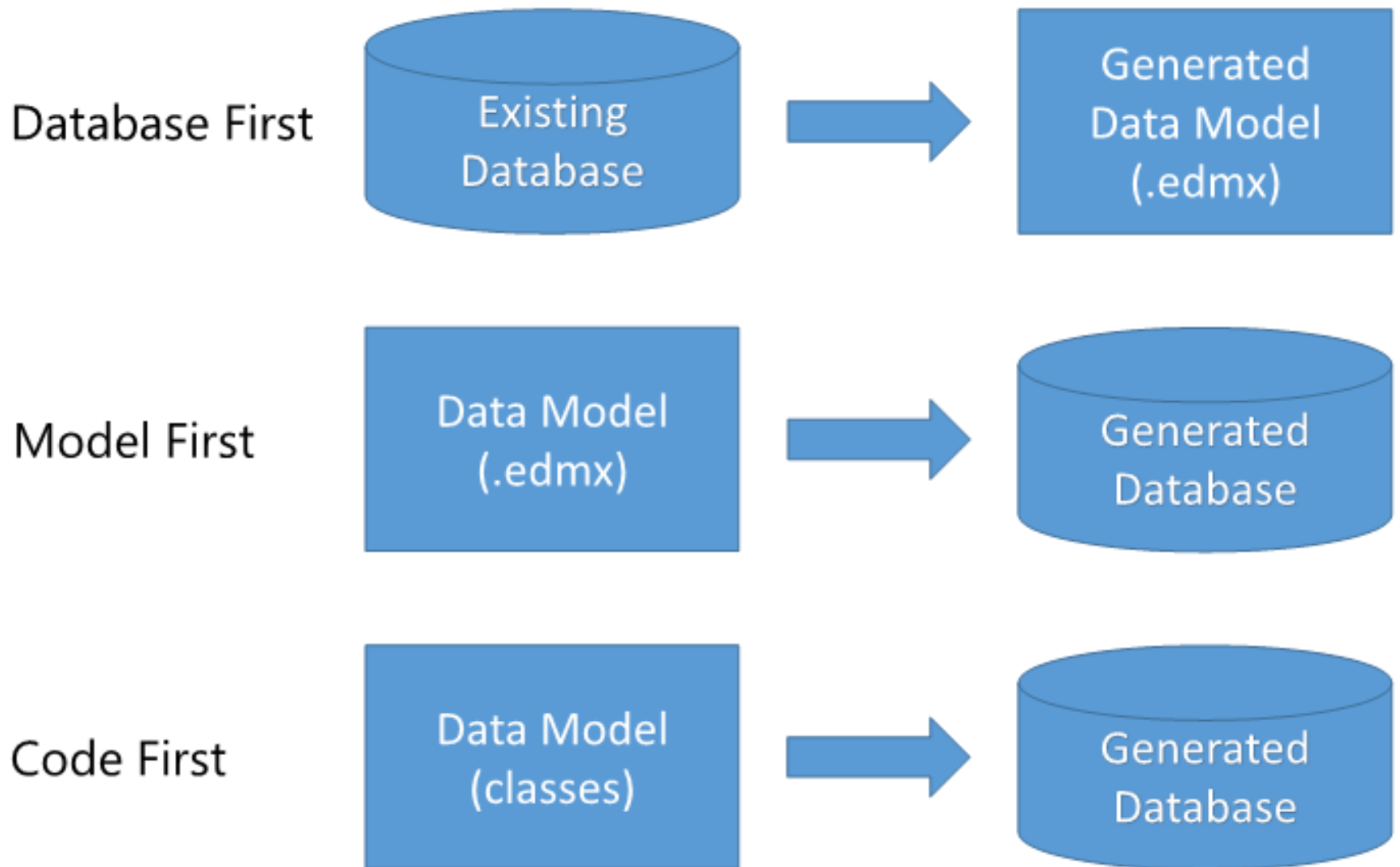


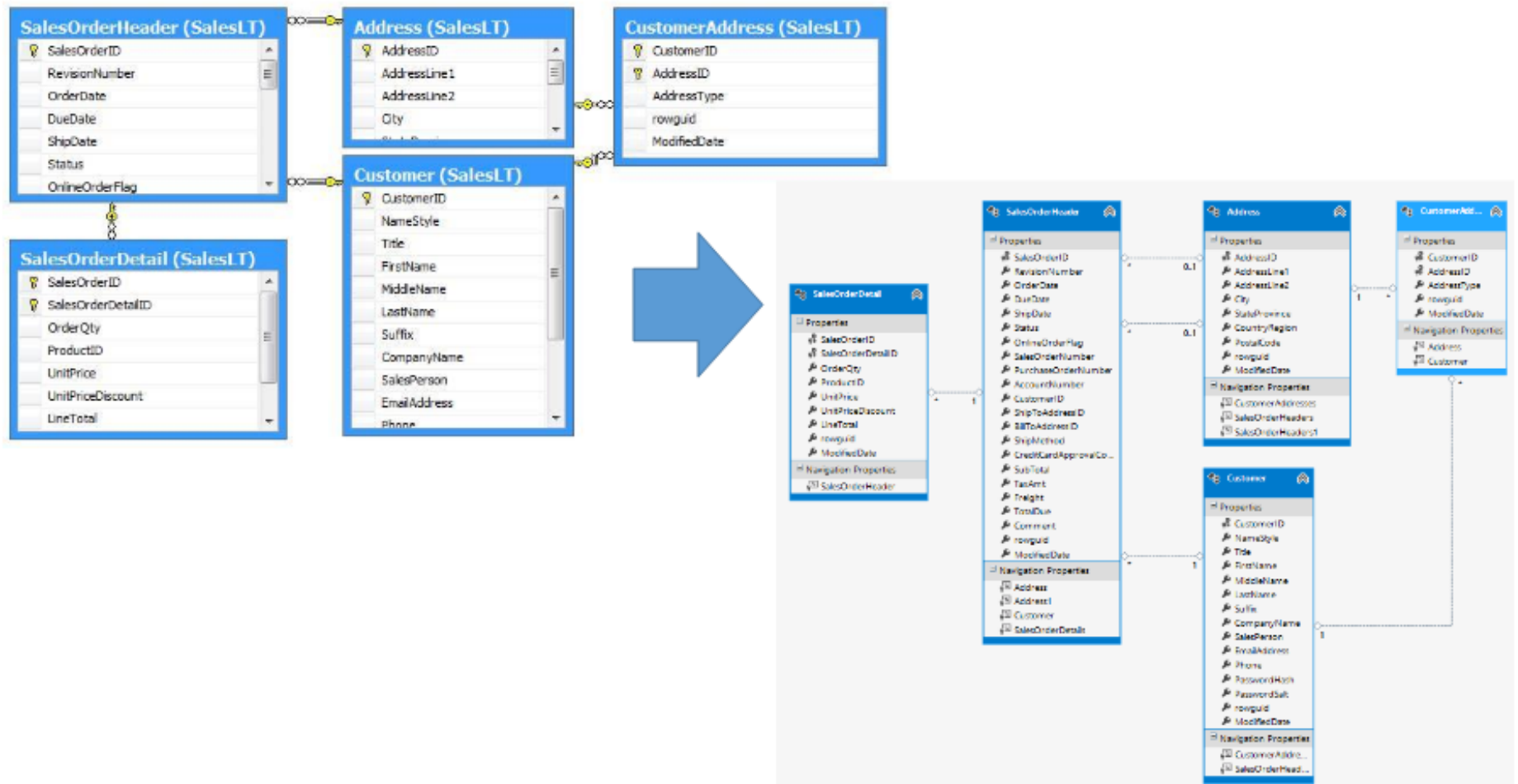
Entity Framework. Code First



Основные способы создания моделей



Database First



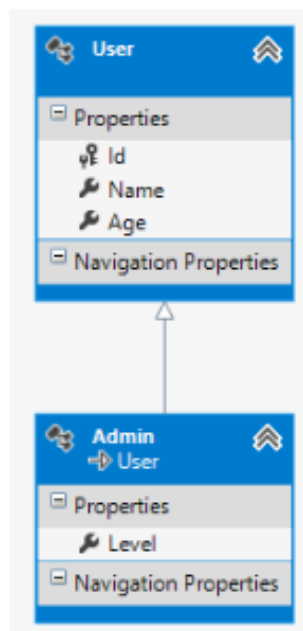
Model First

Model First

Data Model
(.edmx)

