

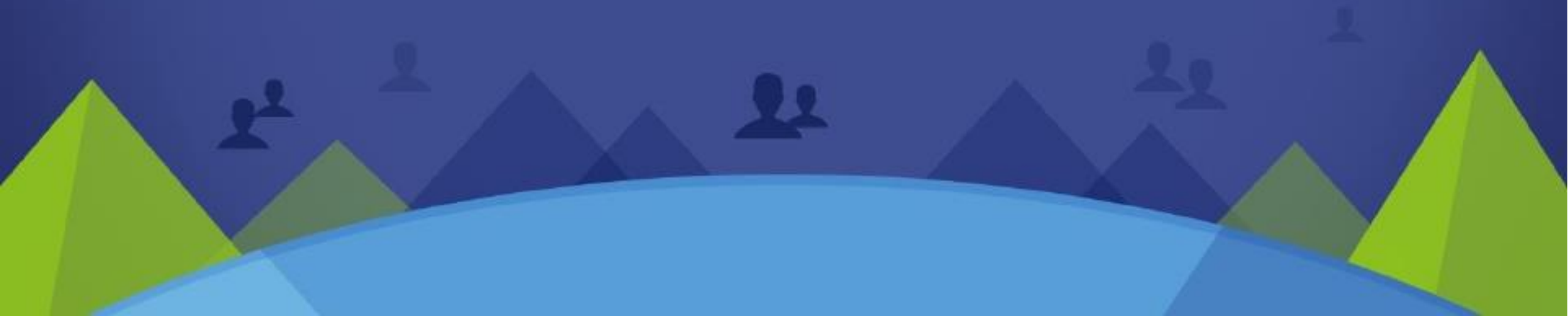
数据库原理及应用教程 (第4版)

“十二五”普通高等教育本科国家级规划教材
微课版，对重点和难点进行视频详解

中国工信出版集团

人民邮电出版社

第3章 关系数据库标准语言——SQL



3.1 SQL的基本概念与特点

3.2 SQL Server 2012简介

3.3 数据库的创建和使用

3.4 数据表的创建和使用

3.5 单关系（表）的数据查询*

3.6 多关系（表）的连接查询*

3.7 子查询*

3.8 其他类型查询*

3.9 数据表中数据的操纵

3.10 视图

3.11 创建与使用索引

3.12 小结

结构化查询语言
Structured Query Language

数据定义

数据查询

数据操纵

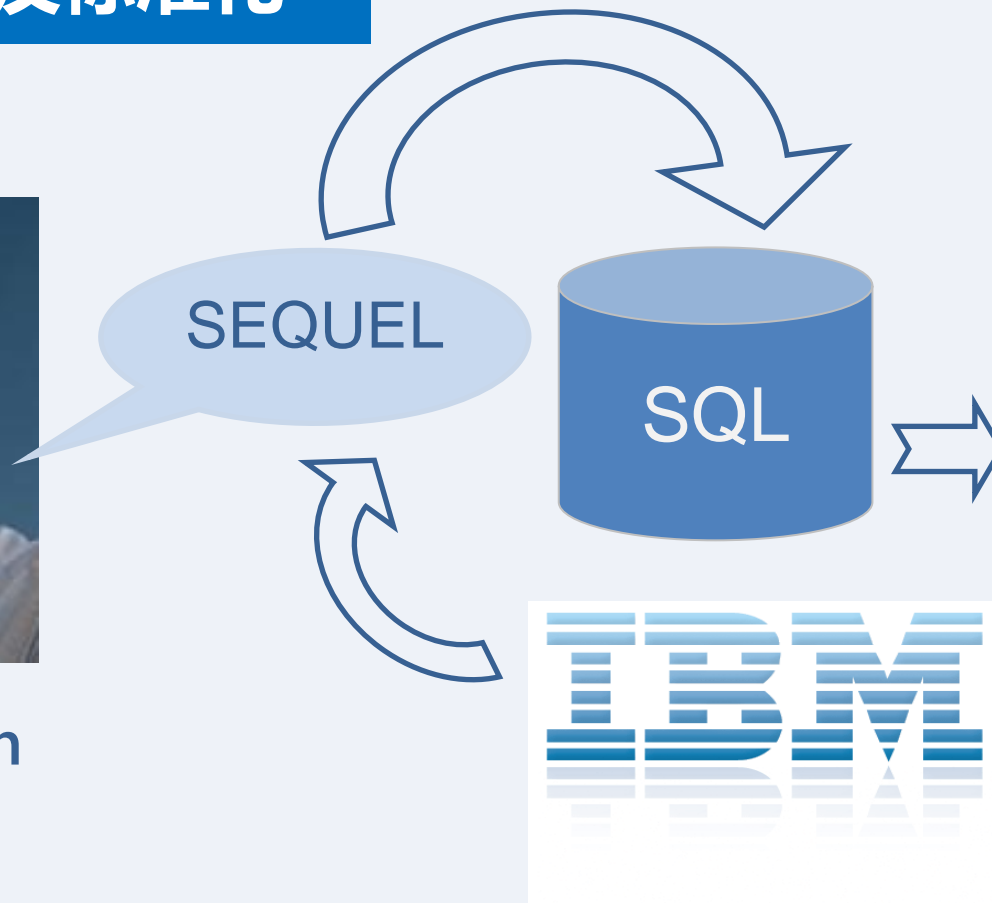
数据控制

3.1.1 SQL的发展及标准化

– SQL的发展



Chamberlin



大型数据库

Sybase
INFORMIX
SQL Server
Oracle
DB2
GaussDB
OceanDB

小型数据库

FoxPro
Access

3.1.2 SQL的基本概念

基本表 (Base Table)

一个关系对应一个基本表

一个或多个基本表对应一个存储文件

视图 (View)

视图是从一个或几个基本表导出的表，是一个虚表

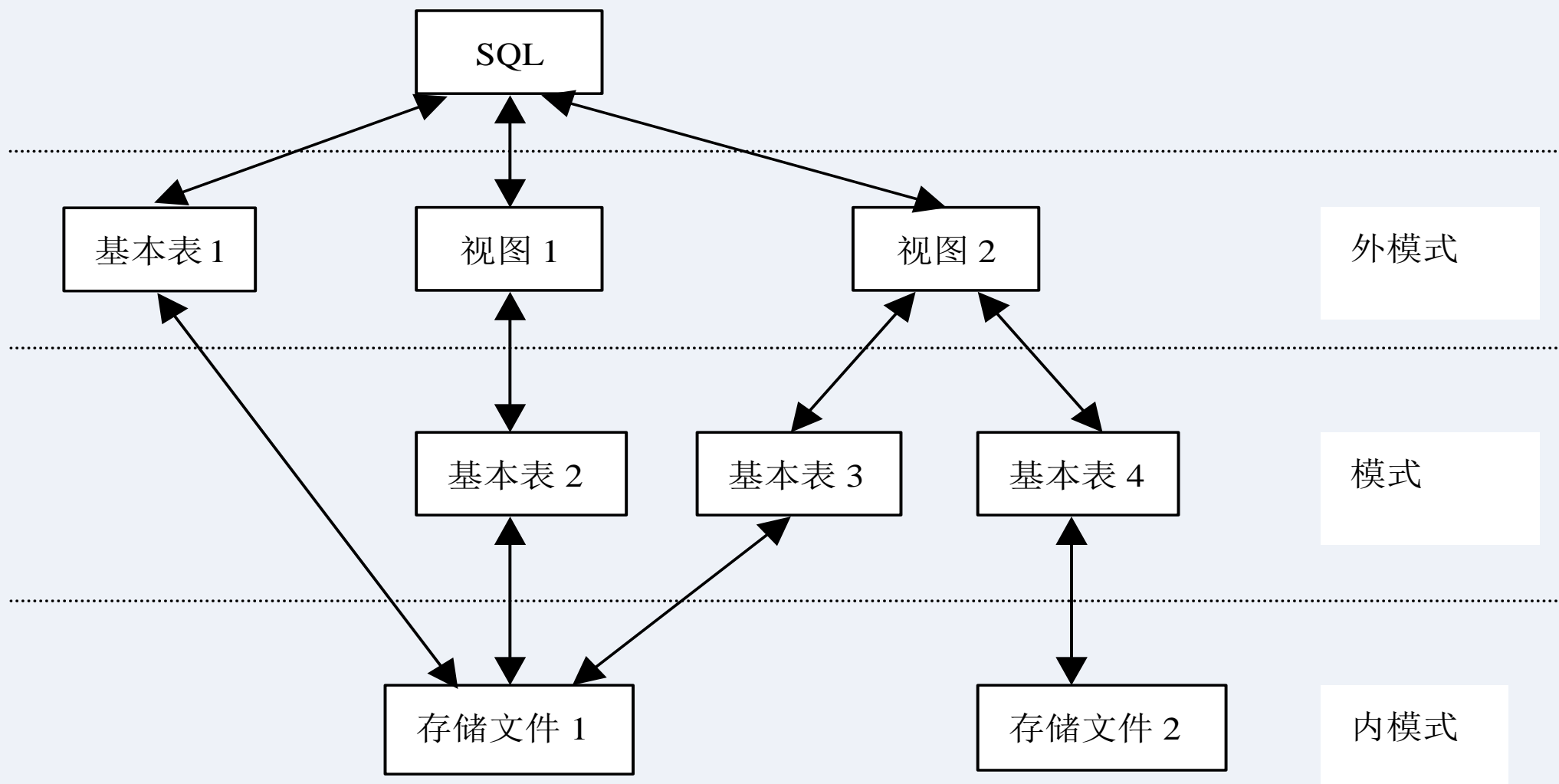
S(SNo, SN, Sex, Age, Dept)



S_Male(SNo, SN, Age, Dept)

无数据，只有定义

在数据库中只存有
S_Male的定义，数
据仍在**S**表中



SQL语言支持的关系数据库的三级模式结构

3.1 SQL语言的基本概念与特点

3.1.3 SQL的主要特点

SQL语言是类似于英语的自然语言，简洁易用

SQL是一种一体化的语言

SQL语言是一种非过程化的语言

SQL语言是一种面向集合的语言

SQL语言既是自含式语言，又是嵌入式语言

**SQL语言具有数据查询、数据定义、
数据操纵和数据控制四种功能**

3.2.1 SQL Server 的发展与版本

SQL Server是一个支持关系模型的关系数据库管理系统

企业版 (Enterprise Edition)

标准版 (Standard Edition)

Web版 (Web Edition)

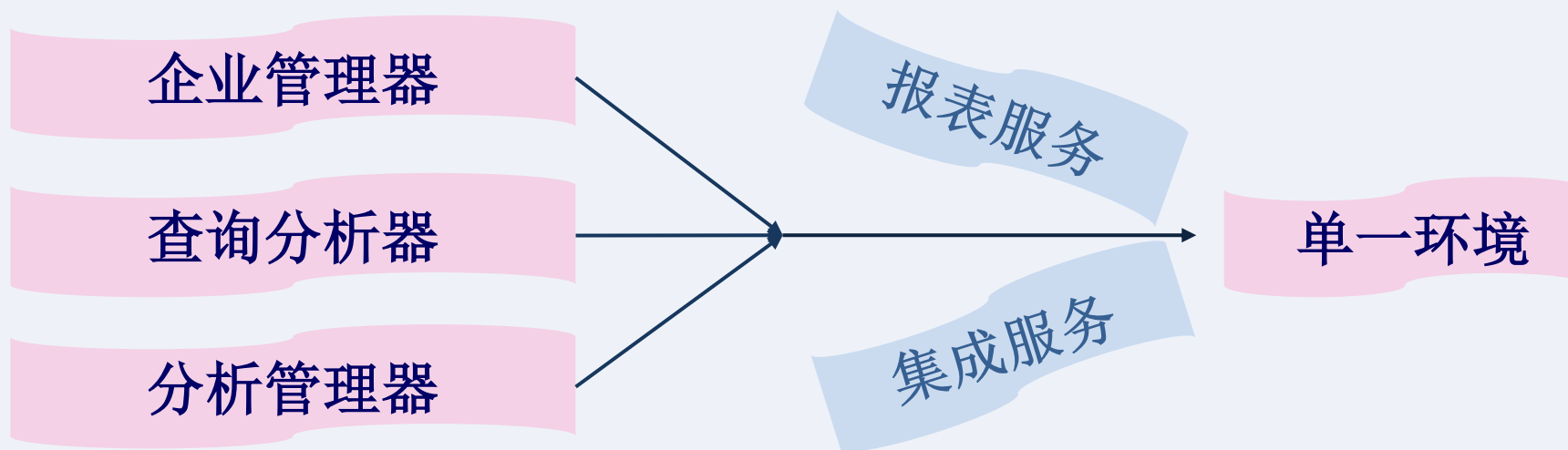
开发者版 (Developer Edition)

3.2.2 SQL Server 2012 的主要组件

组 件	功 能
SQL Server数据库引擎	存储、处理和保护数据的核心引擎，复制，全文搜索以及用于管理关系数据和XML数据的工具。
SQL Server Management Studio	集成环境，用于配置和管理SQL Server的主要组件。
分析服务	创建和管理联机分析处理及数据挖掘应用的工具。
报表服务	开发报表应用程序的可扩展平台，用于创建、管理和部署表格报表、矩阵报表、图形报表以及自由格式报表等应用。
集成服务	一组图形工具和可编程对象。
配置管理器	为SQL Server服务、服务器协议、客户端协议和客户端别名提供配置管理。
数据库引擎优化顾问	协助创建索引、索引视图和分区的最佳组合
商业智能开发向导	集成开发环境，集成了上述分析服务、报表服务和集成服务的功能。
连接组件	安装客户端和服务端通信的组件
联机丛书	查询信息

