


**IT 융 · 복합 비즈니스를 위한**

# **HDFS 동작원리 이해**



# HDFS

## (Hadoop Distributed File System)



HADOOP  
DISTRIBUTED  
FILE  
SYSTEM  
(HDFS)

HADOOP  
DISTRIBUTED  
FILE  
SYSTEM  
(HDFS)

### THE CAST

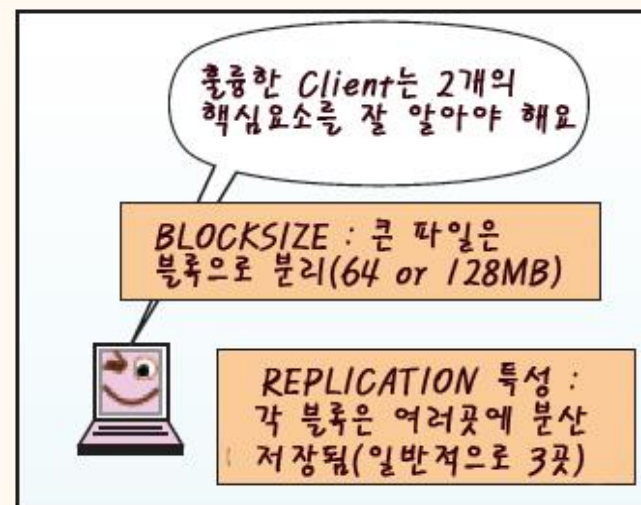
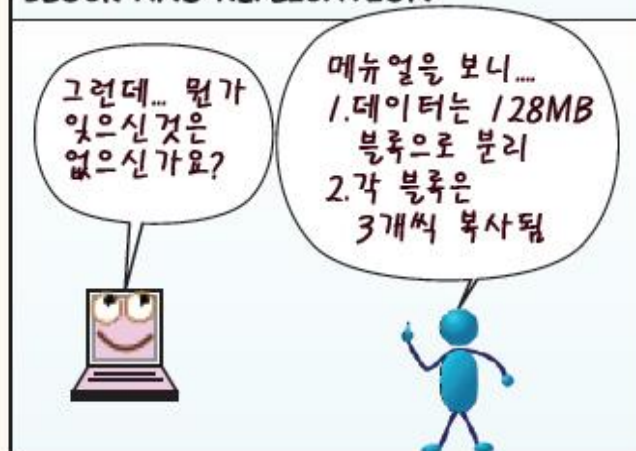


