前　言

在 IT 技术高速发展，互联网已渗透至千家万户的今天，越来越多的商务应用程序都要求用户访问存储于中枢位置的数据，就使得数据库技术始终居于软件开发的核心地位。任何一个投入运行的信息系统中，都会有一个基于客户机/服务器的数据库管理系统作为支撑。各种商务实践使人们认识到，通过精简和分散决策过程从而把信息及时传递给用户具有重要意义。利用设计开发好的数据库管理系统，可以就丰富的原始数据根据具体需求进行复杂的条件查询、分析和汇总，最终给出有远见的商务决策，对于高层管理人员的运筹帷幄起着至关深远的影响。

本书分为管理维护篇、SQL 基础篇和 Transact-SQL 高级篇。

管理维护篇包含第 1 章 SQL Server 概述、第 2 章 数据库的日常管理，适合数据库维护人员阅读和参考。

SQL **基础篇**包含第 3 章 用 E-R 图设计关系数据库、第 4 章 规范化数据及范式、第 5 章 创建数据库、第 6 章 创建并管理表、第 7 章 实施数据完整性、第 8 章 SQL 查询和第 9 章 维护数据库的数据，是数据库设计开发的基本内容，适合于设计开发数据库的初学者以及大、中专院校的学生使用。

Transact-SQL **高级篇**包含第 10 章 索引与视图、第 11 章 Transact-SQL 的编程、第 12 章 游标、第 13 章 存储过程、第 14 章 事务和第 15 章 触发器，是完善优化数据库性能的高级技巧，适合于具有一定基础的数据库开发人员，或者是以数据库为主攻方向的硕士研究生作参考。

本书中所涉及的代码来源于课堂教学的例题和项目实践，均在 SQL Server 环境下顺利通过调试，并能成功运行。为便于读者更好地巩固和掌握书中的重点及难点，作者还给出了习题。每个章节里多处谈到的注意事项，都是同事和我从多年的工作实践中所得出的宝贵经验，尤其是附录里的经验总结——谈如何从设计入手，全盘考虑数据库的开发与实现，希望同行能借鉴而避免走弯路。

光盘里包含作者曾设计开发过的大型数据库系统——某跨国公司人事招聘管理系统和某电子商务应用系统的完整源代码，且这些代码已由作者编译成自动批处理程序。该.exe 文件可以直接运行安装为带有实体样例的数据库供验证例题与习题中的程序代码。

漂泊在外十三载，忙碌于一个又一个课题和项目的脚步一直未能停歇，终于借陪侍父亲于病榻之闲暇，得以静思和感悟人生。也许明天，亲人就会离逝、感情将遭遇聚变、意外可能突袭，尽管悲痛无限，却依然要努力地活着，让生活更灿烂、工作更美丽。同时，也把自己近十年来切身的开发经验和始终在一线的教学经验梳理总结为此书，倾力奉献给厚爱我的人们。

欢迎读者朋友用 E-mail（liulixia1975@126.com）与我一起讨论本书，并给出诚挚的意见。

末了，以“非淡泊无以明志，非宁静无以致远”作为结束来抚慰自己，并与读者朋友共勉。

刘丽霞

2006 年 12 月 21 日于北京

需要全本请访问 www.ertongbook.com

序　言

在科教兴国方针的指引下，高等教育进入了一个新的历史发展时期，招生规模和毕业生人数有了大幅度的增长。与此同时，高等院校计算机基础教育的研究工作不断深化，并已取得许多重要成果。

