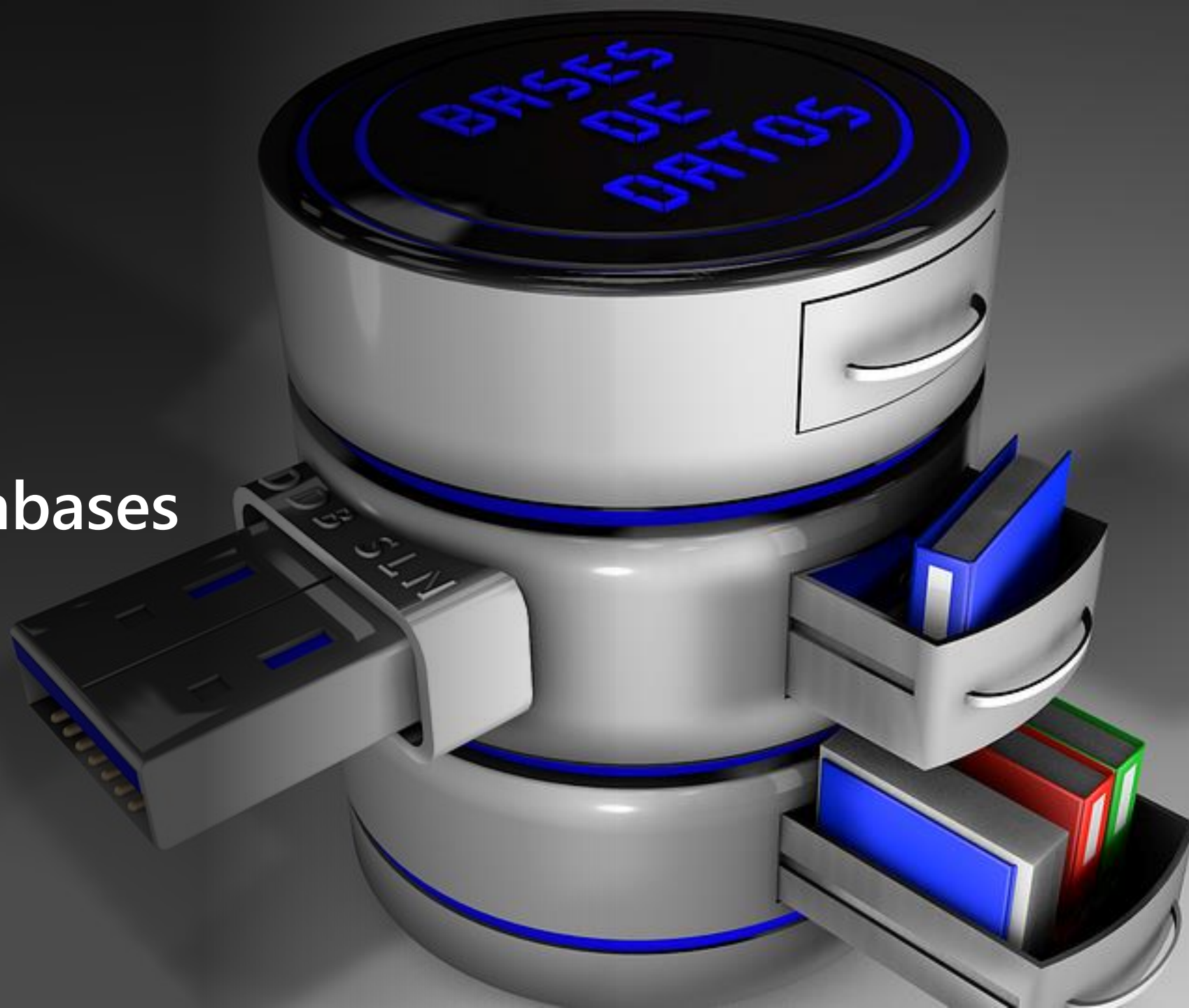
Entity Framework Core

Clean Onions

Lazy vs. Eager Loading



Databases



Databases

Microsoft SQL Server

Oracle Database

IBM Db2

MySQL

MariaDB

PostgreSQL

SQLite



[This Photo](#) by Unknown Author is licensed under [CC BY-SA-NC](#)

