LINQ & Entity Framework Core

Lesson 03 : DB First & Code First Approach

Capgemini

Lesson Objectives

In this lesson we will cover the following

- Database First Approach
- Code First Approach

EF Core Database Providers

- Entity Framework Core uses a provider model that allows EF to map with and access many databases.
- This includes concepts like, performing queries using LINQ, transactions, change tracking to objects etc.
- EF Core also supports database specific concepts e.g. memory-optimized tables provided by SQL Server.
- The library **Microsoft.EntityFrameworkCore.Relational** is used to create and build class providers for relational databases.
- This library provides APIs for managing table and column mapping.

APIs used in EF Core

- **DBContext -** The instance of the DbContext represents a database session that is used for performing CRUD operations.
- **DbSet -** This is used to define mapping with “T”. All the LINQ queries against DbSet will be translated into database queries. The important point here to understand is that all LINQ queries executed against DbSet will contain result returned from database, this may not include change made on database over the DbContext.
- These queries may be evaluated in-memory rather than converting into database queries. This is depending on the database provider.

Packages used for EF Core

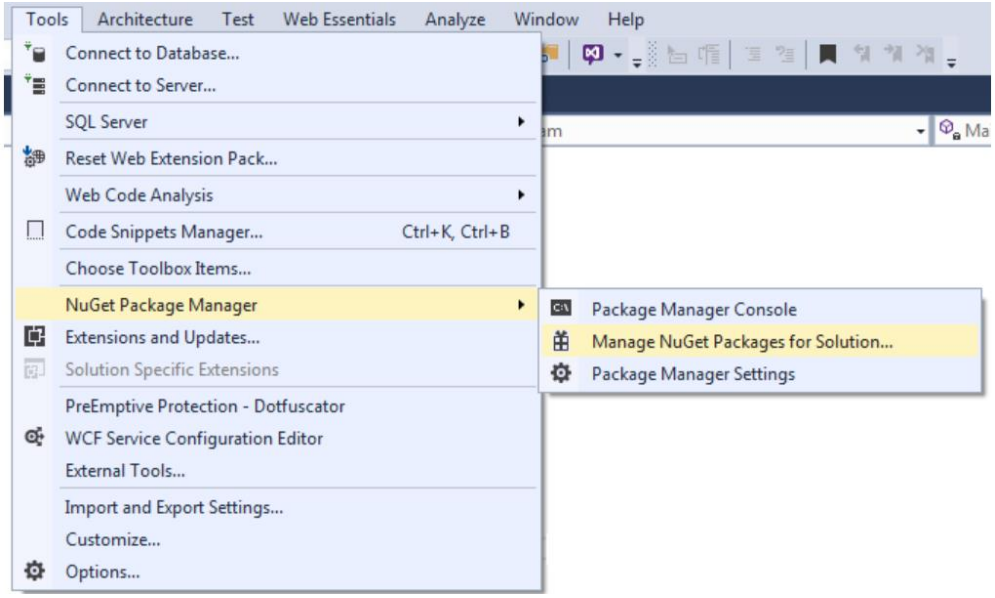
- To use EF Core in the application and since the application will be using Microsoft SQL Server database it need SQL Server provider.

