[SqlSugar ORM 5.X 官网 、文档、教程 - SqlSugar 5x - .NET果糖网 (donet5.com)](https://www.donet5.com/home/Doc)

# 连接数据库

## 数据库连接字符串

* Sqlite: string SqliteConnectionString = @"DataSource=" + DBPath;
* Pg: string PGConnectionString =

"PORT=5433;DATABASE=ORMTestSqlSugar;HOST=localhost;PASSWORD=hnddzy;USER ID=hnddzy";

## 创建对象

### SqlSugarClient 随用随New

SqlSugarClient db = new SqlSugarClient(new ConnectionConfig()

            {

                ConnectionString = PGConnectionString,

                DbType = DbType.PostgreSQL,

                IsAutoCloseConnection = true

            });

### SqlSugarScope 单例模式

private static SqlSugarScope DB = new SqlSugarScope(new ConnectionConfig()

        {

            ConnectionString = SqliteConnectionString,

            DbType = DbType.Sqlite,

            IsAutoCloseConnection = true,

        });

### 区别



# 创建Model

## 手动

[SugarTable("InvGroups", TableDescription = "InvGroup")]

    public class InvGroup

    {

        [SugarColumn(IsPrimaryKey = true, ColumnDescription = "索引，主键")]

        public int GroupID { get; set; }

        public int? CategoryID { get; set; }

        public string GroupName { get; set; }

        [SugarColumn(IsNullable = true)]

        public int IconID { get; set; }

        public bool? UseBasePrice { get; set; }

        public bool? Anchored { get; set; }

        public bool? Anchorable { get; set; }

        public bool? FittableNonSingleton { get; set; }

        public bool? Published { get; set; }

    }

主要通过特性来设置表字段属性，列举一些常用的特性

### 表

* 表名

SugarTable("InvGroups")

不设置表名则默认为类名称

* 表描述

[SugarTable(TableDescription = "InvGroup")]

对于不能写表描述的Sqlite，这个不会有任何作用

### 字段

|  |  |
| --- | --- |
| IsIdentity | 是否创建自增标识 |
| IsPrimaryKey | 是否创建主键标识 |
| ColumnName | 创建数据库字段的名称（默认取实体类属性名称） |
| ColumnDataType | 创建数据库字段的类型 |
| IsIgnore | ORM不处理该列 |
| ColumnDescription | 注释 |
| IsNullable | 是否可以为null默，为false |

不指定IsNullable时，默认不可为null，对于有着大量可为null的情况下，每个都写一个IsNullable会过于麻烦，这时候可以在类型后面加上个?表示可为null

public int? CategoryID { get; set; }

然后还需要在连接配置加上扩展属性

ConfigureExternalServices = new ConfigureExternalServices

            {

                EntityService = (c, p) =>

                {

                    // int?  decimal?这种 isnullable=true

                    if (c.PropertyType.IsGenericType &&

                    c.PropertyType.GetGenericTypeDefinition() == typeof(Nullable<>))

                    {

                        p.IsNullable = true;

                    }

                }

            }

此时完整的连接如下

private static SqlSugarScope DB = new SqlSugarScope(new ConnectionConfig()

        {

            ConnectionString = SqliteConnectionString,

            DbType = DbType.Sqlite,

            IsAutoCloseConnection = true,

            ConfigureExternalServices = new ConfigureExternalServices

            {

                EntityService = (c, p) =>

                {

                    // int?  decimal?这种 isnullable=true

                    if (c.PropertyType.IsGenericType &&

                    c.PropertyType.GetGenericTypeDefinition() == typeof(Nullable<>))

                    {

                        p.IsNullable = true;

                    }

                }

            }

        });

