ALEIYE

让 | 大 | 数 | 据 | 更 | 简 | 单

Aleiye 检索命令说明文档

北京数介科技有限公司

2016-4

文档版本记录

版本号	主题	主要作者	备注

目录

第1章 Aleiye 检索概述

Aleiye 使用类数据库系统提供了一套完整的查询搜索语言，包括关键字搜索、短语搜索、通配符搜索、字段搜索。

Aleiye 可以进行实时的交互式搜索。利用放大或缩小时间轴快速显示事件趋势、峰值和异常。在搜索结果中点击信息，可快速滤除无用信息，在巨量的数据中找到想要的结果。实时搜索意味着事件和攻击发生的可见性，可实时监测应用，可关联分析高速数据流事件，追踪进行中的交易和在线的应用活动情况。

第2章 Aleiye 检索命令列表

Aleiye 命令分为两种，Aleiye 原语搜索和 AleiyeSQL 搜索，其中原语搜索命令由三个部分组成,每部分由竖线间隔。AleiyeSQL 搜索游两个部分组成，每部分由竖线间隔。

语法：*基础搜索|自定义字段|报表命令*

如： *:* |eval alias=parserLong(field) |top alias

语法：*基础搜索|SQL 命令*

如：*:*|select * from demo

其组成规则如下：

- 所有命令必须以基础搜索开始；
- 自定义报表字段可以有任意个，也可以没有；
- 报表命令可以没有，但是如果存在报表命令，语句必须以报表命令结束并且报表命令只能由一个。没有报表只有自定义报表，自定义字段部分没有价值；
- SQL 命令只能和基础搜索组合,不能与自定义字段和报表命令组合使用;

分类	命令	描述
基础搜索	“Aleiye”	单一关键字搜索，可以一个关键字进行搜索。
	“hello Aleiye”	短语搜索，可以对某一短语进行搜索。
	字段名:“字段值”	针对字段和字段值进行搜索。
	?	通配符将查找所有满足通过一个字符替换后符合条件的文档。
	*	通配符将查询 0 个或者多个字符替换后符合条件的。
	*	字段模糊检索,可查询字段 A 值或 B 值。
	missing	查询某字段无值或为空。
	exists	查询某字段值不为空。
	/ /	正则匹配开始和结束的标识。

	~	相近搜索，可以对相似字符进行查询。
	to	范围搜索。
	OR	或操作符，检索条件之间使用 OR 链接 那么检索结果将是符合两个条件的并集。
	AND	与操作符，规定必须所有的检索条件都出现才能满足查询条件。
	NOT	非操作符，规定查询的文档必须不包含 NOT 之后的检索条件。
	\	特殊字符转译。
	（）	使用圆括号来组合语句形成子查询。
自定义字段 eval	Todate	将字符串类型的字段转换成时间戳。
	ToNum	将字段转换成数值型。
	substr	**对字符串进行截取。**
	concat	**对多个字段数据进行拼接**
	trim	**对字段数据首尾进行去空格或制表符，或只去左侧、右侧空格。**
	upper	**将字段类数据转换为大写显示。**
	lower	**将字段类数据转换为小写显示。**
	split	**将字符串类型数据按给定的正则表达式来切分数据，并指写切分后的第几个数据。**
	cal	对字段进行四则混合运算。
	scope	筛选出 field 字段值在 fromNum 与 toNum 之间的记录。
	regex	对字段 field 按照 regexStr 进行正则匹配。
	datebet	求两个日期的间隔，即 to-from 以 span 为单位的间隔。
报表命令	report	此命令用来制作图表，可通过不同维度展示数据，图表中维度用 X 轴来展现。

	top	此命令用来对信息进行统计数量的图表。
	datereport	此命令用来制作以时间为 X 轴的时间趋势图表。
	Rangereport	此命令用来对数值型字段进行分组统计。

第3章 原语检索命令

3.1 单一关键词搜索

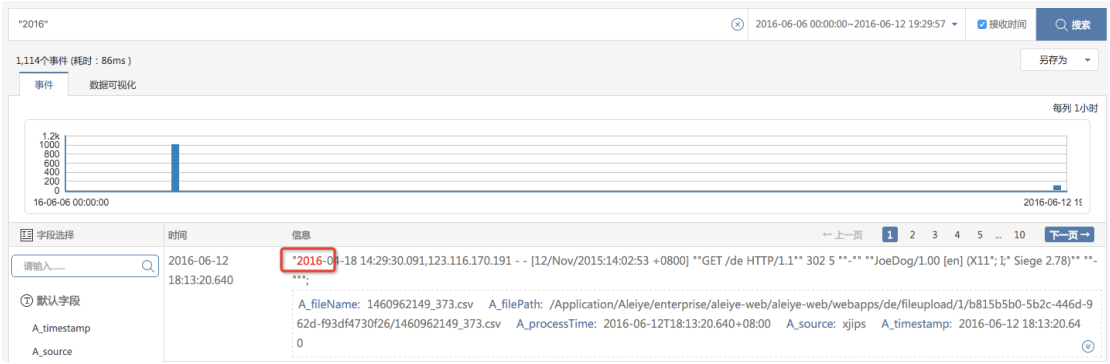
说明：对单一关键字进行检索

语法：” Character”

参数说明：Character:字符 ，”“引号代表精准拼配，如果不加则代表模糊匹配。

应用场景：“2016”

返回结果：



3.2 短语检索

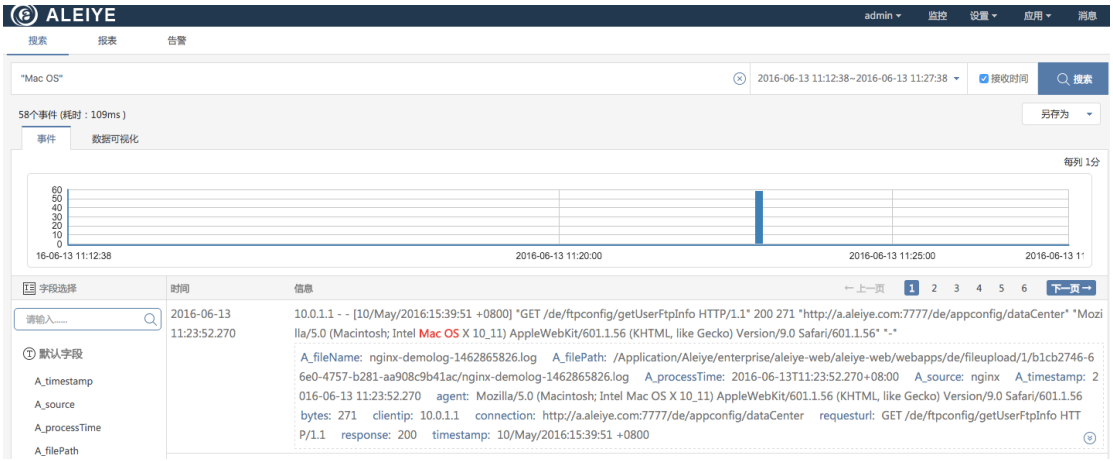
说明：对短语进行检索

语法：” Character Character”

参数说明：Character:字符，””代表精准匹配，如果不加则代表模糊匹配。

应用场景：”Mac OS”

返回结果：



3.3 字段检索

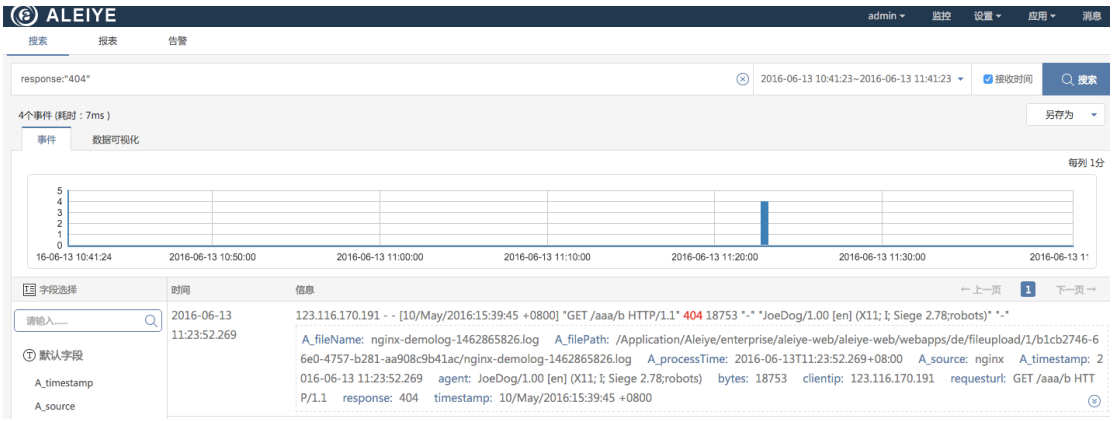
说明：数据根据配置，可将数据且分为多个字段，可以根据字段名和值进行检索。

语法：field:”Vilue”

参数说明：field：字段、Vlilue：字段值

应用场景：response：”404”

返回结果：



3.4 ？

说明：通配符将查找所有满足通过一个字符替换后符合条件的文档。

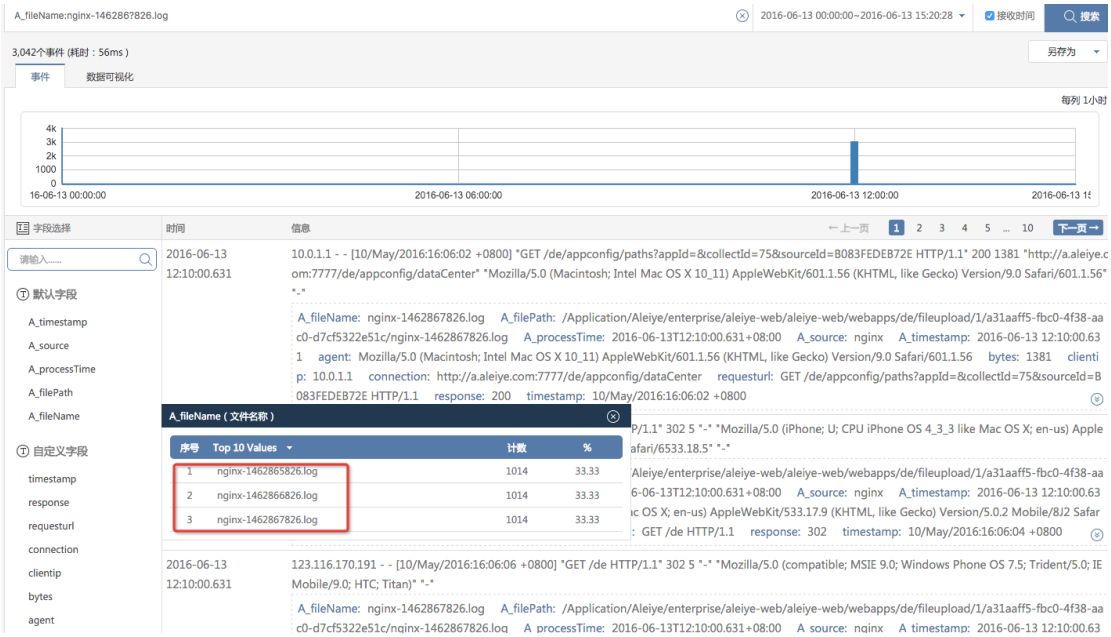
语法：Character?

