Julio 2018

Curso Entity Framework (EF) & LINQ

hchan@plenumsoft.com.mx

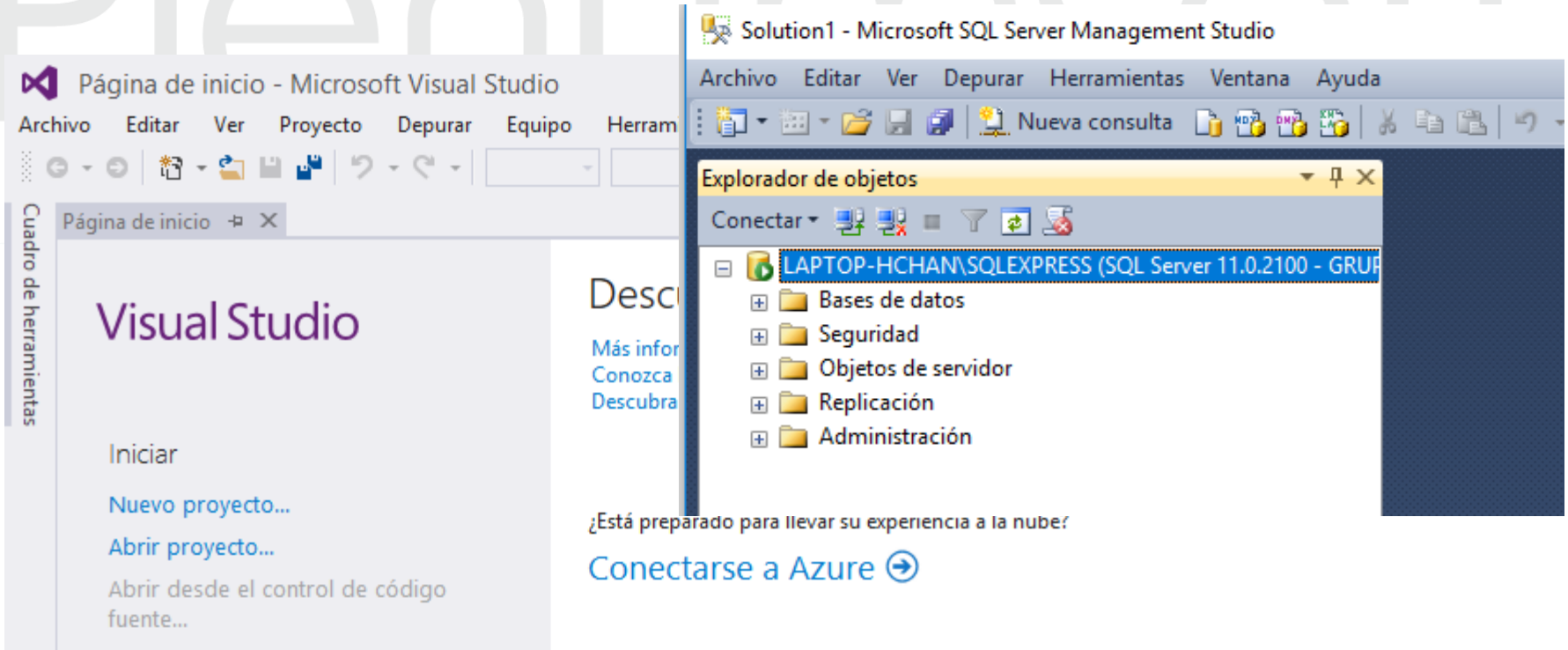
Bienvenidos

¿Qué necesitas saber?

No Requerido	Requerido
Experiencia con .NET	Conocimientos de base de datos
	Conocimientos de Programación
	Desarrollo Orientado a objetos

¿Qué necesitas tener?

- 2 cosas
  - Visual Studio 2015
  - Microsoft SQL Server 2012 o Superior con Management Studio.

