|  |  |
| --- | --- |
|  | |
| JU MJEŠOVITA SREDNJA ELEKTROTEHNIČKA | ŠKOLSKA GODINA | |
| ŠKOLA TUZLA | 2019./2020. | |



PROJEKTNI ZADATAK

*Predmet: Baze podataka*

|  |  |
| --- | --- |
| PROFESOR: | UČENIK: |
| Minina Horozić dipl. ing, el. teh | Elmin Softić |
|  | Razred: 4T2 |

TUZLA, oktobar, 2019.god.

Sadržaj

[Analiza problema 3](#_Toc35708418)

[TEKST ZADATKA 3](#_Toc35708419)

[Opis 3](#_Toc35708420)

[Zadaci 3](#_Toc35708421)

[Kraći opis rješenja 4](#_Toc35708422)

[Objekti i Atributi 5](#_Toc35708423)

[Dijagram Objekti-Veze 6](#_Toc35708424)

[Opis Objekata 6](#_Toc35708425)

[Prevođenje Objekata 9](#_Toc35708426)

[Prevođenje Veza 11](#_Toc35708427)

[Veza Proizvodjac-Uredjaj 11](#_Toc35708428)

[Veza Uredjaj-Karakteristike 12](#_Toc35708429)

[Veza Proizvodjac-Dodatna\_oprema 13](#_Toc35708430)

[Relaciona šema 14](#_Toc35708431)

[Vježbe na časovima 17](#_Toc35708432)

[Vježba 1 – Forward/Reverse engineer, Sync i spajanje na server 17](#_Toc35708433)

[Spajanje na server 17](#_Toc35708434)

[Forward engineer 18](#_Toc35708435)

[Reverse engineer 19](#_Toc35708436)

[Synchronise 21](#_Toc35708437)

[Vježba 2 – Primjer kreiranja baze podataka 24](#_Toc35708438)

[Izvršavanje skripte 25](#_Toc35708439)

[Ručno pravljenje veza 26](#_Toc35708440)

[Import/Export podataka 28](#_Toc35708441)

[Create Model skripta 29](#_Toc35708442)

[Load Classic Models Skripta 31](#_Toc35708443)

[Vježba 3 – Import/Export podataka iz relacione baze podataka 33](#_Toc35708444)

[Vježba 4 - Jednostavni SELECT upiti 35](#_Toc35708445)

[Vježba 5 – Složeni SELECT upiti 42](#_Toc35708446)

[Vježba 6 – SQL Funkcije 48](#_Toc35708447)

[Vježba 7 – Group by, Having + funkcije 50](#_Toc35708448)

[Mogući upiti baze podataka 55](#_Toc35708449)

# Analiza problema

## TEKST ZADATKA

### Opis

Potrebno je osmisliti i realizirati bazu podataka koja može učinkovito poslužiti za potrebe trgovine prijenosnim računarima. U fazi analize, utvrđeno je da korisnik želi imati ažurnu evidenciju proizvođača notebook-a i njihovih generalnih zastupnika u BiH (naziv proizvođača, sjedište, zastupnik, sjedište zastupnika, telefon, ...), uređaja (model, karakteristike, cijena, garancija) i dodatne opreme za prijenosnike (oprema, opis, cijena, ...).

Osnovni zahtjevi korisnika su slijedeći:

* omogućiti uvid u raspoložive prijenosne uređaje (prema raznim kriterijima)
* popis dodatne opreme za određeni model

### Zadaci

1. Napraviti ER dijagram (entity-relationship diagram) dijagram sa svim bitnim elementima.
2. Dati kraći opis odabranog rješenja.
3. Napraviti SQL naredbe za kreiranje baze podataka.
4. Napuniti bazu podacima za potrebe testiranja.
5. Napraviti primjer upita za koje se očekuje da će biti najčešće upotrebljavani od strane korisnika baze podataka i opisati ih riječima

# Kraći opis rješenja

U ovom radu predstavljamo rješenje problema. Napravili smo bazu podataka.

Naime, u tekstu zadatak se traži baza podataka koja omogućava korisnicima da mogu provjeriti proizvođače i uređaje koje se prodaju u Notebook Shop-u. Zadatak je protumačen tako da bi se ovakva baza koristila u, npr. nekoj internet stranici na kojoj korisnik može da uradi sljedeće:

* Provjeri koje sve uređaje prodavnica ima na stanju
* Koje uređaje proizvodi koji od proizvođača
* Dobije uvid u raspoložive uređaje prema raznim kriterijama

Navedene tačke su zapravo i potrebe trgovine.

Definisali smo sve objekte potrebne, njihove atribute i odradili sve potrebno da bi ova baza bila funkcionalna.

# Objekti i Atributi

U ovom koraku ćemo izdvojiti sve stvari o kojima vodimo evidenciju u bazi podataka.

Entiteti koje smo zaključili da ćemo koristiti u ovoj bazi su:

* Proizvođač
* Uređaj
* Dodatna Oprema
* Karakteristike

Atributi objekta proizvođač:

* Naziv
* Sjedište
* Zastupnik
* Sjedište zastupnika
* Br. Telefona
* Br. Telefona zastupnika
* E-Mail

Atributi objekta Uređaj:

* Naziv
* Cijena
* Garancija
* Trajanje garancije
* Količina na stanju

Atributi objekta Dodatna Oprema:

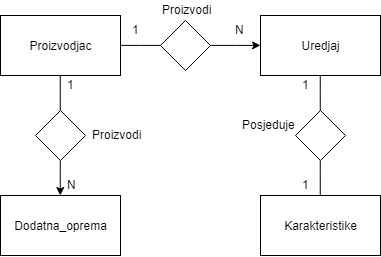
* Naziv
* Opis
* Cijena
* Količina
* Garancija
* Trajanje garancije

Atributi objekta Karakteristike:

* Procesor
* RAM
* SSD
* HDD
* Grafička kartica
* Operativni sistem
* Portovi
* Wi-Fi
* Bluetooth
* Veličina ekrana
* Itd.

# Dijagram Objekti-Veze

Kreiramo dijagram koji vizuelno opisuje veze između objekata. Ovaj korak je bitan jer postavljamo kako treba da se podaci odnose jedni na druge u bazi. Na osnovu ovog dijagrama pravimo bazu podataka.



Slika - Dijagram objekti-veze

# Opis Objekata

Sada opisujemo sve objekte u bazi. U ovom koraku ne pravimo nikakve veze već opisujemo koji su to podaci koje zapisujemo za pojedine objekte. Ovo ćemo izvesti na sljedeći način:

Objekat Proizvodjac

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Jedinstveni identifikator | Opis |
| id | INTEGER | da | da | Jedinstveni identifikator objekta |
| naziv | VARCHAR(35) | da | ne | Naziv proizvođača |
| sjediste | VARCHAR(35) | da | ne | Glavno sjedište proizvođača |
| zastupnik | VARCHAR(35) | da | ne | Naziv zastupnika |
| sjediste\_zastupnika | VARCHAR(35) | da | ne | Sjedište zastupnika |
| brTel | VARCHAR(20) | da | ne | Broj telefona proizvođača |
| brTelZast | VARCHAR(20) | da | ne | Broj telefona zastupnika |
| email | VARCHAR(40) | da | ne | E-mail adresa proizvođača |
| emailZast | VARCHAR(40) | da | ne | E-mail adresa zastupnika |

Objekat Dodatna\_Oprema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Jedinstveni identifikator | Opis |
| id | INTEGER | da | da | Jedinstveni identifikator objekta |
| naziv | VARCHAR(30) | da | ne | Naziv opreme |
| opis | VARCHAR(140) | da | ne | Kratki opis opreme |
| cijena | FLOAT | da | ne | Cijena opreme |
| kolicina | INTEGER | da | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | Da li oprema ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | Trajanje garancije, ukoliko ima |

Objekat Uredjaj

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Jedinstveni identifikator | Opis |
| id | INTEGER | da | da | Jedinstveni identifikator objekta |
| naziv | VARCHAR(35) | da | ne | Naziv uređaja |
| cijena | FLOAT | da | ne | Cijena uređaja |
| kolicina | INTEGER | da | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | Da li uređaj ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | Trajanje garancije, ukoliko ima |

Objekat Karakteristike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Jedinstveni identifikator | Opis |
| id | INTEGER | da | da | Jedinstveni identifikator objekta |
| procesor | VARCHAR(20) | da | ne |  |
| RAM | SMALLINT | da | ne |  |
| imaSSD | BOOL | da | ne |  |
| SSDKap | INTEGER | ne | ne | Kapacitet SSD-a, ukoliko postoji |
| imaHDD | BOOL | da | ne |  |
| HDDKap | INTEGER | ne | ne | Kapacitet HDD-a, ukoliko postoji |
| grafKart | VARCHAR(35) | da | ne |  |
| portovi | VARCHAR(100) | da | ne | Portovi koji su dostupni na uređaju |
| imaWIFI | BOOL | da | ne | Da li ima WiFi |
| imaBT | BOOL | da | ne | Da li ima Bluetooth |
| velEkran | FLOAT | da | ne | Veličina ekrana u inčima |
| operSis | VARCHAR(20) | da | ne | Operativni sistem na računaru |

Dakle, imamo naziv atributa, koji je taj atribut tip podatka, te da li je taj atribut obavezno polje i da li je jedinstveni identifikator.

# Prevođenje Objekata

U ovom koraku prevodimo objekte u tabele koje se koriste u bazi podataka. Ovo se radi tako sto svaki atribut objekta postaje kolona u tabeli, te identifikator postaje primarni ključ

Tabela PROIZVODJAC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Opis |
| id | INTEGER | da | da | Primarni ključ |
| naziv | VARCHAR(35) | da | ne | Naziv proizvođača |
| sjediste | VARCHAR(35) | da | ne | Glavno sjedište proizvođača |
| zastupnik | VARCHAR(35) | da | ne | Naziv zastupnika |
| sjediste\_zastupnika | VARCHAR(35) | da | ne | Sjedište zastupnika |
| brTel | VARCHAR(20) | da | ne | Broj telefona proizvođača |
| brTelZast | VARCHAR(20) | da | ne | Broj telefona zastupnika |
| email | VARCHAR(40) | da | ne | E-mail adresa proizvođača |
| emailZast | VARCHAR(40) | da | ne | E-mail adresa zastupnika |

Tabela DODATNA\_OPREMA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Opis |
| id | INTEGER | da | da | Primarni ključ |
| naziv | VARCHAR(30) | da | ne | Naziv opreme |
| opis | VARCHAR(140) | da | ne | Kratki opis opreme |
| cijena | FLOAT | da | ne | Cijena opreme |
| kolicina | INTEGER | da | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | Da li oprema ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | Trajanje garancije, ukoliko ima |

Tabela UREDJAJ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Opis |
| id | INTEGER | da | da | Primarni ključ |
| naziv | VARCHAR(35) | da | ne | Naziv uređaja |
| cijena | FLOAT | da | ne | Cijena uređaja |
| kolicina | INTEGER | da | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | Da li uređaj ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | Trajanje garancije, ukoliko ima |

Tabela KARAKTERISTIKE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Opis |
| id | INTEGER | da | da | Primarni ključ |
| procesor | VARCHAR(20) | da | ne |  |
| RAM | SMALLINT | da | ne |  |
| imaSSD | BOOL | da | ne |  |
| SSDKap | INTEGER | ne | ne | Kapacitet SSD-a, ukoliko postoji |
| imaHDD | BOOL | da | ne |  |
| HDDKap | INTEGER | ne | ne | Kapacitet HDD-a, ukoliko postoji |
| grafKart | VARCHAR(35) | da | ne |  |
| portovi | VARCHAR(100) | da | ne | Portovi koji su dostupni na uređaju |
| imaWIFI | BOOL | da | ne | Da li ima WiFi |
| imaBT | BOOL | da | ne | Da li ima Bluetooth |
| velEkran | FLOAT | da | ne | Veličina ekrana u inčima |
| operSis | VARCHAR(20) | da | ne | Operativni sistem na računaru |

# Prevođenje Veza

U ovom koraku realiziramo veze sa dijagrama. Ovo ćemo uraditi tako što ćemo najprije definisati vezu, pa onda je realizirati.

## Veza Proizvodjac-Uredjaj

Ova veza je tipa 1:N (jedan naprema više). To znači da:

* Jedan proizvođač može proizvesti više uređaja
* Jedan uređaj može biti napravljen od samo jednog proizvođača

Primjenjujemo pravilo[[1]](#footnote-1):

Svaki 1:N tip veze gdje je strana N totalna (svaki primjerak objekta na strani n je povezan sa jednim objektom na strani 1) ne prevodi se u novu relaciju, nego se primarni ključ objekta sa strane 1 umeće kao atribut u relacionu tabelu na strani N. Ovaj atribut postaje strani ključ relacione tabele na strani N. Svi eventualni atributi veze postaju atributi relacije na strani N.

Dakle, ovu Proizvdjac-Uredjaj realiziramo tako što u tabelu UREDJAJ unosimo primarni ključ tabele PROIZOVDJAC, a taj sada postaje strani ključ, jer nije više u svojoj tabeli.

Tabela UREDJAJ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Strani ključ | Opis |
| id | INTEGER | da | da | ne | Primarni ključ |
| naziv | VARCHAR(35) | da | ne | ne | Naziv uređaja |
| cijena | FLOAT | da | ne | ne | Cijena uređaja |
| kolicina | INTEGER | da | ne | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | ne | Da li uređaj ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | ne | Trajanje garancije, ukoliko ima |
| id\_proizvodjac | INTEGER | da | ne | da | Strani ključ |

## Veza Uredjaj-Karakteristike

Ova veza je tipa 1:1 (jedan naprema jedan). To znači da:

* Jedan uređaj može imati jedne karakteristike
* Jedne karakteristike mogu biti samo od jednog uređaja[[2]](#footnote-2)

Ovu vezu realiziramo tako što u tabelu UREDJAJ unosimo primarni ključ tabele KARAKTERISTIKE, te isti sada postaje strani ključ.

Modificiramo tabelu iz prethodne veze.

Tabela UREDJAJ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Strani ključ | Opis |
| id | INTEGER | da | da | ne | Primarni ključ |
| naziv | VARCHAR(35) | da | ne | ne | Naziv uređaja |
| cijena | FLOAT | da | ne | ne | Cijena uređaja |
| kolicina | INTEGER | da | ne | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | ne | Da li uređaj ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | ne | Trajanje garancije, ukoliko ima |
| id\_proizvodjac | INTEGER | da | ne | da | Strani ključ |
| id\_karak | INTEGER | da | ne | da | Strani ključ |

Ovo je sada konačna tabela UREDJAJ.

## Veza Proizvodjac-Dodatna\_oprema

Ova veza je također veza tipa 1:N.

* Jedan proizvođač može proizvesti više dodatne opreme
* Jedan predmet dodatne opreme proizvodi jedan proizvođač

Ponovo primjenjujemo pravilo:

Svaki 1:N tip veze gdje je strana N totalna (svaki primjerak objekta na strani n je povezan sa jednim objektom na strani 1) ne prevodi se u novu relaciju, nego se primarni ključ objekta sa strane 1 umeće kao atribut u relacionu tabelu na strani N. Ovaj atribut postaje strani ključ relacione tabele na strani N. Svi eventualni atributi veze postaju atributi relacije na strani N.

Dakle, uvodimo primarni ključ tabele PROIZVODJAC u tabelu DODATNA\_OPREMA, a taj sada postaje strani ključ.

Tabela DODATNA\_OPREMA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Strani Ključ | Opis |
| id | INTEGER | da | da | ne | Primarni ključ |
| naziv | VARCHAR(30) | da | ne | ne | Naziv opreme |
| opis | VARCHAR(140) | da | ne | ne | Kratki opis opreme |
| cijena | FLOAT | da | ne | ne | Cijena opreme |
| kolicina | INTEGER | da | ne | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | ne | Da li oprema ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | ne | Trajanje garancije, ukoliko ima |
| id\_proizvodjac | INTEGER | da | ne | da |  |

# Relaciona šema

U ovom koraku zaključujemo tabele koje će se koristiti u bazi. Ovo su konačne tabele.

Tabela DODATNA\_OPREMA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Strani Ključ | Opis |
| id | INTEGER | da | da | ne | Primarni ključ |
| naziv | VARCHAR(30) | da | ne | ne | Naziv opreme |
| opis | VARCHAR(140) | da | ne | ne | Kratki opis opreme |
| cijena | FLOAT | da | ne | ne | Cijena opreme |
| kolicina | INTEGER | da | ne | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | ne | Da li oprema ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | ne | Trajanje garancije, ukoliko ima |
| id\_proizvodjac | INTEGER | da | ne | da |  |

Tabela UREDJAJ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Strani ključ | Opis |
| id | INTEGER | da | da | ne | Primarni ključ |
| naziv | VARCHAR(35) | da | ne | ne | Naziv uređaja |
| cijena | FLOAT | da | ne | ne | Cijena uređaja |
| kolicina | INTEGER | da | ne | ne | Količina na stanju |
| imaGaranciju | BOOL | da | ne | ne | Da li uređaj ima garanciju |
| trajanjeGaran | SMALLINT | da | ne | ne | Trajanje garancije, ukoliko ima |
| id\_proizvodjac | INTEGER | da | ne | da | Strani ključ |
| id\_karak | INTEGER | da | ne | da | Strani ključ |

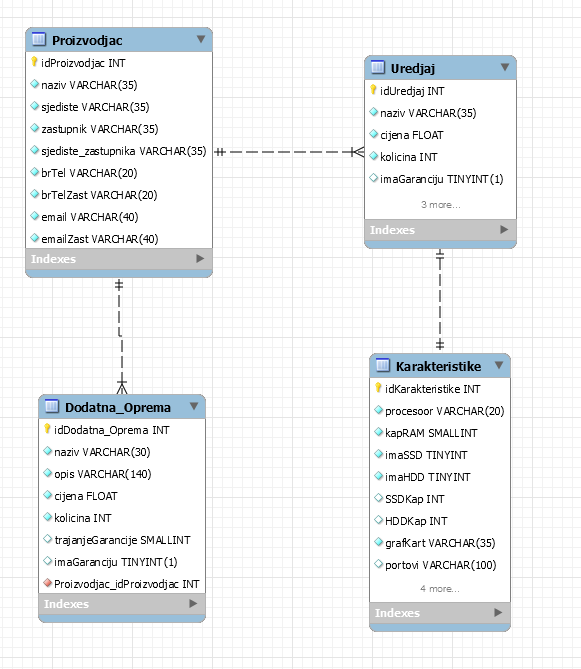
Tabela PROIZVODJAC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Opis |
| id | INTEGER | da | da | Primarni ključ |
| naziv | VARCHAR(35) | da | ne | Naziv proizvođača |
| sjediste | VARCHAR(35) | da | ne | Glavno sjedište proizvođača |
| zastupnik | VARCHAR(35) | da | ne | Naziv zastupnika |
| sjediste\_zastupnika | VARCHAR(35) | da | ne | Sjedište zastupnika |
| brTel | VARCHAR(20) | da | ne | Broj telefona proizvođača |
| brTelZast | VARCHAR(20) | da | ne | Broj telefona zastupnika |
| email | VARCHAR(40) | da | ne | E-mail adresa proizvođača |
| emailZast | VARCHAR(40) | da | ne | E-mail adresa zastupnika |

Tabela KARAKTERISTIKE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naziv atributa | Tip podatka | Obavezno polje | Primarni ključ | Opis |
| id | INTEGER | da | da | Primarni ključ |
| procesor | VARCHAR(20) | da | ne |  |
| RAM | SMALLINT | da | ne |  |
| imaSSD | BOOL | da | ne |  |
| SSDKap | INTEGER | ne | ne | Kapacitet SSD-a, ukoliko postoji |
| imaHDD | BOOL | da | ne |  |
| HDDKap | INTEGER | ne | ne | Kapacitet HDD-a, ukoliko postoji |
| grafKart | VARCHAR(35) | da | ne |  |
| portovi | VARCHAR(100) | da | ne | Portovi koji su dostupni na uređaju |
| imaWIFI | BOOL | da | ne | Da li ima WiFi |
| imaBT | BOOL | da | ne | Da li ima Bluetooth |
| velEkran | FLOAT | da | ne | Veličina ekrana u inčima |
| operSis | VARCHAR(20) | da | ne | Operativni sistem na računaru |

Navedene konačne tabele smo napravili u MySQL Workbench, te smo ih povezali.



