Introduction to NOSQL

NO SQL

1. open source, non-relational, distributed data base
2. It can handle all 3 forms of data
3. does not support ACID properties
4. no proper fixed schema for the table

Type of NO SQL databases

1. key value (key value pair)
   Eg: Fname: Dhan Lname: Nayak
2. schemaless - Document form of data
   i) column - data is stored in column
   ii) graph - consists of node & links

```
{
  btitle: ____
  bauthor: ____
  publisher: ____
}
```

Need of NOSQL

1. relational DB have monolithic but the NOSQL architecture have scale out architecture
2. it can store large amount of structured, unstructured & semi structured data
3. Dynamic schema - (insert/update/ delete are easy changes to appl in real-time is easy)
4. Auto Sharding
   The availabity data is distributed into diffrent arbitrary servers so that there is no downtime if ~~any~~ server crashes
5. Replication
   because of the replication of data into all the arbitrary servers that are present downtime will not be present

Advantages

1. can easily scale up & down.
   - cluster scale
   - performance scale
   - data scale
2. do not require proper schema
3. cheap & easy to implement
4. reduces the consistency of data
5. Data can be replicated & partitioned
   - Sharding
   - replication

Disadvantages

1. There are no joins
   ie inner join, outer join etc & its is group by clause
2. NO ACID properties.
3. integration of the application with other application which support SQL is not possible

uses of NoSql in industry / Application

NEW SQL

- accepts ACID, OLAP features

SQL	NO SQL	NEW SQL
1) ACID properties	yes	NO	yes
2) OLAP/OLTP properties	yes	NO	yes
3) Schema rigidity	yes	NO	Maybe
4) Dataformat flexibility	NO	yes	Maybe
5) Scalability	vertical	Horizontal	Scalable
6) Distributed computing	yes	yes	yes
7) community support	huge.	growing	slowly growing

Mongo db

- cross platform
- Open source
- distributed
- Multinational
- Document Oriented data store
- NO SQL

[illegible]

using JSON : Java script object notation

open standard

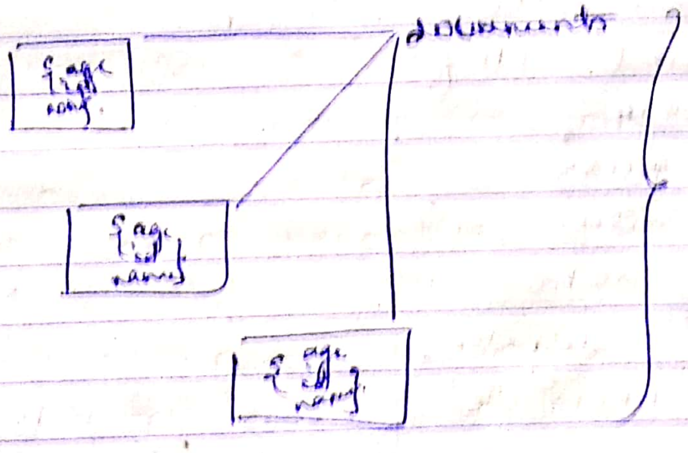
complex structure

```
{
    name : abc
    class : ___
    contactno : ___
}
{ ___
}
```

Syntax of how to store data

② Creating & querying unique key

- Here there are collections where data is stored
- One mongo db server can have multiple data bases.
- Accessing of a collection is done through fields
- Also multiple documents can be available



