希望
Beijing Hope Electronic Press
www.bhp.com.cn

这是一本作者亲历SQL Server 2005数据库管理及应用开发所有实践细节记录的图书

SQL Server 2005 开发者指南

北京希望电子出版社 总策划
蒲卫 吴豪 编 著

- 全面诠释SQL Server知识难点
- 细化数据库结构与范式设计
- 嵌入XML Web及分布查询技术
- 高效开发C/S、B/S应用程序

直揭数据库技术核心

科学出版社
www.sciencep.com

是一本作者亲历SQL Server 2005数据库管理及应用开发所有实践细节记录的图书

SQL Server 2005 开发者指南

北京希望电子出版社 总策划
蒲卫 吴豪 编 著

- 全面诠释SQL Server知识难点
- 细化数据库结构与范式设计
- 嵌入XML Web及分布查询技术
- 高效开发C/S、B/S应用程序

直揭数据库技术核心

科学出版社
www.sciencep.com

内 容 简 介

这是一本作者亲历 SQL Server 2005 数据库管理及应用开发所有实践细节记录的图书。

本书共分 15 章，内容包括 SQL Server 2005 的开发模式，数据建模，全文检索，事务、锁和分布式查询，视图，约束，脚本和批处理，存储过程，用户自定义函数，触发器，SQL 游标，XML 开发，高级查询以及数据仓库等。系统、全面地介绍了 SQL Server 2005 数据库管理及应用开发技术的所有细节。讲练结合，实用性强。

本书非常适用于已经具备 SQL Server 数据库管理技术和数据库应用开发的中、高级用户，同时也是 Web 数据库设计人员以及项目开发人员必须熟悉的后台数据库管理技能，能够帮助他们快速完成大型系统开发与应用设计。另外，也可作为普通高校、社会各行业技术人才的培训教材。

图书在版编目（CIP）数据

SQL Server 2005 开发者指南 / 蒲卫　吴豪 编著. —北京：科学出版社，2008

（数据库管理及应用超级技术）

ISBN 978-7-03-021717-2

Ⅰ. S...　Ⅱ. ①蒲... ②吴...　Ⅲ. 关系数据库—数据库管理系统，SQL Server 2005—指南　Ⅳ. TP311. 138-62

中国版本图书馆 CIP 数据核字（2008）第 056339 号

责任编辑：但明天　　/责任校对：马　君

责任印刷：双　青　　/封面设计：康　欣

科 学 出 版 社 出版

北京东黄城根北街 16 号

邮政编码：100717

http://www.sciencep.com

双 青 印 刷 厂 印刷

科学出版社发行　各地新华书店经销

*

2008 年 6 月第　一　版　开本：787×1092 1/16

2008 年 6 月第一次印刷　印张：46 1/4

印数：1—3000　字数：1078 896

定价：66.00 元

前　言

本书主要面向具有一定基础的开发者，下面简要介绍一下各章节的内容。

第 1 章将重点集中在 SQL Server 2005 的开发模式，包括应用程序的设计概念、数据库设计概念、语言、数据访问编程、管理编程、管理工具、数据库引擎等，主要介绍开发模式下各种角色所需要掌握的内容。

第 2 章重点讨论数学建模的知识，包括数学建模的理论基础、表和实体，重点探讨范式化设计中的各种概念。

第 3 章涵盖全文检索所需要掌握的内容。包括存储方式、索引定义、创建和删除各种索引（包括 XML 索引）、维护索引及全文检索的体系结构，最后介绍全文索引的设置方法与分类、全文查询语法等。

第 4 章介绍事务、锁和分布式查询方面的内容。包括事务、事务日志、锁与并发性、设置隔离层、快照隔离、处理死锁。另外还用大量篇幅介绍故障转移群集、分布式事务、分布式查询和分布式分区视图。

第 5 章快速介绍视图的开发过程，包括加密视图与索引视图。

第 6 章深入介绍约束的类型，并重点探讨键约束、CHECK 约束和默认约束同时讨论规则与默认值之间的关系。

第 7 章讨论脚本与批处理方面的内容，包括系统函数。通过本章的学习，你可以熟练编写出自己的批处理。

第 8 章讨论存储过程，本章通过简单的例子介绍了如何开发自己的存储过程，同时介绍存储过程中的流控制语句，特别重要的是，本章还用了大量篇幅讨论 OLE 存储过程的编写方法。

