**CÔNG VIỆC CỦA NGÀY 10/05+11/05**

**Spark**

Spark là framework để xử lý dữ liệu ETL.

Được xây dựng bằng ngôn ngữ scala

Cung cấp các APIs cho scala, java, python, R

Tốc xử nhanh hơn MapReduce 10x, 100x.

Như chúng ta biết:

Nếu xử lý dữ liệu theo batch: MapReduce

Nếu xử lý dữ liệu stream data: Apache Storm/s4

Xử lý tương tác: Apache Impala, Apache Tez

Xử lý dữ liệu đồ thị : Apache Giraph

Apache tích hợp xử lý các nhiệm vụ trên

Các đặc trưng của Spark

Xử lý nhanh

Lưu trên ram

Khả năng chống chịu lỗi

Qúa trình truyền dữ liệu real time

Cơ chế lazy evaluation

Support nhiều ngôn ngữ

Có chi phí hợp lý

Spark components:

Spark core: cung cấp platform cho các spark application

Spark sql: Xử lý dữ liệu cấu trúc và bán cấu trúc bằng HQL

Spark Stream: Xử lấy dữ liệu stream.

Spark Ml: Cung cấp công cụ cho việc phân tích dữ liệu bằng các model machine learning

Spark GraphX: Xử lý dữ liệu dựa trên lý thuyết đồ thị

Spark R: Xử lý dữ liệu trên R

, spark streaming, spark ml, spark graphx, spark R

Cài đặt Spark trên cmd.

Chạy dòng lên: spark-shell

Giúp chạy các dòng lên scala

Mở website: localhost:4040

Ngôn ngữ scala được viết dựa trên python và java

Khá giống với java nhưng mà gọn hơn java

Spark Context

Là một thực thể của môi trường xử lý dữ liệu

Cho phép các đặt các dịch vụ trung gian, thiết lập các connection tới spark Enviroment chẳng hạn như cấu hình, app name, master URL

Khi 1 spark context chỉ có thể chạy trên 1 máy ảo JVM, nếu muốn tạo ra spark context mới thì cần close spark context hiện tại

RDD (Resillient distributed dataset)

Là một kiểu dữ liệu trên apache spark

RDDs là immutable data

Được xử lý trên memory(ram)

Được thực thi theo cơ chế lazy evaluation

Chịu lỗi

Partitioning

Các cách tạo ra RDDs

Load dữ liệu từ storage

Được tạo ra bởi spark driver: sc.parallelize

Được tạo ra bởi 1 RDD khác

Code:

Xóa các log không cần thiết

Logger.getLogger(“org”).setLevel(Level.ERROR)

Set up SparkContext

Val conf = new SparkConf().setAppName(“TenApp”).SetMaster(“local[\*]”)

Val sc = new.SparkContext(conf)

Tạo ra RDDs

C1

Val data1 = sc.textFile(“url của file”,6)

Data1.foreach(println)

C2 Được tạo ra bởi spark driver

Val data = sc.parallelize(Seq(1,2,3,4,5))

Data2.foreach(println)

C3 Được tạo ra bởi 1 RDD khác

Val data3 = data1.flatMap(ele => ele.split(“ ”)).map(x => (x,1))

Data.foreach(println)

Xem và chỉnh lại repartition()

Print(data.getNumPartitions)

Val data1 = data.repartition(3)

Trong spark RDD có kiểu operators

Transformation: lazy evaluation

Narrow transformation: map, flatMap, Union, Sample, mapPartition, Filter

Wide transformation: groupByKey, reduceByKey, combineByKey, Distinct, Intersect, Join, Repartition, Coalesce

Action: count, collect, take, top, first, foreach

ReduceByKey

Val data3 = data1.flatMap(ele => ele.split(“ ”)).map(x => (x,1))

.sortByKey(true)

.reduceByKey(\_+\_)

Collect()khi in ra không bị shuffle

Data.collect().foreach(println)

Union() nối dữ liệu

Val data3 = data1.Union(data2)

Sample() lấy 1 sample từ 1 tập dữ liệu

Val data 3 = data1.sample(true, 0.5)

Data.collect().sorted().foreach(println)

Lấy có hoàn lại và lấy 50% bộ gốc

Filter

Val data3 = data1.filter(ele => ele%10 == 0)

Take lấy bao nhiêu dữ liệu

Println(Data3.collect().take(4))

ReduceByKey và GroupByKey

Val data3 = data1. groupByKey ().map(ele=> (ele.\_1, ele.\_2.sum))

.sortByKey(true)

Val data3 = data1. reduceByKey ((x,y)=> (x+y))

.sortByKey(true)

Join() nối (1,a), (1,b) thành (1, (a,b))

Intersect() trả về các phần tử giống nhau

Distinct() trả về tập phần tử không trùng lập

toDebugString Xem debug như thế nào

Spark Session: Xuất hiện từ phiên bản spark 2.0 -> cải tiến của Spark context

Thay vì sử dụng nhiều loại context khác nhau, sử dụng spark Session

Có thể dùng cho HQL

toDF chuyển thành Dataframe

Persist() và cache()

Là kỹ thuật được sử dụng trong tối ưu hiệu năng tính toán

Thuận tiện cho việc tính song song

Cache() lưu kết quả trong ram

Persist() có nhiều lựa chọn, trong ram, trong disk, cả ram và disk

Persist(StorageLevel.MEMORY\_ONLY)