XPath函数

xpath中定义了一组核心函数库。每个函数均使用一个函数原型，该原型给出返回类型，函数名字，以及参数的类型。如参数类型后跟随一个问号，那么该参数是可选的，否则，该参数是必需的。

xpath中函数分4类

* 节点集函数
* 字符串函数
* 布尔函数
* 数字函数

这些函数在表达式和谓词中使用。

**节点集函数（7个）**

* number last()  
  last() 函数返回一个数字，该数字等于表达式求值上下文的上下文大小（上下文的概念参考前篇xpath中路径详解），即给定上下文中的节点数。
* number position()  
  函数返回一个数字，该数字等于表达式求值上下文的上下文的位置，即节点在上下文中的当前位置。如position()=1判断处理的节点是否是节点集合中的第一个节点。
* number count(node-set)  
  count() 函数返回在参数节点集中节点的数目。count(//employee)
* node-set id(object)  
  id() 函数根据元素的唯一标识符来选择元素，返回一个节点集。元素的ID类型属性必须在DTD中声明。
* string local-name(node-set?)  
  返回节点集合中依照文档顺序的第一个节点的扩展名（由节点的本地名和名称空间URI组成）的本地部分。如果参数所指定的节点集为空，或者第一个节点没有扩 展名，则返回一个空字符串。如果参数被省略，默认参数为仅有一个上下文节点作为成员的节点集合。也就是说，如果要得到上下文节点扩展名的本地部分，可以不 传递参数调用local-name()函数。
* string namespace-uri(node-set?)  
  返回节点集合中依照文档顺序的第一个节点的扩展名的名称空间URI。如果参数所指定的节点集为空，或者第一个节点没有扩展名，则返回一个空字符串。如果参 数被省略，默认参数为仅有一个上下文节点作为成员的节点集合。也就是说，如果要得到上下文节点扩展名的名称空间URI，可以不传递参数调用 namespace-uri()函数。
* string name(node-set?)  
  返回节点集合中依照文档顺序的第一个节点的完整限定名（名称空间前缀和本地部分）。如果参数所指定的节点集为空，或者第一个节点没有扩展名，则返回一个空 字符串。如果参数被省略，默认参数为仅有一个上下文节点作为成员的节点集合。也就是说，如果要得到上下文节点扩展名的完整限定名，可以不传递参数调用 name()函数。

**字符串函数（10个）**

* string string(objece?)  
  string()函数按照如下规则将对象转换为字符串。
  + 如果对象object是节点集合，则按照文档顺序返回第一个节点的字符串值。如果节点集为空，则返回一个空字符串。
  + 数字按照如下方式转换成字符串
    - NaN（not a number 不是一个数字）转换为字符串NaN
    - 正零转换为字符串0
    - 负零转换为字符串0
    - 正无穷大转换为Infinity
    - 负无穷大转换为-Infinity
    - 如果数字为整数，则被表示为没有小数点和前导零的十进制格式的数字，如果数字为负整数则在它的前面加一个负号（-）
    - 否则，数字被表示为一个包含了小数点（小数点前后至少各有一位数字）的十进制格式的数字。如果数字是负数，则在它的前面加一个符号（-）
  + 布尔值false转换为字符串false，true转换为true
  + 除了四种基本类型（节点集、布尔、数字、字符串）的对象外，其他对象按照所属类型的方式转换为字符串。
  + 如 果参数被省略，默认参数为仅有一个上下文节点作为成员的节点集合。string（）函数的并不是用于将数字转换为字符串，以便呈现给用户而使用的。如果要 将数字格式化为字符串，可以使用xslt中的format-number()函数或<xsl:number>元素。
* string concat（string，string，string\*)  
  函数将参数中各个字符串进行连接，返回连接后的字符串。concat('n','a','m','e')=name
* boolean starts-with(string，string）  
  如果第一个参数中字符串以第二个参数中字符串开头则true，否则false。如果第二个参数为空字符串，则返回true
* boolean contains（string，string）  
  如果第一个参数中字符串中包含第二个参数中字符串则true，否则false。如果第二个参数为空字符串，则返回true
* string substring-before（string，string）  
  返回在第一个字符串参数中首次出现第二个字符串参数之前的子串，或第一个字符串不包含第二个字符串参数，则返回空字符串。substring-before("1900-0-0","-")=1900 如果第二个参数是空字符串，则返回空字符串。
* string substring-after(string，string)  
  返回在第一个字符串参数中首次出现第二个字符串参数之后的子串，或第一个字符串不包含第二个字符串参数，则返回空字符串。substring-after("1900-0-0","-")=0-0 如果第二个参数是空字符串，则返回第一个字符串。
* string substring(string ,number,number)  
  返回第一个字符串参数从第二个参数所指定的位置开始，以第三个参数为长度的子串。例如 substring("12345",2,3)="234" 如果没指定第三个参数，则返回从第二个参数所指定的位置开始到字符结尾的所有字符串。和程序语言不同的是xpath中字符串的第一个字符的位置为1，而不 是0
* number string-length(string?)  
  string-length()返回字符串中字符的数目。如果参数被省略，则默认是将上下文节点转换为字符串，换句话说，也就是上下文节点的字符串值
* string normalize-space(string?)  
  此函数去除字符串前后出现的空白，以及用一个单独的空格替换字符串中连续出现的空白字符，然后返回经过处理后的字符串。如果参数被省略，则默认是将上下文节点转换为字符串，换句话说，也就是上下文节点的字符串值
* string translate（string，string，string）  
  如果在第一个字符串中存在着第二个字符串中的字符，则将该字符用第三个字符串参数中的与第二个参数中位置顺序相同的字符替换，然后返回替换后的字符串。例 如 translate("bar","abc","ABC")=BAr 如果第二个参数中字符在第三个参数的对应位置没有字符(第二个参数字符串比第三个参数字符串长），那么第一个参数里的那个字符将被删除。 translate("--abc--","abc-","ABC")=ABC 如果在第二个字符串参数中，一个字符出现多次，那么第一个出现的字符决定替换的字符。如果第三个字符串参数比第二个字符串参数长，那么多余字符将被忽略。

