Sveučilište Jurja Dobrile u Puli

Fakultet informatike u Puli



**TIM 4: Webshop informatičke opreme**

Loren Bažon

Morena Martan

Bruno Rebić

Fran Barba

Josip Milković

Leo Hrvojić

Smjer : Informatika

Kolegij : Baze podataka II

Mentor : doc. Dr. sc. Goran Oreški

Asistent : mag. inf. Romeo Šajina

Pula, studeni, 2024. godina

Sadržaj

[Opis poslovnog procesa 4](#_gjdgxs)

[ER dijagram 5](#_30j0zll)

[Opis ER dijagrama 5](#_1fob9te)

[Kombinacija shema 5](#_3znysh7)

[EER DIJAGRAM (MYSQL WORKBENCH) 5](#_2et92p0)

[Tablice 6](#_tyjcwt)

[tablice Loren 6](#_3dy6vkm)

[tablice Morena 9](#_1t3h5sf)

[tablice Bruno 9](#_4d34og8)

[tablice Fran 9](#_2s8eyo1)

[tablice Josip 9](#_17dp8vu)

[tablice Leo 9](#_3rdcrjn)

[Upiti i pogledi 10](#_26in1rg)

[Upiti i pogledi Loren 10](#_lnxbz9)

[Upiti i pogledi Morena 14](#_35nkun2)

[Upiti i pogledi Bruno 14](#_1ksv4uv)

[Upiti i pogledi Fran 14](#_44sinio)

[Upiti i pogledi Josip 14](#_2jxsxqh)

[Upiti i pogledi Leo 14](#_z337ya)

[Funkcije 14](#_3j2qqm3)

[Funkcije Loren 14](#_1y810tw)

[Funkcije Morena 15](#_4i7ojhp)

[Funkcije Bruno 15](#_2xcytpi)

[Funkcije Fran 15](#_1ci93xb)

[Funkcije Josip 16](#_3whwml4)

[Funkcije Leo 16](#_2bn6wsx)

[Procedure 16](#_qsh70q)

[Procedure Loren 16](#_3as4poj)

[Procedure Morena 19](#_1pxezwc)

[Procedure Bruno 19](#_49x2ik5)

[Procedure Fran 19](#_2p2csry)

[Procedure Josip 19](#_147n2zr)

[Procedure Leo 20](#_3o7alnk)

[Okidači 20](#_23ckvvd)

[Okidači Loren 20](#_ihv636)

[Okidači Morena 21](#_32hioqz)

[Okidači Bruno 22](#_41mghml)

[Okidači Fran 22](#_2grqrue)

[Okidači Josip 22](#_vx1227)

[Okidači Leo 22](#_3fwokq0)

[Transakcije 22](#_1v1yuxt)

# Opis poslovnog procesa

Webshop informatičke opreme omogućava korisnicima kupnju proizvoda te upravljanje narudžbama kroz jednostavan i intuitivan proces. Glavne aktivnosti obuhvaćaju pregled proizvoda, dodavanje u košaricu, prijavu korisnika, potvrdu narudžbe, plaćanje, dostavu te upravljanje povratima i listom želja. Webshop također osigurava administraciju proizvoda, promocija i narudžbi, dok analitički alati omogućavaju donošenje poslovnih odluka na temelju podataka. Korisnik dolazi na početnu stranicu webshopa, gdje mu se prikazuju promotivni proizvodi, kategorije i preporuke. Proizvodi su rangirani prema popularnosti, popustima ili novitetima. Prijavljeni korisnici vide personalizirane preporuke temeljene na njihovim preferencijama i povijesti kupovine (tablica *preporuceni\_proizvodi*). Korisnici mogu pregledavati proizvode kroz kategorije (*kategorije\_proizvoda*) ili putem tražilice. Klikom na proizvod otvara se detaljna stranica s informacijama o proizvodu (*proizvodi*), recenzijama (*recenzije\_proizvoda*), popustima (*popusti*) i promotivnim kodovima (*kuponi*). Klikom na "Dodaj u košaricu" sustav provjerava dostupnost proizvoda (*kolicina\_na\_skladistu*).Za prijavljenog korisnika stavke se vežu uz njegov *korisnik\_id* u tablici *kosarica*. Neprijavljeni korisnici imaju privremeni *sesija\_id* za pohranu košarice.Korisnik može pregledavati košaricu, mijenjati količine, uklanjati proizvode te vidjeti ukupnu cijenu s popustima i troškovima isporuke (*popusti*, *nacini\_isporuke*). Ako korisnik nije prijavljen, mora kreirati račun ili se prijaviti (*Korisnici*). Sustav validira podatke (npr., jedinstveni email). Prijavljeni korisnici automatski imaju povezane stavke iz košarice s njihovim računom. Korisnik odabire adresu isporuke (*korisnici*, atribut *adresa*) i način dostave (*nacini\_isporuke*). Sustav izračunava ukupnu cijenu te omogućava unos kupona koji umanjuje cijenu (*kuponi*). Klikom na "Potvrdi narudžbu" stvara se zapis u tablici *narudzbe*, a stavke iz košarice prenose se u tablicu *stavke\_narudzbe*. Količina proizvoda na skladištu se ažurira. Korisnik bira način plaćanja, a sustav validira transakciju putem vanjskih servisa (npr., PayPal). Nakon uspješnog plaćanja kreira se račun (*racuni*) koji korisnik može preuzeti. Status narudžbe (*status\_narudzbe*) postavlja se na "u obradi". Korisnik putem webshopa prati status dostave kroz *pracenje\_isporuka*. Korisnik podnosi zahtjev za povrat proizvoda, koji administratori pregledavaju. Status povrata (*status\_povrata*) ažurira se u tablici *povrati\_proizvoda*, a povrat sredstava se obavlja kroz tablicu *placanja*. Korisnici mogu spremiti proizvode za kasniju kupnju u tablicu *Wishlist*. Sustav obavještava korisnika o promjenama cijena proizvoda s liste želja. Administratori upravljaju proizvodima (*proizvodi*), kategorijama (*kategorije\_proizvoda*), popustima (*popusti*), kuponima (*kuponi*) i narudžbama (*narudzbe*). Generiraju se analitička izvješća o prodaji i uspješnosti promocija temeljem podataka iz tablica *narudzbe*, *stavke\_narudzbe* i *recenzije\_proizvoda*. Sustav prati prodaju, povrate i recenzije kako bi unaprijedio ponudu i personalizirao preporuke (*preporuceni\_proizvodi*). Podaci o aktivnostima korisnika (*aktivnosti\_korisnika*) omogućavaju donošenje boljih poslovnih odluka.

# ER dijagram

# Opis ER dijagrama

# Kombinacija shema

# EER DIJAGRAM (MYSQL WORKBENCH)

# Tablice

## tablice Loren

**CREATE TABLE** kategorije\_proizvoda (

id **INT AUTO\_INCREMENT PRIMARY KEY,**

naziv **VARCHAR(255) NOT NULL UNIQUE,**

opis **TEXT NOT NULL**

);

Ova tablica se naziva „kategorije\_proizvoda“ i sadrži sljedeće atribute:

* „id“ je cjelobrojni atribut koji se automatski povećava svaki put kada se unese novi red u tablicu. Služi kao primarni ključ i osigurava jedinstvenu identifikaciju svake kategorije.
* „naziv“ je tekstualni atribut (VARCHAR) s maksimalnom duljinom od 255 znakova. Sadrži naziv kategorije i mora biti jedinstven kako bi se spriječilo dupliciranje naziva kategorija.
* „opis“ je tekstualni atribut (TEXT) koji omogućuje pohranu detaljnog opisa kategorije.

Tablica također ima sljedeća ograničenja:

* atribut „id“ postavljen je kao primarni ključ tablice.
* atribut „naziv“ mora sadržavati jedinstvene vrijednosti, čime se osigurava da svaka kategorija ima jedinstveno ime.

