Optimizing Your Search Experience

How-To Webinar

sumologic

August 2016
Customer Success

Agenda

- Basic Search Structure
- Setting Search Performance Expectations
- Search Optimization Tools
  - Field Extraction Rules
  - Partitions
  - Scheduled Views
- Demo
- Q&A

Basic Search Structure

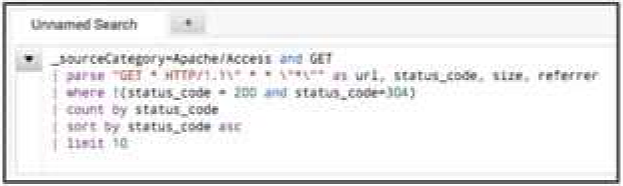
Search Structure

Keywords and operators (separated by pipes) that build on top of each other

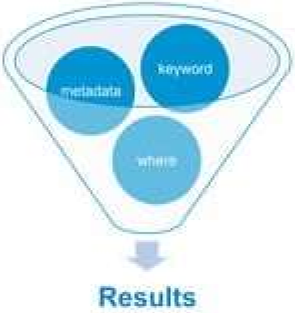
Syntax:

metadata tags + keywords | parse | filter | aggregate | sort | limit

Example Search:



```
_sourceCategory=Apache/Access and GET
| parse "GET * HTTP/1.1\" * * \"*\"" as url, status_code, size, referrer
| where !(status_code = 200 and status_code=304)
| count by status_code
| sort by status_code asc
| limit 10
```



Metadata Fields

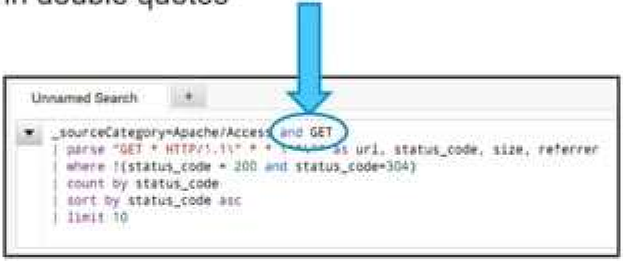
- All messages are tagged during data ingest
- Metadata fields are configured as part of Collector and Source setup

Name	Description
_collector	Name of collector when installed
_source	Name of the source defined during configuration
_sourceHost	The host name of the source
_sourceName	The name of the log file (including path)
_sourceCategory	Category associated with the source

- Properly categorizing your data leads to more efficient searches
- Good Source Category, Bad Source Category

Keyword Search

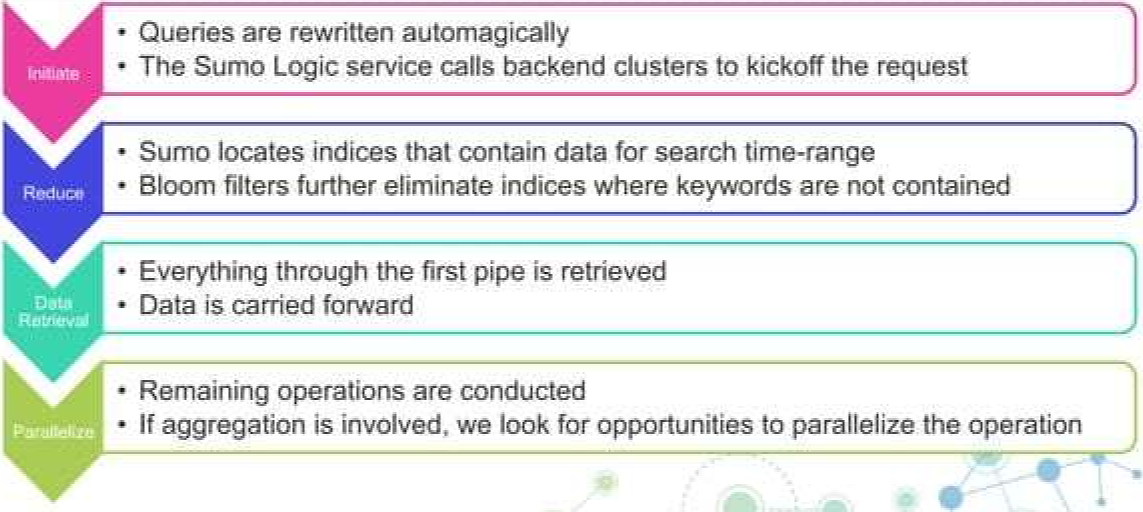
- Case Insensitive unless string is in double quotes
- Wildcard Support (e.g. ERR*)
- Boolean Logic Support
  - AND
  - OR
  - !(A OR B)



```
_sourceCategory=Apache/Access and GET
| parse "GET * HTTP/1.1\" * * [illegible] as uri, status_code, size, referrer
| where !(status_code = 200 and status_code=304)
| count by status_code
| sort by status_code asc
| limit 10
```

