

# Tableau 函数（按字母顺序）

版本: 2018.3 适用于: Tableau Desktop, Tableau Online, Tableau Server

本参考中的 Tableau 函数按字母顺序进行组织。单击某个字母以查看以它开头的函数。如果没有以该字母开头的函数，则将显示以字母表中的下一个字母开头的函数。您也可以按 Ctrl+F（在 Mac 上按 Command-F）打开一个搜索框，您可以使用它来搜索特定功能的页面。

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

## ABS(number)

返回给定数字的绝对值。

### 示例

```
ABS(-7) = 7
```

```
ABS([Budget Variance])
```

第二个示例返回 Budget Variance 字段中包含的所有数字的绝对值。

## ACOS(number)

返回给定数字的反余弦。结果以弧度表示。

### 示例

```
ACOS(-1) = 3.14159265358979
```

## ASCII(string)

返回 string 的第一个字符的 ASCII 代码。

### 示例

```
ASCII('A') = 65
```

## ASIN(number)

返回给定数字的反正弦。结果以弧度表示。

### 示例

```
ASIN(1) = 1.5707963267949
```

## ATAN(number)

返回给定数字的反正切。结果以弧度表示。

## 示例

```
ATAN(180) = 1.5652408283942
```

## ATAN2(y number, x number)

返回两个给定数字（**x** 和 **y**）的反正切。结果以弧度表示。

## 示例

```
ATAN2(2, 1) = 1.10714871779409
```

## ATTR(expression)

如果它的所有行都有一个值，则返回该表达式的值。否则返回星号。会忽略 Null 值。

## AVG(expression)

返回表达式中所有值的平均值。AVG 只能用于数字字段。会忽略 Null 值。

[返回顶部](#)

---

## CASE expression WHEN value1 THEN return1 WHEN value2 THEN return2...ELSE default return END

使用 CASE 函数执行逻辑测试并返回合适值。CASE 比 IIF 或 IF THEN ELSE 更易于使用。CASE 函数可评估 `expression`，并将其与一系列值（`value1`、`value2` 等）比较，然后返回结果。遇到一个与 `expression` 匹配的值时，CASE 返回相应的返回值。如果未找到匹配值，则使用默认返回表达式。如果不存在默认返回表达式并且没有任何值匹配，则会返回 Null。

通常，您使用一个 IF 函数来执行一系列任意测试，并使用 CASE 函数搜索与表达式的匹配值。但 CASE 函数都可以重写为 IF 函数，不过 CASE 函数一般更加简明。

很多时候可以使用组获得与复杂 case 函数相同的结果。

## 示例

```
CASE [Region] WHEN 'West' THEN 1 WHEN 'East' THEN 2 ELSE 3 END
```

```
CASE LEFT(DATENAME('weekday',[Order Date]),3) WHEN 'Sun' THEN 0 WHEN 'Mon' THEN 1 WHEN 'Tue' THEN 2 WHEN 'Wed' THEN 3 WHEN 'Thu' THEN 4 WHEN 'Fri' THEN 5 WHEN 'Sat' THEN 6 END
```

## CEILING(数字)

将数字舍入为值相等或更大的最近整数。

## 示例

```
CEILING(3.1415) = 4
```

---

## 按数据源的可用性

数据源	支持
Microsoft Access	不支持
Microsoft Excel	支持
文本文件	支持
统计文件	支持
Tableau Server	支持
Actian Vectorwise	不支持
Amazon Aurora	不支持
Amazon EMR Hadoop Hive	支持
Amazon Redshift	不支持
Aster Database	不支持
Cloudera Hadoop	支持
DataStax Enterprise	支持
EXASOL	不支持
Firebird	不支持
Google Analytics	支持
Google BigQuery	支持
Google Cloud SQL	不支持
Hortonworks Hadoop Hive	支持
IBM BigInsights	不支持
IBM DB2	不支持
IBM PDA (Netezza)	不支持
MapR Hadoop Hive	支持
MarkLogic	不支持
Microsoft Analysis Services	不支持
Microsoft PowerPivot	不支持
Microsoft SQL Server	不支持
MySQL	不支持
Oracle	不支持
Oracle Essbase	不支持
Actian Matrix (ParAccel)	不支持
Pivotal Greenplum	不支持
PostgreSQL	不支持
Progress OpenEdge	不支持

Salesforce	支持
SAP HANA	不支持
SAP Sybase ASE	不支持
SAP Sybase IQ	不支持
Spark SQL	支持
Splunk	不支持
Teradata	不支持
Teradata OLAP Connector	不支持
Vertica	不支持

### CHAR(number)

返回通过 ASCII 代码 *number* 编码的字符。

#### 示例

`CHAR(65) = 'A'`

### COLLECT(spatial)

将参数字段中的值组合在一起的聚合计算。会忽略 Null 值。

**注意：**COLLECT 函数只能用于空间字段。

#### 示例

`COLLECT ([Geometry])`

### CONTAINS(string, substring)

如果给定字符串包含指定子字符串，则返回 true。

#### 示例

`CONTAINS ("Calculation", "alcu") = true`

### CORR(expression 1, expression2)

返回两个表达式的皮尔森相关系数。

皮尔森相关系数衡量两个变量之间的线性关系。结果范围为 -1 至 +1（包括 -1 和 +1），其中 1 表示精确的正向线性关系，比如一个变量中的正向更改即表示另一个变量中对应量级的正向更改，0 表示方差之间没有线性关系，而 -1 表示精确的反向关系。

CORR 可用于以下数据源：

- Tableau 数据提取（您可以从任何数据源中创建数据提取）
- Cloudera Hive
- EXASOL
- Firebird（版本 3.0 及更高版本）

- Google BigQuery
- Hortonworks Hadoop Hive
- IBM PDA (Netezza)
- Oracle
- PostgreSQL
- Presto
- SybaseIQ
- Teradata
- Vertica

对于其他数据源，请考虑提取数据或使用 WINDOW\_CORR。请参见[表计算函数 \(functions\\_functions\\_tablecalculation.htm\)](#)。

注意：CORR 结果的平方等于线性趋势线模型的 R 平方值。请参见[趋势线模型术语 \(trendlines\\_add.htm#Terms\)](#)。

## 示例

您可以使用 CORR 在解聚散点图中呈现关联。实现此目的的方式是使用表范围详细级别表达式。例如：

```
{CORR(Sales, Profit)}
```

借助详细级别表达式，关联将在所有行上运行。如果您使用像 CORR(Sales, Profit)（不带两边的方括号可使其成为详细级别表达式）这样的公式，视图将显示散点图中每个单独的点与其他每个点（未定义）的关联。

请参见[表范围 \(calculations\\_calculatedfields\\_lod.htm#Table\)](#)。

## COS(number)

返回角度的余弦。以弧度为单位指定角度。

## 示例

```
COS(PI ( ) /4) = 0.707106781186548
```

## COT(number)

返回角度的余切。以弧度为单位指定角度。

## 示例

```
COT(PI ( ) /4) = 1
```

## COUNT(expression)

返回组中的项目数。不对 Null 值计数。

## COUNTD(expression)

返回组中不同项目的数量。不对 Null 值计数。此函数在下列情况下不可用：在 Tableau Desktop 8.2 之前创建的使用 Microsoft Excel 或文本文件数据源的工作簿、使用旧版连接的工作簿和使用 Microsoft Access 数据源的工作簿。将数据提取到数据提取文件以使用此函数。请参见[提取数据 \(extracting\\_data.htm\)](#)。

## COVAR(expression 1, expression2)

返回两个表达式的 *样本协方差*。

协方差对两个变量的共同变化方式进行量化。正协方差指明两个变量趋向于向同一方向移动，平均来说，即一个变量的较大值趋向于与另一个变量的较大值对应。样本协方差使用非空数据点的数量  $n - 1$  来规范化协方差计算，而不是使用总体协方差（可用于 COVARP 函数）所使用的  $n$ 。当数据是用于估算较大总体的协方差的随机样本时，则样本协方差是合适的选择。

COVAR 可用于以下数据源：

- Tableau 数据提取（您可以从任何数据源中创建数据提取）
- Cloudera Hive
- EXASOL
- Firebird（版本 3.0 及更高版本）
- Google BigQuery
- Hortonworks Hadoop Hive
- IBM PDA (Netezza)
- Oracle
- PostgreSQL
- Presto
- SybaseIQ
- Teradata
- Vertica

对于其他数据源，请考虑提取数据或使用 WINDOW\_COVAR。请参见[表计算函数 \(functions\\_functions\\_tablecalculation.htm\)](#)。

如果 expression1 和 expression2 相同 — 例如，COVAR([profit], [profit]) — 则 COVAR 将返回一个值，指明值分布的广泛程度。

注意：COVAR(X, X) 的值等于 VAR(X) 的值，也等于 STDEV(X)^2 的值。

### 示例

以下公式返回“Sales”和“Profit”的样本协方差。

```
COVAR([Sales], [Profit])
```

## COVARP(expression 1, expression2)

返回两个表达式的 *总体协方差*。

协方差对两个变量的共同变化方式进行量化。正协方差指明两个变量趋向于向同一方向移动，平均来说，即一个变量的较大值趋向于与另一个变量的较大值对应。总体协方差等于样本协方差除以  $(n-1)/n$ ，其中  $n$  是非空数据点的总数。如果存在可用于所有相关项的数据，则总体协方差是合适的选择，与之相反，在只有随机项子集的情况下，样本协方差（及 COVAR 函数）较为适合。

COVARP 可用于以下数据源：

- Tableau 数据提取（您可以从任何数据源中创建数据提取）
- Cloudera Hive
- EXASOL
- Firebird（版本 3.0 及更高版本）
- Google BigQuery

- Hortonworks Hadoop Hive
- IBM PDA (Netezza)
- Oracle
- PostgreSQL
- Presto
- SybaseIQ
- Teradata
- Vertica

对于其他数据源，请考虑提取数据或使用 WINDOW\_COVARP。请参见[表计算函数 \(functions\\_functions\\_tablecalculation.htm\)](#)。

如果 expression1 和 expression2 相同 — 例如，COVARP([profit], [profit]) — 则 COVARP 将返回一个值，指明值分布的广泛程度。

注意：COVARP(X, X) 的值等于 VARP(X) 的值，也等于 STDEVP(X)^2 的值。

## 示例

以下公式返回“**Sales**”和“**Profit**”的总体协方差。

```
COVARP([Sales], [Profit])
```

[返回顶部](#)

## DATE(expression)

在给定数字、字符串或日期表达式的情况下返回日期。

## 示例

```
DATE([Employee Start Date])
```

```
DATE("April 15, 2004") = #April 15, 2004#
```

```
DATE("4/15/2004")
```

```
DATE(#2006-06-15 14:52#) = #2006-06-15#
```

第二和第三个示例中的引号不可省略。

## DATEADD(date\_part, interval, date)

返回指定日期，该日期的指定 date\_part 中添加了指定的数字 interval。

## 示例

```
DATEADD('month', 3, #2004-04-15#) = 2004-07-15 12:00:00 AM
```

该表达式会向日期 #2004-04-15# 添加三个月。

## DATEDIFF(date\_part, date1, date2, [start\_of\_week])

返回 date1 与 date2 之差（以 date\_part 的单位表示）。

`start_of_week` 参数（可用于指定哪一天是一周的第一天）是可选的。可能的值为“monday”、“tuesday”等。如果省略，一周的开始由数据源确定。请参见[数据源的日期属性 \(date\\_properties.htm\)](#)。

