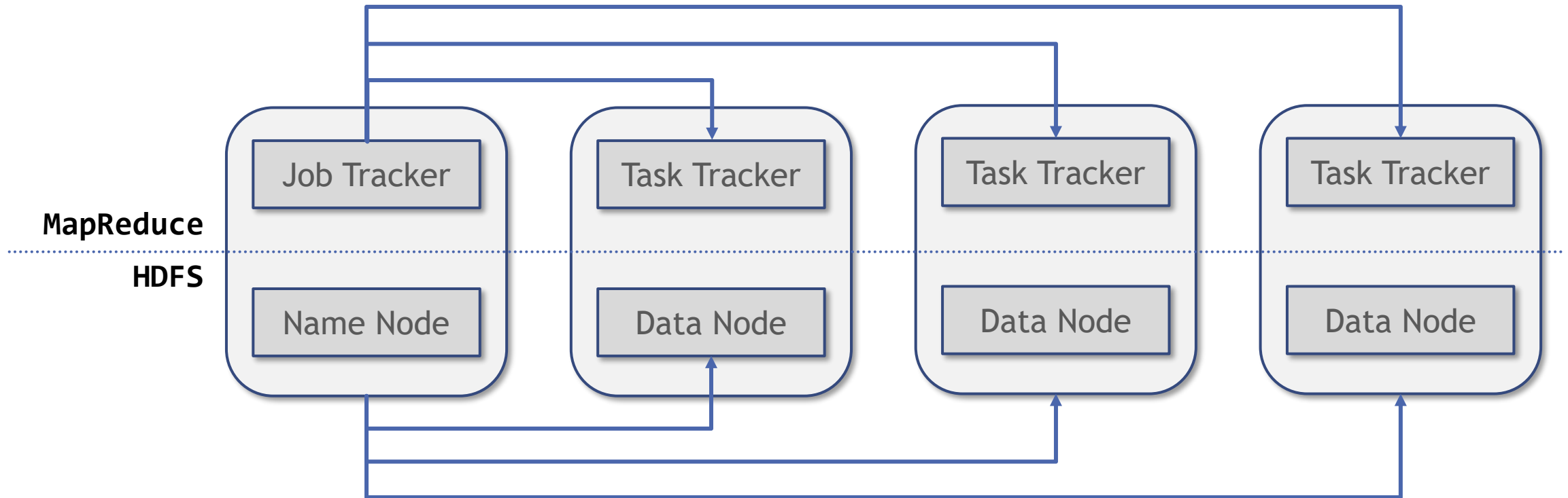


# Hadoop 기반 빅데이터 시스템

# Hadoop

## ▶ 하둡이란?

- ▶ 대용량 데이터 처리를 위해 개발된 오픈 소스 플랫폼
- ▶ 구성 요소
  - ▶ HDFS(Hadoop Distributed File System, 분산 파일 시스템) : 대용량 파일의 분산 저장
  - ▶ MapReduce(MapReduce, 분산 처리 시스템) : 분산 저장된 파일을 이용, 데이터를 처리



# Hadoop

## : HDFS(Hadoop Distributed File System)

- ▶ **HDFS** : 대용량 파일을 여러 서버에 나누어 저장하고 다양한 클라이언트가 빠르게 처리할 수 있도록 설계된 파일 시스템
  - ▶ 저사양 컴퓨터를 여러 대 연결하여 대용량의 스토리지를 구성하고 하나의 서버처럼 사용할 수 있도록 지원
  - ▶ **HDFS**가 제공하는 다양한 **API**를 활용하여 파일을 읽고 저장할 수 있다
  - ▶ 장애 복구, 대용량 파일 저장, 데이터 무결성 등을 지원
    - ▶ 복제본을 여러 서버에 함께 저장 -> 데이터의 유실을 방지
    - ▶ 분산 서버의 주기적인 상태 체크 -> 장애를 인지하고 대처할 수 있음
    - ▶ 한 번 저장된 파일은 수정할 수 없고 읽기만 가능 -> 데이터의 무결성 유지

