DATASTAX

From 0 to Cassandra on AWS in 30 days

Tsunami alerting with Cassandra

LesFurets.com
Comparez et achetez futé

Andrei Arion - Geoffrey Bérard - Charles Herriau
@BeastieFurets

1	Context
2	The project, step by step
3	Feedbacks
4	Conclusion

DATASTAX

LesFurets.com

Comparez et achetez futé

Cassandra Days brought to you by DataStax

LesFurets.com

- 1st independant insurance aggregator
- A unique place to compare and buy hundreds of insurance products
- Key figures:
  - 2 500 000 quotes/year
  - 31% market share on car comparison (january 2015)
  - more than 50 insurers

LesFurets.com

LesFurets.com
Comparez et achetez futé

@BeastieFurets

DATASTAX

LesFurets.com
Comparez et achetez futé

4 Teams | 25 Engineers | Lean Kanban | Daily delivery

JavaOne™

LeanKanban
France

BreizhCamp
Mix de technologies

KANBAN DAY

DEVOXX FRANCE
Du 8 au 10 avril 2015
Conférence Développeurs

mix-IT

LesFurets.com

Java 8

GWT

cassandra

Spark

docker

LesFurets.com
Comparez et achetez futé

DATASTAX

Bigdata@Telcom ParisTech

LesFurets.com

Comparez et achetez futé

Context

LesFurets.com
Comparez et achetez futé

- Master Level course
- Big Data Analysis and Management
    - Non relational Databases
    - 30 hours
        - Lectures: 25 %
        - Hand’s on: 75 %

TELECOM ParisTech

http://www.telecom-paristech.fr/formation-continue/masteres-specialises/big-data.html

Context

DATASTAX

LesFurets.com
Comparez et achetez futé

- 30 students:
  - various backgrounds: students and professionals
  - various skill levels
  - various expertise area: devops, data engineer, marketing
  - SQL and Hadoop exposure

TELECOM ParisTech

Main topics:

mongoDB elasticsearch. cassandra Spark

http://www.telecom-paristech.fr/formation-continue/masteres-specialises/big-data.html

Project goals

- Choose the right technology
- Implement from scratch a full stack solution
- Optimize data model for a particular use case
- Deploy on cloud
- Discuss tradeoffs

One month to deliver

http://www.telecom-paristech.fr/formation-continue/masteres-specialises/big-data.html

DATASTAX

The project, Step by step

LesFurets.com
Comparez et achetez futé

Let’s all become students

Labor day

Discovering the subject

Discovering the subject

An earthquake occurs in the Sea of Japan. A tsunami is likely to hit the coast.

Notify all the population within a 500km range as soon as possible.

Constraints:

- Team work (3 members)
- Choose a technology seen in the module
- Deploy on AWS (300€ budget)
- The closest data center is lost
- Pre-load the data

Discovering Data

How to get the data ?

- Generated logs from telecom providers
- One month of position tracking
- 1 Mb, 1 Gb, 10 Gb, 100 Gb files
- Available on Amazon S3

Data distribution

- 10 biggest cities
- 100 network cells/city
- 1.000.000 different phone numbers

Sapporo
Saitama
Kyoto
Kobe
Tokyo
Fukuoka
Kawasaki
Yokohama
Nagoya
Osaka

Data format

```
2015-01-04 17:10:52,834;Osa_61;34.232793;135.906451;829924
```

Timestamp **Cell ID** **Coordinates** **Phone Number**

Data distribution

Full dataset (100 GB):

- 240 000 logs per hour / city
- ~ 2 000 000 000 rows

Data distribution

```
2015-01-04 17:10:52,834;Osa_61;34.232793;135.906451;829924
2015-01-04 17:10:52,928;Yok_85;35.494120;139.424249;121737
2015-01-04 17:10:52,423;Tok_14;35.683104;139.755020;731737
2015-01-04 17:10:53,923;Osa_61;34.232793;135.906451;861343
2015-01-04 17:10:53,153;Kyo_06;34.980933;135.777283;431737
...
2015-01-04 17:10:55,928;Yok_99;35.030989;140.126021;829924
```

Data distribution

Choosing the stack

Storage Technology short list

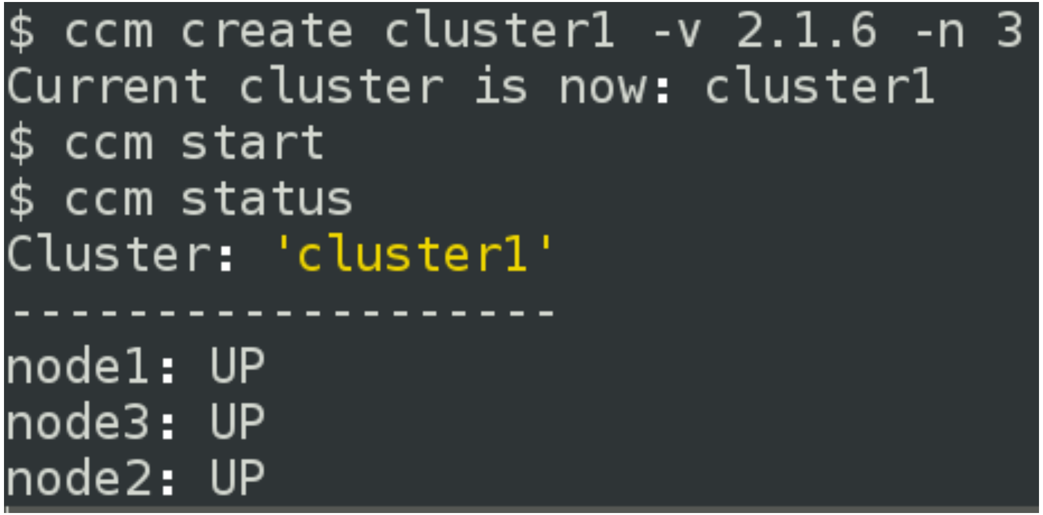
cassandra

- Efficient for logs
- Fault Tolerance

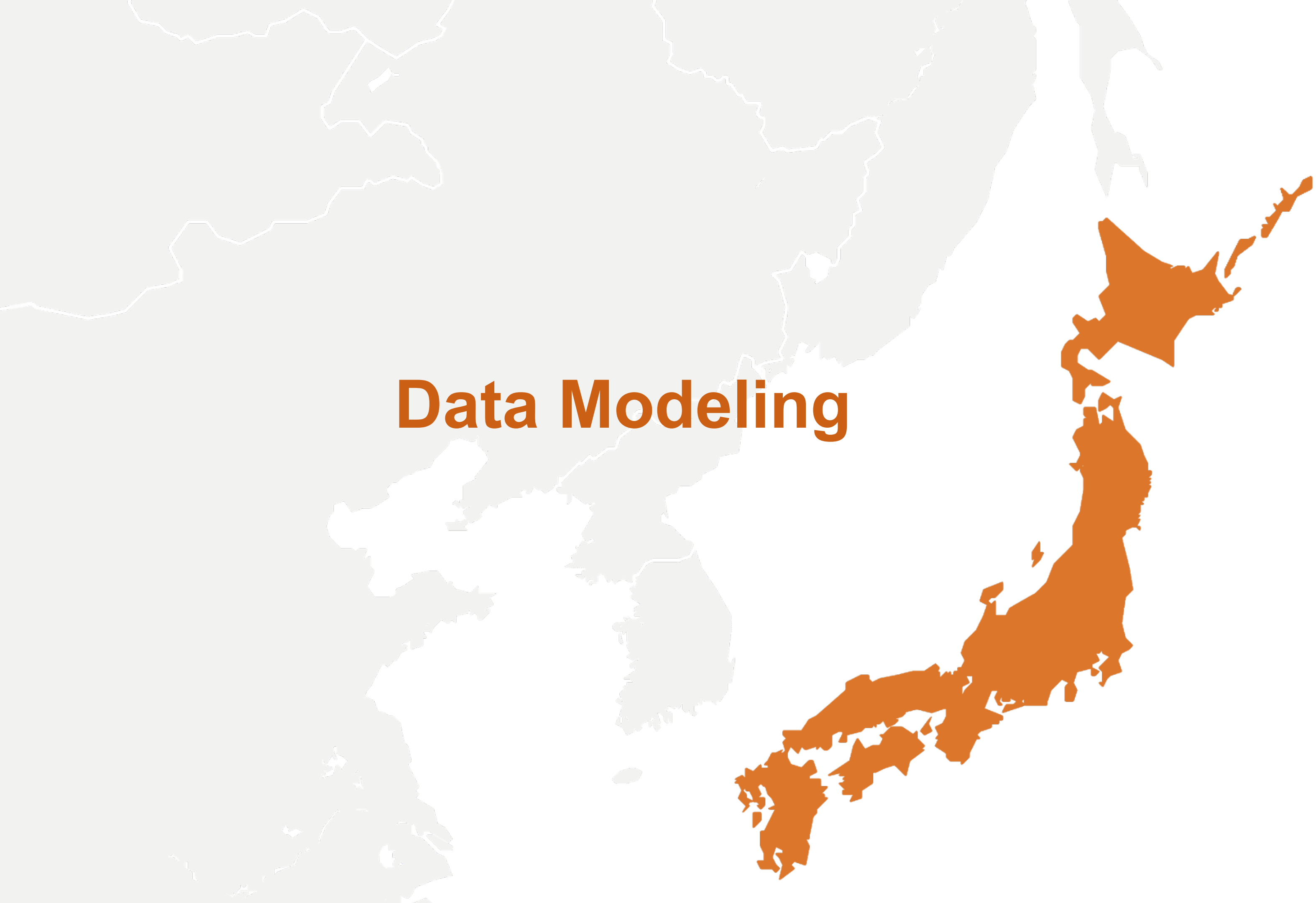
mongoDB

- Geolocation
- Easy transform logs into documents

Install a local Cassandra cluster



```
$ ccm create cluster1 -v 2.1.6 -n 3
Current cluster is now: cluster1
$ ccm start
$ ccm status
Cluster: 'cluster1'
-------------------
node1: UP
node3: UP
node2: UP
```



Data Modeling

Naive model

```
CREATE TABLE phones1 (
     cell text,
     instant timestamp,
     phoneNumber text,
     PRIMARY KEY (cell, instant)
);
```

Osa_19	2015-05-19 15:25:57,369	2015-05-21 15:00:57,551
	456859	012564

Avoid overwrites

```
CREATE TABLE phones2 (
    cell text,
    instant timestamp,
    phoneNumbers Set<text>,
    PRIMARY KEY (cell, instant)
);
```

Osa_19	2015-05-19 15:25:57,369	2015-05-21 15:00:57,551
	456859, 659842	012564

Compound partition key

```
CREATE TABLE phones3 (
     cell text,
     instant timestamp,
     phoneNumber text,
     PRIMARY KEY ((cell, instant), phoneNumber)
);
```

