**Interna struktura i organizacija indeksa MongoDB baze podataka**

Kristina Antić, 934

Sadržaj

[1. uvod 3](#_Toc36555648)

[1.1. Organizacija fajlova 3](#_Toc36555649)

[1.2. Indeksi 3](#_Toc36555650)

[1.2.1. Podela indeksa 5](#_Toc36555651)

[1.2.2. Strukture podataka za organizaciju indeksa 6](#_Toc36555652)

[2. indeksi mongodb baze podataka 8](#_Toc36555653)

[2.1. Tipovi indeksa 9](#_Toc36555654)

[2.2. Svojstva i upravljanje indeksima 15](#_Toc36555655)

[2.3. Index intersection 18](#_Toc36555656)

[2.4. Praćenje korišćenja indeksa 19](#_Toc36555657)

[2.5. Strategije indeksiranja 23](#_Toc36555658)

[3. zaključak 23](#_Toc36555659)

[literatura 24](#_Toc36555660)

# uvod

DBMS čuva veliku količinu podataka, i ti podaci moraju postojati tokom izvršavanja programa. Najjednostavnija apstrakcija podataka u DBMS-u je kolekcija record-a, ili ***fajl***, gde se svaki fajl sastoji od jedne ili više stranica. Podaci se čuvaju na eksternim uređajima za skladištenje, kao što su diskovi i trake, i pribavljaju se u glavnu memoriju kada su potrebni za obradu.

Jedinica informacija koja je pročitana sa diska ili upisana na disk je ***stranica*** (***page***). Veličina stranice je parametar DBMS-a i obično je veličine 4KB ili 8KB. Troškovi I/O (čitanje/pisanje) stranice dominiraju u odnosu na tipične operacije baze podataka, i sistemi baza podataka se pažljivo optimizuju kako bi se ti troškovi smanjili.

Da bi se što bolje i efikasnije upravljalo podacima i njihovim korišćenjem bitna je odgovarajuća organizacija fajlova i indeksa. Organizacija fajlova i indeksiranje će biti detaljnije opisani u nastavku.

## Organizacija fajlova

***Organizacija fajlova*** je način uređenja record-a (slogova) u fajlu kada je fajl sačuvan na disku. Svaka organizacija fajlova je dobra za neke operacije jer ih čini efikasnim, a za neke nije jer ih čini skupim. Svaki slog u fajlu ima svoj jednistveni identifikator (***rid – record id***) koji omogućava fizičko lociranje sloga, odnosno identifikaciju adrese stranice na disku koja sadrži slog sa određenim identifikatorom.

Postoje različite alternative za organizaciju fajlova. Najčešće korišćene alternative su:

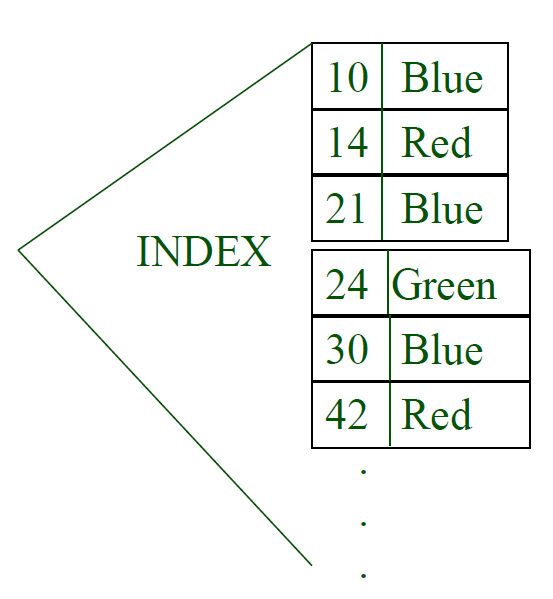
1. *Heap files* – najjednostavnija struktura fajla je neuređeni fajl, ili heap fajl. Slogovi se u fajlu nalaze u onom redosledu u kome su ubacivani, što znači da se novi slogovi dodaju na kraj fajla. Ova alternativa je pogodna kada se koristi tipičan pristup *file scan* koji preuzima sve slogove.
2. *Sorted files* – slogovi u fajlu su uređeni na osnovu vrednosti nekog polja. Ova alternativa je pogodna kada je slogove potrebno vratiti u nekom redosledu, ili ako je potrebno vratiti samo skup slogova iz nekog opsega.
3. *Hashed files* – organizacija fajla se bazira na heširanju, što obezbeđuje brz pristup slogovima pod određenim uslovima pretrage. Uslov pretrage mora da bude uslov jednakosti koji se odnosi na jedno polje, heš polje. Ideja je da se heš funkcija primeni na vrednost nekog polja i da se dobije adresa bloka na disku gde će slog biti sačuvan. Pristup slogovima je jako brz.

## Indeksi

***Indeks*** je struktura podataka koja organizuje slogove na disku kako bi se optimizovalo preuzimanje slogova kod određenih vrsta pretrage. Indeks omogućava efikasno preuzimanje svih slogova koji zadovoljavaju uslov pretrage koji se odnose na ***search key (ključ pretrage)*** polja indeksa. Može se kreirati više indeksa nad određenom kolekcijom slogova sa različitim ključevima pretrage. ***Search key*** može biti bilo koji atribut ili deo atributa i koristi se kao ključ za pretraživanje definisanog indeksa nad skupom slogova.

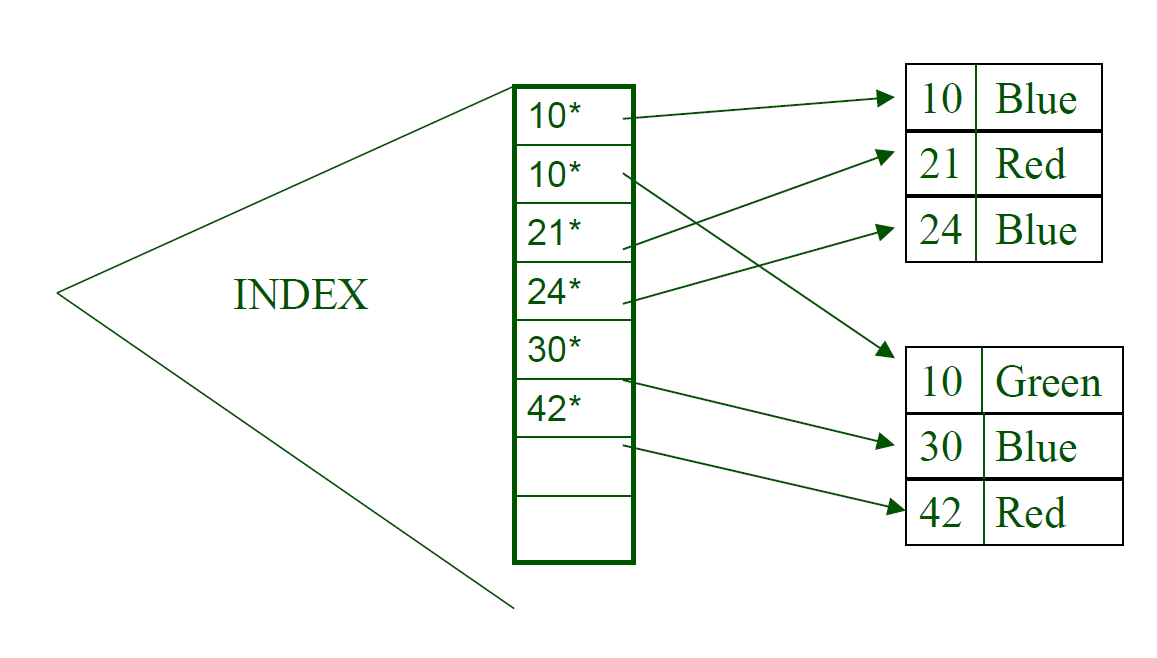
Pojam ***data entry*** se koristi za slogove koji se nalaze u indeksu. Postoji više alternativa šta se čuva kao *data entry* ***k\****:

1. Data entry k\* predstavlja ceo slog sa ključem pretrage k. Indeksna struktura se može posmatrati kao neka vrsta organizacije fajlova. Za neku kolekciju podataka može da postoji samo jedan indeks ovog tipa. Inače dolazi do dupliranja slogova, redundantnosti podataka i potencijalne nekonzistentnosti. Na slici 1.2.1 prikazan je izgled ovakvog indeksa.