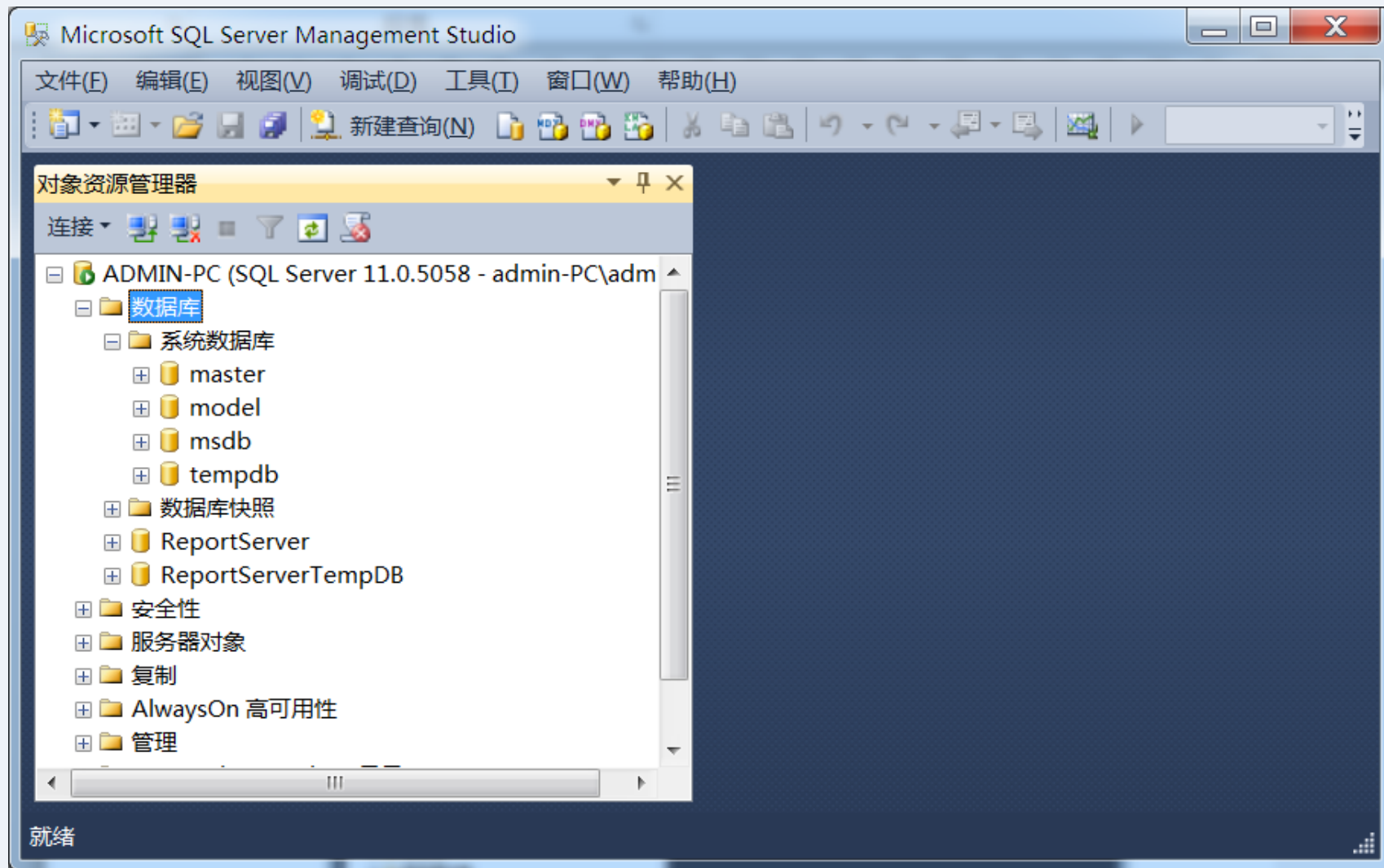
3.2.3 Management Studio

- 开始→所有程序→Microsoft SQL 2012→SQL Server Management Studio”命令，启动Management Studio

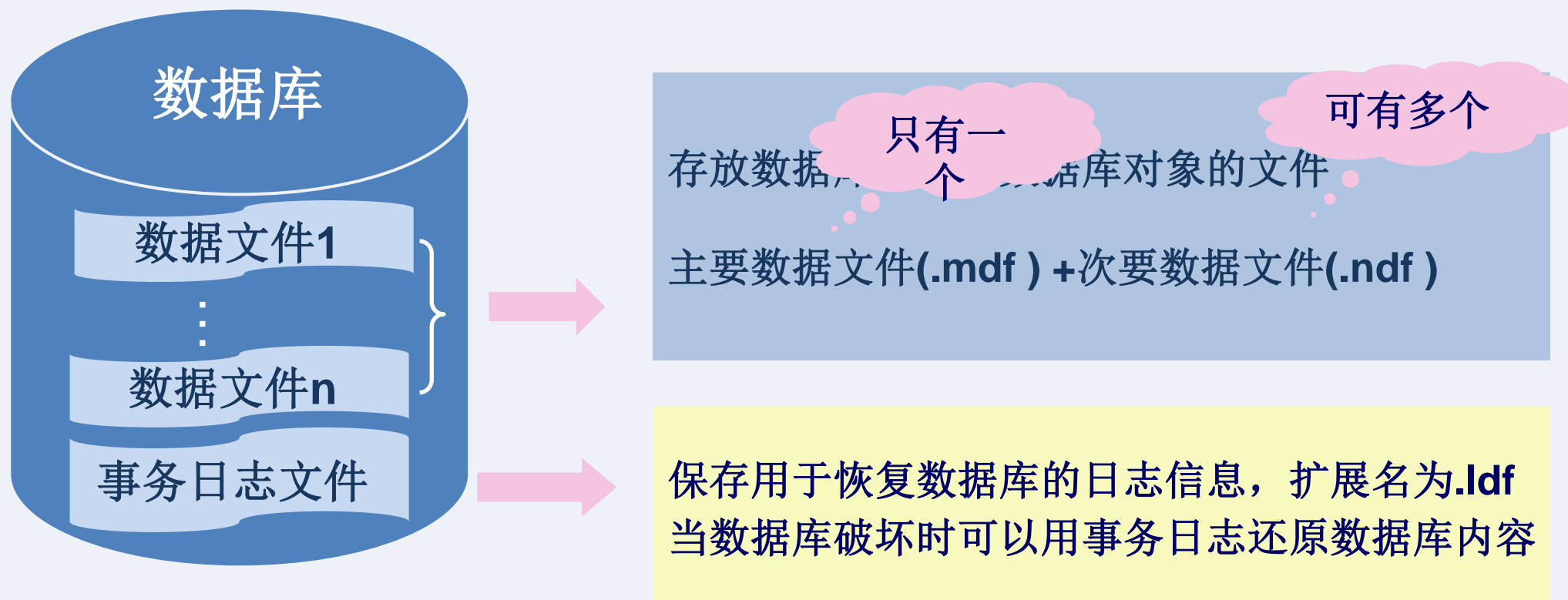


3.2 SQL Server 2012 简介

第3章



3.3.1 数据库的结构



— 文件组

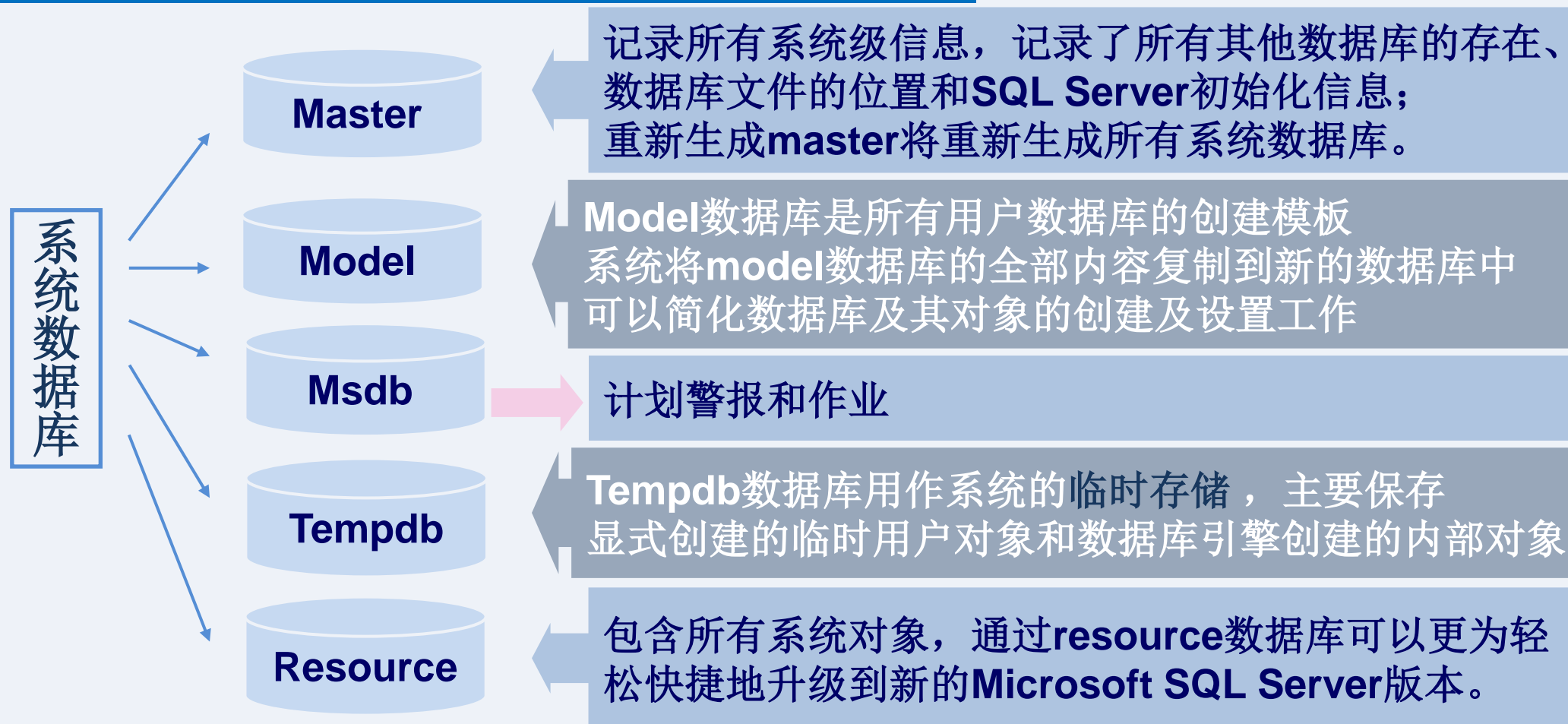
文件组 (File Group) 是将多个数据文件集合起来形成的一个整体

主要文件组+次要文件组

一个数据文件只能存在于一个文件组中，一个文件组也只能被一个数据库使用

日志文件不分组，它不属于任何文件组

3.3.2 SQL Server 2012 的系统数据库



3.3.3 SQL Server的示例数据库

SQL Server 2012提供了AdventureWorks示例数据库。与SQL Server 2000等早期版本不同，SQL Server 2012默认并不安装示例数据库，需要手工下载安装，下载地址为：

<http://go.microsoft.com/fwlink/?LinkId=87843>。SQL Server 2012联机丛书基本都以该数据库为例讲解，建议读者手工下载安装该示例数据库。 (Move to Github)

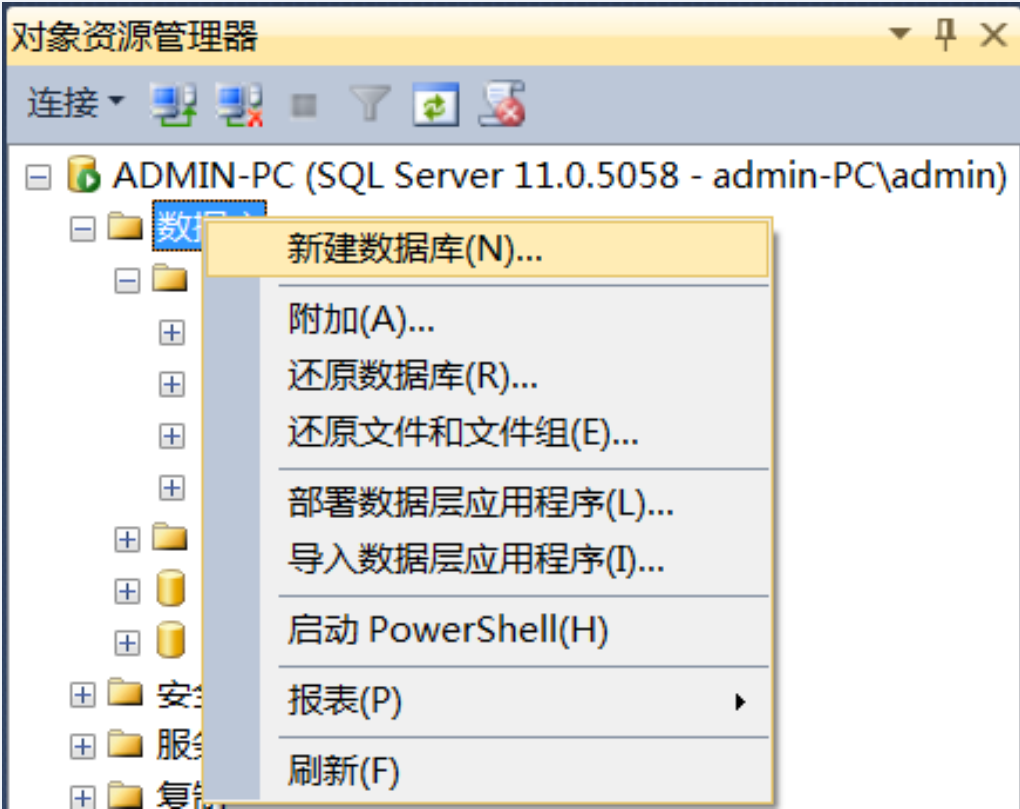
3.3.4 创建用户数据库

@ 用 Management Studio 创建数据库

(1) 在图3-2所示的**Management Studio**界面中，在“对象资源管理器”窗口，右键单击“数据库”节点，在弹出的快捷菜单中选择“新建数据库(N)...”命令（见图3-3），即可打开新建数据库窗口（见图3-4）。

(2) 图3-4 中，在“常规”选项卡的“数据库名称”文本框中输入数据库的名称。在“数据库文件”列表中，指定数据库文件的名称、存储位置、初始容量大小和所属文件组等信息，并进行数据库文件大小、扩充方式和容量限制的设置。

