China-pub.com
下载

第4章　数据库简介

本章要点：

- Microsoft 的新一代网络数据库SQL Server 7.0；
- 了解结构化查询语言。

4.1　MS SQL Server 7.0

在当今提出的多种动态网页 (DHTML)解决方案中都强调了与数据库的连接，其实网页接挂后台数据库也是当前的热门应用，在电子商务等领域有着广泛的应用。如果你不能掌握在 ASP中使用数据库，那么也不能编写出功能强大的 ASP应用程序。 ASP用Database Access 组件与数据库进行连接，Database Access 组件通过 ActiveX Data Objects (ADO) 访问存储在数据库或其他表格化数据结构中的信息。

现在的Web应用程序中，会大量地用到数据库操作。尽管 ASP 可以使用任何 ODBC 兼容的数据库，支持共享文件数据库（如 Microsoft Access 或 Microsoft FoxPro）作为有效的数据源，但我们建议只将此类数据库引擎用于开发小型的，且一般同时访问者不会超过 10个应用程序系统。共享文件数据库可能无法很好地满足高需求、高质量的 Web 应用程序的需要。由于性能和可靠性的原因，一般应用程序都使用带有“客户 /服务器数据库引擎”的关系数据库管理系统，这些关系数据库管理系统包括 Microsoft SQL Server、Oracle 等。其中，我们推荐你使用 SQL Server 7.0，它可以很好地与Microsoft的ASP合作。

在SQL Server 7.0中，一切操作都是基于数据的，而共享文件数据库是基于文件的。就是说，在SQL Server 7.0中检索数据时，将提交一个查询目的，服务器处理后返回的是查询结果，而共享文件数据库检索数据时，它返回相关的整个表，你需要在本地的表中查询出结果。如数据库中的一个表包含 10000条记录，你需要的是其中的一条信息，那么当你是用 SQL Server 7.0时，只要将查询语句发送到服务器，它将你要的结果返回，而共享文件数据库将这 10000条记录的表返回，然后在这个表中查询。其网络的流量差别可想而知。

4.1.1　简介

SQL Server 7.0是Microsoft最新推出的网络数据库系统，它为在其上建立应用程序的开发者提供了一个优秀的关系数据库管理系统。

SQL Server 7.0可以安装到 Microsoft Windows NT Server上，也可以安装到 Microsoft Windows NT Workstation和Windows 95/98上，这在过去是不可能的，而也是别的网络数据库系统所不具备的。因此，无论是基于 Microsoft Windows NT Server 企业级的应用，还是基于桌面工作站的小型数据库应用程序， SQL Server 7.0都为用户提供了完美的数据库支持。

China-pub.com 下载

SQL Server 7.0的使用简单清晰，它通过管理工具Enterprise Manager来实现数据库的大部分操作。

为了提高SQL Server 7.0的工作性能，Microsoft对SQL Server底层做了根本性的改变，不仅改写了数据库引擎，还添加了许多实用的工具。

SQL Server 7.0对数据库引擎所做的改进包括以下几个方面：

- 查询处理器：SQL Server 7.0中的查询处理器经过了重新设计，使能够支持决策支持系统、数据仓库和OLAP应用程序中的大型数据库和复杂的查询。
- 并行查询：SQL Server 7.0可通过多个处理器对单个查询进行并行运算。
- 分布式查询：能够访问存储在同一或不同的计算机上的多个数据源。这些查询利用了OLE DB、访问非相关数据或相关数据资源的标准等功能。
- 索引操作：SQL Server 7.0使用交叉索引和联合索引来完成对一个数据库的多次索引。
- 触发器：在SQL Server 7.0中，可以对单个表追加同种类型的多个触发器，这在以前的版本中是不可能的。
- 页面和行的格式：数据库的页容量从过去的2KB增加到现在的8KB，一行的最大字节数为8060B，字符型和二进制的容量从以前的255B增加到8000B。所支持的表的列数也从以前的250列增加到现在的1024列。
- 动态行级锁定：SQL Server 7.0支持对数据行和索引条目进行全行级锁定。
- 动态内存管理和空间管理：在运行时，SQL Server 7.0可以自动和动态地进行重新配置。
- 数据库和文件：SQL Server 7.0简化了SQL Server数据库和Windows文件系统的联系，使其的可伸缩性得到进一步的加强。用户可以用单一的语句或Enterprise Manager创建一个数据库和所有文件，当一个数据库被删除后，它的文件也同时被删除。SQL Server 7.0允许数据库文件自动地扩展，这减少了手工操作的工作量。
- 对Unicode的支持：SQL Server 7.0支持Unicode数据类型，通过减少转换字符类型和安装多种代码页，使一个数据库中以多种语言存储数据变得更加容易。

4.1.2 数据库操作

SQL Server管理两种类型的数据库：系统数据库和用户数据库。系统数据库存储SQL Server专用的用于管理自身和用户数据库的数据，用户数据库用于存储用户数据。SQL Server创建的系统数据库包括Master、model、tempdb、msdb，还会创建一个叫PUBS的用户数据库样本。

SQL Server中，数据库把所有的数据与数据库对象都放在一系列操作文件中，并用文件和文件组管理这些操作系统文件。文件分为3种：主文件、从属文件和日志文件，每个文件只能从属于一个数据库。

主文件是一个数据库的起始点，一个数据库文件只能有一个主文件而且必须有一个主文件。主文件的扩展名是MDF。从属文件的数目是任意的，一般一个小型的数据库可能没有从属文件，而一个大型数据库可能存在多个从属文件。从属文件和主文件一同存储数据以及数据库对象。从属文件的扩展名是NDF。日志文件用来存放数据库的事务日志信息，这些信息对恢复数据库是十分关键的，一般一个数据库必须至少有一个日志文件。日志文件的扩展名是LDF。

China-pub.com
下载

文件组就是某一方面有相似特性的文件的集合。

1. 建立数据库

在SQL Server中，数据库的建立、使用、更改都可以用管理工具 Enterprise Manager完成。下面，我们用Enterprise Manager建立数据库MIS，它包含一个数据文件和一个日志文件。

文件名	物理文件位置	初始大小	文件组
MIS_DAT	D:\MIS\MIS_DAT.MDF	5MB	PRIMARY
MIS_LOG	D:\MIS\MIS_LOG.LDF	1MB	

首先，在Enterprise Manager 中，扩展到要在其上建立数据库的服务器，用鼠标单击该服务器左边的“ + ”号，然后在该服务器中的 DataBase文件夹上单击鼠标右键，在弹出菜单上选择New DataBase选项（见图 4-1），就会出现 Database Properties对话框，该对话框分为 2页：General 页和Transaction 页（见图4-2），先来看General 页。

General 页用来定义数据库的库名及该数据库所用数据文件的属性。在 NAME一项中填写MIS，这时，我们看见在Database Files项下SQL Server自动生成了一个数据文件，File Name 为MIS_DATA、 Location 为D:\MSSQL7\data\MIS_Data.MDF 、Initial Size 为1、File Group为PRIMARY。在文件名前有一个小钥匙，代表该数据文件为数据库的主文件。我们按要求把文件

