SQL Server 2008:

T-SQL 初学者指南

SQL Server 2008：T-SQL 初学者指南

本技术专题节选自《微软 SQL Server 2008：初学者指南》学习 SQL Server2008 T-SQL 支持的基本元素和基本操作，包括 SQL Server2008 中的 T-SQL 基本对象、在 SQL Server 中使用的 T-SQL 数据类型、SQL Server 2008 T-SQL 函数类型、SQL Server 2008 T-SQL 操作这些方面在这一技术专题中都作了详细的介绍。

SQL Server 2008 T-SQL 基本对象简介

本文节选自《微软 SQL Server 2008：初学者指南》学习 SQL Server2008 T-SQL 支持的基本元素和基本操作。你将了解 T-SQL 的基本对象和操作，包括常量、标识符和分隔符。你还将了解到每个对象与操作的相应的数据类型。作者 Dusan Petkovic 介绍了 SQL Server2008 中的 NULL 值、标量对象和全程标量。

- SQL Server 2008 中的 T-SQL 基本对象

SQL Server 2008 T-SQL 数据类型

所有的列中的数值都必须属于同样的数据类型（除非指定了 SQL_VARIANT 数据类型值）。T-SQL 使用不同的数据类型，具体分类如下：数值型数据类型、字符型数据类型、临时（如日期或时间）数据类型、混合型数据类型、附带有 VARDECIMAL 的 DECIMAL 存储类型。

- 在 SQL Server 2008 中使用 T-SQL 数据类型（一）
- 在 SQL Server 2008 中使用 T-SQL 数据类型（二）
- 在 SQL Server 2008 中使用 T-SQL 数据类型（三）

SQL Server 2008 T-SQL 函数类型

T-SQL 函数可以是聚合函数或者纯量函数。以下部分是对聚合函数、纯量函数、数字函数等等这些函数类型进行的描述。聚合函数适用于列中的一组数据值。聚合函数返回单一值。T-SQL 支持几组聚合函数：便捷型聚合函数、统计型聚合函数、自定义聚合函数以及分析型聚合函数。

- SQL Server 2008 中的 T-SQL 函数类型（一）
- SQL Server 2008 中的 T-SQL 函数类型（二）
- SQL Server 2008 中的 T-SQL 函数类型（三）

SQL Server 2008 T-SQL 操作

梯级算子(scalar operator)用于实现梯度值。T-SQL 支持数字、布尔算子（Boolean operator）以及串联。有一些是一元和二元运算算子。一元算子是用+和-作为符号的。二元运算算子是+， -， *， /和%。前面四个二元算子有它们各自的数学定义，而%就是取模算子。布尔算子有两个不同的符号，这取决于它们是否应用于位串或者其他数据类型。NOT、AND 和 OR 这三个算子适用于所有的数据类型（除 BIT）。

- SQL Server 2008 中的 T-SQL 操作（一）
- SQL Server 2008 中的 T-SQL 操作（二）

SQL Server 2008 中的 T-SQL 基本对象

本文节选自《微软 SQL Server 2008：初学者指南》学习 SQL Server2008 T-SQL 支持的基本元素和基本操作。你将了解 T-SQL 的基本对象和操作，包括常量、标识符和分隔符。你还将了解到每个对象与操作的相应的数据类型。作者 Dusan Petkovic 介绍 SQL Server2008 中的 NULL 值、标量对象和全程标量。

SQL 的基本对象：

数据库语言引擎 T-SQL 和其他程序设计语言有着相同的特征：

字面值（也叫常量）

标识符

分隔符

保留关键字

以下部分主要介绍这些内容。

字面值

字面值是一个包括文字与数字，十六进制或数字常量。一个字符常量包含单引号（’’）或双引号（″ ″）字符集中的一个或多个字符（如果多次使用了双引号就更趋向于使用单引号）。

如果你想在单引号分割的字符串中用到单独的引号，你就可以在这一字符串中用连续的单引号。十六进制的常量表示不可打印的字符或者是其他二进制数据。每个十六进制的

常量都以 0x 开头，后面附带有字符或者数字。例 4.1 和 4.2 就说明了有效和无效字符串常量和十六进制常量。

例 4.1

有效字符常量和十六进制的常量如下：

```
   'Philadelphia'
"Berkeley,  CA 94710"
'9876'
'省略符号如下所示：can''t'（注意两个连续的单引号）
0x53514C0D
例 4.2
```

下面的不属于字符常量：

```
   'AB'C'（奇数个单引号）
'New York"（同样类型的引号——单或双——都必须在每个字符串末尾使用）
```

数字常量包括所有有记号或没记号的整数、定点值和符点值（如例 4.3）。

例 4.3

以下属于数字常量

```


130
-130.00
-0.357E5（科学计数法—nEm 表示 n 乘 10^m）
22.3E-3
```

一个常量通常有一种数据类型和长度，这二者都取决于常量格式。此外，每个数据常量都有一个精确度和比例因子。（不同字面值的数据类型在本章中会提到。）

分隔符

在 T-SQL 中，双引号有两层意思。除了应用字符串之外，双引号还能够用来做分隔符，也就是所谓的定界表示符（delimited identifier）。分割标识符是标识的一种特殊类型，通常将保留关键字当作标识符并且用数据库对象的名称命名空间。

注：单引号和双引号之间的区别就在于前者是用于 SQL92 标准。标识符这种情况中，这种标准用于区分常规和分割标识符。关键的两点就是分割标识符是用双引号引出的而且还区分大小写（T-SQL 还支持用方括号代替双引号）。双引号只用于分割字符串。一般来说，分割标识符说明了标识符的规格，对保留的关键字也起到了同样的作用。分割标识符还可以让你不用名字（标识符、变量名），这些名字也可以在将来的 SQL 标准中用作保留的关键字。并且，分割标识符还可能在标识符名中包含不合规定的字符，如空格。

