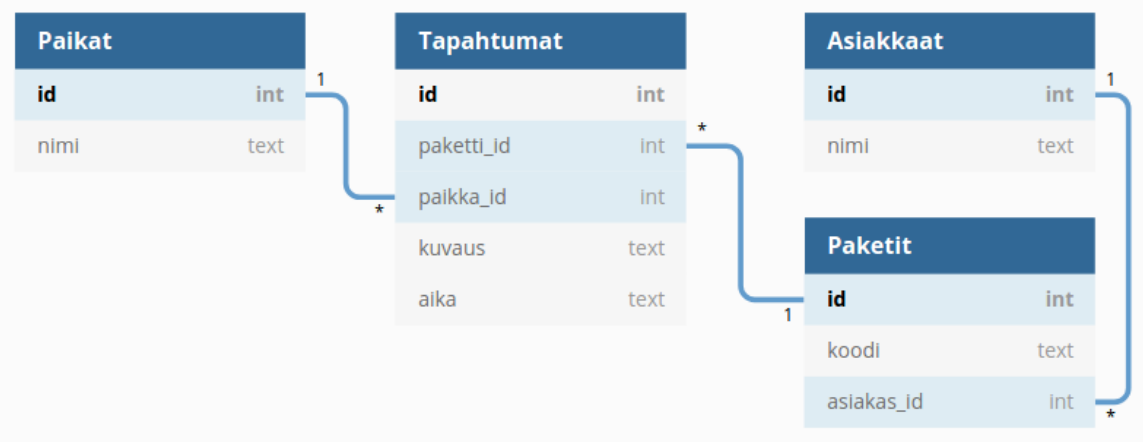
Harjoitustyön tekijä

Antti Laaksonen / 012345678 / ahslaaks@cs.helsinki.fi

Mitä toimintoja on toteutettu?

Harjoitustyössä on toteutettu kaikki vaaditut toiminnot. Lisäksi sovelluksessa on ylimääräinen toiminto “99”, joka sulkee sovelluksen.

Tietokantakaavio



SQL-skeema

```
CREATE TABLE Paikat (id INTEGER PRIMARY KEY, nimi TEXT UNIQUE);
CREATE TABLE Asiakkaat (id INTEGER PRIMARY KEY, nimi TEXT UNIQUE);
CREATE TABLE Paketit (id INTEGER PRIMARY KEY, koodi TEXT UNIQUE, asiakas_id
INTEGER REFERENCES Asiakkaat);
CREATE TABLE Tapahtumat (id INTEGER PRIMARY KEY, paketti_id INTEGER REFERENCES
Paketit, paikka_id INTEGER REFERENCES Paikat, kuvaus TEXT, aika TEXT);
```

Tehokkuustestin tulokset

Ilman indeksejä:

Vaihe 1: 0.05174134 s
Vaihe 2: 0.026400742 s
Vaihe 3: 0.034118238 s
Vaihe 4: 1.698985494 s
Vaihe 5: 0.08004539 s
Vaihe 6: 55.075481665 s

Indeksien kanssa:

Vaihe 1: 0.043521542 s
Vaihe 2: 0.017421849 s
Vaihe 3: 0.020780868 s
Vaihe 4: 5.395822967 s
Vaihe 5: 0.014138468 s
Vaihe 6: 0.053832083 s

Varmistukset

Taulujen määrittelyssä on käytetty UNIQUE-määrettä, minkä ansiosta tietokanta varmistaa, että näissä sarakkeissa joka rivillä on eri arvo. Jos sovelluksessa yritetään laittaa sama arvo uudestaan, tapahtuu SQL-virhe, jonka seurauksena sovellus näyttää virheviestin.

Koska jokainen INSERT-komento on oma transaktionsa, varmistukset toimivat myös silloin, kun sovelluksella on useampi samanaikainen käyttäjä.