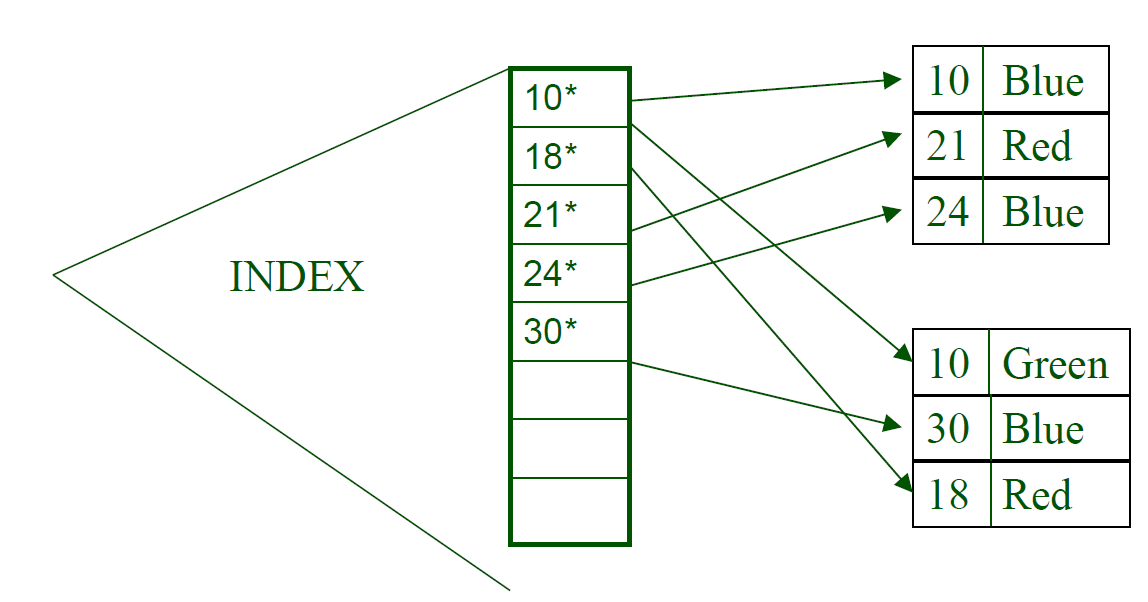
Slika 1.2.1 Indeks sa celim slogom kao data entry-jem

1. Data entry je *(k, rid)* par, gde je *rid* id sloga sa vrednošću ključa pretrage k. Ova alternativa je nezavisna od organizacije fajla. Data entries su mnogo manji nego u prethodnom slučaju. Izgled ovakvog indeksa prikazan je na slici 1.2.2.



Slika 1.2.2 Indeks sa (k, rid) data entry-jem

1. Data entry je *(k, rid-list)* par, gde je *rid-list* lista id-jeva slogova koji imaju vrednost ključa pretrage k. I ova alternativa je nezavisna od organizacije fajla i data entries su manji nego u prvom slučaju. Izgled ovakvog indeksa prikazan je na slici 1.2.3.



Slika 1.2.3 Indeks sa (k, rid-list) data entry-jem

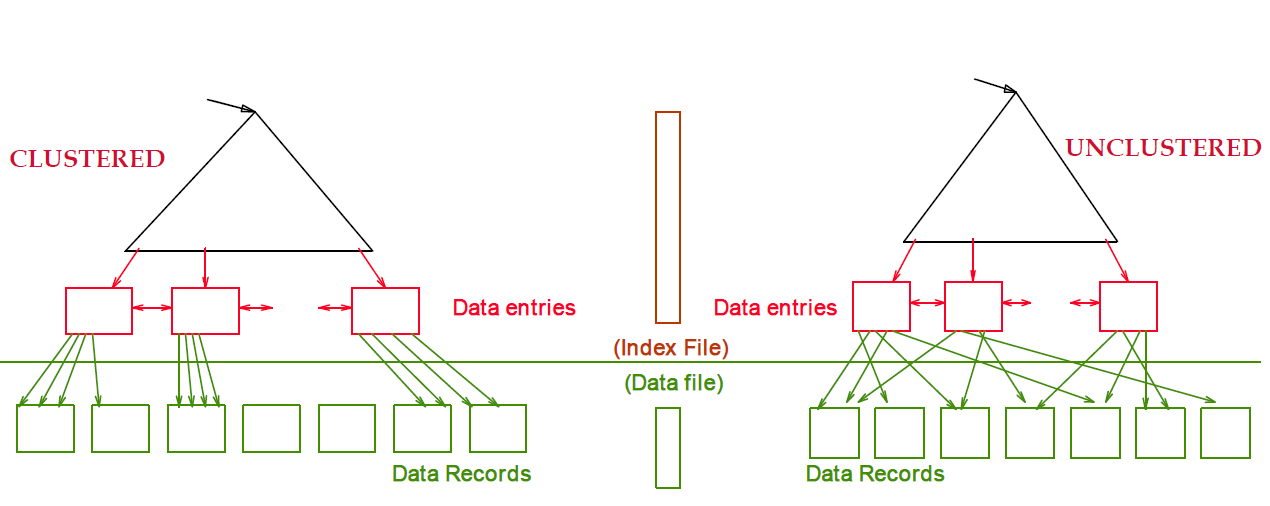
## Podela indeksa

* **Primarni i sekundarni indeksi**

Ukoliko ključ pretrage sadrži primarni ključ, onda se indeks naziva ***primarni***. Ostali indeksi su ***sekundarni***. Dva data entry-ja su ***duplikati*** ukoliko imaju istu vrednost ključa pretrage povezanog sa indeksom. Primarni indeks ne sadrži duplikate, ali indeks nad drugim poljima može da sadrži duplikate. Ukoliko nema duplikata, takav indeks se zove ***unique*** indeks.

* **Klasterovani i neklasterovani indeksi**

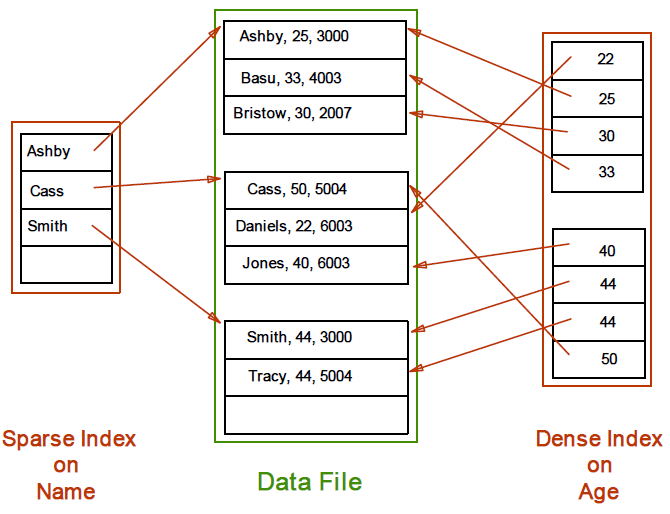
Kada je fajl organizovan tako da je redosled slogova isti ili skoro isti kao redosled data entries u nekom indeksu, tada se kaže da je indeks ***klasterovan***. U suprotnom je ***neklasterovan***. Ilustracija klasterovanog i neklasterovanog indeksa prikazana je na slici 1.2.1.1.



Slika 1.2.1.1 Klasterovan i neklasterovan indeks

* **Gusti i retki indeksi**

***Gusti (dense)*** indeks ima barem jedan data entry za svaku vrednost ključa traženja u nekom slogu podataka. U suprotnom, indeksi su ***retki (sparse)***. Na slici 1.2.1.2 prikazan je izgled retkog i gustog indeksa.