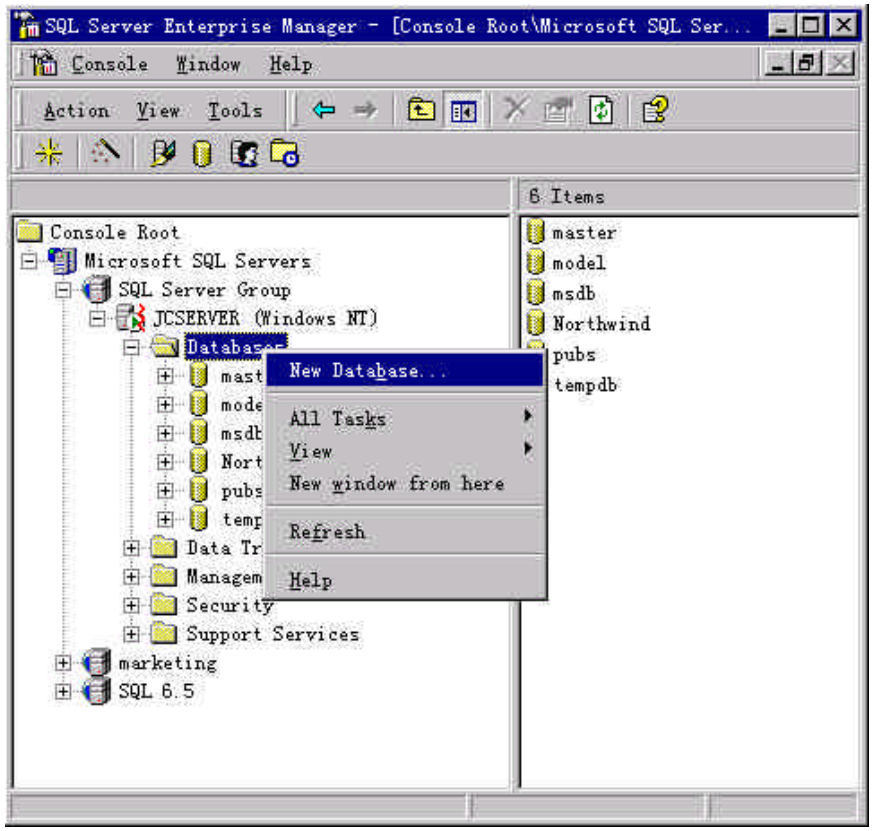

图4-1 New Database菜单

China-pub.com 下载

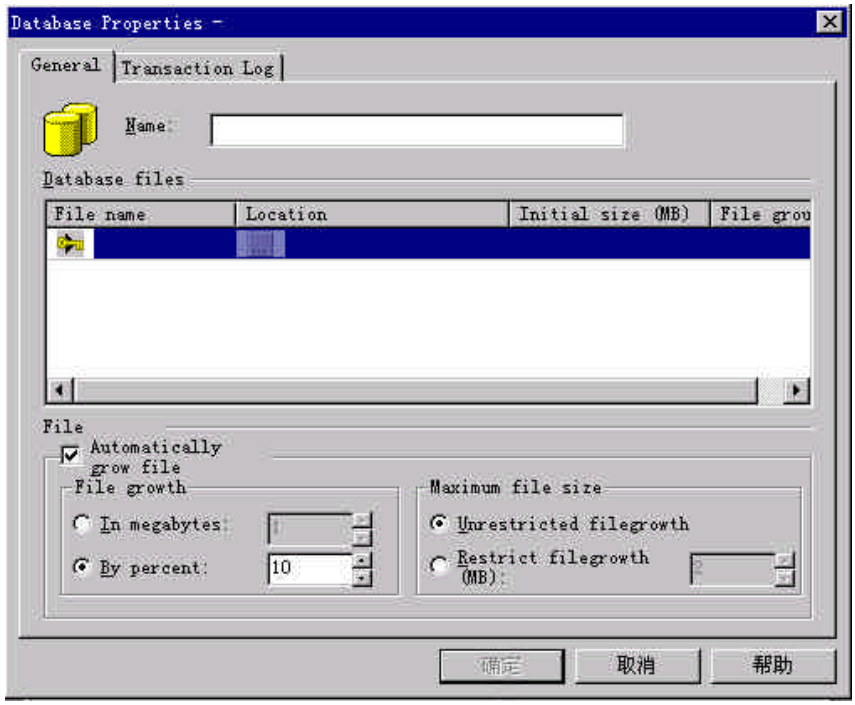

图4-2 建立数据库选项

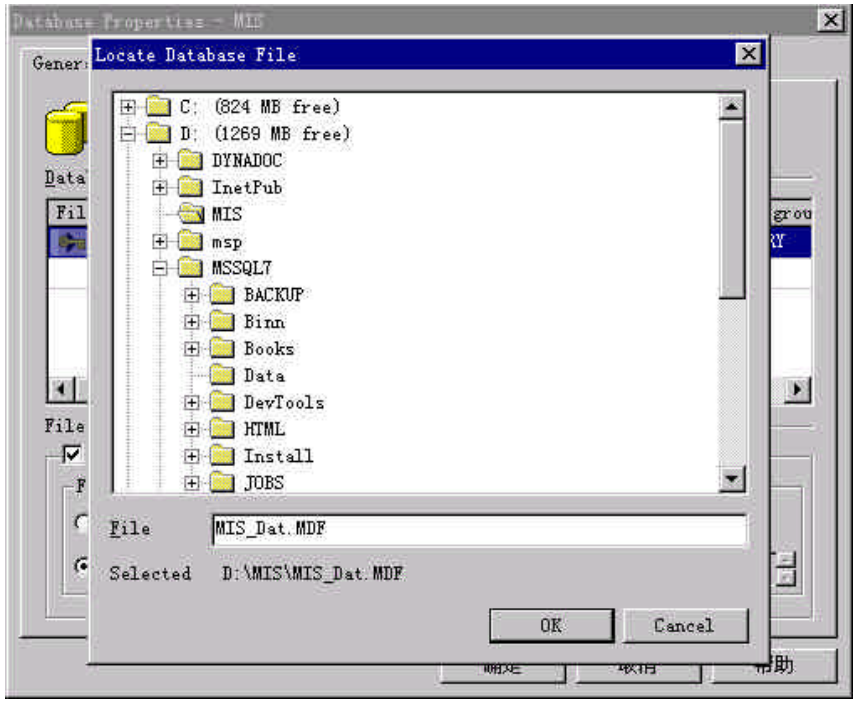

图4-3 选择数据库文件存放的物理位置

China-pub.com
下载

名改为 MIS_DAT，在文件名处单击鼠标左键，更改文件名。 Location改为 D:\MIS\MIS_DAT.MDF，在“...”按钮处单击鼠标左键，设定文件物理位置（见图 4-3），初始大小改为 5MB。在Initial Size处单击鼠标左键，更改文件初始大小。我们看到（见图 4-4）General 页中有一个 Automationcally grow file复选框，如果选上，代表当文件放满数据后，允许文件自动增长以容纳新的数据，In megabyte表示每次增长固定的大小，如 1MB，By percent表示以数据文件大小的一定比例增长，如 10%。如果没选上，则不允许文件自动增长。 Maximum file size设置文件的最大值，如10MB。更改完毕后，我们来设置日志文件，选择 Transaction 页。用同样的方法按要求更改完文件名及物理文件位置后，单击确定按钮，这样，数据库 MIS就建立完毕了 (见图4-5)。在路径“ D:\MIS ”下就建立了MIS_DAT.MDF和MIS_LOG.LDF两个文件。

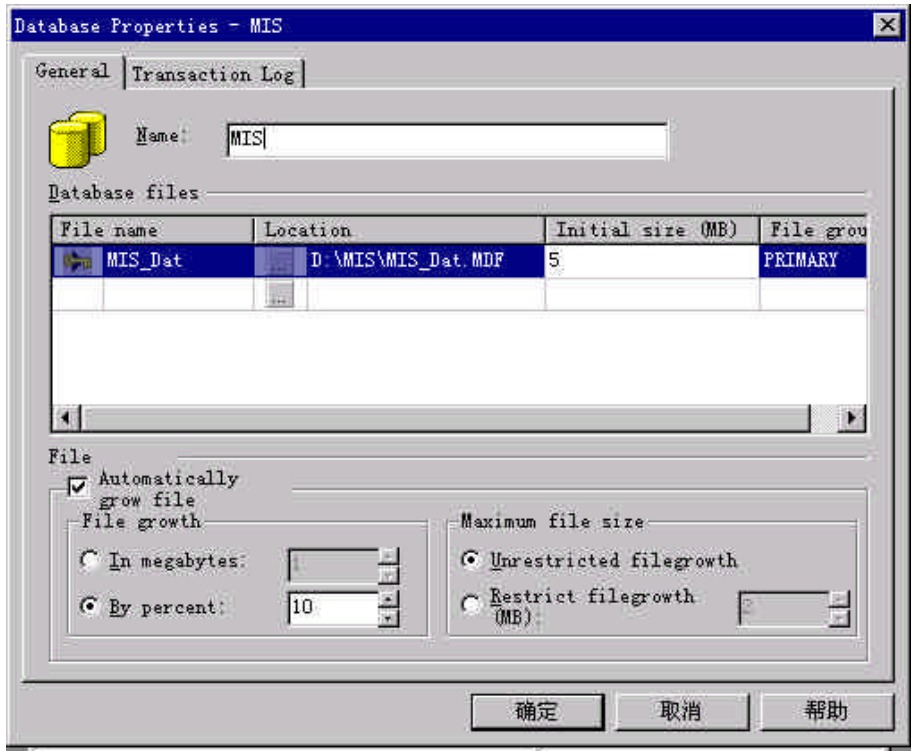

图4-4 数据库建立选项

2. 删除数据库

当一个数据库不再需要时，在那个数据库上单击鼠标左键，选择 Delete选项，在弹出的 Delete Database对话框上（见图4-6）选择“ 是 ”，这样，一个数据库及其所有文件就被删除了。

3. 建立表

单击在你要建立表的数据库左边的“ + ”，用鼠标选种 TABLES，单击鼠标右键，选择 New Table选项。接着，系统提示你输入表的名字（如 TSGL）（见图4-7）。输入完毕后，在系统给的表中依次输入列名、数据类型、列长等，保存你所输入的内容，这样，一个表建立完毕了（见图4-8）。