多年来，全国高校从事计算机基础教育的老师们，始终不渝地在为高等院校计算机教育工作辛勤劳作，深入探索，努力开拓，积累了丰富的教学经验，初步形成了一套行之有效的课程体系和教学理念。其发展经历了 3 个阶段：20 世纪 80 年代的初级阶段、20 世纪 90 年代的规范阶段以及进入 21 世纪以来的深化与提高阶段。当然，在进入计算机基础教育新阶段的同时，我们也充分认识到所面临的挑战：高等院校计算机基础教育必须跟上信息技术发展的潮流，大力更新教学内容，用信息技术的新成就武装当代大学生，为我国国民经济与社会信息化的进程，培养一大批能熟练使用计算机，并能将信息技术应用于本领域的新型专业人才。因此，大力提高我国高等学校计算机基础教育的水平，培养造就出符合 21 世纪信息时代要求的专业人才，已成为广大计算机教育工作者的神圣使命和光荣职责。

在“九五”和“十五”期间，两届计算机基础课程教学指导委员会都把计算机教材建设列入重点工作。为此，各院校在计算机教学改革过程中，把如何实现自己的培养目标以及如何选择适用的教材作为首要任务。掌握先进的计算机知识，无疑是培养新型人才的一个重要环节。计算机知识不仅是现代科学技术的结晶，还将成为大众化的智能工具。学习计算机知识不仅能掌握一种技能，实现其应用的价值，更重要的是能启发人们对先进科技的向往，激发创新意识，培养动手能力，锻炼实践本领。

在西北工业大学出版社的大力支持和配合下，汇集众多高校名师于 2003 年成立的高等院校计算机基础教育规划教材研究与编审委员会，多年来一直致力于研究在新形势下，如何编写出版适应教学需要的教材；集中讨论了教育部计算机基础课程的许多重大的教学改革举措、新的课程体系框架、教学内容组织和课程设置等；经过与各高校老师、专家反复研讨，取得许多共识。在此基础之上，整体规划，设计开发出了编委会组织推荐的新的“21 世纪高等院校计算机基础教育规划教材”。

本系列教材有以下特点：

（1）内容上严格把关，并与“面向 21 世纪课程教材”“普通高等教育‘十一五’国家级规划教材”有机衔接，符合教育部高等学校计算机科学与技术教学指导委员会、非计算机专业计算机基础课程教学指导分委员会“关于进一步加强高等学校计算机基础教学的几点意见”（白皮书）的精神。

（2）吸收了国内外众多同类教材的优点，并充分考虑到教材面向不同办学层次、学科、地域和人才培养模式的差异性，以满足各种层次和类型的教学急需。本系列教材既有供理工科类专业用的，也有供文科和经济类专业用的；既有必修课的教材，也包括一些选修课教材。

（3）立足创新，以“新颖、清晰、实用、通俗、配套”为原则，全面考虑广大读者及

各高校在新形势下的实际教学需求，有较好的可读性和可用性。

本系列教材统一规划，分批组织，陆续出版，希望能对培养信息技术人才，推动国民经济和社会发展作出一定的贡献。

2001—2005 年教育部高等学校计算机科学与技术教学指导委员会副主任

非计算机专业计算机基础课程教学指导分委员会主任委员



第 9 章　维护数据库的数据

一旦创建了数据库和表，就能在数据库里存储数据了。然而，存储数据只是维护数据任务的一部分，也可以根据需要修改或删除数据。通常，数据库里的数据维护工作包括插入、修改和删除操作。SQL Server 提供了专门的数据操纵语言，主要用于处理数据。

本章要点

（1）添加数据。

（2）修改数据。

（3）删除数据。

9.1　存储数据

用 SQL Server 提供的 INSERT 语句可以往表里插入行记录，以存储数据，其语法为

```
INSERT [INTO]  表名  [列列表]
VALUES（缺省值 | 值列表 | select 语句）
```

参数说明：

- 表名

是要插入行记录的表名。

- 列列表

当只插入部分列数据或要以不同的列顺序插入值时，需要指定该参数。

- 缺省值

是用于插入缺省列值的子句。如果列上没有定义缺省值且该列是允许 NULL 值的，则会为该列插入 NULL 值。如果列上没有定义缺省且该列是不允许 NULL 值的，则会返回错误信息，INSERT 操作失败。