Slika 1.2.1.2 Gust i redak indeks

## Strukture podataka za organizaciju indeksa

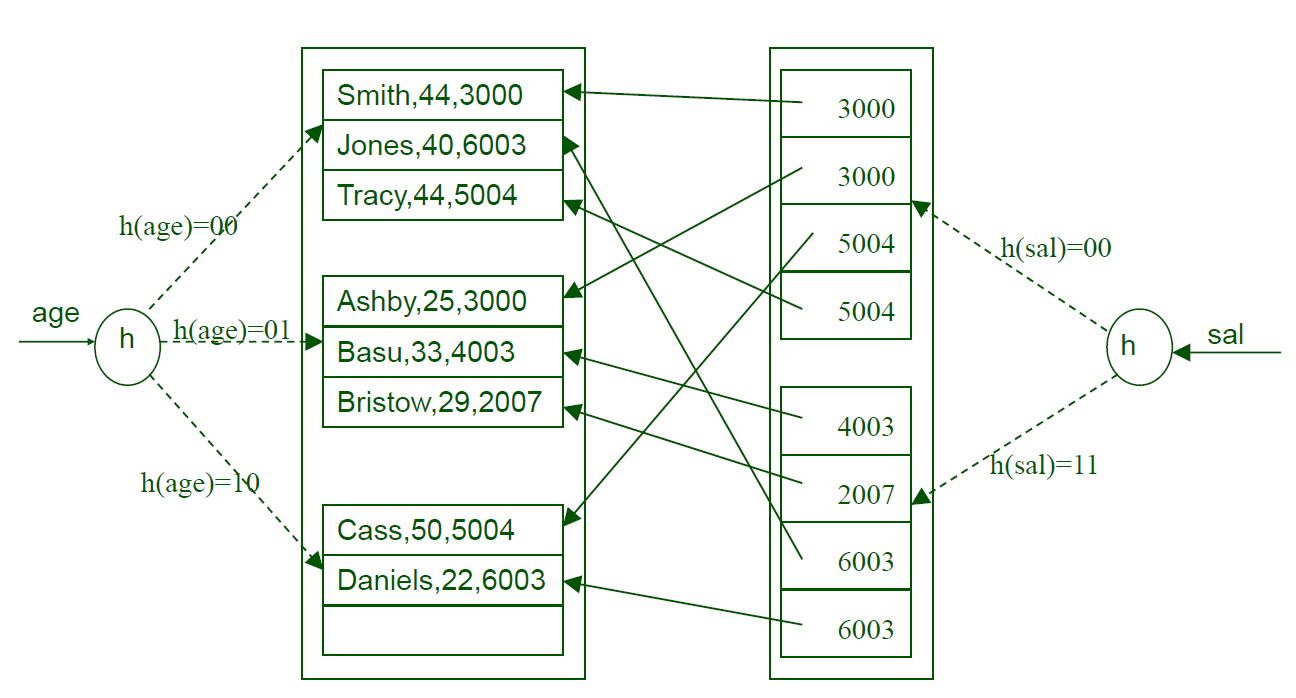
Jedan način za organizaciju data entries u indeksu je ***heširanje*** data entries po ključu pretrage. Drugi način za organizaciju data entries je pravljenje ***strukture stabla*** koja preusmerava traženje data entries. I heš indeksiranje i indeksiranje pomoću stabla se može kombinovati sa bilo kojom od tri alternative za data entries.

* **Hash-based indeksi**

Data entries se mogu organizovati tehnikom koja se naziva ***heširanje*** kako bi se brzo pronašli oni slogovi koji imaju određenu vrednost ključa pretrage. U ovom pristupu, indeks je kolekcija ***bucket-a***. Data entries se organizuju u bucket-e, gde se bucket sastoji od ***primarne*** ***strane*** i, po potrebi, dodatnih, ***overflow*** ***strana*** povezanih u lanac. Bucket kome data entry pripada se može pronaći pomoću posebne funkcije, ***heš funkcije***, primenjene na ključ pretrage.

Ovi indeksi su dobri za pretragu po jednakosti i ne mogu da podrže selekciju po opsegu.

Heš indeksiranje je ilustrovano na slici 1.2.2.1, gde je prvi indeks heširan na osnovu age polja i koristi se alternativa 1 za data entries, a drugi indeks je heširan na osnovu salary polja i koristi se alternativa 2 za data entries.



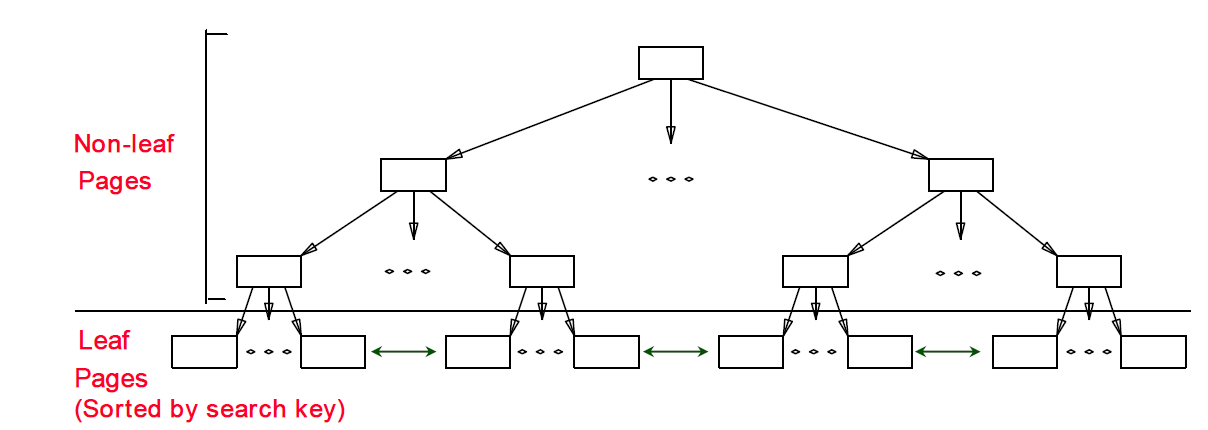
Slika 1.2.2.1 Hash-based indeksi

* **Tree-based indeksi**

Altenativa hash-based indeksiranju je organizacija indeksa korišćenjem stabla, i to uglavnom B-stabla. Data entries su uređeni po vrednosti ključa pretrage i održava se hijerarhijska struktura podataka koja usmerava pretrage do odgovarajuće stranice sa data entries.

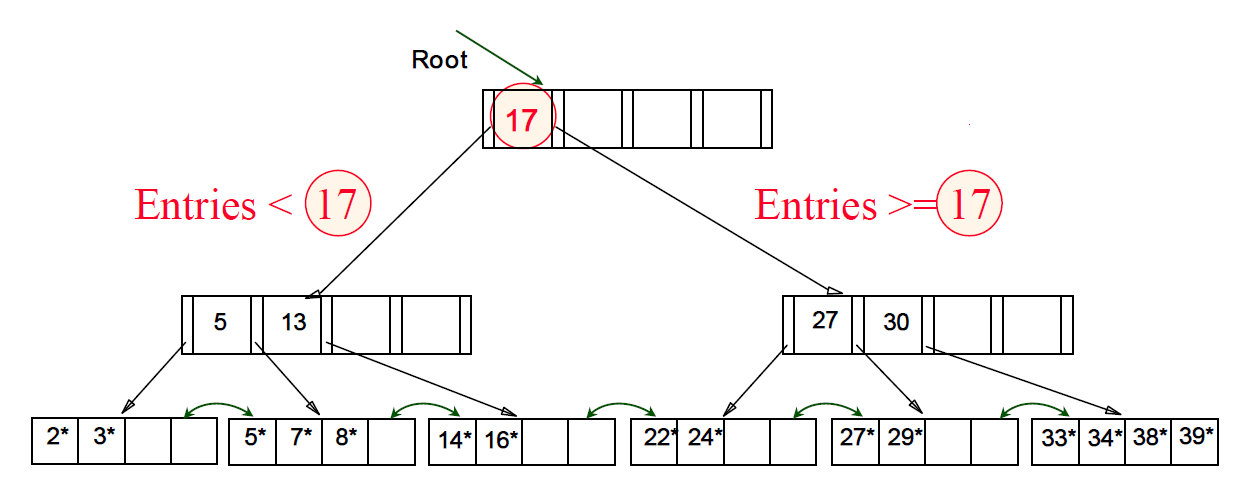
Listovi stabla sadrže *data entries*, koji su sortirani po vrednosti ključa pretrage. Ostali čvorovi sadrže *index entries* i koriste se samo tokom pretrage. Svaki čvor predstavlja po jednu fizičku stranicu.

Često se koristi i alternativa, B+ stablo, kod kog unutrašnji čvorovi sadrže kopije ključeva, dok listovi sadrže ključeve i vrednosti (index entries) i međusobno su ulančani. Ilustracija stabla prikazana je na slici 1.2.2.2.



Slika 1.2.2.2 Struktura stabla

Ova struktura omogućava efikasno lociranje data entries sa vrednošću ključa pretrage u određenom opsegu. Svaka pretraga počinje u root čvoru stabla. Unutrašnji čvorovi stabla sadrže pointere i vrednosti ključa pretrage. Pointer levo od vrednosti ključa pretrage *k* ukazuje na podstablo koje sadrži data entries manje od *k*. Pointer desno od vrednosti ključa pretrage *k* ukazuje na podstablo koje sadrži data entries veće ili jednake *k*. Na slici 1.2.2.3 prikazan je izgled jednog tree-based indeksa. Ovi indeksi su najbolji kod traženja po opsegu, a mogu se iskoristiti i za pretragu po jednakosti.



