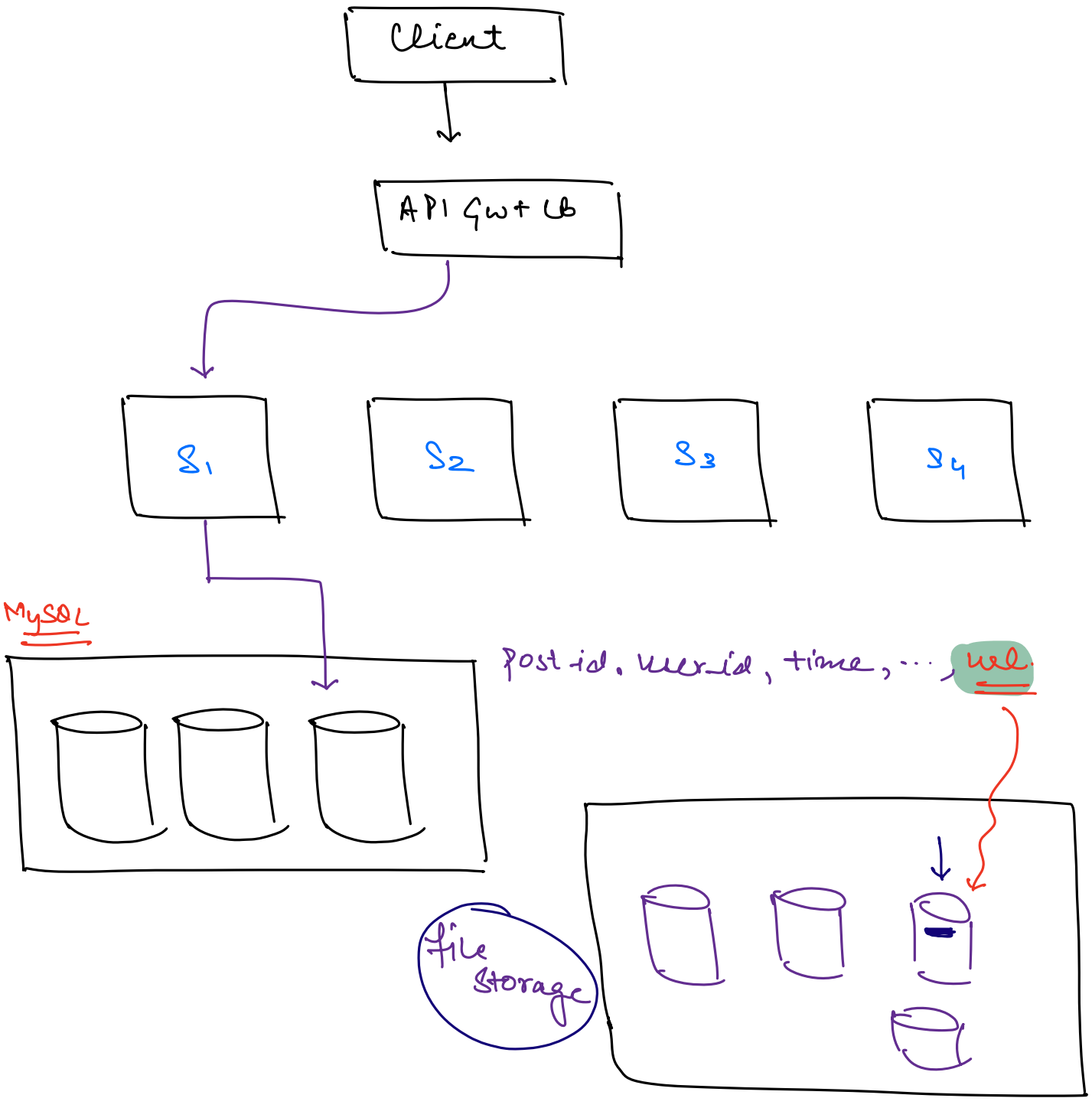
⇒ Design File Storage.

↳ S3 | HDFS | Blob Storage | ....

⇒ File Storage.

↳ Images | Videos | Media Content | log files



Store big files.

↓

10TB | 50TB | ...

Durable.

Performance of uploading & downloading.

