



Distance Learning System

MySql

Indeksi, pogledi, uskladištene rutine i
transakcije

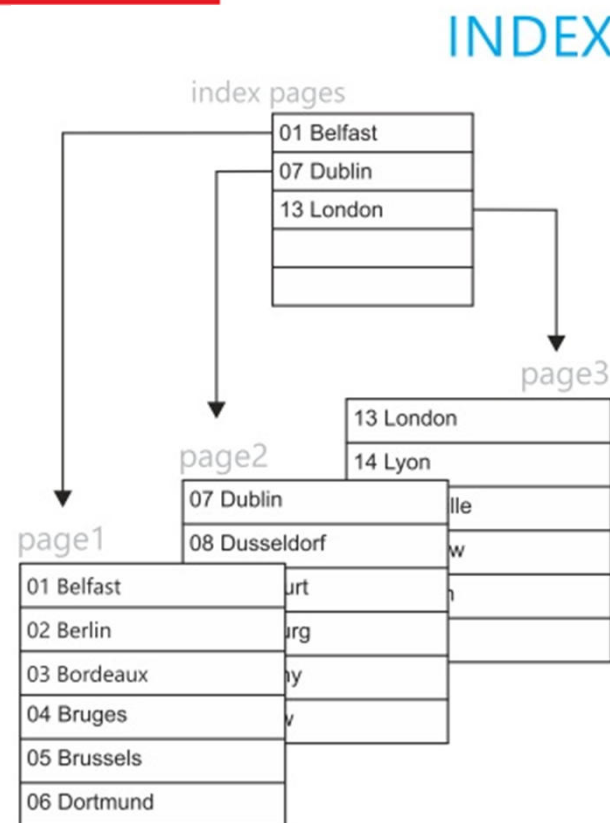
Indeksi

- U rukovanju bazama podataka, ključna stvar je brzo i precizno dobavljanje podataka. Kada su količine podataka u tabelama male, sistem će, u svakom slučaju, funkcionisati brzo. Ali, kada baza počne da sadrži nešto veće količine podataka, brzina postaje jedan od osnovnih problema. Međutim, nije brzina dostavljanja podataka ta koja proces čini sporim. Reč je o samom pronalaženju podataka. Kada se podatak jednom locira, brzina njegovog dostavljanja je irelevantna

1	Paris
2	Bruges
3	London
4	Moscow
5	Krakow

Mehanizam indeksa

- Većina podataka u MySQL bazi pamti na stranama. Veličina jedne strane je ograničena memorijski (obično su 8k ili 16k), ali ne može se tačno znati koliko u tu stranu može stati redova jer to zavisi od raznih faktora. Pre svega, zavisi od količine podataka u redovima.
- Ovako izgledaju strane za tabelu sa gradovima



Klasterovani i neklasterovani indeks

- Pretraga indeksa uvek funkcioniše na isti način, ali se njena finalizacija može izvršiti na dva načina. Ova dva načina čine da razlikujemo i dve vrste indeksa:
 - **klasterovani**
 - **neklasterovani (sekundarni)**
- Često se, sa punim pravom, ova dva indeksa porede sa knjigom i telefonskim imenikom. Ukoliko tražimo neku informaciju iz knjige, prvo ćemo pretražiti sadržaj i u njemu naći stranu za ono što nas zanima. Zatim ćemo prelistati knjigu i brzo doći do podatka, jer znamo stranu. Ako tražimo neki podatak u telefonskom imeniku, naći ćemo slovo koje nas zanima, a zatim direktno pristupiti strani na kojoj se nalaze svi podaci koji počinju tim slovom.
- Klasterovani indeks, na svom najnižem nivou (Leaf level) sadrži zapravo same podatke, dok neklasterovani indeks sadrži reference na prave podatke.
- Zato su neklasterovani indeksi nešto sporiji od klasterovanog

Formiranje Klasterovanog i neklasterovanog indeksa

- Svaki put kada kreiramo primarni ključ, automatski se kreira i klasterovani indeks. Ovaj indeks može biti samo jedan na nivou tabele
- Neklasterovane indekse kreiramo eksplicitno, i može ih biti više po jednoj tabeli

Strategija indeksiranja (klasterovani indeks)

- Kada napravimo jednu tabelu i dodelimo joj primarni ključ, MySQL server automatski, po toj koloni, kreira klasterovani indeks, pa je samo bitno da dobro isplaniramo šta će biti primarni ključ naše tabele, dok na sam klasterovani indeks i nećemo imati preveliki uticaj

```
create table mytable  
(  
  id int primary key auto_increment,  
  name varchar(256)  
);
```

```
show index from mytable
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
mytable	0	PRIMARY	1	id	A	0	NULL	NULL		BTREE

Kreiranje neklasterovanog indeksa

- Neklasterovani indeks je malo pristupačniji i njega možemo postaviti gde god hoćemo.
- To ne znači da je pametno staviti ga na sve kolone tabele
- Indeks je, još jedna velika količina podataka koja će se naći na našem serveru. Takođe, to je i vlika količina podataka koje naš sistem treba da obrađuje.
- Indeksi su u stanju da znatno ubrzaju pretragu, ali, njima se takođe pristupa i prilikom drugih intervencija na tabelama.
- Što više indeksiranih kolona u tabeli, utoliko više usporenja uzrokovanih obradom indeksa prilikom rukovanja podacima te tabele.

