

如何将Hive中的数据导入到Druid中

笔记本: Apache Druid

创建时间: 2020/5/27 15:19

更新时间: 2020/5/27 15:22

将Hive中的数据导入到Druid中

Apache Druid是一个实时OLAP型数据库，现在有一个需求就是将Hive中的一张大表，导入到Druid中，然后应用层直接查询Druid。

接下来就是将Hive中的数据导入到Druid的详细步骤：

我的方案是直接从HDFS上将源文件直接load过去

0. 系统基本信息

Druid版本0.18

hadoop版本cdh集成的2.6.0

使用的是Druid的single-server中的micro-quickstart模式

1. Druid准备hadoop依赖

1.1 配置文件准备

Druid从hdfs上读数据，还要将hadoop的配置文件拷贝到Druid的配置目录下。
在\$DRUID_HOME/conf/druid/single-server/micro-quickstart/_common
下创建一个文件夹hadoop-xml

然后将以下几个文件拷贝进去

```
$ ll
total 28
-rw-r--r-- 1 bigdata bigdata 3850 Apr 28 08:34 core-site.xml
-rw-r--r-- 1 bigdata bigdata 2942 Apr 28 08:34 hdfs-site.xml
-rw-r--r-- 1 bigdata bigdata 5039 Apr 28 08:34 mapred-site.xml
-rw-r--r-- 1 bigdata bigdata 5529 Apr 28 08:34 yarn-site.xml
```

这一步参考官方文档：

<http://druid.apache.org/docs/latest/tutorials/tutorial-batch-hadoop.html#configure-druid-to-use-hadoop>

1.2 Druid深度存储位置更换

这一步就是更换Druid的segment和log的存储位置，默认是local本地存储。

打开 conf/druid/single-server/micro-quickstart/_common/common.runtime.properties文件，更改以下配置

```
#
# Deep storage
#
# For local disk (only viable in a cluster if this is a network mount):
#druid.storage.type=local
#druid.storage.storageDirectory=var/druid/segments
# For HDFS:
druid.storage.type=hdfs
druid.storage.storageDirectory=/druid/segments
#
# Indexing service logs
#
# For local disk (only viable in a cluster if this is a network mount):
#druid.indexer.logs.type=file
#druid.indexer.logs.directory=var/druid/indexing-logs
# For HDFS:
druid.indexer.logs.type=hdfs
druid.indexer.logs.directory=/druid/indexing-logs
```

