Optimiziranje MySQL baze i Laravela za rad s vrlo velikim bazama podataka

Nenad Crnko
Sveučilišni računski centar

srce
Sveučilište u Zagrebu
Sveučilišni računski centar

Sadržaj

- Uvod
- Optimizacija sustava za upravljanje bazom podataka (MySQL)
- Optimizacija logičkog modela baze podataka
- Optimizacija upita na bazu podataka
- Optimizacija razvojnog alata i aplikacije (PHP / Laravel)

Uvod

Što je to vrlo velika baza podataka?

Primjer definicije veličine prema broju slogova

- Mala baza do 100.000
- Srednja od 100.000 do 10.000.000
- Velika od 10.000.000 do 1.000.000.000
- Vrlo velika više od 1.000.000.000

Primjer definicije veličine prema dužini izvođenja upita

- Mala baza

  Upiti se izvode praktično trenutno bez obzira na indekse

- Srednja

  Upiti se izvode duže od sekunde bez odgovarajućih indeksa

- Velika i vrlo velika

  Potrebna je optimizacija indeksa i upita da bi se dobili prihvatljivi rezultati

Primjer definicije veličine prema načinu održavanja

- Mala baza

  Nije potreban DBA, jer se o svemu brine aplikacija

- Srednja

  Potreban je bar jedan DBA

- Velika i vrlo velika

  Potrebno je više DBA ili cijeli tim za konstatno praćenje rada i održavanje

Primjer definicije veličine prema korištenju hardverskih resursa

- Mala baza

  Cijeli skup podataka nad kojim se izvodi upit dostupan je u RAM memoriji servera

- Srednja

  Svi podaci iz baze mogu se spremiti na jedan ili više RAID sustava

- Velika i vrlo velika

  Baza podataka podijeljena je na veći broj servera s vlastitim RAID sustavima

Područja optimizacije

- Optimizacija hardverskih resursa (nije dio seminara)
- Optimizacija operativnog sustava (nije dio seminara)

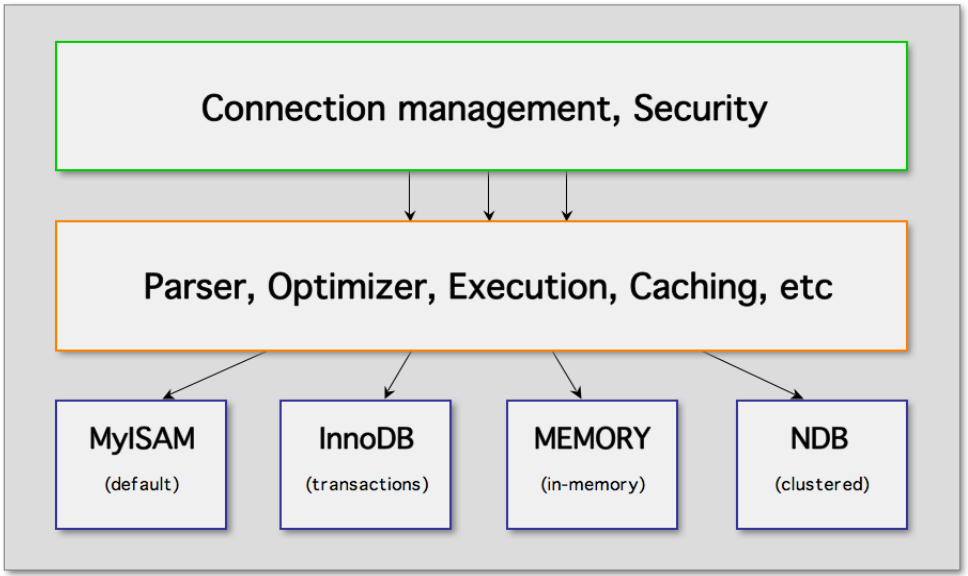
- Optimizacija sustava za upravljanje bazom podataka (MySQL)
- Optimizacija logičkog modela baze podataka
- Optimizacija upita na bazu podataka
- Optimizacija razvojnog alata i aplikacije (PHP / Laravel)

Primjer neoptimizirane tablice (Laravel)

```
Schema::create('users', function(Blueprint $table)
{
        $table->increments('id');
        $table->string('username');
        $table->string('fullname');
        $table->string('phone');
        $table->string('email')->unique();
        $table->string('password');
        $table->string(‘zipcode');
        $table->string(‘status');
        $table->text('comment');
        $table->timestamps();
});
```

Optimizacija sustava za upravljanje bazom podataka (MySQL)
1 dio

MySQL arhitektura - slojevi



MySQL arhitektura - slojevi

Connection management, Security

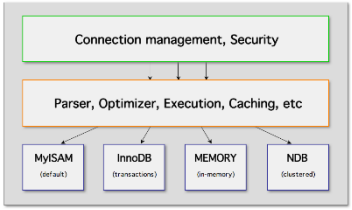
- upravljanje vezama prema bazi podataka, autentikacija, sigurnost,…

Parser, Optimizer, Caching…

- MySQL „mozak”
- Analiza upita, optimizacija, keširanje upita i podataka, podrška za poglede, okidače, funkcije, itd

Storage engines

- MySQL posebnost – spremanje i
- pristup podacima



MySQL storage engines

- Posebnost MySQL-a
- Postoji veći broj različitih sustava sa svojim prednostima i manama
- Više različitih proizvođača
- S višom razinom komuniciraju preko Storage API sustava
- Ne komuniciraju međusobno, ali se mogu koristiti istovremeno
- Neki od najpoznatijih sustava: MyISAM, InnoDB, Memory, NDB, Archive

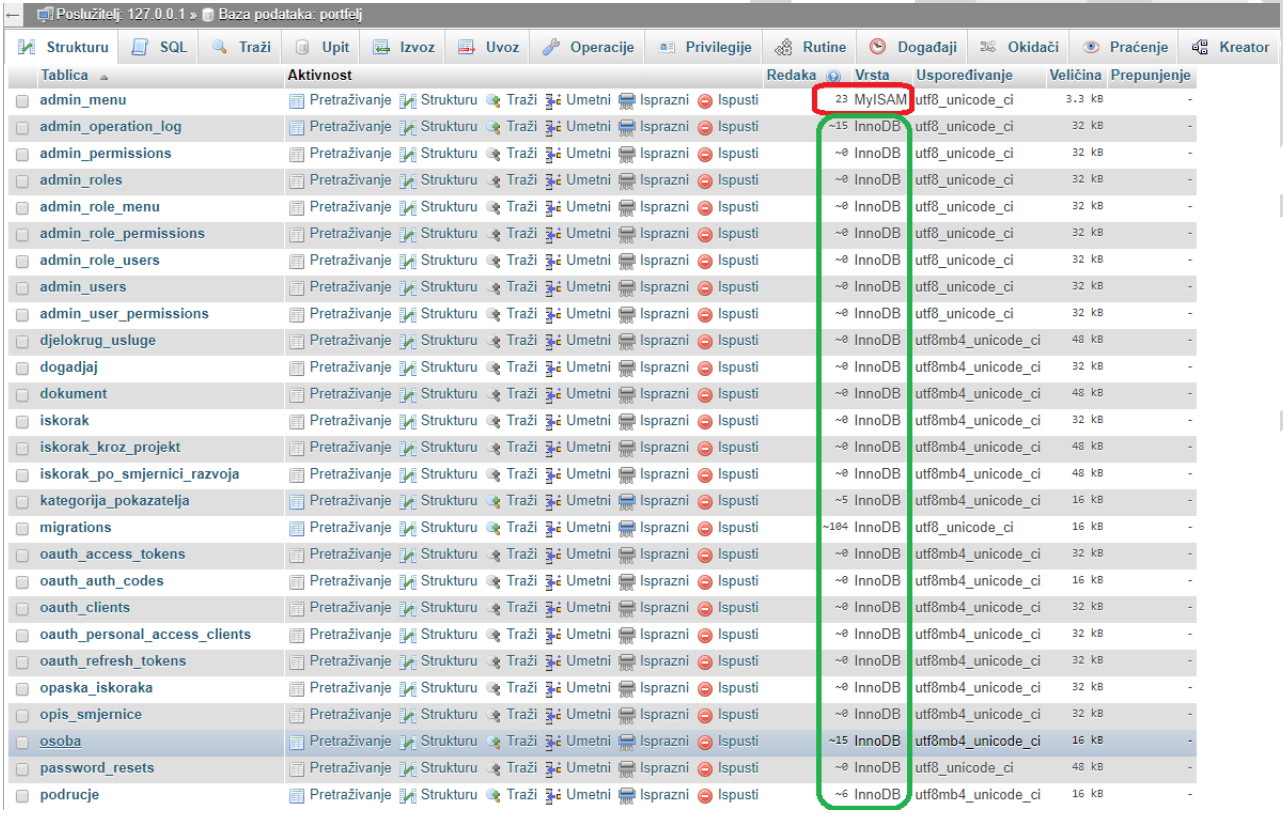
Usporedni prikaz karakteristika

Feature	MyISAM	Memory	InnoDB	Archive	NDB
Storage limits	256TB	RAM	64TB	None	384EB
Transactions	No	No	Yes	No	Yes
Locking granularity	Table	Table	Row	Row	Row
Multiversion Concurrency Control	No	No	Yes	No	No
Geospatial data type support	Yes	No	Yes	Yes	Yes
Geospatial indexing support	Yes	No	Yes	No	No
B-tree indexes	Yes	Yes	Yes	No	No
T-tree indexes	No	No	No	No	Yes
Hash indexes	No	Yes	No	No	Yes
Full-text search indexes	Yes	No	Yes	No	No
Clustered indexes	No	No	Yes	No	No
Data caches	No	N/A	Yes	No	Yes
Index caches	Yes	N/A	Yes	No	Yes
Compressed data	Yes	No	Yes	Yes	No
Encrypted data	Yes	Yes	Yes	Yes	Yes

Usporedni prikaz karakteristika

Cluster database support	No	No	No	No	Yes
Replication support	Yes	Yes	Yes	Yes	Yes
Foreign key support	No	No	Yes	No	Yes
Backup / point-in-time recovery	Yes	Yes	Yes	Yes	Yes
Query cache support	Yes	Yes	Yes	Yes	Yes
Update statistics for data dictionary	Yes	Yes	Yes	Yes	Yes

Primjer istovremenog korištenja



Poslužitelj: 127.0.0.1 » Baza podataka: portfelj

Strukturu | SQL | Traži | Upit | Izvoz | Uvoz | Operacije | Privilegije | Rutine | Događaji | Okidači | Praćenje | Kreator

Tablica	Aktivnost	Redaka	Vrsta	Uspoređivanje	Veličina	Prepunjenje
admin_menu	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	23	MyISAM	utf8_unicode_ci	3.3 kB	-
admin_operation_log	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~15	InnoDB	utf8_unicode_ci	32 kB	-
admin_permissions	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	32 kB	-
admin_roles	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	32 kB	-
admin_role_menu	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	32 kB	-
admin_role_permissions	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	32 kB	-
admin_role_users	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	32 kB	-
admin_users	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	32 kB	-
admin_user_permissions	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	32 kB	-
djelokrug_usluge	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	48 kB	-
dogadjaj	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
dokument	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	48 kB	-
iskorak	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
iskorak_kroz_projekt	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	48 kB	-
iskorak_po_smjernici_razvoja	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	48 kB	-
kategorija_pokazatelja	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~5	InnoDB	utf8mb4_unicode_ci	16 kB	-
migrations	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~104	InnoDB	utf8_unicode_ci	16 kB	-
oauth_access_tokens	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
oauth_auth_codes	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	16 kB	-
oauth_clients	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
oauth_personal_access_clients	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
oauth_refresh_tokens	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
opaska_iskoraka	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
opis_smjernice	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8mb4_unicode_ci	32 kB	-
osoba	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~15	InnoDB	utf8mb4_unicode_ci	16 kB	-
password_resets	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~0	InnoDB	utf8_unicode_ci	48 kB	-
podrucje	Pretraživanje Strukturu Traži Umetni Isprazni Ispusti	~6	InnoDB	utf8mb4_unicode_ci	16 kB	-

MyISAM

- Podrazumijevana tehnologija do verzije 5.1
- Podržava brojna napredna svojstva kao što su full-text indeksi, kompresija podataka ili GIS orijentirani tipovi podataka
- Glavni nedostaci su nedostatak podrške za transakcije, zaključavanje na razini tablice (uz dozvoljeni insert/select), te slabija otpornost na havarije sustava
- Treba izbjegavati za buduće projekte izuzev eventualno novih „read only” rješenja.