- Combine keywords with metadata fields for the best performance
- Bloom filters
  - Using keywords helps bloom filters retrieve data very quickly

Processing Your Search Request



- Queries are rewritten automagically
- The Sumo Logic service calls backend clusters to kickoff the request

- Sumo locates indices that contain data for search time-range
- Bloom filters further eliminate indices where keywords are not contained

- Everything through the first pipe is retrieved
- Data is carried forward

- Remaining operations are conducted
- If aggregation is involved, we look for opportunities to parallelize the operation

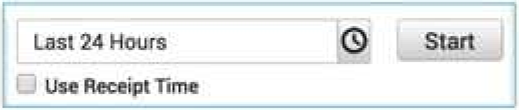
Develop Good Search Habits

- Use metadata and keyword combinations to reduce scope
- Shorten your time-range down as much as possible
- Limit result sets before aggregating data → where user=a | count by user
- Use parse anchor instead of parse regex for structured messages
- Avoid the use of expensive parse regex tokens like .* → \d{2,10}
- Add line breaks after each operation
- Use pre-extracted fields where possible (to be discussed later)

Search Performance Expectations

The Time Range Effect

- More recent data can be accessed quickly
  - We do something special when scanning the last 24 hours of events
  - Why? Over 90% of searches are executed against recent data

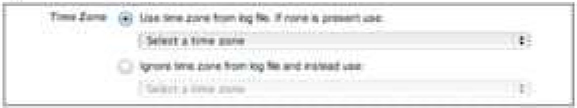

Last 24 Hours

□ Use Receipt Time

- Test queries on very recent data first before saving or publishing
- Our main performance metric (speedup) is essentially a ratio that divides the time-range used by the time it takes for data to return.

Review Your Data Source Time Zone Settings

- Leads to a large gap between message time and receipt time
- Causes backend fragmentation and can affect search speed
- Support of Java 6 Time Zone formats
  - Pacific Standard Time; PST; GMT-08:00
  - -0800
  - **<u>NOT</u>** US/Pacific



Time Zone ● Use time zone from log file. If none is present use:

Select a time zone

○ Ignore time zone from log file and instead use:

Select a time zone

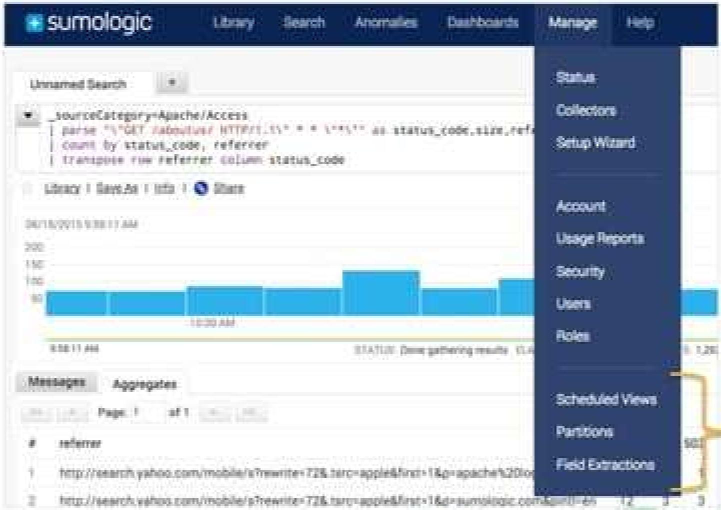
- Data integrity will be questioned by users
- Knowledge Base Article: [Large Time Discrepancies]

Compute-Intensive Operations

- Multiple .* tokens in a single parse regex statement
- Parse using public library (apache/access, iis, cisco/asa, windows/2008)
  - Try to borrow from Field Extraction Rule templates
- LogCompare and LogReduce
- Join
  - Time to run exponentially increases when extending your time-range
- Transaction, Transactionize and Merge
  - Try and limit the 'timewindow' parameter for finding corresponding events
- Outlier / Predict