## 示例

```
DATEDIFF('week', #2013-09-22#, #2013-09-24#, 'monday') = 1
```

```
DATEDIFF('week', #2013-09-22#, #2013-09-24#, 'sunday') = 0
```

第一个表达式返回 1，因为当 `start_of_week` 为 'monday' 时，9 月 22（星期日）和 9 月 24（星期二）不属于同一周。第一个表达式返回 0，因为当 `start_of_week` 为 'sunday' 时，9 月 22（星期日）和 9 月 24（星期二）属于同一周。

## DATENAME(date\_part, date, [start\_of\_week])

以字符串的形式返回 `date` 的 `date_part`。`start_of_week` 参数（可用于指定哪一天是一周的第一天）是可选的。可能的值为“monday”、“tuesday”等。如果忽略 `start_of_week`，则一周的开始由数据源确定。请参见[数据源的日期属性 \(date\\_properties.htm\)](#)。

## 示例

```
DATENAME('year', #2004-04-15#) = "2004"
```

```
DATENAME('month', #2004-04-15#) = "April"
```

## DATEPARSE(format, string)

将字符串转换为指定格式的日期时间。是否支持某些区域设置特定的格式由计算机的系统设置确定。数据中出现的不需要解析的字母应该用单引号 (') 引起来。对于值之间没有分隔符的格式（如 `Mmddyy`），请验证它们是否按预期方式解析。该格式必须是常量字符串，而非字段值。如果数据与格式不匹配，此函数将返回 Null。

此函数可用于多种连接器。有关详细信息，请参见[将字段转换为日期字段 \(data\\_dateparse.htm\)](#)。

## 示例

```
DATEPARSE ("dd.MMMM.yyyy", "15.April.2004") = #April 15, 2004#
```

```
DATEPARSE ("h'h' m'm' s's'", "10h 5m 3s") = #10:05:03#
```

## DATEPART(date\_part, date, [start\_of\_week])

以整数的形式返回 `date` 的 `date_part`。

`start_of_week` 参数（可用于指定哪一天是一周的第一天）是可选的。可能的值为“monday”、“tuesday”等。如果忽略 `start_of_week`，则一周的开始由数据源确定。请参见[数据源的日期属性 \(date\\_properties.htm\)](#)。

**注意：** 当 `date_part` 为工作日时，会忽略 `start_of_week` 参数。这是因为 Tableau 依赖固定工作日顺序来应用偏移。

## 示例

```
DATEPART('year', #2004-04-15#) = 2004
```

```
DATEPART('month', #2004-04-15#) = 4
```



## DATETIME(expression)

在给定数字、字符串或日期表达式的情况下返回日期时间。

### 示例

```
DATETIME("April 15, 2005 07:59:00") = April 15, 2005 07:59:00
```

## DATETRUNC(date\_part, date, [start\_of\_week])

按 **date\_part** 指定的准确度截断指定日期。此函数返回新日期。例如，以月份级别截断处于月份中间的日期时，此函数返回当月的第一天。**start\_of\_week** 参数（可用于指定哪一天是一周的第一天）是可选的。可能的值为“monday”、“tuesday”等。如果忽略 **start\_of\_week**，则一周的开始由数据源确定。请参见[数据源的日期属性 \(date\\_properties.htm\)](#)。

### 示例

```
DATETRUNC('quarter', #2004-08-15#) = 2004-07-01 12:00:00 AM
```

```
DATETRUNC('month', #2004-04-15#) = 2004-04-01 12:00:00 AM
```

## DAY(date)

以整数的形式返回给定日期的天。

### 示例

```
DAY(#2004-04-12#) = 12
```

## DEGREES(number)

将以弧度表示的给定数字转换为度数。

### 示例

```
DEGREES(PI() / 4) = 45.0
```

## DIV(整数 1, 整数 2)

返回将整数 1 除以整数 2 的除法运算的整数部分。

### 示例

```
DIV(11,2) = 5
```

## DOMAIN(string\_url)

*注意：* 仅在连接到 Google BigQuery 时才受支持

在给定 URL 字符串的情况下返回作为字符串的域。

## 示例

```
DOMAIN('http://www.google.com:80/index.html') = 'google.com'
```

[返回顶部](#)

---

## ENDSWITH(string, substring)

如果给定字符串以指定子字符串结尾，则返回 **true**。会忽略尾随空格。

## 示例

```
ENDSWITH("Tableau", "leau") = true
```

## EXP(number)

返回 e 的给定数字次幂。

## 示例

```
EXP(2) = 7.389
```

```
EXP(-(Growth Rate)*[Time])
```

[返回顶部](#)

---

## FIND(string, substring, [start])

返回 substring 在 string 中的索引位置，如果未找到 substring，则返回 0。如果添加了可选参数 start，则函数会忽略在索引位置 start 之前出现的任何 substring 实例。字符串中第一个字符的位置为 1。

## 示例

```
FIND("Calculation", "alcu") = 2
```

```
FIND("Calculation", "Computer") = 0
```

```
FIND("Calculation", "a", 3) = 7
```

```
FIND("Calculation", "a", 2) = 2
```

```
FIND("Calculation", "a", 8) = 0
```

## FINDNTH(string, substring, occurrence)

返回指定字符串内的第 n 个子字符串的位置，其中 n 由 occurrence 参数定义。

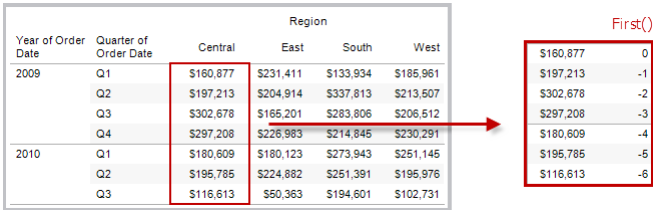
**注意：** 所有数据源都不可使用 FINDNTH。

示例

```
FINDNTH("Calculation", "a", 2) = 7
```

FIRST()

返回从当前行到分区中第一行的行数。例如，下面的视图显示每季度销售额。在 Date 分区中计算 FIRST() 时，第一行与第二行之间的偏移为 -1。



示例

当前行索引为 3 时，FIRST() = -2。

FLOAT(expression)

将其参数转换为浮点数。

示例

```
FLOAT(3) = 3.000
```

FLOAT([Age]) 将 Age 字段中的每个值转换为浮点数。

FLOOR(数字)

将数字舍入为值相等或更小的最近整数。

示例

```
FLOOR(3.1415) = 3
```

按数据源的可用性

数据源	支持
Microsoft Access	不支持
Microsoft Excel	支持
文本文件	支持
统计文件	支持
Tableau Server	支持

Action Vector	不支持
Amazon Aurora	不支持
Amazon EMR Hadoop Hive	支持
Amazon Redshift	不支持
Aster Database	不支持
Cloudera Hadoop	支持
DataStax Enterprise	支持
EXASOL	不支持
Firebird	不支持
Google Analytics	支持
Google BigQuery	支持
Google Cloud SQL	不支持
Hortonworks Hadoop Hive	支持
IBM BigInsights	不支持
IBM DB2	不支持
IBM Netezza	不支持
MapR Hadoop Hive	支持
MarkLogic	不支持
Microsoft Analysis Services	不支持
Microsoft PowerPivot	不支持
Microsoft SQL Server	不支持
MySQL	不支持
Oracle	不支持
Oracle Essbase	不支持
ParAccel	不支持
Pivotal Greenplum	不支持
PostgreSQL	不支持
Progress OpenEdge	不支持
Salesforce	支持
SAP HANA	不支持
SAP Sybase ASE	不支持
SAP Sybase IQ	不支持
Spark SQL	支持
Splunk	不支持
Teradata	不支持

Teradata OLAP Connector	不支持
Vertica	不支持

## FULLNAME( )

返回当前用户的全名。当用户已登录时，这是 Tableau Server 或 Tableau Online 全名；否则为 Tableau Desktop 用户的本地或网络全名。

### 示例

```
[Manager]=FULLNAME( )
```

如果经理 Dave Hallsten 已登录，则仅当视图中的“Manager”字段包含“Dave Hallsten”时，此示例才会返回 True。用作筛选器时，此计算字段可用于创建用户筛选器，该筛选器仅显示与登录到服务器的人员相关的数据。

[返回顶部](#)

## GET\_JSON\_OBJECT(JSON 字符串, JSON 路径)

*注意：* 仅在连接到Hadoop Hive 时才受支持。

根据 JSON 路径返回 JSON 字符串中的 JSON 对象。

## GROUP\_CONCAT(表达式)

*注意：* 仅在连接到Google BigQuery 时才受支持

将来自每个记录的值连接为一个由逗号分隔的字符串。此函数在处理字符串时的作用类似于 SUM()。

### 示例

```
GROUP_CONCAT(Region) = "Central,East,West"
```

[返回顶部](#)

## HEXBINX(number, number)

将 x、y 坐标映射到最接近的六边形数据桶的 x 坐标。数据桶的边长为 1，因此，可能需要相应地缩放输入。

HEXBINX 和 HEXBINY 是用于六边形数据桶的分桶和标绘函数。六边形数据桶是对 x/y 平面（例如地图）中的数据进行可视化的有效而简洁的选项。由于数据桶是六边形的，因此每个数据桶都非常近似于一个圆，并最大程度地减少了从数据点到数据桶中心的距离变化。这使得聚类分析更加准确并且能提供有用的信息。

### 示例

```
HEXBINX([Longitude],[Latitude])
```

## HEXBINY(number, number)

将 x、y 坐标映射到最接近的六边形数据桶的 y 坐标。数据桶的边长为 1，因此，可能需要相应地缩放输入。

### 示例

```
HEXBINARY([Longitude], [Latitude])
```

# HOST(string\_url)

*注意：* 仅在连接到Google BigQuery 时才受支持

在给定的 URL 字符串的情况下返回作为字符串的主机名。

## 示例

```
HOST('http://www.google.com:80/index.html') = 'www.google.com:80'
```

[返回顶部](#)

# IF test THEN value END / IF test THEN value ELSE else END

使用 IF THEN ELSE 函数执行逻辑测试并返回合适值。IF THEN ELSE 函数计算一系列测试条件并返回第一个 true 条件的值。如果没有条件为 true，则返回 ELSE 值。每个测试都必须为布尔值：可以为数据源中的布尔字段或为逻辑表达式的结果。最后一个 ELSE 是可选的，但是如果未提供它并且没有任何 true 测试表达式，则函数返回 Null。所有值表达式都必须为相同类型。

## 示例

```
IF [Cost]>[Budget Cost] THEN 'Over Budget' ELSE 'Under Budget' END

IF [Budget Sales]!=0 THEN [Sales]/[Budget Sales] END
```

