1.Write a note on cap therom?

The CAP Theorem is comprised of three components (hence its name) as they relate to distributed data stores:

* Consistency- All reads receive the most recent write or an error.
* Availability- All reads contain data, but it might not be the most recent.
* Partition tolerance-The system continues to operate despite network failures (ie; dropped partitions, slow network connections, or unavailable network connections between nodes.)

In normal operations, your data store provides all three functions. But the CAP theorem maintains that when a distributed database experiences a network failure, you can provide either consistency or availability.

All other times, all three can be provided. But, in the event of a [network failure,](https://www.bmc.com/blogs/mtbf-vs-mtff-vs-mttr-whats-difference/) a choice must be made.

In the theorem, [partition tolerance is a must.](https://codahale.com/you-cant-sacrifice-partition-tolerance/) The assumption is that the system operates on a distributed data store so the system, by nature, operates with network partitions. Network failures will happen, so to offer any kind of reliable service, partition tolerance is necessary—the P of CAP.

That leaves a decision between the other two, C and A. When a network failure happens, one can choose to guarantee consistency or availability:

* High consistency comes at the cost of lower availability. High availability comes at the cost of lower consistency.

Consistency in CAP is different than that of [ACID.](https://www.bmc.com/blogs/acid-atomic-consistent-isolated-durable/) Consistency in CAP means having the most up-to-date information. (ACID refers to a different database event. In ACID, consistency means any new transaction to the database won’t corrupt the database.)

Database options for consistency:

* MongoDB
* Redis
* HBase

Database options for availability:

* [Cassandra](https://www.bmc.com/blogs/apache-cassandra-introduction/)
* DynamoDB
* Cosmos DB

Some database options, like Cosmos and Cassandra, allow a user to turn a knob on which guarantee they prefer—consistency or availability.

• DynamoDB

• Cosmos DB

***2.Explain Rest API?***

# REST(Representational State Transfer)

* REST protocols overcome SOAP's dependency on XML by supporting data transmission in multiple formats such as JSON (most prominent), HTML, Python, plain text as well as media files
* REST APIs follow a client-server architecture and must be *stateless*.
* *Stateless* communication implies that no client data is stored between GET requests.
* These GET requests must be distinct and disconnected.REST assigns every operation a unique URL, so when the server receives a request, it knows which instructions to execute to fulfil the request.
* REST also supports caching. So, the browser can store the results obtained from the request locally and retrieve it periodically as needed, thereby increasing speed and efficiency.

A typical REST request has the following components:

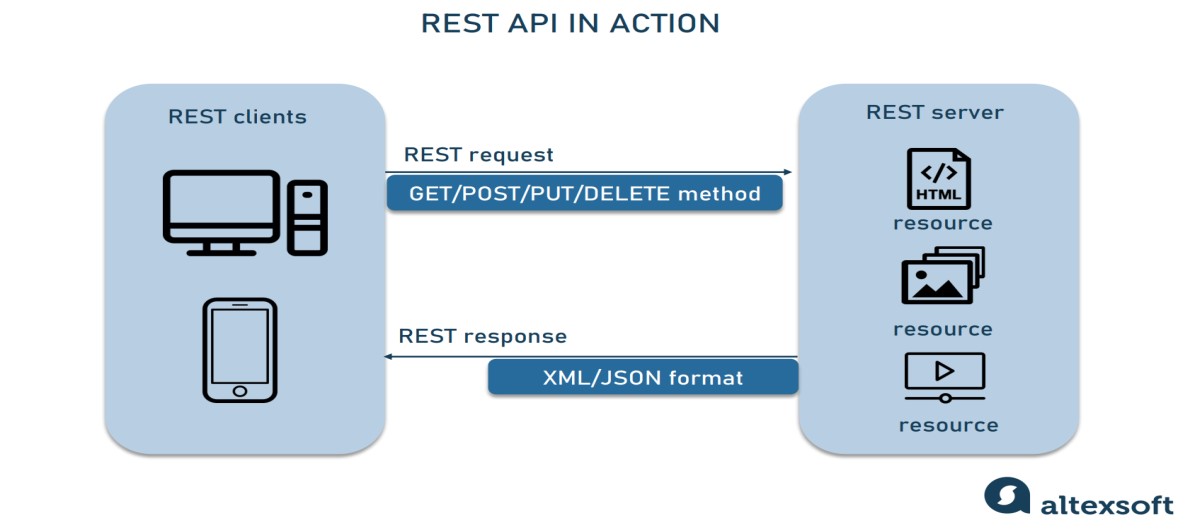
1. ***Endpoint* :** The destination URL from which data is being requested.
2. ***Method* :** We use predefined methods such as GET, POST, PUT or DELETE to fetch the data. These methods vary from one other. Ex. in when using GET, the data is appended to the end of the URL string, whereas in POST, the data is sent along with the HTTP request.
3. ***Headers* :** They define the request's details and dictate the proper format in which the response must be received.
4. ***body (data)* :** The actual data sent by the service.

