Практическое задание к лекции по HDFS (Лекция №4)

---Первая часть задания -- форматы данных ---

Есть большая таблица по имени

create external table hive\_db.citation\_data

(

oci string,

citing string,

cited string,

creation string,

timespan string,

journal\_sc string,

author\_sc string

)

ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

location '/test\_datasets/citation'

Её размер вот такой:

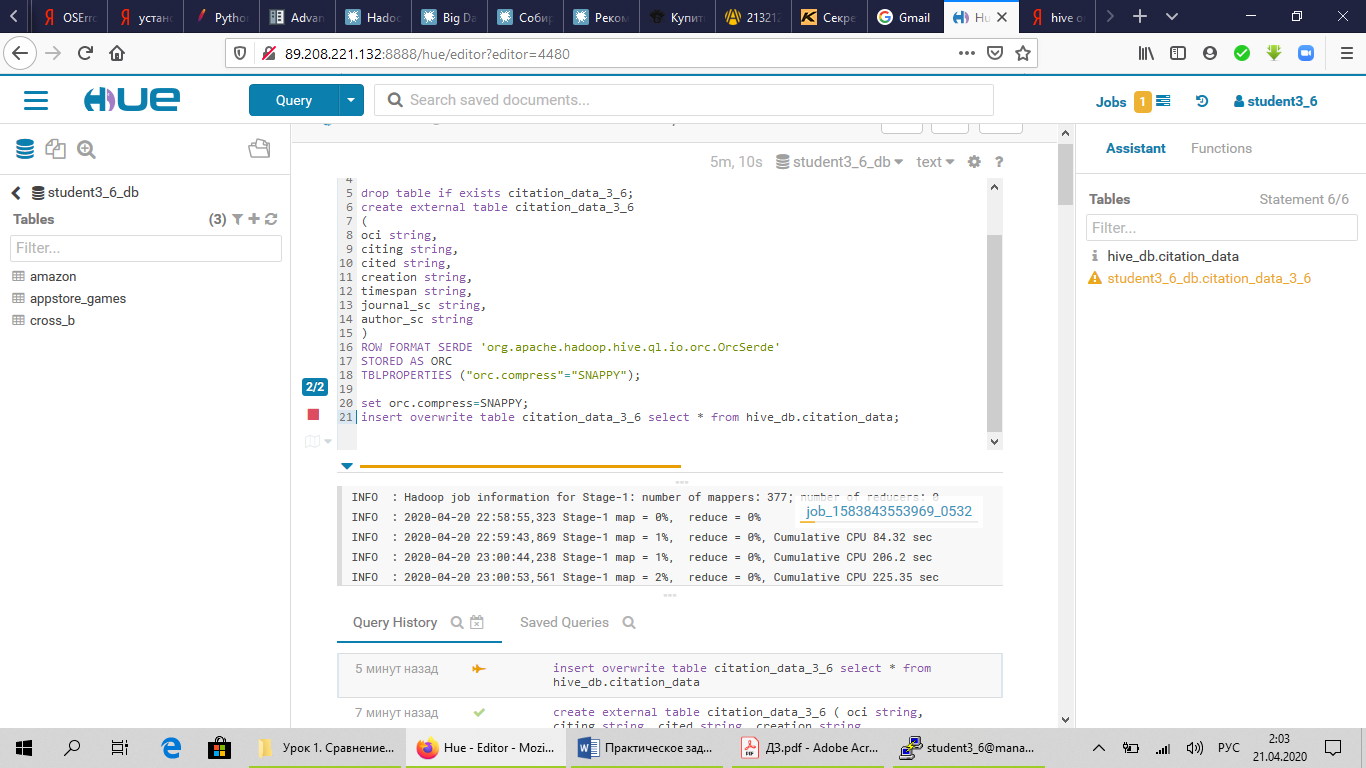
hdfs dfs -du -h -s /test\_datasets/citation