# IF test1 THEN value1 ELSEIF test2 THEN value2 ELSE else END

使用此版本的 IF 函数递归地执行逻辑测试。对于 IF 函数中的 ELSEIF 值的数量没有固有限制，但是各个数据库可能会对 IF 函数的复杂度有所限制。尽管 IF 函数可以重写为一系列嵌套 IIF 语句，不过在表达式计算方式方面却有所差异。具体而言，IIF 语句会区分 TRUE、FALSE 和 UNKNOWN，而 IF 语句仅关注 TRUE 和非 true（包括 FALSE 和 UNKNOWN）。

## 示例

当您依据度量创建数据桶时，默认情况下 Tableau 将会创建大小相等的数据桶。例如，假定您有一个代表年龄的度量。当您依据该度量创建数据桶时，Tableau 会使所有数据桶大小相等。您可以指定所需的数据桶大小，但无法为每个数据桶指定单独的值范围。解决此限制的方法是创建一个计算字段来定义数据桶。然后，您可以为年龄 0 - 20 创建一个数据桶，为年龄 21 - 32 创建另一个数据桶，依此类推。以下过程演示如何能执行该操作。

- 1. 通过选择“分析”>“创建计算字段”来创建一个新计算字段。
- 2. 将该字段命名为“Age Groups”（年龄组），并在定义区域中键入以下内容

```
IF

[Age] < 21 THEN 'Under 21'

ELSEIF

[Age] <= 32 THEN '21-32'
```

```
ELSEIF

[Age] <= 42 THEN '33-42'

ELSEIF

[Age] <= 52 THEN '43-52'

ELSEIF

[Age] <= 64 THEN '53-64'

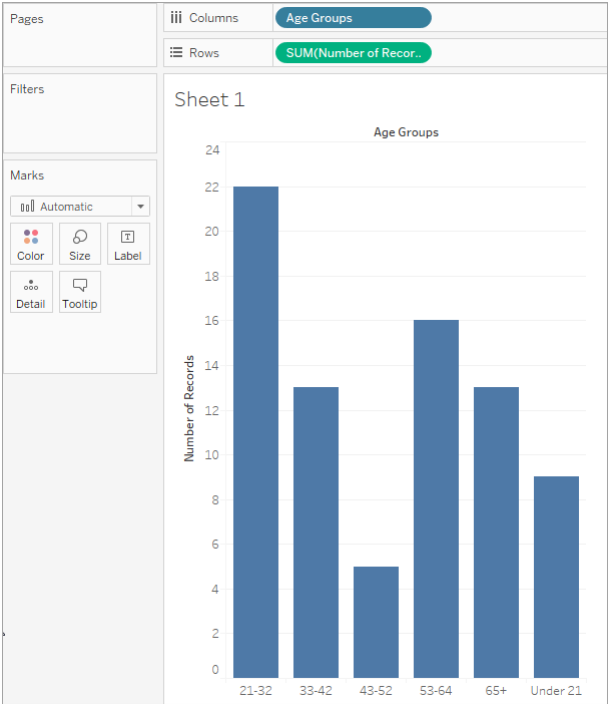
ELSE '65+'

END
```

确认状态消息指明公式有效，然后单击“确定”。

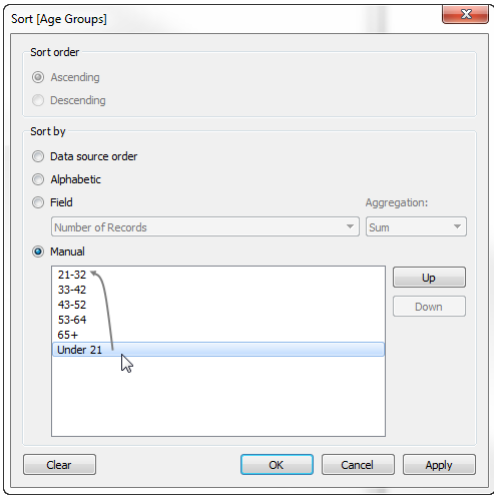
- 3. 从“数据”窗格的“度量”区域中，将“Number of Records”（记录数）拖到“行”。
- 4. 从“数据”窗格的“维度”区域中，将“Age Groups”（年龄组）拖到“列”。

记录现在将在您定义的六个数据桶之间分配：



很遗憾，“Under 21”（21 岁以下）数据桶位于最右侧，而您希望它位于最左侧。Tableau 足够聪明，知道按正确的顺序放置完全以数字组成的名称，但它无法推测以“Under”开头的数据桶名称属于左侧。通过手动排序来修复问题。

- 5. 在“列”上单击“Age Groups”（年龄组）字段右侧的向下箭头，然后单击“排序”。选择“手动”，然后将“Under 21”（21 岁以下）数据桶上移到列表的顶部：



您的视图现在已完成。

### IIF(test, then, else, [unknown])

使用 IIF 函数执行逻辑测试并返回合适值。第一个参数 `test` 必须是布尔值：数据源中的布尔字段或使用运算符的逻辑表达式的结果（或 AND、OR 或 NOT 的逻辑比较）。如果 `test` 计算为 `TRUE`，则 IIF 返回 `then` 值。如果 `test` 计算为 `FALSE`，则 IIF 返回 `else` 值。

布尔比较还可生成值 `UNKNOWN`（既不是 `TRUE` 也不是 `FALSE`），通常是因为测试中存在 `Null` 值。在比较结果为 `UNKNOWN` 时，会返回 IIF 的最后一个参数。如果省略此参数，则会返回 `Null`。

### 示例

```
IIF(7>5, 'Seven is greater than five', 'Seven is less than five')

IIF([Cost]>[Budget Cost], 'Over Budget', 'Under Budget')

IIF([Budget Sales]!=0,[Sales]/[Budget Sales],0)

IIF(Sales>=[Budget Sales], 'Over Cost Budget and Over Sales Budget', 'Over Cost Budget and Under Sales Budget','Under Cost Budget')
```

### IFNULL(expression1, expression2)

如果结果不为 `null`，则 IFNULL 函数返回第一个表达式，否则返回第二个表达式。

### 示例

```
IFNULL([Profit], 0) = [Profit]
```

### INDEX()

返回分区中当前行的索引，不包含与值有关的任何排序。第一个行索引从 1 开始。例如，下表显示每季度销售额。当在 `Date` 分区中计算 `INDEX()` 时，各行的索引分别为 1、2、3、4 等。



Year of Order Date	Quarter of Order Date	Region			
		Central	East	South	West
2009	Q1	\$160,877	\$231,411	\$133,934	\$185,961
	Q2	\$197,213	\$204,914	\$337,813	\$213,507
	Q3	\$302,678	\$165,201	\$283,806	\$206,512
	Q4	\$297,208	\$226,983	\$214,845	\$230,291
2010	Q1	\$180,609	\$180,123	\$273,943	\$251,145
	Q2	\$195,785	\$224,882	\$251,391	\$195,976
	Q3	\$116,613	\$50,363	\$194,601	\$102,731

INDEX()

\$160,877	1
\$197,213	2
\$302,678	3
\$297,208	4
\$180,609	5
\$195,785	6
\$116,613	7

示例

对于分区中的第三行，INDEX() = 3。

INT(expression)

将其参数转换为整数。对于表达式，此函数将结果截断为最接近于 0 的整数。

示例

```
INT(8.0/3.0) = 2
INT(4.0/1.5) = 2
INT(0.50/1.0) = 0
INT(-9.7) = -9
```

字符串转换为整数时会先转换为浮点数，然后舍入。

ISDATE(string)

如果字符串参数可以转换为日期，则 ISDATE 函数返回 TRUE，否则返回 FALSE。

示例

```
ISDATE('January 1, 2003') = TRUE
ISDATE('Jan 1 2003') = TRUE
ISDATE('1/1/03') = TRUE
ISDATE('Janxx 1 2003') = FALSE
```

ISFULLNAME(string)

如果当前用户的全名与指定的全名匹配，则返回 true；如果不匹配，则返回 false。当用户已登录时，此函数使用 Tableau Server 或 Online 全名；否则它使用 Tableau Desktop 用户的本地或网络全名。

示例

```
ISFULLNAME("Dave Hallsten")
如果 Dave Hallsten 为当前用户，则此示例返回 true，否则返回 false。
```

ISNULL(expression)

如果表达式为 Null，则 ISNULL 函数返回 TRUE，否则返回 FALSE。

示例

以下示例将 ISNULL 与 IIF 结合使用，将 null 值替换为 0。

```
IIF(ISNULL([Sales]), 0,[Sales] )
```

ISMEMBEROF(string)

如果当前使用 Tableau 的人员是与给定字符串匹配的组的成员，则返回 true。如果当前使用 Tableau 的人员已登录，则组成员身份由 Tableau Server 或 Tableau Online 上的组确定。如果该人员未登录，则此函数返回 false。

示例

```
IF ISMEMBEROF("Sales") THEN "Sales" ELSE "Other" END
```

ISUSERNAME(string)

如果当前用户的用户名与指定的用户名匹配，则返回 true；如果不匹配，则返回 false。当用户已登录时，此函数使用 Tableau Server 或 Online 用户名；否则它使用 Tableau Desktop 用户的本地或网络用户名。

示例

```
ISUSERNAME("dhallsten")
```

如果 dhallsten 为当前用户，则此示例返回 true，否则返回 false。

[返回顶部](#)

LAST()

返回从当前行到分区中最后一行的行数。例如，下表显示每季度销售额。在 Date 分区中计算 LAST() 时，最后一行与第二行之间的偏移为 5。

Year of Order Date	Quarter of Order Date	Region			
		Central	East	South	West
2009	Q1	\$160,877	\$231,411	\$133,934	\$185,961
	Q2	\$197,213	\$204,914	\$337,813	\$213,507
	Q3	\$302,678	\$185,201	\$283,806	\$206,512
	Q4	\$297,208	\$226,983	\$214,845	\$230,291
2010	Q1	\$180,609	\$180,123	\$273,943	\$251,145
	Q2	\$195,785	\$224,882	\$251,391	\$195,976
	Q3	\$116,613	\$50,363	\$194,601	\$102,731

LAST()

\$160,877	6
\$197,213	5
\$302,678	4
\$297,208	3
\$180,609	2
\$195,785	1
\$116,613	0

示例

当前行索引为 3（共 7 行）时，LAST() = 4。

LEFT(string, number)

返回字符串最左侧一定数量的字符。

示例

```
LEFT("Matador", 4) = "Mata"
```

LEN(string)

返回字符串长度。

示例

```
LEN("Matador") = 7
```

LN(number)

返回数字的自然对数。如果数字小于或等于 0，则返回 Null。

LOG(number [, base])

返回数字以给定底数为底的对数。如果省略了底数值，则使用底数 10。

LOG2(数字)

注意：仅在连接到 Google BigQuery 时才受支持

返回数字的对数底 2。

示例

```
LOG2(16) = '4.00'
```

LOOKUP(expression, [offset])

返回目标行（指定为与当前行的相对偏移）中表达式的值。使用 FIRST() + n 和 LAST() - n 作为相对于分区中第一行/最后一行的目标偏移量定义的一部分。如果省略了 offset，则可以在字段菜单上设置要比较的行。如果无法确定目标行，则此函数返回 NULL。