- To use EF Core, application needs to install the following packages

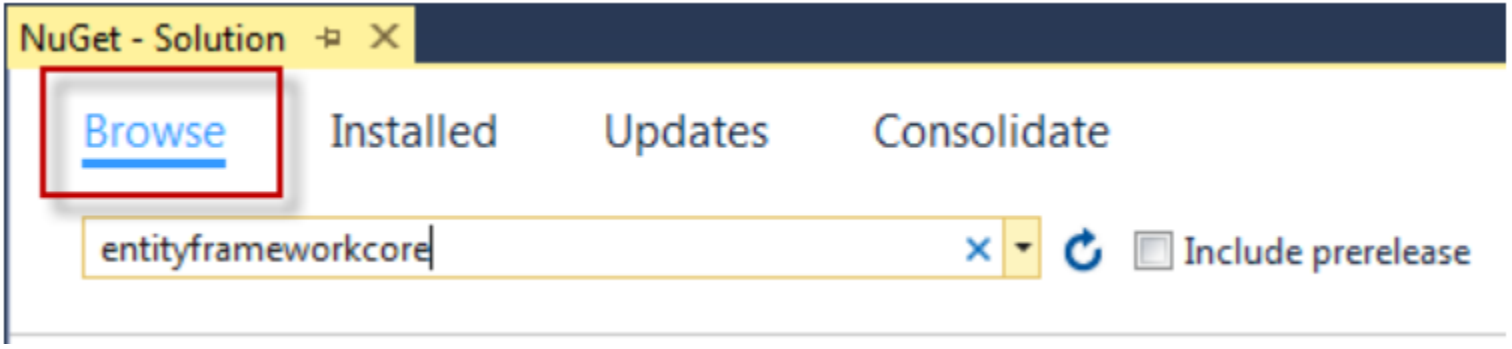
  - **Microsoft.EntityFrameworkCore.SqlServer**
  - **Microsoft.EntityFrameworkCore.Tools**
  - **Microsoft.EntityFrameworkCore.Design**