**CREATE TABLE** proizvodi (

id **INT AUTO\_INCREMENT PRIMARY KEY,**

naziv **VARCHAR(255) NOT NULL,**

opis **TEXT NOT NULL,**

cijena **DECIMAL(10, 2) NOT NULL,**

kategorija\_id **INT NOT NULL,**

kolicina\_na\_skladistu **INT NOT NULL,**

slika **VARCHAR(255),**

specifikacije **TEXT,**

datum\_kreiranja **TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,**

**CONSTRAINT** chk\_cijena\_proizvodi **CHECK (cijena > 0),**

**CONSTRAINT** chk\_kolicina\_proizvodi **CHECK (kolicina\_na\_skladistu >= 0),**

**CONSTRAINT** fk\_kategorija\_proizvoda\_proizvodi **FOREIGN KEY** (kategorija\_id) **REFERENCES** kategorije\_proizvoda(id)

);

Ova tablica se naziva „proizvodi“ i sadrži sljedeće atribute:

* „id“ je cjelobrojni atribut koji se automatski povećava svaki put kada se unese novi red u tablicu. Služi kao primarni ključ i osigurava jedinstvenu identifikaciju svakog proizvoda.
* „naziv“ je tekstualni atribut (VARCHAR) s maksimalnom duljinom od 255 znakova. Sadrži naziv proizvoda.
* „opis“ je tekstualni atribut (TEXT) koji sadrži detaljan opis proizvoda.
* „cijena“ je decimalni atribut (DECIMAL(10, 2)) koji bilježi cijenu proizvoda s do dvije decimalne znamenke.
* „kategorija\_id“ je cjelobrojni atribut koji se odnosi na primarni ključ u tablici kategorije\_proizvoda. Označava kojoj kategoriji proizvod pripada.
* „kolicina\_na\_skladistu“ je cjelobrojni atribut koji prikazuje broj dostupnih proizvoda na skladištu.
* „slika“ je tekstualni atribut (VARCHAR) koji sadrži URL ili putanju do slike proizvoda.
* „specifikacije“ je tekstualni atribut (TEXT) koji pohranjuje dodatne informacije o proizvodu, poput tehničkih specifikacija.
* „datum\_kreiranja“ je vrijeme kada je proizvod kreiran, s podrazumijevanom vrijednošću trenutnog vremena (CURRENT\_TIMESTAMP).

Tablica također ima sljedeća ograničenja:

* atribut „id“ postavljen je kao primarni ključ.
* atribut „kategorija\_id“ povezan je s tablicom „kategorije\_proizvoda“, čime se osigurava referencijalna integracija.
* atribut „cijena“ mora biti veća od 0, što je osigurano ograničenjem CHECK.
* atribut „kolicina\_na\_skladistu“ mora biti veća ili jednaka 0, što je osigurano ograničenjem CHECK.

**CREATE TABLE** recenzije\_proizvoda (

id **INT AUTO\_INCREMENT PRIMARY KEY,**

proizvod\_id **INT NOT NULL,**

korisnik\_id **INT NOT NULL,**

ocjena **INT NOT NULL,**

komentar **TEXT,**

datum\_recenzije **TIMESTAMP DEFAULT CURRENT\_TIMESTAMP**,

**CONSTRAINT** chk\_ocjena\_recenzije\_proizvoda **CHECK** (ocjena **BETWEEN** 1 **AND** 5),

**CONSTRAINT** fk\_proizvod\_recenzija\_recenzije\_proizvoda **FOREIGN KEY** (proizvod\_id) **REFERENCES** proizvodi(id),

**CONSTRAINT** fk\_korisnik\_recenzija\_recenzije\_proizvoda **FOREIGN KEY** (korisnik\_id) **REFERENCES** korisnici(id) );

Ova tablica se naziva „recenzije\_proizvoda“ i sadrži sljedeće atribute:

* „id“ je cjelobrojni atribut koji se automatski povećava svaki put kada se unese novi red u tablicu. Služi kao primarni ključ i osigurava jedinstvenu identifikaciju svake recenzije.
* „proizvod\_id“ je cjelobrojni atribut koji se odnosi na primarni ključ u tablici „proizvodi“. Označava proizvod na koji se odnosi recenzija.
* „korisnik\_id“ je cjelobrojni atribut koji se odnosi na primarni ključ u tablici „korisnici“. Označava korisnika koji je napisao recenziju.
* „ocjena“ je cjelobrojni atribut koji pohranjuje ocjenu proizvoda.
* „komentar“ je tekstualni atribut (TEXT) koji omogućuje korisnicima da napišu svoje mišljenje o proizvodu.
* „datum\_recenzije“ je vrijeme kada je recenzija napisana, s podrazumijevanom vrijednošću trenutnog vremena (CURRENT\_TIMESTAMP).

Tablica također ima sljedeća ograničenja:

* Atribut „id“ postavljen je kao primarni ključ.
* Vrijednost atributa „ocjena“ mora biti između 1 i 5, što je osigurano ograničenjem CHECK.
* Atribut „proizvod\_id“ povezan je s tablicom „proizvodi“, čime se osigurava referencijalna integracija za proizvode.
* Atribut „korisnik\_id“ povezan je s tablicom „korisnici“, čime se osigurava referencijalna integracija za korisnike.

## tablice Morena

## tablice Bruno

## tablice Fran

## tablice Josip

## tablice Leo

CREATE TABLE korisnici (

id INT AUTO\_INCREMENT PRIMARY KEY,

ime VARCHAR(255) NOT NULL,

prezime VARCHAR(255) NOT NULL,

email VARCHAR(255) UNIQUE NOT NULL,

lozinka VARCHAR(255) NOT NULL,

adresa TEXT,

grad VARCHAR(255),

telefon VARCHAR(20),

tip\_korisnika ENUM('kupac', 'admin') NOT NULL,

datum\_registracije DATE NOT NULL

);

Ova tablica se naziva **„korisnici“** i sadrži sljedeće atribute:

1. **„id“**
   * Cjelobrojni atribut koji se automatski povećava (AUTO\_INCREMENT) svaki put kada se unese novi red u tablicu.
   * Služi kao **primarni ključ** i osigurava jedinstvenu identifikaciju svakog korisnika.
2. **„ime“**
   * Tekstualni atribut (VARCHAR) s maksimalnom duljinom od 255 znakova.
   * Sadrži ime korisnika.
3. **„prezime“**
   * Tekstualni atribut (VARCHAR) s maksimalnom duljinom od 255 znakova.
   * Sadrži prezime korisnika.
4. **„email“**
   * Tekstualni atribut (VARCHAR) s maksimalnom duljinom od 255 znakova.
   * Mora biti jedinstven (**UNIQUE**) i služi za identifikaciju korisnika u sustavu.
5. **„lozinka“**
   * Tekstualni atribut (VARCHAR) s maksimalnom duljinom od 255 znakova.
   * Pohranjuje lozinku korisnika (obično hashiranu radi sigurnosti).
6. **„adresa“**
   * Tekstualni atribut (TEXT).
   * Sadrži fizičku adresu korisnika (ulica, broj i slično).
7. **„grad“**
   * Tekstualni atribut (VARCHAR) s maksimalnom duljinom od 255 znakova.
   * Sadrži naziv grada korisnika.
8. **„telefon“**
   * Tekstualni atribut (VARCHAR) s maksimalnom duljinom od 20 znakova.
   * Pohranjuje telefonski broj korisnika.
9. **„tip\_korisnika“**
   * Enumerirani atribut (**ENUM**) s mogućim vrijednostima „kupac“ i „admin“.
   * Označava vrstu korisnika (npr. obični kupac ili administrator sustava).
10. **„datum\_registracije“**
    * Atribut tipa **DATE**.
    * Pohranjuje datum kada je korisnik kreirao račun u sustavu.

