

RDIFramework. NET

平台代码生成器 使用教程

国思软件

目录

1. 前言	4
1.1. 系统菜单	6
1.2. 工具栏	6
1.3. 数据库视图导航区	6
1.4. PowerDesigner 设计文档对象导航区	7
1.5. 业务工作区	8
1.6. 系统状态栏	8
1.7. 起始页	8
1.8. 关于	9
2. 数据库视图导航区	10
2.1. 数据库服务器管理	10
2.1.1. 添加服务器	10
2.1.2. 连接服务器	12
2.1.3. 断开与注销服务器连接	13
2.1.4. 备份服务器配置	13
2.1.5. 导入服务器配置	14
2.2. 数据库管理	14
2.2.1. 浏览数据库	15
2.2.2. 新建查询	17
2.2.3. 生成存储过程	18
2.2.4. 生成数据脚本	21
2.2.5. 导出文件（存储过程与数据脚本）	22
2.3. 表管理	23
2.3.1. 生成 SQL 语句	23
2.3.2. 浏览表数据	25
2.3.3. 生成数据脚本	26
2.3.4. 生成存储过程	27
2.3.5. 导出文件（存储过程与数据脚本）	28
2.3.6. 生成业务代码（重量级）	28
2.3.6.1. 项目属性设置	29
2.3.6.2. 数据表的定义展示	30
2.3.6.3. 表的 DDL	31
2.3.6.4. 业务实体（Entity）	31
2.3.6.5. 契约服务接口	32
2.3.6.6. 契约服务实现	32
2.3.6.7. 生成数据库脚本	33
2.3.6.8. 文档	33
2.3.6.9. 辅助功能	34
2.3.7. 生成敏捷 MvcUI 界面	35
2.3.7.1. Index.cshtml 页面代码	35
2.3.7.2. 编辑界面 Form.cshtml 代码	35
2.3.7.3. Controller 控制器	36

2.3.8. 生成敏捷 CoreMvcUI 界面	36
2.3.8.1. Index.cshtml 页面代码	37
2.3.8.2. 编辑界面 Form.cshtml 代码	37
2.3.8.3. Controller 控制器	38
2.3.9. 生成 WinForm 界面	39
2.3.10. 生成 MvcEasyUI 界面	42
2.3.10.1. Index.cshtml 页面代码	42
2.3.10.2. 编辑界面 Form.cshtml 代码	43
2.3.10.3. Controller 控制器	43
2.3.10.4. js 页面代码	44
2.4. 视图管理	44
2.4.1. 生成脚本	45
2.4.2. 对象定义	46
2.4.3. 浏览表数据	46
2.5. 2.5 存储过程管理	46
2.5.1. 脚本生成	47
2.5.2. 对象定义	47
3. PowerDesigner Objects 导航区	47
3.1. PD 设计文件管理	49
3.1.1. 添加 PD 设计文件	49
3.1.2. 移除 PD 设计文件	49
3.1.3. 展开 PD 设计文件	50
3.2. 代码生成	50
3.3.1. 生成类数据表	51
3.3.2. 生成业务实体（Entity）	51
3.3.3. 生成 MVC 业务实体（MVCEntity）	51
3.3.4. 生成契约服务接口	52
3.3.5. 生成契约服务	52
3.3.6. 生成服务管理器	53
3.3.7. 生成全部代码	53
3.3.8. 生成表设计文档	54
3.3.9. 生成数据库脚本	55
4. 代码批量生成	55
4.1. 基于数据库的代码批量生成	55
4.2. 数据库设计文档生成	57
4.3. 基于 PowerDesigner 设计文件的代码批量生成	59
Q Q: 406590790	61
电话: 13005007127 (同微信)	61

1. 前言

RDIFramework.NET 代码生成器集代码生成、各数据库对象文档生成、数据库常用管理于一身，是软件开发者快速开发的神器。代码的生成支持基于数据库与 PowerDesign 设计文件生成，为企业及个人开发者能快速生成常用代码与文档以达到快速开发，快速应用的目的，节省开发成本。

基于数据库的代码生成不仅可直接在生成器上做数据库 Sql 级的操作，如常用的表、视图、存储过程、函数的创建、查询、修改、删除、生成数据脚本、生成数据库设计文档、表设计文档、生成业务逻辑代码、生成 Web、WinForm 界面代码等。便捷的数据库管理功能，轻松操作数据库。完全不用打开数据库企业管理工具即可完成相应的数据库层面的操作，支持 SqlServer、Oracle、MySQL 版本，其他类型的数据库也在陆续支持中。

基于 PowerDesigner 设计文件的代码生成，是我们代码生成器独居的特色，开发人员不用关心是什么类型的数据库，即可完美的生成项目代码。建议在开发过程中使用 PowerDesigner 设计工具完成数据库的建模，PD 是非常优秀的数据库建模工具，熟练的使用该工具对系统的分析和设计有很大的帮助。可以用 PowerDesigner 做需求分析，各种模型之间的转换，设计完成还可自动把数据库生成出来。在设计阶段我们只需把精力集中在模型上。

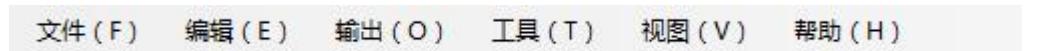


系统主界面



在 RDIFramework.NET 代码生成器主界面中，共包括 6 个不同的工作区域：系统菜单、工具栏、数据库视图导航区、PowerDesigner 设计文档对象导航区，业务工作区、系统状态栏。

1.1. 系统菜单



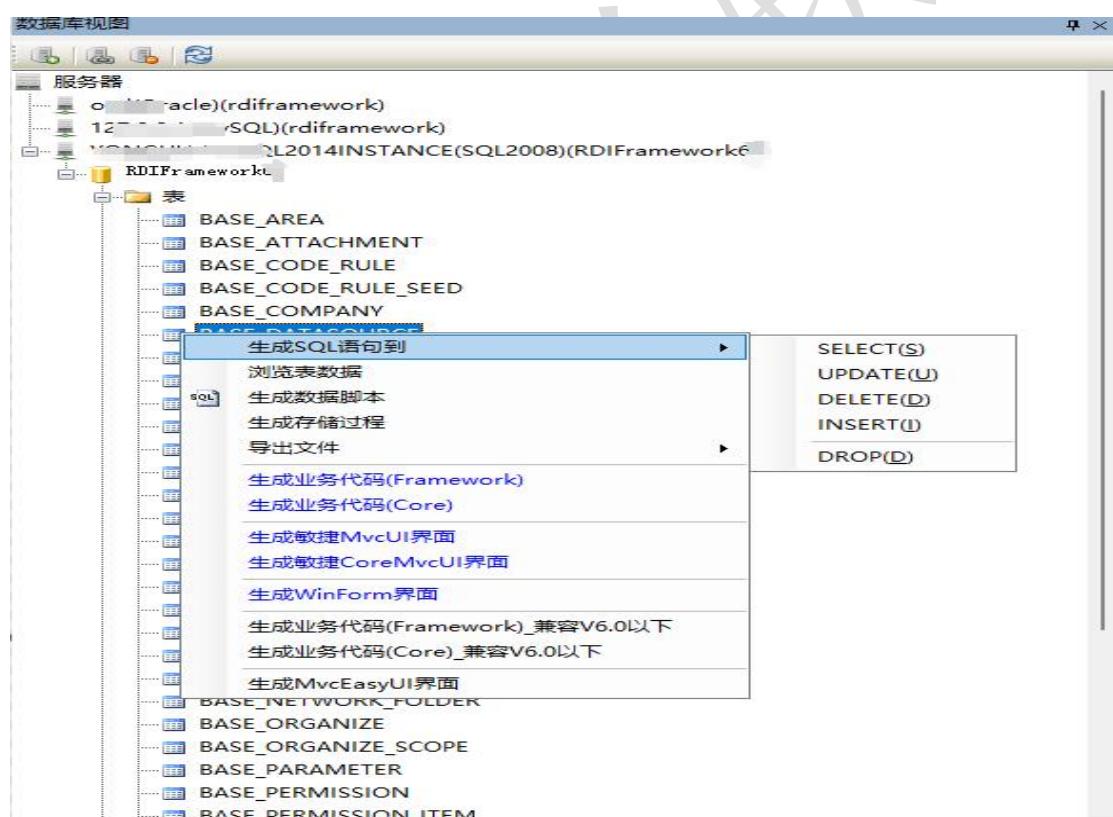
RDIFramework.NET 代码生成器系统菜单由文件、编辑、输出、工具、视图、帮助等菜单组成。选择相应的功能项，系统菜单会有相应的变化。

1.2. 工具栏



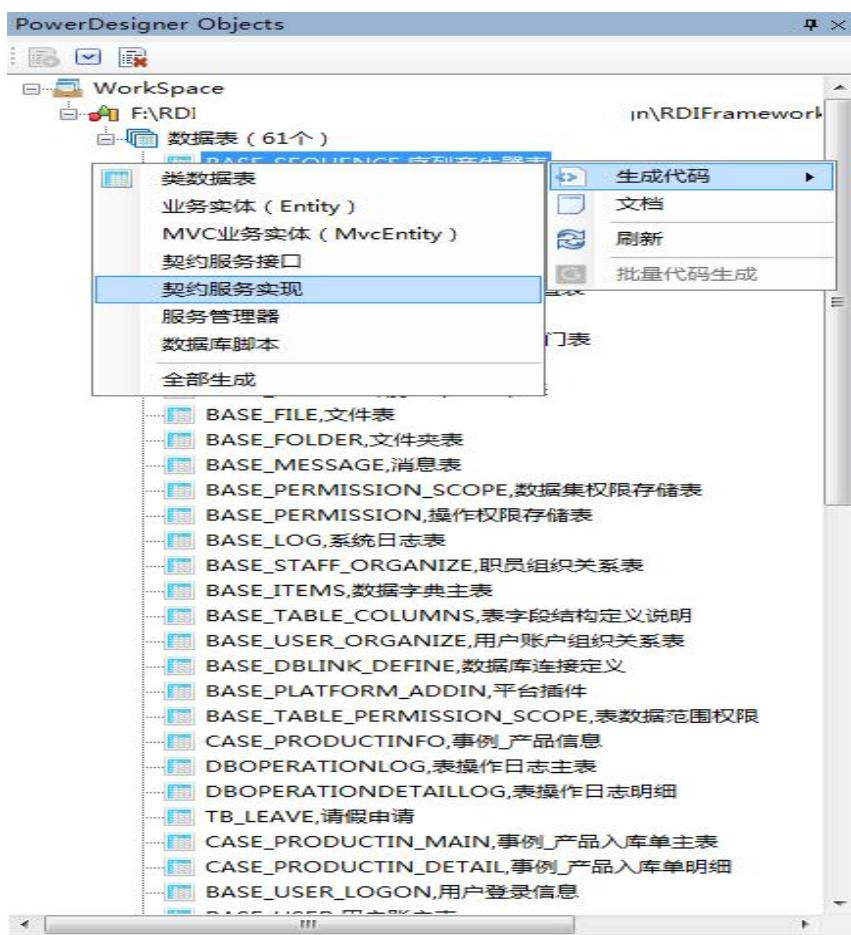
工具栏放置了与系统菜单相当的常用功能模块的快捷操作方式。

1.3. 数据库视图导航区



数据库视图导航区是整个代码生成器主要的工作导航区域，在这儿，我们可以通过右键快捷菜单来进行常用的操作，如：查询数据、修改数据、删除数据、删除表、浏览表的数据、生成表的数据脚本（表数据导出）、生成表的存储过程（主要用于业务代码的数据访问层采用存储过程的方式）、导出数据库对象为常用文件、以及非常强大的基于框架的代码生成主要操作界面（生成业务代码，支持 core 与 framework）、针对项目生成界面 UI 代码。

1.4. PowerDesigner 设计文档对象导航区



在 PowerDesigner 设计文档对象导航区，我们可以导入多个数据库物理设计模型，以通过数据库的设计模型来完成代码及相应文档的生成，通过 PowerDesigner 设计文档，生成的代码与目标数据库的类型无关。

1.5. 业务工作区

序号	属性名称	类型	标题	列名	数据类型	长度	精度	小数位	非空	主键	自增	默认值
1	ID	string	主键	ID	char	40	0	0	是	是		
2	PARENT_ID	string	父节点主键	PARENT_ID	char	40	0	0	否	否		
3	SUB_SYSTEM_ID	string	子系统ID	SUB_SYSTEM_ID	char	40	0	0	否	否		
4	CODE	string	编码	CODE	char	50	0	0	否	否		
5	FULL_NAME	string	名称	FULL_NAME	char	200	0	0	否	否		
6	CATEGORY	string	类型分类	CATEGORY	char	50	0	0	否	否		(N/A)
7	MODULE_TYPE	int	模块类型 (1: WinForm, 2: WebForm)	MODULE_TYPE	int	4	10	0	否	否		(3)
8	IMAGE_INDEX	string	图标索引	IMAGE_INDEX	char	40	0	0	否	否		
9	SELECTED_IMAGE_ID	string	选中状态图标编号	SELECTED_IMAGE_ID	char	40	0	0	否	否		
10	ICON_CSS	string	—CSS样式类显示	ICON_CSS	char	255	0	0	否	否		
11	ICON_URL	string	图标地址 (针对Web)	ICON_URL	char	255	0	0	否	否		
12	NAVIGATE_URL	string	导航地址	NAVIGATE_URL	char	255	0	0	否	否		(N/A)
13	MVC_NAVIGATE_URL	string	MVC导航地址	MVC_NAVIGATE_URL	char	255	0	0	否	否		(N/A)
14	TARGET	string	目标 (针对WinForm)	TARGET	char	100	0	0	否	否		
15	FORM_NAME	string	窗体名称 (针对WinForm)	FORM_NAME	char	100	0	0	否	否		
16	FORM_PARAMETER	string	参数参数 (针对WinForm)	FORM_PARAMETER	char	400	0	0	否	否		
17	ASSEMBLY_NAME	string	组件类名 (针对WCF)	ASSEMBLY_NAME	char	100	0	0	否	否		
18	PERMISSION_ITEM	string	操作权限 (针对Item)	PERMISSION_ITEM	char	50	0	0	否	否		(N/A)
19	PERMISSION_SCOPE	string	策略数据权限 (针对Scope)	PERMISSION_SCOPE	char	255	0	0	否	否		
20	IS_PUBLIC	byte	是否公开	IS_PUBLIC	tinyint	1	3	0	否	否		(0)
21	IS_MENU	byte	菜单项	IS_MENU	tinyint	1	3	0	否	否		(0)
22	EXPAND	byte	展开状态	EXPAND	tinyint	1	3	0	否	否		(0)
23	ALLOW_EDIT	byte	允许编辑	ALLOW_EDIT	tinyint	1	3	0	否	否		(1)
24	ALLOW_DELETE	byte	允许删除	ALLOW_DELETE	tinyint	1	3	0	否	否		(1)
25	IS_SCOPE	byte	数据权限 (需要设置...)	IS_SCOPE	tinyint	1	3	0	否	否		(0)
26	SORT_CODE	int	排序码	SORT_CODE	int	4	10	0	否	否		
27	ENABLED	byte	是否启用	ENABLED	tinyint	1	3	0	否	否		(1)
28	DESCRIPTION	string	备注	DESCRIPTION	char	255	0	0	否	否		
29	CREATE_ON	string	创建时间	CREATE_ON	char	8	27	7	否	否		
30	CREATE_USER_ID	string	创建用户主键	CREATE_USER_ID	char	40	0	0	否	否		
31	CREATE_BY	string	创建者	CREATE_BY	char	50	0	0	否	否		
32	MODIFIED_ON	string	修改时间	MODIFIED_ON	char	8	27	7	否	否		
33	MODIFIED_USER_ID	string	修改用户主键	MODIFIED_USER_ID	char	40	0	0	否	否		

业务工作区是整个代码生成器的核心工作区域，该区域会根据用户相应的操作来加载不同的功能模块或窗体界面以完成用户对应的操作。

1.6. 系统状态栏



系统状态栏上显示了系统当前的处理任务及任务的处理状态，用于通用用户，以便及时了解系统的运行情况。

1.7. 起始页

起始页显示了常用的功能操作快捷方式与关于代码生成器的最新信息与作者的最新信息，通过这儿，可以了解到关于框架或代码生成器的最新情况，如下图所示。

常用操作

- 新增数据库服务器
- 生成代码项目
- 对象资源管理器
- 代码生成器
- SQL 数据库脚本生成器
- 代码批量生成器
- 数据库文档生成器
- 项目属性设置

最新信息

国思RDIF-Wms仓库管理系统助力企业高效数字化（源码交付）

国思RDIF-Wms是一款基于国思RDIF低代码快速开发框架下的仓库管理系统，不仅涵盖了仓库/库区/货架管理，出入库管理，客户/供应商/承运商，库存看板，库存预警与到期提醒，基础数据报表展示等功能，更整合了增值服务、多租户支持、用户权限管理以及各基础数据的维护。从而进一步提升业务执行效率，实现仓库的智能化管理，同时还全源码提供，方便企业扩展，加速落地应用，方便满足不同行业仓库的管理需求。

文章标题	发布时间
国思RDIF-Wms仓库管理系统助力企业高效数字化（源码交付）	2024-06-28 10:02:27
【干货】Vue3 组件通信方式详解	2024-06-26 09:51:10
玩转数据库索引	2024-06-25 10:25:41
一文带你理清同源和跨域	2024-06-20 10:05:15
【长文】带你搞明白Redis	2024-06-17 10:25:38
国思RDIF.vNext全新低代码快速开发框架平台6.1版本发布（支持vue2、vue3）	2024-06-13 10:05:29
解决DevExpress用DevExpress patch工具破解后还能试用框的问题	2024-04-13 10:51:02
恭喜：创造历史，C#正式荣登TIOBE 2023年年度编程语言奖	2024-01-08 10:55:02
.NET敏捷开发框架-RDIFramework.NET V6.0发布	2023-08-25 14:11:43
RDIFramework.NET 快速开发框架 WebEasyUI版本 V6.0发布	2023-07-27 09:16:53
RDIFramework.NET CS敏捷开发框架 V6.0发布(支持.NET6+、Framework双引擎，全网唯一)	2023-07-25 11:03:16
你必知道的 Chrome 前端调试技巧	2023-07-13 11:37:29
VS依赖注入(DI)构造函数自动生成局部私有变量	2023-06-26 11:35:40
【保姆级教程】Vue 项目调试技巧	2023-06-20 08:56:46
Vue 前端开发团队风格指南（史上最全）	2023-05-09 14:25:22
【干货】Vue2.x 组件通信方式详解，这篇讲全了	2023-04-27 11:34:01
工作中要使用Git，看这篇文章就够了	2023-04-25 09:29:38
企业数字化转型如何做？看过来	2023-04-20 09:00:00
window系统git使用ssh方式和gitee进行同步	2023-04-19 08:30:00
DevExpress破解后运行经常弹出试用框的问题处理方式	2023-04-19 08:00:00

1.8. 关于

关于本软件

RDIFramework.NET
企业级.NET快速开发平台(十年专注、精益求精)

海南国思软件科技有限公司 版权所有
Copyright (C) 2011-2024. All Rights Reserved.

程序名称：RDIFramework.NET平台代码生成器 V6.1 -
RDIFramework.CodeMaker程序版本：6.1.9070.26340 (Build: 2024-10-31 02:38:03) 程序标识：5c4c79f5-c95c-4859-af10-905e8092b46b
程序说明：
RDIFramework.NET平台代码生成器，代码、文档一键生成。为企业和个人开发者创造更多价值。本版本支持RDIFramework.NET V6.1+。作者：
RDIFramework.NET (海南国思软件科技有限公司) 主页：
<http://www.rdiframework.net/> 计算机用户名：40659 @ YONGHU-PC 操作系统：
Microsoft Windows NT 6.2.9200.0 程序文件名称：D:\CodeMaker\RDIFramework.CodeMaker.exe 程序内存使用：146,206,720 bytes

想了解更多信息？请访问：
<http://www.rdiframework.net/> <http://www.rdiframework.net/article> [QQ交谈](#) [确定 \(0\)](#)

提供对当前程序相关的描述信息，如程序的名称、版本、说明、作者、操作系统的相关系统等。同时，你还可以通过下面的网址了解最新的信息，或直接点击“QQ交流”按钮与我们QQ进行沟通。

2. 数据库视图导航区

数据库视图导航区主要完成对数据库(包括 MSSQLSERVER、ORACLE、ACCESS、MYSQL等)及数据库对象的管理, 包括:

- 1) 、数据库服务器管理。
- 2) 、数据库管理。
- 3) 、数据库对象的管理。

2.1. 数据库服务器管理

数据库服务器管理主要是对建立、移除、备份、导入数据库服务的连接, 如下图所示。



选择“服务器”根节点, 点击鼠标右键可以添加服务器、备份服务器配置、导入已存在的服务器配置以及刷新。

2.1.1. 添加服务器

在上图中, 我们选择“添加服务器”, 弹出“数据库类型选择”窗口, 如下图所示, 可以选择“SQLSERVER”、“ORACLE”、“MYSQL”、“OLEDB”、“SQLLITE”几种数据库类型, 代码生成器对数据库当前仅支持 SQLSERVER、ORACLE, 在后期的版本中会陆续对其他数据库提供支持, 对于其他类型数据库的代码生成, 可以通过 PowerDesigner 设计源文件进行生成。

一、连接到 SQLServer。



点击下一步，打开“连接到服务器”对话框，如下图所示。



在上图中，输入相关连接信息，即可连接到 SQLSERVER 数据库。

其中：

服务器名称：SQLSERVER 实例所在的服务名称（可为名称或 IP 地址）。

服务器类型：SQLSERVER 实例的数据库类型（SQLSERVER2000、SQLSERVER2005、SQLSERVER2008、SQLSERVER2012、SQLSERVER2014 等）。

身份验证：连接到 SQLSERVER 的身份认证方式，分为“SQL SERVER 身份认证”和“WINDOWS 身份认证”两种方式。

登录名：登录到 SQLSERVER 的用户名。

密码：登录到 SQLSERVER 的用户密码。

全部输入后，单击“连接/测试”，可以测试是否能正确连接到 SQLSERVER 数据库，如果测试成功，即可在数据库列表中列出当前 SQLSERVER 实例的所有数据库列表。如果选择“全部数据库”，则会加载所有数据库到“数据库视图”，如果当前实例，数据库过多，我们可以选择一个特定的需操作的数据库加载即可，这样可以提高加载的速度。

二、连接到 ORACLE。



点击下一步，打开“登录”到 ORACLE 对话框，如下图所示。



输入正确的连接到 ORACLE 的相关信息后，即可成功连接到 ORACLE。

三、连接到 MySql。



点击下一步，打开“登录”到 MySql 对话框，如下图所示。

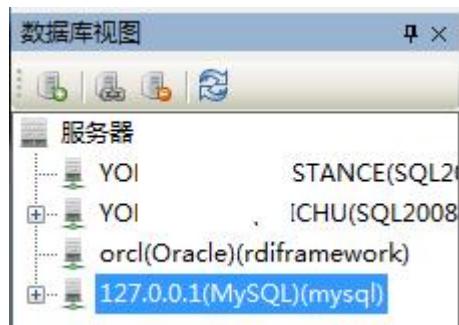


输入正确的连接到 MySql 的相关信息后，即可成功连接到 MySql。

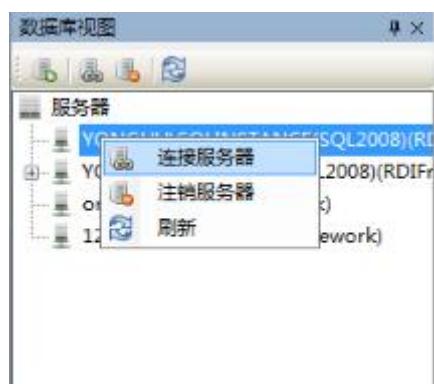
2.1.2. 连接服务器

新增服务器注册成功后，我们就可以连接到新增的服务器上，有两种方式连接到成功注册的服务器上。

方法一：通过数据库视图的工具栏中的“连接到服务器”功能按钮进行连接。



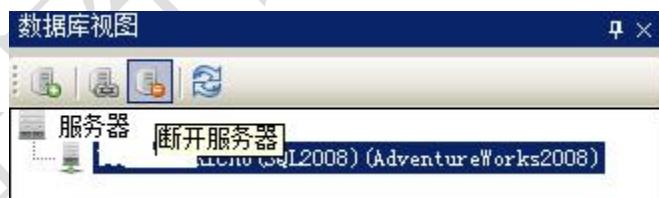
方法二：通过右键功能菜单。



通过上面的两种方式可以建立与注册服务器的连接。

2.1.3. 断开与注销服务器连接

如果已注册的服务器我们不再需要，可以对其进行注销，即删除其注册连接。对于已经成功连接的服务器，我们也可以断开其连接，断开服务器连接如下图所示。

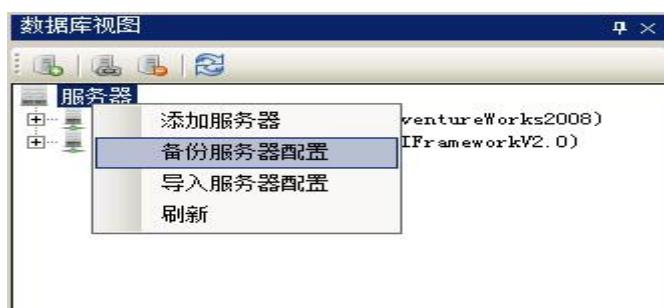


注销服务器连接，如下图所示。

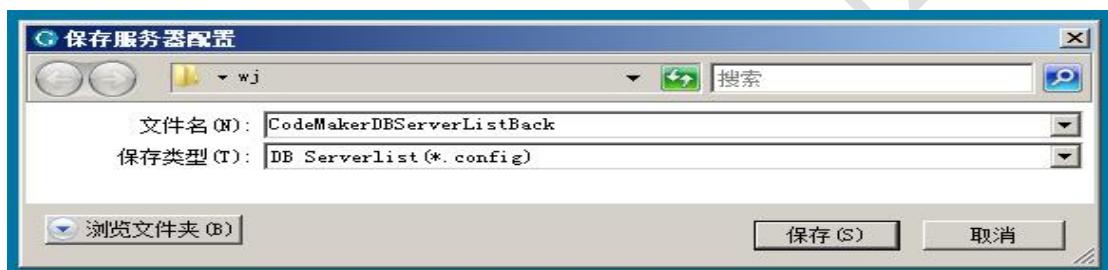


2.1.4. 备份服务器配置

我们可以把当前所有的服务器配置备份到我的电脑，以作他用，要“备份服务器配置”，需要用鼠标右击数据库树视图的“服务器”根节点，如下图所示。



选择“备份服务器配置”，弹出“保存服务器配置”对话框，输入保存的文件名称与保存的目录后，即可对当前所有的服务器连接进行备份。



2.1.5. 导入服务器配置

“导入服务器配置”是“备份服务器配置”的逆操作，操作方式与“备份服务器配置”相当，如下图所示。



2.2. 数据库管理

连接到一个已注册的服务器，我们可以对当前服务器所选“数据库”进行相应的操作，主要包括：浏览数据库、新建查询、生成存储过程、生成数据脚本、导入文件（存储过程与数据脚本）等。



