

# 대용량 로그분석, BigQuery로 간단히 사용하기

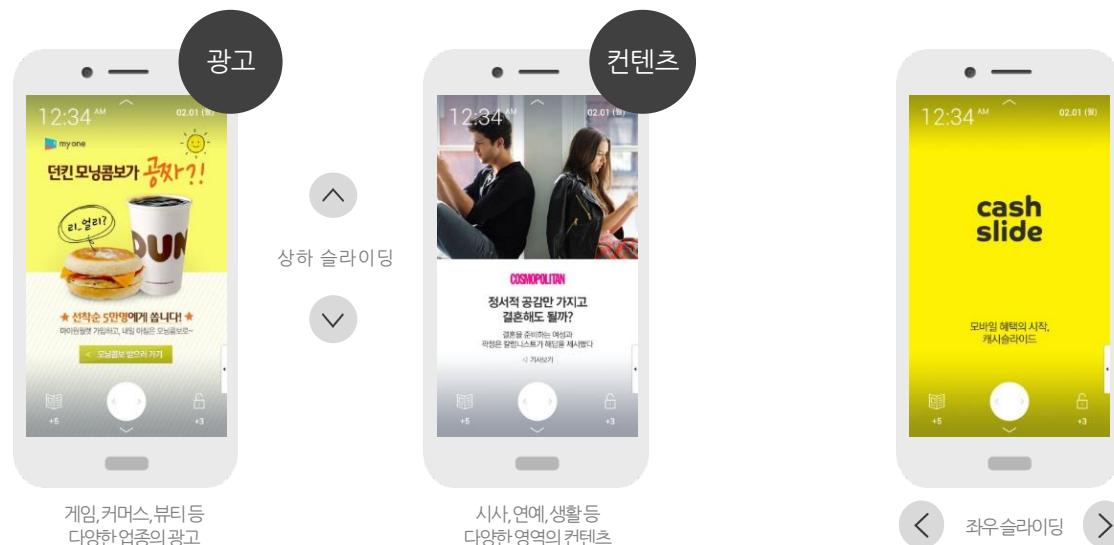
SK T아카데미 (2017. 02. 15)

# Cashslide Service

전세계 최초 모바일 잠금화면에서 광고, 컨텐츠 등을 보고  
**잠금화면을 열 때 캐시 적립**을 받을 수 있는 모바일 서비스

잠금화면 상하 슬라이딩을 통해  
다양한 컨텐츠를 풀스크린으로 감상

잠금화면 좌우 슬라이딩을 통해  
콘텐츠 더 보거나 잠금화면 해제



- 잠금화면을 좌우로 밀 때마다 3~5원 캐시 적립 혜택
- 앱 다운로드, 영상광고 시청 등 추가활동 하면 최대 1500원까지 적립

# Cashslide Service

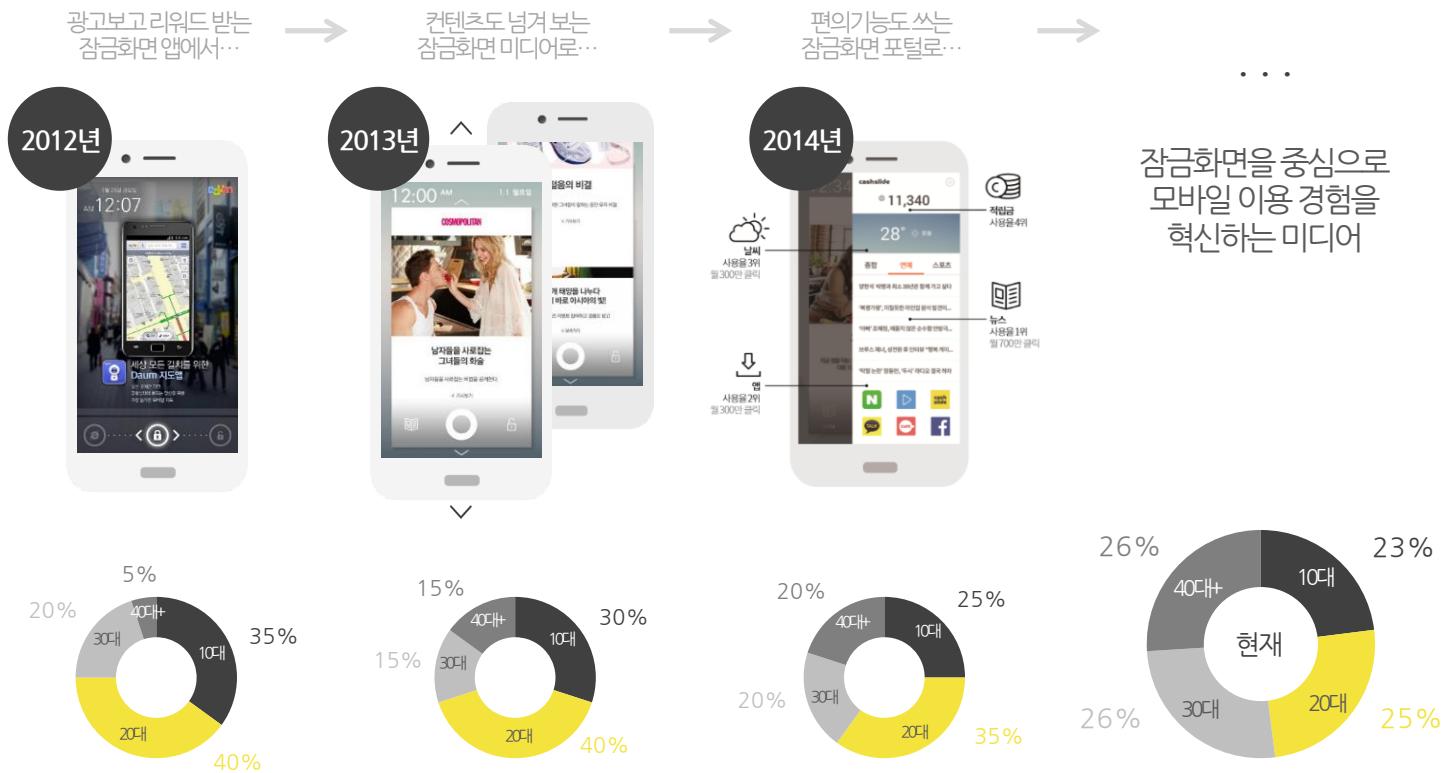
광고 이외에 뉴스, 스포츠 정보, 엔터테인먼트 등  
폭넓은 카테고리의 컨텐츠를 종합적으로 제공



전체 노출량의 40%, 전체 유저의 60% 이상이 사용하는  
캐시슬라이드 서비스의 핵심 value

# Cashslide Service

지속적인 개편을 통해  
국내 유일무이한 모바일 서비스로 성장하고 있습니다



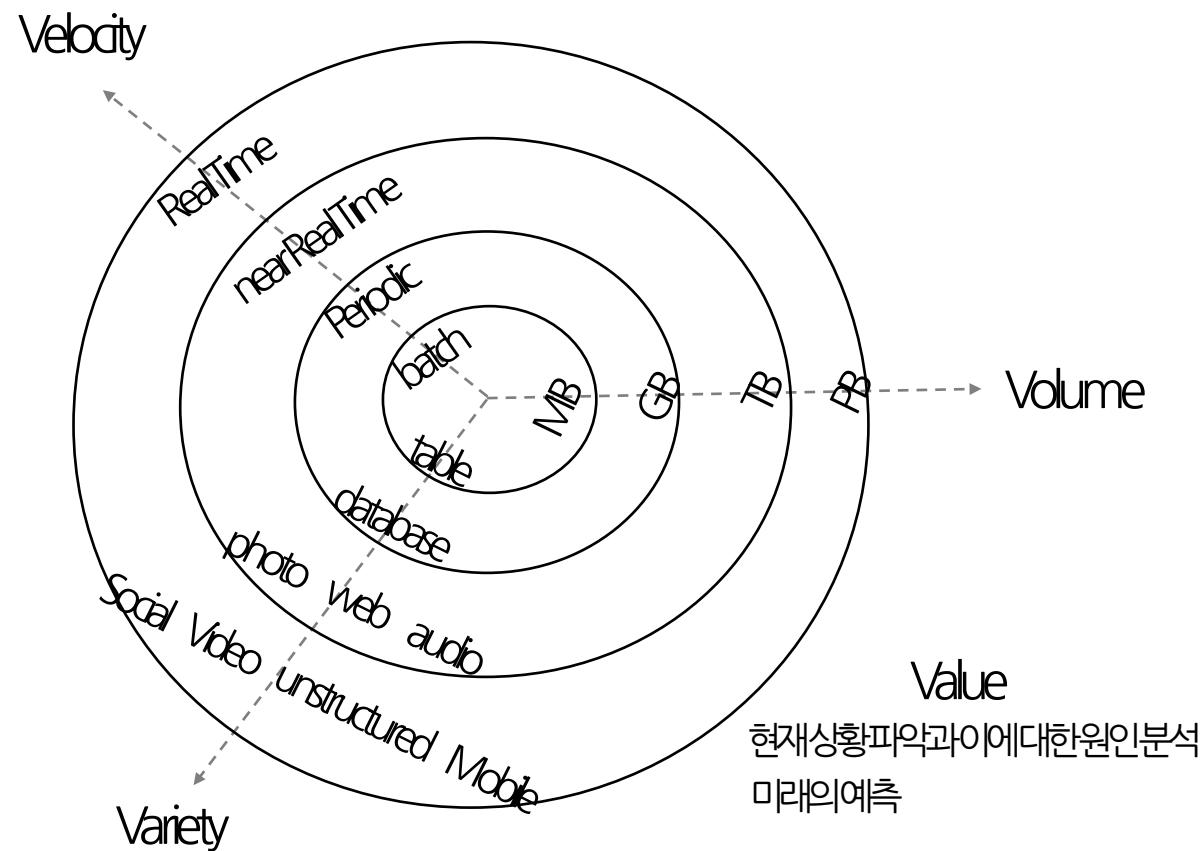


# Big Data

빅데이터란

기존데이터베이스 관리 도구의 능력을 넘어서는 대량(수십 테라바이트)의 정형 또는 심지어 데이터베이스 형태가 아닌 비정형의 데이터 집합 조차 포함한 데이터로부터 가치를 추출하고 결과를 분석하는 기술이다. 발췌: 위키

# Bigdata의 조건 "three Vs" 밸류:TDW



# 남들이 하고 있는 Big Data 활용 (B2C)

facebook



LinkedIn

Sangkyoon Nam's LinkedIn profile. He is a Chief Engineer at NBT. A callout box highlights a feature: "아는 사람 찾아 1촌 맺기" (Find people you know and connect) with the note "70%의 채용이 인맥을 통해 이루어집니다." (70% of hiring is done through connections).

Amazon.com

Amazon.com product page for the Logitech G Pro Gaming FPS Mouse. The price is \$53.00. The page shows related products like the Corsair MM300 mouse and an ASUS monitor.

**Saved for later (1 item)**

**Logitech G Pro Gaming FPS Mouse with Advanced Gaming Sensor for Competitive Play by Logitech** \$53.00

In Stock  
Eligible for FREE Shipping  
Delete | Move to Cart | Move to Wish List

The price and availability of items at Amazon.com are subject to change. The Cart is a temporary place to store a list of your items and reflects each item's most recent price. [Learn more](#)  
Do you have a gift card or promotional code? We'll ask you to enter your claim code when it's time to pay.

**Customers Who Bought Logitech G Pro Gaming FPS Mouse with Advanced Gaming Sensor for Competitive Play Also Bought**

Corsair Gaming MM300...  
ASUS VG248QE 24" Full...  
Logitech G403 Prodigy...  
Logitech G303 Daedalus...

**Your Recently Viewed Items and Featured Recommendations**

Inspired by your purchases

# 남들이 하고 있는 Big Data 활용 (B2B)

새로 만들기      저장된 타겟 사용 ▾

맞춤 타겟 ⓘ

기존 비즈니스 고객에게 광고 타겟팅  
맞춤 타겟을 만들어 광고를 연락처, 웹사이트 방문자 또는 앱 사용자에게 표시할 수 있습니다. 맞춤 타겟 만들기.

