

es应用笔记2-sql查询

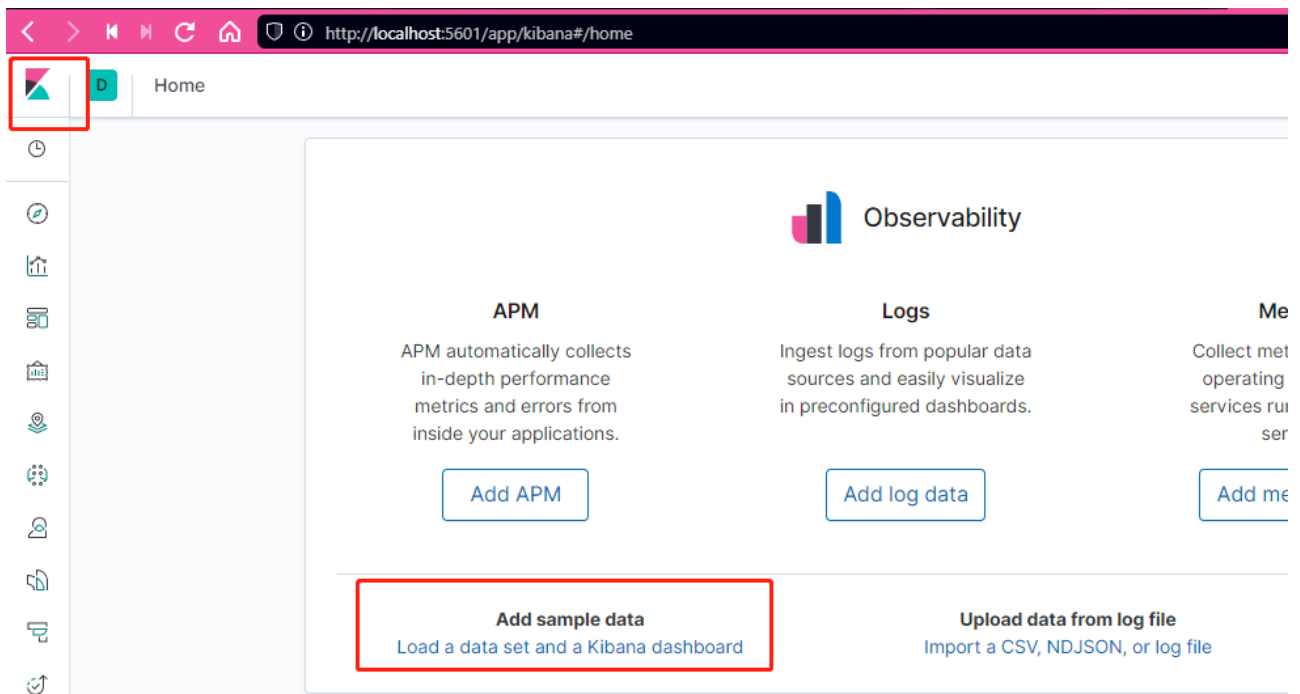
es作为一个搜索索引，在分析场景中，作为明细查询的场景会比kylin、impala、hive等更加合适。

es在6.3版本开始支持sql查询，且其sql基础语法与大数据端的语法较兼容，函数库略有不同。

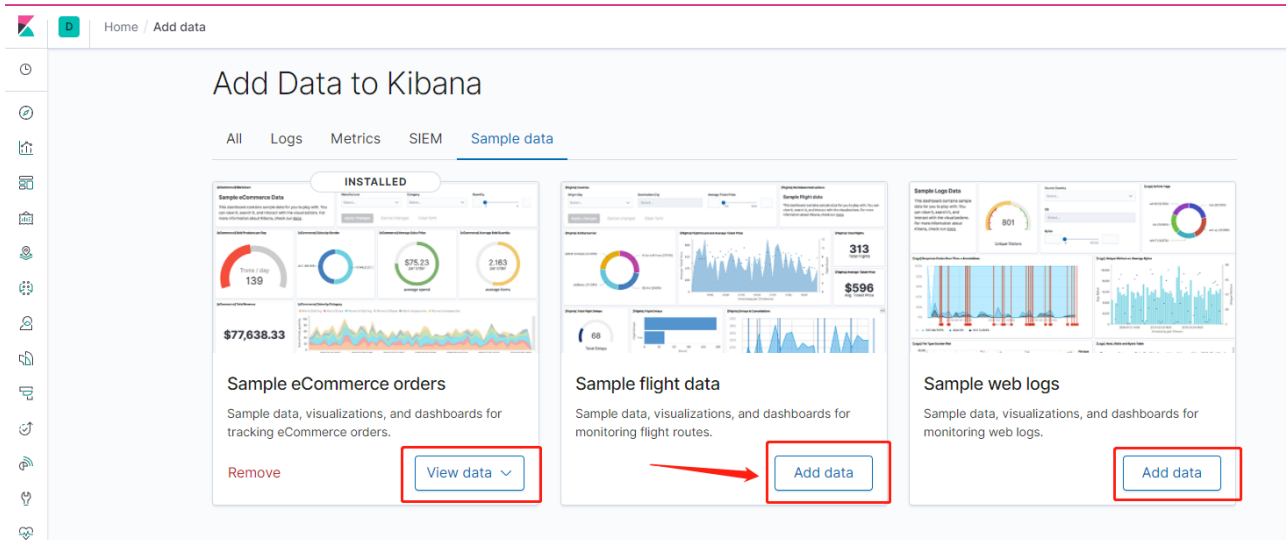
对于多数据源的接入，通过jdbc接入es改造成本较低，但是xpack-sql-jdbc这个客户端的包是收费的，但是其服务端仍提供了rest api 供查询。