# Hadoop

## : HDFS(Hadoop Distributed File System)

### ▶ 블록 구조의 파일 시스템

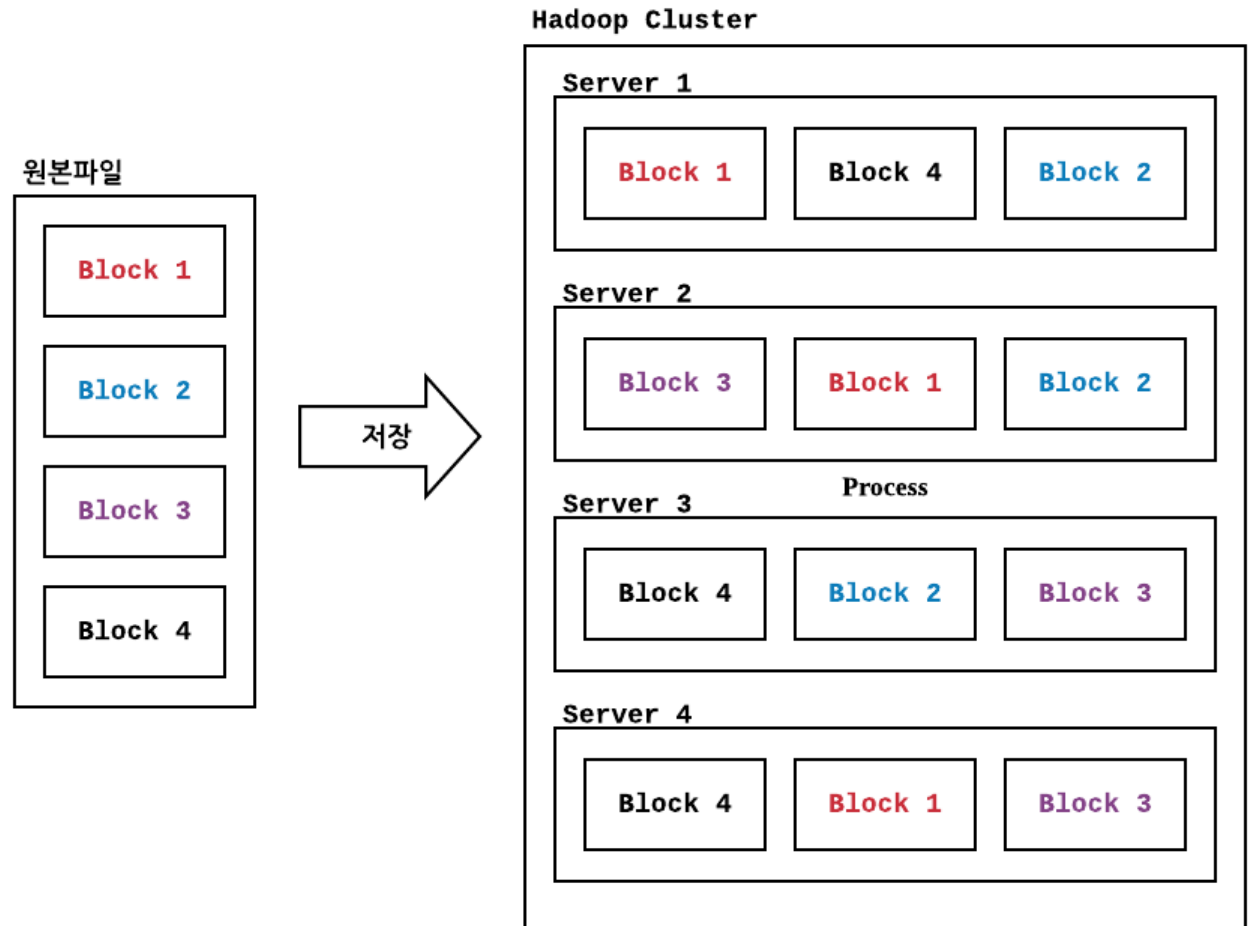
- ▶ 파일은 일정한 크기의 블록으로 분할되어 복수 개의 복제본을 분산된 서버에 나누어 저장

