Week - 1

Understanding Big Data
The Big Picture

Big Data

- Volume
- Variety
- Velocity
- Veracity
- Value

Need for a new technology stack ?

Monolithic vs Distributed

One powerful system

Compute, Memory, Storage

2x resources ≠ 2x performance

Vertical Scaling

Cluster of systems (commodity machines)

Horizontal Scaling (True Scaling)

Design a good Big Data System? Consider

- Storage (distributed storage)
- Process (distributed processing)
- Scalability (↑ number of nodes)

Hadoop → first framework designed to solve Big Data problems

2007 → 2012 → Now
1.0 → 2.0 → 3.0

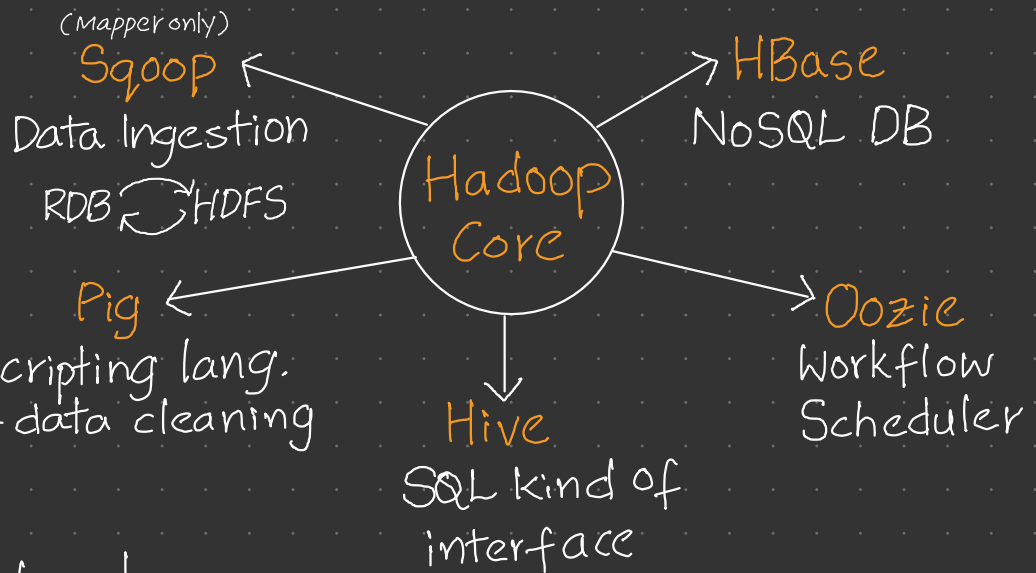
Core Components

- HDFS : Storage
- MapReduce : Process
- YARN : Resource Manager
  (added in 2.0)

Challenges

- MR is very slow and hard
- Need to learn different components for different tasks
  ↓
  under the hood, they convert to MapReduce

Hadoop Ecosystem



Advantages of Cloud

- Scalable
- Minimum OpEx
- Agility
- Geo Distribution
- Disaster Recovery

On-Premise Vs Cloud

On-Premise	Cloud
- Buy the needed Infra, like Space to hold the servers	- Infra is taken care by Cloud providers
- Buy servers	- No need to buy servers
- Setup a cooling system	- Setup the cluster with a click of a button
- Hire a technical team to install and maintain S/W	- Low maintenance cost
- Huge upfront cost / Capex & Opex	- No upfront cost / Capex
- Not very scalable	- Highly Scalable

Cloud Types

- Public : AWS, Azure, GCP
- Private : when the data is very confidential
- Hybrid

