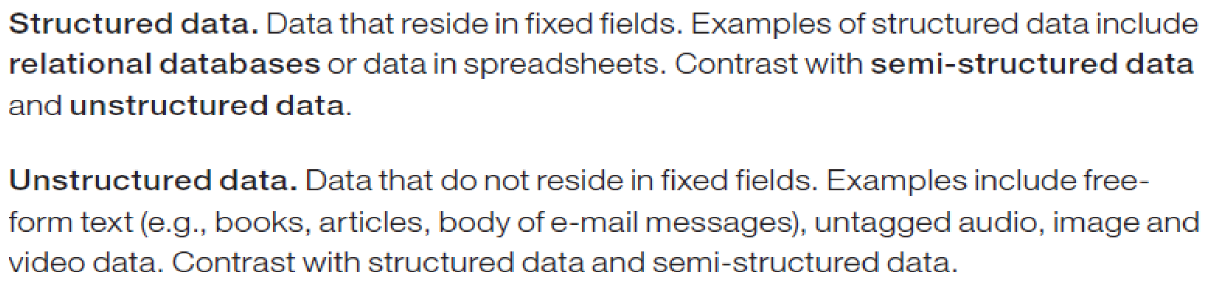
BharatiVidyapeeth's College of Engg. For Women
Computer Department

Class:T.E.ComputerEngg. **Subject:**Database Management System Lab

Important Questions Answers for Oral Exam Group B

1. Define & explain structured & unstructured data with example.



Structured data. Data that reside in fixed fields. Examples of structured data include **relational databases** or data in spreadsheets. Contrast with **semi-structured data** and **unstructured data**.

Unstructured data. Data that do not reside in fixed fields. Examples include free-form text (e.g., books, articles, body of e-mail messages), untagged audio, image and video data. Contrast with structured data and semi-structured data.

Eg.structuredMysql , Oracle unstructured: Mongodb, CouchDB,GraphDB

2. Compare between structured & unstructured database.

Structured Data

Some business experts estimate that about twenty percent of all data used in business decisions is structured data. **Structured data** is easily organized and generally stored in databases. Structured data generally consists of numerical information and is objective. It simply is data that exists - there is no interpretation.

Some types of structured data can be **machine generated**, such as data that comes from medical devices (heart rate, blood pressure), manufacturing sensors (rotation per minute, temperature), or web server logs (number of times a page is visited). Structured data can also be **human generated** - data such as age, zip code, and gender.

If we think about the structured data a web server log might keep, we could find out how many customer inquiries there have been, how long it took to solve a problem, and how the customer rated his experience.

Unstructured Data

Unstructured data may represent approximately 80% of the information that is used to make good business decisions.

Unstructured data is more subjective and is usually text heavy. It can't generally be put into a data structure, like columns or rows. Unstructured data can be found in documents, presentations, audio, images, videos, messages, and books. Unstructured data can also come from social media sites, such as Facebook, LinkedIn, Twitter, Tumblr, Flickr, Yelp, YouTube, and Pinterest. Some examples of unstructured data include customer reviews that describe how they feel about an experience, identified triggers for readmission to hospital care, or call center conversations.

3. Define NoSQL ? State its benefits over SQL.

NoSQL Storage
• Is designed when
– Storage is cheap
– Data transfer is fast
– Much more processing power is available
• Clustering of machines is also possible
– Applications are oriented towards consumption of User Generated Content
– Better on-screen user experience is in demand
NoSQL Storage
• Semi-structured
– Schemaless
• Consistency, Availability, Partition Tolerance
• High Availability through clustering
– expect failures
• Optimized for Reads
• Typically Scale Out

4. Explain unstructured Data models ?(ExplainMongoDB).⁄ Explain data modeling of NOSQL with example.

1. **NoSQL: Key/Value**
2. **NoSQL: Document**

MongoDB, CouchDB etc.
• Object Oriented data models
– Stores data in document objects having fields
– Basic and compound (list, dict) data types
• SQL like queries
• Transparent values
– Can be part of query
• Suited for product info and its reviews

3. **NoSQL: Column Family**

Cassandra, Big Table etc.
• Stores data in columns
• Transparent values
– Can be part of query
• SQL like queries
• Suited for search