### ▶ 네임 노드(Name Node)

- ▶ HDFS 시스템을 유지하기 위한 메타데이터를 관리하고 유실방지를 위한 모든 변경 이력을 저장
- ▶ 데이터 노드에 장애가 발생하면 다른 데이터 노드에 블록을 복제, 항상 같은 수의 복제본을 유지

### ▶ 데이터 노드(Data Node)

- ▶ HDFS에 저장되는 파일이 블록으로 나누어져 실제 저장되는 장소

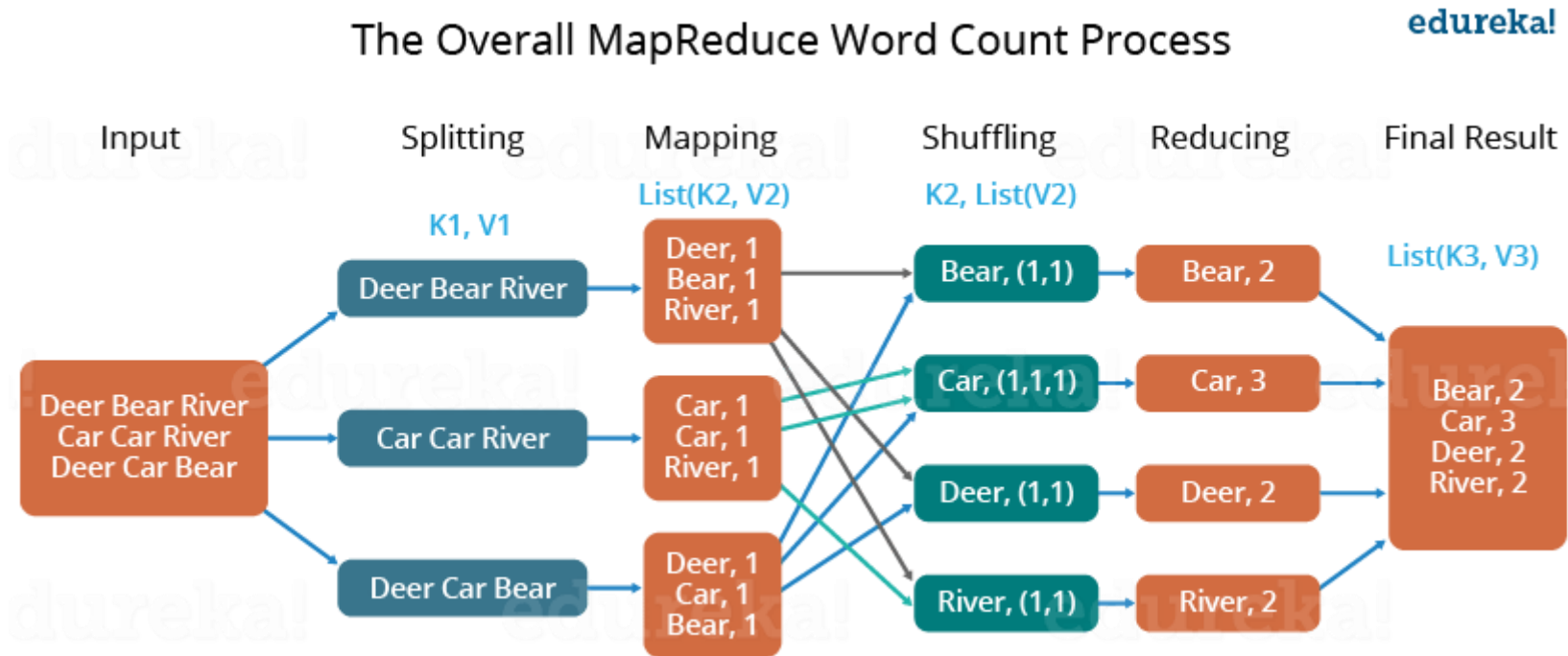


# Hadoop

## : MapReduce

- ▶ 입력 데이터에 대해 최종 결과를 얻기까지 총 4단계로 나누어 동작

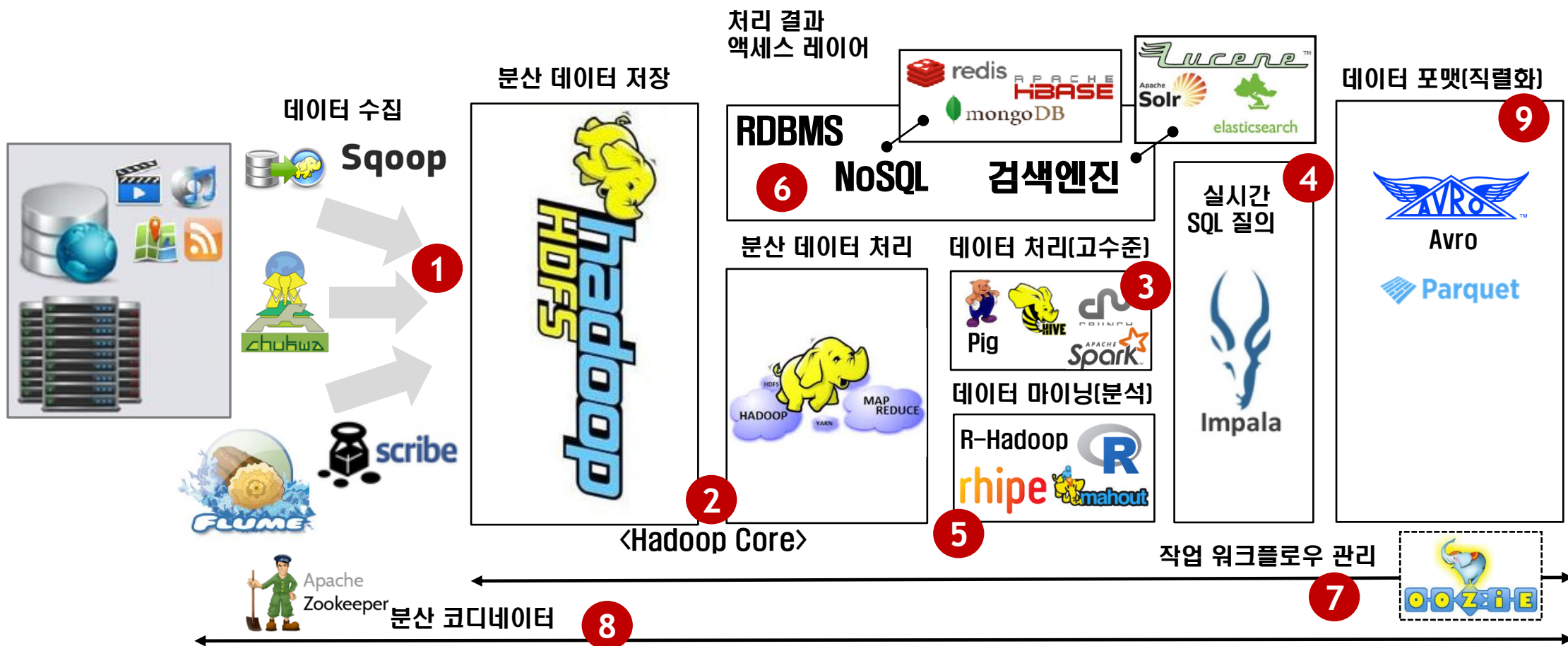
분할(Split) -> 매핑(Mapping) -> 셔플(Shuffle)/정렬(Sort) > 리듀스(Reduce)