위치 ⓘ      이 위치의 모든 사람 ▾

미국  
미국  
포함 ▾ | 위치 추가

일괄 위치 추가...

연령 ⓘ      18 ▾ - 65+ ▾

성별 ⓘ      전체 남성 여성

언어 ⓘ      언어 입력

상세 타겟팅 ⓘ      다음 중 하나 이상과 일치하는 사람 포함 ⓘ

인구 통계학적 특성, 관심사 또는 행동 추가 | 추천 | 찾아보기

취미 및 활동

- ▶ 고통 수단
- ▶ 미술과 음악
- ▶ 애완동물
- ▶ 여행
- ▶ 인테리어 및 정원 가꾸기
- ▶ 정치 및 사회 문제

최신 이벤트

연결 관계 ⓘ

This screenshot shows a user interface for creating a targeted audience. At the top, there are two tabs: '새로 만들기' (Create New) and '저장된 타겟 사용' (Use Existing Target). Below this, a callout box provides information about creating a target for existing business customers. The main section is titled '맞춤 타겟' (Custom Target) and includes fields for location ('미국'), age ('18 - 65+'), gender ('전체'), and language ('언어 입력'). Below this, there's a '상세 타겟팅' (Detailed Targeting) section with a '인구 통계학적 특성, 관심사 또는 행동 추가' (Add Demographic, Interest, or Behavior) button. A sidebar lists various interests like hobbies, travel, and politics, each with a checkbox. At the bottom, there's a '연결 관계' (Relationship) section.

# 우리도 하기 위해 고민했던 것들

\* 우리에게 필요한 분석 결과로 보고 싶은 내용이 무엇인가?

- 서비스 지표 변화 추이(나 자신을 알자)
- 사용자 타겟팅을 위한 사용자 분석 (CTR, CVR을 높이자)
- 정밀 타겟팅을 위한 사용자 행동 분석 (CTR, CVR을 더 높이자)
- 우리가 시도하는 사업들에 대한 검증

데이터의 모수는 많을수록 좋고 이 데이터 어딘가에는 위의 내용들이 포함되어야 한다.

# 빅데이터 분석을 위한 전략 예제

**strength**

- Cashside UV/Profile 정보
- 광고 및 서비스 반응력

**weakness**

- 인력 및 경험 부재 (데이터 분석 / 예측 알고리즘)
- Data Quality 검증 필요

**opportunity**

- DSP(demand side platform) 보유
- AD serving 보유

빠른 검증이 가능한 환경

**threat**

- 경쟁사 / 신규 사업자 시장 진출 가능성
- 개인정보 정책 이슈

# 구축 방안은 간단히

정한다. 모은다.

가공 & 분석한다.

예측 & 자동화한다.

# 빅데이터 활용 (서비스 지표 분석)

Cashslide Metrics

# 빅데이터 활용 (타겟팅)

- \* user profile 기준 타겟팅 그룹 분류
  - 성별, 나이, 결혼여부
- \* device profile 기준 타겟팅 그룹 분류
  - 앱 실행&설치 이력을 활용한 유사도 그룹 분류
- \* user activity 패턴 기준 타겟팅 그룹 분류
  - 상품을 구매한 유저의 패턴과 유사도가 높으나 아직 구매하지 않은 사용자 그룹
  - 추천인, 피추천인
  - LTV 높은 유저와 CVR 높은 유저
- \* 실험에 대한 사용자 반응도 분석

# 빅데이터 활용 (실험 결과 검증)

The screenshot shows a Confluence page titled "데이터 분석 정보" (Data Analysis Information) located under the "07. 유용한 정보" (Useful Information) section. The page is part of the "DEV" space and was last modified on December 4, 2015. It was created by Sookyoung Choi. The page content lists various data analysis topics, each preceded by a blue link [DA#].

Topics listed:

- [DA#1-1] TV 광고 후 신규 가입자 및 DAU 변화
- [DA#1-2] TV광고 후 신규 가입자 및 DAU 변화
- [DA#2] 캐슬 사용자의 클릭(좌 슬라이드)에 대한 이모저모
- [DA#3] 가입시 추천인 입력 여부에 따른 잔존율 분석
- [DA#4] 돌아온 휴면 유저에 대한 분석
- [DA#5] 가입시 추천인 입력 여부에 따른 피추천 횟수 분석
- [DA#6] 리타게팅 광고 반응 유저의 타광고 반응성 분석
- [DA#7] 최근 유저당 노출(PV)이 줄어드는 이유
- [DA#8] 좌 슬라이드(클릭) 안하는 유저가 캐슬을 사용하는 방법
- [DA#9] 플로팅 사용 패턴 분석
- [DA#10] 스토리카드 사용 분석
- [DA#11] 동영상 시청형 광고 개발 그후 3개월
- [DA#12] 캐슬러는 언제 구매하는가
- [DA#13] 이탈자 특성 찾기
- [DA#14] 사용자 잔존여부와 적립금은 관계가 있을까?
- [DA#15] 한 달 동안 사용하면 얼마를 적립하는지...
- [DA#16] 광고 유형에 따른 반응 유저 분포
- [DA#17] 로또 구매에 대한 이모저모
- [DA#18] 로또 구매와 잔존율은 관계가 있을까?
- [DA#19] 광고 타입별 반응 유저에 대한 이모저모
- [DA#20] CPV CPA 반응 유저 수 변화
- [DA#21] 올해 상반기에 사용을 중지한 유저에 대한 분석
- [DA#22] 구매경험은 잔존율에 영향을 미치는가 (feat. JinsooHwang)
- [DA#23] push 메시지에 반응해줘요 제발~ (feat. Jinsoo Hwang)
- [DA#24] 설치형에 대한 모든 것 (사용자 튜토리얼 과제에 바침)
- [DA#25] NW 상태에 따른 사용패턴 분석
- [DA#26] 실행형 경험여부는 잔존율에 영향을 미치는가 (feat. JinsooHwang)
- [DA#27] 사용을 중지하는 유저의 유형

Page footer: Powered by Atlassian Confluence 5.5.4, Team Collaboration Software · Report a bug · Atlassian News

# Data Scientist

*The Sexiest Job of the 21st Century*

# 주변의 Data Scientist

*Full-Stack*

# Data Scientist와 Data Engineer

1. 요구사항 전달

2. 데이터 수집 및 저장 아키텍처 구현

3. 데이터 가공 및 분산 처리 아키텍처 구현

4. 사전 데이터 작업 및 배치 개발

5. 모니터링 및 기술 지원

6. 데이터 분석 및 시각화

# Data Scientist와 Data Engineer

LANG:java,scala,python,R,sql

빅데이터분석모형, 기법, 통계분석의 이해

비즈니스의 이해

고객과의 의사소통

LANG:java,scala,python,bash,sql

Cloud Platform, Hadoop, echo, Spark, Kafka, Zookeeper, ELK, Zeppelin, DFS, Cache, NoSQL, RDBMS 아키텍처 이해와 구현