下面的视图显示每季度销售额。当在 Date 分区中计算 LOOKUP (SUM(Sales), 2) 时，每行都会显示接下来 2 个季度的销售额值。

Year of Order Date	Quarter of Order Date	Region			
		Central	East	South	West
2009	Q1	\$160,877	\$231,411	\$133,934	\$185,961
	Q2	\$197,213	\$204,914	\$337,813	\$213,507
	Q3	\$302,678	\$165,201	\$283,806	\$206,512
	Q4	\$297,208	\$226,983	\$214,845	\$230,291
2010	Q1	\$180,609	\$180,123	\$273,943	\$251,145
	Q2	\$195,785	\$224,882	\$251,391	\$195,976
	Q3	\$116,613	\$50,363	\$194,601	\$102,731

Year of Order Date	Quarter of Order Date	Region			
		Central	East	South	West
2009	Q1	\$302,678	\$165,201	\$283,806	\$206,512
	Q2	\$297,208	\$226,983	\$214,845	\$230,291
	Q3	\$180,609	\$180,123	\$273,943	\$251,145
	Q4	\$195,785	\$224,882	\$251,391	\$195,976
2010	Q1	\$116,613	\$50,363	\$194,601	\$102,731
	Q2				
	Q3				

示例

LOOKUP(SUM([Profit]), FIRST()+2) 计算分区第三行中的 SUM(Profit)。

## LOWER(string)

返回 string，其所有字符为小写。

### 示例

```
LOWER("ProductVersion") = "productversion"
```

## LTRIM(string)

返回移除了所有前导空格的字符串。

### 示例

```
LTRIM(" Matador ") = "Matador "
```

## LTRIM\_THIS(字符串, 字符串)

*注意：* 仅在连接到 Google BigQuery 时才受支持

返回第一个字符串（移除了在前导位置出现的任何第二个字符串）。

### 示例

```
LTRIM_THIS('[-Sales-]', '[-]') = 'Sales-'
```

[返回顶部](#)

---

## MAKEDATE(year, month, day)

返回一个依据指定年份、月份和日期构造的日期值。

可用于 Tableau 数据提取。检查在其他数据源中的可用性。

### 示例

```
MAKEDATE(2004, 4, 15) = #April 15, 2004#
```

## MAKEDATETIME(date, time)

返回合并了 date 和 time 的 datetime。日期可以是 date、datetime 或 string 类型。时间必须是 datetime。此功能仅适用于与 MySQL 兼容的连接（对于 Tableau，除 MySQL 之外，还有 Amazon Aurora 和 Amazon Aurora）。

### 示例

```
MAKEDATETIME("1899-12-30", #07:59:00#) = #12/30/1899 7:59:00 AM#
```

```
MAKEDATETIME([Date], [Time]) = #1/1/2001 6:00:00 AM#
```

## MAKETIME(hour, minute, second)

返回一个依据指定小时、分钟和秒构造的日期值。

可用于 Tableau 数据提取。检查在其他数据源中的可用性。

### 示例

```
MAKETIME(14, 52, 40) = #14:52:40#
```

## MAX(a, b)

返回 a 和 b（必须为相同类型）中的较大值。此函数常用于比较数字，但也对字符串有效。对于字符串，MAX 查找数据库为该列定义的排序序列中的最高值。如果任一参数为 Null，则返回 Null。

### 示例

```
MAX ("Apple", "Banana") = "Banana"
```

## MAX(expression) 或 MAX(expr1, expr2)

通常应用于数字，不过也适用于日期。返回 a 和 b 中的较大值（a 和 b 必须为相同类型）。如果任一参数为 Null，则返回 Null。

### 示例

```
MAX(#2004-01-01# , #2004-03-01#) = 2004-03-01 12:00:00 AM
```

```
MAX([ShipDate1], [ShipDate2])
```

## MAX(number, number)

返回两个参数（必须为相同类型）中的较大值。如果任一参数为 Null，则返回 Null。MAX 也可应用于聚合计算中的单个字段。

### 示例

```
MAX(4, 7)
```

```
MAX(Sales, Profit)
```

```
MAX([First Name], [Last Name])
```

## MEDIAN(expression)

返回表达式在所有记录中的中位数。中位数只能用于数字字段。会忽略 Null 值。此函数不适用于在 Tableau Desktop 8.2 版之前创建或使用旧版连接的工作簿。它也不适用于使用以下任何数据源的连接：

- 访问
- Amazon Redshift
- Cloudera Hadoop

- IBM DB2
- IBM PDA (Netezza)
- Microsoft SQL Server
- MySQL
- SAP HANA
- Teradata
- Vertica

对于其他数据源类型，可以将数据提取到数据提取文件以使用此函数。请参见[提取数据 \(extracting\\_data.htm\)](#)。

## MID(string, start, [length])

返回从索引位置 `start` 开始的字符串。字符串中第一个字符的位置为 1。如果添加了可选参数 `length`，则返回的字符串仅包含该数量的字符。

### 示例

```
MID("Calculation", 2) = "alculation"
```

```
MID("Calculation", 2, 5) = "alcul"
```

## MIN(a, b)

返回 `a` 和 `b`（必须为相同类型）中的较小值。此函数常用于比较数字，但也对字符串有效。对于字符串，`MIN` 查找排序序列中的最低值。如果任一参数为 `Null`，则返回 `Null`。

### 示例

```
MIN ("Apple","Banana") = "Apple"
```

## MIN(expression) or MIN(expr1, expr2)

通常应用于数字，不过也适用于日期。返回 `a` 和 `b` 中的较小值（`a` 和 `b` 必须为相同类型）。如果任一参数为 `Null`，则返回 `Null`。

### 示例

```
MIN(#2004-01-01# ,#2004-03-01#) = 2004-01-01 12:00:00 AM
```

```
MIN([ShipDate1], [ShipDate2])
```

## MIN(number, number)

返回两个参数（必须为相同类型）中的较小值。如果任一参数为 `Null`，则返回 `Null`。`MIN` 也可应用于聚合计算中的单个字段。

### 示例

```
MIN(4,7)
```

```
MIN(Sales,Profit)
```

```
MIN([First Name],[Last Name])
```

## MONTH(date)

以整数的形式返回给定日期的月份。

### 示例

```
MONTH(#2004-04-15#) = 4
```

[返回顶部](#)

## NOW()

返回当前日期和时间。

返回值因连接的特性而异：

- 对于实时、未发布的连接，NOW 返回数据源服务器时间。
- 对于实时、已发布的连接，NOW 返回数据源服务器时间。
- 对于未发布的数据提取，NOW 返回本地系统时间。
- 对于发布的数据提取，NOW 返回 Tableau Server 数据引擎的本地时间。如果在不同时区中有多台工作计算机，这可能会产生不一致的结果。

### 示例

```
NOW() = 2004-04-15 1:08:21 PM
```

[返回顶部](#)

## PARSE\_URL(字符串, url\_part)

*注意：仅在连接到Hadoop Hive 和Cloudera Impala 时才受支持。*

返回给定 URL 字符串的组成部分（由 url\_part 定义）。有效的 url\_part 值包括：'HOST'、'PATH'、'QUERY'、'REF'、'PROTOCOL'、'AUTHORITY'、'FILE' 和 'USERINFO'。

### 示例

```
PARSE_URL('http://www.tableau.com', 'HOST') = 'www.tableau.com'
```

## PARSE\_URL\_QUERY(字符串, 密钥)

*注意：仅在连接到Hadoop Hive 和Cloudera Impala 时才受支持。*

返回给定 URL 字符串中的指定查询参数的值。查询参数由密钥定义。

### 示例

```
PARSE_URL_QUERY('http://www.tableau.com?page=1&cat=4', 'page') = '1'
```

## PERCENTILE(expression, number)

从给定表达式返回与指定数字对应的百分位处的值。数字必须介于 0 到 1 之间（含 0 和 1），例如 0.66，并且必须是数值常量。

此函数可用于以下数据源。

- 非旧版 Microsoft Excel 和文本文件连接。
- 数据提取和纯数据提取数据源类型（例如 Google Analytics、OData 或 Salesforce）。
- Sybase IQ 15.1 及更高版本的数据源。
- Oracle 10 及更高版本的数据源。
- Cloudera Hive 和 Hortonworks Hadoop Hive 数据源。
- EXASOL 4.2 及更高版本的数据源。

对于其他数据源类型，可以将数据提取到数据提取文件以使用此函数。请参见[提取数据 \(extracting\\_data.htm\)](#)。

## PI()

返回数字常量 pi: 3.14159。

## POWER(number, power)

计算数字的指定次幂。

### 示例

```
POWER(5,2) = 5^2 = 25
```

```
POWER(Temperature, 2)
```

也可以使用 ^ 符号：

```
5^2 = POWER(5,2) = 25
```

## PREVIOUS\_VALUE(expression)

返回此计算在上一行中的值。如果当前行是分区的第一行，则返回给定表达式。

### 示例

```
SUM([Profit]) * PREVIOUS_VALUE(1) 计算 SUM(Profit) 的运行产品。
```

[返回顶部](#)

## RADIANS(number)

将给定数字从度数转换为弧度。



示例

```
RADIANS(180) = 3.14159
```

RANK(expression, ['asc' | 'desc'])

返回分区中当前行的标准竞争排名。为相同的值分配相同的排名。使用可选的 'asc' | 'desc' 参数指定升序或降序顺序。默认为降序。

利用此函数，将对值集 (6, 9, 9, 14) 进行排名 (4, 2, 2, 1)。

在排名函数中，会忽略 Null。它们不进行编号，且不计入百分位排名计算的总记录数中。

有关不同排名选项的信息，请参见[排名计算 \(calculations\\_tablecalculations\\_definebasic\\_runningtotal.htm#Rank\)](#)。

示例

下图显示对一组值执行各种排名函数（RANK、RANK\_DENSE、RANK\_MODIFIED、RANK\_PERCENTILE 和 RANK\_UNIQUE）的效果。数据集包含 14 名学生（StudentA 到 StudentN）的相关信息；“Age”列显示每个学生的当前年龄（所有学生都介于 17 岁和 20 岁之间）。其余的列会显示每个排名函数对年龄值集的影响，并始终假定函数的默认顺序（升序或降序）。

Student	Age	RANKofAge	RANK_DENSEofAge	RANK_MODIFIEDofAge	RANK_PERCENTILEofAge	RANK_UNIQUEofAge
StudentA	19	4	2	7	79%	4
StudentB	18	8	3	12	50%	8
StudentC	19	4	2	7	79%	5
StudentD	18	8	3	12	50%	9
StudentE	17	13	4	14	14%	13
StudentF	18	8	3	12	50%	10
StudentG	19	4	2	7	79%	6
StudentH	20	1	1	3	100%	1
StudentI	19	4	2	7	79%	7
StudentJ	20	1	1	3	100%	2
StudentK	20	1	1	3	100%	3
StudentL	17	13	4	14	14%	14
StudentM	18	8	3	12	50%	11
StudentN	18	8	3	12	50%	12

RANK\_DENSE(expression, ['asc' | 'desc'])

