



Zeppelin (powered by Apache Spark)

으로 데이터 분석하기

2014-11-05

스사모 (한국 스파크 사용자 모임)

<https://www.facebook.com/groups/sparkkoreauser/>

김상우, VCNC(비트윈)

sangwookim.me@gmail.com

Apache Spark?

- MapReduce 와 유사한 작업이 가능
- 확장성 (Spark SQL, Spark Streaming, MLlib, GraphX)
- MapReduce보다 훨씬 간단한 인터페이스, 배우기 쉬움 (Scala, REPL)
- 작업 종류에 따라 MapReduce의 5배~50배 빠름 (In-Memory Data)
- Hadoop Storage 호환 (HDFS, HBase, S3, ..)

왜 필요한가?

- MapReduce, Hive (기존의 지배 기술들)
 - 매우 강력하지만, 작업이 복잡할수록 비효율적이다. (중간 결과를 계속해서 HDFS에 저장)
 - API가 복잡하고, MR Job 여러개를 Chaining해서 작업을 만들어놓으면, 유지보수하기가 어렵다.

Spark Key Concept

- RDD (Resilient Distributed Datasets)
 - 클러스터 전체에서 공유되는 리스트, 메모리상에 올라가있음. (메모리 부족한 경우, 디스크에 spill)
 - map, reduce, count, filter, join 등 다양한 작업 가능
 - 여러 작업을 설정해두고, 결과를 얻을 때 lazy하게 계산
- Scala
 - 데이터 분석 하기에 아주 좋은 언어
 - 강력한 expression, Java와의 호환성
 - Interactive Shell (REPL)

Spark은 좋다

- 수십대의 Hadoop Cluster로 큰 작업을 돌려야 했던 경우, 10대 이하의 Cluster로 대체할 수 있다
- 클러스터로 돌려야 하던 작업을 1~2대로 돌릴 수 있다
- 수십분 기다려야 하던 작업이 1분만에 완료된다
- MR 작업 코드 만들고, 패키징하고, submit하고 하던 복잡한 과정이, shell에서 코드 한줄 치는것으로 대체된다
- 처음 접하는 사람도 배우기 쉽다

Code Examples (1)

Word Count

Word Count

```
val file = spark.textFile("hdfs://...")  
  
val counts = file.flatMap(line => line.split(" "))  
  
    .map(word => (word, 1))  
  
    .reduceByKey(_ + _)  
  
counts.saveAsTextFile("hdfs://...")
```

Code Examples (2)

Getting
Between PC Ver. Download

Getting Download Data

```
case class CloudFrontPcVerChart(val date: String, val country: String, val ip: String, val http_method: String, val ua: String)
```

```
val cloudFrontPcVerLogs = "s3n://assets-between-pc-logs/*2014-10-*
```

```
val cloudFrontPcVerDownloadLogs =  
sc.textFile(cloudFrontPcVerLogs).filter(_ contains "/downloads/  
setup.exe").map(x => x.split("\t"))
```

```
cloudFrontPcVerDownloadLogs.first
```

```
val cloudFrontPcVerDownloadChart =  
cloudFrontPcVerDownloadLogs.map(arr => CloudFrontPcVerChart(arr(0),  
IP2C.get(arr(4)), arr(4), arr(5), arr(10)))
```

```
cloudFrontPcVerDownloadChart.registerAsTable("pc_ver_download")
```

Querying Data

```
select country, count(1) value  
from pc_ver_download  
group by country  
order by value desc  
limit 10
```

Simple enough!

