

SISTEMI ZA UPRAVLJANJE BAZAMA PODATAKA

INTERNA STRUKTURA I ORGANIZACIJA INDEKSA KOD MYSQL BAZE PODATAKA

Svetlana Mančić 1423

STRATEGIJE ORGANIZACIJE PODATAKA NA DISKU

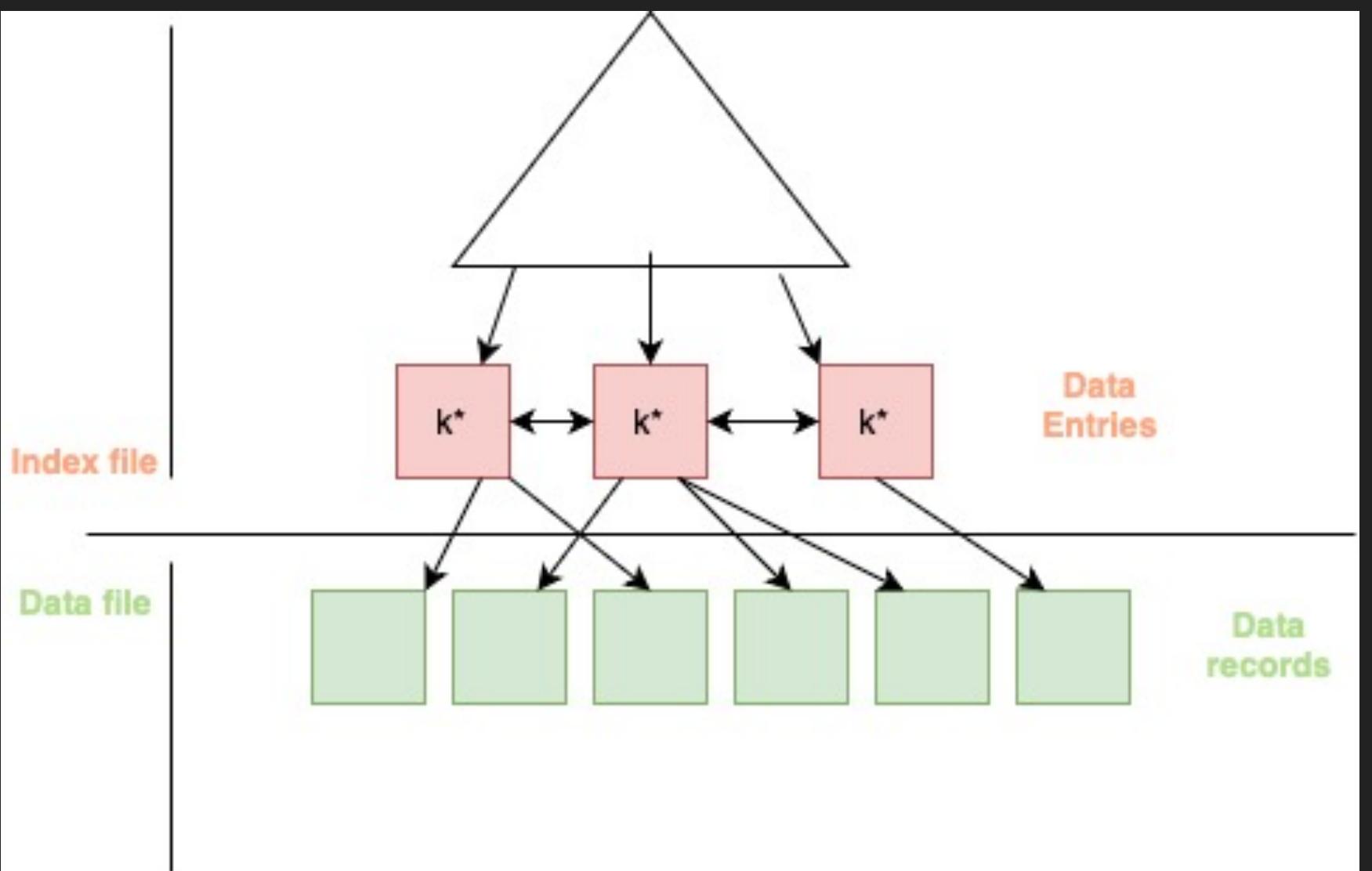
- ▶ DBMS podatke skladišti na:
 - ▶ Diskovima - eksterni uređaju, pristup nasumičnoj stranici po fiksnoj ceni
 - ▶ Trakama - podaci se sekvencijalno čitaju, pa zbog toga nisu pogodne za real-time aplikacije, pogodne su za arhiviranje podataka
 - ▶ Jedinica informacija koja se čita sa spoljnih uređaja i upisuje na spoljne uređaje naziva se ***stranica***.
 - ▶ Podaci su organizovani u kolekcije slogova, datoteke, gde svaka datoteka ima jednu ili više stranica.
- ▶ Strategije organizacije datoteka:
 - ▶ Neuređeni fajlovi
 - ▶ Uređeni fajlovi
 - ▶ Indeksi

INTERNA STRUKTURA I ORGANIZACIJA INDEKSA

- ▶ Indeksiranje je tehnika, koja omogućava brz pristup podacima na osnovu ključeva traženja, bez potrebe za pretraživanjem cele datoteke. Organizuju podatke u strukturu stabla, gde svaki čvor sadrži vrednost ključa traženja k.
- ▶ Za zapise čuvane u indeks fajlu može se koristiti termin **data entry**. Data entry sa vrednošću k za ključ traženja, u oznaci k^* , sadrži dovoljno informacija za lociranje jednog ili više slogova, koji za ključ traženja imaju vrednost k. Ovo znači da svaki data entry ima pokazivač na jedan ili više redova u tabeli. Prilikom pretrage u indeksu, prvo se pronalazi odgovarajući data entry, a zatim se na osnovu pronađenih informacija pristupa podacima u tabeli.

