Linux Málaga

@linux_malaga www.linux-malaga.org

Centrologic

Big Data

cassandra

Bienvenido - Welcome - Witam

Juan Miguel Taboada Godoy

@centrologic_es

http://linkedin.com/user/centrologic

Juan José Soler Ruiz

@soleronline

http://es.linkedin.com/in/soleronline

Bio

Juan Miguel Taboada Godoy (1980 - …)

1996 – Primer ordenador y primera LAN (cable coaxial)

1999 – Universidad de Málaga y **Linux Málaga**
2001 – Grupo de investigación **GEB**.uma.es (4 años)
Cluster computación +20 nodos (**OpenMosix**)
2002 – Presidente de Asociación **Málaga Wireless**
2003 – Beca en **Neurociencia** en SUNY
Teleruta (Ministerio de Fomento – 2 años)

2005 – Autónomo:
- Nace **Centrologic**
- Polonia (2 años)
- Likindoy (Axaragua + **Junta Andalucía**)

2008 – Responsable Sistemas en PontGrup
2011 – Adquisición Datos en Bética Fotovoltáicas
2012 – **SAFECLON** y SCRUM/KANBAN

2013 – **Executive MBA**

2014 – **Aeronáutica: Django + AngularJS**
2015 – **Industria: Likindoy + Big Data**

Juan José Soler Ruiz

2001-2003 – CFGS Administración Sistemas Informáticos

2003 – Primer premio en el concurso “Javier Benjumea”
2003 – Montaje y configuración
de “Cluster Heterogéneo De Computadoras”
bajo SO Red Hat 7.2.

2005-2012 – STEA Telemática
2007-2009 – Primer CRM en PHP

2010-2011 – Administrador de BBDD / Programador
Web en Bética Fotovoltáicas

2010-2012 – Opositometro

2012-.... – **Centrologic**

2013 – Dailymarkets
2013-2014 – CRM en Python/Django

2014-... – **Centrologic**

¿Por qué?

Likindoy

The monitoring company

Software de adquisición masiva de datos:

- 1 dato (Fecha+Valor) cada minuto por se al
- 1000 se ales por dispositivo
- 20 dispositivos por nodo
- 40 nodos por cliente

800.000 registros por minuto

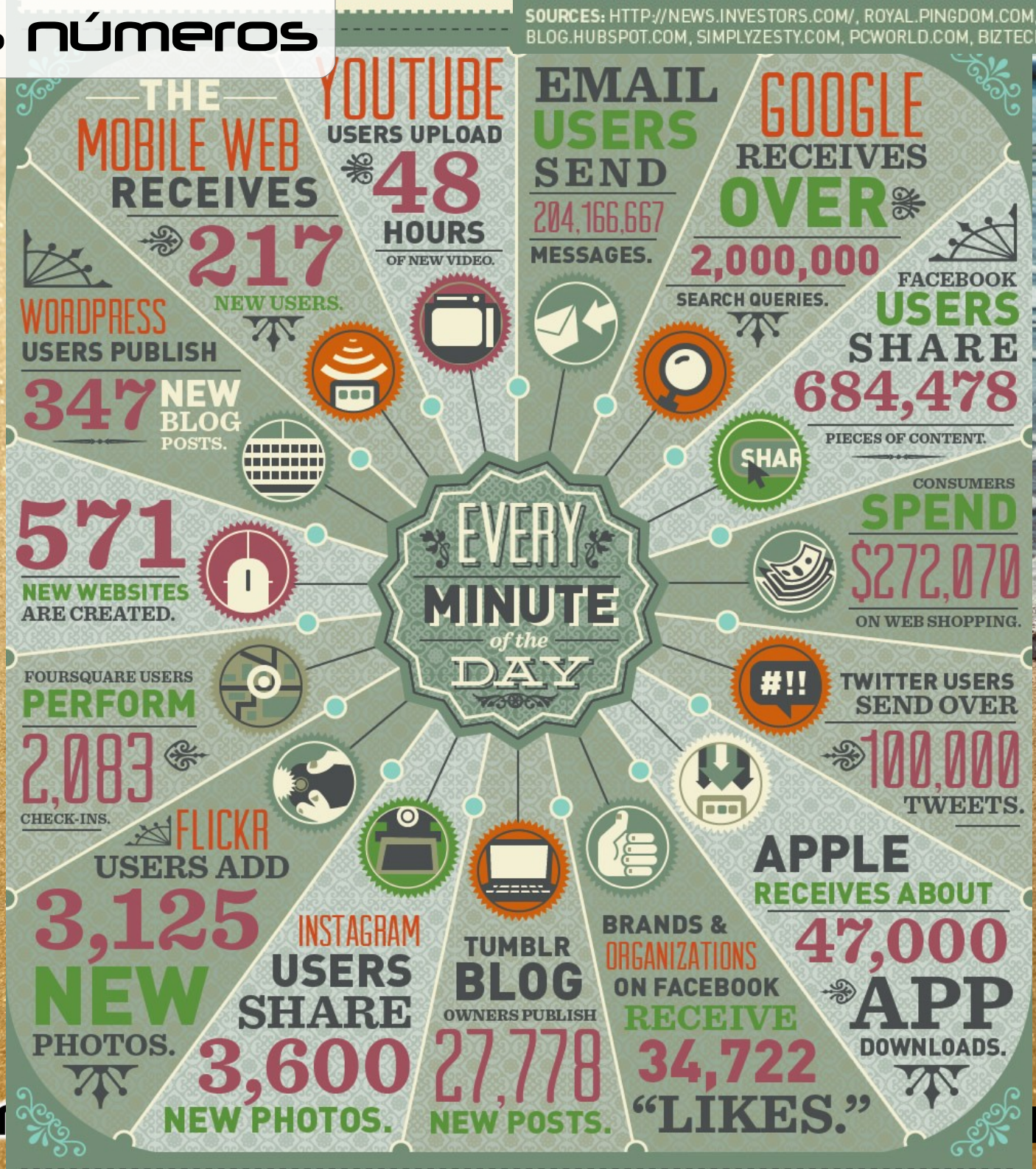
(8.000.000 tomas por minuto)