OS와 Network에 대한 이해

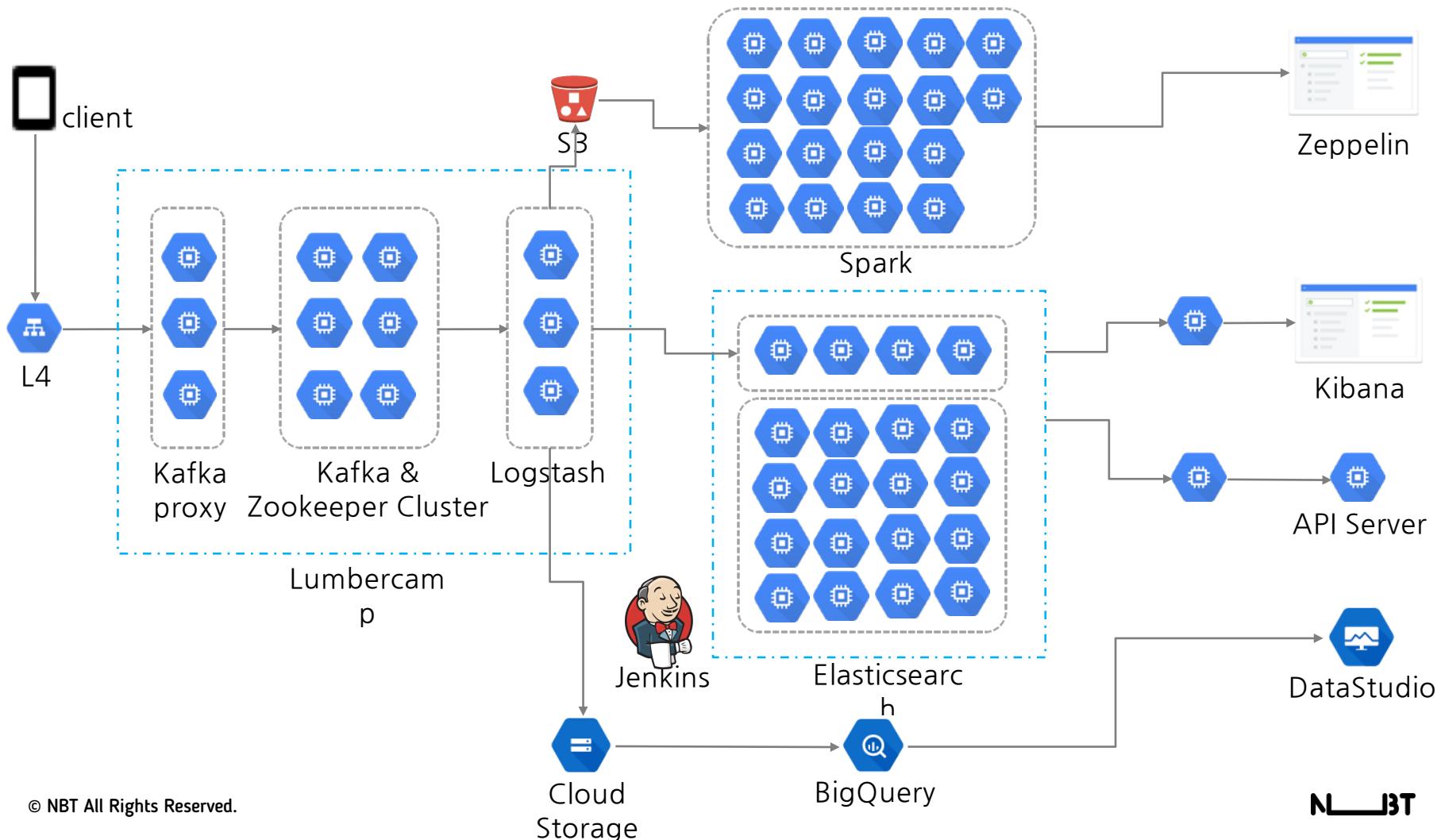
Infra, Security의 이해

JVM, Cluster Monitoring & Troubleshooting

우리가 분석하는 빅데이터 소스는 Client Log와 Server Log  
그리고 RDBMS 조합

# NBT Lumberjack log Analysis Architecture

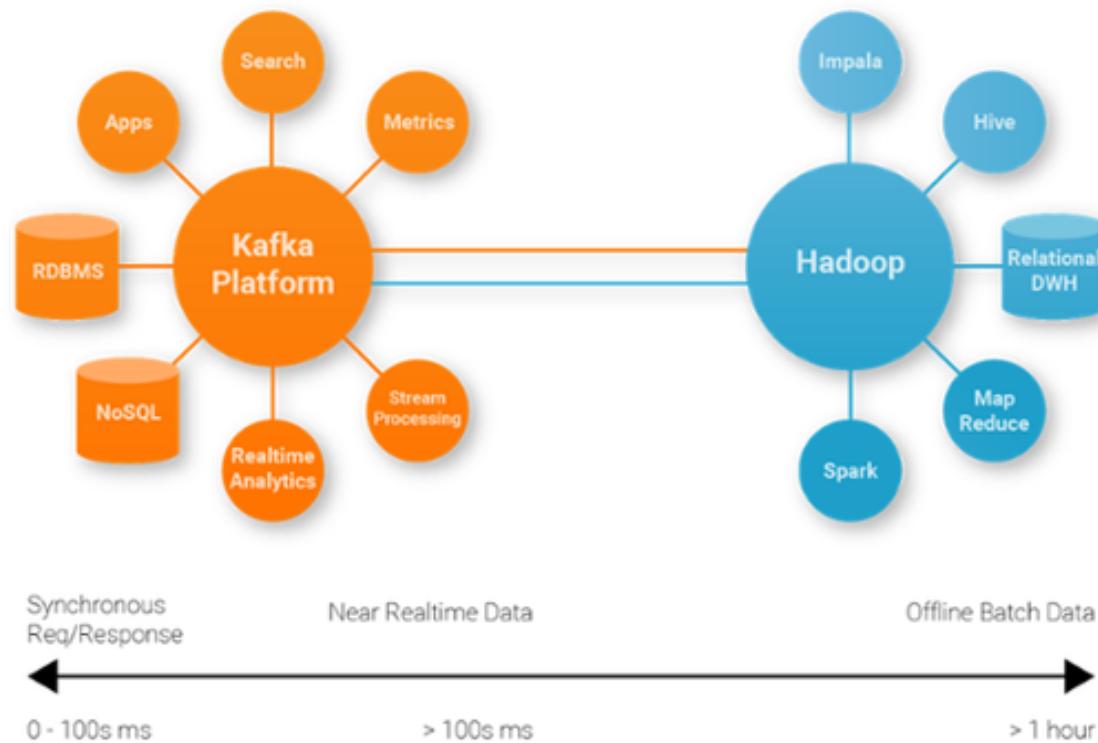
Kafka + Spark + Zeppelin+ELK + Bigqeury + DataStudio



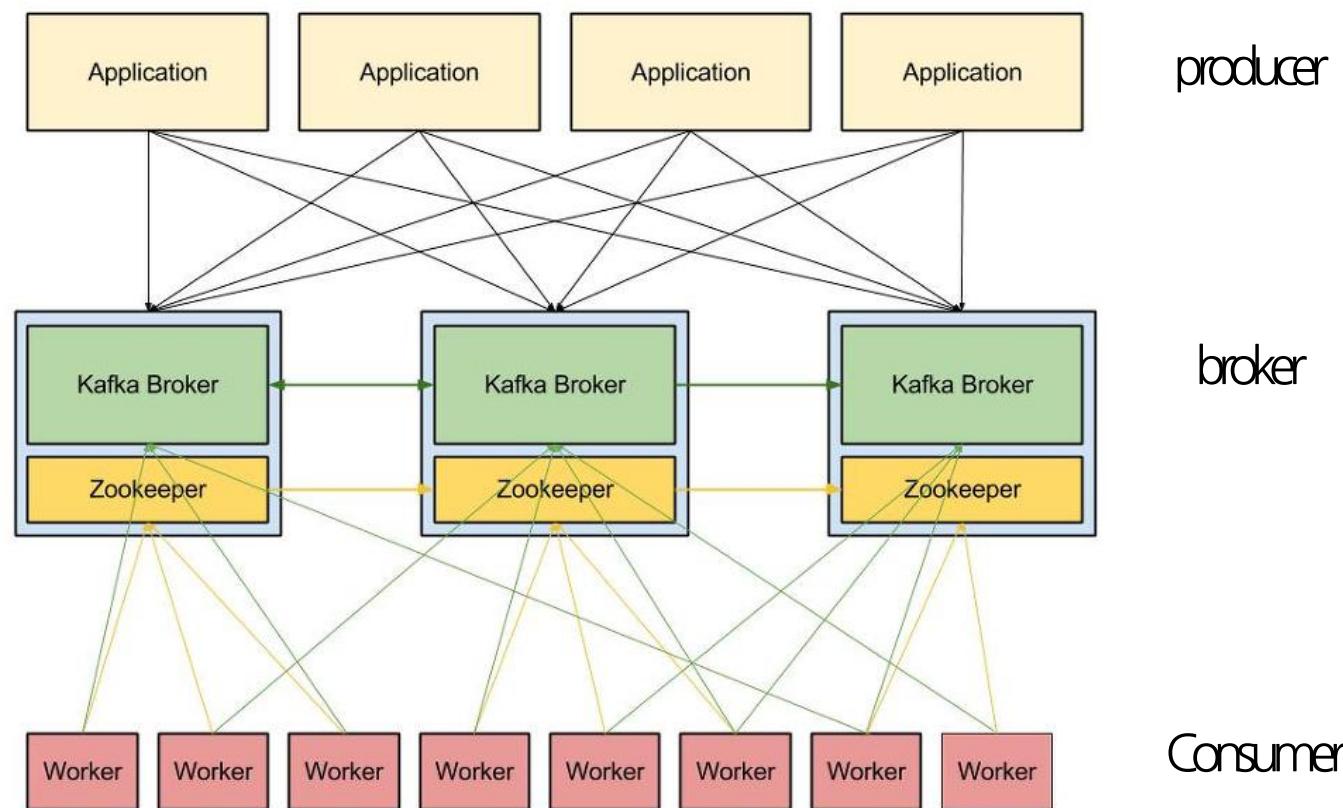
# 데이터 수집 시스템 'Kafka Cluster'

- \* LinkedIn에서 개발된 분산메시징 시스템
- \* 대용량의 실시간 로그 처리에 특화
- \* 범용메시징 시스템 대비 TPS가 높다.
- \* 분산, 복제가 손쉽다.
- \* 파일 시스템을 사용하므로 데이터 영속성이 보장된다.
- \* 2011년 오픈소스로 공개

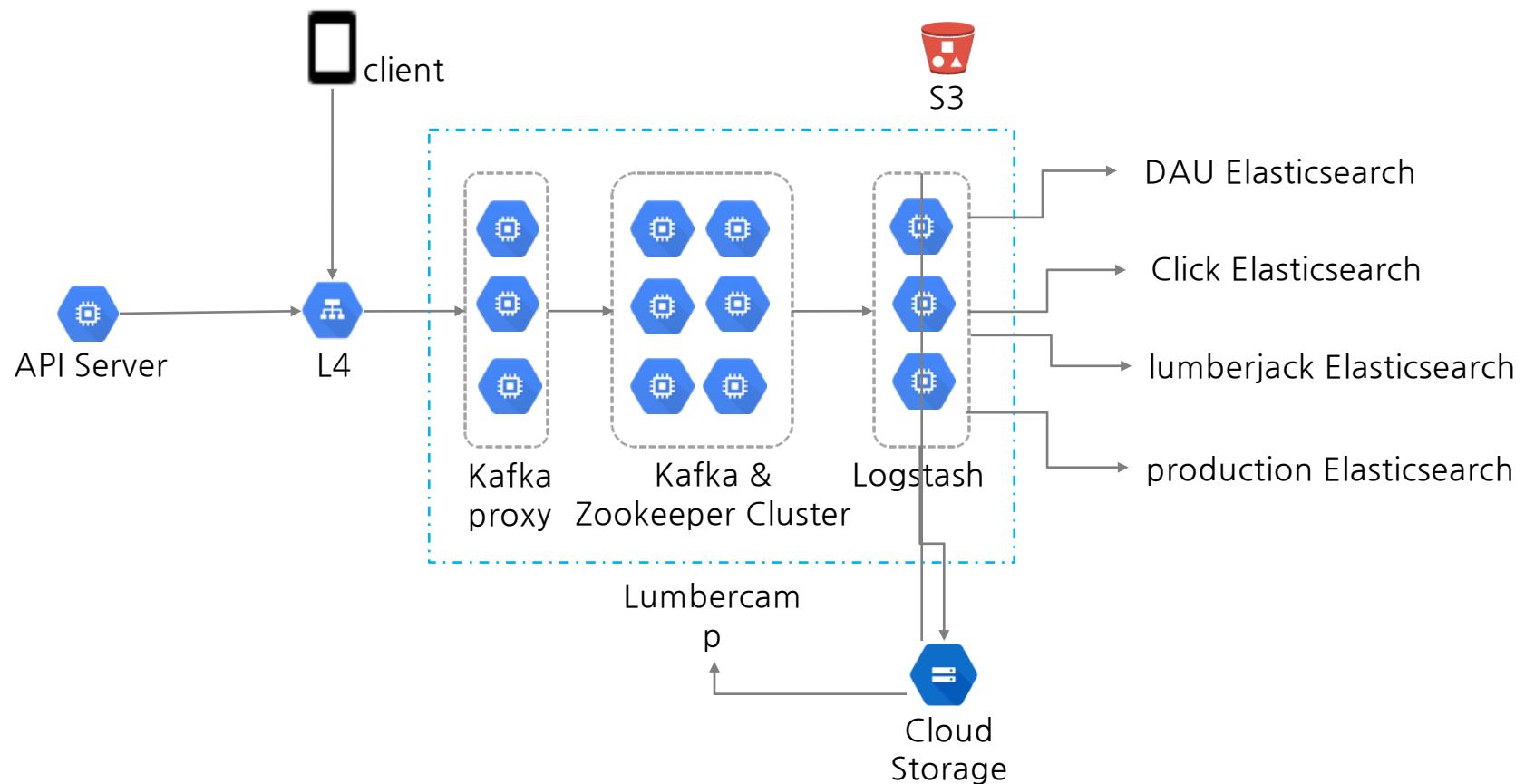
# 데이터 수집 시스템 'Kafka Cluster'



# 데이터 수집 시스템 'Kafka Cluster'



# NBT Kafka Cluster



# data transfer agent

|          | plugin ecosystem | event routing          | performance      |
|----------|------------------|------------------------|------------------|
| logstash | centralized      | algorithmic statements | uses more memory |
| fluentd  | decentralized    | Tags                   | uses less memory |

# Elasticsearch

- \* apache Lucene을 기반으로한 분산 검색 엔진
- \* Opensource : apache license 2.0
- \* Distributed & Horizontally Scalable
- \* 고가용성
- \* full text search 지원
- \* JSON Document based
- \* Schema free

