Sveučilište Jurja Dobrile u Puli

DOKUMENTACIJA PROJEKTNOG ZADATKA

Naziv projekta: *Vođenje web trgovine*
Tim: *14*
Smjer: *Informatika*
Kolegij: *Baze podataka 1*
Mentor: *doc. Dr. Sc. Goran Oreški*
Lokacija i datum: *Pula, svibanj 2025.*

SADRŽAJ

---

1. UVOD

Projekt *Vođenje web trgovine* osmišljen je s ciljem izrade baze podataka koja omogućuje učinkovito upravljanje online trgovinom. Baza podataka pruža podršku za ključne funkcionalnosti poput upravljanja proizvodima, narudžbama, korisnicima, skladištem, dostavom i promocijama.

Svrha baze podataka je osigurati pouzdano i brzo upravljanje podacima, omogućiti jednostavno praćenje poslovnih procesa te podržati analizu i optimizaciju poslovanja.

Razlog odabira ove teme je rastuća popularnost online trgovina i potreba za kvalitetnim sustavima koji omogućuju njihovo vođenje. Projekt pruža priliku za praktičnu primjenu znanja o dizajnu baza podataka, relacijskim modelima i SQL upitima, uz fokus na stvarne poslovne zahtjeve.

---

2. OPIS POSLOVNOG PROCESA

Poslovni proces kojeg smo ovdje modelirali, iako pojednostavljen i limitiran, vjerujemo da omogućuje učinkovito upravljanje ključnim entitetima i njihovim međusobnim odnosima.

Na web trgovini proizvodi su ključni entitet oko kojeg je modelirana baza podataka. Svaki proizvod ima osnovne informacije poput naziva, opisa i cijene. Kako bi proizvodi bili prikazani kupcima, dodali smo mogućnost povezivanja proizvoda sa slikama. Svaki proizvod može imati više slika, a također postoji mogućnost da više proizvoda dijeli istu sliku. Zbog toga smo relaciju između proizvoda i slika modelirali kao vezu "više na više". Svaka slika sadrži podatke o putanji na serveru, nazivu i opisu.

Prije kupnje, kupci trebaju kreirati korisnički račun koji spremamo pod entitet "korisnik", a koji sadrži podatke o kupcu, kao što su ime, prezime, e-mail, lozinka za autentifikaciju u naš sustav, adresa, grad, država i telefon - kako bismo mogli dostaviti narudžbu.

Kupci na web-trgovini očekuju brzo i intuitivno pretraživanje proizvoda, stoga smo slične proizvode grupirali u kategorije. Svaki proizvod može pripadati više kategorijama - primjerice, disk pločice mogu biti dio kategorija "Kočioni sustav", "Potrošni materijal" i "Servisni dijelovi". S druge strane, svaka kategorija može sadržavati više proizvoda, pa je ova relacija modelirana kao veza "više na više".

Također, kada kupac pronađe proizvod koji ga zanima, može ga spremiti na listu želja kako bi mu u budućnosti bio lako dostupan. Svaki kupac ima mogućnost kreiranja više lista želja - primjerice, ako ga zanimaju dijelovi za servis vlastitog automobila, može kreirati listu isključivo za tu svrhu i u nju spremati artikle koje želi pratiti. Svaka lista želja može sadržavati više artikala.

Kada se kupac odluči za kupnju, kreira se narudžba. Za svaku narudžbu spremamo datum, status, ukupni iznos koji je potrebno platiti i način plaćanja koji je odabran. Svaki kupac može kreirati više narudžbi, a svaka narudžba pripada samo jednom korisniku.

Osim samog kreiranja narudžbe, potrebno je evidentirati koje je proizvode kupac kupio. Svaka narudžba može sadržavati jedan ili više proizvoda, a važno nam je zabilježiti količinu kupljenih proizvoda te njihovu cijenu u trenutku kupnje. Jedna narudžba može sadržavati više proizvoda, a jedan proizvod može biti dio više narudžbi - što čini još jednu vezu "više na više".

Kod kreiranja narudžbe kupac ima mogućnost unosa kupona kojima ostvaruje popuste. Popust može biti fiksni (npr. 10 eura na ukupnu cijenu), postotni (npr. 5 % popusta), ili u obliku besplatne dostave. Kuponi se koriste i za promociju web-trgovine. Jednom kada se kupon kreira, možemo ga vremenski ograničiti, a u slučaju problema možemo ga deaktivirati promjenom njegovog statusa (aktivan/neaktivan). Za svaki kupon pratimo naziv, tip kupona (fiksni, postotni ili besplatna dostava), vrijednost popusta te datume aktivacije i deaktivacije. Jedna narudžba može imati više kupona, a jedan kupon se može primijeniti na više narudžbi - što opet čini vezu "više na više".

Nakon kreiranja narudžbe i pridruživanja proizvoda, potrebno je pratiti je li narudžba plaćena. Uplata može biti uspješna (ako je kupac izvršio plaćanje) ili neuspješna (npr. banka odbije transakciju kod plaćanja karticom). Također, zbog mogućnosti obročnog plaćanja, jedna narudžba može imati više uplata. Za svaku uplatu spremamo iznos, datum i status (uspješna ili neuspješna).

Nakon potvrde uplate, zaposlenici šalju narudžbu na željenu adresu. Svaka dostava ima svoj status (naručeno, u transportu, isporučeno), cijenu dostave, opis (za dodatne informacije) te datume kreiranja, slanja i isporuke. Svaka dostava je povezana s narudžbom, a dostava ne može postojati bez prethodno kreirane narudžbe. Svaka narudžba ima točno jednu dostavu jer smo odlučili da jednu narudžbu nećemo slati u više navrata.

Dostavu obavlja kurirska služba. Budući da surađujemo s više kurirskih službi, spremamo njihove podatke u bazu kako bismo pratili isporuke. Za svaku kurirsku službu bilježimo naziv, opis i kontakt. Jednu dostavu

obavlja jedna kurirska služba, ali jedna kurirska služba može obaviti više dostava. Dostava je moguća isključivo putem kurirske službe - kupac ne može osobno preuzeti narudžbu.

Nakon kupnje, korisnik može ostaviti recenziju proizvoda - kako bi podijelio svoje iskustvo i pomogao drugim kupcima. Svaki korisnik može ocijeniti proizvode putem recenzije. Za svaku recenziju spremamo ocjenu (od 1 do 5), komentar i datum kada je recenzija napisana. Jedan korisnik može napisati više recenzija, ali svaka recenzija pripada samo jednom korisniku. Recenzija ne može postojati bez korisnika i bez konkretnog proizvoda koji se ocjenjuje.