Osa_19 : 2015-05-19 15:25:57,369	456859	659842
	-	-

Hourly bucketing

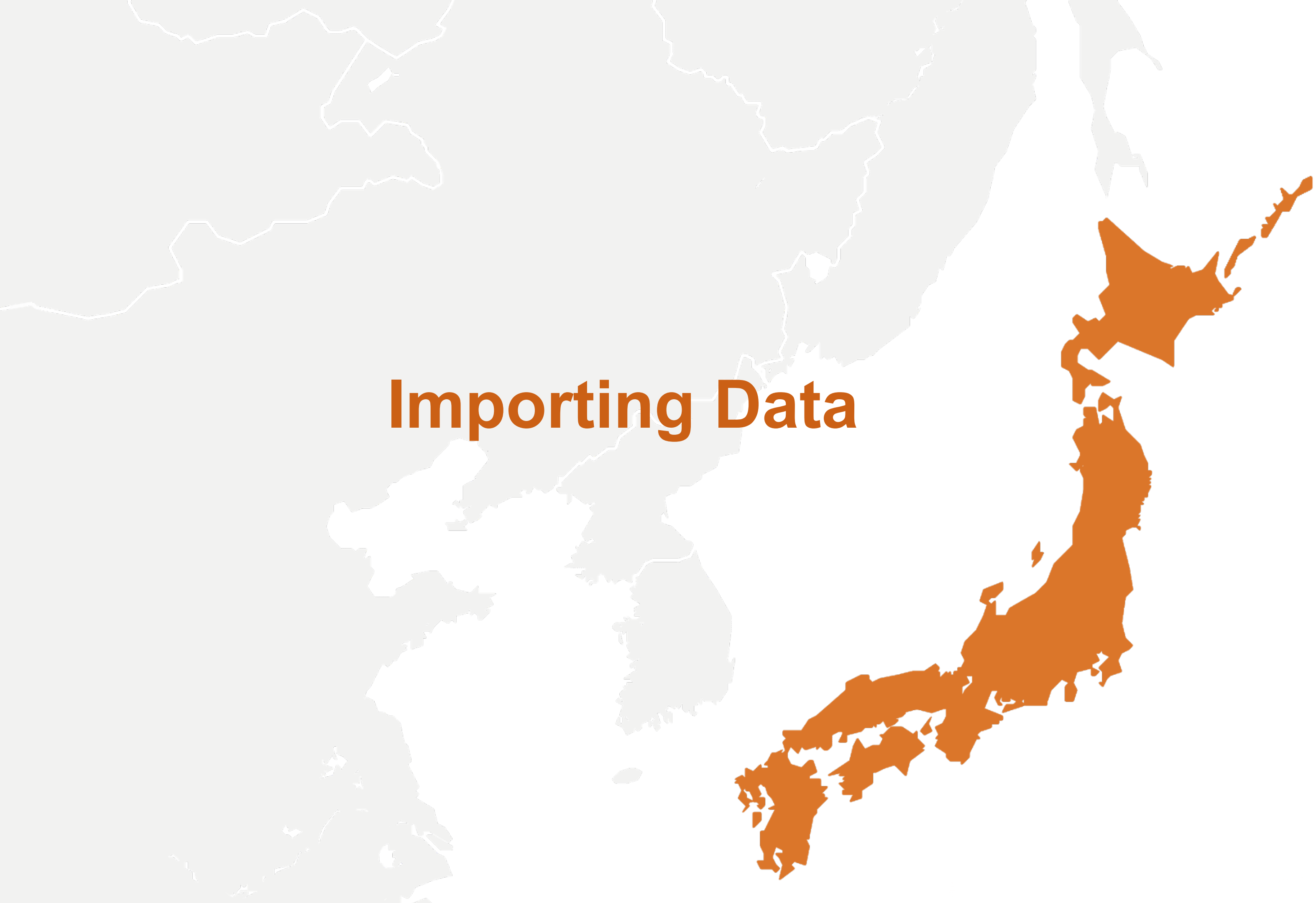
```
CREATE TABLE phones4 (
    cell text,
    instant timestamp,
    phoneNumber text,
    PRIMARY KEY ((cell, instant), phoneNumber)
);
```

Osa_19 : 2015-05-19 15:00	456859	659842
	-	-

Query first

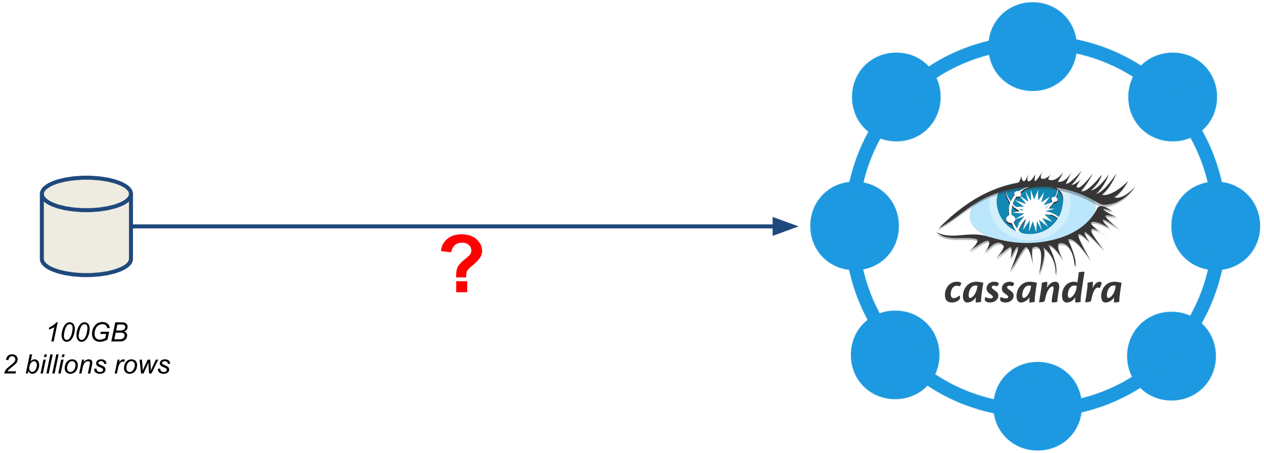
```
CREATE TABLE phones5 (
     cell text,
     instant timestamp,
     numbers text,
     PRIMARY KEY ((cell, instant))
);
```

Osa_19, 2015-05-19 15:00	456859,659842
	-

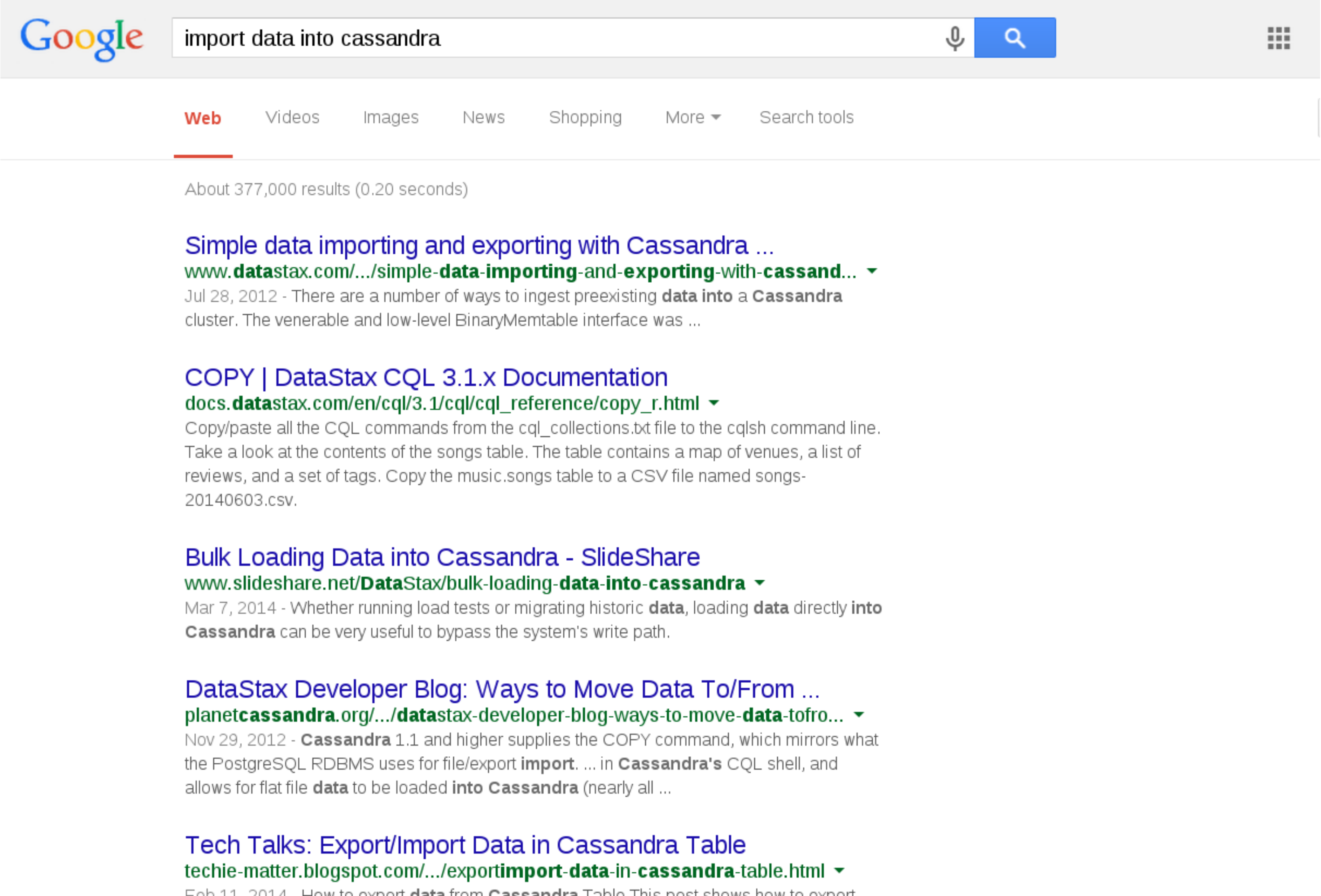


Importing Data

Import data into Cassandra:


100GB
2 billions rows

Import data into Cassandra



About 377,000 results (0.20 seconds)

Simple data importing and exporting with Cassandra ...
www.**datastax**.com/.../simple-**data**-**importing**-and-**exporting**-with-**cassand**...
Jul 28, 2012 - There are a number of ways to ingest preexisting **data into** a **Cassandra** cluster. The venerable and low-level BinaryMemtable interface was ...

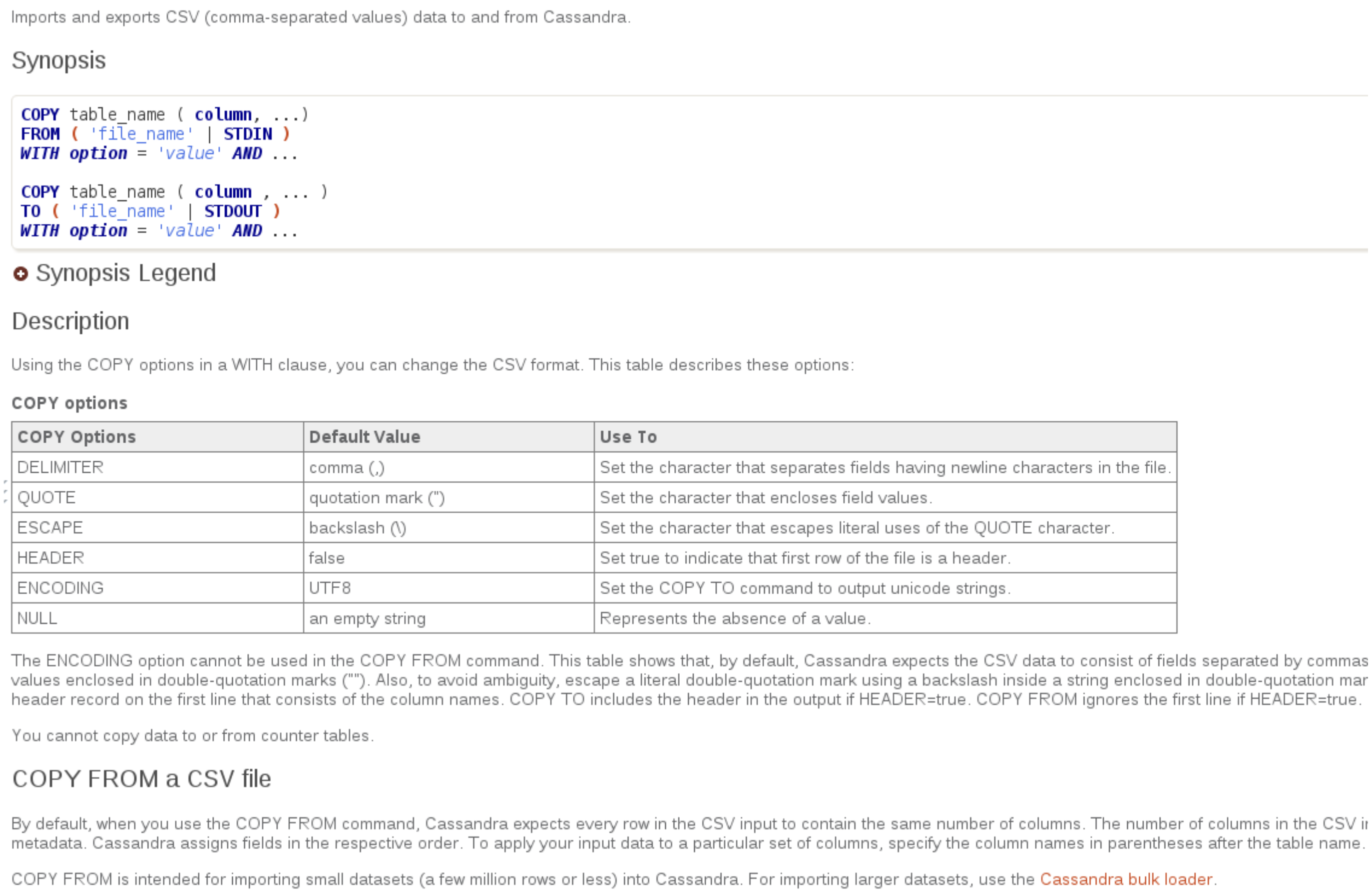
COPY | DataStax CQL 3.1.x Documentation
docs.**datastax**.com/en/cql/3.1/cql/cql_reference/copy_r.html
Copy/paste all the CQL commands from the cql_collections.txt file to the cqlsh command line. Take a look at the contents of the songs table. The table contains a map of venues, a list of reviews, and a set of tags. Copy the music.songs table to a CSV file named songs-20140603.csv.

