# 

[Introducción](#_udwc7mbyhcbx)

[CQL3](#_jrie4p6xszat)

[Keyspace](#_61fkva8kqrth)

[Creando una tabla:](#_a4ek8hytwmn9)

[Tipos de columnas](#_c9wjpxbagnm)

[MAP](#_haokrbcjxxe1)

[Creación](#_lnzusiyf5o06)

[Ingresar valores](#_9i700twi4xu0)

[Actualizar / Agregar](#_bi3a2cvllb9m)

[Consultar](#_l71clyy0d5c)

[Borrar](#_uwnn4aaz0u14)

[SET](#_moit9z66grho)

[Creación de un SET](#_yoeads3mjo4r)

[Ingresar valores](#_1o8vlf7w1b7z)

[Actualizar - Agregar o quitar elementos del SET](#_sfk2axglww42)

[LIST](#_op2dcoooebxh)

[Allow Filtering](#_ibjgk15af6cj)

# 

# 

# 

# **Introducción**

Cuando se trabaja con Cassandra Apache se procura eliminar casi por completo cualquier posibilidad de usar un *antipattern[[1]](#footnote-0)*. Cassandra ofrece una interfaz que sea apta para personas con experiencia en SQL.

(Nishant Neeraj 2013) nos recomienda tener en cuenta las siguientes consideraciones al momento de usar Apache Cassandra:

* Denormalizar, desnormalizar y desnormalizar: olvidar lo aprendido sobre 3FN[[2]](#footnote-1) en el paradigma relacional. Cuanto menor sea el tráfico por la red, mejor será el rendimiento. Desnormalice siempre que pueda para una recuperación más rápida.
* Una fila, una máquina: cada fila permanece en una máquina. Las filas no están fragmentadas entre los nodos. Así que ten cuidado con esto. Una fila de alta demanda puede crear un *hotspot[[3]](#footnote-2)*.
* De la consulta al modelo: a diferencia del RDBMS, donde se modelan la mayoría de las tablas con entidades en la aplicación y luego se ejecutan consultas analíticas para obtener datos, Cassandra no tiene tal disposición.

Por lo tanto, es posible que deba desnormalizar su modelo de tal manera que todas sus consultas queden limitadas a un montón de comandos simples como obtener, cortar, contar, grabar y algunas búsquedas simples indexadas.

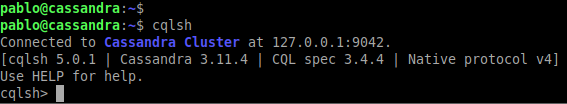
# **CQL3**

## **Keyspace**

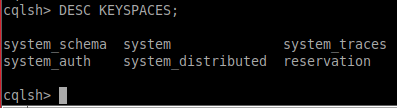
Un *keyspace* puede ser visto como el contenedor más externo para datos en Cassandra. Todos los datos en Cassandra deben estar dentro de un *keyspace*. Esto puede ser considerado como una base de datos en un RDBMS. En el caso de Cassandra, un *keyspace* es una colección de *familias de columnas*.

Sintaxis para conectarse a la shell de cassandra:

cqlsh

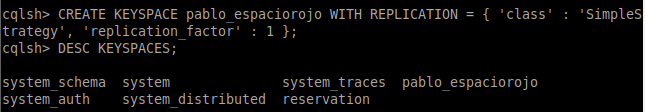


DESC KEYSPACES;

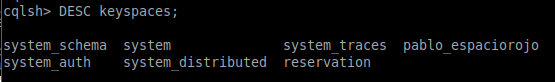


Sintaxis para creación:

CREATE KEYSPACE ***pablo\_espaciorojo*** WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication\_factor' : 1 };

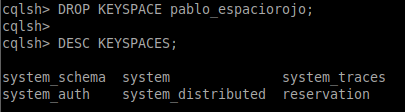


DESC KEYSPACES;

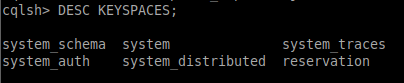


Sintaxis para borrar:

DROP KEYSPACE pablo\_espaciorojo;



DESC KEYSPACES;



## **Creando una tabla:**

Vemos un ejemplo con el [club atlético independiente:](https://es.wikipedia.org/wiki/Club_Atl%C3%A9tico_Independiente)

CREATE KEYSPACE *mundo\_rojo* WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication\_factor' : 1 };

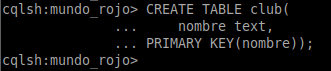


USE mundo\_rojo;