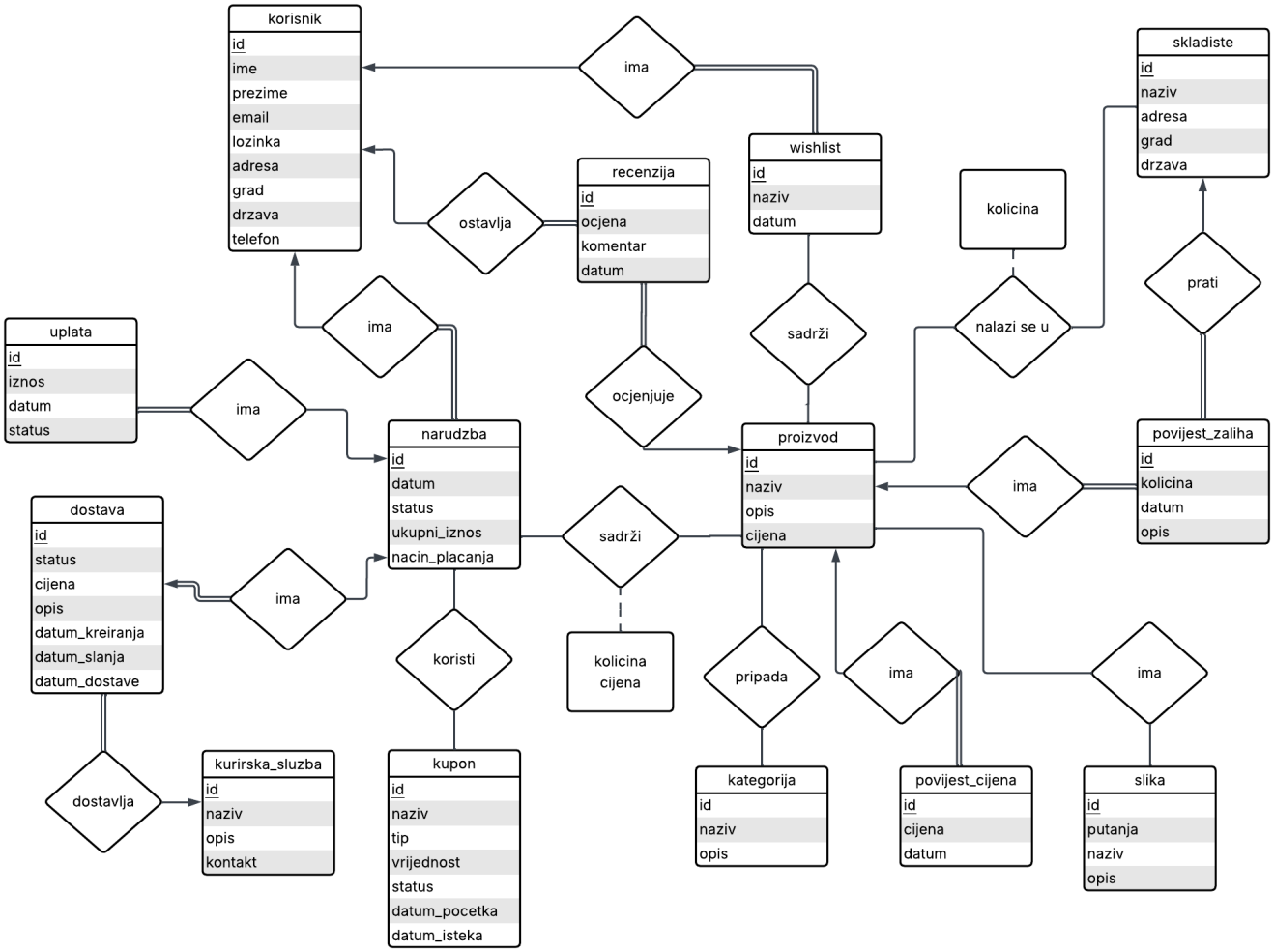
Zaposlenici trgovine često trebaju analizirati promjene cijena proizvoda, zbog čega vodimo evidenciju povijesti cijena. Svaka promjena cijene bilježi se s novom vrijednošću i datumom izmjene, čime se osigurava uvid u razvoj cijena kroz vrijeme. Svaka promjena mora biti vezana uz konkretan proizvod - jedan proizvod može imati više zapisa o cijenama, dok svaki zapis pripada samo jednom proizvodu.

Kako bismo pratili stanje zaliha i jer web-trgovine često imaju više skladišta, kreirali smo entitet "skladište" koji predstavlja fizičko skladište. Svako skladište može sadržavati više proizvoda, pa za svaki proizvod pratimo trenutnu količinu. Jedan proizvod može biti pohranjen u više skladišta - veza je također "više na više".

Radi papirologije i boljeg praćenja, uvedena je povijest zaliha. Tu spremamo sve promjene stanja, primjerice prilikom prodaje - kada se proizvod povuče iz skladišta - bilježi se promjena količine i razlog promjene. Svaki zapis u povijesti zaliha mora biti vezan uz skladište i proizvod. Jedan proizvod može imati više zapisa u povijesti zaliha, kao i jedno skladište.

---

3. ER DIJAGRAM



---

1. Korisnik i Narudžba

- Veza: ima
- Kardinalnost: 1:N

  Korisnik može imati više narudžbi.
  Narudžba pripada jednom korisniku.

2. Korisnik i Recenzija

- Veza: ostavlja
- Kardinalnost: 1:N

  Korisnik može ostaviti više recenzija.
  Recenzija pripada jednom korisniku.

3. Korisnik i Wishlist

- Veza: ima
- Kardinalnost: 1:N

Korisnik može imati više wishlista.
Wishlist pripada jednom korisniku.

4. Wishlist i Proizvod

- Veza: sadrži
- Kardinalnost: M:N

Wishlist može sadržavati više proizvoda.
Proizvod može biti u više wishlista.

5. Recenzija i Proizvod

- Veza: ocjenjuje
- Kardinalnost: N:1

Recenzija je povezana s jednim proizvodom.
Proizvod može imati više recenzija.

6. Narudžba i Proizvod

- Veza: sadrži
- Kardinalnost: M:N
- Veza ima atribute: količina, cijena.

Narudžba može sadržavati više proizvoda.
Proizvod može biti u više narudžbi.

7. Narudžba i Uplata

- Veza: ima
- Kardinalnost: 1:N

Jedna narudžba ima jednu uplatu.
Više uplata mogu biti vezane za različite narudžbe.

8. Narudžba i Dostava

- Veza: ima
- Kardinalnost: 1:1

Jedna narudžba ima jednu dostavu.
Dostava pripada jednoj narudžbi.

9. Dostava i Kurirska služba

- Veza: dostavlja

- Kardinalnost: N:1

  Dostavu vrši jedna kurirska služba.
  Kurirska služba može imati više dostava.

10. Narudžba i Kupon

- Veza: koristi
- Kardinalnost: M:N

  Narudžba može koristiti više kupona.
  Kupon može biti korišten u više narudžbi.

11. Proizvod i Kategorija

- Veza: pripada
- Kardinalnost: M:N

  Proizvod može pripadati više kategorija.
  Jedna kategorija može imati više proizvoda.

12. Proizvod i Povijest cijena

- Veza: ima
- Kardinalnost: 1:N

  Proizvod može imati više zapisa u povijesti cijena.
  Jedan zapis pripada jednom proizvodu.

13. Proizvod i Slika

- Veza: ima
- Kardinalnost: M:N

  Proizvod može imati više slika.
  Slika prikazuje više proizvoda.

14. Proizvod i Skladište

- Veza: nalazi se u
- Kardinalnost: M:N
- Atributi veze: količina.

  Proizvod se može nalaziti u više skladišta.
  Skladište može sadržavati više proizvoda.

15. Skladište i Povijest zaliha

- Veza: prati
- Kardinalnost: 1:N

  Skladište ima više zapisa u povijesti zaliha.
  Svaki zapis se odnosi na jedno skladište.

16. Povijest zaliha i Proizvod

- Veza: ima
- Kardinalnost: N:1

  Više zapisa pripada jednom proizvodu.
  Jedan zapis se odnosi na jedan proizvod.

---

4. RELACIJSKI MODEL

```
PROIZVOD (id (PK), naziv, opis, cijena)
```

```
SLIKA (id (PK), putanja, naziv, opis)
```

```
SLIKA_PROIZVOD (id (PK), proizvod_id (FK), slika_id (FK))
```

```
KATEGORIJA (id (PK), naziv, opis)
```

```
PROIZVOD_KATEGORIJA (id (PK), proizvod_id (FK), kategorija_id (FK))
```

```
KORISNIK (id (PK), ime, prezime, email, lozinka, adresa, grad, drzava, telefon)
```

```
NARUDZBA (id (PK), korisnik_id (FK), datum, status, ukupni_iznos, nacin_placanja)
```

```
NARUDZBA_PROIZVOD (id (PK), narudzba_id (FK), proizvod_id (FK), kolicina, cijena)
```

UPLATA (id (PK), id_narudzba (FK), iznos, datum, status)

RECENZIJA (id (PK), proizvod_id (FK), korisnik_id (FK), ocjena, komentar, datum)

KUPON (id (PK), naziv, tip, vrijednost, status, datum_pocetka, datum_isteka)

KUPON_NARUDZBA (id (PK), kupon_id (FK), narudzba_id (FK))

WISHLIST (id (PK), korisnik_id (FK), naziv, datum)

WISHLIST_PROIZVOD (id (PK), wishlist_id (FK), proizvod_id (FK))

SKLADISTE (id (PK), naziv, adresa, grad, drzava)

SKLADISTE_PROIZVOD (id (PK), skladiste_id (FK), proizvod_id (FK), kolicina)