这一步操作完后重启Druid节点。

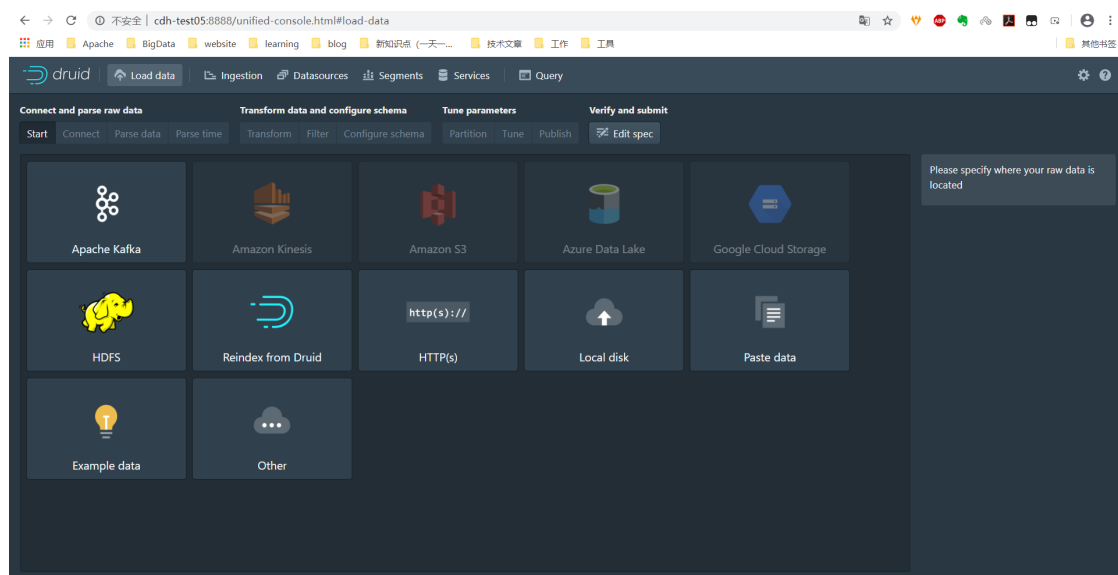
2.Hive表格要求

因为要直接从hdfs上load数据，而hdfs上的文件如果内容是json格式的还处理好一些，但是hive表在hdfs上存储的文件是没有Schema的，所以就要按照一定的格式进行字段切分。然后在Druid的配置文件进行映射，hive默认的列分割符是u0001，这个在Druid里边是不识别的，所以要导入的Hive表最好是其他分隔符，比如我的hive边列分隔符指定的是t。

3. 从页面直接添加数据源

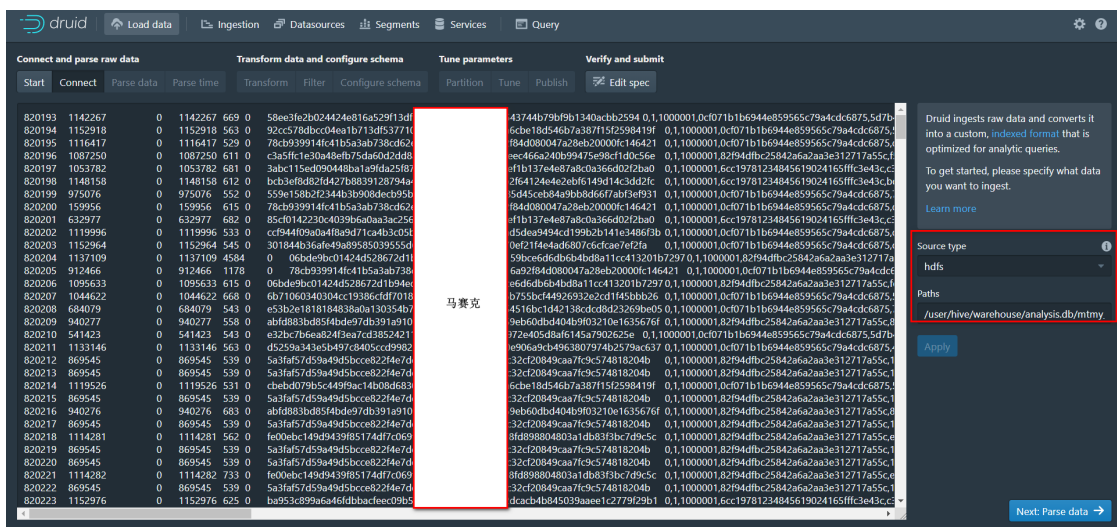
以上步骤做完了就开始load数据

3.1 打开<http://hostname:8888>选择Load data



3.2 load数据

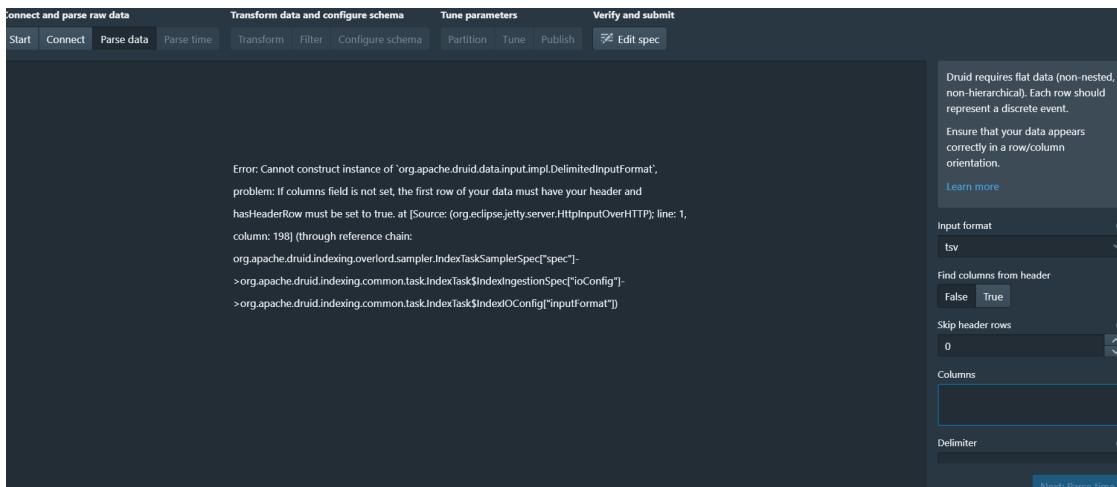
在右侧Source type以及Paths上填写相应的信息后点击Apply之后就加载进了数据



