SAQ

**1. Case study of Google distributed file systems.**

Google file system is a scalable distributed file system developed by Google to provide efficient and reliable access to data using large clusters of commodity hardware.
- It is designed to meet the rapidly growing demand of Google's data processing need.
- It provides performance, scalability, reliability and availability of data across distributed system for handling and processing big data.

Characteristics of GFS
1. Files are organised hierarchically in directories and identified by path name.
2. It supports all the general operations on files like read, writes, open, delete, etc.

Common goals of GFS
1. Performance
2. Reliability
3. Automation
4. Fault Tolerance
5. Scalability
6. Availability

**2. Frame about the various faults and transparencies involved in distributed systems.**

There are three main types of faults: transient, intermittent, and permanent

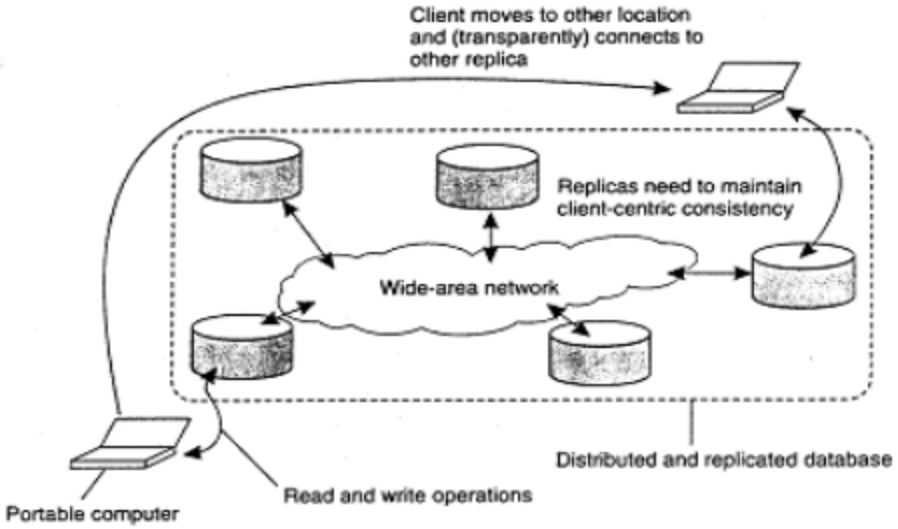
A transient fault is a fault that happens once, and then doesn't ever happen again.An intermittent fault is one that occurs once, seems to go away, and then occurs again! a permanent fault is one that just does not go away after it first occurs.

Types of Transparency in Distributed Systems:

1. Access Transparency: Access Transparency allows the same operations to be used to access local and remote resources.
2. Location Transparency: Location Transparency permits access to resources regardless of their physical or network location.
3. Concurrency Transparency: Concurrency Transparency permits many processes to run in parallel using shared resources without interfering with one another.

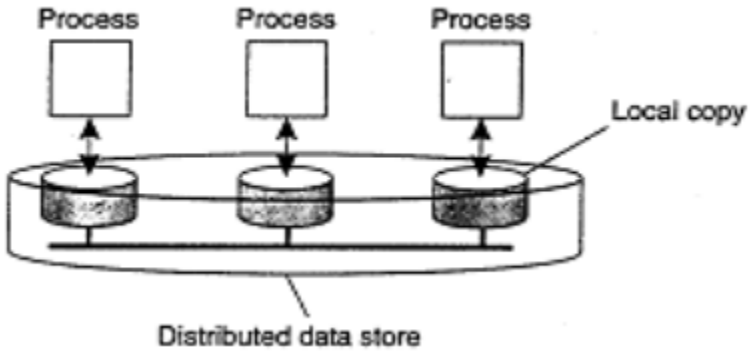
**3. Negotiate between Client-Centric and Data-Centric Consistency models.**

Client centric model:Client-centric Consistency Model defines how a data-store presents the data value to an individual client when the client process accesses the data value across different replicas. It is generally useful in applications where: one client always updates the data-store.