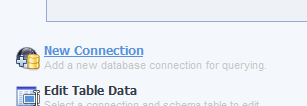
Slika – Relaciona šema

# Vježbe na časovima

## Vježba 1 – Forward/Reverse engineer, Sync i spajanje na server

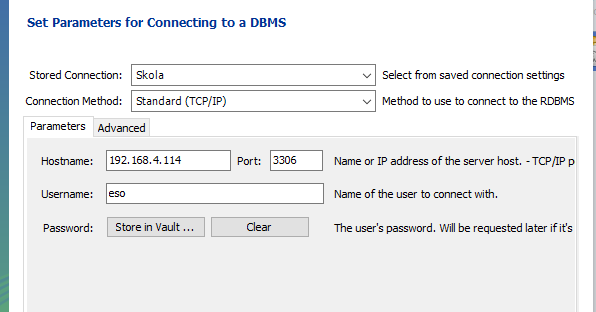
### Spajanje na server

Da bi radili sa našom bazom najprije moramo da se spojimo na server. U ovom radu ne opisujemo kako je napravljen naš profil na serveru već pretpostavljamo da je neki administrator unaprijed napravio naš profil.



Slika - Nova konekcija

Kreiramo novu konekciju klikom na opciju prikazanu na slici iznad.



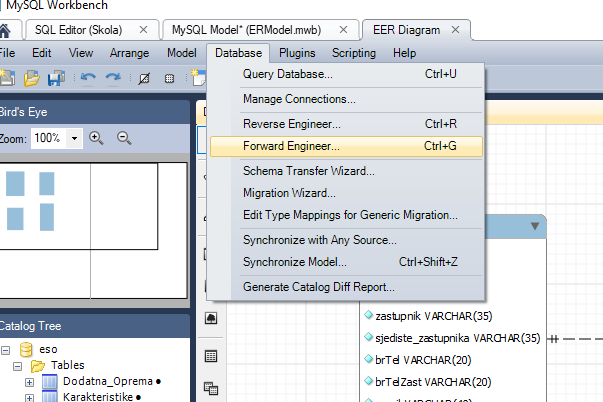
Slika - Postavljanje parametara konekcije

Zatim postavimo parametre konekcije koje smo dobili od administratora.

Kada želimo da se spojimo na server, potrebno je samo učitati konekciju iz liste konekcija.

### Forward engineer

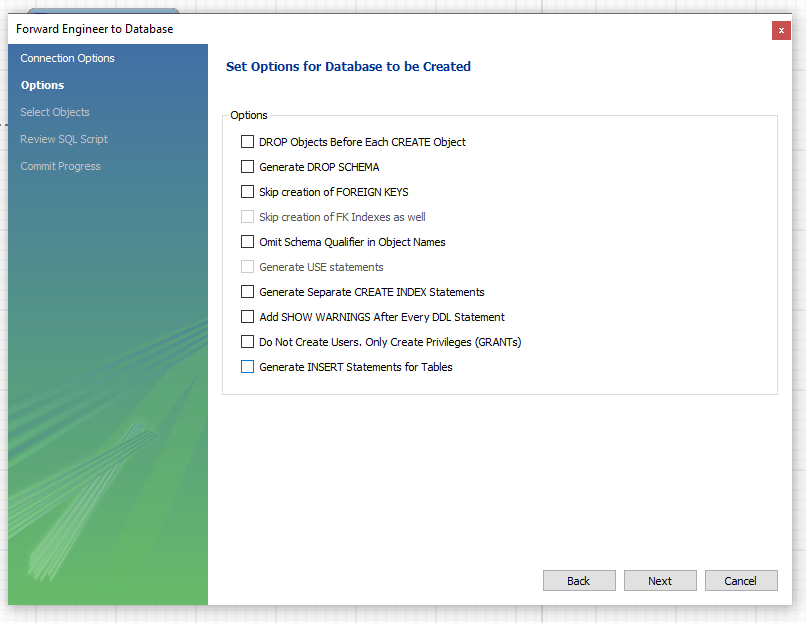
Ova funkcija uploaduje nasu bazu na server. Koristimo je kada želimo da proslijedimo našu šemu na server. Da bi ovo izveli, najprije moramo biti spojeni na server kojem želimo proslijediti našu šemu.



Slika - Opcija "Forward Engineer" u meniju

Iz menija na vrhu ekrana odaberemo „*Forward Engineer“*.

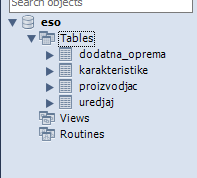
U prozorčiću koji nam se pojavi sada odaberemo server na koji želimo uploadovati šemu. Sljedeći par koraka u ovom novom prozoru su da odaberemo opcije kako uploadovati šemu na server. U biti šta radimo je da pratimo wizard koji se otvorio.



Slika - Opcije uploadovanja

Nakon forwardovanja šeme na server, dobiti ćemo poruku da je operacija uspješna, odnosno nije (ako nije).

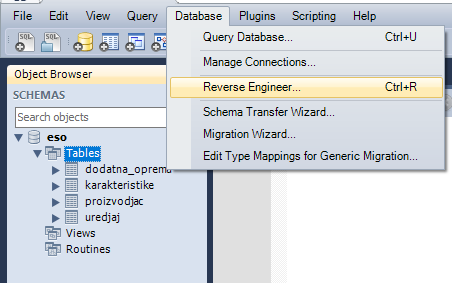
Trebala bi naša šema da se pojavi u SQL Editoru.



Slika - Šema u editoru

### Reverse engineer

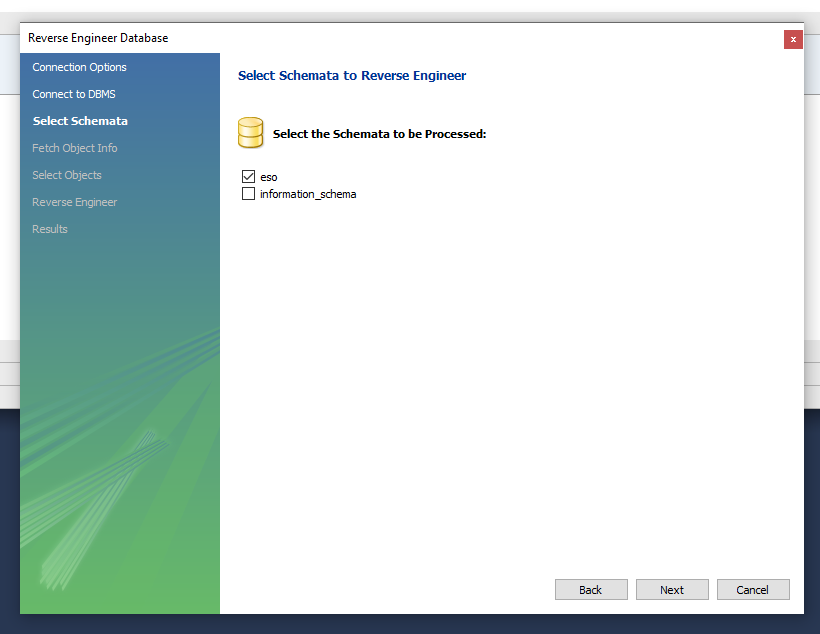
Ukoliko želimo da napravimo šemu od postojeće baze podataka sa servera, možemo koristiti opciju „Reverse Engineer“. Ona se nalazi na traci menija na vrhu: Database .> Reverse Engineer.



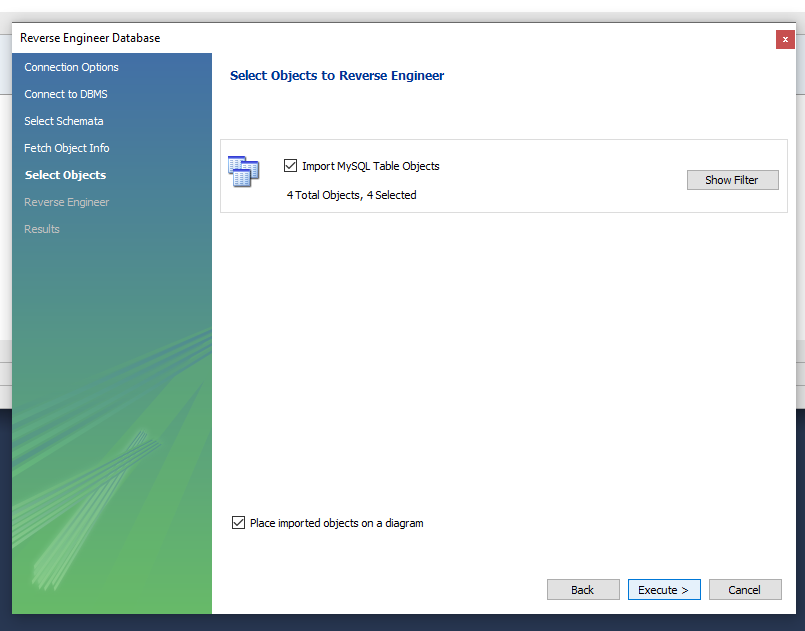
Slika - Reverse Engineer

Kada kliknemo na *Reverse Engineer* pojaviti će nam se opet prozor za odabir konekcije, kao i prije. Prateći wizard dolazimo do dijela gdje biramo *Scheme* koji želimo da reverse engineer-amo.

U nastavku, biramo šta želimo importovati sa servera, te filtriramo tačno šta želimo, ukoliko ima potrebe za tim.[[3]](#footnote-3)

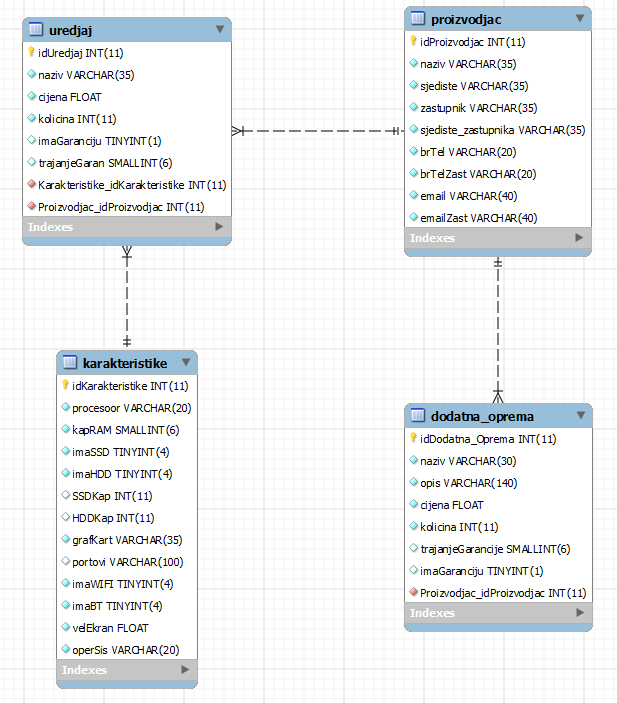


Slika - Odabir Scheme



Slika - Import i Filter

Pojaviti će nam se šema objekata i veza (ukoliko postoje) postavljenih jedno preko druge. Naš zadatak je da složimo sve kako bi nam šema bila preglednija, te napravimo promjene.

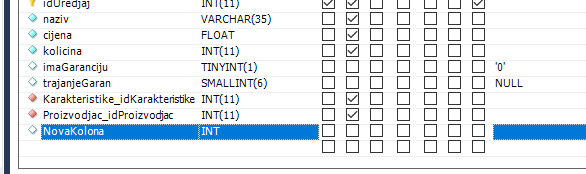


Slika - Naša nova šema sa nekim promjenama

### Synchronise

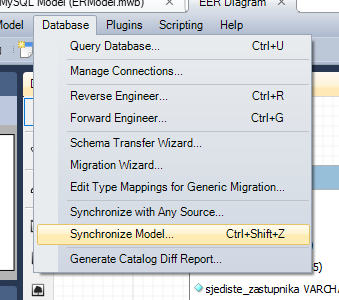
Synchronise je alat koji nam omogućava da promjene koje smo napravili na šemi sa servera ponovo postavimo na server s tim da ne moramo ponovo postavljati čitavu šemu servera, već samo možemo sinhronizovati promjene.

Recimo da smo napravili neke promjene na šemi koju smo povukli sa *„Reverse Engineer“*, te sada želimo da primjenimo te promjene.



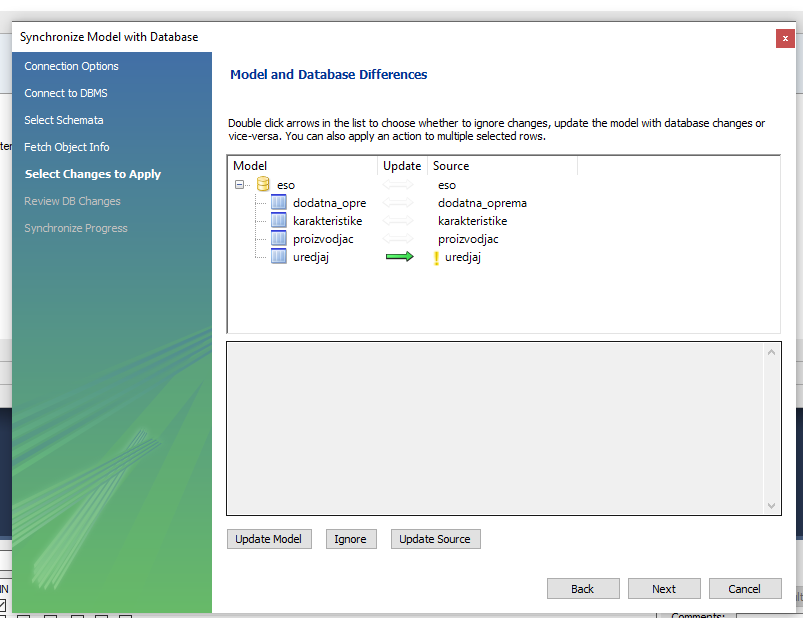
Slika - Promjena na šemi - dodali smo NovuKolonu

Trebamo sa menija odabrati Database > Synchronise Model.



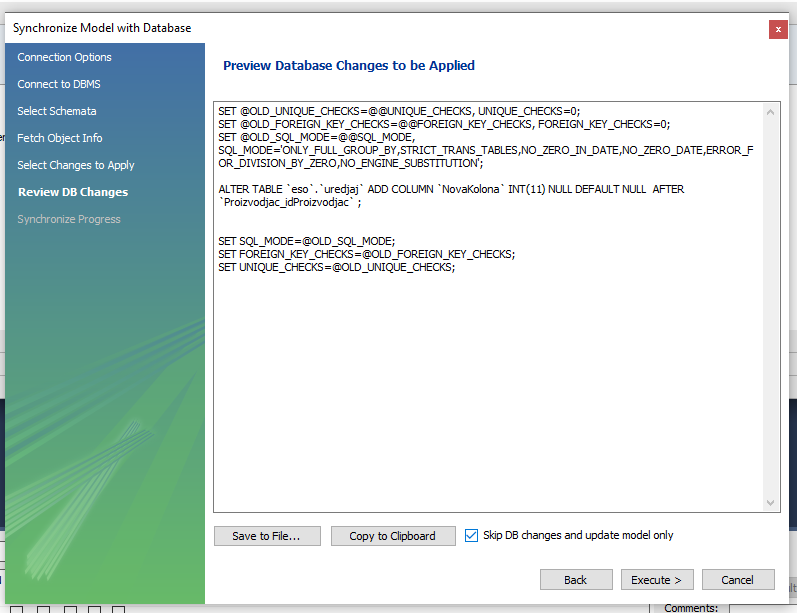
Slika - Opcija u meniju

Iznova, pojavljuje nam se meni gdje biramo server, a nakon njega; meni gdje biramo šta sinhroniziramo.



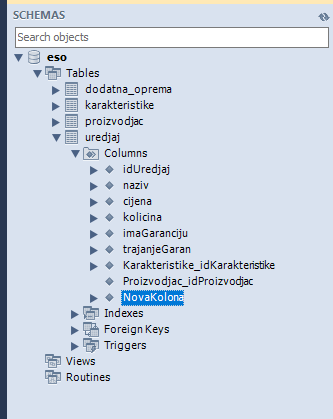
Slika - Šta sinhroniziramo

Kada izvršimo sinhronizaciju, dobićemo priliku da pregledamo šta se zapravo dešava sa SQL skriptom koja će se izvršiti, te je možemo sačuvati.



Slika - Pregled SQL naredbi - skripte

Klikom na dugme „Execute“ će se izvršiti naredbe i promjene će biti sinhronizirane.



