Exploring big data on the cloud

What we know, what is happening, and what we are made of: Open Data, BigQuery, SQL

Felipe Hoffa
Developer Advocate
@felipehoffa, +FelipeHoffa
Felipe Hoffa
Developer Advocate @Google. From Chile to San Francisco to the world.

In 2011 Felipe Hoffa moved from Chile to San Francisco to join Google as a Software Engineer. Since 2013 he's ... See full bio

San Francisco in United States

Spoken at 32 events in 15 countries

BIG DATA
Big Data is LARGE?
Big Data is

HEAVY?
Big Data is SLOW?
Big Data is

COMPLEX?
What if Big Data was

Sizeless
Weightless
Fast
Easy
What has more data?

el hrlaPg0sn uotthshok2e nsyr ph r.tosl us.2usd3extg a.
nor h dfe1tpi c.ria28r0se. .Mrrm iidCt8so0hsih0moi n
crld5n igA ti n0cd ew 1l,scu bonopg50gl.ek f
arbAm2endlswF3ioisn,la http .disb n ,tml

"""""""
000000112222335588AACFMPaaaaaabbbccccddddesee
eeeeffggggghhhhhhhiiiiiiiiiiikkllllllllmmmmnnnrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr
What has more data?

```
00000112222335588AACFMPaaaaabbbccccddddddee

```

29: ,13:s,11:i,10:t,10:r,10:n,9:l,9:o,8:e,8:.,8:h,7:d,6:0,5:a,
5:g,4:m,4:p,4:c,4:2,4:u,3:b,3:,,2:1,2:5,2:A,2:w,2:3,2:k,2:f,
2:8,1:M,1:C,1:F,1:x,1:P,1:y
What has more data?

Melt 300g chocolate, 3tbsp golden syrup, 125g butter. Add 50g marshmallows, 200g crushed shortbread and sprinkles. And to finish... Pour into 18x28cm tin. Finish with sprinkles. Chill for 20 mins.
The 3 big steps for the data revolution
1. Price
10 Megabyte Hard Disk

$3,495*

5418-12 Top Load Drive
* Factory rebuilt 10MB cartridge disk drive only
A new Canons Data Systems controller is available for $4,695
$1,495 for a brand new Ampex 1MB drive only

We are the CP/M** and MF/M** specialists of Southern California. We can supply you with the latest CP/M ($150) or MF/M ($300) and with Standard Billing ($50) or Custom Billing ($398). Immediate delivery worldwide. Domestic and foreign inquiries invited...dealers too.

**CP/M and MF/M are Trademarks of Digital Research.

COMPUTER COMPONENTS

908 Sepulveda Boulevard
Van Nuys, California 91411
(213) 703-7411

Byte Magazine, 1980
2. Access
TBL: Alright, "raw data now"!

http://www.ted.com/talks/tim_berners_lee_on_the_next_web.html
3. Speed
Search in 1996
"Search is solved" -- 1996

Excite – Born in 1993
Yahoo! - Born in 1994
WebCrawler – Born in 1994
Lycos – Born in 1994
Infoseek – Born in 1994
AltaVista – Born in 1995
Inktomi – Born in 1996
"Search is solved" -- 1996

Excite – Born in 1993
Yahoo! – Born in 1994
WebCrawler – Born in 1994
Lycos – Born in 1994
Infoseek – Born in 1994
AltaVista – Born in 1995
Inktomi – Born in 1996

Google - 1998
1. PageRank: A new idea
2. Collect the web
3. Create the technology
Idea

Data ← Tech
Google BigQuery

• Fast: terabytes in seconds
• Simple: SQL
• Scaleable: From bytes to petabytes
• No CAPEX: Always on
• Interoperable: Tableau, R, Python...
• Instant sharing
• Free monthly quota
How many pageviews does Wikipedia have in a month?

SELECT SUM(requests) FROM [fh-bigquery:wikipedia.pagecounts_201412]

bigquery.cloud.google.com/table/fh-bigquery:wikipedia.pagecounts_201412
How to load data into BigQuery

```bash
bq load -F" " --quote "" fh-bigquery:wikipedia.
pagecounts_20150828_10 pagecounts-20150828-100000.gz
language,title,requests:integer,content_size:integer
```
How BigQuery works

Tree Structured Query Dispatch and Aggregation

SELECT state, COUNT(*) count_babies
FROM [publicdata:samples.natality]
WHERE year >= 1980 AND year < 1990
GROUP BY state
ORDER BY count_babies DESC
LIMIT 10

SELECT state, year
"Streaming data into Google BigQuery with special guest Streak..."