2.2.1. 浏览数据库

浏览数据库主要包括浏览当前所选数据库的所有对象（表、视图、存储过程等），当前所选表、视图、存储过程的详细信息。

◆ 浏览数据库的详细信息

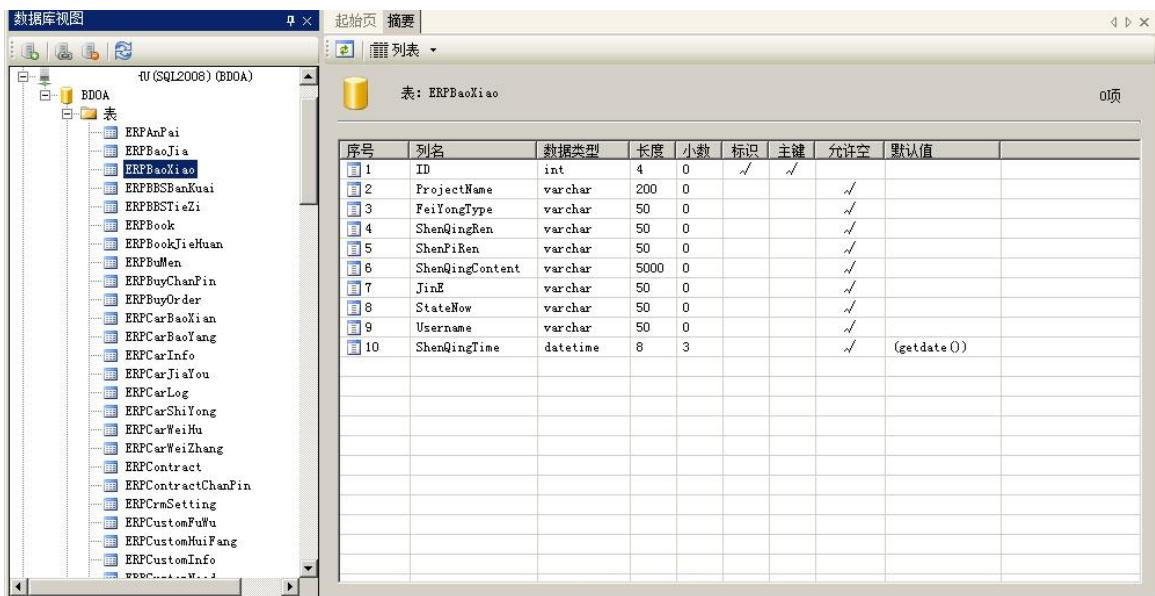
用鼠标选择数据库树节点，如选择“RDIFRAMEWOKR”数据库，可以显示

“RDIFRAMEWOKR”数据库（表、视图、存储过程等）的详细信息，如下图所示。

名称	所有者	类型	创建日期
P_IN_CASE_PRODUCTINFO	dbo	存储过程	2020-11-20 09:42:18
P_JOB_ENTRY	dbo	存储过程	2020-11-20 09:42:18
P_LOCKINFO	dbo	存储过程	2020-11-20 09:42:18
pGetPageData	dbo	存储过程	2020-11-20 09:42:18
pGetRecordByPage	dbo	存储过程	2020-11-20 09:42:18
pGetTableInfo	dbo	存储过程	2020-11-20 09:42:18
pPaging	dbo	存储过程	2020-11-20 09:42:18
pStringSearch	dbo	存储过程	2020-11-20 09:42:18
BASE_AREA	dbo	用户	2020-11-20 09:41:06
BASE_ATTACHMENT	dbo	用户	2020-11-20 09:40:21
BASE_AUTOTEXT_SQL	dbo	用户	2020-11-20 09:40:22
BASE_CODE_RULE	dbo	用户	2021-03-24 11:29:48
BASE_CODE_RULE_SEED	dbo	用户	2021-03-24 11:29:57
BASE_COMPANY	dbo	用户	2020-11-20 09:40:22
BASE_DATASOURCE	dbo	用户	2020-11-20 09:40:22
BASE_DBLINK_DEFINE	dbo	用户	2020-11-20 09:40:23
BASE_EXCEPTION	dbo	用户	2020-11-20 09:40:23
BASE_FILE	dbo	用户	2020-11-20 09:40:23
BASE_FOLDER	dbo	用户	2020-11-20 09:40:24
BASE_FORM_MODULE_RELATION	dbo	用户	2021-05-06 10:48:43
BASE_FORM_TEMPLATE	dbo	用户	2021-05-06 10:48:27
BASE_FORM_TEMPLATE_INFO	dbo	用户	2021-05-06 10:48:35
BASE_ITEMS_DETAIL	dbo	用户	2020-11-20 09:40:24

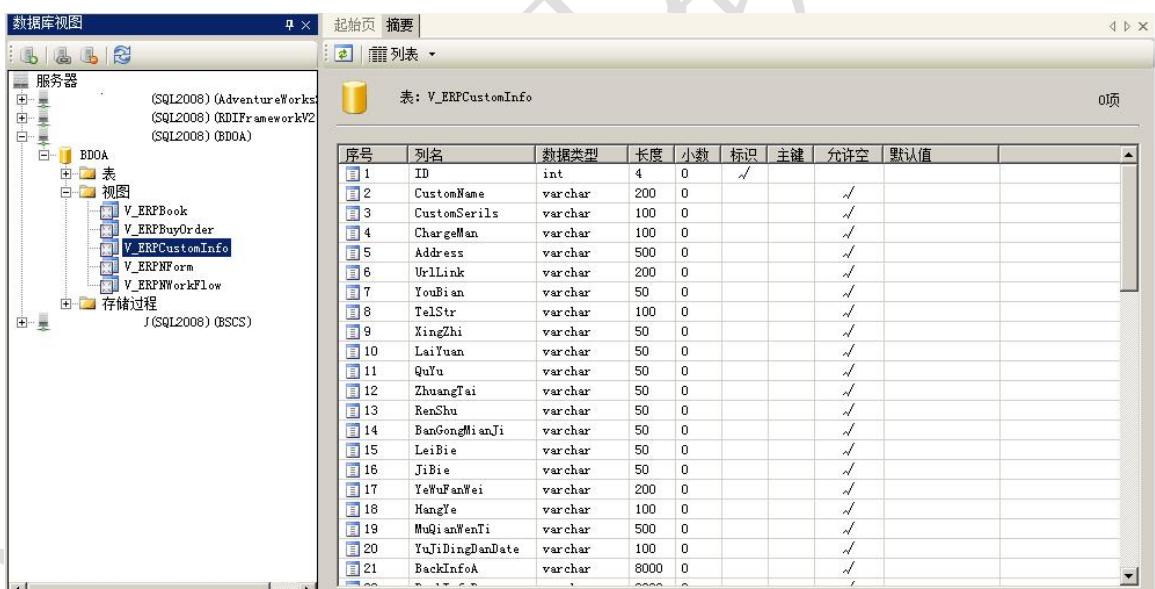
◆ 浏览表的详细信息

用鼠标选择“表”树节点中的任意一个表，我们可以查看该表各字段的详细信息（表名称、数据类型、长度、小数位数、是否是标识列、是否是主键列、是否允许为空、列的默认值等），如下图所示。



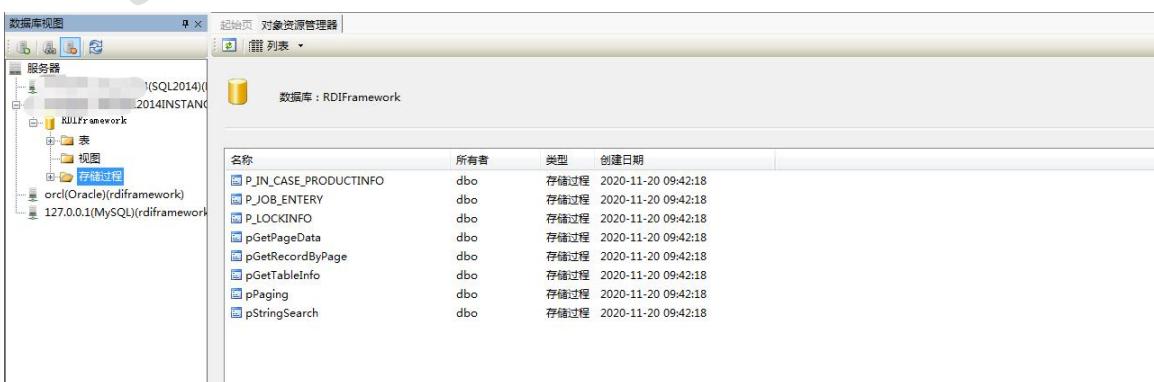
◆ 浏览视图的详细信息

浏览视图的详细信息与表的详细信息差不多，选中一个视图，如下图所示。



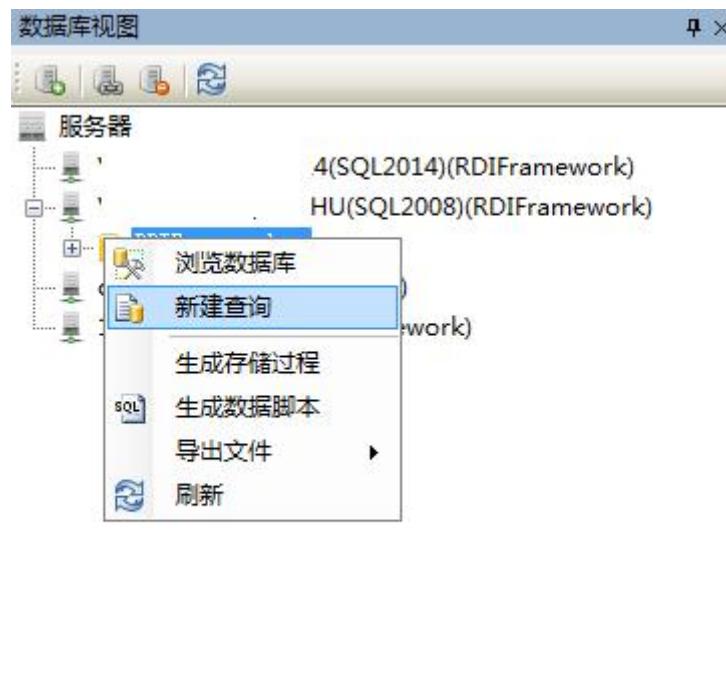
◆ 浏览存储过程的详细信息

选择“存储过程”对节点，显示当前库的所有存储过程信息，如下图所示。



2.2.2. 新建查询

新建查询类似类数据库的查询分析器，我们可以在查询窗体输入正确的 SQL 语句直接进入 SQL 语句操作，需要说明的是：相关的 SQL 语句的执行权限与当前注册服务器的用户名有关。选择某个数据库后，单击鼠标右键，选择“新建查询”，即可打开查询窗口，如下图所示。



单击“新建查询”后，打开当前数据库的查询窗口，如下图所示。

The screenshot shows the RDIFramework.NET query window. The top part displays a SQL script for querying the 'BASE_MODULE' table:

```

select
    [ID],
    [PARENTID],
    [SUBSYSTEMID],
    [CODE],
    [FULLNAME],
    [CATEGORY],
    [MODULETYPE],
    [IMAGEINDEX],
    [SELECTEDIMAGEINDEX],
    [ICONSS],
    [ICONURL],
    [NAVIGATEURL],
    [MVCAVIGATEURL],
    [TARGET],
    [FORMNAME],
    [FORMPARAMETER],
    [ASSEMBLYNAME],
    [PERMISSIONITEMCODE],
    [PERMISSIONSCOPEABLES],
    [IS_SCOPE],
    [ISPUBLIC],
    [ISMENU],
    ...

```

A context menu is open over the script, with options like 'Copy', 'Cut', 'Paste', 'Select All', 'Run Current Query', 'Generate Current Query SQL Statement', 'Generate Current Query Result Script', and 'Script'.

The bottom part of the window shows the execution results of the query, displaying a table with columns: ID, PARENTID, SUBSYSTEMID, CODE, FULLNAME, CATEGORY, MODULETYPE, IMAGEINDEX, SELECTEDIMAG, ICONSS, and IS. The table contains several rows of data, such as CompanyAdmin, OrganizeAdmin, StaffAdmin, UserAdmin, and RoleAdmin.

在“查询”窗口，我们可以输入任意有效的 SQL 语句进行执行，在上图中，我们输入了一条查询数据的 SQL 语句，下窗口下方可以看到其执行结果。输入完 sql 语句后，我们可以通过三种方式进行执行：

- a) 按“F5”功能键执行。
- b) 通过右键菜单，选择“运行当前查询”来执行。
- c) 通过工具栏的“执行 SQL”按钮来执行。

注：如果输入的是查询语句，建议不要一次查询过多数据，可以加入 **TOP n** 关键字，对输出的数据条数加以限制。

2.2.3. 生成存储过程

通过右键“数据库”节点，选择“生成存储过程”功能，即可快速当前数据库所有表的业务逻辑存储过程（生成后，用户可根据需要进行相应的修改），这对倾向于存储过程的开发者大大提高了其写存储过程的效率，缩短开发时间，节约开发成本。



根据数据库表的多少，会花一些时间来进行生成，生成后的存储过程如下图所示。

```


-- 表名: BASE_AREA
-- 时间: 2023-07-13 09:58:20
-- Made by RDIFramework.NET 平台代码生成器

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[UP_BASE_AREA_Exists]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[UP_BASE_AREA_Exists]
GO
-----用途: 是否已经存在
-----项目名称: demo
-----说明:
-----时间: 2023-07-13 09:58:20
CREATE PROCEDURE UP_BASE_AREA_Exists
@ID varchar(40)
AS
DECLARE @TempID int
SELECT @TempID = count(i) FROM [E
IF @TempID = 0
    RETURN 0
ELSE
    RETURN 1
GO

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[UP_BASE_AREA_ADD]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[UP_BASE_AREA_ADD]
GO
-----用途: 增加一条记录
-----项目名称: demo
-----说明:
    

```

默认对每个表会生成以下几类存储过程：

- a) 得到主键字段的最大值（过程命名方式为：表名+“_GetMaxId”）。

```

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[LinkMan_GetMaxId]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[LinkMan_GetMaxId]
GO
-----
--用途：得到主键字段最大值
--项目名称：
--说明：
--时间：2013-07-29 14:35:30
CREATE PROCEDURE LinkMan_GetMaxId
AS
>     DECLARE @TempID int
>     SELECT @TempID = max([Id])+1 FROM [LinkMan]
>     IF @TempID IS NULL
>         RETURN 1
>     ELSE
>         RETURN @TempID
GO

```

b) 数据存在性判断（过程命名方式为：表名+“_Exists”）。

```

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[LinkMan_Exists]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[LinkMan_Exists]
GO
-----
--用途：是否已经存在
--项目名称：
--说明：
--时间：2013-07-29 14:35:30
CREATE PROCEDURE LinkMan_Exists
@Id int
AS
>     DECLARE @TempID int
>     SELECT @TempID = count(1) FROM [LinkMan] WHERE Id=@Id
>     IF @TempID = 0
>         RETURN 0
>     ELSE
>         RETURN 1
GO

```

c) 新增数据（过程命名方式为：表名+“_Add”）。

```

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[LinkMan_ADD]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[LinkMan_ADD]
GO
-----
--用途：增加一条记录
--项目名称：
--说明：
--时间：2013-07-29 14:35:30
CREATE PROCEDURE LinkMan_ADD
@Id int output,@CustomerId int,@Name nvarchar(50),@Sex nvarchar(5),
@Postion nvarchar(50),@Department nvarchar(50),@MainLinkMan int,
@MobilePhone nvarchar(50),@Telephone nvarchar(50),@ShortNumber varchar(50),
@IDCard varchar(50),@OfficeAddress varchar(200),@OfficeFax varchar(50),
@HomePhone varchar(50),@Education varchar(50),@School varchar(50),
@Degree varchar(50),@HomeZipCode varchar(50),@HomeAddress varchar(200),
@HomeFax varchar(50),@NativePlace varchar(50),@Party varchar(50),
@Nation varchar(50),@Nationality varchar(50),@Major nvarchar(50),
@EducationalBackground nvarchar(50),@BirthdayType int,@Birthday date,
@BloodType nvarchar(10),@QQ nvarchar(50),@Email nvarchar(50),
@Interest nvarchar(200),@Description nvarchar(800),@DeleteMark int,
@Enabled int,@SortCode int,@CreateOn datetime,@CreateUserId nvarchar(50),
@CreatedBy nvarchar(50),@ModifiedOn datetime,@ModifyUserId nvarchar(50),
@ModifiedBy nvarchar(50)
AS
>     INSERT INTO [LinkMan] (
>     [CustomerId],[Name],[Sex],[Postion],[Department],[MainLinkMan],[MobilePhone],[Telephone],[ShortNumber],[IDCard],[OfficeAddress],[OfficeF
>     )VALUES(
>     @CustomerId,@Name,@Sex,@Postion,@Department,@MainLinkMan,@MobilePhone,@Telephone,@ShortNumber,@IDCard,@OfficeAddress,@OfficeFax,@HomePho
>     )
>     SET @Id = @@IDENTITY
GO

```

d) 修改数据（过程命名方式为：表名+“_Update”）。

```

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[LinkMan_Update]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[LinkMan_Update]
GO
-----
--用途：修改一条记录
--项目名称：
--说明：
--时间：2013-07-29 14:35:30
CREATE PROCEDURE LinkMan_Update
@Id int,@CustomerId int,@Name nvarchar(50),@Sex nvarchar(5),@Postion nvarchar(50),@Department nvarchar(50),
@MainLinkMan int,@MobilePhone nvarchar(50),@Telephone nvarchar(50),@ShortNumber varchar(50),@IDCard varchar(50),
@OfficeAddress varchar(200),@OfficeFax varchar(50),@School varchar(50),@Education varchar(50),@Degree varchar(50),
@HomeZipCode varchar(50),@HomeAddress varchar(200),@HomeFax varchar(50),@NativePlace varchar(50),
@Party varchar(50),@Nation varchar(50),@Nationality varchar(50),@Major nvarchar(50),@EducationalBackground nvarchar(50),
@BirthdayType int,@Birthday date,@BloodType nvarchar(10),@QQ nvarchar(50),@Email nvarchar(50),@Interest nvarchar(200),
@Description nvarchar(800),@DeleteMark int,@Enabled int,@SortCode int,@CreateOn datetime,@CreateUserId nvarchar(50),
@CreatedBy nvarchar(50),@ModifiedOn datetime,@ModifyUserId nvarchar(50),@ModifiedBy nvarchar(50)
AS
>     UPDATE [LinkMan] SET
>     [CustomerId] = @CustomerId,[Name] = @Name,[Sex] = @Sex,[Postion] = @Postion,[Department] = @Department,[MainLinkMan] = @MainLinkMan,[M
>     WHERE Id=@Id
GO

```

e) 删除数据（过程命名方式为：表名+“_Delete”）。

```

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[LinkMan_Delete]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[LinkMan_Delete]
GO
-----
--用途：删除一条记录
--项目名称：
--说明：
--时间：2013-07-29 14:35:30
-----
CREATE PROCEDURE LinkMan_Delete
@Id int
AS
>   DELETE [LinkMan]
>   WHERE Id=@Id
GO

```

f) 得到实体模型（过程命名方式为：表名+“GetEntity”）。

```

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[LinkManGetEntity]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[LinkManGetEntity]
GO
-----
--用途：得到实体对象的详细信息
--项目名称：
--说明：
--时间：2013-07-29 14:35:30
-----
CREATE PROCEDURE LinkManGetEntity
@Id int
AS
>   SELECT
>     Id,CustomerId,Name,Sex,Postion,Department,MainLinkMan,MobilePhone,Telephone,ShortNumber,IDCard,OfficeAddress,OfficeFax,HomePhone,Educa
>   FROM [LinkMan]
>   WHERE Id=@Id
GO

```

g) 得到数据列表（过程命名方式为：表名+“_GetList”）。

```

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[LinkMan_GetList]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[LinkMan_GetList]
GO
-----
--用途：查询记录信息
--项目名称：
--说明：
--时间：2013-07-29 14:35:30
-----
CREATE PROCEDURE LinkMan_GetList
AS
>   SELECT
>     Id,CustomerId,Name,Sex,Postion,Department,MainLinkMan,MobilePhone,Telephone,ShortNumber,IDCard,OfficeAddress,OfficeFax,HomePhone,Educa
>   FROM [LinkMan]
GO

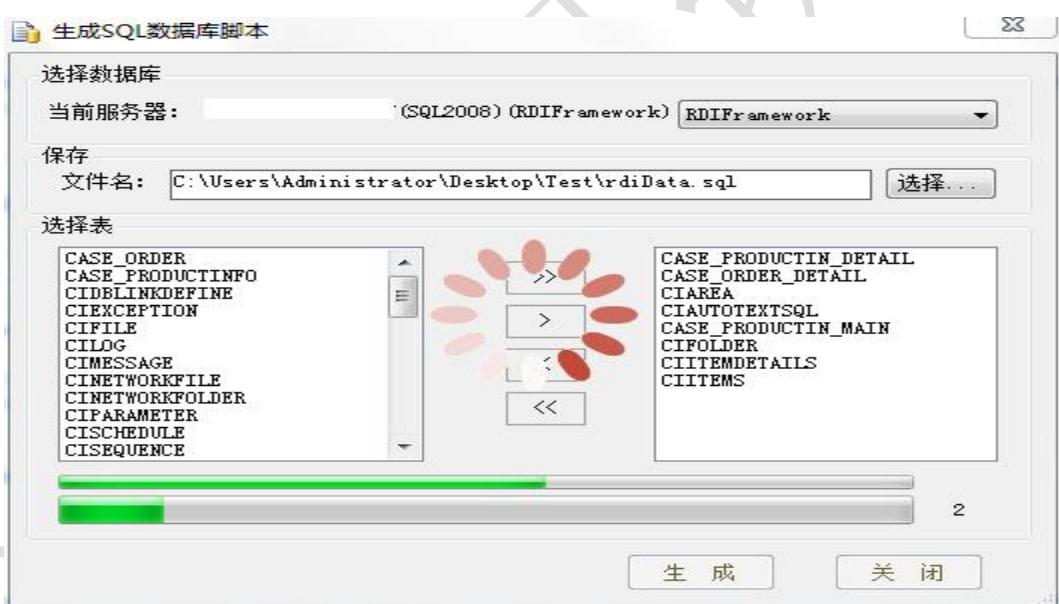
```