**节点测试**

节点测试：从指定轴中选择特定的节点或节点类型  
节点测试的使用

* 指定节点的名称  
  从轴中选择与指定的名称相同的节点。例如：child::name 表示选择上下文节点中的名称为name的子节点
* 使用通配符（\*）  
  使用 \* 表示轴中所有的节点。child::\* 选择上下文节点的所有子元素节点；attribute::\* 表示上下文节点的所有属性。
* node()  
  选择轴中所有类型节点
* text()  
  选择轴中所有文本节点
* comment()  
  轴中所有的注释节点
* processing-instruction()  
  选择轴中所有的处理指令节点。processing-instruction()可以带一个参数，指定处理指令目标的名称。

**谓词**

谓词为一个布尔表达式，用于再次过滤通过轴和节点测试的节点集。针对节点集中 的每一个节点，布尔表达式都将以此节点作为上下文节点来计算，如果布尔表达式结果为真，则在保留节点，否则丢弃此节点。

谓词以[]和其中的布尔表达式组成  
例如 child::employee[position()=1]  
首先得到上下文的所有employee子节点，然后对employee节点集中的每个节点计算postion()=1，找到节点集中位置为1的节点，位置不为1的所有节点都丢弃。

谓 词的处理过程是先计算表达式的值，然后将计算的结果转换为布尔值。如果计算结果为数字，且该数字与上下文位置相等，则结果为真，否则为假。如果计算结果不 是数字，那么结果将被转换为好像调用了boolean()函数后的结果（关于函数请参考xpath中函数介绍）。因此employee[1]和 employee[position()=1]是等价的。

位置路径其实是经过三步过滤最终得到节点集。此节点集包含0个或n个节点。

例如 在xslt中 对<xsl:apply-templates>元素的select属性，可以如下定义其内容  
<xsl:apply-templates select="descendant::monthly\_pay[attribute::mode='cash']/>  
表示上下文节点的后代中，名称为monthly\_pay的所有节点中具有属性mode且属性mode的值为cash的所有节点。

**简写语法**

路径表达式中各种轴的名称会让路径看起来很复杂。为简化路径的书写，xpath为一些常用的轴提供了简写语法

简写 示例 省略child:: employee/name ==child::employee/child::name @代替attribute employee[@sn='e1000']==child::employee[attribute::sn='e1000'] // 代替/descendant-or-self::node() //employee==/descendant-or-self::node()/child::employee即xml文档中所有名称为 employee的节点 (.)代替self::node() self::node() 当前节点 (..) 代替parent::node() parent::node() 当前节点的父节点

