**Обратный инжиниринг и миграция**

**Обратный** – это получение классов на основе существующих (…), используется при DataBase

**Миграция** – это внесение изменений в бд для её синхронизации, моделью данных которая была изменена в приложении. Используется при подходе CodeFirst

**Для работы нужно:**

* библиотека для работы с БД через Core
* Microsoft.EntityFraemworkCore.SqlServer
* В пакете Tools есть пакет Scaffold-DbContext – средство для обратного инжиниринга
* Add-Migration – средство для миграции
* Могут быть запущены консоли диспетчера пакетов или в PowerShell(PS)
* Dotnet ef …
* Scaffold-DbContext “строка подключения” ПакетEfCore

**Дополнительные параметры:**

* НазваниеПараметра
* Context ИмяКонтекстаБД
* ContextDir ДирикторияКонтекста
* OutputDir ДиректорияМоделейДанных
* Tables таблица1, таблица2, …
* Namespace ИмяПространстваМоделейДанных
* ContextNamespace ИмяПространстваКонтекстаБД
* Force (перезаписать существующие модели)
* UseDatabaseNames (использовать имена из БД)

Scaffold-DbContext "Data Source = PRSERVER\SQLEXPRESS; Initial Catalog = ispp3511; User ID = ispp3511; Password = 3511; Trust Server Certificate=True" Microsoft.EntityFrameworkCore.SqlServer -ContextDir DatabaseContext -OutputDir Models -Context AppDbContext -Tables Category, Game –Force

* Сгенерированно в классе, приведён метод OnModelCreating(указ. Ограничения типов данных, PK/FK/Constrain и тп)
* Fluent API – API позволяющие настроить модель данных
* Классы моделей данных генерируются с модификатором partial (можно дописат класс в отдельном файле расширив его функциональность) можно настроить модель используя аннотации [Table("ИмяТаблицы")][PrimaryKey("")][Required] [MaxLength()]
* NotMapped – игнорировать совйтсва
* Если у свойства есть только get или set оно NotMap
* Update-Database ИмяМиграции
* Remove-Migration ИмяМиграции