- 48M por hora
- 1.152M por día

0,4 Billones por a o y cliente

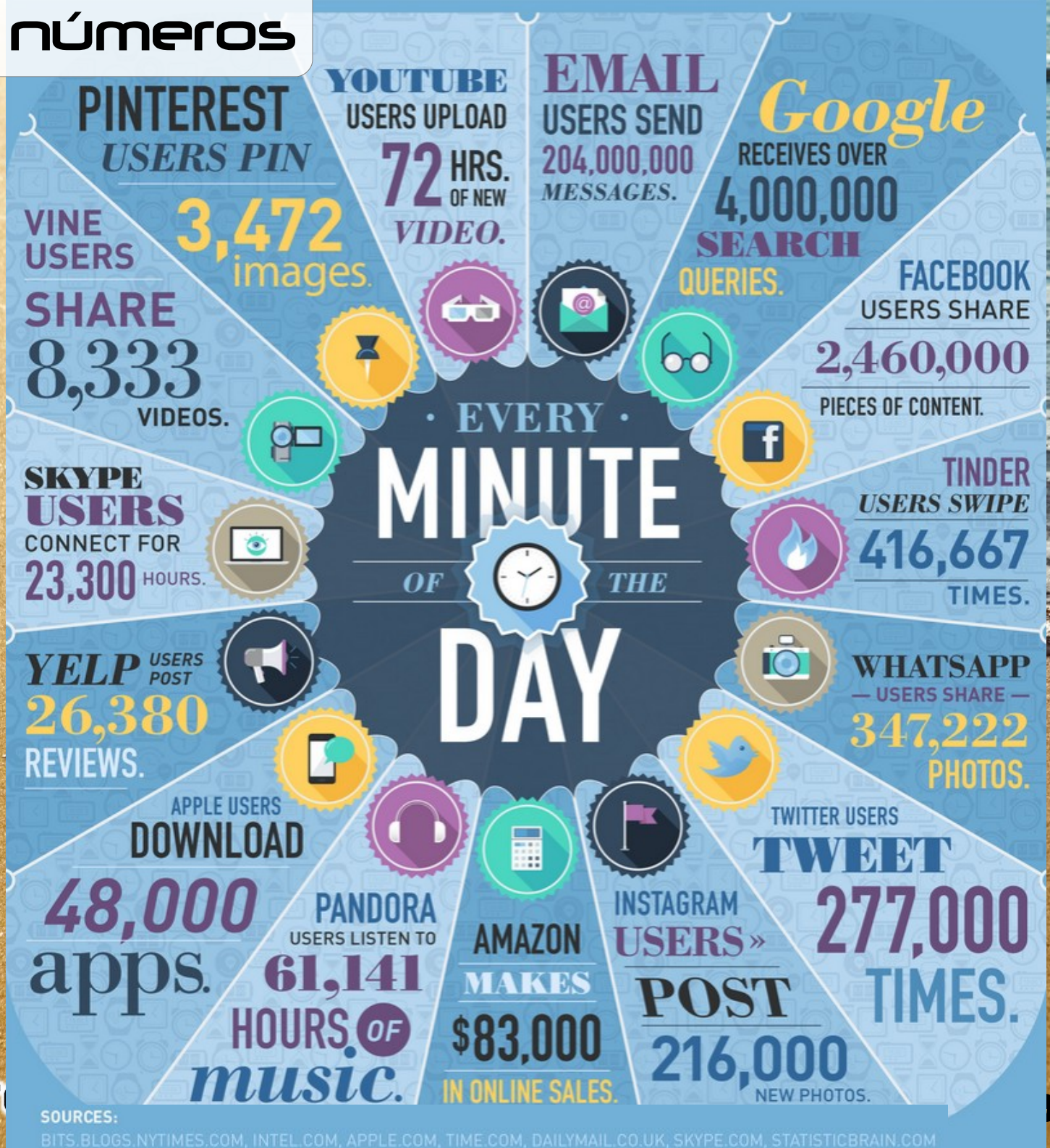
Grandes números

2012



Grandes números

2014



- La gestión de datos crece de momento EXPONENCIALMENTE

- SEAGATE anuncia que el año 2016 será el año del Zettabyte

- 1 ZB equivale al espacio que ocupa 2 billones de años de música

- 1 ZB = 1024 exabytes = casi 1.1 trillones de Gbytes

- 1 zB = 1 sixtillón de bytes

- Ahora estamos en la época del **quintillón** de bytes...

¿Cuanto es un quintillón?

- Una única **gota de agua** contiene:
  1.7 quintillones de moléculas de agua.

- La distancia de la **Via Láctea hasta Andrómeda** es de:
  2 millones de años luz
  18,87 quintillones de kilómetros
  11,73 quintillon miles

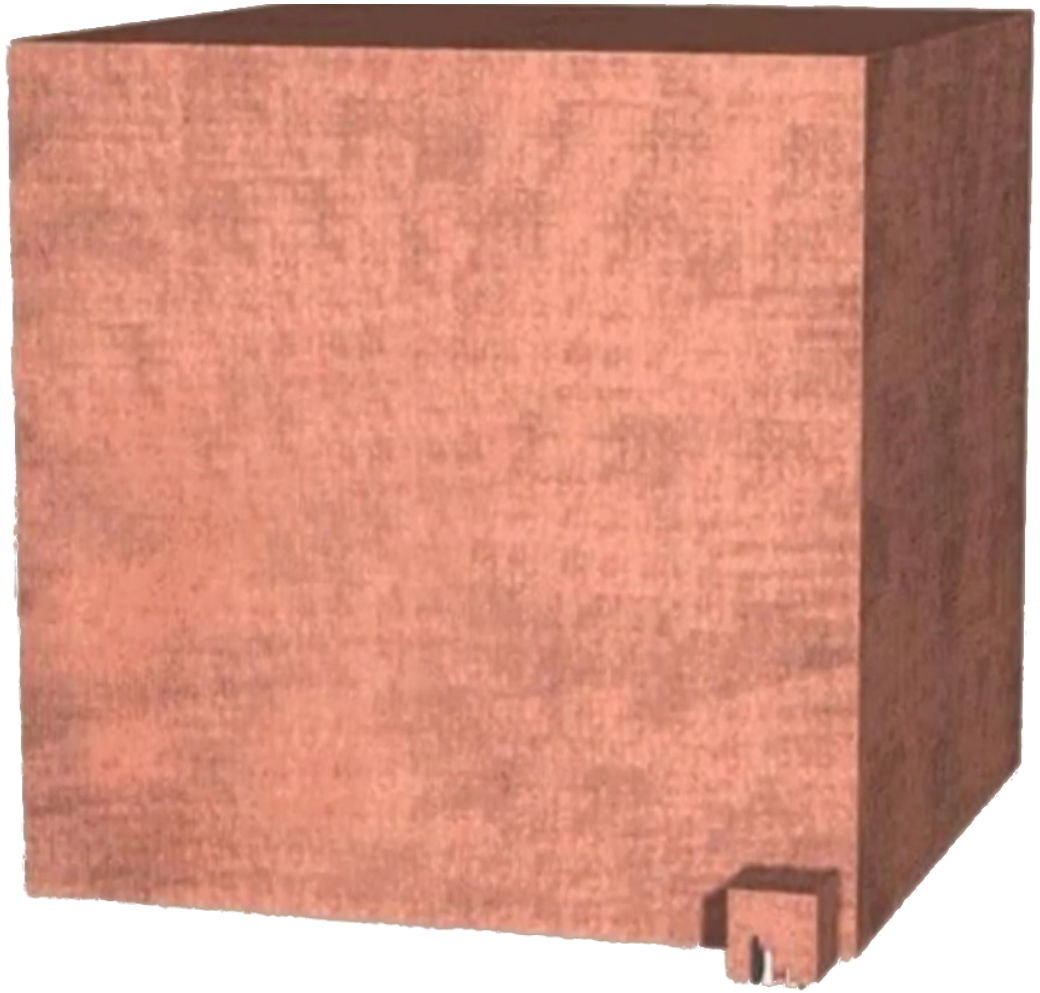
- **La tierra** completa contiene unos:
  1.234 quintillones de litros de agua
  326 quintillon gallons of water

- Si cortamos **la tierra por la mitad**, la sección tendría un área aproximada de:
  1.275 quintillones de centímetros cuadrados

- ¿Cuanto es un quintillón de céntimos o peniques de dólar?

¿Cuanto es un quintillón... de peniques?

¿Cuanto es un quintillón... de peniques?