China-pub.com 下载

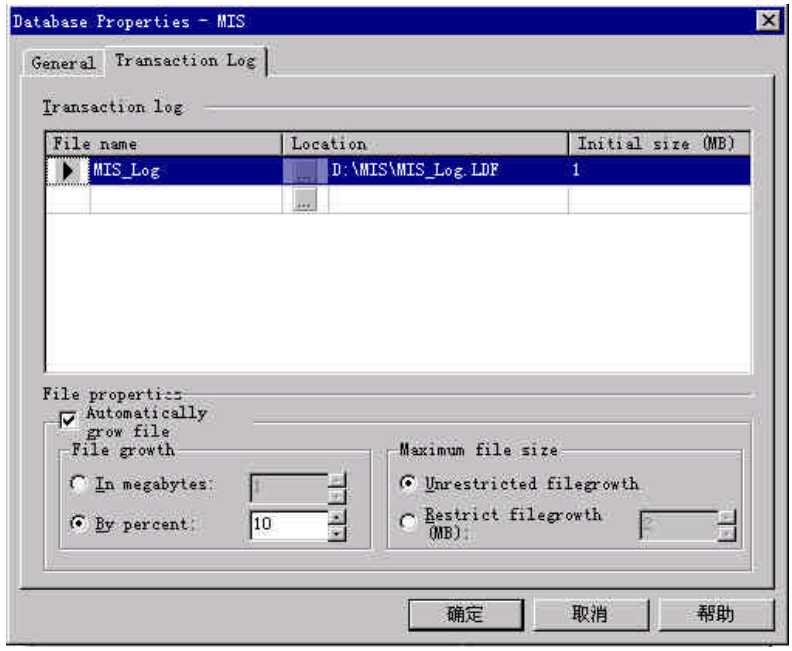

图4-5 建立日志文件

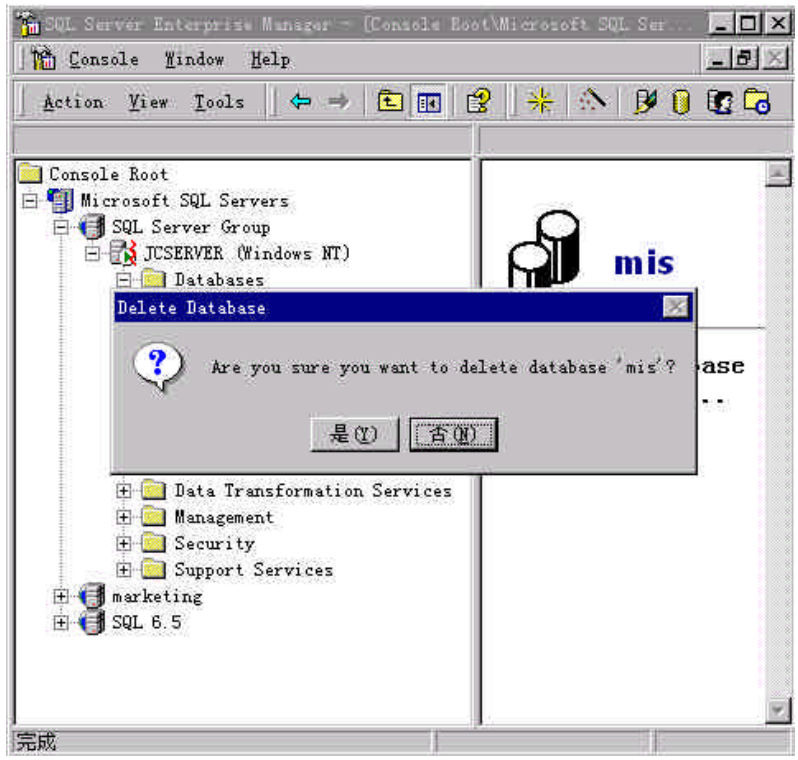

图4-6 删除数据库确认对话框

China-pub.com
下载

图4-7 输入要建立的表的名字

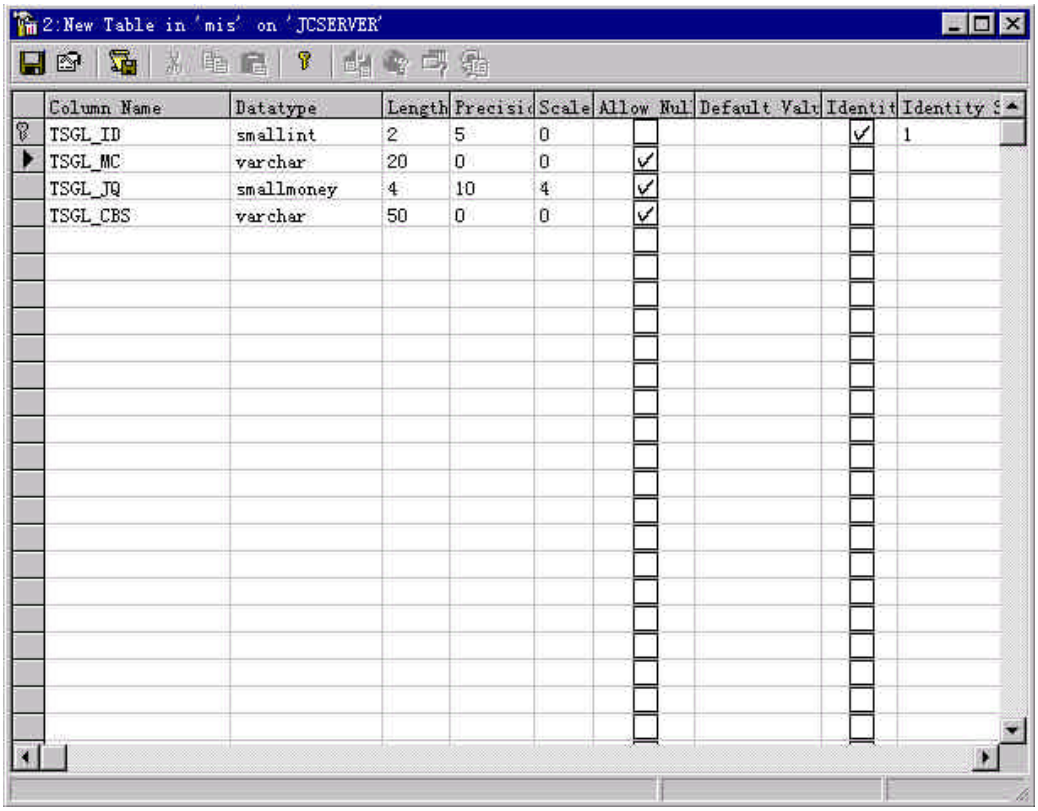

图4-8 表的设计

4. 删除表

选中你要删除的表，单击鼠标右键，选择 Delete选项，出现删除对话框（见图 4-9），单击Drop all删除你选中的表。

5. 建立索引

为一个表建立索引可以使使用者以极快的速度检索到所需要的数据。

表将占用更大的存储空间，而且数据的更新和删除操作也会耗用更多的时间。

索引包括：

China-pub.com 下载

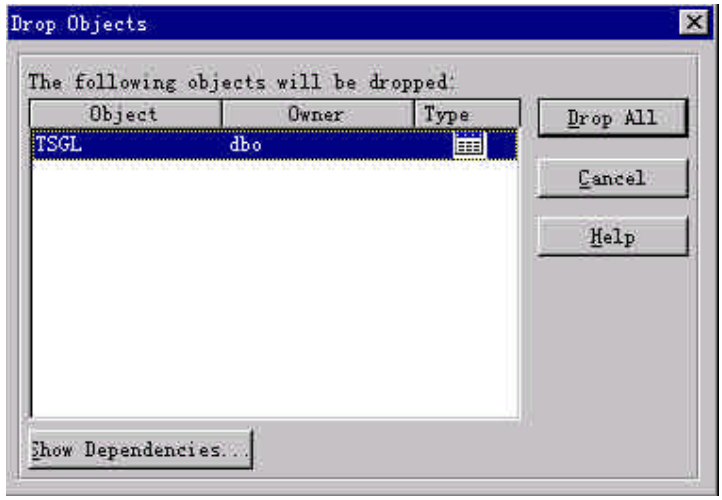

图4-9 删除表

- Clustered（集群）索引：一个表定义了Clustered索引后，记录始终按照被索引的列进行排序。
- NonClustered（非集群）索引：一个表只能有一个 Clustered索引，当定义多个索引后，其余索引被定义为 NonClustered索引。一个或多个列被定义为 NonClustered索引后，该索引中的信息是按照列值顺序存储的，而表中的数据并不是按照该列值顺序存储。

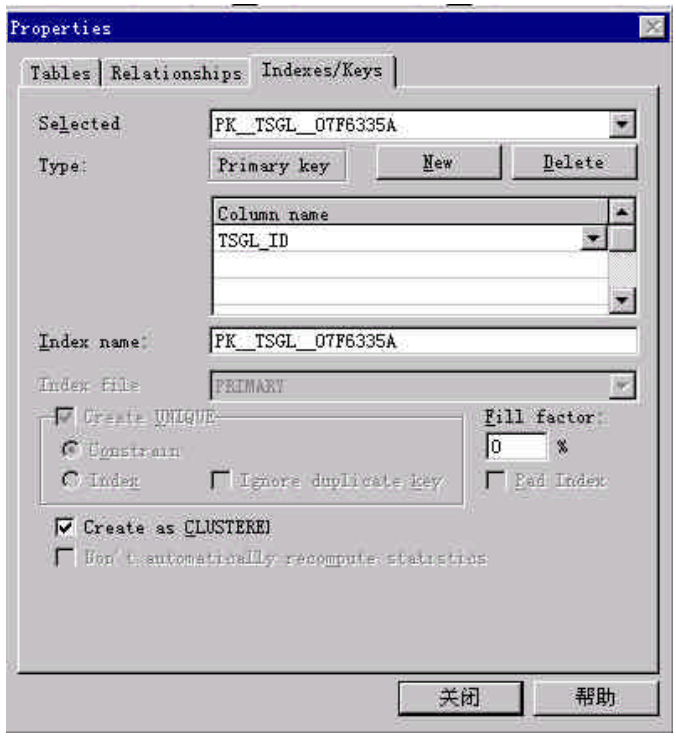

图4-10 建立索引

China-pub.com
下载

• Unique索引：Clustered和NonClustered都可以定义为Unique索引，当一列被定义为Unique索引后，在整张表中，该列将不允许包含重复的值。

建立索引的方法是在新建一个表的时候，在任意一列上单击鼠标右键，选择 Properties选项，都会弹出Properties对话框，选择Index/Keys页，对索引的所有操作都可以用 Index/Keys页完成（见图4-10）。

该对话框的各部分作用如下：

Select列表 选择要操作的索引，如果一个表还没有索引，则该列表框为灰色。

Type框 显示当前正在操作的索引的类型。

New按钮 建立一个新的索引。

Delete按钮 删除Select列表中选中的索引。

Column name列表 确定索引所在的列

Index name编辑框 为索引命名。

Index file列表 选定存放索引的文件组。

Create UNIQUE复选框 创建一个UNIQUE索引。

Create as CLUSTERED复选框 确定是否将一个索引创建为CLUSTERED索引。

6. 创建存储过程

存储过程是一种数据库对象，有一组预编译的 T-SQL语句组成。存储过程类似于其他编程语言的函数或过程：能够使用传递给它的参数，能够调用其他存储过程甚至本身，能够返回一个状态码来表示是否成功。

在SQL Server 中，存储过程是由流控制和 SQL语句书写的过程，这个过程经编译和优化后存储在数据库服务器中，使用时只要调用即可。在 SQL SERVER中，若干个有联系的过程可以组合在一起构成程序包。 使用存储过程有以下的优点：

存储过程的能力大大增强了SQL语言的功能和灵活性。存储过程可以用流控制语句编写，有很强的灵活性，可以完成复杂的判断和较复杂的运算。可保证数据的安全性和完整性。通过存储过程可以使没有权限的用户在控制之下间接地存取数据库，从而保证数据的安全。通过存储过程可以使相关的动作在一起发生，从而可以维护数据库的完整性。

再运行存储过程前，数据库已对其进行了语法和句法分析，并给出了优化执行方案。这种已经编译好的过程可极大地改善 SQL语句的性能。

由于执行SQL语句的大部分工作已经完成，所以存储过程能以极快的速度执行。可以降低网络的通信量。将体现企业规则的运算程序放入数据库服务器中，以便集中控制。当企业规则发生变化时在服务器中改变存储过程即可，无须修改任何应用程序。企业规则的特点是要经常变化，如果把体现企业规则的运算程序放入应用程序中，则当企业规则发生变化时，就需要修改应用程序，工作量非常之大（修改、发行和安装应用程序）。如果把体现企业规则的运算放入存储过程中，则当企业规则发生变化时，只要修改存储过程就可以了，应用程序无须任何变化。

