Druid로 쉽고 빠르게 데이터 분석하기

DEVIEW 2018

송은혜

NAVER

NAVER

CONTENS

DEVIEW 2018

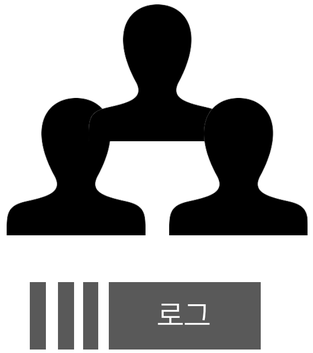
1. 데이터 분석의 필요성
2. Druid? 그게 뭔데?
   - Apache Druid vs. Elasticsearch vs. Apache Kudu
3. Druid로 분석하기 – 기본편
4. Druid로 성능 향상시키기 – 고급편
   - 질의성능 향상시키기
   - 저장성능 향상시키기
   - 편한 운영을 위한 노력

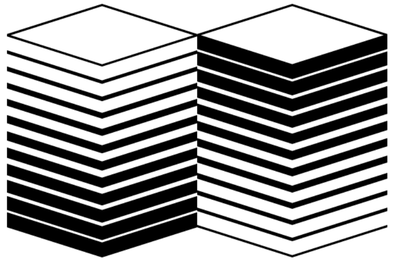
DEVIEW
2018

1.
데이터 분석의 필요성

1. 데이터 분석의 필요성

DEVIEW 2018



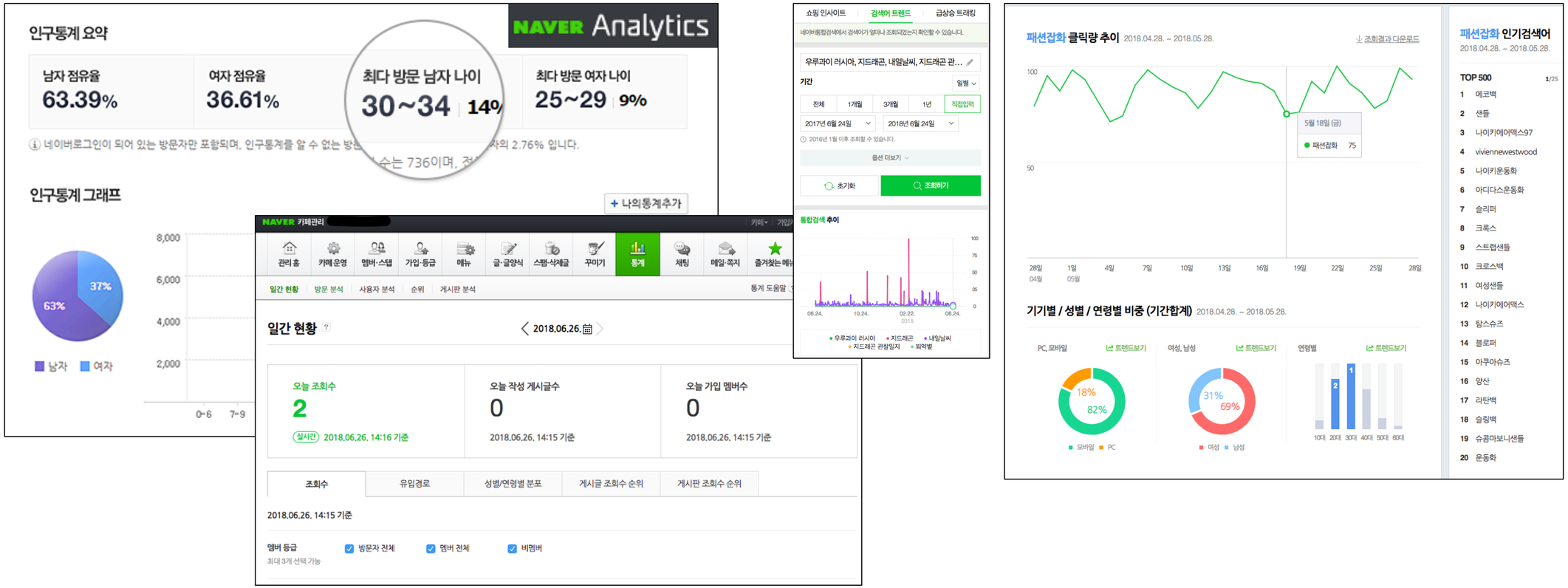


Created by Magicon
from Noun Project

현재에 대한 평가
서비스 품질향상
Data-Driven …

1. 데이터 분석의 필요성

DEVIEW 2018



DEVIEW
2018

2.
Druid? 그게 뭔데?

2.1 druid , 그게 뭔데?

DEVIEW 2018

- 컬럼기반 분산저장소
- Time 기반 partitioning
- 실시간/배치 데이터 저장기능 제공
- 데이터 질의기능 제공

Next Session.
"스트림 저장소 최적화 이야기"

2.2 Druid vs. Elasticsearch vs. Kudu

DEVIEW 2018

검색어트렌드/쇼핑인사이트

요구사항

- 임의의 조건 조합 분석 (원본 < 조합)
- 데이터 유효기간기간 없음
  (데이터 감소X)
- 내부 분석환경
- 개발/운영비용 최소화 (관리인원 5명)

2.2 Druid vs. Elasticsearch vs. Kudu

DEVIEW 2018

후보 선정이유 – 데이터 Size, 운영비용

Apache Druid (v0.10.0)	Elasticsearch (v2.3)	Apache Kudu (v1.4)
• 저장/질의기능 제공 • 빠른 필터처리	• 기존 사용 플랫폼 • 저장/질의기능 제공	• 가장 적은 저장공간 • 빠른 조회(scan)기능 • 질의기능 미제공 (질의엔진 필요)

2.2 Druid vs. Elasticsearch vs. Kudu

DEVIEW 2018

응답시간 비교실험 – 질의 패턴

(2.1억건, 동일 병렬환경)

Random Access

```
SELECT count(*)
  FROM logs
WHERE dimA = ?
```

Ranking

```
SELECT url,
       count(*)
  FROM logs
GROUP BY url
ORDER BY
       count(*) desc
  LIMIT 10
```

Group By

```
SELECT user,
       count(*)
  FROM logs
GROUP BY user
```

Join

```
SELECT …
  FROM logs
INNER JOIN users
    ON …
```

2.2 Druid vs. Elasticsearch vs. Kudu

DEVIEW 2018

응답시간 비교실험

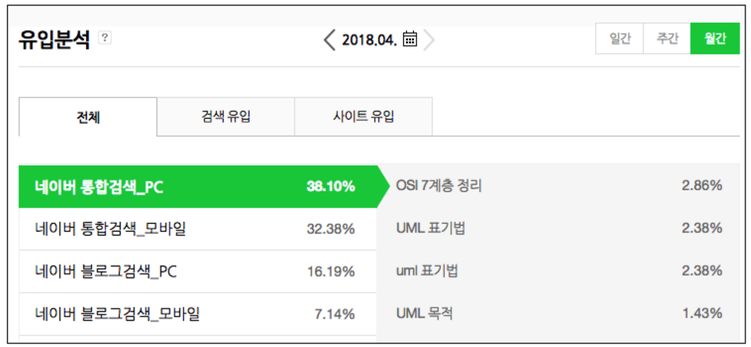
(2.1억건, 동일 병렬환경)

	Random Access	Ranking	Group By	Join
Apache Druid (v0.10.0)	★★★★☆	★★★★★	★★★★☆	★★★★☆
Elasticsearch (v2.3)	★★★★★	★★★☆☆	☆☆☆☆☆	☆☆☆☆☆
Apache Kudu (v1.4) with Impala	★★★★★ (pk) ★☆☆☆☆(pk외)	☆☆☆☆☆	★★★★★	★★★★★

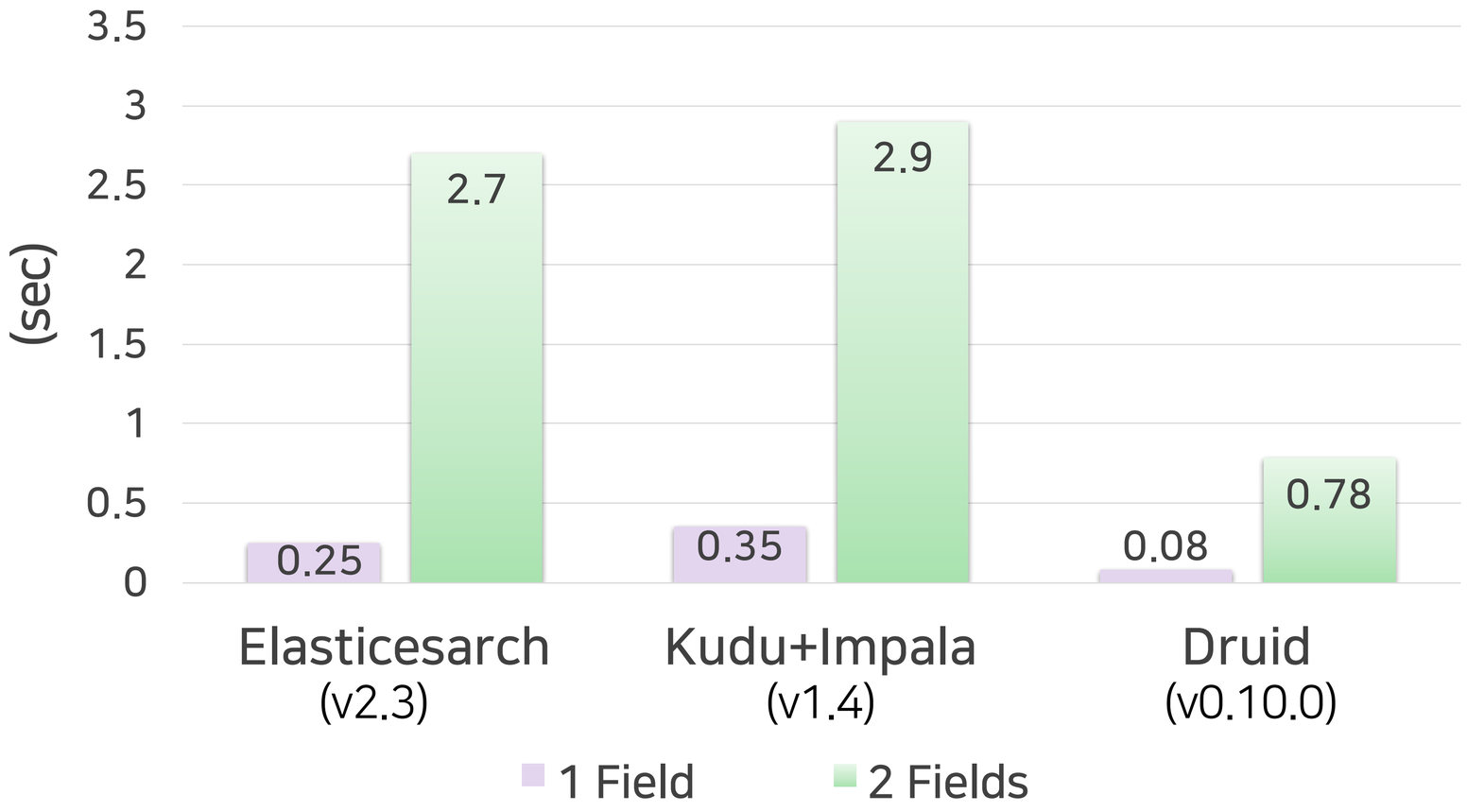
2.2 Druid vs. Elasticsearch vs. Kudu

DEVIEW 2018

응답시간 비교실험 – 다차원 분석



(2.1억건, 동일 병렬환경)



2.2 Druid vs. Elasticsearch vs. Kudu

DEVIEW 2018

	Random Access	Ranking	Group By	Join
Apache Druid	★★★★☆	★★★★★	★★★★☆	★★★★☆

- Secondary Index를 지원하여 PK 외의 key로 조회시에도 빠르게 응답
- Bitmap Index를 사용하여 빠른 필터처리 가능
- 다양한 근사알고리즘 질의기능 제공
- 복잡한 질의처리 가능 (Spark Druid Connector)

