full one-shot videos on :**JK Coding Pathshala YouTube channel**

JK Coding Pathshala

https://youtube.com/@jayeshkande9215?feature=shared

Mapping of Course Outcomes for Unit III	**CO3**	
Unit IV	**DATA STORAGE AND SECURITY IN CLOUD**	**(06 hrs)**
Cloud file systems: GFS and HDFS, BigTable, HBase and Dynamo Cloud data stores: Datastore and Simple DB Gautam Shrauf, Cloud Storage-Overview, Cloud Storage Providers. **Securing the Cloud**- General Security Advantages of Cloud-Based Solutions, Introducing Business Continuity and Disaster Recovery. Disaster Recovery- Understanding the Threats.		

Unit IV: Data Storage and Security in Cloud

1. Cloud File Systems:

- •GFS (Google File System)
- •HDFS (Hadoop Distributed File System)
- •BigTable
- •HBase
- •Dynamo Cloud data stores
- •Datastore and Simple DB (Gautam Shrauf)
- •Cloud Storage Overview
- •Cloud Storage Providers

2. Securing the Cloud:

- •General Security Advantages of Cloud-Based Solutions
- •Business Continuity and Disaster Recovery
- •Disaster Recovery: Understanding the Threats

Q3) a) Describe google file system architecture with neat diagram. **[8]**

b) Explain the features and advantages of Dynamo DB. Explain how it is different than RDBMS. **[9]**

OR

Q4) a) Describe Hadoop Distributed File System with neat diagram. **[8]**

b) What are different types of disasters and how the disaster recovery is done on cloud platform. **[9]**

Q3) a) Draw & explain General Architecture of Google File System (DFS)? **[6]**

b) Discuss the terms: Big Table, HBase and Dynamo cloud data stores in cloud computing. **[6]**

c) Discuss the terms Business Continuity and Disaster Recovery in cloud Computing? **[5]**

OR

Q4) a) Draw & explain General Architecture Hadoop Distributed File System (HDFS)? **[6]**

b) Differentiate between Big Table and HBase of cloud computing. **[6]**

c) What are the General Security Advantages of Cloud-Based Solutions? **[5]**

Q3) a) List and explain the security issues in cloud. **[6]**

b) Draw & explain General Architecture Hadoop Distributed File System (HDFS). **[6]**

c) What is mean by Disaster Recovery? Discuss Threats in Disaster Recovery. **[5]**

OR

P.T.O.

Q4) a) Write a short note on **[6]**

i) How to Approach Business Continuity

ii) Architect for Failure

b) Draw & explain General Architecture Hadoop Distributed File System (HDFS). **[6]**

c) What is fault tolerance. Explain characteristics of fault tolerance. **[5]**

Q3) a) Enlist the different components of google file system, explain the significance of each. **[9]**

b) Explain the features and advantages of DynamoDB. Differentiate between SQL and NoSQL. **[9]**

OR

Q4) a) What are the advantages of HDFS and how it is different than GFS. **[9]**

b) Explain various stages in MapReduce with an Example? **[9]**

Q3) a) Write a Short Note on Simple DB **[6]**

b) What is fault tolerance. Explain characteristics of fault tolerance. **[6]**

c) What is disater recovery? Explain disaster recovery methods. **[5]**

OR

Q4) a) What is Recovery time objectives (RTOs) and Recovery point objectives (RPOs) **[6]**

b) Draw & explain General Architecture Hadoop Distributed File System (HDFS) **[6]**

c) List and explain the security issues in cloud **[5]**

P.T.O.

Q3) a) What are the three important components of HDFS? Enlist the difference between GFS and HDFS. **[9]**

b) Explain the basic components DynamoDB? Explain the advantages of DynamoDB **[9]**

OR

P.T.O.

Q4) a) Describe Google file system with neat diagram. **[9]**

b) Explain various Cloud Computing Security Controls with an example?**[9]**

Cloud File Systems

"Cloud file system ek aisa system hota hai jahan aap apne files ko cloud (internet par) store kar sakte ho, aur kahin se bhi access kar sakte ho — jaise Google Drive, Dropbox ya OneDrive, lekin enterprise level pe jaise GFS, HDFS etc."

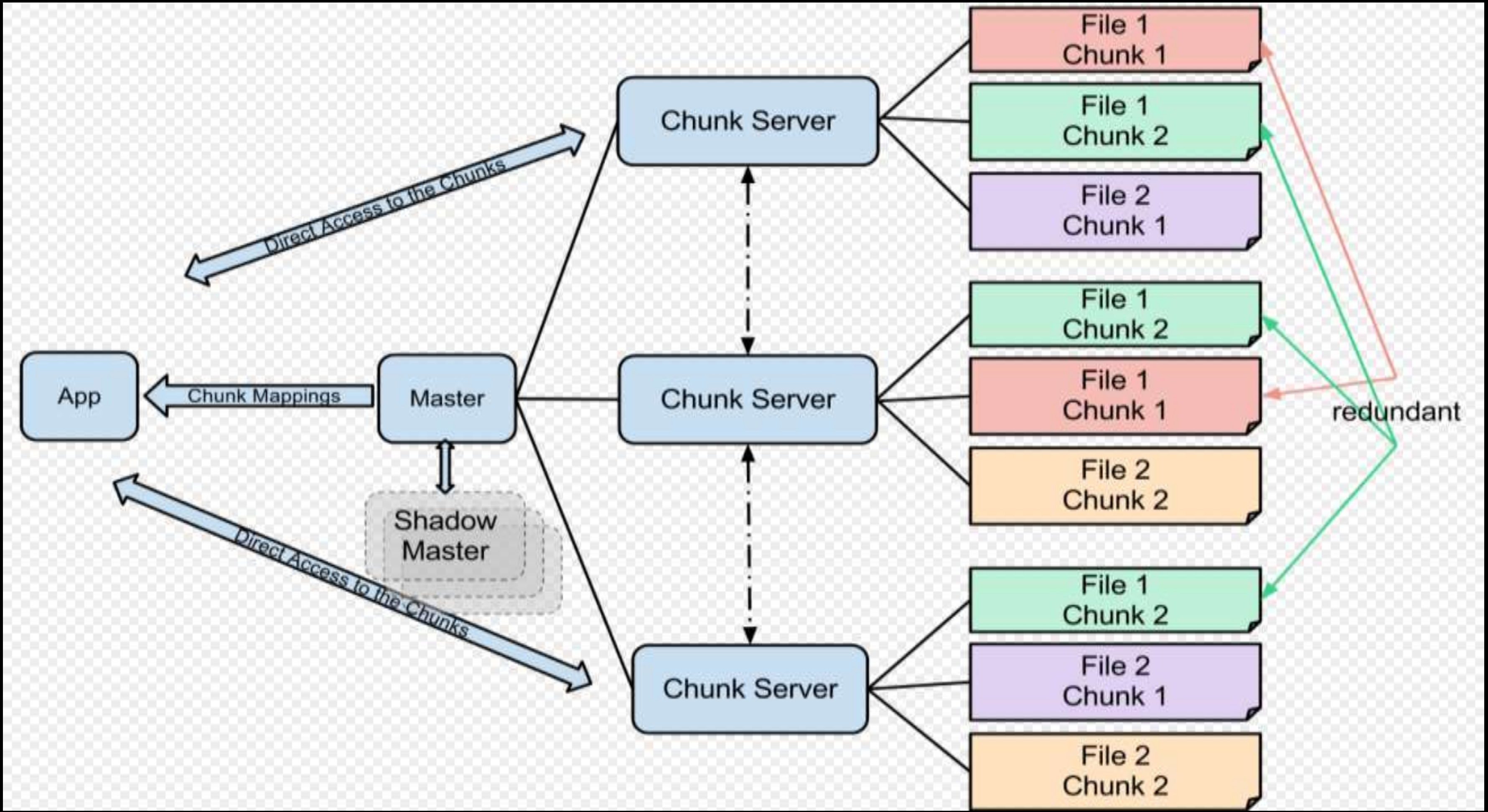
🔍 Features of Cloud File Systems:

- **Scalability** – Bada data store kiya ja sakta hai.
- **Accessibility** – Internet ke zariye kahin se bhi access.
- **Fault Tolerance** – Agar ek server fail ho jaaye to data loss nahi hota.
- **Distributed Storage** – Data ko multiple servers pe store kiya jaata hai.