### WRITING DATA IN HDFS CLUSTER

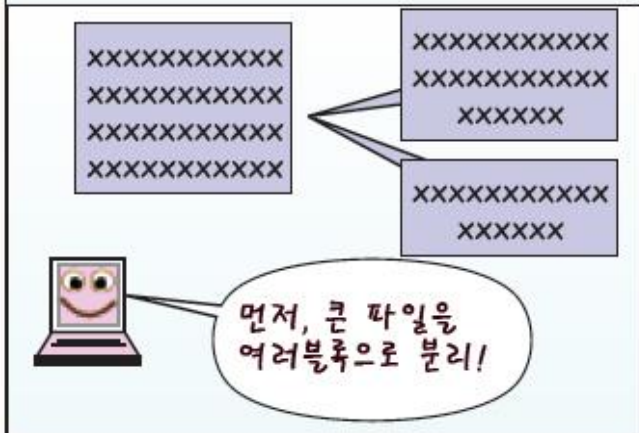
#### REQUEST FROM USER



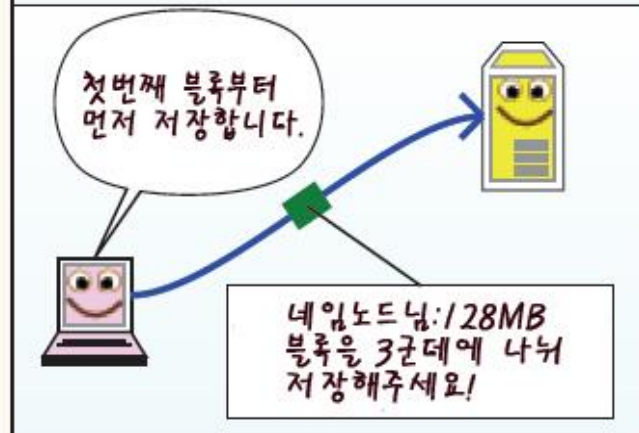
#### BLOCK AND REPLICATION



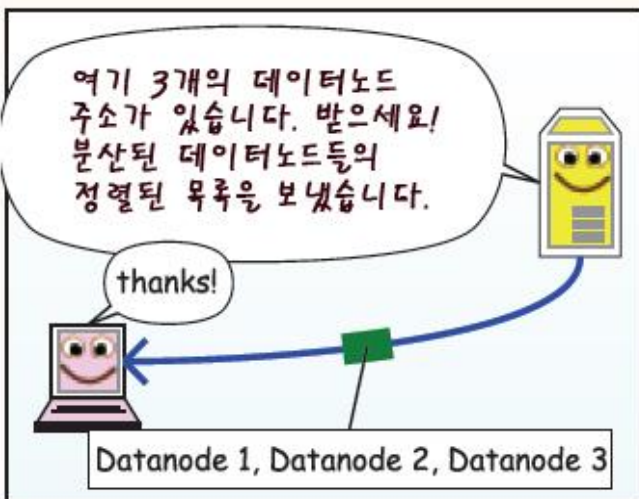
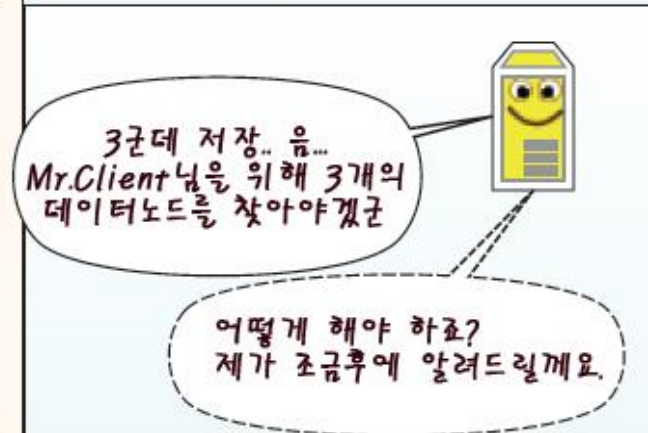
### DIVIDE FILE INTO BLOCKS