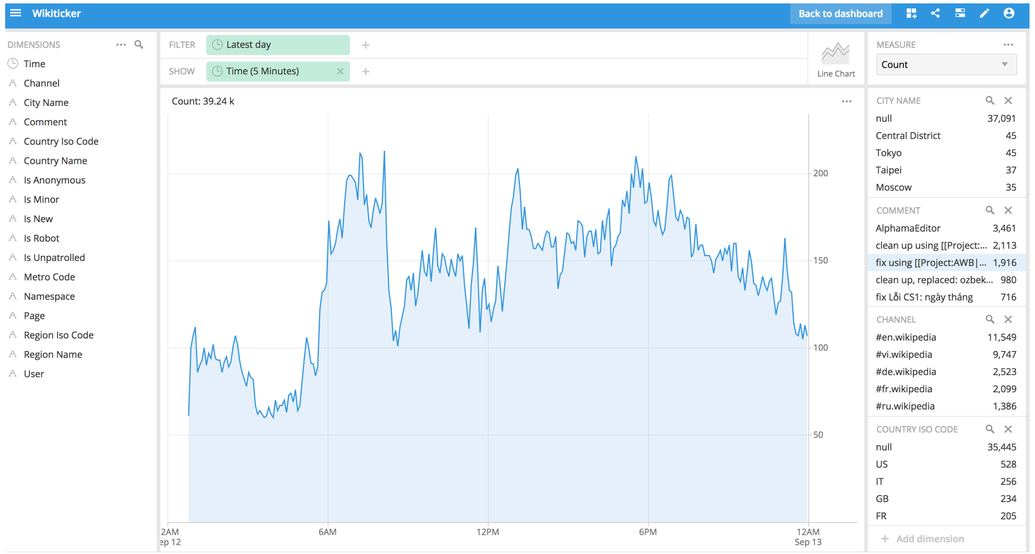
3.
Druid로 분석하기 - 기본편

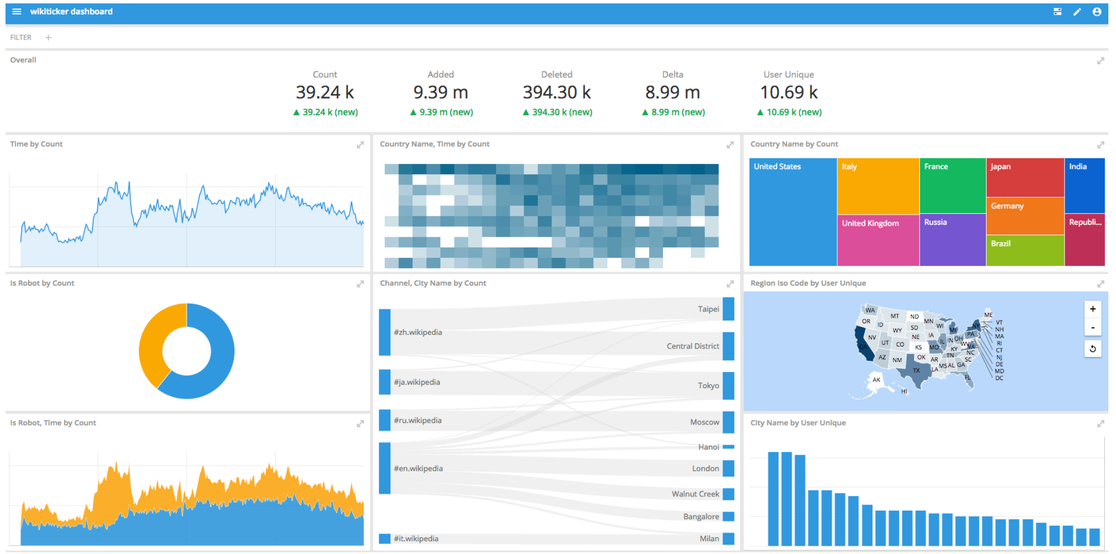
DEVIEW
2018

3.1 Imply-UI

DEVIEW 2018

https://docs.imply.io/on-prem/visualize/





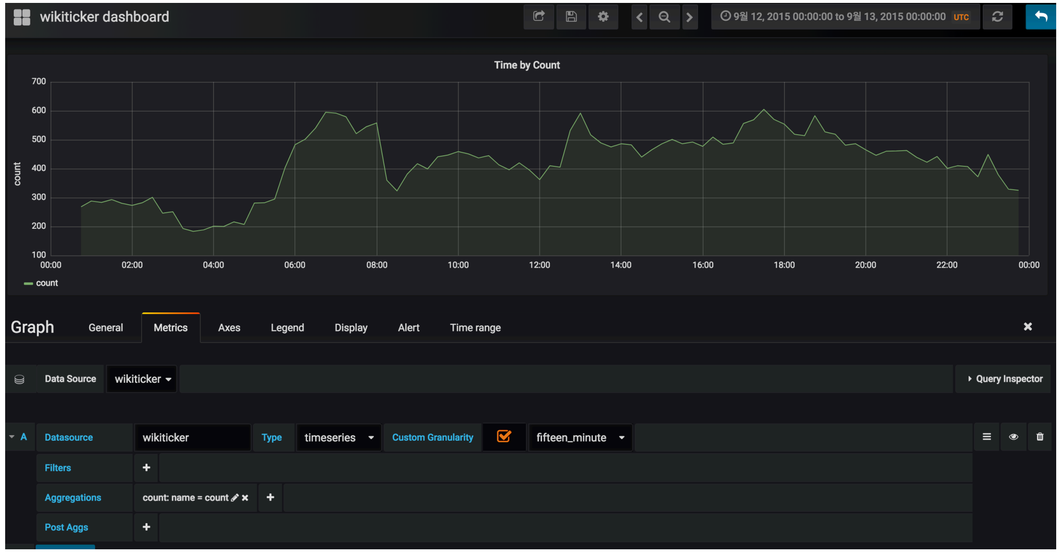
- Imply에서 개발한 Visualize Tool
- Druid 버전에 따라 빠르게 업그레이드

- Drag & Drop으로 쉽게 질의결과 표현 가능
- 값의 분포, 이전기간과의 비교를 쉽게 해볼 수 있음

3.2 Grafana

DEVIEW 2018

https://grafana.com/



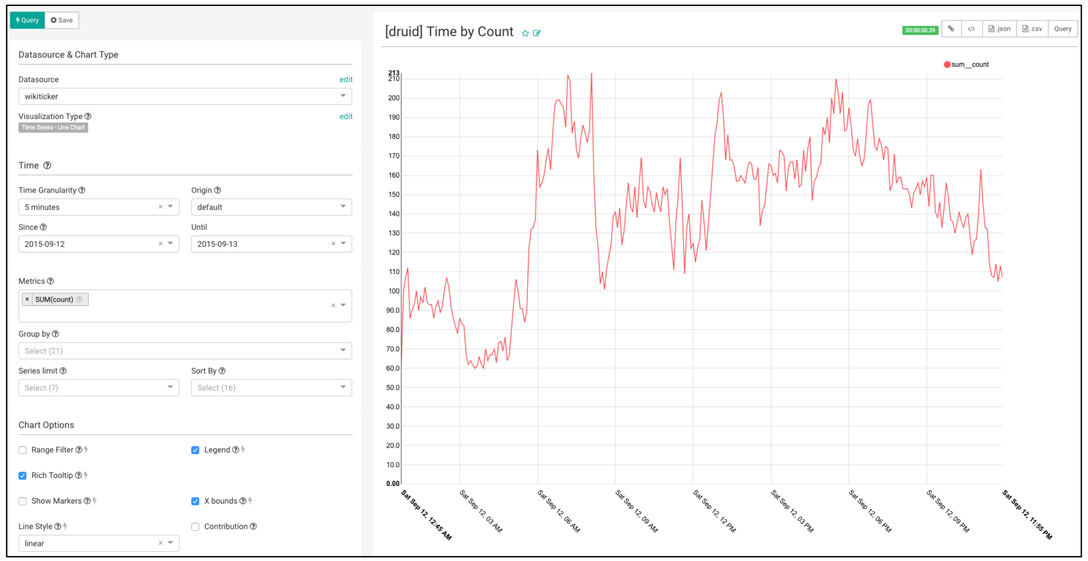
- 다양한 Datasource Visualize Tool
- Druid Plugin을 설치하여 Druid 연동가능

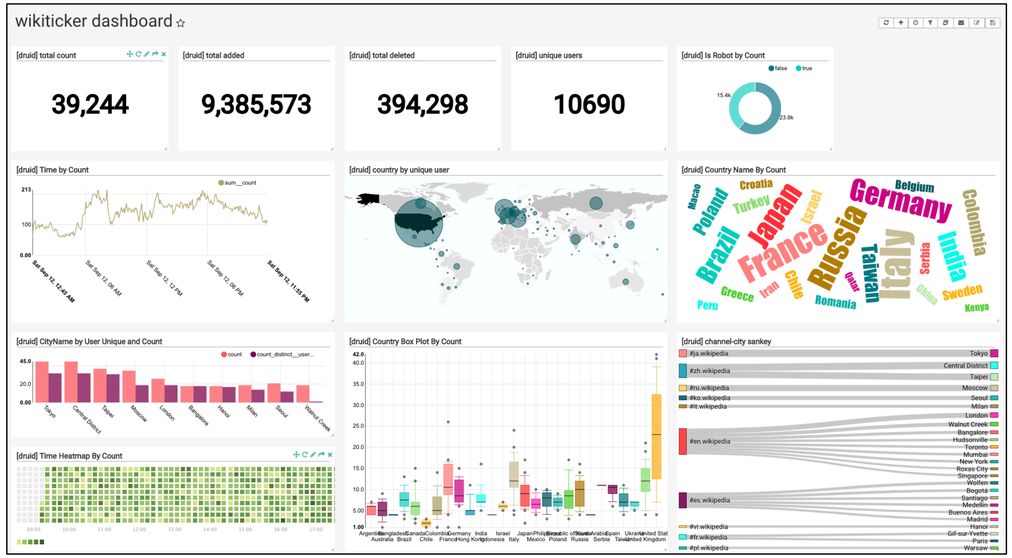
- Plugin을 통해 다양한 그래프 쉽게 표현 가능
- Druid의 DSL 질의방식에 대한 이해가 필요