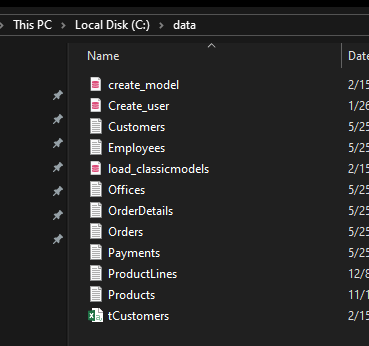
Slika - Promjena koju smo napravili

## Vježba 2 – Primjer kreiranja baze podataka

Na ovoj vježbi smo radili sljedeće:

1. Izvršavanje skripte
2. Upoznali korištenje SQL naredbi za kreiranje
3. Upoznali se sa importom i export podataka
4. Upoznali korištenje SQL naredbi za izmjenu i brisanje tabela
5. Upoznali naredbe za manipulisanje podacima u tabelama

Dobili smo podatke i skripte koje trebamo da obradimo na času.

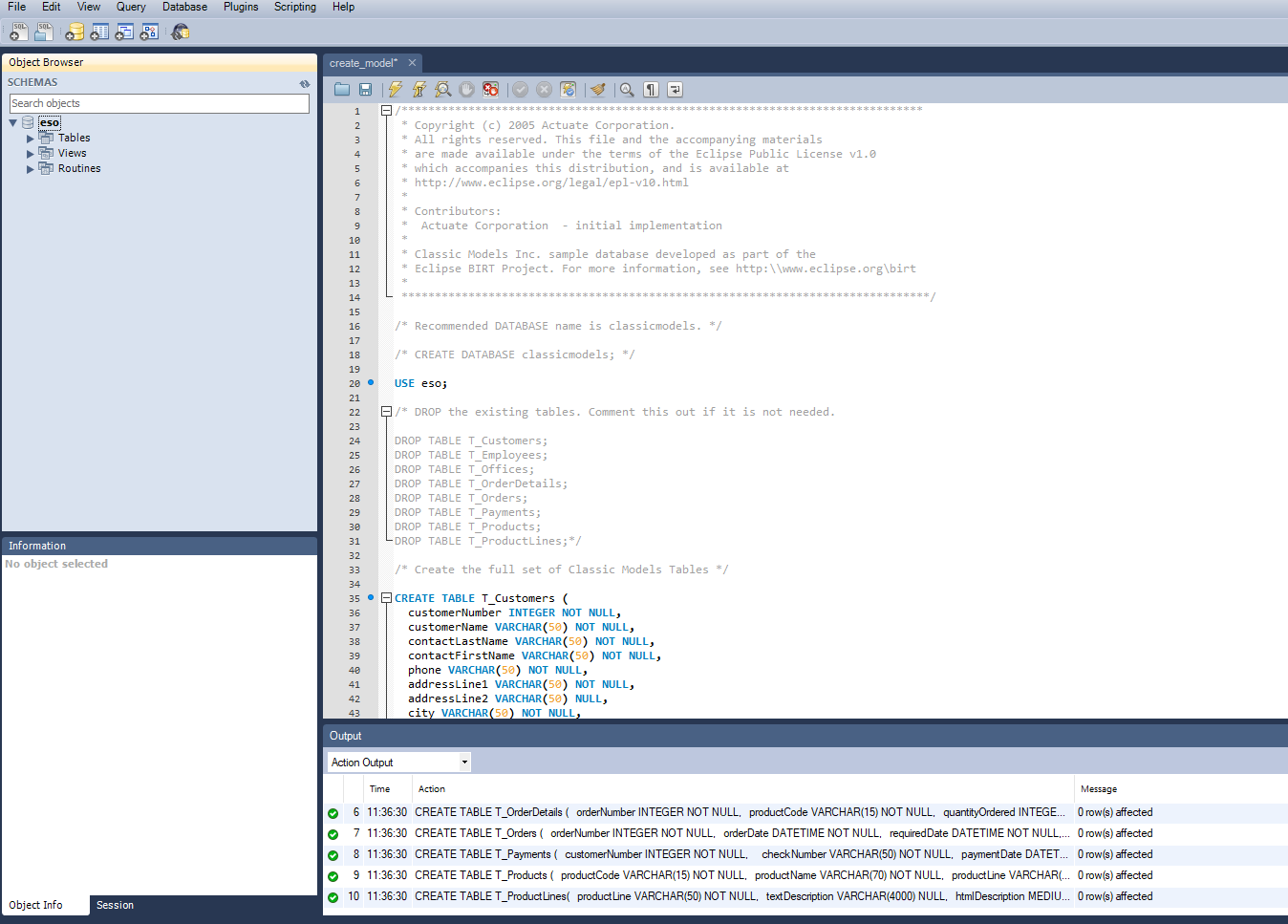


Slika - Podaci dobiveni za rad na času

### Izvršavanje skripte

Najprije je potrebno otvoriti SQL skriptu pomoću menija File > Open SQL script.

Skripta se učitava i prikazuje u Workbenchu.



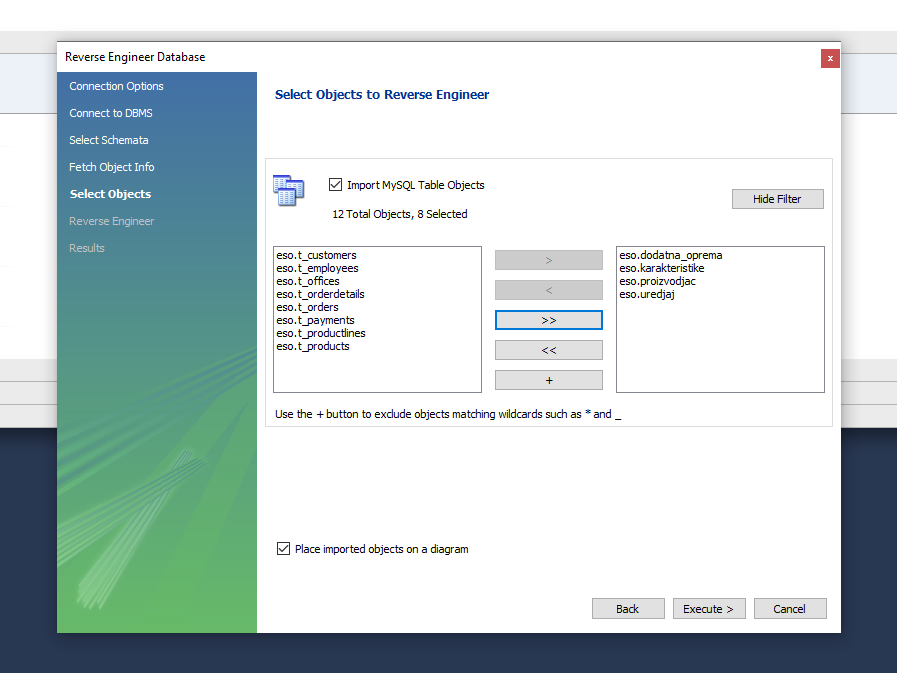
Slika - Učitana skripta

Na traci sa alatom imamo dugme koje pokreće skriptu. Izgleda kao mala žuta munja.

Kada kliknemo to dugme, i odaberemo server na kojem želimo raditi, izvršavamo skriptu.

Dalje slijedi da novo-napravljenu šemu povučemo sa servera da, kao prvo provjerimo da li je sve u redu, a ako nije, da uredimo šemu i sredimo je kako želimo.

Proces Reversa je isti kao što smo navodili ranije, samo što ćemo ovaj put u jednom od koraka uključiti filter. Ne želimo povući sve, već samo tabele koje smo upravo kreirali.



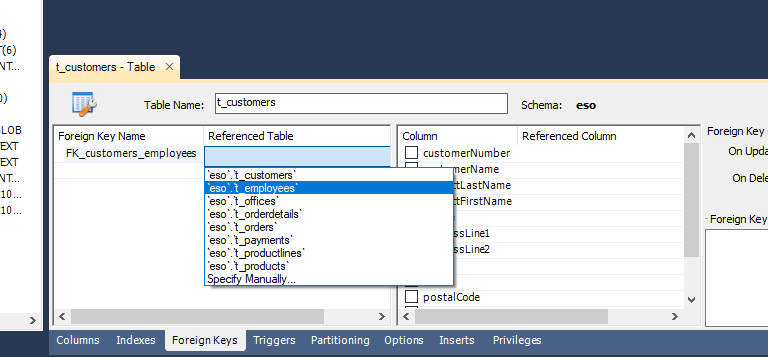
Slika - Postavka filtera

U našem slučaju, tabele su sve bile preturane i nije postojala nijedna veza. To nije nikakav problem. Naša skripta nije ni pravila veze, tako da je to naš zadatak. Ručno ćemo napraviti veze.

### Ručno pravljenje veza

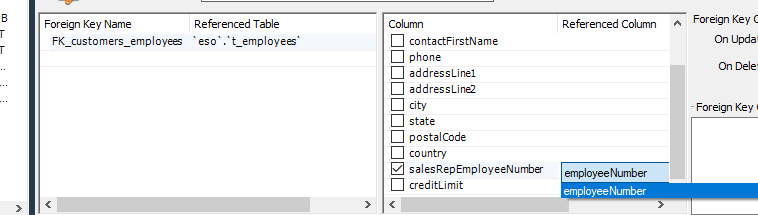
Kada želimo ručno da pravimo veze, potrebno je da u *„Model overview“* u Workbenchu kliknemo na tabelu u koju trebamo unijeti strani ključ, te na dnu ekrana odaberemo tab „*Foreign Keys*“.

Tu dodjeljujemo ime stranog ključa u polju „*Foreign Key Name*“, a u polju „*Referenced Table*“ bitamo tabelu iz koje želimo da uzmemo ključ.



Slika - Dodavanje stranog ključa

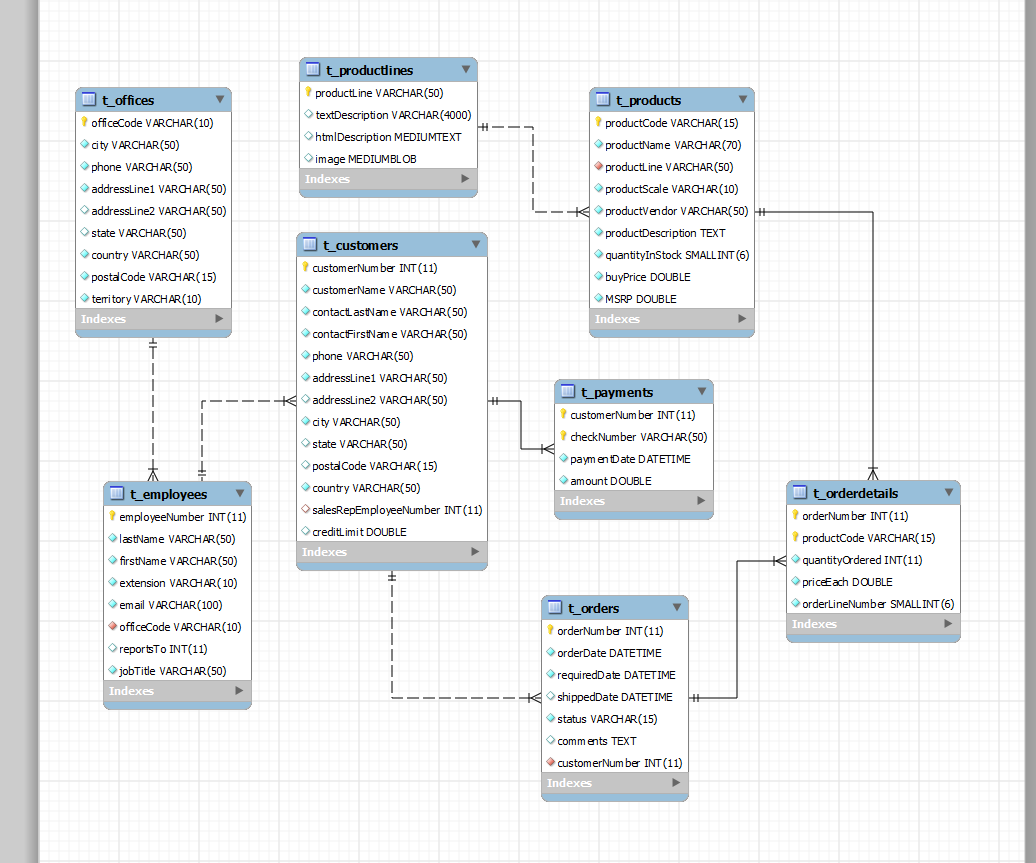
Desno od tih polja nalazi se lista svih kolona odabrane tabele. Tu treba da odaberemo koji je to ključ koji unosimo u tabelu.



Slika - Biramo ključ

Nakon što odaberemo ključ, sačuvamo promjene. Ovaj postupak ponavljamo dok ne napravimo sve potrebne veze.

U našem slučaju, konačni model izgleda ovako:



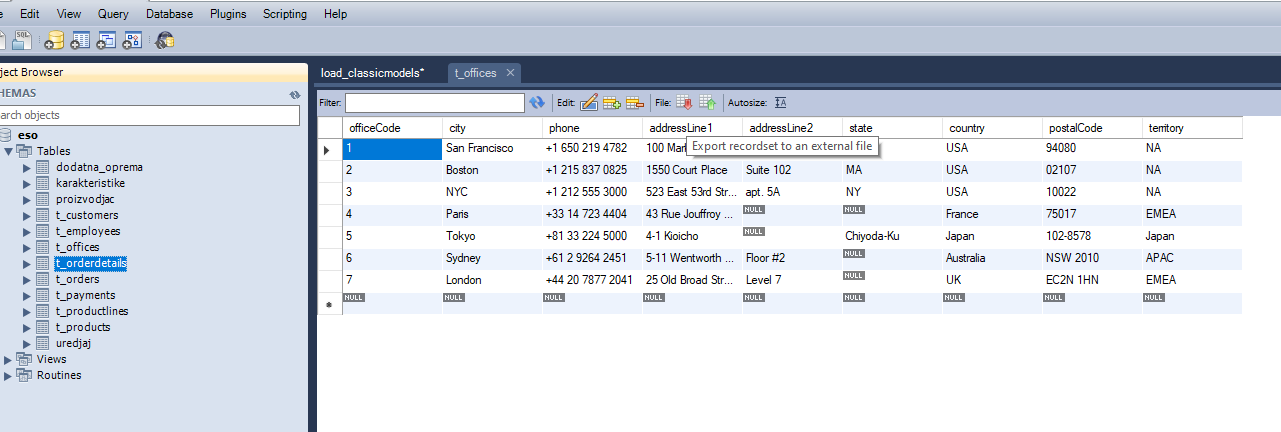
Slika - Konačni model

To je model koji smo napravili od datih fajlova na početku časa.

Sada uradimo sinhrnoizaciju i završili smo sa ovim dijelom.

### Import/Export podataka

Također smo dobili i skripte za populaciju nekih podataka u našu bazu podataka. Otvaramo i pokrećemo ih na isti način kao što smo prethodno dokumentovali i vidimo da su se u našim tabelama sada pojavili unosi podataka.



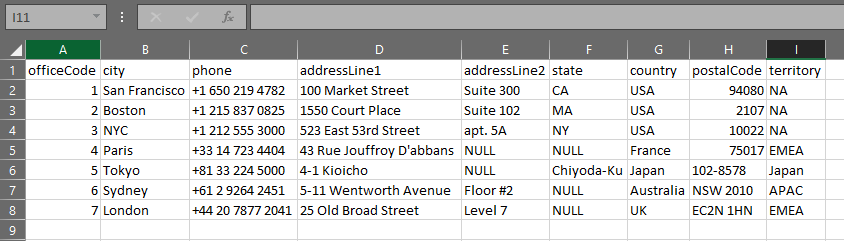
Slika - Podaci koje je skripta dodala

Na traci sa alatima imamo i mogućnost da izvedemo podatke u željeni fajl.

Taj fajl može biti .csv, SQL skripta ili nešto drugo.

Mi smo koristili CSV – ili *Comma Separated Values* – Vrijednosti Odvojene Zarezom.

Rezultirajuće datoteke se mogu otvoriti sa programom poput text editora ili programom za rad sa tablicama.



Slika - Primjer exportovanih podataka

Iste fajlove navedene možemo i importovati ukoliko želimo ponovo napuniti našu bazu sa nekim podacima.

