第一部分 SQL 语言练习

一、实验目的

通过基于关系型网络数据库管理系统 SQL Server 的上机实验，使学生进一步了解关系数据模型及关系数据库管理系统的基本原理，标准的 SQL 语言的使用。

1、掌握数据库的基本概念，熟悉 SQL SERVER 2005，学会使用 SQL SERVER 客户端工具查询分析器

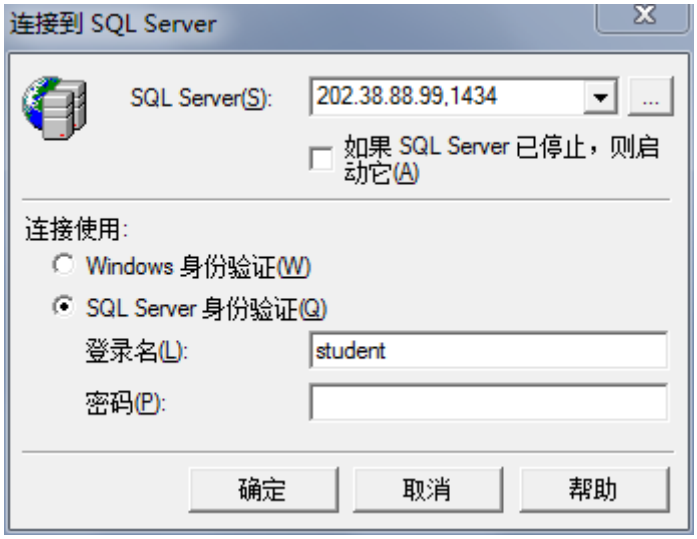
2、熟悉 SQL 语言的结构及主要命令。

3、能对已有数据库系统（DBMS 提供的一组样例表）进行各种检索。

二、实验原理

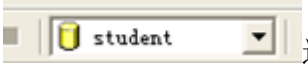
1、使用工具连接数据库

.下载 SQL Server 客户端工具：http://202.38.88.99/querytool.rar,解压后执行 isqlw.exe,【SQL SERVER(s)】输入 202.38.88.99,1433,【连接使用】选择“SQL server 身份验证”，登录名和密码都是 student。 此外，也可以使用实验室机器桌面上的 SQL Server Management Studio Express 登录。



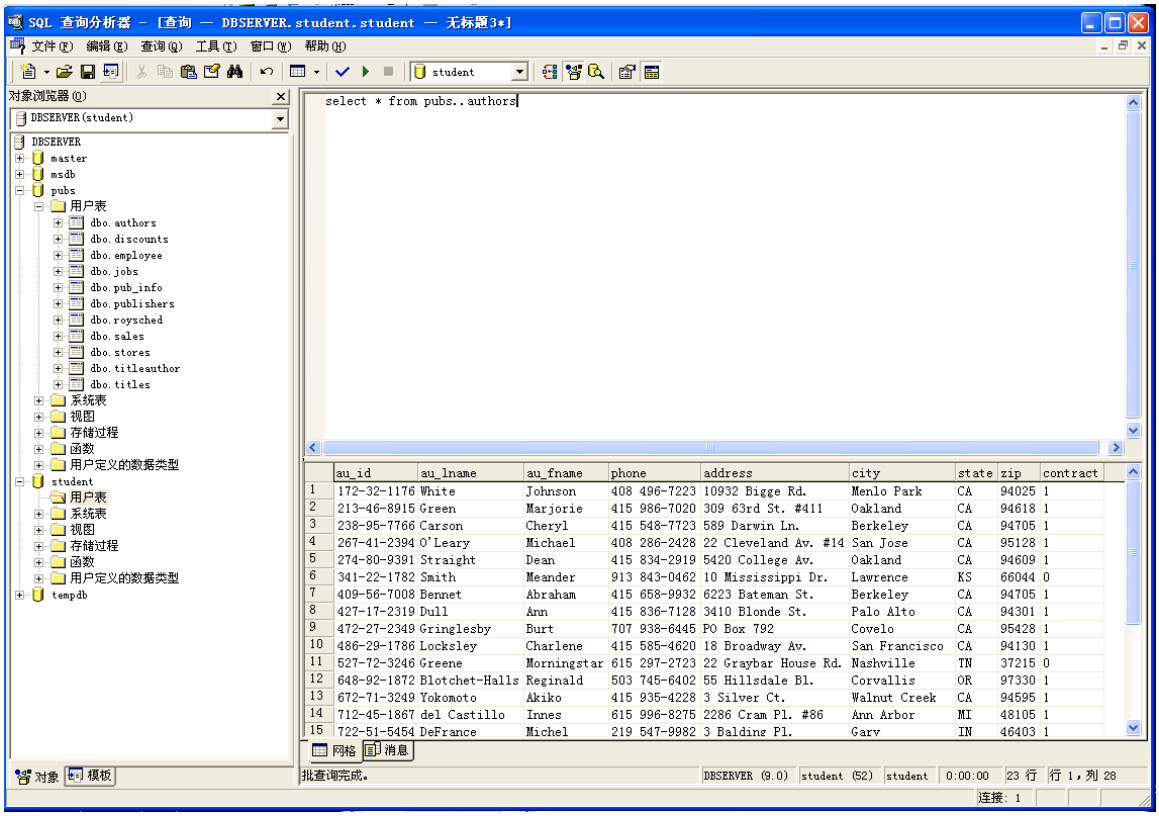
登录后显示如下界面，则可以执行 sql 语句了。

要注意，pubs 数据库是系统样例数据库，里面提供了 titles 等实验中用到的示例表，student 用户对其只有查询权，不可以修改。学生可以在 student 数据库中创建自己的表，创建表时请按照题目中要求的命名规则来命名。

在操作数据库是要注意当前数据库是 pubs 还是 student，可以在界面面的 combox 选择窗口中 

选择，也可以使用命令 use pubs 更改当

前数据库。



2、SQL Server 常用的存储过程说明

存储过程名称	存储过程功能
Sp_tables	查看数据库中已有的数据表
Sp_table_privileges 表名	查看特定数据表的权限信息
Sp_columns 表名	查看特定表的字段定义
Sp_column_privileges 表名	查看特定表的字段的权限信息
Sp_help	查看当前数据库的对象简要信息
Sp_spaceused	查看当前数据库的空间使用情况

3、样本数据库介绍

我们的实验中可以使用的数据库有两个：pubs 和 student。Pubs 是系统数据库，提供了 10 来张已经建立的表供大家查询，但是不能进行查询以外的其他操作；student 数据库中可以进行一切操作。有关 pubs 数据库中各个 table 的定义和关系请参考：http://202.38.88.99/pubs.pdf

