TIETOKANTAOHJELMOINTI KEVÄT 2017

Harjoitustyön dokumentaatio

Ryhmä 28

Juho Haapala, Topi Riikonen ja Vili-Veikko Rauhala

Haapala.Juho.M@student.uta.fi, Riikonen.Topi.M@student.uta.fi ja Rauhala.Vili.V@student.uta.fi

Ohjelman ominaisuudet

- Kanta ja data on määritelty erilliseen skeemaan.
- T1: Koska tehtävänannossa ei käsketty luoda uutta tehtävälistaa, vaan tehtävälistojen piti löytyä valmiiksi tietokannasta, katsomme Tapahtuma T1:n toteutetuksi.
- T2: Ohjelma luo session, joka toteuttaa tehtävälistan.
- R1: Raportti 1 on osana tätä dokumentaatiota.
- Harjoitustyön tietokanta sisältää testiaineistoa.
- Esimerkkitietokanta johdetaan ja esitetään näytöllä käyttäjälle jokaisen tehtävän yhteydessä. Myös käyttäjän tuottama tulostaulu sekä esimerkkivastauksen tuottama tulostaulu näytetään käyttäjälle.
- Ohjelmaa voi käyttää useampi henkilö samanaikaisesti.
- Ohjelmassa on yksinkertainen ja selkeä komentorivipohjainen käyttöliittymä.

Raportointi

R1: raportti, joka näyttää yksittäisen session tiedot: Käyttäjä ja onnistuneiden tehtävien lukumäärä.

```
SELECT kayttaja.e_nimi, kayttaja.s_nimi, COUNT(meniko_oikein) AS onnistuneita_tehtavia

FROM kayttaja INNER JOIN sessio ON kayttaja.o_id = sessio.o_id

    INNER JOIN yritys ON sessio.sessio_id = yritys.sessio_id

WHERE meniko_oikein = 't' AND yritys.sessio_id = 3

GROUP BY kayttaja.e_nimi, kayttaja.s_nimi;
```

```
e_nimi | s_nimi  | onnistuneita_tehtavia

--------+---------+----------------------

 Juho   | Haapala |                    3
```

R2: raportti, joka näyttää tiettyyn tehtävälistaan liittyvien sessioiden yhteenvetotiedot:

session nopein, hitain ja keskimääräinen suoritusaika.

Lukuisista yrityksistä huolimatta emme keksineet ratkaisua.

Jäsenten välinen työnjako

Ryhmänä koitimme jakaa tehtävät mahdollisimman tasapuolisesti. Teimme harjoitustyöstä suurimman osan yhdessä hyödyntäen näytönjako-ohjelmaa sekä VoIP-ohjelmistoa. Jaoimme myös harjoitustyön eri osioita jokaisen oman osaamisalueen mukaan.

Kuvaus toteutuksesta

Ohjelman käynnistyttyä pyydetään käyttäjää syöttämään etunimi, sukunimi, pääaine ja opiskelijanumeronsa. Tämän jälkeen ohjelma tarkistaa automaattisesti, onko kyseiselle opiskelijanumerolle jo luotu käyttäjä. Ohjelma luo uuden käyttäjän, jos opiskelijanumerolle ei ollut tallennettu aikaisemmin käyttäjätietoja. Ohjelma hakee opiskelijanumerolle kuuluvan käyttäjän o_id:n ja lähettää tämän session luovalle metodille.

Sessio luodaan automaattisesti saadun käyttäjä id:n, satunnaisesti valitun tehtävälista id:n sekä sessio id:n avulla, joka saadaan hakemalla kaikkien sessio-taulun rivien lukumäärä ja lisäämällä tähän yksi. Tämän jälkeen käyttäjälle näytetään tehtävälistaan kuuluvia tehtäviä järjestyksessä kolme kappaletta.

Käyttäjän tulee kirjoittaa näytölle tulevien esimerkkitietokannan taulujen perusteella vastaus ohjelman antamaan tehtävään. Käyttäjä kirjoittaa vastauksensa yhdelle riville ja syöttää sen ohjelmalle painamalla enter-näppäintä. Jos käyttäjän antama syöte tulostaa taulun, tulostetaan näytölle kyseisen taulun sisältö.

