USN | | | | | | | | | | |

Internal Assessment Test 4 – Feb 2022 scheme and solution

Sub:	Big Data Analytics					Sub Code:	18CS72	Branch:	CSE
Date:	2/2/22	Duration:	90 mins	Max Marks:	50	Sem / Sec:	VII/A,B,C		OBE

		MARKS	CO	RBT
1	What are the functions in SQL? List the differences between SQL data store and NoSQL data store. Sql functions: 4 Difference : 6	[10]	CO1	L2
2	Explain any 3 distribution models. Each distribution model(3+3+4)	[10]	CO2	L2
3	Write short notes on Zookeeper and Ambari. Zookeeper-5 Ambari-5	[10]	CO2	L2
4	Explain the scoop import and export and Pig features. Scoop import and export with diagram: 6 Pig features- 4	[10]	CO2	L2
5	Explain NOSQL datastore architectures. 1) Document Store - 5 2) BigTable- 5	[10]	CO3	L2

Q1 Offers schema flexibility, simple relationships, dynamic schemas, auto sharding, replication, integrated caching, horizontal scalability of shards, distributable tuples, semi-structures data and flexibility in approach.

NoSQL data stores are considered as semi-structured data.

- NoSQL data store characteristics are as follows:

1. NoSQL is a class of non-relational data storage system with flexible data model.

Examples of NoSQL data-architecture patterns of datasets are key-value pairs, name/value pairs,

Column family Big-data store,

Tabular data store,

Cassandra (used in Facebook/Apache),

HBase, hash table [Dynamo (Amazon S3)],

unordered keys using JSON (CouchDB),

JSON (PNUTS), JSON (MongoDB),

Graph Store, Object Store,

ordered keys and semi-structured data storage systems.

2. NoSQL not necessarily has a fixed schema, such as table;

Do not use the concept of Joins (in distributed data storage systems);

Data written at one node can be replicated to multiple nodes.

Data store is thus fault tolerant.

The store can be partitioned into unshared shards.

Features in NoSQL Transactions NoSQL transactions have following features:

- Relax one or more of the ACID properties.
- Characterize by two out of three properties (consistency, availability and partitions) of CAP theorem, two are at least present for the application/service/process.
- Can be characterized by BASE properties (Section 3.2.3).
- Big Data NoSQL solutions use standalone-server, master-slave and peer-to peer distribution models.

Q2

- Simplest distribution option for NoSQL data store and access is Single Server Distribution (SSD) of an application.
- A graph database processes the relationships between nodes at a server. The SSD model suits well for graph DBs.
- Aggregates of datasets may be key-value, column-family or BigTable data stores which require sequential processing.
- These data stores also use the SSD model. An application executes the data sequentially on a single server.
- Figure 3.9(a) shows the SSD model. Process and datasets distribute to a single server which runs the application.

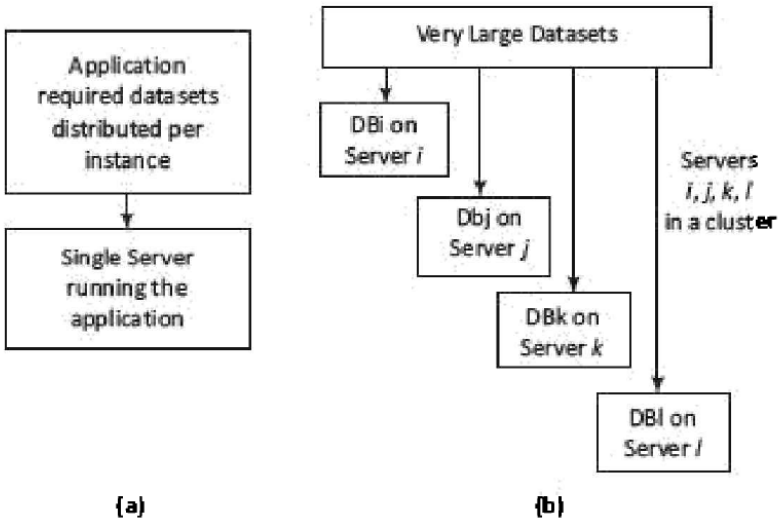


Figure 3.9 (a) Single server model (b) Shards distributed on four servers in a cluster.

