Trabajo para la evaluación de la asignatura NoSQL

Máster Big Data Analytics - Marzo 2018

David Sánchez, Adrián Díaz y Joan Buigues

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

2. Introducción

En la actualidad existen múltiples plataformas dedicadas a la reserva de apartamentos turísticos y, como consecuencia de ello, una elevada competitividad en el sector.

En estas condiciones, analizar las principales métricas del negocio para tratar de optimizarlas y buscar la máxima rentabilidad resultará clave para cualquier empresa que se dedique a ello.

Por tanto, vamos a plantear un modelo de datos para que empresas como “ApartaRent“ puedan extraer conocimiento de sus datos y mejorar la toma de decisiones.

2.1 Diseño conceptual del modelo de datos

El modelo contará con las entidades:

- Cliente
- Reserva
- Alojamiento

2.2 Diseño agregado del modelo de datos

Tabla de Cliente

- ID cliente
- Nombre
- Apellidos
- Edad
- Población
- País
- Email
- Teléfono
- Newsletter

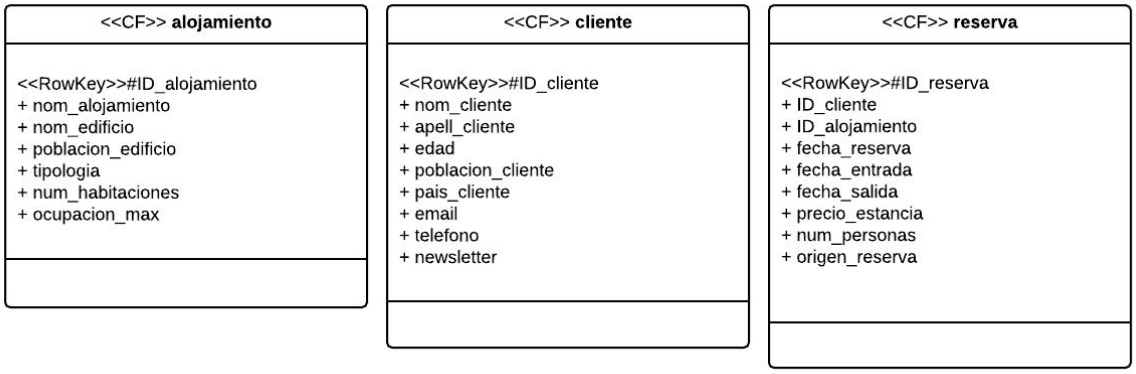
Tabla de Reserva

- ID reserva
- ID alojamiento
- ID cliente
- Fecha de reserva
- Fecha de entrada
- Fecha de salida
- Precio de estancia
- PAX
- Origen

Tabla de Alojamiento

- ID alojamiento
- Nombre del alojamiento
- Nombre del edificio
- Población del edificio
- Tipología
- Nº habitaciones
- Ocupación máxima

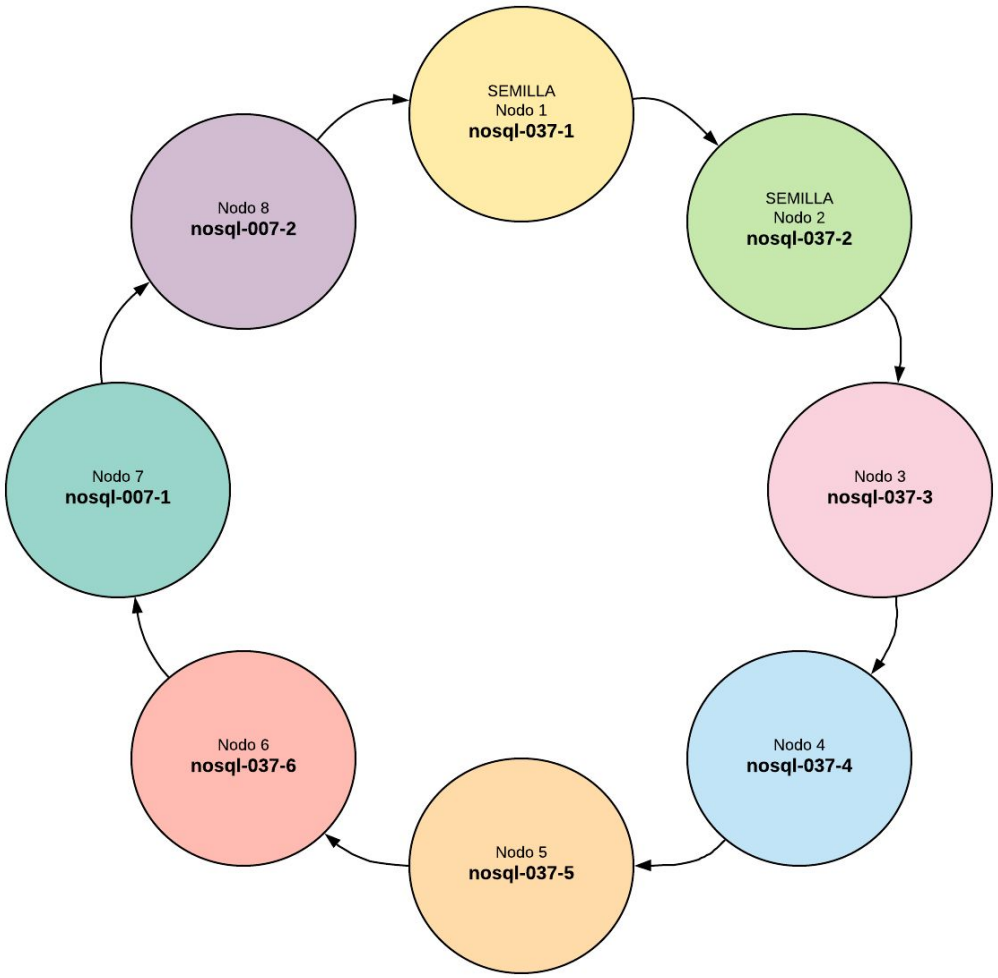
3. Implementación en Cassandra



Para desplegar el cluster de Cassandra vamos a utilizar las siguientes 8 máquina virtuales:

- nosql-037-1.dsic.cloud
- nosql-037-2.dsic.cloud
- nosql-037-3.dsic.cloud
- nosql-037-4.dsic.cloud
- nosql-037-5.dsic.cloud
- nosql-037-6.dsic.cloud
- nosql-007-1.dsic.cloud
- nosql-007-2.dsic.cloud

Siendo **nosql-037-1** y **nosql-037-2** los nodos que actúan como semilla.



Configuración del fichero cassandra.yaml

Todos los nodos tienen la misma configuración definida en el fichero “cassandra.yaml” y replicado en cada uno de ellos. De este fichero, se han modificado los siguientes parámetros:

```
-cluster_name: nosql-037
-seed: “nosql-037-1.dsic.cloud, nosql-037-2.dsic.cloud”
-data_file_directories: /var/lib/cassandra/cluster_lnosql-037/data
-commitlog_directory: /var/lib/cassandra/cluster_lnosql-037/commitlog
-saved_caches_directory: /var/lib/cassandra/
cluster_lnosql-037/saved_caches
-commitlog_total_space_in_mb: 1024
```

Comprobaciones

Para asegurarnos que tenemos el cluster correctamente funcionando, ejecutamos en el terminal la orden `nodetool status`

```
[root@NOSQL-037-1 ~]# nodetool status
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address         Load         Tokens       Owns   Host ID
      Rack
UJ  192.168.231.43   14,43 KB   256          ?
b25509c1-32d0-4aa9-8ea7-77b2187143e2  rack1
UN  192.168.231.44   1,09 MB     256          ?
3b78fe70-c14c-454d-8945-4e03a5251e94  rack1
UN  192.168.231.223  3,17 MB     256          ?
58cd088e-38e7-4e36-97ed-cb1a4b857656  rack1
UN  192.168.231.224  2,61 MB     256          ?
0f74d397-9e78-48d3-a1ae-d96a15206b5e  rack1
UN  192.168.231.225  1,79 MB     256          ?
0dacf412-e886-4420-976c-ec30e6e8935e  rack1
UN  192.168.231.226  3,05 MB     256          ?
a8f0fbf4-fb39-4032-9602-a361d7c45f21  rack1
UN  192.168.231.227  1,73 MB     256          ?
83670d27-bfb3-4257-ab9f-37292bf4c33a  rack1
UN  192.168.231.228  2,11 MB     256          ?
bfdd4f1e-0986-4856-90f0-1f3f13f9dba4  rack1

Note: Non-system keyspaces don't have the same replication settings, effective
ownership information is meaningless
```

3.1 Creación de la base de datos en Cassandra

```
CREATE KEYSPACE alquilerbd WITH replication = { 'class' : 'SimpleStrategy',
'replication_factor' : 3 };
```

Creación de las tablas

Creamos las tablas con las entidades y claves mencionadas anteriormente:

```
CREATE TABLE alquilerbd.alojamiento(id_alojamiento uuid, nom_alojamiento text,
nom_edificio text, poblacion_edificio text, tipologia text, num_habitaciones int,
ocupacion_max int, propietario text,  canon_propietario float, PRIMARY KEY
(id_alojamiento));
```

```
CREATE TABLE alquilerbd.cliente(id_cliente uuid, nom_cliente text, apell_cliente
text, edad int, poblacion_cliente text, pais_cliente text, email text, telefono
text, newsletter boolean, PRIMARY KEY (id_cliente, edad));
```

```
CREATE TABLE alquilerbd.reserva(id_reserva uuid, id_cliente uuid, id_alojamiento
uuid, nom_alojamiento text, fecha_reserva date, fecha_entrada date, fecha_salida
date, precio_estancia float, num_personas int, origen_reserva text, canal text,
gasto_origen float, PRIMARY KEY (Id_reserva, id_cliente, id_alojamiento));
```

3.2 Caso de uso: “Datos básicos de ApartaRent”

Queremos responder una serie de preguntas básicas para conocer el comportamiento de los clientes de ApartaRent. Y partiremos de datos guardados en unos archivos CSV, uno para cada entidad.

