

|  |  |
| --- | --- |
|  | UNIVERSITY OF NIŠ FACULTY OF ELECTRONIC ENGINEERING |

Sigurnost MySQL baze podataka

Seminarski rad

|  |  |
| --- | --- |
| Student: | Mentor: |
| Ivan Šušter, br. ind. 1548 | Doc. dr Aleksandar Stanimirović |

Niš, Februar 2024.

SADRŽAJ

[1. UVOD 1](#_Toc164110790)

[2. SIGURNOST BAZE PODATAKA 2](#_Toc164110791)

[3. SIGURNOST U MySQL-u 3](#_Toc164110792)

[3.1. Sigurnosne mere posle instalacije MySQL-a 4](#_Toc164110793)

[3.2. Korišćenje lozinke u MySQL-u 6](#_Toc164110794)

[4. UPRAVLJANJE PRAVIMA PRISTUPA I NALOZIMA U MySQL-u 8](#_Toc164110795)

[4.1. MySQL uloge (Roles) 10](#_Toc164110796)

[4.2. Upravljanje lozinkama 14](#_Toc164110797)

[4.3. Zaključavanje i ograničavanje resursa naloga 16](#_Toc164110798)

[5. DODATNE SMERNICE ZA SIGURNOST BAZE PODATAKA 18](#_Toc164110799)

[7. ZAKLJUČAK 21](#_Toc164110800)

# 1. UVOD

Obezbeđivanje poverljivosti, integriteta i dostupnosti podataka u bazaman podataka zavisi od njihove sigurnosti. Sigurnosne mere su eophodne da bi se sprečio nedozvoljen pristup kao i drugi sigurnosni rizici.

Bezbednost MySQL baze podataka obuhvata niz aktivnosti uključujući odogovarajuću upotrebu lozinki, kontrolu pristupa i privilegija kao i primenu dodatnih bezbednosnih mehanizama. Upotreba lozinke je neophodna za kontrolu pristupa i autentifikaciju. Davanje korisnicima dozvola u skladu sa njihovim odgovornostima je takođe ključan faktor u MySQL-u. Kontrola pristupa ograničava pristup određenim bazama podataka i tabelama. Opasnost od nenamerne promene podataka ili neovlašćenog pristupa može se smanjiti pomoću dozvola pristupa.

Upravljanje lozinkama i bezbednost lozinki, kao što je bezbedno skladištenje i korišćenje lozinke kako bi se sprečio neovlašćen pristup lozinkama takođe predstavlja deo sigurnosti kako MySQL tako i ostalih baza podataka.

Prilikom upravljanja nalozima važno je zaključati naloge koji se ne koriste kao i upravljati resursima naloga, čime je moguće rešiti višestruke pokušaje prijave korišćenjem pogrešne lozinke.

Da bi se dodatno poboljšala bezbednost MySQL baze podataka mogu se uvesti dodatne mere sigurnosti. Ovo obuhvata korišćenje *firewall-a*, kako bi se ograničio pristup serveru na kome se nalazi baza podaaka, redovno ažuriranje sistema za upravljanje bazama podataka, obavljanje analize i procena sigurnosti baza kako bi se otklonili nedostaci.

# 2. SIGURNOST BAZE PODATAKA

Bezbednost odnosno sigurnost baze podataka je ključna komponenata, koja ima za cilj zaštitu baza podataka od namernih i nenamernih pretnji. To uključuje osiguranje poverljivosti, integriteta i dostupnosti podataka uskladištenih u bazama podataka [1]. Pored toga, bezbednost operativnog sistema igra ključnu ulogu u zaštiti integriteta i poverljivosti baza podataka u Windows i Linux okruženjima. Windows i Linux operativni sistemi predstavljalju osnovu na kojoj baze podataka rade, tako da svaka ranjivost i probijanje operativnog sistema može direktno uticati na bezbednost baza podataka.

Kako se povećava obim podataka, tako se povećavaju i opasnosti koje dolaze sa njima. Kritične informacije koje se skladište u bazama podataka suočavaju se sa ozbiljnim izazovima za njihovu bezbednost, integritet i dostupnost usled zlonamernih napada, ilegalnog pristupa i manipulacije podacima.

Obezbeđivanje privatnosti podataka u bazama podataka je od bitnog zanačaja za sprečavanje neovlašćenog pristupa. Pored toga, integritet upita je od suštinskog zanačaja za održavanje bezbednosti baze podataka, jer svaki kompromis u integritetu upita može dovesti do manipulacije podacima.

Šifrovanje ili enkripcija takođe značajno utiče na bezbednost baze podataka, iako šifrovanje može da poboljša bezbednost podataka, veoma je bitno obratiti pažnju i na performanse baze podataka kada se primenjuje strategija šifrovanja podataka. Efaksnim odabirom podataka koje je potrebno šifrovati se mogu zaštititi osetljivi podaci, bez prevelikog uticaja na efikasnost baze podataka.

Bezbednosni problemi u bazama podataka obuhvataju širok niz problema koji se moraju rešiti da bi se obezbedila sveobuhvatna zaštita. Faktori kao što su kontrola pristupa, mehanizmi autentifikacije i šifrovanje podataka doprinose jačanju ukupne bezbednosti podataka [2]. Pored toga potrebno je rešiti i probleme pravljenja rezervnih kopija podataka, kao i obratiti pažnju na zaštitu fizičkih servera na kojima se nalaze baze podataka od mrežnih i drugih napada.

Obezbeđivanje baza podataka putem mrežne bezbednosti uključuje pristup koji se bavi raličitim kritičnim aspektivam kako bi se osigurala zaštita osetljivih podataka. Jedan od glavnih aspekata je upotreba uskladištenih procedura, za poboljšanje bezbednosti baze podataka [3]. One igraju ključnu ulogu u zaštiti baza podataka od pretnji poput *SQL injection*, čime se poboljšava ukupna bezbednost.

Pored toga mere mrežne bezbednosti kao što je *firewall,* kontrola pristupa i filtriranje zahteva veb servera igraju značajnu ulogu u obezbeđivanju sigurnosti baza podataka. Ove osnovne bezbednosne mere na nivou mreže su suštinske komponente bezbednosti baze podataka, dorinoseći ukupnoj zaštiti baze podataka.

# 3. SIGURNOST U MySQL-u

Kada se koristi MySQL na serveru, neophodno je primeniti bezbednosne mere kako bi se sprečile uobičajene greške. Bezbednosne mere ne bi trebalo da se fokusiraju samo na zaštitu MySQL servera, već i na zaštitu celokupnog servera od raznih vrsta napada.

