Big Data Technologies

Agenda

- Introduction to Big Data

Big Data Technologies - Module

Contents

- Fundamentals
- Hive (SQL)
    - Data warehousing
- Spark (Python)
- Kafka (Python)
- Hadoop (Java)
- HBase (Ruby)
- AirFlow (Python)

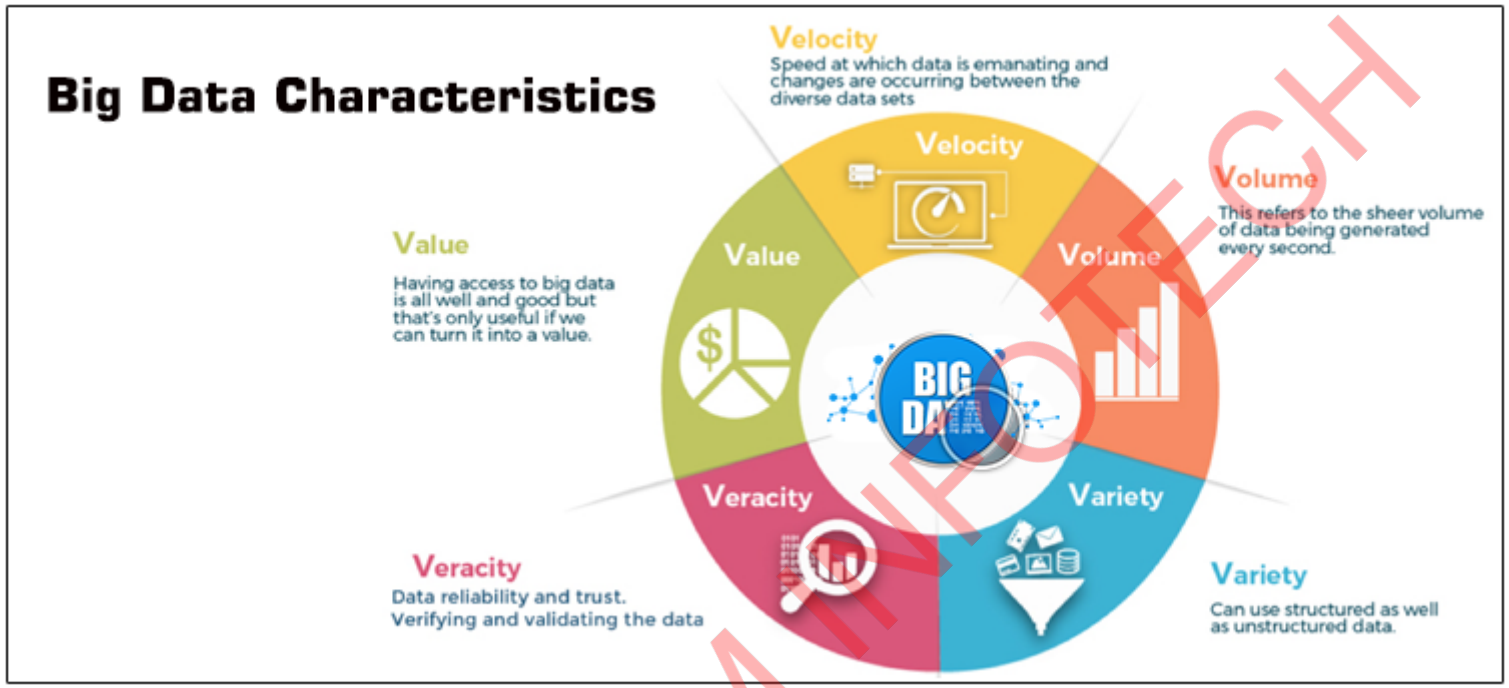
Evaluation

- Theory: 40 marks (CCEE - MCQ)
- Lab: 40 marks (Hive, Spark, Kafka, HDFS, ...)
- Internal: 20 marks (Lab assignment, Quiz)

Introduction

- https://www.youtube.com/live/BxwpqnQ6BgQ?si=ZftRBP5zMrUUZIUJ
- History of Big Data
    - 1970: Database (RDBMS)
    - 1990: Data warehouse

    - 1990-95: Internet (WWW)
    - 1998-08: NoSQL
    - 2000: MPP
    - 2003: Big Data
    - 2006: Cloud computing
- Structured vs Unstructured data vs Semi Structured data
    - Structured data
        - Data with fixed format - Expressed in rows and columns
        - Needs less storage
        - Easier to manage and secure with legacy systems
        - Examples: RDBMS, etc.
    - Unstructured data
        - Data with no format
        - Usually much higher storage
        - Examples: Text, Images, Audio, Video.
    - Semi-structured data
        - Flexible structured data
        - Better storage utilization
        - Usually hierarchical structure or Key-value format
        - Examples: XML, JSON, etc.
- Big Data characteristics
    - Volume: Huge volume of the data (usually in TBs+)
    - Velocity: Data generated/changing at high rate (in seconds)
    - Variety: Data in different formats (structured, semi-structured, or unstructured)
    - Veracity: Data reliability and trust
    - Value: Meaningful and useful data