- ▶ Alternative za čuvanje data entry:

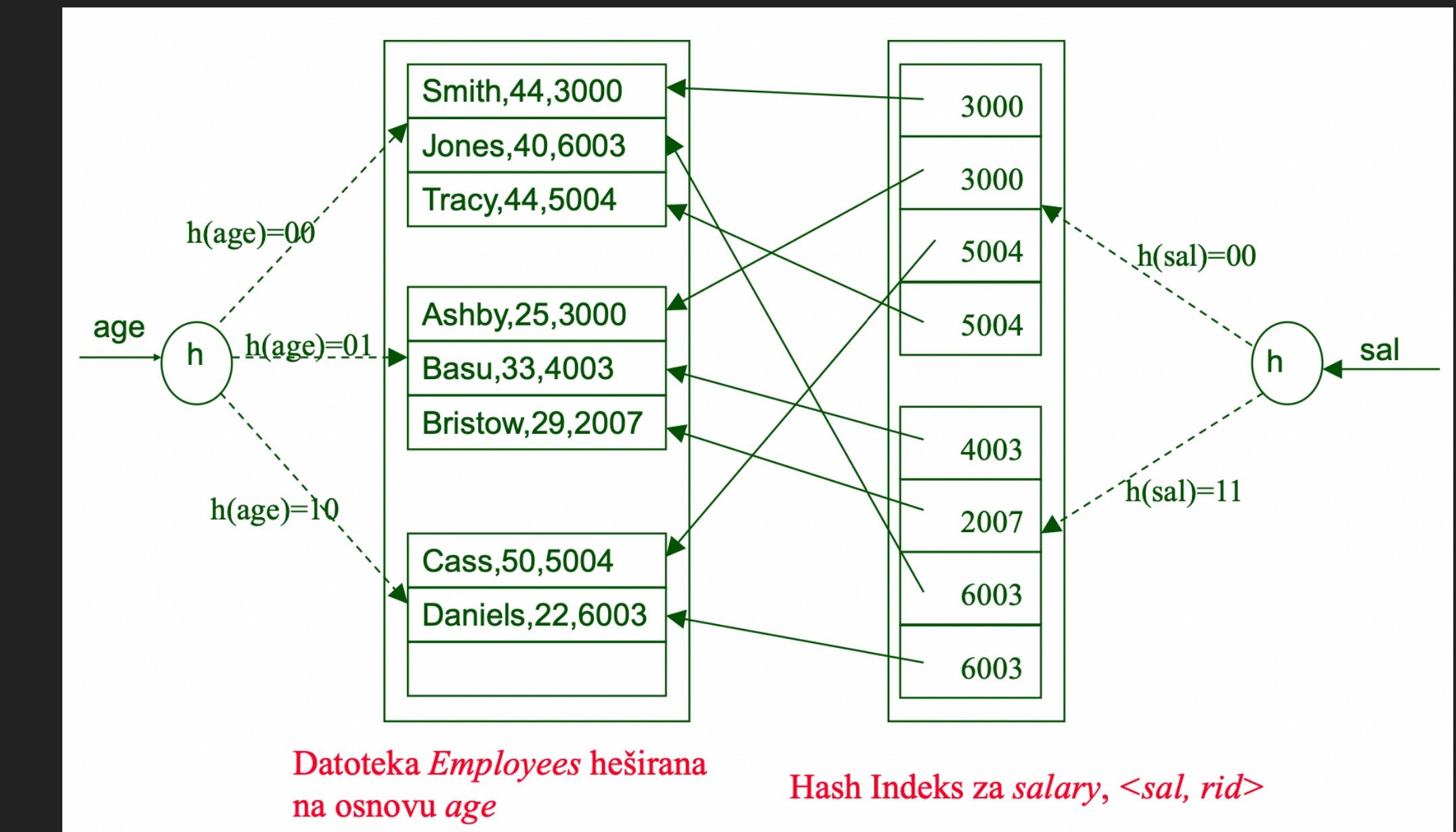
- ▶ Data entry k^* je zapis sa ključem traženja k
- ▶ Data entry je (k, rid) par, gde je rid record ID sloga sa ključem traženja k
- ▶ Data entry je $(k, rid-list)$ par