在 T-SQL 中，使用双引号是来定义 SET 语句的 QUOTED_IDENTIFIER 选项的。如果这一选项设为 ON（即默认值），那么双引号中的标识符就被定义成了分割标识符。在这种情况下，双引号就不能用于分割字符串。

注：说明一个 T-SQL 语句的注释有两种方法。使用一对字符/* and */ ，注释就是对附着在里面的内容进行说名。这种情况中，注释内容可能扩展成很多行。另外，这些字符--（两个连字符）表示当前行剩下的就是注释。（两个连字符就一符合 ANSI SQL 标准，而/*和*/是 T-SQL 的扩展名。

标识符

T-SQL 中，标识符用于识别数据库对象如数据库、表和索引。他们通过字符串表示出来，这些字符串的长度可以达到 128 个字符，还包含字母、数字或者下面的字符：_、@，、#和$。每个名称都必须以一个字母或者以下字符中的一个字符开头：_、@或#。#在

以它开头的表明或存储程序名表示一个临时对象，而以@开头的时间则表示一个变量。就像是之前提到的，这些规则并不适用于分割标识符（也叫作引用标识符），分割标识符可以将这些字符包含在内或者其中的任意字符开头（而不是分割副自己）。

保留关键字

每种程序设计语言都有一系列有保留意义的名称，他们被写入并用在定义格式中。这些名称叫做保留关键字。T-SQL 使用了大量这种名称，很多其他程序设计语言都不能用它们做对象名，除非这些对象是被详细划分或者是引用标识符。

注：在 T-SQL 中，所有数据类型的名称和系统功能，如 CHARACTER 和 INTEGER 都不是保留的关键字。它们是用来表示对象。（不要使用数据类型和系统功能作对象名称！这样会让 T-SQL 语句很难理解）

（作者：SearchSQLServer.com 译者：April 来源：TT 中国）

在 SQL Server 2008 中使用 T-SQL 数据类型（一）

所有的列中的数值都必须属于同样的数据类型（除非指定了 SQL_VARIANT 数据类型值）。T-SQL 使用不同的数据类型，具体分类如下：

- 数值型数据类型
- 字符型数据类型
- 临时（如日期或时间）数据类型
- 混合型数据类型
- 附带有 VARDECIMAL 的 DECIMAL 存储类型

下面详细说明所有的数据类型。

数值数据类型

数值数据类型用于代表数据。下表中列出了所有的数值型数据类型：

数据类型	详细说明
INTEGER	表示能够用4个字节保存的整数值，范围包括 -2,147,483,648至2,147,483,647。INT是INTEGER的缩写形式。
SMALLINT	表示能够用2个字节保存的整数值。这些值的范围从 - 32768至32767。
TINYINT	表示能够用1个字节保存的、零以上的正数值。范围为0至255。
BIGINT	表示能够用8个字节保存的整数值。范围为 -2^63至2^63 - 1。
DECIMAL(p,[s])	描述定点值。p(精确度)指定数字个数和假定的小数点s（刻度）数字。DECIMAL值可以用5-17个字节保存，由p值决定字节个数。DEC为DECIMAL的缩写。
NUMERIC(p,[s])	和DECIMAL相同。
REAL	适用于浮点值。包括正数值和负数值。正数值范围大约为2.23E - 308至1.79E +308，负数值为 - 1.18E - 38至 - 1.18E + 38（0也可以包含在内）。
FLOAT[(p)]	表示浮点值，如REAL。p定义精确度，即p<25表示单精度（4个字节）、p>=25为双倍精密度（8个字节）。
MONEY	表示币值。MONEY值和8字节的DECIMAL值一样，都是小数点后4个数字。
SMALLMONEY	和MONEY数据类型一样，只是用4个字节保存。

字符型数据类型

字符型数据类型有两种普通形式。它们可以是单字节的字符串，还可以使 Unicode 字符串（Unicode 用一些字节指定一个字符）。此外，字符串有固定或可变长度。以下为字符型数据类型：

数据类型	解释
CHAR[(n)]	表示单字节、有固定长度的字符串，其中为字符串内的字符个数。n的最大值为8000。CHAR(n)为CHARACTER(n)的另外一种书写形式。如果省略了n，那么字符串的长度就假定为1。
VARCHAR[(	表示单字节的字符可变长度的字符串（0 < n ≤ 8000）。和CHAR数据类型相反，VARCHAR数据值能够在它的实际长度中保存。这种数据类型有两个同义词：CHAR VARYING和CHARACTER VARYING。
NCHAR[(n)	保存固定长度的Unicode字符。CHAR和NCHAR数据类型之间最主要的区别为NCHAR数据类型的每个字符都可以用2个字节保存，而CHAR数据类型的每个字符串只用1个字节的存储空间。因此，NCHAR列中字符个数最多为4000。
NVARCHAR[	保存具有可变长度的Unicode字符串。VARCHAR和NVARCHAR数据类型之间的最主要的区别为每个NVARCHAR字符都能用2个字节保存，而每个VARCHAR字符只用1个字节的存储空间。因此NVARCHAR列中的字符个数最多为4000。

注：VARCHAR 数据类型和 CHAR 数据类型基本一样，只有一点不同：如果一个 CHAR(n)字符串比 n 个字符要短，那么其余的字符串就用空格来填充。VARCHAR 数据类型经常用实际长度来存储。

