

BigData. Введение в экосистему Hadoop.

Урок 6. ETL

На уроке разбираем основные типы задач по загрузке данными и учимся использовать различные инструменты.

Оглавление

[Оглавление](#)

[Теоретическая часть](#)

[Строчные форматы](#)

[Колоночные форматы](#)

[Бинарные форматы](#)

[Практическая часть](#)

[Домашнее задание](#)

[Используемая литература](#)

Теоретическая часть

Виды получения данных

ETL -- (от англ. Extract, Transform, Load — дословно «извлечение, преобразование, загрузка») — один из основных процессов в управлении хранилищами данных, который включает в себя:

- извлечение данных из внешних источников;
- их трансформация и очистка, чтобы они соответствовали потребностям бизнес-модели;
- и загрузка их в хранилище данных.

ELT — Извлечение, Загрузка и преобразование — это процесс, с помощью которого данные извлекаются из исходной системы, загружаются в выделенное хранилище, а затем преобразуются.

Batch - Типичный сценарий работы с большими данными — это пакетная обработка неактивных данных. В этом случае исходные данные загружаются в хранилище данных либо самим исходным приложением, либо рабочим процессом оркестрации. Затем данные параллельно обрабатываются на месте с помощью задания, которое также может быть инициировано рабочим процессом оркестрации. В рамках обработки может выполняться несколько итерационных шагов до того, как преобразованные результаты будут загружены в хранилище аналитических данных для последующего запроса компонентами аналитики и отчетов.

Streaming - Обработка в режиме реального времени выполняется для потоков данных, получаемых в реальном времени и обрабатываемых с минимальной задержкой для создания отчетов или автоматизированного реагирования в режиме реального времени (или приближенном к реальному времени).

Популярные инструменты

Apache Sqoop - это инструмент, предназначенный для передачи данных между Hadoop и реляционными базами данных или мейнфреймами. Вы можете использовать Sqoop для импорта данных из реляционных систем управления базами данных (СУБД), таких как MySQL или Oracle или мейнфреймов в Hadoop, преобразовывать данные в Hadoop MapReduce, а затем экспортировать данные обратно в СУБД.

Apache Flume — распределенная и высоконадежная система для эффективного сбора, агрегации и сохранения больших объемов логов из множества различных источников в централизованное хранилище данных. Изначально созданный для потоковой обработки логов в конвейерах, Flume масштабируется горизонтально и управляется событиями.

Apache NiFi — это простая платформа обработки событий (сообщений), предоставляющая возможности управления потоками данных из разнообразных источников в режиме реального времени с использованием графического интерфейса.

Apache Kafka — распределенный программный брокер сообщений, проект с открытым исходным кодом, разрабатываемый в рамках фонда Apache. Написан на языках программирования Java и Scala. Спроектирован как распределённая, горизонтально масштабируемая система, обеспечивающая наращивание пропускной способности как при росте числа и нагрузки со стороны источников, так и количества систем-подписчиков.

Практическая часть

SQOOP

1. Получаем список таблиц, которые созданы в postgres (нужно правильно указать ip, сейчас указан localhost)

```
sqoop list-tables --connect jdbc:postgresql://localhost/pg_db --username exporter --password exporter_pass
```

2. Создаем таблицу в hive, куда собираемся выгружать данные

```
CREATE TABLE `lesson6_9.character` (
```

```
    `charid` string,
```

```
    `charname` string,
```

```
    `abbrev` string,
```

```
    `description` string,
```

```
    `speechcount` int)
```

```
CREATE TABLE sell_prices
```

```
(store_id varchar, item_id varchar, wm_yr_wk varchar, sell_price varchar);
```

```
CREATE TABLE `sell_prices` ( `store_id` string, `item_id` string, `wm_yr_wk` string, `sell_price` string)
```

3. Запускаем импорт:

```
sqoop import --connect jdbc:postgresql://localhost/pg_db --username exporter --password exporter_pass  
--table sell_prices --hive-import --hive-database student10_40 --hive-table sell_prices
```

```
CREATE TABLE `lesson6.character_p` (
```

```

`charid` string,

`charname` string,

`abbrev` string,

`description` string,

`speechcount` int)

PARTITIONed BY (p_date string)

ROW FORMAT SERDE

'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'

WITH SERDEPROPERTIES (

'field.delim'='\u0001',

'line.delim'='\n',

'serialization.format'='\u0001')

STORED AS INPUTFORMAT

'org.apache.hadoop.mapred.TextInputFormat'

OUTPUTFORMAT

'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'

LOCATION

'hdfs://10.0.0.8:8020/user/hive/warehouse/character_p'