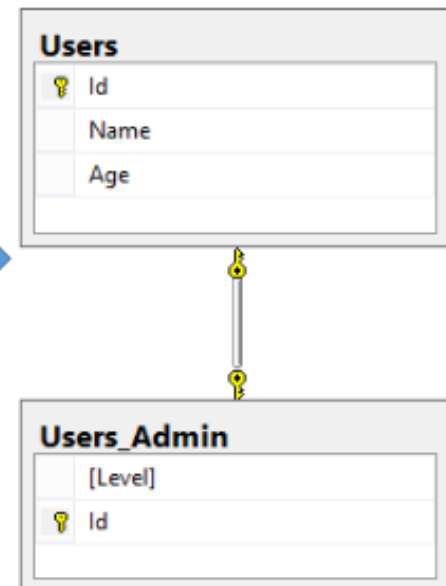


Generated
Database

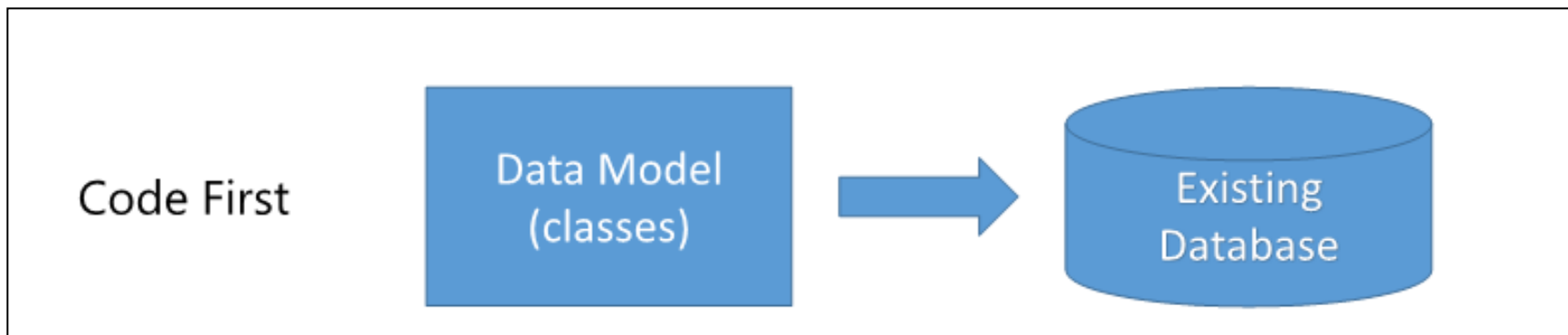


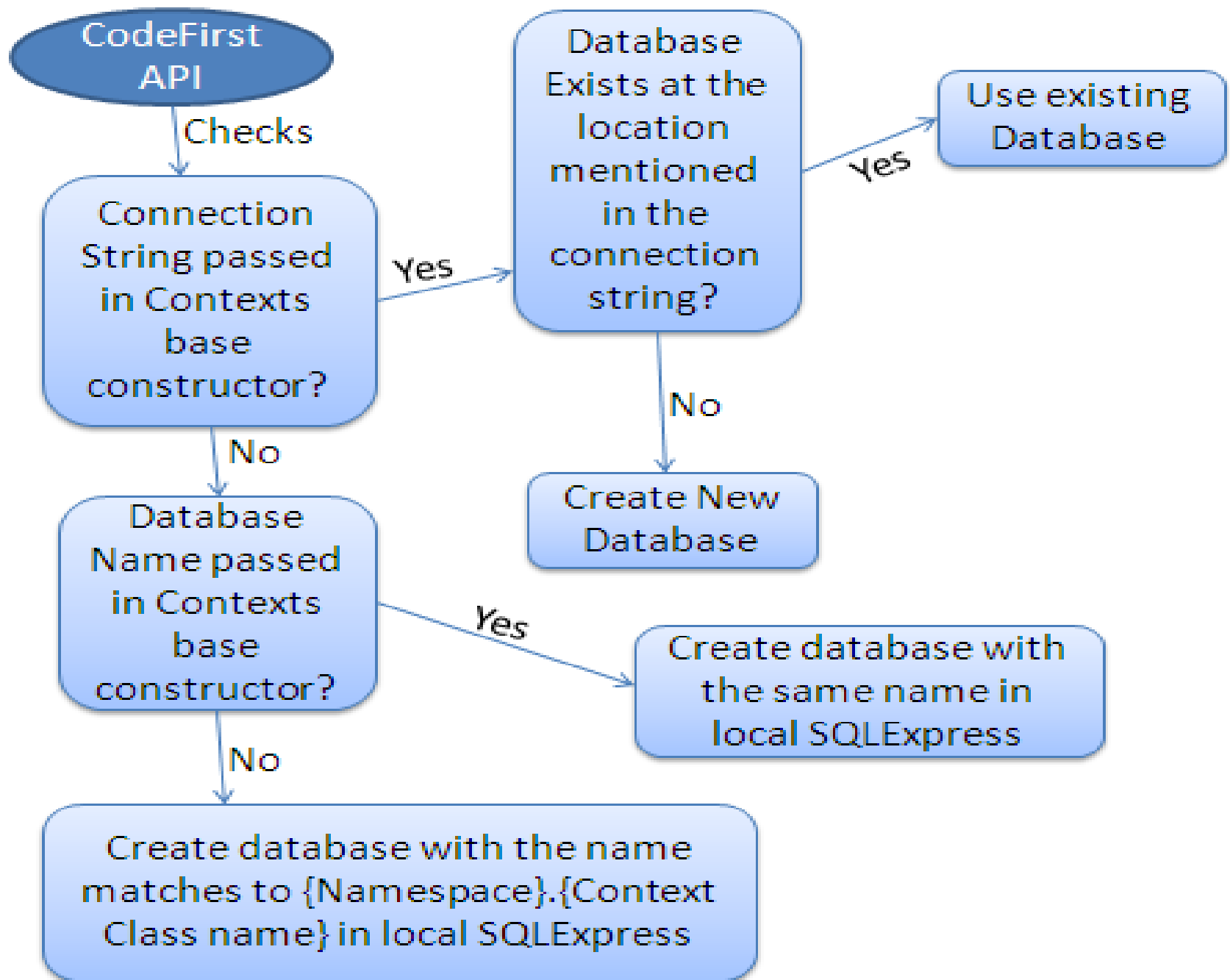
```
public partial class User
{
    public int Id { get; set; }
    public string Name { get; set; }
    public DateTime? Age { get; set; }
}

public partial class Admin : User
{
    public string Level { get; set; }
}
```



CodeFirst – написание кода классов предметной области, при отсутствии модели и БД.
Генерация БД и модели сущностей EDM происходит после построения проекта.

