Modelos de agregación: Bases de datos documentales

Máster en Business Analytics y Big Data

Bases de Datos No Convencionales

Contenidos de la sesión

- ¿Qué es MongoDB?
- Modelo de datos
- Operaciones CRUD: *hands-on*
- Recomendaciones de uso

¿Qué es MongoDB?

¿Qué es MongoDB?

- MongoDB (de *humongous*, descomunal) es un sistema gestor de base de datos creado por 10gen en 2007:
  - Sin esquema
  - Orientado a documentos
    - El documento es la unidad atómica (16 MB)
  - Multiplataforma (Linux, Windows y OS X)
  - De libre distribución: Tiene licencia GNU AGPL 3,0.
  - Está escrito en C++.
  - Se puede consultar a través de consola o de API.
    - Tiene APIs y drivers para múltiples lenguajes de programación (C#, Java, Node.js, PHP, Python, Ruby, C, C++, Perl o Scala)

¿Qué es MongoDB?

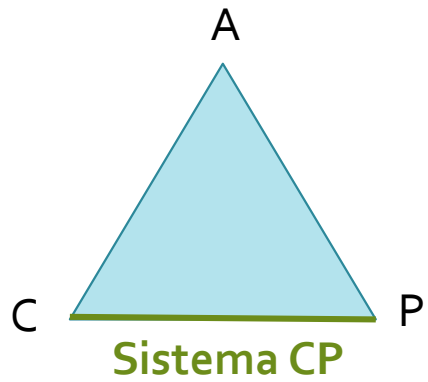
- MongoDB (de *humongous*, descomunal) es un sistema gestor de base de datos creado por 10gen en 2007:
  - Almacena los datos en BSON (formato binario de JSON)
  - Se pueden crear índices sobre el documento o cualquiera de sus partes.
  - Escrituras atómicas (a nivel de documento)
  - No permite transacciones.

¿Qué es MongoDB?

- MongoDB (de *humongous*, descomunal) es un sistema gestor de base de datos creado por 10gen en 2007:
  - Hace replicación *master-slave.*
    - Permite consistencia estricta o final en el tiempo (*eventual consistency*).
  - Distribuye automáticamente los datos en los nodos (*auto-sharding*).
  - Soporta MapReduce.

¿Qué es MongoDB?

- MongoDB garantiza que el sistema contiene una visión consistente de los datos, aunque no esté totalmente disponible en presencia de particiones.
  - Puede configurarse para que se comporte de distinta forma, promocionando la disponibilidad a costa de la consistencia.