InnoDB

- Podrazumijevana storage engine za novije verzije MySQL-a od kada ga je preuzeo Oracle (verzija 5.5 i novije)
- Konstantno se radi na njegovom unapređenju i nadogradnji
- Podrška za većinu naprednih svojstava baza podataka uključujući transakcije i otpornost na havarije
- Optimalno rješenje za baze podataka implementirane na jednom serveru

Memory

- Svi podaci su smješteni u memoriji servera, a ne na disku pa nema potrebe za izvođenjem relativno sporih I/O operacija
- Podaci se gube nakon restarta servera pa se Memory engine ne koristi za trajno spremanje podataka
- Koristi se za privremeno spremanje manje bitnih podataka (npr. Sesije) ili read only podataka koji se često koriste (npr. Određeni šifrarnici)
- MySQL koristi za izvođenje dijela internih operacija

NDB

- Namijenjena prije svega za korištenje u distribuiranim okruženjima s većim brojem servera
- Podrška nije uključena u standardnu distribuciju MySQL-a
- Može se preuzeti s adrese http://dev.mysql.com/downloads/cluster/

Archive

- Namijenjena je za optimalno čuvanje velike količine neindeksiranih podataka (arhiviranje podataka)
- Uključena je u standardnu distribuciju MySQL-a
- Prilikom dodavanja podataka izvodi se njihova automatska kompresija (zlib)
- Nisu podržane SQL naredba UPDATE i DELETE
- Prilikom pretraživanja izvodi se full scan tablica

Još neke alternative

BLACKHOLE

- Omogućava zapisivanje podataka, ali ih u stvarnosti nigdje ne sprema

CSV

- Podaci se čuvaju u obliku tekstualnih tablica s Comma-Separated Values formatu

Percona XtraDB Storage

- Alternativa za InnoDB za MySQL i MariaDB

Infobright

- podrška za rad s bazama podataka koje se temelje na pristupu kolonama

Kako napraviti optimalan izbor

- U velikoj većini slučajeva optimalan izbor na jednom serveru bez obzira na veličinu baze je **InnoDB**
- Za velike arhivske baze podataka zbog uštede prostora može se koristiti **Archive**
- Za velike read only baze (pogotovo ako se distribuiraju na mediju) može se koristiti **MyISAM**
- U distribuiranom okruženju s više servera optimalan izbor može biti **NDB**

Pretvaranje tablica

- Potrebno je u slučaju zamjene storage engine na kojoj se temelji neka tablica.
- U slučaju velike količine podataka po tablici to može biti dugotrajan proces koji utječe na normalno korištenje baze.
- Postupak obavezno treba započeti backupom tablica koje sudjeluju u postupku pretvaranja.
- Postoji nekoliko načina s različitim prednostima i manama.

Pretvaranje tablica

Korištenjem naredbe ALTER TABLE

- Sa stanovišta korištenja najjednostavniji način, ali teško primjenjiv na vrlo velikim tablicama
- Primjer:
  - ALTER TABLE users ENGINE = InnoDB;
- Nedostaci
  - Postupak traje određeno vrijeme ovisno o veličini tablice
  - Za to vrijeme su „I/O” operacije pod velikim opterećenjem
  - Za vrijeme postupka tablica je zaključana
  - Tijekom postupka mogu se izgubiti neka svojstva tablica (npr. strani ključevi)

Pretvaranje tablica

Izvoz i ponovni uvoz podataka

- Za izvoz podataka koristi se alat mysqldump, a za uvoz mysqlimport
- Datoteka s izvezenim podacima može se urediti u editoru tako da tablica koristi drugu storage engine
- Problemi:
  - Uređivanje izvezenih podataka kod vrlo velikih tablica
  - Količina podataka na disku (originalna tablica, izvezeni podaci, nova tablica)

Pretvaranje tablica

Korištenjem naredbi CREATE TABLE i INSERT INTO … SELECT

Primjer:

- CREATE TABLE newtab LIKE oldtab;
- ALTER TABLE newtab ENGINE=InnoDB;
- INSERT INTO newtab SELECT * FROM oldtab;

Karakteristike:

- Nema „međupodataka” kao kod izvoza/uvoza.
- Ne može doći do oštećenja tablice kao kod izravne promjene tablice s ALTER TABLE
- Postupak se može provoditi i inkrementalno

Pretvaranje tablica

Korištenjem alata pt-online-schema-change (Percona Toolkit)

- Dodatni alat koji nije dio standardne distribucije MySQL-a
- Osim promjene vrste storage engine može se koristiti i za druge operacije nad strukturom tablice
- Tijekom korištenja ne dolazi do zaključavanja tablice
- Primjer korištenja:

```
pt-online-schema-change --alter "ENGINE=InnoDB"
D=sakila,t=actor
```

Optimizacija logičkog modela baze podataka

Odabir optimalnih tipova podataka (opća pravila)

Treba izabrati **najmanju moguću veličinu** podatka koja je dovoljna za spremanje određene vrste podataka

- Zauzimaju najmanje mjesta na disku, RAM memoriji i kešu procesora
- U pravilu je potrebno i najmanje resursa procesora za njihovu obradu
- Treba paziti da se ne izabere „premala” veličina podataka, jer kasnije izmjene sheme mogu zahtijevati dosta vremena i resursa

Odabir optimalnih tipova podataka (opća pravila)

Treba izabrati **najjednostavniju vrstu podatka** koja je dovoljna za spremanje određene vrste podataka

- U slučaju da dvije vrste podataka imaju istu veličinu za spremanje podataka treba izabrati jednostavniji tip podataka
- Na primjer – kod spremanje nekoliko mogućih vrijednosti mnogo je djelotvojnije koristiti cjelobrojne vrijednosti (1, 2, 3, 4 …) nego znakove (A, B, C, D) zbog kompliranijeg sortiranja i uspoređivanja (collations)

Odabir optimalnih tipova podataka (opća pravila)

Pravilno odrediti potrebu za korištenjem Null vrijednosti u definiciji stupca

- Određene vrste storage enginea imaju lošije performanse kod korištenja null vrijednosti
- Postavljanje indeksa na stupce s velikim brojem null vrijednosti također može dovesti do smanjenja performansi
- Kod upita na bazu ponekad treba imati „duplu provjeru” u samom upitu

Odabir optimalnih tipova podataka (pregled)

MySQL DATA TYPES

DATE TYPE	SPEC	DATA TYPE	SPEC
CHAR	String (0 - 255)	INT	Integer (-2147483648 to 214748-3647)
VARCHAR	String (0 - 255)	BIGINT	Integer (-9223372036854775808 to 9223372036854775807)
TINYTEXT	String (0 - 255)	FLOAT	Decimal (precise to 23 digits)
TEXT	String (0 - 65535)	DOUBLE	Decimal (24 to 53 digits)
BLOB	String (0 - 65535)	DECIMAL	"DOUBLE" stored as string
MEDIUMTEXT	String (0 - 16777215)	DATE	YYYY-MM-DD
MEDIUMBLOB	String (0 - 16777215)	DATETIME	YYYY-MM-DD HH:MM:SS
LONGTEXT	String (0 - 4294967295)	TIMESTAMP	YYYYMMDDHHMMSS
LONGBLOB	String (0 - 4294967295)	TIME	HH:MM:SS
TINYINT	Integer (-128 to 127)	ENUM	One of preset options
SMALLINT	Integer (-32768 to 32767)	SET	Selection of preset options
MEDIUMINT	Integer (-8388608 to 8388607)	BOOLEAN	TINYINT(1)

Brojčani tipovi podataka

Cjelobrojni tipovi podataka – treba ih izabrati uvijek uvijek kad je to moguće

- Zahtijevaju manje resursa procesora kod obrade
- Veličina podatka u bajtovima određena je tipom podatka INTEGER, INT, SMALLINT, TINYINT, MEDIUMINT, BIGINT
- Tipovi podataka s podrškom za negativne vrijednosti ne troše više prostora ni resursa
- Ograničavanje veličine cjelobrojnog podatka kod definicije (npr. int(5), int(11)) ne utječe na veličinu zauzetog prostora u bazi nego samo na njihov prikaz / unos u određenim alatima

Brojčani tipovi podataka

Brojevi s decimalnim zarezom

- Decimalni tip podataka treba izabrati isključivo kad je potrebna točna preciznost u decimalama (najčešće novčani iznosi)
- U svim ostalim slučajevima treba izabrati FLOAT ili DOUBLE zbog manjih zahtjeva za resursima procesora
- Trik za drastično povećanje performansi obrade brojčanih vrijednosti
  - zamjena DECIMAL vrijednosti s INT ili BIGINT gdje zadnje cjelobrojne vrijednosti u stvari predstavljaju decimale

Znakovni tipovi CHAR i VARCHAR

- CHAR – spremanje niza znakova fiksne dužine
- VARCHAR – spremanje niza znakova promjenjive dužine
- Na prvi pogled sa stanovišta zauzeća prostora na disku uvijek je bolje izabrati VARCHAR s dodatnom rezervom u prostoru, kako se ne bi morala izvoditi naknadna povećanja veličine (dugotrajan postupak na vrlo velikim bazama podataka)
- Postoji nekoliko situacija kad prethodno pravilo dolazi u pitanje i tip podatka CHAR predstavlja bolje rješenje

Znakovni tipovi CHAR i VARCHAR

- Primjer stvarnog zauzeća prostora, jer VARCHAR troši dodatne bajtove za bilježenje stvarne dužine niza znakova

  CHAR(5) = 5 bajtova

  VARCHAR(5) = 6 bajtova

  CHAR(1000) = 1000 bajtova

  VARCHAR(1000) = 1002 bajta
- Za vrlo kratke nizove znakova (1 - 3), u tom smislu CHAR predstavlja bolje rješenje
- CHAR predstavlja bolje rješenje i za nizove znakova uvijek iste dužine (npr. OIB, pošte)

Znakovni tipovi CHAR i VARCHAR

- Problem fragmentacije VARCHAR podataka nakon naknadnog povećanja veličine, jer na isto mjesto na disku ne može biti smješten veći podatak
  - MyISAM – fragmentira podatke
  - InnoDB – ažurira memorijske stranice

- Zbog navedenog problema treba obratiti pažnju na reprezentativnost eventualnih početnih ili test podataka tipa VARCHAR (ne smiju biti bitno manji od stvarnih)

Znakovni tipovi CHAR i VARCHAR

- Problem korištenja nepotrebno velikih VARCHAR tipova podataka u RAM memoriji
- Memory storage engine za izvođenje određenih operacija nad podacima u RAM memoriji kreira privremene tablice, a pri tome nije podržan VARCHAR tip podatka nego samo CHAR
- Nepotrebno veliki CHAR podaci mogu uzrokovati da se privremene tablice iz memorije „presele” na disk što bitno degradira performanse u radu
- Zato treba težiti minimalnoj potrebnoj veličini VARCHAR podataka prilikom definiranja tablica

Znakovni tipovi BINARY i VARBINARY

- Po brojnim karakteristikama slični tipovima CHAR i VARCHAR
- Glavna razlika je u načinu spremanje znakova – umjesto znakova definiranih prema određenoj sekvenci znakova u bazu se spremaju nizovi bajtova
- Uspoređivanje i sortiranje znakova je bitno brže neko kod tipova CHAR i VARCHAR, ako je prikladan takav način rada (npr. razlike u rasporedu posebnih znakova)

Tipovi podataka BLOB i TEXT

- Koriste se za spremanje velike količine binarnih ili znakovnih podataka koji premašuju ostale znakovne tipove podataka
- Dostupno je nekoliko podtipova kao što su MEDIUM, LONG
- Na poseban način spremaju se na disk pa je u glavnoj tablici samo pokazivač na mjesto gdje se stvarno nalazi pojedini objekt u tom formatu
- Mogu bitno utjecati na performanse cijelog sustava, posebno ako se nad njima izvode operacije uspoređivanja ili sortiranja

Tipovi podataka BLOB i TEXT

- Nisu podržani u Memory storage engine pa svako korištenje u privremenim tablicama zahtijeva njihovo kreiranje na disku što bitno degradira performanse sustava
- Zbog toga treba izbjegavati korištenje BLOB i TEXT podataka u operacijama uspoređivanja i sortiranja
- Ako to nije moguće treba iskoristiti trik i izvoditi operacije na temelju relevatnog početnog broja znakova pretvorenih u standardni znakovni tip. Na primjer: order by left(description,100)

Tipovi podataka DATE, TIME, DATETIME, TIMESTAMP

- Koriste se redom spremanje datuma, vremena, datuma i vremena, te datuma i vremena u manjem rasponu.
- Zauzimaju redom 3, 3, 8 i 4 bajtova.
- Njavažnija odluka je odabir DATETIME (raspon '1000-01-01' do '9999-12-31‘) ili TIMESTAMP (raspon 1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' UTC) prema mogućoj pojavi podataka u bazi.

