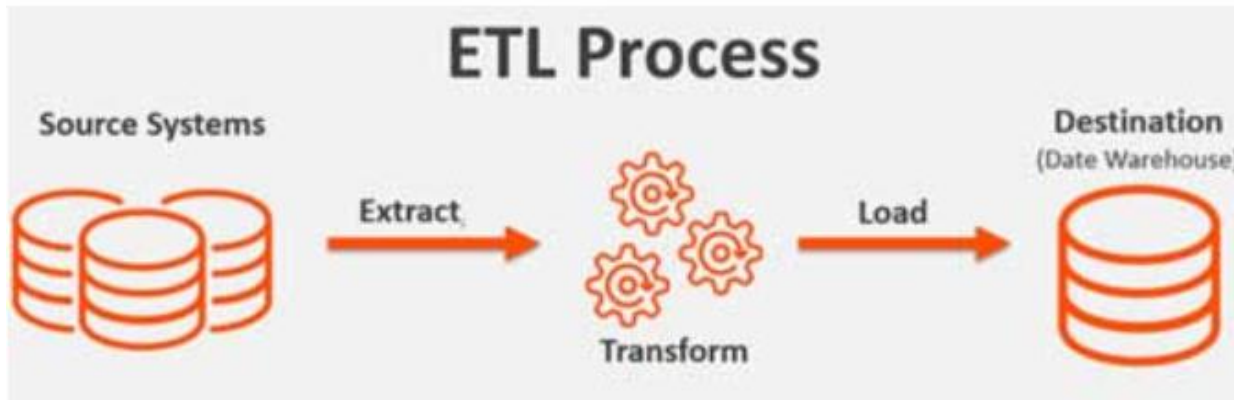


Data Integration

빅데이터

ETL



핵심은 데이터 품질 :

데이터 사이언티스트들이 작업시간의 80% 를 데이터 추출 및 정제 작업에 투자
원천 데이터를 단순히 수집하여 바로 데이터 시스템에 적재해서는 안됨

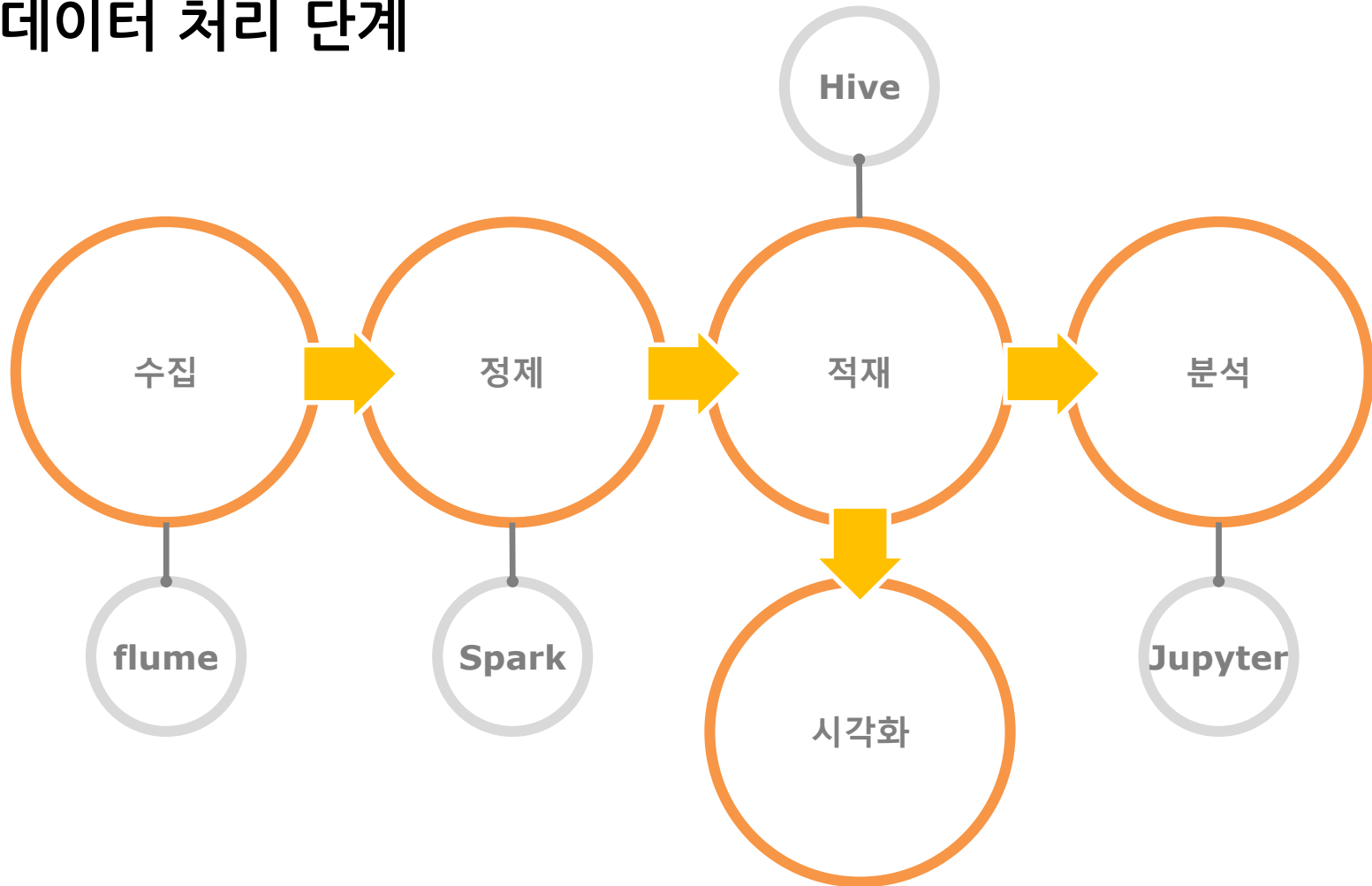
ETL :