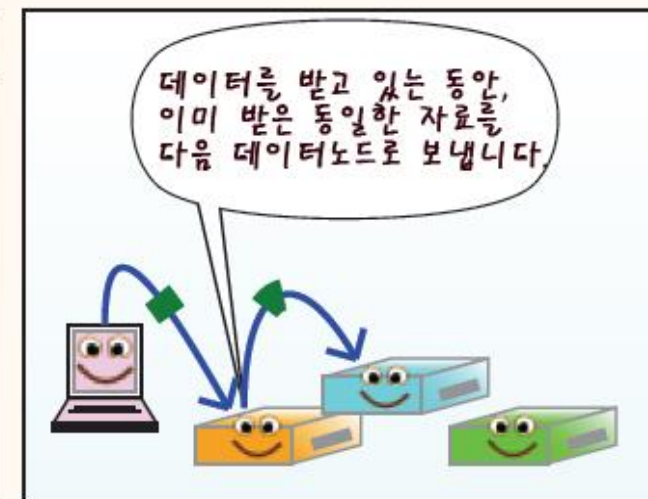
### ASK NAMENODE



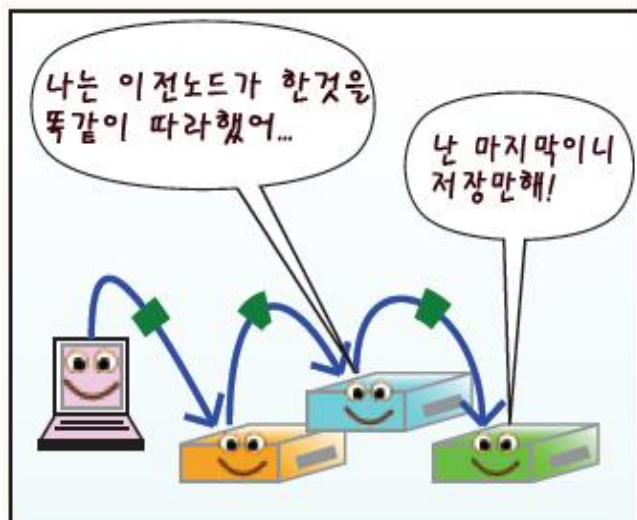
### NAMENODE ASSIGNS DATANODES



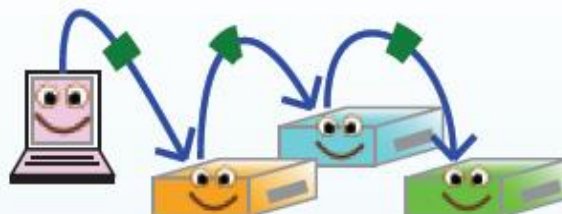
### CLIENT STARTS WRITING DATA





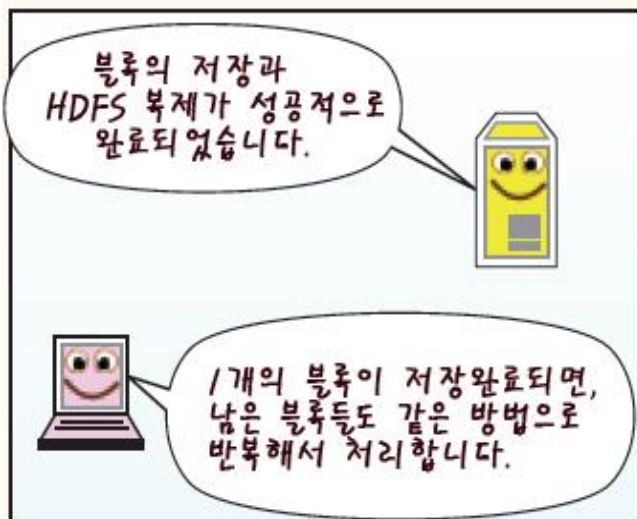
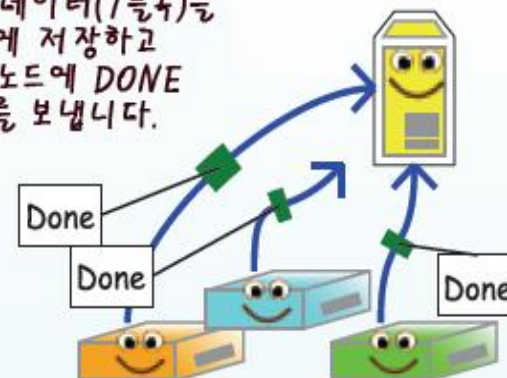


### TA..DA.. REPLICATION PIPELINE



### INFORM NAMENODE WHEN DONE

모든 데이터(/블록)를 하드에 저장하고 네임노드에 DONE 신호를 보냅니다.



### WHEN ALL BLOCKS ARE WRITTEN..

모든 블록 저장완료! 파일을 닫아주세요.



### RECAP

파일을 여러블록으로 분할...

각 블록을 위해... 데이터노드의 주소들을 저장해놓았어요...

우리들은 Replication Pipeline에 데이터를 저장했어요.

