



# TIETOKANTAOHJELMOINTI HARJOITUSTYÖ

Vaihe 2: Toteutus

Ryhmä 12

Antti Pessa, Henri Hakkarainen ja Risto Meltaus

## Sisällysluettelo

1. Johdanto .....	2
2. Ohjelmaan toteutetut ominaisuudet .....	2
Tapahtumat.....	2
Raportit .....	3
Tapahtumien hallinta .....	5
Käyttöliittymäominaisuudet .....	7
Testiaineisto .....	7
Lisätoiminnot .....	7
3. Työnjako.....	8
4. Kuvaus toteutuksesta .....	9
5. Asennusohjeet ja ohjelman käyttö .....	10
Miten ohjelma valmistellaan käyttöä varten? .....	10
Miten ohjelmaa käytetään? .....	10
6. Muutokset.....	12
7. Oma arvio työstä.....	13
8. Päivitetty ER kaavio .....	15
9. Päivitetty tietokantakaavio.....	16
10. ER kaavion muunnos tietokantakaavioksi .....	17

## 1. Johdanto

Tavoitteena on suunnitella ja toteuttaa työtarjouksiin, -suorituksiin sekä laskujenhallintaan käytettävä tietokanta ja sen käytön mahdollistava sovellus. Tietokantana toimii PostgreSQL. Käyttöliittymä ja toiminnallisuudet on kirjoitettu Javalla.

## 2. Ohjelmaan toteutetut ominaisuudet

Toteutimme ohjelmaan kaikki tapahtumat ja raportit. Alla kuvaus siitä, miten ne tehdään / saadaan ulos tehdyssä käyttöliittymässä.

### Tapahtumat

T1: Lisätään asiakkaalle xx uusi työkohde.

1. Siirrytään ohjelmassa "Lisää työkohde" näkymään
2. Haetaan asiakas ja valitaan nimi listasta
3. Valitaan työn tyyppi: urakka vai tunti
4. Kirjoitetaan työkohteen osoite
5. Maksuerien lukumäärä: 1 tai 2
6. Valitaan, onko työkohde tarjous vai suora sopimus
7. Sitten painetaan luo kohde painiketta ja uusi työkohde on lisätty tietokantaan

T2: Tallennetaan työkohteeseen liittyvät tuntityöt ja käytetyistä tarvikkeista tiedot päivän päätteeksi.

1. Siirrytään ohjelmassa "Lisää veloituksia kohteeseen" näkymään
2. Haetaan ja valitaan kohde, jolle halutaan lisätä tunteja ja tarvikkeita
3. Tunnin lisäys: Lisää tunteja pudotusvalikosta valikoidaan annettava tuntityyppi ja tuntien määrään lisätään haluttu määrä ja sitten painetaan lisää painiketta
4. Tarvikkeen lisäys: Haetaan haluttu tarvike ja valitaan se listasta, sitten tarvikkeen määrään asetetaan käytetty määrä ja painetaan lisää painiketta
5. Lisätyt tunnit ja tarvikkeet näkyvät listattuna valittuun kohteeseen

T3: Muodosta muistutuslasku laskuista, joita ei ole maksettu ja joiden eräpäivä umpeutunut, ja joista ei ole aiemmin lähetetty muistutuslaskua

1. Siirrytään ohjelmassa "Laskujen hallinta" näkymään
2. Valitaan pudotusvalikosta Eräpäivä umpeutunut
3. Painetaan lähetä muistutuslaskut painiketta

T4: Muodosta karhulasku (kolmas) muistutuslaskuista, joita ei ole maksettu ja joiden eräpäivä umpeutunut

1. Siirrytään ohjelmassa "Laskujen hallinta" näkymään

2. Valitaan pudotusvalikosta Eräpäivä umpeutunut
3. Painetaan lähetä muistutuslaskut painiketta

**Huom!** Muistutuslaskut ja karhulaskut luodaan samalla painikkeella käyttöliittymässä, jolloin pystytään helposti generoimaan seuraavat laskut kaikille niille laskuille, joissa eräpäivä on mennyt umpeen (tapahtumat T3 ja T4).

T5: Tavarantoimittaja lähettää uuden hinnaston (tekstimuodossa). Pitää korvata olemassa olevat sekä pitää poistaa vanhat ja lisätä uudet. Vanhat tuotteet ja tarvikkeet on toimitettava historiakansioon

1. Siirrytään ohjelmassa "Päivitä hinnasto/Lisää tarvike" näkymään
2. Painetaan selaa painiketta
3. Avataan tiedosto "hinnasto.txt"
4. Tarvikelista on nyt päivitetty tietokantaan ja on nähtävissä "Tarvikelistaus" näkymästä
5. Poistetut tuotteet löytyvät, kun valitsee "Poistettu" pudotusvalikosta

**Huom.** tarvikkeiden on oltava tiedostossa muodossa:  
nimi;yksikko;ostohinta;kate;alv

Jos uusi tuote korvaa olemassa olevan tuotteen, vanhan tuotteen tila muuttuu siten, että sitä ei enää voi lisätä työkohteeseen. Käytöstä poistuneita tuotteita voi selata tarvikelistaus näkymästä.

## Raportit

R1: Muodosta hinta-arvio kohteeseen x, joka sisältää suunnittelua 3 tuntia, asennustyötä 12 tuntia, 3 metriä sähköjohtoa sekä yhden pistorasian.

1. Lisätään halutut tunnit ja tarvikkeet "Lisää veloituksia kohteeseen" näkymästä
2. Siirrytään "Hallinnoi tarjouksia" näkymään
3. Sieltä haetaan kohde ja avautuu hinta-arvio

R2: Tuntityölasku tarvittavine tietoineen

- asiakastiedot (kohteen osoite voi olla eri kuin asiakkaan osoite)
- tarvikkeet (vähintään 2 erityyppistä)
- tuntierittely (vähintään 2 erityyppistä)
- kokonaissumma
- kotitalousvähennyskelpoisuus

### 1. Laskun muodostaminen

1. Siirry ohjelmassa "Päätä kohde ja laskuta" näkymään
2. Haetaan ja valitaan kohde listasta.
3. Painetaan "Päätä työ ja luo lasku" painiketta

### 2. Laskun tarkastelu

1. Siirry ohjelmassa "Laskujen hallinta" näkymään

2. Valitse pudotusvalikosta haluttu näkymä, mitä laskuja listataan (voit suodattaa esim. laskut joiden eräpäivä on umpeutunut, laskut jotka ovat maksamatta, muistutuslaskut, karhulaskut tai maksetut laskut).
3. Valitse lasku jota haluat tarkastella (rivin klikkaaminen avaa uuden ikkunan erittelylle).
4. Erittelystä näet asiakastiedot, tarvikkeet ja tunnit omissa tauluissaan sekä kokonaissumman. Kotitalousvähennyskelpoisuus käy ilmi laskulta eritellyn työn hinnasta (alv 0 ja alvillinen hinta näkyvissä erikseen). Laskulla näkyy myös työsuorituksen tyyppi (urakka- tai tuntityö). Tuntityölaskulla on eritelty tarkemmin yksittäisten rivien (tunnit ja tarvikkeet) hintoja, kun taas urakkalaskulla on ilmoitettu vain näiden yhteissummat.

