**Agenda:**

* Model Creation Methods
* Model First
* Code First
* Database First
* DbContext
* Lifetime
* Connections
* Database

Process means creating a recorder.

**What Model**

What Model to Use

* Model First (will change the schema)
* Create database from model (use designer)
* Classes auto-generate from model with existing db
* Code First
* Define class and mappings
* Reverse engineer migrations for existing database
* Create database from model
* Use Migrations to evolve new database
* Database Frist (company prefer it)
* Reverse engineer model in designer

**Model First**

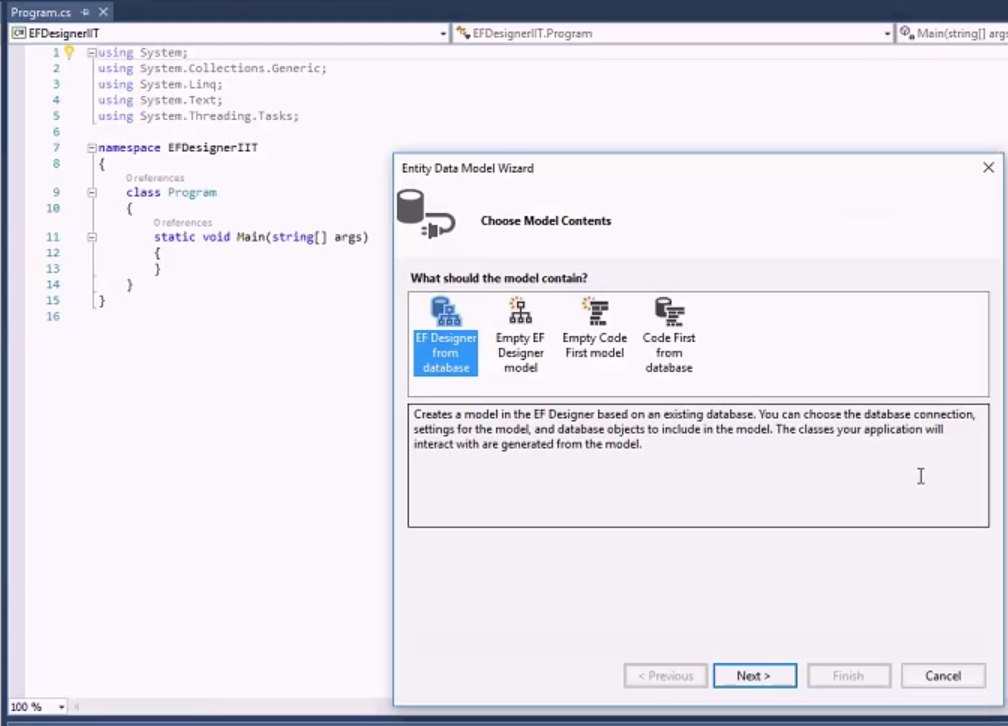
Deeper Dive

* Model First
* Add new ADO.NET Entity Data Model in project
* Empty Model
* Add entities, associations, etc.
* Generate Database from Model

**Demo:**

Create a console project:

Using Entity Data Model



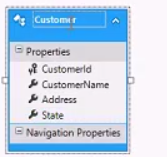
EF Designer from database: use exist database and you will not change it

Empty EF Designer model: you should designer your model

Empty Code First model: totally control

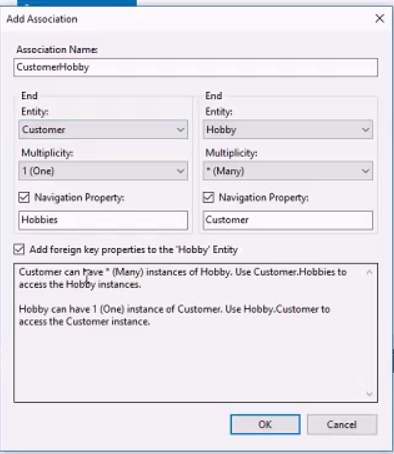
In this case, we choose Empty EF Designer model.

