

DATA.DB.210 - Projektitehtävä, Vaihe 1

Antikvariaattijärjestelmä

Ryhmä 1

Elias Peltonen, Taisto Palo ja Roope Lindroos

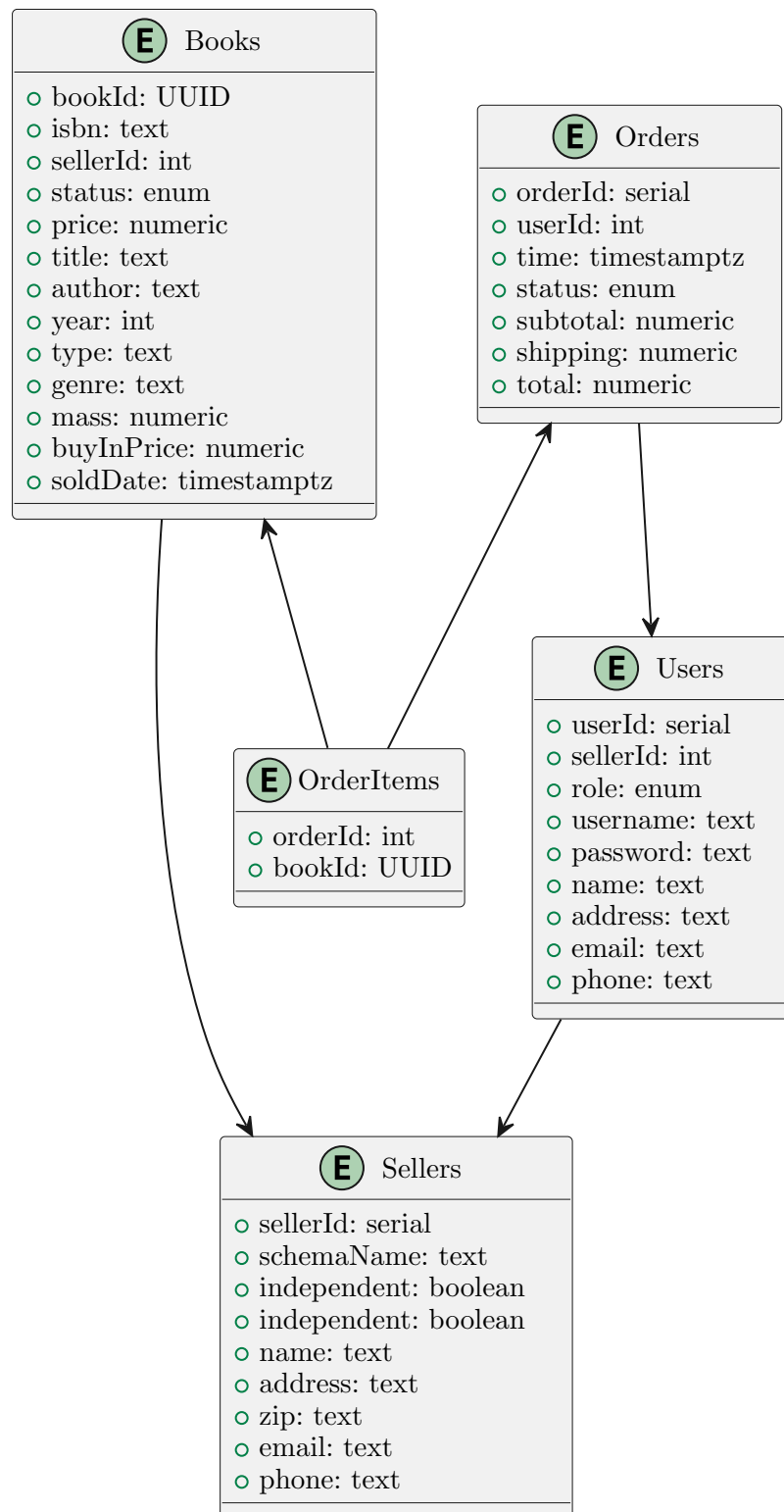
16. tammikuuta 2025

Sisällys


1	UML-kaavio	3
1.1	Keskustietokanta	3
1.2	Yksittäisen divarin tietokanta	4
2	Tietokantakaavio tekstimuodossa	4
3	Näkymät	4
3.1	Johdettavat tiedot	4
3.2	Raporttien näkymät	4
3.2.1	Raportti R1	4
3.2.2	Raportti R2	4
3.2.3	Raportti R3	5
3.2.4	Raportti R4	5
4	Tapahtumakuvaukset	5
5	SQL-luontilauseet	5
6	UML-kaavion muunnos tietokantakaavioksi	7
7	Attribuuttien arvoalueet ja rajoitukset	8
7.1	Sellers-taulu	8
7.2	Books-taulu	8
7.3	Users-taulu	8
7.4	Orders-taulu	8
7.5	OrderItems-taulu	9
7.6	Skeema	9
8	Toteutusvälineet	9
8.1	Palvelin	9
8.2	Käyttöliittymä	9
8.3	Kehitysympäristö	9

1 UML-kaavio

1.1 Keskustietokanta



1.2 Yksittäisen divarin tietokanta

 Books
<ul style="list-style-type: none">○ bookId: UUID○ isbn: text○ sellerId: int○ status: enum○ price: numeric○ title: text○ author: text○ year: int○ type: text○ genre: text○ mass: numeric○ buyInPrice: numeric○ soldDate: timestamptz

2 Tietokantakaavio tekstimuodossa

central.Sellers(sellerId, schemaName, independent, name, address, zip, city, email, phone)

central.Books(bookId, isbn, sellerId, status, price, title, author, year, type, genre, mass, buyInPrice, soldDate)

central.Users(userId, role, sellerId, username, password, name, address, zip, city, email, phone)

central.Orders(orderId, userId, time, status, subtotal, shipping, total)

central.OrderItems(orderId, bookId)