- 值列表

是要插入的列值的列表，它们将组成新行。如果列上缺省值，可以用 DEFAULT 关键字来代替。值也可以是表达式。

- select 语句

是嵌套的 SELECT 语句，可用于往表里插入行。

使用 INSERT 语句时，应当遵循下面的一些规则：

（1）数值的个数必须和表中或列列表中的属性个数相同。

（2）插入信息的次序必须和插入列表中列出的属性次序相同。

（3）信息的数据类型必须和表列中的数据类型匹配。

（4）对 text 或 image 列插入 NULL 值，不能创建有效的文本指针，也不能创建 8K 页。

（5）违反约束条件、默认值或规则，或插入错误的数据类型值，都将导致发生错误。

（6）插入到 varchar 类型列的数据将去除尾空格。

9.1.1　向表中插入数据

添加到数据库中的最小数据单位是行。下面通过例题来讲解如何向数据库里插入数据。

【例 9-1】候选人的招聘通过招聘机构进行。一个新的招聘机构叫“Head Hunters”。Head Hunters 公司的信息如图 9-1 所示，需要将上述信息存储在表 RecruitmentAgencies 中。

Attributes	Data
Agency Code	0010
Name	Head Hunters
Address	223 Hill Street
City	Cleveland
State	Ohio
Zip	44167-5943
Phone	(440)345-8872
Fax	(440)345-8943
Charge	7
Total Paid	1000

图 9-1　一个招聘机构的信息

代码如下：

```
INSERT INTO RecruitmentAgencies
VALUES ('0010','Head Hunters', '223 Hill Street', 'Cleveland', 'Ohio', '44167-5943',
       '(440)345-8872', '(440)345-8943',7,1000)
```

可以验证：

```
SELECT *
FROM RecruitmentAgencies
```

【例 9-2】一个新的合同招聘人和公司签了合同。合同招聘人的信息如图 9-2 所示，将上述信息存储在表 ContractRecruiter 中。

属性	数据
Contract Recruiter Code	0007
Name	George Craig
Address	56, Burgot Street
City	Alexandria
State	Virginia
Zip	22334-6354
Fax	(703)654-1223
Phone	(703)654-2344
Percentage Charge	10
Total Paid	1000

图 9-2　合同招聘人的信息

需要全本请访问 www.ertongbook.com

代码如下：

```
INSERT INTO    ContractRecruiter
VALUES (’0007’,’George Craig’, ’56 Burgot Street’, ’Alexandria’, ’Virginia’, ’22334-6354’,
’(703)654-1223’, ’ (703)654-2344’,10,1000)
```

9.1.2　将现有数据存入新表

可以使用 SELECT INTO 语句将一个表的数据插入到另一个表，其语法为

```
SELECT  列列表
INTO  新表名
FROM  表名
WHERE  条件
```

参数说明：

- 列列表

指定了新表中要包含的列。

- 新表名

指定了要存储数据的新表的名字。

- 表名

指定了要从中检索数据的表的名字。

- 条件

是决定新表中应包含哪些行的条件。

【例 9-3】根据表 Titles 创建一个叫 NewTitles 的新表，该表是单价大于$15 的所有书的 ID 和书名。

代码如下：

```
SELECT cTitleId, cTitle
INTO NewTitles
FROM Titles
WHERE iPrice > 15
```

9.1.3　现有数据的拷贝

也可以使用 INSERT INTO 语句从一个表向另一个表添加数据，其语法为

```
INSERT [INTO]  表名 1
SELECT  列名
FROM  表名 2
[WHERE  条件]
```

参数说明：

- 表名 1

指定了将要插入数据的表的名字。

- 列名

指定了需要从现有表复制数据到新表的列的名字。

- 表名 2

指定了从中复制数据的表。

- 条件

指定了插入的行要满足的条件。

【例 9-4】需要将费用小于$19 的人才交流会的信息从表 JobFair 复制到表 PopularJobFair（表 PopularJobFair 是不存在的）。所以创建了包含所需数据的表 PopularJobFair 之后，可以添加记录。