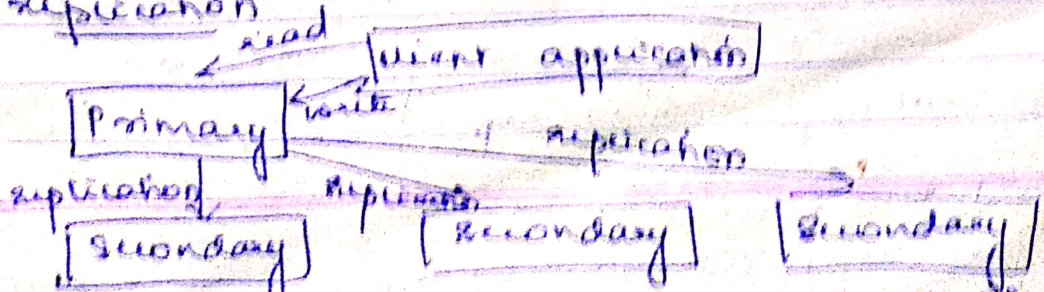
collection of documents

③ Support for dynamic queries

④ Support to store binary data using GridFS

- The meta data stored in form of logs
- The data is divided to chunks & stored in file

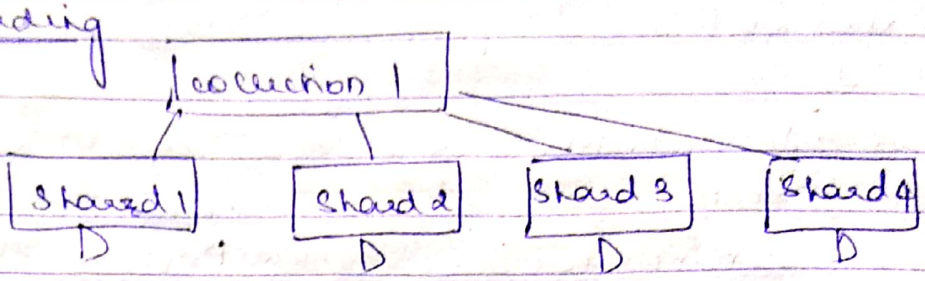
⑤ Replication



When there are multiple copies of data the ...

must retrieve the updated data. The data should be updated in all copies.

⑥ Sharding



Large data set is divided into multiple shards & stores

Each shard will have some data to be stored and processed. Therefore easy to process & main

⑦ updating data in place

More the storage less the performance.

Apache Cassandra

features - highly scalable.
- distributed interface

① peer to peer network
→ It has multiple peers.
⇒ Mutable datastructure
→ easier to access data

② gossip & failure detection
It will come to know the failures in network

③ Partitioner

④ replication factor
no. of copies of rows that is available

simple strategy — network Terminology Strategy
|
allocate the additional note to existing network

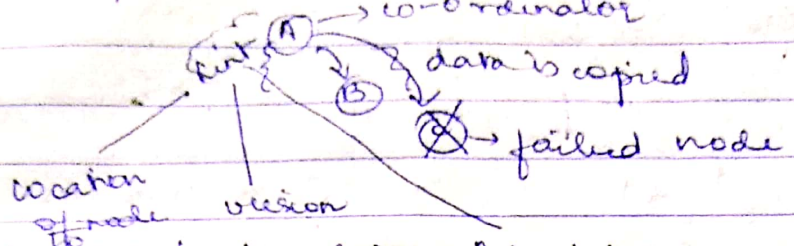
⑤ anti - entropy & Read repair
↓
updated data is compared with all replicated data

⑥ writes in cassandra

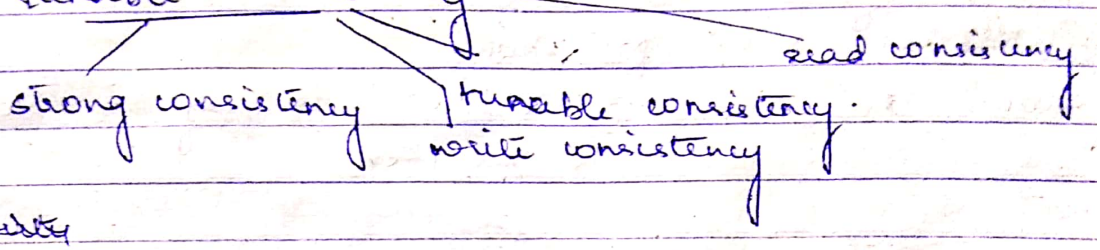
Once write is updated to log it is stored in memtable & written to sstable. Now data is sorted & kept. [SS table = sorted string table]

