

数据仓库实验报告

|  |  |
| --- | --- |
| **实验名称：** | **实验六 报表设计器的使用** |
| **实验日期：** | **2025.11.27** |
| **实验地点：** | **西片4号楼101** |
| **提交日期：** |  |
|  | |
| **学号：** | **36720232204041** |
| **姓名：** | **苏一涵** |
| **专业年级：** | **软工23级** |
| **学年学期：** | **2025-2026学年第一学期** |

1. 实验环境

SQL Server 2019

Visual Studio/SQL Server Data Tools(SSDT)；

SQL Server Reporting Services (SSRS)

1. 实验目的

（1） 理解报表服务器、报表和相关项的概念和功能

（2） 理解并掌握报表设计器的使用方法

1. 实验内容和步骤

**SSRS 的概念和作用**

SSRS（SQL Server Reporting Services）是微软推出的企业级报表服务，属于 SQL Server 套件的核心组件，也是商业智能（BI）生态中的重要工具，核心作用是整合多源数据、生成标准化报表并实现灵活分发与访问。

**核心概念**

本质：基于服务器的报表解决方案，支持从设计、部署到管理的全生命周期报表管理。

核心定位：连接数据与业务决策，将分散在数据库、Excel、CSV 等数据源中的数据，转化为结构化、可视化的报表。

**核心作用**

（1）数据整合与统一报表：支持对接 SQL Server、Oracle、MySQL 等多种数据源，消除数据孤岛，生成格式统一的报表（如财务报表、业务 KPI 报表）。