# Elasticsearch 설치

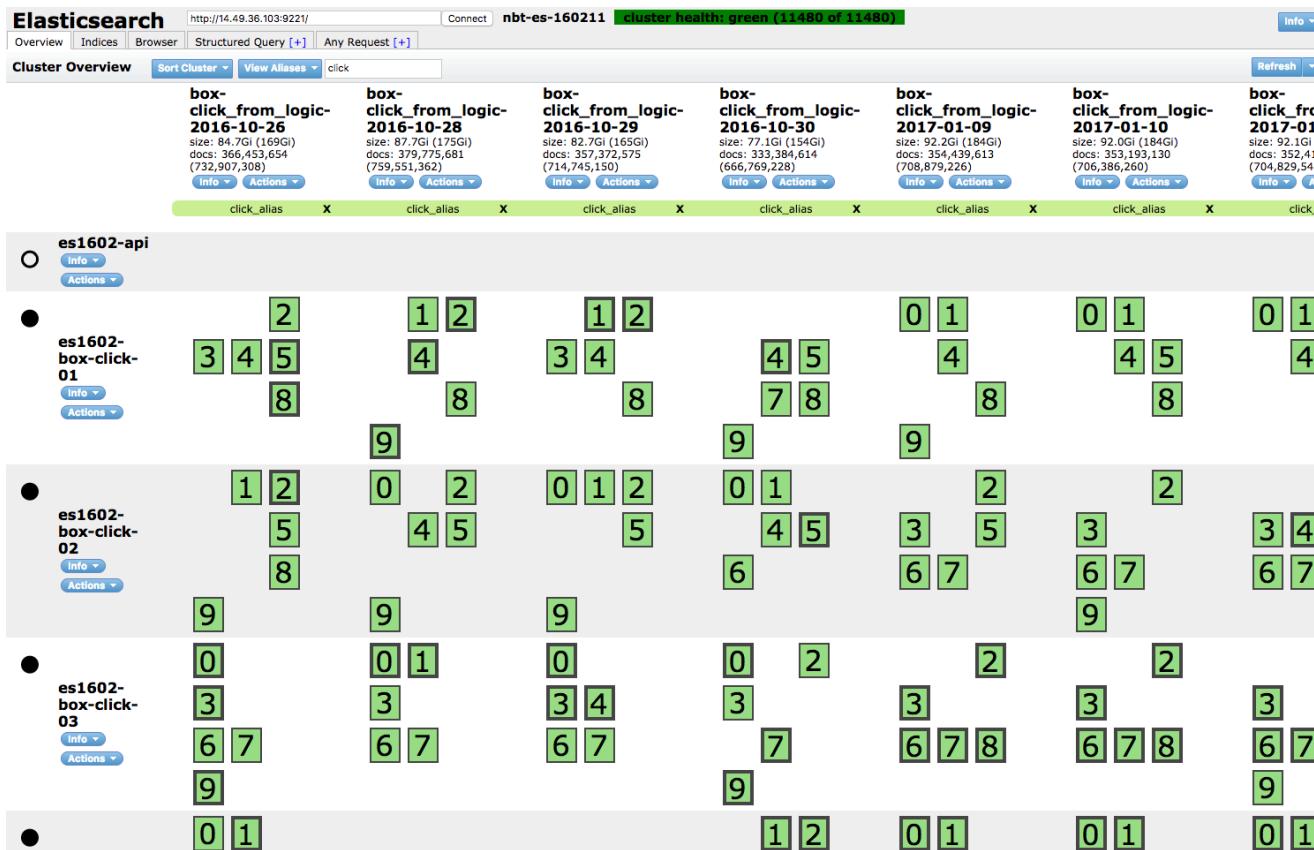
초간단 설치로 유명

1. download 받는다.
2. 실행한다.
3. 살아있는지 테스트한다.

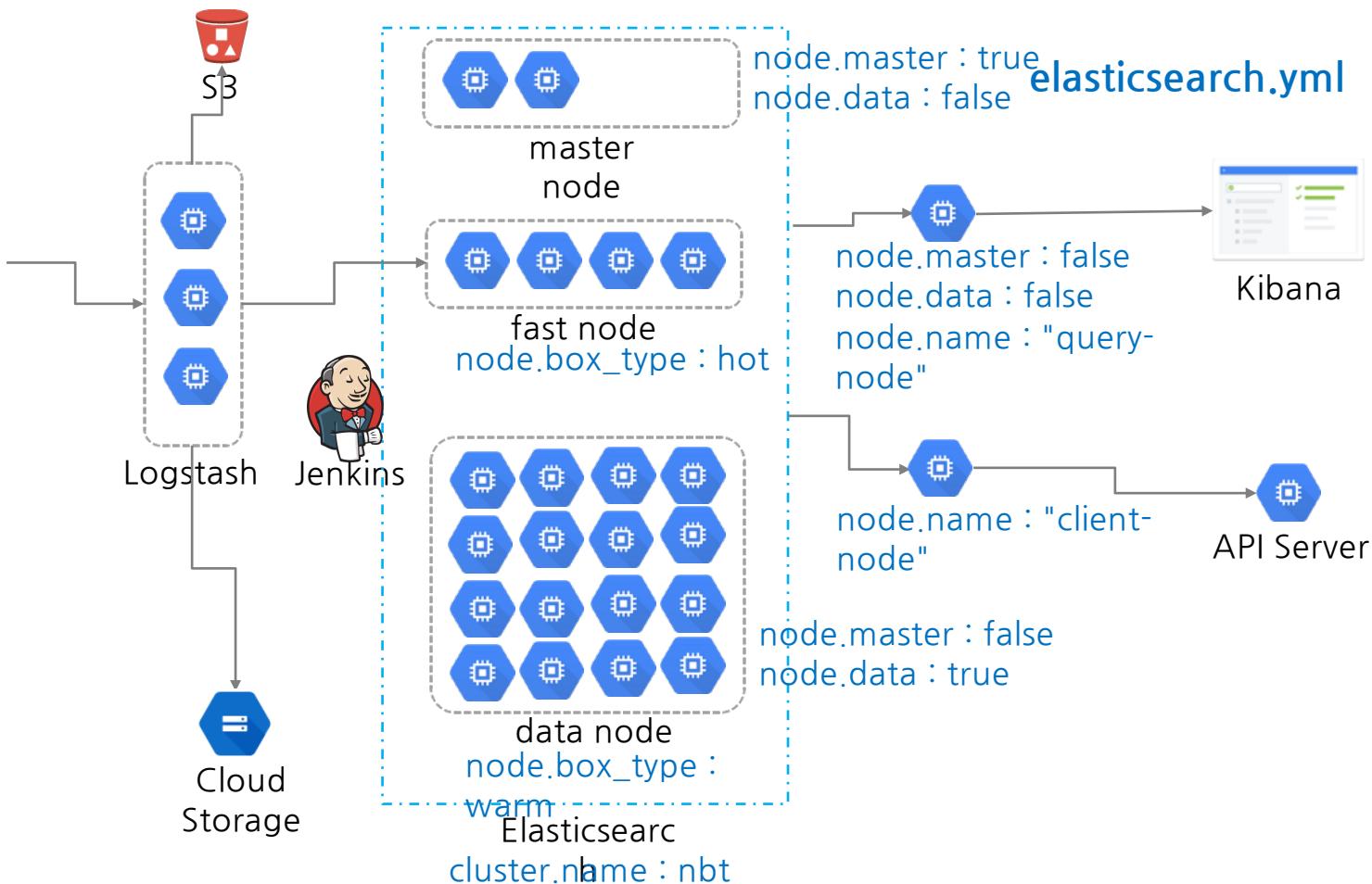
# Elasticsearch 용어

| RDB      | Elasticsearch |
|----------|---------------|
| database | index         |
| table    | type          |
| row      | document      |
| column   | field         |
| schema   | mapping       |

# Elasticsearch index, shard & replicas

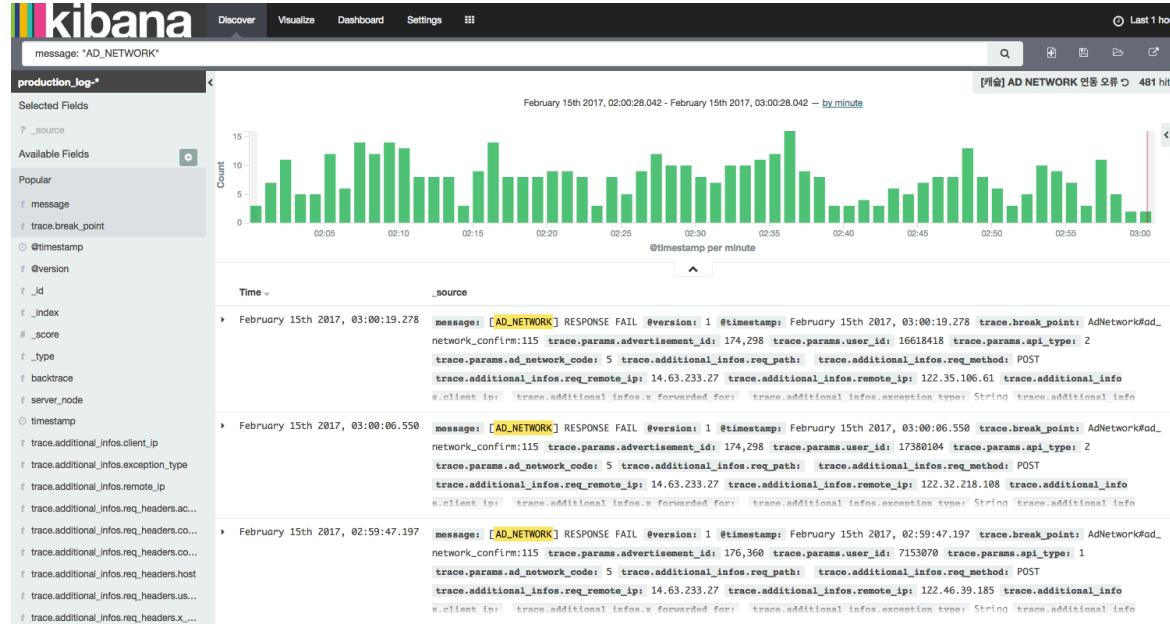


# NBT Elasticsearch



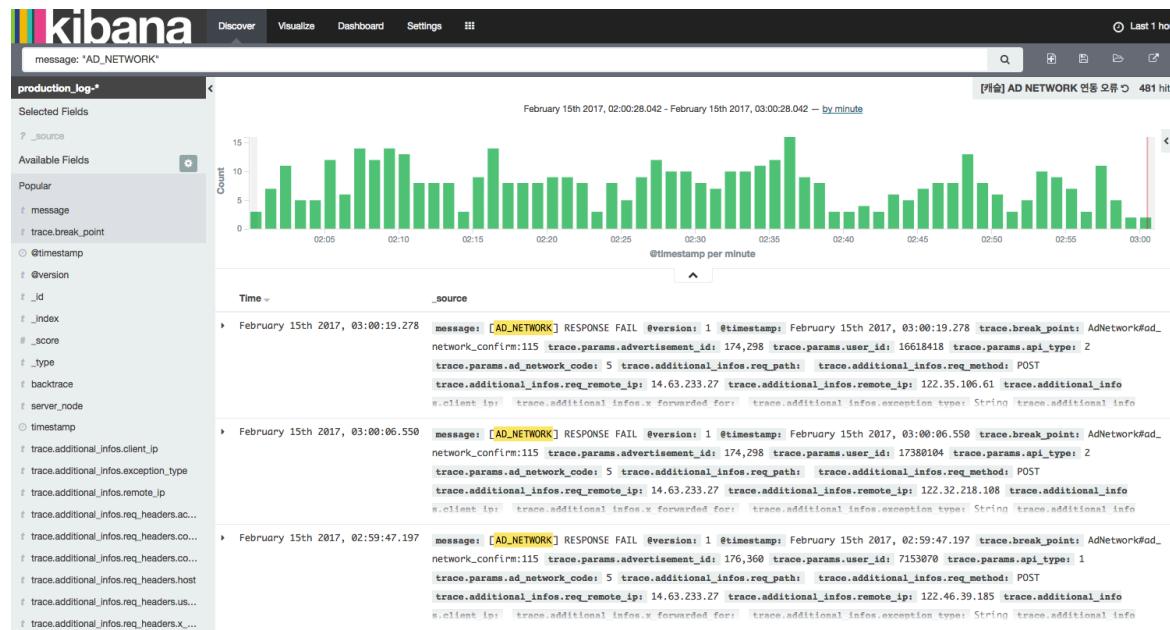
# Kibana

## Elasticsearch + Kibana Live Demo



# Kibana

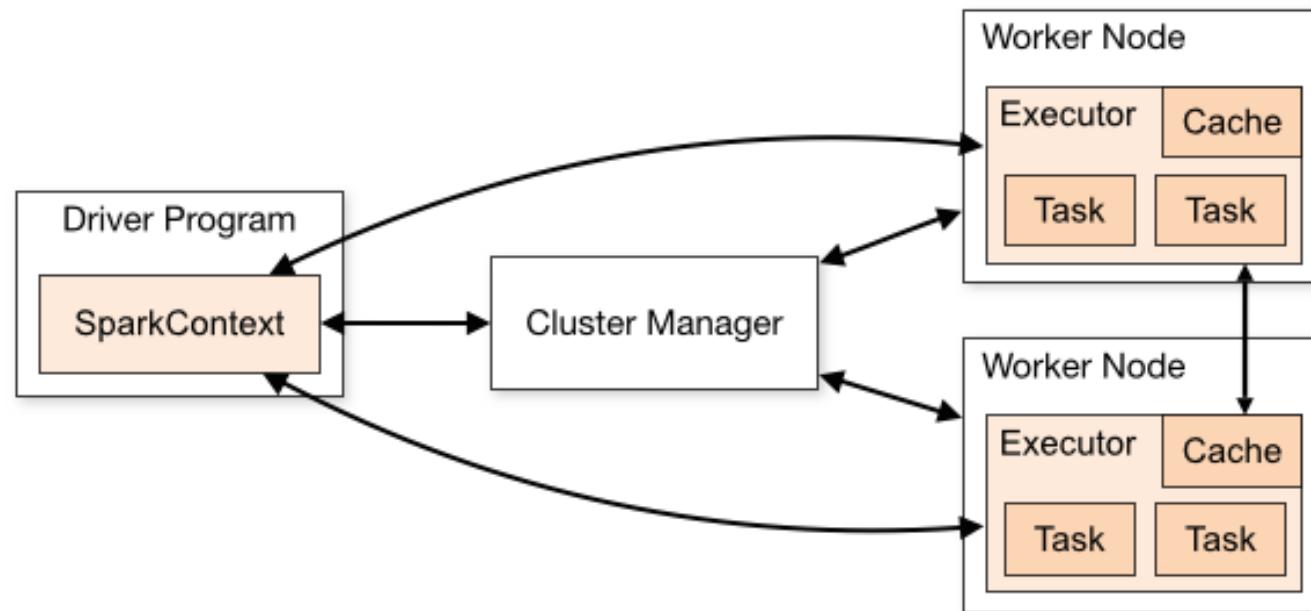
ELK의 최대 장점: 설치가 매우 쉽다.  
사용도 손쉽다. Lucene Query 기반 검색. (AND, OR, NOT)