Ohjelma vertaa käyttäjän antaman vastauksen tuottamaa tulostaulua kyseiseen tehtävään liittyvän esimerkkivastauksen tuottamaan tauluun. Ohjelma tulostaa näytölle viestin, joka kertoo, menikö ratkaisu oikein vai väärin. Jos vastaus on väärin, tulostetaan mahdollinen oikean esimerkkivastauksen tuottama tulostaulu. Lisäksi ohjelma tulostaa näytölle jäljellä olevien yrityskertojen lukumäärän. Yrityskerran jälkeen ohjelma pyytää käyttäjää painamaan enter-näppäintä, siirtyäkseen seuraavaan yritykseen tai tehtävään.

Tehtävän suoritukseen liittyvien yrityksien tiedot tallennetaan yritys-tauluun. Yritys-taulu on heikko entiteetti, jolloin se saa avaimensa sessio id:stä, sessioon kuuluvan tehtävälistan tämänhetkisen tehtävän id:stä sekä yrityskerrasta kyseiseen tehtävään. Yrityksestä tallennetaan myös tehtävän aloitus- ja lopetusajankohdat, käyttäjän syöttämä vastaus sekä totuusarvo siitä oliko vastaus oikein.

Kun kaikki tehtävät on käyty läpi, ohjelma kertoo, montako tehtävää käyttäjä ratkaisi onnistuneesti ja hyvästelee käyttäjän.

Ohjelman käyttö

Kuinka ohjelma valmistellaan käyttöä varten?

Ohjelma valmistellaan siirtämällä lähdekooditiedosto sekä ajuri shell.sis.uta.fi-palvelimelle esimerkiksi käyttäen WinSCP-ohjelmaa. Tämän jälkeen otetaan yhteys esimerkiksi Puttya käyttämällä kyseiseen palvelimeen ja käännetään ohjelma komennolla:

```
javac TikoHarkka.java
```

Tämän jälkeen ohjelma on valmis käytettäväksi.

Miten ohjelmaa käytetään?

Ohjelma ajetaan käyttäen komentoa:

```
java -classpath postgresql-9.4.1212.jar:. TikoHarkka
```

Tämän jälkeen seurataan näytölle ilmestyviä ohjeita. Ohjelma pyytää syötteitä ja ne annetaan näppäimistön avulla. Enteriä painamalla syöte annetaan ohjelmalle.

Mahdolliset 1. vaiheen muutokset perusteluineen

Emme toteuttaneet graafista käyttöliittymää, koska yhteydenotto tietokantaan osoittautui turhan mutkikkaaksi. Sen sijaan toteutimme Javan avulla komentorivipohjaisella käyttöliittymällä toimivan ohjelman.

ER-kaavio koki myös hieman muutoksia, koska edellisen version sessio oli heikkona tyyppinä, joka esti käyttäjää suorittamasta samaa tehtävälistaa useammin kuin kerran. Myös muut ER-kaaviosta riippuvat osat kokivat muutoksia tämän johdosta.

Oma arvio työstä

Tuntemus harjoitustyöstä on, että se oli haastava, mutta mielestämme saimme aikaan suhteellisen toimivan kokonaisuuden. Osa metodeistamme olisi varmasti voitu toteuttaa paljon tehokkaamminkin.

Mikä oli vaikeaa

Suuri osa ajasta meni graafisen käyttöliittymän ja yhteyden luomisen kanssa. Päätimme lopuksi yksinkertaisen tekstikäyttöliittymän pysyäksemme aikataulussa, mikä osoittautui hyväksi ratkaisuksi tässä tilanteessa.

Harjoitustyön tehtävänanto jätti aika paljon asioita epäselviksi. Olkoonkin, että tekijät saivat tulkinnassa vapaat kädet, mikäli asioita ei ollut määritelty valmiiksi. Silti jotkin jopa ohjelman toimintaan liittyvät asiat jäivät hieman epäselviksi. Tämä hidasti aika paljon työn tekemistä ja suunnittelua sekä aiheutti todennäköisesti suurimman osan tapauksista, jossa ohjelma ei välttämättä toimi kuten tehtävänannon laatija on saattanut sen toivoa toimivan. Esimerkiksi haluttujen raporttien kohdalla oltaisiin voitu kertoa, halutaanko, että ohjelma luo raportteja pyynnöstä vai halutaanko raportit vain osaksi dokumentaatiota. Myös ohjelman ominaisuudet ovat jo kysymysmerkki; mikä kaikki katsotaan ominaisuuksiksi?