返回分区中当前行的密集排名。为相同的值分配相同的排名，但不会向数字序列中插入间距。使用可选的 'asc' | 'desc' 参数指定升序或降序顺序。默认为降序。

利用此函数，将对值集 (6, 9, 9, 14) 进行排名 (3, 2, 2, 1)。

在排名函数中，会忽略 Null。它们不进行编号，且不计入百分位排名计算的总记录数中。

有关不同排名选项的信息，请参见[排名计算 \(calculations\\_tablecalculations\\_definebasic\\_runningtotal.htm#Rank\)](#)。

RANK\_MODIFIED(expression, ['asc' | 'desc'])

返回分区中当前行的调整后竞争排名。为相同的值分配相同的排名。使用可选的 'asc' | 'desc' 参数指定升序或降序顺序。默认为降序。

利用此函数，将对值集 (6, 9, 9, 14) 进行排名 (4, 3, 3, 1)。

在排名函数中，会忽略 Null。它们不进行编号，且不计入百分位排名计算的总记录数中。

有关不同排名选项的信息，请参见[排名计算 \(calculations\\_tablecalculations\\_definebasic\\_runningtotal.htm#Rank\)](#)。

## RANK\_PERCENTILE(expression, ['asc' | 'desc'])

返回分区中当前行的百分位排名。使用可选的 'asc' | 'desc' 参数指定升序或降序顺序。默认为升序。

利用此函数，将对值集 (6, 9, 9, 14) 进行排名 (0.25, 0.75, 0.75, 1.00)。

在排名函数中，会忽略 Null。它们不进行编号，且不计入百分位排名计算的总记录数中。

有关不同排名选项的信息，请参见[排名计算 \(calculations\\_tablecalculations\\_definebasic\\_runningtotal.htm#Rank\)](#)。

## RANK\_UNIQUE(expression, ['asc' | 'desc'])

返回分区中当前行的唯一排名。为相同的值分配相同的排名。使用可选的 'asc' | 'desc' 参数指定升序或降序顺序。默认为降序。

利用此函数，将对值集 (6, 9, 9, 14) 进行排名 (4, 2, 3, 1)。

在排名函数中，会忽略 Null。它们不进行编号，且不计入百分位排名计算的总记录数中。

有关不同排名选项的信息，请参见[排名计算 \(calculations\\_tablecalculations\\_definebasic\\_runningtotal.htm#Rank\)](#)。

## RAWSQL\_BOOL("sql\_expr", [arg1], ...[argN])

从给定 SQL 表达式返回布尔结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。

### 示例

在示例中，%1 等于 [Sales]，%2 等于 [Profit]。

```
RAWSQL_BOOL("IIF( %1 > %2, True, False)", [Sales], [Profit])
```

## RAWSQL\_DATE("sql\_expr", [arg1], ...[argN])

从给定 SQL 表达式返回日期结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。

### 示例

在下例中，%1 等于 [Order Date]。

```
RAWSQL_DATE("%1", [Order Date])
```

## RAWSQL\_DATETIME("sql\_expr", [arg1], ...[argN])

从给定 SQL 表达式返回日期和时间结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Delivery Date]。

### 示例

```
RAWSQL_DATETIME("MIN(%1)", [Delivery Date])
```

## RAWSQL\_INT("sql\_expr", [arg1], ...[argN])

从给定 SQL 表达式返回整数结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Sales]。

#### 示例

```
RAWSQL_INT("500 + %1", [Sales])
```

### RAWSQL\_REAL("sql\_expr", [arg1], ...[argN])

从直接传递给基础数据库的给定 SQL 表达式返回数字结果。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Sales]

#### 示例

```
RAWSQL_REAL("-123.98 * %1", [Sales])
```

### RAWSQL\_SPATIAL

从直接传递给基础数据源的给定 SQL 表达式返回空间数据。在 SQL 表达式中将 %n 用作数据库值的替换语法。

#### 示例

在本例中，%1 等于 [Geometry]。

```
RAWSQL_SPATIAL("%1", [Geometry])
```

### RAWSQL\_STR("sql\_expr", [arg1], ...[argN])

从直接传递给基础数据库的给定 SQL 表达式返回字符串。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Customer Name]。

#### 示例

```
RAWSQL_STR("%1", [Customer Name])
```

### RAWSQLAGG\_BOOL("sql\_expr", [arg1], ...[argN])

从给定聚合 SQL 表达式返回布尔结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。

#### 示例

在示例中，%1 等于 [Sales]，%2 等于 [Profit]。

```
RAWSQLAGG_BOOL("SUM( %1) > SUM( %2)", [Sales], [Profit])
```

### RAWSQLAGG\_DATE("sql\_expr", [arg1], ...[argN])

从给定聚合 SQL 表达式返回日期结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Order Date]。

### 示例

```
RAWSQLAGG_DATE("MAX(%1)", [Order Date])
```

## RAWSQLAGG\_DATETIME("sql\_expr", [arg1], ...[argN])

从给定聚合 SQL 表达式返回日期和时间结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Delivery Date]。

### 示例

```
RAWSQLAGG_DATETIME("MIN(%1)", [Delivery Date])
```

## RAWSQLAGG\_INT("sql\_expr", [arg1,] ...[argN])

从给定聚合 SQL 表达式返回整数结果。SQL 表达式直接传递给基础数据库。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Sales]。

### 示例

```
RAWSQLAGG_INT("500 + SUM(%1)", [Sales])
```

## RAWSQLAGG\_REAL("sql\_expr", [arg1,] ...[argN])

从直接传递给基础数据库的给定聚合 SQL 表达式返回数字结果。在 SQL 表达式中将 %n 用作数据库值的替换语法。在下例中，%1 等于 [Sales]

### 示例

```
RAWSQLAGG_REAL("SUM( %1)", [Sales])
```

## RAWSQLAGG\_STR("sql\_expr", [arg1,] ...[argN])

从直接传递给基础数据库的给定聚合 SQL 表达式返回字符串。在 SQL 表达式中将 %n 用作数据库值的替换语法。在此示例中，%1 等于 [Discount]。

### 示例

```
RAWSQLAGG_STR("AVG(%1)", [Discount])
```

## REGEXP\_REPLACE(字符串, 模式, 替换字符串)

返回给定字符串的副本，其中正则表达式模式被替换字符串取代。此函数可用于文本文件、Hadoop Hive、Google BigQuery、PostgreSQL、Tableau 数据提取、Microsoft Excel、Salesforce、Vertica、Pivotal Greenplum、Teradata（版本 14.1 及更高版本）、Snowflake 和 Oracle 数据源。



## 示例

```
REGEXP_EXTRACT_NTH('abc 123', '([a-z]+)\s+(\d+)', 2) = '123'
```

## REPLACE(string, substring, replacement)

在 `string` 中搜索 `substring` 并将其替换为 `replacement`。如果未找到 `substring`，则字符串保持不变。

## 示例

```
REPLACE("Version8.5", "8.5", "9.0") = "Version9.0"
```

## RIGHT(string, number)

返回 `string` 中最右侧一定数量的字符。

## 示例

```
RIGHT("Calculation", 4) = "tion"
```

## ROUND(number, [decimals])

将数字舍入为指定位数。`decimals` 参数指定要在最终结果中包含的小数位数精度。如果省略 `decimals`，则 `number` 舍入为最接近的整数。

## 示例

此示例将每个 `Sales` 值舍入为整数：

```
ROUND(Sales)
```

某些数据库（例如 SQL Server）允许指定负 `length`，其中 `-1` 将 `number` 舍入为 10 的倍数，`-2` 舍入为 100 的倍数，依此类推。此功能并不适用于所有数据库。例如，Excel 和 Access 不具备此功能。

## RTRIM(string)

返回移除了所有尾随空格的 `string`。

## 示例

```
RTRIM(" Calculation ") = " Calculation"
```

## RTRIM\_THIS(字符串, 字符串)

*注意：* 仅在连接到 Google BigQuery 时才受支持

返回第一个字符串（移除了在尾随位置出现的任何第二个字符串）。

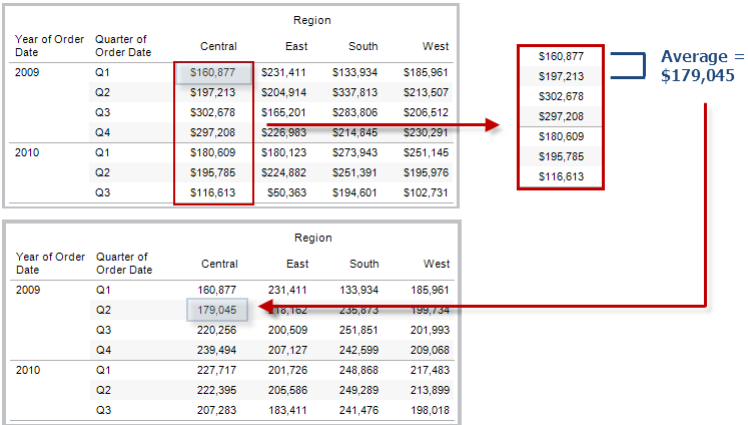
示例

```
RTRIM_THIS('[-Market-'],'-') = '[-Market'
```

RUNNING\_AVG(expression)

返回给定表达式从分区中第一行到当前行的运行平均值。

下面的视图显示每季度销售额。当在 Date 分区中计算 RUNNING\_AVG(SUM([Sales])) 时，结果为每个季度的销售额值的运行平均值。



示例

```
RUNNING_AVG(SUM([Profit])) 计算 SUM(Profit) 的运行平均值。
```

RUNNING\_COUNT(expression)

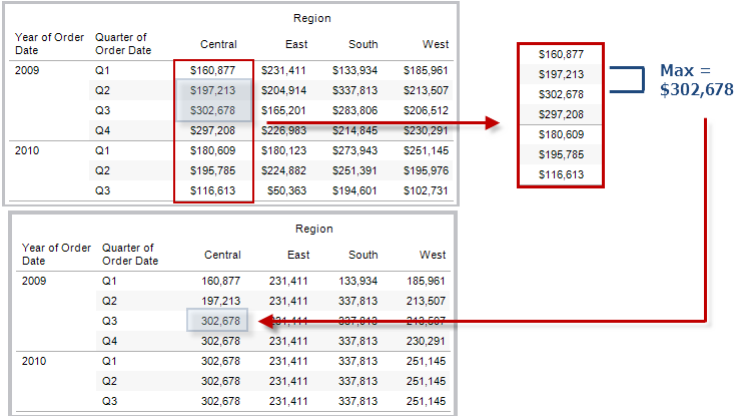
返回给定表达式从分区中第一行到当前行的运行计数。

示例

```
RUNNING_COUNT(SUM([Profit])) 计算 SUM(Profit) 的运行计数。
```

RUNNING\_MAX(expression)

