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

Internal Assessment Test 1 – April 2019

Sub:	Managing Big Data					Sub Code:	18CS21	Branch:	CSE
Date:	15/04/19	Duration:	90 min's	Max Marks:	50	Sem / Sec:	M. Tech (CSE)/ II SEM		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1.	(a) Explain whether all the three below be guaranteed in NoSQL? Consistency, Availability, Partition tolerance. Describe Eric Brewers theorem accordingly.	[1+3]	CO2	L2
	(b) List different levels of tunable consistency.	[2]	CO2	L1
	(c) A user has 3 nodes to be replicated. Calculate the read and write quorum	[4]	CO2	L3
2.	(a) Explain sharding with the different strategies available?	[4]	CO2	L2
	(b) What is read resilience? Recommend a basic model which satisfies the same.	[1+1]	CO2	L3
	(c) Discuss about peer to peer replication and master slave replication models.	[4]	CO2	L2
3.	(a) Describe polyglot persistence in software development with respect to NoSQL	[5]	CO2	L2
	(b) Suggest a data base type / model for the following functionality and considerations. • Shopping cart / High availability • User activity logs / Lot of writes on multiple nodes • Fraud detection / links between orders and purchases • Financial data / fixed schema • Product catalog / Lot of reads and natural aggregates	[5]	CO2	L3
4.	(a) Describe different types of aggregate data models	[6]	CO2	L1
	(b) Discuss about Graph data model and where it finds the usage	[4]	CO2	L2
5.	(a) Describe why RDBMS can not be used in the Big Data for most of the use cases?	[5]	CO1	L2
	(b) List different issues with respect to impedance mismatch in RDBMS	[5]	CO1	L1
6	(a) Illustrate how Big Data can be used in health care.	[5]	CO1	L2
	(b) How Big Data solutions solve the problems on advertising and marketing?	[5]	CO1	L2
7.	(a) Company "A" could not solve an issue through internal resources and knowledge. How Big Data and Open Source Methods help them to solve it	[4]	CO1	L2
	(b) Distinguish intra, inter and trans firewall analytics.	[6]	CO1	L2

1. (a) Explain whether all the three below be guaranteed in NoSQL? Consistency, Availability, Partition tolerance. Describe Eric Brewers theorem accordingly. Marks [4]

At any time, only two of them will be guaranteed.

No distributed system is safe from network failures, thus network partitioning generally has to be tolerated. In the presence of a partition, one is then left with two options: consistency or availability. When choosing consistency over availability, the system will return an error or a time-out if particular information cannot be guaranteed to be up to date due to network partitioning. When choosing availability over consistency, the system will always process the query and try to return the most recent available version of the information, even if it cannot guarantee it is up to date due to network partitioning.

In the absence of network failure – that is, when the distributed system is running normally – both availability and consistency can be satisfied.

CAP is frequently misunderstood as if one has to choose to abandon one of the three guarantees at all times. In fact, the choice is really between consistency and availability only when a network partition or failure happens; at all other times, no trade-off has to be made.

Database systems designed with traditional ACID guarantees in mind such as RDBMS choose consistency over availability, whereas systems designed around NoSQL movement for example, choose availability over consistency

The CAP theorem aka Eric Brewer theorem is a tool used to makes system designers aware of the trade-offs while designing networked shared-data systems. CAP has influenced the design of many distributed data systems. It made designers aware of a wide range of trade offs to consider while designing distributed data systems.

The theorem states that **networked shared-data systems** can only guarantee/strongly support two of the following three properties:

- **Consistency**
- **Availability**
- **Partition Tolerant**

The C and A in ACID represent different concepts than C and in A in the CAP theorem.

The CAP theorem categorizes systems into three categories:

- CP (Consistent and Partition Tolerant) - At first glance, the CP category is confusing, i.e., a system that is consistent and partition tolerant but never available. CP is referring to a category of systems where availability is sacrificed only in the case of a network partition.
- CA (Consistent and Available) - CA systems are consistent and available systems in the absence of any network partition. Often a single node's DB servers are categorized as CA systems. Single node DB servers do not need to deal with partition tolerance and are thus considered CA systems. The only hole in this theory is that single node DB systems are not a network of shared data systems and thus do not fall under the preview of CAP.
- AP (Available and Partition Tolerant) - These are systems that are available and partition tolerant but cannot guarantee consistency.

1 (b) List different levels of tunable consistency. Marks [2]

It is the definition of an operation’s consistency level specifies how many of the replicas need to respond to the coordinator in order to consider the operation a success.

