



[続] Elasticsearch/Kibana のパイプ型 クエリ言語ES|QLの紹介 ~ Pandas と一緒にログ分析してみた

Elasticsearch株式会社

ソリューション・アーキテクト 関屋信彦

本セッションの内容と対象者

- オンデマンドセッションの方のES|QLの紹介のおさらい + 続編としてPandasとES|QLの使い方のお話

対象者

- Elastic Kibanaを使ってデータ分析、ログ分析をしている人
- Elastic Kibanaの最新のデータ分析機能について知りたい人
- Elastic のAIアシスタントでのクエリー生成について知りたい人

ES|QL = Elasticsearch Query Language

v8.11にTech Previewで登場

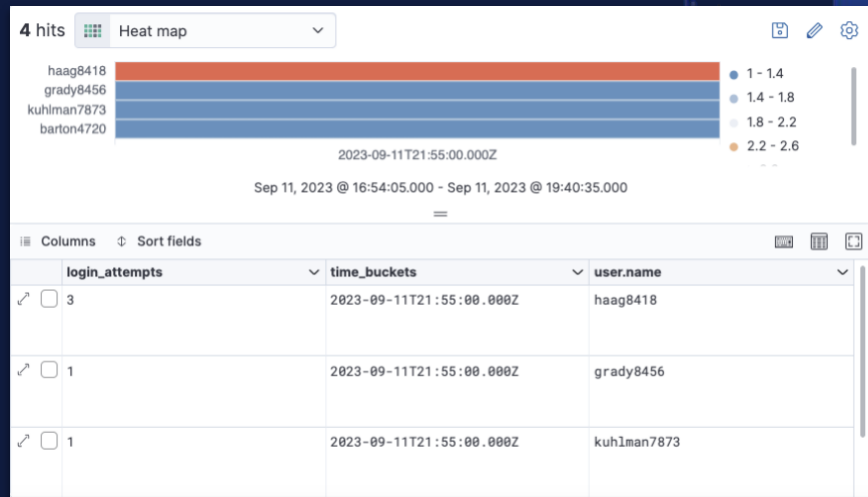
v8.14にGAリリース

License: Basic

セルフインストール版/Elastic Cloud版 共に利用可能

ES|QL のクエリーの例

```
FROM apache-logs
| WHERE url.original == '/login'
| EVAL time_buckets = auto_bucket (@timestamp,
50, "2023-09-11T21:54:05.000Z", "2023-09-
12T00:40:35.000Z")
| STATS login_attempts = count(user.name) by
time_buckets, user.name
| SORT login_attempts desc
```



上記の検索の内容 : Apacheのアクセスログを分析し、アクセスURLが"/login"のログに関して、特定の期間の中でのユーザー名別でのログインカウントを集計し、ログインの多いユーザーとその時間帯を表示する (簡単に言うと、ログインアクセスが集中した時間とユーザー名を特定したい)

強力、コンポーネント化、拡張可能、高速

from logs

まずシンプルに
データを閲覧し
たい

テーブル形式の
フォーマットで
表示

@timestamp	bytes_in	bytes_out	client_ip	dest_ip	status
2017-11-10T21:15:54.000Z	47	388	10.0.1.1	172.27.1.129	OK
2017-11-10T21:15:39.000Z	29	374	10.0.1.1	172.27.1.123	OK
2017-11-10T21:15:39.000Z	35	303	10.0.1.1	172.27.1.113	OK
2017-11-10T21:15:39.000Z	36	312	10.0.1.1	172.27.1.116	OK
2017-11-10T21:15:40.000Z	35	344	10.0.1.1	172.27.1.149	OK
2017-11-10T21:15:40.000Z	31	503	10.0.1.1	172.27.1.122	OK
2017-11-10T21:15:40.000Z	35	458	10.0.1.7	172.27.1.140	OK
2017-11-10T21:15:41.000Z	35	281	10.0.1.8	null	OK

```
from logs
| where status != "OK"
```

新しいコマンド
を追加して結果
を絞り込み

SQL同様の
記述子

@timestamp	bytes_in	bytes_out	client_ip	dest_ip	status
2017-11-10T20:29:25.000Z	32	null	10.0.1.177	172.20.10.1	Error
2017-11-10T21:34:49.000Z	28	null	null	27.58.6.220	Error
2017-11-10T20:33:06.000Z	28	null	10.0.1.166	172.27.1.1	Error
2017-11-10T20:06:49.000Z	30	null	10.0.0.147	90.128.199.24	Error
2017-11-10T19:51:38.000Z	28	null	10.0.0.130	203.131.98.151	Error
2017-11-10T19:51:38.000Z	28	null	10.0.0.107	172.20.10.8	Error
2017-11-10T19:51:38.000Z	28	null	10.0.0.130	203.131.98.151	Error

```
from logs
| where status != "OK"
| stats c = count() by dest_ip
```

sum, avg, min,
maxの他、多数の
関数をサポート

c	dest_ip
1	27.58.6.220
1	90.128.199.24
1	172.20.10.1
1	172.20.10.8
1	172.27.1.1
2	203.131.98.151

