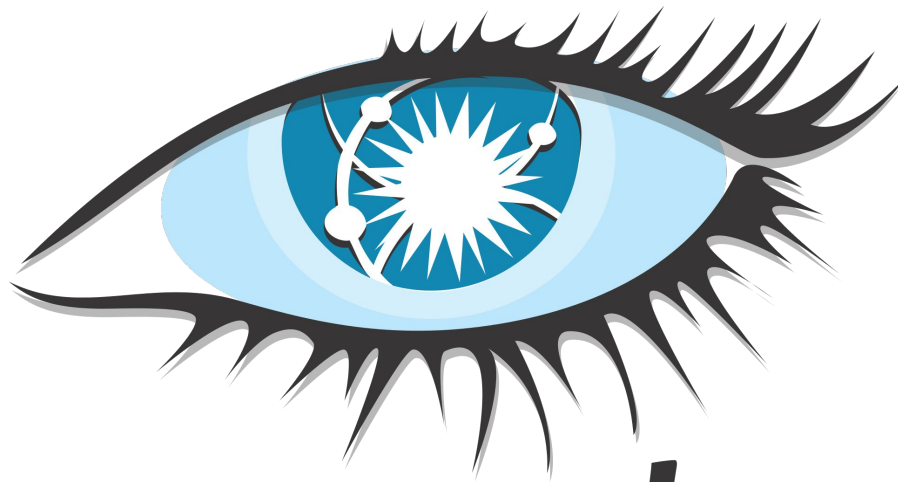


Cassandra NoSQL



cassandra

Proyecto Fin de Ciclo G.S. - ASIR

Nombre: Juan José López Roldán

Promoción: 2016-2018

cassandra

Índice

1. Introducción.
2. ¿Qué es NoSQL?
3. Motivo Aparición.
4. Arquitectura y Características.
5. Modelo De Datos y Lenguaje CQL.
6. Caso Práctico.



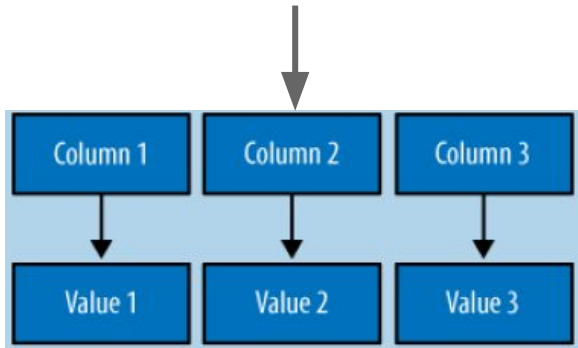
Introducción



cassandra

Introducción

Cassandra es una base de datos de tipo NoSQL distribuida y basada en modelo “Clave-Valor”.



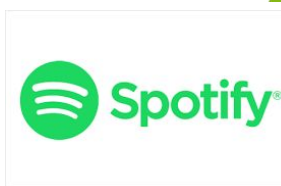
Características:

- Pertenece a la Apache Software Foundation.
- Está escrita en Java.
- Permite manejar gran cantidad de datos.
- Muy recomendable para el Big Data.
- Es utilizada por grandes empresas que manejan mucha información.

NETFLIX



ebay



cassandra



¿Qué es NoSQL?



cassandra

¿Qué es NoSQL?

NoSQL (Bases de datos no relacionales).

Son famosas por:

- Facilidad de desarrollo.
- Desempeño escalable.
- Alta disponibilidad.
- Alta capacidad para volver a un estado coherente.

cassandra



Motivo de Aparición



cassandra

Motivo de Aparición

Motivos Principales de la aparición de estas bases de datos:

- Los pocos problemas de alta escalabilidad para los servicios en la nube.
- Desnormalización de la información “Adios a los JOIN”.
- Almacenamiento versátil.



cassandra

Diferencias de NoSQL y RDBMS

Principales diferencias:

- Ausencia de esquema en los registro de datos.
- Escalabilidad horizontal sencilla.
- Velocidad (en el caso de Cassandra las consultas son veloces ya que los datos a obtener son muy pocos por que se usa para consultas simples).