Spark → general-purpose in-memory compute engine

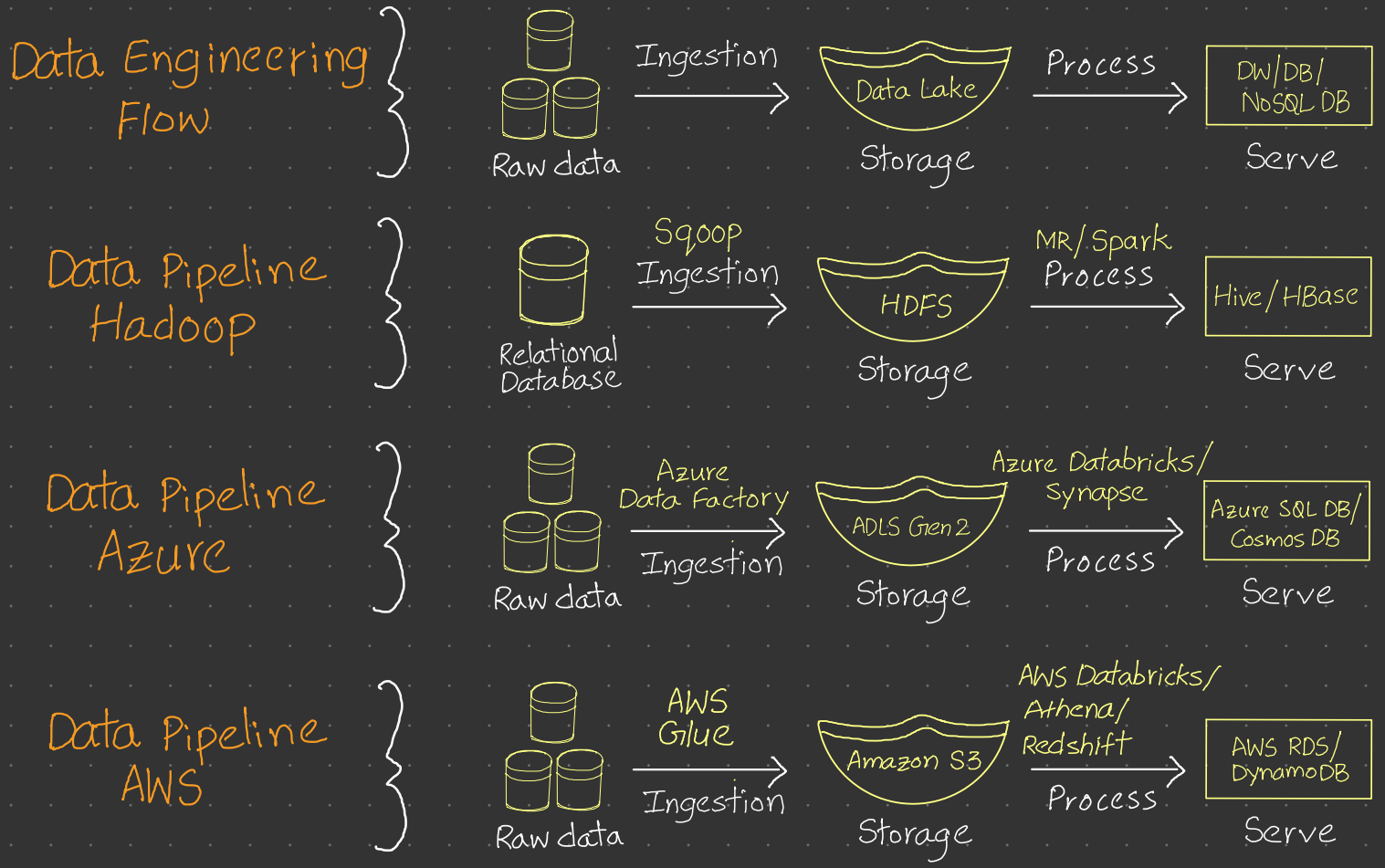
- Spark can act as a replacement for MapReduce (not for Hadoop)
- It is a plug and play compute engine
- Needs storage and resource manager to function.
  - not bound to HDFS, and YARN, only.
    - Amazon S3, ADLS Gen 2, GIC Storage, Local Storage
    - Mesos, Kubernetes

- Spark is 10x - 100x faster than MR

Polyglot

- Provides high-level APIs in Java, Scala, Python and R
- Spark is written in Scala

	Database	Data Warehouse	Data Lake
Workloads	- OLTP	- OLAP	- OLAP
Data type	- Structured or Semi-Structured	- Structured and/or Semi-Structured	- Structured, Semi-Structured and/or unstructured
Schema	- Schema-on-write	- Schema-on-write	- Schema-on-read
Freshness	- operational current data	- current and historical data	- current and historical data
Prior Processing	- ETL	- ETL	- ELT
Cost	- High	- High	- Low
Examples	- PostgreSQL, Oracle, MySQL	- Teradata, AWS Redshift	- Amazon S3, ADLS Gen 2
Users	- Application developers	- Business Analysts	- Data Scientists


Data Engineering Flow
Raw data
Ingestion
Data Lake
Storage
Process
DW/DB/ NoSQL DB
Serve
Data Pipeline Hadoop
Relational Database
Sqoop Ingestion
HDFS
Storage
MR/Spark Process
Hive/HBase
Serve
Data Pipeline Azure
Raw data
Azure Data Factory
Ingestion
ADLS Gen2
Storage
Azure Databricks/ Synapse
Process
Azure SQL DB/ Cosmos DB
Serve
Data Pipeline AWS
Raw data
AWS Glue
Ingestion
Amazon S3
Storage
AWS Databricks/ Athena/ Redshift
Process
AWS RDS/ DynamoDB
Serve

Serverless Computing

- Resources are not dedicated
  - ↓
  - No guaranteed performance
- Less expensive
- Good for scheduled jobs
  - ↓
  - that are not particular about the time in which they get finished
- Eg:- Athena, Synapse serverless