Tip podatka ENUM

- Omogućava bitnu uštedu prostora kod korištenja nekoliko standardnih znakovnih vrijednosti u redovima tablice. Umjesto definiranih nizova znakova sprema se samo brojčana vrijednost veličine 1 bajt
- Trebalo bi koristiti samo u slučaju da standardni niz vrijednosti nije promjenjiv, ili se eventualno može dodati na sam kraj popisa (treba koristiti ALTER TABLE)
- Smanjuje performanse sustava ako se koristi u povezivanju tablica (JOIN)

Tip podatka ENUM

- Primjer korištenja
- CREATE TABLE shirts (name VARCHAR(40), size ENUM('x-small', 'small', 'medium', 'large', 'x-large'));
- INSERT INTO shirts (name, size) VALUES ('dress shirt','large'), ('t-shirt','medium'), ('polo shirt','small');
- SELECT * from shirts;

name	size
dress shirt	large
t-shirt	medium
polo shirt	small

Odabir tipova podataka za primarne i vanjske ključeve

- Sa stanovišta performansi najbolje je da to budu cijeli brojevi s tendencijom rasta (može im se dodati i auto increment)
- Tip podatka ENUM predstavlja vrlo loš izbor i treba ga izbjegavati
- Nizovi znakova kao primarni identifikatori također sa stanovišta performansi ne predstavljaju optimalan izbor (izbjeći ako se može)
- Odabir identifikatora bilo kojeg tipa sa slučajnim vrijednostima podataka predstavlja loš odabir sa stanovišta performansi

Kako optimizirati definiciju tablice

```
Schema::create('users', function(Blueprint $table)
{
        $table->increments('id');
        $table->string('username');
        $table->string('fullname');
        $table->string('phone');
        $table->string('email')->unique();
        $table->string('password');
        $table->string(‘zipcode');
        $table->string(‘status');
        $table->text('comment');
        $table->timestamps();
});
```

Kako optimizirati definiciju tablice

- Ograničiti veličinu string (VARCHAR) vrijednosti
- Stupac Zipcode pretvoriti u CHAR zbog fiksne duljine
- Stupac Phone također se može pretvoriti u CHAR u pod određenim pretpostavkama
- Stupac Status može se zamijeniti s ENUM
- Stupac Comments može se zamijeniti s jednom ili više VARCHAR vrijednosti

Naknadna izmjena definicije tablice

- Standardni način je korištenje ALTER TABLE naredbe
- Automatski postupak pripreme nove tablice, kopiranje podataka iz stare tablice, te brisanje stare tablice na kraju
- Kod vrlo velikih tablica postupak može trajati satima
- Dio promjena izvodi se bez potrebe za ponovnim kreiranjem tablica (npr. default vrijednosti)

Naknadna izmjena definicije tablice

- Primjer – MyISAM (3 sec vs 2.4 sec)

  ALTER TABLE article DISABLE KEYS;

  ALTER TABLE article MODIFY COLUMN content_path varchar(251) NOT NULL DEFAULT '-';

  ALTER TABLE article ENABLE KEYS;

- Primjer – InnoDB (95 sec vs 72 sec)

  SET autocommit=0; SET unique_checks=0; SET foreign_key_checks=0;

  ALTER TABLE article_innodb MODIFY COLUMN content_path varchar(251) NOT NULL DEFAULT '-';

  SET autocommit=1; SET unique_checks=1; SET foreign_key_checks=1;

„Prljava alternativa” za InnoDB

Zamjena za ALTER TABLE – koraci

1. Na temelju postojeće tablice kreirati novu praznu sa željenim promjenama
2. Zaključati korištenje tablica s

   FLUSH TABLES WITH READ LOCK
3. Zamijeniti .frm datoteke u kojima se nalazi definicija strukture tablice

   (npr. new_article_innodb.frm > article_innodb.frm)
4. Omogućiti nesmetano korištenje tablica s

   UNLOCK TABLES

Normalizacija i denormalizacija

Customer Name	Customer Address	Customer Tel No.	Product Name	Unit Cost	Quantity	Total Cost
Alex Wilson	1318 Scenic Avenue, Bothel	697-555-0142	Men's Sports Shorts, S	15.5	2	31
Alex Wilson	1318 Scenic Avenue, Bothel	697-555-0142	Water Bottle - 30 oz.	1.5	3	4.5
Alex Wilson	1318 Scenic Avenue, Bothel	697-555-0142	LL Mountain Handlebars	19.76	2	39.52
Emily Brown	628 Muir Road, Los Angeles	708-555-0141	Long-Sleeve Logo Jersey, S	38.49	1	38.49
Emily Brown	628 Muir Road, Los Angeles	708-555-0141	Sport-100 Helmet, Black	13.08	2	26.16
Emily Brown	628 Muir Road, Los Angeles	708-555-0141	LL Mountain Handlebars	19.76	3	59.28

Customers

Customer ID	Customer Name	Customer address	Customer Tel No.
25	Alex Wilson	1318 Scenic Avenue, Bothel	697-555-0142
48	Emily Brown	628 Muir Road, Los Angeles	708-555-0141

Orders

Order ID	Customer ID
ORD002	25
ORD004	25
ORD009	25
ORD100	48
ORD110	48
ORD125	48

Products

Product ID	Product Name	Unit Cost
P003	LL Mountain Handlebars	19.76
P020	Long-Sleeve Logo Jersey, S	38.49
P065	Sport-100 Helmet, Black	13.08
P080	Men's Sports Shorts, S	15.5
P085	Water Bottle - 30 oz.	1.5

Order Details

Order ID	Product ID	Quantity
ORD002	P080	2
ORD004	P085	3
ORD009	P003	2
ORD100	P020	1
ORD110	P065	2
ORD125	P003	3

Prednosti normalizirane baze

- Nema ponavljajućih podataka pa je održavanje (dodavanje i ažuriranje) podataka u bazi jednostavnije i brže
- Normalizirane baze su obično manje pa se obično bolje iskorištava postojeća RAM memorija servera
- Manja potreba za korištenjem upita s dijelovima poput GROUP BY ili DISTINCT koji utječu na brzinu njihovog izvođenja

Nedostaci normalizirane baze

- Prvenstveno su povezani s performansama kod čitanja i složenije analize podataka
- Upiti obično zahtijevaju veći broj JOIN operacija između tablica što može bitno utjecati na brzinu izvođenja upita
- Takve (optimalne) upite je teže pisati nego kad se svi podaci nalaze u jednoj tablici

Denormalizirana baza

- Podaci iz normalizirane baze podataka se ponovno spajaju u manji broj tablica kako bi pretraživanje i analiza podataka bila jednostavnija i brža
- Denormalizirane tablica mogu biti izdvojene u posebnu bazu podataka na drugom serveru, a sinhronizacija podataka između normalizirane i denormalizirane baze može se izvoditi posebnim upitima ili postupkom replikacije
- Ovakav pristup zahtijeva dodatne resurse u pogledu hardvera

Optimizacija normalizirane baze za čitanje

- Namjerno dodavanje redudantnih podataka u normalizirane tablice
- Primjeri
  - Podaci o broju poruka ili sljedbenika iz child tablica mogu se dodati u master tablice
  - Podaci o ukupnom iznosu narudžbi, porezima i slično mogu se iz tablice s detaljima o narudžbi dodati u osnovnu tablicu narudžbe
  - Suprotan postupak – podaci iz master tablica mogu se dodati u child tablice zbog bržeg pretraživanja ili sortiranja

Dodavanje tablica za keširanje i sumiranje

- Primjer – analiza broja poruka o određenoj temi za određeni period na društvenim mrežama, ili preračunavanje cijena artikala
- Izravan upit s odgovarajućim JOIN operacija između tablica može trajati dosta vremena, a ionako ne može obuhvatiti najnovije podatke nastale tijekom izvođenja upita
- Alternativno rješenje je priprema posebne tablice sa stupcima za spremanje teme i datuma/vremena obrade koja se puni periodičnom obradom
- Postoji kašnjenje u ažurnosti podataka, ali to isto vrijedi i za izravni upit na tablicu
- Naknadne analize podataka izvode se na dodatnim tablicama
- Postoji mogućnost istovremenog korištenja različitih storage enginea
- Zamjena za materialized views koji nisu podržani u MySQL

Dodavanje tablica za brojače

- Umjesto da se tablice za keširanje pune povremenim obradama podataka, pomoćne tablice za brojenje događaja mogu se puniti tijekom nastanka događaja
- Treba paziti da punjenje takvih pomoćnih tablica ne postane usko grlo u radu cijelog sustava kod ogromnog broja događaja koji se događaju i trebaju bilježiti istovremeno
- Problem se može riješiti pravilnim dizajniranjem tablica za brojače

Primjer tablice za brojače

```
CREATE TABLE tmpcounter (
    datapart        tinyint unsigned not null,
    domain          varchar(50) not null,
    counter         int unsigned not null,
    PRIMARY KEY (datapart, domain)
) ENGINE=InnoDB
```

```
UPDATE tmpcounter SET counter = counter + 1
WHERE datapart = RAND() * 100 and domain = ‘sport’
```

Optimizacija indeksa - uvod

Indeksi ubrzavaju operacije pretraživanja na bazi podataka, ali usporavaju operacije ažuriranja podataka jer osim samih podataka treba održavati i indekse

Vrste indeksa

PRIMARY, INDEX, UNIQUE, FULLTEXT

Podjela prema internoj strukturi

B-TREE, HASH, R-TREE, Fraktalni, ...

Korištenje B-TREE indeksa kod pretraživanja

Primjer tablice

```
CREATE TABLE osoba (
prezime varchar(50) not null,
fime varchar(50) not null,
datum_rodjenja date not null,
spol enum('m', 'f')not null,
...
key(prezime, ime, datum_rodjenja));
```

Kad indeksi ubrzavaju pretraživanje?