3.3 Superset

DEVIEW 2018

https://superset.incubator.apache.org/druid.html





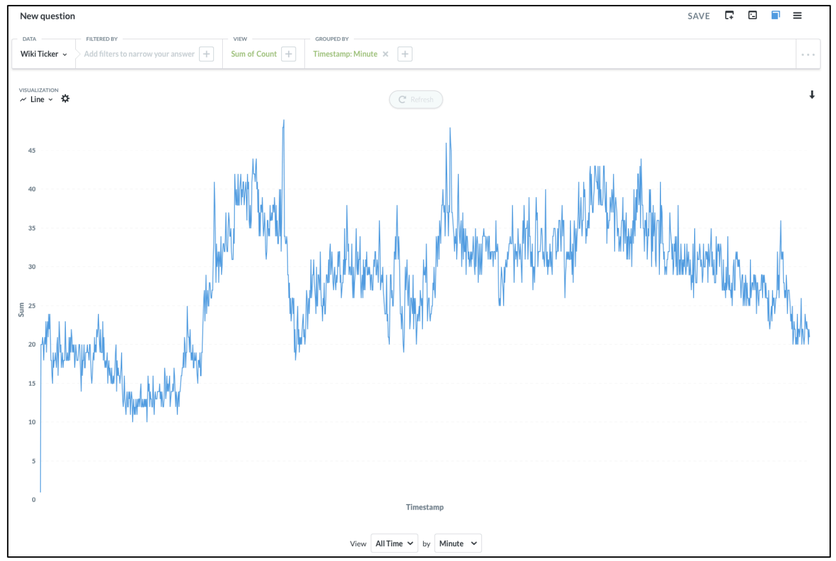
- 기본 제공 그래프 종류가 가장 많음
- 세세한 권한관리 가능

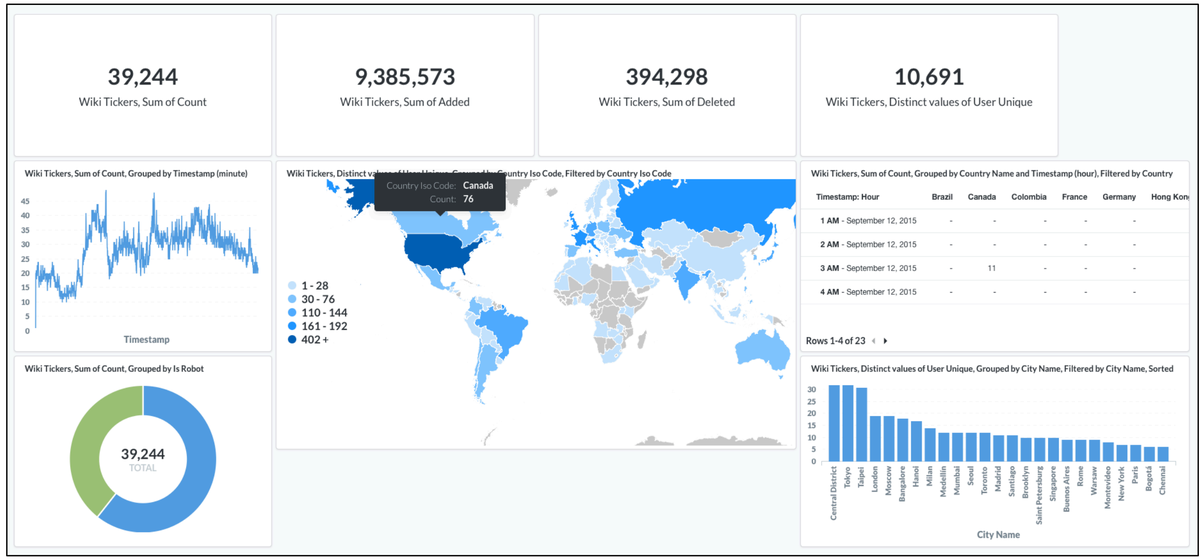
- 표현 자유도가 높음
- 설정할 수 있는게 많아 복잡/어렵게 느껴짐

3.4 Metabase

DEVIEW 2018

https://www.metabase.com/



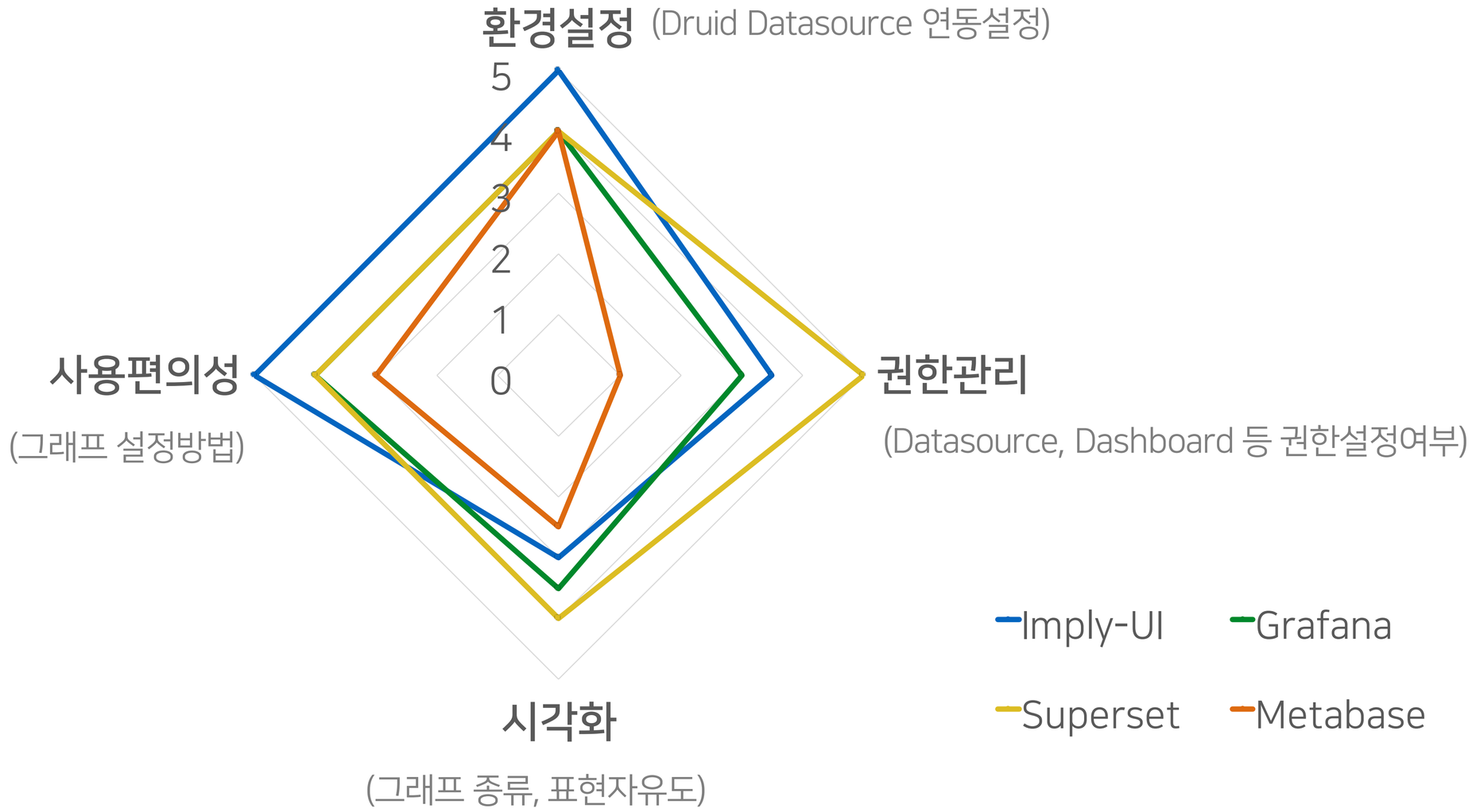


- Application 방식
- 깔끔한 UI

- 그래프의 세부적인 설정 불가

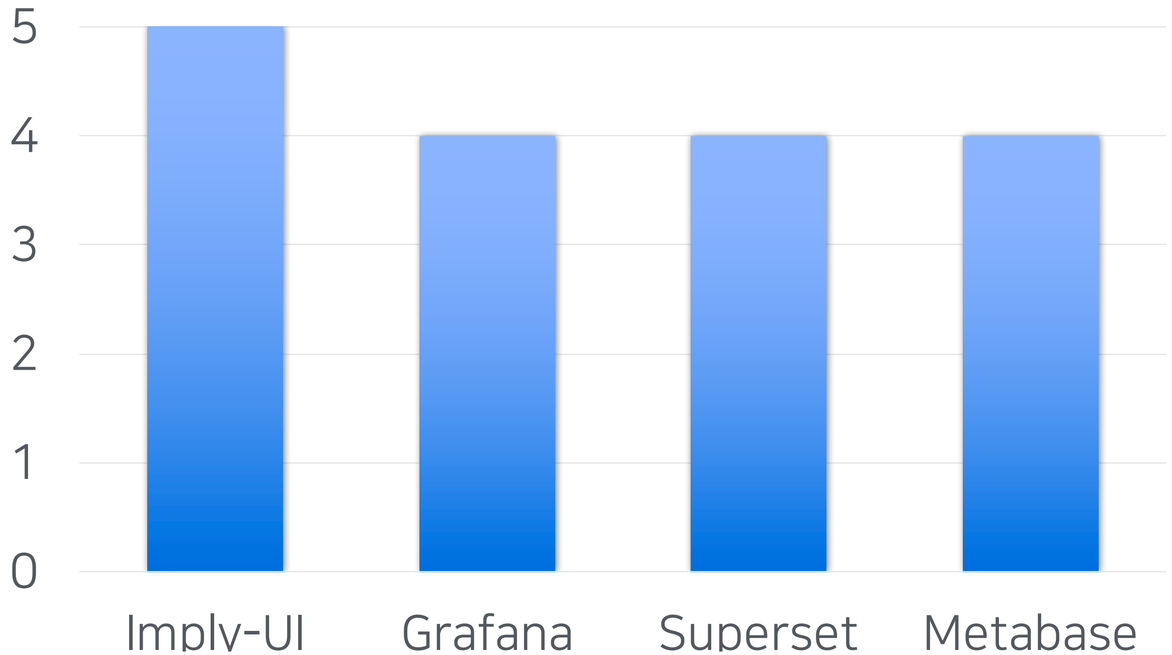
3.5 개인적인 총평

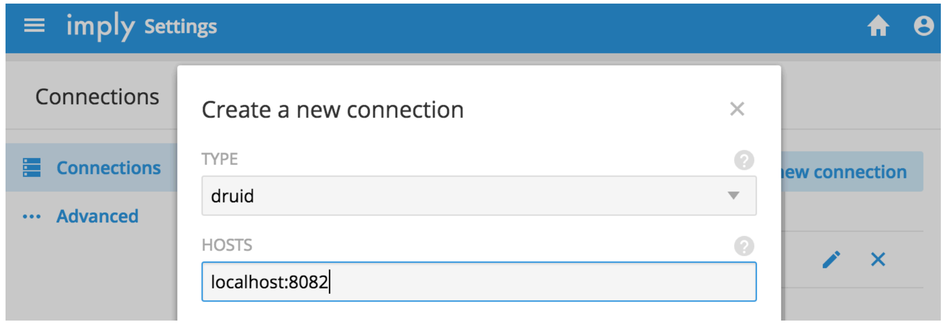
DEVIEW 2018



3.5 개인적인 총평

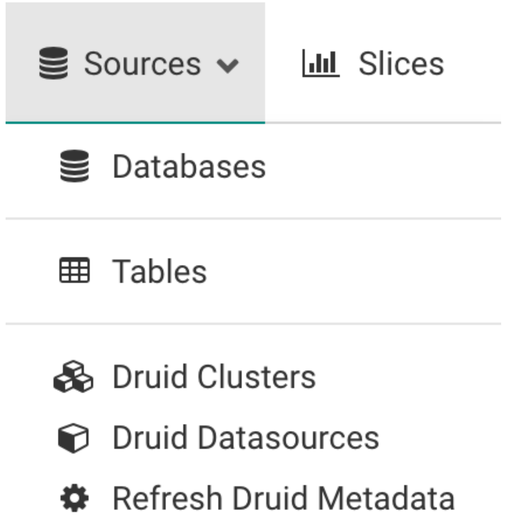
DEVIEW 2018





Imply UI

Superset

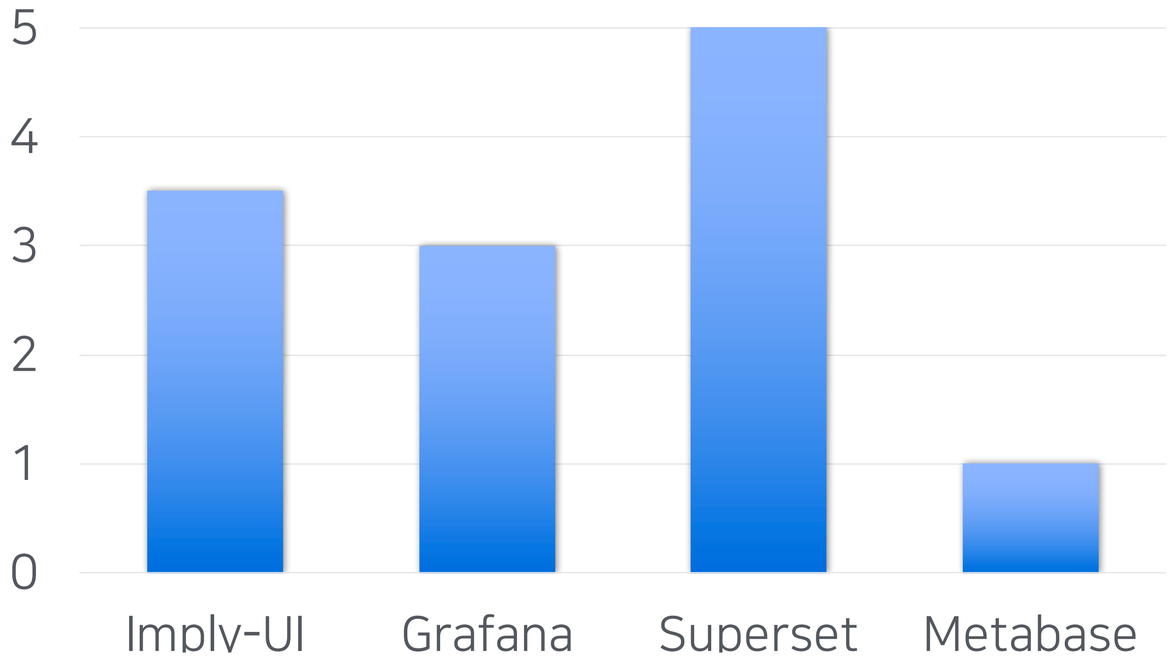


3.5 개인적인 총평

DEVIEW 2018

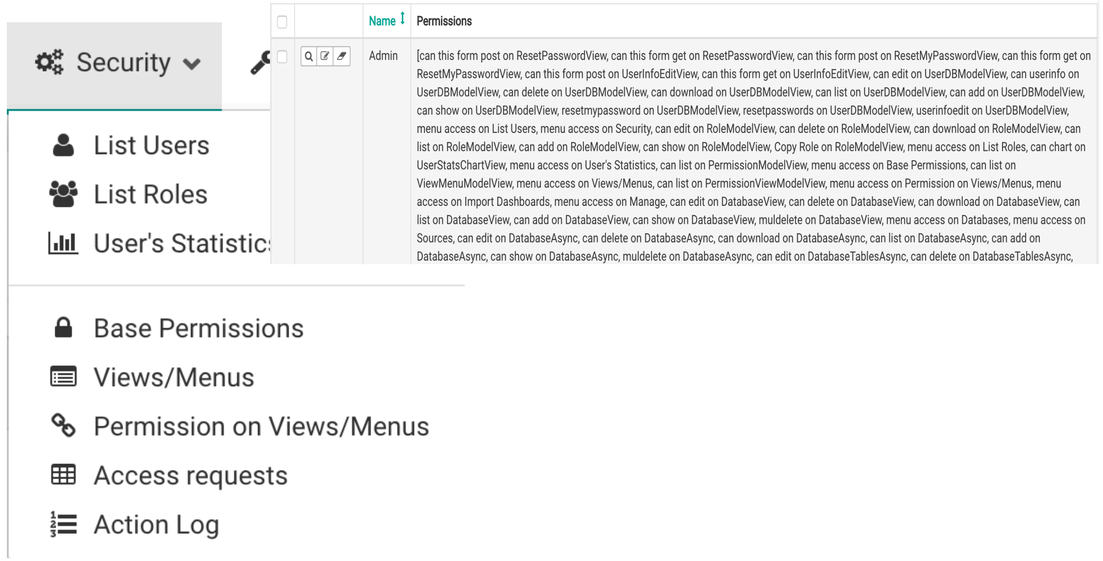
권한관리

(Datasource, Dashboard 권한설정)



Metabase

Superset

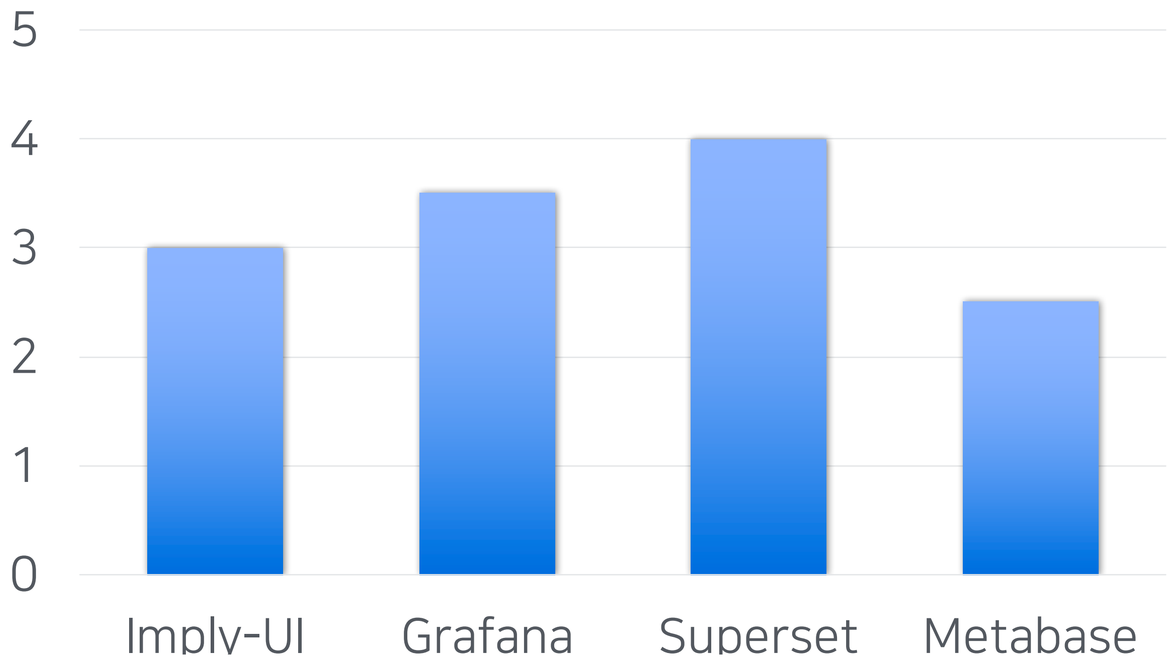


3.5 개인적인 총평

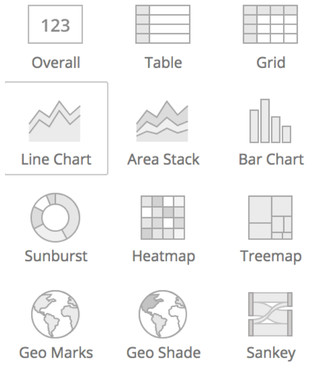
DEVIEW 2018

시각화

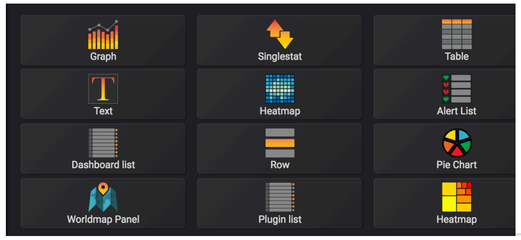
(표현가능 그래프, 표현 자유도)