→ what happens if the n/w gets broken in b/w while uploading/downloading.

10TB file

→ I) Upload the complete file in a single m/c

→ II) Distribute the file across multiple m/c.

Option # I : file stored as a unit in a 1 m/c.

Cons : 1) file size is limited by the m/c size.

2) Parallelism is NOT possible.

Pros :

1) No need to maintain multiple entries for each chunk.

(Cost of entries)

2) We don't have to collate the chunks at the time of downloading.

Option # II : Divide the file into multiple chunks & stores them into different m/c

Pros.

↳ Cons of Option # I

Cons.

↳ Pros of Option # I

⇒ Chunk size shouldn't be very small & shouldn't be very large as well.

⇒ HDFS

↓

Hadoop Distributed File Storage.

1) Machines which stores file chunks.

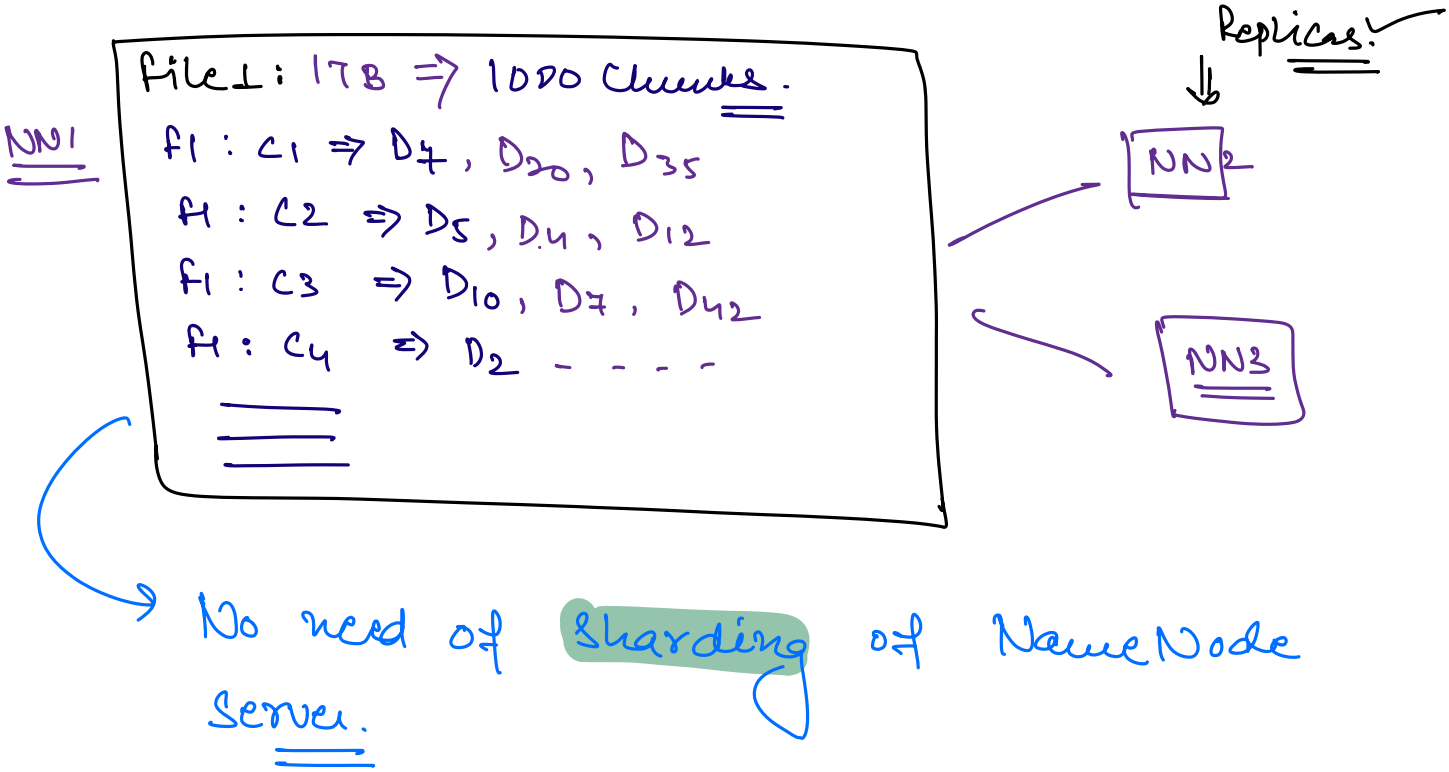
↳ Data Nodes.

