# 支持的数据库

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | SqlSugar | FreeSql | Chloe | EF |
| SqlServer | ✓ | ✓ | ✓ | ✓ |
| Oracle | ✓ | ✓ | ✓ | ✓ |
| PostgreSQL | ✓ | ✓ | ✓ | ✓ |
| MySql | ✓ | ✓ | ✓ | ✓ |
| Sqlite | ✓ | ✓ | ✓ | ✓ |
| Access | ✓ | ✓ | 🗶 | ✓ |
| 人大金仓 | ✓ | ✓ | 🗶 | ✓ |
| 神通 | ✓ | ✓ | 🗶 | ✓ |
| 达梦 | ✓ | ✓ | 🗶 | ✓ |

除了Chloe，其他几个都支持常见的国内外数据库

# 支持的.Net平台

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | SqlSugar | FreeSql | Chloe | EF |
| .Net Framework | ✓ | ✓ | ✓ | ✓ |
| .Net Core | ✓ | ✓ | ✓ | ✓ |
| .Net 5 | ✓ | ✓ | ✓ | ✓ |
| .Net 6 | ✓ | ✓ | ✓ | ✓ |

都支持.Net Framework 4.0及以上

# Code First

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | SqlSugar | FreeSql | Chloe | EF |
| 支持Code First | ✓ | ✓ | 🗶 | ✓ |
| 上手难度 | 有手就行 | 有手就行 | 有手就行 | 有手就行 |
| 支持自动建库 | ✓ | 🗶 | 🗶 | ✓ |

除开Chloe不支持，另外三个使用起来区别不大，都是几行代码就可以搞完。仅有一点，FreeSql不支持创建数据库，需要提前建好。

## SqlSugar

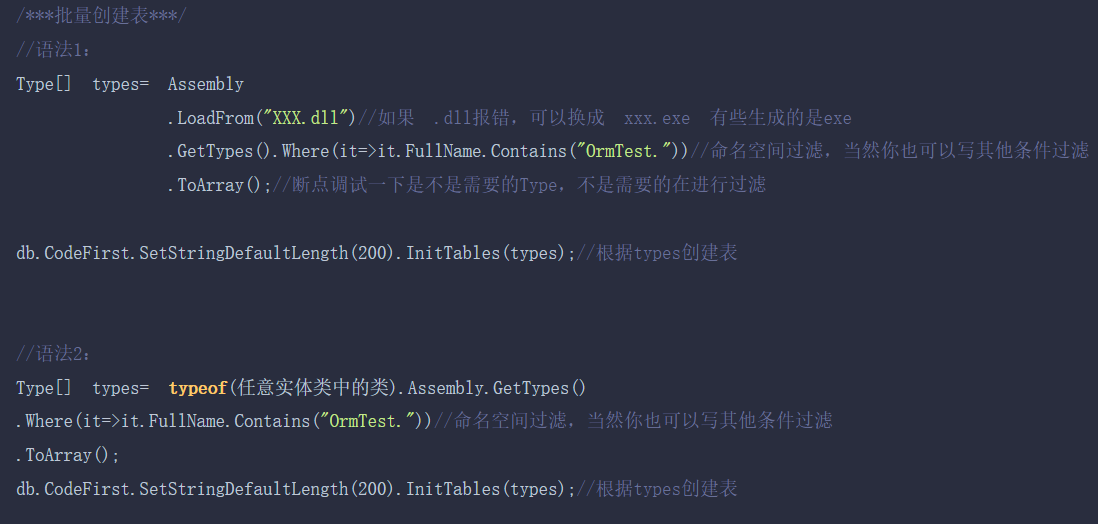
可创建数据库

***DB.DbMaintenance.CreateDatabase();***

可以手动创建单个、多个表

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

可自动批量创建表



## FreeSql

不可创建数据库，可手动创建表

***Fsql.CodeFirst.SyncStructure<InvType>();***

可批量创建表，通过获取所有具有特性标签为 TableAttribute的类或者指定某个命名空间下的所有类。另外有个很有意思的自动创建表方式，在数据库连接启用***UseAutoSyncStructure*** 后，就无需再写任何代码，每个表在使用到的时候会自动判断是否存在，不存在就生成。但是每次都会判断一遍，可能存在细微的性能影响。