(3) 单击“确定”按钮，则创建一个新数据库。



3-3 新建用户数据库

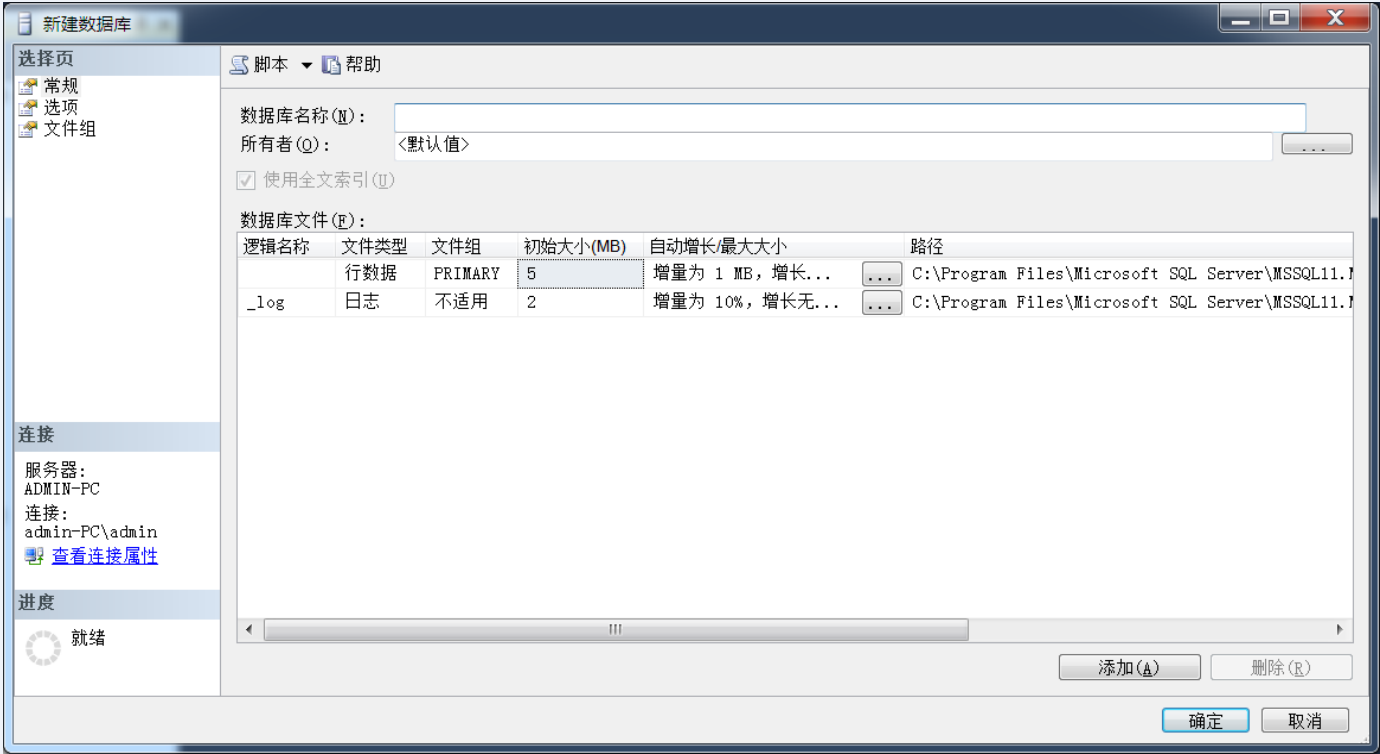


图3-4 新建数据库窗口

@ 用SQL命令创建数据库

创建数据库的**SQL**命令的语法格式如下所示:

CREATE DATABASE 数据库名称

[ON

[FILEGROUP 文件组名称]

(NAME=数据文件逻辑名称,

FILENAME='路径+数据文件名', SIZE=数据文件初始大小,

MAXSIZE=数据文件最大容量,

FILEGROWTH=数据文件自动增长容量,)]

[LOG ON

(NAME=日志文件逻辑名称,

FILENAME='路径+日志文件名' , SIZE=日志文件初始大小,

MAXSIZE=日志文件最大容量,

FILEGROWTH=日志文件自动增长容量,)]

[COLLATE 数据库校验方式名称]

[FOR ATTACH]



(1) 用[]括起来的语句，表示在创建数据库的过程中可以选用或者不选用，例如，在创建数据库的过程中，如果只用第一条语句“CREATE DATABASE 数据库名称”，DBMS将会按照默认的“逻辑名称”、“文件组”、“初始大小”、“自动增长”和“路径”等属性创建数据库。

(2) “FILEGROWTH”可以是具体的容量，也可以是UNLIMITED，表示文件无增长容量限制。

(3) “数据库校验方式名称”可以是Windows校验方式名称，也可以是SQL校验方式名称。

(4) “FOR ATTACH”表示将已经存在的数据库文件附加到新的数据库中。

(5) 用()括起来的语句，除了最后一行命令之外，其余的命令都用逗号作为分隔符。

[例3-1] 用SQL命令创建一个教学数据库Teach，数据文件的逻辑名称为Teach_Data，数据文件存放在E盘根目录下，文件名为TeachData.mdf，数据文件的初始存储空间大小为10MB，最大存储空间为500MB，存储空间自动增长量为10MB；日志文件的逻辑名称为Teach_Log，日志文件物理地存放在E盘根目录下，文件名为TeachData.ldf，初始存储空间大小为5MB，最大存储空间为500MB，存储空间自动增长量为5MB。

```
CREATE DATABASE Teach  
ON  
( NAME=Teach_Data,  
  FILENAME='E:\TeachData.mdf ',  
  SIZE=10,  
  MAXSIZE=500,  
  FILEGROWTH=10)  
LOG ON  
( NAME=Teach_Log,  
  FILENAME='E:\TeachData.ldf ',  
  SIZE=5,  
  MAXSIZE=500,  
  FILEGROWTH=5)
```

3.3.5 修改用户数据库

用 Management Studio修改数据库

打开“对象资源管理器”，右键单击要修改的数据库，从弹出菜单中选择“属性”命令，即可数据库属性对话框，如图3-5所示。

- (1) “常规”选项卡中包含数据库的状态、所有者、创建日期、大小、可用空间、用户数、备份和维护等信息。
- (2) “文件”选项卡中包含数据文件和日志文件的名称、存储位置、初始容量大小、文件增长和文件最大限制等信息。
- (3) “文件组”选项卡中可以添加或删除文件组。但是，如果文件组中有文件则不能删除，必须先将文件移出文件组，才能删除文件组。
- (4) “选项”选项卡中可以设置数据库的许多属性，如排序规则、恢复模式、兼容级别等。
- (5) “更改跟踪”选项卡可以设定是否对数据库的修改进行跟踪。



图3-5 数据库属性对话框

● 用 Management Studio修改数据库

- （6）“权限”选项卡可以设定用户或角色对此数据库的操作权限。
- （7）“扩展属性”选项卡可以设定表或列的扩展属性。在设计表或列时，通常通过表名或列名来表达含义，当表名或列名无法表达含义时，就需要使用扩展属性。
- （8）“镜像”选项卡可以设定是否对数据库启用镜像备份。镜像备份是一种高性能的备份方案，但需要投入一定的设备成本，一般用于高可靠性环境。
- （9）“事务日志传送”选项卡设定是否启用事务日志传送。事务日志传送备份是仅次于镜像的高可靠性备份方案，可以达到分钟级的灾难恢复能力，实施成本远小于镜像备份，是一种经济实用的备份方案。

● 用SQL命令修改数据库

可以使用ALTER DATABASE命令修改数据库。注意，只有数据库管理员（DBA）或者具有CREATE DATABASE权限的人员才有权执行此命令。下面列出常用的修改数据库的SQL命令的语法格式。

```
ALTER DATABASE 数据库名称  
ADD FILE(  
    具体文件格式)  
[,...n]  
[TO FILEGROUP 文件组名]  
|ADD LOG FILE(  
    具体文件格式)  
[,...n]  
|REMOVE FILE 文件逻辑名称  
|MODIFY FILE(  
    具体文件格式)
```

```
|ADD FILEGROUP 文件组名  
|REMOVE FILEGROUP 文件组名  
|MODIFY FILEGROUP 文件组名  
{ READ_ONLY|READ_WRITE,  
  | DEFAULT,  
  | NAME = 新文件组名}  
}
```

其中，“具体文件格式”为：

```
( NAME = 文件逻辑名称  
  [, NEWNAME = 新文件逻辑名称]  
  [, SIZE = 初始文件大小]  
  [, MAXSIZE = 文件最大容量]  
  [, FILEGROWTH = 文件自动增长容量]  
)
```

各主要参数说明如下：

ADD FILE：向数据库中添加数据文件。

ADD LOG FILE：向数据库中添加日志文件。

REMOVE FILE：从数据库中删除逻辑文件，并删除物理文件。如果文件不为空，则无法删除。

MODIFY FILE：指定要修改的文件。

ADD FILEGROUP：向数据库中添加文件组。

REMOVE FILEGROUP：从数据库中删除文件组。若文件组非空，无法将其删除，需先从文件组中删除所有文件。

MODIFY FILEGROUP：修改文件组名称、设置文件组的只读（**READ_ONLY**）或者读写（**READ_WRITE**）属性、指定文件组为默认文件组（**DEFAULT**）。

ALTER DATABASE命令可以在数据库中添加或删除文件和文件组、更改数据库属性或其文件和文件组、更改数据库排序规则和设置数据库选项。应注意的是，只有数据库管理员（**DBA**）或具有**CREATE DATABASE**权限的数据库所有者才有权执行此命令。

[例3-2] 修改Teach数据库中的Teach_Data文件增容方式为一次增加20MB。

```
ALTER DATABASE Teach  
MODIFY FILE  
( NAME = Teach_Data,  
  FILEGROWTH = 20)
```



[例3-3] 用SQL命令修改数据库Teach，添加一个次要数据文件，逻辑名称为Teach_Datanew，存放在E盘根目录下，文件名为Teach_Datanew.ndf。数据文件的初始大小为100MB，最大容量为200MB，文件自动增长容量为10MB。

```
ALTER DATABASE Teach  
ADD FILE(  
    NAME=Teach_Datanew,  
    FILENAME='E:\Teach_Datanew.ndf',  
    SIZE=100,  
    MAXSIZE=200,  
    FILEGROWTH=10)
```

[例3-4] 用SQL命令，从Teach数据库中删除例3-2中增加的次要数据文件。

```
ALTER DATABASE Teach  
REMOVE FILE Teach_Datanew
```



3.3.6 删除用户数据库

– 用Management Studio删除数据库

打开“对象资源管理器”，右键单击要删除的数据库，从弹出菜单中选择“删除”。删除数据库后，与此数据库关联的数据文件和日志文件都会被删除，系统数据库中存储的该数据库的所有信息也会被删除，因此务必要慎重！

– 用SQL命令删除数据库

```
DROP DATABASE  
数据库名称[,...n]
```

[例3-5] 删除数据库Teach。
`DROP DATABASE Teach`

3.3.7 查看数据库信息

... 用Management Studio查看数据库信息

... 用系统存储过程显示数据库信息

Sp_helpdb [[@dbname=] 'name']

用系统存储过程显示数据库结构

Sp_helpfile [[@filename =] 'name']

用系统存储过程显示文件信息

Sp_helpfilegroup [[@filegroupname =] 'name']

用系统存储过程显示文件组信息

EXEC Sp_helpdb AdventureWorks2012

EXEC Sp_helpfile Address

use AdventureWorks2012

EXEC Sp_helpfilegroup

3.3.8 迁移用户数据库

1. 分离和加载

如图3-6所示，在对象资源管理器中，选择要迁移的数据库节点，单击鼠标右键，在快捷菜单中选择“任务”，在之后出现的级联菜单中选择“分离”，会弹出如图3-7所示的“分离数据库”属性对话框，单击“确定”按钮，数据库文件就会从SQL server 2012成功分离。

之后，如图3-8所示，在对象资源管理器中选择“数据库”节点，单击鼠标右键，在快捷菜单中选择“附加”，会弹出“附加数据库”属性对话框，单击其中的“添加”按钮，在弹出的对话框中选择需要的.mdf文件，会得到如图3-9中所示的窗口，单击“确定”，即可把数据库文件附加成功。