- Postoji potpuno podudaranje sa svim vrijednostima u stupcima
- Postoji potpuno podudaranje prema prvom stupcu
- Postoji podudaranje prema početnom dijelu prvog stupca
- Traži se raspon vrijednosti prema prvom stupcu
- Postoji potpuno podudaranje prema prvoj vrijednosti i traži se raspon vrijednosti prema drugom stupcu

Slučajevi kad nisu od koristi

- Kad se pretraživanje ne izvodi prema prvom lijevom stupcu
- Kad se kod pretraživanja preskače određena kolona u indeksa (npr. traži se pretraživanje prema prvoj i trećoj koloni)
- Kad se se u određenoj koloni izvodi pretraživanje prema rasponu za kolone desno od nje se ne koristi indeks

```
WHERE prezime ="Horvat" AND Ime LIKE 'I%' AND
    dob='1970-12-01'
```

(treba probati izbjeći LIKE navođenjem popisa vrijednosti, ako je to moguće)

Slučajevi kad nisu od koristi

Kad se kod stupcima tablice u indeksima izvode operacije

WHERE

TO_DAYS(CURRENT_DATE) - TO_DAYS(datum_rodjenja)

<= 365

WHERE ID + 5 = 10

Dodatne činjenice o B-TREE indeksima

- Po potrebi treba pripremiti nove indekse s drugačijim rasporedom kolona ako je to potrebno
- Raspored kolona ovisi i o tome je li primarni cilj pretraživanje ili sortiranje
- Ako se neki indeks koristi kod pretraživanja, onda pomaže i kod sortiranja prema istom kriteriju
- Covering indeksi omogućavaju čitanje svih vrijednosti izravno iz indeksa

HASH indeksi

- Posebna hash vrijednost (relativno mala u veličini) koja se preračunava za svaki red u tablice
- Podrazumijevani tip indeksa za Memory storage engine
- InnoDB – posebna vrsta (adaptive hash indeks) za najčešće vrijednosti u B-TREE indeksima
- Postoji mogućnost da veći broj redova tablica dobije istu vrijednost
- Mnogo brži način za pronalaženje točno određenog podatka u tablici od B-TREE indeksa
- Mogu se simulirati korištenjem CRC32 funkcije

Ograničenja HASH indeksa

- Ne mogu se koristiti za ubrzavanje sortiranja
- Ne mogu se koristiti kao covering indeksi
- Ne mogu se koristiti kod pretraživanja po djelomičnom podudaranju vrijednosti
- Ubrzavaju samo točne operacije kod pretraživanja (npr. =, ili IN)
- U slučaju da u bazi postoji veliki broj istih hash vrijednosti padaju performanse pretraživanja
- Isto vrijedi i za neke druge operacije poput brisanja

Simuliranje HASH indeksa

- Može se koristiti za ubrzavanje operacija nad složenim podacima (npr. URL adresama)
- Primjer tablice i održavanja tablice trigerima

```
CREATE TABLE IF NOT EXISTS `urlsearch`
(
    `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
    `url` varchar(1000) NOT NULL,
    `urlcrc` int(10) unsigned NOT NULL,
     PRIMARY KEY (`id`),  KEY `idxurlcrc` (`urlcrc`)
) ENGINE=InnoDB
```

Simuliranje HASH indeksa

- Primjer upita za pretraživanje

```
SELECT id FROM url
WHERE url_crc=CRC32("http://
    http://www.index.hr/vijesti/clanak/infografika-koja-je-
    razlika-izmedju-hidrogenske-i-obicne-nuklearne-
    bombe/992309.aspx ")
AND url=" http://www.index.hr/vijesti/clanak/infografika-koja-je-
    razlika-izmedju-hidrogenske-i-obicne-nuklearne-
    bombe/992309.aspx ";
```

'Prefix' indeksi

- Posebna vrsta B-TREE indeksa kod kojih se indeksi kreiraju samo na početnim znakovima nekog stupca
- Može se bitno smanjiti veličina indeksa kod stupaca s velikim tekstualnim vrijednostima
- Primjer kreiranja indeksa

```
ALTER TABLE address ADD KEY (city_name(7))
```

Problem utvrđivanja relevantnosti prefiksa

```
SELECT city_name, count(*) FROM address group by city_name

SELECT left(city_name,2), count(*) FROM address group by left(city_name,2)
```

InnoDB cluster indeksi

- Podaci u tablici sortiraju se prema vrijednostima cluster indeksa – samo jedan indeks po tablici
- Ubrzano je pretraživanje, ali je usporeno dodavanje i ažuriranje, ako nova vrijednost ne dolazi na kraj tablice
- Zauzimaju više prostora od B-TREE indeksa.
- Zato se cluster indeksi često koriste za dodatne (surogat) primarne ključeve. Mnogo djelotvornije od random primarnih ključeva.
- Nakon ažuriranja velike količine podataka dobro je koristiti OPTIMIZE TABLE.
- Dvostruko korištenje indeksa kod pretraživanja

Korištenje indeksa kod sortiranja

- Vrijede ista pravila relevantnosti kao i kod pretraživanja
- Dodatna ograničenja
- Kod JOIN operacija svi stupci u ORDER BY moraju biti u istom indeksu
- ASC i DESC vrijednosti moraju biti jednako definirane kao u indeksu
- Ako se je sortiranje bitno i često može se kreirati novi indeks koji inače nije relevantan kod pretraživanja
- Posebni indeksi za povremeno sortiranje mogu se kreirati i brisati samo kad za tim postoji potreba

Nepotrebni i redudatni indeksi

- Privremene indekse (npr. korištene za sortiranje) treba ukloniti kad nisu potrebni.
- Primjer redudatnih indeksa:

  Indeks (stupac1, stupac2)

  Indeks (stupac1) - u većini slučajeva redudantan, ali ne uvijek

  Indeks (stupac2)

- Pronalaženje redudatnih indeksa

  Upiti na bazu podataka information_schema

  Alati: mysqlindexcheck, Percona *pt-duplicate-key-checker*

Nekorišteni indeksi

- Ne koriste se u operacijama pretraživanja i sortiranja a postoje u bazi podataka
- MySQL nepotrebno troši vrijeme na njihovo ažuriranje kod ostalih operacija
- Pronalaženje nekorištenih indeksa

  INFORMATION_SCHEMA.INDEX_STATISTICS

  (enable userstats)

  Percona pt-index-usage

Fragmentacija tablica i indeksa

- Tijekom dužeg perioda korištenja i tablice i indeksi mogu postati fragmentirati, što utječe na performanse.

- Vrste fragmentacije podataka:
  - Fragmentacija redova
  - Fragmentacija unutar redova
  - Fragmentacija slobodnog prostora

Defragmentacija tablica i indeksa

- Načini defragmentacije
  - OPTIMIZE TABLE
  - ALTER TABLE <tablica> ENGINE=<storage engine>
  - Export/import podataka

