spark practical - 16
==========

cache & persist

consider if you have a rdd which you have generated by doing bunch of transformations..

rdd1

rdd2

rdd3

rdd4.cache

rdd5

rdd5.collect

rdd5.count

cache and persist both have same purpose.

if you want to reuse the results of existing rdd. then you can use them.

speed up the applications that access the same rdd multiple times.

an rdd that is not cached is re evaluated again each time an action is invoked.

the difference is that cache will cache the rdd in memory.

however persist comes with various storage levels.

in memory
in disk
off heap

persist() this is equivalent to cache() and this will persist in memory.

persist(StorageLevel.DISK_ONLY)

MEMORY_ONLY - data is cached in memory.

non serialized format.

serialization is in bytes format. - it takes less storage
non serialized means its in object format. - it takes more storage

DISK_ONLY - data is cached on disk in serialized format. in the form of bytes. and takes less storage.

MEMORY_AND_DISK - data is cached in memory. if enough memory is not available, evicted blocks from memory are serialized to disk.

this mode of operation is recommended when re-evaluation is expensive and memory resources are scarse.

OFF_HEAP - blocks are cached off-heap.

outside the JVM, grabing a piece of raw memory outside the executor.

problem with storing the objects in jvm is that it uses garbage collection for freeing up. garbage collection to free up the space is a time taking process.

but off_heap is a unsafe thing as we have to deal with raw memory outside your jvm.

Block eviction
==========

consider the situation that some of the block partitions are so large (skew) that they will quickly fill up the storage memory used for caching.

when the storage memory becomes full, an eviction policy will be used to make up the space for new blocks.

LRU (least recently used)

serialization increases the processing cost but reduces the memory foot print.

non serialized the processing can be a bit fast but it uses more memory foot print.

DISK_ONLY - SERIALIZED
MEMORY_ONLY - BY DEFAULT THESE ARE NON SERIALIZED
MEMORY_AND_DISK - BY DEFAULT THESE ARE NON SERIALIZED
OFF_HEAP - SERIALIZED
MEMORY_ONLY_SER - SERIALIZED
MEMORY_AND_DISK_SER - SERIALIZED

MEMORY_ONLY_2 - THIS NUMBER 2 INDICATES 2 REPLICAS STORED ON 2 DIFFERENT WORKER NODES.

REPLICATION IS USEFUL FOR SPEEDING UP RECOVERY IN CASE ONE NODE OF THE CLUSTER FAILS.



rdd.toDebugString is to check the lineage graph.
and we need to read it from bottom to top.



if we use cache() and we dont have enough memory then it will skip caching it. it wont give any error.

persist(StorageLevel.MEMORY_AND_DISK)

DO NOT CACHE OR PERSIST YOUR BASE RDD.

spark practical - 17
==========

1. difference between a DAG and a lineage

2. how to create a jar for your spark project and run the jar.

lineage is nothing but the dependency graph.

shows dependency of various rdd. it's a logical plan.

DAG is a acyclic graph.

jobs, stages and tasks.



how to run the jar
==========

spark-submit --class <class_name> <complete path of the jar>

./spark-submit --class WordCount
/Users/trendytech/Desktop/wordcount.jar

spark practical - 18
==========

Top movies

1. Atleast 1000 people should have rated for that movie..

2. average rating > 4.5

(101,Toy Story)
(101,4.7)



//input
(101,(Toy Story,4.7)) x

//output

x._2._1
Toy Story

spark practical - 19
==========

Spark Ecosystem

Structured API's - dataframes, datasets and spark sql
spark streaming to process real time data.

Map vs Map partition

this rdd has 10000 rows and 10 partitions.

each partition holds 1000 records.

func1

rdd.map(func1) - 10000 times

rdd.mapPartitions() - 10 times

http://apachesparkbook.blogspot.com/2015/11/mappartition-example.html

what all transformations and actions you have used?

https://spark.apache.org/docs/latest/rdd-programming-guide.html#transformations

==========