返回给定表达式从分区中第一行到当前行的运行最大值。

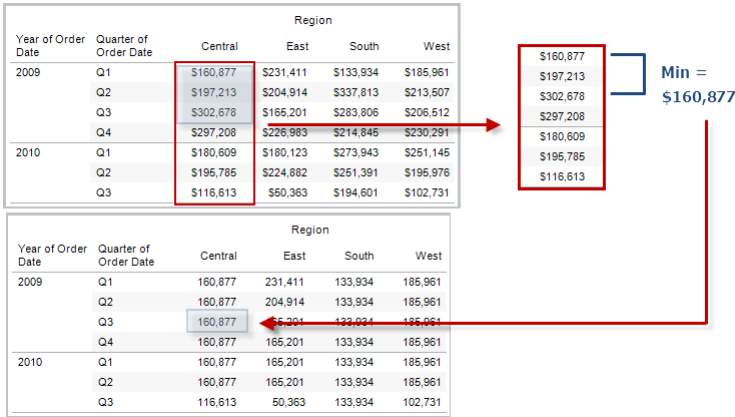


示例

`RUNNING_MAX(SUM([Profit]))` 计算 `SUM(Profit)` 的运行最大值。

`RUNNING_MIN(expression)`

返回给定表达式从分区中第一行到当前行的运行最小值。



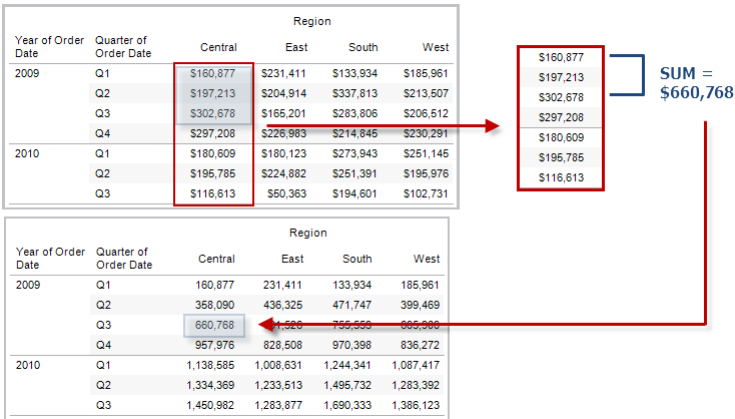
示例

`RUNNING_MIN(SUM([Profit]))` 计算 `SUM(Profit)` 的运行最小值。

`RUNNING_SUM(expression)`

返回给定表达式从分区中第一行到当前行的运行总计。





示例

`RUNNING_SUM(SUM([Profit]))` 计算 `SUM(Profit)` 的运行总计

[返回顶部](#)

SCRIPT\_BOOL

返回指定表达式的布尔结果。表达式直接传递给运行的外部服务实例。

在 R 表达式中，使用 `.argn`（带前导句点）引用参数（`.arg1`、`.arg2` 等）。

在 Python 表达式中，使用 `__argn`（带前导下划线）。

示例

在此 R 示例中，`.arg1` 等于 `SUM([Profit])`：

```
SCRIPT_BOOL("is.finite(.arg1)", SUM([Profit]))
```

对于华盛顿州中的商店 ID，下一示例返回 `True`，否则返回 `False`。此示例可以是标题为 `IsStoreInWA` 的计算字段的定义。

```
SCRIPT_BOOL('grepl(".*_WA", .arg1, perl=TRUE)',ATTR([Store ID]))
```

Python 的命令将采用以下形式：

```
SCRIPT_BOOL("return map(lambda x : x > 0, __arg1)", SUM([Profit]))
```

SCRIPT\_INT

返回指定表达式的整数结果。表达式直接传递给运行的外部服务实例。

在 R 表达式中，使用 `.argn`（带前导句点）引用参数（`.arg1`、`.arg2` 等）。

在 Python 表达式中，使用 `__argn`（带前导下划线）。

示例

在此 R 示例中，`.arg1` 等于 `SUM([Profit])`：

```
SCRIPT_INT("is.finite(.arg1)", SUM([Profit]))
```

在下一示例中，使用 `k-means clustering` 创建三个群集：

```
SCRIPT_INT('result <- kmeans(data.frame(.arg1,.arg2,.arg3,.arg4), 3);result$cluster;', SUM([Petal length]),
SUM([Petal width]),SUM([Sepal length]),SUM([Sepal width]))
```

Python 的命令将采用以下形式：

```
SCRIPT_INT("return map(lambda x : int(x * 5), _arg1)", SUM([Profit]))
```

## SCRIPT\_REAL

返回指定表达式的实数结果。表达式直接传递给运行的外部服务实例。在

在 R 表达式中，使用 `.argn`（带前导句点）引用参数（`.arg1`、`.arg2` 等）

在 Python 表达式中，使用 `__argn`（带前导下划线）。

### 示例

在此 R 示例中，`.arg1` 等于 `SUM([Profit])`：

```
SCRIPT_REAL("is.finite(.arg1)", SUM([Profit]))
```

下一示例将温度值从摄氏值转换为华氏值。

```
SCRIPT_REAL('library(udunits2);ud.convert(.arg1, "celsius", "degree_fahrenheit")',AVG([Temperature]))
```

Python 的命令将采用以下形式：

```
SCRIPT_REAL("return map(lambda x : x * 0.5, _arg1)", SUM([Profit]))
```

## SCRIPT\_STR

返回指定表达式的字符串结果。表达式直接传递给运行的外部服务实例。

在 R 表达式中，使用 `.argn`（带前导句点）引用参数（`.arg1`、`.arg2` 等）

在 Python 表达式中，使用 `__argn`（带前导下划线）。

### 示例

在此 R 示例中，`.arg1` 等于 `SUM([Profit])`：

```
SCRIPT_STR("is.finite(.arg1)", SUM([Profit]))
```

下一示例将从更复杂的字符串（采用原始格式 `13XSL_CA, A13_WA`）中提取州名缩写：

```
SCRIPT_STR('gsub(".*_", "", .arg1)',ATTR([Store ID]))
```

Python 的命令将采用以下形式：

```
SCRIPT_STR("return map(lambda x : x[:2], _arg1)", ATTR([Region]))
```

## SIGN(number)

返回数字的符号：可能的返回值为：在数字为负时为 `-1`，在数字为零时为 `0`，在数字为正时为 `1`。

### 示例

如果 profit 字段的平均值为负值，则

```
SIGN(AVG(Profit)) = -1
```

SIN(number)

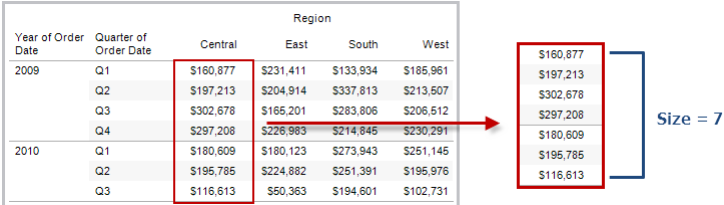
返回角度的正弦。以弧度为单位指定角度。

示例

```
SIN(0) = 1.0  
  
SIN(PI()/4) = 0.707106781186548
```

SIZE()

返回分区中的行数。例如，下面的视图显示每季度销售额。在 Date 分区中有七行，因此 Date 分区的 Size() 为 7。



示例

当前分区包含五行时 SIZE() = 5。

SPACE(number)

返回由指定 number 个重复空格组成的字符串。

示例

```
SPACE(1) = " "
```

SPLIT(string, delimiter, token number)

返回字符串中的一个子字符串，并使用分隔符字符将字符串分为一系列标记。

字符串将被解释为分隔符和标记的交替序列。因此，对于字符串 abc-defgh-i-jkl，分隔符字符为“-”，标记为 abc、defgh、i 和 jkl。将这些标记想像为标记 1 至 4。SPLIT 将返回与标记编号对应的标记。如果标记编号为正，则从字符串的左侧开始计算标记；如果标记编号为负，则从右侧开始计算标记。

示例

```
SPLIT('a-b-c-d', '-', 2) = 'b'  
  
SPLIT('a|b|c|d', '|', -2) = 'c'
```

**注意：**可以为以下数据源类型使用拆分和自定义拆分命令：Tableau 数据提取、Microsoft Excel、文本文件、PDF 文件、Salesforce、OData、Microsoft Azure Market Place、Google Analytics、Vertica、Oracle、MySQL、PostgreSQL、Teradata、Amazon Redshift、Aster 数据、Google Big Query、Cloudera Hadoop Hive、Hortonworks Hive 和 Microsoft SQL Server。

某些数据源在拆分字符串时会有限制。下表显示了哪些数据源支持负标记数量（从右拆分），以及每个数据源允许的拆分数是否有限制。指定负标记编号并且对其他数据源合法的 SPLIT 函数对于这些数据源将返回以下错误：“数据源不支持从右拆分。”

数据源	左/右约束	最大拆分数量	版本限制
Tableau 数据提取	两侧	无限	
Microsoft Excel	两侧	无限	
文本文件	两侧	无限	
Salesforce	两侧	无限	
OData	两侧	无限	
Google Analytics	两侧	无限	
Tableau 数据服务器	两侧	无限	在版本 9.0 中支持。
Vertica	仅左侧	10	
Oracle	仅左侧	10	
MySQL	两侧	10	
PostgreSQL	在版本 9.0 之前仅左侧；版本 9.0 及更高版本中为两侧	10	
Teradata	仅左侧	10	版本 14 及更高版本
Amazon Redshift	仅左侧	10	
Aster Database	仅左侧	10	
Google BigQuery	仅左侧	10	
Hortonworks Hadoop Hive	仅左侧	10	
Cloudera Hadoop	仅左侧	10	版本 2.3.0 中开始支持 Impala。
Microsoft SQL Server	两侧	10	2008 及更高版本

SQRT(number)

返回数字的平方根。

示例

SQRT(25) = 5

SQUARE(number)

返回数字的平方。

示例

```
SQUARE(5) = 25
```

## STARTSWITH(string, substring)

如果 `string` 以 `substring` 开头，则返回 `true`。会忽略前导空格。

### 示例

```
STARTSWITH("Joker", "Jo") = true
```

## STDEV(expression)

基于群体样本返回给定表达式中所有值的统计标准差。

## STDEVP(expression)

基于有偏差群体返回给定表达式中所有值的统计标准差。

## STR(expression)

将其参数转换为字符串。

### 示例

`STR([Age])` 会提取名为 `Age` 的度量中的所有值，并将这些值转换为字符串。

## SUM(expression)

返回表达式中所有值的总计。`SUM` 只能用于数字字段。会忽略 `Null` 值。

[返回顶部](#)

---

## TAN(number)

返回角度的正切。以弧度为单位指定角度。

### 示例

```
TAN(PI ( ) / 4) = 1.0
```

## TIMESTAMP\_TO\_USEC(表达式)

*注意：* 仅在连接到 Google BigQuery 时才受支持

将 `TIMESTAMP` 数据类型转换为 UNIX 时间戳（以微秒为单位）。

### 示例

```
TIMESTAMP_TO_USEC(#2012-10-01 01:02:03#)=1349053323000000
```

TLD(string\_url)

注意：仅在连接到Google BigQuery 时才受支持

在给定 URL 字符串的情况下返回顶层域以及 URL 中的任何国家/地区域。

示例

TLD('http://www.google.com:80/index.html') = '.com'

TLD('http://www.google.co.uk:80/index.html') = '.co.uk'

TODAY( )