Introduction to Google file system

□ Introduction to Google File System (GFS)

Google File System (GFS) ek **distributed file system** hai jo specially design kiya gaya tha Google ke large-scale data processing ke liye. Ye system handle karta hai **billion files aur petabytes of data** ko across multiple machines.


App
Chunk Mappings
Master
Shadow Master
Direct Access to the Chunks
Direct Access to the Chunks
Chunk Server
Chunk Server
Chunk Server
File 1 Chunk 1
File 1 Chunk 2
File 2 Chunk 1
File 1 Chunk 2
File 1 Chunk 1
File 2 Chunk 2
File 1 Chunk 2
File 2 Chunk 1
File 2 Chunk 2
redundant

- **App (Application):**
- Ye user ya client ka software hai jo data ko access karta hai.

- **Master:**
- Ye ek main controller hai jo batata hai ki file kaunsa chunk server mein hai.
- Sirf metadata store karta hai (file name, chunk location, etc.).

- **Shadow Master:**
- Backup master, failover ke liye hota hai in case main master down ho jaaye.

- **Chunk Server:**
- Ye real data chunks store karte hain (file ka tukda).
- Har chunk ka multiple copies bana ke alag-alag servers pe rakha jaata hai for **redundancy**.
- **Chunks and Redundancy:**
- File ko chunks mein tod diya jaata hai.
- Har chunk ka 2-3 copies hota hai different chunk servers pe (shown as colored boxes).
- Agar ek server fail ho jaaye to data still available rahe.
- **App ↔ Chunk Server:**
- Master sirf location batata hai, app directly chunk server se data read/write karta hai (fast performance ke liye).

◆ Working of GFS (How It Works):

1. File is divided into **chunks** (small parts).
2. Each chunk is stored in different **Chunk Servers**.
3. The **Master** knows where each chunk is stored.
4. Client (App) asks Master for chunk location.
5. Then client directly accesses the **Chunk Server** to read or write data.
6. Each chunk has multiple copies on different servers (called **redundancy**) so that if one fails, others can be used.

Advantages of Google File System (GFS):

1. **Fault Tolerance (Data loss nahi hota):**
    1. Agar ek server fail ho jaye to data fir bhi safe rehta hai kyunki har chunk ke multiple copies hoti hain (replication).
2. **High Performance (Tez kaam karta hai):**
    1. Client directly chunk server se data leta hai, Master sirf location batata hai. Isse speed fast hoti hai.
3. **Scalability (Asani se grow kar sakta hai):**
    1. System mein naye servers easily add kiye ja sakte hain without any problem.
4. **Efficient for Large Files (Bade files ke liye best):**
    1. GFS specially design kiya gaya hai bade-bade files ko handle karne ke liye jaise videos, logs, big documents, etc.
5. **Automatic Recovery (System khud theek hota hai):**
    1. Agar koi chunk ya server down ho jaye, GFS automatic naya copy bana deta hai.

❌ Disadvantages of Google File System (GFS):

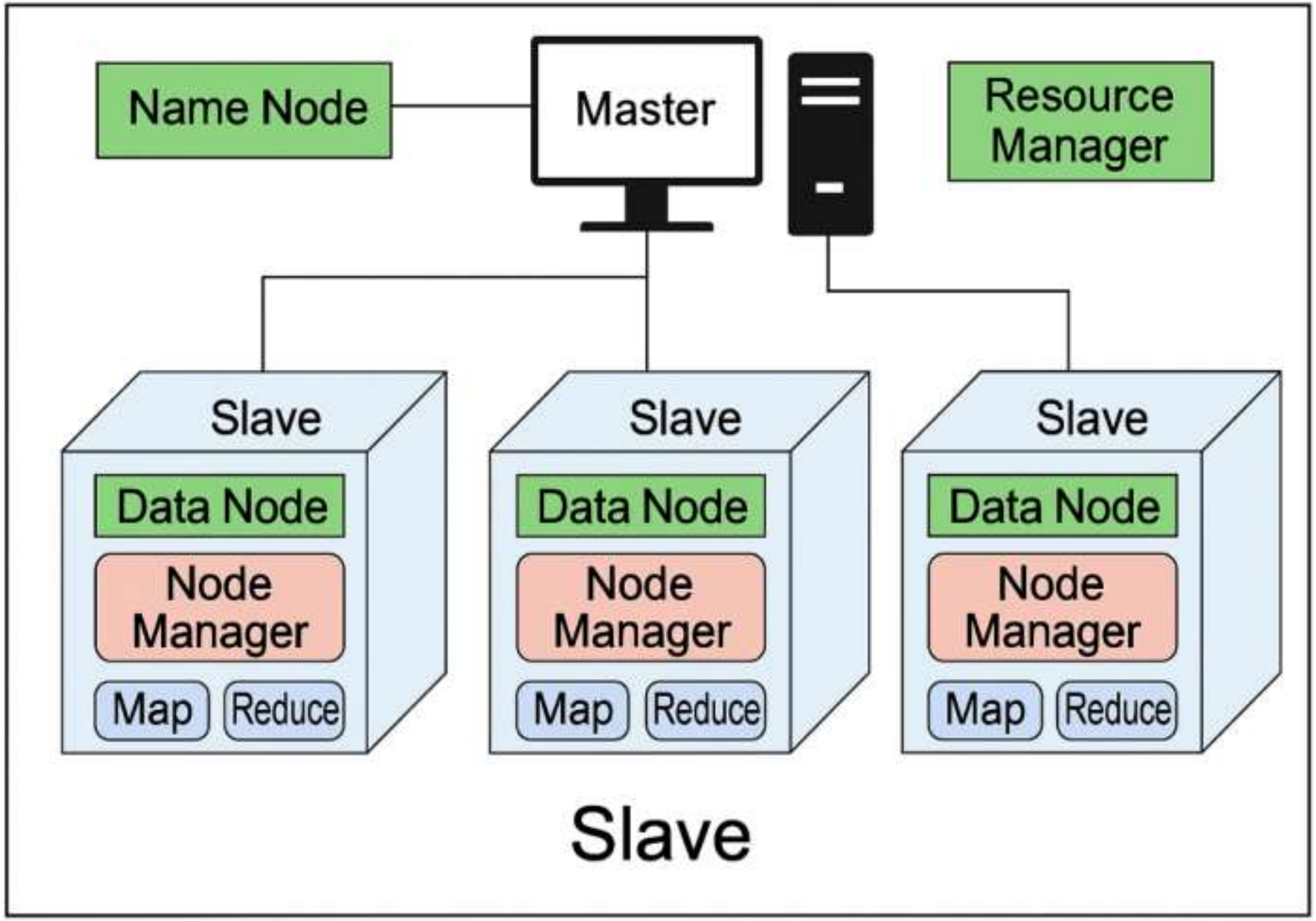
1. **Not for Small Files (Chhoti files ke liye suitable nahi):**
    1. GFS mainly large files ke liye banaya gaya hai. Small files pe use karna inefficient ho sakta hai.
2. **Single Point of Failure (Master fail ho gaya to problem):**
    1. Master server agar down ho gaya to puri file system ruk sakti hai, although Shadow Master help karta hai.
3. **Complex Implementation (Banana aur manage karna tough hai):**
    1. GFS ka design aur working complex hai, normal systems ke comparison mein.
4. **High Network Usage (Zyada network use karta hai):**
    1. Chunk replication aur multiple server communication ki wajah se network load badh jaata hai.

▨ 1. **HDFS (Hadoop Distributed File System)**

◈ 1. HDFS (Hadoop Distributed File System)

- Ye Hadoop ka **main storage system** hai.
- Big files ko **small blocks (default 128 MB)** mein todta hai.
- Har block ki **multiple copies (replication)** banata hai for safety.
- Works on **Master-Slave model**.

Component	Role (Kaam)
NameNode	Master server – metadata store karta hai (block location, file name etc.)
DataNode	Slave servers – actual data blocks store karte hain


Name Node
Master
Resource Manager
Slave
Data Node
Node Manager
Map
Reduce
Slave
Data Node
Node Manager
Map
Reduce
Slave
Data Node
Node Manager
Map
Reduce
Slave

🎁 Hadoop Distributed File System (HDFS) Architecture –

◆ Main Parts of Diagram:

1. Master Node

- Master node ke andar do important parts hote hain:
  - **NameNode** → File system ka master, har file/block ka record rakhta hai.
  - **Resource Manager** → Job scheduling aur cluster resource manage karta hai (YARN ka part).