Performance Optimization Tools

Managing Search Optimization Tools

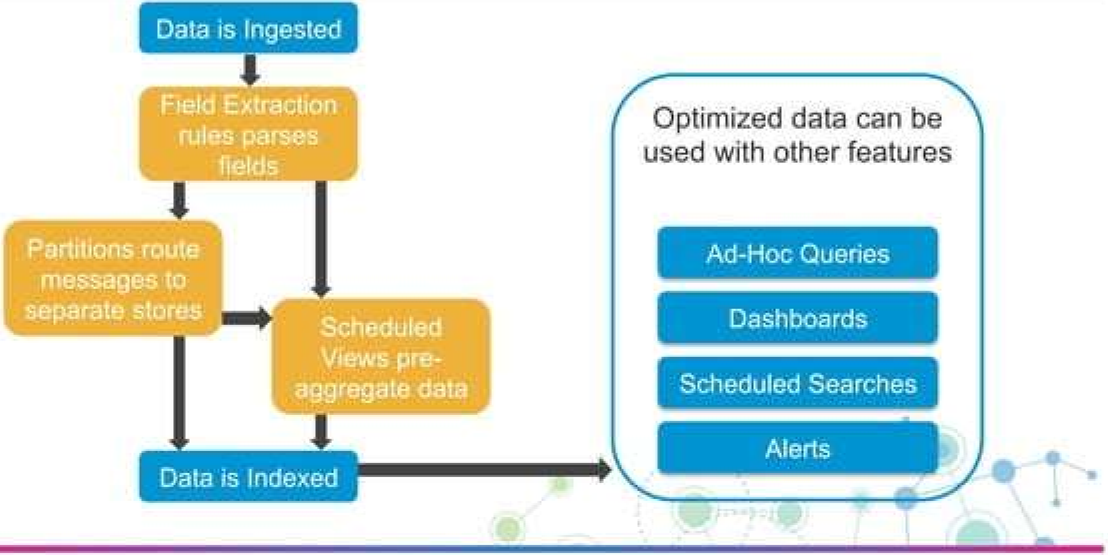


```
_sourceCategory=Apache/Access
| parse "\"GET /aboutus/ HTTP/1.1\" * * \"*\"" as status_code,size,ref
| count by status_code, referrer
| transpose row referrer column status_code
```

#	referrer
1	http://search.yahoo.com/mobile/s?rewrite=72&.tsrc=apple&first=1&p=apache%20lo
2	http://search.yahoo.com/mobile/s?rewrite=72&.tsrc=apple&first=1&p=sumologic.com

These tools are only available to Administrators

How Data is Optimized for Search


Data is Ingested
Field Extraction rules parses fields
Partitions route messages to separate stores
Scheduled Views pre-aggregate data
Data is Indexed
Optimized data can be used with other features
Ad-Hoc Queries
Dashboards
Scheduled Searches
Alerts

Field Extraction Rules

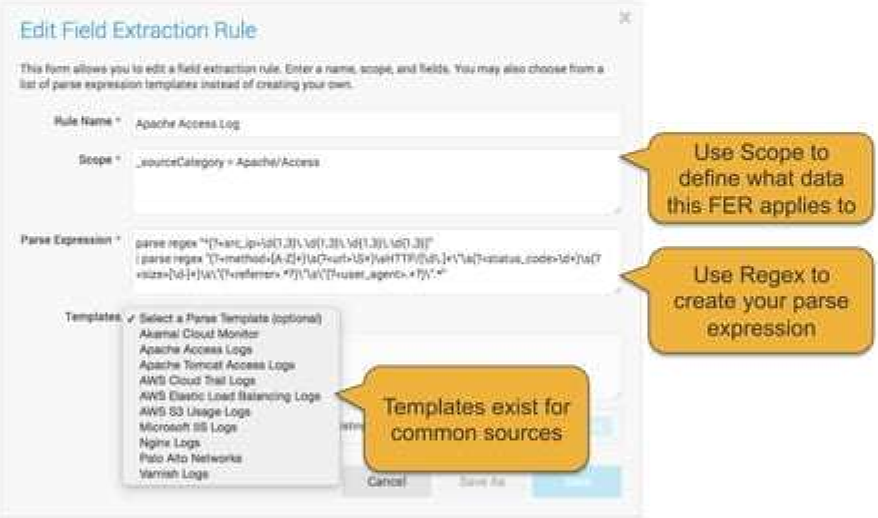
Benefits of Field Extraction Rules

- Extract fields at the time of ingest
- Standardize Searches and Field Names for users
- Simplify searches
  - Narrow results within search scope instead of using 'where' operator (e.g. _sourceCategory=nginx status_code=404)
- Improves Search Performance
  - Eliminates the need to 'parse' data at run time

When to Use Field Extraction Rules

- The same (or very similar) parse statement is being used over and over
- Parsing over a large volume of data
- Constantly filtering data based off of parsed fields
- Disparate logs need to be joined using a Unique ID
  - Session ID
  - User Name
  - Process ID
- Syslog Metadata Overrides

Create Field Extraction Rule



Edit Field Extraction Rule

This form allows you to edit a field extraction rule. Enter a name, scope, and fields. You may also choose from a list of parse expression templates instead of creating your own.