2.2.4. 生成数据脚本

“生成数据脚本”功能就是针对当前数据库，生成所有表的数据脚本，如下图所示，右键某个数据库节点，选择“生成数据脚本”。



打开“生成 SQL 数据库脚本”对话框，如下图所示。



在上图中，我们可以选择对应的数据库，默认选择的是当前所选数据库，选择需要生成数据脚本的表，设置好保存路径后就可以通过单击“生成”按钮，来生成所选数据库所表数据表的数据脚本（包括表的创建脚步与表的数据脚本）。部分脚本列表如下：

```

SET IDENTITY_INSERT [Customer] OFF
if exists (select * from sysobjects where id = OBJECT_ID('CustomerClass') and OBJECTPROPERTY(id, 'IsUserTable') = 1)
DROP TABLE [CustomerClass]

CREATE TABLE [CustomerClass] (
[Id] [int] IDENTITY (1, 1) NOT NULL,
[ParentId] [int] NULL,
[ClassName] [nvarchar] (50) NOT NULL,
[ClassCode] [nvarchar] (30) NULL,
[Description] [nvarchar] (800) NULL,
[SortCode] [int] NULL,
[DeleteMark] [int] NOT NULL DEFAULT ((0)),
[CreateOn] [datetime] NOT NULL DEFAULT (getdate()),
[CreateUserId] [nvarchar] (50) NULL,
[CreateBy] [nvarchar] (50) NULL,
[ModifiedId] [datetime] NULL,
[ModifiedUserId] [nvarchar] (50) NULL,
[ModifiedBy] [nvarchar] (50) NULL)

ALTER TABLE [CustomerClass] WITH NOCHECK ADD CONSTRAINT [PK_CustomerClass] PRIMARY KEY NONCLUSTERED ([Id])
SET IDENTITY_INSERT [CustomerClass] ON

INSERT [CustomerClass] ([Id], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateBy]) VALUES (1, '所有客户', 10000000, 0, '20
INSERT [CustomerClass] ([Id], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateBy]) VALUES (2, '最近使用客户', 10000001, 0
INSERT [CustomerClass] ([Id], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateBy]) VALUES (3, '经常使用客户', 10000002, 0
INSERT [CustomerClass] ([Id], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateBy]) VALUES (4, '固定客户', 10000003, 0, '20
INSERT [CustomerClass] ([Id], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateBy]) VALUES (5, '所有分类', 10000004, 0, '20
INSERT [CustomerClass] ([Id], [ParentId], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateUserId], [CreateBy]) VALUES (6, 5, '地区', 1000
INSERT [CustomerClass] ([Id], [ParentId], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateUserId], [CreateBy]) VALUES (7, 5, '关系', 1000
INSERT [CustomerClass] ([Id], [ParentId], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateUserId], [CreateBy], [ModifiedOn], [ModifiedBy]
INSERT [CustomerClass] ([Id], [ParentId], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateUserId], [CreateBy], [ModifiedOn], [ModifiedBy]
INSERT [CustomerClass] ([Id], [ParentId], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateUserId], [CreateBy], [ModifiedOn], [ModifiedBy]
INSERT [CustomerClass] ([Id], [ParentId], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateUserId], [CreateBy], [ModifiedOn], [ModifiedBy]
INSERT [CustomerClass] ([Id], [ParentId], [ClassName], [SortCode], [DeleteMark], [CreateOn], [CreateUserId], [CreateBy], [ModifiedOn], [ModifiedBy]

```

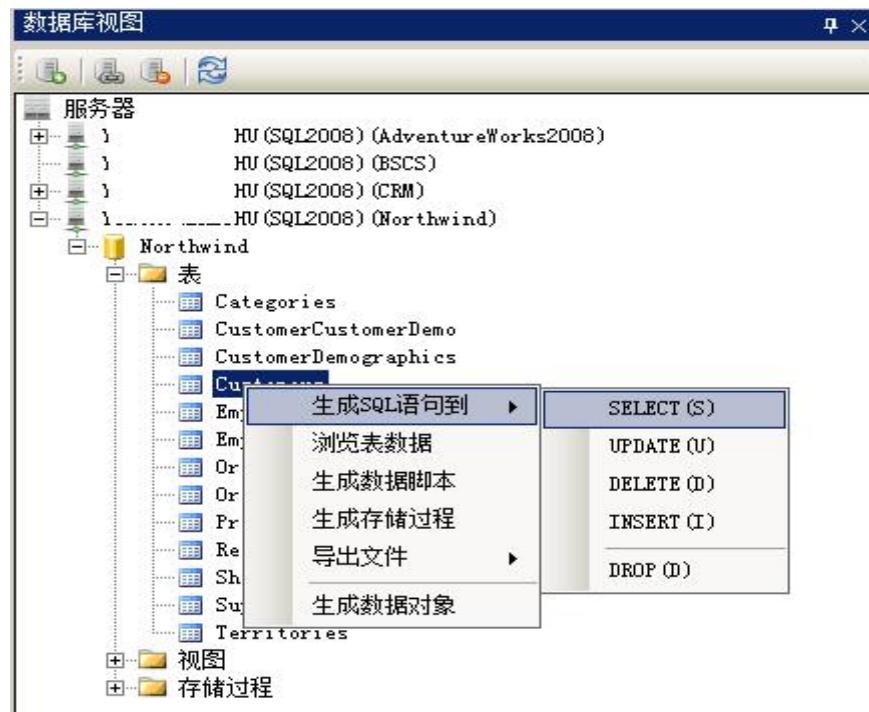
2.2.5. 导出文件（存储过程与数据脚本）

导出存储过程与导出数据脚本就是直接把数据库的存储过程、建表脚本和数据脚本保存到指定目录的文件中，而不是生成在界面上，生成的代码与上面两节的一致。



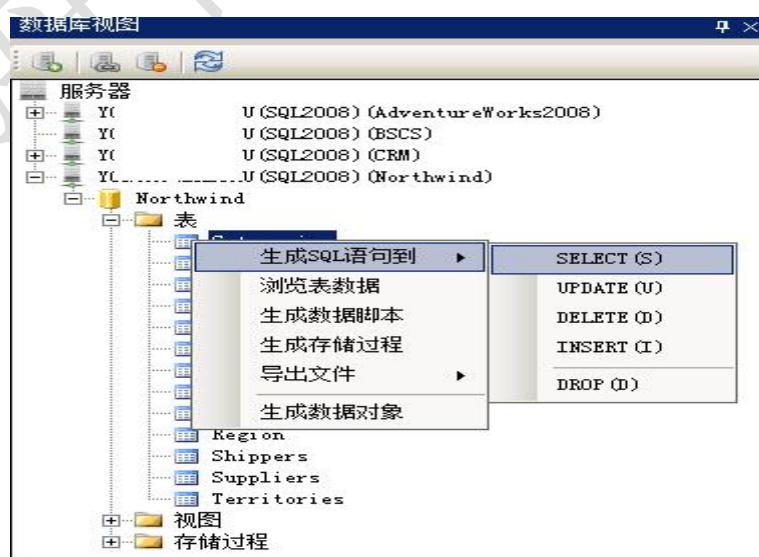
2.3. 表管理

连接到一个已注册的服务器，我们可以对当前服务器所选“数据库”的表进行相应的操作，主要包括：生成 SQL 语句（包括：SELECT 语句、UPDATE 语句、DELETE 语句、INSERT 语句、DROP 语句）、浏览表数据、生成数据脚本、生成存储过程、导出文件（存储过程与数据脚本）、生成数据对象等。



2.3.1. 生成 SQL 语句

“生成 SQL 语句”功能，可能当前所选数据表自动生成其 SELECT 语句、UPDATE 语句、DELETE 语句、INSERT 语句、DROP 语句等，对生成的语句可以直接在查询窗口执行，与在数据库管理器中的执行效果一样。



1) 生成 SELECT 语句。

The screenshot shows the RDIFramework.NET interface with a code editor containing a SQL SELECT statement:

```

1 select
2   [OrderID],
3   [CustomerID],
4   [EmployeeID],
5   [OrderDate],
6   [RequiredDate],|
7   [ShippedDate],
8   [ShipVia],
9   [Freight],
10  [ShipName],
11  [ShipAddress],
12  [ShipCity],
13  [ShipRegion],
14  [ShipPostalCode],
15  [ShipCountry]
16  from [Orders]
17

```

A context menu is open over the code, with the "运行当前查询" (Run Current Query) option highlighted.

Below the code editor is a grid displaying the results of the query:

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName
10251	VICTE	3	1996-07-08	1996-08-05	1996-07-15	1	41.3400	Victuari
10252	SUPRD	4	1996-07-09	1996-08-06	1996-07-11	2	51.3000	Supreme
10253	HANAR	3	1996-07-10	1996-07-24	1996-07-16	2	58.1700	Hanari
10254	RUNPE	5	1996-07-11	1996-08-02	1996-07-22	3	22.0000	Chopin

2) 生成 UPDATE 语句。

The screenshot shows the RDIFramework.NET interface with a code editor containing a SQL UPDATE statement:

```

1 update [Orders]
2   set
3     [OrderID] = <OrderID>,
4     [CustomerID] = <CustomerID>,
5     [EmployeeID] = <EmployeeID>,
6     [OrderDate] = <OrderDate>,
7     [RequiredDate] = <RequiredDate>,
8     [ShippedDate] = <ShippedDate>,
9     [ShipVia] = <ShipVia>,
10    [Freight] = <Freight>,
11    [ShipName] = <ShipName>,
12    [ShipAddress] = <ShipAddress>,
13    [ShipCity] = <ShipCity>,
14    [ShipRegion] = <ShipRegion>,
15    [ShipPostalCode] = <ShipPostalCode>,
16    [ShipCountry] = <ShipCountry>
17  where <搜索条件>
18

```

3) 生成 DELETE 语句。

The screenshot shows the RDIFramework.NET interface with a code editor containing a SQL DELETE statement:

```

1 delete from [Orders]
2   where <搜索条件>

```

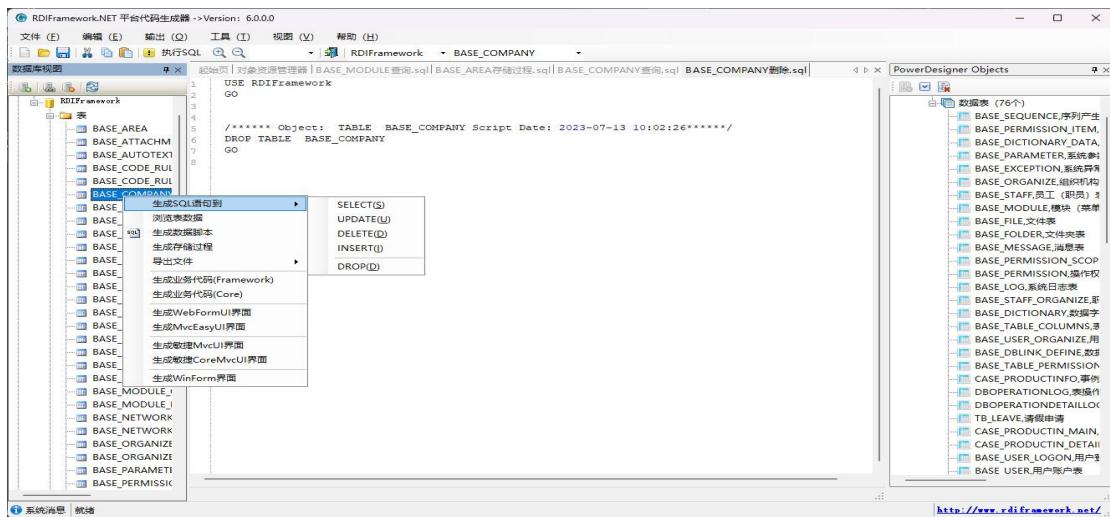
4) 生成 INSERT 语句。

The screenshot shows the RDIFramework.NET interface with the PowerDesigner Objects palette open. A context menu is open over a table named "BASE_COMPANY", with the "生成SQL语句" (Generate SQL Statement) option selected. This menu also includes options like SELECT(S), UPDATE(U), DELETE(D), and INSERT(I).

The PowerDesigner Objects palette lists various database objects:

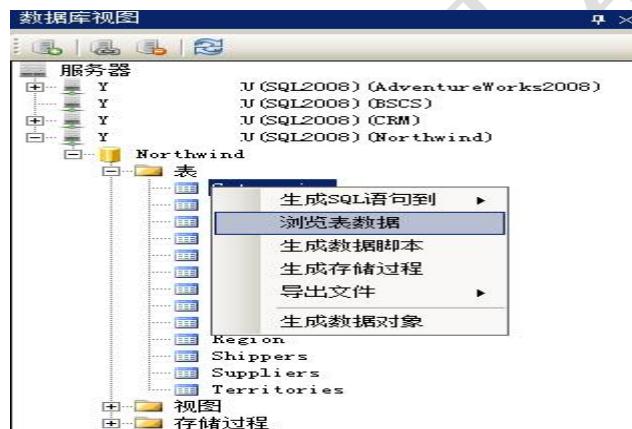
- 数据库视图 (76个)
 - BASE_SEQUENCE
 - BASE_PERMISSION_ITEM
 - BASE_DICTIONARY_DATA
 - BASE_PARAMETER
 - BASE_EXCEPTION
 - BASE_ORGANIZE
 - BASE_STAFF
 - BASE_MODULE
 - BASE_FILE
 - BASE_FOLDER
 - BASE_MESSAGE
 - BASE_PERMISSION_SCOPE
 - BASE_PERMISSION_OBJECT
 - BASE_LOG
 - BASE_STAFF_ORGANIZE
 - BASE_DICTIONARY
 - BASE_TABLE_COLUMNS
 - BASE_USER_ORGANIZE
 - BASE_DBLINK_DEFINE
 - BASE_TABLE_PERMISSION
 - CASE_PRODUCTINFO
 - DBOPERATIONLOG
 - DBOPERATIONDETAILLOG
 - TB_LEAVE
 - CASE_PRODUCTIN_MAIN
 - CASE_PRODUCTIN_DETAIL
 - BASE_USER_LOGON
 - BASE_USER

5) 生成 DROP 语句。



2.3.2. 浏览表数据

浏览表数据类似于在查询窗口执行“`SELECT*FROM 表名`”语句，选择一个表节点，右键单击选择“浏览表数据”，即可查看当前表的所有表数据。



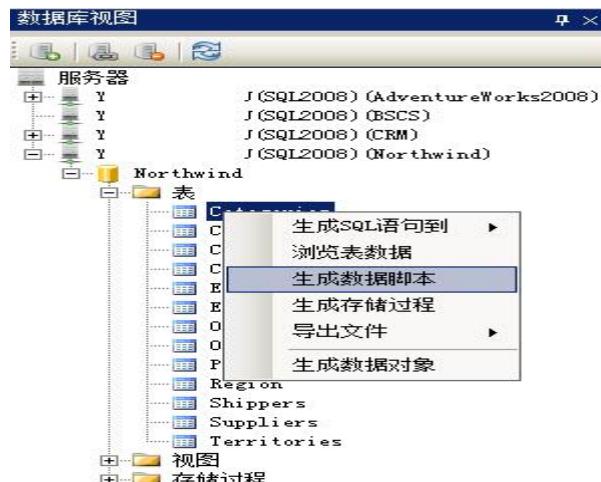
Tips: 如果表数据量过大，建议不要用此方法来浏览数据，可以使用查询窗口直接输入 `SELECT` 语句，使用 `TOP N` 来限制数据的条数。

起始页 Customers					
	CustomerID	CompanyName	ContactName	ContactTitle	Address
▶	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57
▶	ARATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda. de la Constitución 2222
▶	ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	Mataderos 2312
▶	AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.
▶	BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	Berguvsvägen 8
▶	BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Forststr. 57
▶	BLONP	Blondesddsl père et fils	Frédérique Citeaux	Marketing Manager	24, place Kléber
▶	BOLID	Bólido Comidas preparadas	Martín Sommer	Owner	C/ Araquil, 67
▶	BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bouchers
▶	BOTTM	Bottom-Dollar Markets	Elizabet Lincoln	Accounting Manager	23 Tsawassen Blvd.
▶	BSBEV	B's Beverages	Victoria Ashworth	Sales Representative	Fauntleroy Circus
▶	CACTU	Cactus Comidas para llevar	Patricia Simpson	Sales Agent	Cerrito 333
▶	CENTC	Centro comercial Moctezuma	Francisco Chang	Marketing Manager	Sierras de Granada 9993
▶	CHOPS	Chop-suey Chinese	Yang Wang	Owner	Hauptstr. 29
▶	COMMI	Comércio Mineiro	Pedro Afonso	Sales Associate	Av. dos Lusíadas, 23
▶	CONSH	Consolidated Holdings	Elizabeth Brown	Sales Representative	Berkeley Gardens 12 Brewery
▶	DRACD	Drachenblut Delikatessen	Sven Ottlieb	Order Administrator	Walserweg 21
▶	DUMON	Du monde entier	Janine Labrune	Owner	87, rue des Cinquante Otages

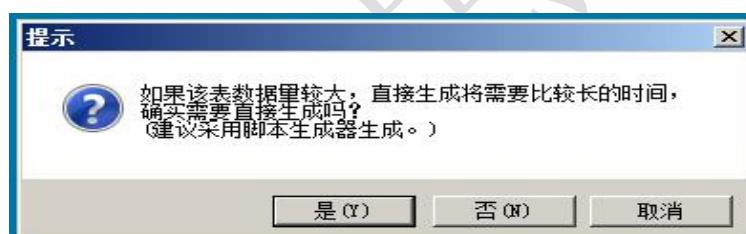
在上图中，我们可以通过右下角查看当前数据库、当前表、执行此操作所花费的时间，数据的行数等信息。

2.3.3. 生成数据脚本

生成数据脚本功能就是生成当前所选表的创建脚本与数据脚本。



在选择“生成数据脚本”功能后，会弹出对话框，让用户选择，如下图所示。



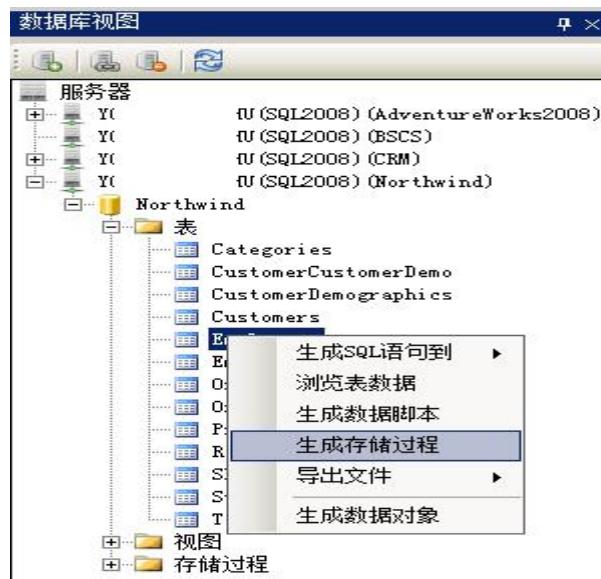
选择“否”即可打开“生成 SQL 数据库脚本”窗口，给 2.2.4 节的一至。选择“是”，则可直接生成当前表的数据脚本，如下图所示。

```
起始页 Categories 脚本.sql
1 if exists (select * from sysobjects where id = OBJECT_ID('Categories') and OBJECTPROPERTY(id, 'IsUserTable') = 1)
2 DROP TABLE Categories
3
4 CREATE TABLE Categories (
5 [CategoryID] [int] IDENTITY (1, 1) NOT NULL,
6 [CategoryName] [nvarchar] (15) NOT NULL,
7 [Description] [ntext] NULL,
8 [Picture] [image] NULL
9
10 ALTER TABLE Categories WITH NOCHECK ADD CONSTRAINT [PK_Categories] PRIMARY KEY NONCLUSTERED ([CategoryID])
11 SET IDENTITY_INSERT Categories ON
12
13 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 1,'Beverages','Soft drinks, coffees, teas, beer
14 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 2,'Condiments','Sweet and savory sauces, relish
15 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 3,'Confections','Desserts, candies, and sweet k
16 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 4,'Dairy Products','Cheeses',System.Byte[])
17 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 5,'Grains/Cereals','Breads, crackers, pasta, an
18 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 6,'Meat/Poultry','Prepared meats',System.Byte[])
19 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 7,'Produce','Dried fruit and bean curd',System.
20 INSERT Categories ([CategoryID], [CategoryName], [Description], [Picture]) VALUES ( 8,'Seafood','Seaweed and fish',System.Byte[])
21 SET IDENTITY_INSERT Categories OFF
```

Tips:如果所选表的数据过多，通过上面的方式生成数据脚本可能会花很长时间，此时建议在单击“生成数据脚本”后弹出的提示框时选择“否”打开“生成 SQL 数据库脚本”窗口来进行表数据脚本的生成。

2.3.4. 生成存储过程

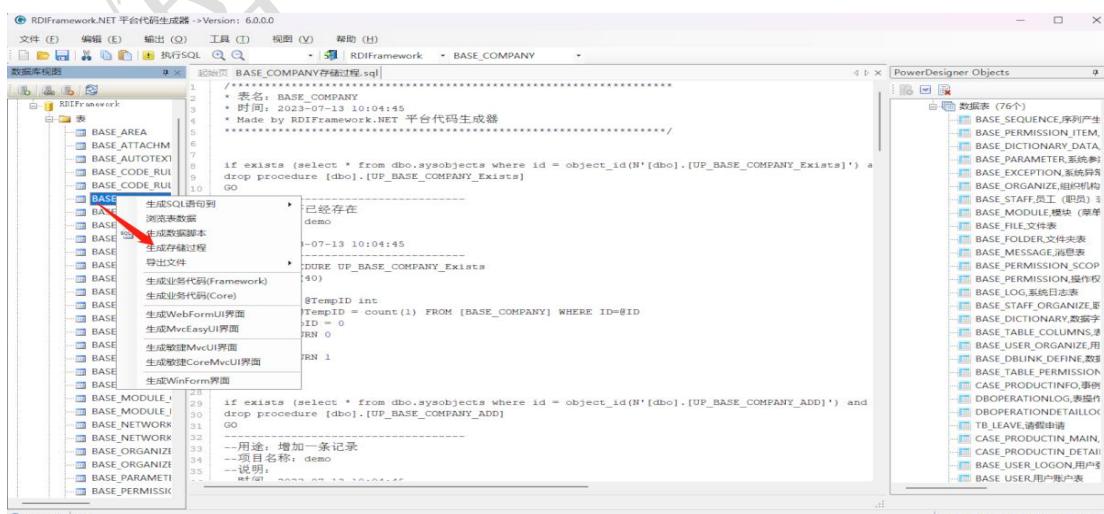
“生成存储过程”就是生成当前所选表的业务逻辑存储过程。



生成的存储过程包括以下几种类型：

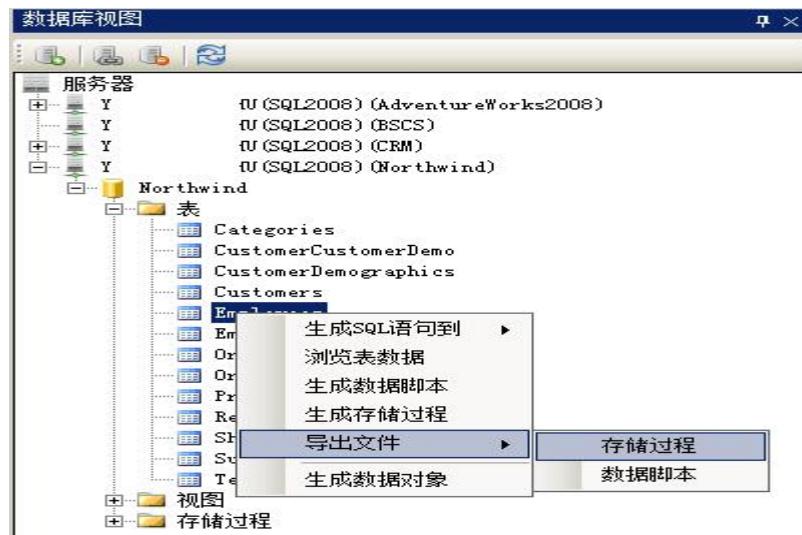
- 得到主键字段的最大值（过程命名方式为：表名+“_GetMaxId”）。
- 数据存在性判断（过程命名方式为：表名+“_Exists”）。
- 新增数据（过程命名方式为：表名+“_Add”）。
- 修改数据（过程命名方式为：表名+“_Update”）。
- 删除数据（过程命名方式为：表名+“_Delete”）。
- 得到实体模型（过程命名方式为：表名+“GetEntity”）。
- 得到数据列表（过程命名方式为：表名+“_GetList”）。