Different levels:

- ONE : at least one single replica must respond
- TWO : at least two replicas must respond
- THREE : at least three replicas must respond
- QUORUM : A majority of the replicas must respond
- ALL : All the replicas must respond
- LOCAL_QUORUM : Majority of the replicas in local data center must respond
- EACH QUORUM : Majority in each of the quorum should be satisfied.

1 (c) A user has 3 nodes to be replicated. Calculate the read and write quorum Marks [4]

As per write quorum, we need W > N / 2 [Where N is the replication factor and W is the right quorum] So, we need W > 3 / 2, then we need at least 2 nodes to satisfy write quorum.

As per read quorum, we need R + W > N [N is 3 as given and W is 2 as per write quorum above.

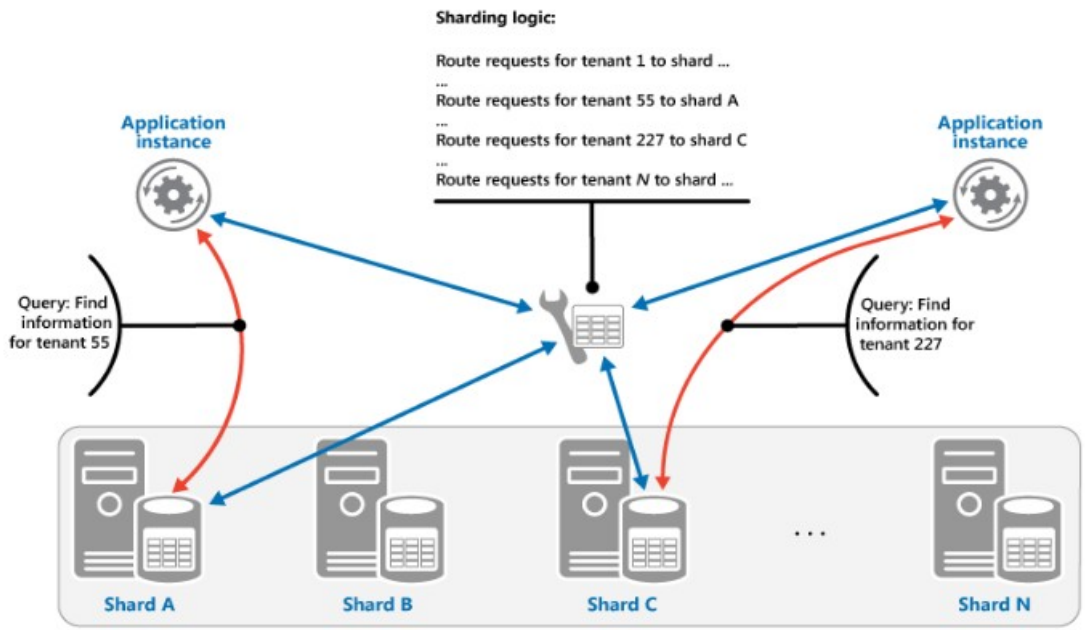
So, R + 2 > 3; So, we need 'R' as 2 at least.

Then the read and write consistencies as per the quorum is 2 and 2 respectively.