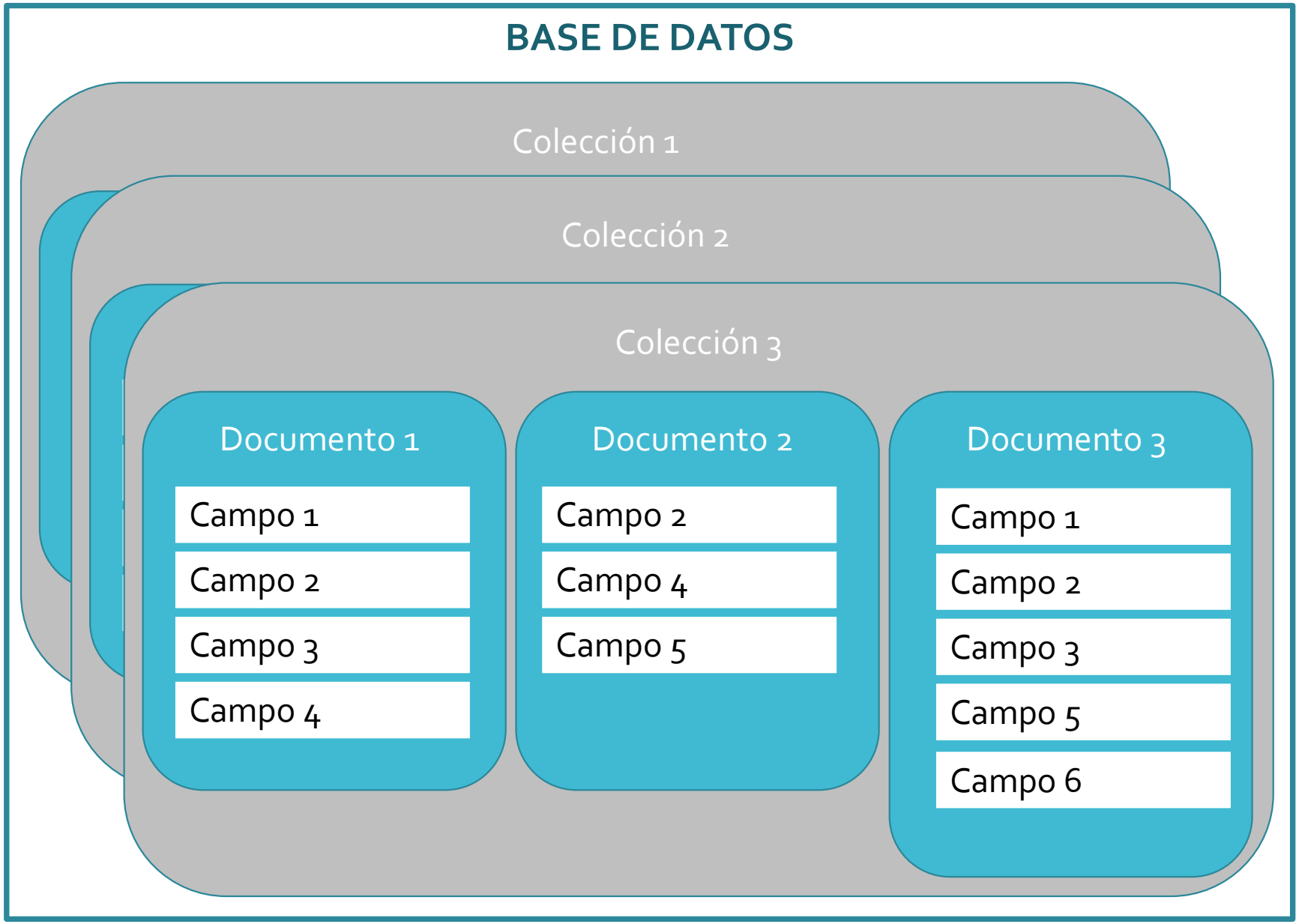


CIFF

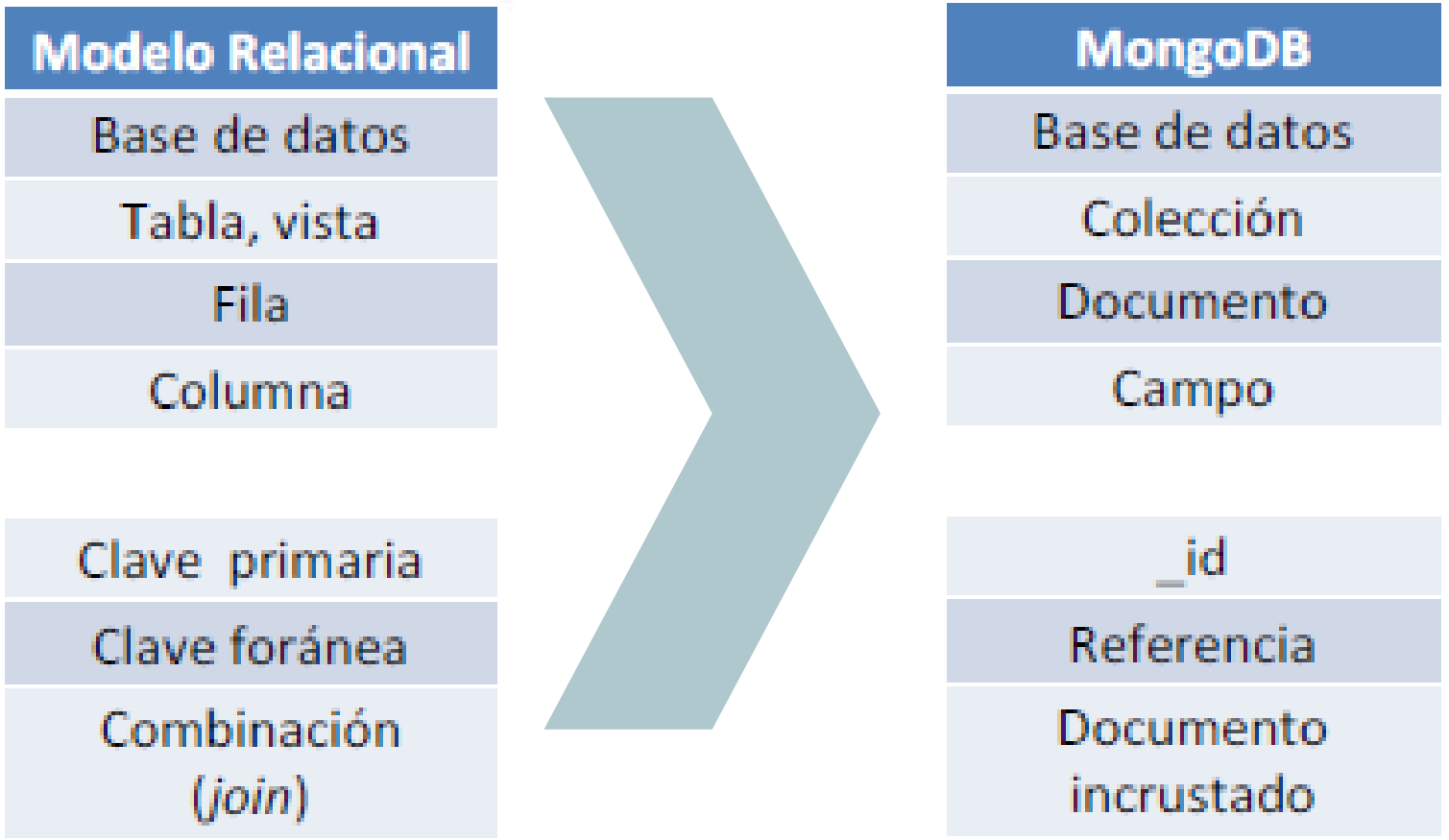
Universidad de Alcalá

Modelo de Datos

Modelo de datos



Modelo de datos



Modelo de datos

```
{ "_id" : "4da2c0e2e999fb56bf000002",
  "title" : "Un ejemplo de entrada de blog",
  "body" : "Lorem ipsum dolor sit amet…",
  "published_at" : "2015-01-02T20:15:07-07:00",
  "author_info" :
        {"_id" : "4dc8919331c0c00001000002",
        "name" : "Pepe Perez"},
  "tags" : ["Arquitectura", "Modernismo", "Gaudi"],
  "comments" : [
                {"author_info" :
                   {"name" : "Luis Lopez",
                    "email" : "ll@example.com" },
                  "body" : "Comentario 1",
                  "created_at" : "2015-01-032T10:14:01-07:00"},
                {"author_info" :
                   {"name" : "Jose Gomez",
                    "email" : "jg@example.com" },
                 "body" : "Comentario2",
                 "created_at" : "2015-01-02T10:14:09-07:00" }
              ]
  "liked_by" : ["4d7cf768e999fb67c0000001", "4da34c62ba875a19d4000001"] }
```

CLAVE

CAMPO

DOCUMENTO INCRUSTADO

DOCUMENTO INCRUSTADO

DOCUMENTO INCRUSTADO

REFERENCIAS

CIFF
Universidad de Alcalá

Operaciones básicas en modo consola

Operaciones básicas

- Las operaciones de creación, recuperación, actualización y borrado (CRUD):
  - Se realizan en el ámbito de una colección.
  - Se realizan a nivel de documento.