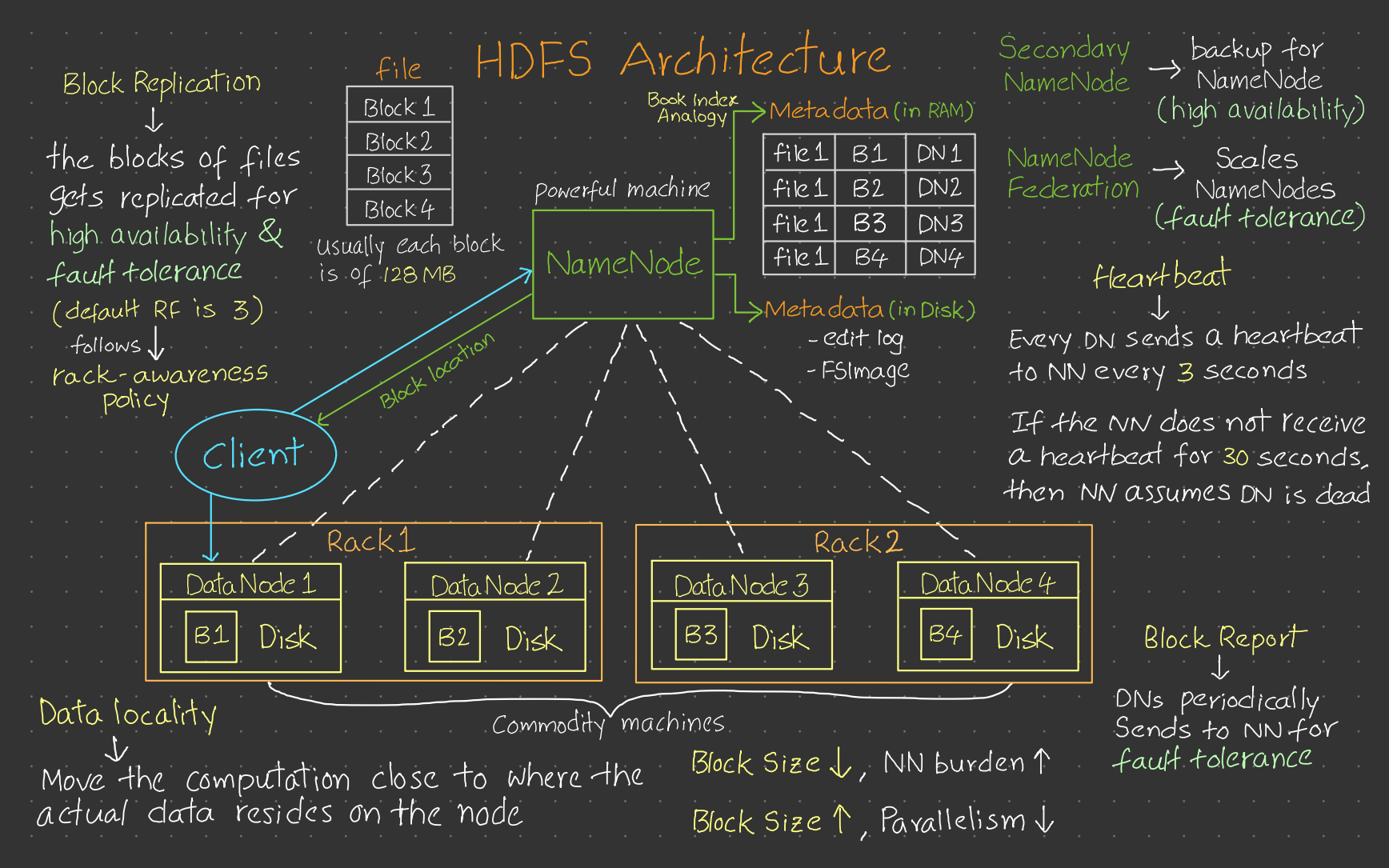
Serverful Computing

- Resources are dedicated
  - ↓
  - guaranteed performance
- More expensive
- Good for ad-hoc jobs
  - ↓
  - that require immediate execution
- Eg:- Redshift, Synapse dedicated pool

