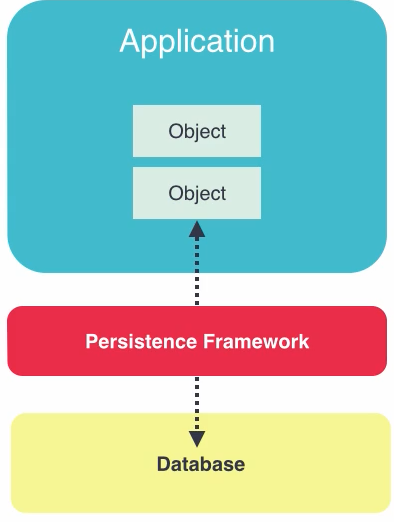
What is entity framework?



Entity framework like Persistence.

Đóng vai trò là frameword trung gian để chuyển dữ liệu database về phía application

Have Workflows

. Database first:

1. Design tables
2. EF generates domain classes

. Code first:

1. Code and create domain classes
2. EF generates database tables

. Model first

1. Create a UML diagram
2. EF generates domain classes and database

Demo Database First:

Install SQL server first

Problem install:

<https://social.msdn.microsoft.com/Forums/sqlserver/en-US/2ffd27d1-cbfe-4233-9baf-9dc00d0d39a4/unable-to-install-sql-server-setupexe-vs-shell-installation-has-failed-with-exit-code-1638?forum=sqlsetupandupgrade>

Code first or Database first?

Code first gives you more control over database.

If you have already database should use database first.