第二部分 用 Delphi 实现数据库应用开发

一、实验目的

通过使用 Delphi 中 BDE 连接 SQL 数据库，掌握 Delphi 环境中数据库操作的最基本的方法和 Delphi 下数据库应用程序的开发方法。

二、实验原理

1、Delphi 中数据库引擎 BDE 介绍

Delphi 对数据库的支持十分丰富，Delphi 的数据库接口是数据库引擎 BDE(Borland Database Engineer)，它是应用程序存取数据的中介层（Middleware Layer）。在 Delphi 下开发的所有数据库应用程序，都是通过 Delphi 的数据库引擎 BDE 来对数据进行操作的。BDE 中包含了内建的驱动程序来处理 Paradox 及 dBase 的表格与文本文件。也可以通过外部的驱动程序（add-on driver）来处理 Oracle、SyBase、IntrBase、Informix、MSSQL 格式的数据。BDE 也包含了允许应用程序使用 ODBC 驱动程序来存取数据的 ODBC socket。

BDE 具有以下突出的优点：

(1)不同的 BDE 间的数据可以共享而不必担心有任何的冲突发生，因为所有的数据存取都由数据库引擎来处理。

(2)使用者可以通过 BDE 对数据库的记录进行双向的查询而不必考虑服务器是否具有这样的功能。

(3)允许使用者在不同的平台上对不同数据库的数据做联集（join）。

(4)增加了数据在不同平台上的可移植性。

数据库引擎 BDE 的设置是通过对数据库配置文件 IDAPI.CFG 来完成的，在 Delphi 安装完成以后，可以通过其主菜单下的 DataBase 选项的 Explore 子选项完成。BDE 配置的基本项目是如下几项：

Drivers:主要设定一些有关数据库驱动程序相关信息的内容，包含 ODBC 驱动程序、MSSQL 驱动程序、ORACLE 驱动程序、SYSBASE 驱动程序等数据库驱动程序。

Aliases:这是 IDAPI.CFG 中最重要的设置内容，它负责记录数据库文件别名的关系，如此做法可以保证数据库系统相关文件改变时，可以不去变动程序代码，只需要更改 Aliases 的设定；使得程序更具有可移植性和扩充性，数据的独立性也由此得到保证。

System:显示 BDE 系统的版本数据；局部的文件共享信息；最大、最小缓冲区；语言驱动程序；系统标志，最低内存使用限制，ODBC 的 Alias；处理 SQL 的查询等一系列设定。

Date:日期的相关信息，如日期的格式、与字符的转换等。

Time:系统时间的格式、与字符串的转换关系。

Number:负责数字和字符串的转换。

ALIASES 是 Delphi 中应用程序与数据库接口的连接点。在设置数据库别名时，还有许

多选项的设置，因不同的数据库驱动文件而有不同的设置内容。当数据库采用标准的数据库驱动程序（即数据库文件是 XBASE 格式，*.DBF）时，只需要指定文件存放的路径。而为其他的数据库驱动程序时，要根据不同的驱动来设置用户名、服务器名、日期格式等内容。

2、Delphi 中数据库控制组件

数据库控制组件是数据库程序的核心。共有两类：数据库存取控制组件、数据库显示控制组件。在 Delphi 中，引入了一个“数据来源”的概念。其作用是在数据库的具体数据与程序中显示的数据之间做中介，即在数据库存取控制组件和数据库显示控制组件间做媒介。借助数据来源可以在应用程序执行时将数据存取组件与数据显示组件独立分开，动态操作、切换这两种组件。下面以实例分别对这两种组件中的常用组件进行介绍。

(1)数据库存取控制组件：

负责连接数据库本身，不包含数据的显示与输入。

TDatabase 组件:

```
Database1.AliasName:= 'mydbs';
    {指定数据库 Database1 的别名是 mydbs，通常是 ODBC 中配置的 DSN 名称 }
Database1.DatabaseName:= 'my_database';
        {用于在程序中被其他数据库集等控件连接的数据库逻辑名 }
Database1.connected :=True;
  {连接数据库，在 Tdatabase 空间中双击，还可以保存帐号密码和默认数据库等信息，
   当保存密码后可以将 Loginpromp 置为 false，不再提示输入密码 t }
```

TTable 组件:

```
Table1.DatabaseName:='my_database';
    { DatabaseName 指向前述 Tdatabase 组件的 DatabaseName 属性即 my_ database，
  TTable 该属性也可以不指向 TDatabase，而直接指向 ODBC 的 DSN，但无法保存密码 }
Table1.TableName:='titles';
    { TableName 对应要访问的数据库中的表名}
Table1.Active:=True;
    {打开表 Table1，执行该操作，数据将被取回到本地机器内存，如果和显示控件连接
     正确，数据就可以被展示出来。}
```

TQuery 组件:

```
Query1.DatabaseName:= 'my_database';
     {TQuerye 组件和 TTable 组件类似，都属于数据集控件，不同的是后者只能访问单表，
     而前者可以使用任何灵活的 SQL 语句。 }
Query1.SQL.Add('select * from    sales');
     {查询对应的 SQL 语句是 select * from sales }
Query1.open;   或者  Query1.execsql;
  {执行查询 Query1,如果有数据集返回使用前者，如 select，否则用前者，如 update}
```

(2)数据库显示控制组件：

提供建立用户界面的基本窗口类型控制组件，让使用者可以输入编辑、显示数据。

TDataSource 组件:

DataSource1.DataSet:=Query1;

{数据来源组件 DataSource1 是数据访问组件与显示组件之间的桥梁，其 dataset 属性对应于前述的 TQuery 或 TTable 组件 }

TDBGrid 组件:

DBGrid1.DataSource:= DataSource1;

{数据库内容显示组件 DBGrid1 的数据来源是前述的 TDataSource 组件}

TDBText 组件:

DBText1.DataSource:= DataSource1;

{数据库字段显示组件 DBText1 的数据来源是 DataSource1}

DBText1.DataField:= 'name';

{数据库字段显示组件 DBText1 对应的表的字段名称是 name }

3、在 Delphi 中动态地使用 SQL 查询语句

在一般的数据库管理系统中，通常都需要应用 SQL 查询语句来提高程序的动态特性。下面介绍如何在 Delphi 中实现这种功能。

在 Delphi 中，使用 SQL 查询语句的途径是：在窗体中置入 TQuery 构件，设置其 SQL 属性的内容值，此内容为一个字符串数组，数组的每个值对应一行 SQL 查询语句。可以在程序设计过程中事先指定，也可以在程序运行中重新赋值，即可以实现动态地改变程序中的查询语句。假定程序的窗体中有一个名为 Query1 的 TQuery 构件，在程序运行过程中需要改变它的 SQL 查询语句内容，则可以引用以下程序行。

```
Query1.close;       {先关闭以前查询的连接}
Query1.SQL.Clear;       {清除以前的查询语句}
Query1.SQL.Add('select * from mytable');
            {增加新的查询语句内容为 select * from mytable}
Query1.open;
        {建立新的查询语句的数据库连接,如是 update 等没有返回数据的语句则使用
        Query1.execsql }
```

