HBase 过滤器语言

英文作者：Anirudh Todi
本中文翻译作者：周海汉
日期：2012-8-22
本文永久更新地址：http://abloz.com/2012/08/22/the-hbases-content-query-2.html
本文作者网站：http://abloz.com/
英文网址：https://issues.apache.org/jira/browse/HBASE-4176

Anirudh Todi 在 HBASE-4176 对过 HBase 滤器语言进行了描述。本文基于 Todi 的文档。

目录

通用过滤器语法

“FilterName (argument, argument, ... , argument)”

过滤器名+用括号括起来的并用逗号隔开的参数列表。如果参数是字符串，用单引号引起来。字符串内如果有单引号，则前面必须再加一个单引号。
如果是 boolean, integer 或 comparison operator 如 <, >, !=等，不要用单引号引用。
过滤器名是一个单词的 ASCII，不要有单引号，括号，空格。

复合过滤和操作

包括二元操作 AND/OR 和一元操作 SKIP/WHILE.
操作符必须大写

AND —— 键值必须同时通过两个 Filter
OR —— 键值必须通过两个 Filter 之一
SKIP—— 对某一行，任何键值对没有通过过滤器，则整行忽略。
WHILE—— 对某一行，持续发出键值对，直到某一键值对没有通过过滤器

复合过滤语法：

“(Filter1 AND Filter2) OR (Filter3 AND Filter4)”

1). 优先级

括号()优先级最高
其次是 SKIP 和 WHILE， 优先级一样
AND 第三
OR 优先级最低

示例：
“Filter1 AND Filter2 OR Filter3” 等价于*“(Filter1 AND Filter2) OR Filter3”*
*“Filter1 AND SKIP Filter2 OR Filter3”*等价于*“(Filter1 AND (SKIP Filter2)) OR Filter3”*

比较操作符和比较器(Comparator)

1). 比较操作符

包括

1. LESS (<)
2. LESS_OR_EQUAL (<=)
3. EQUAL (=)
4. NOT_EQUAL (!=)
5. GREATER_OR_EQUAL (>=)
6. GREATER (>)
7. NO_OP (no operation)

Client 应用用符号(<, <=, =, !=, >, >=) 来表达比较操作

2). 比较器

包括

1. **BinaryComparator** – 按字节索引顺序比较指定字节数组，采用 Bytes.compareTo(byte[], byte[])
2. **BinaryPrefixComparator** - 按字节索引顺序比较指定字节数组，仅比较到字节数组的长度。
3. **RegexStringComparator** – 与指定字节数组比较，采用指定正则表达式。仅 *EQUAL 和 NOT_EQUAL 比较操作符对其有效。*
4. **SubStringComparator** – 检测给出子串是否出现在指定的字节数组里。比较大小写不敏感。仅 *EQUAL 和 NOT_EQUAL 比较操作符对其有效。*

比较器语法： ComparatorType:ComparatorValue

比较器对应的类型(ComparatorType)如下：

1. **BinaryComparator** - binary
2. **BinaryPrefixComparator** - binaryprefix
3. **RegexStringComparator** - regexstring
4. **SubStringComparator** - substring

比较器值(ComparatorValue)可以是任何值。

示例

Example1:

>, 'binary:abc' 匹配索引序大于”abc”的所有情况。

Example2:

=, 'binaryprefix:abc' 匹配任何前三个字符是”abc”的情况。

Example3:

!=, 'regexstring:ab*yz' 匹配任何不以 ab 开始且不以 yz 结束的情况。

Example4:

=, 'substring:abc123' 匹配具有子串"abc123"的任何情况。

过滤器字符串示例

- “PrefixFilter (‘Row’) AND PageFilter (1) AND FirstKeyOnlyFilter ()”

  返回所有 key-value 对匹配如下条件

  1) 行的 key-value 有前缀 “Row”

  2) key-value 必须在表的第一行找到

  3) key-value 对必须是行的第一个 key-value 对

- “(RowFilter (=, ‘binary:Row 1’) AND TimeStampsFilter (74689, 89734)) OR ColumnRangeFilter (‘abc’, true, ‘xyz’, false))”