Install Packages using VS Package Manager

- Go to Tools -» NuGet Package Manager -» Manage NuGet Packages For Solution

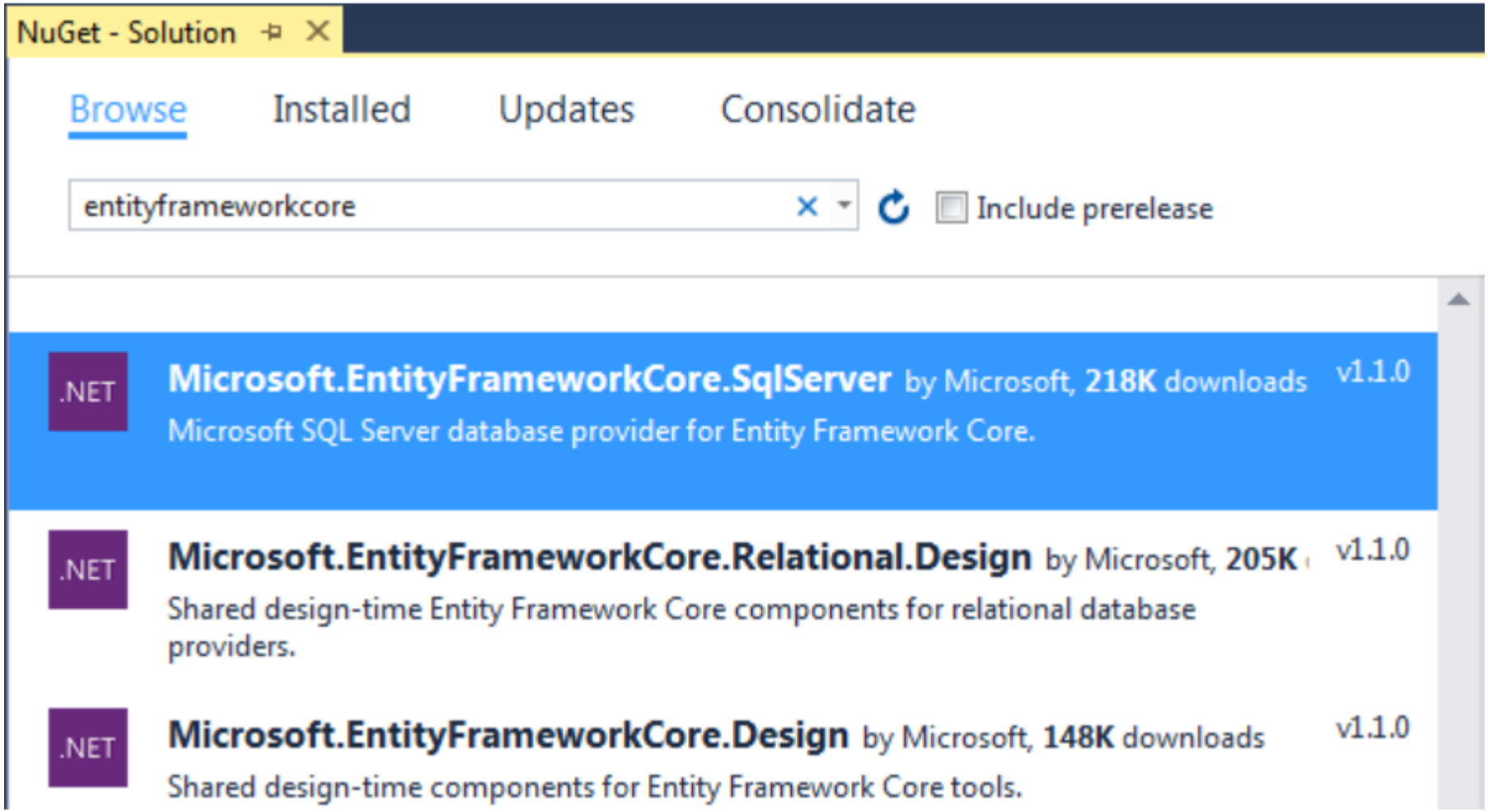


- Ensure that Browse is selected and type "entityframeworkcore" into the search box



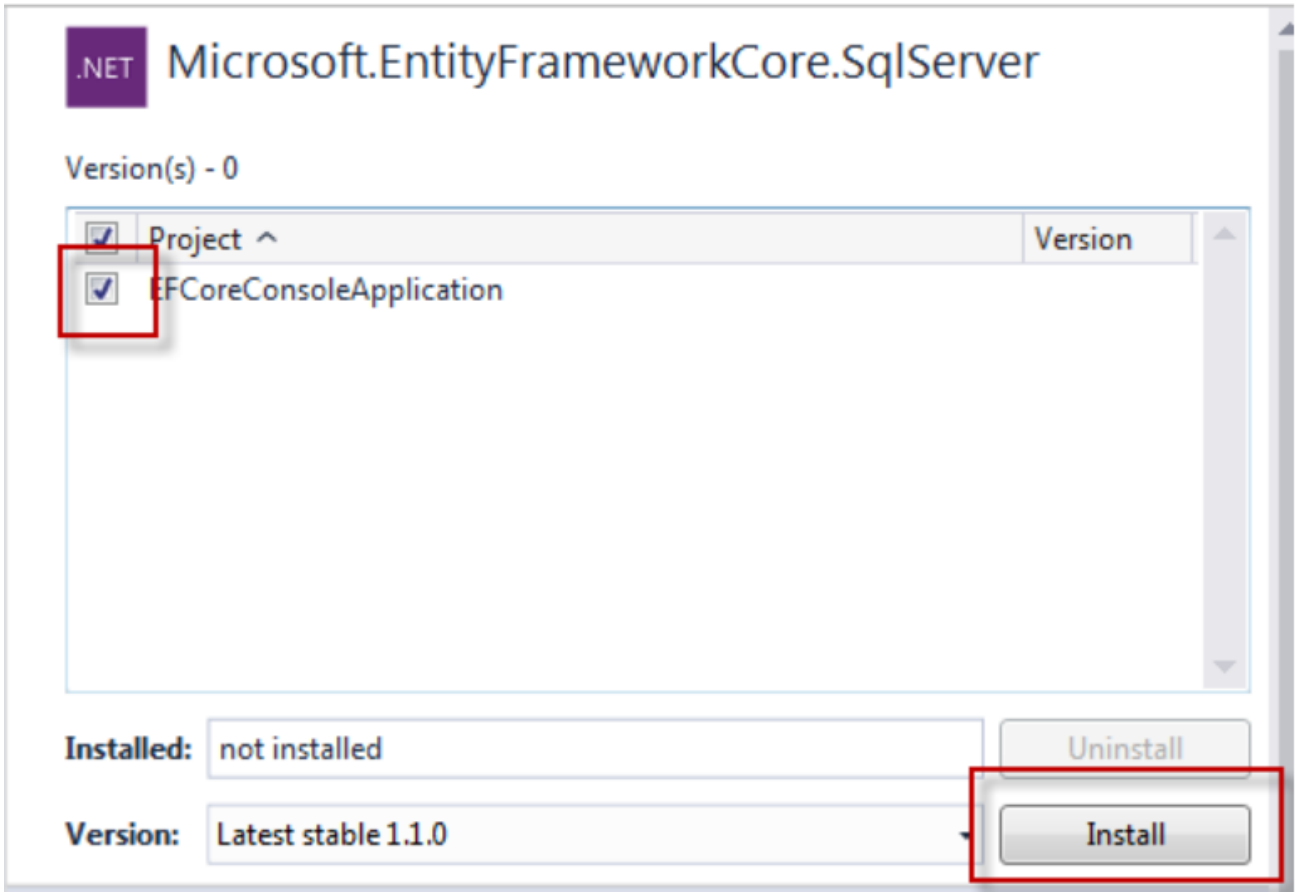
Install Packages using VS Package Manager