（2）可视化与交互能力：提供表格、柱状图、折线图、仪表板等多种可视化形式，支持钻取（点击明细查看详情）、筛选、排序等交互操作。

（3）灵活分发与访问：报表可通过网页直接访问、邮件订阅推送、PDF/Excel 导出下载，也可嵌入到应用系统中，满足不同场景的访问需求。

（4）权限管控与生命周期管理：支持精细化权限设置（控制谁能查看 / 编辑报表），同时提供报表版本管理、运行状态监控等运维功能。

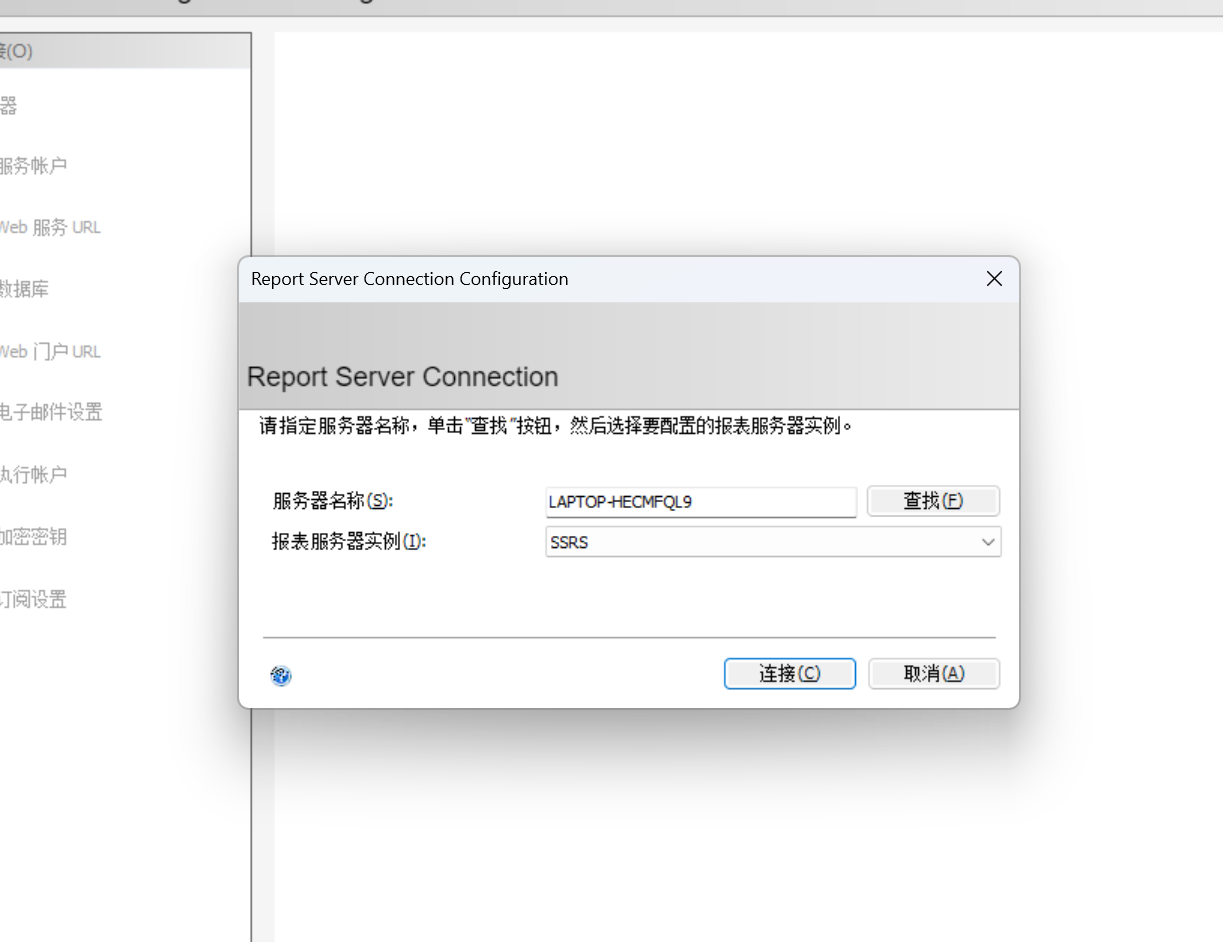
**典型应用场景**

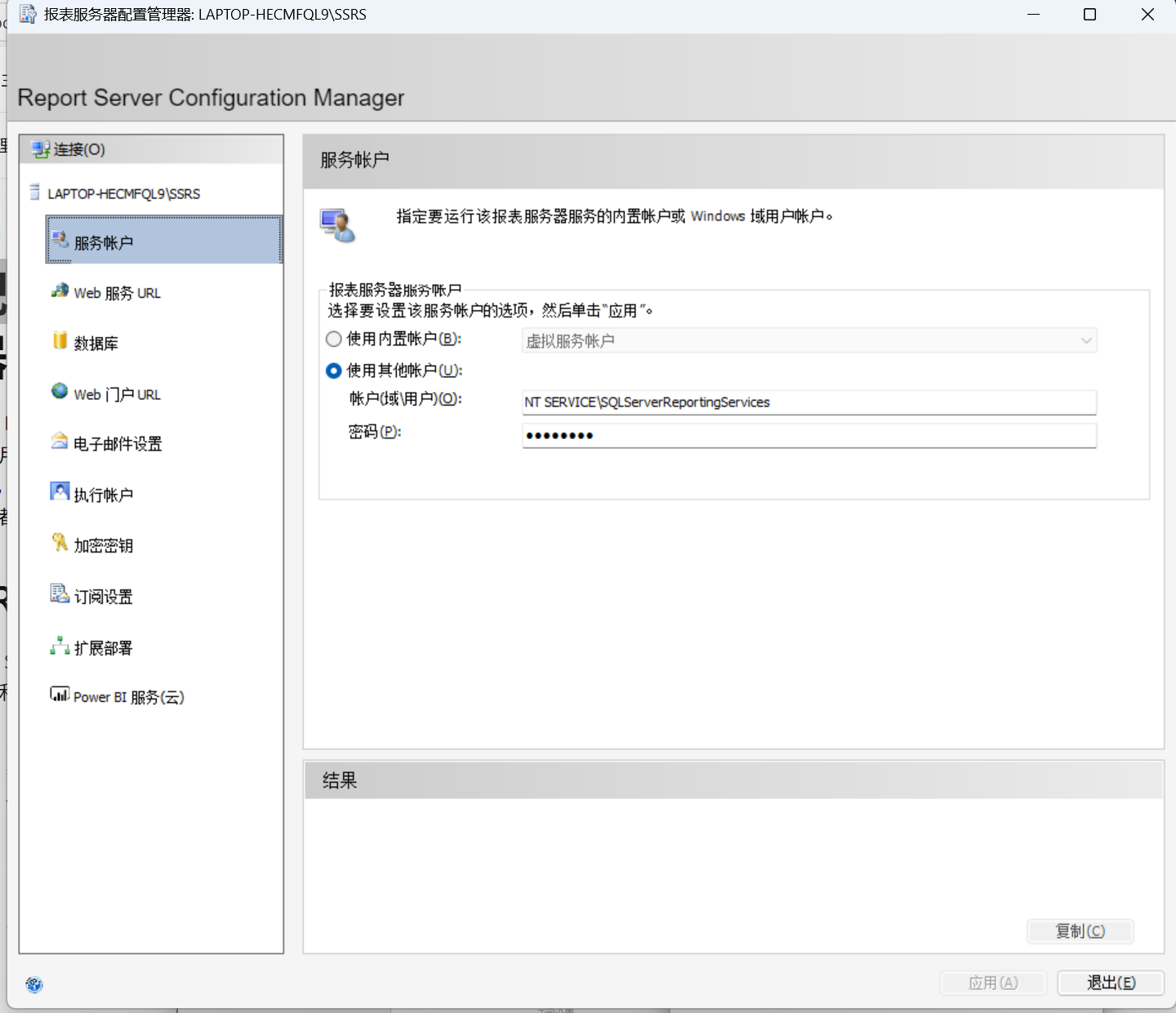
（1）企业日常运营报表：如销售业绩统计、库存盘点报表、员工考勤报表。

（2）数据监控与决策支持：如实时业务 KPI 仪表板、财务预算执行分析报表。

（3）合规与监管报表：如税务申报报表、行业监管要求的标准化报表。

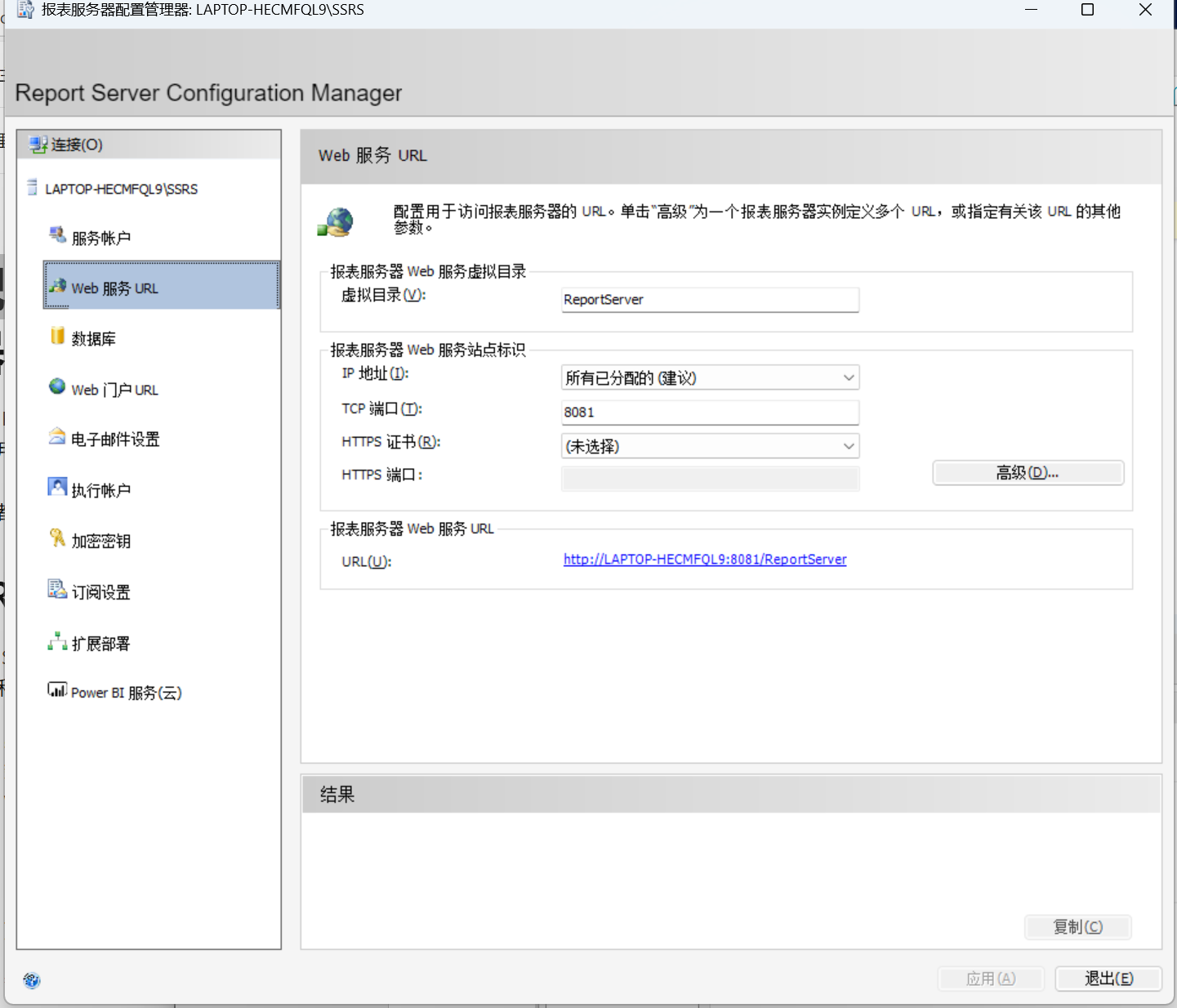
**配置报表服务器本机模式**



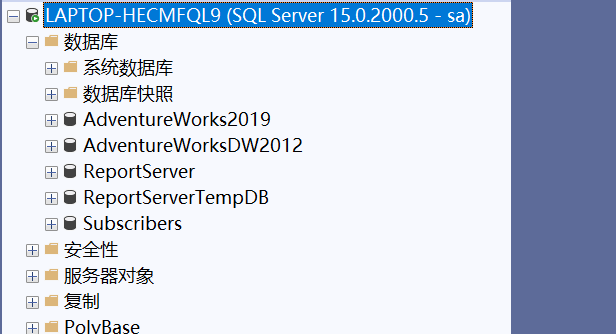


**配置报表服务器 URL**

我这边设置8081为开放端口

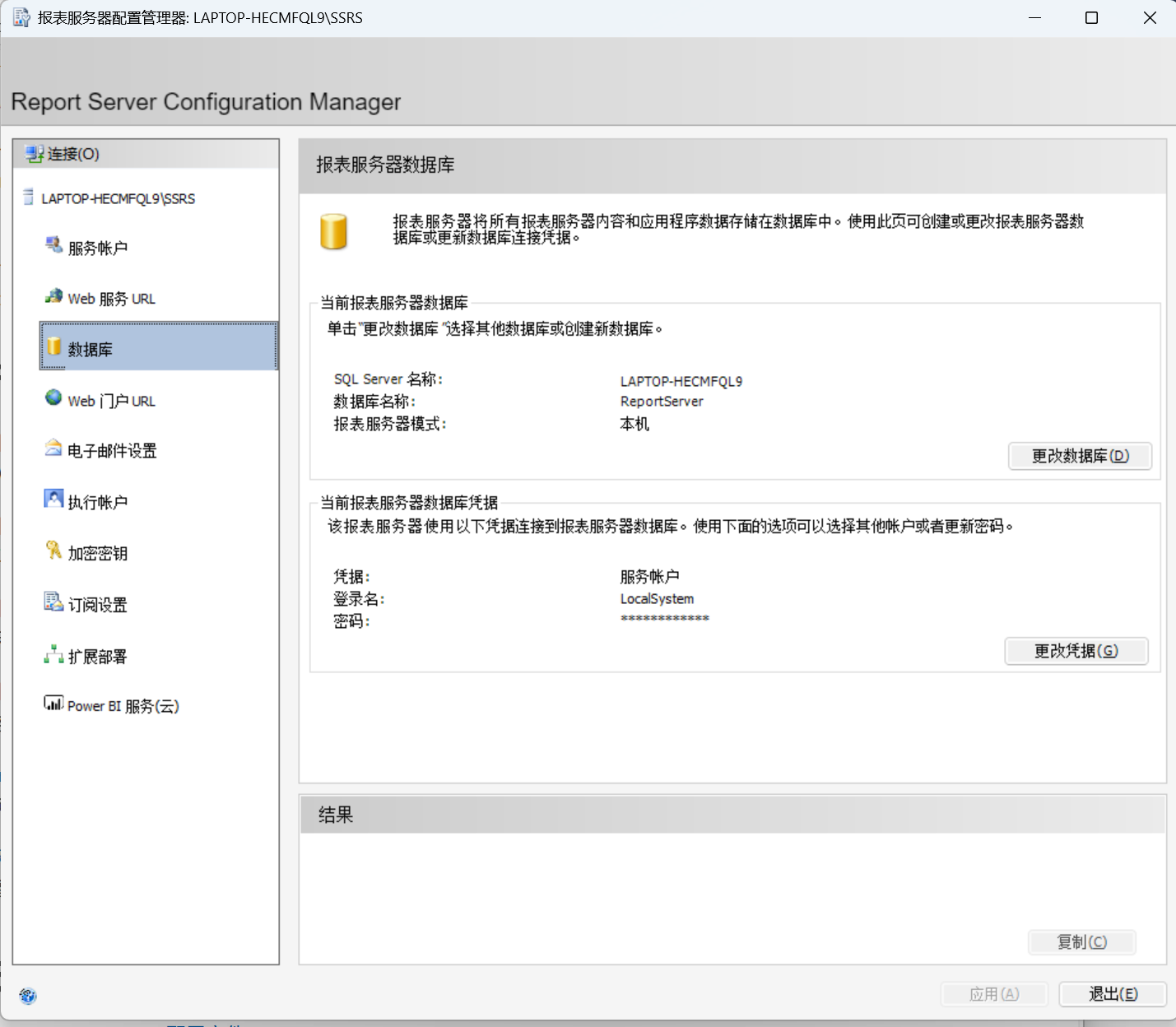


**创建报表服务器数据库，报表服务器配置管理器**

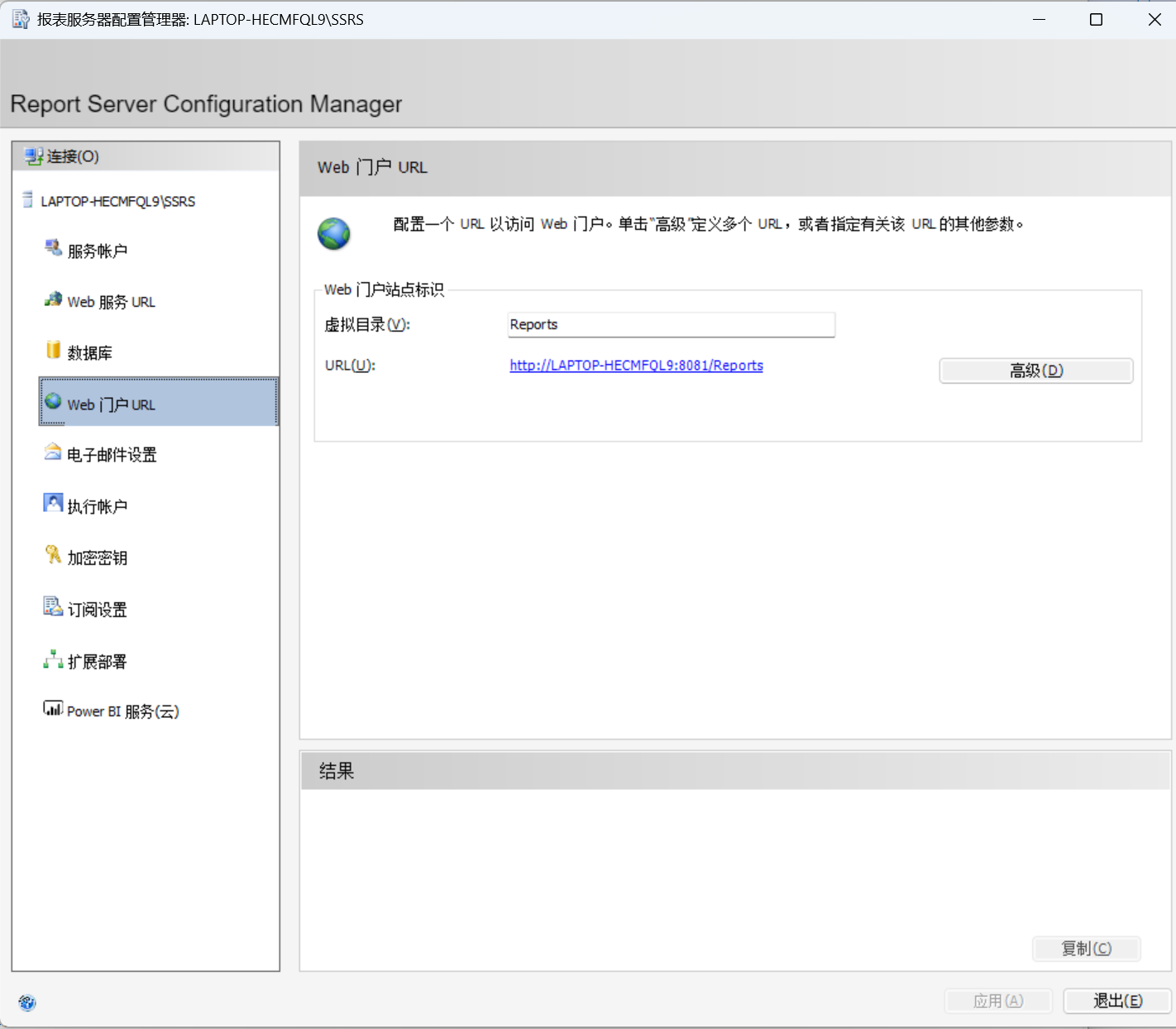


**连接上去**

**配置报表服务器数据库连接**



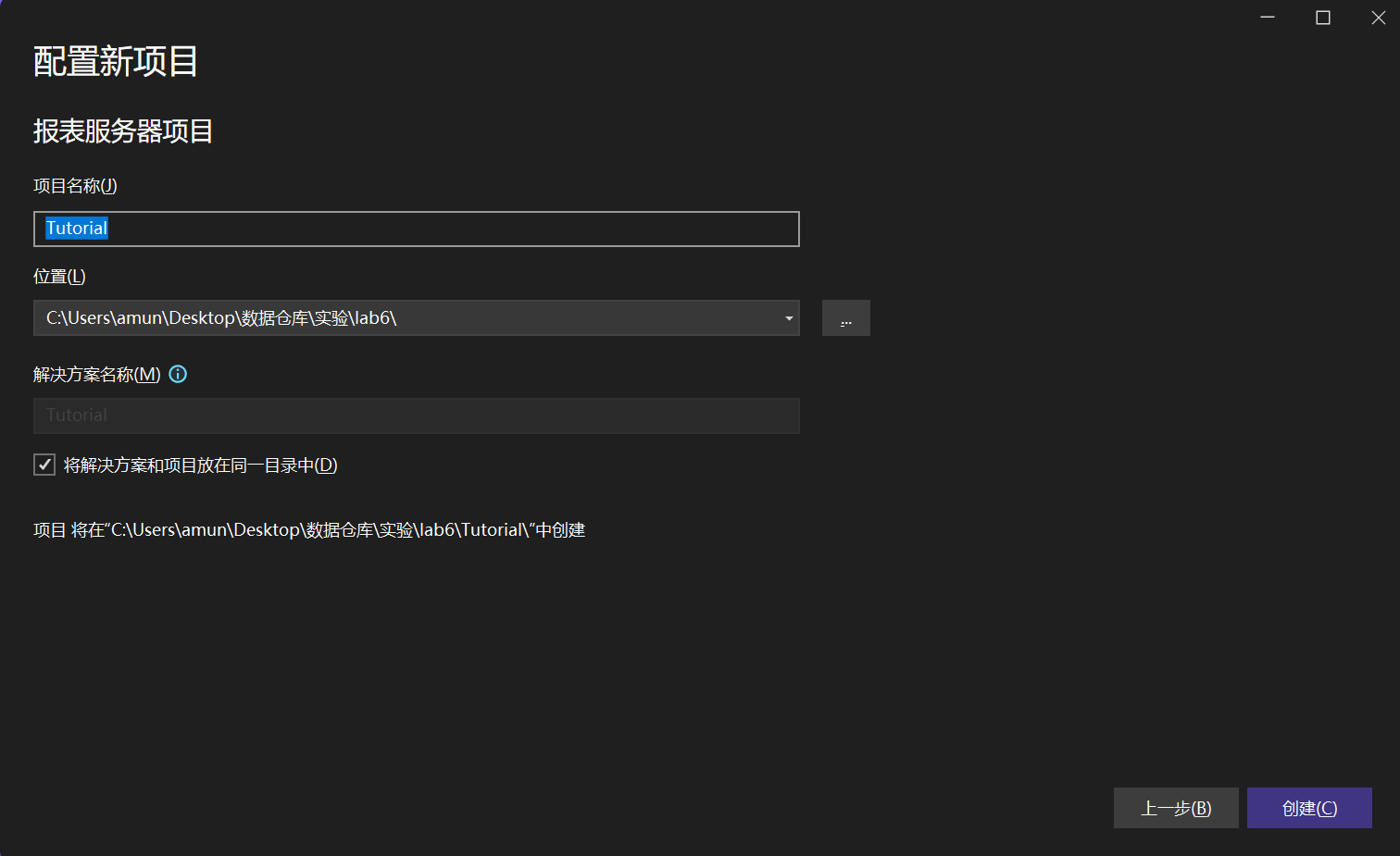
Web门户配置



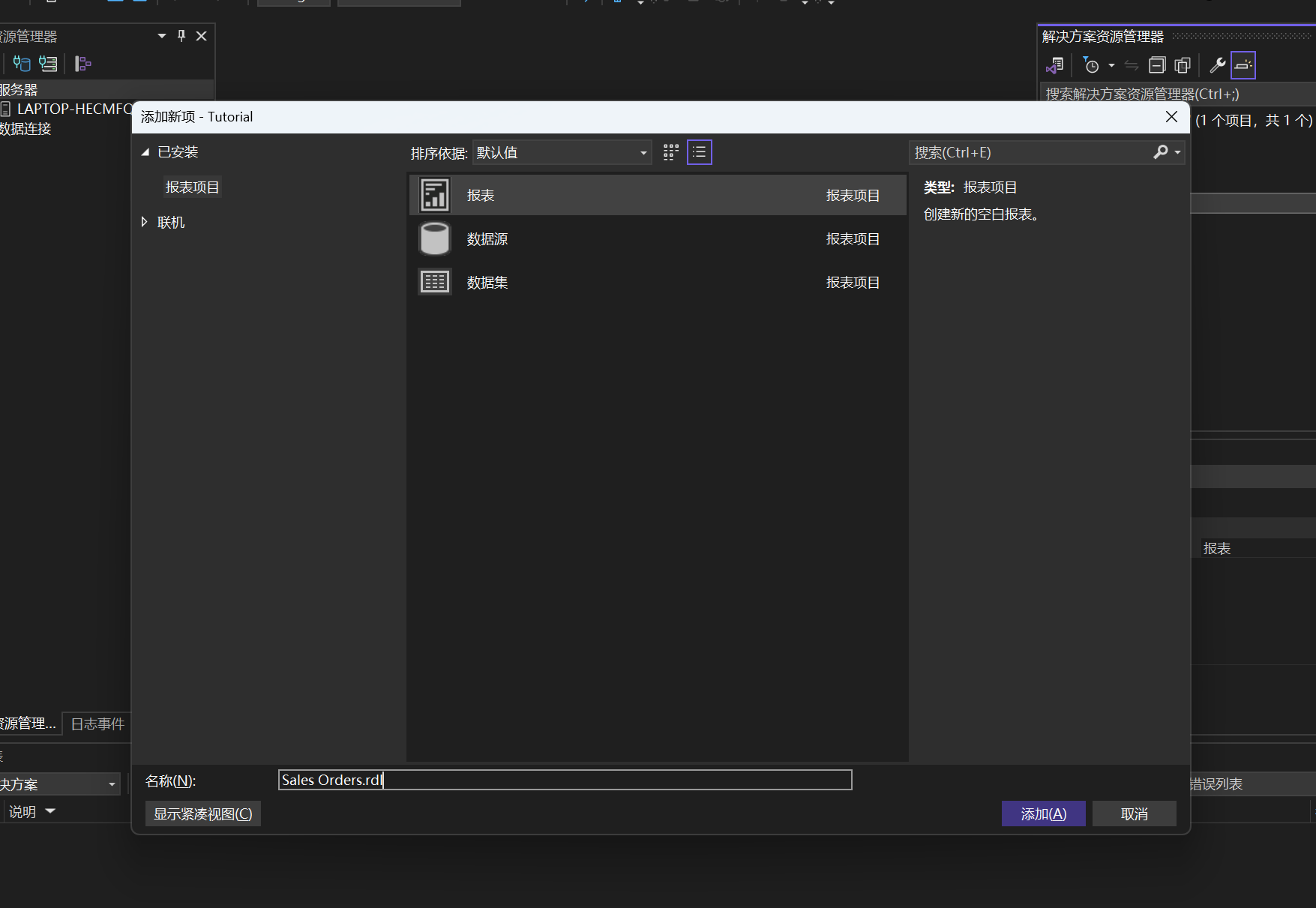
**创建基本表报表的方法和步骤**

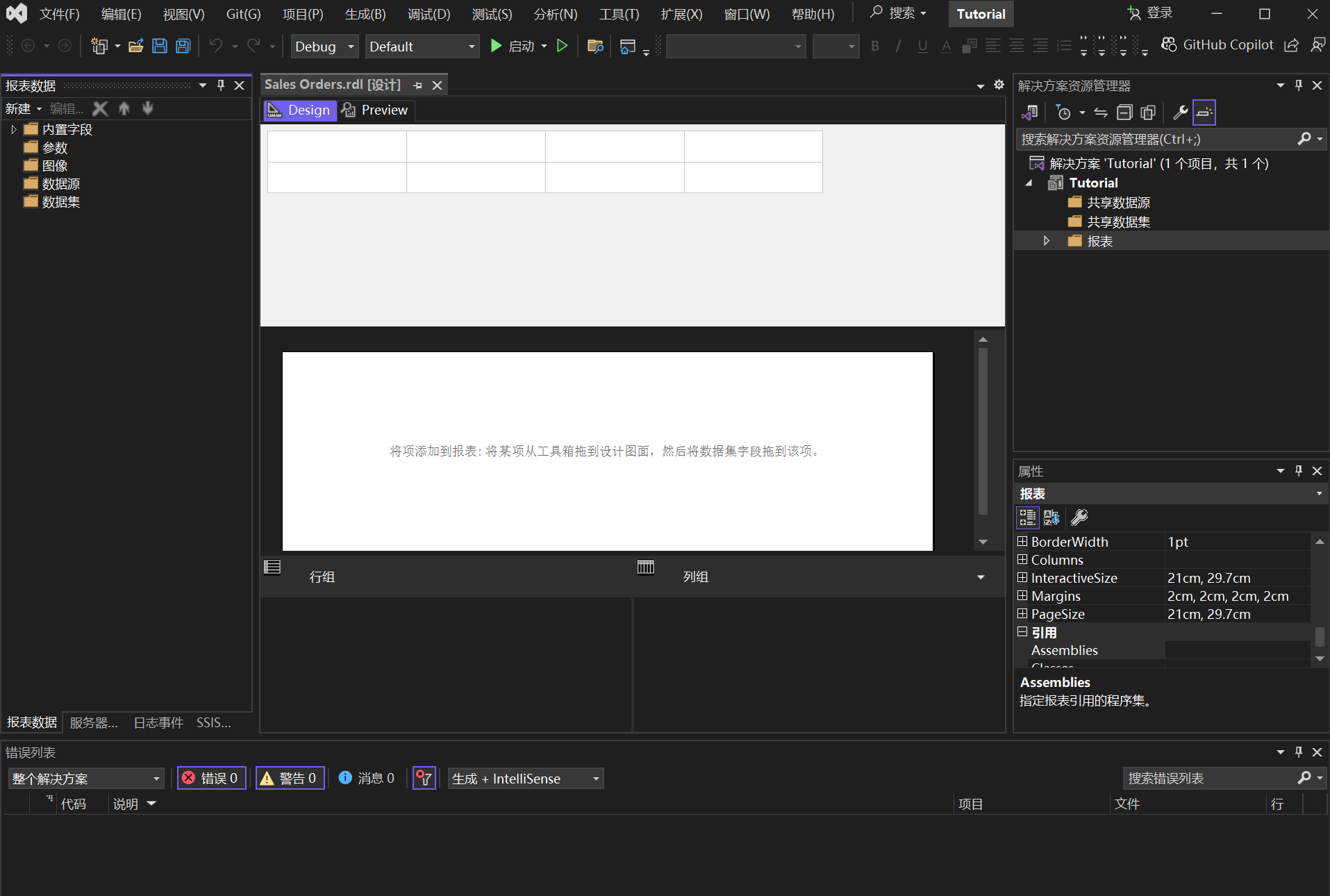
**（1）创建报表服务器项目**

**创建 Report Server 项目**



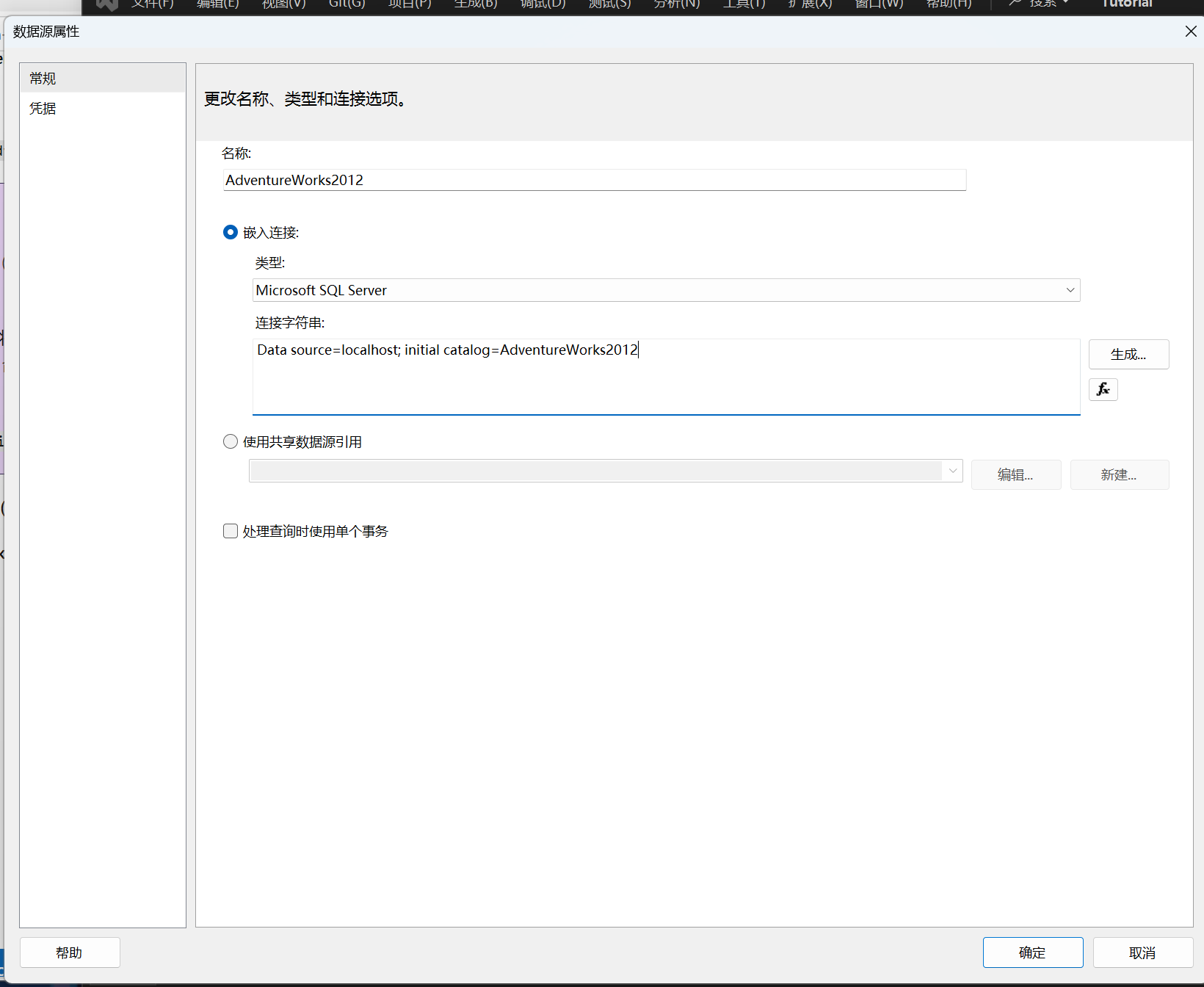