MySQL uključuje bezbednosne mere koje koriste liste za kontrolu pristupa (engl. *Access Control Lists - ACL*) za korisničke operacije i podržava SSL šifrovane veze između klijenta i servera. Ovi bezbedonosni principi nisu isključivi za MySQL i relavantni su za većinu aplikacija.

Da biste osigurali bezbedno korišćenje MySQL-a, ključno je ograničiti pristup tabeli *user* u bazi podataka sistema isključivno na MySQL *root* nalog [4]. Razumevanje sistema privilegija pristupa u MySQL-u i korišćenje komandi kao što su GRANTi REVOKEza upravljanje pristupom je od velikog zančaja. Pored toga, preporučljivo je i čuvanje *hash*-irane lozinke umesto običnog teksta u bazi poadataka kako bi se sprečilo narušavanje privatnosti podataka u slučaju napada na bazu podataka. Redovna provera i prilagođavanje privilegija korisnika korišćenjem komandi kao što su SHOWGRANTza proveru pristupa, kao i REVOKE za uklanjanje prava pristupa se savetuje radi održavanja bezbednog sistema.

Dalje, važno je kreirati jake lozinke sa kombinacijom velikih i malih slova, brojeva kao i specijalnih karaktera, kako bi se odvratili pokušaji *brute force* i drugih napada. Lozinke takođe treba da budu dugačke, nepredvidive i potrebno je izbegavati komponente lozinke koje je lako pogoditi kao što su uobičajene reči ili fraze, takođe se preporučuje korišćenje *salt*-a koji se dodaje šifrovanoj lozinci, kako bi se obezbedila dodatna sigurnost. Korišćenje *firewall*-a kao i podešavanje MySQL-a tako da nije direktno dostupan neopouzdanim hostovima su dodatne mere bezbednosti [5].

Aplikacije koje komuniciraju sa MySQL-om treba da sadrže odogvarajuće tehnike kako bi osigurale bezbednost baze podataka, a to se najviše odnosi na prenošenje neenkriptovanih podataka preko Interneta. Takođe se preporučuje korišćenje šifrovanih protokola kao što su SSL i SSH za komunikaciju radi poboljšanja bezbednosti podataka.

Alat koji se obično koristi za skeniranje portova je **nmap** i ovaj alat je neophodan kako bi se procenila pristupačnost određenim portovima na hostu. Nmap je posebno široko priznat i poplaran alat za skeniranje portova. Skeniranjem portova, pruža vredne informacije za procenu i konfiguraciju mreže kako bi bila sigurna od napada.

Pored toga MySQL preporučuje za brzu proveru dostupnosti porta i **telnet** kojim je moguće pokušati povezivanje preko IP adrese hosta i porta u ovom slučaju 3306.

U nastavku rada biće data detaljnija objašnjenja sa primerima u MySQL sistemu za upravljanje bazama podataka, navedenih smernica koje su od bitnog značaja za očuvanje sigurnosti baze podataka.

## 3.1. Sigurnosne mere posle instalacije MySQL-a

Nakon instaliranja MySQL-a, postoji nekoliko važnih koraka koje treba izvršiti kako bi se osiguralo pravilno funkcionisanje i sigurnost sistema baze podataka. Jedan od ključnih koraka je inicijalizacija direktorijuma podataka i kreiranje MySQL GRANT tabela. Ovaj proces je od suštinskog zanačaja za postavljanje neophodnih struktura unutar MySQL-a za upravljanje privilegijama korisnika i kontrolu pristupa.

U nekim slučajevima, incijalizacija direktorijuma podataka može biti automatizovana u zavisnosti od načina instalacije MySQL-a. Prilikom instalacije MySQL-a u *Windows* okruženjima, koristeći *Window Installer* kao i kod *Linux* okruženja, korišćenjem sistem paketa, obično se automatski inicijalizuje direktorijum podataka.

Međutim, za određene platforme i tipove instalacije je potreban ručna inicijalizacija direktorijuma podataka. Ovo uključuje instalacije iz binarnih i izvornih distribucija na *Unix* sistemima i iz *ZIP* arhiva na *Windowsu*. U ovom radu neće biti prikazani ovi primeri kako ne bi izašli iz okvira teme i zbog jednostavnosti primera, koristiće se automatska instalacija MySQL servera na Ubuntu Server 22.04 distribuciji.

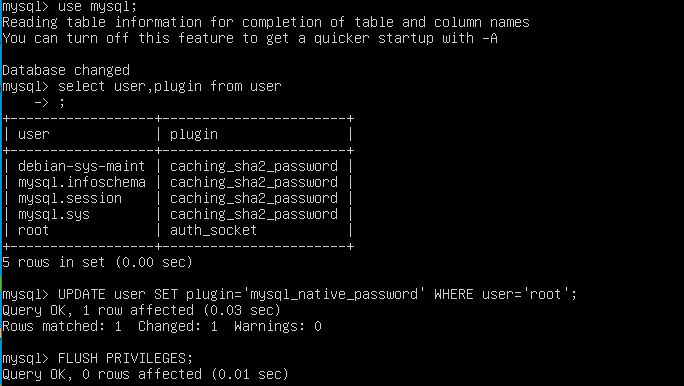
Kada se direktorijum podataka inicijalizuje, sledeći ključni korak je da se pokrene MySQL server i provera da li je moguće pristupiti serveru. Ovo je neophodno da bi se osiguralo da server baze podataka ispravno radi i da može da odgovori na zahteve klijenata. Poredt toga, važno je dodeliti lozinku *root* korisniku, ako to već nije urađeno u procesu inicijalizacije direktorijuma podataka. Postavljanje lozinke za *root* korisnika je osnovna sigurnosna mera koja pomaže u sprečavanju neovlašćenog pristupa MySQL serveru i štiti integritet sistema baze podataka.

Ukoliko je potrebno startovati server može se koristit komanda sudo systemctl start mysql.

Nakon toga sledi dodeljivanje lozinke *root* korisniku, da bi se to izvršilo potrebno je konektovati se na MySQL server. Korišćenjem sudo mysql komande, zatim ono što je karakterištično za Ubuntu distribuciju je da MySQL koristi *auth\_socket* plugin automatski, što znači da će korisnik biti autentifikovan pomoću sistemskih kredencijala korisnika.

Da bi omogućili prijavljivanje na MySQL server pomoću lozinke koja će biti kasnije pokazana potrebno je da odradimo neku od sledećih opcija.

1. Prva opcija jeste ažuriranje plugin-a u tabeli *user,* tako što se kolona *plugin* postavi na novu vrednost *mysql\_native\_password* ili *chaching\_sha2\_password* za *root* korisnika kao što je prikazano na slici ispod.