(ver fichero MongoDB.doc para la sesión hands-on)

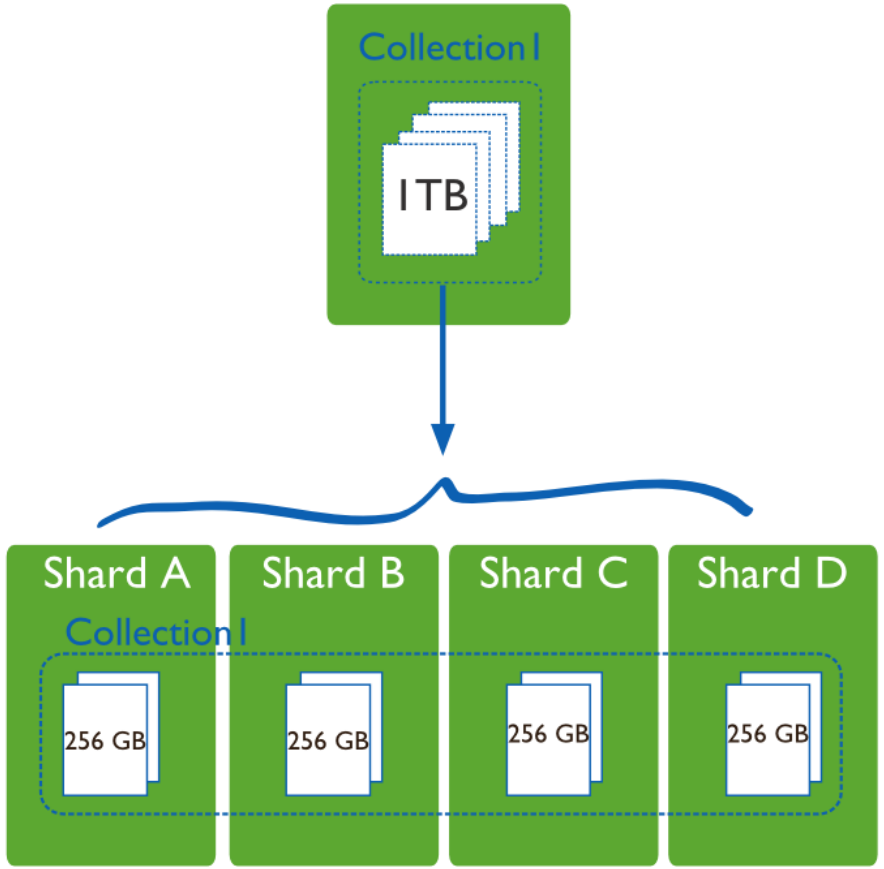
CIFF

Universidad de Alcalá

Distribución de datos

Fragmentación de datos

- MongoDB está diseñado para hacer escalado horizontal o sharding:



Fragmentación de datos

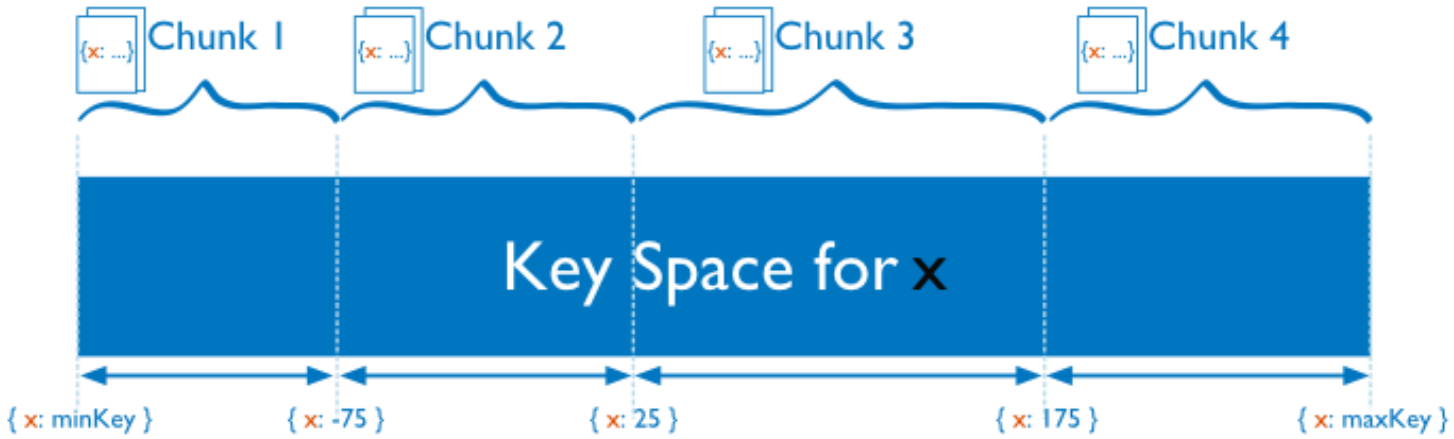
- MongoDB distribuye los datos en *shards* a nivel de colección.
  - Una colección está distribuída en varios *shards* (automáticamente).
  - Necesita una *shard key* por colección
    - Tiene que ser un campo indexado o un índice compuesto.
  - Se divide el ámbito de la *shard key* en *chuncks.*
  - Cada *chunk* se distribuye en un *shard.*