**Client centric model**

Data centric model:Data-centric consistency models describe how the replicated data is kept consistent across different data-stores, and what the process can expect from the data-store. Data-centric models are too strict when: most operations are read operations.




**Data Centric model**

**4. List out any four CORBA services.**

CORBA SERVICES are

- CORBA Naming Service.
- CORBA Event Service.
- CORBA Notification Service.
- CORBA Security Service.

**5. Distinguish between CODA and Andrew File system**

| CODA | AFS |
|---|---|
| Coda is a distributed file system developed as a research project at Carnegie Mellon University in 1987 | Developed by Carnegie Mellon University as part of Andrew distributed computing environments (in 1986) |
| Coda uses a local cache to provide access to server data when the network connection is lost. | A research project to create a campus-wide file system. |
| Continued operation during partial network failures in server network | Public domain implementation is available on Linux (LinuxAFS) |
| Well-defined semantics of sharing, even in the presence of network failure | AFS is implemented as two software components that exist at UNIX processes called Vice and Venus. |

**6. List out any three differences between Map Reduce and Pig**

| Map Reduce | Pig |
|---|---|
| It is a Data Processing Language. | It is a Data Flow Language. |
| It converts the job into map-reduce functions. | It converts the query into map-reduce functions. |
| It is a Low-level Language. | It is a High-level Language |

## 7. What is Grid computing?

Grid computing is a computing infrastructure that combines computer resources spread over different geographical locations to achieve a common goal. All unused resources on multiple computers are pooled together and made available for a single task. Organizations use grid computing to perform large tasks or solve complex problems that are difficult to do on a single computer.

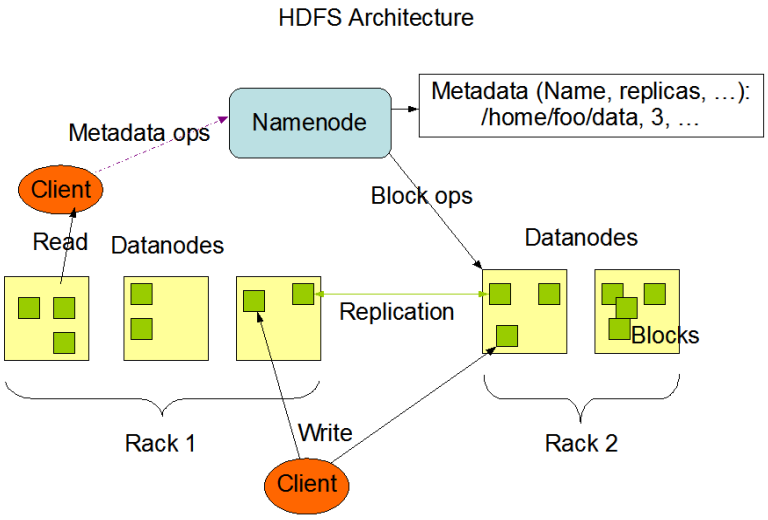
## 8.Discuss REST and Web Services

REST is a style of architecture with a set of constraints which allows data transfer over a standardized interface, such as HTTP. REST relies on a simple URI to make a request

A web service is a set of functions that are published. to a network for use by other programs. Many people regard web services. as a technology only for publishing software services on the Internet via.

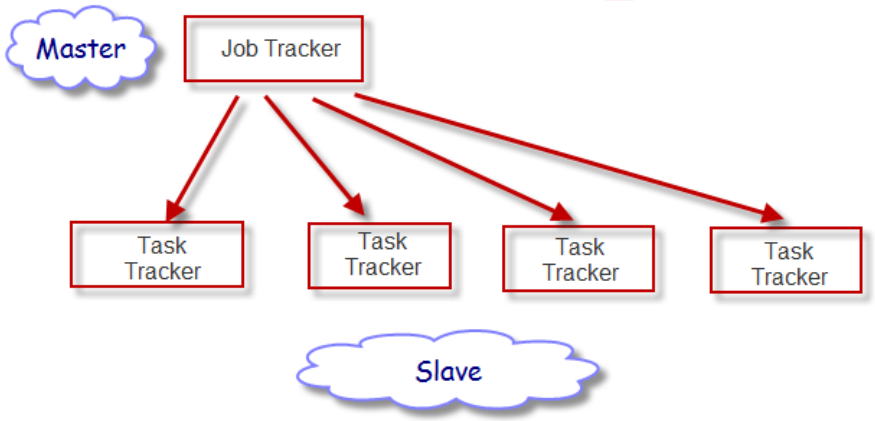
## 3.Justify the role of the Name node, Job Tracker and Test tracker services of HDFS using a neat diagram.

