Josip Milković

Fran Barba

Bruno Rebić

Sveučilište Jurja Dobrile u Puli

Fakultet informatike u Puli

DOKUMENTACIJA UZ PROJEKTNI ZADATAK

„ORGANIZACIJA U PROIZVODNJI METALNIH KONSTRUKCIJA“

TIM 4

Loren Bažon

Morena Martan

Smjer : Informatika

Kolegij : Baze podataka I

Mentor : doc. Dr. sc. Goran Oreški

Pula,

svibanj

, 202

4

. godina



Sadržaj

[1. UVOD 3](#_Toc167706644)

[2. OPIS POSLOVNOG PROCESA 3](#_Toc167706645)

[3. ENTITY RELATIONSHIP (ER) DIJAGRAM 3](#_Toc167706646)

[3.1. OPIS ER DIJAGRAMA 4](#_Toc167706647)

[4. RELACIJSKI MODEL (SHEME) 5](#_Toc167706648)

[5. EER DIJAGRAM (MYSQL) 6](#_Toc167706649)

[6. TABLICE 6](#_Toc167706650)

[6.1. TABLICA klijent 6](#_Toc167706651)

[6.2. TABLICA zaposlenik 7](#_Toc167706652)

[6.3. TABLICA projekt 7](#_Toc167706653)

[6.4. TABLICA segment 8](#_Toc167706654)

[6.5. TABLICA faze\_izrade 8](#_Toc167706655)

[6.6. TABLICA tip\_materijala 9](#_Toc167706656)

[6.7. TABLICA materijal 9](#_Toc167706657)

[6.8. TABLICA proizvodnja 9](#_Toc167706658)

[6.9. TABLICA utrosak\_sati 11](#_Toc167706659)

[6.10. TABLICA utrosak\_materijala 11](#_Toc167706660)

[6.11. TABLICA primka\_materijala 12](#_Toc167706661)

[6.12. TABLICA alat 12](#_Toc167706662)

[6.13. TABLICA lager\_materijala 13](#_Toc167706663)

[6.14. TABLICA isporuka 13](#_Toc167706664)

[6.15. TABLICA zaduženje alata 14](#_Toc167706665)

[7. UPITI 16](#_Toc167706666)

[7.1. UPIT 1 16](#_Toc167706667)

[7.2. UPIT 2 17](#_Toc167706668)

[7.3. UPIT 3 18](#_Toc167706669)

[7.4. UPIT 4 20](#_Toc167706670)

[7.5. UPIT 5 21](#_Toc167706671)

[7.6. UPIT 6 22](#_Toc167706672)

[7.7. UPIT 7 23](#_Toc167706673)

[7.8. UPIT 8 25](#_Toc167706674)

[7.9. UPIT 9 26](#_Toc167706675)

[7.10. UPIT 10 27](#_Toc167706676)

[7.11. UPIT 11 29](#_Toc167706677)

[7.12. UPIT 12 31](#_Toc167706678)

[7.13. UPIT 13 33](#_Toc167706679)

# UVOD

Ovaj projekt je od svoje početne verzije prošao kroz značajne promjene i unapređenja. Kako smo napredovali sa stjecanjem novih znanja i vještina tokom semestra, stalno smo uvodili poboljšanja koja su optimizirala strukturu i funkcionalnost elemenata našeg projekta. Zbog nemogućnosti okupljanja svih članova tima uslijed poslovnih i drugih obaveza, proces izrade projekta je koordiniran putem GitHuba, Whatsappa, Zooma i ostalih komunikacijskih alata.

Izradili smo plan i model organizacije proizvodnje metalnih konstrukcija, s naglaskom na precizno opisivanje svih proizvodnih procesa i konceptualnog modela, prikazanog putem ER dijagrama. Naš projekt obuhvaća samo dio stvarnog poslovanja, pažljivo reducirajući i specificirajući ključne aspekte za potrebe ovog zadatka. Krajnji cilj je prikazati kompletan proces od nabave sirovina, preko proizvodnje, do isporuke gotovih metalnih proizvoda.

Svaki korak proizvodnje, uključujući tablice, unose podataka, upite i ograničenja, temeljito je dokumentiran u narednim stranicama. EER dijagram, odnosno logička shema organizacije proizvodnje, generiran je u MySQL Workbench-u pomoću Reverse Engineering opcije. Svi nazivi zaposlenika, adrese, OIB-ovi i brojevi mobitela su generirani nasumično pomoću online alata, dok su nazivi alata I klijenata preuzeti iz stvarne industrijske prakse.

# OPIS POSLOVNOG PROCESA

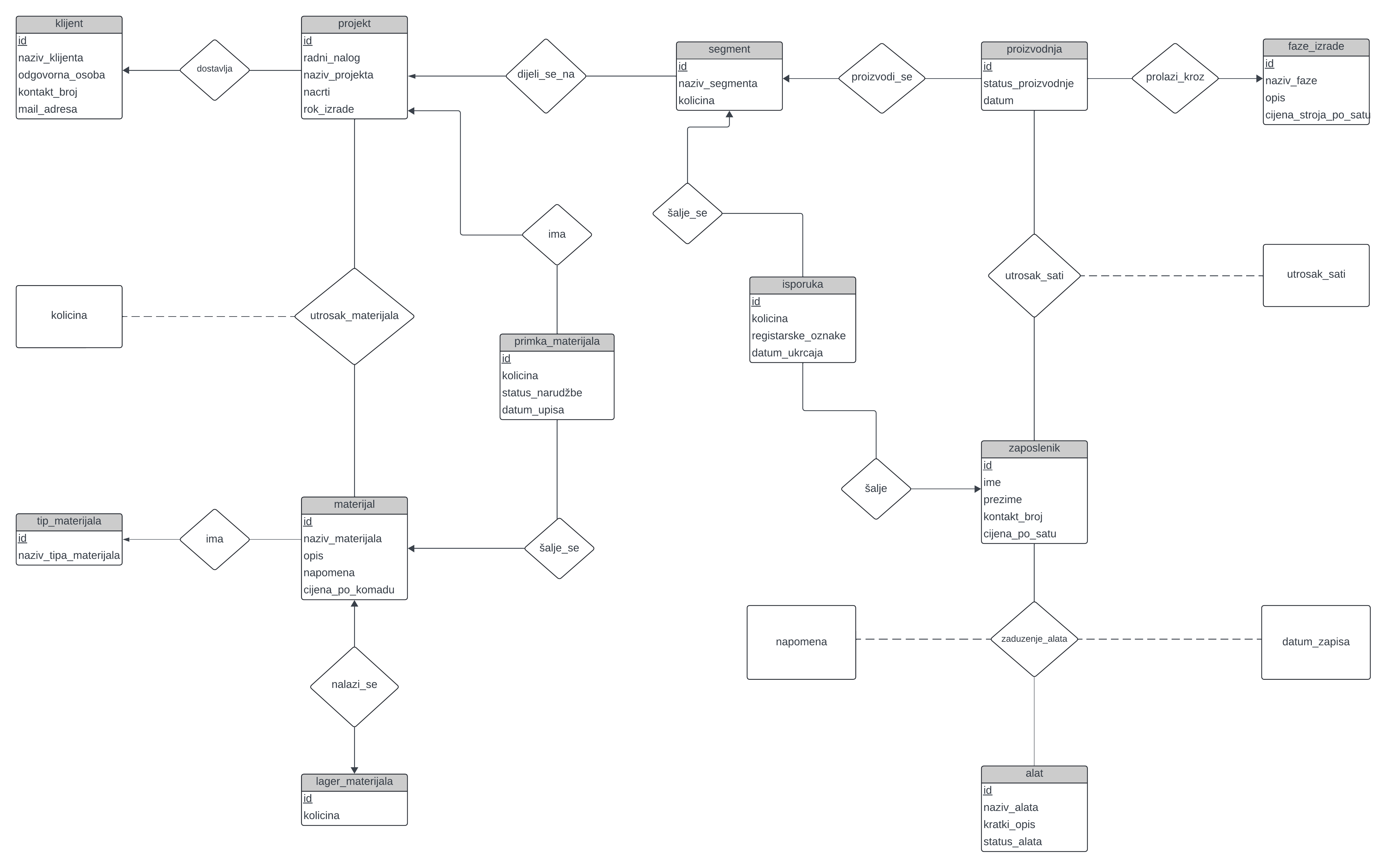
U proizvodnji sve kreće od klijenta koji daje neki projekt koji je potrebno izraditi s potrebnom dokumentacijom (nacrtima). Za potrebe ovog projekta smatra se da klijent ima svu potrebnu dokumentaciju, iako u praksi ne mora biti tako. Za klijenta pratimo naziv, odgovornu osobu, kontakt broj i mail adresu. Za projekt se prati radni nalog, naziv, nacrt i rok izrade.

Nakon dobivanja projekta, inžinjeri proučavaju projekt te daju informacije o potrebnim materijalima za proizvodnju. Za materijale se prati naziv, opis, određena napomena i cijena po komadu. Računa se utrošak materijala za projekt, te se prati tip materijala, kao i njegovo količinsko stanje na lageru. Tada se radi dokument koji se zove primka, te se materijali naručuju i skladište do početka proizvodnje projekta.

Projekt se tada dijeli na segmente. Svakom segmentu se prati naziv i količina. Da bi određeni zaposlenik, kojem pratimo ime, prezime, kontakt broj i cijenu po satu, mogao započeti proizvodnju određenog segmenta on mora zadužiti određene alate. Tu pratimo sve alate koje imamo, te se za svaki alat prati naziv, kratki opis i status. Pri zaduženju svakog alata pratimo datum te kratku napomenu zaposlenika. U proizvodnji se prati status proizvodnje te datum izrade. Također u proizvodnji imamo različite faze izrade kojima se prati naziv, kratki opis te cijena stroja po satu. Nakon završetka proizvodnje, moramo izračunati cijenu rada zaposlenika. To radimo na način da pratimo utrošak sati rada određenog zaposlenika. Na kraju zaposlenik šalje segment koji je proizveo na isporuku u kojoj se prati količina, registarske oznake vozila kojim se šalje te datum ukrcaja.

# ENTITY RELATIONSHIP (ER) DIJAGRAM

Slijedeći ER dijagram detaljno i pregledno opisuje sve skupove - entiteta (njihove atribute) kao i skupove - veza između njih. Kardinalnost mapiranja (strelice) predstavljaju koliko drugih entiteta može biti povezano s entitetom preko određenog skupa – veza.