# Hadoop Ecosystem

## ▶ Hadoop Ecosystem

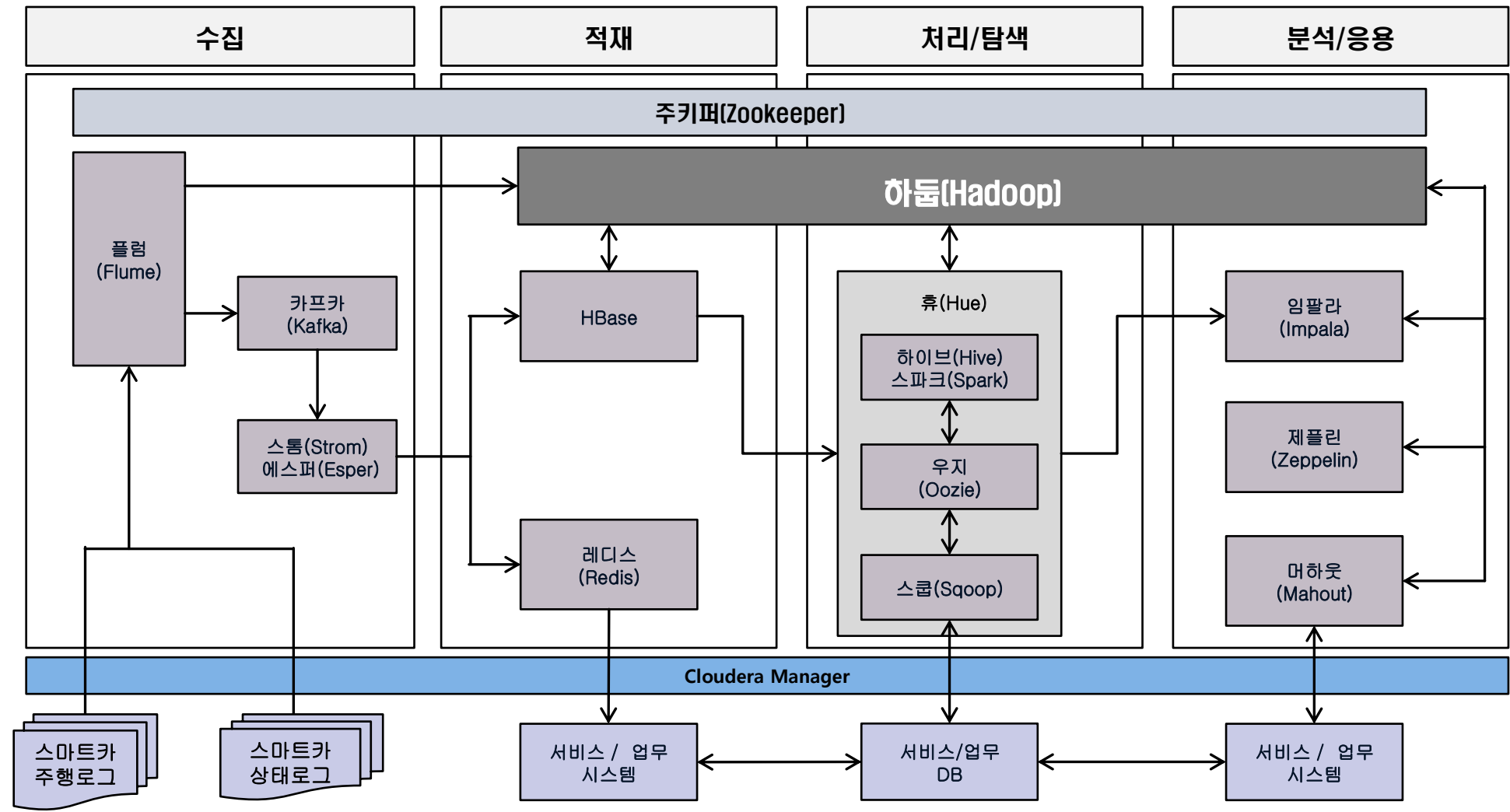
- ▶ 하둡의 기본적인 HDFS와 MapReduce에 여러 오픈 소스 프로젝트를 추가, 하둡 기반의 폭넓은 플랫폼으로 제공
- ▶ 비즈니스 사용 측면에서 요구되는 프로젝트들의 집합



# Hadoop Ecosystem

: 아키텍처 구성 사례

▶ Hadoop Ecosystem을 활용한 아키텍처 구성 사례



# Hadoop Ecosystem

## : 수집 레이어

- ▶ 데이터 수집
  - ▶ 빅데이터 시스템에서 가장 중요
  - ▶ 빅데이터 시스템의 궁극적 목표는 수집된 데이터를 **HDFS** 등의 분산 파일 시스템에 저장하는 것
- ▶ Chukwa
  - ▶ 분산 환경에서 생성되는 데이터를 **HDFS**에 안정적으로 저장하는 플랫폼
  - ▶ 분산된 각 서버에서 **Agent**를 실행하고, **Collector**가 에이전트로부터 데이터를 받아 **HDFS**에 저장
  - ▶ **Collector**는 100개의 에이전트당 하나씩 구동
  - ▶ 데이터 중복 제거 등의 작업은 **MapReduce**로 처리
  - ▶ **Yahoo!**에서 개발, 현재는 아파치 인큐베이션에 포함