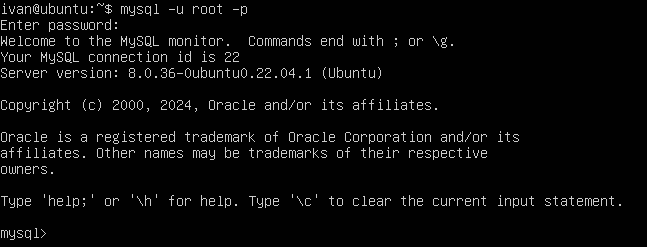
Slika 1 Promena plugina kod korisnika

Sada je potrebno promeniti lozinku korisnika pomoću ALTER USER komande.



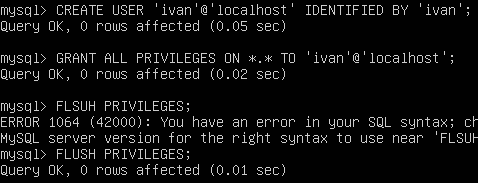
Slika 2 Promena lozinke root korisniku

I nakon toga je moguće prijaviti se na server pomoću komande mysql -u root -p.



Slika 3 Prijavljivanje na MySQL sistem kao root korisnik

1. Druga opcija je kreiranje novog korisnika koji će imati isto korisničko ime kao i sistemski korisnik i istu lozinku kao sistemski korisnik, a koji će koristiti *chaching\_sha2\_password* plugin po defaultu. Ovom korisniku je takođe potrebno dodeliti sve privilegije, kako bi imao iste mogućnosti kao *root* korisnik.

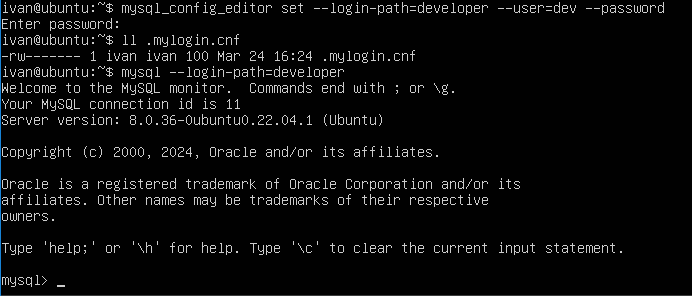


Slika 4 Kreiranje korisnika sa istim kredencijalima kao sistemski

Pored ovih opcija može se i specificirati fajl sa kredencijalima odnosno lozinkom u *home* direktorijumu korisnika, što će biti prikazano u nastavku.

## 3.2. Korišćenje lozinke u MySQL-u

Korsnici MySQL-a bi trebalo da se pridržavaju određenih uputstava kako bi održali bezbednost svojih lozinki. Kada se povezuju na MySQL server, ključno je izbeći izlaganje lozinki drugim korisnicima. Jedan od preporučenih metoda je upotreba programa mysql\_config\_editor za bezbedno skladištenje kredencijala za autentifikaciju u šifrovanoj datoteci. Drugi pristup je da prilikom prijave koriste komandu koja im dozvoljava interaktivni unos lozinke.



Slika 5 Kreiranje šifrovanog fajla za prijavljivanje na MySQL pomću mysql\_config\_editor-a

Opcije koje se koriste prilikom korišćenja mysql\_config\_editor -a su sledeće:

* --login-path - predstavlja oznaku za kredencijale odnosno labelu (ukoliko imamo više kredencija) ako se postavi vrednost *client* moguće je prijavljivanje na MySQL unosom komande mysql, za ostale mora da se navede prilikom prijave,
* --user - predstavlja korisničko ime,
* --password - opcija koja omogućava mysql\_config\_editor-u da zatraži interaktivni unos lozinke za korisnika, tako da se ne beleži u istoriji komandne linije,
* --host - naziv hosta ili IP adresa gde se nalazi MySQL (ako je na udaljenom serveru),
* --port - broj porta na kome radi MySQL.

Korišćenje opcije --password ili -p u komandnoj liniji je zgodno ali nije sigurno jer može izložiti lozinku drugim programima ukoliko se navodi sa specificiranom lozinkom.

Sigurnijij metod je interaktivno unošenje lozinke kada se unosi samo opcija bez lozinke, a zatim se u sledećoj liniji zahteva unos lozinke koja je skrivena. Ova metoda je već prikazana u prehodnom delu kod kreiranja korisnika.

Takođe se preporučuje čuvanje lozinke u datoteci nad kojoj je potrebno ograničiti pristup odnosno promeniti prava pristupa. Na primer moguće je čuvanje lozinke u fajlu koji se nalazi u *home* direktorijumu, .my.cnf fajl treba da sadrži [client] sekciju.



Slika 6 Kreiranje fajla u home direktorijumu trenutnog korisnika



Slika 7 Priaz privilegija i sadržaja kreiranog fajla za prijavu

Prijava na sistem pomoću ovog fajla se takođe vrši samo komandom mysql gde se automatski čitaju kredencijali iz fajla. U ovom fajlu kredencijali nisu šifrovani odnosno mogu se pročitati kao što se vidi sa slike 7, zbog toga je potrebno ograniči prava pristupa fajlu kao jednu od mera kako bi se osigurala bezbednost informacija.

Ukoliko *home* direktorijum nije podrazumevana putanja za korišćenje fajla, moguće je promeniti korišćenjem komande mysql --defaults-file=/home/user/.my.cnf, takođe moguće je logovanje bilo kojim korisničkim imenom i lozinkom u bilo kom trenutku kao u prethodnim primerima.

Pored ovih praksi potrebno je i osigurati fajl na Linux sistemima, ograničavanjem pristupa fajlu, gde MySQL vodi evidenciju o izvršenim komandama kao što su CREATE USER i ALTER USER, gde se lozinka unosi kao običan tekst, što je vidljivo u fajlu .mysql\_history koji se nalazi u *home* direktorijumu. Obratiti pažnju da od verzije 8.0 MySQL-a, ova mogućnost nije dostupna odnosno MySQL je automatski isključio beleženje komandi koje sadrže lozinku kao što je CREATE USER i ALTER USER.

Takođe administratori nikad ne bi trebalo da daju privilegije za pristup sistemskoj tabeli *user,* gde se nalaze i sačuvane lozinke korisnika. Kako bi administratori osigurali da korisnici koriste jake lozinke može se koristiti dodatak validate\_password.

