DHARMSINH DESAI UNIVERSITY, NADIAD
FACULTY OF TECHNOLOGY
ONLINE SESSIONAL EXAMINATION

B.Tech (CE) sem→7
subject : Big Data Analytics

Roll no. → 142
signature → Rocket

Date : 9/10/2020
Time : 10:00 to 11:15
Total pages. : 12

Q.1

(a) reasons of Apache spark becoming more popular

→ Far more faster than other tools for large scale data processing
→ can be used with scala, Java, R & python.
→ provides powerful caching & disk persistance with simple programming layer.
→ deployment possible with Mesos, Hadoop via Yarn or spark's own cluster
→ Machine learning
→ Advance analytics
→ lazy evaluation
→ supports various data formats

(b) replication strategies of Cassandra.

2 replication strategy

(1) simple strategy:
→ move around ring to meet replication factor
→ stop when replication factor is met

(2) Network Topology Strategy:
→ somewhat complicated
→ different replication factor allowed for different data centers
→ Inside a data center, replicas on different racks are stored.

(c) Spark Dynamic Memory Utilization
→ We have to set it true to use it.
→ with this, once work is finished, it kills idle executors automatically.
→ If we set it false, i.e. dont use it, these idle executors are kept as it is.
→ Executors can be killed automatically after completion of jobs but drives has to be stopped. even after job completion, drives will still be there & eat resources

(d) ways related to improved performance via storage techniques of Hive.
→ By partitioning →bucketing.
→ use of indexing
→ vectorisation

Partitioning → without it hive reads all data in directory & applies query on it which is slow & expensive.

Indexing → separate index table is created which acts as reference for original table.

(e) SerDe, and its working.

→ it is short form of Serializer/Deserializer.
→ it is used in hive for IO
→ It allows Hive to read data from a table & write it back to HDFS in any custom format.
→ It handles both serialization & deserialization and also interprets results of serialization as individual fields of processing.

(f) advanced data types available with Hive & Pig.

along with primitive data typs, hive has

(4)

arrays : ARRAY < data_type >
maps : MAP < primitive_dtype , d_type >
structs : STRUCT < colnm : d_type [COMMENT col_comment], ... >

in string it has VARCHAR & CHAR & STRING
in Date/Time it has TIMESTAMP & DATE
in Misc types it has BOOLEAN & BINARY

→ in pig along with primitive types, it has

chararray
bytearray
tuple
bag
map

Q.2

(a) Maekawa's quorum based approach

→ for ensuring mutual exclusion in distributed systems.

→ Here a site does not request permission from every other site but from a subset of sites which is called quorum

3 message types

1. request
2. reply
3. release

algo used for following

1. entering critical section
2. execute " "
3. release " "

(c) (I) File formats supported by Hive with their purpos.

→ TEXTFILE → famous input/output format used in Hadoop
→ if table written in this format, JSON & CSV data can be loaded
→ In this format, each line is considered as a record.

packages → org.apache.hadoop.mapred.TextInputFormat
org.apache.hadoop.mapred.TextOutputFormat

→ SEQUENCEFILE
→ solves issue of working with file of size smaller than typical block size.

→ ~~The~~ SEQUENCE files are flat files consisting of binary key-value pairs
→ they are in binary format which can be split or used for clubbing 2 or more small files to one file.

package → org.apache.hadoop.mapred.SequenceFileInputFormat
org.apache.hadoop.mapred.SequenceFileOutputFormat

→ RCFILE → stands for Record columnar file
→ binary file format.
→ for high compression rate on the top of the rows
→ used when we want to operate multiple rows at a time

package → org.apache.hadoop.hive.ql.io.RCFile-InputFormat
org.apache.hadoop.hive.ql.io.RCFileOutputFormat

→ ORCFILE → optimized row columnar
→ stores data in optimized way than other formats
→ reduces size of data upto 75%.
→ increases speed of processing.
→ stores row data in groups (Stripes) along with file footer

package → org.apache.hadoop.hive.ql.io.orc

(II) Hive Crud operations

* create table.

e.g. creating orc table.

```
CREATE TABLE STUDENT
(
    STD_ID INT,
    STD_NAME STRING,
    ADDRESS STRING )
CLUSTERED BY (ADDRESS) into 3 Buckets
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ","
STORED as orc tblproperties ('transac-
                              tional'='true');
```

* insert

```
INSERT INTO TABLE STUDENT VALUES (101,
    'ABC', 'NADIAD');
```

* select

```
SELECT * FROM STUDENT;
```

* update

```
update STUDENT
SET std_id = 110
WHERE std_id = 101;
```

* delete from table

```
DELETE FROM STUDENT WHERE Std_id=110;
```

* delete all rows

```
DELETE FROM STUDENT;
```

Q.3

(a)(I) mapreduce vs Spark in terms of processing.

→ Spark is able to execute batch processing jobs b/c 10 to 100 times faster than the MapReduce Although both the tools are used for processing Big Data

→ Mapreduce is used when we want linear processing of large dataset & no intermediate solutions are required.

→ Spark is used when fast & interactive data processing is needed.

→ Spark provides Graph processing, Real-time processing, machine learning & Iterative jobs.

→ mapreduce & spark can be used

together in same Hadoop cluster or spark alone can be used as a processing framework

→ Spark was built on top of Hadoop Mapreduce & it extends Mapreduce to efficiently use more types of computations.

(II) Spark can work in following ways with Hadoop.

* Spark + HDFS

Spark can run on top of Hadoop's distributed file system (HDFS) to leverage the distributed replicated storage.

* Spark + mapreduce

Spark can be used along with MapReduce in same Hadoop cluster or we can use it alone as a processing framework

* Spark + Hadoop Yarn

Spark applications can also be run on YARN (Hadoop NextGen)

* Spark is not intended to replace Hadoop but it can be considered as its extension

(4)

→ MapReduce & Spark are used together where MapReduce is used for batch processing & Spark for real-time processing.

(b) different types of collections in Cassandra

Types of collections :-

1. set collection (unordered collection of unique values)
2. list collection (ordered collection of values)
3. map collection (used to map one thing to other like key-value pair)

examples

* set collection → phone nos of employee.
  → when queried returns values in sorted order.

  Alter Table employee_by_id ADD phone set<text>; // add column

  Update employee_by_id set phone = phone + {'sss'} where id=1;
  // add phone no. to column

* list collection

  → retrive in order in which they were entered.

Alter Table employee_by_id ADD language list<text>; //add column

Update employee_by_id set language = language + ['Hindi', Gujrati, English'] where id=1;

* map collection

e.g. to store timestamp related info.

→ each element stored as seperate column & operated individually

Alter Table employee_by_id ADD todo map<timestamp, text>;

Update employee_by_id set todo = {'2020-10-10' : 'Assignment', '2020-10-12' : 'work'} where id=1;

→ end of answer sheet