Bulk Loading Data into Cassandra - SlideShare
www.slideshare.net/**DataStax**/bulk-loading-**data**-**into**-**cassandra**
Mar 7, 2014 - Whether running load tests or migrating historic **data**, loading **data** directly **into** **Cassandra** can be very useful to bypass the system's write path.

DataStax Developer Blog: Ways to Move Data To/From ...
planet**cassandra**.org/.../**datastax**-developer-blog-ways-to-move-**data**-tofro...
Nov 29, 2012 - **Cassandra** 1.1 and higher supplies the COPY command, which mirrors what the PostgreSQL RDBMS uses for file/export **import**. ... in **Cassandra's** CQL shell, and allows for flat file **data** to be loaded **into** **Cassandra** (nearly all ...

Tech Talks: Export/Import Data in Cassandra Table
techie-matter.blogspot.com/.../export**import**-**data**-in-**cassandra**-table.html
Feb 11, 2014 - How to export **data** from **Cassandra** Table This post shows how to export

Import data into Cassandra: Copy



Imports and exports CSV (comma-separated values) data to and from Cassandra.

Synopsis

```
COPY table_name ( column, ...)
FROM ( 'file_name' | STDIN )
WITH option = 'value' AND ...

COPY table_name ( column , ... )
TO ( 'file_name' | STDOUT )
WITH option = 'value' AND ...
```

Synopsis Legend

Description

Using the COPY options in a WITH clause, you can change the CSV format. This table describes these options:

COPY options

COPY Options	Default Value	Use To
DELIMITER	comma (,)	Set the character that separates fields having newline characters in the file.
QUOTE	quotation mark (")	Set the character that encloses field values.
ESCAPE	backslash (\)	Set the character that escapes literal uses of the QUOTE character.
HEADER	false	Set true to indicate that first row of the file is a header.
ENCODING	UTF8	Set the COPY TO command to output unicode strings.
NULL	an empty string	Represents the absence of a value.

The ENCODING option cannot be used in the COPY FROM command. This table shows that, by default, Cassandra expects the CSV data to consist of fields separated by commas values enclosed in double-quotation marks (""). Also, to avoid ambiguity, escape a literal double-quotation mark using a backslash inside a string enclosed in double-quotation mar header record on the first line that consists of the column names. COPY TO includes the header in the output if HEADER=true. COPY FROM ignores the first line if HEADER=true.

You cannot copy data to or from counter tables.

COPY FROM a CSV file

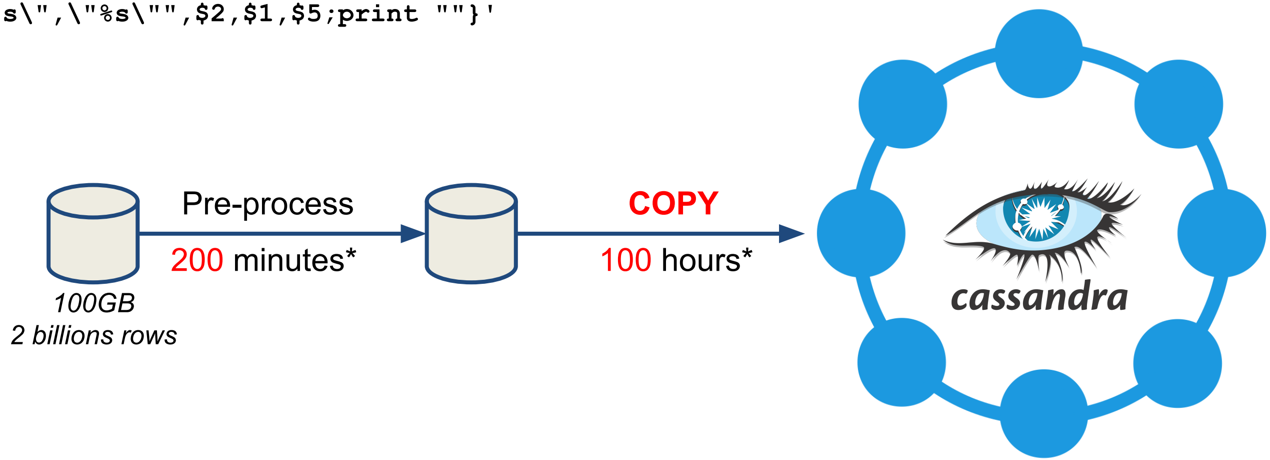
By default, when you use the COPY FROM command, Cassandra expects every row in the CSV input to contain the same number of columns. The number of columns in the CSV i metadata. Cassandra assigns fields in the respective order. To apply your input data to a particular set of columns, specify the column names in parentheses after the table name.

COPY FROM is intended for importing small datasets (a few million rows or less) into Cassandra. For importing larger datasets, use the Cassandra bulk loader.

Import data into Cassandra: Copy

- Pre-processing:
  - project only the needed information
  - cleanup date format (ISO-8601 vs RFC3339)

```
sed 's/,/./g' | awk -F',' '{ printf "\"%s\",\"%
s\",\"%s\"",$2,$1,$5;print ""}'
```



Pre-process

200 minutes*

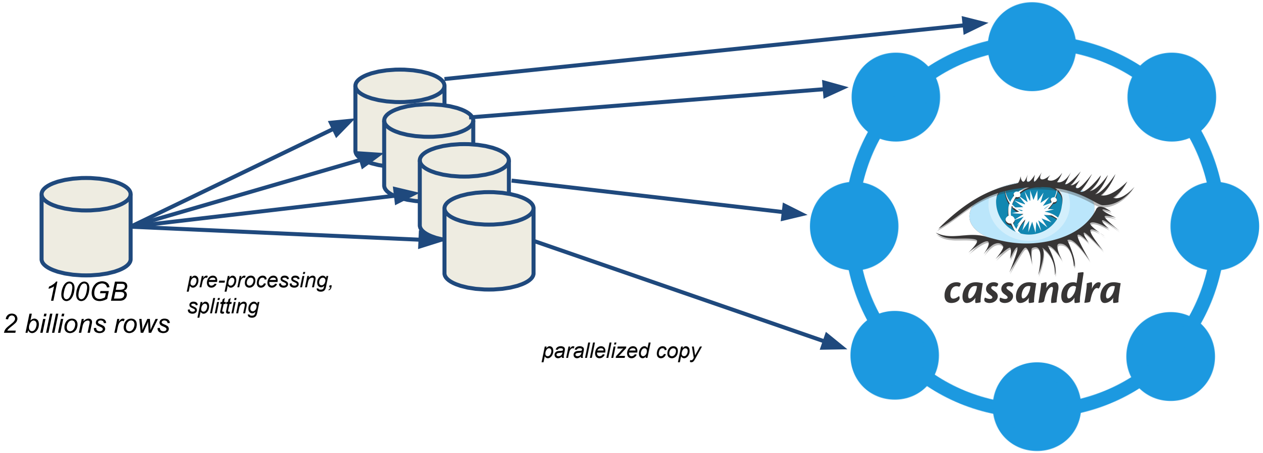
COPY

100 hours*

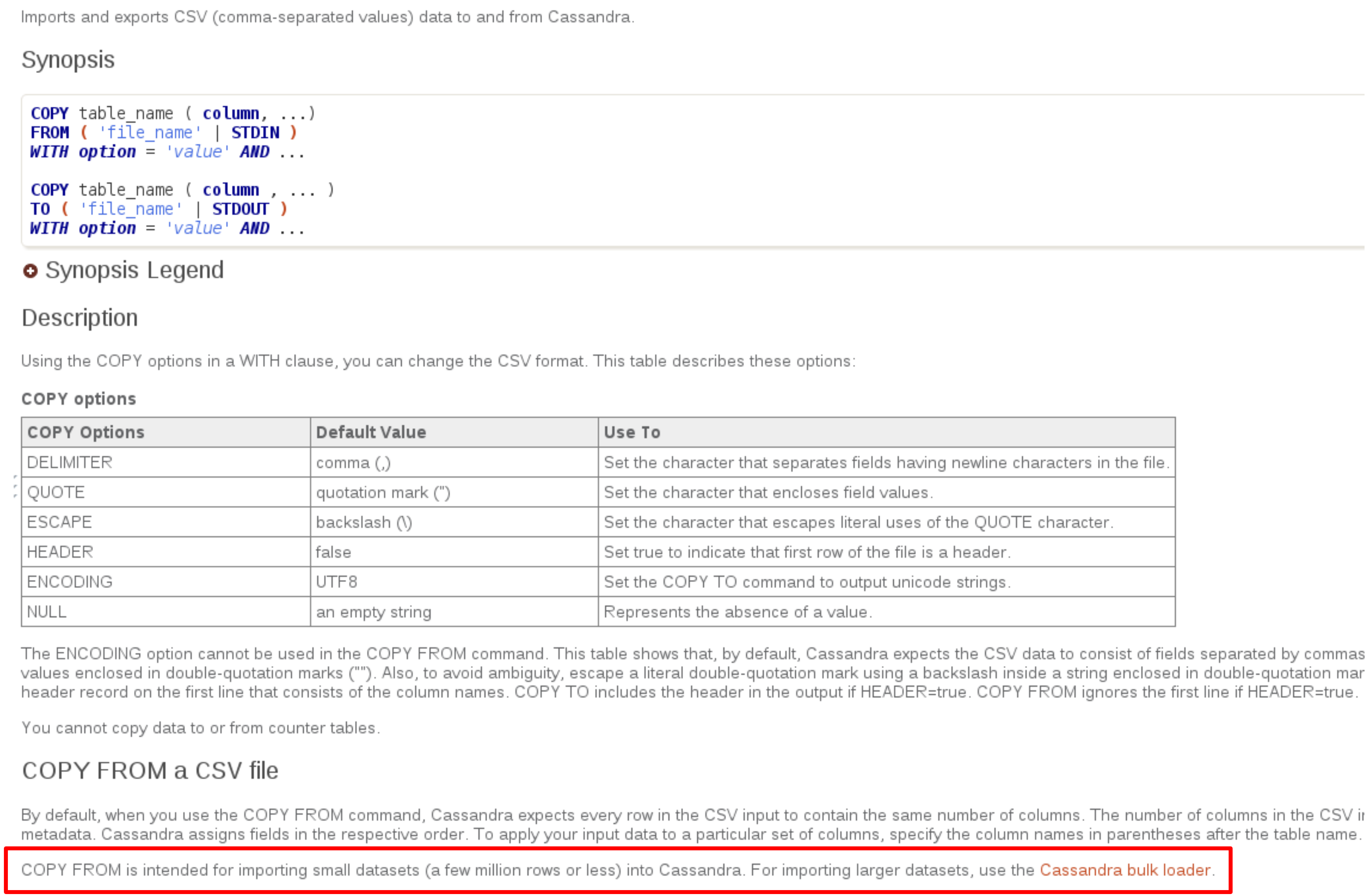
100GB
2 billions rows

Import data into Cassandra: parallelize Copy

- Naive parallelized COPY
  - launch one copy per node
  - launch multiple COPY threads / node



Import data into Cassandra: Copy



Imports and exports CSV (comma-separated values) data to and from Cassandra.