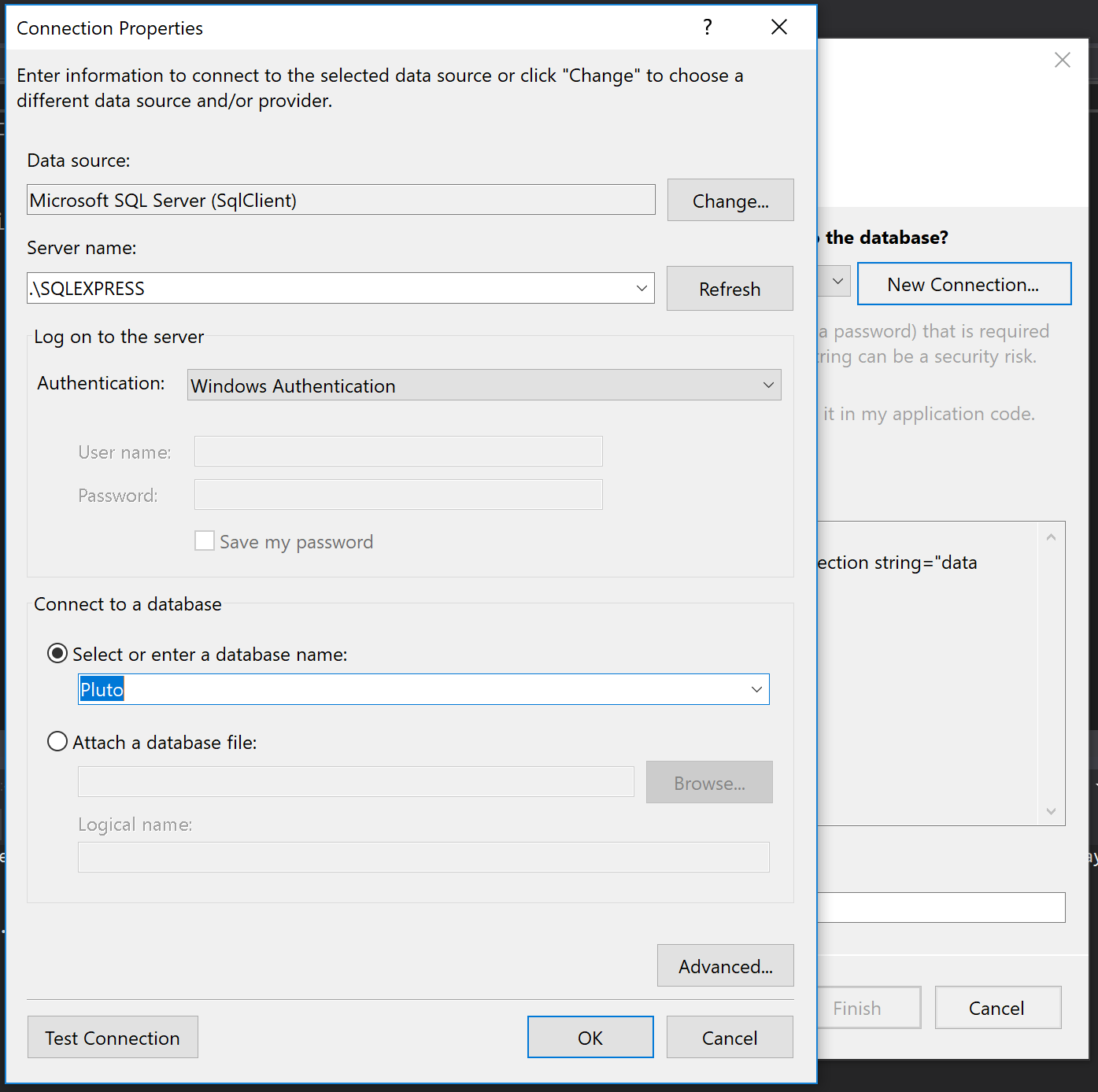
**Code First Advantages**

Full versioning of database. (control over database)

Productivity (faster to write code, no need to design database table)

1. **Demo Database First**

Right click to project, add new item -> ADO.NET Entity Data Model -> Select EF Designer From Database -> next -> click to New Connection ->



->Select Server Name and Select or enter a database name ->Ok

-> Save name PlutoDbContext -> next

->Choose Entity Framework 6.x -> Next

-> Which database objects do you want to include in your model, check to table -> Finish

**Open file edmx to XML file**

Right click on edmx file -> open with -> XML (Text) Editor

**Update Model from Database**

If you add more table from database -> right click to free area choose “Update Model From Database” check on table from add and finish.

**Add Column, delete column, and change type from database table**

Add column from database table -> update from vs -> OK

Delete column from database table -> update from vs -> problem, must be delete from vs

Change type from database table -> problem, didn’t change

1. **install-package EntityFramework –Version:6.1.3**
2. **Create database model**

Create class is table and property is column,

Property with type is object class is foreign key, relationship between 2 column

IList<ObjectClass1> name in table and IList<ObjectClass2> name in another table is define relationship many to many between 2 table

1. Add tag <connectionString></conecctionString>

<connectionStrings>

<add name="DefaultConnection" connectionString="data source=.\SQLEXPRESS; initial catalog=PlutoCodeFirst; integrated security=SSPI;" providerName="System.Data.SqlClient"/>

</connectionStrings>

integrated security = SSPI -> mean window authentication

initial catalog=PlutoCodeFirst -> mean name of database

data source=.\SQLEXPRESS -> connect to server of SQL

name="DefaultConnection" -> this name will be use to constructor of class inherit

DbContext

public PlutoContext() : base("name=DefaultConnection") {}

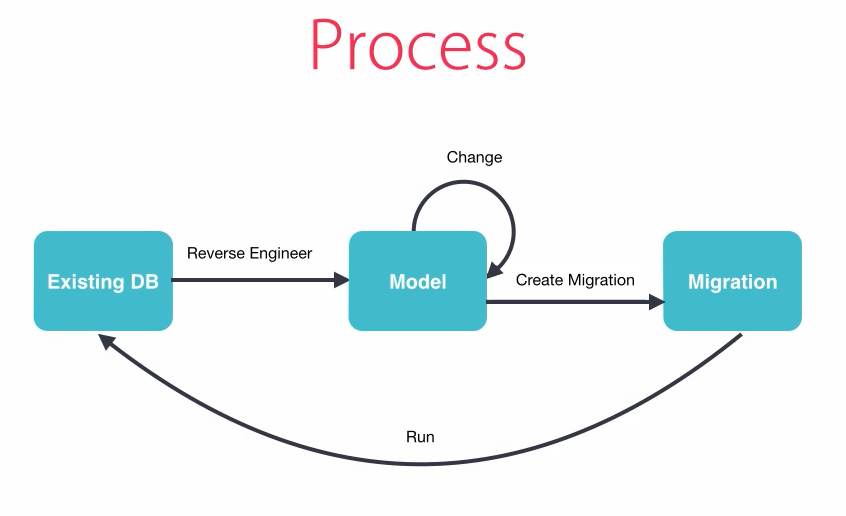
1. Go to Package Manager -> enable-migrations
2. Go to Package Manager -> add-migration <Name of Migration> - VD: InitialModel

