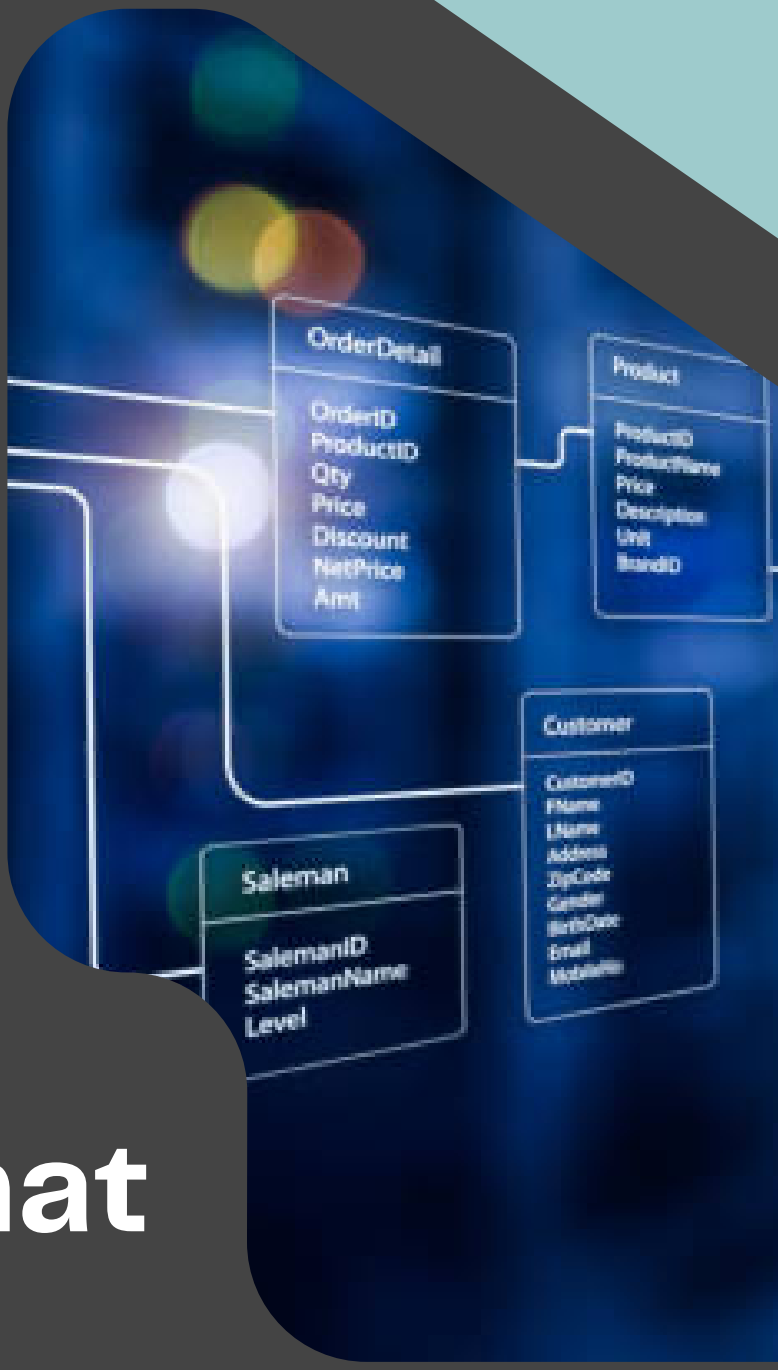


KEVÄT 2023

Tietokannat

Projekti: Osat 1 ja 2

Tekijät:

Tatu Rouhiainen - tatu.rouhiainen@aalto.fi

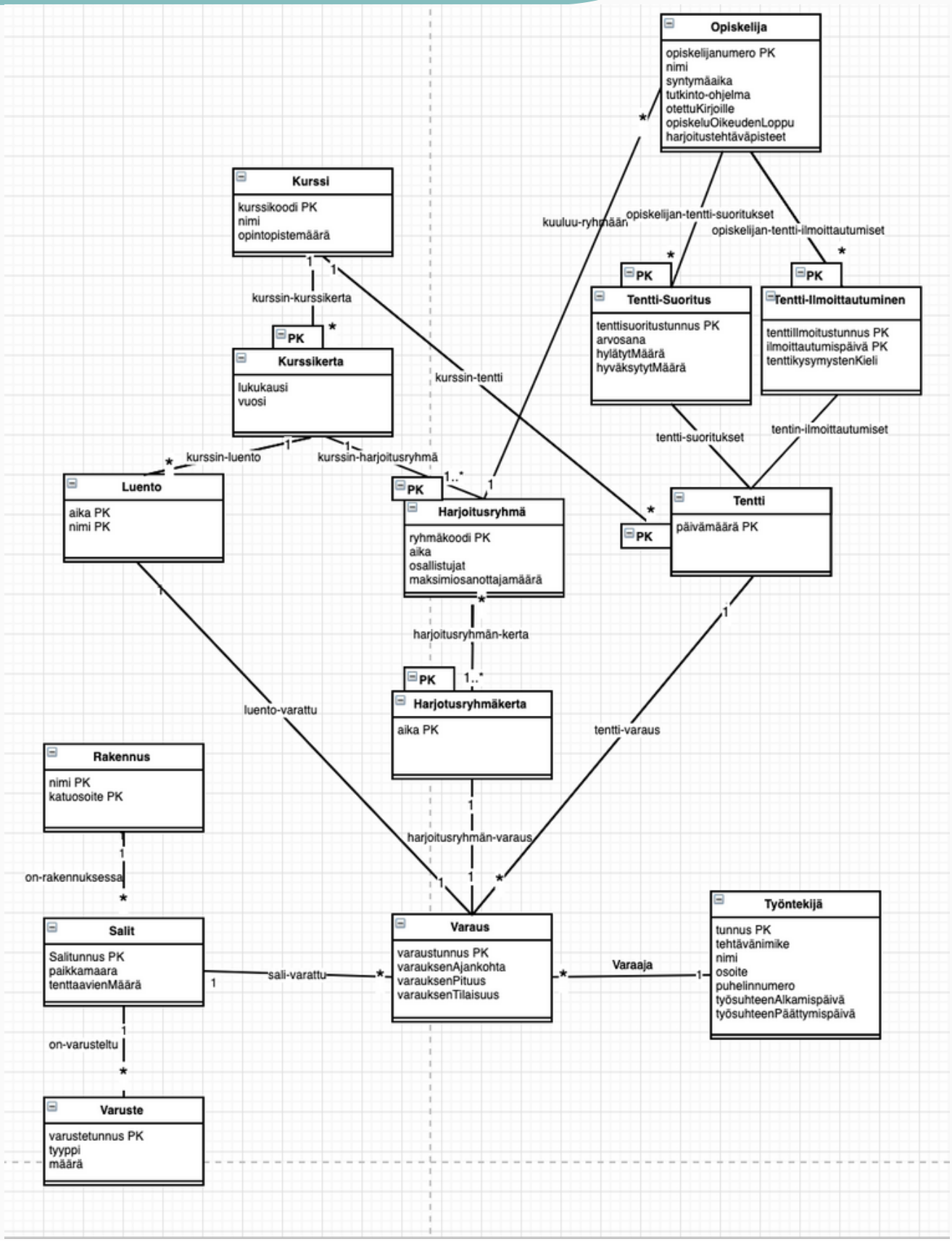
Axel Andersson - axel.w.andersson@aalto.fi

Benjamin Hadaya -benjamin.hadaya@aalto.fi

Aalto Yliopisto

Sisällysluettelo

UML-kaavio


Opiskelija
opiskelijanumero PK
nimi
syntymäaika
tutkinto-ohjelma
otettuKirjoille
opiskeluOikeudenLoppu
harjoitustehtäväpisteet
Kurssi
kurssikoodi PK
nimi
opintopistemäärä
kuuluu-ryhmään
opiskelijan-tentti-suoritukset
opiskelijan-tentti-ilmoittautumiset
kurssin-kurssikerta
PK
Kurssikerta
lukukausi
vuosi
PK
Tentti-Suoritus
tenttisuoritustunnus PK
arvosana
hylätytMäärä
hyväksytytMäärä
PK
Tentti-Ilmoittautuminen
tenttiIlmoitustunnus PK
ilmoittautumispäivä PK
tenttikysymystenKieli
kurssin-tentti
tentti-suoritukset
tentin-ilmoittautumiset
kurssin-luento
kurssin-harjoitusryhmä
Luento
aika PK
nimi PK
PK
Harjoitusryhmä
ryhmäkoodi PK
aika
osallistujat
maksimiosanottajamäärä
PK
Tentti
päivämäärä PK
harjoitusryhmän-kerta
PK
Harjotusryhmäkerta
aika PK
luento-varattu
tentti-varaus
Rakennus
nimi PK
katuosoite PK
harjoitusryhmän-varaus
on-rakennuksessa
Salit
Salitunnus PK
paikkamaara
tenttaavienMäärä
sali-varattu
Varaus
varaustunnus PK
varauksenAjankohta
varauksenPituus
varauksenTilaisuus
Varaaja
Työntekijä
tunnus PK
tehtävänimike
nimi
osoite
puhelinnumero
työsuhteenAlkamispäivä
työsuhteenPäättymispäivä
on-varusteltu
Varuste
varustetunnus PK
tyyppi
määrä