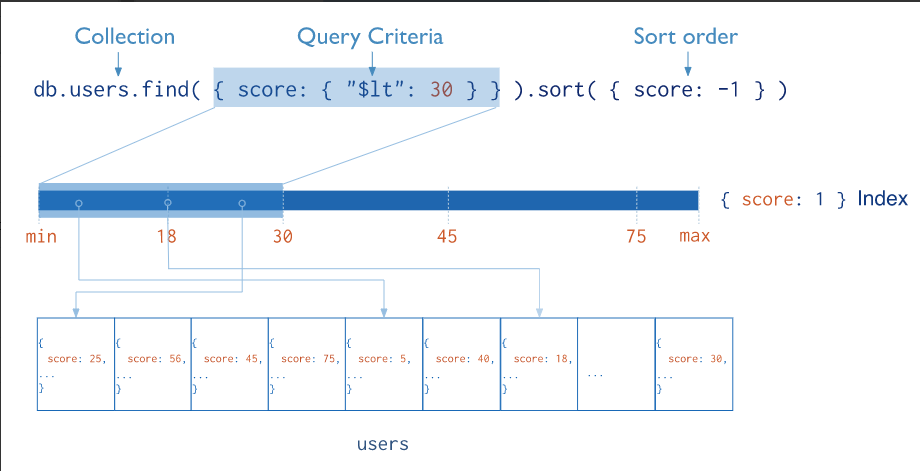
Slika 1.2.2.3 Tree-based indeks

# indeksi mongodb baze podataka

Indeksi podržavaju efikasno izvršavanje upita u MongoDB bazi podataka. Bez indeksa blio bi neophodno izvršiti skeniranje kolekcije, odnosno skeniranje svakog dokumenta (podaci se skladište u obliku JSON dokumenata i MongoDB ne vrši uređenje dokumenata na disku) u kolekciji kako bi se izdvojili oni koji odgovaraju upitu. Ukoliko postoji indeks, MongoDB može da ga koristi kako bi smanjio broj pregledanih dokumenata.

Indeksi su posebne strukture podataka koje čuvaju mali deo podataka iz kolekcije. MongoDB indeksi koriste strukturu B-stabla (tačnije, **B+** stablo). Za čuvanje *data entries* (*index entries*) koristi se alternativa 2. Indeks čuva vrednosti određenog polja ili skupa polja, pri čemu su vrednosti uređene. Uređenje vrednosti omogućava efikasnu pretragu po jednakosti i upite koji vraćaju podatke iz nekog opsega. MongoDB može da vrati uređene rezultate korišćenjem uređenja indeksa. Uređenje je moguće samo za polja koje čine indeks. Na slici 2.1 ilustrovano je izvršenje upita koji vraća uređene dokumente korišćenjem indeksa nad poljem *score*.

MongoDB definiše indekse na nivou kolekcije i podržava indeksiranje nad bilo kojim poljem, delom polja ili nad više polja dokumenta u kolekciji. Kada upit sadrži ***samo indeksirana polja*** MongoDB vraća rezultate direktno iz indeksa, bez skeniranja bilo kog dokumenta ili učitavanja dokumenata u memoriju. Ovi upiti su jako efikasni.

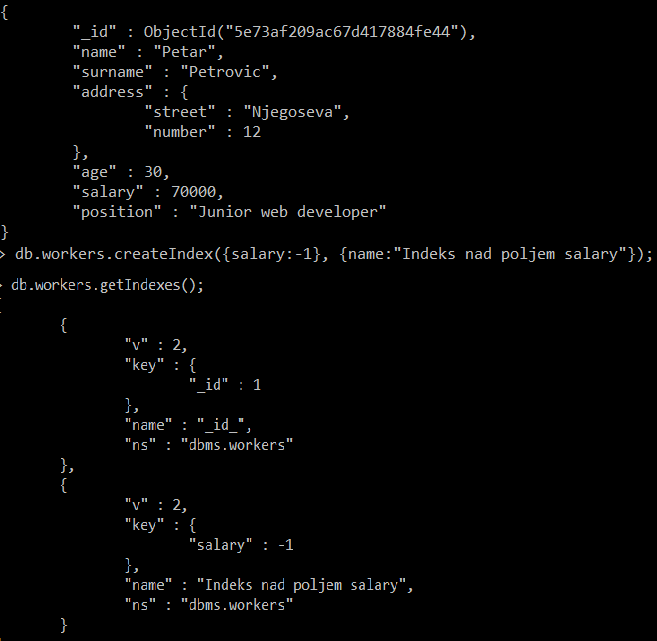


Slika 2.1 Izvršenje upita korišćenjem indeksa

MongoDB kreira ***podrazumevani*** ***unique*** ***indeks*** ***nad*** ***\_id*** poljem prilikom kreiranja kolekcije. Za *\_id* polje se uglavnom koristi *ObjectId* vrednost. Ovaj indeks nije moguće obrisati.

Indeks se kreira korišćenjem ***createIndex(keys, options)*** metoda. ***Keys*** su parovi ***polje:vrednost***, gde se specificiraju polja nad kojima se kreira indeks i, kao vrednost, se specificira da li je indeks sortiran u rastućem (vrednsot 1) ili opadajućem (vrednost -1) redosledu. ***Options*** obauhvata neke dodatne opcije, ne primer, naziv indeksa, da li je unique, itd.

Podrazumevani naziv indeksa je konkatencija naziva polja nad kojima je kreiran indeks i smerova sortiranja za svako polje uz korišćenje donje crte za razdvajanje. Indeksu se može dodeliti i proizvoljan naziv prilikom kreiranja korišćenjem *options* parametra u naredbi za kreiranje. Svi indeksi nad nekom kolekcijom se mogu videti korišćenjem ***getIndexes()*** metoda. Na slici 2.2 prikazan je način kreiranja indeksa nad *salary* poljem *workers* kolekcije, sortiranog u opadajućem redosledu i sa dodeljenim nazivom, kao i prikaz svih kreiranih indeksa nad tom kolekcijom.



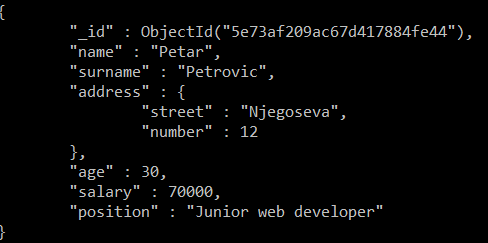
Slika 2.2 Kreiranje indeksa i prikaz svih indeksa nad kolekcijom

## Tipovi indeksa

1. **Single Field indeksi**

Pored podrazumevanog \_id indeksa, MongoDB podržava kreiranje korisnički definisanih rastućih/opadajućih indeksa nad jednim poljem dokumenta – ***single field*** indekse. U prethodnom primeru na slici 2.2 kreiran je ***single field*** indeks koji podržava upite koji vrše selekciju na osnovu *salary* polja.

***Single field*** indeks se može kreirati i nad poljem čija je vrednost neki ugnježdeni dokument, kao i nad bilo kojim poljem ugnježdenog dokumenta. Na slici 2.1.1 je prikazan izgled jednog dokumenta koji sadrži ugnježdeni dokument u okviru polja *address*.



Slika 2.1.1 Ugnježdeni dokument u okviru *address* polja

U okviru primera je kreiran jedan indeks nad poljem *address* i jedan indeks nad poljem *street* ugnježdenog dokumenta. Primer je ilustrovan na slici 2.1.2.

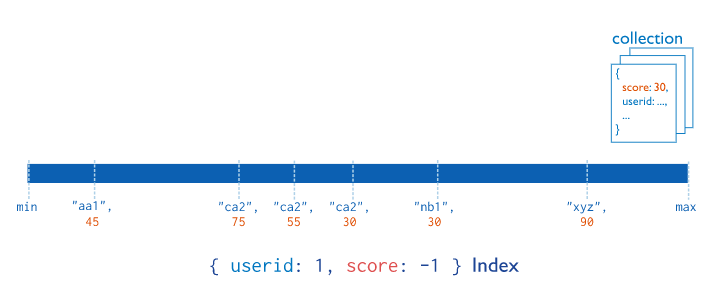


Slika 2.1.2 Indeksi nad ugnježdenim dokumentima

***Single field*** indeksi podržavaju upite sa sortiranjem bez obzira da li se smer sortiranja naveden u upitu poklapa sa smerom sortiranja polja u indeksu jer se može okrenuti smer sortiranja u indeksu.

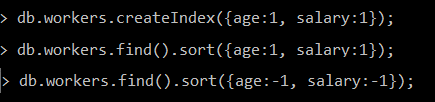
1. **Compound indeksi**

MongoDB podržava i korisnički definisane indekse nad više polja (limit je 32 polja), ***compound indekse***. Redosled navođenja polja prilikom kreiranja indeksa je od značaja. Na primeru na slici 2.1.3 indeks je kreiran nad poljima *userid* i *score*. Najpre se indeks sortira na osnovu vrednosti *userid* polja, a zatim se za svaku *userid* vrednost vrši sortiranje po *score* vrednostima.



