Antti Pessa, Henri Hakkarainen ja Risto Meltaus

Tietokantaohjelmonti harjoitustyö

Vaihe 2: Toteutus

Ryhmä 12

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

**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

1. 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 eräpäiväksi asetetaan ensi vuoden tammikuun 1. päivä.

**Tuki samanaikaiselle käytölle**

Ohjelmassa on käytetty lukitusta tietokanta kyselyissä, jotka muokkaavat tietokannan tilaa. **Esimerkki**:

public void lisaaTarvike(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 tarvikeid, 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 tarvikeid = rs.getInt("tarvikeid");

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 tarvikeid = " + tarvikeid;

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 ovat tapahtuneet. Jos tapahtuu SQLException tehdään rollback.

**Käyttöliittymäominaisuudet**

**Testiaineisto**

**Lisätoiminnot**

* Asiakkaan lisäys onnistuu suoraan käyttöliittymästä
* Hakuominaisuus asiakkaille, tarvikkeille ja työkohteille
* Usean muistutuslaskun luominen kerralla
* Ohjataan käyttäjää ohjelman sisäisillä viesteillä
* Myös tuntitarjouksen voi laskuttaa monessa erässä

# Työnjako

Antti:

Henri:

Risto:

# Kuvaus toteutuksesta

# 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](https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html#isBlank()) funktiota, joka on saatavilla vasta Javan versiosta 11 eteenpäin ja siitä syystä ohjelman ajaminen vanhemmilla Javan versioilla päättyy virheeseen. Käyttöliittymä on toteutettu Javan Swing-kirjastolla ja siksi sen kääntäminen ja suorittaminen esim. SSH-yhteydellä tuni-palvelimelta käsin ei onnistu, koska ohjelma pyrkii avaamaan ulkoisen käyttöliittymäikkunan.

Koska toteuttamamme Java ohjelma täytyy ajaa paikalliselta koneelta käsin, myös ohjelman käyttämä tietokanta täytyy olla jonkin paikallisen portin takana, koska yhteyden ottaminen esim. dbstud2.sis.uta.fi-palvelimelle ei onnistu omalta koneelta käsin. Vaihtoehtoina tähän ovat joko PostgreSQL:n asentaminen omalle koneelle tai vaihtoehtoisesti Dockerin käyttö, jolloin projektia varten luomamme Docker-kontti pystyttää PostgreSQL-tietokannan ja ohjaa sen paikalliseen porttiin 5432 ja tähän pystytään ottamaan yhteys Java-ohjelmasta käsin.

**Huom.** Docker Desktop for Windowsin asennus edellyttää Windows 10 64-bit Pro, Enterprise tai Education käyttöjärjestelmää

## Miten ohjelmaa käytetään?

# Muutokset

Miksi varastotilanne poistettiin?

Miksi ei ole työsuoritus taulua?

Muutokset tietokantaan?

Miksi ei ole työkohde ja työsuoritus?

# 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.

Antti:

Risto:

**Mitä puutteita työhön jäi?**

Kaikki tehtävänannossa vaaditut toiminnallisuudet (tapahtumat ja raportit) saimme toteutettua ainakin jollakin tasolla, mutta ainahan näihin jää parantamisenkin varaa (esim. parempi virheentarkastelu ja mahdollisuus peruuttaa virheellisiä toimintoja).

# ER kaavio

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

# Tietokantakaavio

Kuva, joka sisältää kohteen näyttökuva

Kuvaus luotu automaattisesti

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 tyypin. 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.

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 määritelty, 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 (hinta 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.

# Tapahtumat

Työkohteen lisääminen asiakkaalle:

1. Lue käyttäjältä asiakkaan nimi.
2. Lue asiakkaan asiakasid taulusta asiakkaat.
3. Lue käyttäjältä kohteen tyyppi ja osoite.
4. Lisää tiedot tauluun työkohde.

