**درس : Big Data استاد : خانم دکتر صفری**

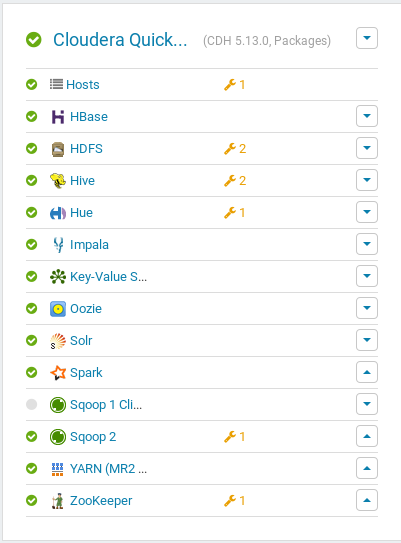
**دانشجو : رقیه یزدانی تمرین : Spark**

# Spark

# ایجاد بستر لازم برای امکان کار با Spark

قبلاً فایل cloudera-quickstart-vm-5.13.0-0-vmware که در واقع ماشین مجازی cloudera می‌باشد، دانلود شده و با استفاده از نرم افزار VMWare Player 15.5 امکان کار با آن فراهم شده است.

برای این تمرین، لازم است که cloudera manager(CM) را launch کرده باشیم که قبلاً در انجام تمرین Hbase این کار انجام شد. همچنین لازم است که سرویس spark بالا باشد که اگر اینگونه نیست میتوان سرویس آن را بصورت دستی از از طریق اینترفیس، start کرد.



برای انجام تمرینها، قرار است با دیتاست هایی که برای بخش join از تمرینهای MapReduce استفاده شد، کار شود.

# تمرین اول و انجام Simple Join

دو فایل داریم به نامهای join1\_FileA.txt و join1\_FileB.txt که در ادامه محتوای هر کدام آورده شده است. هر سطر از فایل اول، شامل یک کلمه به عنوان key و یک عدد به عنوان value است که با , از هم جدا شده‌اند و هر سطر از فایل دوم، شامل یک تاریخ و یک کلمه به عنوان key و یک عدد بعنوان value می‌باشد که با , از هم جدا شده‌اند.

Join1\_FileA.txt

able,991

about,11

burger,15

actor,22

join1\_FileB.txt

Jan-01 able,5

Feb-02 about,3

Mar-03 about,8

Apr-04 able,13

Feb-22 actor,3

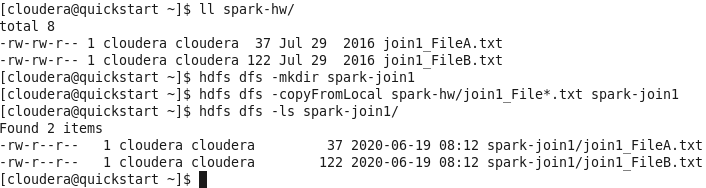
Feb-23 burger,5

Mar-08 burger,2

Dec-15 able,100

هدف این است که این دو فایل روی "کلمه" با هم join شوند و یک خروجی ترکیبی ایجاد شود. مراحل کار بصورت زیر است :

* انتقال این فایلها به hdfs در ماشین cloudera که برای اینکار می‌توانیم ابتدا این دو فایل را از طریق ابزاری مثل Bitwise SSH client به ماشین مجازی cloudera منتقل کرده و سپس مشابه روشی که در تمرین Hadoop استفاده می‌شد، فایلها را به hdfs کپی کرد.



* ورود به spark shell و ایجاد RDD بر اساس محتویات فایل join1\_FileA.txt و سپس اجرای collect

*fileA = sc.textFile("spark-join1/join1\_FileA.txt")*

*fileA.collect()*

* ایجاد RDD بر اساس محتویات فایل join1\_FileB.txt و سپس اجرای collect

*fileB = sc.textFile("spark-join1/join1\_FileB.txt")*

*fileB.collect()*

* ایجاد یک تابع به زبان پایتون با نام split\_fileA که یک سطر از فایل join1\_FileA.txt را به عنوان ورودی دریافت کرده و زوج (مقدار،کلید) را بر اساس آن بر می گرداند :

*def split\_fileA(line):*

*line = line.split(",")*

*word = line[0]*

*count = line[1]*

*return (word, count)*

* درستی عملکرد تابع بالا را با یک عبارت تستی مشابه”yazdani,123” بررسی می کنیم.

*test\_line = " yazdani,123"*

*split\_fileA(test\_line)*

* اجرای عمل map بر اساس تابع بالا (split\_fileA) روی داده های RDD با نام fileA که در مراحل قبل ایجاد شد.

*fileA\_data = fileA.map(split\_fileA)*

*fileA\_data.collect()*

* ایجاد یک تابع به زبان پایتون با نام split\_fileB که یک سطر از فایل join1\_FileB.txt را به عنوان ورودی دریافت کرده و زوج (مقدار،کلید) را بر اساس آن بر می‌گرداند بصورتی که کلید، "کلمه" می‌باشد و مقدار، ترکیب "عدد تاریخ" می‌باشد.