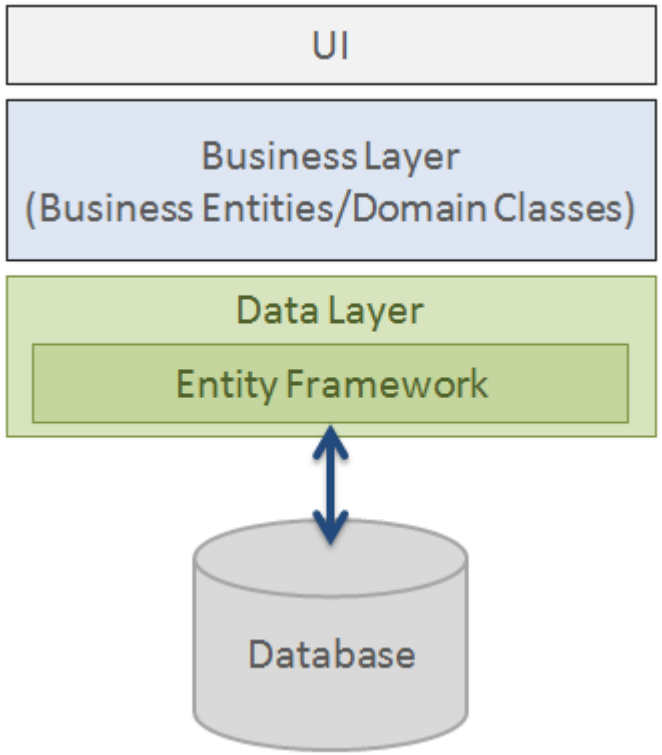


Objetivos del curso

- Entender y conocer el funcionamiento de Entity Framework
- Conocer y aplicar las convenciones de desarrollo de EF
- Conocer y aplicar las configuraciones que ofrece EF con las entidades y sus relaciones.
- Desarrollar un manejo medio-avanzado de EF

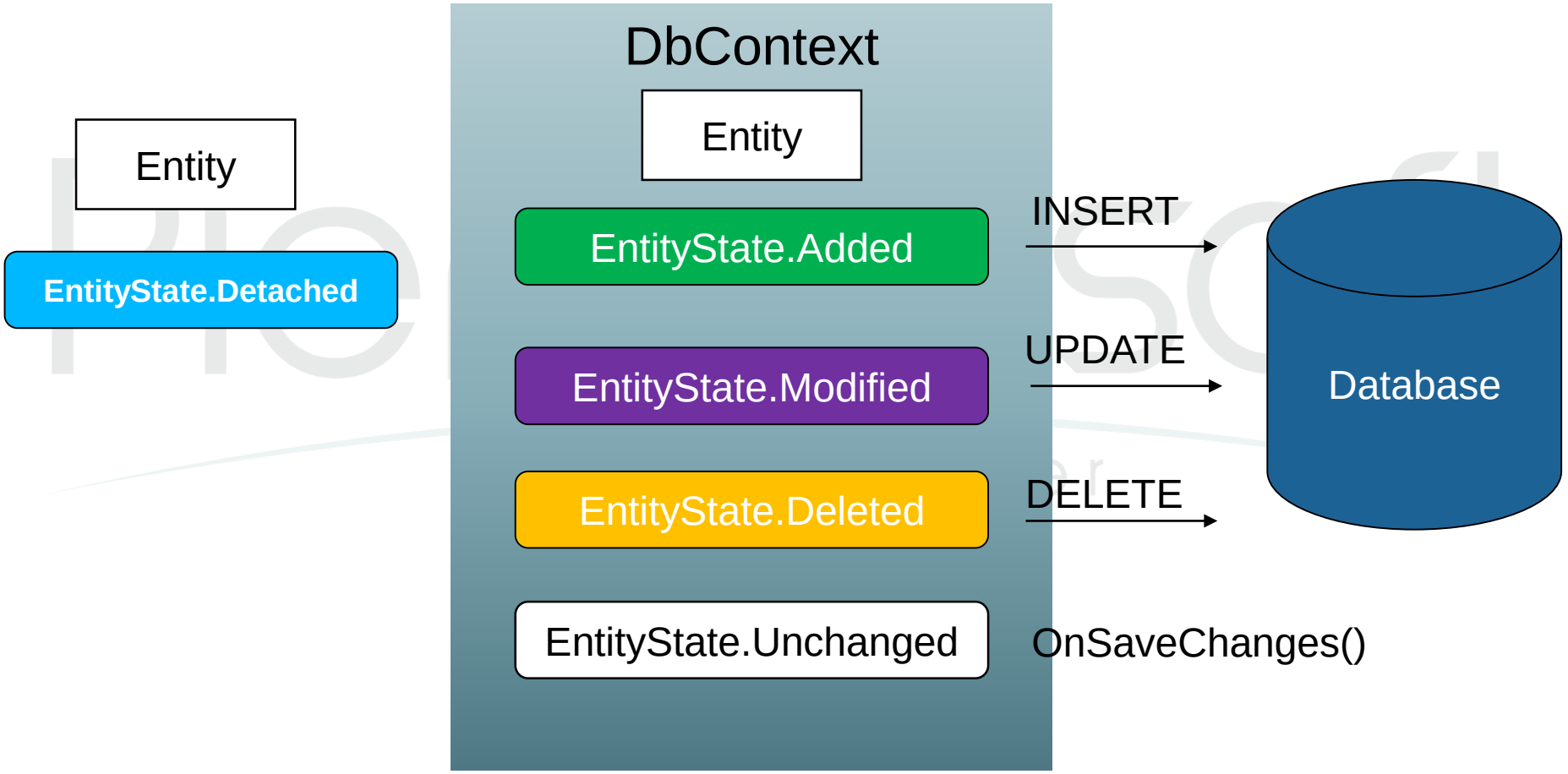
Qué es Entity Framework?