D1.Books(bookId, isbn, price, title, author, year, type, genre, mass, buyInPrice, soldDate)

3 Näkymät

3.1 Johdettavat tiedot

- Tilauksen kokonaispaino ja jakaminen painoluokittain
- Tilauksessa olevien niteiden kokonaishinta
- Tilauksen postituksen kustannukset
- Tilauksen kokonaishinta (niteiden ja postituksen yhteishinta)

3.2 Raporttien näkymät

3.2.1 Raportti R1

Haku kohdistuu suoraan Books-tauluun, joten haussa käytettyä tietoa ei tarvitse varsinaisesti johtaa.

3.2.2 Raportti R2

Raportti voidaan esittää tietokantakaaviona seuraavasti:

R2(genre, totalSoldPrice, averagePrice)

Genre (luokka) saadaan suoraan Books-taulusta. totalSoldPrice johdetaan luokkakohtaisesti kaikkien saman luokan niteille laskemalla näiden kokonaismyyntihinta summaamalla kaikkien luokan niteiden attribuutti price. averagePrice voidaan johtaa totalSoldPrice attribuutin perusteella jakamalla se luokassa olevien niteiden kokonaismäärällä.

3.2.3 Raportti R3

Raportti voidaan esittää tietokantakaaviona seuraavasti:

R3(usersName, boughtBooksCount)

Käyttäjänimet saadaan suoraan Users-taulusta. Teemme haun Order-tauluun ryhmitellen käyttäjänimen perusteella ja rajaten hakutulokset viime vuoden ajalta. Viime vuosi voidaan johtaa nykyisestä vuodesta vähentämällä siitä yksi (ei kiinnitetä). Asiakkaan ostamat niteet saadaan Order-taulusta ja lopuksi kaikkien niteiden summa voidaan laskea asiakaskohtaisesti. Hakua voidaan optimoida myöhemmässä vaiheessa eli tässä demonstroidaan esimerkillä, kuinka tulokset on mahdollista johtaa nykyisestä tietokannasta.