Azure Cosmos DB

Amazon DynamoDB

MongoDB

Couchbase

Redis

Elasticsearch

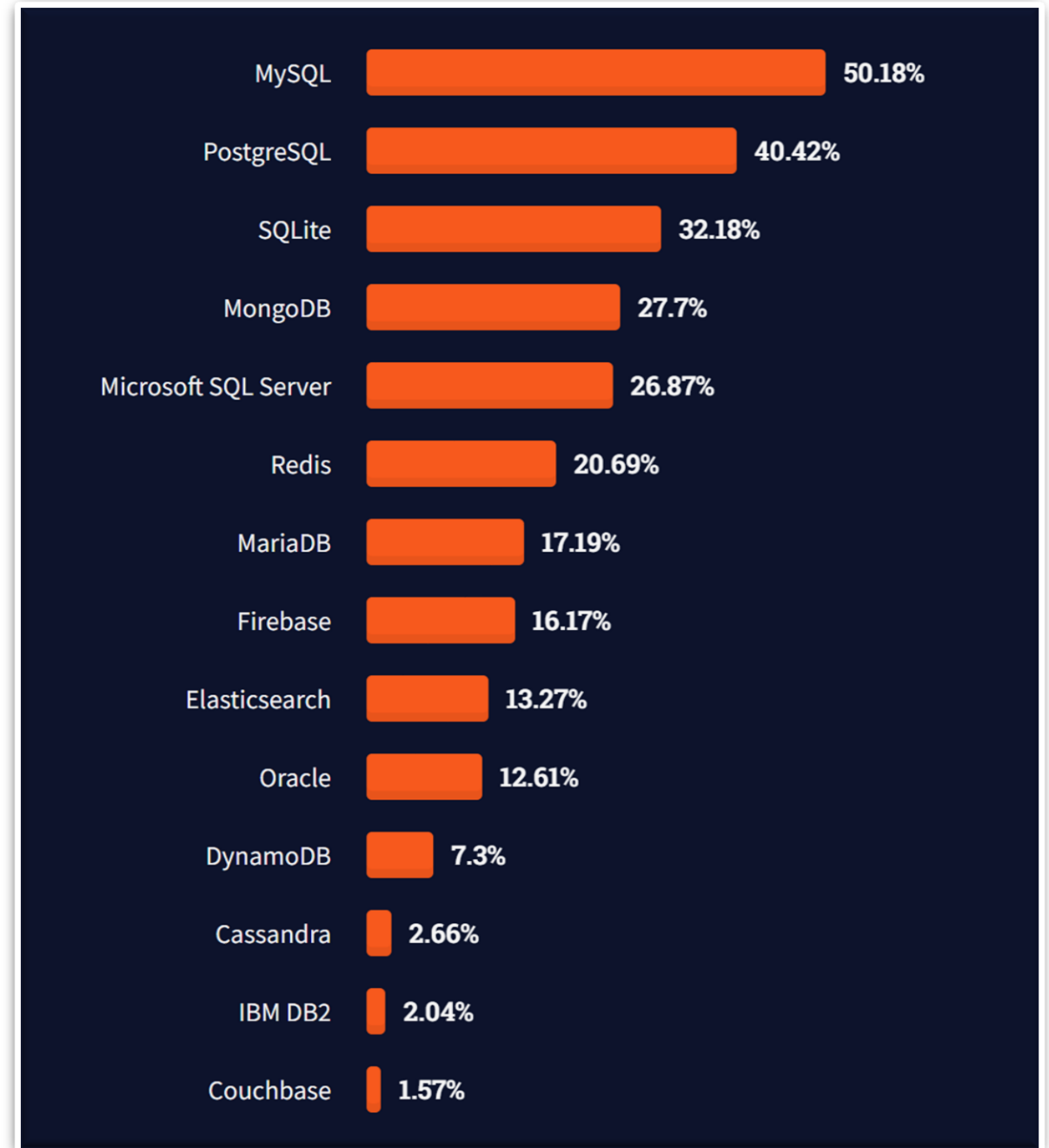
Neo4j



[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)