### R3. Kuten R2, mutta lisäksi

- suunnittelutyölle on annettu 10% alennus
- sähköjohdolle on annettu 10% alennus
- muille tarvikkeille annettu 20% alennus
- Opaskirjan (10 euroa – huom. eri alv)
- Alennukset kohdistuvat alv-verottomaan hintaan

#### 1. Laskun muodostaminen

1. Siirry ohjelmassa ”Päätä kohde ja laskuta” näkymään
2. Haetaan ja valitaan kohde listasta.
3. Alennus annetaan muokkaamalla haluttua riviä valitussa taulukossa ja kuittaamalla tehdyt muutokset painamalla Enteriä. Tässä vaiheessa siis annetaan rivialennus suunnittelutyölle, sähköjohdolle ja muille tarvikkeille rivi kerrallaan.
4. Painetaan ”Päätä työ ja luo lasku” painiketta

#### 2. Laskun tarkastelu

1. Siirry ohjelmassa ”Laskujen hallinta” näkymään
2. Valitse pudotusvalikosta haluttu näkymä, mitä laskuja listataan (voit suodattaa esim. laskut joiden eräpäivä on umpeutunut, laskut jotka ovat maksamatta, muistutuslaskut, karhulaskut tai maksetut laskut).
3. Valitse lasku jota haluat tarkastella (rivin klikkaaminen avaa uuden ikkunan erittelylle).
4. Erittelystä näet asiakastiedot, tarvikkeet ja tunnit omissa tauluissaan sekä kokonaissumman. Kotitalousvähennyskelpoisuus käy ilmi laskulta eritellyn työn hinnasta (alv 0 ja alvillinen hinta näkyvissä erikseen). Laskulla näkyy myös työsuorituksen tyyppi (urakka- tai tuntityö). Tuntityölaskulla on eritelty tarkemmin yksittäisten rivien (tunnit ja tarvikkeet) hintoja, kun taas urakkalaskulla on ilmoitettu vain näiden yhteissummat.
5. Tunneista ja tarvikkeista on eritelty myös näistä kustakin mahdollisesti annetut alennukset prosentteina, sekä alv-prosentin suuruus. Opaskirjan alv on eri kuin muilla tarvikkeilla ja ohjelma osaa laskea kunkin tarvikkeen alvillisen hinnan erikseen siihen liitetyn alv-prosentin perusteella.

### R4. Urakkatarjous, joka sisältää

- asiakkaan ja työkohteen tiedot
- Arvioidun työn osuuden (peruste 5 tuntia suunnittelua, 20 tuntia asennustyötä; annetaan 10% alennus)
- Tarvikkeiden osuudet (3 eri tyyppiä, ainakin 2 kutakin)
- Alv-erittely

## 1. Tarjouksen laatiminen

1. Luodaan uusi työkohde (kts. tapahtuma T1) ja jätetään valintaruutu "Tarjous" valituksi.
2. Lisätään työkohteelle tunteja ja tarvikkeita "Lisää veloituksia kohteeseen" näkymässä (kts. tapahtuma T2).

## 2. Tarjouksen tarkastelu

1. Siirry ohjelmassa "Hallinnoi tarjouksia" näkymään
2. Hae tarjousta kohteen osoitteella ja valitse tarkasteltava tarjous listasta (rivin klikkaaminen avaa tarjouksen erittelyn uuteen ikkunaan).
3. Tarjousikkunassa näkyy eriteltynä asiakkaan nimi, työkohteen osoite, arvioidut työn määrät ja tarvikkeiden osuudet ja näille on laskettu lopulliset hinnat, joissa alv 0 hinta ja lopullinen summa (sis. alv) ovat eriteltynä tuntien ja tarvikkeiden osalta.
4. Tunneille ja tarvikkeille voidaan antaa rivikohtaisesti alennus muokkaamalla valitun rivin Alennus % -saraketta ja kuittaamalla tehdyt muutokset painamalla Enteriä. Alennuksen antaminen päivittää myös loppusumman.

R5. Muodosta hyväksytystä urakkatarjouksesta kaksi samansuuruista laskua siten, että toinen laskutetaan heti ja toinen ensi vuoden tammikuun 1 päivä.

1. Siirry ohjelmassa "Päätä kohde ja laskuta" näkymään
2. Hae työkohde osoitteella ja valitse päätettävä kohde listasta
3. (Tässä vaiheessa voi vielä antaa lisää alennusta tunneille ja tarvikkeille rivikohtaisesti)
4. Painetaan "Päätä työ ja luo lasku" painiketta, joka luo automaattisesti kaksi samansuuruista laskua, mikäli työkohteen erälukumääräksi oli sen luontivaiheessa asetettu  
2. Laskut luodaan siten, että ensimmäinen erä laskutetaan heti (28 päivää maksuaikaa luontihetkestä) ja toisen laskun luontipäiväksi asetetaan ensi vuoden tammikuun 1. päivä samalla maksuajalla 28 päivää.

## Tapahtumien hallinta

Ohjelmassa on käytetty tapahtumanhallintaa metodeissa, joissa kutsutaan useampaa SQL-lausetta jotka muokkaavat tietokannan tilaa. Tapahtumanhallinnalla taataan, että loogiset kokonaisuudet suoritetaan joko kokonaan tai jos joku päivityslauseista epäonnistuu, niin tietokannan tila palautetaan lähtötilanteeseen.