**创建报表定义文件 (RDL)**

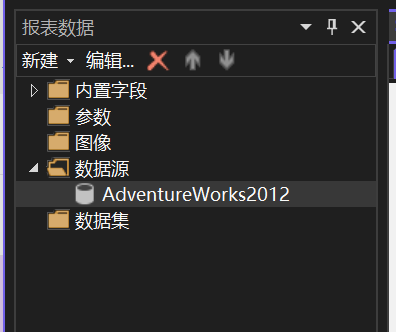




**(2)指定连接信息 (Reporting Services)**

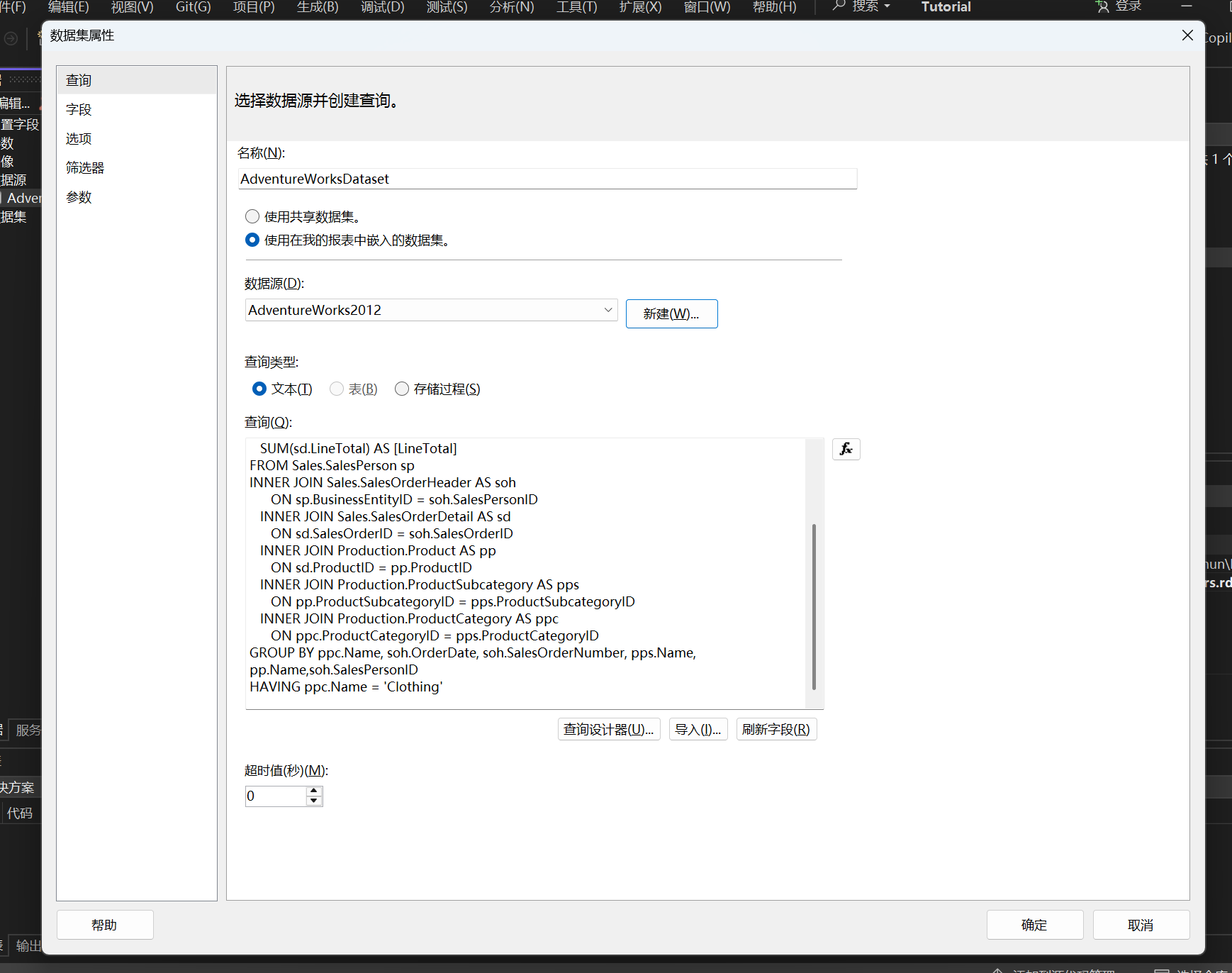
**设置连接**

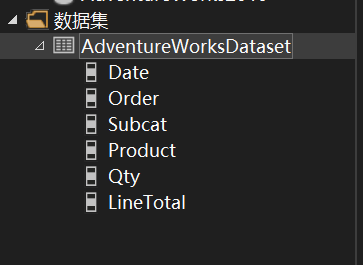




**(3)为表报表定义数据集 (Reporting Services)**

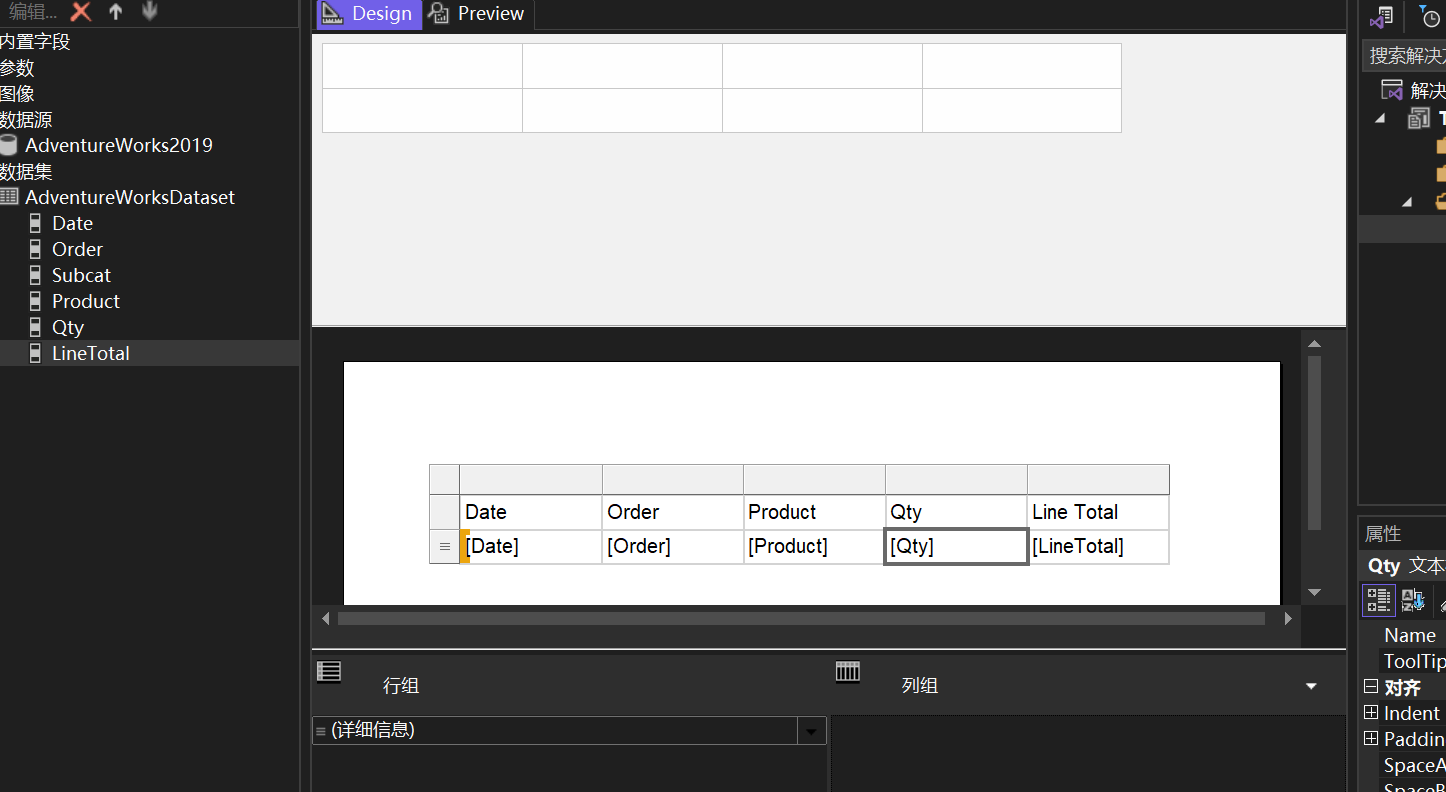
**为报表数据定义 Transact-SQL 查询**





**(4)向报表添加表 (Reporting Services)**

**向报表布局添加表数据区域和字段**

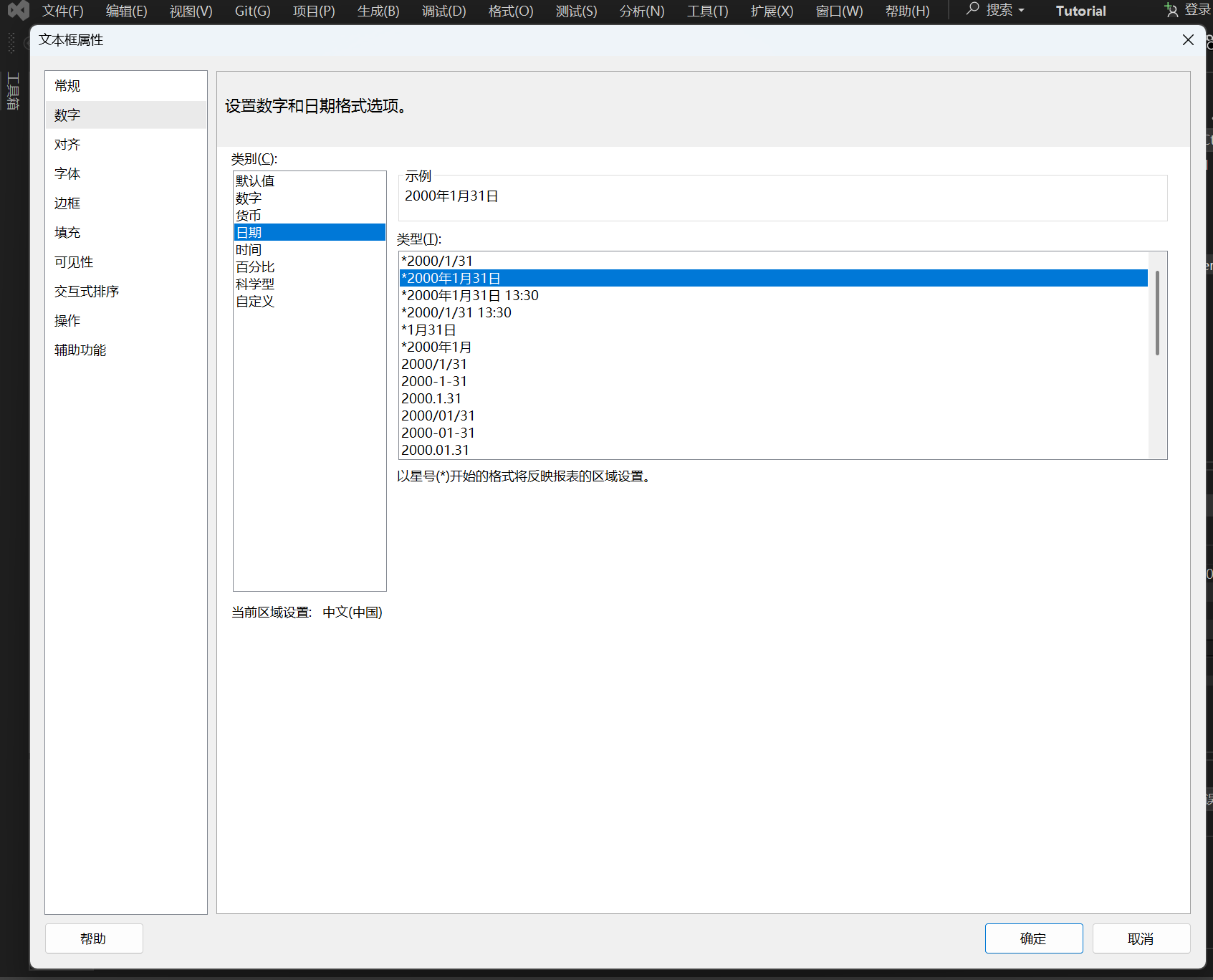


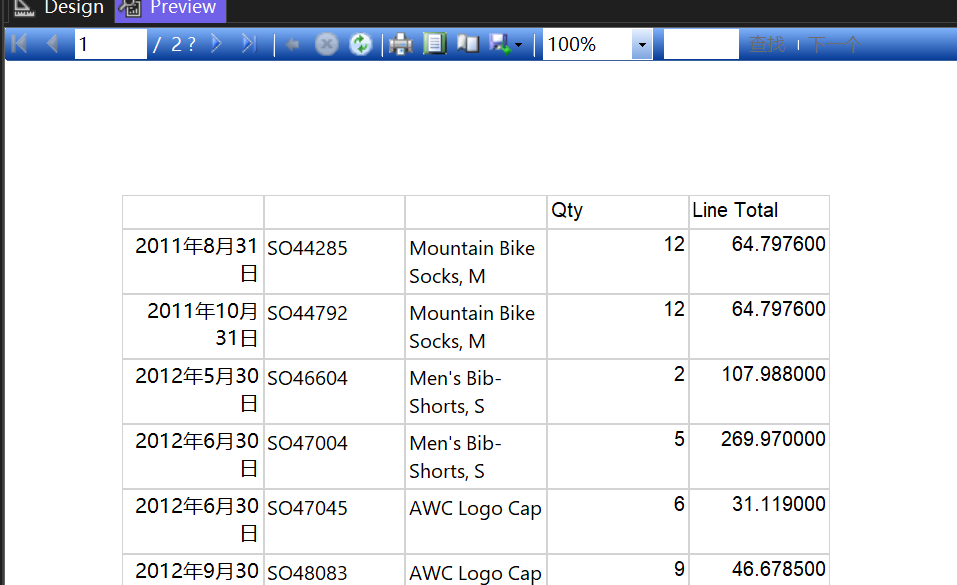
**预览报表**



**(5)设置报表的格式 (Reporting Services)**

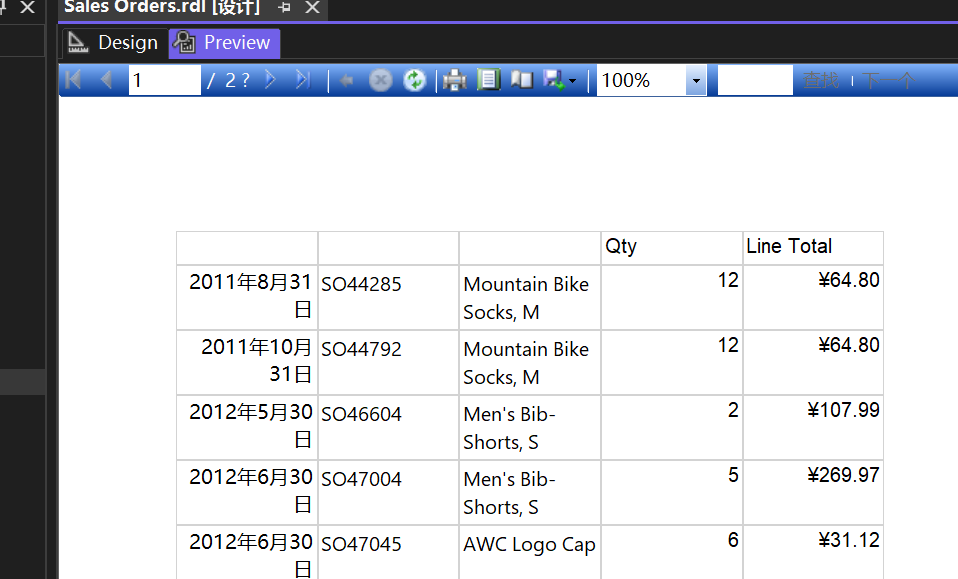
**设置日期格式**



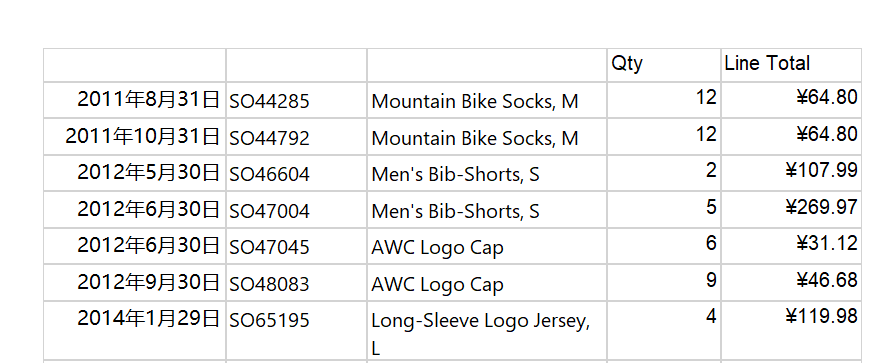


**设置货币格式**





**更改文本样式和列宽**

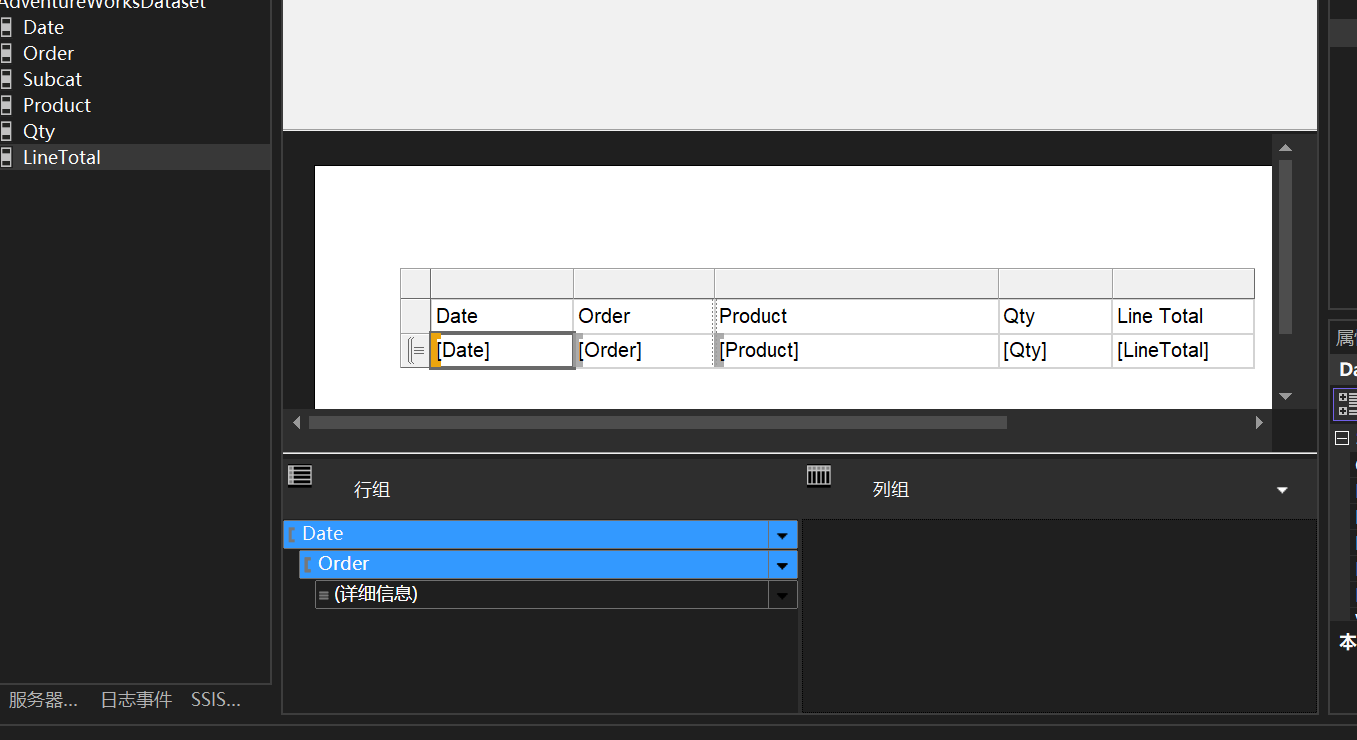


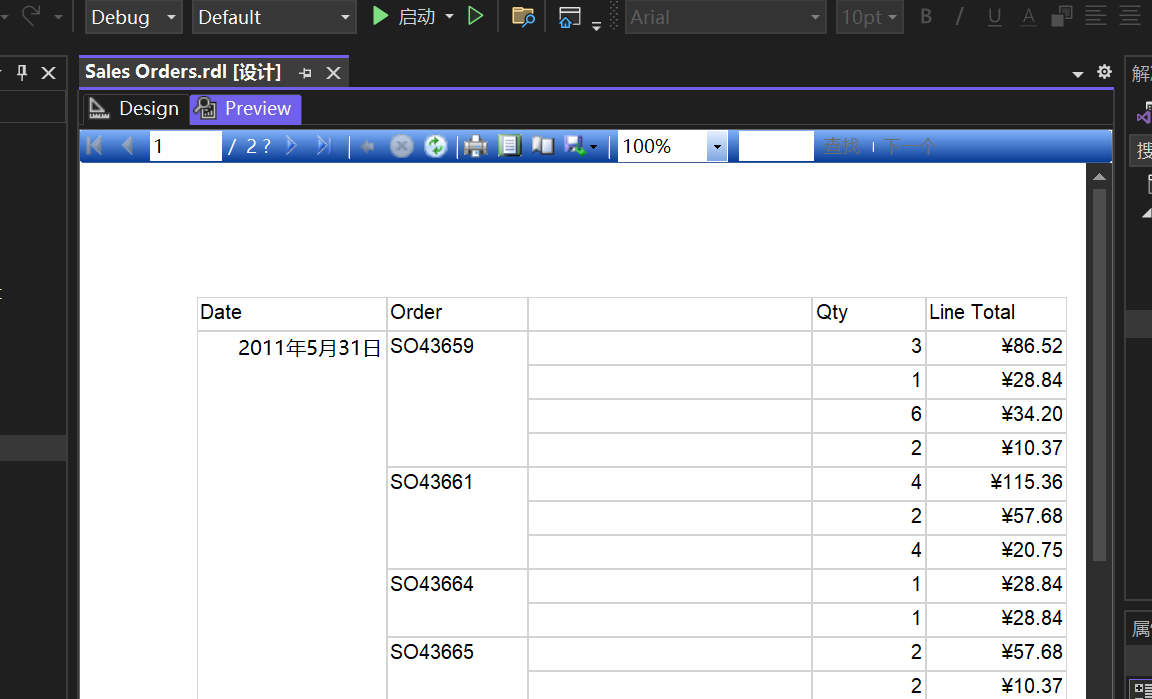
**(6)添加分组和总计 (Reporting Services)**

**在报表中对数据进行分组**

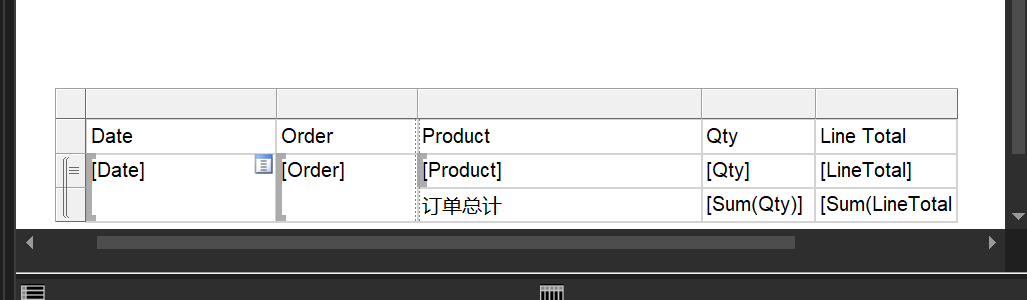
**从“报表数据”窗格将 [Date] 字段拖到“行组”窗格。 将其置于显示为 = (Details) 的行之上。**

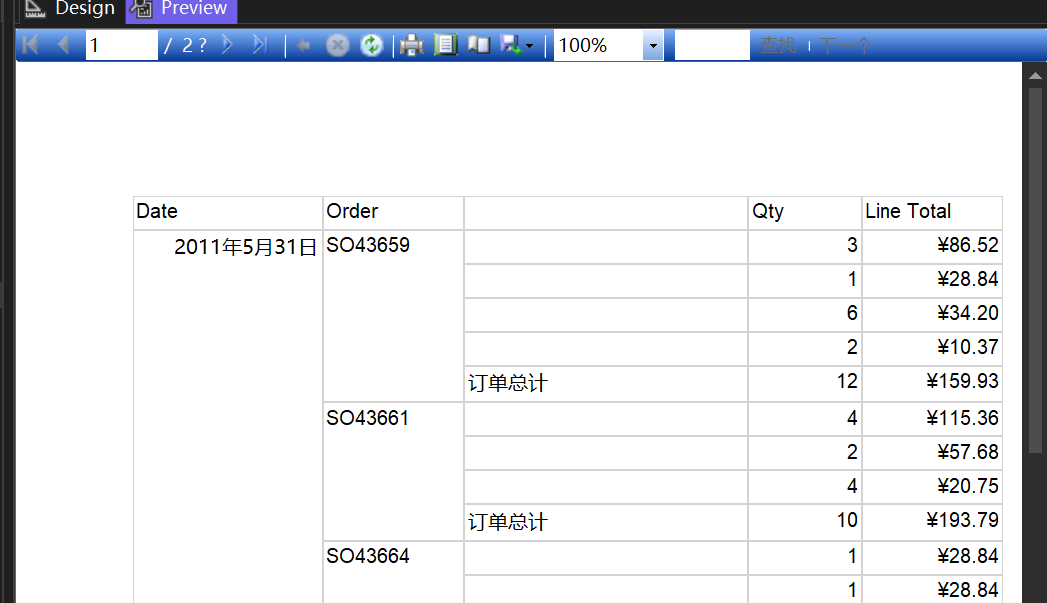
**从“报表数据”窗格将 [Order] 字段拖到“行组”窗格。 并将其放置到 Date 后面和 = (Details) 前面。**



