北京科海培训中心

• Informix 数据库培训教材之二

SQL 查询语言及应用

王 冠 编著

科学出版社

1999

内 容 简 介

本书是基于作者多年 Informix 数据库培训授课经验及实践应用的体验编写的，主要讲述 SQL 查询语言及其实现。

全书内容共分 9 章，分别讲述 SQL 基础、简单查询、多表连接查询、分组统计、子查询、集合运算、SQL 的数据更新功能及存储过程等内容，并通过大量应用实例，深入浅出地讲述了数据库查询的方法与技巧。

本书文字简洁、条理清晰，即适合于作为 Informix 数据库培训教材，也适合于企业的数据库管理人员作为参考手册。

图书在版编目(CIP)数据

SQL 查询语言及应用/王冠编著。—北京：科学出版社，
1999.6
Informix 数据库培训教材之二
ISBN 7-03-007694-X

Ⅰ. S… Ⅱ. 王… Ⅲ. SQL 语言-基本知识 Ⅳ. TP311.13

中国版本图书馆 CIP 数据核字(1999)第 25380 号

科学出版社 出版
北京东黄城根北街 16 号
邮政编码：100717

北京市朝阳区科普印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1999 年 7 月第 一 版 开本：787×1092 1/16
1999 年 9 月第 2 次印刷 印张：12 1/4
印数：5 001—10000 字数：298 000

定价：18.00 元

前　言

SQL 是“Structured Query Language”(结构化查询语言)的首字母缩写,是关系数据库管理系统的接口语言。SQL 最早出现于 IBM 公司的关系数据库管理系统 System/R 中,后被许多关系数据库管理系统采纳为数据操纵语言。由于 SQL 语言具有功能丰富,使用方式灵活,语言简洁易学等优点,先后被美国国家标准局(ANSI)和国际标准化组织(ISO)接纳为关系数据库语言的标准。现在,绝大多数的大型数据库管理系统产品都以 SQL 作为用户接口语言,而一些桌面数据库产品也纷纷提供了 SQL 语言的接口。因此,无论是数据库管理员,还是普通用户或者开发人员,SQL 语言都是一个必须很好掌握的工具。

在高校的计算机专业中,“数据库系统基础”是一门必修课,而 SQL 语言是这门课的一个重要组成部分。笔者为北京工业大学计算机系的一名教师,一直讲授“数据库系统基础”这门课,后来又兼任 Informix 公司联合培训中心的教员,从事 Informix 产品的培训工作。在讲授 SQL 语言的过程中,笔者发现让学生掌握 SQL 语言的语法非常容易,而如何利用 SQL 语句来表达自己的想法,正确操纵数据库中的数据却比较困难。市面上已有的一些相关资料多是介绍 SQL 语法的,对于如何使用 SQL 讲得很少,因此笔者在讲解如何使用 SQL 语句的想法。

完整的 SQL 语言涉及数据库管理与使用的各个方面的功能,包括数据定义、数据查询、数据更新、数据控制以及 SQL 编程等等。各数据库厂商在实现 SQL 的过程中,根据各自的需要又对标准作了一些修改,形成了 SQL 语言的各种方言。但是,SQL 语言的基本语句在各种实现中并没有太大差异,特别是数据定义、数据查询、数据更新等方面的语句。因此,本书在讲解 SQL 语言时,主要介绍 Informix 公司实现的 SQL 语言,同时兼顾 SQL 的标准。因为数据查询和数据更新两方面使用频率较高,又难以掌握,并且在各种实现中差异不大,所以本书内容集中在这两个方面。本书主要内容有:SQL 基础、样例数据库、简单查询、连接查询、分组统计、子查询、集合运算以及数据更新,最后介绍存储过程。

本书是笔者多年教学和培训经验的总结,在介绍 SQL 语法的同时,列举了大量的实例说明如何使用 SQL 语句实现各种功能。书中所举的实例用到了 Informix 数据库附加的一个样例数据库 Stores7。全部实例均在 Informix 数据库环境下调试通过,读者可通过一边学习一边实践的方法,快速掌握 SQL 语句的使用。