2) We need to also maintain the metadata for all the files and their chunks.

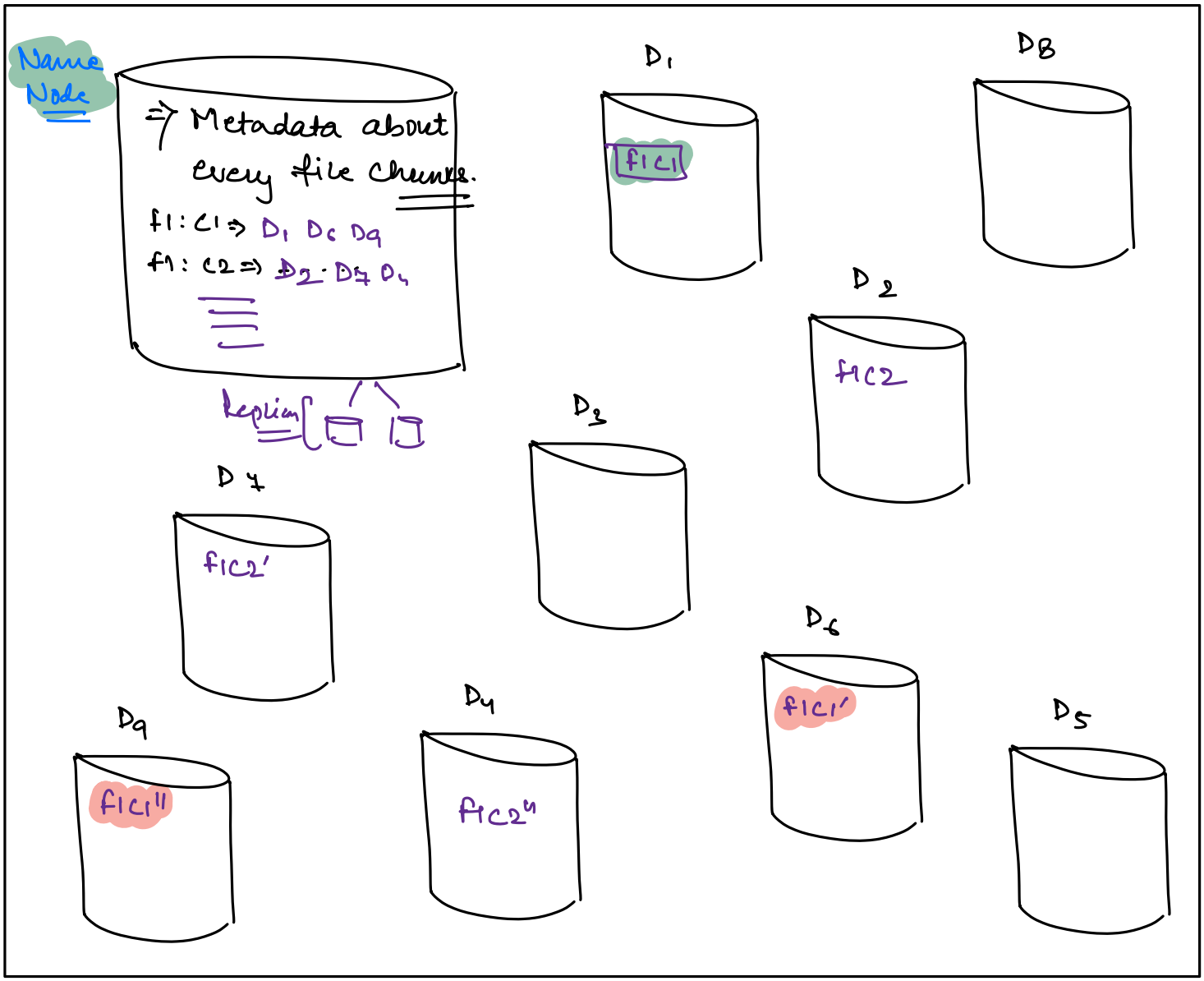
→ Name Nodes.