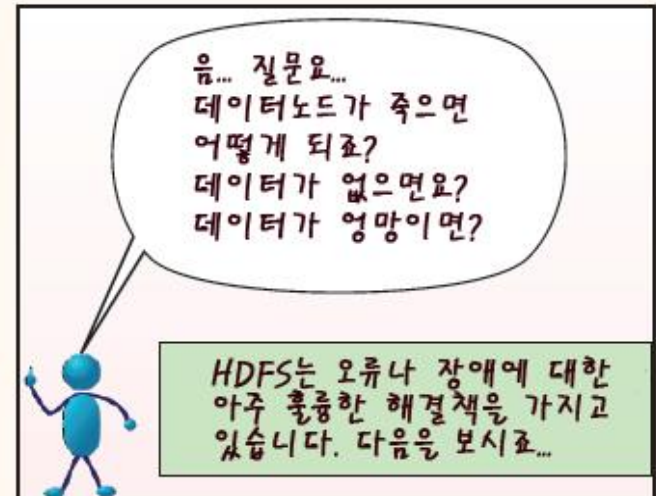
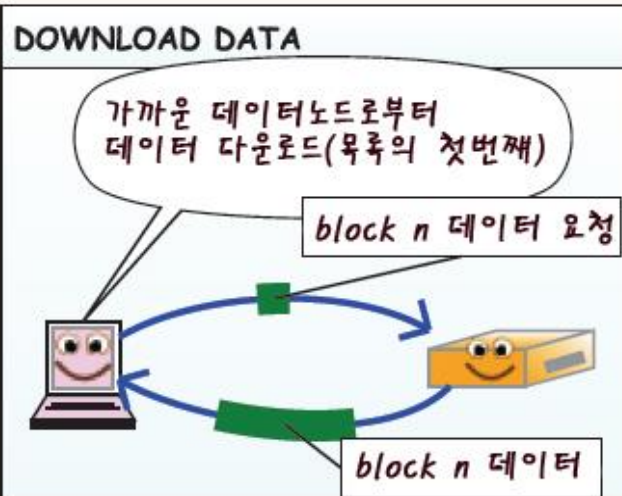
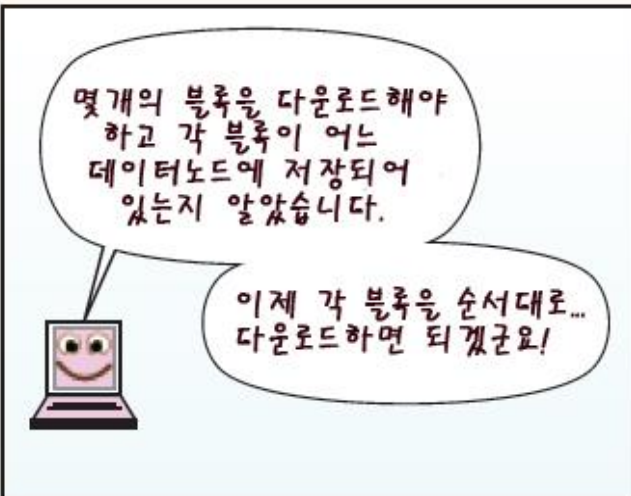
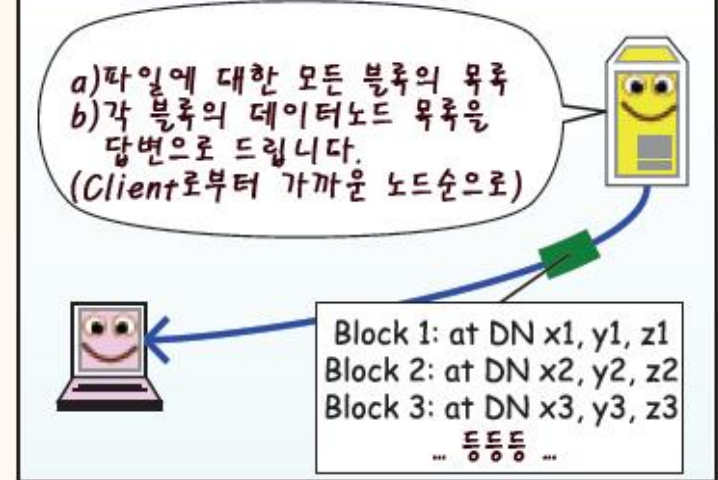
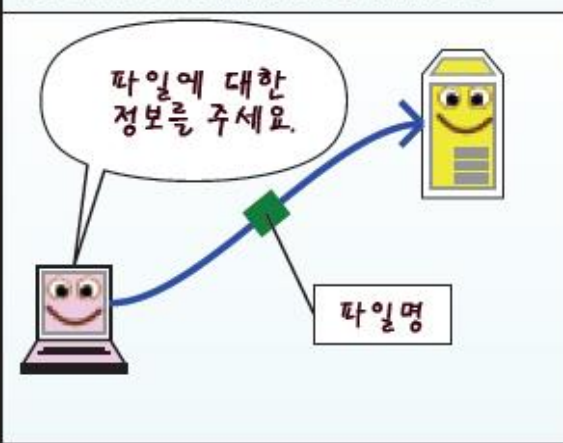


