MONGODB

Tematika

- MongoDB koncepció
- JSON
- „Schemaless” logika
- Replicaset képzés
- Sharding
- Aggregate framework

Koncepció

- párhuzamosítás:
  - hardver infrastruktúra adta lehetőségeket kihasználni (sok szerver, sok core etb.)
- könnyen legyen skálázható
- fejlesztések támogatása:
  - gyors és könnyű legyen változtatni
  - komplex struktúrák, struktúrűtlan vagy polimorf adatok támogatása

MongoDB=„document oriented”

- JSON: Java Script Object Notification
- nyelvfüggetlen

```
{
        x : 3,
        y : “abc”,
        z : [1,2],
        d : { }
}
```

JSON:

- struktúrált dokumentumok tárolására szolgál
- hasonló dokmentom formátum pl. XML
- Adattípusok:
  - string
  - number
  - boolean
  - NULL
  - array
  - object/document
- string típus
- asszociatív tömb,
  de csak string lehet a kulcs

```
{
  “name” : “jill”,
  “age” : 2,
  “voted”: true,
  “school”: null,
  “likes”: [“tennis”, “math”],
  “addr”: {
            “city”: “New York”,
            “addr”: “3rd Street”
            }
}
```

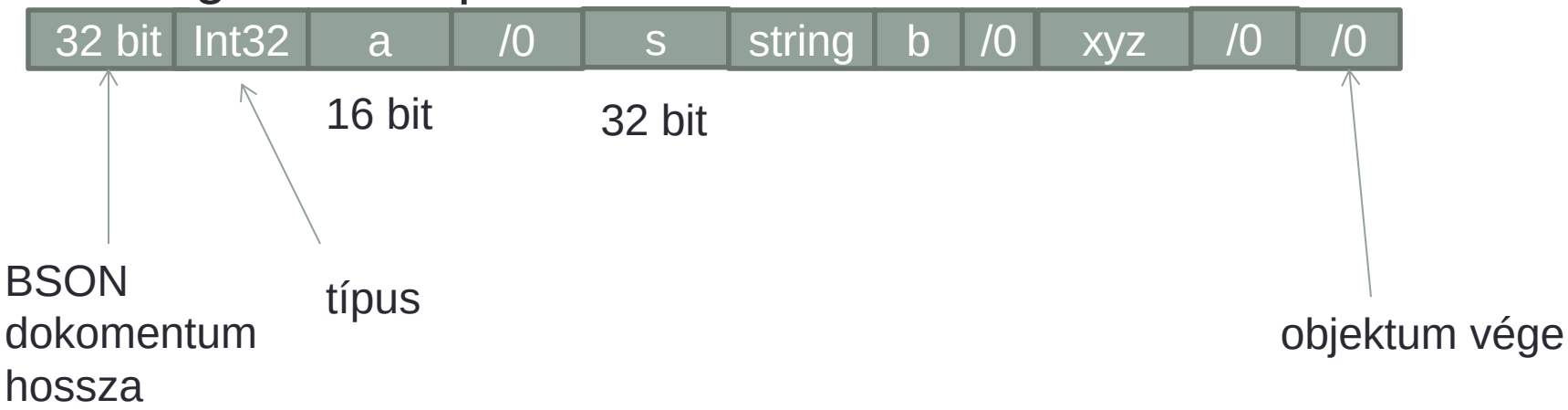
XML vs JSON

```
<phones>
        <phone type =“home”>123</phone>
        <phone type =“mobile”>456</phone>
</phones>
```

```
{
        “phones”:
                [ {“phone”: “123”, “type”: “home”},
                  {“phone”: “456”, “type”: “mobil”}]
}
```

BSON

- JSON bináris reprezentációja
- könnyen skálázható
- mongoimport BSON műveletekhez
- MongoDB-s reprezentáció:



Default: BSON objektum 16MB lehet, ha ennél nagyobb objektumaink vannak pl. videók, azokat GridFS segítségével oldhatjuk meg

Schemaless

- ez nem teljesen igaz
- dinamikus séma design
- rugalmasság
  - sokfajta és másfajta adattípusok támogatása
- statikus adatszerkezet: kevésbé erőteljes

Polimorfia

- {alakzat: “négyszög”, x: 3, y:4, “terület”: 12}
- {alakzat: “kör”, r: 1, “terület”: 3.14}

Adatok beszúrása I.

- „mongo cluster”

Adatok beszúrása II.