*Slika 1/ ER dijagram za našu bazu*

## OPIS ER DIJAGRAMA

* **KLIJENT** dostavlja **PROJEKT** (klijent može dostaviti više projekata, dok je jedan projekt specifično dostavljen od strane jednog klijenta) **jedan na više**
* **PROJEKTU** trebaju određeni **MATERIJALI** (jedan projekt treba više materijala za korištenje, a jedan materijal može biti korišten kod više projekata) **više na više**
* Stvara se dodatni entitet **UTROSAK\_MATERIJALA** gdje se prati količina utrošenog materijala, a primarni ključevi entiteta **PROJEKT** I **MATERIJAL** se referenciraju kao strani ključ za UTROSAK\_MATERIJALA
* **TIP MATERIJALA** opisuje **MATERIJAL** (jedan materijal se sastoji od jednog tipa materijala, dok isti taj tip materijala može biti kod više različitih materijala) **jedan na više**
* **MATERIJAL** se nalazi na **LAGERU** (jedan materijal se nalazi na jednom lageru, taj lager prati količinu i vezan je za taj jedan materijal) **jedan na jedan**
* **MATERIJAL** se šalje na **PRIMKU** (jedan materijal može biti na više različitih primki, dok je jedna primka vezana za jedan materijal) **jedan na više**
* **PROJEKT** ima **PRIMKU** (jedan projekt može imati više različitih primki, dok je jedna primka vezana za jedan projekt) **jedan na više**
* **PROJEKT** se dijeli na **SEGMENTE** (jedan projekt može imati više segmenata, dok je jedan segment specifično vezan za jedan projekt) **jedan na više**
* **SEGMENTU** se prati **PROIZVODNJA** (jedan segment može proći kroz više različitih dijelova proizvodnje, dok se u toj jednoj proizvodnji proizvodi jedan segment) **jedan na više**
* **PROIZVODNJA** prolazi kroz **FAZU IZRADE** (jedna specifična faza izrade može biti kod više različitih proizvodnji, dok u jednoj proizvodnji kada se odradi određena faza, ona ne može biti ponovljena, dakle jedinstvena je) **jedan na više**
* **ZAPOSLENIK** zadužuje **ALAT** (jedan zaposlenik može zadužiti više različitih alata, dok jedan alat može biti zadužen kod više različitih zaposlenika) **više na više**
* Stvara se dodatni entitet **ZADUZENJE\_ALATA** gdje se referenciraju primarni ključevi od **ZAPOSLENIK** i **ALAT** kao strani ključ u ZADUZENJE\_ALATA. Prati se još datum zapisa te određena napomena zaposlenika.
* **ZAPOSLENIK** odrađuje **PROIZVODNJU** (jedan zaposlenik može odraditi više proizvodnji, dok se u jednoj proizvodnji može koristiti više zaposlenika) **više na više**
* Stvara se dodatni entitet **UTROSAK\_SATI** koji prati utrošak sati zaposlenika pri proizvodnji, te referencira primarne ključeve od **ZAPOSLENIK** i **PROIZVODNJA** kao strani ključ.
* **SEGMENT** se nalazi na **ISPORUCI** (jedan segment može biti na više različitih isporuka, dok u jednoj isporuci se nalazi samo jedan segment) **jedan na više**
* **ZAPOSLENIK** šalje **ISPORUKU** (jedan zaposlenik može slati više isporuka, dok je jedna isporuka poslana od strane samo jednog zaposlenika) **jedan na više**

# RELACIJSKI MODEL (SHEME)

**klijent** (id, naziv\_klijenta, odgovorna\_osoba, kontakt\_broj, mail\_adresa)

**zaposlenik** (id, ime, prezime, kontakt\_broj, cijena\_po\_satu)

**projekt** (id, id\_klijent, radni\_nalog, naziv\_projekta, nacrti, rok\_izrade)

**segment** (id, id\_projekt, naziv\_segmenta, kolicina)

**faze\_izrade** (id, naziv\_faze, opis, cijena\_stroja\_po\_satu)

**tip\_materijala** (id, naziv\_tipa\_materijala)

**materijal** (id, id\_tip\_materijala, naziv\_materijala, opis, napomena, cijena\_po\_komadu)

**proizvodnja** (id, id\_segment, id\_faza, status\_proizvodnje, datum)

**utrosak\_sati** (id, id\_proizvodnja, id\_zaposlenik, utrosak\_sati)

**utrosak\_materijala** (id, id\_projekt, id\_materijal, kolicina)

**primka\_materijala** (id, id\_projekt, id\_materijal, kolicina ,status\_narudzbe ,datum\_upisa)

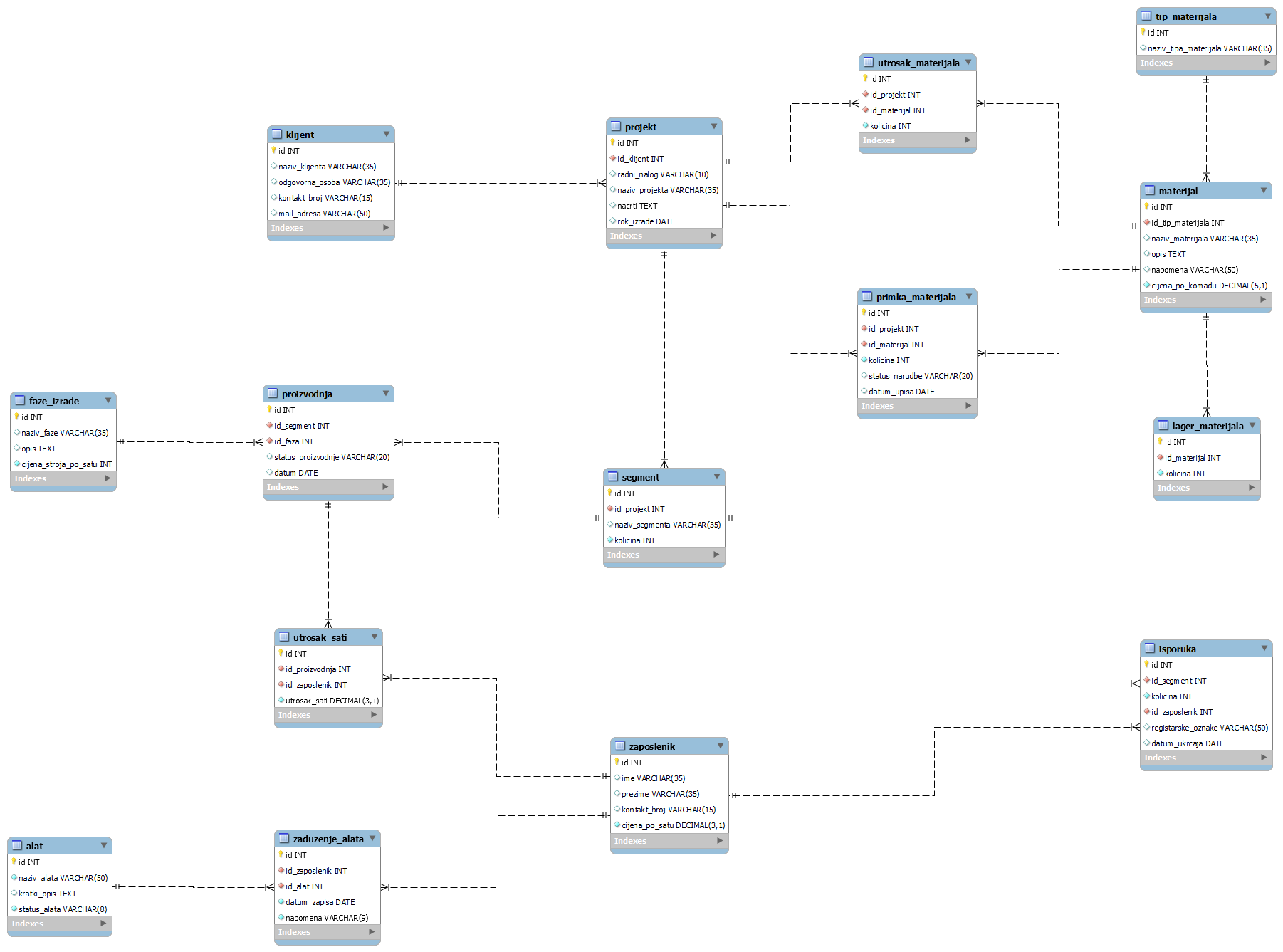
**alat** (id, naziv\_alata, kratki\_opis, status\_alata)

**lager\_materijala** (id, id\_materijal, kolicina)

**isporuka** (id, id\_segment, kolicina, id\_zaposlenik, registarske\_oznake, datum\_ukrcaja)

**zaduzenje\_alata** (id, id\_zaposlenik, id\_alat, datum\_zapisa, napomena)

# EER DIJAGRAM (MYSQL)



*Slika 2/EER dijagram, napravljen u MySQL Workbenchu*

# TABLICE

## TABLICA klijent

Pomoću tablice **klijent** stvaramo bazu podataka o klijentima te informacijama vezanih uz njih**: id, naziv\_klijenta, odgovorna\_osoba,kontakt\_broj, mail\_adresa**. Atribut **id** nam predstavlja interni PRIMARY KEY i njega smo stavili tipa INTEGER. **Naziv\_klijenta** odnosno tvrtke koja naručuje izradu određenog projekta zapisat ćemo pomoću VARCHAR(35). U našu bazu podataka ulazi i **odgovorna\_osoba** koju ćemo upisivati pomoću VARCHAR(35) ,a u slučaju nedoumica ili promjene u projektu dužni smo informirati odgovornu osobu pa ćemo **kontakt\_broj** te osobe također upisati u tablicu i bit će nam tipa VARCHAR(15). Broj telefona nismo stavili tipa integer zbog stranih klijenata čije brojeve moramo upisivati pomoću pozivnog broja zemlje iz koje dolazi a za to ponekad koristimo `+` . Radi lakše komunikacije i evidentiranih dogovora ili eventualnih izmjena, komunicirat ćemo sa odgovornom osobom i putem maila pa nam je potrebna njegova **mail\_adresa** koju ćemo upisivati pomoću VARCHAR(50).

CREATE TABLE klijent (

id INTEGER PRIMARY KEY,

naziv\_klijenta VARCHAR(35),

odgovorna\_osoba VARCHAR(35),

kontakt\_broj VARCHAR(15),

mail\_adresa VARCHAR(50)

);

## TABLICA zaposlenik

Tablica **zaposlenik** nam sadrži atribute: **id, ime, prezime, kontakt\_broj, cijena\_po\_satu**. Pomoću **id**-a koji je PRIMAY KEY tipa INTEGER, povezivat ćemo ovu relaciju sa drugim relacijama s kojima je u međuovisnosti. **Ime** svakog zaposlenika upisivat ćemo pomoću VARCHAR tipa koji smo ograničili na 35 znakova, za **prezime** ćemo također koristiti tip VARCHAR(35) sa istim ograničenjem.Za **kontakt\_broj** koristimo VARCHAR(15) također da bismo omogućili korištenje znaka `+` kao dio pozivnog broja u slučaju da neki naš zaposlenik dolazi iz strane zemlje i ne koristi hrvatski broj telefona. Ovisno o stažu, znanju, iskustvu i zalaganju, svakom zaposleniku je uprava dodijelila i posebnu satnicu pomoću koje će kasnije biti kreirana završna cijena gotovog proizvoda pa nam je za ovu relaciju bitan i atribut **cijena\_po\_satu** koju ćemo unijeti tipa NUMERIC(3,1) NOT NULL zbog mogućeg decimalnog broja.

CREATE TABLE **zaposlenik** (

**id** INTEGER PRIMARY KEY,

**ime** VARCHAR(35),

**prezime** VARCHAR(35),

**kontakt\_broj** VARCHAR(15),

**cijena\_po\_satu** NUMERIC(3,1) NOT NULL,

);

## TABLICA projekt

Tablica **projekt** nam služi kako bismo lakše pratili cijeli proces proizvodnje od klijentovog zahtjeva, podjelu na segmente, utrošak materijala do gotovog krajnjeg proizvoda. Sastavljena je od atributa: **id** koji nam je i ovog puta PRIMARY KEY odnosno jedinstevi broj zapisan u INTEGER tip podatka. Svaki projekt povezujemo sa FOREIGN KEY relacije klijent odnosno, **id\_klijent** tipa INTEGER i stavili smo ograničenje NOT NULL kako bismo spriječili eventualnu pogrešku prilikom unosa podataka. Nadalje, atribut **radni\_nalog** tipa VARCHAR(10) smo ograničili na 10 simbola radi lakše interne kontrole. Svakom projektu smo dodijelili i ime atributom **naziv\_projekta** koji je tipa VARCHAR(35). Atribut **nacrti** je tipa TEXT jer će tamo biti pohranjena adresa nacrta. Za svaki projekt imamo dogovoreno i vrijeme isporuke pa taj podatak pratimo atributom **rok\_izrade** koji ćemo zapisivati DATE tipom.