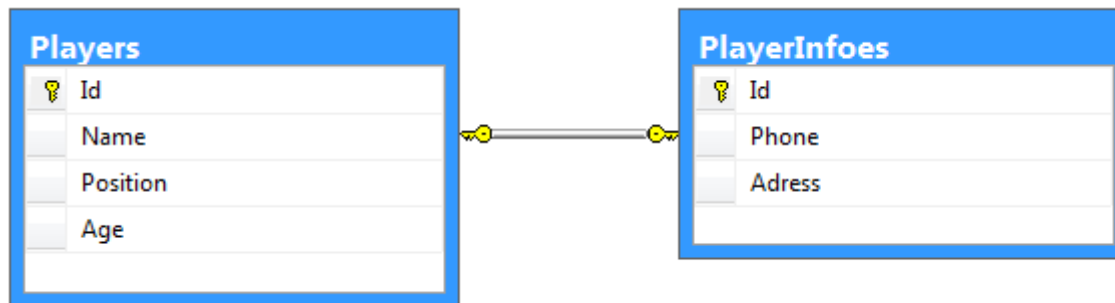
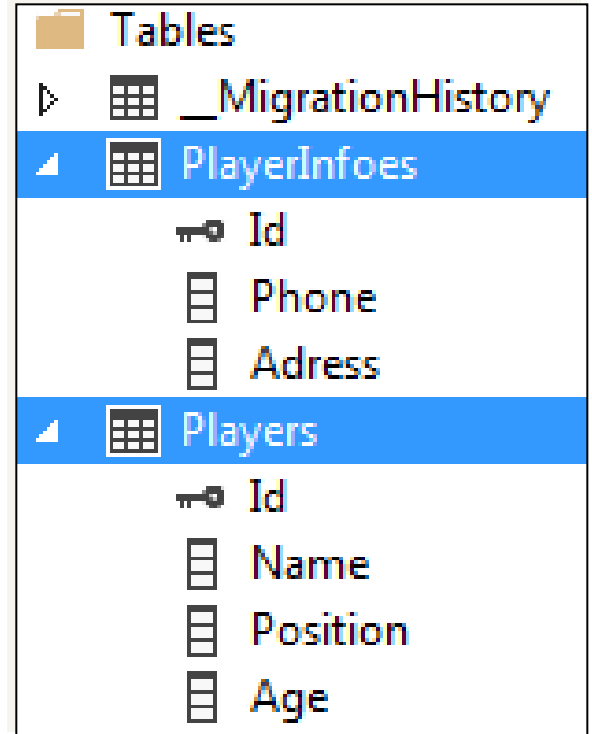




Связи (отношений) «один к одному»

```
public class Player
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Position { get; set; }
    public int Age { get; set; }
    public PlayerInfo PlayerInfo { get; set; }
}
```

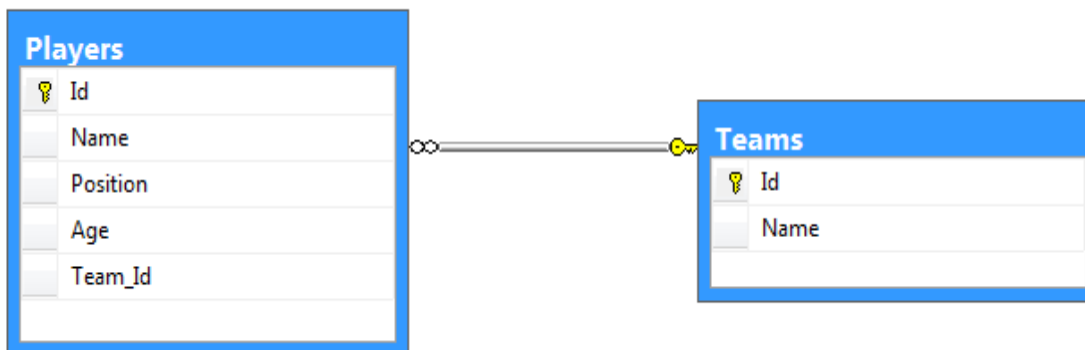
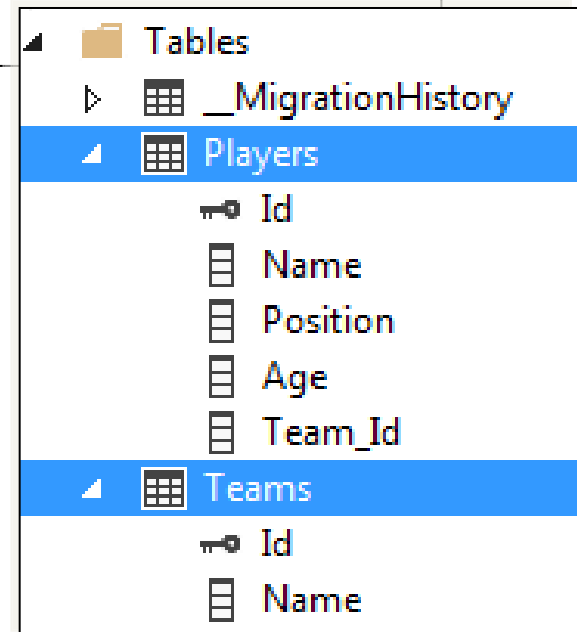
```
public class PlayerInfo
{
    [ForeignKey("Player")]
    public int Id { get; set; }
    public string Phone { get; set; }
    public string Adress { get; set; }
    public Player Player { get; set; }
}
```



