# CS-A1550 Tietokannat

## Harjoitustyö, Osa 1

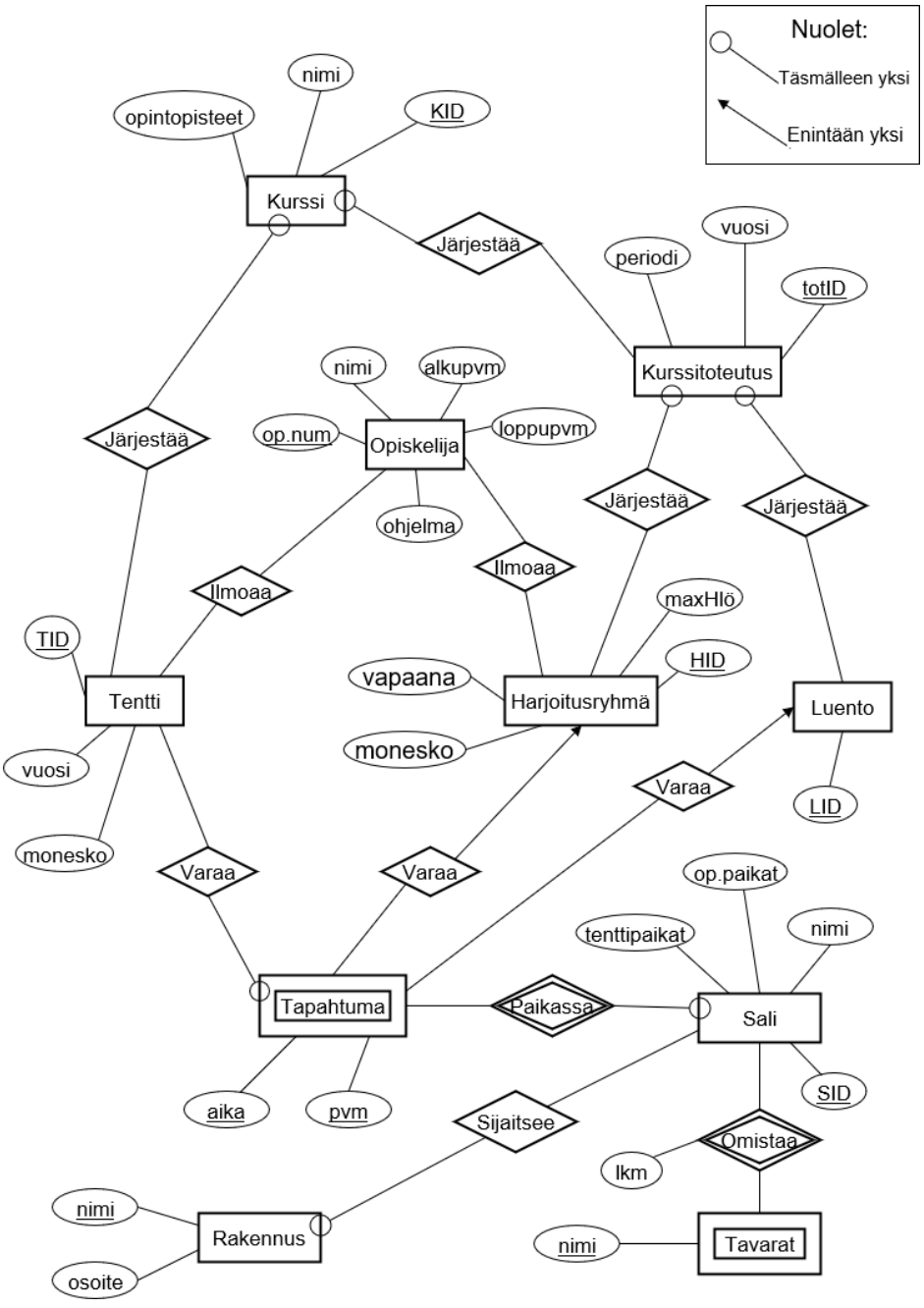
Kristian Arjas 473239
kristian.arjas@aalto.fi

Anna-Maija Valtavirta 68965S
anna-maija.valtavirta@aalto.fi

Palautettu **18. huhtikuuta 2018**

# Sisältö

# 1 ER-kaavio




Kuva 1: ER-kaavio.