가장 보편적이고 검증된 배치 처리 기술

빅데이터 처리 단계

빅데이터

빅데이터 처리 단계

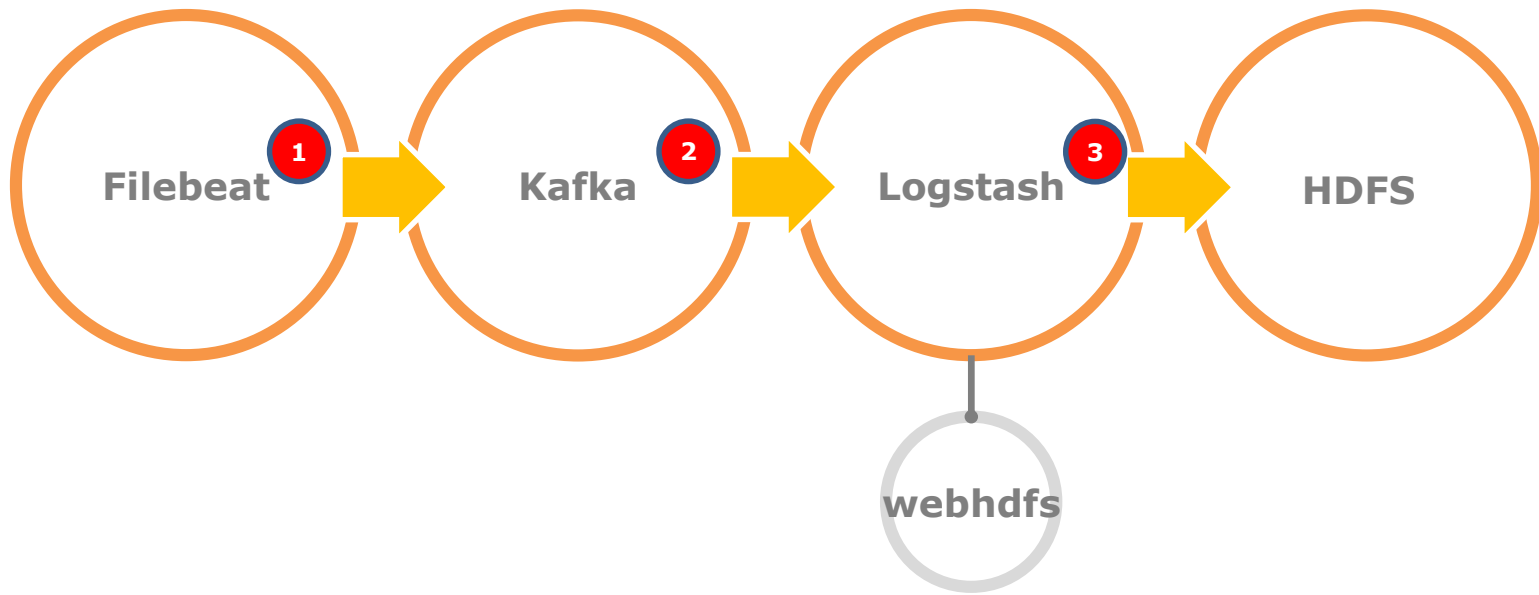


수집

빅데이터

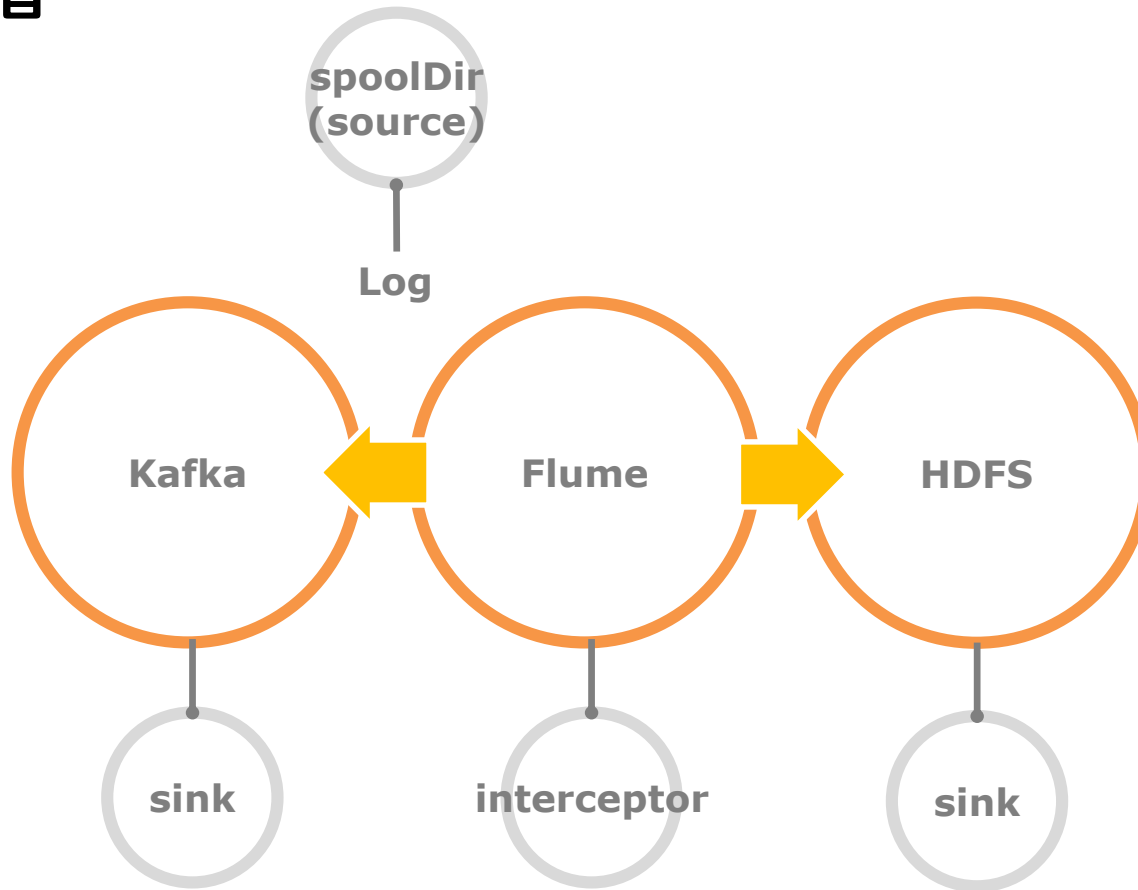
데이터 수집

Log



빅데이터

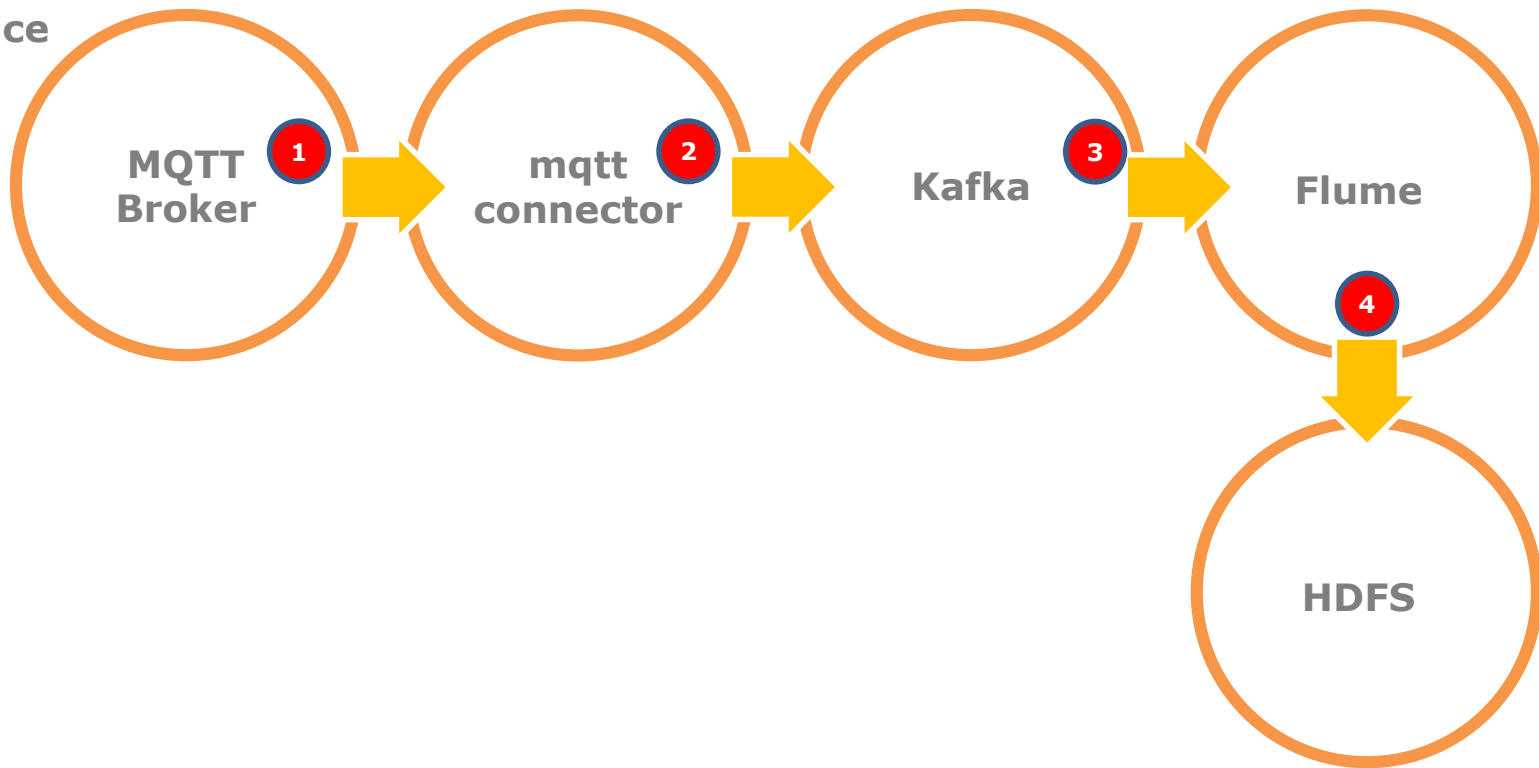
데이터 수집



빅데이터

데이터 수집

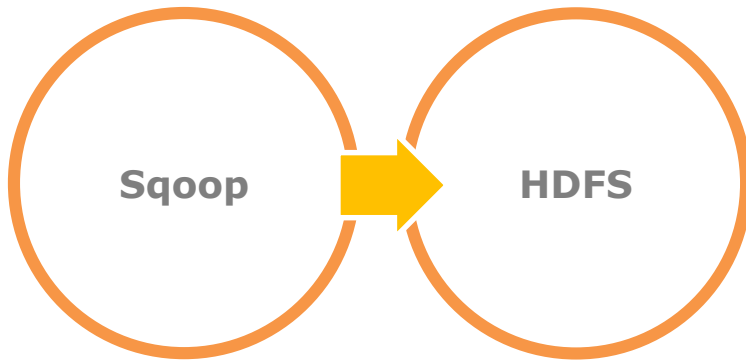
Device



빅데이터

데이터 수집

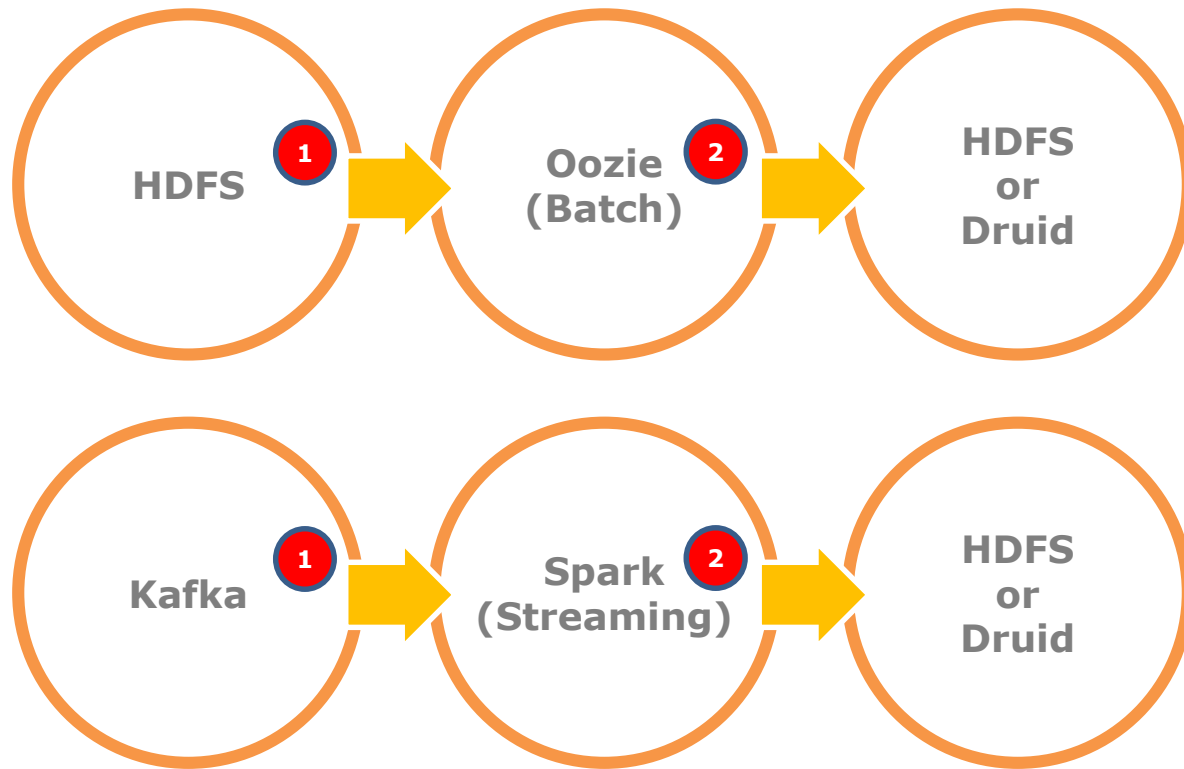
DB



정제, 적재

빅데이터

데이터 정제, 적재



분석

빅데이터

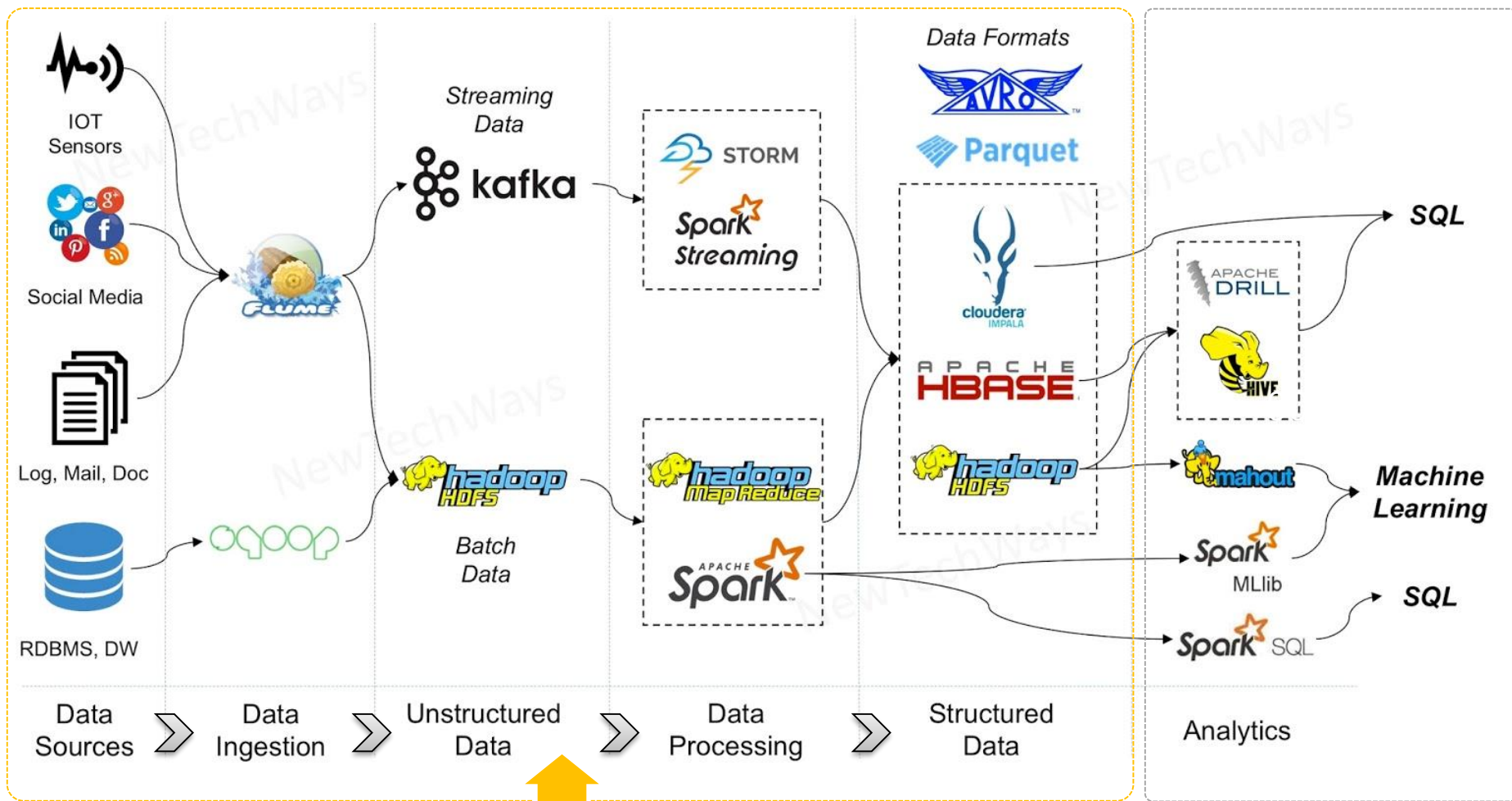
Notebook



Data Pipeline

빅데이터

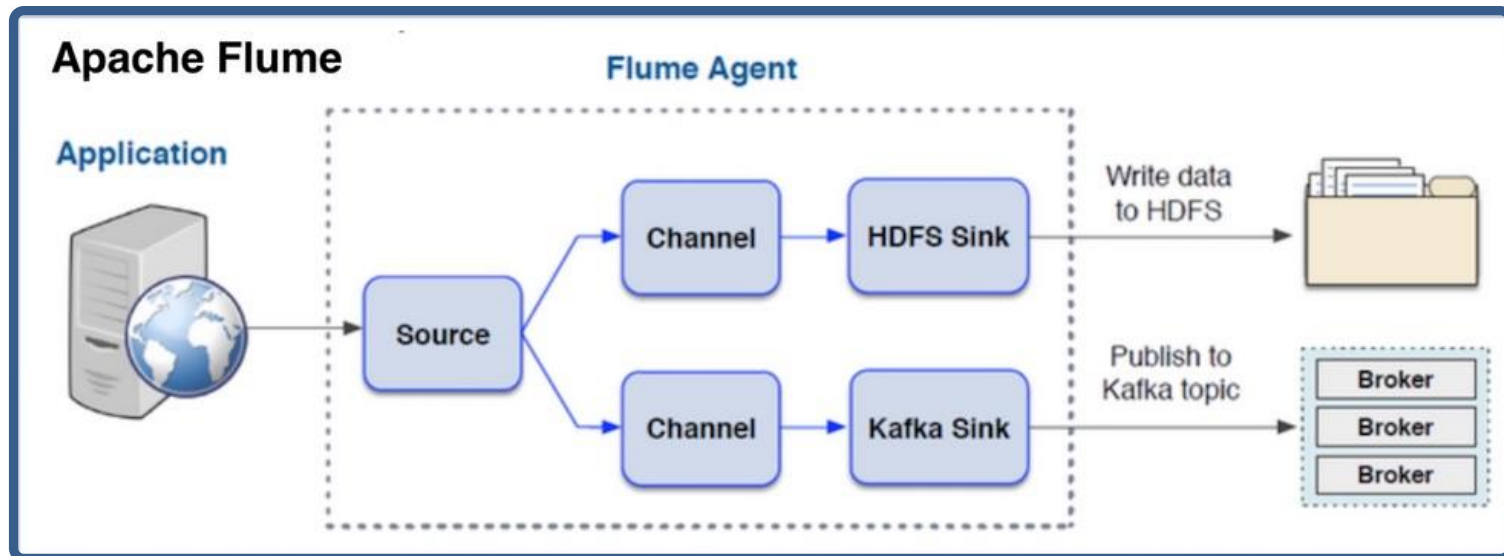
Data Pipeline



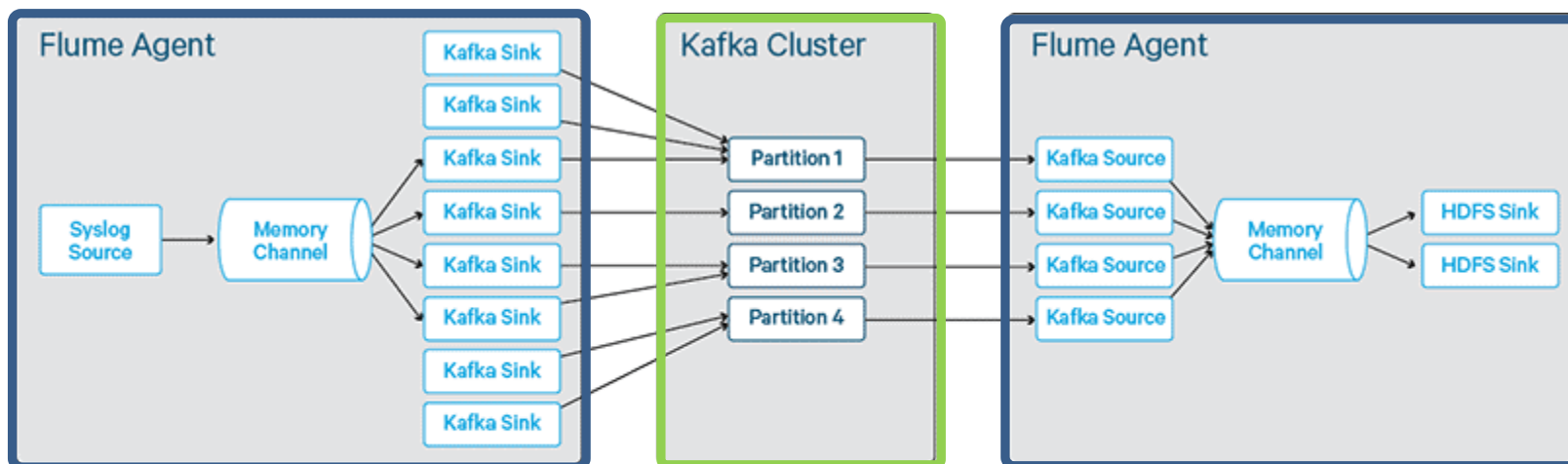
Apache Flume

빅데이터

Apache Flume



Apache Flume



4. The number of sinks must be some multiple of the number of partitions to ensure that there is a reasonable probability of even distribution of events across partitions and hence Kafka sources.

3. The number of partitions should be greater than the number of disks in the Kafka cluster divided by the replication factor. It MUST be greater than or equal to the number of Kafka sources.

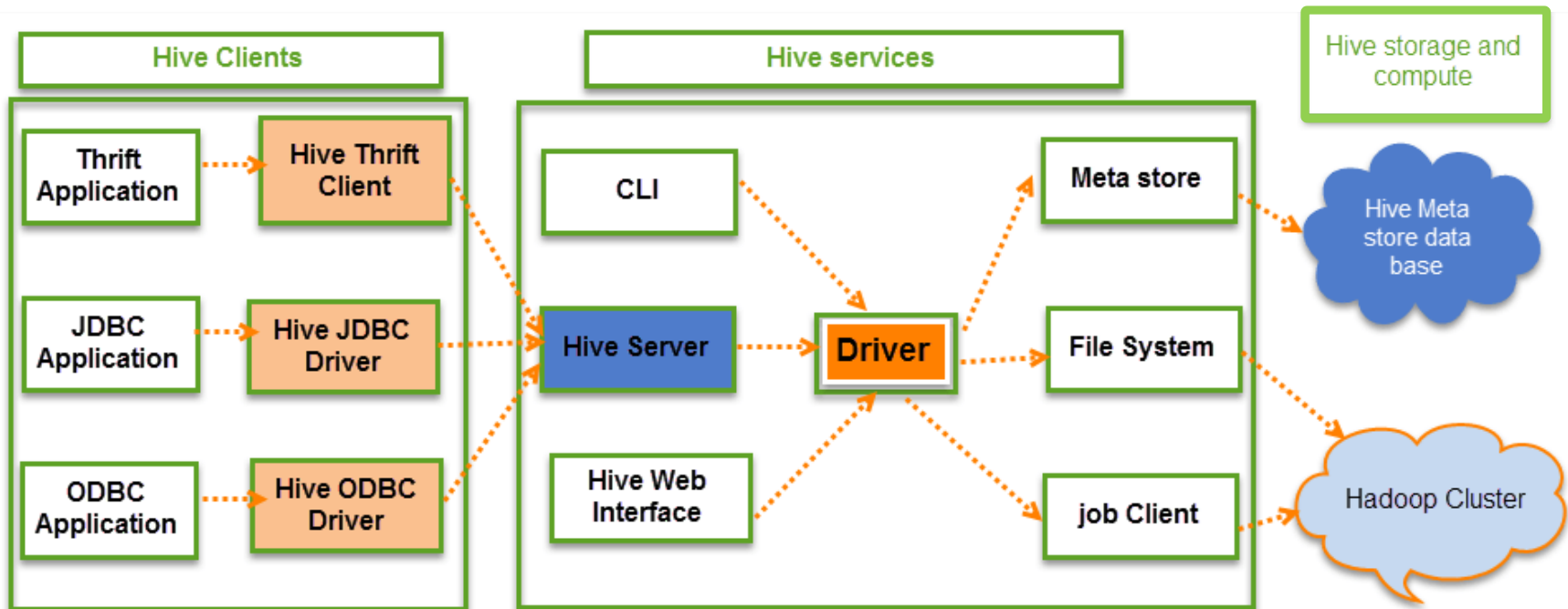
2. The number of Kafka sources may also be a bottleneck, so enough sources need to be provisioned for the throughput required.