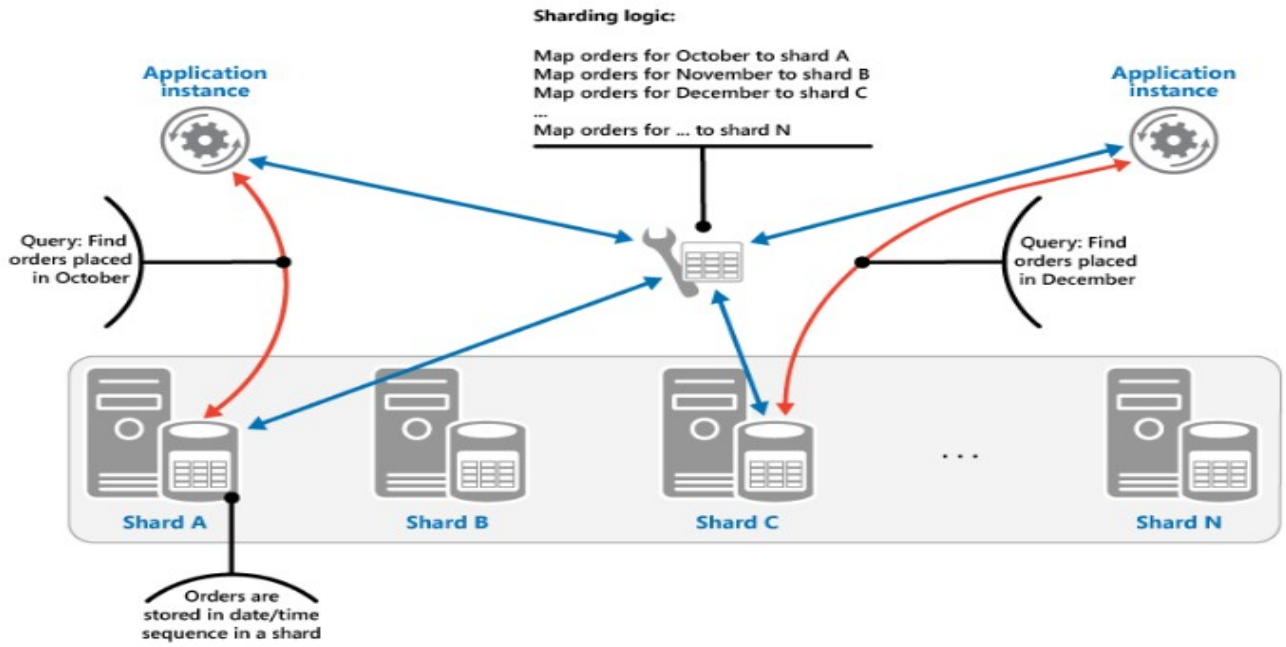
2 (a) Explain sharding with the different strategies available? Marks [4]

Separate very large databases into smaller, faster and more easily managed parts are called shards. It is a database design principle whereby rows of a database table are held separately, rather than being split into columns (which is what normalization and vertical partitioning do, to differing extents). Each partition forms part of a shard, which may in turn be located on a separate database server or physical location.

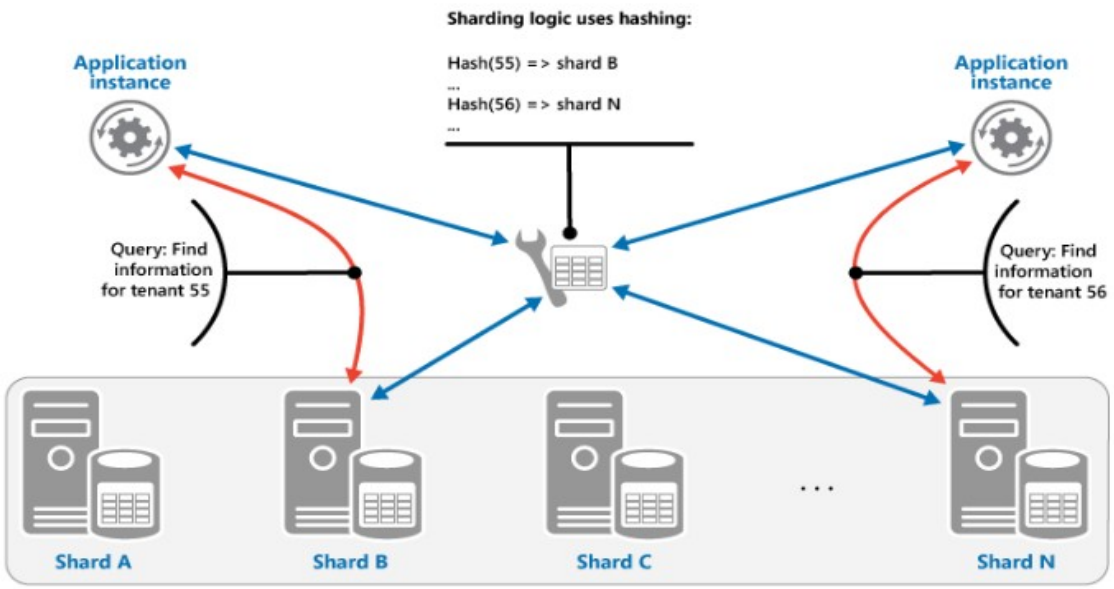
The Lookup strategy. In this strategy the sharding logic implements a map that routes a request for data to the shard that contains that data using the shard key. In a multi-tenant application all the data for a tenant might be stored together in a shard using the tenant ID as the shard key. Multiple tenants might share the same shard, but the data for a single tenant won't be spread across multiple shards.



