Bases de Datos Clave-valor: Riak

Master Big Data y Business Analytics****

**Asignatura**: Bases de Datos No Convencionales

**Módulo**: Tecnologías de Big Data/Gestión de Datos

**Coordinador**: Elena García Barriocanal, [elenagarcia@campusciff.net](mailto:elenagarcia@campusciff.net)

**Profesores:** Jordi Conesa [jordiconesa@campusciff.net](mailto:jordiconesa@campusciff.net)

Elena García Barriocanal, [elenagarcia@campusciff.net](mailto:elenagarcia@campusciff.net)

Contenido

Arrancar el servicio de Riak 2

Comprobar que Riak está funcionando 2

Comprobar que Riak puede ejecutar operaciones de lectura/escritura 2

Parar el servicio Riak 2

Consultar elementos en Riak 3

Inserción de objetos 4

Consulta de objetos 5

Eliminación de objetos 6

Modificación de objetos 7

Link Walking 8

Crear índices 9

Comandos generales

Al arrancar ejecutar el comando “setxkbmap –layout es” para configurar el teclado en castellano.

# Arrancar el servicio de Riak

sudo riak start

# Comprobar que Riak está funcionando

sudo riak ping

pong

# Comprobar que Riak puede ejecutar operaciones de lectura/escritura

sudo riak-admin test

Successfully completed 1 read/write cicle to ‘riak@URL’

# Parar el servicio Riak

sudo riak stop

Operaciones básicas

# Consultar elementos en Riak

Supongamos que tenemos una base de datos de vehículos que contiene vehículos y sus propietarios.

Supongamos que queremos consultar el vehículo con matrícula GI-1654-AZ.

Deberíamos ejecutar el siguiente comando:

curl -X GET http://localhost:8098/riak/vehicle/GI-1654-AZ

La respuesta es:

Not found

Ahora ejecutad el comando añadiendo el parámetro “v” (verbose)

curl -v -X GET http://localhost:8098/riak/vehicle/GI-1654-AZ

El elemento no se encuentra en la base de datos. Entonces añadiremos el vehículo en la base de datos.

# Inserción de objetos

Ahora añadiremos algunos vehículos en el bucket vehículos de la base de datos:

|  |  |
| --- | --- |
| Clave | Valor |
| GI-1654-AZ | Coche de Juan |
| M-5894-KL | {color:”azul”, marca:”Ford”, modelo:”Escort GTX 2.0”, caballos:125, año:2005, propietario:”Pablo Marmol”} |
| <desconocida> | Coche de María. Información sobre la matrícula solicitada pero no recibida aún. |
| CC-10070 | Coche de cuerpo consular, país=Zamundia |
| 5687 BBZ | {color:”rojo”, marca:”Citroen”, modelo:”AX 1.6”, caballos:65, año:1992, propietario:”Juan Valdés”} |
| Añadir vuestro coche |  |

Añadiremos el primer coche sin tener en cuenta su tipo de datos:

curl -v -X PUT http://localhost:8098/riak/vehicle/GI-1654-AZ \

-d ‘Coche de Juan’

Posteriormente, añadiremos el segundo coche. En este caso, los datos a añadir son un documento JSON. Por tanto, indicaremos el formato de datos de entrada.

curl -v -X PUT http://localhost:8098/riak/vehicle/M-5894-KL \

-H "Content-Type: application/json" \

-d ' {color:”azul”, marca:”Ford”, modelo:”Escort GTX 2.0”, caballos:125, año:2005, propietario:”Pablo Marmol”}'

Para la tercera entrada no disponemos de clave. En este caso podríamos crear una clave forzada o añadir el objeto sin indicar una clave y hacer que Riak nos facilite una. Haremos esto segundo:

curl -v -X POST http://localhost:8098/riak/vehicle \

-d 'Coche de María. Información sobre la matrícula solicitada pero no recibida aún.'

Crear en la base de datos los objetos de las tres entradas restantes. Crear también una entrada con tus datos en formato JSON.

# Consulta de objetos

Ahora que ya tenemos objetos en la base de datos podemos consultar los objetos por su clave. Por ejemplo, podemos realizar las siguientes consultas:

curl -X GET http://localhost:8098/riak/vehicle/GI-1654-AZ

curl -X GET http://localhost:8098/riak/vehicle/CC-10070

A continuación consultad los otros 4 objetos creados en la base de datos.

Si queremos consultar los diferentes buckets de Riak o las claves de un determinado bucket podemos usar los siguientes comandos:

curl –X GET <http://localhost:8098/buckets?buckets=true>

curl –X GET <http://localhost:8098/buckets/vehicle/keys?keys=true>

# Eliminación de objetos

Supongamos que queremos eliminar el vehículo con matrícula CC-10070. Realizad la siguiente operación en la base de datos:

curl -i -X DELETE http://localhost:8098/riak/vehicle/CC-10070

# Modificación de objetos

Supongamos que nos han llegado más datos del coche con matrícula GI-1654-AZ y queremos modificar su valor en la base de datos por el siguiente:

{color:”rojo”, marca:”Ferrari”, modelo:”F40”, propietario:”Juan Pujol”}

Ejecutaríamos el siguiente comando:

curl -v -X PUT http://localhost:8098/riak/vehicle/GI-1654-AZ \

-H "Content-Type: application/json" \

-d ' {color:”rojo”, marca:”Ferrari”, modelo:”F40”, propietario:”Juan Pujol”}'

Una vez llegados a este punto, supongamos que queremos evolucionar nuestra base de datos para tener información también de los propietarios de los coches y de qué coche tiene cada propietario. ¿Cómo modificaríais la base de datos para soportar este nuevo requisito? (cabe tener en cuenta que todos los propietarios residen en España).