1.000.067.088.384.000.000 peniques
1 quintillón, 67 trillones, 88 billones, 384 millones de peniques
Un cubo de **8,32 kilómetros** de lado

Teorema de CAP

Eric Brewer (2000)

C: Consistency → Consistencia

A: Availability → Disponibilidad

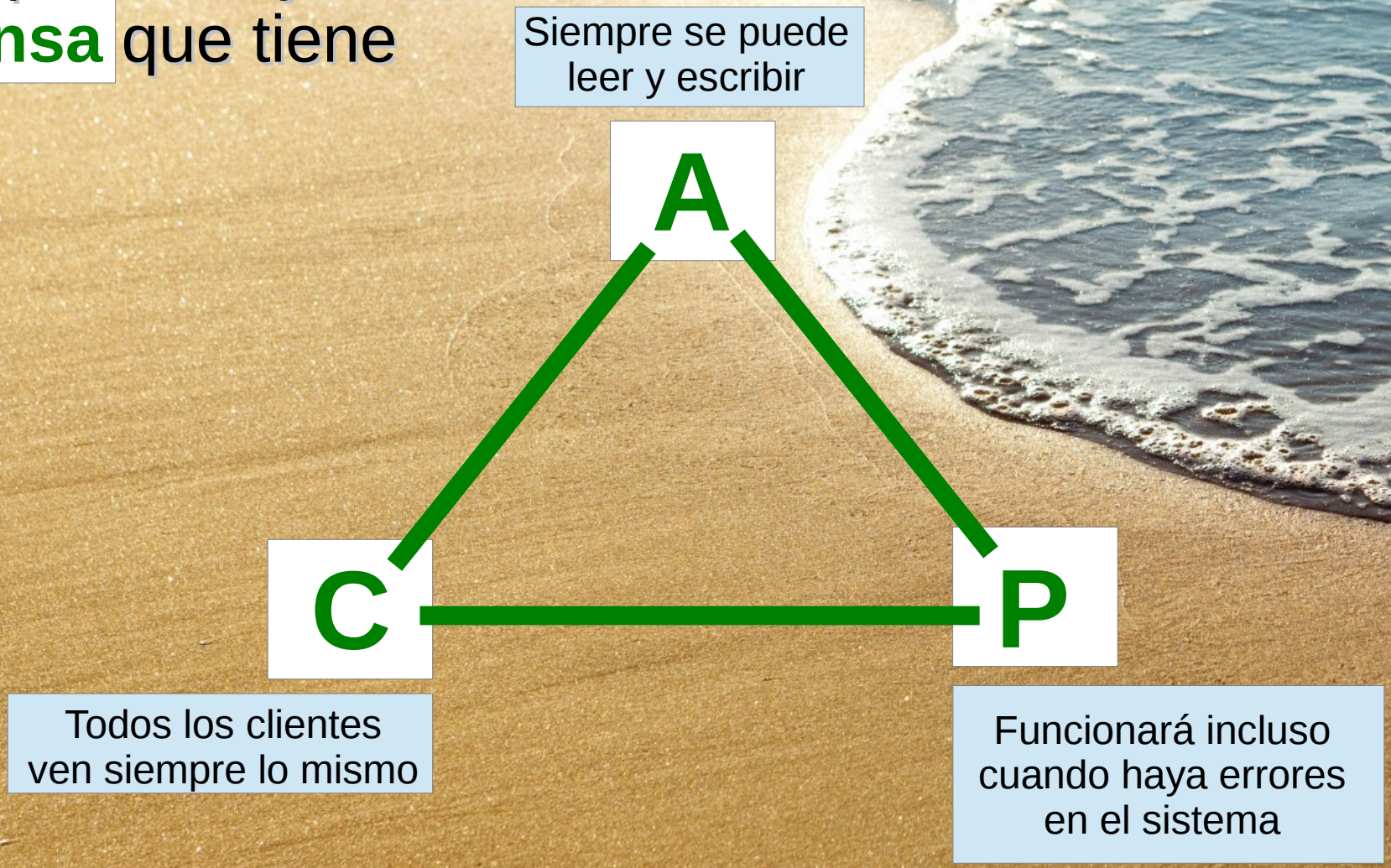
P: Partition tolerance → Tolerancia al particionado

Sólo puedes llegar a 2 de las 3

Es imposible para un sistema de cómputo distribuido garantizar simultáneamente la **consistencia**, la **disponibilidad** y ser tolerante al **particionado** de los datos (separación y distribución).

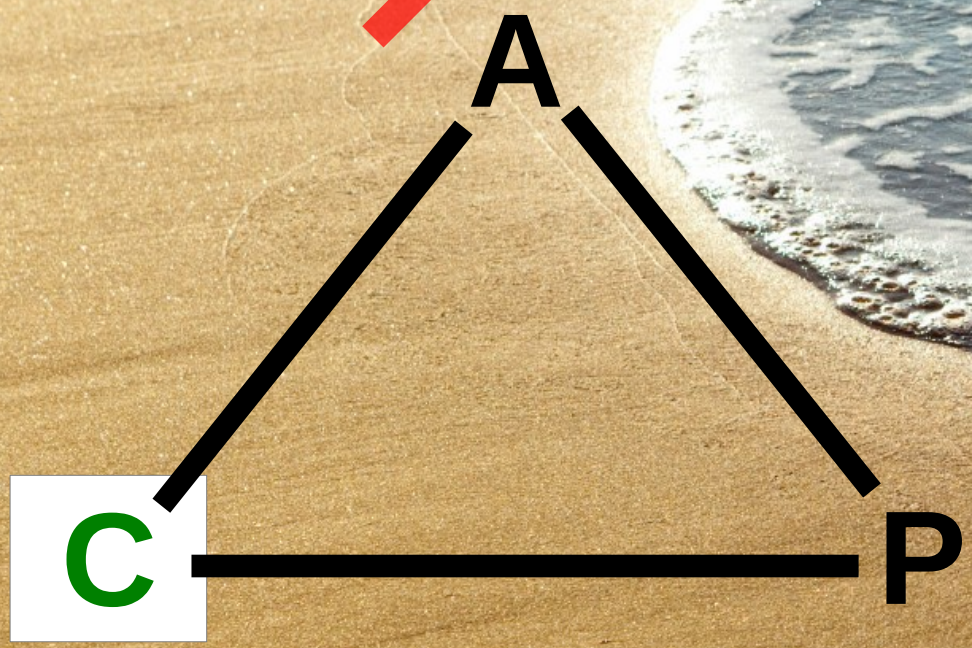
Teorema de CAP

Lo que la mayoría **piensa** que tiene



Teorema de CAP

Lo que la mayoría de **verdad** tiene



Siempre se puede leer y escribir

Todos los clientes ven siempre lo mismo

Funcionará incluso cuando haya errores en el sistema

Lo cierto es que todos buscamos la **disponibilidad (A)**

Pero ... ¡¡¡ tenemos que elegir entre... !!!

Escalabilidad (P)

y

Consistencia (C)

ACID

A: Atomicidad

C: Consistencia

I: Aislamiento (Isolation)

D: Durabilidad

En grandes sistema ocurre que:
Disponibilidad y Rendimiento

BASE

BA: Básicamente disponible

S: Flexible (Soft state)

E: Consistencia eventual

Da **menos** importancia a la **consistencia** en **pro** de la tolerancia al **particionado** aparece la **consistencia eventual**

¿Qué es la consistencia eventual?

Que ... eventualmente será consistente

Podemos introducir un dato y que
no esté disponible inmediatamente después

Convergencia natural a la consistencia

Resolución de conflictos

Anti-entropía (control de versiones)

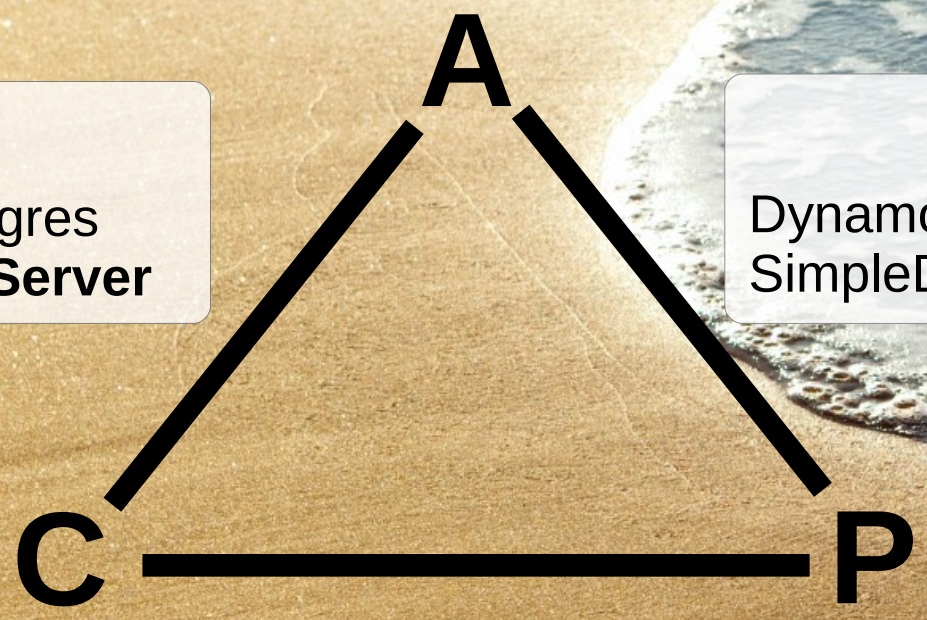
Reconciliación (elección de estado final)
[generalmente: “last write wins”]