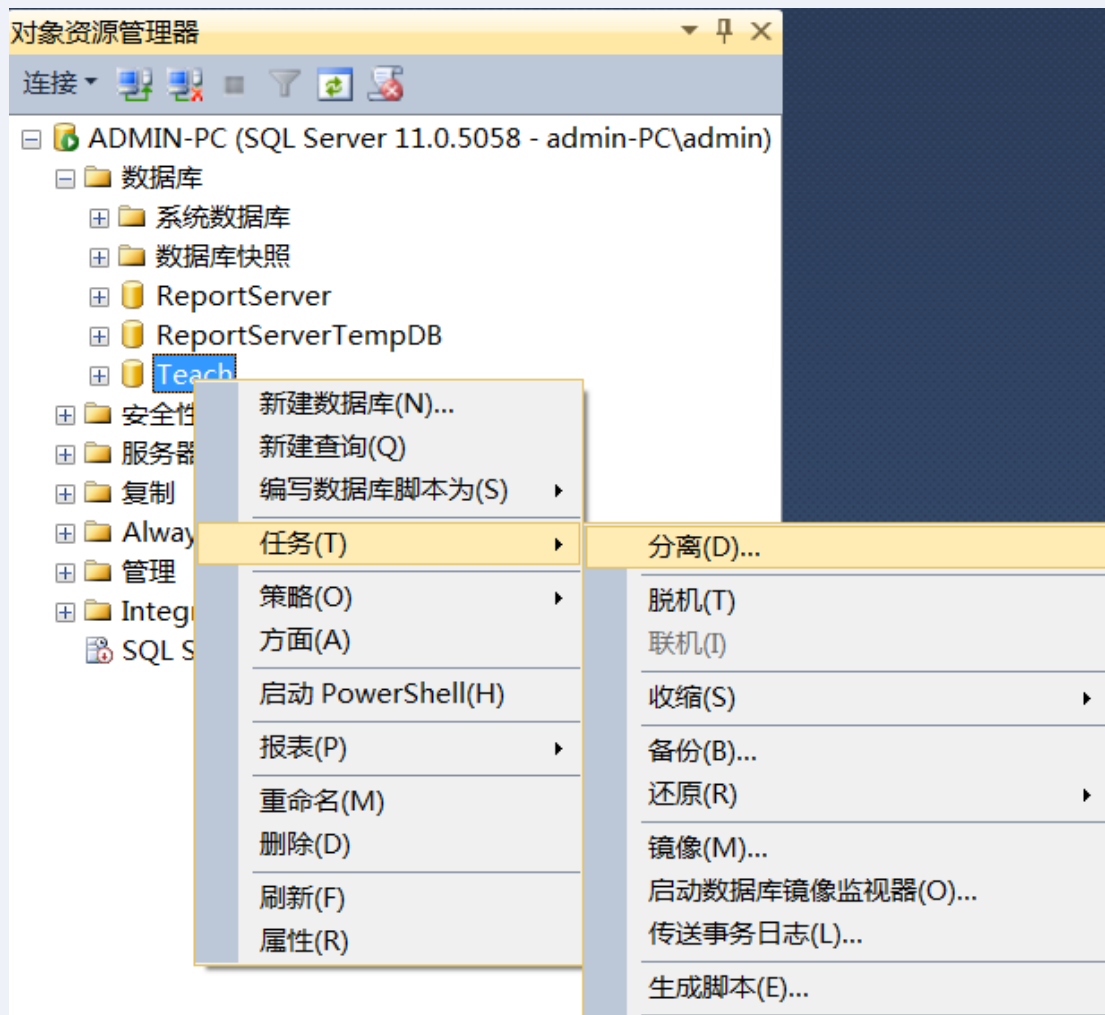


图3-6 分离数据库文件

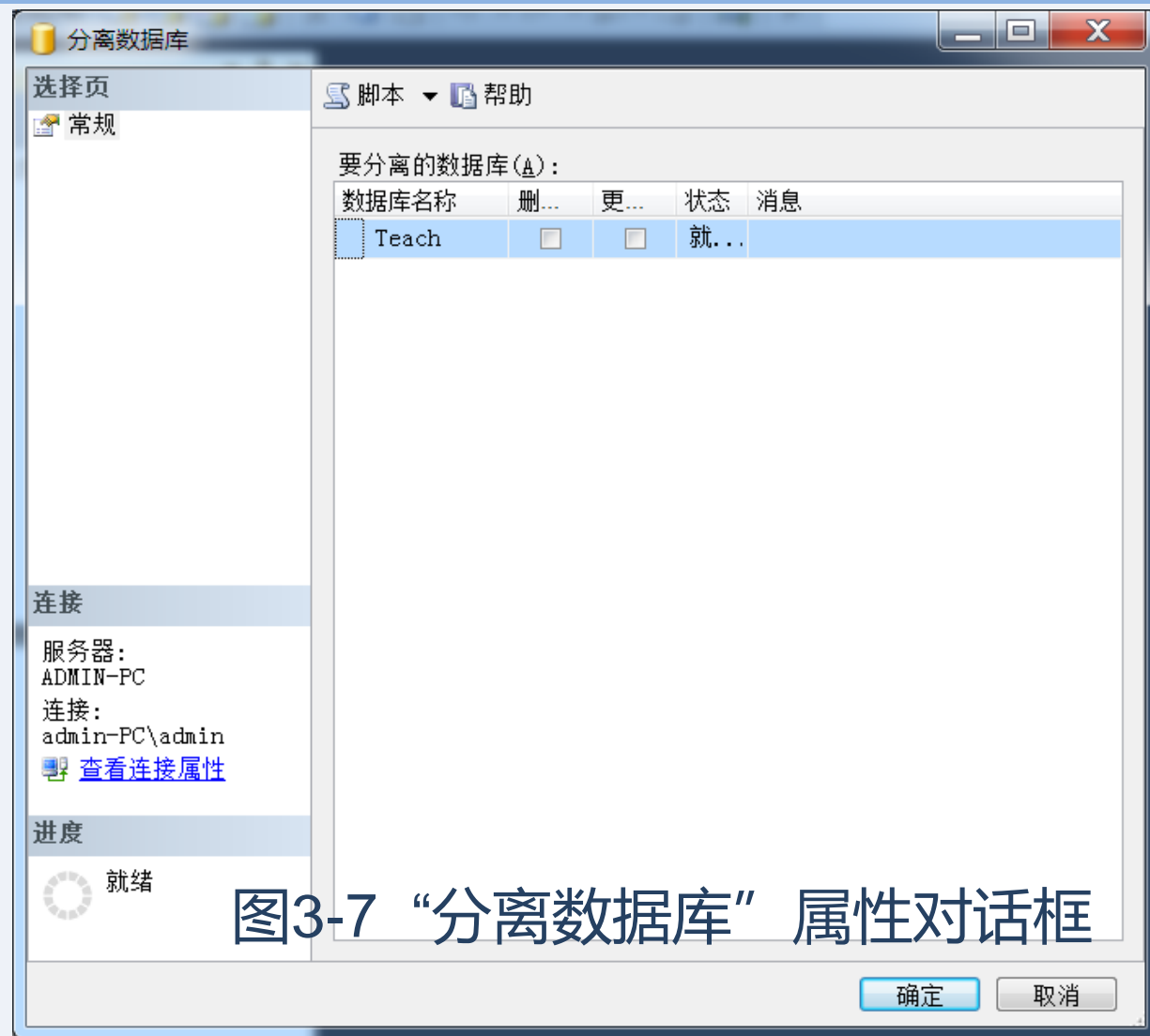


图3-7 “分离数据库” 属性对话框

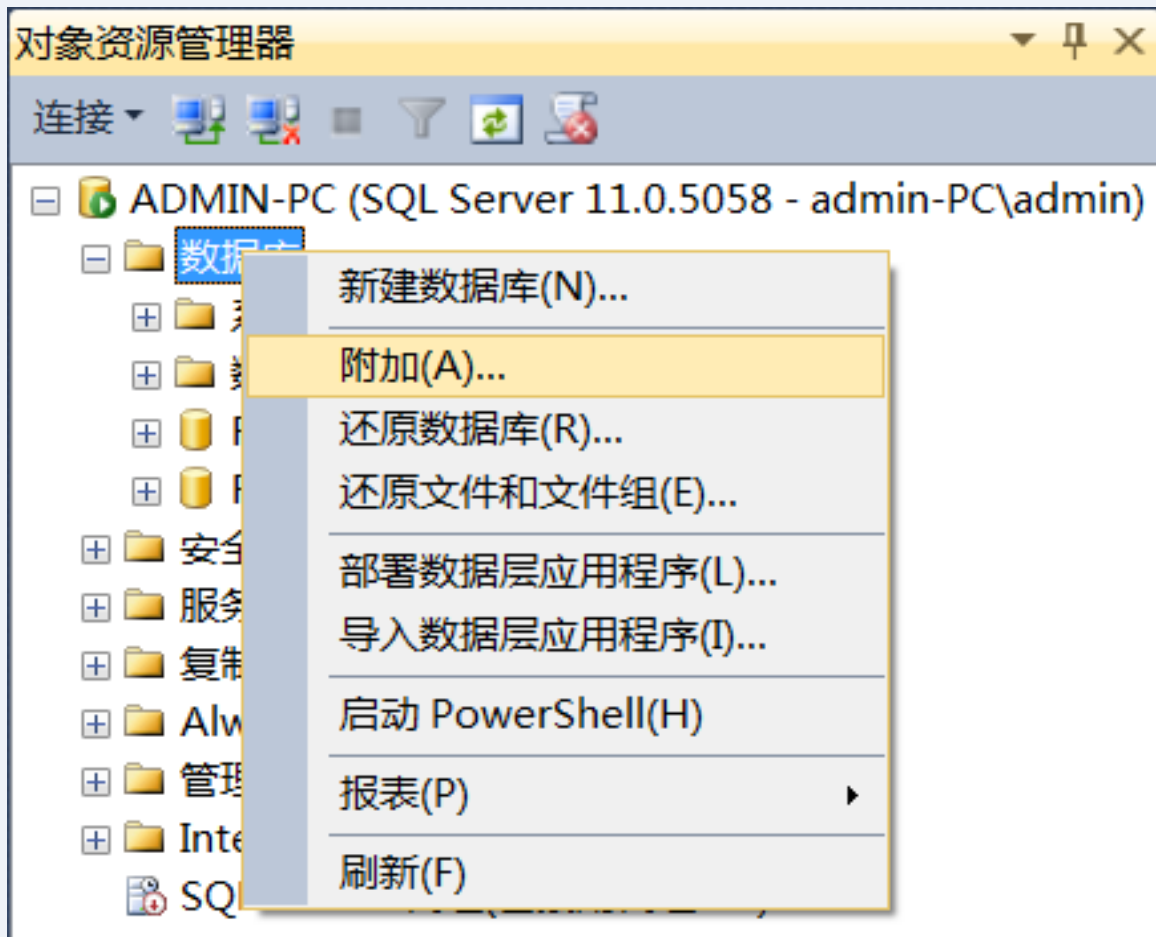


图3-8 附加数据库文件

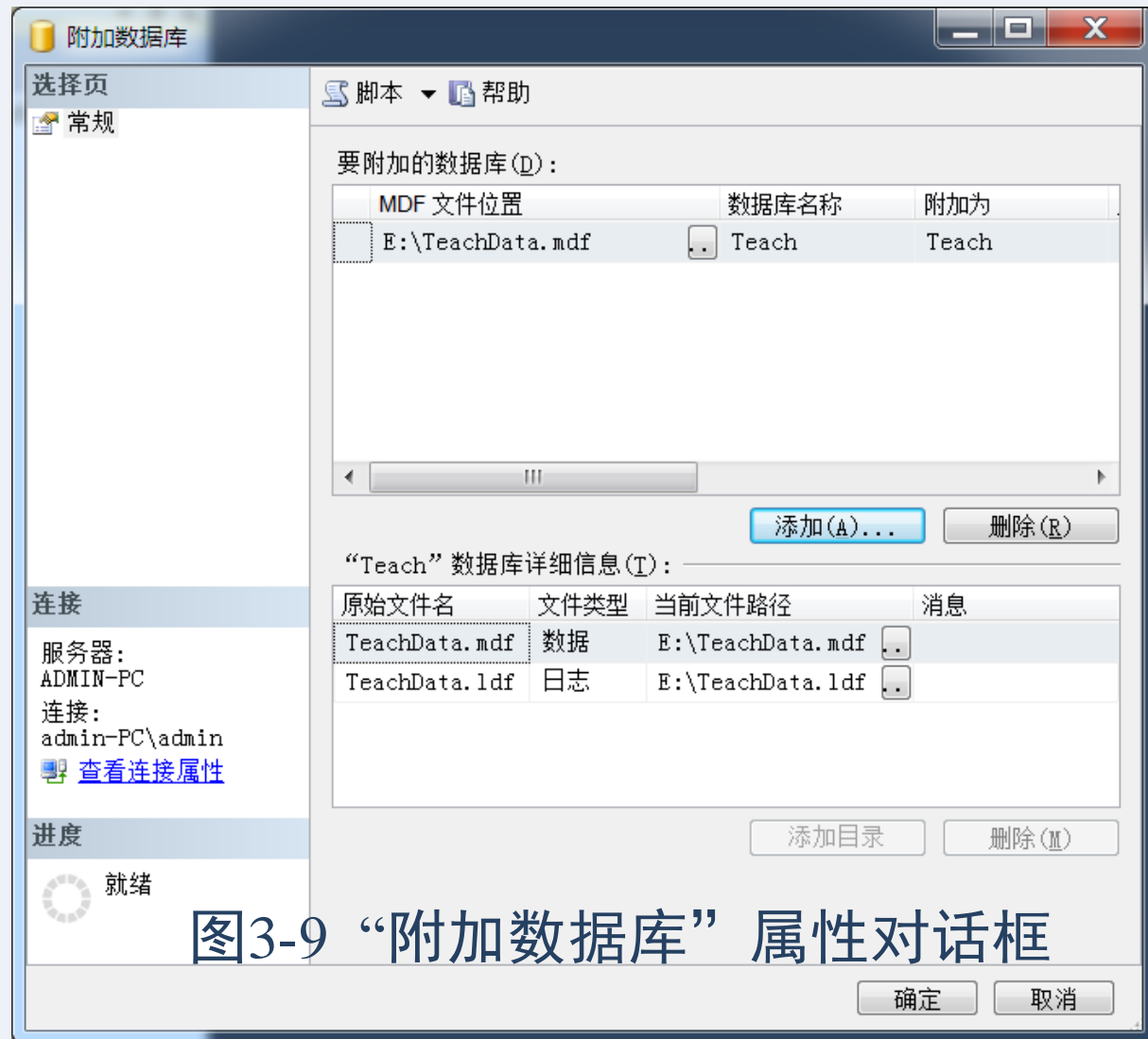


图3-9 “附加数据库” 属性对话框

2. 生成脚本

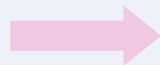
对于选定的数据库节点，在图3-6的级联菜单中，选择“生成脚本”命令，会弹出如图3-10所示的“生成和发布脚本”窗口。

图3-10 “生成和发布脚本”窗口



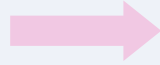
3.4.1 数据类型

精确数值型



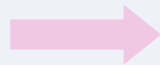
**bigint、int、smallint、tinyint、
bit、numeric、decimal、
money、smallmoney**