# Overview of REST

* **R**epresentational **S**tate **T**ransfer (REST) is an architectural style that defines a set of constraints(rules) to be used for creating web services.
* **REST API** is a way of accessing web services in a simple and flexible way without having any processing.
* REST technology is generally preferred to the more robust Simple Object Access Protocol (SOAP) technology, simple and flexible making it more suitable for internet usage.
* It’s used to fetch or give some information from a web service. All communication done via REST API uses only HTTP request.

# Working

* A request is sent from client to server in the form of a web URL as HTTP GET or POST or PUT or DELETE request.
* After that, a response comes back from the server in the form of a resource which can be anything like HTML, XML, Image, or JSON.
* But now JSON is the most popular format being used in Web Services.



# Properties of REST API

1. **Performance:** REST can support the Performance goal by using caches to keep available data close to where it is being processed.

2 **Scalability:** Scalability means to consistently provide service regardless of the increase or decrease of web users.

1. **Simplicity**: By separating the functionality within components in a system, we can induce the simplicity of the architectural styles.
2. **Modifiability:** Modifiability represents the ease at which changes can be incorporated into the architecture.
3. **Visibility:** Visibility refers to the ability of a component to monitor or mediate the interaction between two components.

***3.Explain https and method?***

**HTTP Concepts** o HTTP stands for Hypertext Transfer Protocol.

* + Hypertext Transfer Protocol is a set of rule which is used for transferring the files like, audio, video, image, text and other multimedia files on the WWW.
  + HTTP is an application-level protocol. The communication usually takes place through TCP/IP sockets, but any reliable transport can also be used. o This latest version is already in use on the web with the help of UDP (User Datagram Protocol) instead of TCP (Transmission Control Protocol) for the underlying transport protocol.
  + HTTP is used to make communication between a variety of hosts and clients. It supports a mixture of network configuration.
  + HTTP is a protocol that is used to transfer the hypertext from the client end to the server end, but HTTP does not have any security.
  + Whenever a user opens their Web Browser, that means the user indirectly uses HTTP

**HTTP working**

* + As a request-response protocol, HTTP gives users a way to interact with web resources such as HTML files by transmitting hypertext messages between clients and servers. HTTP clients generally use [Transmission Control Protocol (TCP)](https://www.extrahop.com/resources/protocols/tcp/) connections to communicate with servers.

**HTTP Method (GET, POST, PUT, DELETE)**

* 1. **GET** requests a specific resource in its entirety
  2. **POST** adds content, messages, or data to a new page under an existing web resource

**PUT** directly modifies an existing web resource or creates a new URI if need be 4. **DELETE** gets rid of a specified resource

***4.Explain the acid base model?***

## ACID Model

The ACID database transaction model ensures that a performed transaction is always consistent. This makes it a good fit for businesses which deal with [online transaction processing](https://phoenixnap.com/kb/oltp-database) (e.g., finance institutions) or online analytical processing (e.g., [data warehousing)](https://phoenixnap.com/kb/data-warehouse-architecture-explained). These organizations need database systems which can handle many small simultaneous transactions. There must be zero tolerance for invalid states.

**ACID stands for:**

Atomic – Each transaction is either properly carried out or the process halts and the database reverts back to the state before the transaction started. This ensures that all data in the database is valid.

Consistent – A processed transaction will never endanger the structural integrity of the database.

Isolated – Transactions cannot compromise the integrity of other transactions by interacting with them while they are still in progress.

Durable – The data related to the completed transaction will persist even in the cases of network or power outages. If a transaction fails, it will not impact the manipulated data.

## ACID Use Case Example

Financial institutions will almost exclusively use ACID databases. Money transfers depend on the atomic nature of ACID.

An interrupted transaction which is not immediately removed from the database can cause a lot of issues. Money could be debited from one account and, due to an error, never credited to another.

One safe way to make sure your database is ACID compliant is to choose a relational database management system. These include MySQL, PostgreSQL, Oracle, SQLite, and Microsoft SQL Server.

5.Explain the key component of MongoDB Architecture?

## Key Components of MongoDB Architecture

Below are a few of the common terms used in MongoDB

**\_id** – This is a field required in every MongoDB document. The \_id field represents a unique value in the MongoDB document. The \_id field is like the document’s primary key. If you create a new document without an \_id field, MongoDB will automatically create the field. So for example, if we see the example of customer table, Mongo DB will add a 24 digit unique identifier to each document in the collection.