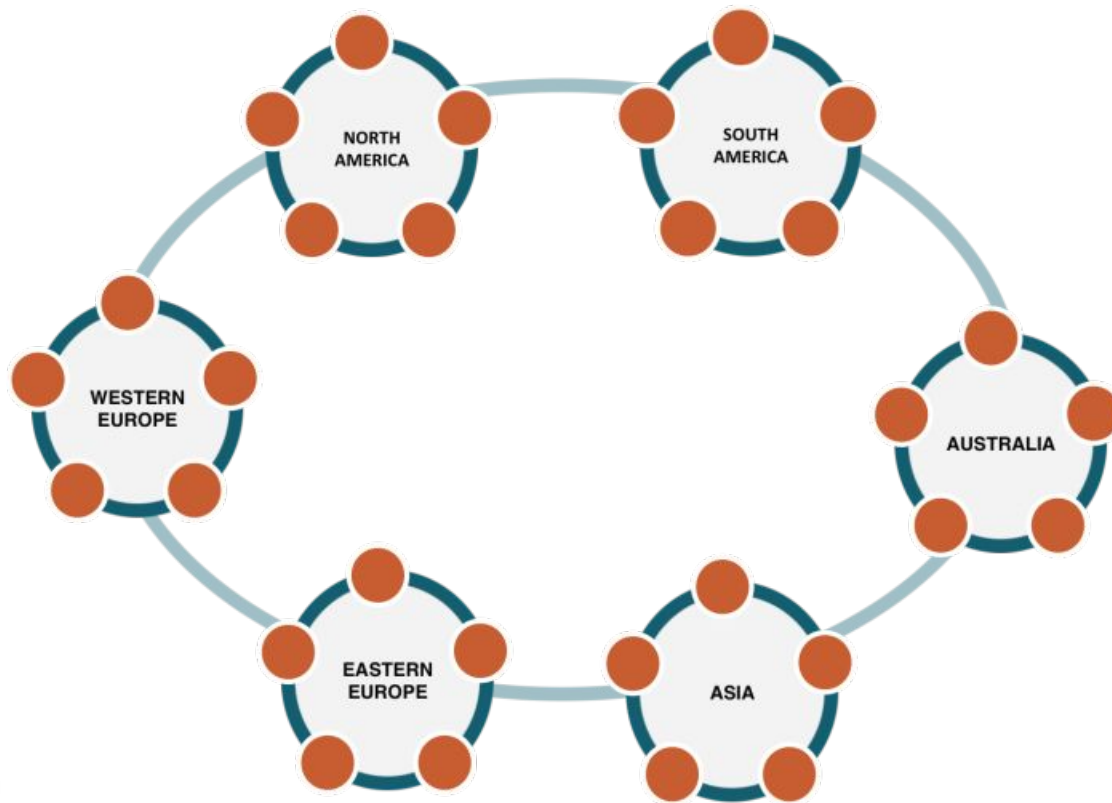
Arquitectura y Características



cassandra

Arquitectura y Características

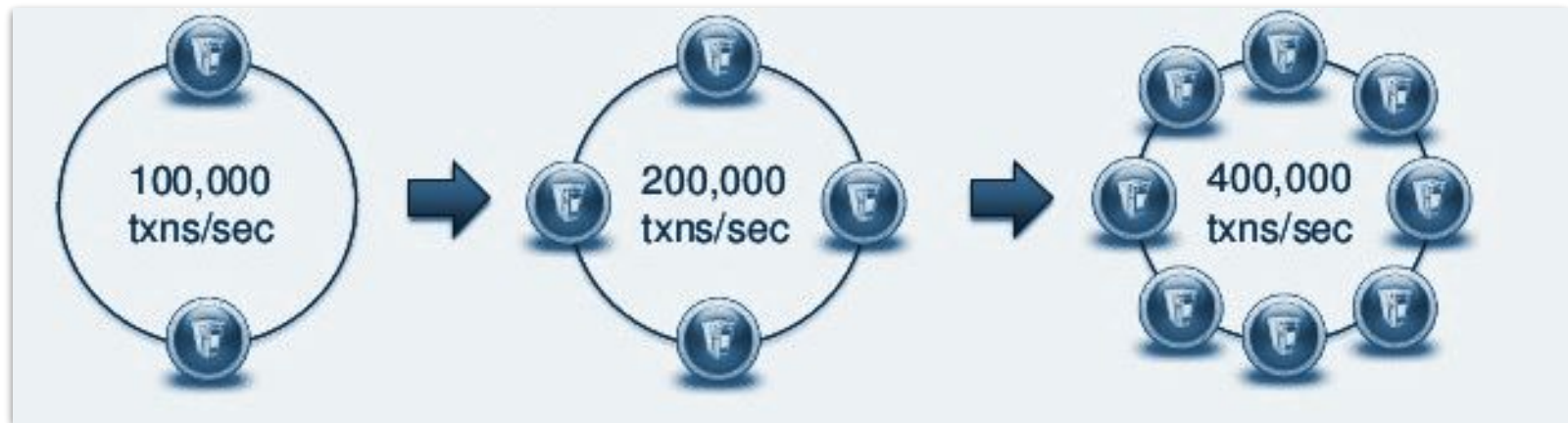
- Escala Distribuida: Información replicada y repartida por todos los