近似数值型

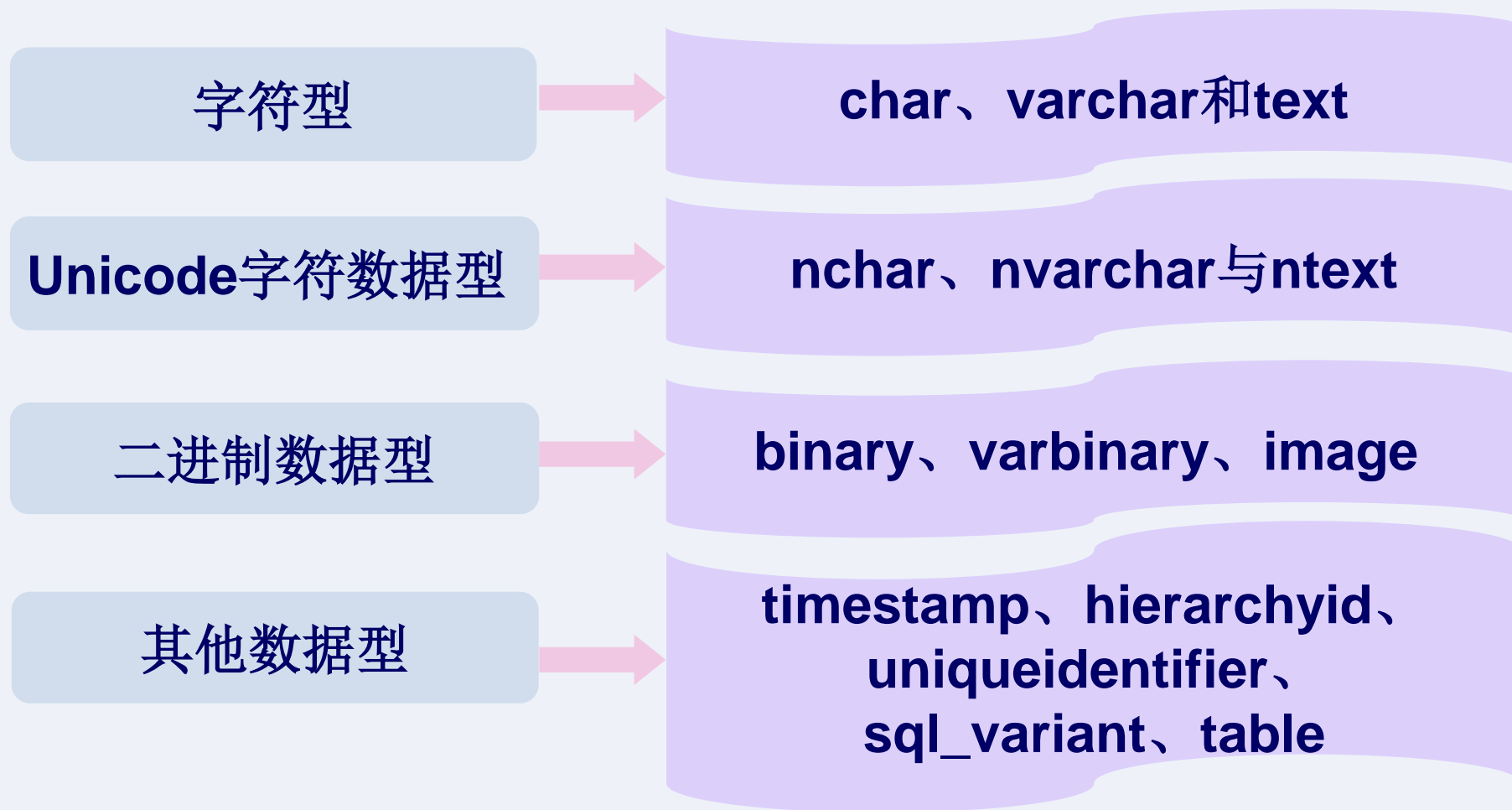


float和real

日期时间型



**date, datetime、datetime2、
time、smalldatetime、**



3.4.2 创建数据表

用Management Studio创建数据表

相关属性定义

- ① “列名” 由用户命名，最长**128**字符，可包含中文、英文、下划线、#号、货币符号（¥）及@符号。同一表中不允许有重名的列。
- ② “数据类型”，定义字段可存放数据的类型。
- ③ 字段的“长度”、“精度”和“小数位数”。字段的长度指字段所能容纳的最大数据量，不同的数据类型，其长度的意义不同。
- ④ “允许空”，当对某个字段的“允许空”列上打勾“√”时，表示该字段的值允许为**NULL**值。
- ⑤ “默认值”，表示该字段的默认值。如果规定了默认值，在向数据表中输入数据时，如果没有给该字段输入数据，系统自动将默认值写入该字段。

用SQL命令创建数据表

```
CREATE TABLE <表名>  
(<列定义>[{, <列定义>|<表约束>}])
```

 **<列名> <数据类型> [DEFAULT] [{<列约束>}]**

[例3-6] 用SQL命令建立一个学生表S。

```
CREATE TABLE S  
( SNo CHAR(6),  
  SN VARCHAR(10),  
  Sex NCHAR(1) DEFAULT '男',  
  Age INT,  
  Dept NVARCHAR(20))
```

→ 缺省值为“男”

3.4.3 定义数据表的约束

数据的完整性

正确性

有效性

相容性

SQL Server的数据完整性机制

约束（**Constraint**）

默认（**Default**）

规则（**Rule**）

触发器（**Trigger**）

存储过程（**Stored Procedure**）

完整性约束的基本语法格式

[CONSTRAINT <约束名>] <约束类型>

NULL/NOT NULL

UNIQUE

PRIMARY KEY

FOREIGN KEY

CHECK

NULL/NOT NULL约束

- NULL表示“不知道”、“不确定”或“没有数据”的意思
- 主键列不允许出现空值

[CONSTRAINT <约束名>][NULL | NOT NULL]

[例3-7] 建立一个S表，对SNo字段进行NC

CREATE TABLE S

**(SNo VARCHAR(6) CONSTRAINT S_CONS NOT NULL,
SN NVARCHAR(10),
Sex NCHAR(1),
Age INT,
Dept NVARCHAR(20))**

可省略约束名称：
SNo VARCHAR(6) NOT NULL

UNIQUE约束（唯一约束）

- 指明基本表在某一系列或多个列的组合上的取值必须唯一
- 在建立UNIQUE约束时，需要考虑以下几个因素：

使用UNIQUE约束的字段允许为NULL值。

一个表中可以允许有多个UNIQUE约束。

可以把UNIQUE约束定义在多个字段上。

UNIQUE约束用于强制在指定字段上创建一个UNIQUE索引，缺省为非聚集索引。

UNIQUE用于定义列约束

[CONSTRAINT <约束名>] UNIQUE

UNIQUE用于定义表约束

[CONSTRAINT <约束名>] UNIQUE (<列名>[{,<列名>}])

[例3-8] 建立一个S表，在SN上定义UNIQUE**约束。**

```
CREATE TABLE S  
( SNo VARCHAR(6),  
  SN NVARCHAR(10) CONSTRAINT SN_UNIQ UNIQUE,  
  Sex NCHAR(1),  
  Age INT,  
  Dept NVARCHAR(20))
```

SN_UNIQ可以省略
SN NVARCHAR(10) UNIQUE,

[例3-9] 建立一个S表，在SN+Sex上定义UNIQUE约束，此约束为表约束。

```
CREATE TABLE S
( SNo VARCHAR(6),
  SN NVARCHAR(10) ,
  Sex NCHAR(1),
  Age INT,
  Dept NVARCHAR(20)
  CONSTRAINT S_UNIQ UNIQUE(SN, Sex))
```


PRIMARY KEY约束（主键约束）

- 用于定义基本表的主键，起唯一标识作用
- PRIMARY KEY与UNIQUE 的区别：

不能为NULL

不能重复

在一个基本表中只能定义一个**PRIMARY KEY**约束，但可定义多个**UNIQUE**约束。

对于指定为**PRIMARY KEY**的一个列或多个列的组合，其中任何一个列都不能出现**NULL**值，而对于**UNIQUE**所约束的唯一键，则允许为**NULL**。

不能为同一个列或一组列，既定义**UNIQUE**约束，又定义**PRIMARY KEY**约束。

PRIMARY KEY用于定义列约束

```
CONSTRAINT <约束名> PRIMARY KEY
```

PRIMARY KEY用于定义表约束

```
[CONSTRAINT <约束名>] PRIMARY KEY (<列名>[{,<列名>}])
```

[例3-10] 建立一个**S**表，定义**SNo**为**S**的主键，建立另外一个数据表**C**，定义**CNo**为**C**的主键。定义数据表**S**：

```
CREATE TABLE S
( SNo VARCHAR(6) CONSTRAINT S_Prim PRIMARY KEY,
  SN NVARCHAR(10) UNIQUE,
  Sex NCHAR(1),
  Age INT,
  Dept NVARCHAR(20))
```

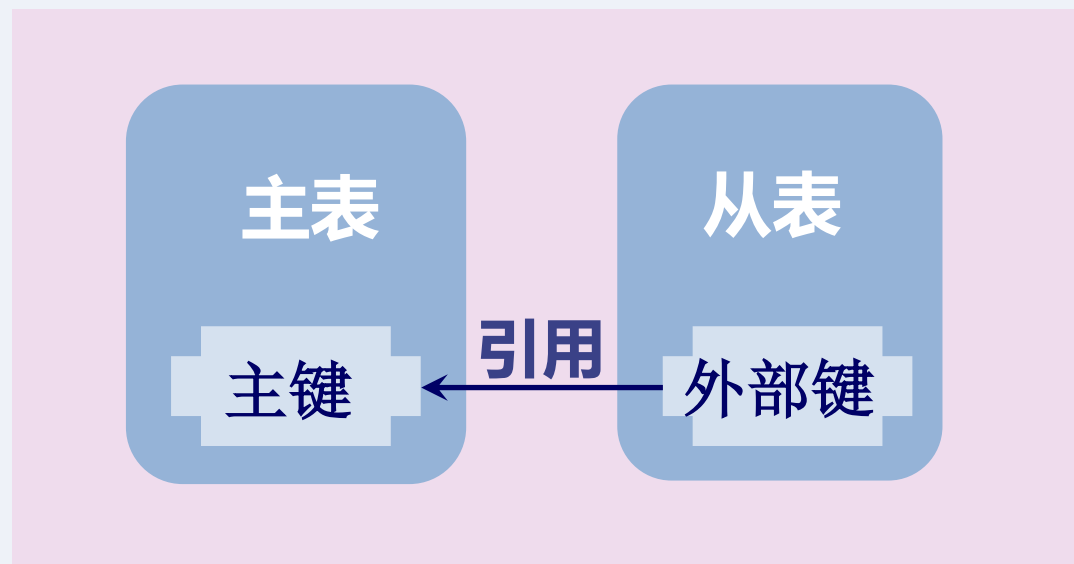
定义数据表**C**：

```
CREATE TABLE C
( CNo VARCHAR(6) CONSTRAINT C_Prim PRIMARY KEY,
  CN NVARCHAR(20),
  CT INT)
```

[例3-11] 建立一个SC表，定义SNo+CNo为SC的主键。

```
CREATE TABLE SC
( SNo VARCHAR(6) NOT NULL,
  CNo VARCHAR(6) NOT NULL,
  Score NUMERIC(4,1),
  CONSTRAINT SC_Prim PRIMARY KEY(SNo,CNo))
```

FOREIGN KEY约束（外键约束）



```
[CONSTRAINT<约束名>] FOREIGN KEY REFERENCES  
<主表名> (<列名>[{,<列名>}])
```

[例3-12] 建立一个SC表，定义SNo，CNo为SC的外部键。

```
CREATE TABLE SC
( SNo VARCHAR(6) NOT NULL CONSTRAINT S_Fore
  FOREIGN KEY REFERENCES S(SNo),
  CNo VARCHAR(6) NOT NULL CONSTRAINT C_Fore
  FOREIGN KEY REFERENCES C(CNo),
  Score NUMERIC(4,1),
  CONSTRAINT S_C_Prim PRIMARY KEY (SNo,CNo))
```

CHECK约束

CHECK约束用来检查字段值所允许的范围

在建立CHECK约束时，需要考虑以下几个因素：

一个表中可以定义多个CHECK约束。

每个字段只能定义一个CHECK约束。

在多个字段上定义的CHECK约束必须为表约束。

当执行INSERT、UPDATE语句时，CHECK约束将验证数据。

[CONSTRAINT <约束名>] CHECK (<条件>)

[例3-13] 建立一个SC表，定义Score的取值范围为0~100之间。

```
CREATE TABLE SC
( SNo VARCHAR(6),
  CNo VARCHAR(6),
  Score NUMERIC(4,1) CONSTRAINT Score_Chk
  CHECK(Score>=0 AND Score <=100))
```


[例3-14] 建立包含完整性定义的学生表S。

```
CREATE TABLE S
( SNo VARCHAR(6) CONSTRAINT S_Prim PRIMARY KEY,
  SN NVARCHAR(10) CONSTRAINT SN_Cons NOT NULL,
  Sex NCHAR(1) CONSTRAINT Sex_Cons NOT NULL DEFAULT '男',
  Age INT CONSTRAINT Age_Cons NOT NULL
      CONSTRAINT Age_Chk CHECK (Age BETWEEN 15 AND 50),
  Dept NVARCHAR(20) CONSTRAINT Dept_Cons NOT NULL)
```

3.4.4 修改数据表

用Management Studio修改数据表的结构

- (1) 在**Management Studio**中的“对象资源管理器”窗口中，展开“数据库”节点。
- (2) 右键单击要修改的数据表，从快捷菜单中选择“设计”命令，则会弹出图**3-12**所示的修改数据表结构对话框。可以在此对话框中修改列的数据类型、名称等属性，添加或删除列，也可以指定表的主关键字约束。
- (3) 修改完毕后，单击工具栏中的保存按钮，存盘退出。

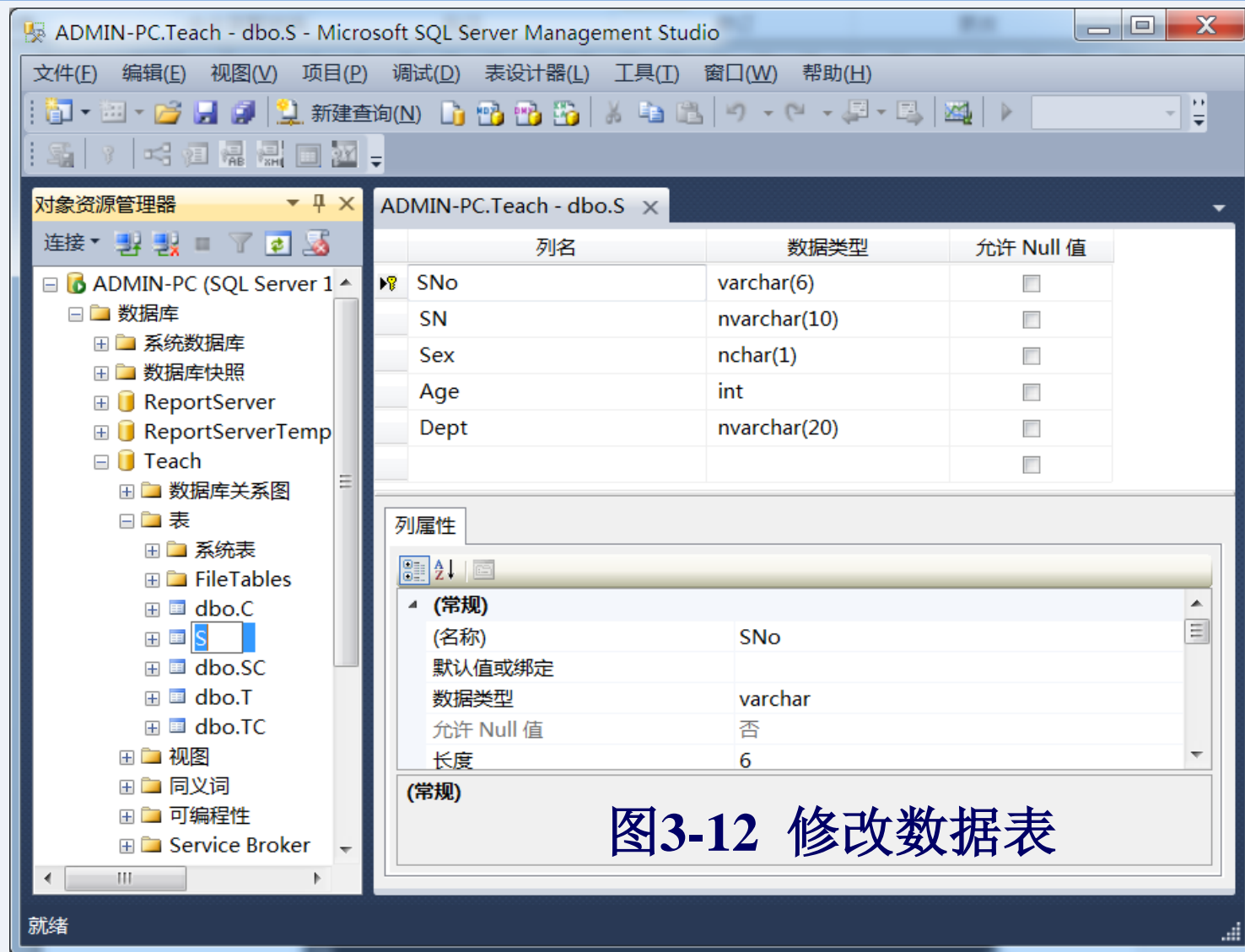


图3-12 修改数据表

用SQL命令修改数据表

```
ALTER TABLE <表名>  
ADD <列定义> | <完整性约束定义>
```

```
ALTER TABLE <表名>  
ALTER COLUMN <列名> <数据类型> [NULL | NOT NULL]
```

```
ALTER TABLE <表名>  
DROP CONSTRAINT <约束名>
```