⑦ Limited tradeoffs

- multiple nodes in network



⑧ tunable consistency



~~Nosql~~

HDFS

Hadoop

① Everyday

facebook - 2.7 billion
google - 2.4 petabyte

② Every minute

FB - 2.5 million
twitter - 3,00,000
instagram - 2.20,000
youtube 72 hours.
email - 20,100,00 message.
Amazon - $800000

why Hadoop

- handle large amount of data.
- different categories of data.

features

① low cost
Since open source & use hardware to store large amt of data

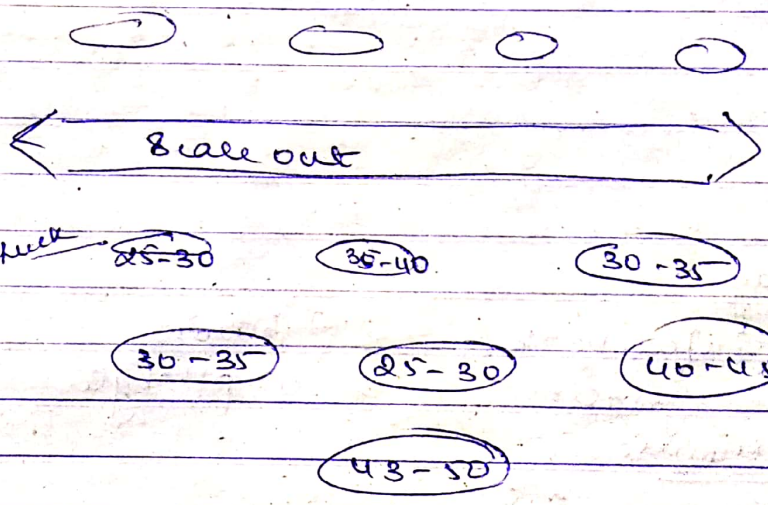
② computing power
uses distributed computing
↑ no. of nodes ↑ computing process

③ Scalability
more network, add extra nodes to improve performance.

④ flexibility in storage
storage of any kind of data is possible

⑤ Inherent data protection
How data is protected. On case of failure other nodes take up the task

Hadoop Framework



- multiple nodes together — chunks
- There can be replication of chunks for backup
- resources are parallely utilize.
- any no. of chunks can be added

Why Hadoop & nor RDBMS?

- nor suitable for advanced analytics — DBMS such as unstructured data
- in DBMS scaling out of data & huge computation is very costly

Parameters	RDBMS	Hadoop
(1) System	Relational Database Management System	node based flat structure
(2) Data	Structured	unstructured - video, audio, images - different format files
(3) Processing	OLTP (online transaction processing)	Big data processing & analytical
(4) choice	consistency & relationship b/w data	Big data & no consistency
(5) Processor	High End processor & costly hardware	cluster & multiple nodes hardware & NIC card
(6) cost	10^{000} to $14000/tera bytes of data	$9000/tera bytes

Distributed computing challenges

(1) Hardware failure

on case of hardware fail - how node access data.

(2) How to process gigantic data

Replication factor = 2

Hadoop Overview

(1) Faster data processing } 2 basic tasks.

