colClasses为每一列指定一个类，例如logical（逻辑型）、numeric（数值型）、character（字符型）、factor（因子）。

函数read.table()还拥有许多微调数据导入方式的追加选项。更多详情，请参阅help(read.table)。

注意 本章中的许多示例都是从用户计算机上已经存在的文件中导入数据。R也提供了若干种通过连接（connection）来访问数据的机制。例如，函数file()、gzfile()、bzfile()、xzfile()、unz()和url()可作为文件名参数使用。函数file()允许用户访问文件、剪贴板和C级别的标准输入。函数gzfile()、bzfile()、xzfile()和unz()允许用户读取压缩文件。函数url()能够让你通过一个含有http://、ftp://或file://的完整URL访问网络上的文件，还可以为HTTP和FTP连接指定代理。为了方便，（用""围住的）完整的URL也经常直接用来代替文件名使用。更多详情，参见help(file)。

2.3.3 导入Excel数据

读取一个Excel文件的最好方式，就是在Excel中将其导出为一个逗号分隔文件（csv），并使用前文描述的方式将其导入R中。在Windows系统中，你也可以使用RODBC包来访问Excel文件。电子表格的第一行应当包含变量/列的名称。

首先，下载并安装RODBC包。

```
install.packages("RODBC")
```

你可以使用以下代码导入数据：

```
library(RODBC)
channel <- odbcConnectExcel("myfile.xls")
mydataframe <- sqlFetch(channel, "mysheet")
odbcClose(channel)
```

这里的myfile.xls是一个Excel文件，mysheet是要从这个工作簿中读取工作表的名称，channel是一个由odbcConnectExcel()返回的RODBC连接对象，mydataframe是返回的数据框。RODBC也可用于从Microsoft Access导入数据。更多详情，参见help(RODBC)。

Excel 2007使用了一种名为XLSX的文件格式，实质上是多个XML文件组成的压缩包。xlsx包可以用来读取这种格式的电子表格。在第一次使用此包之前请务必先下载并安装好。包中的函数read.xlsx()可将XLSX文件中的工作表导入为一个数据框。其最简单的调用格式是read.xlsx(*file*, *n*)，其中*file*是Excel 2007工作簿的所在路径，*n*则为要导入的工作表序号。举例说明，在Windows上，以下代码：

```
library(xlsx)
workbook <- "c:/myworkbook.xlsx"
mydataframe <- read.xlsx(workbook, 1)
```

从位于C盘根目录的工作簿myworkbook.xlsx中导入了第一个工作表，并将其保存为一个数据框mydataframe。xlsx包不仅仅可以导入数据表，它还能够创建和操作XLSX文件。那些需要为R和Excel开发接口的程序员应当研究一下这个较新的包。

2.3.4　导入XML数据

以XML格式编码的数据正在逐渐增多。R中有若干用于处理XML文件的包。例如，由Duncan Temple Lang编写的XML包允许用户读取、写入和操作XML文件。XML格式本身已经超出了本书的范围。对使用R存取XML文档感兴趣的读者可以参阅www.omegahat.org/RSXML，从中可以找到若干份优秀的软件包文档。

2.3.5　从网页抓取数据

在Web数据抓取（Webscraping）的过程中，用户从互联网上提取嵌入在网页中的信息，并将其保存为R中的数据结构以做进一步的分析。完成这个任务的一种途径是使用函数readLines()下载网页，然后使用如grep()和gsub()一类的函数处理它。对于结构复杂的网页，可以使用RCurl包和XML包来提取其中想要的信息。更多信息和示例，请参考可在网站*Programming with R*（www.programmingr.com）上找到的“Webscraping using readLines and RCurl”一文。

2.3.6　导入SPSS数据

SPSS数据集可以通过foreign包中的函数read.spss()导入到R中，也可以使用Hmisc包中的spss.get()函数。函数spss.get()是对read. spss()的一个封装，它可以为你自动设置后者的许多参数，让整个转换过程更加简单一致，最后得到数据分析人员所期望的结果。

首先，下载并安装Hmisc包（foreign包已被默认安装）：

```
install.packages("Hmisc")
```

然后使用以下代码导入数据：

```
library(Hmisc)
mydataframe <- spss.get("mydata.sav", use.value.labels=TRUE)
```