TEXTO OCULTO, DESBLOQUEALO CUANDO TENGAS UNA RESPUESTA

Una posibilidad sería añadir un nuevo tipo de colección (bucket) a la base de datos llamada “propietarios” (o owners), que contenga los datos de los propietarios del vehículo. Las relaciones entre propietarios y sus coches se podrían indicar mediante un *link* de Riak.

Por tanto podríamos crear el siguiente objeto:

curl -v -X PUT http://localhost:8098/riak/owner/46254247P \

-H "Content-Type: application/json” \

-H ‘Link: </riak/vehicle/GI-1654-AZ>; riaktag=”ownedCar"’ \

-d '{name: ”Juan Pujol”, bornAt: 1972}'

Crear los siguientes objetos en la base de datos y asignarlos a los vehículos indicados:

|  |  |  |
| --- | --- | --- |
| Clave | Valor | Vehículo |
| 38882114A | Pablo Marmol | M-5894-KL  CC-10070 |
| 15254897D | Juan Valdés | 5687 BBZ |
| Vosotros |  |  |

Cread también una entrada con vuestros datos en formato textual.

Operaciones avanzadas

# Link Walking

En este punto ya tenemos una base de datos de juguete con dos buckets relacionados entre sí. Recordad que las relaciones (o links) son unidireccionales, por tanto sólo se pueden consultar en un sentido.

Se pide consultar los datos del propietario con DNI 38882114A.

curl -X GET http://localhost:8098/riak/owner/38882114A

Ahora se pide utilizar *link walking* para obtener los coches propiedad del propietario con DNI 38882114A.

curl -X GET http://localhost:8098/riak/owner/38882114A/vehicle, ownedCar, \_

Probad con distintas combinaciones de los parámetros <bucket, tag, keep>. Después ejecutad la consulta que permite obtener vuestros coches. Platearos porque en todos los casos se devuelven los mismos valores.

TEXTO OCULTO, MOSTRARLO CUANDO TENGAS UNA RESPUESTA

Los valores son siempre los mismos porque no hay más relaciones en la base de datos. Por ejemplo, en caso de que el propietario con DNI 38882114A tuviera otro tipo de relación de tipo “hermano” con otro propietario, esta aparecería cuando no forzáramos el tipo esperado ni el nombre de la relación (\_,\_,?). En el caso de que quisiéramos que se mostrará la relación de hermano, podríamos hacerlo forzando el tipo de relación (\_, brother, \_) o forzando el tipo esperado (owner, \_, \_).

Os animo a que deis de alta una relación brother para el propietario con DNI 38882114A para comprobar lo que se ha comentado.

# Crear índices

Supongamos que queremos crear índices secundarios sobre los vehículos para buscar vehículos en función de la última fecha en que pasaron la ITV. ¿Cómo modificaríais el esquema actual para definir este tipo de índices?

Ver <http://docs.basho.com/riak/latest/dev/using/2i/> para más información sobre cómo usar índices secundarios.

curl -v -X PUT http://localhost:8098/riak/vehicle/GI-1654-AZ \

-H "x-riak-index-ITV\_int: 2011" \

-d 'Coche de Juan'

curl -v -X PUT http://localhost:8098/riak/vehicle/M-5894-KL \

-H "x-riak-index-ITV\_int: 2013" \

-H "Content-Type: application/json" \

-d ' {color:”azul”, marca:”Ford”, modelo:”Escort GTX 2.0”, caballos:125, año:2005, propietario:”Pablo Marmol”}'

Los índices se definen sobre los metadatos y no sobre los datos ¿Qué implicaciones creéis que tiene este hecho? La respuesta se encuentra en las trasparencias de la sesión de Riak.

Para buscar los objetos en función del índice podemos usar el siguiente comando:

curl –X GET http://localhost:8098/buckets/<nombre bucket>/index/<nombre índice con sufijo>/<valor a buscar o rango de valores>

Así pues, para buscar los vehículos que pasaron la ITV en el 2011 generaríamos la siguiente consulta:

curl –X GET http://localhost:8098/buckets/vehicle/index/ITV\_int/2011

Por otro lado, para consultar los coches que pasaron la ITV entre el 2011 y el 2014 generaríamos la siguiente consulta:

curl localhost:8098/types/mytype/buckets/vehicle/index/ITV\_int/2011/2014

NOTA: Para que funcionen los índices secundarios es necesario cambiar el motor de almacenamiento por defecto de Riak (Bitcask) por los motores de memória (Memory) o *LevelDB*. Esto es debido a que el sistema de almacenamiento por defecto esta optimizado para el modelo clave-valor pero no para el almacenamiento y gestión de indices.