# Hadoop Ecosystem

## : 수집 레이어

### ▶ Flume

- ▶ Chukwa처럼 분산된 서버에 에이전트가 설치되고 에이전트로부터 데이터를 전달받는 콜렉터로 구성
- ▶ 전체 데이터의 흐름을 관리하는 마스터 서버가 있어서 데이터를 어디서 수집하고 어떤 방식으로 전송하며 어디에 저장할지를 동적으로 변경 가능
- ▶ 클라우드에서 개발, 현재는 아파치 인큐베이션에 포함



### ▶ Scribe

- ▶ 페이스북에서 개발한 데이터 수집 플랫폼
- ▶ Chukwa와는 다르게 데이터를 중앙 집중하여 서버로 전송하는 방식
- ▶ 최종 데이터는 HDFS 외에 다양한 저장소를 활용할 수 있으며, 설치와 구성이 쉽고 다양한 프로그래밍 언어를 지원
- ▶ HDFS에 저장하기 위해서는 JNI(Java Native Interface)를 이용해야 한다



### ▶ Sqoop

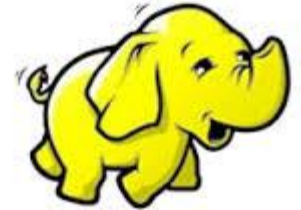
- ▶ 대용량 데이터 전송 솔루션이며 아파치 Top Level 프로젝트
- ▶ HDFS, RDBMS, DW, NoSQL 등 다양한 저장소에 대용량 데이터를 신속하게 전송할 수 있는 방법을 제공
- ▶ Oracle, MS-SQL, DB2 등 상용 RDBMS와 MySQL, PostgreSQL 등 오픈 소스 RDBMS를 지원



# Hadoop Ecosystem

## : 저장 레이어

- ▶ 데이터 저장 처리
  - ▶ 모인 데이터들을 저장하고 처리하는 역할을 담당
  - ▶ 빅데이터 시스템의 핵심이라 할 수 있는 하둡(Hadoop)
  - ▶ 하둡의 두 가지 구성 요소
    - ▶ HDFS : 분산 파일 시스템
    - ▶ MapReduce : 분산 처리 시스템
  - ▶ MapReduce 프로그래밍을 위해서는 C++, Java를 사용할 수 있다
  - ▶ MapReduce 개념 없이 고수준 프로그래밍을 하기 위해서는 Hive, Pig 등의 언어를 사용할 수 있다
  - ▶ 하둡은 기본적으로 큰 데이터를 배치 프로세싱 하는 데 적합하고 실시간 데이터 처리에는 다소 적합하지 않다



**hadoop**



**hadoop  
MapReduce**



**hadoop  
YARN**

# Hadoop Ecosystem

## : 데이터 처리(고수준)

### ▶ Pig

- ▶ 야후에서 개발되었으며 현재 아파치 프로젝트에 속해 있음
- ▶ 복잡한 MapReduce 프로그래밍을 대체할 Pig Latin이라는 자체 언어 제공
- ▶ MapReduce API를 매우 단순화시키고 SQL과 유사한 형태로 설계



### ▶ Hive

- ▶ 하둡 기반 데이터웨어하우스 하우징용 솔루션
- ▶ 페이스북에서 개발, 오픈 소스로 공개되며 주목받은 기술
- ▶ SQL과 매우 유사한 HiveQL이라는 쿼리 언어를 제공
- ▶ Java를 모르는 데이터 분석가들도 쉽게 하둡 데이터를 분석
- ▶ HiveQL은 내부적으로 MapReduce로 변환되어 실행