"BigQuery, IPython, Pandas and R for data science, starring Pearson"

"Shine with BigQuery: The 30 Terabyte challenge"
Many more

- Games and social media analytics
- Advertising campaign optimization
- Sensor data analysis
- POS-Retail analytics
- Web Logs, machine logs, infrastructure monitoring
- Mobile games, application analytics
A query that extracts the lat,lon for the last 24 hours of GDELT news:

```sql
SELECT date, lat, lon, cnt from (
    SELECT date, lat, lon, COUNT(*) as cnt
    FROM (  
        SELECT REGEXP_REPLACE(STRING(date), '(....)(..)(..)(..)(..)(..)', r'\1-\2-\3 \4:\5:\6') date,
        REGEXP_REPLACE(  
            REGEXP_EXTRACT(  
                SPLIT(V2Locations, ';'),  
                r'^(\d\d\d\d).*\..*\..*\..*\..*\..*\..*\..*',  
                '(.*)$'),  
                '\1') AS lat,
        REGEXP_REPLACE(  
            REGEXP_EXTRACT(  
                SPLIT(V2Locations, ';'),  
                r'^(\d\d\d\d).*\..*\..*\..*\..*\..*\..*\..*',  
                '(.*)$'),  
                '\2') AS lon,
        FROM [gdelt-bq:gdeltv2.gkg@-86400000-]
    )
    WHERE lat is not null
    GROUP BY date, lat, lon
    ORDER BY 3 DESC
)
WHERE cnt >= 3;
```
GDELT: Rows per month (Brasil)

```
SELECT TIMESTAMP(STRING(MonthYear)+'01') month,
       SUM(ActionGeo_CountryCode='BR') Brasil
FROM [g delt-bq:full.events]
WHERE MonthYear>0
GROUP BY 1 ORDER BY 1
```
GDELT: Rows per month (Brasil)

SELECT TIMESTAMP(STRING(MonthYear)+'01') month,
     SUM(ActionGeo_CountryCode='BR') Brasil
FROM [gdelt-bq:full.events]
WHERE MonthYear>0
GROUP BY 1 ORDER BY 1
GDELT: Rows per month (Brasil, normalized)

```
SELECT TIMESTAMP(STRING(MonthYear)+'01') month,
       SUM(ActionGeo_CountryCode='BR')/COUNT(*) Brasil
FROM [gdl-t-bq:full.events]
WHERE MonthYear>0
GROUP BY 1 ORDER BY 1
```
GDELT: Rows per month (Chile, normalized)

SELECT TIMESTAMP(STRING(MonthYear)+'01') month,
       SUM(ActionGeo_CountryCode='CI')/COUNT(*) Chile
FROM [gdelt-bq:full.events]
WHERE MonthYear>0
GROUP BY 1 ORDER BY 1
SELECT TIMESTAMP(STRING(MonthYear)+'01') month,  
SUM(ActionGeo_CountryCode='CI')/COUNT(*) Chile  
FROM [gdelt-bq:full.events]  
WHERE MonthYear>0  
GROUP BY 1 ORDER BY 1

GDELT: Rows per month (Chile, normalized)

October 1988
March 2010
October 2010
Live data
MJ no more: Using Concurrent Wikipedia Edit Spikes with Social Network Plausibility Checks for Breaking News Detection

http://wikipedia-live-monitor.herokuapp.com/

Thomas Steiner
Google Germany GmbH
ABC-Str. 19
20354 Hamburg, Germany
tomac@google.com

Seth van Hooland
Université Libre de Bruxelles
Avenue F.D. Roosevelt, 16
1050 Brussels, Belgium
svhoolan@ulb.ac.be

Ed Summers
Library of Congress
101 Independence Ave, SE
Washington, DC 20540, USA
ehs@pobox.com

ABSTRACT

We have developed an application called Wikipedia Live Monitor that monitors article edits on different language versions of Wikipedia—as they happen in realtime. Wikipedia articles in different languages are highly interlinked. For example, the English article “en:2013_Russian_meteor_event” on the topic of the February 15 meteoroid that exploded over the region of Chelyabinsk Oblast, Russia, is interlinked with “ru:Падение_метеорита_на_Урале_в_2013_году”, the Russian article on the same topic. As we monitor multiple language versions of Wikipedia in parallel, we can exploit this fact to detect concurrent edit spikes of Wikipedia articles covering the same topics, both in only one, and in different languages. We treat such concurrent edit spikes as signals for potential breaking news events, whose plausibility we can check using social network analysis.

1. INTRODUCTION

1.1 Motivation

Shortly after the celebrity news website TMZ broke the premature news that the King of Pop Michael Jackson (MJ) had died,\(^1\) the Internet slowed down.\(^2\) Initially, Wikipedia’s website administrators started noting abnormal load spikes [8]. Shortly afterwards, caching issues caused by a so-called edit war [1] led the site to go down: Wikipedia editors worldwide made concurrent edits to the Michael Jackson Wikipedia article, doing and undoing changes regarding the tense of the article, death date, and the circumstances of the (at the time) officially still unconfirmed fatality. While Wikipedia engineers have worked hard to ensure that future load spikes do not take the site down again, there is without dispute a need for a robust system that can warn administrators about potential load spikes prior to their occurrence.
SELECT title, c, day, editors
FROM (SELECT title, COUNT(title) as c, DATE(SEC_TO_TIMESTAMP(timestamp)) day, RANK() OVER(PARTITION BY COUNT_DISTINCT(user) ORDER BY c DESC)
FROM [bgpipelines:wikipediaEdits.wikiStreaming@-2246400000-] 
WHERE NOT title CONTAINS ":
AND NOT bot
GROUP BY title, day
HAVING c>100
AND editors>20
)
WHERE rank<=1
ORDER BY day DESC
Cloud Dataflow