(2) Massive data processing }

Key Aspects of Hadoop

→ Open source - free to use & contribute

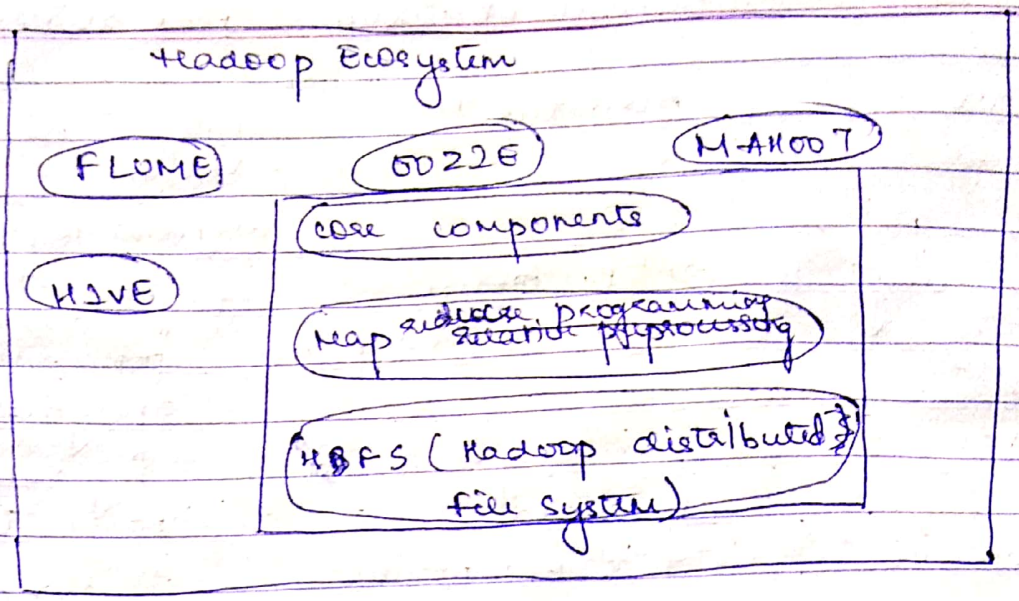
→ Framework - tools program & apps

→ Distributed - among multiple nodes

→ Massive Storage:

→ Faster processing - Quick response

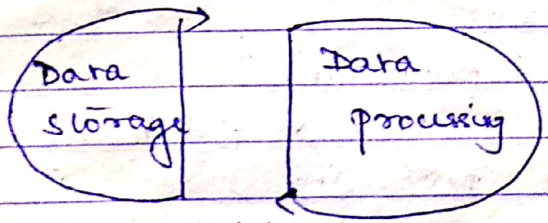
Hadoop Basic components



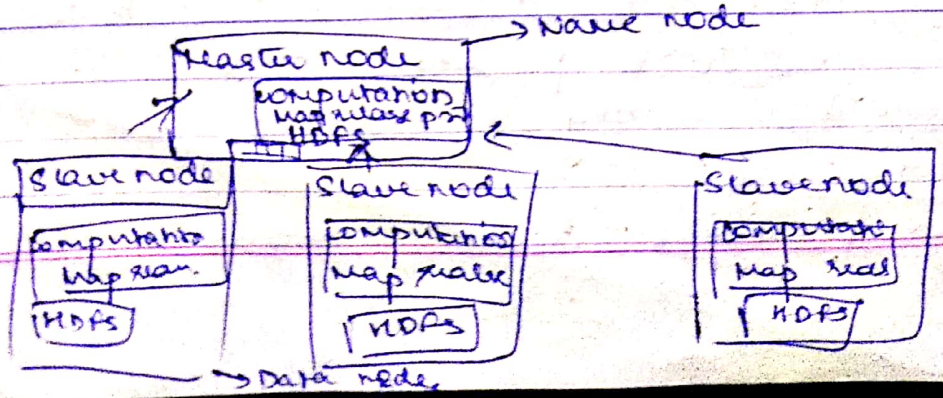
① HDFS:
- distributed system • data distributed across various nodes
- redundant
- storage component.