Synopsis

```
COPY table_name ( column, ...)
FROM ( 'file_name' | STDIN )
WITH option = 'value' AND ...

COPY table_name ( column , ... )
TO ( 'file_name' | STDOUT )
WITH option = 'value' AND ...
```

Synopsis Legend

Description

Using the COPY options in a WITH clause, you can change the CSV format. This table describes these options:

COPY options

COPY Options	Default Value	Use To
DELIMITER	comma (,)	Set the character that separates fields having newline characters in the file.
QUOTE	quotation mark (")	Set the character that encloses field values.
ESCAPE	backslash (\)	Set the character that escapes literal uses of the QUOTE character.
HEADER	false	Set true to indicate that first row of the file is a header.
ENCODING	UTF8	Set the COPY TO command to output unicode strings.
NULL	an empty string	Represents the absence of a value.

The ENCODING option cannot be used in the COPY FROM command. This table shows that, by default, Cassandra expects the CSV data to consist of fields separated by commas values enclosed in double-quotation marks (""). Also, to avoid ambiguity, escape a literal double-quotation mark using a backslash inside a string enclosed in double-quotation mar header record on the first line that consists of the column names. COPY TO includes the header in the output if HEADER=true. COPY FROM ignores the first line if HEADER=true.

You cannot copy data to or from counter tables.

COPY FROM a CSV file

By default, when you use the COPY FROM command, Cassandra expects every row in the CSV input to contain the same number of columns. The number of columns in the CSV i metadata. Cassandra assigns fields in the respective order. To apply your input data to a particular set of columns, specify the column names in parentheses after the table name.

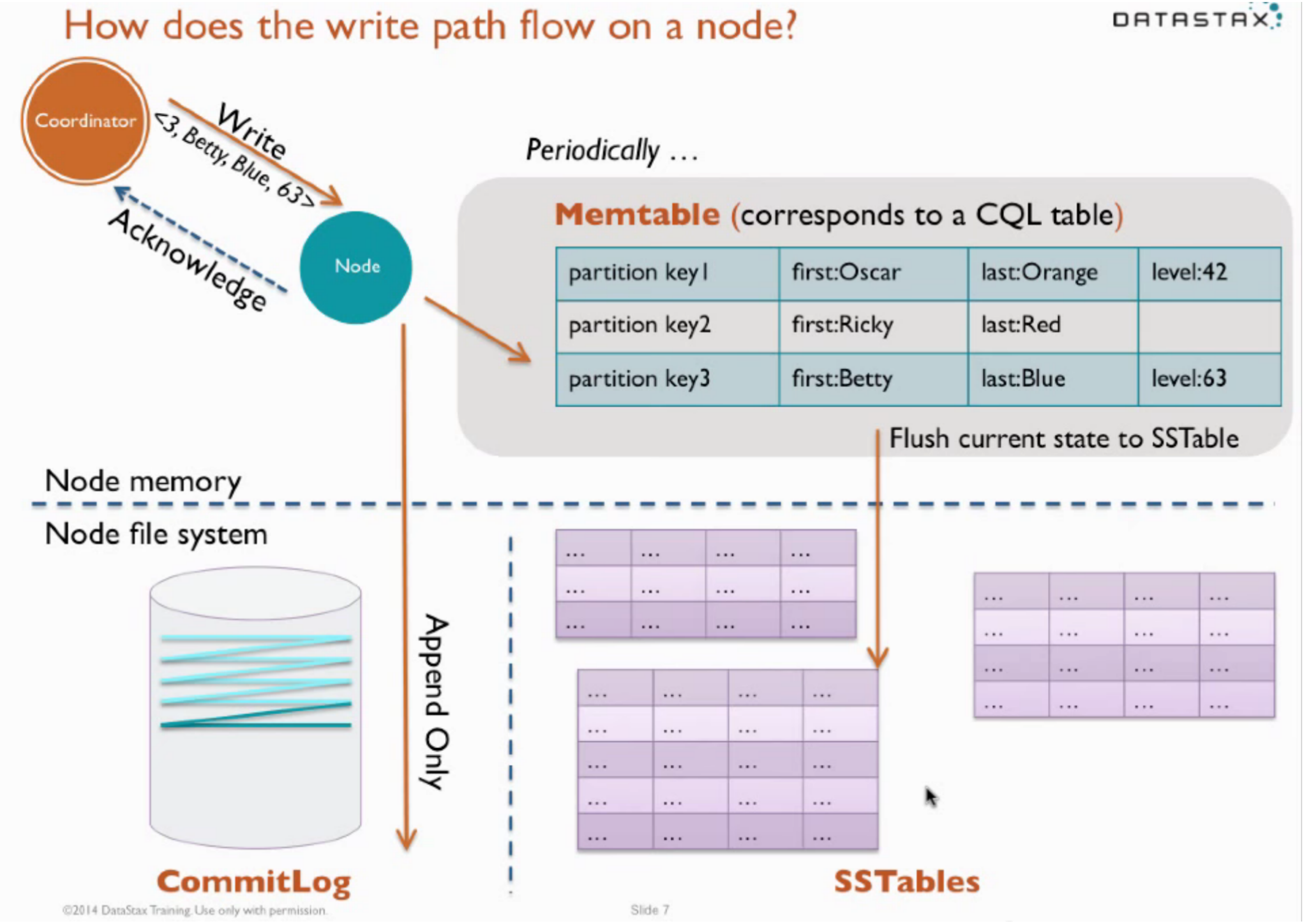
COPY FROM is intended for importing small datasets (a few million rows or less) into Cassandra. For importing larger datasets, use the Cassandra bulk loader.

How to speed up Cassandra import?

- What really happens during the import?
- IMPORT = WRITE
- What really happens during a WRITE

Cassandra Write Path Documentation

DATASTAX



How does the write path flow on a node?

Periodically …

Memtable (corresponds to a CQL table)

partition key1	first:Oscar	last:Orange	level:42
partition key2	first:Ricky	last:Red	
partition key3	first:Betty	last:Blue	level:63

Flush current state to SSTable

CommitLog

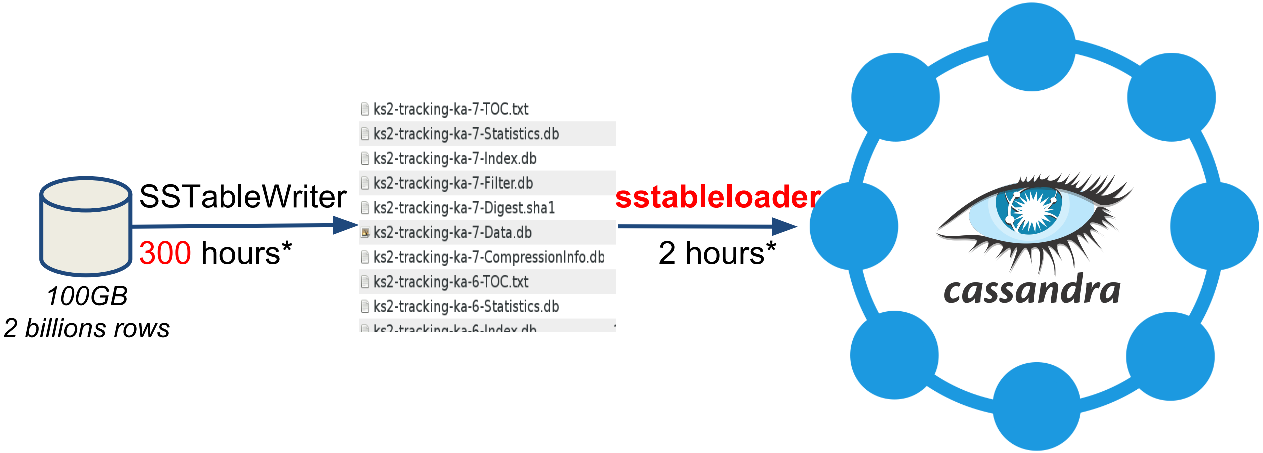
SSTables

Slide 7

LesFurets.com
Comparez et achetez futé

SSTableLoader

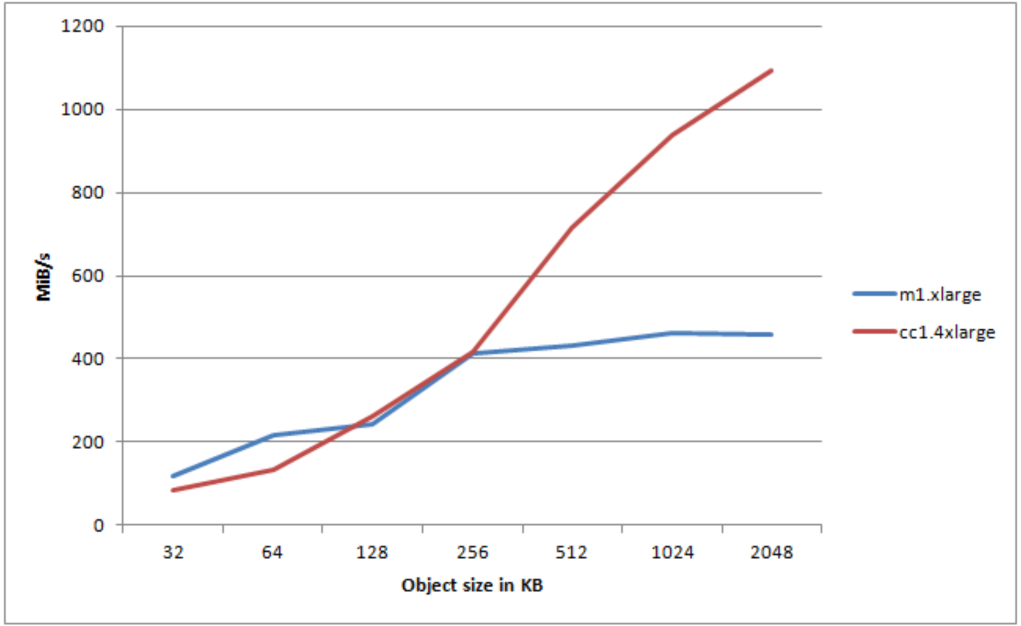
- How to speed up Cassandra import => SSTableLoader
  - generate SSTables using the SSTableWriter API
  - stream the SSTables to your Cassandra cluster using sstableloader



ks2-tracking-ka-7-TOC.txt
ks2-tracking-ka-7-Statistics.db
ks2-tracking-ka-7-Index.db
ks2-tracking-ka-7-Filter.db
ks2-tracking-ka-7-Digest.sha1
ks2-tracking-ka-7-Data.db
ks2-tracking-ka-7-CompressionInfo.db
ks2-tracking-ka-6-TOC.txt
ks2-tracking-ka-6-Statistics.db
ks2-tracking-ka-6-Index.db

Import data into Cassandra from Amazon S3

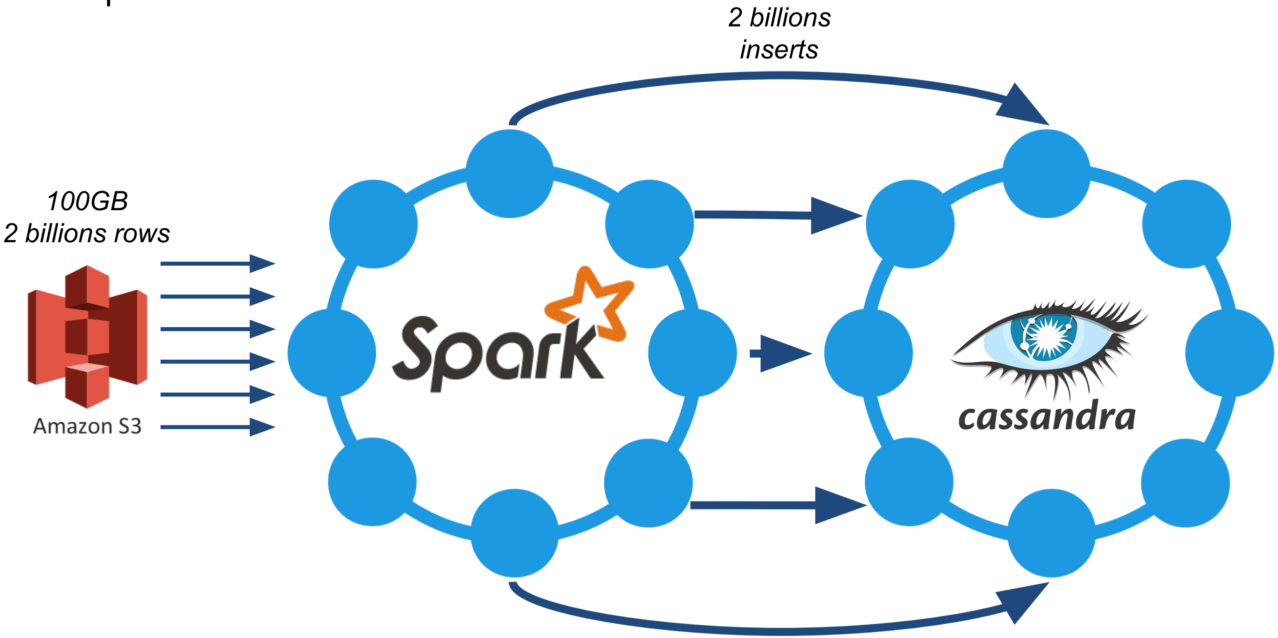
- S3 is distributed!
  - optimized for high-throughput when used in parallel
  - very high throughput to/from EC2