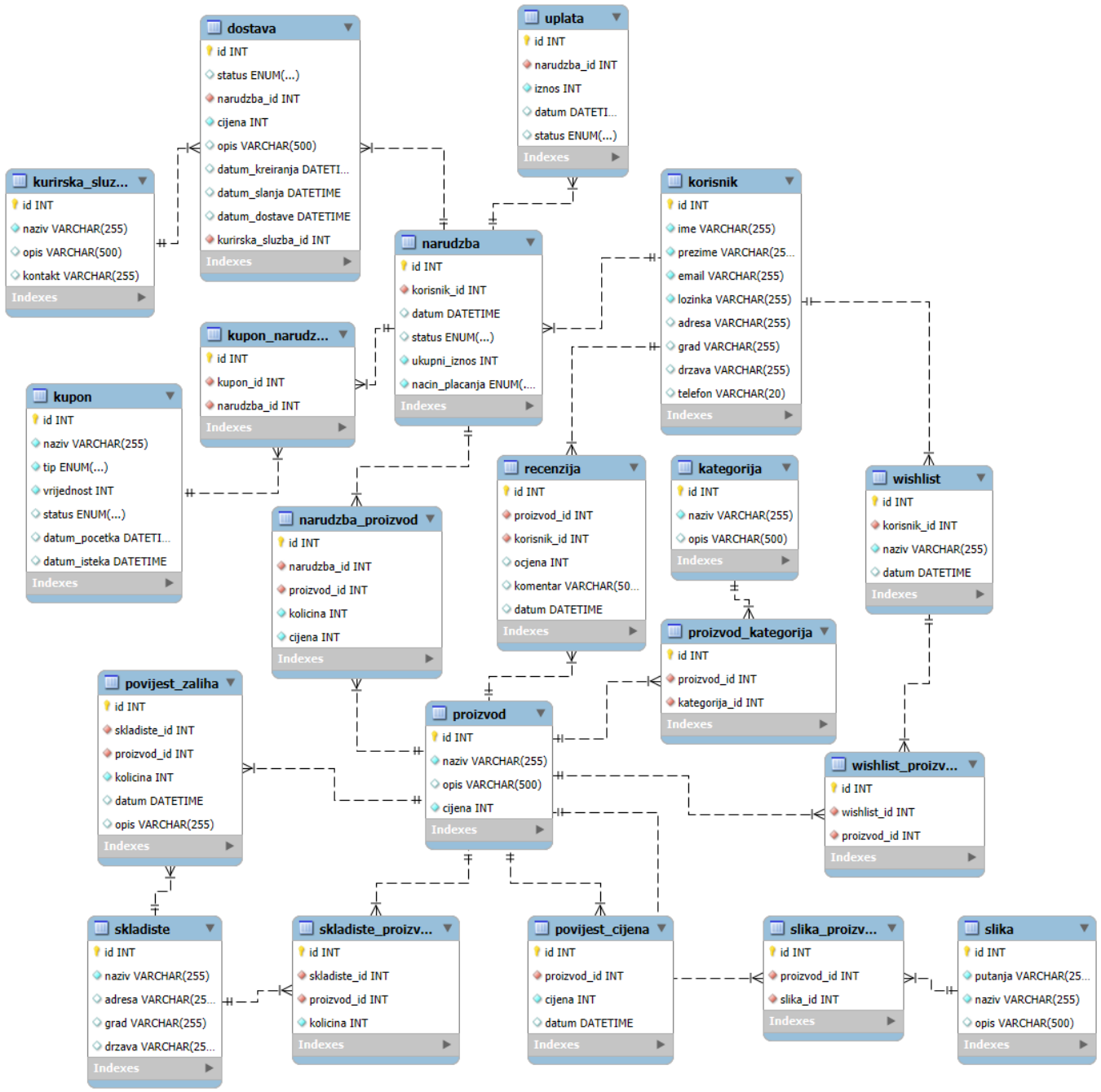
POVIJEST_CIJENA (id (PK), proizvod_id (FK), cijena, datum)

POVIJEST_ZALIHA (id (PK), skladiste_id (FK), proizvod_id (FK), kolicina datum, opis)

KURIRSKA_SLUZBA (id (PK), naziv, opis, kontakt)

DOSTAVA (id (PK), status, narudzba_id (FK), cijena, opis, datum_kreiranja, datum_slanja, datum_dostave, kurirska_sluzba_id (FK))

5. EER DIJAGRAM



---

6. TABLICE

Opis tablica korištenih u bazi podataka.

6.1. PROIZVOD

Tablica `PROIZVOD` sadrži osnovne informacije o proizvodima dostupnim u web trgovini.

- **Primarni ključ:** `id` - Jedinstveni identifikator proizvoda.
- **Atributi:**
  - `naziv` - Naziv proizvoda, koji mora biti definiran.
  - `opis` - Detaljan opis proizvoda, opcionalan.
  - `cijena` - Cijena proizvoda izražena u centima, mora biti veća od 0.

- **Ograničenja:**
  - Provjera (`CHECK`) osigurava da cijena bude pozitivna vrijednost.

6.2. SLIKA

Tablica `SLIKA` sadrži informacije o slikama proizvoda.

- **Primarni ključ:** `id` - Jedinstveni identifikator slike.
- **Atributi:**
  - `putanja` - Putanja do slike na serveru, mora biti jedinstvena.
  - `naziv` - Naziv slike, opisuje sadržaj slike.
  - `opis` - Dodatni opis slike, opcionalan.

6.3. SLIKA_PROIZVOD

Tablica `SLIKA_PROIZVOD` povezuje slike s proizvodima.

- **Primarni ključ:** `id` - Jedinstveni identifikator veze između slike i proizvoda.
- **Strani ključevi:**
  - `proizvod_id` - Referenca na tablicu `PROIZVOD`.
  - `slika_id` - Referenca na tablicu `SLIKA`.

6.4. KATEGORIJA

Tablica `KATEGORIJA` sadrži informacije o kategorijama proizvoda.

- **Primarni ključ:** `id` - Jedinstveni identifikator kategorije.
- **Atributi:**
  - `naziv` - Naziv kategorije, mora biti jedinstven.
  - `opis` - Detaljan opis kategorije, opcionalan.

6.5. PROIZVOD_KATEGORIJA

Tablica `PROIZVOD_KATEGORIJA` povezuje proizvode s kategorijama.

- **Primarni ključ:** `id` - Jedinstveni identifikator veze između proizvoda i kategorije.
- **Strani ključevi:**
  - `proizvod_id` - Referenca na tablicu `PROIZVOD`.
  - `kategorija_id` - Referenca na tablicu `KATEGORIJA`.

6.6. KORISNIK

Tablica `KORISNIK` sadrži informacije o korisnicima web trgovine.

- **Primarni ključ:** `id` - Jedinstveni identifikator korisnika.
- **Atributi:**
  - `ime` - Ime korisnika.
  - `prezime` - Prezime korisnika.
  - `email` - Email korisnika, mora biti jedinstven.
  - `lozinka` - Lozinka korisnika.
  - `adresa`, `grad`, `drzava`, `telefon` - Kontakt podaci korisnika, opcionalni.

6.7. NARUDZBA

Tablica `NARUDZBA` sadrži informacije o narudžbama korisnika.

- **Primarni ključ:** `id` - Jedinstveni identifikator narudžbe.
- **Strani ključ:**
  - `korisnik_id` - Referenca na tablicu `KORISNIK`.
- **Atributi:**
  - `datum` - Datum kreiranja narudžbe.
  - `status` - Status narudžbe (`na čekanju`, `otkazano`, `isporučeno`).
  - `ukupni_iznos` - Ukupna cijena narudžbe izražena u centima.
  - `nacin_placanja` - Način plaćanja (`kreditna kartica`, `pouzeće`, `bankovni transfer`).

6.8. NARUDZBA_PROIZVOD

Tablica `NARUDZBA_PROIZVOD` povezuje narudžbe s proizvodima.

- **Primarni ključ:** `id` - Jedinstveni identifikator veze između narudžbe i proizvoda.
- **Strani ključevi:**
  - `narudzba_id` - Referenca na tablicu `NARUDZBA`.
  - `proizvod_id` - Referenca na tablicu `PROIZVOD`.
- **Atributi:**
  - `kolicina` - Količina proizvoda u narudžbi.
  - `cijena` - Cijena proizvoda izražena u centima.

6.9. UPLATA

Tablica `UPLATA` sadrži informacije o transakcijama vezanim uz narudžbe.

- **Primarni ključ:** `id` - Jedinstveni identifikator uplate.
- **Strani ključ:**
  - `narudzba_id` - Referenca na tablicu `NARUDZBA`.
- **Atributi:**
  - `iznos` - Iznos transakcije izražen u centima.
  - `datum` - Datum transakcije.
  - `status` - Status transakcije (`uspješna`, `neuspješna`).

6.10. RECENZIJA

Tablica `RECENZIJA` sadrži ocjene i komentare korisnika o proizvodima.

- **Primarni ključ:** `id` - Jedinstveni identifikator recenzije.
- **Strani ključevi:**
  - `proizvod_id` - Referenca na tablicu `PROIZVOD`.
  - `korisnik_id` - Referenca na tablicu `KORISNIK`.
- **Atributi:**
  - `ocjena` - Ocjena proizvoda (1-5).
  - `komentar` - Komentar korisnika.
  - `datum` - Datum recenzije.