Kao što je već pomenuto ranije, moguće je da log fajlovi (ukoliko se koristi starija verzija MySQL-a < 8.0) upisuju lozinku koja je uneta pomoću nekih komandi kao tekst, što predstavlja opasnost ukoliko neko pročita log datoteku ukoliko ima pristup. Zato je potrebno izbegavati upisivanje lozinke kao običan tekst u sledećim komandama: CREATE USER, ALTER USER, SET PASSWORD, pored toga i pri unosu podešavanja prilikom replikacije servera u komandama za startovanje replika START SLAVE... PASSWORD. Da bi se sprečio pristup log fajlovima potrebno je ograničiti prava pristupa istima.

Lozinke iz ovih komandi se mogu videti u binarnim log fajlovima, kao i u ostalim log datotekama gde se čuvaju komande koje su izvršene.

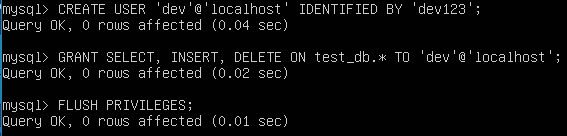
Pozeljno je izbegavati korišćenje lozinki u komandama INSERT i UPDATE za sistemsku tabelu *user* da bi se sprečilo evidentiranje lozinke kao teksta.

Iz ovoga se može zaključiti da je potrebno obratit posebno pažnju na način prijavljivanja na server, kao i kontrola prstupa određenim resursima na serveru kako bi se izbeglo otkrivanje osetljivih informacija.

# 4. UPRAVLJANJE PRAVIMA PRISTUPA I NALOZIMA U MySQL-u

MySQL sistem prava pristupa igra ključnu ulogu u upravljanju korisničkim pristupom serveru i podacima. Kreiranjem nalog, MySQL omogućava korisnicima klijenta da se povežu sa serverom i obavljaju različite operacije nad bazama podataka koje se nalaze na tom serveru. Osnovna funkcija MySQL sistema prava pristupa je autentifikacija korisnika na osnovu njegovog *host-a* i povezivanje sa određenim privilegijama nad bazama podataka, kao što su SELECT, INSERT, UPDATE i DELETE.

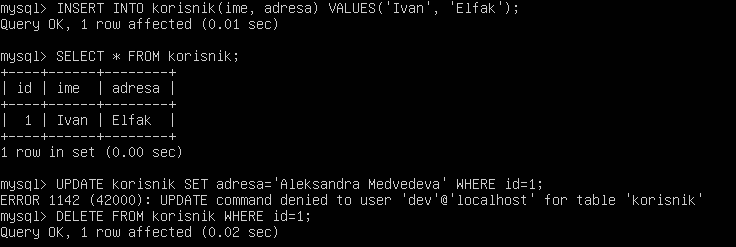
Da bi kontrolisali pristup i odredili koji korisnici mogu da se povežu sa serverom, svakom nalogu su dodeljeni autentifikacioni podaci, obično u obliku lozinke. Komande za upravljanje MySQL nalozim uključuju SQL naredbe poput CREATE USER, GRANT i REVOKE, koje omogućavaju administratorima da efikasno definišu i menjaju korisničke privilegije. MySQL sistem privilegija osigurava da korisnici mogu da izvršavaju samo operacije za koje imaju eksplicitno dodeljene dozvole, održavajući sigurnost i integritet podataka unutra baze podataka.



Slika 8 Kreiranje korisnika i dodeljivanje privilegija za čitanje, dodavanje i brisanje u test\_db bazi nad svim tabelama

Na slici 8 je dat primer kreiranja korisnika dev kome su zatim dodeljena prava čitanja, dodavanja i brisanja u test\_db bazi podataka nad svim tabelama. Komanda FLUSH PRIVILEGES koja je korišćenja na kraju služi kako bi se ponovo učitale tabele u memoriju i preporučuje se ukoliko se vrši direktna izmena u sistemskim tabelam pomoću UPDATE komande.

Na sledećoj slici prikazani su primeri izvršavanje komandi kao dev korisnik kako bi se izvršila provera dodeljenih privilegija korisniku.



Slika 9 Primer komandi kao dev korisnik gde se vidi greška prilikom pokušaja UPDATE-a

Kada korisnik uspostavi vezu sa MySQL serverom, njegov indentitet se utvrđuje na osnovu *host-a* sa kojeg se povezuje i navedenog korisničkog imena i lozinke. Naknadni zahtevi koje korisnik uputi procenjuju se od strane dozvola odnosno privilegija u skladu sa njihovim identitetom i zahtevanim radnjama. MySQL uzima u obzir kako ime *host-a* tako i korisničko ime kako bi jedinstveno identifikovao korisnike, prepoznajući da isto korisničko ime može predstavljati različite korisnike koji se povezuju sa različitih *host-ova.* Ova fleksibilnost omogućava administratorima da dodele različite privilegije korisnicima na osnovu njihove konekcije, obezbeđujući time dodatnu kontrolu nad dozvolama za pristup.

Da bi se pregledale privilegije koje su povezane sa određenim nalogom, može se koristit naredba SHOW GRANTS; koja je data u nastavku na slici 10, kako bi se ove informacije dobile iz tabela dozvola koje su smeštene u *mysql* sistemsku bazu podataka. Ove tabele dozvola sadrže informacije o privilegijama koje MySQL server čita u memoriju prilikom pokretanja, koristeći ih ua donošenje odluka o kontroli privilegija tokom interakcije korisnika sa serverom.

Takođe na slici ispod prikazan je i primer REVOKE za oduzimanje odnosno uklanjanje prava korisnika.



Slika 10 Komanda za prikaz privilegija korisnika i komanda za oduzimanje privilegija

MySQL privilegije takođe uključuju dve ključne faze kada se korisnik povezje sa serverom. U prvoj fazi, provera identiteta i kredencijala:

* Server proverava kredencijale korisnika, kao što je lozinka i ime da bi potvrdio identitet.
* Ispituje stanje zaključavanja naloga kako bi utvrdio da li je nalog zaključan ili otključan.
* Ukoliko u prehodna dva koraka postoji problem, server odbija povezivanje, u suprotom prelazi na drugu fazu.

Druga faza je provera autentičnosti korisnika i kontrola pristupa:

* Server potvrđuje identitet korisnika i kredencijale koristeći informacije uskladištene u *user* sistemskoj tabeli.
* Uslovi za prihvatanje veze uključuju podudaranje imena hosta (ili IP adrese) i korisničkog imena, sa odgovarajućim kolonama u *user* tabeli.
* Status zaključavanja naloga je označen kolonom *account\_locked,* pri čemu N predstavlja otključan nalog.
* Identitet korisnika zasnovan je na njegovom MySQL korisničkom imenu i imenom hosta sa kog se povezuje.
* Server šifruje i upoređuje lozinke tokom procesa autentifikacije.
* Podudaranje u *user* tabeli se vrši sortiranjem na osnovu vrednosti hosta i korisničkih imena.
* Po pravilu prioritet imaju konkretne IP adrese hostova i imena hostova sa ’%’ što predstavlja bilo koji host.