### Ograničenja tablice:

* **Primarni ključ:**Atribut **„id“** postavljen je kao primarni ključ tablice, osiguravajući jedinstvenost svakog korisnika.
* **Jedinstveni email:**Atribut **„email“** mora biti jedinstven kako bi se izbjeglo dupliciranje korisnika s istim mailom.
* **Obavezni atributi:**Atributi **„ime“**, **„prezime“**, **„email“**, **„lozinka“**, **„tip\_korisnika“**, i **„datum\_registracije“** moraju biti popunjeni prilikom unosa (NOT NULL).

CREATE TABLE wishlist (

id INT AUTO\_INCREMENT PRIMARY KEY,

korisnik\_id INT NOT NULL,

proizvod\_id INT NOT NULL,

FOREIGN KEY (korisnik\_id) REFERENCES korisnici(id),

FOREIGN KEY (proizvod\_id) REFERENCES proizvodi(id)

);

Ova tablica se naziva **„wishlist“** i sadrži sljedeće atribute:

1. **„id“**
   * Cjelobrojni atribut koji se automatski povećava (**AUTO\_INCREMENT**) svaki put kada se unese novi red u tablicu.
   * Služi kao **primarni ključ** i osigurava jedinstvenu identifikaciju svakog unosa u „wishlist“.
2. **„korisnik\_id“**
   * Cjelobrojni atribut koji se odnosi na primarni ključ u tablici **„korisnici“**.
   * Predstavlja referencu na korisnika koji je dodao proizvod u svoju „wishlist“.
3. **„proizvod\_id“**
   * Cjelobrojni atribut koji se odnosi na primarni ključ u tablici **„proizvodi“**.
   * Predstavlja referencu na proizvod koji je korisnik dodao u svoju „wishlist“.

### Ograničenja tablice:

* **Primarni ključ:**Atribut **„id“** postavljen je kao primarni ključ, osiguravajući jedinstvenost svakog unosa u „wishlist“.
* **Vanjski ključ „korisnik\_id“:**Atribut **„korisnik\_id“** povezan je s atributom **„id“** iz tablice **„korisnici“**, čime se osigurava referencijalna integracija između korisnika i njihove „wishlist“.
* **Vanjski ključ „proizvod\_id“:**Atribut **„proizvod\_id“** povezan je s atributom **„id“** iz tablice **„proizvodi“**, čime se osigurava referencijalna integracija između proizvoda i „wishlist“.

### Opis funkcionalnosti:

Tablica **„wishlist“** omogućava povezivanje korisnika i proizvoda koje su dodali u svoju „wishlist“. Svaki unos u tablicu predstavlja proizvod koji je određeni korisnik odabrao za „wishlist“.  
Na primjer:

* Korisnik s **„korisnik\_id“** može imati više proizvoda u svojoj „wishlist“, dok svaki proizvod može biti dodan u „wishlist“ više korisnika.
* Relacije između korisnika i proizvoda omogućene su pomoću vanjskih ključeva.

### Napomena:

* Ograničenja stranih ključeva osiguravaju referencijalnu integraciju. Na primjer, proizvod ne može biti dodan u „wishlist“ ako ne postoji u tablici **„proizvodi“**, i „wishlist“ unos se ne može kreirati za korisnika koji ne postoji u tablici **„korisnici“**.

CREATE TABLE preporuceni\_proizvodi (

id INT AUTO\_INCREMENT PRIMARY KEY,

korisnik\_id INT NOT NULL,

proizvod\_id INT NOT NULL,

razlog\_preporuke TEXT,

FOREIGN KEY (korisnik\_id) REFERENCES korisnici(id),

FOREIGN KEY (proizvod\_id) REFERENCES proizvodi(id)

);

Ova tablica se naziva **„preporuceni\_proizvodi“** i sadrži sljedeće atribute:

1. **„id“**
   * Cjelobrojni atribut koji se automatski povećava (**AUTO\_INCREMENT**) svaki put kada se unese novi red u tablicu.
   * Služi kao **primarni ključ** i osigurava jedinstvenu identifikaciju svake preporuke proizvoda.
2. **„korisnik\_id“**
   * Cjelobrojni atribut koji se odnosi na primarni ključ u tablici **„korisnici“**.
   * Predstavlja korisnika koji je preporučio određeni proizvod.
3. **„proizvod\_id“**
   * Cjelobrojni atribut koji se odnosi na primarni ključ u tablici **„proizvodi“**.
   * Predstavlja proizvod koji je preporučen.
4. **„razlog\_preporuke“**
   * Tekstualni atribut (**TEXT**) koji pohranjuje opis ili objašnjenje zašto je korisnik preporučio određeni proizvod.

### Ograničenja tablice:

* **Primarni ključ:**Atribut **„id“** postavljen je kao primarni ključ, osiguravajući jedinstvenost svake preporuke.
* **Vanjski ključ „korisnik\_id“:**Atribut **„korisnik\_id“** povezan je s atributom **„id“** iz tablice **„korisnici“**, čime se osigurava referencijalna integracija između korisnika i njihovih preporuka.
* **Vanjski ključ „proizvod\_id“:**Atribut **„proizvod\_id“** povezan je s atributom **„id“** iz tablice **„proizvodi“**, čime se osigurava referencijalna integracija između preporuka i proizvoda.

### Opis funkcionalnosti:

Tablica **„preporuceni\_proizvodi“** omogućava evidentiranje preporuka proizvoda koje su korisnici predložili, uključujući dodatne razloge zašto je određeni proizvod preporučen.  
Na primjer:

* Jedan korisnik može preporučiti više proizvoda.
* Više korisnika može preporučiti isti proizvod, svaki s različitim razlozima.

### Primjeri upotrebe:

1. **Dodavanje preporuke proizvoda:**
   * Kada korisnik preporuči proizvod, unosi se novi red s njegovim korisničkim ID-jem, ID-jem proizvoda i opisom razloga preporuke.
2. **Prikaz preporuka:**
   * Pomoću relacija između tablica moguće je prikazati popis preporučenih proizvoda za korisnika ili prikazati proizvode koje su korisnici najčešće preporučivali.

### Napomena:

* Ograničenja stranih ključeva osiguravaju referencijalnu integraciju. Na primjer, preporuka ne može biti kreirana za korisnika ili proizvod koji ne postoji u bazi podataka.
* Polje **„razlog\_preporuke“** omogućava fleksibilno objašnjenje razloga za preporuku. Ako nije potrebno, može ostati prazno.

# Upiti i pogledi

## Upiti i pogledi Loren

**Pregled svih proizvoda s kategorijama**

**CREATE VIEW** svi\_proizvodi\_s\_kategorijama **AS**

**SELECT**

p.id **AS** proizvod\_id,

p.naziv **AS** proizvod,

p.cijena,

k.naziv **AS** kategorija,

p.kolicina\_na\_skladistu

**FROM** proizvodi p

**JOIN** kategorije\_proizvoda k **ON** p.kategorija\_id = k.id;

Pogled „svi\_proizvodi\_s\_kategorijama“ kombinira podatke iz tablica „proizvodi“ i „kategorije\_proizvoda“ na temelju zajedničkog atributa „kategorija\_id“. Ovaj pogled pruža način za pregled svih proizvoda zajedno s pripadajućim kategorijama. Stupovi koji se koriste za ovaj pogled su „proizvod\_id“ koji prikazuje jedinstveni identifikator proizvoda iz tablice „proizvodi“, „proizvod“ koji sadrži naziv proizvoda, „cijena“ sa cijenom proizvoda, „kategorija“ sa nazivom kategorije kojoj proizvod pripada i „kolicina\_na\_skladistu“ koja ima trenutno dostupnu količinu proizvoda na skladištu. Pogled omogućava korisnicima pregled informacija o proizvodima i njihovim kategorijama kao da je riječ o jednoj tablici.