第 9 章讨论用户自定义函数的编写方法，特别注意 CLR 用户子函数的编写方法。

第 10 章涵盖触发器的内容，包括如何创建 DML、CLR 和 DDL 触发器。

第 11 章讨论 SQL 游标，本章介绍各种游标的使用与创建方法。

第 12 章重点介绍 SQL Server 2005 下的 XML 开发，包括 XML 基础、命名空间、DTD 和架构、转换、FOR XML 字句、XML 语法规则等。

第 13 章致力于 SQL Server 2005 高级查询，主要包括如何在 SQL Server 2005 中使用 XML、使用本级 XML Web 服务、SOAP 技术、公共语言运行时以及嵌套查询与关联查询等内容。

第 14 章介绍数据仓库，主要介绍数据仓库的组成、建模原则、概念模型、逻辑建模、数据仓库设计、实例构建过程与分析。

第 15 章是全书的示例汇总，介绍 SQL Server 2005 的经典例子——AdventureWorks 的设计、CLR 触发器、数据挖掘算法和 HTTP 访问等内容。

运行环境

本书讨论的所有内容几乎都提供有例子，而且列出所有代码，并在合适位置有图形显

示。为能够运行示例并从中得到更好效果，系统需要如下配备：

- SQL Server 2005。
- Windows NT 4.0 和 SP5 补丁，最好使用 Windows Server 2003（Win9x 也可以工作，但在某些方面会失败）。

常用习惯

本书采用多种文本显示方式，帮助区分信息。下面列举一些常用方式做解释。

注意：以楷体显示，并需要记住，而且与周围的文本有关。

↘ 背景信息、附加信息和参考内容以箭头项目显示。

- 关键词以黑体显示。
- 屏幕上显示的内容（例如菜单项）以相似字体显示在屏幕上。
- 所有对象名、函数名和其他细小内容均以代码风格显示。

书中仅操作步骤才使用（1）、（2）、（3）等编号，标题不采用此类样式。

代码或重要代码均以下列方式显示：

```
SELECT CustomerID,ContactName,Phone
FROM Customers
```

前面见过的或者无须再讨论的代码均不加背，例如：

```
SELECT ProductName FROM Products
```

在本书的编写、录入、排版以及修订过程中，得到宁义、翟新海、徐卸古、李瑞兴、彭东平、黄殿龙、李清杰、杨军、薛涛、杨前风、李卫东、马玉娟、阳亮、周雨、索向军等人的大力支持和帮助，在此一并表示感谢。

尽管笔者已经做了不懈努力，但书中难免仍有疏漏和不妥之处，诚望广大读者多提宝贵意见，批评指正。

目　录

1 SQL Server 2005 的开发模式

本章重点探讨 SQL Server 2005 数据库引擎下的开发模式，从而逐步揭开在 SQL Server 2005 上进行.NET 开发的神秘面纱。

为什么要先介绍数据库引擎呢？我们看一下图 1-1 中显示的内容。

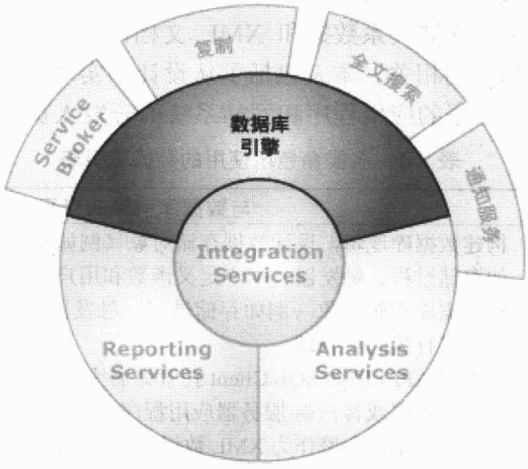

图 1-1　SQL Server 2005 服务组件位置分布

可以看出，数据库引擎在整个服务组件中处于非常重要的位置。数据库引擎下的开发模式与服务分割器、分析服务、报表服务等组件的开发模式具有许多共性，通过介绍数据库引擎的开发模式，可以得到管中窥豹的效果。

另外，数据库引擎是存储、处理和保证数据安全的核心服务。数据库引擎提供控制访问和进行快速地事务处理，满足企业中最需要占用数据的应用程序的要求。数据库引擎还为数据维护的高可用性提供了大量的支持。因此，我们将在本章介绍：