4. **NoSQL: Graph**

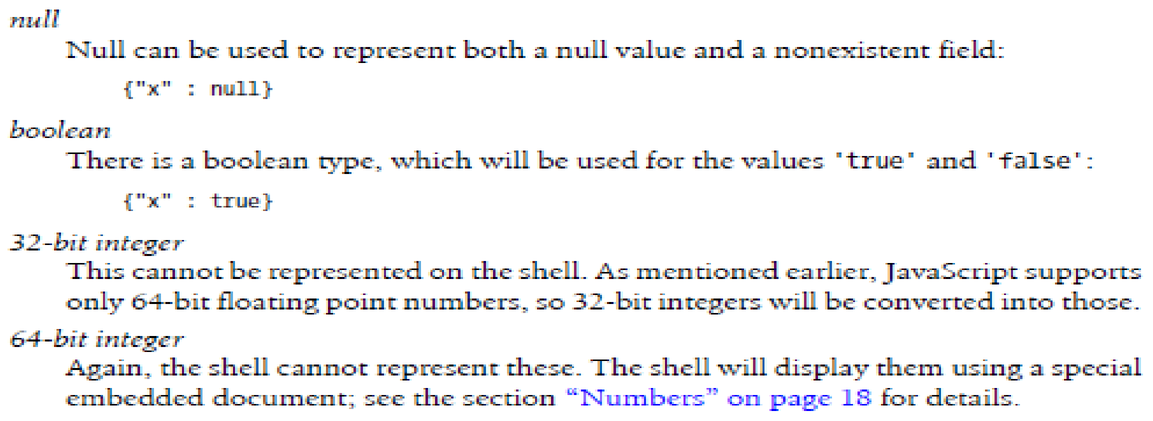
Neo4j
• Stores data in form of nodes and relationships
• Query is in form of traversal
• In-memory

• Suited for social graph

5. Compare NoSQLVs SQL.

SQL	NOSQL
MySql, Postgresql, Oracle etc. • Stores data in tables having columns – Basic (number, text) data types • Strong query language • Transparent values – Query language can read and filter on them – Relationship between tables based on values • Suited for user info and transactions	MongoDB, CouchDB etc. • Object Oriented data models – Stores data in document objects having fields – Basic and compound (list, dict) data types • SQL like queries • Transparent values – Can be part of query • Suited for product info and its reviews

6. List & Explain MongoDBdatatypes.



null

Null can be used to represent both a null value and a nonexistent field:

```
{"x" : null}
```

boolean

There is a boolean type, which will be used for the values 'true' and 'false':

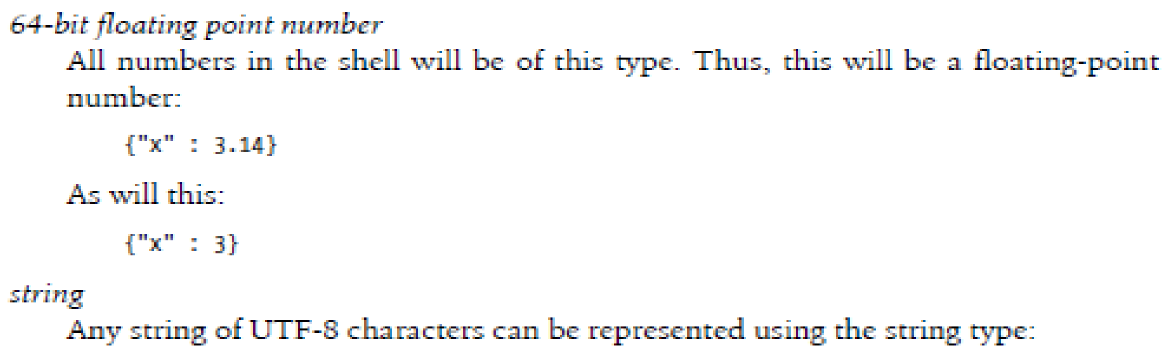
```
{"x" : true}
```

32-bit integer

This cannot be represented on the shell. As mentioned earlier, JavaScript supports only 64-bit floating point numbers, so 32-bit integers will be converted into those.

