CS 580K Advance Topics in Cloud Computing

Ketan Deshpande
B00816854
kdeshpa5@binghamton.edu

1. What does NoSQL mean? Why do we need such an approach to store and organize data? How does it address the problem that exists in the traditional relational databases?

NoSQL means Not Only SQL. It is a non-relational data storage system. It doesn't require a fixed table schema nor do they use the concept of joins.

Need: The new generation applications generate large amount of data every day. To store such large information, we need specific type of database management system. Also, with the new generation application data there is no relation between them. That is the data can be any text, images, audio, etc. To store such type of information we need NoSQL.

Problems it addresses: The traditional database management system's read only cache or other relational databases do not deliver throughput that the new generation's NoSQL database management system provides. There is a shift to dynamically typed data with frequent schema changes. To address dynamic schema changes, we need a database management system which can easily cope up with such changes, which NoSQL does.

2. What is the CAP theorem? Why in most distributed systems, P is needed? Then how do you choose between C and A?

CAP theorem means Consistency, Availability and Partition tolerance. Consistency means all the copies will have same value when fetched. Availability means Every request received by a non-failing node the system must result in a response. Partition tolerance is related to network partition meaning network can break into 2 or more independent parts. You can have at most 2 of the 3 properties at a time.

Most of the distributed systems need scalability which needs network partitioning. That is why P is the necessary for such systems. And therefore, you need to choose between C and A based on needs of the services. For e.g. for the online shopping cart process, you need availability for the products and services. The system should be available so that customers can choose and add items in the cart. On the other hand, while making payment for the cart items, you need consistency. Because to make a successful payment, to place an order, to deliver selected items you need a consistent system.

3. What is eventual consistency? Please use a concrete example to demonstrate the difference between strict consistency and eventual consistency.

Eventual consistency model means eventually all the updates will propagate through the system and all the nodes will be consistent. It is a just matter of time. In other words, when no updates occur for a long period of time, eventually all updates will propagate through the system and all nodes will be consistent.

The strict consistency means all read operations must return the data from the latest completed write operation, regardless of which replica the operations went to. It implies nodes employ some kind of distributed transaction protocol to ensure all data copies have the same value. Hence the strict consistency model is different from eventual consistency. For e.g. the most popular system that implement eventual consistency is DNS (Domain Name System). Updates to a name are distributed according to a configured pattern and with time-controlled caches. Eventually all clients will see new updated in the DNS.

4. What are the main data models for state-of-the-art NoSQL databases? Why there are different types of these data models for NoSQL databases?

There are various types of data models of NoSQL because of the need of the current new age applications. The main data models of NoSQL database are as follows:

i. Document based database – It can model more complex objects. It is collection of documents. Documents are basically JSON objects. JavaScript Object Notation (JSON) is a data model, which supports objects, records, structs, list, array, maps, dates, with nesting. It can contain nested documents hence more complex structure it can handle. Therefore, it is better than simple key-value model.

ii. Column-family – it uses a concept called keyspace. It contains all column families. Column family contains rows. Each row can contain different number of columns. This type of structure can store a large amount of data therefore it is better to use where there is a large volume of data need to be stored.

iii. Graph database – it is based on graph theory which uses node and edge. Each node represents an entity and each edge (relationship) represents how 2 entities are associated. Relationships can be intuitively visualized using graph databases making them useful for heavily interconnected data. It explicitly lays out the dependencies between nodes of data. But other models link them implicitly. Graph databases by design allow simple and fast retrieval of complex hierarchical structures that are difficult to model in relational systems.

5. Why do we need a cloud OS (e.g., openstack)? How many components are involved in launching a VM instance?

Cloud operating system keeps together the cloud computing resources that are server virtualization, network virtualization and storage virtualization. Cloud computing is a model which enables configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. This resource management is done by cloud operating system. The components involved in launching VM instance are vcpu, memory, storage, operating system (which image template to use to launch VM) and network configuration to access the VM.