- Big data processing
  - Batch processing
    - Processing finite set of data (data at rest)
    - Incremental data load is managed by programmer
    - Cluster planned as per data size. High throughput
    - Job run once per batch
  - Stream processing
    - Processing live stream of data (data in motion)
    - Data processing is managed by the framework
    - Less throughput.
    - Job is running forever
  - Interactive processing
    - Results are generated on basis of user interactions

        - User queries for data and result made available
- Big Data domains
    - Health-care
    - Retails
    - Trading/Share market
    - Finance
    - Security
    - Search engines
    - Log Analysis
    - Telecom
    - Traffic Control
    - Manufacturing and lot more.
- Big Data is all about
    - Think
    - Collect
    - Manage
    - Analyze
    - Summarize
    - Visualize
    - Discover Knowledge
    - Take Decisions
- Big Data Job profiles
    - Database engineer / DWH
    - Big Data engineer
    - IT operations / Data ops
    - ETL engineer (pipeline)
    - Big Data Architect

Distributed Systems

- Most of big data frameworks are distributed systems.

- Distributed system contains set of computers connected in a network (e.g. LAN). It is also referred as cluster. Each computer in cluster is referred as a node.
- Distributed systems provides
    - High availability, Fault tolerance, Rich computing, High memory.
    - High scalability (Horizontal scaling), Load balancing.
- There are two prime components of distributed system
    - Distributed storage
    - Distributed computing
- Major challenges for distributed systems
    - Node failure
    - Network failure
    - Distributed synchronization

Distributed Storage

- Each file have data (contents) and metadata (info).
- Information about data blocks is stored into metadata. To read the file first metadata is accessed and then data blocks. To write data blocks are updated and metadata as well.
- Files are organized into file systems. File systems arrange file's data blocks and inodes in systematic manner for efficient storage and access.
- In distributed file system, data blocks and metadata can be scattered on multiple nodes in the cluster. This improves the processing speed of the data.
- However what if any node is failed (containing data) or metadata node is failed?
- DFS gracefully handle these concerns using replication and/or backup node features.