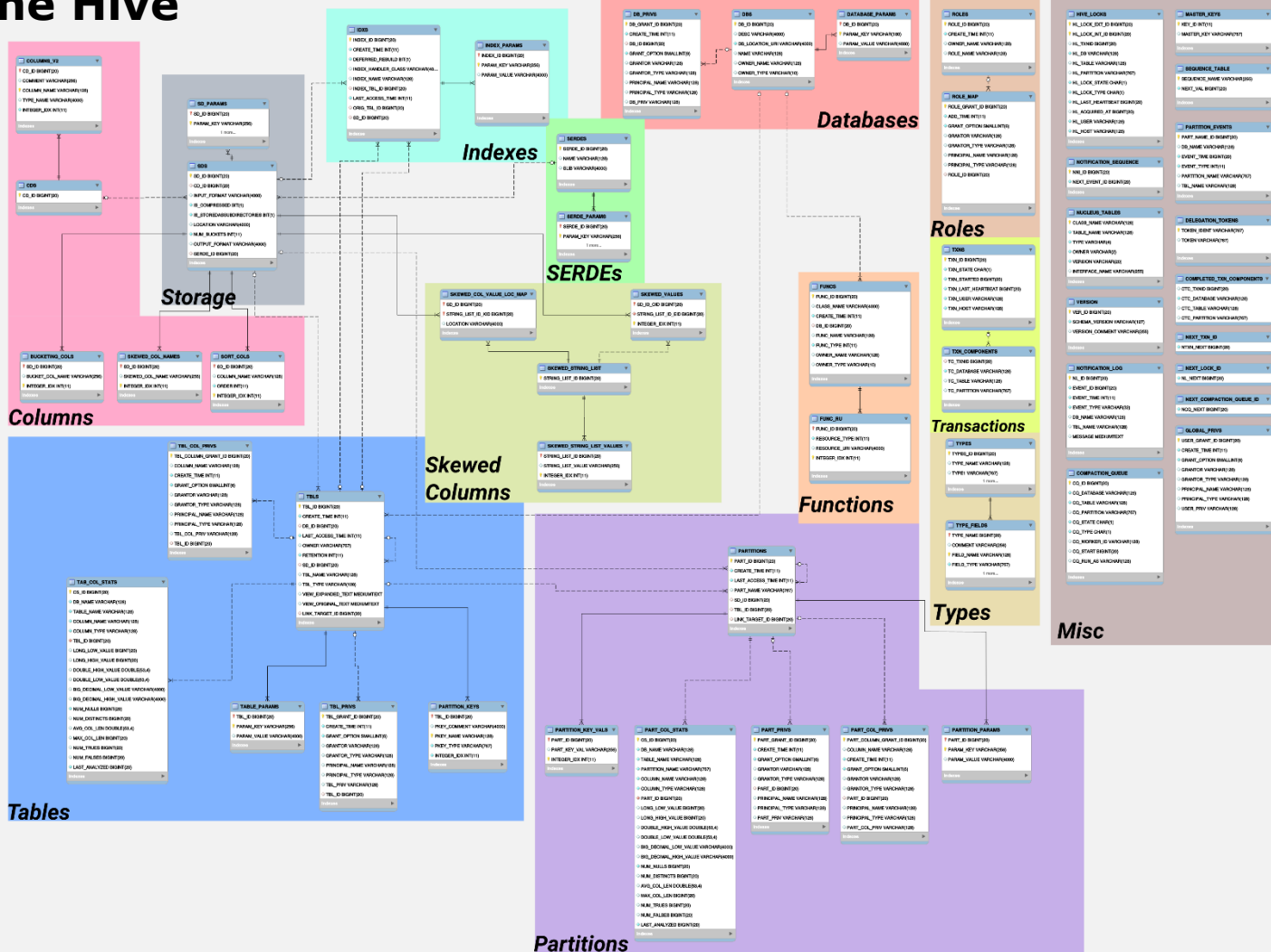
1. The number of HDFS sinks can be calculated from the throughput required and can be scaled independently.

Apache Hive

빅데이터

Apache Hive



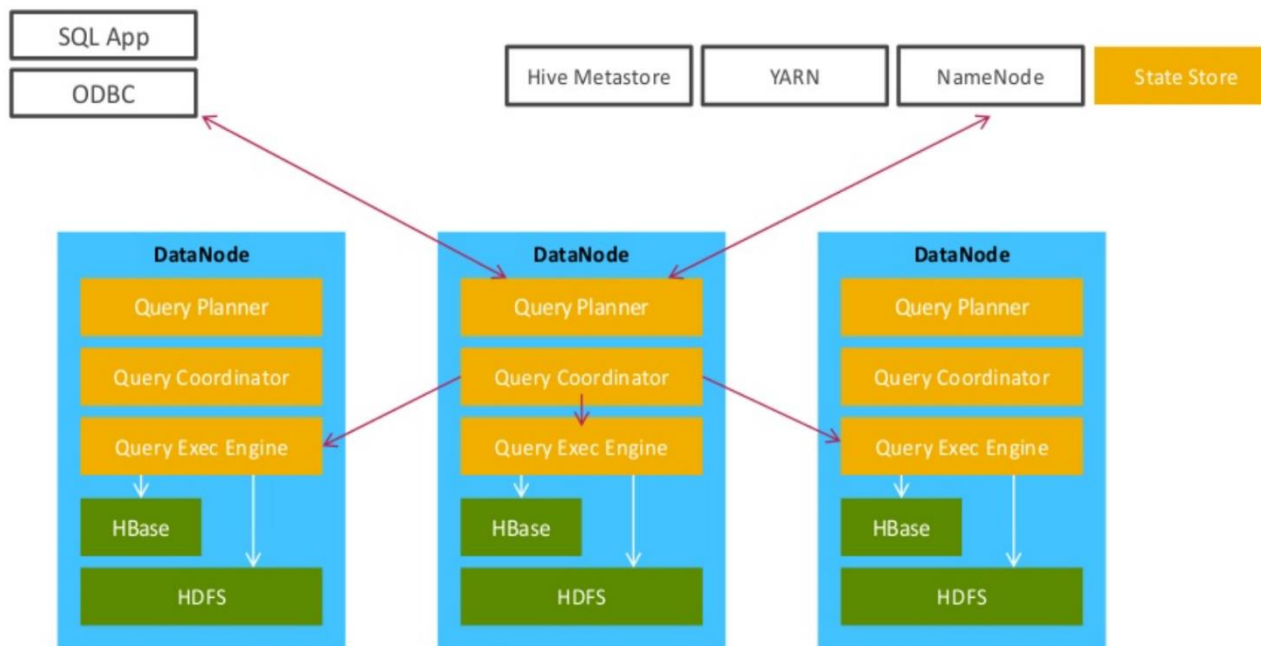


Apache Impala

Apache Impala



- Unified storage : supports HDFS and HBase, Flexible file formats
- Unified Metastore
- Unified security
- Unified client interface : ODBC, SQL syntax, Hue, Beeswax
- Impala: real time SQL queries, native distributive query engine, optimized for low-latency
- Answers as fast as you can ask
- Everyone to ask questions for all data, Big Data storage and analytics together



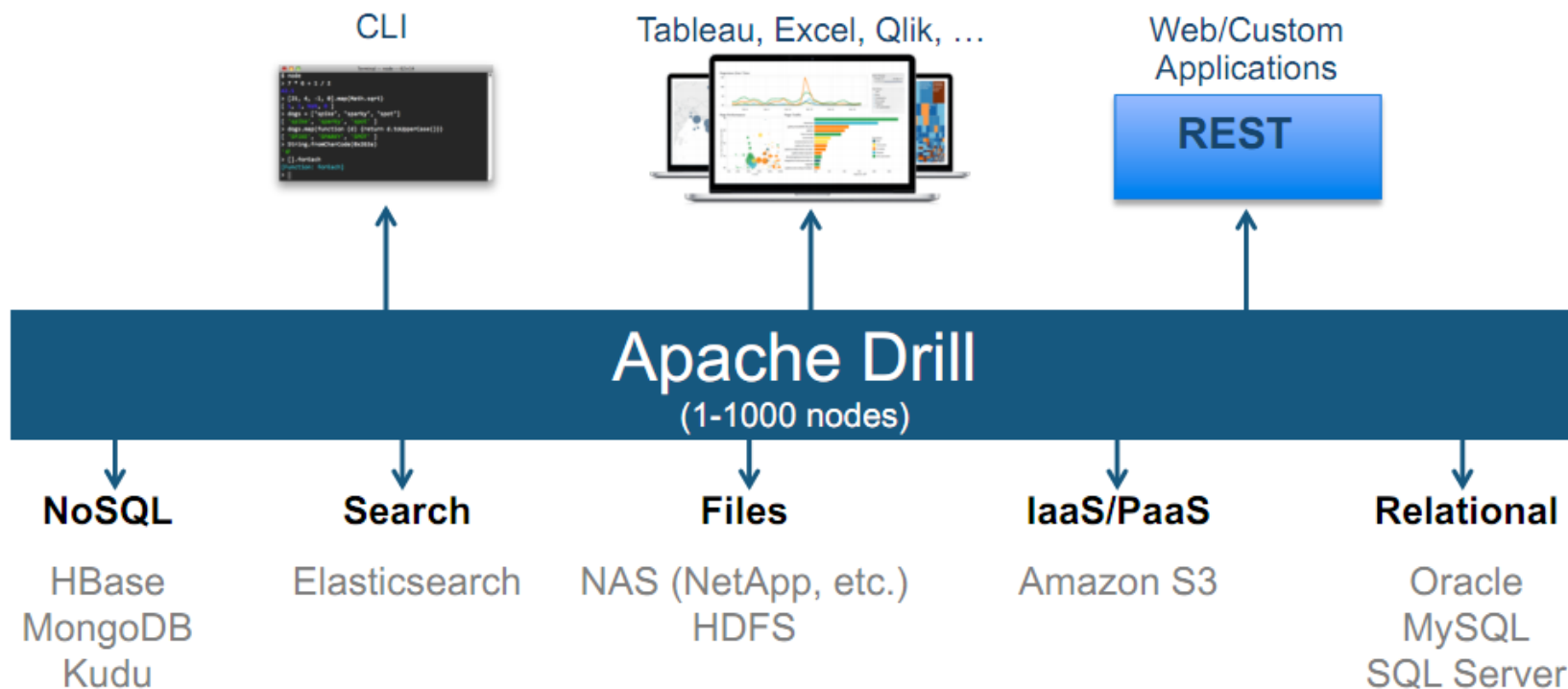
Source : <http://cloudera.com/content/cloudera/en/products-and-services/cdh/impala.html>