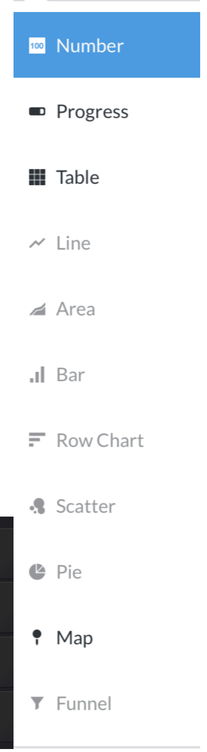
Imply UI



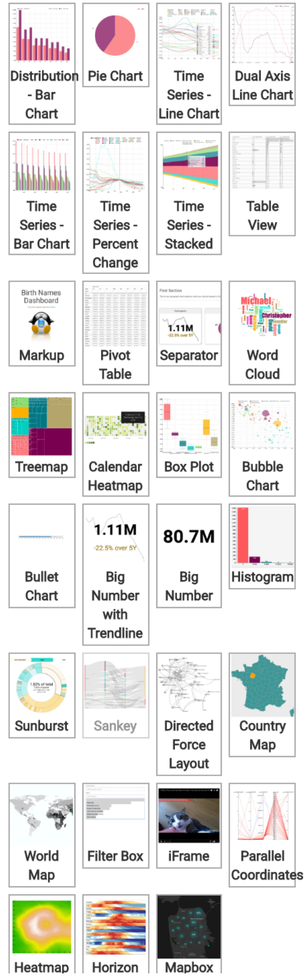
Grafana



Metabase



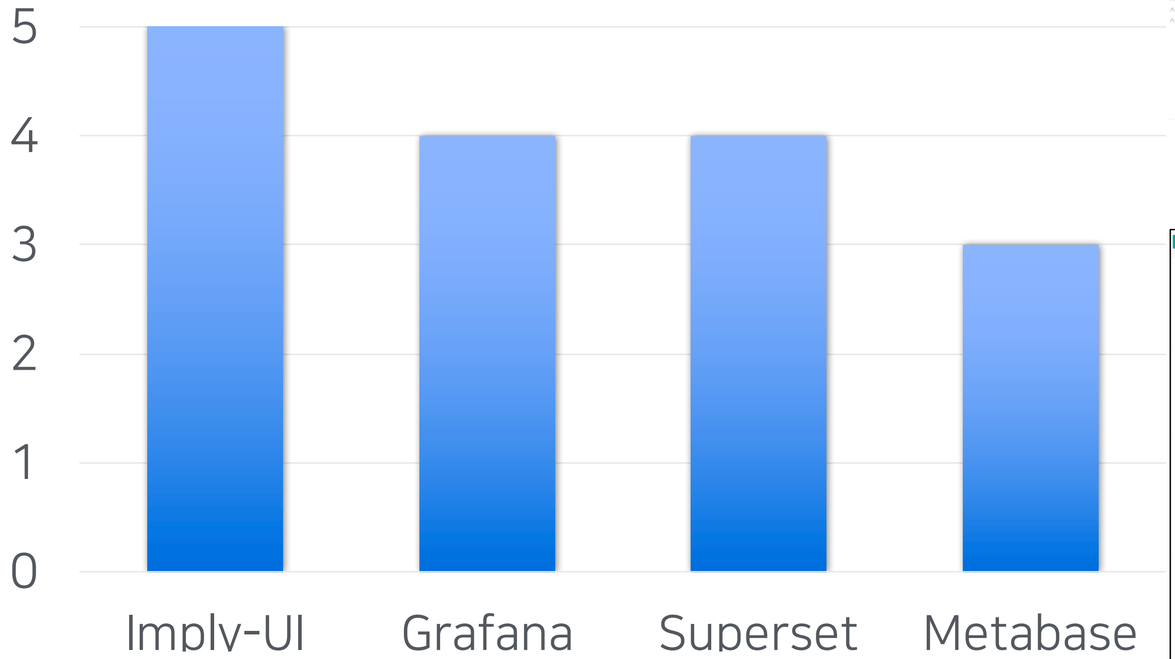
Superset



3.5 개인적인 총평

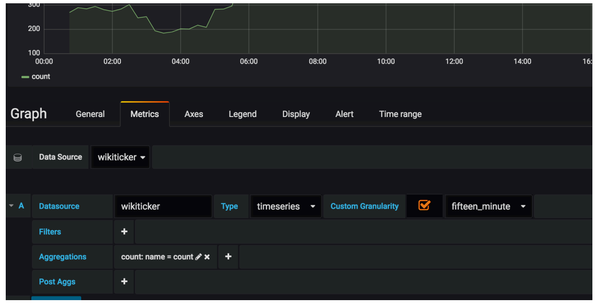
DEVIEW 2018

사용편의성
(그래프설정방법)