Sovelluksen lähdekoodi

```
import java.util.*;
import java.sql.*;

public class Seuranta {
    public static void main(String[] args) throws SQLException {
        Connection db = DriverManager.getConnection("jdbc:sqlite:testi.db");
        Scanner sc = new Scanner(System.in);

        while (true) {
            System.out.print("Valitse toiminto (1-9): ");
            String t = sc.nextLine();
            if (t.equals("1")) {
                Statement s = db.createStatement();
                s.execute("CREATE TABLE Paikat (id INTEGER PRIMARY KEY, nimi
TEXT UNIQUE)");
                s.execute("CREATE TABLE Asiakkaat (id INTEGER PRIMARY KEY, nimi
TEXT UNIQUE)");
                s.execute("CREATE TABLE Paketit (id INTEGER PRIMARY KEY, koodi
TEXT UNIQUE, asiakas_id INTEGER REFERENCES Asiakkaat)");
                s.execute("CREATE TABLE Tapahtumat (id INTEGER PRIMARY KEY,
paketti_id INTEGER REFERENCES Paketit, paikka_id INTEGER REFERENCES Paikat,
kuvaus TEXT, aika TEXT)");
                System.out.println("Tietokanta luotu");
            }
            if (t.equals("2")) {
                System.out.print("Anna paikan nimi: ");
                String paikka = sc.nextLine();
                try {
                    PreparedStatement p = db.prepareStatement("INSERT INTO
Paikat (nimi) VALUES (?)");
                    p.setString(1,paikka);
                    p.executeUpdate();
                    System.out.println("Paikka lisätty");
                } catch (SQLException e) {
                    System.out.println("VIRHE: Paikka on jo olemassa");
                }
            }
            if (t.equals("3")) {
                System.out.print("Anna asiakkaan nimi: ");
                String asiakas = sc.nextLine();
                try {
                    PreparedStatement p = db.prepareStatement("INSERT INTO
Asiakkaat (nimi) VALUES (?)");
                    p.setString(1,asiakas);
                    p.executeUpdate();
                    System.out.println("Asiakas lisätty");
                } catch (SQLException e) {
                    System.out.println("VIRHE: Asiakas on jo olemassa");
```

```
                }
            }
            if (t.equals("4")) {
                System.out.print("Anna seurantakoodi: ");
                String koodi = sc.nextLine();
                System.out.print("Anna asiakkaan nimi: ");
                String asiakas = sc.nextLine();
                PreparedStatement p1 = db.prepareStatement("SELECT id FROM
Asiakkaat WHERE nimi=?");
                p1.setString(1,asiakas);
                ResultSet r = p1.executeQuery();
                if (r.next()) {
                    try {
                        PreparedStatement p2 = db.prepareStatement("INSERT INTO
Paketit (koodi,asiakas_id) VALUES (?,?)");
                        p2.setString(1,koodi);
                        p2.setInt(2,r.getInt("id"));
                        p2.executeUpdate();
                    } catch (SQLException e) {
                        System.out.println("VIRHE: Seurantakoodi on jo
käytetty");
                    }
                } else {
                    System.out.println("VIRHE: Asiakasta ei löytynyt");
                }
            }
            if (t.equals("5")) {
                System.out.print("Anna seurantakoodi: ");
                String koodi = sc.nextLine();
                System.out.print("Anna paikka: ");
                String paikka = sc.nextLine();
                System.out.print("Anna kuvaus: ");
                String kuvaus = sc.nextLine();
                PreparedStatement p1 = db.prepareStatement("SELECT id FROM
Paketit WHERE koodi=?");
                p1.setString(1,koodi);
                ResultSet r1 = p1.executeQuery();
                if (!r1.next()) {
                    System.out.println("VIRHE: Pakettia ei löytynyt");
                    continue;
                }
                PreparedStatement p2 = db.prepareStatement("SELECT id FROM
Paikat WHERE nimi=?");
                p2.setString(1,paikka);
                ResultSet r2 = p2.executeQuery();
                if (!r2.next()) {
                    System.out.println("VIRHE: Paikkaa ei löytynyt");
                    continue;
                }
                PreparedStatement p3 = db.prepareStatement("INSERT INTO
Tapahtumat (paketti_id, paikka_id, kuvaus, aika) VALUES
(?,?,?,datetime('now'))");
                p3.setInt(1,r1.getInt("id"));
                p3.setInt(2,r2.getInt("id"));
                p3.setString(3,kuvaus);
                p3.executeUpdate();
            }
            if (t.equals("6")) {
                System.out.print("Anna seurantakoodi: ");
                String koodi = sc.nextLine();
```

```
                PreparedStatement p = db.prepareStatement("SELECT T.aika,
B.nimi, T.kuvaus FROM Paketit A, Paikat B, Tapahtumat T WHERE A.koodi=? AND
A.id=T.paketti_id AND B.id=T.paikka_id");