©2011 http://improve.dk/pushing-the-limits-of-amazon-s3-upload-performance/

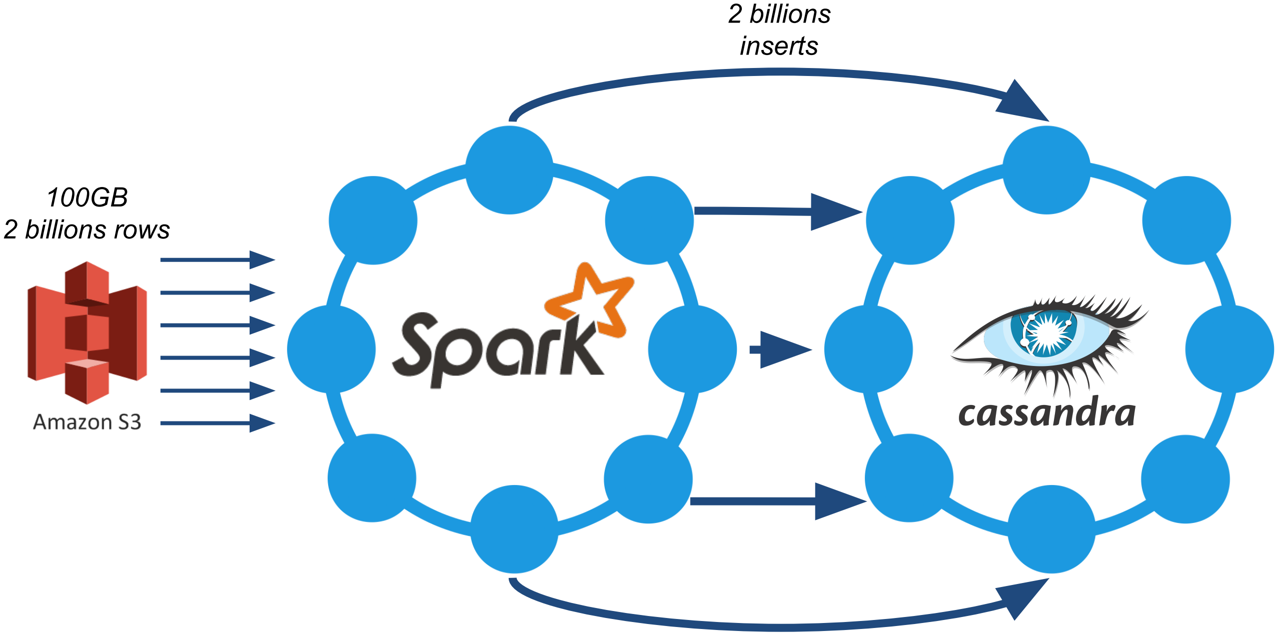
Import data into Cassandra from Amazon S3

- S3 is a distributed file system
  - optimized for high-throughput when used in parallel
- Use Spark as ETL



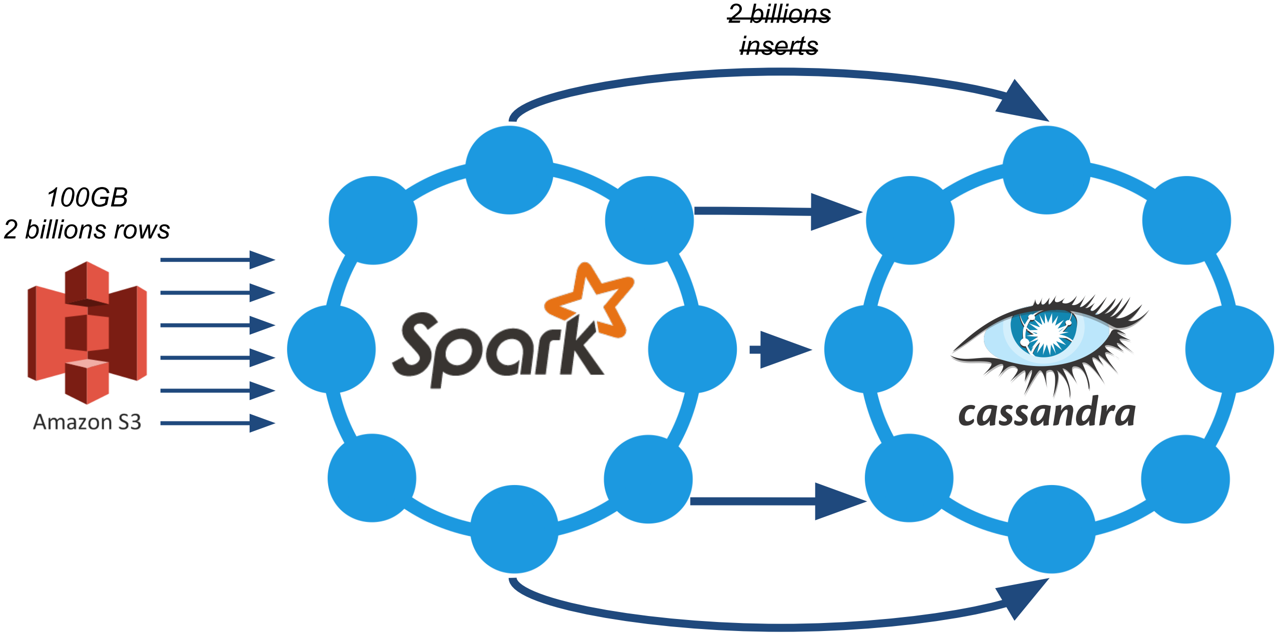
Using Spark to parallelize the insert

- use Spark to read/pre-process/write to Cassandra in parallel
- **map**



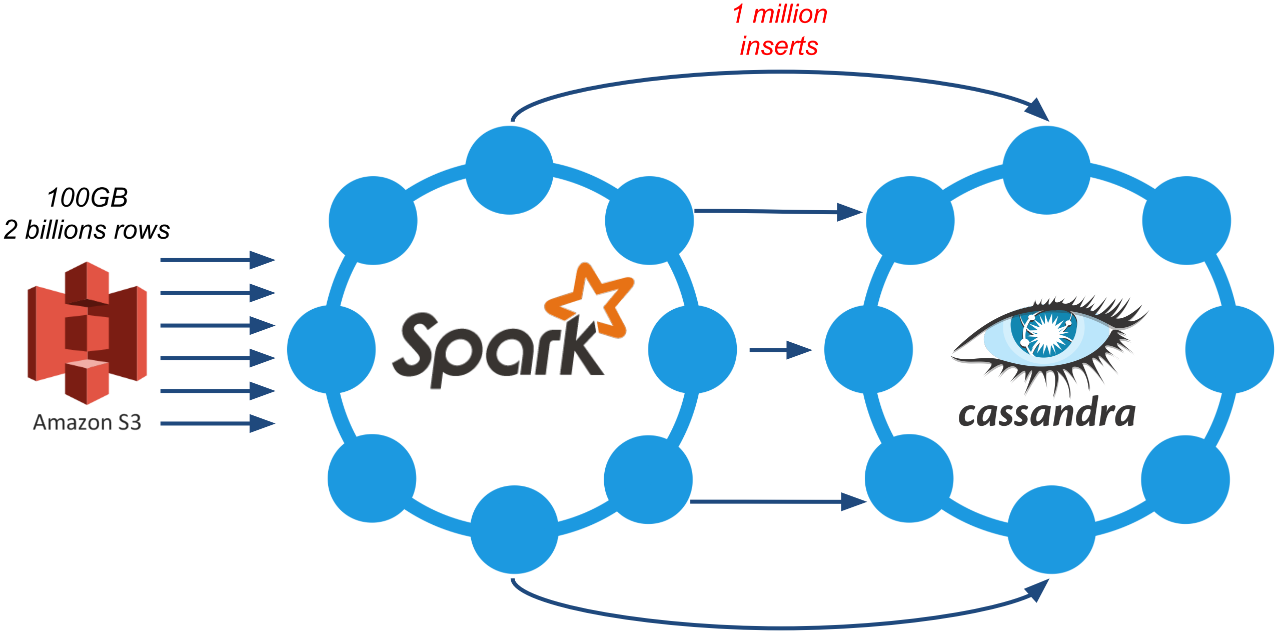
~~Cheating~~ / Modeling before inserting

- instead of importing all the lines from the csv, use Spark to reduce the lines
- **map+reduce**



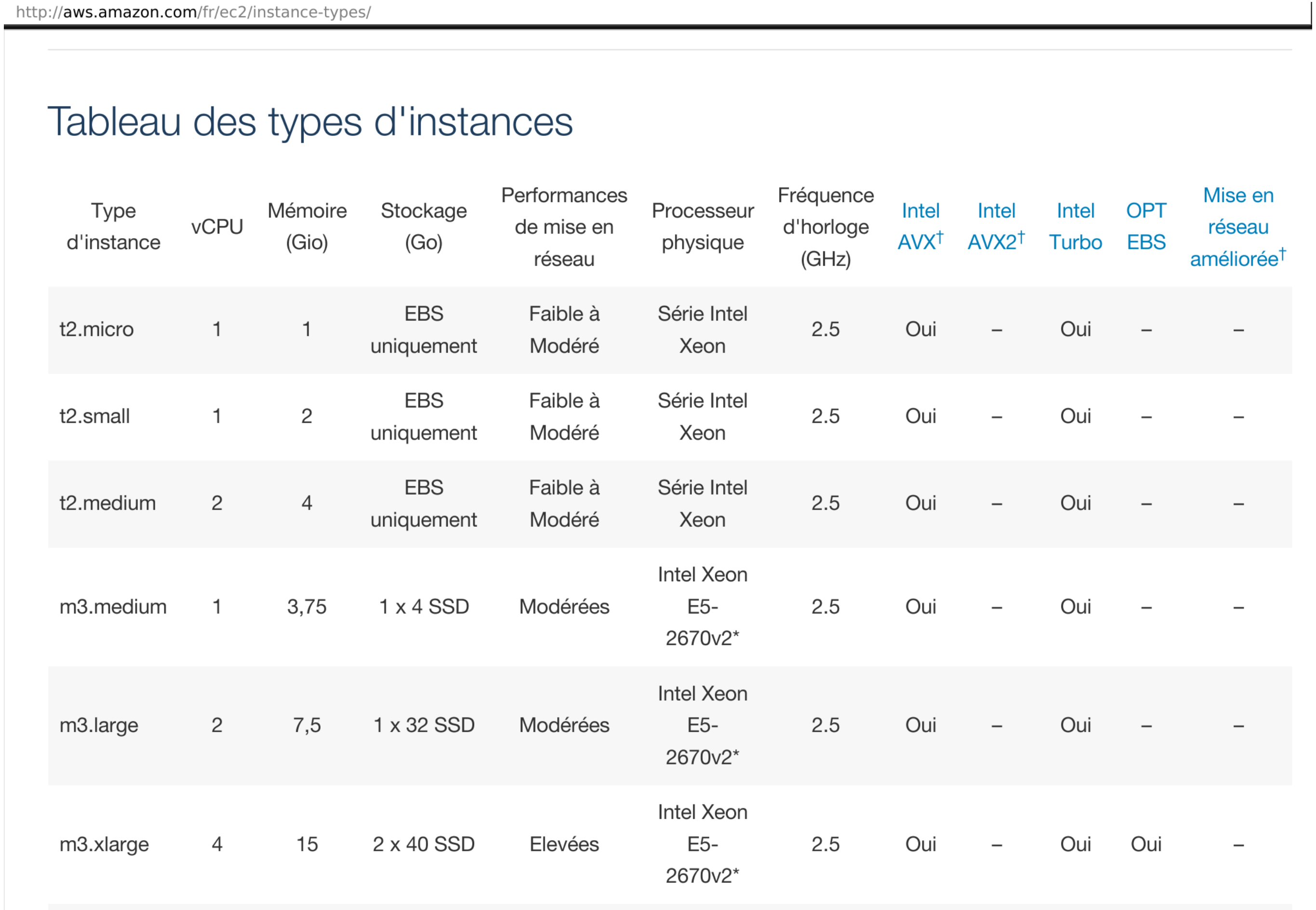
~~Cheating~~ / Modeling before inserting