🎁 2. Slave Nodes (Multiple Machines)

- Ye actual data aur job ko handle karte hain.
- Har Slave node mein hota hai:
  - **DataNode** → Data store karta hai (HDFS ka part).
  - **Node Manager** → Resources ko manage karta hai aur job execute karta hai (YARN ka part).
  - **Map & Reduce Tasks** → Data ko process karte hain (MapReduce engine).

🔄 Working Flow :

1. Jab ek file Hadoop mein daali jaati hai, to **NameNode** us file ko small blocks mein todta hai.
2. Ye blocks **DataNodes** pe store hote hain (multiple copies for safety).
3. Jab data process karna hota hai:
    1. **Resource Manager** decide karta hai ki kaunsa job kahan chalega.
    2. **Node Managers** uss job ko **Map & Reduce** tasks ke through run karte hain.
4. Result finally collect hoke user ko milta hai.

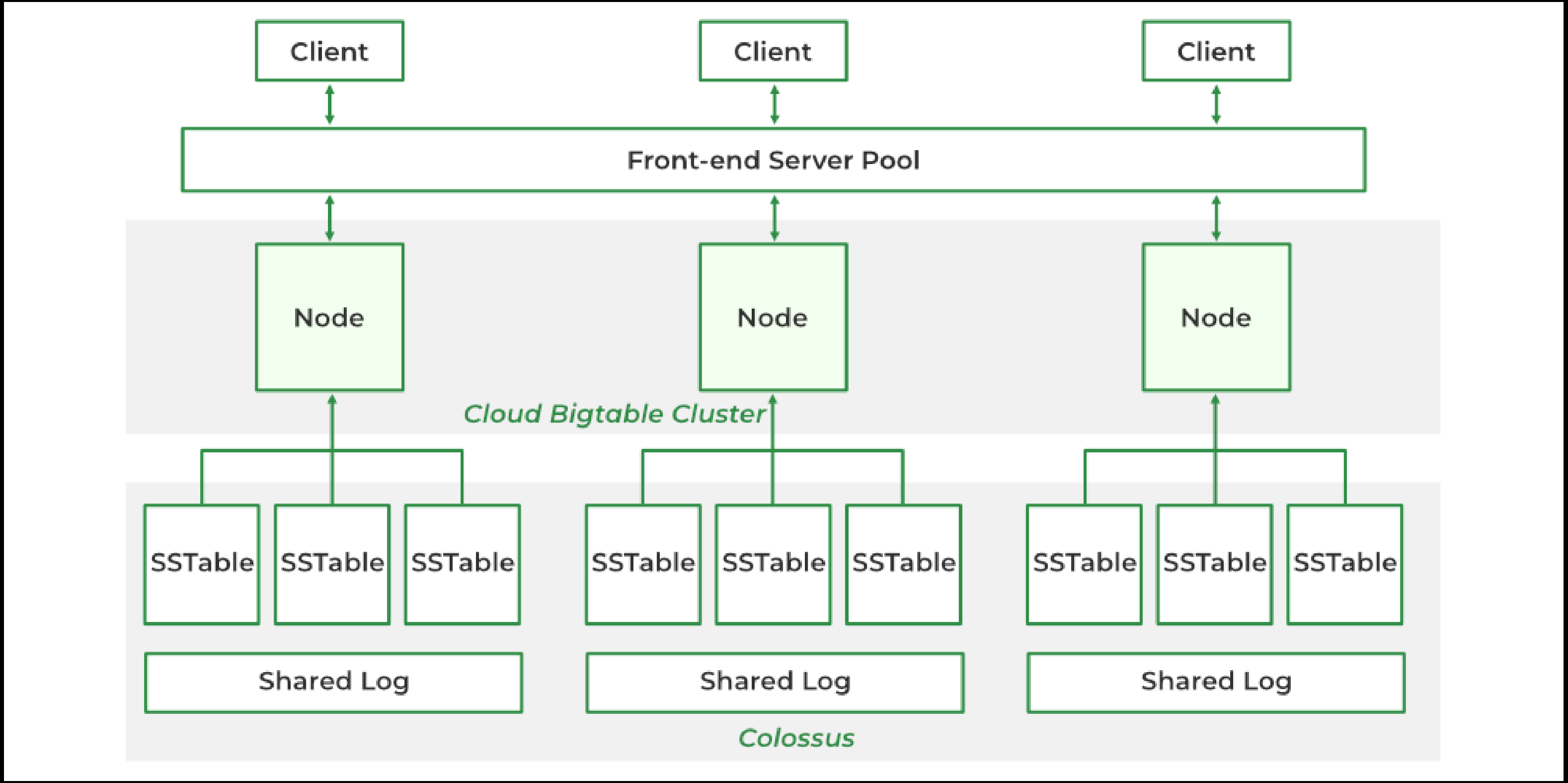
Component	Role (Kaam)
NameNode	Metadata store karta hai (file/block info)
DataNode	Actual data blocks store karta hai
Resource Manager	Job scheduling aur resource handling karta hai
Node Manager	Local node pe job execute karne mein help karta hai
Map/Reduce	Data processing karta hai

Point	HDFS (Hadoop Distributed File System)	GFS (Google File System)
1. Availability	✓ Open Source – Free to use and modify	✕ Proprietary – Only for Google internal use
2. Community Support	✓ Large global community and frequent updates	✕ No external community support
3. Integration	✓ Easily integrates with tools like Hive, Pig, Spark	✕ Limited integration, only within Google
4. Customization	✓ Highly configurable for various use cases	✕ Fixed for Google's internal needs
5. Data Locality	✓ Moves computation to data for faster processing	✕ Not specifically designed for this
6. Ecosystem Use	✓ Widely used in big data platforms across industries	✕ Restricted to Google's internal applications

Component	Description (Hinglish Explanation)
1. NameNode	▯ **Master node** hota hai. Ye pura metadata manage karta hai jaise – file kaha store hai, kis block mein hai, permissions kya hai, etc. 📝 "File kaha rakha gaya hai, ye decide karta hai."
2. DataNode	🎁 **Slave nodes** jo actual data blocks store karte hain. NameNode ke instructions par kaam karte hain. 📝 "Ye real data ko store karte hain."
3. Secondary NameNode	🔄 Ye **backup aur checkpointing** mein madad karta hai. Ye confusion hota hai — ye backup NameNode nahi hai, bas **logs ko periodically merge** karta hai. 📝 "NameNode ki help karta hai crash recovery mein."

