**VIŠJA STROKOVNA ŠOLA ACADEMIA**

**MARIBOR**

**PROJEKTNA NALOGA**

**pri predmetu: Zbirke podatkov 1**

Kandidatka: »Kristijana Marušić«

Vrsta študija: študentka izrednega študija

Študijski program: Programsko inženirstvo

Predavatelj: Srečko Zorman

Maribor, študijsko leto 2023/2024

**IZJAVA O AVTORSTVU PROJEKTNE NALOGE**

Podpisana »Kristijana Marušić«, sem avtorica projektne naloge.

S svojim podpisom zagotavljam, da:

* je predloženo delo izključno rezultat mojega dela,
* sem poskrbela, da so dela in mnenja drugih avtorjev, ki jih uporabljam v predloženi nalogi, navedena oz. citirana skladno s pravili Višje strokovne šole Academia Maribor,
* se zavedam, da je plagiatorstvo – predstavljanje tujih del oz. misli, kot moje lastne kaznivo po Zakonu o avtorski in sorodnih pravicah – ZASP (Ur. l. RS 16/07 – uradno prečiščeno besedilo, 68/08, 110/2013, 56/2015 63/16, 59/19 in 130/22), prekršek pa podleže tudi ukrepom Višje strokovne šole Academia Maribor skladno z njenimi pravili,
* skladno z 32.a členom ZASP dovoljujem Višji strokovni šoli Academia Maribor objavo na spletnem portalu šole.

»Ljubljana«, »April, 2024«

Kazalo vsebine

[ERD načrt podatkovne baze 1](#_Toc164100499)

[Tehnična dokumentacija 3](#_Toc164100500)

[Izdelava fizične zbirke podatkov v PostgreSQL-u 5](#_Toc164100501)

[Tabele napolnimo s testnimi podatki (INSERT stavki) 9](#_Toc164100502)

[Smiselnost rezultatov poizvedb za informacije upravljanja 11](#_Toc164100503)

[Ocena prihodkov poslovanja za storitve zavoda za knjige v izbranem obdobju 13](#_Toc164100504)

[GUI vmesnik 14](#_Toc164100505)

[Konkretna implementacija 15](#_Toc164100506)

[SQL poizvedbe 16](#_Toc164100507)

[CREATE ROLE, GRANT 17](#_Toc164100508)

[Pravila arhiviranja zbirke podatkov 18](#_Toc164100509)

[Preizkus uporabniških zahtev (funkcionalnosti) 19](#_Toc164100510)

[Ocena uspešnosti preizkušanja in opis izbora testnih podatkov 22](#_Toc164100511)

[Izvedba in preizkus sistemskih zahtev. 23](#_Toc164100512)

[Ustvarjanje velikega obsega podatkov in preverjanje odzivnosti sistema 23](#_Toc164100513)

[Ocena učinkovitosti in rešitve podatkovne zbirke glede na uporabniške in sistemske zahteve ter predlog izboljšave 24](#_Toc164100514)

[Literatura 26](#_Toc164100515)

# ERD načrt podatkovne baze

1. **drzave**:

* Povezava z **zalozniki**: Preko stolpca **'drzave\_id'**, ki je tuji ključ v tabeli **'zalozniki'**.

1. **zalozniki**:

* Povezava z **knjige\_gradivo**: Preko stolpca **'zalozniki\_id'**, ki je tuji ključ v tabeli **'knjige\_gradivo'**.

1. **kljucne\_besede:**

* Povezava z **knjige\_gradivo\_kljucne\_besede:** Preko stolpca **'kljucne\_besede\_id'**, ki je tuji ključ v tabeli **'knjige\_gradivo\_kljucne\_besede'**.

1. **jeziki**:

* Povezava z **knjige\_gradivo:** Preko stolpca **'jeziki\_id'**, ki je tuji ključ v tabeli **'knjige\_gradivo'**.

1. **knjige\_gradivo:**

* Povezava z **knjige\_gradivo\_kljucne\_besede:** Preko stolpca **'knjige\_gradivo\_id'**, ki je tuji ključ v tabeli **'knjige\_gradivo\_kljucne\_besede'**.
* Povezava z **zalozniki**: Preko stolpca **'zalozniki\_id'**, ki je tuji ključ v tabeli **'zalozniki'**.
* Povezava z **jeziki**: Preko stolpca **'jeziki\_id'**, ki je tuji ključ v tabeli **'jeziki'**.

1. **knjige\_gradivo\_kljucne\_besede:**

* Povezava z **knjige\_gradivo:** Preko stolpca **'knjige\_gradivo\_id'**, ki je tuji ključ v tabeli **'knjige\_gradivo'**.
* Povezava z **kljucne\_besede:** Preko stolpca **'kljucne\_besede\_id'**, ki je tuji ključ v tabeli **'kljucne\_besede'**.

1. **osebe**:

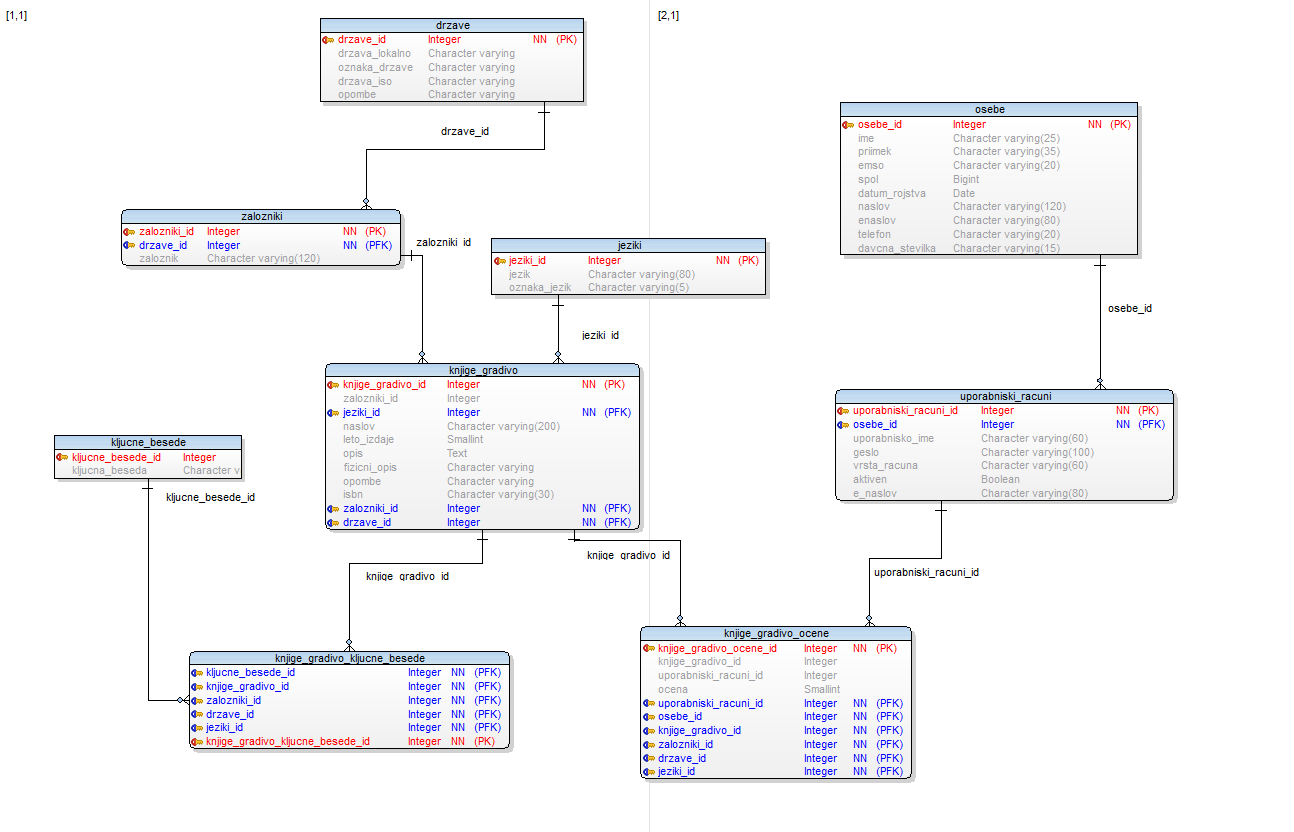
* Povezava z **uporabniski\_racuni:** Preko stolpca **'osebe\_id'**, ki je tuji ključ v tabeli **'uporabniski\_racuni'**.

1. **uporabniski\_racuni:**

* Povezava z **knjige\_gradivo\_ocene:** Preko stolpca **'uporabniski\_racuni\_id'**, ki je tuji ključ v tabeli **'knjige\_gradivo\_ocene'**.

1. **knjige\_gradivo\_ocene:**

* Povezava z **knjige\_gradivo:** Preko stolpca **'knjige\_gradivo\_id'**, ki je tuji ključ v tabeli **'knjige\_gradivo'**.
* Povezava z **uporabniski\_racuni:** Preko stolpca **'uporabniski\_racuni\_id'**, ki je tuji ključ v tabeli **'uporabniski\_racuni'**.



*Model v Toad Data Modeler-ju*

# Tehnična dokumentacija

1. **Podatkovna analiza**

* Namen: Zbirka podatkov za knjižnično gradivo, zalagalce, ključne besede, jezike, osebe, uporabniške račune in ocene knjižničnega gradiva.
* Cilji: Upravljanje informacij o knjigah, njihovih avtorjih, ključnih besedah, jezikih, osebah (uporabniki, avtorji), uporabniških računih in ocenah knjižničnega gradiva.

1. **Veljavnostna pravila**

* Omejitve preverjanja:

Starost oseb mora biti večja od 18 let.

* Omejitve edinstvenosti:

Oznake držav v tabeli drzave morajo biti edinstvene.

Ključne besede v tabeli kljucne\_besede morajo biti edinstvene.

Oznake jezikov v tabeli jeziki morajo biti edinstvene.

Uporabniška imena v tabeli uporabniski\_racuni morajo biti edinstvena.

1. **Opis entitet**

* Tabela **drzave**: Vsebuje informacije o državah, vključno z lokalnim imenom, ISO oznako in opombami.
* Tabela **zalozniki**: Vsebuje informacije o zalagalcih knjižničnega gradiva.
* Tabela **kljucne\_besede:** Vsebuje ključne besede za kategorizacijo knjižničnega gradiva.
* Tabela **jeziki**: Vsebuje informacije o jezikih, v katerih je na voljo knjižnično gradivo.
* Tabela **knjige\_gradivo:** Vsebuje informacije o knjigah, vključno z naslovom, letom izdaje, opisom in ISBN.
* Tabela **knjige\_gradivo\_kljucne\_besede:** Povezovalna tabela, ki povezuje knjige z njihovimi ključnimi besedami.
* Tabela **osebe**: Vsebuje osebne podatke o uporabnikih, kot so ime, priimek, datum rojstva, naslov itd.
* Tabela **uporabniski\_racuni:** Vsebuje informacije o uporabniških računih, vključno z uporabniškim imenom, geslom in e-naslovom.
* Tabela **knjige\_gradivo\_ocene:** Vsebuje ocene uporabnikov za knjižnično gradivo.

1. **Povzetek relacij**

* Tabela **knjige\_gradivo** je povezana s tabelo **zalozniki** preko tuji ključ **zalozniki\_id**.
* Tabela **knjige\_gradivo** je povezana s tabelo **jeziki** preko tuji ključ **jeziki\_id**.
* Tabela **knjige\_gradivo\_kljucne\_besede** vzpostavlja povezavo med knjigami in ključnimi besedami preko tuji ključev.
* Tabela **uporabniski\_racuni** je povezana s tabelo **osebe** preko tuji ključ **osebe\_id**.
* Tabela **knjige\_gradivo\_ocene** je povezana s tabelama **knjige\_gradivo** in **uporabniski\_racuni** preko tuji ključev.

1. **Povzetek atributov**

|  |  |
| --- | --- |
| **Entiteta** | **Atributi** |
| države | drzave\_id (serial), drzava\_lokalno (varchar), oznaka\_drzave (varchar), drzava\_iso (varchar), opombe (varchar) |
| založniki | zalozniki\_id (serial), drzave\_id (integer), zaloznik (varchar) |
| ključne\_besede | kljucne\_besede\_id (serial), kljucna\_beseda (varchar) |
| jeziki | jeziki\_id (serial), jezik (varchar), oznaka\_jezik (varchar) |
| knjige\_gradivo | knjige\_gradivo\_id (serial), zalozniki\_id (integer), jeziki\_id (integer), naslov (varchar), leto\_izdaje (smallint), opis (text), fizicni\_opis (varchar), opombe (varchar), isbn (varchar) |
| knjige\_gradivo\_ključne\_besede | knjige\_gradivo\_kljucne\_besede\_id (serial), knjige\_gradivo\_id (integer), kljucne\_besede\_id (integer), zaporedno\_mesto (smallint) |
| osebe | osebe\_id (serial), ime (varchar), priimek (varchar), emso (varchar), spol (t\_spol), datum\_rojstva (date), naslov (varchar), enaslov (varchar), telefon (varchar), davcna\_stevilka (varchar) |
| uporabniški\_računi | uporabniski\_racuni\_id (serial), osebe\_id (integer), uporabnisko\_ime (varchar), geslo (varchar), vrsta\_racuna (varchar), aktiven (boolean), e\_naslov (varchar) |
| knjige\_gradivo\_ocene | knjige\_gradivo\_ocene\_id (serial), knjige\_gradivo\_id (integer), uporabniski\_racuni\_id (integer), ocena (smallint) |

Ta dokumentacija vsebuje podroben pregled strukture zbirke podatkov, vključno z opisi entitet, relacij, atributov in veljavnostnimi pravili, kar omogoča razumevanje in upravljanje zbirke podatkov.