⇓

Maintains which chunk of which file is present on which m/c.

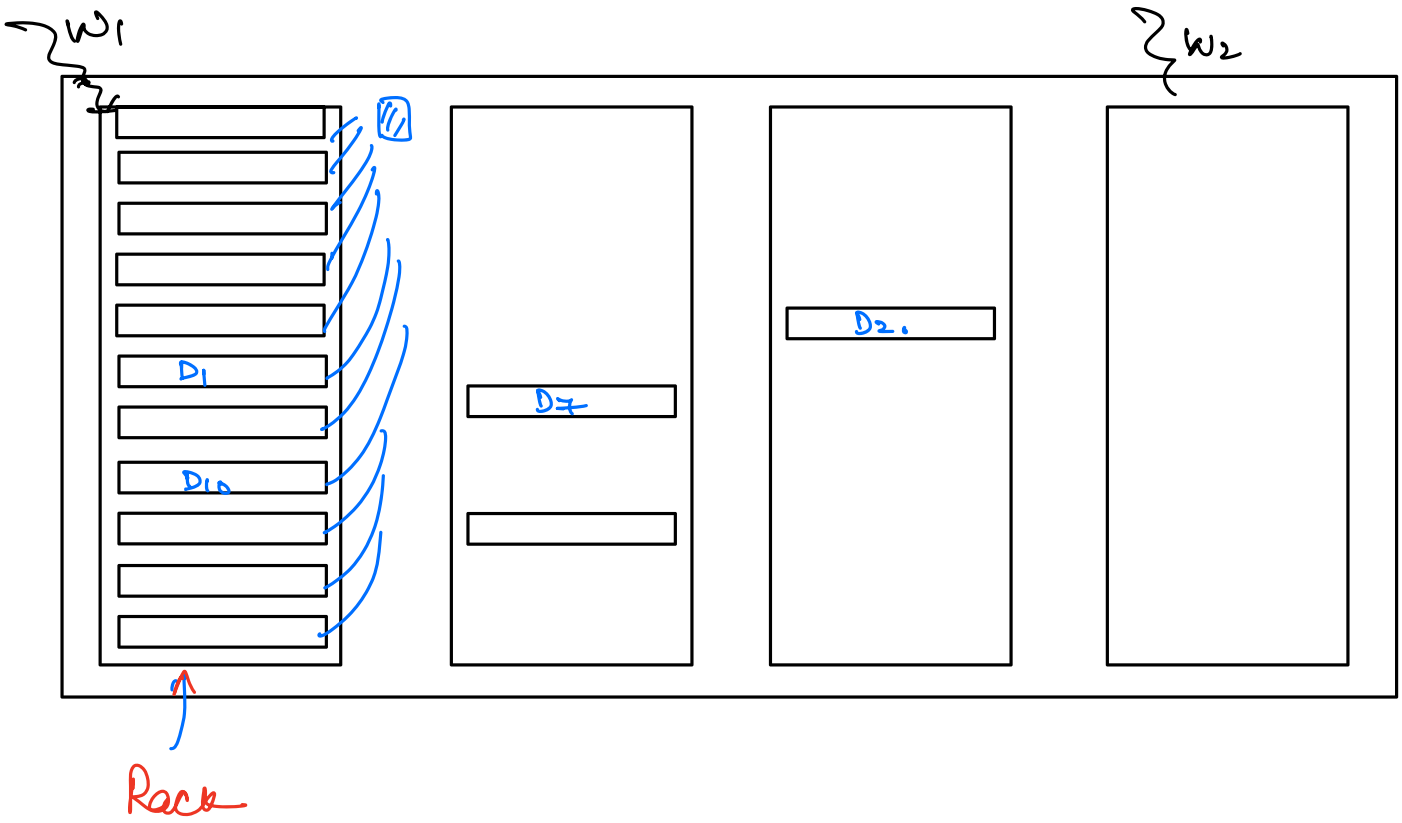


No need of Sharding of NameNode Server.