CREATE TABLE projekt (

id INTEGER PRIMARY KEY,

id\_klijent INTEGER NOT NULL,

FOREIGN KEY (id\_klijent) REFERENCES klijent (id),

radni\_nalog VARCHAR(10),

naziv\_projekta VARCHAR(35),

nacrti TEXT,

rok\_izrade DATE

);

## TABLICA segment

U ovoj tablici vodimo evidenciju i stanje izrade pojedinog **segment**a. Atributi potrebni za ažurno praćenje ovih podataka su: **id, id\_projekt, naziv\_segmenta, kolicina**. **Id** nam je PRIMARY KEY tipa INTEGER, **id\_projekt** je FOREIGN KEY u ovoj relaciji (odnosno, PRIMARY KEY relacije projekt) tipa INTEGER NOT NULL . Ovaj podatak nam ne smije izostati, pa smo stavili ograničenje NOT NULL iz tog razloga. Svaki segment imenujemo atributom **naziv\_segmenta** tipom VARCHAR(35) i unosimo **kolicina** INTEGER također NOT NULL.

CREATE TABLE **segment (**

**id** INTEGER PRIMARY KEY,

**id\_projekt** INTEGER NOT NULL,

FOREIGN KEY (id\_projekt) REFERENCES projekt (id),

**naziv\_segmenta** VARCHAR(35),

**kolicina** INTEGER NOT NULL

);

## TABLICA faze\_izrade

Ova tablica služi za evidentiranje različitih **faza izrade** koje se koriste u proizvodnji metalnih konstrukcija. Svaka faza ima svoj **naziv**, **opis** i **cijenu stroja po satu,** što je ključno za planiranje i praćenje troškova proizvodnje. Tablica sadrži atribute : „**id**“ – INTEGER PRIMARY KEY – što znači da je atribut primarni ključ te da je podatak broj. On nam služi kao jedinstveni idetifikator za svaku fazu izrade; „**naziv\_faze**“ - VARCHAR(35) – fazu izrade imenujemo znakovnim nizom ne dužim od 35 znakova; „**opis**“ – TEXT - detaljni opis faze izrade tipa podatka tekst; „**cijena\_stroja\_po\_satu**“ – INTEGER NOT NULL – cijena stroja po satu tijekom faze izrade koja ne smijem biti 0. Tablica omogućava praćenje troškova proizvodnje po fazama, identifikaciju različitih faza izrade i njihovu uporabu u projektima.

CREATE TABLE **faze\_izrade** (

**id** INTEGER PRIMARY KEY,

**naziv\_faze** VARCHAR(35),

**opis** TEXT,

**cijena\_stroja\_po\_satu** INTEGER NOT NULL

);

## TABLICA tip\_materijala

Ova tablica služi za kategorizaciju materijala koji se koriste u proizvodnji. Omogućuje identifikaciju tipova materijala, što pomaže u organizaciji i upravljanju skladištem. Njezini atributi su: „**id**“ koji nam služi kao primarni ključ za svaki tip materijala i tip podatka mu je INTEGER te „**naziv\_tipa\_materijala**“ koji sadrži naziv tipa materijala i tip podatka mu je VARCHAR(35). Omogućava praćenje i upravljanje različitim tipovima materijala te njihovo povezivanje s konkretnim materijalima u skladištu.

CREATE TABLE **tip\_materijala** (

**id** INTEGER PRIMARY KEY,

**naziv\_tipa\_materijala** VARCHAR(35)

);

## TABLICA materijal

Ova tablica evidentira pojedinačne materijale koji se koriste u proizvodnji. Sadrži informacije o vrsti materijala, količini, opisu i cijeni po komadu, što je ključno za upravljanje troškovima i zalihe materijala. Tablica ima atribute: „**id**“ koji je njezin primarni ključ tipa INTEGER; „**id\_tip\_materijala**“ koji je strani ključ koji povezuje materijal s njegovim tipom te je on INTEGER NOT NULL; „**naziv\_materijala**“ sadrži naziv materijala i tipa je VARCHAR(50); „**opis**“ detaljan opis materijala tip TEXT; „**napomena**“ dodatne napomene o materijalu; „**cijena\_po\_komadu**“ sadrži cijenu materijala po komadu i tipa je NUMERIC(5,1) NOT NULL što znači da ne može pohraniti vrijednosti koje imaju više od ukupno 5 znamenki (npr. 123456.0) ili vipe od 1 znamenke iza decimalne točke (npr. 123.45). Omogućava praćenje zaliha, troškova materijala i specifikacija materijala koji se koriste u projektima.

CREATE TABLE **materijal** (

**id** INTEGER PRIMARY KEY,

**id\_tip\_materijala** INTEGER NOT NULL,

FOREIGN KEY (**id\_tip\_materijala**) REFERENCES **tip\_materijala** (id),

**naziv\_materijala** VARCHAR(35),

**opis** TEXT,

**napomena** VARCHAR(50),

**cijena\_po\_komadu** NUMERIC(5,1) NOT NULL

);

## TABLICA proizvodnja

Ova tablica prati status proizvodnje svakog segmenta projekta kroz različite faze izrade. Evidentira trenutni status i datum svake proizvodne aktivnosti. Tablica ima atribute: „**id“** koji nam je primarni ključ te je tipa INTEGER; „**id\_segment“** strani ključ koji povezuje proizodnju s odgovarajućim segmentom projekta i tip je INTEGER NOT NULL; „**id\_faza**“ strani ključ koji povezuje proizvodnju s fazom izrade i tipa je INTEGER NOT NULL; „**status\_proizvodnje**“ status trenutne proizvodne aktivnosti i tipa je VARCHAR(20); „**datum**“ datum aktivnosti u proizvodnju. Omogućava praćenje statusa proizvodnje, identifikaciju zastoja u proizvodnji i planiranje daljnjih aktivnosti na temelju trenutnog statusa i datuma.

CREATE TABLE **proizvodnja** (

**id** INTEGER PRIMARY KEY,

**id\_segment** INTEGER NOT NULL,

FOREIGN KEY (**id\_segment**) REFERENCES **segment** (**id**),

**id\_faza** INTEGER NOT NULL,

FOREIGN KEY (**id\_faza**) REFERENCES **faze\_izrade** (**id**),

**status\_proizvodnje** VARCHAR(20),

**datum** DATE

);

## TABLICA utrosak\_sati

Tablica **utrosak\_sati** pomaže u evidenciji potrošenih sati na izradu pojedinog segmenta projekta. Ta informacija će nam biti potrebna u kreiranju konačne cijene proizvoda krajnjem kupcu. Tablica sadrži atribute: **id** i **utrosak\_sati** te primarne ključeve tablica o kojima ovisi: **id\_proizvodnje** i **id\_zaposlenik.** Atribut **id** je PRIMARY KEY tablice tipa INTEGER jer se u njega unosi jedinstvena brojčana vrijednost za identifikaciju utrošenih sati za pojedini segment. Nisu nam potrebna dodatna ograničenja za jedinstvenost jer upravo to i jest primarni ključ po svojoj definiciji. Razlog stvaranje ove relacije jest upravo **utrosak\_sati** tipa INTEGER kako bismo pratili zauzetost naših radnika i njihovu angažiranost za pojedini segment. FOREIGN KEY **id\_proizvodnja** spaja relacije **segment**, **utosak\_sati** i **faza\_izrade,** dok **id\_zaposlenik** kao strani ključ spaja broj utrošenih sati sa zaposlenikom koji radi na određenom segmentu projekta.

CREATE TABLE utrosak\_sati (

id INTEGER PRIMARY KEY,

id\_proizvodnja INTEGER NOT NULL,

FOREIGN KEY (id\_proizvodnja) REFERENCES proizvodnja (id),

id\_zaposlenik INTEGER NOT NULL,

FOREIGN KEY (id\_zaposlenik) REFERENCES zaposlenik (id),

Utrosak\_sati NUMERIC(3,1) NOT NULL

);

## TABLICA utrosak\_materijala

Tablica utrosak\_materijala služi evidenciji materijala koji je potrošen na određenom projektu kako bismo kasnije mogli kreirati cijenu. Ova relacija sadrži atribute: **id**, **id\_projekt**, **id\_materijal**, **kolicina.** Atribut **id** je tipaINTEGER jer je jedinstveni broj za identifikaciju količine materijala potrošene na određeni projekt. Zbog međuovisnosti projekta i materijala, u ovoj relaciji navodimo i njihove PRIMARY KEY koji nam ovdje postaju FOREIGN KEY. Stavili smo ograničenje NOT NULL jer nam materijal mora biti povezan nekim projektom i materijalom čije stanje pratimo. Atribut **kolicina** smo također ograničili na neki pozitivan broj jer u suprotnom ova relacija nema smisla.

CREATE TABLE utrosak\_materijala (

id INTEGER PRIMARY KEY,

id\_projekt INTEGER NOT NULL,

FOREIGN KEY (id\_projekt) REFERENCES projekt (id),

id\_materijal INTEGER NOT NULL,

FOREIGN KEY (id\_materijal) REFERENCES materijal (id),

kolicina INTEGER NOT NULL CHECK (kolicina > 0)

);

## TABLICA primka\_materijala

Tablica primka\_materijala sadrži atribute: **id**, **id\_projekt**, **id\_materijal**, **kolicina, status\_narudzbe**, **datum\_upisa**. I ovdje nam je **id** kao i kod ostalih relacija tipa INTEGER, PRIMARY KEY po svojoj definiciji, jedinstveni broj za identifikaciju primke. FOREIGN KEY koji povezuje ovu relaciju s relacijama koje vode evidenciju o projektu i materijalima nam je također tipa INTEGER. Atribut **kolicina** označuje količinu materijala kojeg smo naručili, odnosno ima li potrebe naručivati još materijala ili je onaj na lageru dovoljan te ćemo za njega koristiti INTEGER NOT NULL. Stavljamo ograničenje kako bismo spriječili moguće pogreške prilikom unosa podataka. **Status\_narudzbe** nam označava ako je meterijal dostavljen ili tek naručen. **datum\_upisa** koristimo kako bismo lakše predvidjeli sam datum isporuke obzirom na potrebne sate za izradu ovog projekta i projekata koji su u procesu izrade, odnosno na pravilan poredak na „listi čekanja“. Za ovaj atribut koristit ćemo DATETIME NOT NULL.

CREATE TABLE primka\_materijala(

id INTEGER PRIMARY KEY,

id\_projekt INTEGER NOT NULL,

FOREIGN KEY (id\_projekt) REFERENCES projekt (id),

id\_materijal INTEGER NOT NULL,

FOREIGN KEY (id\_materijal) REFERENCES materijal (id),

kolicina INTEGER NOT NULL,

status\_narudzbe VARCHAR(20),

datum\_upisa DATE

);

## TABLICA alat

Relacija alat sadrži atribute: id, naziv\_alata, kratki\_opis, status. Atribut id nam je INTEGER, PRIMARY KEY. Naziv\_alata smo stavili VARCHAR(50), tj. znakovni niz koji smo ograničili na 50 znakova. Atribut kratki\_opis smo stavili tipa TEXT. Ovaj stupac nam služi kako bismo korisnicima naše baze podataka dali kratki opis alata i oznaku kako bi se laše prepoznalo ako se radi baš o tom alatu. Status će nam davati informaciju o eventualnoj nedostupnosti određenog alata. Bitno je znati koji su alati dostupni u kojem trenutku kako bi voditelj što kvalitetnije iskoristio i koordinirao radnike obzirom na dostupnost alata. Atribut status je tipa VARCHAR (8) te se nude samo dvije mogućnosti upisa: „ispravno“ ili „servis“ i nije potrebno zauzimati više memorije.