SN

- **T**he application programming model in SN(shared nothing) architecture is such that an application process runs on multiple shards in parallel.
- Sharding provides horizontal scalability.
- A data store may add an auto-sharding feature.
- The performance improves in the SN.
- However, in case of a link failure with the application, the application can migrate the shard DB to another node.
- Refer fig 3.9 b

Master-Slave Distribution Model

- A node serves as a master or primary node and the other nodes are slave nodes.
- Master directs the slaves.
- Slave nodes data replicate on multiple slave servers in Master Slave Distribution (MSD) model. When a process updates the master, it updates the slaves also.
- A process uses the slaves for read operations.
- Processing performance improves when process runs large datasets distributed onto the slave nodes.
- Figure 3.10 shows an example of MongoDB. MongoDB database server is *mongod* and the client is *mongo.*

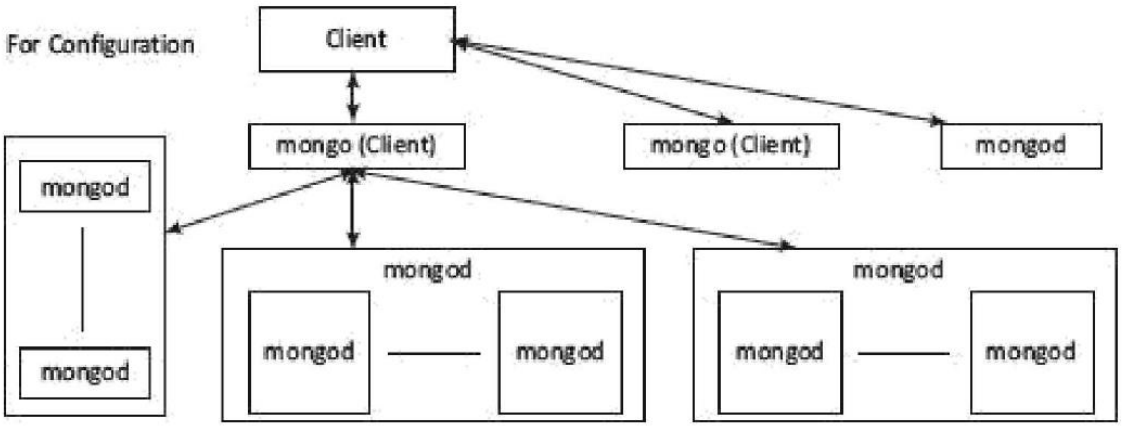


Figure 3.10 Master-slave distribution model. Mongo is a client and mongod is the server

- **Master-Slave Replication** Processing performance decreases due to replication in MSD distribution model.
- Resilience for read operations is high, which means if in case data is not available from a slave node, then it becomes available from the replicated nodes.
- Master uses the distinct write and read paths.
- **Complexity** Cluster-based processing has greater complexity than the other architectures.
- Consistency can also be affected in case of problem of significant time taken for updating

Peer-to-Peer Distribution Model

- Peer-to-Peer distribution (PPD) model and replication show the following characteristics:

1. All replication nodes accept read request and send the responses.
2. All replicas function equally.
3. Node failures do not cause loss of write capability, as other replicated node responds.

- Cassandra adopts the PPD model. The data distributes among all the nodes in a cluster.
- Performance can further be enhanced by adding the nodes. Since nodes read and write both, a replicated node also has updated data. Therefore, the biggest advantage in the model is consistency. When a write is on different nodes, then write inconsistency occurs.

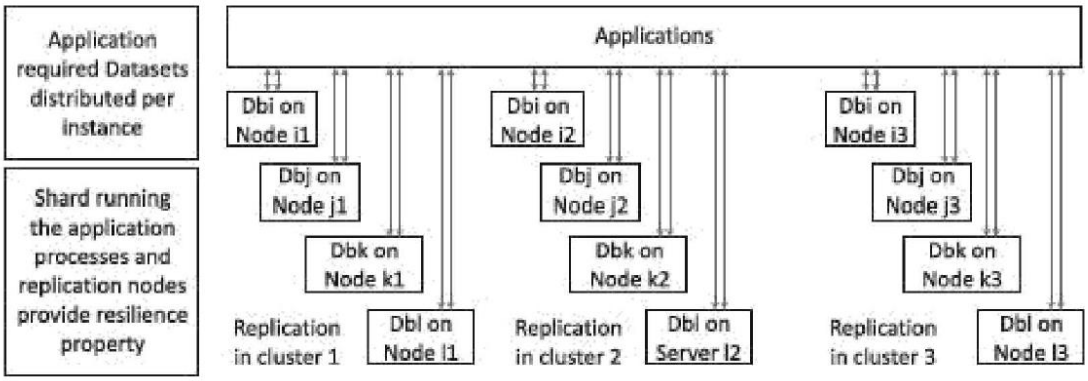


Figure 3.11 Shards replicating on the nodes, which does read and write operations both