Importamos los datos de estos archivos:

```
copy alquilerbd.alojamiento
(id_alojamiento,canon_propietario,nom_alojamiento,nom_edificio,num_habitaciones,ocup
acion_max,poblacion_edificio,propietario,tipologia)
FROM 'csv_alojamiento.csv' WITH DELIMITER=',' AND HEADER = TRUE;
```

```
copy alquilerbd.cliente
(id_cliente,nom_cliente,apell_cliente,edad,poblacion_cliente,pais_cliente,email,tele
fono,newsletter)
FROM 'csv_cliente.csv' WITH DELIMITER=',' AND HEADER = TRUE;
```

```
copy alquilerbd.reserva (id_reserva,id_cliente,
Id_alojamiento,nom_alojamiento,fecha_reserva,fecha_entrada,fecha_salida,precio_estan
cia,num_personas,origen_reserva,canal,gasto_origen)
FROM 'csv_reserva.csv' WITH DELIMITER=',' AND HEADER = TRUE;
```

Índices

Después de crear las tablas, necesitamos establecer índices en cada columna para poder buscar en dicha columna. Por ejemplo:

```
CREATE INDEX "nom_edificioid" ON alquilerbd.alojamiento(nom_edificio);
```

Y buscamos un alojamiento concreto para comprobar que la semilla ha quedado definida correctamente:

```
SELECT * FROM alojamiento WHERE nom_edificio = 'PUEBLA MARINA' ;
```

Cuestión 1: Identificar los clientes que entran el día 1 de julio de 2018.

```
CREATE INDEX fecha_entrada_idx ON alquilerbd.reserva(fecha_entrada);

SELECT id_cliente FROM reserva WHERE fecha_entrada = '2018-07-01' ;

id_cliente
--------------------------------------
 8a0310e4-2172-11e8-b467-0ed5f89f718b
 8a031ca6-2172-11e8-b467-0ed5f89f718b
 8a029056-2172-11e8-b467-0ed5f89f718b
 8a02979a-2172-11e8-b467-0ed5f89f718b
 8a031e36-2172-11e8-b467-0ed5f89f718b
 8a029506-2172-11e8-b467-0ed5f89f718b
```

```
SELECT * FROM cliente WHERE id_cliente IN (8a0310e4-2172-11e8-b467-0ed5f89f718b,
8a031ca6-2172-11e8-b467-0ed5f89f718b,  8a029056-2172-11e8-b467-0ed5f89f718b,
8a02979a-2172-11e8-b467-0ed5f89f718b, 8a031e36-2172-11e8-b467-0ed5f89f718b,
```

```
8a029506-2172-11e8-b467-0ed5f89f718b) ;

 id_cliente                           | apell_cliente | edad | email
      | newsletter | nom_cliente | pais_cliente | poblacion_cliente | telefono
--------------------------------------+---------------+------+---------------------------
--------+------------+-------------+--------------+-------------------+-----------
 8a029056-2172-11e8-b467-0ed5f89f718b |       kulawy |   20 |
marek.kulawy@gmail.com |  False |      marek |  Reino Unido |    Costa Teguise |
600666123
 8a029506-2172-11e8-b467-0ed5f89f718b |       Bower |   21 |
Brian.Bower@gmail.com |   False |      Brian |  Reino Unido |    Costa Teguise |
600666124
 8a02979a-2172-11e8-b467-0ed5f89f718b |       Smart |   22 |
Lee.Smart@gmail.com |     False |      Lee |  Reino Unido |            Moraira |
600666125
 8a0310e4-2172-11e8-b467-0ed5f89f718b |       Bowes |   44 |
Tracy.Bowes@gmail.com |   False |      Tracy |  Reino Unido |    Costa Teguise |
600666193
 8a031ca6-2172-11e8-b467-0ed5f89f718b | SuárezGallego |   24 |
Inma.SuárezGallego@gmail.com |   False |      Inma |       España |  Pobla de
Farnals | 600666201
 8a031e36-2172-11e8-b467-0ed5f89f718b |       Renzi |   25 |
Rolando.Renzi@gmail.com |        False |      Rolando |    Italia |
Valencia | 600666202
```

De esta manera **obtenemos la lista de clientes** que entran el 1 de julio de 2018 y así podemos mandarles un recordatorio y confirmar la reserva.

Cuestión 2: ¿Predominan clientes nacionales o extranjeros?

```
// creamos índice pais_cliente
CREATE INDEX pais_clienteidx ON alquilerbd.cliente(pais_cliente);

// ver los paises
SELECT pais_cliente FROM cliente;


// contamos los totales
cqlsh:alquilerbd> SELECT COUNT(pais_cliente) FROM cliente;

 system.count(pais_cliente)
----------------------------
                         89
```

```
// contamos los que tiene españa
SELECT COUNT(pais_cliente) FROM cliente WHERE pais_cliente = 'España' ;
 system.count(pais_cliente)
----------------------------
                         51
```

Después de realizar estas consultas, sabemos que hay 89 países diferentes de donde proceden los clientes de ApartaRent. Y de España sabemos que son 51, por tanto, podemos concluir que **los clientes nacionales son mayoritarios**.

Cuestión 3: ¿Qué franja de edad es la que más reserva?

```
CREATE INDEX edadidx ON alquilerbd.cliente(edad);

SELECT COUNT (id_cliente) FROM cliente WHERE edad > 20  AND edad <= 30 ALLOW
FILTERING;

 system.count(id_cliente)
--------------------------
                      29
```

```
 SELECT COUNT (id_cliente) FROM cliente WHERE edad > 30  AND edad <= 40  ALLOW
FILTERING;

 system.count(id_cliente)
--------------------------
                      14
```

```
SELECT COUNT (id_cliente) FROM cliente WHERE edad > 40  AND edad <= 50 ALLOW
FILTERING;

 system.count(id_cliente)
--------------------------
                      35
```

```
SELECT COUNT (id_cliente) FROM cliente WHERE edad > 50  AND edad <= 60 ALLOW
FILTERING;

 system.count(id_cliente)
--------------------------
                         8
```

Después de consultar el número de reservas que realiza cada franja de edad (separando en grupos de 10 años), podemos concluir que el grupo comprendido **entre 40 y 50 años** es el que **más reservas realiza** para esta muestra de datos.

Cuestión 4: ¿Mediante qué canal se factura más?

```
cqlsh:alquilerbd> CREATE INDEX canalidx ON alquilerbd.reserva(canal);


cqlsh:alquilerbd> SELECT SUM (precio_estancia) FROM reserva;

 system.sum(precio_estancia)
-----------------------------
                   41735.625
```

```
cqlsh:alquilerbd> SELECT SUM (precio_estancia) FROM reserva WHERE canal = 'WEB';

 system.sum(precio_estancia)
-----------------------------
                        2533
```

```
cqlsh:alquilerbd> SELECT SUM (precio_estancia) FROM reserva WHERE canal =
'COMERCIAL';

 system.sum(precio_estancia)
-----------------------------
                   5157.97998
```

```
cqlsh:alquilerbd> SELECT SUM (precio_estancia) FROM reserva WHERE canal = 'OTA';

 system.sum(precio_estancia)
-----------------------------
             34044.64844
```

Sacamos el valor de todas las estancias de la muestra por canal de facturación y llegamos a la conclusión de que **OTA** (Online Travel Agencies) **es el canal que más factura**, con 34.044, 65€.

4. Implementación en MongoDB

4.1 Despliegue de la infraestructura de MongoDB

Para desplegar el cluster de MongoDB disponemos de 12 equipos virtuales (NOSQL-007-X y NOSQL-028-X, donde X es un número del 1 al 6).

El cluster que hemos desplegado consta de:

- **7 particiones** con un **factor de replicación de 2** (es decir, los datos se guardarán igualmente en un primario y un secundario, a los que añadiremos un árbitro).

Para facilitar la convivencia de diferentes servicios en una misma máquina, los primeros se han desplegado en diferentes puertos. De esta manera:

- Los servicios primarios se han instalado en el puerto 27001
- Los secundarios en el puerto 28001
- Los árbitros en los puertos 29001
- Los servicios de configuración en el puerto 26001
- Los únicos servicios que respetan los puertos por defecto son los servicios "mongos", que sirven para aceptar las peticiones de los usuarios a través del puerto 27017

La arquitectura del cluster será la siguiente:

- **Conjunto de réplicas "tarea_s1":**
  - NOSQL-007-1.dsic.cloud: primario del conjunto. mongod: 27001 shard_s2.conf
  - NOSQL-028-1.dsic.cloud: secundario del conjunto. mongod: 28001 shard_s2.conf
  - NOSQL-028-2.dsic.cloud: arbitro del conjunto. mongod: 29001 arb_s2.conf

- **Conjunto de réplicas "tarea_s2":**
  - NOSQL-007-2.dsic.cloud: primario del conjunto. mongod: 27001 shard_s2.conf
  - NOSQL-028-2.dsic.cloud: secundario del conjunto. mongod: 28001 shard_s2.conf
  - NOSQL-028-3.dsic.cloud: arbitro del conjunto. mongod: 29001 arb_s2.conf

- **Conjunto de réplicas "tarea_s3":**
  - NOSQL-007-3.dsic.cloud: primario del conjunto. mongod: 27001 shard_s3conf
  - NOSQL-028-3.dsic.cloud: secundario del conjunto. mongod: 28001 shard_s3.conf
  - NOSQL-028-4.dsic.cloud: arbitro del conjunto. mongod: 29001 arb_s3.conf

- **Conjunto de réplicas "tarea_s4":**
  - NOSQL-007-4.dsic.cloud: primario del conjunto. mongod: 27001 shard_s4.conf
  - NOSQL-028-4.dsic.cloud: secundario del conjunto. mongod: 28001 shard_s4.conf
  - NOSQL-028-5.dsic.cloud: arbitro del conjunto. mongod: 29001 arb_s4.conf

- **Conjunto de réplicas "tarea_s5":**
  - NOSQL-007-5.dsic.cloud: primario del conjunto. mongod: 27001 shard_s5.conf

    - NOSQL-028-5 dsic.cloud: secundario del conjunto. mongod: 28001 shard_s5.conf
    - NOSQL-028-6.dsic.cloud: arbitro del conjunto. mongod: 29001 arb_s5.conf

- **Conjunto de réplicas “tarea_s6”:**
    - NOSQL-007-6.dsic.cloud: primario del conjunto. mongod: 27001 shard_s6.conf
    - NOSQL-028-6.dsic.cloud: secundario del conjunto. mongod: 28001 shard_s6.conf
    - NOSQL-028-1.dsic.cloud: arbitro del conjunto. mongod: 29001 arb_s6.conf

- **Conjunto de réplicas “tarea_s7”:**
    - NOSQL-028-6.dsic.cloud: primario del conjunto. mongod: 27001 shard_s7.conf
    - NOSQL-007-6.dsic.cloud: secundario del conjunto. mongod: 28001 shard_s7.conf
    - NOSQL-007-5.dsic.cloud: arbitro del conjunto. mongod: 29001 arb_s7.conf

- **Conjunto de réplicas de servidores de configuración “cfg”:**
    - NOSQL-007-1.dsic.cloud: mongod: 26001 cfg.conf
    - NOSQL-007-3.dsic.cloud: mongod: 26001 cfg.conf
    - NOSQL-007-5.dsic.cloud: mongod: 26001 cfg.conf

- **Servicios “mongos”**
    - En todas las máquinas. mongos:27017 mongos.conf

Estructura de directorios en cada máquina (todos son a partir del directorio home del usuario “mongod”:

- **NOSQL-007-1.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s1/** (servidor primario de la partición tarea_s1; puerto 27001)
    - **/var/lib/mongo/tarea/cfg/** (servidor de configuración mongod de conjunto de réplicas tarea; puerto 26001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-007-2.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s2/** (servidor primario de la partición tarea_s2; puerto 27001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-007-3.dsic.cloud:**
    - **/var/lib/mongo/tarea/shard_s3/** (servidor primario de la partición tarea_s3; puerto 27001)
    - **/var/lib/mongo/tarea/cfg/** (servidor de configuración mongod de conjunto de réplicas tarea; puerto 26001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-007-4.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s4/** (servidor primario de la partición tarea_s4; puerto 27001)