UML-kaaviota vastaavat relaatiokaaviot

Kurssit(kurssikoodi, nimi, opintopistemäärä)

Kurssikerrat(kurssikoodi, lukukausi, vuosi)

Harjoitusryhmät(kurssikoodi, ryhmäkoodi, aika, osallistujat, maksimiosanottajamäärä)

Harjoitusryhmäkerrat(kurssikoodi, ryhmäkoodi, aika)

Luennot(aika, nimi)

Tentit(kurssikoodi, päivämäärä)

Opiskelijat(opiskelijanumero, nimi, syntymäaika, tutkinto-ohjelma, otettuKirjoille, opiskeluOikeudenLoppu, harjoitustehtäväpisteet)

Tentti-ilmoittautumiset(opiskelijanumero, tenttiIlmoitustunnus, ilmoittautumispäivä, tenttikysymystenKieli)

Tenttisuoritukset(opiskelijanumero, tenttisuoritustunnus, arvosana, hylätytMäärä, hyväksyttyMäärä)

Työntekijät(tunnus, tehtävänimike, nimi, osoite, puhelinnumero, työsuhteenAlkamispäivä, työsuhteenPäättymispäivä)

Rakennukset(nimi, katuosoite)

Salit(salitunnus, paikkamäärä, tenttaavienMäärä)

Varusteet(varustetunnus, tyyppi, määrä)

Varaus(varaustunnus, varauksenAjankohta, varauksenPituus, varauksenTilaisuus)

UML-kaaviota vastaavat relaatiokaaviot

Harjoitusryhmän-varaus(kurssikoodi, ryhmäkoodi, aika, varaustunnus)

KurssinLuento(aika, nimi, kurssikoodi)

LuentoVarattu(aika, nimi, varaustunnus)

KuuluuRyhmään(opiskelijanumero, kurssikoodi, ryhmäkoodi)

suorittaa-tentin(opiskelijanumero, tenttisuoritustunnus, kurssikoodi, päivämäärä)

ilmoittautuu-tenttiin(opiskelijanumero, tenttiIlmoitustunnus, ilmoittautumispäivä, kurssikoodi, tentinPäivämäärä)

Tentti-varaus(varaustunnus, kurssikoodi, päivämäärä)

Varaaja(varaustunnus, työntekijätunnus)

Sali-varattu(varaustunnus, Salitunnus)

On-Rakennuksessa(Salitunnus, rakennuksenNimi, katuosoite)

On-Varustettu(varustetunnus, salitunnus)

Selostus Ratkaisusta

Kurssit ja kurssikerrat

Kurssit ja kurssikerrat
Tietokannassa on tiedot siitä, mitä kursseja yliopiston opetusohjelmassa on. Kursseista on tiedossa kurssikoodi, kurssin nimi, sekä kurssikohtainen opintopistemäärä. Avain attribuuttina toimii jokaiselle kurssille yksilöity kurssikoodi. Sama kurssi voidaan järjestää useita kertoja joko samana lukukautena tai eri lukukausien aikana. Tästä johtuen luokka Kurssikerta käyttää vain luokan Kurssit avain attribuuttia kurssikoodi. Kurssikerroista tallennetaan lukukausi ja vuosi. Edellä mainitut luokat pystyvät tallentamaan kaikki tarvittavat tiedot kursseista ja kurssikerroista. Yhteen kurssiin voi liittyä mieletön määrä kurssikertoja.

Tietokannassa voi selvittää mitä kursseja järjestetään haluttuna ajankohtana. Halutun ajankohdan ja kurssikerran lukukausi ja vuosi attribuutteja vertaamalla voidaan selivttää ensiksi mihin lukukauteen ja vuoteen ajankohta sijoittuu, ja selivittää kaikki kyseisellä ajanhetkellä pyörivät kurssikerrat.

Tietokannassa säilyy myös kurssien ja kurssikertojen historialliset tiedot sekä tulevat tiedot aina niin pitkälle kun opetusohjelma on suunniteltu, eli on mahdollista selvittää koska haluttu kurssi on järjestetty tai koska haluttu kurssi tullaan järjestämään.

Luennot, Harjoitusryhmät ja Tentit

Yhteen kurssikertaan (määrättyyn kurssin toteutukseen) liittyy luennot ja harjoitusryhmät, mutta tentit suoraan itse kurssiin. Tämä selittyy sillä, että tentit ovat aina kurssikohtaisia. Useampi määrätyn kurssin toteutus voi käyttää samaa tenttiä kurssin arvioimiseen. Harjoitusryhmät ja luennot ovat puolestaan määrätyn kurssin toteutus kohtaisia.

Luennoista tallennetaan aika ja nimi, jotka toimivat yhdessä luokan avain attribuutteina. Harjoitusryhmistä tallennetaan ryhmäkoodi, joka toimii avain attribuuttina, sekä ajankohta ja maksimi osanottajamäärä. Tenteistä tallennetaan päivämäärä.

Harjoitusryhmillä voi olla maksimiosanottajamäärät, ja opiskelija voi ilmoittautua ainoastaan sellaiseen harjoitusryhmään, jossa on tilaa. Tietokannassa voidaan vertailla kyseisen ajanhetkellä olevaa osallistujamäärää maksimi osanottajamäärän, kun opiskelija olisi ilmoittautumassa harjoitusryhmään. Tietokannassa on mahdollista selvittää mitä harjoitusryhmiä haluttuun kurssikertaan liittyy, sekä milloin harjoitusryhmäkertoja järjestetään (aika attribuutilla) tai missä harjoitusryhmäkerrat järjestetään (linkit Varausten kautta Saleihin).

Selostus Ratkaisusta

Tietokannasta voi siis selvittää mitä luentoja haluttuun kurssikertaan liittyy. Luentoja voi olla yhdellä kurssikerralla mieletön määrä. Joillakin kurssikerroilla ei välttämättä ole luentoja ollenkaan. Yhdellä kurssikerralla on vähintään yksi harjoitusryhmä, sillä opiskelija ilmoittautuu kurssille ilmoittautumalla harjoitusryhmään. Yksinkertaisuuden vuoksi UML-kaaviossa oletetaan, että jokaisella kurssikerralla on vähintään yksi harjoitusryhmä - muuten kurssille ei voisi ilmoittautua. Yhteen kurssiin voi liittyä useita tenttimahdollisuuksia (tentti heti kurssikerran jälkeen ja rästitentit). Jokaisella kurssilla ei kuitenkaan välttämättä ole tenttiä.

Tietokannasta voi etsiä myös kaikki harjoitusryhmät, joissa on tilaa. Jos attribuutti osallistujamäärä ja maksimiosanottajamäärää vertailee siten, että etsitään kaikki harjoitusryhmät, joissa osallistujamäärä on pienempi kuin maksimi, silloin tuloksena palautuu kaikki harjoitusryhmät joissa on tilaa.

Opiskelijat ja Tentit

Opiskelijat voivat ilmoittautua kurssille ilmoittautumalla harjoitusryhmään (luokka: Harjoitusryhmät), tai ilmottautumalla tenttiin (luokka: Tentti-Ilmottautumiset). Harjoitusryhmät, esim. H01, H02, H03, kokoontuvat kurssikerran aikana useampia kertoja, joten yhdellä harjoitusryhmällä on useita harjoitusryhmäkertoja. Harjoitusryhmäkertoja on kuitenkin vähintään yksi - muuten harjoitusryhmä olisi järjetön. Harjoitusryhmäkerroista tallennetaan harjoitusryhmänkerran aika, ja se toimii avain attribuuttina yhdessä harjoitusryhmäkoodin kanssa. Samalla harjoitusryhmällä ei ole päällekkäisiä aikoja.

Opiskelijoista tallennetaan hänet yksilöivä opiskelijanumero, nimi, syntymäaika, tutkinto-ohjelma, milloin opiskelija on otettu kirjoille, milloin opiskeluoikeus loppuu ja hänen kurssikerta kohtaiset harjoitustehtäväpisteet. Tietokannasta on siis mahdollista selvittää kuinka monta harjoitustehtäväpistettä kullakin opiskelijalla on kutakin kurssikertaa kohtaan.

Opiskelijan pitää ilmoittautua erikseen sellaiseen tenttiin, johon hän haluaa osallistua. Tästä syystä tentti-ilmoittautumisia voi olla mieletön määrä. Tentti-ilmoittautumisista tallennetaan tietokannassa ilmoittautumispäivä ja tenttikysymysten kielivalinta. Avain attribuutteina toimii opiskelijanumero, ilmoittautumispäivä sekä tenteistä tulevat avain attribuutit tentin päivämäärä, sekä kurssikoodi.

Selostus Ratkaisusta

Opiskelija suorittaa kurssin suorittamalla kurssin tentin. Tietokannassa pidetään yllä tietoa siitä, mihin tentteihin opiskelija on osallistunut ja minkä arvosanan hän on niistä saanut (myös niistä tenteistä, joissa opiskelijan arvosana on ollut hylätty). Opiskelijat linkittyvät tentteihin Tentti-Suoritukset luokan kautta, johon tallennetaan arvosanan lisäksi hylättyjen, että hyväksyttyjen tenttien lukumäärä. Tentti suoritusten ja ilmoittautumisten erottelu kahteen eri luokkaan varmistaa sen, että osallistuminen ei ole sama kuin tenttiin ilmoittautuminen, koska opiskelija voi jättää tulematta tenttiin, johon hän on ilmoittautunut. Näin opiskelijalle voidaan kirjata myös tentin suoritus, ja selvittää minkä kurssien suorituksia yksittäisellä opiskelijalla on. Tentti suoritusten avain attribuuttina toimii jokaisen yksittäisen tenttisuorituksen yksilöivä tenttisuoritus tunnus sekä opiskelijoista tuleva opiskelijanumero.