（作者：SearchSQLServer.com 译者：April 来源：TT 中国）

在 SQL Server 2008 中使用 T-SQL 数据类型（二）

临时数据类型

- T-SQL 支持下面的临时数据类型：
- DATETIME
- SMALLDATETIME
- DATE
- TIME
- DATETIME2
- DATETIMEOFFSET

DATETIME 和 SMALLDATETIME 数据类型指定了日期和时间，每个值都为整数并分别用 4 个字节或 2 个字节保存。DATETIME 和 SMALLDATETIME 的值是作为两个单独的数值保存的。DATETIME 日期值的范围为 01/01/1900 至 12/31/9999。SMALLDATETIME 相应的值为 01/01/1900 至 06/06/2079。时间部分能够再用 4 个字节(或者用两个字节保存 SMALLDATETIME)保存，它表示午夜过后的一秒的三百分之一（DATETIME）或分钟（SMALLDATETIME）。

如果你只想保存日期或时间部分，那么使用 DATETIME 和 SMALLDATETIME 非常不方便。出于这种原因，SQL Server 2008 介绍了新的数据类型：DATE 和 TIME，这两种数据类型只能分别保存 DATETIME 数据的 DATE 或 TIME 部分。DATE 数据类型用 3 个字节的保存并且范围为 01/01/0001 至 12/31/9999。TIME 数据类型用 3－5 个字节保存，且精确到 100ns。

DATETIME2 数据类型也是一种新的数据类型，用于保存精确度很高的日期和时间数据。这种数据类型能够根据要求定义可变长度。 存储大小为 6-8 个字节。时间部分的精确度为 100ns。这种数据类型不包括 Daylight Saving Time。

到现在为止我们介绍的临时数据类型都不支持时域。DATETIMEOFFSET 这种新的数据类型就有时区偏移量。所以，它用 6-8 个字节来保存。这种数据类型的其他部分都和 DATETIME2 相应量类似。

T-SQL 日期值默认为单引号或双引号里的字符串，格式为'mmm dd yyyy'（e.g.，'Jan 10 1993'）。（注意月份、日其和年之间的顺序是能够由 SET DATEFORMAT 语句控制的。此外，系统还能识别带有分隔符/或 - 的月份数值）。同样，这一时间值的格式为：'hh:mm'，Database Engine 使用 24 小时制（如 23:24）。

注：T-SQL 支持很多 DATETIME 值得输入格式。就如同你已经知道的，两个对象都能单独分开。因此，日期和时间值能够指定为任意顺序或者单独运用。如果省略其中的一个值，系统就能用默认值（时间默认值为 12:00 AM）。

例 4.4 和 4.5 表示不同方式，使用不同格式书写日期和时间值：

例 4.4

能够使用户以下日期描述：

```
   '28/5/1959' (with SET DATEFORMAT dmy)
'May 28,  1959'

   '1959 MAY 28'
```

例 4.5

以下描述能够使用：

```
    '8:45 AM'
'4 pm'
```

混合型数据类型

T-SQL 还支持一些其他的数据类型，这些数据类型不属于先前描述的任何一种数据类型。

- Binary 数据类型
- BIT
- 大对象数据类型
- CURSOR（在第 8 章中讨论）
- UNIQUEIDENTIFIER
- SQL_VARIANT
- TABLE（在第 8 章中讨论）
- XML（在第 28 章中讨论）
- Spatial（如 GEOGRAPHY 和 GEOMETRY）数据类型（在第 29 章中讨论）
- HIERARCHYID
- TIMESTAMP 数据类型
- 自定义数据类型（在第五章中讨论）

以下部分主要描述每种数据类型（除了在其他章节中之指定的之外）。

Binary 和 BIT 数据类型

BINARY 和 VARBINARY 数据类型是两种二进制的数据类型。它们表示系统内部格式的数据对象。它们用来存储位串，因此这些值都是十六进制的数字。

BIT 数据类型

Binary 和 BIT 数据类型都用一位存储，因此，一个字节能够存储高达 8 位的列。以下是对这些数据类型的属性进行的总结：

数据类型	解释
BINARY[(n)]	指定有固定长度和n个字节的位串（0 < n ≤ 8000）。
VARBINARY[(n)]	指定一个位串的可变长度，最高值为n个字节（0 < n ≤ 8000）。
BIT	用于指定Boolean数据类型，它包含三个值：FALSE、TRUE以及NULL。

（作者：SearchSQLServer.com 译者：April 来源：TT 中国）

在 SQL Server 2008 中使用 T-SQL 数据类型（三）

大对象数据库类型

大对象（LOBs）就是最大长度为 2GB 的数据对象。这些数据对象一般用来存储大型的文本数据、加载模块和音频/视频文件。T-SQL 支持两种不同的指定和访问方式：

1、使用 VARCHAR(MAX)， NVARCHAR(MAX)和 VARBINARY(MAX)的数据类型。

2、使用所谓的文本和图像数据类型。

MAX Specifier 是从 SQL Server 2005 开始的，你能用相同的程序模式访问标准数据类型和 LOBs 的值，换句话说，你能用简单的系统函数和字符串进行对 LOBs 操作。

Database Engine 用的是 MAX 分类符，包括以下数据类型：VARCHAR、NVARCHAR 和 VARBINARY 定义可变长度列。当你使用 MAX 默认值（不是确切值时），系统分析这一特殊字符的长度并决定是否将该值当作 LOB 来存储。MAX 分类符表示列值得大小能够达到目前系统的 LOB 最大值（在将来 SQL Server 版本中，MAX 将可能有一个更大值。

尽管数据库系同决定如何存储 LOB，你也可以通过使用 sp_tableoption 系统程序 LARGE_ VALUE_TYPES_OUT_OF_ROW 不考虑默认细则。如果选项值设为 1，那么列中的数据显然就是用的 MAX 分类符，就会与其他的数据区分开进行单独保存。如果选项设定为 0，Database Engine stores 就保存了所有表示行大小的值，一般行数据<8060 字节。