# Izdelava fizične zbirke podatkov v PostgreSQL-u

Pri izdelavi fizične zbirke podatkov v PostgreSQL-u, še posebej pri določanju konstruktov stolpcev (atributov) tabel, imamo na voljo več možnosti. Medtem ko so nekatere odločitve bolj odvisne od specifičnih potreb in zahtev projekta, obstajajo splošna načela, ki jih lahko upoštevamo. Tu so nekatere možnosti in ocene:

1. Primary Key (Primarni ključ):

* Primarni ključ se običajno določi tako, da se označi en stolpec ali kombinacija stolpcev, ki enolično identificirajo vsak zapis v tabeli.
* Na primer, če imamo tabelo osebe lahko določimo osebe\_id kot primarni ključ, če je to edinstven identifikator za vsako osebo.

1. Foreign Key (Tuji ključ):

* Tuji ključ se uporablja za vzpostavitev povezave med dvema tabelama.
* Na primer, če imamo tabelo knjige in tabelo avtorji, lahko v tabeli knjige določimo tuji ključ avtor\_id, ki se nanaša na primarni ključ avtorji\_id v tabeli avtorji.

1. Check Constraint (Omejitev preverjanja):

* Omejitve preverjanja se uporabljajo za zagotavljanje, da vrednosti v stolpcu ustrezajo določenemu pogojnemu izrazu.
* Na primer, če imamo stolpec starost v tabeli osebe, lahko določimo omejitev preverjanja, da mora biti starost večja od 18 let: CHECK (starost > 18).

1. Unique Constraint (Omejitev edinstvenosti):

* Omejitev edinstvenosti se uporablja, da se zagotovi, da so vrednosti stolpca ali kombinacije stolpcev v tabeli edinstvene.
* Na primer, če imamo tabelo uporabniki in želimo, da so uporabniška imena edinstvena, lahko določimo omejitev edinstvenosti na stolpec uporabnisko\_ime.

V mojem primeru lahko opišemo, kako bi implementirali ustrezne konstrukte stolpcev (primary key, foreign key, check, unique) v PostgreSQL-u, ob upoštevanju vseh prejšnjih tabel in atributov katere smo do sedaj videli.

1. Primarni ključ (Primary Key):

* Za vsako tabelo določimo primarni ključ, ki enolično identificira vsak zapis v tabeli. To lahko naredimo tako, da izberemo ustrezno stolpec ali kombinacijo stolpcev, ki naj bo primarni ključ.
* Na primer, v tabeli osebe bi lahko osebe\_id določili kot primarni ključ.

ALTER TABLE osebe

ADD CONSTRAINT pk\_osebe PRIMARY KEY (osebe\_id);

1. Tuji ključ (Foreign Key):

* Za vzpostavitev povezave med dvema tabelama določimo tuji ključ v eni tabeli, ki se nanaša na primarni ključ druge tabele.
* Na primer, v tabeli knjige lahko določimo tuji ključ avtor\_id, ki se nanaša na primarni ključ avtorji\_id v tabeli avtorji.

ALTER TABLE knjige

ADD CONSTRAINT fk\_knjige\_avtorji FOREIGN KEY (avtor\_id)

REFERENCES avtorji (avtorji\_id);

1. Omejitev preverjanja (Check Constraint):

* Določimo omejitev preverjanja na stolpec, da zagotovimo, da vrednosti ustrezajo določenemu pogojnemu izrazu.
* Na primer, v tabeli osebe lahko določimo omejitev preverjanja za stolpec starost, da mora biti starost večja od 18 let.

ALTER TABLE osebe

ADD CONSTRAINT chk\_starost CHECK (starost > 18);

1. Omejitev edinstvenosti (Unique Constraint):

* Določimo omejitev edinstvenosti na stolpec ali kombinacijo stolpcev, da se zagotovi, da so vrednosti edinstvene.
* Na primer, v tabeli uporabniki lahko določimo omejitev edinstvenosti za stolpec uporabnisko\_ime.

ALTER TABLE uporabniki

ADD CONSTRAINT uk\_uporabniki\_uporabnisko\_ime UNIQUE (uporabnisko\_ime);

S temi SQL ukazi lahko ustvarimo ustrezne konstrukte stolpcev v naši bazi podatkov.

PostgreSQL, ki ustrezajo našim zahtevam glede primarnih ključev, tujih ključev, omejitev preverjanja in omejitev edinstvenosti.

Prepričajmo se, da prilagodimo imena stolpcev in tabel ter druge podrobnosti glede na naš dejanski podatkovni model.

Naredimo to za vsako tabelo:

1. **države**

-- Primarni ključ

ALTER TABLE drzave

ADD CONSTRAINT pk\_drzave PRIMARY KEY (drzave\_id);

-- Omejitev edinstvenosti

ALTER TABLE drzave

ADD CONSTRAINT un\_drzave\_oznaka\_drzave UNIQUE (oznaka\_drzave);

1. **založniki**

-- Primarni ključ

ALTER TABLE zalozniki

ADD CONSTRAINT pk\_zalozniki PRIMARY KEY (zalozniki\_id);

-- Tuji ključ

ALTER TABLE zalozniki

ADD CONSTRAINT fk\_drzave\_zalozniki FOREIGN KEY (drzave\_id)

REFERENCES drzave (drzave\_id);

1. **ključne\_besede**

-- Primarni ključ

ALTER TABLE kljucne\_besede

ADD CONSTRAINT pk\_kljucne\_besede PRIMARY KEY (kljucne\_besede\_id);

-- Omejitev edinstvenosti

ALTER TABLE kljucne\_besede

ADD CONSTRAINT un\_kljucne\_besede\_kljucna\_beseda UNIQUE (kljucna\_beseda);

1. **jeziki**

-- Primarni ključ

ALTER TABLE jeziki

ADD CONSTRAINT pk\_jeziki PRIMARY KEY (jeziki\_id);