**The NameNode** is the centerpiece of an HDFS file system. It keeps the directory tree of all files in the file system, and tracks where across the cluster the file data is kept.
The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients.




Job Tracker

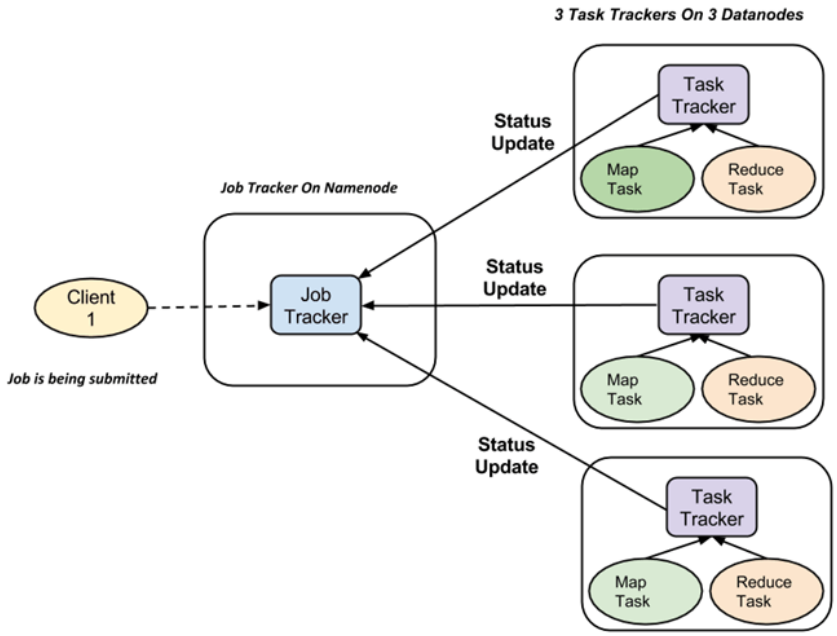
. Client applications (2) submit jobs to the Job tracker. The JobTracker submits the work to the chosen TaskTracker nodes
The JobTracker submits the work to the chosen TaskTracker nodes. The TaskTracker nodes are (3) monitored. If they do not submit heartbeat signals often enough, they are deemed to have failed and the work is scheduled on a different TaskTracker.




(1) Job tracker's role is resource management, tracking resource availability and tracking the progress of fault tolerance.

Job tracker communicates with the Namenode to determine the location of data. Finds the task tracker nodes to execute the task on given nodes. It tracks the execution of MapReduce from local to the Slave