СВЯЗИ (ОТНОШЕНИЙ) «ОДИН КО МНОГИМ»

```
public class Team
{
    public int Id { get; set; }
    public string Name { get; set; }
    public ICollection<Player> Players { get; set; }
}
```

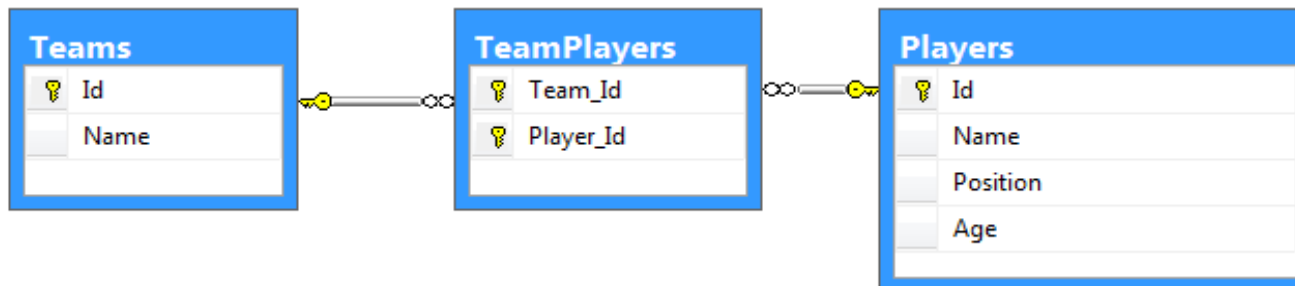
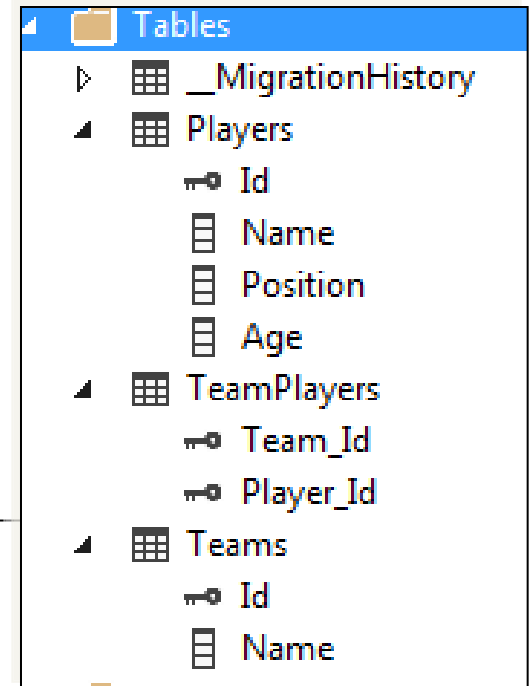
```
public class Player
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Position { get; set; }
    public int Age { get; set; }
    public Team Team { get; set; }
}
```