You will see we have the Migration folder with 2 class is Configuration.cs and 201800\_NameMigration.cs

1. Go to Package Manager -> update-database

This will create database in SQL Server

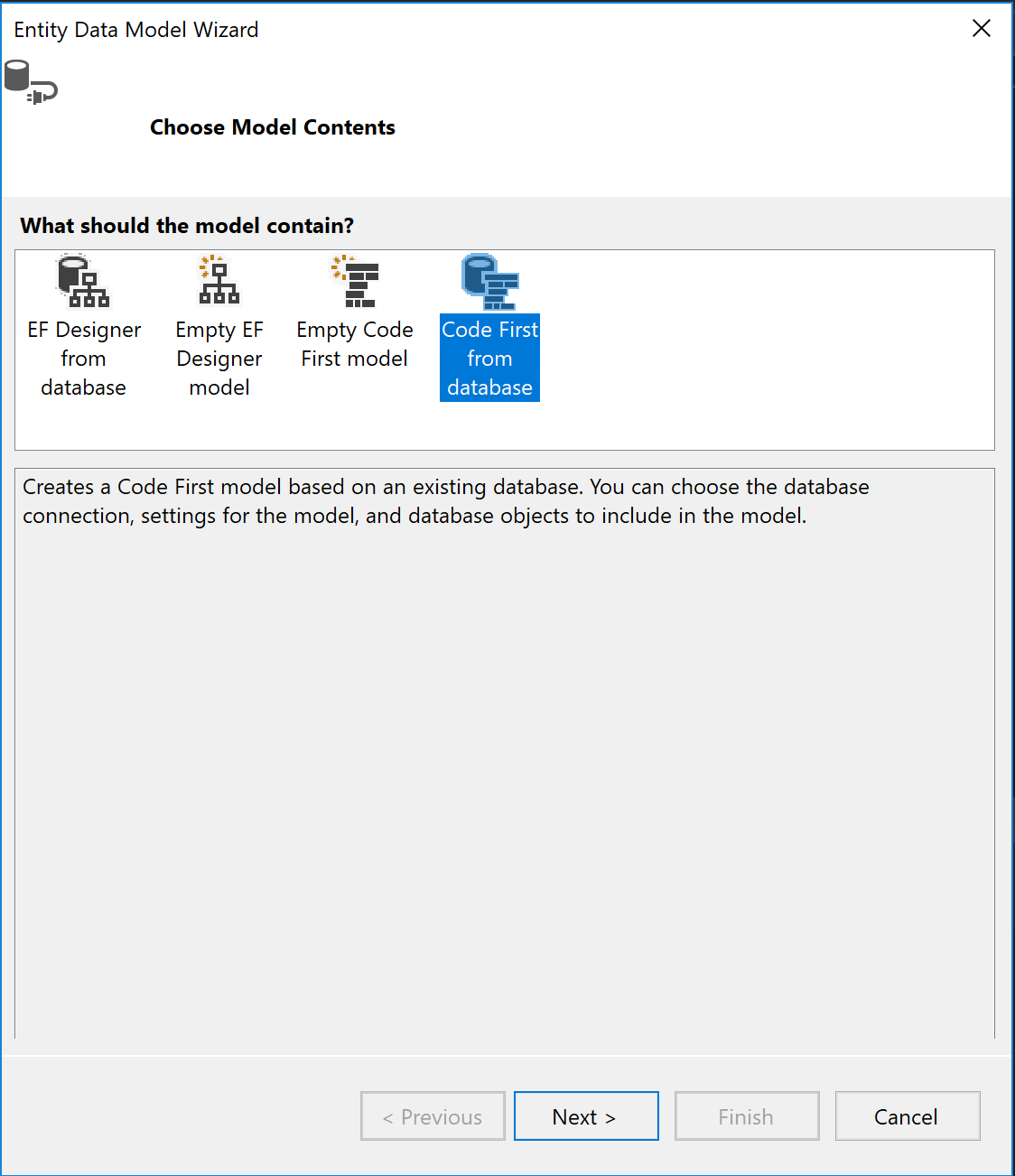