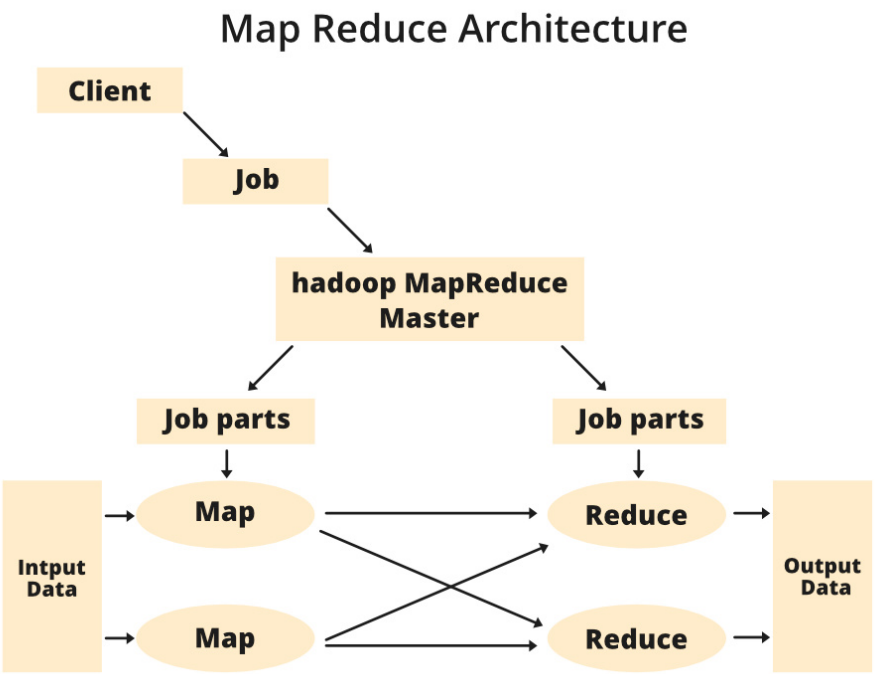
**A TaskTracker i**s a node in the cluster that accepts tasks - Map, Reduce and Shuffle operations - from a JobTracker. Every TaskTracker is configured with a set of slots, these indicate the number of tasks that it can accept




here are the five core **Hadoop services**:
namenode, secondary namenode, datanode, jobtracker, and tasktracker, each a separate daemon. Services are run on nodes of the cluster

## 6.Discuss the Map Reduce Model?

MapReduce is a programming paradigm that enables massive scalability across hundreds or thousands of servers in a Hadoop cluster. As the processing component, MapReduce is the heart of Apache Hadoop.



The term "MapReduce" refers to two separate and distinct tasks that Hadoop programs perform. The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs).
The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.

MapReduce programming offers several benefits to help you gain valuable insights from your big data:

- Scalability. Businesses can process petabytes of data stored in the Hadoop Distributed File System (HDFS).
- Flexibility. Hadoop enables easier access to multiple sources of data and multiple types of data.
- Speed. With parallel processing and minimal data movement, Hadoop offers fast processing of massive amounts of data.
- Simple. Developers can write code in a choice of languages, including Java, C++ and Python.

Generally MapReduce paradigm is based on sending the computer to where the data resides!

MapReduce program executes in three stages, namely map stage, shuffle stage, and reduce stage.

- Map stage – The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.
- Reduce stage – This stage is the combination of the Shuffle stage and the Reduce stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.

Shuffle Stage: Shuffle phase in Hadoop transfers the map output from Mapper to a Reducer in MapReduce.

# Distributed System

## Andrew File System

→ Andrew File System is a distributed file system. It uses the client/server model, where all the files are stored on the files server machine

→ Andrew File system is a location independent file system. Here files can be accessed or shared from any location

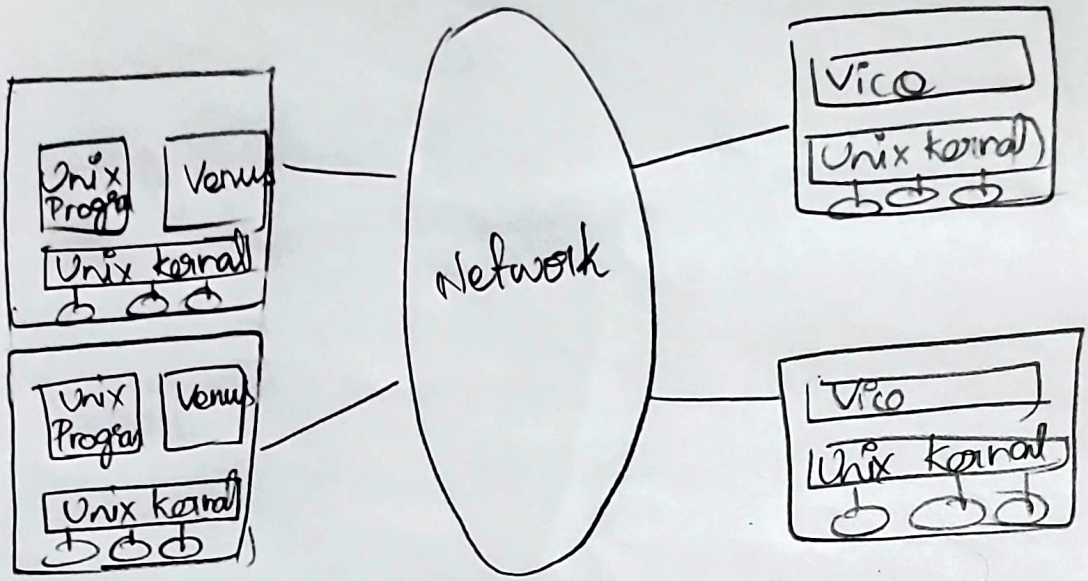
→ In this fib system internet is required