The Range strategy. This strategy groups related items together in the same shard, and orders them by shard key—the shard keys are sequential. It's useful for applications that frequently retrieve sets of items using range queries (queries that return a set of data items for a shard key that falls within a given range). For example, if an application regularly needs to find all orders placed in a given month, this data can be retrieved more quickly if all orders for a month are stored in date and time order in the same shard. If each order was stored in a different shard, they'd have to be fetched individually by performing a large number of point queries (queries that return a single data item). The next figure illustrates storing sequential sets (ranges) of data in shard.



The Hash strategy. The purpose of this strategy is to reduce the chance of hotspots (shards that receive a disproportionate amount of load). It distributes the data across the shards in a way that achieves a balance between the size of each shard and the average load that each shard will encounter. The sharding logic computes the shard to store an item in based on a hash of one or more attributes of the data. The chosen hashing function should distribute data evenly across the shards, possibly by introducing some random element into the computation. The next figure illustrates



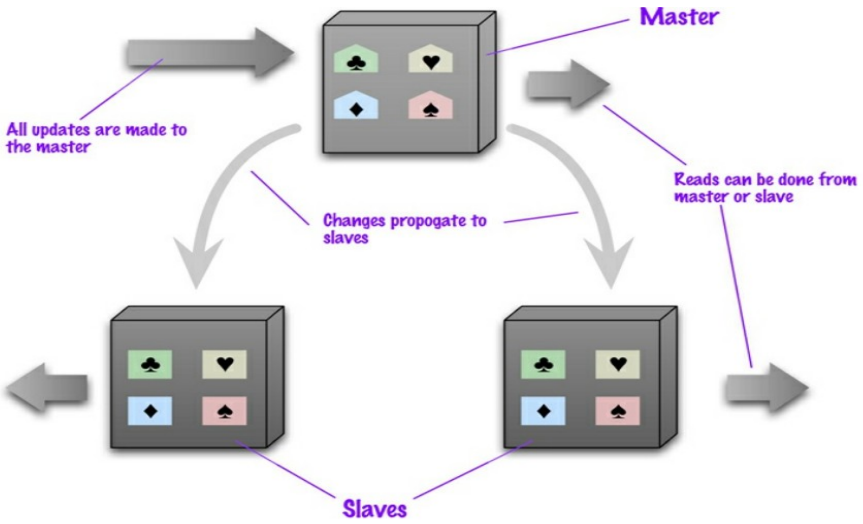
2 b) What is read resilience? Recommend a basic model which satisfies the same. Marks [2]

Read resilience is when the master node fails, the data can be safely read from the slaves. Master and Slave replication model satisfies read resilience. Even if the master fails, one of the slaves can be automatically elected as new master and the write / updates can continue.

2 c) Discuss about peer to peer replication and master slave replication models. Marks [4]

Master Slave Replication:

- With master-slave distribution, you replicate data across multiple nodes.
- One node is designated as the master, or primary. This master is the authoritative source for the data and is usually responsible for processing any updates to that data.
- The other nodes are slaves, or secondaries.
- A replication process synchronizes the slaves with the master.



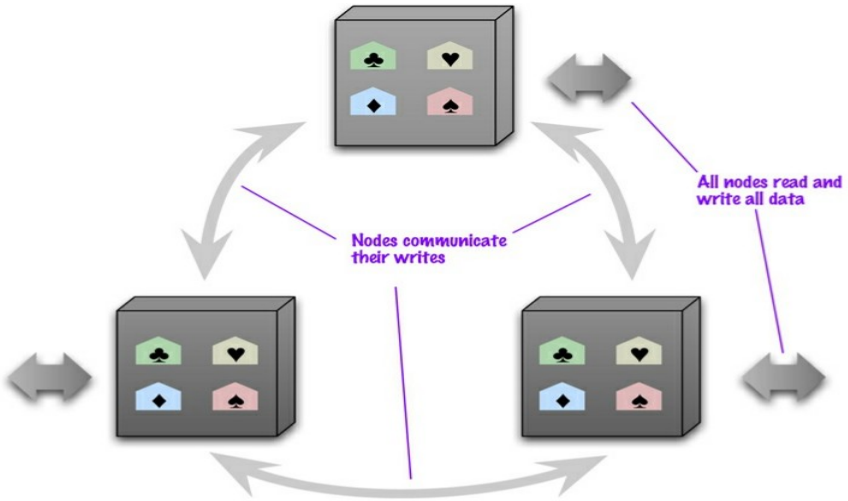
It is very useful when you have lot of reads. You can add as much of slave nodes and read the data faster. When the master node fails, the data can be safely read from the slaves. Even if the master fails, one of the slaves can be automatically elected as new master and the write / updates can continue. All the read / write can be made as master only and the slaves can just act as a hot backup – The replica of single server model.