- Click on the provider that you want to install. SQL Server is selected in this case.



Install Packages using VS Package Manager

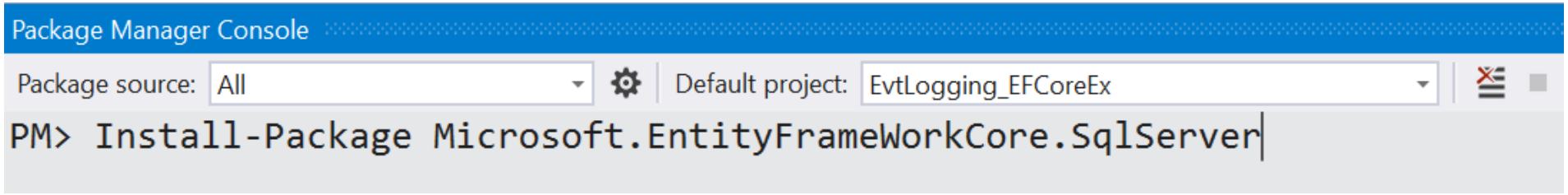
- Check the project that you want to install the package into, then click Install

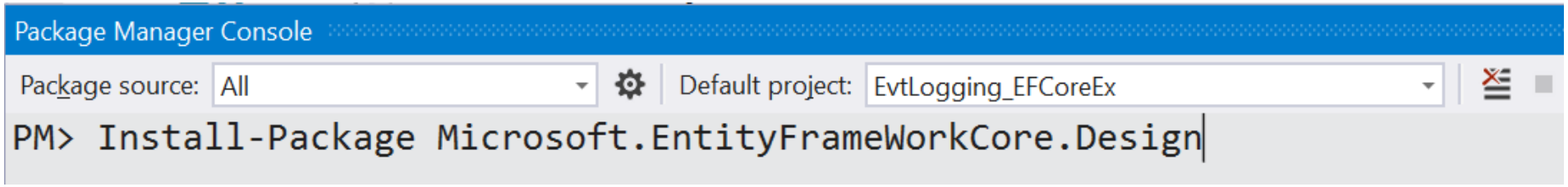


- Repeat above steps to install rest of the EF Core Packages

Install EF Core - Package Manager Console

- Go to Tools -» NuGet Package Manager -» Package Manager Console
- Install below packages using **"Install-Package"** Command





Overview : Database First Approach

- While developing an application, if the database is already available, then Model classes can be created using the Database
- The advantage of this approach is that the application knows that the database is production ready and no changes (or modifications) are required in the database design (Table design).
- This scenario is more useful when the application is developed from scratch except the database.