- instead of importing all the lines from the csv, use Spark to reduce the lines
- **map+reduce**



Using AWS

Choosing the right AWS Instance type



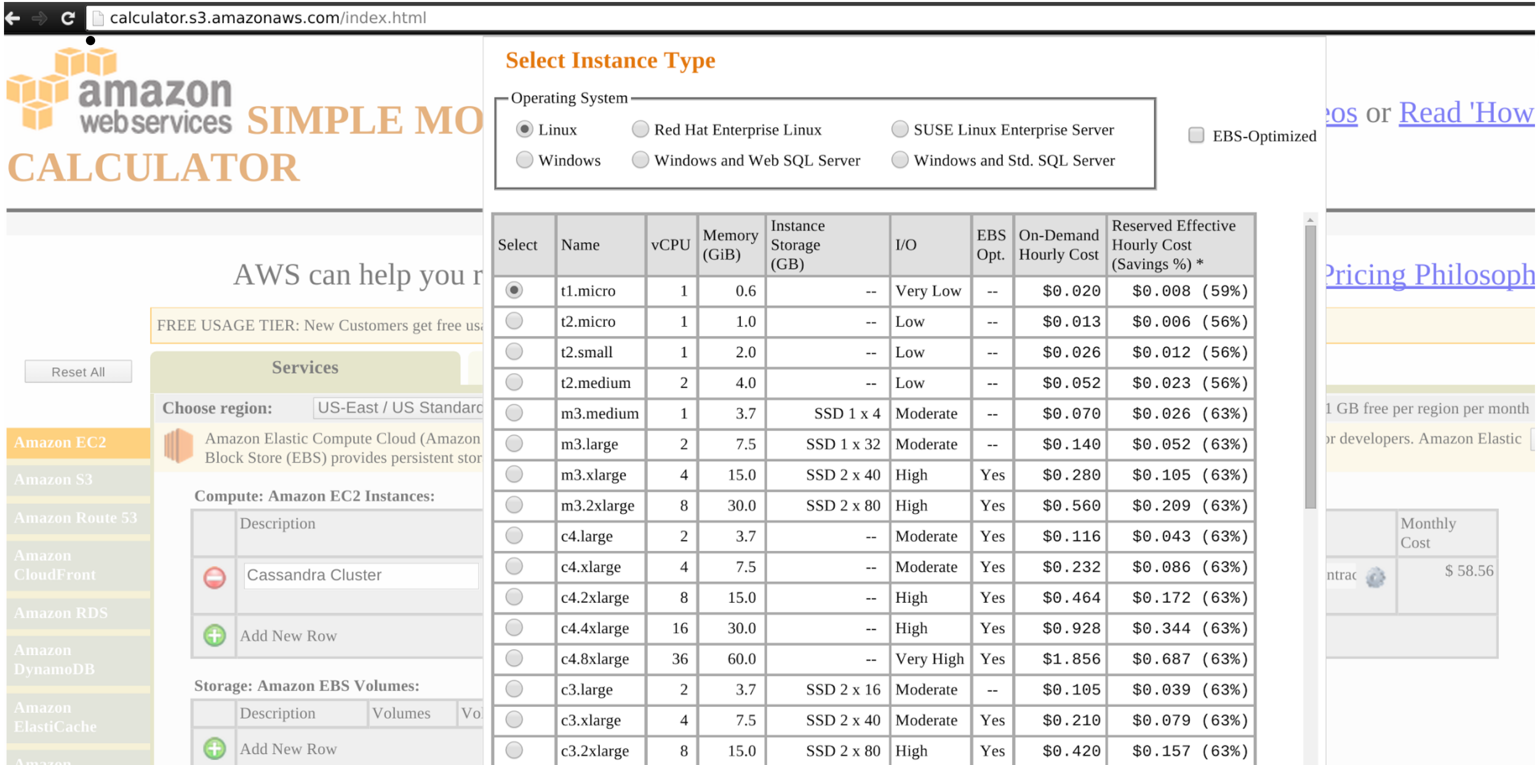
http://aws.amazon.com/fr/ec2/instance-types/

Tableau des types d'instances

Type d'instance	vCPU	Mémoire (Gio)	Stockage (Go)	Performances de mise en réseau	Processeur physique	Fréquence d'horloge (GHz)	Intel AVX†	Intel AVX2†	Intel Turbo	OPT EBS	Mise en réseau améliorée†
t2.micro	1	1	EBS uniquement	Faible à Modéré	Série Intel Xeon	2.5	Oui	–	Oui	–	–
t2.small	1	2	EBS uniquement	Faible à Modéré	Série Intel Xeon	2.5	Oui	–	Oui	–	–
t2.medium	2	4	EBS uniquement	Faible à Modéré	Série Intel Xeon	2.5	Oui	–	Oui	–	–
m3.medium	1	3,75	1 x 4 SSD	Modérées	Intel Xeon E5-2670v2*	2.5	Oui	–	Oui	–	–
m3.large	2	7,5	1 x 32 SSD	Modérées	Intel Xeon E5-2670v2*	2.5	Oui	–	Oui	–	–
m3.xlarge	4	15	2 x 40 SSD	Elevées	Intel Xeon E5-2670v2*	2.5	Oui	–	Oui	Oui	–

Shopping on Amazon

- What can 300E get you: xHours of y Instance Types + Z EBS



calculator.s3.amazonaws.com/index.html

amazon webservices SIMPLE MO
CALCULATOR

AWS can help you r

FREE USAGE TIER: New Customers get free us

Reset All

Services

Choose region: US-East / US Standard

Amazon EC2
Amazon S3
Amazon Route 53
Amazon CloudFront
Amazon RDS
Amazon DynamoDB
Amazon ElastiCache

Amazon Elastic Compute Cloud (Amazon
Block Store (EBS) provides persistent stor

Compute: Amazon EC2 Instances:

Description

Cassandra Cluster

Add New Row

Storage: Amazon EBS Volumes:

Description	Volumes	Vo

Add New Row

eos or Read 'How

Pricing Philosoph

1 GB free per region per month

or developers. Amazon Elastic

Monthly Cost

$ 58.56

Select Instance Type

Operating System

● Linux ○ Red Hat Enterprise Linux ○ SUSE Linux Enterprise Server
○ Windows ○ Windows and Web SQL Server ○ Windows and Std. SQL Server

□ EBS-Optimized

Select	Name	vCPU	Memory (GiB)	Instance Storage (GB)	I/O	EBS Opt.	On-Demand Hourly Cost	Reserved Effective Hourly Cost (Savings %) *
●	t1.micro	1	0.6	--	Very Low	--	$0.020	$0.008 (59%)
○	t2.micro	1	1.0	--	Low	--	$0.013	$0.006 (56%)
○	t2.small	1	2.0	--	Low	--	$0.026	$0.012 (56%)
○	t2.medium	2	4.0	--	Low	--	$0.052	$0.023 (56%)
○	m3.medium	1	3.7	SSD 1 x 4	Moderate	--	$0.070	$0.026 (63%)
○	m3.large	2	7.5	SSD 1 x 32	Moderate	--	$0.140	$0.052 (63%)
○	m3.xlarge	4	15.0	SSD 2 x 40	High	Yes	$0.280	$0.105 (63%)
○	m3.2xlarge	8	30.0	SSD 2 x 80	High	Yes	$0.560	$0.209 (63%)
○	c4.large	2	3.7	--	Moderate	Yes	$0.116	$0.043 (63%)
○	c4.xlarge	4	7.5	--	Moderate	Yes	$0.232	$0.086 (63%)
○	c4.2xlarge	8	15.0	--	High	Yes	$0.464	$0.172 (63%)
○	c4.4xlarge	16	30.0	--	High	Yes	$0.928	$0.344 (63%)
○	c4.8xlarge	36	60.0	--	Very High	Yes	$1.856	$0.687 (63%)
○	c3.large	2	3.7	SSD 2 x 16	Moderate	--	$0.105	$0.039 (63%)
○	c3.xlarge	4	7.5	SSD 2 x 40	Moderate	Yes	$0.210	$0.079 (63%)
○	c3.2xlarge	8	15.0	SSD 2 x 80	High	Yes	$0.420	$0.157 (63%)

Capacity planning

Good starting point: 6 analytics nodes, M3.large or M3.xlarge, SSD

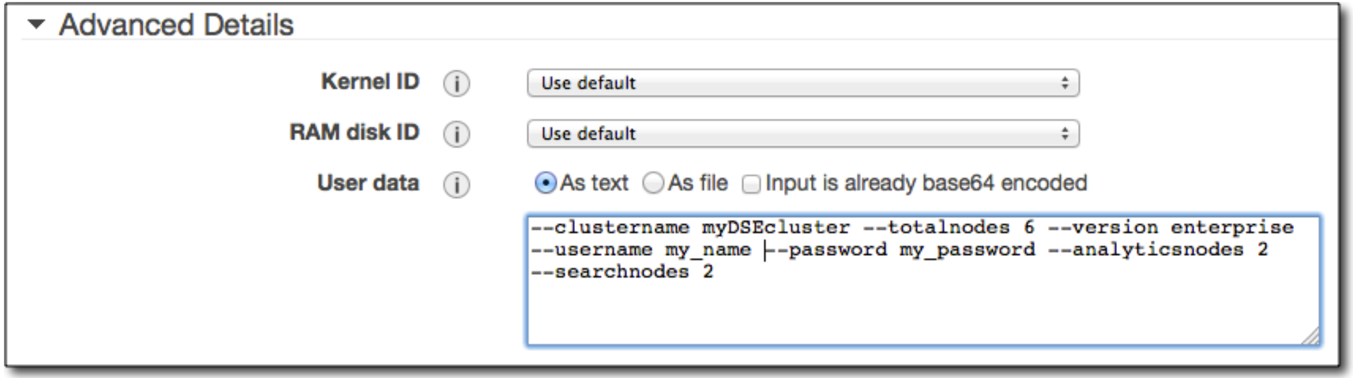
Finally : not so difficult to choose

Install from scratch

- Avantages
  - install an open source stack
  - full control
  - latest version
- Inconvenients
  - devops skills required
  - lot of configuration to do
  - version compatibility
  - time / budget consuming
- NOT AN OPTION !