  返回键值对同时匹配下列条件：

  1) key-value 具有键 “Row 1”

  2) key-value 有 timestamp 74689 或 89734 之一

  或匹配

  1) key-value 对必须在列名字典索引序 >= abc 且 < xyz

- “SKIP ValueFilter (0)”

  任何 Value 不是 0，则跳过整行。

单独过滤器语法

1). KeyOnlyFilter

Description: 该过滤器无参，仅返回 Key。

Syntax: KeyOnlyFilter ()

Example: "KeyOnlyFilter ()"

2). FirstKeyOnlyFilter

Description: 该过滤器无参，每行仅返回第一个键值。

Syntax: FirstKeyOnlyFilter ()

Example: "FirstKeyOnlyFilter ()"

3). PrefixFilter

Description: 该过滤器一个参数 —— 行键前缀，返回行键前缀为指定参数的行。

Syntax: PrefixFilter (‘<row_prefix>’)

Example: "PrefixFilter (‘Row’)"

4). ColumnPrefixFilter

Description: 该过滤器有一个参数 ——列名称前缀。

Syntax: ColumnPrefixFilter (‘<column_prefix>’)

Example: "ColumnPrefixFilter (‘Col’)"

5). MultipleColumnPrefixFilter

Description: 一个列名称前缀参数列表，返回匹配任何列前缀之一的键值对。

Syntax: MultipleColumnPrefixFilter (‘<column_prefix>’, ‘<column_prefix>’, …,

‘<column_prefix>’)

Example: "MultipleColumnPrefixFilter (‘Col1’, ‘Col2’)"

6). ColumnCountGetFilter

Description: 一个参数——限制值. 返回表中第一个到达限制列数的列。

Syntax: ColumnCountGetFilter (‘<limit>’)

Example: "ColumnCountGetFilter (4)"

7). PageFilter

Description: 该过滤器有一个参数 —— 页大小，返回表中页大小指定的行数。

Syntax: PageFilter (‘<page_size>’)

Example: "PageFilter (2)"

8). ColumnPaginationFilter

Description: 该过滤器有两个参数 —— 限制值和偏移量。返回偏移量开始限制值个数列。影响所有行。

Syntax: ColumnPaginationFilter (‘<limit>’, ‘<offest>’)

Example: "ColumnPaginationFilter (3, 5)"

9). InclusiveStopFilter

Description: 该过滤器有一个参数 —— 停止扫描的行键。返回所有键值对直到包含停止扫描的行。

Syntax: InclusiveStopFilter (‘<stop_row_key>’)

Example: "InclusiveStopFilter (‘Row2’)”

10). TimeStampsFilter

Description: 该过滤器有一个时间戳参数列表。返回所有匹配时间戳之一的键值对。

Syntax: TimeStampsFilter (<timestamp>, <timestamp>, ... ,<timestamp>)

Example: "TimeStampsFilter (5985489, 48895495, 58489845945)"

11). RowFilter

Description: 该过滤器有一个比较操作符和一个比较器。将比较器和每行用比较操作符比较，如果返回真，则返回该行所有键值对。

Syntax: RowFilter (<compareOp>, ‘<row_comparator>’)

Example: "RowFilter (<=, ‘binary:xyz)"

12). Family Filter

Description: 该过滤器有一个比较操作符和一个比较器。将比较器和每个列族用比较操作符比较，如果返回真，则返回该列族所有键值对。

Syntax: FamilyFilter (<compareOp>, ‘<family_comparator>’)

Example: "FamilyFilter (>=, ‘binaryprefix:FamilyB’)"

13). QualifierFilter

Description: 该过滤器有一个比较操作符和一个比较器。将比较器和每个列(修饰)用比较操作符比较，如果返回真，则返回该列所有键值对。

Syntax: QualifierFilter (<compareOp>, ‘<qualifier_comparator>’)