### Esimerkki:

```
public void lisääTarvike(String nimi, String yksikko, double ostohinta, double kate, double alv) throws
SQLException {
```

```

try {
    con.setAutoCommit(false);
    Statement stmt = con.createStatement();
    String select, insert, update;
    select = "SELECT tarvikkeid, nimi, yksikko, ostohinta, kate, alv "
        + "FROM tarvike WHERE nimi = '%s' AND tila = 'käytössä'";
    ResultSet rs = stmt.executeQuery(String.format(select, nimi));
    if (rs.next()) {
        int tarvikkeid = rs.getInt("tarvikkeid");
        String n = rs.getString("nimi");
        String y = rs.getString("yksikko");
        double oh = rs.getDouble("ostohinta");
        double k = rs.getDouble("kate");
        double a = rs.getDouble("alv");
        // Kaikki tiedot eivät vastaa uuden lisättävän tarvikkeen kanssa
        if (!n.equals(nimi) || !y.equals(yksikko) || oh != ostohinta || k != kate || a != alv) {
            // Lisätään tarvike päivitetyillä tiedoilla
            insert = "INSERT INTO tarvike (nimi, yksikko, ostohinta, kate, alv) "
                + "VALUES ('%s', '%s', %s, %s, %s)";
            stmt.executeUpdate(String.format(insert, nimi, yksikko, ostohinta, kate, alv));
            // Päivitetään vanhan tarvikkeen tila
            update = "UPDATE tarvike SET tila = 'vanhentunut' WHERE tarvikkeid = " + tarvikkeid;
            stmt.executeUpdate(update);
        }
    } else {
        // Lisätään uusi tarvike
        insert = "INSERT INTO tarvike (nimi, yksikko, ostohinta, kate, alv) "
            + "VALUES ('%s', '%s', %s, %s, %s)";
        stmt.executeUpdate(String.format(insert, nimi, yksikko, ostohinta, kate, alv));
    }
    con.commit();
    rs.close();
    stmt.close();

} catch (SQLException e) {
    con.rollback();
    throw new SQLException(e.getMessage());
} finally {
    con.setAutoCommit(true);
}
}

```

Metodin alussa setAutoCommit(false) ja sitten kun metodin suoritus päättyy, asetetaan se takaisin true. Tehdään commit vasta kun kaikki tapahtumat on suoritettu onnistuneesti. Jos tapahtuu SQLException, eli joku tapahtumista epäonnistuu, tehdään rollback ja palautetaan tietokannan tila lähtötilanteeseen.

## Käyttöliittymäominaisuudet

Käyttöliittymäominaisuudet toteutettiin mielestämme laajasti. Graafisella käyttöliittymällä mahdollistetaan tottumattomalle käyttäjälle tehokas ohjelman käyttö ilman komentojen opettelua ohjaamalla käyttötapausten kulkua ja erittelemällä erilaisia toimintoja omiin näkymiinsä.

## Testiaineisto

Tietokannan luontilauseet ja lisäyslauseet löytyvät projektikansion juuresta sql-kansion sisältä. Ajamalla nämä lauseet tietokantaan, saadaan luotua työtä varten tarvittavat relaatiot ja lisättyä niihin hyvä määrä rivejä ohjelman testausta varten.

Tiedostojen nimet:

- luontilauseet.sql
- insert.sql

## Lisätoiminnot

- Asiakkaan lisäys onnistuu suoraan käyttöliittymästä
- Hakuominaisuus asiakkaille, tarvikkeille ja työkohteille  
*Haut toimivat siten, että jos hakukenttä on tyhjänä, kun haku suoritetaan, palautuu kaikki rivit tietokannasta, mutta jos hakukenttä sisältää tekstiä, niin tietokannasta haetaan ne rivit, joissa syöte esiintyy taulusta haettavan sarakkeen kohdalla (esim. työkohteen osoite tai tarvikkeen / asiakkaan nimi)*
- Usean muistutuslaskun luominen kerralla
- Ohjataan käyttäjää ohjelman sisäisillä viesteillä
- Myös tuntitarjouksen voi laskuttaa monessa erässä
- Tarvikkeita voi lisätä käyttöliittymässä myös yksitellen syöttämällä tiedot input-kenttiin ja luomalla tarvikkeen näiden tietojen pohjalta.
- Lasku erittelee erien lukumäärät ja kuinka mones erä se itse on.
- Muistutuslaskut ja karhulaskut kertovat mihin aiempaan laskuun ne liittyvät.
- Pystyy selaamaan laskuja ja kohdistamaan suodatuksen eräpäivän ylittäneisiin laskuihin, maksamattomiin laskuihin, muistutuslaskuihin, karhulaskuihin ja maksettuihin laskuihin.
- Keskeneräisestä työkohteesta pystyy poistamaan tunteja ja tarvikkeita.



### 3. Työnjako

Java-ohjelmaan liittyvä työnjako on melko hankala eritellä, sillä kaikki tekivät koodia ja useimpia metodeja ja käyttöliittymän näkymiä muokattiin varmaan jokaisen toimesta vuorotellen ja useaan otteeseen. Todettakoon siis, että Java-koodauksen työnjako meni jokaiselle n. 33-prosenttisesti eli kaikki tekivät hyvin hommia käyttöliittymän ja ohjelman toimivuuden eteen. Alla lueteltu ns. muut tehtävät projektin onnistumisen eteen.

#### Antti:

Suunnitteluvaiheessa tein SQL luontilauseita ja dokumentoin tarvittavia osioita. Toteutuksessa tein paljon koodia tarvikkeen lisäys näkymään, mutta kuten ylempänä on todettu, on kaikki auttanut ja koodannut aika lailla kaikkea projektissa.

#### Henri:

ER- ja SQL-kaavioiden teko ja näihin liittyvä dokumentointi, Docker-tiedostojen ja ympäristön pystyttäminen (ohjelmiston kehitystyössä käytetty), rajoitusten lisääminen tietokannan luontilauseisiin sekä useiden tietokannan lisäyslauserivien teko, jotta saataisiin hyvä alkutila tietokannan testausta varten.

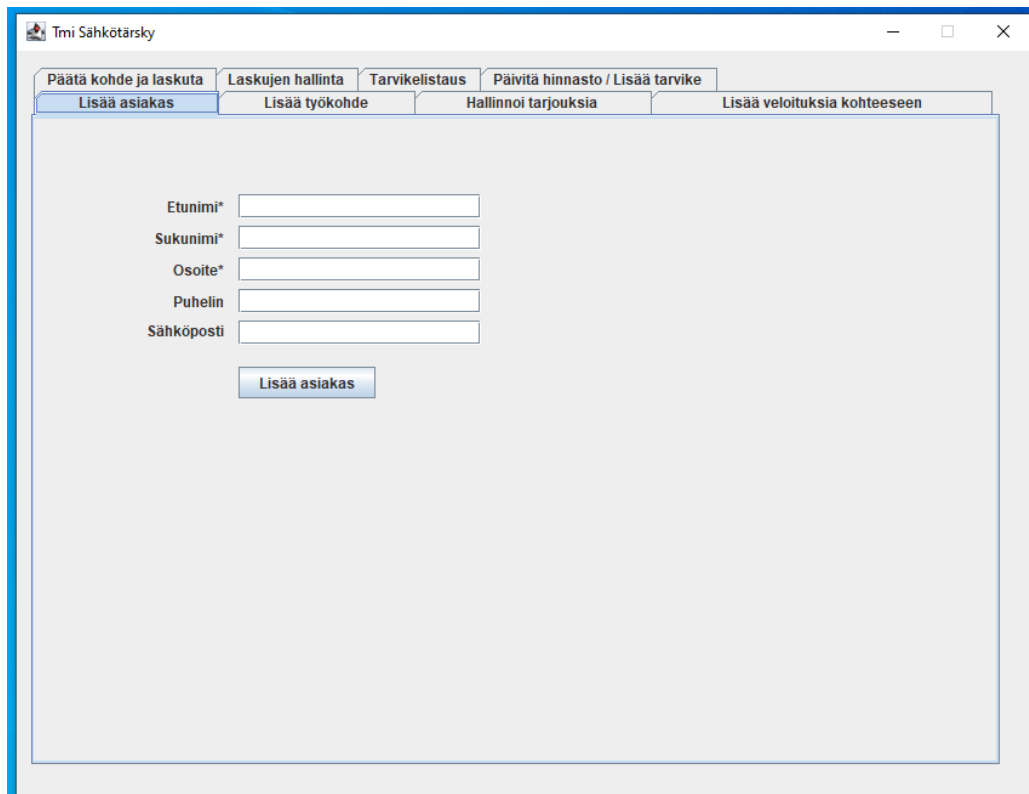
#### Risto:

Tapahtumien ja raporttien työnkulku ja jonkinmoiset suuntaviivat toteutukselle.

## 4. Kuvaus toteutuksesta

Toteutimme ohjelman Javalla ja teimme käyttöliittymän Swing käyttöliittymäkirjastolla. Ohjelma jakautuu kolmeen tiedostoon Tiko2020, GUI ja DBManager, missä tehdään kaikki tietokantakyselyt. Tiko2020 on pääluokka, josta ohjelma käynnistyy.

Swing mahdollistaa graafisten käyttöliittymien toteutuksen helposti. Esimerkki tuotoksesta kuvassa 1. Swing -käyttöliittymä palveli tämän tehtävän toiminallisuuksia ja ominaisuuksia oikein hyvin.



The screenshot shows a Java Swing window titled "Tmi Sähkötärsky". The window has a menu bar with the following items: "Päätä kohde ja laskuta", "Laskujen hallinta", "Tarvikelistaus", and "Päivitä hinnasto / Lisää tarvike". Below the menu bar is a toolbar with four buttons: "Lisää asiakas", "Lisää työkohde", "Hallinnoi tarjouksia", and "Lisää veloituksia kohteeseen". The "Lisää asiakas" button is currently selected. The main area of the window contains a form with five text input fields, each with a label to its left: "Etunimi\*", "Sukunimi\*", "Osoite\*", "Puhelin", and "Sähköposti". Below these fields is a "Lisää asiakas" button.

Kuva 1: "Lisää asiakas" näkymä.

## 5. Asennusohjeet ja ohjelman käyttö

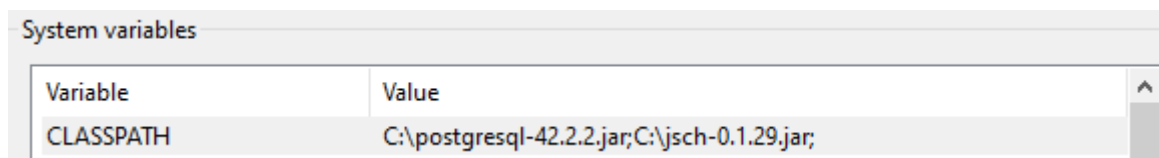
### Miten ohjelma valmistellaan käyttöä varten?

Java-ohjelman ajaminen omalla koneella edellyttää asennettua Java JDK:ta (versio 11 tai uudempi). Ohjelman koodissa on käytetty paljon mm. String-luokan [isBlank](#) funktiota, joka on saatavilla vasta Javan versiosta 11 eteenpäin ja siitä syystä ohjelman ajaminen vanhemmilla Javan versioilla päättyy virheeseen.

### Miten ohjelmaa käytetään?

Tarjoamme palautusta varten kaksi vaihtoehtoa siihen, mitä tietokantaa ohjelma käyttää. Yhteyden voi muodostaa joko tunipalvelimella sijaitsevaan tietokantaan ssh-tunneloinnin avulla tai käyttää paikallisella työasemalla olevaa tietokantaa.

Ohjeet toimivat Windows-ympäristössä. Tärkeä askel on lisätä harjoitustyökansiossa olevat jsch-0.1.29.jar ja postgresql-42.2.2.jar CLASSPATH muuttujaan, kuva 2, ja tämän jälkeen käynnistää tietokone uudelleen.



Kuva 2: Kriittiset järjestelmämuuttujat.

### Vaihtoehto 1 – tietokanta sijaitsee koulun palvelimella

1. Syötä halutut luonti- ja lisäyslauseet tietokantaan.
2. Aseta tunikanta kansiossa olevaan DBManager.java tiedostoon SSH yhteyden vaatimat käyttäjätunnus sekä salasana sekä tietokannan vaatima käyttäjätunnus ja salasana.
3. Käännä kansiossa tunikanta olevat .java tiedostot komennolla: `javac -encoding utf8 *.java`
4. Käynnistä ohjelma tunikanta kansiossa ajamalla komento: `Java Tiko2020`

Ssh-tunneloinnin vaatima JSch paketti on tallennettu suorituskansioon eikä sen suhteen tarvitse tehdä erityisiä toimenpiteitä.

### Vaihtoehto 2 – tietokanta sijaitsee omalla työasemalla (vaatii Postgres asennuksen)

Lokaalin PostgreSQL tietokannan pystyttäminen:

1. Asenna postgres 12 <https://www.postgresql.org/download/>
2. Avaa PGADMIN4
3. Siellä pitäisi olla valmiiksi postgres niminen tietokanta
4. Avaa Tools valikosta Query tool ja aja tietokannan luonti- ja lisäyslauseet

Lokaalikanta kansion DBManager.java tiedostossa seuraavat vakiot pitää asettaa vastaamaan oman PostgreSQL-tietokannan speksejä:

```
private static final String PROTOKOLLA = "jdbc:postgresql:";
private static final String PALVELIN = "localhost";
private static final int PORTTI = 5432;           // default portti on 5432
private static final String TIETOKANTA = "postgres"; // tähän tietokannan nimi
private static final String KAYTTAJA = "postgres";  // tähän tietokannan käyttäjä
private static final String SALASANA = "salasana";  // tähän tietokantakäyttäjän salasana
```

Ohjelman suoritusta varten siirrytään ensin kansioon lokaalikanta ja siellä käännetään kaikki ohjelman lähdekooditiedostot komennolla: `javac -encoding utf8 *.java`

Tämän jälkeen ohjelma ajetaan (Windows -ympäristössä) samassa kansiossa komennolla:  
`Java Tiko2020`

### **Toiminta**

Ohjelma ottaa ensin yhteyden tietokantaan DBManager-luokassa annettujen speksien mukaisesti ja mikäli yhteys luodaan onnistuneesti, aukeaa ruudulle uusi ikkuna, joka toimii ohjelman käyttöliittymänä. Mikäli tietokantayhteys epäonnistuu, ohjelman suoritus pysähtyy (tämä käy ilmi komentotulkin tuottamasta tulosteesta) ja tässä tapauksessa tulee tarkistaa DBManager-luokassa annettujen yhteysparametrien oikeellisuus.