参数说明：Character：字符，？可以放在字符中的任何位置。当使用字段搜

索时，去掉字段值中的引号。

应用场景：A_source: nginx-146286?826.log，nginx-1462865826.log、nginx-1462866826.log、nginx-1462867826.log 三个文件名称成，通过？快速搜索出三个文件名称。

返回结果：



3.5 *

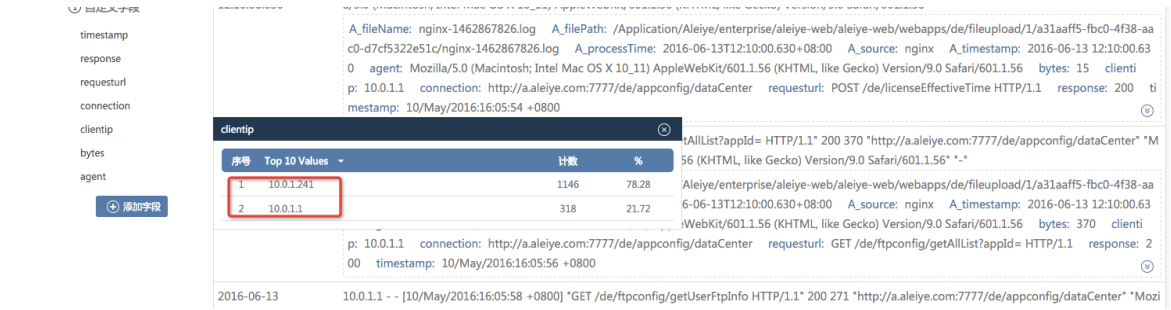
说明：通配符将查询 0 个或者多个字符替换后符合条件的。

语法：Character*

参数说明：Character：字符，*可以放在字符中的任何位置。当使用字段搜索时，去掉字段值中的引号。

应用场景：clientip:10.0.1.*，通过*，搜索出前缀为 10.0.1 的所有 IP 地址。

返回结果：



3.6 *

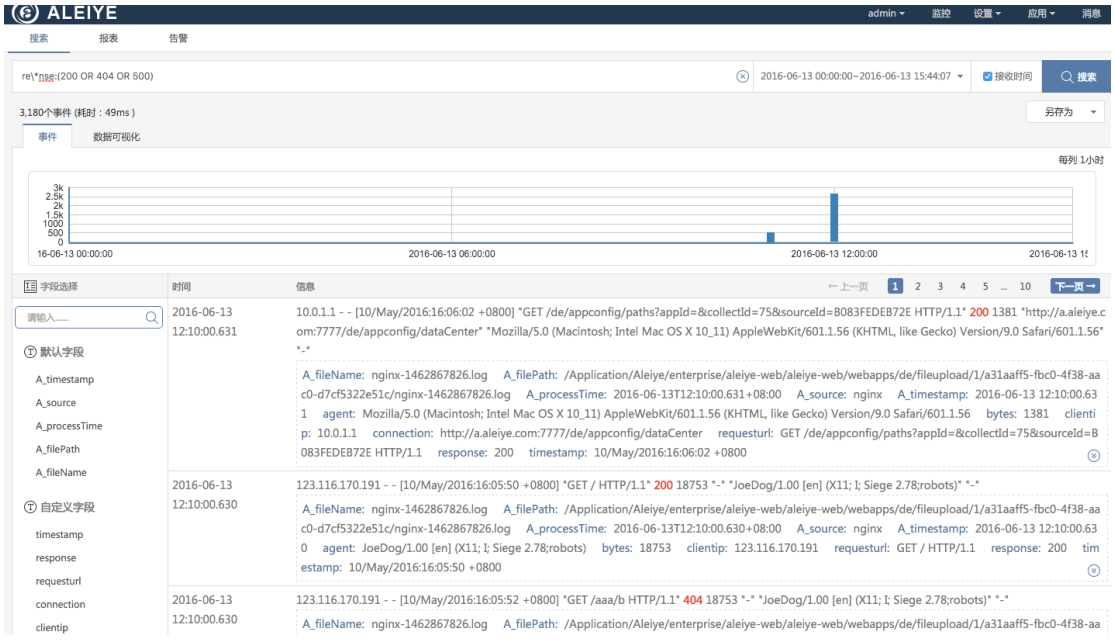
说明：段模糊检索，当查询任务字段包括 A 值或 B 值时，可使用。

语法：Character *:value OR value

参数说明：Character：字符，*可以放在字符中的任何位置，可以写多个字段值，中间用 OR。

应用场景：re*nse:(200 OR 404)，搜索 response 的值是 200 或者 404 的数据。

返回结果：



3.7 _missing_

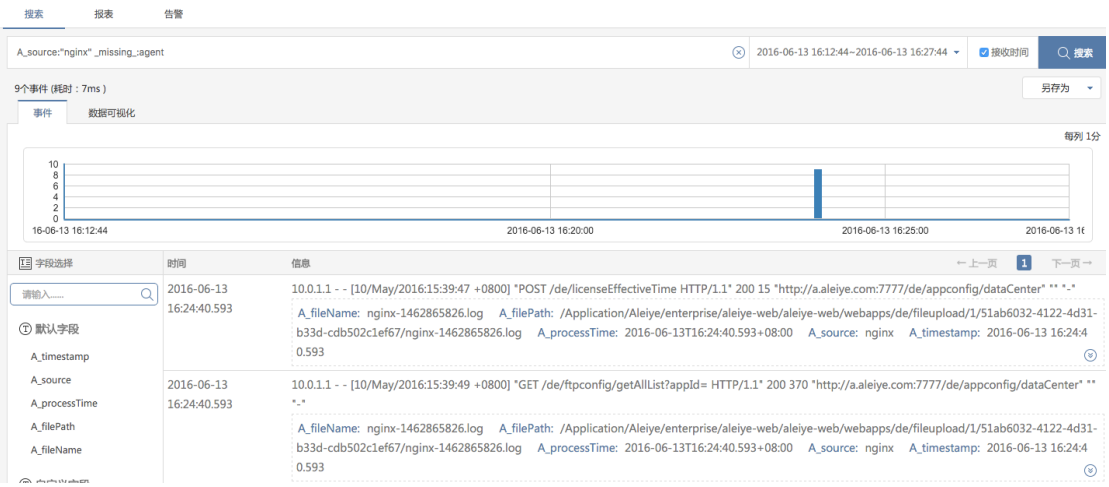
说明：查询某字段无值或为空。

语法：_missing_：field

参数说明：field：字段

应用场景：_missing_:agent

返回结果：



3.8 _exists_

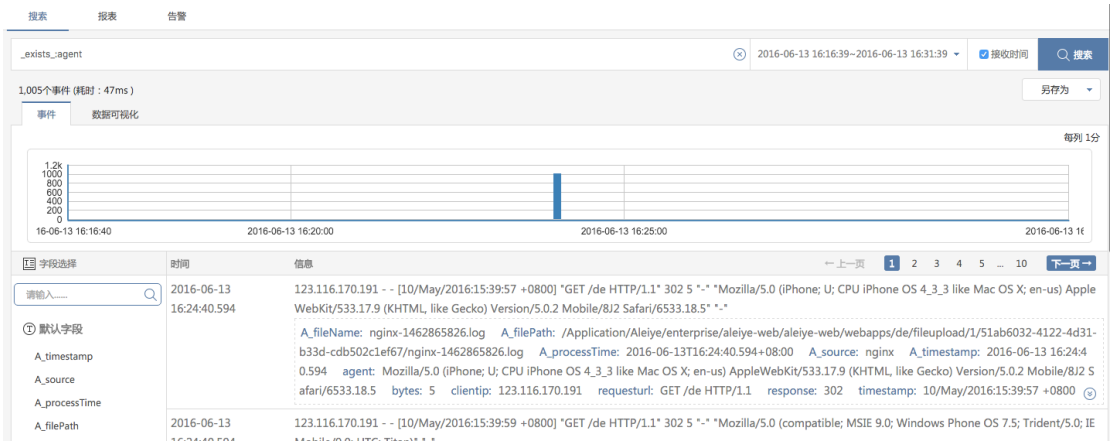
说明：查询某字段包含不为空的值。

语法：__exists_:field

参数说明：field：字段

应用场景：_missing_:agent

返回结果：



3.9 / /

说明：正则匹配开始、结束标识。此处正则表达式支持以下格式

语法	描述	示例（字符：aaabbb）
+	用于之前字符集重复一次或多次	a+b+ aa+bb+ aa+bbb+
{}	用于指定最小和最大次数	{5} # 重复5次 {2,5} # 重复最少2次，最大5次 {2,} #重复最小 2 次
[]	用于可能包含的字符集	[abc] #'a'或'b'或'c'
^	用于非	[^abc] # 不包含'a'或'b'或'c'
-	用于两字符中间，可指定两个字符之间范围	[a-c] # #'a'或'b'或'c'