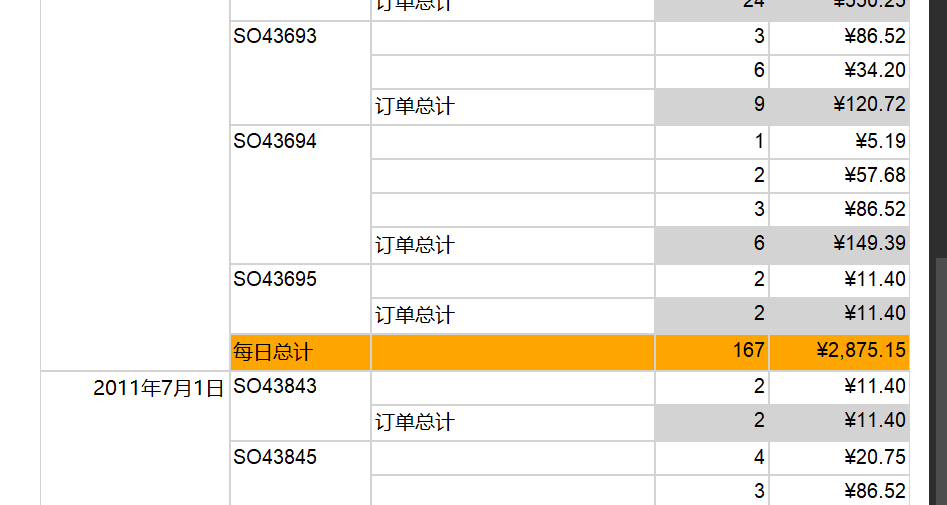


**向报表添加总计**





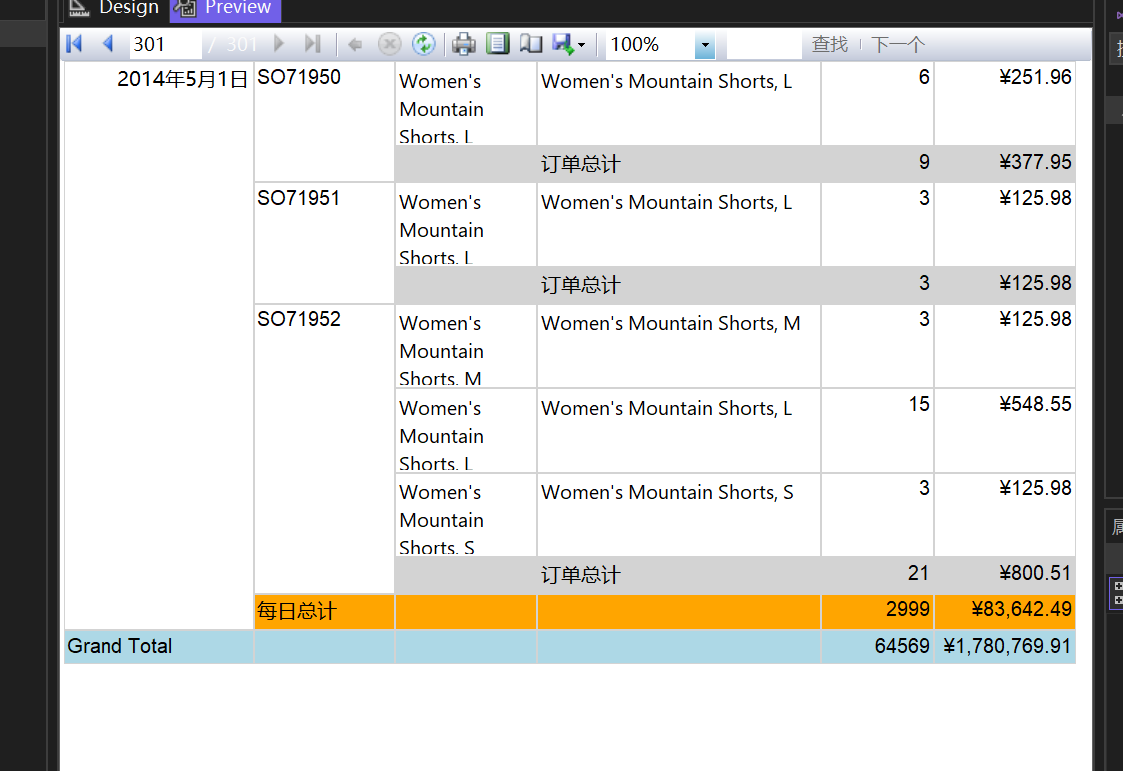
**向报表添加每日总计**



**向报表添加总计**



**预览报表**



**不知道为什么拉俩个product就会显示，只有一个就不显示**

**（6）将报表发布到报表服务器**

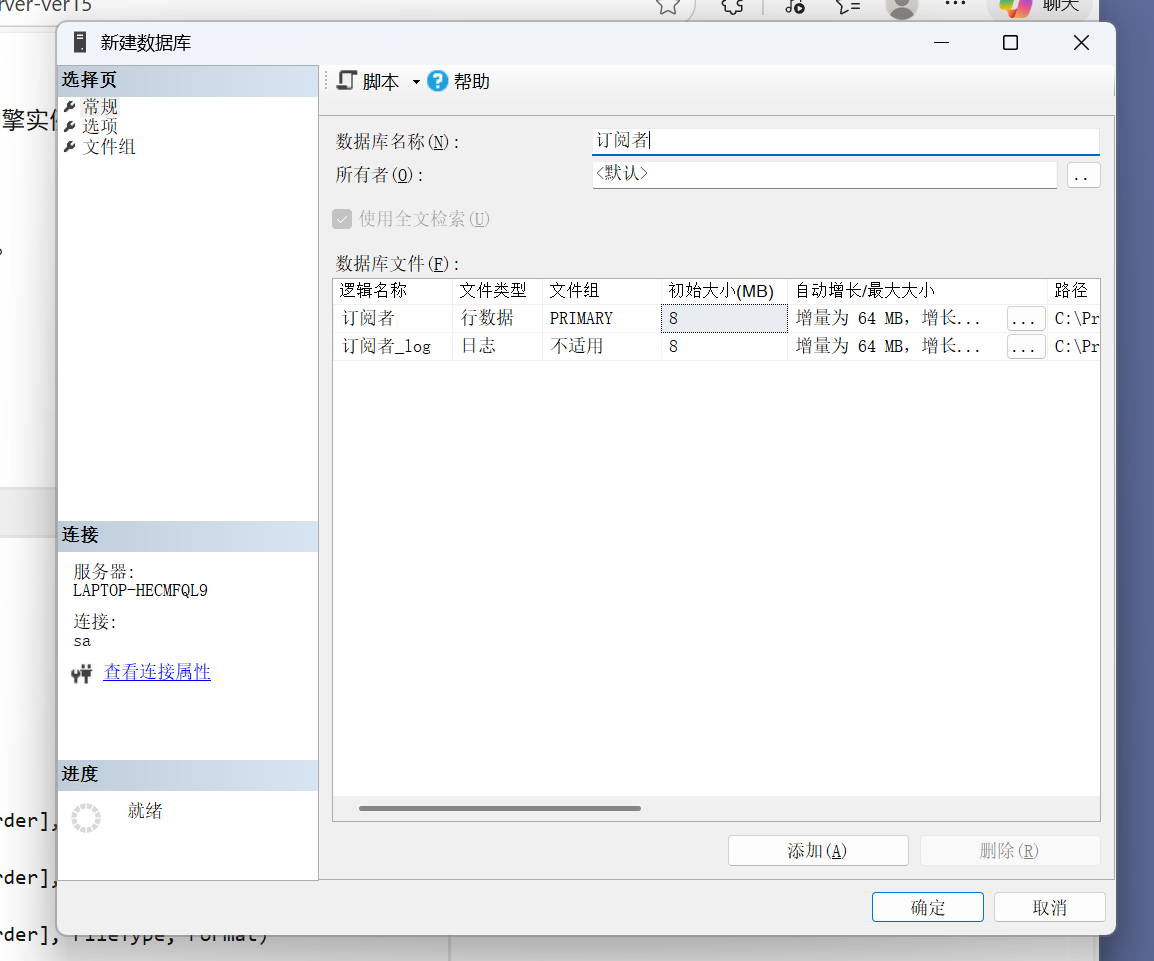


**创建数据驱动订阅的方法和步骤**

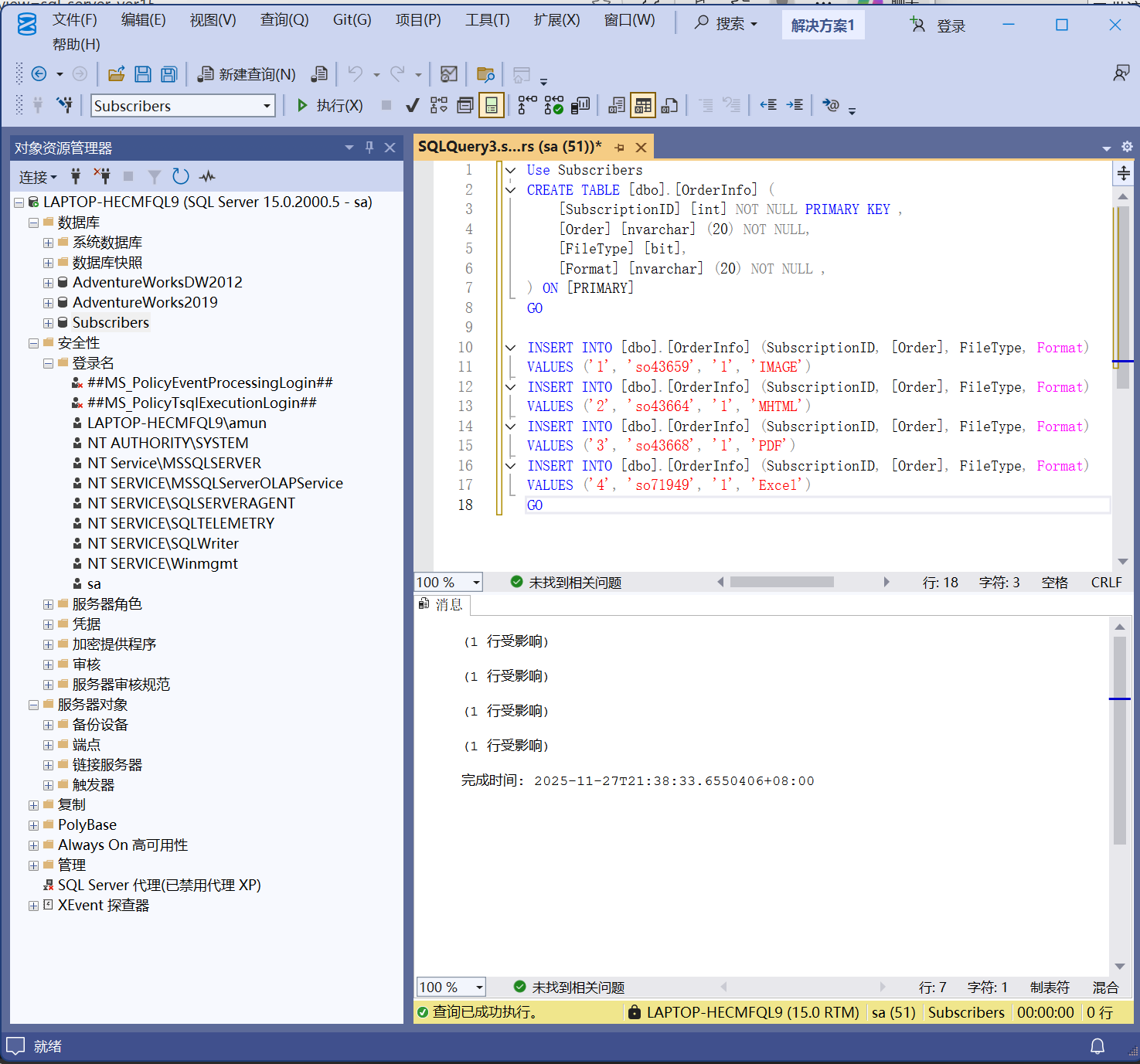
**创建数据驱动订阅**

**（1）创建示例订阅服务器数据库**

**在“新建数据库”对话框的“数据库名称”中，键入“订阅者”**

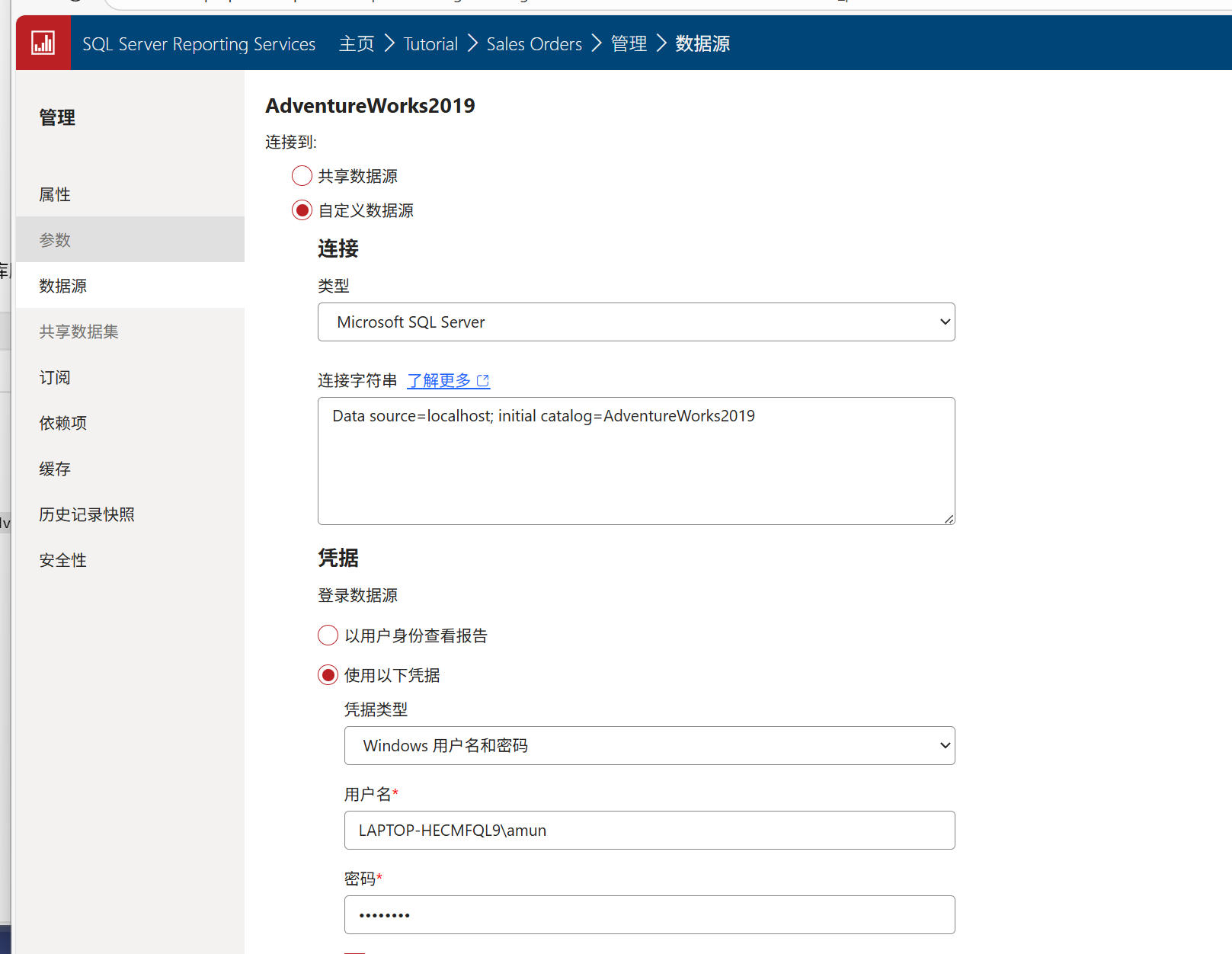


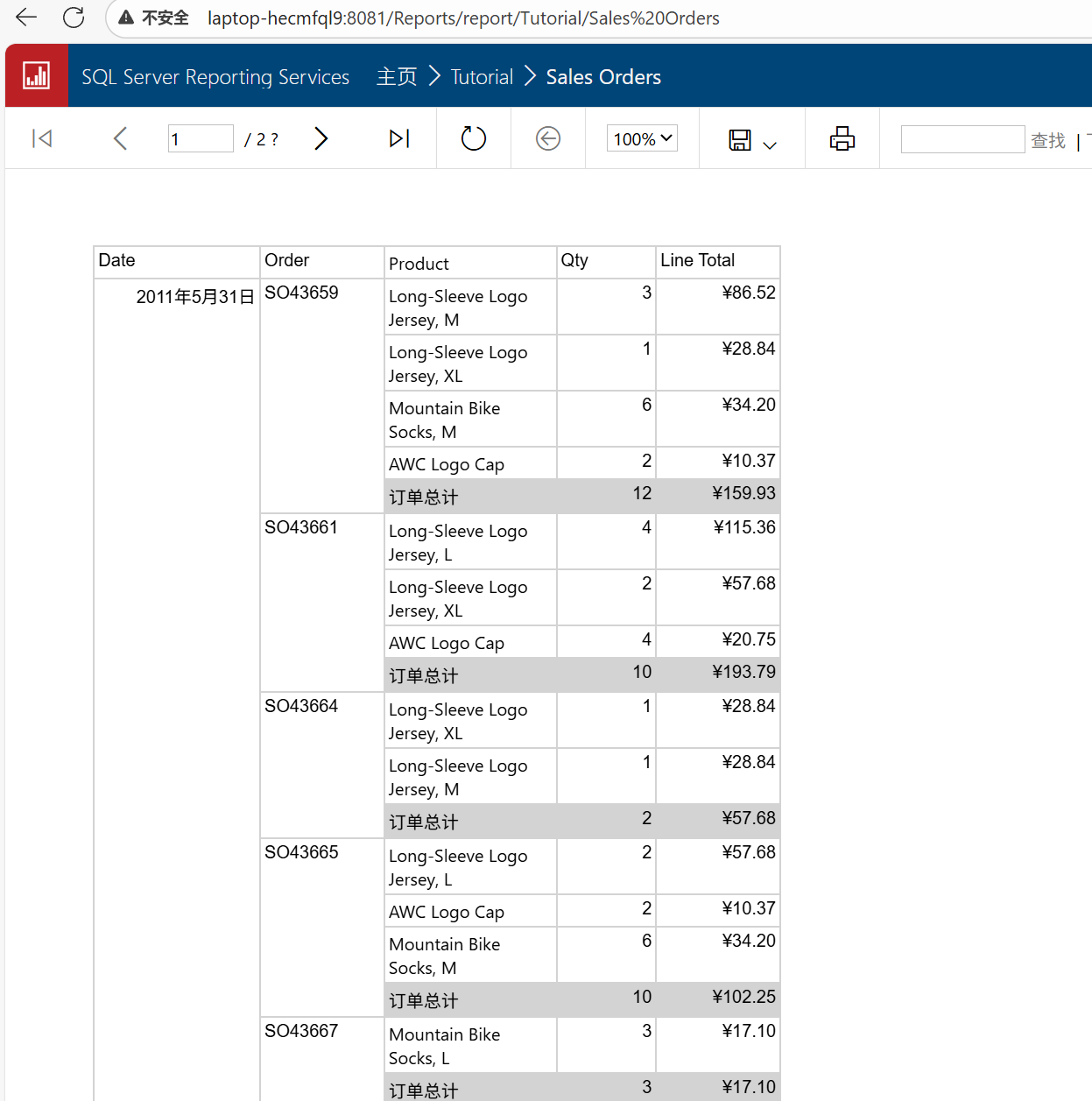
**进行查询**



**（2）修改报表数据源属性**

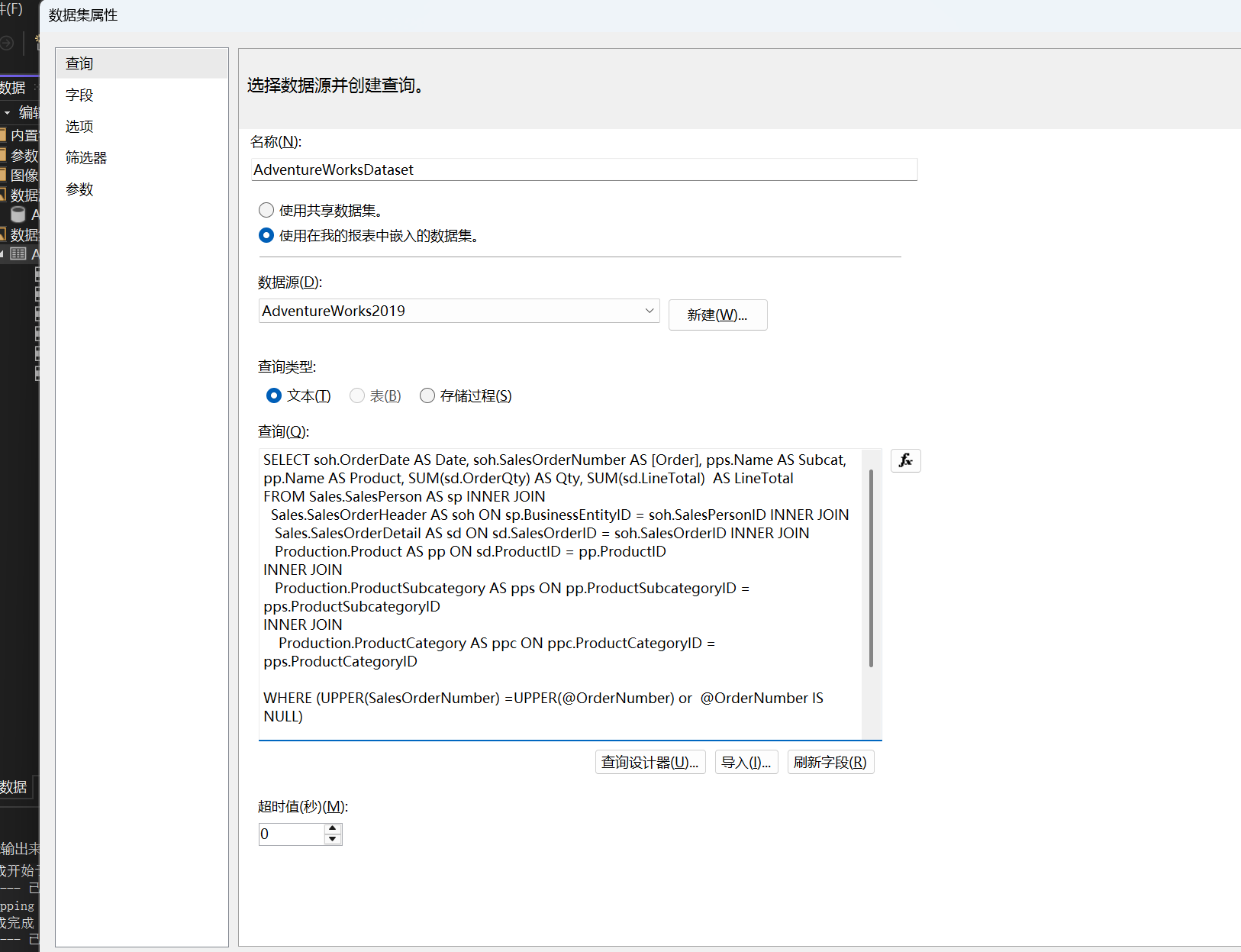
**修改数据源以使用存储的凭据**



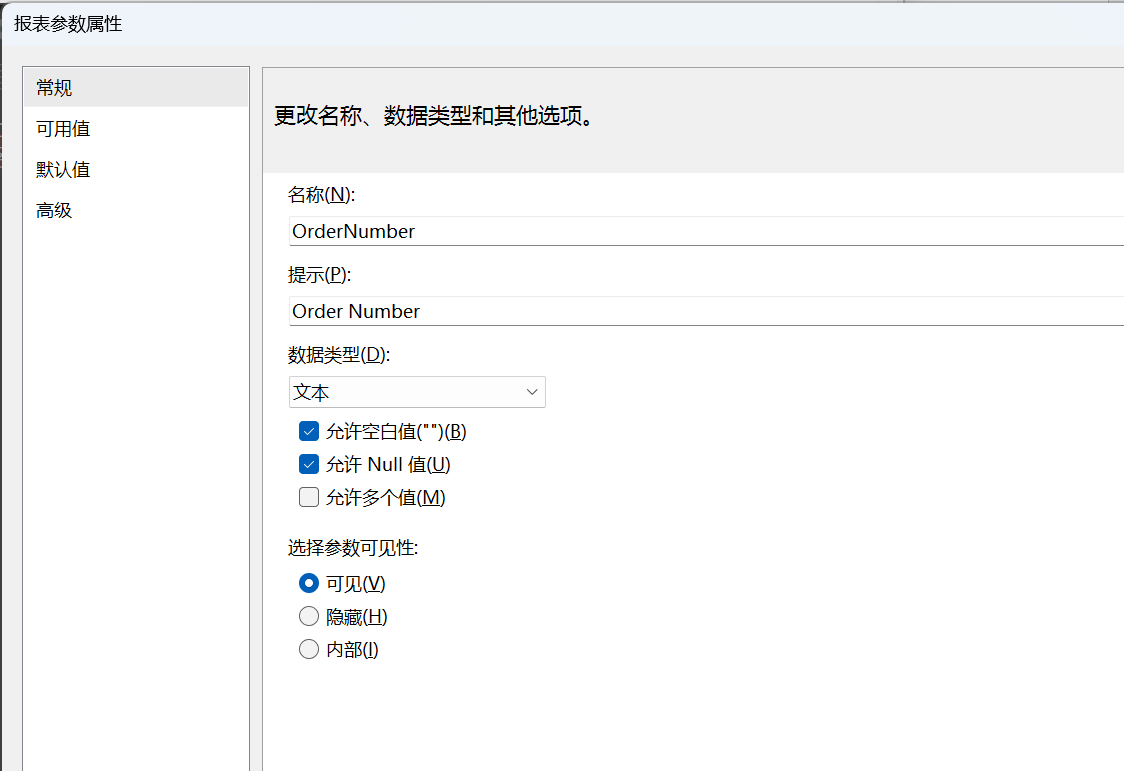


**修改 AdventureWorksDataset**

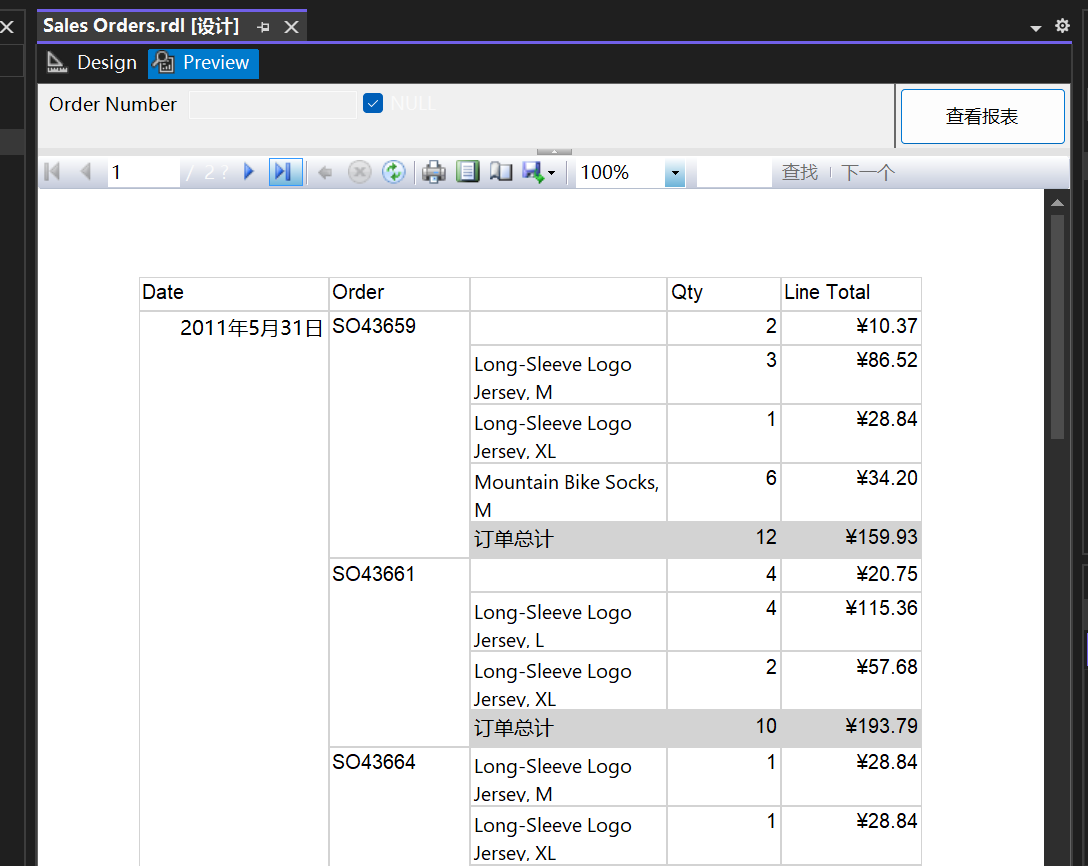
**将语句 WHERE (UPPER(SalesOrderNumber) =UPPER(@OrderNumber) or @OrderNumber IS NULL) 添加到 Group By 语句之前**



