PAGE NO.:
DATE:

Big-Data Assignment

Drishi Dugar (2073/BCT/13)

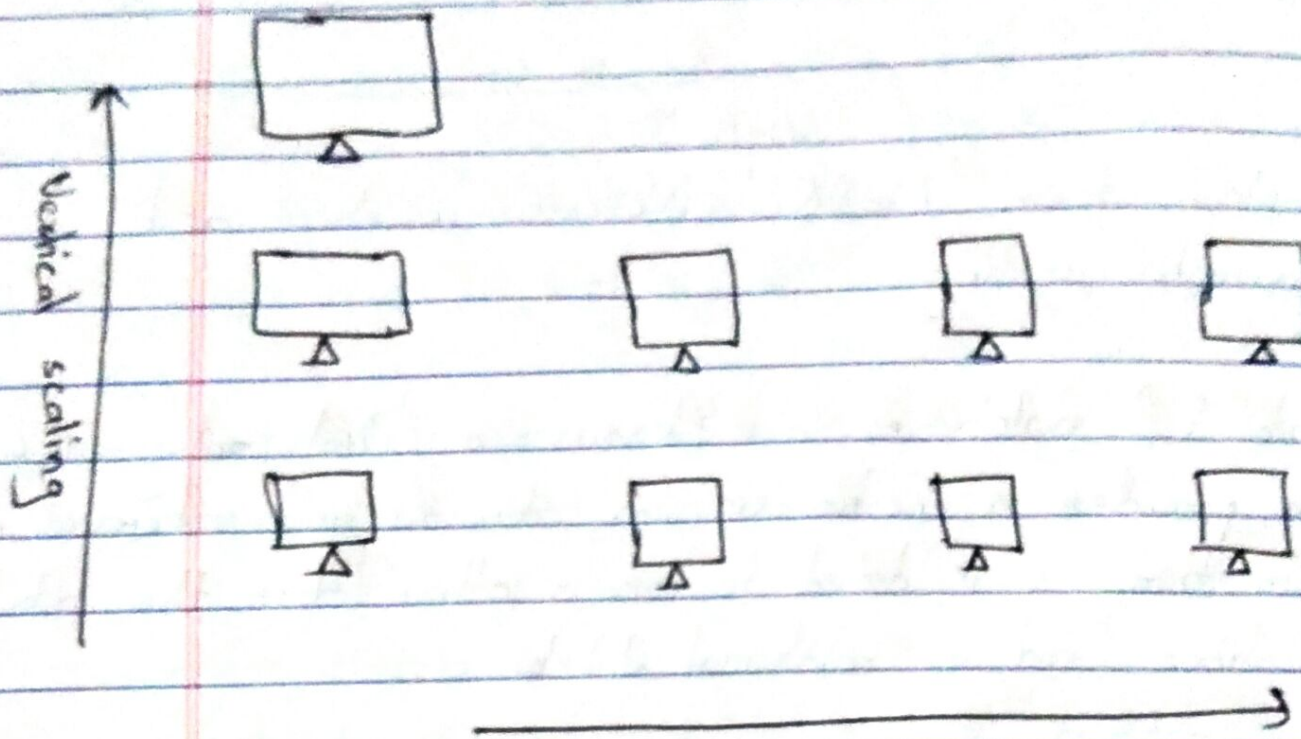
(1) Explain term 'NoSQL'. Describe vertical and horizontal scaling.

Ans ⇒ "NoSQL" databases also known as "Not only SQL" ~~are~~ provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.

⇒ Vertical scaling:
Vertical scaling means adding more CPU power and storage resource to a single instance. The major benefit is that all the resources are in a single machine so there is no need to manage multiple instance. The major drawback is the cost efficiency. For eg: MySQL- Amazon RDS.

⇒ Horizontal scaling:
It divides the data set and distributes the data over multiple servers. The major benefit is that the data is in smaller chunks so processing is fast. The drawback is about managing the instances and complex distributed

architecture. For eg: MongoDB and Cassandra.



(2) Why does normalization fail in data analytics scenario?