Example: "QualifierFilter (=, ‘substring:Column1’)"

14). ValueFilter

Description: 该过滤器有一个比较操作符和一个比较器。将比较器和每个值用比较操作符比较，如果返回真，则返回该键值对。

Syntax: ValueFilter (<compareOp>, ‘<value_comparator>’)

Example: "ValueFilter (!=, ‘binary:Value’)"

15). DependentColumnFilter

Description: 该过滤器有两个参数 —— 列族和列修饰。 尝试找到该列所在的每一行，并返回该行具有相同时间戳的全部键值对。如果某一行不包含指定的列，则该行的任何键值对都不返回。
该过滤器还可以有一个可选布尔参数 —— dropDependentColumn. 如果为 true, 从属的列不返回。
该过滤器还可以有两个可选参数 —— 一个比较操作符和一个值比较器，用于列族和修饰的进一步检查。如果从属的列找到，其值还必须通过值检查，然后就是时间戳必须考虑。

Syntax: DependentColumnFilter (‘<family>’, ‘<qualifier>’, <boolean>, <compare operator>, ‘<value comparator’)
DependentColumnFilter (‘<family>’, ‘<qualifier>’, <boolean>)
DependentColumnFilter (‘<family>’, ‘<qualifier>’)

Example: "DependentColumnFilter (‘conf’, ‘blacklist’, false, >=, ‘zebra’)"
"DependentColumnFilter (‘conf’, ‘blacklist’, true)"
"DependentColumnFilter (‘conf’, ‘blacklist’)"

16). SingleColumnValueFilter

Description: 该过滤器有一个列族，一个列修饰，一个比较操作符和一个比较器做参数。如果某行没有找到指定列，该行的所有列都会发出来。如果找到列，且比较操作返回真，该行

的所有列会发出来。如果失败，则该行不发出来。

该过滤器也有两个可选布尔参数——filterIfColumnMissing 和 setLatestVersionOnly

如果 filterIfColumnMissing 标志设为真，如果该行没有指定的列，那么该行的所有列将不发出。缺省值为假。

如果 setLatestVersionOnly 标志设为假，将检查此前的版本。缺省值为真。

该可选参数要么不设，要么两个都设。

Syntax: SingleColumnValueFilter '<family>', '<qualifier>', <compare operator>, '<comparator>', <filterIfColumnMissing_boolean>, <latest_version_boolean>)
Syntax: SingleColumnValueFilter ('<family>', '<qualifier>', <compare operator>, '<comparator>')

Example: "SingleColumnValueFilter ('FamilyA', 'Column1', <=, 'abc', true, false)"
Example: "SingleColumnValueFilter (FamilyA', 'Column1', <=, 'abc')"

17). SingleColumnValueExcludeFilter

Description: 该过滤器具有 SingleColumnValueFilter 一样的参数和行为。然而，如果找到列，且比较操作返回真，该行的所有列会发出来。

Syntax: SingleColumnValueExcludeFilter (<family>, <qualifier>, <compare operators>, <comparator>, <latest_version_boolean>, <filterIfColumnMissing_boolean>)
Syntax: SingleColumnValueExcludeFilter (<family>, <qualifier>, <compare operator> <comparator>)

Example: "SingleColumnValueExcludeFilter ('FamilyA', 'Column1', '<=', 'abc', 'false', 'true')"
Example: "SingleColumnValueExcludeFilter ('FamilyA', 'Column1', '<=', 'abc')"

18). ColumnRangeFilter

Description: 该过滤器用于选择列在 minColumn 和 maxColumn 之间的键。
该过滤器也有两个可选布尔参数用于指示是否包含 minColumn 和 maxColumn 。
如果不指定 minColumn 或 the maxColumn ——可以不传参数。

Syntax: ColumnRangeFilter (‘<minColumn >’, <minColumnInclusive_bool>, ‘<maxColumn>’, <maxColumnInclusive_bool>)

Example: "ColumnRangeFilter (‘abc’, true, ‘xyz’, false)"