**Pregled proizvoda koji su trenutno na skladištu**

**CREATE VIEW** dostupni\_proizvodi **AS**

**SELECT**

p.id **AS** proizvod\_id,

p.naziv **AS** proizvod,

p.kolicina\_na\_skladistu

**FROM** proizvodi p

**WHERE** p.kolicina\_na\_skladistu > 0;

Ovaj pogled pod imenom „dostupni\_proizvodi“ filtrira proizvode iz tablice „proizvodi“ koji imaju pozitivnu količinu na skladištu, odnosno proizvode koji su trenutno dostupni za prodaju. Atributi koji se koriste u ovom pogledu su „proizvod\_id“ koji sadrži jedinstveni identifikator proizvoda, „proizvod“ s nazivom proizvoda, „kolicina\_na\_skladistu“ s dostupnom količinom proizvoda na skladištu. Pogled se koristi za dohvat samo onih proizvoda koji su odmah dostupni za kupnju.

**Pregled svih recenzija s podacima o proizvodima i korisnicima**

**CREATE VIEW** recenzije\_s\_proizvodima\_i\_korisnicima **AS**

**SELECT**

r.id **AS** recenzija\_id,

p.naziv **AS** proizvod,

k.ime **AS** korisnik\_ime,

r.ocjena,

r.komentar,

r.datum\_recenzije

**FROM** recenzije\_proizvoda r

**JOIN** proizvodi p **ON** r.proizvod\_id = p.id

**JOIN** korisnici k **ON** r.korisnik\_id = k.id;

Pogled s nazivom „recenzije\_s\_proizvodima\_i\_korisnicima“ povezuje podatke iz tablica „recenzije\_proizvoda“, „proizvodi“ i „korisnici“ kako bi omogućio pregled recenzija zajedno s informacijama o proizvodima i korisnicima koji su ih napisali. Atributi koji se koriste pogledu su „recenzija\_id“ s jedinstvenim identifikatorom recenzije, „proizvod“ sa nazivom proizvoda koji je recenziran, „ime“ s imenom korisnika koji je napisao recenziju, „ocjena“ s ocjenom proizvoda (vrijednost od 1 do 5), „komentar“ koji sadrži komentar korisnika o proizvodu i „datum\_recenzije“ koji prikazuje datum kada je recenzija napisana. Pogled se koristi za dohvat podataka o recenzijama proizvoda s detaljnim informacijama o proizvodima i korisnicima.

**Pregled najpopularnijih proizvoda (najveći broj recenzija)**

**CREATE VIEW** najpopularniji\_proizvodi **AS**

**SELECT**

p.id **AS** proizvod\_id,

p.naziv **AS** proizvod,

**COUNT**(r.id) AS broj\_recenzija

**FROM** proizvodi p

**LEFT JOIN** recenzije\_proizvoda r **ON** p.id = r.proizvod\_id

**GROUP BY** p.id, p.naziv

**ORDER BY** broj\_recenzija **DESC**;

Pogled „najpopularniji\_proizvodi“ i prikazuje proizvode rangirane prema broju recenzija. Kombinira podatke iz tablica „proizvodi“ i „recenzije\_proizvoda“ koristeći LEFT JOIN kako bi uključio sve proizvode, čak i one bez recenzije. Stupcci koji se koriste u pogledu su „proizvod\_id“ sa jedinstvenim identifikatorom proizvoda, „proizvod“ sa nazivom proizvoda, „broj\_recenzija“ koji sadži broj recenzija za svaki proizvod. Proizvodi su sortirani prema broju recenzija u opadajućem redoslijedu, tako da budu najpopularnijih proizvodi na samom vrhu pogleda.

**Upit proizvoda koji su dostupni na skladištu, imaju više od 10 recenzija i prosječnu ocjenu iznad 4**

**SELECT**

p.id **AS** proizvod\_id,

p.naziv **AS** proizvod,

p.kolicina\_na\_skladistu,

**COUNT**(r.id) **AS** broj\_recenzija,

**AVG**(r.ocjena) **AS** prosjecna\_ocjena

**FROM** proizvodi p

**LEFT JOIN** recenzije\_proizvoda r **ON** p.id = r.proizvod\_id

**WHERE** p.kolicina\_na\_skladistu > 0

**GROUP BY** p.id, p.naziv, p.kolicina\_na\_skladistu

**HAVING** broj\_recenzija > 10 **AND** prosjecna\_ocjena > 4;

Ovaj upit dohvaća proizvode koji su dostupni na skladištu i koji imaju više od 10 recenzija s prosječnom ocjenom većom od 4. Podaci se kombiniraju iz tablica „proizvodi“ i „recenzije\_proizvoda“. Podaci se grupiraju prema svakom proizvodu, a rezultati su filtrirani s „HAVING“, gdje su uključeni samo proizvodi s više od 10 recenzija i prosječnom ocjenom većom od 4.

**Upit proizvoda s najvećom i najmanjom cijenom po kategoriji**

**SELECT**

k.naziv **AS** kategorija,

**MAX**(p.cijena) **AS** najskuplji\_proizvod,

**MIN**(p.cijena) **AS** najjeftiniji\_proizvod

**FROM** proizvodi p

**JOIN** kategorije\_proizvoda k **ON** p.kategorija\_id = k.id

**GROUP BY** k.naziv;

Ovaj upit pronalazi proizvode s najvećom i najmanjom cijenom unutar svake kategorije. Tablice „proizvodi“ i „kategorije\_proizvoda“ povezane su preko atributa „kategorija\_id“. Podaci se grupiraju po kategorijama, a funkcije agregacije „MAX“ i „MIN“ koriste se za izračunavanje maksimalnih i minimalnih cijena.

**Upit za korisnike koji su napisali najviše recenzija i njihove prosječne ocjene**

**SELECT**

k.id **AS** korisnik\_id,

k.ime,

k.prezime,

**COUNT**(r.id) **AS** broj\_recenzija,

**AVG**(r.ocjena) **AS** prosjecna\_ocjena

**FROM** korisnici k

**JOIN** recenzije\_proizvoda r **ON** k.id = r.korisnik\_id

**GROUP BY** k.id, k.ime, k.prezime

**ORDER BY** broj\_recenzija **DESC**

**LIMIT** 5;

Ovaj upit pronalazi korisnike koji su napisali najviše recenzija, zajedno s prosječnim ocjenama koje su dali. Podaci se kombiniraju iz tablica „korisnici“ i „recenzije\_proizvoda“. Podaci se grupiraju prema korisnicima, a rezultati su sortirani prema broju recenzija u silaznom redoslijedu. Prikazuju se podaci za najviše 5 korisnika pomoću ograničenja „LIMIT 5“.

## Upiti i pogledi Morena

## Upiti i pogledi Bruno

## Upiti i pogledi Fran

## Upiti i pogledi Josip

## Upiti i pogledi Leo

CREATE VIEW profil\_korisnika AS

SELECT

id AS korisnik\_id,

ime,

prezime,

email,

adresa,

grad,

telefon,

datum\_registracije

FROM korisnici;

### Pogled: Profil Korisnika

Pogled **„profil\_korisnika“** koristi se za dohvaćanje osnovnih informacija o korisnicima iz tablice **„korisnici“**. Služi za pojednostavljenje upita koji prikazuju korisničke profile, omogućavajući aplikaciji da lako prikaže informacije poput imena, prezimena, email adrese, adrese stanovanja, grada, telefona i datuma registracije.