② Map reduce programming
- computational framework
- task & distributed across nodes
- parallel processing of data

Hadoop conceptual layer



- master slave architecture
- high level architecture of Hadoop



Master HDFS — keeps location of data in slave
- distributes data among slave

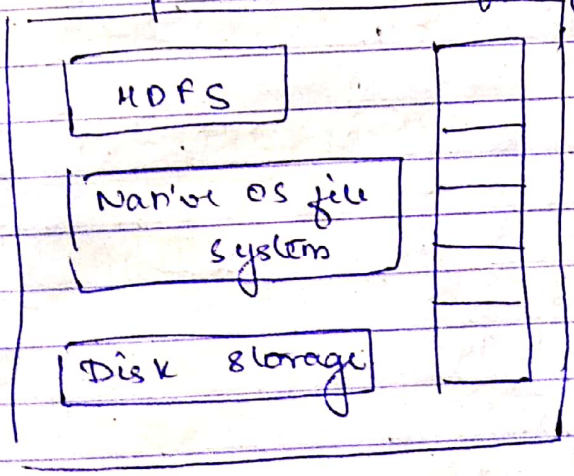
Master MapReduce - computations & distributed among slaves equally

use case of Hadoop
→ clickstream data - analysis on different data based on click
→ customer relationship management - know behaviour of particular customer
→ scalability - increase storage.
→ Business analysis - Apache, Hive.
→ HDFS (Hadoop Distributed file system)

Storage components
- Distributed file system
- Optimized for file throughput
- Modelled after google file system
- replicate the file any no. of times
- replicate data on failure
- read / write

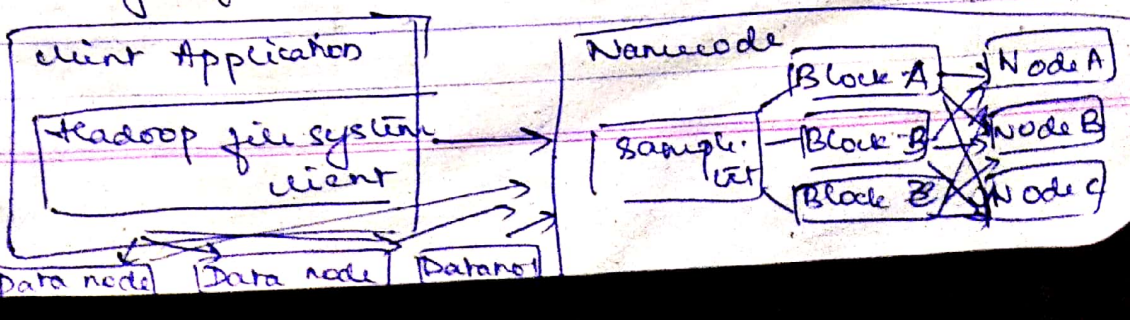
HDFS Daemons

Hadoop distributed file system

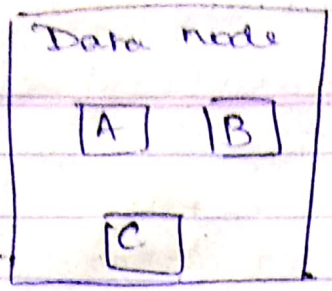


- store data in block - block structure file
- replicate data - replication factor is 3
- block size is 64MB

A large file is splitted into blocks, They are.