Drawback:

- Inconsistency: Danger of different clients reading different slaves where there is a chance that the master / slave sync has not happened yet.
- In efficient for frequent writes as there is a dependency on master node.

Peer to Peer Replication:

There is no master in the cluster. All the replicas have equal weight, they can all accept writes, and the loss of any of them doesn't prevent access to the data store.



Drawback:

Inconsistency: Danger of different clients reading different slaves where there is a chance that the master / slave sync has not happened yet.

3 (a) Describe polyglot persistence in software development with respect to NoSQL
Marks [5]

Polyglot Persistence is a fancy term to mean that when storing data, it is best to use multiple data storage technologies, chosen based upon the way data is being used by individual applications or components of a single application. Different kinds of data are best dealt with different data stores. In short, it means picking the right tool for the right use case. It's the same idea behind Polyglot Programming, which is the idea that applications should be written in a mix of languages to take advantage of the fact that different languages are suitable for tackling different problems.

An example Use case of Twitter where polyglot persistence is used.

Any decent sized enterprise will have a variety of different data storage technologies for different kinds of data. There will still be large amounts of it managed in relational stores, but increasingly we'll be first asking how we want to manipulate the data and only then figuring out what technology is the best bet for it.

This polyglot affect will be apparent even within a single application. A complex enterprise application uses different kinds of data, and already usually integrates information from different sources.

Increasingly we'll see such applications manage their own data using different technologies depending on how the data is used. This trend will be complementary to the trend of breaking up application code into separate components integrating through web services. A component boundary is a good way to wrap a particular storage technology chosen for the way its data in manipulated.

(b) Suggest a data base type / model for the following functionality and considerations. **Marks [5]**

- Shopping cart / High availability — Document or Key Value
- User activity logs / Lot of writes on multiple nodes — Document or Key Value
- Fraud detection / links between orders and purchases — Graph
- Financial data / fixed schema — RDBMS
- Product catalog / Lot of reads and natural aggregates — Document

4 (a) Describe different types of aggregate data models **Marks [6]**

Key Value Data Model:

- Type of non relational database that uses a simple key-value method to store data.
- Key serves as unique identifier.
- Both keys and values can be anything, ranging from simple objects to complex compound objects.
- Key-value databases are highly partitionable and allow horizontal scaling at scales that other types of databases cannot achieve.
- Amazon DynamoDB allocates additional partitions to a table if an existing partition fills to capacity and more storage space is required.

Examples:
Amazon DynamoDB
Apache Cassandra
Riak

Key	Value
123456789	APPL, Buy, 100, 84.47
234567890	CERN, Sell, 50, 52.78
345678901	JAZZ, Buy, 235, 145.06
456789012	AVGO, Buy, 300, 124.50

Document Data Model

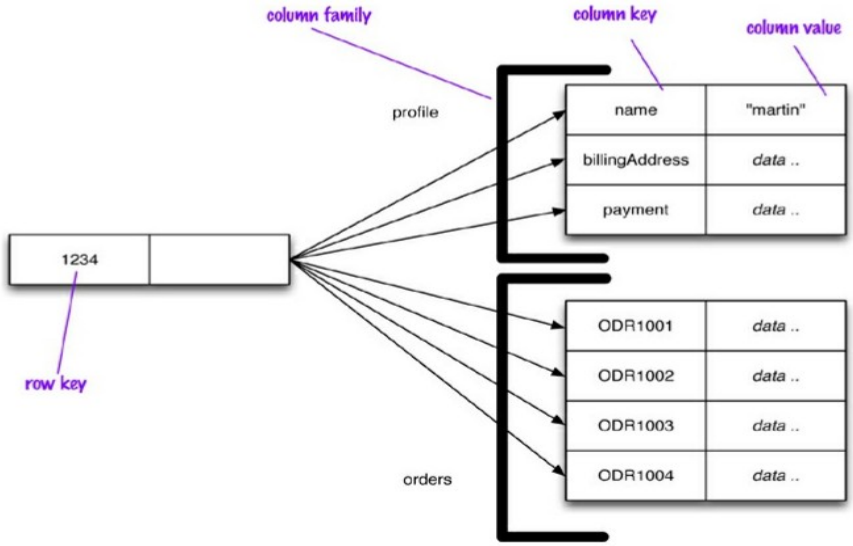
- Documents in a document store are roughly equivalent to the programming concept of an object.
- They are not required to adhere to a standard schema, nor will they have all the same sections, slots, parts or keys.
- Generally, programs using objects have many different types of objects, and those objects often have many optional fields.
- Every object, even those of the same class, can look very different.
- Document stores are similar in that they allow different types of documents in a single store, allow the fields within them to be optional, and often allow them to be encoded using different encoding systems.
- Example :JSON, BSON, XML, YAML
  - Amazon DocumentDB
  - MongoDB
  - Couchbase