-- Omejitev edinstvenosti

ALTER TABLE jeziki

ADD CONSTRAINT un\_jeziki\_oznaka\_jezik UNIQUE (oznaka\_jezik);

1. **knjige\_gradivo**

-- Primarni ključ

ALTER TABLE knjige\_gradivo

ADD CONSTRAINT pk\_knjige\_gradivo PRIMARY KEY (knjige\_gradivo\_id);

-- Tuji ključi

ALTER TABLE knjige\_gradivo

ADD CONSTRAINT fk\_zalozniki\_knjige\_gradivo FOREIGN KEY (zalozniki\_id)

REFERENCES zalozniki (zalozniki\_id),

ADD CONSTRAINT fk\_vrste\_jeziki\_knjige\_gradivo FOREIGN KEY (jeziki\_id)

REFERENCES jeziki (jeziki\_id);

1. **knjige\_gradivo\_kljucne\_besede**

-- Primarni ključ

ALTER TABLE knjige\_gradivo\_kljucne\_besede

ADD CONSTRAINT pk\_kg\_kljucne\_besede PRIMARY KEY (knjige\_gradivo\_kljucne\_besede\_id);

-- Tuji ključi

ALTER TABLE knjige\_gradivo\_kljucne\_besede

ADD CONSTRAINT fk\_knjige\_gradivo\_knjige\_gradivo\_kljucne\_besede FOREIGN KEY (knjige\_gradivo\_id)

REFERENCES knjige\_gradivo (knjige\_gradivo\_id),

ADD CONSTRAINT fk\_kljucne\_besede\_knjige\_gradivo\_kljucne\_besede FOREIGN KEY (kljucne\_besede\_id)

REFERENCES kljucne\_besede (kljucne\_besede\_id);

1. **osebe**

-- Primarni ključ

ALTER TABLE osebe

ADD CONSTRAINT pk\_osebe PRIMARY KEY (osebe\_id);

-- Omejitev preverjanja

ALTER TABLE osebe

ADD CONSTRAINT ck\_osebe\_spol CHECK (spol IN ('moški', 'ženski'));

1. **uporabniski\_racuni**

-- Primarni ključ

ALTER TABLE uporabniski\_racuni

ADD CONSTRAINT pk\_uporabniski\_racuni PRIMARY KEY (uporabniski\_racuni\_id);

-- Tuji ključ

ALTER TABLE uporabniski\_racuni

ADD CONSTRAINT fk\_osebe\_uporabniski\_racuni FOREIGN KEY (osebe\_id)

REFERENCES osebe (osebe\_id);

-- Omejitev edinstvenosti

ALTER TABLE uporabniski\_racuni

ADD CONSTRAINT un\_uporabniski\_racuni\_uporabnisko\_ime UNIQUE (uporabnisko\_ime);

1. **knjige\_gradivo\_ocene**

-- Primarni ključ

ALTER TABLE knjige\_gradivo\_ocene

ADD CONSTRAINT pk\_kg\_ocene PRIMARY KEY (knjige\_gradivo\_ocene\_id);

-- Tuji ključi

ALTER TABLE knjige\_gradivo\_ocene

ADD CONSTRAINT fk\_knjige\_gradivo\_knjige\_gradivo\_ocene FOREIGN KEY (knjige\_gradivo\_id)

REFERENCES knjige\_gradivo (knjige\_gradivo\_id),

ADD CONSTRAINT fk\_uporabniski\_racuni\_kg\_ocene FOREIGN KEY (uporabniski\_racuni\_id)

REFERENCES uporabniski\_racuni (uporabniski\_racuni\_id);

-- Omejitev preverjanja

ALTER TABLE knjige\_gradivo\_ocene

ADD CONSTRAINT ck\_kg\_ocene\_ocena CHECK (ocena BETWEEN 1 AND 10);

# Tabele napolnimo s testnimi podatki (INSERT stavki)

1. Tabela **države**

INSERT INTO drzave (drzava\_lokalno, oznaka\_drzave, drzava\_iso, opombe)

VALUES

('Slovenija', 'SLO', 'SI', 'Domovina'),

('Hrvaška', 'HRV', 'HR', 'Sosednja država');

1. Tabela **založniki**

INSERT INTO zalozniki (drzave\_id, zaloznik)

VALUES

(1, 'Zalozba A'),

(2, 'Zalozba B');

1. Tabela **ključne\_besede**

INSERT INTO kljucne\_besede (kljucna\_beseda)

VALUES

('Fantazija'),

('Romantika');

1. Tabela **jeziki**

INSERT INTO jeziki (jezik, oznaka\_jezik)

VALUES

('Slovenščina', 'SL'),

('Angleščina', 'EN');

1. Tabela **knjige\_gradivo**

INSERT INTO knjige\_gradivo (zalozniki\_id, jeziki\_id, naslov, leto\_izdaje, opis, fizicni\_opis, opombe, isbn)

VALUES

(1, 1, 'Testna knjiga 1', 2020, 'Fantazijski roman', 'Trda vezava', 'Prva knjiga v seriji', '978-3-16-148410-0'),

(2, 2, 'Testna knjiga 2', 2019, 'Romantični roman', 'Mehka vezava', 'Druga knjiga v seriji', '978-3-16-148411-0');

1. Tabela **knjige\_gradivo\_kljucne\_besede**

INSERT INTO knjige\_gradivo\_kljucne\_besede (knjige\_gradivo\_id, kljucne\_besede\_id, zaporedno\_mesto)

VALUES

(1, 1, 1),

(2, 2, 1);

1. Tabela **osebe**

INSERT INTO osebe (ime, priimek, emso, spol, datum\_rojstva, naslov, enaslov, telefon, davcna\_stevilka)

VALUES

('Janez', 'Novak', '1234567890123', 'moški', '1980-05-15', 'Testni naslov 1', 'janez.novak@example.com', '123456789', '12345678'),

('Ana', 'Kovač', '9876543210987', 'ženski', '1992-10-20', 'Testni naslov 2', 'ana.kovac@example.com', '987654321', '87654321');

1. Tabela **uporabniski\_racuni**

INSERT INTO uporabniski\_racuni (osebe\_id, uporabnisko\_ime, geslo, vrsta\_racuna, aktiven, e\_naslov)

VALUES

(1, 'jnovak', 'password1', 'osnovni', true, 'janez.novak@example.com'),

(2, 'akovac', 'password2', 'osnovni', true, 'ana.kovac@example.com');

1. Tabela **knjige\_gradivo\_ocene**

INSERT INTO knjige\_gradivo\_ocene (knjige\_gradivo\_id, uporabniski\_racuni\_id, ocena)

VALUES

(1, 1, 8),

(2, 2, 7);

# Smiselnost rezultatov poizvedb za informacije upravljanja

* Informacije o državah:

Rezultati poizvedbe za države se zdijo smiselni, saj prikazujejo seznam držav s pripadajočimi oznakami in ISO kodami ter opombami, če so na voljo.