在 Delphi 中，要灵活地使用 SQL 查询语句，还需要使用 TQuery 构件的 Params 特性，即在查询语句中使用参数。Delphi 中的 SQL 语句如需参数，则在参数名称前使用一个冒号':'来作为标识。例如：'select * from mytable where id_no=:p'一句，其中 p 为参数，可以在程序运行过程进行在赋值。在程序运行期动态地创建带参数的 SQL 查询语句较为复杂，其过程如下：

```
Query1.close;      {先关闭以前查询的连接}
Query1.SQL.Clear;    {清除以前的查询语句}
Query1.SQL.Add('select * from title1 where NO_GLOBE=:p');
                {增加新的查询语句内容为 select * from title1 where NO_GLOBE=:p}
                {其中 p 为待传入的参数}
Query1.ParamByName('p').asstring:= '1';   {给名为 p 参数赋值为'1'}
Query1.open;    {建立新的查询语句的数据库连接}
```

以上两个示例中，SQL 查询语句都是 SELECT 语句，而当 SQL 语句是 UPDATE 或 INSERT、DELETE 时，其中的 Query1.Open 需要改为 Query1.ExecSQL。

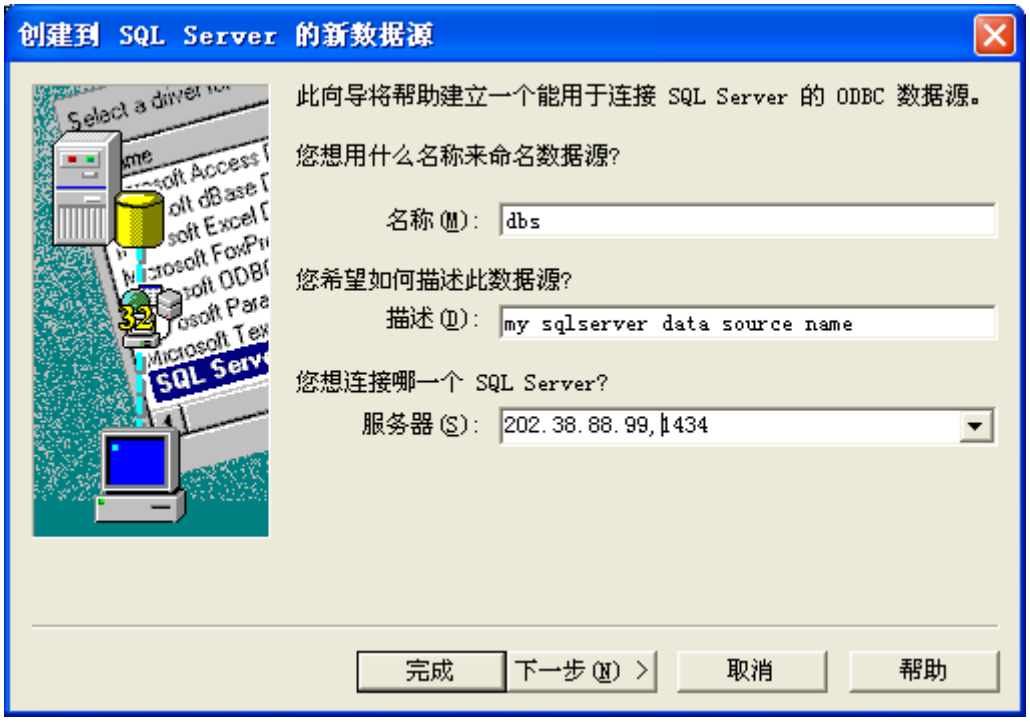
三、实验内容

1、ODBC 设置

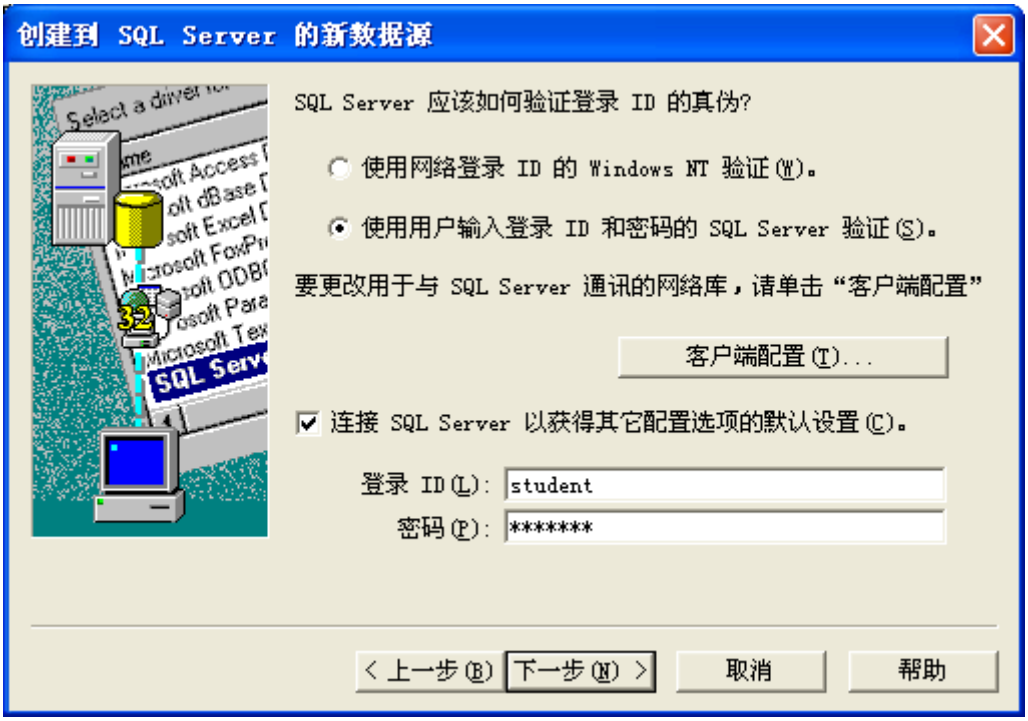
在 window 中打开控制面板->性能和维护->管理工具->数据源(ODBC)，点击添加，选择 SQL Server 数据源驱动，如图。

如果是其他数据库系统，如 mysql 或 oracle 数据库，都需要安装相应的客户端驱动程序。

双击 SQLServer，如下图所示：



输入名称（将来在 database 组件中对应的 AliasName），描述可以不写，服务器填写网络 IP 和端口号，如果 SQL Server 的端口号是默认的 1433,则可以不加端口号，要注意的是端口号和 ip 之间是逗号分隔而不是冒号。单击下一步，出现下图：