- Unlike a relational database where every record contains the same fields, leaving unused fields empty; there are no empty 'fields' in either document (record)

```
<contact>
  <firstname>Bob</firstname>
  <lastname>Smith</lastname>
  <phone type="Cell">(123) 555-0178</phone>
  <phone type="Work">(890) 555-0133</phone>
  <address>
    <type>Home</type>
    <street1>123 Back St.</street1>
    <city>Boys</city>
    <state>AR</state>
    <zip>32225</zip>
    <country>US</country>
  </address>
</contact>
```

Column Family Data Model

- Column family as a way to store and organize data
- Table as a two-dimensional view of a multi-dimensional column family
- Operations on tables using the Cassandra Query Language (CQL)
- Skinny row: has a fixed, relatively small number of column keys
- Wide row: has a relatively large number of column keys (hundreds or thousands); this number may increase as new data values are inserted



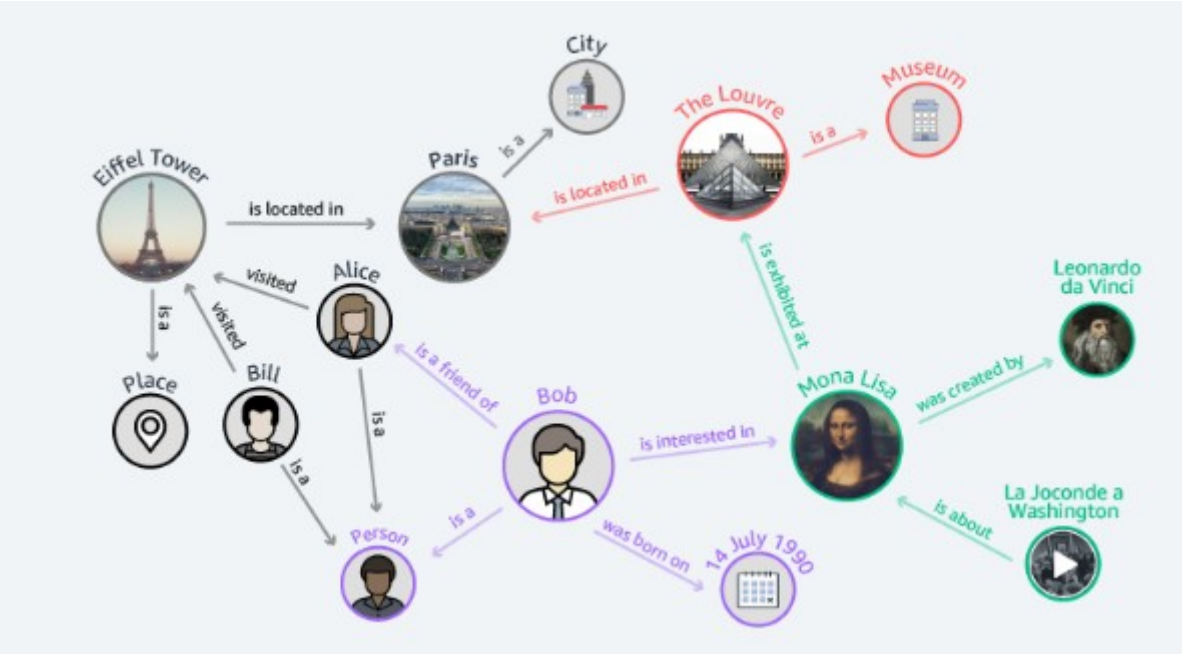
Example: Google's BigTable

4 (b) Discuss about Graph data model and where it finds the usage Marks [4]