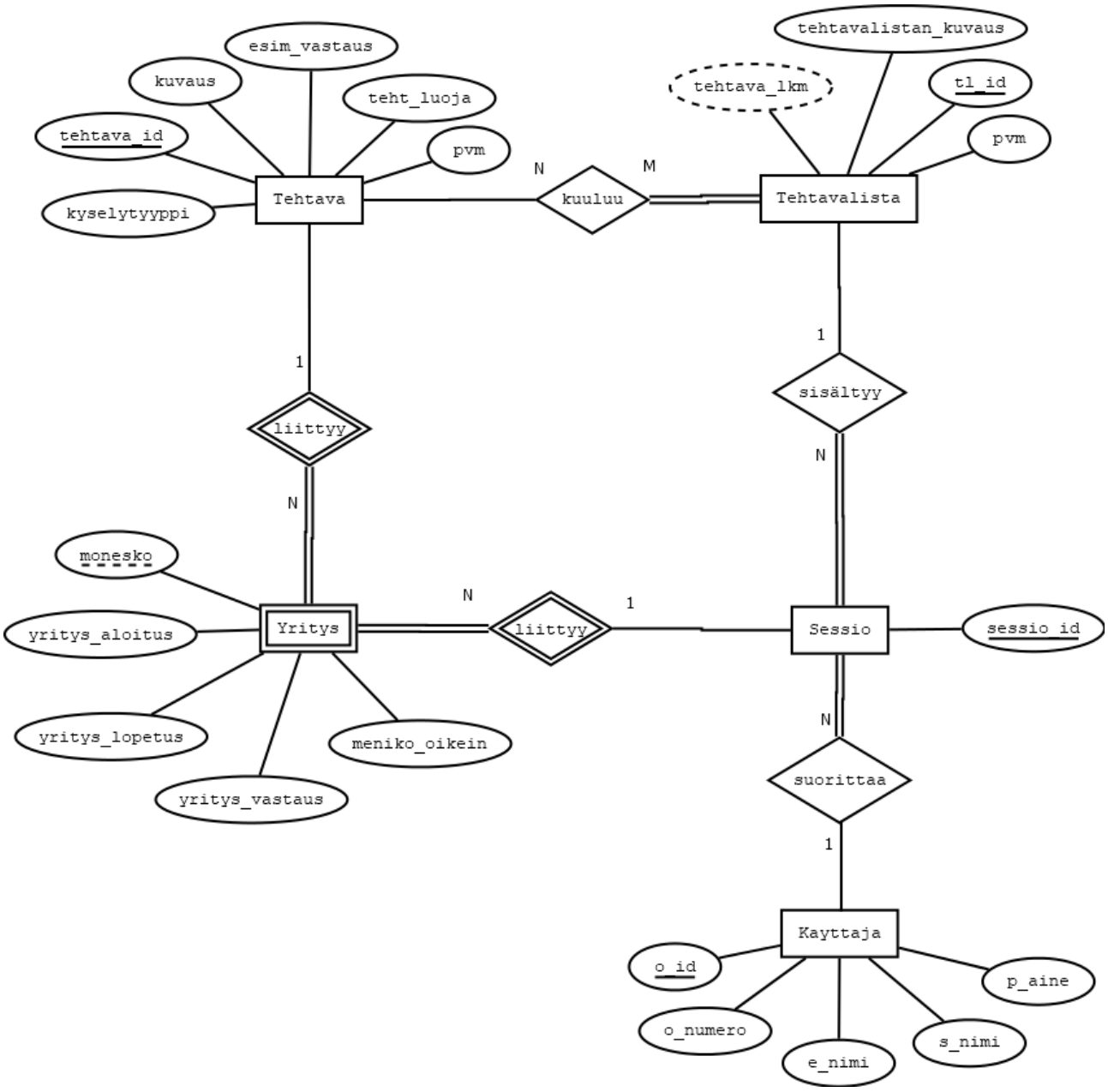
Vaikeaa oli myös ratkaista INSERT- ja DELETE-tehtävien tarkistus. Emme tarkalleen tienneet onko suotavaa/haluttavaa, että käyttäjä kykenee oikeasti poistamaan/lisäämään tietokannasta tietoja. Mielestämme tällainen toiminto voisi olla haitaksi koko tietokannalle, mikäli käyttäjä päättäisi joko tahallaan tai vahingossa poistaa jotain muuta kuin tehtävätaulujen rivejä.