Scalar property: tuber in database

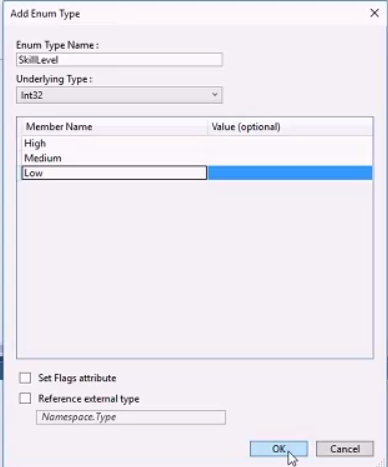


Build up relations:

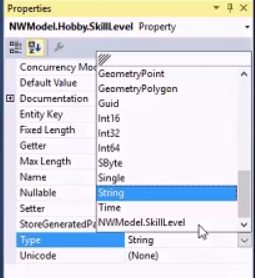
Add new associates:



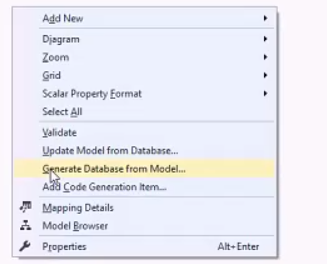
Add Enum Type:



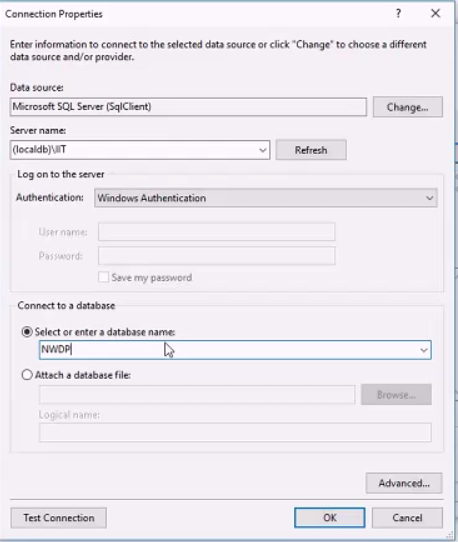
And change the type

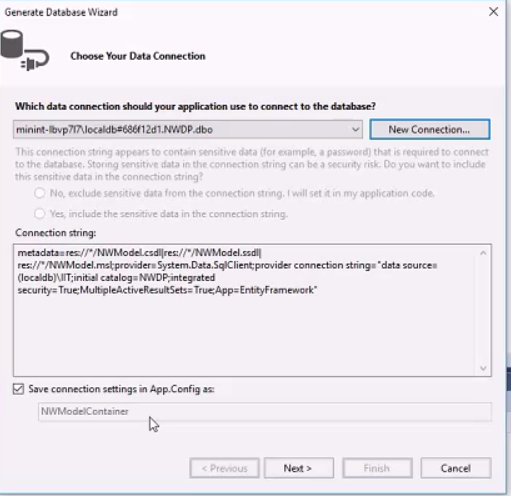


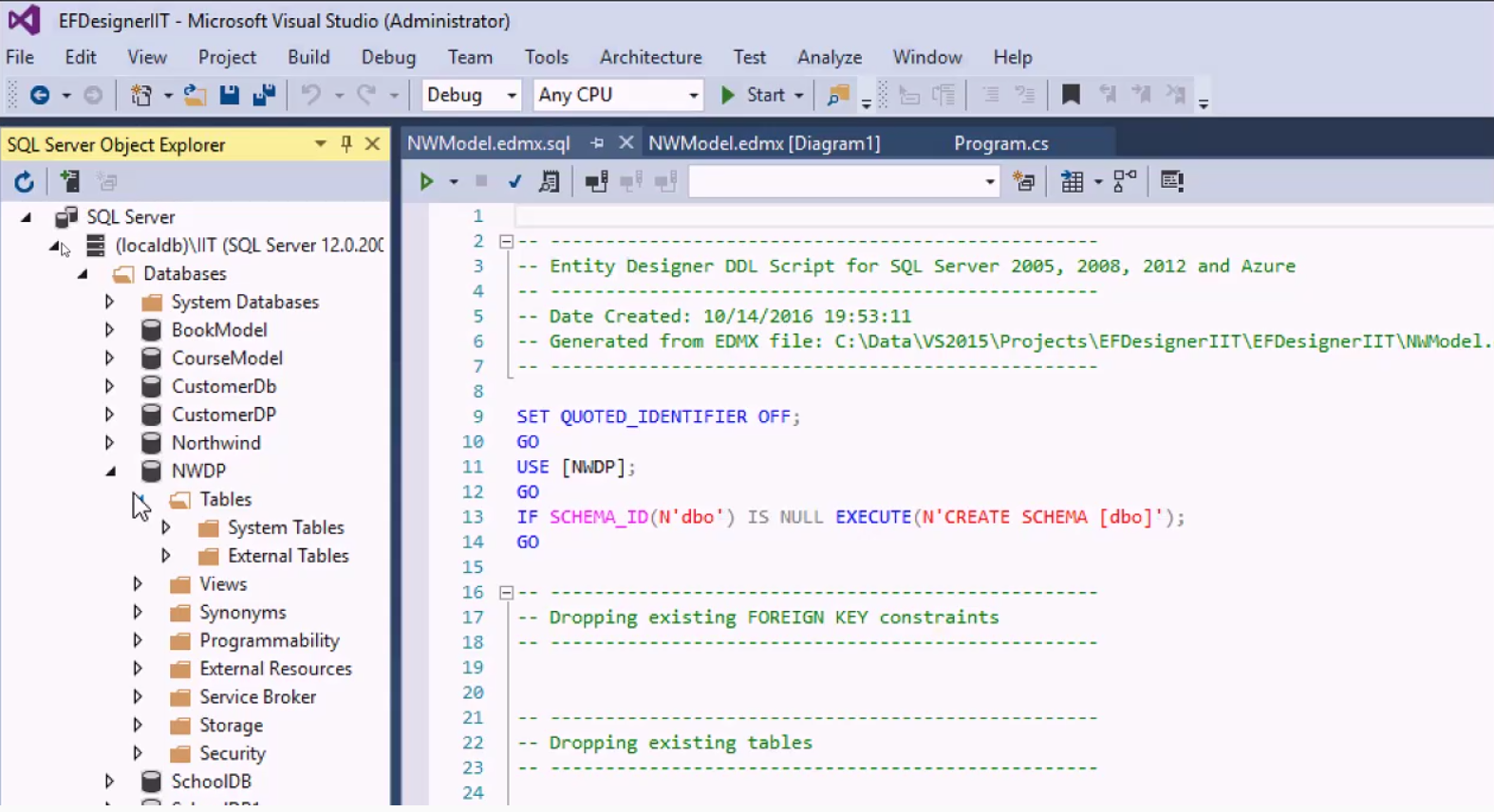
Generate database from model



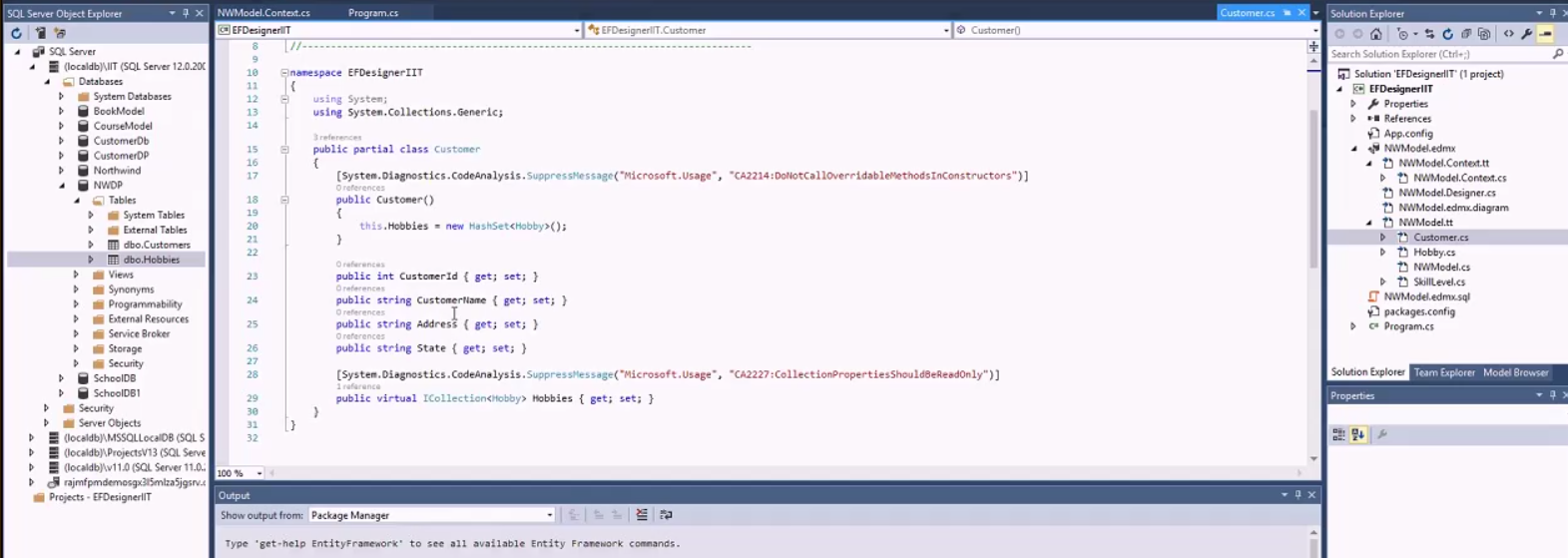
Create a new connection:



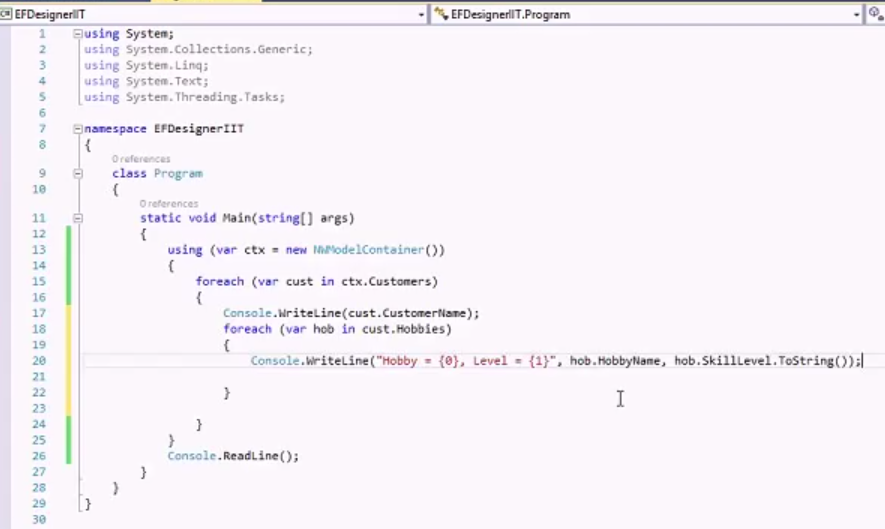


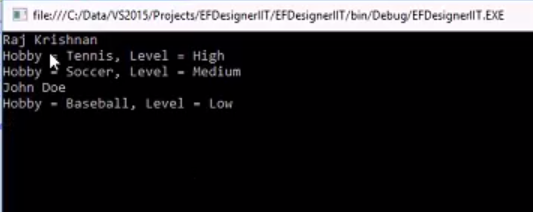


Connect with database



Program.cs





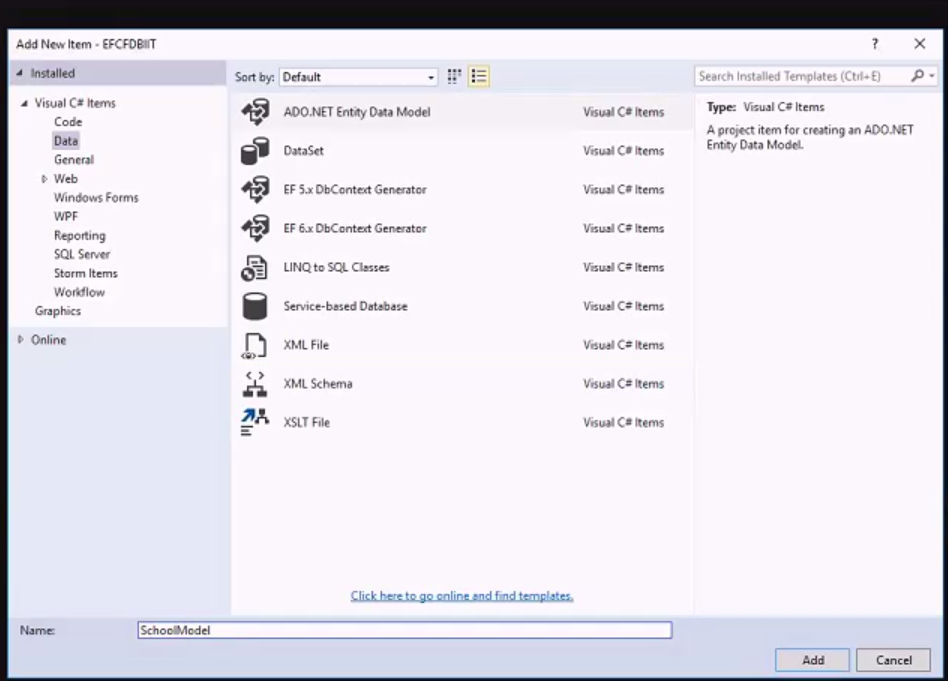
**Code First with Existing Database**

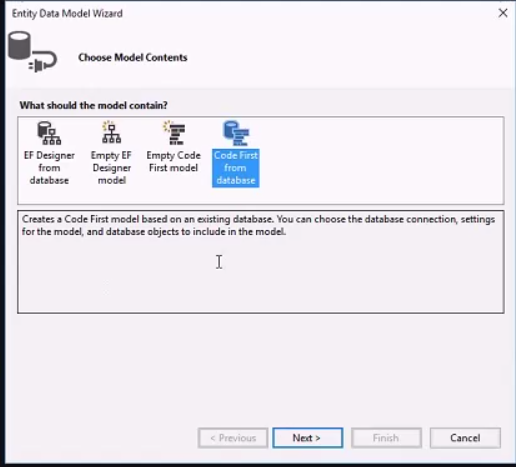