Fragmentación de datos

- MongoDB distribuye los datos en *shards* a nivel de colección.
  - Cuando un *chunk* crece por encima de un parámetro, se divide automáticamente.
  - Cuando los *shards* están desbalanceados, se producen migraciones de *chunks* automáticamente.
  - Los valores de las *shard keys* se distribuyen en los *chunks*:
    - Particionamiento basado en rango.
    - Particionamiento basado en funciones de distribución (*hash*).

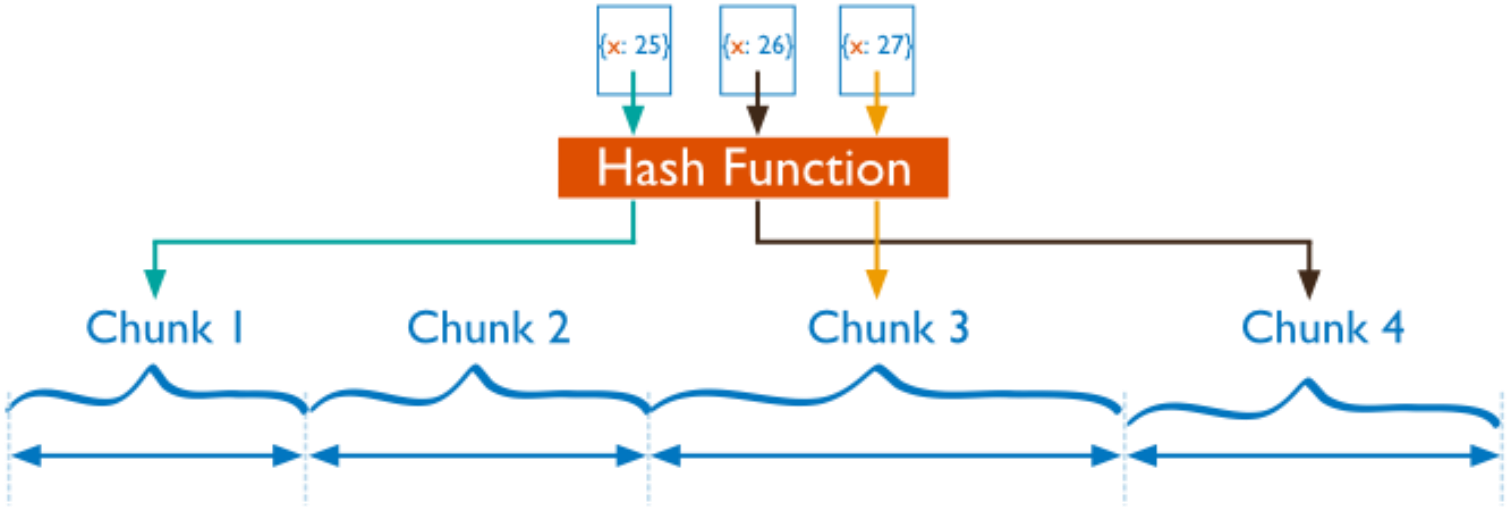
Fragmentación de datos

- Particionamiento basado en rango:
  - Se dividen los posibles valores de la *shard key* en intervalos.
  - Cada intervalo se lleva a un *chunk*.
  - Los documentos se alojan en los *chunks* de acuerdo a su valor de la *shard key*.



Fragmentación de datos

- Particionamiento basado en funciones de distribución (*hash*):
  - Utiliza una función de distribución sobre el valor de la *shard key*.
  - Aloja un documento en el *chunk* que corresponde al valor devuelto por la función *hash* sobre su valor de la *shard key*.