在 SQL Server 2008 中，你可以对 VARBINARY(MAX)列申请新的 FILESTREAM 属性将大型 binary 数据直接存储在 NTFS 文件系统中。这一属性的主要优点就是相应的 LOB 大小仅仅受限于文件系统的列大小。

TEXT、NTEXT 和 IMAGE 数据类型

TEXT、NTEXT 和 IMAGE 数据类型组成了所谓的文本/图像数据类型。IMAGE 类型的数据对象包含所有的数据（下载模块、音频/视频），而 TEXT 和 NTEXT 数据类型的数据对象包含在文本数据（也就是可打印的数据）。

文本/图像数据类型和数据库的其他值区别保存默认值，存储方式为使用指向该数据字段的 B-树型结构（A B-树型结构就是如同树的数据结构，所有远离树根的底端节点数字相同）。而对于含有文本/图像数据类型的列来说，Database Engine 在数据行中存储了 16 字节的指示器，指定能找到数据的地方。

如果文本/图像数据小于 32KB，指示器就指向了 B-树型结构中的根部节点，该节点长度约为 84 个字节。根部节点指向数据的物理模块。如果数据总量大于 32KB，Database Engine 就会在数据模块和根部节点建立中间节点。

由于每个表都包含一个以上有这种数据的列，列中的所有值都存储在一起。然而，一个物理页面能只包含单个表中的文本/图像数据。

虽然文本/图像数据是和所有其他数据分开保存的，你也可以用 sp_tableoption 系统程序的 TEXT_IN_ROW 选项进行修改。用这一选项你能指定最大字节数字，也可以让它和一般数据保存在一起。

文本/图像数据类型有很多局限性。你不能把它们当作本地变量（在存储程序或在 T-SQL 语句中）。同样，它们也不能作为索引的一部分、不能用在下列 SELECT 语句的子句中：WHERE、ORDER BY 和 GROUP BY。所有文本/图像数据类型最大的问题就是你必须使用特殊工具（READTEXT、WRITETEXT 和 UPDATETEXT）对它进行操作。

注：文本/图像数据类型具有 deprecated 特征，并且会迁移到将来的 Database Engine 版本中，用 VARCHAR(MAX)、NVARCHAR(MAX)和 VARBINARY(MAX)代替。

UNIQUEIDENTIFIER 数据类型

从它的名字来理解，UNIQUEIDENTIFIER 数据类型值是唯一辨别数字，它是用 16 个字节存储的二进制字符串。数据类型和 globally unique identifier（全球唯一识别码）密切相关，它保证了在世界范围内的唯一性。因此，通过运用数据类型，你能在分布式计算机系统中用非常独特的眼光识别数据和对象。

使用 NEWID 和 NEWSEQUENTIALID 函数可以进行列或 UNIQUEIDENTIFIER 类型的变量初始化，并且字符串常量是用十六进制数字和连字符这种特殊格式书写的。NEWID 和 NEWSEQUENTIALID 函数在本章以后的系统函数这部分会讲到。

在查询里用关键字 ROWGUIDCOL 能够查看 UNIQUEIDENTIFIER 数据类型列，并且指定包含 ID 值的列。这一关键字不能生成任何列。一个表中有多个 UNIQUEIDENTIFIER 列，但是只有一个列里包含 ROWGUIDCOL 关键字。

SQL_VARIANT 数据类型

SQL_VARIANT 数据类型用于同时存储各种数据类型，如数值、字符串以及日期值等等。只有 TIMESTAMP 类型值不能存储。SQL_VARIANT 列中的每个值都有两部分：日期值和描述值的信息。这些信息包括数据类型的所有属性，如长度、范围和精确度。

T-SQL 支持 SQL_VARIANT_PROPERTY 函数，它表示 SQL_VARIANT 列中每个值得附加信息。SQL_VARIANT 数据类型的用法，查看第五章例 5.5。

注：只有在非常必要的情况下才用 SQL_VARIANT 数据类型表中的列。如果这些值属于不同的数据类型或者在数据库设计过程中很难决定列中的数据类型，那么表中的列就应该包含这种数据类型。

HIERARCHYID 数据类型

HIERARCHYID 数据类型用于保存整个层面。它是作为 Common Language Runtime（CLR）自定义类型安装的，包括一些层级节点上创建和操作的系统函数。以下的一些函

数，属于这种数据类型：GetAncestor()， GetDescendant()， Read()， and Write()（有关这种数据类型的详细介绍不属于本章范围）。

TIMESTAMP 数据类型

TIMESTAMP 数据类型指定其中一栏为 VARBINARY(8)或 BINARY(8)，这是由列中的无效值决定的。系统包含每个数据库的当前值（不是日期或时间），这样在插入或更新含有 TIMESTAMP 列这样的行时就自动增加了。

注：该值不能自动保存在 TIMESTAMP 列中。因为 TIMESTAMP 列常用于查看是否一个具体的行已经在上次访问的时候发生了变化。

用 VARDECIMAL 格式存储的 DECIMAL 数据库

DECIMAL 数据类型一般以固定长度数据存储在磁盘上。自从 SQL Server 2005 SP2 以来，这种数据库就能用新的叫做 VARDECIMAL 存储格式以可变长度列存储。通过使用 VARDECIMAL，你能大大减少 DECIMAL 列的存储空间，在这些列里的值的长度也大不相同。

注：VARDECIMAL 是一种存储格式而不是一种数据类型。

当你需要指定最大可能值范围时 VARDECIMAL 存储格式就很有帮助，通常存储值要更小。 如表 4-1 表示：

精确度	字节个数: VARDECIMAL	字节个数: 固定长度
0 or NULL	2	5
1	4	5
20	12	13
30	16	17

38	20	17

注：VARDECIMAL 存储格式和 DECIMAL 数据类型工作原理一样，都是包括文字和数字的 VARCHAR 数据类型。

表 4-1 为存储 VARDECIMAL 和固定长度的字节个数。

要启用 VARDECIMAL 存储格式，你首先要为数据库启用这种格式，然后为数据库特殊的表启用这种格式。sp_db_ vardecimal_storage_format 系统程序用于第一步，如例 4.6 所示：