- GFS (2002)
- MapReduce (2004)
- Dremel (2006)
- BigQuery (2008)
- Flume (2010)
- MillWheel (2013)
- Cloud Dataflow (2012)
Pipeline is a Directed Acyclic Graph (DAG) of data flow, just like:
MapReduce/Hadoop jobs
Apache Spark jobs

Multiple **Inputs** and **Outputs**
Cloud Storage
BigQuery tables
Cloud Datastore
Cloud Pub/Sub
Cloud Bigtable
... and any external systems
The Auto Complete Example

Pipeline p = Pipeline.create();
p.begin();
  .apply(TextIO.Read.from(...))
  .apply(ParDo.of(new ExtractTags()))
  .apply(Count.create())
  .apply(ParDo.of(new ExpandPrefixes()))
  .apply(Top.largestPerKey(3))
  .apply(TextIO.Write.to(...));
p.run()
Worker Scaling

- 800 RPS
- 1,200 RPS
- 5,000 RPS
- 50 RPS
Real-time Monitoring UI
Living cities
SELECT ROUND(pickup_latitude, 4) as lat, ROUND(pickup_longitude, 4) as long, 
COUNT(*) as num_pickups
FROM [nyc-tlc:yellow.trips_2015]
WHERE (pickup_latitude BETWEEN 40.61 AND 40.91) AND (pickup_longitude BETWEEN -74.06 
AND -73.77 )
GROUP BY lat, long
NYC taxi pickups 00:00

- Short trips
- Medium trips
- Long trips

https://www.reddit.com/r/bigquery/comments/3fo9ao/nyc_taxi_trips_now_officially_shared_by_the_nyc/ctt5jsi
NYC TAXI HACKATHON

https://www.youtube.com/watch?v=l8MLIfU21pk
Taxi pickups in NYC by hour and trip length [OC]

- 4 Points
- 4 ups, 0 downs
- 835,637 Views
- 2.69 TB Bandwidth usage
- 5 Comments
- 4 Favorites

Uploaded Aug. 7th

Hourly Cumulative

Times displayed are UTC. Your local timezone is UTC-7.

Aug 7, 2015 - Aug 8, 2015

View counts over time.
Reddit
Cliques: Subreddits that share the same commenters
Cliques part II: More subreddits that share the same commenters
Most popular words for each sub

May 2015

/r/movies

/movies

/r/trees

/smoke

/r/politics

/r/science

@felipehoffa

https://www.reddit.com/r/bigquery/comments/3cej2b/17_billion_reddit_comments_loaded_on_bigquery/
The weather
Weather in Rio de Janeiro, 2014

```
SELECT TIMESTAMP(year+mo+da) day, min, max, IF(prcp=99.99,0,prcp) prcp
FROM [fh-bigquery:weather_gsod.gsod2014] a
JOIN [fh-bigquery:weather_gsod.stations] b
ON a.wban=b.wban and b.usaf=a.stn
WHERE country='BZ' AND name='RIO DE JANEIRO /AER'
ORDER BY 1
```
What if Big Data was

Fast & Easy
Questions?

News: reddit.com/r/bigquery
Ask: stackoverflow.com

Felipe Hoffa
@felipehoffa +FelipeHoffa

Rate me?

bit.ly/bqfeedback
Freebase: What we know
Free and open. Licensed as **CC-BY**

Open for anyone to contribute.

A source for Google’s Knowledge Graph

Download the entire graph as RDF

**42.9M** people places and things

**2.4B** triples about those things

**DEC 2014 Update: Migrating to Wikidata**
Age distribution

SELECT age, COUNT(*) c
FROM [compute_ages]
WHERE age BETWEEN 1 AND 80
GROUP BY 1
ORDER BY 1
Exploring the Notability Gender Gap
SELECT title, count, iso FROM (SELECT title, count, c.iso iso, RANK() OVER (PARTITION BY iso ORDER BY count DESC) rank FROM (SELECT a.title title, SUM(requests) count, b.person person FROM [fh-bigquery:wikipedia.pagecounts_20140410_150000] a JOIN (SELECT REGEXP_REPLACE(obj, '/wikipedia/id/', '') title, a.sub person FROM [fh-bigquery:freebase20140119.triples_nolang] a JOIN (SELECT sub FROM [fh-bigquery:freebase20140119.people_gender] WHERE gender='/m/02zsn') b ON a.sub=b.sub WHERE obj CONTAINS '/wikipedia/id/' AND pred = '/type/object/key' GROUP BY 1,2) b ON a.title = b.title GROUP BY 1,3) a JOIN EACH [fh-bigquery:freebase20140119.people_place_of_birth] b ON a.person=b.sub JOIN [fh-bigquery:freebase20140119.place_of_birth_to_country] c ON b.place_of_birth=c.place) WHERE rank=1 ORDER BY count DESC

http://devnook.github.io/GenderMaps/maplabels/