这段代码中，mydata.sav是要导入的SPSS数据文件，use.value.labels=TRUE表示让函数将带有值标签的变量导入为R中水平对应相同的因子，mydataframe是导入后的R数据框。

2.3.7　导入SAS数据

R中设计了若干用来导入SAS数据集的函数，包括foreign包中的read.ssd()和Hmisc包中的sas.get()。遗憾的是，如果使用的是SAS的较新版本（SAS 9.1或更高版本），你很可能会发现这些函数并不能正常工作，因为R尚未跟进SAS对文件结构的改动。个人推荐两种解决方案。

你可以在SAS中使用PROC EXPORT将SAS数据集保存为一个逗号分隔的文本文件，并使用

2.3.2节中叙述的方法将导出的文件读取到R中。下面是一个示例：

SAS程序：

```
proc export data=mydata
     outfile="mydata.csv"
     dbms=csv;
run;
```

R程序：

```
mydata <- read.table("mydata.csv", header=TRUE, sep=",")
```

另外，一款名为Stat/Transfer的商业软件（在2.3.12节介绍）可以完好地将SAS数据集（包括任何已知的变量格式）保存为R数据框。

2.3.8 导入Stata数据

要将Stata数据导入R中非常简单直接。所需代码类似于：

```
library(foreign)
mydataframe <- read.dta("mydata.dta")
```

这里，`mydata.dta`是Stata数据集，`mydataframe`是返回的R数据框。

2.3.9 导入netCDF数据

Unidata项目主导的开源软件库netCDF（network Common Data Form，网络通用数据格式）定义了一种机器无关的数据格式，可用于创建和分发面向数组的科学数据。netCDF格式通常用来存储地球物理数据。`ncdf`包和`ncdf4`包为netCDF文件提供了高层的R接口。

`ncdf`包为通过Unidata的netCDF库（版本3或更早）创建的数据文件提供了支持，而且在Windows、Mac OS X和Linux上均可使用。`ncdf4`包支持netCDF 4或更早的版本，但在Windows上尚不可用。

考虑如下代码：

```
library(ncdf)
nc <- nc_open("mynetCDFfile")
myarray <- get.var.ncdf(nc, myvar)
```

在本例中，对于包含在netCDF文件mynetCDFfile中的变量`myvar`，其所有数据都被读取并保存到了一个名为`myarray`的R数组中。

值得注意的是，`ncdf`包和`ncdf4`包最近进行了重大升级，使用方式可能与旧版本不同。另外，这两个包中的函数名称也不同。请阅读在线文档以了解详情。

2.3.10 导入HDF5 数据

HDF5（Hierarchical Data Format，分层数据格式）是一套用于管理超大型和结构极端复杂数据集的软件技术方案。`hdf5`包能够以那些理解HDF5格式的软件可以读取的格式，将R对象写入到一个文件中。这些文件可以在之后被读回R中。这个包是实验性质的，并假设用户已经安装了

HDF5库（1.2版或更高）。目前，R对于HDF5格式的支持非常有限。

2.3.11　访问数据库管理系统

R中有多种面向关系型数据库管理系统（DBMS）的接口，包括Microsoft SQL Server、Microsoft Access、MySQL、Oracle、PostgreSQL、DB2、Sybase、Teradata以及SQLite。其中一些包通过原生的数据库驱动来提供访问功能，另一些则是通过ODBC或JDBC来实现访问的。使用R来访问存储在外部数据库中的数据是一种分析大数据集的有效手段（参见附录G），并且能够发挥SQL和R各自的优势。

1. ODBC接口

在R中通过RODBC包访问一个数据库也许是最流行的方式，这种方式允许R连接到任意一种拥有ODBC驱动的数据库，其实几乎就是市面上的所有数据库。

第一步是针对你的系统和数据库类型安装和配置合适的ODBC驱动——它们并不是R的一部分。如果你的机器尚未安装必要的驱动，上网搜索一下应该就可以找到。

针对选择的数据库安装并配置好驱动后，请安装RODBC包。你可以使用命令`install.packages("RODBC")`来安装它。

RODBC包中的主要函数列于表2-2中。

表2-2　RODBC中的函数

函　数	描　述
odbcConnect(*dsn*,uid="",pwd="")	建立一个到ODBC数据库的连接
sqlFetch(*channel*,*sqltable*)	读取ODBC数据库中的某个表到一个数据框中
sqlQuery(*channel*,*query*)	向ODBC数据库提交一个查询并返回结果
sqlSave(*channel*,*mydf*,tablename=*sqtable*,append=FALSE)	将数据框写入或更新（append=TRUE）到ODBC数据库的某个表中
sqlDrop(*channel*,*sqtable*)	删除ODBC数据库中的某个表
close(*channel*)	关闭连接