代码如下：

```
INSERT INTO PopularJobFair
SELECT *
FROM JobFair
WHERE mFee < 19
```

9.2　修改数据

SQL Server 提供了 UPDATE 语句来对先前已经插入到表中的数据进行修改。利用该语句，可以修改现有行中的数据，也可以添加新的数据。

使用 UPDATE 语句的语法如下：

```
UPDATE 表名
SET 列名 = 值 [,列名 = 值]
[WHERE 条件]
```

参数说明：

- 表名

指定要修改的表或视图的名字。

- 列名

指定要修改的列的名字。

- 值

指定要修改列的值。可以是表达式、列名、变量名等，也可以用 DEFAULT 和 NULL 关键字。

- 条件

指定要更新哪些行。

【例 9-5】Peter Prescott 从财务部转到市场部。根据该信息更新表 Employee。

代码如下：

```
UPDATE Employee
SET cDepartmentCode =’0003’
WHERE cEmployeeCode=’000015’
```

可以验证：

```
SELECT * from Employee
```

```
WHERE cEmployeeCode='000015'
```

【例 9-6】Jane Schaffer(cCandidateCode 000049)的测试成绩已经宣布，她得了 78 分。同时，需要在表 ExternalCandidate 中更新她的成绩。

代码如下：

```
UPDATE ExternalCandidate
SET siTestScore=78
WHERE cCandidateCode='000049'
```

可以验证：

```
SELECT * FROM ExternalCandidate
WHERE cCandidateCode='000049'
```

【例 9-7】由于在试卷复查过程中发现了批改错误，外部候选人 Jane Schaffer (cCandidateCode 000049) 的测试成绩需要增加 2 分。

代码如下：

```
UPDATE ExternalCandidate
SET siTestScore=siTestScore+2
WHERE cCandidateCode='000049'
```

可以验证：

```
SELECT *FROM ExternalCandidate
WHERE cCandidateCode='000049'
```

9.3　删除数据

SQL Server 允许从表中删除无用的数据，下面进行具体介绍。

9.3.1　删除行

使用 DELETE 语句可以从表中删除行。在操作过程中，一次删除一行，将每行的删除过程记录在事务日志中，并回收数据占用的空间。

使用 DELETE 语句的语法如下：

```
DELETE [FROM]  表名
[FROM  表]
[WHERE  条件]
```

参数说明：

- 表名

是要删除行所在的表的名字。

- 条件

指定要删除哪些行。

【例 9-8】公司有许多旧数据，这些数据是关于一些在测试中被淘汰的外部候选人的。这些数据

已有两年之久。现在，需要确保这些旧数据从表 ExternalCandidate 上删除。

代码如下：

```
DELETE FROM ExternalCandidate
WHERE dTestDate < DATEADD(yy,-2,getdate())
```

可以验证：

```
SELECT * FROM ExternalCandidate
WHERE dTestDate < DATEADD(yy,-2,getdate())
```

【例 9-9】需要从表 InternalJobPosting 中删除超过 6 个月的有关内部职位提供的信息。

代码如下：

```
DELETE InternalJobPosting
WHERE dNoticeReleaseDate< DATEADD(mm,-6,getdate())
```

可以验证：

```
SELECT * FROM InternalJobPosting
WHERE dNoticeReleaseDate< DATEADD(mm,-6,getdate())
```

【例 9-10】到了年底，所有关于内部候选人的信息都应从表 InternalCandidate 中删除。

代码如下：

```
DELETE InternalCandidate
```

可以验证：

```
SELECT * FROM InternalCandidate
```

9.3.2　清空表中数据

使用 TRUNCATE 语句能删除表里的所有数据。TRUNCATE 和 DELETE 语句在功能上是相同的。然而，TRUNCATE 执行速度更快。另外，TRUNCATE 语句并不解除作用于该对象的触发器。