- ORM Framework
- Características
  - Modelado
  - Querying
  - Seguimiento de los cambios
  - Concurrencia
  - Transacciones
  - Cache
  - Configurable
  - Migraciones



Componentes principales

- DbContext
- Entidades
  - POCOs
  - Dynamic Proxy Entities (POCO Proxy)
    - Clase proxy en tiempo de ejecución permite Lazy loading


Entity
EntityState.Detached
DbContext
Entity
EntityState.Added
EntityState.Modified
EntityState.Deleted
EntityState.Unchanged
INSERT
UPDATE
DELETE
OnSaveChanges()
Database

Enfoques de desarrollo EF

- Database-First
  - Contexto y entidades se generan a partir de una base de datos existente.
- Code-First
  - Recomendado cuando no existe una base de datos para la aplicación.
  - Entidades y contexto crean la base de datos a partir de comandos de migración.
- Model First
  - Creación de entidades, relaciones y contexto a partir de un modelo visual

Enfoques de desarrollo EF

Database First

- Base de datos existente
- Entidades y el contexto a partir de la BD
- EDM Wizard
- Comandos EF

Code First

- BD a partir de las entidades y el contexto
- Diseño guiado por el Dominio
- Comandos EF

Model First

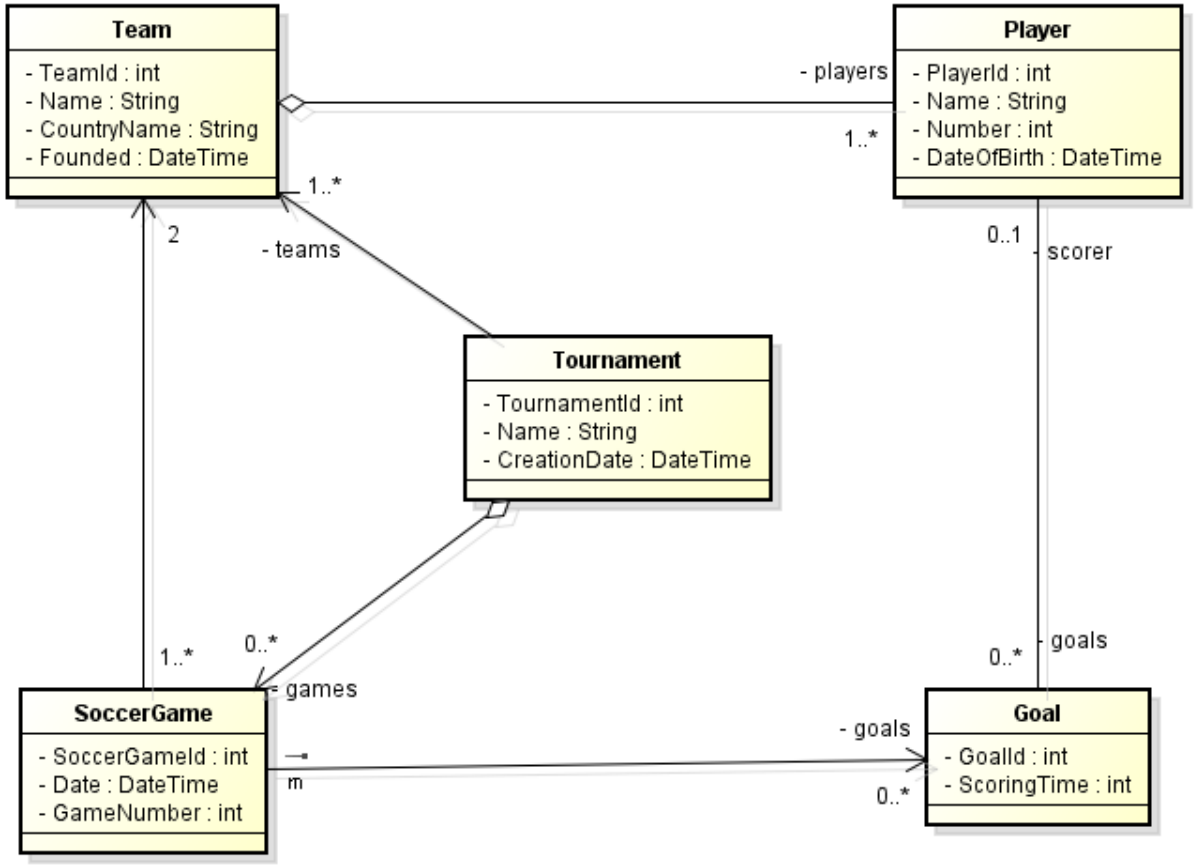
- Entidades y relaciones a partir de un modelo visual

