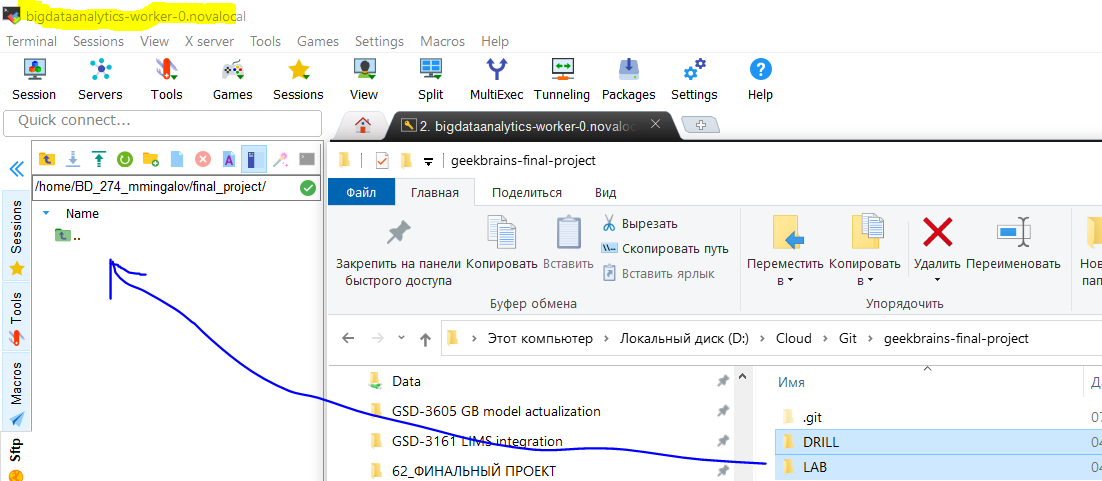
# **ПОДГОТОВКА ДАННЫХ**

**Открываем сессию mobaXterm**

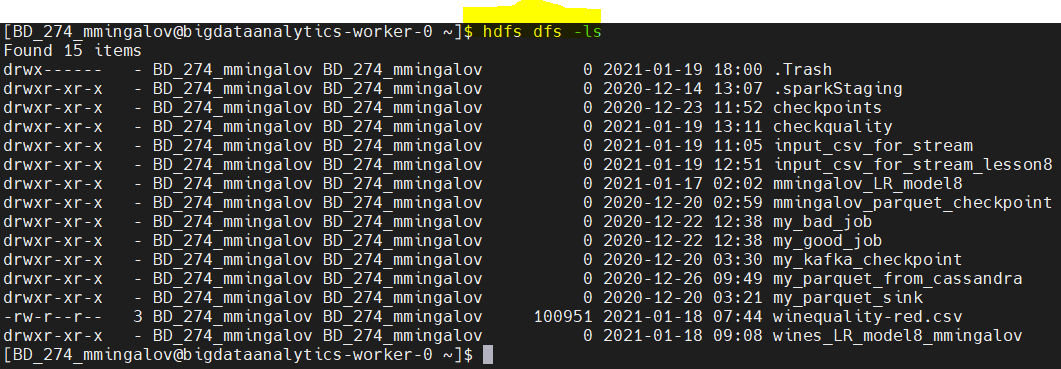
**Выполняем drag&drop копирование файлов в директорию проекта:**