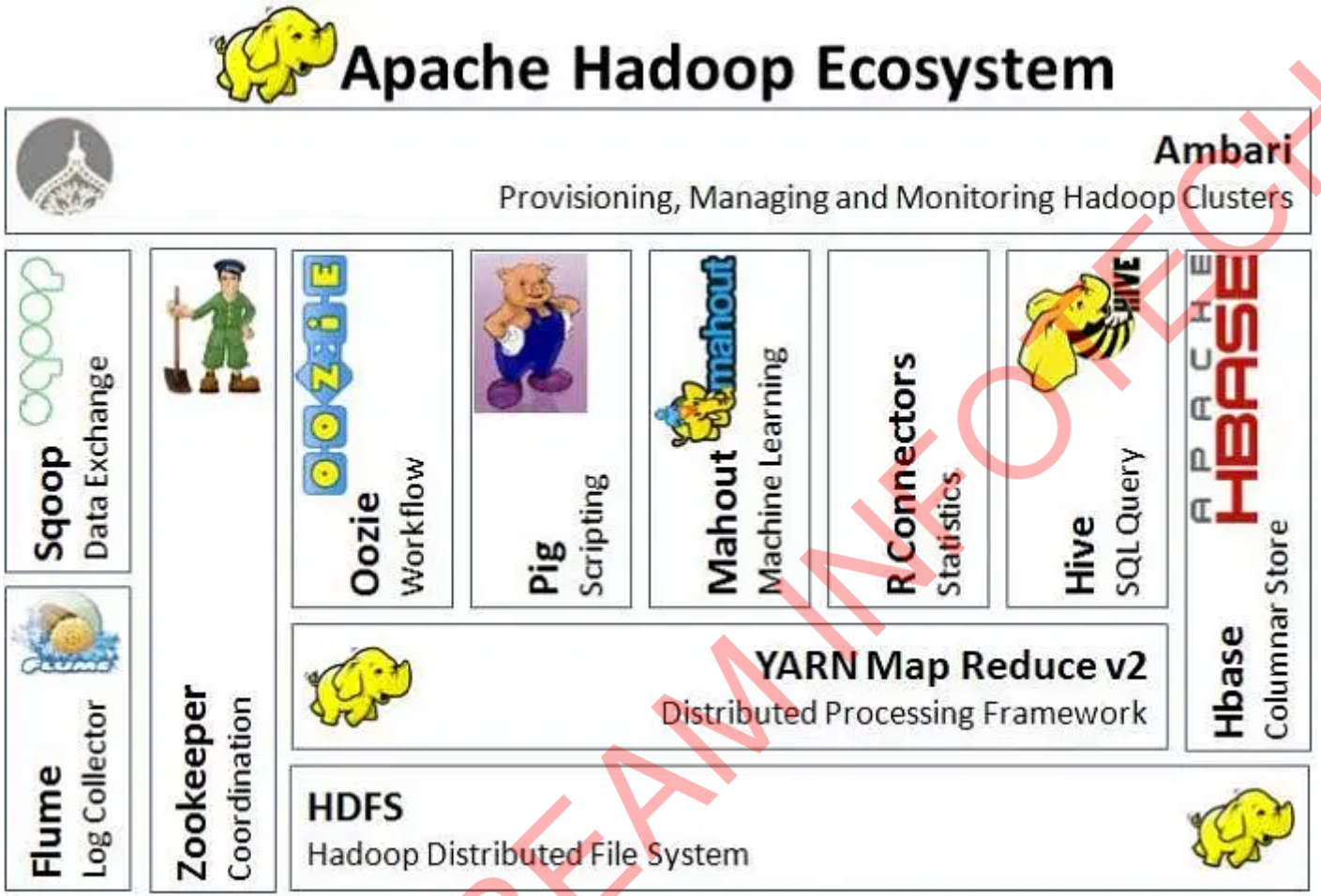
Distributed Computing

- Traditionally program loads data to be processed from the source and perform operations on it.
- This approach in not suitable for Big Data, considering data size and read/write speed of storage.
- Since data is stored on multiple nodes (distributed storage), program is also executed on multiple nodes processing partial data. These partial results are collected on a node and processed to yield final result.
- Distributed computing follows map-reduce design pattern.
    - Map stage process each record individually.
    - Reduce stage performs aggregation operation.

- Where does individual nodes process the data in memory or on disk? What if any node fails?

