Exam Solutions

Ahmed Alhasan ‘ahmal787’

2020-06-02

Q1

- Wrong, everything that can be achieved by horizontal scalability can be achieved by vertical scalability by just adding more resources to it like CPUs, memory, …etc. exept for fault tolerance which does not directly affect read or even write scalability

Q2

name: UsMis
budget: 1,000,000
final report:{
>id: 391
>pages: 70
>location: http://acme.com/beerep
}

Q3

- in Key-Value database values are opaque to the system which which means no support for value-related queries

while in document database, queries can go through values/fields

- In Key-value database querying can only be done through CRUD operations in terms of keys (CRUD: create, read, update, delete)

while in document database queries expressed in terms of program code using an API or in a system-specific query language which can be done in terms of conditions on document content.

Q4

- Wrong. Because in Master-Slave replication every node could be a master for one replica and a slave for another replica.

Q5

- For strong consistency we need to have W+R > N, in this case N=4 and R=1, therefore we need W=4 to ensure strong consistency

Q6

- Because for smaller blocks we use a lot of I/O tasks that have high start-up time (where time = sender overhead + latency + reciever overhead + n/brandwidth) for n bytes

Q7

-a) - Record Reader: Parses an input file block from stdin into key value pairs that define input data records - Output Formatter: Translates the final (key,value) pair from the reduce function and writes it to stdout to a file in HDFS - Shuffle and sort: Downloads the needed files written by the partitioners to the node on which the reducer is running

-b) - Partitioner: Splits the intermediate elements from the mapper/combiner into shards (64MB blocks stored in local files) - Reducer: Run a user defined reduce function once per key grouping (Can aggregate, filter, and combine data)

Q8

- Transformations: are elementwise operations, fully parallelizable Mostly variants of Map and reading from distributed file, it creates RDD from each other but are not excuted
  - exmaple: map, applies a user defined function to transform previous RDD to a new one
- Actions: are operations with internally global dependence structure, mostly variants of Reduce and writing back to non-distr. file / to master, when Spark program reach to an Action it will excute the RDDs that lead to it and it does not create an RDD after excution
  - exmaple: reduce, Combine RDD elements using an associative binary function to produce a (scalar) result at the driver program

Q9

- Spark will provide a perforemance boost for applications that have the following characteristics:
  - iterative computations like some of the Machine Learning algorithms, i.e. K-Means, EM ..etc.
  - No replication of data blocks for fault tolerance in case of task failure worker failure), recompute it from available , earlier computed data blocks according to the data flow graph
  - Streaming: scalable, high throughput, fault tolerant stream processing of live data streams.
  - Anything that require speed or affected by I/O operations to disk by offering data-locality and keeping data in memory

Q10