Tentti suorituksia voi olla olematta, tai niitä voi olla mieletön määrä, sillä hylättyjä yrityksiä voi olla mieletön määrä. Opiskelija ei voi kuitenkaan korottaa hyväksyttyä tenttiä kuin kerran, joten tietokannassa voidaan vertailla attribuuttia “hyväksytytMäärä” tai käyttää esimerkiksi COUNT MySQL funktiota, jolla voidaan varmistaa, että hyväksyttyjen tenttien määrä ei ylitä kahta. Jos ylittää, opiskelija ei voi ilmoittautua enää kyseisen kurssin tenttiin.

Tietokannassa voi selvittää mitä tenttejä halutulla kurssilla on halutulla aikavälillä, selvittämällä Tentit luokasta kaikki halutun kurssikoodin tentit ja vertaamalla tenttien päivämääriä, jotka asettuvat halutulle aikavälille.

Tietokannasta voi hakea Opiskelijoista kulkevien linkkien kautta harjoitusryhmiin ja tentti-ilmoittautumisiin, että mitä opiskelijoita haluttuun harjoitusryhmään, tenttiin tai kurssiin osallistuu tai on osallistunut.

Rakennukset, Salit ja Varusteet

Yliopistolla on rakennuksia, joissa on opetuskäyttöön tarkoitettuja tiloja eli saleja. Rakennuksista tallennetaan niiden nimi ja katuosoite, jotka toimivat yhdessä luokan avain attribuutteina. Yhdellä rakennuksella voi olla useita saleja, jotka yksilöidään salitunnuksilla.

Saleista tallennetaan, montako paikkaa niissä on opiskelijoille sekä kuinka monta tenttiijää saliin mahtuu tenttitilaisuudessa. Tenttaavien määrä voi olla pienempi pelkästään siitä syystä, että tentissä opiskelijat eivät voi istua liian lähellä toisiaan. Yhdellä salilla voi olla lukuisia eri varusteita.

Selostus Ratkaisusta

Varusteista tallennetaan varustetunnus avain attribuuttina, sekä varusteen tyyppi sekä määrä. Tämä mahdollistaa varusteiden yksilöinnin, määrän ilmaisemisen sekä varusteen tyypin selvittämisen.

Tietokannassa on mahdollista siis etsiä sali, jossa on vähintään haluttu määrä paikkoja ja joka on vapaa haluttuna aikana. Paikkamäärää voi vertailla luonnollisesti sitä vastaavan attribuutin avulla, ja vapautta varauksen ajankohta attribuutin avulla. Ajankohtaan tallentuu päivämäärä, että kellonaika.

Tietokannasta voi hakea myös tiedon siitä, mihin tarkoitukseen määrätty sali on varattu haluttuna aikana tai kuka varauksen on tehnyt. Varaukseen linkittyy aina varaaja (luokasta Työntekijät) ja käyttötarkoitus (Luennot, harjoituryhmäkerrat, tentit).

Varaukset

Tietokantaan tallennetaan tiedot kaikista saliin kohdistuvista varauksista. Varauksia voi tehdä luentoja, harjoitusryhmäkertoja tai tenttejä varten. Samassa salissa voi samaan aikaan olla usean kurssin tentti, mutta vain yhden kurssin luento tai harjoitusryhmä kerrallaan. Kaaviossa tämä näkyy jokaista luokkaa koskevissa merkinnöissä.

Varaus voi olla pidempi kuin se tilaisuus, jota varten sali on varattu (esimerkiksi tenteissä varataan ylimääräistä aikaa ennen tentin alkua ja sen loppumisen jälkeen). Tämän vuoksi varaukselle tallennetaan varauksen pituus, eikä niinkään tilaisuuden pituutta. Jokaisella salivarauksilla on yksikäsitteinen varaustunnus. Varauksista tallenetaan tiedot siitä, mitä tilaisuutta varten varaus on - tätä varten on linkit tenteistä, luennoista ja harjoitusryhmäkerroista. Varauksista tallennetaan myös tieto siitä koska varaus on tehty attribuuttiin varauksenAika. Varauksista tallennetaan tieto siitä, kuka työntekijä on tehnyt varauksen. Tästä syystä luokka Työntekijät liittyy Varaus luokkaan.

Varauksista on mahdollista selvittää myös historia tiedot. On mahdollista selvittää esimerkiksi minkä kurssin millekkä harjoitusryhmälle sali oli varattu viime vuonna määrättyyn aikaan tutkimalla varauksen ajankohtaa, ja vertaamalla sitä harjoitusryhmäkertojen ajankohtaa, ja katsomalla tuloksista mihin kurssikertaan ja kurssiin tämä liittyy.

Selostus Ratkaisusta

Työntekijät

Tietokannassa on yliopiston työntekijöistä yksikäsitteinen tunnus avain attribuuttina, tehtävänimike, nimi, osoite, puhelinnumero sekä työsuhteen alkamis- ja mahdollinen loppumispäivämäärä. Yhdestä työntekijästä on tallennettu vain yhden työsuhteen tiedot ja tietokannassa ei oteta huomioon sitä mahdollisuutta, että samalla työntekijällä olisi useita määräaikaisia työsuhteita. Työntekijöiden ainoa merkitys tietokannassa on mahdollisuus tehdä varauksia saleihin eri tilaisuuksia varten. Yhden työntekijän tekemien varausten määrää ei ole rajoitettu.

Funktionaaliset Riippuvuudet

Funktionaaliset riippuvuudet, Anomaliot sekä Boyce–Codd

Tietokannassamme ei ole redundansseja tai anomalioita. Esimerkiksi tenteillä, teintti-ilmoittautumisilla, luennoilla ja harjoituskerroilla on omat päivämäärät ja ajat, mutta esimerkiksi varauksen ajankohta on eri asia, joten tässä ei ole päällekkäisyyksiä. Lisäksi on olemassa joitakin tilanteita, joissa kahdella yhteydessä olevalla luokalla on molemmilla avainattribuutit esimerkiksi luennolla on “nimi” ja rakennuksella on “nimi”, mutta nämä avainattribuutit aina yksilöllisiä niiden luokkien sisällä, joissa ne sijaitsevat, joten tässäkään ei ole toistoa.

Voisimme myös olettaa, että opiskelijan puhelinnumero on yksilöllinen. Kuitenkin tämä ei ole funktionaalinen riippuvuus, sillä puhelinnumero voi olla NULL-arvoinen. Lisäksi sitä ei tulisi käyttää avaimena, koska se on ulkoisesti annettu tieto, emmekä voi olettaa sen olevan todella yksilöllinen.

Tietokantamme noudattaa Boyce-Coddin normalimuotoa, mikä tarkoittaa, että ainoastaan relaatioiden pääavaimet aiheuttavat funktionaalisia riippuvuuksia. Tämä takaa sen, ettei tietokannassa ole tarpeetonta tai toistuvaa tietoa. Toisin sanoen, ensisijaiset avaimet määrittelevät yksiselitteisesti kaikki muut attribuutit kussakin relaatiossa, mikä tekee tietokannasta tehokkaan ja helpommin ylläpidettävän.

Ensimmäisen osan palautuksen jälkeen tehdyt muutokset

Tarkastajan kommenttien perusteella tehdyt muutokset

Olemme tehneet ensimmäisen osan palautukseen saamiemme kommenttien, sekä luodessamme SQL-tietokantaa SQLiteStudio-ympäristössä huomattujen virheiden perusteella pieniä muutoksia ja tarkennuksia luokkiin ja attribuutteihin:

- Ensimmäisessä osassa tietyn varusteen määrä oli merkitty Varuste-luokan attribuutiksi.Tällöin määrä kuvasi kaikkien eri saleissa olevien samantyyppisten varusteiden määrää. Tämä attribuutti osoittautui turhaksi, sillä esimerkiksi videotykkien määrän voi saada tietokannasta laskemalla halutun varustetyyppi rivien lukumäärän COUNT() funktiolla. Tämä attribuutti on poistettu Varuste-luokasta.

- Tentti-luokalla voi olla useita varauksia, joka mahdollistaa useiden salien varaamisen isoille tenteille. Tämän takia tentti-varaus assosiaatio on muutettu monesta moneen.

- Harjoitusryhmä-luokkaan ei kuulu aikoja ilmaisevia attribuutteja, vaan ne kuuluvat harjoitusryhmäkerta-luokkaan. Tämän takia harjoitusryhmä-luokasta on poistettu aika attribuutti.

- Assosiaation harjoitusryhmän-kerta multiplisiteetti on ollut vihreellisesti harjoitusryhmän päässä tähti merkiten mielivaltaista määrää harjoitusryhmiä yhtä harjoitusryhmäkertaa kohtaan. Tämä on muutettu monesta yhteen.

- Luokasta Kurssikerta puutui avainmerkintä, jonka takia kurssikerta-luokkaan on lisätty aivaimiksi molemmat lukukausi sekä vuosi, jotka muodostavat yhdessä kurssin kurssikoodin kanssa avaimen.