## 6. Muutokset

### Miksi varastotilanne poistettiin tarvike-aulusta?

Varastotilanne-sarake jätettiin pois tarvike-aulusta, koska totesimme että sen päivittäminen lisää ohjelmaan turhaa monimutkaisuutta, eikä varastohallinta ole laskutusjärjestelmän tehtävä. Esimerkiksi kun työkohteelle lisätään tarvikkeita, niin olisiko mahdollista lisätä tarvikkeita joita ei ole sillä hetkellä varastossa (varastotilanne menisi miinukselle), koska periaatteessahan esim. urakkatarjousta laatiessa listataan vain mitä työhön arvioidaan tarvittavan ja siinä vaiheessa tavarat pysyisivät yhä varastossa, mutta sitten kun päivän päätteeksi lisätään työsuoritukseen päivän aikana käytetyt tavarat, niin näiden tulisi sitten taas ollakin varastossa koska muutenhan niitä ei olisi alun perinkään voitu käyttää. Tuumimme siis, että Sepolla on yleisimmin työssään tarvitsemia tarvikkeita ”riittävästi” ja erikoisempia asiakkaiden haluamia hankitaan kysynnän mukaan. Varastotilanne-sarakkeen päivittämisen monimutkaisuuden takia tuo tieto jätettiin pois toteutuksesta.

### Miksi ei ole työsuoritus-aulua?

Välipalautuksen kommenttina oli ehdotettu, että kannattaisi lisätä uusi entiteettityyppi ”työsuoritus”, johon lasku-, tarvike- ja tuntityyppi-aulut linkitettäisiin. Emme kuitenkaan lähteneet monimutkaistamaan tietokannan rakennetta, sillä saimme kaiken toiminnallisuuden tehtyä nykyisellä mallilla. Samaan työkohteeseen (sama kohdeid) ei nyt voi tehdä useampia työsuorituksia, mutta samaan osoitteeseen voi tehdä myöhemmin lisätoita luomalla uuden työkohteen tätä varten ja linkittämällä sitten laskun sekä tarvikkeet ja tunnit tähän kohteeseen. Harjoitustyön tehtävänannossa ei teknisesti ottaen edellytetty, että samaan työkohteeseen pitää pystyä tekemään useampia työsuorituksia, joten ajattelimme oman mallimme olevan riittävä, sillä tuskin niin monia työsuorituksia tehdään samalle työkohteelle, että tällä tavalla tulisi ainakaan hirveästi turhaa toistoa tietokannassa.

### Muutokset tietokantaan?

Lisäsimme ER-kaavion liittyy-suhteeseen perintakulu-attribuutin, joka kertoo laskuun liittyvän laskutuslisän suuruuden (jokaiseen muistutuslaskuun lisätään 5€ laskutuslisä). SQL-kaaviossa ja tietokannassa tämä attribuutti lisättiin siis lasku-relaatiolle.

Tietokannan luontilauseisiin tehtiin myös toteutusvaiheen aikana jonkin verran lisää rajoituksia ja tarkistuksia, jotta taulun sarakkeissa olevat tiedot pysyvät järkevinä. Esimerkkinä vaikkapa laskun tila-sarakkeen rajoitukset, eli lisäys- tai muokkauslauseilla tilaksi voi asettaa joko ”kesken”, ”siirtynyt” tai ”valmis” ja näiden tietojen perusteella pystyimme ohjelmassa hakemaan halutun tyyppiset laskut.

Lisäsimme myös työkohde-auluun uuden attribuutin joka kertoo onko työkohteen tila vielä tarjous vai onko tarjous hyväksytty ja työ näin ollen toteutus/laskutusvaiheessa. Sarakkeen nimi relaatiossa on ”tarjous” ja sen tietotyyppi on BOOLEAN, eli true, jos tarjous on hyväksymättä ja false jos työ on edennyt toteutukseen.

## 7. Oma arvio työstä

### Mikä oli vaikeaa ja muut kommentit harjoitustyön tekemisestä?

#### Henri:

Vaikeinta oli mallintaa toteutukseen sopivanlainen tietokantamalli tauluineen ja rajoituksineen. Suunnitteluvaiheen aikana meinaisi mennä useampanakin iltana pää jumiin kun yritti saada esim. tarvike ja tuntityyppi -taulut liitettyä järkevästi siten, että ne löytyvät sitten myöhemmin laskuilta, työsuorituksista, urakkatarjouksista ja ties mistä.

Javalla ohjelman toteutus oli helpompi ja miellyttävämpi osuus harjoitustyöstä. Oli myös hyvä päästä käyttämään entuudestaan itselle vieraampaa Java kirjastoa (Swing) ja harjoitustyötä tehdessä oppi myös jonkin verran uusia asioita erityisesti Java-ohjelmointiin liittyen, mutta myös SQL tuli entistä tutummaksi.

Harjoitustyön tehtävänanto oli hyvin monipuolinen ja saimme ryhmänä tosiaan miettiä useampaan kertaan tiettyjen vaatimusten toteutustapaa, jotka muuttuivatkin useaan kertaan vielä suunnitteluvaiheen jälkeenkin. Tehtävänanto vaikutti toisinaan hieman vaikeasti ymmärrettävältä, koska siinä kokonaisuus on erittäin tiiviisti esitetty, mutta projektityön kannalta tämä on ehkä hyväkin asia koska se pakottaa käyttämään enemmän aikaa pohdintaan ja ryhmän tekemään omia päätöksiä sen sijaan, että olisi tarkasti määritelty miten mikäkin tietokantataulu tms. tulisi toteuttaa.

Henkilökohtaisesti olen myös tyytyväinen ryhmän toimintaan, kaikki tekivät oman osuutensa ja työnjako oli melko selkeä. Pidimme toteutusvaiheessa Zoom-palavereita n. kerran viikossa, joissa keskustelimme työn etenemisestä ja jaoimme tehtäviä, että kaikki toiminnallisuudet saataisiin valmiiksi.

#### Antti:

Suunnitteluvaiheessa kokonaisuuden hahmottaminen vei hieman aikaa, kun piti miettiä, kuinka monta taulua tarvittaisiin ja kuinka ne kaikki yhdistyisivät. Toteutuksessa eniten päänvaivaa aiheutti lokaalin PostgreSQL tietokannan pystyttäminen. Docker Toolboxissa pyörinyt kontti ei saanut yhteyttä Java ohjelmaan, joten piti asentaa lokaali Postgres ja senkin kanssa piti tapella tovi.