Result

country	value
KR	,529
CN	,338
TH	999
MY	424
US	733
IN	251
JP	065
ID	047
SG	044
TW	5

* Visualization powered by Zeppelin

확장 프로젝트들

- Spark SQL
- Spark Streaming
- MLlib
- GraphX
- SparkR (예정)
- Zeppelin

Zeppelin

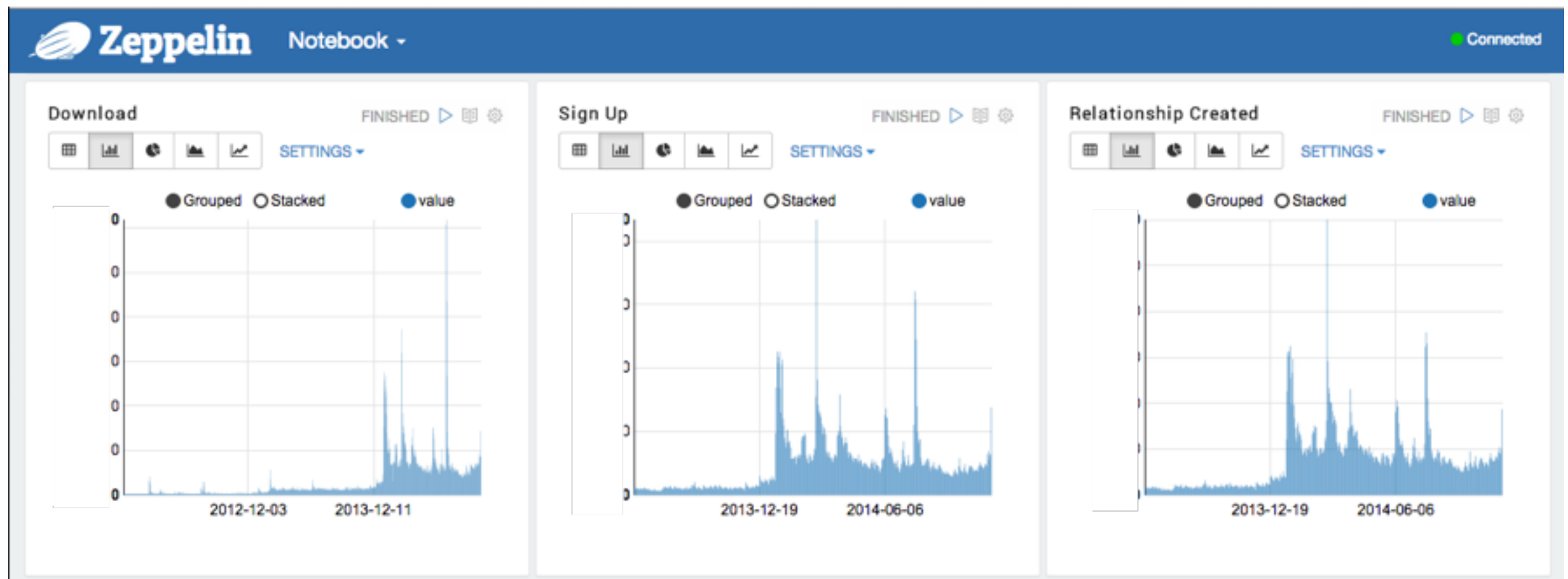
- A web-based notebook for Apache Spark (<http://zeppelin-project.org>)
- Open source (<https://github.com/NFLabs/zeppelin>)



Zeppelin

- Early stage 프로젝트 (Github 50 Star)
- 1~2년 사이에 엄청 유명해질 프로젝트
- 10줄만 커밋해도 contributor 로 넣어주는 좋은 프로젝트
- 쉬운 설치, 실행하면 Spark을 내부에서 띄워줌 (외부 Cluster와 연결도 가능)