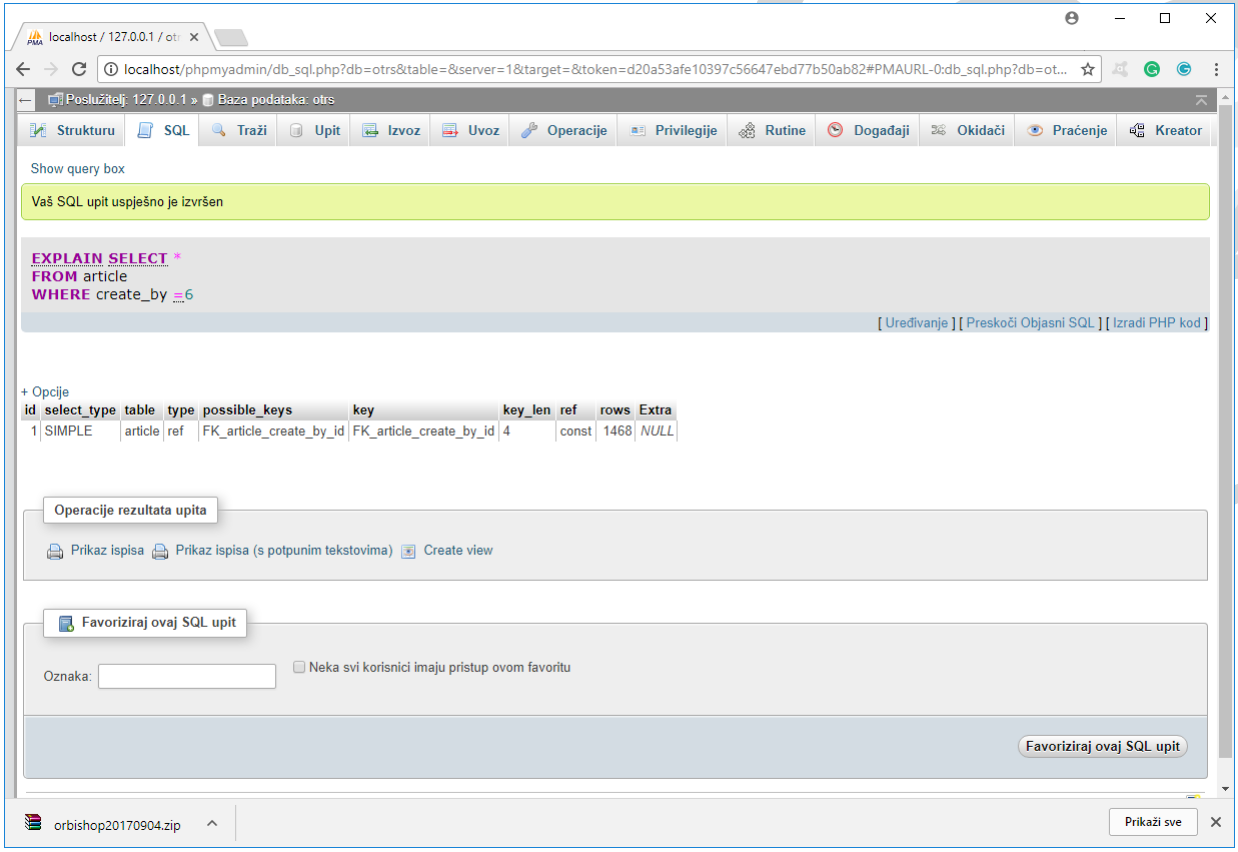
Optimizacija upita na bazu podataka

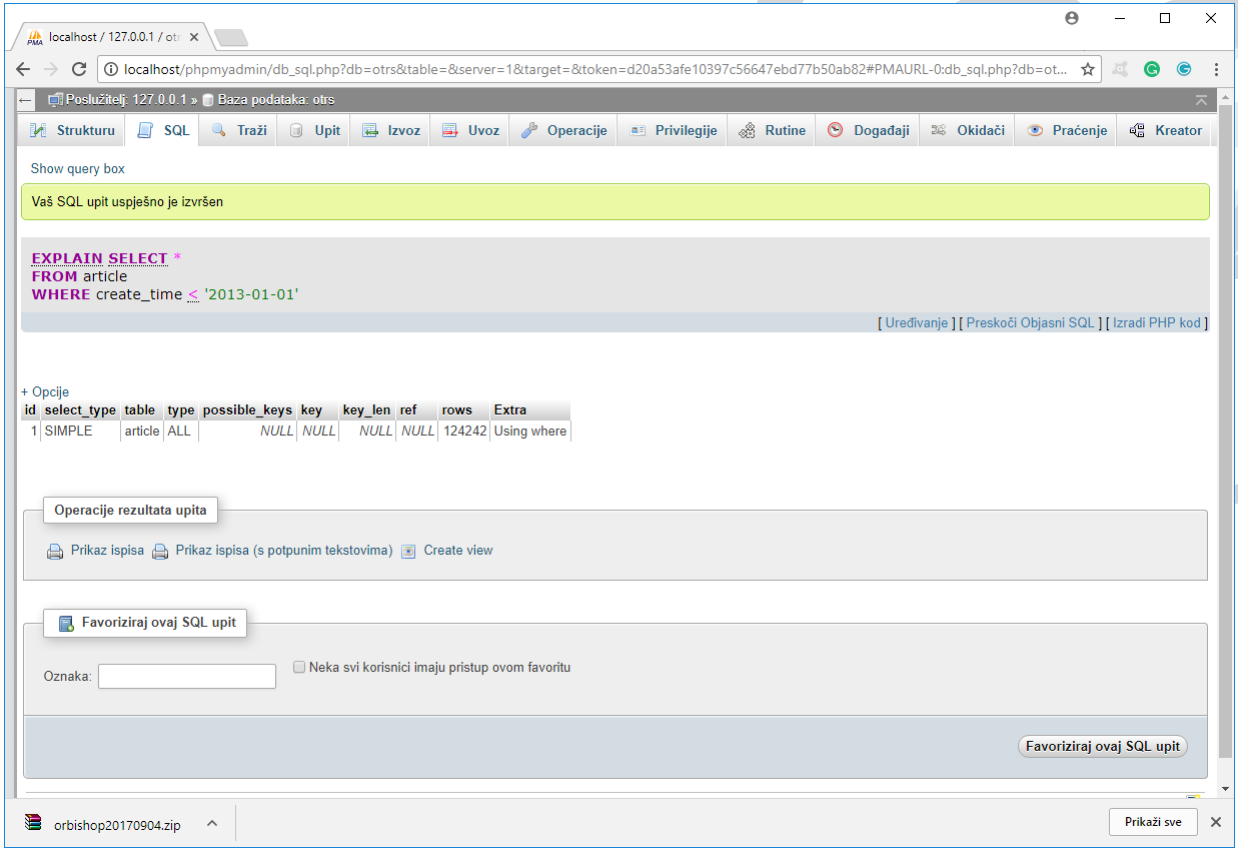
Preduvjeti za brzo izvođenje upita

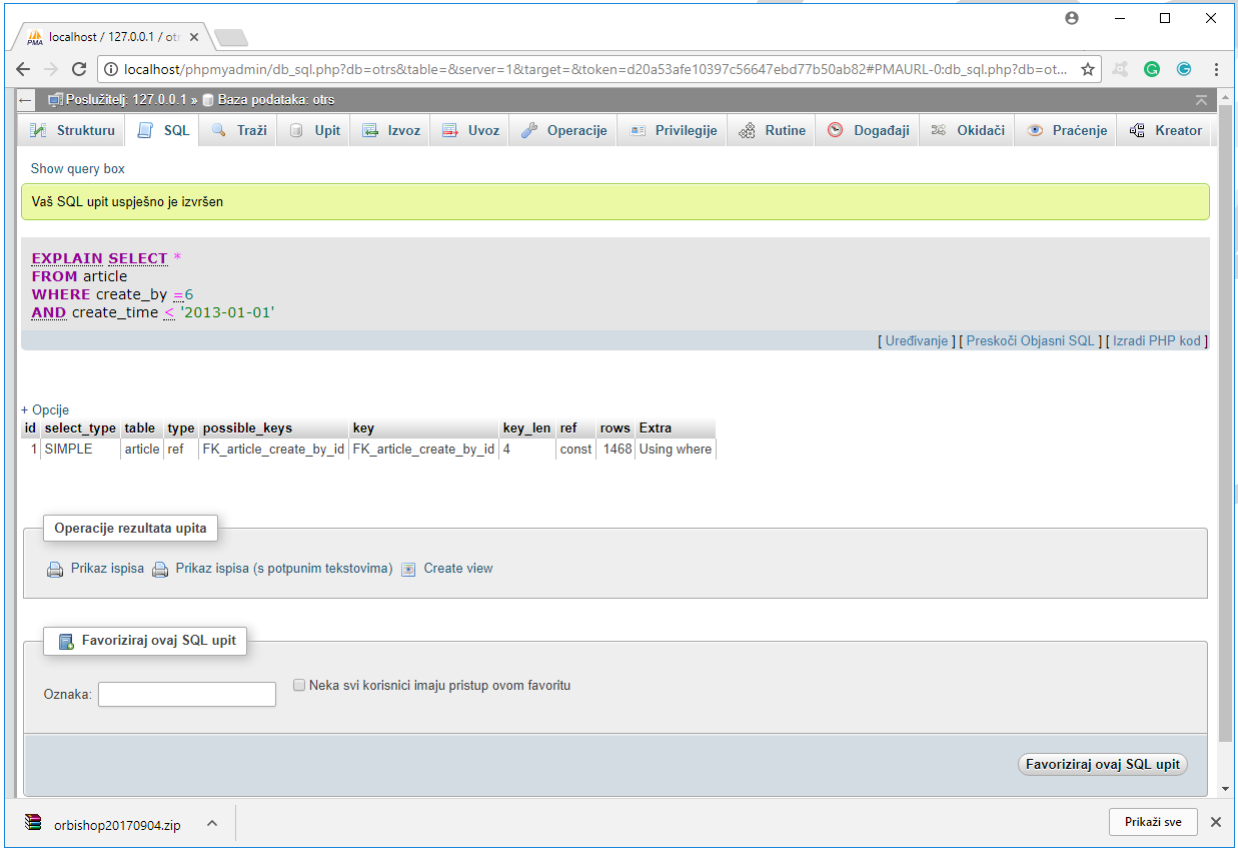
- Usko su povezani s obrađenim točkama:
  - Optimizacija sustava za upravljanje bazom podataka (MySQL)
  - Optimizacija logičkog modela baze podataka
- Dobro poznavanje logičkog dizajna baze podataka
- Brzina izvođenja upita ne ovisi samo o vremenu potrebnom da MySQL pročita slogove iz baze, nego to obuhvaća i analizu upita, zaključavanje resursa, prijenos podataka po mreži i slično

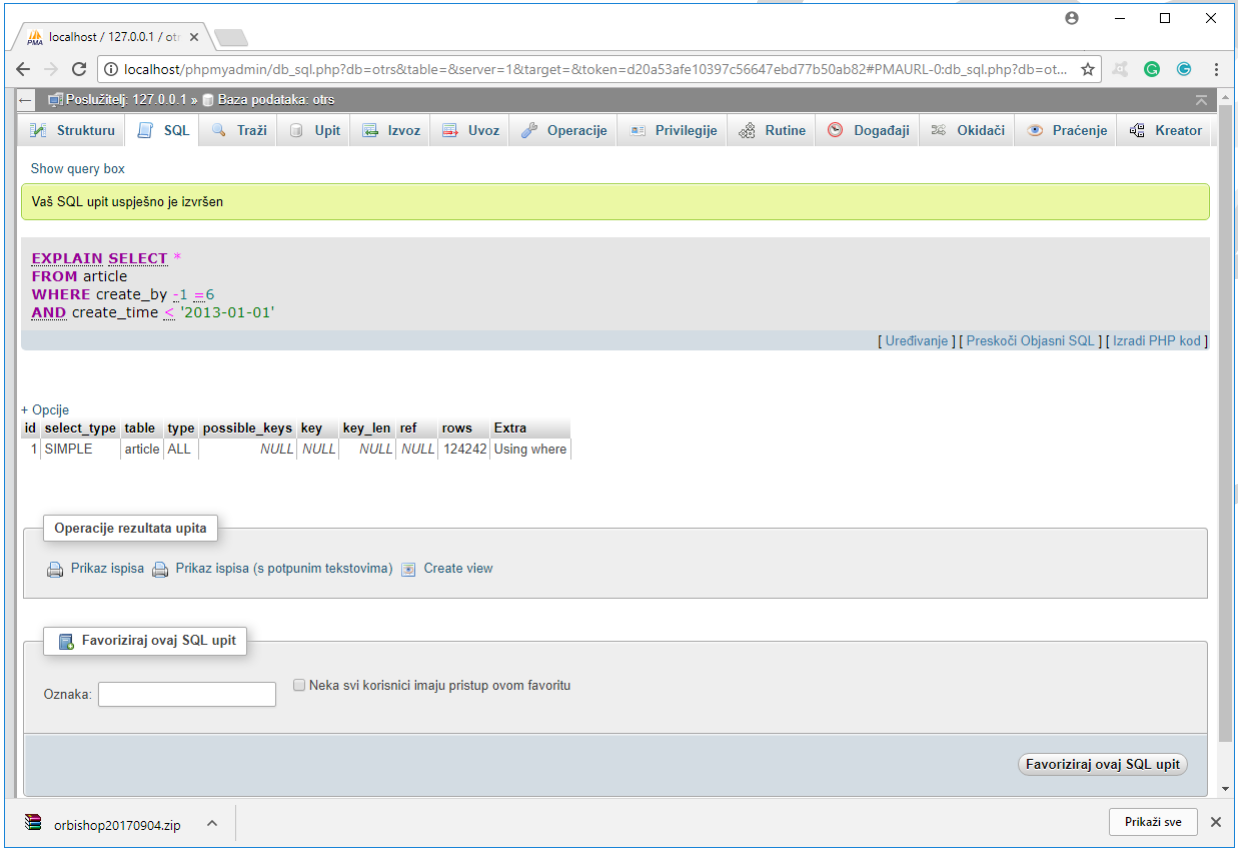
Koraci u izvođenju upita i analiza izvođenja

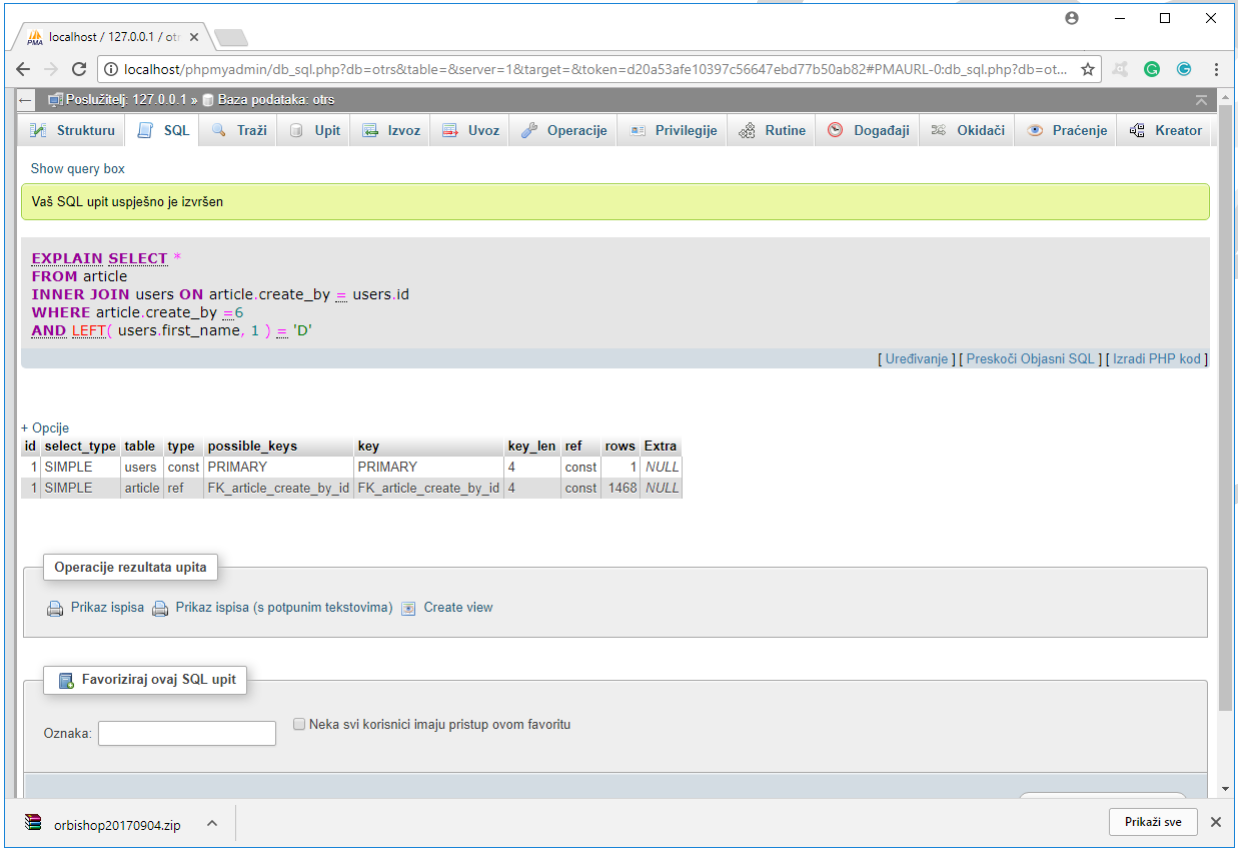
1. Klijent (aplikacija) šalje upit serveru
2. Server provjerava cache. Ako u njemu postoje rezultati za potpuno identični upit, rezultati se vraćaju i keša i time se dovršava izvođenje upita
3. Server analizira, obrađuje i optimizira upit za izvođenje, to jest priprema plan izvođenja
4. Query execution engine izvodi plan izvođenja pozivanje odgovarajućih API funkcija u storage engine
5. Rezultati izvođenja vraćaju se klijentu.