97.2 G 291.5 G /test\_datasets/citation

Что вам нужно сделать

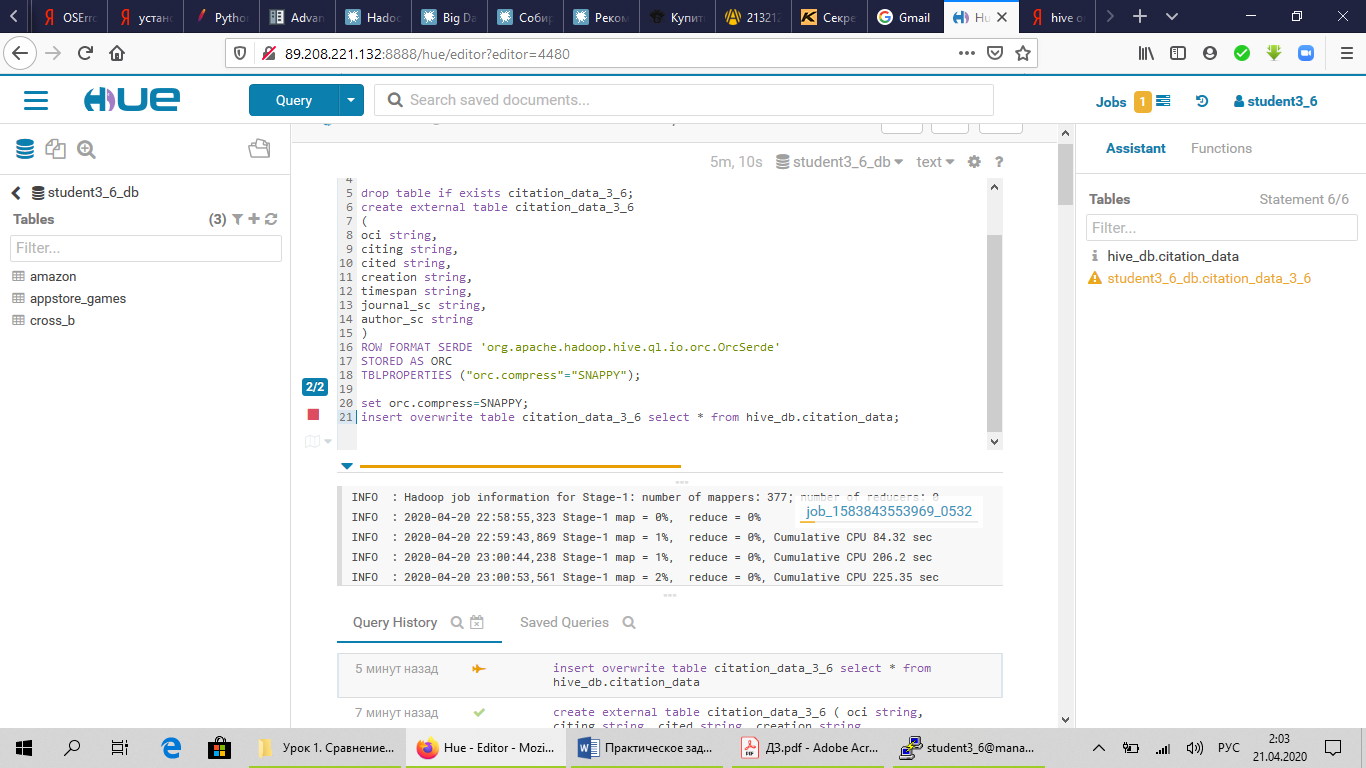
1. Создать таблицы в форматах PARQUET/ORC/AVRO c компрессией и без оной. (Выберите один вариант, например ORC с компрессией)

Создал таблицу citation\_data\_3\_6:

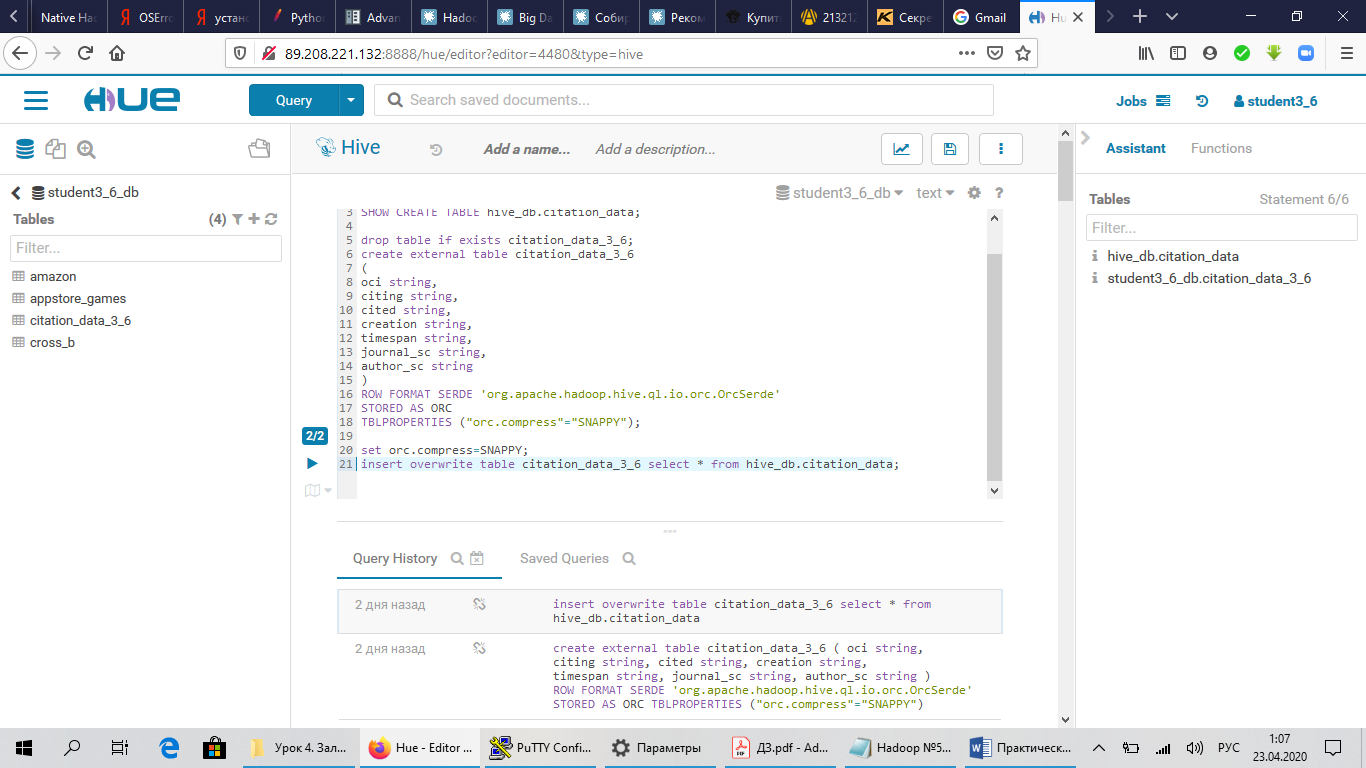


2. Заполнить данными из большой таблицы hive\_db.citation\_data

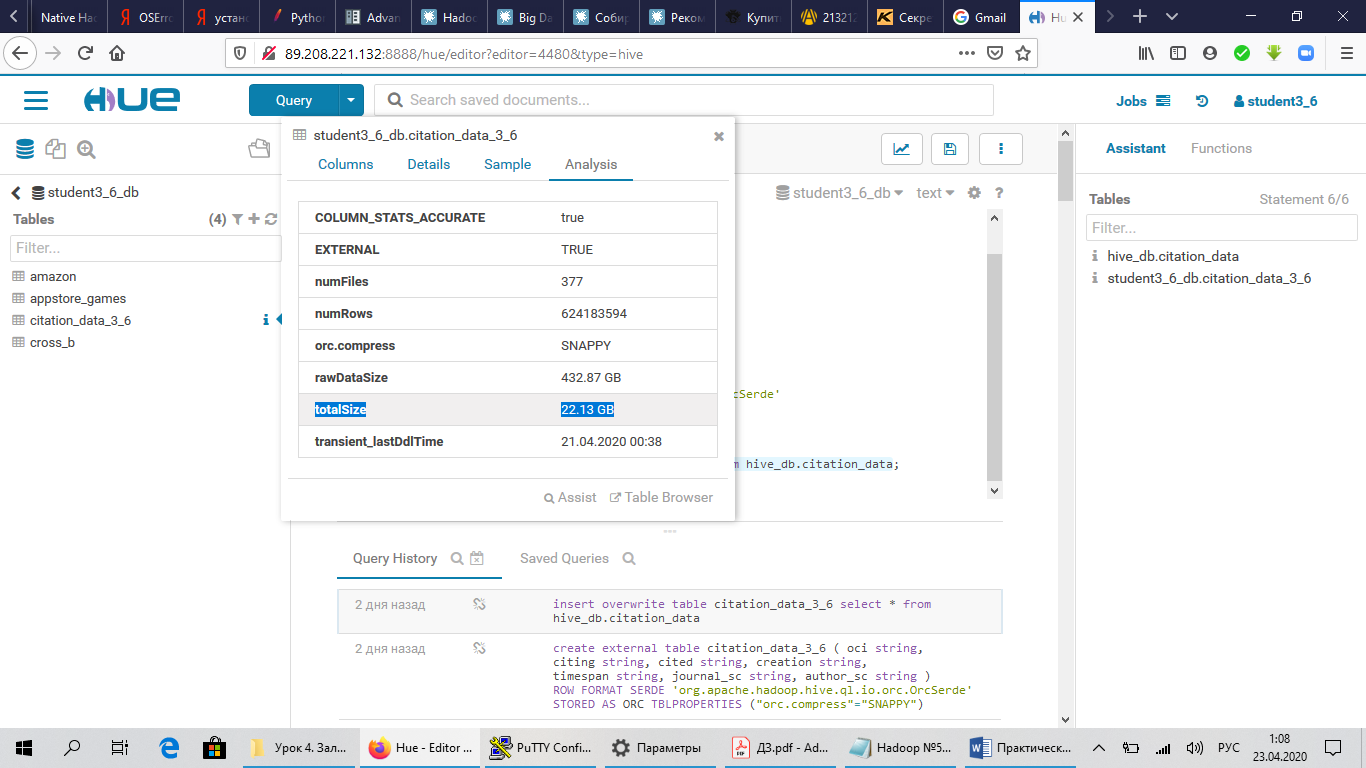
Создал job на заполнение данными из материнской таблицы в свою в формате ORC с компрессией SNAPPY:



Job выполнен и база создана:



3. Посмотреть на получившийся размер данных



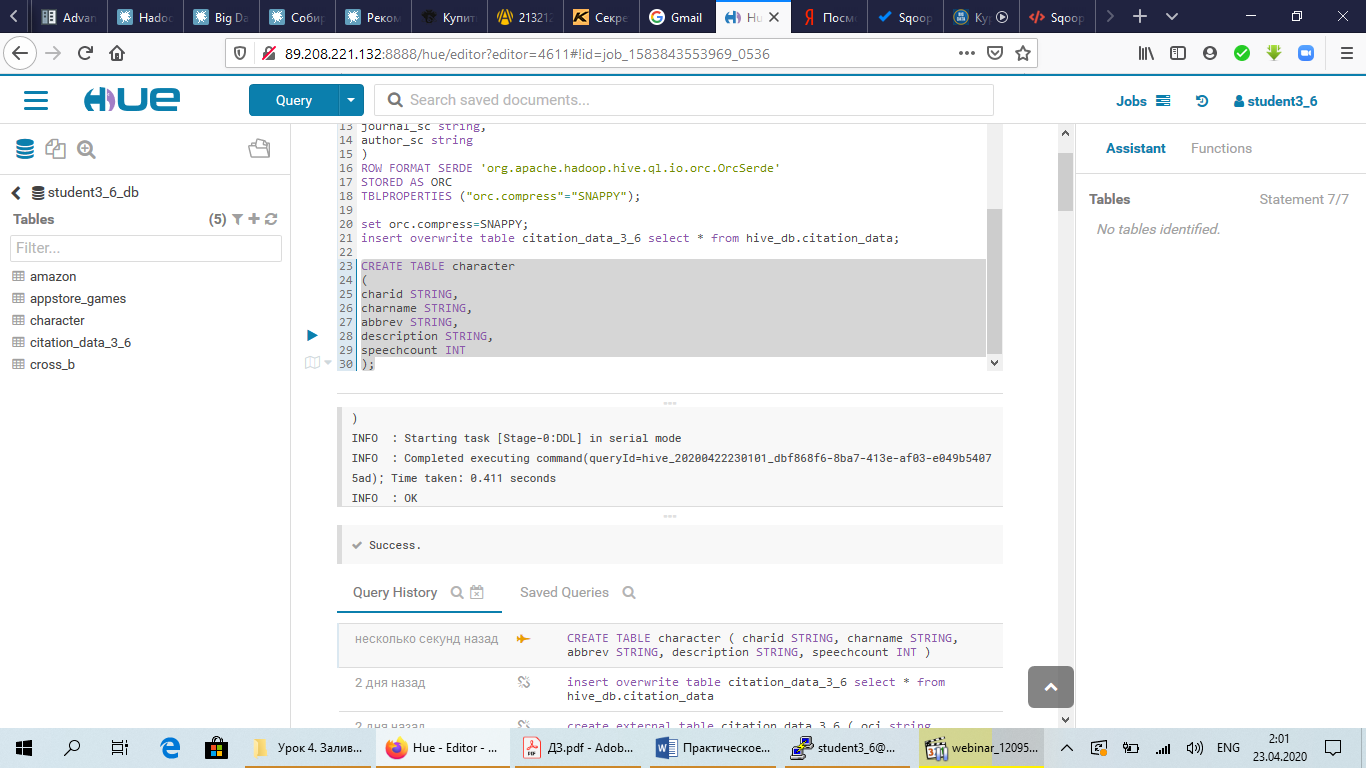
4. Сделать выводы о эффективности хранения и компрессии

В итоге размер базы данных стал 22,13 GB вместо 97,2 GB. Сжатие примерно в 4,4 раза.

---Вторая часть задания -- SQOOP ---

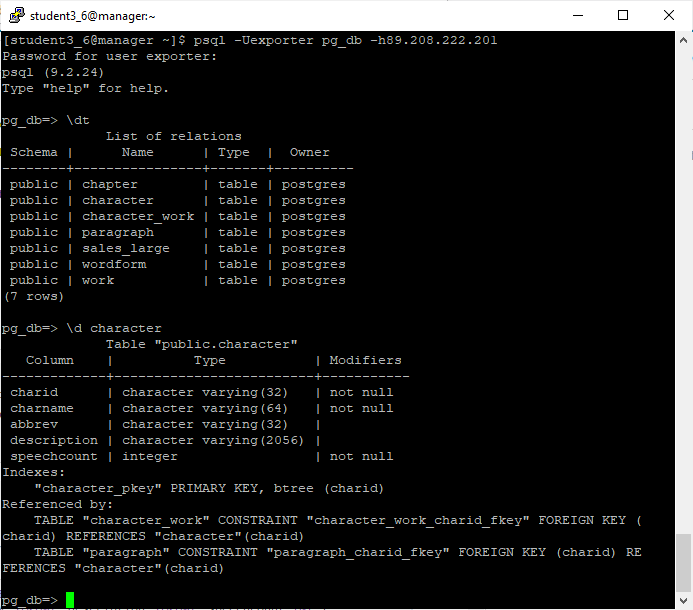
Простое задание довольно :)

1. Создать отдельную БД в HIve

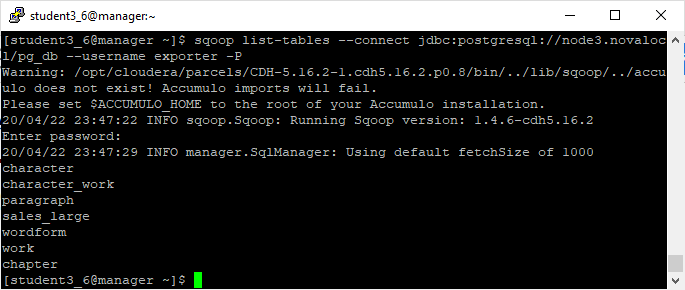


2. Посмотреть при помощи SQOOP содержимое БД в POsgresql

Подсоединился к базе pg\_db через терминал и посмотрел наличие таблицы character



Посмотрел наличие таблиц в базе данных pg\_db через sqoop:



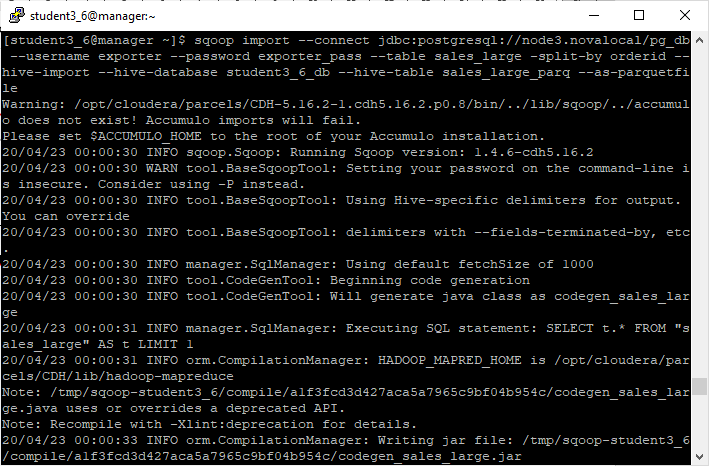
3. Импортировать в нее три любые таблицы из базы pg\_db в Postgresql используя SQOOP. Для каждой таблицы используйте отдельный формат хранения -- ORC/Parquet/AVRO