Slika 1. Tipična arhitektura indeksa

INTERNA STRUKTURA I ORGANIZACIJA INDEKSA

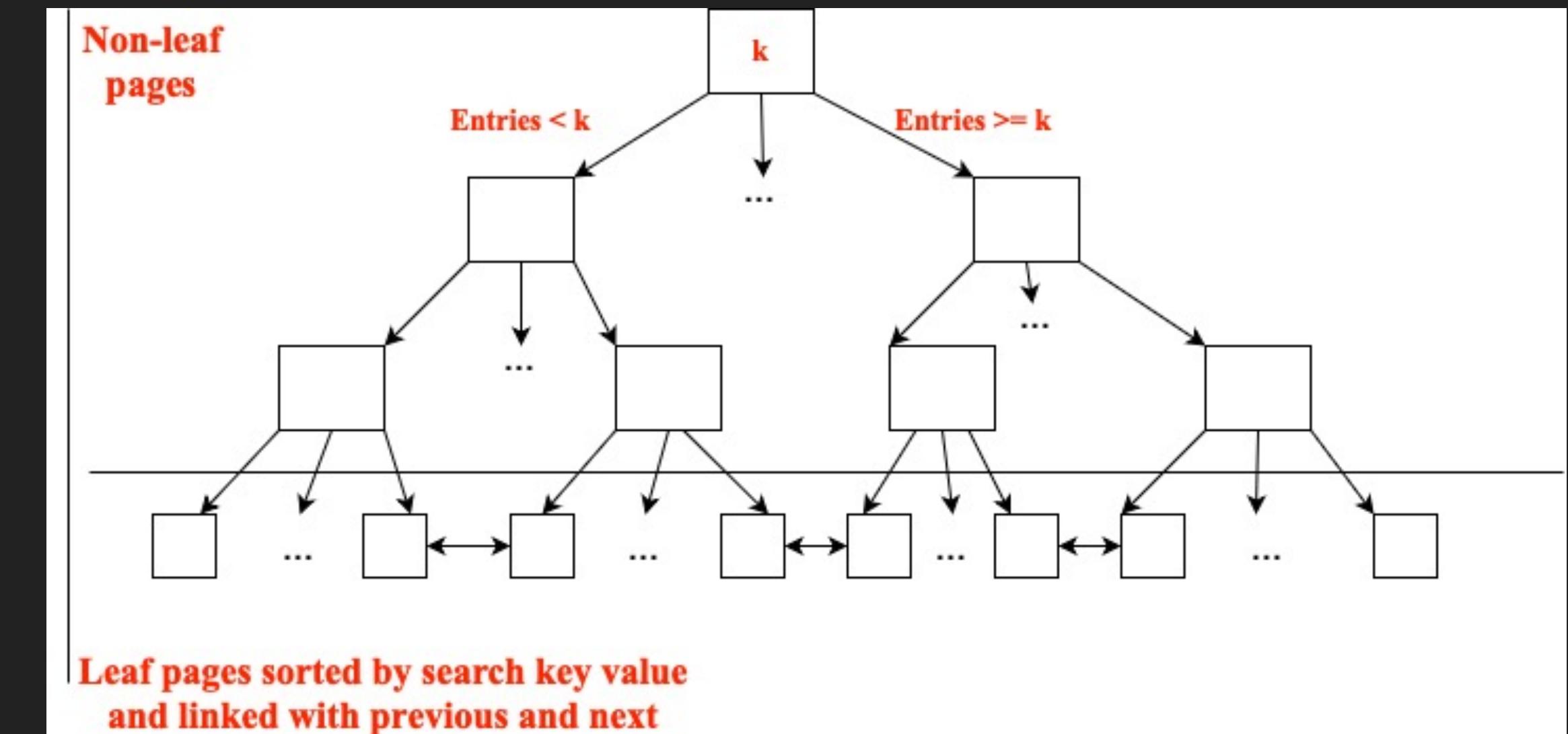
- ▶ Hash-based
 - ▶ Slogovi se grupišu u bucket-e
 - ▶ Za pronalaženje bucket-a kom pripada slog, primenjuje se heš funkcija, koja se primenjuje na vrednost ključa traženja
 - ▶ Optimizovani za selekcije jednakosti, loše performanse kod selekcije opsega
 - ▶ Efikasne operacije umetanja, brisanja i ažuriranja



Slika 2. Struktura heš indeksa

INTERNA STRUKTURA I ORGANIZACIJA INDEKSA

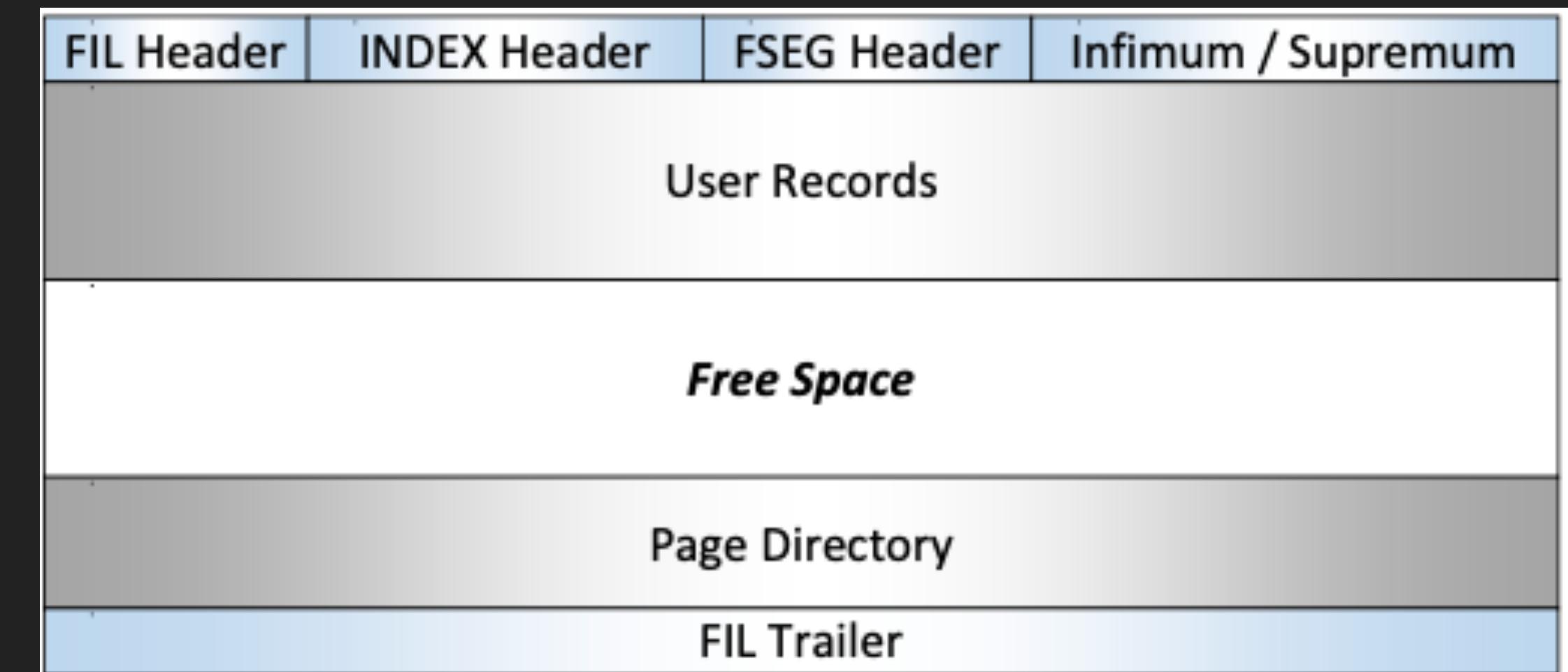
- ▶ Tree-based
 - ▶ Data entries su organizovani u strukturu stabla, sortirani po vrednosti ključa traženja
 - ▶ Pogodni za selekciju po jednakosti i za selekciju po opsegu
- ▶ R-tree indeksi za prostorne podatke