[例3-15] 在S表中增加一个班号列和住址列。

```
ALTER TABLE S  
ADD  
Class_No VARCHAR(6),  
Address NVARCHAR(20)
```

使用此方式增加的新列自动填充NULL值，所以不能为增加的新列指定NOT NULL约束。

[例3-16] 在SC表中增加完整性约束定义，使Score在0~100之间。

```
ALTER TABLE SC  
ADD  
CONSTRAINT Score_Chk CHECK(Score BETWEEN 0 AND 100)
```

[例3-17] 把S表中的SN列加宽到12个字符。

```
ALTER TABLE S  
ALTER COLUMN  
SN NVARCHAR(12)
```

- 不能改变列名；
- 不能将含有空值的列的定义修改为NOT NULL约束；
- 若列中已有数据，则不能减少该列的宽度，也不能改变其数据类型；
- 只能修改NULL/NOT NULL约束，其他类型的约束在修改之前必须先
将约束删除，然后再重新添加修改过的约束定义。

[例3-18] 删除S表中的主键。

```
ALTER TABLE S  
DROP CONSTRAINT S_Prim
```

3.4.5 删除基本表

- 用Management Studio删除数据表
- 用SQL命令删除数据表

```
DROP TABLE <表名>
```

只能删除自己建立的表，不能删除其他用户所建的表

3.4.6 查看数据表

查看数据表的属性

属性包括：表名，所有者，创建日期，文件组，记录行数，数据表中的字段名称、结构和类型等。

查看数据表中的数据

在Management Studio的对象资源管理器中，用右键单击要查看数据的表，从快捷菜单中选择“选择前1000行(W)”命令，则会显示表中的前1000条数据

3.5.1 单关系（表）的数据查询结构

投影

SELECT [ALL|DISTINCT][TOP N [PERCENT][WITH TIES]]

〈列名〉 [AS 别名1] [{, 〈列名〉 [AS 别名2]]

FROM 〈表名〉 [[AS] 表别名]

WHERE 〈检索条件〉]

[GROUP BY <列名1>[HAVING <条件表达式>]]

选取

[ORDER BY <列名2>[ASC|DESC]]

3.5.2 无条件查询

无条件查询是指只包含“SELECT...FROM”的查询，这种查询最简单，相当于只对关系（表）进行投影操作。

[例3-20] 查询全体学生的学号、姓名和年龄。

```
SELECT SNo, SN, Age  
FROM S
```

在菜单栏下方的快捷工具中，单击“新建查询”，会弹出如图3-16所示的查询窗口（即对象资源管理器右侧的窗口）。在查询窗口中输入上述查询语句，单击“! 执行”，即可得到如图3-17所示的查询结果界面，可以看出，在查询语句的下方，是其对应的查询结果。

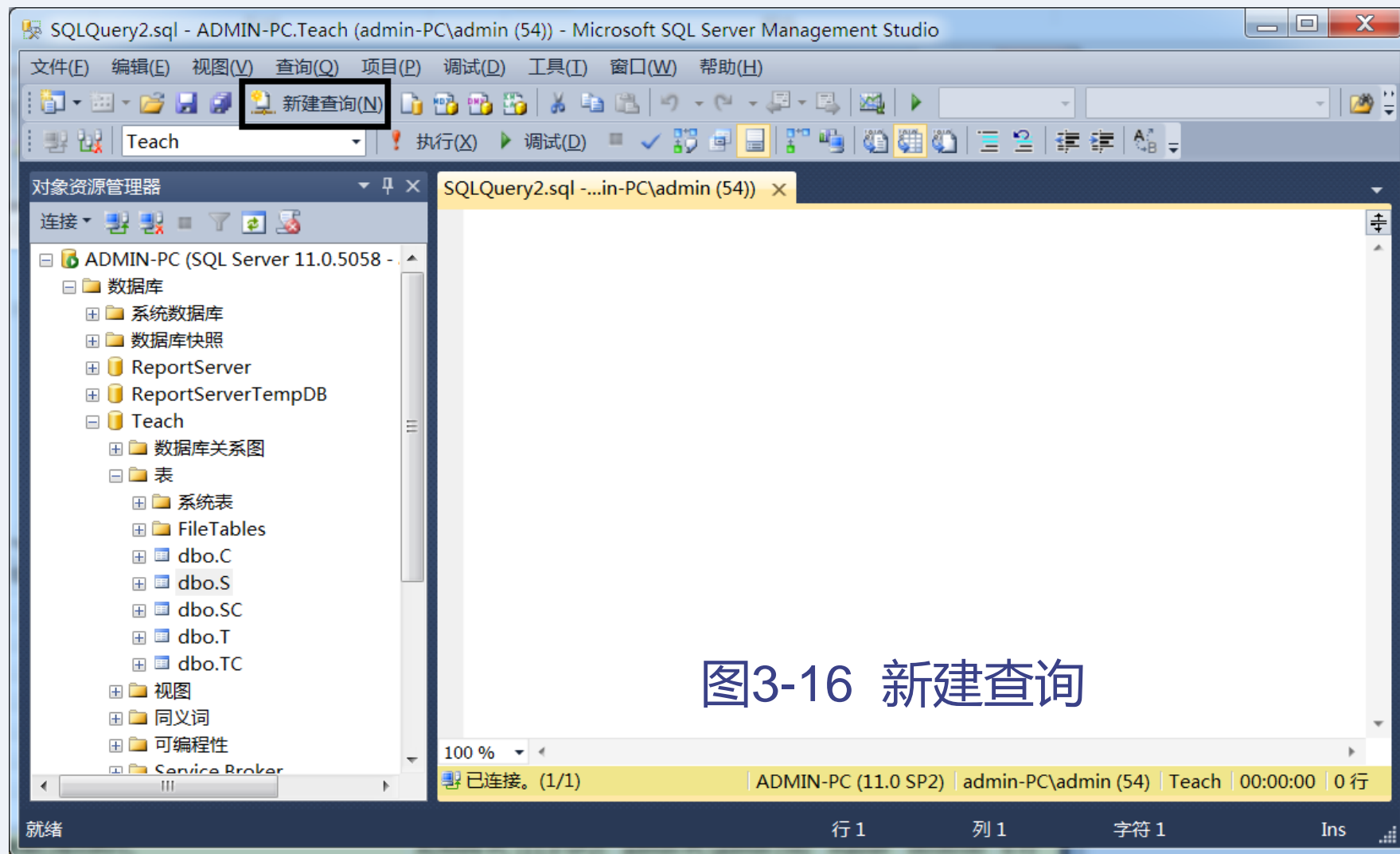
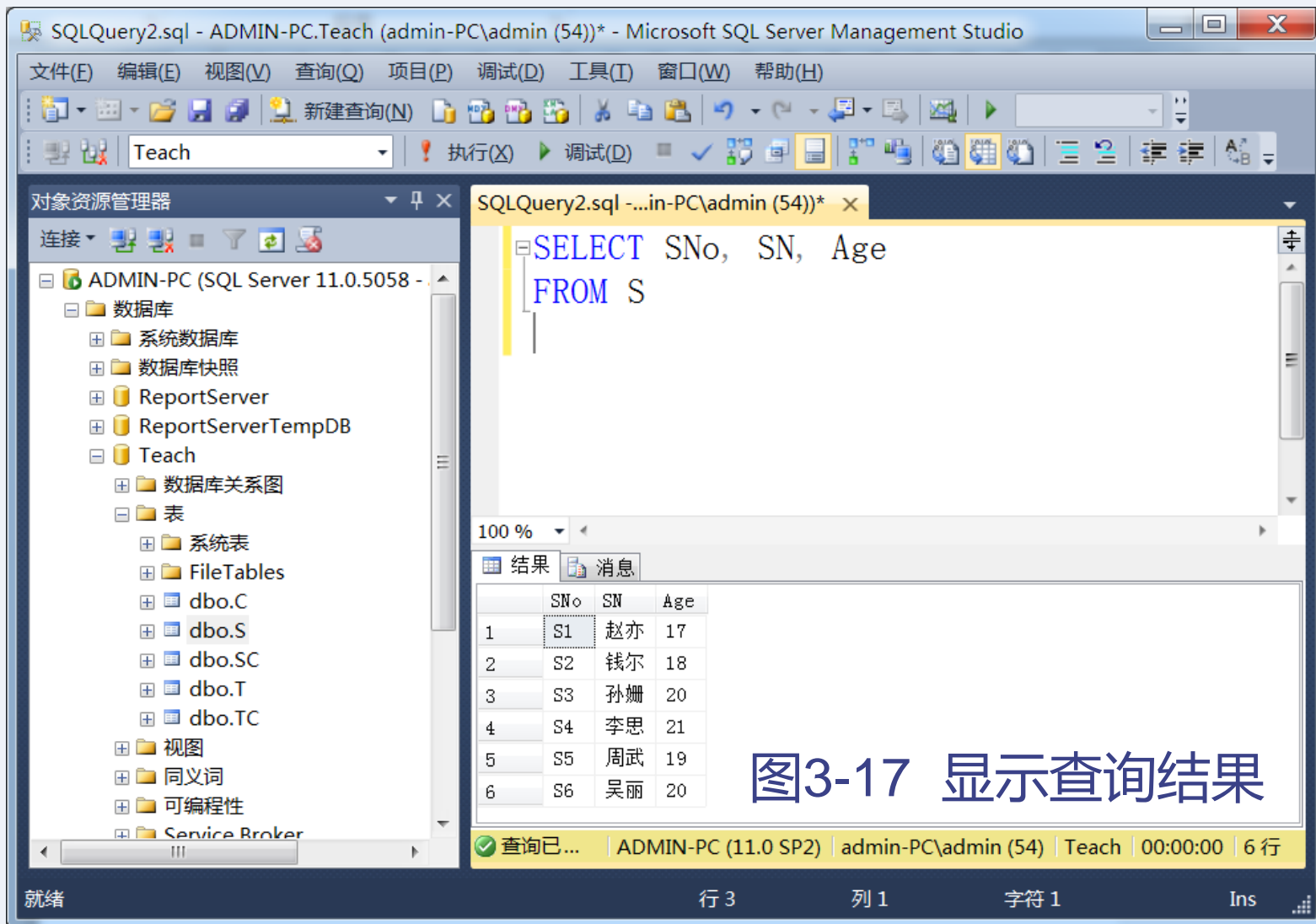


图3-16 新建查询

3.5 单关系（表）的数据查询

第3章



The screenshot displays the Microsoft SQL Server Management Studio interface. The title bar indicates the file is 'SQLQuery2.sql' and the server is 'ADMIN-PC.Teach (admin-PC\admin (54))*'. The menu bar includes '文件(F)', '编辑(E)', '视图(V)', '查询(Q)', '项目(P)', '调试(D)', '工具(T)', '窗口(W)', and '帮助(H)'. The toolbar contains icons for '新建查询(N)', '执行(X)', and '调试(D)'. The '对象资源管理器' (Object Explorer) on the left shows the 'Teach' database selected, with its tables listed: '系统表', 'FileTables', 'dbo.C', 'dbo.S', 'dbo.SC', 'dbo.T', 'dbo.TC', '视图', '同义词', '可编程性', and 'Service Broker'. The 'SQLQuery2.sql' editor window shows the following query:

```
SELECT SNo, SN, Age
FROM S
```

The '结果' (Results) pane at the bottom displays the query output as a table with 6 rows and 3 columns: SNo, SN, and Age.

	SNo	SN	Age
1	S1	赵亦	17
2	S2	钱尔	18
3	S3	孙姗	20
4	S4	李思	21
5	S5	周武	19
6	S6	吴丽	20

The status bar at the bottom shows '就绪' (Ready) on the left, and '行 3' (Line 3), '列 1' (Column 1), '字符 1' (Character 1), and 'Ins' on the right. A yellow banner at the bottom of the results pane indicates '查询已...' (Query completed), 'ADMIN-PC (11.0 SP2)', 'admin-PC\admin (54)', 'Teach', '00:00:00', and '6 行' (6 rows).