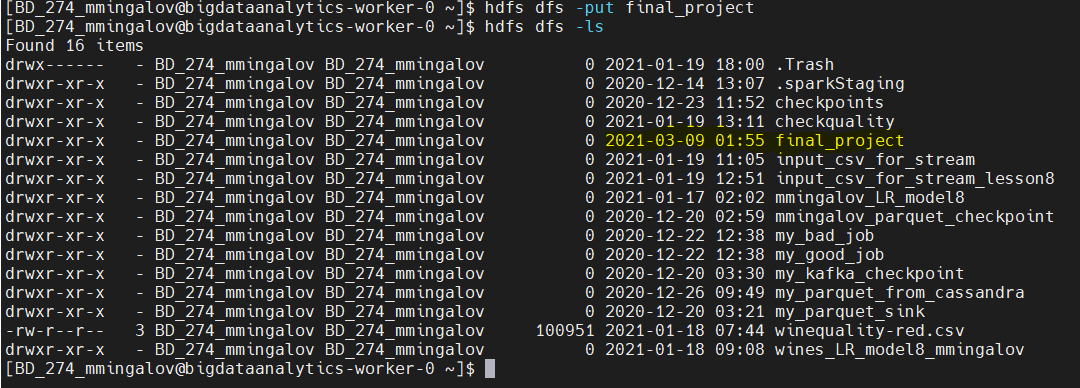
**Файлы скопированы в локальный каталог worker-0**

**Нам необходимо перенести их в HDFS**

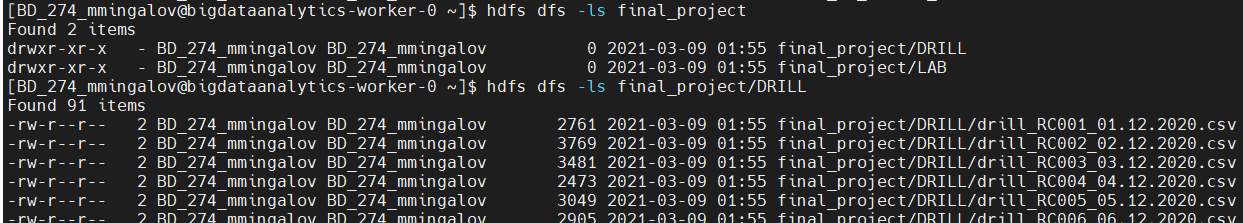
**Посмотрим, что за папки на данный момент есть в HDFS:**



**Создадим директорию под финальный проект: просто перенесем final\_project в HDFS командой** **put**



**Посмотрим внутрь..**



**Видим, что данные на месте и готовы к дальнейшей работе.**

# **ЧТЕНИЕ ДАННЫХ**

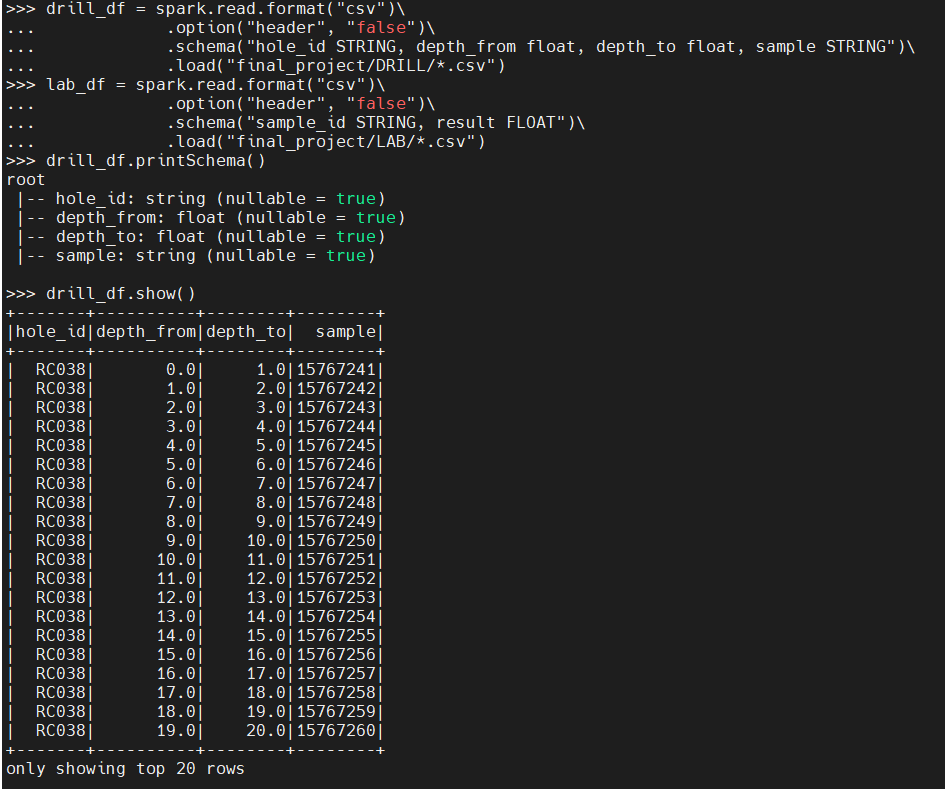
**Примечание: нижеприведенный код есть в проекте в виде .py файла**