- 开发人员信息中心。
- 管理员信息中心。
- 结构设计师信息中心。
- 工作者信息中心。
- 设计与部署。
- 创建对象。

这些内容就是要介绍数据库开发人员、管理员、结构设计师和工作者在数据库开发中应具备哪些条件。

本章内容涉及的理论知识不是很深，但是这些内容具有较强的实用性和经验性，因此需要读者好好掌握，从而为自己充当数据库设计与使用阶段的不同角色打下良好的基础。

1.1 开发人员信息中心

在 SQL Server 2005 中，数据库引擎开发人员角色涵盖开发人员及与数据库引擎进行交互操作的全部人员。许多类型的开发人员（比如数据库设计人员或网站开发人员）都使用数据库引擎，他们都属于开发人员的范畴。在不同的单位，数据库开发人员和数据库管理员之间分配任务的方式也会有所不同。有些单位将某种类型的任务（例如数据库设计）分配给管理员，而其他单位则把相同的任务分配给开发人员。那么，开发人员和管理员这两个角色的职责到底该如何划分呢？从实际情况来说，开发人员角色与管理员角色很难区分，有时候这两个角色是相互交织的。

表 1-1 列出了一些较常见的开发人员类型以及他们与数据库引擎进行交互的方式。由于数据库引擎实例用于集中存储关系数据和 XML 文档，因此数据库引擎文档与任何需要使用该数据的开发人员都密切相关。本表包括了从设计和实现数据库的开发人员，到构建用户能够使用数据库引擎数据的应用程序和网站系统的开发人员的所有开发人员类型。

表 1-1 各类角色所使用的数据库引擎

开发人员类型	与数据库引擎的交互方式
数据库设计人员	构建数据库逻辑，设计数据存储对象（例如表和视图），编写逻辑对象（例如存储过程、触发器、用户定义函数和用户定义类型）的规范
数据库开发人员	对数据库逻辑对象（例如存储过程、触发器、用户定义函数和用户定义类型）进行编码和测试
数据访问开发人员	对使用 API（例如 SQL Client 托管命名空间或 OLE DB）访问关系数据的多层应用程序或客户端/服务器应用程序进行编码和测试
XML 开发人员	对使用数据库引擎作为 XML 数据存储库，并通过诸如 HTTP 端点和 XQuery 语言等功能，实现其数据访问的网站或数据驱动应用程序进行编码和测试
管理应用程序开发人员	对通过使用管理 API（例如 SMO 或 WMI 提供程序）或通过执行 T-SQL 语句实现数据库管理功能的应用程序进行编码和测试

数据库引擎管理员角色负责计划和管理数据库引擎实例的日常运行情况。具体而言，包含系统可用性、性能监视和优化、部署、升级、故障排除和配置等各个方面。中小型单位可能只有一个数据库管理员职位，该管理员承担所有管理员任务。大型单位可能会设置多个职位来分担管理员任务。表 1-2 列出了一些较常见的管理员类型以及他们与数据库引擎进行交互的方式。

表 1-2 管理员角色所使用的数据引擎

管理员类型	与数据库引擎的交互方式
数据库管理员	为一个或多个数据库引擎实例设计操作过程，并解决已发布的过程中未涵盖的异常情况
数据中心操作员	实施管理员定义的操作过程，监视系统运行状况，诊断并升级已发布的过程无法解决的异常情况
技术支持人员	向普通用户解释系统过程，或帮助用户解决系统出现的问题

1.1.1　应用程序设计概论

应用程序设计内容很多，主要包括：使用 T-SQL 元素、过程 T-SQL，用 SQL Native Client 建立应用程序，用集成的 CLR（Common Language Runtime）建立数据库对象，更改数据库中的数据，操作结果集，处理应用程序中的错误和消息，优化物理数据库设计及大容量导入、导出等内容。

1. T-SQL 元素

有关 T-SQL 语句的话题，我们已经在《SQL Server 2005 初学者指南》中进行了详细介绍，这里只就必要的进行一些补充，需要再介绍 T-SQL 中大多数语句都使用或受之影响的元素。

T-SQL 元素通常包括：标识符、数据类型、函数、表达式、运算符、注视和保留关键字等。

■ 标识符

标识符指表、视图、列、数据库和服务器等对象的名称。SQL Server 2005 中的所有内容都可以有标识符，例如服务器、数据库和数据库对象（比如表、视图、列、索引、触发器、过程、约束及规则等）。大多数对象要求有标识符，但对有些对象（例如约束）的标识符是可选的。