                p.setString(1,koodi);
                ResultSet r = p.executeQuery();
                while (r.next()) {
                    System.out.println(r.getString("aika")+" -
"+r.getString("nimi")+" - "+r.getString("kuvaus"));
                }
            }
            if (t.equals("7")) {
                System.out.print("Anna asiakkaan nimi: ");
                String asiakas = sc.nextLine();
                PreparedStatement p = db.prepareStatement("SELECT P.koodi,
COUNT(T.id) maara FROM Asiakkaat A, Paketit P LEFT JOIN Tapahtumat T ON
P.id=T.paketti_id WHERE P.asiakas_id=A.id AND A.nimi=? GROUP BY P.id");
                p.setString(1,asiakas);
                ResultSet r = p.executeQuery();
                while (r.next()) {
                    System.out.println(r.getString("koodi")+" -
"+r.getString("maara"));
                }
            }
            if (t.equals("8")) {
                System.out.print("Anna paikan nimi: ");
                String paikka = sc.nextLine();
                System.out.print("Anna päivämäärä: ");
                String paiva = sc.nextLine();
                PreparedStatement p = db.prepareStatement("SELECT COUNT(*) maara
FROM Tapahtumat T, Paikat P WHERE T.paikka_id=P.id AND P.nimi=? AND T.aika
LIKE ?");
                p.setString(1,paikka);
                p.setString(2,paiva+"%");
                ResultSet r = p.executeQuery();
                System.out.println("Tapahtumien määrä: "+r.getInt("maara"));
            }
            if (t.equals("9")) {
                System.out.print("Haluatko indeksit (K/E)? ");
                Statement s = db.createStatement();
                Random r = new Random();
                if (sc.nextLine().equals("K")) {
                    s.execute("CREATE INDEX idx1 ON Paketit (asiakas_id)");
                    s.execute("CREATE INDEX idx2 ON Tapahtumat (paketti_id)");
                }
                long aika1 = System.nanoTime();
                s.execute("BEGIN TRANSACTION");
                PreparedStatement p1 = db.prepareStatement("INSERT INTO Paikat
(nimi) VALUES (?)");
                for (int i = 1; i <= 1000; i++) {
                    p1.setString(1,"P"+i);
                    p1.executeUpdate();
                }
                s.execute("COMMIT");
                long aika2 = System.nanoTime();
                System.out.println("Vaihe 1: "+(aika2-aika1)/1e9+" s");
                s.execute("BEGIN TRANSACTION");
                PreparedStatement p2 = db.prepareStatement("INSERT INTO
Asiakkaat (nimi) VALUES (?)");
                for (int i = 1; i <= 1000; i++) {
                    p2.setString(1,"A"+i);
```

```
                    p2.executeUpdate();
                }
                s.execute("COMMIT");
                long aika3 = System.nanoTime();
                System.out.println("Vaihe 2: "+(aika3-aika2)/1e9+" s");
                s.execute("BEGIN TRANSACTION");
                PreparedStatement p3 = db.prepareStatement("INSERT INTO Paketit
(koodi,asiakas_id) VALUES (?,?)");
                for (int i = 1; i <= 1000; i++) {
                    p3.setString(1,"K"+i);
                    p3.setInt(2,r.nextInt(1000)+1);
                    p3.executeUpdate();
                }
                s.execute("COMMIT");
                long aika4 = System.nanoTime();
                System.out.println("Vaihe 3: "+(aika4-aika3)/1e9+" s");
                s.execute("BEGIN TRANSACTION");
                PreparedStatement p4 = db.prepareStatement("INSERT INTO
Tapahtumat (paikka_id,paketti_id,kuvaus) VALUES (?,?,?)");
                for (int i = 1; i <= 1000000; i++) {
                    p4.setInt(1,r.nextInt(1000)+1);
                    p4.setInt(2,r.nextInt(1000)+1);
                    p4.setString(3,"abc");
                    p4.executeUpdate();
                }
                s.execute("COMMIT");
                long aika5 = System.nanoTime();
                System.out.println("Vaihe 4: "+(aika5-aika4)/1e9+" s");
                PreparedStatement p5 = db.prepareStatement("SELECT COUNT(*) FROM
Paketit WHERE asiakas_id=?");
                for (int i = 1; i <= 1000; i++) {
                    p5.setInt(1,r.nextInt(1000)+1);
                    p5.executeQuery();
                }
                long aika6 = System.nanoTime();
                System.out.println("Vaihe 5: "+(aika6-aika5)/1e9+" s");
                PreparedStatement p6 = db.prepareStatement("SELECT COUNT(*) FROM
Tapahtumat WHERE paketti_id=?");
                for (int i = 1; i <= 1000; i++) {
                    p6.setInt(1,r.nextInt(1000)+1);
                    p6.executeQuery();
                }
                long aika7 = System.nanoTime();
                System.out.println("Vaihe 6: "+(aika7-aika6)/1e9+" s");
            }
            if (t.equals("99")) {
                break;
            }
        }
    }
}
```