- “write concern”
- Az a garancia, amit a MongoDb nyújt az írási műveletek sikerességével kapcsolatban
- A garancia mértéke egyenesen arányos az írási művelet ellenőrzésének mértékével
- ha az insert, update ill. delete műveleteknél ez az ellenőrzés gyenge, akkor gyors a visszatérés
- ha erős az ellenőrzés, akkor a műveletek garantáltak de lassabb az írás
- az alkalmazás adatainak igényeihez állítható
- _id egyedi kulcs, ha nincs kitöltve a rendszer ad egy azonosítót

„Write concern” szintjei

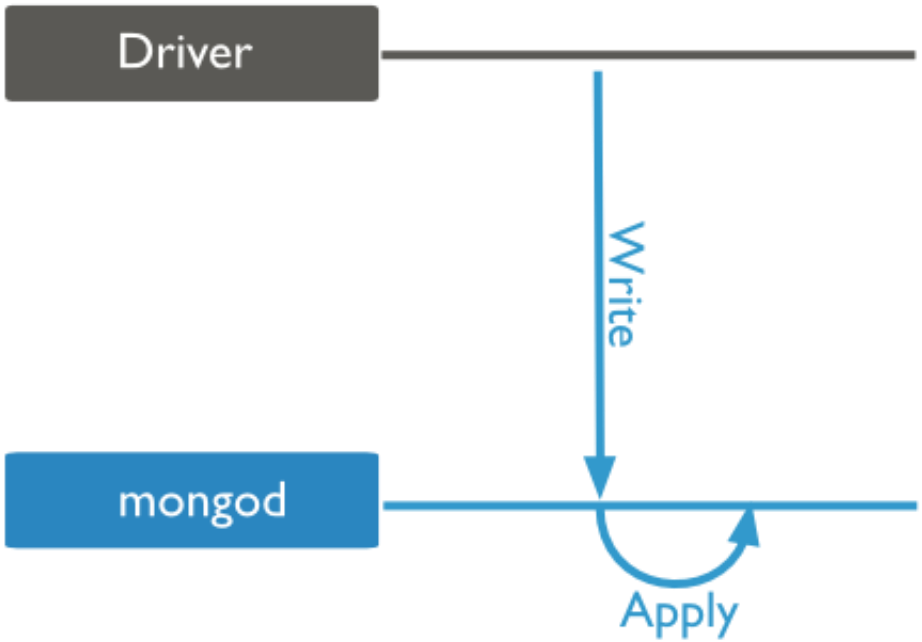
- Errors Ignored
- Unacknowledged
- Acknowledged
- Journaled
- Replica Set Acknowledged

Errors Ignored

- a MongoDB nem nyugtázza a kéréseket
- kliens nem kap visszajelzést a hibás írási műveletekről, mint pl.
  - kapcsolati hiba
  - mongod kivételek pl. kulcsütközés
- cserébe: nagyon gyors az írás
- ára: gondok lehetnek az adat perzisztenciával és tartósságával
- éles üzemben nem ajánlott
- driver beállítás: -1

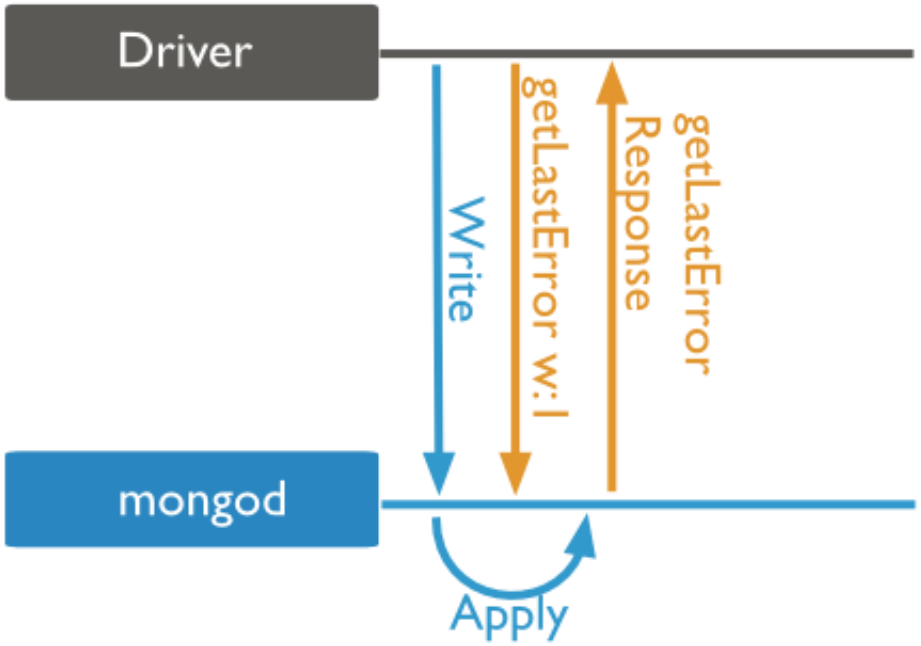
Unacknowledged

- Hasonló az előzőhöz
- Különbség: a kliens a hálózati hibákat konfigurációtól függően detektálhatja
- Driver beállítás: 0
- 2012. november release-ig ez volt a default