### Create Model skripta

USE orders;

CREATE TABLE T\_Customers (

  customerNumber INTEGER NOT NULL,

  customerName VARCHAR(50) NOT NULL,

  contactLastName VARCHAR(50) NOT NULL,

  contactFirstName VARCHAR(50) NOT NULL,

  phone VARCHAR(50) NOT NULL,

  addressLine1 VARCHAR(50) NOT NULL,

  addressLine2 VARCHAR(50) NULL,

  city VARCHAR(50) NOT NULL,

  state VARCHAR(50) NULL,

  postalCode VARCHAR(15) NULL,

  country VARCHAR(50) NOT NULL,

  salesRepEmployeeNumber INTEGER NULL,

  creditLimit DOUBLE NULL,

  PRIMARY KEY (customerNumber)

);

CREATE TABLE T\_Employees (

  employeeNumber INTEGER NOT NULL,

  lastName VARCHAR(50) NOT NULL,

  firstName VARCHAR(50) NOT NULL,

  extension VARCHAR(10) NOT NULL,

  email VARCHAR(100) NOT NULL,

  officeCode VARCHAR(10) NOT NULL,

  reportsTo INTEGER NULL,

  jobTitle VARCHAR(50) NOT NULL,

  PRIMARY KEY (employeeNumber)

);

CREATE TABLE T\_Offices (

  officeCode VARCHAR(10) NOT NULL,

  city VARCHAR(50) NOT NULL,

  phone VARCHAR(50) NOT NULL,

  addressLine1 VARCHAR(50) NOT NULL,

  addressLine2 VARCHAR(50) NULL,

  state VARCHAR(50) NULL,

  country VARCHAR(50) NOT NULL,

  postalCode VARCHAR(15) NOT NULL,

  territory VARCHAR(10) NOT NULL,

  PRIMARY KEY (officeCode)

);

CREATE TABLE T\_OrderDetails (

  orderNumber INTEGER NOT NULL,

  productCode VARCHAR(15) NOT NULL,

  quantityOrdered INTEGER NOT NULL,

  priceEach DOUBLE NOT NULL,

  orderLineNumber SMALLINT NOT NULL,

  PRIMARY KEY (orderNumber, productCode)

);

CREATE TABLE T\_Orders (

  orderNumber INTEGER NOT NULL,

  orderDate DATETIME NOT NULL,

  requiredDate DATETIME NOT NULL,

  shippedDate DATETIME NULL,

  status VARCHAR(15) NOT NULL,

  comments TEXT NULL,

  customerNumber INTEGER NOT NULL,

  PRIMARY KEY (orderNumber)

);

CREATE TABLE T\_Payments (

  customerNumber INTEGER NOT NULL,

  checkNumber VARCHAR(50) NOT NULL,

  paymentDate DATETIME NOT NULL,

  amount DOUBLE NOT NULL,

  PRIMARY KEY (customerNumber, checkNumber)

);

CREATE TABLE T\_Products (

  productCode VARCHAR(15) NOT NULL,

  productName VARCHAR(70) NOT NULL,

  productLine VARCHAR(50) NOT NULL,

  productScale VARCHAR(10) NOT NULL,

  productVendor VARCHAR(50) NOT NULL,

  productDescription TEXT NOT NULL,

  quantityInStock SMALLINT NOT NULL,

  buyPrice DOUBLE NOT NULL,

  MSRP DOUBLE NOT NULL,

  PRIMARY KEY (productCode)

);

CREATE TABLE T\_ProductLines(

  productLine VARCHAR(50) NOT NULL,

  textDescription VARCHAR(4000) NULL,

  htmlDescription MEDIUMTEXT NULL,

  image MEDIUMBLOB NULL,

  PRIMARY KEY (productLine)

);

### Load Classic Models Skripta

USE orders;

/\* First make sure all the tables are empty \*/

DELETE FROM T\_Customers;

DELETE FROM T\_Employees;

DELETE FROM T\_Offices;

DELETE FROM T\_OrderDetails;

DELETE FROM T\_Orders;

DELETE FROM T\_Payments;

DELETE FROM T\_Products;

DELETE FROM T\_ProductLines;

# Load records into the tables. There should be no warnings.

LOAD DATA LOCAL INFILE 'C:\\data\\customers.txt' INTO TABLE T\_Customers

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n';

SHOW WARNINGS LIMIT 10;

LOAD DATA LOCAL INFILE 'C:\\data\\employees.txt' INTO TABLE T\_Employees

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n';

SHOW WARNINGS LIMIT 10;

LOAD DATA LOCAL INFILE 'C:\\data\\offices.txt' INTO TABLE T\_Offices

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n';

SHOW WARNINGS LIMIT 10;

LOAD DATA LOCAL INFILE 'C:\\data\\orderdetails.txt' INTO TABLE T\_OrderDetails

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n';

SHOW WARNINGS LIMIT 10;

LOAD DATA LOCAL INFILE 'C:\\data\\orders.txt' INTO TABLE T\_Orders

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n';

SHOW WARNINGS LIMIT 10;

LOAD DATA LOCAL INFILE 'C:\\data\\payments.txt' INTO TABLE T\_Payments

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n';

SHOW WARNINGS LIMIT 10;

LOAD DATA LOCAL INFILE 'C:\\data\\products.txt' INTO TABLE T\_Products

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n';

SHOW WARNINGS LIMIT 10;

LOAD DATA LOCAL INFILE 'C:\\data\\ProductLines.txt' INTO TABLE T\_ProductLines

          FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n'

          (productLine , textDescription);

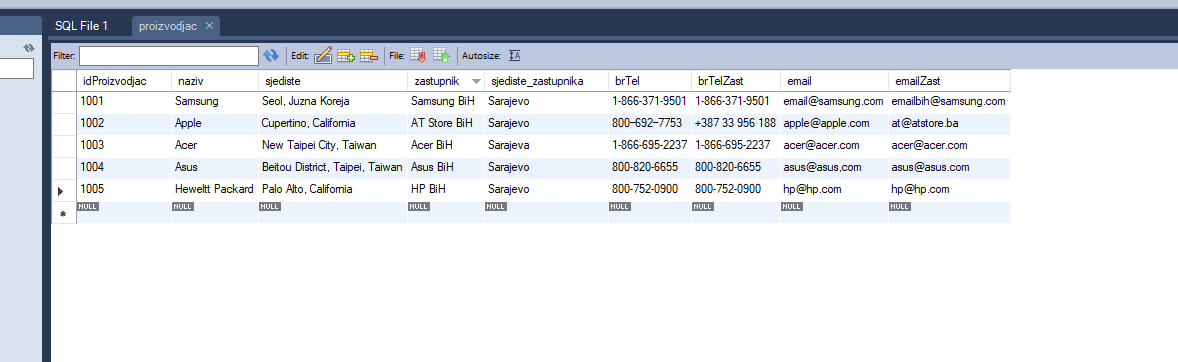
SHOW WARNINGS LIMIT 10;

## Vježba 3 – Import/Export podataka iz relacione baze podataka

U ovoj vježbi smo radili u biti iste zadatke kao u prethodnoj, samo što smo sada unosili podatke u našu bazu podataka, te smo čuvali skripte kako bi ih predstavili u ovom radu.

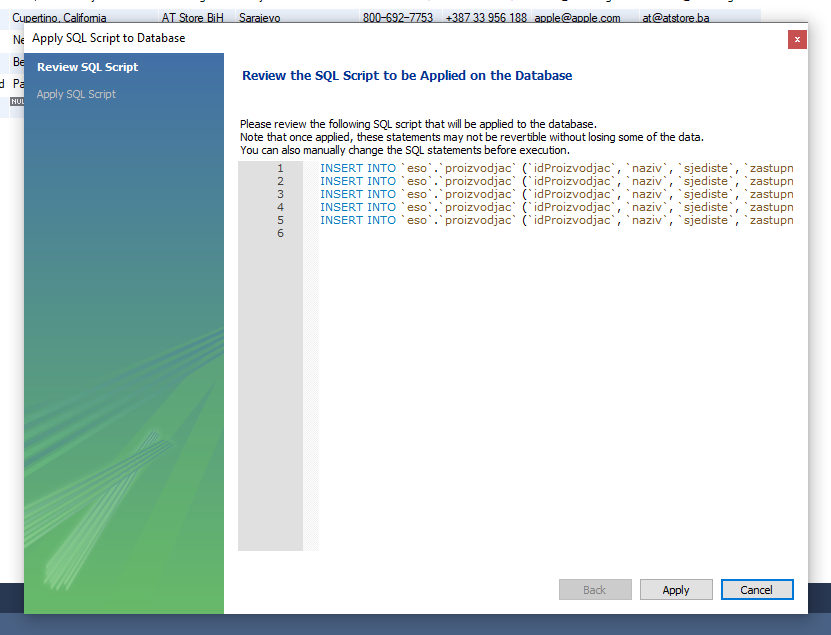
Unosili smo podatke ručno u naše tabele. Dakle, otvorili tabelu tako što smo kliknuli desnim klikom, a potom na opciju „*Edit Table Data*“ koja se otvorila u padajućem meniju.

Sada imamo tabelu ispred nas u koju unosimo podatke kao što bi unosili u neku Excel tabelu.

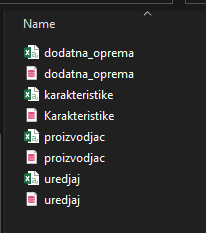


Slika - Unošenje podataka u našu tabelu

Nakon što unesemo podatke, vrijeme je da izvršimo promjene. Otvara nam se meni u kojem dobivamo pregled skripti koje će se izvršiti nakon što kliknemo „*Execute*“. Skripte smo sačuvali te ćemo dodati sada jednu kao primjer jedne takve skripte.



Slika - Pregled skripte



Slika - Sačuvane skripte i CSV fajlovi

Primjer jedne skripte:

/\*

-- Query: SELECT \* FROM eso.proizvodjac

LIMIT 0, 1000

-- Date: 2019-12-05 12:53

\*/

INSERT INTO `proizvodjac` (`idProizvodjac`,`naziv`,`sjediste`,`zastupnik`,`sjediste\_zastupnika`,`brTel`,`brTelZast`,`email`,`emailZast`) VALUES (1001,'Samsung','Seol, Juzna Koreja','Samsung BiH','Sarajevo','1-866-371-9501','1-866-371-9501','email@samsung.com','emailbih@samsung.com');

INSERT INTO `proizvodjac` (`idProizvodjac`,`naziv`,`sjediste`,`zastupnik`,`sjediste\_zastupnika`,`brTel`,`brTelZast`,`email`,`emailZast`) VALUES (1002,'Apple','Cupertino, California','AT Store BiH','Sarajevo','800–692–7753','+387 33 956 188','apple@apple.com','at@atstore.ba');

INSERT INTO `proizvodjac` (`idProizvodjac`,`naziv`,`sjediste`,`zastupnik`,`sjediste\_zastupnika`,`brTel`,`brTelZast`,`email`,`emailZast`) VALUES (1003,'Acer','New Taipei City, Taiwan','Acer BiH','Sarajeva','1-866-695-2237','1-866-695-2237','acer@acer.com','acer@acer.com');

INSERT INTO `proizvodjac` (`idProizvodjac`,`naziv`,`sjediste`,`zastupnik`,`sjediste\_zastupnika`,`brTel`,`brTelZast`,`email`,`emailZast`) VALUES (1004,'Asus','Beitou District, Taipei, Taiwan','Asus BiH','Sarajevo','800-820-6655','800-820-6655','asus@asus,com','asus@asus.com');

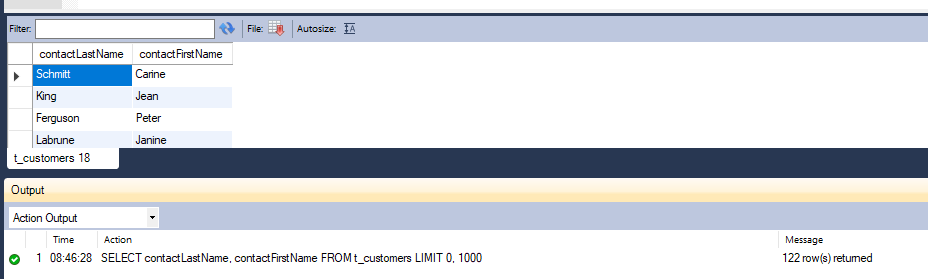
INSERT INTO `proizvodjac` (`idProizvodjac`,`naziv`,`sjediste`,`zastupnik`,`sjediste\_zastupnika`,`brTel`,`brTelZast`,`email`,`emailZast`) VALUES (1005,'Heweltt Packard','Palo Alto, California','HP BiH','Sarajevo','800-752-0900','800-752-0900','hp@hp.com','hp@hp.com');

(korištene su boje kako bi istakli naredbe od podataka)

## Vježba 4 - Jednostavni SELECT upiti

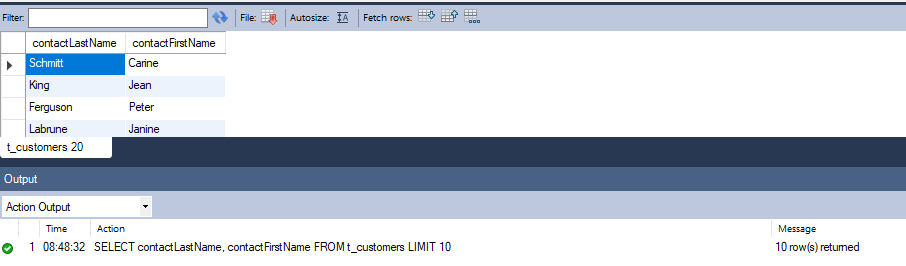
1. Iz tabele **Customers** selektovati kolone **contactLastName** i **contactFirstName**. Ispisati sve redove.

SELECT contactLastName, contactFirstName FROM t\_customers;



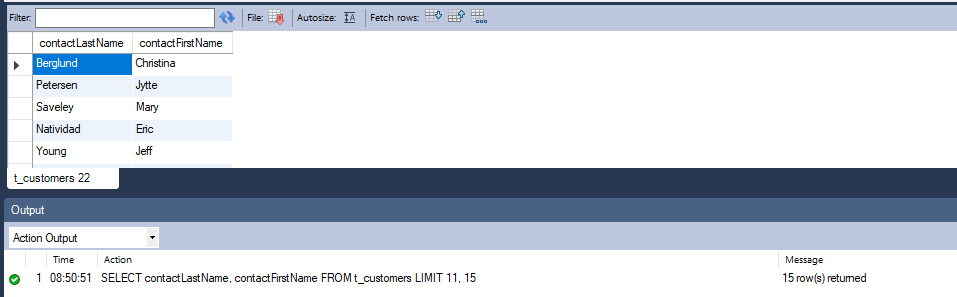
1. Modifkovati prethodnu naredbu tako da se izlista samo 10 prvih slogova.

SELECT contactLastName, contactFirstName FROM t\_customers LIMIT 10;



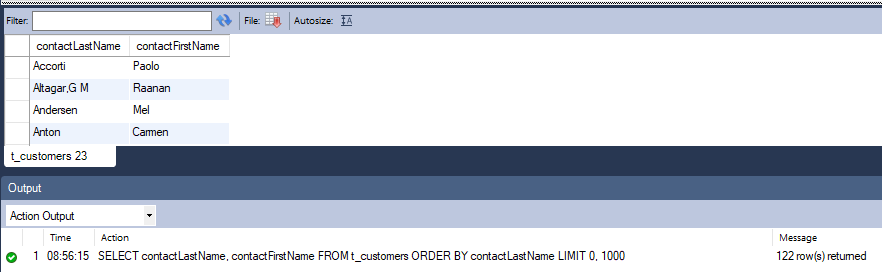
1. Naredbu 1 modifikovati tako da se ispisuju slogovi počevši od 11-og pa onda 15 narednih.

SELECT contactLastName, contactFirstName FROM t\_customers LIMIT 11, 15;



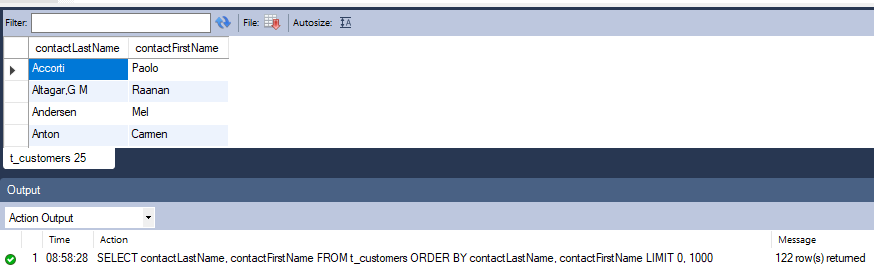
1. Modifikovati naredbu 1 dodavanjem odredbe za sortiranje po koloni **contactLastName**.

SELECT contactLastName, contactFirstName FROM t\_customers ORDER BY contactLastName;



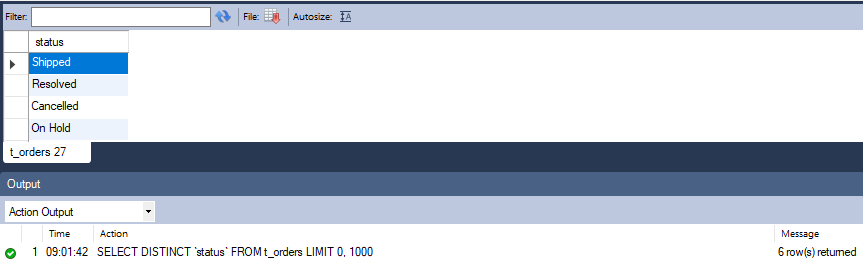
1. Modifikovati prethodnu naredbu tako da sortiranje bude po koloni contactLastName a zatim i koloni **contactFirstName**

SELECT contactLastName, contactFirstName FROM t\_customers ORDER BY contactLastName, contactFirstName;



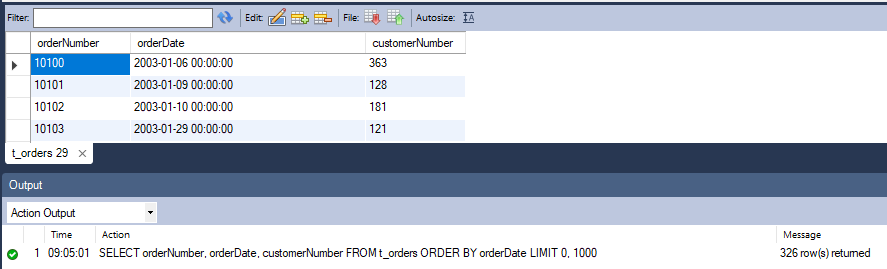
1. Napisati naredbu kojom se iz tabele **Orders** ispisuju samo jedinstvene vrijednosti u koloni **status**.