Q3. **Zookeeper:**

- Apache Zookeeper is a coordination service that enables synchronization across a cluster in distributed applications
- Since multiple machines are involved, the race condition and deadlock are common problems when running a distributed application.
- Zookeeper in Hadoop behaves as a centralized repository where distributed applications can write data at a node called JournalNode and read the data out of it.
- Zookeeper uses synchronization, serialization and coordination activities. It enables functioning of a distributed system as a single function.
- ZooKeeper's main coordination services are:

1. Name service
2. Concurrency control
3. Configuration management - A requirement of a distributed system is a central configuration manager. A new joining node can pick up the up-to-date centralized configuration from the ZooKeeper coordination service as soon as the node joins the system.
4. Failure - Distributed systems are susceptible to the problem of node failures. This requires implementing an automatic recovering strategy by selecting some alternate node for processing (Using two MasterNodes with a NameNode each).

ii) Oozie

- Apache Oozie is an open-source project of Apache that schedules Hadoop jobs.
- The two basic Oozie functions are:
- Oozie workflow jobs are represented as Directed Acylic Graphs (DAGs), specifying a sequence of actions to execute.

- Oozie coordinator jobs are recurrent Oozie workflow jobs that are triggered by time and data availability.

Oozie provisions for the following:

- Integrates multiple jobs in a sequential manner
- Stores and supports Hadoop jobs for MapReduce, Hive, Pig, and Sqoop
- Runs workflow jobs based on time and data triggers
- Manages batch coordinator for the applications
- Manages the timely execution of tens of elementary jobs lying in thousands of workflows in a Hadoop cluster.

iii) Ambari

- Apache Ambari is a management platform for Hadoop. It is open source. Ambari enables an enterprise to plan, securely install, manage and maintain the clusters in the Hadoop. Ambari provisions for advanced cluster security capabilities, such as Kerberos Ambari.
- Features of Ambari and associated components are as follows:

1. Simplification of installation, configuration and management
2. Enables easy, efficient, repeatable and automated creation of clusters
3. Manages and monitors scalable clustering
4. Provides an intuitive Web User Interface and REST API The provision enables automation of cluster operations.
5. Visualizes the health of clusters and critical metrics for their operations
6. Enables detection of faulty node links
7. Provides extensibility and customizability.

- Provides Secuirity
- Ambari helps automation of the setup and configuration of Hadoop using Web User Interface and REST APis.
- Single harmonized view on console makes administering the task easier. Visualization can be up to individual components level on drilling down. Nodes addition and deletion are easy using the console.

Q4

- Pig : Apache Pig is an open source, high-level language platform.
- Pig was developed for analyzing large-data sets.
- Pig executes queries on large datasets that are stored in HDFS using Apache Hadoop.

- The language used in Pig is known as Pig Latin.

Additional features of Pig are as follows:

- Loads the data after applying the required filters and dumps the data in the desired format.
- Requires Java runtime environment for executing Pig Latin programs.
- Converts all the operations into map and reduce tasks. The tasks run on Hadoop.
- Allows concentrating upon the complete operation, irrespective of the individual Mapper and Reducer functions to produce the output results.

Scoop import and export:

Figure describes the Sqoop data import (to HDFS) process.

The data import is done in two steps.

In the first step, shown in the figure, Sqoop examines the database to gather the necessary metadata for the data to be imported.

The second step is a map-only (no reduce step) Hadoop job that Sqoop submits to the cluster.

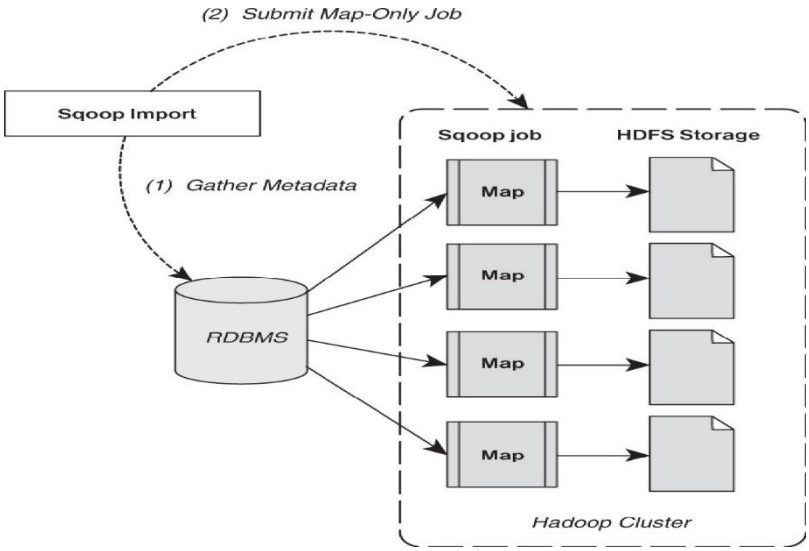
This job does the actual data transfer using the metadata captured in the previous step. Note that each node doing the import must have access to the database.