6.11. KUPON

Tablica `KUPON` sadrži informacije o kuponima za popuste.

- **Primarni ključ:** `id` - Jedinstveni identifikator kupona.
- **Atributi:**
  - `naziv` - Naziv kupona, mora biti jedinstven.
  - `tip` - Tip kupona (`fiksni`, `postotak`, `besplatna dostava`).
  - `vrijednost` - Vrijednost kupona.
  - `status` - Status kupona (`aktivan`, `neaktivan`).
  - `datum_pocetka`, `datum_isteka` - Vrijeme trajanja kupona.
- **Ograničenja:**
  - Provjera (`CHECK`) osigurava ispravne vrijednosti za svaki tip kupona.

6.12. KUPON_NARUDZBA

Tablica `KUPON_NARUDZBA` povezuje kupone s narudžbama.

- **Primarni ključ:** `id` - Jedinstveni identifikator veze između kupona i narudžbe.
- **Strani ključevi:**
  - `kupon_id` - Referenca na tablicu `KUPON`.
  - `narudzba_id` - Referenca na tablicu `NARUDZBA`.

6.13. WISHLIST

Tablica `WISHLIST` sadrži liste želja korisnika.

- **Primarni ključ:** `id` - Jedinstveni identifikator liste želja.
- **Strani ključ:**
  - `korisnik_id` - Referenca na tablicu `KORISNIK`.
- **Atributi:**
  - `naziv` - Naziv liste želja.
  - `datum` - Datum kreiranja liste.

6.14. WISHLIST_PROIZVOD

Tablica `WISHLIST_PROIZVOD` povezuje proizvode s listama želja.

- **Primarni ključ:** `id` - Jedinstveni identifikator veze između liste želja i proizvoda.
- **Strani ključevi:**
  - `wishlist_id` - Referenca na tablicu `WISHLIST`.
  - `proizvod_id` - Referenca na tablicu `PROIZVOD`.

6.15. SKLADISTE

Tablica `SKLADISTE` sadrži informacije o skladištima.

- **Primarni ključ:** `id` - Jedinstveni identifikator skladišta.
- **Atributi:**
  - `naziv` - Naziv skladišta, mora biti jedinstven.

    - `adresa`, `grad`, `drzava` - Lokacija skladišta.

6.16. SKLADISTE_PROIZVOD

Tablica `SKLADISTE_PROIZVOD` povezuje proizvode sa skladištima.

- **Primarni ključ:** `id` - Jedinstveni identifikator veze između skladišta i proizvoda.
- **Strani ključevi:**
    - `skladiste_id` - Referenca na tablicu `SKLADISTE`.
    - `proizvod_id` - Referenca na tablicu `PROIZVOD`.
- **Atributi:**
    - `kolicina` - Količina proizvoda u skladištu.

6.17. POVIJEST_CIJENA

Tablica `POVIJEST_CIJENA` sadrži povijest promjena cijena proizvoda.

- **Primarni ključ:** `id` - Jedinstveni identifikator promjene cijene.
- **Strani ključ:**
    - `proizvod_id` - Referenca na tablicu `PROIZVOD`.
- **Atributi:**
    - `cijena` - Nova cijena proizvoda.
    - `datum` - Datum promjene cijene.

6.18. POVIJEST_ZALIHA

Tablica `POVIJEST_ZALIHA` sadrži povijest promjena zaliha proizvoda u skladištima.

- **Primarni ključ:** `id` - Jedinstveni identifikator promjene zaliha.
- **Strani ključevi:**
    - `skladiste_id` - Referenca na tablicu `SKLADISTE`.
    - `proizvod_id` - Referenca na tablicu `PROIZVOD`.
- **Atributi:**
    - `kolicina` - Nova količina proizvoda.
    - `datum` - Datum promjene zaliha.
    - `opis` - Opis promjene.

6.19. KURIRSKA_SLUZBA

Tablica `KURIRSKA_SLUZBA` sadrži informacije o kurirskim službama.

- **Primarni ključ:** `id` - Jedinstveni identifikator kurirske službe.
- **Atributi:**
    - `naziv` - Naziv kurirske službe, mora biti jedinstven.
    - `opis` - Detaljan opis kurirske službe.
    - `kontakt` - Kontakt informacije.

6.20. DOSTAVA

Tablica `DOSTAVA` sadrži informacije o dostavama narudžbi.

- **Primarni ključ:** `id` - Jedinstveni identifikator dostave.
- **Strani ključevi:**
  - `narudzba_id` - Referenca na tablicu `NARUDZBA`.
  - `kurirska_sluzba_id` - Referenca na tablicu `KURIRSKA_SLUZBA`.
- **Atributi:**
  - `status` - Status dostave (`naručeno`, `u transportu`, `isporučeno`).
  - `cijena` - Cijena dostave izražena u centima.
  - `opis` - Dodatni opis dostave.
  - `datum_kreiranja`, `datum_slanja`, `datum_dostave` - Vremenski podaci vezani uz dostavu.

(Za svaku tablicu kopiraj strukturu opisa kao za KLUB, prilagođeno tvojem projektu.)

---

7. INICIJALIZACIJA

Ovaj vodič objašnjava kako postaviti i pokrenuti projekt baze podataka za web trgovinu. Projekt koristi MySQL i sastoji se od SQL skripti za definiranje sheme baze podataka i učitavanje podataka iz CSV datoteka ili manualno kroz INSERT-ove.

Slijedite ove korake za postavljanje projekta kroz **INSERT-ove**:

1. **Preuzimanje potrebnih datoteka:**
   - Preuzmite SQL skriptne datoteke (`shema.sql`, `insert_data.sql`).
2. **Izvršavanje SQL skripti:**
   - Spojite se na MySQL server koristeći konfiguriranu konekciju.
   - Otvorite SQL skriptu `shema.sql`.
   - Izvršite cijelu skriptu `shema.sql`. Ovo će stvoriti bazu podataka `web_trgovina_bp1` i sve potrebne tablice.
   - Nakon uspješnog izvršavanja `shema.sql`, otvorite SQL skriptu `insert_data.sql`.
   - Izvršite cijelu skriptu `insert_data.sql`. Ovo će popuniti tablice podacima s vrijednostima iz INSERT-ova.

Slijedite ove korake za postavljanje projekta kroz **CSV datoteke**:

1. **Preuzimanje potrebnih datoteka:**
   - Preuzmite SQL skriptne datoteke (`shema.sql`, `load_data_from_csv.sql`).
   - Preuzmite cijelu mapu `data` koja sadrži sve potrebne CSV datoteke.
2. **Postavljanje CSV datoteka za MySQL:**
   - MySQL zahtijeva da se CSV datoteke za `LOAD DATA INFILE` nalaze u sigurnosno odobrenom direktoriju. Da biste pronašli putanju ovog direktorija, izvršite sljedeći SQL upit u MySQL Workbenchu (ili drugom MySQL klijentu):

```
SHOW VARIABLES LIKE "secure_file_priv";
```

- Kopirajte cijelu mapu `data` (sa svim CSV datotekama) u putanju koju ste dobili gornjim upitom. Česta zadana putanja je `C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\`.

3. **Konfiguracija MySQL Workbencha za `LOAD DATA LOCAL INFILE`:**

   - Otvorite MySQL Workbench.
   - Na početnom zaslonu, desnom tipkom miša kliknite na konekciju koju ćete koristiti za spajanje na bazu podataka.
   - Odaberite `Edit Connection...`.
   - Idite na karticu `Advanced`.
   - U polje `Others` (ili slično, ovisno o verziji Workbencha) unesite: `OPT_LOCAL_INFILE=1`.
   - Kliknite `Test Connection` (opcionalno) da provjerite radi li sve ispravno.
   - Kliknite `Close`. Možda će biti potrebno ponovno pokrenuti MySQL Workbench kako bi promjena stupila na snagu.

4. **Izvršavanje SQL skripti:**

   - Spojite se na MySQL server koristeći konfiguriranu konekciju.
   - Otvorite SQL skriptu `shema.sql`.
   - Izvršite cijelu skriptu `shema.sql`. Ovo će stvoriti bazu podataka `web_trgovina_bp1` i sve potrebne tablice.
   - Nakon uspješnog izvršavanja `shema.sql`, otvorite SQL skriptu `load_data_from_csv.sql`.
   - Izvršite cijelu skriptu `load_data_from_csv.sql`. Ovo će popuniti tablice podacima iz CSV datoteka. **Napomena:** Provjerite jesu li putanje do CSV datoteka unutar `load_data_from_csv.sql` ispravno postavljene na direktorij iz koraka 2 (npr. `C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/data/naziv_datoteke.csv`).