Swingi toimi tehtävään mainiosti, mutta koska kaikki tehtiin samaan formiin tuli koodista aika vaikealukuista, kun kaikki komponentit tulivat samaan tiedostoon.

Huomasimme että Netbeans ja IntelliJ IDEA käyttävät erilaisia formeja joten teimme kaiken koodin Netbeansilla jotta form editori toimi. Alun perin olin halunnut tehdä harjoitustyön IntelliJ IDEAlla. :(

#### Risto:

Mielenkiintoisin osuus työstä oli ehdottomasti alun suunnitteluvaihe. Se oli osin myös hieman hankala. Aivan kaikkia työssä tarvittavia asioita ei tullut otettua huomioon ja joitain ylimääräisiä asioita karsittiin myöhemmin pois. Juuri ideointivaiheessa ryhmässä toiminta oli suuri plussa, sillä ideoiden pallottelu tuotti tuloksia.

Suurimmat vaikeudet nousivat kehitysympäristöjen kanssa taistelusta ja SSH tunnelointia vaativasta tietokannasta, johon ei ole yhteistä käyttäjätiliä.

Koodin määrä räjähti taas käsiin ja Swing ei ole niin tuttu, että sitä olisi osannut pilkkoa.

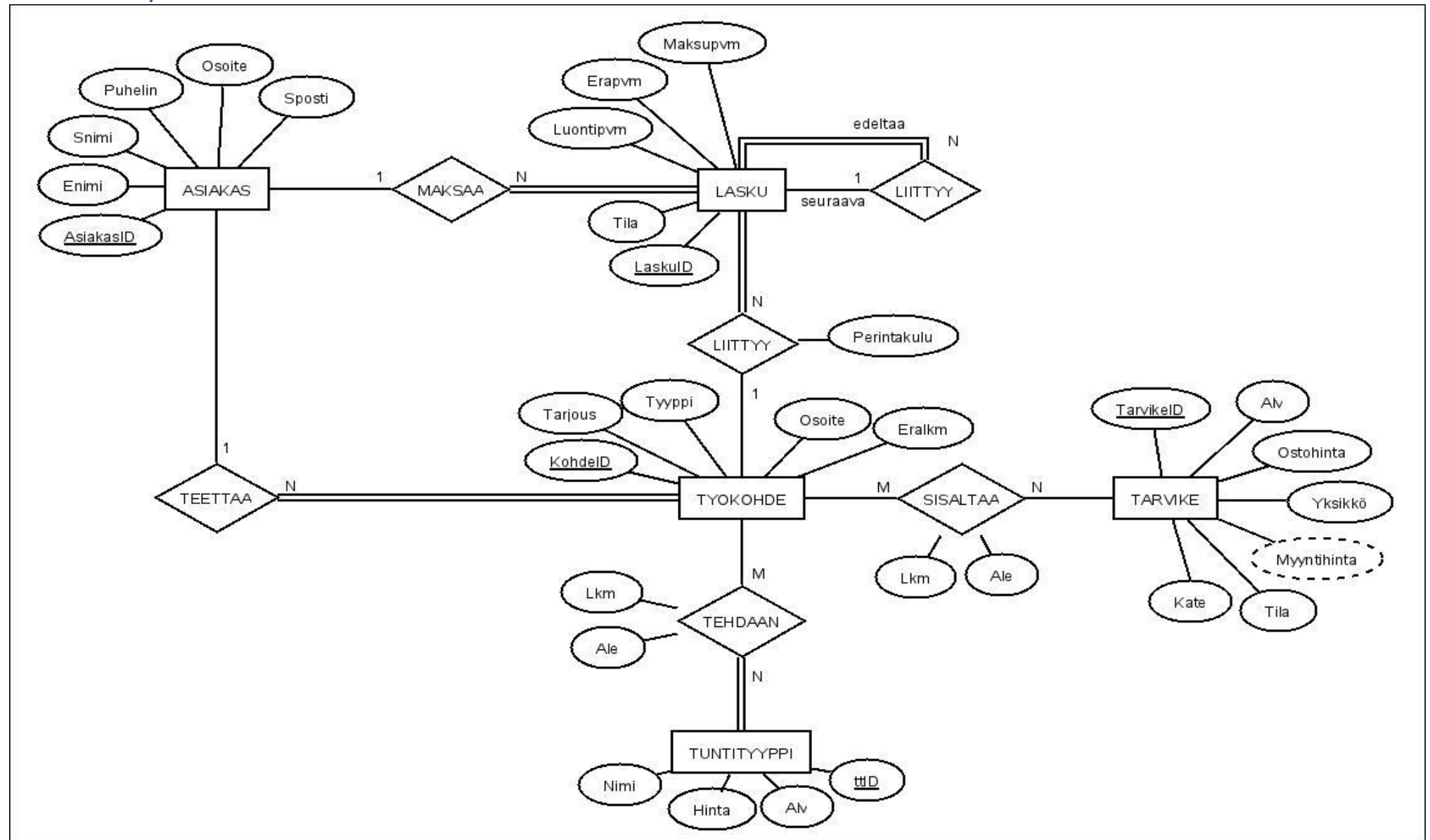
### **Mitä puutteita työhön jäi?**

Kaikki tehtävänannossa vaaditut toiminnallisuudet (tapahtumat ja raportit) saimme toteutettua ainakin jollakin tasolla. Parempi virheentarkastelu ja mahdollisuus peruuttaa virheellisiä toimintoja olisivat varmasti olleet käytännöllisiä.

Koodia olisi voinut nykyistä paremmin refaktoroida eli jakaa koodia useampaan luokkaan jolloin koodia olisi helpompi hahmottaa lyhyempinä kokonaisuuksina.