Zeppelin



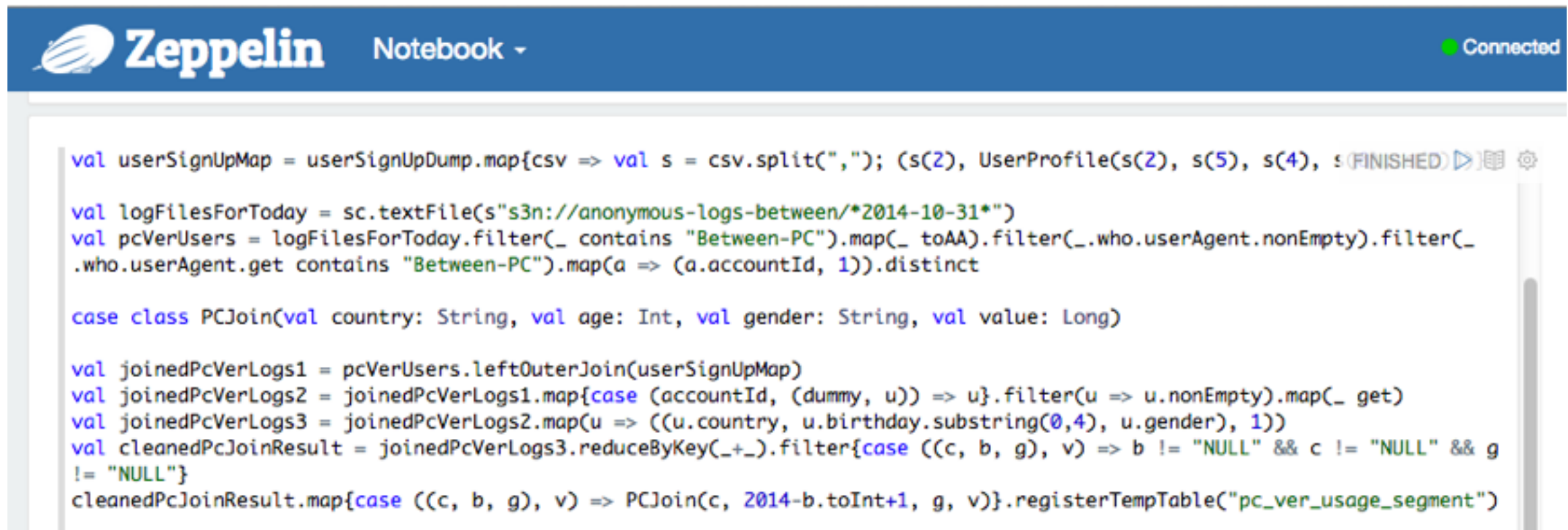
Implementing dashboard via Zeppelin with few codes and queries