*def split\_fileB(line):*

*line = line.split(",")*

*key\_in = line[0].split(" ")*

*count = line[1]*

*date = key\_in[0]*

*word = key\_in[1]*

*return (word, date + " " + count)*

* درستی عملکرد تابع بالا را با یک عبارت تستی بررسی می کنیم.

*test\_line = "Jan-01 yazdani,123"*

*split\_fileB(test\_line)*

* اجرای عمل map بر اساس تابع بالا (split\_fileB) روی داده های RDD با نام fileB

*fileB\_data = fileB.map(split\_fileB)*

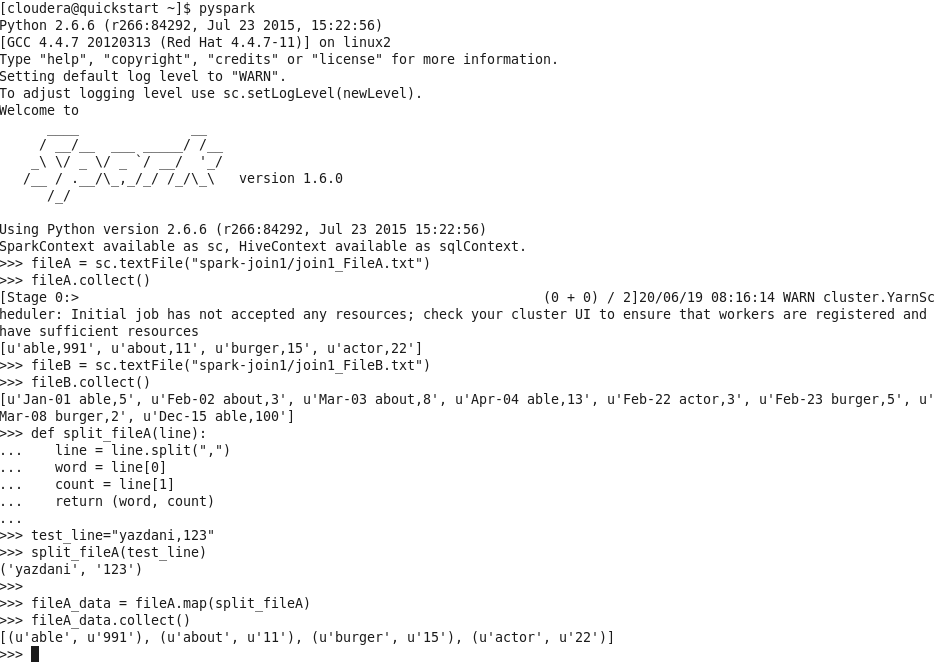
*fileB\_data.collect()*

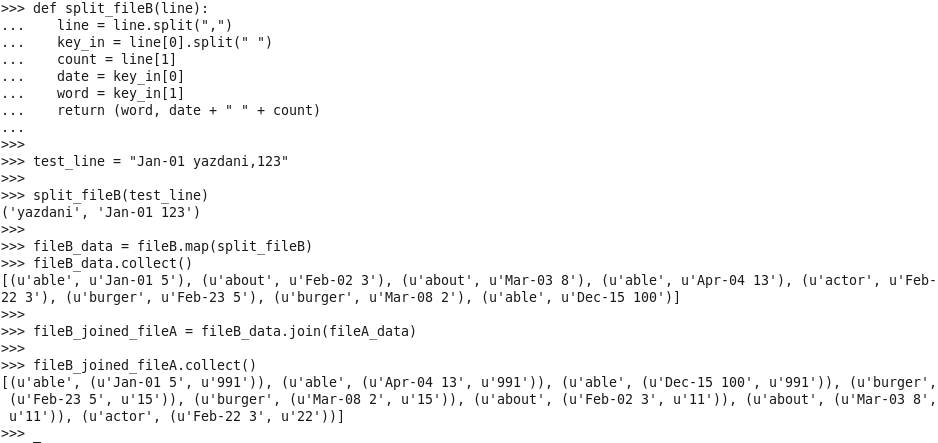
* اجرای عمل join روی دو RDD با نامهای fileA\_data و fileB\_data و سپس اجرای collect

*fileB\_joined\_fileA = fileB\_data.join(fileA\_data)*

*fileB\_joined\_fileA.collect()*

نتیجه اجرای تمام مراحل بالا در شکلهای زیر آورده شده است :





بر اساس نتیجه join در تصویر بالا، برای کلمه actor، زوج (‘actor’,(‘Feb-22 3’,’22’)) تولید شده است.