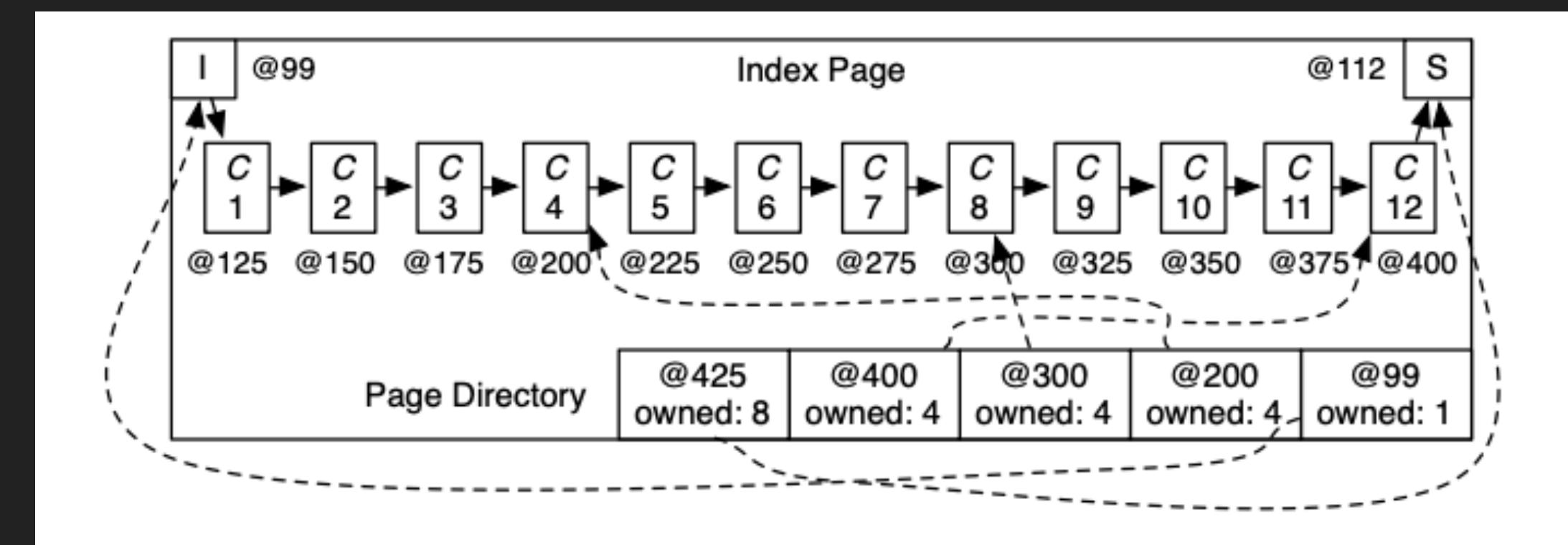
Slika 3. Indeksna struktura stabla

ORGANIZACIJA PODATAKA KOD MYSQL

- ▶ Podrazumevano skladište od verzije 8.0 je InnoDB.
 - ▶ 2 prostora tabela: ibdataX (sistemske tabele) i *.ibd (datoteke tabela)
- ▶ InnoDB nema posebnu strukturu skladišta, podaci se skladište u indeksu.
- ▶ Veličina stranice je 16KB, stranice su grupisane u extent-e veličine 1MB (64 uzastopnih stranica)
- ▶ Infimum i supremum polja u header-u su pokazivači na prvi i poslednji slog indeksa
- ▶ User Records sadrži slogove stranice
 - ▶ Slogovi su ulančani u vidu lančane liste, od infimuma do supremuma
 - ▶ Svaki slog ima pokazivač na sledeći u rastućem redosledu
- ▶ Page Directory sadrži ključeve slogova u rastućem redosledu i pokazivače na slogove
 - ▶ Pošto je sekvencijalno traženje skupo, poseduje pokazivače na svaki 4-8 slog što omogućava binarno traženje