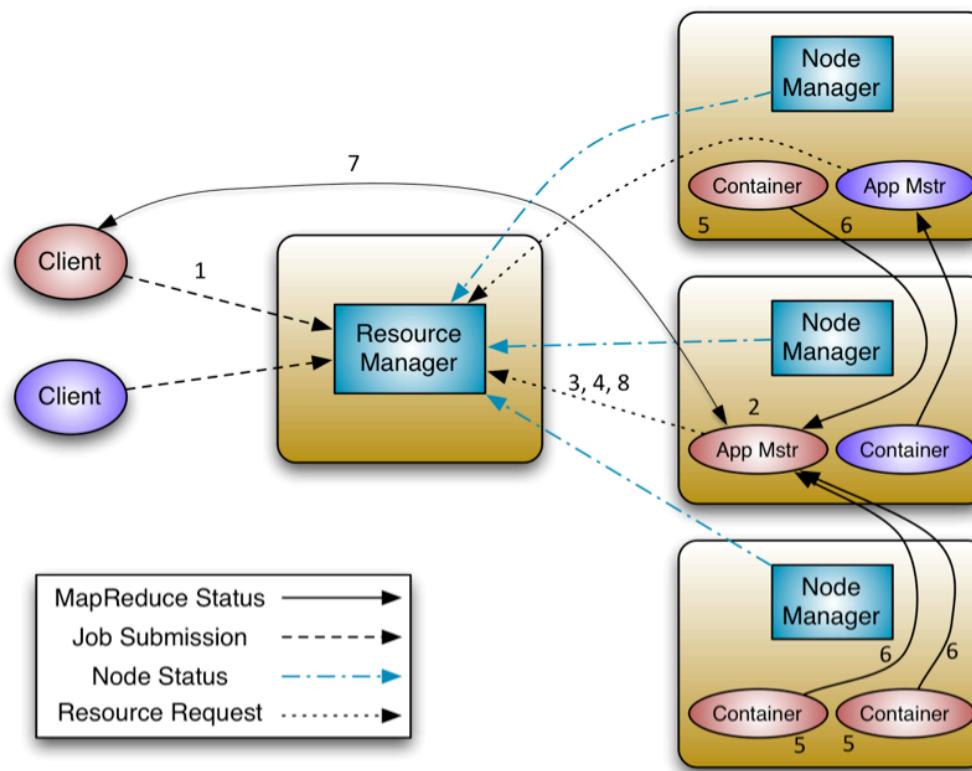
# Hadoop + Spark + Zeppelin

- \* 오픈 소스 분산 처리 시스템
- \* 메모리 기반 (hadoop에 비해 속도가 빠르다)
- \* Mapreduce code(java)에 비해 매우 간결한 code 작성이 가능하다. (scala)
- \* java, scala, python, R
- \* Apache Zeppelin으로 손쉽게 사용 가능하다.
- \* Apache Zeppelin으로 손쉽게 사용 가능하다.
- \* Apache Zeppelin으로 손쉽게 사용 가능하다.

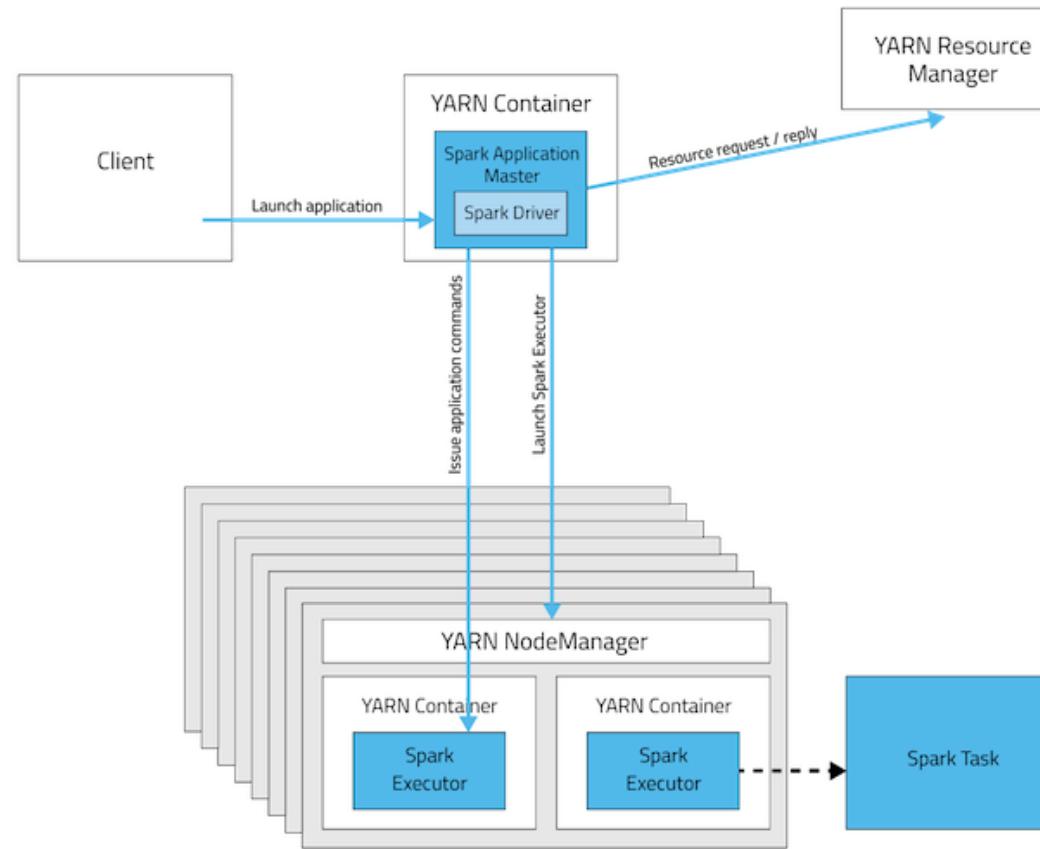
# spark execution flow



# spark 사용 시 알아야 할 개념 hadoop YARN flow

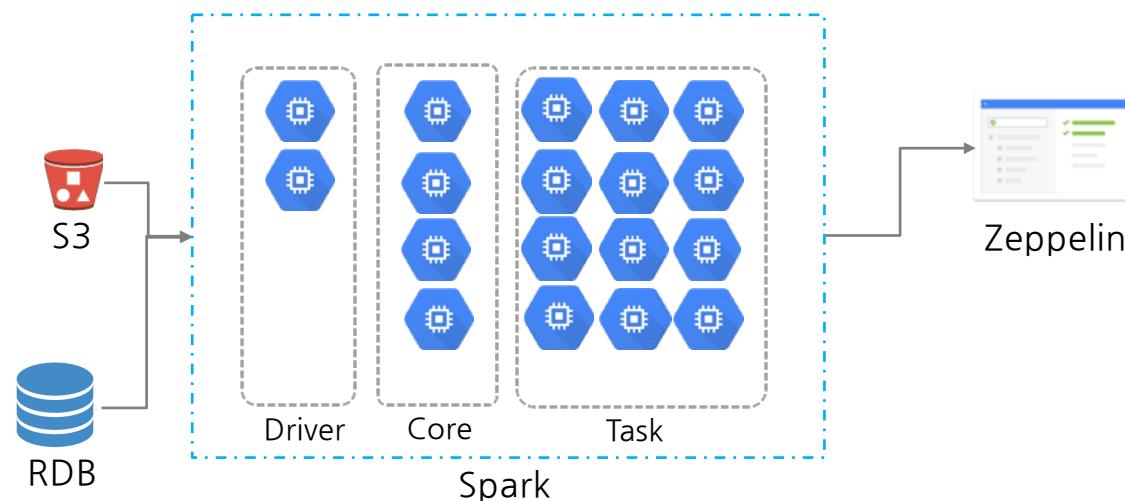


# hadoop YARN + spark



# 개념을 이해했다면 aws emr을 편하게 사용합니다.

- \* spot을 적절히 사용합니다.
- \* spot instance를 이용하여 schedule 기반 resize를 진행합니다.
- \* 외부 데이터를 불러오는 경우 Task를 다수로 구성하여 병렬처리 합니다.



# 그리고 Zeppelin을 사용하여 데이터를 분석합니다.

## Zeppelin live demo

 **Zeppelin** Notebook ▾

### Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.  
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

**Notebook** 

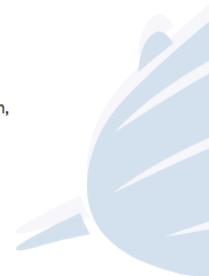
-  [Import note](#)
-  [Create new note](#)

-  [\[01. 시스템\]](#)
-  [\[02. 테스트\]](#)
-  [\[ADID\]설치형, 실행형 광고 참여 완료 유저](#)
-  [\[ADID\]설치형, 실행형 광고 참여 완료 유저 with DEVICE\\_ID](#)
-  [\[ADID\]추천실행형](#)
-  [\[CCCS\] 구매 포스트백 유저 닉네임 추출 \(푸쉬 발송용\)](#)
-  [\[CCCS\]구매 대상자 PUSH 용 토큰 추출](#)
-  [\[CCCS\]시간별 \( 노출 횟수, 주문 포스트백 \)](#)
-  [\[CCCS\]시간별 \( 주문 포스트백 \)](#)
-  [\[광고\]광고 별 유니크 클릭 유저 수](#)
-  [\[광고\]광고 컨텐츠 연관성 분석](#)
-  [\[광고\]광고별 Frequency \(클릭수, 유저수,설치수\)](#)
-  [\[광고\]광고별 클릭로그의 타임스탬프 추출](#)
-  [\[광고\]날짜별 UV, 클릭](#)
-  [\[광고\]동적 소재 리포트](#)
-  [\[광고\]동적소재](#)
-  [\[광고\]성별, 연령별 광고 UV](#)
-  [\[광고\]성별, 연령별 광고 도달률 데이터 \(오토뷰 전용\)](#)
-  [\[광고\]성별, 연령별 광고 도달률 데이터 \(오토뷰 전용, 시간단위\)](#)
-  [\[광고\]성별, 연령별 광고 도달률 데이터 \(오퍼월 추가\)](#)
-  [\[광고\]성별, 연령별, 디바이스 모델별 MAU,WAU,DAU](#)
-  [\[광고\]성별,연령별 광고 도달률 데이터](#)
-  [\[광고\]앱 설치 로그](#)