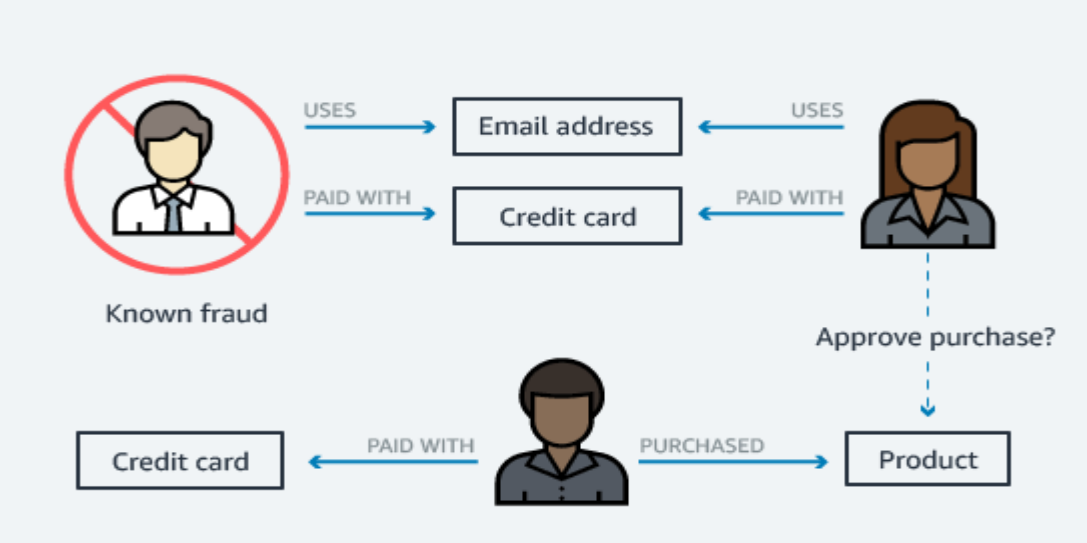
- Graph databases portrays the data as it is viewed conceptually.
- This is accomplished by transferring the data into nodes and its relationships into edges.
- Nodes represent entities or instances such as people, businesses, accounts, or any other item to be tracked.
- They are roughly the equivalent of the record, relation or row in a relational database, or the document in a document-store database.
- Edges, also termed graphs or relationships, are the lines that connect nodes to other nodes;

- edges can either be directed or undirected.
- In an undirected graph, an edge from a point to another has one meaning.
- In a directed graph, the edges connecting two different points have different meanings depending on their direction

Example : Amazon Neptune



Graph Data model can be used in fraud detection



5 (a) Describe why RDBMS can not be used in the Big Data for most of the use cases?
Marks [5]

- Seek time is improving slower than Transfer time
- If the data pattern is dominated by seeks, then it is difficult using RDBMS.
- B-Tree / RDBMS works well when you have small updates.
- RDBMS is good if you have structured data (based on schema)
- BigData typically contains unstructured and semi structured data (audio, video, pictures, text etc)

The RDBMS is not preferred for the following distinguished characters over HDFS.

	RDBMS	MapReduce
Data Size	GigaBytes	PetaBytes
Access	Interactive and Batch	Batch
Updates	Read / Write many times	Read many / Write once
Structure	Structured	Semi / Unstructured
Integrity	High	Low
Scaling	Non linear	linear

5 (b) List different issues with respect to impedance mismatch in RDBMS Marks [5]

- Term refers to the differences between database model and programming language model.
- Data type of attribute in RDBMS would be different from what we use in the program.
- Results of queries from the execution is multi set tuples which would not be supported by the programming model.
- Structural differences: Relational data used to be set of global, unnested data, tuples confirming to some schema. In OOP, all of them would be considered as objects.
- Manipulative differences: The operators defined are less. But, in programming model, you can custom define also.
- Transactional differences: The amount of transaction as such is native to specific attributes or structures in OO, but RDBMS support higher amount of transactions
- Structure Vs Behavior: RDBMS focuses on efficiency, integrity, fault tolerance. OO focuses on maintainability, extensibility, reusability etc

6 (a) Illustrate how Big Data can be used in health care. Marks [5]

- Management of chronic diseases to delivery of personalised medicine.
- Replace the guest work and intuition with precise analysis using data driven science.
- Evidence Based Medicines (EBM)
  - Collect data from EMR/EHR
  - Details about the diseases / symptoms
  - Details of their environments
  - Details of their origin / heredity / race
  - Details of their food habits
  - Details of their medicine / dosage
- Translational Medicine
  - Three pillars benchside, bedside and community.
    - harnessing knowledge from basic sciences to produce new drugs, devices, and treatment options for patients
    - end point is the production of a promising new treatment that can be used clinically or commercialized
- Insurance Companies
  - Comparative Effectiveness Research (CER) for reimbursement
    - Mine large claims
    - EHR/EMR
    - Economic / Geographic / Demographic data to determine treatment and therapies
- Genetic Analysis
  - Interactions between environmental factors and diseases in a neurological disease.
    - Huge data sets : 100K to 500K genetic variations
    - 1000 by 2000 matrix to interact with 500K variations
    - First order results : 500K
    - Second order results : 500K ^2
    - Third order results : 500K ^3
  - The goal is to find similarities between those data sets.