SELECT DISTINCT `status` FROM t\_orders;



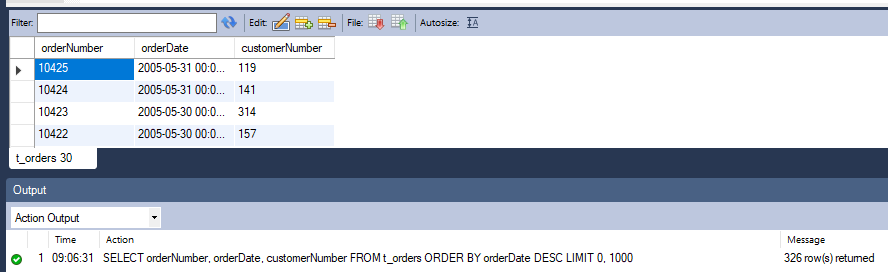
1. Iz tabele **Orders** ispisati **orderNumber**, **orderDate** i **customerNumber** sortirane po polju **orderDate** u rastućem redoslijedu

SELECT orderNumber, orderDate, customerNumber FROM t\_orders ORDER BY orderDate;



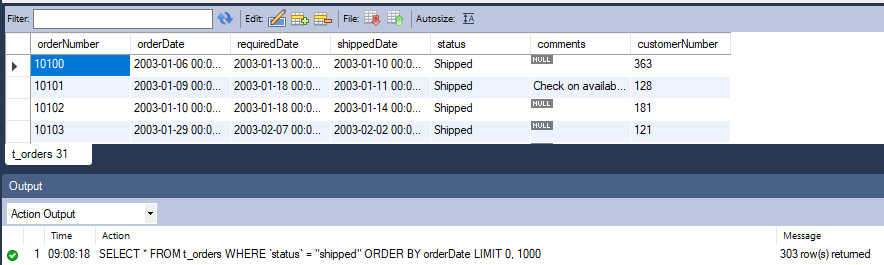
1. Iz tabele **Orders** ispisati **orderNumber**, **orderDate** i **customerNumber** sortirane po polju **orderDate** u opadajućem redoslijedu

SELECT orderNumber, orderDate, customerNumber FROM t\_orders ORDER BY orderDate DESC;



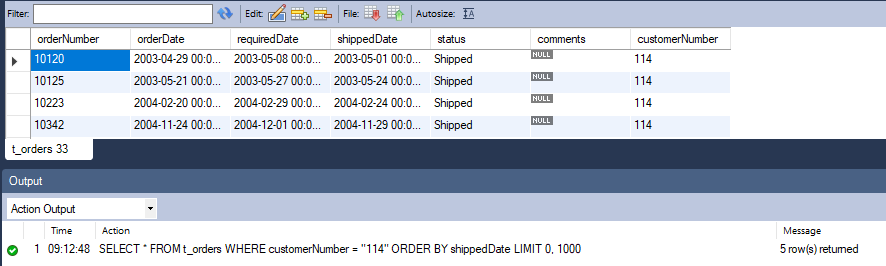
1. Ispisati sve kolone iz tabele **Order** samo ako je vrijdnost kolone **status =“shipped“** (isporučeno). Sortirati ih po **orderDate**

SELECT \* FROM t\_orders WHERE `status` = "shipped" ORDER BY orderDate;



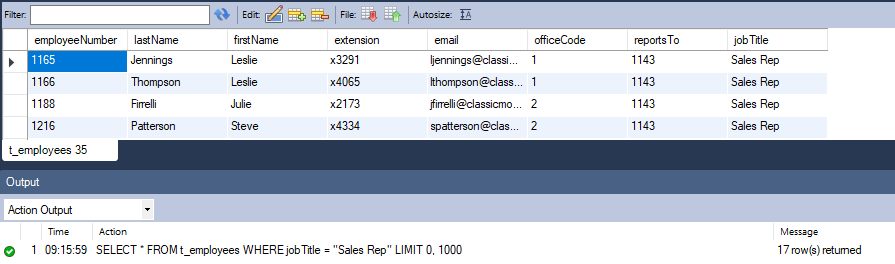
1. Ispisati sve podatke iz tabele **Order** koje se odnose na kupca sa brojem 114 (**CustomerNumber**). Sortirati po polju s**hippedDate**.

SELECT \* FROM t\_orders WHERE customerNumber = "114" ORDER BY shippedDate;



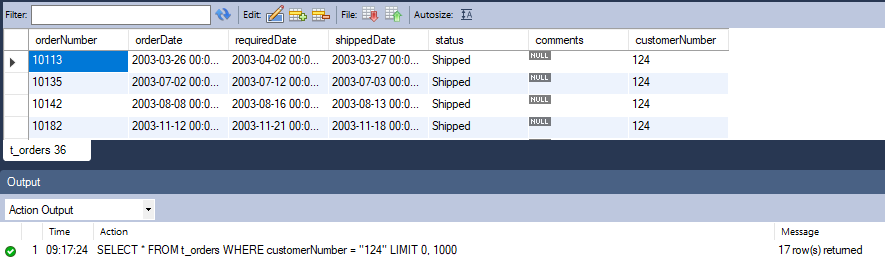
1. Iz tabele **Employees** ispisati samo one slogove kod kojih kolona **jobTitle** ima vrijednost „**Sales Rep“**.

SELECT \* FROM t\_employees WHERE jobTitle = "Sales Rep";



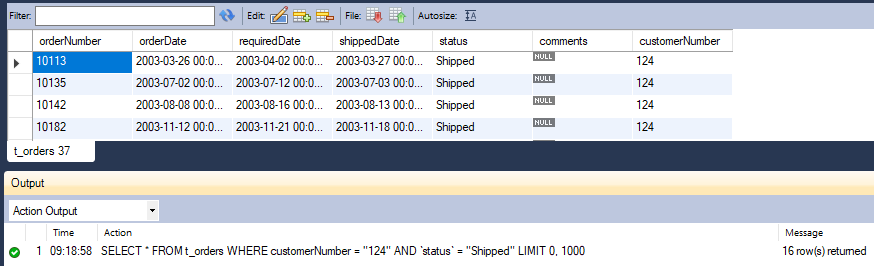
1. Iz tabele **Orders** izlistati sve narudzbe koje pripadaju kupcu broj **124** (**CustomerNumber**).

SELECT \* FROM t\_orders WHERE customerNumber = "124";



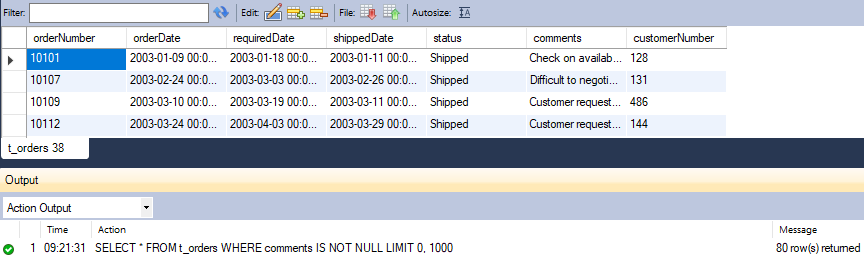
1. Modifikovati prethodni upit tako da se ispisuju samo narudzbe navedenog korisnika u statusu ‚**Shipped** ‘.

SELECT \* FROM t\_orders WHERE customerNumber = "124" AND `status` = "Shipped";



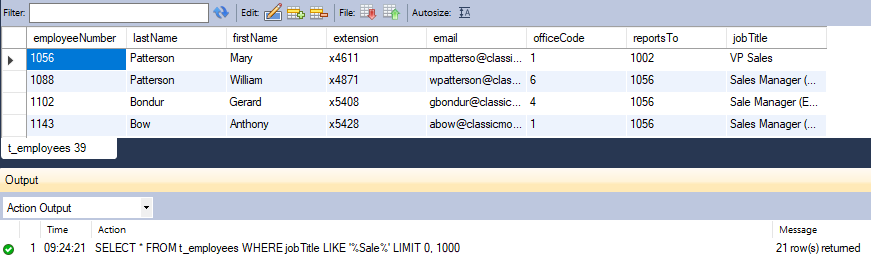
1. Iz tabele **Orders** izlistati sve narudzbe koje u polju **comments** imaju neki podatak.

SELECT \* FROM t\_orders WHERE comments IS NOT NULL;



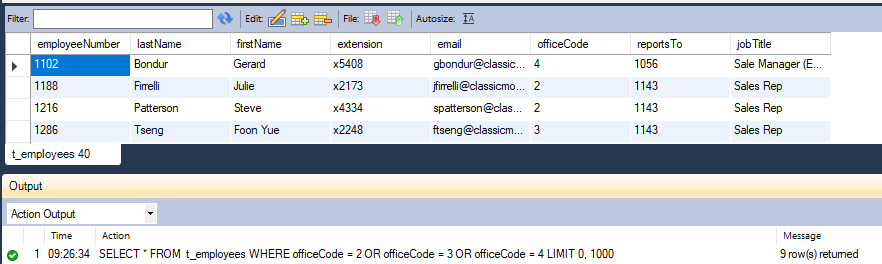
1. Iz tabele **Employees** izlistati podatke o svim radnicima koji u polju **JobTitle** imaju niz karaktera ‚**Sale**‘ bilo gdje u polju.

SELECT \* FROM t\_employees WHERE jobTitle LIKE '%Sale%';

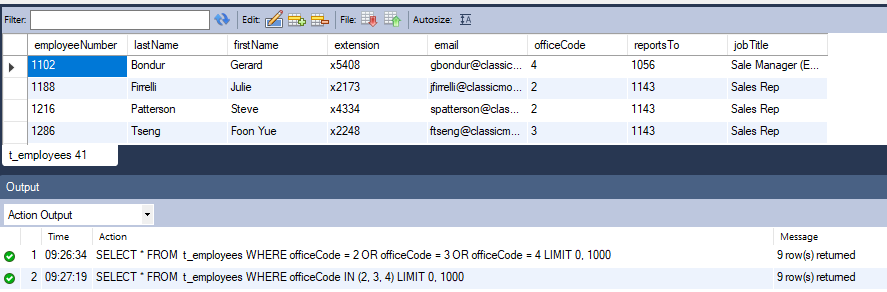


1. Iz tabele **Employees** izlistati podatke o radnicima kod kojih je u polju **officeCode** 2,3 ili 4. Uraditi na dva načina.

SELECT \* FROM t\_employees WHERE officeCode = 2 OR officeCode = 3 OR officeCode = 4;

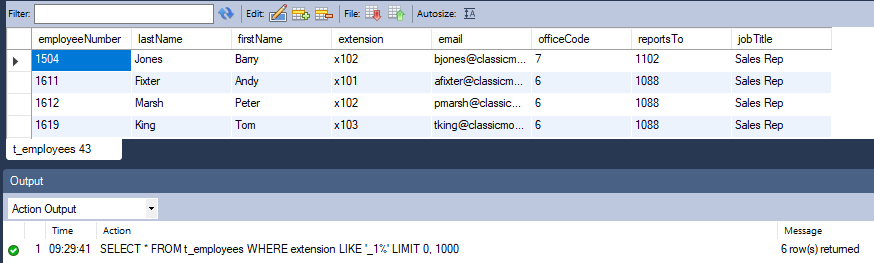


SELECT \* FROM t\_employees WHERE officeCode IN (2, 3, 4);



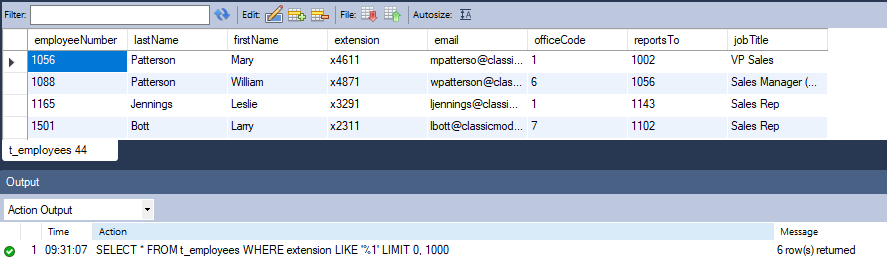
1. Iz tabele **Employees** izlistati podatke o radnicima kod kojih je u polju **extension** na drugom mjestu karakter „**1**“.

SELECT \* FROM t\_employees WHERE extension LIKE '\_1%';



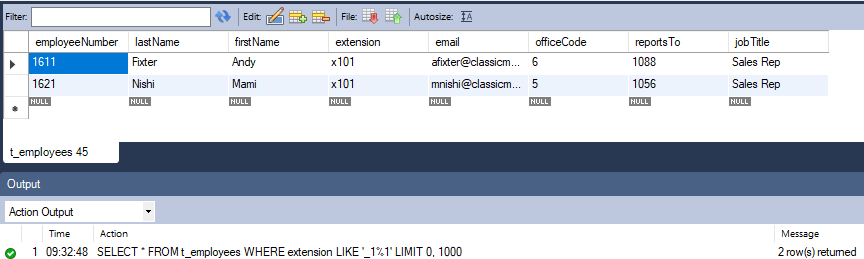
1. Iz tabele **Employees** izlistati podatke o radnicima kod kojih polje **extension** završava karakterom „**1**“.

SELECT \* FROM t\_employees WHERE extension LIKE '%1';



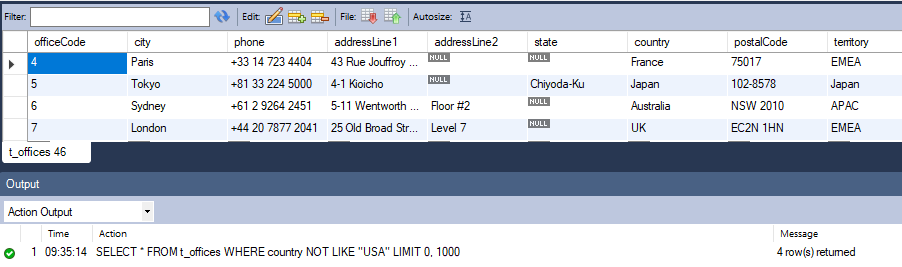
1. Iz tabele **Employees** izlistati podatke o radnicima kod kojih je u polju **extension** na drugom mjestu karakter „1“ i koje završava karakterom „1“.

SELECT \* FROM t\_employees WHERE extension LIKE '\_1%1';



1. Iz tabele **Offices** ispisati podatke o kancelarijama koje nisu iz **USA** (korisiti polje **Contry**).

SELECT \* FROM t\_offices WHERE country NOT LIKE "USA";



## Vježba 5 – Složeni SELECT upiti

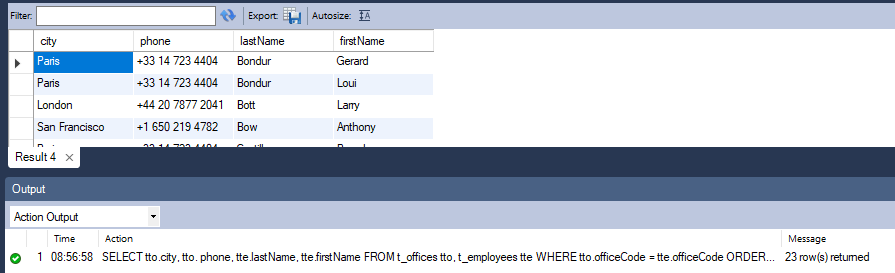
1. Iz tabele **Offices** ispisati **city** i **phone**, a iz tabele **Employees** ispisati kolone **lastName** i **firstName**. Sortirati po poljima **lastName** i **firstName**.

SELECT tto.city, tto. phone, tte.lastName, tte.firstName

FROM t\_offices tto, t\_employees tte

WHERE tto.officeCode = tte.officeCode

ORDER BY lastName, firstName



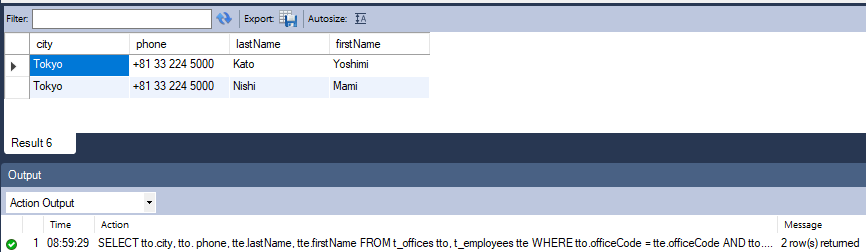
1. Modifikovati prethodni upit tako da se ispisuju podaci o radnicima iz kancelarije koja je iz Tokija (city = Tokyo).

SELECT tto.city, tto. phone, tte.lastName, tte.firstName

FROM t\_offices tto, t\_employees tte

WHERE tto.officeCode = tte.officeCode AND tto.city = "Tokyo"

ORDER BY lastName, firstName



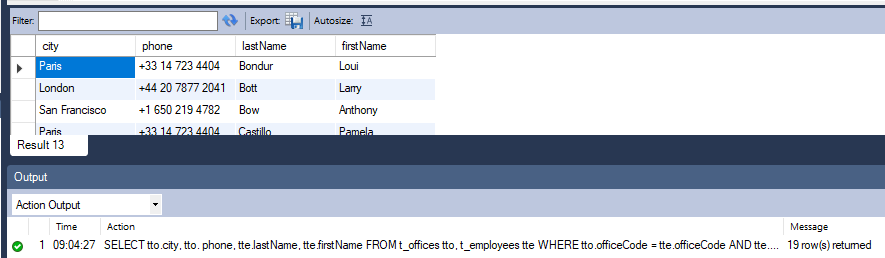
1. Modifikovati upit iz zadatka 1 tako da prikazuje samo podatke o radnicima koji u polju  
   **jobTitle** imaju string “**Sales**”.

SELECT tto.city, tto. phone, tte.lastName, tte.firstName

FROM t\_offices tto, t\_employees tte

WHERE tto.officeCode = tte.officeCode AND tte.jobTitle LIKE "Sales%"

ORDER BY lastName, firstName



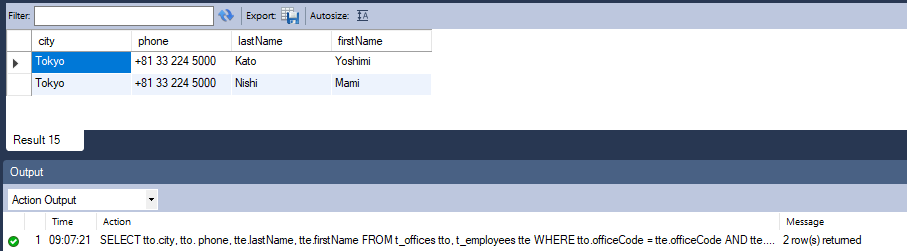
1. Modifikovati upit iz zadatka 1 tako da se ispisuju podaci o radnicima iz kancelarije koja je iz Tokija (city = Tokyo) i da prikazuje samo podatke o radnicima koji u polju **jobTitle** imaju string “**Sale**”.

SELECT tto.city, tto. phone, tte.lastName, tte.firstName

FROM t\_offices tto, t\_employees tte

WHERE tto.officeCode = tte.officeCode AND tte.jobTitle LIKE "Sales%" AND tto.city = "Tokyo"

ORDER BY lastName, firstName

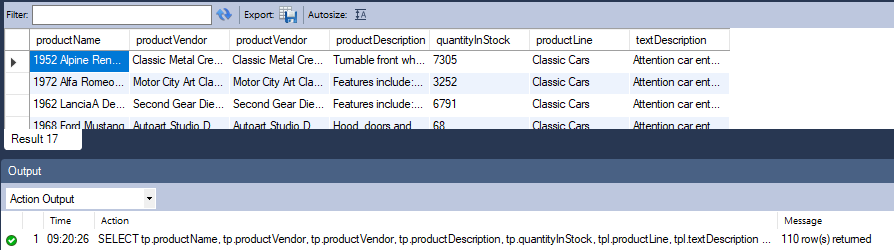


1. Povezati tabele **products** i **productLines** . Iz tabele products ispisati **productName**,  
   **productVendor**, **productDescription** i **quantityInStock**, a iz tabele **ProductLines**ispisati **textDescription**.

