Урок 5. Потоковая обработка данных. Flume/Flink/SparkStreaming

Выполнил Колеганов Н.Д.

Задание

Для части по SQOOP

Провести импорт таблицы из вашего сервера БД в Hadoop с использованием SQOOP в любых двух вариантах из перечисленных ниже.

- а. в Hive-таблицу (--hive-import)
- b. в HDFS в формате avro (--as-avrodatafile)
- с. в HDFS в формате sequencefile (--as-sequencefile)

Если у вас нет своего сервера то можно использовать тот Postgres, который я показал на лекции. Пароль expoter_pass

Для части по потоковой обработке (Flume)

- 1. Создать Flume-агент с именем, соответствующим имени своего пользователя (например Flume4 20)
- 2. Создать любой Flume поток используя Flume сервис соответствующего номера.
- Тип источника источник ехес
- Тип канала memory
- Тип слива hdfs
- 3. Убедиться что данные поступают в слив.
- 4. Создать поверх данных в hdfs таблицу через которую можно просмотреть полученные данные.
- 5. [Продвинутый вариант] Сделать то-же самое используя несколько сливов в разные места, например в HDFS и в HIve одновременно
- 6. [Продвинутый вариант] Повторить стандартный пример с выборкой сообщений из Twitter.

Создал БД pg_db_test1

Создал таблицу character в формате parquet без компрессии

```
1.37s pg_db_test1 * text * ?

1 CREATE DATABASE pg_db_test1;
2
3 SET parquet.compession=none;
4
5 CREATE TABLE pg_db_test1.character (
6 charid string,
7 charname string,
8 abbrev string,
9 description string,
10 speechcount int)
11 STORED AS PARQUET;
```

sqoop import --connect jdbc:postgresql://node3.novalocal/pg_db --username exporter -P --table character --hive-import --hive-database pg_db_test1 —as-parquetfile

```
[student4_2@manager ~]$ hdfs dfs -du -h -s /user/hive/warehouse/pg_db_test1.db/character

69.6 K 208.8 K /user/hive/warehouse/pg db test1.db/character
```

Итоговый размер около 69.6 кб

Попробовал различные запросы на таблице character:

SELECT SUM(speechcount) FROM pg_db_test1.character WHERE abbrev='First Musician'

SELECT count(*) FROM pg_db_test1.character WHERE description IS NOT NULL

SELECT charname, SUM(speechcount) FROM pg_db_test1.character WHERE description IS NOT



NULL GROUP BY charname

Импорт таблицы sales_large в формате parquet

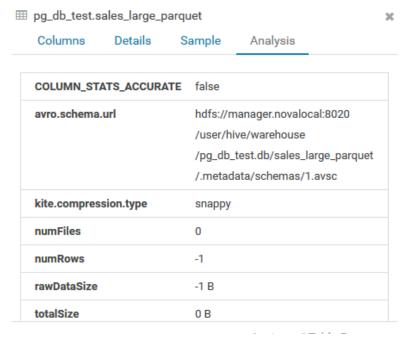
sqoop import --connect jdbc:postgresql://node3.novalocal/pg_db --username exporter -P --table sales_large --hive-import --hive-database pg_db_test1 --hive-table sales_large_parquet -as-parquetfile - m 1

[student4_2@manager ~]\$ sqoop import --connect jdbc:postgresql://node3.novalocal/pg_db --username exporter -P --table sales_la rge --hive-import --hive-database pg_db_test1 --hive-table sales_large_parquet —as-parquetfile -m 1

[student4 2@manager ~]\$ hdfs dfs -du -h -s /user/hive/warehouse/pg db test1.db/sales large parquet

458.7 M 1.3 G /user/hive/warehouse/pg db test1.db/sales large parquet

Размер таблицы 458.7 Мб.



Как видно на скриншоте по умолчанию для parquet установлена компрессия snappy. Если пробовать импорт с ключом -z компрессия все равно остается snappy.

Проверим читабельность импортируемых данных:

■ pg_db_test.sales_large_parquet				
Columns Details Sampl		Sample	Analysis	
	sales_large_parquet.region		sales_large_parquet.coun	
1	Middle East and North Africa		Bahrain	
2	Middle East and North Africa		Pakistan	
3	Asia		Kazakhstan	
4	Central America and the Caribbean		Guatemala	
5	Sub-Saharan Africa		Kenya	
6	Sub-Saharan Africa		Nigeria	
7	Australia and Oceania		East Timor	
R	Central America and the Caribbean		Fl Salvador	

Импорт таблицы sales_large в формате avro

sqoop import --connect jdbc:postgresql://node3.novalocal/pg_db --username exporter -P --table sales_large --as-avrodatafile --target-dir /user/hive/warehouse/pg_db_test.db/sales_large_avro -m 1

[student4 2@manager ~]\$ hdfs dfs -du -h -s /user/hive/warehouse/pg db test.db/sales large avro

1.5 G 4.5 G /user/hive/warehouse/pg_db_test.db/sales_large_avro

Размер таблицы 1.5 Gb.