### ▶ Impala

- ▶ 클라우드에서 개발한 하둡 기반 실시간 SQL 질의 시스템
- ▶ 맵리듀스를 사용하지 않고, 자체 개발한 엔진을 사용 성능이 빠르다
- ▶ 데이터 조회를 위한 인터페이스로 HiveQL을 사용
- ▶ 수초 내에 SQL 질의 결과를 확인할 수 있으며 Hbase와도 연동이 가능



# Hadoop Ecosystem

## : 데이터 마이닝(분석)

### ▶ 데이터 마이닝(분석)

- ▶ 대용량 데이터에서 특정 패턴을 찾기 위한 과정
- ▶ 데이터에서 어떤 의미를 찾아서 가치를 생성하는 과정
- ▶ 데이터 사이언스(통계/수학) 기법이 활용된다

### ▶ R

- ▶ 통계 계산과 그래픽(데이터 시각화)을 위한 프로그래밍 언어 혹은 소프트웨어 환경
- ▶ 통계 소프트웨어 개발과 자료 분석에 널리 활용되며 통계학자들이 통계 분석 모델 개발에 많이 활용

### ▶ R-Hadoop

- ▶ R 인터페이스를 하둡에서 제공해 주는 프로젝트
- ▶ R-MR, R-HDFS, R-Hbase 등 하둡 기반 기술들에 R을 접목한 프로젝트들도 주목



# Hadoop Ecosystem

## : 처리 결과 액세스 레이어

- ▶ 처리 결과를 외부에서 실시간으로 액세스 해야 하는 경우, 다음 세 가지 정도의 기술을 고려해 볼 수 있다.
  - ▶ **RDBMS**
    - ▶ 처리 결과 데이터가 상대적으로 작아 별다른 검색이 필요하지 않은 경우
    - ▶ Oracle, MySQL 등에 테이블을 정의하고 처리 결과를 삽입한다
    - ▶ Sqoop을 조합할 수 있다
  - ▶ **NoSQL**
    - ▶ 보통 처리 결과 데이터의 크기가 크며 스키마가 고정되어 있지 않은 비정형 데이터의 처리
    - ▶ RDBM에 맞지 않는 데이터일 수 있으며 요청 트래픽이 클 가능성도 있음
    - ▶ 이럴 경우, Hbase, Cassandra, MongoDB 등의 NoSQL을 활용
    - ▶ NoSQL 데이터베이스들은 하둡에서 바로 인덱스를 만들어낼 수 있어 하둡과 연동에 적합
  - ▶ **검색엔진**
    - ▶ 하둡에서 처리된 결과를 키/밸류 형태보다 더 복잡한 형태로 접근해야 하는 경우 활용
    - ▶ 자바 기반 Lucene과 이를 기반으로 한 Solr 패키지를 주로 활용/개발
    - ▶ Lucene을 기반으로 한 Elasticsearch 검색 엔진은 분산 환경을 고려해 개발된 검색 엔진

# Hadoop Ecosystem

## : 작업 워크플로우 관리

### ▶ 작업 워크플로우 관리

- ▶ 대부분 하둡의 작업들은 여러 개의 잡(Job)들을 연속적으로 이어 붙여 실행
- ▶ 이런 잡(Job)들을 주기적으로 실행하거나 어떤 조건이 만족되면 실행되도록 관리해주는 소프트웨어

### ▶ Oozie

- ▶ 하둡 작업을 관리하는 워크플로우 및 코디네이터 시스템
- ▶ 자바 서블릿 컨테이너에서 실행되는 자바 웹 어플리케이션
- ▶ MapReduce 작업이나 Pig 작업 등 특화된 액션들로 구성된 워크플로우를 제어



# Hadoop Ecosystem

## : 분산 코디네이터

### ▶ 분산 코디네이터

- ▶ 분산 환경에서 서버들 간 상호 조정이 필요한 다양한 서비스를 제공하는 시스템

### ▶ Zookeeper

- ▶ 하나의 서버에만 서비스가 집중되지 않도록 서비스를 분산하여 동시 처리하도록 해 줌
- ▶ 하나의 서버에서 처리한 결과를 다른 서버들과도 동기화하여 데이터의 안정성을 보장
- ▶ 운영(**Active**) 서버가 문제가 발생하여 서비스를 제공할 수 없을 경우, 다른 대기 중인 서버를 운영 서버로 교체하여 서비스가 중지 없이 제공되도록 해 줌
- ▶ 분산 환경을 구성하는 서버들의 환경 설정을 통합적으로 관리