- Assosiaation kuuluu-ryhmään multiplisiteetti luokan Harjoitusyhmä päässä on ollut virheellisesti yksi. Opiskelija voi ilmoittautua eri kurssien harjoitusryhmiin, joten multiplisiteetti on muutettu yhdestä moneen.

Ensimmäisen osan palautuksen jälkeen tehdyt muutokset

Muut muutokset / parannukset

Tarkastajan kommenttien lisäksi olemme huomanneet muutamia epäsäännölisyyksiä ja parannuksia, jotka tehostavat tietokantamme toimintaa:

- Opiskelijat-luokan assosiaatioiden nimet tentti_suoritukset ja tentti_ilmoittautumiset luokkien kanssa on muutettu UML-kaaviossa vastaamaan relaatioihin kirjoitettuihin assosiaatioiden nimiä. Samoin on tehty salit ja varusteet luokkien väliselle assosiaatiolle.

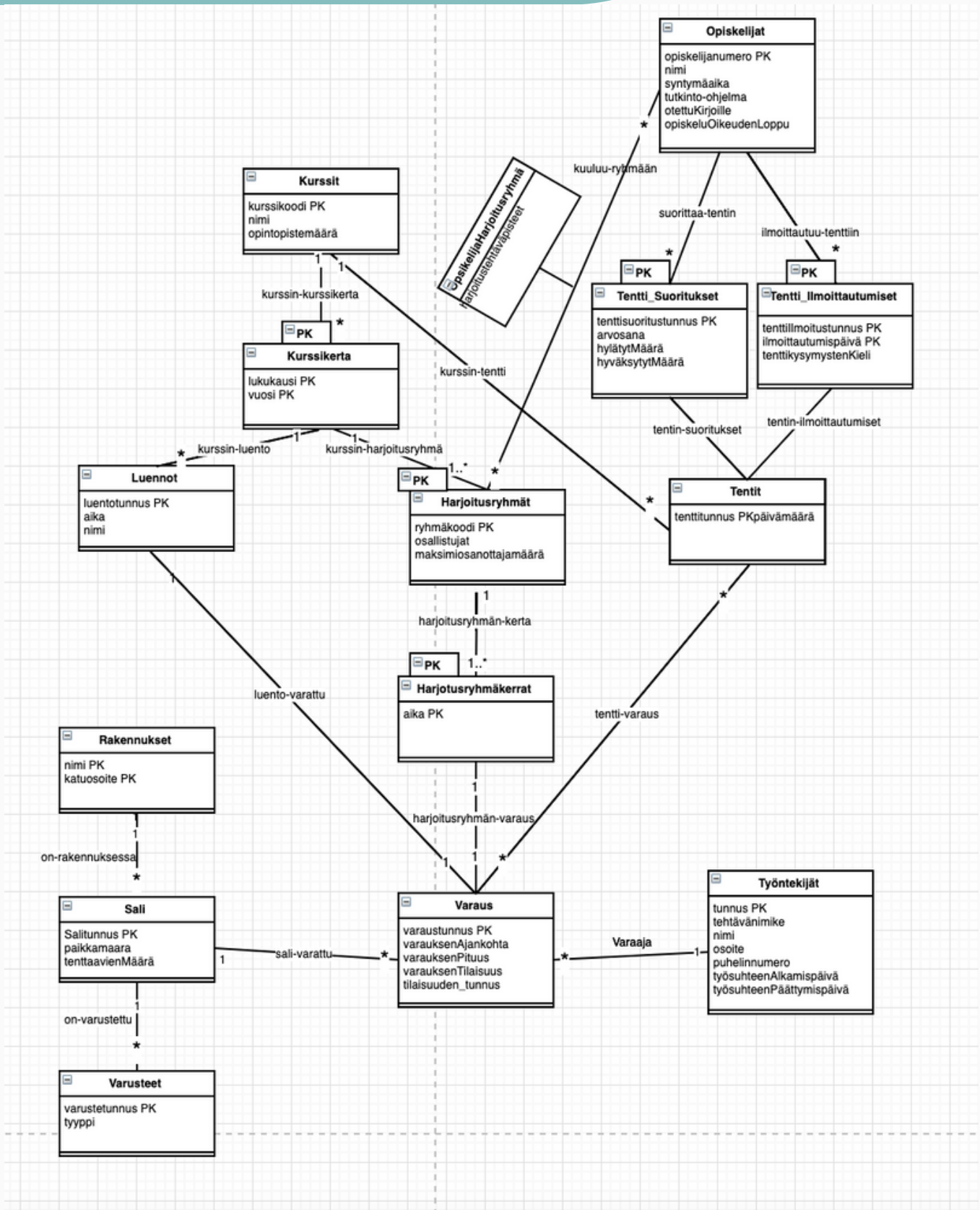
- Tentti_suoritukset ja Opiskelijat luokkien välisen assosiaation nimi on muutettu olemaan eri kuin tentti_suoritukset luokan nimi.

- Varaus-luokassa ajankohta attribuutti on jaettu kahteen attribuuttiin: päivämäärä ja kellonaika. Varaus-luokkaan on lisäksi lisätty tilaus_tunnus attribuutti, johon tallennetaan varauksen tilaisuuden tunnus.

- Luento-luokkaan ja Tentit-luokkaan on luotu uniikit tunnukset avaimiksi: luentotunnus ja tenttitunnus.

- Opiskelijat-luokan ja Harjoitusryhmä-luokan väliin luotu assosiaatioluokka OpiskelijaHarjoitusryhmä, johon siirretty harjoitustehtäväpisteet.

Päivitetty UML-kaavio



Tietokannan muunto SQL-muotoon

CREATE TABLE -käskyt

Tietokanta on muutettu päivitetyn UML-kaavion perusteella SQL-muotoon, luomalla CREATE TABLE -käskyt itse. CREATE TABLE -käskyt voidaan syöttää suoraan SQLLiteStudio -ympäristöön. Attribuuteille on annettu järkevät tietotyypit, joiden valinta on perusteltu käskyjen alapuolella.

```
CREATE TABLE Kurssit (
    kurssikoodi CHAR(10) PRIMARY KEY,
    nimi VARCHAR(255) NOT NULL,
    opintopistemäärä INT CHECK (opintopistemäärä >= 0)
);

CREATE TABLE Kurssikerrat (
    kurssikoodi CHAR(10),
    lukukausi VARCHAR(255) NOT NULL,
    vuosi INT CHECK (vuosi > 1900),
    PRIMARY KEY (kurssikoodi, lukukausi, vuosi),
    FOREIGN KEY (kurssikoodi) REFERENCES Kurssit(kurssikoodi)
);

CREATE TABLE Harjoitusryhmät (
    kurssikoodi CHAR(10),
    ryhmäkoodi CHAR(10),
    osallistujat INT CHECK (osallistujat >= 0),
    maksimiosanottajamäärä INT CHECK (osallistujat < maksimiosanottajamäärä),
    PRIMARY KEY (kurssikoodi, ryhmäkoodi),
    FOREIGN KEY (kurssikoodi) REFERENCES Kurssit(kurssikoodi)
);

CREATE TABLE Harjoitusryhmäkerrat (
    kurssikoodi CHAR(10),
    ryhmäkoodi CHAR(10),
    aika DATETIME,
    PRIMARY KEY (ryhmäkoodi, aika),
    FOREIGN KEY (kurssikoodi, ryhmäkoodi) REFERENCES
Harjoitusryhmät(kurssikoodi, ryhmäkoodi)
);
```

Tietokannan muunto SQL-muotoon


```
CREATE TABLE Luennot (
    luentotunnus CHAR(10),
    aika DATETIME,
    nimi VARCHAR(255) NOT NULL,
    kurssikoodi CHAR(10),
    lukukausi VARCHAR(255) NOT NULL,
    vuosi INT CHECK (vuosi > 1900),
    PRIMARY KEY (luentotunnus)
    FOREIGN KEY (kurssikoodi, lukukausi, vuosi) REFERENCES Kurssikerrat(kurssikoodi,
lukukausi, vuosi)
);

CREATE TABLE Tentit (
    tenttitunnus CHAR(10),
    kurssikoodi CHAR(10),
    päivämäärä DATE,
    PRIMARY KEY (kurssikoodi, päivämäärä),
    FOREIGN KEY (kurssikoodi) REFERENCES Kurssit(kurssikoodi)
);

CREATE TABLE Opiskelijat (
    opiskelijanumero INT,
    nimi VARCHAR(255) NOT NULL,
    syntymäaika DATE,
    tutkinto_ohjelma VARCHAR(255),
    otettuKirjoille DATE,
    opiskeluOikeudenLoppu DATE,
    PRIMARY KEY (opiskelijanumero)
);

CREATE TABLE OpiskelijaHarjoitusryhmät (
    opiskelijanumero INT,
    kurssikoodi CHAR(10),
    ryhmäkoodi CHAR(10),
    harjoitustehtäväpisteet INT DEFAULT 0 CHECK (harjoitustehtäväpisteet >= 0),
    PRIMARY KEY (opiskelijanumero, kurssikoodi, ryhmäkoodi),
    FOREIGN KEY (opiskelijanumero) REFERENCES Opiskelijat(opiskelijanumero),
    FOREIGN KEY (kurssikoodi, ryhmäkoodi) REFERENCES Harjoitusryhmät(kurssikoodi,
ryhmäkoodi)
);
```