### READING DATA IN HDFS CLUSTER

#### REQUEST FROM USER



#### CONTACT NAMENODE FIRST..



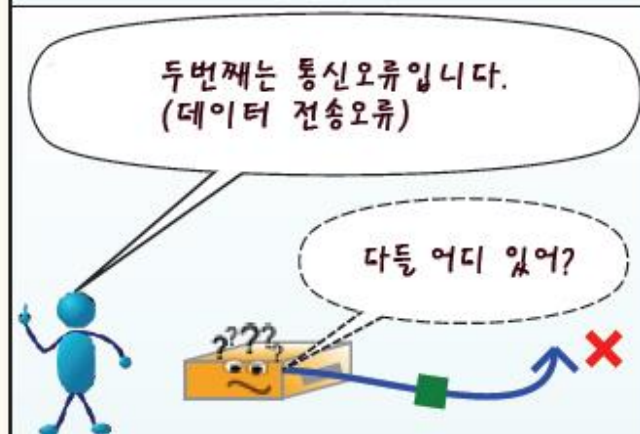


### FAULT TOLERANCE IN HDFS. PART I: TYPES OF FAULTS AND THEIR DETECTION

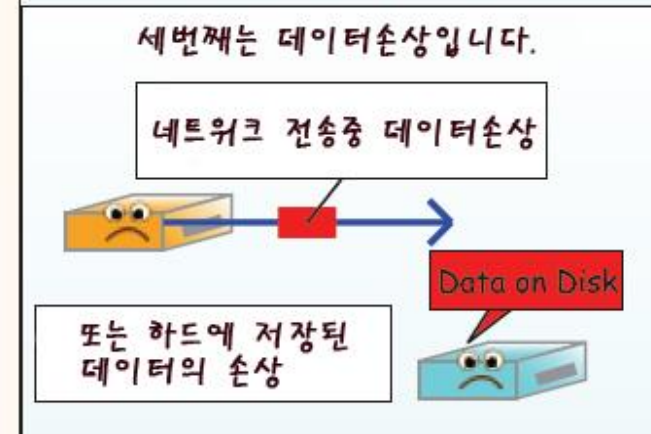
#### FAULT I: NODE FAILURE



#### FAULT II: COMMUNICATION FAILURE



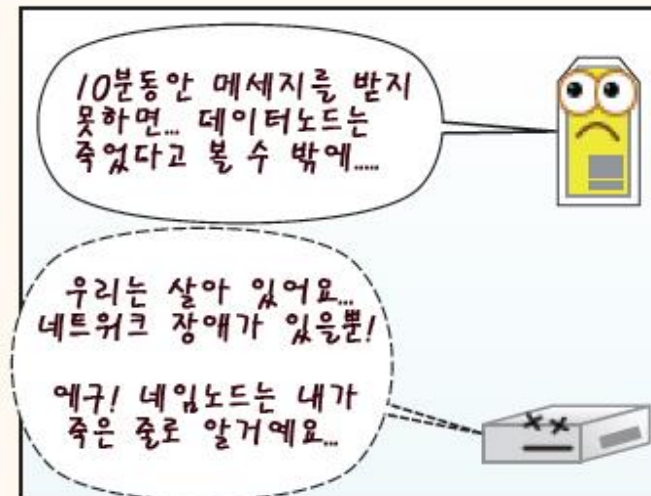
#### FAULT III: DATA CORRUPTION



#### DETECTION #1: NODE FAILURES



#### DETECTING DATANODE FAILURE



### DETECTION #2: NETWORK FAILURES

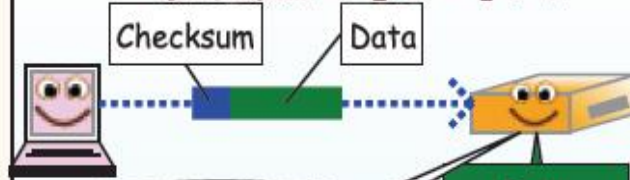
데이터를 보낼때마다  
노드는 ACK 응답을 합니다.



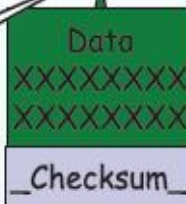
ACK 응답이오지 않으면(여러번 시도후)  
송신자는 노드가 죽었거나  
네트워크 오류로 판단합니다.

### DETECTION #3: CORRUPTED DATA

데이터를 전송할때  
Checksum도 같이 보냅니다.



데이터를 하드에 저장할때  
checksum도 같이  
저장합니다.