图3-17 显示查询结果

例3-21] 查询学生的全部信息。

```
SELECT *  
FROM S
```

用“*”表示S表的全部列名，而不必逐一列出。

[例3-22] 查询选修了课程的学生学号。

```
SELECT DISTINCT SNo  
FROM SC
```

[例3-23] 查询全体学生的姓名、学号和年龄。

```
SELECT SN Name, SNo, Age  
FROM S
```



```
SELECT SN AS Name, SNo, Age
```

3.5.3 条件查询

常用的比较运算符

运算符	含义
=, >, <, >=, <=, !=, <>	比较大小
AND, OR, NOT	多重条件
BETWEEN AND	确定范围
IN	确定集合
LIKE	字符匹配
IS NULL	空值

比较大小

[例3-24] 查询选修课程号为'C1'的学生的学号和成绩。

```
SELECT SNo,Score  
FROM SC  
WHERE CNo= 'C1'
```

多重条件查询

NOT、AND、OR

高

低

(用户可以使用括号改变优先级)

[例3-26] 查询选修C1或C2且分数大于等于85分学生的学号、课程号和成绩。

```
SELECT SNo, CNo, Score
```

```
FROM SC
```

```
WHERE (CNo = 'C1' OR CNo = 'C2') AND (Score >= 85)
```


确定范围

[例3-27] 查询工资在1000元~1500元之间的教师的教师号、姓名及职称。

```
SELECT TNo,TN,Prof  
FROM T  
WHERE Sal BETWEEN 1000 AND 1500
```

WHERE Sal>=1000
AND Sal<=1500

[例3-28] 查询工资不在1000元~1500元间的教师的教师号、姓名及职称。

```
SELECT TNo,TN,Prof  
FROM T  
WHERE Sal NOT BETWEEN 1000 AND 1500
```

确定集合

- 利用 “IN”操作可以查询属性值属于指定集合的元组。

[例3-29] 查询选修**C1**或**C2**的学生的学号、课程号和成绩。

```
SELECT SNo, CNo, Score  
FROM SC  
WHERE CNo IN('C1', 'C2')
```

此语句也可以使用逻辑运算符 “OR”实现。

```
WHERE CNo='C1'OR CNo= 'C2'
```

利用 “NOT IN”可以查询指定集合外的元组。

[例3-30] 查询没有选修**C1**，也没有选修**C2**的学生的学号、课程号和成绩。

```
SELECT SNo, CNo, Score  
FROM SC  
WHERE CNo NOT IN('C1', 'C2')
```

部分匹配查询

- 当不知道完全精确的值时，用户可以使用LIKE或NOT LIKE进行部分匹配查询（也称模糊查询）

<属性名> LIKE <字符串常量>

[例3-31] 查询所有姓张的教师的教师号和姓名。

```
SELECT TNo, TN  
FROM T  
WHERE TN LIKE '张%'
```

[例3-34] 查询姓名中第二个汉字是“力”的教师号和姓名。

```
SELECT TNo, TN  
FROM T  
WHERE TN LIKE '_力%'
```

空值查询

- 某个字段没有值称之为具有空值（NULL）
- 空值不同于零和空格，它不占任何存储空间

[例3-33] 查询没有考试成绩的学生的学号和相应的课程号。

```
SELECT SNo, CNo  
FROM SC  
WHERE Score IS NULL
```

3.5.4 常用库函数及统计汇总查询

函数名称	功 能
AVG	按列计算平均值
SUM	按列计算值的总和
MAX	求一列中的最大值
MIN	求一列中的最小值
COUNT	按列值计个数

[例3-34] 求学号为S1的学生的总分和平均分。

```
SELECT SUM(Score) AS TotalScore, AVG(Score) AS  
AvgScore  
FROM SC  
WHERE (SNo = 'S1')
```

[例3-35] 求选修C1号课程的最高分、最低分及之间相差的分数。

```
SELECT MAX(Score) AS MaxScore, MIN(Score) AS  
MinScore, MAX(Score) - MIN(Score) AS Diff  
FROM SC  
WHERE (CNo = 'C1')
```

[例3-36] 求计算机系学生的总数。

```
SELECT COUNT (SNo) FROM S  
WHERE Dept= '计算机'
```

[例3-37] 求学校中共有多少个系。

```
SELECT COUNT(DISTINCT Dept) AS DeptNum  
FROM S
```

DISTINCT消去重复行

[例3-38] 统计有成绩同学的人数。

```
SELECT COUNT (Score)  
FROM SC
```

上例中成绩为0的同学也计算在内，没有成绩（即为空值）的不计算。

[例3-39] 利用特殊函数COUNT(*)求计算机系学生的总数。

```
SELECT COUNT(*) FROM S  
WHERE Dept='计算机'
```

COUNT (*) 用来统计元组的个数，不消除重复行，不允许使用DISTINCT关键字。

3.5.5 分组查询

GROUP BY子句可以将查询结果按属性列或属性列组合在行的方向上进行分组，每组在属性列或属性列组合上具有相同的值。

[例3-40] 查询各个教师的教师号及其任课的门数。

```
SELECT TNo,COUNT(*) AS C_Num  
FROM TC  
GROUP BY TNo
```

GROUP BY子句按**TNo**的值分组，所有具有相同**TNo**的元组为一组，对每一组使用函数**COUNT**进行计算，统计出各位教师任课的门数。

若在分组后还要按照一定的条件进行筛选，则需使用HAVING子句

[例3-41] 查询选修两门以上（含两门）课程的学生学号和选课门数。

```
SELECT SNo, COUNT(*) AS SC_Num  
FROM SC  
GROUP BY SNo  
HAVING (COUNT(*) >= 2)
```

GROUP BY子句按**SNo**的值分组，所有具有相同**SNo**的元组为一组，对每一组使用函数**COUNT**进行计算，统计出每位学生选课的门数。
HAVING子句去掉不满足**COUNT (*) >= 2**的组

3.5.6 查询结果的排序

当需要对查询结果排序时，应该使用**ORDER BY**子句，**ORDER BY**子句必须出现在其他子句之后。排序方式可以指定，**DESC**为降序，**ASC**为升序，缺省时为升序。

[例3-42] 查询选修**C1**的学生学号和成绩，并按成绩降序排列。

```
SELECT SNo, Score  
FROM SC  
WHERE (CNo = 'C1')  
ORDER BY Score DESC
```

[例3-43] 查询选修**C2，C3，C4或C5**课程的学号、课程号和成绩，查询结果按学号升序排列，学号相同再按成绩降序排列。

```
SELECT SNo, CNo, Score  
FROM SC  
WHERE CNo IN ('C2', 'C3', 'C4', 'C5')  
ORDER BY SNo, Score DESC
```

3.6.1 多关系（表）的连接查询结构

表的连接方法有以下两种：

...表之间满足一定条件的行进行连接时，**FROM**子句指明进行连接的表名，**WHERE**子句指明连接的列名及其连接条件。

...利用关键字**JOIN**进行连接：当将**JOIN** 关键词放于**FROM**子句中时，应有关键词**ON**与之对应，以表明连接的条件。

JOIN的分类

INNER JOIN	显示符合条件的记录，此为默认值
LEFT (OUTER) JOIN	为左（外）连接，用于显示符合条件的数据行以及左边表中不符合条件的数据行，此时右边数据行会以NULL来显示
RIGHT (OUTER) JOIN	右（外）连接，用于显示符合条件的数据行以及右边表中不符合条件的数据行。此时左边数据行会以NULL来显示
FULL (OUTER) JOIN	显示符合条件的数据行以及左边表和右边表中不符合条件的数据行。此时缺乏数据的数据行会以NULL来显示
CROSS JOIN	将一个表的每一个记录和另一表的每个记录匹配成新的数据行

3.6.2 内连接查询

[例3-44] 查询“刘伟”老师所讲授的课程，要求列出教师号、教师姓名和课程号。

(1) 方法1

```
SELECT T.TNo,TN,CNo  
FROM T,TC  
WHERE (T.TNo = TC.TNo) AND (TN='刘伟')
```

这里TN='刘伟'为查询条件，而T.TNo = TC.TNo为连接条件，TNo为连接字段。连接条件的一般格式为：

[<表名1>.] <列名1> <比较运算符> [<表名2>.] <列名2>

(2) 方法2

```
SELECT T.TNo, TN, CNo  
FROM T INNER JOIN TC  
ON T.TNo = TC.TNo  
WHERE TN = '刘伟'
```

(3) 方法3

```
SELECT R1.TNo R2.TN, R1.CNo  
FROM  
(SELECT TNo,CNo FROM TC ) AS R1  
INNER JOIN  
(SELECT TNo ,TN FROM T  
WHERE TN='刘伟') AS R2  
ON R1.TNo=R2.TNo
```

[例3-45] 查询所有选课学生的学号、姓名、选课名称及成绩。

```
SELECT S.SNo,SN,CN,Score  
FROM S,C,SC  
WHERE S.SNo=SC.SNo AND SC.CNo=C.CNo
```

[例3-46] 查询每门课程的课程号、课程名和选课人数。

```
SELECT C.CNO,CN,COUNT(SC.SNo) as 选课人数  
FROM C,SC  
WHERE SC.CNo=C.CNo  
GROUP BY C.CNo,CN
```


3.6.3 外连接查询

符合连接条件的数据将直接返回到结果集中，对那些不符合连接条件的列，将被填上NULL值后再返回到结果集中。

外部连接分为左外部连接和右外部连接两种。以主表所在的方向区分外部连接，主表在左边，则称为左外部连接；主表在右边，则称为右外部连接。

[例3-47] 查询所有学生的学号、姓名、选课名称及成绩（没有选课的同学的选课信息显示为空）。

```
SELECT S.SNo,SN,CN,Score  
FROM S  
LEFT OUTER JOIN SC  
ON S.SNo=SC.SNo  
LEFT OUTER JOIN C  
ON C.CNo=SC.CNo
```

3.6.4 交叉查询

交叉查询（CROSS JOIN）相当对连接查询的表没有特殊的要求，任何表都可以进行交叉查询操作。

[例3-48] 对学生表**S**和课程表**C**进行交叉查询。

```
SELECT *  
FROM S CROSS JOIN C
```

3.6.5 自连接查询

[例3-49] 查询所有比“刘伟”工资高的教师姓名、工资和刘伟的工资。

方法1:

```
SELECT X.TN,X.Sal AS Sal_a,Y.Sal AS Sal_b  
FROM T AS X ,T AS Y  
WHERE X.Sal>Y.Sal  
AND Y.TN='刘伟'
```

方法2:

```
SELECT X.TN, X.Sal,Y.Sal  
FROM T AS X INNER JOIN T AS Y  
ON X.Sal>Y.Sal  
AND Y.TN='刘伟'
```

在WHERE子句中包含一个形包含子查询的语句称为父查询或外部查询。如SELECT-FROM-WHERE的查询块，此查询块称为子查询或嵌套查询，包含子查询的语句称为父查询或外部查询。

3.7.1 普通子查询

返回一个值的子查询

使用比较运算符
(=, >, <, >=, <=, !=)

[例3-51] 查询与“刘伟”老师职称相同的教师号、姓名。

```
SELECT TNo,TN
FROM T
WHERE Prof= (SELECT Prof
              FROM T
              WHERE TN= '刘伟')
```


返回一组值的普通子查询

使用**ANY**或**ALL**

- 使用**ANY**

[例3-52] 查询讲授课程号为C5的教师姓名。

```
SELECT TN FROM T
WHERE TNo = ANY (SELECT TNo
                  FROM TC
                  WHERE CNo = 'C5')
```



```
SELECT TN
FROM T,TC
WHERE T.TNo=TC.TNo
AND TC.CNo= 'C5 '
```

[例3-53] 查询其他系中比计算机系某一教师工资高的教师的姓名和工资。

```
SELECT TN, Sal
FROM T
WHERE (Sal > ANY ( SELECT Sal
                    FROM T
                    WHERE Dept = '计算机'))
AND (Dept <> '计算机')
```



```
SELECT TN, Sal
FROM T
WHERE Sal > ( SELECT MIN(Sal)
              FROM T
              WHERE Dept = '计算机')
AND Dept <> '计算机'
```

- **使用IN**

使用IN代替 “=ANY”

[例3-54] 查询讲授课程号为C5的教师姓名（使用IN）。

```
SELECT TN  
FROM T  
WHERE (TNo IN ( SELECT TNo  
                  FROM TC  
                  WHERE CNo = 'C5'))
```


- 使用ALL

[例3-55] 查询其他系中比计算机系所有教师工资都高的教师的姓名和工资。

```
SELECT TN, Sal
FROM T
WHERE (Sal > ALL ( SELECT Sal
                   FROM T
                   WHERE Dept = '计算机'))
AND (Dept <> '计算机')
```

Sal > (SELECT MAX(Sal)

3.7.2 相关子查询

相关子查询的执行顺序是：首先选取父查询表中的第一行记录，内部的子查询利用此行中相关的属性值进行查询，然后父查询根据子查询返回的结果判断此行是否满足查询条件。如果满足条件，则把该行放入父查询的查询结果集合中。重复执行这一过程，直到处理完父查询表中的每一行数据。

使用EXISTS

- 带有**EXISTS**的子查询不返回任何实际数据，它只得到逻辑值“真”或“假”。
- 当子查询的查询结果集合为非空时，外层的**WHERE**子句返回真值，否则返回假值。
- **NOT EXISTS**与此相反。

[例3-57] 用含有**EXISTS**的语句完成例3-52的查询，即查询讲授课程号为**C5**的教师姓名。

```
SELECT TN
FROM T
WHERE EXISTS ( SELECT *
                FROM TC
                WHERE TNo = T.TNo AND CNo = 'C5')
```


[例3-59] 查询选修所有课程的学生姓名。

```
SELECT SN
FROM S
WHERE (NOT EXISTS ( SELECT *
                     FROM C
                     WHERE NOT EXISTS ( SELECT *
                                       FROM SC
                                       WHERE SNo = S.SNo
                                       AND CNo = C.CNo)))
```

S	C	SC
→ S ₁	→ C ₁	→ S ₁ C ₁
	C ₂	S ₁ C ₂
S ₂	C ₃	S ₁ C ₃
		S ₂ C ₁
		S ₂ C ₂

3.8.1 集合运算查询

- 合并查询就是使用UNION操作符将来自不同查询的数据组合起来，形成一个具有综合信息的查询结果，UNION操作会自动将重复的数据行剔除。
- 参加合并查询的各子查询的使用的表结构应该相同，即各子查询中的数据数目和对应的数据类型都必须相同。

[例3-60] 从SC数据表中查询出学号为“S1”同学的学号和总分，再从SC数据表中查询出学号为“S5”的同学的学号和总分，然后将两个查询结果合并成一个结果集。

```
SELECT SNo AS 学号, SUM(Score) AS 总分  
FROM SC  
WHERE (SNo = 'S1')  
GROUP BY SNo  
UNION  
SELECT SNo AS 学号, SUM(Score) AS 总分  
FROM SC  
WHERE (SNo = 'S5')  
GROUP BY SNo
```

3.8.2 存储查询结果到表中

- 使用SELECT...INTO 语句可以将查询结果存储到一个新建的数据库表或临时表中。

[例3-61] 从SC数据表中查询出所有同学的学号和总分，并将查询结果存放到一个新的数据表Cal_Table中。

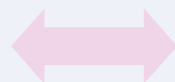
```
SELECT SNo AS 学号, SUM(Score) AS 总分  
INTO Cal_Table  
FROM SC  
GROUP BY SNo
```