创建存储过程的方法是在要创建的数据库左边单击“ +”，在该数据库的Stored Procedures子文件夹上单击鼠标右键，选择 New Stored Procedure...选项，就会弹出 Stored Procedures Properties对话框（见图4-11）。

China-pub.com 下载

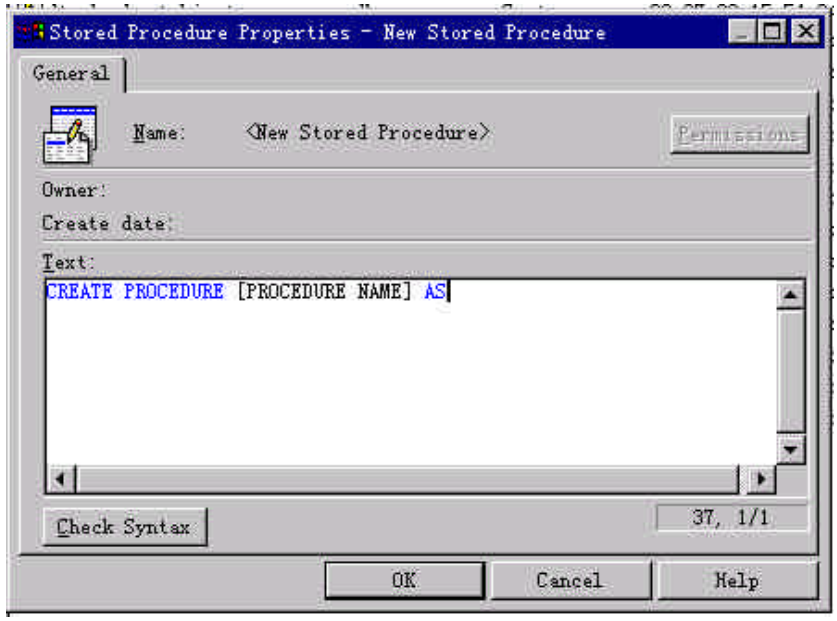

图4-11 建立存储过程

在Stored Procedures Properties对话框的TEXT文本框中输入创建存储过程的T-SQL语句，然后用鼠标单击Check Syntex按钮进行语法检查，检查无误后单击OK按钮，这样，就创建了一个存储过程。

7. 创建触发器

触发器是一种特殊的存储过程，它在插入、删除或修改特定表中的数据时触发执行，它比数据库本身标准的功能有更精细和更复杂的数据控制能力。数据库触发器有以下的作用：

- 安全性：可以基于数据库的值使用户具有操作数据库的某种权利。可以基于时间限制用户的操作，例如不允许下班后和节假日修改数据库数据。可以基于数据库中的数据限制用户的操作，例如不允许股票的价格的升幅一次超过10%。
- 审计：可以跟踪用户对数据库的操作。审计用户操作数据库的语句。把用户对数据库的更新写入审计表。 实现复杂的数据完整性规则。实现非标准的数据完整性检查和约束。触发器可以引用列或数据库对象。提供可变的缺省值。实现复杂的非标准的数据库相关完整性规则。触发器可以对数据库中相关的表进行连环更新。在修改或删除时级连修改或删除其他表中的与之匹配的行。在修改或删除时把其他表中的与之匹配的行设成 NULL值。在修改或删除时把其他表中的与之匹配的行级连设成缺省值。触发器能够拒绝或回退那些破坏相关完整性的变化，取消试图进行数据更新的事务。当插入一个与其主健不匹配的外部键时，这种触发器会起作用。

创建触发器的方法是在要创建的数据库左边单击“ + ”，在该数据库的Tables子文件夹上单击鼠标左键，选择要建立触发器的表，单击鼠标右键，用鼠标指向 Task选项，选择 Mange Triggers...选项，就会弹出Triggers Properties对话框（见图4-12）。

China-pub.com
下载

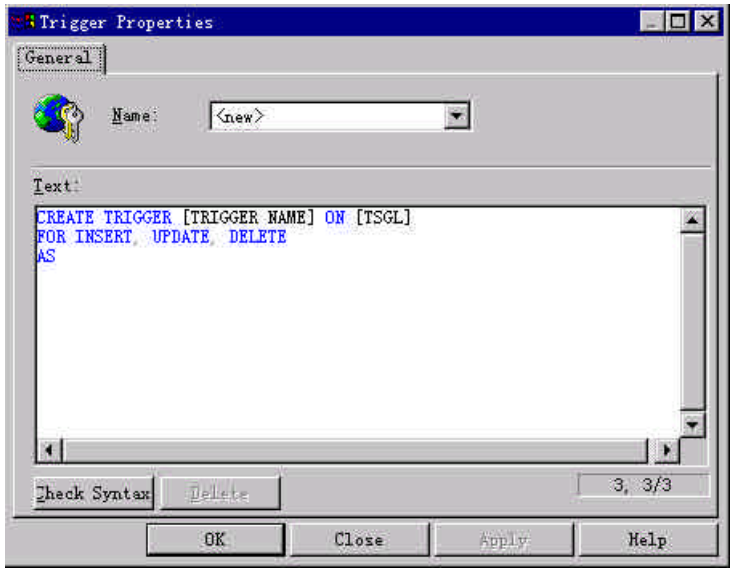

图4-12 建立触发器

在Triggers Properties对话框的TEXT文本框中输入创建触发器的T-SQL语句，然后用鼠标单击Check Syntex按钮进行语法检查，检查无误后单击OK按钮，这样，就创建了一个触发器。

以上是SQL Server 7.0的简单介绍，欲了解SQL Server 7.0的更多内容请参考有关方面的专业书籍。

4.2 结构化查询语言

4.2.1 简介

结构化查询语言（Structured Query Language, SQL）是关系数据库管理系统(Database Management System，DBMS)中的标准语言，已被众多的关系数据库管理系统所采用，如MS SQL Server、Access、Oracle等。事实上，关于SQL语言有一个专门的ANSI标准。目前，所有关系数据库管理系统都打算遵守这一标准，但由于标准SQL命令集的限制太多，不能为一个关系数据库管理系统应用程序所需的所有操作提供支持，很多的关系数据库管理系统又在标准SQL的基础上加入自己的扩展功能，使SQL的功能更加完善。如MS SQL Server支持Transact SQL，简称T-SQL。

4.2.2 SQL语言的功能特点

使用SQL语言，可以从数据库中获取数据、建立数据库和数据库对象，增加数据、修改数据和实现复杂的查询功能。SQL语言广泛地被应用到各种关系数据库管理系统中。SQL是一个非过程化的语言。它允许用户在高层的数据结构上工作，而不对单个记录进行操作，可操作记录集。所有SQL语句接受集合作为输入，返回集合作为输出。SQL的语言特性允许一条SQL语句的结果

China-pub.com 下载

作为另一条SQL语句的输入。

SQL不要求用户指定数据的存放方法，这种特性使用户可以集中精力于要得到的结果。

SQL语言是统一的语言。以前的关系数据库管理系统为实现各种数据操作而定义了各种单独的命令，而SQL将全部的任务统一在一种命令中。由于所有主要的关系数据库管理系统都支持SQL语言，用户可以将用标准的SQL编写的应用程序从一个关系数据库管理系统移植到另一个关系数据库管理系统。

4.2.3 T-SQL

T-SQL为MS SQL Server的SQL语言，在SQL Server 7.0 环境下，用Query Analyzer管理并创建。

1. 创建数据库

语法：

```
CREATE DATABASE database_name
[ ON [PRIMARY]
     [ <filespec> [,...n] ]
     [, <filegroup> [,...n] ]
]
[ LOG ON { <filespec> [,...n]} ]
[ FOR LOAD | FOR ATTACH ]

<filespec> ::=
( [ NAME = logical_file_name, ]
   FILENAME = 'os_file_name'
   [, SIZE = size]
   [, MAXSIZE = { max_size | UNLIMITED } ]
   [, FILEGROWTH = growth_increment] ) [,...n]
<filegroup> ::=
FILEGROUP filegroup_name <filespec> [,...n]
```

参数：

database_name 要创建的数据库名称，在SQL Server中，数据库的名称必须是唯一的。

ON 指明存放数据库的磁盘文件，及其所属文件组（可以省略）。

PRIMARY 定义一个主文件或主文件组，如果PRIMARY缺省，则CREATE DATABASE语句中的第一个数据文件为主文件。

n 一个文件组中的文件个数。

LOG ON 定义一个日志文件及存放日志文件的磁盘文件。

FOR LOAD 为了与早期的SQL SERVER保持兼容。

NAME= logical_file_name 定义数据文件的逻辑文件名。

FILENAME=‘ os_file_name ’ 定义存放数据文件的操作系统文件名及路径。

SIZE 定义数据文件的大小。

MAXSIZE 定义数据文件能增长到的最大值。

China-pub.com
下载

FILEGROWTH 定义数据文件的自动增长值。

语句CREATE DATABASE 需要从master数据库中执行。

下面语句创建数据库 MIS，包含一个名为 MIS_dat的主数据文件及一个名为 MIS_log的日志文件：

```
CREATE DATABASE MIS
ON PRIMARY
(
  NAME = 'MIS_Dat',
  FILENAME = 'D:\MIS\MIS_Dat.mdf',
  SIZE = 5MB,
  MAXSIZE = 10MB,
  FILEGROWTH = 5MB
)
LOG ON
(
  NAME = 'MIS_Log',
  FILENAME = 'D:\MIS\MIS_Log.ndf',
  SIZE = 2MB,
  MAXSIZE = 5MB,
  FILEGROWTH = 20%
)
```