**Help**  
Get started with [Zeppelin documentation](#)

**Community**  
Please feel free to help us to improve Zeppelin,  
Any contribution are welcome!

-  [Mailing list](#)
-  [Issues tracking](#)
-  [Github](#)



# 최근 선택한 Google Bigquery

# 우리가 경험했던 BigQuery

Google Cloud Account 생성이 제일 어려웠을 만큼 쉽게 구성 가능했던 BigQuery



# 손이 갈 만한 것들을 추려보자면

대략 몇군데만 손보면 금방 구현될 듯



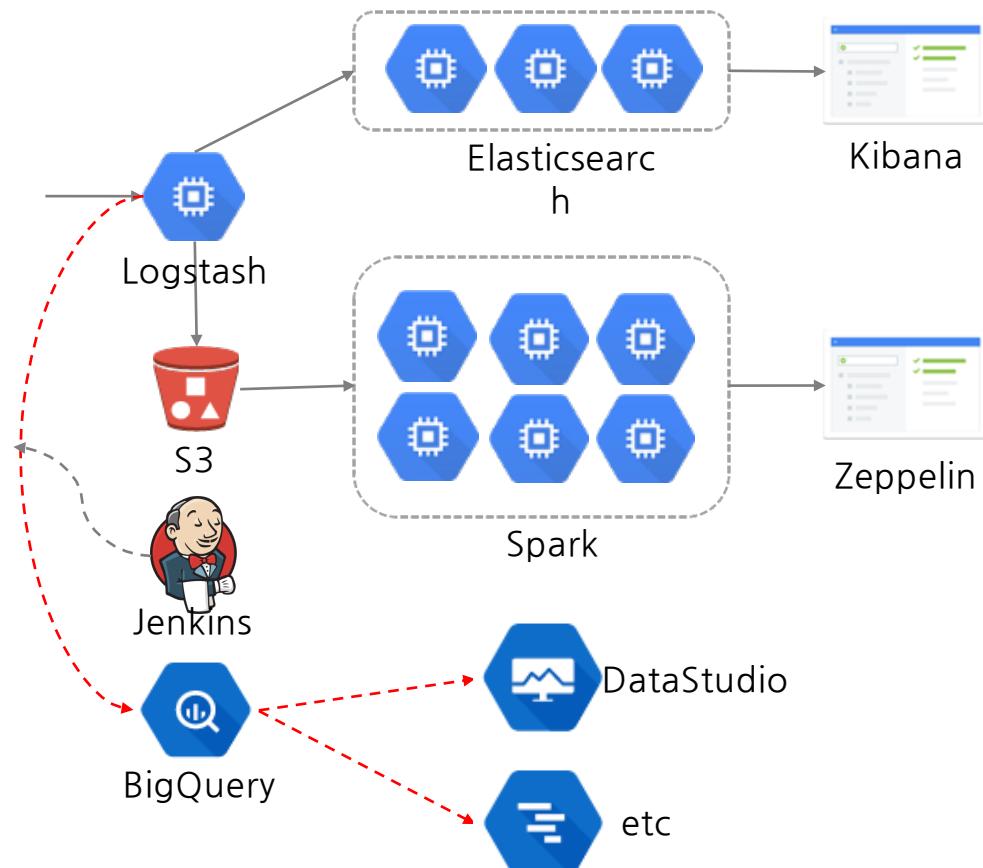
# proto-type 구현

이런 작업은 하루면 됩니다.



# 실전 구현 시나리오 작성

기존 logstash 서버에 저장된 json log file을 BigQuery로 전송 후 조회



## 고려 사항

- 서버의 자원 부담 최소화  
(현재도 부담 되는 중)
- 전송 처리 실패에 대한 보장
- 전송 처리 작업 시간 개선
- 전송 작업 이력 관리
- 네트워크 구간 전송 효율 개선

# BigQuery 구축 시 고민 1

BigQuery Schema 생성 시 ES의 Nested type에서 문제 발생

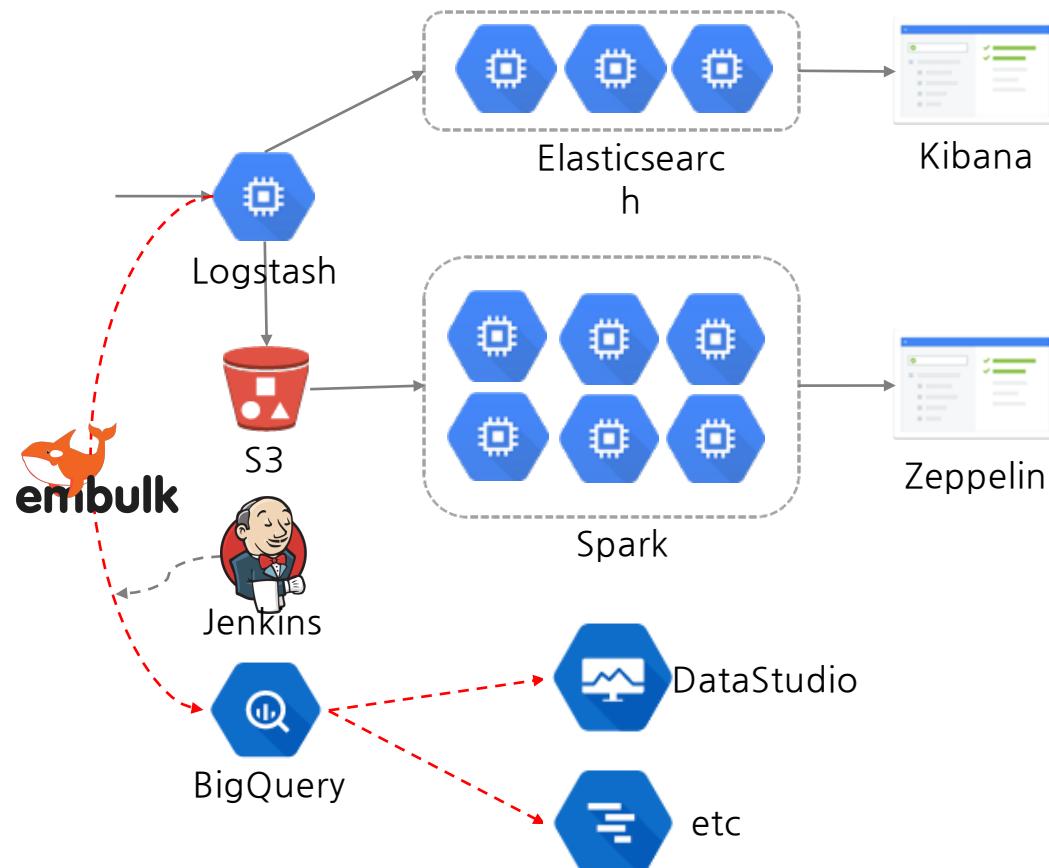
```
{  
    "name": "info",  
    "type": "RECORD",  
    "mode": "REPEATED",  
    "fields": [  
        {  
            "mode": "NULLABLE",  
            "name": "step_i",  
            "type": "INTEGER"  
        },  
        {  
            "mode": "NULLABLE",  
            "name": "recommended",  
            "type": "STRING"  
        },  
        {  
            "name": "app_config",  
            "type": "RECORD",  
            "mode": "REPEATED",  
            "fields": [  
                {  
                    "mode": "NULLABLE"  
                    "name": "adison_enables"  
                    "type": "STRING"  
                },  
                {  
                    "mode": "NULLABLE"  
                    "name": "abusing_devices"  
                    "type": "STRING"  
                },  
            ]  
        }  
    ]  
}
```

고려 사항

- Nested와 유사한 Repeated Mode
- 'info' column을 Record type의 Repeated Mode로 생성
- Repeated Mode의 column 안에 또 한번의 Repeated Mode는 지원 안됨. ES도 마찬가지.
- 이 경우는 String 처리에 대한 고민
- 일단 제외

# Embulk를 이용한 BigQuery Data 적재

json file size : 7 ~ 35G per hour, 7.5G json file loading 작업이 약 1시간 정도 소요



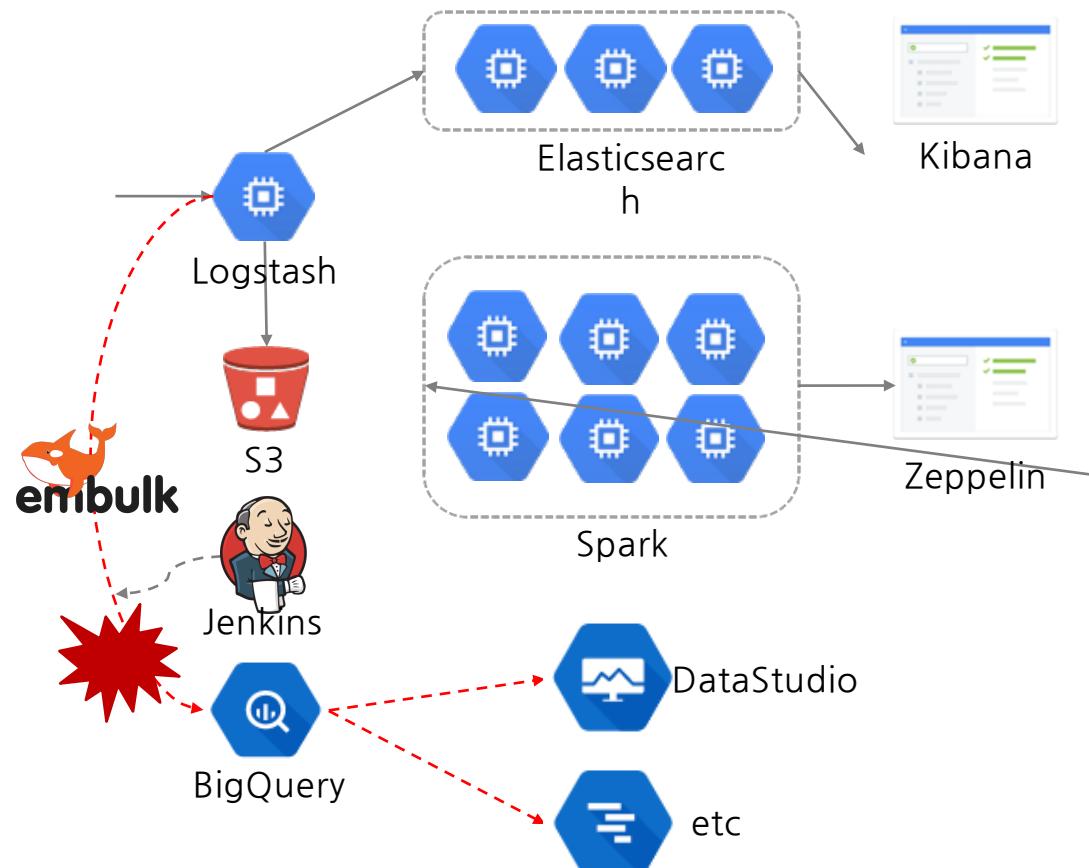
## 고려 사항

- 서버의 자원 부담 최소화  
(현재도 부담 되는 중)
- 전송 처리 실패에 대한 보장
- 전송 처리 작업 시간 개선
- 전송 작업 이력 관리
- 네트워크 구간 전송 효율 개선