### 1.1 ER-kaaviossa käytetyt lyhenteet

KID = kurssikoodi (esim. MS-A0101)
TID = tenttiID (esim. MS-A0101_2018_T02)
totID = kurssitoteutusID (esim. MS-A0101_2018_III)
HID = harjoitusryhmäID (esim. MS-A0101_2018_III_H03)
LID = luentoID (esim. MS-A0101_2018_III_L10)

## 2 Relaatiokaaviot

Opiskelija(op.num, nimi, ohjelma, alkupvm, loppupvm)
Kurssi(KID, opintopisteet, nimi)
Tentti(TID, vuosi, monesko)
Kurssitoteutus(totID, vuosi, periodi)
Harjoitusryhmä(HID maxHlö, monesko, vapaana)
Sali(SID, nimi, op.paikat, tenttipaikat)
Rakennus(nimi, osoite)
JärjestäTentti(KID, TID)
JärjestäToteutus(totID,KID)
JärjestäHarjoitus(totID, HID)
JärjestäLuento(totID, LID)
TenttiIlmo(op.num,TID)
HarjRyhmäIlmo(op.num,HID)
TenttiVaraus(TID, SID, alkuaika, pvm)
HarjRyhmäVaraus(HID, SID, alkuaika,pvm)
LuentoVaraus(LID, SID, alkuaika, pvm)
TavaratSijaitsee(tav.nimi, SID, lkm)

Luento-yksilöjoukolle ei ole tehty omaa relaatiokaaviota, sillä kaikki sen tiedot löytyvät LuentoVaraus-relaatiokaaviosta. Sama tavaroille: koska tavaroilla on tällä hetkellä ainoastaan nimi, niiden tiedot löytyvät TavaratSijaitsee-taulussa.

Olemme olettaneet, että jokainen tapahtuma järjestetään jossain tunnetuista saleista.

# 3 Funktionaaliset riippuvuudet

op.num $\rightarrow$ nimi, ohjelma, alkupvm, loppupvm
KID $\rightarrow$ opintopisteet, nimi
totID $\rightarrow$ vuosi, periodi, KID
TID $\rightarrow$ vuosi, monesko, KID
HID $\rightarrow$ maxHlö, monesko, vapaana, totID
LID $\rightarrow$ totID
SID $\rightarrow$ nimi, op.paikat, tenttipaikat, rak.nimi
rak.nimi $\rightarrow$ osoite
SID, tavaraNimi $\rightarrow$ lkm
SID, aika, pvm $\twoheadrightarrow$ KID

## 3.1 Anomaliat

Tällä hetkellä relaatiokaavioissa ei ole erikseen Tavarat-ryhmää, joten jos jotain tavaraa ei ole missään salissa, se katoaa tiedoista (poistoanomalia). Tavarat-ryhmä kannattaisi lisätä heti, jos haluttaisiin pitää kirjaa myös salittomista tavaroista tavaroilla olisi mitään muita ominaisuuksia kuin nimi. Samanlainen anomalia on myös Luento-ryhmässä: jos luennolle ei ole LuentoVarausta, sitä ei ole olemassa. Myös Luento-taulu kannattaisi tehdä, mikäli täytyisi pitää muistissa luentoja, joilla ei ole salivarausta tai jos luennoilla olisi lisäominaisuuksia (esim. erilaisia tavaravaatimuksia tai opettajia).