Apache Drill

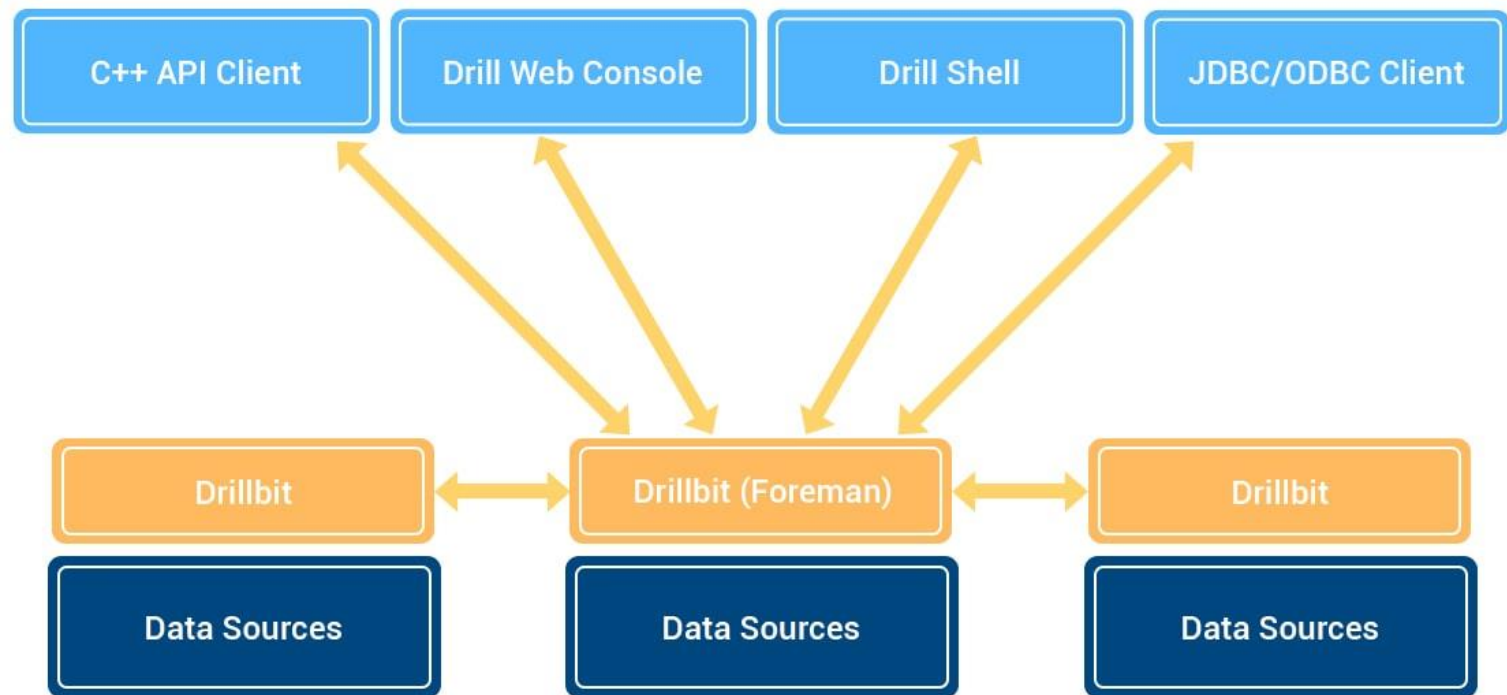
Apache Drill



SQL-on-Everything with Apache Drill

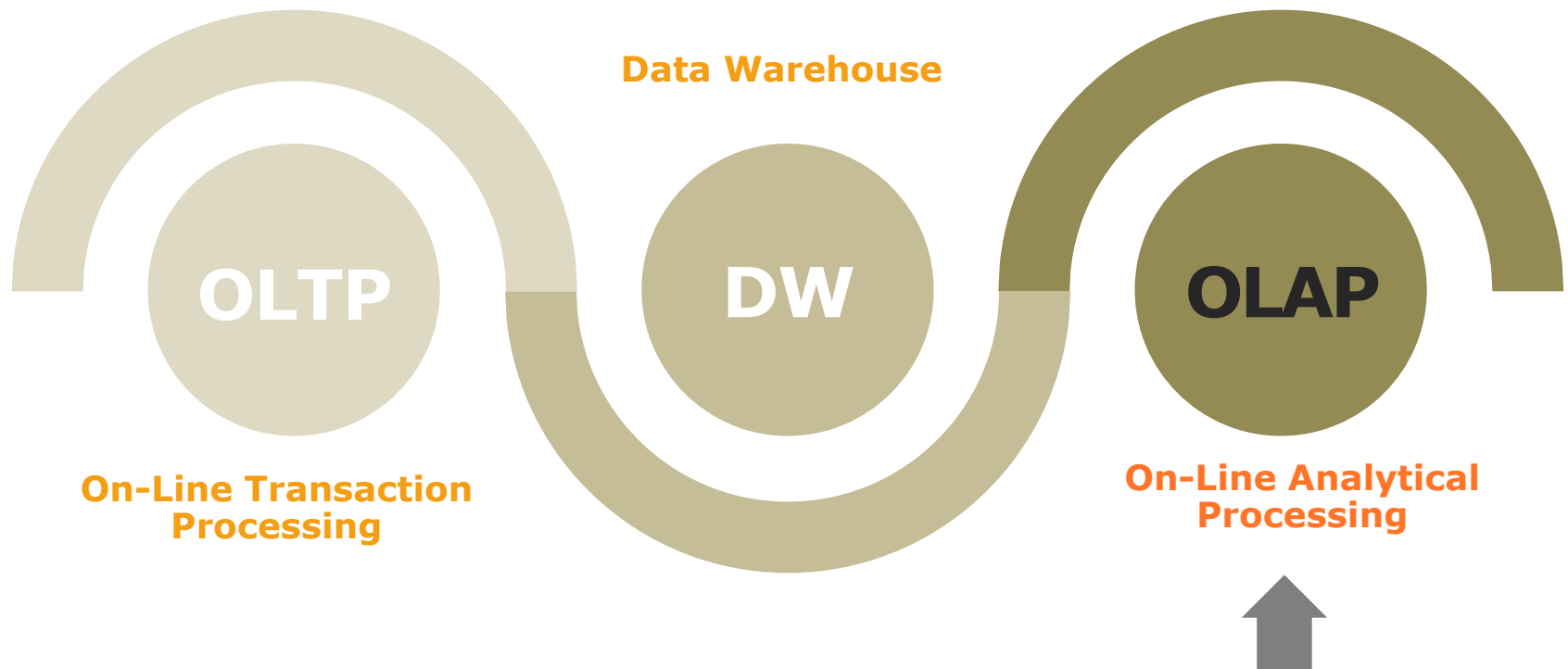


Apache Drill



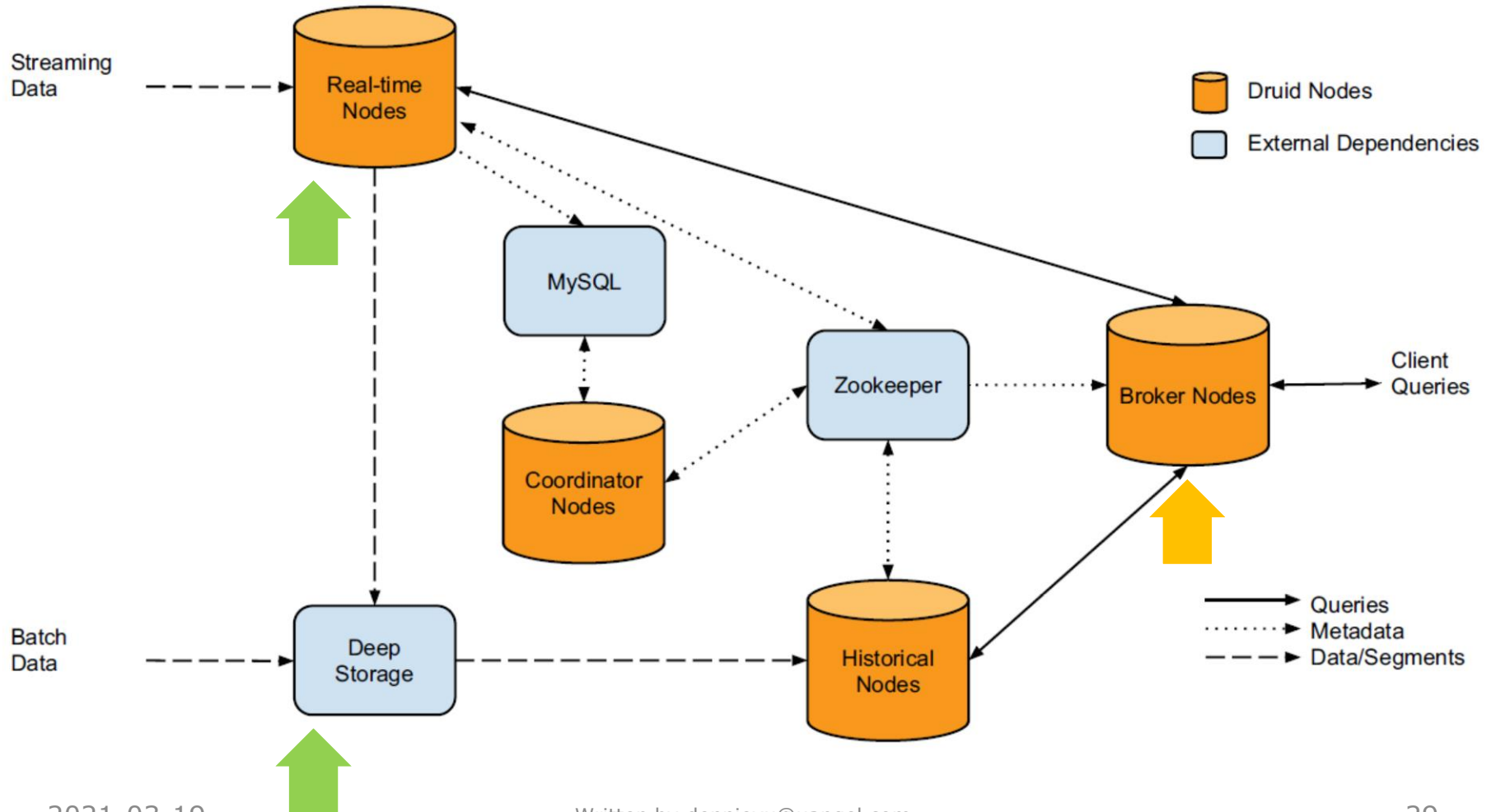
Druid

OLAP



빅데이터

Architecture



빅데이터

Overview



Historical Node

- *Deep Storage*에서 세그먼트 다운로드
- 세그먼트에 대한 *Broker Node*의 쿼리 실행 후 결과 반환
- *Zookeeper*를 통해 세그먼트 로드

Broker Node

- 쿼리 인입점
- 쿼리 실행, 결과 수집, 병합을 담당
- *Zookeeper*를 통해 *Realtime Node*, *Historical Node*가 어떤 것인지 판단

Coordinator Node

- 세그먼트 관리
- *Historical Node*에 지시하기 위해 *Zookeeper*를 사용

Realtime Node

- 실시간 데이터 수집, 데이터 인덱싱 및 세그먼트를 *Historical Node*에 전달

External Dependencies

- *Zookeeper* - 클러스터 내부에서의 **서비스 디스커버리** 및 데이터 토폴로지 운영
- *Metadata storage instance* - 세그먼트에 대한 **메타데이터 저장소**
- *Deep Storage / File System* - **세그먼트 저장**

배경

빅데이터



PROBLEM DEFINITION

- 리얼 타임이어야 하고
- 멀티 테넌시 하며
- 컬럼 지향
- 쿼리 속도 보장

Hadoop.. 위 4가지를 만족시킬 수 없다.

Druid의 메타데이터는 3가지 요소로 구성되어 있다.

- **Timestamp**가 있다.
- **Dimension** 컬럼이 많다. 데이터 속성이 많다는 것.
- **Metric** 컬럼이 많다. 수집하고자 하는 지표가 많다는 것.

이러한 **Metric**은 수집되고 **groupby**되고 **aggregate**되는 대상이다.

정의

빅데이터



OLAP(On-Line Analytical Processing) 즉, 온라인 분석 처리는 다차원 데이터 구조를 이용하여 다차원의 복잡한 질의를 **고속으로** 처리하는 **데이터 분석 기술**이다. 기업의 분석가, 관리자 및 임원들은 **OLAP** 기술을 통해 필요한 정보에 대해 **대화형**으로 빠르게 접근 가능하다.

“Druid is NOT time series DB”

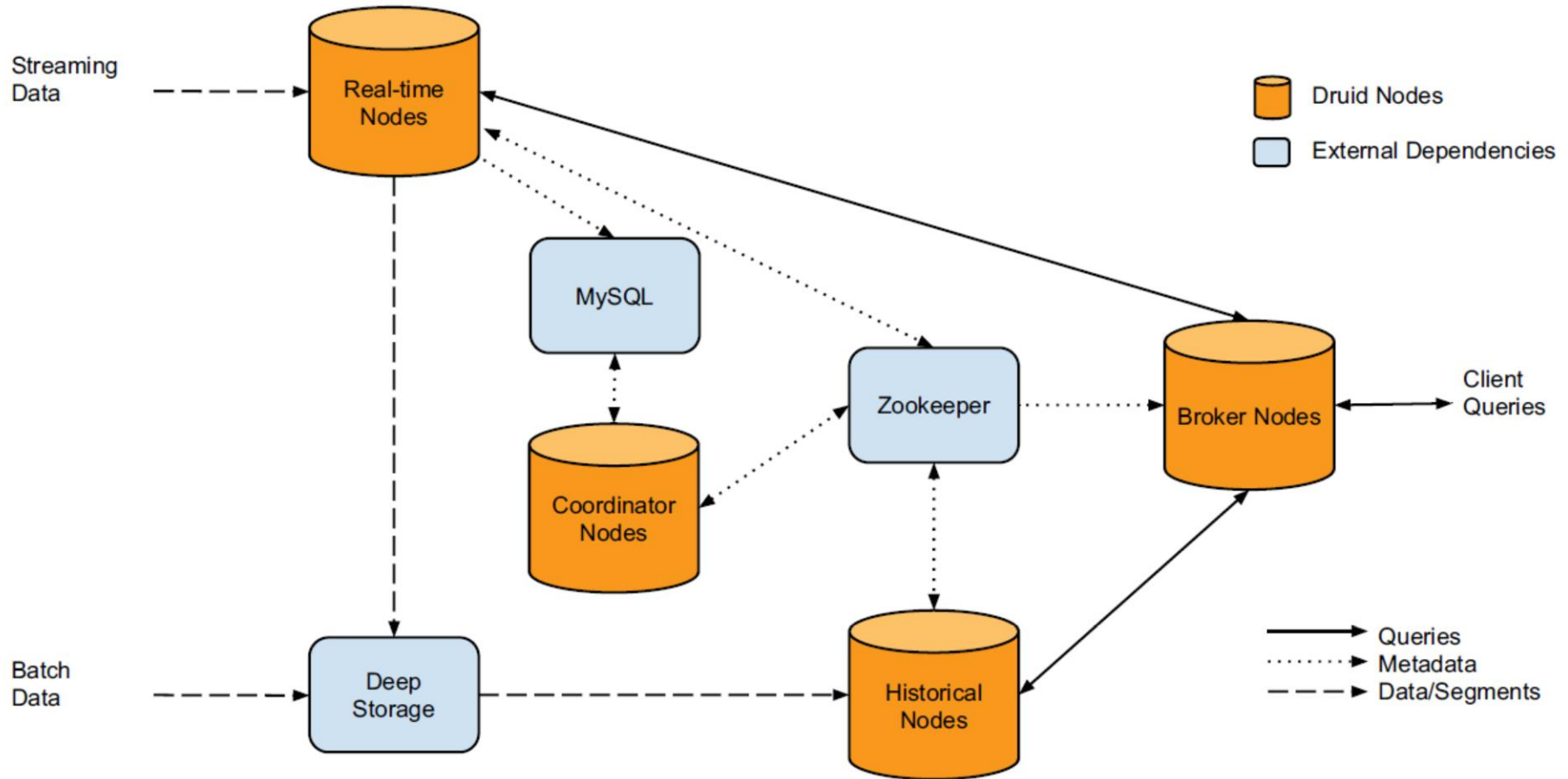
즉, **Druid**는 **원본**을 저장하지 않으며 기존 데이터에서 **OLAP**질의가 가능하도록 **indexing**하는 구조를 가지고 있다.

같은 시간 범위 내에 존재하는 **데이터를 분할**해 저장하는 점은 다른 시계열 DB (**Machbase, OpenTSDB, ..**)에서도 취하는 형태이다.