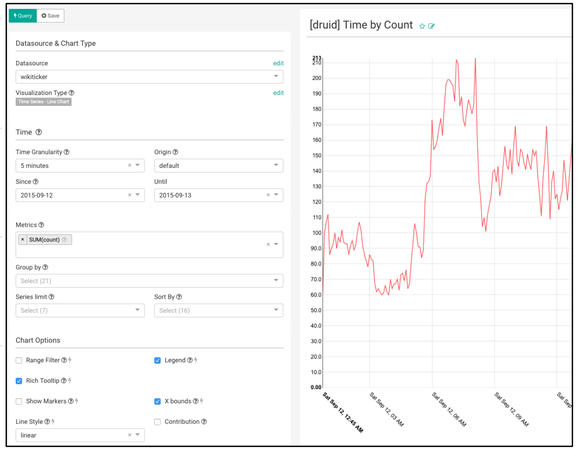


Imply UI

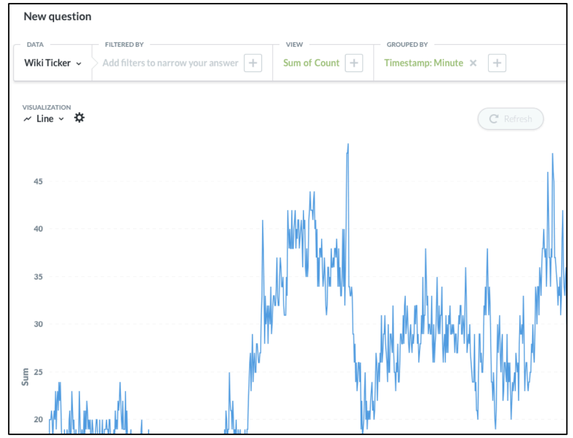
Grafana



Superset

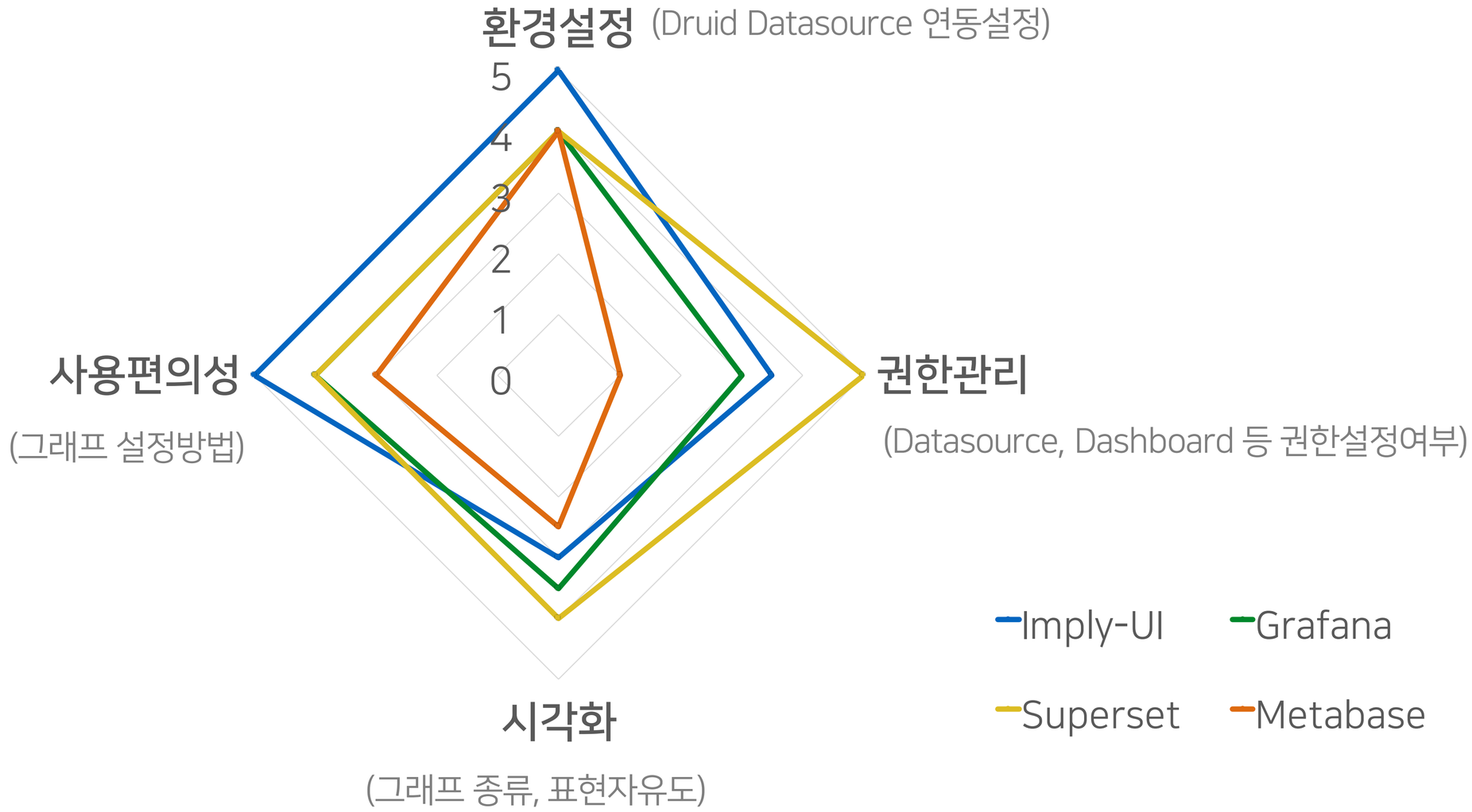


Metabase



3.5 개인적인 총평

DEVIEW 2018

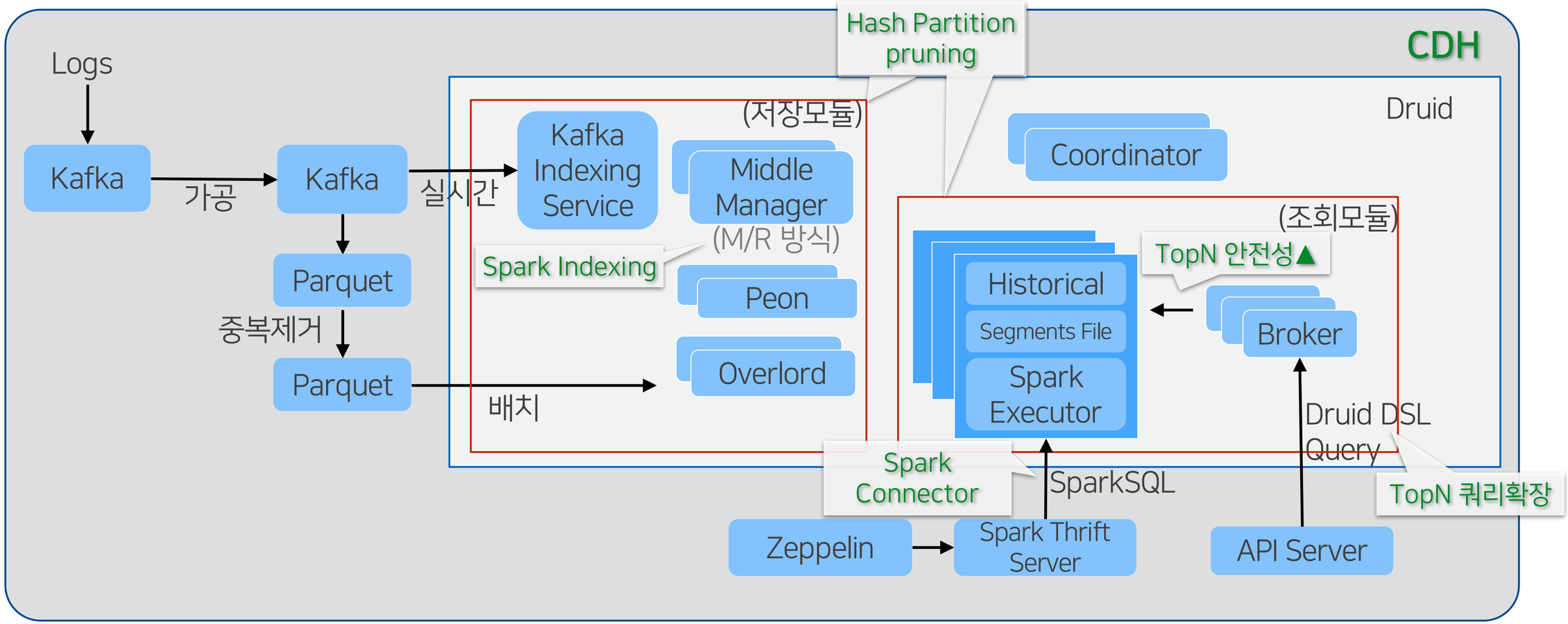


4.
Druid로 분석하기 - 고급편

DEVIEW
2018

4.1 Service Architecture

DEVIEW 2018



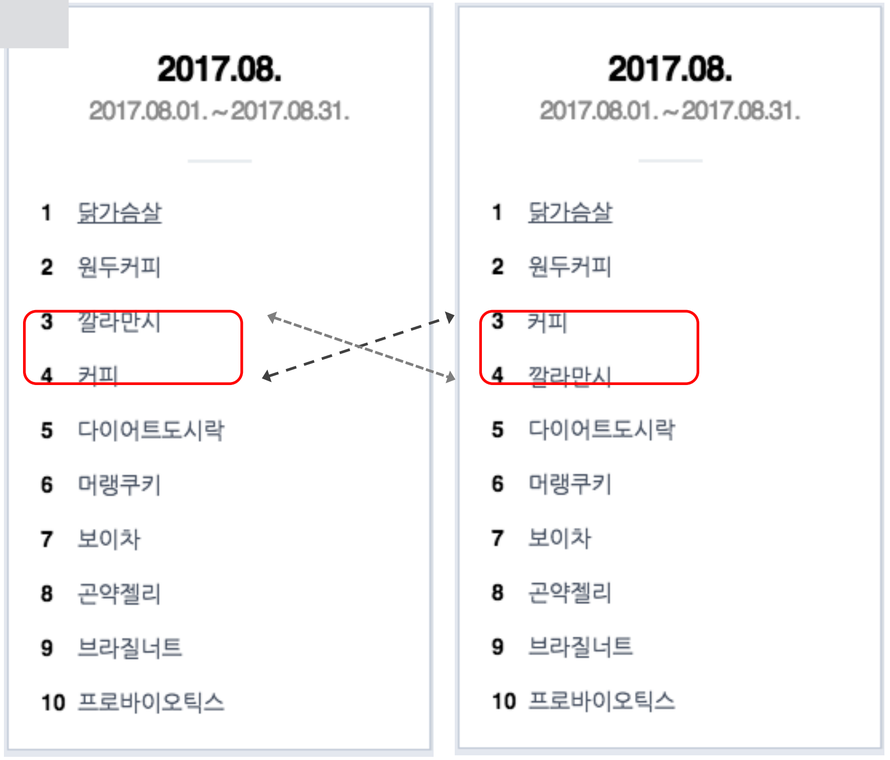
4.2 질의성능 향상시키기

DEVIEW 2018

1) TopN 결과 안정성 향상

지난기간(2017년 8월) 월간 데이터

순위가 바뀌는 경우가 있습니다.



데이터 변경은
없었는데…? ? ??

4.2 질의성능 향상시키기

DEVIEW 2018

$$(A + B) + C \; != \; A + (B + C)$$

country	SUM(duration)
korea	114
uk	47
usa	21

Seg. A Ranking

country	SUM(duration)
uk	67
usa	25
korea	17

Seg. B Ranking

country	SUM(duration)
china	33
korea	23
japan	27

Seg. C Ranking

country	SUM(duration)
korea	131
uk	114
usa	46

A + B

country	SUM(duration)
korea	154
uk	114
usa	46

(A + B) + C

≠

B + C

A + (B + C)

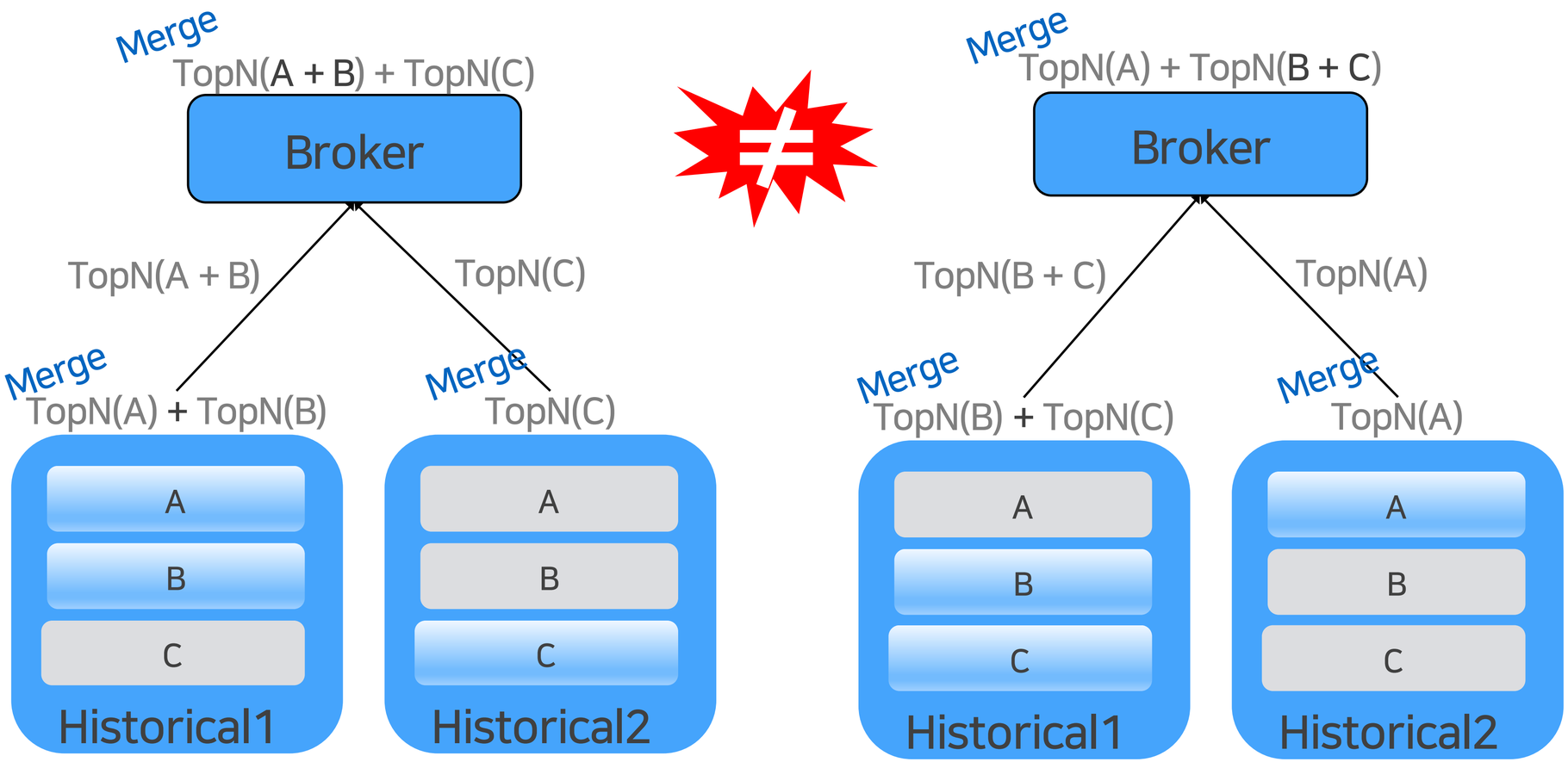
country	SUM(duration)
uk	67
korea	40
china	33
~~japan~~	~~27~~
~~usa~~	~~25~~

country	SUM(duration)
korea	154
uk	114
china	33

Final Ranking Result

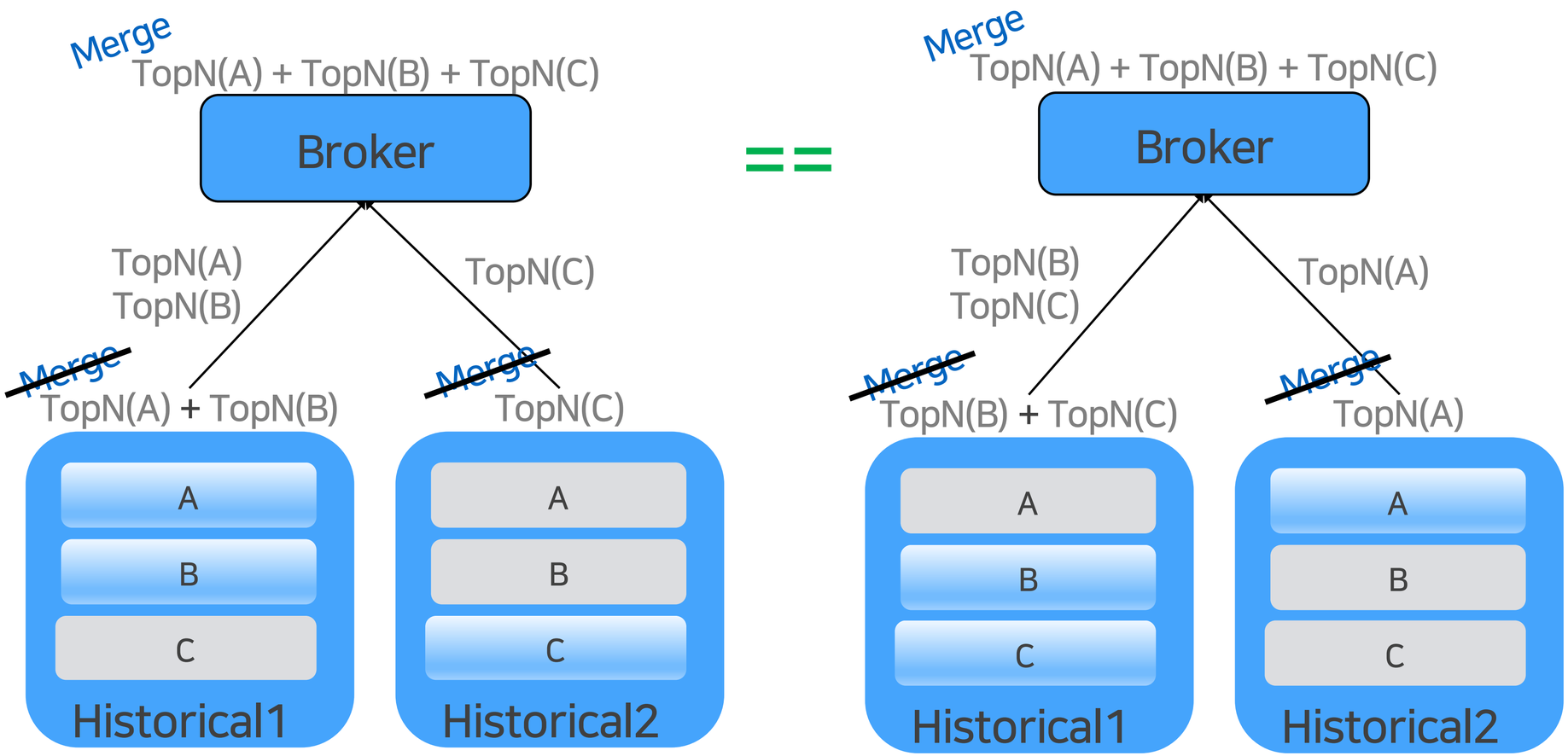
4.2 질의성능 향상시키기

DEVIEW 2018



4.2 질의성능 향상시키기

DEVIEW 2018

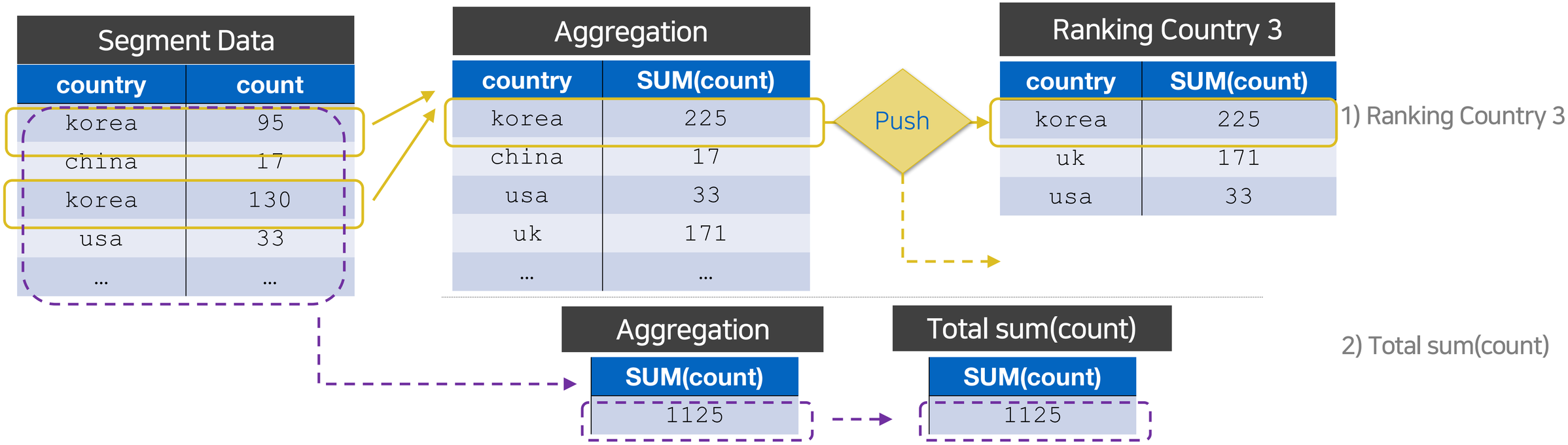


4.2 질의성능 향상시키기

DEVIEW 2018

Country	Sum(count)	Ratio over total count
korea	225	20%
uk	171	15.2%
usa	33	2.9%

처리과정이 비슷한데
한번에
처리할 수 없을까?