Tietokannan muunto SQL-muotoon


```
CREATE TABLE Tentti_ilmoittautumiset (
    opiskelijanumero INT,
    tenttiilmoitustunnus CHAR(10),
    ilmoittautumispäivä DATE,
    tenttikysymystenKieli VARCHAR(255) NOT NULL,
    kurssikoodi CHAR(10),
    päivämäärä DATE,
    PRIMARY KEY (opiskelijanumero, tenttiilmoitustunnus, ilmoittautumispäivä),
    FOREIGN KEY (opiskelijanumero) REFERENCES Opiskelijat(opiskelijanumero)
    FOREIGN KEY (kurssikoodi, päivämäärä) REFERENCES Tentit(kurssikoodi, päivämäärä)
);

CREATE TABLE Tentti_suoritukset (
    opiskelijanumero INT,
    tenttisuoritustunnus CHAR(10),
    arvosana INT CHECK (arvosana BETWEEN 0 AND 5),
    kurssikoodi CHAR(10),
    päivämäärä DATE,
    PRIMARY KEY (opiskelijanumero, tenttisuoritustunnus),
    FOREIGN KEY (opiskelijanumero) REFERENCES Opiskelijat(opiskelijanumero)
    FOREIGN KEY (kurssikoodi, päivämäärä) REFERENCES Tentit(kurssikoodi, päivämäärä)
);

CREATE TABLE Työntekijät (
    tunnus INT,
    tehtävänimike VARCHAR(255) NOT NULL,
    nimi VARCHAR(255) NOT NULL,
    osoite VARCHAR(255) NOT NULL,
    puhelinnumero VARCHAR(255) NOT NULL,
    työsuhteenAlkamispäivä DATE,
    työsuhteenPäättymispäivä DATE CHECK (työsuhteenAlkamispäivä <
työsuhteenPäättymispäivä),
    PRIMARY KEY (tunnus)
);
```

Tietokannan muunto SQL-muotoon


```
CREATE TABLE Rakennukset (
    nimi VARCHAR(255) NOT NULL,
    katuosoite VARCHAR(255) NOT NULL,
    PRIMARY KEY (nimi, katuosoite)
);

CREATE TABLE Salit (
    salitunnus CHAR(10),
    paikkamäärä INT CHECK (paikkamäärä > 0),
    tenttaavienMäärä INT CHECK (tenttaavienMäärä >= 0),
    rakennus_nimi VARCHAR(255) NOT NULL,
    rakennus_katuosoite VARCHAR(255) NOT NULL,
    PRIMARY KEY (salitunnus),
    FOREIGN KEY (rakennus_nimi, rakennus_katuosoite) REFERENCES Rakennukset(nimi,
katuosoite)
);

CREATE TABLE Varusteet (
    varustetunnus CHAR(10),
    tyyppi VARCHAR(255) NOT NULL,
    PRIMARY KEY (varustetunnus)
);

CREATE TABLE Varaus (
    varaustunnus CHAR(10),
    päivämäärä DATE,
    kellonaika TIME,
    varauksenPituus INT CHECK (varauksenPituus > 0),
    varauksenTilaisuus VARCHAR(255) CHECK (varauksenTilaisuus IN ('Tentti', 'Luento',
'Harjoitusryhmäkerta')) NOT NULL,
    tilaisuuden_tunnus CHAR(10),
    varauksen_tekijän_tunnus INT,
    salitunnus CHAR(10),
    PRIMARY KEY (varaustunnus),
    FOREIGN KEY (varauksen_tekijän_tunnus) REFERENCES Työntekijät(tunnus),
    FOREIGN KEY (salitunnus) REFERENCES Salit(salitunnus)
);
```

Tietotyypit

Tietotyyppien valinnan perustelu

Tietokanta, johon tallennetaan tietoja yliopiston kursseista, tiloista, tilavarauksista, työntekijöistä, opiskelijoista, ilmoittautumisista ja kurssien suorituksista vaatii oikeanlaiset tietotyypit toimiakseen halutulla tavalla. Olemme valinneet käytetyt tietotyypit seuraavin perustein:

CHAR(10) - Tietotyyppiä käytetään luokissa, joilla on uniikki identifioiva tunnus tai koodi, poislukien ihmisistä tietoa tallentavat luokat (Opiskelijat, Työntekijät). Tämä tietotyyppi mahdollistaa sen, että koodit voivat sisältää kirjaimia, erikoismerkkejä ja numeroita. Esimerkiksi kurssikoodi voi olla tietotyyppi valinnan ansiosta esimerkiksi “ELEC-A0110”. 10 merkkiä tarjoaa tarpeeksi paljon mahdollisuuksia luoda uniikkeja merkkiyhdistelmiä, mutta koodin tai tunnuksen ei tarvitse olla kuitenkaan välttämättä juuri 10 merkkiä pitkä. Lyhyemmätkin tunnukset ovat mahdollisia, kuten “CS-A1150”. Kirjaimien, erikoismerkkien ja numeroiden yhdistämisen mahdollisuus tarkoittaa, että koodeista voidaan tehdä johdonmukaisempia, ja helpompia ymmärtää näkemällä. Kurssikoodista esimerkiksi voi huomata kuinka etuosa kuvaa korkeakoulua, jälkimmäinen osa itse kurssia ja erikoismerkkejä voidaan käyttää jäsentelyyn.

INT - Tietotyyppiä käytetään yleisesti paljon määrää kuvaavissa attribuuteissa, joissa esiintyy vain kokonaisia numeroita, kuten esimerkiksi opintopistemäärä, vuosi tai maksimiosanottajamäärä. Varauksen pituus ilmaistaan myös käyttäen INT tietotyyppiä, sillä voimme olettaa ja tehdä linjauksen, että tietokantaan tallennetaan pituus minuuteissa. Tätä tarkempaa ei ole tarvetta tallentaa tietoa pituudesta, joten desimaaleille ei ole tarvetta. Varauksien pituudet eivät myöskään ole järjettömän pitkiä, joten minuutit riittävät oikein hyvin. INT-tietotyyppiä käytetään myös luokkien Opiskelijat ja Työntekijät uniikkien tunnusten luomisessa. Opiskelijanumerot yleensä ovatkin eri pituisia numerosarjoja, kuten 100810721, ja kirjaimille ei ole erityisempää tarvetta. Tästä johtuen olemme tehneet tietokannassamme linjauksen, että henkilöitä käsittelevät luokat saavat numerosarjan tunnuksekseen.

Tietotyypit

DATETIME, DATE, TIME - Tietotyypit mahdollistavat aikaan ja päivämäärään liittyvän tiedon tallentamisen. DATETIME mahdollistaa tiedon tallentamisen sekä päivämäärästä, että ajasta, kuten Harjoitusryhmäkerran tai Luennon aika attribuutissa on suotavaa. Näissä kellonajan tallentaminen on aivan yhtä tärkeää. DATE-tietotyyppiä voidaan käyttää, kun pelkkä päivämäärän tallentaminen on riittävää, kuten esimerkiksi syntymäaikaa, ilmoittautumispäivää tai opiskelu oikeuden loppua tallennettaessa. Vastaavilla attribuuteilla kellonajalla ei ole juurikaan merkitystä. TIME-tietotyyppiä voidaan käyttää, kun pelkkä kellonajan tallentaminen on riittävää, kuten varauksen alkamisen ajankohta.

VARCHAR(255) - Tämä tietotyyppi esiintyy lähes kaikissa muissa yhteyksissä, kuin edellä mainituissa. Tämä tietotyyppi mahdollistaa monenlaisten merkkiyhdistelmien tiedon tallentamisen, joilla ei ole tarkempia vaatimuksia tai säännöllisyyttä. Tätä tietotyyppiä käytetään muun muassa osoitetietojen, nimien tai lukukauden tallentamiseen. Myös puhelinnumero tallennetaan tällä tietotyypillä, sillä tämä mahdollistaa eripituisten ja erilaisia merkkejä käyttävien kansainvälisten numeroiden tallentamisen. Pituudeksi on asetettu 255, sillä näin monta merkkiä voidaan tallentaa tietokantaan käyttäen vain yhtä tavua. Tätä pidemmät merkki sarjat vaativat vähintäänkin kaksi tavua tiedon tallentamiseen, ja näin pitkille merkkijonoille ei ole tietokannassamme tarvetta.

Eheysrajoitteet ja viite-eheys

Eheyrajoitteet ja viite-eheys

Olemme määritelleet tarkkaan eheysrajoitteita varmistaaksemme sen, että tietokantaan syötettävä data on halutun mukaista. Emme ole kuitenkaan lisänneet rajoitteita liikaa, jotta tietokannasta ei tule liian rajoitettu.

NOT NULL eheysrajoitetta käytetään usein VARCHAR(255) tietotyypin yhteydessä, jolla varmistetaan, että attribuuttiin ei syötetä tyhjää arvoa.

CHECK eheysrajoitteella tarkistetaan usein INT tietotyyppien yhteydessä (jos kyse ei ole tunnuksesta), että taulukkoon syötetään positiivinen luku jos on kyse määrästä. Jos on kyse vuodesta, tarkistetaan, että vuosi ei ole mielivaltaisen kaukana menneisyydessä. CHECK eheysrajoitetta käytetään myös tarkistamaan, että syötetty maksimiosanottajamäärä on esimerkiksi suurempi kuin osanottajamäärä tai että DATE tietotyypin yhteydessä työsuhteen lopetus on kauempana tulevaisuudessa kuin aloitus.