---

8. UPITI

Erik Fakin

Upit 1:

Pogled nazvan "top10_najprodavanijih_proizvoda" kombinira podatke iz tablica "proizvod", "narudzba_proizvod" i "narudzba" kako bi prikazao deset najprodavanijih proizvoda u zadnjih 30 dana od određenog datuma.

Unutar pogleda koristi se SELECT naredba za dohvaćanje podataka. SELECT upit odabire stupce "id", "naziv" i "cijena" iz tablice "proizvod", te izračunava ukupnu količinu prodanih jedinica proizvoda koristeći funkciju SUM nad stupcem "kolicina" iz tablice "narudzba_proizvod".

Zatim se koristi INNER JOIN za spajanje tablica "proizvod", "narudzba_proizvod" i "narudzba" koristeći odgovarajuće ključeve. WHERE uvjet filtrira narudžbe koje su napravljene u zadnjih 30 dana od određenog datuma koristeći funkciju DATE_SUB. Rezultat se grupira po stupcima "id", "naziv" i "cijena", te se sortira prema ukupnoj količini prodanih jedinica u opadajućem redoslijedu. Konačno, LIMIT ograničava rezultat na prvih deset proizvoda.

```
CREATE OR REPLACE VIEW top10_najprodavanijih_proizvoda AS
SELECT
    p.id,
    p.naziv,
    p.cijena,
    SUM(np.kolicina) AS ukupna_kolicina
FROM proizvod
INNER JOIN narudzba_proizvod np ON p.id = np.proizvod_id
INNER JOIN narudzba n ON np.narudzba_id = n.id
WHERE n.datum >= DATE_SUB('2025-05-26 00:00:00', INTERVAL 30 DAY)
GROUP BY p.id, p.naziv, p.cijena
ORDER BY ukupna_kolicina DESC
LIMIT 10;
```

Upit 2:

Pogled "info_proizvod" kombinira podatke iz više tablica kako bi prikazao sve informacije o jednom proizvodu, uključujući detalje, slike, kategorije, recenzije i količinu na skladištu.

Unutar pogleda koristi se SELECT naredba za dohvaćanje podataka. SELECT upit odabire stupce "id", "naziv", "opis" i "cijena" iz tablice "proizvod". Koristi se funkcija GROUP_CONCAT za dohvaćanje svih kategorija i slika povezanih s proizvodom i kombiniranje u jedan string, te CONCAT za formatiranje recenzija koje uključuju ocjenu, komentar, datum i ime korisnika. SUM funkcija se koristi za izračun ukupne količine proizvoda na skladištu.

LEFT JOIN se koristi za spajanje tablica "proizvod", "proizvod_kategorija", "kategorija", "slika_proizvod", "slika", "skladiste_proizvod", "recenzija" i "korisnik". WHERE uvjet filtrira podatke za određeni proizvod prema njegovom "id". Rezultat se grupira po stupcu "id".

```
CREATE OR REPLACE VIEW info_proizvod AS
SELECT
    p.id,
    p.naziv,
    p.opis,
    p.cijena,
    GROUP_CONCAT(DISTINCT k.naziv) AS kategorije,
    GROUP_CONCAT(DISTINCT s.putanja) AS slike,
    SUM(DISTINCT sp.kolicina) AS ukupna_kolicina_na_skladistu,
    GROUP_CONCAT(DISTINCT CONCAT(r.ocjena, '::', r.komentar, '::', r.datum, '::',
ko.ime, ' ', ko.prezime) SEPARATOR ' || ') AS recenzije
FROM proizvod p
LEFT JOIN proizvod_kategorija pk ON p.id = pk.proizvod_id
LEFT JOIN kategorija k ON pk.kategorija_id = k.id
LEFT JOIN slika_proizvod spv ON p.id = spv.proizvod_id
LEFT JOIN slika s ON spv.slika_id = s.id
LEFT JOIN skladiste_proizvod sp ON p.id = sp.proizvod_id
LEFT JOIN recenzija r ON p.id = r.proizvod_id
LEFT JOIN korisnik ko ON r.korisnik_id = ku.id
```

```
WHERE p.id = 11
GROUP BY p.id;
```

Upit 3:

Pogled "najcesce_kupljeno_zajedno" kombinira podatke iz tablica "narudzba_proizvod" i "proizvod" kako bi prikazao proizvode koji se najčešće prodaju zajedno s definiranim proizvodom.

Unutar pogleda koristi se SELECT naredba za dohvaćanje podataka. SELECT upit odabire stupce "id" i "naziv" iz tablice "proizvod", te izračunava broj zajedničkih kupovina koristeći funkciju COUNT.

INNER JOIN se koristi za spajanje tablica "narudzba_proizvod" i "proizvod" na temelju zajedničkih narudžbi. WHERE uvjet filtrira podatke za definirani proizvod prema njegovom "id" i isključuje sam proizvod iz rezultata. Rezultat se grupira po stupcu "id", te se sortira prema broju zajedničkih kupovina u opadajućem redoslijedu.

```
CREATE OR REPLACE VIEW najcesce_kupljeno_zajedno AS
SELECT
    p2.id AS proizvod_id,
    p2.naziv AS proizvod_naziv,
    COUNT(*) AS broj_zajednickih_kupovina
FROM narudzba_proizvod np1
INNER JOIN narudzba_proizvod np2 ON np1.narudzba_id = np2.narudzba_id
INNER JOIN proizvod p2 ON np2.proizvod_id = p2.id
WHERE
    np1.proizvod_id = 11
    AND np2.proizvod_id <> np1.proizvod_id
    AND np2.kolicina > 0
GROUP BY p2.id
ORDER BY broj_zajednickih_kupovina DESC;
```

Upit 4:

Pogled "proizvodi_po_kategoriji_i_cijeni" kombinira podatke iz tablica "proizvod", "proizvod_kategorija", "kategorija", "recenzija", "slika_proizvod", "slika" i "skladiste_proizvod" kako bi prikazao proizvode iz odabrane kategorije unutar definiranog raspona cijena.

Unutar pogleda koristi se SELECT naredba za dohvaćanje podataka. SELECT upit odabire stupce "naziv", "cijena" i "slike" iz tablice "proizvod", te izračunava prosječnu ocjenu proizvoda koristeći funkciju AVG i stanje na zalihi koristeći funkciju SUM.

LEFT JOIN se koristi za spajanje tablica na temelju odgovarajućih ključeva. WHERE uvjet filtrira proizvode prema id kategorije i rasponu cijena. Rezultat se grupira po stupcu "id", te se sortira prema cijeni u rastućem redoslijedu i prosječnoj ocjeni u opadajućem redoslijedu.

```
CREATE OR REPLACE VIEW proizvodi_po_kategoriji_i_cijeni AS
SELECT
    p.naziv AS proizvod_naziv,
```

```
    GROUP_CONCAT(DISTINCT s.putanja) AS slike,
    p.cijena AS proizvod_cijena,
    AVG(r.ocjena) AS prosjecna_ocjena,
    SUM(sp.kolicina) AS stanje_na_zalihi
FROM proizvod p
LEFT JOIN proizvod_kategorija pk ON p.id = pk.proizvod_id
LEFT JOIN kategorija k ON pk.kategorija_id = k.id
LEFT JOIN recenzija r ON p.id = r.proizvod_id
LEFT JOIN slika_proizvod spv ON p.id = spv.proizvod_id
LEFT JOIN slika s ON spv.slika_id = s.id
LEFT JOIN skladiste_proizvod sp ON p.id = sp.proizvod_id
WHERE k.id = 1 AND p.cijena BETWEEN 100 AND 500
GROUP BY p.id
ORDER BY p.cijena ASC, prosjecna_ocjena DESC;
```

Upit 5:

Pogled "analiza_prodaje_po_mjesecima" kombinira podatke iz tablica "proizvod", "narudzba_proizvod" i "narudzba" kako bi prikazao analizu prodaje po mjesecima.

Unutar pogleda koristi se SELECT naredba za dohvaćanje podataka. SELECT upit odabire stupce "mjesec" koristeći funkciju MONTH i "naziv" iz tablice "proizvod", te izračunava ukupnu količinu prodanih jedinica i ukupni prihod koristeći funkcije SUM.