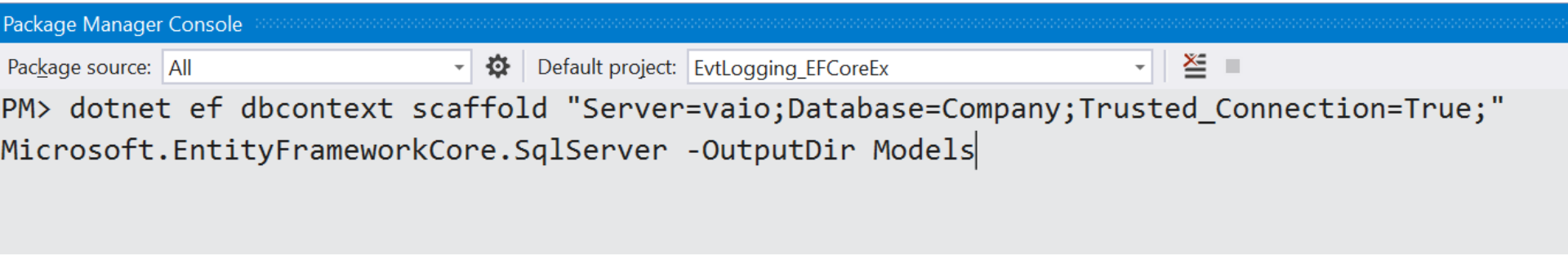
Overview : Database First Approach

Steps for Creating DB First Approach:

- Create a .NET Core Console application
- Install EFCore Packages using Mangage NuGet Packages for Solution or PM
- Generate Model classes from the database
- Perform Database Read/Write operations

Command to Create Model Classes from Database:

dotnet ef dbcontext scaffold << Options >>



Demo

- Demo of Implementing Database First Development approach in EF Core Console Application.

Overview : Code First Approach

- In the code-first approach, EF Core API creates the database and tables using migration based on the conventions and configuration provided in your domain classes.
- In an enterprise application, it is important to design the application model. An accurate application model makes the application development easy.
- The maintainability of the application is primarily dependent on the model design.
- In EF Core, the model is based on the design of the **Entity Classes**.
- These classes contain necessary properties based on which the application can decide what data is accepted through the application and saved in database.
- This approach is useful in Domain Driven Design (DDD).

Overview : Code First Approach

Steps for Creating Code First Approach:

- Create a .NET Core Console application
- Install EFCore Packages using Mangage NuGet Packages for Solution or PM
- Add Entity classes to the project. This will be considered as Models classes (POCO)
- Add DbContext Class and override OnConfiguring() method
- Run Migration Commands from PM Console
- Update the structure changes using “Update-Database” command using PM Console
- Perform Database Read/Write operations

Migration in Code First Approach

- **Migration:** It is used to keep the database synchronized with the domain (entity) classes.
- When developing a project the programmers keep on updating the entity classes. For that, they need to run migrations to keep the database schema up to date.
- Migrations are enabled by default in EF Core.
- If you have Visual Studio, you can use the **Package Manager Console (PMC)** or **CLI** to manage migrations.

On your Tools > NuGet Package Manager > Package Manager Console execute any of the following 2 commands to create the migration.

```
PM> add-migration Migration1
```

Or DotNet CLI Command:

```
PM> dotnet ef migrations add Migration1
```

Update Database in Code First Approach

- Update Command will do the changes based on the migration created by the **Add migration** command.