아키텍처

빅데이터

Diagram



빅데이터

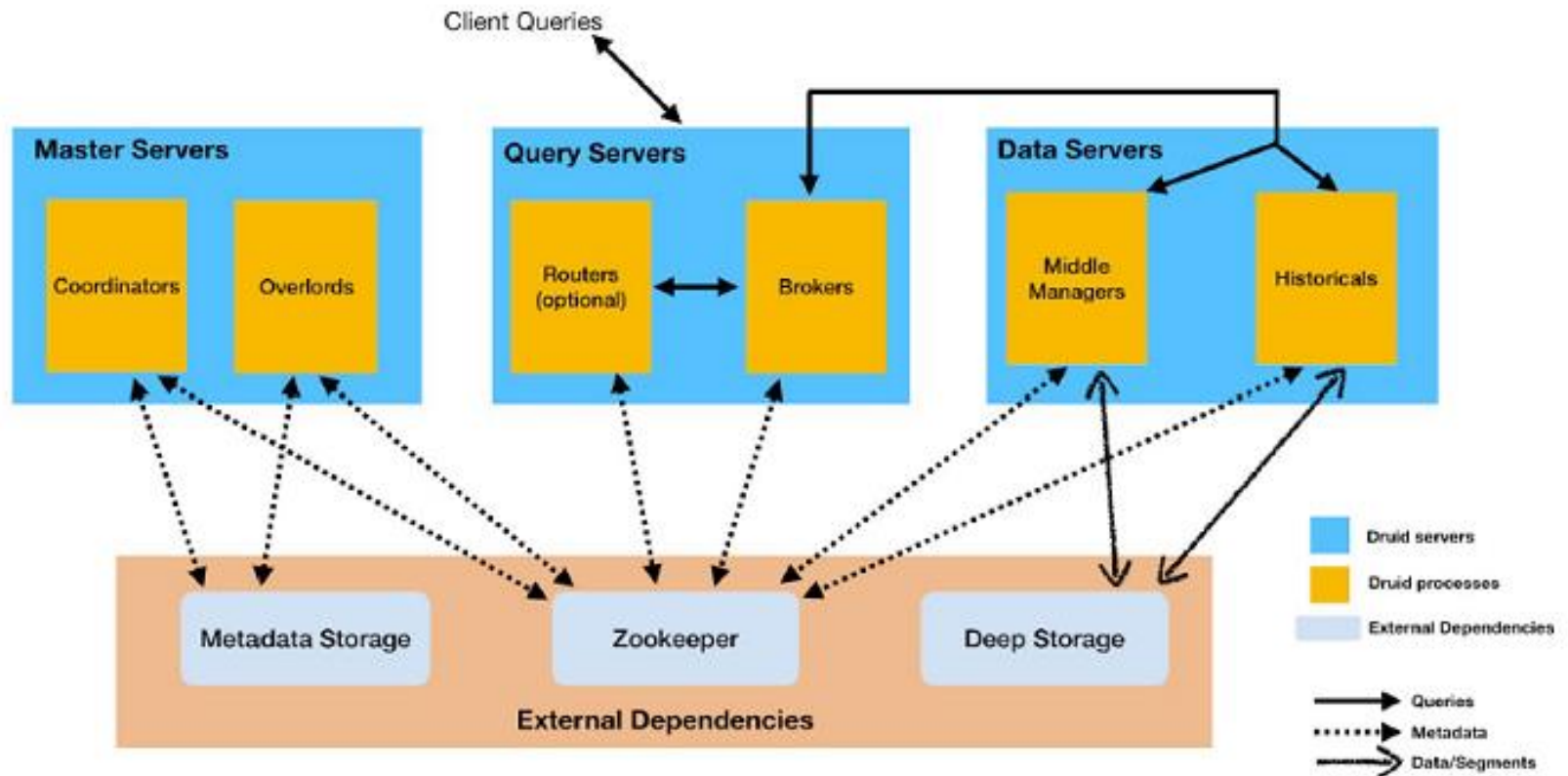
Diagram



- **Druid** 는 **실시간** 아키텍처와 **배치** 아키텍처를 모두 수용한다. 원본 데이터가 실시간인 경우 **Real-time node**에서 색인되며 **segment(time 설정 기준)**으로 **Deep Storage**로 내리게 된다. 배치 데이터는 **Deep Storage**에 저장되며 **Deep Storage**는 **HDFS, S3**등으로 구성될 수 있다.
- **사용자 쿼리**는 **Broker node**가 담당한다. **Segment** 정보를 생성하지 않은 **fresh**한 데이터는 **Real-time node**에서 그리고 배치로 색인된 데이터는 **Historical node**에서 가져오게 된다. (람다아키텍처) **Druid** 내부 컴포넌트간 통신은 **zookeeper**로 되어있다.
- **Druid** 관련 **메타정보**는 **mysql**등 **RDBMS**에 저장할 수있다. **Coordinator**는 **메타정보**를 통해 **segment life cycle**(복제 갯수 조정 및 **Load/Drop**)을 관리한다.

빅데이터

Diagram

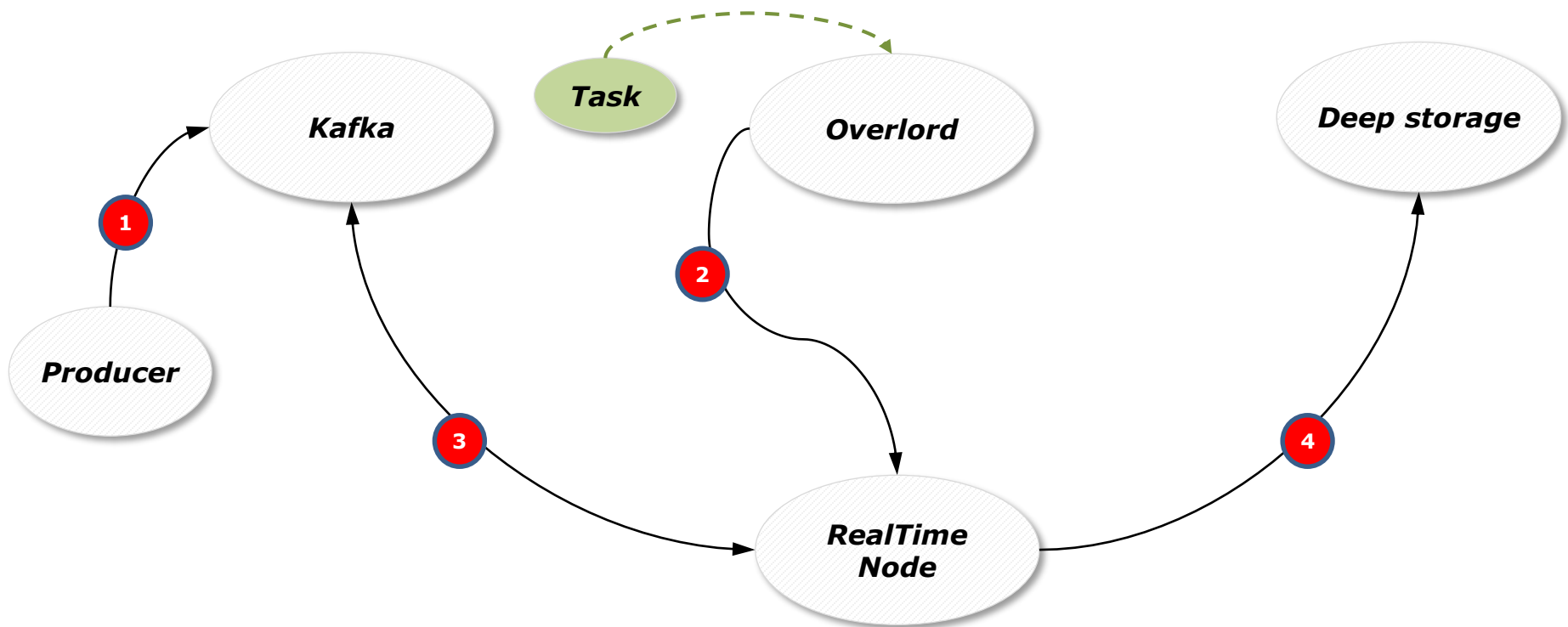


인덱싱

Realtime Ingestion



http://{OVERLORD_IP}:{port}/druid/indexer/v1/supervisor



Realtime Ingestion



http://{OVERLORD_IP}:{port}/druid/indexer/v1/supervisor

```
- {
-   "type": "kafka",
-   "dataSchema": {
-       "dataSource": "___cpoak",
-       "parser": {
-           "type": "string",
-           "parseSpec": {
-               "format": "json",
-               ...
-           }
-       },
-       ...
-   },
-   ...
- },
-   "tuningConfig": {
-       "type": "kafka",
-       "maxRowsPerSegment": 5000000
-   },
-   "ioConfig": {
-       "type": "kafka",
-       "topic": "example1",
-       "consumerProperties": {
-           "bootstrap.servers": "192.168.7.35:9092"
-       },
-       "taskCount": 1,
-       "replicas": 1,
-       "taskDuration": "PT1H"
-   }
- }
```

Batch Ingestion



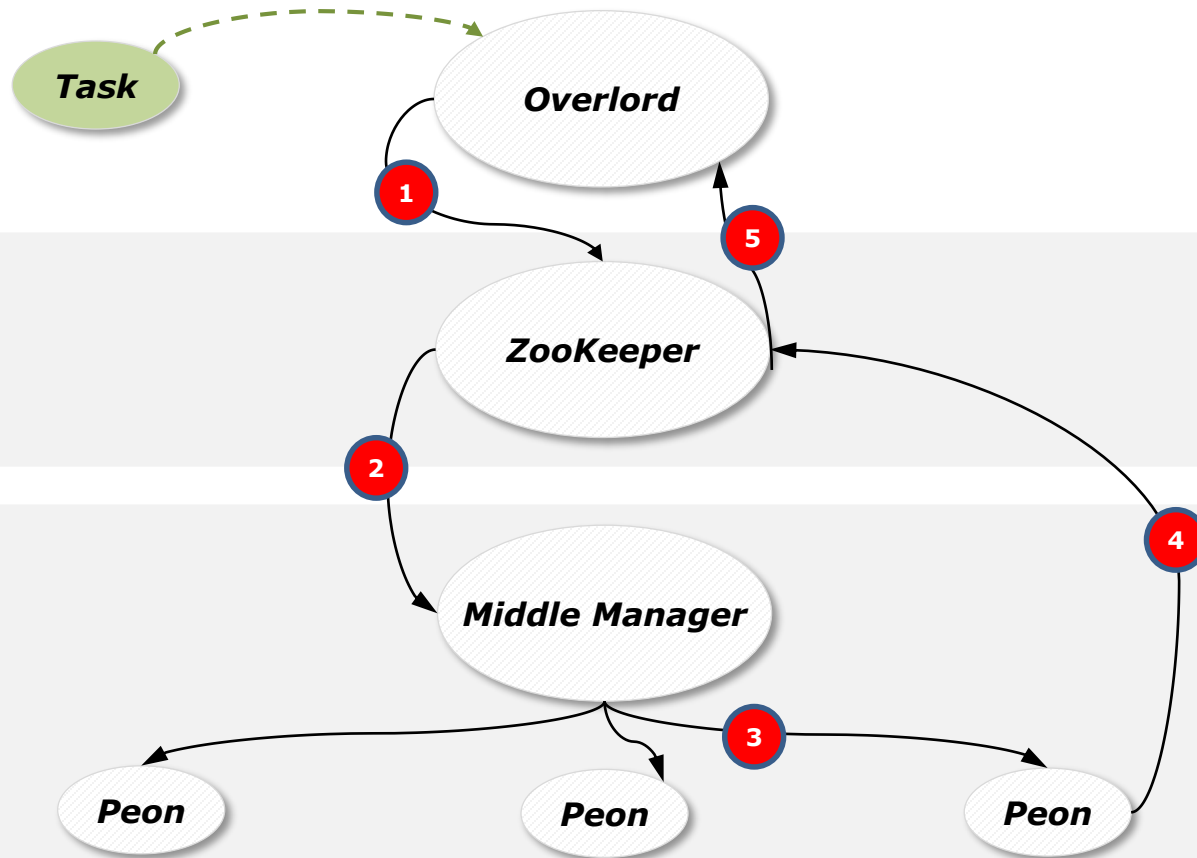
- **Druid**는 **실시간** 처리 외에도 **정적 파일 (HDFS/S3)**에 대한 **ingestion**도 지원한다.
- **HDFS**에서 **ingestion**을 하는 경우 **hadoop MR job**으로 수행되며 앞서 언급한 **overlord node**에 **ingestion job**을 **submit**하면 된다.
- **batch ingestion**의 경우 **type**은 **index-hadoop**이다. 실시간과 다른점은 **batch**의 경우 **intervals**를 꼭 지정해주어야 한다.
- 이 후 **MR job**이 수행 된다. (**1st MR** -determinePartitions, **2nd MR** - IndexGeneration)

빅데이터

Batch Ingestion



`http://{OVERLORD_IP}:{port}/druid/indexer/v1/task`



Batch Ingestion



```
- {  
-   "type": "index",           http://{OVERLORD_IP}:{port}/druid/indexer/v1/task  
-   "spec": {  
-       "dataSchema": {  
-           "dataSource": "____dhnpj",  
-           "parser": {  
-               "type": "csv.stream",  
-               "timestampSpec": {  
-                   "column": "create_time",  
-                   ...  
-               },  
-               ...  
-           },  
-           ...  
-       },  
-       "ioConfig": {  
-           "type": "index",  
-           "firehose": {  
-               "type": "local",  
-               "filter": "____dhnpj_1970-01-01T00:00:00.000Z.csv",  
-               "baseDir": "/tmp"  
-           }  
-       },  
-       "tuningConfig": {  
-           "type": "index",  
-           "maxRowsInMemory": 75000,  
-           "ignoreInvalidRows": true,  
-           "buildV9Directly": true  
-       }  
-   },  
-   "context": ...  
- }
```

UI

빅데이터

Druid Overlord Console

Supervisors

Showing 1 to 3 of 3 entries

Search all columns:

Showing 1 to 3 of 3 entries

First

Previous

1

Next

Last

Running Tasks

Showing 1 to 3 of 3 entries

Search all columns:

| id | type | createdTime | queueInsertionTime | statusCode | status | runnerStatusCode | duration | location host | location port | dataSource | errorMsg | more |
|---|-------------|--------------------------|--------------------------|------------|---------|------------------|----------|---------------|---------------|-----------------|----------|---|
| index_kafka___cpoak_6c473e2a7b84f31_jbnkploml | index_kafka | 2020-12-04T11:26:12.391Z | 2020-12-04T11:26:12.392Z | RUNNING | RUNNING | RUNNING | null | platform-dev2 | 8102 | ___cpoak | null | payload status log (all) log (last 8kb) |
| index_kafka___nirm_8c89d7b334a2bf5_mcijaahf | index_kafka | 2020-12-04T11:23:10.715Z | 2020-12-04T11:23:10.716Z | RUNNING | RUNNING | RUNNING | null | platform-dev2 | 8100 | ___nirm | null | payload status log (all) log (last 8kb) |
| index_kafka_kafka_microtrip_cf5f4be7b85e3ce_ckknknp | index_kafka | 2020-12-04T10:42:12.427Z | 2020-12-04T10:42:12.428Z | RUNNING | RUNNING | RUNNING | null | platform-dev2 | 8103 | kafka_microtrip | null | payload status log (all) log (last 8kb) |

Showing 1 to 3 of 3 entries

First

Previous

1

Next

Last

Pending Tasks - Tasks waiting to be assigned to a worker

Waiting Tasks - Tasks waiting on locks

Complete Tasks - Tasks recently completed

Showing 1 to 10 of 10 entries

Search all columns:

| id | type | createdTime | queueInsertionTime | statusCode | status | runnerStatusCode | duration | location host | location port | dataSource | errorMsg |
|--|-------|--------------------------|--------------------------|------------|---------|------------------|----------|---------------|---------------|------------|---|
| index___dhnpj_2020-12-04T00:27:59.038Z | index | 2020-12-04T00:27:59.039Z | 1970-01-01T00:00:00.000Z | SUCCESS | SUCCESS | NONE | 6212 | null | -1 | ___dhnpj | null |
| index___dhnpj_2020-12-04T01:20:03.217Z | index | 2020-12-04T01:20:03.217Z | 1970-01-01T00:00:00.000Z | FAILED | FAILED | NONE | 5659 | null | -1 | ___dhnpj | Exception: [io.druid.indexing.common.task.IndexTask.run(IndexTask.java:226), io.druid.indexing... |
| index___dhnpj_2020-12-04T01:30:03.133Z | index | 2020-12-04T01:30:03.134Z | 1970-01-01T00:00:00.000Z | SUCCESS | SUCCESS | NONE | 6248 | null | -1 | ___dhnpj | null |
| index___dhnpj_2020-12-04T01:40:03.112Z | index | 2020-12-04T01:40:03.112Z | 1970-01-01T00:00:00.000Z | FAILED | FAILED | NONE | 5630 | null | -1 | ___dhnpj | Exception: [io.druid.indexing.common.task.IndexTask.run(IndexTask.java:226), io.druid.indexing... |
| index___dhnpj_2020-12-04T01:50:03.112Z | index | 2020-12-04T01:50:03.112Z | 1970-01-01T00:00:00.000Z | SUCCESS | SUCCESS | NONE | 6206 | null | -1 | ___dhnpj | null |
| index___dhnpj_2020-12-04T02:10:03.112Z | index | 2020-12-04T02:10:03.112Z | 1970-01-01T00:00:00.000Z | SUCCESS | SUCCESS | NONE | 6262 | null | -1 | ___dhnpj | null |
| index___dhnpj_2020-12-04T02:40:03.112Z | index | 2020-12-04T02:40:03.112Z | 1970-01-01T00:00:00.000Z | SUCCESS | SUCCESS | NONE | 6177 | null | -1 | ___dhnpj | null |
| index___dhnpj_2020-12-04T02:50:03.112Z | index | 2020-12-04T02:50:03.112Z | 1970-01-01T00:00:00.000Z | FAILED | FAILED | NONE | 5448 | null | -1 | ___dhnpj | Exception: [io.druid.indexing.common.task.IndexTask.run(IndexTask.java:226), io.druid.indexing... |
| index___dhnpj_2020-12-04T03:00:03.120Z | index | 2020-12-04T03:00:03.120Z | 1970-01-01T00:00:00.000Z | SUCCESS | SUCCESS | NONE | 6206 | null | -1 | ___dhnpj | null |
| index___dhnpj_2020-12-04T03:10:03.108Z | index | 2020-12-04T03:10:03.108Z | 1970-01-01T00:00:00.000Z | FAILED | FAILED | NONE | 5789 | null | -1 | ___dhnpj | Exception: [io.druid.indexing.common.task.IndexTask.run(IndexTask.java:226), io.druid.indexing... |