Zeppelin

Spark & Zeppelin

Live Demo

ETL부터 분석, visualisation까지 하나의 툴로 모두 처리



The image shows a screenshot of the Zeppelin Notebook interface. The top header is blue with the Zeppelin logo on the left, the word "Notebook" in the center, and a green "Connected" status indicator on the right. The main area contains Scala code for processing user sign-up data and PC usage logs. The code includes variable declarations, file reading, filtering, joining, and saving results to a temporary table.

```
val userSignUpMap = userSignUpDump.map{csv => val s = csv.split(","); (s(2), UserProfile(s(2), s(5), s(4), s(6)))}

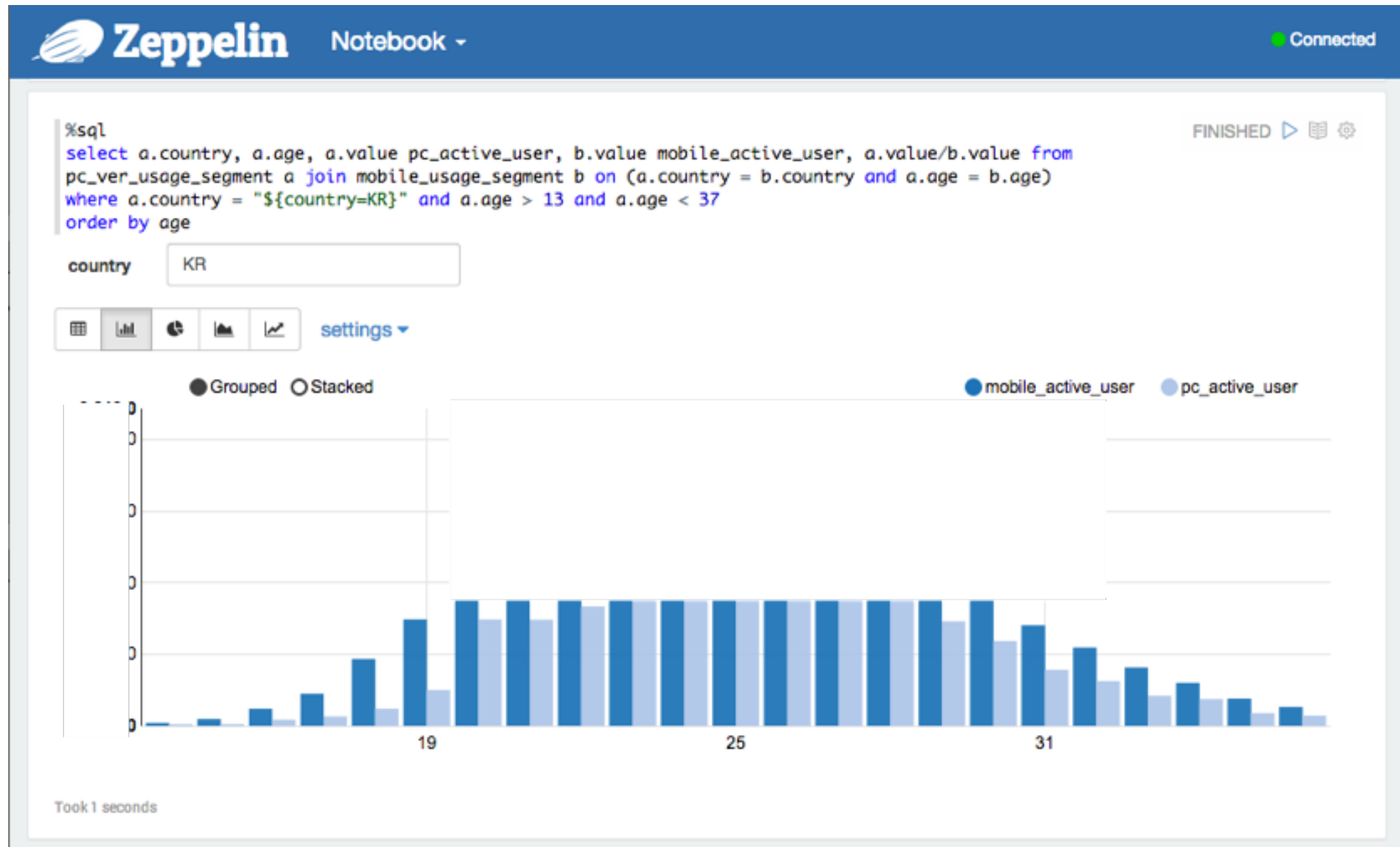
val logFilesForToday = sc.textFile(s"s3n://anonymous-logs-between/*2014-10-31*")
val pcVerUsers = logFilesForToday.filter(_ contains "Between-PC").map(_ toAA).filter(_ who.userAgent.nonEmpty).filter(_ who.userAgent.get contains "Between-PC").map(a => (a.accountId, 1)).distinct

case class PCJoin(val country: String, val age: Int, val gender: String, val value: Long)

val joinedPcVerLogs1 = pcVerUsers.leftOuterJoin(userSignUpMap)
val joinedPcVerLogs2 = joinedPcVerLogs1.map{case (accountId, (dummy, u)) => u}.filter(u => u.nonEmpty).map(_ get)
val joinedPcVerLogs3 = joinedPcVerLogs2.map(u => ((u.country, u.birthday.substring(0,4), u.gender), 1))
val cleanedPcJoinResult = joinedPcVerLogs3.reduceByKey(_+_).filter{case ((c, b, g), v) => b != "NULL" && c != "NULL" && g != "NULL"}
cleanedPcJoinResult.map{case ((c, b, g), v) => PCJoin(c, 2014-b.toInt+1, g, v)}.registerTempTable("pc_ver_usage_segment")
```