* **Zašto se koristi ovaj pogled?**Umesto ponavljanja složenih upita u različitim delovima aplikacije, pogled pruža strukturiran i uvek ažuran prikaz korisničkih podataka. To čini aplikaciju efikasnijom i čitljivom.
* **Specifične prednosti:**Podaci se preuzimaju direktno u obliku koji je potreban aplikaciji, smanjujući potrebu za dodatnim manipulacijama na strani aplikacijskog koda.

-- Pogled: Preporuke drugih (Leo)

CREATE VIEW preporuke\_drugih AS

SELECT

k.id AS korisnik\_id,

k.ime,

k.prezime,

p.naziv AS proizvod\_naziv,

pp.razlog\_preporuke

FROM preporuceni\_proizvodi pp

JOIN korisnici k ON pp.korisnik\_id = k.id

JOIN proizvodi p ON pp.proizvod\_id = p.id;

### Pogled: Preporuke drugih

Pogled **„preporuke\_drugih“** koristi se za dohvaćanje informacija o proizvodima koje su preporučili drugi korisnici u aplikaciji. Pomoću ovog pogleda, aplikacija može jednostavno i efikasno prikazati proizvode koje su drugi korisnici preporučili, zajedno sa osnovnim informacijama o korisniku koji je dao preporuku, kao što su ime, prezime i razlog preporuke.

#### Zašto se koristi ovaj pogled?

Korištenje ovog pogleda omogućava jednostavno i brzo dohvaćanje svih potrebnih podataka o preporukama drugih korisnika bez potrebe za ponavljanjem složenih upita u različitim dijelovima aplikacije. Umjesto da svaki put radimo join između tablica korisnici, preporuceni\_proizvodi i proizvodi u aplikaciji, ovaj pogled centralizira upit i omogućava jednostavno preuzimanje relevantnih podataka u obliku koji je direktno potreban.

#### Specifične prednosti:

* **Pojednostavljenje upita**: Ovaj pogled omogućava aplikaciji da dobije sve potrebne informacije o preporukama drugih korisnika u samo jednom upitu, što poboljšava efikasnost i smanjuje dupliciranje koda.
* **Brzi pristup podacima**: Podaci o korisnicima i njihovim preporukama dostupni su odmah, što omogućava brže učitavanje stranica i bolji korisnički doživljaj.
* **Ažurnost podataka**: Pogled je uvijek ažuran jer se podaci iz baze povlače u trenutnom stanju, što znači da preporuke drugih korisnika neće biti zastarjele.

-- Pogled: Popularni proizvodi (najviše puta dodati u wishlist) (Leo)

CREATE OR REPLACE VIEW popularni\_proizvodi AS

SELECT

p.id AS proizvod\_id,

p.naziv,

COUNT(w.proizvod\_id) AS broj\_dodavanja

FROM proizvodi p

LEFT JOIN wishlist w ON p.id = w.proizvod\_id

GROUP BY p.id, p.naziv

ORDER BY broj\_dodavanja DESC;

### Pogled: Popularni Proizvodi

Pogled **„popularni\_proizvodi“** koristi se za identifikaciju proizvoda koji su najpopularniji među korisnicima prema broju dodavanja na **wishlist**. Kombinuje podatke iz tablica **„proizvodi“** i **„wishlist“** pomoću **LEFT JOIN**.

* **Zašto koristiti LEFT JOIN?  
  LEFT JOIN** osigurava da svi proizvodi iz tablice **„proizvodi“** budu uključeni u rezultat, čak i ako nijedan korisnik nije dodao taj proizvod na wishlist. Ovo je korisno za pružanje sveobuhvatne analize popularnosti proizvoda.
* **Šta omogućava ovaj pogled?**Funkcija **COUNT** broji koliko puta je svaki proizvod dodat na wishlist, dok **GROUP BY** grupiše rezultate po proizvodima. **ORDER BY** sortira proizvode po popularnosti u opadajućem redoslijedu, tako da se najpopularniji proizvodi nalaze na vrhu.
* **Prednosti:**Pogled omogućava brzi uvid u popularnost proizvoda i olakšava donošenje odluka, kao što su kreiranje marketinških kampanja ili ažuriranje ponude na osnovu popularnosti.

-- Pogled: Narudžbe Korisnika (Leo)

CREATE VIEW narudzbe\_korisnika AS

SELECT

n.id AS narudzba\_id,

n.datum\_narudzbe,

n.status\_narudzbe AS status,

s.proizvod\_id,

p.naziv AS proizvod\_naziv,

s.kolicina,

s.kolicina \* p.cijena AS ukupna\_cijena\_proizvoda,

n.korisnik\_id

FROM narudzbe n

JOIN stavke\_narudzbe s ON n.id = s.narudzba\_id

JOIN proizvodi p ON s.proizvod\_id = p.id;

### Pogled: Narudžbe Korisnika

Pogled **„narudzbe\_korisnika“** omogućava pregled narudžbi korisnika sa svim relevantnim detaljima, uključujući povezane proizvode i njihove troškove. Povezuje tablice **„narudzbe“**, **„stavke\_narudzbe“** i **„proizvodi“** pomoću **INNER JOIN**.

* **Zašto koristiti INNER JOIN?  
  INNER JOIN** osigurava da se u rezultate uključuju samo narudžbe koje imaju validne stavke i proizvode, čime se izbegava prikazivanje nepotpunih ili nevažećih podataka.
* **Kako funkcionira ovaj pogled?**
  + Informacije o narudžbama, kao što su datum, status i ukupni troškovi, dolaze iz tablice **„narudzbe“**.
  + Pojedinačne stavke narudžbi povezane su preko tablice **„stavke\_narudzbe“**, dok su podaci o proizvodima dohvaćeni iz tablice **„proizvodi“**.
  + Ukupna cena svake stavke računa se dinamički množenjem količine proizvoda sa njegovom cijenom.
* **Prednosti:**Pogled pruža sveobuhvatne informacije o narudžbama korisnika, što omogućava transparentan pregled narudžbi unutar korisničkog profila.

-- Upit: Provjera popularnosti proizvoda (koji proizvodi su najčešće dodati u wishlist) (Leo)

SELECT

p.id AS proizvod\_id,

p.naziv AS proizvod\_naziv,

COUNT(w.proizvod\_id) AS broj\_dodavanja

FROM proizvodi p

LEFT JOIN wishlist w ON p.id = w.proizvod\_id

GROUP BY p.id, p.naziv

ORDER BY broj\_dodavanja DESC

LIMIT 10; -- Dohvata prvih 10 proizvoda

**Upit: Provera popularnosti proizvoda (koji proizvodi su najčešće dodati u wishlist)**

**Opis:**Ovaj upit koristi se za identifikaciju proizvoda koji su najpopularniji među korisnicima na osnovu broja dodavanja u **wishlist**. Upit kombinuje podatke iz tablica **„proizvodi“** i **„wishlist“** pomoću **LEFT JOIN** kako bi uključio sve proizvode, čak i one koji nisu dodati na wishlist.

* **Detalji implementacije:**
  + **LEFT JOIN** omogućava da svi proizvodi budu uključeni u rezultat, čak i oni bez unosa u **wishlist**.
  + Funkcija **COUNT** računa broj puta koliko je svaki proizvod dodat u **wishlist**.
  + **GROUP BY** osigurava grupiranje rezultata prema jedinstvenim proizvodima.
  + **ORDER BY** sortira proizvode prema broju dodavanja u padajućem redoslijedu.
  + **LIMIT 10** ograničava rezultat na prvih 10 najpopularnijih proizvoda.