→ It is set of trusted servers

→ It konsist of two elements

vice -- It is a server side process on Unix kernel

venus – It is a client side process on Unix kernel




Features:

1. File backup
2. File Security
3. Physical Security
4. Availability
5. Authentication
6. Space to User

→ Distributed file System that enables files from any AFS machine across the country to be accessed easily as it stored on local servers

CODA:- It is a distributed file System that offers replication.

↳ Replicates on Server.

↳ Creates a Cache on client and lets user make changes to the cache.

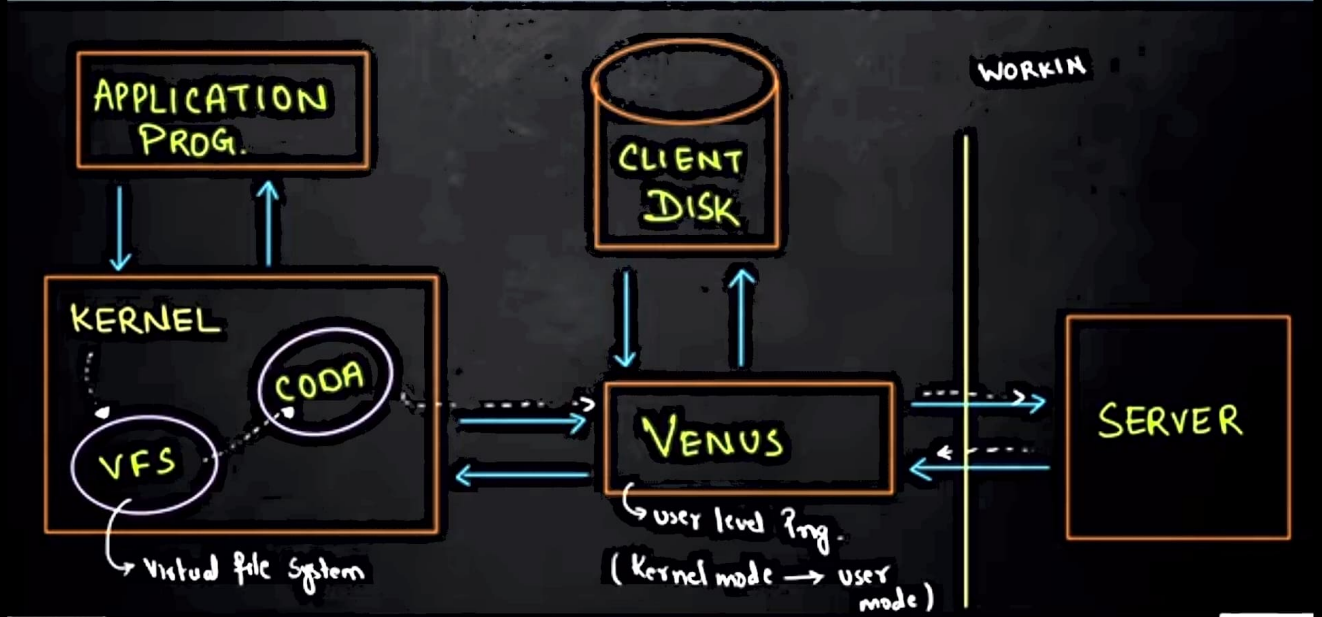
Features:-

↳ i) Freely available Source Code.
ii) Server replication.
iii) High Performance.
iv) Robust during partial N/W failure.
v) Scalable
vi) Security < authentication, encryption

CODA implemented as prototype of highly available DFS for disconnected operation of clients.