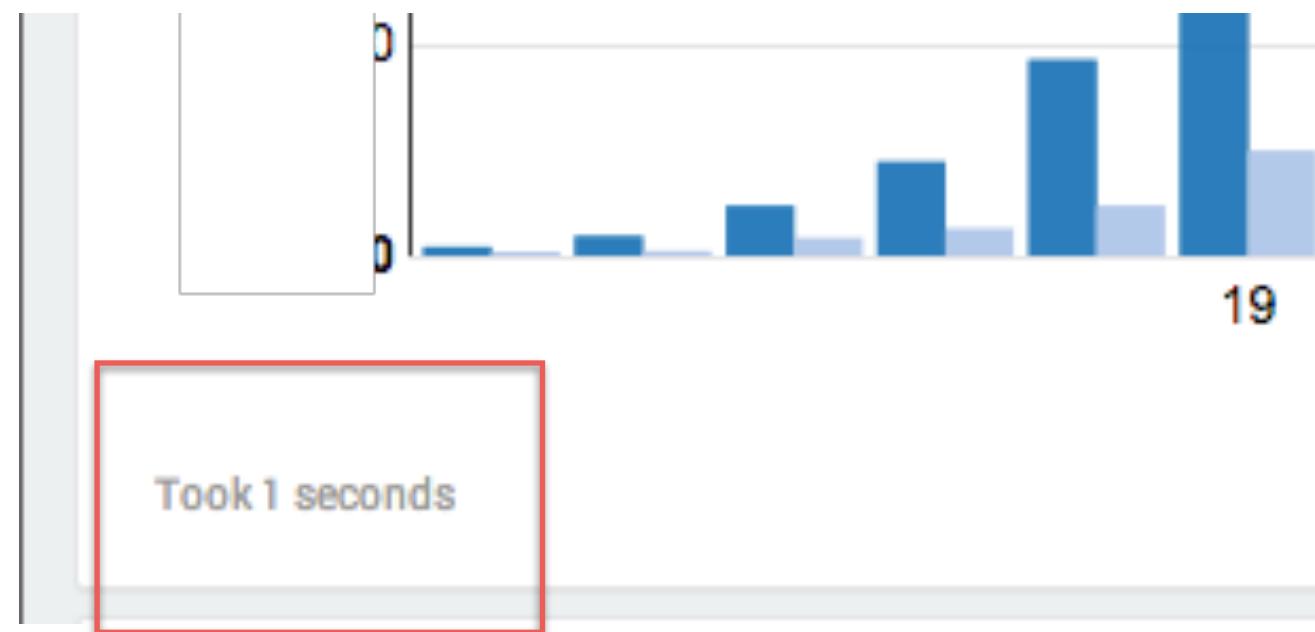
Live Demo를 Keynote에 넣기가 어려워 스크린샷으로 대체합니다