Deeper Dive

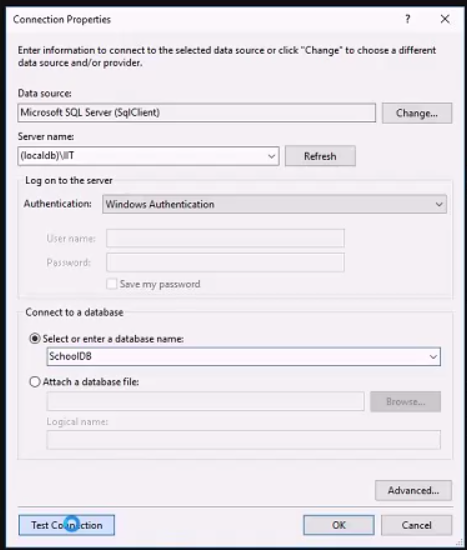
* Code First / Existing Database
* Tools introduced in EF6.1

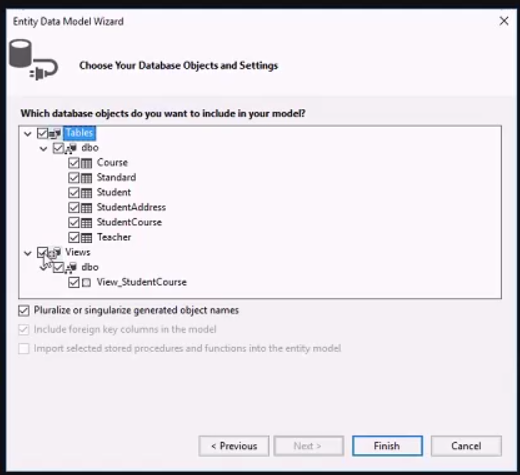
### Demo: Using Code first from DB

Create a console project:

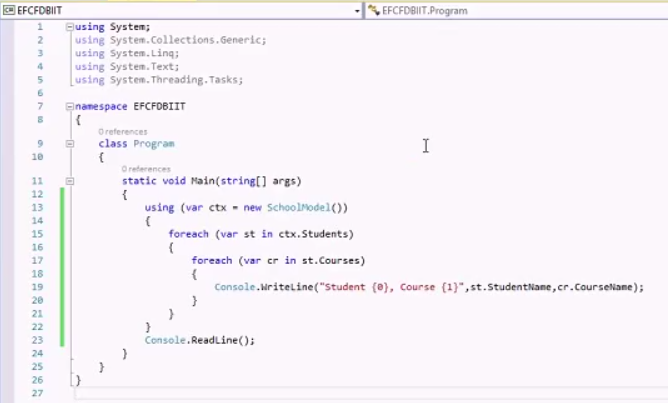




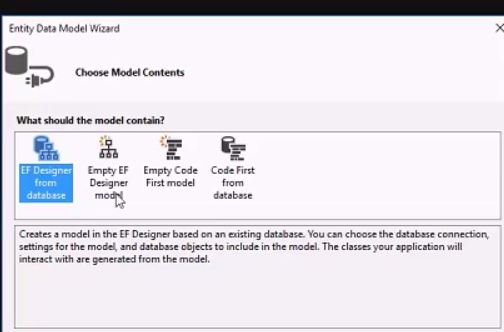




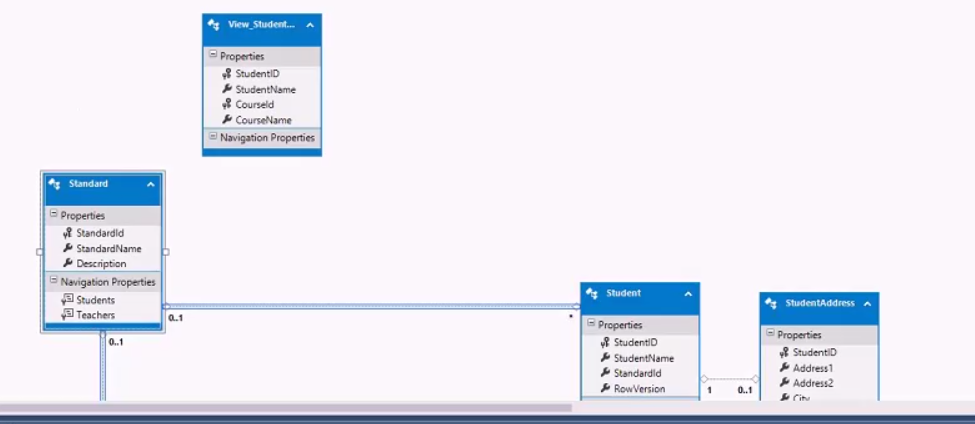
Program.cs

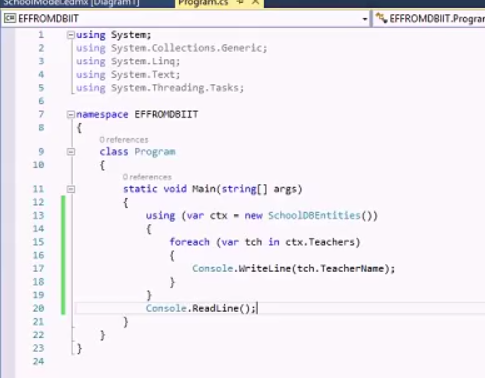


### Another new project: EF designer from DB

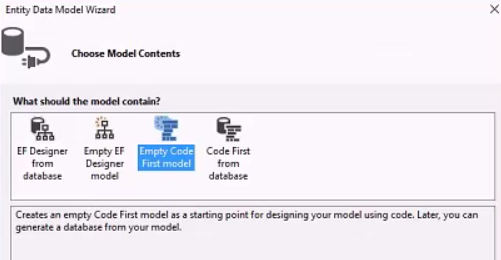


This time you have diagram. Mapper convert

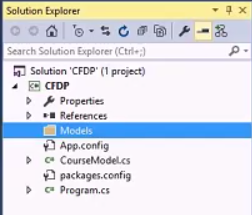




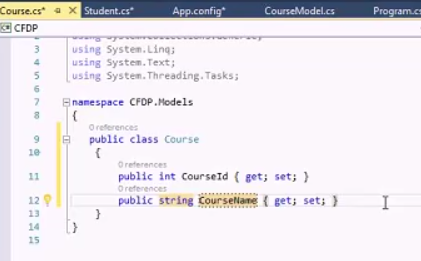
Another Demo: pure code first without database