Most popular databases

<https://insights.stackoverflow.com/survey/2021#technology-most-popular-technologies>



SQL Server

```
docker pull mcr.microsoft.com/mssql/server:2019-latest
```

```
$password = New-Guid
```

```
docker run -e "ACCEPT_EULA=Y" -e "SA_PASSWORD=$password" `
  -p 1433:1433 `
  -d mcr.microsoft.com/mssql/server:2019-latest
```

SQL Server Demo

SQL Server Docker Container

Old School SQL

```
dotnet add package System.Data.SqlClient
```

```
var cmdText = "SELECT * FROM Animals";  
using var connection = new SqlConnection(connectionString);  
connection.Open();  
using var command = new SqlCommand(cmdText, connection);  
using var reader = command.ExecuteReader();  
while (reader.Read())  
{  
    ...  
}
```




Package Manager

NuGet Package Manager

Source: <https://www.nuget.org/>

dotnet add package ...

Criteria:

- Downloads
- License
- Dependencies
- Update frequency
- Check source repository

Secrets



Secrets

```
dotnet user-secrets init
```

```
dotnet user-secrets set "ConnectionStrings:ConnectionString" "..."
```

```
dotnet add package Microsoft.Extensions.Configuration.UserSecrets
```

Secrets

```
using Microsoft.Extensions.Configuration;
```

```
...
```

```
var configuration = new ConfigurationBuilder()  
    .AddUserSecrets<Program>()  
    .Build();
```

```
var connectionString = configuration.GetConnectionString("ConnectionString");
```


**HAVING TROUBLE WITH: THE CONNECTION
WAS NOT CLOSED. THE CONNECTION'S
CURRENT STATE IS OPEN. - SQL SERVER & C#**

SAY THAT AGAIN I DARE YOU

imgflip.com

https://twitter.com/overflow_meme/status/1223835574848630784

Disposable



IDisposable

```
var resource = new Resource();  
try  
{  
    ...  
}  
finally  
{  
    if (resource != null)  
    {  
        resource.Dispose();  
    }  
}
```


IDisposable II

```
var resource = new Resource();  
try  
{  
    ...  
}  
finally  
{  
    resource?.Dispose();  
}
```