Garantizar la seguridad de una operación

Strong eventual consistency (SEC)

Teorema de CAP

Siempre se puede leer y escribir



CA

MySQL Postgres
Oracle **SQLServer**

AP

Dynamo **Cassandra**
SimpleDB CouchDB

Todos los clientes ven siempre lo mismo

CP

MongoDB Redis
MemcacheDB Hbase
Berkely DB

Funcionará incluso cuando haya errores en el sistema

Teorema de CAP

CA

MySQL
Postgres
Oracle
SQLServer
Aster Data
Greenplum
Vertica

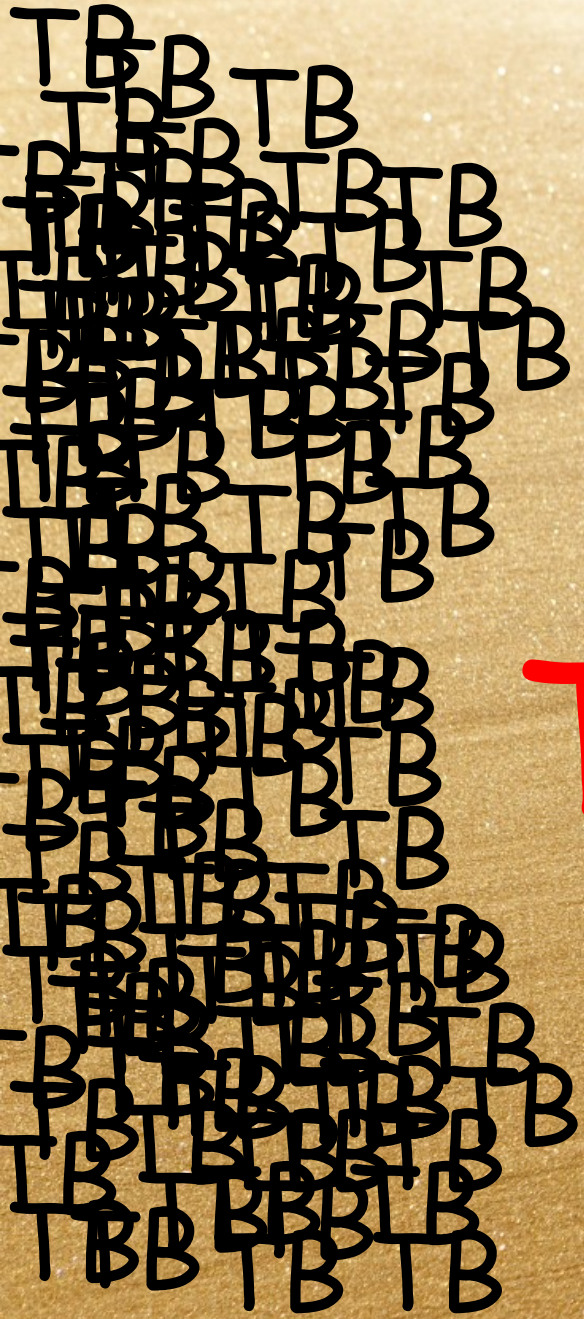
AP

Dynamo
Voldemort
Tokyo Cabinet
KAI
Cassandra
SimpleDB
CouchDB
Riak

CP

BigTable
Hypertable
HBase
MongoDB
Terrastore
Scalaris
Berkely DB
MemcacheDB
Redis

...BREAK...



¡Hey tío!, somos 1023,
¿te vienes y nos hacemos un Peta?

TB

TB

¡Hecho!

Disponibilidad contínua

Simplicidad en la gestión entre servidores
Sin un único punto de fallo

Escalabilidad linear
- Si 2 nodos procesan 100 transacciones/seg
- 4 nodos procesan 200 transacciones/seg
- 8 nodos procesan 400 transacciones/seg

Sistema **descentralizado** (sin master)

Relaciones por grupo: Nodo → Datacenter → Cluster
Replicación personalizada

Gossip: mantiene la red informada

Partitioner: deciden como se distribuyen los datos

Replication factor: número de réplicas en el cluster

Replica placement strategy: donde poner las réplicas

Snitch: define grupo de máquinas destinadas a réplicas

Gossip:

- Protocolo punto a punto

- Comunicación entre nodos

- Detección de fallos y recuperación

- Autodetección de topología e información

Particionador: encargado de “**esparcir**” los datos.

Desaconsejados:
Random: Hashes con MD5 (mejor usar Murmur3)
ByteOrdered: orden lexicográfico sobre las claves
OrderPreserving: se asumen claves en formato UTF8

Murmur3: Funcionalmente idéntico a Random pero es más rápido sin efectos colaterales. Se centra en la distribucción espacial.

Planificar el deploy de un cluster

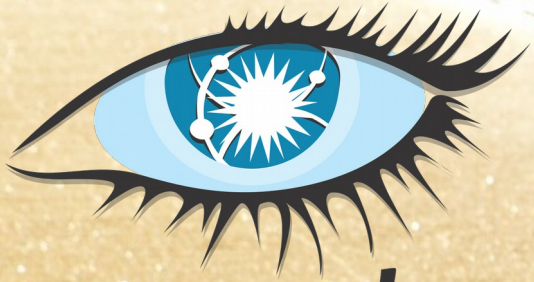
Hardware:

- RAM: 16GB-64Gb (mínimo 8Gb)
- CPU: 8-cores dedicados o 4-8 cores en virtuales
- Disco: mejor entre 500Gb y 1Tb por nodo (según I/O)
- 2 discos (commit log + data)
- Sistema de ficheros XFS
- Red mínimo Gigabit

Planificar el deploy de un cluster

Espacio útil disco: 45%-70% de espacio total RAW

Antipatrones:
- Usar un NAS
- Sistemas de ficheros compartidos
- SELECT … IN
- Leer antes de escribir (multiples hits)
- Balanceadores de carga
- Falta de testing
- Bajo conocimiento de Linux



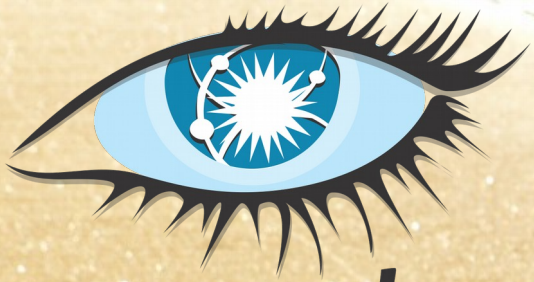
cassandra CQL

SQL

```
USE myDatabase;

/* Creating Tables */
CREATE TABLE IF NOT EXISTS myTable (id INT PRIMARY KEY);

/* Altering Tables /*
ALTER TABLE myTable ADD myField INT;

/* Creating Indexes */
CREATE INDEX myIndex ON myTable (myField);

/* Inserting Data */
INSERT INTO myTable (id, myField) VALUES (1, 7);

/* Selecting Data */
SELECT * FROM myTable WHERE myField = 7;

/* Counting Data */
SELECT COUNT(*) FROM myTable;

/* Deleting Data */
DELETE FROM myTable WHERE myField = 7;
```