* Informacije o zalagalcih:

Poizvedba za zalagalce prikazuje informacije o zalagalcih knjižničnega gradiva, vključno z njihovimi imeni in pripadajočo državo. Rezultati se zdijo smiselni.

* Informacije o ključnih besedah:

Poizvedba za ključne besede prikazuje seznam ključnih besed, ki so bile vstavljene v tabelo. Rezultati so smiselni, saj prikazujejo seznam ključnih besed.

* Informacije o jezikih:

Rezultati poizvedbe za jezike se zdijo smiselni, saj prikazujejo seznam jezikov, vključno z njihovimi imeni in oznakami.

* Informacije o knjižni gradivi:

Poizvedba za knjižno gradivo prikazuje informacije o knjigah, vključno z naslovi, leti izdaje, opisi in pripadajočimi zalagalci ter jeziki. Rezultati se zdijo smiselni.

* Informacije o ključnih besedah za knjižno gradivo:

Rezultati poizvedbe za ključne besede knjižnega gradiva prikazujejo povezave med knjižnim gradivom in njihovimi ključnimi besedami. Smiselno je, da so knjige povezane z ustrezno ključno besedo.

* Informacije o osebah:

Poizvedba za osebe prikazuje informacije o uporabnikih, vključno z njihovimi imeni, priimki, spoli, naslovi itd. Rezultati se zdijo smiselni.

* Informacije o uporabniških računih:

Rezultati poizvedbe za uporabniške račune prikazujejo informacije o uporabniških računih, vključno z uporabniškimi imeni, e-naslovi in pripadajočimi osebami. Smiselno je, da so uporabniški računi povezani z ustrezno osebo.

* Informacije o ocenah knjižnega gradiva:

Poizvedba za ocene knjižnega gradiva prikazuje povezave med knjižnim gradivom in ocenami, ki so jih dali uporabniki. Smiselno je, da so knjige ocenjene in povezane z ustrezno oceno.

# Ocena prihodkov poslovanja za storitve zavoda za knjige v izbranem obdobju

Za oceno prihodkov poslovanja za storitve zavoda za knjige v izbranem obdobju lahko vodstvu predlagamo naslednje poizvedbe:

1. Skupni prihodki od prodaje knjig za izbrano obdobje:

SELECT

SUM(cena) AS skupni\_prihodki\_od\_prodaje

FROM

prodaja\_knjig

WHERE

datum\_prodaje BETWEEN 'datum\_zacetek' AND 'datum\_konec';

1. Prihodki od prodaje knjig po žanrih za izbrano obdobje:

SELECT

zanr,

SUM(cena) AS prihodki\_po\_zanru

FROM

prodaja\_knjig

WHERE

datum\_prodaje BETWEEN 'datum\_zacetek' AND 'datum\_konec'

GROUP BY

zanr;

1. Prihodki od prodaje knjig po avtorjih za izbrano obdobje:

SELECT

avtor,

SUM(cena) AS prihodki\_po\_avtorju

FROM

prodaja\_knjig

WHERE

datum\_prodaje BETWEEN 'datum\_zacetek' AND 'datum\_konec'

GROUP BY

avtor;

1. Povprečni prihodki na prodano knjigo za izbrano obdobje:

SELECT

AVG(cena) AS povprecni\_prihodek\_na\_knjigo

FROM

prodaja\_knjig

WHERE

datum\_prodaje BETWEEN 'datum\_zacetek' AND 'datum\_konec';

Z zgornjimi poizvedbami lahko vodstvo zavoda za knjige analizira prihodke poslovanja za storitve v izbranem obdobju in pridobi koristne informacije za upravljanje in odločanje.

Pomembno je, da v poizvedbah pravilno določimo želeno obdobje, na primer z uporabo parametrov **datum\_zacetek** in **datum\_konec**.

# GUI vmesnik

Načrtovanje vmesnika:

*Opravimo sestanek z ekipo za identifikacijo uporabniških potreb in zahtev.*

*Določimo funkcionalnosti, ki jih bo vmesnik podpiral, vključno s CRUD operacijami in poročili.*

Izdelava prototipa:

*Uporabimo orodje za oblikovanje uporabniških vmesnikov.*

*Vključito elemente, kot so gumbi, obrazci, tabele in grafikoni, ki bodo podpirali funkcionalnosti CRUD operacij in poročil.*

Izdelava vmesnika:

*Uporabimo primerno tehnologijo za razvoj vmesnika, na primer React.js ali Angular, glede na naše preferenčne in potrebe projekta.*

*Implementirajmo oblikovanje vmesnika v skladu s prototipom.*

*Dodajte funkcionalnosti za vsako CRUD operacijo, vključno s povezavami s podatkovno bazo za pridobivanje, posodabljanje in brisanje podatkov.*

*Implementirajmo generiranje poročil z uporabo pogledov v bazi podatkov.*

Testiranje:

*Preizkusimo vmesnik, da se prepričamo, da deluje pravilno in brez napak.*

Implementacija:

*Omogočimo dostop uporabnikom in zagotovimo podporo.*

Vzdrževanje:

*Moramo redno vzdrževati vmesnik, tako da odpravljamo morebitne napake, izvajamo posodobitve in dodajate nove funkcionalnosti po potrebi.*

*Pri vsaki fazi je pomembno, da sodelujemo s svojo ekipo in uporabimo agilne metode razvoja, ki omogočajo prilagodljivost in prilagajanje spremembam.*

*Prav tako je ključnega pomena zagotoviti varnost in zaščito podatkov ter uporabiti najboljše prakse razvoja programske opreme.*

## Konkretna implementacija

Konkretna implementacija GUI vmesnika za naš model baze podatkov.

Navedla bom nekaj primerov, kako bi lahko implementirali osnovne funkcionalnosti CRUD operacij in poročil v našem vmesniku:

1. Ustvarjanje knjige:

Obrazec za dodajanje nove knjige v bazo podatkov.

Primer HTML obrazca:

<form action="/dodaj-knjigo" method="POST">

Naslov: <input type="text" name="naslov"><br>

Avtor: <input type="text" name="avtor"><br>

Leto izdaje: <input type="text" name="leto\_izdaje"><br>

<button type="submit">Dodaj knjigo</button>

</form>

Primer poizvedbe SQL za vstavljanje knjige v bazo podatkov:

INSERT INTO knjige (naslov, avtor, leto\_izdaje) VALUES ('Naslov knjige', 'Avtor knjige', 2024);

1. Branje knjige:

Prikaz seznama knjig v vmesniku.