打开Query Analyzer，连接到SQL SERVER服务器，输入以上语句并运行。

2. 修改数据库

语法：

```
ALTER DATABASE database
{  ADD FILE <filespec> [,...n] [TO FILEGROUP filegroup_name]
   | ADD LOG FILE <filespec> [,...n]
   | REMOVE FILE logical_file_name
   | ADD FILEGROUP filegroup_name
   | REMOVE FILEGROUP filegroup_name
   | MODIFY FILE <filespec>
   | MODIFY FILEGROUP filegroup_name filegroup_property
}

<filespec> ::=
  (NAME = logical_file_name
   [, FILENAME = 'os_file_name' ]
   [, SIZE = size]
   [, MAXSIZE = { max_size | UNLIMITED } ]
   [, FILEGROWTH = growth_increment] )
```

参数：

database 欲更改的数据库名。

ADD FILE <filespec> [,...n] [TO FILEGROUP filegroup_name] 为数据库增加数据文件到某文件组。

China-pub.com 下载

ADD LOG FILE 为数据库增加事务日志文件。

REMOVE FILE 为数据库删除其数据文件。

ADD FILEGROUP 为数据库增加数据文件组。

REMOVE FILEGROUP 为数据库删除数据文件组，文件组必须为空才能被删除。

下面语句为MIS数据库添加数据从属文件MIS_TEST.NDF。

```
ALTER DATABASE MIS
ADD FILE
(
 NAME = MIS_TEST,
 FILENAME = 'D:\MIS\MIS_TEST.NDF',
 SIZE = 5MB,
 MAXSIZE = 100MB,
 FILEGROWTH = 5MB
)
GO
```

打开Query Analyzer，连接到SQL SERVER服务器，选择MIS数据库，输入以上语句并运行。

下面语句删除上面添加的数据从属文件：

```
ALTER DATABASE MIS
REMOVE FILE MIS_TEST
GO
```

打开Query Analyzer，连接到SQL SERVER服务器，选择MIS数据库，输入以上语句并运行。

3. 删除数据库

语法：

```
DROP DATABASE database_name [,...n]
```

参数：

database_name 被删除的数据库名。

下面语句删除上面建立的MIS数据库：

```
DROP DATABASE MIS
GO
```

只有DBO 和SA 有DROP DATABASE 语句权限，并且该语句权限不能转交。 DBO必须在MASTER数据库中执行该语句。

4. 创建表

语法：

```
CREATE TABLE
[
   database_name.[owner].
   | owner.
] table_name
(
   {      <column_definition>
          | column_name AS computed_column_expression
```

China-pub.com
下载


```
        | <table_constraint>
    } [,...n]
)
[ON {filegroup | DEFAULT} ]
[TEXTIMAGE_ON {filegroup | DEFAULT} ]
<column_definition> ::= { column_name data_type }
[ [ DEFAULT constant_expression ]
  | [ IDENTITY [(seed, increment ) [NOT FOR REPLICATION] ] ]
]
[ ROWGUIDCOL ]
[ <column_constraint>] [ ...n]
<column_constraint> ::= [CONSTRAINT constraint_name]
{
   [ NULL | NOT NULL ]
   | [    { PRIMARY KEY | UNIQUE }
          [CLUSTERED | NONCLUSTERED]
          [WITH FILLFACTOR = fillfactor]
          [ON {filegroup | DEFAULT} ]]
     ]
   | [    [FOREIGN KEY]
          REFERENCES ref_table [(ref_column) ]
          [NOT FOR REPLICATION]
     ]
   | CHECK [NOT FOR REPLICATION]
          (logical_expression)
}
<table_constraint> ::= [CONSTRAINT constraint_name]
{
   [ { PRIMARY KEY | UNIQUE }
          [ CLUSTERED | NONCLUSTERED]
          { ( column[,...n] ) }
          [ WITH FILLFACTOR = fillfactor]
          [ON {filegroup | DEFAULT} ]
   ]
   | FOREIGN KEY
               [(column[,...n])]
               REFERENCES ref_table [(ref_column[,...n])]
               [NOT FOR REPLICATION]
   | CHECK [NOT FOR REPLICATION]
          (search_conditions)
}
```

参数：

table_name **欲创建的表的名字。**

[ON {filegroup | DEFAULT}] **指明表放在哪个文件组中，如果为** DEFAULT **则放在缺省文件组中。**

TEXTIMAGE_ON **指明存储在某特定数据文件组中的文本和图像，如表中无文本或图像，**

China-pub.com 下载

则不允许使用TEXTIMAGE_ON关键字。

<column_definition> ::= { column_name data_type } 表中列的定义，column_definition为列的名字，在一个表中，列名必须是唯一的。column_name data_type 列的数据类型。

<column_constraint> ::= 列的约束，是可选项

<table_constraint> ::= [CONSTRAINT constraint_name] 表的约束，是可选项。

下面语句在数据库 MIS中创建一个表，名字是 TSGL，该表有 4个列，数据类型分别是SMALLINT、VARCHAR、VARCHAR、DATATIME：

```
CREATE TABLE TSGL
(
  TSGL_ID   SMALLINT
        IDENTITY(1,1)
        PRIMARY KEY CLUSTERED,
  TSGL_MC   VARCHAR(20) NOT NULL,
  TSGL_CBS  VARCHAR(50),
  TSGL_CBRQ DATETIME DEFAULT getdate(),
)
GO
```

定义TSGL_CBRQ时，如果不输入，则用GETDATA()函数返回当前日期。

打开Query Analyzer，连接到SQL Server服务器，选择MIS数据库，输入以上语句并运行。

5. 修改表

语法：

```
ALTER TABLE table
{  [ALTER COLUMN column_name
          {      new_data_type [ (precision[, scale] ) ]
                                [ NULL | NOT NULL ]
                 | {ADD | DROP} ROWGUIDCOL
          }
   ]
   | ADD
          {      [ <column_definition> ]
                 |  column_name AS computed_column_expression
          }[,...n]
   | [WITH CHECK | WITH NOCHECK] ADD
          { <table_constraint> }[,...n]
   | DROP
          {      [CONSTRAINT] constraint_name
                 | COLUMN column
          }[,...n]
   | {CHECK | NOCHECK} CONSTRAINT
          {ALL | constraint_name[,...n]}
   | {ENABLE | DISABLE} TRIGGER
          {ALL | trigger_name[,...n]}
}

<column_definition> ::= { column_name data_type }
```

China-pub.com
下载


```
[ [ DEFAULT constant_expression ]
  | [ IDENTITY [(seed, increment ) [NOT FOR REPLICATION] ] ]
]
[ ROWGUIDCOL ]
[ <column_constraint>] [ ...n]
<column_constraint> ::= [CONSTRAINT constraint_name]
{
   [ NULL | NOT NULL ]
   | [    { PRIMARY KEY | UNIQUE }
          [CLUSTERED | NONCLUSTERED]
          [WITH FILLFACTOR = fillfactor]
          [ON {filegroup | DEFAULT} ]]
     ]
   | [    [FOREIGN KEY]
          REFERENCES ref_table [(ref_column) ]
          [NOT FOR REPLICATION]
     ]
   | CHECK [NOT FOR REPLICATION]
          (logical_expression)
}

<table_constraint> ::= [CONSTRAINT constraint_name]
{  [      { PRIMARY KEY | UNIQUE }
          [ CLUSTERED | NONCLUSTERED]
          { ( column[,...n] ) }
          [ WITH FILLFACTOR = fillfactor]
          [ON {filegroup | DEFAULT} ]
   ]
   |      FOREIGN KEY
                [(column[,...n])]
                REFERENCES ref_table [(ref_column[,...n])]
                [NOT FOR REPLICATION]
   |      DEFAULT constant_expression
                [FOR column]
   |      CHECK [NOT FOR REPLICATION]
          (logical_expression)
}
```

参数：

tabel **欲修改表的名字。**

ALTER COLUMN **指明要修改的列。**

column_name **欲更改列的列名。**

new_data_type **欲更改列的新数据类型。**

ADD **为表增加列。**

DROP **为表删除约束或列。**

下面语句为上面建立的表TSGL**增加一列，类型为**INT**，并且不能为空。**

China-pub.com 下载

```
ALTER TABLE TSGL
ADD TSGL_SH  INT NOT NULL
GO
```

打开Query Analyzer，**连接到**SQL Server**服务器，选择**MIS**数据库，输入以上语句并运行。**

6. **删除表**

语法：

```
DROP TABLE table_name
```

参数：

table_name **欲删除表的名字。**

下面语句删除上面建立的表TSGL**：**

```
DROP TABLE TSGL
GO
```

7. **检索数据**

语法：

```
SELECT select_list
[INTO new_table_]
FROM table_source
[WHERE search_condition]
[GROUP BY group_by_expression]
[HAVING search_condition]
[ORDER BY order_expression [ASC | DESC] ]
```

参数：

select_list **所选择表的列。**

FROM table_source **从那个表进行选择数据。**

WHERE search_condition **所选择表的行。**

SELECT**语句功能强大，可以实现各种各样的数据检索、查询等功能，下面用例子进行详细的说明。**

建立数据库MIS**，其中包含有表**TSGL**，结构为：**

```
TSGL_ID    SMALLINT
        IDENTITY(1,1)
        PRIMARY KEY CLUSTERED,
(表中唯一的ID标示，自动递增，是PRIMARY KEY)
TSGL_MC    VARCHAR(20) NOT NULL,
(图书名称)
TSGL_JQ    SMALLMONEY NOT NULL,
(图书价钱)
TSGL_CBS   VARCHAR(50),
(出版社)
```

建立数据库与表的方法见前面有关建立数据库与表的描述。

检索表TSGL**中的所有数据：**

China-pub.com
下载


```
Select * From TSGL
```

返回表TSGL中的所有数据。

检索表TSGL中的图书名称及出版社这两列数据：

```
Select TSGL_MC,TSGL_CBS From TSGL
```

返回表TSGL中TSGL_MC、TSGL_CBS这两列的所有数据。

检索表TSGL中的图书名称及出版社这两列数据，并改变显示列的名字，使查询结果可读：

Select 图书名称=TSGL_MC,出版社=TSGL_CBS From TSGL

返回表TSGL中TSGL_MC、TSGL_CBS这两列的所有数据，如下表所示：

图书名称	出版社
Microsoft Server 7.0	机械工业出版社
实用组网指南	机械工业出版社
Oracle 8初学教程	机械工业出版社
Linux 自学通	机械工业出版社
数据库编程指南	清华大学出版社
Visual C++速成	清华大学出版社
...	...