-- Upit: Provjera korisnika sa najviše narudžbi (Leo)

SELECT

k.id AS korisnik\_id,

k.ime,

k.prezime,

COUNT(n.id) AS broj\_narudzbi

FROM korisnici k

JOIN narudzbe n ON k.id = n.korisnik\_id

GROUP BY k.id, k.ime, k.prezime

ORDER BY broj\_narudzbi DESC

LIMIT 5; -- Prikazuje prvih 5 korisnika sa najviše narudžbi

**Upit: Provjera korisnika sa najviše narudžbi**

**Opis:**Ovaj upit omogućava identifikaciju korisnika koji su kreirali najviše narudžbi. Kombinuje podatke iz tablica **„korisnici“** i **„narudzbe“** pomoću **INNER JOIN** kako bi obezbedio da se uključe samo korisnici sa valjanim narudžbama.

* **Detalji implementacije:**
  + **INNER JOIN** osigurava povezivanje samo onih korisnika koji imaju najmanje jednu narudžbu.
  + Funkcija **COUNT** broji broj narudžbi za svakog korisnika.
  + **GROUP BY** grupiše rezultate prema jedinstvenim korisnicima.
  + **ORDER BY** sortira rezultate prema broju narudžbi u padajućem redoslijedu.
  + **LIMIT 5** ograničava prikaz na prvih 5 korisnika sa najviše narudžbi.

-- Upit: Ukupna zarada po korisnicima na osnovu narudžbi(Leo)

SELECT

k.id AS korisnik\_id,

k.ime,

k.prezime,

SUM(n.ukupna\_cijena) AS ukupna\_zarada

FROM korisnici k

JOIN narudzbe n ON k.id = n.korisnik\_id

GROUP BY k.id, k.ime, k.prezime

ORDER BY ukupna\_zarada DESC;

**Upit: Ukupna zarada po korisnicima na osnovu narudžbi**

**Opis:**Ovaj upit koristi se za izračunavanje ukupne zarade koju je svaki korisnik generisao na osnovu svojih narudžbi. Kombinuje podatke iz tablica **„korisnici“** i **„narudzbe“** pomoću **INNER JOIN**.

* **Detalji implementacije:**
  + **INNER JOIN** osigurava da se uzimaju u obzir samo korisnici koji imaju validne narudžbe.
  + Funkcija **SUM** računa ukupnu vrijednost svih narudžbi za svakog korisnika.
  + **GROUP BY** grupira rezultate prema jedinstvenim korisnicima.
  + **ORDER BY** sortira rezultate prema ukupnoj zaradi u padajućem redoslijedu

# Funkcije

## Funkcije Loren

**Funkcija koja vraća ukupnu vrijednost svih proizvoda na skladištu**

DELIMITER //

**CREATE FUNCTION** UkupnaVrijednostSkladista ()

**RETURNS DECIMAL**(15,2)

**DETERMINISTIC**

**BEGIN**

**DECLARE** ukupna\_vrijednost **DECIMAL**(15,2);

**SELECT** **SUM**(kolicina\_na\_skladistu \* cijena)

**INTO** ukupna\_vrijednost

**FROM** proizvodi;

**RETURN** ukupna\_vrijednost;

**END** //

DELIMITER ;

Funkcija „UkupnaVrijednostSkladista“ izračunava ukupnu vrijednost svih proizvoda na skladištu tako da množi cijenu svakog proizvoda s količinom na skladištu i zbroji sve rezultate. Prvo funkcija deklarira varijablu „ukupna\_vrijednost“ za spremanje ukupne vrijednosti. Zatim izračunava zbroj vrijednosti svih proizvoda koristeći „SUM“. I na kraju vraća rezultat tipa „DECIMAL“ s dvije decimale.

## Funkcije Morena

## Funkcije Bruno

## Funkcije Fran

## Funkcije Josip

## Funkcije Leo

-- Funkcija: Formatiranje datuma (Leo)

DELIMITER $$

CREATE FUNCTION formatiraj\_datum(input\_date DATE)

RETURNS VARCHAR(10)

DETERMINISTIC

BEGIN

RETURN DATE\_FORMAT(input\_date, '%d.%m.%Y');

END$$

DELIMITER ;

Funkcija: Formatiranje datuma

Opis:  
Ova funkcija omogućava formatiranje datuma u željenom formatu „dan.mjesec.godina“. Namenjena je za upotrebu u raznim SQL upitima ili pogledima kako bi datumi bili prikazani u čitljivom formatu za korisnike.

* Detalji implementacije:
  + Funkcija koristi DATE\_FORMAT, ugrađenu MySQL funkciju, za prilagođavanje formata prikaza datuma.
  + Ulazni parametar input\_date očekuje vrijednost tipa DATE.
  + Vraća vrijednost tipa VARCHAR(10) sa formatom „dd.mm.yyyy“, gde:
    - dd predstavlja dan u mesecu,
    - mm predstavlja mesec u godini,
    - yyyy predstavlja četvorocifrenu godinu.
  + Funkcija je deklarisana kao DETERMINISTIC, što znači da će uvek vratiti isti izlaz za isti ulaz.

# Procedure

## Procedure Loren

**Procedura za dodavanje proizvoda s provjerom grešaka**

DELIMITER //

**CREATE PROCEDURE** DodajProizvod (

**IN** p\_naziv **VARCHAR**(255),

**IN** p\_opis **TEXT**,

**IN** p\_cijena **DECIMAL**(10, 2),

**IN** p\_kategorija\_id **INT**,

**IN** p\_kolicina **INT**,

**IN** p\_slika **VARCHAR**(255),

**IN** p\_specifikacije **TEXT**

)

**BEGIN**

**IF** p\_cijena <= 0 **THEN**

**SIGNAL SQLSTATE '45501'**

**SET MESSAGE\_TEXT** = 'Cijena mora biti veća od 0.';

**ELSEIF** p\_kolicina < 0 **THEN**

**SIGNAL SQLSTATE '45502'**

**SET MESSAGE\_TEXT** = 'Količina na skladištu ne može biti negativna.';

**ELSE**

**INSERT INTO** proizvodi (naziv, opis, cijena, kategorija\_id, kolicina\_na\_skladistu, slika, specifikacije)

**VALUES** (p\_naziv, p\_opis, p\_cijena, p\_kategorija\_id, p\_kolicina, p\_slika, p\_specifikacije);

**END IF**;

**END** //

DELIMITER ;

Procedura „DodajProizvod“ omogućava dodavanje proizvoda u tablicu „proizvodi“ uz provjeru valjanosti unosa. Prvo provjerava valjanost unosa tj. cijena mora biti veća od 0 i količina ne smije biti negativna. Ako je unos ispravan, proizvod se unosi u tablicu „proizvodi“. U slučaju neispravnog unosa, generira se greška pomoću „SIGNAL“.

**Procedura za brisanje proizvoda s provjerom povezanih podataka**

DELIMITER //

**CREATE PROCEDURE** ObrisiProizvod (

**IN** p\_proizvod\_id **INT**

)

**BEGIN**

**IF EXISTS** (

**SELECT** 1

**FROM** stavke\_narudzbe

**WHERE** proizvod\_id = p\_proizvod\_id

) **THEN**

**SIGNAL SQLSTATE '45503'**

**SET MESSAGE\_TEXT** = 'Proizvod ne može biti obrisan jer postoje povezane narudžbe.';

**ELSE**

**DELETE FROM** popusti

**WHERE** proizvod\_id = p\_proizvod\_id;

**DELETE FROM** recenzije\_proizvoda

**WHERE** proizvod\_id = p\_proizvod\_id;

**DELETE FROM** wishlist

**WHERE** proizvod\_id = p\_proizvod\_id;

**DELETE FROM** proizvodi

**WHERE** id = p\_proizvod\_id;

**END IF**;

**END** //

DELIMITER ;

Procedura „ObrisiProizvod“ briše proizvod i sve povezane zapise iz tablica „popusti“, „recenzije\_proizvoda“, i „wishlist“. Međutim, ne dopušta brisanje ako postoje zapisi u tablici „stavke\_narudzbe“. Najprije provjerava postoji li zapis u tablici „stavke\_narudzbe“ za neki proizvod i ako postoji, generira grešku. Ako nema povezanih podaka, briše sve povezane n-torke i sam proizvod iz tablice „proizvodi“.