例 4.6

```
EXEC sp_db_vardecimal_storage_format 'sample', 'ON';
```

sp_table 选项系统程序中的 VARDECIMAL STORAGE FORMAT 选项常用于为该表开启这种存储格式。例 4.7 就为该项目表启用了 VARDECIMAL 存储格式。

例 4.7

```
EXEC sp_tableoption 'project', 'vardecimal storage format', 1
```

就像你已经了解到的一样，用 VARDECIMAL 存储格式最主要的原因就是减少数据的存储大小。如果你想测试它能够获得多少存储空间，你可以用 sys.sp_estimated_rowsize_reduction_for_vardecimal 动态管理视图。这种管理视图能对这种表进行详细概算。

(作者： SearchSQLServer.com 译者：April 来源：TT 中国)

SQL Server 2008 中的 T-SQL 函数类型（一）

T-SQL 函数

T-SQL 函数可以是聚合函数或者纯量函数。以下部分是对这些函数类型的描述：

聚合函数

聚合函数适用于列中的一组数据值。聚合函数返回单一值。T-SQL 支持几组聚合函数：

- 便捷型聚合函数
- 统计型聚合函数
- 自定义聚合函数
- 分析型聚合函数

统计型和分析型聚合函数在 24 章中已经讨论过了。自定义聚合函数不属于本章范围。以下是对便捷型聚合函数的描述：

AVG——计算列中数值得算数中项（平均值）。该列中包含的值必须是数字类型的。

MAX 和 MIN——分别计算列中的最大和最小数据值。该列中的值可以属于数字、字符串以及日期/时间型的。

SUM——计算列中的所有数据值之和。这些列中的值必须是数字。

COUNT——计算列中的（非零）数据值。COUNT(*)是唯一一种不适用于列的聚合函数。该函数是返回的是行数（不管列中的数字是否有 NULL 值）。

COUNT_BIG——和 COUNT 类似，唯一的区别就是 COUNT_BIG 返回的值属于 BIGINT 数据类型。

SELECT 语句便捷函数的使用在第六章中进行了详细介绍。

纯量函数

除了聚合函数之外，T-SQL 还提供了一些纯量函数，用于创建标量表达式。纯量函数主要针对单一值或列表值；和聚合函数相反，聚合函数主要是针对多行中的数据。纯量函数可以分成以下几类：

- 数字函数
- 日期函数
- 字符串函数
- 系统函数
- 元数据函数

以下是对这些函数类型的详细描述：

数字函数

T-SQL 中的数字函数就是修改数值的数字函数。可以使用以下列数字函数：

函数	说明
ABS(n)	返回数字表达式 n 的绝对值(也就是说负数值返回的结果就是正数)。例: SELECT ABS(-5.767) = 5.767， SELECT ABS(6.384) = 6.384

ACOS(n)	计算 n. n 的反余弦值，结果属于 FLOAT 数据类型。
ASIN(n)	计算 n. n 的正弦值，结果属于 FLOAT 的数据类型。
ATAN(n)	计算 n. n 的反正切值，结果属于 FLOAT 数据类型。
ATN2(n, m)	计算 n/m. n, m 的反正切值，结果属于 FLOAT 数据类型。
CEILING(n)	返回的整数值大于或等于具体参数。例: SELECT CEILING(4.88) = 5 SELECT CEILING(-4.88) = -4
COS(n)	计算 n. n 的余弦值，结果值属于 FLOAT 数据类型。
COT(n)	计算 n. n 的余切值，并且结果值属于 FLOAT 数据类型。
DEGREES(n)	将弧度转变成度数。例: SELECT DEGREES(PI()/2) = 90.0 SELECT DEGREES(0.75) = 42.97
EXP(n)	计算 e^n 的值。例: SELECT EXP(1) = 2.7183

FLOOR(n)	计算小于或等于给定值 n 的最大整数值。例: SELECT FLOOR(4.88) = 4
LOG(n)	计算 n 的自然对数（也就是说基数为 e）。例: SELECT LOG(4.67) = 1.54 SELECT LOG(0.12) = -2.12
LOG10(n)	计算 n 的对数（基数为 10）例:
PI()	返回圆周率值（3.14）。
POWER(x, y)	计算 x^y 值。例: SELECT POWER(3.12, 5) = 295.65 SELECT POWER(81, 0.5) = 9
RADIANS(n)	将度数转换成弧度。如: SELECT RADIANS(90.0) = 1.57 SELECT RADIANS(42.97) = 0.75
<? xml:namespace prefix = st1 ns = "urn:schemas-microsoft-com:office:smarttags" />RAND	返回 0-1 之间的任意值，结果属于 FLOAT 数据类型。

ROUND(n， p，[t])	对 n 进行四舍五入，精确度为 p。p 为正数时，就对小数点右边的数字进行四舍五入。如果是负数的话，就对小数点左边的数字进行四舍五入。可选参数 t 就删除了 n。例: SELECT ROUND(5.4567，3) = 5.4570 SELECT ROUND(345.4567，-1) = 350.0000 SELECT ROUND(345.4567，-1，1) = 340.0000
ROWCOUNT_BIG	返回系统执行的、受最后一行 T-SQL 语句影响的行数。该函数的返回值为 BIGINT 数据类型。
SIGN(n)	返回 n 值的符号数字 (+1 为正数，-1 f 为负数，0 就是 0).例:SELECT SIGN(0.88) = 1
SIN(n)	计算 n. n 的正弦，结果值属于 FLOAT 数据类型。
SQRT(n)	计算 n 的平方根。例: SELECT SQRT(9) = 3
SQUARE(n)	返回给定式的平方值。例: SELECT SQUARE(9) = 81