Slika 4. Struktura stranice indeksa



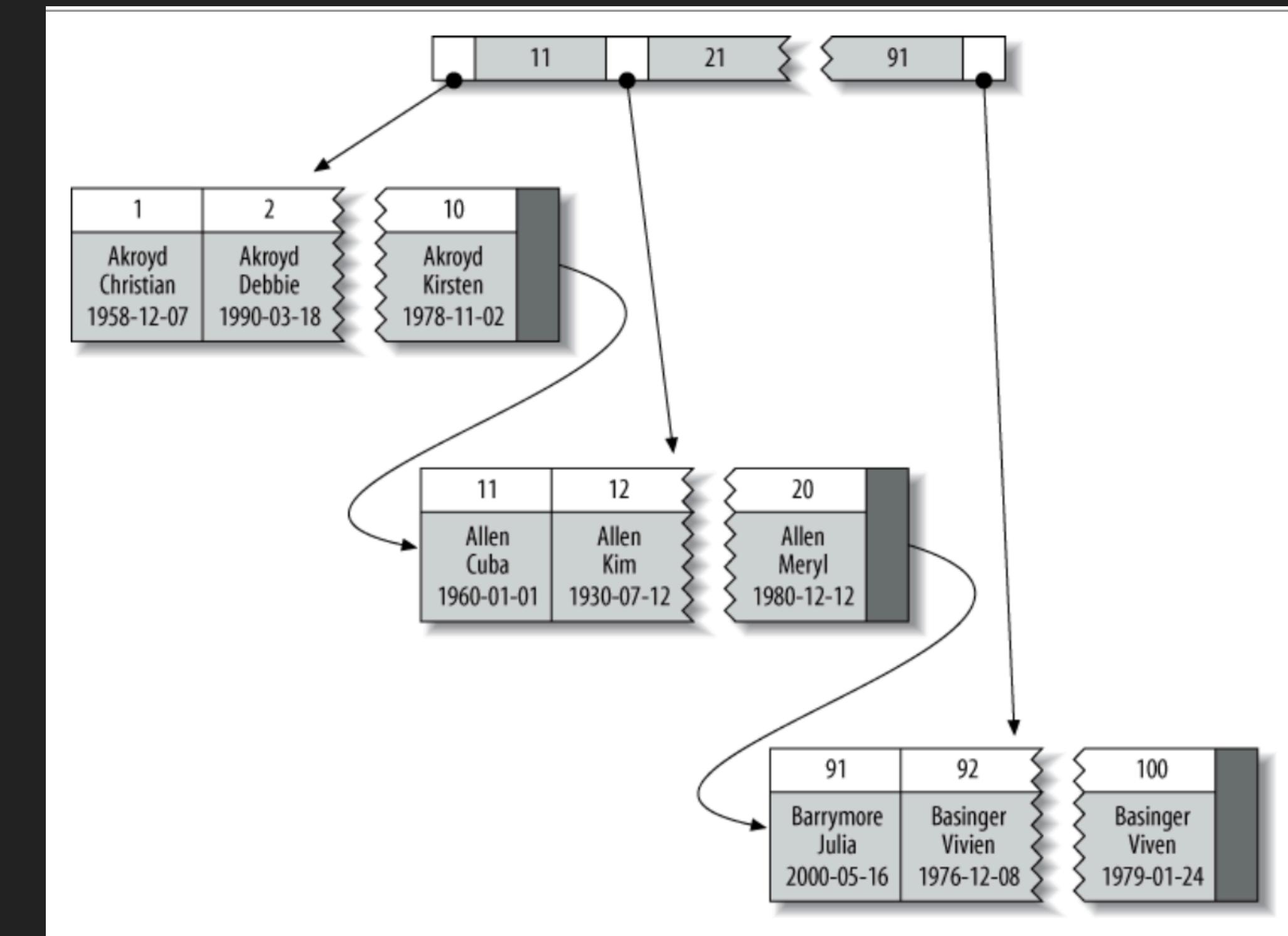
Slika 5. Logička povezanost slogova na stranici

KLASIFIKACIJA INDEKSA KOD INNODB SKLADIŠTA

- ▶ Postoji dva osnovna tipa: klasterizovani i neklasterizovani.
- ▶ Osim njih, postoje i fulltext indeksi, prostorni indeksi i adaptive hash indeks.

KLASTERIZOVANI INDEKS

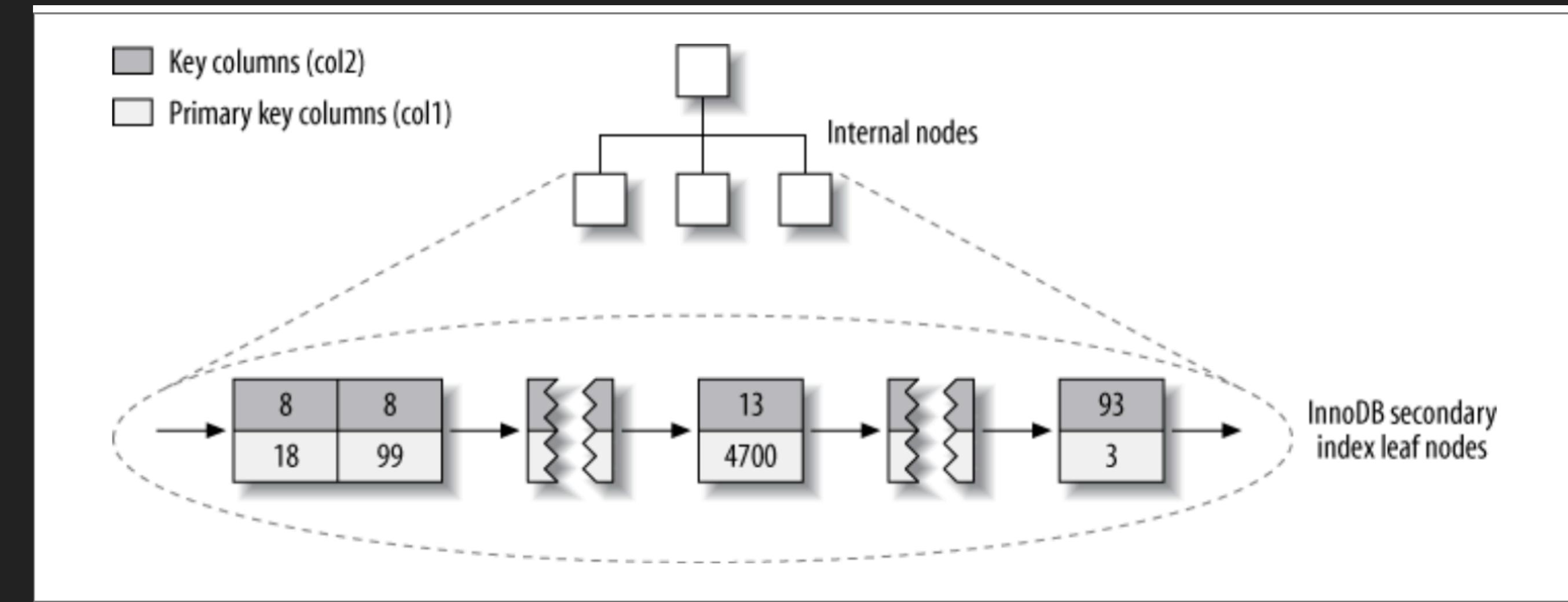
- ▶ Kod InnoDB skladišta, redovi tabele skladiše se u stranicama listova klasterizovanog indeksa.
- ▶ Kreira se automatski, indeksiranjem podataka na osnovu primarnog ključa.
- ▶ Ukoliko nije definisan primarni ključ, koristi se unique nonnullable indeks ako postoji.
- ▶ Ako nema ni unique indeksa, indeksiranje se vrši na koloni, koja ima vrednost row ID.