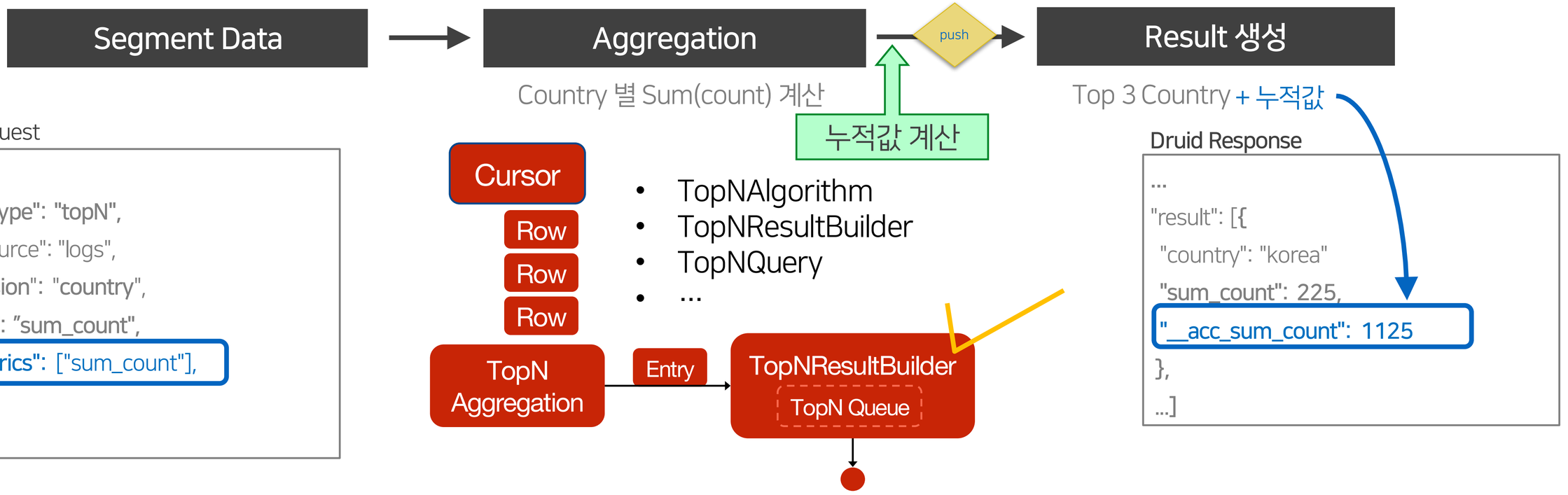
4.2 질의성능 향상시키기

DEVIEW 2018

2) TopN 쿼리 확장

Country	Sum(count)	Ratio over total count
korea	225	20%
uk	171	15.2%
usa	33	2.9%

1) Ranking Country 3
+ Total sum(count)



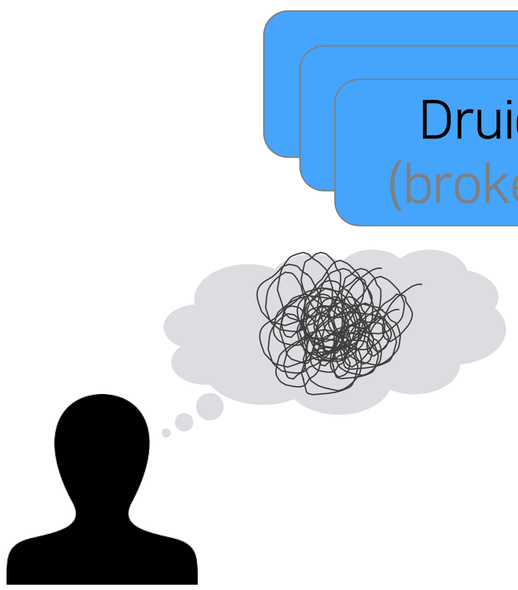
4.2 질의성능 향상시키기

DEVIEW 2018

3) Spark-Druid Connector

다양한 분석을 할 수 있게 해주세요

union, left join, inner join, groupby, having, ……




```
{
 "queryType": "topN",
 "dataSource": "logs",
 "dimension": "sample_dim",
 "granularity": "all",
 "threshold": 5,
 "aggregation": [...]
}
```

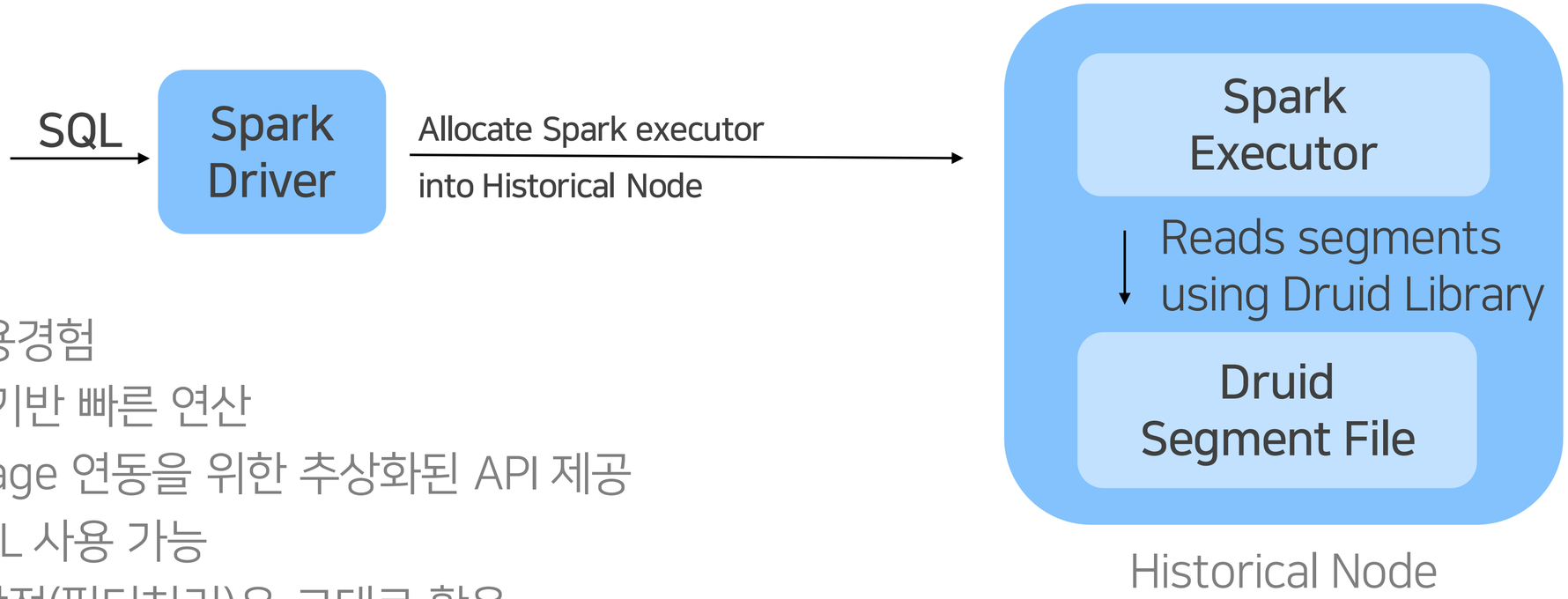
```
{"query": "select dim1 from logs limit 5"}
```

```
{
 "query":
  "select dim1 from logs limit 5
   union all
   select dim1 from ds2 limit 5"
}
```

4.2 질의성능 향상시키기

DEVIEW 2018

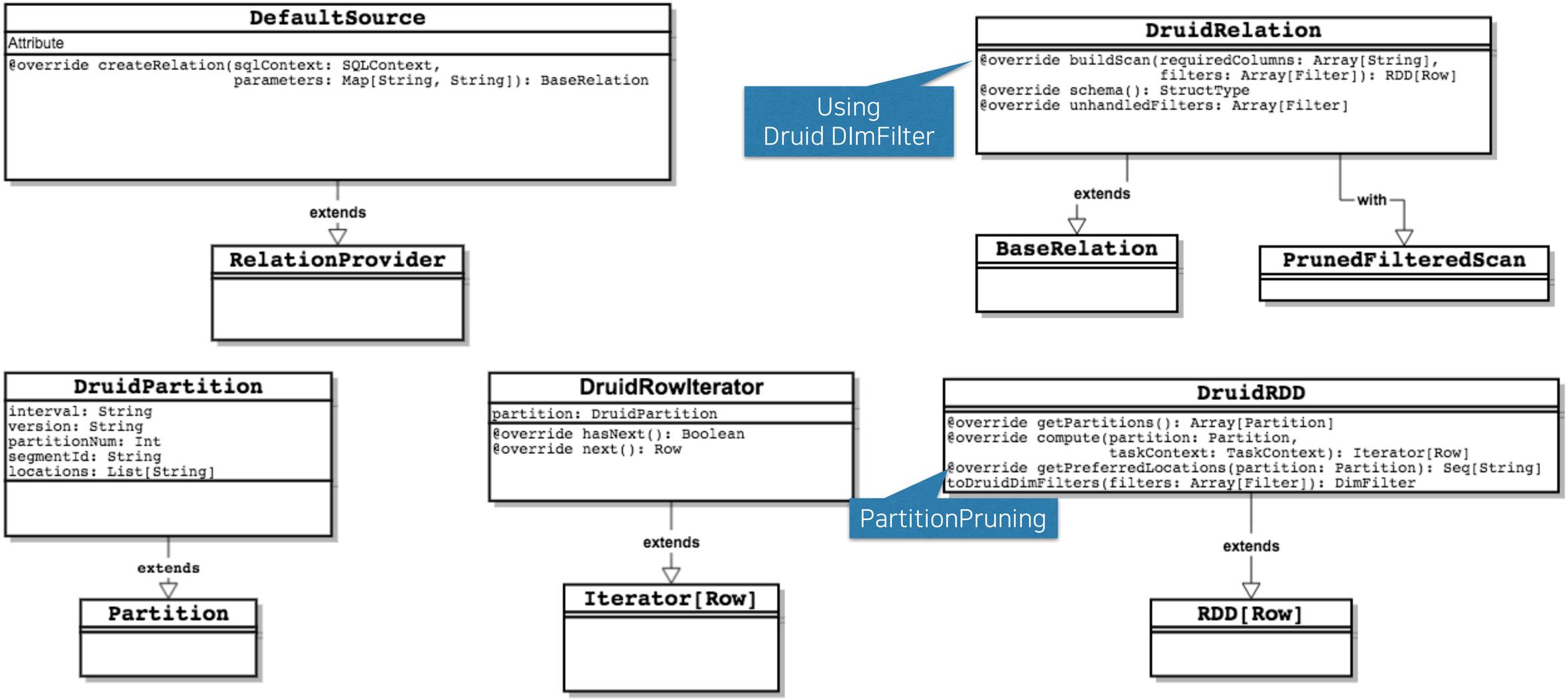
3) Spark-Druid Connector



Why Spark?

