DBContext and Relationship in Models (Entity Framework )

DBcontext is bridge between database and Model . It provide session of connection between both ends.

Public class AppDbContext: DbContext // to create any DBcontext in our application need to inherit from DBContext class

{

Public AppDbConext(DbContextOptions<AppDbConext> options):base (options)

{

}

Public DbSet<Book> Books {get;set;} // this line will create a Books Table in DB

}

// Now put connection string in AppSetting.json

// then Call in Configure Services

//In Startup.cs

Var ConnectionString = Configuration.GetConnectionString(“DefaultString”);

Public void ConfgureService(IServiceCollection services)

{

Services.AddDbConext<AppDbConext>(options =>

{

Option.UseSqlServer(ConnectionString); // here use can use SQl/Oracle /DB2 , to do that we need to install Package from NugetPackage Manager

});

}

// Now this will create Books Table in DB

**Relationship and Modeling**

1. One Publisher can publish multiple books , put one book can be published by only one publisher . Then how we design our Model and its relationship to create DB in CF(code first).

[One to Many relationship ]

Instance of Book Class

Publisher

Book3

Book2

Book1

Public class Publisher

{

Public int Id { get;set;}

Public string Name {get;set;}

// Navigation property

Public List<Book> Books {get;set;} // publisher will have multiple Books

}

Public class Book

{

Public int Id {get;set;}

Public string Name {get;set ;}

Public int? PublisherId {get;set;} // this tell EF that Book will have PublisherID as ForeignKey and it can be null “?” , because If someone has written any book and still no publisher is finalised

Public Publisher Publisher {get;set;} // this tell EF that this class have FK reference with Book model(table) and will help to navigate property to EF about FKID

}

// Now Add Migration

Open PM(Package Manager)

1. PM> Add-Migration AddPublisherBook // this line will create a class with DateTime Property in project “<DateTime>\_AddPublisherBook”.cs file
2. Public class AddPublisherBook : Migration

{

// This class will have full details of Model how it will created and its reference , datatype

}

// Add into Database

1. PM> Update-database // This command will create/Update Table Book into DB with PK, FK .
2. Add Id Property is created as Primary Key(Identity column) .

**Many To many RelationShip {Book\_Author}**

|  |  |  |
| --- | --- | --- |
| **ID** | **BookID** | **AuthorID** |
| 1 | **1** | **2** |
| **2** | **1** | **3** |
| 3 | **2** | **3** |
| **4** | **4** | **2** |

**Means 1 physics Book** will have multiple Author ( H.c. verma , R.K Agarwal ) and 1 Author( H.c.verma) can write multiple Books ( Phyics for Beginner, Concept of Physics etc…).

Public class Book

{

Public int Id {get;set;}

Public string Name {get;set ;}

Public int? PublisherId {get;set;} // this tell EF that Book will have PublisherID as ForeignKey and it can be null “?” , because If someone has written any book and still no publisher is finalised

Public Publisher Publisher {get;set;} // this tell EF that this class have FK reference with Book model(table) and will help to navigate property to EF about FKID

Public List<Author> Authors {get;set;}

}

Public class Author

{

Public int Id {get;set;}

Public string Name {get;set ;}

Public List<Book> Books {get;set;}

}

Public class Book\_Author

{

Public int Id{get;set;}

Public int BookId {get;set;}

Public Book Book {get;set;}

Public int AuthorId {get;set;}

Public Author Author {get;set;}

}

// Now in Startup.cs class we have to define the relationship between these class to do that we have to override onModelCreating method.

Protected void override OnModeCreating(ModelBinder modelbinder)

{

// set relationship between book and Book\_Author

modelbinder.Entity<Book\_Author>() // telling EF that while creating Model

. HasOne(b= > b.Book) // will have Book entity

.WithMany(ba => ba.Book\_Author) // which will be at multiple rows

.hasForeignKey(bi => bi.BookId); // and this will have BookID as Foreign key

modelbinder.Entity<Book\_Author>()

.hasOne(a=>a.Author)

.WithMany(ba => ba.Book\_Author)

.HasForeignKey(ai => ai.AuthorId)

}

Public DbSet<Author> Authors {get;set;}

Public DbSet<Book> Books {get;set;}

Public DbSet<Book\_Author> {get;set;}