其他示例：  
**.//name** == self::node()/descendant::node()/child::name 当前节点下的所有名称为name的节点

**存取函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:node-name(node) | 返回参数节点的节点名称。 |
| fn:nilled(node) | 返回是否拒绝参数节点的布尔值。 |
| fn:data(item.item,...) | 接受项目序列，并返回原子值序列。 |
| * fn:base-uri() * fn:base-uri(node) | 返回当前节点或指定节点的 base-uri 属性的值。 |
| fn:document-uri(node) | 返回指定节点的 document-uri 属性的值。 |

**错误和跟踪函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| * fn:error() * fn:error(error) * fn:error(error,description) * fn:error(error,description,error-object) | 例子：error(fn:QName('http://example.com/test', 'err:toohigh'), 'Error: Price is too high')  结果：向外部处理环境返回 http://example.com/test#toohigh 以及字符串 "Error: Price is too high"。 |
| fn:trace(value,label) | 用于对查询进行 debug。 |

**有关数值的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:number(arg) | 返回参数的数值。参数可以是布尔值、字符串或节点集。  例子：number('100')  结果：100 |
| fn:abs(num) | 返回参数的绝对值。  例子：abs(3.14)  结果：3.14  例子：abs(-3.14)  结果：3.14 |
| fn:ceiling(num) | 返回大于 num 参数的最小整数。  例子：ceiling(3.14)  结果：4 |
| fn:floor(num) | 返回不大于 num 参数的最大整数。  例子：floor(3.14)  结果：3 |
| fn:round(num) | 把 num 参数舍入为最接近的整数。  例子：round(3.14)  结果：3 |
| fn:round-half-to-even() | 例子：round-half-to-even(0.5)  结果：0  例子：round-half-to-even(1.5)  结果：2  例子：round-half-to-even(2.5)  结果：2 |