# Hadoop Ecosystem

## : 데이터 포맷(직렬화)

### ▶ Avro

- ▶ RPC(Remote Procedure Call)과 데이터 직렬화를 지원하는 프레임워크
- ▶ JSON을 이용해 데이터 형식과 프로토콜을 정의하며 작고 빠른 바이너리 포맷으로 데이터를 직렬화





# Hadoop 기반 빅데이터 시스템

## : 사례

### ▶ Ebay 쿼리 로그 마이닝

- ▶ MapReduce 프레임워크
- ▶ 1억 명이 넘는 사용자, **PB** 단위의 클릭 로그 데이터, **10억** 개 이상 사용자 검색어 처리
- ▶ 현재 이슈가 되고 있는 검색어, 연관 검색어 찾아내기



### ▶ 페이스북 메시지 시스템

- ▶ HDFS 기반 NoSQL인 HBase 위에서 작동
- ▶ 세계에서 가장 많은 트래픽을 처리하는 시스템 중 하나
- ▶ 월 평균 **25TB** 데이터 처리



# Hadoop 기반 빅데이터 시스템

## : 사례

- ▶ 넷플릭스 영화 추천 서비스
  - ▶ 영화/드라마 스트리밍 서비스
  - ▶ 서비스 스트리밍 시간의 **80%**, 영화 감상이 영화 추천 시스템에 의한 것
  - ▶ **Markov Chain** 기반의 거대한 행렬 계산
  - ▶ 하둡 도입 이전에는 **MySQL**을 사용하여 장시간에 걸쳐 계산
    - ▶ 데이터 목사 시간, 최적화를 위한 여러 데이터베이스 기술이 필요
  - ▶ 하둡 시스템으로 교체 후 단시간 계산으로 **1일 1회**에 가능해짐



# Hadoop 기반 빅데이터 시스템

## : 사례

### ▶ 트위터

- ▶ 1억 5천만이 넘는 사용자, 3억 4천만 개 트윗의 방대한 데이터를 처리하기 위해 수십 페타바이트 규모의 하둡 시스템을 활용
- ▶ Pig를 사용
- ▶ 팔로우 형태 분석, 정서 분석 등을 위해 하둡 기반 머신 러닝 시스템을 구축하여 사용



### ▶ 기타 활용 사례

- ▶ 추천 시스템
  - ▶ 예) 링크드인(알 수도 있는 사람), eHarmony(파트너 추천 시스템)
  - ▶ 검색 서비스(크롤링, 인덱스 생성, 페이지 랭크 계산 등)

# 빅 데이터 시스템 도입시 고려할 점

## ▶ ROI(Return On Investment) 고려

- ▶ 빅데이터 처리 시스템을 만드는 것은 많은 비용과 시간이 필요하다
- ▶ 왜 하는지, 무엇을 할 것인지, 수익성이 있을지 등을 우선 고려
- ▶ 아마존 등 클라우드 서비스를 먼저 사용해서 가능성을 타진하고 시스템을 구축하는 것도 추천

## ▶ 오픈 소스로 구축된 시스템

- ▶ 소스가 완전히 공개되기 때문에 보안에 취약할 수 있다
- ▶ 변경이 빠르기 때문에 프로젝트들 간 호환성에 문제가 발생할 수 있다(버전 충돌)
- ▶ 지원을 보장받을 수 없다

## ▶ 대용량 데이터의 중앙 수집

- ▶ 빅데이터를 한 군데로 모으는 것에 어려움이 있을 수 있다
- ▶ 빅데이터 시스템 도입 전에 데이터가 어디에 어떤 형태로 저장되고 있는지 확인 작업에 많은 노력이 필요하다
- ▶ 중앙에 수집하기 위해 기존 시스템의 변경이 필요한 경우도 있다