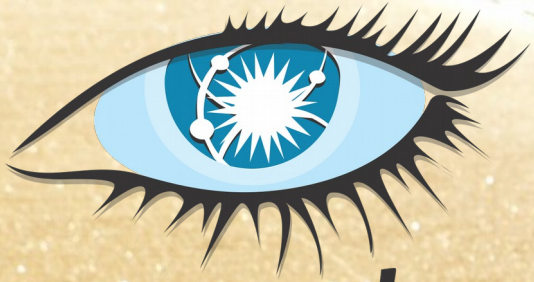
CQL

```
USE myDatabase;

/* Creating Tables */
CREATE TABLE IF NOT EXISTS myTable (id INT PRIMARY KEY);

/* Altering Tables /*
ALTER TABLE myTable ADD myField INT;

/* Creating Indexes */
CREATE INDEX myIndex ON myTable (myField);

/* Inserting Data */
INSERT INTO myTable (id, myField) VALUES (1, 7);

/* Selecting Data */
SELECT * FROM myTable WHERE myField = 7;

/* Counting Data */
SELECT COUNT(*) FROM myTable;

/* Deleting Data */
DELETE FROM myTable WHERE myField = 7;
```

cassandra **CQL**

```
USE miBaseDatos;

/* Creando Tablas */
CREATE TABLE IF NOT EXISTS miTabla (id INT PRIMARY KEY);

/* Cambiando Tablas /*
ALTER TABLE miTabla ADD miCampo INT;

/* Creating Indexes */
CREATE INDEX miIndice ON miTabla (miCampo);
```

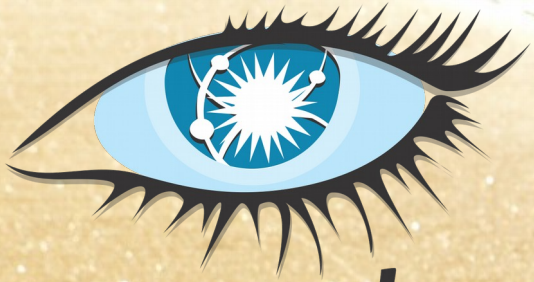
cassandra CQL

/* Insertando Datos */
INSERT INTO miTabla (id, miCampo) VALUES (1, 7);

/* Seleccionando Datos */
SELECT * FROM miTabla WHERE miCampo = 7;

/* Contando Datos */
SELECT COUNT(*) FROM miTabla;

/* Borrando Datos */
DELETE FROM miTabla WHERE miCampo = 7;

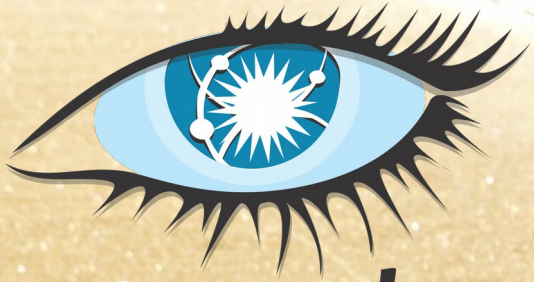


cassandra **CQL**

/* Crear un nuevo keyspace en CQL */
CREATE KEYSPACE miBaseDatos WITH replication =
{'class': 'SimpleStrategy', 'replication_factor': 1};

/* Crear una nueva base de datos en SQL */
CREATE DATABASE miBaseDatos;

No existen:
JOIN, **GROUP BY**, y **FOREIGN KEY**



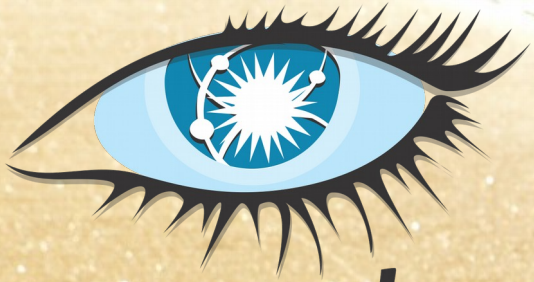
cassandra CQL

1.- Las escrituras son baratas. Escribe todo del modo en que vas a leerlo.

2.- UPSERT: INSERT (inserta ó actualiza), UPDATE (inserta ó actualiza), ya que Cassandra no hace un read durante estos procesos.

3.- Filas con TTL, pasado X tiempo la fila caduca.

4.- DELETE ... ¡no borra!

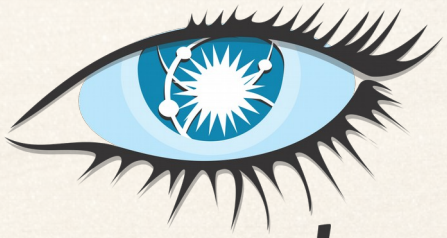


cassandra CQL

/* Seleccionar datos de un rango */
SELECT * FROM myTable
WHERE miCampo > 5000 AND miCampo < 100000;

Bad Request: Cannot execute this query as it might involve data filtering and thus may have unpredictable performance. If you want to execute this query despite the performance unpredictability, use ALLOW FILTERING.

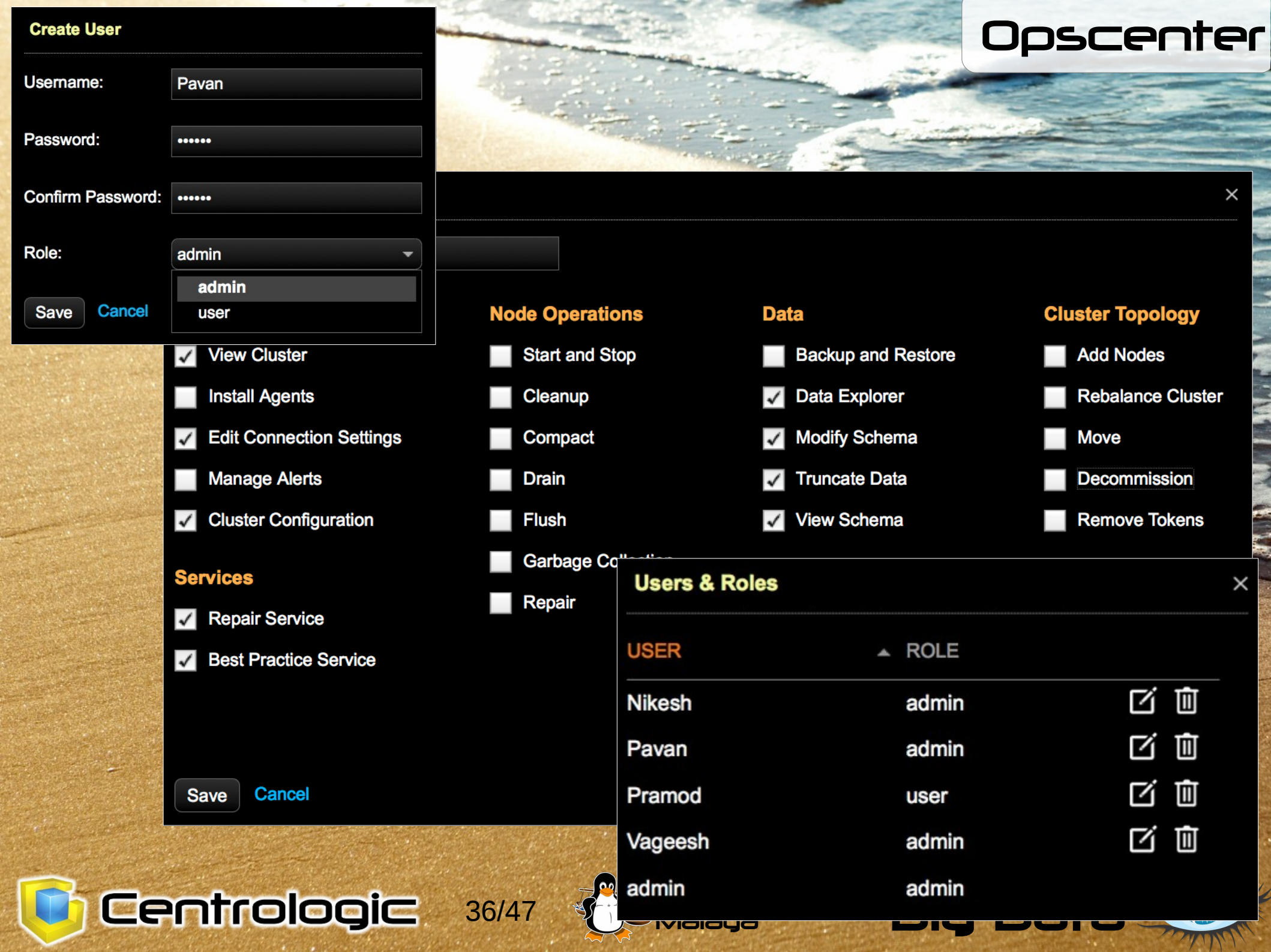
¿Cómo se gestiona esto?