部分截图如下：



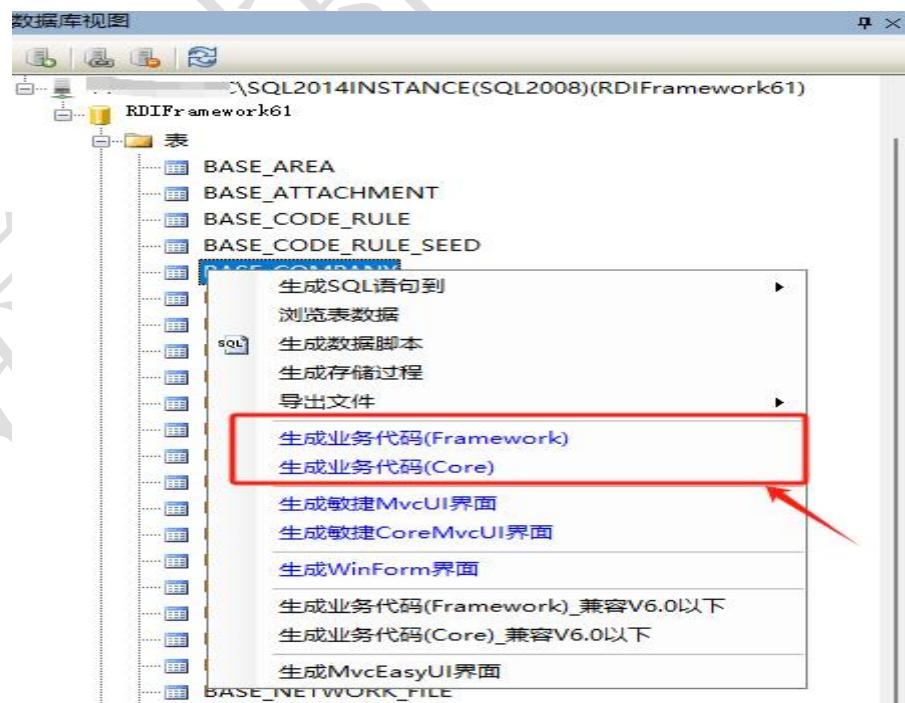
2.3.5. 导出文件（存储过程与数据脚本）

导出存储过程与导出数据脚本就是直接把所选表的存储过程、建表脚本和表数据脚本保存到指定目录的文件中，而不是生成在界面上，生成的代码与上面两节的一致。

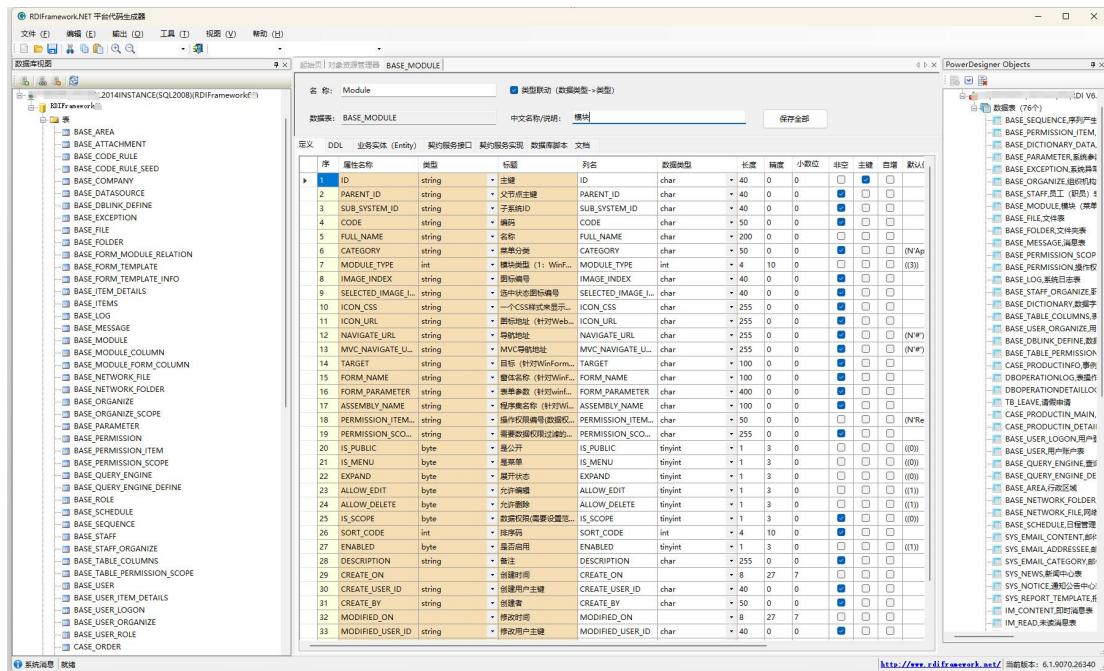


2.3.6. 生成业务代码（重量级）

“生成业务代码”分为生成支持.NET Framework 与.NET Core 版本的业务逻辑代码，通过此模块，可以生成所需的核心业务逻辑，注意：我们的代码生成器，是基于 RDI Framework.NET 框架的代码生成器。选择一个数据表，单击右键，选择“业务逻辑代码（Framework）”或“生成业务逻辑代码（Core）”功能，如下图所示。



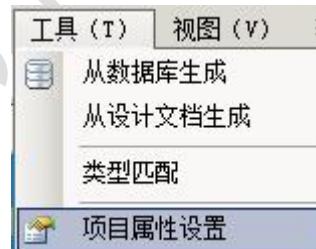
打开所选表的数据库对象界面，在该界面中，显示了当前所选表的概要信息以及包含八个选项卡信息，下面的讲解主要围绕这八个选项卡进行，如下图所示。



在进入下面的内容前，需要简单说明一下针对代码生成的公开设置部分，那就是项目属性设置。

2.3.6.1. 项目属性设置

“项目属性”设置，是对代码生成器的公共设置，如设置项目的名称、公司名称、作者名称、代码输出目录等，后面的代码生成相关的信息都要以此设置作为公共引用，要打开“项目属性”窗口，需要通过主菜单“工具”菜单，选择“项目属性设置”，即可打开“项目属性设置”窗口。



打开的“项目属性设置”窗口如下：



在“项目属性设置”的“项目”Tab 窗口，设置代码生成的所需的公共信息如下：

项目名称(程序集名称): 代码生成所需的命名空间的名称。

公司名称: 代码生成时版权部分需引用。

开发人员: 代码生成的作者信息。

输出区域 (Area 名称) : 针对 Mvc 的区域名称设置。

代码输出目录: 此主要用于代码批量生成时，保存代码的目录。

字段规范化: 选中则会对生成的代码做规范化处理，如：FULL_NAME 为变为：FullName，不选中就会原样处理。

去掉的前缀: 生成的表需要去掉的前缀，如：设置为 BASE_，则针对 BASE_AREA，会去掉 BASE_，多个以英文逗号隔开。

在项目属性的“其他”Tab 页，可以针对一些保留字段、扩展字段等进行设置。



通过以上设置，接下来我们就可以进行下面的工作了。

2.3.6.2. 数据表的定义展示

数据表的定义主要显示的是当前表的列信息与生成的实体代码的对应关系，后期还会增加在此就可以新增列、修改列等，表的定义（DDL）展示如下图所示。

序	属性名称	类型	标题	列名	数据类型	长度	编码	小数位	非空	主键	白墙	默认
1	ID	string	· 主键	ID	char	40	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	PARENT_ID	string	· 父节点主键	PARENT_ID	char	40	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	SUB_SYSTEM_ID	string	· 子系统ID	SUB_SYSTEM_ID	char	40	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
4	CODE	string	· 编码	CODE	char	50	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5	FULL_NAME	string	· 名称	FULL_NAME	char	200	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
6	CATEGORY	string	· 单分类	CATEGORY	char	50	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(N'Ap
7	MODULE_TYPE	int	· 模块类型 (1: WinForm, 2: WebForm)	MODULE_TYPE	int	4	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(3))
8	IMAGE_INDEX	string	· 图标索引	IMAGE_INDEX	char	40	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
9	SELECTED_IMAGE_INDEX	string	· 选中状态图标索引	SELECTED_IMAGE_INDEX	char	40	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
10	ICON_CSS	nvarchar	· 一个CSS式样示...	ICON_CSS	char	255	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
11	ICON_URL	nvarchar	· 地址	ICON_URL	char	255	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
12	NAVIGATE_URL	nvarchar	· 导航地址	NAVIGATE_URL	char	255	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(N'W')
13	MVC_NAVIGATE_URL	nvarchar	· MVC导航地址	MVC_NAVIGATE_URL	char	255	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(N'W')
14	TARGET	string	· 目标 (针对WinForm, 2: WebForm)	TARGET	char	100	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
15	FORM_NAME	string	· 表单名称 (针对WinForm, 2: WebForm)	FORM_NAME	char	100	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
16	FORM_PARAMETER	string	· 表单参数 (针对WinForm, 2: WebForm)	FORM_PARAMETER	char	400	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
17	ASSEMBLY_NAME	string	· 程序集名 (不写Web, ASSEMBLY_NAME)	ASSEMBLY_NAME	char	100	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
18	PERMISSION_ITEM_CODE	string	· 操作权限代码...	PERMISSION_ITEM_CODE	char	50	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(N'Re
19	PERMISSION_SCOPE	string	· 需要授权权限过...	PERMISSION_SCOPE	char	255	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
20	IS_PUBLIC	byte	· 是公开	IS_PUBLIC	tinyint	1	3	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(0))
21	IS_MENU	byte	· 是菜单	IS_MENU	tinyint	1	3	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(0))
22	EXPAND	byte	· 展开状态	EXPAND	tinyint	1	3	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(0))
23	ALLOW_EDIT	byte	· 允许编辑	ALLOW_EDIT	tinyint	1	3	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(1))
24	ALLOW_DELETE	byte	· 允许删除	ALLOW_DELETE	tinyint	1	3	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(1))
25	IS_SCOPE	byte	· 数据权限(需要设置...)	IS_SCOPE	tinyint	1	3	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(0))
26	SORT_CODE	int	· 排序码	SORT_CODE	int	4	10	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
27	ENABLED	byte	· 是否启用	ENABLED	tinyint	1	3	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	(1))
28	DESCRIPTION	string	· 备注	DESCRIPTION	char	255	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
29	CREATE_ON	datetime2	· 创建时间	CREATE_ON	char	8	27	7	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
30	CREATE_USER_ID	nvarchar	· 创建用户主键	CREATE_USER_ID	char	40	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
31	CREATE_BY	nvarchar	· 创建人	CREATE_BY	char	50	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
32	MODIFIED_ON	datetime2	· 修改时间	MODIFIED_ON	char	8	27	7	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
33	MODIFIED_USER_ID	nvarchar	· 修改用户主键	MODIFIED_USER_ID	char	40	0	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

在上图中，以淡黄色底色显示的是列对应到实体的类型对应关系，后半部分显示的是表列的详细信息。

2.3.6.3. 表的 DDL

表的 DDL 可以查看当前表的 CREATE 代码，如下图所示。

```

if exists (SELECT * FROM sysobjects where id = OBJECT_ID('[BASE_MODULE]') and OBJECTPROPERTY(id, 'IsUserTable') = 1)
DROP TABLE [BASE_MODULE]

CREATE TABLE [BASE_MODULE] (
    [ID] [varchar] (40) NOT NULL,
    [PARENT_ID] [varchar] (40) NULL,
    [SUB_SYSTEM_ID] [varchar] (40) NULL,
    [CODE] [nvarchar] (50) NULL,
    [FULL_NAME] [nvarchar] (200) NOT NULL,
    [CATEGORY] [nvarchar] (50) NULL DEFAULT ('N'Application'),
    [MODULE_TYPE] [int] NOT NULL DEFAULT ((3)),
    [IMAGE_INDEX] [varchar] (40) NULL,
    [SELECTED_IMAGE_INDEX] [varchar] (40) NULL,
    [ICON_CSS] [nvarchar] (255) NULL,
    [ICON_URL] [nvarchar] (255) NULL,
    [NAVIGATE_URL] [nvarchar] (255) NULL DEFAULT ('N'#),
    [MVC_NAVIGATE_URL] [nvarchar] (255) NULL DEFAULT ('N'#'),
    [TARGET] [nvarchar] (100) NULL,
    [FORM_NAME] [nvarchar] (100) NULL,
    [FORM_PARAMETER] [nvarchar] (400) NULL,
    [ASSEMBLY_NAME] [nvarchar] (100) NULL,
    [PERMISSION_ITEM_CODE] [nvarchar] (50) NOT NULL DEFAULT ('N'Resource.AccessPermission'),
    [PERMISSION_SCOPE_TABLES] [nvarchar] (255) NULL,
    [IS_PUBLIC] [tinyint] NOT NULL DEFAULT ((0)),
    [IS_MENU] [tinyint] NOT NULL DEFAULT ((0)),
    [EXPAND] [tinyint] NOT NULL DEFAULT ((0)),
    [ALLOW_EDIT] [tinyint] NOT NULL DEFAULT ((1)),
    [ALLOW_DELETE] [tinyint] NOT NULL DEFAULT ((1)),
    [IS_SCOPE] [tinyint] NULL DEFAULT ((0)),
    [SORT_CODE] [int] NULL,
    [ENABLED] [tinyint] NOT NULL DEFAULT ((1)),
    [DESCRIPTION] [nvarchar] (255) NULL,
    [CREATE_ON] [datetime2] NOT NULL,
    [CREATE_USER_ID] [varchar] (40) NULL,
    [CREATE_BY] [nvarchar] (50) NULL,
    [MODIFIED_ON] [datetime2] NOT NULL,
    [MODIFIED_USER_ID] [varchar] (40) NULL,
    [MODIFIED_BY] [nvarchar] (50) NULL,
    [MODIFIED_BY] [datetime2] NOT NULL
)

```

2.3.6.4. 业务实体 (Entity)

根据数据表，自动生成业务实体（也可称 Model），如下图所示。



```

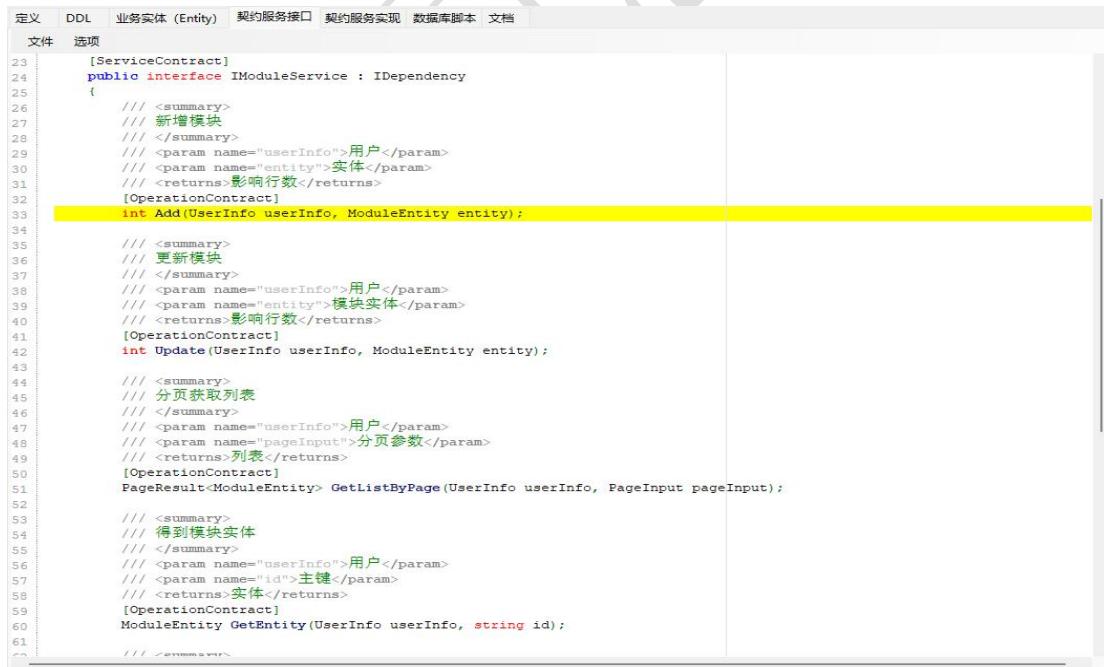
1 [RDIEntity("BASE_MODULE")]
2 public partial class ModuleEntity : BaseCUDEntity
3 {
4     /// <summary>
5     /// 父节点主键
6     /// </summary>
7     [DataMember]
8     [RDIColumn.ColumnName = "PARENT_ID", ColumnDescription = "父节点主键"]
9     public string ParentId { get; set; }
10
11    /// <summary>
12    /// 子系统ID
13    /// </summary>
14    [DataMember]
15    [RDIColumn.ColumnName = "SUB_SYSTEM_ID", ColumnDescription = "子系统ID"]
16    public string SubSystemId { get; set; }
17
18    /// <summary>
19    /// 编码
20    /// </summary>
21    [DataMember]
22    [RDIColumn.ColumnName = "CODE", ColumnDescription = "编码"]
23    public string Code { get; set; }
24
25    /// <summary>
26    /// 名称
27    /// </summary>
28    [DataMember]
29    [RDIColumn.ColumnName = "FULL_NAME", ColumnDescription = "名称"]
30    public string FullName { get; set; }
31
32    /// <summary>
33    /// 菜单分类
34    /// </summary>
35    [DataMember]
36    [RDIColumn.ColumnName = "CATEGORY", ColumnDescription = "菜单分类"]
37    public string Category { get; set; }
38
39
40
41
42
43
44
45
46
47
48

```

2.3.6.5. 契约服务接口

“契约服务接口”代码，是业务的接口实现，框架的代码生成器自动生成了一些常用的业务逻辑接口，如：

新增数据接口、修改数据接口、删除数据接口、得到数据表、得到分页数据表、得到实体，通过条件得到数据、批量保存等等，如下图所示。



```

1 [ServiceContract]
2 public interface IModuleService : IDependency
3 {
4     /// <summary>
5     /// 新增模块
6     /// </summary>
7     /// <param name="userInfo">用户</param>
8     /// <param name="entity">实体</param>
9     /// <returns>影响行数</returns>
10    [OperationContract]
11    int Add(UserInfo userInfo, ModuleEntity entity);
12
13    /// <summary>
14    /// 更新模块
15    /// </summary>
16    /// <param name="userInfo">用户</param>
17    /// <param name="entity">模块实体</param>
18    /// <returns>影响行数</returns>
19    [OperationContract]
20    int Update(UserInfo userInfo, ModuleEntity entity);
21
22    /// <summary>
23    /// 分页获取列表
24    /// </summary>
25    /// <param name="userInfo">用户</param>
26    /// <param name="pageInput">分页参数</param>
27    /// <returns>列表</returns>
28    [OperationContract]
29    PageResult<ModuleEntity> GetListByPage(UserInfo userInfo, PageInput pageInput);
30
31    /// <summary>
32    /// 得到模块实体
33    /// </summary>
34    /// <param name="userInfo">用户</param>
35    /// <param name="id">主键</param>
36    /// <returns>实体</returns>
37    [OperationContract]
38    ModuleEntity GetEntity(UserInfo userInfo, string id);
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
69
70
71
72
73
74
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
89
89
90
91
92
93
94
95
96
97
98
99
99
100
101
102
103
104
105
106
107
108
109
109
110
111
112
113
114
115
116
117
118
119
119
120
121
122
123
124
125
126
127
128
129
129
130
131
132
133
134
135
136
137
138
139
139
140
141
142
143
144
145
146
147
148
149
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
169
170
171
172
173
174
175
176
177
178
179
179
180
181
182
183
184
185
186
187
187
188
189
189
190
191
192
193
194
195
196
197
197
198
199
199
200
201
202
203
204
205
205
206
207
208
209
209
210
211
212
213
214
215
215
216
217
217
218
219
219
220
221
222
223
223
224
225
225
226
227
227
228
229
229
230
231
231
232
232
233
233
234
234
235
235
236
236
237
237
238
238
239
239
240
240
241
241
242
242
243
243
244
244
245
245
246
246
247
247
248
248
249
249
250
250
251
251
252
252
253
253
254
254
255
255
256
256
257
257
258
258
259
259
260
260
261
261
262
262
263
263
264
264
265
265
266
266
267
267
268
268
269
269
270
270
271
271
272
272
273
273
274
274
275
275
276
276
277
277
278
278
279
279
280
280
281
281
282
282
283
283
284
284
285
285
286
286
287
287
288
288
289
289
290
290
291
291
292
292
293
293
294
294
295
295
296
296
297
297
298
298
299
299
300
300
301
301
302
302
303
303
304
304
305
305
306
306
307
307
308
308
309
309
310
310
311
311
312
312
313
313
314
314
315
315
316
316
317
317
318
318
319
319
320
320
321
321
322
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
```

```
49  public class ModuleService : System.MarshalByRefObject, IModuleService
50  {
51      private string serviceName = "Module";
52
53      #region public int Add(UserInfo userInfo, ModuleEntity entity) 新增模块
54      /// <summary>
55      /// 新增模块
56      /// </summary>
57      /// <param name="userInfo">用户</param>
58      /// <param name="entity">模块实体</param>
59      /// <returns>影响行数</returns>
60      public int Add(UserInfo userInfo, ModuleEntity entity)
61      {
62          int returnValue = 0;
63          var parameter = ParameterUtil.CreateWithMessage(userInfo, MethodBase.GetCurrentMethod(), this.serviceName, "新增实体");
64
65          ServiceSugarUtil.ProcessBusinessDb(userInfo, parameter, dbProvider =>
66          {
67              returnValue = dbProvider.Insertable(entity).CallEntityMethod(it => it.Create(userInfo)).IgnoreColumnsNull(isIgnoreNull: true).Execute();
68          });
69          return returnValue;
70      }
71      #endregion
72
73      #region public int Update(UserInfo userInfo, ModuleEntity entity) 更新模块
74      /// <summary>
75      /// 更新模块
76      /// </summary>
77      /// <param name="userInfo">用户</param>
78      /// <param name="entity">模块实体</param>
79      /// <returns>影响行数</returns>
80      public int Update(UserInfo userInfo, ModuleEntity entity)
81      {
82          int returnValue = 0;
83          var parameter = ParameterUtil.CreateWithMessage(userInfo, MethodBase.GetCurrentMethod(), this.serviceName, "更新实体");
84
85          ServiceSugarUtil.ProcessBusinessDb(userInfo, parameter, dbProvider =>
86          {
87              returnValue = dbProvider.Updateable(entity).CallEntityMethod(it => it.LastModify(userInfo)).ExecuteCommand();
88          });
89      }
90  }
```

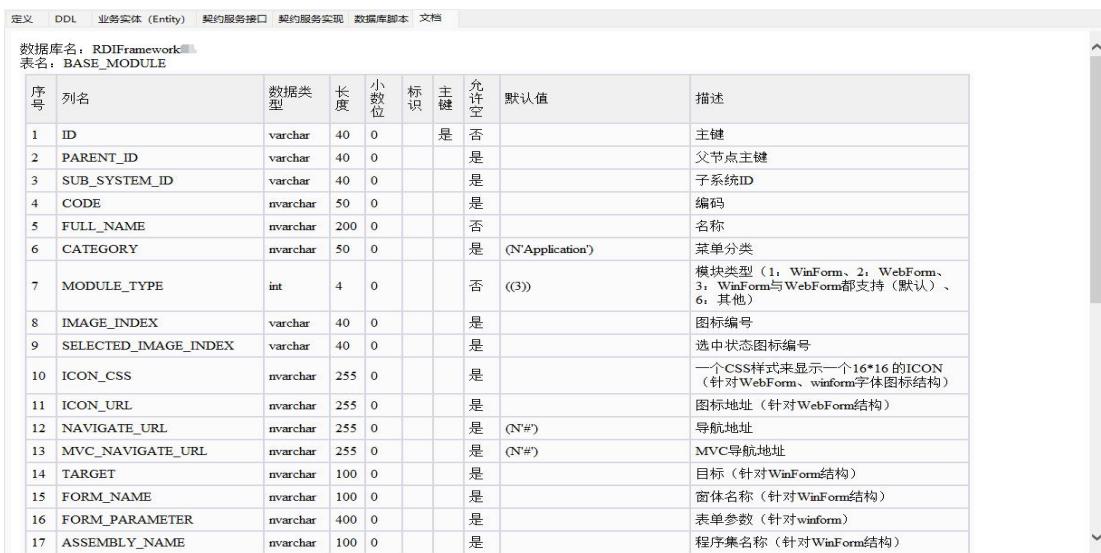
2.3.6.7. 生成数据库脚本

数据库脚本主要用于权限控制表中使用，我们在以通过代码生成器生成待控制的权限控制表脚本数据，如下图所示。

```
1 定义 DDL 业务实体 (Entity) 约契约服务接口 约契约服务实现 数据库脚本 文档
2 文件 选项
3
4 USE [RDIFramework]
5
6 -- 先删除重复的数据
7 DELETE FROM BASE_TABLE_COLUMNS WHERE (TABLE_CODE = 'BASE_MODULE');
8
9 -- 插入字段说明数据
10 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'I', 'I')
11 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'F', 'F')
12 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'S', 'S')
13 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'C', 'C')
14 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'U', 'U')
15 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'S', 'S')
16 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'I', 'I')
17 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'F', 'F')
18 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'R', 'R')
19 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'N', 'N')
20 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'T', 'T')
21 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'U', 'U')
22 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'F', 'F')
23 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'A', 'A')
24 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'E', 'E')
25 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'P', 'P')
26 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'I', 'I')
27 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'F', 'F')
28 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'E', 'E')
29 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'R', 'R')
30 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'N', 'N')
31 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'T', 'T')
32 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'U', 'U')
33 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'F', 'F')
34 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'I', 'I')
35 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'C', 'C')
36 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'U', 'U')
37 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'F', 'F')
38 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'I', 'I')
39 INSERT INTO BASE_TABLE_COLUMNS (TABLE_CODE, COLUMN_CODE, COLUMN_NAME, USE_CONSTRAINT, DATA_TYPE, ENABLED , SORT_CODE) VALUES ('BASE_MODULE', 'F', 'F')
```

2.3.6.8. 文档

文档就是生成当前数据表的表结构文档，如下图所示。



The screenshot shows the RDIFramework.NET Entity definition interface. At the top, there's a navigation bar with tabs: 定义 (Definition), DDL, 业务实体 (Entity), 契约服务接口 (Contract Service Interface), 契约服务实现 (Contract Service Implementation), 数据库脚本 (Database Script), and 文档 (Documentation). Below the navigation bar, it says '数据库名: RDIFramework' and '表名: BASE_MODULE'. The main area is a table with 17 rows, each representing a column in the database table. The columns are: 序号 (Index), 列名 (Column Name), 数据类型 (Data Type), 长度 (Length), 小数位 (Decimal Places), 标识 (Identity), 主键 (Primary Key), 允许空 (Allow Null), 默认值 (Default Value), and 描述 (Description). The table includes detailed descriptions for some columns, such as '模块类型 (1: WinForm、2: WebForm、3: WinForm与WebForm都支持 (默认)、6: 其他)' for MODULE_TYPE.