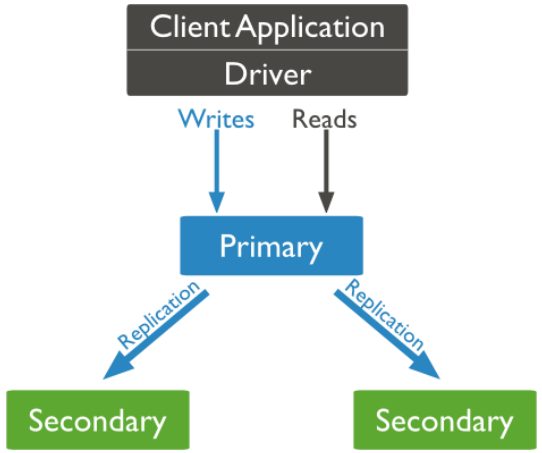


Fragmentación de datos

- Consideraciones de rendimiento:
  - *Sharding* basado en rangos:
    - Más eficiente cuando se consultan documentos por rango.
    - Escala peor si la *shard key* es incremental.
  - *Sharding* basado en función *hash*:
    - Distribuyen los documentos uniformemente en los *shards*.
    - Menos eficiente para consultar documentos secuenciales o por rango.

Replicación de datos

- La replica consiste en duplicar datos en dos o más nodos para garantizar la redundancia:
  - Facilita tener copias de seguridad y la recuperación de datos en caso de catástrofe.
  - Aumentar la locabilidad y disponibilidad de los datos.
- Mongo replicar utilizando *replica sets*:
  - Un replica set incluye un nodo primario y varios secundario que contienen el mismo conjunto de datos.



Replicación de datos

- Gestión de réplicas *master-slave:*
  - Existe un solo maestro (o replica primaria) por cada *replica set*.
  - Las operaciones de lectura y escritura se realizan sólo sobre la copia primaria.
  - Los datos de las copias secundarias se sincronizan a partir de las operaciones realizadas en la copia primaria de forma asíncrona.
  - En caso de que el nodo que almacena la copia primaria caiga, se promociona una copia secundaria (o *slave*) como nueva copia primaria.
  - Opcionalmente, pueden añadirse nodos de tipo "Arbitro" a un *replica set*.
    - Desempatar las votaciones que puedan surgir al escoger nuevas copias primarias.
  - Se pueden definir prioridades sobre las distintas copias secundarias para incrementar (o decrementar) su probabilidad de ser escogidas como nodo primario.
  - Permite configuraciones para soportar consultas directas sobre las copias secundarias.
    - En este caso la consistencia es final en el tiempo.

CIFF

Universidad de Alcalá

Recomendaciones de uso

¿Cuándo usar MongoDB?

- ¿Cuándo usar MongoDB?
  - Cuando se necesite almacenar datos semi-estructurados.
    - El esquema reside en la aplicación.
    - El esquema lo definen las consultas que se vayan a realizar con más frecuencia.
  - Útil en entornos que requieran escalabilidad.
    - Sencillez en la configuración de la replicación y el sharding.

¿Cuándo no usar MongoDB?

- ¿Cuándo NO usar MongoDB?
  - Cuando se necesiten transacciones.
  - Cuando se necesitan frecuentemente datos relacionados en dos o más colecciones.
    - No existe la posibilidad de hacer *joins.*
  - Cuando se necesitan explotar informes complejos, que requieran agregaciones complejas.

Lecturas

- Página oficial de MongoDB: http://www.mongodb.org
- Documentación oficial: http://docs.mongodb.org/manual/
- Listado de *drivers*: http://docs.mongodb.org/ecosystem/drivers/
- Grupo de usuarios de MongoDB-Spain: http://www.meetup.com/mongodb-spain/
- K. Chodorow (2011). 50 Tips for MongoDB Developers O'Really Media.
- MongoDB Corp (2014) RDBMS to MongoDB Migration Guide. http://www.mongodb.com/lp/white-paper/migration-rdbms-nosql-mongodb
- Página Oficial de cursos/tutoriales sobre MongoDB: https://university.mongodb.com/