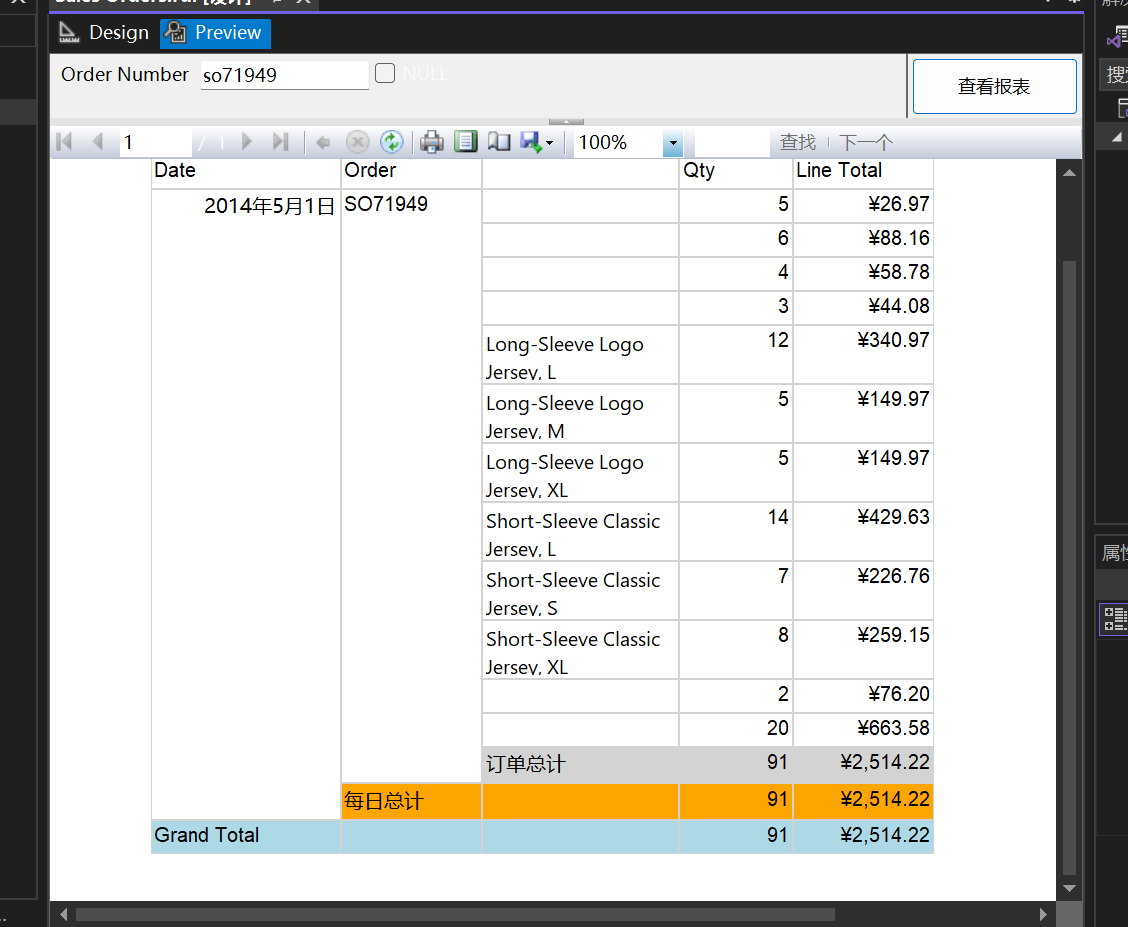
**添加报表参数并重新发布报表**



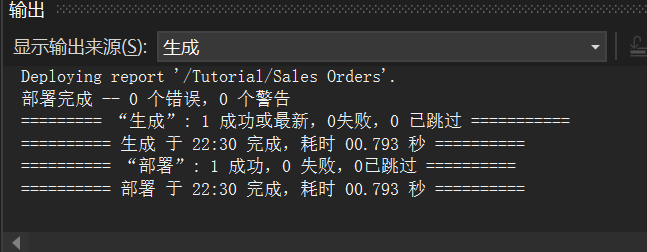
**在不使用参数的情况下选择“查看报表”以便看到完整的报表**



**取消选择“Null”选项并输入订单号，例如 *so71949*，然后选择“查看报表”以便只在报表中查看这一个订单**



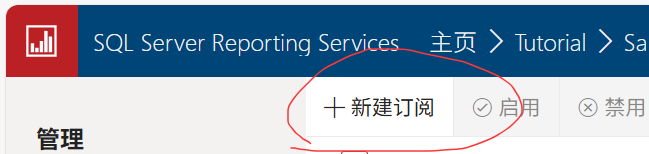
**重新部署报表**



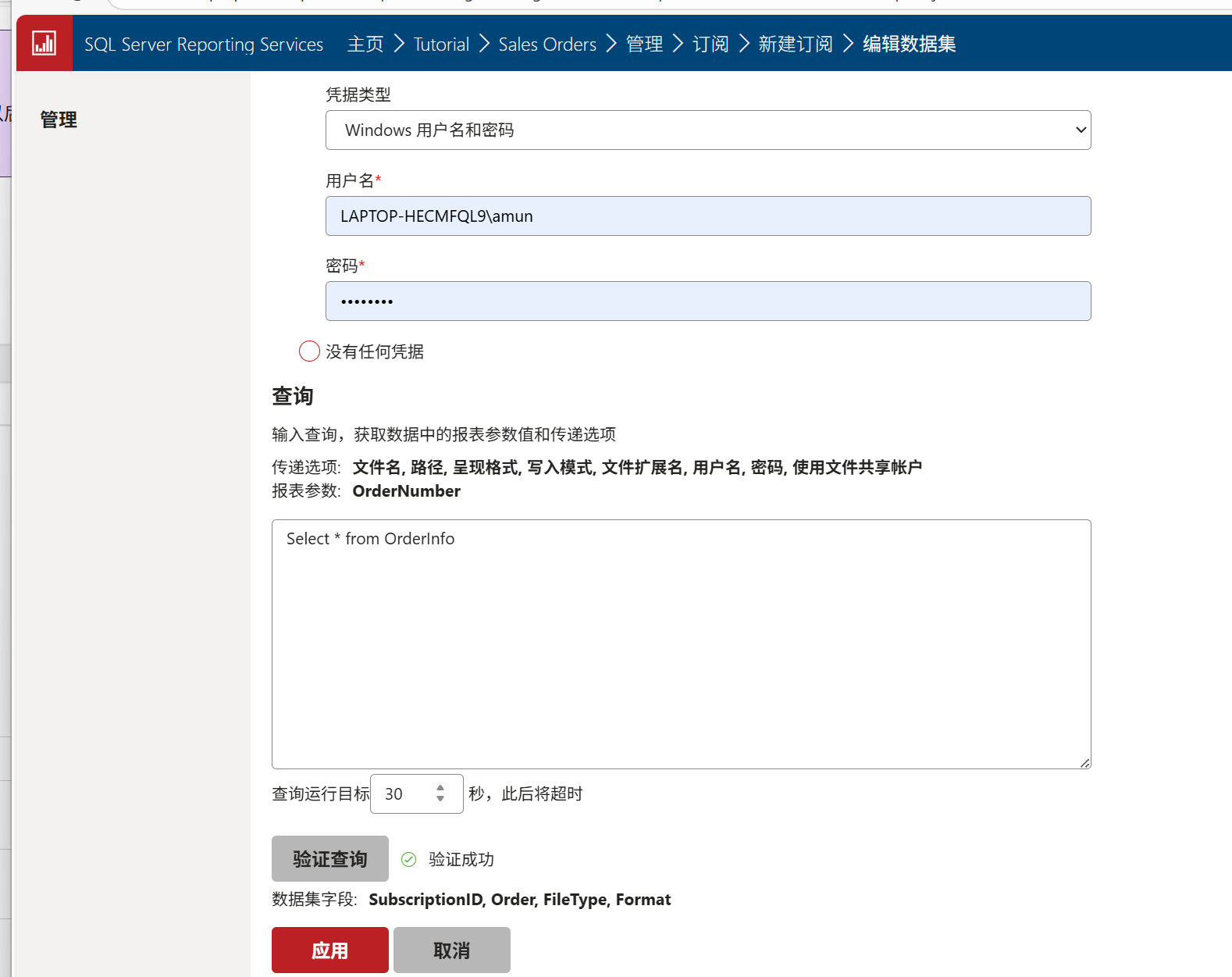
**（3）定义数据驱动订阅**

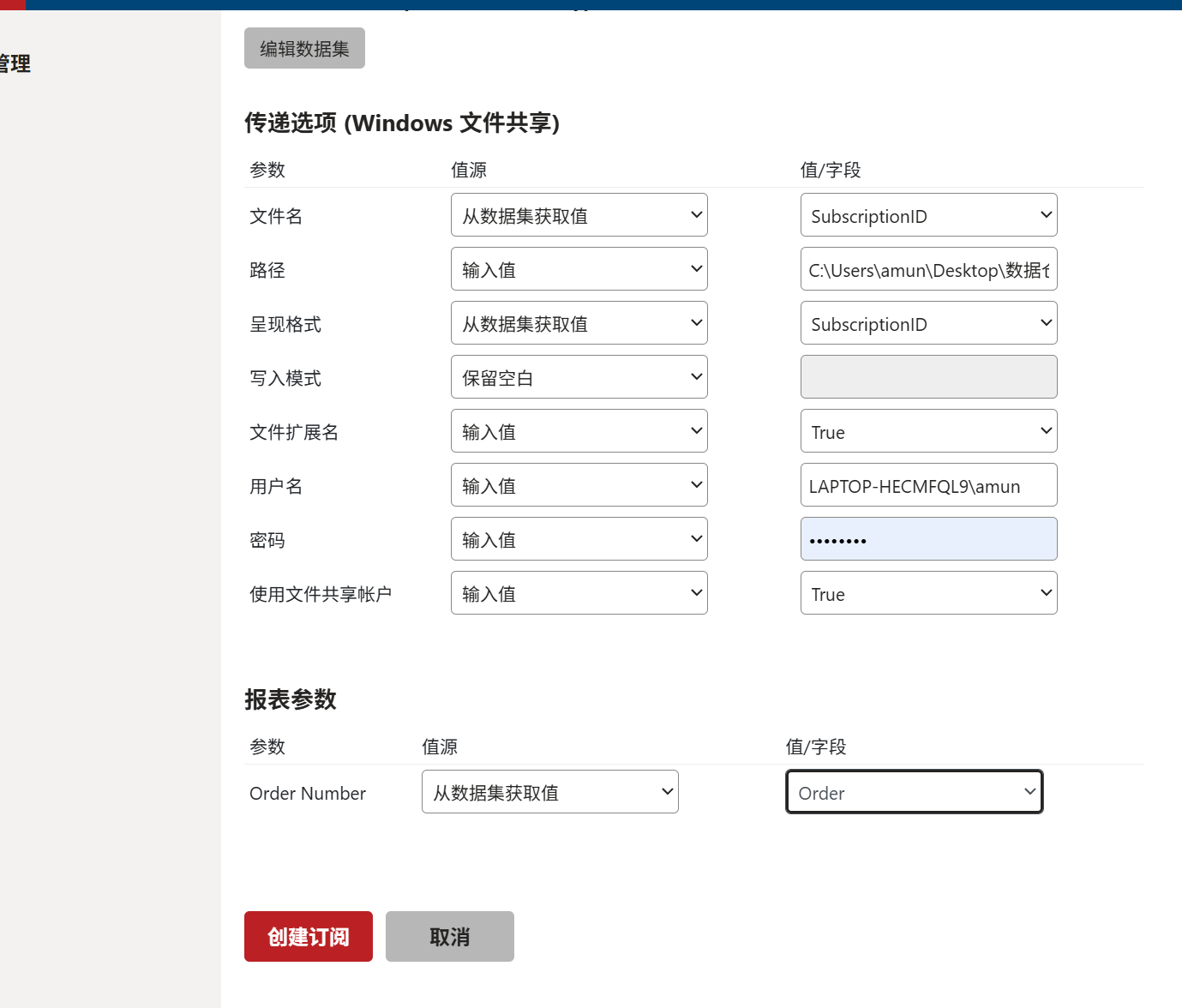
**启动数据驱动订阅向导**

**点击新建订阅**



**按要求填写完所有数据**





**点击创建订阅**

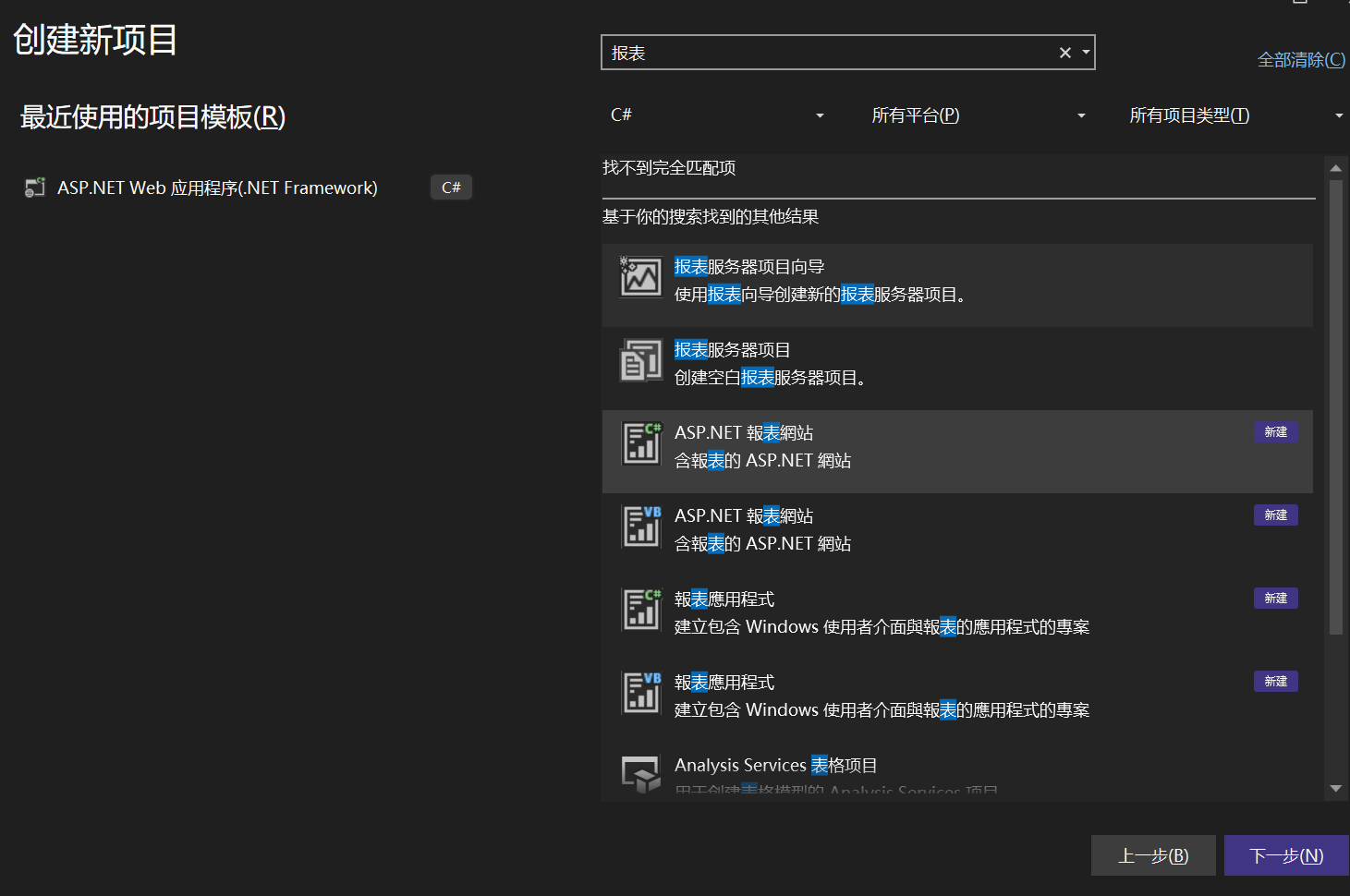


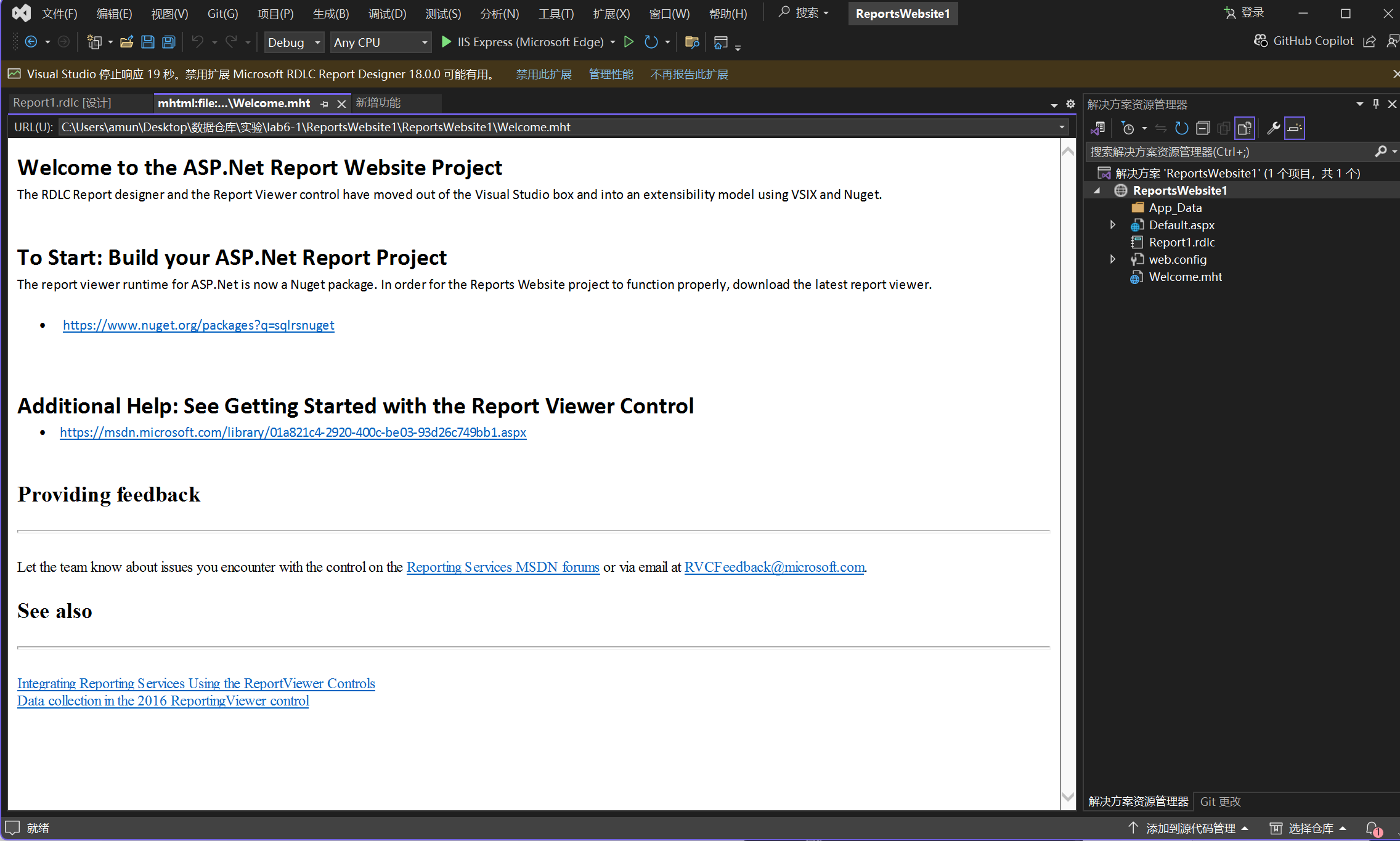
**使用 ReportViewer 创建带有参数的钻取报表的方法与步骤**

**创建带有参数的钻取 (RDLC) 报表 – ReportViewer**

**（1）创建一个新网站**

**在VS上创建一个ASP.NET 报表网站项目**

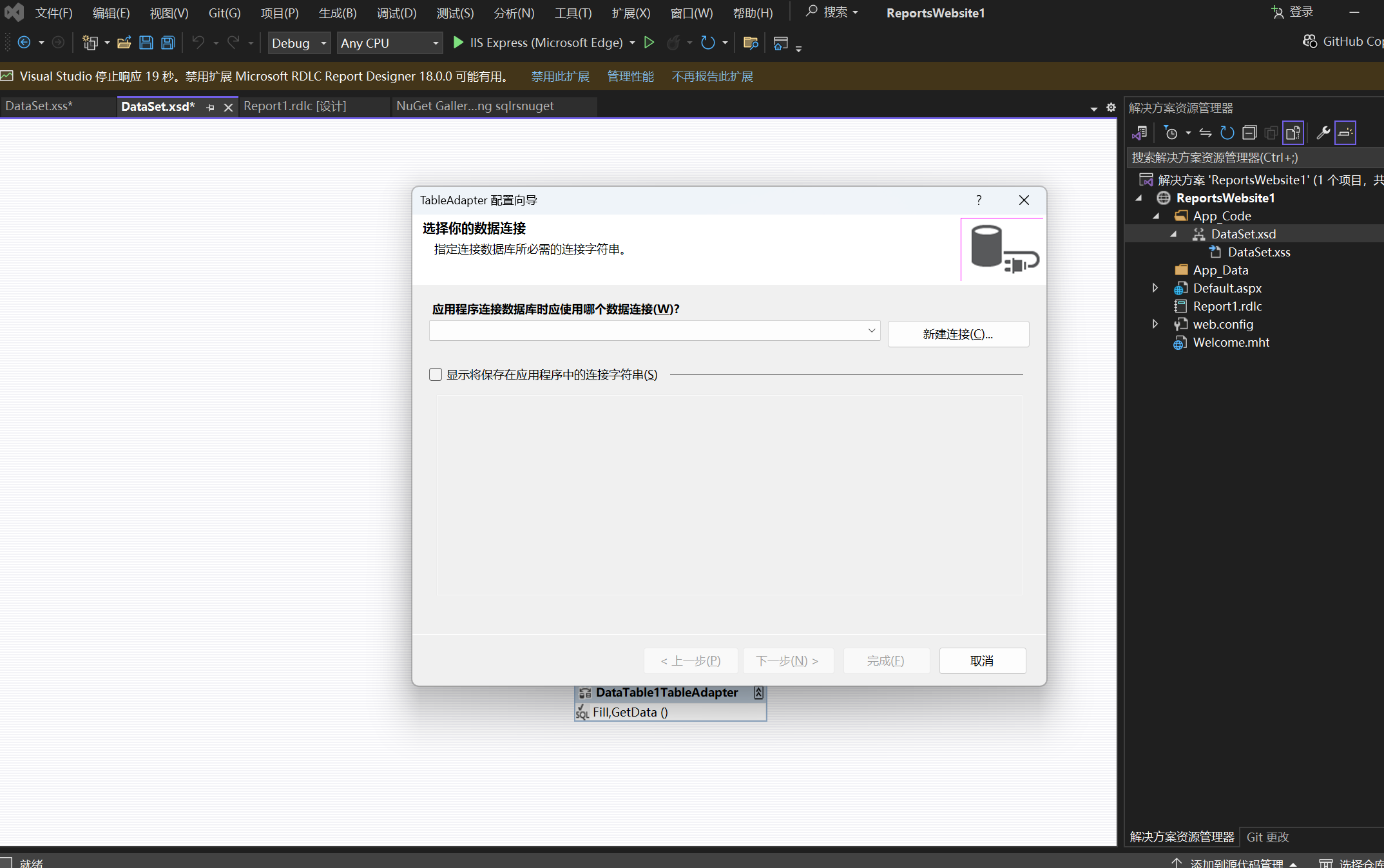




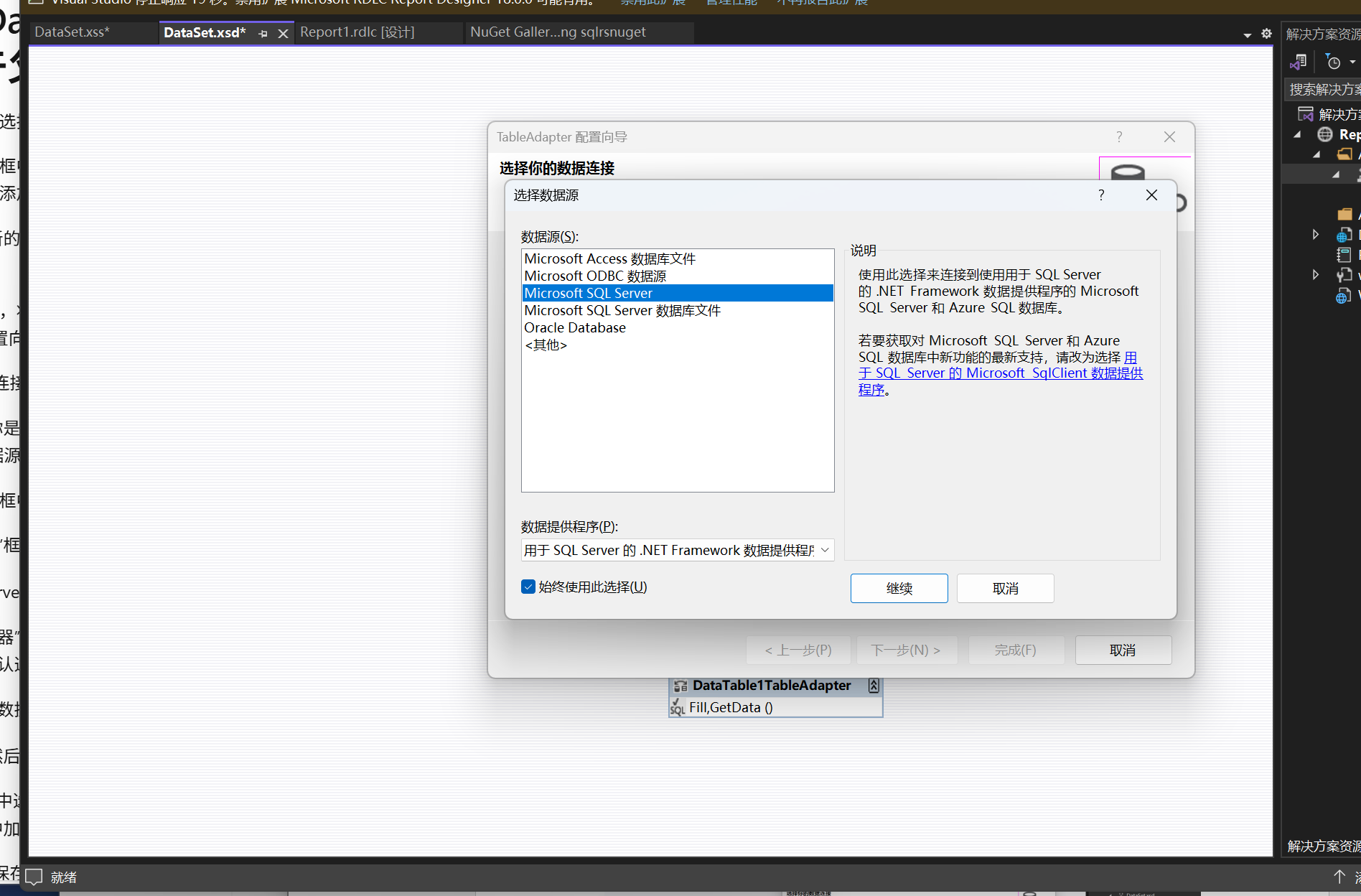
**（2）定义用于父报表的数据连接和数据表**

**通过添加 DataSet 定义数据连接和数据表（用于父报表）**

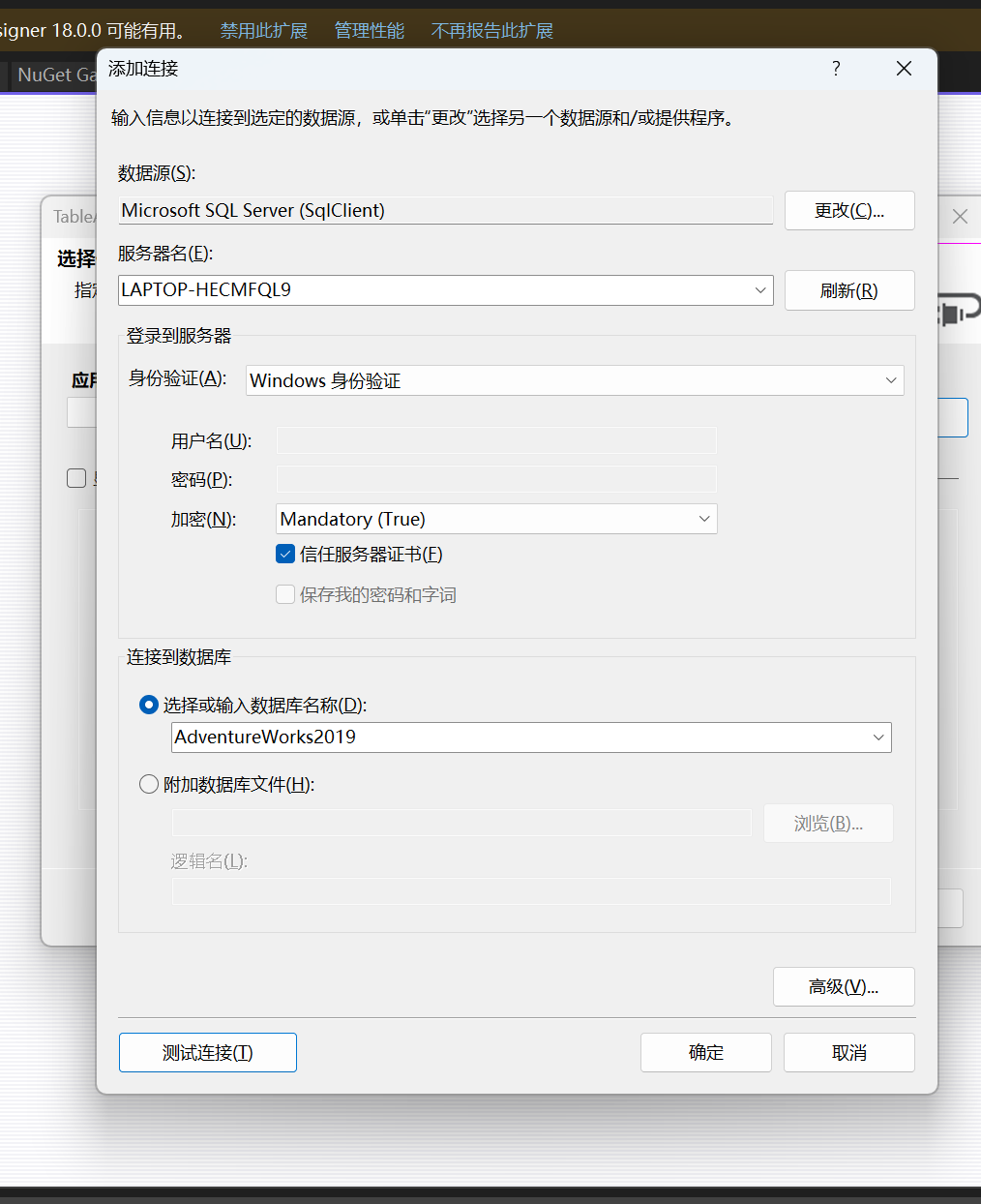
**从“工具箱”窗口中，将 [TableAdapter](https://learn.microsoft.com/zh-cn/visualstudio/data-tools/fill-datasets-by-using-tableadapters) 控件拖到设计图面上**