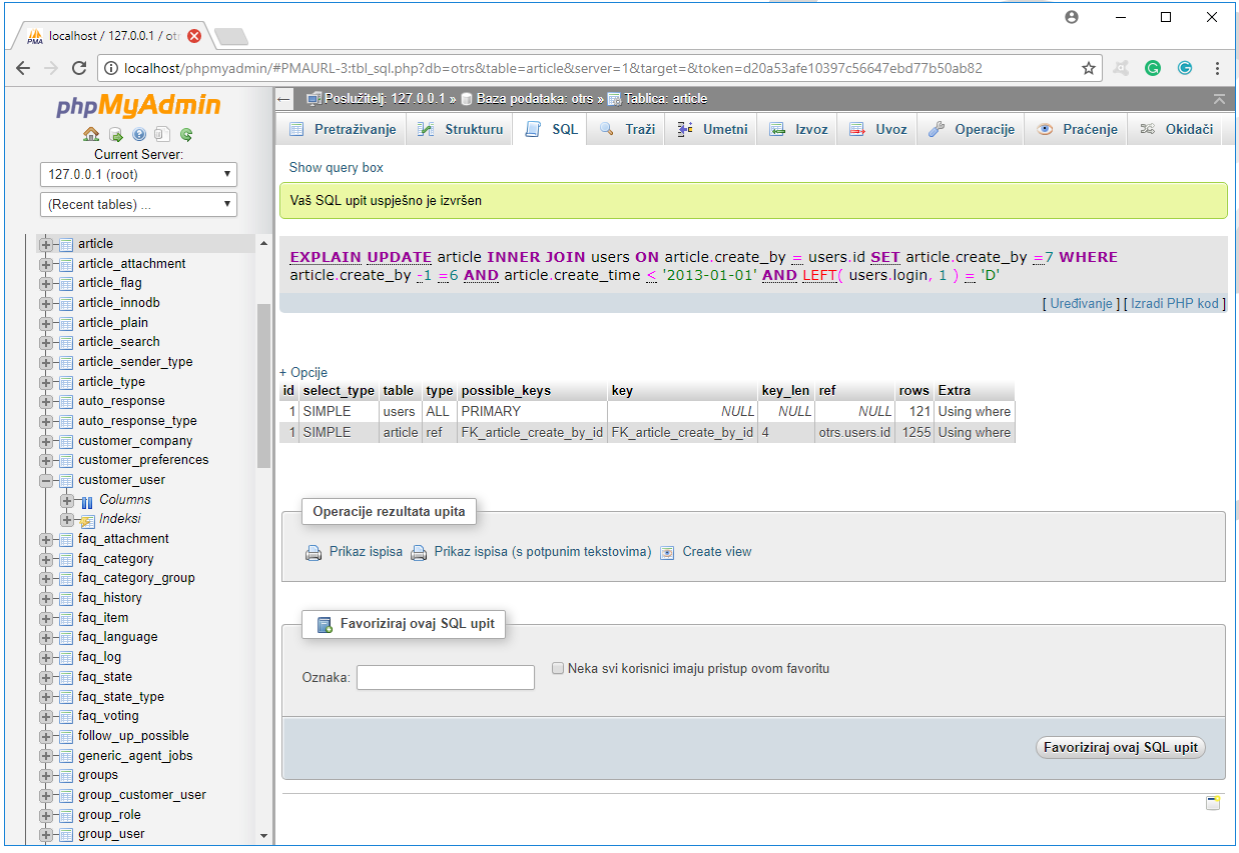

localhost / 127.0.0.1 / otr
localhost/phpmyadmin/db_sql.php?db=otrs&table=&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82#PMAURL-0:db_sql.php?db=ot...
Poslužitelj: 127.0.0.1 » Baza podataka: otrs
Strukturu
SQL
Traži
Upit
Izvoz
Uvoz
Operacije
Privilegije
Rutine
Događaji
Okidači
Praćenje
Kreator
Show query box
Vaš SQL upit uspješno je izvršen
EXPLAIN SELECT *
FROM article
WHERE create_by =6
[Uređivanje] [Preskoči Objasni SQL] [Izradi PHP kod]
+ Opcije
id select_type table type possible_keys key key_len ref rows Extra
1 SIMPLE article ref FK_article_create_by_id FK_article_create_by_id 4 const 1468 NULL
Operacije rezultata upita
Prikaz ispisa
Prikaz ispisa (s potpunim tekstovima)
Create view
Favoriziraj ovaj SQL upit
Oznaka:
Neka svi korisnici imaju pristup ovom favoritu
Favoriziraj ovaj SQL upit
orbishop20170904.zip
Prikaži sve

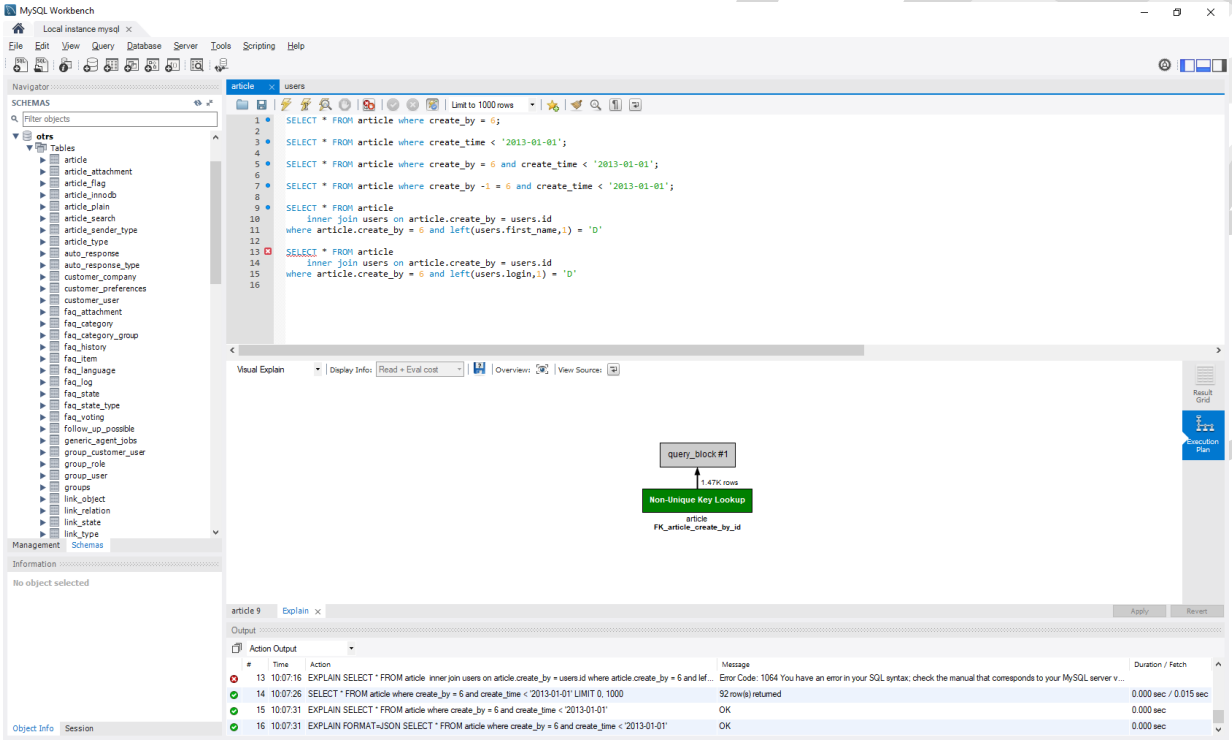

localhost / 127.0.0.1 / otr
localhost/phpmyadmin/db_sql.php?db=otrs&table=&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82#PMAURL-0:db_sql.php?db=ot...
Poslužitelj: 127.0.0.1 » Baza podataka: otrs
Strukturu
SQL
Traži
Upit
Izvoz
Uvoz
Operacije
Privilegije
Rutine
Događaji
Okidači
Praćenje
Kreator
Show query box
Vaš SQL upit uspješno je izvršen
EXPLAIN SELECT *
FROM article
WHERE create_time < '2013-01-01'
[Uređivanje] [Preskoči Objasni SQL] [Izradi PHP kod]
+ Opcije
id select_type table type possible_keys key key_len ref rows Extra
1 SIMPLE article ALL NULL NULL NULL NULL 124242 Using where
Operacije rezultata upita
Prikaz ispisa
Prikaz ispisa (s potpunim tekstovima)
Create view
Favoriziraj ovaj SQL upit
Oznaka:
Neka svi korisnici imaju pristup ovom favoritu
Favoriziraj ovaj SQL upit
orbishop20170904.zip
Prikaži sve


localhost / 127.0.0.1 / otr
localhost/phpmyadmin/db_sql.php?db=otrs&table=&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82#PMAURL-0:db_sql.php?db=ot...
Poslužitelj: 127.0.0.1 » Baza podataka: otrs
Strukturu
SQL
Traži
Upit
Izvoz
Uvoz
Operacije
Privilegije
Rutine
Događaji
Okidači
Praćenje
Kreator
Show query box
Vaš SQL upit uspješno je izvršen
EXPLAIN SELECT *
FROM article
WHERE create_by =6
AND create_time < '2013-01-01'
[Uređivanje] [Preskoči Objasni SQL] [Izradi PHP kod]
+ Opcije
id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra
1 | SIMPLE | article | ref | FK_article_create_by_id | FK_article_create_by_id | 4 | const | 1468 | Using where
Operacije rezultata upita
Prikaz ispisa
Prikaz ispisa (s potpunim tekstovima)
Create view
Favoriziraj ovaj SQL upit
Oznaka:
Neka svi korisnici imaju pristup ovom favoritu
Favoriziraj ovaj SQL upit
orbishop20170904.zip
Prikaži sve


localhost / 127.0.0.1 / otr
localhost/phpmyadmin/db_sql.php?db=otrs&table=&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82#PMAURL-0:db_sql.php?db=ot...
Poslužitelj: 127.0.0.1 » Baza podataka: otrs
Strukturu
SQL
Traži
Upit
Izvoz
Uvoz
Operacije
Privilegije
Rutine
Događaji
Okidači
Praćenje
Kreator
Show query box
Vaš SQL upit uspješno je izvršen
EXPLAIN SELECT *
FROM article
WHERE create_by =1 =6
AND create_time < '2013-01-01'
[Uređivanje] [Preskoči Objasni SQL] [Izradi PHP kod]
+ Opcije
id select_type table type possible_keys key key_len ref rows Extra
1 SIMPLE article ALL NULL NULL NULL NULL 124242 Using where
Operacije rezultata upita
Prikaz ispisa
Prikaz ispisa (s potpunim tekstovima)
Create view
Favoriziraj ovaj SQL upit
Oznaka:
Neka svi korisnici imaju pristup ovom favoritu
Favoriziraj ovaj SQL upit
orbishop20170904.zip
Prikaži sve


localhost / 127.0.0.1 / otr
localhost/phpmyadmin/db_sql.php?db=otrs&table=&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82#PMAURL-0:db_sql.php?db=ot...
Poslužitelj: 127.0.0.1 » Baza podataka: otrs
Strukturu
SQL
Traži
Upit
Izvoz
Uvoz
Operacije
Privilegije
Rutine
Događaji
Okidači
Praćenje
Kreator
Show query box
Vaš SQL upit uspješno je izvršen
EXPLAIN SELECT *
FROM article
INNER JOIN users ON article.create_by = users.id
WHERE article.create_by =6
AND LEFT(users.first_name, 1) = 'D'
[Uređivanje] [Preskoči Objasni SQL] [Izradi PHP kod]
+ Opcije
id select_type table type possible_keys key key_len ref rows Extra
1 SIMPLE users const PRIMARY PRIMARY 4 const 1 NULL
1 SIMPLE article ref FK_article_create_by_id FK_article_create_by_id 4 const 1468 NULL
Operacije rezultata upita
Prikaz ispisa
Prikaz ispisa (s potpunim tekstovima)
Create view
Favoriziraj ovaj SQL upit
Oznaka:
Neka svi korisnici imaju pristup ovom favoritu
orbishop20170904.zip
Prikaži sve


localhost / 127.0.0.1 / otr
localhost/phpmyadmin/#PMAURL-3:tbl_sql.php?db=otrs&table=article&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82
phpMyAdmin
Current Server:
127.0.0.1 (root)
(Recent tables) ...
article
article_attachment
article_flag
article_innodb
article_plain
article_search
article_sender_type
article_type
auto_response
auto_response_type
customer_company
customer_preferences
customer_user
Columns
Indeksi
faq_attachment
faq_category
faq_category_group
faq_history
faq_item
faq_language
faq_log
faq_state
faq_state_type
faq_voting
follow_up_possible
generic_agent_jobs
groups
group_customer_user
group_role
group_user
Poslužitelj: 127.0.0.1 » Baza podataka: otrs » Tablica: article
Pretraživanje
Strukturu
SQL
Traži
Umetni
Izvoz
Uvoz
Operacije
Praćenje
Okidači
Show query box
Vaš SQL upit uspješno je izvršen
EXPLAIN UPDATE article INNER JOIN users ON article.create_by = users.id SET article.create_by =7 WHERE article.create_by =1 =6 AND article.create_time < '2013-01-01' AND LEFT(users.login, 1) = 'D'
[Uređivanje] [Izradi PHP kod]
+ Opcije
id select_type table type possible_keys key key_len ref rows Extra
1 SIMPLE users ALL PRIMARY NULL NULL NULL 121 Using where
1 SIMPLE article ref FK_article_create_by_id FK_article_create_by_id 4 otrs.users.id 1255 Using where
Operacije rezultata upita
Prikaz ispisa
Prikaz ispisa (s potpunim tekstovima)
Create view
Favoriziraj ovaj SQL upit
Oznaka:
Neka svi korisnici imaju pristup ovom favoritu
Favoriziraj ovaj SQL upit