64-bit integer

Again, the shell cannot represent these. The shell will display them using a special embedded document; see the section "Numbers" on page 18 for details.



64-bit floating point number

All numbers in the shell will be of this type. Thus, this will be a floating-point number:

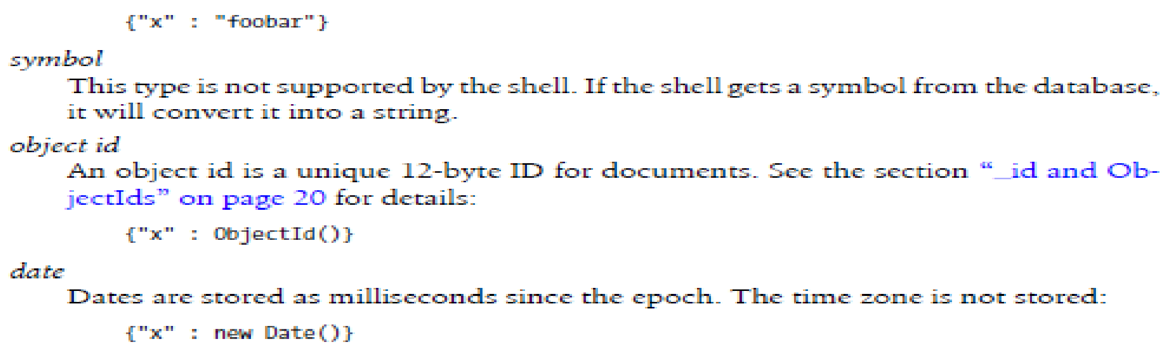
```
{"x" : 3.14}
```

As will this:

```
{"x" : 3}
```

string

Any string of UTF-8 characters can be represented using the string type:



```
{"x" : "foobar"}
```

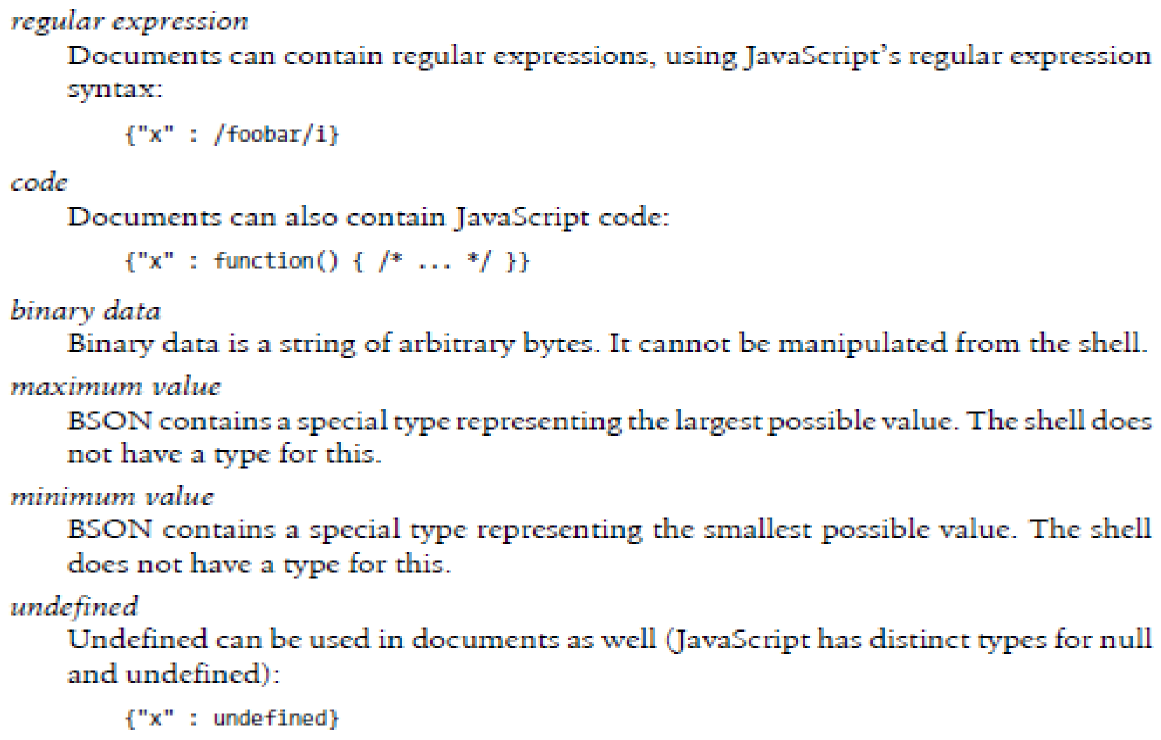
symbol
This type is not supported by the shell. If the shell gets a symbol from the database, it will convert it into a string.