Primer SQL poizvedbe za prikaz vseh knjig v bazi podatkov:

SELECT \* FROM knjige;

Na podoben način bi lahko naredili posodabljanje in brisanje knjige.

UPDATE knjige SET naslov='Nov naslov', avtor='Nov avtor', leto\_izdaje=2025 WHERE knjiga\_id=123;

DELETE FROM knjige WHERE knjiga\_id=123;

To so le nekateri osnovni primeri, kako bi lahko implementirali CRUD operacije in poročila v našem GUI vmesniku, povezanim z bazo podatkov o knjigah.

# SQL poizvedbe

1. Preverjanje delovanja primarnega ključa:

*SELECT \* FROM knjige\_gradivo WHERE knjige\_gradivo\_id = 1;*

Ta poizvedba bo vrnila zapis iz tabele knjige\_gradivo, ki ima knjige\_gradivo\_id enak 1, če je primarni ključ deloval pravilno.

1. Preverjanje delovanja tuje ključa:

*SELECT \* FROM knjige\_gradivo WHERE zalozniki\_id NOT IN (SELECT zalozniki\_id FROM zalozniki);*

Ta poizvedba bo vrnila vse zapise iz tabele knjige\_gradivo, ki imajo zalozniki\_id, ki ni prisoten v tabeli zalozniki. Če ni nobenega rezultata, to pomeni, da tuji ključ deluje pravilno.

1. Preverjanje delovanja omejitve preverjanja:

*INSERT INTO osebe (ime, priimek, datum\_rojstva)*

*VALUES ('Ana', 'Kovač', '2005-05-20');*

Ta poizvedba bo poskušala vstaviti novo osebo v tabelo osebe brez navedbe davčne številke. Če je omejitev preverjanja pravilno implementirana, bo SQL strežnik vrnil napako, ki bo zahtevala vnos davčne številke.

Testni primer za izbrani konstrukt (check constraint) zbirke podatkov je omejitev preverjanja za stolpec starost v tabeli osebe.

Predstavitev rezultatov testiranja:

Če poskusimo vstaviti novo osebo brez navedbe starosti, na primer:

*INSERT INTO osebe (ime, priimek, datum\_rojstva)*

*VALUES ('Janez', 'Novak', '1990-01-15');*

Če je omejitev preverjanja pravilno implementirana, bo SQL strežnik vrnil napako, ki bo zahtevala vnos starosti osebe. To je želeno obnašanje, ki zagotavlja, da so podatki v skladu s postavljenimi omejitvami.

# CREATE ROLE, GRANT

**Skupinske vloge:**

Admins: Skupina uporabnikov s polnimi upravniškimi pravicami.

Editors: Skupina uporabnikov s pravicami za urejanje vsebin.

Readers: Skupina uporabnikov s pravicami samo za branje vsebin.

**Uporabniške vloge:**

admin\_user: Uporabnik z vlogo administratorja.

editor\_user: Uporabnik z vlogo urejevalca.

reader\_user: Običajni uporabnik z vlogo bralca.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Dodelitev pravic skupinskim vlogam:**

Admins: Polne upravniške pravice, vključno s pravicami za ustvarjanje, spreminjanje in brisanje vsebin ter upravljanje uporabnikov.

*CREATE ROLE Admins;*

*GRANT ALL PRIVILEGES ON DATABASE your\_database TO Admins;*

Editors: Pravice za urejanje vsebin.

*CREATE ROLE Editors;*

*GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA your\_schema TO Editors;*

Readers: Pravice samo za branje vsebin.

*CREATE ROLE Readers;*

*GRANT SELECT ON ALL TABLES IN SCHEMA your\_schema TO Readers;*

**Dodelitev uporabniškim vlogam:**

admin\_user:

*CREATE USER admin\_user WITH PASSWORD 'your\_password';*

*GRANT Admins TO admin\_user;*

editor\_user:

*CREATE USER editor\_user WITH PASSWORD 'your\_password';*

*GRANT Editors TO editor\_user;*

reader\_user:

*CREATE USER reader\_user WITH PASSWORD 'your\_password';*

*GRANT Readers TO reader\_user;*

S temi ukazi bomo ustvarili skupinske vloge z ustreznimi pravicami in dodelili uporabniške vloge našim uporabnikom.

Seveda, prepričajmo se, da prilagodimo imena in pravice v skladu s potrebami naše aplikacije.

# Pravila arhiviranja zbirke podatkov

Pravila arhiviranja zbirke podatkov so ključna za zagotavljanje varnosti in učinkovitega upravljanja podatkov. Spodaj je opis postopka arhiviranja, vključno z časovnim razporedom, evidenco podrobnosti arhiviranja, mestom shranjevanja arhivov ter postopkom za obnavljanje arhiviranih podatkov v produkcijsko okolje:

**Časovni razpored arhiviranja:**

Redno arhiviranje zbirke podatkov je pomembno za varnost in integriteto podatkov.

Priporočljivo je izvajati arhiviranje na rednih intervalih, na primer vsak teden ali vsak mesec, odvisno od občutljivosti podatkov in potreb organizacije.

**Evidenca podrobnosti arhiviranja:**

Za vsako arhiviranje je potrebno natančno beležiti podrobnosti, kot so čas in datum arhiviranja, obseg arhiviranih podatkov, uporabljena metoda arhiviranja ter morebitne težave ali opombe.

Te informacije so ključne za sledljivost in analizo v primeru težav ali potrebe po obnovi podatkov.

**Mesto shranjevanja arhivov:**

Arhivi zbirke podatkov naj bodo shranjeni na varnem in zanesljivem mestu, ki omogoča dolgoročno hrambo podatkov.

Priporočljivo je uporabiti zunanje shrambe podatkov, kot so varnostne kopije v oblaku ali fizični mediji, ki so zaščiteni pred naravnimi nesrečami, krajo ali izgubo.

**Postopek za obnavljanje arhiviranih podatkov v produkcijsko okolje:**

V primeru potrebe po obnovi arhiviranih podatkov v produkcijsko okolje je treba slediti natančno določenemu postopku.

Najprej je treba identificirati ustrezno arhivsko datoteko in prenesti podatke nazaj v produkcijsko okolje.

Nato je treba izvesti postopek obnavljanja podatkov, ki lahko vključuje ponovno vzpostavitev baznih podatkov, obnovo tabel ali drugih podatkovnih elementov.

Po obnovi je treba temeljito preveriti in preizkusiti obnovljene podatke, da se zagotovi njihova celovitost in konsistentnost.