nodos del cluster, si cae un nodo el servicio no se degrada.



cassandra

Arquitectura y Características

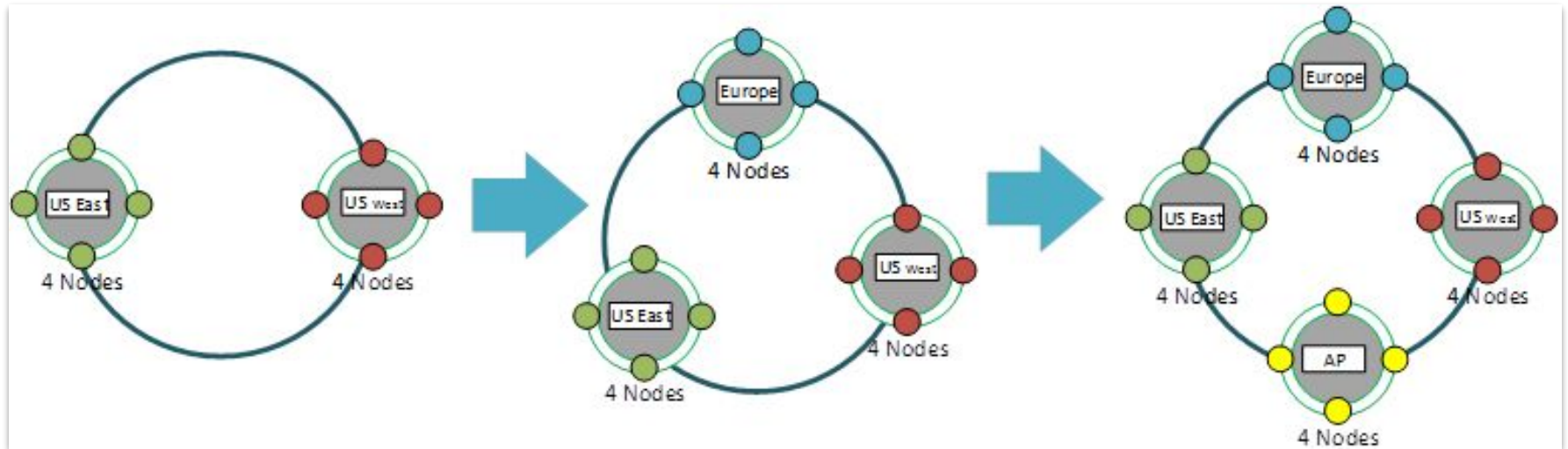
- Escala Linealmente: Rendimiento de forma líneal (Relación Nodo-Operaciones/seg)