3.3 Parse data

点击Parse data后会有如下报错信息，这是因为我们的schema信息还没有配置，Druid解析失败。

```
Error: Cannot construct instance of
`org.apache.druid.data.input.impl.DelimitedInputFormat`,
problem: If columns field is not set, the first row of your data must have your
header and
hasHeaderRow must be set to true. at [Source:
(org.eclipse.jetty.server.HttpInputOverHTTP);
line: 1, column: 198] (through reference chain:
org.apache.druid.indexing.overlord.sampler.IndexTaskSamplerSpec["spec"]-
>org.apache.druid.indexing.common.task.IndexTask$IndexIngestionSpec["ioConfig"]-
>org.apache.druid.indexing.common.task.IndexTask$IndexIOConfig["inputFormat"])
```




然后在右边工具栏，input format选择tsv，Find columns from header选择False（默认），Skip header rows写0（默认），再Columns栏里填写Hive表中对应的列信息，Delimiter不填（注：不能写t，直接在框里打一个Tab空格，或者直接不写系统会自动识别并填入），List delimiter不填，接下来点击Apply就出现了以下内容，Schema信息就配置好了

3.4 Parse time

time字段是Druid中必须要指定的，因此Druid也可以作为时序数据库，这一步系统一般会默认给我们选择一个字段作为time字段，如果这个字段不是我们想要的就需要我们自己指定，拉动进度，点击我们这指定的time字段的表头，右边信息栏就会有相应的信息，格式默认是auto，如果时间格式是固定的话，我们也可以直接指定。

选择之后点击Apply，预览一下数据：

<div>  <div> <div>Load data</div> <div>Ingestion</div> <div>Datasources</div> <div>Segments</div> <div>Services</div> <div>Query</div> </div> </div>						
Connect and parse raw data			Transform data and configure schema			Tune parameters
Start	Connect	Parse data	Parse time	Transform	Filter	Configure schema
<div> <div>Search columns</div> <div> <input type="checkbox"/> Suggested columns only </div> </div>			<div> <div>Partition</div> <div>Tune</div> <div>Publish</div> </div>			
_time from: 'create_date'	id	group_id	sendtouserflag	goods_mappin	goods_id	batch_id
2017-12-10T17:27:56.000Z	820193	1142267	0	1142267	669	0
2017-12-10T17:27:56.000Z	820194	1152918	0	1152918	563	0
2017-12-10T17:28:02.000Z	820195	1116417	0	1116417	529	0
2017-12-10T17:28:04.000Z	820196	1087250	0	1087250	611	0
2017-12-10T17:28:06.000Z	820197	1053782	0	1053782	681	0
2017-12-10T17:28:13.000Z	820198	1148158	0	1148158	612	0
2017-12-10T17:28:17.000Z	820199	975076	0	975076	552	0
2017-12-10T17:28:47.000Z	820200	159956	0	159956	615	0
2017-12-10T17:28:48.000Z	820201	632977	0	632977	682	0
2017-12-10T17:29:29.000Z	820202	1119996	0	1119996	533	0
2017-12-10T17:29:30.000Z	820203	1152964	0	1152964	545	0
2017-12-10T17:29:37.000Z	820204	1137109	0	1137109	4584	0
2017-12-10T17:29:52.000Z	820205	912466	0	912466	1178	0
2017-12-10T17:29:56.000Z	820206	1095633	0	1095633	615	0

3.5 Transforms

Druid可以对每行列值执行**转换**，从而创建新的派生列或更改现有列
这一步是可选的，我们的应用场景没有相关需求，直接跳过。

3.6 Filter

制定规则过滤掉不需要的数据
这一步是可选的，我们的应用场景没有相关需求，直接跳过。

3.7 Configure schema

这一步是最最最关键的，详细可以查看官网相关资料
<https://druid.apache.org/docs/0.18.0/ingestion/schema-design.html>
界面是这样子的：