但是只能对值类型这样操作，对于string之类的引用类型，只能手动指定IsNullable = true

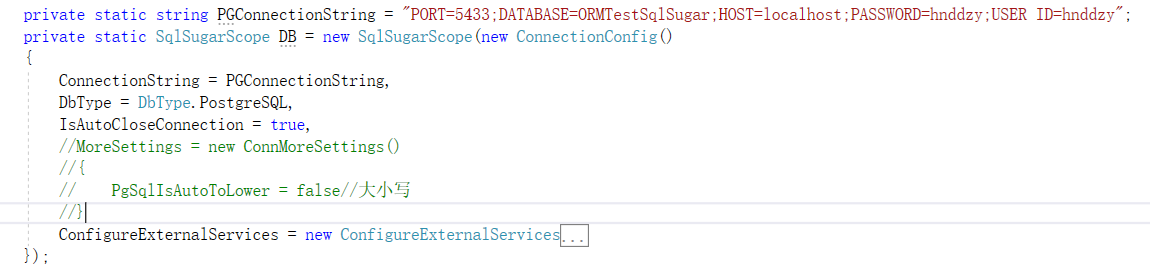
## 自动Db First

Db First可以通过已有的数据库自动创建好类文件

#### 方式一：代码

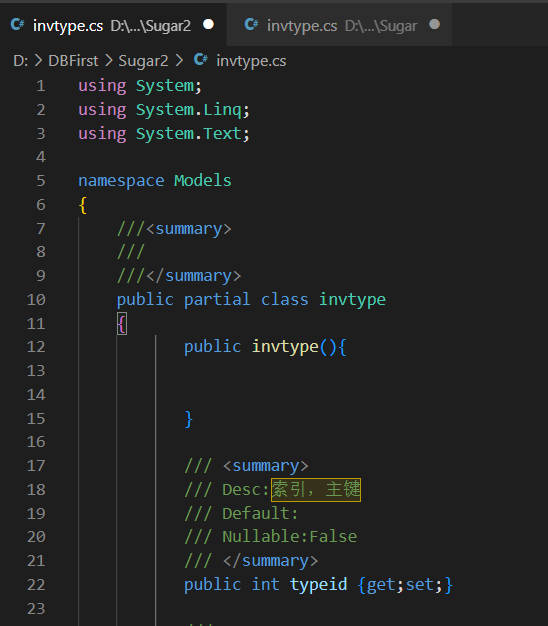
适用于简单的应用场景，很简单方便

首先需要按常规配置好数据库连接，是否单例均可



1. DB.DbFirst.CreateClassFile("D:\\DBFirst\\Sugar2", "Models");

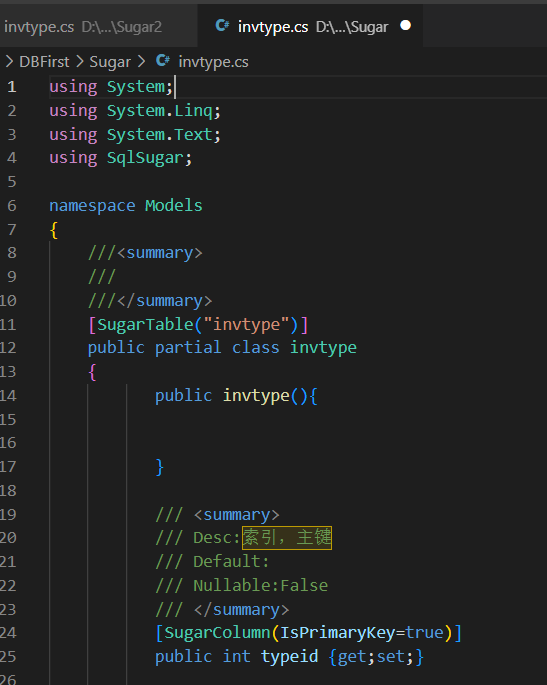
直接在指定路径下生成数据库全部表的实体类文件，命名空间均为Models



这样生成的类是不带有任何SqlSuagr特性的，比较通用，只会匹配好是否为空，其他主键、表名之类的不会指定

1. 如果需要自动生成SqlSugar特性，加多个*IsCreateAttribute*即可

DB.DbFirst.IsCreateAttribute().CreateClassFile("D:\\DBFirst\\Sugar2", "Models");



1. 如果不需要生成全部的表，只需添加个*where*筛选

DB.DbFirst.Where(p=>p.StartsWith("inv")).IsCreateAttribute().CreateClassFile("D:\\DBFirst\\Sugar2", "Models");