**Procedura za povećanje cijene za 5% na sve proizvode iz kategorije "Matične ploče"**

DELIMITER //

**CREATE PROCEDURE** PovecajCijenu (

**IN** kategorija\_naziv **VARCHAR**(255)

)

**BEGIN**

**UPDATE** proizvodi p

**INNER JOIN** kategorije\_proizvoda k **ON** p.kategorija\_id = k.id

**SET** p.cijena = p.cijena \* 1.05

**WHERE** k.naziv = kategorija\_naziv;

**END** //

DELIMITER ;

Procedura „PovecajCijenu“ povećava cijenu svih proizvoda u određenoj kategoriji za 5%. Povezuje tablicu „proizvodi“ s tablicom „kategorije\_proizvoda“. Filtrira proizvode prema nazivu kategorije. I na kraju ažurira cijene tako da ih množi s 1,05 da bi dobili povećanje od 5%.

**Procedura za pretragu proizvode s ključnom riječi**

DELIMITER //

**CREATE PROCEDURE** PronadjiProizvode (

**IN** kljucna\_rijec **VARCHAR**(255)

)

**BEGIN**

**SELECT** \*

**FROM** proizvodi

**WHERE** naziv **LIKE** **CONCAT**('%', kljucna\_rijec, '%')

**OR** opis **LIKE CONCAT**('%', kljucna\_rijec, '%');

**END** //

DELIMITER ;

Procedura „PronadjiProizvode“ omogućava pretraživanje proizvoda na temelju ključne riječi u nazivima i opisima proizvoda. Koristi se „LIKE“ operator za pretraživanje proizvoda prema ključnim riječima u atributima „naziv“ i „opis“. Na kraju vraća rezultat kao tablicu sa svim n-torkama koje se podudaraju s ključnom riječi.

## Procedure Morena

## Procedure Bruno

## Procedure Fran

## Procedure Josip

## Procedure Leo

-- Procedura: Dodavanje korisnika (Leo)

DELIMITER //

CREATE PROCEDURE dodaj\_korisnika(

IN p\_ime VARCHAR(255),

IN p\_prezime VARCHAR(255),

IN p\_email VARCHAR(255),

IN p\_lozinka VARCHAR(255),

IN p\_adresa TEXT,

IN p\_grad VARCHAR(255),

IN p\_telefon VARCHAR(20)

)

BEGIN

IF EXISTS (SELECT 1 FROM korisnici WHERE email = p\_email) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Korisnik sa ovim email-om već postoji!';

ELSE

INSERT INTO korisnici (ime, prezime, email, lozinka, adresa, grad, telefon, tip\_korisnika, datum\_registracije)

VALUES (p\_ime, p\_prezime, p\_email, p\_lozinka, p\_adresa, p\_grad, p\_telefon, 'kupac', CURDATE());

END IF;

END//

DELIMITER ;

Procedura dodaj\_korisnika omogućava dodavanje novog korisnika u bazu podataka. Pre unosa podataka, proverava da li već postoji korisnik sa istim mailom kako bi se osigurala jedinstvenost. Ako korisnik sa unetim email-om već postoji, procedura generira SQL grešku pomoću SIGNAL SQLSTATE. U slučaju da mail ne postoji, novi korisnik se unosi u tabelu korisnici zajedno sa svim podacima: ime, prezime, email, lozinka, adresa, grad, telefon, tip korisnika i datum registracije. Ova procedura osigurava referencijalni integritet i sprečava dupliciranje podataka u bazi.

CREATE PROCEDURE prikazi\_preporuke(korisnik\_id INT, page INT, per\_page INT)

BEGIN

DECLARE offset\_value INT;

-- Izračunavanje offset-a

SET offset\_value = (page - 1) \* per\_page;

SELECT

p.id AS proizvod\_id,

p.naziv,

p.opis,

p.cijena

FROM proizvodi p

WHERE p.kategorija\_id IN (

-- Preporučujemo proizvode iz kategorija koje je korisnik dodao u wishlist ili kupio

SELECT DISTINCT p2.kategorija\_id

FROM proizvodi p2

JOIN wishlist w ON p2.id = w.proizvod\_id

WHERE w.korisnik\_id = korisnik\_id

UNION

SELECT DISTINCT p3.kategorija\_id

FROM proizvodi p3

JOIN stavke\_narudzbe s ON p3.id = s.proizvod\_id

JOIN narudzbe n ON s.narudzba\_id = n.id

WHERE n.korisnik\_id = korisnik\_id

)

AND p.id NOT IN (

-- Iznimamo proizvode koje je korisnik već dodao u wishlist

SELECT proizvod\_id

FROM wishlist

WHERE korisnik\_id = korisnik\_id

)

ORDER BY p.cijena DESC

LIMIT per\_page OFFSET offset\_value; -- Koristimo izračunati offset

END//

DELIMITER ;

Procedura „prikazi\_preporuke“ koristi podatke o korisničkim aktivnostima, kao što su proizvodi koje je korisnik dodao u wishlist ili kupio, kako bi generirala personalizirane preporuke proizvoda. Na temelju tih podataka, aplikacija preporučuje proizvode iz kategorija koje su korisniku već bile zanimljive, čime se povećava relevantnost preporuka. Ujedno, izuzimaju se proizvodi koje je korisnik već dodao u wishlistu, čime se izbjegava ponavljanje. Ova procedura također implementira paginaciju, omogućujući korisnicima da pregledavaju proizvode stranica po stranica, čime se smanjuje opterećenje aplikacije i poboljšava korisničko iskustvo.

Kod je osmišljen tako da omogućava brzo i učinkovito dohvaćanje podataka o proizvodima koji pripadaju kategorijama koje je korisnik već pratio, bez potrebe za ponavljanjem složenih SQL upita u aplikaciji. Upit koristi UNION za kombiniranje kategorija proizvoda iz wishlist i narudžbi, dok ORDER BY sortira proizvode po cijeni. Paginacija se postiže korištenjem LIMIT i OFFSET, što omogućava prikazivanje samo određenog broja proizvoda po stranici. Na taj način, aplikacija ostaje efikasna i optimizirana, dok korisnici uživaju u personaliziranim i lako dostupnim preporukama.

-- Procedura: Brisanje korisnika (Leo)

DELIMITER //

CREATE PROCEDURE obrisi\_korisnika(

IN p\_korisnik\_id INT

)

BEGIN

DECLARE korisnik\_ima\_aktivne\_narudzbe BOOLEAN;

-- Provera da li korisnik ima aktivne narudžbe

SELECT EXISTS (SELECT 1 FROM narudzbe WHERE korisnik\_id = p\_korisnik\_id)

INTO korisnik\_ima\_aktivne\_narudzbe;

IF korisnik\_ima\_aktivne\_narudzbe THEN

SIGNAL SQLSTATE '45001'

SET MESSAGE\_TEXT = 'Korisnik ima aktivne narudžbe i ne može se obrisati!';

ELSE

-- Brisanje povezanih podataka

DELETE FROM placanja WHERE narudzba\_id IN (SELECT id FROM narudzbe WHERE korisnik\_id = p\_korisnik\_id);

DELETE FROM stavke\_narudzbe WHERE narudzba\_id IN (SELECT id FROM narudzbe WHERE korisnik\_id = p\_korisnik\_id);

DELETE FROM narudzbe WHERE korisnik\_id = p\_korisnik\_id;

DELETE FROM wishlist WHERE korisnik\_id = p\_korisnik\_id;

-- Na kraju, brisanje korisnika

DELETE FROM korisnici WHERE id = p\_korisnik\_id;

END IF;

END//

DELIMITER ;

Procedura obrisi\_korisnika omogućava sigurno brisanje korisnika iz baze podataka. Pre nego što se korisnik obriše, procedura provjerava da li ima aktivne narudžbe u tabeli narudzbe. Ako postoje aktivne narudžbe, procedura generira SQL grešku pomoću SIGNAL SQLSTATE i prekida izvršavanje. Ako korisnik nema aktivnih narudžbi, procedura briše sve povezane podatke iz tablica placanja, stavke\_narudzbe, narudzbe i wishlist, pre nego što izbriše samog korisnika iz tabele korisnici. Ova procedura osigurava da se korisnik može obrisati samo ako nema neriješenih obaveza u bazi.