SELECT tp.productName, tp.productVendor, tp.productVendor, tp.productDescription, tp.quantityInStock, tpl.productLine, tpl.textDescription

FROM t\_products tp, t\_productlines tpl

WHERE tp.productLine = tpl.productLine

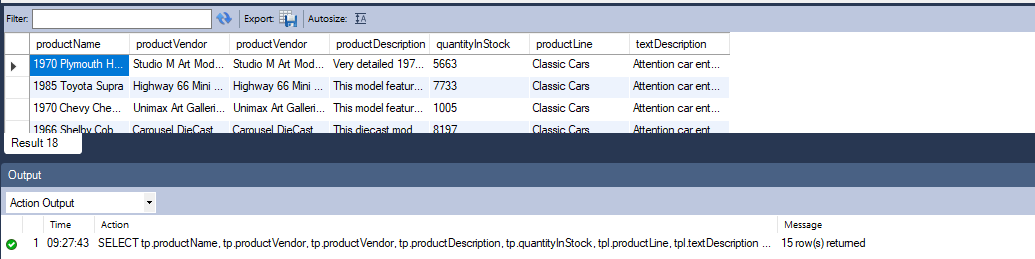


1. Modifikovati upit iz prethodnog zadatka tako da se selektuju samo proizvodi (**products**) koji u polju productDesription imaju string ‘model’ i proizvodne linije (**ProductLines**) koje u polju textDescription imaju riječ „CAR.“

SELECT tp.productName, tp.productVendor, tp.productVendor, tp.productDescription, tp.quantityInStock, tpl.productLine, tpl.textDescription

FROM t\_products tp, t\_productlines tpl

WHERE tp.productLine = tpl.productLine AND tp.productDescription LIKE "%CAR%"

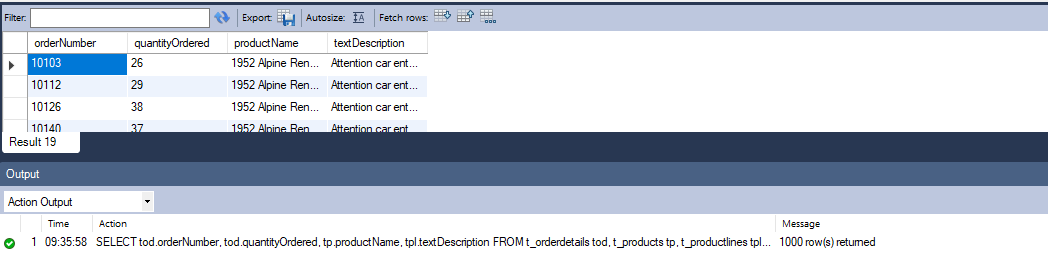


1. Povezati tabele **orderDetails**, **products** i **productLines**. Napraviti upit koji prikazuje iz tabele **orderDetails**: **orderNumber** i **quantityOrdered**, iz tabele **products productName** i iz tabele **ProductLines** polje **textDescription**.

SELECT tod.orderNumber, tod.quantityOrdered, tp.productName, tpl.textDescription

FROM t\_orderdetails tod, t\_products tp, t\_productlines tpl

WHERE tod.productCode = tp.productCode AND tp.productLine = tpl.productLine



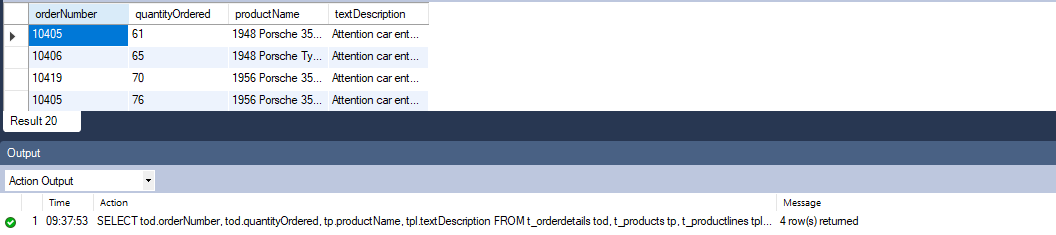
1. Modifikovati prethodni upit tako da pokazuje samo stavke kod kojih je naručena količina (**quantityOrdered**) veća od 60 a ime proizvoda (**productName**) sadrži string “Porsche”. Definisati sortiranje po koloni **productName.**

SELECT tod.orderNumber, tod.quantityOrdered, tp.productName, tpl.textDescription

FROM t\_orderdetails tod, t\_products tp, t\_productlines tpl

WHERE tod.productCode = tp.productCode AND tp.productLine = tpl.productLine AND tod.quantityOrdered > 60 AND tp.productName LIKE "%Porsche%"

ORDER BY productName



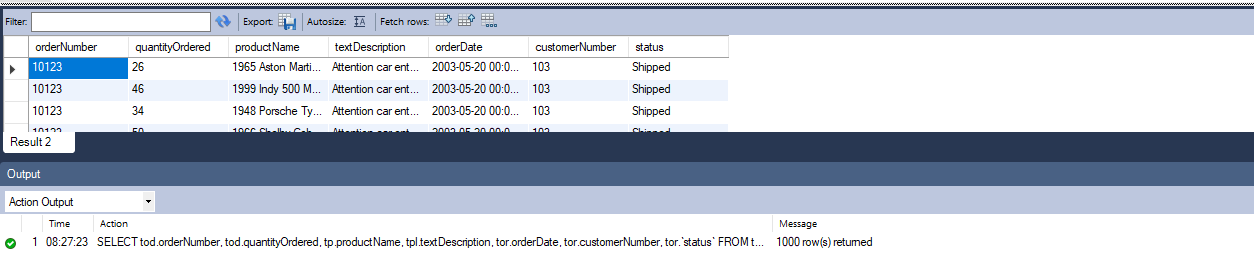
1. Upit iz zadatka 7 proširiti povezivanjem tabele **orders**. Iz ove tabele prikazati **orderDate, customerNumber** i **status** , a iz preostale 3 iste kolone kao u upitu 7. Sortirati podatke po koloni **customerNumber**.

SELECT tod.orderNumber, tod.quantityOrdered, tp.productName, tpl.textDescription, tor.orderDate, tor.customerNumber, tor.`status`

FROM t\_orderdetails tod, t\_products tp, t\_productlines tpl, t\_orders tor

WHERE tod.productCode = tp.productCode AND tp.productLine = tpl.productLine AND tod.orderNumber = tor.orderNumber

ORDER BY tor.customerNumber



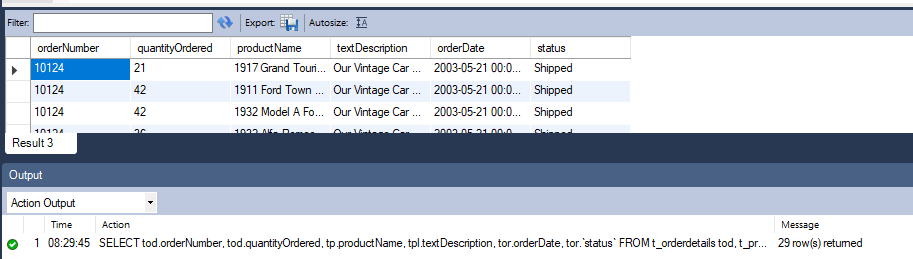
1. Modifikovati upit iz prethodnog zadatka tako da prikazuje podatke samo o kupcu čiji je broj 112. Iz liste kolona izbrisati **customerNumber**

SELECT tod.orderNumber, tod.quantityOrdered, tp.productName, tpl.textDescription, tor.orderDate, tor.`status`

FROM t\_orderdetails tod, t\_products tp, t\_productlines tpl, t\_orders tor

WHERE tod.productCode = tp.productCode AND tp.productLine = tpl.productLine AND tod.orderNumber = tor.orderNumber AND tor.customerNumber = 112

ORDER BY tor.customerNumber



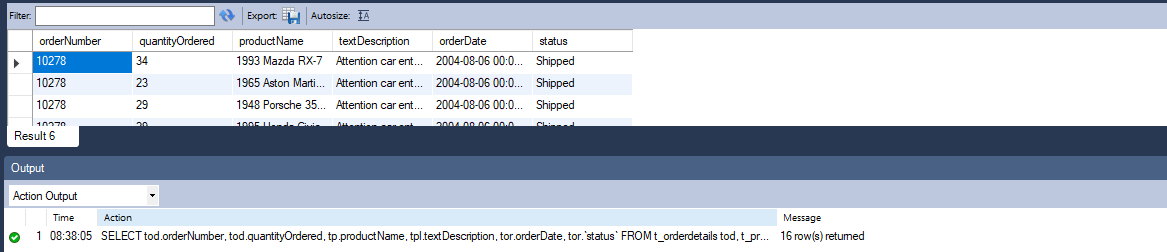
1. Modifikovati upit iz prethodnog zadatka tako da prikazuje podatke o narudžbama  
   napravljenim poslije 1.8.2003

SELECT tod.orderNumber, tod.quantityOrdered, tp.productName, tpl.textDescription, tor.orderDate, tor.`status`

FROM t\_orderdetails tod, t\_products tp, t\_productlines tpl, t\_orders tor

WHERE tod.productCode = tp.productCode AND tp.productLine = tpl.productLine AND tod.orderNumber = tor.orderNumber AND tor.customerNumber = 112 AND tor.orderDate > "2003-08-01"

ORDER BY tor.customerNumber



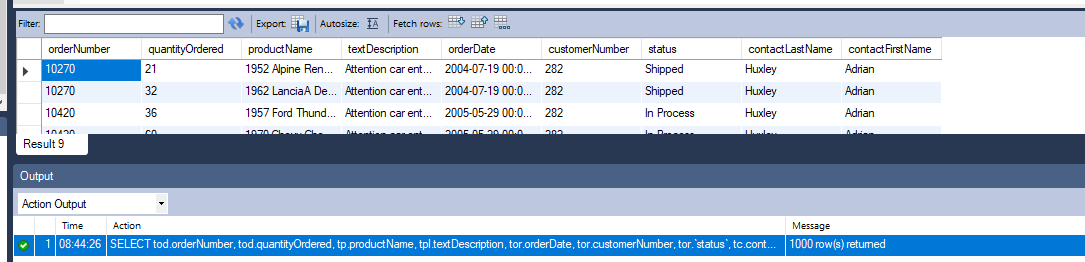
1. Modifikovati upit iz zadatka 9 tako što povezujemo i tabelu **customers**. **Iz ove tabele  
   prikazati kolone contactLastName i contactFirstName.** Sortirati po istim kolonama.

SELECT tod.orderNumber, tod.quantityOrdered, tp.productName, tpl.textDescription, tor.orderDate, tor.customerNumber, tor.`status`, tc.contactLastName, tc.contactFirstName

FROM t\_orderdetails tod, t\_products tp, t\_productlines tpl, t\_orders tor, t\_customers tc

WHERE tod.productCode = tp.productCode AND tp.productLine = tpl.productLine AND tod.orderNumber = tor.orderNumber AND tor.customerNumber = tc.customerNumber

ORDER BY tc.contactFirstName, tc.contactLastName



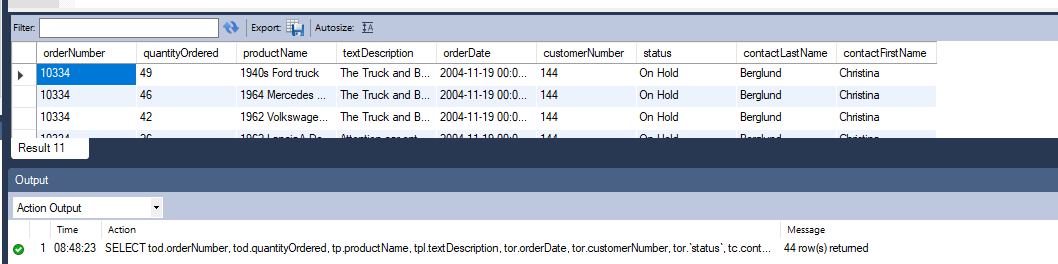
1. Modifikovati upit iz zadatka 12 tako da prikazuje samo narudžbe u statusu 'On Hold'.

SELECT tod.orderNumber, tod.quantityOrdered, tp.productName, tpl.textDescription, tor.orderDate, tor.customerNumber, tor.`status`, tc.contactLastName, tc.contactFirstName

FROM t\_orderdetails tod, t\_products tp, t\_productlines tpl, t\_orders tor, t\_customers tc

WHERE tod.productCode = tp.productCode AND tp.productLine = tpl.productLine AND tod.orderNumber = tor.orderNumber AND tor.customerNumber = tc.customerNumber AND tor.`status` = "On Hold"

ORDER BY tc.contactFirstName, tc.contactLastName



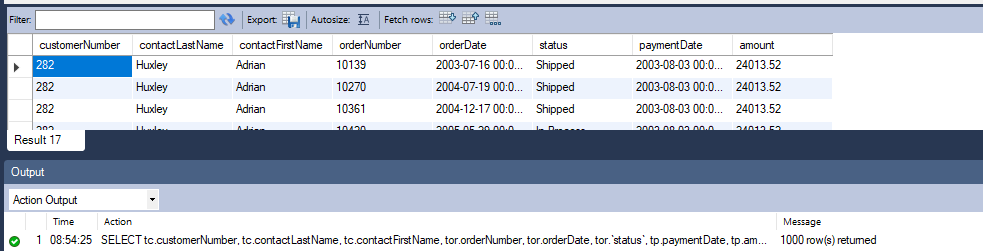
1. Povezati tabele **customers, orders i payments**. Prikazati kolone: **Customers->customerNumber, contactLastname, contactFirstname, orders ->orderNumber, orderDate,status, payment->paymentDate, amount.** Sortirati poi menu i prezimenu kupca.

SELECT tc.customerNumber, tc.contactLastName, tc.contactFirstName, tor.orderNumber, tor.orderDate, tor.`status`, tp.paymentDate, tp.amount

FROM t\_customers tc, t\_orders tor, t\_payments tp

WHERE tp.customerNumber = tc.customerNumber AND tor.customerNumber = tc.customerNumber

ORDER BY tc.contactFirstName, tc.contactLastName;



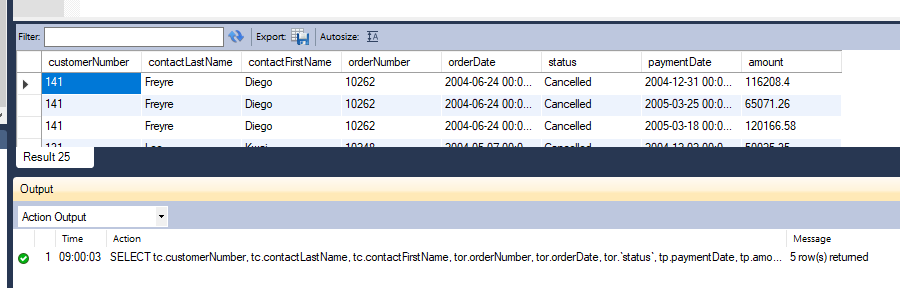
1. Modifikovati prethodni upit tako da prikazuje samo plaćanja izvršena poslije 1.12.2004 i ona kod kojih je iznos veći od 50000, a za narudžbe u status “Cancelled”

SELECT tc.customerNumber, tc.contactLastName, tc.contactFirstName, tor.orderNumber, tor.orderDate, tor.`status`, tp.paymentDate, tp.amount

FROM t\_customers tc, t\_orders tor, t\_payments tp

WHERE tp.customerNumber = tc.customerNumber AND tor.customerNumber = tc.customerNumber AND tp.paymentDate > "2004-12-01" AND tp.amount > 50000 AND tor.`status` = "Cancelled"

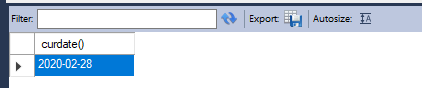
ORDER BY tc.contactFirstName, tc.contactLastName;



## Vježba 6 – SQL Funkcije

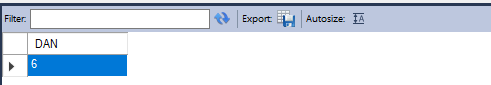
1. Napiši upit koji će prikazati tekući datum

SELECT curdate();



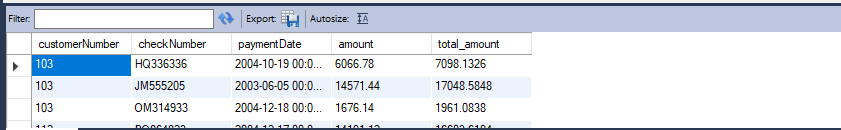
1. Napiši upit koji će ispisati tekući dan

SELECT dayofweek(curdate()) "DAN";



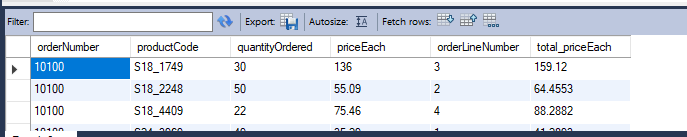
1. Napiši upit koji će iz tabele **payments** ispisati sve postojeće kolone i dadati novu koja se računa kao **amount** uvećan za PDV (**amount**\*1.17). Kolonu nazvati **total\_amount**.