Rule Name * Apache Access Log

Scope * _sourceCategory = Apache/Access

Parse Expression *

```
parse regex "^(?<src_ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
| parse regex "(?<method>[A-Z]+)\s(?<url>\S+)\sHTTP/[\d\.]+"\s(?<status_code>\d+)\s(?<size>[\d-]+)\s\"(?<referrer>.*?)\"\s\"(?<user_agent>.+?)\".*"
```

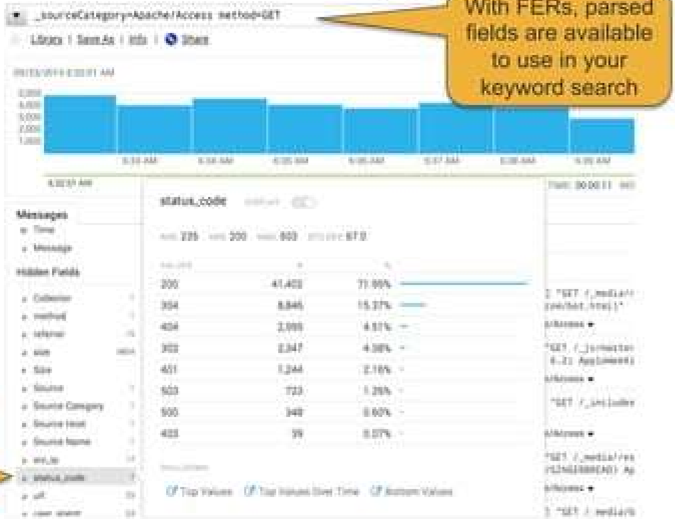
Templates ✓ Select a Parse Template (optional)

Use Scope to define what data this FER applies to

Use Regex to create your parse expression

Templates exist for common sources

Using Pre-Parsed Fields When Querying



```
_sourceCategory=Apache/Access method=GET
```

Messages
- Time
- Message

Hidden Fields
- Collector
- method
- referrer
- size
- Size
- Source
- Source Category
- Source Host
- Source Name
- src_ip
- status_code
- url
- user_agent

status_code

235 200 503 87.0

status_code	#	%
200	41,403	71.99%
304	8,846	15.37%
404	2,595	4.51%
302	2,347	4.08%
451	1,244	2.16%
503	733	1.27%
500	348	0.60%
403	39	0.07%

With FERs, parsed fields are available to use in your keyword search

Parsed fields are available in your Field Browser for further analysis

Field Extraction Rule Recommendations

- Test the rule by running a search over a small time-range that has data
- The scope and parse statement should not change
- Ensure your rule covers common searches
- Only extract the minimum fields necessary
    - Use 'fields' operator to limit results

FER Caveats

- Max of 50 Rules
- Max of 200 Total Fields
- Supported Operators
  - Parse Anchor / Regex / Nodrop
  - Double
  - Fields
  - Num
  - If
  - Where
  - Concat
  - Keyvalue (not 'kv auto')
  - NEW! JSON (not 'json auto')

NOTE: Deleted rules and fields defined in them will still count towards the max

Partitions

Benefits of Partitions

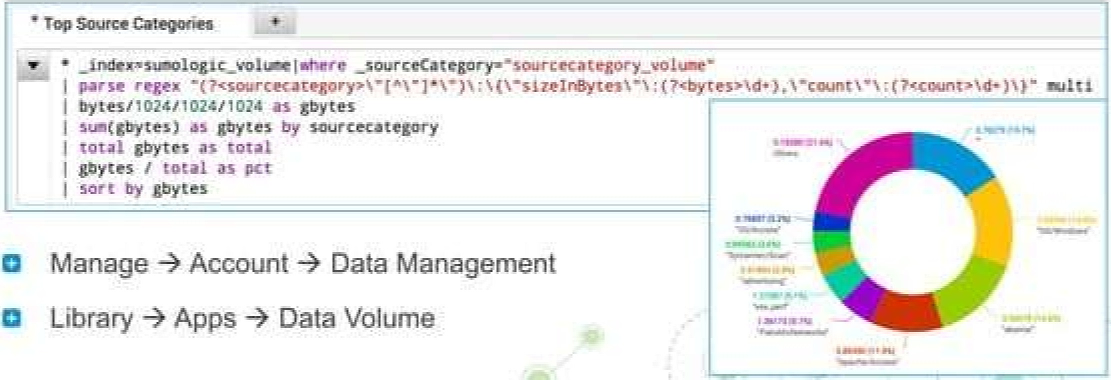
- Divides your data into smaller chunks to be searched on
- Takes advantage of your source categorization; similar data can actually be grouped together
- Improves performance when used in searches
- It can eliminate the need for lengthy scope definitions