使用 TRUNCATE 语句的语法：

```
TRUNCATE TABLE 表名
```

参数说明：

- 表名

是要清空数据的表的名字。

【注意】TRUNCATE TABLE 并没有 WHERE 子句。

【例 9-11】从 Categories 表删除所有的行记录。

代码如下：

```
TRUNCATE TABLE Categories
```

9.3.3　删除带数据的表

使用 DROP 语句可以删除表。这时，表里所有的数据被永久地删除了。不能用它来删除系统表。

DELETE 与 DROP 语句的区别在于：

（1）DELETE 删除的是表中的记录，表仍存在。

（2）DROP 删除记录连同表结构，表不存在。

【例 9-12】删除表 Item 及表里的数据。

代码如下：

```
DROP TABLE Item
```

本章小结

本章对数据库的插入、更新和删除等维护方面的操作进行了介绍，并比较了几种删除操作的不同。

（1）INSERT 语句可以往表里插入行记录，以存储数据。

（2）UPDATE 语句可以通过更新行记录来修改表里的值。

（3）使用 DELETE 语句可以从表中删除行。

（4）TRUNCATE 语句能更快地删除表里的所有数据，以迅速清空表。

（5）DROP 语句能删除行记录连同表结构。

习 题 九

1．在数据库 Toys 中存储下列品牌，如图 9-3 所示。

Brand Id	Brand Name
001	Bobby
002	Frances-Price
003	Bernie Kids
004	Largo

图 9-3　要添加的品牌

2．将下列种类的玩具存储在数据库中，如图 9-4 所示。

CategoryId	Category	Deasription
001	Activity	Activity toys encourage the children social skills and interest in the world around them.
002	Dolls	A wide range of dolls from all the leading brands.
003	Arts And Crafts	Encourage children to create masterpieces with these incredible craft kits.

图 9-4　要存储的玩具

3．将下列信息存储到数据库中，如图 9-5 所示。

4．将玩具代码为 000001 的玩具的 Toy Rate 增加$1。

5．从数据库中删除名称为 Largo 的品牌。

6．将种类为 Activity 的信息复制到一张新表中。该表叫 PreferredCategory。

7．将种类为 Dolls 的信息从表 Category 复制到表 PreferredCategory。

属性名	数据
Toy Id	000001
Toy Name	Robby the Whale
Toy Description	A giant Blue Whale with two heavy-duty handles that allow a child to ride on its back.
Category Id	001
Toy Rate	8.99
Brand Id	001
Photo	NULL
Toy Quantity on Hand	50
Lower Age	3
Upper Age	9
Toy Weight	1
Toy Image Path	NULL

图 9-5　要存储的一条记录

需要全本请访问 www.ertongbook.com

第三部分

Transact-SQL 高级篇

本部分要点

Transact-SQL 高级篇涉及 SQL 的交互编程，使用结构化程序设计的那些通用的流程控制语句来编程，以批的形式提交给系统进行处理与执行。

往往复杂的大型数据库对系统的性能要求尤为重要。本篇主要围绕着设计人员如何优化和提高系统性能来具体讲解索引、视图、游标、存储过程、事务和触发器的实现方法。

本部分主要内容

- 索引与视图
- 游标
- 存储过程
- 事务
- 触发器



第 10 章　索引与视图

索引和视图是为了提高系统性能而引入的。创建索引以占用磁盘空间为代价，却在一定程度上实现了数据的逻辑排序从而加快了查询速度。视图利用的是将多张表中的数据虚拟成一张表的原理，不仅保护了原始数据，而且提高了查询性能。