- Spark 사용경험
- Memory 기반 빠른 연산
- 외부 Storage 연동을 위한 추상화된 API 제공
- 친숙한 SQL 사용 가능
- Druid의 장점(필터처리)은 그대로 활용

4.2 질의성능 향상시키기



4.2 질의성능 향상시키기

3) Spark-Druid Connector

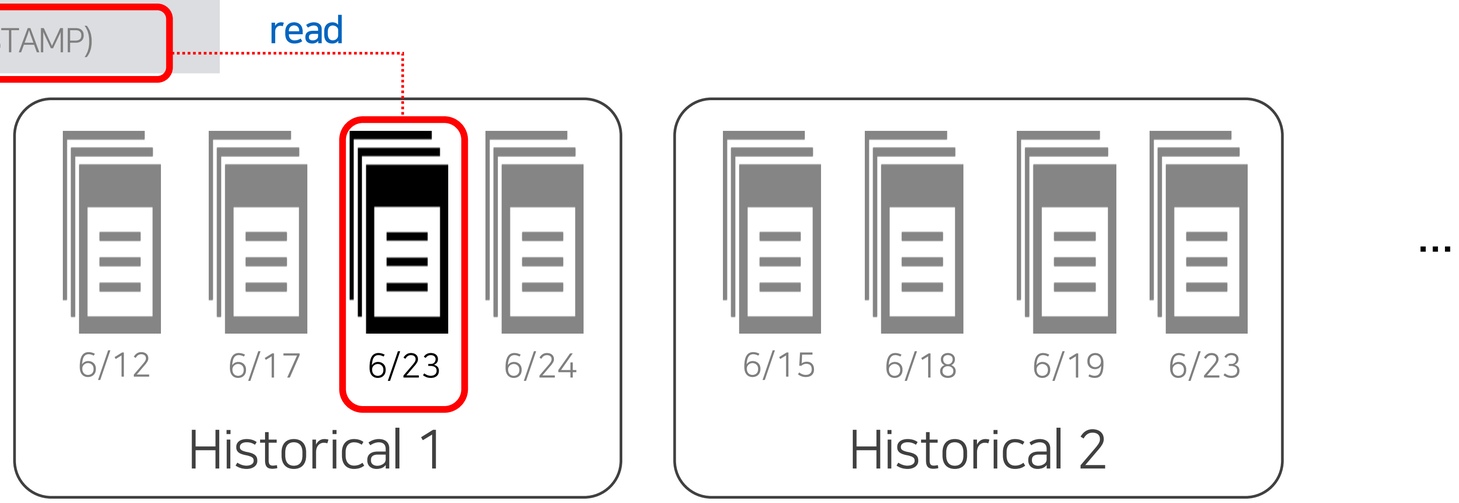
- Locality

  Historical node에 download & unzip된 segment file을 read (N/W 절약)

- Segment Partition pruning

  SQL의 조건을 분석하여 조회할 데이터가 있는 partition만 read (I/O, Memory 절약)

```
WHERE country = 'Korea'
AND time = CAST('2018-06-23'  AS TIMESTAMP)
```



Coordinator API

GET /.../intervals/2018-06-23/serverview

4.2 질의성능 향상시키기

DEVIEW 2018

3) Spark-Druid Connector

- Using Druid Filter

where statement (SQL Filter) -> Spark Filter (load data in memory and filter process)

-> Druid Filter (load filtered data by druid)

Country
Korea
UK
Korea
UK
China
China

Input Data

Korea → 101000

UK → 010100

China → 000011

Bitmap Index for Korea

→ UK : 2nd, 6th rows

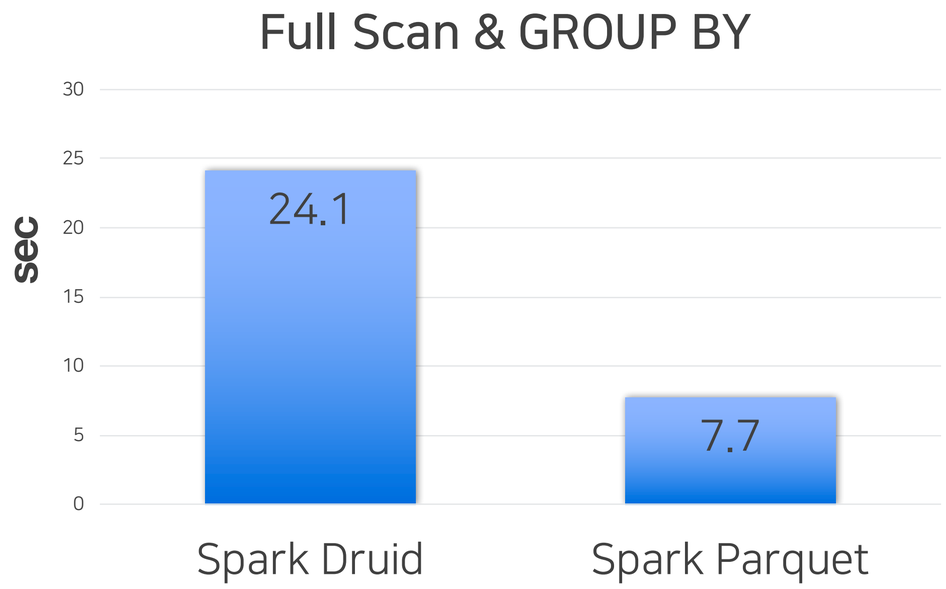
→ China : 7th , 8th rows

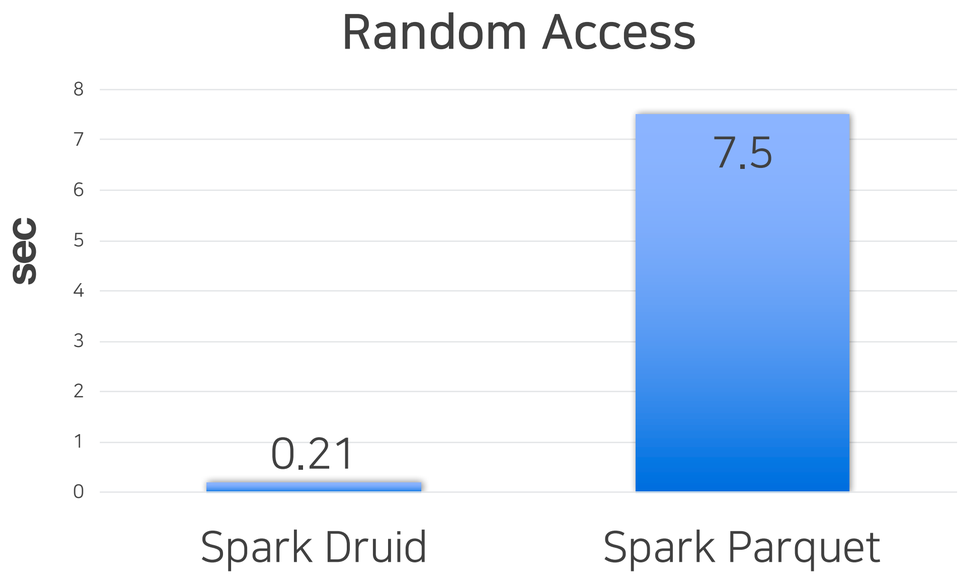
4.2 질의성능 향상시키기

DEVIEW 2018

3) Spark-Druid Connector – 응답시간 비교

(Dataset : **44억건 로그 저장**)





4.3 저장성능 향상시키기

DEVIEW 2018

1) Hash Partition 확장

Single Dim. Partition처럼
Hash Partition을
사용할 수 있지 않을까?

	Ingest Job	특징	아쉬운점
Single Dimension	• GroupBy Dim. • Dim. Selection • Index Generator	• DimValue 사전순 분포 • Shard Pruning 가능	• 중간결과로 인한 Resource 사용 증가 • Shard 개수 지정 불가
Hash	• Index Generator	• 빠른 입수성능 • 입수 Resource ▼	• Shard Pruning 불가

4.3 저장성능 향상시키기

DEVIEW 2018

1) Hash Partition 확장

Ingest

Index Generator Job

- Shard Spec에 다음정보 전달
  - partitionDimensions
  - hashPrunable

HashBasedNumberedShardSpec

- hashPrunable = true

```
partitionDimensions.length = 1
&& isSingleValueDimension
```

Query

HashBasedNumberedShardSpec

- getDomain : shard range 반환
- *getHashShardNum* 추가
  - partition value -> shardNum

DimFilterUtils

- filterShards 수정

```
if (hashPrunable == true)
    include(getHashShardNum(v))
```

4.3 저장성능 향상시키기

DEVIEW 2018

1) Hash Partition 확장

	Ingest Job	특징	아쉬운점
Single Dimension	• GroupBy Dim. • Dim Selection • Index Generator	• DimValue 사전순 분포 • Shard Pruning 가능	• 중간결과로 인한 Resource 사용 증가 • Shard 개수 지정 불가
Hash (Partition Dimension 1개)	• Index Generator	• 빠른 입수성능 • 입수 Resource ▼ • Shard Pruning 가능	• ~~Shard Pruning 불가~~

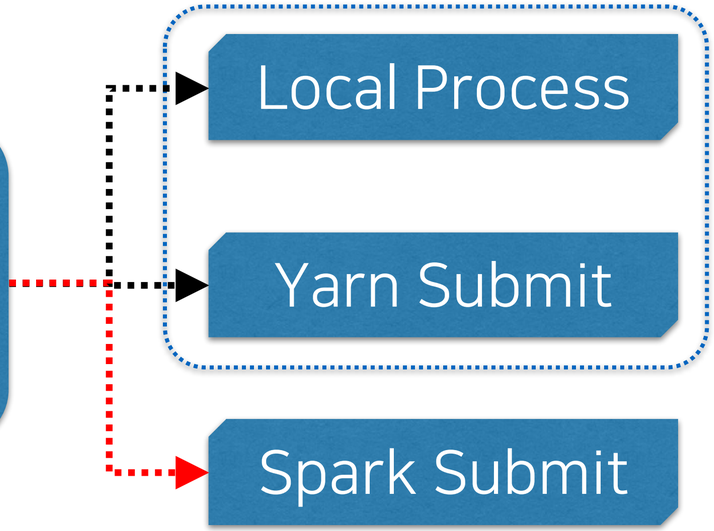
4.3 저장성능 향상시키기

DEVIEW 2018

2) Druid Spark Batch

- 참고) https://github.com/metamx/druid-spark-batch

```
{
  "type": "index_spark",
  "spec": {
    "ioConfig": {"type": "hadoop"...},
    "dataSchema": {"dataSource": "logs"...},
    "tuningConfig": {
      "indexSpec": {...},
      "buildV9Directly": "true",
      "jobProperties": {
        "spark.executor.memory": "6G",
        "spark.executor.instances": "4",
        "spark.executor.cores": "2"
      },
      "type": "hadoop",
      "partitionsSpec": {"type": "hashed"...}
    }
  }
}
```



MapReduce

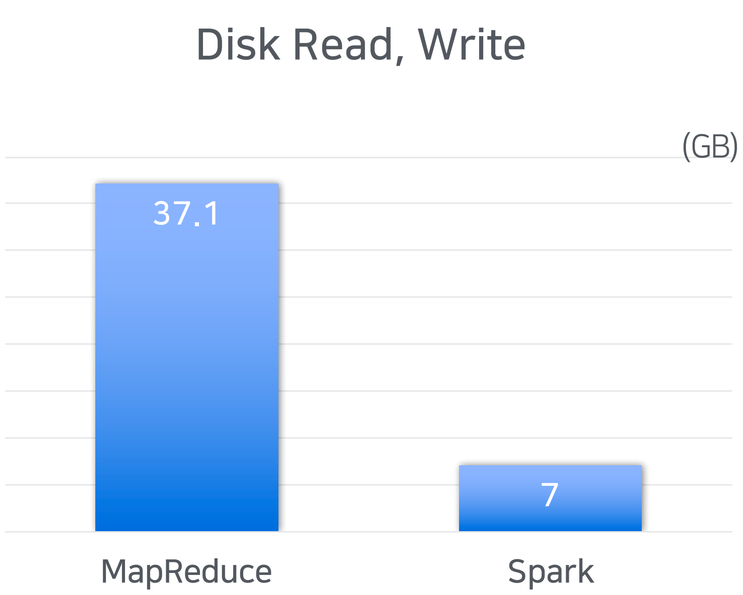
- Ingest 1.4GB Parquet
  - Read : 16.6GB
  - Write : 20.5GB