SELECT \*, amount\*1.17 "total\_amount" FROM t\_payments;



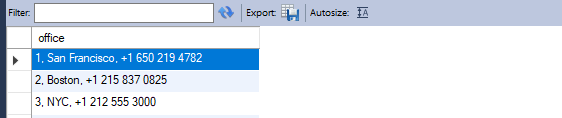
1. Napiši upit koji će iz tabele **orderDetails** ispisati sve postojeće kolone i dadati novu koja se računa kao **priceEach** uvećan za PDV (**priceEach** \*1.17). Kolonu nazvati **total\_ priceEach**

SELECT \*, priceEach\*1.17 "total\_priceEach" FROM t\_orderdetails;



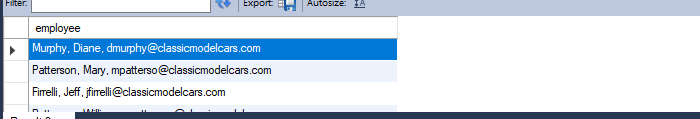
1. Napiši upit koji iz tabele **offices** kreira novu kolonu koju čine kolone **officeCode**, **city** i **phone**, razdvojene zarezom. Novu kolonu nazvati **office**. Npr: 1, San Francisco, +1 650 219 4782

SELECT concat(officeCode, ", ", city, ", ", phone) "office" FROM t\_offices;



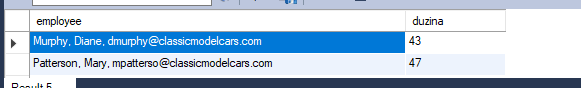
1. Napiši upit koji iz tabele **employees** kreira novu kolonu koju čine kolone **lastname**, **firstname** i **email**, razdvojene zarezom. Novu kolonu nazvati **employee**. Npr: Murphy, Diane, dmurphy@classicmodelcars.com

SELECT concat(lastName, ", ", firstName, ", ", email) "employee" FROM t\_employees;



1. Napiši upit koji proširuje prethodni upit kolonom koja prikazuje broj karaktera nove kolone.

SELECT concat(lastName, ", ", firstName, ", ", email) AS employee, length(concat(lastName, ", ", firstName, ", ", email)) "duzina" FROM t\_employees;



1. Napiši upit koji prikazuje **orderNumber** i dan kada je roba isporučena (**shippedDate**). Kolonu nazvati **shippedDay**. Podatke čitati iz tabele **Orders**

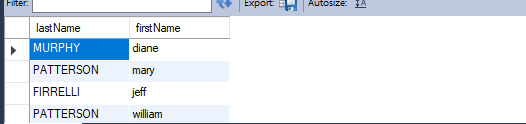
SELECT concat(orderNumber, ", ", dayname(shippedDate)) AS shippedDay FROM t\_orders;

1. Napiši upit koji prikazuje **orderNumber** i mjesec kada je roba isporučena (**shippedDate**). Kolonu nazvati **shippedMonth**. Podatke čitati iz tabele **Orders**

SELECT concat(orderNumber, ", ", monthname(shippedDate)) AS shippedDay FROM t\_orders;

1. Napiši upit koji iz tabele **employees** ispisuje kolone **lastname** i **firstname, ali tako da je lastname ispisano** velikim slovima, a firstname malim slovima

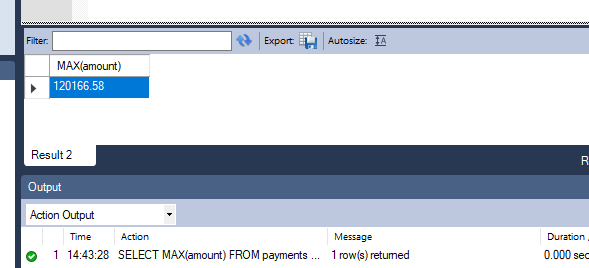
SELECT upper(lastName) lastName, lower(firstName) firstName FROM t\_employees;



## Vježba 7 – Group by, Having + funkcije

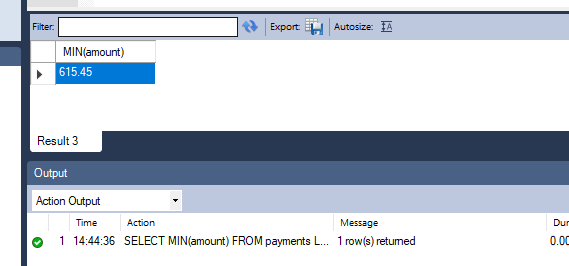
1. Napisati SQL upit koji će u tabeli t\_payments naći maximalnu vrijednost kolone amount?

SELECT MAX(amount) FROM payments;



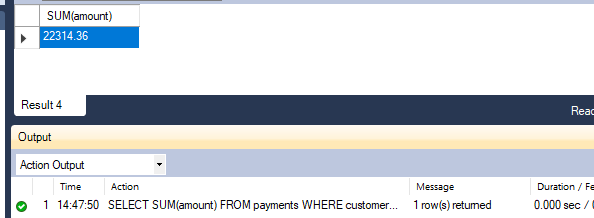
1. Napisati SQL upit koji će u tabeli t\_payments naći minimalnu vrijednost kolone amount?

SELECT MIN(amount) FROM payments;



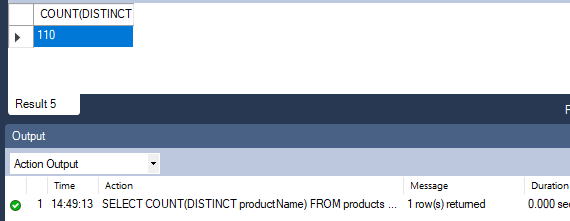
1. Napisati SQL upit koji će u tabeli t\_payments naći sumu vrijednosti kolone amount gdje je customerNumber = 103;

SELECT SUM(amount) FROM payments WHERE customerNumber = 103;



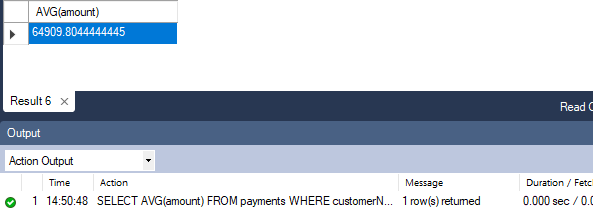
1. Napisati upit koji će prebrojati sve proizvode iz tabele products.

SELECT COUNT(DISTINCT productName) FROM products;



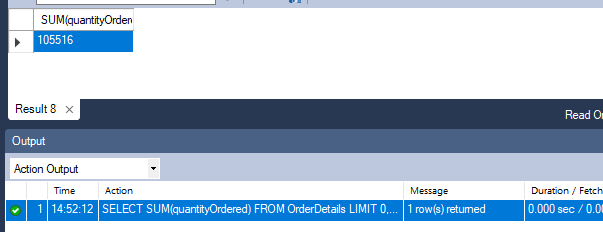
1. Napisati SQL upit koji će u tabeli t\_payments naći prosjek kolone amount gdje je customerNumber = 124;

SELECT AVG(amount) FROM payments WHERE customerNumber = 124;



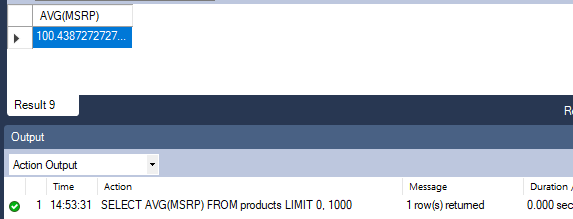
1. Naći ukupan broj naručenih proizvoda( quantityOrdered) iz tabele OrderDetails.

SELECT SUM(quantityOrdered) FROM OrderDetails;



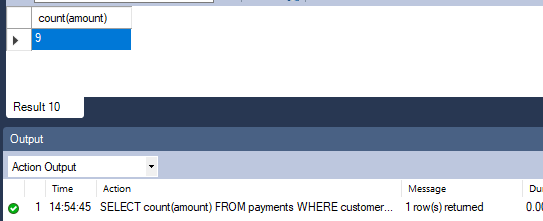
1. Naći prosječnu cijenu svih proizvoda iz tabele products

SELECT AVG(MSRP) FROM products;



1. Napisati SQL upiti koji u tabeli t\_paymens izbroji broj redova kolone amount gdje je customerNumber=124;

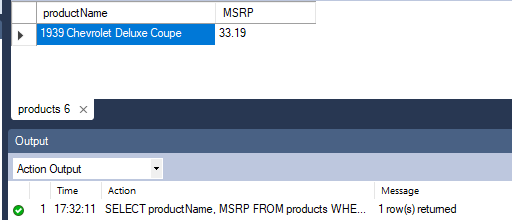
SELECT count(amount) FROM payments WHERE customerNumber = 124;



1. Napisati naziv najjeftinijeg proizvoda iz tabele products

SELECT productName, MSRP FROM products WHERE MSRP = (SELECT MIN(MSRP) FROM products);

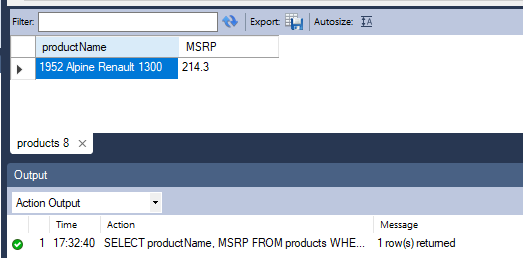
(korištena maloprodajna cijena – MSRP)



1. Napisati naziv najskupljeg proizvoda iz tabele products.

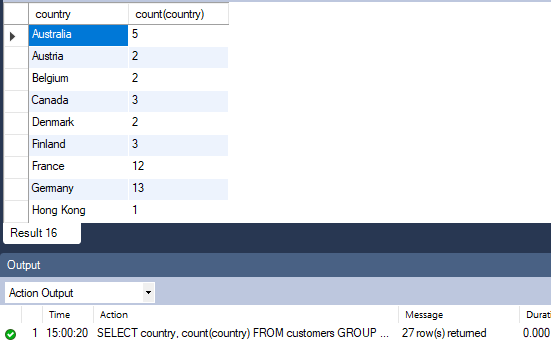
SELECT productName, MSRP FROM products WHERE MSRP = (SELECT MAX(MSRP) FROM products);

(korištena maloprodajna cijena – MSRP)



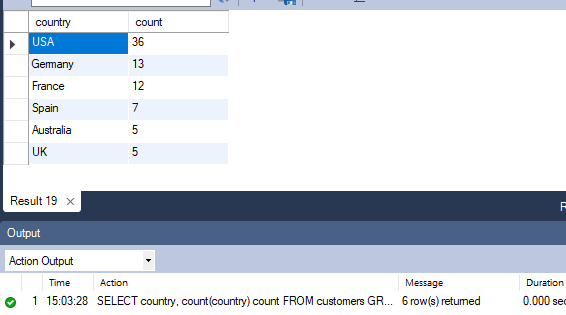
1. Napisati upit koji će izlistati koliko je kupaca u svakoj zemlji, grupisati po zemljama.

SELECT country, count(country) FROM customers GROUP BY country;



1. Napisati upit koji će izlistati broj kupaca za svaku zemlju iz koje dolaze, sortirati u opadajućem redoslijedu , ali uključiti samo one zemlje sa više od 5 kupaca.

SELECT country, count(country) count FROM customers GROUP BY country HAVING count(country)>=5 ORDER BY count DESC



1. Napraviti upit koji će omogućiti ispis imena i prezimena zaposlenika koji je registrovao više od 10 narudžbi.

SELECT firstName, lastName, count(DISTINCT ordernumber) orderCount

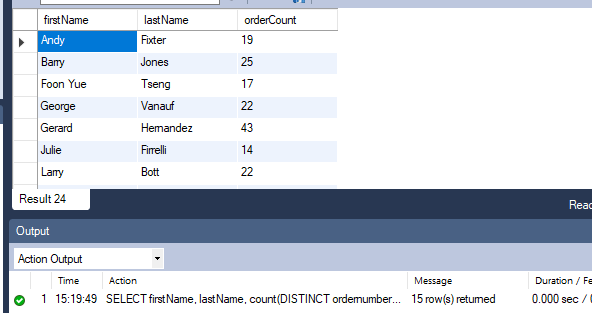
FROM employees, orders, customers

WHERE orders.customerNumber = customers.customerNumber

AND customers.salesRepEmployeeNumber = employees.employeeNumber

GROUP BY firstName, lastName

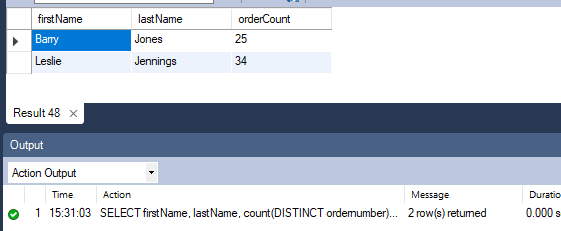
HAVING orderCount > 10;



1. Napraviti upit koji će omogućiti ispis imena i prezimena zaposlenika koji je registrovao više od 25 narudžbi i samo ako se zaposlenik preziva Davolio ili Fuller

Ne postoji niko sa prezimenom „Davolio“ ili „Fuller“ niti sa sličnim prezimenima, tako da sam koristio druga prezimena.

SELECT firstName, lastName, count(DISTINCT ordernumber) orderCount   
FROM employees, orders, customers   
WHERE orders.customerNumber = customers.customerNumber   
AND customers.salesRepEmployeeNumber= employees.employeeNumber   
AND (lastName = "Jones" OR lastName = "Jennings")   
GROUP BY firstName, lastName HAVING orderCount >= 25;

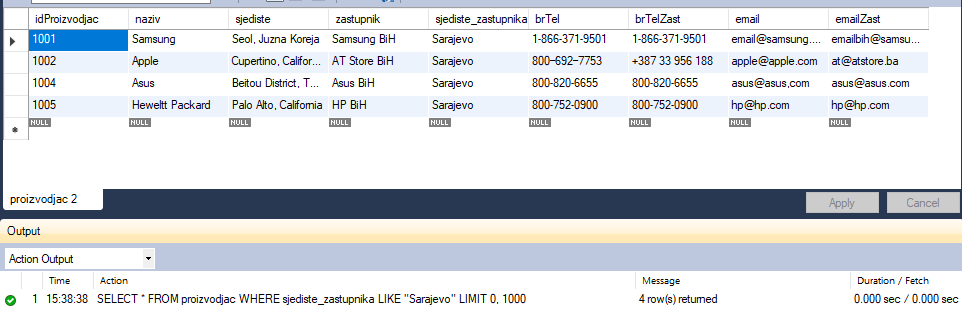


# Mogući upiti baze podataka

U ovom dijelu ćemo opisati nekoliko mogućih upita koji korisnik može imati vezano za ovu bazu podataka, te dati konkretan primjer rezultata.

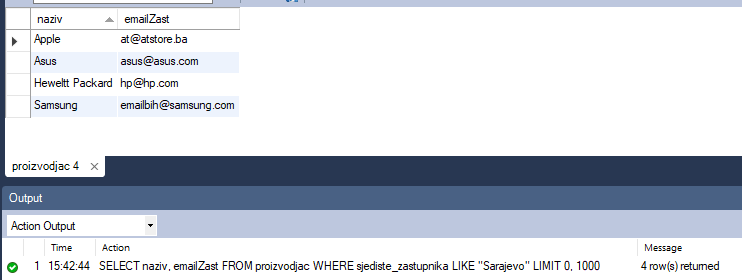
1. SELECT \* FROM proizvodjac WHERE sjediste\_zastupnika LIKE "Sarajevo";

Korisnik traži sve informacije vezane za proizvođače čiji su zastupnici u Sarajevu.



1. SELECT naziv, emailZast FROM proizvodjac WHERE sjediste\_zastupnika LIKE "Sarajevo";

Konkretniji primjer sličan prvom se može napisati ovako. Korisnik traži naziv proizvođača kao i email zastupnika proizvođača koji se nalazi u Sarajevu.



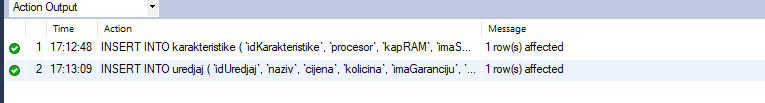
1. Sljedeća dva upita su potrebna da bi se upisao novi uređaj i njegove karakteristike u bazu.

INSERT INTO karakteristike (`idKarakteristike`, `procesor`, `kapRAM`, `imaSSD`, `imaHDD`, `SSDKap`, `HDDKap`, `grafKart`, `portovi`, `imaWIFI`, `imaBT`, `velEkran`, `operSis`)

VALUES (3008, "i7-9750H", 32, 1, 0, 1024, 0, "NVIDIA GeForce RTX 2080", "HDMI, DisplayPort, ethernet, audio, 4x USB, USB-C, Thunderbolt", 1, 1, 15.6, "Windows 10");

INSERT INTO uredjaj (`idUredjaj`, `naziv`, `cijena`, `kolicina`, `imaGaranciju`, `trajanjeGaran`, `Karakteristike\_idKarakteristike`, `Proizvodjac\_idProizvodjac`)

VALUES (2008, "Predator Triton 500", 2699.99, 2, 1, 24, 3008, 1003);



Važno je napomenuti da moramo najprije upisati karakteristike, a tek onda uređaj iz razloga što uređaj ima u svojoj tabeli primarni ključ tabele karakteristike. Ne možemo upisati ključ koji ne postoji u tabelu, tako da je to razlog.

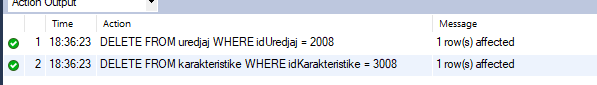
Slično i sa proizvođačem, da bi upisali uređaj novog proizvođača moramo najprije upisati novog proizvođača, karakteristike pa tek onda uređaj.

1. Sa brisanjem je drugačija situacija. Moramo tada obrisati red koji koristi ključ druge tabele, da bi obrisali i red iz te druge tabele. Npr. da bi obrisali neki uređaj iz baze, najprije moramo obrisati red iz tabele uređaj, a potom i red iz tabele karakteristike.

Razlog tome je to što ne možemo obrisati ključ neke tabele ukoliko je neka druga tabela koristi.

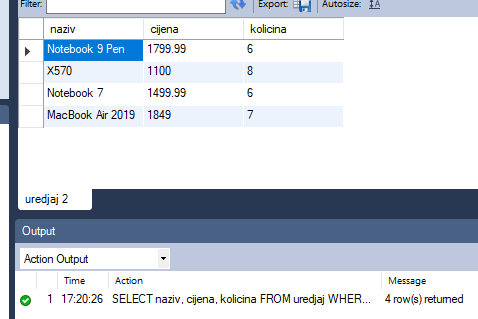
DELETE FROM uredjaj WHERE idUredjaj = 2008;

DELETE FROM karakteristike WHERE idKarakteristike = 3008;



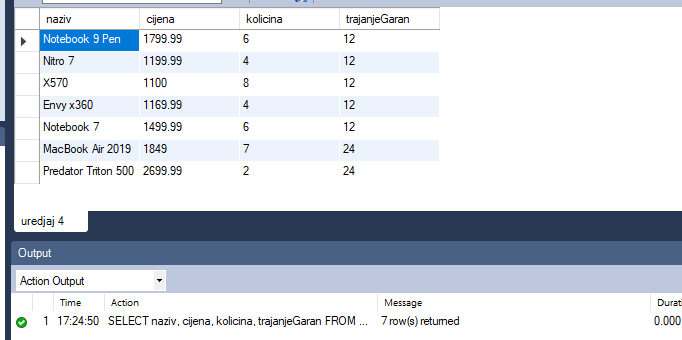
1. SELECT naziv, cijena, kolicina FROM uredjaj WHERE kolicina > 5;

Korisnik traži sve uređaje (njihova imena i cijene), kojeg ima više od 5 na stanju.