3.9.1 添加数据表中的数据

- 用Management Studio添加数据

不能应付数据的大量添加

- 用SQL命令添加数据



INSERT INTO

添加一行新记录

[例3-63] 在SC表中添加一条选课记录('S7', 'C1')。

```
INSERT INTO SC (SNo, CNo)  
VALUES ('S7', 'C1')
```

添加多行记录

```
INSERT INTO <表名> [(<列名1>[,<列名2>...])]  
子查询
```


[例3-64] 求出各系教师的平均工资，把结果存放在新表AvgSal中。

首先建立新表AvgSal，用来存放系名和各系的平均工资。

```
CREATE TABLE AvgSal  
( Department VARCHAR(20),  
  Average SMALLINT)
```

然后利用子查询求出T表中各系的平均工资，把结果存放在新表AvgSal中。

```
INSERT INTO AvgSal  
SELECT Dept,AVG(Sal)  
FROM T  
GROUP BY Dept
```

3.9.2 修改数据表中的数据

- 用Management Studio修改数据
不能应付数据的大量修改
- 用SQL命令修改数据  UPDATE

```
UPDATE <表名>  
SET <列名>=<表达式> [,<列名>=<表达式>]...  
[WHERE <条件>]
```

修改一行

[例3-65] 把刘伟老师转到信息系
UPDATE T
SET Dept= '信息'
WHERE SN= '刘伟'

修改多行

[例3-66] 将所有学生的年龄增加1岁
UPDATE S
SET Age=Age+1
[例3-67] 把教师表中工资小于或等于
1000元的讲师的工资提高20%。
UPDATE T
SET Sal = 1.2 * Sal
WHERE (Prof = '讲师 ')
AND (Sal <= 1000)

用子查询选择要修改的行

[例3-68] 把讲授C5课程的教师的岗位津贴增加100元。

```
UPDATE T
SET Comm = Comm + 100
WHERE (TNo IN (SELECT TNo
                FROM T, TC
                WHERE T.TNo =
                TC.TNo AND TC.CNo = 'C5'))
```

用子查询提供要修改的值

[例3-69] 把所有教师的工资提高到平均工资的1.2倍。

```
UPDATE T
SET Sal = (SELECT 1.2 * AVG(Sal)
           FROM T)
```

3.9.3 删除数据

- 用Management Studio删除数据
比较适合于少量记录等简单情况
- 用SQL命令删除数据 ↔ DELETE

```
DELETE  
FROM<表名>  
[WHERE <条件>]
```

删除一行记录

[例3-70] 删除刘伟老师的记录。

```
DELETE  
FROM T  
WHERE TN= '刘伟'
```

删除多行记录

[例3-71] 删除所有教师的授课记录。

```
DELETE  
FROM TC
```

利用子查询选择要删除的行

[例3-72] 删除刘伟老师授课的记录。

```
DELETE  
FROM TC  
WHERE (TNo =  
        ( SELECT TNo  
          FROM T  
          WHERE TN = '刘伟'))
```

3.10 视图

视图是一个虚拟表，其内容由查询定义。同基本表一样，视图包含一系列带有名称的列和行数据。视图在数据库中并不是以数据值存储集形式存在。行和列数据来自定义视图的查询所引用的基本表，并且在引用视图时动态生成。

3.10.1 创建视图

- 用 Management Studio 创建视图
- 用 SQL 命令创建视图

```
CREATE VIEW view_name [ (column [ ,...n ] ) ]  
[ WITH <view_attribute> [ ,...n ] ]  
AS select_statement  
[ WITH CHECK OPTION ] [ ; ]  
<view_attribute> ::=  
{  
    [ ENCRYPTION ]  
    [ SCHEMABINDING ]  
    [ VIEW_METADATA ] }
```

[例3-73] 创建一个计算机系教师情况的视图Sub_T。

```
CREATE VIEW Sub_T  
    AS SELECT TNo, TN, Prof  
    FROM T  
    WHERE Dept = '计算机'
```


[例3-74] 创建一学生情况视图**S_SC_C**（包括学号、姓名、课程名及成绩）。

```
CREATE VIEW S_SC_C(SNo, SN, CN, Score)  
AS SELECT S.SNo, SN, CN, Score  
FROM S, C, SC  
WHERE S.SNo = SC.SNo AND SC.CNo = C.CNo
```

[例3-75] 创建一学生平均成绩视图**S_Avg**。

```
CREATE VIEW S_Avg(SNo, Avg)  
AS SELECT SNo, Avg(Score)  
FROM SC  
GROUP BY SNo
```

3.10.2 修改视图

- 用 Management Studio修改视图
- 用SQL命令修改视图

```
ALTER VIEW <视图名>[(<视图列表>)]  
AS <子查询>
```

[例3-76] 修改学生情况视图S_SC_C（包括姓名、课程名及成绩）。
ALTER VIEW S_SC_C(SN, CN, Score)
AS SELECT SN, CN, Score
FROM S, C, SC
WHERE S.SNo = SC.SNo AND SC.CNo = C.CNo

3.10.3 删除视图

- 用 Management Studio删除视图
- 用SQL命令删除视图

DROP VIEW <视图名>

[例3-77] 删除计算机系教师情况的视图Sub_T。

DROP VIEW Sub_T

3.10.4 查询视图

- 视图定义后，对视图的查询操作如同对基本表的查询操作一样。

[例3-78] 查询视图Sub_T中职称为教授的教师号和姓名。

```
SELECT TNo, TN  
FROM Sub_T  
WHERE Prof = '教授'
```



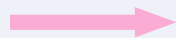
```
SELECT TNo, TN  
FROM T  
WHERE Dept = '计算机'  
AND Prof = '教授'
```

视图的建立简化了查询操作

3.10.5 更新视图

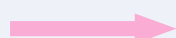
- 由于视图是一张虚表，所以对视图的更新，最终转换成对基本表的更新。
- 其语法格式如同对基本表的更新操作一样。

添加



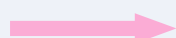
INSERT

修改



UPDATE

删除



DELETE

3.11.1 索引概述

3.11.2 索引的类型

聚集索引

非聚集索引

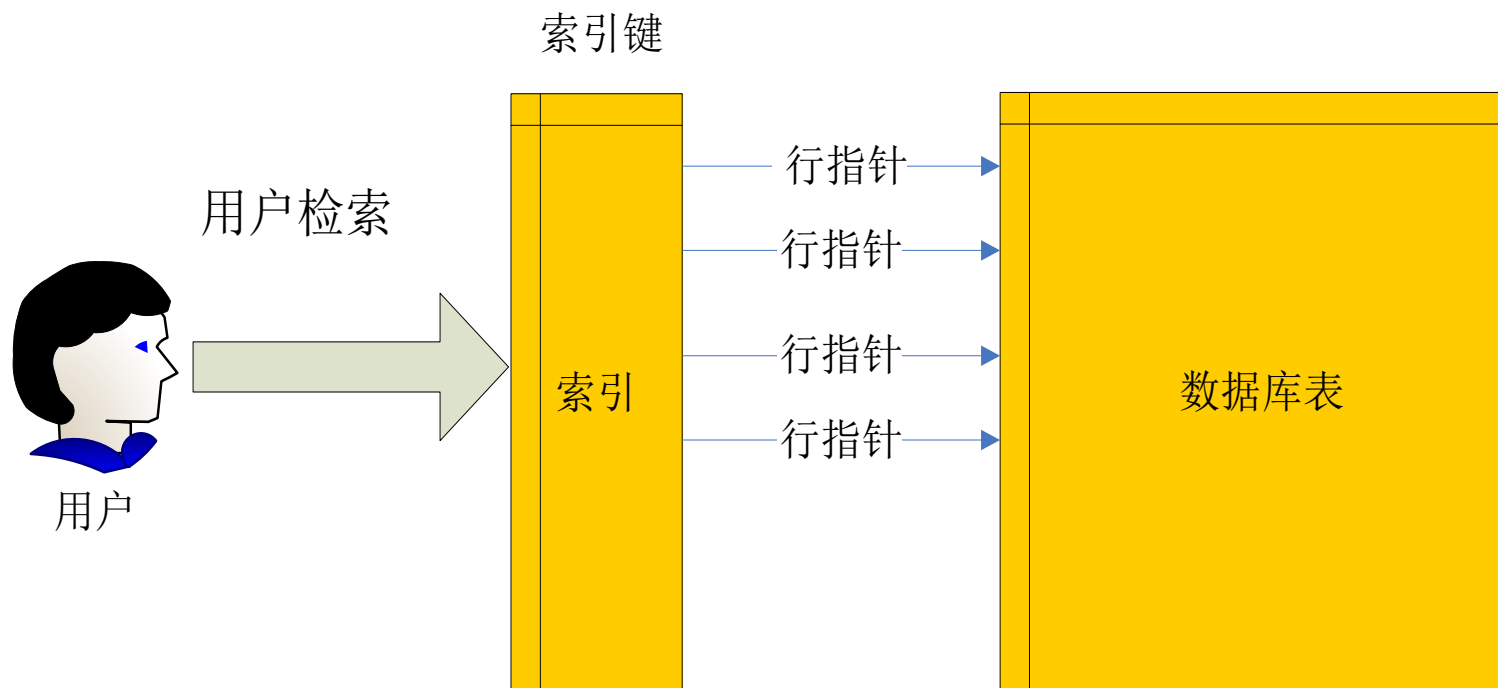
唯一索引

视图索引

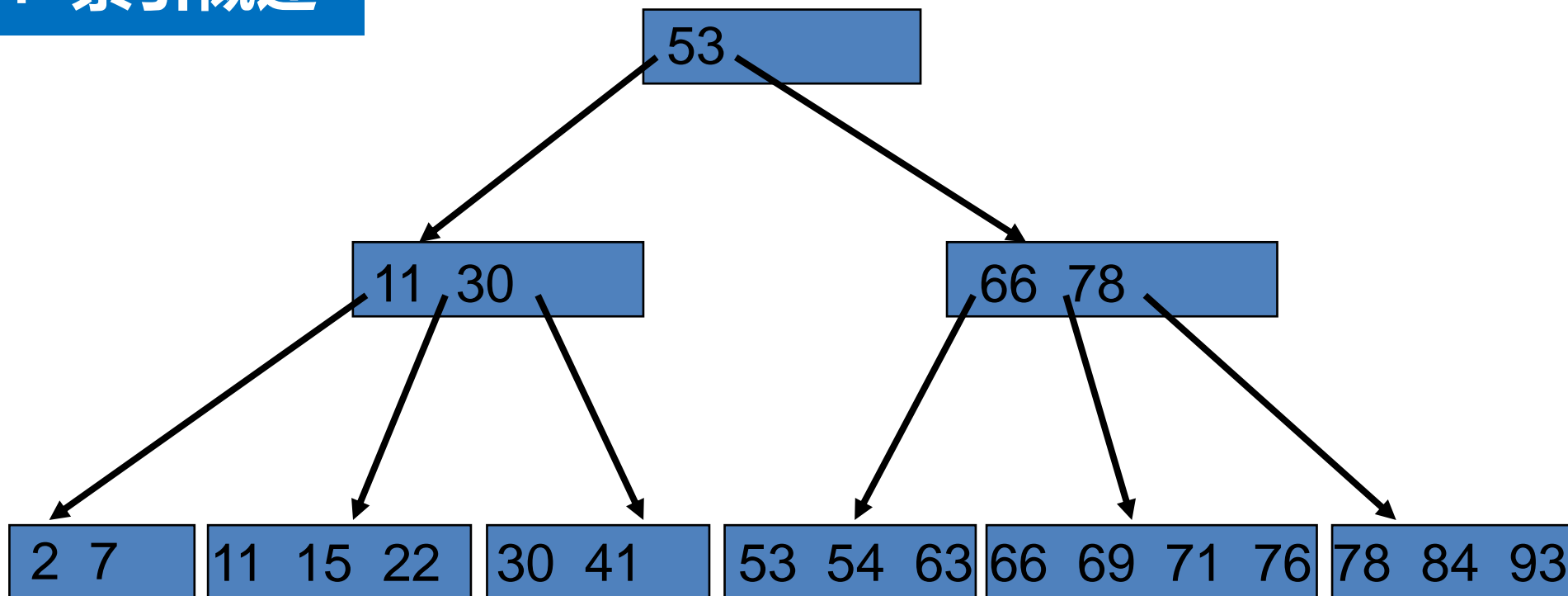
全文索引

XML索引

3.11.1 索引概述



3.11.1 索引概述



B+-Tree of order 2

each node can hold up to four keys

3.11.3 创建索引

- 用Management Studio创建索引
- 用SQL语句创建索引

[例3-82] 为表SC在SNo和CNo上建立惟一索引。
CREATE UNIQUE INDEX SCI ON SC(SNo,CNo)

[例3-83] 为教师表T在TN上建立聚集索引。
CREATE CLUSTER INDEX TI ON T(TN)

3.11.4 修改索引

修改索引的SQL命令
语法如下：

```
ALTER INDEX { index_name | ALL }  
ON table_or_view_name  
{ REBUILD  
  [ [PARTITION = ALL]  
    [ WITH ( <rebuild_index_option> [ ,...n ] ) ]  
  | [ PARTITION = partition_number  
    [ WITH ( <single_partition_rebuild_index_option>  
      [ ,...n ] )  
    ] ] ]  
  | DISABLE  
  | REORGANIZE  
    [ PARTITION = partition_number ]  
    [ WITH ( LOB_COMPACTION = { ON | OFF } ) ]  
  | SET ( <set_index_option> [ ,...n ] )  
}[ ; ]
```

3.11.5 删除索引

- 用Management Studio删除索引
- 用SQL语句删除索引

```
DROP INDEX <table or view name>.<index name>  
DROP INDEX <index name> ON <table or view name>
```

3.11.6 查看索引

- 用Management Studio 查看索引
- 用Sp_helpindex存储过程查看索引

Sp_helpindex [@objname =] 'name'

→ 表的名称

[例3-84] 查看表SC的索引。

```
EXEC Sp_helpindex SC
```

如果要更改索引名称，可利用**Sp_rename**存储过程更改，其语法如下：
Sp_rename '数据表名.原索引名', '新索引名'

[例3-85] 更改T表中的索引TI名称为T_Index。
EXEC Sp_rename 'T.TI', 'T_Index'

SQL的动词

SQL功能	动 词
数据定义	CREATE、DROP、 ALTER
数据查询	SELECT
数据操纵	INSERT、UPDATE、 DELETE
数据控制	GRANT、REVOKE