cassandra

Arquitectura y Características

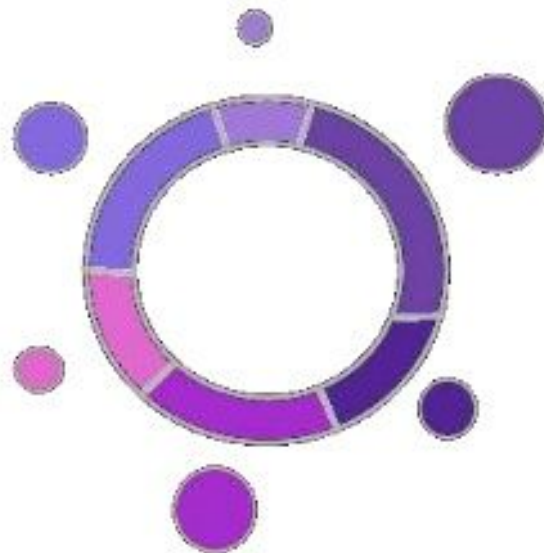
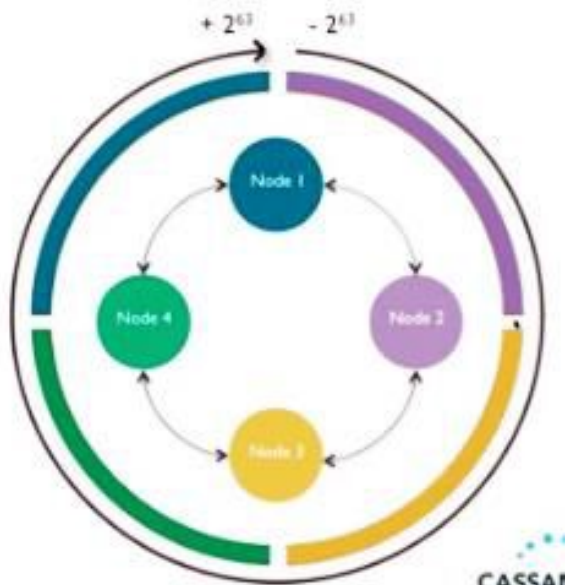
- Escala Horizontalmente: Escalado de nuestro sistema añadiendo nodos de bajo costo.



cassandra

Arquitectura y Características

- Implementa Arquitectura Peer-to-Peer (igual importancia).
- No Master-Slave
- Cualquiera es nodo controlador (gestión interna).
- Información replicada y repartida por todos los nodos.
- Tiene un Hash y está repartido por los nodos.
 - Se reparte dependiendo de la potencia computacional.
 - El hash tiene un rango de -2^{63} a 2^{63} .