Slika 2.1.3 Compound indeks

Kod compound indeksa kada upit zahteva sortiranje, da bi indeks mogao da se iskoristi, smer sortiranja polja naveden u upitu i smer sortiranja polja u indeksu su bitni. Mogu se koristiti isti smerovi svih polja kao u indeksu ili, ukoliko je smer obrnut u odnosu na indeks, mora biti obrnut za ***sva*** polja. Takođe, polja moraju biti navedena u ***istom*** redosledu kao što se nalaze u indeksu. Na slici 2.1.4 kreiran je indeks nad poljima *age* i *salary workers* kolekcije i prikazani su upiti sa sortiranjem koje ovaj indeks podržava.



Slika 2.1.4 Compound indeks i podržani upiti sa sortiranjem

Osim upita koji se poklapaju po svim poljima sa indeksom, compound indeksi podržavaju i upite koji se poklapaju po ***prefiksu*** indeksiranih polja. ***Prefiksi indeksa*** su početni podskupovi indeksiranih polja. Na primer, ukoliko je kreiran sledeći indeks:



Indeks se može koristiti za upite nad sledećim poljima (prefiksima):

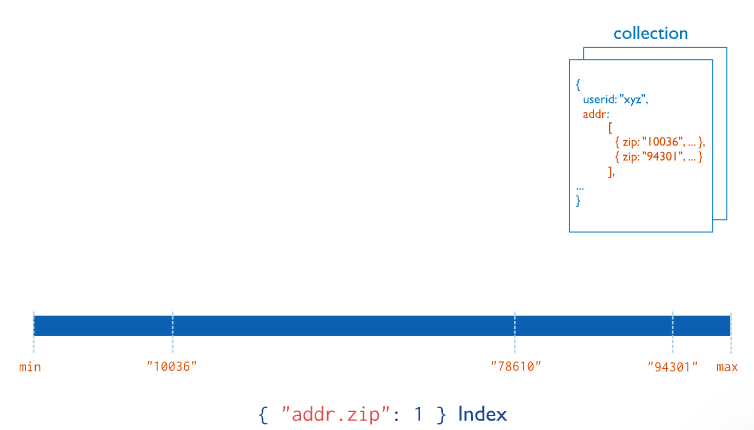
* age polje
* age i salary polja
* age, salary i position polja

Indeks, takođe, podržava operacije ***sortiranja*** gde se sortiranje vrši po poljima koja predstavljaju ***prefiks*** indeksa. Sortiranje je podržano i po podskupu polja koji nije prefiks indeksa, ali samo pod uslovom da upit sadrži uslove jednakosti za sva polja koja prethode poljima po kojima se sortira, kao u sledećem primeru:



1. **Multikey indeksi**

MongoDB koristi ***multikey indekse*** za indeksiranje ***nizova***. Ukoliko se indeksira polje čija je vrednost niz, MongoDB kreira poseban data entry za svaki element niza. Multikey indeksi se mogu kreirati nad nizovima koji sadrže skalarne vrednosti ili nad nizovima koji sadrže ugnježdene dokumente. Multikey indeksi podržavaju upite koji selektuju dokumente koji sadrže nizove poklapanjem jednog ili više elemenata niza. MongoDB automatski određuje da li se pravi multikey indeks na osnovu toga da li indeksirano polje sadrži niz, nije potrebno eksplicitno specificiranje tipa indeksa. Na slici 2.1.5 prikazan je izgled jednog multikey indeksa.

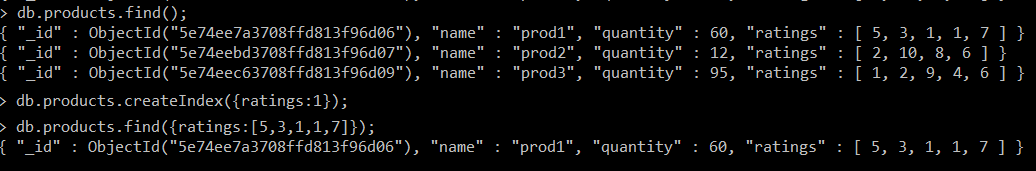


Slika 2.1.5 Multikey indeks

Ukoliko se kreira ***compound multikey indeks***, za svaki indeksirani dokument važi da ***najviše jedno indeksirano polje može da ima niz kao vrednost***.

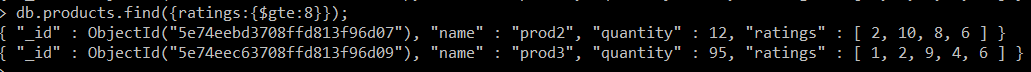
Ukoliko niz sadrži ugnježdene dokumente kao vrednosti, indeks se može kreirati i nad nekim od polja ugnježdenih dokumenata. U primeru ilustrovanom na slici 2.1.5 kreiran je indeks nad *zip* poljem dokumenta koji predstavlja element niza *addr*.

Kada filter u upitu zahteva podudaranje celog niza, MongoDB može da koristi multikey indeks da pronađe prvi element niza navedenog u upitu, ali ne može da ga koristi kako bi pronašao ceo niz. Umesto toga, nakon što pomoću indeksa izdvoji dokumente koji sadrže prvi element niza iz upita, MongoDB dalje filtrira samo dokumente koji sadrže traženi niz. U primeru na slici 2.1.6 dat je izgled dokumenata *products* kolekcije, zatim se kreira multikey indeks nad *ratings* poljem, a nakon toga se pomoću upita izdvajaju dokumenti sa zadatim nizom ocena. MongoDB koristi multikey indeks da pronađe sve dokumente koji sadrže 5 na bilo kojoj poziciji u nizu ocena. Nakon toga se izdvajaju dokumenti čiji se ratings niz poklapa sa nizom iz upita.



Slika 2.1.6 Kreiranje i korišćenje multikey indeksa

Ukoliko upit ne zahteva poklapanje celog niza, već su uslovi, veće, manje, itd., onda se korišćenjem indeksa izdvajaju dokumenti kod kojih barem jedan član niza zadovoljava zadati uslov. U primeru na slici 2.1.7 izdavjaju se dokumenti kod kojih barem jedan element niza *ratings* zadovoljava uslov da je veći ili jednak 8.



Slika 2.1.7 Korišćenje multikey indeksa bez poklapanja celog niza

1. **Geospatial indeksi**

Da bi podržao efikasno izvršenje upita nad podacima koji predstavljaju geoprostorne koordinate, MongoDB koristi dva specijalna indeksa: ***2d indekse*** koji koriste planarnu geometriju prilikom vraćanja rezultata i ***2dsphere indekse*** koji koriste sfernu geometriju prilikom vraćanja rezultata.

1. **Text indeksi**

Ovaj tip indeksa podržava pretragu tekstualnog sadržaja u kolekciji. ***Tekstualni indeksi*** mogu da sadrže bilo koje polje čija je vrednost ***string ili niz stringova***. Ovi indeksi ne čuvaju ***stop reči*** i vrši se ***tokenizacija*** i ***stemovanje*** svake izdvojene reči. Na nivou kolekcije može postojati najviše ***jedan*** text indeks i operacije sortiranja iz upita ne mogu da koriste redosled iz tekstualnog indeksa. Pretraga indeksiranih tekstualnih polja se vrši korišćenjem ***$text*** operatora.

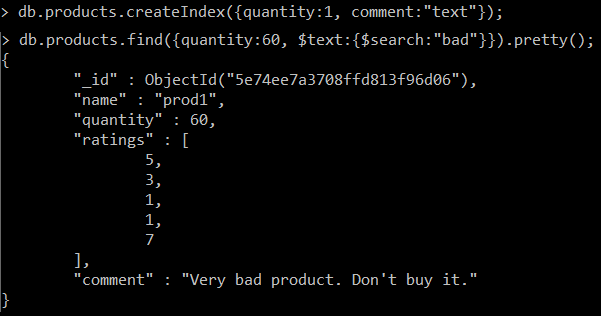
Prilikom kreiranja indeksa, nakon navođenja polja nad kojim se kreira indeks, umesto smera sortiranja se navodi ***tip indeksa***, odnosno ***text***. Može se kreirati text indeks nad više tekstualnih polja. Može se kreirati i ***compound text indeks*** nad tekstualnim poljem i drugim rastućim/opadajućim poljima. Takvi indeksi imaju neka ograničenja:

* Compound text indeks ne može da uključuje nijedan drugi tip indeksa, kao multikey ili geospatial indeksna polja.
* Ukoliko compound text indeks sadrži polja koja prethode tekstualnom polju, da bi se pretraživalo tekstualno polje, predikat u upitu mora da sadrži uslove jednakosti za sva polja koja prethode tekstualnom polju.
* Prilikom kreiranja compound text indeksa, sva tekstualna polja moraju da budu susedna.