返回当前日期。

示例

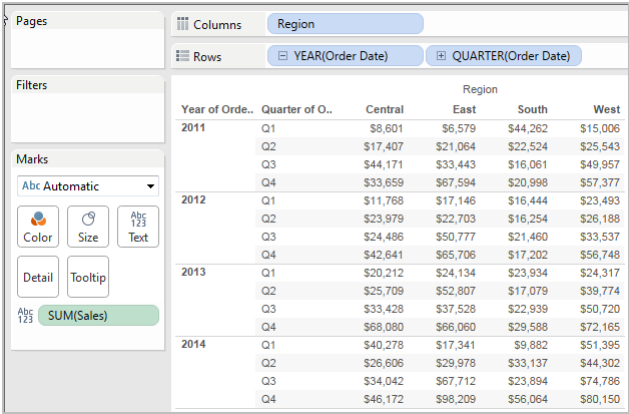
TODAY( ) = 2004-04-15

TOTAL(expression)

返回表计算分区内表达式的总计。

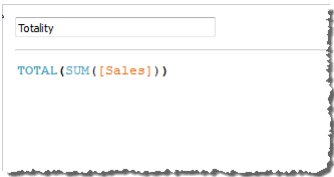
示例

假定您从此视图开始：

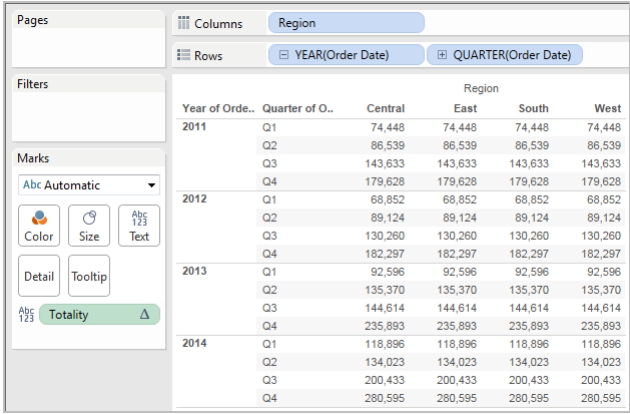


		Region			
Year of Order		Quarter of O..			
		Central	East	South	West
2011	Q1	\$8,601	\$6,579	\$44,262	\$15,006
	Q2	\$17,407	\$21,064	\$22,524	\$25,543
	Q3	\$44,171	\$33,443	\$16,061	\$49,957
	Q4	\$33,659	\$67,594	\$20,998	\$57,377
2012	Q1	\$11,768	\$17,146	\$16,444	\$23,493
	Q2	\$23,979	\$22,703	\$16,254	\$26,188
	Q3	\$24,486	\$50,777	\$21,460	\$33,537
	Q4	\$42,641	\$65,706	\$17,202	\$56,748
2013	Q1	\$20,212	\$24,134	\$23,934	\$24,317
	Q2	\$25,709	\$52,807	\$17,079	\$39,774
	Q3	\$33,428	\$37,528	\$22,939	\$50,720
	Q4	\$68,080	\$66,060	\$29,588	\$72,165
2014	Q1	\$40,278	\$17,341	\$9,882	\$51,395
	Q2	\$26,606	\$29,978	\$33,137	\$44,302
	Q3	\$34,042	\$67,712	\$23,894	\$74,786
	Q4	\$46,172	\$98,209	\$56,064	\$80,150

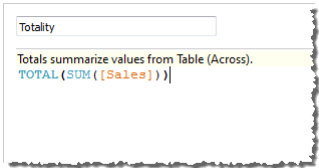
您可以打开计算编辑器并创建名为“总额”的新字段。



然后可以将“总额”拖到“文本”上以替换 SUM(Sales)。您的视图会发生更改，使其总计值基于默认的“计算依据”值：

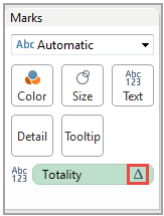


这会产生问题，默认的“计算依据”值是什么？如果在“数据”窗格中右键单击（在 Mac 上按住 Control 单击）“总额”，并选择“编辑”，则会提供一点额外的信息：

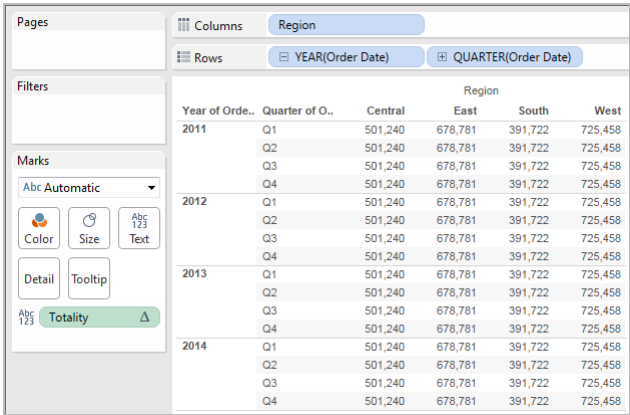


默认的“计算依据”值是“表(横穿)”。结果是，“总额”会汇总每个表行的值。因此，您看到的每一行的值是表原始版本中值的总和。  
原始表内 2011/Q1 行中的值为 \$8601、\$6579、\$44262 和 \$15006。在“总额”替换 **SUM(Sales)** 之后，表中的值都为 \$74,448，这是四个原始值的总和。

请注意，将“总额”拖到“文本”上之后“总额”旁边会出现小三角：



这表明该字段正在使用表计算。您可以右键单击字段并选择“编辑表计算”，以将您的函数重定向到不同的“计算依据”值。例如，可以将其设置为“表(向下)”。在此情况下，您的表将如下所示：



## TRIM(string)

返回移除了前导和尾随空格的字符串。例如，`TRIM(" Calculation ") = "Calculation"`

[返回顶部](#)

## UPPER(string)

返回字符串，其所有字符为大写。

### 示例

```
UPPER("Calculation") = "CALCULATION"
```

## USEC\_TO\_TIMESTAMP(表达式)

*注意：* 仅在连接到 Google BigQuery 时才受支持

将 UNIX 时间戳（以微秒为单位）转换为 TIMESTAMP 数据类型。

### 示例

```
USEC_TO_TIMESTAMP(1349053323000000) = #2012-10-01 01:02:03#
```

## USERDOMAIN()

当当前用户已登录到 Tableau Server 时，返回该用户的域。如果 Tableau Desktop 用户在域上，则返回 Windows 域。否则，此函数返回一个空字符串。

### 示例

```
[Manager]=USERNAME () AND [Domain]=USERDOMAIN ()
```

## USERNAME()

返回当前用户的用户名。当用户已登录时，这是 Tableau Server 或 Tableau Online 用户名；否则为 Tableau Desktop 用户的本地或网络用户名。

### 示例

```
[Manager]=USERNAME ( )
```

如果经理 dhallsten 已登录，则仅当视图中的“Manager”字段为“dhallsten”时，此函数才会返回 True。用作筛选器时，此计算字段可用于创建用户筛选器，该筛选器仅显示与登录到服务器的人员相关的数据。

[返回顶部](#)

## VAR(expression)

基于群体样本返回给定表达式中所有值的统计方差。

## VARP(expression)

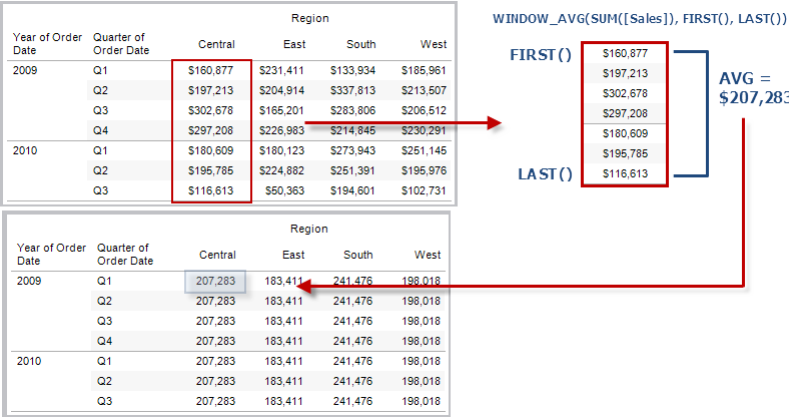
对整个群体返回给定表达式中所有值的统计方差。



## WINDOW\_AVG(expression, [start, end])

返回窗口中表达式的平均值。窗口用与当前行的偏移定义。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

例如，下面的视图显示每季度销售额。Date 分区中的窗口平均值返回所有日期期间的平均销售额。



### 示例

WINDOW\_AVG(SUM([Profit]), FIRST()+1, 0) 计算从第二行到当前行的 SUM(Profit) 平均值。

## WINDOW\_CORR(expression1, expression2, [start, end])

返回窗口内两个表达式的皮尔森相关系数。窗口定义为与当前行的偏移。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了 start 和 end，则使用整个分区。

皮尔森相关系数衡量两个变量之间的线性关系。结果范围为 -1 至 +1（包括 -1 和 +1），其中 1 表示精确的正向线性关系，比如一个变量中的正向更改即表示另一个变量中对应量级的正向更改，0 表示方差之间没有线性关系，而 -1 表示精确的反向关系。

有一个等效的聚合函数：CORR。

### 示例

以下公式返回 **SUM(Profit)** 和 **SUM(Sales)** 从前五行到当前行的皮尔森相关系数。

WINDOW\_CORR(SUM([Profit]), SUM([Sales]), -5, 0)

## WINDOW\_COUNT(expression, [start, end])

返回窗口中表达式的计数。窗口用与当前行的偏移定义。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

### 示例

WINDOW\_COUNT(SUM([Profit]), FIRST()+1, 0) 计算从第二行到当前行的 SUM(Profit) 计数

## WINDOW\_COVAR(expression1, expression2, [start, end])

返回窗口内两个表达式的 *样本协方差*。窗口定义为与当前行的偏移。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了 start 和 end 参数，则窗口为整个分区。

样本协方差使用非空数据点的数量  $n - 1$  来规范化协方差计算，而不是使用总体协方差（及 WINDOW\_COVARP 函数）所使用的  $n$ 。当数据是用于估算较大总体的协方差的随机样本时，则样本协方差是合适的选择。

有一个等效的聚合函数：COVAR。

### 示例

以下公式返回 **SUM(Profit)** 和 **SUM(Sales)** 从前两行到当前行的样本协方差。

```
WINDOW_COVAR(SUM([Profit]), SUM([Sales]), -2, 0)
```

## WINDOW\_COVARP(expression1, expression2, [start, end])

返回窗口内两个表达式的 *总体协方差*。窗口定义为与当前行的偏移。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了 start 和 end，则使用整个分区。

总体协方差等于样本协方差除以  $(n-1)/n$ ，其中  $n$  是非空数据点的总数。如果存在可用于所有相关项的数据，则总体协方差是合适的选择，与之相反，在只有随机项子集的情况下，样本协方差（及 WINDOW\_COVAR 函数）较为适合。