# BigQuery 구축 시 고민 2

embulk를 도입했어도 Json file을 BigQuery로 적재하기에는 너무 느림

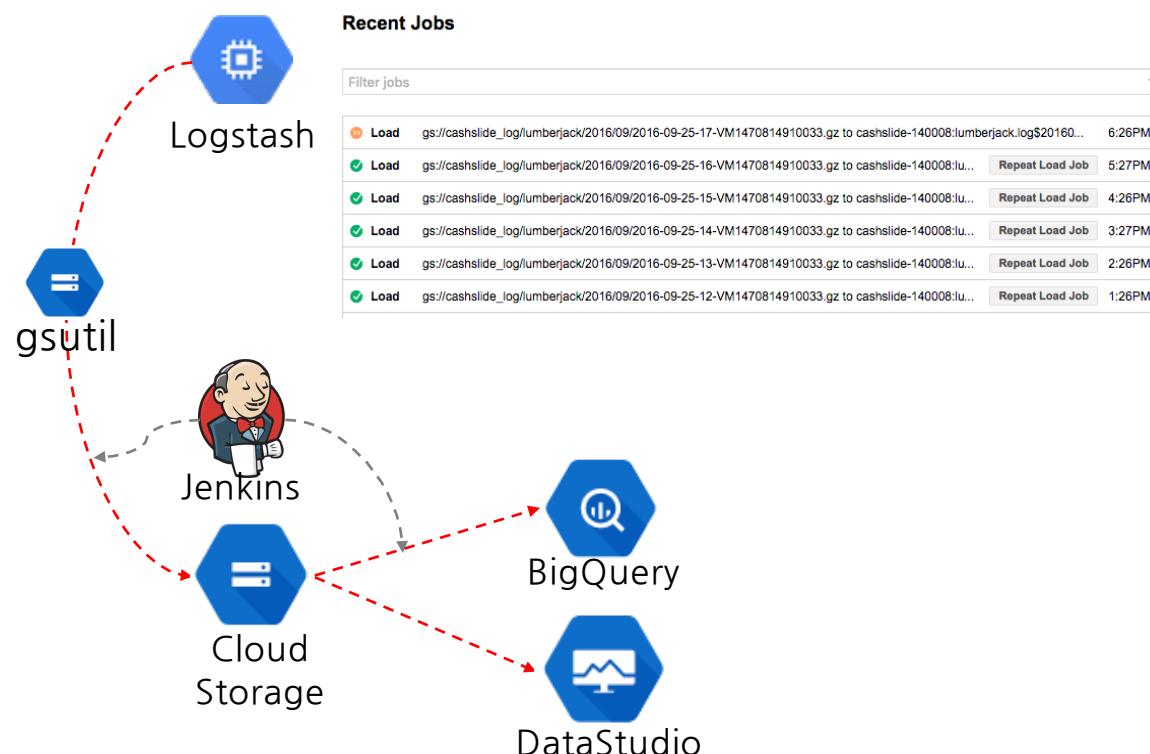


## 고려 사항

- 1시간 이상 걸리는 BigQuery Data 적재 작업
- Parsing 처리 비용이 너무 높다.
- 시간당 30G 이상의 파일을 전송하는 경우 발생하는 네트워크 전송량도 너무 많다.
- 바이너리 압축 전송을 하면 BigQuery Loading은?

# GCS to BigQuery

Jenkins를 이용해서 gz file을 GCS로 전송 후 BigQuery Loading Job을 추가 실행



## 해결

- Transport Tool을 embulk에서 gsutil로 교체
- gsutil을 이용한 바이너리 파일 병렬 전송으로 작업 속도 향상
- Gzip으로 GCS 전송 후 BigQuery로 Data 적재 작업을 수행하는 경우 Job 형태로 async 수행 및 관리 가능

# BigQuery의 Extract & Loading

BigQuery의 Computing Resource 비용은 무료

BigQuery pricing

| Action              | Cost                     | Notes  |
|---------------------|--------------------------|--|
| Storage             | \$0.02 per GB, per month | See <a href="#">Storage pricing</a> .  |
| Long Term Storage   | \$0.01 per GB, per month | See <a href="#">Long term storage pricing</a> .                                  |
| Streaming Inserts   | \$0.01 per 200 MB        | See <a href="#">Storage pricing</a> .  |
| Queries             | \$5 per TB               | First 1 TB per month is free, subject to <a href="#">query pricing details</a> . |
| Loading data        | Free                     | See <a href="#">Loading data into BigQuery</a> .                                 |
| Copying data        | Free                     | See <a href="#">Copying an existing table</a> .                                  |
| Exporting data      | Free                     | See <a href="#">Exporting data from BigQuery</a> .                               |
| Metadata operations | Free                     | List, get, patch, update and delete calls.                                       |

무료

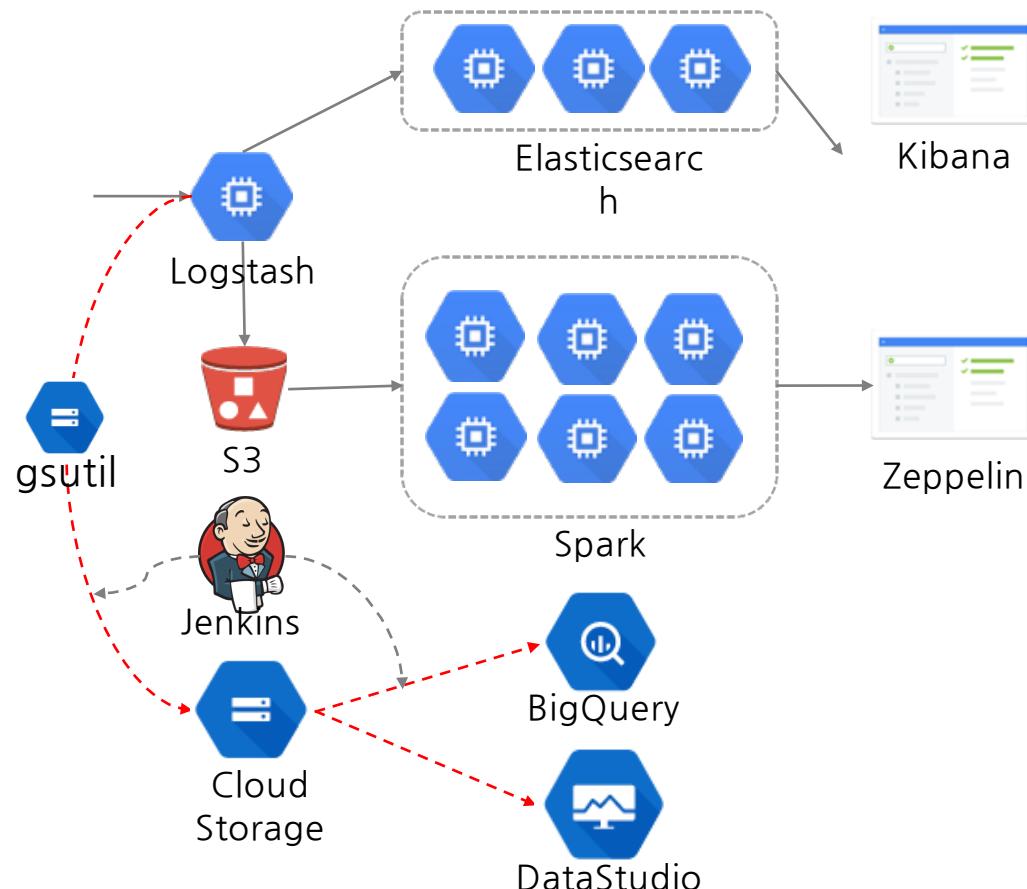
- 비용은 Storage와 Streaming Inserts, Queries 요금만 존재
- 속도도 빠르고 관리도 용이한 BigQuery Loading이 무료

Storage (GCS pricing)

| Standard Storage<br>(per GB per Month) | Durable Reduced Availability (DRA)<br>Storage<br>(per GB per Month) | Nearline Storage<br>(per GB per Month) |
|--|---|--|
| \$0.026                                | \$0.02  | \$0.01                                 |

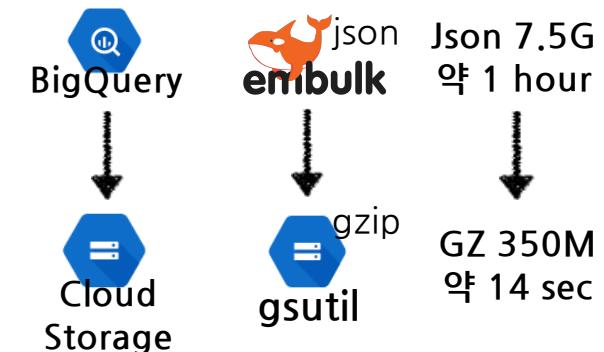
# 실전 구현 시나리오 변경

기존 S3로 전송하던 gz file을 GCS로 전송 후 BigQuery로 적재



## 변경된 시나리오

- json file → gzip file
- embulk → gsutil
- to BigQuery → to GCS
- sync → async



# BigQuery 조회 해보기

BigQuery 속도는 빠른가

✓ SELECT \* FROM lumberjack.ym\_2016\_08 where session\_id = 't147293'

```
1 SELECT
2   *
3   FROM
4     lumberjack.ym_2016_08
5   where
6     session_id = 't1472936324-26ca50eb-d122-4cf1-b07'
```

|                   |  |
|-------------------|--|
| Job ID            | cashslide-140008:bquijob_6ab18db4_15703d63fb1    |
| Creation Time     | Sep 7, 2016, 5:49:15 PM                          |
| Start Time        | Sep 7, 2016, 5:49:15 PM                          |
| End Time          | Sep 7, 2016, 5:49:21 PM                          |
| Bytes Processed   | 1.34 TB  |
| Bytes Billed      | 1.34 TB  |
| Billing Tier      | 1  |
| Destination Table | cashslide-140008:_11096407d334955b446b85f93f516c |

고려 사항

- 한달치 1.34TB data를 스캔해서 result 까지 걸린 시간 6초
- 7명이 동시에 쿼리를 수행해도 동일하게 빠른 성능을 보여줌
- 그런데 한번 조회에 1.34TB면 얼마지?

# BigQuery 구축 시 고민 3

## How much BigQuery

| Action              | Cost                     | Notes  |
|---------------------|--------------------------|--|
| Storage             | \$0.02 per GB, per month | See <a href="#">Storage pricing</a> .  |
| Long Term Storage   | \$0.01 per GB, per month | See <a href="#">Long term storage pricing</a> .                                  |
| Streaming Inserts   | \$0.01 per 200 MB        | See <a href="#">Storage pricing</a> .  |
| Queries             | \$5 per TB               | First 1 TB per month is free, subject to <a href="#">query pricing details</a> . |
| Loading data        | Free                     | See <a href="#">Loading data into BigQuery</a> .                                 |
| Copying data        | Free                     | See <a href="#">Copying an existing table</a> .                                  |
| Exporting data      | Free                     | See <a href="#">Exporting data from BigQuery</a> .                               |
| Metadata operations | Free                     | List, get, patch, update and delete calls.                                       |

1.34TB는 약 \$6.7

### 비용 요소

- GCS 비용 + BigQuery Storage 비용  
+ Query 비용
- BigQuery는 10MB 단위로 청구  
(월 1TB 무료)

### 고려 사항

- 사용자가 range query를 제대로 수행  
하지 않으면 많은 비용이 발생할 수도  
있다.
- 대안으로 테이블을 사용자 용도에 맞  
게 복제해주거나 구글의 cost control  
을 통한 Quota 제한 고민

# 기간별 table을 쉽게 생성하는 방법?

# BigQuery가 제공하는 Decorator

## Table Decorator, Partition Decorator

테이블 생성 후 1시간 이전까지의 데이터 조회

```
SELECT COUNT(*) FROM [cashslide:dataset.table@-3600000]
```

2016년 8월 30일 데이터 조회

```
SELECT COUNT(*) FROM cashslide:dataset.table$20160830
```

### 장점

- 날짜별 table이 존재하듯이 사용
- 손쉬운 range 검색
- 기간별, 일자별 table 생성이 용이

### 고려 사항

- Table 생성 시 Partitioned Table로 생성되어야 함

# 편리한 Partitioned tables

나중에 이걸 확인한다면 BigQuery 적재 작업의 처음으로 돌아가시오.