### DETECTING CORRUPTED HARD DRIVES

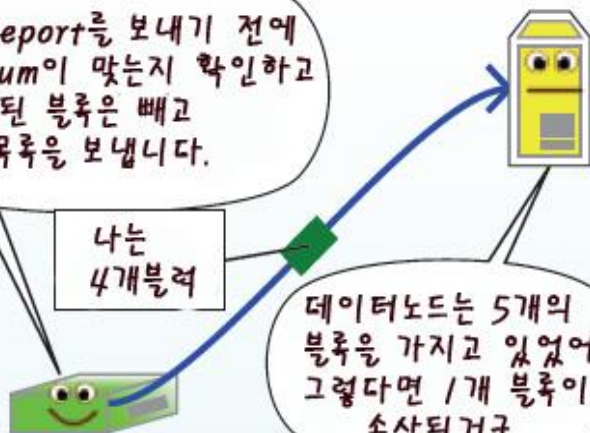
주기적으로 모든  
데이터노드는 네임노드에  
BlockReport를 보냅니다.



BlockReport를 보내기 전에  
checksum이 맞는지 확인하고  
손상된 블록은 빼고  
블록목록을 보냅니다.

나는  
4개블럭

데이터노드는 5개의  
블록을 가지고 있었어.  
그렇다면 1개 블록이  
손상된거군



### RECAP: HEARTBEAT MESSAGES AND BLOCK REPORTS

노드가 살아 있다는 것을  
알리기 위해 3초에 한번씩  
Heartbeat를 보냅니다.



손상된 블록을 제외하고  
BlockReport를  
네임노드에 보냅니다.

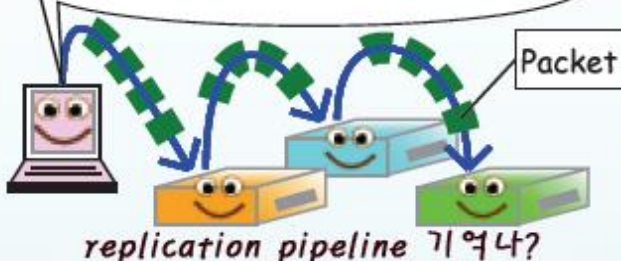
그러면 네임노드는  
어떤 블록이  
손상되었는지  
자연히 알게되요



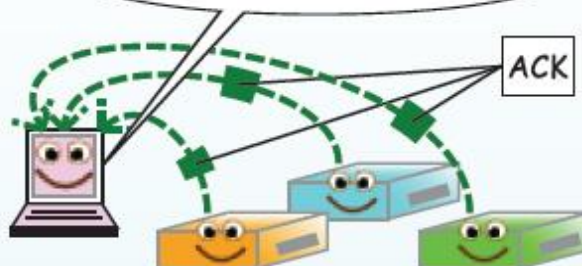
### FAULT TOLERANCE IN HDFS. PART II: HANDLING READING AND WRITING FAILURES

#### HANDLING WRITE FAILURES

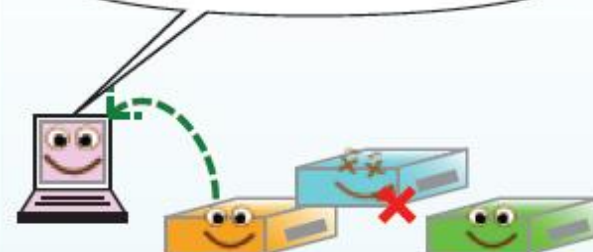
앞에서 “packet” 이라 불리는 동일한 데이터집합을 여러 데이터노드에 저장했습니다.



그리고 데이터노드들은 패킷을 저장하고 나서 ACK 응답을 한다고 했어요



어떤 데이터노드로부터 ACK 응답을 받지 못하면 죽었다고 판단하고 다시 pipeline을 설정해야 합니다.



여기 다시 설정된 pipeline이 있습니다. 하지만 데이터는 “완전히 복제” 되지는 않았습니다. 나중에 네임노드가 처리합니다.

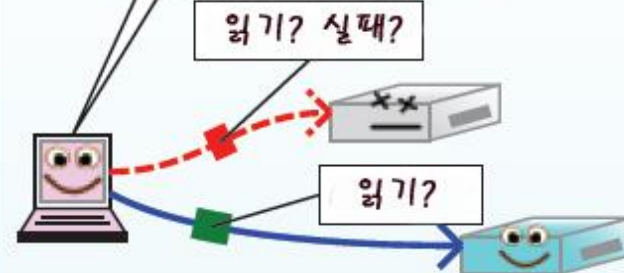


#### HANDLING READ FAILURES

블록의 위치를 요청했을때 네임노드가 모든 데이터노드의 위치를 알려준것 기억나세요?