修改登录验证方式为登录 ID 和密码，并修改下方相应的 ID 和密码。一直点击下一步，直到出现下面的界面：

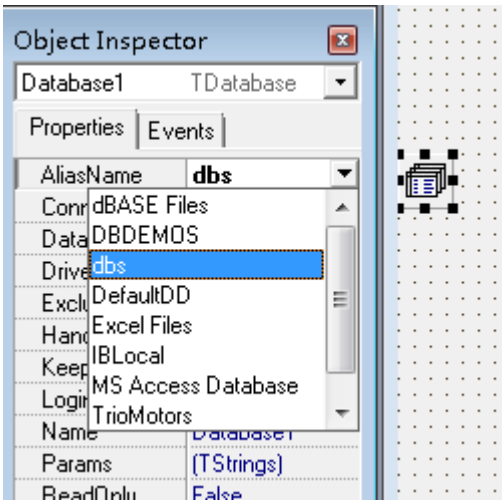
点击测试数据源后应该显示测试成功。否则检查网络是否正常、数据库服务器是否正常启动或之前输入的参数是否正确。

点击确定回到前一窗口，再点击确定，至此，ODBC 数据源建立成功。

2、在 Delphi 中创建一个数据库程序

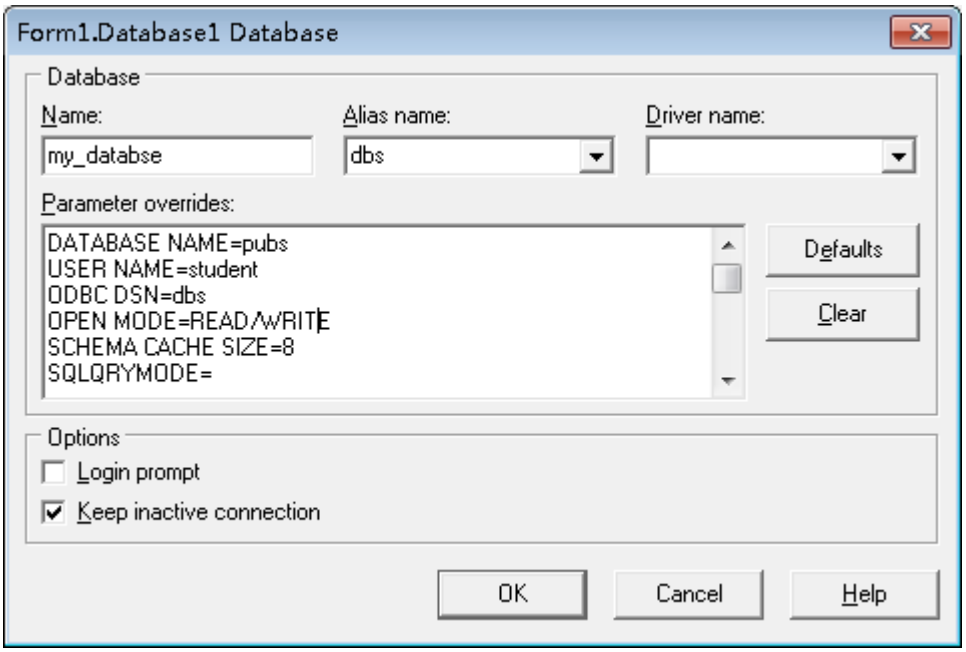
选择菜单项“File”中的“New Application”来创建一个新的应用程序。

1）在窗体上放置一个 BDE 组中的 Tdatabase 控件，选择其 AliasName 为之前配置的



ODBC 数据源“dbs”。

双击 Tdatabase 控件，点击 default，会出现下面的默认数据源参数。



填写 name 为“my_database”，填写 parameter overrides 中的

DATABASE NAME=student（或者 pubs）

USER NAME=student

PASSWORD=student

并将“Login Prompt”的勾去掉。

2）再从 BDE 控件组中选择一个 Table 控件，设置

DatabaseName=my_database，

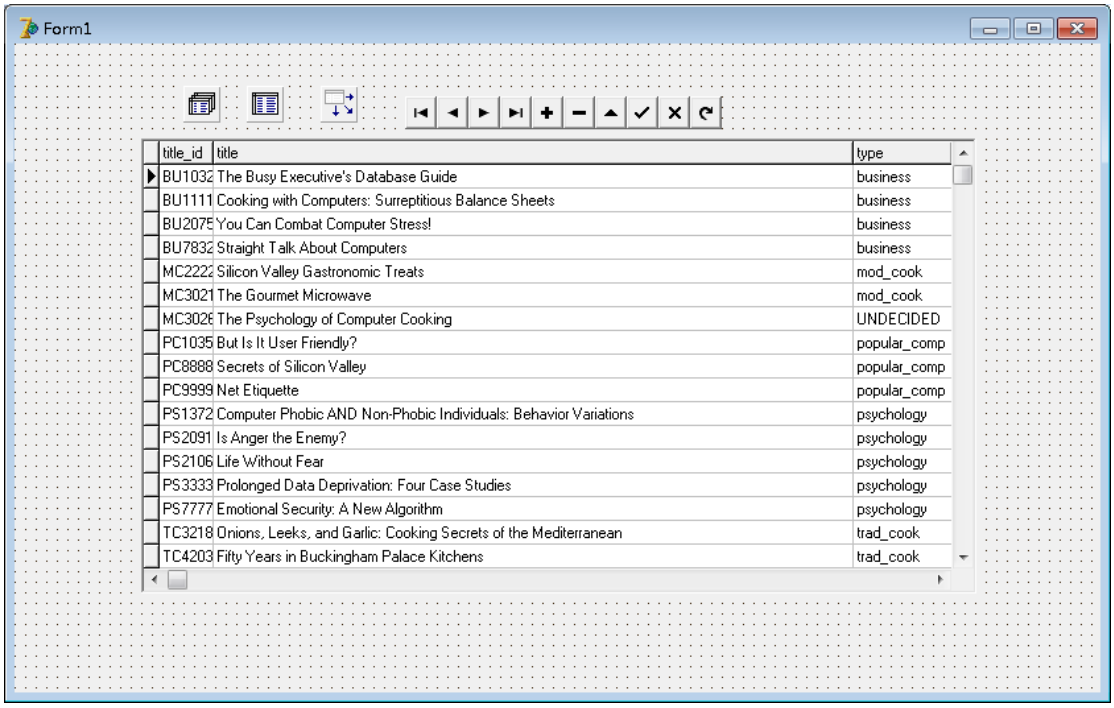
其“TableName”属性可以相应选择为某个表，如“dbo.titles”。

3）在窗体上放置一个“DataAccess”控件组中的 DataSource 控件并设置 DataSet= Table1（即 2）中放置的表）

4）再放置一个“Data Control”控件组中的“DBGrid”控件并设置 DataSource= DataSource1

5）再放置一个“Data Control”控件组中的“DBNavigator”控件，设置 DataSource= DataSource1

于是，可以得到以下设计结果。



运行程序，操控 DBNavigator 可以观察效果。

3、使用 Delphi 中其他的数据库组件

请同学们在熟悉了以上要求之后，采用 Delphi 中 Query 构件创建自己的数据库连接；有时间的同学可以尝试一下 Delphi 中的动态 SQL 语句。在编制程序的时候，尽量多使用一些组件，以便能做出有自己特色的程序。