CREATE TABLE alat(

id INTEGER PRIMARY KEY,

naziv\_alata VARCHAR (50) NOT NULL,

kratki\_opis TEXT,

status\_alata VARCHAR (8) NOT NULL

);

## TABLICA lager\_materijala

Tablica **lager materijala** je dizajnirana za praćenje lagera materijala u skladištu. Struktura tablice obuhvaća sljedeća polja:

**id**: Ovo je jedinstveni identifikator za svaki zapis u tablici. Polje je tipa INTEGER i postavljeno je kao PRIMARY KEY, što znači da svaka vrijednost mora biti jedinstvena i ne može biti NULL.

**id\_materijala**: Ovo polje služi za identifikaciju materijala. Također je tipa INTEGER i ne može biti NULL. Očekuje se da će ovo polje vjerojatno biti povezano s drugim tablicama koje detaljnije opisuju materijale, iako u ovom primjeru ta relacija nije eksplicitno navedena.

**količina**: Ovo polje predstavlja količinu materijala koja je trenutno na lageru. Polje je tipa INTEGER i ne može biti NULL, što znači da svaki zapis mora imati definiranu količinu materijala.

CREATE TABLE lager\_materijala (

id INTEGER PRIMARY KEY,

id\_materijal INTEGER NOT NULL,

FOREIGN KEY (id\_materijal) REFERENCES materijal (id),

kolicina INTEGER NOT NULL

);

## TABLICA isporuka

Tablica isporuka je dizajnirana za praćenje isporuka segmenta proizvoda. Struktura tablice obuhvaća sljedeća polja:

**id**: Ovo je jedinstveni identifikator za svaki zapis u tablici. Polje je tipa INTEGER i postavljeno je kao PRIMARY KEY, što znači da svaka vrijednost mora biti jedinstvena i ne može biti NULL.

**id\_segment**: Ovo polje služi za identifikaciju segmenta proizvoda koji je isporučen. Također je tipa INTEGER i ne može biti NULL.

**kolicina**: Ovo polje predstavlja količinu segmenta proizvoda koja je isporučena. Polje je tipa INTEGER i ne može biti NULL.

**id\_zaposlenik**: Ovo polje služi za identifikaciju zaposlenika koji je odgovoran za isporuku. Također je tipa INTEGER i ne može biti NULL.

**registarske\_oznake**: Ovo polje služi za unos registarskih oznaka vozila koje je korišteno za isporuku. Polje je tipa VARCHAR s maksimalnom duljinom od 50 znakova.

**datum\_ukrcaja**: Ovo polje predstavlja datum kada je segment proizvoda ukrcan za isporuku. Polje je tipa DATE

CREATE TABLE isporuka (

id INTEGER PRIMARY KEY,

id\_segment INTEGER NOT NULL,

FOREIGN KEY (id\_segment) REFERENCES segment (id),

kolicina INTEGER NOT NULL,

id\_zaposlenik INTEGER NOT NULL,

FOREIGN KEY (id\_zaposlenik) REFERENCES zaposlenik (id),

registarske\_oznake VARCHAR(10),

datum\_ukrcaja DATE

);

## TABLICA zaduženje alata

Tablica **zaduženje alata** je dizajnirana za praćenje zaduživanja alata od strane zaposlenika. Struktura tablice obuhvaća sljedeća polja:

**id**: Ovo je jedinstveni identifikator za svaki zapis u tablici. Polje je tipa INTEGER i postavljeno je kao PRIMARY KEY, što znači da svaka vrijednost mora biti jedinstvena i ne može biti NULL.

**id\_zaposlenik**: Ovo polje služi za identifikaciju zaposlenika koji je zadužio alat. Također je tipa INTEGER i ne može biti NULL.

**id\_alat**: Ovo polje služi za identifikaciju alata koji je zadužen. Također je tipa INTEGER i ne može biti NULL.

**datum\_zapisa**: Ovo polje predstavlja datum kada je alat zadužen. Polje je tipa DATE I ne može biti NULL.

**napomena**: Ovo polje služi za dodatne napomene vezane za zaduženje alata. Polje je tipa VARCHAR s maksimalnom duljinom od 9 znakova, odnosno ‘zaduzeno’ ili ‘razduzeno’.

Ova tablica je ključna za upravljanje zaduživanjem alata, omogućujući praćenje koji je alat zadužen, od koga i kada, kao i eventualne dodatne napomene vezane za zaduženje.

CREATE TABLE zaduzenje\_alata (

id INTEGER PRIMARY KEY,

id\_zaposlenik INTEGER NOT NULL,

FOREIGN KEY (id\_zaposlenik) REFERENCES zaposlenik (id),

id\_alat INTEGER NOT NULL,

FOREIGN KEY (id\_alat) REFERENCES alat (id),

datum\_zapisa DATE NOT NULL,

napomena VARCHAR(9) NOT NULL

­­

# UPITI

* Morena Martan (7.1 - 7.5)
* Loren Bažon (7.6 – 7.10)
* Josip Milković (7.11 – 7.13)
* Fran Barba (7.14 – 7.18)

## UPIT 1

Količina velikih kutnih brusilica i količina malih kutnih brusilica.

S obzirom na veliku količinu projekata ove godine i uzimajući u obzir da su nam velika i mala kutna brusilica najčešći alat, voditelja proizvodnje zanima količina ispravnih brusilica. Kako bi se moglo nabaviti nove velike i male kutne brusilice ukoliko je potrebno.

TRAŽENO RJEŠENJE:

**tip\_brusilice, broj\_komada**

KOD ZA UPIT:

SELECT

CASE

WHEN kratki\_opis LIKE '%Mala kutna brusilica%' THEN 'Mala kutna brusilica'

WHEN kratki\_opis LIKE '%Velika kutna brusilica%' THEN 'Velika kutna brusilica'

END AS tip\_brusilice,

COUNT(\*) AS broj\_komada

FROM

alat

WHERE

kratki\_opis LIKE '%Mala kutna brusilica%' AND status\_alata = 'ispravno'

OR kratki\_opis LIKE '%Velika kutna brusilica%' AND status\_alata = 'ispravno'

GROUP BY

tip\_brusilice;

OPIS UPITA:

Ovaj upit prebrojava koliko ima 'Malih kutnih brusilica' i 'Velikih kutnih brusilica' u relaciji **alat** na temelju sadržaja atributa **kratki\_opis**. Prvo koristi CASE izraz za klasifikaciju brusilica kao **'Mala kutna brusilica**' ili **'Velika kutna brusilica'**, a zatim koristi COUNT(\*) za brojanje N-torki u svakoj grupi. Konačni rezultat prikazuje broj komada za svaki tip brusilice.

* SELECT naredba odabire atribute koji će biti uključeni u rezultat upita.
* CASE je uvjetna logika koja se koristi za određivanja tipa busilice, u nastavku koda je WHEN **kratki\_opis** LIKE **'%Mala kutna brusilica%'** THEN **'Mala kutna brusilica'** koji nam govori ako neka N-torka sadrži tekst 'Mala kutna brusilica' , tada će vrijednost atributa **tip\_brusilice** biti **'Mala kutna brusilica'**.Nakon toga ista stvar za **'Velika kutna brusilica'** te na kraju END AS **tip\_brusilice** koji nam rezultat CASE izraza daje aliasu **tip\_brusilice**.
* COUNT(\*) AS **broj\_komada** nam broji svaku n-torku koja odgovara jednom od 2 izraza prije i te brojeve prikazuje u atributu **broj\_komada**.
* FROM alat specificira iz koje se relacije preuzimaju podaci. WHERE filtrira n-torke koje će se uključiti u grupiranje, zatim **kratki\_opis** LIKE **'%Mala kutna brusilica%'** AND **status\_alata = 'ispravno'** OR **kratki\_opis** LIKE **'%Velika kutna brusilica%'** AND **status\_alata = 'ispravno'** uključuje samo one n-torke gdje gdje **kratki\_opis** sadrži tekst **'Mala kutna brusilica'** ili '**Velika kutna brusilica'** i **status\_alata 'ispravno'** kako bi filtrirali ispravne od onih koje su na servisu.
* GROUP BY **tip\_brusilice** nam grupira rezultate prema vrijednosti koja je određena u CASE izrazu. TJ. Sve n-torke koje imaju isti tip brusilice bit će grupirane zajedno.

REZULTAT:



## UPIT 2

Popis svih zaposlenika s trenutnom i povećanom satnicom za 0,6 €

Obzirom na porast BDP-a, rastu i plaće u državi. Računovodstvo nas je tražilo popis svih zaposlenika te ispis trenutne satnice za svakog pojedinog radnika. Obzirom na veliku dobit, planiraju povećanje satnice svakom radniku za 0.60EUR pa su tražili da dodamo i taj podatak.

TRAŽENO RJEŠENJE:

**id, ime, prezime, cijena\_po\_satu, nova\_cijena\_po\_satu**

KOD ZA UPIT:

SELECT id, ime, prezime, cijena\_po\_satu, cijena\_po\_satu + 0.6 AS nova\_cijena\_po\_satu

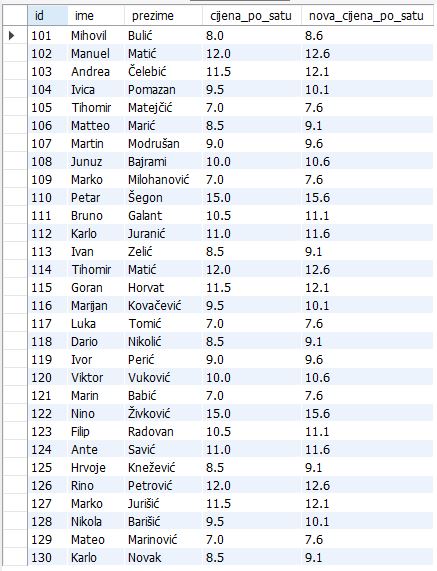
FROM zaposlenik;

OPIS UPITA:

Ovaj upit pretražuje tablicu **zaposlenik** i odabire atribute **id, ime, prezime, cijena\_po\_satu**. Prikazuje trenutnu satnicu za svakog zaposlenika i izračunava novu satnicu dodavanjem 0,6 EUR na trenutnu satnicu. Konačni rezultat je popis zaposlenika s trenutnom i novom satnicom.

* SELECT **id** odabire jedinstveni identifikator svakog zaposlenika. **ime** odabire ime zaposlenika. **prezime** odabire prezime zaposlenika. **cijena\_po\_satu** odabire trenutnu satnicu zaposlenika. **cijena\_po\_satu + 0.6** AS **nova\_cijena\_po\_satu** izračunava novu satnicu povećanjem trenutne satnice za 0,6 EUR i daje ovom izračunu alias **nova\_cijena\_po\_satu.**
* FROM **zaposlenik** specificira tablicu **zaposlenik** kao izvor podataka za upit.

REZULTAT:



## UPIT 3

Popis projekata s rokom do kraja 2024. godine i potrebnim materijalom

Projektni menadžer ide na godišnji odmor. Zatražio je da mu dostavimo popis svih projekata s rokom do kraja 2024 i potrebnim materijalom kako bi ga naručio prije nego ode, da bi se mogao neometano odmoriti.