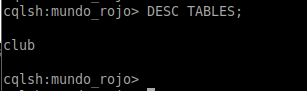


CREATE TABLE club(  
 nombre text,

PRIMARY KEY(nombre));



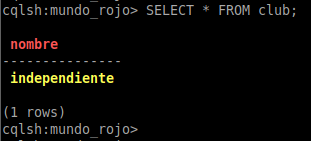
DESC TABLES;



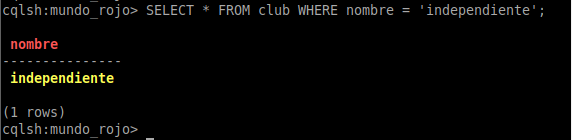
INSERT INTO club (nombre) VALUES(‘independiente’);



SELECT \* FROM *club*;



SELECT \* FROM club WHERE club=’independiente’



DROP TABLE club;



DESC TABLES;

# **Tipos de columnas**

## **MAP[[4]](#footnote-3)**

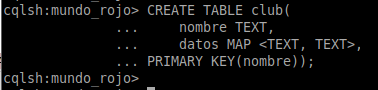
Colección con pares clave-valor. Un MAP es un tipo de datos que se utiliza para almacenar un par de elementos clave-valor.

### **Creación**

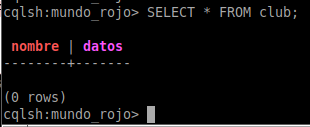
CREATE TABLE club(  
 nombre TEXT,

datos MAP <TEXT, **TEXT**>,

PRIMARY KEY(nombre));

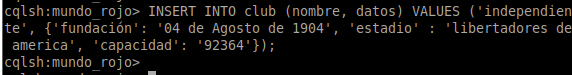


SELECT \* FROM club;

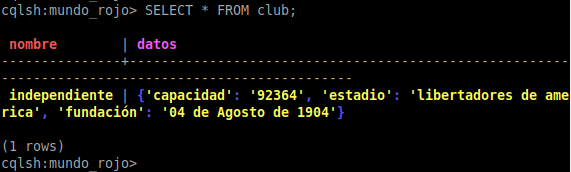


### **Ingresar valores**

INSERT INTO club VALUES ('independiente', **{**fundación: '04 de Agosto de 1904', 'estadio' : 'libertadores de america', 'capacidad': '92364'**}**);



SELECT \* FROM club;

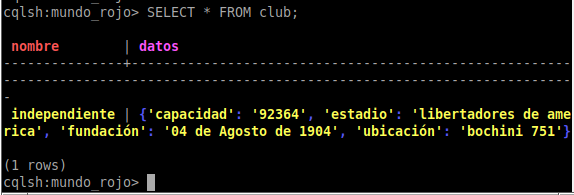


### **Actualizar / Agregar**

UPDATE club SET datos = datos + {'ubicación': 'bochini 751'} where nombre='independiente';

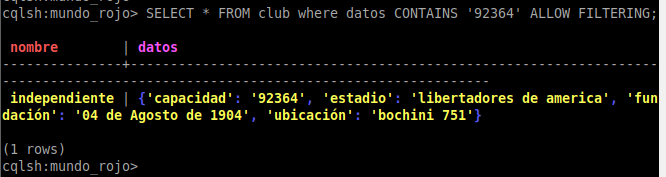


SELECT \* FROM club;



### **Consultar**

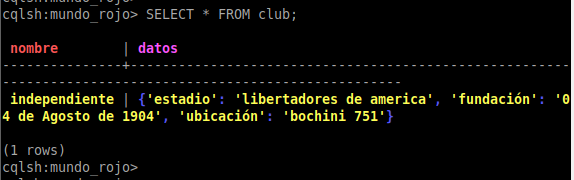
SELECT \* FROM club where datos CONTAINS '92364' ALLOW FILTERING;



### **Borrar**

DELETE datos['capacidad'] FROM club WHERE nombre = 'independiente';



SELECT \* FROM club;

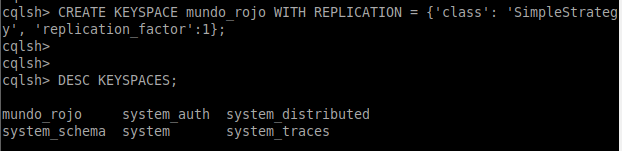
DROP TABLE club;



## **SET[[5]](#footnote-4)**

Colección de uno o más elementos ordenados.