本书适用于所有想了解和学习 SQL 语言的学生、数据库用户、开发人员和数据库管理员,也可以作为“数据库系统基础”课程的教辅教材,或者 SQL 语言的培训教程。

由于本书没有包括 SQL 语言的全部内容,因而不适合作为 SQL 手册使用。作者正在考虑编写一本书全面介绍 Informix－SQL 语言各方面特性的手册,以弥补这一缺憾。如果读者有这方面的建议或问题,请与作者联系。

作者的 Email 地址为:wangguan@163.net。

目　录

第 1 章　SQL 基础

SQL 是关系数据库中使用得最广泛的工具，已成为关系数据库的标准数据操纵语言，它具有简单灵活、功能强大的特点。本章主要介绍 SQL 的一些基本知识，主要内容包括：

- 什么是 SQL
- SQL 的发展历史
- SQL 语句
- 数据类型

1.1　什么是 SQL

对 SQL 要了解的第一件事情是：它不像 Cobol，Fortran，C 和 Pascal 程序设计语言。在这些语言中要解决一个问题，都必须写一个程序，描述出解决问题的每一个步骤。SQL 是非过程化的，要用 SQL 解决一个问题，不必告诉系统如何去获得你需要的东西，只要简单地告诉系统你想要什么，那么数据库管理系统就会用一个比较好的方法找出你需要的东西。

例如，我们需要了解由美国新泽西州的客户签订的并且发货时间为 1998 年 7 月的订单有哪些，就可以用下列形式向系统提出请求：

```
select *
from customer, orders
where customer.customer_num=orders.customer_num
    and state = 'NJ'
    and ship_date between
        date('7/1/98') and date('7/31/98')
```

这就是 SQL 语句的一个例子，在这个语句中，我们只描述了要找的数据所具有的特征而并没有告诉系统如何去做，这是 SQL 语言的一个重要特征。

一般认为 SQL 是英文 Structured Query Language 的首字母缩写，即结构化查询语句。由于历史的原因，SQL 通常读作“sequel”，也可以分开读作“ess cue ell”。尽管 SQL 被称作结构化查询语言，但该语言的功能不仅仅限于数据查询，而是涵盖了数据库系统从设计到运行维护的各个方面。SQL 的主要功能包括：

- **数据定义(Data definition)**　定义数据存放的结构，以及数据项之间的关系；
- **数据检索(Data retrieval)**　使用户或应用程序可以从数据库中检索数据，并使用这些数据；
- **数据操纵(Data manipulation)**　增加、修改或删除数据库中的数据；
- **存取控制(Access control)**　限制用户检索、增加和删改数据的权限，以保护数据库中的数据不被非法存取；
- **数据共享(Data Sharing)**　保证用户对数据进行访问时互不干扰；

· **数据完整性(Data integrity)** 保证数据库中存放的数据的正确性和一致性。

SQL 语言可以有两种使用方法,第一种是交互方式,即用户输入一个 SQL 语句,系统马上执行这条语句,并将结果反馈给用户;第二种是将 SQL 语句嵌入到某一高级语言中,编写成一段程序批处理运行。在这两种使用方式中,SQL 语句的语法基本上相同。

1.2 SQL 的发展历史

SQL 的发展离不开关系数据库的发展。1970 年,IBM 公司的研究人员 E. F. Codd 发表了题为"A Relational Model of Data for Large Shared Data Banks"(大型共享数据库的关系模型)的论文,奠定了关系数据库和 SQL 的基础。

20 世纪 70 年代中期,IBM 公司启动了 System/R 项目,目的是证明关系数据库概念的实用性,并提供一些实现关系 DBMS(数据库管理系统)的经验。该项目取得一定的成果,诞生了一个关系 DBMS 的模型系统,并开发出了一个名为 SEQUEL 的查询语言,即结构式英语查询语言(Structured English Query Language),后来由于法律方面的原因更名为 SQL。

System/R 的发表引起了加利福尼亚州 Menlo Park 公司的工程师们的注意,1977 年,他们组成一个称作 Relational Software Inc. 的公司,致力于关系数据库管理系统(RDBMS)的开发。1979 年,该公司将称为 Oracle 的商用 RDBMS 投放市场,这是世界上第一个商用 RDBMS 产品。Oracle 运行在 Digital VAX 小型机上,只用了两年时间就击败了 IBM 公司投放到市场的第一个产品。现在该公司(后更名为 Oracle Corporation)已成为世界上最大的数据库厂商。随后推出的产品还有 Ingres、DB2 等等。