## Chloe

完全不支持

## EF

CodeFirst通过程序包管理控制台来实现，写好每个表的DbContext及配置好数据库连接后，控制台依次运行以下命令即可创建每个DbContext对应的表。如果不存在数据库，会自动创建

***Enable-Migrations***

***Add-Migration Init***

***Update-Database***

在Add-Migration Init 执行完后，会生成Migrations文件，可以手动修改来进一步自定义要生成的表，Update-Database是依据这里的Migrations文件来创建表的

# Db First

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | SqlSugar | FreeSql | Chloe | EF |
| 支持Db First | ✓ | ✓ | ✓ | ✓ |
| 一两行代码即可 | ✓ | 🗶 | 🗶 | 🗶 |
| 支持模板文件 | ✓ | ✓ | ✓ | 🗶 |
| 支持图形化界面 | ✓ | 🗶 | 🗶 | ✓ |

SqlSugar支持的方式最多，简单的，复杂的，傻瓜式的都有，Chloe最难用，自成一套的模板语法，默认模板只支持MySql，简单的需求都不能开箱即用。FreeSql、EF凑合能用

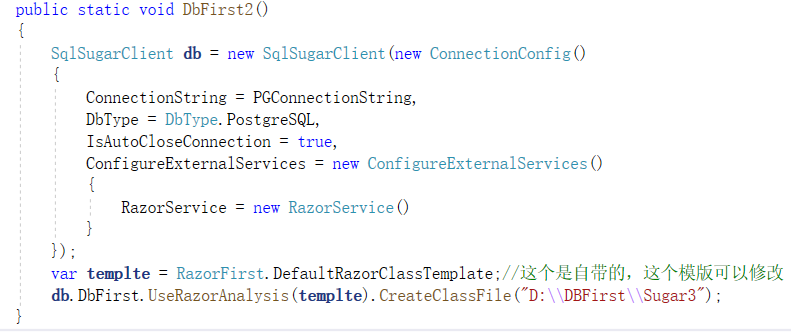
## SqlSugar

有多种方式实现

1. 最简单的，一行代码搞定

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

1. 进阶版，基于Razor模板



一般情况下，使用默认的模板即可，需要更好自定义，则需要改Razor模板

1. 终极傻瓜式图形界面操作



## FreeSql

与SqlSugar类似，支持基于Razor模板的方式创建，但是仅支持在控制台项目使用，需要单独创建一个控制台程序，安装FreeSql.Generator包来实现。同样带有满足一般需求的默认模板，也可自己创建新模板。用起来也很简单，除开自己写模板部分



## Chloe

并非作为Chloe功能的一部分实现，而是单独开了个坑：acgen，一个控制台程序。

“acgen 未使用其它第三方模版引擎，是鄙人根据个人喜好自主设计的一套简单模版语法。”

不再像SqlSugar、FreeSql那样用Razor模板了，但是，官方提供的模板只支持MySql，至于其他数据库：

“如需连接其它类型数据库，请参考 mysql 实现自行扩展即可。”