Syntax: Update-Database <<Migration Name>>

- You can execute either of the 2 update migration command given below

```
PM> Update-Database
```

CLI version:

```
PM> dotnet ef database update
```

Overview : Code First Approach

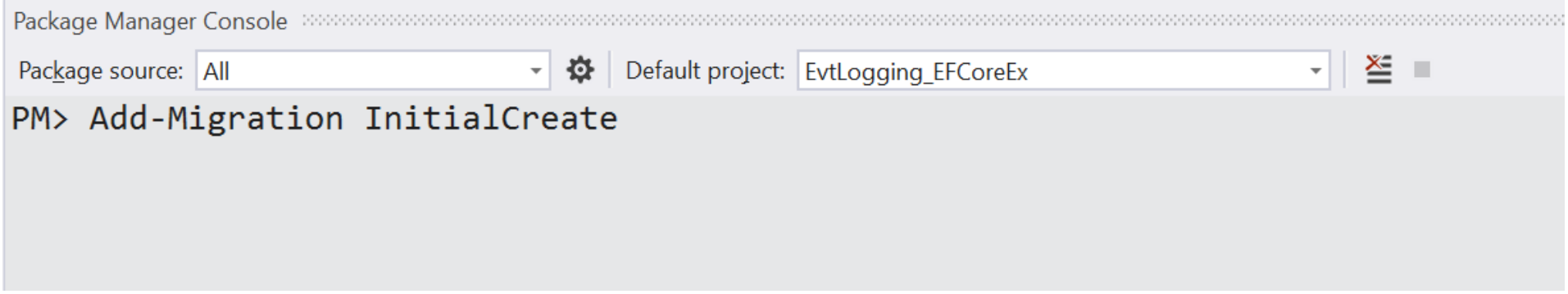
Override OnConfigure Method in DbContext Class:

```
using Microsoft.EntityFrameworkCore;
namespace EFCoreDemo
{
    public class EFCoreDemoContext : DbContext
    {
        public DbSet<Book> Books { get; set; }
        public DbSet<Author> Authors { get; set; }
        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            optionsBuilder.UseSqlServer(@"Server=.\;Database=EFCoreDemo;Trusted_Connection=Tru
e;MultipleActiveResultSets=true");
        }
    }
}
```

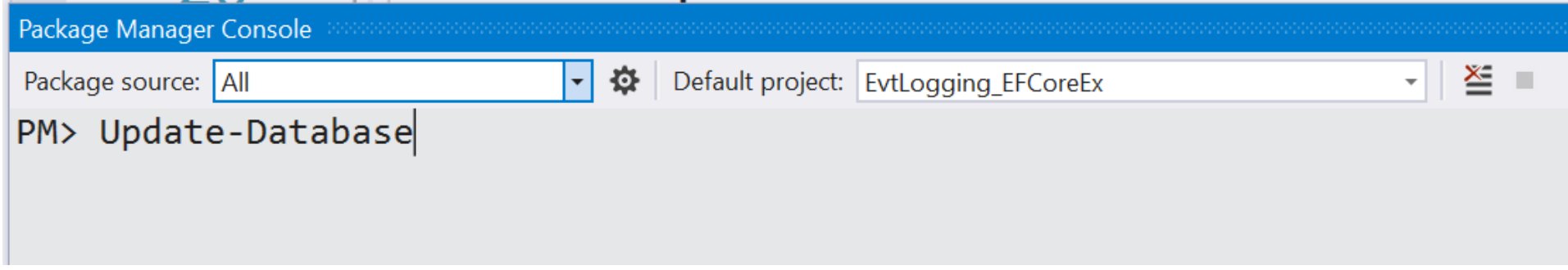
Overview : Code First Approach

Adding Migration:

- Migrations are used to keep the database schema in sync with the model.
- There will be situation like we will not have any database at the moment, so the first migration will create it and add tables for the entities represented by the DbSet properties on the Model classes what we created.



Command to Update Database



Demo

- Demo of Implementing Code First Development approach in EF Core Console Application.

Lab

- Lab Activities to perform DB First & Code First Approach in EF Core Console Application

Summary

In this lesson we covered the following topics

- Database First Approach
- Code First Approach

Review Question

- Which of the following model allows user to generate classes based on Existing Tables?
  - Design First
  - Code First
  - Model First
  - Database First

- We need to Override OnConfigurating() method is EF Core Code First Approach.
  - True
  - False

Review Question

- _____ allow a developer to create a new Model using the Entity Class
  - Code First Development
  - Designer First Development
  - Model First Development