DEFAULT käskyä käytetään tietokannassa asettamaan Opiskelijan rekisteröityessä kurssille harjoitusryhmän kautta hänen harjoitustehtäväpisteensä nollaan.

Hakemistot

Tietokantaan luodut hakemistot

Tässä tietokannassa, johon tallennetaan tietoja yliopiston kursseista, tiloista, tilavarauksista, työntekijöistä, opiskelijoista, ilmoittautumisista ja kurssien suorituksista tavanomaisessa käytössä yleisimmät haut saattavat perustua kurssien, työntekijä- tai opiskelijatietojen hakemiseen kuin myös tenttivarausten ja tilavarausten hallitsemiseen. Pohtien tavanomaista käyttöä olemme luoneet seuraavat hakemistot, jotka tukevat tietokannan tarpeenmukaista käyttöä. Oheen on liitetty esimerkki käyttötilanteesta.

CREATE INDEX index_tentit_päivämäärä ON Tentit (päivämäärä);

Tenteistä tehty hakemisto päivämäärällä on tärkeää erityisesti silloin, kun halutaan tietää tiettynä päivänä olevat tentit. Tätä hakemistoa voidaan käyttää, kun halutaan näyttää opiskelijalle kalenteriin kyseisen päivän tentit, tai tulostaa tieto rakennuksessa olevista tenteistä kyseisenä päivänä.

CREATE INDEX index_tentti_ilmoittautumiset_opiskelijanumero ON Tentti_ilmoittautumiset(opiskelijanumero);

CREATE INDEX index_tentti_suoritukset_opiskelijanumero ON Tenttisuoritukset(opiskelijanumero);

Tentti-ilmoittautumisista ja tenttisuorituksista tehty hakemisto opiskelijanumerolla nopeuttaa yhden halutun opiskelijan kaikkien tenttitietojen hakemista. Tätä tietoa voidaan tarvita esimerkiksi kun halutaan seurata yhden kyseisen opiskelijan akateemista edistystä tai yliopiston omilla opiskelijan sivuilla halutaan näyttää opiskelijalle itselleen hänen tentti historiansa ja suorituksensa, ja tulevat ilmoittautumiset.

CREATE INDEX index_työntekijät_tehtävänimike ON Työntekijät(tehtävänimike);

Työntekijöistä tehty hakemisto tehtävänimikkeen perusteella voi olla erittäin hyödyllinen esimerkiksi silloin, kun halutaan saada tietoa työntekijöistä tietyllä tehtävänimikkeellä.

Hakemistot

CREATE INDEX index_varaus_päivämäärä_salitunnus ON Varaus(päivämäärä, salitunnus);

CREATE INDEX index_varaus_päivämäärä ON Varaus(päivämäärä);

CREATE INDEX index_varaus_salitunnus ON Varaus(salitunnus);

Varauksista tehty hakemisto päivämäärän ja salin perusteella voi olla äärimmäisen hyödyllinen, kun halutaan tieto yhden tietyn päivän aikana olevista varauksista. Päivämäärän ja salin yhdistelmän hakemiston tietoa voidaan tarvita esimerkiksi kun halutaan hakea kyseisen salin ja haluttujen päivien tiedot, että voidaan tulostaa aikataulut paperille ja laittaa paperi luentosalin oveen. Pelkällä sali attribuutilla tehtyä hakemistoa voidaan tarvita, kun halutaan tutkia halutun salin tulevia aikatauluja mahdollisia luentoja tai tenttejä suunnitellessa. Pelkällä päivämäärä attribuutilla tehty hakemistoa voidaan tarvita, kun halutaan tutkia tietyn päivän aikana olevia vapaita tiloja tenttiä, luentoja tai muita tilaisuuksia varten.

CREATE INDEX index_harjoitusryhmät_kurssikoodi ON Harjoitusryhmät(kurssikoodi);

Harjoitusryhmien taulusta voi tulla yliopistossa hyvin pitkä, ja hakemiston avulla koko taulua ei tarvitse käydä läpi. Hakemiston avulla halutun kurssin harjoitusryhmät löytää nopeasti ja tämä on erityisen hyödyllistä, kun halutaan näyttää opiskelijalle mihin harjoitusryhmiin hän voi ilmoittautua.

CREATE INDEX index_opiskelijat_tutkinto_ohjelma ON Opiskelijat(tutkinto_ohjelma);

Opiskelijoita voi yliopistossa olla kymmeniä tuhansia. Hakemisto nopeuttaa tiedon hakua kun tarvitsee tietää kuka opiskelee mitäkin tutkinto-ohjelmaa. Jos halutaan selvittää esimerkiksi mitkä informaatioteknologia opiskelijat on otettu yliopistoon kirjoille 2022 jälkeen, voidaan tieto etsiä hakemistosta menemättä kymmenien tuhansien opiskelijoiden läpi.

Näkymät

Tulevat tentit

Yksi hyödyllinen näkymä on esimerkiksi näkymä, joka tarjoaa yhteenvedon tulevista tenteistä näyttäen niistä kurssin nimen, tentin päivämäärän, salitunnuksen, rakennuksen nimen ja katuosoitteen. Näkymällä voi olla useita eri käyttötarkoituksia niin isossa kaavassa kuin pienemmässäkin. Opiskelijalle voidaan näyttää koulun järjestelmissä hänen tulevat tentit, koska tiedetään mille kursseille opiskelija on ilmoittautunut, ja näkymästä voidaan hakea haluttujen kurssien avulla halutut tentit. Myös yliopiston työntekijät voivat käyttää näkymää tutkiakseen tulevia aikatauluja ja ehkäistä esimerkiksi päällekkäisyyksiä. Näkymä yhdistelee tarpeelliset tiedot Tentit, Kurssit, Varaus, Salit ja Rakennukset luokista näiden yhteisten attribuuttien avulla, ja näyttää kaiken tulevista tenteistä tarvittavan tiedon selkeästi yhdessä näkymässä.

```
CREATE VIEW Tulevat_Tentit AS
SELECT
    K.nimi AS Kurssin_Nimi,
    T.päivämäärä AS Tentin_Päivämäärä,
    S.salitunnus AS Sali,
    R.nimi AS Rakennuksen_Nimi,
    R.katuosoite AS Rakennuksen_Osoite
FROM
    Tentit T
    JOIN Kurssit K ON T.kurssikoodi = K.kurssikoodi
    JOIN Varaus V ON T.tenttitunnus = V.tilaisuuden_tunnus AND T.päivämäärä =
V.päivämäärä AND V.varauksenTilaisuus = 'Tentti'
    JOIN Salit S ON V.salitunnus = S.salitunnus
    JOIN Rakennukset R ON S.rakennus_nimi = R.nimi AND S.rakennus_katuosoite
= R.katuosoite
WHERE
    T.päivämäärä >= CURRENT_DATE
ORDER BY
    T.päivämäärä;
```

Näkymät

Tulevat luennot

Opiskelijat saattavat useinkin tarvita tietoa tulevista luennoista, jotta he osaavat tulla oikeaan aikaan oikeeseen paikkaan. Tietokantaan voidaan luoda näkymä, joka tarjoaa yhteenvedon tulevista luennoista näyttäen niistä kurssin nimen, luennon ajankohdan, luennon nimen, salitunnuksen, rakennuksen nimen ja rakennuksen katuosoitteen. Näkymää voidaan hyödyntää esimerkiksi, kun halutaan siirtää opiskelijan kalenteriin tieto hänen tulevistaan luennoistaan.

```
CREATE VIEW Tulevat_Luennot AS
SELECT
    K.nimi AS Kurssin_Nimi,
    L.aika AS Luennon_Aika,
    L.nimi AS Luennon_Nimi,
    S.salitunnus AS Sali,
    R.nimi AS Rakennuksen_Nimi,
    R.katuosoite AS Rakennuksen_Osoite
FROM
    Luennot L
     JOIN Kurssikerrat KK ON L.kurssikoodi = KK.kurssikoodi AND L.lukukausi =
KK.lukukausi AND L.vuosi = KK.vuosi
    JOIN Kurssit K ON KK.kurssikoodi = K.kurssikoodi
    JOIN Varaus V ON L.luentotunnus = V.tilaisuuden_tunnus AND DATE(L.aika) =
V.päivämäärä AND V.varauksenTilaisuus = 'Luento'
    JOIN Salit S ON V.salitunnus = S.salitunnus
    JOIN Rakennukset R ON S.rakennus_nimi = R.nimi AND S.rakennus_katuosoite
= R.katuosoite
WHERE
    L.aika >= CURRENT_TIMESTAMP
ORDER BY
    L.aika;
```