MySQL Workbench
Local instance mysql
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
otrs
Tables
article
article_attachment
article_flag
article_innodb
article_plain
article_search
article_sender_type
article_type
auto_response
auto_response_type
customer_company
customer_preferences
customer_user
faq_attachment
faq_category
faq_category_group
faq_history
faq_item
faq_language
faq_log
faq_state
faq_state_type
faq_voting
follow_up_possible
generic_agent_jobs
group_customer_user
group_role
group_user
groups
link_object
link_relation
link_state
link_type
Management Schemas
Information
No object selected
Object Info Session
article users
Limit to 1000 rows
1 SELECT * FROM article where create_by = 6;
3 SELECT * FROM article where create_time < '2013-01-01';
5 SELECT * FROM article where create_by = 6 and create_time < '2013-01-01';
7 SELECT * FROM article where create_by -1 = 6 and create_time < '2013-01-01';
9 SELECT * FROM article
10 inner join users on article.create_by = users.id
11 where article.create_by = 6 and left(users.first_name,1) = 'D'
13 SELECT * FROM article
14 inner join users on article.create_by = users.id
15 where article.create_by = 6 and left(users.login,1) = 'D'
Visual Explain
Display Info: Read + Eval cost
Overview:
View Source:
query_block #1
1.47K rows
Non-Unique Key Lookup
article
FK_article_create_by_id
Result Grid
Execution Plan
article 9 Explain
Apply Revert
Output
Action Output
Time Action Message Duration / Fetch
13 10:07:16 EXPLAIN SELECT * FROM article inner join users on article.create_by = users.id where article.create_by = 6 and lef... Error Code: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MySQL server v...
14 10:07:26 SELECT * FROM article where create_by = 6 and create_time < '2013-01-01' LIMIT 0, 1000 92 row(s) returned 0.000 sec / 0.015 sec
15 10:07:31 EXPLAIN SELECT * FROM article where create_by = 6 and create_time < '2013-01-01' OK 0.000 sec
16 10:07:31 EXPLAIN FORMAT=JSON SELECT * FROM article where create_by = 6 and create_time < '2013-01-01' OK 0.000 sec

Pravila kod pisanja upita

- Treba ograničiti broj redova na koje utječe upit na što manji broj (obrada, prijenos, zaključavanje) - LIMIT
- Treba ograničiti broj stupaca na koje utječe upit
- Treba ograničiti broj ponavljajućih upita na bazu (optimizacija na razini aplikacije)
- Prilikom pisanja upita treba optimalno koristiti indekse (posebno složene)

Pravila kod pisanja upita

- Po potrebi prije upita dodati indekse, a poslije izvođenja ukloniti
- Treba koristiti denormalizirane podatke u tablicama ako postoje
- Treba koristiti dodatne pomoćne tablice ako postoje (sumarne tablice i tablice brojači)
- Ako je moguće treba iskoristiti postojeće pohranjene procedure

Pravila kod pisanja upita

- Ako je moguće izvođenje složenih upita pomaknuti u vrijeme manjeg opterećenja servera
- Uzeti u obzir druge korisnike koji koriste isti server zbog načina pisanja upita i vremena izvođenja
- Postavke servera po potrebi definirati na razini izvođenja upita
- **Obavezno mjeriti performanse upita**

Dodatni parametri za optimizaciju upita

- HIGH_PRIORITY, LOW_PRIORITY
  - Prioritet SELECT naredbe u odnosu na druge naredbe koje pristupaju istoj tablici (MyISAM)
- STRAIGHT_JOIN
  - Spajanje tablica točno prema redoslijedu navođenja u upitu
- SQL_SMALL_RESULT, SQL_BIG_RESULT
  - Definiraju način na koji će se čuvati rezultati (memorija ili tablice na disku)
- SQL_BUFFER_RESULT
  - Brzo oslobađanje resursa kod izvođenja upita

Dodatni parametri za optimizaciju upita

- SQL_CACHE, SQL_NO_CACHE
  - Korištenje ili zabrana korištenja keša
- SQL_CALC_FOUND_ROWS
  - Priprema upita tako se izračuna ukupan broj redova bez obzira na korištenje LIMIT opcije
- FOR UPDATE, LOCK IN SHARE MODE
  - Zaključavanje slogova kod SELECT naredbe (InnoDB)
- USE INDEX, FORCE INDEX, IGNORE INDEKS
  - Definira upute o korištenju ili zabrani korištenja indeksa
- DELAYED – odgođeno zapisivanje u bazu

Optimizacija sustava za upravljanje bazom podataka (MySQL)
2 dio

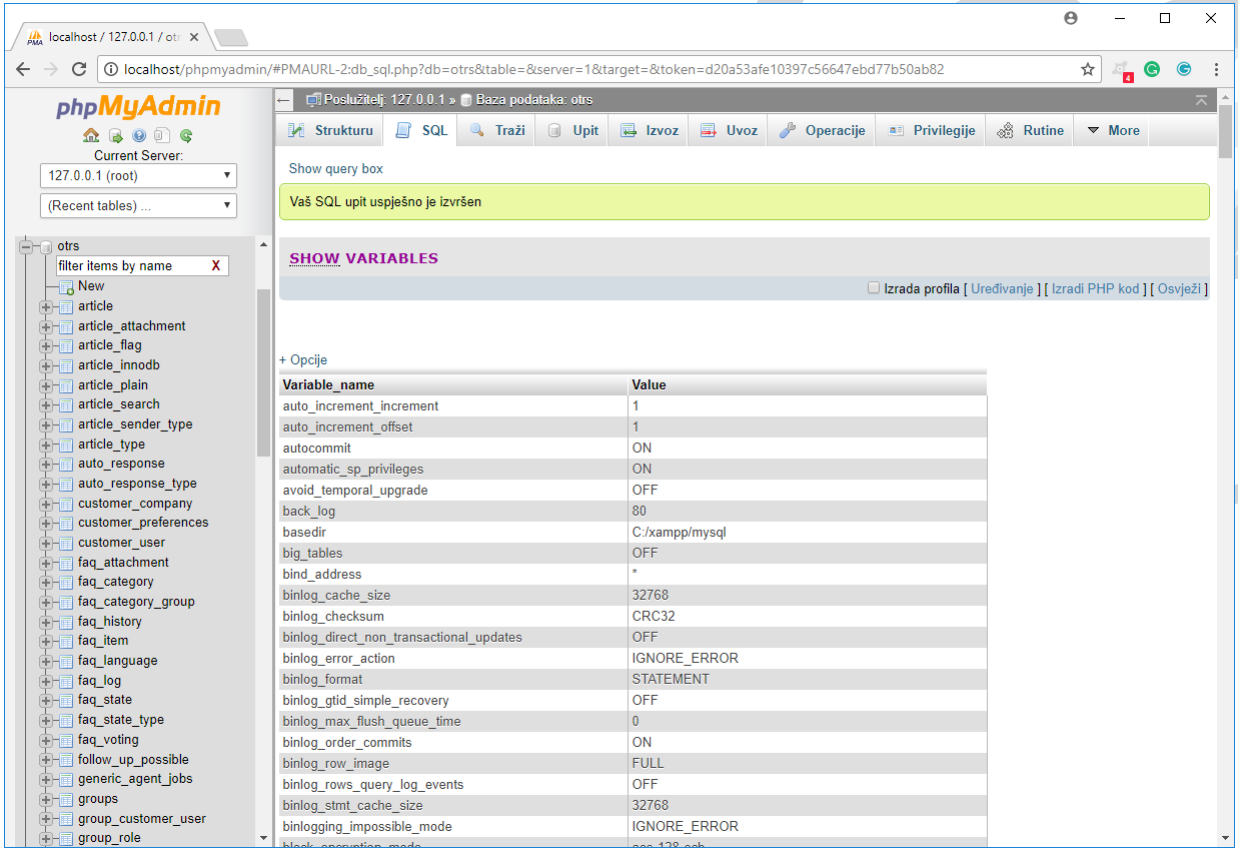
Parametri za definiranje konfiguracije

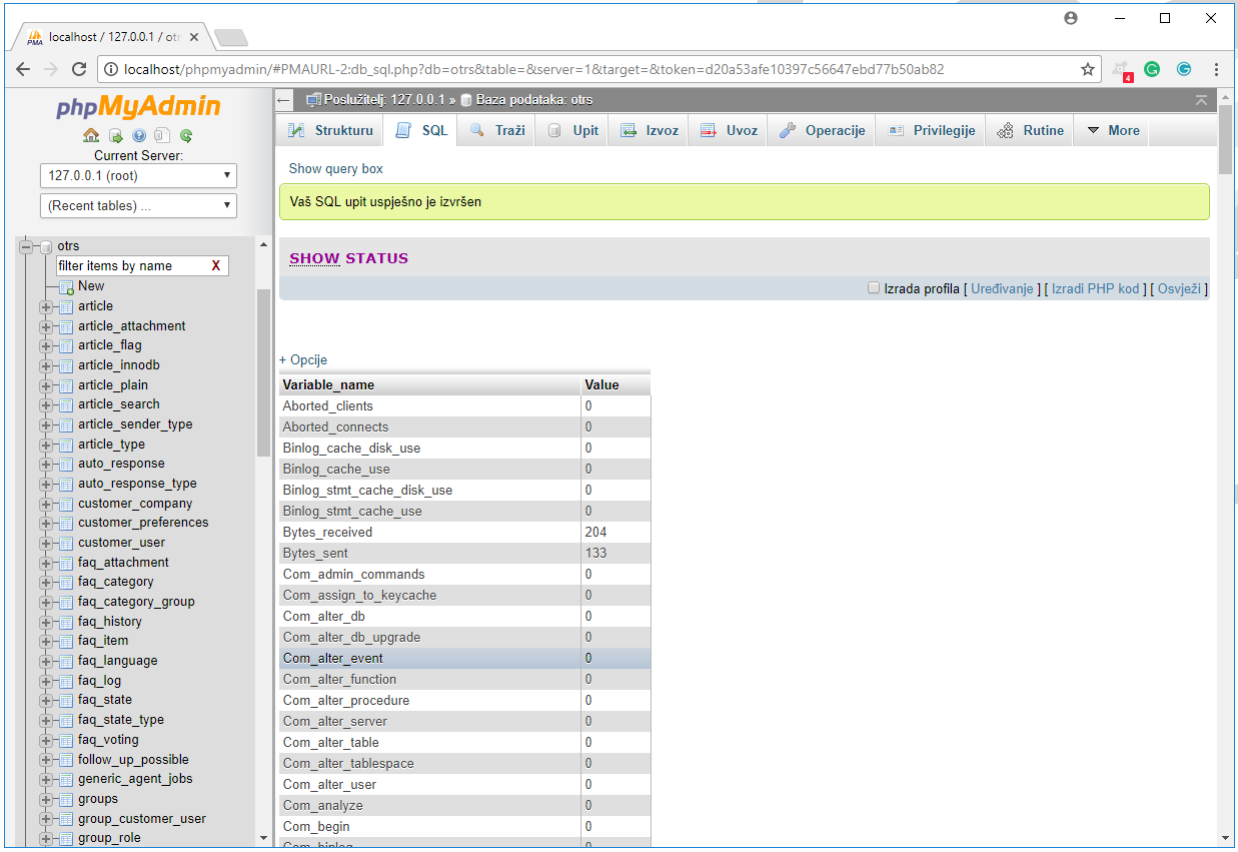
- U podrazumijevanom stanju (nakon instalacije) konfiguracija MySQL-a nije optimizirana za vrlo velike baze, tako da je u pravilu potrebno izmijeniti neke parametre
- Prilikom izmjene parametara treba voditi računa o ukupnom korištenju servera
- Eventualno probati koristiti automatske skripte
- Nakon svake izmjene parametara treba napraviti mjerenja o utjecaju na rad MySQL-a