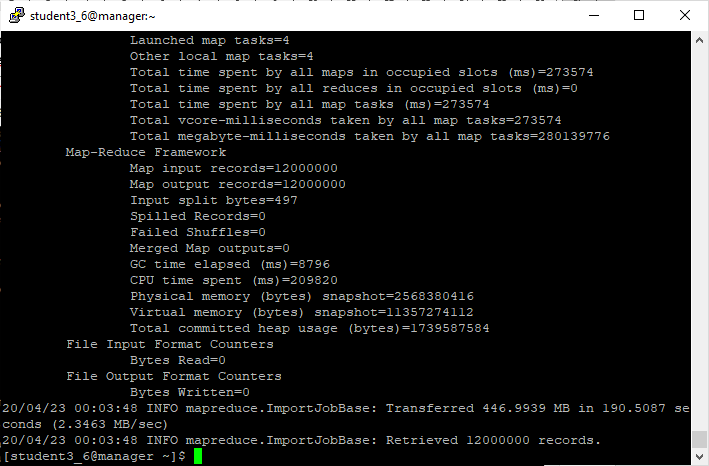
Рекомендую захватить таблицу sales\_large -- там порядка 10 миллионов записей, она будет достаточно репрезентативна для проверки компрессии.

Parquet:

sqoop import --connect jdbc:postgresql://node3.novalocal/pg\_db --username exporter -P --table sales\_large -split-by orderid --hive-import --hive-database student3\_6\_db --hive-table sales\_large\_parq --as-parquetfile

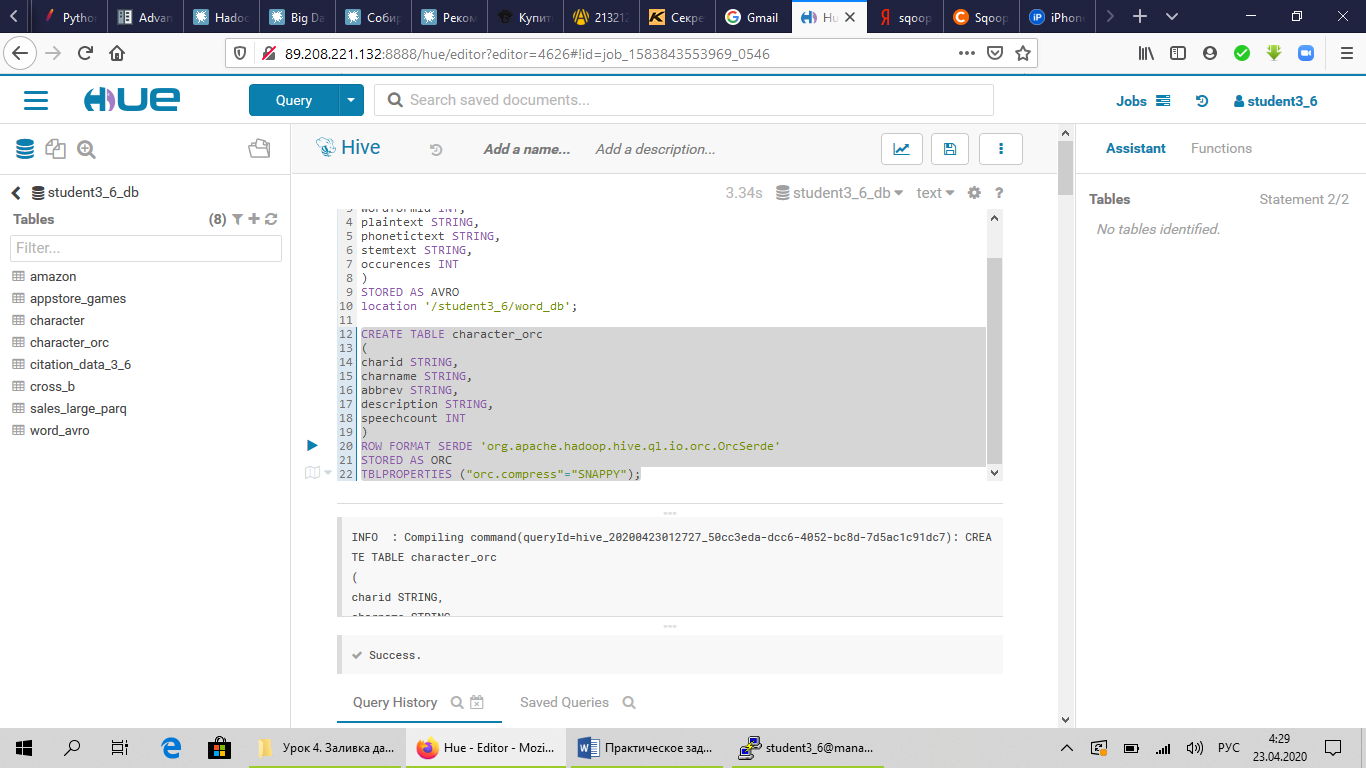
Всё отлично отработало:





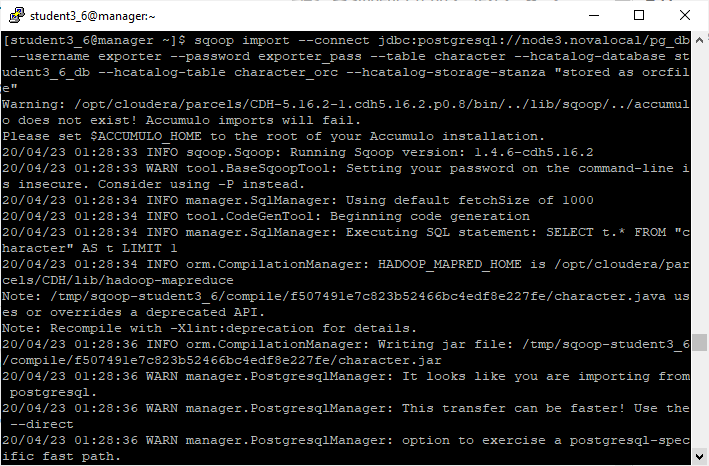
ORC:

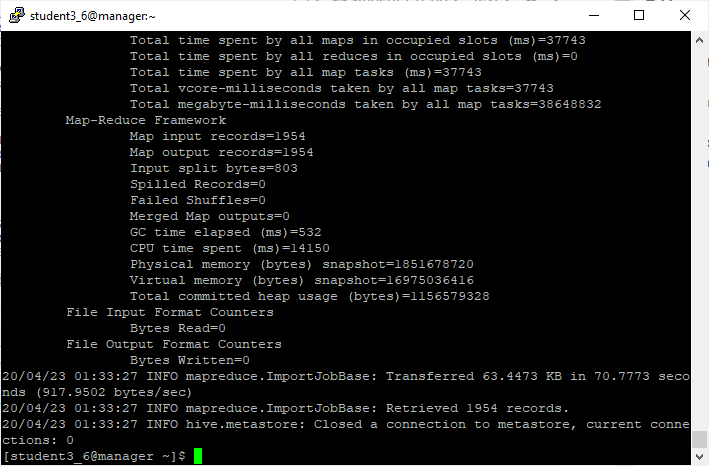
Создаём таблицу в Hive:



Закидываем данные через sqoop:

sqoop import --connect jdbc:postgresql://node3.novalocal/pg\_db --username exporter --password exporter\_pass --table character --hcatalog-database student3\_6\_db --hcatalog-table character\_orc --hcatalog-storage-stanza "stored as orcfile"

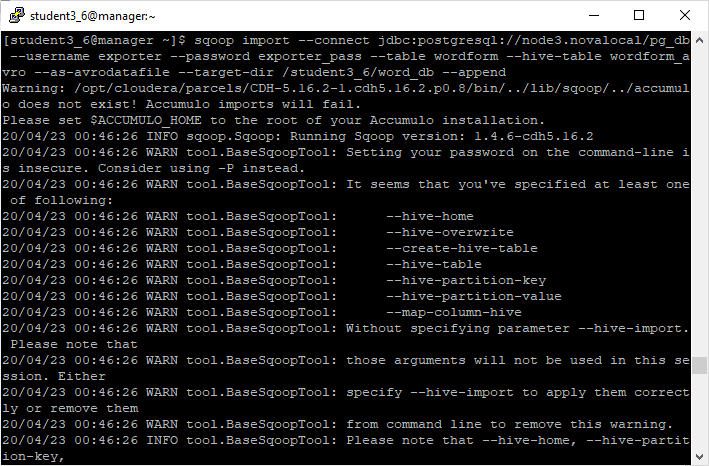


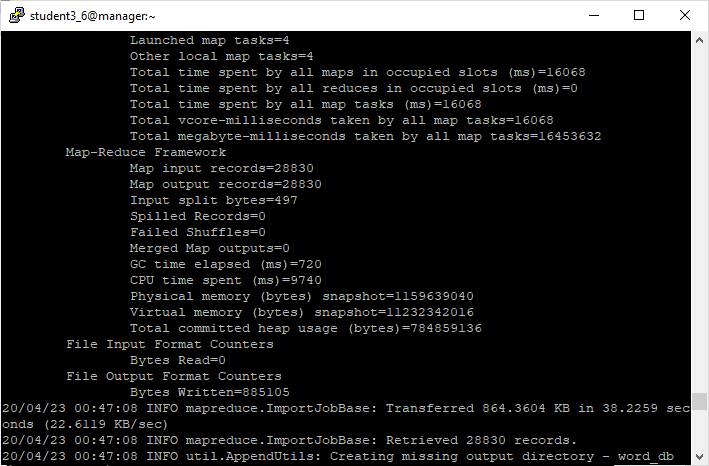


AVRO:

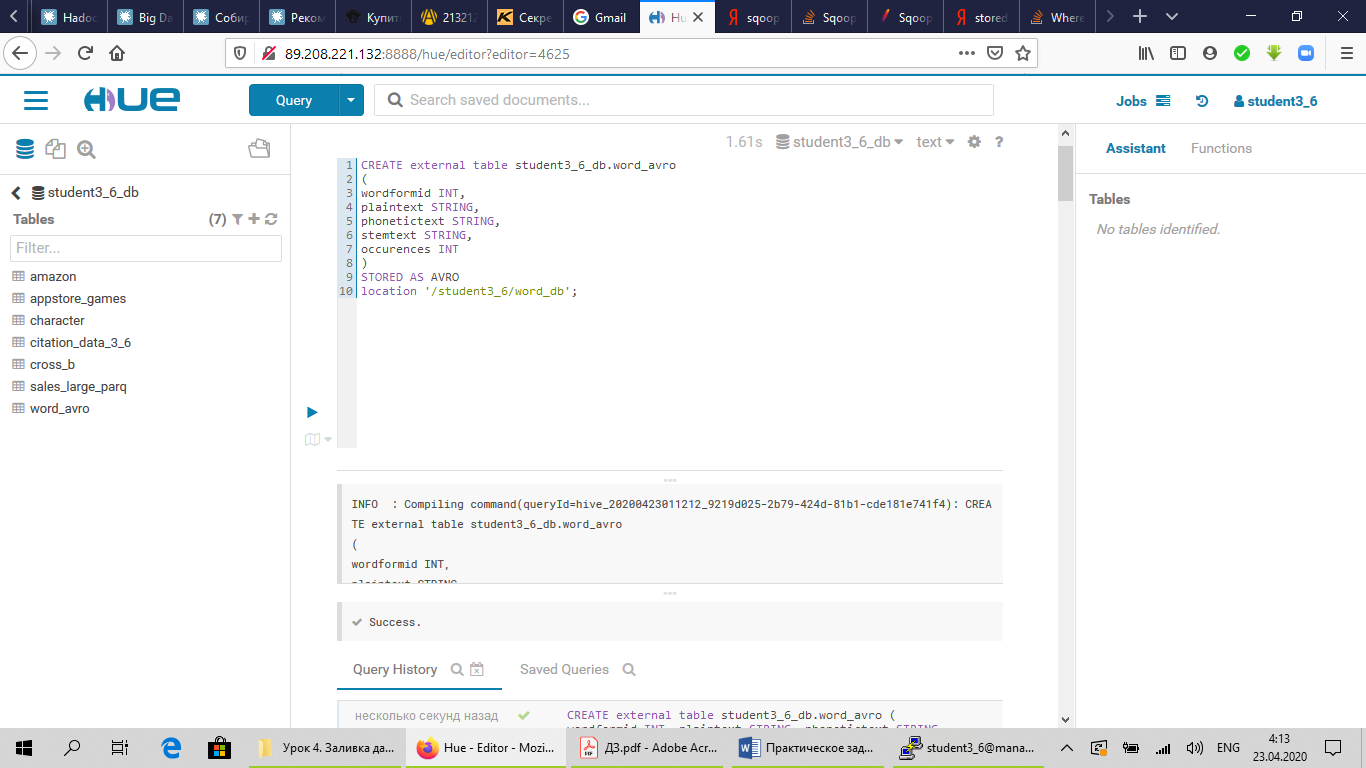
Так как напрямую загрузить данные не получилось, то сначала закинул данные в HDFS:

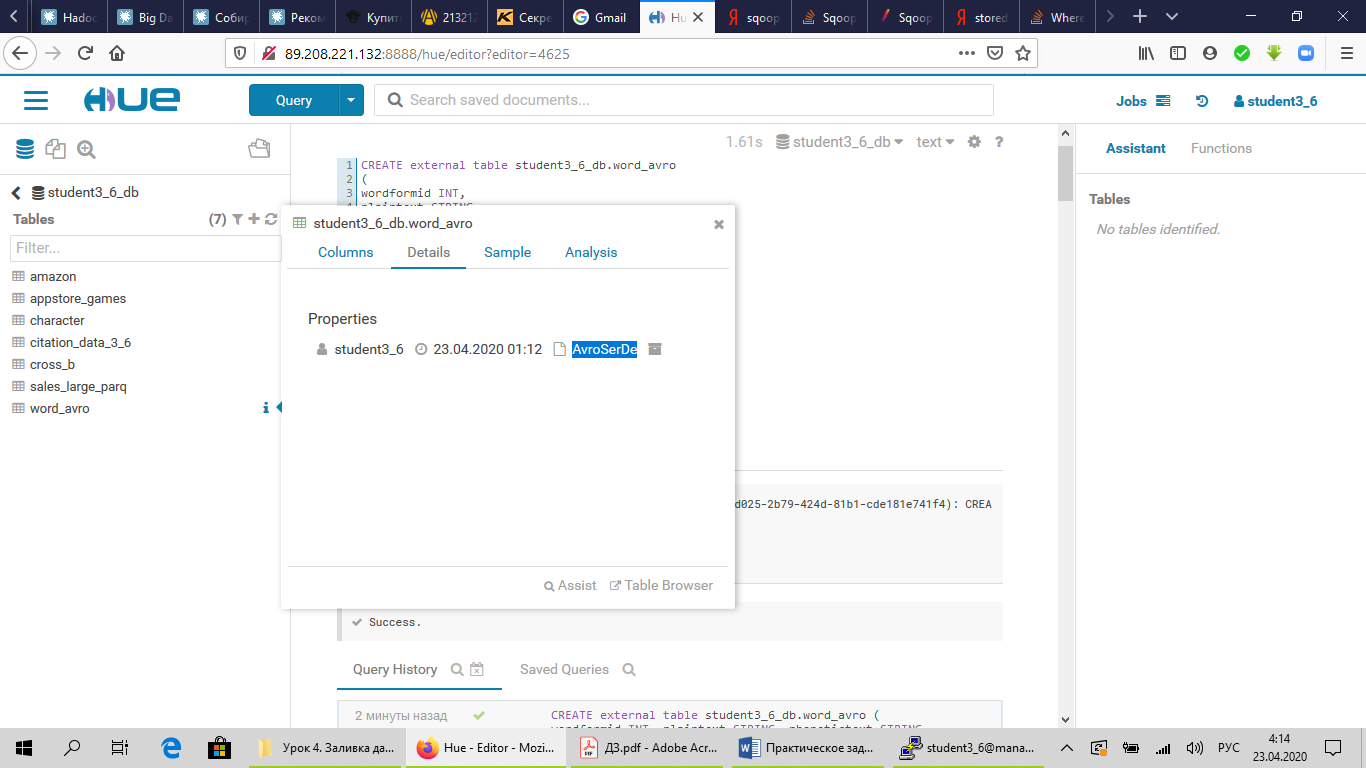
sqoop import --connect jdbc:postgresql://node3.novalocal/pg\_db --username exporter --password exporter\_pass --table wordform --hive-table wordform\_avro --as-avrodatafile --target-dir /student3\_6/word\_db –append