Slika 6. Struktura klasterizovanog indeksa kod InnoDB skladišta

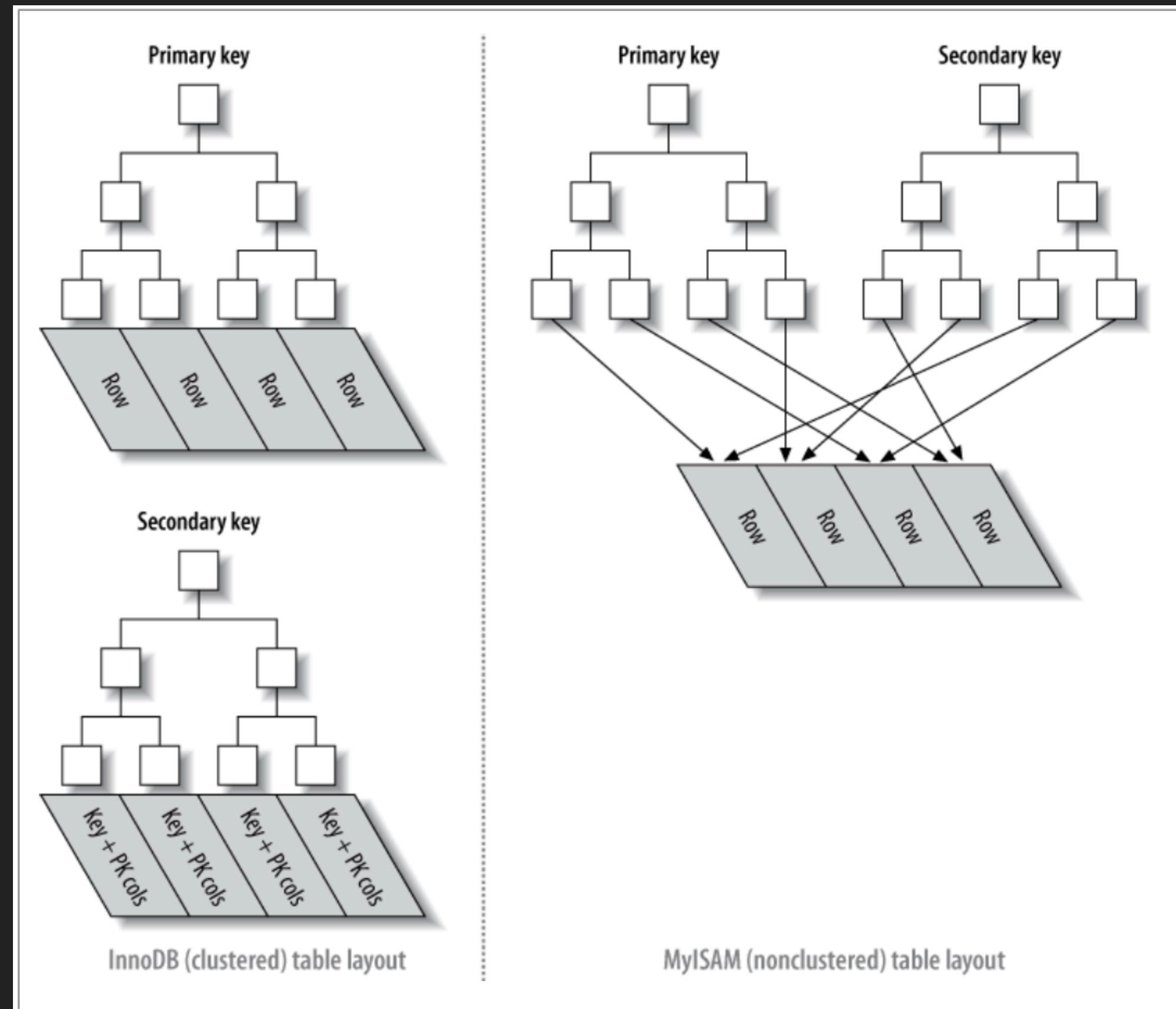
NEKLASTERIZOVANI INDEKSI

- ▶ Sinonim za sekundarne indekse.
- ▶ U stranicama listova nalazi se vrednost indeksirane kolone i vrednost primarnog ključa.
- ▶ Za indeksiranje se može koristiti prefiks ili celu kolonu, više kolona ili virtuelna kolona, gde se za ključ traženja ne navodi naziv kolone nego funkcija koja se primenjuje na neku od kolona.