Na slici 2.1.8 dat je primer kreiranja jednog tekstualnog indeksa nad *comment* poljem *products* kolekcije i pretraga po tekstualnom polju (izdvajaju se dokumenti koji u komentaru sadrže reč bad). Na slici 2.1.9 dat je primer jednog compound text indeksa nad poljima *quantity* i *comment* *products* kolekcije (prilikom kreiranja compound text indeksa, prethodno kreiran text indeks je morao biti izbrisan zbog ograničenja) i upita.



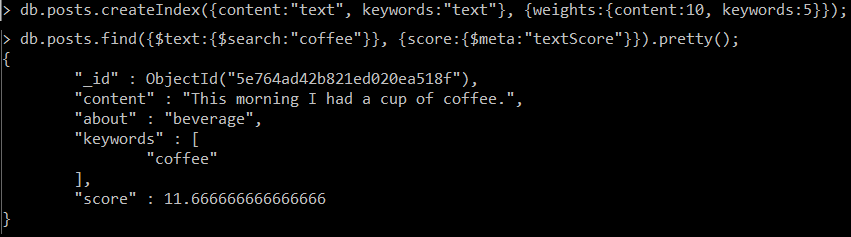
Slika 2.1.8 Text indeks i pretraga tekstualnog polja



Slika 2.1.9 Compound text indeks i upit

U tekstualnom indeksu, može se za svako indeksirano ***tekstualno*** polje navesti ***težina*** tog polja koja označava značaj tog polja u odnosu na druga indeksirana polja u pogledu rezultata pretraživanja teksta. Za svako indeksirano polje u dokumentu, MongoDB množi broj poklapanja sa težinom i sumira rezultate. Korišćenjem sume, MongoDB zatim računa ***skor*** dokumenta. Skor svakog vraćenog dokumenta se može videti korišćenjem ***$meta*** operatora uz koji se navodi ključna reč ***textScore***. Težina polja se specificira prilikom kreiranja indeksa korišćenjem ***weights*** opcije. Podrazumevana težina je 1.

Na slici 2.1.10 je primer gde su specificirane težine tekstualnih polja *content* i *keywords* *posts* kolekcije, izvršen je upit i prikazan *skor* izdvojenog dokumenta.



Slika 2.1.10 Text indeks sa težinama i pretraga

Indeks koristi pravila za stemovanje i listu stop reči specifične za jezik. Za svaku stemovanu reč text indeks čuva po jedan data entry. Podrazumevani jezik je engleski, ali se to može promeniti.

Za promenu podrazumevanog jezika na nivou kolekcije koristi se ***default\_language*** opcija prilikom kreiranja indeksa. Ukoliko kolekcija sadrži dokumente ili ugnježdene dokumente koji su na različitim jezicima, uključivanjem polja ***language*** u dokument ili ugnježdeni dokument sa vrednošću korišćenog jezika rešava se taj problem. Prilikom kreiranja indeksa koristi se jezik specificiran u okviru language polja. Na slici 2.1.11 je primer kreiranja text indeksa sa promenom podrazumevanog jezika, kao i primer korišćenja *language* opcije.



Slika 2.1.11 Promena podrazumevanog jezika i korišćenje *language* polja

1. **Hashed indeksi**

Kod ovog tipa indeksa čuva se heširana vrednost indeksiranog polja. Ovaj tip indeksa podržava samo traženje po jednakosti i ne podržava traženje po opsegu.

Hashed indeksi koriste ***heš funkciju*** da izračunaju heš vrednost indeksiranog polja. Ukoliko je vrednost ugnježdeni dokument, heš vrednost se računa za ceo dokument. Compound i multikey hashed indeksi nisu podržani.

Prilikom kreiranja indeksa, nakon navođenja polja nad kojim se kreira indeks, umesto smera sortiranja se navodi ***tip indeksa***, odnosno ***hashed***.

Ne može se staviti unique ograničenje za hashed indekse. Međutim, mogu se kreirati hashed indeks i rastući/opadajući indeks nad istim poljem. MongoDB će koristiti skalarni indeks za pretragu po opsegu.

## Svojstva i upravljanje indeksima

Neka od osnovnih svojstava indeksa MongoDB baze podataka su:

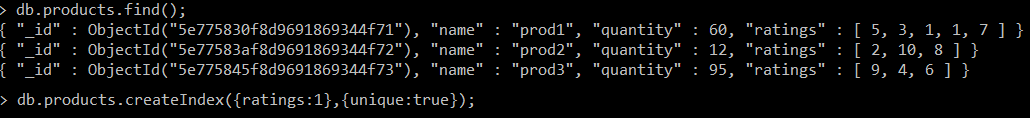
1. **Unique indeksi**

***Unique svojstvo*** indeksa uzrokuje da MongoDB odbija duplikate vrednosti indeksiranog polja. Podrazumevani indeks nad \_id poljem je unique. Da bi indeks bio unique, prilikom kreiranja je potrebno u *options* delu metoda dodati opciju ***unique:true***.

I compound indeksi mogu da budu unique. To znači da kombinacije vrednosti indeksiranih polja moraju da budu jedinstvene.

MongoDB ne može da kreira unique indeks nad specificiranim poljima ukoliko kolekcija već sadrži podatke koji ne zadovoljavaju unique ograničenje. Što se multikey indeksa tiče, pošto se unique ograničenje primenjuje na svaki dokument iz kolekcije, dokument može da ima niz elemenata koji se ponavljaju ukoliko se indeksirane vrednosti ne pojavljuju ni u jednom drugom dokuemntu u kolekciji.

Na slici 2.2.1 dat je primer kreiranja multikey unique indeksa nad *ratings* poljem *products* kolekcije. Ocene za različite proizvode se ne smeju ponavljati, dok se za jedan proizvod mogu ponoviti.



Slika 2.2.1 Unique multikey indeks

Ukoliko dokument nema vrednost za indeksirano polje u unique indeksu, ***null vrednost*** će biti sačuvana za taj dokument. Zbog unique ograničenja, MongoDB dozvoljava da samo jedan dokument nema vrednost za indeksirano polje.

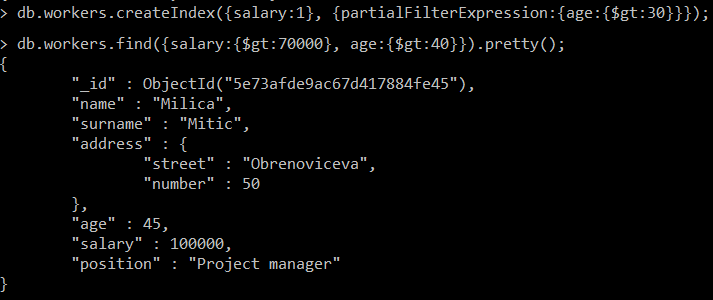
1. **Partial indeksi**

***Partial indeksi*** indeksiraju samo dokumente koji zadovoljavaju specificirani filter. Kreiranje partial indeksa se vrši dodavanjem ***partialFilterExpress*** opcije u metodi za kreiranje indeksa. Filter se specificira korišćenjem:

* Izraza jednakosti (field:value ili $eq operator)
* $exists:true izraza
* $gt, $gte, $lt, $lte izraza
* $type izraza.

***PartialFilterExpression*** opcija se može specificirati za bilo koju vrstu indeksa. Nad \_id poljem se ne može kreirati partial indeks i ne može se kreirati više indeksa koji se razlikuju samo po izrazu za filtriranje.

MongoDB neće koristiti partial indeks za upit ukoliko korišćenje indeksa vraća nekompletne rezultate. Da bi se koristio partial indeks, upit mora da sadrži izraz za filtriranje (ili modifikovani izraz za filtriranje koji predstavlja podskup izraza za filtriranje) kao deo upita. Na slici 2.2.2 je primer kreiranja partial indeksa nad *salary* poljem *workers* kolekcije, gde se indeksiraju samo dokumenti koji se odnose na radnike sa više od 30 godina. Zatim se upitom izdvajaju radnici sa platom većom od 70000 i stariji od 40 godina (drugi uslov izdvaja podksup indeksiranih dokumenata).



Slika 2.2.2 Kreiranje i korišćenje partial indeksa

Ukoliko se pored ***partialFilterExpression*** opcije specificira i ***unique*** ograničenje za indeks, ***unique*** ograničenje se odnosi samo na dokumente koji zadovoljavaju uslov filtriranja. Unique ograničenje ne sprečava dodavanje dokumenata koji ne zadovoljavaju ovo ograničenje ukoliko dokumenti ne zadovoljavaju uslov filtriranja.