Start small

- DataStax Enterprise AMI : 1 analytics node shared between Cassandra and Spark



Advanced Details

Kernel ID	Use default
RAM disk ID	Use default
User data	● As text ○ As file □ Input is already base64 encoded

```
--clustername myDSEcluster --totalnodes 6 --version enterprise
--username my_name --password my_password --analyticsnodes 2
--searchnodes 2
```

Optimizing budget

- Requirement:
  - Preload data
- Goal:
  - Minimize cluster uptime
- Strategies:
  - use EBS -> persistent storage (expensive, slow)
  - load the dataset on a node and turn off the others (free, time consuming)
  - do everything “the night before” (free, risky)

Preparing for D-Day

Preparing for the D Day

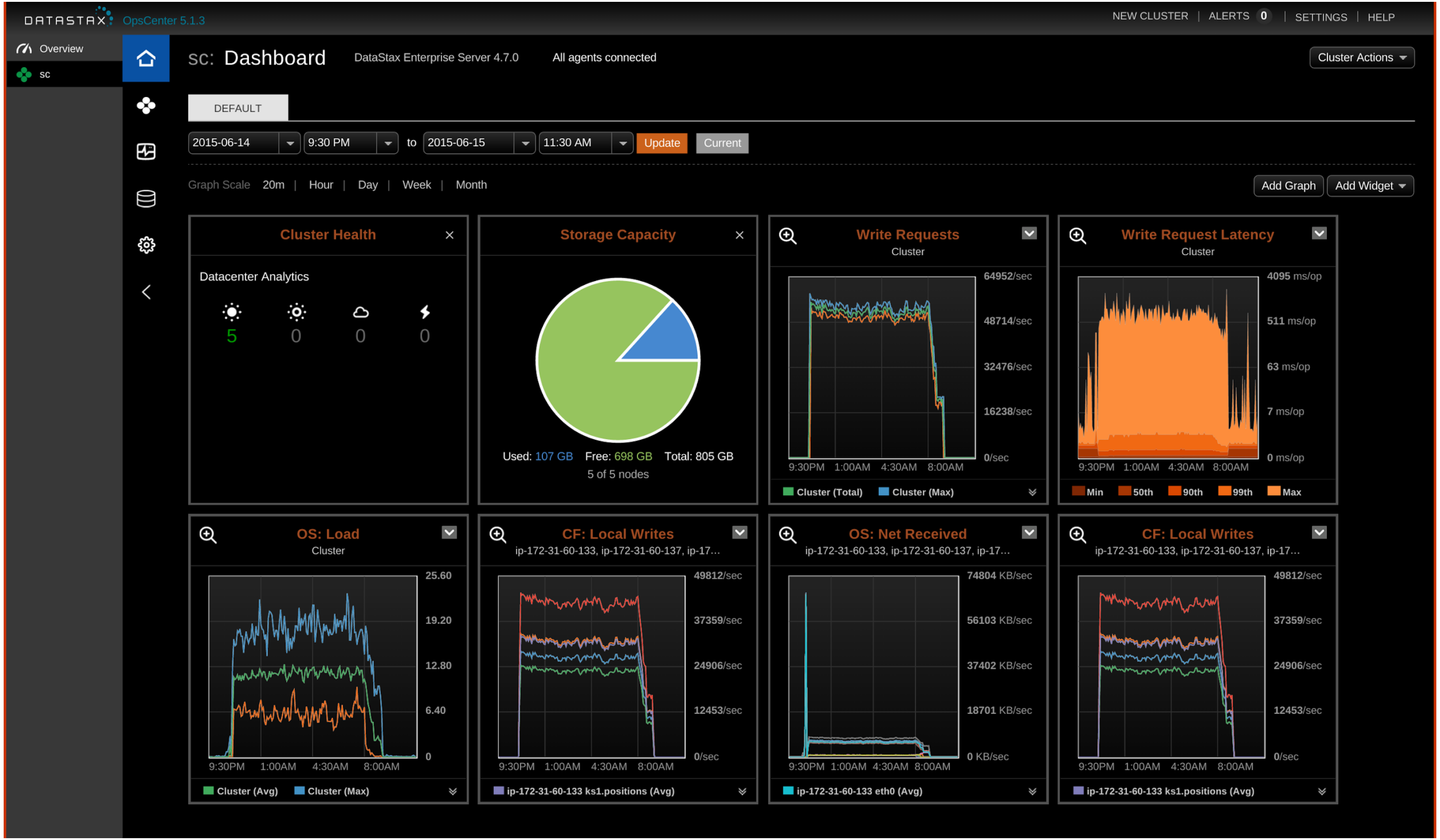
- Preload the data into Cassandra on the AWS cluster
- Input: earthquake date and location
- Kill the closest node
- Start simulation
    - monitor in realtime the alerting performance
- Tuning (model, import)

Lather, rinse, repeat...

How to stop a specific node of the cluster ?

- Alternatives:
  - stop/kill the Cassandra process
  - turn off the gossip
  - AWS instance shutdown
- The effect is the same

Pre-loading data



sc: Dashboard

DataStax Enterprise Server 4.7.0

All agents connected

2015-06-14 9:30 PM to 2015-06-15 11:30 AM

Graph Scale 20m | Hour | Day | Week | Month

Cluster Health

Datacenter Analytics: 5 0 0 0

Storage Capacity

Used: 107 GB Free: 698 GB Total: 805 GB

5 of 5 nodes

Waiting our turn

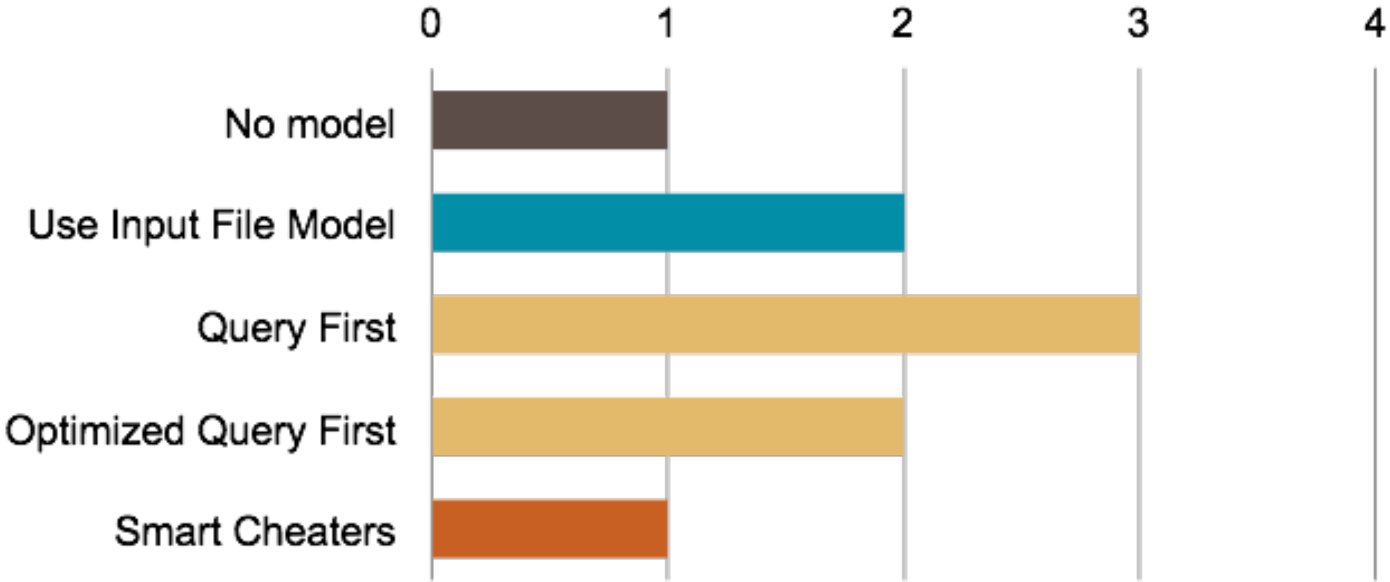
MAY
30

Conclusion

Comparez et achetez futé

BACK
TO THE
TEACHER

Data Model

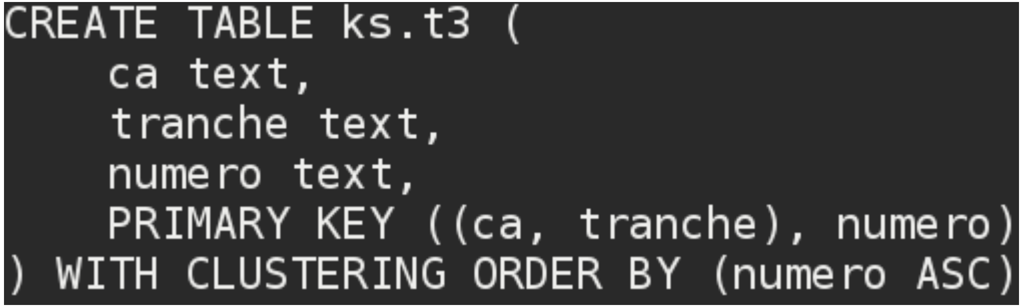

0
1
2
3
4
No model
Use Input File Model
Query First
Optimized Query First
Smart Cheaters

Modelling in SQL vs CQL

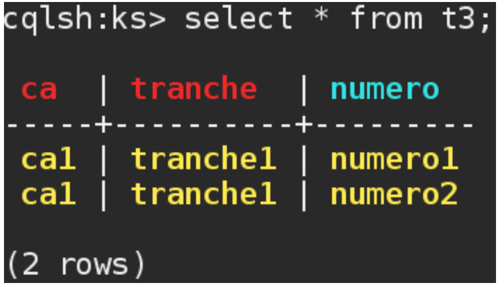
- SQL => CQL : state of mind shifting
  - 3NF vs denormalization
  - model first vs query first

- SQL and CQL are syntactically very similar that can be confusing
  - no Joins, range queries on partition keys...

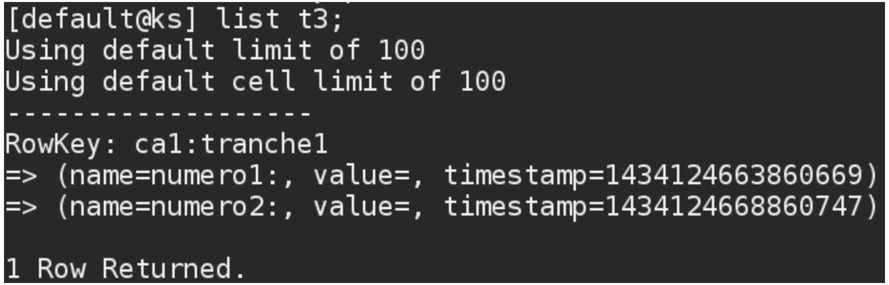
Modeling data: cqlsh vs ~~cli~~



```
CREATE TABLE ks.t3 (
    ca text,
    tranche text,
    numero text,
    PRIMARY KEY ((ca, tranche), numero)
) WITH CLUSTERING ORDER BY (numero ASC)
```

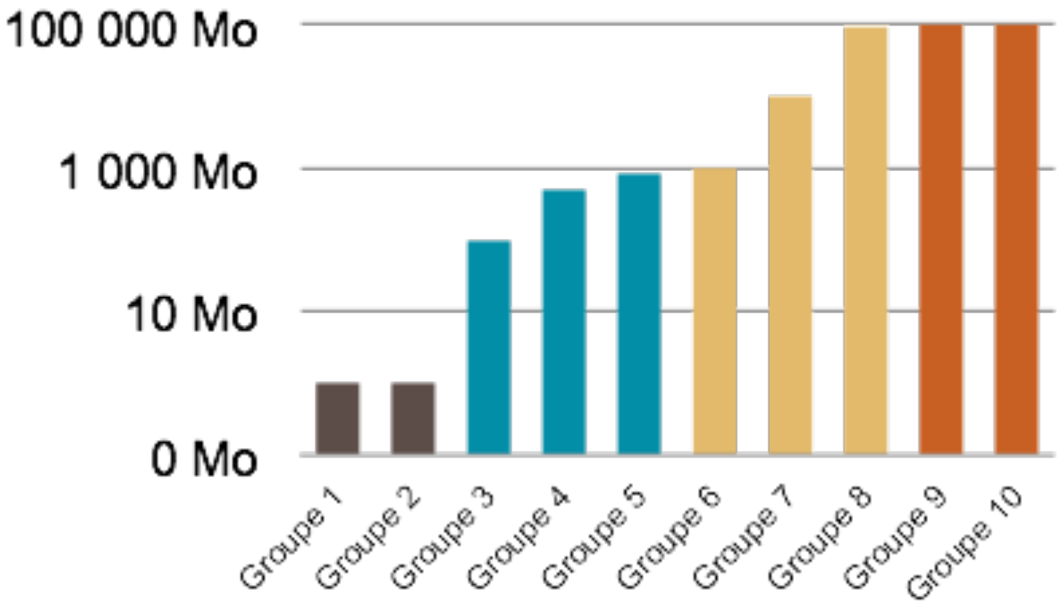


```
cqlsh:ks> select * from t3;

 ca  | tranche  | numero
-----+----------+---------
 ca1 | tranche1 | numero1
 ca1 | tranche1 | numero2

(2 rows)
```