1. SELECT naziv, cijena, kolicina, trajanjeGaran FROM uredjaj WHERE trajanjeGaran BETWEEN 12 AND 24 AND cijena < 2700;

Korisnik traži sva imena, cijene i količine na stanju uređaja, kao i trajanje njihovih garancija koje su između 12 i 24 mjeseca, a da njihova cijena ne prelazi 2700.



1. SELECT pr.naziv Proizvodjac, ur.naziv Uredjaj, ur.cijena, ka.procesor, ka.kapRAM, ka.SSDKap

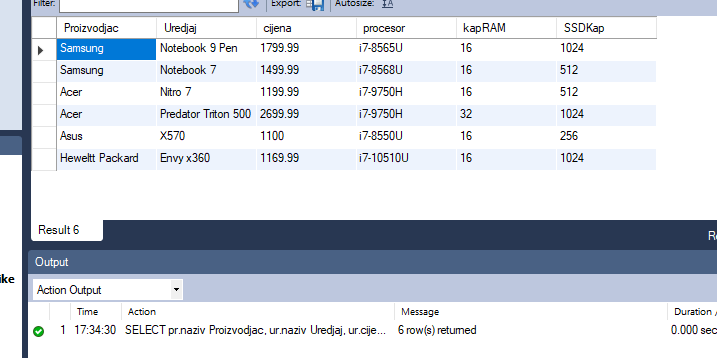
FROM uredjaj ur, karakteristike ka, proizvodjac pr

WHERE ur.Proizvodjac\_idProizvodjac = pr.idProizvodjac

AND ur.Karakteristike\_idKarakteristike = ka.idKarakteristike

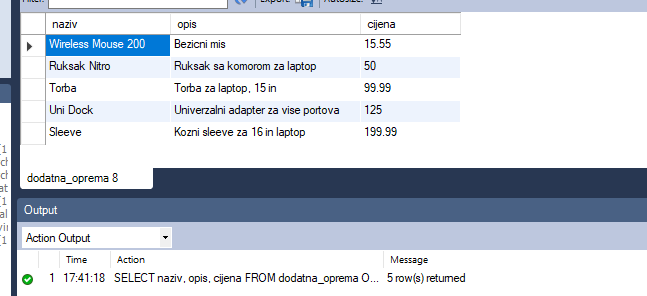
AND ka.operSis LIKE "Windows 10";

Korisnik traži željene informacije (proizvođač, naziv uređaja, cijenu, procesor, RAM i SSD) svih uređaja koji imaju Windows 10 kao operativni sistem.



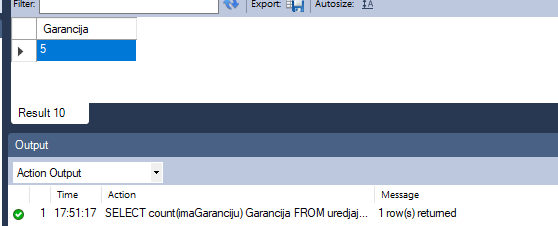
1. SELECT naziv, opis, cijena FROM dodatna\_oprema ORDER BY cijena;

Korisnik želi pregledati svu dodatnu opremu raspoloživu, s tim da želi ispisati rezultate u rastućem redoslijedu cijene.



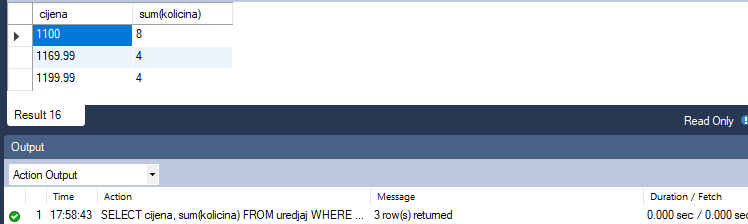
1. SELECT count(imaGaranciju) Garancija FROM uredjaj WHERE trajanjeGaran = 12;

Korisnik želi znati koliko ima uređaja koji imaju trajanje garancije 12 mjeseci.



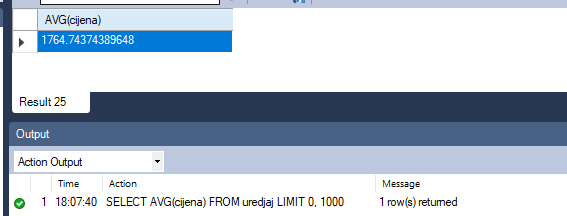
1. SELECT cijena, sum(kolicina) FROM uredjaj WHERE cijena < 1200 GROUP BY cijena;

Korisnik traži koliko ima uređaja na stanju za svaku cijenu ispod 1200



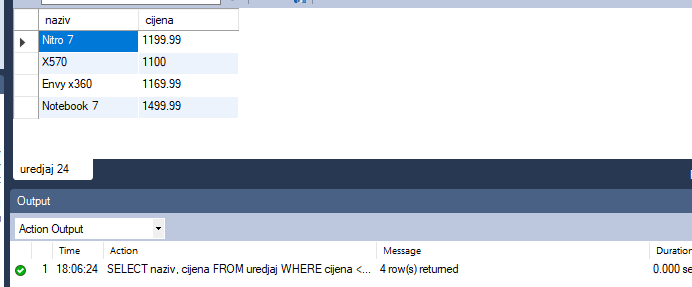
1. SELECT AVG(cijena) FROM uredjaj;

Korisnik provjerava prosječnu cijenu uređaja



1. SELECT naziv, cijena FROM uredjaj WHERE cijena < (SELECT AVG(cijena) FROM uredjaj);

Korisnik traži sve uređaje čija je cijena manja od prosječne.



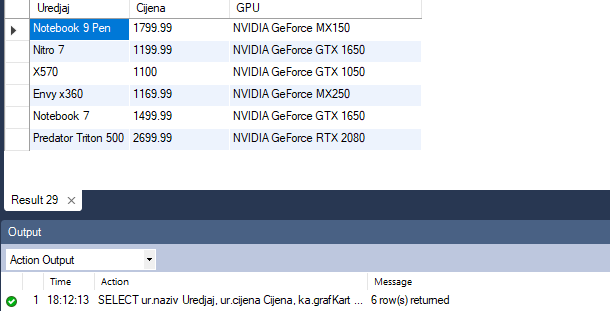
1. SELECT ur.naziv Uredjaj, ur.cijena Cijena, ka.grafKart GPU

FROM uredjaj ur, karakteristike ka

WHERE ur.Karakteristike\_idKarakteristike = ka.idKarakteristike

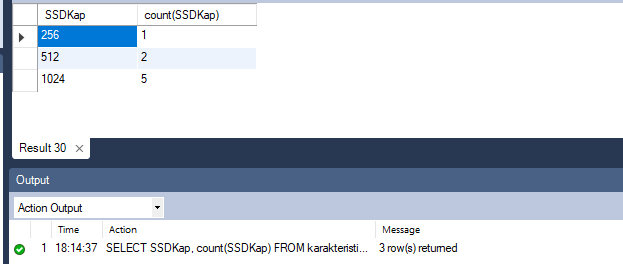
AND ka.grafKart LIKE "%NVIDIA%"

Korisnik traži sve uređaje koji imaju NVIDIA GPU, te njihove cijene.



1. SELECT SSDKap, count(SSDKap) FROM karakteristike GROUP BY SSDKap;

Korisnik želi znati koliko uređaja ima za svaki od dostupnih kapaciteta SSD-ova.



1. SELECT pr.naziv Proizvodjac, count(DISTINCT ur.idUredjaj) "Broj Uredjaja"

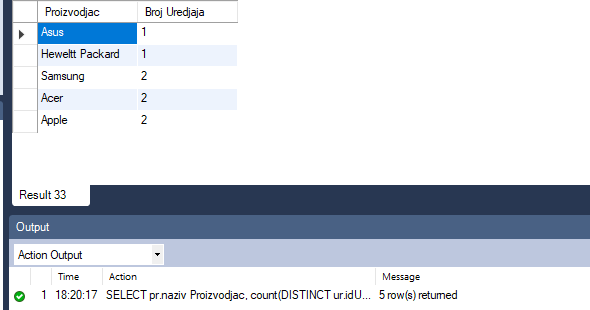
FROM proizvodjac pr, uredjaj ur

WHERE ur.Proizvodjac\_idProizvodjac = pr.idProizvodjac

GROUP BY pr.naziv

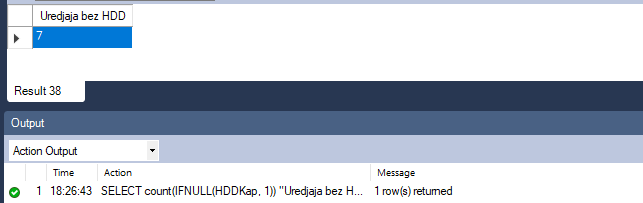
ORDER BY count(DISTINCT ur.idUredjaj) ASC;

Korisnik želi znati koliko svaki od proizvođača ima svoje uređaje u prodavnici koja koristi bazu.



1. SELECT count(IFNULL(HDDKap, 1)) "Uredjaja bez HDD" FROM karakteristike WHERE HDDKap IS NULL;

Korisnik traži koliko uređaja (ne količinu, već koliko unikatnih uređaja) ima koji nemaju HDD.



# Vježba 8 – implementacija sigurnosti podataka

Naša baza podataka sadrži vrijedne informacije koje sakupljamo i obrađujemo. Bilo bi veoma loše kada bi neko ko ne bi smio da manipuliše tim podacima to i učini.

Iz tog razloga korisnicima baze treba dati samo one privilegije upravljanju baze koje su potrebne za njihov posao.

Za našu bazu ćemo najprije definisati nekoliko korisnika i njihove privilegije, a zatim ćemo koristiti GRANT i REVOKE naredbe kako bi omogućili (odnosno onemogućili) korisnicima da obavljaju određene upite.

## Definisanje i kreiranje korisnika

Kao što smo ranije spomenuli, najprije trebamo opisati ko su korisnici koji će koristiti bazu.

Korisnici:

1. **Root**

Ovaj korisnik ima sve privilegije upravljanja nad bazom i on je ujedno i serverski korisnik. On je napravljen samim pravljenjem servera i ne može se mijenjati.

1. **Administrator**

Ovaj korisnik je sličan korisniku Root, s tim da se ne zove root. Njeg koristimo kako bi bili sigurni da uvijek imamo korisnika Root koji može popraviti stvari ukoliko izgubimo pristup korisniku Administraor. Jedino će se razlikovati po tome što će administrator upravljati samo šemom na kojoj se nalaze 4 tabele ove baze, a root ima privilegiju nad svime.

1. **Prodavač**

To je korisnik koji može da pogleda sve informacije u bazi, i da modificira neke (kao npr. količina na stanju)

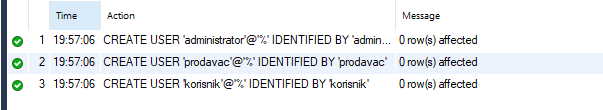
1. **Korisnik**

„Korisnik“ je krajnji korisnik baze. Kao što smo pisali na početku, ova baza je namijenjena da se koristi kao baza za neku online prodavnicu. To znači da trebaju da se ispisu podaci iz baze na internet stranici, a ovaj korisnik je upravo predstava toga.

On može samo da pročita šta je u bazi, ali ne može da mijenja niti jedan podatak.

Najprije kreiramo korisnike. Pomoću postojećeg *root* korisnika izvršavamo slijedeće naredbe:

CREATE USER 'administrator'@'%' IDENTIFIED BY 'administrator';  
CREATE USER 'prodavac'@'%' IDENTIFIED BY 'prodavac';  
CREATE USER 'korisnik'@'%' IDENTIFIED BY 'korisnik';



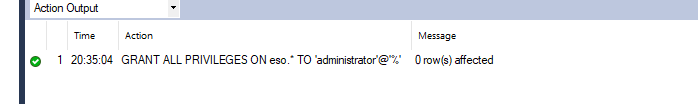
Nakon pokretanja u konzoli vidimo da su korisnici kreirani.

## Određivanje privilegija korisnicima

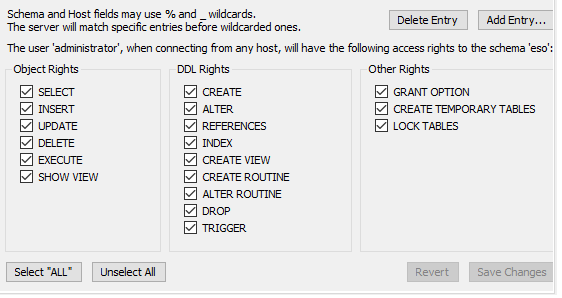
**Administrator**

Korisnik „administrator“ treba da može upravljati svime vezanim za šemu u kojoj su tabele ove baze. Takve privilegije možemo dodijeliti sljedećom naredbom:

GRANT ALL PRIVILEGES ON eso.\* TO 'administrator'@'%' WITH GRANT OPTION;



Pomoću MySQL workbench-a možemo provjeriti sve privilegije date korisniku.

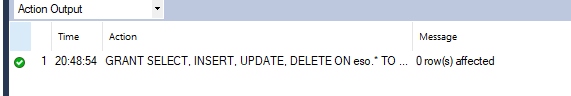


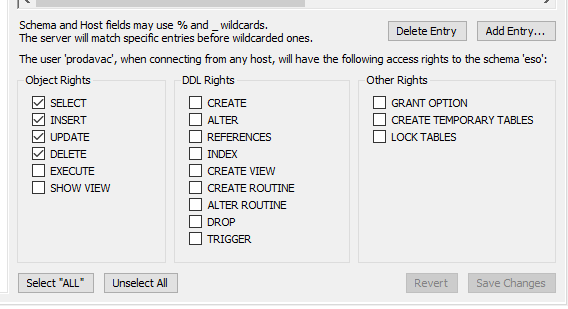
**Prodavač**

Korisnik „prodavac“ treba da ima privilegije samo da modifikuje podatke u tabelama. Razlog ove privilegije je da prodavač može promijeniti trenutno količinsko stanje kada se ono promijeni. Količinsko stanje se može promijeniti kada prodavač proda neki uređaj u poslovnici (ako se baza bude koristila u poslovnici, pored online prodavnice).

Još jedan od razloga zašto prodavaču treba privilegija nad podacima je to što prodavač može unositi nove uređaje ili brisati neke koje više prodavnica ne prodaje.

GRANT SELECT, INSERT, UPDATE, DELETE ON eso.\* TO 'prodavac'@"%";



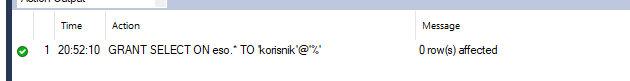


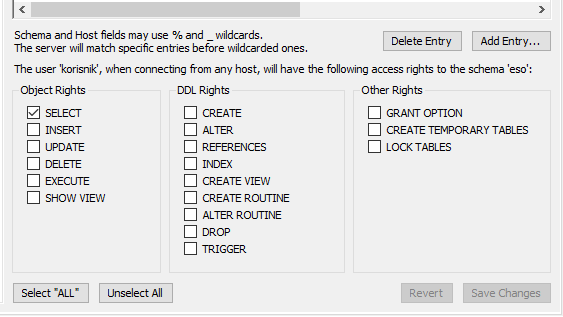
**Korisnik**

Korisnik „korisnik“ treba da ima privilegiju samo da pogleda informacije u tabelama, ali ne da ih i mijenja.

On je korisnik koji će se primjenjivati na internet stranici koja čita podatke iz baze.

GRANT SELECT ON eso.\* TO 'korisnik'@'%';





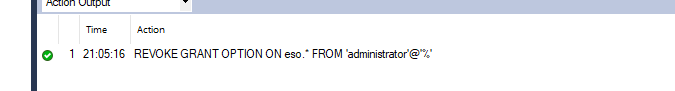
Sada su naša nova 3 korisnika potpuna i mogu da koriste bazu kako im je dopušteno. Ovim smo onemogućili neovlaštenim korisnicima da naruše stabilnost naše baze.

Naravno, nijedan sistem nije u potpunosti siguran, pa time ni ovaj. Problem može nastati ako, recimo, neko dođe do šifre administratora pa naruši stabilnost. Ovakav problem se može izbjeći mijenjanjem šifre korisnika, pravljenjem sigurnosne kopije baze i osiguravanjem sistema i platformi preko kojih se pristupa na bazu... ali to je oblast za neku drugu vježbu.

**REVOKE**

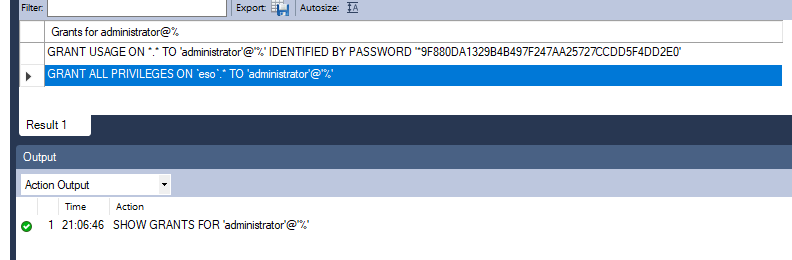
Ukoliko kasnije primjetimo da je privilegija neke naredbe suvišna kod nekog korisnika, možemo je ukloniti pomoću REVOKE naredbe.

Primjer: REVOKE GRANT OPTION ON eso.\* FROM 'administrator'@'%';



Još jedan način da provjerimo privilegije je naredbom SHOW GRANTS FOR \_\_\_

SHOW GRANTS FOR 'administrator'@'%';



Kao što vidimo, korisnik administrator ima sve privilegije osim GRANT OPTION nad bazom „eso“ (koja uključuje 4 tabele).

1. Autor ovog rada je siguran da je svako kopirao ovo pravilo iz praktikuma dobivenog na času, te je i on sam to isto uradio. [↑](#footnote-ref-1)
2. Poznato nam je da dva uređaja mogu imati iste karakteristike, ali s obzirom da mi zapravo trebamo da pratimo koje to karakteristike ima svaki uređaj, zbog logičkih i razloga efikasnosti baze izdvojili smo podatke o svakom uređaju u zasebnu tabelu. [↑](#footnote-ref-2)
3. Ovo ćemo još obraditi u nastavku rada. [↑](#footnote-ref-3)