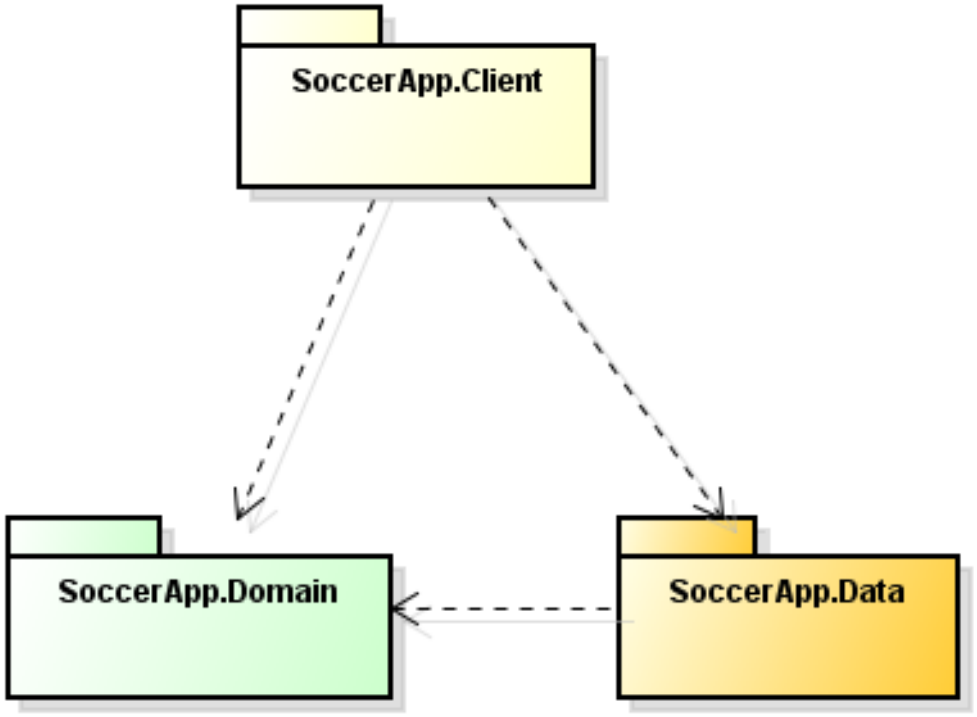
Herramientas

Middleware

Ejercicio 01

- Modelo de una “app” que registra resultados de un torneo de futbol.