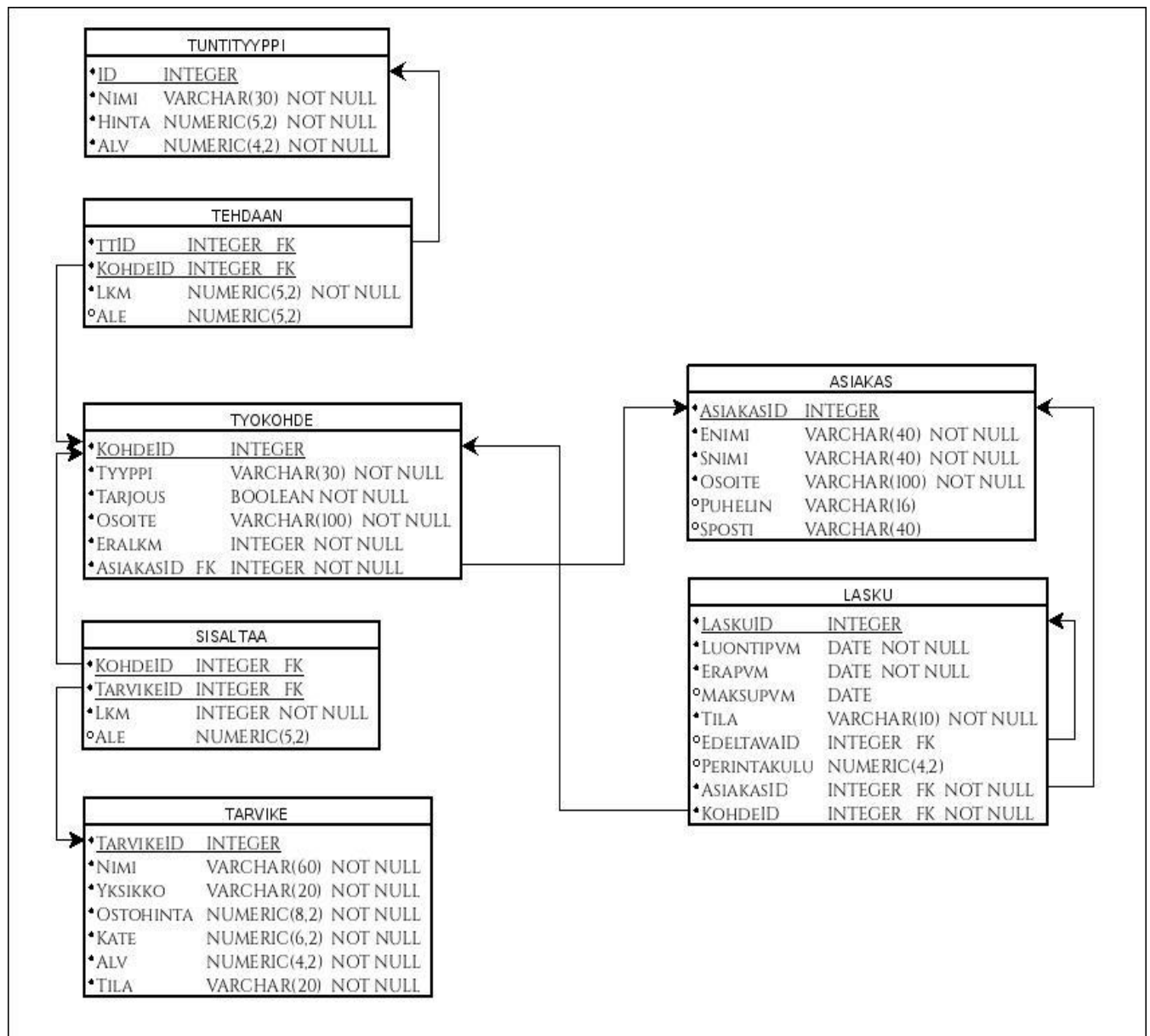
Tarjouksia ei pysty poistamaan järjestelmästä ja ne tukkivat näkymää etsittäessä viimeisimpiä uusia tarjouksia. Tarjousten hyväksyminen siis onnistuu, mutta jos luodaan tarjous joka ei etene työvaiheeseen, niin tarjous jää ohjelman listaan roikkumaan ikuisesti.

## 8. Päivitetty ER kaavio





## 9. Päivitetty tietokantakaavio



## 10. ER kaavion muunnos tietokantakaavioksi

Asiakkaalla voi olla useita työkohteita ja kukin työkohde liittyy aina yhteen asiakkaaseen. ASIAKAS ja TYOKOHDE relaatioiden välillä on siis 1:N suhde, joista N-puoli (työkohde) osallistuu täydellisesti. Tietokantakaaviossa TYOKOHDE-relaatioon lisätään siis ASIAKAS-relaation pääavain NOT NULL-määreellä varustettuna ja tästä tehdään viiteavain, joka viittaa ASIAKAS-relaation pääavaimeen.

Asiakkaalla voi olla maksettavanaan useita laskuja ja kuhunkin laskuun liittyy aina yksi asiakas, jonka maksettavaksi lasku kuuluu. ASIAKAS ja LASKU relaatioiden välinen suhde on myöskin 1:N, joista N-puoli (lasku) osallistuu täydellisesti. Tietokantakaaviossa LASKU-relaatioon lisätään ASIAKAS-relaation pääavain NOT NULL-määreellä varustettuna ja tästä tehdään viiteavain, joka viittaa ASIAKAS-relaation pääavaimeen. Työkohteeseen liittyy lasku aina, kun se on edennyt siihen pisteeseen, että työ aiotaan toteuttaa (sopimus on tehty). Mutta mikäli työkohteen tyyppi on vielä esimerkiksi urakkatarjous-vaiheessa, niin laskua ei vielä tässä kohtaa ole tehtynä. Toteutusvaiheessa työkohteeseen voi liittyä useampiakin laskuja, jos laskueriä on enemmän kuin yksi, tai joudutaan lähettämään muistutuslaskuja. Yksittäinen lasku liittyy aina ainoastaan yhteen työkohteeseen. LASKU ja TYOKOHDE relaatioiden välinen suhde on 1:N, joista N-puoli (lasku) osallistuu täydellisesti. Tietokantakaaviossa LASKU-relaatioon lisätään siis myöskin TYOKOHDE-relaation pääavain NOT NULL-määreellä varustettuna ja tästä tehdään viiteavain, joka viittaa TYOKOHDE-relaation pääavaimeen.

Lasku voi liittyä myös toisiin laskuihin, mikäli tarvitaan muistutus- tai karhulaskuja. Nämä laskut liittyvät siis näitä edeltäviin laskuihin, joita ei ole maksettu eräpäivään mennessä. Laskulla voi olla useampia muistutuslaskuja, mutta kullakin laskulla on enintään yksi edeltävä lasku, joka määrittää laskun tyyppin. LASKU-relaatio liittyy itseensä suhdetyypin ollessa 1:N, jossa N-puoli osallistuu täydellisesti. Tietokantakaaviossa tämä kuvataan siten, että LASKU-relaatioon lisätään sarake EdeltavaID josta tehdään viiteavain, joka viittaa relaation pääavaimeen. Viiteavain voi tässä kuitenkin olla NULL, koska ensimmäisillä laskuilla ei tietenkään ole viitettä niitä edeltävään laskuun.