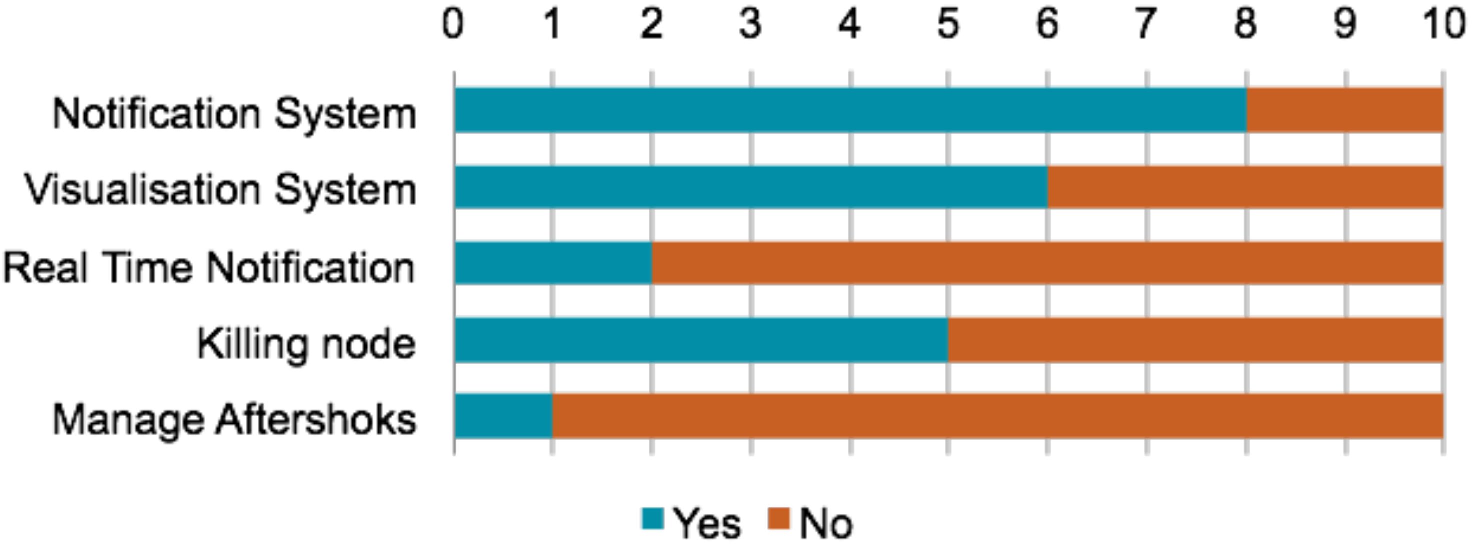


```
[default@ks] list t3;
Using default limit of 100
Using default cell limit of 100
-------------------
RowKey: ca1:tranche1
=> (name=numero1:, value=, timestamp=1434124663860669)
=> (name=numero2:, value=, timestamp=1434124668860747)

1 Row Returned.
```

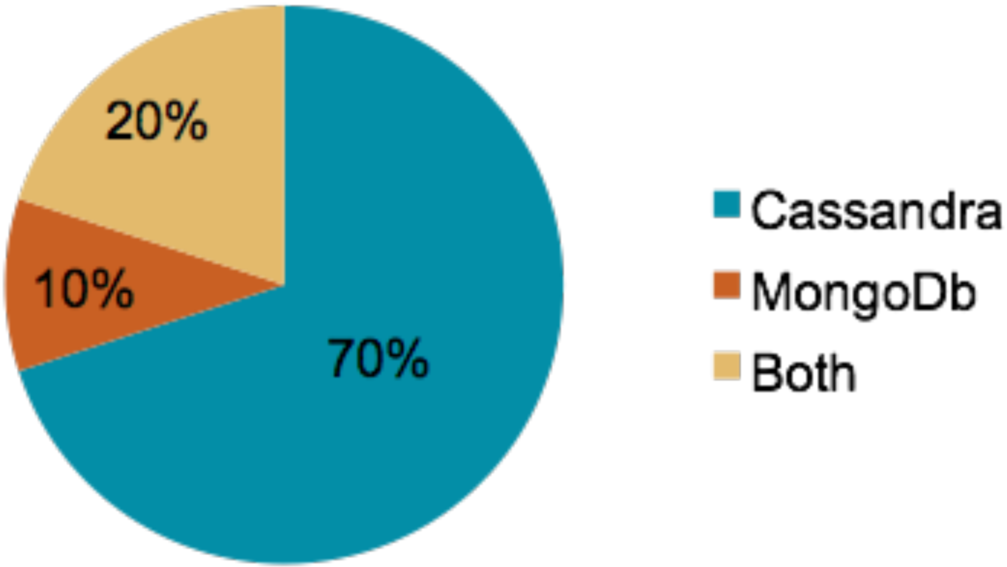
Importing Data



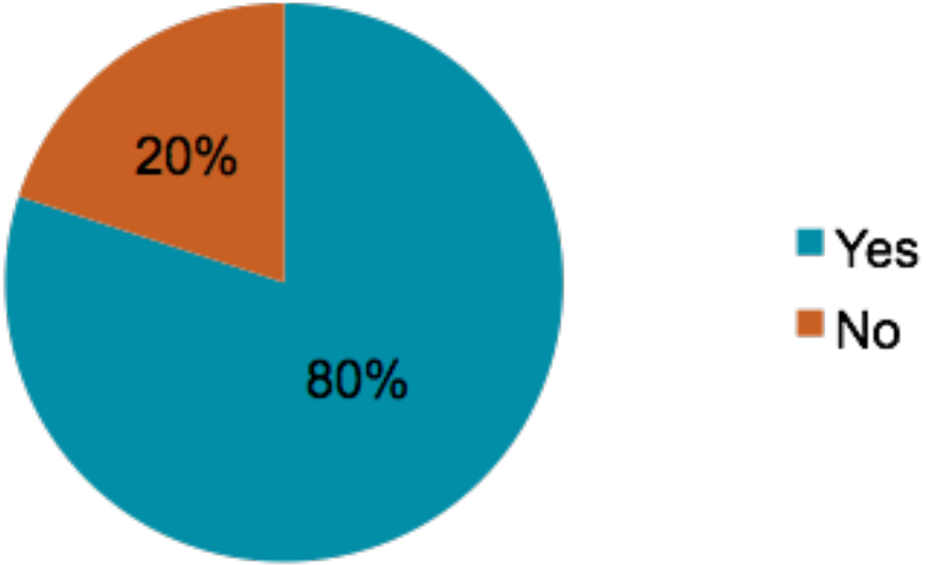
Development



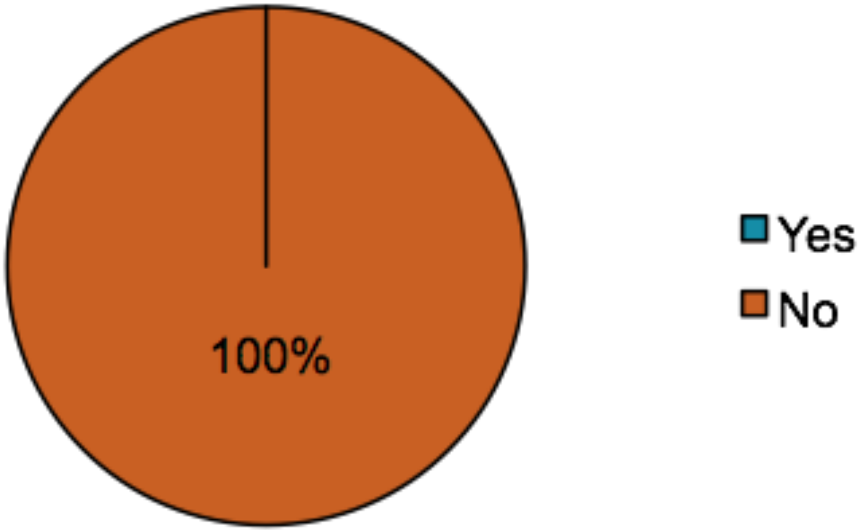
Architecture



Amazon Web Services



Quality



What is the main problem ?

Modeling Data

Importing Data

Architecture

Development

Deployment

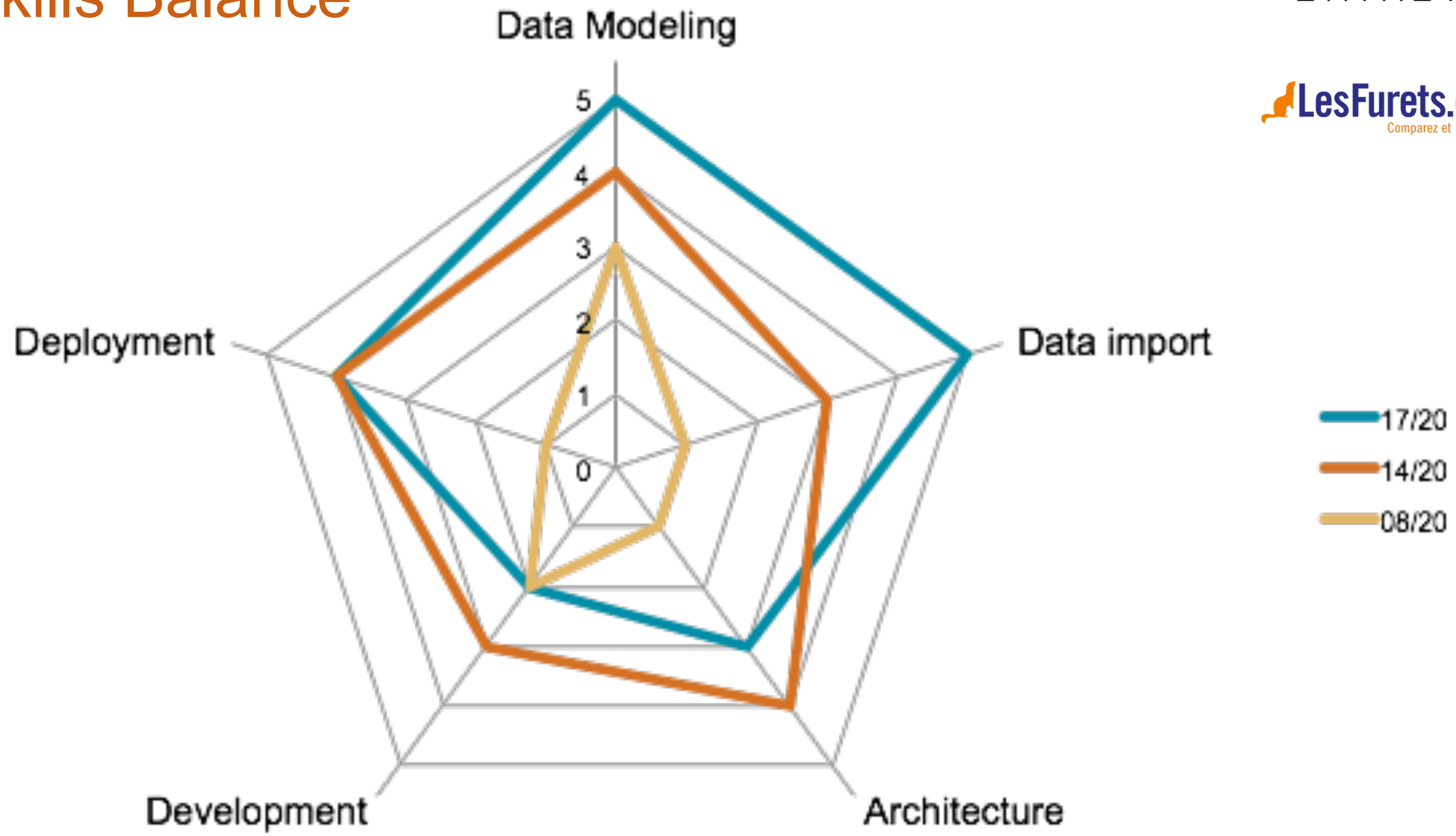
Quality

=> Multidisciplinary Approach

Multidisciplinary approach

Team	1	2	3	4	5	6	7	8	9	10
Modeling Data	√	√		√		√	√	√	√	√
Importing Data								√	√	√
Architecture					√	√	√	√	√	√
Development			√		√		√	√	√	√
Deployment			√	√	√	√	√	√	√	√
Quality			√		√	√				√
Grade	8	10	12	13	13	14	14.5	16	16.5	17

Team Skills Balance



Conclusion

- Easy to learn and easy to teach
- Easy to deploy on AWS
- Performance monitor for tuning the models

- Passing from SQL vs CQL can be challenging
- Fun to implement on a concrete use-case with time/budget constraints

Thank you

Andrei Arion - aar@lesfurets.com
Geoffrey Bérard - gbe@lesfurets.com
Charles Herriau - che@lesfurets.com
@BeastieFurets