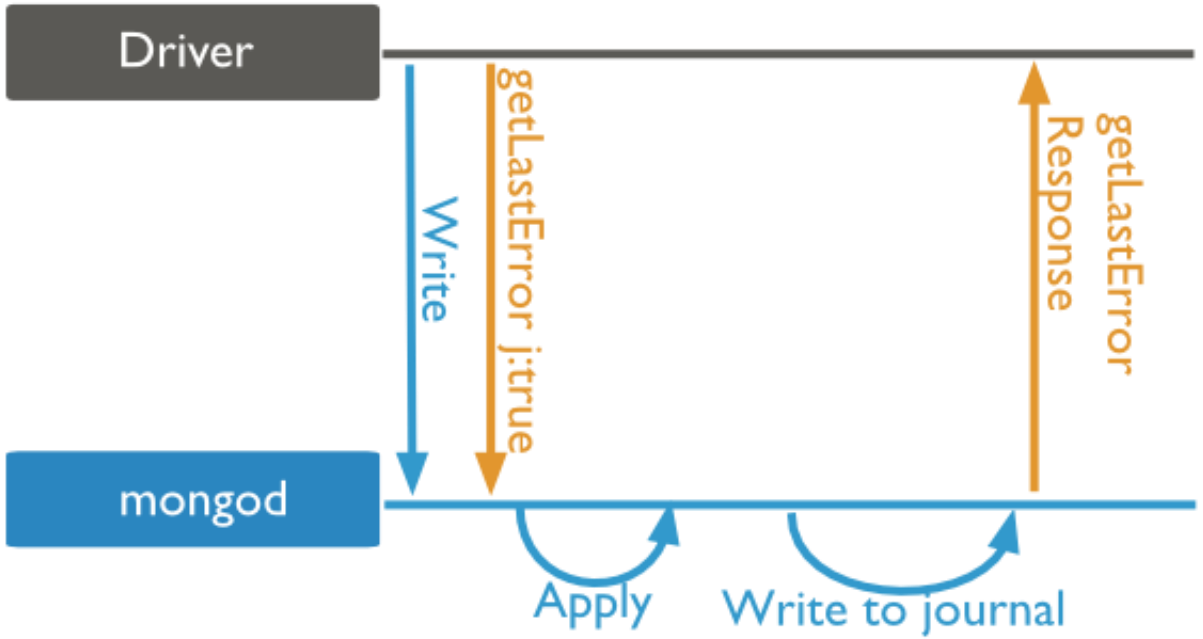
Acknowledged

- mongod ebben az esetben küld nyugtát a kliens felé
- kliens képes detektálni hálózati hibákat, kulcsütközést stb.
- driver beállítás: 1
- ez a default érték