有一个等效的聚合函数：COVARP。请参阅 [Tableau 函数（按字母顺序）](#)。

### 示例

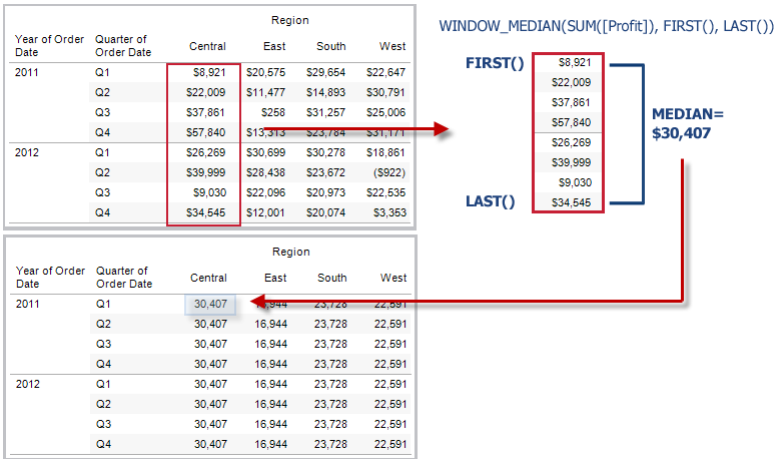
以下公式返回 **SUM(Profit)** 和 **SUM(Sales)** 从前两行到当前行的总体协方差。

```
WINDOW_COVARP(SUM([Profit]), SUM([Sales]), -2, 0)
```

## WINDOW\_MEDIAN(expression, [start, end])

返回窗口中表达式的中值。窗口用与当前行的偏移定义。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

例如，下面的视图显示每季度利润。Date 分区中的窗口中值返回所有日期的中值利润。



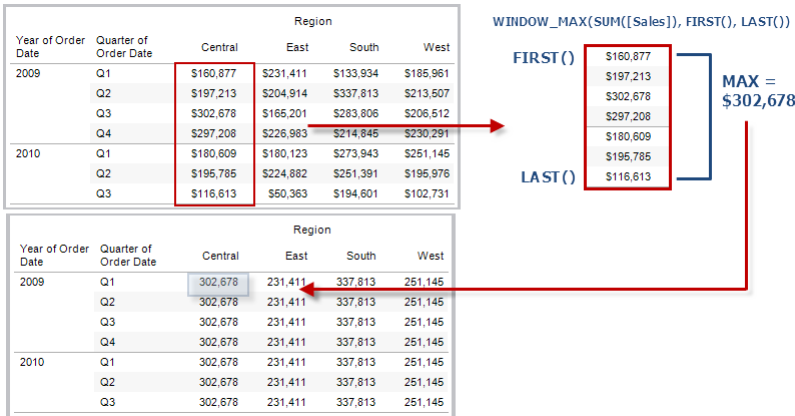
示例

WINDOW\_MEDIAN(SUM([Profit]), FIRST()+1, 0) 计算从第二行到当前行的 SUM(Profit) 中值。

WINDOW\_MAX(expression, [start, end])

返回窗口中表达式的最大值。窗口用与当前行的偏移定义。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

例如，下面的视图显示每季度销售额。Date 分区中的窗口最大值返回所有日期期间的最大销售额。



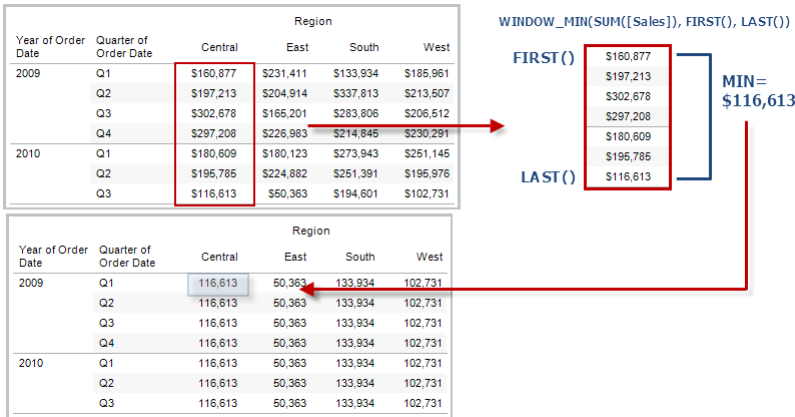
示例

WINDOW\_MAX(SUM([Profit]), FIRST()+1, 0) 计算从第二行到当前行的 SUM(Profit) 最大值。

WINDOW\_MIN(expression, [start, end])

返回窗口中表达式的最小值。窗口用与当前行的偏移定义。使用 FIRST()+n 和 LAST()-n 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

例如，下面的视图显示每季度销售额。Date 分区中的窗口最小值返回所有日期期间的最小销售额。



示例

`WINDOW_MIN(SUM([Profit]), FIRST()+1, 0)` 计算从第二行到当前行的 `SUM(Profit)` 最小值。

WINDOW\_PERCENTILE(expression, number, [start, end])

返回与窗口中指定百分位相对应的值。窗口用与当前行的偏移定义。使用 `FIRST()+n` 和 `LAST()-n` 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

示例

`WINDOW_PERCENTILE(SUM([Profit]), 0.75, -2, 0)` 返回 `SUM(Profit)` 的前面两行到当前行的第 75 个百分位。

WINDOW\_STDEV(expression, [start, end])

返回窗口中表达式的样本标准差。窗口用与当前行的偏移定义。使用 `FIRST()+n` 和 `LAST()-n` 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

示例

`WINDOW_STDEV(SUM([Profit]), FIRST()+1, 0)` 计算从第二行到当前行的 `SUM(Profit)` 标准差。

WINDOW\_STDEVP(expression, [start, end])

返回窗口中表达式的有偏差标准差。窗口用与当前行的偏移定义。使用 `FIRST()+n` 和 `LAST()-n` 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

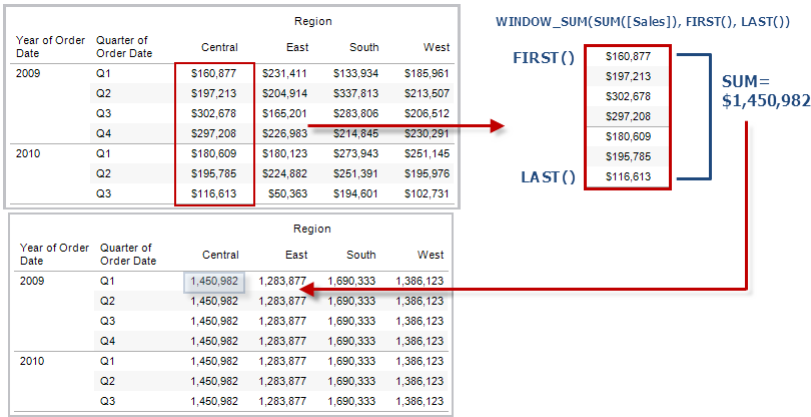
示例

`WINDOW_STDEVP(SUM([Profit]), FIRST()+1, 0)` 计算从第二行到当前行的 `SUM(Profit)` 标准差。

WINDOW\_SUM(expression, [start, end])

返回窗口中表达式的总计。窗口用与当前行的偏移定义。使用 `FIRST()+n` 和 `LAST()-n` 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

例如，下面的视图显示每季度销售额。Date 分区中计算的窗口总计返回所有季度的销售额总计。



示例

`WINDOW_SUM(SUM([Profit]), FIRST()+1, 0)` 计算从第二行到当前行的 `SUM(Profit)` 总和。

WINDOW\_VAR(expression, [start, end])

返回窗口中表达式的样本方差。窗口用与当前行的偏移定义。使用 `FIRST()+n` 和 `LAST()-n` 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

示例

`WINDOW_VAR(SUM([Profit]), FIRST()+1, 0)` 计算从第二行到当前行的 `SUM(Profit)` 方差。

WINDOW\_VARP(expression, [start, end])

返回窗口中表达式的有偏差方差。窗口用与当前行的偏移定义。使用 `FIRST()+n` 和 `LAST()-n` 表示与分区中第一行或最后一行的偏移。如果省略了开头和结尾，则使用整个分区。

示例

`WINDOW_VARP(SUM([Profit]), FIRST()+1, 0)` 计算从第二行到当前行的 `SUM(Profit)` 方差。

[返回顶部](#)

XPATH\_BOOLEAN(XML 字符串, XPath 表达式字符串)

注意：仅在连接到Hadoop Hive时才受支持

如果 XPath 表达式匹配节点或计算为 true，则返回 true。

示例

`XPATH_BOOLEAN('<values> <value id="0">1</value><value id="1">5</value>', 'values/value[@id="1"] = 5') = true`

XPATH\_DOUBLE(XML 字符串, XPath 表达式字符串)

注意：仅在连接到Hadoop Hive时才受支持

返回 XPath 表达式的浮点值。

## 示例

```
XPATH_DOUBLE('<values><value>1.0</value><value>5.5</value> </values>', 'sum(value/*)') = 6.5
```

## XPATH\_FLOAT(XML 字符串, XPath 表达式字符串)

*注意：仅在连接到 Hadoop Hive 时才受支持*

返回 XPath 表达式的浮点值。

## 示例

```
XPATH_FLOAT('<values><value>1.0</value><value>5.5</value> </values>', 'sum(value/*)') = 6.5
```

## XPATH\_INT(XML 字符串, XPath 表达式字符串)

*注意：仅在连接到 Hadoop Hive 时才受支持*

返回 Xpath 表达式的数值；或者，如果 Xpath 表达式无法计算为数字，则返回零。

## 示例

```
XPATH_INT('<values><value>1</value><value>5</value> </values>', 'sum(value/*)') = 6
```

## XPATH\_LONG(XML 字符串, XPath 表达式字符串)

*注意：仅在连接到 Hadoop Hive 时才受支持*

返回 Xpath 表达式的数值；或者，如果 Xpath 表达式无法计算为数字，则返回零。

## 示例

```
XPATH_LONG('<values><value>1</value><value>5</value> </values>', 'sum(value/*)') = 6
```

## XPATH\_SHORT(XML 字符串, XPath 表达式字符串)

*注意：仅在连接到 Hadoop Hive 时才受支持*

返回 Xpath 表达式的数值；或者，如果 Xpath 表达式无法计算为数字，则返回零。

## 示例

```
XPATH_SHORT('<values><value>1</value><value>5</value> </values>', 'sum(value/*)') = 6
```

## XPATH\_STRING(XML 字符串, XPath 表达式字符串)

*注意：仅在连接到 Hadoop Hive 时才受支持*

返回第一个匹配节点的文本。

## 示例

```
XPATH_STRING('<sites ><url domain="org">http://www.w3.org</url> <url domain="com">http://www.tableau.com</url></sites>', 'sites/url[@domain="com"]') = 'http://www.tableau.com'
```

[返回顶部](#)

## YEAR (date)

以整数的形式返回给定日期的年份。

## 示例

```
YEAR (#2004-04-15#) = 2004
```

[返回顶部](#)

## ZN(expression)

如果表达式不为 Null，则返回该表达式，否则返回零。使用此函数可使用零值而不是 Null 值。

## 示例

```
ZN([Profit]) = [Profit]
```

## 想要了解函数的更多信息？

请阅读[函数主题 \(functions.htm\)](#)。

## 另请参见

[Tableau 函数（按类别） \(functions\\_all\\_categories.htm\)](#).

[Tableau 中的函数 \(functions.htm\)](#).