**从Hive导入数据到ES**

大数据方兴未艾，Hive在业界，是大数据的标配了。因此hive数据添加到ES的应用场景还是比较常见的。

学习ES官方的es-hadoop, 有从hive导数据到ES. 实验可行。

hive的版本： hive-1.1.0-cdh5.9.0

具体的步骤如下：

step1 将elasticsearch-hadoop-hive-version.jar添加到hive

wget https://artifacts.elastic.co/downloads/elasticsearch-hadoop/elasticsearch-hadoop-6.3.0.zip

unzip elasticsearch-hadoop-6.3.0.zip

hdfs dfs -mkdir /user/test/es\_hadoop/

hdfs dfs -put elasticsearch-hadoop-hive-6.3.0.jar /user/test/es\_hadoop/

ADD JAR hdfs://test/user/test/es\_hadoop/elasticsearch-hadoop-hive-6.3.0.jar;

step2 创建Hive表：

CREATE EXTERNAL TABLE elastic\_table(

uuid string,

key1 int,

key2 int,

day string

)

STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'

TBLPROPERTIES('es.resource'='index/type',

'es.nodes'='serverIP:port',

'es.index.auto.create'='TRUE',

'es.mapping.id' = 'uuid'

);

step3 添加数据

INSERT OVERWRITE TABLE elastc\_table

SELECT uuid, key1,key2, day FROM source s;

1.

2.

为了避免客户端版本的问题，es-hadoop使用es的restfull接口导入数据,该接口使用的是Http协议。

通常使用ES, 首当其冲的问题就是： 如何快速将海量数据导入ES? 由于ES的数据需要建立倒排索引，所以导入数据到ES的瓶颈往往在ES这里。

本文记录了将Hive表的数据导入ES的方法。这里背后隐藏了mapreduce，即集群的威力。 这里有个系列博客，讲述如何最大限度的挖掘ES索引数据的性能，立足点是ES。

https://qbox.io/blog/series/how-to-maximize-elasticsearch-indexing-performance

1.

作者总结有3点：

根据应用场景创建mapping, 去除不必要的字段，如\_all, \_source;

这里是从应用场景下手，以避免存储不必要的信息来提升索引数据的性能。

修改es/lucene默认的设置，比如

refresh\_interval,

index.number\_of\_replicas,

index.merge.scheduler.max\_thread\_count,

index.translog.interval,

indices.memory.index\_buffer\_size

index.index\_concurrency

等参数。 这里是从集群的角度进行调优， 通常用于大批量导入数据到ES。

如果前面两种还是没能解决问题，那就需要对集群进行横向扩展了，比如增加集群的分片数量。

集群大了后，各个结点的功能就需要单一化，专注化了。

比如节点只承担数据相关的任务。

node.master: false

node.data: true

node.ingest: false

bulk api的批量值需要实验，找到最佳参数。建议bulk的大小在5M～10M.

使用SSD硬盘。索引数据时，副本数设置为0。

参考:

 http://note4code.com/2016/06/17/hive-向-elasticsearch-导出数据/

**Hive 向 ElasticSearch 导出数据**

一般来说，Hive 里的表经过初步的处理之后就可以向 ElasticSearch 中导入数据了。

ElasticSearch 提供了和 Hive 的整合方案 elasticsearch-hadoop.jar，简单来说我们可以按照如下步骤进行操作。

1. *如果已存在同名 Index 则先删除旧的 Index。*
2. *创建 Index，设置好 mappings。*
3. *创建导入数据用的映射表。*
4. *导入数据*
5. *删除临时表。*

**1. 删除已存在的 Index**

因为我在使用的时候将每天的分区数据映射到一个 Index 中，所以为了回溯数据，每次导数开始之前先删除旧有的 Index。这个操作使用 ElasticSearch 提供的 RESTful API 就行：

curl -XDELETE "http://${IP\_ADDRESS}:9200/${PATITION\_NAME}"

**2. 创建 Index，设置 mappings**

其实这一步可以省略，在导入数据的时候 ElasticSearch 会自动创建对应的 mappings，但是提前创建 mappings 的好处是可以做自定义的设置。

比如下面的 "index": "not\_analyzed" 属性禁止 ElasticSearch 对字符串的分析。如果不加上这个设置那么在匹配字符串的时候只能用 matchPhrase 而不能用 term，而且在 group by 某个字段的时候这个字段可能只返回字符串值的一部分，比如“北京”只返回一个“北”等等。