界面查询

kibana中添加简单数据

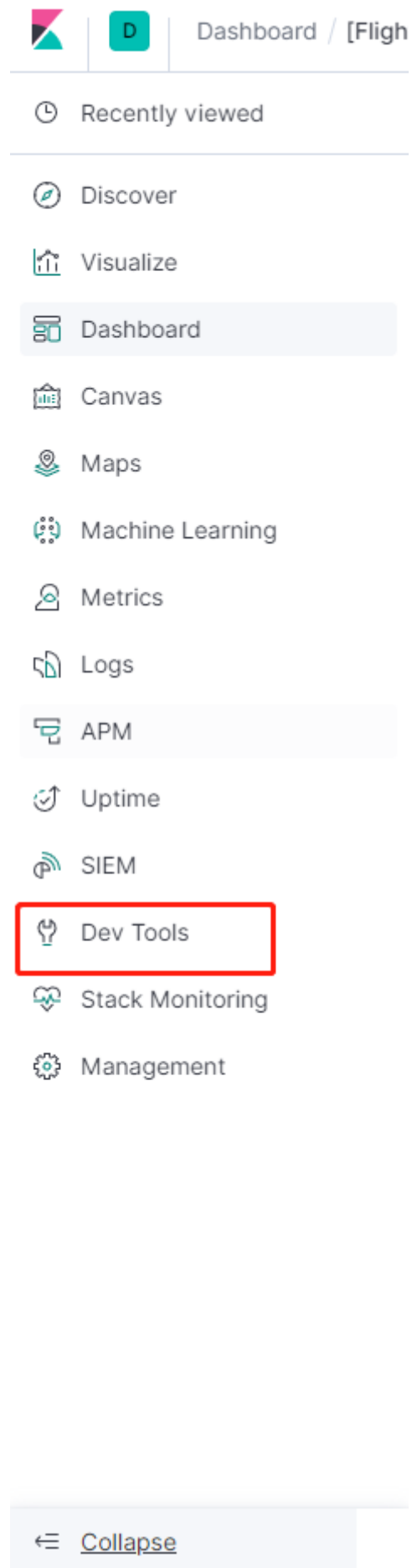


选择想要的一个栗子



开发者工具查询

- 进入开发者工具界面



- 查看有什么表

使用 SHOW TABLES 查询

Console Search Profiler Grok Debugger		
History Settings Help		
<pre>1 POST /_sql?format=txt 2 { 3 "query": "SHOW TABLES" 4 }</pre>		
1	name	type
2		kind
3	.apm-agent-configuration	BASE TABLE INDEX
4	.kibana	VIEW ALIAS
5	.kibana_1	BASE TABLE INDEX
6	.kibana_2	BASE TABLE INDEX
7	.kibana_task_manager	VIEW ALIAS
8	.kibana_task_manager_1	BASE TABLE INDEX
9	.kibana_task_manager_2	BASE TABLE INDEX
10	tasks	BASE TABLE INDEX
11	kibana_sample_data_ecommerce	BASE TABLE INDEX
12	kibana_sample_data_flights	BASE TABLE INDEX
13		

- 查看表有什么列

使用 DESCRIBE [TABLENAME]

Console Search Profiler Grok Debugger		
History Settings Help		
<pre>1 POST /_sql?format=txt 2 { 3 "query": "DESCRIBE kibana_sample_data_flights" 4 }</pre>		
1	column	type
2		mapping
3	AvgTicketPrice	REAL float
4	Cancelled	BOOLEAN boolean
5	Carrier	VARCHAR keyword
6	Dest	VARCHAR keyword
7	DestAirportID	VARCHAR keyword
8	DestCityName	VARCHAR keyword
9	DestCountry	VARCHAR keyword
10	DestLocation	GEOMETRY geo_point
11	DestRegion	VARCHAR keyword
12	DestWeather	VARCHAR keyword
13	DistanceKilometers	REAL float
14	DistanceMiles	REAL float
15	FlightDelay	BOOLEAN boolean
16	FlightDelayMin	INTEGER integer
17	FlightDelayType	VARCHAR keyword
18	FlightNum	VARCHAR keyword
19	FlightTimeHour	VARCHAR keyword
20	FlightTimeMin	REAL float
21	Origin	VARCHAR keyword
22	OriginAirportID	VARCHAR keyword
23	OriginCityName	VARCHAR keyword
24	OriginCountry	VARCHAR keyword
25	OriginLocation	GEOMETRY geo_point
26	OriginRegion	VARCHAR keyword
27	OriginWeather	VARCHAR keyword
28	dayOfWeek	INTEGER integer
29	timestamp	INTEGER datetime
30		

