- mamw dokumencie wszystko, szybsz i lepsza skalowalnosc

- wybor miedzy szybkoscia a consistency na wilu serwerach,

- replikuje na wszystkie przy zapisie, na wybrane, potem- zwraca odpowiedz nie czekajac az sie zreplikuje – fire and forget, moze powodowac, ze record zaginie, powstaja replication lagi, co prze duzym uzyciu aplikacji powoduje, ze jeden zapisze do primary a inny w miedzyczasie z secondary czyta stare

- cappend collection – utrzymuje zafixowana ilosc dokumentow, dzieki temu kontrola nad rozmiarem

- nie wymagana schema na dokumenty

- po sciagnieciu i wypakowaniu duzo plikow, ale glownie uzywane mongodump – serwer, i mongo – shell dla operacji

- pozostale to narzedzia np. do impoer/export, monitor, shard

- przy pierwszym odpaleniu ustawiamy gdzie 3mane pliki

-- default/data/db/

- mongod – startuje serwer, mowi jaki port

- opcje dodatkowe, gdzie logi, port, ile connection, directory etc. (dbpath, logpath)

- mozna ustawic w pliku i odpalenie mongod –f mongod.conf

- chcemy zeby chodzil ciagle, w windows as a service (--install), w linux skryptem, do tego trzeba admina

- potem start net start mongodb

- polaczenie – mongo – do default bez parametrow, exit to wyjscie

- show dbs – listuje bazy, na wejscie local, wewnetrzna dla mongo

- use foo – laczy do bazy foo

- replica potrzebuje minimal set up, mozemy ustawic failover

--

- kazdy document musi byc ID, i tyle, domument nie moze przekraczac duzego rozmiaru

- mamy wlasne kolekce dokumentow i mamy system.indexes

-- index po id z default

- tylko array nie moze byc id, mozna ominac po konwersji na byte

- ObjectId() – zwraca randomowy id

- ma timestamp ten objectId – daje nam dokument creation time

- jest Bson ObjectId w java

- to daje dobre insert, szybkie dodawanie,

- jak ten sam id, to nadpisuje nowym – save

- jak uzywam insert to nie pozwoli z tym samym id

- @Field – zapisuje pole pod inna nazwa w dokumencie

- @DBRef – dziala jak foreign key

- @TypeAlias, zamienia nazwe klasy nad podany przez nas string, nad klasa

- dokumenty w tej samej kolekcji niezalezne, moga byc calkiem rozne , moze byc uzyte przez dziedziczenie obiektow

- update jest atomowe, jak 2 to po sobie

- jak 2 update niezalezne i oddzielne kawalki, to merge zmian robi

- atomowe update za pomoca findAndModify i update komendy

- przy wydobyciu mozliwa projekcja, zeby calego dokumentu nie pobierac z bazy

- przy szukaniu w tablicach musi byc dokladne dopasowania – case sensitive

- jest np. cos w liscie tagow in, nin, all co zawieraja dany tag

- jest notacja kropki, mozeby sie zagniezdzac, znajdz gdzie info.type= i info.name=

- poza tym schema elastyczna i pola nie wymagane w dokumentach

- jak szukamy po poddokumencie to musi byc jego dokladne dopadowanie (wszystkie pola i kolejnosc), wic lepiej notacji kropki uzywac

- jak szukamy po takim co xx=null, to wyszukia takie co maja pole = null, badz nie maja pola wogole wypelnionego

- moge poszukac z exists takie co maja to pole (moze byc null)

- , poiedzy kryteraimi w zapytaniu dziala jako end

- w latwy sposob przy projekcji mozna jako 1 pole include, 0 to exclude, wypisze wybrane pola , badz pozostale z excludowanych

- id to jedyne, co zawsze zwracane jak nie wybrane i trzeba excludowac

- NIE mozna mixowac includ i exclude – jest albo albo

- podczas wydobycia mongo ma kursor, dane moga sie nie miescic do pamieci naszej (jak b. Duzo dokumentow), ale mogo zwraca batch i przesuwa kursor , na koniec zamyka kurso, jako sygnal

- mongo zwraca kursor do klienta, mozna spr ile elementow i przez niego iterowac

- sortowanie po stronie serwera, mozna po dowolnym elemencie poddokumentu

- mamy limit, by okreslana liczbe pobrac resultatow

- mozna pageowanie po stronie serwera ze skip and limit , skip 5 i limit 5 – to od 6-10 da

- findOne() – zwroci jeden rekord, nie kursor

Index

- bez index, przechodzi przez kazdy dokument i sprawdza x w nim, wolne

- index ma kursor do dokumentu z wartoscia – do lokacji na dysku

- sortowanie uzywa indexow

- rozne indexy, regular-btree, text index, geo – do geografii np. szukania reastauracji, hash index, i ttl lindex, time to live (do expired, nie trzeba samemu, tylko zalatwia to mongo )

W spring :

@Indexed(name = "first\_name\_index", direction = IndexDirection.DESCENDING)

@Indexed(name = "expire\_after\_seconds\_index", expireAfterSeconds = 10)

Ale musi data byc

@Indexed(name="someDateFieldIndex", expireAfterSeconds=3600)

Date someDateField;

Mozna listowac jakie indexy mamy, id jest default

- mozna po kazdym zapytaniu explain(), mowi jakie uzywal indexy, ile dok skanowal,

- tez na pola porownywane <>

- jak chcemy zmienic definicje indexu to drop robimy starego

- jak indexujemy tablice, to kazdy elem bedzie indexowany jako wartosc, latwe szukanie w tagach daje

- dodatkowo index uniques jest

- sparse index, robi index tylko na dok co posiadaja dane pole, trzeba pamietac, ze nawet sort na tym pomija te co nie maja pola elementy

- mozna komponowane indexy, ale musi match od lewej do prawej

- jak pobieramy tylko pole indexowane, to b szybkie, bo nie szukamy na dysku tylko zwracamy z indexu

- jak index na wilu polach to nazwa to ich kompozycja