2021-03-19

Written by dennieyu@uangel.com

46

빅데이터



| Cluster Servers | | | | | | |
|--|----------------|------------------|---------------|-----------------|-----------------|----------------------|
| Average Server Percent Used: 0.001578481730769231% | | | | | | |
| Show 10 entries | | | | | | |
| Server host | Server maxSize | Server type | Server tier | Server priority | Server currSize | Server percentUsed |
| platform-dev2.8083 | 130000000000 | historical | _default_tier | 0 | 8208105 | 0.006313926923076924 |
| platform-dev2.8100 | 0 | indexer-executor | _default_tier | 0 | 0 | 0 |
| platform-dev2.8101 | 0 | indexer-executor | _default_tier | 0 | 0 | 0 |
| platform-dev2.8102 | 0 | indexer-executor | _default_tier | 0 | 0 | 0 |
| Showing 1 to 4 of 4 entries | | | | | | |

First Previous 1 Next Last

| Cluster Segments | | | | | |
|--------------------------------|----------------|---|--------------------------|---|-------------------|
| Show 10 entries | | | | | |
| Server host | Segment dataSo | Segment interval | Segment version | Segment dimensions | Segment metr |
| platform-dev2.8083 | fifa_ | 2017-01-01T00:00:00.000Z/2018-01-01T00:00:00.000Z | 2020-10-31T08:45:06.797Z | | count,ARG,BRA |
| platform-dev2.8083 | sales_geo | 2011-12-01T00:00:00.000Z/2012-01-01T00:00:00.000Z | 2019-04-05T07:41:05.100Z | Category,City,Country,CustomerName,OrderID,PostalCode,ProductName,Quantity,Region,Segment,ShipDate,ShipMode,State,Sub-Category,ShipStatus,orderprofitable,SalesAboveTarget,latitude,longitude | count,Discount,F |
| platform-dev2.8083 | _hfohg | 2020-09-01T00:00:00.000Z/2020-10-01T00:00:00.000Z | 2020-09-16T01:08:13.129Z | id,kids_id,alias_name,address,active_yn | geo_point,lat,lon |
| platform-dev2.8083 | sales_geo | 2014-08-01T00:00:00.000Z/2014-09-01T00:00:00.000Z | 2019-04-05T07:41:05.100Z | Category,City,Country,CustomerName,OrderID,PostalCode,ProductName,Quantity,Region,Segment,ShipDate,ShipMode,State,Sub-Category,ShipStatus,orderprofitable,SalesAboveTarget,latitude,longitude | count,Discount,F |
| platform-dev2.8083 | fifa_ | 2000-01-01T00:00:00.000Z/2001-01-01T00:00:00.000Z | 2020-10-31T08:45:06.797Z | | count,ARG,BRA |
| platform-dev2.8083 | sales_geo | 2013-04-01T00:00:00.000Z/2013-05-01T00:00:00.000Z | 2019-04-05T07:41:05.100Z | Category,City,Country,CustomerName,OrderID,PostalCode,ProductName,Quantity,Region,Segment,ShipDate,ShipMode,State,Sub-Category,ShipStatus,orderprofitable,SalesAboveTarget,latitude,longitude | count,Discount,F |
| platform-dev2.8083 | _unejj | 2019-01-04T01:00:00.000Z/2019-01-04T02:00:00.000Z | 2020-10-26T08:50:01.360Z | tid,kids_mobile_phone,parent_mobile_phone,status,create_user | count,id,kids_id |
| platform-dev2.8083 | fifa_ | 1997-01-01T00:00:00.000Z/1998-01-01T00:00:00.000Z | 2020-10-31T08:45:06.797Z | | count,ARG,BRA |
| platform-dev2.8083 | fifa_ | 2004-01-01T00:00:00.000Z/2005-01-01T00:00:00.000Z | 2020-10-31T08:45:06.797Z | | count,ARG,BRA |
| platform-dev2.8083 | sales_geo | 2012-11-01T00:00:00.000Z/2012-12-01T00:00:00.000Z | 2019-04-05T07:41:05.100Z | Category,City,Country,CustomerName,OrderID,PostalCode,ProductName,Quantity,Region,Segment,ShipDate,ShipMode,State,Sub-Category,ShipStatus,orderprofitable,SalesAboveTarget,latitude,longitude | count,Discount,F |
| Showing 1 to 10 of 185 entries | | | | | |

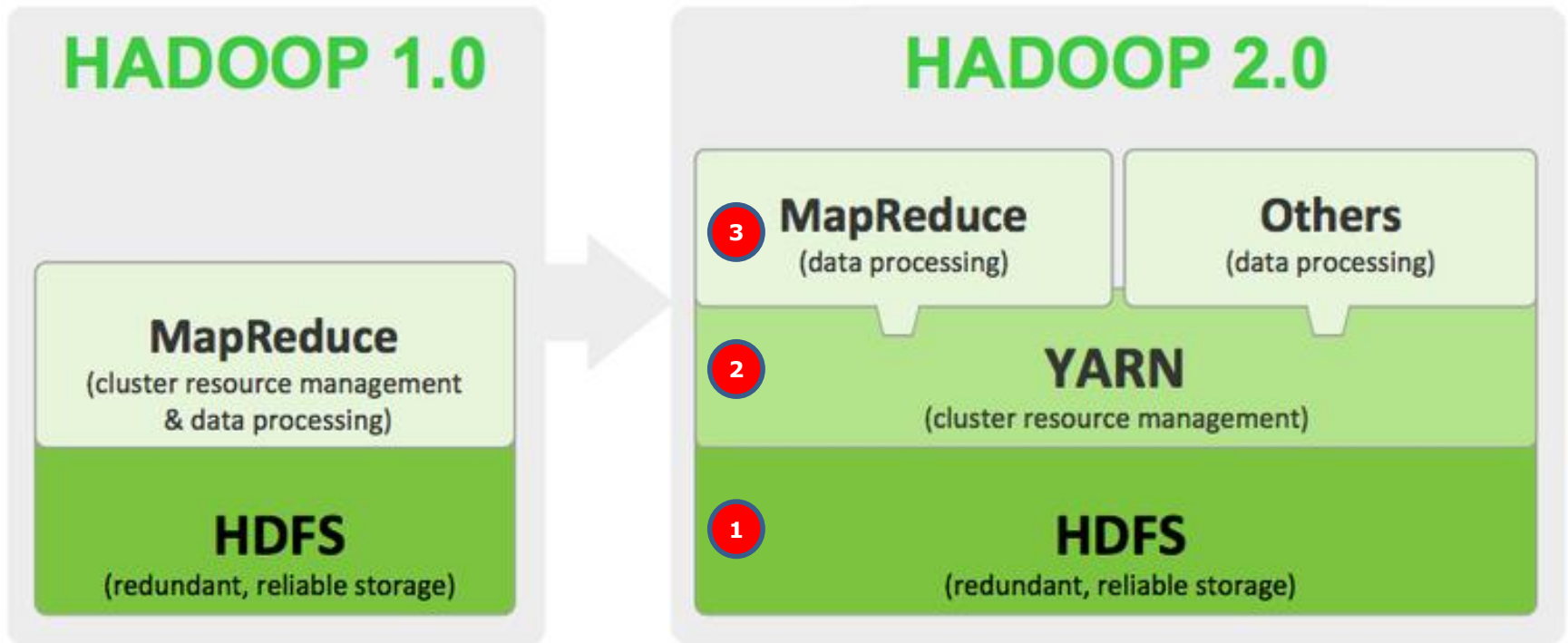
First Previous 1 2 3 4 5 Next Last

Coordinator UI – Cluster Servers

하둡

빅데이터

Architecture



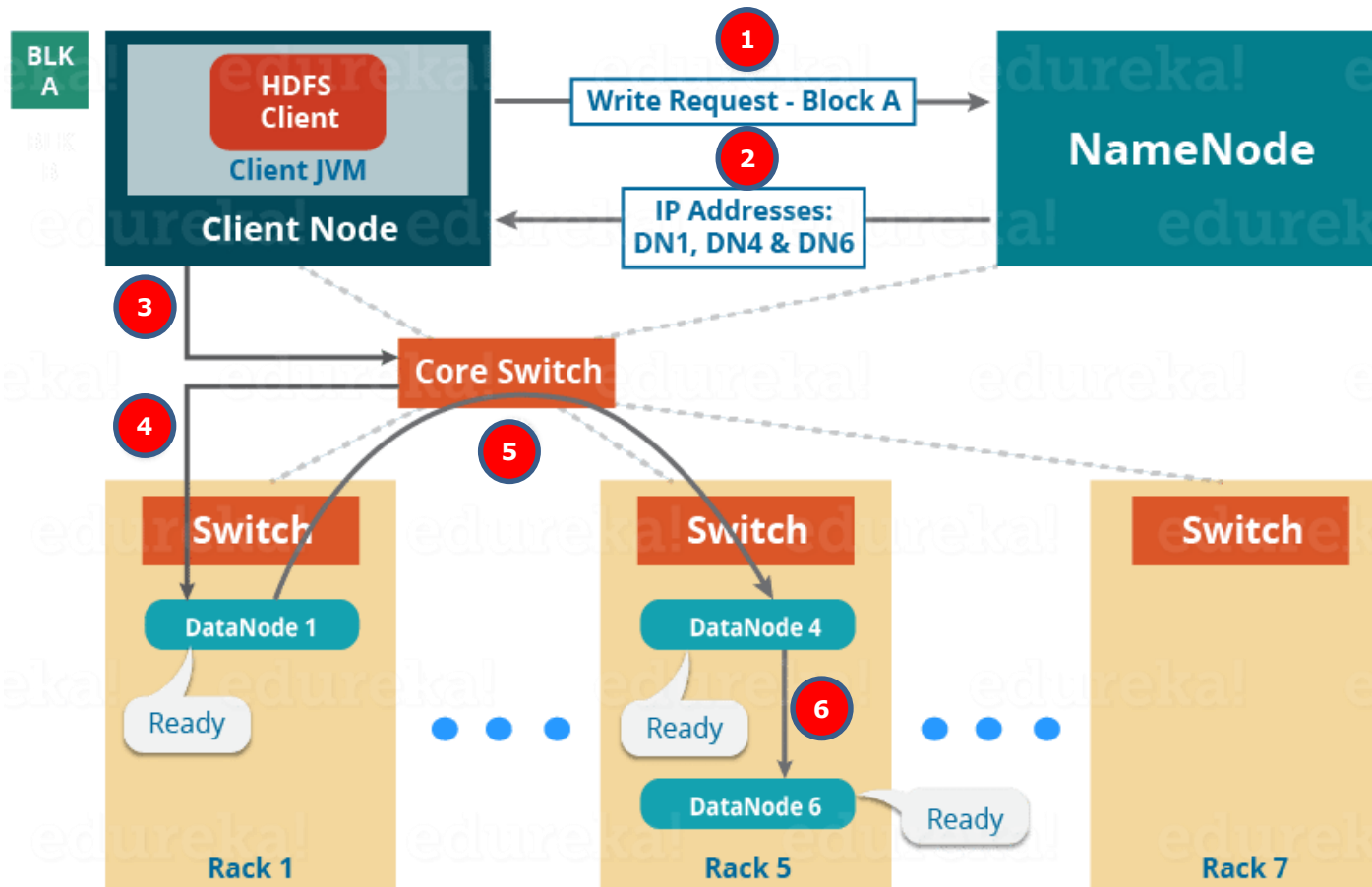
※ 현재 최신버전은 v3 이고, *Erasure Coding* 을 지원함.