**CODE FIRST WITH EXISTING DATABASE**

****

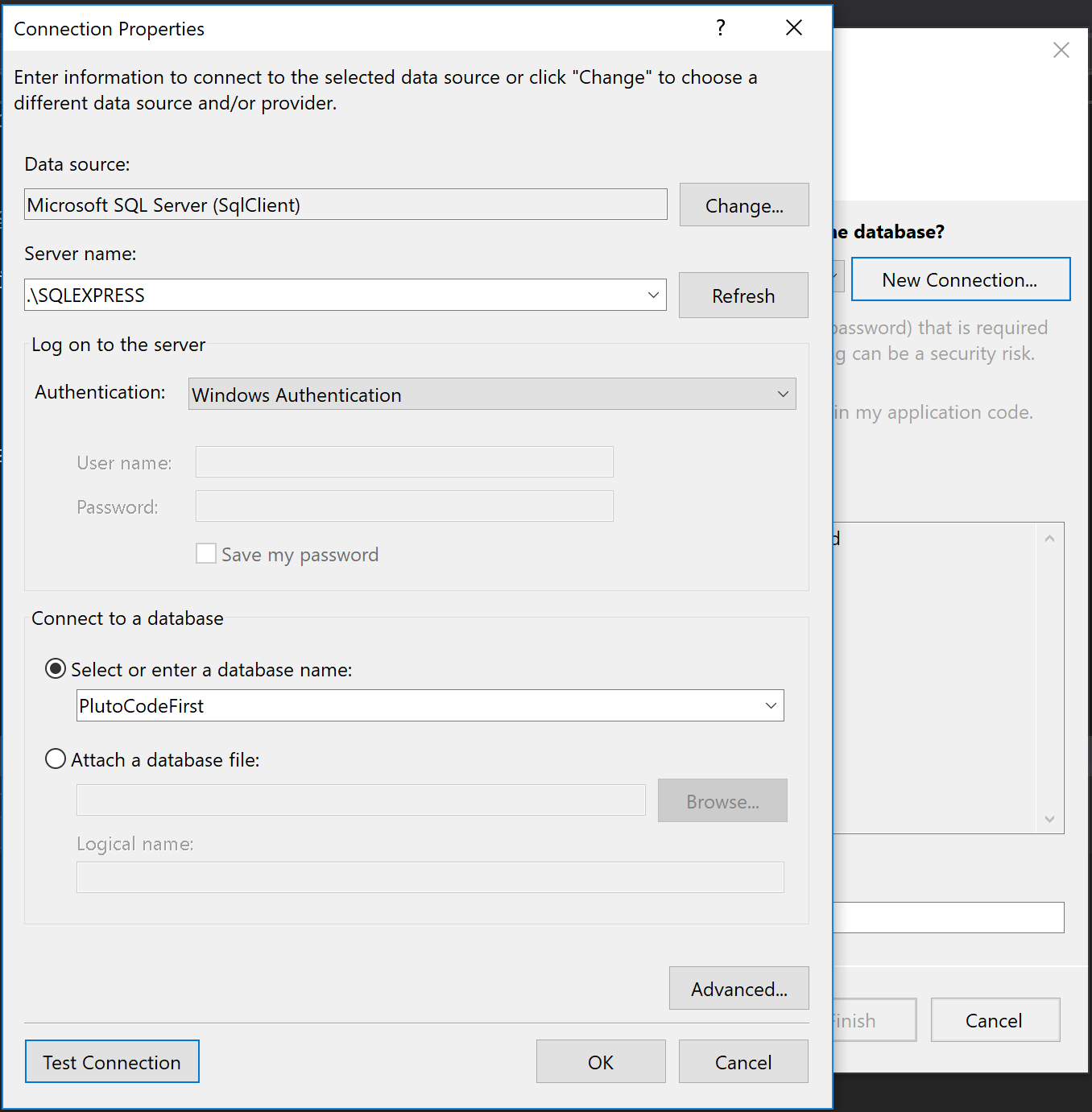
**Benefits.**

* Full versioning of the database ( take all control with sql ( ) )
* Less time wasted in designers database, table