Iako MySQL sistem pruža snažnu kontrolu nad privilegijama korisnika, postoje određena ograničenja na koja treba obratiti pažnju. Administrator ne može eksplicitno zabraniti pristup određenom korisniku niti specificirati dozvole popout dozvole za manipulaciju tabelama bez prava kreiranja baza podataka.

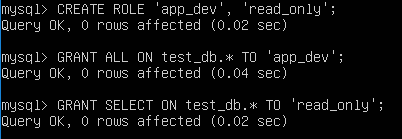
Pored privilegija rada sa bazama podataka administratori mogu dodeliti i druga prava, kao što su na primer prava za administraciju servera: CREATE\_ROLE, DROP\_ROLE, CREATE\_USER, DROP\_USER, itd. Pored ovih postoje i privilegije za kreiranje *view-a, index-a* itd. Sve ove privilegije se dodeljuju pomoću naredbe GRANT i oduzimaju pomoću naredbe REVOKE.

## 4.1. MySQL uloge (Roles)

Uloge (engl. *Roles*) u MySQL-u takođe predstavljaju jedan oblik upravljanja privilegijama unutar sistema baza podataka. MySQL uloga je suštinski nazvana kolekcijom privilegija koje mogu biti dodeljene ili oduzete, slično korisničkim nalozima. Dodeljivanjem uloga korisničkim nalozima, privilegije koje su povezane sa svakom ulogom mogu biti nasleđene. Time se pojednostavljuje proces upravljanja privilegijama i pruža se efikasniji način za upravljanje kontrolom pristupa.

Upravljanje ulogama u MySQL-u obuhvata nekoliko ključnih naredbi:

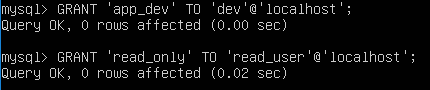
CREATE ROLE i DROP ROLE se koriste za kreianje i uklanjanje uloga unutra sistema baze podataka.



Slika 11 Primer kreiranja role app\_dev i read\_only kao i dodeljivanje privilegija tim rolama

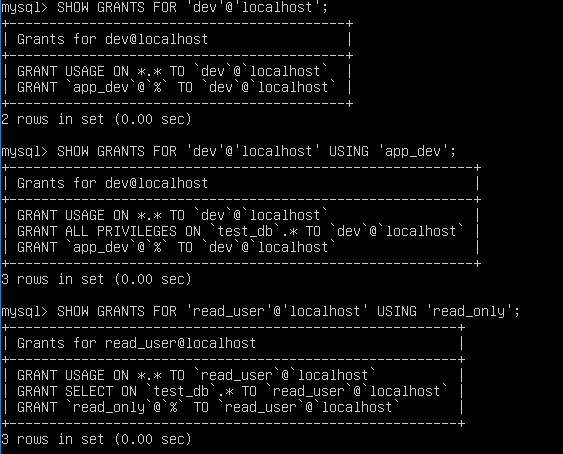
U primeru na slici 11 kreirane su dve uloge (engl. *role*) kao i dodeljivanje svih privielgija *app\_dev* ulozi u bazi test\_db nad svim tabelama i dodeljivanje SELECT naredbe, *read\_only* ulozi u bazi test\_db nad svim tabelama.

GRANT i REVOKE su ključne za dodeljivanje privilegija kako korisnicima tako i ulogama kao i za oduzimanje privilegija. Na slici ispod prikazano je dodeljivanje uloga korisnicima. Odnosno *app\_dev* uloga je dodeljena *dev* korisniku, dok je *read\_only* uloga dodeljena *read\_user-u.*



Slika 12 Dodeljivanje uloga korisnicima

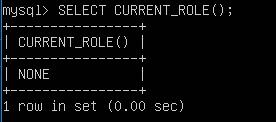
SHOW GRANTS se koristi za prikazivanje dodeljenih privilegija i uloga kako korisnicima tako i ulogama. Na slici 13 može se videti da je dodeljivanjem uloge, korisnik nasledio privilegije te uloge što i jeste cilj uloga.



Slika 13 Prikaz prava koja su dodeljena korisnicima

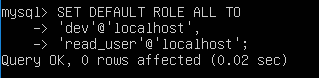
Prilikom kreiranja uloga i dodeljivanja privilegija u MySQL-u važno je razmotriti specifične zahteve aplikacije. Da bi se izbeglo ručno dodeljivanje privilegija pojedinačnim korisnicima, uloge mogu biti kreirane i korišćene za dodelu privilegija i time pojednostaviti proces upravljanja privilegijama.

Nakon kreiranja i dodeljivanja uloga, potrebno je aktivirati uloge zato što one mogu biti aktivne ili neaktivne u toku sesije. Ako su kreirane i dodeljene uloge aktivne u sesiji onda se privilegije primenjuju, u suprotnom one ne važe za korisnike. Da bi proverili da li su uloge aktivne koristi se CURRENT\_ROLE() funkcija. U nastavku na slici kao *read\_user* pomoću funkcije proverava da li su uloge aktivne.



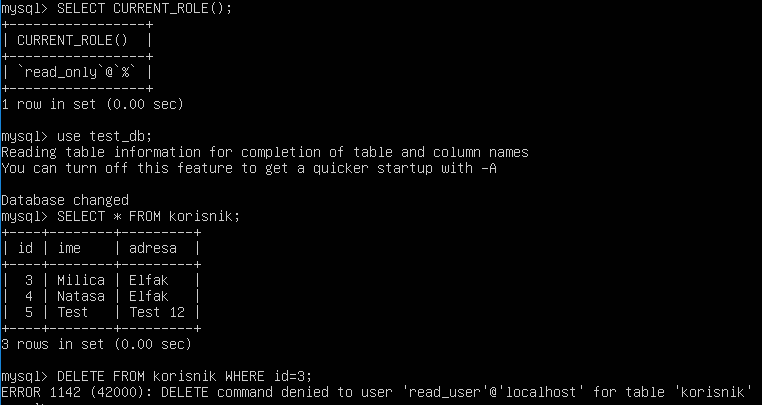
Slika 14 Prikaz statusa uloge za korisnika read\_user

S obzirom da uloge nisu aktivne potrebno je aktivirati ih korišćenjem naredbe SET DEFAULT ROLE kao na slici ispod.