- SQL查询记录

查询一下延误的航班

Console Search Profiler Grok Debugger		
History Settings Help		
<pre>1 POST /_sql?format=txt 2 { 3 "query": "select t.Dest, t.DestCityName, t.FlightDelay from kibana_sample_data_flights t 4 where t.FlightDelay=true order by timestamp desc limit 20"</pre>		
1	Dest	DestCityName
2		FlightDelay
3	Xi'an Xiangyang International Airport	Xi'an true
4	Leonardo da Vinci_Fiumicino Airport	Rome true
5	Vienna International Airport	Vienna true
6	Cologne Bonn Airport	Cologne true
7	Jorge Chavez International Airport	Lima true
8	Ukrainka Air Base	Belogorsk true
9	Munich Airport	Munich true
10	Treviso-Sant'Angelo Airport	Treviso true
11	Zurich Airport	Zurich true
12	Portland International Airport	Portland true
13	Melbourne International Airport	Melbourne true
14	Verona Villafranca Airport	Verona true
15	Ukrainka Air Base	Belogorsk true
16	Winnipeg / James Armstrong Richardson International Airport	Winnipeg true
17	Sydney Kingsford Smith International Airport	Sydney true
18	Xi'an Xiangyang International Airport	Xi'an true
19	Montreal / Pierre Elliott Trudeau International Airport	Montreal true
20	Winnipeg / James Armstrong Richardson International Airport	Winnipeg true
21	Winnipeg / James Armstrong Richardson International Airport	Winnipeg true
22	Venice Marco Polo Airport	Venice true
23		

REST API

REST API 才是其他程序可以通过SQL查询ES的关键。

kibana rest api

通过浏览器F12可以获取到查询kibana的api接口，不过我们并不关心它的API:

```
curl 'http://localhost:5601/api/console/proxy?
path=%2F_sql%3Fformat%3Dtxt&method=POST' \
-H 'Connection: keep-alive' \
-H 'sec-ch-ua: "Chromium";v="98", " Not A;Brand";v="99"' \
-H 'Accept: text/plain, */*; q=0.01' \
-H 'Content-Type: application/json' \
-H 'sec-ch-ua-mobile: ?0' \
-H 'User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.136
Safari/537.36' \
-H 'kbn-version: 7.6.2' \
-H 'sec-ch-ua-platform: "Windows"' \
-H 'Origin: http://localhost:5601' \
-H 'Sec-Fetch-Site: same-origin' \
-H 'Sec-Fetch-Mode: cors' \
-H 'Sec-Fetch-Dest: empty' \
-H 'Referer: http://localhost:5601/app/kibana' \
-H 'Accept-Language: zh-CN,zh;q=0.9,zh-Hans;q=0.8,en;q=0.7' \
--data-raw '${\r\n  "query": "select t.Dest from
kibana_sample_data_flights t limit 20"\r\n}\n' \
--compressed
```

es rest api

其实kibana的开发者工具以及告诉我们ES的查询API为 `POST /_sql?format=txt`，那么稍作改造直接发给ES:

```
curl 'http://localhost:9200/_sql?format=txt' \
-H 'Connection: keep-alive' \
-H 'Accept: text/plain, */*; q=0.01' \
-H 'Content-Type: application/json' \
-d '${\r\n  "query": "select t.Dest from
kibana_sample_data_flights t limit 20"\r\n}\n' \
--compressed
```

其结果如下:

```
sh-4.2# curl 'http://localhost:9200/_sql?format=txt' \
> -H 'Connection: keep-alive' \
> -H 'Accept: text/plain, */*; q=0.01' \
> -H 'Content-Type: application/json' \
> -d '${\r\n  "query": "select t.Dest from
kibana_sample_data_flights t limit 1"\r\n}\n' \
> --compressed

          Dest
-----
Sydney Kingsford Smith International Airport
```