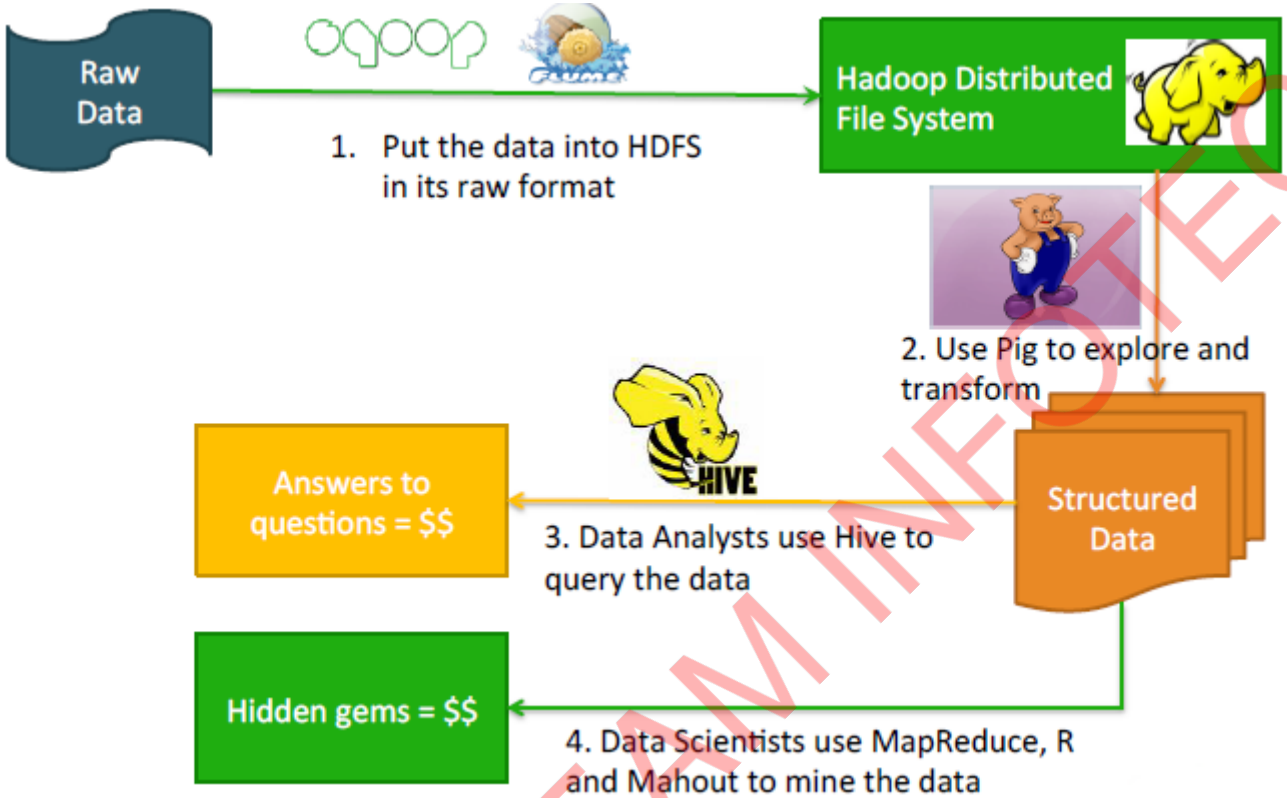
Apache Hadoop

- Hadoop Development
  - Developed by - Doug cutting - Developer of Nutch (Web crawler)
  - Distributed computing and storage needed to process huge data produced by the crawler.
  - Joined Yahoo. Developed Hadoop with Yahoo team.
  - Hadoop 0.1.0 is released in April 2006.
  - Hadoop open sourced under Apache license.
  - Development of Hadoop is inspired from Google white-papers on GFS (2003) & MapReduce (2004).
  - Hadoop is implemented in Java.
  - Hadoop is named after Doug Cutting kid's toy elephant.
- Hadoop major components
  - Distributed storage: HDFS
  - Distributed computing: MapReduce
- Hadoop is batch processing framework.
- Hadoop is like a Kernel/Platform on which many different applications are built (eco-systems).
- Hadoop Eco-System
  - HBase -- Columnar NoSQL database for high-speed searching.
  - Sqoop -- RDBMS to Hadoop data transfer and vice-versa.
  - Flume -- Live/streaming data ingestion into Hadoop.
  - Pig -- Data processing using Pig Latin langauge.
  - Hive -- SQL execution/RDBMS on Hadoop.
  - Impala -- SQL execution/RDBMS on Hadoop -- Low latency.
  - Oozie -- Job scheduler.
  - ZooKeeper -- Distributed synchronization and coordination service.
  - Spark -- Distributed computing framework.



- Hadoop Distributions
  - Cloudera + Hotonworks
  - AWS EMR
  - MapR
- Hadoop Eco-System Dataflow
  - Extract data from sources (RDBMS or Live streaming)
  - Load (raw) data into HDFS.
  - Data is processed, transformed and/or summarized to convert into structured (tabular) format.
  - This structured data is loaded into the Hive.

- Hive tables can be queried to analyse the data.
- ML algorithms can be used on the data to get insights and/or predictions.



Hadoop Distributed File System

- HDFS is fault tolerant, redundant distributed file system.
- HDFS is "Write Once Read Multiple times" file system.
- It is implemented following white-paper on Google File System.
- HDFS has three components
  - Name Node - Manage file metadata.
  - Data Node - Manage files data.
  - Secondary Name Node - Metadata backup.
- HDFS stores file's data into data blocks. Size of data block is 64 MB or 128 MB.

- Each HDFS block is replicated on 3 nodes (while write operation). It ensures that if any node fails, data can be taken from some replica node.
- The metadata backup is maintained on secondary name node. In case of name node failure, metadata can be retrieved from secondary name node. This makes HDFS fault-tolerant.

Hadoop Map-Reduce

- Hadoop MR is implemented following Google's white paper MapReduce: Simplified Data Processing on Large Clusters.
- MR job processes the data stored in HDFS.
    - Mapper -- Process individual record. Typical processing includes data cleaning, transformation, filtering, etc.
    - Reducer -- Process group of records. Typical processing includes aggregation operations e.g. sum, average, standard deviation, etc.
- Hadoop MR job is implemented in Java or other programming language (using Hadoop streaming).
- This MR job is submitted to Hadoop cluster (via HDFS) and its mapper and reducer components are scheduled to execute on multiple nodes in the cluster.
- Hadoop MapReduce execution changed drastically across Hadoop 1.x and Hadoop 2.x.
- Hadoop 2.x has two components
    - ResourceManager - Cluster manager (Resource Negotiator)
    - NodeManager - Manage executions on each node
- For each application, one MRAppMaster process is created that tracks application execution and number of mappers/reducers are executed by YarnChild processes.

Java Pre-requisites

- JDBC
- Stream programming -- Lambda expressions, map(), filter(), reduce(), sorted(), flatMap(), etc.
- Java OOP -- class, object, extends, implements.
- Java generics, collections (Iterable, Key-Value pairs)
- Java File IO (File, DataInputStream, DataOutputStream)