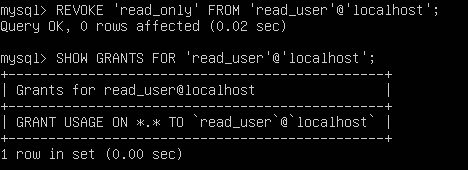
Slika 15 Aktiviranje uloga za korisnike

Kada se opet prijavi *read\_user* i iskoristi funkciju CURRENT\_ROLE() videće da je uloga aktivna i moći će da pristupi bazi odnosno da procita podatke iz baze, što se takođe može videti na slici ispod.



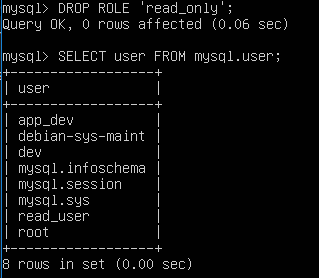
Slika 16 Provera statusa uloge za korisnika read\_user i testiranje komande

Uklanjanje uloge nekom korisniku može se uraditi primenom naredbe REVOKE i nakon toga može se izvršiti provera gde se vidi da je uloga uklonjena korisniku *read\_user*.



Slika 17 Uklanjanje uloge korisniku read\_user

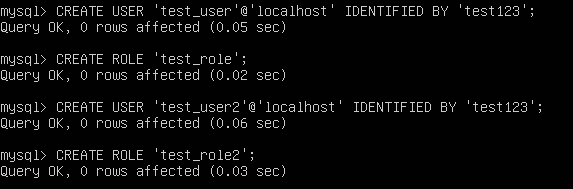
Pored uklanjanja uloge korisnicima REVOKE se može koristiti i za uklanjanje privilegija ulogama. Kasnije će biti reči o tome kako se uloge razlikuju od korisnika, s obzirom da se može primetiti da su komande iste kao kod dodeljivanja i uklanjanja privilegija kod korisnika. Brisanje uloge se obavlja komandom DROP kao što je prikazano na slici ispod.



Slika 18 Primer brisanja uloge

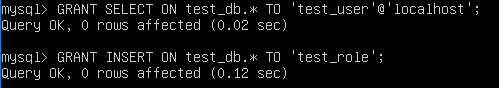
Uloge su različite od korisničkih naloga po tome što su zaključane, nemaju lozinku i dodeljena im je podrazumevani *plugin* za autentifikaciju prilikom kreiranja. Ove karakteristike kasnije mogu biti izmenjene koristeći ALTER USER naredbu od strane korisnika sa neophodnim privilegijama.

S obzirom da su uloge i korisnici slični u MySQL-u je moguće dodeliti korisnika drugom korisniku čime on nasleđuje uloge prvog korisnika, isto je moguće i sa ulogama odnosno moguće je dodeliti korisnika ulozi, primer je dat u nastavku na slici.



Slika 19 Kreiranje test\_user, test\_role, test\_user2, test\_role2 za primer nasleđivanja

Nakon kreiranja uloga i korisnika, korisniku test\_user i ulozi test\_role biće dodeljene privilegije kao na slici 10.



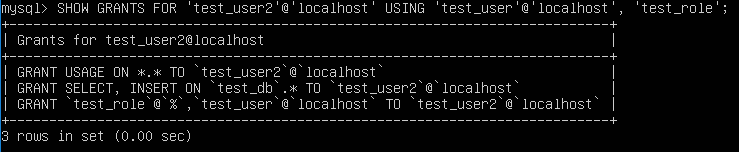
Slika 20 Dodeljivanje privilegija korisniku i ulozi

Nakon toga korisniku test\_user2 biće dodeljen korisnik test\_user kao i uloga test\_role, time test\_user2 dobija privilegije i korisnika i uloge.



Slika 21 Dodeljivanje test\_user i test\_role korisniku test\_user2

Nakon toga, korišćenjem komande SHOW GRANTS za korisnika test\_user2 može se videti da je nasledio sve privilegije korisnika test\_user i uloge test\_role (Slika 22).



Slika 22 Prikaz privilegija za korisnika test\_user2

## 4.2. Upravljanje lozinkama

MySQL pruža skup mogućnosti upravljanja lozinkama kako bi poboljšao sigurnost i kontrolu nad korisničkim nalozima unutra sistema baze podataka. Ove mogućnosti uključuju traženje od korsnika da unese novu lozinku, ograničenja ponovne upotrebe lozinke, verifikaciju lozinke, procenu snage lozinke, generesianje slučajnih lozinki i privremeno zaključavanje naloga.

Funkcionalnost isteka lozinke u MySQL-u omogućava administratorima da primoraju korisnike na periodične promene lozinke za naloge koje koriste. Ovo se može uraditi korišćenjem ALTER USER naredbr ili automatski, globalno korišćenjem sistemskih promenljivih *default\_password\_lifetime.* Potrebno je odrediti starost lozinke ondnosno broj dana koliko lozinka važi, pri čemu sistem označava lozinke kao istekle kada istekne taj broj dana.



Slika 23 Primer izmene trajanja lozinke 90 dana

MySQL takođe podržava ograničenja ponovne upotrebe lozinke kako bi sprečio korisnike da biraju stare lozinke. Ovo može biti zasnovano na broju promena lozinke ili vremenu proteklog od poslednjeg korišćenja lozinke (Slike 24 i 25).



Slika 24 Ograničenje korišćenja istih lozinki



Slika 25 Vreme pre ponovnog korišćenja iste lozinke

Takođe je moguće podesiti globalnu promenljivu koja omogućava da se od korisnika zatraži stara lozinka prilikom promene lozinke, ovaj mehanizam omogućava zaštitu od neovlašćenog menjanja lozinke nekog korisnika (Slika 26).



Slika 26 Postavljanje globalne promenljive za zahtevanje trenutne lozinke ukoliko se menja

U nastavku biće testirano podešavanje sa slike 26. Korišćenjem klauzule PASSWORD EXPIRE u komandi ALTER USER, primorava se korisnik da promeni lozinku nakon logovanja, što se vidi na slici 27. gde sistem ne dozvoljava korišćenje SELECT naredbe.



Slika 27 Greška prilikom izvršvanja komande zbog istekle lozinke

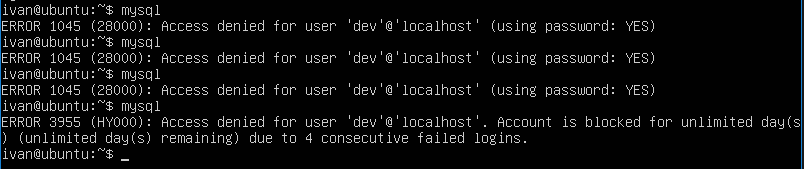
Pored toga, MySQL ima mogućnost privremenog zaključavanja naloga nakon previše uzastopnih neuspelih pokušaja prijave. Ova funkcija pomaže u sprečavanju napada kao što je *brute force* napad, blokirajući naloge na određeni vremenski period. Broj neuspelih pokušaja se može konfigurisati koristeći komandu kao na slici ispod.