빅데이터

1. HDFS



Setting up HDFS - Write Pipeline

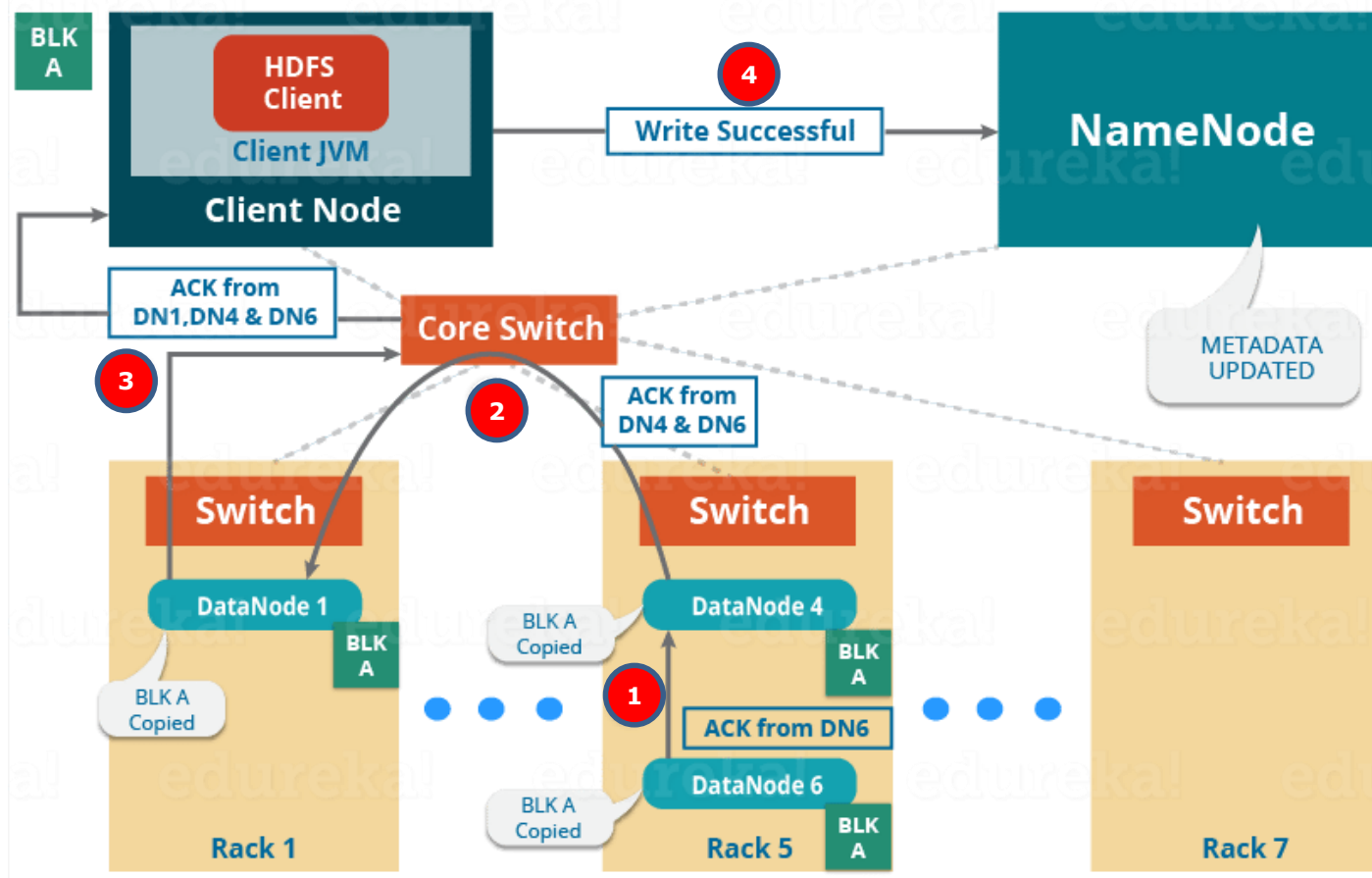


빅데이터

1. HDFS



Acknowledgement in HDFS - Write

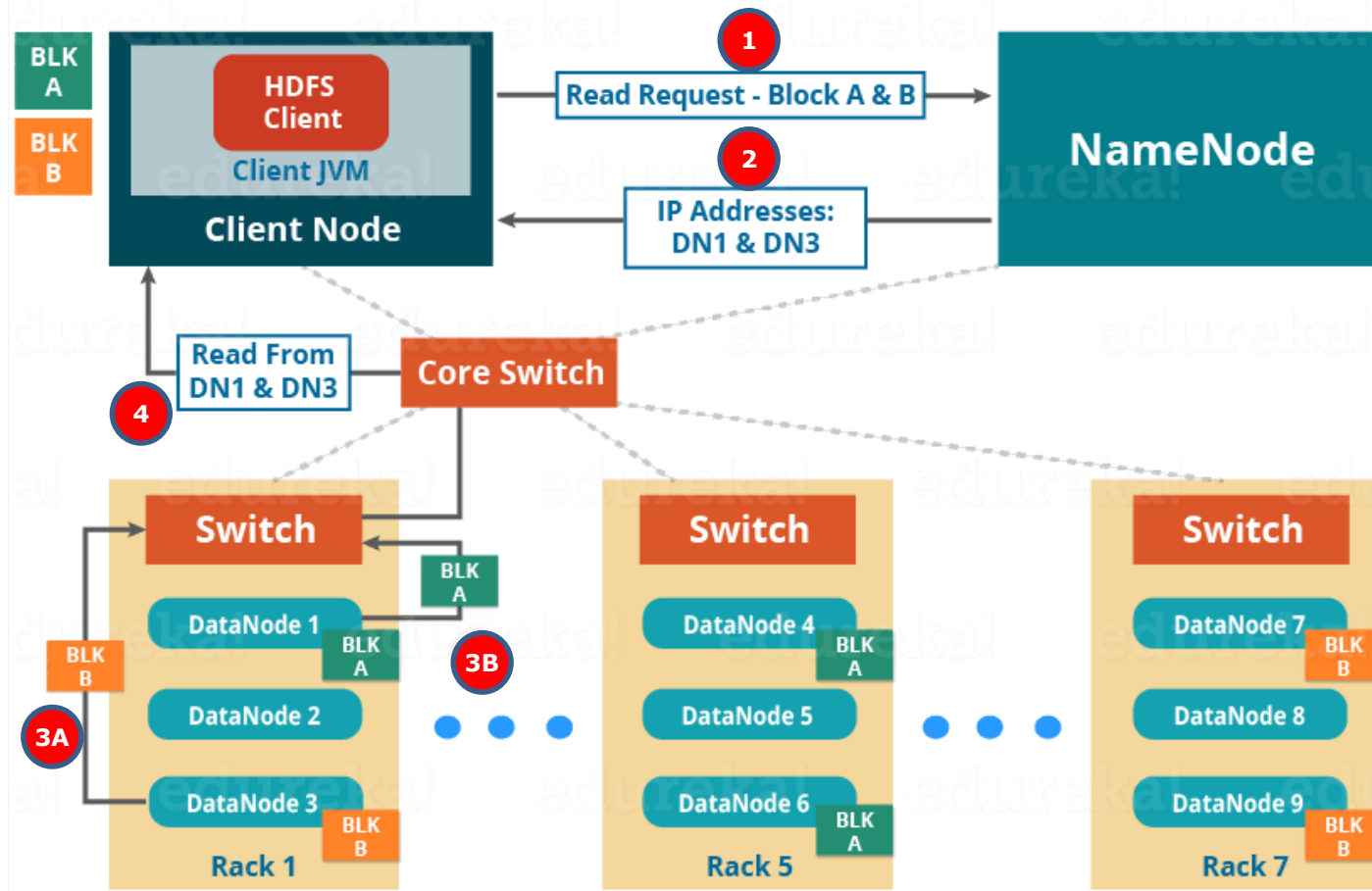


빅데이터

1. HDFS



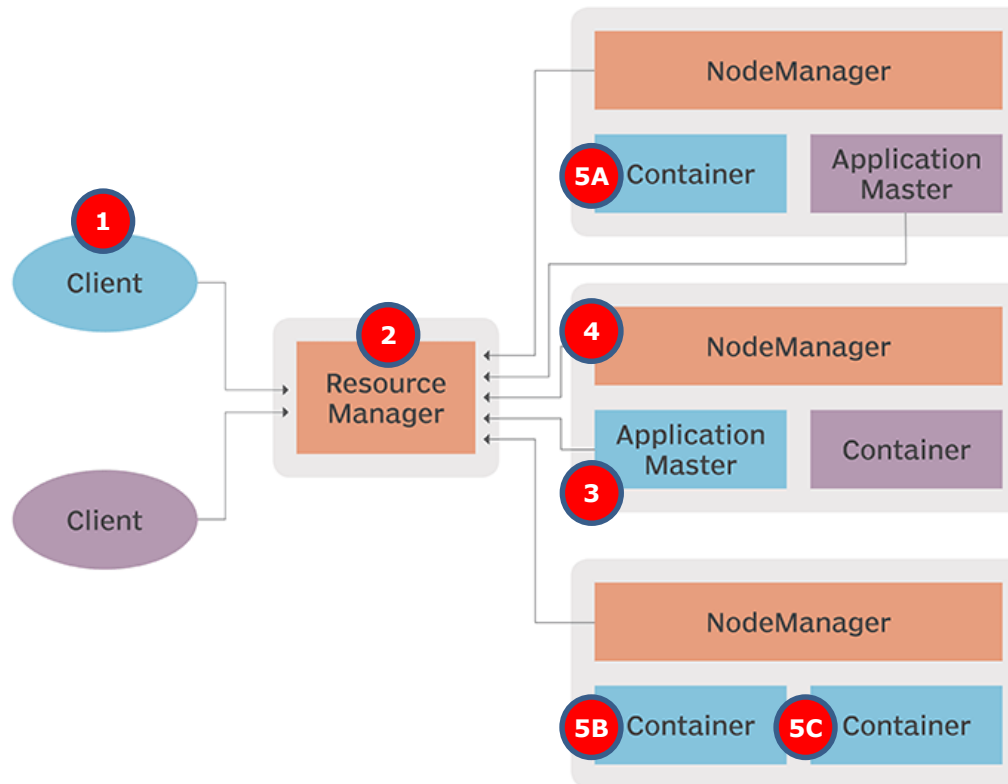
HDFS - Read Architecture



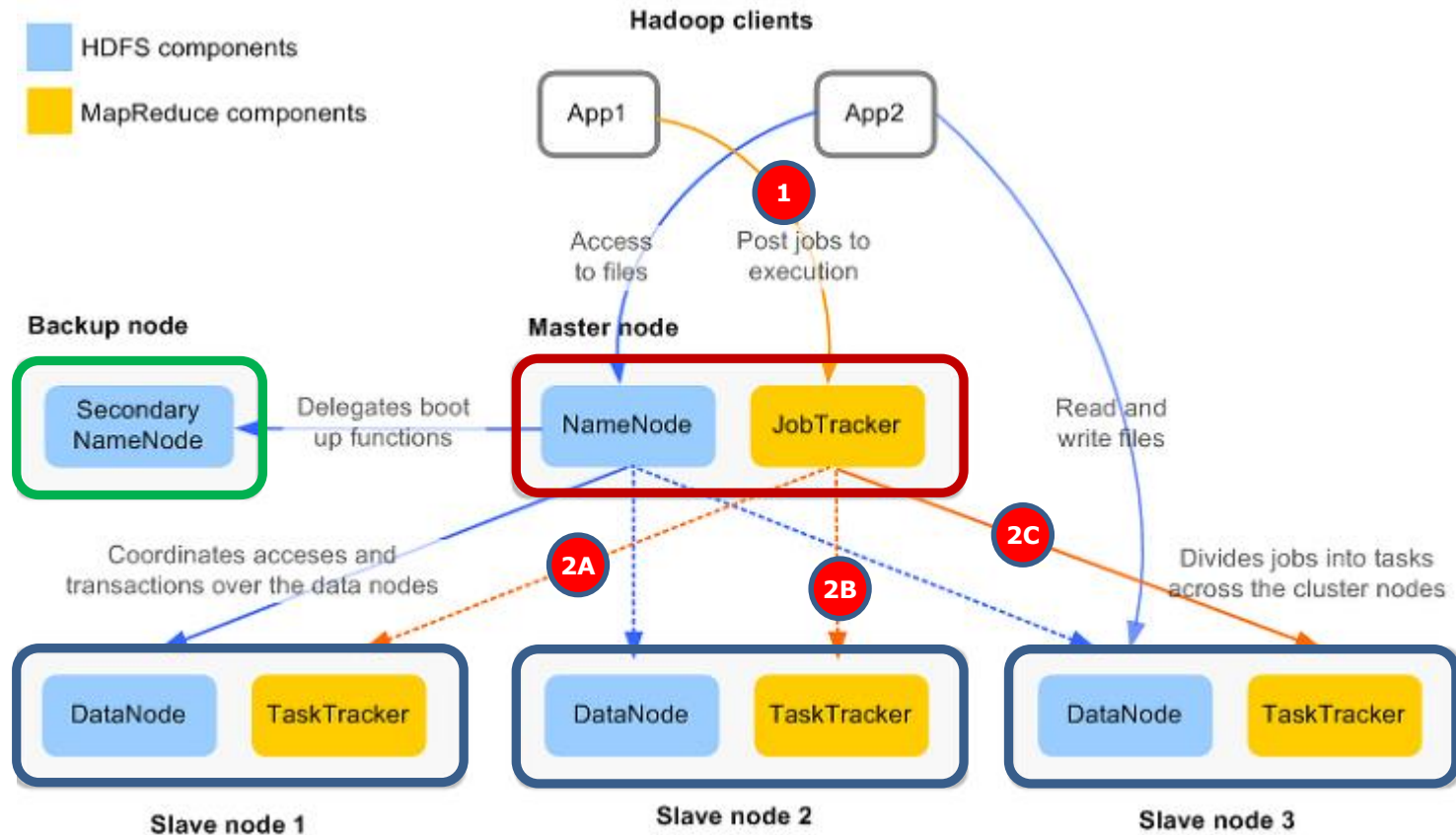
2. YARN



Specialized application clients submit processing jobs to YARN's ResourceManager, which works with ApplicationMasters and NodeManagers to schedule, run and monitor the jobs.



3. MR (MapReduce)





4. HDFS 코드로 접근하기

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import java.io.IOException;

public class SingleFileWriteRead {
    public static void main(String[] args) {
        if (args.length != 2) { System.err.println("Usage: SingleFileWriteRead <input> <output>"); System.exit(2); }
        Configuration conf = new Configuration();
        try {
            FileSystem hdfs = FileSystem.get(conf);
            // 경로를 체크하고 존재한다면 지운다
            Path path = new Path(args[0]);
            if (hdfs.exists(path)) { hdfs.delete(path, true); }

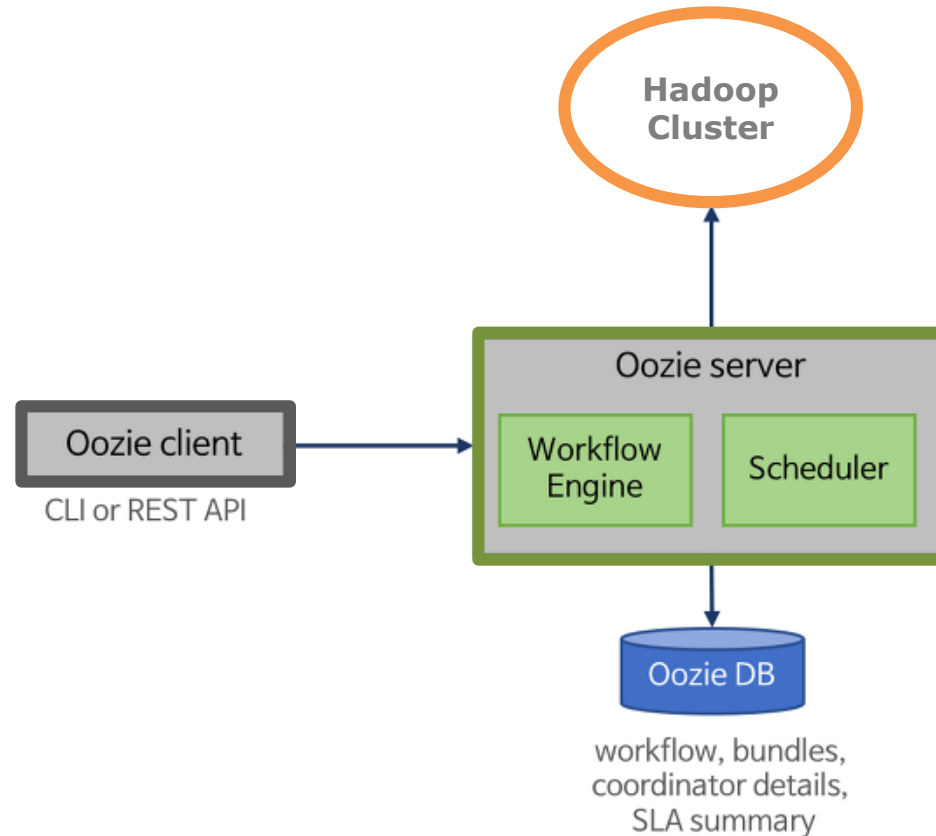
            // 파일을 저장하고 스트림을 닫는다
            FSDataOutputStream outputStream = hdfs.create(path);
            outputStream.writeUTF(args[1]);
            outputStream.close();

            // 앞서 저장한 파일을 읽어 string에 저장하고 닫는다
            FSDataInputStream inputStream = hdfs.open(path);
            String inputString = inputStream.readUTF();
            inputStream.close();

            // 읽은 내용을 출력
            System.out.println("## inputString: " + inputString);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Oozie