СВЯЗИ (ОТНОШЕНИЙ) «МНОГИЕ КО МНОГИМ»

```
public class Player
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Position { get; set; }
    public int Age { get; set; }
    public ICollection<Team> Teams { get; set; }
}
```

```
public class Team
{
    public int Id { get; set; }
    public string Name { get; set; }
    public ICollection<Player> Players { get; set; }
}
```



Демонстрация

- Code First

Data Annotations

```
[MaxLength(20), MinLength(5)]  
public string Name
```

Fluent API

```
modelBuilder.Entity<Post>()  
    .Property(p => p.Title).HasMaxLength(10);
```

Набор атрибутов Data Annotations

- System.ComponentModel.DataAnnotations
 - ▲ { } System.ComponentModel.DataAnnotations
 - ▷ AssociatedMetadataTypeTypeDescriptionProvider
 - ▷ AssociationAttribute
 - ▷ BindableTypeAttribute
 - ▷ CompareAttribute
 - ▷ ConcurrencyCheckAttribute
 - ▷ KeyAttribute
 - ▷ MaxLengthAttribute
 - ▷ MetadataTypeAttribute
 - ▷ MinLengthAttribute
 - ▷ PhoneAttribute
 - ▷ RangeAttribute
 - ▷ RegularExpressionAttribute
 - ▷ RequiredAttribute
 - ▷ ScaffoldColumnAttribute
 - ▷ ScaffoldTableAttribute
 - ▷ StringLengthAttribute
 - ▷ TimestampAttribute

- ▲ { } System.ComponentModel.DataAnnotations.Schema
 - ▷ ColumnAttribute
 - ▷ ComplexTypeAttribute
 - ▷ DatabaseGeneratedAttribute
 - ▷ DatabaseGeneratedOption
 - ▷ ForeignKeyAttribute
 - ▷ InversePropertyAttribute
 - ▷ NotMappedAttribute
 - ▷ TableAttribute

DataAnnotations: KeyAttribute

1. Свойство с именем **Id**



```
public partial class Item
{
    public Guid Id { get; set; }
}
```

1. Свойство с именем
[имя_класса]Id



```
public partial class Item
{
    public Guid ItemId { get; set; }
}
```

!

Необходим атрибут **[Key]**, если
имя ключевого свойства произвольно!

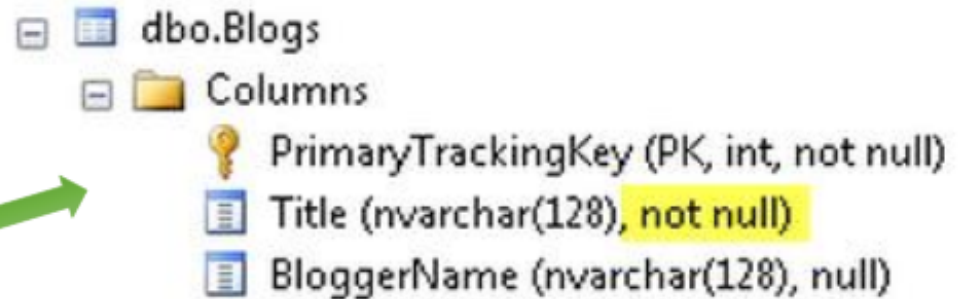
```
public partial class Item
{
    [Key]
    public Guid GlobalItemKey { get; set; }
}
```

CyberBionics

DataAnnotations: RequiredAttribute

Задание обязательности значения

```
public partial class Blog
{
    [Key]
    public int PrimaryTrackingKey { get; set; }
    [Required]
    public string Title { get; set; }
}
```



blog

Title

The Title field is required.