INNER JOIN se koristi za spajanje tablica na temelju odgovarajućih ključeva. Rezultat se grupira po stupcima "mjesec" i "id", te se sortira prema mjesecu u rastućem redoslijedu i ukupnom prihodu u opadajućem redoslijedu.

```
CREATE OR REPLACE VIEW analiza_prodaje_po_mjesecima AS
SELECT
    MONTH(n.datum) AS mjesec,
    p.naziv AS proizvod_naziv,
    SUM(np.kolicina) AS ukupna_kolicina,
    SUM(np.kolicina * np.cijena) AS ukupni_prihod
FROM proizvod p
INNER JOIN narudzba_proizvod np ON p.id = np.proizvod_id
INNER JOIN narudzba n ON np.narudzba_id = n.id
GROUP BY mjesec, p.id
ORDER BY mjesec ASC, ukupni_prihod DESC;
```

Tomas Carlin

Upit 1:

Upit koristi tablice "korisnik" i "narduzba" kojoj pridružujemo skaraćnice k i n. U tablici priključujemo id korisnika, potom spajamo sa CONCAT "ime" i "prezime" u stupac "ime_prezime", te stupac "mail"-a i na kraju zbroj ukupnog iznosa u stupac "ukupna_potrosnja".

U nastavku povezujemmo tablice "korsnik" i "narudzba" preko JOIN operacije u kojoj svaki id korisnika je povezan sa svojim id-om iz tablica "narudzba".

Nakon toga grupiramo tablicu po id korisnika kojom postižemo mogućnost zbrajanja narudžbi korisiteći SUM. Koristeći naredbu HAVING dodajemo uvjet o potrošnji većoj od 50000. Na kraju, postavljamo padajući poredak cijena od ukupne potrosnje.

```
SELECT
    k.id AS korisnik_id,
    CONCAT(k.ime, ' ', k.prezime) AS ime_prezime,
    k.email,
    SUM(n.ukupni_iznos) AS ukupna_potrosnja
FROM korisnik k
JOIN narudzba n ON k.id = n.korisnik_id
GROUP BY k.id
HAVING ukupna_potrosnja > 50000
ORDER BY ukupna_potrosnja DESC;
```

Upit koristi tablice "povijest_cijena" i "proizvod", kojima dodjeljuje skraćenice pc i p. U tablici dohvaćamo id proizvoda, naziv proizvoda, datum promjene cijene te trenutnu cijenu.

Zatim, koristimo funkciju LAG kako bismo dohvatili prethodnu cijenu za isti proizvod, pazeći da se usporedba vrši po datumu (ORDER BY datum) unutar svakog proizvoda (PARTITION BY proizvod_id).

U nastavku računamo postotnu promjenu cijene tako da od trenutne cijene oduzmemo prethodnu, podijelimo s prethodnom i pomnožimo s 100, te zaokružimo rezultat na dvije decimale.

Na kraju koristimo JOIN da spojimo podatke iz tablice "proizvod" radi naziva proizvoda, te sortiramo rezultat po id-u proizvoda i datumu promjene kako bismo dobili bolji pregled promjena po vremenskom slijedu.

Upit 2:

```
SELECT
    pc.proizvod_id,
    p.naziv AS proizvod_naziv,
    pc.datum,
    pc.cijena AS cijena_u_centima,
    LAG(pc.cijena) OVER (PARTITION BY pc.proizvod_id ORDER BY pc.datum) AS
prethodna_cijena,
    ROUND(
        ((pc.cijena - LAG(pc.cijena) OVER (PARTITION BY pc.proizvod_id ORDER BY
pc.datum)) /
        LAG(pc.cijena) OVER (PARTITION BY pc.proizvod_id ORDER BY pc.datum)) *
100, 2)
        AS postotna_promjena
FROM povijest_cijena pc
JOIN proizvod p ON pc.proizvod_id = p.id
ORDER BY pc.proizvod_id, pc.datum;
```

Upit 3:

Upit koristi tablice "skladiste_proizvod", "proizvod" i "skladiste", kojima dodjeljujemo skraćenice sp, p i s. U SELECT dijelu prikazujemo id i naziv proizvoda, id i naziv skladišta, te količinu tog proizvoda u skladištu.

Tablice spajamo pomoću JOIN-a, gdje sp.proizvod_id povezujemo s p.id, a sp.skladiste_id sa s.id, kako bismo mogli prikazati nazive proizvoda i skladišta umjesto samo id-eva.

Glavna logika nalazi se u WHERE dijelu, gdje tražimo samo one retke u kojima kombinacija (proizvod_id, kolicina) odgovara najvećoj količini za svaki proizvod. To radimo pomoću IN i podupita koji koristi MAX(kolicina) i GROUP BY za grupiranje po proizvodu.

Konačan rezultat prikazuje sva skladišta koja imaju najveću zalihu za svaki proizvod.

```
SELECT
    sp.proizvod_id,
    p.naziv AS proizvod_naziv,
    sp.skladiste_id,
    s.naziv AS skladiste_naziv,
    sp.kolicina
FROM skladiste_proizvod sp
JOIN proizvod p ON sp.proizvod_id = p.id
JOIN skladiste s ON sp.skladiste_id = s.id
WHERE (sp.proizvod_id, sp.kolicina) IN (
    SELECT proizvod_id, MAX(kolicina)
    FROM skladiste_proizvod
    GROUP BY proizvod_id
);
```

Upit 4:

Upit koristi tablice "narudzba", "kupon_narudzba", "kupon" i "dostava", kojima se dodjeljuju skraćenice n, kn, k i d.

Prikazujemo id narudžbe i datum narudžbe, ukupni iznos proizvoda, vrijednost kupona (ili 0 ako nije primijenjen), cijenu dostave i na kraju računamo ukupan trošak narudžbe.

Spajamo tablice pomoću LEFT JOIN, kako bi sve narudžbe bile prikazane, čak i ako nema kupona ili dostave.

Izračun ukupnog troška se radi tako da se od cijene proizvoda oduzme vrijednost kupona i doda cijena dostave. Ako nema kupona, koristi se nula.

```
SELECT
    n.id AS narudzba_id,
    n.datum AS datum_narudzbe,
    n.ukupni_iznos AS ukupni_iznos_proizvoda,
    IFNULL(k.vrijednost, 0) AS vrijednost_kupona,
    d.cijena AS cijena_dostave,
    (n.ukupni_iznos - IFNULL(k.vrijednost, 0) + d.cijena) AS ukupni_trosak
```

```
FROM narudzba n
LEFT JOIN kupon_narudzba kn ON n.id = kn.narudzba_id
LEFT JOIN kupon k ON kn.kupon_id = k.id
LEFT JOIN dostava d ON n.id = d.narudzba_id;
```

Evan Dagostini

Upit 1:

Pogled nazvan "top_kupci_30_dana" kombinira podatke iz tablica "korisnik" i "narudzba" kako bi prikazao deset korisnika koji su u zadnjih 30 dana ostvarili najveću ukupnu potrošnju.

SELECT naredba dohvaća korisnikove osnovne podatke ("id", "ime", "prezime", "email") te koristi COUNT za broj narudžbi i SUM za izračun ukupne potrošnje ("ukupni_iznos") unutar tog perioda.

INNER JOIN spaja tablice "korisnik" i "narudzba" prema "korisnik_id", dok WHERE uvjet filtrira samo narudžbe koje su napravljene unutar posljednjih 30 dana pomoću funkcije DATE_SUB. Rezultat se grupira po korisniku, sortira po ukupnoj potrošnji i ograničava na prvih deset korisnika.

```
CREATE OR REPLACE VIEW top_kupci_30_dana AS
SELECT
    k.id AS korisnik_id,
    k.ime,
    k.prezime,
    k.email,
    COUNT(n.id) AS broj_narudzbi,
    SUM(n.ukupni_iznos) AS ukupna_potrosnja
FROM korisnik k
JOIN narudzba n ON k.id = n.korisnik_id
WHERE n.datum >= DATE_SUB('2025-05-26 00:00:00', INTERVAL 30 DAY)
GROUP BY k.id, k.ime, k.prezime, k.email
ORDER BY ukupna_potrosnja DESC
LIMIT 10;
```

Upit 2:

Pogled "proizvodi_bez_narudzbi_60_dana" koristi LEFT JOIN između tablica "proizvod", "narudzba_proizvod" i "narudzba" kako bi prikazao sve proizvode koji u zadnjih 60 dana nisu imali nijednu narudžbu.

Upit dohvaća osnovne podatke o proizvodu i koristi LEFT JOIN kako bi uključio sve proizvode, čak i one koji nemaju pripadajuće narudžbe. Koristi se COALESCE(SUM(...), 0) kako bi se broj prodanih jedinica prikazao kao 0 za takve slučajeve.