序号	列名	数据类型	长度	小数位	标识	主键	允许空	默认值	描述
1	ID	varchar	40	0		是	否		主键
2	PARENT_ID	varchar	40	0		是			父节点主键
3	SUB_SYSTEM_ID	varchar	40	0		是			子系统ID
4	CODE	nvarchar	50	0		是			编码
5	FULL_NAME	nvarchar	200	0		否			名称
6	CATEGORY	nvarchar	50	0		是	(N'Application')		菜单分类
7	MODULE_TYPE	int	4	0		否	(3)		模块类型 (1: WinForm、2: WebForm、3: WinForm与WebForm都支持 (默认)、6: 其他)
8	IMAGE_INDEX	varchar	40	0		是			图标编号
9	SELECTED_IMAGE_INDEX	varchar	40	0		是			选中状态图标编号
10	ICON_CSS	nvarchar	255	0		是			一个CSS样式来显示一个16*16 的ICON (针对WebForm、winform字体图标结构)
11	ICON_URL	nvarchar	255	0		是			图标地址 (针对WebForm结构)
12	NAVIGATE_URL	nvarchar	255	0		是	(N'#')		导航地址
13	MVC_NAVIGATE_URL	nvarchar	255	0		是	(N'#')		MVC导航地址
14	TARGET	nvarchar	100	0		是			目标 (针对WinForm结构)
15	FORM_NAME	nvarchar	100	0		是			窗体名称 (针对WinForm结构)
16	FORM_PARAMETER	nvarchar	400	0		是			表单参数 (针对winform)
17	ASSEMBLY_NAME	nvarchar	100	0		是			程序集名称 (针对WinForm结构)

2.3.6.9. 辅助功能

在生成的各个代码界面窗口中，有一个工具栏，可对生成的代码做相应的操作（如：保存当前生成的代码等），如下图所示。



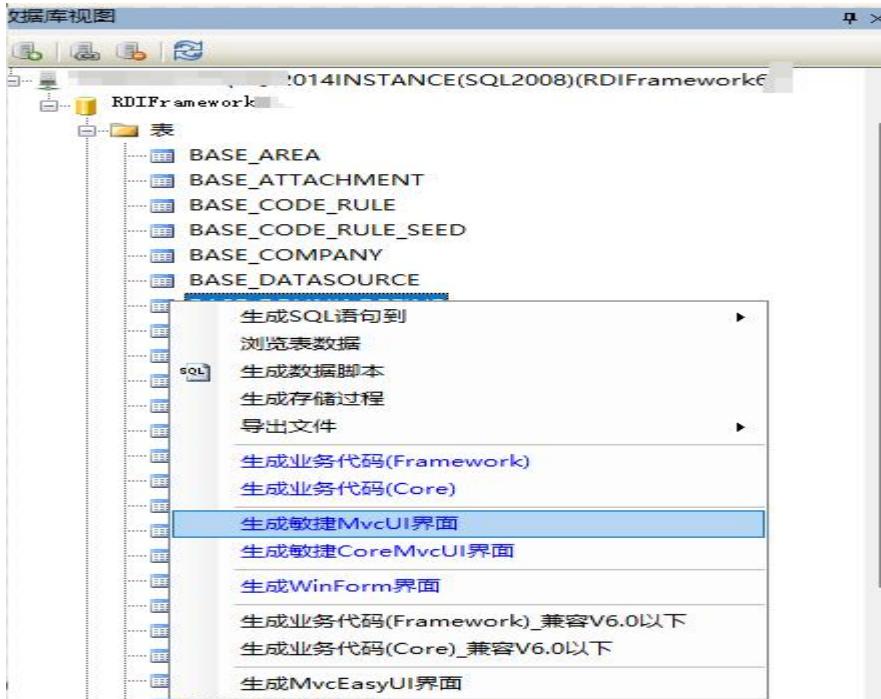
通过“文件”菜单，用户可以保存当前生成的代码。现在说明一下“选项”菜单的功能，“选项”菜单各个子菜单的功能主要是对当前代码生成窗口的代码进行格式控制，所有的代码生成窗口，都共享这些选项菜单，“选项”菜单的子菜单功能如下：

- 1) 拆分窗口。
- 2) 显示空格和制表符。
- 3) 显示换行标记。
- 4) 显示无效标记。
- 5) 显示行号。
- 6) 高亮当前行。
- 7) 高亮匹配括号当前光标在其后时。
- 8) 启用虚线。
- 9) 制表符转换为空格。

10) 字体（代码的字体设置）。

2.3.7. 生成敏捷 MvcUI 界面

“生成敏捷 MvcUI 界面”功能模块是 3.5 版本开始新增的功能，可以生成敏捷 MVC 的 UI 界面代码，这样就可以大大缩小界面的开发时间，对生成的 UI 代码做适量的修改即可。选择一个数据表，单击右键，选择“生成敏捷 MvcUI 界面”功能，如下图所示。



2.3.7.1. Index.cshtml 页面代码

MvcUI 的 Index.cshtml 页面代码如下图所示。

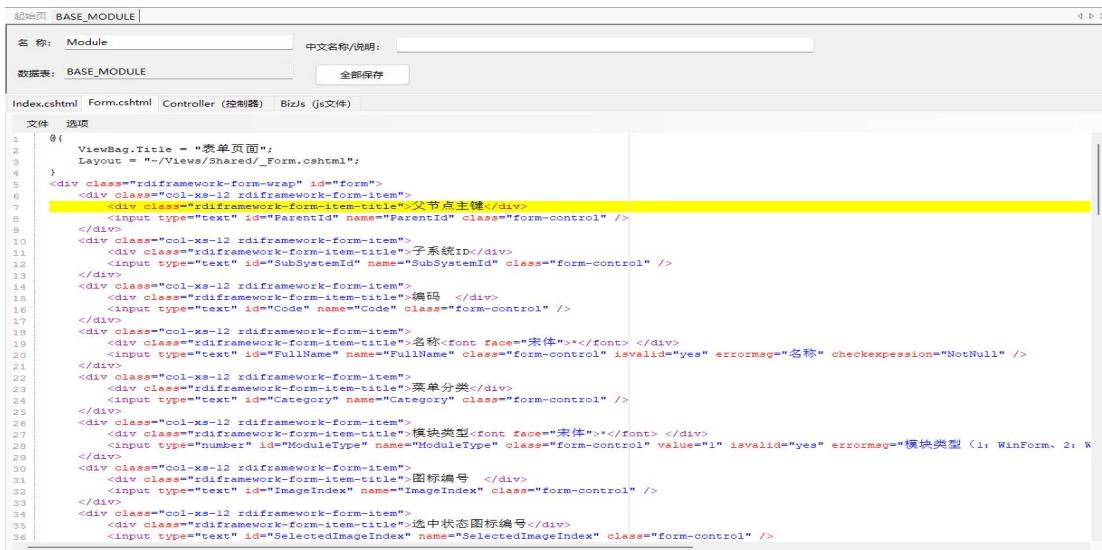
```

@{
    ViewBag.Title = "Title";
    Layout = "~/Views/Shared/_Admin.cshtml";
}
<div class="rdi-layout">
    <div class="rdi-layout-center">
        <div class="rdi-layout-wrap-notitle">
            <div class="rdi-layout-tool-right">
                <div class="rdi-layout-tool-item">
                    <input id="txt_Keyword" type="text" class="form-control" placeholder="请输入查询关键字" />
                </div>
                <div class="rdi-layout-tool-item">
                    <a id="btn_Search" class="btn btn-primary btn-sm"><i class="fa fa-search"></i>&ampnbsp查询</a>
                </div>
            </div>
            <div class="rdi-layout-tool-right">
                <div class="btn-group btn-group-sm">
                    <a id="rdi_refesh" class="btn btn-default"><i class="fa fa-refresh"></i></a>
                </div>
                <div class="btn-group btn-group-sm">
                    @if (((bool) ViewData["QX.Add"]))
                    {
                        <a id="rdi_Add" class="btn btn-default"><i class="fa fa-plus"></i>&ampnbsp@Html.DisplayByResource("Add", "新增")</a>
                    }
                    @if (((bool) ViewData["QX.Edit"]))
                    {
                        <a id="rdi_Edit" class="btn btn-default"><i class="fa fa-pencil-square-o"></i>&ampnbsp@Html.DisplayByResource("Edit", "编辑")</a>
                    }
                    @if (((bool) ViewData["QX.Delete"]))
                    {
                        <a id="rdi_Delete" class="btn btn-default"><i class="fa fa-trash-o"></i>&ampnbsp@Html.DisplayByResource("Delete", "删除")</a>
                    }
                    @if (((bool) ViewData["QX.Export"]))
                    {
                        <a id="rdi_Export" class="btn btn-default"><i class="fa fa-file-excel-o"></i>&ampnbsp@Html.DisplayByResource("Export", "导出")</a>
                    }
                </div>
            </div>
        </div>
    </div>
</div>

```

2.3.7.2. 编辑界面 Form.cshtml 代码

编辑代码 Form.cshtml 代码如下图所示。



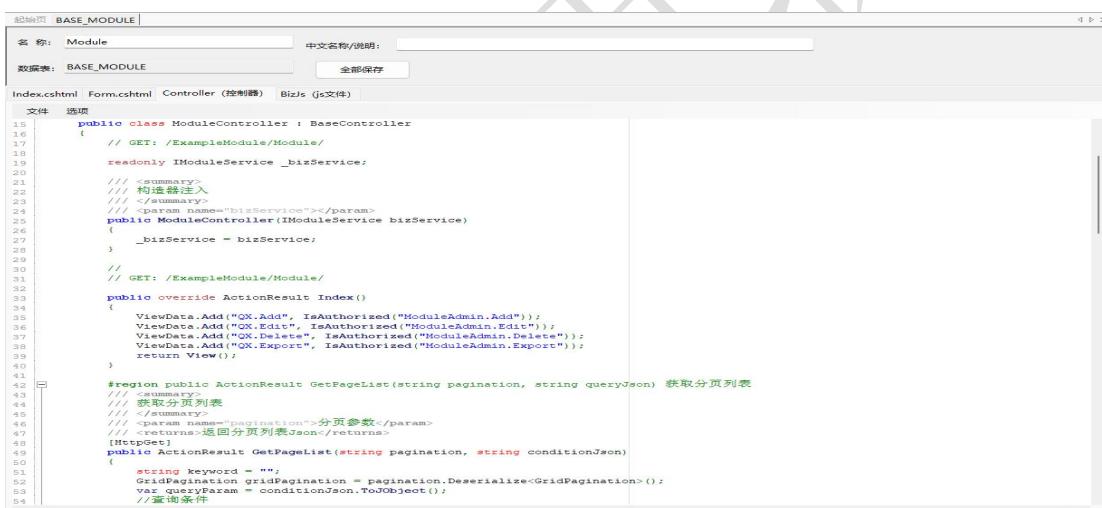
```

起始页 BASE_MODULE
名 称： Module 中文名称/说明：
数据表： BASE_MODULE 全部保存
Index.cshtml Form.cshtml Controller (控制器) BizJs (js文件)
文件 选项
1  @{
2     ViewBag.Title = "表单页面";
3     Layout = "~/Views/Shared/_Form.cshtml";
4 }
5 <div class="rdiframework-form-wrap" id="form">
6     <div class="col-xs-12 rdiframework-form-item">
7         <div class="rdiframework-form-item-title">父节点主键</div>
8         <input type="text" id="ParentId" name="ParentId" class="form-control" />
9     </div>
10    <div class="col-xs-12 rdiframework-form-item">
11        <div class="rdiframework-form-item-title">子系统ID</div>
12        <input type="text" id="SubSystemId" name="SubSystemId" class="form-control" />
13    </div>
14    <div class="col-xs-12 rdiframework-form-item">
15        <div class="rdiframework-form-item-title">编码 </div>
16        <input type="text" id="Code" name="Code" class="form-control" />
17    </div>
18    <div class="col-xs-12 rdiframework-form-item">
19        <div class="rdiframework-form-item-title"><font face="宋体">名称</font></div>
20        <input type="text" id="FullName" name="FullName" class="form-control" invalid="yes" errmsg="名称" checkexpression="NotNull" />
21    </div>
22    <div class="col-xs-12 rdiframework-form-item">
23        <div class="rdiframework-form-item-title">菜单分类</div>
24        <input type="text" id="Category" name="Category" class="form-control" />
25    </div>
26    <div class="col-xs-12 rdiframework-form-item">
27        <div class="rdiframework-form-item-title">模块类型<font face="宋体"></font></div>
28        <input type="text" id="ModuleType" name="ModuleType" value="1" invalid="yes" errmsg="模块类型 (1: WinForm、2: W" />
29    </div>
30    <div class="col-xs-12 rdiframework-form-item">
31        <div class="rdiframework-form-item-title">图标编号 </div>
32        <input type="text" id="ImageIndex" name="ImageIndex" class="form-control" />
33    </div>
34    <div class="col-xs-12 rdiframework-form-item">
35        <div class="rdiframework-form-item-title">选中状态图标编号</div>
36        <input type="text" id="SelectedImageIndex" name="SelectedImageIndex" class="form-control" />

```

2.3.7.3. Controller 控制器

Controller 控制器代码如下图所示。



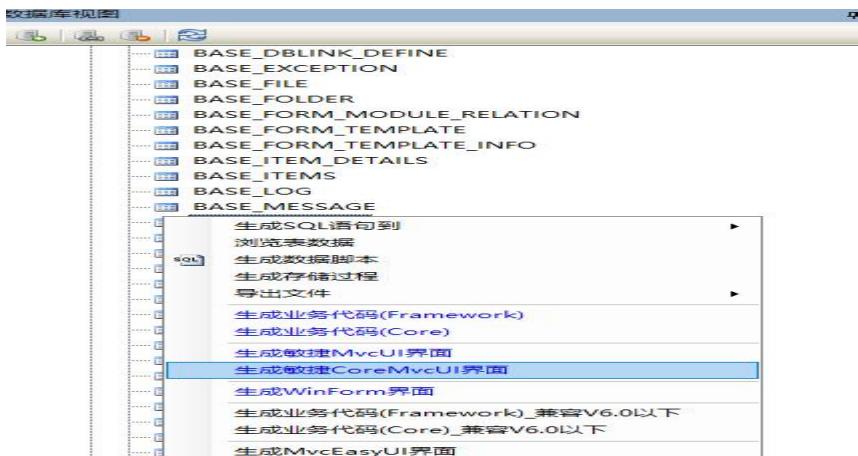
```

起始页 BASE_MODULE
名 称： Module 中文名称/说明：
数据表： BASE_MODULE 全部保存
Index.cshtml Form.cshtml Controller (控制器) BizJs (js文件)
文件 选项
15 public class ModuleController : BaseController
16 {
17     // GET: /ExampleModule/Module/
18     readonly IModuleService _bizService;
19
20     /// <summary>
21     /// 构造器注入
22     /// </summary>
23     /// <param name="bizService"></param>
24     public ModuleController(IModuleService bizService)
25     {
26         _bizService = bizService;
27     }
28
29     // GET: /ExampleModule/Module/
30
31     public override ActionResult Index()
32     {
33         ViewData.Add("QX.Add", IsAuthorized("ModuleAdmin.Add"));
34         ViewData.Add("QX.Edit", IsAuthorized("ModuleAdmin.Edit"));
35         ViewData.Add("QX.Delete", IsAuthorized("ModuleAdmin.Delete"));
36         ViewData.Add("QX.Export", IsAuthorized("ModuleAdmin.Export"));
37         return View();
38     }
39
40     [Region]
41     [Region]
42     // 获取分页列表
43     // <param name="pagination">分页参数</param>
44     // <param name="returnJson">返回分页列表Json</param>
45     // <returns>返回分页列表Json</returns>
46     [HttpGet]
47     public ActionResult GetPageList(string pagination, string conditionJson)
48     {
49         string keyword = "";
50         GridPagination gridPagination = pagination.Deserialize<GridPagination>();
51         var queryParam = conditionJson.To JObject();
52         //查询条件
53     }
54

```

2.3.8. 生成敏捷 CoreMvcUI 界面

“生成敏捷 CoreMvcUI 界面”功能模块用于生成.NET Core 版本的敏捷 MVC 的 UI 界面代码，选择一个数据表，单击右键，选择“生成敏捷 CoreMvcUI 界面”功能，如下图所示。



2.3.8.1. Index.cshtml 页面代码

敏捷 CoreMvcUI 的 Index.cshtml 页面代码如下图所示。

起步页 BASE_MODULE |

名 称: Module 中文名称/说明:

数据表: BASE_MODULE 全部保存

Index.cshtml Form.cshtml Controller (控制器) Bizs (js文件)

文件 选项

```

1  @{
2     ViewBag.Title = "Title";
3     Layout = "~/Views/Shared/_Admin.cshtml";
4 }
5 <div class="rdi-layout">
6     <div class="rdi-layout-center">
7         <div class="rdi-layout-wrap rdi-layout-wrap-notitle">
8             <div class="rdi-layout-tool-left">
9                 <div class="rdi-layout-tool-item">
10                    <div class="rdi-layout-tool-item">
11                        <input id="txt_Keyword" type="text" class="form-control" placeholder="请输入查询关键字" />
12                    </div>
13                    <div class="rdi-layout-tool-item">
14                        <a href="#" id="btn_Search" class="btn btn-primary btn-sm"><i class="fa fa-search"></i>&ampnbsp查询</a>
15                    </div>
16                </div>
17            <div class="rdi-layout-tool-right">
18                <div class="btn-group btn-group-sm">
19                    <a href="#" id="rdi_refresh" class="btn btn-default"><i class="fa fa-refresh"></i></a>
20                </div>
21                <div class="btn-group btn-group-sm">
22                    @if (((bool) ViewData["QX.Add"]))
23                    {
24                        <a href="#" id="rdi_add" class="btn btn-default"><i class="fa fa-plus"></i>&ampnbsp@Html.DisplayByResource("Add", "新增")</a>
25                    }
26                    @if (((bool) ViewData["QX.Edit"]))
27                    {
28                        <a href="#" id="rdi_edit" class="btn btn-default"><i class="fa fa-pencil-square-o"></i>&ampnbsp@Html.DisplayByResource("Edit", "编辑")
29                    }
30                    @if (((bool) ViewData["QX.Delete"]))
31                    {
32                        <a href="#" id="rdi_delete" class="btn btn-default"><i class="fa fa-trash-o"></i>&ampnbsp@Html.DisplayByResource("Delete", "删除")
33                    }
34                    @if (((bool) ViewData["QX.Export"]))
35                    {
36                        <a href="#" id="rdi_export" class="btn btn-default"><i class="fa fa-file-excel-o"></i>&ampnbsp@Html.DisplayByResource("Export", "导出")
37                    }
38                </div>
39            </div>
40        </div>
41    </div>
42</div>
```

2.3.8.2. 编辑界面 Form.cshtml 代码

编辑代码 Form.cshtml 代码如下图所示。

起始页 BASE_MODULE

名 称： Module 中文名称/说明：

数据表： BASE_MODULE 全部保存

```

Index.cshtml Form.cshtml Controller (控制器) BizJs (js文件)
文件 选项
1  @{
2   ViewBag.Title = "菜单页面";
3   Layout = "~/Views/Shared/_Form.cshtml";
4 }
5 <div class="rdiframework-form-wrap" id="form">
6   <div class="col-xs-12 rdiframework-form-item">
7     <div class="rdiframework-form-item-title">父节点主键</div>
8     <input type="text" id="ParentId" name="ParentId" class="form-control" />
9   </div>
10  <div class="col-xs-12 rdiframework-form-item">
11    <div class="rdiframework-form-item-title">子系统ID</div>
12    <input type="text" id="SubSystemId" name="SubSystemId" class="form-control" />
13  </div>
14  <div class="col-xs-12 rdiframework-form-item">
15    <div class="rdiframework-form-item-title">编码 </div>
16    <input type="text" id="Code" name="Code" class="form-control" />
17  </div>
18  <div class="col-xs-12 rdiframework-form-item">
19    <div class="rdiframework-form-item-title">名称<font face="宋体"></font> </div>
20    <input type="text" id="FullName" name="FullName" class="form-control" isvalid="yes" errormsg="名称" checkedexpression="NotNull" />
21  </div>
22  <div class="col-xs-12 rdiframework-form-item">
23    <div class="rdiframework-form-item-title">菜单分类</div>
24    <input type="text" id="Category" name="Category" class="form-control" />
25  </div>
26  <div class="col-xs-12 rdiframework-form-item">
27    <div class="rdiframework-form-item-title">模块类型<font face="宋体"></font> </div>
28    <input type="number" id="ModuleType" name="ModuleType" class="form-control" value="1" isvalid="yes" errormsg="模块类型 (1: WinForm、2: W" />
29  </div>
30  <div class="col-xs-12 rdiframework-form-item">
31    <div class="rdiframework-form-item-title">图标编号 </div>
32    <input type="text" id="ImageIndex" name="ImageIndex" class="form-control" />
33  </div>
34  <div class="col-xs-12 rdiframework-form-item">
35    <div class="rdiframework-form-item-title">选中状态图标编号</div>
36    <input type="text" id="SelectedImageIndex" name="SelectedImageIndex" class="form-control" />

```

2.3.8.3. Controller 控制器

Controller 控制器代码如下图所示。

起始页 BASE_MODULE

名 称： Module 中文名称/说明：

数据表： BASE_MODULE 全部保存

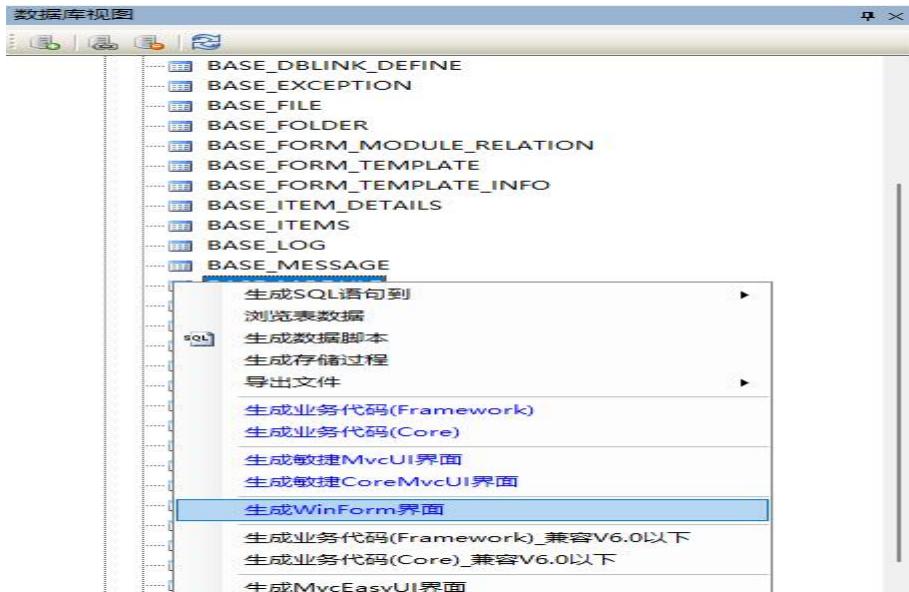
```

Index.cshtml Form.cshtml Controller (控制器) BizJs (js文件)
文件 选项
15  public class ModuleController : BaseController
16  {
17    // GET: /ExampleModule/Module/
18    readonly IModuleService _bizService;
19
20    /// <summary>
21    /// 构造器注入
22    /// </summary>
23    /// <param name="bizService"></param>
24    public ModuleController(IModuleService bizService)
25    {
26      _bizService = bizService;
27    }
28
29
30
31    // GET: /ExampleModule/Module/
32
33    public override ActionResult Index()
34    {
35      ViewData.Add("OK.Add", IsAuthorized("ModuleAdmin.Add"));
36      ViewData.Add("OK.Edit", IsAuthorized("ModuleAdmin.Edit"));
37      ViewData.Add("OK.Delete", IsAuthorized("ModuleAdmin.Delete"));
38      ViewData.Add("OK.Export", IsAuthorized("ModuleAdmin.Export"));
39      return View();
40    }
41
42    #region public ActionResult GetPageList(string pagination, string queryJson) 获取分页列表
43    /// <summary>
44    /// 获取分页列表
45    /// </summary>
46    /// <param name="pagination">分页参数</param>
47    /// <returns>返回分页列表Json</returns>
48    [HttpGet]
49    public ActionResult Getpagelist(string pagination, string conditionJson)
50    {
51      string keyword = "";
52      GridPagination gridpagination = pagination.Deserialize<GridPagination>();
53      ViewBagParam = conditionJson.DeserializeObject();
54      //查询条件

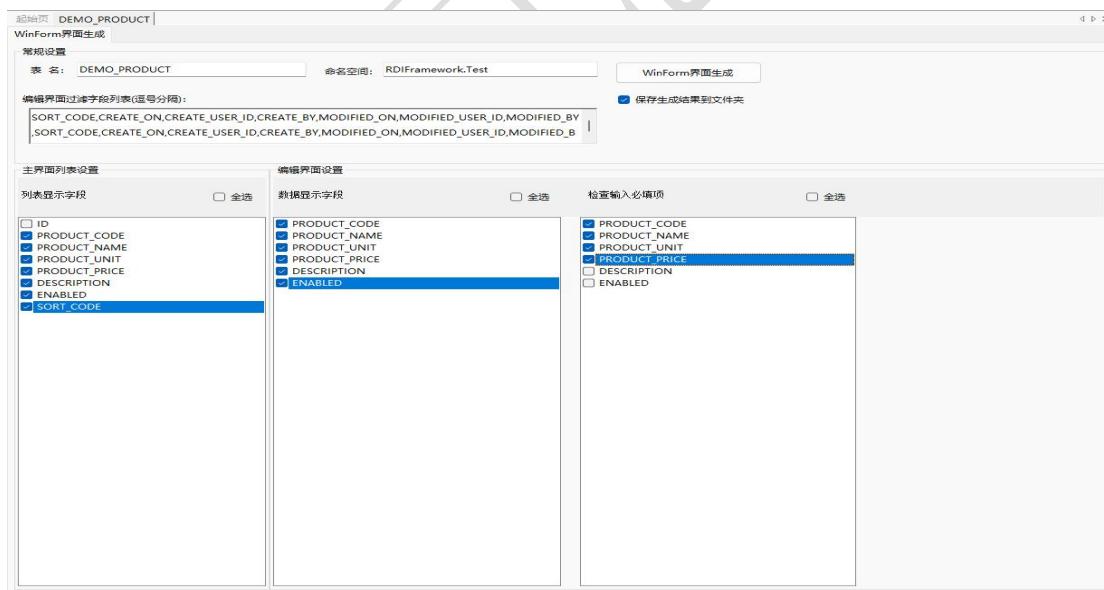
```