Relaatiokaavio ei ole erityisen altis päivitysanomalioille johtuen avainpohjaisesta suunnittelusta. Mikäli jotain attribuuttia tulisi muuttaa, riittää että sitä muutetaan kyseisen avaimen omaavassa taulussa. Tämä tieto sitten välittyy muille tietokannoille aina tarvittavien relaatioiden kautta. Mikäli haluamme muuttaa jotain avainta, esimerkiksi kurssikoodia, tulee avain muuttaa sekä itse kurssioliolla että tätä reunustavissa relaatioissa. Tietokanta on rakennettu siten, että avaimia sisältävät relaatiot ovat rajoittuneet yksilöjoukkoihin ja näitä ympäröiviin relaatioihin. Täten avainta muutettaessa tarvittavat yksilöjoukot ja relaatiot ovat helppo paikantaa ER-kaaviosta.

Mainittakoon, että ER-kaaviossa saliID näyttää ylettyvän useamman kuin yhden askeleen päähän Salista (varaa-relaatiot). Ylimääräistä toistoa on pyritty välttämään, ja merkittävin toisto aiheutuu ajan diskretisoinnista (selitetty myöhemmin).

## 3.2 Boyce-Codd normaalimuioto

Kun tarkastelemme edellä määriteltyjä funktionaalisia riippuvuuksia ja kun vertaamme näitä ER-kaavioon 1, havaitsemme että avaimen lisäksi jokaisella relaatiolla on vain tähän relaatioon liittyviä attribuutteja. Avaimet on pyritty valitsemaan uniikisti siten, että jokainen avain kuvaa jokaista uniikkia attribuuttijoukkoa. Toisin sanottuna jokaisen valitun avaimen sulkeuma on koko ryhmä, eli valitut avaimet ovat relaatioiden yliavaimia.

# 4 Toiminnallisuus

Tietokanta on pyritty suunnittelemaan siten, että uuden tiedon lisääminen olisi mahdollisimman yksinkertaista. Konkreettisia olioita (opiskelijat, kurssit, kurssitoteutus, tentit, luennot, harjoitusryhmät, tavarat, salit ja rakennukset) voidaan luoda tallentamalla uusi alkio vastaavaan tietokantaan uniikin avaimen kera (opiskelija numero, kurssikoodi yms.). Opiskelijoiden ilmoittautumiset ovat talletettu aina jokaista vastaavaan relaatioon. Esimerkiksi kurssi-ilmoittautumiset ovat tallennettu HarjRyhmäIlmo -relaatioon. Kyseisessä taulussa on avaimina opiskelijanumero ja harjoitusryhmän avain. Opiskelija on ilmoittautunut kurssille, jos opiskelijanumeroa ja kurssin jotain harjoitusryhmää vastaava avainpari löytyy tietokannasta. Kun tiedot on tallennettu, pystytään näistä relaatioista löytämään kaikki kurssitoteutukset ja tentit, joille opiskelija on ilmoittautunut, sekä kaikki johonkin kurssitoteutukseen tai tenttiin ilmoittautuneet opiskelijat.

Mainittakoon vielä, että ilmoittautuessa ohjelman tulisi pitää huolta vapaiden paikkojen määrästä. Tästä syystä Harjoitusryhmä -relaatiossa on maxHlö- ja vapaana-parametrit. Kun instanssi luodaan, nämä arvot ovat yhtäsuuret. Aina kun opiskelija ilmoittautuu, vähennetään vapaana olevien lukumäärää yhdellä. Ryhmän ilmoittautuminen sulkeutuu kun vapaana on saavuttanut arvon 0.

Tietokannan kalenteri abstraktiin perustuu Tapahtuma-luokkaan. Tämä on heikko yksilöluokka, joka koostuu ajasta (päivämäärä ja kellonaika) ja paikasta (saliID). Itse varaukset ovat olemassa kukin omaa tyyppiään vastaavassa varaustietokannassa (HarjRyhmäVaraus, TenttiVaraus, LuentoVaraus). Nämä taulut koostuvat tapahtumien avaimista jotka määräytyvät luvun 2 relaatiokaavioiden mukaan. Jokainen avainyhdistelmä sisältää tiedot mitä, milloin ja missä.