Each column in Druid must have an assigned type (string, long, float, double, complex, etc). Default primitive types have been automatically assigned to your columns. If you want to change the type, click on the column header. [Learn more](#)

☒ Explicitly specify dimension list

☒ Rollup

Query granularity: DAY

Add dimension

Add metric

Next: Partition

右侧有两个选项Explicitly specify dimension list和Rollup

Explicitly specify dimension list: 选择是否显式指定维度列表，官网是这样解释的：选择是否要设置维度和指标的明确列表。明确设置维度和指标可以带来更好的压缩和性能。如果禁用此选项，则Druid将尝试自动检测数据中的字段并将其视为单独的列。

Rollup:

如果启用Rollup，则Druid会尝试在索引之前为数据进行预聚合以节省存储空间。主时间戳将被截断为指定的查询粒度，并且包含相同字符串字段值的行将被聚合在一起。

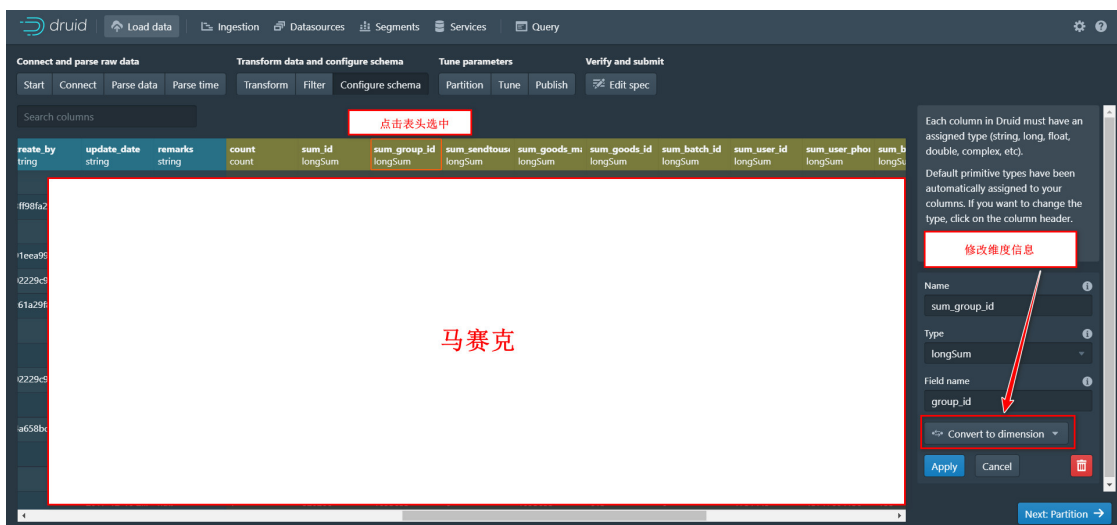
如果启用Rollup，则必须指定哪些列是维度（要对其进行分组和过滤的字段），哪些是度量（要对其进行汇总的字段）。