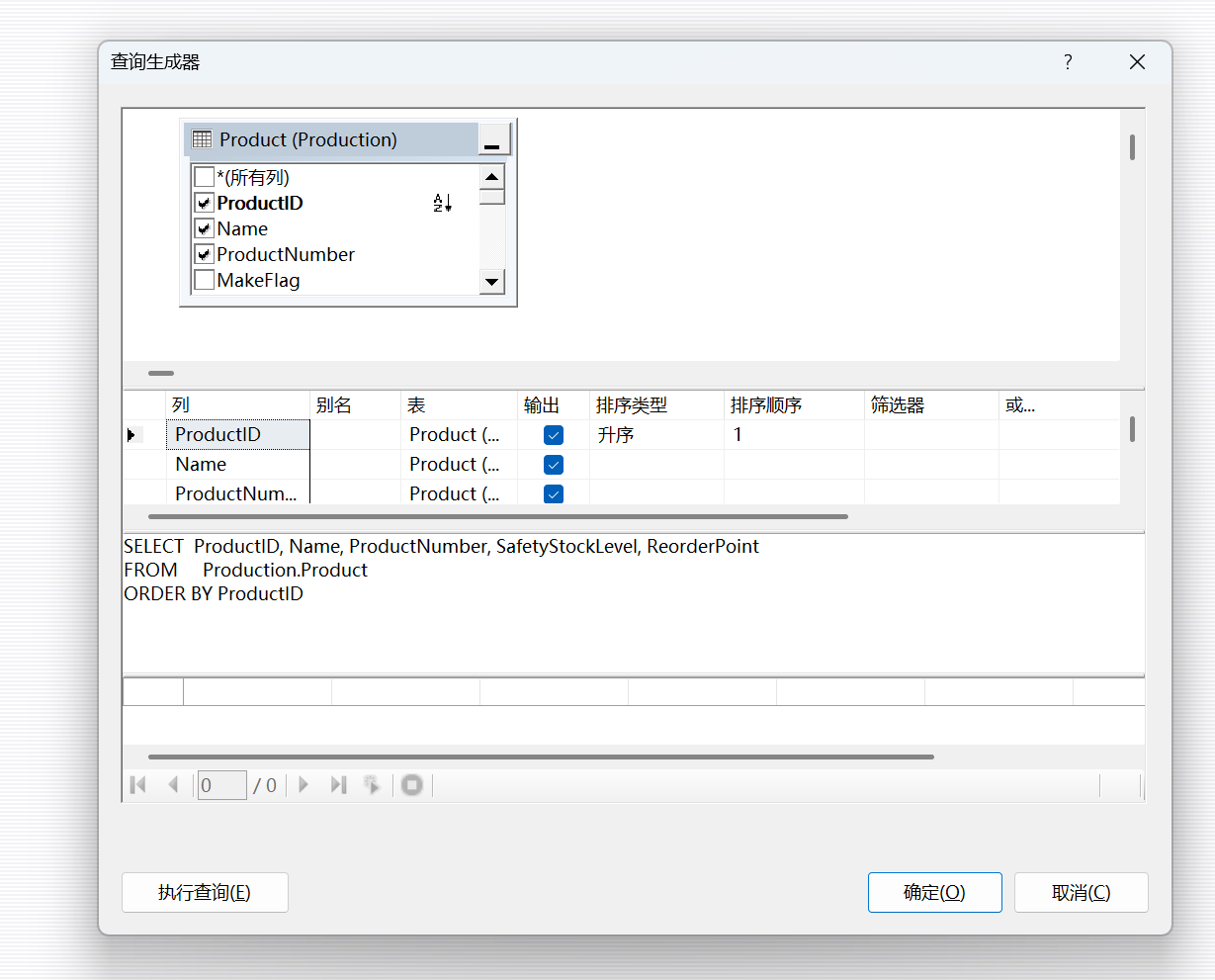


**新建连接**

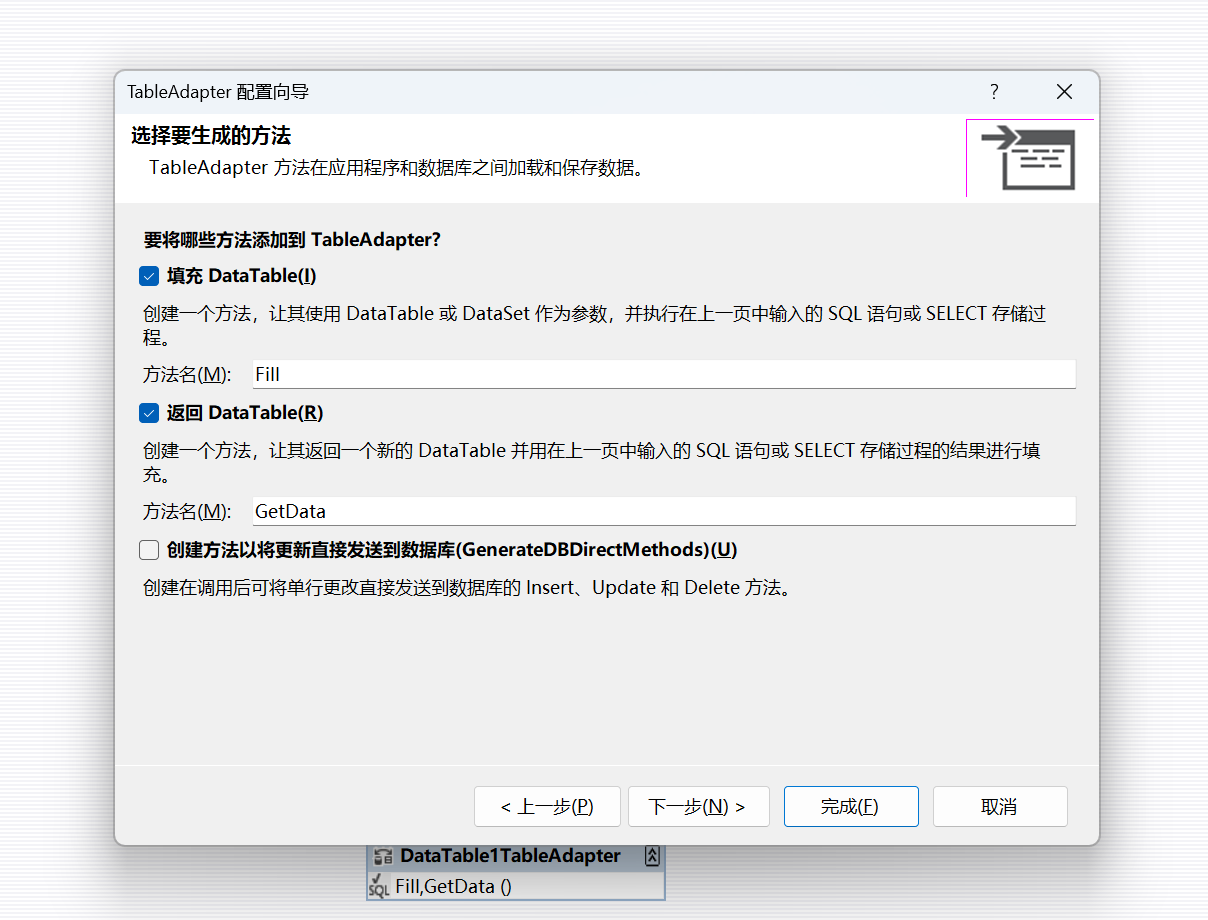


**添加连接**



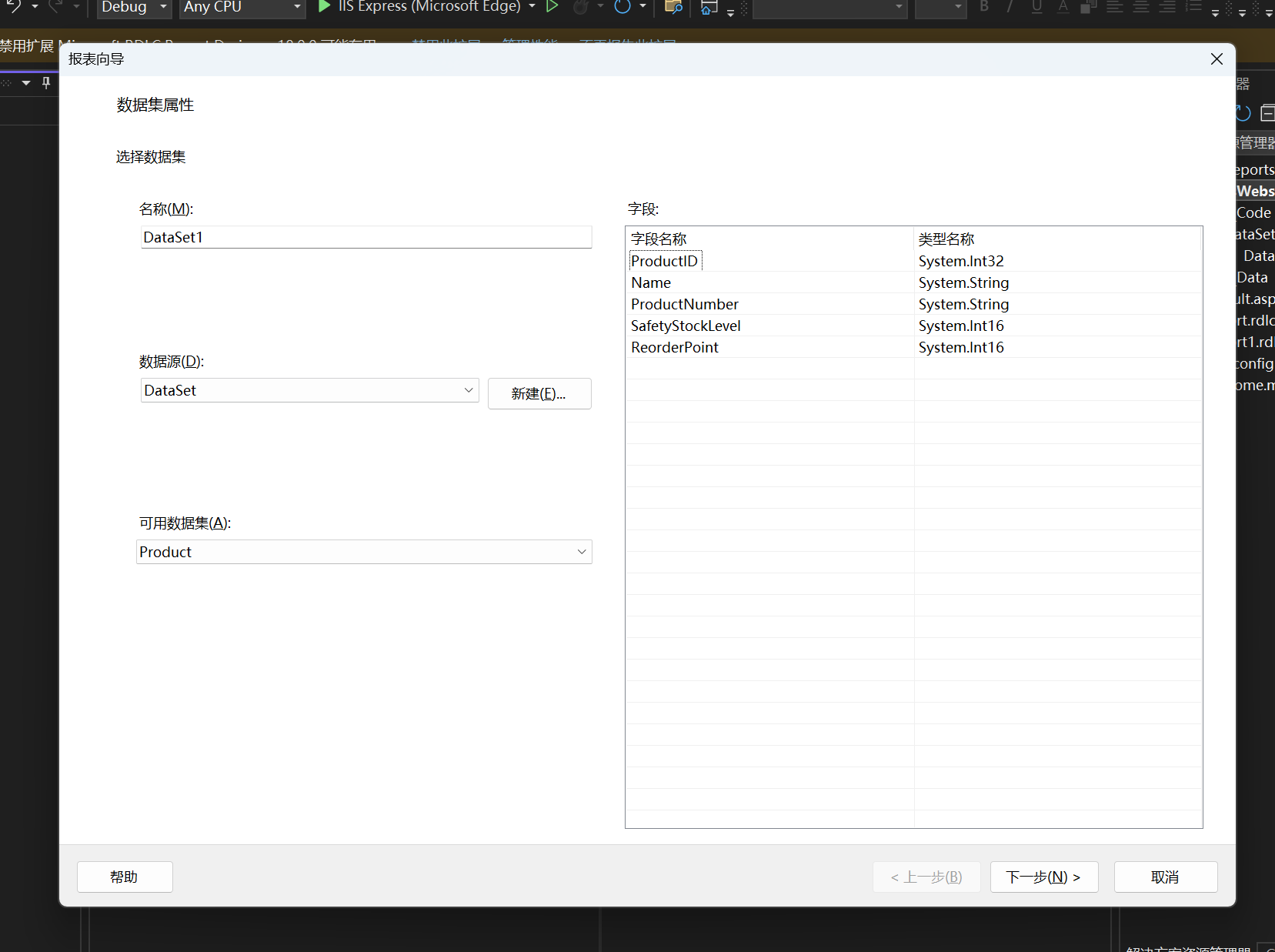


**配置完成**

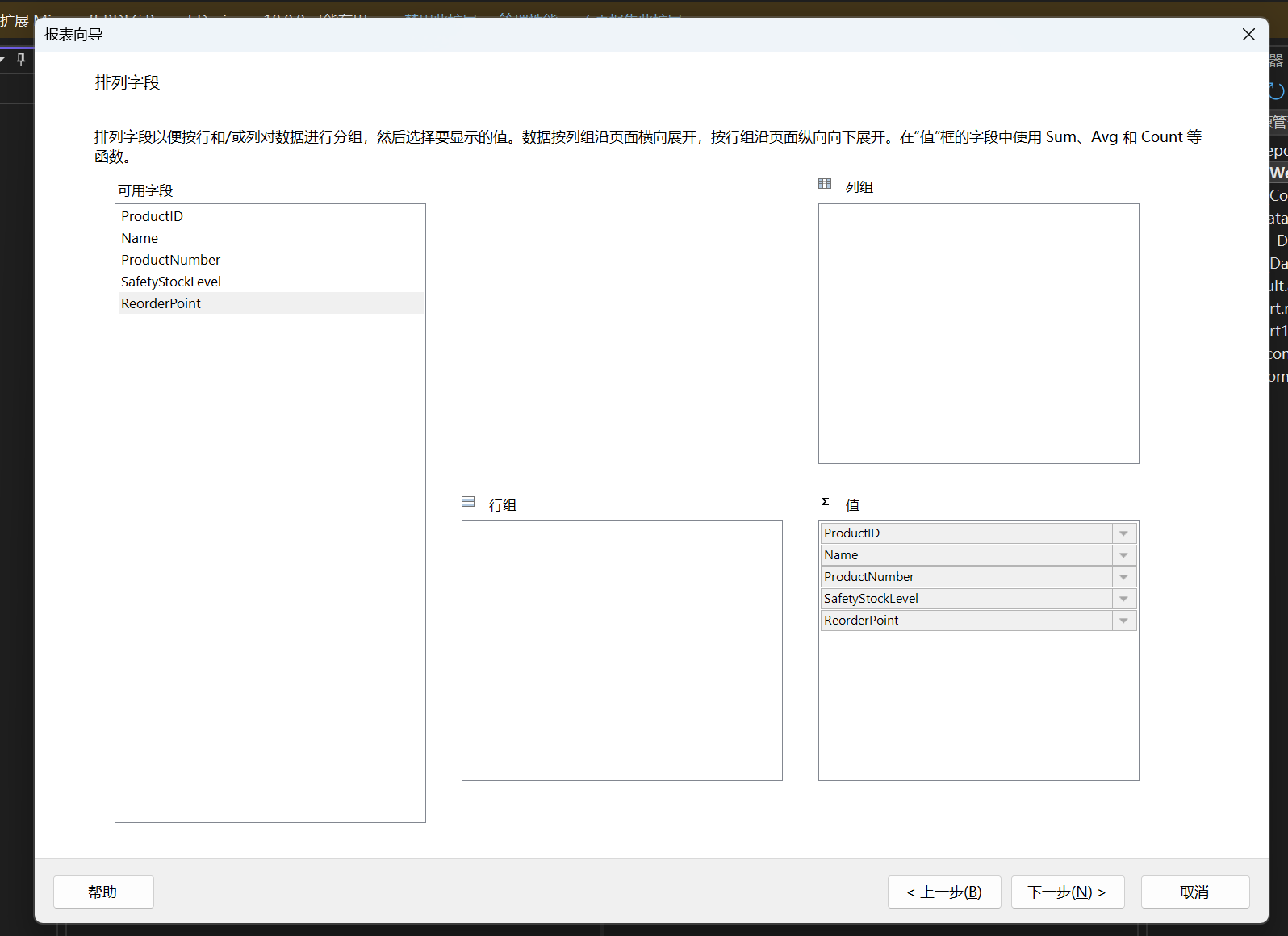


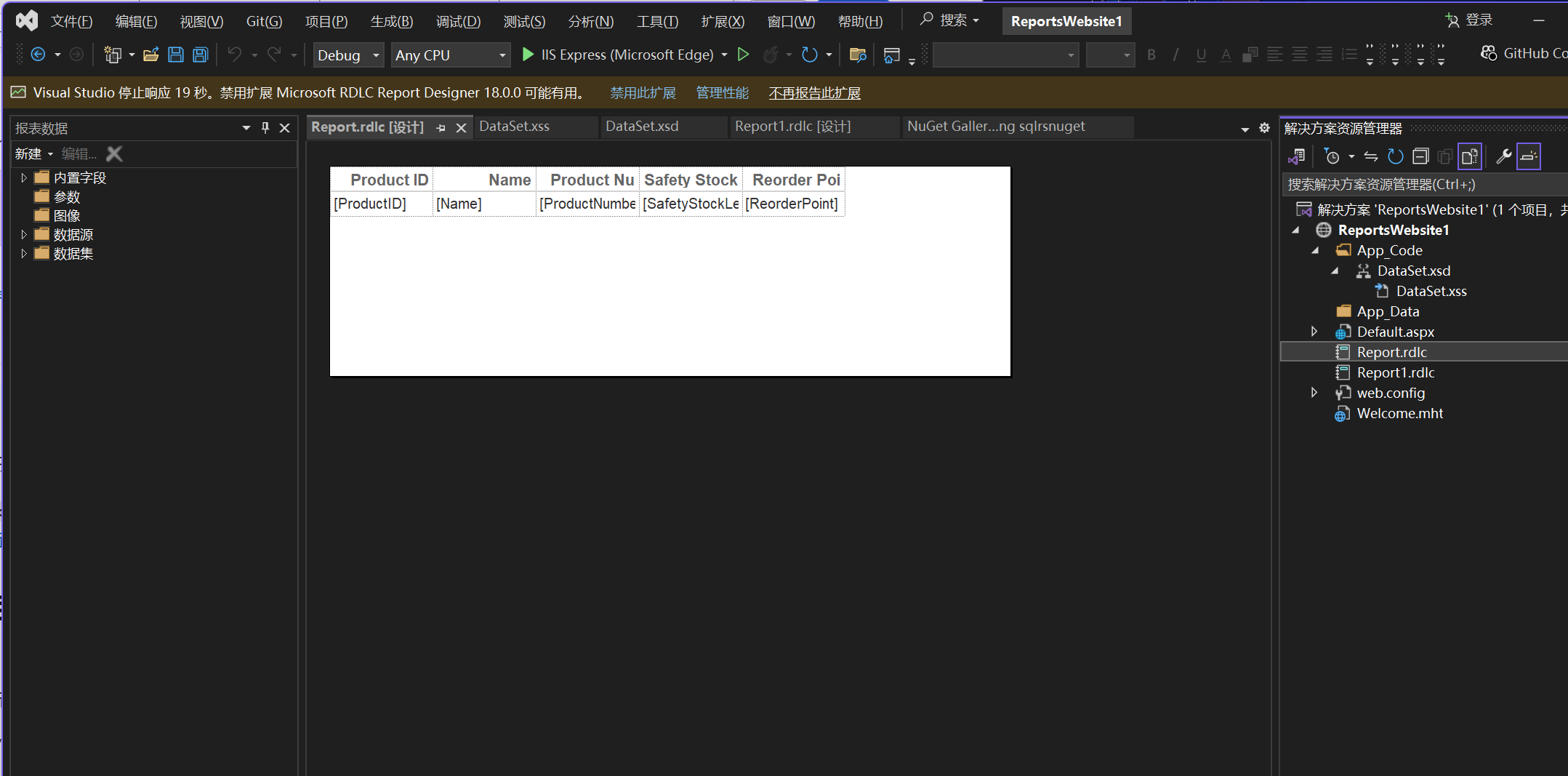
**（3）使用报表向导设计父报表**

**创建报表向导**



1. **将“ProductID”、“Name”、“ProductNumber”、“SafetyStockLevel”和“ReorderLevel”从“可用字段”拖至“值”框中 。**
2. **选择“Sum(ProductID)”、“Sum(SafetyStockLevel)”,、“Sum(ReorderLevel)”旁的箭头，然后取消选择“Sum”**



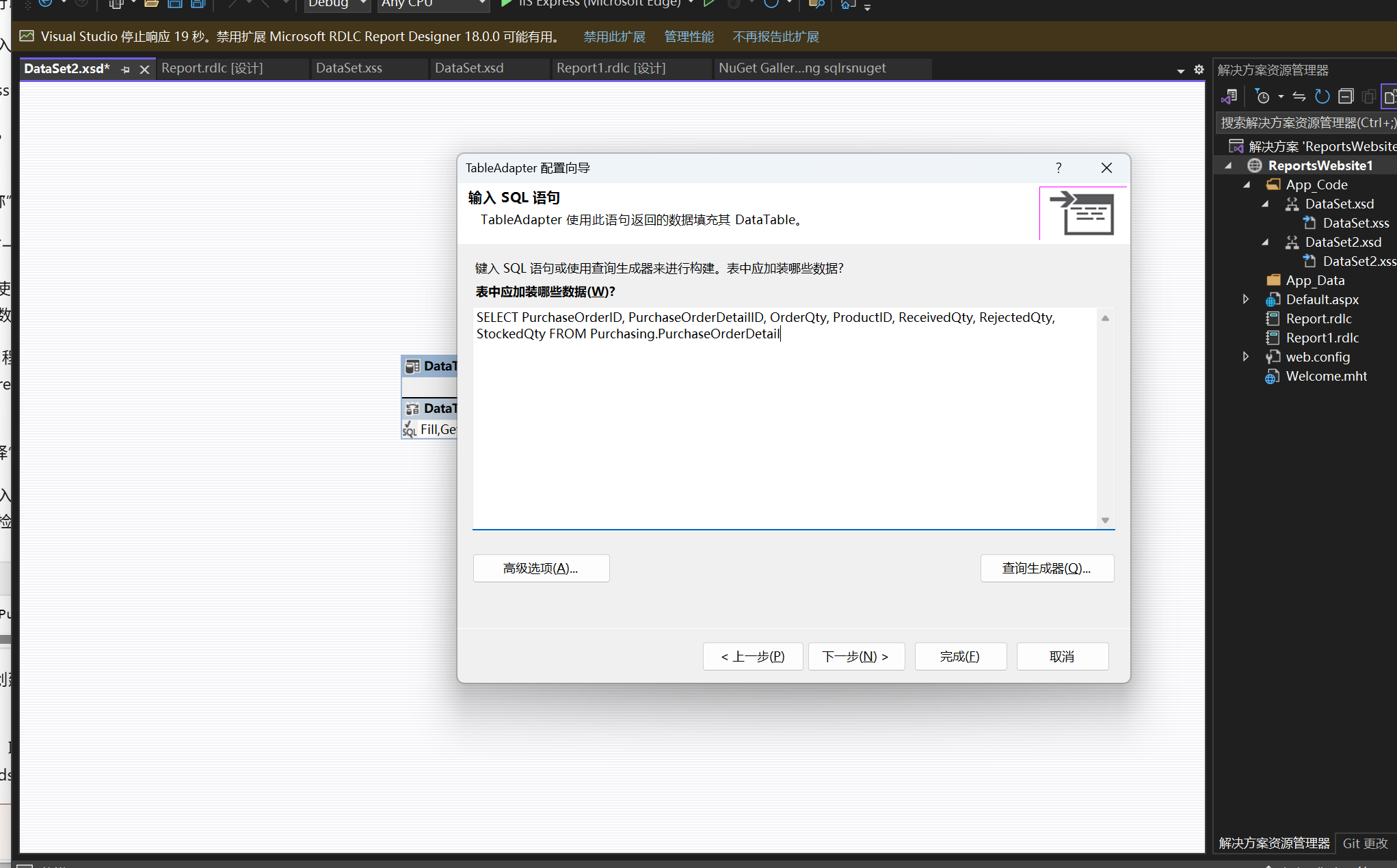


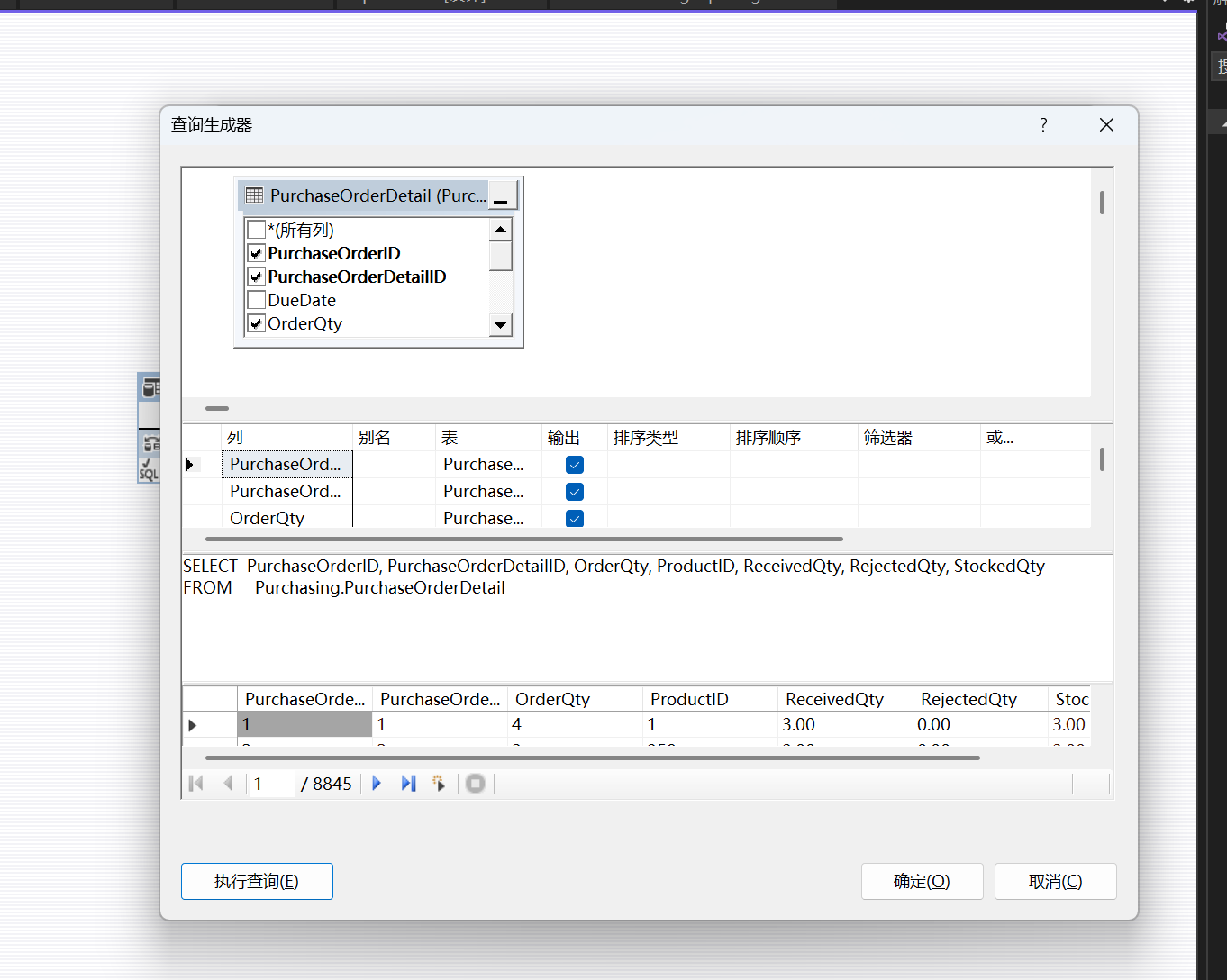
**（4）定义用于子报表的数据连接和数据表**

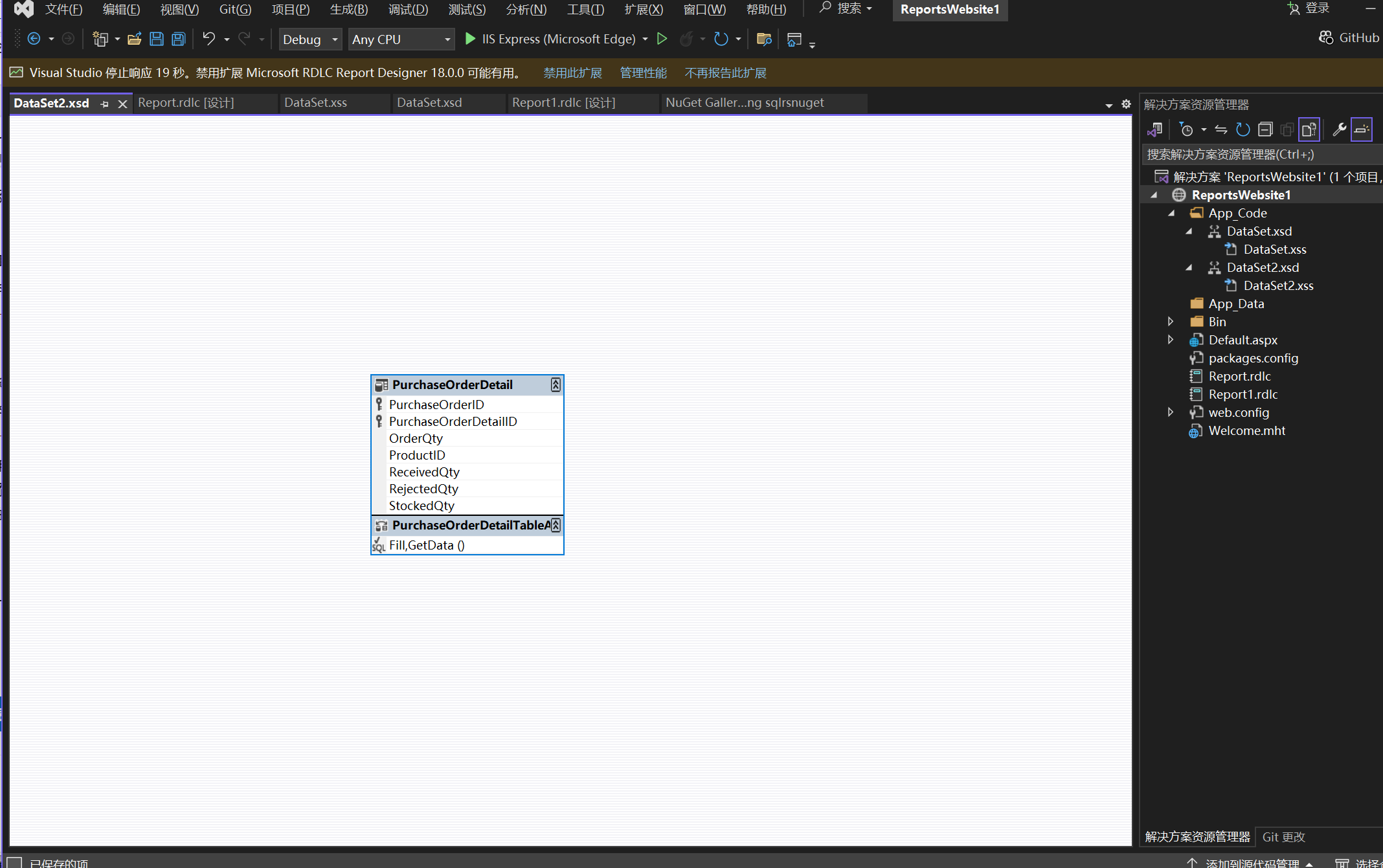
**通过添加 DataSet 定义数据连接和 DataTable（用于子报表）**

**在“添加新项”对话框中，选择“数据集”，然后选择“添加”。 出现提示后，应选择“是”，将该项添加到 App\_Code 文件夹**

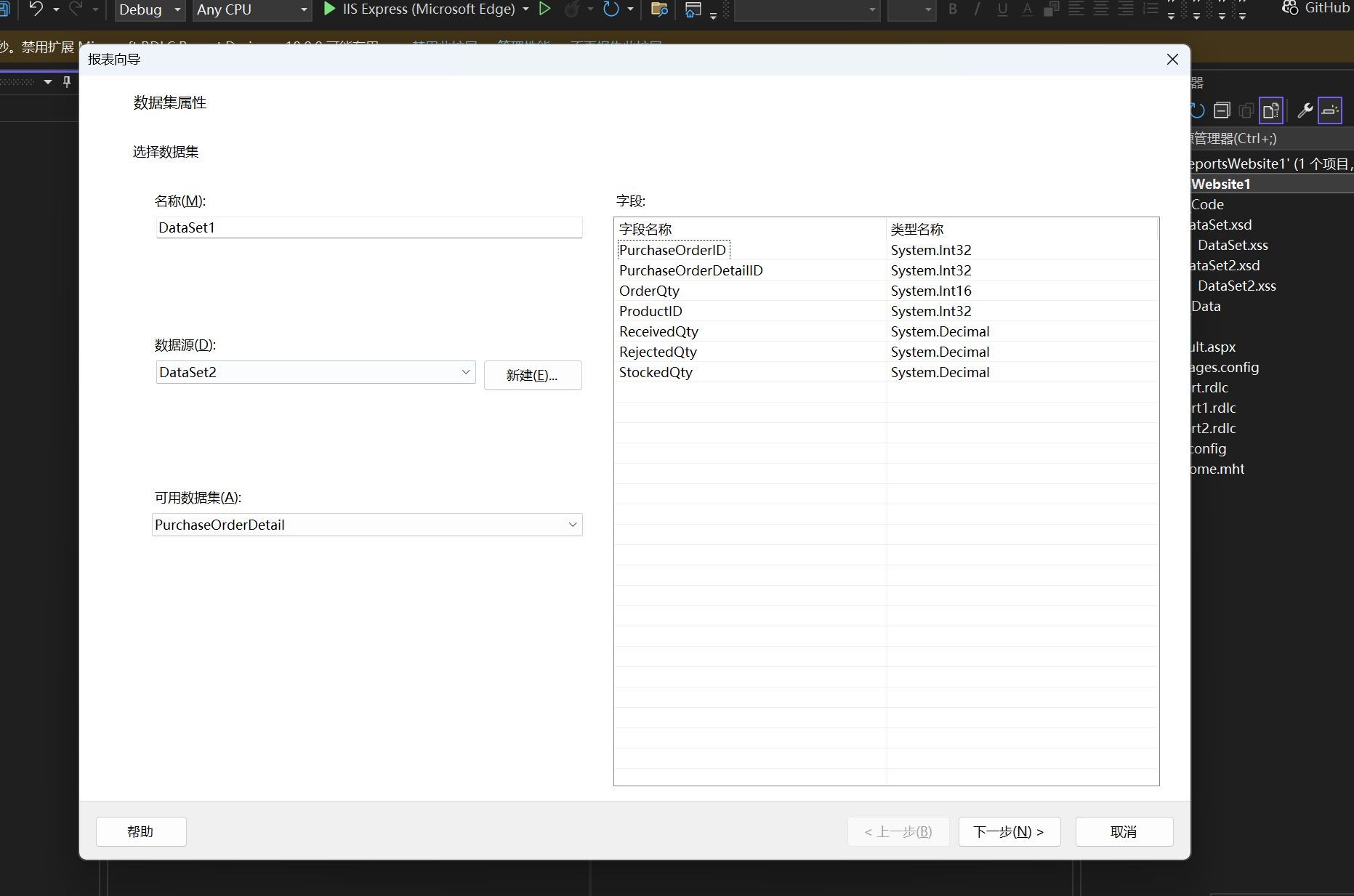
**从“工具箱”窗口中，将 TableAdapter 控件拖到设计图面上。 此操作将启动 TableAdapter 配置向导**