3.10 ~

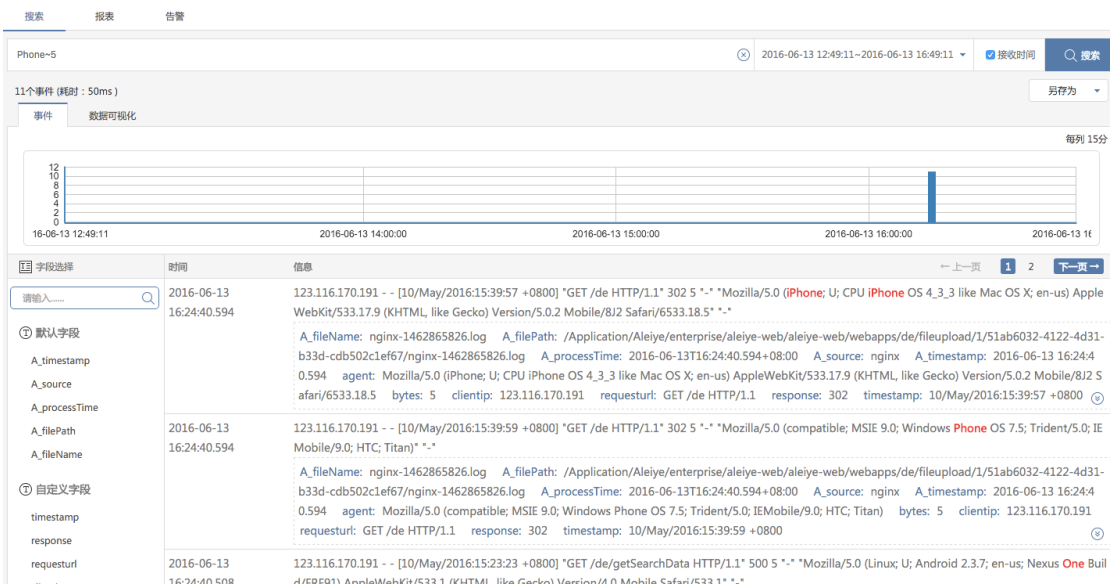
说明：可以通过~符号查询相近的关键字。

语法：Character~

参数说明：Character：字符，在~后面可以填写 1-10 来判定其相似度高低，数值越大相似度低反之越高。

应用场景：Phone~5

返回结果：



3.11 to

说明：范围检索，范围查询允许指定某个字段最大值和最小值，查询在二者之间的所有文档。范围查询可以包含或者不包含最大值和最小值。

语法：field：[value TO value]

参数说明：field：字段，value：字段值， []：包含关系，{}:不包含关系

应用场景：byte:[0 TO 1000]，流量数在 0 到 1000 字节间的范围。

返回结果：

3.12 OR

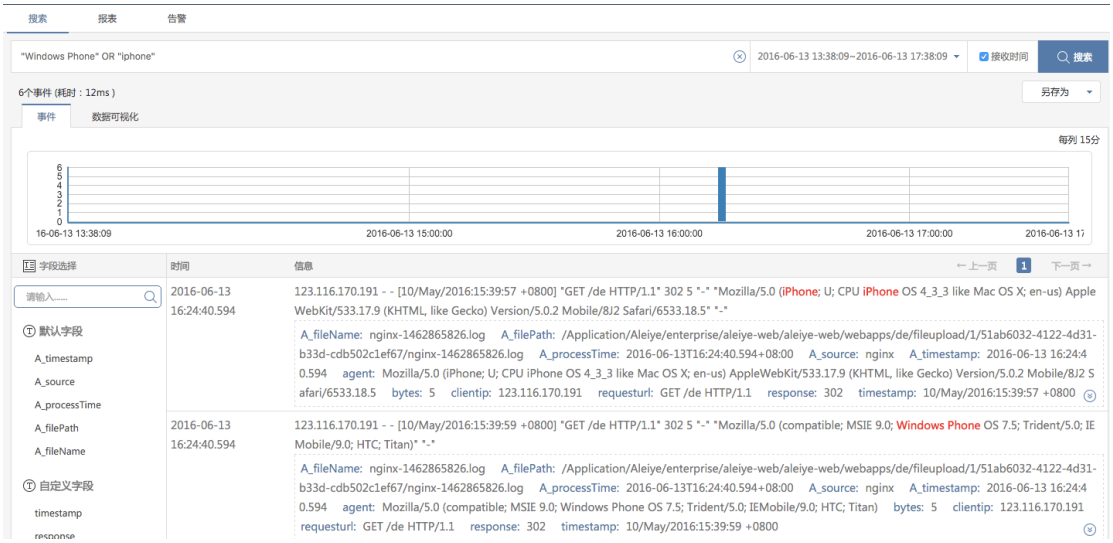
说明：或操作符，检索条件之间使用 OR 链接 那么检索结果将是符合两个条件的并集。

语法：Character OR Character

参数说明：Character:字符

应用场景：”OS 7.5” OR “IE”。

返回结果：



3.13 AND

说明：与操作符，规定必须所有的检索条件都出现才能满足查询条件。

语法：Character AND Character

参数说明：Character:字符

应用场景：”OS 7.5” AND “IE”

返回结果：

3.14 NOT

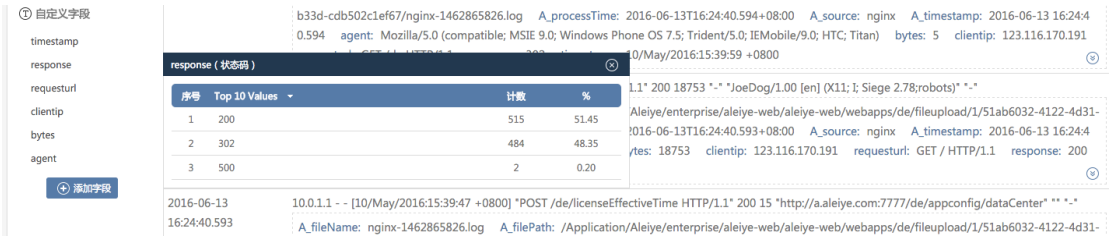
说明：非操作符，规定查询的文档必须不包含 NOT 之后的检索条件。

语法：NOT Character

参数说明：Character:字符

应用场景：NOT ″404″。

返回结果：



3.15 \

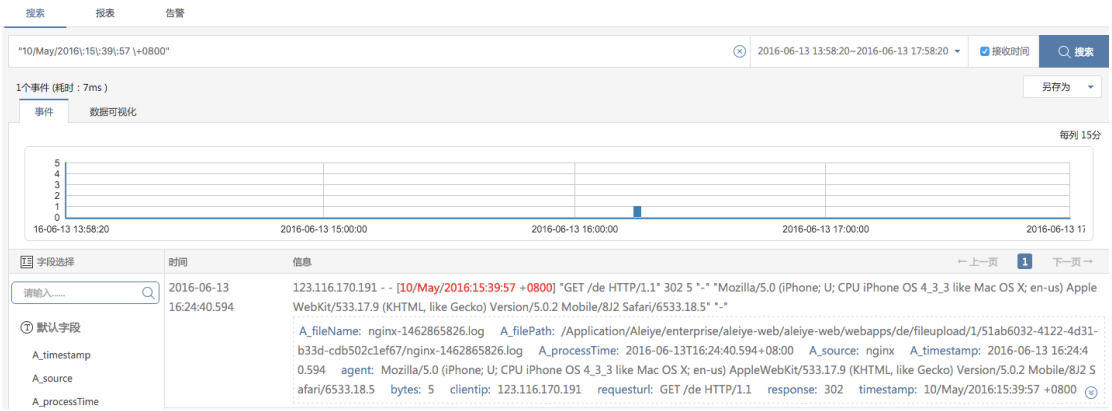
说明：特殊字符转译

语法：\特殊字符

参数说明：特殊字符包括：+ - && || ! ( ) { } [ ] ^ " ~ * ? : \

应用场景："10/May/2016\:15\:39\:57 \+0800",对数据中的：和+进行了转译。

返回结果：



3.16 （）

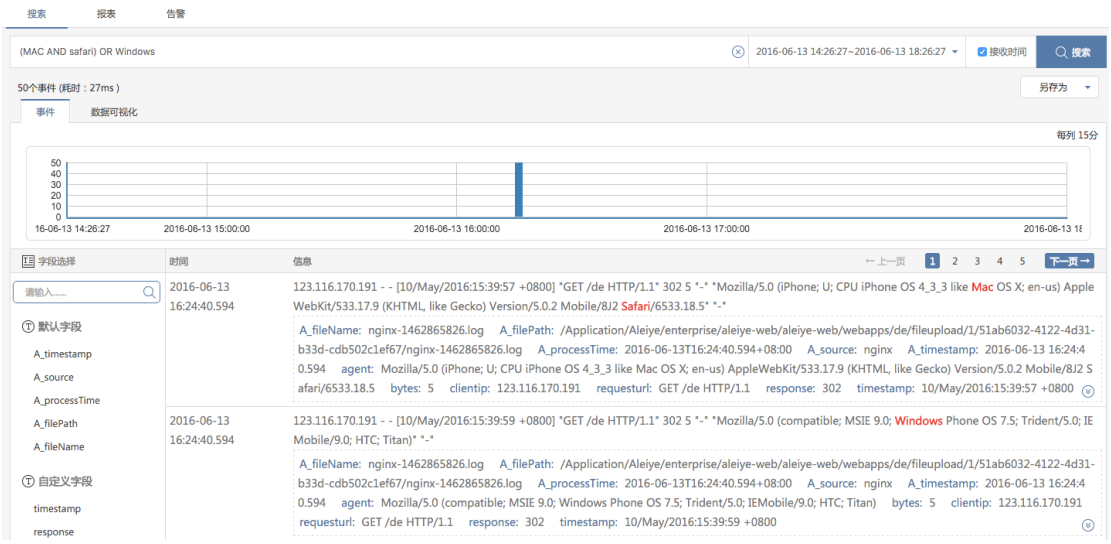
说明：使用圆括号来组合语句形成子查询。

语法：（ ）

参数说明：无

应用场景：(MAC AND safari) OR Windows

返回结果：



3.17 toDate

说明：将字符串类型字段，按格式转化为时间戳。

语法：*:* | eval alias=toDate(field , format)

参数说明：alias：给自定义字段起名；field：字段，必须是字符串类型；format：日期格式化字符串，如 yyyy-MM-dd hh:mm:ss，有特殊字符时需加双引号。

应用场景 1：timestamp= 10/May/2016:15:39:59 +0800,该字段类型为字符串，通过 toDate 命令将其转化成日期格式。

A_source:"test" | eval mytime=toDate(timestamp,"dd/MMM/yyyy:HH:dd:ss +0800")

返回结果：

应用场景 1：在上述场景的基础上，可以在语句追加报表命令。

A_source:"test" |eval mytime=toDate(timestamp,"dd/MMM/yyyy:HH:dd:ss +0800") | datereport count(bytes) over mytime on clientip span=1h

返回结果：

3.18 toNum

说明：将字段转换为数值类型。

语法：*:* | eval alias=toNum(field)