## EF

EF可以通过VS的图形界面来实现，看起来挺方便的



但是默认只支持SqlServer连接



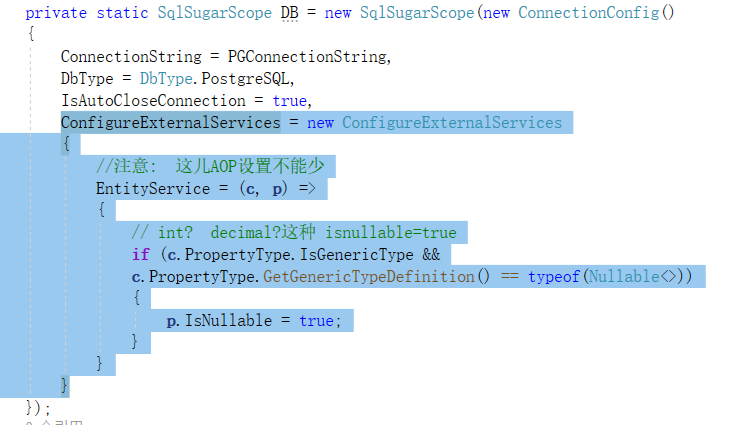
如果想支持其他数据库，需要找对应的插件扩展

# 数据库实体

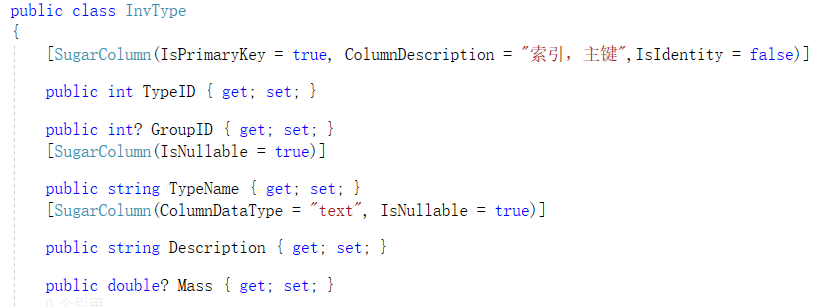
四个ORM在实体类写法上都是很简单的，什么多余的东西都不用写。思路也是极为相似，通过各种特性来定义表字段属性。需要注意一点，除开SqlSugar，其他三个在定义字段是否可为null时，只需在类型后面加个?就行

***public int? GroupID { get; set; }***

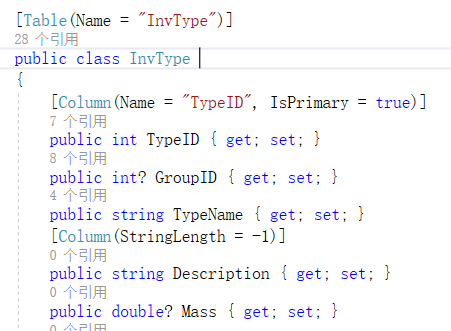
但是SqlSugar默认不支持，需要在配置数据库连接时额外加多几行代码。



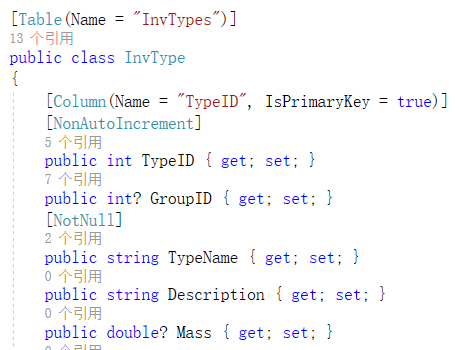
## SqlSugar



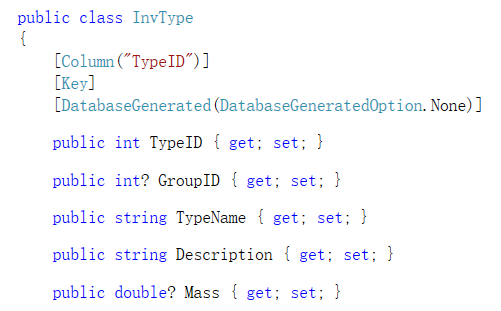
## FreeSql



## Chloe



## EF



# 单表CURD

只要不是DbHelper，用起来几乎都一样的简单好用，也都支持混合原生sql语句进行查询、事务等常见操作，但是EF的分页查询需要自己实现。

## SqlSugar

* ***DB.Queryable<InvType>().ToList();//简单查询所有***
* ***DB.Queryable<InvType>().Where(p => p.GroupID == 6).ToList();//where条件***
* ***DB.Queryable<InvType>().Where(p => p.GroupID == 6).Select(p => p.TypeID).ToList();//查询部分字段***
* ***DB.Queryable<InvType>()***

***.Where(p => p.GroupID == 6).Select(p => new { p.TypeID, p.GroupID }).ToList();//查询多个字段***

* ***DB.Queryable<InvType>().ToPageList(1,10).ToList();//分页***
* ***DB.Ado.SqlQuery<InvType>("SELECT \* FROM \"invtype\"");//原生Sql***
* ***DB.Insertable(invType).ExecuteCommand();//插入***
* ***DB.Updateable(tpye).ExecuteCommand();//更新***
* ***DB.Deleteable(tpye).ExecuteCommand();//删除***

## FreeSql

* ***Fsql.Select<InvType>().ToList();//简单查询所有***
* ***Fsql.Select<InvType>().Where(p=>p.GroupID == 6).ToList();//where条件***
* ***Fsql.Select<InvType>()***