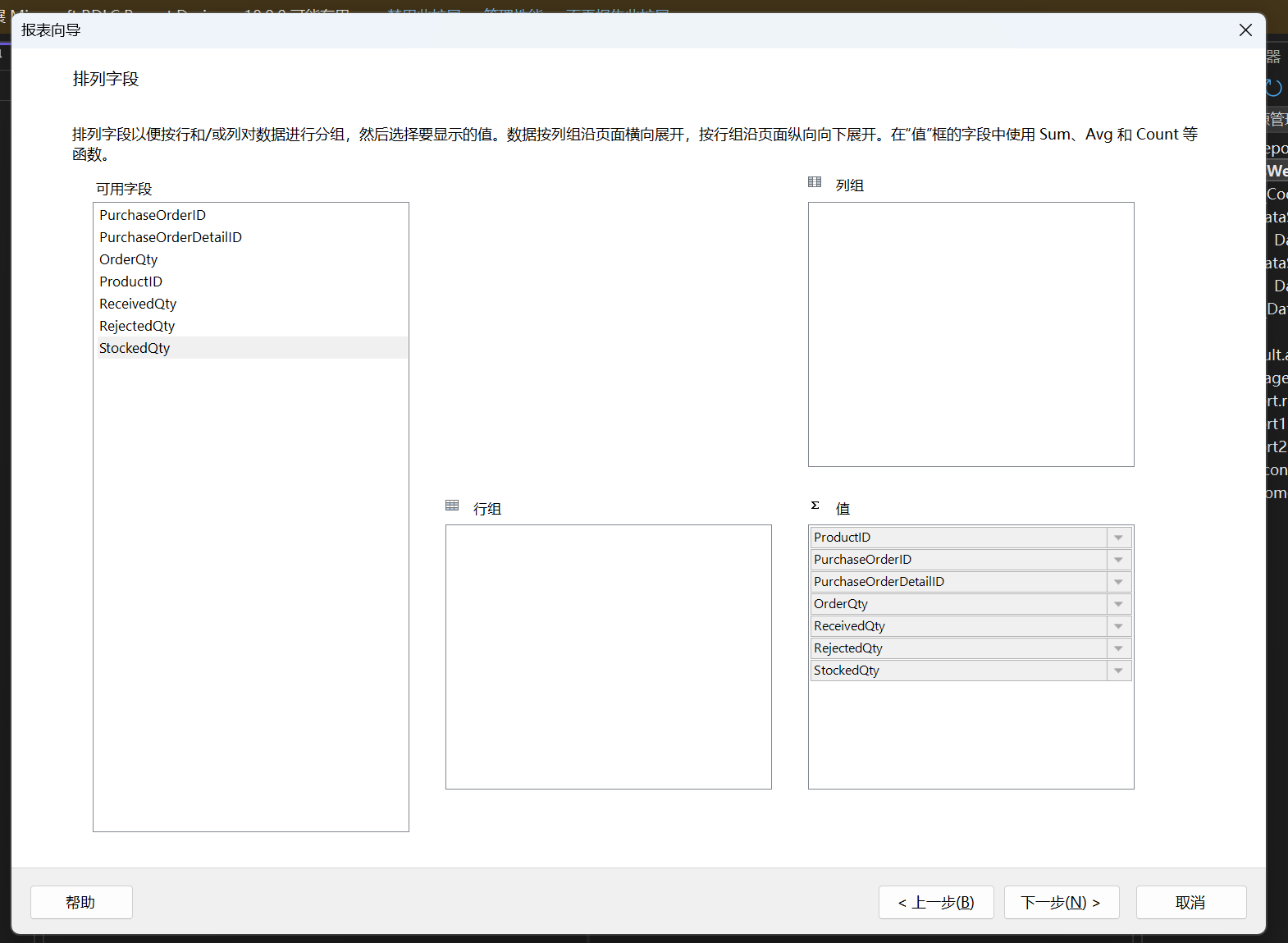


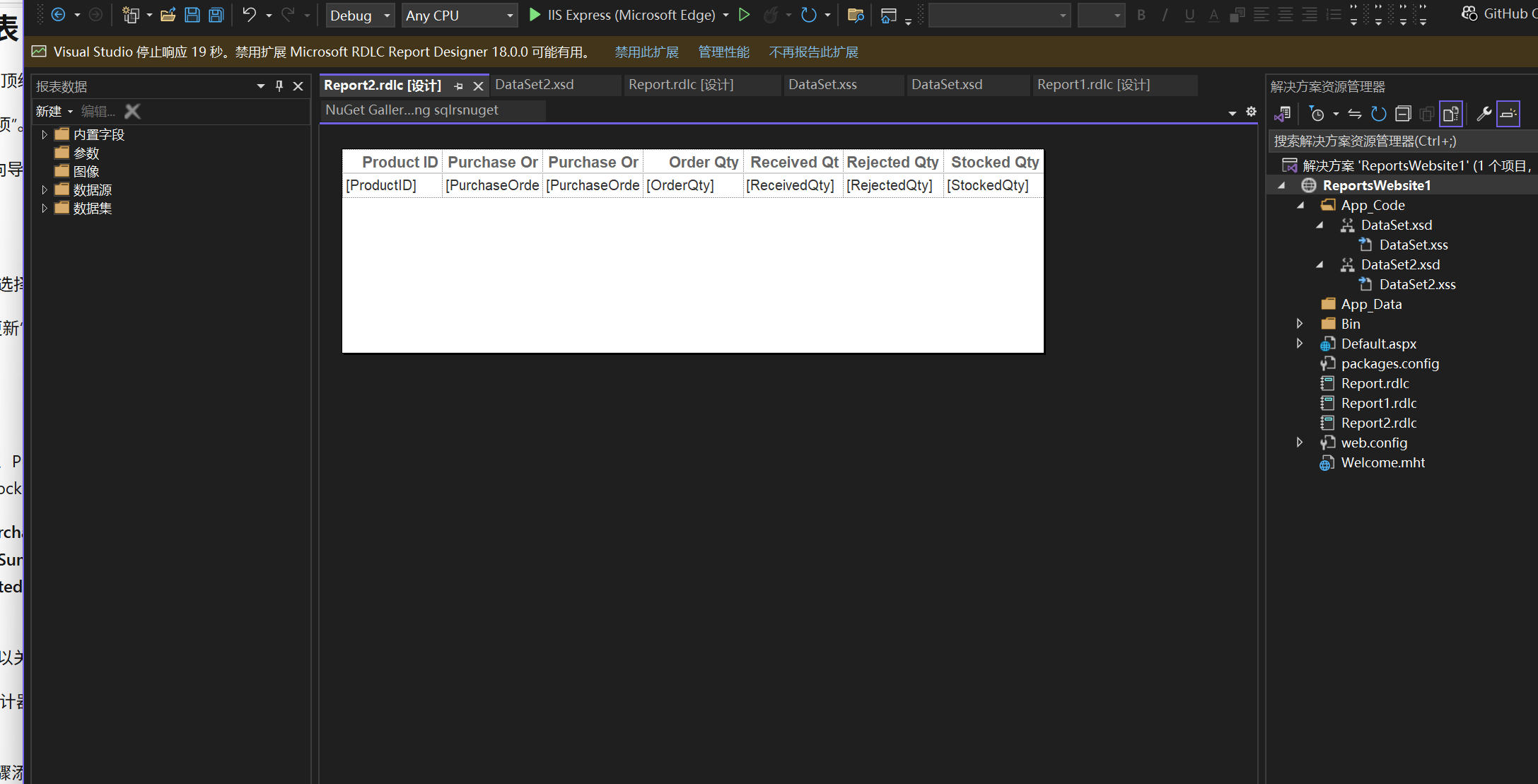




**（5）使用报表向导设计子报表**

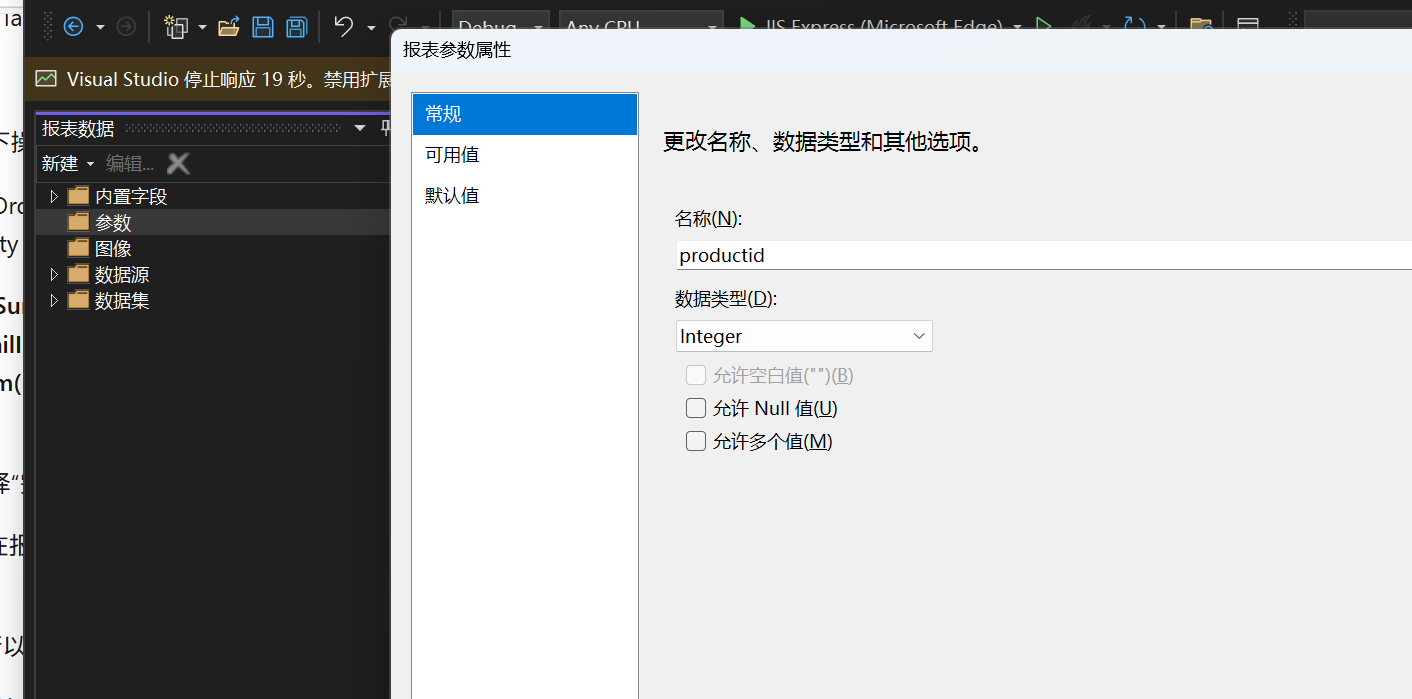




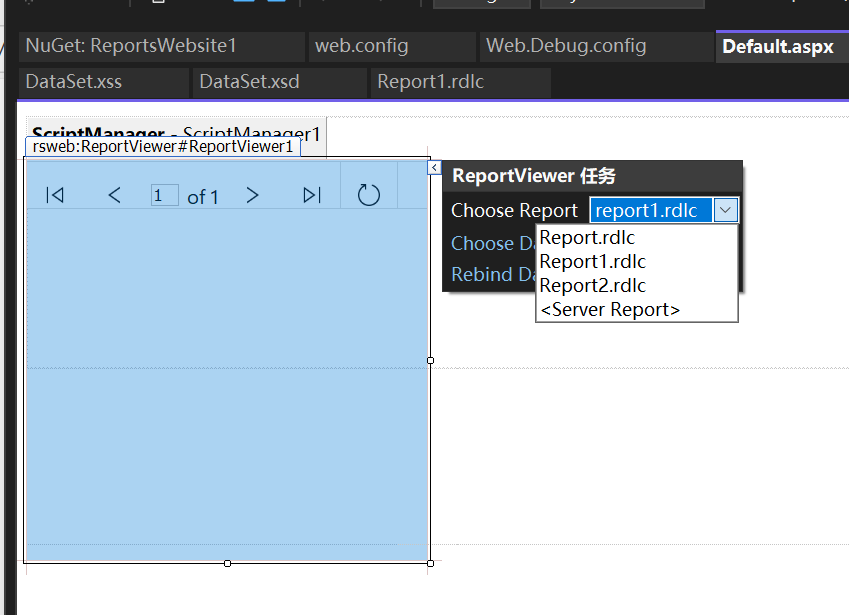


**打开 .rdlc 文件后，通过执行以下步骤添加参数：**

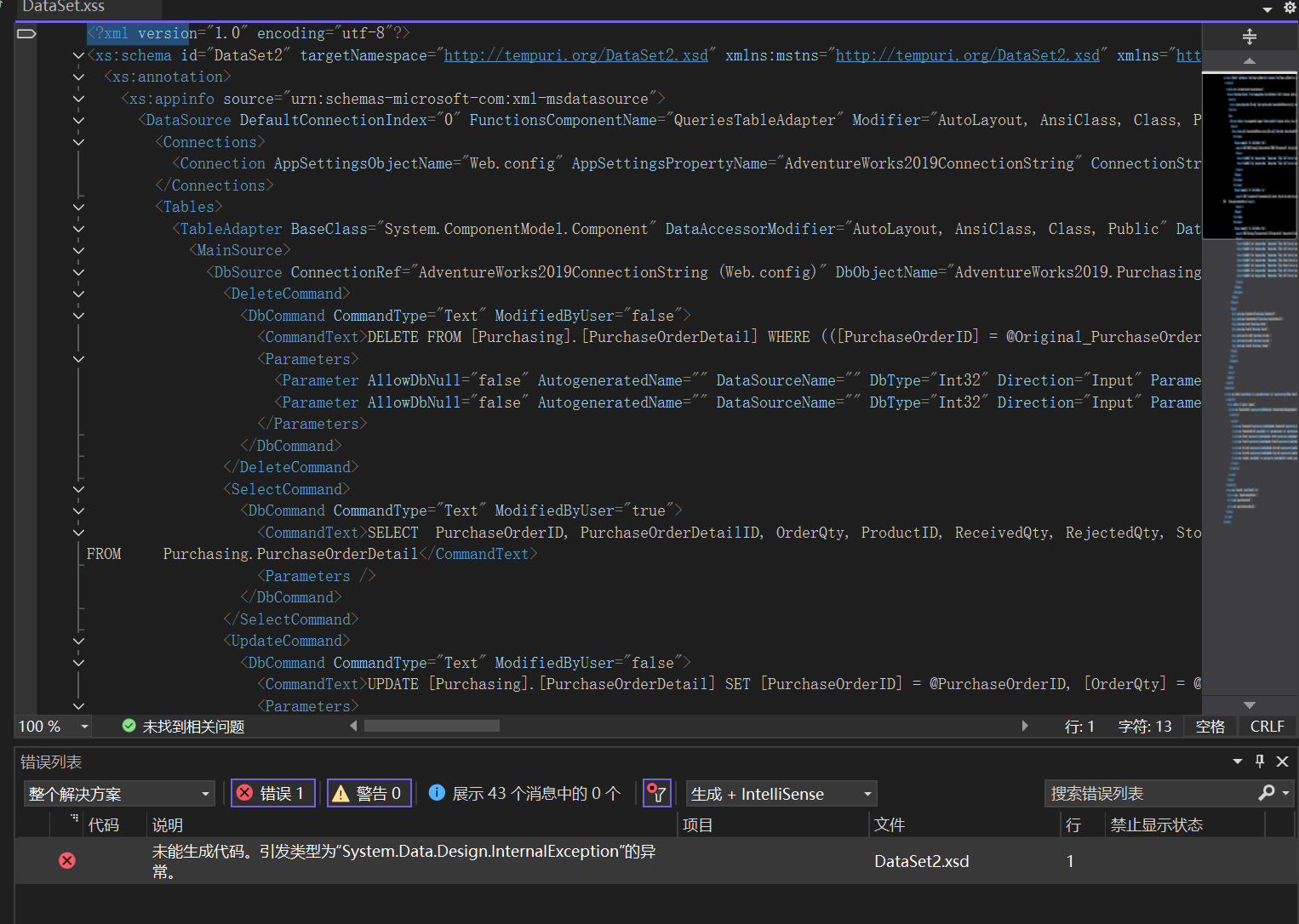
* 1. **在“报表数据”窗格中右键单击“参数”，然后选择“添加参数” 。**
  2. **在“名称”框中输入“productid” 。**
  3. **确认在“数据类型”列表框中选择了“整数” 。**
  4. **选择“确定”。**