本章要点

（1）创建聚集索引和非聚集索引。

（2）视图的定义和优缺点。

（3）创建和删除视图。

10.1　实现索引

索引是一个内部的表结构，是 SQL Server 根据一列或多列的值来实现对表中行的快速访问。索引是对数据库表中一个或多个列（例如，employee 表的姓氏 (lname) 列）的值进行逻辑排序的结构。因为适当的经过规划的索引能通过降低访问时间而极大地提高数据库的性能，例如想按特定职员的姓来查找他或她，则与在表中搜索所有的行相比，索引有助于更快地获取信息。

索引提供指针以指向存储在表中特定列的数据值，然后根据指定的排序次序来排列这些指针。数据库使用索引的方式与书使用目录很相似，通过搜索索引找到特定的值，然后跟随指针找到包含该值的行。

通常情况下，只有当经常查询索引列中的数据时，才需要在表上创建索引。索引将占用磁盘空间，并且降低添加、删除和更新行的速度。不过在多数情况下，索引所带来的数据检索速度的优势大大超过它的不足之处。然而，如果应用程序非常频繁地更新数据，或磁盘空间有限，那么最好限制索引的数量。

可基于数据库表中的单列或多列创建索引。当某些行中的某一列具有相同的值时，多列索引能区分开这些行。

经常同时搜索两列或多列，并且按两列或多列排序时，索引很有帮助。例如，如果经常在同一查询中为姓和名两列设置准则，那么在这两列上创建多列索引将很有意义。

简单地说，使用索引的优点是：提高查询执行的速度、强制实施数据的唯一性和提高表之间连接的速度。同时，索引也存在一定的缺点：存储索引要占用磁盘空间、数据修改需要更长的时间（因为索引在更新和创建的时候要花费时间）。

通常作为索引的列最好具备下述条件：

（1）是表的主键或外键。

（2）是查询经常要用到的列。

（3）通常包括 WHERE 子句。

（4）是 ORDER BY 子句中经常要用到的列。

（5）属于经常搜索值的范围。

（6）是 UNION 操作符中的规则表达式。

不适合作为索引的列有：

（1）只包含几个选择值的列，例如 TRUE 或 FALSE，YES 或 NO，male 或 female 等。

（2）查询此列超过表的百分之二十的行被返回。

（3）包含的值超过 900 字节。

为了进一步确定索引的有效性，在设计数据库系统时请注意考虑下面几点：

（1）检查查询中的 WHERE 和 JOIN 子句。在任一子句中包括的每一列都是索引可以选择的对象。

（2）试验新的索引，检查它对执行查询性能的影响。

（3）考虑表中已创建的索引数量。最好不要在一个表中创建大量的索引。

（4）检查表中已创建的索引定义。最好避免包含共享列的重叠索引。

10.1.1　聚集索引和非聚集索引

SQL Server 支持两种类型的索引，即聚集索引和非聚集索引。

聚集索引就是指行的物理次序与行的索引次序相同，即数据被物理排序。一般，只能为每个表创建一个聚集索引。为数据访问而创建的聚集索引就是在一定数据范围基础上进行的查询。这是因为行被物理地存放在同一或相邻的数据页上，很容易进行范围内的检索。