参数说明：alias：给自定义字段起名；field：字段。

应用场景 1：bytes 字段为字符串类型，想对该字段进行统计，通过 toNum 命令将其转化成数值类型

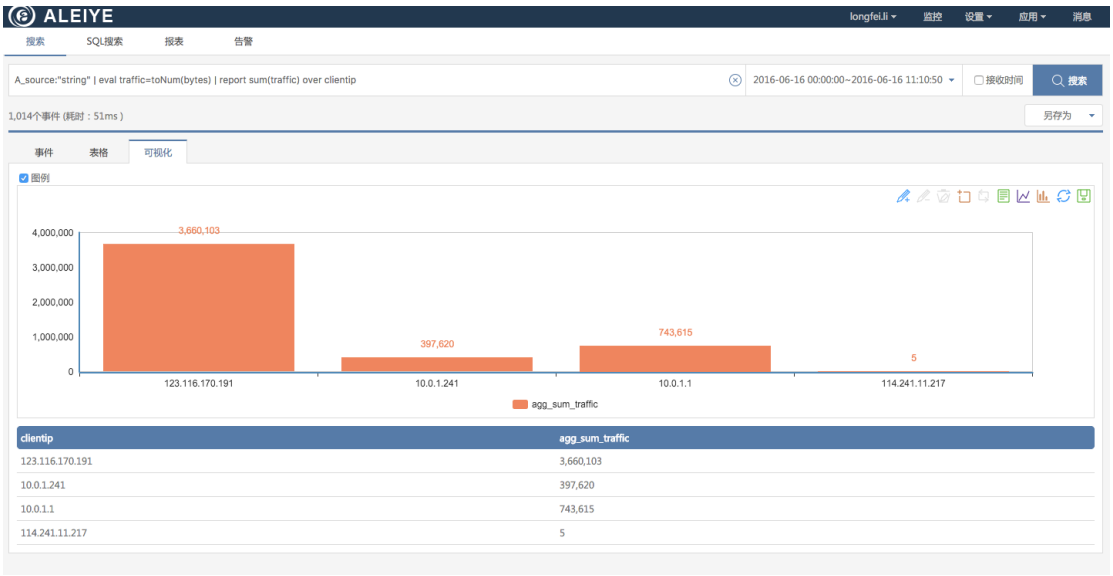
A_source:"test" | eval traffic=toNum(bytes)

返回结果：

应用场景 2：在上述场景的基础上，将转化好的 bytes 字段进行每个 IP 所产生的流量进行统计。

A_source:"test" | eval traffic=toNum(bytes) | report sum(traffic) over clientip

返回结果：



3.19 substr

说明：**对字符串进行截取**。

语法：*:* | eval alias= substr(field, startIndex [, endIndex]?)

参数说明：alias：给自定义字段起名；field：字段；startIndex：起始位置，从 0 开始；endIndex:截止位置。

应用场景 1：clientIp=123.116.170.191，通过 substr 命令可以将该字段进行截取网段。

A_source:”test” | eval ip=substr(clinentIp,12)

返回结果：

3.20 concat

说明：**对多个字段数据进行拼接**。

语法：*:* | eval alias= concat(field1, field2)

参数说明：alias：给自定义字段起名；field1：字段 1；field2：字段 2

应用场景： ip 和访问状态码两个字段，通过 concat 命令进行拼接，可以直观的看到每个访问 IP 所返回的状态是什么。

A_source:”test” | eval ipresponse=concat(clientip,response)

返回结果：

3.21 trim

说明：移除字段 field 值前后空格。

语法：*:* | eval alias= trim(field)

参数说明：alias：给自定义字段起名；field：字段。

应用场景 1：

A_source:”test” | eval request=trim(requesturl)

3.22 upper

说明：将字段类数据转换为大写显示。

语法：*:* | eval alias= upper(field)

参数说明：alias：给自定义字段起名；field：字段。

应用场景：将数据类型的名称通过 upper 命令将其转化成大写。

A_source:”test” | eval source=upper(A_source)

返回结果：

3.23 lower

说明：将字段值转换为小写。

语法：*:* | eval alias= lower(field)

参数说明：alias：给自定义字段起名；field：字段。

应用场景：

A_source:”test” | eval source=lower(A_source)

返回结果：

3.24 split

说明：将字符串类型数据按给定的正则表达式来切分数据，并指写切分后的第几个数据。

语法：*:* | eval alias=split(field [, seperator]?)[[num]]?

参数说明：alias：给自定义字段起名；field：字段；seperator：分隔符，需加双引号,默认使用逗号分隔；num：切分出的数组的下标。

应用场景 1：基于 clintIp=10.0.1.132 的字段进行最后网段的统计

A_source:"test" | eval ip=split(clintIp,".") [4]

返回结果：

应用场景 2：对切分完的字段值，进行排名统计。

A_source:"test" | eval ip=split(clintIp,".") [2] | top 10 ip

3.25 cal

说明：对字段进行四则混合运算。

语法：*:* | eval alias=cal(string)

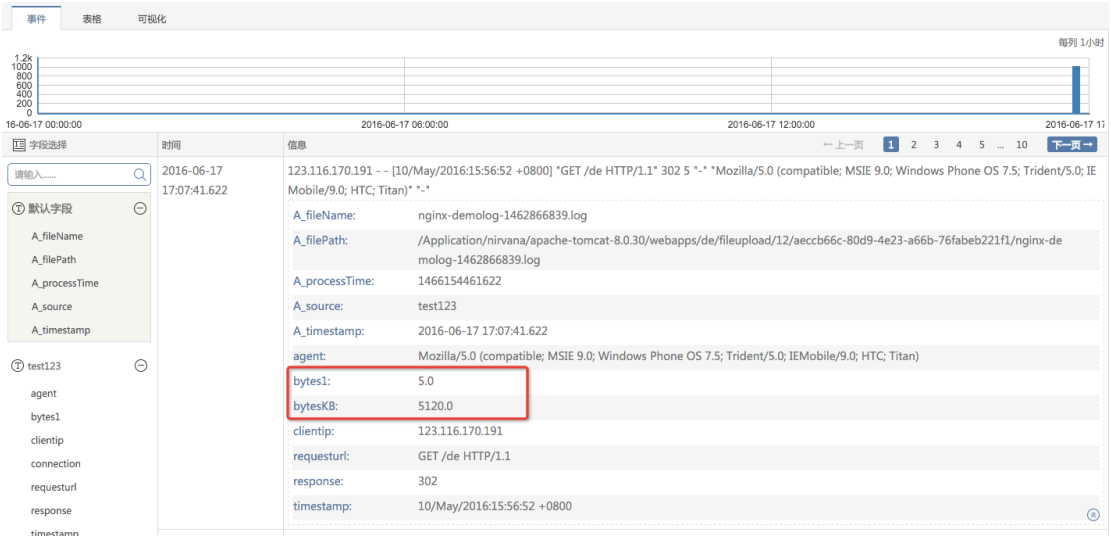
参数说明：alias：c 给自定义字段起名；field：字段；string：需加双引号，支持+、-、*、/、();。

应用场景 1：bytes：3400，默认 bytes 值的单位是字节，可以通过 cal 命令

转化成 kb

A_source:"test123" | eval bytesKB=cal("bytes1/1024")

返回结果：



3.26 scope

说明：筛选出 field 字段值在与 toNum 之间的记录。

语法：*:* | eval alias= scope(field, fromNum [, includeFrom]?, toNum [, includeTo]?)

参数说明：alias：给自定义字段起名；field：文档中的字段，字段为数值类型；fromNum：最小值；includeFrom：是否包含最小值,该值可省略，默认为 true；toNum：最大值；includeTo：是否包含最大值，该值可省略，默认为 false。

应用场景:对流量在 0 到 5000 的字段值通过 scope 命令进行筛选操作。

A_source:"test" | eval byte=scope(bytes,0,5000)

返回结果：

时间	信息
2016-06-16 18:21:31.669	123.116.170.191 - - [10/May/2016:15:39:57 +0800] "GET /de HTTP/1.1" 302 5 "-" "Mozilla/5.0 (iPhone; U; CPU iPhone OS 4_3_3 like Mac OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8J2 Safari/6533.18.5" "-"
	A_fileName: nginx-1462865826.log
	A_filePath: /Application/nirvana/apache-tomcat-8.0.30/webapps/de/fileupload/12/71d59522-7649-488b-a773-ef1ce996039e/nginx-1462865826.log
	A_processTime: 1466072491669
	A_source: test1
	A_timestamp: 2016-06-16 18:21:31.669
	agent: Mozilla/5.0 (iPhone; U; CPU iPhone OS 4_3_3 like Mac OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8J2 Safari/6533.18.5
	byte: 5.0
	bytes: 5
	clientip: 123.116.170.191
	requesturl: GET /de HTTP/1.1
	response: 302
	timestamp: 10/May/2016:15:39:57 +0800

3.27 regex

说明：对字段按照正则进行匹配。

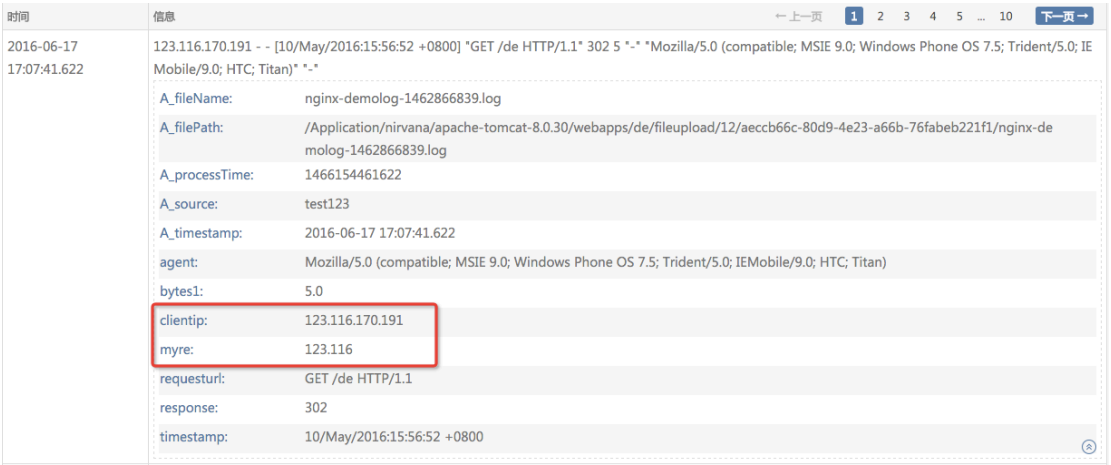
语法：*:* | eval alias= regex(field, regexStr, [, group]?)