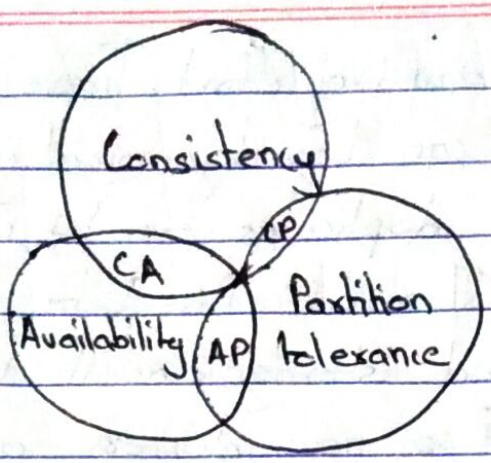
Ans ⇒ Data analytics is associated with big data and its 'three V's' i.e. volume, velocity and variety. NoSQL design is made keeping these in mind whereas RDBMS are strict and have to follow predefined schema. Schema are designed by normalization of various attributes of data. When new data sources and new questions arise, the schema and related applications have to be updated, which usually requires an

PAGE NO.:
DATE:

expensive, time-consuming effort. There is no schema in NoSQL. Attributes can be dynamically added. Normalization is done so that duplicates can be minimized as far as possible, but NoSQL and Big data do not care about ~~dubli~~ duplicates and storage as it is distributed over multiple clusters. so new clusters and storage can be easily added. But, in RDBMS, most are single node storage which leads to the failure of normalization in data analytics.

(3) Explain CAP theorem and its implications in Distributed database like NoSQL.

Ans⇒ CAP (Consistency, Availability, Partition tolerance) theorem states that a distributed system ~~host~~ has to make a tradeoff between Consistency and Availability when a Partition occurs. i.e. a distributed system can only have ~~of~~ two of the ~~three~~.



→ Consistency means that a data item has the same and latest value among different nodes.

→ Availability means that a non-failing node always returns response.

→ Partition-tolerance means that nodes continue to function in the presence of network partitions.

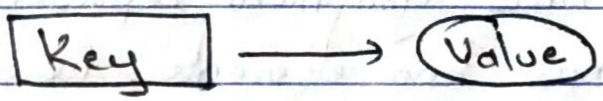
Since only two of three properties can be achieved simultaneously, a distributed system can be characterized as either an AP, CP or a CA system. If a partition occurs in a distributed system, it is impossible to provide consistent data on both nodes and availability of complete data. CA type databases are generally the monolithic databases that work on a single node and provide no distribution.

(4) List and explain the types of NoSQL database.

PAGE NO.:
DATE:

Ans⇒ The types of NoSQL database are:

(a) Key-value database:
It is the most simplest of NoSQL database and also the most scalable, allowing horizontal scaling of large amounts of data. These databases have a dictionary structure that consists of a set of objects that represent fields of data. Each object is assigned a unique key.



For eg: Redis, Riak etc.

(b) Document database:
This type of database consists of sets of key-value pairs stored into a document. It is one step up in complexity from key-value database. For eg: MongoDB, CouchDB etc.

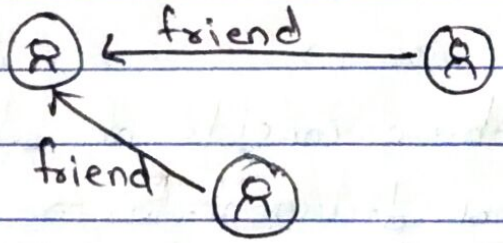
(c) Wide-column database:
In these, data is stored and grouped into seperately store columns instead of rows. Such database organize information into columns that function similarly to tables in relational database. For eg: Cassandra, HBase etc.

Name	ID
John	001
Karen	002
Bill	003

Grade	ID
A	001
B	002
C	003

Fig: Column oriented.

(d) Graph database:
It uses flexible graphical representation to manage data. It has two elements i.e. nodes (for storing data entities) and edges (for storing relationship between entities). For eg: OrientDB, Neo4j etc.



(5) Explain eventual consistency and tunable consistency in the context of Cassandra.

Ans) Eventual consistency in cassandra is achieved by trading off consistency for availability i.e. rather than to maintain strict consistency, eventual consistency

enables high availability at the cost of every instance of data not being synced up across all servers right away.

-) Cassandra extends the concept of eventual consistency by offering tunable consistency, for any given read or write operation, the client application decides how consistent the data request should be. This is achieved by read and write consistency levels. These levels specify the number of replicas required to respond.

---

(6) What is Lucene? Describe the typical components involved in search application.

Ans⇒ Lucene core is a Java library providing powerful indexing and search features, as well as spellchecking, hit highlighting and advanced analysis/tokenization capabilities.

⇒ The typical components involved in search applications are:

(a) Web crawler ⇒ It is also known as spider or bots. It is a software component that traverses the web to gather information.

(b) Database ⇒ All the information on the web is stored in database. It consists of huge web resources.

(c) Search interfaces ⇒ It is an interface between user and database. It helps the user to search through the database.

---

(7) What are primary analyzers in Lucene? Describe them.