Parametri za definiranje konfiguracije

- Ponekad je bolje dio parametara izmijeniti privremeno, nego ih imati kao podrazumijevane vrijednosti tijekom cijelog vremena rada servera

- U pravilu je bolje potrošiti vrijeme na upoznavanje i optimizaciju logičkog modela baze podataka, nego na parametre za definiranje konfiguracije baze


localhost / 127.0.0.1 / otr
localhost/phpmyadmin/#PMAURL-2:db_sql.php?db=otrs&table=&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82
phpMyAdmin
Current Server:
127.0.0.1 (root)
(Recent tables) ...
otrs
filter items by name
New
article
article_attachment
article_flag
article_innodb
article_plain
article_search
article_sender_type
article_type
auto_response
auto_response_type
customer_company
customer_preferences
customer_user
faq_attachment
faq_category
faq_category_group
faq_history
faq_item
faq_language
faq_log
faq_state
faq_state_type
faq_voting
follow_up_possible
generic_agent_jobs
groups
group_customer_user
group_role
Poslužitelj: 127.0.0.1 » Baza podataka: otrs
Strukturu
SQL
Traži
Upit
Izvoz
Uvoz
Operacije
Privilegije
Rutine
More
Show query box
Vaš SQL upit uspješno je izvršen
SHOW VARIABLES
Izrada profila [Uređivanje] [Izradi PHP kod] [Osvježi]
+ Opcije
Variable_name
Value
auto_increment_increment
1
auto_increment_offset
1
autocommit
ON
automatic_sp_privileges
ON
avoid_temporal_upgrade
OFF
back_log
80
basedir
C:/xampp/mysql
big_tables
OFF
bind_address
*
binlog_cache_size
32768
binlog_checksum
CRC32
binlog_direct_non_transactional_updates
OFF
binlog_error_action
IGNORE_ERROR
binlog_format
STATEMENT
binlog_gtid_simple_recovery
OFF
binlog_max_flush_queue_time
0
binlog_order_commits
ON
binlog_row_image
FULL
binlog_rows_query_log_events
OFF
binlog_stmt_cache_size
32768
binlogging_impossible_mode
IGNORE_ERROR


localhost / 127.0.0.1 / otr
localhost/phpmyadmin/#PMAURL-2:db_sql.php?db=otrs&table=&server=1&target=&token=d20a53afe10397c56647ebd77b50ab82
phpMyAdmin
Current Server:
127.0.0.1 (root)
(Recent tables) ...
otrs
filter items by name
New
article
article_attachment
article_flag
article_innodb
article_plain
article_search
article_sender_type
article_type
auto_response
auto_response_type
customer_company
customer_preferences
customer_user
faq_attachment
faq_category
faq_category_group
faq_history
faq_item
faq_language
faq_log
faq_state
faq_state_type
faq_voting
follow_up_possible
generic_agent_jobs
groups
group_customer_user
group_role
Poslužitelj: 127.0.0.1 » Baza podataka: otrs
Strukturu
SQL
Traži
Upit
Izvoz
Uvoz
Operacije
Privilegije
Rutine
More
Show query box
Vaš SQL upit uspješno je izvršen
SHOW STATUS
Izrada profila [Uređivanje] [Izradi PHP kod] [Osvježi]
+ Opcije
Variable_name
Value
Aborted_clients 0
Aborted_connects 0
Binlog_cache_disk_use 0
Binlog_cache_use 0
Binlog_stmt_cache_disk_use 0
Binlog_stmt_cache_use 0
Bytes_received 204
Bytes_sent 133
Com_admin_commands 0
Com_assign_to_keycache 0
Com_alter_db 0
Com_alter_db_upgrade 0
Com_alter_event 0
Com_alter_function 0
Com_alter_procedure 0
Com_alter_server 0
Com_alter_table 0
Com_alter_tablespace 0
Com_alter_user 0
Com_analyze 0
Com_begin 0

Pregled parametara

- key_buffer_size
  - Definira veličinu međuspremnika za indekse – preporučljivo oko 25% ukupno raspoložive RAM memorije (MyISAM)
  - Prevelika količina može dovesti do usporenja sustava zbog utjecaja na operativni sustav

- query_cache_size
  - veličina memorije za keširanje rezultata upita (podrazumijevano 0)
  - Ne pretjerivati s veličinom zbog internih operacija održavanja međuspremnika (oko 100 MB)

Pregled parametara

- read_buffer_size
  - veličina buffera koji se koristi kod sekvencijalnog skeniranja tablica
  - minimum 8200b, a max 2GB. Preporučljivo podrazumijevano 128 K

- read_rnd_buffer_size
  - ubrzava operacije sortiranja
  - minimum 8200b, a max 2GB. Preporučljivo podrazumijevano 256 K (ne koristiti global)

Pregled parametara

- sort_buffer_size
  - veličina buffera koji se koristi kod sortiranja podataka
  - minimum 32768b, a max 4GB. Preporučljivo podrazumijevano 2M

- max_sort_length
  - početni broj bajtova koji se se stvarno koristi kod sortiranja
  - Povećanje zahtijeva i povećanje prethodnog parametra

Pregled parametara (InnoDB)

- innodb_buffer_pool_instances
  - Broj međuspremnika koje koristi InnoDB (podrazumijevano 1, maksimalno 64)
  - Ima efekta jedino ako se povezani parametar innodb_buffer_pool_size postavi na najmanje 1 GB koje se dijeli između buffera
  - Jedan od najboljih načina za povećanje InnoDB performansi kod sustava s većim brojem procesora (jezgri) i velikom količinom RAM memorije

Pregled parametara (InnoDB)

- innodb_io_capacity
  - Broj I/O operacija koje se mogu izvesti u pozadini u sekundi (minimalno 100, podrazumijevano 200, maksimalno 2**32-1)
  - Pojavom sve bržih i bržih tehnologija diskova povećanjem ovog parametra mogu se značajno povećati ukupne performanse, ali treba paziti da se ne premaše mogućnosti diskova
  - Treba uzeti u obzir da procesi koji ne pripadaju MySQL koriste I/O operacije

Pregled parametara (InnoDB)

- innodb_read_io_threads
  - Broj niti za čitanje koje se mogu izvesti u pozadini u sekundi (minimalno 1, podrazumijevano 4, maksimalno 64)

- innodb_write_io_threads
  - Broj niti za pisanje koje se mogu izvesti u pozadini u sekundi (minimalno 1, podrazumijevano 4, maksimalno 64)

Pregled parametara

- Ponekad je bolje dio parametara izmijeniti privremeno, nego ih imati kao podrazumijevane vrijednosti tijekom cijelog vremena rada servera

- **U pravilu je bolje potrošiti vrijeme na upoznavanje i optimizaciju logičkog modela baze podataka, nego na parametre za definiranje konfiguracije baze**

Optimizacija razvojnog alata i aplikacije (PHP / Laravel)

Optimizacija baze u aplikaciji (Laravel)

- Treba dobro poznavati logički model baze podataka uključujući sve korištene indekse
- Treba poznavati i koristiti eventualno naknadno dodane „trikove” za ubrzanje rada s bazom (tablice za sumiranje ili brojenje)
- Pisati upite koji utječu na minimalno potrebnu količinu podataka potrebnu za normalan rad aplikacije
- Koristiti MySQL za obradu dijela podataka kad je to moguće (npr. sumiranje)
- Po potrebi kreirati i uklanjati dio indeksa u aplikaciji
- Ako je izvedivo, napraviti keširanje dijela podataka u aplikacije umjesto konstatnog slanja upita serveru
- Pratiti stvarni izgled upita i vrijeme izvođenja

Optimizacija upita

DB::enableQueryLog();

```
$pokazateljAll = Usluga::join("POKAZATELJ", "POKAZATELJ.USLUGA_ID", "=", "USLUGA.ID")
```

- ->where('USLUGA.USLUGA_ID', $id) ->where('POKAZATELJ.Status_aktualno_povijesno', Config::STATUS_AKTUALNO) ->unionAll($pokazateljUsluga)
- ->orderBy('Oznaka_pokazatelja', 'asc')
- >select('Status_zadnje_vrijednosti_pokazatelja','Oznaka_pokazatelja','Naziv_pokazatelja')
- ->get();

dd(DB::getQueryLog());

```
array:1 [▼
  0 => array:3 [▼
    "query" => "(select `Status_zadnje_vrijednosti_pokazatelja`, `Oznaka_pokazatelja`,
`Naziv_pokazatelja` from `USLUGA` inner join `POKAZATELJ` on `POKAZATELJ`.`USLUGA_ID`
= `USLUGA`.`ID` where `USLUGA`.`USLUGA_ID` = ? and
`POKAZATELJ`.`Status_aktualno_povijesno` = ?) union all (select
`Status_zadnje_vrijednosti_pokazatelja`, `Oznaka_pokazatelja`, `Naziv_pokazatelja`
from `USLUGA` inner join `POKAZATELJ` on `POKAZATELJ`.`USLUGA_ID` = `USLUGA`.`ID`
where `USLUGA`.`ID` = ? and `POKAZATELJ`.`Status_aktualno_povijesno` = ?) order by
`Oznaka_pokazatelja` asc ◀"
    "bindings" => array:4 [▼
      0 => "2"
      1 => "aktualno"
      2 => "2"
      3 => "aktualno"
    ]
    "time" => 43.88
  ]
]
```

Optimizacija upita

```
array:1 [▼
  0 => array:3 [▼
    "query" => """
      select distinct USLUGA.ID, USLUGA.Oznaka_usluge, USLUGA.Naziv_usluge, \r\n
      \t\t\t\t\t\tSTATUS_USLUGE.Opis, OSOBA.Ime, OSOBA.Prezime\r\n
      \t\t\t\t\tfrom USLUGA\r\n
      \t\t\t\t\t\tinner join STATUS_USLUGE on USLUGA.STATUS_USLUGE_ID = STATUS_USLUGE.ID \r\n
      \t\t\t\t\t\tinner join ROLA_OSOBE_NA_USLUZI on USLUGA.ID = ROLA_OSOBE_NA_USLUZI.USLUGA_ID \r\n
      \t\t\t\t\t\tinner join OSOBA on ROLA_OSOBE_NA_USLUZI.OSOBA_ID = OSOBA.ID \r\n
      \t\t\t\t\t\tleft join TIP_KORISNIKA_USLUGE on USLUGA.ID = TIP_KORISNIKA_USLUGE.USLUGA_ID \r\n
      \t\t\t\t\t\tleft join DJELOKRUG_USLUGE on USLUGA.ID = DJELOKRUG_USLUGE.USLUGA_ID \r\n
      \t\t\t\t\t\tleft join RAZINA_STRUCNOSTI_KORISNIKA_USLUGE on USLUGA.ID =
RAZINA_STRUCNOSTI_KORISNIKA_USLUGE.USLUGA_ID \r\n
      \t\t\t\t\twhere USLUGA.Usluga_ili_element = 'Usluga' and USLUGA.Usluga_ili_element is not null and
ROLA_OSOBE_NA_USLUZI.ROLA_ID = 1 and ROLA_OSOBE_NA_USLUZI.Status_aktualno_povijesno = 'aktualno' order by
USLUGA.Oznaka_usluge ◀
      """
    "bindings" => []
    "time" => 28.12
  ]
]
```

Komentari i pitanja?