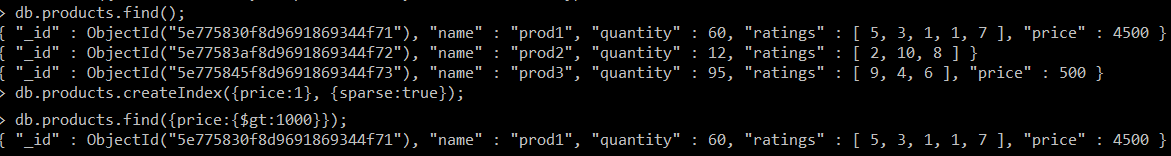
1. **Sparse indeksi**

***Sparse svojstvo*** indeksa osigurava da indeks sadrži samo podatke iz dokumenata koji sadrže indeksirano polje.

Za kreiranje sparse indeksa se u metodi za kreiranje postavlja ***sparse*** opcija na ***true***. Ukoliko korišćenje sparse indeksa za upite rezultuje nekompletnim rezultatima, MongoDB neće koristiti indeks ukoliko to nije eksplicitno navedeno.

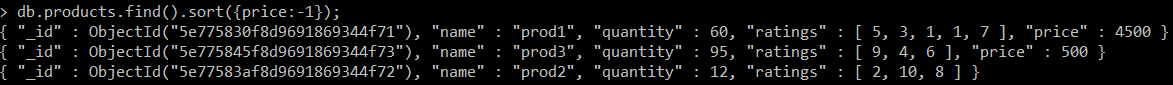
***Geospatial*** i ***text*** indeksi su uvek ***sparse***. ***Sparse comound*** indeksi koji sadrže samo rastuće/opadajuće ključeve će indeksirati dokument ukoliko sadrži barem jedan od navedenih ključeva (sadrži barem jedno od indeksiranih polja). Ukoliko se radi o ***geospatial*** ili ***text compound*** indeksima, dokument se indeksira samo ukoliko sadrži geospatial ili tekstualno polje.

***Sparse opcija*** se može kombinovati sa ***unique*** opcijom kako bi se sprečilo dodavanje dokumenata koji imaju duplikate vrednosti indeksiranih polja. Na slici 2.2.3 dat je primer kreiranja sparse indeksa nad *price* poljem *products* kolekcije i pretraga dokumenata čija je cena veća od 1000 (prilikom pretrage koristi se kreirani indeks jer se dobijaju potpuni rezultati).



Slika 2.2.3 Kreiranje i korišćenje sparse indeksa

Na slici 2.2.4 je primer upita za istu kolekciju gde korišćenje indeksa ne bi vratilo potpune rezultate jer je potrebno vratiti sve dokumente iz kolekcije.



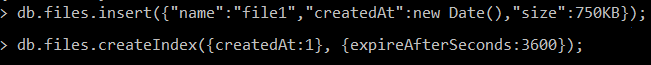
Slika 2.2.4 Primer upita koji ne koristi sparse indeks

1. **TTL indeksi**

***TTL (time to live) indeksi*** su posebna vrsta ***single field*** indeksa koju MongoDB koristi da automatski ukloni dokumente iz kolekcije posle određenog vremena. Ovo je korisno za podatke kao što su logovi, informacije o sesijama, koji treba da ostanu u bazi određeni vremenski period.

Da bi se kreirao TTL indeks, u metodi za kreiranje se dodaje opcija ***expireAfterSeconds***. Indeks se kreira nad poljem čija je vrednost ***datum ili niz datuma*** (ukoliko je polje nekog drugog tipa, dokument neće isteći). Dokumenti ističu nakon što prođe specificiran broj sekundi od vrednosti indeksiranog polja. Ukoliko je vrednost indeksiranog polja niz, MongoDB koristi najmanju vrednost niza da izračuna vreme isticanja. Pozadinska nit briše dokumente koji su istekli. Pozadinska nit koja uklanja dokumente se pokreće na svakih 60 sekundi.

Nad \_id poljem se ne može kreirati TTL indeks. Ukoliko nad nekim poljem već postoji single field indeks koji nije TTL, ne može se kreirati TTL indeks nad istim poljem ukoliko se prethodno ne obriše postojeći indeks. Na slici 2.2.5 dat je primer kreiranja TTL indeksa nad *createdAt* poljem *files* kolekcije. Dokument se briše 60 minuta od vremena kreiranja.



Slika 2.2.5 Kreiranje TTL indeksa

Funkcije za kreiranje i pregled svih indeksa kolekcije su navedene na početku dela **2. indeksi mongodb baze podataka**.

Metod za brisanje određenog indeksa iz kolekcije je ***dropIndex***, čiji je parametar naziv ili specifikacija indeksa (parovi field:value nad kojima je kreiran indeks, ne može se koristiti kod text indeksa). Za brisanje svih indeksa kolekcije koristi se metod ***dropIndexes***. Na slici 2.2.6 prikazan je način brisanja indeksa kreiranog nad *age*, *salary* i *position* poljima *workers* kolekcije i sa nazivom i sa specifikacijom kao parametrom odgovarajuće funkcije.

MongoDB ne podržava modifikaciju indeksa, može se samo obrisati, pa ponovo kreirati indeks.



Slika 2.2.6 Brisanje indeksa

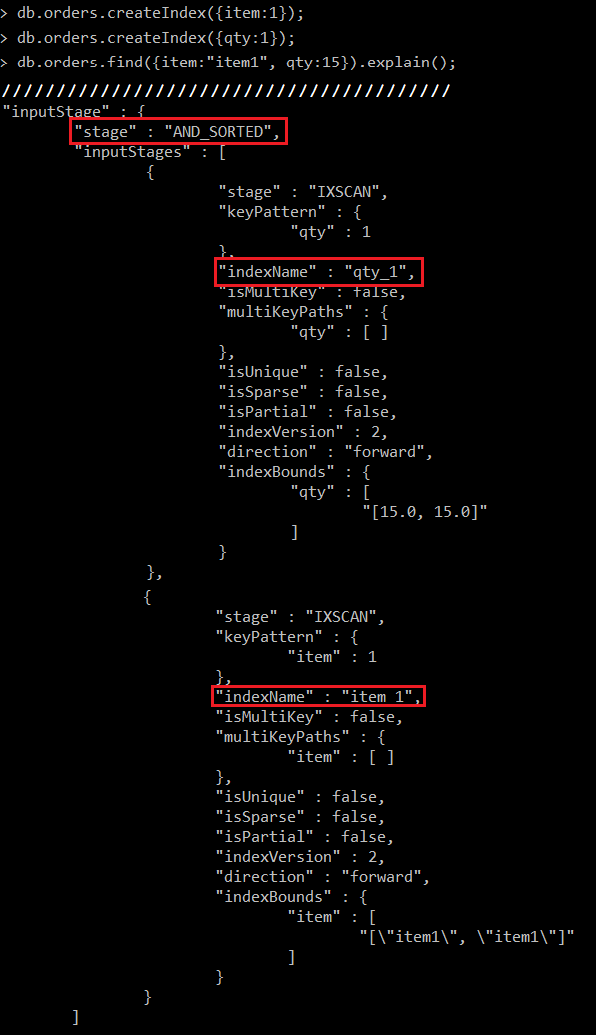
## Index intersection

MongoDB može da koristi presek više indeksa kako bi izvršio upit. Generalno, svaki presek indeksa uključuje dva indeksa, međutim, MongoDB može da iskoristi više/ugnježdene preseke indeksa da bi izvršio upit. Na slici 2.3.1 je dat primer kreiranja dva indeksa, nad poljem *item* i nad poljem *qty* kolekcije *orders*, kao i upit koji može da iskoristi presek ova dva indeksa. MongoDB koristi presek indeksa uglavnom kod traženja po jednakosti, retko kod traženja po opsegu. U većini slučajeva najefikasniji plan izvršenja upita ne sadrži presek indeksa i jako je teško pronaći primer gde MongoDB bira plan izvršenja upita koji sadrži presek indeksa. ***Explain*** funkcija daje objašnjenje izvršenja upita i tu se može videti koji plan je iskorišćen, a koji planovi su razmatrani, pa odbačeni. U primeru sa slike 2.3.1 među odbačenim planovima se nalazi i onaj koji koristi presek indeksa. Ukoliko postoji plan koji koristi presek indeksa, u rezultatima explain funkcije postoji faza (stage) označena sa ***AND\_SORTED*** ili ***AND\_HASH***. Detaljnije opis rezultata explain funkcije je dat u delu **2.4 Praćenje korišćenja indeksa**.