Työhön voi sisältyä useita eri tarvikkeita ja tarvikkeita voidaan tarvita useissa eri työkohteissa. TYOKOHDE ja TARVIKE relaatioiden välinen suhde on M:N, joten muodostetaan suhdetaulu, joka sisältää molempien relaatioiden pääavaimet (KohdeID ja TarvikeID), joiden yhdistelmästä tulee suhdetaulun pääavain. KohdeID ja TarvikeID avaimista tehdään myös viiteavaimet, jotka viittaavat TYOKOHDE ja TARVIKE relaatioiden pääavaimiin.

Työhön voi liittyä myös useita eri tuntityyppejä (esim. suunnittelua, aputyötä...) joten myös TYOKOHDE ja TUNTITYYPPI relaatioiden välillä on M:N tyyppinen suhde. Näiden välille muodostetaan samalla tavalla suhdetaulu kuin yllä TYOKOHDE ja TARVIKE relaatioiden välille.

## 11. Attribuuttien arvoalueet ja rajoitukset

### ASIAKAS-relaatio:

Etunimi (ENIMI) ja sukunimi (SNIMI) attribuuteille on kullekin varattu pituudeksi 40 merkkiä, jonka uskotaan riittävän asiakkaan nimen tallettamiseksi. Osoitteelle on varattu 100 merkkiä, koska tähän tulisi mahtua myös postinumero ja postitoimipaikka. Puhelinnumero ja sähköpostiosoite kentät sallivat NULL-arvoja, koska näitä tietoja asiakkaan ei katsota pakolliseksi luovuttaa.

### TYOKOHDE-relaatio:

Tyyppi-sarake kuvastaa työn nykyistä tilaa (esim. urakkatarjous, urakkasopimus, jne.) ja tälle sarakkeelle riittää 30 merkkiä. Tarjous-sarake on BOOLEAN-tyyppinen ja kertoo siis tiedon siitä, onko työkohteesta annettu tarjous hyväksytty vai ei. Työkohteella on myöskin osoite ja tälle katsotaan asiakkaan osoitteen tavoin riittävän 100 merkkiä. Työkohteella on myös eralkm-sarake, joka kertoo maksun erien lukumäärän, eli kuinka monelle laskulle kokonaissumma jaetaan. Tämä arvo ei voi olla NULL ja tietokannan luontilauseissa oletusarvoksi asetetaan 1. AsiakasID-viiteavaimelle ei myöskään sallita NULL-arvoja, koska työkohteen teettää aina jokin asiakas.

### LASKU-relaatio:

Laskulla on luontipäivämäärä ja eräpäivämäärä (sarakkeet luontipvm ja erapvm), joille ei sallita NULL-arvoja. Oletuksena luontipäivämääräksi asetetaan senhetkinen päivämäärä ja eräpäivä asetetaan neljän viikon päähän. Maksupäivämäärä voi olla NULL, koska eihän laskua luodessa tiedetä, milloin lasku tullaan todellisuudessa maksamaan. Tila-sarake kuvaa laskun tilaa (kesken/valmis/siirtynyt) ja tämäkään ei saa olla NULL (oletusarvoksi asetetaan arvo "kesken"). Viiteavaimista EdeltavaID voi olla NULL, koska laskulla ei välttämättä ole edeltävää laskua. AsiakasID ja KohdeID viiteavaimet eivät salli NULL-arvoja, sillä laskulla on oltava asiakas, jolle lasku kuuluu maksettavaksi sekä lasku liittyy aina johonkin työkohteeseen. Laskuun liittyy myös tieto siitä, paljonko sen perintäkulut ovat (muistutuslaskuihin lisätään kuhunkin laskutuslisä 5€).

### TUNTITYYPPI-relaatio:

Tuntityypillä on oltava nimi (esim. suunnittelu, aputyö, jne.), joten Nimi-sarake ei saa olla NULL. Kullekin tuntityypille on myös määriteltävä hinta (NUMERIC (5,2) riittänee, koska yhden tunnin hinta ei koskaan ylitä 1000e) ja alv-prosentti (tämä kuvataan kokonaislukuna, esim. 24,00%, joten tässä NUMERIC (4,2) riittää) eli näiden sarakkeiden arvot eivät saa olla NULL. Alv-sarakkeen oletusarvoksi asetetaan taulussa 24,00, koska työn osalta sovelletaan yleistä arvonlisäverokantaa 24%.

### TEHDAAN-relaatio:

Lkm-sarake kuvaa sitä, kuinka monta tuntia kutakin tuntityyppiä tehdään työkohteelle. NUMERIC (5,2) kuvitellaan riittävän tässä, sillä sähkötöiden kestoksi yhtä työkohdetta kohden katsotaan riittävän alle 1000 tuntia. Lukumäärä ei voi olla NULL, eli jos jotakin tuntityyppiä tehdään kohteelle, niin tämä arvo on oltava > 0. Ale-sarake kuvastaa tunneista annetun alennusprosentin. Alennusprosentti on välillä 99,99% - 0,00% eli NUMERIC (5,2) riittää hyvin. Alennusta ei aina ole määriteltä, eli tämän sarakkeen arvo voi olla NULL.

### TARVIKE-relaatio:

Tarvikkeella on oltava nimi (60 merkkiä riittää pituudeksi) ja yksikkö (esim. kpl tai metri, tässä pituudeksi riittää 20 merkkiä) ja nämä sarakkeet eivät saa sisältää NULL-arvoja. Tarvikkeelle on myös tiedettävä sen sisäänostohinta (hintaa saa olla mitä tahansa 0 ja miljoonan väliltä) ja myynnistä haluttu katteen suuruus (tämä ilmoitetaan prosenttiyksiköinä ja koska katteeksi voidaan haluta yli 100%, arvoalueeksi määritellään NUMERIC (6,2)), joten näiden arvo ei saa myöskään olla NULL. Tarvikkeella on tila-attribuutti, joka kertoo siitä, onko tarvikkeen tilanne (esim. hinta) päivittynyt eli onko ID:tä vastaava tarvike ajantasainen vai vanhentunut. Tämä tieto on aina annettava, eli ei saa olla NULL. Tarvikkeille annetaan myös alv-prosentti (kuvataan kokonaislukuna, esim. 24,00%, joten tässä NUMERIC (4,2) riittää). Alv-sarake ei saa olla NULL, koska arvonlisävero on ilmoitettava aina.

**SISALTAA-relaatio:**

Vastaavasti, kuin TEHDAAN-relaation tapauksessa, Lkm-sarake kuvaa sitä montako yksikköä kutakin tarviketta liittyy yhteen työkohteeseen, eli lukumäärä ei voi olla NULL, kun jotakin tarviketta tarvitaan työkohteessa. Ale-sarake kuvastaa tarvikkeesta annetun alennusprosentin ja tähän riittää NUMERIC (5,2) koska alennusprosentiksi ei haluta antaa yli 100%. Tämän sarakkeen arvo voi olla NULL, koska alennusta ei välttämättä anneta.