object id
An object id is a unique 12-byte ID for documents. See the section "_id and ObjectIds" on page 20 for details:

```
{"x" : ObjectId()}
```

date
Dates are stored as milliseconds since the epoch. The time zone is not stored:

```
{"x" : new Date()}
```



regular expression
Documents can contain regular expressions, using JavaScript's regular expression syntax:

```
{"x" : /foobar/i}
```

code
Documents can also contain JavaScript code:

```
{"x" : function() { /* ... */ }}
```

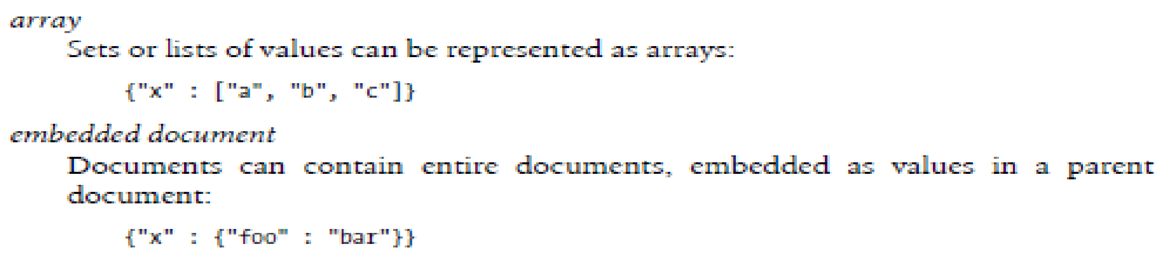
binary data
Binary data is a string of arbitrary bytes. It cannot be manipulated from the shell.

maximum value
BSON contains a special type representing the largest possible value. The shell does not have a type for this.

minimum value
BSON contains a special type representing the smallest possible value. The shell does not have a type for this.

undefined
Undefined can be used in documents as well (JavaScript has distinct types for null and undefined):

```
{"x" : undefined}
```



array
Sets or lists of values can be represented as arrays:

```
{"x" : ["a", "b", "c"]}
```

embedded document
Documents can contain entire documents, embedded as values in a parent document:

```
{"x" : {"foo" : "bar"}}
```

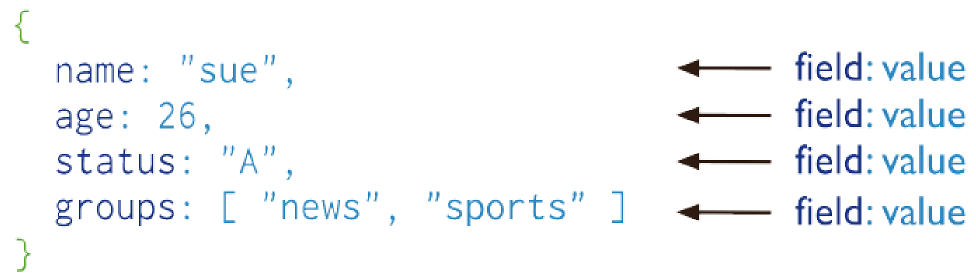
7. Explain data manipulation command (Insert , Upadate , Delete) for documents.