В Ние таблицу не видно, поэтому создаем там EXTERNAL TABLE:

```
1 CREATE EXTERNAL TABLE pg_db_test.sales_large_avro (
   region string,
   country string,
 4 itemtype string,
   saleschannel string,
  orderpriority string,
 7
   orderdate string,
 8 orderid int,
 9 shipdate string,
10 unitssold decimal(10,0),
11 unitprice decimal(10,0),
12 unitcost decimal(10,0),
13 totalrevenue decimal(10,0),
14 totalcost decimal(10,0),
15 totalprofit decimal(10,0))
16 STORED AS AVRO;
```

■ pg_db_test.sales_large_avro

Details

Columns

	sales_large_avro.region	sales_large_avro.country
1	Middle East and North Africa	Bahrain
2	Middle East and North Africa	Pakistan
3	Asia	Kazakhstan
4	Central America and the Caribbean	Guatemala
5	Sub-Saharan Africa	Kenya
6	Sub-Saharan Africa	Nigeria
7	Australia and Oceania	East Timor
8	Central America and the Caribbean	El Salvador

Sample

Analysis

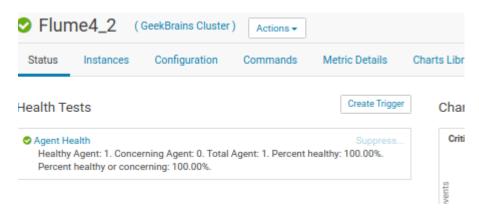
Импорт таблицы sales_large в формате avro с компрессией gzip

sqoop import --connect jdbc:postgresql://node3.novalocal/pg_db --username exporter -P --table sales_large --as-avrodatafile --target-dir /user/hive/warehouse/pg_db_test.db/sales_large_avro -z -m 1

Размер таблицы получается около 450 Мб.

Для части по потоковой обработке (Flume)

Создал Flum



Flume конфиг:

Agent Name Agent Default Group Flume4_2 Configuration File Agent Default Group 5 # Naming the components on the current agent Flume4_2.sources = ExecSource Flume4_2.channels = MemChannel Flume4_2.sinks = HdfsSink # Describing/Configuring the source Flume4_2.sources.ExecSource.type = exec Flume4_2.sources.ExecSource.command = tailf /var/log/messages Flume4_2.sources.ExecSource.interceptors = TimestampInterceptor Flume4_2.sources.ExecSource.interceptors.TimestampInterceptor.type = timestamp # Describing/Configuring the HDFS sink Flume4_2.sinks.HdfsSink.type = hdfs Flume4 2.sinks.HdfsSink.hdfs.path=/flume/flume4 2/log/%v-%m-%d/ Flume4_2.sinks.HdfsSink.hdfs.filePrefix = events # Describing/Configuring the channel Flume4_2.channels.MemChannel.type = memory Flume4_2.channels.MemChannel.capacity = 10000 Flume4_2.channels.MemChannel.transactionCapacity = 10 # Bind the source and sink to the channel Flume4_2.sources.ExecSource.channels = MemChannel Flume4_2.sinks.HdfsSink.channel = MemChannel

Naming the components on the current agent

Flume4 2.sources = ExecSource

Flume4_2.channels = MemChannel

 $Flume4_2.sinks = HdfsSink$

Describing/Configuring the source

Flume4 2.sources.ExecSource.type = exec

Flume4_2.sources.ExecSource.command = tailf /var/log/messages

Flume4_2.sources.ExecSource.interceptors = TimestampInterceptor

Flume4_2.sources.ExecSource.interceptors.TimestampInterceptor.type = timestamp

Describing/Configuring the HDFS sink

Flume4_2.sinks.HdfsSink.type = hdfs

Flume4_2.sinks.HdfsSink.hdfs.path= /flume/flume4_2/log/%y-%m-%d/

Flume4 2.sinks.HdfsSink.hdfs.filePrefix = events

Describing/Configuring the channel

Flume4 2.channels.MemChannel.type = memory

Flume4_2.channels.MemChannel.capacity = 10000

Flume4_2.channels.MemChannel.transactionCapacity = 10

Bind the source and sink to the channel

Flume4 2.sources.ExecSource.channels = MemChannel

Flume4_2.sinks.HdfsSink.channel = MemChannel

Проверил статус:



Проверил ведутся ли записи