Ans) Lucene analyzers split text into tokens so it mainly consists of tokenizers and filters. Some common analyzers are:

(a) Standard analyzer ⇒ It can recognize URLs and emails. It also removes stop words and lowercases generated tokens.

(b) Stop analyzer ⇒ It consists of Letter Tokenizer, Lowercase filter and stop filter. It splits text by non-letter characters and removes stop words from token list but doesn't recognize URLs.

(c) Simple analyzer ⇒ It consists of Letter Tokenizer and a lower case Filter. It doesn't remove stop words and also doesn't recognize URLs.

(d) Whitespace analyzer ⇒ It uses only a Whitespace Tokenizer which splits text by whitespace characters.

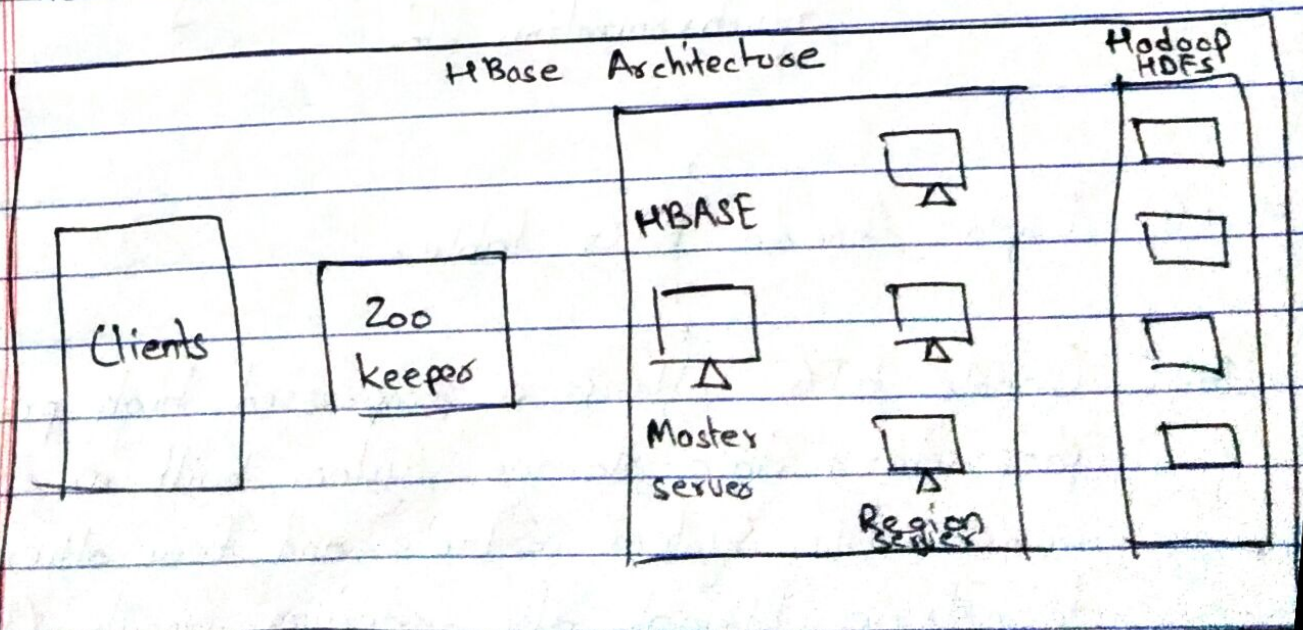
(e) Keyword analyzer ⇒ It tokenizes input into a single token. It is useful for fields like ids and zipcodes.

(f) Language analyzer ⇒ They are special analyzers for different ~~langs~~ languages.

---

(8) Describe HBase architecture in detail.

Ans ⇒ In HBase, tables are split into regions and are served by the region servers. Regions are vertically divided by column families into 'Stores' which are saved as files in HDFS.



HBase has the following components:

(a) Master server ⇒ It assigns region to the region servers and takes the help of zookeeper. It handles load balancing of regions across region servers. It maintains the state of the cluster by negotiating the load balancing.

(b) Regions ⇒ They are tables that are split up and spread across the region servers.

(c) Region server ⇒ It communicates with the client and handles data-related operations. It handles read and write requests for all regions under it. Decides the size of region.

(d) Zookeeper ⇒ It is an open-source project that provide services like maintaining configuration information, naming, providing distributed synchronization etc.

---

(9) Explain Google BIG table.

Ans⇒ Google BIG table is a compressed, high performance, proprietary data storage system built on Google File System, SSTable and few other Google technologies. Bigtable also underlies Google Cloud

PAGE NO. :
DATE :

Datastore, which is available as a part of the Google Cloud Platform. It is one of the prototypical examples of a wide-column store. Tables are split into multiple tablets.