ETL부터 분석, visualisation까지 하나의 툴로 모두 처리



Live Demo를 Keynote에 넣기가 어려워 스크린샷으로 대체합니다

Interactive! 코드나 쿼리를 넣고 거의 즉시 결과가 나옴



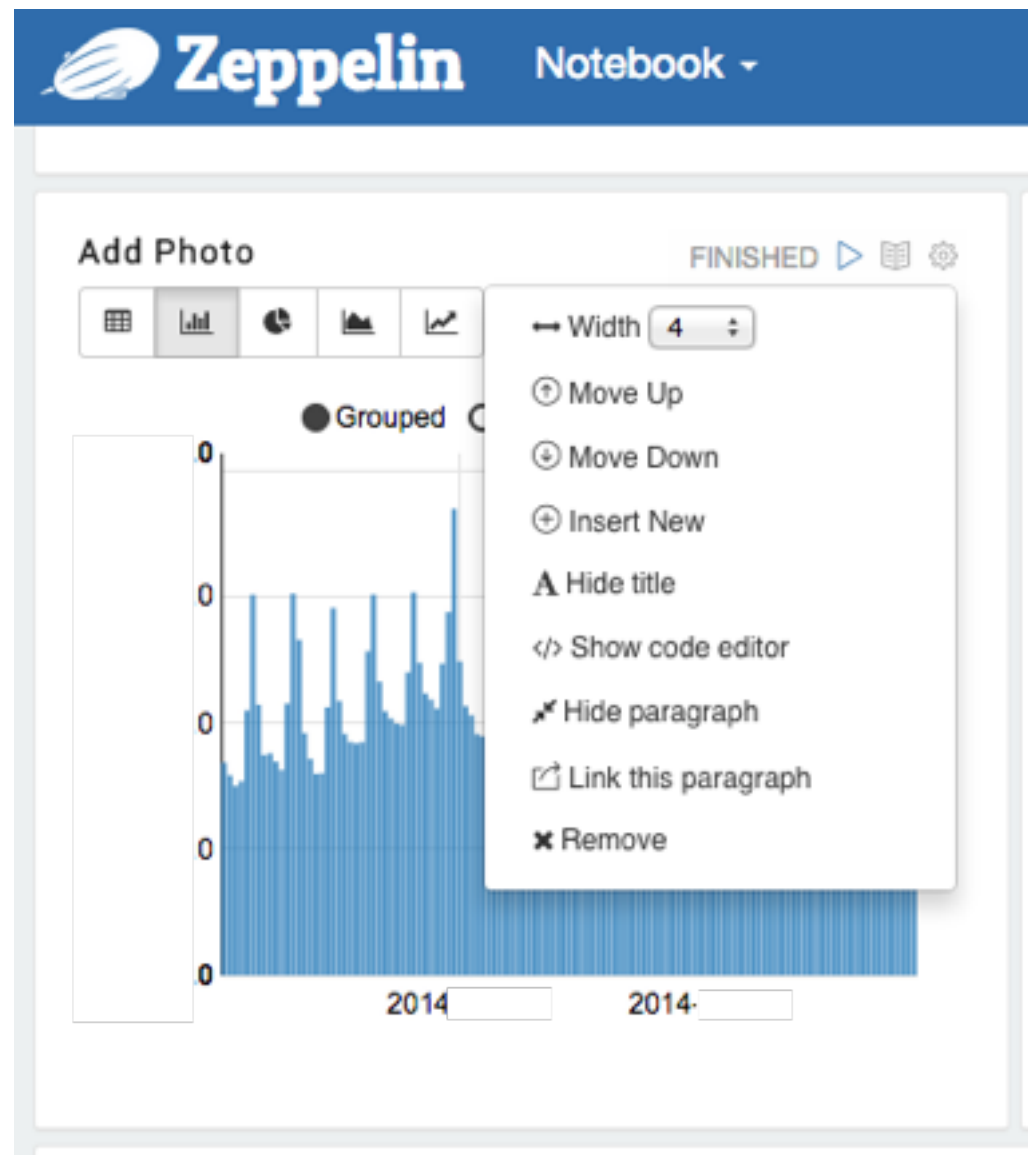
Live Demo를 Keynote에 넣기가 어려워 스크린샷으로 대체합니다