四、思考题

1、 什么是 BDE？简介 BDE 的功能，并说明在 Delphi 中 BDE 配置文件保存在什么路径下。

2、 简述用 Table 构件创建数据库连接和用 Query 创建数据库连接有什么区别？你认为哪一种方法比较有优势？

第三部分 用 VC 做数据库开发

一、 实验目的

1、 掌握 ODBC 的概念并学习使用 ODBC 访问数据库；
2、 学习 Visual C++下如何通过 ODBC 访问数据库；
3、 学习用 Visual C++制作简单数据库应用程序的方法。

二、 实验原理

Microsoft 推出的 ODBC(Open Database Connectivity)技术为异质数据库的访问提供了统一的接口。ODBC 基于 SQL(Structured Query Language)，并把它作为访问数据库的标准。这个接口提供了最大限度的相互可操作性：一个应用程序可以通过一组通用的代码访问不同的数据库管理系统。一个软件开发者开发的客户/服务器应用程序不会被束定于某个特定的数据库之上。ODBC 可以为不同的数据库提供相应的驱动程序。

ODBC 的灵活性表现在以下几个方面:

应用程序不会受制于某种专用的 API
SQL 语句以源代码的方式直接嵌入在应用程序中
应用程序可以以自己的格式接收和发送数据
ODBC 的设计完全和 ISO Call-Level Interface 兼容
现在的 ODBC 数据库驱动程序支持 55 家公司的数据产品

最新推出的 ODBC 3.0 已经升级到 32 位,支持 Win32。

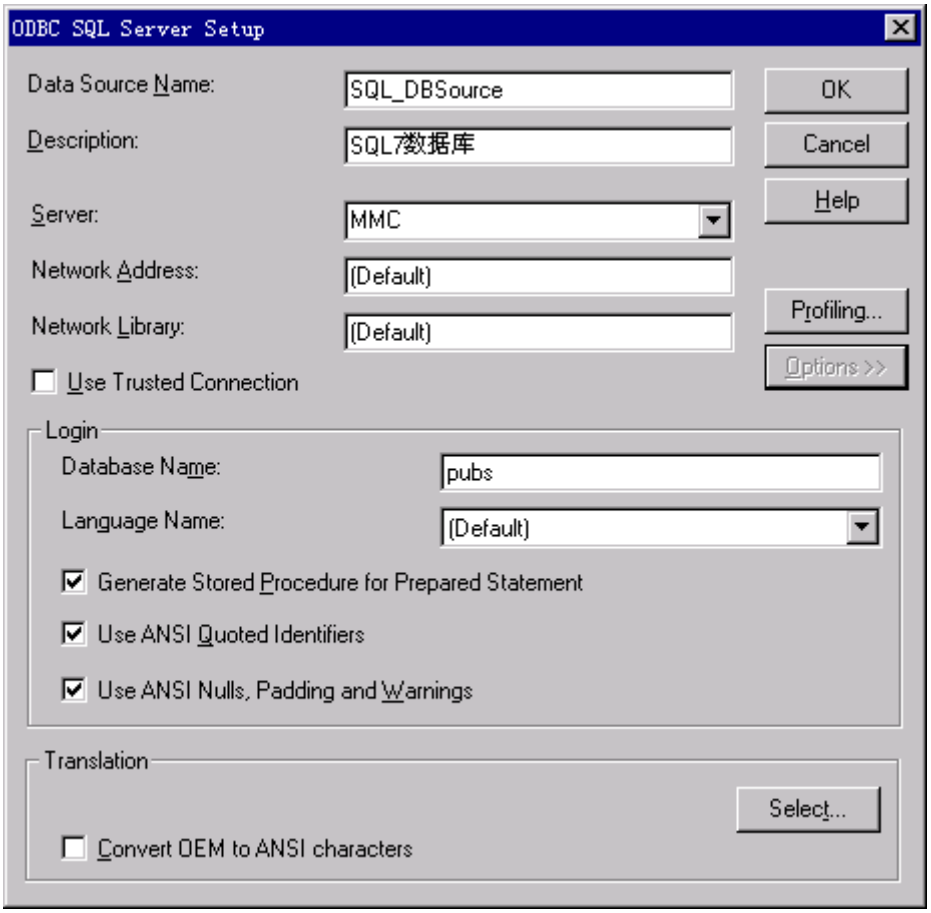
在 Visual C++中，提供了良好的数据库支持。输入数据源是遵循开放式数据库互接(ODBC)标准，还是微软的数据访问对象(DAO)标准，或 OLE 数据库(OLE DB)标准。ODBC 功能在数据库管理系统，如 Microsoft Access、Oracle 或 dBase 的特定的驱动程序中实现。Visual C++提供了一个 ODBC 驱动程序的集合；其他的可从经销商那里买到。实验中即利用 ODBC 接口来创建 VC 下的数据库支持。