DATASTAX

Opscenter

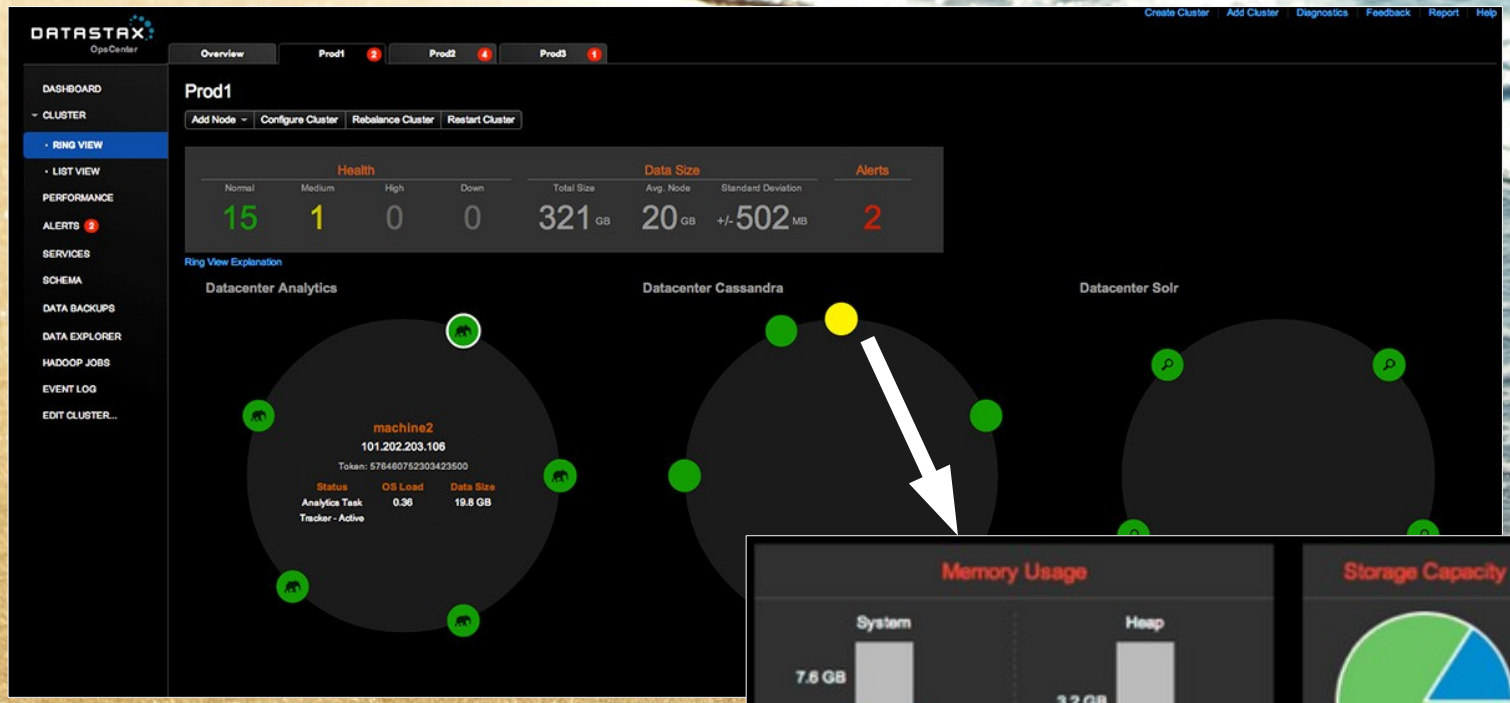

RING
LIST
Health
Normal
Medium
High
Down
983
17
0
0
Data Size
Total Size
Avg. Node
Standard Deviation
20 TB
20 GB
+/- 593 MB
Alerts
0
Datacenter Analytics
Workload: Spark Analytics
328
5
0
0
Datacenter Cassandra
332
2
0
0
Datacenter Solr
Workload: Search
323
10
0
0
RING
LIST
Health
Normal
Medium
High
Down
524
303
173
0
Data Size
Total Size
Avg. Node
Standard Deviation
20 TB
20 GB
+/- 594 MB
Alerts
0
Datacenter Analytics
Workload: Spark Analytics
171
109
53
0
Datacenter Cassandra
192
90
52
0
Datacenter Solr
Workload: Search
161
104
68
0