检索表TSGL中所有出版社为机械工业出版社的数据：

Select TSGL_MC,TSGL_CBS From TSGL Where TSGL_CBS=“机械工业出版社”

返回如下数据：

TSGL_MC	TSGL_CBS
Microsoft Server 7.0	机械工业出版社
实用组网指南	机械工业出版社
Oracle 8初学教程	机械工业出版社
Linux 自学通	机械工业出版社
...	...

SELECT语句可以用复合条件的检索，下面语句检索出版社为机械工业出版社并且图书名称为实用组网指南的数据：

Select TSGL_MC,TSGL_CBS From TSGL Where TSGL_CBS=“机械工业出版社” AND TSGL_MC=“实用组网指南”

返回如下数据：

TSGL_MC	TSGL_CBS
实用组网指南	机械工业出版社

8. 在表中插入行

语法：

```
INSERT [INTO]
   {
     table_name WITH ( <table_hint_limited> [...n])
```

China-pub.com 下载

```
     | view_name
     | rowset_function_limited
    }

    {       [(column_list)]
            { VALUES ( { DEFAULT
                                        |       NULL
                                        |       expression
                                        }[,...n]
                  )
            | derived_table
            | execute_statement
            }
    }
    | DEFAULT VALUES

<table_hint_limited> ::=
    {       INDEX(index_val [,...n])
            | FASTFIRSTROW
            | HOLDLOCK
            | PAGLOCK
            | READCOMMITTED
            | REPEATABLEREAD
            | ROWLOCK
            | SERIALIZABLE
            | TABLOCK
            | TABLOCKX
    }
```

参数：

table_name **欲插入数据的表。**

Column_lisr **指定新插入的行中只包含部分列值，而不是整行插入，可选。**

VALUES **指定要插入的数据。**

下面语句向表authors**中插入一行数据：**

```
INSERT authors VALUES("123_45","john","214 Main St.","Kent","WA","98000",0
```

9. **更新数据行**

语法：

```
UPDATE
          {
            table_name WITH ( <table_hint_limited> [...n])
          | view_name
          | rowset_function_limited
          }
          SET
          {column_name = {expression | DEFAULT | NULL}
          | @variable = expression
```

China-pub.com
下载


```
        | @variable = column = expression } [,...n]
    {{[FROM {<table_source>} [,...n] ]
        [WHERE
            <search_condition>] }
        |
        [WHERE CURRENT OF
        { { [GLOBAL] cursor_name } | cursor_variable_name}
        ] }
        [OPTION (<query_hint> [,...n] )]
```

参数：

table_name **欲更新的表名。**

SET **指定列及改变后的值。**

WHERE **指定要更新的行。**

下面语句把表TSGL**中的**TSGL_CBS**字段全部更新为“机械工业出版社”：**

Update TSGL SET TSGL_CBS=**“机械工业出版社”**

10. **删除数据行**

语法：

```
DELETE
    [FROM ]
        {
          table_name WITH ( <table_hint_limited> [...n])
         | view_name
         | rowset_function_limited
        }

        [ FROM {<table_source>} [,...n] ]
    [WHERE
        { <search_condition>
        |     { [ CURRENT OF
                    {
                        { [ GLOBAL ] cursor_name }
                        | cursor_variable_name
                    }
                ]
        }
    ]
    [OPTION (<query_hint> [,...n])]
```

参数：

DELETE [FROM] table_name **从指定的表中删除一行或多行数据。**

WHERE **指定删除的条件。**

下面语句在表TSGL**中删除所有**TSGL_CBS**字段为“机械工业出版社”的记录**

```
Delete from TSGL Where TSGL_CBS="机械工业出版社"
```

China-pub.com 下载

4.2.4 深入了解T-SQL结构化查询语言

上面所描述的查询都是单一的查询，实际上，在T-SQL中，查询语句是可以被嵌套的，即一个SELECT语句被嵌套在另一个T-SQL语句中，这个语句称为子查询。子查询有两种类型：简单型和关联型。一般情况下，子查询用于另一个 SELECT、INSERT、UPDATE或者DELETE语句中的WHERE或HAVING短语中；也可用于另一个子查询甚至作为一个表达式。

1. 简单子查询

包含一个简单子查询的SELECT语句如下面程序语句所示。

```
SELECT   title,price
FROM    titles
WHERE   title_id IN
     (SELECT   title_id
      FROM     sales
      WHERE    qty>30)
```

该语句返回售出册数大于30本的所有图书列表。用户使用pubs数据库时可以得到四行数据。只有title_id包含在sales表中且售出册数大于30本的书名才会返回。该列表作为将要返回的书名列表的基准。用户也可采用另外一种方法获得相同的结果；尽管程序清单和上面的查询语句截然不同，但二者在语义上是相同的。

使用EXISTS短语进行简单子查询

```
SELECT      title,price
FROM        titles
WHERE      EXISTS
           (SELECT  *
            FROM     sales
            WHERE    sales.title_id = titles.title_id
            AND      qty>30)
```

上面两条查询语句的结果都是：

```
title                                                                  price
-------------------------------------------------------------------------------
You Can Combat Computer Stress!                                        2.9900
Secrets of Silicon Valley                                                     20.0000
Is Anger the Enemy?                                                    10.9500
Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean        20.9500

(4 row(s) affected)
```

在内部，SQL Server对这两个子过程的处理是相同的。用户可以通过其它方式获得相同的结果，例如使用表联结代替子查询，可以返回相同的数据集，并且能控制服务器的处理过程。

2. 联结和子查询

考察如下的SELECT语句及其查询计划。

下面语句使用联结在Sales中选择titles

```
SELECT      title,price
```

China-pub.com
下载


```
FROM      titles JOIN sales ON sales.title_id = titles.title_id
WHERE     sales.qty>30
```

上面语句的的结果

```
title                                                                          price
-------------------------------------------------------------------------------
Secrets of Silicon Valley                                                      20.0000
Is Anger the Enemy?                                                  10.9500
Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean    20.9500
You Can Combat Computer Stress!                                       2.9900

(4 row(s) affected)
```

返回的结果与上面两条查询语句的执行结果相同，但顺序不同。因为语义有很大差别，所以需对它们作进一步讨论。

大多情况下，在SELECT语句中的联结比一个等效子查询更容易，效率更高。本书推荐尽可能地使用联结来代替子查询以取得到较好的性能。使用索引会大大增加 T-SQL语句的I/O开销。但要确保两个表中的关键字段（被联结的列）进行索引时能够改善查找性能。

到目前为止，使用子查询好象没有什么好处，那么为什么还要使用它们呢？因为现实中确实存在子查询能工作但联结的SELECT语句就不能完成这项任务。

下面语句选择出大于平均销售数量的书名

```
SELECT      titles.title,titles.price
FROM        titles JOIN sales ON sales.title_id = titles.title_id
WHERE       sales.qty>
        (SELECT AVG(qty)
        FROM sales)
```

为什么必须使用子查询？简单而言，是因为用户不能比较聚合值 —— 本例中的AVG(qty)，以及非聚合值 —— 本例中的qty。在执行SELECT语句时，如果用户选择平均数量和数量，就必须在第二个数量上使用GROUP BY（因为该值不是聚合的），这样反过来寻找平均数量的过程又会被阻止。

下面语句从sales表中选择qty和AVG(qty)

```
SELECT      qty,AVG(qty) AS avgqty
FROM        sales
GROUP BY  qty
```

注意avgqty列和qty列总是相等的。这是由于用户需要使用 qty列进行分组，从而仅能看到每个单独的数量的平均数量。3的平均值是3，3和3的平均值仍是3，如此等等。

3. 关联子查询

一个关联子查询在其执行过程中领先于主查询。在本质上一个关联子查询不能单独存在。如果用户将简单和关联两种子查询与英语语法相对比的话，一个简单子查询相当于一个独立的分句，一个关联子查询相当于从句。因此用户会经常看到主查询中的一列被该子查询所引用。

China-pub.com 下载

一个关联子查询

```
SELECT      DISTINCT au_lnanme,au_fname
FROM        authors
WHERE      100 IN
        (SELECT royaltyper
        FROM titleauthor
        WHERE titleauthor.au_id = authors.au_id)
```

我们注意到子查询是如何引用来自主查询一列的。子查询不能单独存在的原因是，在查询执行过程中主查询为au_id提供值。通常大多数关联子查询都能用联结来代替。而在很多情况下，联结更容易读懂且内部执行效率也更高些。那我们为什么还要使用关联子查询呢？就像上面语句所示，有时只有用子查询才能做。但最好尽量使用联结，用户不仅能获得很快的速度，而且返回来看自己的代码时会更易理解。

4. 更高级的DELETE和UPDATE语句

使用DELETE和UPDATE的简单方式是：根据某一列值来删除和更新一行。当估计出应该删除或更新什么数据时，就像SELECT语句那样，DELETE和UPDATE能够使用联结和子查询，这是一种不常用的方式，但在修改或删除数据时有一些很强大的功能。

注意，练习使用DELETE和UPDATE时，要确保将DELETE和UPDATE语句放在BEGIN TRAN和ROLLBACK TRAN之间；以便所作的改变不是永久性的。

考虑如下简单的DELETE语句：使用所提供的title_id将相应行从sales表中删除。

从sales表中删除行

```
DELETE   sales
WHERE   title_id = 'BU1032'
```

该语句根据值为“BU1032”的rirle_id将相应两行从sales表中删去。但如果用户想要删除某一特定出版商相关联的那些书名，请看下面提供的例子。

通过id删除某个出版商的sales

```
DELETE      sales
FROM        sales
        JOIN titles t ON sales.title_id = t.title_id
WHERE      t.pub_id = '1389'
```