TRAŽENO RJEŠENJE:

**naziv\_projekta, rok\_izrade, kolicina, naziv\_materijala, opis, napomena**

KOD ZA UPIT:

SELECT p.naziv\_projekta, p.rok\_izrade, u.kolicina, m.naziv\_materijala, m.opis, m.napomena

FROM utrosak\_materijala AS u

JOIN projekt AS p ON p.id = u.id\_projekt

JOIN materijal AS m ON u.id\_materijal = m.id

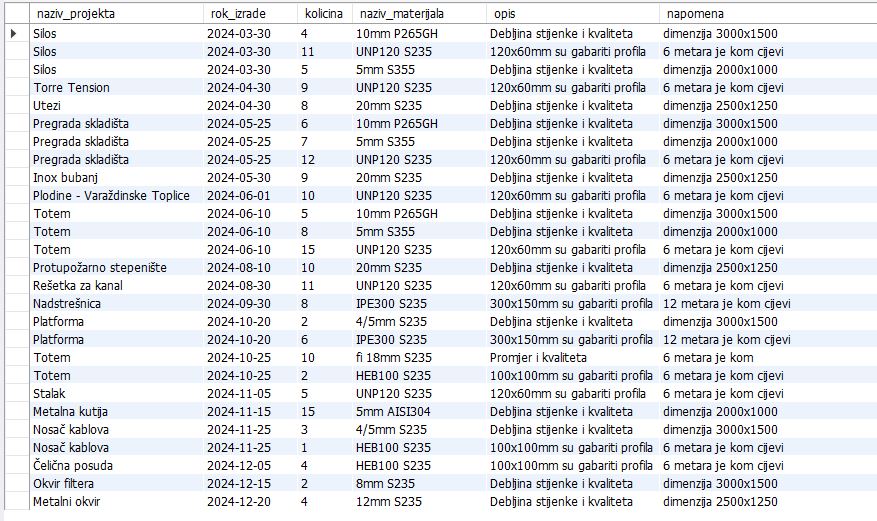
WHERE p.rok\_izrade <= '2024-12-31';

OPIS UPITA:

Ovaj upit pretražuje bazu podataka kako bi pronašao sve projekte s rokom izrade do kraja 2024. godine i potrebnim materijalom za te projekte. Upit odabire naziv projekta, rok izrade, količinu potrebnog materijala, naziv materijala, opis materijala i napomenu vezanu za materijal. Konačni rezultat je popis projekata s potrebnim materijalom, što omogućava projektnom menadžeru da naruči sav potreban materijal prije nego ode na godišnji odmor.

* SELECT **p.naziv\_projekta** odabire naziv projekta iz tablice **projekt**. **p.rok\_izrade** odabire rok izrade projekta iz tablice **projekt**. **u.kolicina** odabire količinu materijala potrebnu za projekt iz tablice **utrosak\_materijala**. **m.naziv\_materijala** odabire naziv materijala iz tablice **materijal**. **m.opis** odabire opis materijala iz tablice **materijal**. **m.napomena** odabire napomenu vezanu za materijal iz tablice **materijal**.
* FROM **utrosak\_materijala** AS **u** specificira tablicu **utrosak\_materijala** kao izvor podataka za upit i daje joj alias **u**.
* JOIN **projekt** AS **p** ON **p.id = u.id\_projekt** pridružuje tablicu **projekt** tablici **utrosak\_materijala** na temelju zajedničkog atributa id projekta.
* JOIN **materijal** AS **m** ON **u.id\_materijal = m.id** pridružuje tablicu **materijal** tablici **utrosak\_materijala** na temelju zajedničkog atributa id materijala.
* WHERE **p.rok\_izrade <= '2024-12-31'** filtrira rezultate tako da se prikažu samo projekti s rokom izrade do kraja 2024. godine.

REZULTAT:



## UPIT 4

Prikaz popisa svih alata koji su dostavljeni na servis.

Neki alati nam se nisu vratili sa servisa, pa je serviser Ivan tražio popis svih naših alata koji su mu dostavljeni kako bi ih lakše našao u svojoj radioni

TRAŽENO RJEŠENJE:

**naziv\_alata, kratki\_opis**

KOD ZA UPIT:

SELECT naziv\_alata, kratki\_opis

FROM alat

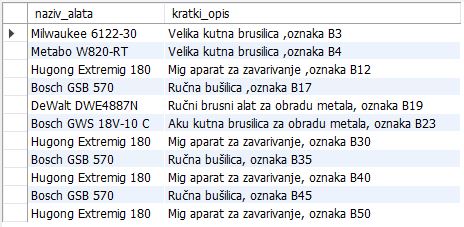
WHERE status\_alata = 'servis';

OPIS UPITA:

Ovaj upit pretražuje bazu podataka kako bi pronašao sve alate koji su trenutno na servisu. Upit odabire naziv i kratki opis alata iz tablice **alat** i filtrira rezultate tako da se prikažu samo oni alati koji imaju status 'servis'. Konačni rezultat je popis alata koji su trenutno na servisu, što omogućava serviseru Ivanu da ih lakše pronađe u svojoj radioni.

* SELECT **naziv\_alata** odabire naziv alata iz tablice **alat**. **kratki\_opis** odabire kratki opis alata iz tablice **alat**.
* FROM **alat** specificira tablicu **alat** kao izvor podataka za upit.
* WHERE **status\_alata = 'servis'** filtrira rezultate tako da se prikažu samo one n-torke gdje je status alata 'servis'.

REZULTAT:



## UPIT 5

Prikaz svih ukrcaja kamiona registarskih tablica PU-5566-C sa popisom segmenata, količinom i datumom ukrcaja.

Šofer Luka je napustio skladište i zaboravio ponijeti papir s popisom segmenata, količinom i datumom. Trenutno je na granici sa Slovenijom i granična policija ga traži dokument tereta i sve predhodne ukrcaje koji su ukrcani iz proizvodnje . Moramo mu hitno dostaviti tražene podatke.

TRAŽENO RJEŠENJE:

**kolicina, datum\_ukrcaja, registarske\_oznake, naziv\_segmenta**

KOD ZA UPIT:

SELECT u.kolicina, u.datum\_ukrcaja, u.registarske\_oznake, s.naziv\_segmenta

FROM isporuka AS u

JOIN segment AS s ON u.id\_segment = s.id

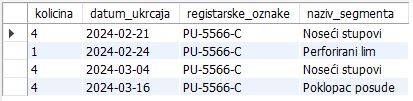
WHERE u.registarske\_oznake = 'PU-5566-C';

OPIS UPITA:

Ovaj upit pretražuje bazu podataka kako bi pronašao sve relevantne informacije o ukrcaju kamiona s registarskim oznakama "PU-5566-C". Upit spaja (JOIN) tablice **isporuka** i **segment** kako bi dobio nazive segmenata, a zatim filtrira rezultate prema registarskim oznakama. Konačni rezultat je popis segmenata, količina, datuma ukrcaja i registarskih oznaka, što omogućava vozaču da graničnoj policiji pruži potrebne informacije o teretu.

* SELECT **u.kolicina** odabire količinu iz tablice **isporuka**. **u.datum\_ukrcaja** odabire datum ukrcaja iz tablice **isporuka**. **u.registarske\_oznake** odabire registarske oznake iz tablice **isporuka**. **s.naziv\_segmenta** odabire naziv segmenta iz tablice **segment**.
* FROM **isporuka** AS **u** specificira tablicu **isporuka** kao početnu točku za upit i daje joj alias **u**.
* JOIN **segment** AS **s** ON **u.id\_segment = s.id** povezuje tablicu **segment** s tablicom **isporuka** na temelju **id\_segment** iz tablice **isporuka** i **id** iz tablice **segment**, te daje tablici **segment** alias **s**.
* WHERE **u.registarske\_oznake = 'PU-5566-C'** filtrira rezultate tako da se prikažu samo one n-torke gdje su registarske oznake kamiona "PU-5566-C".

REZULTAT:



## UPIT 6

Prikaz projekata po klijentima, uključujući broj segmenata za svaki projekt

Naša firma ima dvije podružnice proizvodnje sa istim brojem radnika. Odlučili su nagraditi sve radnike proizvodnje koja je napravila više projekata pa su nas tražili popis projekata po klijentima, uključujući broj segmenata za svaki projekt.

TRAŽENO RJEŠENJE:

**naziv\_klijenta, naziv\_projekta, broj\_segmenata**

KOD ZA UPIT:

SELECT k.naziv\_klijenta, p.naziv\_projekta, COUNT(s.id) AS broj\_segmenata

FROM klijent k

JOIN projekt AS p ON k.id = p.id\_klijent

JOIN segment AS s ON p.id = s.id\_projekt

GROUP BY k.naziv\_klijenta, p.naziv\_projekta;

OPIS UPITA:

Ovaj upit pretražuje bazu podataka kako bi prikazao sve projekte grupirane po klijentima, uključujući broj segmenata za svaki projekt. Upit koristi spajanje (JOIN) tablica **klijent**, **projekt**, i **segment** kako bi dobio sve potrebne informacije. Konačni rezultat je popis klijenata s njihovim projektima i brojem segmenata za svaki projekt, što pomaže u ocjeni proizvodnih pogona na temelju broja dovršenih projekata.

* SELECT **k.naziv\_klijenta** odabire naziv klijenta iz tablice **klijent**. **p.naziv\_projekta** odabire naziv projekta iz tablice **projekt**. COUNT(**s.id**) AS **broj\_segmenata** broji broj segmenata (komponenti) u svakom projektu i daje tom broju alias **broj\_segmenata**.
* FROM **klijent k** specificira tablicu klijent kao početnu točku za upit i daje joj alias k.
* JOIN **projekt** AS **p** ON **k.id = p.id\_klijent** povezuje tablicu **projekt** s tablicom **klijent** na temelju **id\_klijent** iz tablice **projekt** i **id** iz tablice **klijent**, te daje tablici **projekt** alias **p**.
* JOIN **segment** AS **s** ON **p.id = s.id\_projekt** povezuje tablicu **segment** s tablicom **projekt** na temelju **id\_projekt** iz tablice **segment** i **id** iz tablice **projekt**, te daje tablici segment alias **s**.
* GROUP BY **k.naziv\_klijenta**, **p.naziv\_projekta** grupira rezultate prema nazivu klijenta i nazivu projekta. Ovo osigurava da se broj segmenata računa za svaki pojedini projekt.

REZULTAT:



## UPIT 7

Prikaz svih projekata koje treba završiti u narednih 30 dana, uključujući informacije o klijentu i odgovornoj osobi

Zbog planiranja godišnjih odmora, potrebno je utvrditi koje projekte je potrebno dovršiti u narednih 30 dana. Uz to, voditelj je zatražio i informacije o klijentu i odgovornoj osobi za te projekte.

TRAŽENO RJEŠENJE:

**id\_projekt, naziv\_projekta, rok\_izrade, naziv\_klijenta, odgovorna\_osoba**

KOD ZA UPIT:

SELECT p.id AS id\_projekta, p.naziv\_projekta, p.rok\_izrade, k.naziv\_klijenta, k.odgovorna\_osoba

FROM projekt AS p

JOIN klijent AS k ON p.id\_klijent = k.id

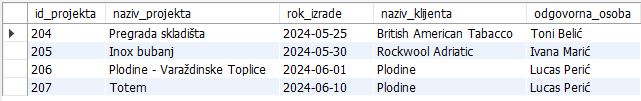
WHERE p.rok\_izrade BETWEEN CURDATE() AND DATE\_ADD(CURDATE(), INTERVAL 30 DAY);

OPIS UPITA:

Ovaj upit pretražuje bazu podataka kako bi pronašao sve projekte koje je potrebno dovršiti u narednih 30 dana. Spaja tablice **projekt** i **klijent** kako bi uključio informacije o klijentu i odgovornoj osobi za svaki projekt. Konačni rezultat je popis projekata s njihovim ID-evima, nazivima, rokovima izrade, nazivima klijenata i odgovornim osobama, što omogućava planiranje rada i godišnjih odmora s obzirom na nadolazeće rokove.