总之，如果不需要对文本做检索，所有字符串类型的属性最好都加上 "index": "not\_analyzed"。

curl -XPUT "http://${IP\_ADDRESS}:9200/${PATITION\_NAME}" -d '

{

"mappings": {

"data":{

"properties": {

"id": {

"type": "long"

},

"stat\_date": {

"format": "strict\_date\_optional\_time||epoch\_millis",

"type": "date"

},

"some\_str\_field": {

"type": "string",

"index": "not\_analyzed"

},

"some\_num\_field": {

"type": "long"

}

},

"\_all": {

"enabled": false

}

}

}

}'

**3. 创建导入数据用的映射表**

创建一个外部表，并用 es.nodes 指定几个连接用的节点，es.resource指定映射的 Index 和 Type。给个例子：

add jar elasticsearch-hadoop-hive-2.2.0.jar;

drop table if exists ${TEMP\_TABLE};

CREATE EXTERNAL TABLE ${TEMP\_TABLE}(

id bigint comment 'id',

stat\_date string comment '日期',

some\_str\_field string comment '字符串类型',

some\_num\_field bigint comment '数值类型'

)

STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'

TBLPROPERTIES(

'es.nodes' = '${NODE\_IP\_ADDR\_1}:9200,${NODE\_IP\_ADDR\_2}:9200,${NODE\_IP\_ADDR\_3}:9200',

'es.resource' = '${PATITION\_NAME}/${TYPE}'

);

**4. 导入数据**

SET hive.mapred.reduce.tasks.speculative.execution = false;

SET mapreduce.map.speculative = false;

SET mapreduce.reduce.speculative = false;

insert overwrite table ${TEMP\_TABLE}

select ${COLUMNS}

from ${DBNAME}.${FILENAME}

where concat(year,month,day)='$V\_TS';

**5. 删除临时表**

导数用的临时表在导入数据结束后就可以删除了。

drop table ${TEMP\_TABLE};

**6. 此处有坑**

在使用 Hive 向 ElasticSearch 中导出数据的时候，有一个大坑非常关键：并发！

**这种导入数据的方法会启动 mapper 来执行，但是如果 mapper 数过高，ElasticSearch 过载（Overload）或者多出冗余的数据。据同事分析，原因是 mapper 会批量地将数据插入 ElasticSearch 中，而当 mapper 失败的时候会被重启，已经导入的数据又会重新插入 ElasticSearch 导致数据重复。**

其中的一个手段就是适当减少并发的 mapper 的数量。通过合并文件的手段可以降低 mapper 的数量。

SET mapred.max.split.size=1000000000;

SET mapred.min.split.size=1000000000;

set mapred.min.split.size.per.node=1000000000;

set mapred.min.split.size.per.rack=1000000000;

set hive.input.format=org.apache.hadoop.hive.ql.io.CombineHiveInputFormat;

另外，还可以通过设置 ElasticSearch 的 \_id 字段避免重复。比如为每一行生成一个 uuid，把 uuid 作为 mappings 中的 \_id。ElasticSearch 对于 uuid 相同的记录会覆盖旧的记录，这样相同 uuid 的记录就不会重复了。

add jar elasticsearch-hadoop-hive-2.2.0.jar;

drop table if exists ${TEMP\_TABLE};

CREATE EXTERNAL TABLE ${TEMP\_TABLE}(

id bigint comment 'id',

stat\_date string comment '日期',

some\_str\_field string comment '字符串类型',

some\_num\_field bigint comment '数值类型',

uuid string COMMENT '为了导入ES而准备的id'

)

STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'

TBLPROPERTIES(

'es.nodes' = '${NODE\_IP\_ADDR\_1}:9200,${NODE\_IP\_ADDR\_2}:9200,${NODE\_IP\_ADDR\_3}:9200',

'es.resource' = '${PATITION\_NAME}/${TYPE}',

'es.mapping.id' = 'uuid'

);

**最近更新：最近公司架构部门搞了一套通过 Kafka 导入数据的方案，先把数据写入到 Kafka，然后 Kafka 写入到 ElasticSearch。但是 Kafka 保证不会丢数，还是可能有重复数据项，所以也加了 uuid 作为 \_id 来去重。这个套方案运行的很好，可惜同时导入的时候并发导入任务数也并不能太高，有些回溯数据的任务也只能一个个跑。**