Big Table

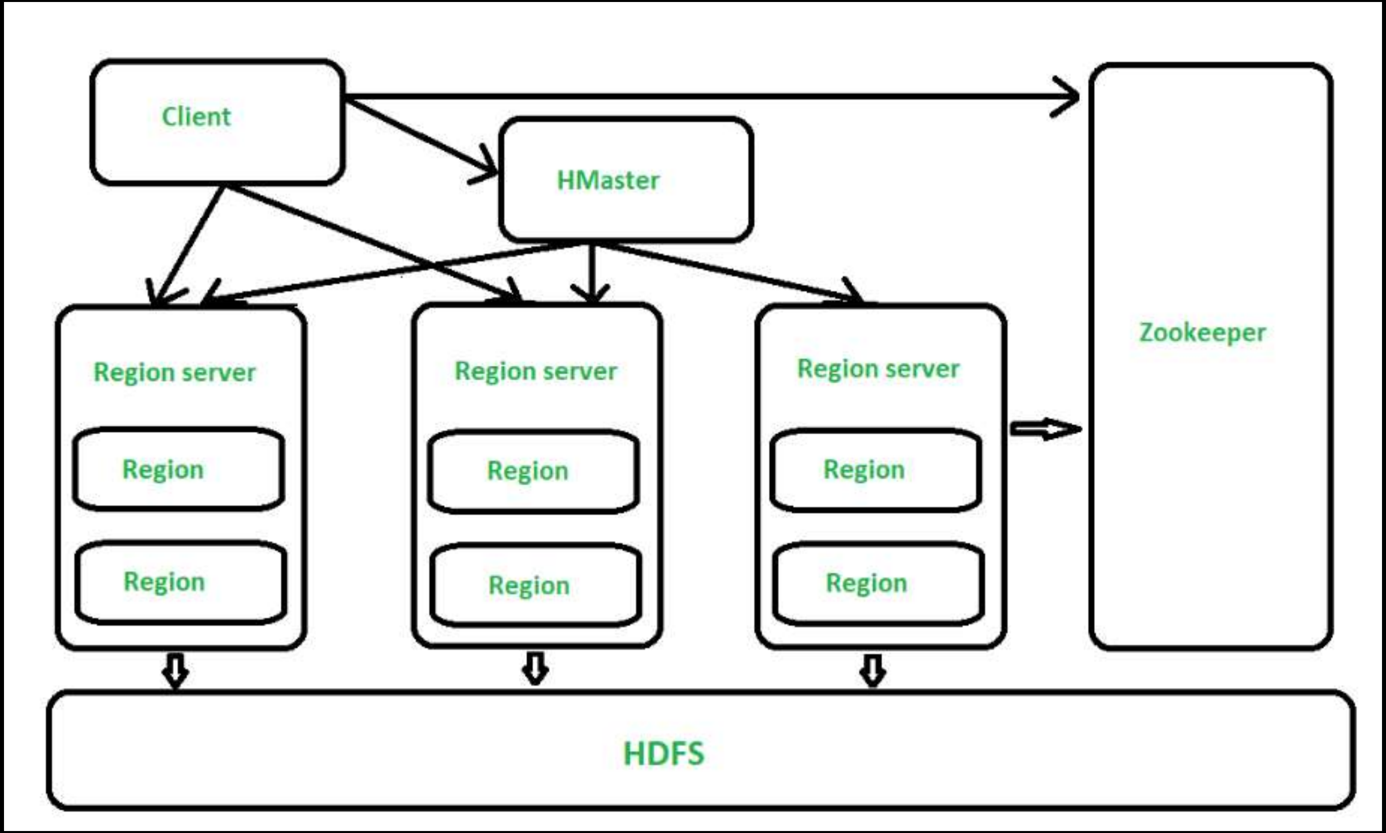
Aspect	Explanation (Hinglish)
Definition	BigTable ek **distributed storage system** hai Google ka, jo **structured data** store karta hai.
Developer	Ye Google ne banaya tha large-scale applications ke liye.
Type	Ye ek **NoSQL wide-column database** hai.
Data Model	Data ko **rows, columns, aur timestamps** mein store karta hai – like a giant spreadsheet.
Use Cases	Web indexing, Google Earth, Google Maps, Gmail, etc. mein use hota hai.
Scalability	Bina performance ke loss ke **terabytes to petabytes** data store kar sakta hai.
High Availability	Multiple servers pe data replicate hota hai – **fault tolerant** system hai.
Access Method	APIs ke through data access hota hai – SQL jaisa nahi hota.
Open Version	Apache HBase, BigTable ka open-source version hai jo Hadoop ke sath kaam karta hai.


Client
Client
Client
Front-end Server Pool
Node
Node
Node
Cloud Bigtable Cluster
SSTable
SSTable
SSTable
SSTable
SSTable
SSTable
SSTable
SSTable
SSTable
Shared Log
Shared Log
Shared Log
Colossus

Component	Explanation (Hinglish)
Client	Client user hota hai jo data ko read/write karta hai.
Front-end Server Pool	Yeh ek load balancer ki tarah kaam karta hai — client request ko right node tak bhejta hai.
Node	Ek node ek server hota hai jo client request handle karta hai aur SSTables access karta hai.
Cloud Bigtable Cluster	Yeh cluster multiple nodes ka group hota hai jo milkar data ko manage karta hai.
SSTable	Yeh file format hai jisme sorted data store hota hai (Sorted String Table).
Shared Log	Sabhi writes logs mein record hote hain — fault tolerance ke liye important hota hai.
Colossus	Google ka scalable storage system hai jo SSTables aur logs ko store karta hai.

HBase

Aspect	Explanation (Hinglish)
Definition	HBase ek **open-source, distributed NoSQL database** hai jo **Hadoop** par kaam karta hai.
Inspired By	Ye Google ke **BigTable** se inspired hai.
Data Model	Data ko **tables, rows, columns aur column families** mein store karta hai.
Scalability	**Horizontally scalable** – large data sets ko easily manage karta hai.
Use Case	Real-time read/write access for big data (like logs, sensor data, social media feeds).
Storage	**HDFS (Hadoop Distributed File System)** ka use karta hai data ko store karne ke liye.
Processing	Hadoop ke tools jaise **MapReduce** ke sath integrate hota hai.
Schema	**Flexible schema** – columns pre-define nahi karne padte, dynamically add kar sakte ho.
Access Method	Java APIs aur REST APIs ke through access hota hai (SQL jaisa nahi hota).
Real Example	Facebook’s messaging system, and time-series data storage.


Client
HMaster
Zookeeper
Region server
Region
Region
Region server
Region
Region
Region server
Region
Region
HDFS

Component	**Explanation (Hinglish)**
Client	User ya application jo HBase se data **read/write** karta hai. Directly Region Server se connect hota hai.
HMaster	HBase ka **master node** – Region servers ko manage karta hai, region assign karta hai, balancing karta hai.
Region Server	Multiple hote hain – yeh **actual data** ko handle karte hain. Har server mein kai **regions** hote hain.
Region	Table ka horizontal partition (subset of data). Ek region ek ya zyada column families ko store karta hai.
HDFS	HBase ka **storage layer** – data ko yeh store karta hai in file blocks. High fault tolerance deta hai.
ZooKeeper	Coordination service – **client requests ko route** karta hai, master fail ho toh naye master ko activate karta hai.

Feature	Bigtable (Google)	HBase (Apache)
Developer	Google ne banaya hai (part of Google Cloud Platform)	Apache open-source community ne banaya hai
Storage System	Uses **Colossus** (Google ka distributed storage system)	Uses **HDFS** (Hadoop Distributed File System)
Managed/Unmanaged	Fully managed service by Google (koe tension nahi)	Developer ko khud manage karna padta hai (setup, maintenance, tuning)
Integration	Tight integration with **Google Cloud tools** (BigQuery, Dataflow, etc.)	Integrates with **Hadoop ecosystem** (Hive, Pig, MapReduce, etc.)
Scalability	Highly scalable, auto-sharding support	Manually scalable, tuning required
Latency	Low latency reads/writes via gRPC	Higher latency in comparison, especially with heavy load
Security	Google Cloud IAM and VPC for fine-grained security	Security needs to be configured manually via Kerberos, ACL, etc.
Ease of Use	Easy to use (no setup, just APIs and console)	Steep learning curve (needs setup, Zookeeper, HDFS, tuning)
Use Case	Best for cloud-native apps, IoT, analytics	Best for big data processing pipelines with Hadoop

HBase

Component / Feature	**Explanation (Hinglish)**
Service Name	**Amazon DynamoDB** – AWS ka fully managed NoSQL database service
Data Model	**Key-Value & Document Store** – Har item ek unique key se access hota hai
Partitioning	Uses **consistent hashing** – data ko multiple partitions (nodes) me auto-distribute karta
Replication	Data **multi-AZ (Availability Zones)** me replicate hota hai for high availability
Consistency Models	Supports **eventual consistency** & **strong consistency** (on demand)
Write & Read Throughput	**Provisioned** ya **On-demand capacity mode** – dono options available
Durability	Uses **quorum-based replication** (majority nodes) for durability and fault tolerance
Failure Handling	Uses **hinted handoff** & **vector clocks** for handling temporary node failures
Use Cases	IoT, real-time analytics, gaming, session data, shopping cart
Integration	Tight integration with AWS tools (Lambda, API Gateway, IAM, CloudWatch, etc.)

b) Explain the basic components DynamoDB? Explain the advantages of DynamoDB **[9]**

b) Explain the features and advantages of Dynamo DB. Explain how it is different than RDBMS. [9]

🔢 No.	⍰ Component Name	🔍 Explanation (Hinglish)
1️⃣	**Table**	Main storage unit jisme data store hota hai. Each table has a name & primary key.
2️⃣	**Item**	Table ka ek row. Har item unique hota hai apni primary key ke basis par.
3️⃣	**Attribute**	Item ke andar ke fields jaise "name", "age", "email". Similar to columns in RDBMS.
4️⃣	**Primary Key**	Unique ID jisse item identify hota hai. Do types hote hain:
		- **Partition Key** (Simple key)
		- **Partition + Sort Key** (Composite key)
5️⃣	**Secondary Index**	Alternate way to query data. Do types hote hain:
		- **GSI** (Global Secondary Index)
		- **LSI** (Local Secondary Index)
6️⃣	**Provisioned/On-Demand Capacity**	Batata hai kitni read/write capacity chahiye. Auto-scaling bhi available hai.
7️⃣	**DynamoDB Streams**	Real-time changes (insert/update/delete) ko track karta hai for further processing.
8️⃣	**Partitions**	Backend system jo data ko divide karta hai for scalability & performance.
9️⃣	**Global Tables**	Multi-region replication ke liye use hota hai. Distributed apps mein kaam aata hai.