我们使用Druid想要查询快，这两个选项是必须开启的。

还有一个选项是Query granularity，这是用来指定查询粒度的，系统提供了5种选择，我们的应用场景粒度到天就够了。

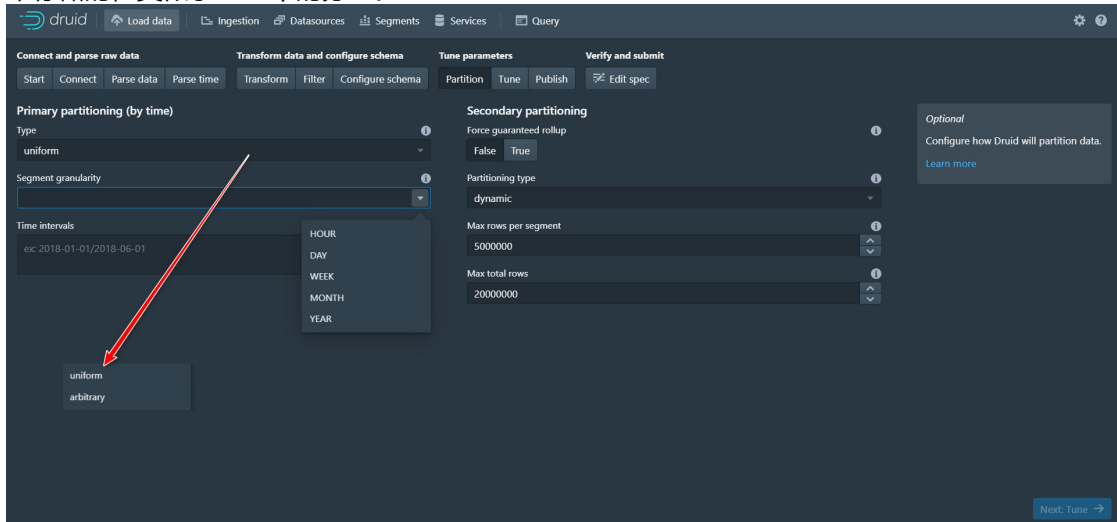
到这一步后，就该调整维度和度量信息了，系统会默认给我们指定一些维度和度量的列，往往这些都不是我们想要的，如下图，列名为count或者sum_开头的就是系统自动给指定的度量列，按照图示方法就行度量列修改或者删除。

当然要是将维度列修改为度量列这是同样的方法。

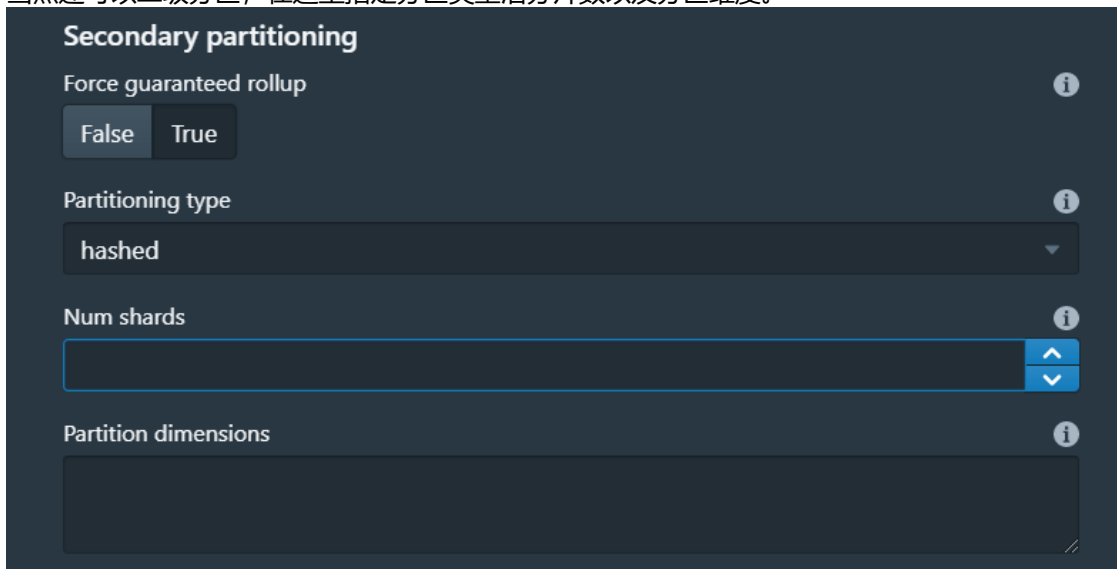


3.8 Partition

这一步是指定segment分区设置，类型可以选择统一的或者任意，一般按照时间查询的比较多，我们这里也是指定按照天来分区segment，设置好后实际上segment也是按照每个分区一个文件来存储的，类似于Hive中的分区。



然后设置一下每个segment最大的行，以及所有segment中等待推送的最大的行数，关于这两个参数，想进一步了解可以查看官网的详细解释。当然还可以二级分区，在这里指定分区类型活分片数以及分区维度。