Spark SQL과 결합하여 Visualisation 툴로도 높은 가능성



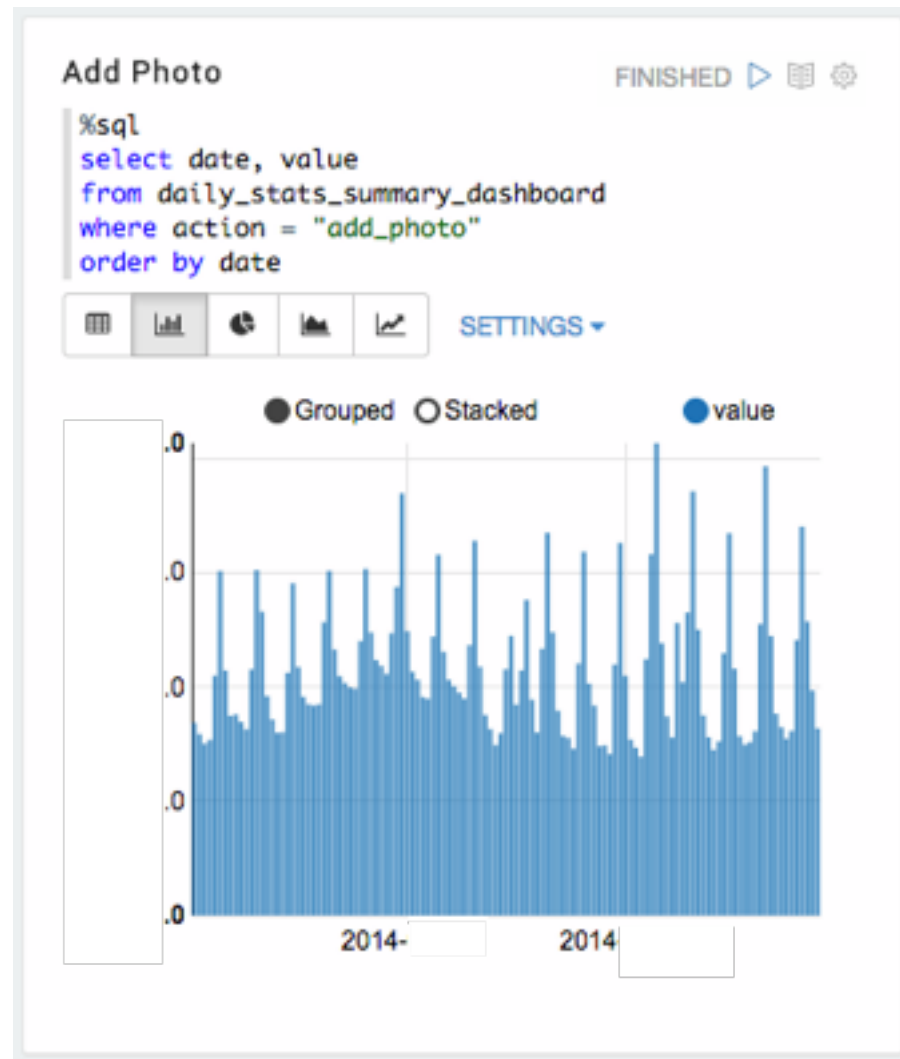
Live Demo를 Keynote에 넣기가 어려워 스크린샷으로 대체합니다