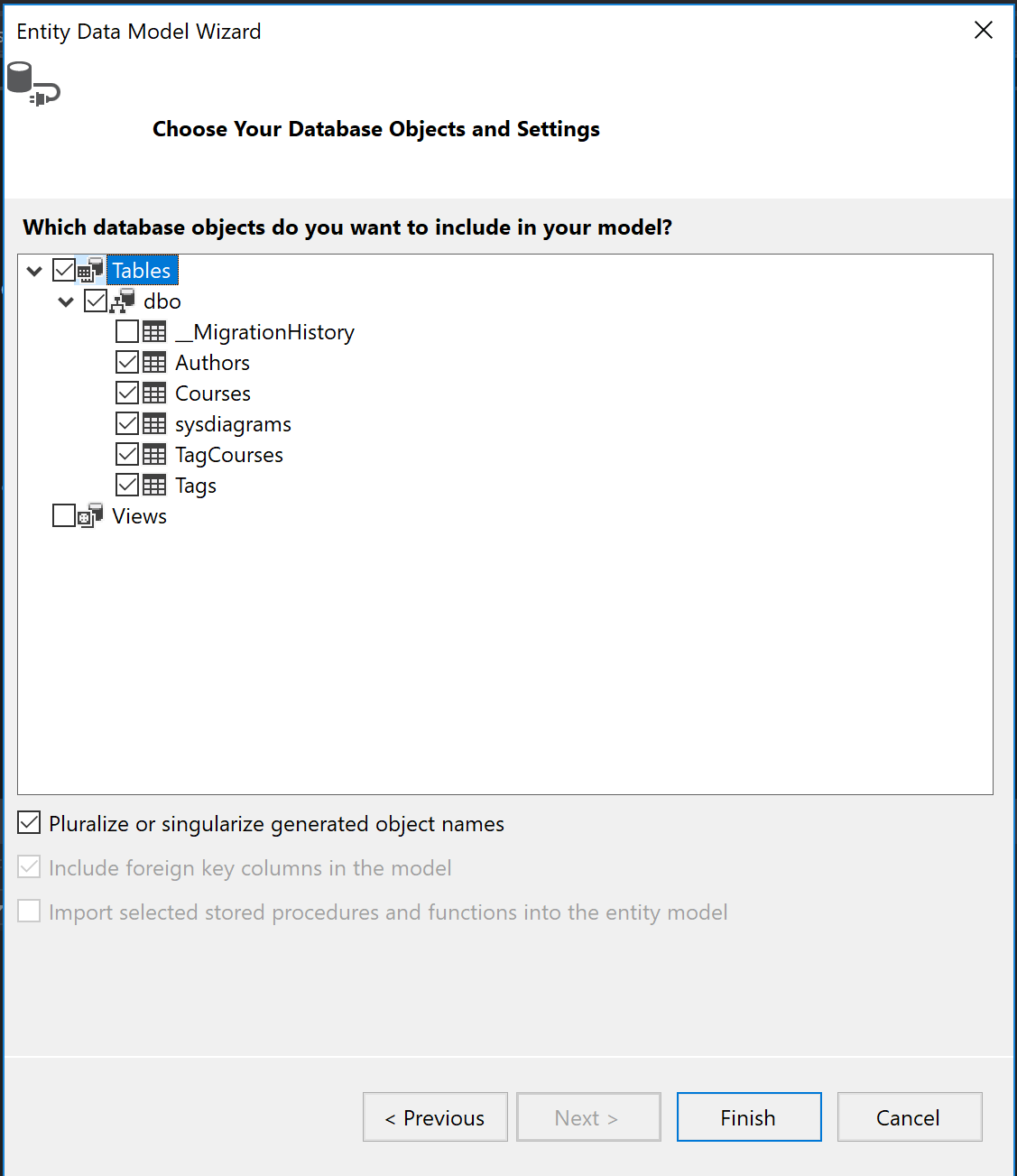
1. Add new item ADO.NET Entity Data Model name PlutoContext ( Pluto is database name, Context is from DbContext )











1. **Migrations**

Goto Package Manager Console -> enable-migrations

PM -> add-migration InitialModel ( which this way you will see the method up and down to create database but cannot change to database so use console below )

PM -> add-migration InitialModel -IgnoreChanges –Force ( which this way you see the method up and down with blank space )

PM -> updata-database

**Note:**

* Always work on small changes and small migrations.