三、 实验要求

1、ODBC 数据源设置

打开控制面板，即可以看到 ODBC 设置程序的图标。利用该管理程序，可以完成对数据库源的定义工作。

选择“添加(D)”，可以看到弹出的对话框中含有关于数据库服务器的内容设置，如图做常规设置。

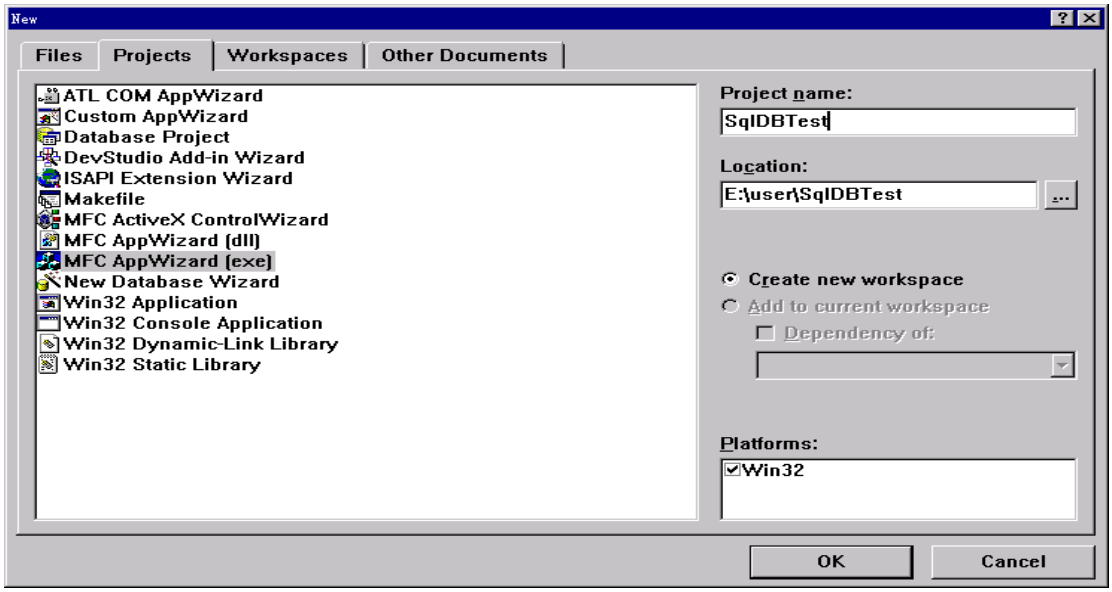


2、Visual C++制作数据库应用

（1）用 AppWizard 创建带数据库支持的应用程序

数据库编程的第一步是用 AppWizard 创建带数据库支持的应用程序：

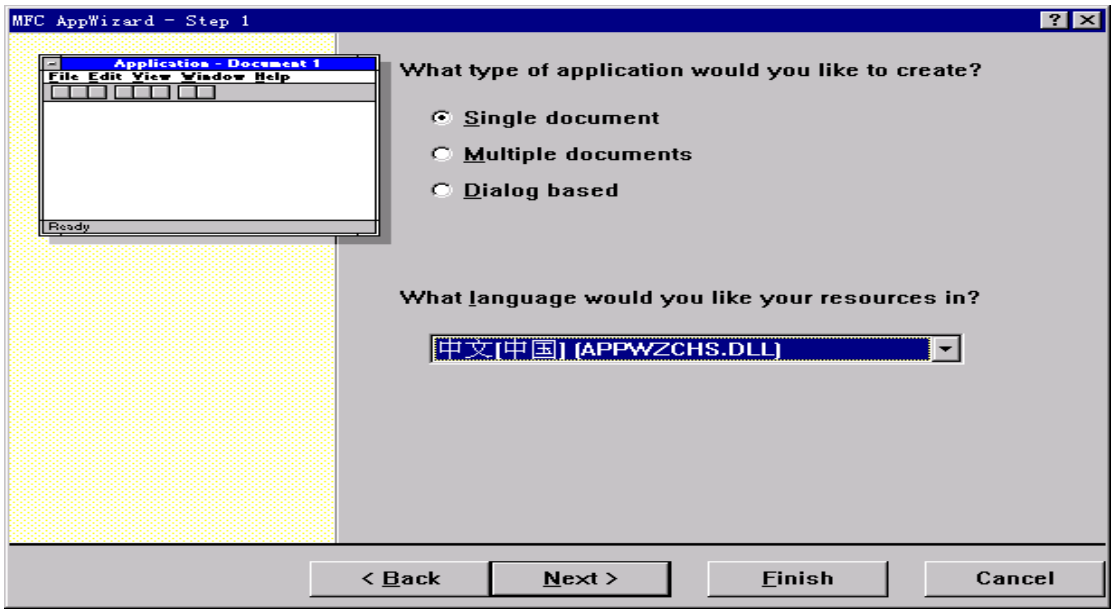
- 从“File”菜单选择“New”选项，打开“New”对话框。
- 切换到“Projects”选项卡，选择项目类型为“MFC AppWizard(exe)”。



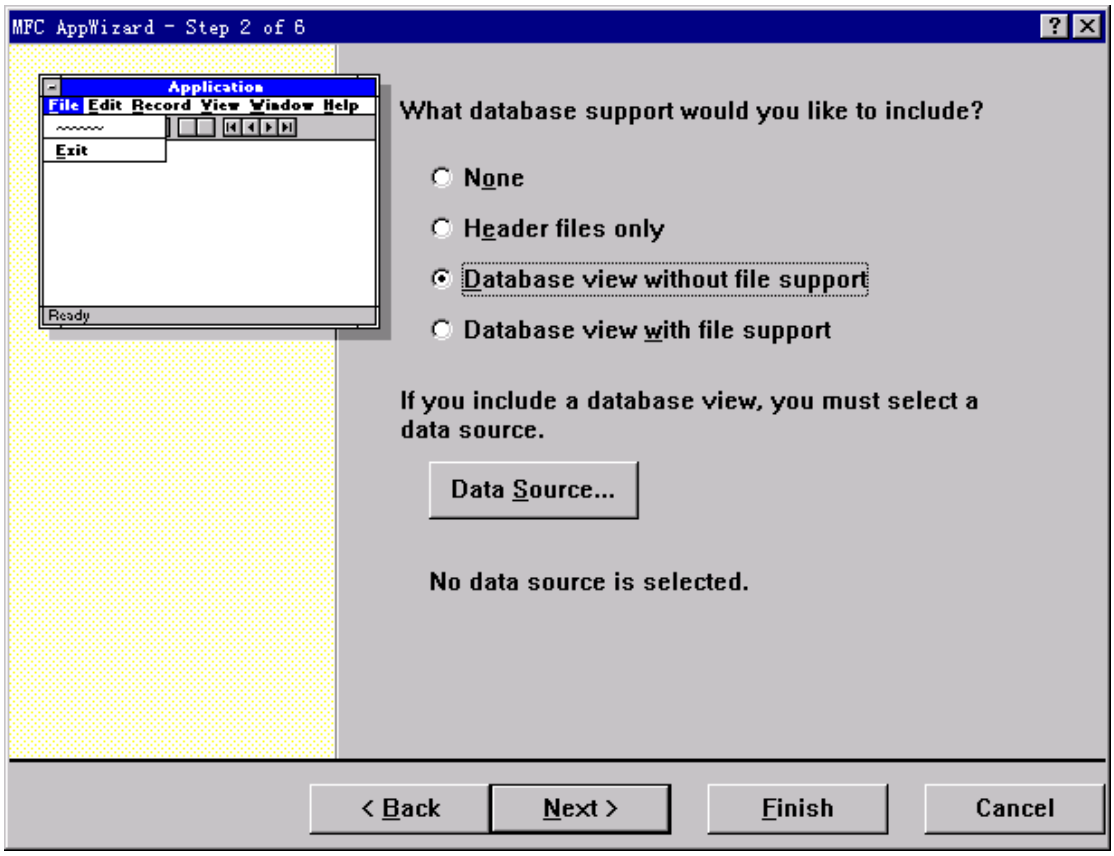
- 从“Projects name”编辑框中键入项目名字，如：“sqlDBTest”。