◆ Features of DynamoDB:

1.Fully Managed NoSQL Database
AWS khud manage karta hai infrastructure, backups, scaling etc.
2.High Performance & Low Latency
Millisecond response time milta hai even at scale.
3.Scalable
Automatically scale hoti hai read/write throughput ke according.
4.Flexible Data Model
Key-value aur document data models support karta hai.
5.Serverless Architecture
Server setup/maintenance ki zarurat nahi hoti.
6.Built-in Security
Encryption at rest & in transit, IAM ke through access control.
7.Global Tables
Multi-region, multi-master replication support karta hai.
8.Backup & Restore
On-demand aur continuous backup support karta hai.

📌 Feature	✓ Advantage
Serverless	AWS manage karta hai infra, scaling, backups etc.
High Scalability	Auto-scaling support deta hai massive traffic ke liye.
Low Latency	Millisecond response time ensure karta hai.
Flexible Schema	Har item alag attributes rakh sakta hai.
Built-in Security	IAM roles, encryption, and fine-grained access control available.
Global Tables	Multi-region data replication support karta hai.
Backup & Restore	On-demand and continuous backup options available.
Integration with AWS Ecosystem	Lambda, API Gateway, S3, etc. ke sath easy integration.

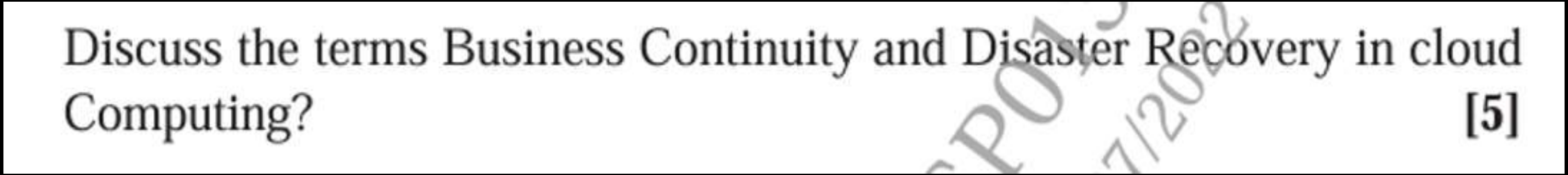
Feature	DynamoDB (NoSQL)	RDBMS (Relational DB)
Data Model	Key-Value / Document	Tables with Rows and Columns
Schema	Schema-less (Flexible)	Fixed Schema (Structured)
Scalability	Horizontally Scalable (auto-scaling)	Vertically Scalable (limited)
Query Language	NoSQL API (Proprietary SDKs)	SQL (Structured Query Language)
Transactions Support	Limited but available	Full ACID Transactions
Performance	High at scale (low latency)	Can degrade at high scale
Use Cases	IoT, Gaming, Real-time Analytics, Mobile Apps	Banking, ERP, Traditional Business Apps
Maintenance	Serverless (AWS manages it)	Requires DBA for tuning and backups
Joins & Relationships	Not Supported (manually handled)	Fully Supported

Google **Datastore** (by Google Cloud)

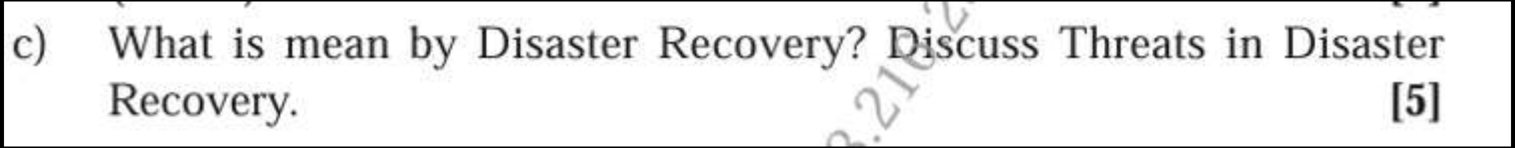
🔢 No.	Feature	Explanation (Hinglish)
1️⃣	**Data Model**	Entity-Property model (like rows with key-value pairs)
2️⃣	**Based On**	Built on **BigTable** (Google’s distributed storage)
3️⃣	**Indexing**	Automatic indexing; supports composite indexes
4️⃣	**Query Language**	GQL (Google Query Language) - SQL jaisa syntax
5️⃣	**Transactions**	Supports **ACID transactions** (but limited to entity groups)
6️⃣	**Consistency**	Strong consistency **within entity group**, eventual otherwise
7️⃣	**Scalability**	High scalability using auto-sharding
8️⃣	**Use Cases**	Complex web apps, real-time apps (with transactions and queries)

SimpleDB

🔢 No.	Feature	Explanation (Hinglish)
1️⃣	**Data Model**	Attribute-value pairs (schema-less, flexible)
2️⃣	**Based On**	Amazon Web Services (AWS) NoSQL offering
3️⃣	**Indexing**	Automatically indexes all attributes
4️⃣	**Query Language**	SimpleDB Query Language (not SQL)
5️⃣	**Transactions**	No full ACID transactions, only limited consistency
6️⃣	**Consistency**	**Eventually consistent** reads and writes
7️⃣	**Scalability**	Horizontally scalable with auto-partitioning
8️⃣	**Use Cases**	Simple applications like logging, metadata store, IoT data



Discuss the terms Business Continuity and Disaster Recovery in cloud Computing? **[5]**

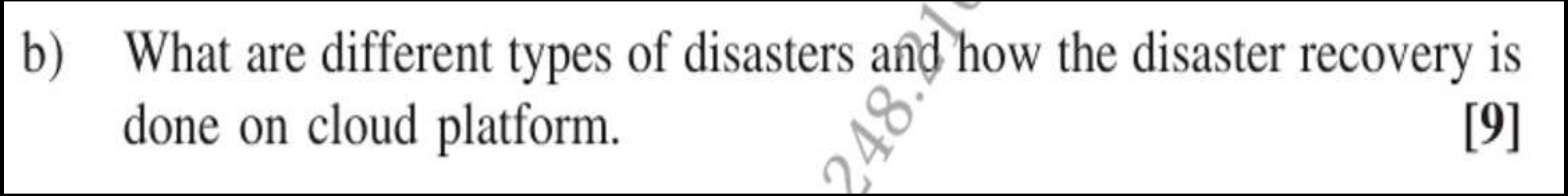


c) What is mean by Disaster Recovery? Discuss Threats in Disaster Recovery. **[5]**

Business Continuity ka matlab hota hai ki agar koi problem ho jaaye, jaise system down ho gaya ya server fail ho gaya, fir bhi business ka kaam rukna nahi chahiye. Cloud computing mein data aur applications har jagah se access kiye jaa sakte hain, isliye agar ek system band ho jaye to doosre system se kaam continue kiya jaa sakta hai.

Disaster Recovery ka matlab hota hai ki jab koi bada problem ho jaata hai jaise hacking, data loss, ya natural disaster, tab system ko wapas kaise jaldi normal kiya jaaye. Cloud platform backup, replication aur automatic recovery features deta hai, jisse system aur data ko easily restore kiya jaa sakta hai.