随着关系数据库的普及,作为关系数据库操纵语言的 SQL 也迅速发展起来。1986 年美国国家标准局 ANSI 采用 SQL 作为关系数据库的标准操纵语言,奠定了 SQL 在关系数据库的统治地位。1987 年 ANSI 标准同时被国际标准化组织 ISO 采纳,成为国际标准。该标准在 1989 年稍加扩展,通常称作"SQL-89"和"SQL1",它是绝大多数商用产品的基础。

在制定 SQL1 时,为了照顾许多数据库厂商的利益,国际标准化组织将 SQL 语言的一部分从标准中分离出来,由实现者确定,造成了标准的不统一。为了弥补这个缺陷,ANSI 致力于制定一个更健全的标准——SQL2。与 SQL1 不同,SQL2 几乎不受现有 SQL 产品的限制。1992 年 10 月,SQL2 标准最终被采纳。

但是标准归标准,实现归实现,各个厂商推出的 SQL 产品均有所不同。Informix SQL 也是这样,它绝大部分都与 ANSI 标准兼容,但是对标准又作了一定的扩充,为某些语句增加了一部分标准没有规定的选项,或者没有实现标准中某些指定的内容。在使用 Informix-SQL 时,可以要求 SQL 处理程序检查输入的 SQL 语句是否使用了 Informix 的扩充成分。

本书主要针对 Informix 产品介绍 SQL,Informix 的扩充部分会特别指出来。

1.3 SQL 语句

Informix SQL 语言包括 47 条语句。每条语句都请求 DBMS 完成一个动作,如建立表、检索数据或更新数据。所有的 SQL 语句都具有基本相同的格式,如图 1-1 所示。

每条 SQL 语句都以一个动词开头,用于描述该语句的功能,如 select,delete,grant 等,

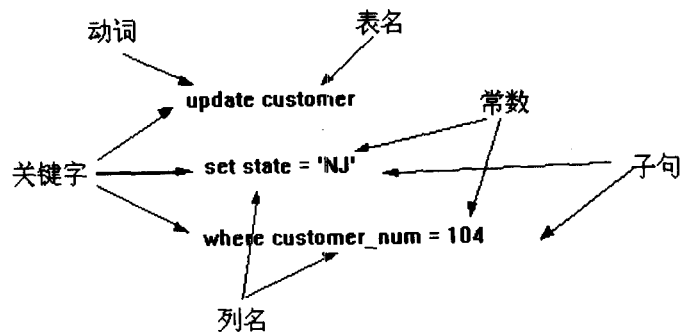

图 1-1　SQL 语句的格式

接着是一个或几个子句，子句指明该语句施加的对象或提供关于该语句行为的更详细说明。每个子句也以关键字开头，如 set，where，having 等。有的包含表名或列名，有的含有附加关键字、常数或表达式。

在本书中，我们用语法图的形式描述 SQL 语句的格式，如图 1-2 所示。每个图都从左上角的关键字开始，以短竖线结束。从开始到结束经过的任意一条路径都形成一条合法语句，关键字在语法图中以英文单词表示，如图 1-2 中的 delete，where，可变部分用汉字表示，这些项由用户每次书写该语句时确定，如表名、搜索条件。

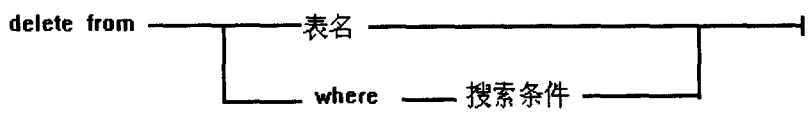

图 1-2　语法图示例

在 Informix SQL 中，关键字以及数据库对象名都是不区分大小写的，因此，delete 和 DELETE 对数据库来说是一样的。由用户定义的对象名（包括表名、列名、视图名、索引名等）必须由 1～18 个字符组成（数据库名限制在 10 个字符之内），以字母开头，不能包含空格和特殊符号。用户定义的对象名不能与系统的保留字相重。

下面是保留字的清单：

ADA	execute	order
all	exists	pascal
and	fetch	pli
any	float	precision
as	for	primary
asc	fortran	procedure
authorization	found	privileges
avg	from	public
begin	go	real
between	goto	rollback
by	group	schema
char	having	section
character	in	select

check	indicator	set
close	insert	smallint
cobol	int	some
commit	integer	sql
continue	into	sqlcode
count	is	sqlerror
create	language	sum
current	like	table
cursor	max	to
dec	min	union
decimal	module	unique
declare	not	update
delete	null	user
desc	numeric	values
distinct	of	view
double	on	whenever
end	open	where
escape	option	with
exec	or	work