TAN(n)	计算 n. n 的正切，结果值属于 FLOAT 数据类型。

(作者：SearchSQLServer.com 译者： April 来源：TT 中国)

SQL Server 2008 中的 T-SQL 函数类型（二）

日期函数

日期函数计算表达式的日期和时间或者是从时间间隔中返回该值。T-SQL 语句支持以下函数：

函数	说明
GETDATE()	返回目前系统日期和时间。例: SELECT GETDATE() = 2008-01-01 13:03:31.390
DATEPART(item, date)	返回日期指定的 item，date 为一个整数。例: SELECT DATEPART(month, '01.01.2005') = 1 (1 = January) SELECT DATEPART(weekday, '01.01.2005') = 7 (7 = Sunday)
DATENAME(item, date)	返回日期指定的 item，date 为一个字符串。例: SELECT DATENAME(weekday, '01.01.2005') = Saturday
DATEDIFF(item, dat1, dat2)	计算两个日期部分 dat1 和 dat2 之间的区别，返回的结果为 item 表示单元的整数值。 例:

	SELECT DATEDIFF(year, BirthDate, GETDATE()) AS age FROM employee; ->返回每个员工的年龄
DATEADD(i, n, d)	将 i 值单元里的数字 n 增加到指定日期 d。例: SELECT DATEADD(DAY, 3, HireDate) AS age FROM employee; ->在每个员工聘用日期的基础上增加 3 天（详见 sample 数据库）

字符串函数

字符串函数用于处理列中的数据值，通常属于字符型数据类型。T-SQL 支持以下字符串函数：

函数	说明
ASCII(character)	将具体的字符转换为相应的整数（ASCII）代码。返回结果为正数。例： SELECT ASCII('A') = 65
CHAR(integer)	将 ASCII 代码转换为相应的字符。例： SELECT CHAR(65) = 'A'。
CHARINDEX(z1, z2)	返回部分字符串 z1 在字符串 z2 中首次出现的起始位置。如果 z1 没有在 z2 中出现，那么返回值就为 0。例: SELECT CHARINDEX('bl', 'table') =

	3。
DIFFERENCE(z1, z2)	返回值为 0-4 之间的整数，这就是 z1 和 z2 这两个字符串 SOUNDEX 之间的区别。SOUNDEX 返回的数字指定的是字符串的语音。这种方法能够判断有相同发音的字符串）例: SELECT DIFFERENCE('spelling', 'telling') = 2 (发音有些相近， 0 =发音部相同)。
LEFT(z, length)	返回字符串 z 中的第一个字符长度。
LEN(z)	返回指定的字符串表达式的字符个数而不是字节个数，包括后面的空格。
LOWER(z1)	将字符串 z1 中所有的大写字母转换成小写字母。小写字母和数字以及其它的字母保持不变。例: SELECT LOWER('BiG') = 'big'。
LTRIM(z)	去掉字符串 z 开头的空格。例: SELECT LTRIM(' String') = 'String'。
NCHAR(i)	返回由统一码标准定义的、有指定整数代码的统一码字符。
QUOTENAME(char_string)	返回有分隔符的统一码字符串，使输入字符串变成有效分隔符。
PATINDEX(%p%, expr)	返回指定表达式 expr 中模式 p 第一次出现的起始位置，如果没有找到这个模式，那么返回值就为零。例:

	1) SELECT PATINDEX('%gs%', 'longstring') = 4; 2) SELECT RIGHT(ContactName, LEN(ContactName)-PATINDEX('% %', ContactName)) AS First_name FROM Customers; (第二个查询从 customers 列中返回所有名。)
REPLACE(str1, str2, str3)	将所有 str1 中出现的 str2 替换为 str3。例: SELECT REPLACE('shave', 's', 'be') = behave
REPLICATE(z, i)	将字符串 z 重复 i 次。例: SELECT REPLICATE('a', 10) = 'aaaaaaaaaa'
REVERSE(z)	将字符串 z 显示为倒序。例: SELECT REVERSE('calculate') = 'etaluclac'
RIGHT(z, length)	在字符串 z 中返回最后字符的长度。例: SELECT RIGHT('Notebook', 4) = 'book'
RTRIM(z)	取消字符串 z 最后的空格。例: SELECT RTRIM('Notebook ') = 'Notebook'

SOUNDEX(a)	返回四个字符的 SOUNDEX 代码判断两个字符串中的相似性。例: SELECT SOUNDEX('spelling') = S145
SPACE(length)	返回一个字符串，length 为其指定的空间长度。例: SELECT SPACE = ' '
STR(f, [len [, d]])	将指定的 float 表达式 f 转换成字符串。len 是指字符串的长度，包括小数点、正负号、数字和空格（默认值为 10），d 为小数点右边的被返回的数字。例: SELECT STR(3.45678, 4, 2) = '3.46'
STUFF(z1, a, length, z2)	用字符串 z2 中的位于 a 处的部分字符串代替 z1 中的部分字符串，代替 z1 中的 length 字符。例: SELECT STUFF('Notebook', 5, 0, ' in a ') = 'Note in a book' SELECT STUFF('Notebook', 1, 4, 'Hand') = 'Handbook'
SUBSTRING(z, a, length)	在字符串 z 中的 a 处创建部分字符串，length 为新创建字符串的长度。例: SELECT SUBSTRING('wardrobe', 1, 4) = 'ward'
UNICODE	返回由统一码定义的整数值，该值为输入表达式的第一个字符。
UPPER(z)	将字符串 z 中的所有小写字母转换成大