对于应用程序，我们选择接收JSON，那么 `format=json` 即可，结果如下：

```
sh-4.2# curl 'http://localhost:9200/_sql?format=json' \
> -H 'Connection: keep-alive' \
> -H 'Accept: text/plain, */*; q=0.01' \
> -H 'Content-Type: application/json' \
> -d '${\r\n  "query": "select t.Dest from
kibana_sample_data_flights t limit 1"\r\n}\n' \
> --compressed
{"columns":[{"name":"Dest","type":"keyword"}],"rows":[["Sydney
Kingsford Smith International Airport"]]}sh-4.2#
```

主要参数介绍

format

格式化返回结果，摘抄自官网：

FORMAT	ACCEPT HTTP HEADER	DESCRIPTION
Human Readable		
csv	text/csv	Comma-separated values
json	application/json	JSON (JavaScript Object Notation) human-readable format
tsv	text/tab-separated-values	Tab-separated values
txt	text/plain	CLI-like representation

FORMAT	ACCEPT HTTP HEADER	DESCRIPTION
yaml	application/yaml	YAML (YAML Ain't Markup Language) human-readable format
Binary Formats		
cbor	application/cbor	Concise Binary Object Representation
smile	application/smile	Smile binary data format similar to CBOR

分页

如果在查询时，使用了DSL的`fetch_size`如：

```
POST /_sql?format=json
{
  "query": "SELECT * FROM library ORDER BY page_count DESC",
  "fetch_size": 5
}
```

其返回中就会有游标：

```
{
  "columns": [

  ],
  "rows": [

  ],
  "cursor":
  "SDXF1ZXJ5QW5kRmV0Y2gBAAAAAAAAAAEWWdrR1VfSS1TbDYtcw9lc1FjNm1Ydw==:
  BAFmBmF1dGhvcgFmBG5hbWUBZgpwYwd1X2NvdW50AWYMcmVsZWZzV9kYXR1+v//w8
  ="
}
```

可以通过发送游标进行下一页查询，同时，游标还必须手动进行关闭。

```
POST /_sql/close
{
  "cursor":
  "sDXF1ZXJ5QW5kRmV0Y2gBAAAAAAAAAAEWYUpOYk1QMhRUet1d3RsNnFtYU1hQQ==:
  BAFmBGRhdGUBZgVsawt1cwFzB211c3NhZ2UBZgR1c2Vy9f///w8="
}
```

columnar

是否返回列信息

默认为true，查询返回列信息。

```
POST /_sql?format=json
{
  "query": "SELECT * FROM library ORDER BY page_count DESC",
  "fetch_size": 5,
  "columnar": true
}
```

结果:

```
{
  "columns": [
    {"name": "author", "type": "text"},
    {"name": "name", "type": "text"},
    {"name": "page_count", "type": "short"},
    {"name": "release_date", "type": "datetime"}
  ],
  "values": [
  ],
  "cursor":
  "sDXF1ZXJ5QW5kRmV0Y2gBAAAAAAAAAAEWwdrR1VfSS1TbDYtcw91c1FJNm1Ydw==:
  BAFmBmF1dGhvY2gFmBG5hbWUBZgVpY2V1X2NvdW50AWYMcmVsZWZzZV9kYXR1+v///w8
  ="
}
```

官方推荐在分页查询第一次查询时返回列信息，后续查询不再返回列信息的方式。

其他rest参数

官网链接: <https://www.elastic.co/guide/en/elasticsearch/reference/7.6/sql-rest-fields.html>

fetch_size、filter、request_timeout、page_timeout也是会用到的参数。

SQL转DSL

可以通过/_sql/translate进行转换

```
POST /_sql/translate
{
  "query": "SELECT * FROM library ORDER BY page_count DESC",
  "fetch_size": 10
}
```

SQL语法、命令

<https://www.elastic.co/guide/en/elasticsearch/reference/7.6/sql-spec.html>

函数

<https://www.elastic.co/guide/en/elasticsearch/reference/7.6/sql-functions.html>

限制

<https://www.elastic.co/guide/en/elasticsearch/reference/7.6/sql-limitations.html>

SQL查询并非ES查询主流，有许多限制需要注意，这里仅将常见的列出来。

1. 查询返回结果不能过大，会抛出异常 `ParsingException`
2. where和 order by时， scalar函数不能在嵌套字段上使用
3. 两个不同的结构的嵌套字段不能同时使用
4. 嵌套字段不能分页
5. keyword 属性需要常态化
6. array类型不能搜索，可以配置 `field.multi.value.leniency` 争取宽大处理

7. 聚合的排序不支持，将其放在客户端实现，且不允许超过512行
8. 聚合函数中必须是直接属性，而不能是scalar函数加工的属性
9. 嵌套子查询的实力只有小学生级别，超出这个范围就不支持了：
`SELECT X FROM
(SELECT ...) WHERE [simple_condition]`
10. 不能再having 中使用FIRST/LAST
11. TIME类型的属性不可以在GROUP BY / HISTOGRAM中使用
12. PIVOT中只能接收一个聚合函数