Week - 2

Distributed Storage

HDFS Architecture



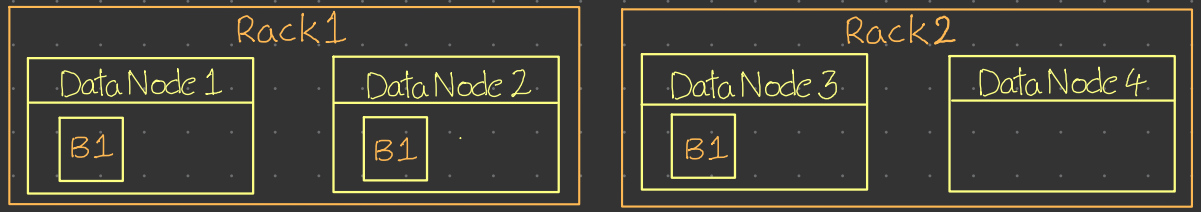
Rack Awareness Policy

Data Replication → High Availability
Data Replication → Fault Tolerance

Rack awareness policy says:

- Not more than 1 replica be placed on 1 DN
- Not more than 2 replicas are placed on the same rack
- The #racks used in a cluster must be < #replicas

Helps to maximize the Network Bandwidth

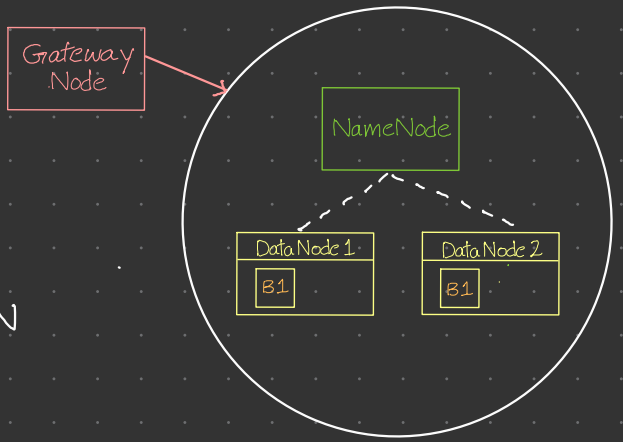


Hadoop mainly works on 3 different modes

Standalone Mode	Pseudo-distributed	Fully-distributed
- Default mode	- Single-node cluster	- Production mode
- Uses local file system	- Uses HDFS	- Uses HDFS
- Used for debugging	- Used for testing	
- No need to change Configuration files	- Need to change Configuration files	- Need to change Configuration files

Gateway or Edge node

- Connects to the distributed cluster, but does not run any of the daemons.
- Prevent users from a direct connection to critical components such as NN or DN



Linux Commands

`pwd` show present working dir

`whoami` show user name

`cd` to change a particular dir

`cd` or `cd ~` navigate to home dir

`cd /` navigate to root dir

`cd .` current dir

`cd ..` move one level up

`cd -` navigate to previous dir

`ls` list files and directories

`ls -l` long listing format

`ls -t` sort by last modified

`ls -r` reverse order by dict

`ls -a` list hidden files as well

`ls -R` recursive list

`ls -S` sort by file size

`ls -h` human readable size

Absolute Path starts from the root /

Relative Path starts relative to the current location

`touch` creates an empty file

`vi` edit or create a file

chmod change the permissions

r	read	4
w	write	2
x	Execute	1

chmod 644 file_name

4+2	4	4
rw-	r--	r--
Owner	Group	Other

chmod -R change the permissions of a dir recursively

Cat view, create, concatenate

Cat file_name view

Cat > new_file_name create

Cat file1 >> file2 concatenate

Cat file1 file2 > merged_file_name

mkdir make new dir

rmdir removes an empty dir

rm removes files

rm -R removes a dir recursively

cp copy files and dir's

mv move/rename files & dir's

head show first 10 lines

tail show last 10 lines

grep global search for regex and print out

du disk space used

du -h human readable

HDFS Commands - linux based

To execute a linux command on a distributed environment, just prefix it with : `hadoop fs -` or `hdfs dfs -`

HDFS Specific Commands

Copying files/dir's from local to hdfs :

`hadoop fs -put` or `-copyFromLocal <local file path> <hdfs file path>`

Copying files/dir's from hdfs to local :

`hadoop fs -get` or `-copyToLocal <hdfs file path> <local file path>`

File syste check - used to check the health of HDFS

`hadoop fsck <hdfs file path> -files -blocks -locations`

HDFS

- Distributed file system that stores data as blocks
- HDFS is not persistent
  i.e; data is lost when the Hadoop cluster is terminated
  (newer versions have storage mgmt. features & functions consistent with persistenting data)

* - Storage is tightly coupled with compute (I/O Performance benefits)
  i.e; Increasing storage capacity entails increasing Compute & Memory
- The data in HDFS is tied to the cluster it is stored on
  i.e; Cannot access it from another cluster

Cloud Data Lake

- Object-based storage that stores data as objects { Unique id, value, metadata
  E.g; Amazon S3, ADLS Gen2
- Amazon S3 and ADLS Gen2 are persistent

- Storage is decoupled from compute
  i.e; We only need to pay for the storage we use
- Any number of clusters can access the same data