Harjoitustyössä oli mukavan haastavaa päästä tekemään projektia ryhmänä.

Mitä puutteita työhön jäi?

INSERT- ja DELETE-lauseiden tarkistus ei luultavastikaan ole halutun tapainen. Tarkistus toimii vain vertaamalla käyttäjän syötettä valmiiksi annettuihin oikeisiin lauseisiin. Lauseet eivät myöskään tee konkreettisia muutoksia tietokantaan. Raportti 2 jäi puuttumaan, koska emme keksineet siihen ratkaisua.

Tietokannan ER-Kaavio



ER-Kaavion muunnos tietokantakaavioksi

Tehtava(tehtava_id, kuvaus, esim_vastaus, teht_luoja, pvm, kyselytyyppi)

Tehtavalista(tl_id, tehtavalista_kuvaus, pvm)

Tehtava_lkm(tl_id, tehtava_id)

Kuuluu(tehtava_id, tl_id)

Sessio(sessio_id)

Kayttaja(o_id, o_numero, e_nimi, s_nimi, p_aine)

Yritys(o_id, tehtava_id, monesko, yritys_aloitus, yritys_lopetus, yritys_vastaus, meniko_oikein)

Kuvaus kaavion tarvittavista näkymistä sekä muusta johdettavasta tiedosta

Yksittäisestä tehtävästä tallennetaan tehtäväkohtainen ID, joka toimii tehtävän pääavaimena. Tehtävään tallennetaan myös tehtävän kyselytyyppi, kuvaus, esimerkki oikeasta syötteestä jolla saadaan aikaan oikea vastaus, tehtävän luojan nimi sekä luomispäivämäärä.

Tehtävälista koostuu yksittäisistä tehtävistä. Koska tehtävälistassa voi olla 0-N määrä tehtäviä, ja yksittäisen tehtävän ei ole pakko kuulua mihinkään tehtävälistaan, on tehtävä- ja tehtävälista-taulujen välinen suhde N-M. Suhdetyypin takia on entiteettien välille tehtävä uusi taulu, joka sisältää tehtävän sekä tehtävälistan avaimen johon kyseinen tehtävä kuuluu.

Tehtävälistalla toimii pääavaimena sille määritelty ID. Tehtävälistaan tallennetaan myös tehtävälistan kuvaus, päivämäärä jolloin tämä on tehty, sekä moniarvoinen attribuutti tehtävien lukumäärälle kyseiseen tehtävälistaan. Tehtävien lukumäärä saadaan haettua ”Kuuluu” taulusta, jossa on määritelty tehtävien ja tehtävälistojen suhde.

Käyttäjästä tallennetaan opiskelijanumero, etunimi, sukunimi sekä pääaine. Käyttäjälle määritellään myös oma ID, joka saadaan hakemalla käyttäjä-taulusta käyttäjien lukumäärä, johon lisätään yksi. Uutta ID:tä ei määritellä, mikäli käyttäjän opiskelijanumeroon liittyy jo valmiiksi ID.

Käyttäjän ja tehtävälistan välillä on sessio. Sessio sisältää tiedon käyttäjästä, ja tälle tarjotusta tehtävälistasta. Sessiolla on myös oma pääavain. Käyttäjän syöttäessä tiedot ohjelmaan luodaan siis sessio, joka saa käyttäjälle luodun ID:n sekä satunnaisesti valitun tehtävälistan ID:n.

Sessioon liittyvien tehtävien kohdalla tallennetaan erilliseen tauluun tietoa tehtäviin kohdistuvista yrityskerroista. Yritys-tauluun tallennetaan viittaus session ID:hen,

kyseisen yrityksen tehtävän ID:hen, yrityskerran numero, yrityksen aloitusajankohta, lopetusajankohta, yrityksen vastaus ja tieto siitä tuottaako annettu vastaus oikean tuloksen. Yritys-taulu on siis heikko entiteetti, jonka omana heikkona avaimena toimii tehtävään liittyvä yrityskerta, itse tehtävän ID, sekä kyseiseen sessioon liittyvä ID jossa tehtävälistaa suoritetaan.

Tarvittavien tapahtumien kuvaukset

Käyttäjän tunnistus ja id:n luominen käyttäjälle:

Lue O_ID, O_NUMERO, E_NIMI, S_NIMI, P_AINE relaatiosta Kayttaja

E_NIMI = käyttäjän antama syöte.