Slika 7. Struktura neklasterizovanog indeksa kod InnoDB skladišta

PRIMARNI I SEKUNDARNI INDEKSI KOD INNODB I MYISAM SKLADIŠTA



Slika 8. Razlika u strukturi primarnih i sekundarnih indeksa kod InnoDB i MyISAM skladišta

SEKUNDARNI INDEKSI

```
create_people_sql = """
CREATE TABLE `people` (
  `id` int NOT NULL AUTO_INCREMENT,
  `first_name` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `last_name` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `email` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `city` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `country` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `phone_number` varchar(25) COLLATE utf8_unicode_ci NOT NULL,
  `birthdate` date NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
"""
```

Slika 9. Tabela people

```
mysql> create index name_composite on people(first_name, last_name desc);
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Slika 10. Kreiranje kompozitnog indeksa

```
mysql> explain select * from people where first_name like 'Christine' and last_name like 'Evans';
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows | filtered | Extra
+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE     | people | NULL      | range | name_composite | name_composite | 304    | NULL  | 1    | 100.00  | Using index condition
+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> explain select * from people where first_name like 'Christine';
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows | filtered | Extra
+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE     | people | NULL      | range | name_composite | name_composite | 152    | NULL  | 198 | 100.00  | Using index condition
+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> explain select * from people where first_name like 'Chris%';
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows | filtered | Extra
+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE     | people | NULL      | range | name_composite | name_composite | 152    | NULL  | 1392 | 100.00  | Using index condition
+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> explain select * from people where first_name like 'Christine' and (last_name like 'Evans' or last_name like 'Walker');
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows | filtered | Extra
+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE     | people | NULL      | range | name_composite | name_composite | 304    | NULL  | 2    | 100.00  | Using index condition
+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> explain select * from people where first_name like 'Christine' and last_name>='E' and last_name<='Z';
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows | filtered | Extra
+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE     | people | NULL      | range | name_composite | name_composite | 304    | NULL  | 150 | 100.00  | Using index condition
+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)
```

Slika 11. Upiti koji koriste kompozitni indeks

UNIQUE INDEKS

- ▶ Unique indeks uvodi ograničenje za kolonu nad kojom se kreira da sve vrednosti moraju biti jedinstvene.
- ▶ Za kreiranje unique indeksa može se koristiti jedna kolona, više kolona ili prefiks kolone.

```
[mysql] > alter table people add unique(phone_number);
Query OK, 0 rows affected (0.14 sec)
Records: 0  Duplicates: 0  Warnings: 0

[mysql] > explain select * from people where phone_number like '+381%';
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows | filtered | Extra          |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE      | people | NULL       | range | phone_number    | phone_number | 77     | NULL  | 181  | 100.00    | Using index condition |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)
```

Slika 12. Kreiranje unique indeksa za kolonu phone_number

FULLTEXT INDEKSI

- ▶ InnoDB i MyISAM skladišta podržavaju kreiranje fulltext indeksa nad kolonama tekstualnog tipa (CHAR, VARCHAR, TEXT).
- ▶ Fulltext pretraga koristi se kod upita gde u WHERE klauzuli imamo MATCH()...AGAINST()
- ▶ Struktura invertovanog indeksa, gde se za svaku reč pamti lista dokumenata i pozicija u dokumentu u vidu byte offset-a.

```
create_paragraphs_table = """  
CREATE TABLE `paragraphs` (  
    `id` int NOT NULL AUTO_INCREMENT,  
    `paragraph` varchar(4000) COLLATE utf8_unicode_ci NOT NULL,  
    PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;  
"""
```

Slika 13. Tabela paragraphs

FULLTEXT INDEKSI

- ▶ Kreira se skup indeksnih tabela
- ▶ Imena pomoćnih tabela počinju sa `fts_` i završavaju se sa `index_#`
- ▶ Povezane su sa indeksiranim tabelom uz pomoć heksadecimalne vrednosti, koja se nalazi u nazivu pomoćnih tabela.
- ▶ Ulazni dokumenti se tokenizuju, svaki token se upisuje u indeks tabele zajedno sa informacijama o poziciji i `DOC_ID`. Reči su potpuno sortirane i partitionisane unutar 6 indeksnih tabela na osnovu težine prvog karaktera.
- ▶ Ostale tabele su zajedničke za sve fulltext indekse, koriste se za rukovanje brisanjem i skladištenjem internog stranja indeksa.