-- Procedura: Ažuriranje korsinika(Leo)

DELIMITER //

CREATE PROCEDURE azuriraj\_korisnika(

IN p\_korisnik\_id INT,

IN p\_ime VARCHAR(255),

IN p\_prezime VARCHAR(255),

IN p\_email VARCHAR(255),

IN p\_adresa TEXT,

IN p\_grad VARCHAR(255),

IN p\_telefon VARCHAR(20)

)

BEGIN

UPDATE korisnici

SET

ime = p\_ime,

prezime = p\_prezime,

email = p\_email,

adresa = p\_adresa,

grad = p\_grad,

telefon = p\_telefon

WHERE id = p\_korisnik\_id;

END//

DELIMITER ;

Procedura azuriraj\_korisnika omogućava ažuriranje podataka korisnika u tabeli korisnici. Procedura prihvata podatke kao što su ime, prezime, email, adresa, grad i telefon, zajedno sa jedinstvenim ID-jem korisnika. Koristeći SQL naredbu UPDATE, ažurira sve navedene podatke u tabeli korisnici na osnovu unijetog ID-ja. Ova procedura omogućava jednostavno i sigurno ažuriranje podataka korisnika u bazi, uz očuvanje postojećih veza i integriteta.

# Okidači

## Okidači Loren

**Okidač za obavijest o niskoj zalihi**

**CREATE TEMPORARY TABLE** privremene\_obavijesti (

poruka **TEXT**,

vrijeme\_kreiranja **DATETIME**

);

DELIMITER //

**CREATE TRIGGER** au\_ObavijestNiskaZaliha

**AFTER UPDATE ON** proizvodi

**FOR EACH ROW**

**BEGIN**

**IF NEW**.kolicina\_na\_skladistu < 5 **THEN**

**INSERT INTO** privremene\_obavijesti (poruka, vrijeme\_kreiranja)

**VALUES** (

**CONCAT**('Upozorenje: Zaliha za proizvod "', **NEW**.naziv, '" je pala ispod 5.'),

**NOW**()

);

**END IF**;

**END** //

DELIMITER ;

Okidač „au\_ObavijestNiskaZaliha“ automatski generira obavijest kada zaliha proizvoda padne ispod 5. Obavijest se sprema u privremenu tablicu „privremene\_obavijesti“ koju kreiramo prije samog okidača. Okidač provjerava je li nova količina manja od 5. Ako je, u privremenu tablicu „privremene\_obavijesti“ unosi poruku „Zaliha za proizvod 'proizvod' je pala ispod 5.“ i trenutni datum i vrijeme.

**Okidač koji ne dopušta recenziju ako korisnik nije kupio proizvod ili ako je već napisao recenziju**

DELIMITER //

**CREATE TRIGGER** bi\_RestrikcijaRecenzije

**BEFORE INSERT ON** recenzije\_proizvoda

**FOR EACH ROW**

**BEGIN**

**IF NOT EXISTS** (

**SELECT** 1

**FROM** stavke\_narudzbe sn

**JOIN** narudzbe n **ON** sn.narudzba\_id = n.id

**WHERE** sn.proizvod\_id = **NEW**.proizvod\_id **AND** n.korisnik\_id = **NEW**.korisnik\_id

) **THEN**

**SIGNAL SQLSTATE '45505'**

**SET MESSAGE\_TEXT** = 'Korisnik nije kupio ovaj proizvod i ne može ostaviti recenziju.';

**END IF**;

**IF EXISTS** (

**SELECT** 1

**FROM** recenzije\_proizvoda

**WHERE** proizvod\_id = **NEW**.proizvod\_id **AND** korisnik\_id = **NEW**.korisnik\_id

) **THEN**

**SIGNAL SQLSTATE '45506'**

**SET MESSAGE\_TEXT** = 'Korisnik je već ostavio recenziju za ovaj proizvod.';

**END IF**;

**END** //

DELIMITER ;

Okidač „bi\_RestrikcijaRecenzije“ sprječava dodavanje recenzije za proizvod ako korisnik nije kupio taj proizvod. Prije dodavanja novog podatka u tablicu „recenzije\_proizvoda“, provjerava se postoji li odgovarajuća narudžba u tablici „stavke\_narudzbe“, gdje se podudaraju „proizvod\_id“ iz narudžbe i „korisnik\_id“ iz tablice „recenzije\_proizvoda“. Ako podudaranje ne postoji, generira se greška "Korisnik nije kupio ovaj proizvod i ne može ostaviti recenziju". Dodatno se provjerava da li već postoji recenzija za isti „proizvod\_id“ i „korisnik\_id“ u tablici „recenzije\_proizvoda“. Ako postoji, generira se greška "Korisnik je već ostavio recenziju za ovaj proizvod".

## Okidači Morena

## Okidači Bruno

## Okidači Fran

## Okidači Josip

## Okidači Leo

-- Okidač: Automatsko postavljanje datuma registracije korisnika (Leo)

DELIMITER //

CREATE TRIGGER postavi\_datum\_registracije

BEFORE INSERT ON korisnici

FOR EACH ROW

BEGIN

IF NEW.datum\_registracije IS NULL THEN

SET NEW.datum\_registracije = CURDATE();

END IF;

END//

DELIMITER ;

Ovaj okidač se izvršava prije svakog unosa (BEFORE INSERT) u tablicu korisnici. Njegova svrha je da automatski postavi vrijednost atributa datum\_registracije na trenutni datum (CURDATE()), ukoliko nije eksplicitno definirano prilikom unosa. Ovim se osigurava da svaki korisnik koji se unosi u bazu ima ispravno postavljen datum registracije. Okidač doprinosi dosljednosti i točnosti podataka u tablici korisnici, eliminisanjem mogućnosti unosa zapisa sa nedostajućim ili neispravnim datumom.

-- Okidač: spreči duplikate wishlist (Leo)

DELIMITER //

CREATE TRIGGER spreči\_duplikate\_wishlist

BEFORE INSERT ON wishlist

FOR EACH ROW

BEGIN

IF EXISTS (

SELECT 1

FROM wishlist

WHERE korisnik\_id = NEW.korisnik\_id AND proizvod\_id = NEW.proizvod\_id

) THEN

SIGNAL SQLSTATE '45002'

SET MESSAGE\_TEXT = 'Proizvod je već u wishlist-u ovog korisnika!';

END IF;

END//

DELIMITER ;

Ovaj okidač se izvršava pre svakog unosa (BEFORE INSERT) u tablicu wishlist. Njegova svrha je da spreči dodavanje duplikata u tabelu wishlist, tj. da isti proizvod ne može biti dodan u wishlist istog korisnika više puta. Provjerava postojanje odgovarajućeg zapisa koristeći EXISTS sa odgovarajućim kriterijima (korisnik\_id i proizvod\_id). Ukoliko zapis već postoji, okidač generira SQL grešku sa porukom „Proizvod je već u wishlist-u ovog korisnika!“ koristeći SIGNAL SQLSTATE. Ovaj okidač je ključan za održavanje integriteta podataka i sprečavanje redundantnih zapisa u tabeli wishlist.

# Transakcije