Cloud Features	HiEnglish Explanation
Data Backup	Cloud automatically data ka backup rakhta hai, isliye kuch delete ho jaye to wapas mil jaata hai.
Replication	Data ek se zyada location par store hota hai, isliye agar ek fail ho jaye to dusra use hota hai.
Auto Recovery	System automatically crash ke baad restart ho jaata hai without manual work.
Anywhere Access	Kaam kahin se bhi kiya jaa sakta hai, isliye office band ho to bhi work chalu rehta hai.



b) What are different types of disasters and how the disaster recovery is done on cloud platform. [9]

Type (प्रकार)	Explanation (विवरण)
Natural Disaster	जैसे earthquake, flood, fire – ये physical जगह को damage करते हैं जैसे data centers।
Hardware Failure	Server, hard disk, router वगैरह का खराब हो जाना।
Software Failure	App या OS में bug, crash या update की वजह से issue आ जाना।
Human Error	गलती से data delete कर देना या गलत configuration कर देना।
Cyber Attack	जैसे hacking, ransomware, DDoS attack – ये systems को compromise करते हैं।
Power Failure	बिजली चले जाना, जिससे systems बंद हो जाते हैं।
Network Failure	Internet या connectivity की problem – cloud access नहीं हो पाता।

Method (तरीका)	**Explanation (विवरण)**
Backup & Replication	Data को automatically अलग-अलग जगह copy करके रखा जाता है।
Geo-Redundancy	Multiple locations (regions/zones) use होते हैं ताकि एक जगह fail हो जाए तो दूसरा चले।
Auto Scaling & Load Balancing	Traffic को smart तरीके से manage किया जाता है during heavy load or failure.
DRaaS (Disaster Recovery as a Service)	Cloud provider खुद recovery services देता है – fast और reliable way से।
Snapshots & Versioning	System का पुराना version restore करने के लिए snapshot save रहते हैं।
Monitoring & Alerts	Real-time में issue detect होता है और alert मिलते हैं action लेने के लिए।
Testing & Drill	Time-time पर recovery plan को test किया जाता है ताकि सब ready रहे।

c) What are the General Security Advantages of Cloud-Based Solutions?[5]

Security Advantage	Explanation (Hinglish)
1. Data Redundancy & Backup	Cloud automatically **duplicate karta hai data** multiple locations mein. Agar ek server fail ho gaya, to data safe rehta hai.
2. Centralized Security Management	Saari **security settings ek jagah se manage hoti hain**, jisse errors kam hote hain aur policies easily apply ho jaati hain.
3. Advanced Encryption	Data ko **encrypt kiya jaata hai** (rest aur transit dono mein), taaki unauthorized log use access na kar sakein.
4. Continuous Monitoring	Cloud providers **24x7 monitoring aur threat detection** dete hain. Agar kuch suspicious dikhta hai, to turant alert milta hai.
5. Scalable Security Tools	Jaise-jaise traffic ya usage badhta hai, **security tools bhi automatically scale ho jaate hain**.
6. Access Control & Identity Mgmt	Features jaise **MFA (OTP), RBAC (roles), aur SSO (ek login sab jagah)** se unauthorized access prevent hota hai.
7. Compliance Support	Cloud providers **international rules aur standards** follow karte hain (jaise GDPR, ISO), jisse aapka business compliant rehta hai.
8. Auto Patch & Updates	Software ka **patching aur updates automatic hote hain**, jisse security loopholes close ho jaate hain.
9. AI/ML-Powered Security	Cloud **AI/ML use karta hai** taaki unusual activity detect ho sake (jaise phishing, hacking attempts).
10. DDoS Protection	Built-in **DDoS attack protection** hoti hai taaki traffic overload se system down na ho.

a) Write a short note on [6]

i) How to Approach Business Continuity

ii) Architect for Failure

Business Continuity ka matlab hota hai kisi bhi unexpected disruption (jaise disaster, cyberattack, power failure) ke baad business operations ko **jaldi se wapas normal** karna.

📊 Table 1: Steps to Approach Business Continuity

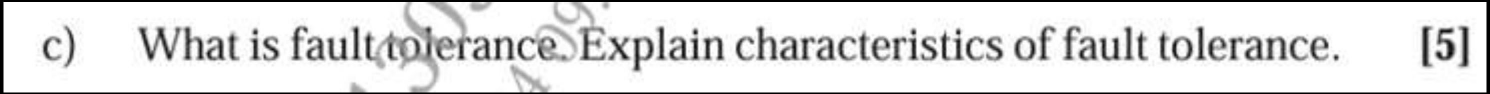
Step	💬 HiEnglish Explanation
1. **Risk Assessment**	Sabse pehle possible threats aur unke impact ko identify karo (jaise natural disaster, data loss).
2. **Business Impact Analysis**	Identify karo kaunse systems critical hain aur unke down hone se kitna loss ho sakta hai.
3. **BCP Plan Creation**	Ek detailed plan banao jisme bataya ho ki har system ka backup aur alternate method kya hai.
4. **Training & Awareness**	Employees ko train karo ki disaster ke waqt unko kya steps lene hain.
5. **Testing & Updating**	Plan ko regularly test karo aur naye risks ke according update karte raho.

ii) Architect for Failure

"**Architect for Failure**" ka matlab hai system ko aise design karna ki agar koi part fail bhi ho jaaye, to poora system **down na ho**.

📊 Table 2: Key Principles to Architect for Failure

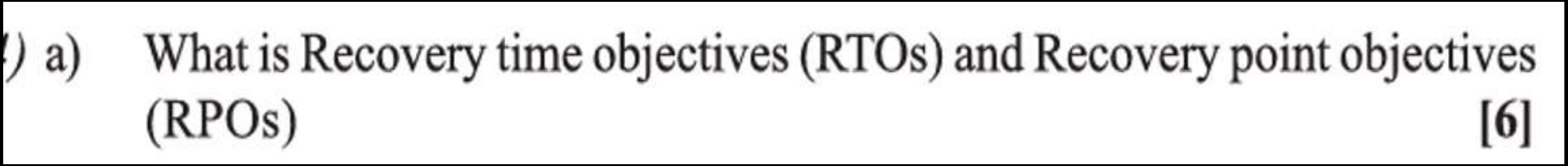
⚙️ Principle	💬 HiEnglish Explanation
1. **Redundancy**	Ek system ka backup ya duplicate rakhna (e.g. multiple servers).
2. **Failover Mechanism**	Agar ek server fail ho jaye to doosra automatically kaam shuru kare.
3. **Decoupling Components**	Systems ko loosely connect karo taaki ek part fail ho to baaki pe impact na pade.
4. **Backup & Recovery**	Regular data backups aur quick restore plans hone chahiye.
5. **Auto-scaling & Load Balancing**	Traffic ka load distribute karo taaki overload se failure na ho.
6. **Monitoring & Alerts**	System ka real-time monitoring karo aur issue detect hote hi alert mile.


c) What is fault tolerance. Explain characteristics of fault tolerance. [5]

💡 **What is Fault Tolerance? (Fault Tolerance kya hota hai?)**
Fault Tolerance ek system ki **ability hoti hai ki wo proper kaam karta rahe**, even if kuch parts fail ho jaayein.
⚙️ **Goal**: System kabhi down na ho, chahe koi error, hardware failure, ya bug ho jaaye.
🏢 Mostly used in: Cloud computing, servers, banking systems, healthcare IT, etc.

📊 Characteristics of Fault Tolerant Systems

🔰 Characteristic	💬 HiEnglish Explanation
1. Redundancy	System ke important parts ke **multiple copies** hote hain (backup), jo failure pe active ho jaate hain.
2. Failover Mechanism	Agar ek part fail ho jaye, to **automatically dusra part activate** ho jata hai without downtime.
3. Error Detection	System **real-time monitoring** karta hai aur fault detect karte hi alert ya action leta hai.
4. Recovery Mechanism	Fault ke baad system **automatically repair ya recover** karne ki koshish karta hai.
5. Isolation of Faults	Fault ek component tak **limit** hota hai, baaki system normal kaam karta rehta hai.
6. Consistency Maintenance	Chahe fault ho ya failover, system **data consistency aur accuracy** banaye rakhta hai.
7. Self-Testing	System apne aapko **regularly test karta hai** to ensure sab sahi chal raha hai.



a) What is Recovery time objectives (RTOs) and Recovery point objectives (RPOs) [6]

RTO (Recovery Time Objective)

🔰 **Aspect**	💬 **HiEnglish Explanation**
Full Form	Recovery Time Objective
Meaning	System ko **kitni der ke andar wapas chalu karna hai**
Focus	□ **Downtime duration** ke upar focus karta hai
Purpose	Service ko **quickly restore** karne ka target set karta hai
Measured In	Hours, Minutes, or Seconds
Example	Agar RTO = 2 hours hai, to system ko 2 ghante ke andar recover karna hoga