cassandra



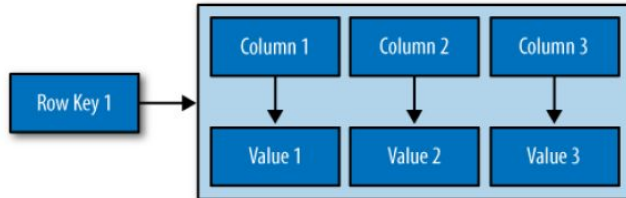
Modelo de datos y Lenguaje CQL



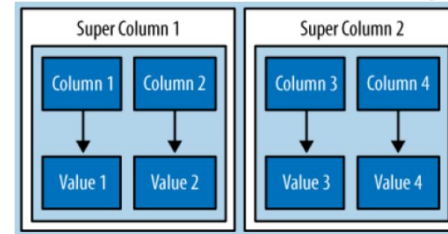
cassandra

Modelo de datos

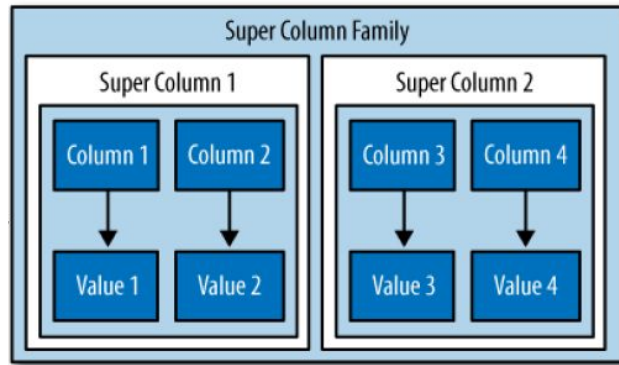
1. Columna



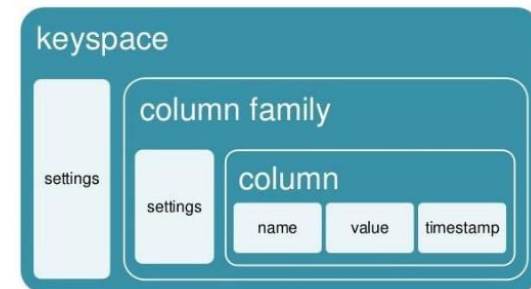
2. Super-Column



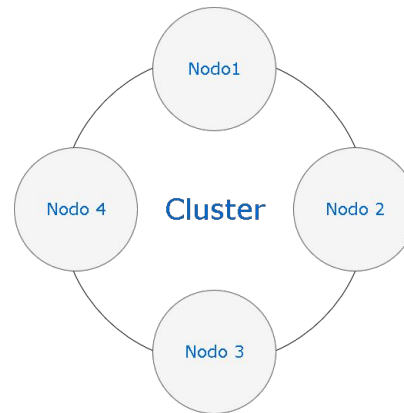
3. Familia de Columnas (Tabla)



4. Keyspace



5. Cluster



cassandra

Lenguaje CQL

Cassandra Query Language (**CQL**) es el lenguaje que utiliza para realizar el acceso a la base de datos (No es SQL pero se le parece).

CONSULTAS

- Consultas simples, por rangos y de igualdad.

```
SELECT * WHERE Nombre = 'Pepito' and numero IN (2, 3, 4);
```

```
SELECT * WHERE Nombre = 'Pepito' and numero >=6 and numero <=10;
```

CREACIÓN DE OBJETOS

```
CREATE KEYSPACE IF NOT EXISTS "nombre keyspace" WITH replication = { 'class':  
'SimpleStrategy', 'replication_factor': '1'} AND durable_writes = true;
```

```
CREATE TABLE IF NOT EXISTS  
personas  
(  
id text,  
name text,  
surname text,  
date time,  
PRIMARY KEY (date, name));
```

ELIMINACIÓN DE OBJETOS

```
DROP KEYSPACE "nombre de keyspace";
```

```
DROP TABLE "nombre tabla";
```

cassandra

Lenguaje CQL

ACTUALIZACIÓN DE OBJETOS

- Actualización de Keyspaces y tablas.

```
ALTER KEYSPACE "nombre keyspace" WITH REPLICATION = { 'class' : 'SimpleStrategy',  
'replication_factor' : 3 };
```

```
ALTER TABLE personas ADD email text;
```

INSERCIÓN DE DATOS

```
INSERT INTO personas (id, name, surname, email) VALUES ('001', 'antonio', 'jimenez',  
'antonio@correo.es');
```

ACTUALIZACIÓN DE DATOS

- Actualización de datos en una tabla.

```
UPDATE personas SET email='correojuanjose@correo.com' WHERE id='001';
```

Lenguaje CQL

ELIMINACIÓN DE DATOS

- Eliminación de contenido de una fila.

```
DELETE nombre FROM personas WHERE id='001';
```