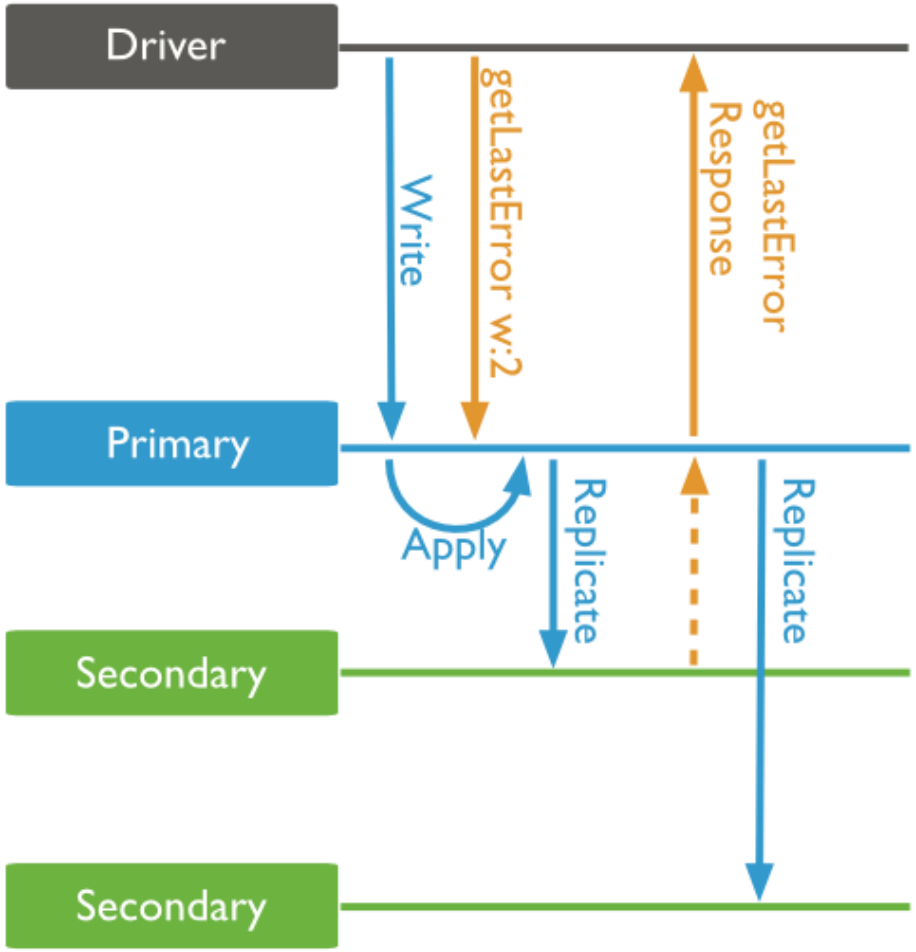
Journaled

- mongod addig nem nyugtázza az írást, amíg a journal-ba nem vezette át
- journal-t engedélyezni kell



Replica Set Acknowledged

- Alap „write concern” az egy mongod példány érint
- amennyiben replica seteket használunk, megtehetjük, hogy a nyugtázzuk az írási műveletet a replica set tagjainál is



Update

- Teljes dokumentum cseréje
- Részleges update

- _id nem változtatható update művelettel

>db.<collection name>.update(<hol>,<dokumentum>)
vagy partial update (részleges frissítés)

Delete/Remove

- több dokumentumot is töröhet egyszerre
- reguláris kifejezést is támogat

```
db.<collection name>.remove(where)
```