S_NIMI = käyttäjän antama syöte.

P_AINE = käyttäjän antama syöte.

O_NUMERO = käyttäjän antama syöte.

O_ID = ohjelma luo käyttäjälle id:n, mikäli annettuun opiskelijanumeroon ei ennestään liity id:tä.

Tallenna O_NUMERO, E_NIMI, S_NIMI, P_AINE, O_ID relaatioon Kayttaja

Session luominen ja tehtävälistan hakeminen:

Lue TL_ID, O_ID, SESSIO_ID relaatiosta Sessio

Lue O_ID relaatiosta Kayttaja.

Lue TL_ID relaatiosta Tehtavalista.

SESSIO_ID = ohjelman toimesta luotu numerosarja, joka poikkeaa jo luoduista numerosarjoista.

O_ID = haetaan opiskelijan id relaatiosta Kayttaja.

TL_ID = haetaan sattumanvaraisesti valittu tehtävälistan id relaatiosta Tehtavalista.

Tallenna TL_ID, O_ID, SESSIO_ID relaatioon Sessio.

Käyttäjän antamien vastausten tietojen tallennus:

Lue SESSIO_ID relaatiosta Sessio.

Lue TEHTAVA_ID relaatiosta Tehtava.

Lue TEHTAVA_ID, SESSIO_ID, MONESKO, YRITYS_ALOITUS,

YRITYS_LOPETUS, YRITYS_VASTAUS, MENIKO_OIKEIN relaatiosta Yritys.

Lue SESSIO_ID relaatiosta Sessio.

Ohjelman avulla saadaan käyttäjän tekemän tehtävän yritysten tiedot: aloitus- ja lopetusajat, kuinka mones yritys, ja tieto siitä menikö vastaus oikein. Yrityskerran vastaus saadaan käyttäjän syötteenä. Session_id haetaan relaatiosta sessio ja tehtava_id haetaan relaatiosta tehtava.

Yritysten tiedot tallennetaan suoraan Yritys-relaatioon ohjelmassa.

Raporttien tietojen hakeminen:

Lue * relaatiosta Sessio.

Lue * relaatiosta Yritys

Lue * relaatiosta Kayttaja

R1 = Valitun session käyttäjä ja onnistuneiden tehtävien lukumäärä. Onnistuneiden tehtävien lukumäärä saadaan laskemalla yhteen ’t’ arvot MENIKÖ_OIKEIN attribuutista, jossa käyttäjän nimi on esimerkiksi ’Matti’.

R2 = Tiettyyn tehtävälistaan liittyvien sessioiden nopein, hitain ja keskimääräinen suoritusaika. Lasketaan YRITYS_ALOTUS ja YRITYS_LOPETUS attribuuttien avulla. Lasketaan ensin jokaiselle tehtävälistaan liittyvälle sessiolle suoritusaika, jonka jälkeen vertaillaan sessioiden aikoja keskenään ja valitaan nopein ja hitain aika sekä luodaan keskiarvo sessioiden suoritusajoista.

Tietokannan luontilauseet

```
CREATE TABLE tehtävä (
tehtava_id INT,
kuvaus VARCHAR(100) NOT NULL,
esim_vastaus VARCHAR(140) NOT NULL,
teht_luoja VARCHAR(25) NOT NULL,
pvm DATE NOT NULL,
PRIMARY KEY (tehtava_id));

CREATE TABLE tehtavalista (
tl_id INT,
tehtavalista_kuvaus VARCHAR(150) NOT NULL,
pvm DATE NOT NULL,
tehtava_lkm INT NOT NULL,
PRIMARY KEY (tl_id));

CREATE TABLE kuuluu (
tehtava_id INT,
tl_id INT,
FOREIGN KEY (tehtava_id) REFERENCES tehtava(tehtava_id),
FOREIGN KEY (tl_id) REFERENCES tehtavalista(tl_id),
PRIMARY KEY (tehtava_id, tl_id));

CREATE TABLE kayttaja (
opiskelija_id INT,
o_numero INT NOT NULL,
nimi VARCHAR(25) NOT NULL,
paa_aine VARCHAR(25) NOT NULL,
PRIMARY KEY (opiskelija_id));

CREATE TABLE sessio (
tl_id INT,
opiskelija_id INT,
```

```
sessio_id INT,
FOREIGN KEY (tl_id) REFERENCES tehtavalista(tl_id),
FOREIGN KEY (opiskelija_id) REFERENCES kayttaja(opiskelija_id),
PRIMARY KEY (tl_id, opiskelija_id, sessio_id));

CREATE TABLE Yritys (
tehtava_id INT,
sessio_id INT,
monesko INT,
yritys_aloitus TIMESTAMP NOT NULL,
yritys_lopetus TIMESTAMP NOT NULL,
yritys_vastaus VARCHAR(100),
meniko_oikein BOOLEAN NOT NULL,
FOREIGN KEY (tehtava_id) REFERENCES Tehtava(tehtava_id),
FOREIGN KEY (sessio_id) REFERENCES Sessio(sessio_id),
PRIMARY KEY (monesko));
```