MongoDB može da koristi presek celog indeksa ili njegovog prefiksa (početni podskupovi indeksiranih polja). Presek indeksa se ne koristi kada sortiranje zahteva potpuno drugi indeks od onog koji se koristi za selekciju dokumenata. Ukoliko su kreirani indeksi kao u primeru sa slike 2.3.1, presek indeksa se ne može koristiti u realizaciji sledećeg upita sa sortiranjem:

*db.orders.find({item:“item1“}).sort({qty:1}).*

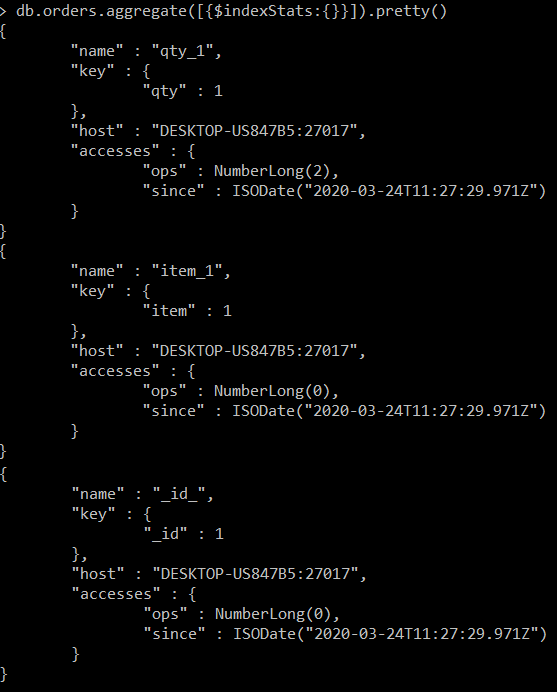
Predikat upita zahteva korišćenje indeksa nad poljem *item*, dok operacija sortiranja zahteva korišćenje indeksa nad poljem *qty*.



Slika 2.3.1 Primer korišćenje preseka indeksa

## Praćenje korišćenja indeksa

Da bi se dobila statistika koja se odnosi na upotrebu indeksa neke kolekcije, koristi se ***$indexStats***. Za svaki indeks se može videti njegov naziv (***name*** polje), specifikacija (polja nad kojima je kreiran; ***key*** polje), informacije o hostu (***host*** polje) i statistika korišćenja indeksa (***accesses*** polje), odnosno broj operacija koje su koristile indeks (***ops*** polje) i vreme od kada je MongoDB prikupljao podatke (***since*** polje). Na sledećem primeru na slici 2.4.1 prikazana je statistika korišćenja indeksa *orders* kolekcije.



Slika 2.4.1 Statistika korišćenja indeksa *orders* kolekcije

Za pregled statistike o upitu, uključujući korišćene indekse, broj skeniranih dokumenata, kao i vreme izvršavanja upita u ms, koristi se ***explain*** metoda. Informacije koje se dobijaju pozivom ove funkcije su:

* U okviru ***queryPlanner*** dela mogu se videti informacije o planu izvršenja koji je izabran od strane optimizatora upita. Te informacije obuhvataju ***namespace*** nad kojim se izvršava upit, da li je MongoDB koristio ***index filter***, ***parsirani upit***, plan izvršenja upita koji je query optimizer izabrao (***winningPlan***), planove koji je query optimizer razmatrao i odbacio (***rejectedPlans***). Za planove izvršenja se mogu videti informacije koje obuhvataju naziv faze izvršenja (***stage***), ***filter*** primenjen u toj fazi, podfaze u okviru faze (***inputStages***). Svaka faza se sastoji od informacija specifičnih za tu fazu. Na primer ***IXSCAN*** faza se odnosi na pretragu indeksa i sadrži informacije o korišćenom indeksu (naziv, specifikacija, granice, itd.). Na slici 2.4.2 prikazan je izgled ovog dela rezultata explain funkcije za upit.



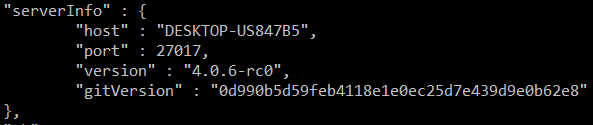
Slika 2.4.2 queryPlanner deo

* U okviru ***executionStats*** dela mogu se videti informacije o izvršenju upita po izabranom planu. Ove informacije obuhvataju broj dokumenata koji odgovaraju upitu (***nReturned***), vreme izvršenja upita (***executionTimeMillis***), broj skeniranih index etries (***totalKeysExamined***), broj ispitanih dokumenata (***totalDocsExamined***), detalji izvršenja izabranog plana (***executionStages***), delimične informacije o odbačenim planovima, prikupljene tokom faze selekcije plana (***allPlansExecution***). Na slici 2.4.3 je prikazan izgled ovog dela.



Slika 2.4.3 executionStats deo

* U delu ***serverInfo*** mogu se videti informacije o pokrenutoj MongoDB instanci. Na slici 2.4.4 prikazan je izgled ovog dela.



Slika 2.4.4 serverInfo deo

Kako bi se MongoDB primorao da koristi određeni indeks za ***find*** operaciju, koristi se ***hint*** metoda kojoj se prosleđuje indeks koji treba koristiti. Hint funkciji se prosleđuje naziv ili specifikacija indeksa koji treba koristiti. Prosleđivanjem ***{$natural:1}*** hint funkciji sprečava se da MongoDB koristi bilo koji indeks, forsira se skeniranje kolekcije. Na slici 2.4.5 prikazan je upit iz primera za explain funkciju, gde se bira plan izvršenja koji koristi indeks nad poljem *qty* *orders* kolekcije, sa korišćenjem hint metoda kome se prosleđuje indeks nad poljem *item*.



Slika 2.4.5 Korišćenje hint metode

## Strategije indeksiranja

Najbolji indeksi za aplikaciju moraju da uzmu u obzir brojne faktore, uključujući tipove upita koji se očekuju, odnos čitanja i upisa, kao i količinu slobodne memorije kojom sistem raspolaže. Prilikom razvijanja strategije indeksiranja mora da postoji razumevanje upita koje aplikacija zahetva. Generalno, MongoDB koristi jedan indeks za realizaciju većine upita.

Strategije indeksiranja koje je preporučljivo koristiti su:

1. ***Kreiranje indeksa koji podržavaju upite***

Indeks podržava upit kada indeks sadrži sva polja koja se skeniraju u upitu. Kreiranjem indeksa koji podržavaju upite, performanse upita se dosta poboljšavaju.

1. ***Korišćenje indeksa za sortiranje rezultata***

Kod MongoDB baze, operacije sortiranja mogu da dobiju sortirane dokumente korišćenjem indeksa i njegovog sortiranja. Ukoliko se indeks ne može iskoristiti za sortiranje, rezultati će biti sortirani u memoriji. Operacije sortiranja koje koriste indeks uglavnom imaju bolje performanse od onih koje ga ne koriste. Operacije sortiranja koje ne koriste indeks će biti prekinute kada koriste 32MB memorije.

1. ***Osiguravanje da se indeksi uklapaju u količinu raspoloživog RAM-a***

Kada se indeks uklapa u količinu raspoloživog RAM-a, sistem može da izbegne čitanje indeksa sa diska i na taj način se dobija brža obrada.

1. ***Kreiranje upita koji osiguravaju selektivnost***

Selektivnost je mogućnost upita da suzi rezultate korišćenjem indeksa. Efektivni indeksi omogućavaju korišćenje indeksa za veći deo posla prilikom izvršavanja upita. Da bi se osigurala selektivnost potrebno je pisati upite koji ograničavaju broj mogućih dokumenata pomoću indeksiranog polja. Cilj je skenirati što manje dokumenata korišćenjem indeksa prilikom izvršavanja upita.

# zaključak

Postoje brojne prednosti korišćenje indeksa. Osnovni razlog je poboljšanje performansi upita koji koriste indekse u odnosu na one koji vrše skeniranje kolekcije.

MongoDB podržava različite vrste indeksa. Svaka vrsta se koristi za određene tipove polja koji se indeksiraju i podržava upite određenog tipa. Zbog toga je MongoDB pogodan za korišćenje u različitim sistemima i aplikacijama. Uz to, postoje i opcije za praćenje korišćenja indeksa koje nude pregled različitih statističkih podataka koji obuhvataju, kako informacije o samim indeksima, tako i informacije o upitima koji se izvršavaju. Mogućnost pregleda ovih statističkih podataka omogućava praćenje i dodatnu optimizaciju korišćenja MongoDB baze podataka.

# literatura

1. Prezentacije sa sajta predmeta
2. Database Management Systems, R. Ramakrishnan and J. Gehrke, McGraw-Hill; 3rd edition, 2002
3. Fundamentals of Database Systems, Addison Wesley, 6th edition, 2010
4. Dokumentacija MongoDB baze podataka

<https://docs.mongodb.com/manual/indexes/>

1. <http://source.wiredtiger.com/mongodb-3.4/tune_page_size_and_comp.html>
2. <https://medium.com/swlh/introduction-to-mongodb-indexes-cdb216f54f80>