ES|QLのUX

データ探索、変換、
視覚化をすべて
1つに統合

ES|QL

```
1 from logstash-* |
2 stats avg_bytes = avg(bytes) by geo.src |
3 eval avg_bytes_kb = round(avg_bytes/1024, 2) |
4 enrich geo-data on geo.src with country, continent |
5 keep avg_bytes_kb, geo.src, country, continent
```

5 lines @timestamp detected

Search field names 0 27 hits Reset search

Selected fields 4

- avg_bytes_kb
- geo.src
- country
- continent

Available fields 4

- avg_bytes_kb
- continent
- country
- geo.src

Columns Sort fields

avg_bytes_kb	geo.src	country
8.84	BD	Bangladesh
6.02	JP	Japan

結果 (チャート)

Sep 11, 2023 @ 15:54:05.000 - Sep 11, 2023 @ 18:40:35.000

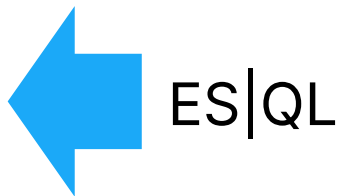
結果 (表)



Observability

ES|QLはどのタイプのデータに対して
利用可能！

- ログ
- メトリック
- トレース



ES|QL

アラートルールとしても利用可能です

```
1 from metrics* |
2 stats max_cpu = max(kubernetes.pod.cpu.usage.node.pct),
   avg_mem = max(kubernetes.pod.memory.usage.bytes) by
   kubernetes.pod.name |
3 sort max_cpu desc | limit 10
```

3 lines @timestamp detected Run query ⌘ + Enter

	max_cpu	avg_mem	kubernetes.pod.name
<input type="checkbox"/>	-	-	-
<input type="checkbox"/>	0.125	945872896	heartbeat-synthetics-6c9497b68-pljxr
<input type="checkbox"/>	0.117	943742976	heartbeat-synthetics-tokyo-5b9f74dd57-27h1v
<input type="checkbox"/>	0.099	2220580864	relevance-workbench-app-ui-f7cbd657c-dpd7d
<input type="checkbox"/>	0.097	1999900672	elastic-agent-cxjv4
<input type="checkbox"/>	0.09	232505344	kafka-loadgen-deco-green-5cf8cc7988-pxcnp



Security

複雑な検知ルールをES|QLで定義し、
不審な動きをログから見つけ出す

```
1 //This query counts the number of outbound connections made to external IP
  addresses broken down by user and host. It uses a case statement to add a new
  field called "follow_up". If the sum of connections is greater or equal to 100,
  the value of the follow_up field is set to true. It also enriches the user names
  with their respective ldap groups.
2
3 FROM logs-*
4 | WHERE NOT CIDR_MATCH(destination.ip, "10.0.0.0/8", "172.16.0.0/12", "192.168.0.0/
  16")
5 | STATS destcount = COUNT(destination.ip) by user.name, host.name
6 | ENRICH ldap_lookup_new ON user.name
7 | WHERE group.name IS NOT NULL
8 | EVAL follow_up = CASE(
9 |   destcount >= 100, "true",
10 |   "false")
11 | SORT destcount desc
12 | KEEP destcount, host.name, user.name, group.name, follow_up
```

12 lines @timestamp detected Run query * + Enter

4 hits [Reset search](#)

Columns Sort fields

	destcount	host.name	user.name	group.name	follow_up
<input type="checkbox"/>	213	omm-win-detect	Administrator	local_admins	true
<input type="checkbox"/>	127	omm-win-detect	SYSTEM	system_users	true
<input type="checkbox"/>	98	omm-win-prevent	SYSTEM	system_users	false
<input type="checkbox"/>	86	omm-win-prevent	Administrator	local_admins	false



Search

Elasticsearchに保存されているビジネスデータの分析をもっと手軽に。

ElasticsearchのAPIでもES|QLで検索。

```
1 from kibana_sample_data_ecommerce
2 | where products.base_price >15 and geoip.city_name == "New York"
3 | stats avgbaseprice = avg(products.base_price) by category, day_of_week
```