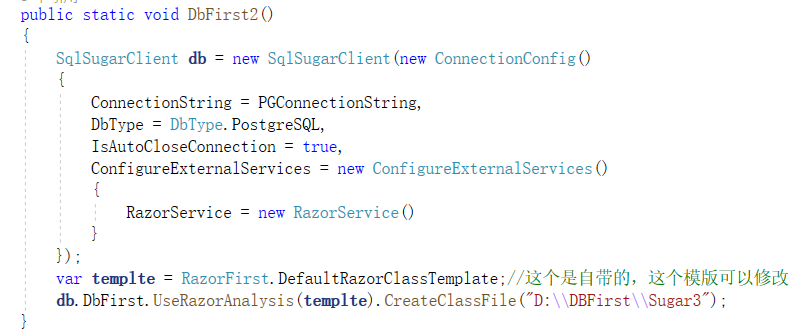
1. 还可以带有默认值、格式化文件名、类名、属性名等等设置

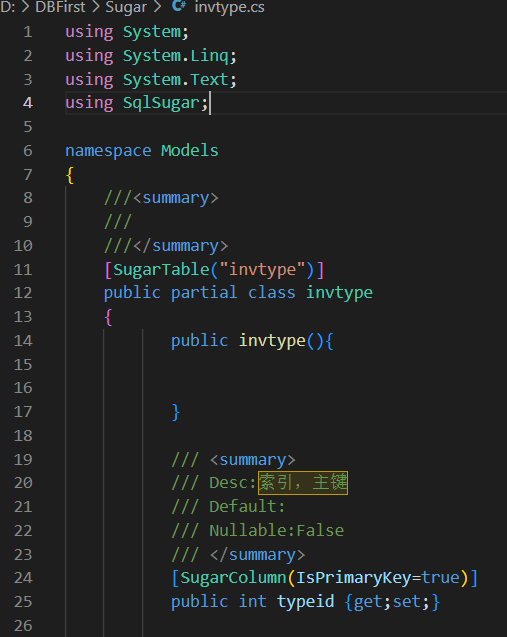
#### 方法二：Razor模版

1. 安装*RazorEngine* 包
2. 创建一个Razor类



1. 使用代码按Razor模板创建类文件





这样创建出来的类是带有SqlSugar特性的，需要自定义创建出来的类内容，可以修改模板文件，怎么改，我不知道

#### 方法三：WebFirst

直接用工具WebFirst在图形界面上生成实体



# 3.建库建表

使用Code First可以很方便地从已有Model生成表

## 建库

自动按数据库连接创建好一个库

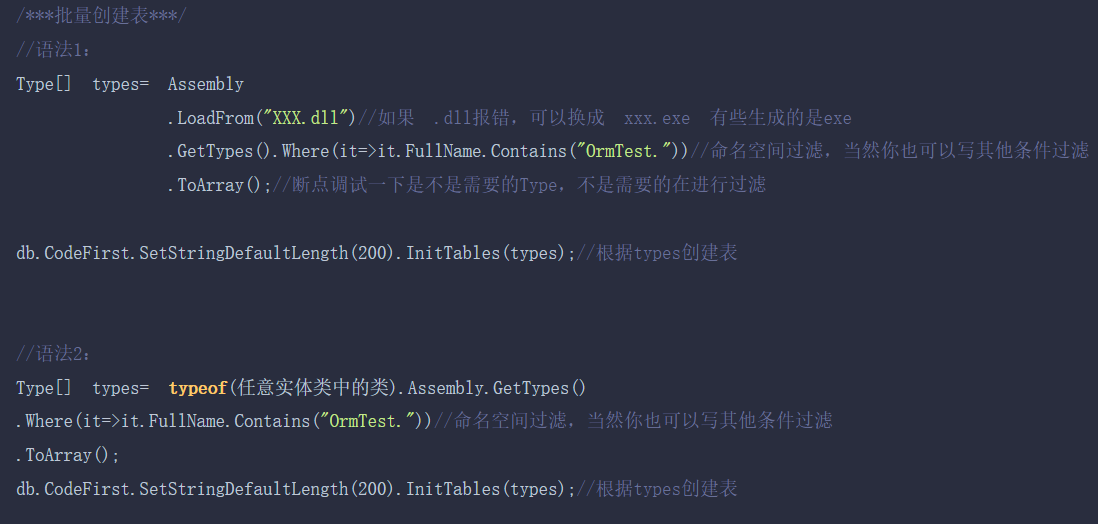
DB.DbMaintenance.CreateDatabase()

## 建表

### 逐个建表

DB.CodeFirst.InitTables(typeof(InvType));

### 批量建表



# 增删改查

## 增

InvType invType = new InvType();

invType.TypeID = 5233;

invType.TypeName = "5233";

DB.Insertable(invType).ExecuteCommand();

New一个对象，然后Insertable即可，可以是单个，也可以是List。如果是大量数据，还可以用BulkCopy

DB.Fastest<InvType>().BulkCopy(list);

## 删

DB.Deleteable(obj).ExecuteCommand();

支持单个、批量

## 改

DB.Updateable(obj).ExecuteCommand();

支持单个、批量

## 查

DB.Queryable<InvType>().ToList();

DB.Queryable<InvType>().Where(p => p.GroupID == 6).ToList();

DB.Queryable<InvType>().Where(p => p.GroupID == 6).Select(p => p.TypeID).ToList();

DB.Queryable<InvType>().Where(p => p.GroupID == 6).Select(p => new { p.TypeID, p.GroupID }).ToList();

DB.Queryable<InvType>().ToPageList(1,10).ToList();

DB.Queryable<InvType>()

                .LeftJoin<InvGroup>((t, g) => t.GroupID == g.GroupID)

                .Select((t, g) => new { Type = t, Group = g })

                .ToList();

# 动态表达式

上面的查询方式是固定的，应付不了一些业务需要设置动态条件去查询，这时候就可以用上动态表达式了。动态表达式可以扩展成多种多样用法，下面简单列举几个常见的

## 基本条件查询

var exp = SqlSugar.Expressionable.Create<DbModels.TowerLine>();

            if(Voltage!=null)

            {

                exp.And(p => p.VoltageId == Voltage.Id);

            }

            if(DataSouce!=null)

            {

                exp.And(p => p.DataSourceId == DataSouce.Id);

            }

            db.Queryable<DbModels.TowerLine>().Where(exp.ToExpression()).ToList();

if(Voltage!=null)

{

exp.And(p => p.VoltageId == Voltage.Id);

}

还可以写成这样

exp.AndIF(Voltage != null, p => p.VoltageId == Voltage.Id);

## 多表查询

不仅可以条件查询，还可以进行多表查询

var exp=Expressionable.Create<Order,T2,T3>()

.And((x,y,z)=>z.id==1).ToExpression();

## In模糊搜索

foreach (var item in names)

{

    exp.Or(it => it.Name.Contains(item.ToString()));

}