Työkohteeseen käytettyjen resurssien päivittäminen:

1. Lue käyttäjältä työkohde.
2. Lue käyttäjältä kulunut tarvike ja määrä.
3. Hae tarvikkeen id taulusta tarvike.
4. Hae tarvike taulusta sisaltaa.
5. Jos arvoa ei ole, luodaan tauluun uusi rivi.
6. Lisätään käyttäjän syöttämä määrä olemassa olevaan määrään.
7. Lue käyttäjältä tehty työ ja määrä.
8. Hae tuntityyppi taulusta tehdään.
9. Jos arvoa ei ole, luodaan tauluun uusi rivi.
10. Lisätään käyttäjän syöttämä määrä olemassa olevaan määrään.

Tietokanta ei ole varsinaisesti tietoinen laskutuslisistä, vaan ne lasketaan laskun tasosta hakujen yhteydessä käyttöohjelmiston puolella. Seuraavissa laskutuslisä on kuitenkin selvyyden vuoksi mukana.

Muistutuslaskun muodostaminen maksamattomista ja eräpäivänsä ylittäneistä laskuista, joista ei ole aiemmin lähetetty muistutuslaskua:

1. Lue laskuid taulusta lasku tapauksille, joille pätee kaikki seuraavista:
2. Tila on maksamatta
3. Eräpäivä on aiemmin kuin nykyinen päivä
4. Laskun taso on 1 (edeltävien laskujen määrä on 0).
5. Luo vanhaan laskuun linkitetty rivi tauluun lasku.
6. Lisää laskutuslisä.
7. Muuta vanhan laskun tilaksi siirtynyt.

Karhulaskun muodostaminen maksamattomista ja eräpäivänsä ylittäneistä laskuista, joista on aiemmin lähetetty yksi muistutuslasku:

1. Lue laskuid taulusta lasku tapauksille, joille pätee kaikki seuraavista:
2. Tila on maksamatta
3. Eräpäivä on aiemmin kuin nykyinen päivä
4. Laskun taso on 2 (edeltävien laskujen määrä on 1).
5. Luo vanhaan laskuun linkitetty rivi tauluun lasku.
6. Lisää laskuun viivästyskorko.
7. Lisää laskutuslisä.
8. Muuta vanhan laskun tilaksi siirtynyt.

Tarvikelistauksen päivitys:

1. Hae tarvikkeen nimellä taulusta tarvike
2. Jos arvo löytyy, vaihda vanhan rivin tila-attribuutti arvoon ’vanhentunut’.
3. Lisää rivi tauluun tarvike.

# Raportit

Hinta-arvion muodostaminen:

1. Lisää uusi työkohde asiakkaalle xx, jonka tyypiksi tulee tarjous.
2. Lisää hinta-arvion sisältö (kts. esim. tapahtuma 2).
3. Hae työkohteella tunnit tehdaan taulusta ja laske työn hinta.
4. Hae työkohteella tarvikkeet sisaltaa taulusta ja laske tarvikkeiden hinta.
5. Laske yhteishinta.

Tuntityölasku tarvittavine tietoineen R2, R3:

1. Hae asiakasid taulusta työkohde.
2. Hae asiakkaan tiedot taulusta asiakas.
3. Hae työkohteeseen kuuluvat tarvikkeet ja lukumäärät taulusta sisaltaa.
4. Hae tarvikeid perusteella taulusta tarvike käytetyt tarvikkeet.
5. Rivisumman saa kertomalla ostohinnanhinnan katteella, alennusprosentilla (1- ale), alvilla ja lukumäärällä.
6. Hae tuntityypit ja lukumäärät taulusta tehdään.
7. Rivisumman saa kertomalla tuntiveloituksen, alennusprosentilla (1- ale), alvilla ja lukumäärällä.
8. Työn osuus eritelty tarvikkeista (kotitalousvähennyskelpoisuus).
9. Lasketaan rivisummat yhteen laskun loppusummaksi.

Urakkatarjous: katso R1: hinta-arvion muodostaminen, tyypiksi urakkatarjous.

Hyväksytty urakkatarjous:

1. Vaihda työkohteen tilaksi urakka.
2. Vaihda työkohteen erälukumääräksi 2.
3. Luo erälukumäärän verran uusia laskuja.
4. Aseta jälkimmäisten laskujen eräpäivä oikein.
5. Jokaisen laskun loppusumma on summa / erälukumäärä.