1.4 数据类型

Informix SQL 支持的数据类型非常丰富，能满足绝大多数的应用需求。数据类型可分为四大类，分别是数值型、时间类型、字符类型和 BLOB 类型。

1.4.1 数值型

Informix 支持 8 种数值类型，为用户提供了丰富的选择余地，有些适合用作计数，有些可以用作保存工程数据，还有一些可以用作金融方面。

integer 和 smallint 类型

integer 和 smallint 类型有较小的表数范围，比较适合于用作计数、序列号、数据型标识。两种数据类型都是以带符号位的二进制方式存储的，integer 占用 32 位二进制位，可以表示从 -2^{31} 到 $2^{31}-1$（即 $-2147483648 \sim 2147483647$）的整数。smallint 占用 16 位二进制位，可以表示从 $-32767 \sim 32767$ 的十进制整数。

这两种数据类型的优点是占用较小的空间，因而运算速度比较快；缺点是它们的表数范围有限，比较容易溢出。

serial 类型

serial 类型在内部存储上类似于 integer 类型，也占用 32 位二进制位。当向表中插入一行数据时，数据库系统会自动为 serial 类型的列生成一个值。由于 serial 类型的数值可以自动产生，因此它特别适合作为标识。

缺省情况下，serial 类型的值从 1 变化到 $2^{31}-1$，当变化到 $2^{31}-1$ 时，若欲插入新行，serial 类型的值又绕回来，从 1 开始重新计数。不过这种情况较少遇到，毕竟 2^{31} 是一个足够大的数。

Informix 限制每一个数据库表只能有一个列是 serial 类型的，并且 serial 类型的值只增

长，当删除一行数据后，该行数据对应的数值不会被重新使用。

近似值：float和smallfloat类型。

这两种数据类型适用于不要求精度而要求量值的场合，如科学计算、工程问题等。使用这两种数据类型，可以很容易地记录地球到太阳的平均距离(1.5×10^{9}公里)或普朗克常数(6.625×10^{-34}焦·秒)。

float类型为双精度数，在计算机中用C语言的二进制浮点数表示，通常占用8个字节，有16位的十进制有效位；smallfloat类型是单精度的二进制浮点数，占用4个字节，大约有8位的十进制有效位。

浮点数的好处是，能同时保存非常大的数和非常小的数，并且运算速度较快；缺点是若数值超出精度范围时会被视为0。

decimal类型和money类型

许多商业应用需要以定点数方式存储数值。例如，现金要求保留小数点后两位小数。这类要求可用decimal类型和money类型实现。

decimal类型的形式为decimal(p,s)，p表示整个数据的有效位数，s表示小数点右边的有效位数，p的值可以从1到32，s值可以取0，此时表示存储的是一个整数。用图1-3可以很清楚地描述出p与s的关系。

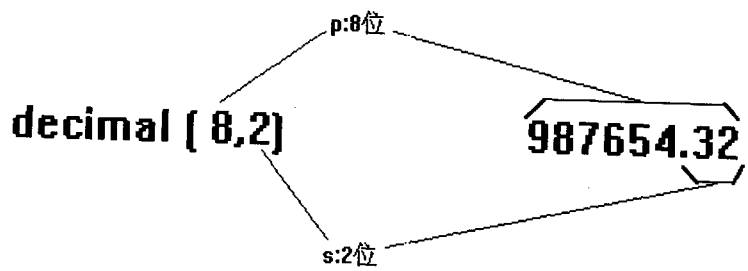

图1-3 p与s的关系

类似于float和smallfloat类型，decimal(p)也是浮点数，它们的主要差别是，我们可以调整decimal(p)类保留的有效位数，用P表示。P的值从1到32，表示1到32位十进制有效位，保存的数值范围从10^{-10}到10^{124}。

money类型与decimal类型在内部存储方面完全相同，只是系统在为显示money类型的数值而将它转换成字符时，会自动加上现金符号。

decimal(p,s)类型的主要优点是它的有效位数可调，并且调整的范围较大。主要缺点是算术运算速度慢，并且许多程序设计语言不支持这种格式的数据。

1.4.2 时间类型

Informix支持三种时间类型，date类型用于存储日期；datetime类型用于记录时间点，其精度可以从年变化到千分之一秒，inteval类型保存时间间隔。

date类型

date类型数值记录日期。date类型数值在内部以一个带符号的整数存储，占用4个字节，保存从1900年1月1日零点开始的正数。本世纪以后的日期存储为正数，本世纪以前的