6 b) How Big Data solutions solve the problems on advertising and marketing? Marks [5]

Advertising:

- Three basic needs addressed by big data.
  - How much I need to spend on the next year?
  - How do I allocate that spend across marketing communication points.
  - How do I optimize advertising effectiveness against brand equity and Return of Investment (ROI)
- Measurement of advertising
  - End to end advertising (reach, resonance and reaction)
  - Across platforms (TV, digital, print, Social media)
  - Real time measurement (wherever possible)

Marketing:
Consumer Product Companies: P & G Generally get linked through social media.

Big Challenge is how to analyze 80% of irrelevant data and still do marketing.

- Customers are changed:
  - Customers are informative, wanted suitable information.
  - Integration and automation are the keys.
- Social and Affiliate Marketing
  - Old Era: Tupperware : Word of mouth.
  - New Era: Coupons - lot of sites
    - Anyone can recommend the users by just clicks.
- Marketing with Social Intelligence
  - Derives behavior from status updates, blogs, photos, likes

7 (a) Company "A" could not solve an issue through internal resources and knowledge. How Big Data and Open Source Methods help them to solve it Marks [4]

The company can solve the problems using crowd sourcing analytics.

Crowdsourcing is a recognition that you can 't possibly always have the best and bright-est internal people to solve all your big problems.

Netflix, an online DVD rental business, announced a contest to create a new predictive model for recommending movies based on past user ratings. The grand prize was $1,000,000! While this may seem like a PR gimmick, it wasn 't. Netfl ix already had an algorithm to solve the problem but thought there was an opportunity to realize additional model "lift," which would translate to huge top-line revenue. Netflix was an innovator in a space now being termed crowdsourcing.

Kaggle describes itself as "an innovative solution for statistical/analytics outsourcing." That 's a very formal way of saying that Kaggle manages competitions among the world 's best data scientists. Here 's how it works: Corporations, governments, and research laboratories are confronted with complex statistical challenges. They describe the problems to Kaggle and provide data sets. Kaggle converts the problems and the data into contests that are posted on its web site. The contests feature cash prizes ranging in value from $100 to $3 million. Kaggle 's clients range in size from tiny start-ups to multinational corporations such as Ford Motor Company and government agencies such as NASA.

7 (b) Distinguish intra, inter and trans firewall analytics. Marks [6]

- Intra Firewall
  - Insight and data analyzed inside the company
- Inter Firewall
  - Data is primarily kept inside the company domain
  - Insight is shared across the firewall
- Trans Firewall
  - Analyze data obtained from the public domain

Supply chains have evolved to connect multiple companies and enable them to collaborate to create enormous value to the end-consumer via concepts like CPFR, VMI, etc. Decision sciences is witnessing

a similar trend as enterprises are beginning to collaborate on insights across the value chain. For instance, in the health care industry, rich consumer insights can be generated by collaborating on data and insights from the health insurance provider, pharmacy delivering the drugs and the drug manufacturer. In-fact, this is not necessarily limited to companies within the traditional demand-supply value chain. For example there are instances where a retailer and a social media company can come together to share insights on consumer behavior that will benefit both players. Some of the more progressive companies are taking this a step further and working on leveraging the large volumes of data outside the firewall such as social data, location data, etc. In other words, it will be not very long before internal data and insights from within the fire-wall is no longer a differentiator. We call this trend the move from intra- to inter and trans-firewall analytics. Yesterday companies were doing functional silo based analytics. Today they are doing intra firewall analytics with data within the firewall. Tomorrow they will be collaborating on insights with other companies to do inter firewall analytics as well as leveraging the public domain spaces to do trans-firewall analytics.

Doing inter-firewall and trans firewall analytics is not without its challenges. Firstly as one moves outside the firewall the information to noise ratio increases putting additional requirements on analytical methods and technology requirements. Further, organizations are often limited by a fear of collaboration and an over-reliance on proprietary information. The fear of collaboration is mostly driven by competitive fears, data privacy concerns and proprietary orientations that limit opportunities for cross-organizational learning and innovation. While it is clear that the transition to inter and trans-firewall paradigm is not easy, we feel it will continue to grow and at some point, it will become a key arsenal available for decisions scientists to drive disruptive value and efficiencies.