Esimerkkikannan luontilauseet

```
CREATE TABLE opiskelijat (
nro INT,
nimi VARCHAR(5) NOT NULL,
p_aine VARCHAR(3) NOT NULL,
PRIMARY KEY (nro));

INSERT INTO opiskelijat (nro, nimi, p_aine)
VALUES (1, 'Maija', 'TKO');
INSERT INTO opiskelijat (nro, nimi, p_aine)
VALUES (2, 'Ville', 'TKO');
INSERT INTO opiskelijat (nro, nimi, p_aine)
VALUES (3, 'Kalle', 'VT');
INSERT INTO opiskelijat (nro, nimi, p_aine)
VALUES (4, 'Liisa', 'VT');

CREATE TABLE kurssit (
id INT,
nimi VARCHAR(5) NOT NULL,
opettaja VARCHAR(2) NOT NULL,
PRIMARY KEY (id));

INSERT INTO kurssit (id, nimi, opettaja)
VALUES (1, 'tkp', 'KI');
INSERT INTO kurssit (id, nimi, opettaja)
VALUES (2, 'oope', 'JL');
INSERT INTO kurssit (id, nimi, opettaja)
VALUES (3, 'tiko', 'MJ');

CREATE TABLE suoritukset (
k_id INT NOT NULL,
op_nro INT NOT NULL,
```

```
arvosana INT NOT NULL,
FOREIGN KEY (k_id) REFERENCES kurssit(id),
FOREIGN KEY (op_nro) REFERENCES opiskelijat(nro));

INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (1, 1, 5);
INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (1, 2, 4);
INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (1, 3, 2);
INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (2, 1, 5);
INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (2, 2, 3);
INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (2, 4, 3);
INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (3, 1, 5);
INSERT INTO suoritukset (k_id, op_nro, arvosana)
VALUES (3, 2, 4);
```

Esimerkkikyselyjen tulokset

```
-- KYSELY 1
SELECT opettaja
FROM kurssit;

 opettaja
----------
 KI
 JL
 MJ
```

```
-- KYSELY 2
SELECT nimi FROM opiskelijat
WHERE p_aine = 'TKO';

 nimi
-------
 Maija
 Ville
```

```
-- KYSELY 3
SELECT suoritukset.arvosana
FROM opiskelijat, suoritukset
WHERE opiskelijat.nro =
suoritukset.op_nro AND
opiskelijat.nimi = 'Ville';

 arvosana
----------
        4
        3
        4
```

```
-- KYSELY 4
INSERT INTO opiskelijat
VALUES(1234, 'Matti', 'VT');

INSERT 0 1
```

nro	nimi	p_aine
1	Maija	TKO
2	Ville	TKO
3	Kalle	VT
4	Liisa	VT
1234	Matti	VT

```
-- KYSELY 5
DELETE FROM opiskelijat WHERE
nro = 1234;

DELETE 1
```

nro	nimi	p_aine
1	Maija	TKO
2	Ville	TKO
3	Kalle	VT
4	Liisa	VT

Harjoitustyön toteutuksesta

Käytämme harjoitustyössä ohjelmointikielenä Javaa. Tavoitteemme on tehdä ohjelmasta graafinen käyttöliittymä, joka on mahdollista ajaa .jar tiedostosta. Attribuuttien rajoitukset on kuvattu taulujen luontilauseissa. Kaavion kuvauksessa on kerrottu ER-kaavion muunnoksesta tietokantakaavioksi.