→ Provides Common Namespace for all files that client Share.

→ 'CODA' running in client will fetch req. files from appropriate Server and make it available to the client.


APPLICATION PROG.
CLIENT DISK
WORKIN
KERNEL
CODA
VFS
VENUS
SERVER
user level Prog.
(Kernel mode → user mode)
Virtual file System

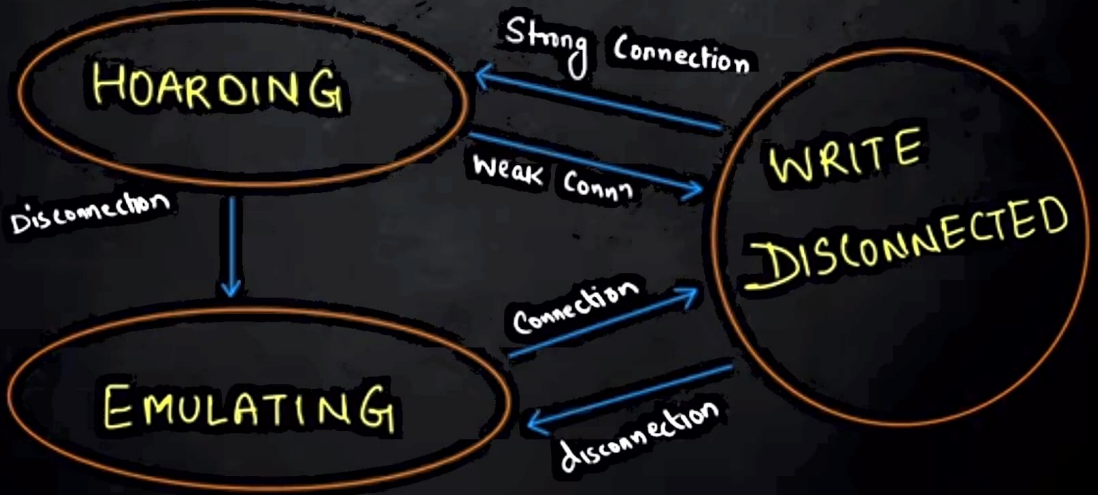
## How CODA Supports mobility:-

**HOARDING:-** Prefetching mechanism in CODA.

↳ If client is Connected to a Server with Strong Connection, hoarding prefetches files Currently used.

↳ As Soon as client is disconnected, Appl" works on replicates and goes to emulating State.

(IMP) ↳ After reconnection CODA Compares the replicates with files on Server. If 'CODA' notice that two diff. users have changed file, reintegration fails and CODA Saves changed file as a copy on Server to allow mannual reintegration.