```
ALTER TABLE mytable ADD INDEX custom_nonclustered_index (name ASC)
```

Full text index

- Standardni indeks sa kojim smo se upoznali u dosadašnjem toku lekcije, kod tekstualnih kolona, sposoban je da prilikom pretrage uzme u razmatranje samo početne karaktere unosa. U situacijama kada u nekoj tekstualnoj koloni imamo zabeležen tekst koji se sastoji od više reči, običan indeks je praktično beskoristan.
- U takvim situacijama na scenu stupa Full-text Index.
- Full-text Index omogućava brzu pretragu kroz velike količine teksta. Ovo je dobar, funkcionalan, jednostavan i brz sistem i, što je najvažnije, nezamenljiv u nekim situacijama.
- Korišćenjem ovog indeksa, prilikom indeksiranja, indekserski parsira tekst i preuzima iz njega reči. Svaka reč smešta se u indeks pod određenim brojem. Takođe, polje sa koga je tekst preuzet pamti se pod određenim brojem. Na kraju, indekserski pravi relaciju između polja sa tekстом i rečima koje tom polju pripadaju.
- Kada dođe do pretrage, Engine prvo pronađe tražene reči, zatim izlista njihove relacije (u relacijama se pamti i broj pojavljivanja reči u polju), a zatim na osnovu tih relacija sortiranih po broju pojavljivanja reči, dobijamo i sama polja (odnosno, brojeve tih polja)

Kreiranje full text indeksa

- Ponekad ćemo pokušati da izvršimo pretragu na sledeći način:

```
select * from film where description like '%drama%'
```

- Kada kolone imaju mnogo teksta i ima mnogo redova u tabeli, ovo nije dobar koncept. Umesto njega, treba kreirati full text index:

Index Name	Type
PRIMARY	PRIMARY
idx_title	INDEX
idx_fk_language_id	INDEX
idx_fk_original_language_id	INDEX
idx_description	INDEX



Column	#	Order	Length
<input type="checkbox"/> film_id		ASC	
<input type="checkbox"/> title		ASC	
<input checked="" type="checkbox"/> description	1	ASC	
<input type="checkbox"/> release_year		ASC	
<input type="checkbox"/> language_id		ASC	
<input type="checkbox"/> original_language...		ASC	
<input type="checkbox"/> rental_duration		ASC	
<input type="checkbox"/> rental_rate		ASC	
<input type="checkbox"/> length		ASC	
<input type="checkbox"/> replacement_cost		ASC	
<input type="checkbox"/> rating		ASC	
<input type="checkbox"/> special_features		ASC	
<input type="checkbox"/> last_update		ASC	

Korišćenje full text indeksa

- Kada je full text indeks postavljen, postaju raspoložive i dodatne komande za pretragu:

```
SELECT * FROM film  
WHERE MATCH(description) AGAINST('epic drama')
```

Filtracija rezultata (Search Expressions)

- Pored ovakvog standardnog načina definisanja upita za pretragu kolona korišćenjem Full Text Indexa, postoje i načini na koje možemo da utičemo na samu pretragu

```
SELECT description FROM film  
WHERE MATCH(description) AGAINST('+epic +drama' IN BOOLEAN MODE);
```

```
SELECT description FROM film  
WHERE MATCH(description) AGAINST('+epic -drama' IN BOOLEAN MODE);
```

Simbol

+word

-word

~word

<word

>word

word*

"word1 word2"

()

Značenje

reč se mora pojaviti u rezultatu

reč se ne sme pojaviti u rezultatu

reč se može prijaviti u rezultatu, ali joj je data manja važnost

reči se daje manje značenje prilikom pretrage

reči se daje veće značenje prilikom pretrage

sve reči koje počinju sa word (npr. word, words, wordless)

traži se fraza identična onoj navedenoj među navodnicima

zgrade se koriste za grupisanje izraza, npr. '+drama +(epic saga) ',
tj. traže se svi izrazi koji sadrže fraze drama ili epic saga, ili i jedno i drugo