- GroupBy, DimSelection on Memory
- Indexing with Druid Library

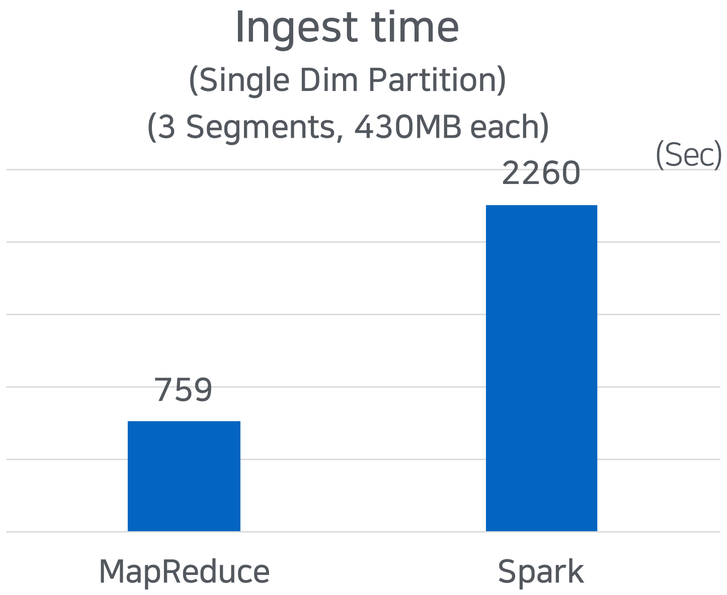
4.3 저장성능 향상시키기

DEVIEW 2018

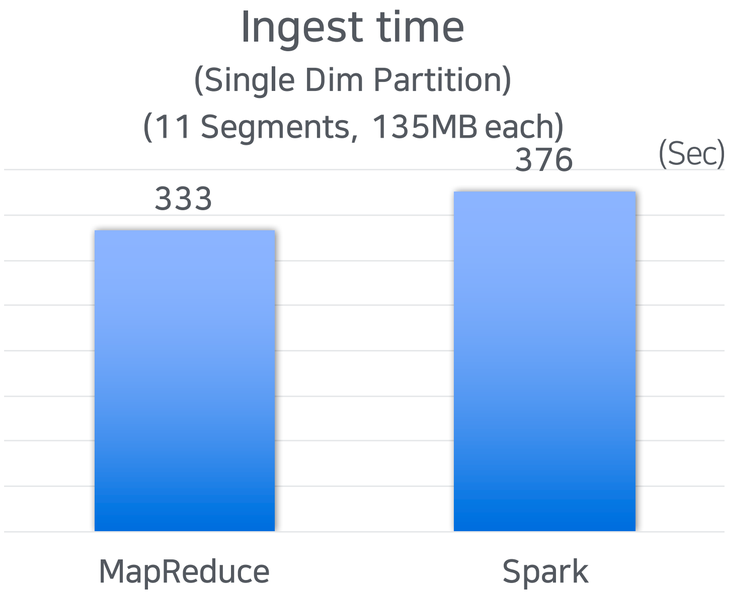
2) Druid Spark Batch



GB, lower is better



Seconds, lower is better



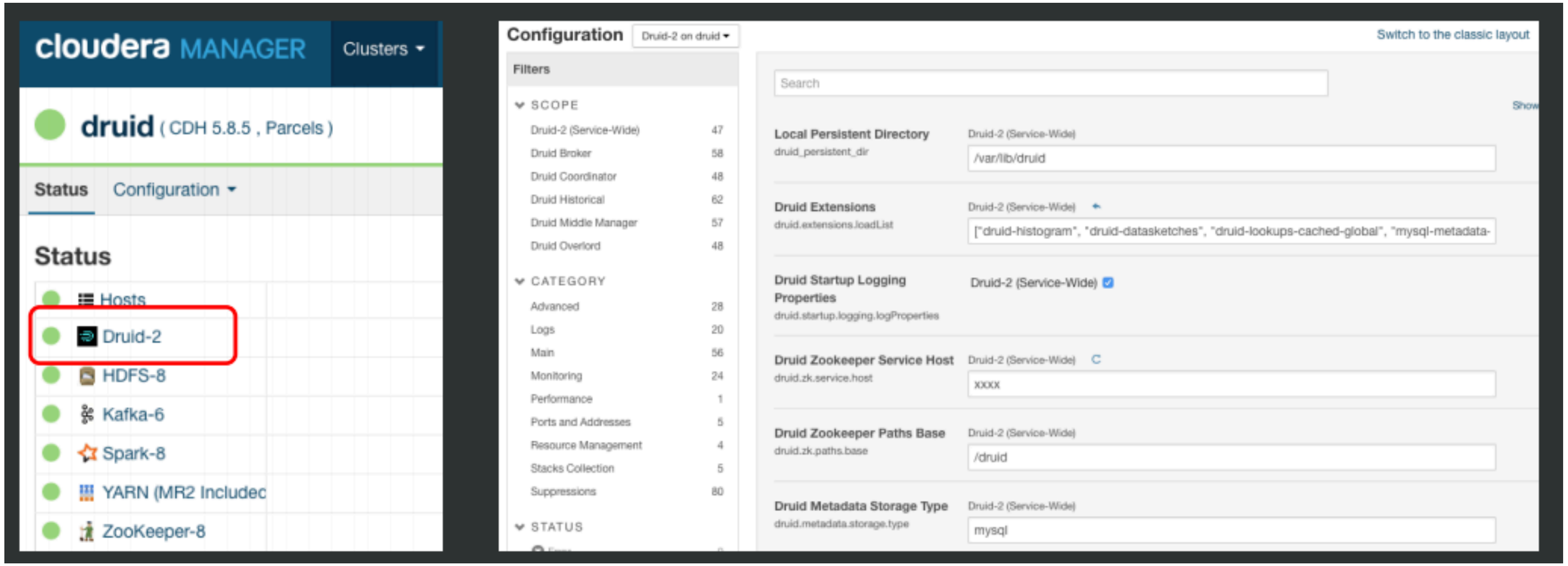
Seconds, lower is better

4.4 편한 운영을 위한 노력

DEVIEW 2018

CDH 연동

- 참고) https://github.com/knoguchi/cm-druid



Q & A

질문은 Slido에 남겨주세요.

sli.do

#deview

TRACK 1

Appendix

Appendix

DEVIEW 2018

- About team
- Comparison of Data Size
- Druid Configuration
- Ingest Spec for External Resource (Yarn, HDFS)
- How Kafka Indexing Service Works
- Druid Query Flow

About team

DEVIEW 2018

- 네이버 콘텐츠 통계서비스 소개 및 구현 경험 공유
- 빅데이터 다차원 분석 플랫폼, Kylin
- Web Analytics at Scale with Elasticsearch @ naver.com
  - Part1
  - Part2
- Kudu를 이용한 빅데이터 다차원 분석 시스템 개발
- Web analytics at scale with Druid at Naver (Strata 2018)

Comparison of Data Size

DEVIEW 2018

Primary data size

Parquet	Apache Druid	Elasticsearch	Apache Kudu
7GB	34GB	50GB	14GB

Druid Configuration

DEVIEW 2018

Druid Deployment

- 10 Broker Nodes
- 40 Historical Nodes
- 2 Overlord & MiddleManager Nodes
- 2 Coordinator Nodes
- 10 Yarn & HDFS Nodes for Batch Ingestion
- Spark Standalone Cluster runs on Historical Nodes for Locality

Druid Configuration

DEVIEW 2018

Druid Deployment

- Version : 0.11.0
- H/W Spec for Broker & Historical
  - CPU : 40 cores (w/ hyperthread)
  - RAM : 128GB
  - HDD : SSD w/ RAID 5

Druid Configuration

DEVIEW 2018

Memory Configuration

Configuration	Broker	Historical
-Xmx	20GB	12GB
-XX:MaxDirectMemorySize	30GB	45GB
druid.processing.numMergeBuffers	10	20
druid.processing.numThreads	20	30
druid.processing.buffer.sizeBytes	512MB	800MB
druid.cache.sizeInBytes	0	5GB
druid.server.http.numThreads	40	40

Ingest Spec for External Resource

DEVIEW 2018

Use Yarn External Resource for Batch Ingestion

```
"tuningConfig": {
  "type": "hadoop",
  "jobProperties": {
     "yarn.resourcemanager.hostname" : "host1.com",
     "yarn.resourcemanager.address" : "host1.com:8032",
     "yarn.resourcemanager.scheduler.address": "host1.com:8030",
     "yarn.resourcemanager.webapp.address": "host1.com:8088",
     "yarn.resourcemanager.resource-tracker.address": "host1.com:8031",
     "yarn.resourcemanager.admin.address": "host1.com:8033"
   }
}
```

Ingest Spec for External Resource

DEVIEW 2018

Use External HDFS for intermediate M/R output

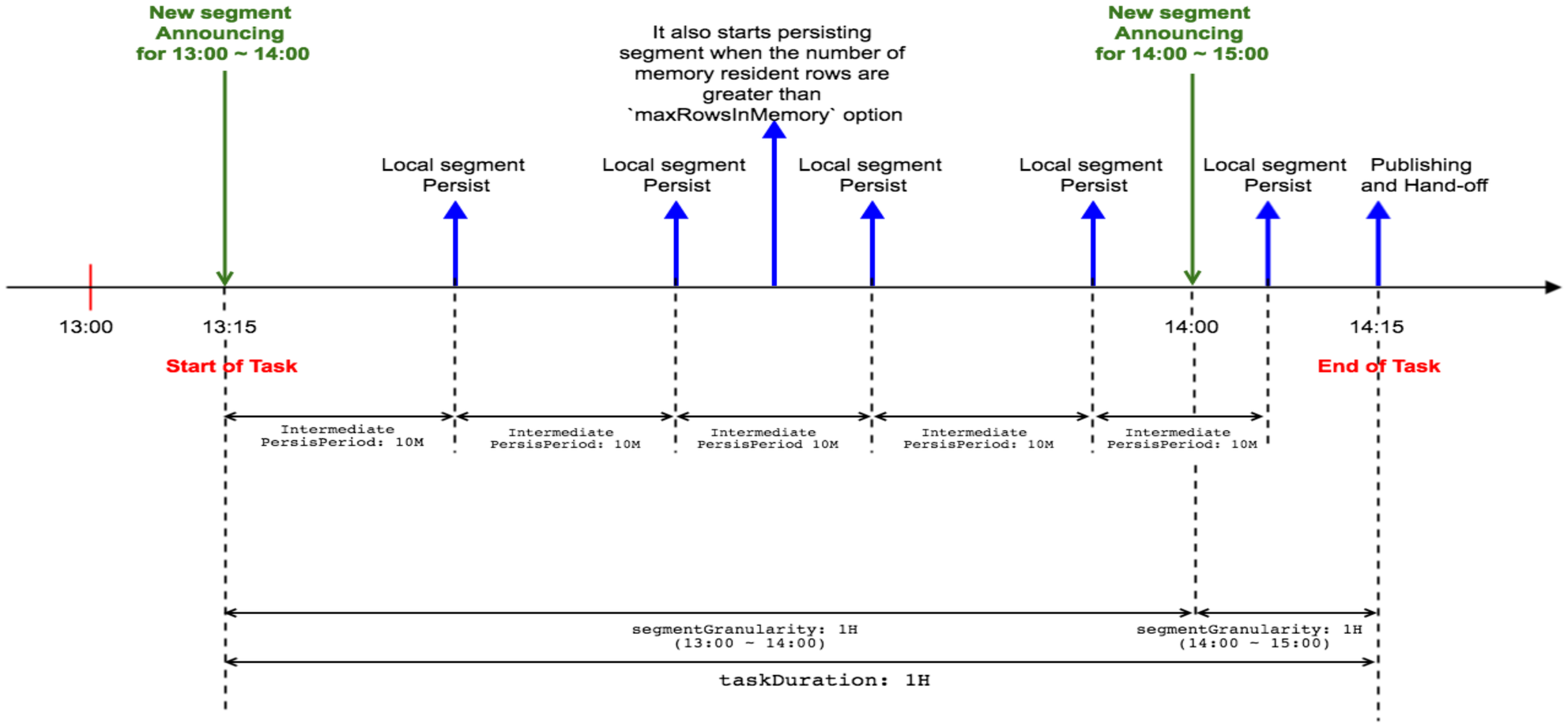
```
"tuningConfig": {
  "type": "hadoop",
  "jobProperties": {
     "fs.defaultFS": "hdfs://DEFAULT_FS:8020",
     "dfs.namenode.http-address": "NAMENODE:50070",
     "dfs.namenode.https-address": "NAMENODE:50470",
     "dfs.namenode.servicerpc-address": "NAMENODE:8022"
   }
}
```

How Kafka Indexing Service Works

DEVIEW 2018

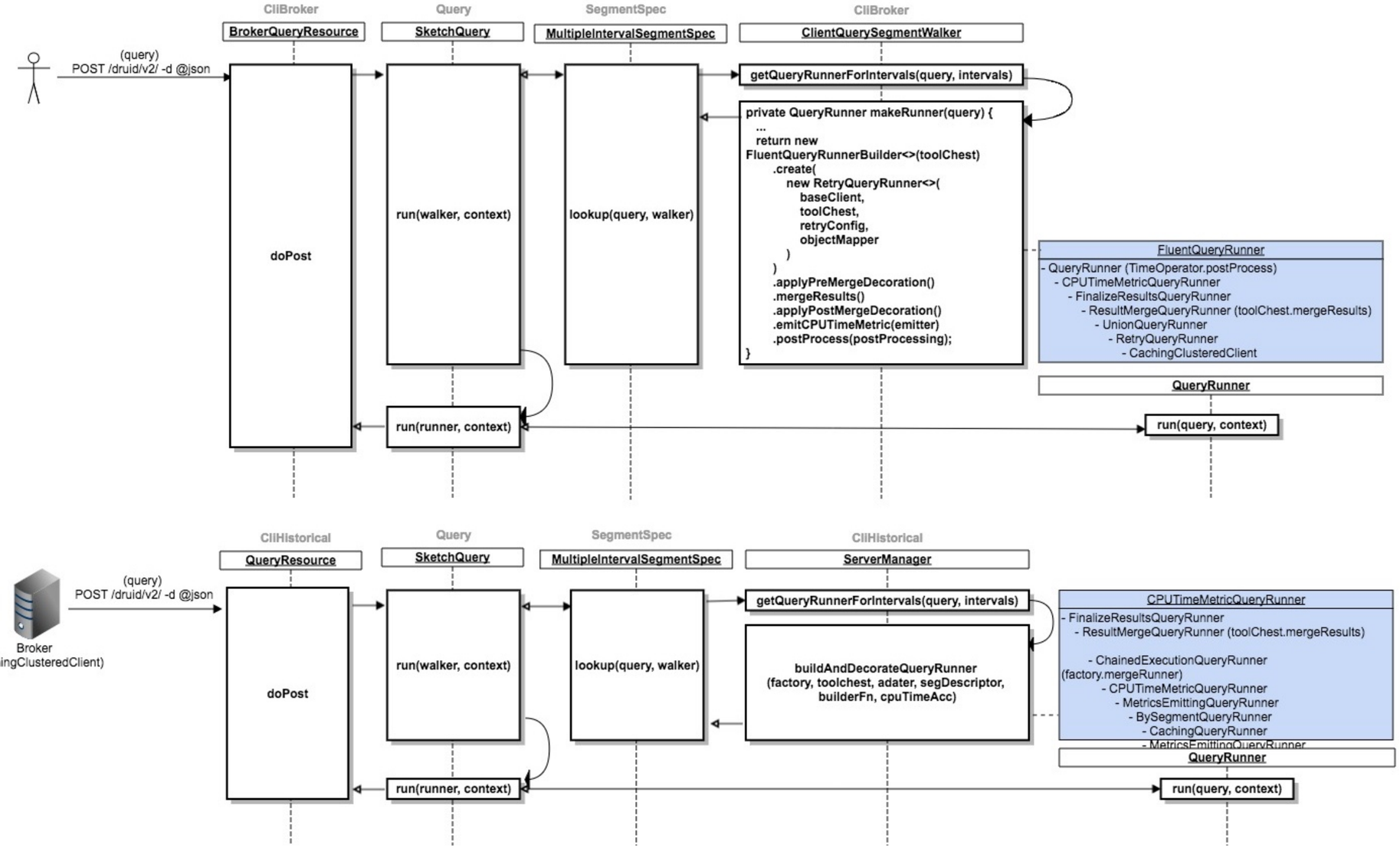
```
[Kafka Indexing Service Conf.]

segmentGranularity: 1H
taskDuration: 1H
intermediatePersistPeriod: 10M
maxRowsInMemory: 600000
maxRowsPerSegment: 7000000
```



Druid Query Flow

DEVIEW 2018



Thank you