# تمرین دوم و انجام Advanced Join

برای این بخش، برنامه پایتون به نام make\_join2data.py در اختیار قرار داده شده تا بکمک آن بتوان فایلهای داده را ایجاد کرد. کد این برنامه در ادامه آمده است :

make\_join2data.py

#!/usr/bin/env python

import sys

# --------------------------------------------------------------------------

# (make\_join2data.py) Generate a random combination of titles and viewer

#    counts, or channels

# this is a simple version of a congruential generator,

# not a great random generator but enough

# --------------------------------------------------------------------------

chans = ['ABC','DEF','CNO','NOX','YES','CAB','BAT','MAN','ZOO','XYZ','BOB']

sh1 =['Hot','Almost','Hourly','PostModern','Baked','Dumb','Cold','Surreal'

    ,'Loud']

sh2 =['News','Show','Cooking','Sports','Games','Talking','Talking']

vwr =range(17,1053)

chvnm=sys.argv[1] #get number argument, if its n, do numbers not channels,

lch=len(chans)

lsh1=len(sh1)

lsh2=len(sh2)

lvwr=len(vwr)

ci=1

s1=2

s2=3

vwi=4

ri=int(sys.argv[3])

for i in range(0,int(sys.argv[2])): #arg 2 is the number of lines to output

if chvnm=='n': #no numuber

print('{0}\_{1},{2}'.format(sh1[s1],sh2[s2],chans[ci]))

else:

print('{0}\_{1},{2}'.format(sh1[s1],sh2[s2],vwr[vwi]))

ci=(5\*ci+ri) % lch

s1=(4\*s1+ri) % lsh1

s2=(3\*s1+ri+i) % lsh2

vwi=(2\*vwi+ri+i) % lvwr

if (vwi==4): vwi=5

این برنامه داده‌هایی تصادفی ایجاد می‌کند که با شش بار اجرای آن بصورت زیر، شش فایل تولید خواهد شد :

python make\_join2data.py y 1000 13 > join2\_gennumA.txt

python make\_join2data.py y 2000 17 > join2\_gennumB.txt

python make\_join2data.py y 3000 19 > join2\_gennumC.txt

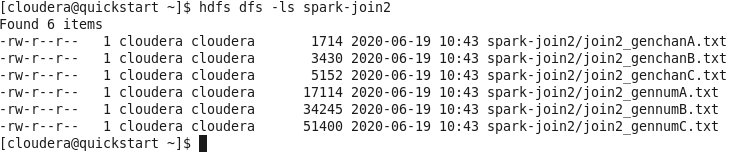
python make\_join2data.py n 100 23 > join2\_genchanA.txt

python make\_join2data.py n 200 19 > join2\_genchanB.txt

python make\_join2data.py n 300 37 > join2\_genchanC.txt

هر سطر از فایلهای gennum حاوی نام یک برنامه تلویزیونی و تعداد دفعات تماشای آن (views) است و هر سطر از فایلهای genchan شامل نام برنامه تلویزیونی و channel پخش آن می‌باشد. هدف این است که مجموع تماشاگران کلیه برنامه های تلویزیونی کانال BAT را بدست آوریم. مراحل کار بصورت زیر است :

* انتقال فایلهای داده تولید شده به hdfs در ماشین cloudera مشابه تمرین 1



* ایجاد RDD بر اساس محتویات فایلهای gennum و سپس اجرای take برای رویت نمونه هایی از آن

*show\_views\_file = sc.textFile("spark-join2/join2\_gennum?.txt")*

*show\_views\_file.take(3)*

* نوشتن تابعی به زبان پایتون برای تبدیل هر سطر از فایلهای gennum به زوج (show,views)

*def split\_show\_views(line):*

*line = line.split(",")*

*show = line[0]*

*views = line[1]*

*return (show,int(views))*

* اجرای map روی show\_views\_file

*show\_views = show\_views\_file.map(split\_show\_views)*

*show\_views.take(3)*

* ایجاد RDD بر اساس محتویات فایلهای genchan و سپس اجرای take برای رویت نمونه هایی از آن

*show\_channel\_file = sc.textFile("spark-join2/join2\_genchan?.txt")*

*show\_channel\_file.take(3)*

* نوشتن تابعی برای تبدیل هر سطر از فایلهای genchan به زوج (show,(channel,views))

*def split\_show\_channel(line):*

*line = line.split(",")*

*show = line[0]*

*channel = line[1]*

*return (show,channel)*

* اجرای map روی show\_channel\_file

*show\_channel = show\_channel\_file.map(split\_show\_channel)*

*show\_channel.collect()*

* اجرای join روی دو RDD

*joined\_dataset = show\_views.join(show\_channel)*

* نوشتن تابعی برای استخراج زوج (channel,view) از هر عضو دیتاست نتیجه join

*def extract\_channel\_views(show\_views\_channel):*

*show = show\_views\_channel[0]*

*views = show\_views\_channel[1][0]*

*channel = show\_views\_channel[1][1]*

*return (channel,int(views))*

* اجرای map

*channel\_views = joined\_dataset.map(extract\_channel\_views)*

* اجرای reduceByKey

*channel\_views.reduceByKey(lambda x,y:x+y).collect()*

**اجرای کلیه مراحل در تصویر زیر آمده و بر اساس نتیجه اجرای آخرین دستور، آمار تماشاگران کانال BAT تعداد 5099141 بدست آمده است.**