3.2.4 Raportti R4

Raportti R4 on jatkojalostus raporttiin R1, jolloin vakiojärjestys toteutetaan relevanssin perusteella. Tämä voidaan toteuttaa esimerkiksi vektorihaulla tai muulla vastaavalla menetelmällä.

4 Tapahtumakuvaukset

TODO...

5 SQL-luontilauseet

```
-- Enable the UUID extension for globally unique IDs
CREATE EXTENSION IF NOT EXISTS "uuid-oss";

-- Initialize the Central Database
CREATE SCHEMA central;

-- Create enums
CREATE TYPE bookStatus AS ENUM('available', 'reserved', 'sold');

CREATE TYPE userRole AS ENUM('admin', 'seller', 'customer');

CREATE TYPE orderStatus AS ENUM('pending', 'completed');

-- Central Database Tables
CREATE TABLE central.Sellers (
  sellerId SERIAL PRIMARY KEY,
  schemaName TEXT NOT NULL,
  independent BOOLEAN NOT NULL DEFAULT FALSE,
  name TEXT NOT NULL,
```

```
    address TEXT,
    zip TEXT,
    city TEXT,
    email TEXT UNIQUE,
    phone TEXT
);

CREATE TABLE central.Books (
    bookId UUID DEFAULT uuid_generate_v4 () PRIMARY KEY,
    isbn TEXT NOT NULL,
    sellerId INT REFERENCES central.Sellers (sellerId),
    status bookStatus NOT NULL DEFAULT 'available',
    price NUMERIC NOT NULL,
    title TEXT NOT NULL,
    author TEXT NOT NULL,
    year INT,
    type TEXT,
    genre TEXT,
    mass NUMERIC NOT NULL,
    buyInPrice NUMERIC DEFAULT 0,
    soldDate TIMESTAMPTZ
);

CREATE TABLE central.Users (
    userId SERIAL PRIMARY KEY,
    role userRole NOT NULL,
    sellerId INT REFERENCES central.Sellers (sellerId),
    username TEXT UNIQUE NOT NULL,
    password TEXT NOT NULL, -- Must be hashed
    name TEXT NOT NULL,
    address TEXT,
    zip TEXT,
    city TEXT,
    email TEXT UNIQUE,
    phone TEXT
);

CREATE TABLE central.Orders (
    orderId SERIAL PRIMARY KEY,
    userId INT NOT NULL REFERENCES central.Users (userId),
    time TIMESTAMPTZ DEFAULT CURRENT_TIMESTAMP,
    status orderStatus NOT NULL,
    subtotal NUMERIC,
    shipping NUMERIC,
    total NUMERIC
);

CREATE TABLE central.OrderItems (
```

```
    orderId INT NOT NULL REFERENCES central.Orders (orderId),
    bookId UUID NOT NULL REFERENCES central.Books (bookId),
    PRIMARY KEY (orderId, bookId)
);

-- Example of creating a seller schema and its tables
CREATE SCHEMA D1;

CREATE TABLE D1.Books (
    bookId UUID DEFAULT uuid_generate_v4 () PRIMARY KEY,
    isbn TEXT,
    price NUMERIC,
    title TEXT NOT NULL,
    author TEXT NOT NULL,
    year INT,
    type TEXT,
    genre TEXT,
    mass NUMERIC,
    buyInPrice NUMERIC,
    soldDate TIMESTAMPTZ
);
```

6 UML-kaavion muunnos tietokantakaavioksi

UML-kaavion perusteella luotiin tietokantaskeema keskustietokantaa varten (central).

Sellers-entiteetistä otettiin central.Scheman pääavaimeksi attribuutti bookId ja vierasavaimeksi attribuutti schemaName. Loput attribuutit lisättiin mukaan. Sellersillä (myyjä) tarkoitetaan yksittäistä keskustietokantaan kuuluvaa divaria.