参数说明：alias：给自定义字段起名；field：文档中的字段，字段为字符串类型；regexStr：正则表达式；group：子序列索引，group 为空时返回匹配成功的字段 field 值，group 不为空时，返回匹配成功的 group 子序列。

应用场景：基于 clientip 的字段，对其进行正则切分。

:|eval myre = regex(clientip,"(?<gv>\d{1,3}\.\d{1,3})",gv)

返回结果：



时间	信息
2016-06-17 17:07:41.622	123.116.170.191 - - [10/May/2016:15:56:52 +0800] "GET /de HTTP/1.1" 302 5 "-" "Mozilla/5.0 (compatible; MSIE 9.0; Windows Phone OS 7.5; Trident/5.0; IEMobile/9.0; HTC; Titan)" "-"
	A_fileName: nginx-demolog-1462866839.log
	A_filePath: /Application/nirvana/apache-tomcat-8.0.30/webapps/de/fileupload/12/aeccb66c-80d9-4e23-a66b-76fabeb221f1/nginx-demolog-1462866839.log
	A_processTime: 1466154461622
	A_source: test123
	A_timestamp: 2016-06-17 17:07:41.622
	agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows Phone OS 7.5; Trident/5.0; IEMobile/9.0; HTC; Titan)
	bytes1: 5.0
	clientip: 123.116.170.191
	myre: 123.116
	requesturl: GET /de HTTP/1.1
	response: 302
	timestamp: 10/May/2016:15:56:52 +0800

3.28 datebet

说明：求两个日期的间隔，即 to-from 以 span 为单位的间隔。

语法：*:* | eval alias= datebet(from [, formatStr1]? [,default=defaultValue1]?) (to [, formatStr2]? [,default=defaultValue2]?) [span]?

参数说明：alias：给自定义字段起名；from：起始时间字段；formatStr1：日期格式化字符串，用来将 from 格式化为日期对象；defaultValue1：from 字段的值不存在时，默认使用 defaultValue1 的值作为起始时间；to：截止时间字段；formatStr2：日期格式化字符串，用来将 to 格式化为日期对象；span：取值'y'|'M'|'w'|'d'|'h'|'m'|'s'，分别表示年、月、周、天、小时、分、秒。

应用场景：

A_source:"test" | eval agent= datebet(time1,"yyyy-MM-dd HH:mm:ss.SSS")(time2,"yyyyMMddHHmmss.SSS")

返回结果：

3.29 report

说明：此命令用来制作图表，可通过不同维度展示数据，图表中维度用 X 轴来展现。

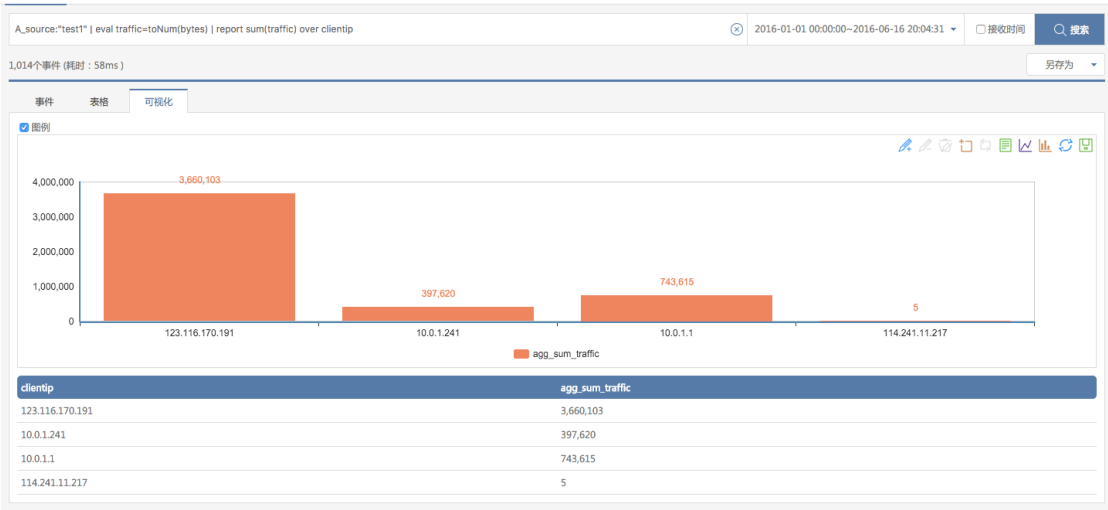
语法：report [stats-function]+ [over field [as alias]? limit?]? [on field [as alias]? limit?]+]?

参数说明：stats-function：统计表达式，支持 count、max、min、avg、dcount；field：field 为文档中的字段名称(英文名)，或者是设置的别名；limit：取值为正整数，指分组的最大个数，默认为 100；over field：数据以 field 字段聚合分组，展现在图表中时自动作为 x 轴；on field：数据以 field 字段聚合分组；as alias：给字段取别名，返回的数据标识是别名.

应用场景：

A_source:"test1" | eval traffic=toNum(bytes) | report sum(traffic) over clientip

返回结果：



3.30 datereport

说明：此命令用来制作以时间为 X 轴的时间趋势图表。

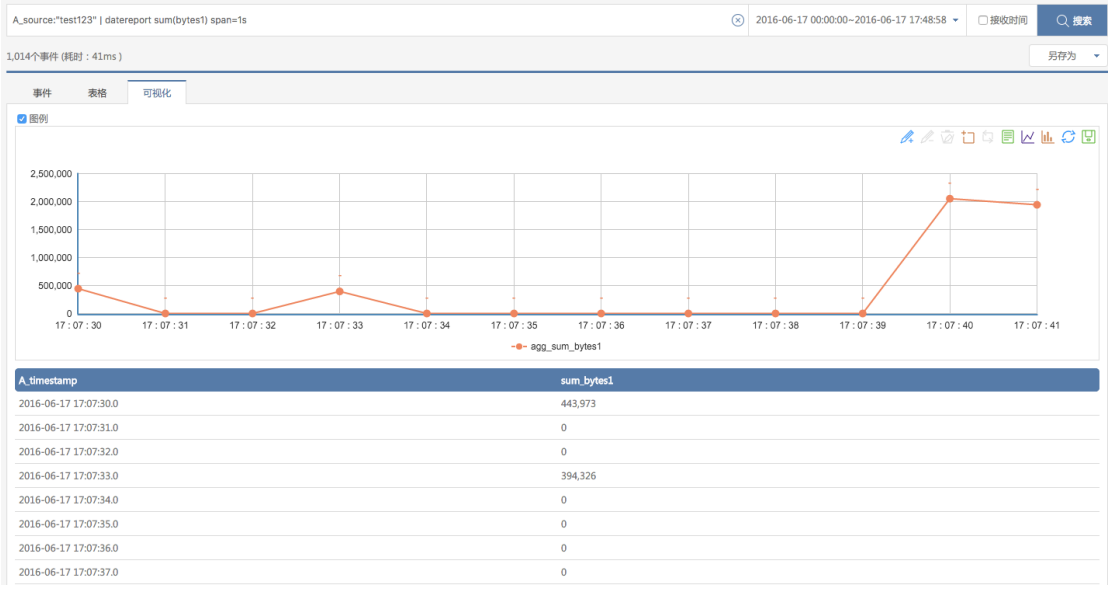
语法：datereport [stats-function]+[on [field [as alias] limit?]+]? [span=num[yMwdhms]]?

参数说明：stats-function: 统计表达式，支持 count、max、min、avg、dcount; as alias: 给字段取别名，返回的数据标识是别名; on field: 数据以 field 字段聚合分组; limit: 取值为正整数，指分组的最大个数，默认为 100; span=num[yMwdhms]: 时间粒度.

应用场景：

A_source:"test123" | datereport sum(bytes1) span=1s

返回结果：



A_timestamp	sum_bytes1
2016-06-17 17:07:30.0	443,973
2016-06-17 17:07:31.0	0
2016-06-17 17:07:32.0	0
2016-06-17 17:07:33.0	394,326
2016-06-17 17:07:34.0	0
2016-06-17 17:07:35.0	0
2016-06-17 17:07:36.0	0
2016-06-17 17:07:37.0	0

3.31 TOP

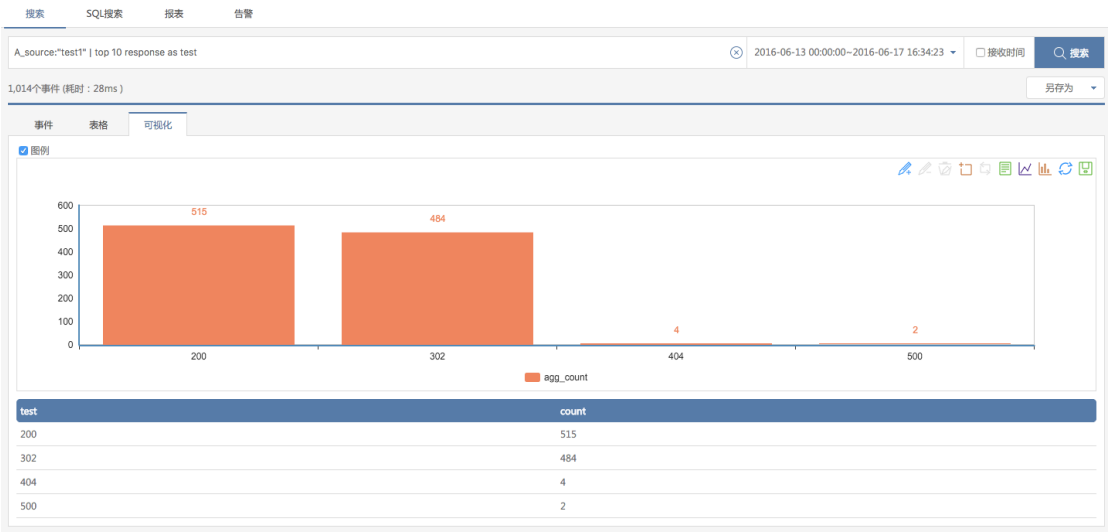
说明：此命令用来对信息进行统计数量的图表。

语法：top [limit]? field [as alias]?