RPO (Recovery Point Objective)

🔰 **Aspect**	💬 **HiEnglish Explanation**
Full Form	Recovery Point Objective
Meaning	Failure hone pe **kitna data loss acceptable** hai
Focus	💾 **Data loss** ke time ke upar focus karta hai
Purpose	Backup plan ka **frequency** decide karne mein help karta hai
Measured In	Hours, Minutes, or Seconds
Example	Agar RPO = 30 minutes hai, to max 30 minutes ka data loss allow hai

c) List and explain the security issues in cloud

🔒 Security Issue	💬 HiEnglish Explanation
1. Data Breaches	Jab hackers aapka data unauthorized access karke chura lete hain.
2. Data Loss	Data accidentally delete ho jaye ya corrupt ho jaye without backup.
3. Account Hijacking	Unauthorized users aapke login credentials chura kar account ka misuse karte hain.
4. Insecure APIs	Cloud services APIs agar properly secure nahi hain to attackers unka misuse kar sakte hain.
5. DoS (Denial of Service)	Attacker system ko itna traffic bhejta hai ki wo crash ho jaye, users access nahi kar paate.
6. Insider Threats	Organization ke khud ke employees hi kabhi kabhi malicious activity karte hain.
7. Lack of Data Control	Cloud mein data control third-party ke paas hota hai, so aapko direct access nahi hota.
8. Misconfigured Storage	Galat cloud settings ki wajah se data public ho sakta hai (e.g. AWS S3 buckets open ho jana).
9. Weak Access Management	Jab proper password policies, MFA (Multi-factor authentication) use nahi hoti.
10. Compliance Issues	Cloud provider agar legal ya industry standards (like GDPR, HIPAA) follow nahi karta to trouble ho sakta hai.

b) Explain various Cloud Computing Security Controls with an example?**[9]**

🔐 What are Security Controls in Cloud Computing?

Security controls are **methods, tools, and practices** used to **protect cloud data, applications, and infrastructure** from threats.
In short, yeh controls cloud ko **secure aur reliable** banate hain.

🔰 Type of Security Control	⍰ HiEnglish Explanation	📃 Example
1. Preventive Controls	Yeh controls **attack hone se pehle hi rokne** ke liye use hote hain.	🔐 Encryption, Firewalls, MFA (Multi-Factor Authentication)
2. Detective Controls	Yeh controls help karte hain **threat ya breach ko detect** karne mein.	📊 Intrusion Detection System (IDS), Activity Logs, Monitoring Tools
3. Corrective Controls	Attack ke baad system ko **restore** karne ya issue fix karne mein help karte hain.	🔄 Backup restore, Patch management, Recovery scripts
4. Deterrent Controls	Attackers ko **darr ya warning** dene ke liye use hote hain.	⚠▯ Legal warning banners, Surveillance cameras, Security policies
5. Compensating Controls	Jab original control possible nahi hota, tab **alternative control** lagaya jata hai.	🔁 Manual monitoring instead of automatic logs

□□ Hadoop MapReduce Paradigm

🔍 **MapReduce kya hota hai?**

MapReduce ek programming model hai jo **large data** ko **parallel** process karta hai. Ye do main steps mein kaam karta hai:

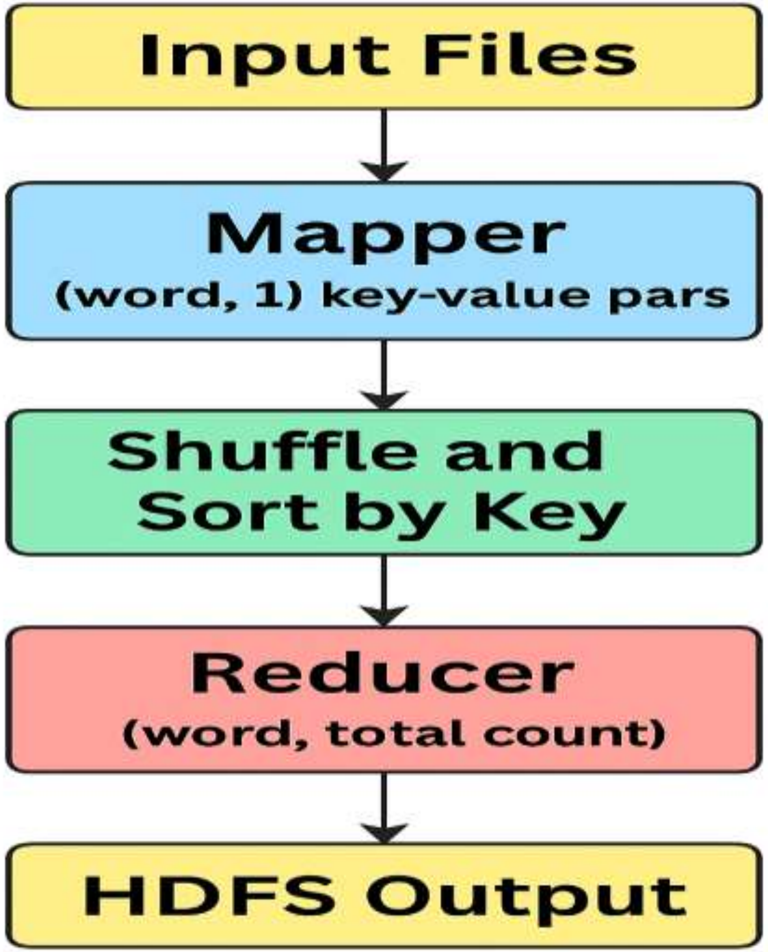
1. **Map Phase** – Data ko todta hai (chunk banaata hai) aur key-value pairs create karta hai.
2. **Reduce Phase** – Same key wale values ko add ya summarize karta hai.

📚 Word Count Example (Simple Explanation)

📝 **Goal:**
Har word kitni baar aaya hai, ye count karna.

🔄 Steps: Word Count using MapReduce

🔢 Step	🔧 Phase	📖 Explanation (Hinglish)
1	**Input**	Input file read hoti hai line by line.
2	**Mapper**	Har line ke words ko tod kar (word, 1) pair banata hai.
3	**Shuffle/Sort**	Same words ko ek group mein le aata hai.
4	**Reducer**	Sabhi 1s ko add karke word ka total count nikalta hai.
5	**Output**	Result file banata hai jisme (word, count) hote hain.


Input Files
Mapper
(word, 1) key-value pars
Shuffle and
Sort by Key
Reducer
(word, total count)
HDFS Output

🔤 Input Example:

```
Line 1: hadoop is open source
Line 2: hadoop is powerful
```

🔀 Shuffle & Sort Phase:

```
(hadoop, [1,1])
(is, [1,1])
(open, [1])
(source, [1])
(powerful, [1])
```

+ Reduce Phase Output:

```
(hadoop, 2)
(is, 2)
(open, 1)
(source, 1)
(powerful, 1)
```

