# **Домашнее задание №4**

БПИ227 Артемьев Александр

Следующая инструкция представляет из себя пошаговое руководство по развертыванию Apache Spark. После выполнения третьего дз у нас уже есть развернутый на 3 нодах hdfs и узлы со следующими адресами:

jn\_glob\_ip – узел для входа в jn

jn\_local\_ip – локальный адрес jn

nn\_ip – локальный адрес nn

dn-00\_ip – локальный адрес dn-00

dn-01\_ip – локальный адрес dn-01

user\_name – имя пользователя, который может подключиться к jn

А также развернутый на этих узлах YARN, с веб интерфейсом и Apache Hive.

Порядок действий:

1) Для начала необходимо проверить, что поднятая структура с hadoop, yarn и hive уже успешно функционирует. Например, проверкой логов или веб-интерфейса.

2) Подключаемся к серверу

ssh <user\_name>@<jn\_glob\_ip>

3) Для начала установим python venv и python pip, чтобы запустить Spark

sudo apt install python3-venv

sudo apt install python3-pip

4)Переходим на пользователя hadoop и скачиваем дистрибутив Spark

sudo -i -u hadoop

wget <https://archive.apache.org/dist/spark/spark-3.5.3/spark-3.5.3-bin-hadoop3.tgz>

5) Распоковываем архив с дистрибутивом  
tar -xzvf spark-3.5.3-bin-hadoop3.tgz

6) Объявим ряд переменных:

Папка с конфигом hadoop

export HADOOP\_CONF\_DIR="/home/hadoop/hadoop-3.4.0/etc/hadoop"

Расположение Hive

export HIVE\_HOME="/home/hadoop/apache-hive-4.0.1-bin"

Доп библиотеки Hive

export HIVE\_CONF\_DIR=$HIVE\_HOME/conf

export HIVE\_AUX\_JARS\_PATH=$HIVE\_HOME/lib/\*

Добавляем исполняемые файлы hive в путь

export PATH=$PATH:$HIVE\_HOME/bin

Указать локальный ip Spark

export SPARK\_LOCAL\_IP=<jn\_local\_ip>

Указываем необходимые Spark библиотеки

export SPARK\_DIST\_CLASSPATH="/home/hadoop/spark-3.5.3-bin-hadoop3/jars/\*:/home/hadoop/hadoop-3.4.0/etc/hadoop:/home/hadoop/hadoop-3.4.0/share/hadoop/common/lib/\*:/home/hadoop/hadoop-3.4.0/share/hadoop/common/\*:/home/hadoop/hadoop-3.4.0/share/hadoop/hdfs:/home/hadoop/hadoop-3.4.0/share/hadoop/hdfs/lib/\*:/home/hadoop/hadoop-3.4.0/share/hadoop/hdfs/\*:/home/hadoop/hadoop-3.4.0/share/hadoop/mapreduce/\*:/home/hadoop/hadoop-3.4.0/share/hadoop/yarn:/home/hadoop/hadoop-3.4.0/share/hadoop/yarn/lib/\*:/home/hadoop/hadoop-3.4.0/share/hadoop/yarn/\*:/home/hadoop/apache-hive-4.0.0-alpha-2-bin/\*:/home/hadoop/apache-hive-4.0.0-alpha-2-bin/lib/\*"

7) Переходим в папку со Spark

cd spark-3.5.3-bin-hadoop3/

8) И объявляем еще ряд переменных для Spark

export SPARK\_HOME=`pwd`

export PYTHONPATH=$(ZIPS=("$SPARK\_HOME"/python/lib/\*.zip); IFS=:; echo "${ZIPS[\*]}"):$PYTHONPATH

Добавляем исполняемые файлы Spark в path

export PATH=$SPARK\_HOME/bin:$PATH

9) Теперь создадим новое виртуальное окружение

cd ..

python3 -m venv venv

10) И активируем созданное окружение

source venv/bin/activate

11) В данном окружение обновим pip, установим ipython и onetl

pip install -U pip

pip install ipython

pip install onetl[files]

12) Теперь запустим сессию Spark для преобразования csv, для этого запускаем интерактивную оболочку питона и выполняем следующие команды:

ipython

Модуль, отвечающий за создание Spark сессии

from pyspark.sql import SparkSession

from pyspark.sql import functions as F

Объект подключение к hdfs

from onetl.connection import SparkHDFS

Подключение к hive

from onetl.connection import Hive

Чтобы читать файлы в dataframe

from onetl.file import FileDFReader

from onetl.file.format import CSV

запись данных через hive

from onetl.db import DBWriter

Создание сессии spark

spark = SparkSession.builder.master("yarn").appName("spark-with-yarn").config("spark.sql.warehouse.dir", "/user/hive/warehouse").config("spark.hive.metastore.uris", "thrift://jn:9083").enableHiveSupport().getOrCreate()

Подключение к hdfs

hdfs = SparkHDFS(host="nn", port=9000, spark=spark, cluster="test")

С помощью reader можем прочитать данные

reader = FileDFReader(connection=hdfs, format=CSV(delimiter=",", header=True), source\_path="/input")

Читаем данные из .csv файла

df = reader.run(["<название csv файла>"])

Применение трансформаций данных

df = df.orderBy([<столбцы сортировки>], ascending=[<возрастание/убывание для столбца >])

Создаем объект для подключения к hive

hive = Hive(spark=spark, cluster="test")

writer = DBWriter(

connection=hive,

table="filename",

options={

"if\_exists": "replace\_entire\_table",

"partition\_by": “<столбцы партицирования>”

}

)

writer.run(df)

Завершаем работу Spark

spark.stop()

Мои данные:

Env team-1

User team

пароль для входа x=T35T\_sMdm4

узел для входа 176.109.91.3

jn 192.168.1.6

nn 192.168.1.7

dn-00 192.168.1.8

dn-01 192.168.1.9