BloggerName

DataAnnotations: MinLength и MaxLength

Задание допустимой длины значения

```
public partial class Blog
{
    [Key]
    public int PrimaryTrackingKey { get; set; }
    [Required]
    public string Title { get; set; }
    [MinLength(5), MaxLength(10)]
    public string BloggerName { get; set; }
}
```

Columns

- PrimaryTrackingKey (PK, int, not null)
- Title (nvarchar(128), not null)
- BloggerName (nvarchar(10), null)

Blog

Title

BloggerName

Create

DataAnnotations: NotMappedAttribute

Вычисляемое свойство

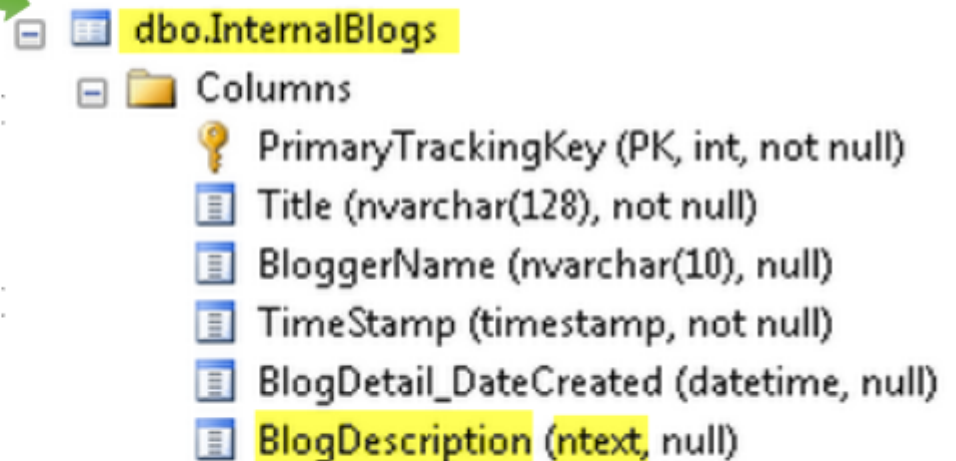
```
public partial class Blog
{
    [Key]
    public int PrimaryTrackingKey { get; set; }
    [Required]
    public string Title { get; set; }
    [MinLength(5), MaxLength(10)]
    public string BloggerName { get; set; }
    [NotMapped]
    public int BlogCode
    {
        get
        {
            return Title.Substring(0, 1) + ":" + BloggerName.Substring(0, 1);
        }
    }
}
```

Поле не сохраняется в БД.

DataAnnotations: TableAttribute и ColumnAttribute

Переименование таблиц/колонок

```
[Table("InternalBlogs")]
public partial class Blog
{
    [Key]
    public int PrimaryTrackingKey { get; set; }
    [Required]
    public string Title { get; set; }
    [MinLength(5), MaxLength(10)]
    public string BloggerName { get; set; }
    public BlogDetails BlogDetail { get; set; }
}
[ComplexType]
public partial class BlogDetails
{
    public DateTime? DateCreated { get; set; }
    [Column("BlogDescription", TypeName="ntext")]
    public string Description { get; set; }
}
```



FluentAPI

```
protected override void OnModelCreating
                        (DbModelBuilder modelBuilder)
{
    base.OnModelCreating(modelBuilder);
    // конфигурация модели с применением Fluent Api
    modelBuilder.Configurations.Add(new TeamEntityTypeConfig());
    modelBuilder.Configurations.Add(new PlayerEntityTypeConfig());
}
```

```
public class PlayerEntityTypeConfig :
                        EntityTypeConfiguration<Player>
{
    public PlayerEntityTypeConfig()
    {
        ToTable("TeamPlayers");
        HasKey(p => p.Id);
        Property(p => p.Name).IsRequired().HasMaxLength(20);
        Property(p => p.Position).IsRequired().HasColumnName("Post");
    }
}
```