参数说明：limit：取值为正整数，指字段出现次数最多的前多少个；field：field 为文档中的字段名称(英文名)，或者是设置的别名；

应用场景：A_source:"test1" | top 10 response as test

返回结果：



test	count
200	515
302	484
404	4
500	2

3.32 Rangereport

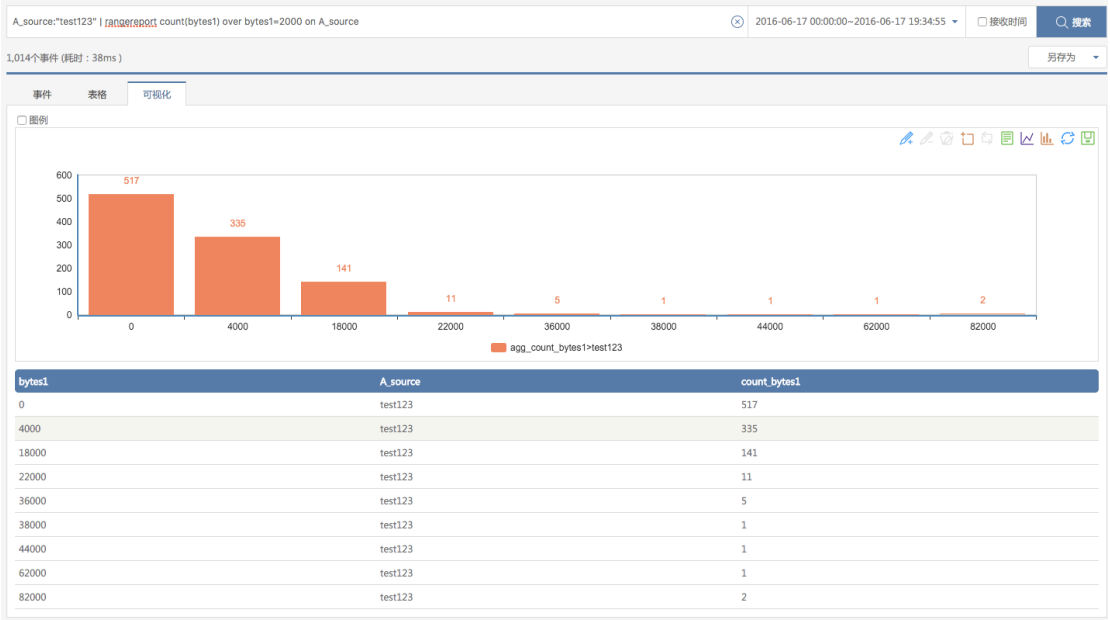
说明：此命令用来对数值型字段进行分组统计。

语法：RANGEREPORT [stats-function]+ OVER field [AS alias]? RANGE=num [on [field [as alias] limit?]+]?

参数说明：stats-function：统计表达式，支持 count、max、min、avg、dcount；over field [as alias]? range=num：以 field 字段分组，num 取值正整数，分组时每隔 num 范围聚合，该 field 字段做 x 轴；on [field [as alias] limit?]+：以 field 字段将数据分组，over field 与 on field 中的 field 没有关联关系，可以是不同的字段

应用场景：A_source:"wu001" | rangereport sum(bytes) over bytes=10000

返回结果：



bytes1	A_source	count_bytes1
0	test123	517
4000	test123	335
18000	test123	141
22000	test123	11
36000	test123	5
38000	test123	1
44000	test123	1
62000	test123	1
82000	test123	2

第4章 AleiyeSQL 检索命令列表

分类	命令	描述
关系运算	=	等值比较
	<>	不等值比较
	<	小于比较
	<=	小于等于比较
	>	大于比较
	>=	大等于比较
	Is null	控制判断
	Is notnull	非空判断
	Like	Like 比较
	Rlike	Jave 的 lnke 操作
	Regexp	Regexp 操作
数学运算	+	加法操作
	-	减法操作
	*	乘法操作
	/	除法操作
	%	取余操作
	&	位与操作
	|	位或操作
	^	位异或操作
	~	位取反操作
逻辑运算	AND	逻辑与操作
	OR	逻辑或操作
	NOT	逻辑非操作
数值计算	Round	取整函数

	Round	指定精度取整函数
	Fllor	向下取整函数
	Ceil	向上取整函数
	Rand	取随机函数
	Exp	自然指数函数
	Log10	以 10 为底对数函数
	Log2	以 2 为底对数函数
	Log	对数函数
	Pow	幂运算函数
	Power	幂运算函数
	Sqrt	开平方函数
	Bin	二进制函数
	Hex	十六进制函数
	Unlex	反转十六进制函数
	Conv	进制转换函数
	Abs	绝对值函数
	Pmod	正取余函数
	Sin	正弦函数
	Asin	反正弦函数
	Cos	余弦函数
	Acos	反余弦函数
	Positive	Positive 函数
	Negative	Negative 函数
条件函数	If	If 函数
	Coalesce	非空查找函数
	Case	条件判断函数
字符串函数	Length	字符串长度函数

	Reverse	字符串反转函数
	Concat	字符串连接函数
	Concat_ws	带分隔符字符串连接函数
	Substr,substring	字符串截取函数
	Upper,ucase	字符串转大写函数
	Lower,lcase	字符串转小写函数
	Trim	去空格函数
	Ltrim	左边去空格函数
	Rtrim	右边去空格函数
	Regexp_replace	正则表达式替换函数
	Regexp_extract	正则表达式解析函数
	Parse_url	URL 解析函数
	Get_json_object	Josn 解析函数
	Space	空格字符串函数
	Repeat	重复字符串函数
	Ascii	首字符 ascii 函数
	lpad	左补足函数
	Rpad	右补足函数
	Split	分隔字符串函数
	Find_in_set	集合查找函数
集合统计函数	Count	个数统计函数
	Sum	总和统计函数
	Avg	平均值统计函数
	Min	最小值统计函数
	Max	最大值统计函数
	Var_pop	非空集合总体变量函数
	Var_samp	非空集合样本变量函数

	Stddev_pop	总体标准偏离函数
	Stddev_samp	样本标准偏离函数
	Percentile	中位数函数
	percentile_approx	近似中位数函数
	percentile_approx	近似中位数函数
	histogram_numeric	直方图
字段类型转换	lDateToChar	将 long 型的时间转换成字符串
	strDateToLong	将字符串类型的时间转换成 long 型

第5章SQL 命令

5.1 等值比较：=

语法：A=B

操作类型：所有基本类型

描述： 如果表达式 A 与表达式 B 相等，则为 TRUE；否则为 FALSE

举例：#>select 1 from lxw_dual where 1=1;

返回结果：1

5.2 不等值比较：<>

语法：A <> B

操作类型： 所有基本类型

描述： 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 与表达式 B 不相等，则为 TRUE；否则为 FALSE

举例：

#> select1 from lxw_dual where 1 <> 2;

返回结果：1

5.3 小于比较：<

语法：A < B

操作类型： 所有基本类型

描述： 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 小于表达式 B，则为 TRUE；否则为 FALSE

举例：

```
#> select1 from lxw_dual where 1 < 2;
```

返回结果：1

5.4 小于等于比较：<=

语法：A <= B

操作类型： 所有基本类型

描述： 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 小于或者等于表达式 B，则为 TRUE；否则为 FALSE

举例：

```
#> select1 from lxw_dual where 1 <= 1;
```

返回结果：1

5.5 大于比较：>

语法：A > B

操作类型： 所有基本类型

描述： 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 大于表达式 B，则为 TRUE；否则为 FALSE

举例：

```
#> select1 from lxw_dual where 2 > 1;

1
```

5.6 大于等于比较：>=

语法: A >= B

操作类型: 所有基本类型

描述: 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 大于或者等于表达式 B，则为 TRUE；否则为 FALSE

举例：

```
#> select1 from lxw_dual where 1 >= 1;

1
```

5.7 空值判断：is null

语法: A IS NULL

操作类型: 所有类型

描述: 如果表达式 A 的值为 NULL，则为 TRUE；否则为 FALSE

举例：

```
#> select1 from lxw_dual where null is null;

1
```

5.8 非空判断：Is notnull

语法: A IS NOT NULL

操作类型: 所有类型

描述: 如果表达式 A 的值为 NULL，则为 FALSE；否则为 TRUE

举例：

```
#> select1 from lxw_dual where 1 is not null;
```

1

5.9 Like 比较：Like

语法: A LIKE B

操作类型: strings

描述: 如果字符串 A 或者字符串 B 为 NULL，则返回 NULL；如果字符串 A 符合表达式 B 的正则语法，则为 TRUE；否则为 FALSE。B 中字符”_”表示任意单个字符，而字符”%”表示任意数量的字符。

举例：

```
#> select1 from lxw_dual where 'football' like 'foot%';
```

1

```
#> select1 from lxw_dual where 'football' like 'foot____';
```

1

5.10 ave 的 lnke 操作：rlike

语法: A RLIKE B

操作类型: strings

描述: 如果字符串 A 或者字符串 B 为 NULL，则返回 NULL；如果字符串 A 符合 Java 正则表达式 B 的正则语法，则为 TRUE；否则为 FALSE。

举例：

```
#> select1 from lxw_dual where 'footbar' rlike '^f.*r$' ;
1
```

注意：判断一个字符串是否全为数字：

```
#>select 1from lxw_dual where '123456' rlike '^\\d+$';
1
#> select1 from lxw_dual where '123456aa' rlike '^\\d+$';
```

5.11 Regexp 操作：Regexp

语法: A REGEXP B

操作类型: strings

描述: 功能与 RLIKE 相同

举例：

```
#> select1 from lxw_dual where 'footbar' REGEXP '^f.*r$';
```

1

5.12 加法操作：+

语法: A + B

操作类型：所有数值类型

说明：返回 A 与 B 相加的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型（详见数据类型的继承关系）。比如，int + int 一般结果为 int 类型，而 int + double 一般结果为 double 类型

举例：

```
#> select1 + 9 from lxw_dual;

10

#> createtable lxw_dual as select 1 + 1.2 from lxw_dual;

#> describelxw_dual;

_c0  double
```

5.13 减法操作：-

语法: A - B

操作类型：所有数值类型

说明：返回 A 与 B 相减的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型（详见数据类型的继承关系）。比如，int - int 一般结果为 int 类型，而 int - double 一般结果为 double 类型

举例：

```
#> select10 - 5 from lxw_dual;

5

#> createtable lxw_dual as select 5.6 - 4 from lxw_dual;

#>describe lxw_dual;

_c0  double
```

5.14 乘法操作：*

语法: A * B

操作类型：所有数值类型

说明：返回 A 与 B 相乘的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型（详见数据类型的继承关系）。注意，如果 A 乘以 B 的结果超过默认结果类型的数值范围，则需要通过 cast 将结果转换成范围更大的数值类型

举例：

```
#> select40 * 5 from lxw_dual;

200
```

5.15 除法操作：/

语法: A / B

操作类型：所有数值类型

说明：返回 A 除以 B 的结果。结果的数值类型为 double

举例：

```
#> select40 / 5 from lxw_dual;
```

8.0

注意：#中最高精度的数据类型是 double,只精确到小数点后 16 位，在做除法运算的时候要特别注意

```
#>select ceil(28.0/6.999999999999999999) from lxw_duallimit 1;
```

结果为 4

```
#>select ceil(28.0/6.99999999999999) from lxw_dual
limit1;
```

结果为 5

5.16 取余操作：%

语法: A % B

操作类型：所有数值类型

说明：返回 A 除以 B 的余数。结果的数值类型等于 A 的类型和 B 的类型的最小父类型（详见数据类型的继承关系）。

举例：

```
#> select 41 % 5 from lxw_dual;
```

1

```
#> select 8.4 % 4 from lxw_dual;
```

0.40000000000000036

注意：精度在#中是个很大的问题，类似这样的操作最好通过 round 指定精度

```
#> select round(8.4 % 4 , 2) from lxw_dual;
```

