Week 2 Quiz

Bookmark this page

Shuffling

1/1 point (graded)

Which of the following are true about the shuffle operation in Spark?
You are encouraged to read more about shuffle here.

- [ ] All Spark operations involve shuffling
- [x] Shuffling requires large amounts of network communication which makes the operation expensive
- [x] Shuffling is Spark's mechanism for re-distributing data across partitions
- [x] To avoid shuffling, we should think about ways to leverage existing partitions. For example, leveraging partial aggregation to reduce data transfer
- [ ] No communication between worker nodes is required for the shuffling operation
- [ ] Shuffling can only work on a cached RDD
- [x] `groupByKey()` shuffles all the data, which is slow. Whereas, `reduceByKey()` shuffles only the results of sub-aggregations in each partition of the data, which is relatively faster.
- [ ] `reduceByKey()` shuffles all the data, which is slow. Whereas, `groupByKey()` shuffles only the results of sub-aggregations in each partition of the data, which is relatively faster.

Submit You have used 2 of 5 attempts Show Answer

Properties of RDDs

1/1 point (graded)

Which of the following are properties of a Spark RDD?

- [ ] Is mutable
- [x] Is immutable
- [x] Resides in memory by default
- [x] Is partitioned
- [ ] Resides only on master node
- [x] Resides on worker nodes
- [x] Is fault tolerant
- [ ] Individual Elements can be easily accesed
- [x] Supports Lazy Evaluation
- [ ] Gets cached as soon as the `cache()` method is called on it

Submit You have used 5 of 5 attempts Show Answer

Lazy Eval 1

1/1 point (graded)

The function `foo` takes 15µs to execute on a single element. We are using a cluster with 6 workers. An RDD with 1 million elements is mapped using `foo` and then passed through a `reduce` function which sums up the elements.

Which of these options is correct about the command `RDD.map(foo).reduce(lambda x,y:x+y)`

- [ ] The command will complete after about 15 seconds.
- [x] The command will complete after about 2.5 seconds.
- [ ] The `reduce` operation starts after all elements have beeen mapped.
- [x] The `map` operation is added to an execution plan that is executed after the reduce operation is called.

Submit You have used 2 of 5 attempts Show Answer

Lazy Eval 2

1/1 point (graded)

The function `foo` takes 15µs to execute for a single element input. An RDD with 1 million elements is mapped using `foo` and then passed through a `reduce` function which sums up the elements.

What is an estimate of the execution time (in seconds) for the `map - reduce` operation without accounting for the speedups provided by Spark?
You can calculate the rough estimate the same way as in the "Execution Plans Lazy Evaluation" video. You can assume that the system is a single core processor with a single node.

15

15

Submit You have used 2 of 5 attempts Show Answer

Large Datasets

1/1 point (graded)

Which of the following are true when working with a very large dataset?

- [x] Storing data in a columnar format such as Parquet can save disk space
- [ ] Data files are usually stored locally and not on HDFS
- [x] We can use Spark SQL on dataframes instead of writing long chains of complex map-reduce like operations on the underlying RDD.
- [ ] Using Spark SQL results in increased runtime than writing long chains of complex map-reduce like operations on the underlying RDD.
- [ ] CSV is the best format for storing and accessing large datasets
- [x] Parquet enables reading just the required data into memory using SQL like query syntax

Submit You have used 2 of 5 attempts Show Answer

Set operations

1/1 point (graded)

```
a = sc.parallelize([1, 4, 6, 7])
b = sc.parallelize([2, 4, 5, 7])
c = sc.parallelize([3, 5, 6, 7])
```

Which of the following series of operations results in an RDD **d** such that `d.collect() = [6]` ?

- [ ] `d = a.subtract(b)`
- [x] `d = a.intersection(c).subtract(b)`
- [ ] `d = a.subtract(c).intersection(b)`
- [ ] `d = a.intersection(b).subtract(c)`
- [x] `d = a.subtract(b).intersection(c)`
- [ ] `d = a.union(c).distinct().subtract(b)`
- [ ] `d = a.union(b).distinct().subtract(c)`
- [x] `d = a.intersection(c).subtract(a.intersection(b).intersection(c))`

Submit You have used 2 of 5 attempts Show Answer

© All Rights Reserved

edX
About
edX for Business
Legal
Terms of Service & Honor Code
Privacy Policy
Accessibility Policy
Connect
Blog
Contact Us
Help Center
Download on the App Store GET IT ON Google play
© 2020 edX Inc. All rights reserved.
| 深圳市恒宇博科技有限公司 粤ICP备17044299号-2