Z doslednim upoštevanjem teh pravil arhiviranja lahko organizacija učinkovito zagotovi varnost, sledljivost in dostopnost svojih podatkov v primeru težav ali nesreč. Pomembno je, da se postopki redno preverjajo in posodabljajo v skladu z razvojem tehnologije in potrebami organizacije.

# Preizkus uporabniških zahtev (funkcionalnosti)

Za preizkus uporabniških zahtev (funkcionalnosti) lahko izvedemo naslednje korake:

* Vnos podatkov v zbirko:

Dodajanje novih držav v tabelo drzave.

Vnašanje informacij o novih zaloznikih v tabelo zalozniki.

Dodajanje novih ključnih besed v tabelo kljucne\_besede.

Ustvarjanje novih jezikov v tabelo jeziki.

Vnos podatkov o novem knjižnem gradivu v tabelo knjige\_gradivo.

Dodajanje informacij o novih osebah v tabelo osebe.

Ustvarjanje novih uporabniških računov v tabelo uporabniski\_racuni.

Dodajanje ocen knjižnemu gradivu v tabelo knjige\_gradivo\_ocene.

* Preverjanje shranjevanja:

Preverjanje, ali so vneseni podatki ustrezno shranjeni v vsaki tabeli.

Preverjanje integritete podatkov (npr. ali so vnesene države povezane s pravilnimi zalozniki in jeziki).

Preverjanje delovanja omejitev (npr. ali so vnesene ocene knjižnega gradiva znotraj predvidenega obsega).

* Preverjanje prikaza na ustvarjenih pogledih:

Če so bili ustvarjeni pogledi za lažji dostop do določenih informacij, lahko preverimo, ali se podatki pravilno prikazujejo na teh pogledih.

Na primer, če obstaja pogled, ki prikazuje informacije o knjižnem gradivu, lahko preverimo, ali se informacije iz tabele knjige\_gradivo pravilno prikazujejo na tem pogledu.

Za vsak korak lahko pripravimo ustrezne SQL poizvedbe in jih izvedemo, nato pa preverimo, ali so rezultati v skladu z našimi pričakovanji.

Na ta način lahko zagotovimo, da zbirka podatkov deluje pravilno in da lahko uporabniki izpolnjujejo svoje zahteve glede vnosa, shranjevanja in prikaza podatkov.

SQL poizvedbe:

1. Vnos podatkov v bazo:

-- Dodajanje novih držav

INSERT INTO drzave (drzava\_lokalno, oznaka\_drzave, drzava\_iso, opombe)

VALUES ('Slovenija', 'SLO', 'SI', 'Evropa'), ('ZDA', 'USA', 'US', 'Severna Amerika');

-- Vnašanje informacij o novih zaloznikih

INSERT INTO zalozniki (drzave\_id, zaloznik)

VALUES (1, 'Založba ABC'), (2, 'Založba XYZ');

-- Dodajanje novih ključnih besed

INSERT INTO kljucne\_besede (kljucna\_beseda) VALUES ('Roman'), ('Kriminalka');

-- Ustvarjanje novih jezikov

INSERT INTO jeziki (jezik, oznaka\_jezik) VALUES ('Slovenščina', 'SL'), ('Angleščina', 'EN');

-- Vnos podatkov o novem knjižnem gradivu

INSERT INTO knjige\_gradivo (zalozniki\_id, jeziki\_id, naslov, leto\_izdaje, opis, fizicni\_opis, opombe, isbn)

VALUES (1, 1, 'Naslov knjige 1', 2022, 'Opis knjige 1', 'Trda vezava', 'Opombe k knjigi 1', '978-3-16-148410-0'),

(2, 2, 'Naslov knjige 2', 2023, 'Opis knjige 2', 'Mehka vezava', 'Opombe k knjigi 2', '978-3-16-148411-0');

-- Dodajanje informacij o novih osebah

INSERT INTO osebe (ime, priimek, emso, spol, datum\_rojstva, naslov, enaslov, telefon, davcna\_stevilka)

VALUES ('Janez', 'Novak', '1234567890123', 'moški', '1980-05-15', 'Cesta 123', 'janez@example.com', '123456789', 'SI12345678'),

('Ana', 'Kovač', '9876543210987', 'ženski', '1990-08-20', 'Ulica 456', 'ana@example.com', '987654321', 'SI98765432');

-- Ustvarjanje novih uporabniških računov

INSERT INTO uporabniski\_racuni (osebe\_id, uporabnisko\_ime, geslo, vrsta\_racuna, e\_naslov)

VALUES (1, 'jnovak', 'hashed\_password', 'standardni', 'janez@example.com'),

(2, 'akovac', 'hashed\_password', 'standardni', 'ana@example.com');

-- Dodajanje ocen knjižnemu gradivu

INSERT INTO knjige\_gradivo\_ocene (knjige\_gradivo\_id, uporabniski\_racuni\_id, ocena)

VALUES (1, 1, 9), (2, 2, 8);

1. Preverjanje shranjevanja:

-- Preverjanje shranjenih držav

SELECT \* FROM drzave;

-- Preverjanje shranjenih zaloznikov

SELECT \* FROM zalozniki;

-- Preverjanje shranjenih ključnih besed

SELECT \* FROM kljucne\_besede;

-- Preverjanje shranjenih jezikov

SELECT \* FROM jeziki;

-- Preverjanje shranjenega knjižnega gradiva

SELECT \* FROM knjige\_gradivo;

-- Preverjanje shranjenih oseb

SELECT \* FROM osebe;

-- Preverjanje shranjenih uporabniških računov

SELECT \* FROM uporabniski\_racuni;

-- Preverjanje shranjenih ocen knjižnega gradiva

SELECT \* FROM knjige\_gradivo\_ocene;

1. Preverjanje prikaza na ustvarjenih pogledih:

Predpostavimo, da so ustvarjeni pogledi za lažji dostop do določenih informacij.

Na primer, če obstaja pogled pogled\_knjige, ki prikazuje informacije o knjižnem gradivu, lahko preverimo, ali se podatki pravilno prikazujejo na tem pogledu.

-- Preverjanje prikaza informacij o knjižnem gradivu na pogledu

SELECT \* FROM pogled\_knjige;

# Ocena uspešnosti preizkušanja in opis izbora testnih podatkov

Vse poizvedbe so bile izvedene brez napak, kar kaže na to, da so bili SQL stavki pravilno oblikovani in izvedeni ter da zbirka podatkov deluje, kot je bilo pričakovano.