- Eliminación de la fila completa.

```
DELETE FROM personas WHERE id='001';
```

CREACIÓN DE USUARIOS

```
CREATE USER IF NOT EXISTS "nombre usuario" WITH PASSWORD  
'password usuario' NOSUPERUSER | SUPERUSER;
```

GESTIÓN DE PRIVILEGIOS

- Asignación de privilegios.

```
GRANT "privilegio a conceder" ON "objeto que concede" TO  
"usuario/rol";
```

- Revocación de privilegios.

```
REVOKE "privilegio a revocar" ON "objeto a revocar" FROM  
"usuario/rol";
```

Lenguaje CQL

OTRO TIPOS DE CREACIONES EN CASSANDRA

Además de las creaciones indicadas anteriormente, podemos realizar la creación de los siguientes objetos en Cassandra.

- Indices

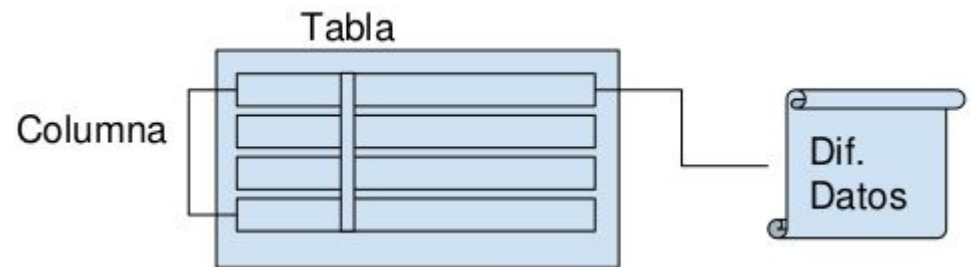
```
CREATE INDEX idx_nombreusuario ON  
personas(nombre);
```

- Triggers

```
CREATE TRIGGER IF NOT EXISTS trigger_name ON table_name USING  
'java_class';
```

- Type

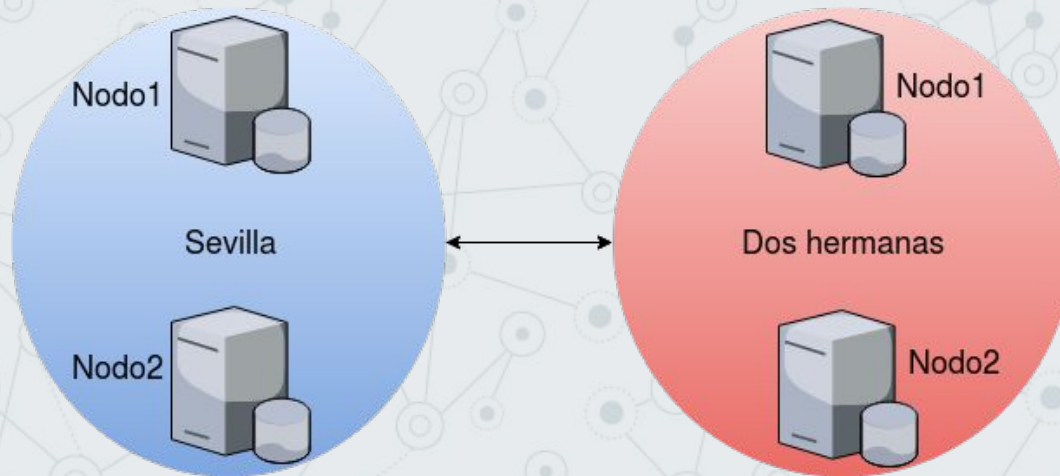
```
CREATE TYPE direcciones  
(  
  calle text,  
  ciudad text,  
  cod_postal int,  
  movil set<text>  
);
```



cassandra

Caso Práctico

Alta disponibilidad y tolerancia a fallos con Cassandra



cassandra

Caso Práctico

Replicación de tabla de autenticación

```
alter keyspace system_auth WITH replication = {'class':  
'NetworkTopologyStrategy', 'sevilla': 2, 'doshermanas':  
2};
```