	写字母。大写字母和数字不变。例: SELECT UPPER('loWer') = 'LOWER'

(作者：SearchSQLServer.com 译者：April 来源：TT 中国)

SQL Server 2008 中的 T-SQL 函数类型（三）

系统函数

T-SQL 系统函数提供了更多有关数据库对象的信息。大部分系统函数用的是内部数字标识符（ID），系统将标识付值给每个数据库对象。使用这类标识符，系统就能独立识别每个数据库对象。系统函数提供了一些有关数据库系统的信息。以下为一些描述系统函数的表（要查看完整的系统函数清单，请参考联机丛书）。

函数	说明
CAST(a AS type [(length)]	将表达式 a 转换成指定的数据类型 type （如果可能的话）。A 可以是任一有效表达式。例: SELECT CAST(3000000000 AS BIGINT) = 3000000000
COALESCE(a1，a2，…)	返回给定清单上的表达式 a1、a2……，并且第一个表达式的返回值不是 NULL 值。
COL_LENGTH(obj，col)	返回 col 列的长度，该长度值属于数据库对象（表或视图）obj 。 例: SELECT COL_LENGTH('customers'，'cust_ID') = 10

CONVERT(type[(length)], a)	和 CAST 相等，但是对这两个参数指定条件不同。CONVERT 能用于任意数据类型。
CURRENT_TIMESTAMP	返回目前的日期和时间。例：SELECT CURRENT_TIMESTAMP = '2008-01-01 17:22:55.670'
CURRENT_USER	返回目前用户的姓名。
DATALENGTH(z)	计算表达式 z 的结果长度（字节）例： SELECT DATALENGTH(ProductName) FROM products. （该查询返回每个域名的长度）
GETANSINULL('dbname')	如果按照 ANSI SQL 标准在数据库 dbname 中使用 NULL 值，那么返回值为 1（参考本章末对 NULL 值的详细说明）。例： SELECT GETANSINULL('AdventureWorks') = 1
ISNULL(expr, value)	如果 expr 不为零，就返回 expr 值；否则就返回 value （查看例 5.22）。
ISNUMERIC(expression)	判断表达式是否属于无效的数字型。
NEWID()	创建由 16 个字节组成的二进制字符串存储 UNIQUEIDENTIFIER 数据

	类型。
NEWSEQUENTIALID()	在指定的计算机上创建 GUID，它比该函数之前产生的 GUID 值要大。我们只可以将这个函数设置为默认值。
NULLIF(expr1, expr2)	如果表达式 expr1 和 expr2 相等，返回 NULL 值。例: SELECT NULLIF(project_no, 'p1') FROM projects（该查询返回带有 project_no = 'p1'的项目值为 NULL）。
SERVERPROPERTY(propertyname)	返回数据库服务器的属性信息。
SYSTEM_USER	返回目前用户的登陆 ID。例: SELECT SYSTEM_USER = LTB13942dusan
USER_ID([user_name])	返回用户 user_name 的标识符。如果没有指定名字，就检索当前用户的标识符。例: SELECT USER_ID('guest') = 2
USER_NAME([id])	返回带有标识符 id 的用户名字。如果没有指定用户名，就检索当前用户的名字。例: SELECT USER_NAME = 'guest'

所有字符串函数可以以任意顺序嵌入，例如：REVERSE(CURRENT_USER)。

元数据函数

一般来说，元数据函数返回的是有关指定数据库和数据对象的信息。下表介绍了几种元数据函数（查看完整元数据函数清单，请参考联机丛书）。

函数	说明
COL_NAME(tab_id, col_id)	返回列名，这些列都属于带有 ID tab_id 以及列 ID col_id 的表。例: SELECT COL_NAME(OBJECT_ID('employee'), 3) = 'emp_lname'
COLUMNPROPERTY(id, col, property)	返回指定列的信息。例: SELECT COLUMNPROPERTY(object_id('project'), 'project_no', 'PRECISION') = 4
DATABASEPROPERTY(database, property)	返回指定的数据库和属性指定数据库属性值。例: SELECT DATABASEPROPERTY('sample', 'IsNullConcat') = 0. (IsNullConcat 属性和 CONCAT_NULL_YIELDS_NULL 选项一致，在本章末尾进行了介绍。)
DB_ID([db_name])	返回数据库 db_name 的标识符，即返回当前数据库的标识符。例: SELECT DB_ID('AdventureWorks') =

	6
DB_NAME([db_id])	返回带有标识符 db_id 的数据库名。如果没有指定标识符，就显示当前数据库名称。例: SELECT DB_NAME(6) = 'AdventureWorks'
INDEX_COL(table， i， no)	返回表 table 中的索引列，该索引列由索引标识符 i 及该列在索引中的位置 no 指定。
INDEXPROPERTY(obj_id，index_name， property)	返回指定表标识号、索引或统计名称及属性名称的指定索引值或统计属性值。
OBJECT_NAME(obj_id)	返回有标识符 obj_id 的数据库对象名称例: SELECT OBJECT_NAME(453576654) = 'products'
OBJECT_ID(obj_name)	返回数据库对象 obj_name 的标识符。例: SELECT OBJECT_ID('products') = 453576654
OBJECTPROPERTY(obj_id，property)	返回当前数据库对象。

(作者：SearchSQLServer.com 译者：April 来源：TT 中国)

SQL Server 2008 中的 T-SQL 操作（一）

梯级算子(scalar operator)用于实现梯度值。T-SQL 支持数字、布尔算子（Boolean operator）以及串联。

有一些是一元和二元运算算子。一元算子是用+和-作为符号的。二元运算算子是+，－， *， /和%。前面四个二元算子有它们各自的数学定义，而%就是取模算子。

布尔算子有两个不同的符号，这取决于它们是否应用于位串或者其他数据类型。NOT、AND 和 OR 这三个算子适用于所有的数据类型（除 BIT）。第六章详细介绍了这种情况。处理位串的最好的算子在下表中已经列出，例 4.8 详细说明了它们如何运用：