CREATE KEYSPACE *mundo\_rojo* WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication\_factor' : 1 };





### **Creación de un SET**

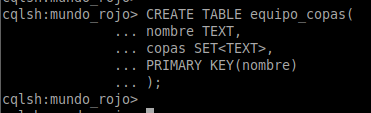
CREATE TABLE equipo\_copas (

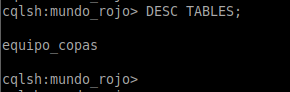
nombre TEXT,

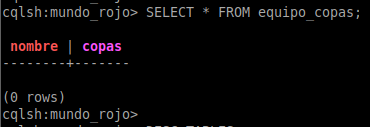
copas SET<TEXT>,

PRIMARY KEY (nombre)

);

****

****

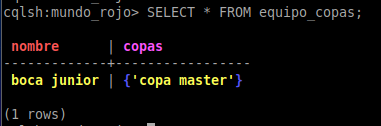
****

### **Ingresar valores**

INSERT INTO equipo\_copas (nombre, copas)

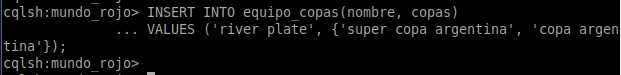
VALUES (‘boca junior’, { ‘Master’ } )

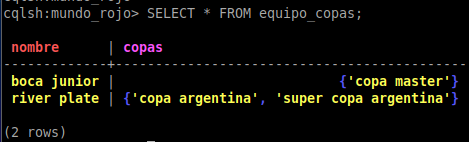




INSERT INTO equipo\_copas (nombre, copas)

VALUES (‘river plate’, { ‘super copa argentina’, ‘copa argentina’ } )

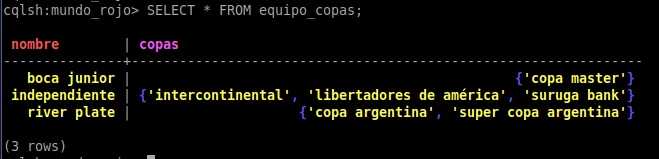




INSERT INTO equipo\_copas (nombre, copas)

VALUES (‘independiente’, { ‘suruga bank’, ‘intercontinental’, ‘libertadores de américa’} )



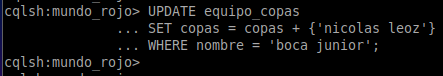


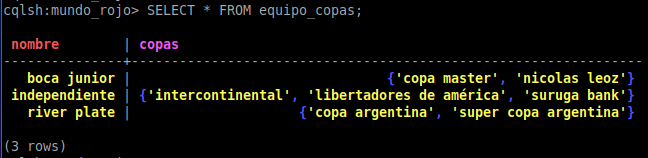
### **Actualizar - Agregar o quitar elementos del SET**

UPDATE equipo\_copas

SET copas = copas + { ‘nicolas leoz’ }

WHERE nombre = ‘boca junior’ ;

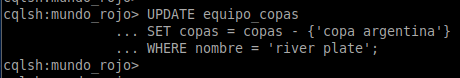


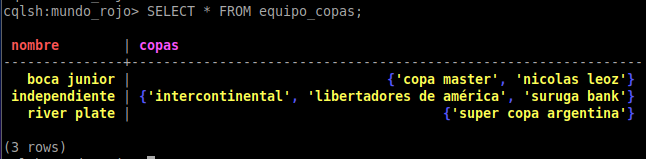


UPDATE equipo\_copas

SET copas = copas - { ‘copa argentina’ }

WHERE nombre = ‘river plate’ ;

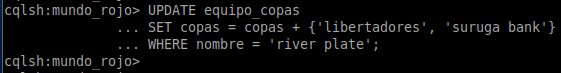


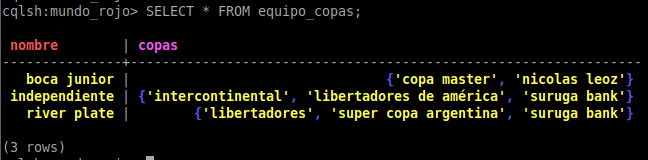


UPDATE equipo\_copas

SET copas = copas + { ‘libertadores’, ‘suruga bank’ }

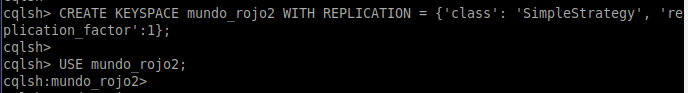
WHERE nombre = ‘river plate’ ;