2.3.9. 生成 WinForm 界面

“生成 WinForm 界面”用于生成 WinForm 的 UI 界面代码，这样就可以大大缩小界面的开发时间，对生成的 UI 代码只需要稍微做调整即可完成一个模块的快速开发，生成 WinForm 界面功能如下图所示。



单击“生成 WinForm 界面”功能，即可打开当前表的 WinForm 界面生成设置窗口，如下图所示。



在上面的 WinForm 界面生成窗口，可以设置主界面列表需要展示的字段，编辑或新增界面需要处理的字段以及非空验证的字段，可以把生成的结果直接保存在文件夹中，也可以直接把生成的结果展示在 Tab 标签中。单击 WinForm 界面生成按钮，即可对所选表按设置进行生成，主要生成主界面与主界面的页面设计代码、编辑界面与编辑界面的页面设计代码，如下图所示。

1、主界面页面设计代码

```

1 起始页 | SYS_NEWS | DemoProduct | FrmDemoProductAdmin.cs | FrmDemoProductAdmin.Designer.cs | FrmDemoProductEdit.Designer.cs | FrmDemoProductEdit.cs | 
2 文件 选项
3     namespace RDIFramework.Test
4     {
5         partial class FrmDemoProductAdmin
6         {
7             /// <summary>
8             /// Required designer variable.
9             /// </summary>
10            private System.ComponentModel.IContainer components = null;
11
12            /// <summary>
13            /// Clean up any resources being used.
14            /// </summary>
15            /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
16            protected override void Dispose(bool disposing)
17            {
18                if (disposing && (components != null))
19                {
20                    components.Dispose();
21                }
22                base.Dispose(disposing);
23            }
24
25            #region Windows Form Designer generated code
26
27            /// <summary>
28            /// Required method for Designer support - do not modify
29            /// the contents of this method with the code editor.
30            /// </summary>
31            private void InitializeComponent()
32            {
33                 this.components = new System.ComponentModel.Container();
34                 this.layoutControl1 = new DevExpress.XtraLayout.LayoutControl();
35                 this.btnFind = new RDIFramework.Controls.UcButton();
36                 this.btnClose = new RDIFramework.Controls.UcButton();
37                 this.btnPrint = new RDIFramework.Controls.UcButton();
38                 this.btnRefresh = new RDIFramework.Controls.UcButton();
39                 this.btnAdd = new RDIFramework.Controls.UcButton();
40
41                 ((System.ComponentModel.ISupportInitialize)(this.layoutControl1)).BeginInit();
42                 this.layoutControl1.SuspendLayout();
43
44            #endregion Windows Form Designer generated code
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81

```

2、主界面代码

```

1 起始页 | DEMO_PRODUCT | FrmDemoProductAdmin.cs | FrmDemoProductAdmin.Designer.cs | FrmDemoProductEdit.Designer.cs | FrmDemoProductEdit.cs | 
2 文件 选项
3     public partial class FrmDemoProductAdmin : BaseForm
4     {
5         public FrmDemoProductAdmin()
6         {
7             InitializeComponent();
8         }
9
10        List<DemoProductEntity> listData = new List<DemoProductEntity>();
11        private string searchValue = "";
12
13        #region public override string EntityId 主键
14        /// <summary>
15        /// 主键
16        /// </summary>
17        public override string EntityId => BasePageLogic.GetDataGridViewEntityId(grdView, "Id");
18
19        #endregion
20
21        #region 权限控制部分
22        /// <summary>
23        /// 本模块的添加权限
24        /// </summary>
25        private bool permissionAdd = false;
26
27        /// <summary>
28        /// 本模块的修改权限
29        /// </summary>
30        private bool permissionEdit = false;
31
32        /// <summary>
33        /// 本模块的删除权限
34        /// </summary>
35        private bool permissionDelete = false;
36
37        /// <summary>
38        /// 本模块的打印权限
39        /// </summary>
40        private bool permissionPrint = false;
41
42        #region public override void GetPermission() 获得权限
43        /// <summary>
44        /// 获得权限
45        /// </summary>
46        public override void GetPermission()
47        {
48            this.permissionAdd = this.IsAuthorized("DemoProductAdmin.Add");
49            this.permissionEdit = this.IsAuthorized("DemoProductAdmin.Edit");
50            this.permissionDelete = this.IsAuthorized("DemoProductAdmin.Delete");
51        }
52
53
54
55
56
57
58
59
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81

```

3、编辑或新增界面设计代码

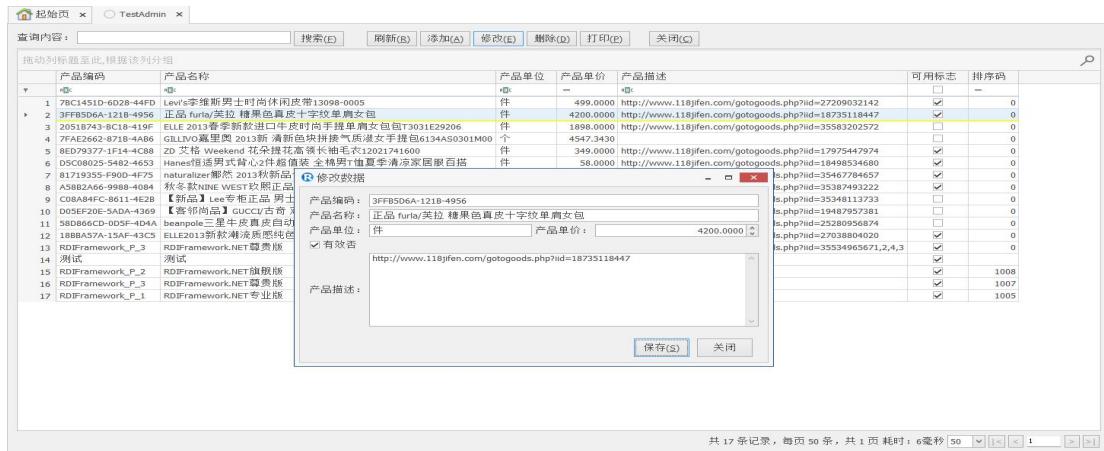
```
1  namespace RDIFramework.Test
2  {
3      partial class FrmDemoProductEdit
4      {
5          /// <summary>
6          /// Required designer variable.
7          /// </summary>
8          private System.ComponentModel.IContainer components = null;
9
10         /// <summary>
11         /// Clean up any resources being used.
12         /// </summary>
13         //<param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
14         protected override void Dispose(bool disposing)
15         {
16             if (disposing && (components != null))
17             {
18                 components.Dispose();
19             }
20             base.Dispose(disposing);
21         }
22
23     #region Windows Form Designer generated code
24
25     /// <summary>
26     /// Required method for Designer support - do not modify
27     /// the contents of this method with the code editor.
28     /// </summary>
29     private void InitializeComponent()
30     {
31         this.components = new System.ComponentModel.Container();
32         //1.创建LayoutControl对象
33         this.layoutControl = new DevExpress.XtraLayout.LayoutControl();
34         this.layoutControlGroup = new DevExpress.XtraLayout.LayoutControlGroup();
35
36         //<br> 34 11 2017 8:33 PM
```

4、编辑或新增界面代码

```
16 partial class FrmDemoProductEdit : BaseEditForm
17 {
18     private DemoProductEntity currentEntity = null;
19
20     #region 构造函数
21     public FrmDemoProductEdit()
22     {
23         InitializeComponent();
24     }
25
26     /// <summary>
27     /// 构造函数，修改时调用
28     /// </summary>
29     /// <param name="keyId">当前待修改数据的主键</param>
30     public FrmDemoProductEdit(string keyId) : this()
31     {
32         this.EntityId = keyId;
33     }
34
35     #endregion
36
37     FormValidator formValidate = null;
38
39     public override void FormDataValidator()
40     {
41         //界面通用验证封装
42         formValidate = new FormValidator(this, new DevExpress.XtraEditors.DXErrorProvider.DXErrorProvider());
43         Dictionary<Control, string> dic = new Dictionary<Control, string>;
44
45         //以下为参考
46         //!{ ProductCode, "isLength(1-50)" },
47         //!{ ProductName, "isLengthNotNull(3-50)" },
48         //!{ ProductPrice, "isNotNull" },
49         //!{ ProductUnit, "isNotNull" },
50         //!{ ProductName, "nonNull" },
51         //!{ ProductName, "nonNull" },
52         //!{ ProductUnit, "nonNull" },
53         //!{ ProductPrice, "nonNull" },
54     }
55
56     ValidatorHelper.ForRegex(formValidate, dic);
57
58
59     #region public override bool CheckInput() 页面输入检查
60     /// <summary>
61     /// 页面输入检查
62     /// </summary>
63     /// <returns>检查通过返回true;否则返回false</returns>
64     public override bool CheckInput()
65     {
66
67
68
69
70
71
72
73
74
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
89
89
90
91
92
93
94
95
96
97
98
99
99
100
101
102
103
104
105
106
107
108
109
109
110
111
112
113
114
115
116
117
118
119
119
120
121
122
123
124
125
126
127
128
129
129
130
131
132
133
134
135
136
137
138
139
139
140
141
142
143
144
145
146
147
148
149
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
169
170
171
172
173
174
175
176
177
178
179
179
180
181
182
183
184
185
186
187
187
188
189
189
190
191
192
193
194
195
196
197
197
198
199
199
200
201
202
203
204
205
206
207
207
208
209
209
210
211
212
213
214
215
215
216
217
217
218
218
219
219
220
221
222
223
223
224
224
225
225
226
226
227
227
228
228
229
229
230
230
231
231
232
232
233
233
234
234
235
235
236
236
237
237
238
238
239
239
240
240
241
241
242
242
243
243
244
244
245
245
246
246
247
247
248
248
249
249
250
250
251
251
252
252
253
253
254
254
255
255
256
256
257
257
258
258
259
259
260
260
261
261
262
262
263
263
264
264
265
265
266
266
267
267
268
268
269
269
270
270
271
271
272
272
273
273
274
274
275
275
276
276
277
277
278
278
279
279
280
280
281
281
282
282
283
283
284
284
285
285
286
286
287
287
288
288
289
289
290
290
291
291
292
292
293
293
294
294
295
295
296
296
297
297
298
298
299
299
300
300
301
301
302
302
303
303
304
304
305
305
306
306
307
307
308
308
309
309
310
310
311
311
312
312
313
313
314
314
315
315
316
316
317
317
318
318
319
319
320
320
321
321
322
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
14
```

对于生成的代码放在框架中打开测试页面，可以看到生成的主界面代码如下。

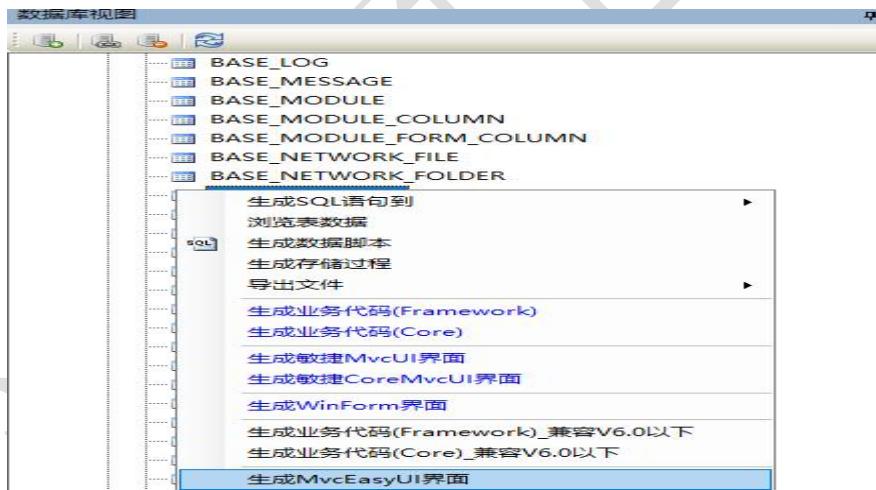
整个主界面已经集成了关键字搜索、刷新、添加、修改、删除、打印、分页、分组、表格内搜索等功能。单击添加或修改按钮即可打开编辑界面，如下图所示。



整个界面的生成如果不满意可以直接通过 VisualStudio 设计器直接拖动控件(使用了 layoutControl 布局，拖动非常方便)即可快速完成布局，非常之方便与高效。调试完成后即可在框架中定义好模块自动加载。

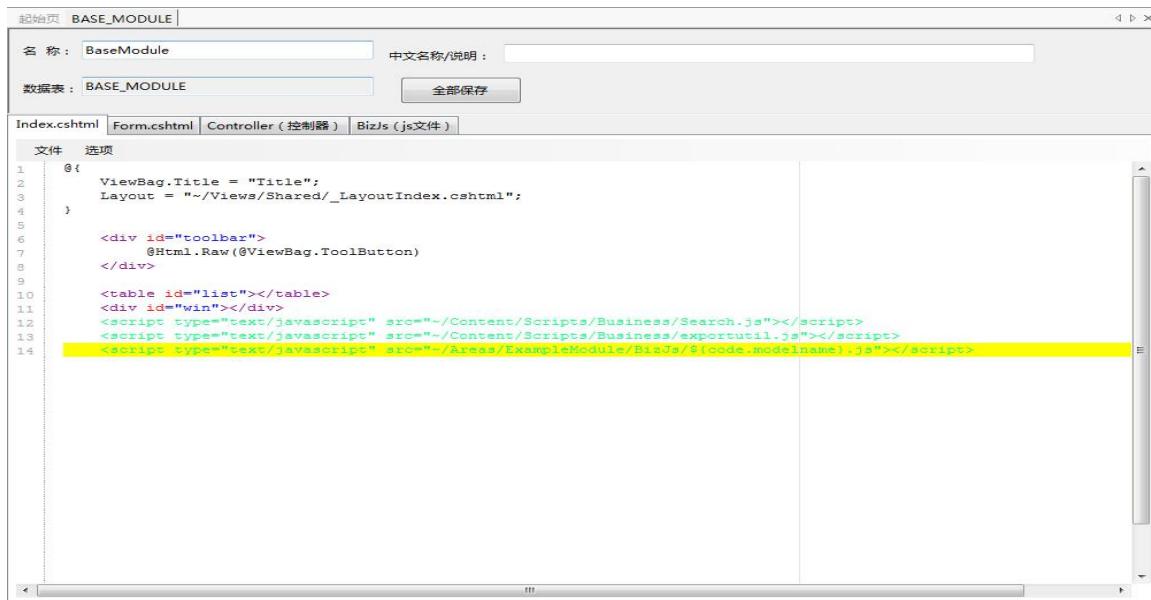
2.3.10. 生成 MvcEasyUI 界面

“生成 MvcEasyUI 界面”可以生成针对 MVC 的 EasyUI 风格的界面代码，这样就可以大大缩小界面的开发时间，对生成的 UI 代码做适量的修改即可。选择一个数据表，单击右键，选择“生成 MvcEasyUI 界面”功能，如下图所示。



2.3.10.1. Index.cshtml 页面代码

MvcEasyUI 的 Index.cshtml 页面代码如下图所示。



The screenshot shows the RDIFramework.NET interface for editing a view file. The title bar says "起始页 BASE_MODULE". The main area has tabs for "Index.cshtml", "Form.cshtml", "Controller (控制器)", and "BizJs (js文件)". The "Index.cshtml" tab is selected. The code editor displays the following C# Razor code:

```

1  ViewBag.Title = "Title";
2  Layout = "~/Views/Shared/_LayoutIndex.cshtml";
3
4
5
6  <div id="toolbar">
7      @Html.Raw(@ViewBag.ToolButton)
8  </div>
9
10 <table id="list"></table>
11 <div id="win"></div>
12 <script type="text/javascript" src("~/Content/Scripts/Business/Search.js")></script>
13 <script type="text/javascript" src "~/Content/Scripts/Business/exportutil.js"></script>
14 <script type="text/javascript" src "~/Areas/ExampleModule/BizJs/%(code.modulename%).js"></script>

```

The last line of code, which includes a placeholder for the module name, is highlighted with a yellow background.

2.3.10.2. 编辑界面 Form.cshtml 代码

编辑代码 Form.cshtml 代码如下图所示。



The screenshot shows the RDIFramework.NET interface for editing a view file. The title bar says "起始页 BASE_MODULE". The main area has tabs for "Index.cshtml", "Form.cshtml", "Controller (控制器)", and "BizJs (js文件)". The "Form.cshtml" tab is selected. The code editor displays the following C# Razor code:

```

1  <form id="uniform">
2      <table class="grid2">
3          <tr>
4              <td>父节点主键: </td>
5              <td>
6                  <input type="text" id="Parentid" name="Parentid" class="txt02" />
7              </td>
8          </tr>
9          <tr>
10             <td>子系统主键（对应数据字典代码subSystem的取值主键）: </td>
11             <td>
12                 <input type="text" id="Subsystemid" name="Subsystemid" class="txt02" />
13             </td>
14         </tr>
15         <tr>
16             <td>编号: </td>
17             <td>
18                 <input type="text" id="Code" name="Code" class="txt02" />
19             </td>
20         </tr>
21         <tr>
22             <td>名称: </td>
23             <td>
24                 <input type="text" id="Fullname" name="Fullname" class="txt02" required="" />
25             </td>
26         </tr>
27         <tr>
28             <td>菜单分类: </td>
29             <td>
30                 <input type="text" id="Category" name="Category" class="txt02" />
31             </td>
32         </tr>
33         <tr>
34             <td>模块的类型 (1: WinForm、2: WebForm、3: WinForm与WebForm都支持、6: 其他): </td>
35             <td>
36             </td>
37         </tr>

```

The first input field in the first row is highlighted with a yellow background.

2.3.10.3. Controller 控制器

Controller 控制器代码如下图所示。

```

起始页 BASE_MODULE | 
名称: BaseModule 中文名称/说明: 
数据表: BASE_MODULE 全部保存 
Index.cshtml Form.cshtml Controller (控制器) BizJs (js文件) 
文件 选项 
25 //</summary> 
26 public class BaseModuleController : PublicController 
27 { 
28     // 
29     // GET: /ExampleModule/BaseModule/ 
30 
31     /// <summary> 
32     /// 加载工具栏 
33     /// </summary> 
34     /// <returns>工具栏HTML</returns> 
35     public override string BuildToolBarButtons() 
36     { 
37         var sb = new StringBuilder(); 
38         const string linkbtnTemplate = "<a id=\"a_{0}\" class=\"easyui-linkbutton\" style=\"float:left\" plain=" 
39         sb.Append("<a id=\"a_refresh\" class=\"easyui-linkbutton\" style=\"float:left\" plain=\"true\" href=\"j"); 
40         sb.Append("sb.Append("<div class='datagrid-btn-separator'></div>"); 
41         sb.Append(string.Format(linkbtnTemplate, "add", "icon16_table_add", IsAuthorized("BaseModuleAdmin.Add"))); 
42         sb.Append(string.Format(linkbtnTemplate, "edit", "icon16_table_edit", IsAuthorized("BaseModuleAdmin.Edit"))); 
43         sb.Append(string.Format(linkbtnTemplate, "delete", "icon16_table_delete", IsAuthorized("BaseModuleAdmin.Delete"))); 
44         sb.Append("<div class='datagrid-btn-separator'></div>"); 
45         sb.Append(string.Format(linkbtnTemplate, "export", "icon16_table_export", IsAuthorized("BaseModuleAdmin.Export"))); 
46         sb.Append("<div class='datagrid-btn-separator'></div>"); 
47         sb.Append(string.Format(linkbtnTemplate, "search", "icon16_table_filter", IsAuthorized("BaseModuleAdmin.Search"))); 
48         return sb.ToString(); 
49     } 
50 
51     IDbProvider dbProvider 
52     { 
53         get 
54     } 

```

2.3.10.4. js 页面代码

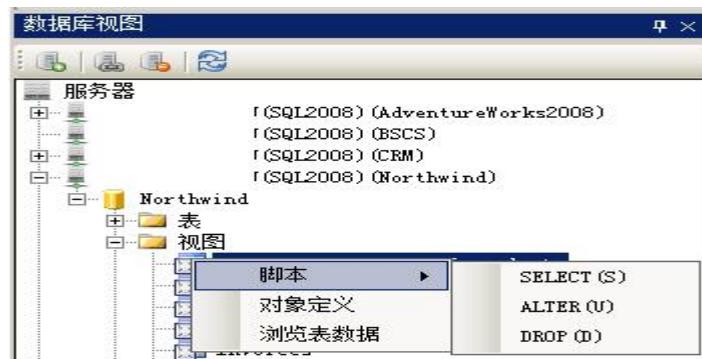
Js 页面代码如下图所示。

```

起始页 BASE_MODULE | 
名称: BaseModule 中文名称/说明: 
数据表: BASE_MODULE 全部保存 
Index.cshtml Form.cshtml Controller (控制器) BizJs (js文件) 
文件 选项 
1 var _gridlist, 
2     actionURL = '/ExampleModule/BaseModule/', 
3     formurl = '/ExampleModule/BaseModule/Form'; 
4 
5 $(function () { 
6     autoResize({_gridlist: $('#list'), gridType: 'datagrid', callback: grid.bind, height: 5}); 
7     $('#a_add').attr('onclick', 'CRUD.add()'); 
8     $('#a_delete').attr('onclick', 'CRUD.delete()'); 
9     $('#a_export').attr('onclick', 'CRUD.exportData()'); 
10    $('#a_search').attr('onclick', 'CRUD.search()'); 
11    $('#a_refresh').attr('onclick', 'CRUD.refresh()'); 
12}); 
13 
14 var grid = { 
15     bind: function (winSize) { 
16         _gridlist = $('#list').datagrid({ 
17             url: actionURL + 'GridPageListJson', 
18             toolbar: '#toolbar', 
19             title: "数据列表", 
20             iconCls: 'icon16_table', 
21             width: winSize.width, 
22             height: winSize.height, 
23             nowrap: false, //折行 
24             rownumbers: true, //行号 
25             striped: true, //隔行变色 
26             idField: 'ID',//主键 
27             singleSelect: true, //单选 
28             onRowContextMenu: pageContextMenu.createDataGridContextMenu, 
29             onDoubleClickRow: function (rowIndex, rowData) { 
30                 document.getElementById('a_edit').click(); 
31             } 
32         }); 
33     } 
34 }; 
35 
```

2.4. 视图管理

连接到一个已注册的服务器，我们可以对当前服务器所选“数据库”的视图进行相应的操作，主要包括：脚本生成（包括：SELECT 语句、ALTER 语句、DROP 语句）、对象定义、浏览表数据等。



2.4.1. 生成脚本

“脚本生成”可以生成当前所选视图的 SELECT 语句、ALTER 语句、DROP 语句。

1) 生成 SELECT 语句。

起始页 Orders Qry查询.sql

```

1 select
2   > [OrderID],
3   > [CustomerID],
4   > [EmployeeID],
5   > [OrderDate],
6   > [RequiredDate],
7   > [ShippedDate],
8   > [ShipVia],
9   > [Freight],
10  > [ShipName],
11  > [ShipAddress],
12  > [ShipCity],
13  > [ShipRegion],
14  > [ShipPostalCode],
15  > [ShipCountry],
16  > [CompanyName],
17  > [Address],
18  > [City],
19  > [Region],
20  > [PostalCode],
21  > [Country]
22  from [Orders Qry]
23

```

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia
10643	ALFKI	6	1997-08-25	1997-09-22	1997-09-02	1
10692	ALFKI	4	1997-10-03	1997-10-31	1997-10-13	2
10702	ALFKI	4	1997-10-13	1997-11-24	1997-10-21	1
10835	ALFKI	1	1998-01-15	1998-02-12	1998-01-21	3
10952	ALFKI	1	1998-03-16	1998-04-27	1998-03-24	1
11011	ALFKI	3	1998-04-09	1998-05-07	1998-04-13	1
10926	ANATR	4	1998-03-04	1998-04-01	1998-03-11	3