    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-007-5.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s5/** (servidor primario de la partición tarea_s5; puerto 27001)
    - **/var/lib/mongo/tarea/arb_s7/** servidor mongod (árbitro) de la partición tarea_s7; puerto 29001)
    - **/var/lib/mongo/tarea/cfg/** (servidor de configuración mongod de conjunto de réplicas tarea; puerto 26001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-007-6.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s6/** (servidor primario de la partición tarea_s6; puerto 27001)
    - **/var/lib/mongo/tarea/shard_s7/** (servidor secundario de la partición tarea_s7; puerto 28001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-028-1.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s1/** (servidor secundario de la partición tarea_s1; puerto 28001)
    - **/var/lib/mongo/tarea/arb_s6/** servidor mongod (árbitro) de la partición tarea_s6; puerto 29001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-028-2.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s2/** (servidor secundario de la partición tarea_s2; puerto 28001)
    - **/var/lib/mongo/tarea/arb_s1/** servidor mongod (árbitro) de la partición tarea_s1; puerto 29001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-028-3.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s3/** (servidor secundario de la partición tarea_s3; puerto 28001)
    - **/var/lib/mongo/tarea/arb_s2/** servidor mongod (árbitro) de la partición tarea_s2; puerto 29001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-028-4.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s4/** (servidor secundario de la partición tarea_s4; puerto 28001)
    - **/var/lib/mongo/tarea/arb_s3/** servidor mongod (árbitro) de la partición tarea_s3; puerto 29001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-028-5.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s5/** (servidor secundario de la partición tarea_s5; puerto 28001)
    - **/var/lib/mongo/tarea/arb_s4/** servidor mongod (árbitro) de la partición tarea_s4; puerto 29001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

- **NOSQL-028-6.dsic.cloud**:
    - **/var/lib/mongo/tarea/shard_s7/** (servidor primario de la partición tarea_s7; puerto 27001)
    - **/var/lib/mongo/tarea/shard_s6/** (servidor secundario de la partición tarea_s6; puerto 28001)
    - **/var/lib/mongo/tarea/arb_s5/** servidor mongod (árbitro) de la partición tarea_s5; puerto 29001)
    - **/var/lib/mongo/tarea/mongos/** (servidor mongos; puerto 27017)

Ficheros de configuración

- **NOSQL-007-1.dsic.cloud**:
    - Servidor de configuración **(/var/lib/mongo/tarea/cfg/mongod.conf)** :

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/lib/mongo/tarea/cfg/mongod.log

# Where and how to store data.
storage:
  dbPath: /var/lib/mongo/tarea/cfg
  journal:
      enabled: true
#  engine:
#  mmapv1:
#  wiredTiger:

# how the process runs
processManagement:
  fork: true  # fork and run in background
  pidFilePath: /var/lib/mongo/tarea/cfg/mongod.pid  # location of pidfile
  timeZoneInfo: /usr/share/zoneinfo
```

```
# network interfaces
net:
  port: 26001
  bindIp: 127.0.0.1,nosql-007-1.dsic.cloud  # Listen to local interface only,
comment to listen on all interfaces.


#security:

#operationProfiling:

replication:
  replSetName: tarea_s1_cfg

sharding:
  clusterRole: configsvr

## Enterprise-Only Options

#auditLog:

#snmp:
```

- Servidor primario del conjunto de réplicas tarea_s1 **(/var/lib/mongo/tarea/shard_s1/mongod.conf)**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s1/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s1
   journal:
    enabled: true
# engine:
# mmapv1:
# wiredTiger:

# how the process runs
```

```
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s1/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 27001
   bindIp: 127.0.0.1,nosql-007-1.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:

#operationProfiling:
replication:
   replSetName: tarea_cfg

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-007-1.dsic.cloud # Listen to local interface only,
```

```
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

*** El resto de archivos de configuración está disponible como anexo 1, al final del trabajo.**

Inicio de los servicios

Encendemos los servidores adecuados (excepto los “mongos”) en cada nodo:

- **NOSQL-007-1.dsic.cloud**:
  - **Servidor de configuración:**
    `$ mongod --config /var/lib/mongo/tarea/cfg/mongod.conf`
  - **Servidor primario de la partición tarea_s1**
    `$ mongod --config /var/lib/mongo/tarea/shard_s1/mongod.conf`

- **NOSQL-007-2.dsic.cloud**:
  - **Servidor primario de la partición tarea_s2**
    `$ mongod --config /var/lib/mongo/tarea/shard_s2/mongod.conf`

- **NOSQL-007-3.dsic.cloud:**
  - **Servidor de configuración:**
    `$ mongod --config /var/lib/mongo/tarea/cfg/mongod.conf`
  - **Servidor primario de la partición tarea_s3**
    `$ mongod --config /var/lib/mongo/tarea/shard_s3/mongod.conf`

- **NOSQL-007-4.dsic.cloud**:
  - **Servidor primario de la partición tarea_s4**
    `$ mongod --config /var/lib/mongo/tarea/shard_s4/mongod.conf`

- **NOSQL-007-5.dsic.cloud**:
  - **Servidor de configuración:**
    `$ mongod --config /var/lib/mongo/tarea/cfg/mongod.conf`
  - **Servidor primario de la partición tarea_s5**
    `$ mongod --config /var/lib/mongo/tarea/shard_s5/mongod.conf`
  - **Árbitro la partición tarea_s7**

```
$ mongod --config /var/lib/mongo/tarea/arb_s7/mongod.conf
```

- **NOSQL-007-6.dsic.cloud**:
  - **Servidor primario de la partición tarea_s6**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s6/mongod.conf
    ```
  - **Servidor secundario de la partición tarea_s7**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s7/mongod.conf
    ```

- **NOSQL-028-1.dsic.cloud**:
  - **Servidor secundario de la partición tarea_s1**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s1/mongod.conf
    ```
  - **Árbitro la partición tarea_s6**
    ```
    $ mongod --config /var/lib/mongo/tarea/arb_s6/mongod.conf
    ```

- **NOSQL-028-2.dsic.cloud**:
  - **Servidor secundario de la partición tarea_s2**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s2/mongod.conf
    ```
  - **Árbitro la partición tarea_s1**
    ```
    $ mongod --config /var/lib/mongo/tarea/arb_s1/mongod.conf
    ```

- **NOSQL-028-3.dsic.cloud**:
  - **Servidor secundario de la partición tarea_s3**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s3/mongod.conf
    ```
  - **Árbitro la partición tarea_s2**
    ```
    $ mongod --config /var/lib/mongo/tarea/arb_s2/mongod.conf
    ```

- **NOSQL-028-4.dsic.cloud**:
  - **Servidor secundario de la partición tarea_s4**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s4/mongod.conf
    ```
  - **Árbitro la partición tarea_s3**
    ```
    $ mongod --config /var/lib/mongo/tarea/arb_s3/mongod.conf
    ```

- **NOSQL-028-5.dsic.cloud**:
  - **Servidor secundario de la partición tarea_s5**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s5/mongod.conf
    ```
  - **Árbitro la partición tarea_s4**
    ```
    $ mongod --config /var/lib/mongo/tarea/arb_s4/mongod.conf
    ```

- **NOSQL-028-6.dsic.cloud**:
  - **Servidor primario de la partición tarea_s7**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s7/mongod.conf
    ```
  - **Servidor secundario de la partición tarea_s6**
    ```
    $ mongod --config /var/lib/mongo/tarea/shard_s6/mongod.conf
    ```
  - **Árbitro la partición tarea_s5**
    ```
    $ mongod --config /var/lib/mongo/tarea/arb_s5/mongod.conf
    ```

Habilitar el autenticado de acceso y la acreditación entre servidores. Dar de alta los usuarios administradores necesarios. Crear una base de datos y un usuario que pueda gestionar información dentro de ella.

Creamos un fichero de claves: en cualquier máquina hacemos los siguiente

```
$ chmod 400 /var/lib/mongo/tarea/mongodb-keyfile
```

Cambiamos los permisos

```
$ chmod 400 /var/lib/mongo/tarea/mongodb-keyfile
```

Copiamos este fichero a todas las máquinas del cluster.

```
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-007-2:/var/lib/mongo/tarea/
mongod@nosql-007-2's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-007-3:/var/lib/mongo/tarea/
mongod@nosql-007-3's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-007-4:/var/lib/mongo/tarea/
mongod@nosql-007-4's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-007-5:/var/lib/mongo/tarea/
mongod@nosql-007-5's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-007-6:/var/lib/mongo/tarea/
mongod@nosql-007-6's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-028-1:/var/lib/mongo/tarea/
mongod@nosql-028-1's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
```

```
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-028-2:/var/lib/mongo/tarea/
mongod@nosql-028-2's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-028-3:/var/lib/mongo/tarea/
mongod@nosql-028-3's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-028-4:/var/lib/mongo/tarea/
mongod@nosql-028-4's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-028-5:/var/lib/mongo/tarea/
mongod@nosql-028-5's password:
mongodb-keyfile
      100% 1024    1.0KB/s   00:00
-bash-4.2$ scp /var/lib/mongo/tarea/mongodb-keyfile
mongod@nosql-028-6:/var/lib/mongo/tarea/
mongod@nosql-028-6's password:
mongodb-keyfile
```

Nos conectamos con una mongo Shell a un mongos y paramos el balanceador

```
mongos> sh.stopBalancer()
{
      "ok" : 1,
      "$clusterTime" : {
            "clusterTime" : Timestamp(1523465274, 2),
            "signature" : {
                  "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                  "keyId" : NumberLong(0)
            }
      },
      "operationTime" : Timestamp(1523465274, 2)
}
mongos> sh.getBalancerState()
false
mongos>
```

Paramos todos los servidores de todo tipo de todas las máquinas y cambiamos todos los ficheros de configuración incluyendo este parámetro:

```
-bash-4.2$ ps -A | grep mongo
 2843 ?       00:08:23 mongod
 2964 ?       00:06:36 mongod
 4757 ?       00:01:37 mongos
-bash-4.2$ kill 2843
-bash-4.2$ kill 2964
-bash-4.2$ kill 4757
```

```
security:
  keyFile: /var/lib/mongo/tarea/mongodb-keyfile
```

Arrancamos los servidores de configuración

Arrancamos los servidores de particionamiento de todas las particiones

Arrancamos los servidores árbitros de todas las particiones

Arrancamos el servidor mongos

```
mongos --config /var/lib/mongo/tarea/mongos/mongos.conf
```

Nos conectamos con una mongo Shell a un servidor mongos (en modo local) y creamos el primer usuario

```
$ mongo
mongos>use admin
mongos>db.createUser(
{ user: "admin_cluster",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )
```

Nos autenticamos con este usuario y creamos un administrador total

```
mongos>db.auth("admin_cluster","abc123")
mongos>db.createUser(
{ user: "root_cluster",
pwd: "abc123",
roles: ["root"] } )
```

En el primero de cada conjunto de réplicas y creamos el usuario administrativo (tarea_s1)

```
$ mongo --port 27001

tarea_s1:PRIMARY> use admin
tarea_s1:PRIMARY > db.createUser(
{ user: "admin_s1",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )
```

Nos autenticamos con este usuario y creamos un administrador total:

```
tarea_s1:PRIMARY> db.auth("admin_s1","abc123")
tarea_s1:PRIMARY > db.createUser(
{ user: "root_s1",
pwd: "abc123",
roles: ["root"] } )
```

Y repetimos el proceso en todos los demás conjuntos de réplicas:

```
db.createUser(
{ user: "admin_s2",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )

db.createUser(
{ user: "root_s2",
pwd: "abc123",
roles: ["root"] } )

use admin
db.auth("root_s2","abc123")

-------------------------------------------------
use admin

db.createUser(
{ user: "admin_s3",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )


db.auth("admin_s3","abc123")

db.createUser(
{ user: "root_s3",
pwd: "abc123",
```

```
roles: ["root"] } )

use admin
db.auth("root_s3","abc123")

-------------------------------------------------
use admin

db.createUser(
{ user: "admin_s4",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )

db.auth("admin_s4","abc123")

db.createUser(
{ user: "root_s4",
pwd: "abc123",
roles: ["root"] } )


db.auth("root_s4","abc123")