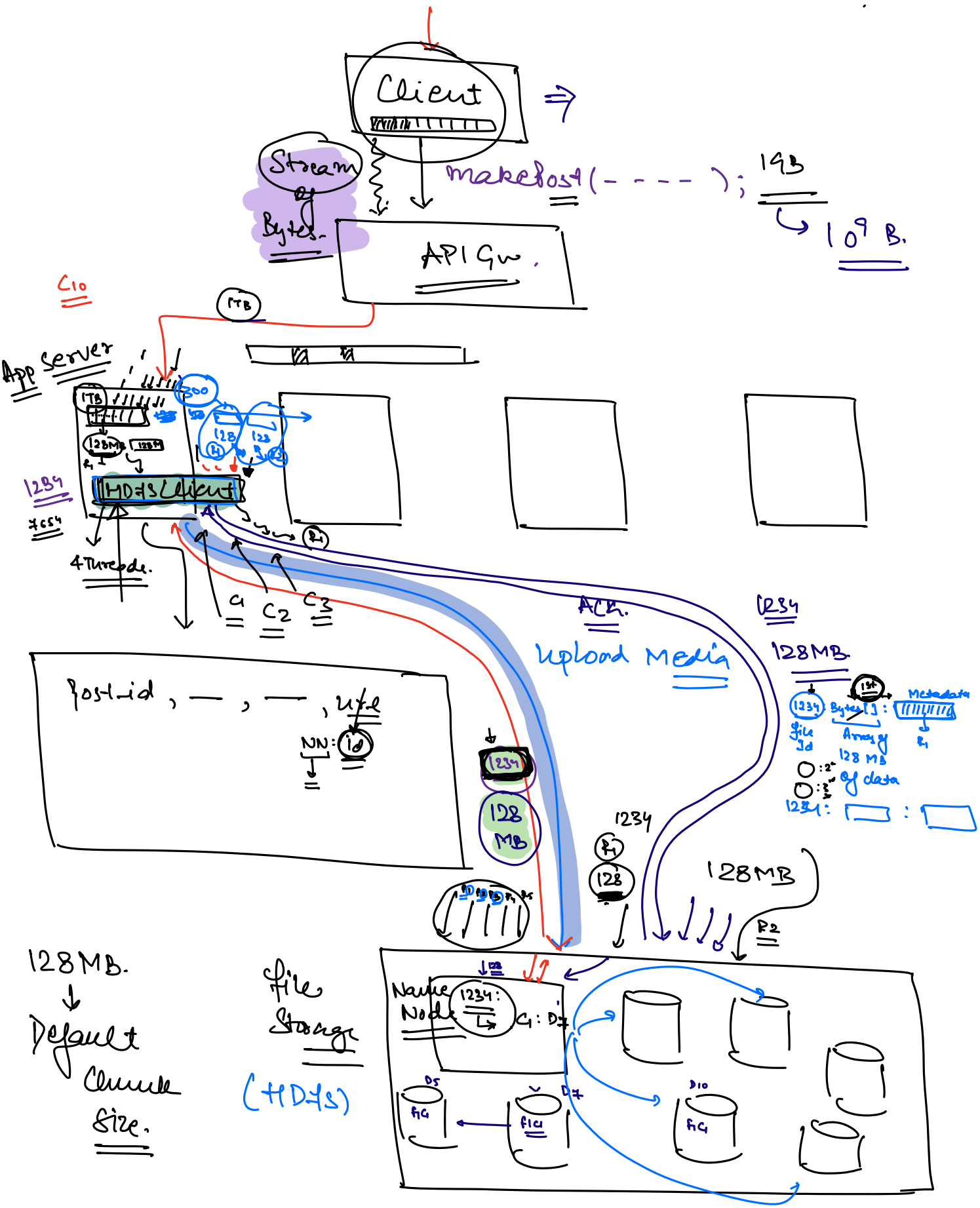

Name Node
⇒ Metadata about every file chunks.
f1 : C1 ⇒ D1 D6 D9
f1 : C2 ⇒ D2 D7 D4
Replica
D1
f1C1
D8
D2
f1C2
D3
D7
f1C2'
D6
f1C1'
D9
f1C1''
D4
f1C2''
D5

Rack Aware Algorithm.

⇒ Data centre



How the Read | Write operations happens in HDFS ?



7654 : C_1 : D_7 - - - -

C_2 : D_5 - - - -

C_3 : D_{10}

C_4 : D_{20}