Like and Share the Video – Don't Forget to Subscribe Channel and Press

EEC Classes

# 2-phase Commit Protocol

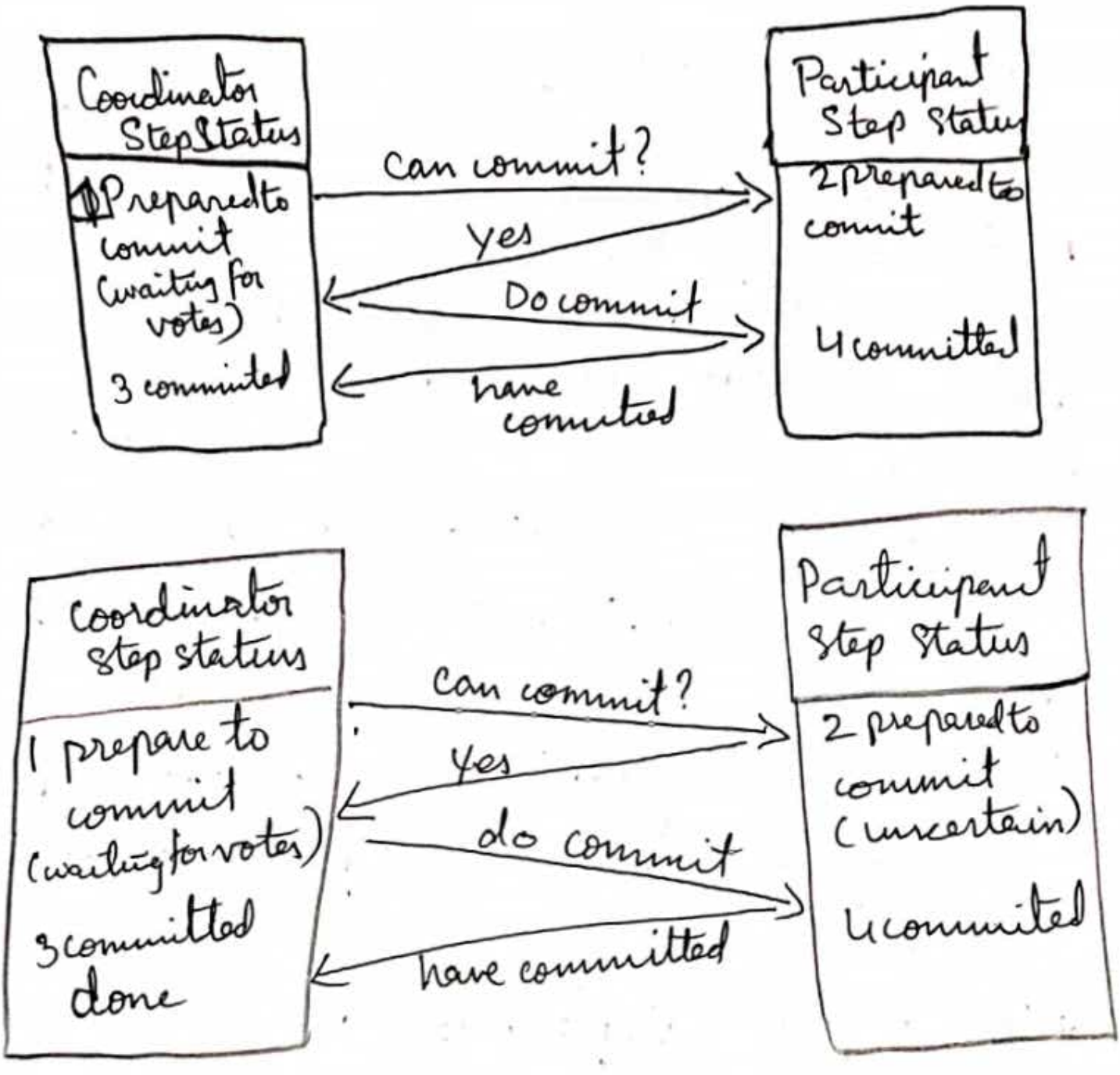
A 2-phase commit is a standardized protocol that ensures that a database commit is implementing in the situation where a commit operation must be broken into two seperate parts.

* In database management, saving data changes is known as a commit and undoing changes is known as rollback. Both can be achieved easily using transaction logging when a single server is involved, in Two phase commit as this implies the arranges activities and synchronization between distributed servers.

* The 2-phase implemented as

Phase 1: Each server that needs to commit data writes its data records to the log. it is a server is unsuccessful. it responds with a failure message. If successful the server replies with an OK message.

Phase 2: This phase begins after all participants respond OK. Then, the coordinator sends a signal to each error with commit instruction. After committing, each writes and sends the coordinator a message that fits commit has been succesfully implemented.


Coordinator Step Status
1 Prepared to commit (waiting for votes)
3 committed
Participant Step Status
2 Prepared to commit
4 committed
Can commit?
Yes
Do commit
have committed
Coordinator step status
1 prepare to commit (waiting for votes)
3 committed
done
Participant Step Status
2 prepared to commit (uncertain)
4 commited
Can commit?
Yes
do commit
have committed