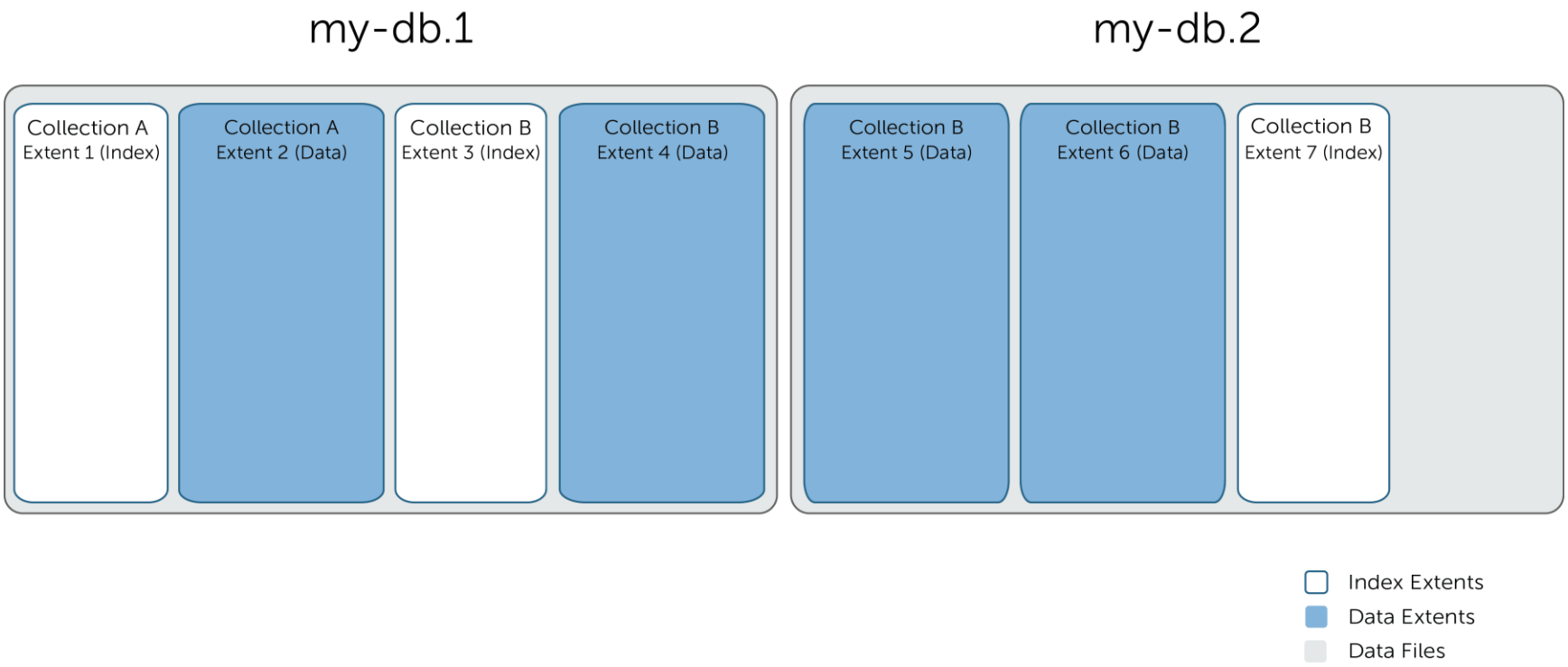
Storage

adat file

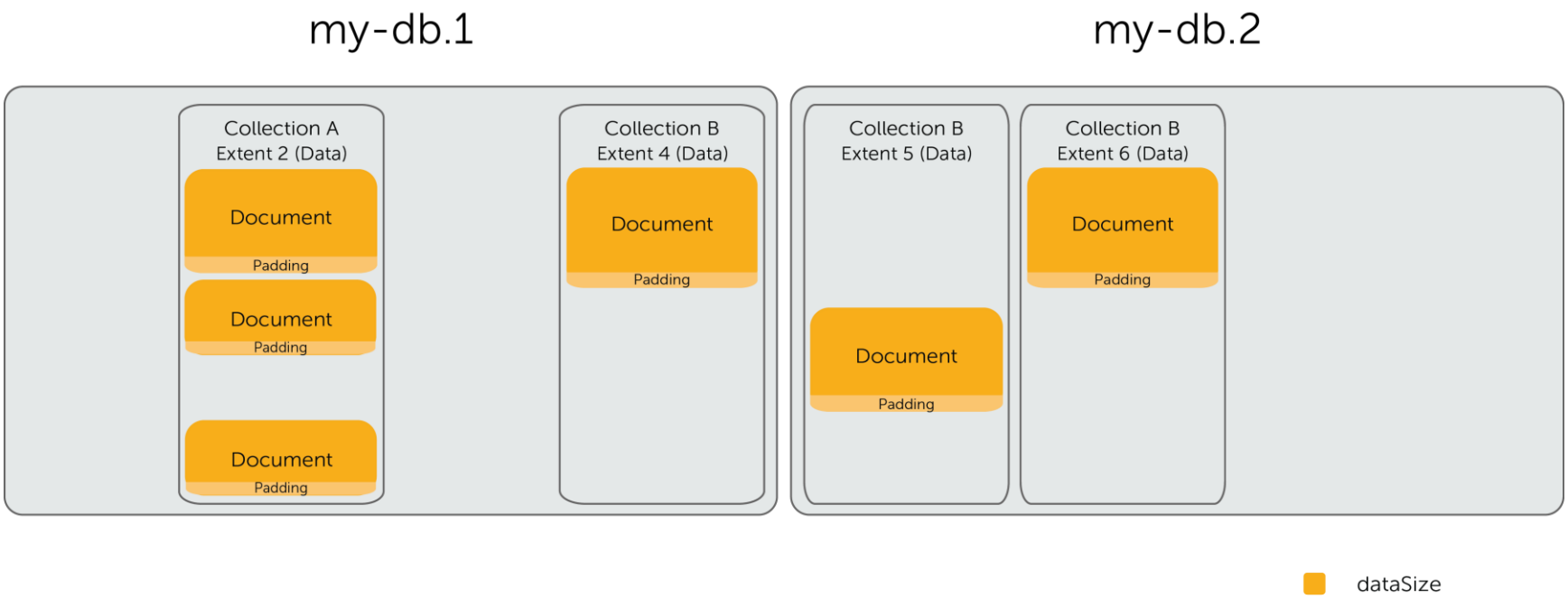
extents: egy kollekció adatait tartalmazza

record: itt vannak a dokumentumok, ennek a végén van szabad hely, azért, hogy ha nő a dokumentum, akkor legyen, amennyiben ez kevés, akkor a mongod elmozgatja