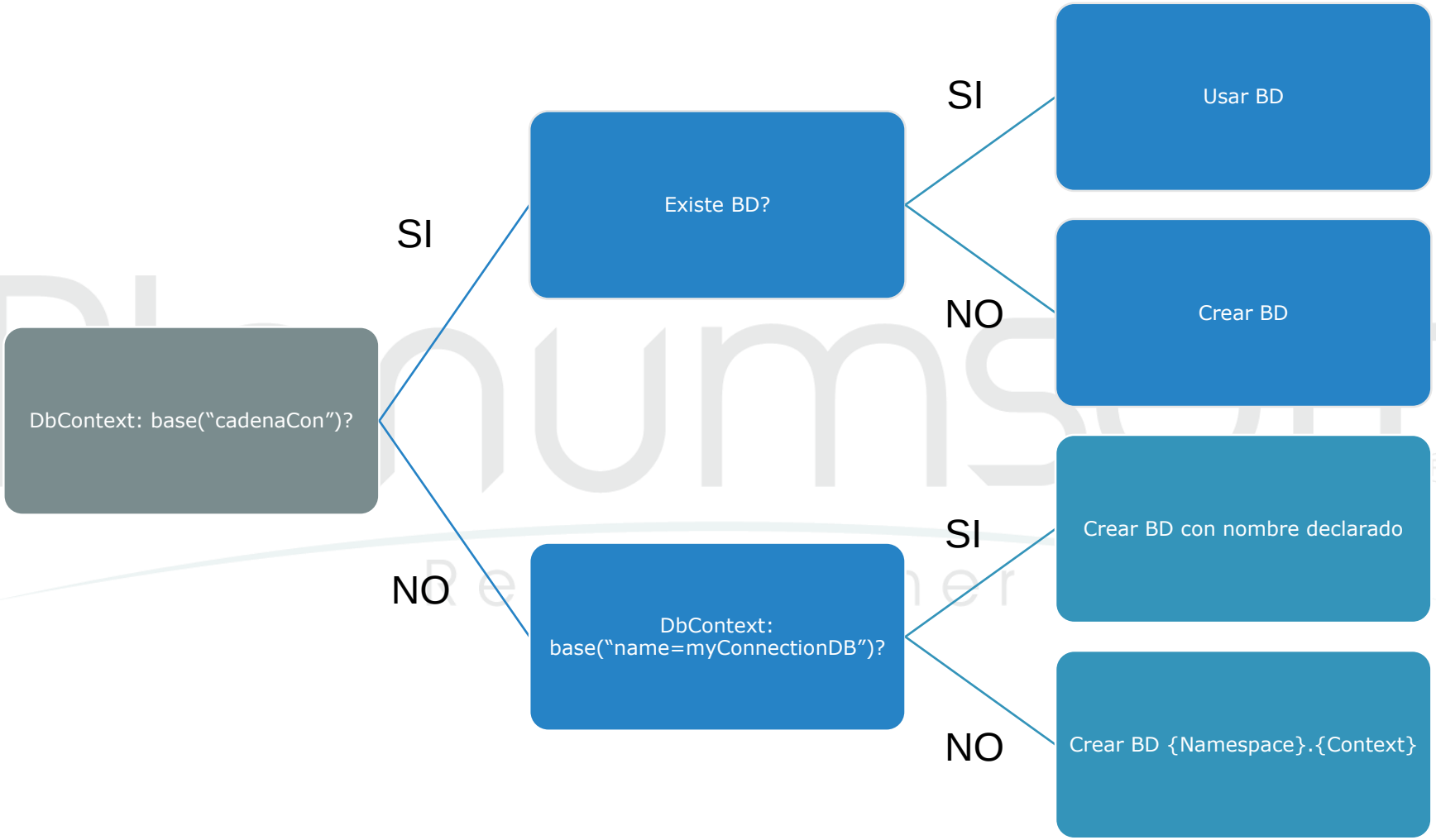



SoccerApp.Client
SoccerApp.Domain
SoccerApp.Data

Ejercicio 01 - Checklist

- Entidades
- Librería
- Configurar conexiones
- Crear Contexto
- Inicializar configuraciones enfoque Code-First
- Code-Based Migrations
- Insertar datos

Criterios EF sobre Base de datos – Code First



SESIÓN DOS

Continuación ejercicio 01

Tipos de configuraciones (Anotaciones & FluentApi)

Configuración de entidades con anotaciones

Convenciones por defecto

Convención para	Descripción
Schema	Por defecto en el dbo schema
Tables	Por defecto sufijo “s” a la endidad: <EntityName>+s
Primary key	Id, <EntityName>+Id No es sensible a mayúsculas o minúsculas
Foreign key	Si no se especifica la propiedad para la relación de la llave foránea por defecto se usa: <Dependent Navigation Property Name> + "_" + <Principal Entity Primary Key Property Name>
Null Column	Por defecto las propiedades marcadas como “nullable” y las propiedades de relación
Not Null Column	Por defecto las propiedades de tipos primitivos que no se marquen como nullable
Mapping	Por defecto todas las propiedades excepto las marcadas con [NotMapped]
Cascade delete	Por defecto

Ejercicio 01

- Agregar propiedad *active* a las entidades Team y Player
- Consultar todos los teams que participan en un torneo
- Agregar un soccergame a un torneo
- Registrar un goal en un soccergame
- Eliminar un jugador de un equipo

Configuración de entidades con EF

- Ya observamos como usar EF siguiendo las convenciones con el enfoque Code-First
- Existen dos maneras de configurar nuestras entidades:
  - Anotaciones sobre los atributos
    - *System.ComponentModel.DataAnnotations*
  - Fluent API
    - Sobrecargando en el contexto

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    //Configure domain classes using modelBuilder here..