- ~补足语（如 NOT）
- &位串的连接词（如 AND）
- |分离位串（如 OR）
- ^不可兼取的分离符号(如 XOR 或 Exclusive OR)

```
   例：~(1001001) = (0110110)
(11001001) | (10101101) = (11101101)
(11001001) & (10101101) = (10001001)
(11001001) ^ (10101101) = (01100100)
```

串联算子+用于串联两个字符串和位串。

全程变量

全程变量是特殊变量系统，如果它们是标量常数，就能够使用。T-SQL 支持许多全程变量，但是这些变量的必须加上前缀@@。下表就详细介绍了几种全程变量（了解全程变量的所有清单，请查看联机丛书）。

变量	说明
@@CONNECTIONS	在启动系统之前返回登陆次数。
@@CPU_BUSY	在启动系统之前返回 CPU 花费的总时间（在毫秒单元里）。
@@ERROR	返回上次 T-SQL 执行语句的返回值。
@@IDENTITY	返回上次插入到列中、具有 IDENTITY 属性的值。（详见第六章）。
@@LANGID	返回目前数据库系统使用的语言标识符。
@@LANGUAGE	返回数据库系统目前使用的语言名称。
@@MAX_CONNECTIONS	返回和系统实际连接的最大数字。
@@PROCID	返回当前执行的存储程序的标识符。
@@ROWCOUNT	返回系统执行的最后一个 T-SQL 语句的行数。

@@SERVERNAME	检索本地数据库服务器信息。该信息还包括服务器名和实例名。
@@SPID	返回服务器程序标识符。
@@VERSION	返回数据库系统软件的当前版本。

NULL 值

NULL 值是赋给列的特殊值。通常在不清楚列中的信息或者这些信息不能用的时候就用到这这个值。例如，一个未知的公司员工的家庭电话号码，我们就可以对 home_telephone 列赋 NULL 值。

如果该表达是的操作数本身就是 NULL 值的话，那么任何算数表达式都可能导致 NULL 值。因此，很多一元算数表达式（如果 A 就是带有 NULL 值的表达式），+A 和 - A 返回的值就是 NULL。二元表达式如果 A 或 B 操作数中的一个（或者是两个）是 NULL 值，A + B，、A - B、A * B、 A / B 和 A % B 结果也有可能是 NULL。（操作数 A 和 B 必须是数值表达式。）

T-SQL 系统函数提供了更多有关数据库对象的信息。大部分系统函数用的是内部数字标识符（ID），系统将标识付值给每个数据库对象。使用这类标识符，系统就能独立识别每个数据库对象。系统函数提供了一些有关数据库系统的信息。以下为一些描述系统函数的表（要查看完整的系统函数清单，请参考联机丛书）。

（作者：SearchSQLServer.com 译者：April 来源：TT 中国）

SQL Server 2008 中的 T-SQL 操作（二 ）

如果表达式包含相关操作并且其中一个（或两个）操作数含有 NULL 值，那么该操作结果就是 NULL。因此，以下表达式：A = B、A <> B、 A < B 和 A > B 就会返回 NULL。

AND、OR 和 NOT 中，下列真值表详细说明了 NULL 值的行为。T 表示真实，U 表示未知（NULL），F 表示错误。在这些表中，行和列表示布尔表达式算子操作值，它们之间的交叉值就是结果值。

AND	T	U	F	OR	T	U	F	NOT	
T	T	U	F	T	T	T	T	T	F
U	U	U	F	U	T	U	U	U	U
F	F	F	F	F	T	U	F	F	T

在聚合函数 AVG、SUM、MAX、MIN 以及 COUNT 计算出结果之前，就已经排除了它们自变量的 NULL 值（COUNT(*)函数除外）。如果一个列只包括 NULL 值，那么该函数就返回 NULL。和非 NULL 值已一样，聚合函数 COUNT(*)能处理所有 NULL 值。如果列中只包含 NULL 值，那么函数 COUNT（DISTINCT column_name）返回的结果就是 0。

一个 NULL 值必须是和另外一个不一样。例如在数值型数据类型中，0 和 NULL 之间就有很大的区别。同样在字符型数据类型中，空字符串和 NULL 值之间也有很大的区别。

如果列表定义中明确表示包含 0 ，它就会允许 NULL 值。换句话说，如果列的定义中包含 NOT NULL，就不允许 NULL 值。如果用户没有在列中指定一种数据类型的 NULL 或 NOT NULL（TIMESTAMP 除外），那就赋予了下列值：

NULL——如果 SET 语句 ANSI_NULL_DFLT_ON 选项设定值为 ON。

NOT NULL——如果 SET 语句 ANSI_NULL_DFLT_OFF 选项设定值为 ON。

如果 SET 语句没有被激活，列中就包含了 NOT NULL 的默认值。TIMESTAMP 数据类型这一列只能叫 NOT NULL 列。

SET 语句还有一个选项：CONCAT_NULL_YIELDS_ NULL。这一选项影响了 NULL 值场串联操作，所以你串联到 NULL 值的任何数据都会再次产生 NULL。例如：

```
'San Francisco' + NULL = NULL
```

总结

T-SQL 的基本特征就是数据类型、谓词和函数。数据类型都是参照 ANSI SQL92 标准的数据类型。T-SQL 支持一系列有用的系统函数。

下一章将向大家介绍与 T-SQL 语句相关的 SQL 的数据定义语言。T-SQL 这部分内容包含所有用于创建、改变、迁移数据对象的语句。

（作者：SearchSQLServer.com 译者：April 来源：TT 中国）