日期存储为负数。

datetime 类型

datetime 类型用于记录时间点，表示的精度可以任意指定，可包含年份、月份、日、时、分、秒、千分之一秒等等，共有 28 种形式，每种形式的占用空间各不相同，从 2 字节到 11 字节不等。见表 1-1。

表 1-1 datetime 类型表示精度与所占字节数的对照表

精 度	占用字节数	精 度	占用字节数
year to year	3	day to hour	3
year to month	4	day to minute	4
year to day	5	day to second	5
year to hour	6	day to fraction	5+f/2
year to minute	7	hour to hour	2
year to second	8	hour to minute	3
year to fraction(f)	8+f/2	hour to second	4
month to month	2	hour to fraction(f)	4+f/2
month to day	3	minute to minute	2
month to hour	4	minute to second	3
month to minute	5	minute to fraction(f)	3+f/2
month to second	6	second to second	2
month to fraction(f)	6+6/2	second to fraction(f)	2+f/2
day to day	2	fraction to fraction(f)	1+f/2

interval 类型

interval 类型的数值用于表示两个时间点之间的间隔。例如，某人某年的工作时间为 254 天；飞机飞行了 2 小时 12 分等等。与 datetime 类似，interval 类型也可以定义为不同的精度，其占用空间的大小从 2 字节到 12 字节不等。见表 1-2。

表 1-2 interval 类型表示精度与所占字节数的对照表

精 度	占用字节数	精 度	占用字节数
year(p) to year	1+p/2	hour(p) to minute	2+p/2
year(p) to month	2+p/2	hour(p) tosecond	3+p/2
month(p) to month	1+p/2	hour(p) to fraction(f)	4+(p+f)/2
day(p) to day	1+p/2	minute(p) to minute	1+p/2
day(p) to hour	2+p/2	minute(p) to second	2+p/2
day(p) to minute	3+p/2	minute(p) to fraction(f)	3+(p+f)/2
day(p) to second	4+p/2	second(p) to second	1+p/2
day(p) to fration(f)	5+(p+f)/2	second(p) to fraction	2+(p+f)/2
hour(p) to hour	1+p/2	fraction to fraction(f)	1+f/2

1.4.3 字符类型

Informix支持的字符类型主要有两类,char(n)和varchar(n)。这两类都可以用于存储字符串,字符串的长度由n指定。char类型最多可存储32767个字符,varchar类型的最大长度为255个字符。

char类型占用的存储空间大小是固定的,与所存储字符串的长短无关。例如,定义某一列的类型为char(400),而我们只存储了一个包含100个字符的字串,但是该列依然占据400个字节的存储空间。

varchar类型占用的存储空间在一定范围内可变,依据所存数据的不同而不同。例如,定义某一列的类型为varchar(100),若我们只保存了30个字符,则该列占用31个字节的存储空间,额外的一个字节记录字串的长度。

若我们将一个实际长度大于定义长度的字符串保存到某列中,则字符串会被截断。

1.4.4 BLOB类型

BLOB代表Binary Large Object,是任意数值和长度的字节流。BLOB数据可能是扫描到计算机中的图片、一篇文章或一段音乐等等。Informix支持两种类型的BLOB数据:text和byte。

text只能存储可打印的ASCII字符以及ctrl-j,ctrl-i和ctrl-l。因此,text类型的列中适宜存储文档性质的数据,如源程序、文章、工程规范等等。

byte可能存储任意的数值,如图纸、可执行程序、电影片断等等。

BLOB类型数据的长度可以很大,在Informix中,限制其长度为21亿字节。

1.5 表达式

SQL语言的表达式用于检索数据库的值和向数据库中插入数值。表达式主要用于select语句、带条件的delete语句和update语句以及execute procedure语句。