3 lines @timestamp not detected Run query * + Enter

Columns	Sort fields	
avgbaseprice	category	day_of_week
↗ <input type="checkbox"/> 65	Women's Clothing	Sunday
↗ <input type="checkbox"/> 60	Women's Clothing	Monday
↗ <input type="checkbox"/> 60.833333333333336	Women's Clothing	Tuesday
↗ <input type="checkbox"/> 33	Men's Clothing	Wednesday
↗ <input type="checkbox"/> 61.25	Women's Clothing	Thursday
↗ <input type="checkbox"/> 67.5	Women's Clothing	Friday
↗ <input type="checkbox"/> 65	Women's Clothing	Wednesday



ElasticのAIアシスタントで のES|QL活用パターン

AI Assistant

会話からES|QLを作成。そしてその場で実行。

The screenshot shows a chat interface with three messages:

- User:** Visualize the average memory usage of my kubernetes pods using lens, data is in metrics* and mapped to ECS
- Elastic Assistant:** requested the function visualize_query. It provides an ES|QL query:

```
FROM metrics-*
| WHERE kubernetes.pod.name IS NOT NULL
| STATS avg_memory_usage = AVG(kubernetes.pod.memory.usage.node.pct) BY kubernetes.pod.name
| SORT avg_memory_usage DESC
```
- Elastic Assistant:** executed the function visualize_query. It displays a bar chart showing average memory usage for two pods.

Kubernetesのメモリの平均値の表示をAIに指示

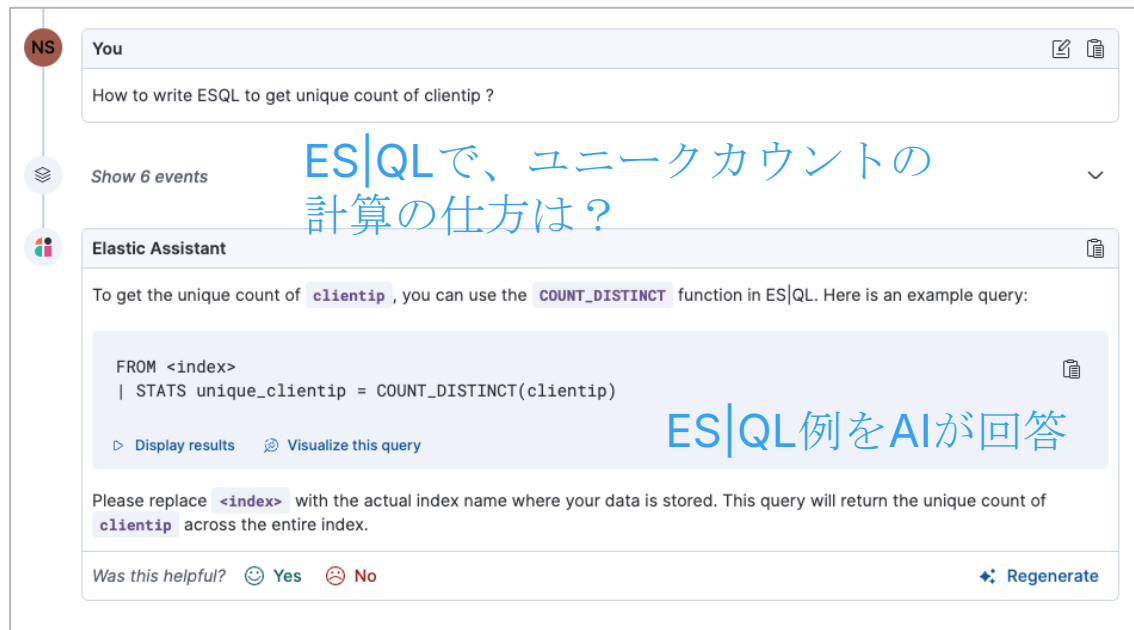
ES|QLをAIが回答

kubernetes.pod.name	avg_memory_usage
otel-demo-green-3-shippingservice-Sb4d84dcc4-2lh2x	0.18
metrics-server-fbb469ccc-fqtkm	0.16

AI Assistant

ES|QLの書き方を質問
してみましょう

(一般のLLMにはまだ存在
しないES|QLの知識を回
答できます)



The screenshot shows a chat interface with a user asking a question and an AI assistant providing an answer. The user's message is: "How to write ES|QL to get unique count of clientip?". The AI assistant's response includes a sample ES|QL query: `FROM <index> | STATS unique_clientip = COUNT_DISTINCT(clientip)`. The response also includes instructions to replace the placeholder index name and a feedback prompt.

You

How to write ES|QL to get unique count of clientip ?