0.4

5.17 位与操作：&

语法: A & B

操作类型：所有数值类型

说明：返回 A 和 B 按位进行与操作的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型（详见数据类型的继承关系）。

举例：

```
#> select 4 & 8 from lxw_dual;
```

0

```
#> select 6 & 4 from lxw_dual;
```

4

5.18 位或操作：|

语法: A | B

操作类型：所有数值类型

说明：返回 A 和 B 按位进行或操作的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型（详见数据类型的继承关系）。

举例：

```
#> select 4 | 8 from lxw_dual;

12

#> select 6 | 8 from lxw_dual;

14
```

5.19 位异或操作：^

语法：A ^ B

操作类型：所有数值类型

说明：返回 A 和 B 按位进行异或操作的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型（详见数据类型的继承关系）。

举例：

```
#> select 4 ^ 8 from lxw_dual;

12

#> select 6 ^ 4 from lxw_dual;

2
```

5.20 位取反操作：~

语法：~A

操作类型：所有数值类型

说明：返回 A 按位取反操作的结果。结果的数值类型等于 A 的类型。

举例：

```
#> select ~6 from lxw_dual;

-7

#> select ~4 from lxw_dual;

-5
```

5.21 逻辑与操作：AND

语法：A AND B

操作类型：boolean

说明：如果 A 和 B 均为 TRUE，则为 TRUE；否则为 FALSE。如果 A 为 NULL 或 B 为 NULL，则为 NULL

举例：

```
#> select 1 from lxw_dual where 1=1 and 2=2;

1
```

5.22 逻辑或操作：OR

语法: A OR B

操作类型：boolean

说明：如果 A 为 TRUE，或者 B 为 TRUE，或者 A 和 B 均为 TRUE，则为 TRUE；否则为 FALSE

举例：

```
#> select 1 from lxw_dual where 1=2 or 2=2;
1
```

5.23 逻辑非操作：NOT

语法: NOT A

操作类型：boolean

说明：如果 A 为 FALSE，或者 A 为 NULL，则为 TRUE；否则为 FALSE

举例：

```
#> select 1 from lxw_dual where not 1=2;
1
```

5.24 取整函数：Round

语法: round(double a)

返回值: BIGINT

说明： 返回 double 类型的整数值部分 （遵循四舍五入）

举例：

```
#> select round(3.1415926) from lxw_dual;

3

#> select round(3.5) from lxw_dual;

4

#> create table lxw_dual as select round(9542.158) fromlxw_dual;

#> describe lxw_dual;

_c0  bigint
```

5.25 指定精度取整函数：Round

语法: round(double a, int d)

返回值: DOUBLE

说明： 返回指定精度 d 的 double 类型

举例：

```
#> select round(3.1415926,4) from lxw_dual;

3.1416
```

5.26 向下取整函数：Fllor

语法: floor(double a)

返回值: BIGINT

说明: 返回等于或者小于该 double 变量的最大的整数

举例:

```
#> select floor(3.1415926) from lxw_dual;
3
#> select floor(25) from lxw_dual;
25
```

5.27 向上取整函数：Ceil

语法: ceil(double a)

返回值: BIGINT

说明: 返回等于或者大于该 double 变量的最小的整数

举例:

```
#> select ceil(3.1415926) from lxw_dual;
4
#> select ceil(46) from lxw_dual;
46
```

5.28 取随机函数：Rand

语法: rand(),rand(int seed)

返回值：double

说明： 返回一个 0 到 1 范围内的随机数。如果指定种子 seed，则会等到一个稳定的随机数序列

举例：

```
#> select rand() from lxw_dual;

0.5577432776034763

#> select rand() from lxw_dual;

0.6638336467363424

#> select rand(100) from lxw_dual;

0.7220096548596434

#> select rand(100) from lxw_dual;

0.7220096548596434
```

5.29 自然指数函数：Exp

语法：exp(double a)

返回值：double

说明： 返回自然对数 e 的 a 次方

举例：

```
#> select exp(2) from lxw_dual;
```

7.38905609893065

自然对数函数：ln

语法：ln(double a)

返回值：double

说明： 返回 a 的自然对数

举例：

#> select ln(7.38905609893065) from lxw_dual;

2.0

5.30 以 10 为底对数函数：Log10

语法：log10(double a)

返回值：double

说明： 返回以 10 为底的 a 的对数

举例：

#> select log10(100) from lxw_dual;

2.0

5.31 以 2 为底对数函数：Log2

语法：log2(double a)

返回值：double

说明： 返回以 2 为底的 a 的对数

举例：

```
#> select log2(8) from lxw_dual;
```

3.0

5.32 对数函数：Log

语法：log(double base, double a)

返回值：double

说明： 返回以 base 为底的 a 的对数

举例：

```
#> select log(4,256) from lxw_dual;
```

4.0

5.33 幂运算函数：Pow

语法：pow(double a, double p)

返回值：double

说明： 返回 a 的 p 次幂

举例：

```
#> select pow(2,4) from lxw_dual;
```

16.0

5.34 幂运算函数：Power

语法: power(double a, double p)

返回值: double

说明: 返回 a 的 p 次幂,与 pow 功能相同

举例：

#> select power(2,4) from lxw_dual;

16.0

5.35 开平方函数：Sqrt

语法: sqrt(double a)

返回值: double

说明: 返回 a 的平方根

举例：

#> select sqrt(16) from lxw_dual;

4.0

5.36 二进制函数：Bin

语法: bin(BIGINT a)

返回值: string

说明: 返回 a 的二进制代码表示

举例:

```
#> select bin(7) from lxw_dual;
```

111

5.37 十六进制函数：Hex

语法: hex(BIGINT a)

返回值: string

说明: 如果变量是 int 类型，那么返回 a 的十六进制表示；如果变量是 string 类型，则返回该字符串的十六进制表示

举例:

```
#> select hex(17) from lxw_dual;
```

11

```
#> select hex( ‘abc’ ) from lxw_dual;
```

616263

5.38 反转十六进制函数：Unlex

语法: unhex(string a)

返回值: string

说明： 返回该十六进制字符串所代码的字符串

举例：

```
#> select unhex( ‘616263’ ) from lxw_dual;
abc
#> select unhex( ‘11’ ) from lxw_dual;
_
#> select unhex(616263) from lxw_dual;
abc
```

5.39 进制转换函数：Conv

语法: conv(BIGINT num, int from_base, int to_base)

返回值: string

说明： 将数值 num 从 from_base 进制转化到 to_base 进制

举例：

```
#> select conv(17,10,16) from lxw_dual;
11
#> select conv(17,10,2) from lxw_dual;
10001
```

5.40 绝对值函数：Abs

语法: abs(double a)　　abs(int a)

返回值: double　　int

说明:　返回数值 a 的绝对值

举例:

```
#> select abs(-3.9) from lxw_dual;
3.9
#> select abs(10.9) from lxw_dual;
10.9
```

5.41 正取余函数：Pmod

语法: pmod(int a, int b),pmod(double a, double b)

返回值: int double

说明:　返回正的 a 除以 b 的余数

举例:

```
#> select pmod(9,4) from lxw_dual;
1
#> select pmod(-9,4) from lxw_dual;
3
```

5.42 正弦函数：Sin

语法: sin(double a)

返回值: double

说明: 返回 a 的正弦值

举例：

```
#> select sin(0.8) from lxw_dual;
```

0.7173560908995228

5.43 反正弦函数：Asin

语法: asin(double a)

返回值: double

说明: 返回 a 的反正弦值

举例：

```
#> select asin(0.7173560908995228) from lxw_dual;
```

0.8

5.44 余弦函数：Cos

语法: cos(double a)

返回值: double

说明: 返回 a 的余弦值

举例：

#> select cos(0.9) from lxw_dual;

0.6216099682706644

5.45 反余弦函数：Acos

语法: acos(double a)

返回值: double

说明: 返回 a 的反余弦值

举例：

#> select acos(0.6216099682706644) from lxw_dual;

0.9

5.46 Positive 函数：Positive

语法: positive(int a), positive(double a)

返回值: int double

说明: 返回 a

举例：

#> select positive(-10) from lxw_dual;

-10

#> select positive(12) from lxw_dual;

12

5.47 Negative 函数：Negative

语法: negative(int a), negative(double a)

返回值: int double

说明: 返回-a

举例：

#> select negative(-5) from lxw_dual;

5

#> select negative(8) from lxw_dual;

-8

5.48 If 函数函数：If

语法: if(boolean testCondition, T valueTrue, T valueFalseOrNull)

返回值: T

说明: 当条件 testCondition 为 TRUE 时，返回 valueTrue；否则返回 valueFalseOrNull

举例：

#> select if(1=2,100,200) from lxw_dual;

200

```
#> select if(1=1,100,200) from lxw_dual;

100
```

5.49 非空查找函数：Coalesce

语法: COALESCE(T v1, T v2, …)

返回值: T

说明: 返回参数中的第一个非空值；如果所有值都为 NULL，那么返回 NULL

举例：

```
#> select COALESCE(null,'100','50' ) from lxw_dual;

100
```

5.50 条件判断函数：Case

语法: CASE a WHEN b THEN c [WHEN d THEN e]* [ELSE f] END

返回值: T

说明：如果 a 等于 b，那么返回 c；如果 a 等于 d，那么返回 e；否则返回 f

举例：

```
#> Select case 100 when 50 then 'tom' when 100 then 'mary'else 'tim' end
from lxw_dual;

mary
```

```
#> Select case 200 when 50 then 'tom' when 100 then 'mary' else 'tim' end
from lxw_dual;

tim
```

5.51 字符串长度函数：Length

语法: length(string A)

返回值: int

说明：返回字符串 A 的长度

举例：

```
#> select length('abcedfg') from lxw_dual;

7
```

5.52 字符串反转函数：Reverse

语法: reverse(string A)

返回值: string

说明：返回字符串 A 的反转结果

举例：

```
#> select reverse(abcedfg' ) from lxw_dual;

gfdecba
```

5.53 字符串连接函数：Concat

语法: concat(string A, string B…)

返回值: string

说明：返回输入字符串连接后的结果，支持任意个输入字符串

举例：

#> select concat(‘abc’ ,’def’ ,’gh’) from lxw_dual;

abcdefgh

5.54 带分隔符字符串连接函数：Concat_ws

语法: concat_ws(string SEP, string A, string B…)

返回值: string

说明：返回输入字符串连接后的结果，SEP 表示各个字符串间的分隔符

举例：

#> select concat_ws(’,’,’abc’,’def’,’gh’) from lxw_dual;

abc,def,gh

5.55 字符串截取函数：Substr,substring

语法: substr(string A, int start),substring(string A, int start)