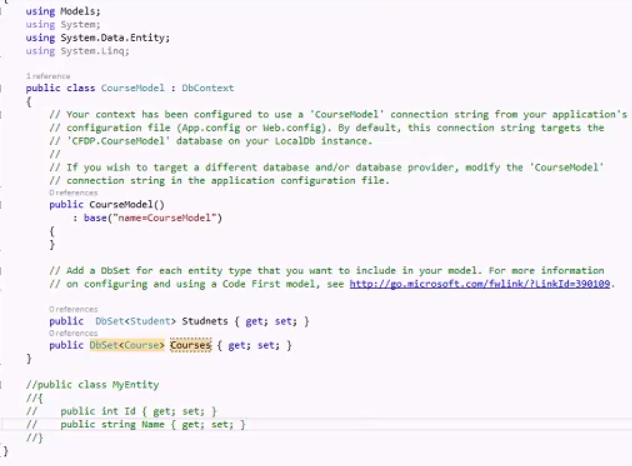
Empty, create a folder of models



Create classes, student.cs, course.cs

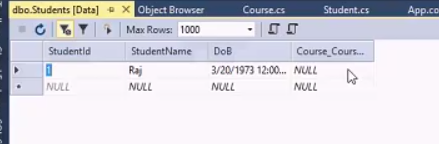


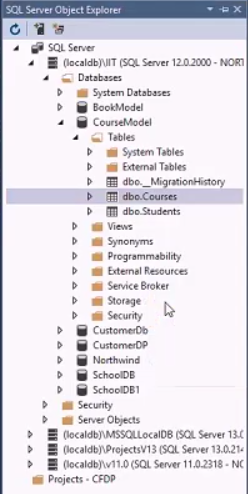
Add them to CourseModel.cs



Program.cs







**Database First**

Deeper Dive

* Database First
* Straight forward
* Add ADO.NET Entity Model to project
* Add tables, etc.
* Automatically creates a model
* TT files for code generation

**DbContext Overview**

* About the DBContext
* Primary class to interact with data objects
* Manages Entity objects (CLR objects)
* Populates objects from data repository
* Change tracking
* Persisting data to data repository
* EF Designer will auto-generate (Code-First need to write yourself)

**DbContext**

Defining DbContext

* Contains DbSet property
* DbSet represents collections of entities

Public class SchoolContext: DbContext

{

Public DbSet<Course> Course {get; set;}

Public DbSet<Person> People {get; set;}

}

**DbContext Lifetime**

Lifetime

* For web application – context per request
* For WPF – context per form
* For long-running context
* Load more objects, the memory consumption of the context may increase rapidly
* Dispose context
* Unable to recover by an exception, may terminate application
* Concurrency-related issues growth
* Use **using** or **finally** to dispose object

**DbContext Connections**

Database Connection DbContext

* Parameterless
* DbContext will run in Code First Mode
* Database name is <Namespace>.<Derived Context Class>
* Database – SQL Express(Default) or LocalDb
* Explicit Connection String

**Parameterless Connection**

Database Connection

* Parameterless
* App.config/Web.config
* Will use config connection string only if
* Derived context is the same name as connection string name
* With or without Namespace included

<configuration>

<connecitonStrings>

<add name=”SchoolCompactDatabase”

providerName=”System.Data.SqlSeverCe.4.0”

connectionString=“Data Source=School.saf”/>

</ connecitonStrings >

</configuration>

**Config file Connection**

Database Connection

* Connection String name parameter
* Code First will create a connection string parameter value
* Or look in App.config/Web.config
* Explicit with “name=<connection string name>”

Public class SchoolContext:DbContext

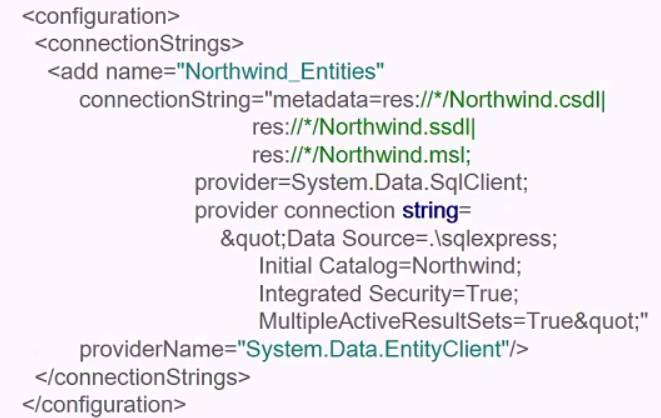
{

Public SchoolContext()

: base(“name=SchoolCompactDatabase”)

}

* Database/Mode First
* EF Designer generates in App.config/Web.config



**DbContext & Database**

Working with the Database

* Ctx.Database
* Create
* CreateIfNotExists
* Exists
* Deletes