```

DATE=`date '+%Y%m%d'`

```

sqoop import --connect jdbc:postgresql://10.0.0.8/pg_db --username exporter --password exporter_pass
--table character --target-dir /user/hive/warehouse/character_p/p_date=$DATE --delete-target-dir

```

```

hive -e 'msck REPAIR TABLE lesson6.character_p'

```

```

msck REPAIR TABLE lesson6.character_p

```

FLUME

1. Создаем shell-скрипт и сохраняем в файл heartbeat.sh

```
START_DATE=`date`

COUNT=0

while [ true ]

do

    NOW_DATE=`date`

    echo I live for $(( `date -d "$NOW_DATE" +%s` - `date -d "$START_DATE" +%s` )) seconds\;`(date -d "$START_DATE" +%Y-%m-%d:%H.%M.%S)\;`(date -d "$START_DATE" +%Y-%m-%d:%H.%M.%S)\;I did it $(( $COUNT + 1 )) times

    COUNT=$(( $COUNT + 1 ))

    sleep 10

Done
```

2. Создаем конфигурационный файл, правильно указываем пути до shell-скрипта и hdfs-папки

```
# Naming the components on the current agent
```

```
StudentFlume.sources = Exec
```

```
StudentFlume.channels = MemChannel
```

```
StudentFlume.sinks = MySink
```

```
# Describing/Configuring the source
```

```
StudentFlume.sources.Exec.type = exec
```

```
StudentFlume.sources.Exec.command = /home/centos/flm/heartbeat.sh
```

```
# Describing/Configuring the hdfs sink
```

```
StudentFlume.sinks.MySink.type=hdfs
```

```
StudentFlume.sinks.MySink.hdfs.path=/user/centos/flm
```

```
StudentFlume.sinks.MySink.hdfs.fileSuffix=.log
```

```
# Describing/Configuring the channel
```

```
StudentFlume.channels.MemChannel.type = memory
```

```
StudentFlume.channels.MemChannel.capacity=10000000
```

```
StudentFlume.channels.MemChannel.transactionCapacity = 100
```

```
# Bind the source and sink to the channel
```

```
StudentFlume.sources.Exec.channels = MemChannel
```

```
StudentFlume.sinks.MySink.channel = MemChannel
```

3. Запускаем flume (нужно правильно указать пути до конфигурационной папки и конфигурационного файла)

```
/opt/apache-flume/bin/flume-ng agent --conf /home/user/flm/ --conf-file /home/user/flm/flm.conf --name StudentFlume -Dflume.root.logger=INFO,console
```

4. Создаем таблицы в hive (найдите ошибку в скрипте, чтобы данные сразу раскладывались по колонкам, укажите правильно location)

```
create external table lesson10_6.flm_logs (first string, second string, third string, fourth string, fifth string)
```

```
ROW FORMAT DELIMITED
```

```
FIELDS TERMINATED BY ','
```



```
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.SequenceFileInputFormat'
```

```
OUTPUTFORMAT 'org.apache.hadoop.hive ql.io.HiveSequenceFileOutputFormat'
```

```
location '/user/centos/flm';
```

5. Проверяем результат

```
select * from flm_logs
```

Домашнее задание

Sqoop:

1. Получаем список таблиц, которые созданы в postgres (нужно правильно указать ip, сейчас указан localhost)
`sqoop list-tables --connect jdbc:postgresql://localhost/pg_db --username exporter --password exporter_pass`
2. Создаем таблицу в hive, куда собираемся выгружать данные
`CREATE TABLE lesson6_9.character(
charid string,
charname string,
abbrev string,
description string,
speechcount int)`
3. Запускаем импорт:
`sqoop import --connect jdbc:postgresql://node3.novalocal/pg_db --username exporter --password exporter_pass --table character --hive-import --hive-database lesson6_9 --hive-table character`

Flume:

1. Запускаем flume (нужно правильно указать пути до папки и файла)
`/opt/apache-flume/bin/flume-ng agent --conf /home/admin/flm/ --conf-file /home/admin/flm/flm.conf --name StudentFlume -Dflume.root.logger=INFO,console`
2. Создаем таблицы в hive
`create external table lesson10_6.flm_logs (first string, second string, third string, fourth string, fifth string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.SequenceFileInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveSequenceFileOutputFormat'
location '/user/centos/flm'`

Задачи со * предназначены для продвинутых учеников, которым мало сделать обычное ДЗ.

Используемая литература

Для подготовки данного методического пособия были использованы следующие ресурсы:

1. <https://ru.wikipedia.org/wiki/ETL>
2. https://ru.bmstu.wiki/Apache_Sqoop
3. https://ru.wikipedia.org/wiki/Apache_Kafka