**有关字符串的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:string(arg) | 返回参数的字符串值。参数可以是数字、逻辑值或节点集。  例子：string(314)  结果："314" |
| fn:codepoints-to-string(int,int,...) | 根据代码点序列返回字符串。  例子：codepoints-to-string(84, 104, 233, 114, 232, 115, 101)  结果：'Thérèse' |
| fn:string-to-codepoints(string) | 根据字符串返回代码点序列。  例子：string-to-codepoints("Thérèse")  结果：84, 104, 233, 114, 232, 115, 101 |
| fn:codepoint-equal(comp1,comp2) | 根据 Unicode 代码点对照，如果 comp1 的值等于 comp2 的值，则返回 true。(http://www.w3.org/2005/02/xpath-functions/collation/codepoint)，否则返回 false。 |
| * fn:compare(comp1,comp2) * fn:compare(comp1,comp2,collation) | 如果 comp1 小于 comp2，则返回 -1。如果 comp1 等于 comp2，则返回 0。如果 comp1 大于 comp2，则返回 1。（根据所用的对照规则）。  例子：compare('ghi', 'ghi')  结果：0 |
| fn:concat(string,string,...) | 返回字符串的拼接。  例子：concat('XPath ','is ','FUN!')  结果：'XPath is FUN!' |
| fn:string-join((string,string,...),sep) | 使用 sep 参数作为分隔符，来返回 string 参数拼接后的字符串。  例子：string-join(('We', 'are', 'having', 'fun!'), ' ')  结果：' We are having fun! '  例子：string-join(('We', 'are', 'having', 'fun!'))  结果：'Wearehavingfun!'  例子：string-join((), 'sep')  结果：'' |
| * fn:substring(string,start,len) * fn:substring(string,start) | 返回从 start 位置开始的指定长度的子字符串。第一个字符的下标是 1。如果省略 len 参数，则返回从位置 start 到字符串末尾的子字符串。  例子：substring('Beatles',1,4)  结果：'Beat'  例子：substring('Beatles',2)  结果：'eatles' |
| * fn:string-length(string) * fn:string-length() | 返回指定字符串的长度。如果没有 string 参数，则返回当前节点的字符串值的长度。  例子：string-length('Beatles')  结果：7 |
| * fn:normalize-space(string) * fn:normalize-space() | 删除指定字符串的开头和结尾的空白，并把内部的所有空白序列替换为一个，然后返回结果。如果没有 string 参数，则处理当前节点。  例子：normalize-space(' The XML ')  结果：'The XML' |
| fn:normalize-unicode() | 执行 Unicode 规格化。 |
| fn:upper-case(string) | 把 string 参数转换为大写。  例子：upper-case('The XML')  结果：'THE XML' |
| fn:lower-case(string) | 把 string 参数转换为小写。  例子：lower-case('The XML')  结果：'the xml' |
| fn:translate(string1,string2,string3) | 把 string1 中的 string2 替换为 string3。  例子：translate('12:30','30','45')  结果：'12:45'  例子：translate('12:30','03','54')  结果：'12:45'  例子：translate('12:30','0123','abcd')  结果：'bc:da' |
| fn:escape-uri(stringURI,esc-res) | 例子：escape-uri("http://example.com/test#car", true())  结果："http%3A%2F%2Fexample.com%2Ftest#car"  例子：escape-uri("http://example.com/test#car", false())  结果："http://example.com/test#car"  例子：escape-uri ("http://example.com/~bébé", false())  结果："http://example.com/~b%C3%A9b%C3%A9" |
| fn:contains(string1,string2) | 如果 string1 包含 string2，则返回 true，否则返回 false。  例子：contains('XML','XM')  结果：true |
| fn:starts-with(string1,string2) | 如果 string1 以 string2 开始，则返回 true，否则返回 false。  例子：starts-with('XML','X')  结果：true |
| fn:ends-with(string1,string2) | 如果 string1 以 string2 结尾，则返回 true，否则返回 false。  例子：ends-with('XML','X')  结果：false |
| fn:substring-before(string1,string2) | 返回 string2 在 string1 中出现之前的子字符串。  例子：substring-before('12/10','/')  结果：'12' |
| fn:substring-after(string1,string2) | 返回 string2 在 string1 中出现之后的子字符串。  例子：substring-after('12/10','/')  结果：'10' |
| fn:matches(string,pattern) | 如果 string 参数匹配指定的模式，则返回 true，否则返回 false。  例子：matches("Merano", "ran")  结果：true |
| fn:replace(string,pattern,replace) | 把指定的模式替换为 replace 参数，并返回结果。  例子：replace("Bella Italia", "l", "\*")  结果：'Be\*\*a Ita\*ia'  例子：replace("Bella Italia", "l", "")  结果：'Bea Itaia' |
| fn:tokenize(string,pattern) | 例子：tokenize("XPath is fun", "\s+")  结果：("XPath", "is", "fun") |

**针对 anyURI 的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:resolve-uri(relative,base) |  |

**关于布尔值的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:boolean(arg) | 返回数字、字符串或节点集的布尔值。 |
| fn:not(arg) | 首先通过 boolean() 函数把参数还原为一个布尔值。如果该布尔值为 false，则返回 true，否则返回 true。  例子：not(true())  结果：false |
| fn:true() | 返回布尔值 true。  例子：true()  结果：true |
| fn:false() | 返回布尔值 false。  例子：false()  结果：false |