MongoDB provides rich semantics for reading and manipulating data. CRUD stands for create, read, update, and delete. These terms are the foundation for all interactions with the database.**study syntax with Example**

8. Explain Define MongoDB , collection , documents, Object ID.

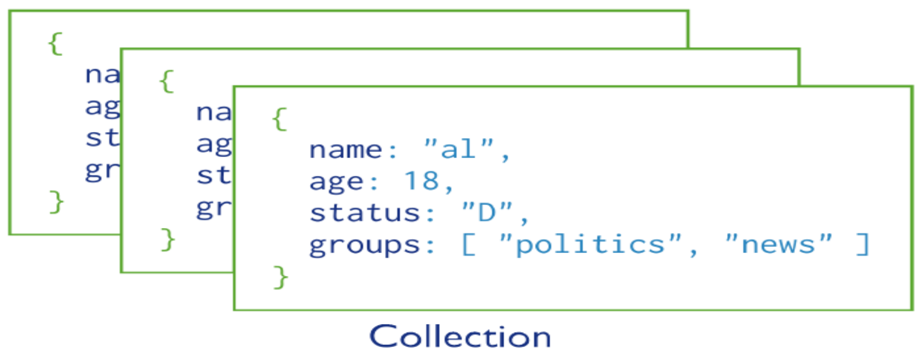
MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling.

MongoDB Document Structure



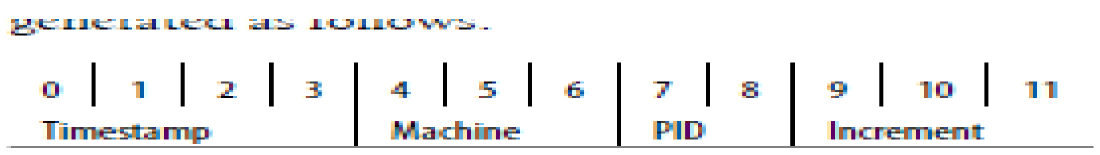
```
{
   name: "sue",
   age: 26,
   status: "A",
   groups: [ "news", "sports" ]
}
```

MongoDB Collection Structure



```
{
   name: "al",
   age: 18,
   status: "D",
   groups: [ "politics", "news" ]
}
```

Object ID is of 12 Bytes as follows:-



generated as follows:

0	1	2	3	4	5	6	7	8	9	10	11
Timestamp				Machine			PID		Increment		

9. Explain MongoDB

1. Document Database

A document is essentially an associative array
• Document = JSON object
• Document = PHP Array
• Document = Python Dict
• Document = Ruby Hash
• etc

2. **Open Source**

MongoDB is an open source project
• On GitHub
• Licensed under the AGPL
• Started & sponsored by MongoDBInc (formerly

known as 10gen)
• Commercial licenses available
• Contributions welcome

3. **. High Performance**
Written in C++
• Extensive use of memory-mapped files
i.e. read-through write-through memory caching.
• Runs nearly everywhere
• Data serialized as BSON (fast parsing)
• Full support for primary & secondary indexes
• Document model = less work
4. **. Scalability**
Auto-Sharding
• Increase capacity as you go
• Commodity and cloud architectures
• Improved operational simplicity and cost visibility

10. Explain Find method / Limit method / Skip method / Sort method.

The most common query options are limiting the number of results returned, skipping a number of results, and sorting. All of these options must be added before a query is sent to the database.

To set a limit, chain the limit function onto your call to find. For example, to only return three results, use this:

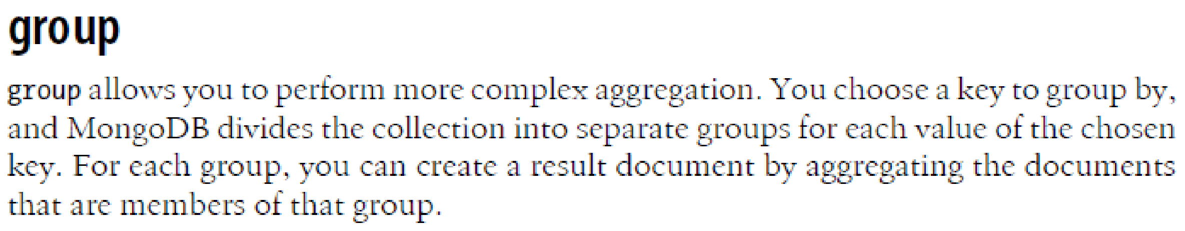
```
>db.c.find().limit(3)
```

If there are fewer than three documents matching your query in the collection, only the number of matching documents will be returned; limit sets an upper limit, not a lower limit.skip works similarly to limit:

```
>db.c.find().skip(3)
```