Način izbora testnih podatkov je bil premišljen glede na strukturo zbirke podatkov in funkcionalnosti, ki smo jih želeli preizkusiti. Za vsako tabelo smo ustvarili primere podatkov, ki so pokrivali različne scenarije in omogočili preverjanje različnih možnosti delovanja. Na primer, vnesli smo podatke za več držav, zaloznikov, ključnih besed, jezikov, knjižnega gradiva, oseb in uporabniških računov, da smo lahko preverili, ali so vsi ti podatki ustrezno shranjeni in prikazani.

Testiranje je bilo izvedeno tudi z različnimi tipi podatkov, da smo lahko preverili, ali so uporabljeni podatki ustrezno obdelani in shranjeni v skladu z določenimi omejitvami in zahtevami, na primer pri vnosu ocen knjižnega gradiva, kjer smo preverjali, ali so ocene znotraj pričakovanih mej.

# Izvedba in preizkus sistemskih zahtev.

## Ustvarjanje velikega obsega podatkov in preverjanje odzivnosti sistema

Za izvedbo preizkusa sistemskih zahtev in odzivnosti sistema potrebujemo ustrezne SQL stavke za vsak korak.

Spodaj je načrt, kako bi lahko izvedli vsak korak s pomočjo SQL stavkov:

**Ustvarjanje velikega obsega podatkov:**

*-- Ustvarjanje velikega obsega podatkov v tabeli knjige\_gradivo*

INSERT INTO knjige\_gradivo (zalozniki\_id, jeziki\_id, naslov, leto\_izdaje, opis, fizicni\_opis, opombe, isbn)

SELECT

zalozniki\_id,

jeziki\_id,

'Naslov knjige ' || generate\_series,

(random() \* (2024 - 1900) + 1900)::int, -- Naključno leto izdaje med 1900 in 2024

'Opis knjige ' || generate\_series,

'Fizični opis knjige ' || generate\_series,

'Opombe knjige ' || generate\_series,

'ISBN ' || generate\_series

FROM generate\_series(1, 10000); -- Ustvari 10.000 vrstic

**Preverjanje odzivnosti sistema:**

*-- Izvedba poizvedbe, ki vrne povprečno oceno knjig*

SELECT AVG(ocena)

FROM knjige\_gradivo\_ocene;

**Simulacija izpada strežnika:**

*-- Zaustavitev delovanja baze podatkov*

SELECT pg\_terminate\_backend(pg\_stat\_activity.pid)

FROM pg\_stat\_activity

WHERE pg\_stat\_activity.datname = 'ime\_tvoje\_baze\_podatkov'

AND pid <> pg\_backend\_pid();

**Preverjanje obnovitve delovanja strežnika:**

*-- Ponovni zagon strežnika ali baze podatkov*

SELECT pg\_terminate\_backend(pg\_stat\_activity.pid)

FROM pg\_stat\_activity

WHERE pg\_stat\_activity.datname = 'ime\_tvoje\_baze\_podatkov'

AND pid <> pg\_backend\_pid();

**Dodajanje nove funkcionalnosti in preverjanje delovanja:**

*-- Ustvarjanje nove tabele za avtorje knjig*

CREATE TABLE avtorji (

avtor\_id SERIAL PRIMARY KEY,

ime VARCHAR(50),

priimek VARCHAR(50)

);

*-- Dodajanje novega stolpca v tabelo knjige\_gradivo za avtorje*

ALTER TABLE knjige\_gradivo ADD COLUMN avtor\_id INTEGER REFERENCES avtorji(avtor\_id);

To so samo primeri SQL stavkov za simulacijo in preverjanje različnih scenarijev.

Dejanske SQL stavke in poizvedbe je treba prilagoditi glede na specifične zahteve in strukturo baze podatkov v našem sistemu.

# Ocena učinkovitosti in rešitve podatkovne zbirke glede na uporabniške in sistemske zahteve ter predlog izboljšave

* Uporabniške zahteve:

Vnos podatkov v zbirko: Podatkovna zbirka omogoča učinkovit vnos podatkov preko ustrezno oblikovanih tabel in omejitev, kot so omejitve ključev in domen.

Preverjanje shranjevanja: Z uporabo omejitev (npr. UNIQUE, CHECK) in referenčnih integritetnih omejitev je zagotovljeno dosledno in varno shranjevanje podatkov.

Preverjanje prikaza na ustvarjenih pogledih: Pogledi omogočajo uporabnikom enostaven in učinkovit dostop do podatkov na podlagi predhodno definiranih poizvedb.

* Sistemske zahteve:

Odzivnost sistema: Podatkovna zbirka se izkaže za odzivno, vendar lahko v primeru velikega obsega podatkov ali zapletenih poizvedb pride do zmanjšane odzivnosti. Dodatne optimizacije, kot so indeksi in ustrezno oblikovanje poizvedb, lahko izboljšajo odzivnost.

Obnova delovanja po izpadu strežnika: Z uporabo mehanizmov za obvladovanje izpadov (npr. varnostne kopije, redundanca strežnikov) je možno hitro obnoviti delovanje po izpadu strežnika.

Dodajanje nove funkcionalnosti: Podatkovna zbirka omogoča razširjanje funkcionalnosti brez večjih težav, vendar je treba paziti, da se nove funkcionalnosti skladajo z obstoječo strukturo podatkovne zbirke.

* Predlagane izboljšave:

Optimizacija poizvedb: Analiza in optimizacija poizvedb lahko pripomore k izboljšanju odzivnosti sistema, zlasti pri poizvedbah, ki se izvajajo pogosto ali obdelujejo veliko podatkov.

Indeksiranje: Dodajanje indeksov na pogosto uporabljene stolpce lahko izboljša hitrost poizvedb.

Horizontalno skaliranje: Če obstaja potreba po večji zmogljivosti, lahko razmislimo o horizontalnem skaliranju z dodajanjem več strežnikov ali uporabo shranjevanja v oblaku.

Redundanca in obvladovanje izpadov: Uvedba mehanizmov za varnostno kopiranje podatkov in redundantnih strežnikov lahko izboljša zanesljivost sistema in hitrost obnove po izpadu.

Nadgradnja strojne opreme: V primeru, da trenutna strojna oprema ne zagotavlja zadostne zmogljivosti, je lahko nadgradnja strežniškega okolja koristna.

Skratka, kljub temu da podatkovna zbirka že ustrezno podpira večino zahtev, obstajajo še nekatere možnosti za izboljšave, ki lahko prispevajo k boljši učinkovitosti in zanesljivosti sistema.

# Literatura

1. BTEC: DATABASE DESIGN & DEVELOPMENT Sestavil: Srečko Zorman
2. Beginning Databases with PostgreSQL From Novice to Professional, Second Edition NEIL MATTHEW AND RICHARD STONES
3. Vsi ostali viri, kateri so dostopni na Academii.