Books-entiteetin perusteella luotiin central.Books, jonka pääavaimeksi tuli bookId ja vierasavaimeksi sellerId. Loput attribuutit lisättiin muuttamattomina mukaan. Books-aulussa jokainen rivi on yksittäinen nide, jonka yksilöi bookId (UUID). ISBN-attribuutin avulla saadaan rajattua kaikki yksittäisen teoksen niteet. Tällä rajauksella saamme laskettua teosten lukumäärät samalla säilyttäen yksittäisten niteiden yksilöinnin. Esimerkiksi samaa teosta voi olla usealla divarilla myynnissä ja hinnoissa voi olla eroavaisuuksia.

Users-entiteetin perusteella luotiin central.Users, jonka pääavaimeksi valittiin userId ja vierasavaimeksi sinne valittiin sellerId. Loput attribuutit lisättiin kaavioon muuttumattomina perään. Users-taulu sisältää pääkäyttäjät, myyjät sekä asiakkaat. Käyttäjän rooli määritetään enum-tyypin attribuutilla. Käyttäjällä voi olla sellerId, mikäli käyttäjän rooli on "seller".

Orders-entiteetin perusteella luotiin central.Orders, jonka pääavaimeksi valittiin orderId ja vierasavaimeksi userId. Loput entiteetin attribuutit lisättiin kaavioon mukaan. Tilaus-tauluun tallennetaan kokonaistilauksen summa kirjanpitoa ajatellen (attribuutti total) sekä pelkkien kirjojen osuun kokonaistilauksesta (attribuutti subtotal).

OrderItems-entiteetin perusteella lisättiin relaatio kirjojen ja tilausten välille, jotta tilaukseen liittyvät niteet voidaan tallentaa tietokantaan. Täten esimerkiksi voidaan tilausnumeron perusteella voidaan hakea kaikki siihen liittyvät niteet.

Central-skeeman lisäksi luotiin skeema D1 divaria varten, johon tallennetaan kyseisen divarin niteet. D1-skeemaan ei tarvitse tallentaa muuta tietoa, sillä keskusdivari huolehtii asikastietojen ja tilausten tallentamisesta.

7 Attribuuttien arvoalueet ja rajoitukset

7.1 Sellers-taulu

Myyjä-taulun pääavain on serial-tyyppinen ja myyjällä on myös tekstimuotoinen skeeman nimi, joka ei saa olla tyhjä. Mikäli myyjällä ei ole omaa tietokantaa (käyttää keskustietokantaa), skeeman nimi on "central". Attribuutti "independent" on arvoltaan "tosi", kun divarilla on oma erillinen tietokanta. Myyjän nimi ei saa olla tyhjä. Sähköpostien tulee olla uniikkeja. Muut osoite-/yhteystiedot eivät ole pakollisia ja ne ovat tekstimuotoisia.

7.2 Books-taulu

Kirja-taulun pääavain on UUID-tyyppinen (bookId), jotta niteiden tunnukset säilyvät yksilöivinä eri tietokantojen välillä. ISBN on pakollinen tekstimuotoinen attribuutti, joka yksilöi teokset (teoksista voi olla useampi nide). Niteellä on vierasavain sellerId, joka liittyy niteen johonkin myyjään. SellerId:n avulla voidaan hakea niteitä myyjäkohtaisesti. Niteellä on attribuutti "status", joka ei saa olla tyhjä ja on oletuksena "available". Status on enum-tyyppinen ja sisältää arvot "available", "reserved" ja "sold". Niteellä on attribuutti hinta, joka ei saa olla "null" eli ilmaiseksi annettavat kirjat saavat hinnan 0. Niteellä on otsikko ja kirjailija, jotka ovat tekstimuotoisia pakollisia tietoja. Kirjalle voi lisätä myös kokonaisluku- tai tekstimuotoisen julkaisuvuoden, mikäli se on tiedossa. Kirjan kuvausta voidaan tarkentaa lisäämällä tekstimuotoisen tyyppi ja genre. Tyyppi määrittää, onko kirja esimerkiksi sarjakuva vai romaani, ja genre määrittää kirjan luokan (esim. "jännitys"). Jokaiselle kirjalle tulee lisätä myös kokonaisluku- tai tekstimuotoisen massa (punnitse, jos et tiedä). Jokaiselle kirjalle voidaan merkitä sisäänostohinta. Mikäli sisäänostohintaa ei ole merkitty, saa se arvon 0. Niteellä on myös timestamp-tyylinen myyntipäivämäärä ja kellonaika, joka lisätään onnistuneen myyntitapahtuman yhteydessä.