-------------------------------------------------
use admin

db.createUser(
{ user: "admin_s5",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )

db.auth("admin_s5","abc123")

db.createUser(
{ user: "root_s5",
pwd: "abc123",
roles: ["root"] } )


db.auth("root_s5","abc123")

-------------------------------------------------
use admin

db.createUser(
{ user: "admin_s6",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )

db.auth("admin_s6","abc123")
```

```
db.createUser(
{ user: "root_s6",
pwd: "abc123",
roles: ["root"] } )


db.auth("root_s6","abc123")

-----------------------------------------------
use admin

db.createUser(
{ user: "admin_s7",
pwd: "abc123",
roles: [ { role: "userAdminAnyDatabase", db: "admin" } ] } )

db.auth("admin_s7","abc123")

db.createUser(
{ user: "root_s7",
pwd: "abc123",
roles: ["root"] } )


db.auth("root_s7","abc123")
```

Configurar el particionamiento

Definir el conjunto de réplicas `tarea_s1_cfg` de servidores de configuración:

1. En `nosql-007-1.dsic.cloud` nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 26001
> rs.initiate()
PRIMARY> rs.add("nosql-007-3.dsic.cloud:26001")
PRIMARY> rs.add("nosql-007-5.dsic.cloud:26001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s1_cfg:PRIMARY> rs.status()
{
```

```
        "set" : "tarea_s1_cfg",
        "date" : ISODate("2018-04-12T12:08:49.158Z"),
        "myState" : 1,
        "term" : NumberLong(2),
        "configsvr" : true,
        "heartbeatIntervalMillis" : NumberLong(2000),
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1523534919, 1),
                        "t" : NumberLong(2)
                },
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1523534919, 1),
                        "t" : NumberLong(2)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1523534919, 1),
                        "t" : NumberLong(2)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1523534919, 1),
                        "t" : NumberLong(2)
                }
        },
        "members" : [
                {
                        "_id" : 0,
                        "name" : "nosql-007-1.dsic.cloud:26001",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 807,
                        "optime" : {
                                "ts" : Timestamp(1523534919, 1),
                                "t" : NumberLong(2)
                        },
                        "optimeDate" : ISODate("2018-04-12T12:08:39Z"),
                        "electionTime" : Timestamp(1523534123, 1),
                        "electionDate" : ISODate("2018-04-12T11:55:23Z"),
                        "configVersion" : 3,
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "nosql-007-3.dsic.cloud:26001",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 21,
                        "optime" : {
```

```
                                "ts" : Timestamp(1523534919, 1),
                                "t" : NumberLong(2)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1523534919, 1),
                                "t" : NumberLong(2)
                        },
                        "optimeDate" : ISODate("2018-04-12T12:08:39Z"),
                        "optimeDurableDate" : ISODate("2018-04-12T12:08:39Z"),
                        "lastHeartbeat" : ISODate("2018-04-12T12:08:47.528Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-12T12:08:44.597Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                },
                {
                        "_id" : 2,
                        "name" : "nosql-007-5.dsic.cloud:26001",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 9,
                        "optime" : {
                                "ts" : Timestamp(1523534919, 1),
                                "t" : NumberLong(2)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1523534919, 1),
                                "t" : NumberLong(2)
                        },
                        "optimeDate" : ISODate("2018-04-12T12:08:39Z"),
                        "optimeDurableDate" : ISODate("2018-04-12T12:08:39Z"),
                        "lastHeartbeat" : ISODate("2018-04-12T12:08:47.532Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-12T12:08:44.952Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                }
        ],
        "ok" : 1,
        "operationTime" : Timestamp(1523534919, 1),
        "$gleStats" : {
                "lastOpTime" : {
                        "ts" : Timestamp(1523534919, 1),
                        "t" : NumberLong(2)
                },
                "electionId" : ObjectId("7fffffff0000000000000002")
        },
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523534919, 1),
                "signature" : {
                        "hash" : BinData(0,"LkTaMzvdd+qsj9t8nLkjKx5jH1I="),
```

```
                        "keyId" : NumberLong("6543519388559998995")
                }
        }
}
```

Definir el conjunto de réplicas tarea_s1:

1. En nosql-007-1.dsic.cloud nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 27001
> rs.initiate()
PRIMARY> rs.add("nosql-028-1.dsic.cloud:28001")
PRIMARY> rs.addArb("nosql-028-2.dsic.cloud:29001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s1:PRIMARY> rs.status ()
{
        "set" : "tarea_s1",
        "date" : ISODate("2018-04-11T11:34:34.701Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "heartbeatIntervalMillis" : NumberLong(2000),
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1523446466, 1),
                        "t" : NumberLong(1)
                },
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1523446466, 1),
                        "t" : NumberLong(1)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1523446466, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1523446466, 1),
                        "t" : NumberLong(1)
                }
        },
        "members" : [
                {
                        "_id" : 0,
                        "name" : "nosql-007-1.dsic.cloud:27001",
                        "health" : 1,
```

```
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 64970,
                        "optime" : {
                                "ts" : Timestamp(1523446466, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T11:34:26Z"),
                        "electionTime" : Timestamp(1523382670, 2),
                        "electionDate" : ISODate("2018-04-10T17:51:10Z"),
                        "configVersion" : 3,
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "nosql-028-1.dsic.cloud:28001",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 63763,
                        "optime" : {
                                "ts" : Timestamp(1523446466, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1523446466, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T11:34:26Z"),
                        "optimeDurableDate" : ISODate("2018-04-11T11:34:26Z"),
                        "lastHeartbeat" : ISODate("2018-04-11T11:34:34.571Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T11:34:34.570Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "nosql-007-1.dsic.cloud:27001",
                        "configVersion" : 3
                },
                {
                        "_id" : 2,
                        "name" : "nosql-028-2.dsic.cloud:29001",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 63731,
                        "lastHeartbeat" : ISODate("2018-04-11T11:34:33.952Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T11:34:30.805Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                }
        ],
        "ok" : 1
```

```
}
```

**** La definición del resto de conjuntos de réplicas está disponible como anexo 2, al final del trabajo.**

Configuramos las particiones y comprobamos el estado de ellas:

```
mongos> sh.addShard("tarea_s1/nosql-007-1.dsic.cloud:27001")
{
	"shardAdded" : "tarea_s1",
	"ok" : 1,
	"$clusterTime" : {
		"clusterTime" : Timestamp(1523455442, 6),
		"signature" : {
			"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
			"keyId" : NumberLong(0)
		}
	},
	"operationTime" : Timestamp(1523455442, 6)
}
mongos> sh.addShard("tarea_s2/nosql-007-2.dsic.cloud:27001")
{
	"shardAdded" : "tarea_s2",
	"ok" : 1,
	"$clusterTime" : {
		"clusterTime" : Timestamp(1523455455, 6),
		"signature" : {
			"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
			"keyId" : NumberLong(0)
		}
	},
	"operationTime" : Timestamp(1523455455, 6)
}
mongos> sh.addShard("tarea_s3/nosql-007-3.dsic.cloud:27001")
{
	"shardAdded" : "tarea_s3",
	"ok" : 1,
	"$clusterTime" : {
		"clusterTime" : Timestamp(1523455469, 5),
		"signature" : {
			"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
			"keyId" : NumberLong(0)
```

```
                }
        },
        "operationTime" : Timestamp(1523455469, 5)
}
mongos> sh.addShard("tarea_s4/nosql-007-4.dsic.cloud:27001")
{
        "shardAdded" : "tarea_s4",
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523455478, 6),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1523455478, 6)
}
mongos> sh.addShard("tarea_s5/nosql-007-5.dsic.cloud:27001")
{
        "shardAdded" : "tarea_s5",
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523455491, 6),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1523455491, 6)
}
mongos> sh.addShard("tarea_s6/nosql-007-6.dsic.cloud:27001")
{
        "shardAdded" : "tarea_s6",
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523455504, 6),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1523455504, 6)
}
mongos> sh.addShard("tarea_s7/nosql-028-6.dsic.cloud:27001")
{
        "shardAdded" : "tarea_s7",
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523455527, 5),
                "signature" : {
```

```
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1523455527, 5)
}

mongos> sh.status()
--- Sharding Status ---
  sharding version: {
        "_id" : 1,
        "minCompatibleVersion" : 5,
        "currentVersion" : 6,
        "clusterId" : ObjectId("5acf4037b923a4efca182ac1")
  }
  shards:
        {  "_id" : "tarea_s1",  "host" :
"tarea_s1/nosql-007-1.dsic.cloud:27001,nosql-028-1.dsic.cloud:28001",  "state" : 1 }
        {  "_id" : "tarea_s2",  "host" :
"tarea_s2/nosql-007-2.dsic.cloud:27001,nosql-028-2.dsic.cloud:27001",  "state" : 1 }
        {  "_id" : "tarea_s3",  "host" :
"tarea_s3/nosql-007-3.dsic.cloud:27001,nosql-028-3.dsic.cloud:28001",  "state" : 1 }
        {  "_id" : "tarea_s4",  "host" :
"tarea_s4/nosql-007-4.dsic.cloud:27001,nosql-028-4.dsic.cloud:28001",  "state" : 1 }
        {  "_id" : "tarea_s5",  "host" :
"tarea_s5/nosql-007-5.dsic.cloud:27001,nosql-028-5.dsic.cloud:28001",  "state" : 1 }
        {  "_id" : "tarea_s6",  "host" :
"tarea_s6/nosql-007-6.dsic.cloud:27001,nosql-028-6.dsic.cloud:28001",  "state" : 1 }
        {  "_id" : "tarea_s7",  "host" :
"tarea_s7/nosql-007-6.dsic.cloud:28001,nosql-028-6.dsic.cloud:27001",  "state" : 1 }
  active mongoses:
        "3.6.0" : 1
  autosplit:
        Currently enabled: yes
  balancer:
        Currently enabled:  yes
        Currently running:  no
        Failed balancer rounds in last 5 attempts:  0
        Migration Results for the last 24 hours:
                No recent migrations
  databases:
        {  "_id" : "alquilerbd",  "primary" : "tarea_s2",  "partitioned" : true }
                alquilerbd.alojamiento
                        shard key: { "_id" : 1 }
                        unique: true
                        balancing: true
                        chunks:
                                tarea_s2        1
                        { "_id" : { "$minKey" : 1 } } -->> { "_id" : { "$maxKey" :
1 } } on : tarea_s2 Timestamp(1, 0)
```

```
            alquilerbd.cliente
                        shard key: { "_id" : 1 }
                        unique: true
                        balancing: true
                        chunks:
                              tarea_s2      1
                        { "_id" : { "$minKey" : 1 } } -->> { "_id" : { "$maxKey" :
1 } } on : tarea_s2 Timestamp(1, 0)
            alquilerbd.prueb
                        shard key: { "_id" : 1 }
                        unique: true
                        balancing: true
                        chunks:
                              tarea_s2      1
                        { "_id" : { "$minKey" : 1 } } -->> { "_id" : { "$maxKey" :
1 } } on : tarea_s2 Timestamp(1, 0)
            alquilerbd.reserva
                        shard key: { "_id" : 1 }
                        unique: true
                        balancing: true
                        chunks:
                              tarea_s2      1
                        { "_id" : { "$minKey" : 1 } } -->> { "_id" : { "$maxKey" :
1 } } on : tarea_s2 Timestamp(1, 0)
      {  "_id" : "config",  "primary" : "config",  "partitioned" : true }
            config.system.sessions
                        shard key: { "_id" : 1 }
                        unique: false
                        balancing: true
                        chunks:
                              tarea_s1      1
                        { "_id" : { "$minKey" : 1 } } -->> { "_id" : { "$maxKey" :
1 } } on : tarea_s1 Timestamp(1, 0)
```

Creamos la base de datos:

```
mongos> use alquilerbd
switched to db alquilerbd
mongos> sh.enableSharding("alquilerbd_2")
{
      "ok" : 1,
      "$clusterTime" : {
            "clusterTime" : Timestamp(1523458254, 8),
            "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
            }
      },
```

```
        "operationTime" : Timestamp(1523458254, 8)
}

mongos> sh.shardCollection("alquilerbd_2.alojamiento",{_id:1},true)
{
        "collectionsharded" : "alquilerbd.alojamiento",
        "collectionUUID" : UUID("05e9d5b5-5779-4c30-a9d2-b46e0b015acf"),
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523458539, 14),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1523458539, 14)
}
mongos> sh.shardCollection("alquilerbd_2.cliente",{_id:1},true)
{
        "collectionsharded" : "alquilerbd.cliente",
        "collectionUUID" : UUID("3df11807-f16a-4b68-ad89-d00d67971542"),
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523458549, 14),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1523458549, 14)
}
mongos> sh.shardCollection("alquilerbd_2.reserva",{_id:1},true)
{
        "collectionsharded" : "alquilerbd.reserva",
        "collectionUUID" : UUID("f80e9abb-93f4-4ce0-a1aa-15f891004fff"),
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1523458557, 14),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1523458557, 14)
}
```

Cargamos los datos:

```
$ mongoimport -u root_cluster -p abc123 --authenticationDatabase admin --db
```

```
alquilerbd_2 --type=csv --collection=alojamiento --headerline
/var/lib/mongo/tarea/csv_alojamiento.csv

$ mongoimport -u root_cluster -p abc123 --authenticationDatabase admin --db
alquilerbd_2 --type=csv --collection=cliente --headerline
/var/lib/mongo/tarea/csv_cliente.csv

$ mongoimport -u root_cluster -p abc123 --authenticationDatabase admin --db
alquilerbd_2 --type=csv --collection=reserva --headerline
/var/lib/mongo/tarea/csv_reserva.csv
```

Transformamos los datos, añadimos los documentos de clientes y alojamiento a la colección reserva:

```
var cliente_cursor=db.cliente.find()
var cliente_actual
while (cliente_cursor.hasNext()) {
    cliente_actual=cliente_cursor.next();
    var reserva_cliente=db.reserva.find({id_cliente:cliente_actual.id_cliente});
    var reserva_actual;
    while (reserva_cliente.hasNext()) {
        reserva_actual=reserva_cliente.next();
        db.reserva.update({_id:reserva_actual._id},{$set :
{"alquila":cliente_actual.nom_cliente,
        "apellidos_alquila":cliente_actual.apell_cliente,
        "telefono_alquila":cliente_actual.telefono,
        "edad_alquila":cliente_actual.edad,
        "pais_alquila":cliente_actual.pais_cliente,
        "email_alquila":cliente_actual.email}});
        };
}
```

```
var alojamiento_cursor=db.alojamiento.find()
var alojamiento_actual
while (alojamiento_cursor.hasNext()) {
    alojamiento_actual=alojamiento_cursor.next();
    var
reserva_alojamiento=db.reserva.find({id_alojamiento:alojamiento_actual.id_alojamient
o});
    var reserva_actual;
    while (reserva_alojamiento.hasNext()) {
        reserva_actual=reserva_alojamiento.next();
        db.reserva.update({_id:reserva_actual._id},{$set :
{"Nombre_Edificio":alojamiento_actual.nom_edificio,
        "Canon_propietario":alojamiento_actual.canon_propietario,
        "Numero_habitaciones":alojamiento_actual.num_habitaciones,
        "Ocupacion_maxima":alojamiento_actual.ocupacion_max,
```

```
        "Poblacion_edificio":alojamiento_actual.poblacion_edificio,
        "Propietario":alojamiento_actual.propietario,
        "Tipologia":alojamiento_actual.tipologia}});
        };
};
```

Cuestión 1: Identificar los clientes que entran el día 1 de julio de 2018.

```
mongos> db.reserva.find({"fecha_entrada":"2018-7-1"}).pretty()
{
        "_id" : ObjectId("5adee50b3e6c010980c5c6d6"),
        "id_reserva" : "cdd50a98-21e0-11e8-b467-0ed5f89f718b",
        "id_cliente" : "8a029056-2172-11e8-b467-0ed5f89f718b",
        "id_alojamiento" : "85533c74-2166-11e8-b467-0ed5f89f718b",
        "nom_alojamiento" : "CT BEACH 112 1D",
        "fecha_reserva" : "2018-1-10",
        "fecha_entrada" : "2018-7-1",
        "fecha_salida" : "2018-7-6",
        "precio_estancia" : 691.2,
        "num_personas" : 2,
        "origen_reserva" : "Booking.com",
        "canal" : "OTA",
        "gasto_origen" : 0.15,
        "alquila" : "marek",
        "apellidos_alquila" : "kulawy",
        "email_alquila" : "marek.kulawy@gmail.com",
        "telefono_alquila" : 600666123,
        "Nombre_Edificio" : "COSTA TEGUISE BEACH",
        "Canon_propietario" : 0.5,
        "Numero_habitaciones" : 1,
        "Ocupacion_maxima" : 4,
        "Poblacion_edificio" : "Costa Teguise",
        "Propietario" : "PACO",
        "Tipologia" : "Costa Teguise Beach 1D- 4 pax- 112",
        "edad_alquila" : 20,
        "pais_alquila" : "Reino Unido"
}
{
        "_id" : ObjectId("5adee50b3e6c010980c5c6d7"),
        "id_reserva" : "cdd51088-21e0-11e8-b467-0ed5f89f718b",
        "id_cliente" : "8a02979a-2172-11e8-b467-0ed5f89f718b",
        "id_alojamiento" : "855335c6-2166-11e8-b467-0ed5f89f718b",
        "nom_alojamiento" : "CALAMORA-2-2º G MEDIO",
        "fecha_reserva" : "2017-12-18",
        "fecha_entrada" : "2018-7-1",
        "fecha_salida" : "2018-7-9",
        "precio_estancia" : 194.4,
```

```
        "num_personas" : 2,
        "origen_reserva" : "Booking.com",
        "canal" : "OTA",
        "gasto_origen" : 0.15,
        "alquila" : "Lee",
        "apellidos_alquila" : "Smart",
        "email_alquila" : "Lee.Smart@gmail.com",
        "telefono_alquila" : 600666125,
        "Nombre_Edificio" : "CALAMORA",
        "Canon_propietario" : 0,
        "Numero_habitaciones" : 2,
        "Ocupacion_maxima" : 6,
        "Poblacion_edificio" : "Moraira",
        "Propietario" : "PROPIEDAD",
        "Tipologia" : "Calamora 2D Doble/Literas - Apto.6 PAX",
        "edad_alquila" : 22,
        "pais_alquila" : "Reino Unido"
}
{
        "_id" : ObjectId("5adee50b3e6c010980c5c6d3"),
        "id_reserva" : "cdd5084a-21e0-11e8-b467-0ed5f89f718b",
        "id_cliente" : "8a031e36-2172-11e8-b467-0ed5f89f718b",
        "id_alojamiento" : "85535074-2166-11e8-b467-0ed5f89f718b",
        "nom_alojamiento" : "J VIVER ES 05",
        "fecha_reserva" : "2018-1-31",
        "fecha_entrada" : "2018-7-1",
        "fecha_salida" : "2018-7-8",
        "precio_estancia" : 187.92,
        "num_personas" : 2,
        "origen_reserva" : "Booking.com",
        "canal" : "OTA",
        "gasto_origen" : 0.15,
        "alquila" : "Rolando",
        "apellidos_alquila" : "Renzi",
        "email_alquila" : "Rolando.Renzi@gmail.com",
        "telefono_alquila" : 600666202,
        "Nombre_Edificio" : "JARDINES DE VIVEROS",
        "Canon_propietario" : 0.55,
        "Numero_habitaciones" : 0,
        "Ocupacion_maxima" : 3,
        "Poblacion_edificio" : "Valencia",
        "Propietario" : "ANDRES",
        "Tipologia" : "Jardines de Viveros Estudio - 3 PAX",
        "edad_alquila" : 25,
        "pais_alquila" : "Italia"
}
{
        "_id" : ObjectId("5adee50b3e6c010980c5c6d4"),
        "id_reserva" : "cdd4ffbc-21e0-11e8-b467-0ed5f89f718b",
        "id_cliente" : "8a031ca6-2172-11e8-b467-0ed5f89f718b",
```

```
        "id_alojamiento" : "85536b86-2166-11e8-b467-0ed5f89f718b",
        "nom_alojamiento" : "PUEBLA 6º-12",
        "fecha_reserva" : "2018-1-28",
        "fecha_entrada" : "2018-7-1",
        "fecha_salida" : "2018-7-7",
```

De esta manera **obtenemos la lista de clientes** que entran el 1 de julio de 2018 y así podemos mandarles un recordatorio y confirmar la reserva.

Cuestión 2: ¿Predominan clientes nacionales o extranjeros?

```
mongos> db.cliente.aggregate(
... [
... {
... $group:
... {
... _id:{pais_cliente:"$pais_cliente"},
... total_pais:{$sum:1}
... }
... }
...
... ]
...
... )
{ "_id" : { "pais_cliente" : "Irlanda" }, "total_pais" : 1 }
{ "_id" : { "pais_cliente" : "Estados Unidos" }, "total_pais" : 2 }
{ "_id" : { "pais_cliente" : "Noruega" }, "total_pais" : 1 }
{ "_id" : { "pais_cliente" : "Alemania" }, "total_pais" : 2 }
{ "_id" : { "pais_cliente" : "Reino Unido" }, "total_pais" : 15 }
{ "_id" : { "pais_cliente" : "Holanda" }, "total_pais" : 4 }
{ "_id" : { "pais_cliente" : "España" }, "total_pais" : 50 }
{ "_id" : { "pais_cliente" : "Francia" }, "total_pais" : 5 }
{ "_id" : { "pais_cliente" : "Argentina" }, "total_pais" : 1 }
{ "_id" : { "pais_cliente" : "Italia" }, "total_pais" : 3 }
{ "_id" : { "pais_cliente" : "Almenara" }, "total_pais" : 1 }
{ "_id" : { "pais_cliente" : "Rumanía" }, "total_pais" : 1 }
{ "_id" : { "pais_cliente" : "República Checa" }, "total_pais" : 1 }
{ "_id" : { "pais_cliente" : "Qatar" }, "total_pais" : 1 }
{ "_id" : { "pais_cliente" : "Eslovenia" }, "total_pais" : 1 }
```

Vemos el conteo de clientes por país y concluimos que **los clientes nacionales son predominantes**.

Cuestión 3: ¿Qué franja de edad es la que más reserva?

Menores de 20:

```
mongos> db.cliente.find({"edad":{$lte:20}}).count()
3
```

Entre 21 y 30 años:

```
mongos> db.cliente.find(
... {$and:[{"edad":{$gt:20}},
... {"edad":{$lte:30}}]
... }
... ).count()
28
```

Entre 31 y 40 años:

```
mongos> db.cliente.find(
... {$and:[{"edad":{$gt:30}},
... {"edad":{$lte:40}}]
... }
... ).count()
14
```

Entre 41 y 50 años:

```
mongos> db.cliente.find(
... {$and:[{"edad":{$gt:40}},
... {"edad":{$lte:50}}]
... }
... ).count()
35
```

Mayores de 50 años:

```
mongos> db.cliente.find({"edad":{$gt:50}}).count()
8
```

Vemos, respectivamente para cada grupo, un total de reservas de 3, 28, 14, 35 y 8. Por tanto, el grupo de edad que más reserva es el comprendido **entre los 41 y 50 años**.

Cuestión 4: ¿Mediante qué canal se factura más?