另外，如果用户希望根据出版商姓名进行删除时，可以使用下面语句。

通过姓名来删除某个出版商的sales

```
DELETE      sales
FROM        sales
        JOIN titles t ON sales.title_id = t.title_id
        JOIN publishers p ON t.pub_id = p.pub_id
WHERE      p.pub_name = 'Algodata infosystems'
```

可见，在DELETE语句中执行联结操作可以找到并删除那些与其他表中的数据相匹配的行。联结需要进行比较操作并且限制了在sales表中将要删除的行数，也就是说，只有当找到一个匹

China-pub.com
下载

配的书名（根据某个出版商的已知信息）时，才删除 sales。

用户也可以使用子查询来限制表中要删除的行。下面语句与上一个例子得到相同的结果，只不过是用子查询代替联结而已。

使用子查询删除某出版商的sales

```
DELETE      sales
WHERE      title_id IN
        (SELECT title_id from titles
         WHERE pub_id = '1389')
```

注意，尽管该查询效率校低，但可读性好。通常情况下，用户应该尽可能地使用联结，但也有例外的情况，如下面的语句所示。

根据sales的平均数量删除sales

```
DELETE      sales
WHERE      qty<
        (SELECT AVG(qty) from sales)
```

该查询删除所有销售量小于 sales表中的平均数量的 sales。不存在使用联结完成同样任务的合理的DELETE语句。子查询在这种情况下不仅十分有用，而且也是必须使用的。

改进的新UPDATE语句

实际上，UPDATE语句对于T-SQL不是什么新内容。但正像 DELETE语句，UPDATE语句可以使用联结和子查询对需要更新的数据加以限制。它能使用联结或子查询限制要更新的行；而且也能使用子查询作为更新值的表达式。

就像使用其他的查询那样，用户可以使用子查询限制想要更新的行。

更新畅销书的特权

```
UPDATE      titleauthor
SET          royaltyper = royaltyper * 1.1
WHERE       title_id IN
         (SELECT title_id
         FROM  sales
         GROUP BY title_id
         HAVING  sum(qty)>=30)
```

读者可以看到，那些总销量大于 30的所有图书的royaltyper增加10%。并且，因为比较中涉及到使用总计来确定哪本书销售最好，所以只能使用子查询。

使用子查询分配更新值

考虑使用子查询的结果作为某列的更新值的情况，上面代码将 titleauthor表中的royaltyper列更新为指定书的销售总量。尽管这在生活中可能不太现实，但却说明了子查询在更新操作中的能力。

进一步考虑：将royaltyper更新为比那些销售量不小于 30的所有书的总销售量大 10%。见下面程序语句所示

根据总销售量更新royaltyper

China-pub.com 下载

```
UPDATE     titleauthor
SET         royaltyper = (SELECT SUM(qty)
                FROM   sales
                WHERE  sales.title_id = titleauthor.title_id)
```

使用两个子查询更新royaltyper

```
UPDATE     titleauthor
SET         royaltyper = 1.1 * (SELECT SUM(qty)
                FROM    sales
                WHERE  sales.title_id = titleauthor.title_id)
WHERE     title_id IN
       (SELECT    title_id
       FROM       sales
       GROUP BY   title_id
       HAVING     sum(qty)>=30)
```

注意，如果用户确实要在不使用BEGIN TRAN和ROLLBACK TRAN的情况下运行这些数据修改语句，那么可以选择SQL Server的\mssq17\install\目录下的instpubs.sql脚本运行。该脚本会重新创建原始pubs数据库。用户必须确保是以sa登录的，否则将得到非期望的结果。

5. 聚合函数

SQL Server中有几种函数称为聚合函数（见表4-1）。聚合函数对值集进行计算，并给用户返回一个单值。它们可以对表中的所有行计数，确定表中的最大或最小值，或者确定表中所有值的平均值。聚合函数只能用在SELECT短语、COMPUTE或COMPUTE BY短语或者HAVING短语中。

表4-1 聚合函数表

函 数	动 作
AVG	返回组中所有非空值的平均值
COUNT	返回组中所有数据项的数目
MAX	返回组中所有值的最大值
MIN	返回组中所有值的最小值
SUM	返回列表中所有非空值的总和，SUM只能用于数字数据类型的表达式
STDEV	返回表达式中所有值的标准方差
STDEVP	返回表达式中所有值的人口标准方差
VAR	返回表达式中所有值的方差
VARP	返回表达式中所有值的人口方差

AVG 该函数返回指定组中所有值的平均值。在使用AVG函数时，NULL值被忽略。函数的语法如下：

```
AVG([ALL I DISTINCT] expression)
```

参数	描述
ALL	参数ALL强制SQL Server对组中的所有值作函数操作，它是默认值。
DISTINCT	参数DISTINCT强制SQL Server使对每个值的单个实例作函数操作，而

China-pub.com
下载

忽略该值的任何重复实例。

expression expression是要考虑的值的列表，它们必须是数字数据类型。

COUNT 该函数返回指定值中所有值的数目。函数的语法如下：

```
COUNT({[ALL I DISTINCT] expression I * })
```

参数	描述
ALL	参数ALL强制SQL Server对组中的所有值作函数操作，它是默认值。
DISTINCT	参数DISTINCT强制SQL Server使对每个值的单个实例作函数操作，而忽略该值的任何重复实例。
expression	expression是要考虑的值的列表，其类型可以是 uniqueidentifier、text、image或者ntext以外的任何数据类型。
*	表明表中的所有行都要计数。

MAX 该函数返回指定组中所有值中的最大值。函数语法如下：

```
MAX([ALL I DISTINCT] expression)
```

参数	描述
ALL	参数ALL强制SQL Server对组中的所有值作函数操作，它是默认值。
DISTINCT	参数DISTINCT强制SQL Server使对每个值的单个实例作函数操作，而忽略该值的任何重复实例。
expression	expression是要考虑的值的列表，其类型是除位数据类型以外的任何数据类型。

MIN 该函数返回指定组中所有值中的最小值。函数语法如下：

```
MIN([ALL I DISTINCT] expression)
```

参数	描述
ALL	参数ALL强制SQL Server对组中的所有值作函数操作，它是默认值。
DISTINCT	参数DISTINCT强制SQL Server使对每个值的单个实例作函数操作，而忽略该值的任何重复实例。
expression	expression是要考虑的值的列表，其类型是除位数据类型以外的任何数据类型。

SUM 该函数返回指定组中所有值的和。函数语法如下：

```
SUM([ALL I DISTINCT] expression)
```

参数	描述
ALL	参数ALL强制SQL Server对组中的所有值作函数操作，它是默认值。
DISTINCT	参数DISTINCT强制SQL Server使对每个值的单个实例作函数操作，而忽略该值的任何重复实例。
expression	expression是要考虑的值的列表，其类型是除位数据类型以外的任何数

China-pub.com 下载

字数据类型。

STDEV 该函数返回指定组中所有值的标准方差，函数的语法如下：

```
STDEV(expression)
```

参数	描述
expression	expression是考虑的值的列表，其类型是除位数据类型以外的任何数字数据类型。

STDEVP 该函数返回指定组中所有值的人口标准方差，函数语法如下：

```
STDEVP(expression)
```

参数	描述
expression	expression是把考虑的值的列表，其类型是除位数据类型以外的任何数字数据类型。

VAR 该函数返回指定组中所有值的方差，函数语法如下：

```
VAR(expression)
```

参数	描述
expression	expression是把考虑的值的列表，其类型是除位数据类型以外的任何数字数据类型。

VARP 该函数返回指定组中所有值的人口方差，函数语法如下：

```
VARP(expression)
```

参数	描述
expression	expression是把考虑的值的列表，其类型可以是除位数据类型之外的任何数字数据类型。

6. GROUP BY和HAVING短语

GROUP BY和HAVING短语应用于SELECT语句中，提供数据的特别分组，尤其是在使用聚合函数时使用它们。

GROUP BY 该短语在SELECT语句中使用，它指定外部数据将要进入的分组；当存在聚合函数时，它将计算每个组的总值，GROUP BY短语的语法如下：

```
[GROUP BY [ALL] group_by_expression][WITH{CUBE | ROLLUP}]
```

参数	描述
ALL	当ALL选项被指定时，将返回所有的行和组，也包括不满足SELECT语句中的 WHERE短语所提要求的那些行。当使用CUBE和ROLLUP操作时，不能指定ALL选项。
Group_by_expression	group_by_expression是要分组的结果所在的表达式的名字。
WITH CUBE	当WITH CUBE操作被指定时，累计数据会加进结果集。这个累计数据用来表示每一个组或子组的每种可能组合。

China-pub.com
下载

WITH ROLLUP	ROLLUP操作用来将结果集中提供累计数据。按从最低层到最高层的分层顺序对组进行累计。

HAVING 该短语对组或者聚合函数指定搜索条件。HAVING短语类似于WHERE语句。当HAVING短语不和GROUP BY或者聚合函数一起使用时，HAVING短语实际上和WHERE短语一样。HAVING短语的语法如下：

[HAVING<*search_condition*>]

参数	描述
<search_condition>	<search_condition>是与分组和聚合函数相一致的任何条件。

7. 合并

UNION是将两个或多个查询全并到一个结果集中的方法，合并后的结果集包含并组中的所有查询。它可以用来结合属于不同表的数据。例如，在一个订货处理系统中，每个地区都包含一个表，这些表都包含在该系统中，公司在每个地区都有库存。考虑这种情况，用户要处理密西西比河东西两岸的库存，就要生成两个表：stores_east和stores_west。要获得所有库存清单，用户就必须运行如下的查询：

```
SELECT * FROM stores_east
UNION
SEECT * from stores_west
```

使用UNION短语要遵循以下两个规则：

1）在所有的查询中列的数目必须相同。如果不相同，用户必须为缺少的列提供明确的NULL值。

2）所有的数据类型必须兼容。这并不意味着所有的数据类型必须相同，而是经过隐含地转换能够兼容即可。

4.2.5 SQL语言运用技巧

根据编者使用经验和对刚接触SQL语言的人员了解，我们往往最关心的是语句是否能够顺利执行且达到预期效果。为了体现本书以方便用户为原则，现将一些在实际工作中遇见的问题和解决方法，作如下解答。