The imported data are saved in an HDFS directory.

Sqoop will use the database name for the directory, or the user can specify any alternative directory where the files should be populated

Default format is csv, but can be changed

Once placed in HDFS, the data are ready for processing.



Scoop Export

- Data export from the cluster works in a similar fashion.
- The export is done in two steps, as shown in Figure
- As in the import process, the first step is to examine the database for metadata.
- The export step again uses a map-only Hadoop job to write the data to the database.
- Sqoop divides the input data set into splits, then uses individual map tasks to push the splits to the database.
- Again, this process assumes the map tasks have access to the database.

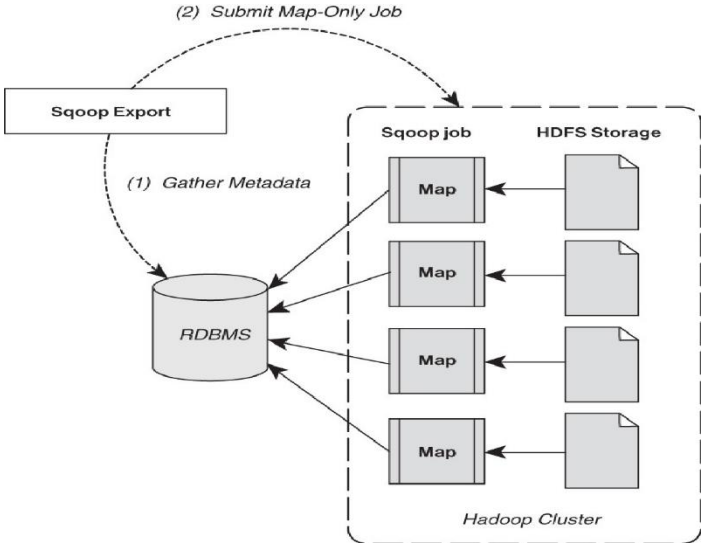


Figure 7.2 Two-step Sqoop data export method (Adapted from Apache Sqoop Documentation)

Q5 Explain NOSQL datastore architectures. 1) Document Store - 5
2) BigTable- 5

Document Store

Characteristics of Document Data Store are high performance and flexibility. Scalability varies, depends on stored contents. Complexity is low compared to tabular, object and graph data stores.

Following are the features in Document Store:

1. Document stores unstructured data.

2. Storage has similarity with object store.

3. Data stores in nested hierarchies. For example, in JSON formats data model XML document object model (DOM), or machine-readable data as one BLOB. Hierarchical information stores in a single unit called document tree. Logical data stored together in a unit.

4. Querying is easy. For example, using section number, sub-section number and figure caption and table headings to retrieve document partitions.

5. No object relational mapping enables easy search by following paths from the root of document tree.

6. Transactions on the document store exhibit ACID properties.

Typical uses of a document store are: (i) office documents, (ii) inventory store,

(iii) forms data, (iv) document exchange and (v) document search.

Examples:

- The database stores and retrieves documents, such as XML, JSON, BSON
- Some of the popular document data stores are CouchDB, MongoDB, Terrastore, OrientDB and RavenDB.
- NoSQL DBs enable ACID rule-based : MongoDB, Apache Couchbase and MarkLogic.

BigTable

Following are features of a BigTable:

1. Massively scalable NoSQL. BigTable scales up to 100s of petabytes.

2. Integrates easily with Hadoop and Hadoop compatible systems.

3. Compatibility with MapReduce, HBase APis which are open-source Big Data platforms.

4. Key for a field uses not only row_ID and Column_ID but also timestamp and attributes.

Values are ordered bytes. Therefore, multiple versions of values may be present in the BigTable.

5. Handles million of operations per second.

6. Handle large workloads with low latency and high throughput

7. Consistent low latency and high throughput

8. APis include security and permissions

9. BigTable, being Google's cloud service, has global availability and its service is seamless.