对象标识符是在定义对象时创建的，随后用于引用该对象。例如，下列语句创建一个标识符为 TableX 的表，该表中有两列的标识符分别是 KeyCol 和 Description：

```
CREATE TABLE TableX
(KeyCol INT PRIMARY KEY,
Descriptionnvarchar(80))
```

此表还有一个未命名的约束，PRIMARY KEY 约束没有标识符。

标识符的排序规则取决于定义标识符时所在的级别。为实例级对象（如登录名和数据库名）的标识符指定的排序规则是实例的默认排序规则。为数据库对象（例如表、视图和列名）的标识符指定的排序规则是数据库的默认排序规则，此排序规则在安装数据库时进行设置，如图 1-2 所示。例如，对于名称差别仅在于大小写的两张表，可以在使用区分大小写排序规则的数据库中创建，而不能在使用不区分大小写排序规则的数据库中创建。

标识符有两类：常规标识符和分隔标识符。在 T-SQL 语句中使用常规标识符时不用将其分隔开，不符合常规标识符格式规则的标识符通常用分隔标识符（即双引号或者方括号）分隔。符合标识符格式规则的标识符可以分隔，也可以不分隔。在 T-SQL 语句中，必须对不符合所有标识符规则的标识符进行分隔。例如：

```
SELECT  *
FROM  [MyTable]
WHERE [order]=10
```

常规标识符和分隔标识符包含的字符数必须在 1～128 之间。对于本地临时表，标识符最多可

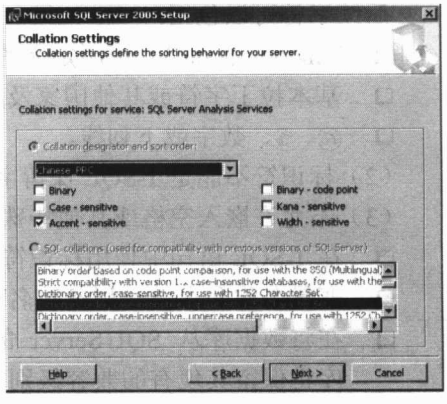

图 1-2　校验和排序设置

以有 116 个字符。常规标识符格式规则取决于数据库兼容级别。可以使用 sp_dbcmptlevel 设置该级别。语法如下：

```
sp_dbcmptlevel [ [ @dbname = ] name ]
     [ , [ @new_cmptlevel = ] version ]
```

参数说明如下：

- [@dbname =] name 要为其更改兼容级别的数据库的名称。数据库名称必须符合标识符的规则，name 的数据类型为 sysname，默认值为 NULL。
- [@new_cmptlevel =] version 数据库要与之兼容的 SQL Server 的版本。version 的数据类型为 tinyint，默认值为 NULL，该值必须为下列值之一：

60 = SQL Server 6.0

65 = SQL Server 6.5

70 = SQL Server 7.0

80 = SQL Server 2000

90 = SQL Server 2005

例如，以下示例将 AdventureWorks 数据库的兼容级别更改为 80：

```
EXEC sp_dbcmptlevel 'AdventureWorks', '80';
GO
```

当兼容级别为 90 时，下列规则适用：

（1）第一个字符必须是下列字符之一。

- Unicode 标准 3.2 所定义的字母。Unicode 中定义的字母包括拉丁字符 a-z 和 A-Z，以及来自其他语言的字母字符。
- 下划线、@、#或者数字符号。

  在 SQL Server 中，某些位于标识符开头位置的符号具有特殊意义。以@符号开头的标识符表示局部变量或参数，以一个数字符号开头的标识符表示临时表或过程。以两个数字符号开头的标识符表示全局临时对象。

  某些 T-SQL 函数的名称以两个@@符号开头。为了避免与这些函数混淆，不应使用以@@开头的名称。

后续字符可以包括：

- Unicode 标准 3.2 中所定义的字母。
- 基本拉丁字符或其他国家及地区字符中的十进制数字。
- @、$、数字或下划线。

（2）标识符不能是 T-SQL 保留字。SQL Server 保留其保留字的大写和小写形式。

（3）不允许嵌入空格或其他特殊字符。

- 在 T-SQL 语句中使用标识符时，必须用双引号或括号分隔不符合规则的标识符。
- 变量名称和存储过程参数名称必须符合常规标识符的规则。
- 在将数据库从 SQL Server 的任何早期版本升级到 SQL Server 2005 之后，该数据库将保留其现有的兼容级别。