1. **And New Class**

Create new class name Category with 2 prop Id and Name

PM -> add-migration AddCategoriesTable ( Method Up and Down is blank)

* Go to add PlutoContext and add DbSet for new Class and override migration

PM -> add-migration AddCategoriesTable -Force

( -Force mean : override )

* Inside migration can use Sql( “sql query” )

1. **Modification**

Adding a new property

Modification an existing property

Deleting an existing property

When you rename the column must be use Sql ( ) to update set all the value of old column to new column.

public partial class RenameTitleToNameInCoursesTable : DbMigration

{

public override void Up()

{

//this way will delete the column so much be add Sql query

AddColumn("dbo.Courses", "Name", c => c.String(nullable: false));

Sql("UPDATE Courses SET Name = Title");

DropColumn("dbo.Courses", "Title");

//this way will rename the column - option 1

//RenameColumn("dbo.Courses", "Title","Name");

}

//Down method is opposite with Up method

public override void Down()

{

AddColumn("dbo.Courses", "Title", c => c.String(nullable: false));

Sql("UPDATE Courses SET Title = Name");

DropColumn("dbo.Courses", "Name");

}

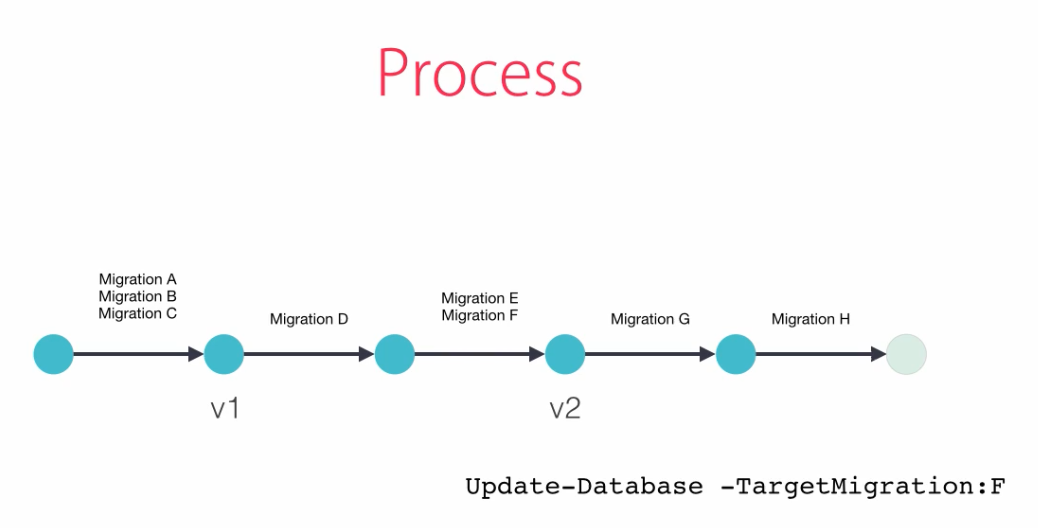
}

1. **Migrations Downgrading a Database**

Option 1: using a blank database

* Checkout the older version
* Change the db name in the connection string
* Update-Database

Option 2: Using an existing Database



When you use Update-Database –TargetMigration:F

After that you must be update-database

And fix some thing (add-migration, …)

Finished update-database

**Seeding Database**

Use in Configuration

public Configuration()

{

AutomaticMigrationsEnabled = false; //No need to do manual migration in PackageManager like Code First

//We use code first so when we finished change model we must be manual migration by use the comment in PackageManager

}

protected override void Seed(CodeFirstExistingDatabase.PlutoContext context)

{

// This method will be called after migrating to the latest version.

// You can use the DbSet<T>.AddOrUpdate() helper extension method

// to avoid creating duplicate seed data.

context.Authors.AddOrUpdate(a => a.Name, new Author

{

Name = "Author 1",

Courses = new List<Course>()

{

new Course(){Name = "Course for Author 1", Description = "Des 1"}

}

});

}

Use method Seed ( ) when you want to add data to database in the first time like the list of Catagories , list of city,… So use