1.5.1 SQL表达式语法

SQL表达式语法见图1-4,该图仅列出表达式的主要组成部分,详细的语法参见相关手册。SQL表达式有五种基本形式,通过算术运算符或串接运算符将基本形式连接成复杂表达式。

1.5.2 列表达式

列表达式由列名或取列的子串构成,如:

```
company
customer.lname
phone[1,3]
```

其中第二种形式为列名的限定性引用,左边是表名或表别名,用于区分来自于不同表中的同名列名。第三种形式是取列的子串,此时,该列的数据类型只能为:char,var char,byte

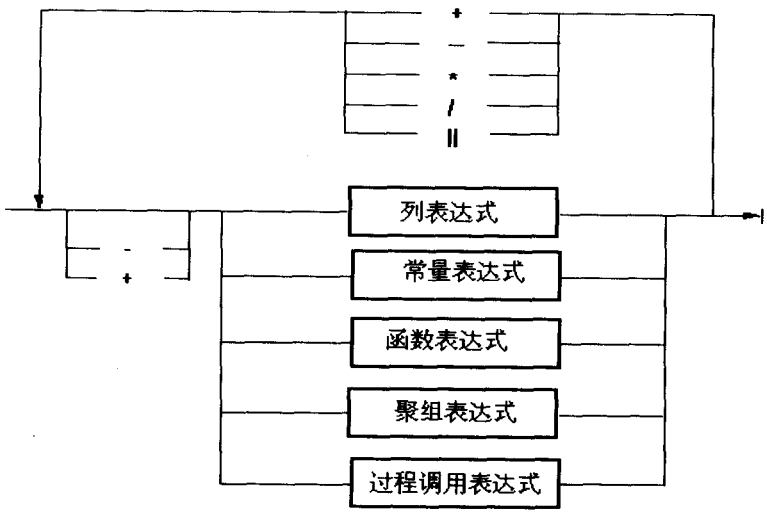

图 1-4 SQL 表达式语法

或 text。下标指明子串的起始位置和终止位置。

此外，列表达式还可以取 rowid，rowid 是每张表的一个隐含列，非分割表的每一行数据都有唯一的 rowid 值。

1.5.3 常量表达式

常量表达式由字符串、数字、时间常量或系统常量构成，如：

```
'Name:'
123.5
interval (4 10:10) day to minute
datetime (4 10:10) day to minute
TODAY
DBSERVERNAME
```

其中，TODAY 和 DBSERVERNAME 为系统常量，TODAY 表示系统日期，DBSERVERNAME 代表数据库服务器的名字。此外，USER 返回当前用户的登录名，CURRENT 返回当前时间，为 datetime 类型的常量。

1.5.4 函数表达式

函数表达式由函数调用构成，下面是函数表达式的一些例子：

```
MDY(7,6,1998)
LENGTH('wang')
HEX(customer_num)
EXP(4,3)
```

Informix SQL 提供了丰富的函数，有关函数详情参见 1.6 节。

1.5.5 聚组表达式

聚组表达式由聚组函数构成，用于计算一组数值的统计结果。有关聚组函数的使用参见

第5章。

1.5.6 过程调用表达式

该表达式由存储过程名和实参值组成，如：

```
read_address('Miller')
read_address(lastname='Miller')
```

有关存储过程的使用参见第9章。

1.5.7 算术运算符

算术运算符包括+、-、*、/，用算术运算符可将基本的表达式组合成复杂表达式。如：

```
quantity * anit_price
count(*)/2
```

如果在计算表达式时遇到null值，则整个表达式的值为null值。

1.5.8 串接运算符

串接运算符为两个相邻的管道符"||"，用于连接两个表达式，下面列出串接运算符的一些形式：

```
phone [1,3]||phone[5,12]
'Date:|| TODAY
```

1.6 函 数

Informix SQL为使用者提供了大量函数，本节简要介绍这些函数的功能和使用方法。

1.6.1 算术函数

算术函数包括：

- ABS(数值型表达式)
- MOD(被除数，除数)
- POW(基数，指数)
- ROOT(数值型表达式,根次)
- ROUND(表达式，精度)
- SQRT(数值型表达式)
- TRUNC(表达式，精度)

ABS函数返回给定表达式的绝对值，参数为一个数值型表达式，返回值与参数类型一致。MOD函数计算被除数与除数的余数(模)，返回值为integer类型，除数不能为0。例如，下面函数的值为1：

```
MOD(10,3)
```