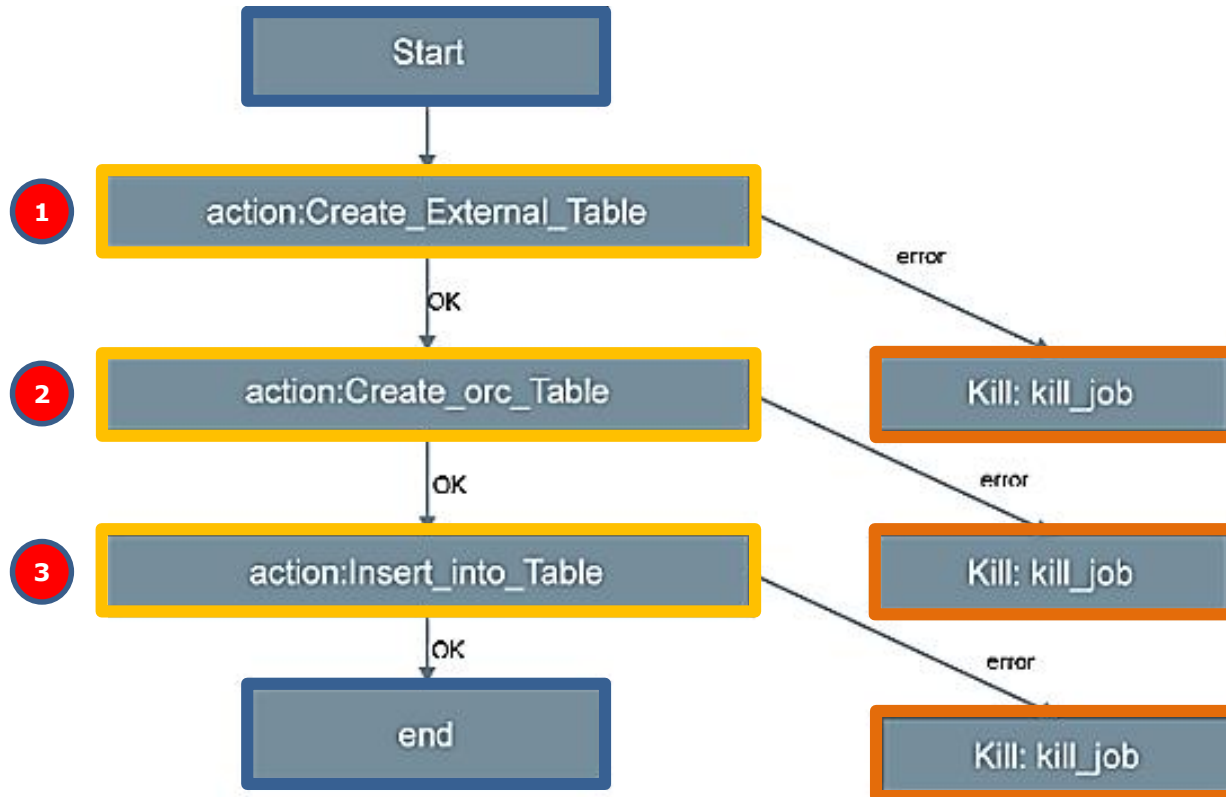
Architecture



- Hadoop의 Job을 관리하기 위한 서버 기반의 **Workflow Scheduler System**
- **Java servlet-container**에서 수행되는 서버 기반의 **Workflow Engine**

빅데이터

DAG (Directed Acyclic Graph) 예시



Step 1 - DDL for Hive external table (**external.hive**)



```
Create external table external_table (  
  name string,  
  age int,  
  address string,  
  zip int  
)  
row format delimited  
fields terminated by ','  
stored as textfile  
location '/test/abc';
```

Step 2 - DDL for Hive ORC table (**orc.hive**)



```
Create Table orc_table (  
  name string, -- Concat value of first name and last name with space as separator  
  yearofbirth int,  
  age int, -- Current year minus year of birth  
  address string,  
  zip int  
)  
STORED AS ORC;
```

Step 3 - Hive script to insert data from external table to ORC table (Copydata.hql)

```
use ${database_name}; -- input from Oozie
insert into table orc_table
select
concat(first_name,',',last_name) as name,
yearofbirth,
year(from_unixtime) --yearofbirth as age,
address,
zip
from external_table
;
```

빅데이터

Step 4 - Create a workflow to execute all the above three steps. (**workflow.xml**)

```
<!-- This is a comment -->
<workflow-app xmlns = "uri:oozie:workflow:0.4" name = "simple-Workflow">
  <start to = "Create_External_Table" />

  <!--Step 1 -->

  <action name = "Create_External_Table">
    <hive xmlns = "uri:oozie:hive-action:0.4">
      <job-tracker>xyz.com:8088</job-tracker>
      <name-node>hdfs://rootname</name-node>
      <script>hdfs_path_of_script/external.hive</script>
    </hive>


    <ok to = "Create_orc_Table" />
    <error to = "kill_job" />
  </action>

  <!--Step 2 -->

  <action name = "Create_orc_Table">
    <hive xmlns = "uri:oozie:hive-action:0.4">
      <job-tracker>xyz.com:8088</job-tracker>
      <name-node>hdfs://rootname</name-node>
      <script>hdfs_path_of_script/orc.hive</script>
    </hive>

    <ok to = "Insert_into_Table" />
    <error to = "kill_job" />
  </action>
```

빅데이터

Step 4 - Create a workflow to execute all the above three steps. (**workflow.xml**)  Cont'd

```
<!--Step 3 -->

<action name = "Insert_into_Table">
  <hive xmlns = "uri:oozie:hive-action:0.4">
    <job-tracker>xyz.com:8088</job-tracker>
    <name-node>hdfs://rootname</name-node>
    <script>hdfs_path_of_script/Copydata.hql</script>
    <param>database_name</param>
  </hive>

  <ok to = "end" />
  <error to = "kill_job" />
</action>

<kill name = "kill_job">
  <message>Job failed</message>
</kill>

<end name = "end" />

</workflow-app>
```



UI

빅데이터

Enable Oozie Web Console



[Documentation](#)

Oozie Web Console

Workflow Jobs | Coordinator Jobs | Bundle Jobs | System Info | Instrumentation | Settings

All Jobs | Active Jobs | Done Jobs | Custom Filter ▼

| | Job Id | Name | Status | Run | User ▲ | Group | Created | Started | Last Modified | |
|----|-----------------------------------|--------------|-----------|-----|--------|-------|-------------------------------|-------------------------------|-------------------------------|---|
| 1 | 0000008-140712104057581-oozie... | FuelWorkflow | SUCCEEDED | 0 | hadoop | | Sat, 12 Jul 2014 06:15:38 GMT | Sat, 12 Jul 2014 06:15:46 GMT | Sat, 12 Jul 2014 06:19:43 GMT | ▲ |
| 2 | 0000007-140712104057581-oozie... | FuelWorkflow | SUCCEEDED | 0 | hadoop | | Sat, 12 Jul 2014 06:07:03 GMT | Sat, 12 Jul 2014 06:07:12 GMT | Sat, 12 Jul 2014 06:11:36 GMT | |
| 3 | 0000006-140712104057581-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Sat, 12 Jul 2014 05:57:59 GMT | Sat, 12 Jul 2014 06:02:34 GMT | Sat, 12 Jul 2014 06:04:30 GMT | |
| 4 | 0000005-140712104057581-oozie... | FuelWorkflow | FAILED | 0 | hadoop | | Sat, 12 Jul 2014 05:00:09 GMT | Sat, 12 Jul 2014 05:00:24 GMT | Sat, 12 Jul 2014 05:00:26 GMT | |
| 5 | 0000004-140712104057581-oozie... | FuelWorkflow | SUCCEEDED | 0 | hadoop | | Sat, 12 Jul 2014 04:50:32 GMT | Sat, 12 Jul 2014 04:50:43 GMT | Sat, 12 Jul 2014 04:58:33 GMT | |
| 6 | 0000003-140712104057581-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Sat, 12 Jul 2014 04:35:51 GMT | Sat, 12 Jul 2014 04:36:04 GMT | Sat, 12 Jul 2014 04:44:48 GMT | |
| 7 | 0000002-140712104057581-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Sat, 12 Jul 2014 03:58:22 GMT | Sat, 12 Jul 2014 03:58:52 GMT | Sat, 12 Jul 2014 04:31:26 GMT | |
| 8 | 0000001-140712104057581-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Sat, 12 Jul 2014 03:55:44 GMT | Sat, 12 Jul 2014 03:55:51 GMT | Sat, 12 Jul 2014 04:31:04 GMT | E |
| 9 | 0000000-140712104057581-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Sat, 12 Jul 2014 03:50:26 GMT | Sat, 12 Jul 2014 03:50:39 GMT | Sat, 12 Jul 2014 04:29:54 GMT | |
| 10 | 0000001-140710174734611-oozie-... | FuelWorkflow | SUCCEEDED | 0 | hadoop | | Thu, 10 Jul 2014 06:13:54 GMT | Thu, 10 Jul 2014 06:14:05 GMT | Thu, 10 Jul 2014 06:15:45 GMT | |
| 11 | 0000000-140710174734611-oozie-... | FuelWorkflow | KILLED | 0 | hadoop | | Thu, 10 Jul 2014 06:06:42 GMT | Thu, 10 Jul 2014 06:06:54 GMT | Thu, 10 Jul 2014 06:16:31 GMT | |
| 12 | 0000001-140709174422442-oozie... | FuelWorkflow | FAILED | 0 | hadoop | | Wed, 09 Jul 2014 06:40:56 GMT | Wed, 09 Jul 2014 06:41:11 GMT | Thu, 10 Jul 2014 05:48:05 GMT | |
| 13 | 0000000-140709174422442-oozie... | FuelWorkflow | FAILED | 0 | hadoop | | Wed, 09 Jul 2014 06:38:16 GMT | Wed, 09 Jul 2014 06:38:30 GMT | Thu, 10 Jul 2014 05:48:05 GMT | |
| 14 | 0000003-140708174838379-oozie... | FuelWorkflow | SUCCEEDED | 0 | hadoop | | Tue, 08 Jul 2014 07:13:47 GMT | Tue, 08 Jul 2014 07:14:07 GMT | Tue, 08 Jul 2014 07:16:29 GMT | |
| 15 | 0000002-140708174838379-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Tue, 08 Jul 2014 06:27:35 GMT | Tue, 08 Jul 2014 06:27:57 GMT | Tue, 08 Jul 2014 06:28:27 GMT | |
| 16 | 0000001-140708174838379-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Tue, 08 Jul 2014 06:19:10 GMT | Tue, 08 Jul 2014 06:19:45 GMT | Tue, 08 Jul 2014 06:20:36 GMT | |
| 17 | 0000000-140708174838379-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Tue, 08 Jul 2014 06:01:47 GMT | Tue, 08 Jul 2014 06:01:58 GMT | Tue, 08 Jul 2014 06:02:37 GMT | |
| 18 | 0000003-140707180534862-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Mon, 07 Jul 2014 07:08:49 GMT | Mon, 07 Jul 2014 07:08:57 GMT | Mon, 07 Jul 2014 07:09:26 GMT | |
| 19 | 0000002-140707180534862-oozie... | FuelWorkflow | KILLED | 0 | hadoop | | Mon, 07 Jul 2014 07:06:56 GMT | Mon, 07 Jul 2014 07:07:06 GMT | Mon, 07 Jul 2014 07:07:34 GMT | ▼ |

Page 1 of 1

1 - 25 of 25

Druid 기반 데이터 플랫폼 예시

빅데이터



Ambari



Zeppelin



druid



APACHE
ZooKeeper™



druid

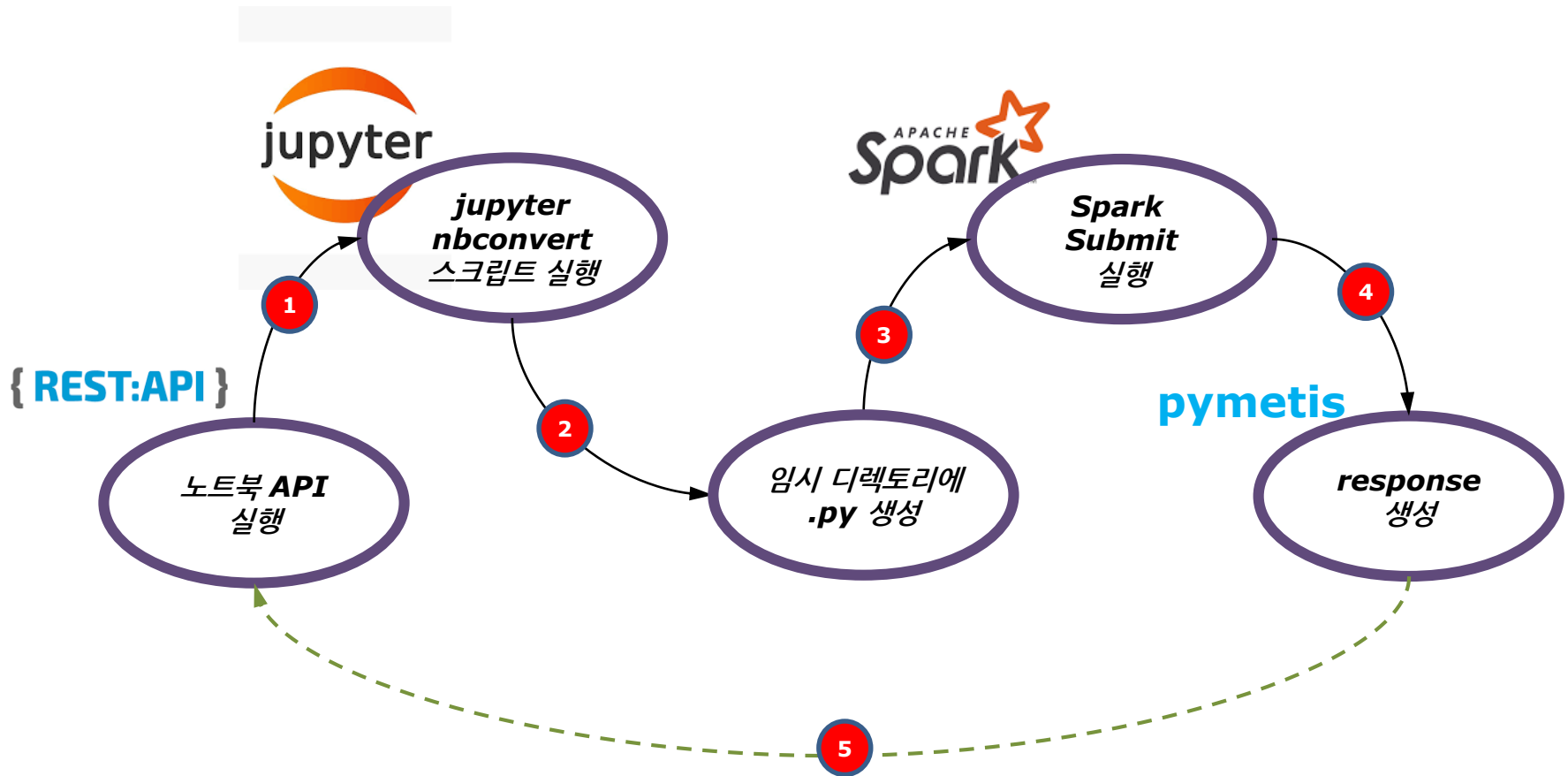


APACHE
ZooKeeper™

노트북 활용

빅데이터

Notebook API



국내 Open Platform

통합 데이터 지도

KDX 한국데이터거래소