命令已成功完成。

2) 生成 ALTER 语句。

起始页 Orders Qry修改.sql

```

1 ALTER VIEW "Orders Qry" AS
2 SELECT Orders.OrderID, Orders.CustomerID, Orders.EmployeeID, Orders.OrderDate, Orders.RequiredDate,
3   > Orders.ShippedDate, Orders.ShipVia, Orders.Freight, Orders.ShipName, Orders.ShipAddress, Orders.ShipCity,
4   > Orders.ShipRegion, Orders.ShipPostalCode, Orders.ShipCountry,
5   > Customers.CompanyName, Customers.Address, Customers.City, Customers.Region, Customers.PostalCode, Customers.Country
6 FROM Customers INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID
7

```

命令成功完成 (所影响的行数为: -1 行)

生成当前查询SQL语句的拼接代码
生成当前查询结果的数据脚本
脚本中断

命令已成功完成。

3) 生成 DROP 语句。

```

1 USE Northwind
2 GO
3
4
5 /****** Object: VIEW - Orders Qry Script Date: 2013-07-29 17:48:35 *****/
6 DROP VIEW [Orders Qry]
7 GO

```

2.4.2. 对象定义

对象定义就是生成当前视图的创建脚本，如下图所示。

```

CREATE VIEW "Orders Qry" AS
SELECT Orders.OrderID, Orders.CustomerID, Orders.EmployeeID, Orders.OrderDate, Orders.RequiredDate,
       Orders.ShippedDate, Orders.ShipVia, Orders.Freight, Orders.ShipName, Orders.ShipAddress, Orders.ShipCity,
       Orders.ShipRegion, Orders.ShipPostalCode, Customers.CompanyName, Customers.Address, Customers.PostalCode, Customers.Country
FROM Customers INNER JOIN Orders ON Cu

```

数据库中已存在名为 'Orders Qry' 的对象。

命令执行失败！

2.4.3. 浏览表数据

“浏览表数据”就是查看当前所选视图的数据，如下图所示。

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	Freight	ShipName	ShipAddress	ShipRegion
10643	ALFKI	6	1997-08-25	1997-09-22	1997-09-02	29.4800	Alfreds Futterkiste	Obere Str. 57	Bremen
10692	ALFKI	4	1997-10-03	1997-10-31	1997-10-13	2	61.0200	Alfred's Futterkiste	Bremen
10702	ALFKI	4	1997-10-13	1997-11-24	1997-10-21	1	23.9400	Alfred's Futterkiste	Bremen
10835	ALFKI	1	1998-01-15	1998-02-12	1998-01-21	3	69.5300	Alfred's Futterkiste	Bremen
10952	ALFKI	1	1998-03-16	1998-04-27	1998-03-24	1	40.4200	Alfred's Futterkiste	Bremen
11011	ALFKI	3	1998-04-09	1998-05-07	1998-04-13	1	1.2100	Alfred's Futterkiste	Bremen
10926	ANATR	4	1998-03-04	1998-04-01	1998-03-11	3	39.9200	Ana Trujillo Emparedados y helados	Avda. de la Constitución 2222
10759	ANATR	3	1997-11-28	1997-12-26	1997-12-12	3	11.9900	Ana Trujillo Emparedados y helados	Avda. de la Constitución 2222
10625	ANATR	3	1997-08-08	1997-09-05	1997-08-14	1	43.9000	Ana Trujillo Emparedados y helados	Avda. de la Constitución 2222
10308	ANATR	7	1996-09-18	1996-10-16	1996-09-24	3	1.6100	Ana Trujillo Emparedados y helados	Avda. de la Constitución 2222
10365	ANTON	3	1996-11-27	1996-12-25	1996-12-02	2	22.0000	Antonio Moreno Taquería	Mataderos 2312
10507	ANTON	7	1997-04-15	1997-05-13	1997-04-22	1	47.4500	Antonio Moreno Taquería	Mataderos 2312
10535	ANTON	4	1997-05-13	1997-06-10	1997-05-21	1	15.8400	Antonio Moreno Taquería	Mataderos 2312
10677	ANTON	1	1997-09-22	1997-10-20	1997-09-26	3	4.0300	Antonio Moreno Taquería	Mataderos 2312
10673	ANTON	7	1997-06-10	1997-07-17	1997-06-20	2	64.8400	Antonio Moreno Taquería	Mataderos 2312

2.5. 2.5 存储过程管理

连接到一个已注册的服务器，我们可以对当前服务器所选“数据库”的存储过程进行相应的操作，主要包括：脚本生成（包括：ALTER语句、DROP语句）、对象定义等。



2.5.1. 脚本生成

“脚本生成”可以生成当前所选存储过程的 ALTER 语句、DROP 语句。

- 1) 生成 ALTER 语句。

```
起始页 CustOrderHist修改.sql
1 ALTER PROCEDURE [dbo].[CustOrderHist] @CustomerID nchar(5)
2 AS
3 SELECT ProductName, Total=SUM(Quantity)
4 FROM Products P, [Order Details] OD, Orders O, Customers C
5 WHERE C.CustomerID = @CustomerID
6 AND C.CustomerID = O.CustomerID AND O.OrderID = OD.OrderID AND OD.ProductID = P.ProductID
7 GROUP BY ProductName
8
```

- 2) 生成 DROP 语句。

```
起始页 CustOrderHist删除.sql
1 USE Northwind
2 GO
3
4 **** Object: PROCEDURE CustOrderHist Script Date: 2013-07-29 18:07:33 ****
5 DROP PROCEDURE CustOrderHist
6 GO
7
8
```

2.5.2. 对象定义

对象定义就是生成当前存储过程的创建脚本，如下图所示。

```
起始页 CustOrderHist定义.sql
1 CREATE PROCEDURE [dbo].[CustOrderHist] @CustomerID nchar(5)
2 AS
3 SELECT ProductName, Total=SUM(Quantity)
4 FROM Products P, [Order Details] OD, Orders O, Customers C
5 WHERE C.CustomerID = @CustomerID
6 AND C.CustomerID = O.CustomerID AND O.OrderID = OD.OrderID AND OD.ProductID = P.ProductID
7 GROUP BY ProductName
```

3. PowerDesigner Objects 导航区

RDIFramework.NET 代码生成器最大的特点就是不仅可以通过连接到数据库来生成代码，还可以直接通过 PowerDesigner（下面简称 PD）设计源文件来生成代码，通过 PD 源文件来生成代码最大的好处就是不依赖于具体的数据库类型。我们设计数据库一般都是借助于 PD 工具进行设计，使用 PD 设计数据库有什么好处呢？PowerDesigner 是 Sybase 公司开发的数据库建模 CASE 工具，它是一种数据库开发环境专门提供数据库的需求分析、概念数据模型

CDM 设计、物理数据模型 PDM 设计以及数据库建表、建索引、建视图、建存储过程、建触发器等功能，同时还可做到模型数据共享，修改、设计、沟通方便。

PowerDesigner Objects 导航区可对多个 PD 设计文件进行集中管理。PD 导航区如下图所示。



在上图在我们加载了 3 个 PD 设计源文件，加载成功后，我们就可以对加载的源文件进行代码的生成。在 PowerDesigner Objects 导航区，PD 源文件是以树型结构加载显示的，选择不同的树节点，会有不同的功能。树节点类型主要分为以下几类：

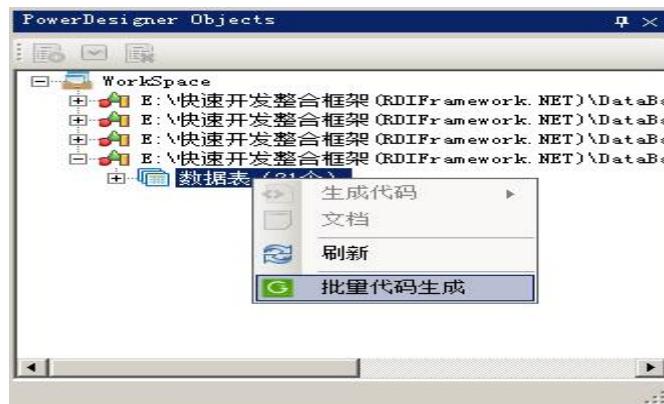
- 1) 工作区域 (WorkSpace) 树节点：在当前节点，我们可以添加 PD 设计文档到 WorkSpace 中。



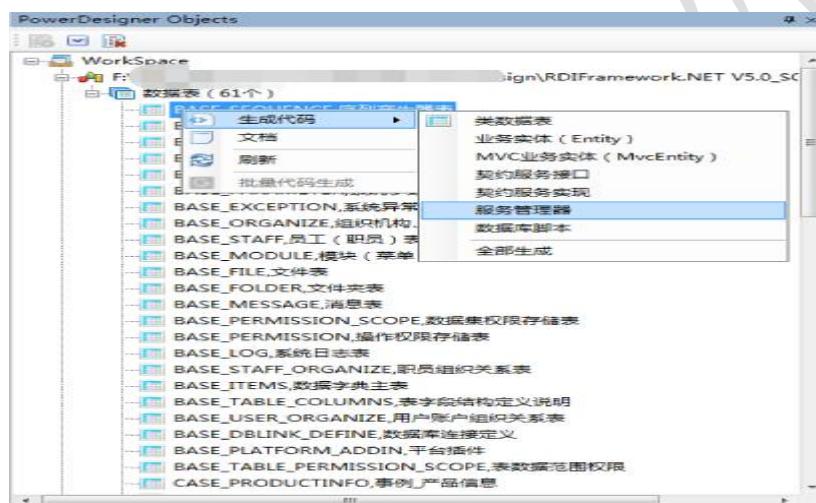
- 2) PD 源文件树节点：在当前节点，我们可以加载当前 PD 设计文件、展开当前 PD 设计文件、移除当前 PD 设计文件。



- 3) 数据表父树节点：在当前节点，我们可以进行批量代码生成（生成所有表的全部代码），刷新当前数据表。



4) 具体的表节点：选择具体的表节点，我们可以生成代码（类数据表、业务实体、契约服务接口、契约服务、服务管理器、全部生成）、生成当前表的数据库文档、刷新等。



3.1. PD 设计文件管理

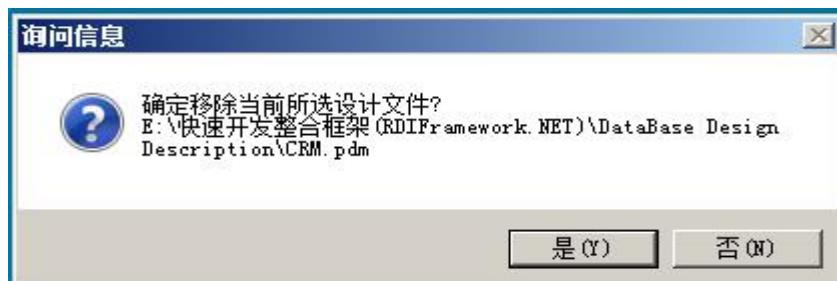
PD 设计文件管理包括添加 PD 设计文件到 PD 管理工作区、移除已添加到 PD 管理工作区中的 PD 设计文件、加载 PD 设计文件、刷新 PD 设计文件等。

3.1.1. 添加 PD 设计文件

要通过 PD 设计文件进行代码生成，必须先把 PD 设计文件添加到 PD 管理工作区中，添加 PD 设计文件，需要鼠标选中“WorkSpace”树节点，然后单击 PowerDesigner Objects 导航区的工具栏上的“增加 PowerDesigner 设计文件”按钮，图标为：，在弹出的文件选择对话框中，选择一个 PD 设计文件，即可把 PD 设计文件加载到工作区中。

3.1.2. 移除 PD 设计文件

对于已成功添加的 PD 设计文件，如果不再需要，我们可以把它从工作区中移除，方法就是选择需要移除的“PD 设计文件”后，单击 PowerDesigner Objects 导航区的工具栏上的“移除 PowerDesigner 设计文件”按钮，图标为：，在弹出的询问对话框，如下图所示，选择“是”即可从工作区中移除 PD 设计文件。



3.1.3. 展开 PD 设计文件

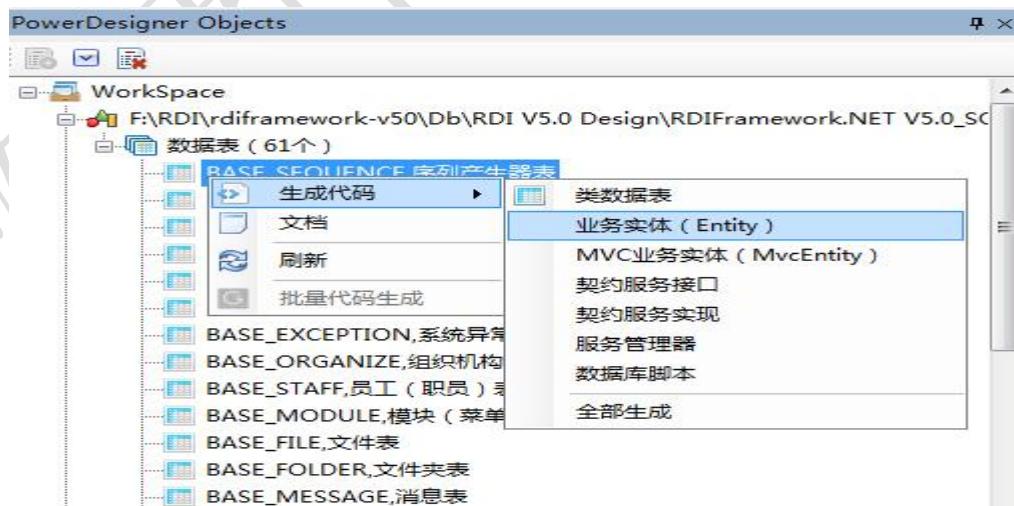
对于已经添加的 PD 设计文件，我们可以展开（如果是初次，就是加载当前设计文件的所有表）它，对应图标为：。

3.2. 代码生成

下面我们对如何通过“PD 设计文件”进行代码生成进行讲解，主要包括以下内容：

- 1) 生成类数据表。
- 2) 生成业务实体（Entity）。
- 3) 生成契约服务接口。
- 4) 生成契约服务。
- 5) 生成服务管理器。
- 6) 全部生成。
- 7) 生成表设计文档。

具体的代码说明与根据数据库生成的代码是一致的，具体概念就不再说明了，生成的代码公共部分的设计同样依赖于“项目属性设置”，具体可参考 2.3.6.1 “项目属性设置”一节。



3.3.1. 生成类数据表

The screenshot shows the RDIFramework.NET Platform Code Generator interface. On the left, the 'Database View' pane lists various database objects like BASE_PERMISSION, BASE_SCHEDULE, etc. In the center, the code editor displays the generated C# code for the `CaseProductinfoTable` class. The code includes XML documentation for properties such as `Productcode`, `Productname`, and `Productmodel`. The right side shows the 'PowerDesigner Objects' palette with a list of 76 database objects.

```

using System;
using System.Collections.Generic;
using System.Linq;
namespace RDIFramework.BizLogic
{
    /// <summary>
    /// CaseProductinfoTable
    /// 事例_产品信息
    /// 修改纪录
    /// 2023-07-13 版本: 5.1 RDIFramework.NET 创建主键。
    /// 版本: 6.0
    /// <author>
    /// <name>RDIFramework.NET</name>
    /// <date>2023-07-13</date>
    /// </author>
    /// </summary>
    public partial class CaseProductinfoTable
    {
        /// <summary>
        /// 事例_产品信息
        /// </summary>
        [NonSerialized]
        public static string TableName = "CASE_PRODUCTINFO";
        /// <summary>
        /// 主键
        /// </summary>
        [NonSerialized]
        public static string FieldID = "ID";
        ...
    }
}

```

3.3.2. 生成业务实体 (Entity)

The screenshot shows the RDIFramework.NET Platform Code Generator interface. The central code editor displays the generated C# code for the `CaseProductinfoEntity` class, which inherits from `BaseEntity`. It includes validation attributes like `StringLength` and `Required` for properties `Productcode`, `Productname`, and `Productmodel`. The right side shows the 'PowerDesigner Objects' palette and a context menu for generating code.

```

[Serializable]
public partial class CaseProductinfoEntity : BaseEntity
{
    /// <summary>
    /// 主键
    /// </summary>
    [StringLength(40, ErrorMessage = "主键不能超过40个字符")]
    [Required(ErrorMessage = "需要输入主键")]
    [DataMember]
    public string Id { get; set; }

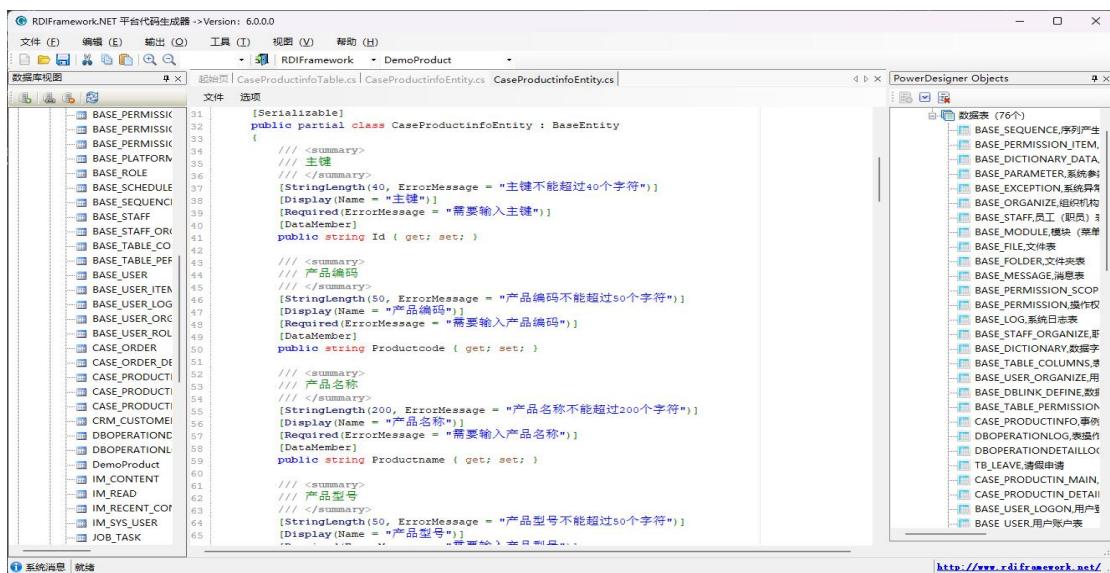
    /// <summary>
    /// 产品编码
    /// </summary>
    [StringLength(50, ErrorMessage = "产品编码不能超过50个字符")]
    [Required(ErrorMessage = "需要输入产品编码")]
    [DataMember]
    public string Productcode { get; set; }

    /// <summary>
    /// 产品名称
    /// </summary>
    [StringLength(200, ErrorMessage = "产品名称不能超过200个字符")]
    [Required(ErrorMessage = "需要输入产品名称")]
    [DataMember]
    public string Productname { get; set; }

    /// <summary>
    /// 产品型号
    /// </summary>
    [StringLength(50, ErrorMessage = "产品型号不能超过50个字符")]
    [Required(ErrorMessage = "需要输入产品型号")]
    [DataMember]
    public string Productmodel { get; set; }
}

```

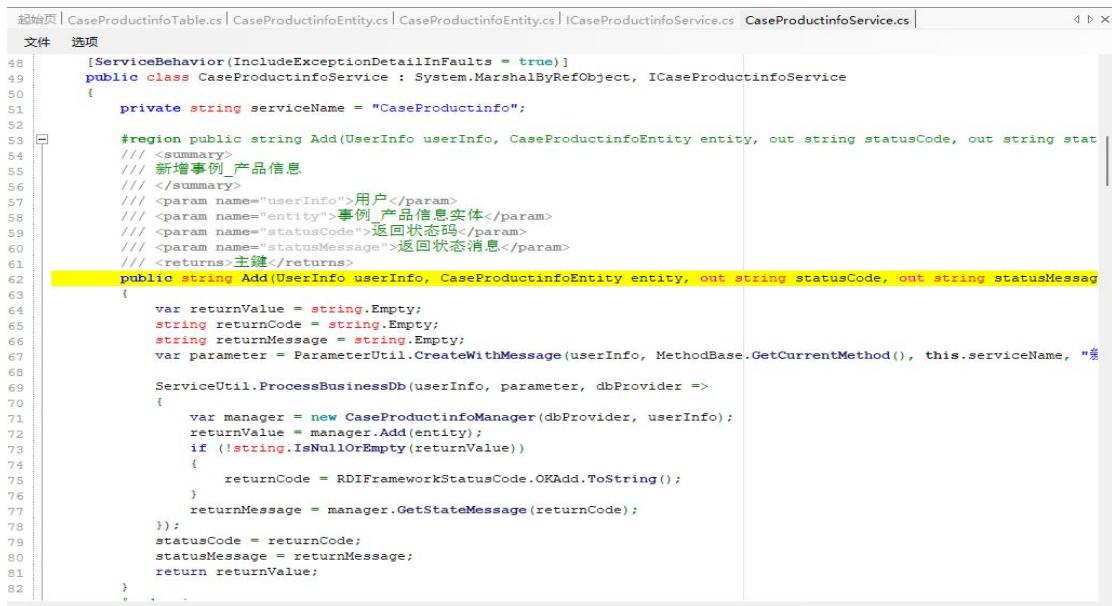
3.3.3. 生成 MVC 业务实体 (MVCEntity)



3.3.4. 生成契约服务接口



3.3.5. 生成契约服务

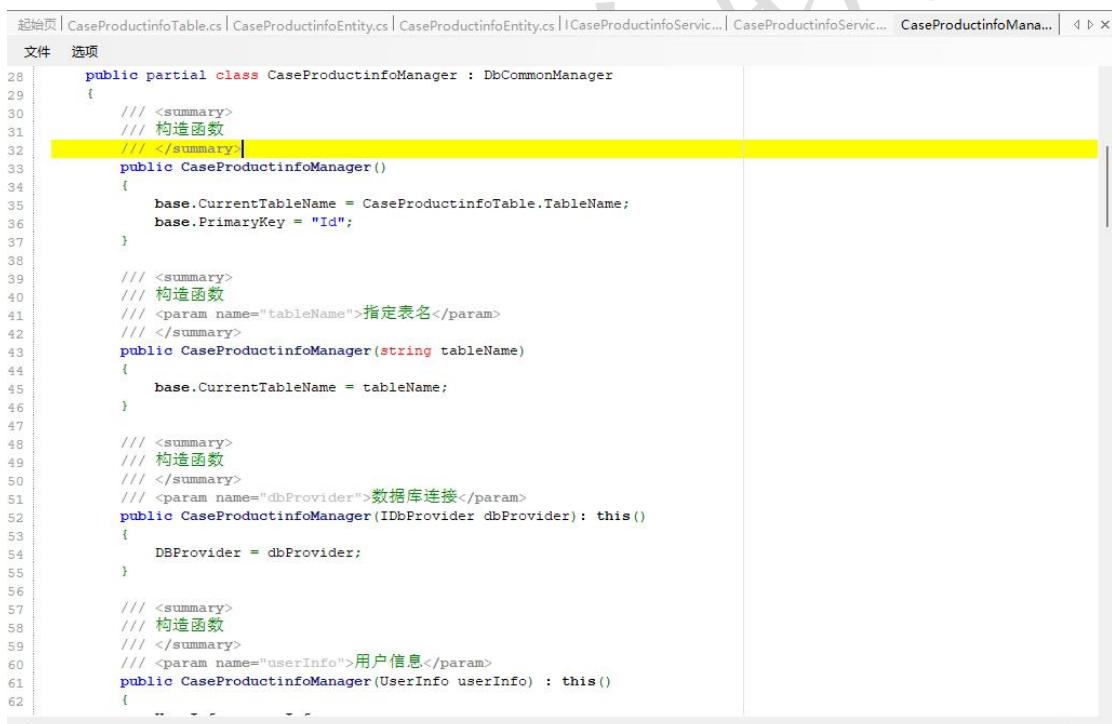


```

起始页 | CaseProductinfoTable.cs | CaseProductinfoEntity.cs | CaseProductinfoEntity.cs | ICaseProductinfoService.cs | CaseProductinfoService.cs | 4 ↻ ×
文件 选项
48 [ServiceBehavior(IncludeExceptionDetailInFaults = true)]
49 public class CaseProductinfoService : System.MarshalByRefObject, ICaseProductinfoService
50 {
51     private string serviceName = "CaseProductinfo";
52
53     #region public string Add(UserInfo userInfo, CaseProductinfoEntity entity, out string statusCode, out string stat
54     /// <summary>
55     /// 新增事例_产品信息
56     /// </summary>
57     /// <param name="userInfo">用户</param>
58     /// <param name="entity">事例_产品信息实体</param>
59     /// <param name="statusCode">返回状态码</param>
60     /// <param name="statusMessage">返回状态消息</param>
61     /// <returns>主键</returns>
62     public string Add(UserInfo userInfo, CaseProductinfoEntity entity, out string statusCode, out string statusMessag
63     {
64         var returnValue = string.Empty;
65         string returnCode = string.Empty;
66         string returnMessage = string.Empty;
67         var parameter = ParameterUtil.CreateWithMessage(userInfo, MethodBase.GetCurrentMethod(), this.serviceName, "新增事例_产品信息");
68
69         ServiceUtil.ProcessBusinessDb(userInfo, parameter, dbProvider =>
70         {
71             var manager = new CaseProductinfoManager(dbProvider, userInfo);
72             returnValue = manager.Add(entity);
73             if (!string.IsNullOrEmpty(returnValue))
74             {
75                 returnCode = RDIFrameworkStatusCode.OKAdd.ToString();
76             }
77             returnMessage = manager.GetStateMessage(returnCode);
78         });
79         statusCode = returnCode;
80         statusMessage = returnMessage;
81         return returnValue;
82     }
}