```
mongos> db.reserva.aggregate(
... [
... {
... $group:
... {
... _id:{canal:"$canal"},
... suma_estancias:{$sum:"$precio_estancia"}
... }
... }
...
... ]
...
... )
{ "_id" : { "canal" : "COMERCIAL" }, "suma_estancias" : 5157.98 }
{ "_id" : { "canal" : "WEB" }, "suma_estancias" : 2533 }
{ "_id" : { "canal" : "OTA" }, "suma_estancias" : 34044.65 }
```

Sacamos el valor de todas las estancias de la muestra por canal de facturación y llegamos a la conclusión de que **OTA** (Online Travel Agencies) **es el canal que más factura**, con 34.044, 65€.

5. Implementación Neo4j

Crear la BBDD “Alquiler.db” en la carpeta db

En la carpeta databases que se encuentra dentro de Neo4j creamos una carpeta con el nombre que le vamos a dar a nuestra base de datos donde se guardará todo lo que vayamos creando.

Guardar las tablas (archivo CSV) en la carpeta import

Al utilizar unas tablas de datos que ya tenemos guardadas en CSV, para poder cargarlas con Neo4j debemos alojarlas en la carpeta *Import* que se encuentra dentro de Neo4j.

Cambiar el archivo de configuración para meter la nueva BD

Para crear la base de datos dentro de Neo4j y poder empezar a operar en ella, debemos entrar en el archivo de *neo4j.conf* y donde nos dice el nombre de la base de datos que montar, la creamos con el comando `dbms.active_database=alquiler.db` y lo dejamos descomentado para que nos seleccione la que queremos.

```
#*****************************************************************
# Neo4j configuration
#
# For more details and a complete list of settings, please see
# https://neo4j.com/docs/operations-manual/current/reference/configuration-settings/
#*****************************************************************

# The name of the database to mount
#dbms.active_database=graph.db
#dbms.active_database=movie.db
dbms.active_database=alquiler.db
#dbms.active_database=northwind.db
#dbms.active_database=ciclismo.db
```

Start el demonio

Una vez creada la base de datos ya podemos cargar neo4j con el siguiente comando `~/neo4j-community-3.3.0/bin/neo4j start`

Importación de las tablas

Conectados desde la interfaz de Neo4j hemos importado las 3 tablas. Para poder interactuar con neo4j utilizamos la interfaz web a través de la dirección "http://nosql-007-1.dsic.cloud:7474". Una vez dentro comprobamos que se ha levantado correctamente nuestra base de datos. Y lo primero que hacemos es cargar los CSV creando los nodos para nuestra base de datos.

```
LOAD CSV WITH HEADERS FROM 'file:///alquiler/csv_cliente.csv' AS row CREATE
(c:Cliente)
SET
c.id_cliente=row.id_cliente,
c.poblacion_cliente=row.poblacion_cliente,
c.pais_cliente=row.pais_cliente,
c.nom_cliente=row.nom_cliente,
c.apell_cliente=row.apell_cliente,
c.edad=toInteger(row.edad),
c.telefono=toInteger(row.telefono),
c.email=row.email;
```

```
LOAD CSV WITH HEADERS FROM 'file:///alquiler/csv_reserva.csv' AS row CREATE
(re:Reserva)
SET
re.id_reserva=row.id_reserva,
re.id_cliente=row.id_cliente,
re.id_alojamiento=row.id_alojamiento,
re.nom_alojamiento=row.nom_alojamiento,
re.fecha_reserva=row.fecha_reserva,
re.fecha_entrada=row.fecha_entrada,
re.fecha_salida=row.fecha_salida,
re.precio_estancia=toFloat(row.precio_estancia),
re.num_personas=toInteger(row.num_personas),
re.origen_reserva=row.origen_reserva,
re.canal=row.canal;
```

```
LOAD CSV WITH HEADERS FROM 'file:///alquiler/csv_alojamiento.csv' AS row CREATE
(n:Alojamiento)
SET
n.id_alojamiento=row.id_alojamiento,
n.canon_propietario=toFloat(row.canon_propietario),
n.nom_alojamiento=row.nom_alojamiento,
n.nom_edificio=row.nom_edificio,
n.num_habitaciones=toInteger(row.num_habitaciones),
n.ocupacion_max=toInteger(row.ocupacion_max),
n.poblacion_edificio=row.poblacion_edificio,
n.propietario=row.propietario,
n.tipologia=row.tipologia;
```

Cuestión 1: Identificar los clientes que entran el día 1 de julio de 2018.

Para ello filtramos el id del cliente que ha realizado una reserva para el día 1 de Julio de 2018 en la tabla de Reserva.

```
MATCH (re:Reserva) WHERE re.fecha_entrada = "2018-7-1"
RETURN re.nom_alojamiento,re.fecha_entrada ,re.id_cliente
```

```
╒═══════════════════════╤═══════════════════╤═══════════════════════════════════════╕
│"re.nom_alojamiento"   │"re.fecha_entrada" │"re.id_cliente"                        │
╞═══════════════════════╪═══════════════════╪═══════════════════════════════════════╡
│"CT BEACH 108 1D"      │"2018-7-1"         │"8a0310e4-2172-11e8-b467-0ed5f89f718b" │
├───────────────────────┼───────────────────┼───────────────────────────────────────┤
│"J VIVER ES 05"        │"2018-7-1"         │"8a031e36-2172-11e8-b467-0ed5f89f718b" │
├───────────────────────┼───────────────────┼───────────────────────────────────────┤
│"PUEBLA 6º-12"         │"2018-7-1"         │"8a031ca6-2172-11e8-b467-0ed5f89f718b" │
├───────────────────────┼───────────────────┼───────────────────────────────────────┤
│"CT BEACH 108 1D"      │"2018-7-1"         │"8a029506-2172-11e8-b467-0ed5f89f718b" │
├───────────────────────┼───────────────────┼───────────────────────────────────────┤
│"CT BEACH 112 1D"      │"2018-7-1"         │"8a029056-2172-11e8-b467-0ed5f89f718b" │
├───────────────────────┼───────────────────┼───────────────────────────────────────┤
│"CALAMORA-2-2º G MEDIO"│"2018-7-1"         │"8a02979a-2172-11e8-b467-0ed5f89f718b" │
└───────────────────────┴───────────────────┴───────────────────────────────────────┘
```

Posteriormente con el id, podemos identificar el nombre del cliente en la tabla de Cliente.

```
MATCH (c:Cliente) WHERE c.id_cliente =
"8a031e36-2172-11e8-b467-0ed5f89f718b"  RETURN c.nom_cliente
╒═══════════════╕
│"c.nom_cliente"│
```

```
┌──────────────┐
│ "Rolando"    │
└──────────────┘
```

Cuestión 2: ¿Predominan clientes nacionales o extranjeros?

Sabiendo que tenemos un tabla con 89 filas, realizamos un conteo sobre los clientes de nacionalidad española para saber qué clase de clientela predomina más.

```
MATCH (c:Cliente{pais_cliente:"España"}) RETURN count(c) AS reserva_nacional
```

```
┌──────────────────┐
│ "reserva_nacional" │
├──────────────────┤
│ 50               │
└──────────────────┘
```

Y obtenemos más de la mitad, por tanto, podemos concluir que **los clientes nacionales son mayoritarios**.

Cuestión 3: ¿Qué franja de edad es la que más reserva?

Al igual que antes, queremos calificar nuestra clientela y queremos saber cual es la edad de nuestros clientes. Para ello realizamos la media de edad de los que alquilan nuestros alojamientos.

```
MATCH (n:Cliente) Return round(avg(n.edad)) as Edad_media
```

```
┌─────────────┐
│ "Edad_media" │
├─────────────┤
│ 37          │
└─────────────┘
```

La media de edad es de **37 años**.

Cuestión 4: ¿Mediante qué canal se factura más?

Aquí nos interesa averiguar mediante qué canal de los existentes, es el más utilizado por los clientes a la hora de realizar una reserva.

```
MATCH (r:Reserva{canal:"OTA"}) RETURN sum(r.precio_estancia) AS
Facturacion_Total

┌──────────────────┐
│ "Facturacion_Total" │
├──────────────────┤
│ 34044.65 │
└──────────────────────────────────────┘
```

Una vez más, obtenemos que el canal **OTA** es el canal que **más factura** con 34.044,65€.

*** ANEXO 1: Archivos de configuración del resto de nodos**

- **NOSQL-007-2.dsic.cloud**:
  - Servidor primario del conjunto de réplicas tarea_s2 **(/var/lib/mongo/tarea/shard_s2/mongod.conf)**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s2/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s2
   journal:
    enabled: true
# engine:
# mmapv1:
# wiredTiger:

# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s2/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 27001
   bindIp: 127.0.0.1,nosql-007-2.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
```

```
#security:

#operationProfiling:
replication:
   replSetName: tarea_s2

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-007-2.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
```

```
#snmp:
```

- **NOSQL-007-3.dsic.cloud:**
    - Servidor primario del conjunto de réplicas tarea_s3 **(/var/lib/mongo/tarea/shard_s3/mongod.conf)**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s3/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s3
   journal:
    enabled: true
# engine:
# mmapv1:
# wiredTiger:

# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s3/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 27001
   bindIp: 127.0.0.1,nosql-007-3.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:

#operationProfiling:
replication:
   replSetName: tarea_s3

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/lib/mongo/tarea/cfg/mongod.log

# Where and how to store data.
storage:
  dbPath: /var/lib/mongo/tarea/cfg
  journal:
       enabled: true
#  engine:
#  mmapv1:
#  wiredTiger:

# how the process runs
processManagement:
  fork: true  # fork and run in background
  pidFilePath: /var/lib/mongo/tarea/cfg/mongod.pid  # location of pidfile
  timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
  port: 26001
  bindIp: 127.0.0.1,nosql-007-3.dsic.cloud  # Listen to local interface only,
comment to listen on all interfaces.