Each data node has following

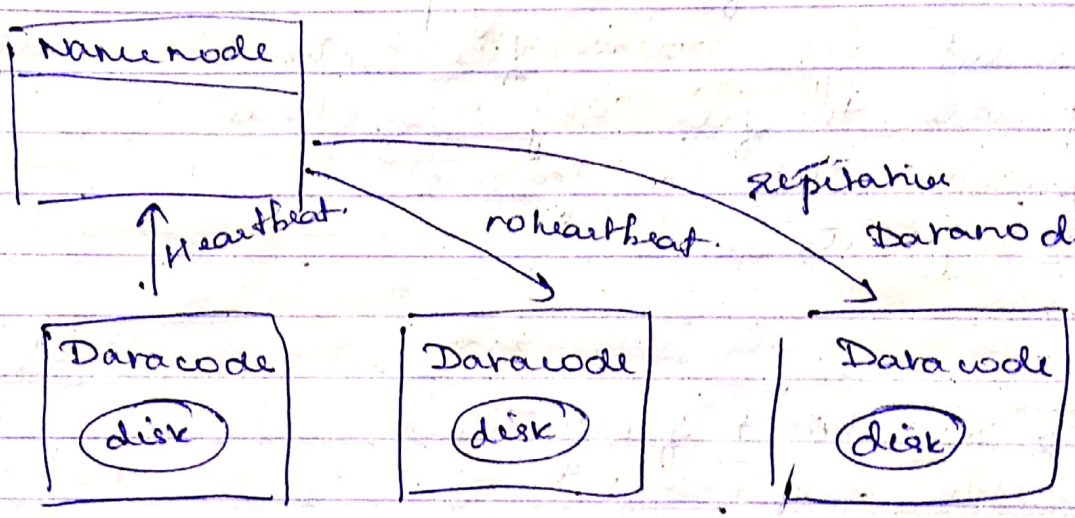


Name node manages various file operations that are done & manages file systems (read, write, update, delete)

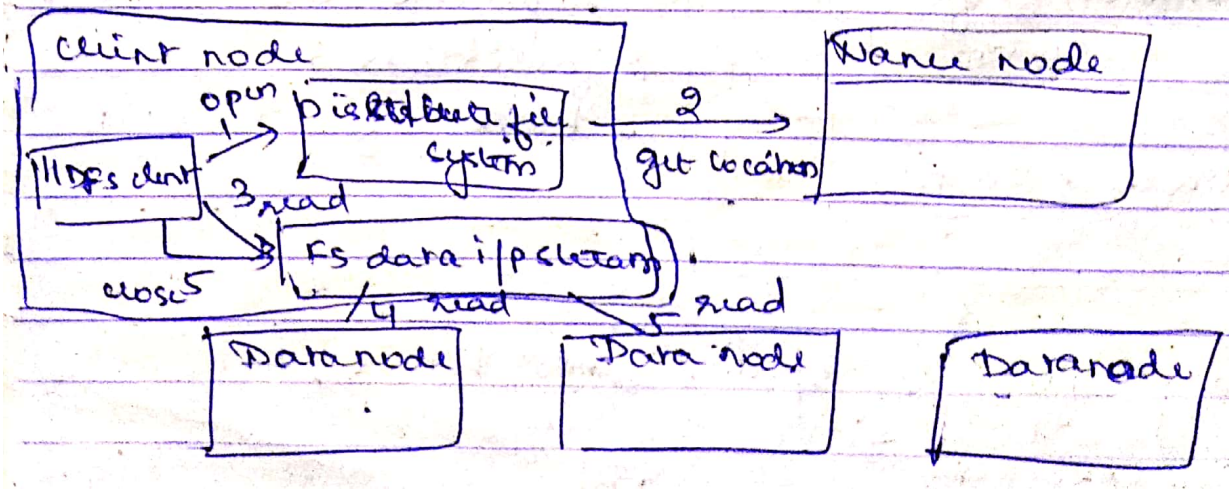
FS image – Interface to map between input file & blocks