从“Location”编辑框中键入用于存放项目的根目录，如“E:\USER”。

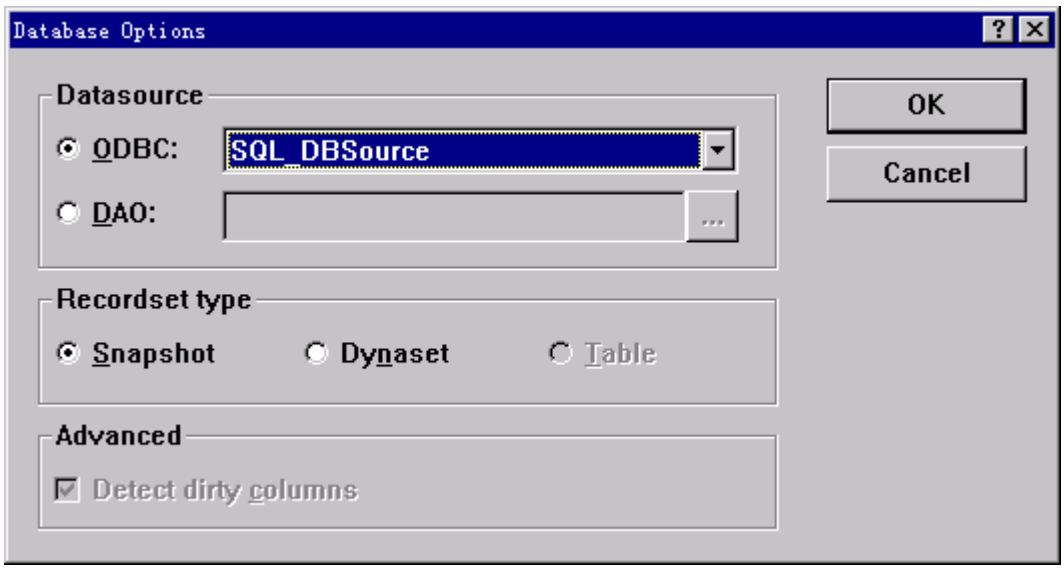
- 单击“OK”按钮，从弹出的对话框选择程序结构。这里选择“Single Document”，表示是单文档结构，并将资源中使用的语言设置成中文。



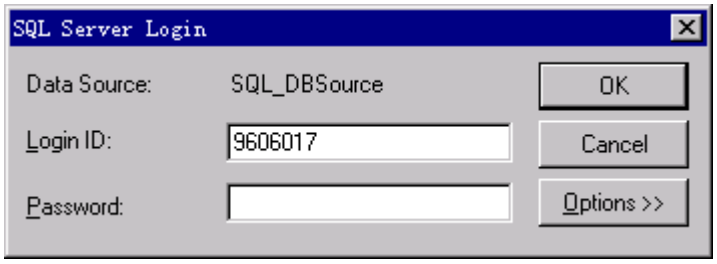
- 单击“Next”按钮，从弹出的对话框选择数据库支持。这里选中“Database view without file support”，这样创建的应用程序是用户可以查看和更新记录。注意：选择“Header files only”（只有头文件）：在生成过程中，包含数据库头文件和库文件，但 AppWizard（应用程序向导）不为数据库类生成源代码。你必须自己写所有的源代码。这个选项适用于那些开始时不使用数据库，但你打算以后添加数据库支持的项目。



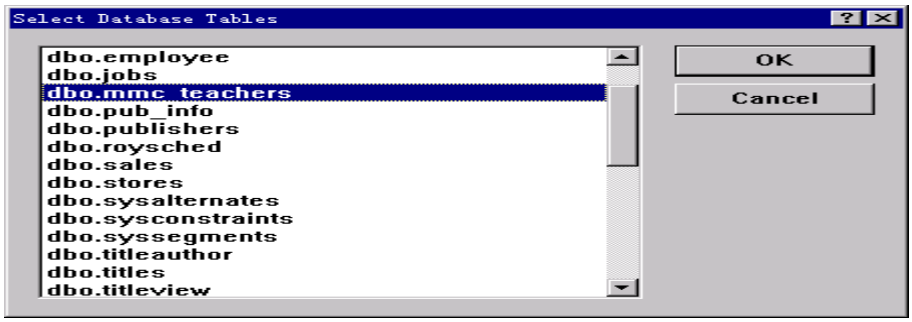
- 单击“Data Source”按钮，弹出“Database Options”对话框。



- 选中“ODBC”然后选择数据源，如“SQL_DBSource”。
- 单击“OK”按钮，在“Password”对话框中输入密码后，从弹出的对话框中选择一个表名，如“dbo.mmc_teachers”。