IDisposable III

```
using (var resource = new Resource())  
{  
    ...  
}
```

IDisposable IV

```
using var resource = new Resource();
```

```
...
```


SQL Injection

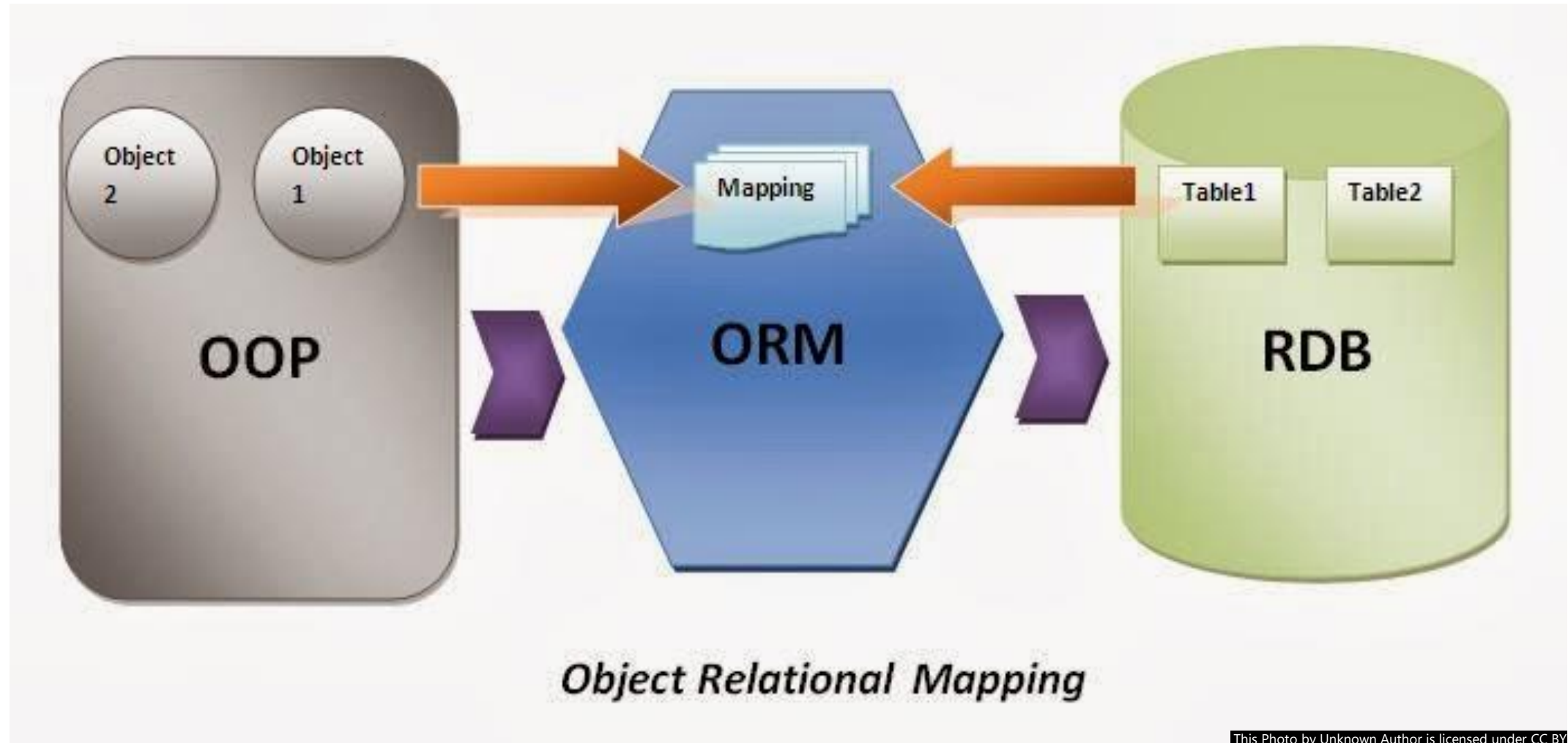
SQL Injection

A **SQL injection** attack consists of insertion or “injection” of a SQL query via the input data from the client to the application.

A successful SQL injection exploit can:

- read sensitive data from the database,
- modify database data (Insert/Update/Delete),
- execute administration operations on the database (such as shutdown the DBMS),
- or worse