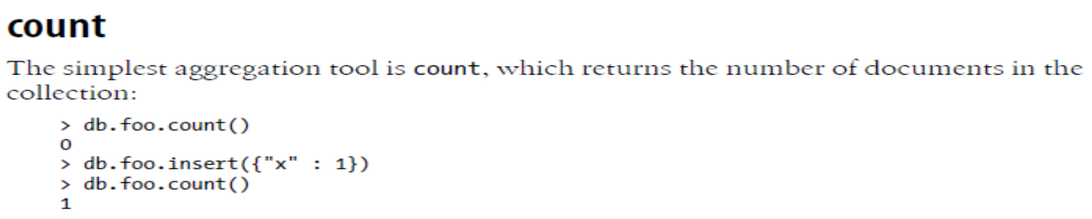
11. Explain Aggregation with MongoDB with example.

Group, COUNT,Distict,



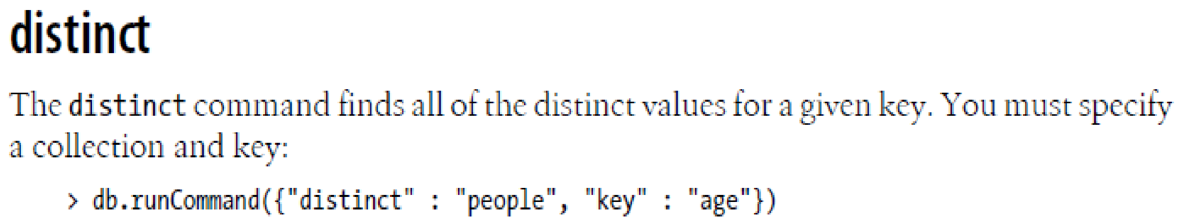
group

group allows you to perform more complex aggregation. You choose a key to group by, and MongoDB divides the collection into separate groups for each value of the chosen key. For each group, you can create a result document by aggregating the documents that are members of that group.


count

The simplest aggregation tool is **count**, which returns the number of documents in the collection:

```
> db.foo.count()
0
> db.foo.insert({"x" : 1})
> db.foo.count()
1
```


distinct

The **distinct** command finds all of the distinct values for a given key. You must specify a collection and key:

```
> db.runCommand({"distinct" : "people", "key" : "age"})
```

12. Explain any four / six aggregate function.

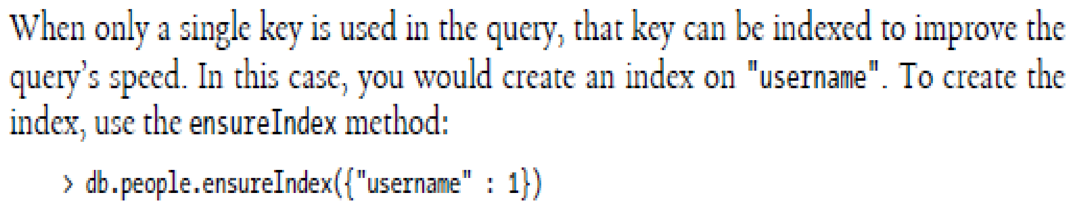
1]$group 2] $project
3] $match 4]$group
5]$sort 6]$limit

- Can be used as pipeline operator of linux
- Find total amount of order for individual customers having status A…..
- Db.orders.aggregate([{$match:{status:"A"}},{$group:{_id:"$cust_id",total:{$sum:"amount"}}}])

13. What is index ? Explain indexing with MongoDB.

Indexes are the way to make queries go *vroom. Database indexes are similar to a book's* index: instead of looking through the whole book, the database takes a shortcut and just looks in the index, allowing it to do queries orders of magnitude faster. Once it finds the entry in the index, it can jump right to the location of the desired document

14. Explain ensure index method with example.


When only a single key is used in the query, that key can be indexed to improve the query's speed. In this case, you would create an index on "username". To create the index, use the ensureIndex method:

```
> db.people.ensureIndex({"username" : 1})
```

15. Define & explain Big Data with example.

Big data is often characterized by 3Vs: the extreme volume of data, the wide variety of data types and the velocity at which the data must be processed. Although big data doesn't equate to any specific volume of data, the term is often used to describe terabytes, petabytesand even exabytes of data captured over time.