#security:

#operationProfiling:

replication:
  replSetName: tarea_s1_cfg

sharding:
  clusterRole: configsvr

## Enterprise-Only Options

#auditLog:

#snmp:
```

- **NOSQL-007-4.dsic.cloud**:
    - Servidor primario del conjunto de réplicas tarea_s4 **(/var/lib/mongo/tarea/shard_s4/mongod.conf)**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s4/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s4
   journal:
    enabled: true
# engine:
# mmapv1:
# wiredTiger:

# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s4/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 27001
   bindIp: 127.0.0.1,nosql-007-4.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:

#operationProfiling:
replication:
   replSetName: tarea_s4

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-007-4.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

- **NOSQL-007-5.dsic.cloud**:
  - Servidor primario del conjunto de réplicas tarea_s5 **(/var/lib/mongo/tarea/shard_s5/mongod.conf)**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/
```

```
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s5/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s5
   journal:
    enabled: true
# engine:
# mmapv1:
# wiredTiger:

# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s5/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 27001
   bindIp: 127.0.0.1,nosql-007-5.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:

#operationProfiling:
replication:
   replSetName: tarea_s5

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Árbitro del conjunto de réplicas tarea_s7 **(/var/lib/mongo/tarea/arb_s7/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
```

```
   path: /var/lib/mongo/tarea/arb_s7/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/arb_s7
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/arb_s7/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 29001
   bindIp: 127.0.0.1,nosql-007-05.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s7
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor de configuración **(/var/lib/mongo/tarea/cfg/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/lib/mongo/tarea/cfg/mongod.log

# Where and how to store data.
storage:
  dbPath: /var/lib/mongo/tarea/cfg
  journal:
       enabled: true
#  engine:
#  mmapv1:
```

```
#  wiredTiger:

# how the process runs
processManagement:
  fork: true  # fork and run in background
  pidFilePath: /var/lib/mongo/tarea/cfg/mongod.pid  # location of pidfile
  timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
  port: 26001
  bindIp: 127.0.0.1,nosql-007-5.dsic.cloud  # Listen to local interface only,
comment to listen on all interfaces.


#security:

#operationProfiling:

replication:
  replSetName: tarea_s1_cfg

sharding:
  clusterRole: configsvr

## Enterprise-Only Options

#auditLog:

#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
```

```
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-007-5.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

- **NOSQL-007-6.dsic.cloud**:
    - Servidor primario del conjunto de réplicas tarea_s6 **(/var/lib/mongo/tarea/shard_s6/mongod.conf)**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s6/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s6
   journal:
    enabled: true
# engine:
# mmapv1:
# wiredTiger:
```

```
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s6/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 27001
   bindIp: 127.0.0.1,nosql-007-6.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:

#operationProfiling:
replication:
   replSetName: tarea_s6

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor secundario mongod del conjunto de réplicas tarea_s7 **(/var/lib/mongo/tarea/shard_s7/mongod.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s7/mongod.log
# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s7
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s7/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
```

```
   port: 28001
   bindIp: 127.0.0.1,nosql-007-6.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s7
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-007-6.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
```

```
#auditLog:
#snmp:
```

- **NOSQL-028-1.dsic.cloud**:
    - Servidor secundario mongod del conjunto de réplicas tarea_s1 **(/var/lib/mongo/tarea/shard_s1/mongod.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s1/mongod.log
# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s1
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s1/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 28001
   bindIp: 127.0.0.1,nosql-028-1.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s1
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

    - Árbitro del conjunto de réplicas tarea_s6 **(/var/lib/mongo/tarea/arb_s6/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/arb_s6/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/arb_s6
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/arb_s6/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 29001
   bindIp: 127.0.0.1,nosql-028-1.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s6
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log
```

```
# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-028-1.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

- **NOSQL-028-2.dsic.cloud**:
    - Servidor secundario mongod del conjunto de réplicas tarea_s2**(/var/lib/mongo/tarea/shard_s2/mongod.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s2/mongod.log
# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s2
   journal:
     enabled: true
# engine:
# mmapv1:
```

```
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s2/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 28001
   bindIp: 127.0.0.1,nosql-028-2.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s2
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Árbitro del conjunto de réplicas tarea_s1 **(/var/lib/mongo/tarea/arb_s1/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/arb_s1/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/arb_s1
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/arb_s1/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 29001
   bindIp: 127.0.0.1,nosql-028-2.dsic.cloud # Listen to local interface only,
```

```
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s1
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-028-2.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

- **NOSQL-028-3.dsic.cloud**:
    - Servidor secundario mongod del conjunto de réplicas tarea_s3 **(/var/lib/mongo/tarea/shard_s3/mongod.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s3/mongod.log
# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s3
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s3/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 28001
   bindIp: 127.0.0.1,nosql-028-3.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s3
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

    - Árbitro del conjunto de réplicas tarea_s2 **(/var/lib/mongo/tarea/arb_s2/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
```

```
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/arb_s2/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/arb_s2
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/arb_s2/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 29001
   bindIp: 127.0.0.1,nosql-028-3.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s2

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
```

```
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-028-3.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

- **NOSQL-028-4.dsic.cloud**:
    - Servidor secundario mongod del conjunto de réplicas tarea_s4**(/var/lib/mongo/tarea/shard_s4/mongod.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s4/mongod.log
# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s4
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
```

```
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s4/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 28001
   bindIp: 127.0.0.1,nosql-028-4.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s4
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Árbitro del conjunto de réplicas tarea_s3 **(/var/lib/mongo/tarea/arb_s3/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/arb_s3/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/arb_s3
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/arb_s3/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 29001
   bindIp: 127.0.0.1,nosql-028-4.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
```

```
#operationProfiling:
replication:
   replSetName: tarea_s3

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-028-4.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

- **NOSQL-028-5.dsic.cloud**:
    - Servidor secundario mongod del conjunto de réplicas tarea_s5**(/var/lib/mongo/tarea/shard_s5/mongod.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s5/mongod.log
# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s5
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s5/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 28001
   bindIp: 127.0.0.1,nosql-028-5.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s5
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

    - Árbitro del conjunto de réplicas tarea_s4 **(/var/lib/mongo/tarea/arb_s4/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
```

```
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/arb_s4/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/arb_s4
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/arb_s4/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 29001
   bindIp: 127.0.0.1,nosql-028-5.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s4

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
```

```
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-028-5.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

- **NOSQL-028-6.dsic.cloud**:
    - Servidor primario del conjunto de réplicas tarea_s7 **(/var/lib/mongo/tarea/shard_s7/mongod.conf)**

```
 mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s7/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s7
   journal:
    enabled: true
# engine:
# mmapv1:
# wiredTiger:
```

```
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s7/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 27001
   bindIp: 127.0.0.1,nosql-028-6.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:

#operationProfiling:
replication:
   replSetName: tarea_s7

sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor secundario mongod del conjunto de réplicas tarea_s6 **(/var/lib/mongo/tarea/shard_s6/mongod.conf):**

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/shard_s6/mongod.log
# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/shard_s6
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/shard_s5/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
```

```
   port: 28001
   bindIp: 127.0.0.1,nosql-028-6.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s6
sharding:
   clusterRole: shardsvr
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Árbitro del conjunto de réplicas tarea_s5 **(/var/lib/mongo/tarea/arb_s5/mongod.conf)**:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/arb_s5/mongod.log

# Where and how to store data.
storage:
   dbPath: /var/lib/mongo/tarea/arb_s5
   journal:
     enabled: true
# engine:
# mmapv1:
# wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/arb_s5/mongod.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
   port: 29001
   bindIp: 127.0.0.1,nosql-028-6.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.
#security:
#operationProfiling:
replication:
   replSetName: tarea_s5

sharding:
   clusterRole: shardsvr
```

```
## Enterprise-Only Options
#auditLog:
#snmp:
```

- Servidor mongos (**/var/lib/mongo/tarea/mongos/mongos.conf):**

```
# mongod.conf

# for documentation of all options, see:
#    http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
   destination: file
   logAppend: true
   path: /var/lib/mongo/tarea/mongos/mongos.log

# Where and how to store data.
#storage:
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
   fork: true # fork and run in background
   pidFilePath: /var/lib/mongo/tarea/mongos/mongos.pid # location of pidfile
   timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
   port: 27017
   bindIp: 127.0.0.1,nosql-028-6.dsic.cloud # Listen to local interface only,
comment to listen on all interfaces.

#security:
#operationProfiling:
#replication:
sharding:
   configDB: tarea_s1_cfg/nosql-007-1.dsic.cloud:26001,nosql-007-3.dsic.cloud:26001,
nosql-007-5.dsic.cloud:26001
## Enterprise-Only Options
#auditLog:
#snmp:
```

**** ANEXO 2: Definición del resto de conjuntos de réplicas:**

Definir el conjunto de réplicas tarea_s2:

1. En nosql-007-2.dsic.cloud nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 27001
> rs.initiate()
PRIMARY> rs.add("nosql-028-2.dsic.cloud:27001")
PRIMARY> rs.addArb("nosql-028-3.dsic.cloud:29001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s2:PRIMARY> rs.status()
{
	"set" : "tarea_s2",
	"date" : ISODate("2018-04-11T11:39:39.575Z"),
	"myState" : 1,
	"term" : NumberLong(1),
	"heartbeatIntervalMillis" : NumberLong(2000),
	"optimes" : {
		"lastCommittedOpTime" : {
			"ts" : Timestamp(1523446771, 1),
			"t" : NumberLong(1)
		},
		"readConcernMajorityOpTime" : {
			"ts" : Timestamp(1523446771, 1),
			"t" : NumberLong(1)
		},
		"appliedOpTime" : {
			"ts" : Timestamp(1523446771, 1),
			"t" : NumberLong(1)
		},
		"durableOpTime" : {
			"ts" : Timestamp(1523446771, 1),
			"t" : NumberLong(1)
		}
```

```
	},
	"members" : [
		{
			"_id" : 0,
			"name" : "nosql-007-2.dsic.cloud:27001",
			"health" : 1,
			"state" : 1,
			"stateStr" : "PRIMARY",
			"uptime" : 64820,
			"optime" : {
				"ts" : Timestamp(1523446771, 1),
				"t" : NumberLong(1)
			},
			"optimeDate" : ISODate("2018-04-11T11:39:31Z"),
			"electionTime" : Timestamp(1523382787, 2),
			"electionDate" : ISODate("2018-04-10T17:53:07Z"),
			"configVersion" : 3,
			"self" : true
		},
		{
			"_id" : 1,
			"name" : "nosql-028-2.dsic.cloud:27001",
			"health" : 1,
			"state" : 2,
			"stateStr" : "SECONDARY",
			"uptime" : 63889,
			"optime" : {
				"ts" : Timestamp(1523446771, 1),
				"t" : NumberLong(1)
			},
			"optimeDurable" : {
				"ts" : Timestamp(1523446771, 1),
				"t" : NumberLong(1)
			},
			"optimeDate" : ISODate("2018-04-11T11:39:31Z"),
			"optimeDurableDate" : ISODate("2018-04-11T11:39:31Z"),
			"lastHeartbeat" : ISODate("2018-04-11T11:39:39.248Z"),
			"lastHeartbeatRecv" : ISODate("2018-04-11T11:39:38.050Z"),
			"pingMs" : NumberLong(0),
			"syncingTo" : "nosql-007-2.dsic.cloud:27001",
			"configVersion" : 3
		},
		{
			"_id" : 2,
			"name" : "nosql-028-3.dsic.cloud:29001",
			"health" : 1,
			"state" : 7,
			"stateStr" : "ARBITER",
			"uptime" : 63856,
			"lastHeartbeat" : ISODate("2018-04-11T11:39:37.914Z"),
```