Object Relational Mapping



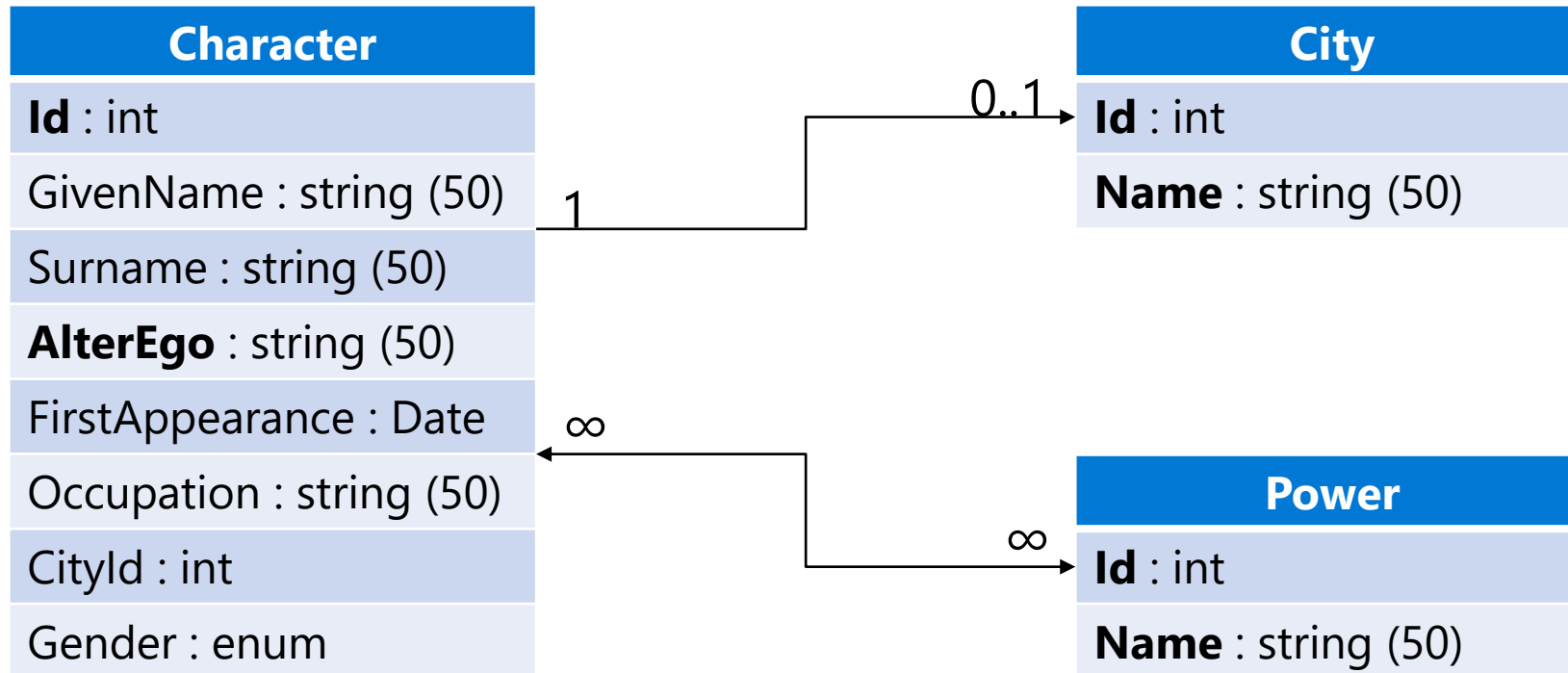
Object Relational Mapping

The act of converting incompatible types in OOP to tables/columns/rows/relations in SQL