1. 怎样列举出SQL Server中已存在的数据库？

答：
```
USE MASTER
SELECT * FROM sysdatabases
GO
```

MASTER数据库中的sysdatabases表中记录着SQL Server当前所有数据库。

2. 怎样用一个表中的几个字段且不要重复记录做成一个新表？

答：
```
SELECT DISTINCT jdc_cx,jdc_cph INTO new_table FROM jdcgl
```

选择jdcgl表中的jdc_cx,jdc_cph字段排除重复记录作为生成新的new_table表中字段。

China-pub.com 下载

3. 怎样向临时表中动态添加一个时间字段，并以当前时间填充？

答：
```
ALTER TABLE jdcgl ADD jdc_sj DATETIME DEFAULT getdate() NOT NULL
```

像jdcgl表中添加jdc_sj时间字段，并以当前时间填充，此字段不能为空。

4. 怎样得知表中某字段记录的长度？

答：
```
SELECT LEN(tsgl_mc) FROM tsgl WHERE tsgl_id=2
```

可用LEN函数是获取记录长度，这条语句表示获取 TSGL表中tsgl_mc字段ID号等于2的记录长度。

5. 怎样在程序中限制不显示记录前后有多余的空格？

答：
```
SELECT LTRIM(RTRIM(tsgl_mc))FROM tsgl
```

SQL Server提供了LTRIM，RTRIM两个函数分别去除左，右的多余空格。

6. 现需要删除小于某天的所有数据，但是 SQL Server 7上的字段类型关于日期时间的好象只有datetime类型没有date类型, 用SQL该怎样删除？

答：
```
DELETE FROM tsgl WHERE DATEDIFF(DAY,tsgl_cbrq,'2000-07-21')>0
```

用DAY取得天数，DATEDIFF函数算出时间差，小于此天数的就做删除处理，两个函数共用便可很巧妙地达到预期效果。

7. 用Mssql7建一表table1，有一个字段field0(类型为char)，其数据如下：

```
field0
 a11
 a12
 a13
```

如何使字段field0的数据变为：

```
field0
 a 1 1
 a 1 2
 a 1 3
```

如何写SQL语句?

答：
```
UPDATE table1 SET field0=SUBSTRING(field0,1,1)+' '
  +SUBSTRING(field0,2,1)+' '+SUBSTRING(field0,3,1)
```

SQL Server提供的SUBSTRING函数用于截取字符串，此SQL语句含义是将field0字段中记录的字符分别取出赋值后进行替换。

8. 表a有一个字段id，表b有一个字段id，表a有以下记录:

1,2,3,4,5,6；表b有以下记录：2,4,6；如何在表a中查出表b中没有的记录？

答：
```
SELECT * FROM a WHERE id NOT IN (SELECT id FROM b)
```

用NOT IN来限定子查询，这样便可获得表 B中没有的记录。

9. 用Select INTO语句生成新表时提示Server: Msg 268, Level 16, State 3, Line 3

```
Cannot run SELECT INTO in this database.
The database owner must run sp_dboption to enable this option.
```

China-pub.com
下载

如何解决？

答：
```
sp_dboption 'MIS', 'SELECT INTO', 'TRUE'
```

sp_dboption是系统提供的存储过程，用语对数据库使用者赋予或撤消某些权限 MIS 是需要管理的 database，“ select into ” 是你想要打开的数据库选项TRUE是打开，如果想关掉的话请用“ FALSE ”。

10. 如何实现SQL Server调用外部程序？

答：
```
EXEC master.dbo.xp_cmdshell 'ipconfig'
```

可以用MASTER数据库Extended Stored Procedures中的xp_cmdshell来实现。

11. 存储过程的问题

例子：

```
CREATE PROCEDURE sp_temp
AS
DECLARE @tmpstr char(10)
SELECT @tmpstr = "abcd" + "%"
SELECT @tmpstr
SELECT * from mytable
where myzd like @tmpstr
GO
```

问题：执行的 Select语句出不了正确的结果，但如果把 (6)行的like @tmpstr 替换成like “ abcd% ”却可以得到正确结果，为什么？请各位高手帮帮忙，谢谢。

答：用下面的语句：试试便知道怎么回事。

```
DECLARE @tmpstr char(10)
SELECT @tmpstr = 'abcd'+'%'
PRINT '['+ @tmpstr + ']'
```

很明显CHAR是空格填充的。所以，正确的写法应该是将 CHAR类型换成VARCHAR试试便知道怎么回事了。 你不知道char是空格填充的？所以，正确的写法应该是：

```
CREATE PROCEDURE sp_temp
AS
DECLARE @tmpstr VARCHAR(10)将CHAR类型换成VARCHAR
SELECT @tmpstr = "abcd" + "%"
SELECT @tmpstr
SELECT * from mytable
where myzd like @tmpstr
GO
```

12. 如何实现表aa有一字段aaid, 表bb也有一字段bbid,现在要将aa表的一条记录删除后自动实现将bb表中凡bbid的值与被删除的aaid相同的记录。

答：可用触发器来实现，代码如下：

```
CREATE TRIGGER test
```

China-pub.com 下载

```
ON aa
FOR DELETE
AS
  DELETE FROM bb JION deleted ON (bb.bbid=deleted.aaid)
```

13. 如何实现用存储过程两个参数，一个作为要插入的数，另一个作为要插入的个数。应该如何做？

答：在存储过程中用一个循环来实现，具体程序如下：

```
CREATE PROC usp_insert_tsgltmp
   @ins_num VARCHAR(10),
   @ins_con VARCHAR(10)
AS
  DECLARE @tmp_num INT
  SET @tmp_num=1
  WHILE @tmp_num<=CONVERT(INT,@ins_con)
   BEGIN
  BEGIN TRAN
    INSERT INTO tsgltmp(tmp_je) VALUES(CONVERT(SMALLMONEY,@ins_num))
    IF @@ERROR=0
       COMMIT TRAN
    ELSE
       ROLLBACK TRAN
 SET @tmp_num=@tmp_num+1
   END
```

@@ERROR每条语句后都应该检查一下。INSERT出错后，应该IF @@ERROR<>0 BEGIN … …END，否则当DELETE FROM执行后，如果DELETE不出错，就算前一条insert出错，@@error一样会被置0的。

14. 有一台SQL服务器，现将该数据库备份，然后抄到另一SQL服务器上，进行恢复。假如另一台SQL服务器没有第一台服务器数据库所在磁盘分区时应该怎样做？

答：在恢复的时候使用 MOVE TO 选项

```
RESTORE DATABASE database_name' FROM 'backup_file'
WITH MOVE'logical_file_name' TO 'operating_system_file_name'
```

15. SQL Server的推荐内存容量为多少？

答：Microsoft SQL Serve最多允许使用2GB虚拟内存。Windows NT 为每个32位Windows应用程序提供4GB的虚拟内存空间， 这4GB内存中低端的2GB是这个进程的私有空间，高端的2GB为系统所用。

Windows NT虚拟内存管理器(VMM)将4GB地址空间映射到实际物理内存中。实际物理内存最多可为4GB，这取决于硬件平台。

像SQL Server这样的32位Windows应用程序只识别虚拟或者说逻辑地址，而不是物理地址。在某一确定时间一个应用程序到底可以使用多少物理内存取决于实际物 理内存大小及VMM。应用程序不能直接控制内存的分配。

China-pub.com
下载

Windows NT的虚拟地址系统允许虚拟内存与实际物理内存之比大于 1:1。 因此，大型应用程序可以在各种硬件系统配置条件下运行。然而，在绝大多数情况下，如果系统虚拟内存远远大于运行的所有应用程序所用的系统内存总和，系统性能将大打折扣。

16. 何种原因导致错误日志中出现17824，17832和1608错误信息？

答：各种与通信相关的Microsoft SQL Server错误都有可能。一般来说，这些错误信息并不意味着SQL Server错误， 而是网络、网络配置和客户端应用程序错误。 无论在服务器端还是在客户端， SQL Server和它的应用程序处于ISO网络层以上的层次。 建立和维护一条可靠的网络连接责任属于网络层和SQL Server之下的系统层次。

可能的错误包括:

服务器端错误

17832 不能读取登录包

17852 不能关闭服务器端的连接

17824 不能向服务器端的连接写入数据

10058 套接字关闭后不能发送数据

10054 对端连接重置

10053 软件导致连接失效

1608 当发送结果至前端时发生一个网络错误

232 管道被关闭

109 管道被中止

客户端错误

10008 从SQL Server端来的损坏令牌: 数据流处理无法同步

10010 无法从SQL Server读取数据

10018 关闭网络连接时出错

10025 无法向SQL Server写数据

17. 为何将一个在ISQL/W中可执行正常的脚本写成 stored procedure后，执行此stored procedure就会有问题，并且出现：Msg 202, Level 11, State 2 “Inte rnal error -- Unable to open table at query execution time.”的出错信息，如何解决?

答：如果此stored procedure包含了下列各项：

1）以CREATE TABLE建permanent table。

2）Drops在此stored procedure中建立的permanent table。

3）建temporary table。

4）在temporary table中宣告cursor。

5）Drops在此stored procedure中建立的temporary table。

解决方法为：采取下列任一方式，重建 stored procedure。

1）使用“ SELECT INTO ” statement建permanent table。

2）避免使用“ DROP TABLE ”，或是只drop permanent table或只drop temporary table。

China-pub.com 下载

3）将DECLARE CURSOR statement以括号包含在EXEC()中。

4）建立stored procedure时使用WITH RECOMPILE参数或是执行时使用 WITH REC OMPILE选项。

在上面章节的介绍中，我们主要讲解了数据库系统SQL Server 7.0 的基本概念及其操作方法。在SQL Server 7.0中，我们可以用T-SQL语言实现数据库操作的所有功能，包括建立库文件表，对库文件及表的修改删除查询等基本操作。在下面的章节和例子中，我们将看到在 ASP中使用何种方法去操纵数据库，以及如何应用T-SQL语言等内容。