}
```

Anotaciones

[Table(string name, Properties:[Schema = string])	[Column (string name, Properties:[Order = int],[TypeName = string])
[Key]	[NotMapped]
[ForeignKey(name string)]	[Index]
[Required]	[MaxLength(50)] string & byte[]
[Timestamp] byte[]	[InverseProperty("EntityRef")]

SESIÓN TRES

Fluent Api

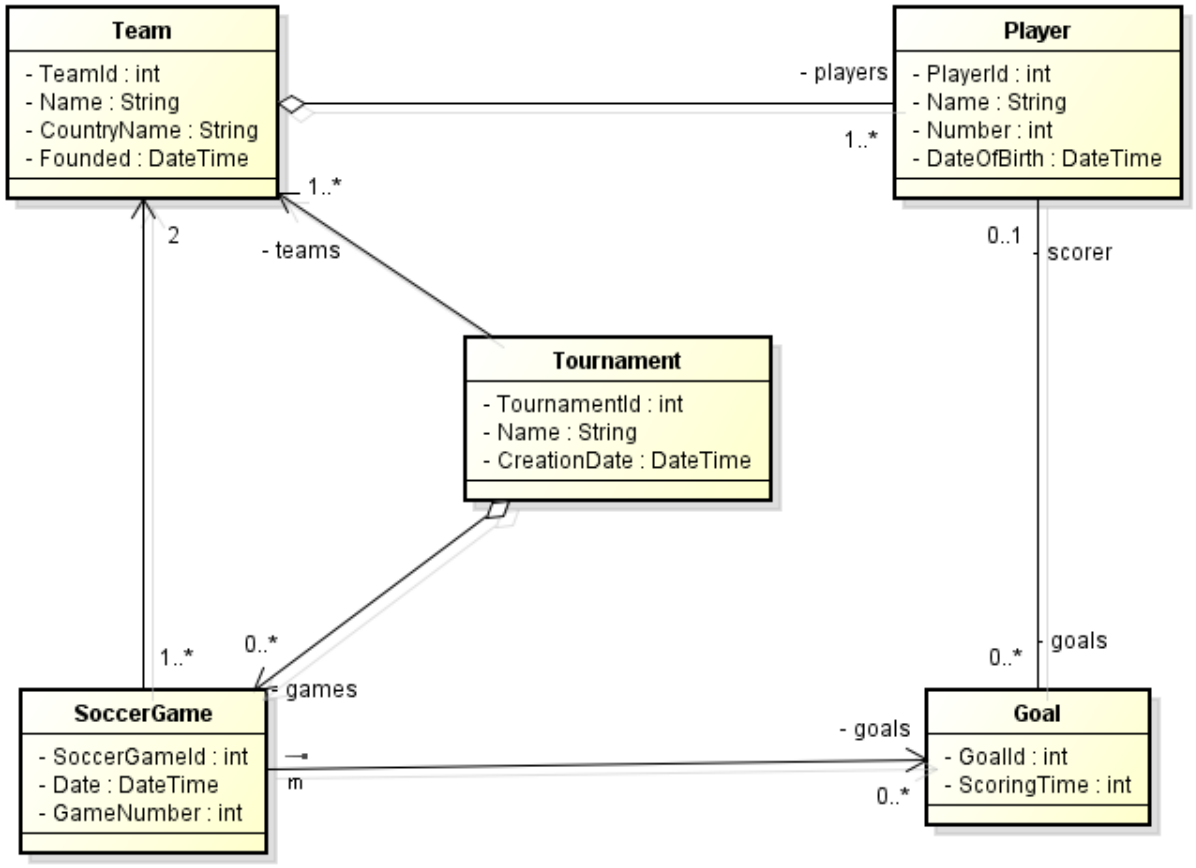
Tipos de relaciones

Herencia

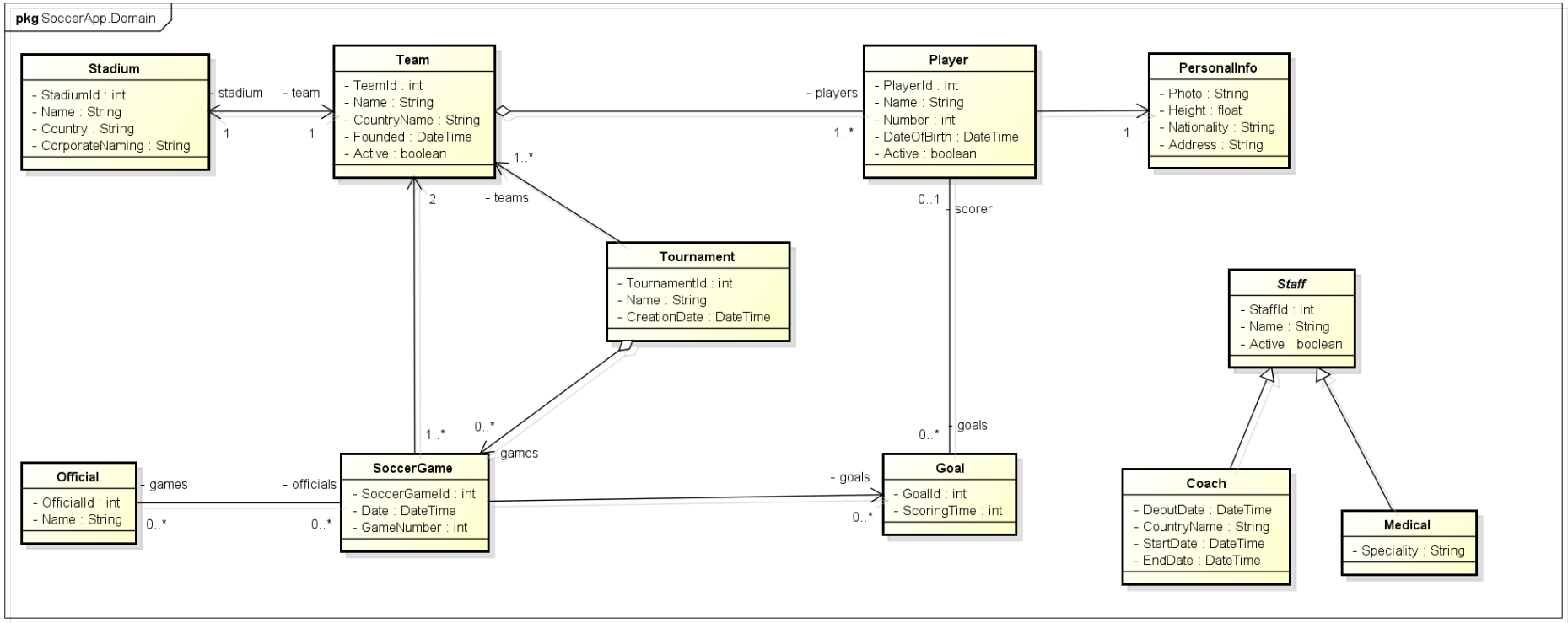
FluentApi

- Aplica conceptos de “Method Chaining”
- Sintaxis que invoca múltiples métodos que permite llamar en cadena entre si en una sola sentencia de código.
- Configura los siguientes aspectos de un modelo:
  - Schema
  - Entidades
  - Propiedades

Ejercicio 01 – Fluent API



Agregando nuevas entidades al modelo