返回值: string

说明：返回字符串 A 从 start 位置到结尾的字符串

举例：

#> select substr('abcde',3) from lxw_dual;

cde

#> select substring('abcde',3) from lxw_dual;

cde

#> selectsubstr('abcde',-1) from lxw_dual; （和 ORACLE 相同）

e

5.56 字符串转大写函数：Upper,ucase

语法: substr(string A, int start, int len),substring(string A, intstart, int len)

返回值: string

说明：返回字符串 A 从 start 位置开始，长度为 len 的字符串

举例：

#> select substr('abcde',3,2) from lxw_dual;

cd

#> select substring('abcde',3,2) from lxw_dual;

cd

```
#>select substring('abcde',-2,2) from lxw_dual;
```

de

5.57 字符串转小写函数：Lower,lcase

语法: lower(string A) lcase(string A)

返回值: string

说明：返回字符串 A 的小写格式

举例：

```
#> select lower('abSEd') from lxw_dual;
```

absed

```
#> select lcase('abSEd') from lxw_dual;
```

absed

5.58 去空格函数：Trim

语法: trim(string A)

返回值: string

说明：去除字符串两边的空格

举例：

```
#> select trim(' abc ') from lxw_dual;
```

abc

5.59 左边去空格函数：Ltrim

语法: ltrim(string A)

返回值: string

说明：去除字符串左边的空格

举例：

#> select ltrim(' abc ') from lxw_dual;

abc

5.60 右边去空格函数：Rtrim

语法: rtrim(string A)

返回值: string

说明：去除字符串右边的空格

举例：

#> select rtrim(' abc ') from lxw_dual;

abc

5.61 正则表达式替换函数：Regexp_replace

语法: regexp_replace(string A, string B, string C)

返回值: string

说明：将字符串 A 中的符合 java 正则表达式 B 的部分替换为 C。注意，在有些情况下要使用转义字符,类似 oracle 中的 regexp_replace 函数。

举例：

```
#> select regexp_replace('foobar', 'oo|ar', '') from lxw_dual;

fb
```

5.62 正则表达式解析函数：Regexp_extract

语法: regexp_extract(string subject, string pattern, int index)

返回值: string

说明：将字符串 subject 按照 pattern 正则表达式的规则拆分，返回 index 指定的字符。

举例：

```
#> select regexp_extract('foothebar', 'foo(.*?)(bar)', 1)
fromlxw_dual;

the

#> select regexp_extract('foothebar', 'foo(.*?)(bar)', 2)
fromlxw_dual;

bar

#> select regexp_extract('foothebar', 'foo(.*?)(bar)', 0)
fromlxw_dual;

foothebar
```

注意,在有些情况下要使用转义字符,下面的等号要用双竖线转义,这是 java 正则表达式的规则。

```
select data_field,

        regexp_extract(data_field,'.*?bgStart\\=([^&]+)',1) as
aaa,

        regexp_extract(data_field,'.*?contentLoaded_headStart\\=(
[^&]+)',1) as bbb,

        regexp_extract(data_field,'.*?AppLoad2Req\\=([^&]+)',1)
as ccc

        from pt_nginx_loginlog_st

        where pt = '2012-03-26'limit 2;
```

5.63 URL 解析函数：Parse_url

语法: parse_url(string urlString, string partToExtract [, stringkeyToExtract])

返回值: string

说明:返回 URL 中指定的部分。partToExtract 的有效值为:HOST, PATH, QUERY, REF, PROTOCOL, AUTHORITY, FILE, and USERINFO.

举例:

```
#>
selectparse_url('http://facebook.com/path1/p.php?k1=v1&k2=v2#Ref1',
'HOST') fromlxw_dual;

facebook.com

#>
selectparse_url('http://facebook.com/path1/p.php?k1=v1&k2=v2#Ref1',
'QUERY','k1') from lxw_dual;

v1
```

5.64 Josn 解析函数：Get_json_object

语法: get_json_object(string json_string, string path)

返回值: string

说明：解析 json 的字符串 json_string,返回 path 指定的内容。如果输入的 json 字符串无效，那么返回 NULL。

举例：

```
#> select    get_json_object('{"store":
>      {"fruit":\[{"weight":8,"type":"apple"},{"weight":9,"type":"p
ear"}],
>        "bicycle":{"price":19.95,"color":"red"}
>      },
>    "email":"amy@only_for_json_udf_test.net",
>    "owner":"amy"
> }
> ','$.owner') from lxw_dual;
amy
```

5.65 空格字符串函数：Space

语法: space(int n)

返回值: string

说明：返回长度为 n 的字符串

举例：

```
#> select space(10) from lxw_dual;

#> select length(space(10)) from lxw_dual;

10
```

5.66 重复字符串函数：Repeat

语法: repeat(string str, int n)

返回值: string

说明：返回重复 n 次后的 str 字符串

举例：

```
#> select repeat('abc',5) from lxw_dual;

abcabcabcabcabc
```

5.67 首字符 ascii 函数：Ascii

语法: ascii(string str)

返回值: int

说明：返回字符串 str 第一个字符的 ascii 码

举例：

```
#> select ascii('abcde') from lxw_dual;

97
```

5.68 左补足函数：lpad

语法: lpad(string str, int len, string pad)

返回值: string

说明：将 str 进行用 pad 进行左补足到 len 位

举例：

```
#> select lpad('abc',10,'td') from lxw_dual;
tdtdtdtabc
```

注意：与 GP，ORACLE 不同，pad 不能默认

5.69 右补足函数：Rpad

语法: rpad(string str, int len, string pad)

返回值: string

说明：将 str 进行用 pad 进行右补足到 len 位

举例：

```
#> select rpad('abc',10,'td') from lxw_dual;
abctdtdtdt
```

5.70 分隔字符串函数：Split

语法: split(string str, stringpat)

返回值: array

说明: 按照 pat 字符串分割 str，会返回分割后的字符串数组

举例：

```
#> select split('abtcdtef','t') from lxw_dual;
["ab","cd","ef"]
```

5.71 集合查找函数：Find_in_set

语法: find_in_set(string str, string strList)

返回值: int

说明: 返回 str 在 strlist 第一次出现的位置，strlist 是用逗号分割的字符串。如果没有找该 str 字符，则返回 0

举例：

```
#> select find_in_set('ab','ef,ab,de') from lxw_dual;

2

#> select find_in_set('at','ef,ab,de') from lxw_dual;

0
```

5.72 个数统计函数：Count

语法: count(*), count(expr), count(DISTINCT expr[, expr_.])

返回值: int

说明: count(*)统计检索出的行的个数，包括 NULL 值的行；count(expr)返回指定字段的非空值的个数；count(DISTINCTexpr[, expr_.])返回指定字段的不同的非空值的个数

举例：

```
#> select count(*) from lxw_dual;

20

#> select count(distinct t) from lxw_dual;
```

10

5.73 总和统计函数：Sum

语法: sum(col), sum(DISTINCT col)

返回值: double

说明: sum(col)统计结果集中 col 的相加的结果; sum(DISTINCT col)统计结果中 col 不同值相加的结果

举例：

#> select sum(t) from lxw_dual;

100

#> select sum(distinct t) from lxw_dual;

70

5.74 平均值统计函数：Avg

语法: avg(col), avg(DISTINCT col)

返回值: double

说明: avg(col)统计结果集中 col 的平均值; avg(DISTINCT col)统计结果中 col 不同值相加的平均值

举例：

#> select avg(t) from lxw_dual;

50

#> select avg (distinct t) from lxw_dual;

30

5.75 最小值统计函数：Min

语法：min(col)

返回值：double

说明： 统计结果集中 col 字段的最小值

举例：

#> select min(t) from lxw_dual;

20

5.76 最大值统计函数：Max

语法：maxcol)

返回值：double

说明： 统计结果集中 col 字段的最大值

举例：

#> select max(t) from lxw_dual;

120

5.77 非空集合总体变量函数：Var_pop

语法：var_pop(col)

返回值：double

说明： 统计结果集中 col 非空集合的总体变量（忽略 null）

5.78 非空集合样本变量函数：Var_samp

语法：var_samp (col)

返回值：double

说明： 统计结果集中 col 非空集合的样本变量（忽略 null）

5.79 总体标准偏离函数：Stddev_pop

语法：stddev_pop(col)

返回值：double

说明： 该函数计算总体标准偏离，并返回总体变量的平方根，其返回值与 VAR_POP 函数的平方根相同

5.80 样本标准偏离函数：Stddev_samp

语法：stddev_samp (col)

返回值：double

说明： 该函数计算样本标准偏离

5.81 中位数函数：Percentile

语法：percentile(BIGINT col, array(p1 [, p2]…))

返回值：array<double>

说明： 功能和上述类似，之后后面可以输入多个百分位数，返回类型也为 array<double>，其中为对应的百分位数。

举例：

select percentile(score,<0.2,0.4>) from lxw_dual;　取 0.2，0.4 位置的数据

5.82 近似中位数函数：percentile_approx

语法: percentile_approx(DOUBLE col, p [, B])

返回值: double

说明:　求近似的第 pth 个百分位数，p 必须介于 0 和 1 之间，返回类型为 double，但是 col 字段支持浮点类型。参数 B 控制内存消耗的近似精度，B 越大，结果的准确度越高。默认为 10,000。当 col 字段中的 distinct 值的个数小于 B 时，结果为准确的百分位数

5.83 近似中位数函数：percentile_approx

语法: percentile_approx(DOUBLE col, array(p1 [, p2]…) [, B])

返回值: array<double>

说明:　功能和上述类似，之后后面可以输入多个百分位数，返回类型也为 array<double>，其中为对应的百分位数。

5.84 直方图：histogram_numeric

语法: histogram_numeric(col, b)

返回值: array<struct { ‘x’ , ‘y’ }>

说明:　以 b 为基准计算 col 的直方图信息。

举例：

#> select histogram_numeric(100,5) from lxw_dual;

[{"x":100.0,"y":1.0}]

5.85 字段类型转换：lDateToChar

数据中 long 型的时间，转换为字符串：lDateToChar

语法：lDateToChar(A_timestamp,'yyyy-MM-dd')

返回值：String

示例：select lDateToChar(A_timestamp,'yyyy-MM-dd') from test

5.86 字段类型转换：strDateToLong

数据将字符串类型的时间，转换为 long

语法：strDateToLong('2016061520','yyyyMMddHH')

返回值：long

示例：select a from test where A_timestamp between strDateToLong('2016061520','yyyyMMddHH') and strDateToLong('2016061521','yyyyMMddHH')

ALEIYE

让 | 大 | 数 | 据 | 更 | 简 | 单

公司地址：北京市西城区新街口外大街28号普天德胜大厦A座405

邮　　编：100088

联系电话：010-82053991

电子邮箱：service@ALEIYE.cn