```
mysql> create fulltext index words on paragraphs(paragraph);
Query OK, 0 rows affected, 1 warning (0.38 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql> select table_id, name, space from information_schema.innodb_tables where name like 'BazePodataka/%';
+-----+-----+-----+
| table_id | name          | space |
+-----+-----+-----+
| 1112    | bazepodataka/people      | 51    |
| 1152    | bazepodataka/locations    | 91    |
| 1173    | bazepodataka/fts_00000000000048e_being_deleted | 112   |
| 1174    | bazepodataka/fts_00000000000048e_being_deleted_cache | 113   |
| 1175    | bazepodataka/fts_00000000000048e_config      | 114   |
| 1176    | bazepodataka/fts_00000000000048e_deleted      | 115   |
| 1177    | bazepodataka/fts_00000000000048e_deleted_cache | 116   |
| 1167    | bazepodataka/fts_00000000000048e_000000000000139_index_1 | 106   |
| 1168    | bazepodataka/fts_00000000000048e_000000000000139_index_2 | 107   |
| 1169    | bazepodataka/fts_00000000000048e_000000000000139_index_3 | 108   |
| 1170    | bazepodataka/fts_00000000000048e_000000000000139_index_4 | 109   |
| 1171    | bazepodataka/fts_00000000000048e_000000000000139_index_5 | 110   |
| 1172    | bazepodataka/fts_00000000000048e_000000000000139_index_6 | 111   |
| 1166    | bazepodataka/paragraphs     | 105   |
+-----+-----+-----+
14 rows in set (0.00 sec)
```

Slika 14. Tabele fulltext indeksa

FULLTEXT INDEKSI

```
[mysql]> select * from paragraphs where match(paragraph) against("letter") and match(paragraph) against("prevent");
+----+
| id | paragraph
+----+
| 4457 | Rock language rock. Avoid career bad prevent. Them apply letter local newspaper represent. Hospital lay maybe
player across option election product.
| 8309 | Rate prepare adult picture. Tax pass moment nature fly prevent. Defense ahead southern under part family theor
y sign. Win upon central protect see letter century.
| 9725 | Behavior citizen next. Contain least visit control allow evidence Democrat. Mouth want letter mind series prev
ent. Could until second language civil send shoulder toward.
+----+
3 rows in set (0.01 sec)

[mysql]> explain select * from paragraphs where match(paragraph) against("letter") and match(paragraph) against("prevent");
+-----+
| id | select_type | table      | partitions | type      | possible_keys | key     | key_len | ref      | rows | filtered | E
xtra
+-----+
| 1 | SIMPLE      | paragraphs | NULL      | fulltext   | words       | words    | 0       | const    | 1 | 100.00 | U
sing where; Ft_hints: sorted |
+-----+
1 row in set, 1 warning (0.00 sec)
```

Slika 15. Primer upita za fulltext indeks

PROSTORNI INDEKSI

- ▶ Ograničenja koja kolone moraju da ispune:
 - ▶ Prostorni tip
 - ▶ Not null
 - ▶ Definisan SRID atribut
 - ▶ Koristi se cela kolona za indeksiranje
- ▶ Struktura R-stabla.
- ▶ Da bi se prostorni indeks koristio u procesu optimizacije neophodno je da kolone imaju definisan SRID atribut.
- ▶ Za podatke prostornog tipa postoje posebne funkcije, koje se mogu primeniti.

```
create_geo_table = """
CREATE TABLE `locations` (
    `id` int NOT NULL AUTO_INCREMENT,
    `coords` geometry NOT NULL SRID 4326,
    `city` varchar(255) UNIQUE NOT NULL,
    `country` varchar(5) NOT NULL,
    `continent` varchar(15) NOT NULL,
    `capital` varchar(50) NOT NULL,
    PRIMARY KEY (`id`),
    SPATIAL KEY `g` (`coords`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
"""
```

Slika 16. Tabela locations

PROSTORNI INDEKS

```
mysql> select city,st_astext(coords) as coordinates, country, continent from locations where st_within(coords,st_srid(st_geomfromtext('polygon((-10 -10, -10 10, 10 10, 10 -10, -10 -10))'),4326));
+-----+-----+-----+-----+
| city | coordinates | country | continent |
+-----+-----+-----+-----+
| Kafanchan | POINT(9.58126 8.2926) | NG | Africa |
| Efon-Alaaye | POINT(7.65649 4.92235) | NG | Africa |
| Ilesa | POINT(7.62789 4.74161) | NG | Africa |
| Olupona | POINT(7.6 4.18333) | NG | Africa |
| Shagamu | POINT(6.8485 3.64633) | NG | Africa |
| Tchaourou | POINT(8.88649 2.59753) | BJ | Africa |
| Mampong | POINT(7.06273 -1.4001) | GH | Africa |
| New Yekepa | POINT(7.57944 -8.53778) | LR | Africa |
| Idenao | POINT(4.2475 9.00472) | CM | Africa |
| Nkpor | POINT(6.15038 6.83042) | NG | Africa |
| Yenagoa | POINT(4.92675 6.26764) | NG | Africa |
| Tarkwa | POINT(5.30383 -1.98956) | GH | Africa |
| Bibiani | POINT(6.46346 -2.31938) | GH | Africa |
| Bonoua | POINT(5.27247 -3.59625) | CI | Africa |
| Lakota | POINT(5.84752 -5.682) | CI | Africa |
+-----+-----+-----+-----+
15 rows in set (0.00 sec)
```

Slika 17. Upit pronalazi lokacije obuhvaćene poligonom

```
mysql> explain select city,st_astext(coords) as coordinates, country, continent from locations where st_within(coords,st_srid(st_geomfromtext('polygon((-10 -10, -10 10, 10 10, 10 -10, -10 -10))'),4326));
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | locations | NULL | range | g | g | 34 | NULL | 15 | 100.00 | Using where |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)
```

Slika 18. EXPLAIN naredba za isti upit

ADAPTIVE HASH INDEX

- ▶ InnoDB skladište interno koristi heš indekse.
- ▶ Na osnovu zapaženih šabloni pretrage, kreira se heš indeks korišćenjem prefiksa ključa traženja.

```
[mysql]> set global innodb_adaptive_hash_index=OFF;  
Query OK, 0 rows affected (0.00 sec)
```

Slika 19. Isključivanje adaptive hash indeksa

HVALA NA PAŽNJI!