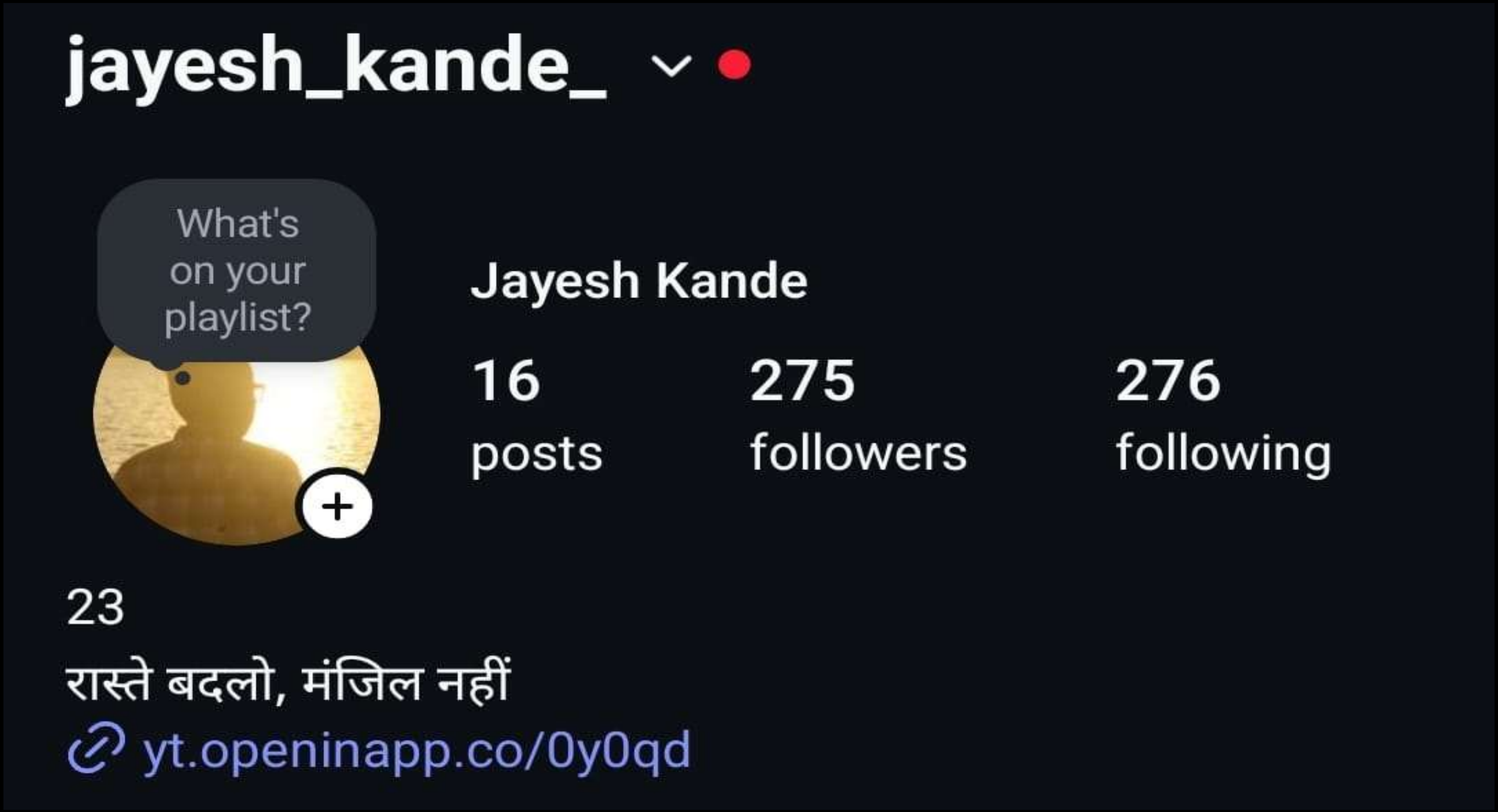
MapReduce Tasks List

🔢 No.	📌 Task Name	▯ Explanation (Hinglish)
1	**Input Splitting**	Input file ko chhote-chhote splits (parts) me divide kiya jaata hai.
2	**Mapping**	Har input split par mapper kaam karta hai aur (key, value) pairs banata hai.
3	**Shuffling**	Mapper se nikle data ko sort aur group kiya jaata hai key ke according.
4	**Sorting**	Same keys ke data ko ek saath laane ke liye sort kiya jaata hai.
5	**Reducing**	Har key ke liye values ko summarize ya aggregate kiya jaata hai.
6	**Output Writing**	Final result ko HDFS (ya kisi aur output format) me store kiya jaata hai.

📃 Real Life Analogy:

Socho ek school me students ke test scores hai:

•**Map**: Har student ka naam aur score nikala gaya → (name, score)
•**Shuffle/Sort**: Sabhi same students ke scores ek saath kiye gaye
•**Reduce**: Har student ke total score ya average nikala gaya
•**Output**: Final result ban gaya (like result sheet)



jayesh_kande_

What's on your playlist?

Jayesh Kande

16 posts

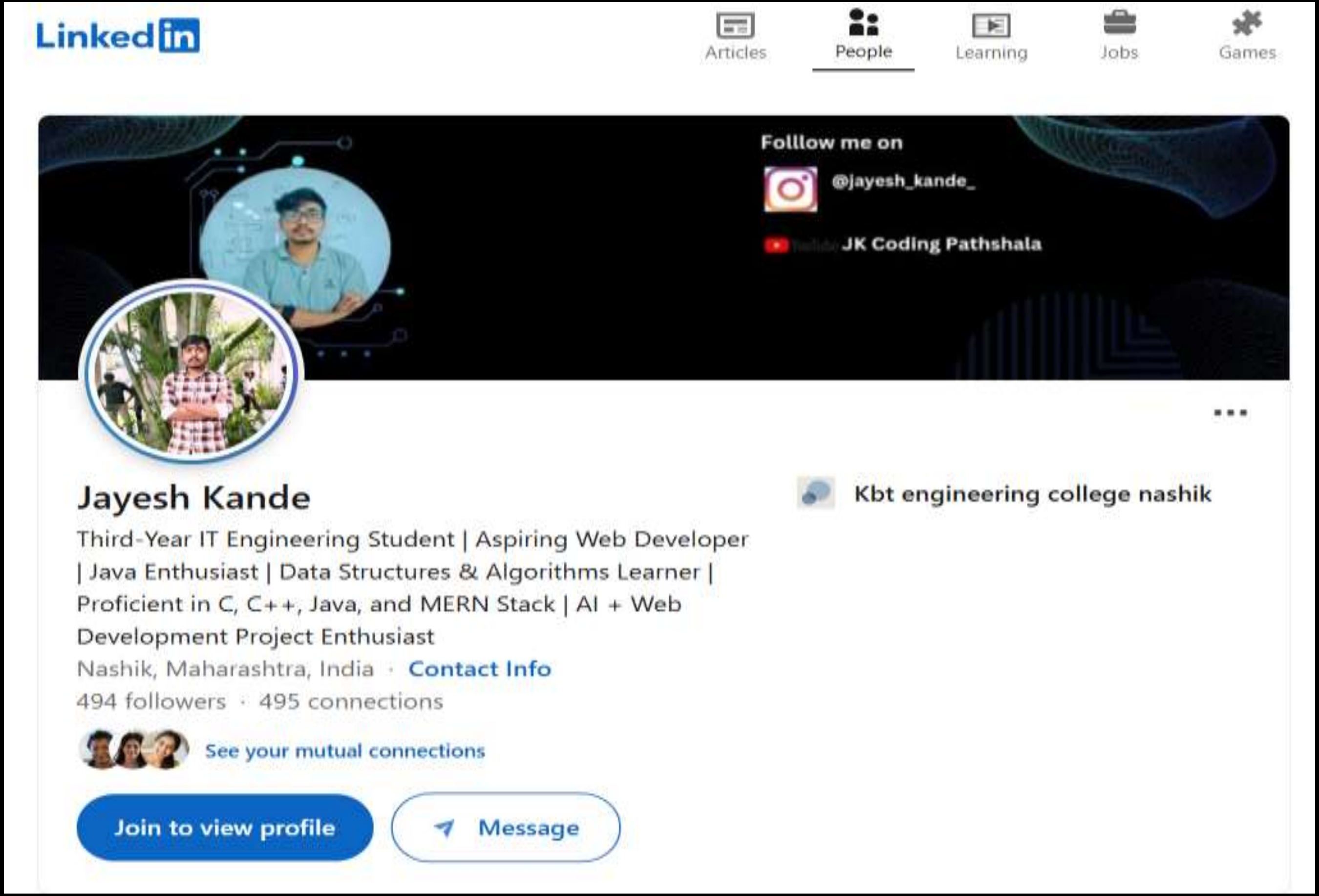
275 followers

276 following

23

रास्ते बदलो, मंजिल नहीं

yt.openinapp.co/0y0qd


Articles People Learning Jobs Games

Folllow me on

@jayesh_kande_

JK Coding Pathshala

Jayesh Kande

Third-Year IT Engineering Student | Aspiring Web Developer | Java Enthusiast | Data Structures & Algorithms Learner | Proficient in C, C++, Java, and MERN Stack | AI + Web Development Project Enthusiast

Nashik, Maharashtra, India · Contact Info

494 followers · 495 connections

See your mutual connections

Join to view profile   Message

Kbt engineering college nashik

✨ Thank You for Watching! ✨

➔📱Follow us on Instagram:**@jayesh_kande_**

🔗 Connect with us on LinkedIn:[**Jayesh Kande**]