如图 10-1 所示是聚集索引的含义。

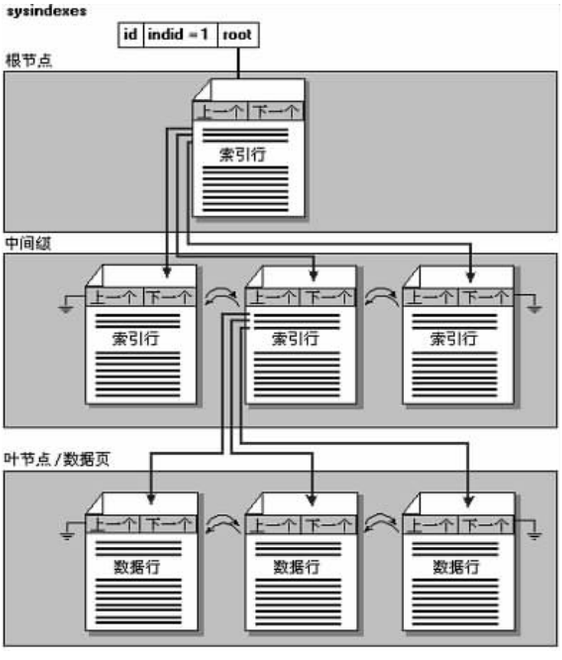


图 10-1　聚集索引

从图 10-1 可以看到，聚集索引每个表只能有一个，而且表中的行顺序和索引中的行顺序相同。同时这样也可以维护键值的唯一性。

在设计和创建聚集索引时，应遵循下面的规则：

（1）因为一个表只能定义一个聚集索引，所以要在唯一值频率高的列上来建立。

（2）如果要创建非聚集索引，则必须先创建聚集索引。

（3）创建聚集索引时使用 FILLFACTOR 选项，可以确保新行插入时有足够的可用空间。

下面来讲解聚集索引是如何工作的。

在一个聚集索引中，数据被存储在 B 树的叶层。B 树的结构可以看成是一个档案柜，表的数据页就像是以字母的顺序存储在档案柜里的文件夹，而数据就像是存储在文件夹里的文件。SQL Server 在使用聚集索引检索数据时，按照下列步骤来进行：

（1）SQL Server 从系统表 sysindexes 获得根页地址。

（2）比较检索值与根页上的关键字。

（3）找出该页上的小于或等于检索值的最大关键字值。

（4）沿循页指针到索引的下一层。

（5）重复（3）、（4）步，直到达到数据页的底部。

（6）检索数据页上的行，查找匹配检索值的记录。如果在该页上未发现匹配记录，则查询没有返回的行。

非聚集索引是 SQL Server 默认的索引类型，即行的物理顺序和索引顺序不同。表的逻辑次序由索引指定，而数据行可以在整个表中随机分布。非聚集索引是通过访问页指针来收集数据的。非聚集索引树包含排序的索引键和叶层，包含指向数据页的指针和数据页的行编号。每个表最多可以有 249 个非聚集索引。