Slika 28 Ograničenje broja pokušaja prijavljivanja pre zaključavanja naloga na neodređeno vreme

Klauzula za ograničavanje pokušaja prijave je moguće i prilikom korišćenja komande za kreiranje korisnika, kao i sve ostale prikazane komande u ovom poglavlju.

Praćenje neuspelih pokušaja prijave i privremenog zaključavanje naloga imaju specifične karakteristike u MySQL-u. Obe opcije moraju biti različite od nule da bi bile omogućene, uspešne prijave resetuju broj neuspelih pokušaja. Primer neuspelih pokušaja prijavljivanja dat je na slici ispod.



Slika 29 Neuspešna prijava nakon podešavanja broja neuspelih pokušaja prijave

Da bi korisnik ponovo mogao da koristi nalog potrebno je da administrator baze podataka otključa nalog korisnika, primer otključavanja naloga biće dat u narednom delu.

## 4.3. Zaključavanje i ograničavanje resursa naloga

U MySQL-u, funkcionalnost zaključavanja i otključavanja korisničkih naloga olakšana je korišćenjem ACCOUNT LOCK i ACCOUNT UNLOCK klauzula unutar CREATE USER i ALTER USER nardbe. Ove klauzule imaju različite svrhe kada se koriste pri kreiranju novih naloga ili modifikaciji postojećih.

Kada se koriste zajedno sa CREATE USER naredbom, ACCOUNT LOCK i ACCOUNT UNLOCK klauzule definišu početni status zaključavanja novog korisničkog naloga. U situacijama kada ove klauzule nisu eksplicitno navedene, nalog se podrazumevano generiše u otključanom stanju. Važno je napomenuti da je, ako je komponenta *validate\_password* aktivna, kreiranje naloga bez lozinke zabranjeno, bez obzira na to da li je nalog zaključan ili ne.



Slika 30 Kreiranje korisnika sa zaključanim nalogom

Kada se koristi u ALTER USER naredbi, ove klauzule igraju ključnu ulogu u promeni statusa zaključavanja već postojećeg naloga. U slučajevima kada ove klauzule nisu navedene, trenutno stanje zaključavanja naloga ostaje nepromenjeno. Važno je napomenuti da, korišćenje ALTER USER... UNLOCK ima sposobnost otključavanja bilo kog naloga koji je privremeno zaključan zbog prekoračenja broja neuspelih pokoušaja prijavljivanja.



Slika 31 Otključavanje naloga korisnika

Status zaključanog stanja naloga čuva se u koloni *account\_locked* unutar *mysql.user* sistemske tabele. Ove informacije mogu se dobiti pomoću naredbe SHOW CREATE USER, koja pruža uvid da li je nalog trenutno zaključan ili otključan.

U situacijama kada korisnik pokušava da se poveže sa zaključanim nalogom, pokušaj povezivanja nije uspešan. Server beleži ovaj događaj povećavanjem statusne promenljive *locked\_connects*, koja označava broj neuspelih pokušaja povezivanja na zaključane naloge. Korisniku se prikazuje greška i odgovorajuća poruka se ispisuje.

Važno je napomenuti da zaključavanje naloga ne sprečava koršćenje ili izvršavanje uskladišteni procedura i pogleda (view) povezanih sa zaključanim nalogom.

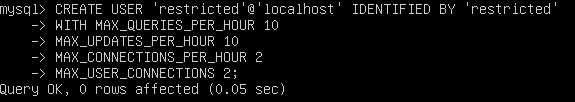
Sredstva za ograničavanje korišćenja resursa MySQL servera mogu se postići postavljanjem globalne promenljive *max\_user\_connection* na vrednost različitu od nule. Ovo podešavanje ograničava broj istovremenih veza koje može uspostaviti bilo koji nalog, ali ne nameće ograničenja na radnje koje korisnik može da izvrši nakon povezivanja. Takođe ovaj metod ne omogućava individualno upravljanje nalozima.

Da bi se obezbedila detaljnija kontrola nad korišćenjem resursa servera od strane nekog naloga, MySQL nudi mogućnost postavljanja ograničenja na različite aspekte korišćenja resursa za svaki nalog. Ova ograničenja uključuju:

* Broj upita koje nalog može izvršiti u toku jednog sata.
* Broj ažuriranja koje nalog može izvršiti u toku jednog sata.
* Učestalost kojom nalog može da se poveže sa serverom u toku jednog sata.
* Maksimalni broj istovremenih veza dozvoljen za jedan nalog.

Važno je napomenuti da se svaka naredba izdata od strane korisnika računa kao ograničenja upita, dok se samo naredbe koje uključuju modifikacije baze podataka ili tabela računaju kao ograničenja ažuriranja. Ograničenja se uvek odnose na nalog bez obzira sa kog hosta se povezuje.

Da bi se postavila ograničenja resursa na nalog prilikom njegovog kreiranja, koristi se CREATE USER naredba. Da bi se postojeća ograničenja naloga izmenila koristi se ALTER USER naredba, koja uključuje WITH klauzulu koja specificira ograničenja resursa. Podrazumevana vrednost za svako ograničenje je nula, što znači da nema ograničenja.



Slika 32 Primer kreiranja korisnika sa ograničenjima upita, ažuriranja, povezivanja i istovremenih konekcija

WITH klauzula u ALTER USER naredbi može uključivati bilo koju kombinaciju ograničcenja, pri čemu svako ograničenje predstavlja brojčanu vrednost po satu. Za ograničenje MAX\_USER\_CONNECTIONS, postavljanje na nulu odnosi se na globalnu vrednost *max\_user\_connections* sistemske promenljive. Ako su oba ograničenja nula, nema ograničenja na nalogu.

Server prati korišćenje resursa na osnovu naloga, ne po klijentu. Ako nalog dosigne svoje ograničenje na veze, upite ili ažuriranja u toku jednog sata, server odbija dalje radnje sve dok ne prođe jedan sat, prikazjući odgovarajuće greške.

Brojači korišćenja resursa mogu se resetovati globalno za sve naloge korišćenjem FLUSH USER\_RESOURCES naredbe ili pojedinačno za određeni nalog ponovnim definisanjem njegovih ograničenja. Ovi reseti ne utiču na ograničenje MAX\_USER\_CONNECTIONS i svi brojači se resetuju na nulu prilikom pokretanja servera bez prenošenja vrednosti.