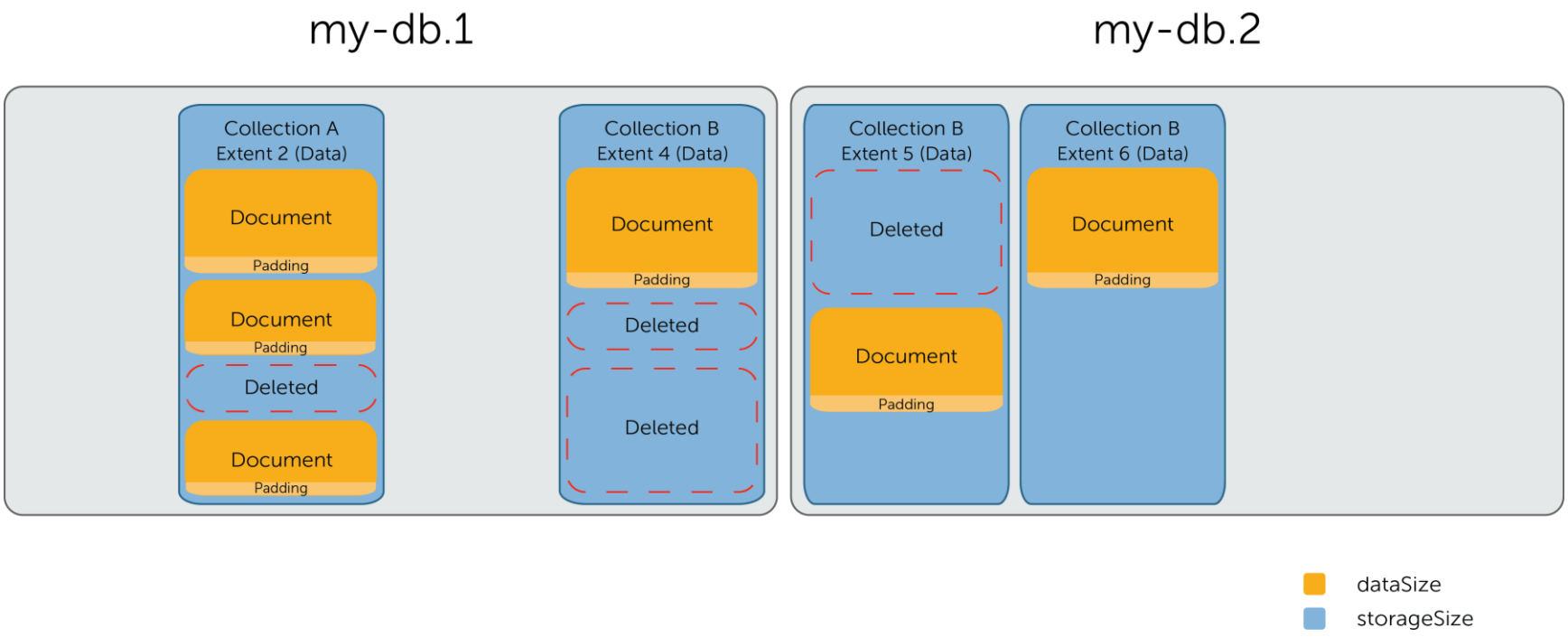
MongoDB Storage



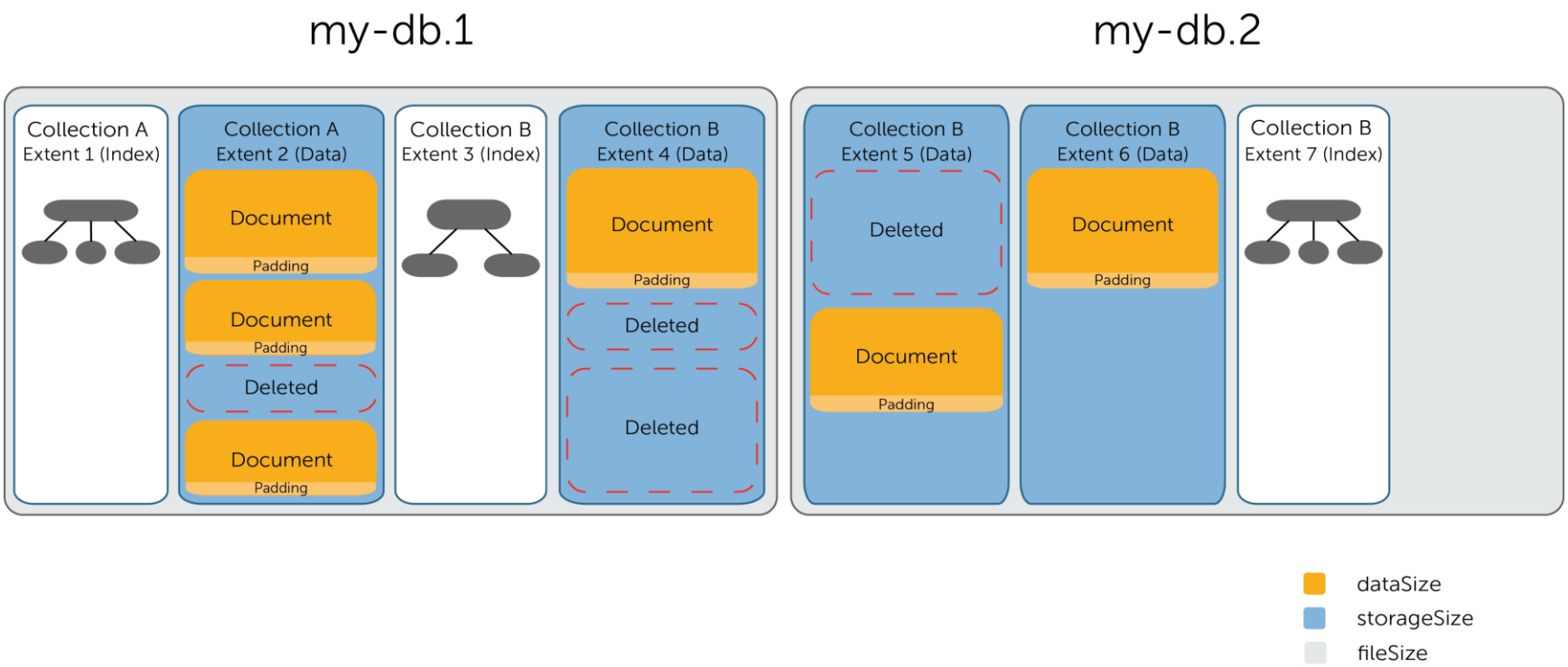
dataSize


my-db.1
my-db.2
Collection A
Extent 2 (Data)
Document
Padding
Document
Padding
Document
Padding
Collection B
Extent 4 (Data)
Document
Padding
Collection B
Extent 5 (Data)
Document
Padding
Collection B
Extent 6 (Data)
Document
Padding
dataSize