Creación keyspace denominado Comvive

```
create keyspace if not exists comvive with replication = { 'class'  
: 'NetworkTopologyStrategy', 'sevilla': 2, 'doshermanas': 2};
```

Creación usuario denominado juanjosé

```
create user if not exists juanjose with password 'juanjose' superuser;
```

Modificamos los privilegios del usuario

```
GRANT ALL ON KEYSPACE comvive TO  
juanjose;
```


Caso Práctico

Conectamos con el usuario y usamos el keyspace comvive

```
cqlsh -u juanjose -p juanjose 10.10.10.104 9042  
--cqlversion=3.4.4
```

```
use comvive
```

Realizamos la creación de tablas

```
CREATE TABLE empleados (  
    dni text,  
    nombre text,  
    apellidos text,  
    correo text,  
    puesto text,  
    contratacion timestamp,  
    PRIMARY KEY (contratacion,dni));
```

```
CREATE TABLE transportes (  
    tipo text,  
    origen text,  
    destino text,  
    duracion time,  
    PRIMARY KEY (tipo, duracion));
```

Caso Práctico

Poblamos la tabla empleados

```
INSERT INTO empleados (dni, nombre, apellidos, correo, puesto,
contratacion) VALUES ('49231685Y', 'Mario', 'Perez',
'mario@correo.com', 'marketing', '2017-02-15 15:00:00');
```

```
INSERT INTO empleados (dni, nombre, apellidos, correo, puesto,
contratacion) VALUES ('51968723Z', 'Jose', 'Marquez',
'jm@correo.com', 'ventas', '2017-05-22 12:30:30.000');
```

```
INSERT INTO empleados (dni, nombre, apellidos, correo, puesto,
contratacion) VALUES ('33968574L', 'Laura', 'Romero',
'lauritar@correo.com', 'atencion al cliente', '2017-05-22
14:12:10.020');
```

```
INSERT INTO empleados (dni, nombre, apellidos, correo, puesto,
contratacion) VALUES ('47859632H', 'Elisabet', 'Lopez',
'elilopez@correo.com', 'directora', '2017-06-22 14:59:00.00');
```

```
INSERT INTO empleados (dni, nombre, apellidos, correo, puesto,
contratacion) VALUES ('98765448A', 'Juan Jose', 'Lopez',
'jjlr@correo.com', 'gerente', '2018-02-28 19:45:00.060');
```


Caso Práctico

Poblamos la tabla transportes

```
INSERT INTO transportes (tipo, origen, destino, duracion) VALUES  
( 'tren', 'dos hermanas', 'el cañamo', '00:35:30');
```

```
INSERT INTO transportes (tipo, origen, destino, duracion) VALUES  
( 'tren', 'dos hermanas', 'la rinconada', '00:27:56');
```

```
INSERT INTO transportes (tipo, origen, destino, duracion) VALUES  
( 'tren', 'utrera', 'el cañamo', '01:10:00');
```

```
INSERT INTO transportes (tipo, origen, destino, duracion) VALUES  
( 'coche', 'sevilla', 'la rinconada', '00:25:00');
```

```
INSERT INTO transportes (tipo, origen, destino, duracion) VALUES  
( 'moto', 'sevilla', 'la rinconada', '00:26:00');
```

Caso Práctico

Paramos una máquina

```
poweroff
```

Comprobamos el estado del cluster

```
nodetool status
```

Realizamos un insert en otro nodo

```
INSERT INTO transportes (tipo, origen, destino, duracion) VALUES  
('bicicleta','la rinconada','el cañamo','00:03:00');
```

Iniciamos la máquina de nuevo y realizamos la reparación

```
nodetool repair -dcpar -full
```

Comprobamos el estado del cluster

```
nodetool status
```

Gracias a todos

¿Alguna duda?



cassandra