如图 10-2 所示是非聚集索引的含义。

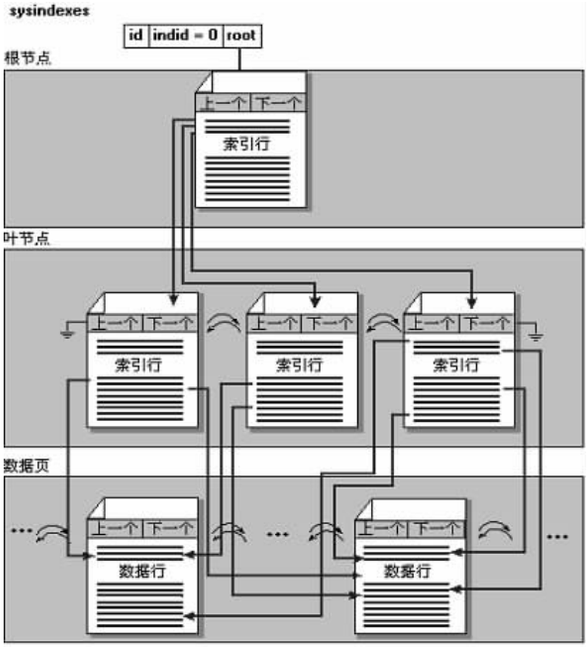


图 10-2　非聚集索引

创建非聚集索引时，应遵循下面的原则：

（1）可以为每个表创建多个非聚集索引，以提供不同方式的数据访问。

（2）如果选定的范围代表很小的数据比率，则使用非聚集索引十分有效。

（3）默认情况下用创建索引的语句来创建非聚集索引。

（4）因为费用开销与每个非聚集索引的维护是相关的，所以仅当要进行时间的有限搜索时才创建它们。

SQL Server 在使用非聚集索引检索数据时，将执行下面的步骤：

（1）SQL Server 从系统表 sysindexes 获得根页地址。

（2）比较检索值与根页上的关键字。

（3）找出该页上的小于或等于检索值的最大关键字值。

（4）沿循页指针到索引的下一层。

（5）重复（3）、（4）步，直到达到数据页的底部。

（6）在页层上查找检索值。如果未发现匹配记录，则表里没有匹配行；如果发现匹配记录，则指针指向数据页和行代码，所需的数据行就找到了。

一般，在现有聚集索引被删除、新的聚集索引被建立和在 DROP_EXISTING 选项改变了定义聚集索引的列的时候，SQL Server 系统自动重建非聚集索引。

10.1.2　索引的特性

一般来说，索引有以下的特性：

（1）索引加速连接多个表的查询，并且执行排序和分组。

（2）索引用于实施行的唯一性。

（3）索引对于多数数据是唯一的列来说是有效的，对于包含大量重复数据的列是无效的。

（4）当要修改索引列上的数据时，有关索引也被自动更新。

（5）维护索引需要时间和资源，设计数据库系统不应该创建不会被频繁使用的索引。

（6）聚集索引应该创建在非聚集索引之前。聚集索引改变行的物理顺序，如果非聚集索引被创建在聚集索引前面，则非聚集索引需要在聚集索引被创建之后被重建。

（7）一般地，聚集索引创建在主关键字上，非聚集索引创建在外关键字上。

10.1.3　创建索引

用 CREATE INDEX 语句可以创建索引，其语法为

```
CREATE [UNIQUE] [CLUSTERED|NONCLUSTERED]
INDEX 索引名
ON 表名(列名[,列名]…)
```

参数说明：

- UNIQUE

创建的索引每行都有不同的索引值。

- CLUSTERED

创建聚集索引，数据以索引的属性来排序。

- NONCLUSTERED

创建非聚集索引，数据只是逻辑排序，而不是物理排序。

● 索引名

指要创建的索引的名字。在一个表上必须是唯一的，但在数据库范围内可以重复。

● 表名

要在其上创建索引的表的名字。

● 列名

指定建立索引的哪些列的名字。

【例 10-1】表 ExternalCandidate 中包含了大量数据。现在所需的报表输出包含候选人的名字和招聘机构的名称。执行下列查询要花费很长的时间：

```
SELECT    vFirstName, cName
FROM ExternalCandidate JOIN RecruitmentAgencies
ON    ExternalCandidate.cAgencyCode = RecruitmentAgencies.cAgencyCode
```

现在要求创建索引以达到优化查询的目的。

代码如下：

```
CREATE CLUSTERED
INDEX idxRecruitment
ON RecruitmentAgencies(cAgencyCode)

CREATE NONCLUSTERED
INDEX idxExternalCandidate
ON ExternalCandidate(cAgencyCode)
```

检验索引是否已经创建，可以用系统的存储过程 sp_helpindex 来验证。

当创建索引之后再执行查询，如果返回的数据记录很多，就能体会到使用索引查询和普通查询在速度上的差异。

【例 10-2】显示大学名称和招聘起始日期的查询执行速度太慢。以下查询需要优化以提高速度：

```
SELECT cCollegeName.dRecruitmentStartDate
FROM College JOIN CampusRecruitment
WHERE College.cCollegeCode = CampusRecruitment.cCollegeCode
```

代码如下：

```
CREATE CLUSTERED
INDEX idxCollege
ON College(cCollegeCode)

CREATE NONCLUSTERED
INDEX idxCampusREcruitment
ON CampusRecruitment(cCollegeCode)
```

可以进行验证：

```
sp_helpindex College
sp_helpindex CampusRecruitment
```

需要全本请访问 www.ertongbook.com