# 5. DODATNE SMERNICE ZA SIGURNOST BAZE PODATAKA

Zaštita MySQL baze podataka iz ugla mrežne sigurnosti kao i sigurnost aplikacija je od velikog značaja kako bi se sprečio neovlašćeni pristup bazi podataka. Zaštita od *SQL injection* napada je ključna kako bi se očuvao integritet podataka. Da bi se smanjio rizik od *SQL injection* napda bitno je vršiti validaciju vrednosti koje korisnik unosi pre nego što se unos primeni na upitu. To obuhvata da unos odgovora tipu podataka koji se očekuje, dužini i formatu, tako da je potrebno odbiti unos koji sadrži neočekivane znakove ili obrasce. Dodeljivanje minimalnih privilegija korisnicima baze podataka i aplikacijama je ključno, jer korišćenje naloga sa previše privilegija za zadatke koji se mogu obaviti sa nekim minumom privilegija, povećava uticaj uspešnog *SQL injection* napada.

Takođe, održavanje MySQL-a i *plugin*-ova ažuriranim je od velikog značaja, jer neki propusti iz prethodnih verzija mogu biti iskorišćene za neovlašćen pristup bazi podataka.

Pored toga, primena ograničenja na dužinu polja za unos pomaže u sprečavanju *buffer overflow* napada, a takođe i implementacija dogovarajućih mehanizama za upravljanje greškama je ključna kako bi se izbeglo otkrivanje osetljivih informacija u porukama o greškama.

Kada se govori o pristupu serveru preko mreže, da bi se zaštitila baza moguće je dozvoliti pristup samo sa lokalnog hosta, pomoću konfiguracionog fajla, gde se za *bind-adress* upisuje 127.0.0.1 što je obično i upisano kao podrazumevana vrednost kao na slici ispod.



Slika 33 Podešavanje bind-address u konfiguracionom fajlu

A pored toga je moguće koristiti i *firewall* za filtriranje dolaznih konekcija kao što je dato na slici ispod pomoću *ufw*-a.



Slika 34 Dodavanje pravila za pristup serveru samo sa specifične IP adrese

**5.1. Rezervna kopija podataka**

MySQL rezervna kopija je od suštinskog zančaja za obezbeđivanje integriteta podataka, dostupnost i mogućnost oporavka podataka u slučaju incidenata kao što su gubitak podataka, oštećenje podataka ili napad i nedozvoljen pristup bazi podataka. Pravljenje rezervnih kopija MySQL baze podataka pomaže u zaštiti od gore navednih problema koji se mogu desiti.

Uprkos bezbednosnim karakteristikama MySQL-a, ranjivost i dalje može postojati, što bazu čini podložnom napadima. Zbog toga je pravljenje rezervnih kopija od suštinskog značaja za ublažavanje uticaja takvih kršenja bezbednosti.

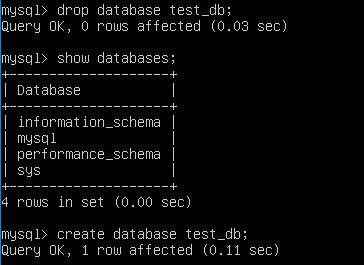
Strategije rezervnih kopija, kao što je sinhroizacija podataka sa centralizovanim repozitorijumima odnosno replikacija i eksternim uređajima za skladištenje, igraju važnu ulogu u obezbeđivanju sigurnosti podataka.

Na slici ispod prikazan je primer pravljenja rezervne kopije baze podataka test\_db, korišćenjem mysqldump komande.



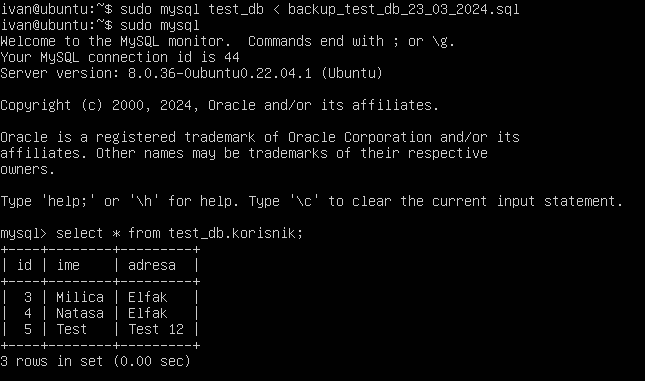
Slika 35 Korišćenje mysqldump komande

Nakon toga može se slobodno ukloniti baza sa servera i naraviti nova u kojoj će biti vraćeni podaci, kao što je prikazano u nastavku.



Slika 36 Brisanje i kreiranje test\_db baze gde će biti vraćeni podaci

Kada je nova baza kreirana može se izvršiti komanda za vraćanje podataka i strukture stare baze podataka (Slika 37).



Slika 37 Vraćanje podataka iz rezervne kopije

Komanda mysqldump nudi širok opseg raznih opcija koje je moguće kombinovati kako bi se postigli željeni rezultati, kao na primer moguće je korišćenje opcije za konktovanje na udaljenu bazu podataka i pravljenje rezervne kopije, takođe je moguće korišćenje mysql\_config\_editor datotetke za prijavljivanje prilikom pravljenja kopije podataka i tako dalje.

# 7. ZAKLJUČAK

Implementacija sigurnosnih mera i praksi, osigurava zaštitu podataka od neovlašćenog pristupa i manipuacije, ali i pruža osnovu za pouzdan rad MySQL baze podataka. Stalno unapređivanje sigurnosnih praksi i prilagođavanje promenama i standarima predstavlja osnov za očuvanje sigurnosti podataka.

Pored toga veoma je bitno obartiti pažnju i na kontrolu pristupa korisnika, kako ne bi došlo do zloupotrebe nekog od korisničkih naloga, kao i na vreme upoznati korisnike baza podataka sa načinima za pravilno rukovanje lozinkama.

LITERATURA

[1] P. Paul and P. Aithal, "Database security: an overview and analysis of current trend", SSRN Electronic Journal, 2019. https://doi.org/10.2139/ssrn.3497728

[2] A. Gupta, V. Bibhu, & R. Hussain, "Security measures in data mining", International Journal of Information Engineering and Electronic Business, vol. 4, no. 3, p. 34-39, 2012. https://doi.org/10.5815/ijieeb.2012.03.05

[3] K. Ahmad and M. Karim, "A method to prevent sql injection attack using an improved parameterized stored procedure", International Journal of Advanced Computer Science and Applications, vol. 12, no. 6, 2021. https://doi.org/10.14569/ijacsa.2021.0120636.

[4] https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/security-guidelines.html (Pregledano 27.02.2024.)

[5] J. Sloan, "Methods to secure databases against vulnerabilities",, 2015. https://doi.org/10.21236/ad1009288