7.3 Users-taulu

Käyttäjä-taulun pääavain on serial-muotoinen (userId). Käyttäjällä on pakollinen enum-tyyppinen rooli, joka sisältää arvot "customer", "seller" ja "admin". Käyttäjällä on vierasavain (sellerId), mikä lisätään vain käyttäjän roolin ollessa "seller". Käyttäjällä tulee olla yksilöivä tekstimuotoinen käyttäjänimi ja siihen liittyvä salasana. Salasana hajautetaan ja suolataan (hashing and salting) ennen tietokantaan tallentamista. Salasanan tallennus ei välttämättä vastaa oikean tietoturvallisen tuotantokelpoisen sovelluksen standardeja. Kirjautumistietoja tarvitaan sovelluksen toiminnan testaamisessa. Käyttäjä voi lisätä omiin tietoihinsa tekstimuotoisen nimen, osoitteen, postinumeron, kaupungin, sähköpostin ja puhelinnumeron. Näistä nimi on pakollinen ja sähköpostin tulee olla yksilöivä. Käyttäjän yhteystiedot tarvitaan tilausvaiheessa, mutta tästä vastuussa on käyttöliittymä.

7.4 Orders-taulu

Tilaus-taulun pääavain on serial-muotoinen (orderId). Tilaustapahtumassa tallennetaan käyttäjän yksilöivä userId. Tilaustapahtuman aikaleima lisätään onnistuneen tilauksen yhteydessä. Tilauksella on enum-tyyppinen attribuutti status, joka voi saada arvot "pending" ja "completed". Tietyn kuluneen ajanjakson jälkeen tilauksen statuksen ollessa edelleenkin "pending", varatut niteet palautetaan tilaan "available" ja peruttu tilaus poistetaan. Asiakas voi myös itse perua tilauksen. Tilauksella on numeeriset

attribuutit subtotal, shipping ja total. Subtotal käsittää tilaukseen kuuluvien niteiden kokonaishinnan, shipping postikulut ja total tilauksen kokonaishinnan.

7.5 OrderItems-taulu

Tilaustuotteet-aulussa lisätään relaatio tilausten ja niteiden välille, eli se toimii ns. linkkinä näiden välillä. Tilaustuote-aulussa on vain attribuutit orderId ja bookId, jotka ovat samanaikaisesti pääavaimia sekä vierasavaimia.

7.6 Skeema

Tietokannassa käytetään skeemaa erottamaan keskusdivari ja yksittäiset divarit toisistaan. Skeemat kuuluvat myyjille ja myyjät viittavat skeemoihin tietokantaa päivittäessä. Yksittäinen nide voidaan yksilöidä UUID:n avulla, jolloin se voidaan tallentaa turvallisesti eri skeemojen (tietokantojen) välillä.

8 Toteutusvälineet

Ohjelmointikielenä käytetään JavaScriptiä. Alla erittely käytetyistä työkaluista.

8.1 Palvelin

- **Suoritusympäristö:** Node.js
- **Verkkokehys:** Express.js
- **Tietokantamoduuli:** pg
- **Tietokanta:** PostgreSQL

8.2 Käyttöliittymä

- **Verkkokehys:** React
- **Tilanhallinta:** Redux

8.3 Kehitysympäristö

- **Käyttöliittymäkehys:** Vite
- **Ohjelmakoodin formatointityökalu:** Prettier
- **Ohjelmakoodin tarkistustyökalu:** ESLint
- **Versiohallinta:** Git