* SELECT **p.id** AS **id\_projekta** odabire **id** projekta iz tablice **projekt** i daje mu alias **id\_projekta**. **p.naziv\_projekta** odabire naziv projekta iz tablice **projekt**. **p.rok\_izrade** odabire rok izrade projekta iz tablice **projekt**. **k.naziv\_klijenta** odabire naziv klijenta iz tablice **klijent**. **k.odgovorna\_osoba** odabire odgovornu osobu iz tablice **klijent**.
* FROM **projekt** AS **p** specificira tablicu **projekt** kao početnu točku za upit i daje joj alias **p**.
* JOIN **klijent** AS **k** ON **p.id\_klijent = k.id** povezuje tablicu **klijent** s tablicom **projekt** na temelju **id\_klijent** iz tablice **projekt** i **id** iz tablice **klijent**, te daje tablici **klijent** alias **k**.
* WHERE **p.rok\_izrade** BETWEEN CURDATE() AND DATE\_ADD(CURDATE(), INTERVAL 30 DAY) filtrira projekte prema roku izrade tako da se prikažu samo oni projekti čiji je rok izrade između današnjeg datuma (CURDATE()) i datuma koji je 30 dana od današnjeg dana (DATE\_ADD(CURDATE(), INTERVAL 30 DAY))

REZULTAT:



## UPIT 8

Ukupan trošak materijala po projektu

Zbog mjesečne inventure, koordinator za nabavu materijala je zatražio ukupan trošak materijala po pojedinom projektu.

TRAŽENO RJEŠENJE:

**naziv\_projekta, ukupan\_trosak\_materijala, naziv\_klijenta**

KOD ZA UPIT:

SELECT klijent.naziv\_klijenta, projekt.naziv\_projekta, SUM(materijal.cijena\_po\_komadu \* utrosak\_materijala.kolicina) AS ukupno\_trosak\_materijala

FROM projekt

JOIN utrosak\_materijala ON projekt.id = utrosak\_materijala.id\_projekt

JOIN materijal ON utrosak\_materijala.id\_materijal = materijal.id

JOIN klijent ON projekt.id\_klijent = klijent.id

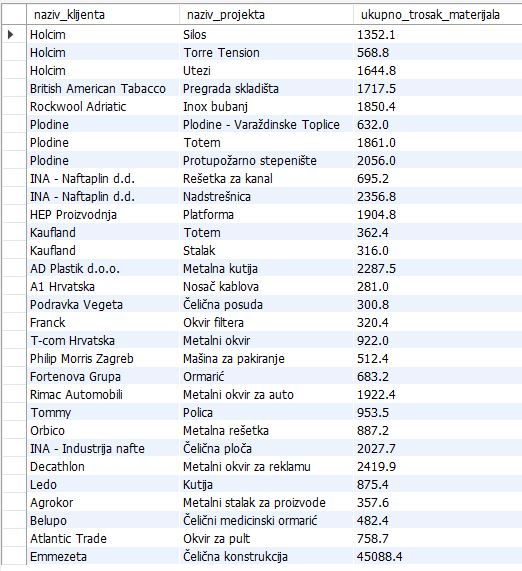
GROUP BY projekt.id;

OPIS UPITA:

Ovaj upit pretražuje bazu podataka kako bi izračunao ukupan trošak materijala za svaki projekt i povezanog klijenta. Prvo spaja tablice **projekt**, **utrosak\_materijala**, **materijal**, i **klijent** na temelju zajedničkih ključeva. Zatim koristi funkciju SUM za zbrajanje troškova materijala po projektu, gdje se trošak izračunava množenjem cijene po komadu materijala sa količinom utrošenog materijala za taj projekt. Konačni rezultat je popis projekata s njihovim nazivima, nazivima klijenata, i ukupnim troškovima materijala. Ove informacije omogućuju koordinatoru za nabavu materijala da provede mjesečnu inventuru i prati troškove po projektima.

* SELECT **klijent.naziv\_klijenta** odabire naziv klijenta iz tablice **klijent**. **projekt.naziv\_projekta** odabire naziv projekta iz tablice **projekt**.
* SUM(**materijal.cijena\_po\_komadu \* utrosak\_materijala.kolicina**) AS **ukupno\_trosak\_materijala** izračunava ukupan trošak materijala za svaki projekt. Množi cijenu po komadu materijala sa količinom materijala utrošenog za taj projekt, a zatim sve te vrijednosti zbraja za svaki projekt.
* FROM **projekt** specificira tablicu **projekt** kao početnu točku za upit.
* JOIN **utrosak\_materijala** ON **projekt.id = utrosak\_materijala.id\_projekt** povezuje tablicu **utrosak\_materijala** s tablicom **projekt** na temelju **id\_projekt** iz tablice **utrosak\_materijala** i **id** iz tablice **projekt**.
* JOIN **materijal** ON **utrosak\_materijala.id\_materijal = materijal.id** povezuje tablicu materijal s tablicom **utrosak\_materijala** na temelju **id\_materijal** iz tablice **utrosak\_materijala** i **id** iz tablice **materijal**.
* JOIN **klijent** ON **projekt.id\_klijent = klijent.id** povezuje tablicu **klijent** s tablicom projekt na temelju **id\_klijent** iz tablice **projekt** i **id** iz tablice **klijent**.
* GROUP BY **projekt.id** grupira rezultate prema **id** iz tablice **projekt**. Ovaj korak je potreban kako bi se dobio ukupan trošak materijala za svaki pojedini projekt.

REZULTAT:



## UPIT 9

Svi projekte koji su prekoračili rok izrade

Upravitelj je dobio mail nezadovoljnog klijenta zbog kašnjenja isporuke projekta. Projektni menadžer tražio nas je popis svih projekata koji su prekoračili rok izrade kako bi mogao lakše procjeniti od kojeg nezadovoljnog klijenta je stigla kritika.

TRAŽENO RJEŠENJE:

**naziv\_klijenta, naziv\_projekta, naziv\_segmenta, kolicina, rok\_izrade**

KOD ZA UPIT:

SELECT DISTINCT k.naziv\_klijenta, p.naziv\_projekta, s.naziv\_segmenta, s.kolicina, p.rok\_izrade

FROM projekt AS p

JOIN segment AS s ON p.id = s.id\_projekt

JOIN proizvodnja AS pr ON pr.id\_segment = s.id

JOIN klijent AS k ON k.id = p.id\_klijent

WHERE p.rok\_izrade < CURDATE() AND pr.status\_proizvodnje = 'u izradi';

OPIS UPITA:

Ovaj upit pretražuje bazu podataka kako bi pronašao sve projekte koji su prekoračili rok izrade i trenutno su u izradi. Prvo spaja tablice **projekt**, **segment**, **proizvodnja**, i **klijent** na temelju zajedničkih ključeva. Zatim filtrira projekte čiji je rok izrade prošao i trenutno su u statusu 'u izradi'. Konačni rezultat je popis jedinstvenih projekata s nazivima klijenata, nazivima projekata, nazivima segmenata, količinama i rokovima izrade. Ovaj popis omogućava projektnom menadžeru da lakše identificira projekte koji kasne i potencijalno nezadovoljne klijente.

* SELECT DISTINCT koristi se za odabir jedinstvenih redaka, eliminirajući duplikate iz rezultata. **k.naziv\_klijenta** odabire naziv klijenta iz tablice **klijent**. **p.naziv\_projekta** odabire naziv projekta iz tablice **projekt**. **s.naziv\_segmenta** odabire naziv segmenta iz tablice **segment**. **s.kolicina** odabire količinu iz tablice **segment**. **p.rok\_izrade** odabire rok izrade iz tablice **projekt**.
* FROM **projekt** AS **p** specificira tablicu **projekt** kao početnu točku za upit.
* JOIN **segment** AS **s** ON **p.id = s.id\_projekt** povezuje tablicu **segment** s tablicom **projekt** na temelju **id\_projekt** iz tablice **segment** i **id** iz tablice **projekt**.
* JOIN **proizvodnja** AS **pr** ON **pr.id\_segment = s.id** povezuje tablicu **proizvodnja** s tablicom **segment** na temelju **id\_segment** iz tablice **proizvodnja** i **id** iz tablice **segment**.
* JOIN **klijent** AS **k** ON **k.id = p.id\_klijent** povezuje tablicu **klijent** s tablicom **projekt** na temelju **id** iz tablice **klijent** i **id\_klijent** iz tablice **projekt**.
* WHERE **p.rok\_izrade <** CURDATE() filtrira projekte čiji je rok izrade prošao (manji od trenutnog datuma). AND **pr.status\_proizvodnje = 'u** izradi' filtrira samo one segmente koji su trenutno u izradi.

REZULTAT:



## UPIT 10

Ukupna količina materijala na lageru

Dobili smo poveći broj zahtjeva za nove projekte, voditelj traži popis i količinu materijala na lageru kako bi znao koje projekte prihvatiti a koje mora ostaviti na čekanju dok ne stigne još materijala.

TRAŽENO RJEŠENJE:

**naziv\_materijala, ukupna\_kolicina, napomena**

KOD ZA UPIT:

SELECT materijal.naziv\_materijala, SUM(lager\_materijala.kolicina) AS ukupna\_kolicina, materijal.napomena

FROM materijal

JOIN lager\_materijala ON materijal.id = lager\_materijala.id\_materijal

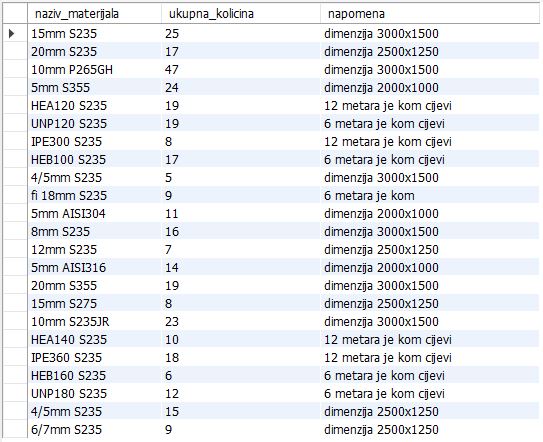
GROUP BY materijal.id;

OPIS UPITA:

Ovaj upit prebrojava ukupnu količinu svakog materijala koji je na lageru, grupirajući rezultate prema tipu materijala. Prvo spaja tablice **materijal** i **lager\_materijala** na temelju zajedničkog **id\_materijal** i **id**. Zatim koristi SUM za zbrajanje količina materijala za svaki tip i prikazuje naziv materijala, ukupnu količinu i napomenu o materijalu. Konačni rezultat je popis materijala s njihovim ukupnim količinama na lageru, što pomaže voditelju da donese odluke o prihvaćanju novih projekata na temelju dostupnosti materijala.

* SELECT **materijal.naziv\_materijala,** SUM**(lager\_materijala.kolicina)** AS **ukupna\_kolicina, materijal.napomena** nam odabire **naziv\_materijala** i **napomena** iz tablice **materijal** uz to koristi agregatnu funkciju SUM za zbrajanje količine materijala iz tablice **lager\_materijala** i daje toj sumi alias **ukupna\_kolicina**.
* FROM **materijal** nam specificira tablicu materijal kao izvor podataka.
* JOIN **lager\_materijala** ON **materijal.id = lager\_materijala.id\_materijal** koristi JOIN za spajanje dviju tablica. Povezuje tablicu **materijal** s tablicom **lager\_materijala** koristeći **id** iz tablice **materijal** i **id\_materijal** iz tablice **lager\_materijala**.
* GROUP BY **materijal.id** grupira rezultate prema **id** iz tablice **materijal**. To znači da će se za svaki materijal izračunati ukupna količina.