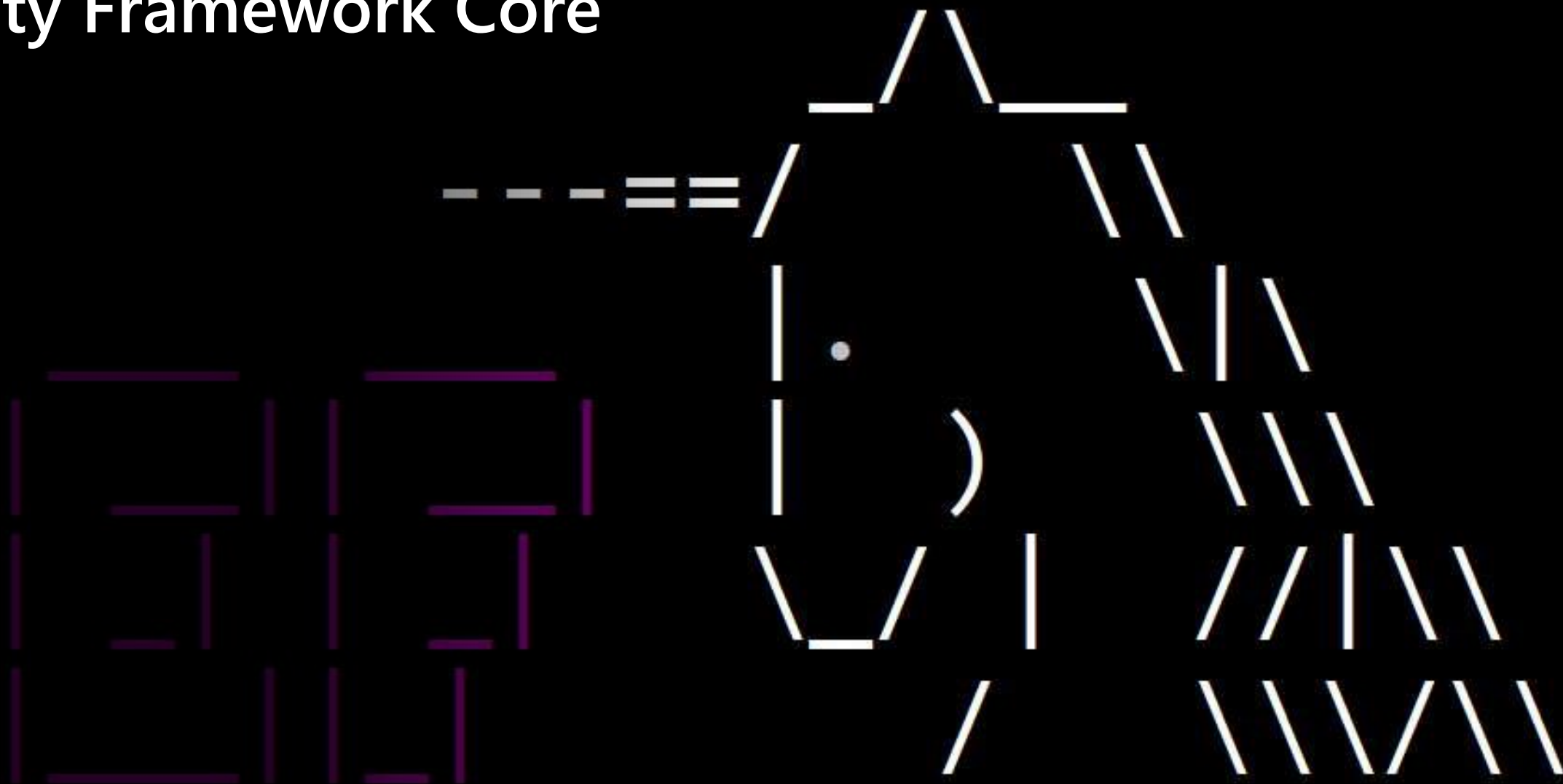
Object–relational impedance mismatch

- Me not understand object me table

Model



Entity Framework Core



Entity Framework Core

```
dotnet tool install --global dotnet-ef
```

```
dotnet add package Microsoft.EntityFrameworkCore.Design
```

```
dotnet ef migrations add InitialCreate
```

```
dotnet ef database update
```