storageSize



fileSize

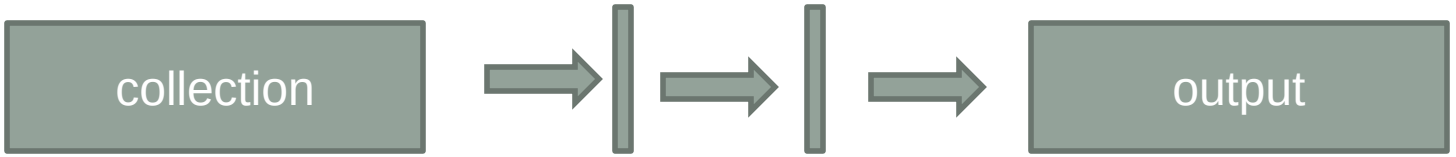

my-db.1
my-db.2
Collection A
Extent 1 (Index)
Collection A
Extent 2 (Data)
Document
Padding
Document
Padding
Deleted
Document
Padding
Collection B
Extent 3 (Index)
Collection B
Extent 4 (Data)
Document
Padding
Deleted
Deleted
Collection B
Extent 5 (Data)
Deleted
Document
Padding
Collection B
Extent 6 (Data)
Document
Padding
Collection B
Extent 7 (Index)
dataSize
storageSize
fileSize

Lekérdezések készítése

- Map/Reduce
  - built-in
  - single thread: teljesítmény limit
  - gyors és könnyű (javascript)
  - akkor ajánlott ha olyat szeretnénk, amit a beépített framework nem tud
- Aggregation framework(2.2.4-estől, aktuális verzió: 2.4.9)
  - ez a default és az ajánlott
  - teljesítmény szempontjából jobb
- Hadoop connector

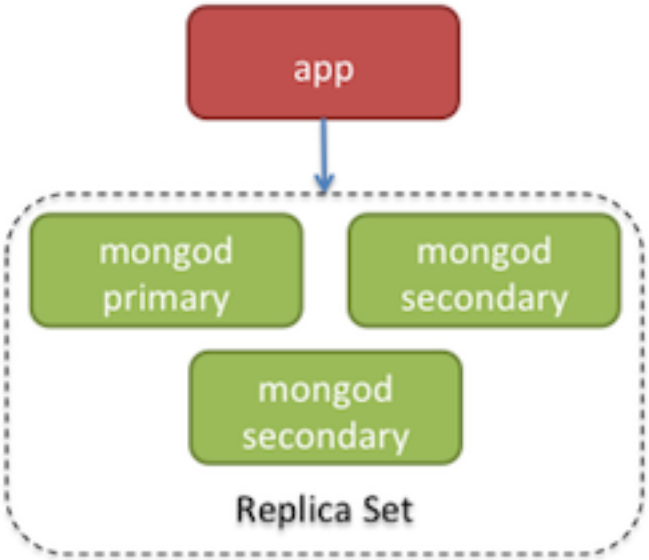
Aggregation framework

- egész a pipeline filozófiára épül
- mint a unix pipe-ok



Ezt holnap a gyakorlaton fogjuk kipróbálni

Replica set I.



- Másolatok az eredeti adatról
- egy időpillanatban csak egy primary
- Master-master replikáció megoldható „tag-aware” shardinggal
- több sort érintő műveletek pl. multi update vagy multi delete soronként replikálódnak

