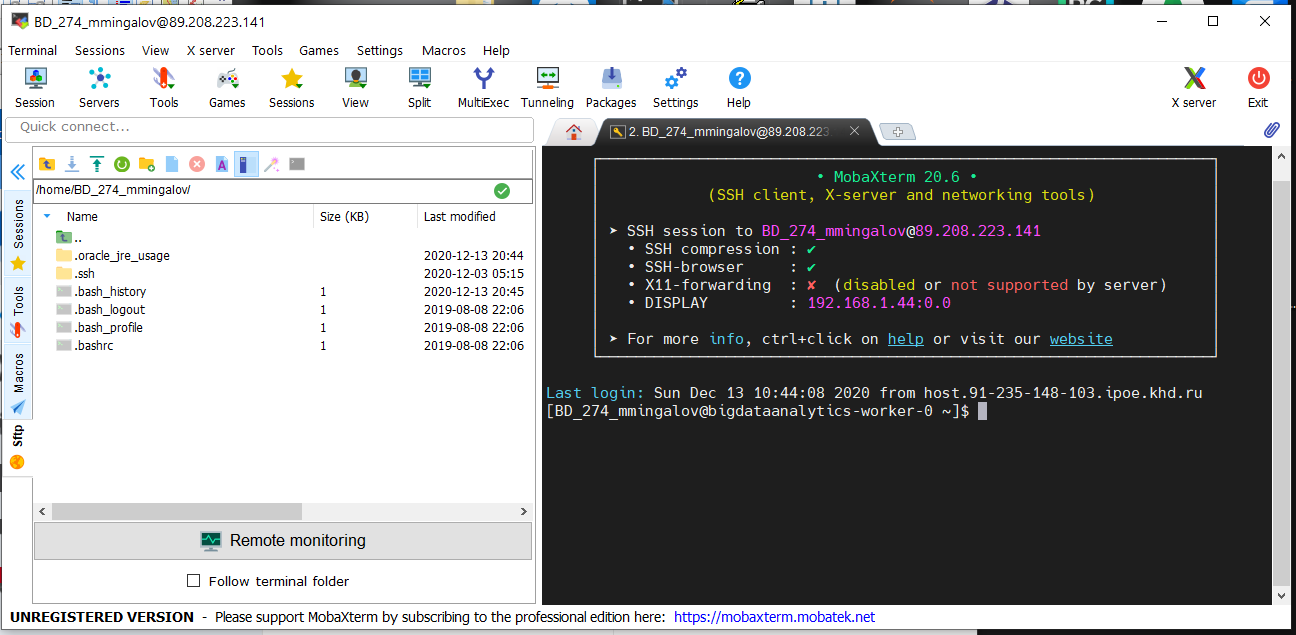
**ЗАДАНИЕ 1: Повторить чтение файлов со своими файлами со своей схемой.**

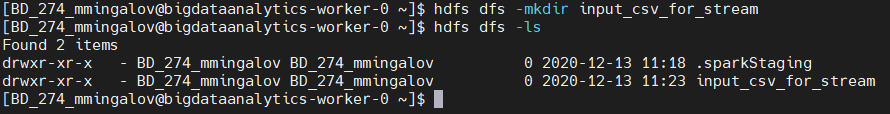
1.