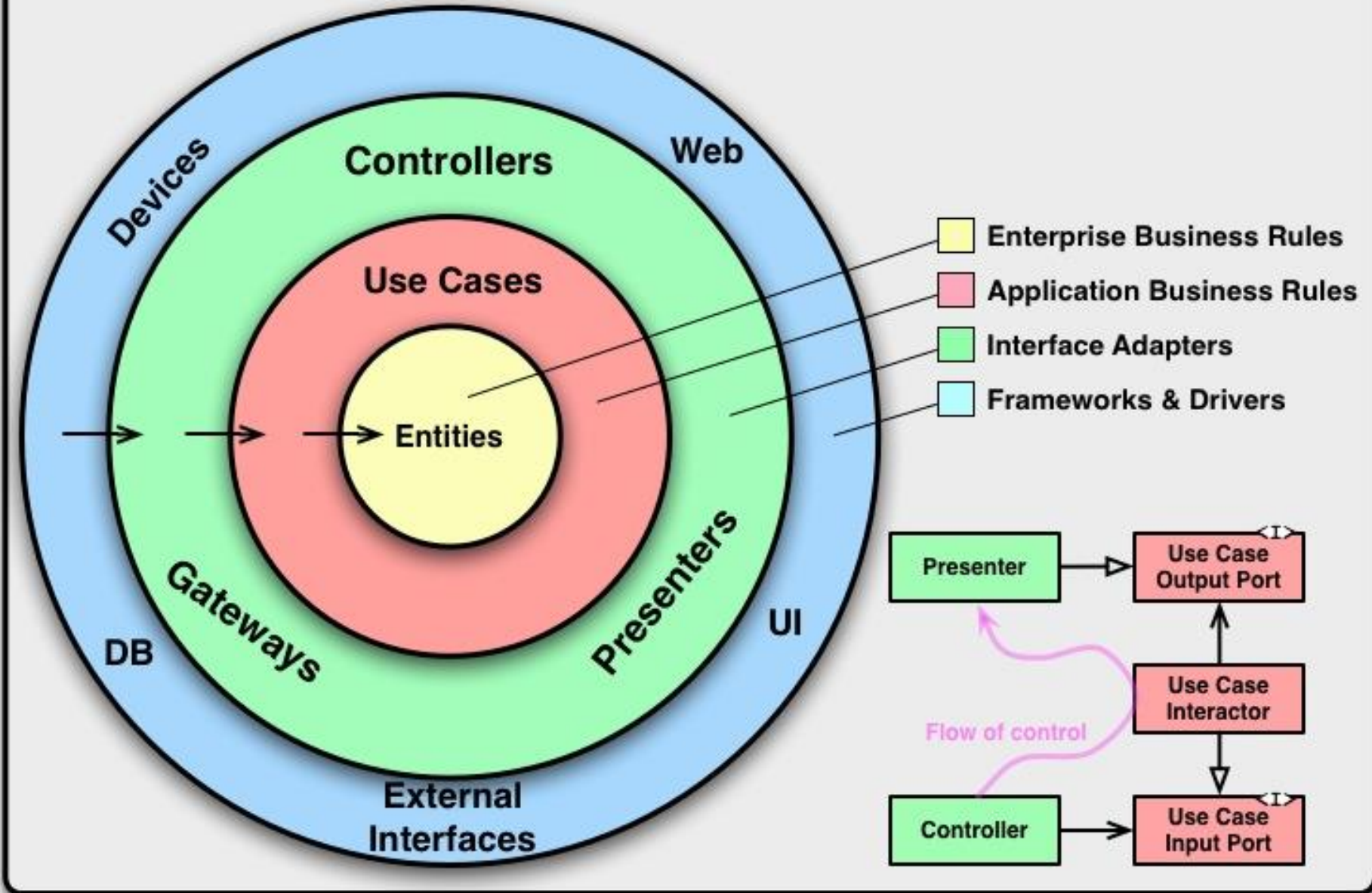
<https://docs.microsoft.com/en-us/ef/core/miscellaneous/cli/dotnet>

<https://docs.microsoft.com/en-us/ef/core/modeling/>



Onion Architecture

The Clean Architecture



Lazy Loading

<https://docs.microsoft.com/en-us/ef/core/querying/related-data/lazy>

Lazy Loading

```
dotnet add package Microsoft.EntityFrameworkCore.Proxies
```

```
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)  
    => optionsBuilder.UseLazyLoadingProxies()  
        .UseSqlServer(...);
```

Thank you