**有关持续时间、日期和时间的函数**

日期、时间、持续时间的组件提取函数

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:dateTime(date,time) | 把参数转换为日期和时间。 |
| fn:years-from-duration(datetimedur) | 返回参数值的年份部分的整数，以标准词汇表示法来表示。 |
| fn:months-from-duration(datetimedur) | 返回参数值的月份部分的整数，以标准词汇表示法来表示。 |
| fn:days-from-duration(datetimedur) | 返回参数值的天部分的整数，以标准词汇表示法来表示。 |
| fn:hours-from-duration(datetimedur) | 返回参数值的小时部分的整数，以标准词汇表示法来表示。 |
| fn:minutes-from-duration(datetimedur) | 返回参数值的分钟部分的整数，以标准词汇表示法来表示。 |
| fn:seconds-from-duration(datetimedur) | 返回参数值的分钟部分的十进制数，以标准词汇表示法来表示。 |
| fn:year-from-dateTime(datetime) | 返回参数本地值的年部分的整数。  例子：year-from-dateTime(xs:dateTime("2005-01-10T12:30-04:10"))  结果：2005 |
| fn:month-from-dateTime(datetime) | 返回参数本地值的月部分的整数。  例子：month-from-dateTime(xs:dateTime("2005-01-10T12:30-04:10"))  结果：01 |
| fn:day-from-dateTime(datetime) | 返回参数本地值的天部分的整数。  例子：day-from-dateTime(xs:dateTime("2005-01-10T12:30-04:10"))  结果：10 |
| fn:hours-from-dateTime(datetime) | 返回参数本地值的小时部分的整数。  例子：hours-from-dateTime(xs:dateTime("2005-01-10T12:30-04:10"))  结果：12 |
| fn:minutes-from-dateTime(datetime) | 返回参数本地值的分钟部分的整数。  例子：minutes-from-dateTime(xs:dateTime("2005-01-10T12:30-04:10"))  结果：30 |
| fn:seconds-from-dateTime(datetime) | 返回参数本地值的秒部分的十进制数。  例子：seconds-from-dateTime(xs:dateTime("2005-01-10T12:30:00-04:10"))  结果：0 |
| fn:timezone-from-dateTime(datetime) | 返回参数的时区部分，如果存在。 |
| fn:year-from-date(date) | 返回参数本地值中表示年的整数。  例子：year-from-date(xs:date("2005-04-23"))  结果：2005 |
| fn:month-from-date(date) | 返回参数本地值中表示月的整数。  例子：month-from-date(xs:date("2005-04-23"))  结果：4 |
| fn:day-from-date(date) | 返回参数本地值中表示天的整数。  例子：day-from-date(xs:date("2005-04-23"))  结果：23 |
| fn:timezone-from-date(date) | 返回参数的时区部分，如果存在。 |
| fn:hours-from-time(time) | 返回参数本地值中表示小时部分的整数。  例子：hours-from-time(xs:time("10:22:00"))  结果：10 |
| fn:minutes-from-time(time) | 返回参数本地值中表示分钟部分的整数。  例子：minutes-from-time(xs:time("10:22:00"))  结果：22 |
| fn:seconds-from-time(time) | 返回参数本地值中表示秒部分的整数。  例子：seconds-from-time(xs:time("10:22:00"))  结果：0 |
| fn:timezone-from-time(time) | 返回参数的时区部分，如果存在。 |
| fn:adjust-dateTime-to-timezone(datetime,timezone) | 如果 timezone 参数为空，则返回没有时区的 dateTime。否则返回带有时区的 dateTime。 |
| fn:adjust-date-to-timezone(date,timezone) | 如果 timezone 参数为空，则返回没有时区的 date。否则返回带有时区的 date。 |
| fn:adjust-time-to-timezone(time,timezone) | 如果 timezone 参数为空，则返回没有时区的 time。否则返回带有时区的 time。 |

**与 QNames 相关的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:QName() |  |
| fn:local-name-from-QName() |  |
| fn:namespace-uri-from-QName() |  |
| fn:namespace-uri-for-prefix() |  |
| fn:in-scope-prefixes() |  |
| fn:resolve-QName() |  |

**关于节点的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| * fn:name() * fn:name(nodeset) | 返回当前节点的名称或指定节点集中的第一个节点。 |
| * fn:local-name() * fn:local-name(nodeset) | 返回当前节点的名称或指定节点集中的第一个节点 - 不带有命名空间前缀。 |
| * fn:namespace-uri() * fn:namespace-uri(nodeset) | 返回当前节点或指定节点集中第一个节点的命名空间 URI。 |
| fn:lang(lang) | 如果当前节点的语言匹配指定的语言，则返回 true。  例子：Lang("en") is true for <p xml:lang="en">...</p>  例子：Lang("de") is false for <p xml:lang="en">...</p> |
| * fn:root() * fn:root(node) | 返回当前节点或指定的节点所属的节点树的根节点。通常是文档节点。 |