Partitioned tables 생성 예제

```
$bq mk --time_partitioning_type=DAY \
--time_partitioning_expiration=259200 \
mydataset.table
```

생성된 테이블 정보 확인

```
$bq show --format=prettyjson mydataset.table
{
  ...
  "tableReference": {
    "datasetId": "mydataset",
    "projectId": "myproject",
    "tableId": "table2"
  },
  "timePartitioning": {
    "expirationMs": "2592000000",
    "type": "DAY"
  },
  "type": "TABLE"
}
```

고려 사항

- Dataset table 생성 시 Date-Partitioned tables로 생성 가능
- 생성 시 Expiration time은 second 기준
- \_PARTITIONTIME 이름의 pseudo column 을 포함하며, data 적재 시점을 기준으로 TIMESTAMP("2016-04-15") 형식의 값을 저장함. UTC 기준

# 유용한 Decorator 2가지 (1/2)

## Table Decorator

### Syntax

```
# Snapshot decorators  
@<time>  
  
# Range decorators  
@<time1>-<time2>
```

### 고려 사항

- 최대 7일 이전의 데이터까지 조회 가능 (rollback)
- Milliseconds 단위

### Example

```
# Snapshot example  
SELECT COUNT(*) FROM [cashslide:dataset.table@-3600000]  
  
# Range example  
SELECT  
    COUNT(*)  
FROM  
    [cashslide:dataset.table@-3600000--1800000]
```

# 유용한 Decorator 2가지 (2/2)

## Partition Decorator

### Syntax

```
[TABLE_NAME]$YYYYMMDD
```

### Example

```
# Snapshot example
SELECT COUNT(*) FROM cashslide:dataset.table$20160505
```

```
# Range example
SELECT
  field
FROM
  table
WHERE
  _PARTITIONTIME BETWEEN TIMESTAMP('2016-01-01')
  AND TIMESTAMP('2016-01-21');
```

### 고려 사항

- \$YYYYMMDD decorator를 이용한 snapshot 검색
- Pseudo column을 이용한 range 검색

# 이렇게 구축한 BigQuery를 잘 활용하는데 도움되는 TIP 몇가지

# BigQuery Tip 1

BigQuery Composer 여려모로 편리합니다.

The screenshot shows the BigQuery Composer interface. In the top-left, there's a 'New Query' section with a code editor containing the following SQL:

```
1 select timestamp_millis(session_created_at + time_offset),
2       timestamp_millis(time*1000 + time_offset), user.nickname, type, name
3   from lumberjack.log
4  where user.nickname = 'yarrie'
5    AND _PARTITIONTIME BETWEEN TIMESTAMP('2016-09-01')
6    AND TIMESTAMP('2016-09-10')
7   limit 5;
```

The bottom part of the interface shows the 'Results' tab, which displays the execution details of the query. It includes a 'Stage timing' section with four stages (Stage 1, Stage 2, Stage 3) and a 'Rows' section showing input and output counts. The 'Explanation' tab is also visible.

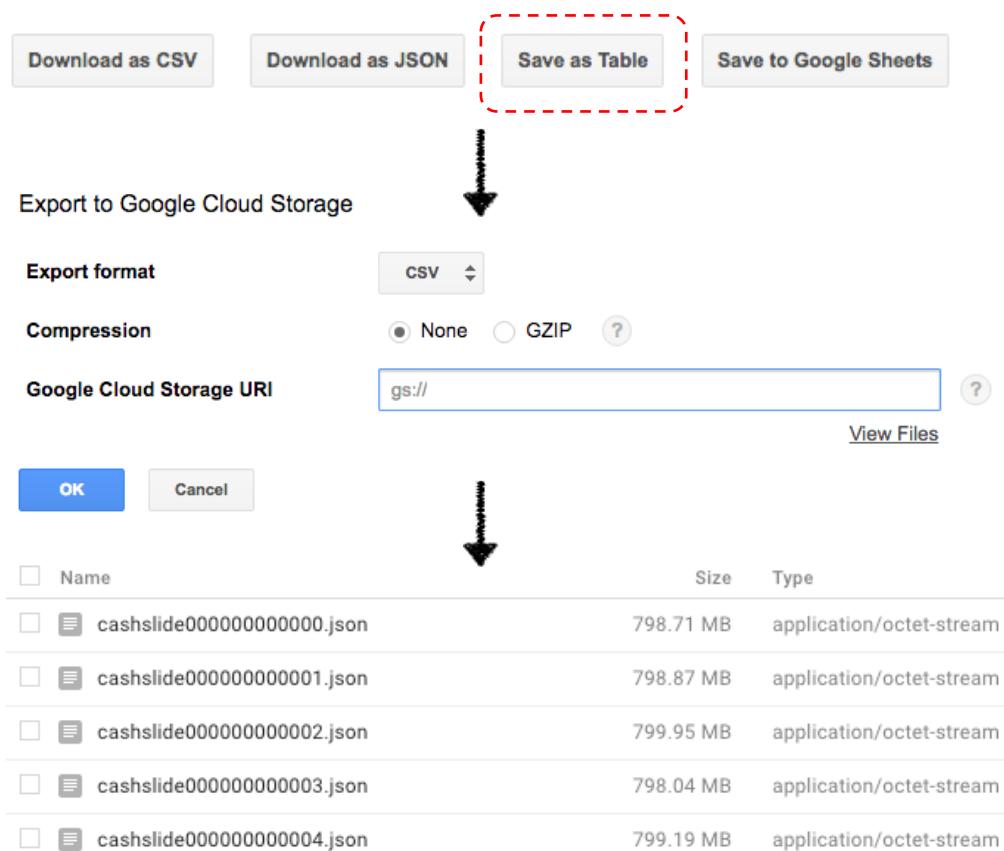
| Stage   | Wait       | Read | Compute | Write | Input | Output |
|---------|------------|------|---------|-------|-------|--------|
| Stage 1 | 14,203,136 | 5    |         |       |       |        |
| Stage 2 | 6          | 5    |         |       |       |        |
| Stage 3 | 5          | 5    |         |       |       |        |

## TIP

- Query box 안의 Color를 보면 현재 syntax, dataset name, table name, column name, function name 의 오류를 쉽게 확인 가능
- Results Tab 옆의 Explanation 을 보면 Query 수행 과정이 알기 쉽게 표기되어 Query 성능 개선 가능

# BigQuery Tip 2

File export 는 file당 1G, 일일 최대 10TB 가능



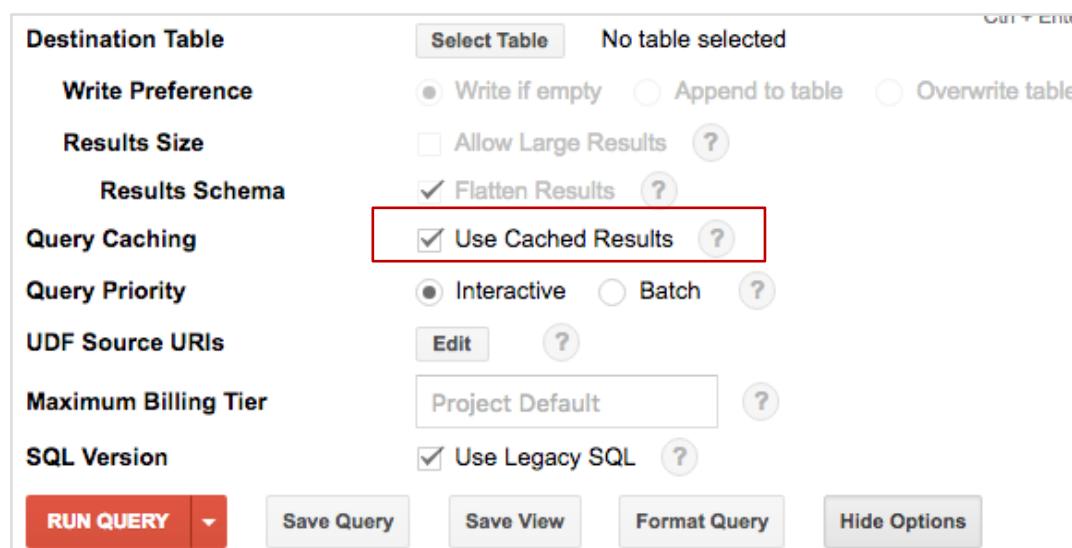
## TIP

- 쿼리 결과의 export는 파일당 최대 1G 가능
- 그 이상의 결과를 export 하려는 경우 쿼리 결과에서 'Save as Table'로 저장 후 export 항목에 '\*.json' 형태로 기입

# BigQuery Tip 3

## BigQuery 비용 절감을 위한 Use Cache 사용

Query Composer에서 'Show Options'를 눌러보면



### TIP

- 'Use Cached Results'를 사용하여 캐싱된 이전 쿼리 결과를 사용하면 중복되는 쿼리들의 비용을 절감 가능

# BigQuery Tip 4

Zeppelin 0.6.1부터 '%bigquery' 인터프리터 지원

The screenshot shows the Apache Zeppelin 0.6.1 interface. At the top, there is a navigation bar with the Zeppelin logo, version 0.6.1, and links for Quick Start, Interpreter, and Display System. Below the header, the main content area has a title "BigQuery Interpreter for Apache Zeppelin". Under this title is a list of topics: Overview, Configuration, BigQuery API, Enabling the BigQuery Interpreter (with a sub-item "Setup service account credentials"), Using the BigQuery Interpreter, and Technical description. In the bottom section, there is a code editor containing a sample SQL query:

```
%bigquery.sql
SELECT departure_airport, count(case when departure_delay>0 then 1 else 0 end) as no_of_delays
FROM [bigquery-samples:airline_onime_data.flights]
group by departure_airport
order by 2 desc
limit 10
```

## TIP

- Zeppelin 0.6.1 버전부터 BigQuery Interpreter를 제공
- Data뿐만 아니라 BigQuery Computing resource를 활용하여 작업 수행 가능 (AWS EMR, Google Dataproc의 비용 불필요)
- 기존 Notebook의 이전 작업이 필요

# BigQuery Tip 5

손쉽게 연계 사용이 가능한 Data Studio가 무료

The image shows two screenshots side-by-side. On the left is the Google Cloud BigQuery interface, displaying a list of datasets and tables. On the right is a Data Studio report titled 'YouTube Sample Channel Report' for the period from January 18, 2017, to February 14, 2017.

**BigQuery Interface:**

- Left sidebar: 커넥터 (Add-ons, Attribution 360, BigQuery, Cloud SQL, DCM, Google Analytics, Google Sheets, MySQL, PostgreSQL, Search Console, YouTube Analytics).
- Main area: BigQuery project list. Projects shown: cashslide, bigdata, com\_cashslide\_ANDROID, log.

**Data Studio Report:**

### YouTube Sample Channel Report (2017. 1. 18. - 2017. 2. 14.)

**Trending by Views, Watch Time, & Shares:**

- Views: 1.1만 (1.1 million)
- Watch Time: 03:52
- Video Shares: 1.3천 (1.3 thousand)

**Top Videos Watched:**

| External Video ID | View Count | Watch Time | Video ID |
|-------------------|------------|------------|----------|
| uZ548Xiphc        | 373        | 00:03:59   |          |
| uZ548AIShpc       | 230        | 00:01:09   |          |
| uW548XlShpc       | 125        | 00:03:39   |          |
| uZA48XlSkpc       | 62         | 00:03:35   |          |
| u9548XlWhpc       | 81         | 00:04:05   |          |
| uZ548SlShpc       | 108        | 00:06:09   |          |
| uZA48XlShpc       | 99         | 00:03:31   |          |
| uW548XlShpe       | 102        | 00:05:17   |          |
| u9548XlShpc       | 76         | 00:02:35   |          |
| uZ548XlAhpc       | 38         | 00:05:58   |          |

**Metrics:**

- Likes Added & Removed: + 602.0
- Subscriptions Added & Removed: + 36.0

결론 :  
실시간 트렌드 변화는 Kibana  
통계 분석에는 Zeppelin  
이 두 가지가 모두 가능한 Data Studio

하지만  
도구는 거들뿐..

# Q&A

# THANK YOU

NBT

주소 : 서울시 서초구 서초동 1341-7 조이빌딩