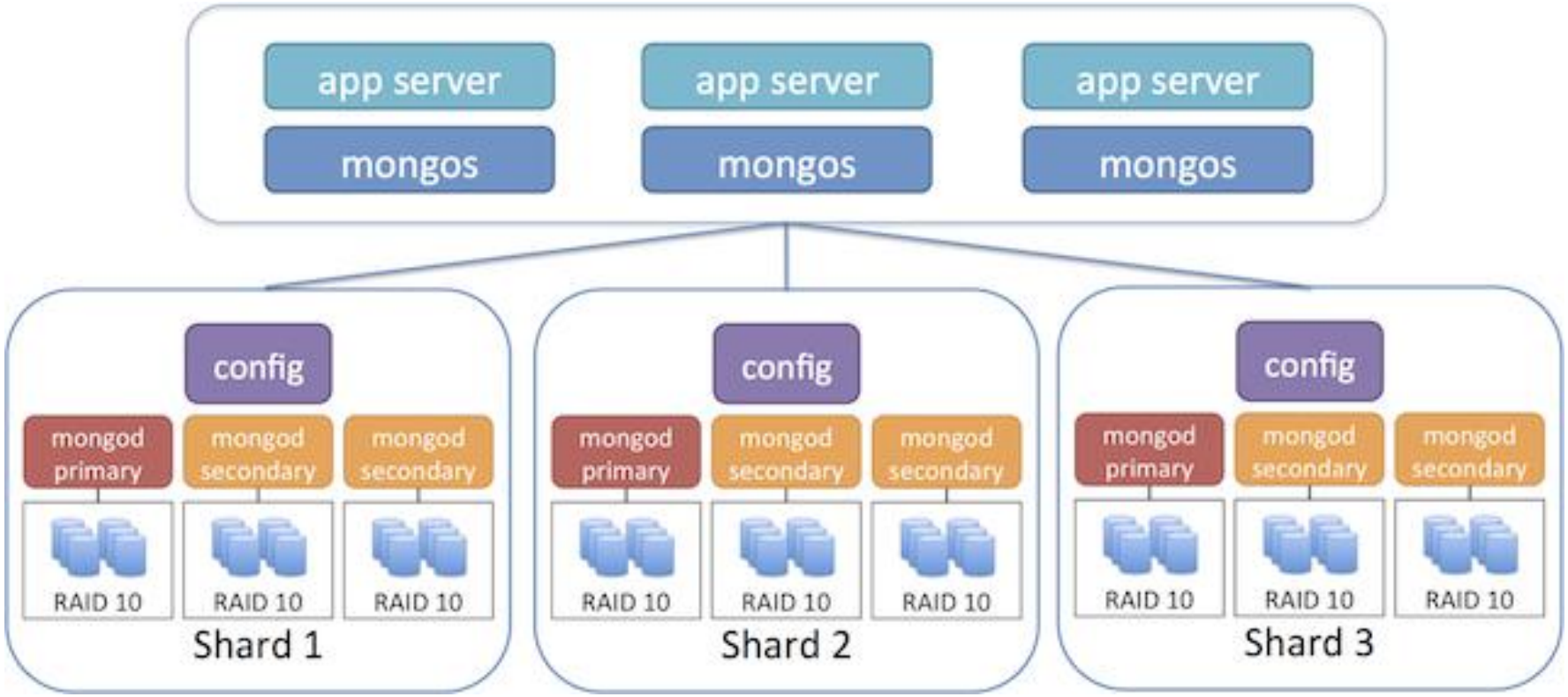
Replica set II.

- kliens mindig a primary-re kapcsolódik
- az írás mindig a primary-re megy
- az olvasás mehet a secondary-re is
- „konszenzus”: primary down esemény esetén szavazás
- atomic failover
- atomic node recovery: azon írási műveletek, amik nem mentek, visszagörgetődnek, archiválódnak és a replikéció elindul
- OpLog: replikációs művelet logja

Replica set III.

- Inicializálás:
  - konfig elkészítése
  - alap adatok
  - logikai hostneveket használjunk
  - ne használjunk IP-t, ne használjunk /etc/hosts
- a system db nem replikálódik csak az adat
- csak akkor enged a slave-ről olvasni, ha ezt explicit elfogadjuk
- a konfiguráció írása atomi művelet

Sharding


app server
app server
app server
mongos
mongos
mongos
config
mongod primary
mongod secondary
mongod secondary
RAID 10
RAID 10
RAID 10
Shard 1
config
mongod primary
mongod secondary
mongod secondary
RAID 10
RAID 10
RAID 10
Shard 2
config
mongod primary
mongod secondary
mongod secondary
RAID 10
RAID 10
RAID 10
Shard 3

Indexek

- B-fa indexek
- ensureIndex()
  - ha nagy a kollekció az index létrehozás sokáig tarhat, az index létrehozás blokkol is, ezért célszerű ezt karbantartás alatt végezni
- _id implicit létrejön
- minden egyéb esetben explicit kell létrehozni
- olvasást gyorsítja, írást lassítja

Mentés és helyreállítás

- mongodump –oplog
- file rendszer snapshot – journaling enabled
  - db.fsyncLock()
  - db.fsynchUnlock()
  - nincsenek query ilyenkor
- secondary-ről mentés: shutdown, copy, restart
- Backup és sharding:
  - Balancer kikapcs
    - sh.stopBalancer(), chunk migrálás miatt
  - config db backup
    - mongodump –dbconfig
  - backup minden shard replica setre
  - sh.startBalancer()

Hasznos linkek

- http://en.wikipedia.org/wiki/Mongodb
- http://docs.mongodb.org/manual/core/sharding/
- http://docs.mongodb.org/manual/core/write-concern/
- http://blog.mongolab.com/2014/01/how-big-is-your-mongodb/

- Letölthető:
- http://www.mongodb.org/downloads

Kérdések és válaszok