WHERE dio filtrira samo narudžbe u zadnjih 60 dana, a HAVING uvjet osigurava da se prikazuju samo proizvodi s 0 prodanih jedinica. Rezultat se grupira po proizvodu.

```
CREATE OR REPLACE VIEW proizvodi_bez_narudzbi_60_dana AS
SELECT
```

```
    p.id,
    p.naziv,
    p.cijena,
    COALESCE(SUM(np.kolicina), 0) AS kolicina_prodanih
FROM proizvod p
LEFT JOIN narudzba_proizvod np ON p.id = np.proizvod_id
LEFT JOIN narudzba n ON np.narudzba_id = n.id AND n.datum >= DATE_SUB('2025-05-26
00:00:00', INTERVAL 60 DAY)
GROUP BY p.id, p.naziv, p.cijena
HAVING COALESCE(SUM(np.kolicina), 0) = 0;
```

Upit 3:

Pogled "kupon_statistika" prikazuje osnovne informacije o aktivnim kuponima iz tablice "kupon", te broj narudžbi u kojima je svaki kupon iskorišten, koristeći tablicu "kupon_narudzba".

U SELECT dijelu dohvaćaju se atributi poput naziva, tipa i vrijednosti kupona, dok se COUNT koristi za brojanje koliko puta je kupon iskorišten. MIN i MAX prikazuju raspon trajanja kupona (početak i kraj).

Koristi se LEFT JOIN između "kupon" i "kupon_narudzba" kako bi se prikazali i kuponi koji još nisu iskorišteni. WHERE ograničava rezultat samo na aktivne kupone. Rezultat se grupira po kuponu.

```
CREATE OR REPLACE VIEW kupon_statistika AS
SELECT
    k.id AS kupon_id,
    k.naziv,
    k.tip,
    k.vrijednost,
    COUNT(kn.narudzba_id) AS broj_koristenja,
    MIN(k.datum_pocetka) AS od_kada,
    MAX(k.datum_isteka) AS do_kada
FROM kupon k
LEFT JOIN kupon_narudzba kn ON k.id = kn.kupon_id
WHERE k.status = 'aktivan'
GROUP BY k.id, k.naziv, k.tip, k.vrijednost;
```

Pogled "povijest_zaliha_proizvoda" kombinira podatke iz tablica "povijest_zaliha", "proizvod" i "skladiste" kako bi prikazao sve zabilježene promjene zaliha za jedan određeni proizvod, uključujući količinu, skladište, opis promjene i vrijeme.

SELECT dohvaća naziv proizvoda, naziv skladišta, količinu i opis promjene, a rezultat se sortira po vremenu u silaznom redoslijedu.

Koristi se JOIN kako bi se povezale sve tri tablice na temelju stranih ključeva. WHERE ograničava prikaz na konkretan proizvod (npr. proizvod_id = 5), a ORDER BY prikazuje najnovije promjene na vrhu.

Upit 4:

```sql
CREATE OR REPLACE VIEW povijest_zaliha_proizvoda AS
SELECT
    pz.proizvod_id,
    p.naziv AS proizvod_naziv,
    s.naziv AS skladiste_naziv,
    pz.kolicina,
    pz.opis,
    pz.datum
FROM povijest_zaliha pz
JOIN proizvod p ON p.id = pz.proizvod_id
JOIN skladiste s ON s.id = pz.skladiste_id
WHERE pz.proizvod_id = 5
ORDER BY pz.datum DESC;
```

Evan Anić Parencan

Upit 1:

Upit "Korisnika koji su naručili više puta isti proizvod i koliko puta".

"ponovljene_kupnje_istog_proizvoda" dohvaća podatke iz tablica: "korisnik", "narudzba", "narudzba_proizvod" i "proizvod" kako bi prikazali korisnike koji su više puta kupili isti proizvod.

Korištenjem naredbe "SELECT" dohvaćamo korisničke i proizvodne podatke, dok sa naredbom "COUNT(*)" brojimo koliko je puta određeni korisnik kupio isti proizvod.

Tablice se spajaju pomoću "JOIN" naredbi, gdje tablica "narudzba" povezuje korisnike s narudžbama, a tablica "narudzba_proizvod" s proizvodima. Koristimo "GROUP BY" za grupiranje po korisniku i proizvodu, a "HAVING" filtrira rezultate kako bi prikazali samo one korisnike koji su isti proizvod kupili više od jednog puta. Rezultat se na kraju sortira po broju kupnji i po imenu korisnika.

```sql
SELECT
    k.id AS korisnik_id,
    CONCAT(k.ime, ' ', k.prezime) AS ime_prezime,
    p.id AS proizvod_id,
    p.naziv AS naziv_proizvoda,
    COUNT(*) AS broj_ponovljenih_kupnji
FROM narudzba n
JOIN narudzba_proizvod np ON n.id = np.narudzba_id
JOIN korisnik k ON n.korisnik_id = k.id
JOIN proizvod p ON np.proizvod_id = p.id
GROUP BY k.id, p.id
HAVING broj_ponovljenih_kupnji > 1
ORDER BY broj_ponovljenih_kupnji DESC, ime_prezime;
```

Upit 2:

Upit "Pregleda proizvoda s ukupnom zaradom i brojem skladišta u kojima se nalaze".

"zarada_i_skladista_po_proizvodu" spaja podatke iz tablica "proizvod", "narudzba_proizvod" i "skladiste_proizvod" s krajnjim rezultatom prikazivanja ukupne zarade po proizvodu i po broja skladišta u kojima se proizvod nalazi.

"SELECT-om" dohvaćamo osnovne informacije o proizvodu, dok "SUM (np.kolicina * np.cijena)" računa ukupnu zaradu, a "COUNT (DISCTINCT sp.skladiste_id)" broj skladišta.

Koristimo "LEFT JOIN" kako bi prikazali i time uračunali, one proizvode koji se možda ne nalaze u skladištima ili nisu još naručeni. Grupiramo podatke po ID-u proizvoda, a sortiramo po ukupnoj zaradi kako bi lakše vidjeli koji proizvodi dovode najveću zaradu.

```
SELECT
    p.id AS proizvod_id,
    p.naziv AS naziv_proizvoda,
    p.cijena,
    SUM(np.kolicina * np.cijena) AS ukupna_zarada,
    COUNT(DISTINCT sp.skladiste_id) AS broj_skladista
FROM proizvod p
LEFT JOIN narudzba_proizvod np ON p.id = np.proizvod_id
LEFT JOIN skladiste_proizvod sp ON p.id = sp.proizvod_id
GROUP BY p.id, p.naziv, p.cijena
ORDER BY ukupna_zarada DESC;
```

Upit 3:

Upit "Pregled korisnika koji su koristili više različitih kupona".

"korisnici_s_razlicitim_kuponima" spaja podatke iz tablica "korisnik", "narudzba", "kupon_narudzba" i "kupon" kako bi prikazao korisnike koji su koristili više od jednog različitog kupona u svojim narudžbama.

"SELECT-om" dohvaćamo osnovne informacije o korisniku, uključujući njegov ID, ime i prezime, broj različitih kupona koje je koristio te popis naziva tih kupona. Funkcija "COUNT(DISTINCT kn.kupon_id)" računa broj jedinstvenih kupona, dok "GROUP_CONCAT(DISTINCT ku.naziv)" dohvaća nazive kupona i formatira ih u jedan string.

Koristimo "JOIN" za spajanje tablica kako bismo povezali korisnike s njihovim narudžbama i kuponima. Grupiramo podatke po ID-u korisnika, a "HAVING" filtrira rezultate kako bi prikazao samo korisnike koji su koristili više od jednog kupona. Rezultat se sortira prema broju različitih kupona u opadajućem redoslijedu.

```
SELECT
    k.id AS korisnik_id,
    CONCAT(k.ime, ' ', k.prezime) AS ime_prezime,
    COUNT(DISTINCT kn.kupon_id) AS broj_razlicitih_kupona,
    GROUP_CONCAT(DISTINCT ku.naziv ORDER BY ku.naziv SEPARATOR ', ') AS kuponi
FROM korisnik k
JOIN narudzba n ON k.id = n.korisnik_id
JOIN kupon_narudzba kn ON n.id = kn.narudzba_id
JOIN kupon ku ON kn.kupon_id = ku.id
GROUP BY k.id
```

```
HAVING broj_razlicitih_kupona > 1
ORDER BY broj_razlicitih_kupona DESC;
```

Upit 4:

Upit "Prosječni broj proizvoda po narudžbi po korisniku i njihova ukupna količina".

"prosjecno_proizvoda_po_narudzbi" prikazuje prosječnu količinu proizvoda pri korisničkovoj narudžbi, te i njihovu ukupnu količinu i broj narudžbi.

