ORM框架使用说明

**(V0.1.0)**

****

2018年5月

**目录**

1. 框架结构简介 4

1.1. 框架的基本约定 5

2. 数据库连接操作 6

2.1. 框架连接Oracle方式 6

2.2. 配置oracle连接 6

3. 使用框架进行数据操作说明 7

3.1. IDataSetAccessor接口使用说明 7

3.1.1. 查询单个实体 7

3.2. IDataAccessor接口使用说明 7

3.3. 多表联查使用说明 7

# 框架结构简介

ORM框架基本结构如下：



框架的核心是IDataSetAccessor和IDataAccessor，用户通过工厂方法获接口的默认实例并直接对数据进行CURD操作，框架可以自动完成数据库连接打开和关闭操作。二者的主要区别在于

IDataSetAccessor主要使用实体类(PO)信息拼接出SQL语句，在使用的时候只需要提供操作条件和条件对应的参数即可，最大化屏蔽数据库操作细节，适合对于PO对象直接的增删改查操作和对制定表格的增删改查操作。而IDataAccessor接收完整sql语句，并需要提供sql语句中的参数占位符与参数对的集(Dictionary)。该接口使用更加灵活，适合复杂数据库操作场景，用户对于sql操作十分熟悉，可以直接执行用户编写的sql语句。

## 框架的基本约定

框架编写的约定如下：

* 框架只面对oracle数据库设计
* 框架主要面向PO对象的CRUD操作
* 框架不提供多表关联查询，用户需要在DAO层解决关联查询（参考样例）
* 框架不提供事务操作，用户可根据需要，对框架进行扩展
* PO和数据库列名默认映射关系是数据库列名采用PO属性的全小写
* PO和数据库表明默认映射关系是数据库表名采用PO名称

框架提供的数据库类型转换包括：

|  |  |  |
| --- | --- | --- |
| .net类型 | oracle数据类型 | 说明 |
| Short | number |  |
| Int | number |  |
| Long | number |  |
| Float | number |  |
| Double | number |  |
| Char | varchar2 |  |
| String | varchar2 |  |
| bool | int | oracle没有bool类型，框架默认采用0表示false，1表示true |
| Decimal | Decimal | 待测试 |
| DateTime | DateTime | 待测试 |

# 数据库连接操作

## 框架连接Oracle方式

目前了解C#中连接Oracle数据库的方法有3种，分布是微软的System.Data.OracleClient，Oracle的Oracle.DataAccess.Client和Oracle的Oracle.ManagedDataAccess.dll(最优)，本框架采用第三种方式操作oracle数据库。

操作连接方法在OracleConnectionFactory列中，该类提供如下方法：

**public static OracleConnection OpenConn()**

每次操作的时候，都会通过连接工厂获取一个oracle数据库连接。

**public static void CloseConn(OracleConnection conn)**

工厂类提供了关闭连接的方法，框架每次完成操作后，会自动释放连接。

## 配置oracle连接

配置oracle数据库连接在OracleConnectionFactory类中，目前采用硬编码方式，后续可以采用读取配置方式。

连接配置样例如下：

ConnectionString ="DataSource=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=**10.2.32.189**)(PORT=**1521**))(CONNECT\_DATA=(SERVICE\_NAME=**oracle.db.server**)));Persist Security Info=True;User ID=**crscd**;Password=**crscd123@**;"

使用中只需要根据实际情况配置上述字符串中红色标记的部分即可。

# 使用框架进行数据操作说明

## IDataSetAccessor接口使用说明

IDataSetAccessor接口中定义的方法如下：

// 查询对应实体，返回单条记录

T Select<T>(string condition, params object[] paramList) where T : class;

// 查询实体列表

List<T> SelectList<T>(string condition, params object[] paramList) where T : class;

// 插入实体

bool Insert<T>(T t) where T : class;

// 更新实体

bool Update<T>(T t, string condition, params object[] paramList) where T : class;

// 删除实体

bool Del<T>(string condition, params object[] paramList) where T : class;

// 查询实体数量

long GetCount<T>(string condition, params object[] paramList) where T : class;

// 查询单列数据

List<T> GetColumnList<T>(string table, string column, string condition, params object[] paramList);

// 向指定表格插入指定列数据

bool InsertColumnData(string table, Dictionary<string, object> columnDatMap);

// 向指定表格更新指定列的数据

bool UpdateColumnData(string table, Dictionary<string, object> columnDatMap, string condition, params object[] paramList);

// 在指定表格中删除满足条件的数据

bool DelData(string table, string condition, params object[] paramList);

### 查询单个实体

查询单个实体需要指定查询条件和条件中的参数，首先在数据库中建立一个表格person，其中包含三列，id（number），name(varchar2)，note(varchar2)，然后定义一个实体类(PO)Person：

public class Person

{

public int Id { set; get; }

public string Name { set; get; }

public string Note { set; get; }

}

在数据库中存在一个id为1的数据，则从数据库中查询出来person的流程如下：

var condition = "id=?";

var person = dataSet.Select<Person>(condition,1);

用户只需要定义查询条件，并在查询的时候按照顺序制定参数值就可以完成查询过程，将符合条件的一条数据查出。

条件的定义约定为:

**[列名(默认为属性的小写)]=？ AND [[列名(默认为属性的小写)]=？]… [约束条件]**

查询语句中的参数采用可变参数方式，当有多个参数的时候，需要按照条件中定义的顺序，将参数写入：

dataSet.Select<Person>(condition,parm1,parm2,…);

**注意：**如果数据库中不存在符合条件的数据，则会返回null。

当存在多个符合条件的实例时，框架会默认返回从数据库查出的第一行数据。

### 查询一组实例

查询一组实例的操作与查询一个实例的操作类似，流程如下：

var condition = "note like ?%";

var person = dataSet.Select<Person>(condition,test);

上述操作会将符合条件的一组信息都查询出来并存放到List中返回。

注意：如果数据库中不存在符合条件的数据，则会返回一个空的List。

### 插入一条实例

### 更新一个实例

### 删除一条实例

### 查询实体数量

### 查询制定表格中单个列的数据

## IDataAccessor接口使用说明

## 多表联查使用说明