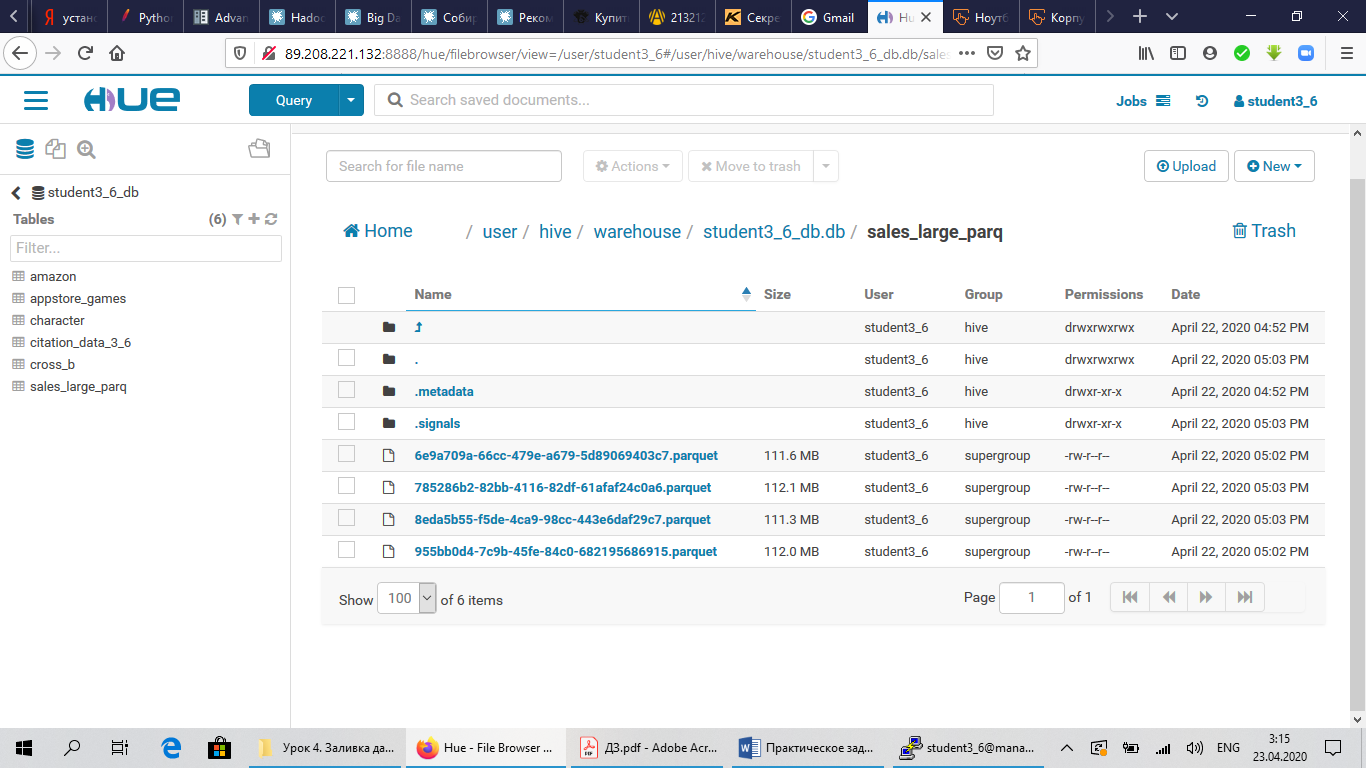


Далее закидываю данные через Hive:





4. Найдите папки на файловой системе куда были сохранены данные. Посмотрите их размер.



5. Сделайте несколько произвольных запросов к этим таблицам.

6. [Продвинутое задание] Сделать тоже самое с любой другой таблицей в любой другой БД вне кластера. Это задание автоматически покрывает предыдущие пять пунктов -- если сделаете, то пункты 1-5 не обязательны :)

Пример запуска SQOOP

Импорт:

sqoop import --connect jdbc:postgresql://node3.novalocal/pg\_db --username exporter -P --table character --hive-import --hive-database default --hive-table character

Посмотреть в схему:

sqoop import --connect jdbc:postgresql://node3.novalocal/pg\_db --username exporter -P --table character --hive-import --hive-database default --hive-table character

Доступ к Postgres внутри кластера

Хост 89.208.222.201

Порт 5432

БД pg\_db

Пользователь exporter

Пароль exporter\_pass