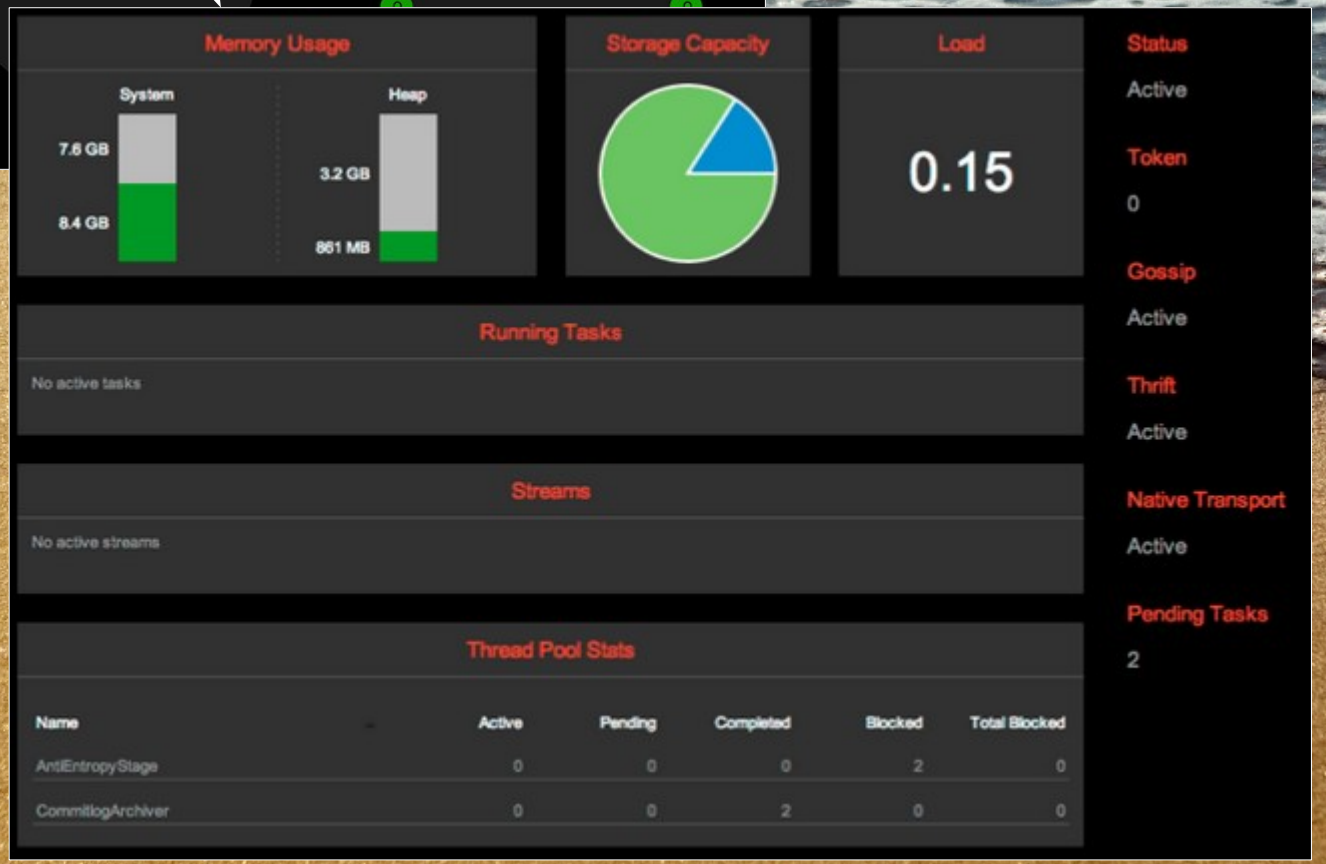

Opscenter
Create User
Username:
Pavan
Password:
Confirm Password:
Role:
admin
admin
user
Save
Cancel
Node Operations
Data
Cluster Topology
View Cluster
Install Agents
Edit Connection Settings
Manage Alerts
Cluster Configuration
Start and Stop
Cleanup
Compact
Drain
Flush
Repair
Backup and Restore
Data Explorer
Modify Schema
Truncate Data
View Schema
Add Nodes
Rebalance Cluster
Move
Decommission
Remove Tokens
Services
Repair Service
Best Practice Service
Save
Cancel
Users & Roles
USER
ROLE
Nikesh admin
Pavan admin
Pramod user
Vageesh admin
admin admin
Centrologic
36/47
Malaga

Opscenter


DATASTAX
NEW CLUSTER | ALERTS 9 | SETTINGS | HELP admin
Overview
Prod1
Prod2
Prod3
Dev1
Prod1: Dashboard
DataStax Enterprise Server 4.5.0
All agents connected
Cluster Actions
DEFAULT
MEMORY
OBJECTS
OPERATING SYSTEM
2014-06-18
3:38 PM
to
2014-07-18
3:38 PM
Update
Current
Graph Scale 20m | Hour | Day | Week | Month
Add Graph
Add Widget
Active Alerts
Write Request Latency at 20.139 ms/op - 101.202.203.105
Cluster Health
Datacenter Cassandra
3
4
1
0
Storage Capacity
Used: 146 GB Free: 186 GB Total: 332 GB
8 of 8 nodes
Write Requests
Cluster
4307/sec
3869/sec
3431/sec
2993/sec
2555/sec
Jun 18 Jun 26 Jul 3 Jul 11
Cluster (total) Cluster (max)
Write Request Latency
Cluster
1 ms/op
0.8 ms/op
0.6 ms/op
0.4 ms/op
0.2 ms/op
Jun 18 Jun 26 Jul 3 Jul 11
Cluster (avg) Cluster (max)
OS: Disk Utilization
Cluster
100%
80%
60%
40%
20%
0%
Jun 18 Jun 26 Jul 3 Jul 11
Cluster (avg) Cluster (max)
OS: Load
Cluster
1
0.75
0.50
0.25
0
Jun 18 Jun 26 Jul 3 Jul 11
Cluster (avg) Cluster (max)
OS: Memory (stacked)
machine1, machine2, machine3, 101.202.203.108,...
8 MB
6 MB
4 MB
2 MB
0 MB
Jun 18 Jun 26 Jul 3 Jul 11
OS: Memory Used (avg)

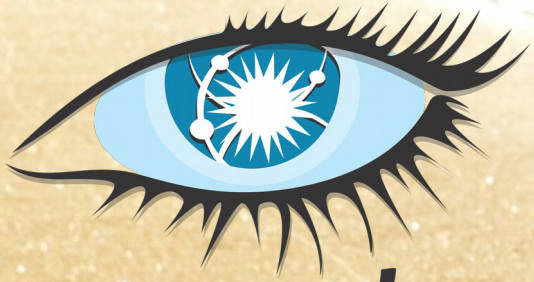
Opscenter


DATASTAX
OpsCenter
Create Cluster
Add Cluster
Diagnostics
Feedback
Report
Help
Overview
Prod1
Prod2
Prod3
DASHBOARD
CLUSTER
RING VIEW
LIST VIEW
PERFORMANCE
ALERTS
SERVICES
SCHEMA
DATA BACKUPS
DATA EXPLORER
HADOOP JOBS
EVENT LOG
EDIT CLUSTER...
Prod1
Add Node
Configure Cluster
Rebalance Cluster
Restart Cluster
Health
Normal
Medium
High
Down
15
1
0
0
Data Size
Total Size
Avg. Node
Standard Deviation
321 GB
20 GB
+/-502 MB
Alerts
2
Ring View Explanation
Datacenter Analytics
Datacenter Cassandra
Datacenter Solr
machine2
101.202.203.106
Token: 57646075230342350O
Status
OS Load
Data Size
Analytics Task Tracker - Active
0.36
19.8 GB


Memory Usage
System
Heap
7.6 GB
8.4 GB
3.2 GB
861 MB
Storage Capacity
Load
0.15
Status
Active
Token
0
Gossip
Active
Thrift
Active
Native Transport
Active
Pending Tasks
2
Running Tasks
No active tasks
Streams
No active streams
Thread Pool Stats
Name
Active
Pending
Completed
Blocked
Total Blocked
AntiEntropyStage
0
0
0
2
0
CommitlogArchiver
0
0
2
0
0

¿Quién usa Cassandra?

Fuente: http://planetcassandra.org/companies/



CALL OF DUTY
CERN
COMCAST
Alcatel·Lucent
agoda
CREDIT SUISSE
The New York Times
accenture
High performance. Delivered.
KASPERSKY lab
OPERA software
Aol.
NBCUniversal
reddit
Adobe
Spotify
THOMSON REUTERS
vmware
Disney
AVIRA
Bla Bla Car
FedEx
SanDisk
hp
avast!
Centrologic
NASA
rackspace
UBISOFT
CISCO
PayPal
Expedia
Sony Entertainment
softonic
Symantec
ebay
DELL
ERICSSON
GitHub
NETFLIX
BlackBerry
IBM
despegar.com
ING
SHAZAM
Telefónica
Instagram
Fast beautiful photo sharing
MuleSoft
connecting the new enterprise
Microsoft
unity

¿Quién usa Cassandra?

cassandra

CERN

Al finalizar la última iteración del LHC en el que se descubrió el "**Bosón de Higgs**", el CERN almacenaba más de **100Pbytes**

El LHC del CERN arrancó en **Abril de 2015** en busca de la **Supersimetría** (capaz de producir 1Pbytes/segundo)