***.Where(p => p.GroupID == 6).Select(p => new { p.TypeID, p.GroupID }).ToList();//查询部分字段***

* ***Fsql.Select<InvType>().Where(p=>p.TypeID > 1).Count(out var total).Page(1,2).ToList();//分页***
* ***Fsql.Select<InvType>()***

***.WithSql("SELECT \* FROM \"InvType\"").ToDataTable("\"TypeID\",\"GroupID\"");//原生Sql***

* ***Fsql.Insert(invType).ExecuteAffrows();//插入***
* ***Fsql.Update<InvType>(type).ExecuteAffrows();//更新***
* ***Fsql.Delete<InvType>(type).ExecuteAffrows();//删除***

## Chloe

* ***DBContext.Query<InvType>().ToList();//简单查询所有***
* ***DBContext.Query<InvType>().Where(p => p.GroupID == 6).ToList();//where条件***
* ***DBContext.Query<InvType>()***

***.Where(p => p.GroupID == 6).Select(p => new {p.TypeID,p.GroupID}).ToList();//查询部分字段***

* ***DBContext.Query<InvType>().OrderBy(p => p.TypeID).TakePage(1, 10).ToList();//分页***
* ***DBContext.SqlQuery<InvType>("SELECT \* FROM \"InvTypes\"");//原生sql***
* ***DBContext.Insert(invType);//插入***
* ***DBContext.Update(tpye);//更新***
* ***DBContext.Delete(tpye);//删除***

## EF

* ***db.InvTypes.ToList();//简单查询所有***
* ***db.InvTypes.Where(p => p.GroupID == 6).ToList();//where条件***
* ***db.InvTypes.Where(p => p.GroupID == 6).Select(p => new { p.TypeID,p.GroupID}).ToList();//查询部分字段***
* ***不支持其他三个那样的分页功能，官方文档建议是记录上一次的id来结合Where、Take手动分页***

***db.InvTypes.Where(p => p.TypeID > lastId).Take(10).ToList();***

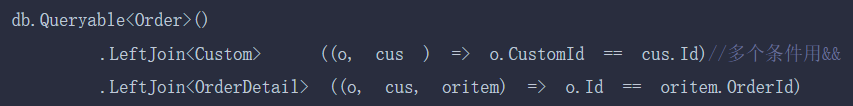
* ***db.InvTypes.FromSqlRaw("SELECT \* FROM \"InvTypes\"");//原生sql***
* ***db.InvTypes.Add(invType);*** ***db.SaveChanges(); //插入***
* ***修改实体后直接db.SaveChanges();即可更新***
* ***db.Remove(type); db.SaveChanges();//删除***

# 多表查询

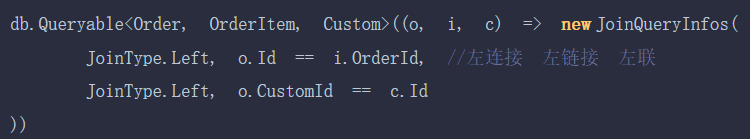
SqlSugar集大成，同时支持FreeSql、Chloe两种写方式，但是其实用起来区别不大，可能更多个表的时候Chloe这种一个一个一次拼起来的写起来可能有点累。

## SqlSugar

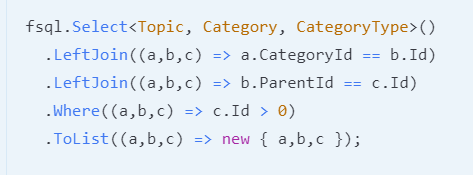
* 简单版本



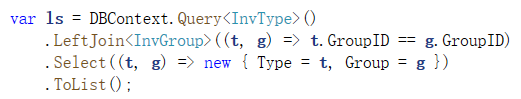
* 适合更多表的



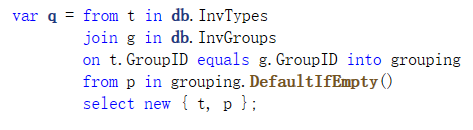
## FreeSql



## Chloe



## EF



# 批量插入

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | SqlSugar | FreeSql | Chloe | EF |
| 普通批量插入 | ✓ | ✓ | ✓ | ✓ |
| BulkCopy批量插入 | ✓ | 仅支持SqlServer/PostgreSQL/MySql | ✓ | ✓ |