\_Id

CustomerID

CustomerName

OrderID

27d784

a8a4246bd

563479cc8

11

Guru99

111

47d784

a8a4246bd

563479cc7

22

Trevor Smith

222

57d784

a8a4246bd

563479cc9

33

Nicole

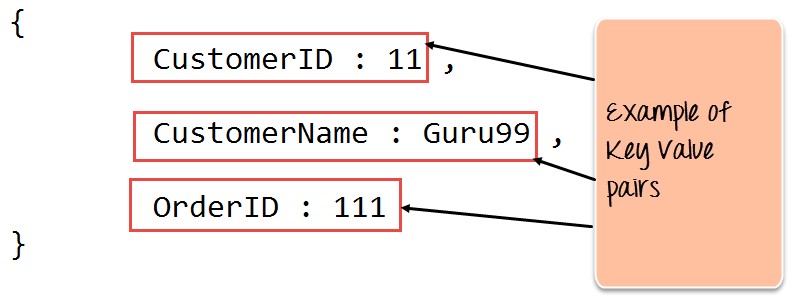
333

**Collection** – This is a grouping of MongoDB documents. A collection is the equivalent of a table which is created in any other RDMS such as Oracle or MS SQL. A collection exists within a single database. As seen from the introduction collections don’t enforce any sort of structure. **Cursor** – This is a pointer to the result set of a query. Clients can iterate through a cursor to retrieve results.

**Database** – This is a container for collections like in RDMS wherein it is a container for tables. Each database gets its own set of files on the file system. A MongoDB server can store multiple databases.

**Document** – A record in a MongoDB collection is basically called a document. The document, in turn, will consist of field name and values.

**Field** – A name-value pair in a document. A document has zero or more fields. Fields are analogous to columns in relational databases.The following diagram shows an example of Fields with Key value pairs. So in the example below CustomerID and 11 is one of the key value pair’s defined in the document.



**JSON** – This is known a[s JavaScript O](https://www.guru99.com/interactive-javascript-tutorials.html)bject Notation. This is a human-readable, plain text format for expressing structured data. JSON is currently supported in many programming languages.

6.Explain the basic features of mysql database?

**NoSQL** refers to a database that is non-relational in nature. NoSQL simply means “No use of Structural Query Language,” making it a non-traditional kind of database. Some of the basic features of non-relational databases can be seen below:

### Schema flexibility

The data stored in non-relational databases normally has no fixed format. It can exist in free form or a no-schema manner. This entails that a NoSQL database is characterized by **flexible schema**.We can see the schema flexibility in the MongoDB code below.

var personalSchema = new Schema({

    name: { type: String, default: 'unknown' },

    age: { type: Number, min: 18, index: true },

        date: { type: Date, default: Date.now },

});

var personalModel = mongoose.model('Person', personalSchema);

var comment1 = new personalModel({

    name: 'Alvan',

    age: '20', 

});

comment1.save(function (err, comment) {

    if (err) console.log(err);

    else console.log('Preview your entered details as was saved:', comment);

});

### Horizontal scaling

Scaling in a database entails increasing the storage capacity of a database as the application users begin to grow. The increase in the number of users of an application would necessitate an increase in the application’s storage capacity in order to store the data of new users.

This attribute has previously been a difficult one to incorporate into databases. But in NoSQL, **horizontal scaling** is a possible feature that involves adding additional nodes to the database so as to share the increasing load of storage. This feature is only present in non-relational databases, not in relational databases.

### NoSQL data model enables quicker queries

Data is not modeled as fixed tables with fixed rows and columns in a NoSQL database. Instead, it is modeled as graphs with nodes and edges, as well as key-value pairs and other forms. One does not need to determine and declare a table schema before inputting data into a NoSQL database. The flexibility in the data entry format allows for easier data entry in NoSQL databases.

{

\_id: <objectId1>,

 username: "mongoose231",

 Contact:{

          phone: "081456893"

          email: "mongoosemodel@gmail.com"

}

access : {

          level:5,

          group: "dev"

         }

}

The data structure in MongoDB given above permits one to embed document structures in a field within a document. This allows for easy retrieval and manipulation of related data in any given database operation.

### Easy use for developers

NoSQL database is characterized by a **flexible storage structure**. For example, MongoDB is a NoSQL database that uses a document structure in its data storage. One can store both structured and unstructured data in a NoSQL database.

The reason for this flexibility is that most document formats match the object format in most programming languages. This makes it easier for developers to input new data in the database without having to normalize the new data entry with the existing format in the database.