1Petabyte = 1.000 Terabytes = 1 Millón de Gigabytes

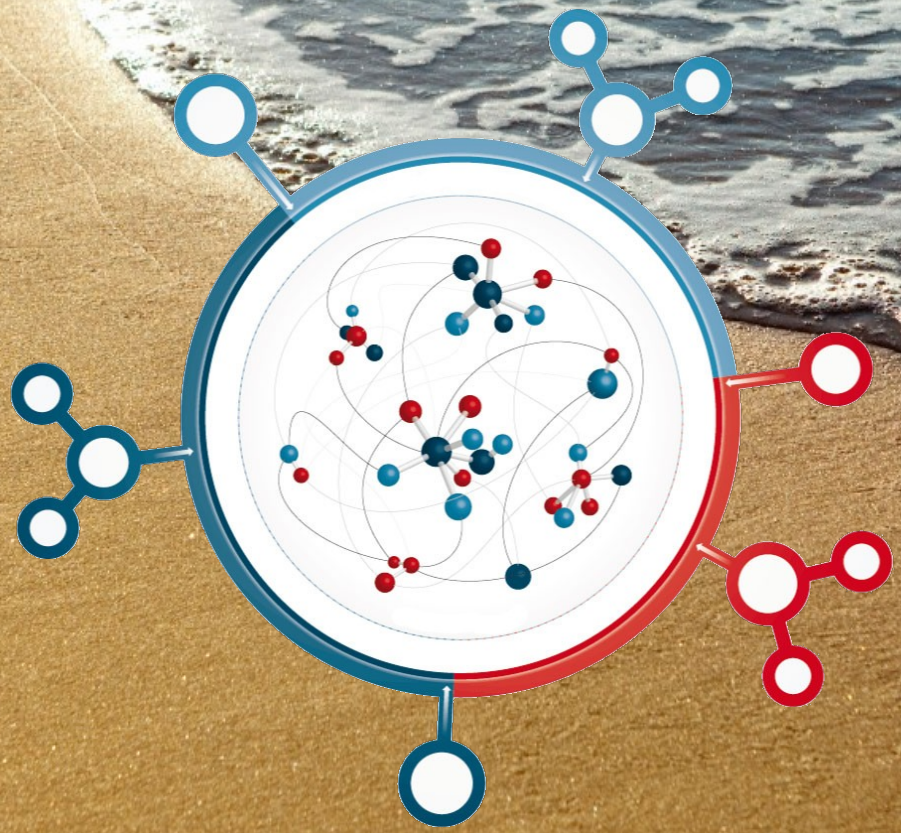
NETFLIX

Ejecuta 235 clusters separados, con un total de 7.000 nodos.
1 Millón de escrituras por segundo (factor 3)

unity

Ecosistema para desarrolladores de juegos que teniendo problemas con MongoDB migraron a Cassandra.

…esta historia parecía de cuento...
…los datos se repartían por los nodos...
…existía resilencia...
…había alta disponibilidad...
…crecía linealmente f(nodos)...
…todo parecía ideal...



...pero
con el paso del tiempo
a la bella Cassandra
le surgió un amante...

...con el paso del tiempo

...conoció a KairosDB

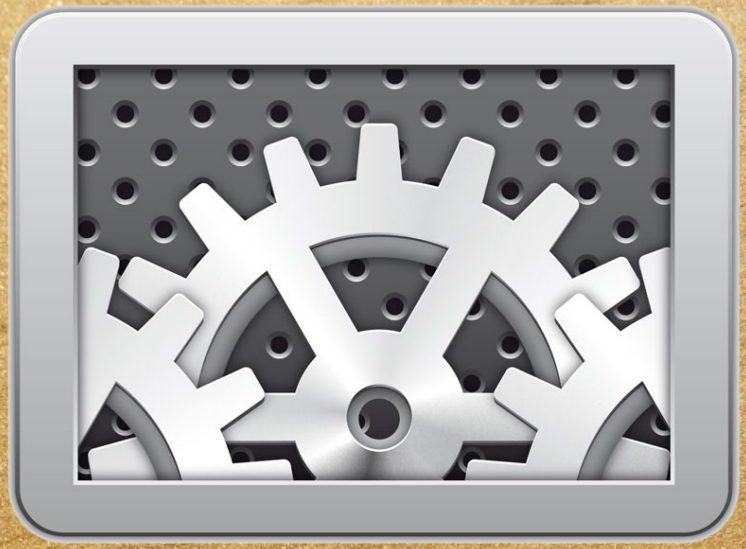
KairosDB

KairosDB

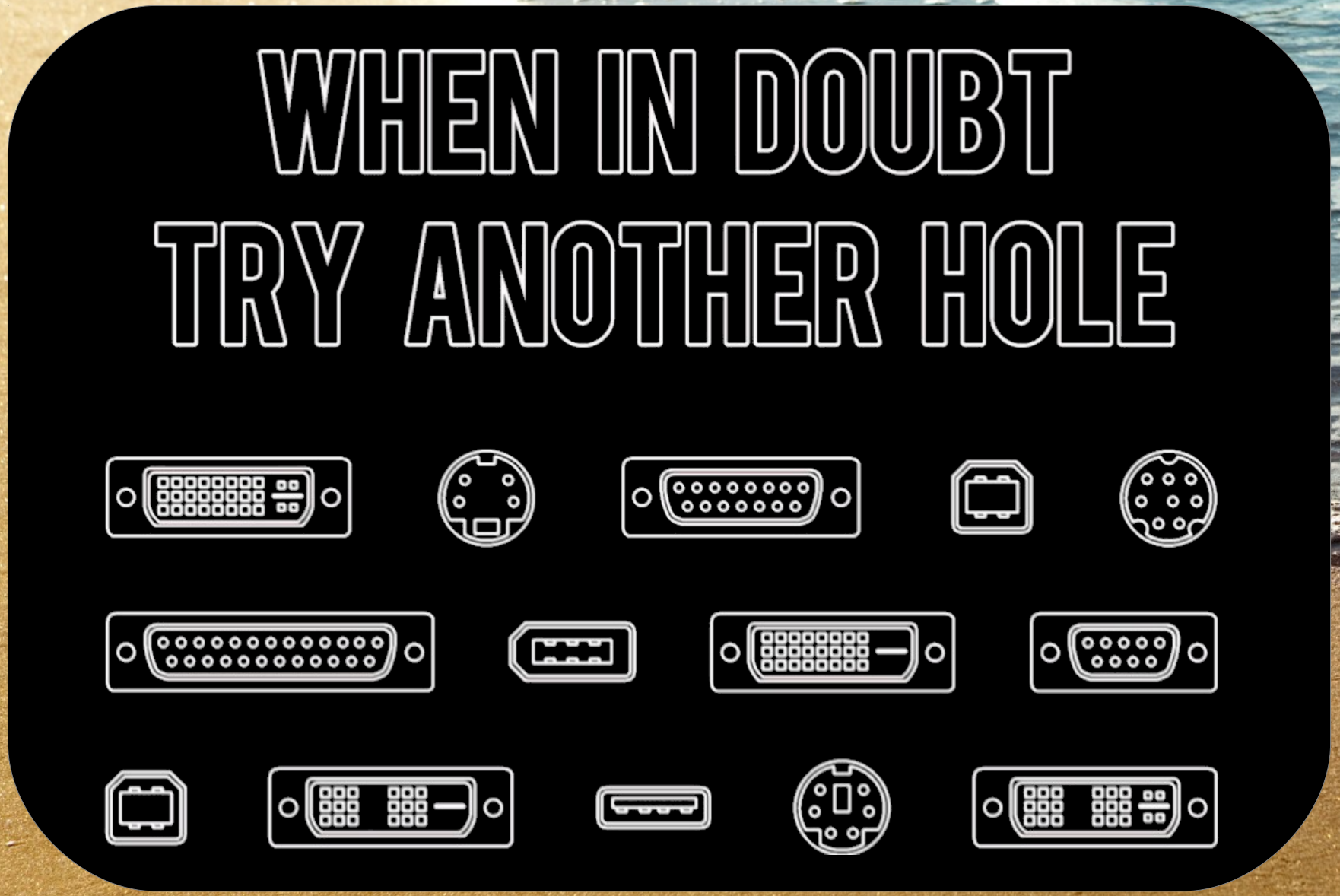
“No podemos confundir potencia con potencial”

- Brian Hawkins (Salt Lake) y Jeff Sabin (Utah) de Proofpoint. (Link)
- Base de datos para series temporales sobre Cassandra
- 11 de abril de 2013 primera versión Beta
- Cada registro tiene: Métrica + Fecha + Tipo + Tags
- 3 semanas de datos o 1.814,4 Millones de **columnas** exactamente
- Genial...pero ¿cómo mejora esto el rendimiento?

DEMO



¿Dudas?



Juan Miguel Taboada Godoy
http://www.centrologic.com
@centrologic_es
http://linkedin.com/user/centrologic

Juan José Soler Ruiz
@soleronline
http://es.linkedin.com/in/soleronline