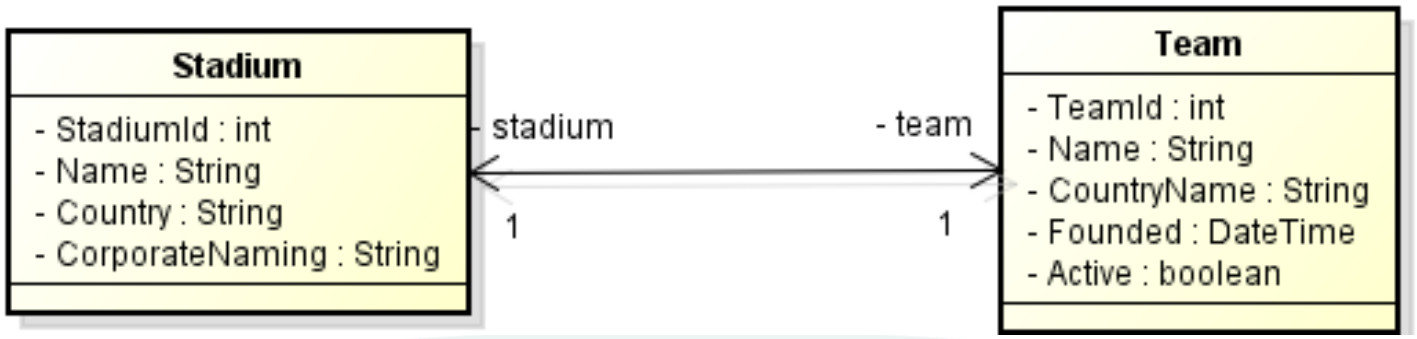


Tipos de relaciones – Fluent API

- One-to-One
- One-to-Many
- Many-to-Many

One-To-One - Checklist

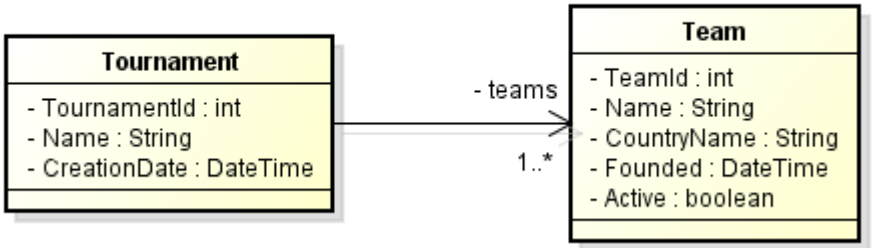
- One-to-One en nuestro modelo



- Definir ambas relaciones en el contexto
- Agregar migración
- Aplicar migración

One-To-Many - Checklist

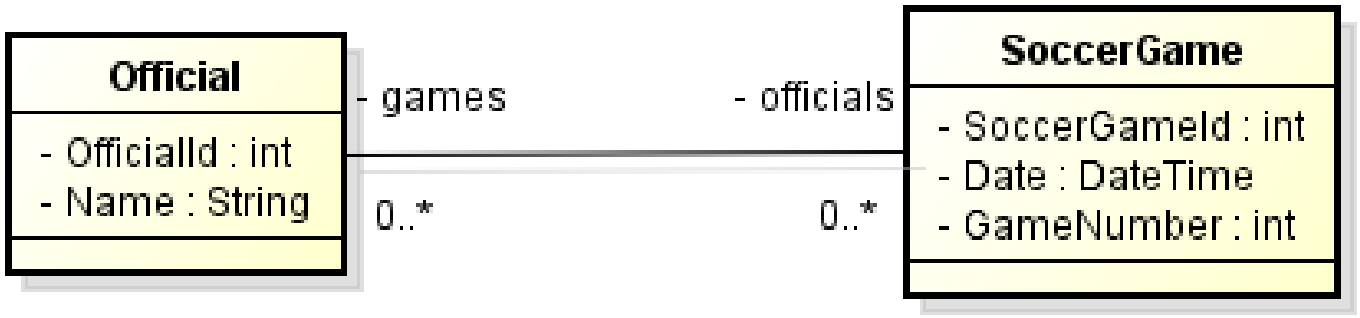
- One-To-Many en nuestro modelo



- Definir ambas relaciones en el contexto
- Agregar migración
- Aplicar migración

Many-To-Many - CheckList

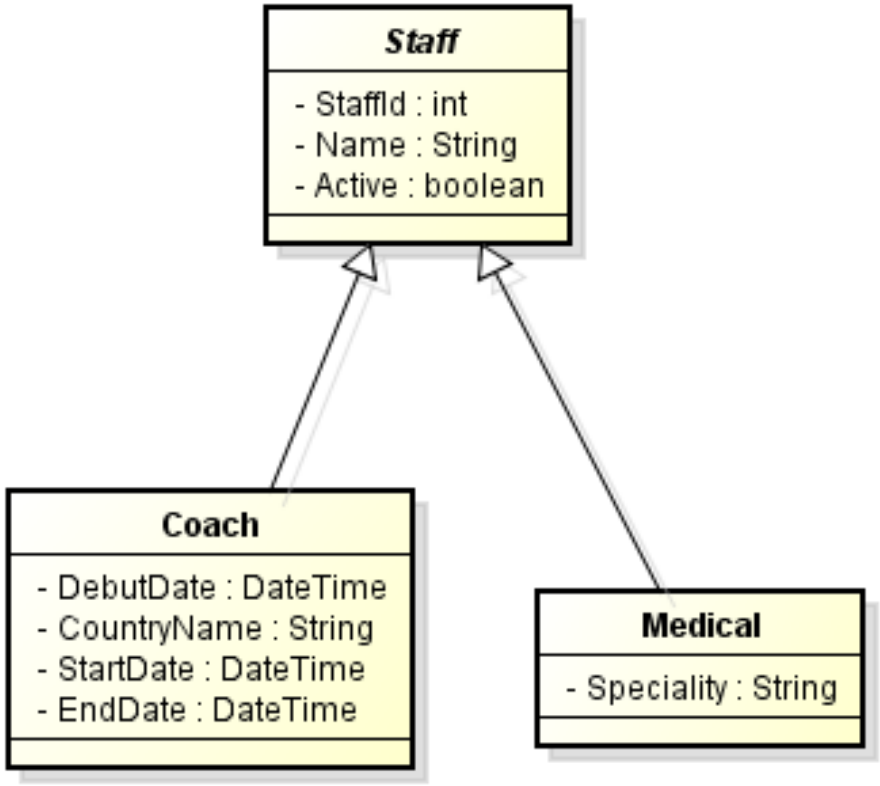
- Many-To-Many en nuestro modelo



- Definir ambas relaciones en el contexto
- Agregar migración
- Aplicar migración

Tipos de Herencia – Fluent API

- TPH – Tabla por herencia
  - Una tabla que controla toda la herencia
  - Discriminado para distinguir clases concretas
  - Mapeo por defecto de EF
  - Permite polimorfismo
- TPT – Tabla por tipo
  - Cada clase o subclase tiene su propia tabla, incluyendo la clase padre
- TPC – Tabla por clase concreta
  - Se usa una tabla por cada clase concreta
  - Descarta el polimorfismo y relaciones de herencia en el SQL


Staff
- StaffId : int
- Name : String
- Active : boolean
Coach
- DebutDate : DateTime
- CountryName : String
- StartDate : DateTime
- EndDate : DateTime
Medical
- Speciality : String

Herencia - Checklist

- Agregar clases a nuestro modelo
- Configurar clases en nuestro contexto usando TPH
- Agregar migración
- Aplicar migración
- Probar