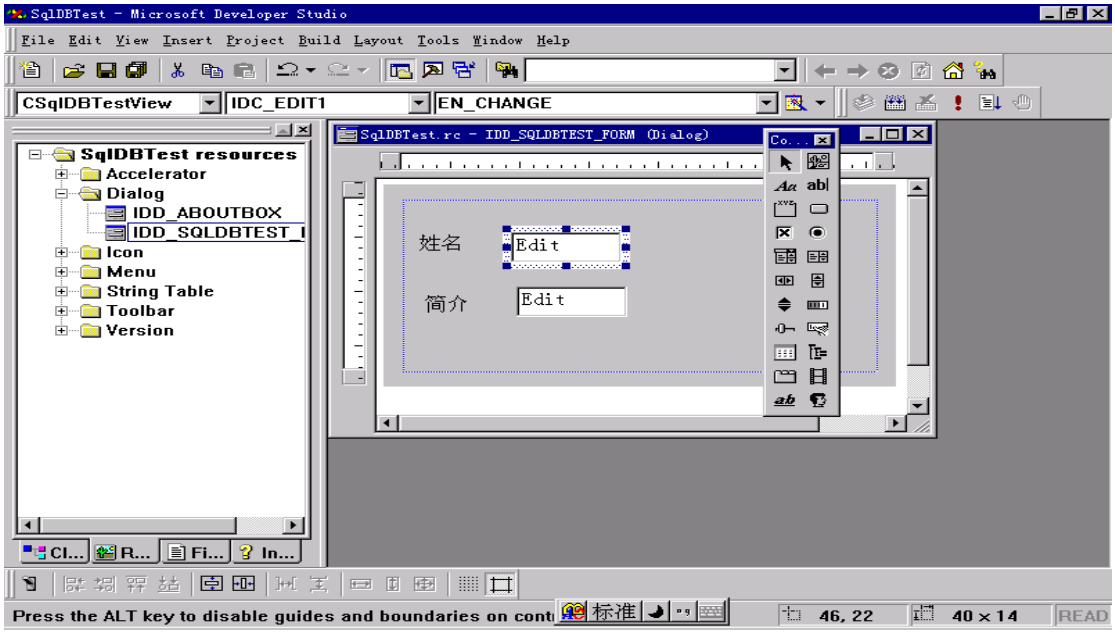


- 单击“OK”按钮，返回数据源选择的对话框。单击“Finish”按钮，完成。



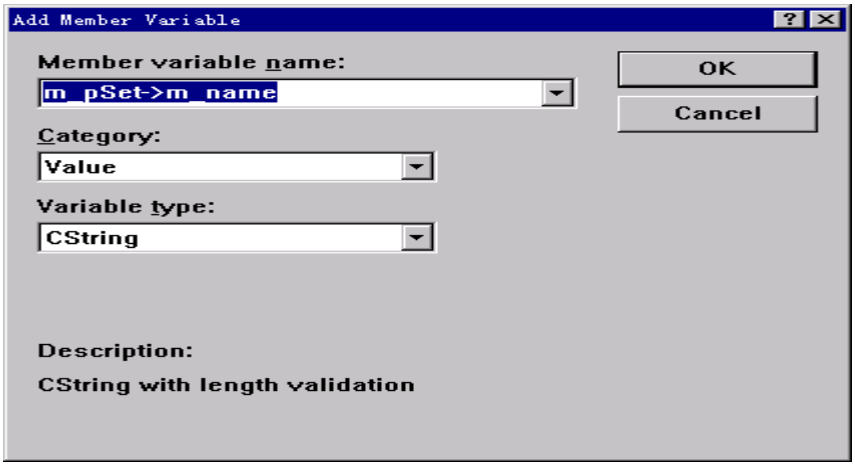
（2）定制对话框模板

- 在项目工作区窗口，切换到“ResourceView”面板。
- 打开 AppWizard 提供的对话框模板。设计如图示。



（3）在控件与数据库记录集成员变量之间建立关联

- 在对话框编辑器中，按住 CTRL 键双击某一输入框，弹出“Add Member Variable”对话框并如下图选择。



- 重复以上步骤。
- 从“Build”菜单中选择“Execute sqlDBTest.exe”命令来运行程序。

四、 思考题

1、 本实验中 ODBC 是指什么？它和上一个实验中的 BDE 有什么异同？
2、 请简述在数据库应用程序开发过程中有那些主要的步骤？

第四部分　用 PowerBuilder 做数据库开发

一、 实验目的

1、 进一步理解 ODBC 以及使用 ODBC 访问数据库；
2、 学习 PowerBuilder 下如何通过 ODBC 访问数据库；

二、 实验原理

PowerBuilder 6.0 中对数据库得访问是通过“PowerBuilder Databse”画板来实现的。

当你第一次打开数据库画板时，你会看到“PowerB 的 der Demo DB V6”显示在画板的标题栏中，这是 PowerBuilder 安装用作应用范例的数据库。

通过选择菜单中的 File→Create Database，PowerBuilder 可以创建自己的数据库－－Sybase SQL Anywhere。

通过选择菜单中的 File→Connect，PowerBuilder 允许你连接到其它 Profile 已经在 PowerBuilder 中定义过了的任一数据库上。File→Connect→Setup 则可以为已存在的数据库创建新的 profile。

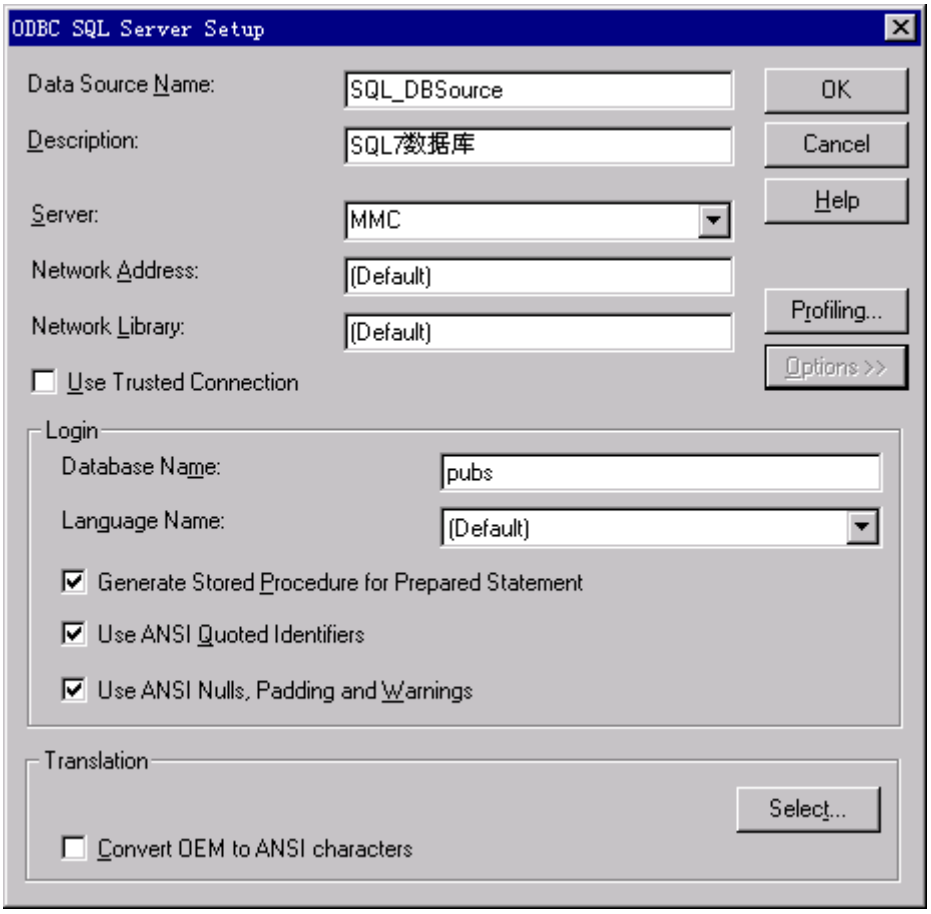
PowerBuilder 提供了访问与修改库中数据的若干种方法：Data ManiPulation Window(数据操纵窗口)，在数据库管理员画板使用 SQL 语句以及 Data PiPeline(数据管道)。

三、 实验要求

1、ODBC 数据源设置

打开控制面板，即可以看到 ODBC 设置程序的图标。利用该管理程序，可以完成对数据库源的定义工作。

选择“添加(D)”，可以看到弹出的对话框中含有关于数据库服务器的内容设置，如图做常规设置。



2、建立本地数据库

使用 PowerBuilder Database 画板建立一个本地数据库(Sybase SQL Anywhere)。

3、与 SQL Server 相连

使用 PowerBuilder Database 画板与 1 中设立的 ODBC 数据源（SQL Server）相连。

4、操作 SQL Server 中的数据

使用 Data ManiPulation Window 访问 SQL Server 中的数据，查询、修改等。

四、 思考题

与 VC++相比，在数据库应用程序开发过程中 PowerBuilder 有那些优势？