- 对于 SQL Server 2005 的所有安装，默认兼容级别均为 90。不能修改 master 数据库的兼容级别，但可以修改 model 数据库的兼容级别。这样你便可以用非默认兼容级别创建新数据库，包含索引视图的数据库的兼容级别不能更改为低于 80。
- sp_dbcmptlevel 存储过程只影响指定数据库的行为，而不影响整个服务器的行为。sp_dbcmptlevel 只提供与 SQL Server 的早期版本的部分兼容性。将 sp_dbcmptlevel 用作中间迁移助手，可解决相关兼容级别设置所控制的行为之间存在的版本差异问题。如果现有的 SQL Server 应用程序受到 SQL Server 2005 中的行为差异的影响，请将该应用程序进行转换，使之能正常运行，然后使用 sp_dbcmptlevel 将兼容级别更改为 90。数据库的新兼容性设置将在该数据库下次成为当前数据库（无论是在登录时作为默认数据库还是在 USE 语句中指定）时生效。

每个 sp_dbcmptlevel 调用都必须单独提交，sp_dbcmptlevel 不能从其他上下文中调用，例如：

- 存储过程。
- 用 EXEC(string) 语法执行的 Transact-SQL 字符串。
- Transact-SQL 语句批。

■ 数据类型

数据类型用于定义数据对象（比如列、变量和参数）所包含的数据的类型。

包含数据的对象都有一个相关联的数据类型，它定义对象所能包含的数据种类，例如字符、整数或二进制。具有数据类型的对象有：表和视图中的列，存储过程中的参数、变量，返回一个或多个特定数据类型数据值的 T-SQL 函数，具有返回代码（始终为 integer 数据类型）的存储过程。

为对象分配数据类型时，可以为对象定义 4 个属性：对象包含的数据种类、所存储值的长度或大小，数值的精度（仅适用于数字数据类型）和数值的小数位数（仅适用于数字数据类型）。T-SQL 具有的系统数据类型如表 1-3 所示。

表 1-3　T-SQL 支持的数据类型

bigint	binary	bit	char	cursor
datetime	decimal	float	image	int
money	nchar	ntext	numeric	nvarchar
real	smalldatetime	smallint	smallmoney	sql_variant
table	text	timestamp	tinyint	varbinary
varchar	uniqueidentifier	xml		

存储在 SQL Server 2005 中的所有数据必须与上述这些基本数据类型之一相兼容。cursor 数据类型是唯一不能分配给表列的系统数据类型，只能用于变量和存储过程参数。一些基本数据类型具有同义词（例如，rowversion 是 timestamp 的同义词，nationalcharactervarying 是 nvarchar 的同义词）。

你还可以创建两种用户定义数据类型：

- 可以从基本数据类型创建别名数据类型，它们提供了一种可以将更能清楚地说明

对象中值的类型的名称应用于数据类型的机制，使程序员或数据库管理员能够更容易地理解用该数据类型定义的对象的用途。

```
--创建生日的类型可以为空
CREATE TYPE birthday
FROM  datetime NULL
GO
--创建一张使用新的用户定义数据类型的表
CREATE TABLE
Employee
(emp_id char(5),
emp_first_name char(30),
emp_last_name char(40),
emp_birthday birthday)
```

- CLR 用户定义数据类型基于托管代码中创建的数据类型，并以 SQL Server 程序集的形式上载。有关这一内容，将在 15.3 节中进行介绍。

■ 函数

这里的函数指 SQL Server 2005 提供的可用于执行特定操作的内置函数，这里函数可用以下情况：

（1）在使用 SELECT 语句的查询选择列表中，以返回一个值。

```
SELECT  DB_NAME();
GO
```

（2）在 SELECT 或数据修改（SELECT 、INSERT、DELETE 或 UPDATE）语句的 WHERE 子句搜索条件中，以限制符合查询条件的行。

```
USE AdventureWorks;
GO
SELECT  SalesOrderID,ProductID,OrderQty
FROM  Sales.SalesOrderDetail
WHERE Quantity=
(SELECT  MAX(Quantity) FROM  [OrderDetails]);
GO
```

（3）在视图的搜索条件（WHERE 子句）中，以使视图在运行时与用户或环境动态地保持一致。

```
CREATE VIEW ShowMyEmploymentInfo AS
SELECT  FirstName,LastName
FROM  Person.Contact
WHERE ContactID=SUSER_SID();
GO
```

（4）在任意表达式中。

（5）CHECK 约束或触发器中，以在插入数据时查找指定的值。