**有关序列的函数**

**一般性的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:index-of((item,item,...),searchitem) | 返回在项目序列中等于 searchitem 参数的位置。  例子：index-of ((15, 40, 25, 40, 10), 40)  结果：(2, 4)  例子：index-of (("a", "dog", "and", "a", "duck"), "a")  Result (1, 4)  例子：index-of ((15, 40, 25, 40, 10), 18)  结果：() |
| fn:remove((item,item,...),position) | 返回由 item 参数构造的新序列 - 同时删除 position 参数指定的项目。  例子：remove(("ab", "cd", "ef"), 0)  结果：("ab", "cd", "ef")  例子：remove(("ab", "cd", "ef"), 1)  结果：("cd", "ef")  例子：remove(("ab", "cd", "ef"), 4)  结果：("ab", "cd", "ef") |
| fn:empty(item,item,...) | 如果参数值是空序列，则返回 true，否则返回 false。  例子：empty(remove(("ab", "cd"), 1))  结果：false |
| fn:exists(item,item,...) | 如果参数值不是空序列，则返回 true，否则返回 false。  例子：exists(remove(("ab"), 1))  结果：false |
| fn:distinct-values((item,item,...),collation) | 返回唯一不同的值。  例子：distinct-values((1, 2, 3, 1, 2))  结果：(1, 2, 3) |
| fn:insert-before((item,item,...),pos,inserts) | 返回由 item 参数构造的新序列 - 同时在 pos 参数指定位置插入 inserts 参数的值。  例子：insert-before(("ab", "cd"), 0, "gh")  结果：("gh", "ab", "cd")  例子：insert-before(("ab", "cd"), 1, "gh")  结果：("gh", "ab", "cd")  例子：insert-before(("ab", "cd"), 2, "gh")  结果：("ab", "gh", "cd")  例子：insert-before(("ab", "cd"), 5, "gh")  结果：("ab", "cd", "gh") |
| fn:reverse((item,item,...)) | 返回指定的项目的颠倒顺序。  例子：reverse(("ab", "cd", "ef"))  结果：("ef", "cd", "ab")  例子：reverse(("ab"))  结果：("ab") |
| fn:subsequence((item,item,...),start,len) | 返回 start 参数指定的位置返回项目序列，序列的长度由 len 参数指定。第一个项目的位置是 1。  例子：subsequence(($item1, $item2, $item3,...), 3)  结果：($item3, ...)  例子：subsequence(($item1, $item2, $item3, ...), 2, 2)  结果：($item2, $item3) |
| fn:unordered((item,item,...)) | 依据实现决定的顺序来返回项目。 |

**测试序列容量的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:zero-or-one(item,item,...) | 如果参数包含零个或一个项目，则返回参数，否则生成错误。 |
| fn:one-or-more(item,item,...) | 如果参数包含一个或多个项目，则返回参数，否则生成错误。 |
| fn:exactly-one(item,item,...) | 如果参数包含一个项目，则返回参数，否则生成错误。 |

**Equals, Union, Intersection and Except**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:deep-equal(param1,param2,collation) | 如果 param1 和 param2 与彼此相等（deep-equal），则返回 true，否则返回 false。 |

**合计函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:count((item,item,...)) | 返回节点的数量。 |
| fn:avg((arg,arg,...)) | 返回参数值的平均数。例子：avg((1,2,3)) 结果：2 |
| fn:max((arg,arg,...)) | 返回大于其它参数的参数。例子：max((1,2,3)) 结果：3 例子：max(('a', 'k')) 结果：'k' |
| fn:min((arg,arg,...)) | 返回小于其它参数的参数。例子：min((1,2,3)) 结果：1 例子：min(('a', 'k')) 结果：'a' |
| fn:sum(arg,arg,...) | 返回指定节点集中每个节点的数值的总和。 |

**生成序列的函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:id((string,string,...),node) | Returns a sequence of element nodes that have an ID value equal to the value of one or more of the values specified in the string argument |
| fn:idref((string,string,...),node) | Returns a sequence of element or attribute nodes that have an IDREF value equal to the value of one or more of the values specified in the string argument |
| fn:doc(URI) |  |
| fn:doc-available(URI) | 如果 doc() 函数返回文档节点，则返回 true，否则返回 false。 |
| * fn:collection() * fn:collection(string) |  |

**上下文函数**

|  |  |
| --- | --- |
| **名称** | **说明** |
| fn:position() | 返回当前正在被处理的节点的 index 位置。  例子：//book[position()<=3]  结果：选择前三个 book 元素 |
| fn:last() | 返回在被处理的节点列表中的项目数目。  例子：//book[last()]  结果：选择最后一个 book 元素 |
| fn:current-dateTime() | 返回当前的 dateTime（带有时区）。 |
| fn:current-date() | 返回当前的日期（带有时区）。 |
| fn:current-time() | 返回当前的时间（带有时区）。 |
| fn:implicit-timezone() | 返回隐式时区的值。 |
| fn:default-collation() | 返回默认对照的值。 |
| fn:static-base-uri() | 返回 base-uri 的值。 |