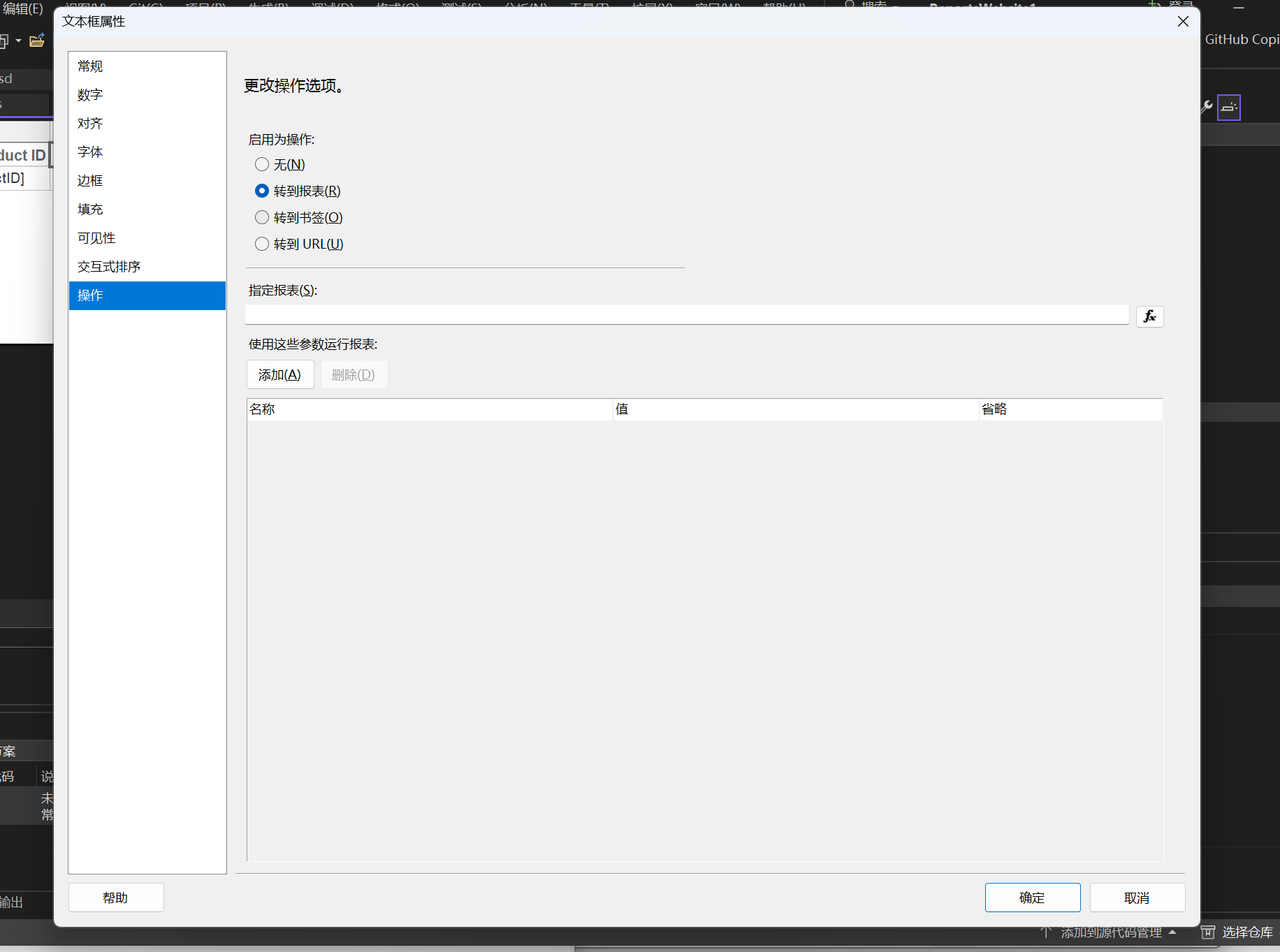
**（6）向应用程序添加 ReportViewer 控件**



**生成网站**

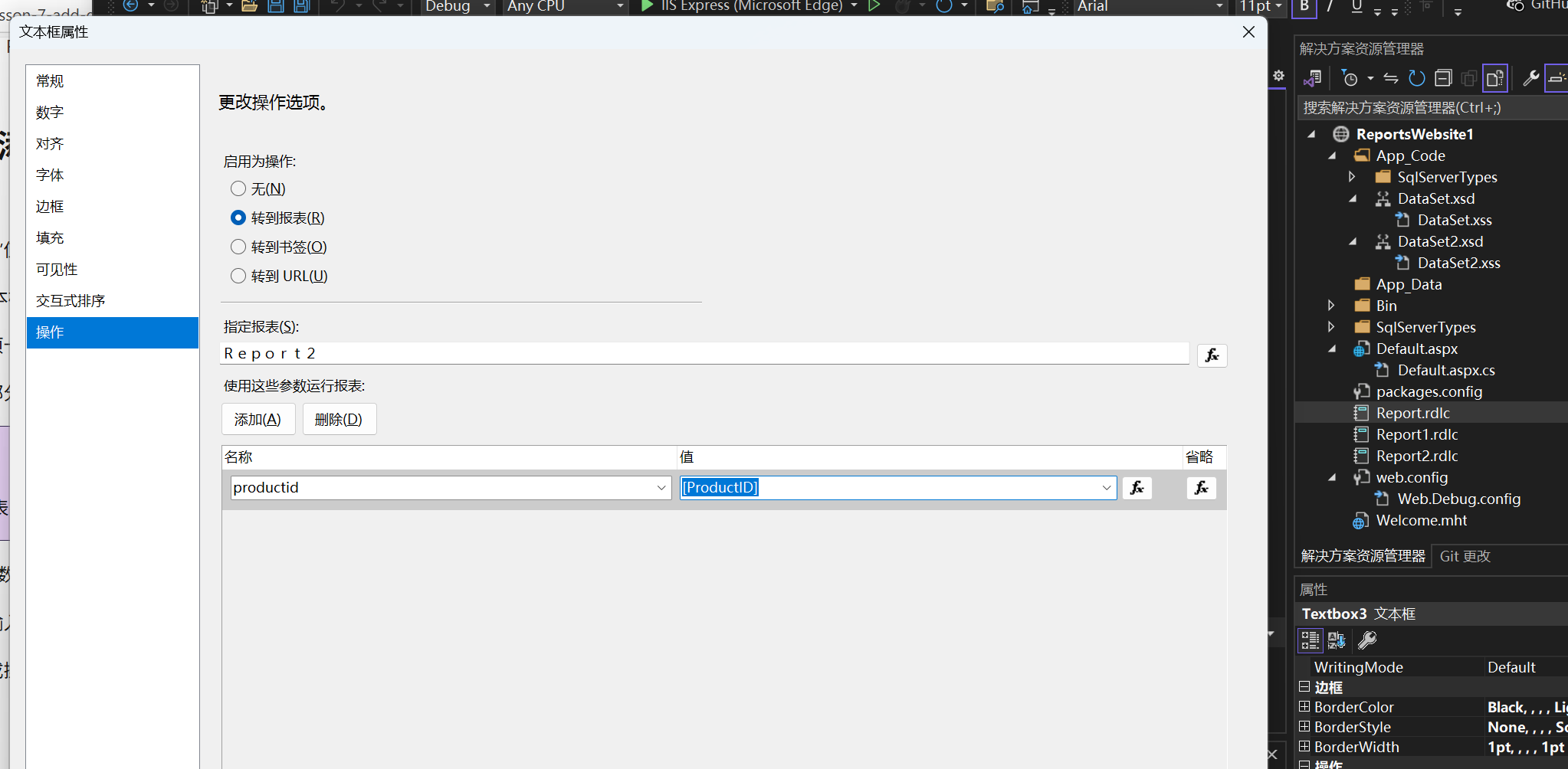


**（7）在父报表上添加钻取操作**



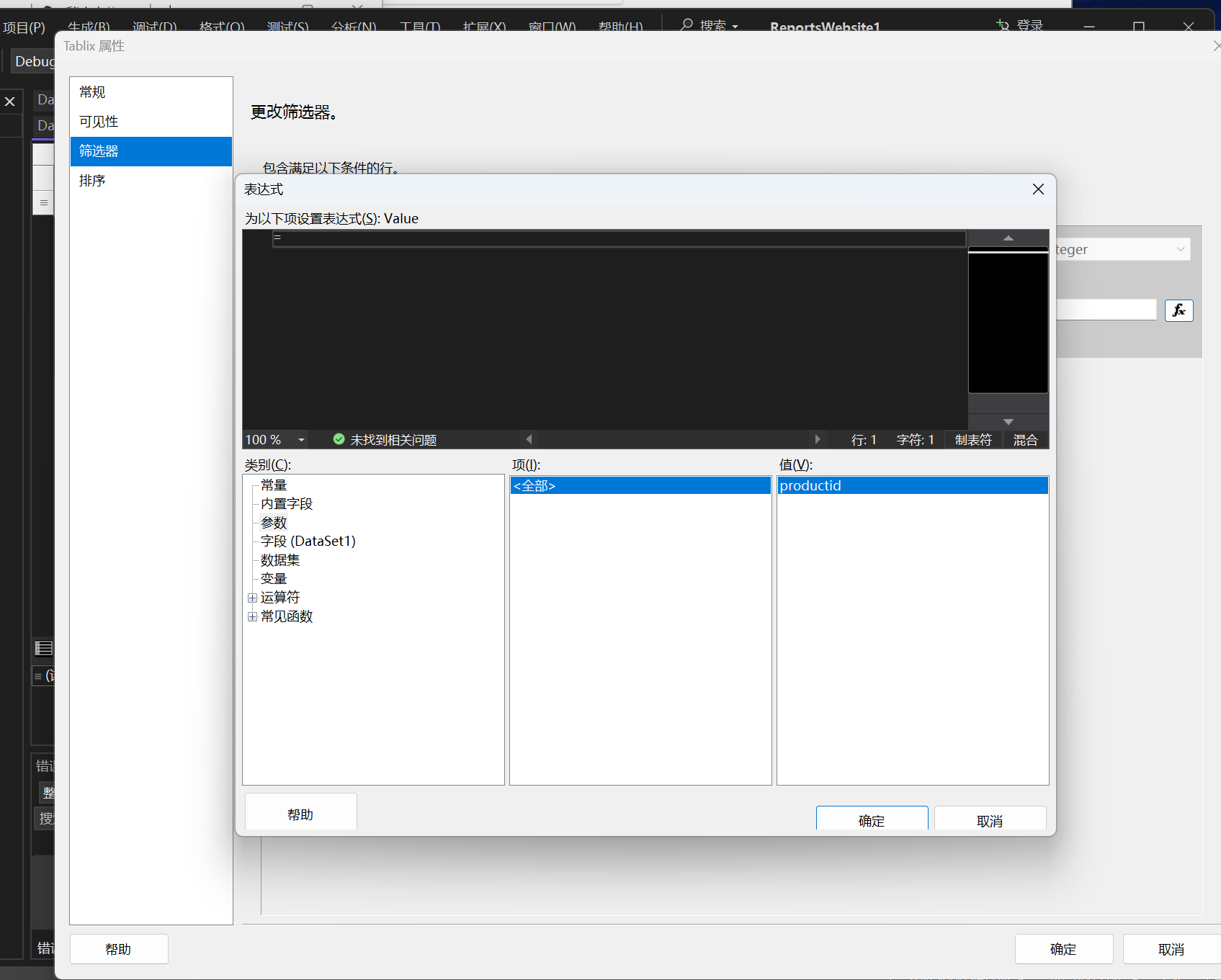
**在“使用这些参数运行报表”部分下，单击“添加” 。**

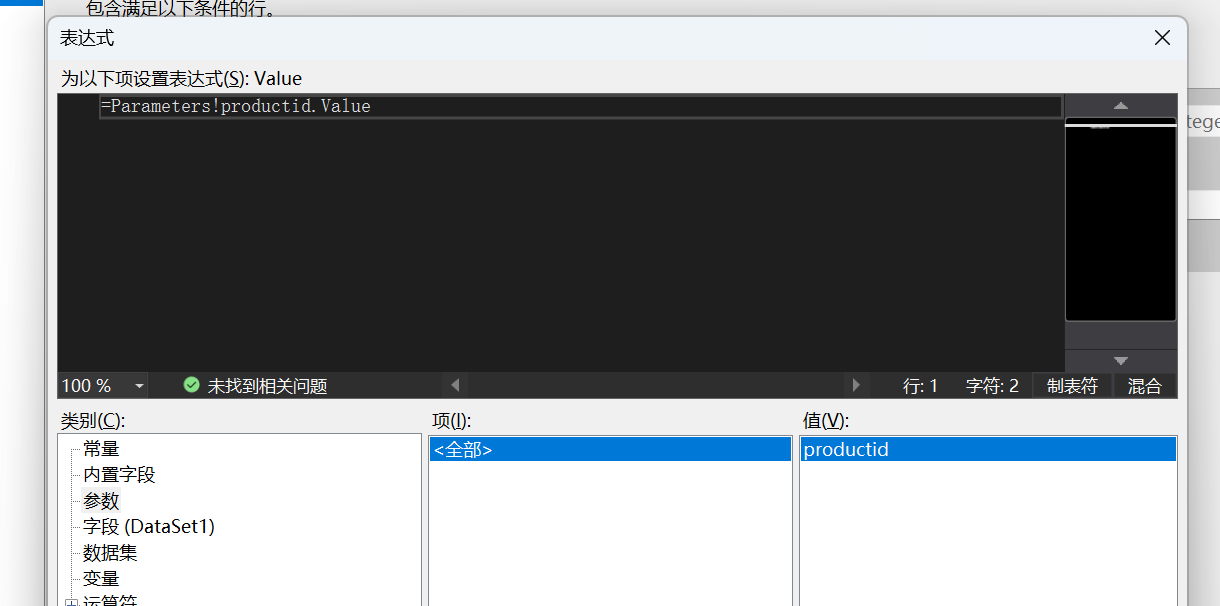
**在“名称”框中输入 productid，然后在“值”下拉列表中选择 ProductID**



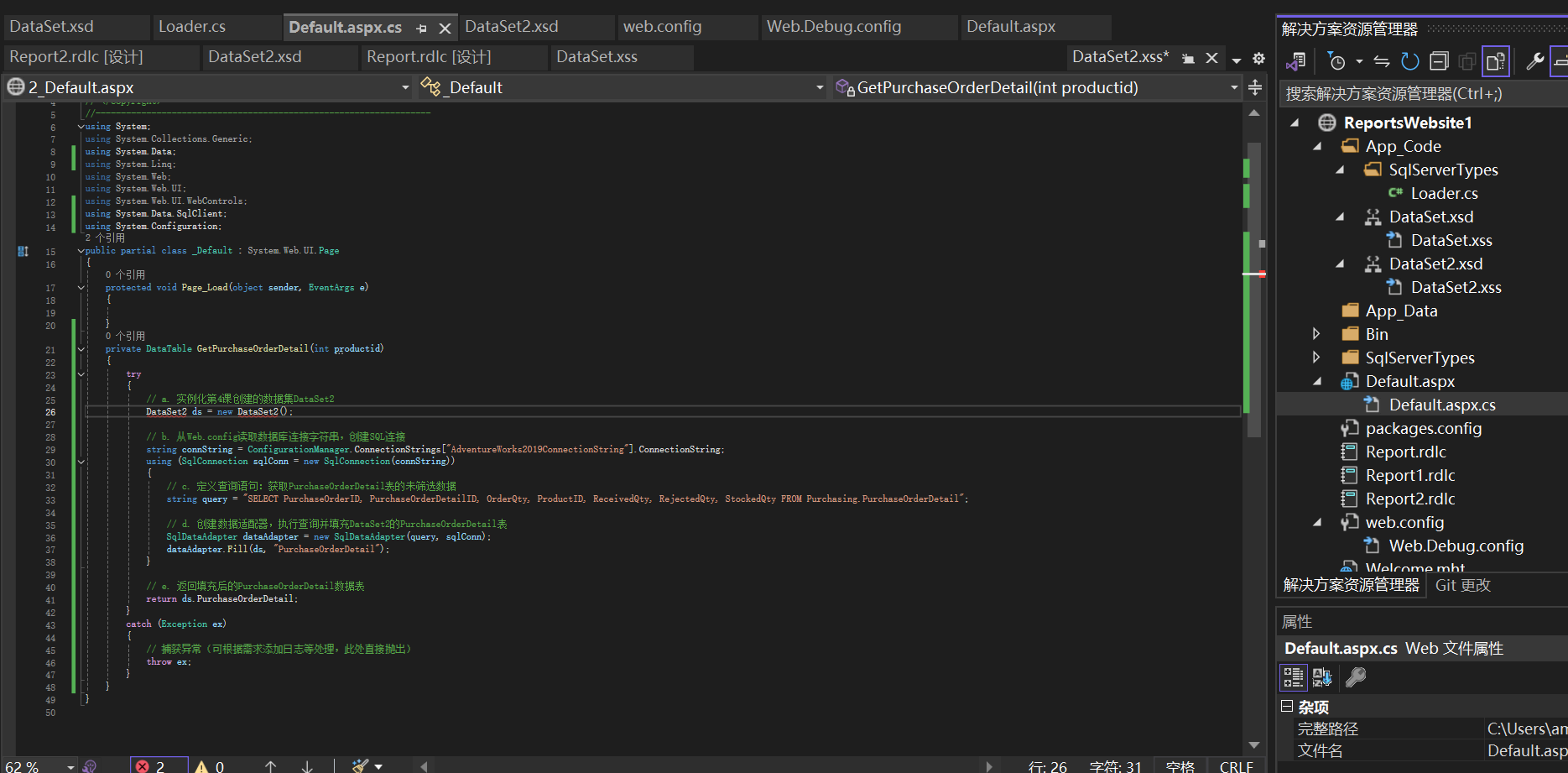
**（8）创建数据筛选器**

**向子报表中的 tablix 添加一个筛选表达式**

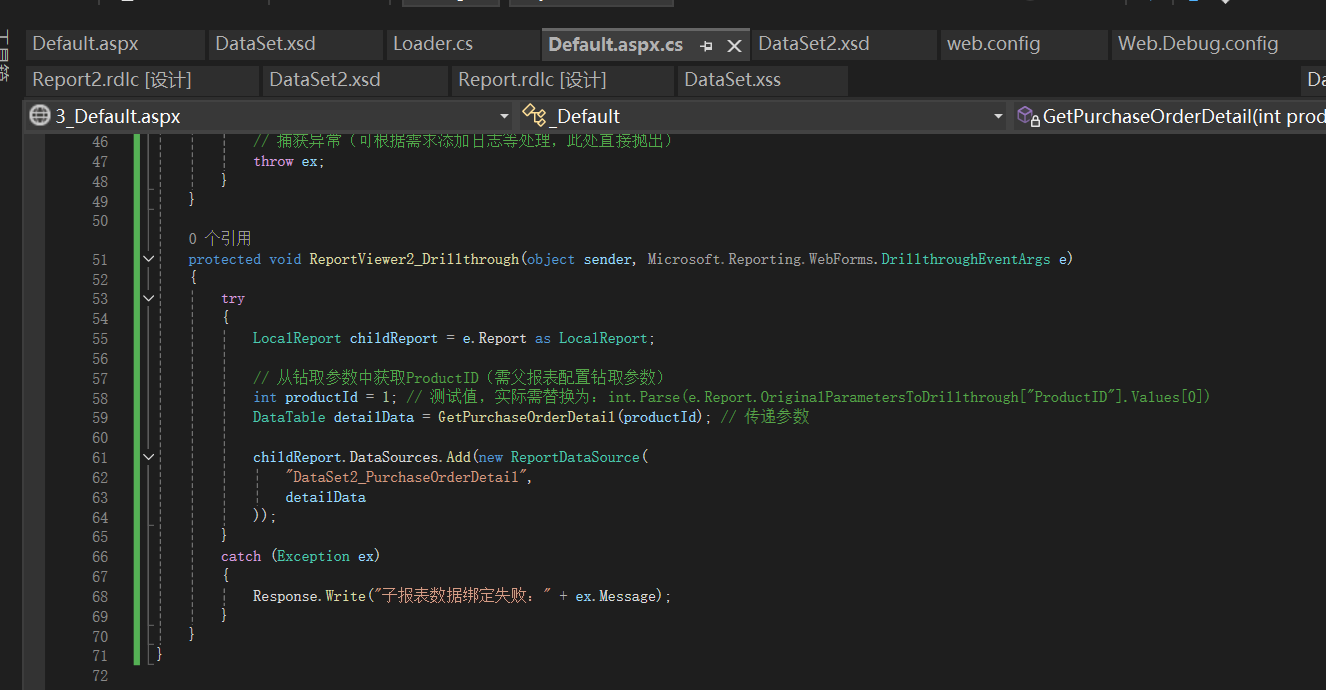




**创建一个函数，用于从 PurchaseOrdeDetail 表选择未筛选的数据**

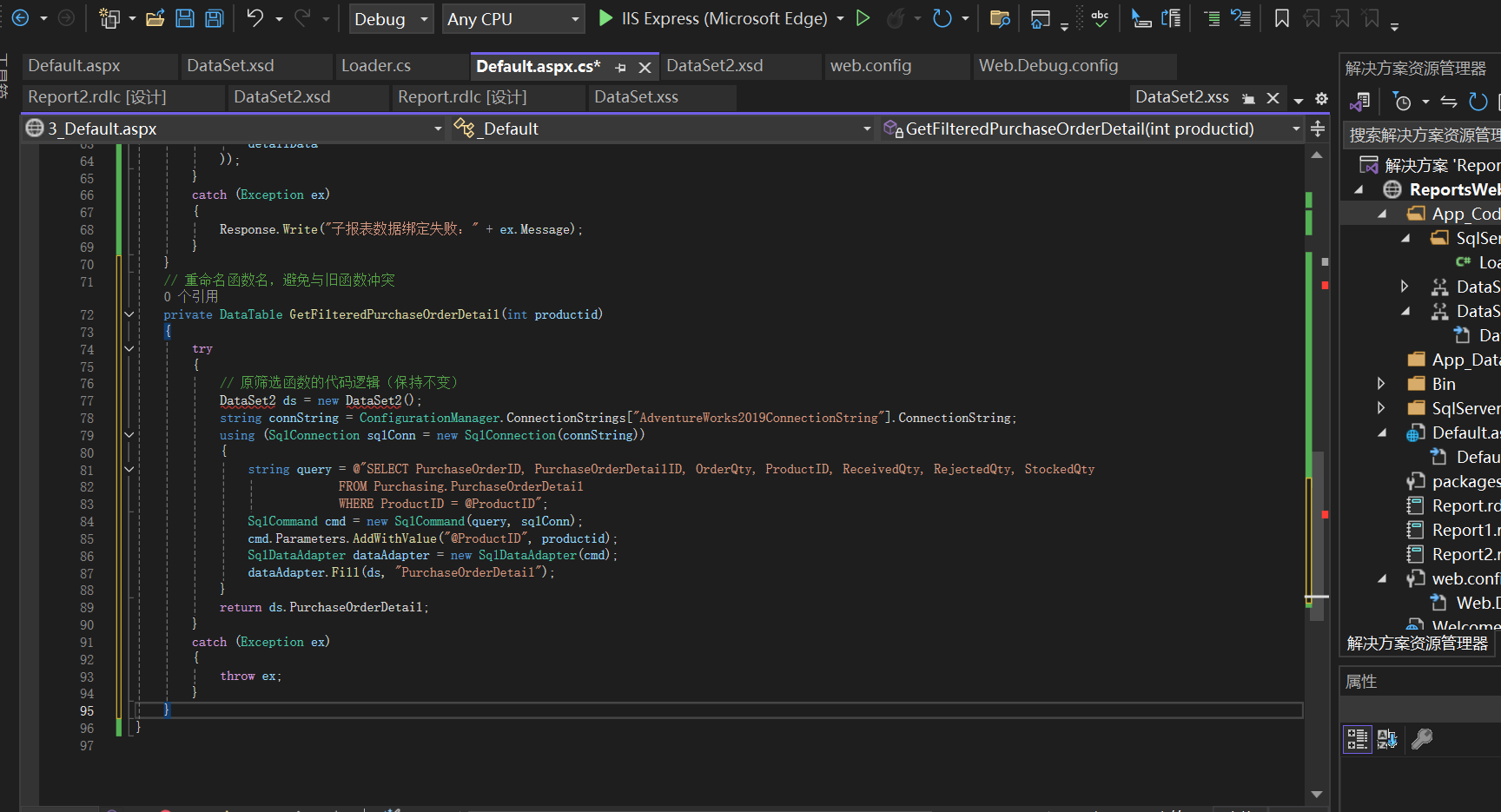


**添加一个事件处理程序，用于将 PurchaseOrderDetail DataTable 绑定到子报表**

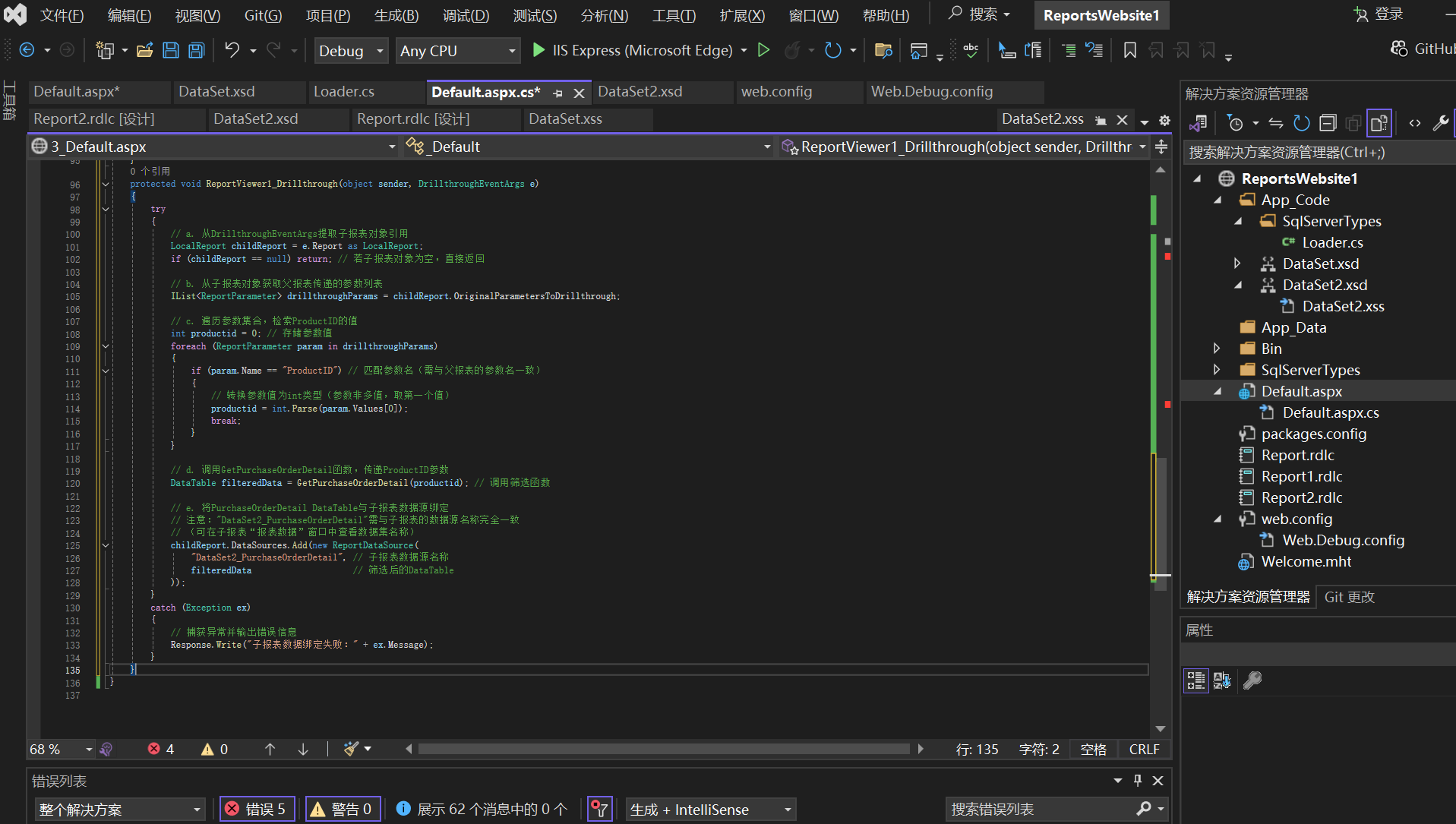


**查询筛选器**

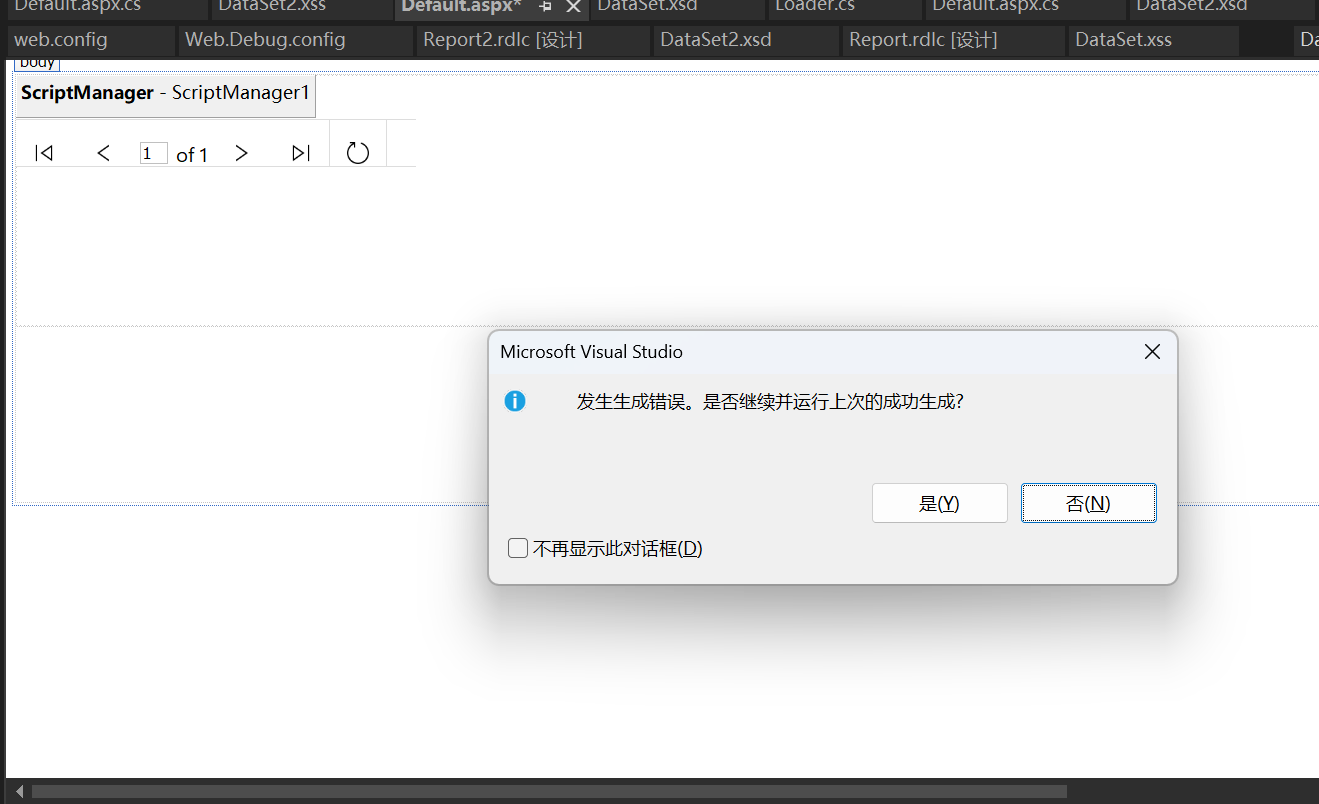
**创建一个函数，用于从 PurchaseOrderDetail 表选择经过筛选的数据**

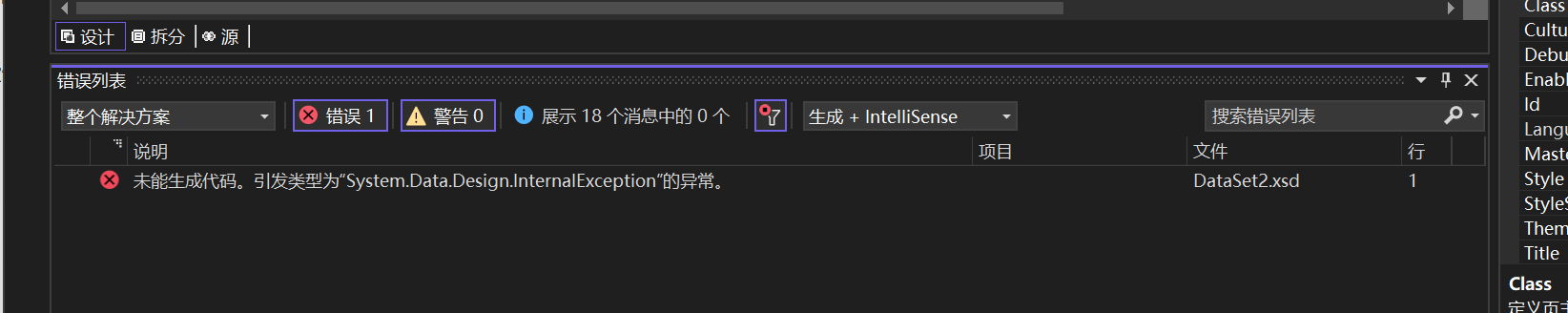


**添加一个事件处理程序，用于检索参数值并将 PurchaseOrdeDetail DataTable 绑定到子报表**



**生成并运行应用程序**





**不知道为什么一直生成失败，找不到问题，可能是跟VS2022不兼容？**

1. 实验总结(完成的工作、对实验的认识、遇到的问题及解决方法)

本次实验围绕 SQL Server Reporting Services（SSRS）报表设计器的使用展开，完成了从报表服务器配置、基本表报表创建、分组与总计设置，到数据驱动订阅和钻取报表开发的全过程。通过实践，我深入理解了 SSRS 作为企业级报表平台的核心功能：整合多源数据、实现可视化呈现、支持灵活分发与权限管理，并掌握了在 Visual Studio 环境下使用 ReportViewer 控件开发本地 RDLC 报表的关键技术。

在实验过程中，我成功完成了以下工作：

**配置了本地报表服务器**，设置了端口与数据库连接，为后续部署奠定基础；

**创建了基于 AdventureWorks 数据库的表报表**，实现了数据查询、表格布局、格式化（日期、货币、列宽）及按日期和订单分组汇总；

**实现了钻取报表功能**，通过父报表传递 ProductID 参数至子报表，动态展示采购明细，增强了报表的交互性；

**尝试配置数据驱动订阅**，理解了自动化报表分发的机制。

然而，在开发过程中也遇到了若干典型问题：

**RDLC 报表数据绑定异常**：初期因未为主报表提供数据源，出现 “A data source instance has not been supplied for the data source 'DataSet1'” 错误。通过在 Page\_Load 中显式绑定 ReportDataSource 并确保名称与 .rdlc 文件中的数据集一致得以解决。

**DataSet 生成失败**：使用强类型 DataSet（如 DataSet2）时，常因 VS2022 与旧版 .NET Framework 工具链兼容性问题导致代码生成失败。最终改用纯 DataTable 方式绕过此限制，提高了代码稳定性。

**SQL 连接异常**：在使用 Microsoft.Data.SqlClient 时，因缺少 SNI 本地依赖或连接字符串关键字不兼容，引发 DLL 加载失败或参数错误。通过统一使用 System.Data.SqlClient 或正确安装运行库予以修复。

**报表字段显示异常**：曾出现“拉两个 Product 字段才显示”的奇怪现象，推测是 ReportViewer 渲染引擎对单字段 Tablix 的处理存在边界情况，通过调整字段数量或刷新设计视图可规避。

通过本次实验，我不仅掌握了 SSRS 报表开发的基本流程，更深刻体会到企业级报表系统在数据整合、交互分析和自动化运维方面的价值。同时，调试过程锻炼了我在 .NET Web 应用中集成报表组件的实战能力，为后续从事 BI 开发或数据可视化工作打下了坚实基础。