16. Map and Reduce in MongoDB

- MapReduceis aggregation tools. Everything described with count, distinct, and group can be done with MapReduce, and more. It is a method of aggregation that can be easily parallelized across multiple servers.
- It splits up a problem, sends chunks of it to different machines, and lets each machine solve its part of the problem. When all of the machines are finished, they merge all of the pieces of the solution back into a full solution.
- MapReduce has a couple of steps. It starts with the map step, which *maps an operation* onto every document in a collection. That operation could be either “do nothing” or “emit these keys with *X values.” There is then an intermediary stage called the shuffle* step: keys are grouped and lists of emitted values are created for each key. The reduce takes this list of values and *reduces it to a single element. This element is returned to* the shuffle step until each key has a list containing a single value: the result.

17. Explain benefits of NOSQL

When compared to relational databases, NoSQL databases are more scalable and provide superior performance, and their data model addresses several issues that the relational model is not designed to address:

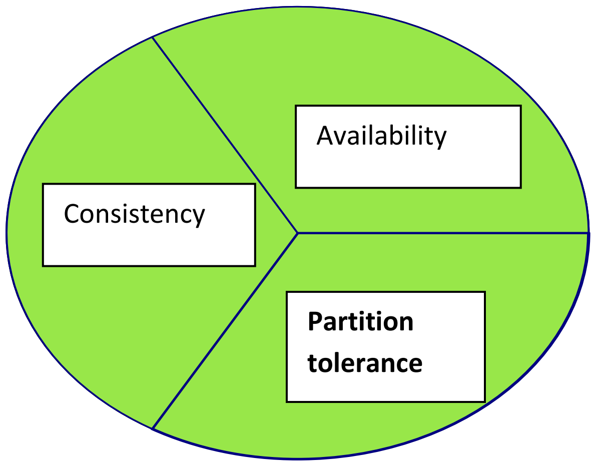
- Large volumes of rapidly changing structured, semi-structured, and unstructured data
- Agile sprints, quick schema iteration, and frequent code pushes
- Object-oriented programming that is easy to use and flexible
- Geographically distributed scale-out architecture instead of expensive, monolithic architecture

18. Explain Base properties of CAP theorem

AvailabilitySystem is available during software and hardware upgrades and node failures.

ConsistencyOnce a writer has written, all readers will see that write

Partition toleranceA system can continue to operate in the presence of a network partitions.



- CAP theorem states that there are three basic requirements which exist in a special relation when designing applications for a distributed architecture.
- Consistency - This means that the data in the database remains consistent after the execution of an operation. For example after an update operation all clients see the same data.
- Availability - This means that the system is always on (service guarantee availability), no downtime.
- Partition Tolerance - This means that the system continues to function even the communication among the servers is unreliable, i.e. the servers may be partitioned into multiple groups that cannot communicate with one another.

19. What is JSON?

MongoDB stores data in the form of documents, which are JSON-like field and value pairs. Documents are analogous to structures in programming languages that associate keys with values (e.g. dictionaries, hashes, maps, and associative arrays). Formally, MongoDB documents are BSON documents. BSON is a binary representation of JSON with additional type information. In the documents, the value of a field can be any of the BSON data types, including other documents, arrays, and arrays of documents.

20. Explain BASE Properties.

In a ‘shared nothing’ environment, BASE is implemented: Optimistic behaviour: accept temporary database inconsistencies –

Basically Available [guaranteed thanks to replication] –

Soft state [it’s the user’s (application’s) task to guarantee consistency] –

Eventually consistent (weakly consistent) [database will be consistent in the longer run; ‘stale’ data is OK]

21.CompareACID against BASE

ACID(RDBMS)	**BASE(NoSQL)**
strong consistency	weak consistency (=> allow stale data)
isolation	last write wins
transaction	program managed
robust database	simple database
simple code ((SQL)	complex
code available & consistent	available & partition-tolerant
scale-up (limited)	scale-out (unlimited)
shared-something (disk, mem, proc)	shared-nothing (parallellizable)