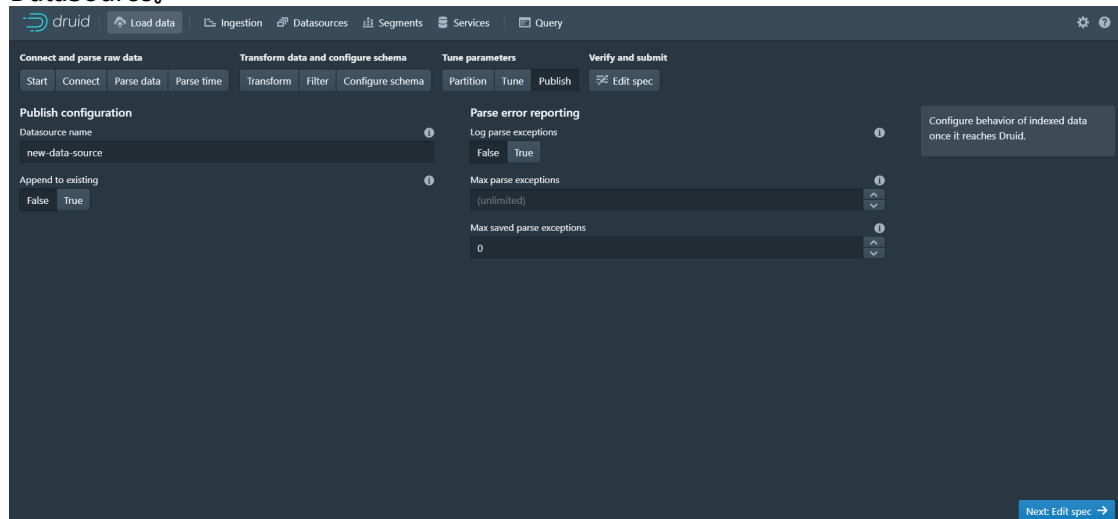


3.9 Tune

这一步是可选的，作用是微调Druid将如何提取数据

3.10 Publish

在这一步对整个DataSource进行配置，需要更改的是DataSource name以及是否追加到现有的DataSource。



3.11 Edit spec

编辑摄取规范，到这一步，Druid会根据我们之前的设置，自动生成一个json串，数据的加载将会按照json串中的配置进行，如果之前的步骤都是按照需求设置的，这一步就是预览一下不用修改，就可以直接提交了。

```
{
  "type": "index_parallel",
  "spec": {
    "ioConfig": {
      "type": "index_parallel",
      "inputSource": {
        "type": "hdfs",
        "paths": "/user/hive/warehouse/xxx.db/xxx"
      },
    },
    "inputFormat": {
      "type": "tsv",
      "findColumnsFromHeader": false,
      "columns": [
        "id",
        "group_id",
        .....
        "flag"
      ]
    },
    "appendToExisting": false
  },
  "tuningConfig": {
    "type": "index_parallel",
    "partitionsSpec": {
      "type": "dynamic"
    },
    "forceGuaranteedRollup": false
  },
  "dataSchema": {
    "dataSource": "demo_test",
```



```

"granularitySpec": {
  "type": "uniform",
  "queryGranularity": "DAY",
  "rollup": true
},
"timestampSpec": {
  "column": "create_date",
  "format": "yyyy-MM-dd HH:mm:ss"
},
"dimensionsSpec": {
  "dimensions": [
    "beautician_id",
    "beautician_name",
    .....
    "update_date",
    "remarks"
  ]
},
"metricsSpec": [
  {
    "name": "count",
    "type": "count"
  },
  .....
  {
    "name": "sum_post_fee",
    "type": "longSum",
    "fieldName": "post_fee"
  }
]
}
}
}

```

4. 选择ingestion，查看数据load进度

直到status变为SUCCESS状态后，就可以查询数据了。

The screenshot shows the Apache Druid Ingestion interface. The 'Ingestion' tab is selected in the top navigation bar. Below the 'Supervisors' section, which shows 'No supervisors', is the 'Tasks' section. The 'Tasks' section has a table with columns: Task ID, Group ID, Type, Datasource, Location, Created time, Status, and Duration. A single task is listed with the status 'SUCCESS', which is highlighted by a red box. The task details are as follows:

Task ID	Group ID	Type	Datasource	Location	Created time	Status	Duration
index_parallel-[redacted]	2020-04-29T04:24:30.166Z	index_parallel	ibjicam...	localhost:8100	2020-04-29T04:24:30.167Z	SUCCESS	6:27:38

5. 查询数据

1

```
SELECT COUNT(*) AS cnt FROM hogetbl
```

▶ Run

⋮

☒ Auto run

☒ Smart query limit

1 result in 0.07s

cnt

9953055