When to Create Partitions

- Sets of data are being searched in isolation
- A large amount of data being sent daily (> 5 GB's)
  - Navigate to Manage → Account if you don't know
- Different groups are focused on specific logs
- RBAC filtering is required for data provisioning

Use Data Volume Index

- Helps to determine possible ways to partition data
- Recommended partition size → Up to 30% of data volume

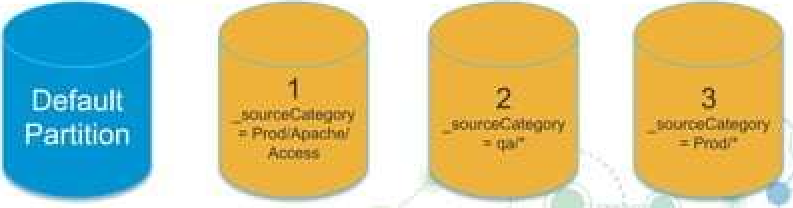


```
* _index=sumologic_volume|where _sourceCategory="sourcecategory_volume"
| parse regex "(?<sourcecategory>\"[^\"]*\")\:\{\"sizeInBytes\"\:(?<bytes>\d+),\"count\"\:(?<count>\d+)\}" multi
| bytes/1024/1024/1024 as gbytes
| sum(gbytes) as gbytes by sourcecategory
| total gbytes as total
| gbytes / total as pct
| sort by gbytes
```

- Manage → Account → Data Management
- Library → Apps → Data Volume

How Search Scans Partitions

Sample Search Scope	Partition(s) Searched
_sourceCategory=*apache*	→ Default Partition
_source=Prod/Nginx	→ ALL
_sourceCategory=Prod/Apache/Access	→ Partitions 1 OR 3
_index=1 _sourceName=sample.log	→ Partition 1



Partitions Caveats

- Overlapping data between partitions are counted towards your contracted data volume quota
- Maximum of 500 indexes can be created with no available overrides
- Data cannot be backfilled
- Not editable after creation

Scheduled Views

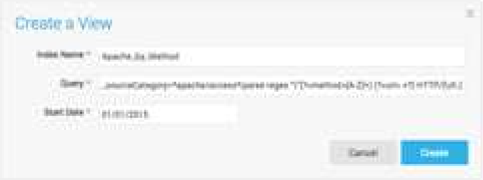
Benefits of Scheduled Views

- Similar to relational DB materialized views
- Allows you to pre-aggregate data
- Allows for long term trending analysis
- Can significantly improve performance for high selectivity queries
  - (_source=A or _source=B) and _sourceName=C and keyword1 and keyword2
- Unlike partitions, data can be backfilled

When to Use Scheduled Views

- Specific aggregate operators are used heavily in queries
  - Count
  - Sum
- Data is being trended over a long period of time (e.g. Last 30 Days)
  - Failed logins on critical servers
  - Number of 404 errors
- A highly selective query does not perform well

Scheduled Views Recommendations



Create a View

Index Name *: Apache_by_Method

Query *: _sourceCategory=*apache/access* | parse regex [illegible]

Start Date *: 01/01/2015

- Include aggregation
- Timeslice 1m
- Use queries that are not likely to change
- Take advantage of existing partitions and FER's
- Only backfill data needed for analyses

Scheduled View Caveats

- Data in scheduled views are counted towards your quota
- Parsed fields in views count towards field extraction limitation (200)
- Data can only be backfilled through your plan's retention period
- Not editable after creation
- Supported aggregate operators
  - Difference
  - Count
  - Sum

Quick Review

Review: Factors in Search Performance

- Query structure
  - Data Selectivity (keywords, metadata fields)
  - Heavy Operations (join, transaction, summarize)
- Search Time Range
- Possible Time Zone Misconfiguration at Source Level
- Total Data Volume for Account
- Use of Performance Optimization Tools
- Service Anomalies

Review: Search Optimization Tools

What I want to do is	Partition	Scheduled View	Field Extraction
Parse the same type of log message repeatedly			✓
Identify long-term trends		✓	
Group related data together	✓		
Pre-compute or aggregate data before querying		✓	
Use RBAC to deny or grant access to the data	✓		

customer-success@sumologic.com

optimize@sumologic.com