Esimerkkitietueet

INSERT –käskyt

Näiden INSERT -käskyjen syöttämisen jälkeen, voi tulevia tyypillisiä käyttötapauksia koeistaa siten, että SQLiteStudio palauttaa myös rivejä.

```
-- Kurssit
INSERT INTO Kurssit(kurssikoodi, nimi, opintopistemäärä) VALUES
('CS-A1150', 'Tietokannat', 5),
('MS-A0003', 'Matriisilaskenta', 5),
('ELEC-A0110', 'Johdatus opiskeluun Sähkötekniikan kandidaattiohjelmassa', 2);

-- Kurssikerrat
INSERT INTO Kurssikerrat(kurssikoodi, lukukausi, vuosi) VALUES
('CS-A1150', 'Kevät', 2023),
('MS-A0003', 'Kevät', 2023),
('ELEC-A0110', 'Syksy', 2022);

-- Harjoitusryhmät
INSERT INTO Harjoitusryhmät(kurssikoodi, ryhmäkoodi, osallistujat, maksimiosanottajamäärä) VALUES
('CS-A1150', 'HR01', 10, 20),
('MS-A0003', 'HR02', 15, 25),
('ELEC-A0110', 'HR03', 20, 30);

-- Harjoitusryhmäkerrat
INSERT INTO Harjoitusryhmäkerrat(kurssikoodi, ryhmäkoodi, aika) VALUES
('CS-A1150', 'HR01', '2023-05-10 10:00:00'),
('MS-A0003', 'HR02', '2023-05-11 10:00:00'),
('ELEC-A0110', 'HR03', '2023-05-12 10:00:00');

-- Luennot
INSERT INTO Luennot(luentotunnus, aika, nimi, kurssikoodi, lukukausi, vuosi) VALUES
('LT01', '2023-05-13 13:00:00', 'Johdatus kurssille', 'CS-A1150', 'Kevät', 2023),
('LT02', '2023-05-14 14:00:00', 'Gaussin eliminaatio', 'MS-A0003', 'Kevät', 2023),
('LT03', '2023-05-15 15:00:00', 'Vaihto-opinnot', 'ELEC-A0110', 'Syksy', 2022);
```

Esimerkkitietueet

INSERT –käskyt

```
-- Tentit
INSERT INTO Tentit(tenttitunnus, kurssikoodi, päivämäärä) VALUES
('T01', 'CS-A1150', '2023-06-01'),
('T02', 'MS-A0003', '2023-06-02'),
('T03', 'ELEC-A0110', '2023-06-03');

-- Opiskelijat
INSERT INTO Opiskelijat(opiskelijanumero, nimi, syntymäaika, tutkinto_ohjelma, otettuKirjoille, opiskeluOikeudenLoppu) VALUES
(10002001, 'Aamos Rex', '2000-01-06', 'Informaatioverkostot', '2020-09-01', '2024-08-31'),
(10002002, 'Gabriella Tolvanen', '2001-01-08', 'Bioinformaatioteknologia', '2021-09-01', '2025-08-31'),
(10002003, 'Kalle Berg', '2002-01-10', 'Tietotekniikka', '2022-09-01', '2026-08-31');

-- OpiskelijaHarjoitusRyhmät
INSERT INTO OpiskelijaHarjoitusryhmät(opiskelijanumero, kurssikoodi, ryhmäkoodi)
SELECT O.opiskelijanumero, H.kurssikoodi, H.ryhmäkoodi
FROM Opiskelijat O
CROSS JOIN Harjoitusryhmät H;

-- Tentti_ilmoittautumiset
INSERT INTO Tentti_ilmoittautumiset(opiskelijanumero, tenttiilmoitustunnus, ilmoittautumispäivä, tenttikysymystenKieli, kurssikoodi, päivämäärä) VALUES
(10002001, 1, '2023-05-20', 'Englanti', 'CS-A1150', '2023-06-01'),
(10002001, 2, '2023-05-20', 'Englanti', 'MS-A0003', '2023-06-02'),
(10002001, 3, '2023-05-20', 'Englanti', 'ELEC-A0110', '2023-06-03'),
(10002002, 4, '2023-05-21', 'Suomi', 'CS-A1150', '2023-06-01'),
(10002002, 5, '2023-05-21', 'Suomi', 'MS-A0003', '2023-06-02'),
(10002002, 6, '2023-05-21', 'Suomi', 'ELEC-A0110', '2023-06-03'),
(10002003, 7, '2023-05-22', 'Ruotsi', 'CS-A1150', '2023-06-01'),
(10002003, 8, '2023-05-22', 'Ruotsi', 'MS-A0003', '2023-06-02'),
(10002003, 9, '2023-05-22', 'Ruotsi', 'ELEC-A0110', '2023-06-03');
```

Esimerkkitietueet

INSERT –käskyt

-- Tentti_suoritukset
INSERT INTO Tentti_suoritukset(opiskelijanumero, tenttisuoritustunnus, arvosana, kurssikoodi, päivämäärä) VALUES
(10002001, 1, 3, 'CS-A1150', '2023-06-01'),
(10002001, 2, 5, 'MS-A0003', '2023-06-02'),
(10002001, 3, 5, 'ELEC-A0110', '2023-06-03'),
(10002002, 4, 3, 'CS-A1150', '2023-06-01'),
(10002002, 5, 2, 'MS-A0003', '2023-06-02'),
(10002002, 6, 1, 'ELEC-A0110', '2023-06-03'),
(10002003, 7, 5, 'CS-A1150', '2023-06-01'),
(10002003, 8, 4, 'MS-A0003', '2023-06-02'),
(10002003, 9, 4, 'ELEC-A0110', '2023-06-03');

-- Työntekijät
INSERT INTO Työntekijät(tunnus, tehtävänimike, nimi, osoite, puhelinnumero, työsuhteenAlkamispäivä, työsuhteenPäättymispäivä) VALUES
(1001, 'Professori', 'Prof. A', 'Yliopistonkatu 1', '0123456789', 2020-01-01, 2025-01-01),
(1002, 'Assistantti', 'Tauno Palo', 'Töölönkatu 2', '9876543210', 2021-01-01, 2026-01-01),
(1003, 'Siivooja', 'Silvia Laine', 'Puistokatu 3', '1234567890', 2022-01-01, 2023-01-01);

-- Rakennukset
INSERT INTO Rakennukset(nimi, katuosoite) VALUES
('TUAS-talo', 'Yliopistonkatu 1'),
('Väre', 'Tiedontie 2'),
('Kandikeskus', 'Matematiikankatu 3');

-- Salit
INSERT INTO Salit(salitunnus, paikkamäärä, tenttaavienMäärä, rakennus_nimi, rakennus_katuosoite) VALUES
('S01', 100, 50, 'TUAS-talo', 'Yliopistonkatu 1'),
('S02', 200, 100, 'Väre', 'Tiedontie 2'),
('S03', 300, 150, 'Kandikeskus', 'Matematiikankatu 3');

Esimerkkitietueet

INSERT –käskyt

```
-- Varusteet
INSERT INTO Varusteet(varustetunnus, tyyppi) VALUES
('V01', 'Projektori'),
('V02', 'Liitutaulu'),
('V03', 'Tietokone');

-- Varaus
INSERT INTO Varaus(varaustunnus, päivämäärä, kellonaika, varauksenPituus, varauksenTilaisuus, tilaisuuden_tunnus, varauksen_tekijän_tunnus, salitunnus) VALUES
('VR01', '2023-05-30', '10:00:00', 180, 'Tentti', 'T01', 1002, 'S01'),
('VR02', '2023-05-30', '10:00:00', 180, 'Tentti', 'T01', 1002, 'S02'),
('VR03', '2023-05-30', '10:00:00', 180, 'Tentti', 'T01', 1002, 'S03'),
('VR04', '2023-05-28', '11:15:00', 105, 'Luento', 'LT01', 1001, 'S02'),
('VR05', '2023-05-31', '11:15:00', 105, 'Luento', 'LT02', 1001, 'S02'),
('VR06', '2023-06-01', '12:00:00', 90, 'Harjoitusryhmäkerta', 'HR01', 1003, 'S03');
```

Käyttötapaukset

Käyttötapaus 1: Uuden opiskelijan rekisteröinti

Rekisteröidään uusi opiskelija Maija Meikäläinen tietokantaan ja annetaan hänelle uusi opiskelijanumero. Haetaan ensiksi vapaana oleva opiskelijanumero:

SELECT MAX(opiskelijanumero) + 1 AS uusi_opiskelijanumero
FROM Opiskelijat;

	uusi_opiskelijanumero
1	10002004

Tarkistetaan, että opiskelijanumero on vapaana:

SELECT opiskelijanumero
FROM Opiskelijat
WHERE opiskelijanumero = 10002004;

	opiskelijanumero

Kysely ei palauta rivejä, joten opiskelijanumero on vapaana.

INSERT INTO Opiskelijat (opiskelijanumero, nimi, syntymäaika, tutkinto_ohjelma, otettuKirjoille, opiskeluOikeudenLoppu)
VALUES(10002004, 'Maija Meikäläinen', '1990-01-01', 'Tietotekniikka', CURRENT_DATE, '2030-06-01');

	opiskelijanu	nimi	syntymäaika	tutkinto_ohjelma	otettuKirjoille	opiskeluOikeudenLoppu
1	10002001	Aamos Rex	2000-01-06	Informaatioverkostot	2020-09-01	2024-08-31
2	10002002	Gabriella Tolvanen	2001-01-08	Bioinformaatioteknologia	2021-09-01	2025-08-31
3	10002003	Kalle Berg	2002-01-10	Tietotekniikka	2022-09-01	2026-08-31
4	10002004	Maija Meikäläinen	1990-01-01	Tietotekniikka	2023-05-10	2030-06-01

Käyttötapaukset

Käyttötapaus 2: Ilmoitetaan opiskelija tenttiin

Ilmoitetaan opiskelija Maija Meikäläinen kurssin Tietokannat tenttiin. Tarkistetaan onko opiskelijan Maija Meikäläinen opiskelijanumero voimassa, onko kurssin Tietokannat kurssikoodi on voimassa ja seuraava vapaan oleva tenttisuoritustunnus

SELECT opiskelijanumero
FROM Opiskelijat
WHERE opiskelijanumero = 10002004;

	opiskelijanumero
1	10002004

SELECT kurssikoodi
FROM Kurssit
WHERE kurssikoodi = 'CS-A1150';

	kurssikoodi
1	CS-A1150

SELECT MAX(tenttiilmoitustunnus) + 1 AS uusi_tenttiilmoitustunnus
FROM Tentti_ilmoittautumiset;

	uusi_tenttiilmoitustunnus
1	10

Opiskelijanumero, sekä kurssikoodi ovat voimassa, joten voimme lisätä tentti-ilmottautumisen kyseiselle kurssille. Kysely palauttaa vapaaksi tenttitunnukseksi tunnuksen 10. Lisätään tentti-ilmoittautuminen tietokantaan seuraavalla vapaana olevalla tentti-ilmoitustunnuksella.