Запускаем mobaexterm

Выбираем Putty-сессию на worker-2



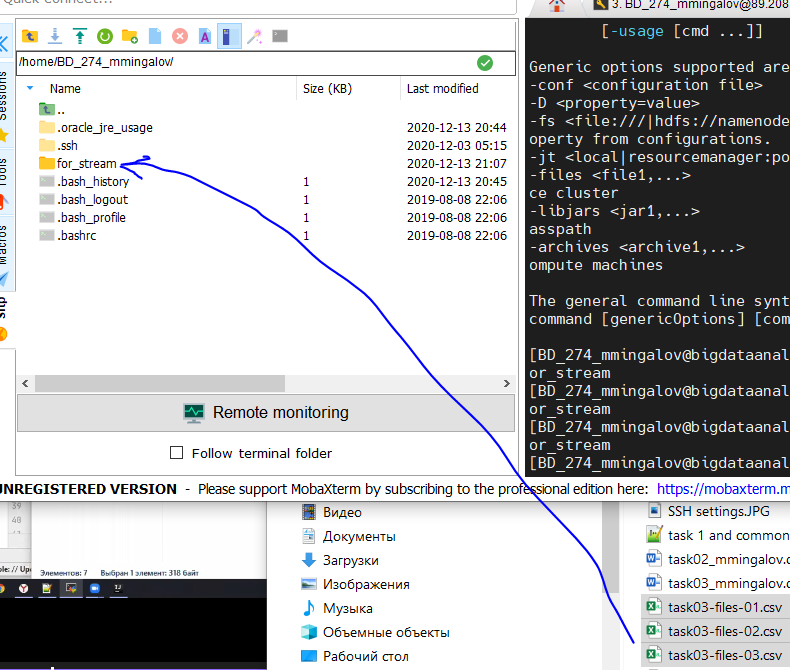
----  
создадим директорию в HDFS



-----

Создадим локальную директорию **for\_stream**/ командой **mkdir for\_stream**

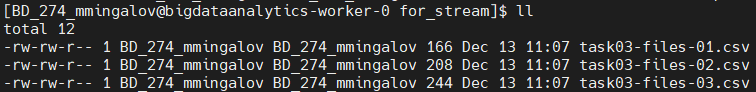
Подготовленные файлы методом drag&drop кидаем в локальную директорию for\_stream на worker-2



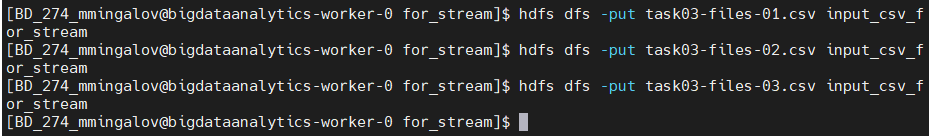
Переходим в эту директорию



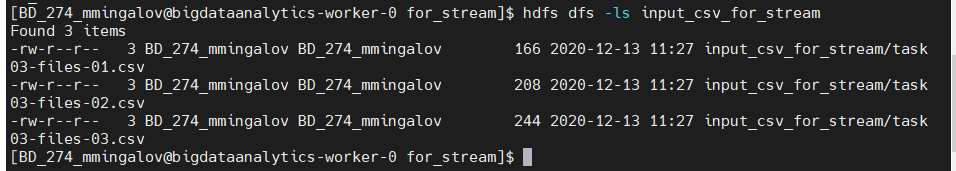
Смотрим файлы в ней такой командой:



----  
копируем файлы в HDFS директорию



Проверим



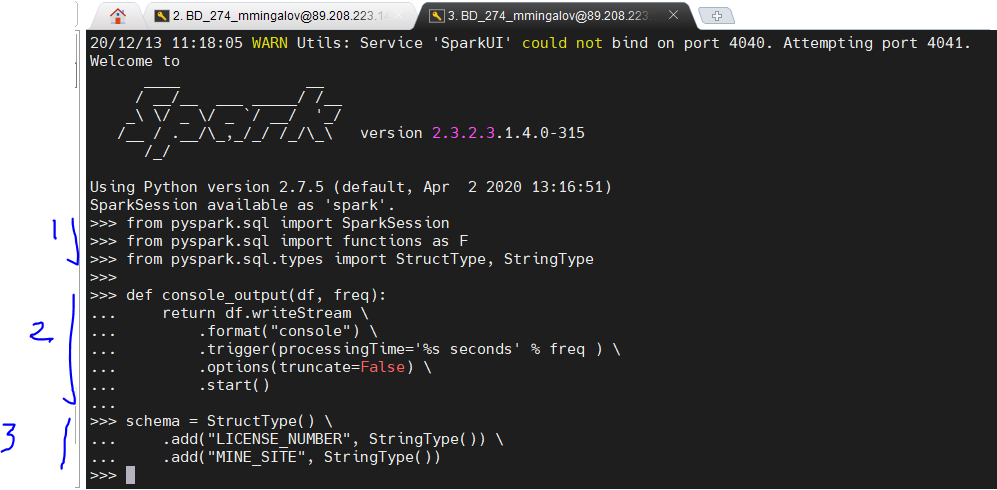
2.

ОТКРОЕМ ОТДЕЛЬНУЮ ВКЛАДКУ и выполним следующие сценарии

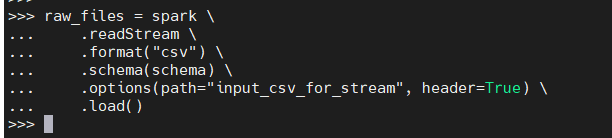
Для начала:

**pyspark**

далее выполняем:

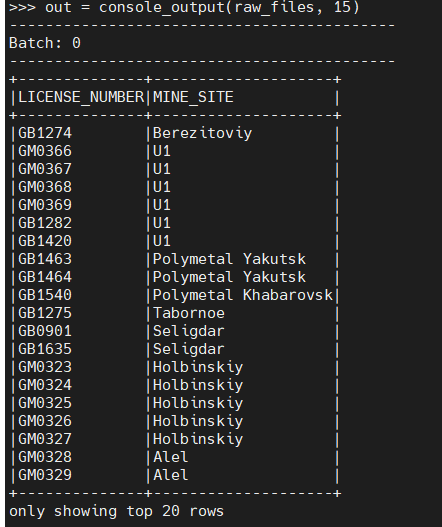


Далее



Запускаем процесс

**out = console\_output(raw\_files, 15)**



**Были обработаны все файлы за раз (отображено 20 строк только).**

**Важно:** если мы сейчас из директории input\_for\_csv все файлы удалим, а затем опять их туда поместим через -put, то наш stream ничего с ними не сделает. Он знает, что уже читал их

Stream постоянно проверяет что он уже сделал

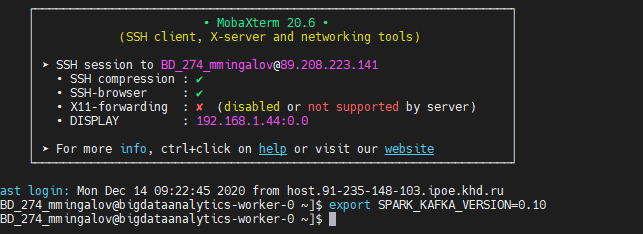
**ЗАДАНИЕ 2: Создать свой топик/топики, загрузить туда через консоль осмысленные данные с kaggle. Лучше в формате json. Много сообщений не нужно, достаточно штук 10-100.**

**Прочитать свой топик так же, как на уроке.**

Начинаем с того, что создаем новую сессию.

Даем команду:

*export SPARK\_KAFKA\_VERSION=0.10*



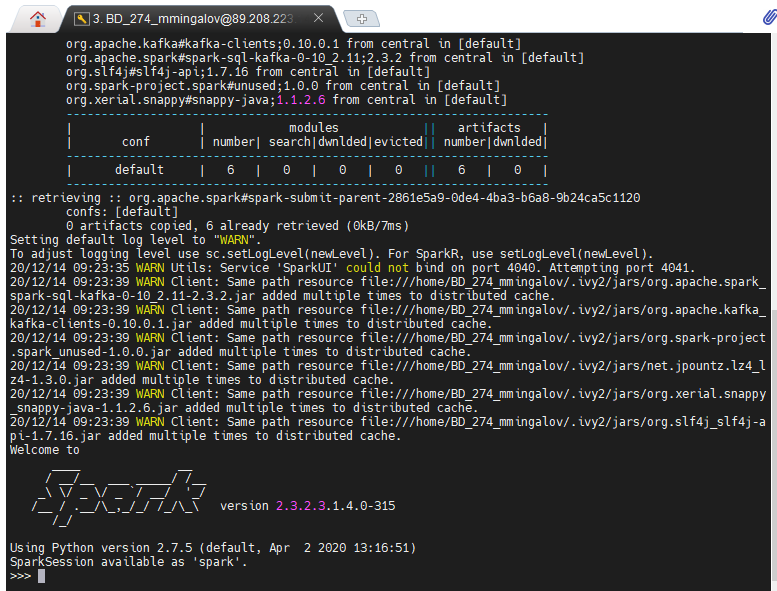
Далее запускаем **pyspark** с указанием нужного нам **packages.** Это нужно для того, чтобы Spark увидел новые функции, которые мы будем использовать

(использование packages означает, что он сам найдет его в интернете и скачает. Т.е. не нужно как в случае с jar предварительно качать их себе на локалку)

Итак, начинаем командой:

*pyspark --packages org.apache.spark:spark-sql-kafka-0-10\_2.11:2.3.2*

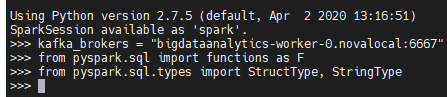
Packages успешно скачался:



Добавляем kafka-брокер, импортируем типы объектов:

kafka\_brokers = **"bigdataanalytics-worker-0.novalocal:6667"**

**from** pyspark.sql **import** functions **as** F  
**from** pyspark.sql.types **import** StructType, StringType



*#функция, чтобы выводить на консоль вместо show()***def** console\_output(df, freq):  
 **return** df.writeStream \  
 .format(**"console"**) \  
 .trigger(processingTime=**'%s seconds'** % freq ) \  
 .options(truncate=**True**) \  
 .start()  
  
*#читаем без стрима*raw\_orders = spark.read. \  
 format(**"kafka"**). \  
 option(**"kafka.bootstrap.servers"**, kafka\_brokers). \  
 option(**"subscribe"**, **"orders\_json"**). \  
 option(**"startingOffsets"**, **"earliest"**). \  
 load()

Прочитаем командой .show(), посмотрим схему: :