Tietokanta on rakennettu oletukselle, että aika on diskretisoitu tarpeeksi pieniin paloihin. Tämä tarkoittaa, että pidemmät tapahtumat varataan luomalla lukuisia saman tapahtuman varausolioita samaan tilaan peräkkäisin aloitusajoin. Esimerkiksi, mikäli aikayksikkö olisi tunnin mittainen, kahden tunnin luento varattaisiin luomalla kaksi saman luentoID:n omaavaa saman salin varausta samana päivänä peräkkäisin alkamisajoin.

Jos olemme kiinnostuneita järjestettävistä tapahtumista (luennot, kurssit, harjoitusryhmät tai tentit) tulee meidän tarkastella kyseisen tapahtumatyypin varaustaulukkoa kyseisellä aikavälillä. Mikäli meitä kiinnostaa esimerkiksi mitä kursseja järjestetään jollain aikavälillä, tulee meidän etsiä kursseihin liittyvät HID:t, LID:t ja TID:t varaustauluista kyseiseltä aikaväliltä ja yhdistää ne sitten oikeisiin KID:eihin vastaavien Järjestä-relaatioiden kautta.

Vastaavasti, mikäli olemme kiinnostuneita siitä, mihin tarkoitukseen sa-

lia on käytetty (jos on käytetty) jonain kyseisenä ajanhetkenä, tulee meidän tarkastella näitä varaustaulujen unionia aika- ja paikka-avaimilla. Mikäli sali on ollut kyseisellä hetkellä käytössä, haun pitäisi palauttaa jokin ID. Mainittakoon, että ohjelmiston kannalta sillä ei ole merkitystä, että koska sitä käytetään. Ohjelmisto ei tee eroa tulevaisuuden ja menneisyyden tapahtumien jäljellä.

Jos olemme kiinnostuneita jostain spesifimmästä, esimerkiksi siitä että mitä tenttejä jollain kurssilla on jollain aikavälillä, tulee meidän taas tutkia kyseistä varaustaulukkoa (tässä tapauksessa TenttiVaraus) ja järjestystaulukkoa. Järjestystaulukosta meidän tulee löytää jotain kurssia vastavat kaikki tentit, joita lähdemme sitten kaivamaan tenttivaraustaulukosta asetetuilla aikarajoilla.

Kurssien järjestysajankohdan selvittämiseen on myös yleisempi metodi. Edelläkuvattujen ajankohtienselvitysmetodien lisäksi olemme tallettaneet kurssitoteutuksen yhteen periodin ja vuoden aina kyseiselle toteutukselle. Jotta löytäisimme jonkin tietyn kurssin kaikki toteutuskerrat, tulee ne vielä lisäksi selvittä JärjestäToteutus -relaatiosta.

Kurssikertaan liittyvät luennot voimme löytää LuentoVaraus -taulusta käyttämällä halutun toteutuksen ID:tä.

Voimme selvittää kyseisen kurssikerran harjoitusryhmät ja näiden sijainnit ja kokoontumisajat hakemalla tietoa relaatioista harjRyhmäVaraus avaimella toteutusID.

Salin, jossa on vähintään haluttu määrä vapaita paikkoja, löytäminen on hieman mutkikkaampi operaatio, mutta kuitenkin mahdollinen. Ohjelman tulisi aloittaa tämä haku katsomalla kaikkia tapahtumia kyseisenä ajankohtana. Haku suoritetaan kolmessa varaustaulussa ja tämän haun tulisi antaa jokaiselle salille joko tyhjän tapahtuman (tapahtumaa ei ole) tai jonkin edellä keskustelluista kolmesta tapahtumatyypistä. Tämän jälkeen ohjelman tulisi selvittää jokaisen salin paikkojen määrä Sali -taulusta. Vapaat paikat laskettaisiin tämän jälkeen joko palauttamalla maksimipaikkojen lukumäärä (ei tapahtumaa) tai vähentämällä maksimipaikoista tapahtumaan varattujen paikkojen määrä (tieto löytyy totID:n kautta). Lopuksi palautetaan kaikki SID:t, jotka täyttävät annetun paikkamääräehdon.