Show 6 events

ES|QLで、ユニークカウントの計算の仕方は？

Elastic Assistant

To get the unique count of `clientip`, you can use the `COUNT_DISTINCT` function in ES|QL. Here is an example query:

```
FROM <index>
| STATS unique_clientip = COUNT_DISTINCT(clientip)
```

ES|QL例をAIが回答

Display results Visualize this query

Please replace `<index>` with the actual index name where your data is stored. This query will return the unique count of `clientip` across the entire index.

Was this helpful? Yes No Regenerate

更なるデータ分析と自動化へ！
Pandasと**ES|QL**を一緒に使う

PandasとESQLを一緒に使う方法

```
PandasToESQL.ipynb ☆
ファイル 編集 表示 挿入 ランタイム ツール ヘルプ すべての変更を保存しました

+ コード + テキスト

[21] !pip install -q elasticsearch==8.14 pandas

from elasticsearch import Elasticsearch
from elasticsearch.helpers import bulk
from getpass import getpass # standard library module for secure credential input

# https://www.elastic.co/search-labs/tutorials/install-elasticsearch/elastic-cloud#f
ELASTIC_CLOUD_ID = getpass("Elastic Cloud ID: ")

# https://www.elastic.co/search-labs/tutorials/install-elasticsearch/elastic-cloud#cr
ELASTIC_API_KEY = getpass("Elastic API Key: ")

# Create the client instance
client = Elasticsearch(
    cloud_id=ELASTIC_CLOUD_ID,
    api_key=ELASTIC_API_KEY
)

Elastic Cloud ID: .....
Elastic API Key: .....
```

```
from io import StringIO
import pandas as pd

query = 'FROM kibana_sample_data_flights | WHERE DestCityName == ? | LIMIT 3'
response = client.esql.query(
    query=query,
    format="csv",
    params=["Sydney"]
)
df = pd.read_csv(StringIO(response.body), parse_dates=['timestamp'])
display(df)
```

	AvgTicketPrice	Cancelled	Carrier	Dest	DestAirportID	DestCityName	DestCountry	DestLocation	DestRegion	DestWeather
0	841.265625	False	Kibana Airlines	Sydney Kingsford Smith International Airport	SYD	Sydney	AU	POINT (151.177002 -33.94609833)	SE-BD	Rain
1	355.668701	False	Kibana Airlines	Sydney Kingsford Smith International Airport	SYD	Sydney	AU	POINT (151.177002 -33.94609833)	SE-BD	Rain
2	750.148254	False	Kibana Airlines	Sydney Kingsford Smith International Airport	SYD	Sydney	AU	POINT (151.177002 -33.94609833)	SE-BD	Damaging Wind

3 rows x 27 columns

どんな時にPandas
と一緒に使いたい
か



複数のES|QLクエリーの結
果を組み合わせたとき



Jupyter Notebook等を使
い、バッチで複数のクエリ
ーを流したい



プログラムで動的にES|QL
クエリーを作りたい

複数のES|QLクエリーの結果を組み合わせたとき

こんなとき

- 同じIndex内のES|QLクエリー結果同士の比較や計算
例：現在と1時間前のメトリックデータのそれぞれのCPU利用率の差分を表示
- 最初のIndex AからのES|QLクエリー結果を、Index Bに対するクエリーの条件とする
例：アクセスログからユーザーAのログイン時刻を取得し、その後の5分間のアプリケーションログを表示

Jupyter NotebookからES|QLを使いたい

こんなとき

- Python/Jupyter Notebookが得意な高度分析にElasticsearchのデータを使いたい
- 定型のクエリー集を一気に流したい
- トラブルシューティング用のプレイブックとして使う
- クエリー結果をファイルに残したい

プログラムで動的にES|QLクエリーを作りたい

こんなとき

- 最初のES|QLクエリー結果や、その他データから得られたリストに対して、ループ処理でES|QLを一気に実行したい

例：ITの変更業務の変更履歴のログを元に、システム別の月間の統計とともに、履歴データをレポートとしてまとめたい

負荷のかけすぎに注意！

バッチで多くのクエリーを短期間で実行する場合、本番のElasticのデプロイメントとは別のデプロイメントを使うことを検討しましょう。

Elastic Cloudの場合は簡単！

1. 新しいElasticのデプロイメントを作成
2. データが入っている本番デプロイメントのスナップショット（オブジェクトストレージ）から分析対象のIndexデータを新しいデプロイメントにリストアする
～ テスト実績～
1 index: 2,708,746 docs (182 MB) ... 数秒でリストア完了
1 index: 81,463,624 docs (24.4 GB) ... 9分でリストア完了
3. 分析が終わったらデプロイメントを削除

ご清聴ありがとうございました