RODBC包允许R和一个通过ODBC连接的SQL数据库之间进行双向通信。这就意味着你不仅可以读取数据库中的数据到R中，同时也可以使用R修改数据库中的内容。假设你想将某个数据库中的两个表（Crime和Punishment）分别导入为R中的两个名为crimedat和pundat的数据框，可以通过如下代码完成这个任务：

```
library(RODBC)
myconn <-odbcConnect("mydsn", uid="Rob", pwd="aardvark")
crimedat <- sqlFetch(myconn, Crime)
pundat <- sqlQuery(myconn, "select * from Punishment")
close(myconn)
```

这里首先载入了RODBC包，并通过一个已注册的数据源名称（mydsn）和用户名（rob）以及密码（aardvark）打开了一个ODBC数据库连接。连接字符串被传递给sqlFetch，它将Crime

表复制到R数据框crimedat中。然后我们对Punishment表执行了SQL语句select并将结果保存到数据框pundat中。最后，我们关闭了连接。

函数sqlQuery()非常强大，因为其中可以插入任意的有效SQL语句。这种灵活性赋予了你选择指定变量、对数据取子集、创建新变量，以及重编码和重命名现有变量的能力。

2. DBI相关包

DBI包为访问数据库提供了一个通用且一致的客户端接口。构建于这个框架之上的RJDBC包提供了通过JDBC驱动访问数据库的方案。使用时请确保安装了针对你的系统和数据库的必要JDBC驱动。其他有用的、基于DBI的包有RMySQL、ROracle、RPostgreSQL和RSQLite。这些包都为对应的数据库提供了原生的数据库驱动，但可能不是在所有系统上都可用。详情请参阅CRAN（http://cran.r-project.org）上的相应文档。

2.3.12 通过Stat/Transfer导入数据

在我们结束数据导入的讨论之前，值得提到一款能让上述任务的难度显著降低的商业软件。Stat/Transfer（www.stattransfer.com）是一款可在34种数据格式之间作转换的独立应用程序，其中包括R中的数据格式（见图2-4）。

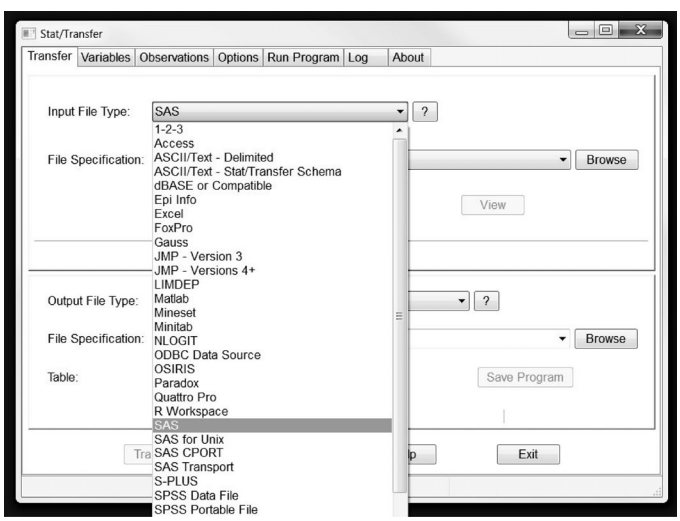

图2-4 Windows上Stat/Transfer的主对话框

此软件拥有Windows、Mac和Unix版本，并且支持我们目前讨论过的各种统计软件的最新版本，也可通过ODBC访问如Oracle、Sybase、Informix和DB/2一类的数据库管理系统。

2.4 数据集的标注

为了使结果更易解读，数据分析人员通常会对数据集进行标注。通常这种标注包括为变量名