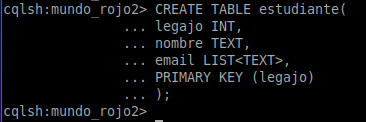


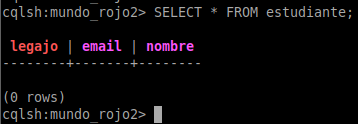


## LIST

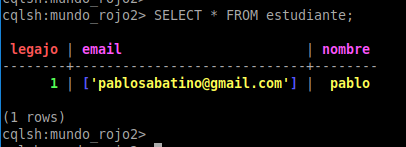
Colección de uno o más elementos.



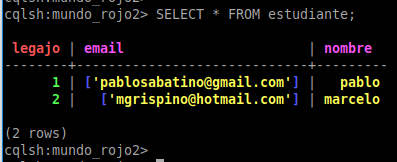




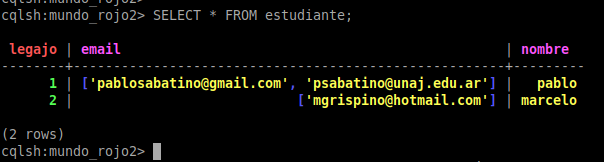






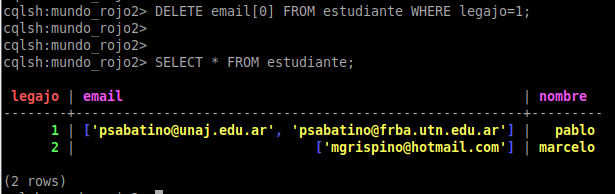


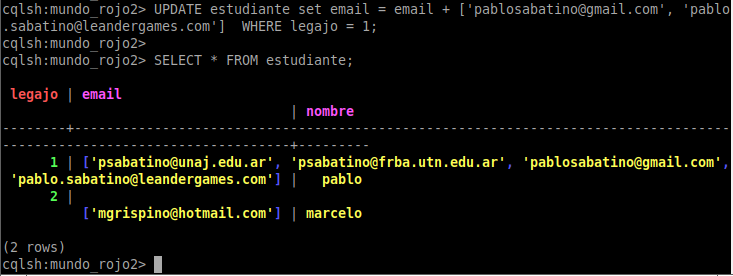


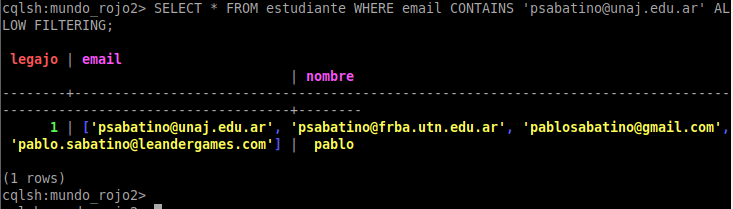


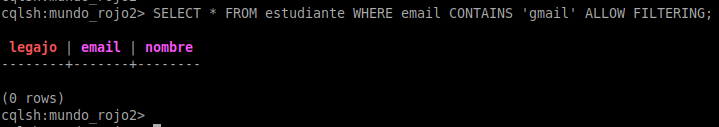




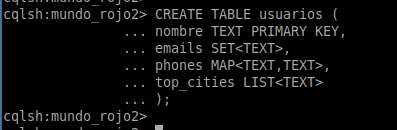


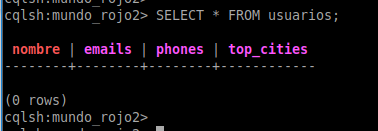


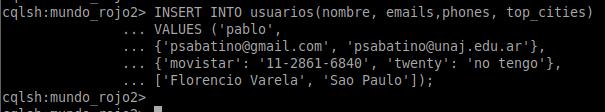


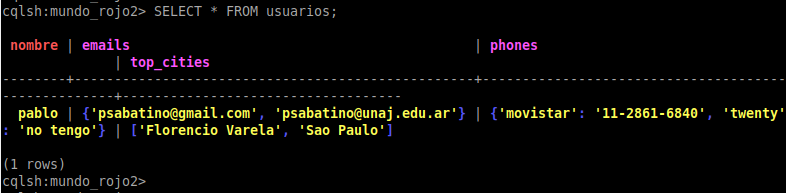


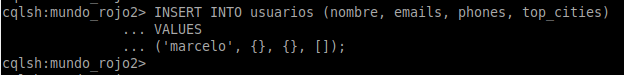
# Otro ejemplo:









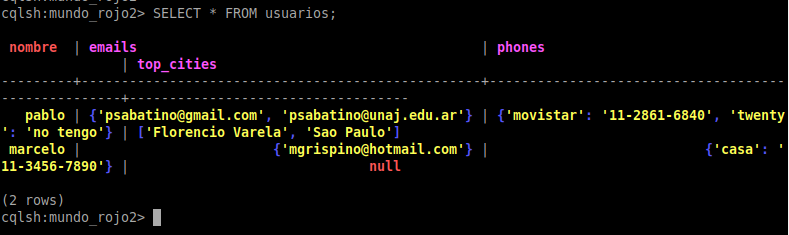




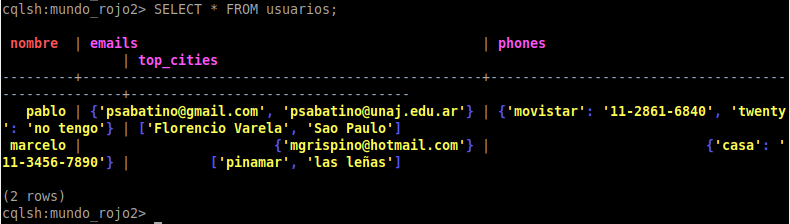






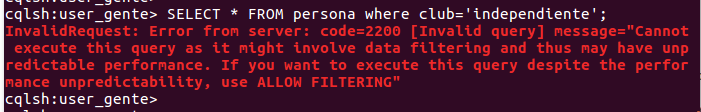






## Allow Filtering

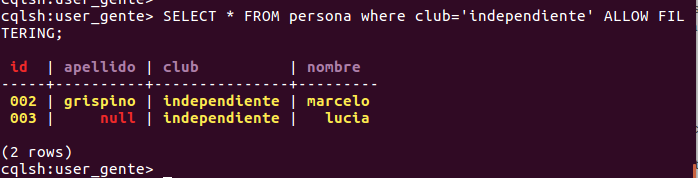
SELECT \* FROM persona WHERE club=’independiente’;



¿Qué es lo que sucede?

Cassandra sabe que podría no ser capaz de ejecutar la consulta de una manera eficiente. Por lo tanto, es una advertencia: "*Tene cuidado. Ejecutar esta consulta como tal podría no ser una buena idea, ya que puede usar muchos de sus recursos informáticos*".  
  
La única forma en la que Cassandra puede ejecutar esta consulta es recuperando todas las filas de la tabla y luego filtrando las que corresponden al club “independiente”.  
  
Si dicha tabla tiene, por ejemplo, 1 millón de filas y el 95% de ellas tiene el valor solicitado para la columna time1, la consulta seguirá siendo relativamente eficiente y deberá utilizar ALLOW FILTERING.  
  
Por otro lado, si la tabla contiene 1 millón de filas y solo 2 filas contienen el valor solicitado, la consulta resultará extremadamente ineficiente. Cassandra cargará 999, 998 filas para nada. Si la consulta se usa con frecuencia, probablemente sea mejor agregar un índice en la columna club.  
  
Desafortunadamente, Cassandra no tiene manera de diferenciar entre los 2 casos anteriores, ya que dependen de la distribución de datos de la tabla. Por lo tanto, Cassandra te está advirtiendo y confiando para que tomes la mejor decisión.

SELECT \* FROM persona WHERE club=’independiente’ ALLOW FILTERING;



Borramos la tabla:

DROP TABLE persona;

1. <https://es.wikipedia.org/wiki/Antipatr%C3%B3n_de_dise%C3%B1o> Disponible Mayo 2019. [↑](#footnote-ref-0)
2. <https://es.wikipedia.org/wiki/Tercera_forma_normal> Disponible Mayo 2019 [↑](#footnote-ref-1)
3. <https://es.wikipedia.org/wiki/Hotspot_(telecomunicaciones)> Disponible Mayo 2019 [↑](#footnote-ref-2)
4. <https://docs.datastax.com/en/cql/3.3/cql/cql_using/useMap.html> Disponible Mayo 2019 [↑](#footnote-ref-3)
5. <https://docs.datastax.com/en/cql/3.3/cql/cql_using/useSet.html> Disponible Junio 2019 [↑](#footnote-ref-4)