# Tietokannan luontilauseet (PostgreSQL)

# CREATE TABLE asiakas ( asiakasid SERIAL PRIMARY KEY, enimi VARCHAR(40) NOT NULL, snimi VARCHAR(40) NOT NULL, osoite VARCHAR(100) NOT NULL, puhelin VARCHAR(16), sposti VARCHAR(40) );

# CREATE TABLE tyokohde (

# kohdeid SERIAL PRIMARY KEY,

# asiakasid INTEGER NOT NULL,

# tyyppi VARCHAR(30) NOT NULL,

# tarjous BOOLEAN NOT NULL,

# osoite VARCHAR(100) NOT NULL,

# eralkm INTEGER DEFAULT 1 NOT NULL,

# FOREIGN KEY (asiakasid) REFERENCES asiakas(asiakasid),

# CHECK (LOWER(tyyppi) IN ('urakka', 'tunti')),

# CHECK (eralkm >= 0)

# );

# CREATE TABLE lasku (

# laskuid SERIAL PRIMARY KEY,

# asiakasid INTEGER NOT NULL,

# kohdeid INTEGER NOT NULL,

# edeltavaid INTEGER,

# luontipvm DATE DEFAULT CURRENT\_DATE NOT NULL,

# erapvm DATE DEFAULT CURRENT\_DATE + INTERVAL '28 day' NOT NULL,

# maksupvm DATE,

# tila VARCHAR(10) DEFAULT 'kesken' NOT NULL,

# perintakulu NUMERIC(4,2),

# FOREIGN KEY (asiakasid) REFERENCES asiakas(asiakasid),

# FOREIGN KEY (kohdeid) REFERENCES tyokohde(kohdeid),

# FOREIGN KEY (edeltavaid) REFERENCES lasku(laskuid),

# CHECK (LOWER(tila) IN ('kesken', 'siirtynyt', 'valmis')),

# CHECK (luontipvm < erapvm)

# );

# CREATE TABLE tarvike (

# tarvikeid SERIAL PRIMARY KEY,

# nimi VARCHAR(60) NOT NULL,

# yksikko VARCHAR(20) NOT NULL,

# ostohinta NUMERIC(8,2) NOT NULL,

# kate NUMERIC(6,2) DEFAULT 40 NOT NULL,

# alv NUMERIC(4,2) DEFAULT 24 NOT NULL,

# tila VARCHAR(20) DEFAULT 'käytössä' NOT NULL,

# CHECK (LOWER(tila) IN ('käytössä', 'vanhentunut'))

# );

# CREATE TABLE sisaltaa (

# kohdeid INTEGER,

# tarvikeid INTEGER,

# lkm INTEGER DEFAULT 0 NOT NULL,

# ale NUMERIC(5,2) DEFAULT 0,

# PRIMARY KEY (kohdeid, tarvikeid),

# FOREIGN KEY (kohdeid) REFERENCES tyokohde(kohdeid),

# FOREIGN KEY (tarvikeid) REFERENCES tarvike(tarvikeid),

# CHECK (lkm >= 0)

# );

# CREATE TABLE tuntityyppi (

# ttid SERIAL PRIMARY KEY,

# nimi VARCHAR(30) NOT NULL,

# hinta NUMERIC(5,2) NOT NULL,

# alv NUMERIC(4,2) DEFAULT 24 NOT NULL

# );

# CREATE TABLE tehdaan (

# ttid INTEGER,

# kohdeid INTEGER,

# lkm NUMERIC(5,2) DEFAULT 0 NOT NULL,

# ale NUMERIC(5,2) DEFAULT 0,

# PRIMARY KEY (ttid, kohdeid),

# FOREIGN KEY (ttid) REFERENCES tuntityyppi(ttid),

# FOREIGN KEY (kohdeid) REFERENCES tyokohde(kohdeid),

# CHECK (lkm >= 0)

# );

# Toteutusvälineet

Ohjelma toteutetaan Javalla. Versionhallintaan käytetään gittiä.