```

3.3.6. 生成服务管理器



```

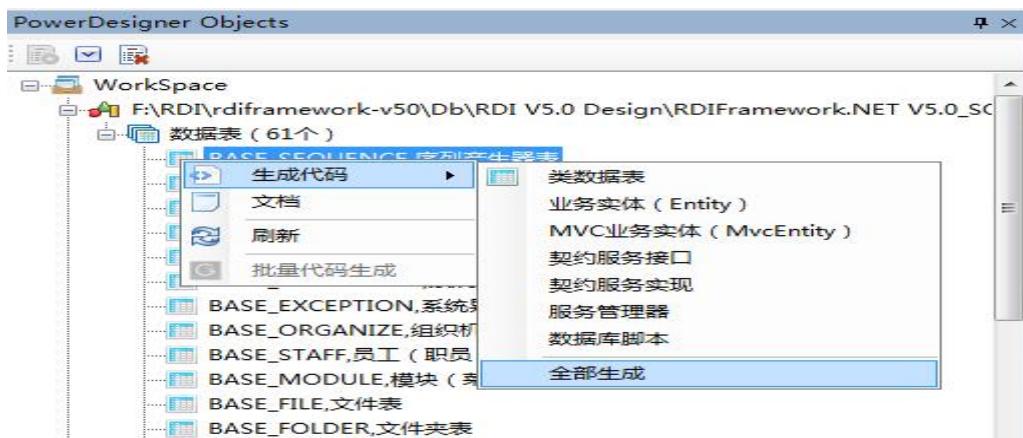
起始页 | CaseProductinfoTable.cs | CaseProductinfoEntity.cs | CaseProductinfoEntity.cs | ICaseProductinfoService.cs | CaseProductinfoService.cs | CaseProductinfoManager.cs | 4 ↻ ×
文件 选项
28 public partial class CaseProductinfoManager : DbCommonManager
29 {
30     /// <summary>
31     /// 构造函数
32     /// </summary>
33     public CaseProductinfoManager()
34     {
35         base.CurrentTableName = CaseProductinfoTable.TableName;
36         base.PrimaryKey = "Id";
37     }
38
39     /// <summary>
40     /// 构造函数
41     /// <param name="tableName">指定表名</param>
42     /// </summary>
43     public CaseProductinfoManager(string tableName)
44     {
45         base.CurrentTableName = tableName;
46     }
47
48     /// <summary>
49     /// 构造函数
50     /// <param name="dbProvider">数据库连接</param>
51     public CaseProductinfoManager(IDbProvider dbProvider) : this()
52     {
53         DBProvider = dbProvider;
54     }
55
56     /// <summary>
57     /// 构造函数
58     /// <param name="userInfo">用户信息</param>
59     public CaseProductinfoManager(UserInfo userInfo) : this()
60     {
61     }
}

```

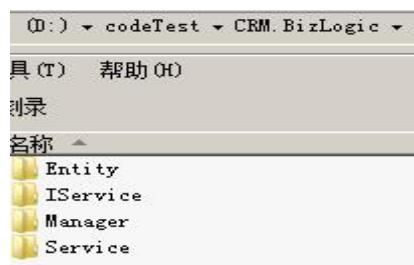
3.3.7. 生成全部代码

生成当前代码就是把当前所选设计表的全部代码（类数据表、业务实体、服务接口、服务实现、服务管理器）生成到本地目录中（保存的位置可通过“项目属性设置”窗口的“代码输出目录”进行设置）。

单击“全部生成”，如下图所示。



即可把当前表的所有代码全部生成到指定目录中，生成的代码如下：



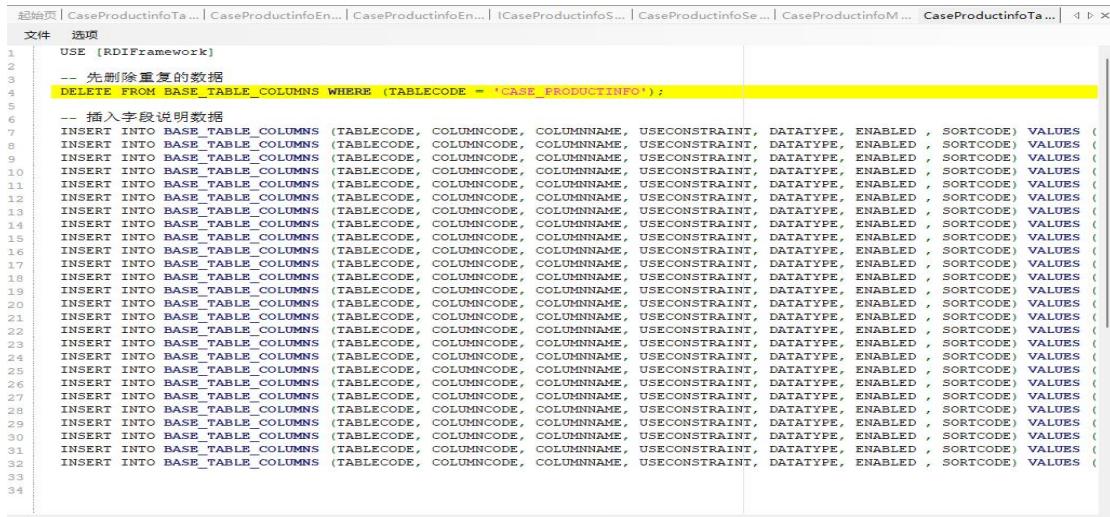
3.3.8. 生成表设计文档

表设计文档，就是当前所选表的数据库设计文档，如下图所示。

CUSTOMER (客户信息)										
序号	列代码	列名称	数据类型	长度	小数位	标识	主键	允许空	默认值	描述
1	ID	主键	Int			Y	Y	N		主键
2	CODE	客户编号	Nvarchar(50)	50		N	N	N		客户编号
3	FULLNAME	客户名称	Nvarchar(200)	200		N	N	Y		客户全称
4	POSTCODE	邮政编码	Nvarchar(6)	6		N	N	Y		
5	COUNTRY	国家	Nvarchar(50)	50		N	N	Y		
6	PROVINCE	省份	Nvarchar(50)	50		N	N	Y		
7	CITY	城市	Nvarchar(50)	50		N	N	Y		
8	REGISTERDATE	登记日期	date			N	N	Y		
9	NEXTCONTACTDATE	下次联系时间	date			N	N	Y		
10	LASTCONTACTDATE	最后联系时间	date			N	N	Y		
11	ADDRESS	公司地址	Nvarchar(500)	500		N	N	Y		
12	ONLINEADDRESS	公司主页	Nvarchar(500)	500		N	N	Y		
13	TURNOVER	营业额	bigint			N	N	Y		营业额（单位：人民币元）。
14	LEVEL	客户等级	nvarchar(50)	50		N	N	Y		客户等级
15	STATUS	客户状态	nvarchar(50)	50		N	N	Y		
16	TYPE	客户类型	nvarchar(50)	50		N	N	N	3	信用度（1至5），默认为3。
17	SOURCE	客户来源	nvarchar(50)	50		N	N	Y		
18	TRADE	行业类型	nvarchar(50)	50		N	N	Y		
19	AREA	区域	nvarchar(50)	50		N	N	Y		
20	HONOR	信用等级	NVarchar(50)	50		N	N	Y		公司邮编
21	REMARK	备注	Nvarchar(50)	50		N	N	Y		公司电话

3.3.9. 生成数据库脚本

数据库脚本主要用于权限控制表中使用，我们在以通过代码生成器生成待控制的权限控制表脚本数据，如下图所示。



```

1 USE [RDIFramework]
2
3 -- 先删除重复的数据
4 DELETE FROM BASE_TABLE_COLUMNS WHERE (TABLECODE = 'CASE_PRODUCTINFO');
5
6 -- 插入字段说明数据
7 INSERT INTO BASE_TABLE_COLUMNS (TABLECODE, COLUMNCODE, COLUMNNAME, USECONSTRAINT, DATATYPE, ENABLED, SORTCODE) VALUES (
8     'CASE_PRODUCTINFO', 'C001', 'CASEID', 1, 'INT', 1, 1
9 ), ('CASE_PRODUCTINFO', 'C002', 'PRODUCTID', 1, 'INT', 1, 2
10 ), ('CASE_PRODUCTINFO', 'C003', 'NAME', 1, 'NVARCHAR(50)', 1, 3
11 ), ('CASE_PRODUCTINFO', 'C004', 'DESCRIPTION', 1, 'NVARCHAR(200)', 1, 4
12 ), ('CASE_PRODUCTINFO', 'C005', 'PRICE', 1, 'DECIMAL(10,2)', 1, 5
13 ), ('CASE_PRODUCTINFO', 'C006', 'STOCK', 1, 'INT', 1, 6
14 ), ('CASE_PRODUCTINFO', 'C007', 'CREATE_DATE', 1, 'DATETIME', 1, 7
15 ), ('CASE_PRODUCTINFO', 'C008', 'UPDATE_DATE', 1, 'DATETIME', 1, 8
16 ), ('CASE_PRODUCTINFO', 'C009', 'DELETE_DATE', 1, 'DATETIME', 1, 9
17 ), ('CASE_PRODUCTINFO', 'C010', 'IS_DELETED', 1, 'BIT', 1, 10
18 ), ('CASE_PRODUCTINFO', 'C011', 'CREATE_BY', 1, 'NVARCHAR(50)', 1, 11
19 ), ('CASE_PRODUCTINFO', 'C012', 'UPDATE_BY', 1, 'NVARCHAR(50)', 1, 12
20 ), ('CASE_PRODUCTINFO', 'C013', 'DELETE_BY', 1, 'NVARCHAR(50)', 1, 13
21 ), ('CASE_PRODUCTINFO', 'C014', 'VERSION', 1, 'INT', 1, 14
22 ), ('CASE_PRODUCTINFO', 'C015', 'CATEGORY_ID', 1, 'INT', 1, 15
23 ), ('CASE_PRODUCTINFO', 'C016', 'SUBCATEGORY_ID', 1, 'INT', 1, 16
24 ), ('CASE_PRODUCTINFO', 'C017', 'MANUFACTURER_ID', 1, 'INT', 1, 17
25 ), ('CASE_PRODUCTINFO', 'C018', 'SUPPLIER_ID', 1, 'INT', 1, 18
26 ), ('CASE_PRODUCTINFO', 'C019', 'UNIT_ID', 1, 'INT', 1, 19
27 ), ('CASE_PRODUCTINFO', 'C020', 'UNIT_OF_MEASURE_ID', 1, 'INT', 1, 20
28 ), ('CASE_PRODUCTINFO', 'C021', 'WEIGHT', 1, 'DECIMAL(10,2)', 1, 21
29 ), ('CASE_PRODUCTINFO', 'C022', 'HEIGHT', 1, 'DECIMAL(10,2)', 1, 22
30 ), ('CASE_PRODUCTINFO', 'C023', 'WIDTH', 1, 'DECIMAL(10,2)', 1, 23
31 ), ('CASE_PRODUCTINFO', 'C024', 'DEPTH', 1, 'DECIMAL(10,2)', 1, 24
32 ), ('CASE_PRODUCTINFO', 'C025', 'QUANTITY', 1, 'DECIMAL(10,2)', 1, 25
33 )

```

4. 代码批量生成

4.1. 基于数据库的代码批量生成

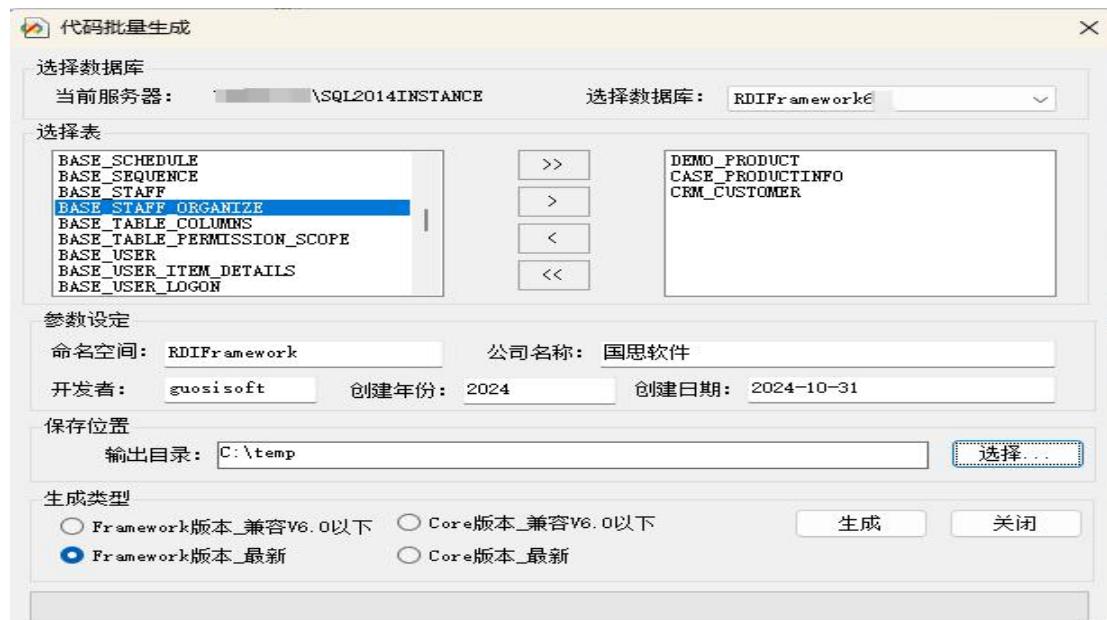
基于数据库的代码批量生成，是快速生成项目业务逻辑的最佳方式，对生成的代码只需要稍加修改，即可完成业务逻辑部分，大大提高开发的效率，节省开发时间与开发成本。基于数据库的代码批量生成，可以选择指定数据库下特定的表进行业务逻辑代码的生成。

要基于数据库进行批量代码生成，可以利用 RDIFramework.Net 代码生成器“起始页”页面上的常用操作中的“代码批量生成器”功能按钮来完成，如下图所示。



选择“代码批量生成器”，打开“代码批量生成”窗口，如下图所示。

Tips:如果提示“没有可用的数据库连接，请先连接数据库服务器”，则可在数据库视图导航区域，选择一个数据库服务器连接，再重新链接即可。

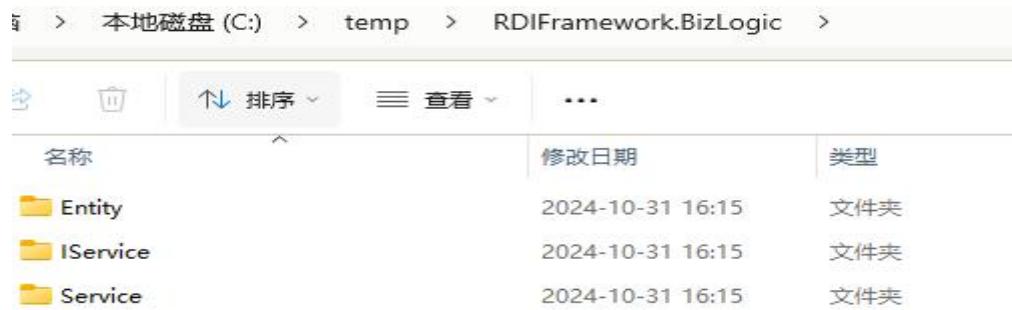


在上图中，显示了当前所选的服务器连接，针对所选“服务器连接”，在数据库列表框中列出了当前连接下的可用数据库，我们可以选择我们需要生成代码的数据库，会在选择表区域列出当前所选数据表的所有数据表，我们可以选择需要生成业务代码的数据表至右侧列表中，再设置项目的相关参数，如：命名空间、公司名称、开发者、创建年份、创建日期等，这些参数已经通过“项目属性”的设置进行了默认选择，当然你也可以在这进行修改。在保存位置设置区域，“输出目录”就是设置代码批量生成后保存的文件目录，生成类型可以选择生成支持.NET Framework 或.NET Core 版本的代码。

这一切都设置就绪后，单击“导出”按钮，即可对当前数据库所选表批量生成代码，如下图所示。



批量生成成功后，我们可以打开代码保存的目录“C:\Temp”，查看生成的代码，如下图所示。



可以看到，我们的代码批量生成成功了。

4.2. 数据库设计文档生成

“数据库设计文档生成”是快速生成当前数据表所选数据表的数据库设计文档，一键生成，相当方便，对项目文档中自动生成数据库设计文档相当有帮助。在 RDI Framework.NET 代码生成器中，可以通过起始页的常用操作中的“数据库文档生成器”来生成数据库的设计文档，如下图所示。



单击“数据库文档生成器”功能按钮，打开“生成数据库设计文档”窗口，如下图所示。



在上图中，我们可以选择当前所选连接服务器下的数据库列表，选择一个数据库后，在选择表区域的左侧列出来当前数据库的所有可用表，我们可以选择需要生成的数据表至右侧，也可以选择全部数据表。选择好待生成的数据表后，我们还需要设置数据库设计文档输

出的格式，现在支持两种格式的生成，一种是生成 Word 文档格式，一种是生成 HTML 格式（支持三种风格）。

1) 生成 Word 格式

选择生成 Word 格式，需要本机安装 Office 软件方可，单击“生成”按钮，等待片刻（会有滚动条提示），即可完成 Word 格式的数据库设计文档的生成，生成的效果如下图所示。



The screenshot shows a Microsoft Word document titled "CRM1.3" with the subtitle "表名：CUSTOMER". Below the subtitle is a table with 16 columns, each representing a field in the CUSTOMER table. The columns are labeled: 序号 (Index), 列名 (Name), 数据类型 (Type), 长度 (Length), 小数位 (Decimal Places), 标识 (Identity), 主键 (Primary Key), 允许空 (Allow Null), 默认值 (Default Value), and 说明 (Description). The table includes rows for ID, CODE, FULLNAME, POSTCODE, COUNTRY, PROVINCE, CITY, REGISTERDATE, NEXTCONTACTDATE, LASTCONTACTDATE, ADDRESS, ONLINEADDRESS, TURNOVER, LEVEL, STATUS, and TYPE.

序号	列名	数据类型	长度	小数位	标识	主键	允许空	默认值	说明
1	ID	int	4	0	是	是	否		
2	CODE	nvarchar	50	0			否		
3	FULLNAME	nvarchar	200	0			是		
4	POSTCODE	nvarchar	6	0			是		
5	COUNTRY	nvarchar	50	0			是		
6	PROVINCE	nvarchar	50	0			是		
7	CITY	nvarchar	50	0			是		
8	REGISTERDATE	date	3	0			是		
9	NEXTCONTACTDATE	date	3	0			是		
10	LASTCONTACTDATE	date	3	0			是		
11	ADDRESS	nvarchar	500	0			是		
12	ONLINEADDRESS	nvarchar	500	0			是		
13	TURNOVER	bigint	8	0			是		
14	LEVEL	nvarchar	50	0			是		
15	STATUS	nvarchar	50	0			是		
16	TYPE	nvarchar	50	0			否	('3')	
17	COMMENT								

2) 生成 HTML 格式

同样，数据库设计文档还可以输出为 HTML 格式，HTML 格式提供了三种风格，如下图所示，你可以选择一种适合你的风格进行生成。



风格一生成效果图如下：



The screenshot shows a Microsoft Word document titled "CRM1.3" with the subtitle "表名：CUSTOMER". Below the subtitle is a table with 16 columns, each representing a field in the CUSTOMER table. The columns are labeled: 序号 (Index), 列名 (Name), 数据类型 (Type), 长度 (Length), 小数位 (Decimal Places), 标识 (Identity), 主键 (Primary Key), 允许空 (Allow Null), 默认值 (Default Value), and 说明 (Description). The table includes rows for ID, CODE, FULLNAME, POSTCODE, COUNTRY, PROVINCE, CITY, REGISTERDATE, NEXTCONTACTDATE, LASTCONTACTDATE, ADDRESS, ONLINEADDRESS, TURNOVER, LEVEL, STATUS, and TYPE.

序号	列名	数据类型	长度	小数位	标识	主键	允许空	默认值	说明
1	ID	int	4	0	是	是	否		
2	CODE	nvarchar	50	0			否		
3	FULLNAME	nvarchar	200	0			是		
4	POSTCODE	nvarchar	6	0			是		
5	COUNTRY	nvarchar	50	0			是		
6	PROVINCE	nvarchar	50	0			是		
7	CITY	nvarchar	50	0			是		
8	REGISTERDATE	date	3	0			是		
9	NEXTCONTACTDATE	date	3	0			是		
10	LASTCONTACTDATE	date	3	0			是		
11	ADDRESS	nvarchar	500	0			是		
12	ONLINEADDRESS	nvarchar	500	0			是		
13	TURNOVER	bigint	8	0			是		
14	LEVEL	nvarchar	50	0			是		
15	STATUS	nvarchar	50	0			是		
16	TYPE	nvarchar	50	0			否	('3')	
17	COMMENT								

风格二生成效果图如下：

数据库名: CRM1.3 表名: CUSTOMER									
序号	列名	数据类型	长度	小数位	标识	主键	允许空	默认值	说明
1	ID	int	4	0	是	是	否		
2	CODE	nvarchar	50	0			否		
3	FULLNAME	nvarchar	200	0			是		
4	POSTCODE	nvarchar	6	0			是		
5	COUNTRY	nvarchar	50	0			是		
6	PROVINCE	nvarchar	50	0			是		
7	CITY	nvarchar	50	0			是		
8	REGISTERDATE	date	3	0			是		
9	NEXTCONTACTDATE	date	3	0			是		
10	LASTCONTACTDATE	date	3	0			是		
11	ADDRESS	nvarchar	500	0			是		
12	ONLINEADDRESS	nvarchar	500	0			是		
13	TURNOVER	bigint	8	0			是		
14	LEVEL	nvarchar	50	0			是		
15	STATUS	nvarchar	50	0			是		
16	TYPE	nvarchar	50	0			否	(3')	
17	SOURCE	nvarchar	50	0			是		

风格三生成效果图如下：

数据库名: CRM1.3 表名: CUSTOMER									
序号	列名	数据类型	长度	小数位	标识	主键	允许空	默认值	说明
1	ID	int	4	0	是	是	否		
2	CODE	nvarchar	50	0			否		
3	FULLNAME	nvarchar	200	0			是		
4	POSTCODE	nvarchar	6	0			是		
5	COUNTRY	nvarchar	50	0			是		
6	PROVINCE	nvarchar	50	0			是		
7	CITY	nvarchar	50	0			是		
8	REGISTERDATE	date	3	0			是		
9	NEXTCONTACTDATE	date	3	0			是		
10	LASTCONTACTDATE	date	3	0			是		
11	ADDRESS	nvarchar	500	0			是		
12	ONLINEADDRESS	nvarchar	500	0			是		
13	TURNOVER	bigint	8	0			是		
14	LEVEL	nvarchar	50	0			是		
15	STATUS	nvarchar	50	0			是		
16	TYPE	nvarchar	50	0			否	(3')	
17	SOURCE	nvarchar	50	0			是		

4.3. 基于 PowerDesigner 设计文件的代码批量生成

基于数据库的代码批量生成完全依赖于数据库，有一定的局限性，现在我们看看如何使用 PD 设计文档来进行代码的批量生成。PD 设计文档不限数据库类型，任意数据库类库都可进行代码生成，相当的方便。要利用 PD 设计文档进行代码生成，需要在 PowerDesigner Objects 导航区选择“数据表”树节点后单击右键，在弹出的快捷菜单中选择“批量代码生成”，如下图所示。



通过此方法即可对当前 PD 设计文件的所有表进行批量业务代码的生成，生成的业务代码与 4.1 节介绍的代码完全一致，在此就不再进行展示。需要说明的说，在批量代码生成前请先到“项目属性”设置窗口，设置代码相关信息（如：版权信息、代码的命名空间，代码的作者信息、代码的年代、日期等），“批量代码生成”依赖于这些设置。

RDIFramework.NET，基于全新.NET Framework 与.NET Core 的快速信息化系统敏捷开发、整合框架，给用户和开发者最佳的.Net 框架部署方案。为企业快速构建跨平台、企业级的应用提供了强大支持。

框架官方：<http://www.rdiframework.net/>
<http://www.guosisoft.com/>

框架其他博客：

<http://www.cnblogs.com/huyong/>
<https://yonghu.blog.csdn.net/>

联系方式：

邮件：406590790@qq.com

Q Q：406590790

电话：13005007127 (同微信)

RDIFramework.NET 敏捷开发框架由海南国思软件科技有限公司专业团队长期打造、一直在更新、一直在升级，请放心使用！

欢迎关注 RDIFramework.NET 敏捷开发框架官方公众微信（微信号：guosisoft），及时了解最新动态。

扫描二维码立即关注

微信咨询号



微信公众号