Spark SQL과 결합하여 Visualisation 툴로도 높은 가능성



Live Demo를 Keynote에 넣기가 어려워 스크린샷으로 대체합니다

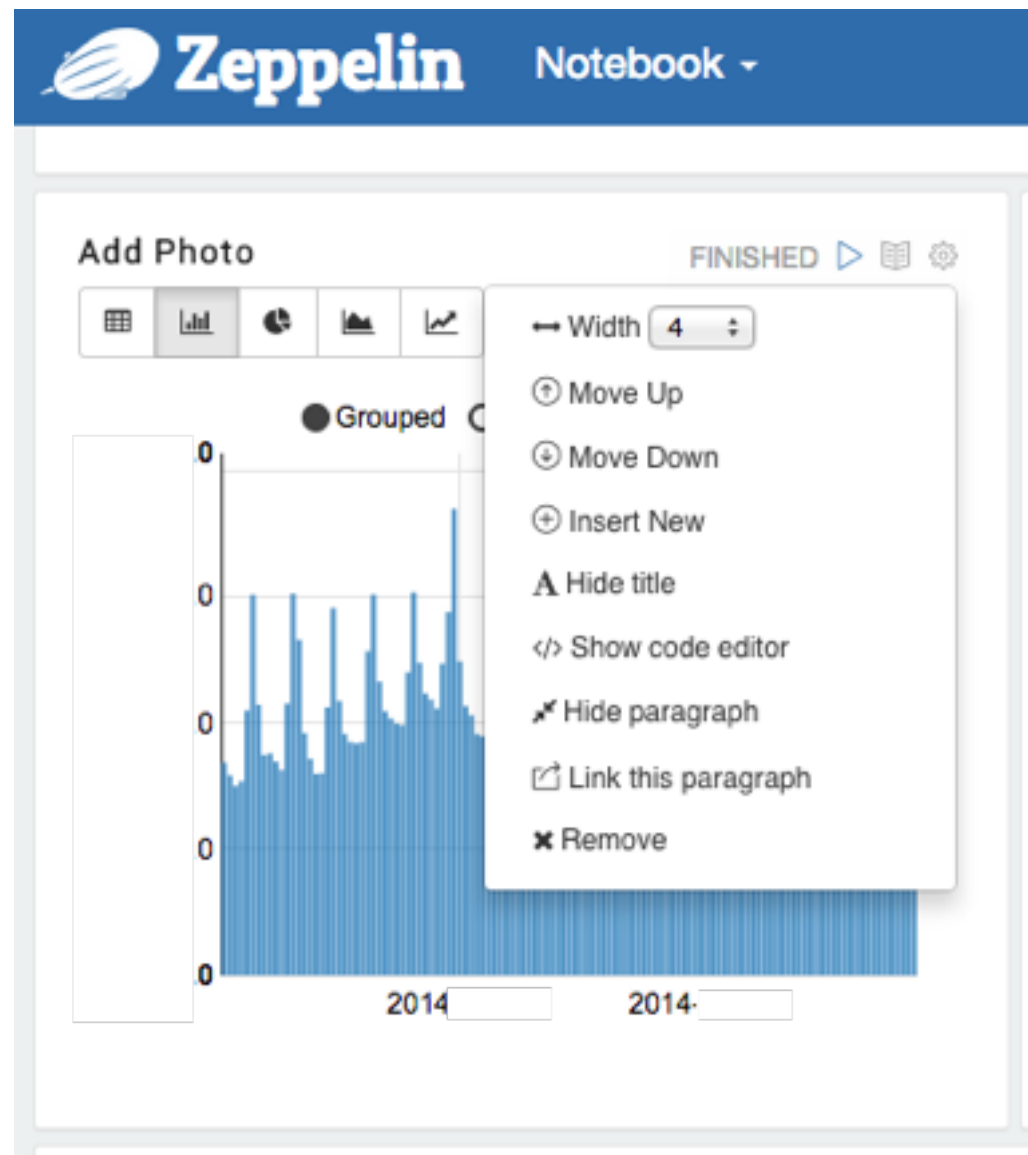
Spark SQL과 결합하여 Visualisation 툴로도 높은 가능성



간단한 SQL Query로 대쉬보드를
순식간에 만듦

Live Demo를 Keynote에 넣기가 어려워 스크린샷으로 대체합니다

Spark SQL과 결합하여 Visualisation 툴로도 높은 가능성



위치, 넓이 등 조절

Live Demo를 Keynote에 넣기가 어려워 스크린샷으로 대체합니다

Zeppelin

- 간단하게 데이터 분석을 시작해보려는 사람들에게 추천
- 민첩하게 이런저런 데이터를 살펴보고 분석하려는 사람들에게 추천
- Dashboard을 빠르게 만들고 싶은 사람들에게 추천
- Hot한 Open Source에 참여해보고 싶은 사람들에게 추천
- Spark을 처음 사용하는 경우는 Spark Shell을 먼저 사용해보는것을 추천 (Zeppelin Code Editor의 Auto Completion기능이 보강될 때 까지)

감사합니다