Koristimo "SUM (np.kolicina)" za ukupnu količinu proizvoda i COUNT (DISTINCT n.id) za broj narudžbi. Rezultat prosjeka dobijemo tako da podijelimo ukupnu količinu s brojem narudžbi.

Tablice "korisnik", "narudzba" i "narudzba_proizvod" povezuju se pomoću "JOIN" naredbi. Grupiramo po korisniku, a "HAVING" osigurava da prikažemo samo korisnike s više narudžbi. Sortiranje se radi prema prosječnoj količini proizvoda po narudžbi.

```
SELECT
    k.id AS korisnik_id,
    CONCAT(k.ime, ' ', k.prezime) AS ime_prezime,
    COUNT(DISTINCT n.id) AS broj_narudzbi,
    SUM(np.kolicina) AS ukupna_kolicina_proizvoda,
    SUM(np.kolicina) / COUNT(DISTINCT n.id) AS prosjecno_proizvoda_po_narudzbi
FROM korisnik k
JOIN narudzba n ON k.id = n.korisnik_id
JOIN narudzba_proizvod np ON n.id = np.narudzba_id
GROUP BY k.id
HAVING broj_narudzbi > 1
ORDER BY prosjecno_proizvoda_po_narudzbi DESC;
```

Denis Beletić

Upit 1:

Upit "najbolje_ocijenjeni_proizvodi" dohvaća podatke iz tablica "proizvod", "recenzija" i "korisnik" kako bi prikazali proizvode s najvišim prosječnim ocjenama, skupa s brojem recenzija i podacima o korisnicima koji su ih ostavili.

Koristimo SELECT naredbu za dohvaćanje podataka, odnosno stupce "id", "naziv" i "cijena" iz tablice "proizvod". Izračunavamo prosječnu ocjenu proizvoda koristeći AVG funkciju i broj recenzija koristeći funkciju COUNT.

Kako bi dohvatili sve proizvode, uključujući i one koji nemaju niti jednu recenziju, koristimo LEFT JOIN za spajanje tablica "proizvod", "recenzija" i "korisnik. Filtriramo proizvode čija je prosječna ocjena veća od 4 pomoću HAVING uvjeta, a rezultat grupiramo po stupcu "id" i sortiramo prema prosječnoj ocjeni u padajućem redoslijedu.

```
```

```
SELECT
    p.id AS proizvod_id,
    p.naziv AS proizvod_naziv,
    p.cijena AS proizvod_cijena,
    AVG(r.ocjena) AS prosjecna_ocjena,
    COUNT(r.id) AS broj_recenzija,
    GROUP_CONCAT(DISTINCT CONCAT(k.ime, ' ', k.prezime) SEPARATOR ', ') AS
korisnici
FROM proizvod AS p
LEFT JOIN recenzija AS r ON p.id = r.proizvod_id
LEFT JOIN korisnik AS k ON r.korisnik_id = k.id
GROUP BY p.id
HAVING AVG(r.ocjena) > 4
ORDER BY prosjecna_ocjena DESC;
```

Upit 2:

Upit "analiza_najcesce_koristenih_kupona" dohvaća podatke iz tablica "kupon", "kupon_narudzba" i "narudzba", tako da bi mogli prikazati analizu kupona koji su najčešće korišteni. Prikazuju se podaci o broju korištenja, ukupnoj vrijednosti popusta i prosječnoj vrijednosti popusta po narudžbi.

Pomoću SELECT-a dohvaćamo stupce "id" i "naziv" iz tablice "kupon". COUNT nam omogućuje računanje ukupnog broja korištenja kupona, SUM omogućuje računanje ukupnu vrijednost popusta a AVG omogućuje računanje prosječne vrijednosti popusta.

INNER JOIN-om spajamo tablica "kupon", "kupon_narudzba" i "narudzba", što nam omogućuje dohvaćivanje svih korištenih kupona u narudžbama. Rezultat grupiramo po stupcu "id" kako bismo dobili agregirane podatke za svaki kupon. Rezultat filtriramo s HAVING uvjetom zato da bi smo prikazali samo kupone koji su korišteni više od 3 puta. Na kraju, sortiramo prema broju korištenja u padajućem redoslijedu.

```
SELECT
    k.id AS kupon_id,
    k.naziv AS naziv_kupona,
    COUNT(kn.narudzba_id) AS broj_koristenja,
    SUM(k.vrijednost) AS ukupna_vrijednost_popusta,
    AVG(k.vrijednost) AS prosjecna_vrijednost_popusta
FROM kupon AS k
INNER JOIN kupon_narudzba AS kn ON k.id = kn.kupon_id
INNER JOIN narudzba AS n ON kn.narudzba_id = n.id
GROUP BY k.id
HAVING broj_koristenja > 3
ORDER BY broj_koristenja DESC;
```

Upit 3:

Upit "analiza_kupaca_po_potrosnji" kombinira podatke iz tablica "korisnik" i "narudzba", na način da napravimo analizu kupaca prema njihovoj ukupnoj potrošnji, broj narudžbi i prosječnu vrijednost narudžbe.

SELECT-om dohvaćamo stupce "id", "ime" i "prezime" iz tablice "korisnik". Ukupnu potrošnju računamo pomoću SUM, broj narudžbi pomoću COUNT i prosječnu vrijednost pomoću AVG.

Koristimo LEFT JOIN za spajanje tablica "korisnik" i "narudzba". Ovako možemo dohvatiti sve korisnike, čak i one bez ikakvih narudžbi. Rezultat grupiramo po stupcu "id" i sortiramo prema ukupnoj potrošnji u padajućem redoslijedu.

```
SELECT
    k.id AS korisnik_id,
    CONCAT(k.ime, ' ', k.prezime) AS ime_prezime,
    COUNT(n.id) AS broj_narudzbi,
    SUM(n.ukupni_iznos) AS ukupna_potrosnja,
    AVG(n.ukupni_iznos) AS prosjecna_vrijednost_narudzbe
FROM korisnik AS k
LEFT JOIN narudzba AS n ON k.id = n.korisnik_id
GROUP BY k.id
ORDER BY ukupna_potrosnja DESC;
```

Upit 4:

Upit "promjene_zaliha_po_skladistima" kombinira podatke iz tablica "povijest_zaliha", "proizvod" i "skladiste" kako bi mogli analizirati promjene zaliha po skladištima za određeni proizvod.

SELECT naredbom odabiremo stupce "id" i "naziv" iz tablica "proizvod" i "skladiste", te prikazujemo količinu, opis i datum promjene zaliha.

INNER JOIN koristimo kako bi spojili tablice "povijest_zaliha", "proizvod" i "skladiste" na temelju odgovarajućih ključeva. WHERE uvjet filtrira podatke za određeni proizvod prema njegovom "id"-u. Rezultat se sortira prema datumu promjene u padajućem redoslijedu.

```
SELECT
    pz.skladiste_id,
    s.naziv AS skladiste_naziv,
    pz.proizvod_id,
    p.naziv AS proizvod_naziv,
    pz.kolicina,
    pz.opis,
    pz.datum
FROM povijest_zaliha AS pz
INNER JOIN proizvod p ON pz.proizvod_id = p.id
INNER JOIN skladiste s ON pz.skladiste_id = s.id
WHERE pz.proizvod_id = 11 -- zamijeniti po potrebi
ORDER BY pz.datum DESC;
```

---

9. ZAKLJUČAK

Projekt nam je poboljšao razumijevanje procesa izrade i implementacije baze podataka za web trgovinu. Kroz praktičan rad stekli smo uvid u sveukupni proces razvoja - od planiranja relacijskog modela do njegove implementacije u SQL-u.

Tijekom projekta morali smo riješiti određene probleme, kao npr. neispravno modeliranje relacijskog modela i nepravilno povezivanje entiteta. S vremenom smo kroz učenje i malo *trial and error*-a uspjeli razumjeti greške te ih ispravili tijekom razvoja baze.

Sustav nam olakšava rad s podacima, omogućuje pregled promjena kroz vrijeme i izradu analitičkih izvještaja. Ove funkcionalnosti omogućuju primjenu sustava u stvarnome svijetu, pa čak i na drugim područjima s nekoliko promjena.

Zaključno, projekt nam je dao skoro *first hand experience* u radu s bazama podataka - naučili smo puno više nego samo iz knjiga, što je i očekivano jer smo morali sami ispravljati greške i rješavati probleme. Takvo iskustvo sigurno će nam biti korisno ako budemo imali slične zadatke u budućnosti.