REZULTAT:



## UPIT 11

Prikaz ukupnog troška rada zaposlenika po projektu

Radi se evidencija ukupnog troska rada zaposlenika po projektu.

TRAŽENO RJEŠENJE:

**naziv\_projekta, ukupan\_trosak\_rada**

KOD ZA UPIT:

SELECT p.naziv\_projekta, SUM(us.utrosak\_sati \* z.cijena\_po\_satu) AS ukupni\_trosak\_rada

FROM projekt AS p

JOIN segment AS s ON p.id = s.id\_projekt

JOIN proizvodnja AS pr ON s.id = pr.id\_segment

JOIN utrosak\_sati AS us ON pr.id = us.id\_proizvodnja

JOIN zaposlenik AS z ON us.id\_zaposlenik = z.id

GROUP BY p.naziv\_projekta

ORDER BY ukupni\_trosak\_rada DESC;

**OPIS UPITA:**

Ovaj upit koristi nekoliko JOIN operacija za pridruživanje više tablica i dobivanje potrebnih podataka. Cilj je izračunati ukupni trošak rada za svaki projekt. Kroz više JOIN operacija, upit spaja tablice **projekt**, **segment**, **proizvodnja**, **utrosak\_sati**, i **zaposlenik** kako bi se dobili podaci o nazivima projekata, utrošenim satima rada, cijenama po satu zaposlenika, te ukupnim troškovima rada. Rezultati su grupirani po nazivima projekata i sortirani prema ukupnim troškovima rada u silaznom redoslijedu.

#### SELECT p.naziv\_projekta, SUM(us.utrosak\_sati \* z.cijena\_po\_satu) AS ukupni\_trosak\_rada

Ova naredba odabire dva podatka:

* **p.naziv\_projekta**: Naziv projekta iz tablice projekt.
* **SUM(us.utrosak\_sati \* z.cijena\_po\_satu) AS ukupni\_trosak\_rada**: Izračunava ukupni trošak rada za svaki projekt. Ovdje se množe utrošeni sati (us.utrosak\_sati) s cijenom po satu zaposlenika (z.cijena\_po\_satu) i zbrajaju rezultati za sve zaposlenike koji su radili na projektu.

#### FROM projekt AS p

Definira tablicu projekt kao osnovnu tablicu za upit i dodjeljuje joj alias p za lakše referenciranje.

JOIN segment AS s ON p.id = s.id\_projekt

Pridružuje tablicu segment s tablicom projekt na temelju podudaranja stupaca id\_projekt iz tablice segment i id iz tablice projekt. Ovo znači da se za svaki projekt pridružuju svi njegovi segmenti.

JOIN proizvodnja AS pr ON s.id = pr.id\_segment

Pridružuje tablicu proizvodnja s tablicom segment na temelju podudaranja stupaca id\_segment iz tablice proizvodnja i id iz tablice segment. Ovo znači da se za svaki segment pridružuju sve faze proizvodnje koje se odnose na taj segment.

JOIN utrosak\_sati AS us ON pr.id = us.id\_proizvodnja

Pridružuje tablicu utrosak\_sati s tablicom proizvodnja na temelju podudaranja stupaca id\_proizvodnja iz tablice utrosak\_sati i id iz tablice proizvodnja. Ovo znači da se za svaku fazu proizvodnje pridružuju svi zapisi o utrošenim satima rada zaposlenika.

JOIN zaposlenik AS z ON us.id\_zaposlenik = z.id

Pridružuje tablicu zaposlenik s tablicom utrosak\_sati na temelju podudaranja stupaca id\_zaposlenik iz tablice utrosak\_sati i id iz tablice zaposlenik. Ovo znači da se za svaki zapis o utrošenim satima pridružuju podaci o odgovarajućem zaposleniku.

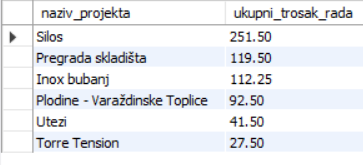
GROUP BY p.naziv\_projekta

Grupira rezultate prema nazivu projekta (p.naziv\_projekta). Ovo omogućuje agregaciju podataka (korištenje funkcije SUM) za svaki projekt zasebno.

ORDER BY ukupni\_trosak\_rada DESC;

Sortira rezultate prema izračunatom ukupnom trošku rada u silaznom redoslijedu (DESC). To znači da će projekti s najvećim ukupnim troškom rada biti prikazani prvi.

REZULTAT:



## UPIT 12

Prikaz materijala koji se najviše koristi po segmentima

Radi se evidencija utroska materijala po segmentima

TRAŽENO RJEŠENJE:

**naziv\_segmenta, naziv\_materijala, ukupno\_koristeno**

KOD ZA UPIT:

SELECT s.naziv\_segmenta, m.naziv\_materijala, SUM(um.kolicina) AS ukupno\_koristeno

FROM segment AS s

JOIN utrosak\_materijala AS um ON s.id\_projekt = um.id\_projekt

JOIN materijal AS m ON um.id\_materijal = m.id

GROUP BY s.naziv\_segmenta, m.naziv\_materijala

ORDER BY ukupno\_koristeno DESC;

**OPIS UPITA:**

Ovaj upit koristi nekoliko **JOIN** operacija za pridruživanje više tablica i dobivanje potrebnih podataka. Cilj je izračunati ukupnu količinu korištenog materijala za svaki segment. Kroz više **JOIN** operacija, upit spaja tablice **segment**, **utrosak\_materijala**, i **materijal** kako bi se dobili podaci o nazivima segmenata, nazivima materijala, i ukupnoj količini korištenog materijala. Rezultati su grupirani po nazivima segmenata i materijala, te sortirani prema ukupnoj količini korištenog materijala u silaznom redoslijedu.

**SELECT s.naziv\_segmenta, m.naziv\_materijala, SUM(um.kolicina) AS ukupno\_koristeno**

Ova naredba odabire tri podatka:

* **s.naziv\_segmenta**: Naziv segmenta iz tablice segment.
* **m.naziv\_materijala**: Naziv materijala iz tablice materijal.
* **SUM(um.kolicina) AS ukupno\_koristeno**: Izračunava ukupnu količinu materijala korištenog za svaki segment. Funkcija SUM zbraja količine materijala (um.kolicina) za svaki segment i materijal.

**FROM segment AS s**

Definira tablicu segment kao osnovnu tablicu za upit i dodjeljuje joj alias s za lakše referenciranje.

**JOIN utrosak\_materijala AS um ON s.id\_projekt = um.id\_projekt**

Pridružuje tablicu utrosak\_materijala s tablicom segment na temelju podudaranja stupaca id\_projekt iz tablice segment i id\_projekt iz tablice utrosak\_materijala. Ovo znači da se za svaki segment pridružuju svi zapisi o utrošku materijala koji pripadaju istom projektu.

**JOIN materijal AS m ON um.id\_materijal = m.id**

Pridružuje tablicu materijal s tablicom utrosak\_materijala na temelju podudaranja stupaca id\_materijal iz tablice utrosak\_materijala i id iz tablice materijal. Ovo znači da se za svaki utrošeni materijal pridružuju podaci o materijalu.

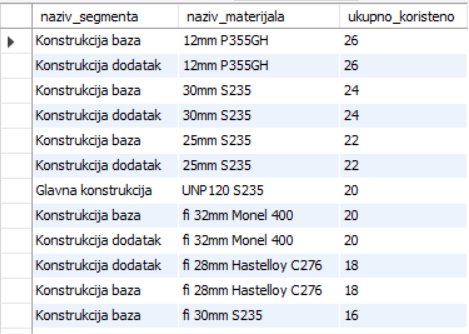
**GROUP BY s.naziv\_segmenta, m.naziv\_materijala**

Grupira rezultate prema nazivu segmenta (s.naziv\_segmenta) i nazivu materijala (m.naziv\_materijala). Ovo omogućuje agregaciju podataka (korištenje funkcije SUM) za svaki segment i materijal zasebno.

**ORDER BY ukupno\_koristeno DESC**

Sortira rezultate prema izračunatoj ukupnoj količini korištenog materijala u silaznom redoslijedu (DESC). To znači da će segmenti i materijali s najvećom ukupnom količinom korištenog materijala biti prikazani prvi.

REZULTAT:



Napomena: Tablica ima 88 redova ,ali zbog veličine samo zalijepio prvih desetak redova.

## UPIT 13

#### Prikaz alata koji su najviše zaduživani od strane zaposlenika

Radi se evidencija najviše korištenih alata

TRAŽENO RJEŠENJE:

**naziv\_alata, broj\_zaduzenja**

KOD ZA UPIT:

SELECT a.naziv\_alata, COUNT(za.id) AS broj\_zaduzenja

FROM alat AS a

JOIN zaduzenje\_alata AS za ON a.id = za.id\_alat

GROUP BY a.naziv\_alata

ORDER BY broj\_zaduzenja DESC;

**OPIS UPITA:**

Ovaj upit koristi **JOIN** operaciju za pridruživanje tablica **alat** i **zaduzenje\_alata** kako bi se dobili potrebni podaci. Cilj je izračunati koliko puta je svaki alat zadužen. Kroz **JOIN** operaciju, upit spaja tablice **alat** i **zaduzenje\_alata** na temelju podudaranja ID-ova alata.

**SELECT a.naziv\_alata, COUNT(za.id) AS broj\_zaduzenja**

Ova naredba odabire dva podatka:

* **a.naziv\_alata**: Naziv alata iz tablice alat.
* **COUNT(za.id) AS broj\_zaduzenja**: Izračunava broj zaduženja za svaki alat. Funkcija COUNT broji koliko puta se svaki alat pojavljuje u tablici zaduzenje\_alata.

**FROM alat AS a**

Definira tablicu alat kao osnovnu tablicu za upit i dodjeljuje joj alias a za lakše referenciranje.

**JOIN zaduzenje\_alata AS za ON a.id = za.id\_alat**

Pridružuje tablicu zaduzenje\_alata s tablicom alat na temelju podudaranja stupaca id iz tablice alat i id\_alat iz tablice zaduzenje\_alata. Ovo znači da se za svaki alat pridružuju svi zapisi o njegovom zaduženju.

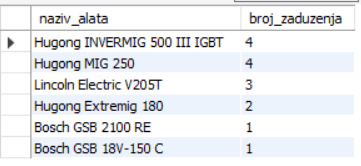
**GROUP BY a.naziv\_alata**

Grupira rezultate prema nazivu alata (a.naziv\_alata). Ovo omogućuje agregaciju podataka (korištenje funkcije COUNT) za svaki alat zasebno.

**ORDER BY broj\_zaduzenja DESC**

Sortira rezultate prema izračunatom broju zaduženja u silaznom redoslijedu (DESC). To znači da će alati s najvećim brojem zaduženja biti prikazani prvi.

REZULTAT:



## UPIT 14

Prikaz svih projekata po statusu proizvodnje

trazeni rezukltat:

naziv\_projekt, status\_proizvodnje

kod :

SELECT p.naziv\_projekta, pr.status\_proizvodnje

FROM projekt AS p

JOIN segment AS s ON p.id = s.id\_projekt

JOIN proizvodnja AS pr ON s.id = pr.id\_segment

GROUP BY p.naziv\_projekta, pr.status\_proizvodnje

ORDER BY p.naziv\_projekta, pr.status\_proizvodnje;

Opis:

Ovaj upit je koristan za dobivanje pregleda o stanju proizvodnje svih projekata. Može se koristiti za identifikaciju koliko segmenata je trenutno u određenom statusu proizvodnje (npr. "u izradi", "završeno") za svaki projekt. Na taj način, menadžeri mogu pratiti napredak svakog projekta i vidjeti koji projekti ili segmenti zahtijevaju više pažnje ili resursa.