```
                        "lastHeartbeatRecv" : ISODate("2018-04-11T11:39:34.606Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                }
        ],
        "ok" : 1
}
```

Definir el conjunto de réplicas tarea_s3:

1. En nosql-007-3.dsic.cloud nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 27001
> rs.initiate()
PRIMARY> rs.add("nosql-028-3.dsic.cloud:28001")
PRIMARY> rs.addArb("nosql-028-4.dsic.cloud:29001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s3:PRIMARY> rs.status()
{
        "set" : "tarea_s3",
        "date" : ISODate("2018-04-11T11:59:56.704Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "heartbeatIntervalMillis" : NumberLong(2000),
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1523447990, 1),
                        "t" : NumberLong(1)
                },
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1523447990, 1),
                        "t" : NumberLong(1)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1523447990, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1523447990, 1),
                        "t" : NumberLong(1)
                }
        },
        "members" : [
                {
                        "_id" : 0,
```

```
                        "name" : "nosql-007-3.dsic.cloud:27001",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 65964,
                        "optime" : {
                                "ts" : Timestamp(1523447990, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T11:59:50Z"),
                        "electionTime" : Timestamp(1523383136, 2),
                        "electionDate" : ISODate("2018-04-10T17:58:56Z"),
                        "configVersion" : 3,
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "nosql-028-3.dsic.cloud:28001",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 64835,
                        "optime" : {
                                "ts" : Timestamp(1523447990, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1523447990, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T11:59:50Z"),
                        "optimeDurableDate" : ISODate("2018-04-11T11:59:50Z"),
                        "lastHeartbeat" : ISODate("2018-04-11T11:59:56.550Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T11:59:55.121Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "nosql-007-3.dsic.cloud:27001",
                        "configVersion" : 3
                },
                {
                        "_id" : 2,
                        "name" : "nosql-028-4.dsic.cloud:29001",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 64819,
                        "lastHeartbeat" : ISODate("2018-04-11T11:59:56.550Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T11:59:54.502Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                }
```

```
        ],
        "ok" : 1
}
```

Definir el conjunto de réplicas tarea_s4:

1. En nosql-007-4.dsic.cloud nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 27001
> rs.initiate()
PRIMARY> rs.add("nosql-028-4.dsic.cloud:28001")
PRIMARY> rs.addArb("nosql-028-5.dsic.cloud:29001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s4:PRIMARY> rs.status()
{
        "set" : "tarea_s4",
        "date" : ISODate("2018-04-11T13:47:22.940Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "heartbeatIntervalMillis" : NumberLong(2000),
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1523454440, 1),
                        "t" : NumberLong(1)
                },
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1523454440, 1),
                        "t" : NumberLong(1)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1523454440, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1523454440, 1),
                        "t" : NumberLong(1)
                }
        },
        "members" : [
                {
                        "_id" : 0,
                        "name" : "nosql-007-4.dsic.cloud:27001",
                        "health" : 1,
                        "state" : 1,
```

```
                        "stateStr" : "PRIMARY",
                        "uptime" : 72315,
                        "optime" : {
                                "ts" : Timestamp(1523454440, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T13:47:20Z"),
                        "electionTime" : Timestamp(1523383196, 2),
                        "electionDate" : ISODate("2018-04-10T17:59:56Z"),
                        "configVersion" : 3,
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "nosql-028-4.dsic.cloud:28001",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 71224,
                        "optime" : {
                                "ts" : Timestamp(1523454440, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1523454440, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T13:47:20Z"),
                        "optimeDurableDate" : ISODate("2018-04-11T13:47:20Z"),
                        "lastHeartbeat" : ISODate("2018-04-11T13:47:22.294Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:47:21.550Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "nosql-007-4.dsic.cloud:27001",
                        "configVersion" : 3
                },
                {
                        "_id" : 2,
                        "name" : "nosql-028-5.dsic.cloud:29001",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 71206,
                        "lastHeartbeat" : ISODate("2018-04-11T13:47:21.865Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:47:20.912Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                }
        ],
        "ok" : 1
}
```

Definir el conjunto de réplicas tarea_s5:

1. En nosql-007-5.dsic.cloud nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 27001
> rs.initiate()
PRIMARY> rs.add("nosql-028-5.dsic.cloud:28001")
PRIMARY> rs.addArb("nosql-028-6.dsic.cloud:29001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s5:PRIMARY> rs.status ()
{
	"set" : "tarea_s5",
	"date" : ISODate("2018-04-11T13:52:13.135Z"),
	"myState" : 1,
	"term" : NumberLong(1),
	"heartbeatIntervalMillis" : NumberLong(2000),
	"optimes" : {
		"lastCommittedOpTime" : {
			"ts" : Timestamp(1523454730, 1),
			"t" : NumberLong(1)
		},
		"readConcernMajorityOpTime" : {
			"ts" : Timestamp(1523454730, 1),
			"t" : NumberLong(1)
		},
		"appliedOpTime" : {
			"ts" : Timestamp(1523454730, 1),
			"t" : NumberLong(1)
		},
		"durableOpTime" : {
			"ts" : Timestamp(1523454730, 1),
			"t" : NumberLong(1)
		}
	},
	"members" : [
		{
			"_id" : 0,
			"name" : "nosql-007-5.dsic.cloud:27001",
			"health" : 1,
			"state" : 1,
			"stateStr" : "PRIMARY",
			"uptime" : 72558,
			"optime" : {
				"ts" : Timestamp(1523454730, 1),
				"t" : NumberLong(1)
			},
```

```
                        "optimeDate" : ISODate("2018-04-11T13:52:10Z"),
                        "electionTime" : Timestamp(1523383247, 2),
                        "electionDate" : ISODate("2018-04-10T18:00:47Z"),
                        "configVersion" : 3,
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "nosql-028-5.dsic.cloud:28001",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 71468,
                        "optime" : {
                                "ts" : Timestamp(1523454730, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1523454730, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T13:52:10Z"),
                        "optimeDurableDate" : ISODate("2018-04-11T13:52:10Z"),
                        "lastHeartbeat" : ISODate("2018-04-11T13:52:11.525Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:52:13.058Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "nosql-007-5.dsic.cloud:27001",
                        "configVersion" : 3
                },
                {
                        "_id" : 2,
                        "name" : "nosql-028-6.dsic.cloud:29001",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 71456,
                        "lastHeartbeat" : ISODate("2018-04-11T13:52:11.502Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:52:11.180Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                }
        ],
        "ok" : 1
}
```


Definir el conjunto de réplicas tarea_s6:

1. En nosql-007-6.dsic.cloud nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 27001
> rs.initiate()
PRIMARY> rs.add("nosql-028-6.dsic.cloud:28001")
PRIMARY> rs.addArb("nosql-028-1.dsic.cloud:29001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s6:PRIMARY> rs.status()
{
	"set" : "tarea_s6",
	"date" : ISODate("2018-04-11T13:54:37.010Z"),
	"myState" : 1,
	"term" : NumberLong(1),
	"heartbeatIntervalMillis" : NumberLong(2000),
	"optimes" : {
		"lastCommittedOpTime" : {
			"ts" : Timestamp(1523454872, 1),
			"t" : NumberLong(1)
		},
		"readConcernMajorityOpTime" : {
			"ts" : Timestamp(1523454872, 1),
			"t" : NumberLong(1)
		},
		"appliedOpTime" : {
			"ts" : Timestamp(1523454872, 1),
			"t" : NumberLong(1)
		},
		"durableOpTime" : {
			"ts" : Timestamp(1523454872, 1),
			"t" : NumberLong(1)
		}
	},
	"members" : [
		{
			"_id" : 0,
			"name" : "nosql-007-6.dsic.cloud:27001",
			"health" : 1,
			"state" : 1,
			"stateStr" : "PRIMARY",
			"uptime" : 72634,
			"optime" : {
				"ts" : Timestamp(1523454872, 1),
				"t" : NumberLong(1)
			},
			"optimeDate" : ISODate("2018-04-11T13:54:32Z"),
			"electionTime" : Timestamp(1523383288, 2),
			"electionDate" : ISODate("2018-04-10T18:01:28Z"),
			"configVersion" : 3,
			"self" : true
```

```
            },
            {
                        "_id" : 1,
                        "name" : "nosql-028-6.dsic.cloud:28001",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 71570,
                        "optime" : {
                              "ts" : Timestamp(1523454872, 1),
                              "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                              "ts" : Timestamp(1523454872, 1),
                              "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T13:54:32Z"),
                        "optimeDurableDate" : ISODate("2018-04-11T13:54:32Z"),
                        "lastHeartbeat" : ISODate("2018-04-11T13:54:35.890Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:54:35.889Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "nosql-007-6.dsic.cloud:27001",
                        "configVersion" : 3
            },
            {
                        "_id" : 2,
                        "name" : "nosql-028-1.dsic.cloud:29001",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 71555,
                        "lastHeartbeat" : ISODate("2018-04-11T13:54:35.890Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:54:34.486Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
            }
      ],
      "ok" : 1
}
```

Definir el conjunto de réplicas tarea_s7:

1. En nosql-028-6.dsic.cloud nos conectamos al servidor mongod que va a ser el primario del conjunto de réplicas:

```
$ mongo --port 27001
> rs.initiate()
PRIMARY> rs.add("nosql-007-6.dsic.cloud:28001")
PRIMARY> rs.addArb("nosql-007-5.dsic.cloud:29001")
```

2. Comprobamos que todo funciona correctamente:

```
tarea_s7:PRIMARY> rs.status()
{
	"set" : "tarea_s7",
	"date" : ISODate("2018-04-11T13:57:30.576Z"),
	"myState" : 1,
	"term" : NumberLong(1),
	"heartbeatIntervalMillis" : NumberLong(2000),
	"optimes" : {
		"lastCommittedOpTime" : {
			"ts" : Timestamp(1523455046, 1),
			"t" : NumberLong(1)
		},
		"readConcernMajorityOpTime" : {
			"ts" : Timestamp(1523455046, 1),
			"t" : NumberLong(1)
		},
		"appliedOpTime" : {
			"ts" : Timestamp(1523455046, 1),
			"t" : NumberLong(1)
		},
		"durableOpTime" : {
			"ts" : Timestamp(1523455046, 1),
			"t" : NumberLong(1)
		}
	},
	"members" : [
		{
			"_id" : 0,
			"name" : "nosql-028-6.dsic.cloud:27001",
			"health" : 1,
			"state" : 1,
			"stateStr" : "PRIMARY",
			"uptime" : 72740,
			"optime" : {
				"ts" : Timestamp(1523455046, 1),
				"t" : NumberLong(1)
			},
			"optimeDate" : ISODate("2018-04-11T13:57:26Z"),
			"electionTime" : Timestamp(1523383012, 2),
			"electionDate" : ISODate("2018-04-10T17:56:52Z"),
			"configVersion" : 3,
			"self" : true
		},
		{
			"_id" : 1,
			"name" : "nosql-007-6.dsic.cloud:28001",
			"health" : 1,
			"state" : 2,
```

```
                        "stateStr" : "SECONDARY",
                        "uptime" : 72001,
                        "optime" : {
                                "ts" : Timestamp(1523455046, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1523455046, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2018-04-11T13:57:26Z"),
                        "optimeDurableDate" : ISODate("2018-04-11T13:57:26Z"),
                        "lastHeartbeat" : ISODate("2018-04-11T13:57:29.631Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:57:29.631Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "nosql-028-6.dsic.cloud:27001",
                        "configVersion" : 3
                },
                {
                        "_id" : 2,
                        "name" : "nosql-007-5.dsic.cloud:29001",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 71983,
                        "lastHeartbeat" : ISODate("2018-04-11T13:57:29.749Z"),
                        "lastHeartbeatRecv" : ISODate("2018-04-11T13:57:27.463Z"),
                        "pingMs" : NumberLong(0),
                        "configVersion" : 3
                }
        ],
        "ok" : 1
}
```

***** ANEXO 3: Acceso MongoDB**

```
use admin

db.auth("admin_s2","abc123")
db.auth("root_s2","abc123")

db.auth("admin_cluster","abc123")
db.auth("root_cluster","abc123")

ps -A | grep mongo

mongos --config /var/lib/mongo/tarea/mongos/mongos.conf

mongo --port 27001
```