**Запускаем SPARK командой:**

**export SPARK\_KAFKA\_VERSION=0.10**

**/spark2.4/bin/pyspark --packages org.apache.spark:spark-sql-kafka-0-10\_2.11:2.4.5 --driver-memory 512m --num-executors 1 --executor-memory 512m --master local[1]**

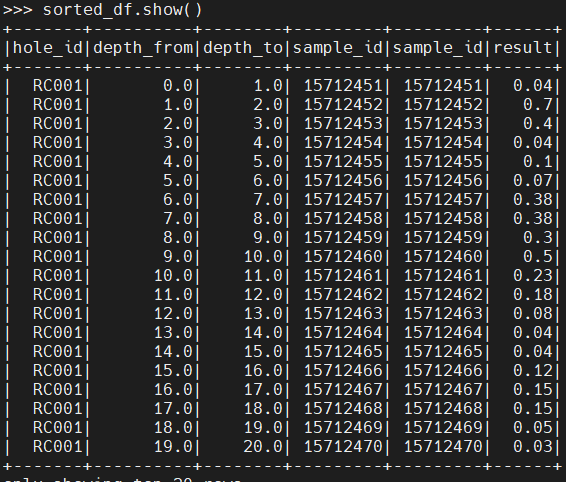
drill\_df = spark.read.format("csv")\  
 .option("header", "false")\  
 .schema("hole\_id STRING, depth\_from float, depth\_to float, sample\_id STRING")\  
 .load("final\_project/DRILL/\*.csv")  
lab\_df = spark.read.format("csv")\  
 .option("header", "false")\  
 .schema("sample\_id STRING, result FLOAT")\  
 .load("final\_project/LAB/\*.csv")



# **СЛИЯНИЕ ДАННЫХ**

**Выполняем слияние двух spark\_df в общий датасет, сортируя по «номеру скважины» и глубине «ОТ»:**

join\_df = drill\_df\  
 .join(lab\_df,lab\_df.sample\_id == drill\_df.sample\_id,"left")  
  
sorted\_df = join\_df\  
 .sort("hole\_id","depth\_from")  
  
sorted\_df.show()



# **РАСЧЕТ КОМПОЗИТОВ (UDF)**

**Для расчета композитов нам понадобятся вспомогательные столбцы.**

**Будем рассчитывать их при помощи пользовательских функций**

**UDF**

from pyspark.sql.functions import udf

**при расчете композитов у нас есть глобальные параметры: содержание по борту и мощность рудного интервала**

Cb = 0.37  
Mr = 2

**Для определения композита понадобится выполнить 4 шага**

# # Create step1 function  
def f\_step1(s):  
 if s < Cb:  
 result = 0 #'П'  
 else:  
 result = 1 #'Р?'  
 return result  
#assigning UDF function  
udf1 = udf(f\_step1)

df = join\_df \  
 .withColumn("step1", udf1(join\_df["result"])) \  
 .withColumn("step2", lit("П")) \  
 .withColumn("step3", lit("П")) \  
 .withColumn("composite", lit("")) \  
 .sort("hole\_id", "depth\_from") \  
 .show()