Objasnjenje koda:

SELECT p.naziv\_projekta, pr.status\_proizvodnje

Ovaj dio upita odabire dva podatka:

p.naziv\_projekta: Naziv projekta iz tablice projekt.

pr.status\_proizvodnje: Status proizvodnje iz tablice proizvodnja.

FROM projekt AS p

Definira tablicu projekt kao osnovnu tablicu za upit i dodjeljuje joj alias p za lakše referenciranje.

JOIN segment AS s ON p.id = s.id\_projekt

Pridružuje tablicu segment s tablicom projekt na temelju podudaranja stupaca id\_projekt iz tablice segment i id iz tablice projekt. Ovo znači da se za svaki projekt pridružuju svi njegovi segmenti.

JOIN segment AS s ON p.id = s.id\_projekt: Ovo spaja tablicu projekt s tablicom segment, povezujući projekte s njihovim segmentima.

JOIN proizvodnja AS pr ON s.id = pr.id\_segment

Pridružuje tablicu proizvodnja s tablicom segment na temelju podudaranja stupaca id\_segment iz tablice proizvodnja i id iz tablice segment. Ovo znači da se za svaki segment pridružuju sve faze proizvodnje koje se odnose na taj segment.

JOIN proizvodnja AS pr ON s.id = pr.id\_segment: Ovo spaja tablicu segment s tablicom proizvodnja, povezujući segmente s njihovim proizvodnim fazama.

GROUP BY p.naziv\_projekta, pr.status\_proizvodnje

Grupira rezultate prema nazivu projekta (p.naziv\_projekta) i statusu proizvodnje (pr.status\_proizvodnje). Ovo omogućuje agregaciju podataka (korištenje funkcije COUNT) za svaki projekt i status proizvodnje zasebno.

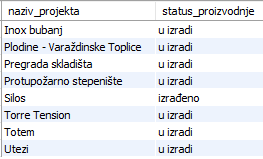
GROUP BY p.naziv\_projekta, pr.status\_proizvodnje: Ovo grupira rezultate tako da svaki jedinstveni par naziv\_projekta i status\_proizvodnje ima svoj vlastiti redak u rezultatu upita.

ORDER BY p.naziv\_projekta, pr.status\_proizvodnje

Sortira rezultate prema nazivu projekta i statusu proizvodnje. To znači da će rezultati biti prikazani abecednim redoslijedom po nazivu projekta, a zatim po statusu proizvodnje unutar svakog projekta.

ORDER BY p.naziv\_projekta, pr.status\_proizvodnje: Ovo sortira rezultate prema nazivu projekta, a zatim prema statusu proizvodnje unutar svakog projekta.

Rezultat:



## UPIT 15

Prikaz zaduženih alata po zaposlenicima

Trazeni rezultat:

ime,prezime,naziv\_alata,broj\_zaduzenja

Kod:

SELECT z.ime, z.prezime, a.naziv\_alata, COUNT(za.id) AS broj\_zaduzenja

FROM zaposlenik AS z

JOIN zaduzenje\_alata AS za ON z.id = za.id\_zaposlenik

JOIN alat AS a ON za.id\_alat = a.id

GROUP BY z.ime, z.prezime, a.naziv\_alata

ORDER BY z.prezime, z.ime, broj\_zaduzenja DESC;

Opis:

Ovaj upit je koristan za dobivanje pregleda o tome koji su alati najviše zaduživani od strane zaposlenika. Može se koristiti za identifikaciju zaposlenika koji najviše koriste određene alate, kao i alata koji su najviše korišteni. Na taj način, menadžeri mogu pratiti upotrebu alata i potencijalno planirati održavanje ili nabavu dodatnih alata.

Opis koda:

SELECT z.ime, z.prezime, a.naziv\_alata, COUNT(za.id) AS broj\_zaduzenja

Ovaj dio upita odabire četiri podatka:

z.ime: Ime zaposlenika iz tablice zaposlenik.

z.prezime: Prezime zaposlenika iz tablice zaposlenik.

a.naziv\_alata: Naziv alata iz tablice alat.

COUNT(za.id) AS broj\_zaduzenja: Izračunava broj zaduženja svakog alata po zaposleniku. Funkcija COUNT broji koliko puta se svaki alat pojavljuje u tablici zaduzenje\_alata.

FROM zaposlenik AS z

Definira tablicu zaposlenik kao osnovnu tablicu za upit i dodjeljuje joj alias z za lakše referenciranje.

JOIN zaduzenje\_alata AS za ON z.id = za.id\_zaposlenik

Pridružuje tablicu zaduzenje\_alata s tablicom zaposlenik na temelju podudaranja stupaca id iz tablice zaposlenik i id\_zaposlenik iz tablice zaduzenje\_alata. Ovo znači da se za svakog zaposlenika pridružuju svi zapisi o njegovim zaduženjima alata.

JOIN zaduzenje\_alata AS za ON z.id = za.id\_zaposlenik: Ovo spaja tablicu zaposlenik s tablicom zaduzenje\_alata, povezujući zaposlenike s njihovim zaduženjima alata.

JOIN alat AS a ON za.id\_alat = a.id

Pridružuje tablicu alat s tablicom zaduzenje\_alata na temelju podudaranja stupaca id iz tablice alat i id\_alat iz tablice zaduzenje\_alata. Ovo znači da se za svako zaduženje pridružuju podaci o alatu koji je zadužen.

JOIN alat AS a ON za.id\_alat = a.id: Ovo spaja tablicu zaduzenje\_alata s tablicom alat, povezujući zaduženja s podacima o alatima.

GROUP BY z.ime, z.prezime, a.naziv\_alata

Grupira rezultate prema imenu i prezimenu zaposlenika te nazivu alata. Ovo omogućuje agregaciju podataka (korištenje funkcije COUNT) za svaki alat po zaposleniku zasebno.

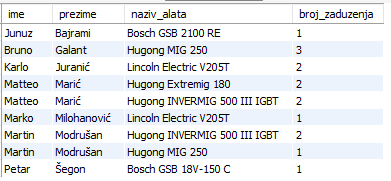
GROUP BY z.ime, z.prezime, a.naziv\_alata: Ovo grupira rezultate tako da svaki jedinstveni par ime, prezime, i naziv\_alata ima svoj vlastiti redak u rezultatu upita.

ORDER BY z.prezime, z.ime, broj\_zaduzenja DESC

Sortira rezultate prema prezimenu i imenu zaposlenika, a zatim prema broju zaduženja u silaznom redoslijedu. To znači da će rezultati biti prikazani abecednim redoslijedom po prezimenu i imenu zaposlenika, a zatim sortirani prema broju zaduženja alata u silaznom redoslijedu.

ORDER BY z.prezime, z.ime, broj\_zaduzenja DESC: Ovo sortira rezultate prema prezimenu i imenu zaposlenika, a zatim prema broju zaduženja u silaznom redoslijedu.

Rerezultat:



## UPIT 16

Prikaz broja zaposlenika po projektu

Trazeni rezultat:

naziv\_projekta, broj\_zaposlenika

Kod:

SELECT p.naziv\_projekta, COUNT(DISTINCT us.id\_zaposlenik) AS broj\_zaposlenika

FROM projekt AS p

JOIN segment AS s ON p.id = s.id\_projekt

JOIN proizvodnja AS pr ON s.id = pr.id\_segment

JOIN utrosak\_sati AS us ON pr.id = us.id\_proizvodnja

GROUP BY p.naziv\_projekta

ORDER BY broj\_zaposlenika DESC;

Opis:

Ovaj upit je koristan za dobivanje pregleda o tome koliko je zaposlenika radilo na svakom projektu. Može se koristiti za identifikaciju projekata koji zahtijevaju najviše radne snage i za planiranje resursa. Na taj način, menadžeri mogu pratiti angažman zaposlenika na različitim projektima i optimizirati dodjelu radne snage.

Opis koda:

SELECT p.naziv\_projekta, COUNT(DISTINCT us.id\_zaposlenik) AS broj\_zaposlenika

Ovaj dio upita odabire dva podatka:

p.naziv\_projekta: Naziv projekta iz tablice projekt.

COUNT(DISTINCT us.id\_zaposlenik) AS broj\_zaposlenika: Izračunava broj jedinstvenih zaposlenika koji su radili na svakom projektu. Funkcija COUNT(DISTINCT ...) broji broj jedinstvenih zaposlenika (us.id\_zaposlenik) za svaki projekt.

FROM projekt AS p

Definira tablicu projekt kao osnovnu tablicu za upit i dodjeljuje joj alias p za lakše referenciranje.

JOIN segment AS s ON p.id = s.id\_projekt

Pridružuje tablicu segment s tablicom projekt na temelju podudaranja stupaca id\_projekt iz tablice segment i id iz tablice projekt. Ovo znači da se za svaki projekt pridružuju svi njegovi segmenti.

JOIN segment AS s ON p.id = s.id\_projekt: Ovo spaja tablicu projekt s tablicom segment, povezujući projekte s njihovim segmentima.

JOIN proizvodnja AS pr ON s.id = pr.id\_segment

Pridružuje tablicu proizvodnja s tablicom segment na temelju podudaranja stupaca id\_segment iz tablice proizvodnja i id iz tablice segment. Ovo znači da se za svaki segment pridružuju sve faze proizvodnje koje se odnose na taj segment.

JOIN proizvodnja AS pr ON s.id = pr.id\_segment: Ovo spaja tablicu segment s tablicom proizvodnja, povezujući segmente s njihovim proizvodnim fazama.

JOIN utrosak\_sati AS us ON pr.id = us.id\_proizvodnja

Pridružuje tablicu utrosak\_sati s tablicom proizvodnja na temelju podudaranja stupaca id\_proizvodnja iz tablice utrosak\_sati i id iz tablice proizvodnja. Ovo znači da se za svaku fazu proizvodnje pridružuju svi zapisi o utrošenim satima rada zaposlenika.

JOIN utrosak\_sati AS us ON pr.id = us.id\_proizvodnja: Ovo spaja tablicu proizvodnja s tablicom utrosak\_sati, povezujući proizvodne faze s utrošenim satima rada zaposlenika.

GROUP BY p.naziv\_projekta

Grupira rezultate prema nazivu projekta (p.naziv\_projekta). Ovo omogućuje agregaciju podataka (korištenje funkcije COUNT) za svaki projekt zasebno.

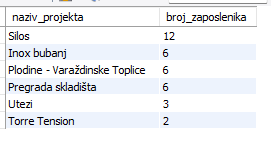
GROUP BY p.naziv\_projekta: Ovo grupira rezultate tako da svaki projekt ima svoj vlastiti redak u rezultatu upita.

ORDER BY broj\_zaposlenika DESC

Sortira rezultate prema broju zaposlenika u silaznom redoslijedu (DESC). To znači da će projekti s najvećim brojem zaposlenika biti prikazani prvi.

ORDER BY broj\_zaposlenika DESC: Ovo sortira rezultate prema broju zaposlenika u silaznom redoslijedu.

Rezlutat:



# ZAKLJUČAK

Smatramo da smo napravili solidnu bazu podataka za temu 'Organizacija u proizvodnji metalnih konstrukcija'. Naravno, svi smo svjesni da ima još mjesta za napredak. Ova tema je veoma opširna i još uvijek postoji mnogo situacija koje smo zanemarili u ovom projektu. Isto tako smatramo da smo ispunili sve zahtjeve prema pravilima za izradu projekta. Za izradu ovog projekta koristili smo:

* za komunikaciju: Discord , Whatsapp , Zoom
* za pisanje SQL skripte: MySQL Workbench
* za kreiranje ER dijagrama: Lucidchart
* za izradu dokumentacije: GitHub ,MS Office Word