1개의 데이터노드가 죽었다면 I 목록에 있는 다른노드에서 데이터를 읽습니다.



### FAULT TOLERANCE IN HDFS. PART III: HANDLING DATANODE FAILURES

저는 네임노드입니다.  
2개의 목록(블록,노드위치)을  
저장하고 있어요.



List of Blocks  
Block 1 - stored at DN1, DN2, DN3  
Block 2 - stored at DN1, DN4, DN5

List of Datanodes  
Datanode 1 - has block 1, 2, ..  
Datanode 2 - has block 1, 5, ..

2개의 목록을 지속적으로  
업데이트하고 있어요



장애가 발생한 노드를 찾고  
첫번째 목록(블록)을 업데이트합니다.  
(오류가 난 데이터노드를 삭제)

그리고 두번째 목록(노드)도  
업데이트 합니다.

#### UNDER REPLICATED BLOCKS



첫번째 목록을 주기적으로  
스캔합니다. 복제가 완전하지  
않은 블록을 찾습니다.

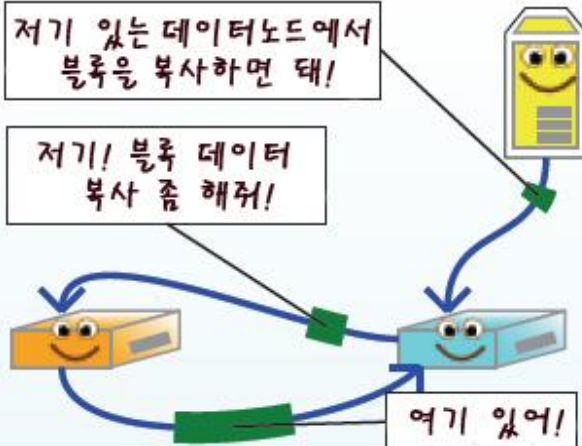
이러한 블록을  
“불완전 복제” 라고 부릅니다.

[불완전 복제] 블록을 위해서,  
새로운 복제소로 복사하기 위해  
다른 데이터노드에 요청을 합니다.



저기 있는 데이터노드에서  
블록을 복사하면 돼!

저기! 블록 데이터  
복사 좀 해줘!



음.. 한가지 질문이 있어요:  
이런 작업을 위해서는 적어도  
1개의 정확한 데이터가  
있어야 하잖아요?

맞습니다. HDFS는  
적어도 1개의 복제소에  
대해 확실히 보장을 합니다.

최적의 복제소 선정이 중요  
하죠. 다음을 보시면 됩니다.





### REPLICA PLACEMENT STRATEGY

해당 블록을 저장할 데이터노드를 선정하는 방법을 알려드리겠습니다. 약속했습니다.

조금만 기다리세요.



### RACKS AND DATANODES

클러스터는 여러개의 데이터노드를 가지는 랙으로 분리되어 있습니다.



Rack 1



Rack 2



Rack 3

### SELECTING FIRST REPLICA LOCATION

첫번째 복제소는 간단합니다.

Client 데이터가 같은 랙의 노드에 있다면, 첫번째 복제소로 선택됩니다.

그렇지 않으면 랜덤으로 복제노드를 선택합니다.



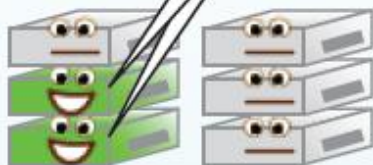
### NEXT TWO REPLICA LOCATIONS

첫번째 노드와는 다른 랙에서 2개의 다른 데이터노드를 선택합니다.

첫번째 노드



다음 2개 노드



### SUBSEQUENT REPLICA LOCATIONS

2가지 조건을 만족시키는 데이터노드를 랜덤으로 선정

우리 랙에서 유일한 복제소



랙에 있는 최대 2개의 복제소



가끔 2가지 조건을 만족하지 않는 경우가 발생합니다. 그럴때... 무시!...

HDFS는 자신만의 배치 알고리즘을 허용합니다. 더 좋은 알고리즘이 있다면 ... 부끄러워하지 말고 ...