Editlog – different transactions are dotted down

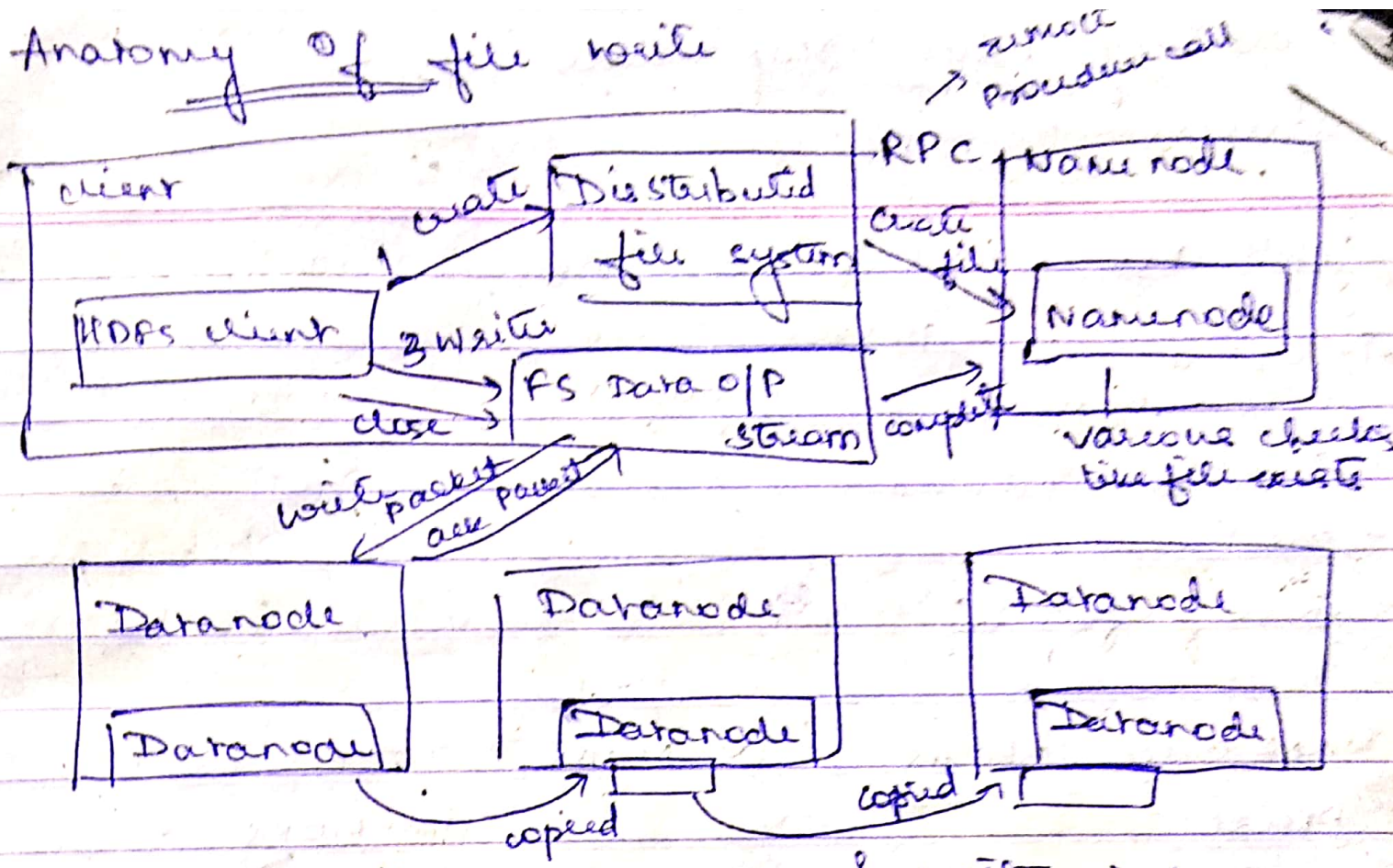
communication between Namenode & data node



Anatomy of File read

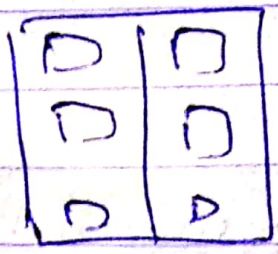


Anatomy of file write



data is divided into packets & written into internal Queue

Replica placement Stratiegy



Special features of HDFS

i) Data replication : client will check for Namenode

ii) Data pipeline : Data is put to first node & then moved & replica is in all node of pipeline