INSERT INTO Tentti_ilmoittautumiset (opiskelijanumero, tenttiilmoitustunnus, ilmoittautumispäivä, tenttikysymystenKieli, kurssikoodi, päivämäärä)
VALUES (10002004, 10, '2023-05-23', 'Suomi', 'CS-A1150', '2023-06-01');

	opiskelijanu	tenttiilmo	ilmoittautumi	tenttikysy	kurssikoodi	päivämäärä
1	10002001	1	2023-05-20	Englanti	CS-A1150	2023-06-01
2	10002001	2	2023-05-20	Englanti	MS-A0003	2023-06-02
3	10002001	3	2023-05-20	Englanti	ELEC-A0110	2023-06-03
4	10002002	4	2023-05-21	Suomi	CS-A1150	2023-06-01
5	10002002	5	2023-05-21	Suomi	MS-A0003	2023-06-02
6	10002002	6	2023-05-21	Suomi	ELEC-A0110	2023-06-03
7	10002003	7	2023-05-22	Ruotsi	CS-A1150	2023-06-01
8	10002003	8	2023-05-22	Ruotsi	MS-A0003	2023-06-02
9	10002003	9	2023-05-22	Ruotsi	ELEC-A0110	2023-06-03
10	10002004	10	2023-05-23	Suomi	CS-A1150	2023-06-01

Käyttötapaukset

Käyttötapaus 3: Tenttitulosten lisääminen

Lisätään opiskelijan *Maija Meikäläinen* kurssin CS-A1150 tentin tulokset tietokantaan, jotta tenttitulos voidaan ilmoittaa opiskelijalle. Haetaan seuraava vapaana oleva tenttisuoritustunnus:

SELECT MAX(tenttisuoritustunnus) + 1 AS uusi_tenttisuoritustunnus
FROM Tentti_suoritukset;

	uusi_tenttisuoritustunnus
1	10

Kysely palauttaa tulokseksi 10, eli 10 on seuraava vapaana oleva tenttiilmoitustunnus. Lisätään tenttisuoritus tietokantaan:

INSERT INTO Tentti_suoritukset(opiskelijanumero, tenttisuoritustunnus, arvosana, kurssikoodi, päivämäärä)
VALUES(10002004, 10, 5, 'CS-A1150', '2023-06-01');

	opiskelijanumero	tenttisuoritustunnus	arvosana	kurssikoodi	päivämäärä
1	10002001	1	3	CS-A1150	2023-06-01
2	10002001	2	5	MS-A0003	2023-06-02
3	10002001	3	5	ELEC-A0110	2023-06-03
4	10002002	4	3	CS-A1150	2023-06-01
5	10002002	5	2	MS-A0003	2023-06-02
6	10002002	6	1	ELEC-A0110	2023-06-03
7	10002003	7	5	CS-A1150	2023-06-01
8	10002003	8	4	MS-A0003	2023-06-02
9	10002003	9	4	ELEC-A0110	2023-06-03
10	10002004	10	5	CS-A1150	2023-06-01

Käyttötapaukset

Käyttötapaus 4: Kurssin tenttitulosten keskiarvo

Tarkistetaan kurssin CS-A1150 tenttitulokset ja lasketaan niistä kurssin keskiarvo. Haetaan ensiksi kaikki tenttisuoritukset, jotka liittyvät haluttuun kurssiin

```
SELECT ts.opiskelijanumero, o.nimi AS opiskelijan_nimi, ts.arvosana
FROM Tentti_suoritukset AS ts
JOIN Tentti_ilmoittautumiset AS ti ON ts.opiskelijanumero = ti.opiskelijanumero
JOIN Opiskelijat AS o ON ts.opiskelijanumero = o.opiskelijanumero
WHERE ti.kurssikoodi = 'CS-A1150';
```

	opiskelijanumero	opiskelijan_nimi	arvosana
1	10002001	Aamos Rex	3
2	10002001	Aamos Rex	5
3	10002001	Aamos Rex	5
4	10002002	Gabriella Tolvanen	3
5	10002002	Gabriella Tolvanen	2
6	10002002	Gabriella Tolvanen	1
7	10002003	Kalle Berg	5
8	10002003	Kalle Berg	4
9	10002003	Kalle Berg	4
10	10002004	Maija Meikäläinen	5

Lasketaan keskiarvo tentille:

```
SELECT AVG(ts.arvosana) AS keskiarvo
FROM Tentti_suoritukset AS ts
JOIN Tentti_ilmoittautumiset AS ti ON ts.opiskelijanumero = ti.opiskelijanumero
WHERE ti.kurssikoodi = 'CS-A1150';
```

	keskiarvo
1	3.7

:

Käyttötapaukset

Käyttötapaus 5: Opiskelijoiden arvosanajakauma I

Haetaan opiskelijoiden arvosanajakauma kurssilla MS-A0003.

```
SELECT ts.arvosana, COUNT(*) as lukumäärä
FROM Tentti_suoritukset AS ts
JOIN Tentti_ilmoittautumiset AS ti ON ts.opiskelijanumero = ti.opiskelijanumero
WHERE ti.kurssikoodi = 'MS-A0003'
GROUP BY ts.arvosana
ORDER BY ts.arvosana;
```

	arvosana	lukumäärä
1	1	1
2	2	1
3	3	2
4	4	2
5	5	3

Käyttötapaus 6: Opiskelijoiden arvosanajakauma II

Haetaan eri tutkinto-ohjelmien opiskelijoiden arvosanajakauma:

```
SELECT o.tutkinto_ohjelma, AVG(ts.arvosana) AS keskiarvo, COUNT(*) as opiskelijoiden_lukumäärä
FROM Opiskelijat AS o
JOIN Tentti_suoritukset AS ts ON ts.opiskelijanumero = o.opiskelijanumero
GROUP BY o.tutkinto_ohjelma
ORDER BY keskiarvo DESC
```

	tutkinto_ohjelma	keskiarvo	opiskelijoiden_lukumäärä
1	Tietotekniikka	4.5	4
2	Informaatioverkostot	4.33333333333333	3
3	Bioinformaatioteknologia	2	3

Käyttötapaukset

Käyttötapaus 7: Opiskelijoiden opintopisteet

Haetaan opiskelijoiden kokonaisopintopistemäärä laskevassa järjestyksessä

```
SELECT o.opiskelijanumero, o.nimi, SUM(k.opintopistemäärä) as
kokonaisopintopisteet
FROM Opiskelijat AS o
JOIN Tentti_ilmoittautumiset AS ti ON o.opiskelijanumero = ti.opiskelijanumero
JOIN Kurssit AS k ON ti.kurssikoodi = k.kurssikoodi
JOIN Tentti_suoritukset AS ts ON ti.opiskelijanumero = ts.opiskelijanumero
WHERE ts.arvosana > 0
GROUP BY o.opiskelijanumero, o.nimi
ORDER BY kokonaisopintopisteet DESC;
```

	opiskelijanumero	nimi	kokonaisopintopisteet
1	10002001	Aamos Rex	36
2	10002002	Gabriella Tolvanen	36
3	10002003	Kalle Berg	36
4	10002004	Maija Meikäläinen	5

Käyttötapaus 8: Yliopiston salit

Haetaan kaikki salit, niiden paikkamäärät, sekä rakennukset ja katuosoitteet missä ne sijaitsevat. Järjestetään ensiksi katuosoitteen mukaan ja sitten laskevan paikkamäärän mukaan.

```
SELECT Salitunnus, paikkamäärä, nimi AS RakennuksenNimi, katuosoite
FROM Salit, Rakennukset
ORDER BY katuosoite, paikkamäärä DESC;
```

	Salitunnus	paikkamäärä	RakennuksenNimi	katuosoite
1	S03	300	Kandikeskus	Matematiikankatu 3
2	S02	200	Kandikeskus	Matematiikankatu 3
3	S01	100	Kandikeskus	Matematiikankatu 3
4	S03	300	Väre	Tiedontie 2
5	S02	200	Väre	Tiedontie 2
6	S01	100	Väre	Tiedontie 2
7	S03	300	TUAS-talo	Yliopistonkatu 1
8	S02	200	TUAS-talo	Yliopistonkatu 1
9	S01	100	TUAS-talo	Yliopistonkatu 1

Käyttötapaukset

Käyttötapaus 9: Kurssille ilmoittautuneet opiskelijat

Etsitään kaikki opiskelijat, jotka ovat ilmoittautuneet kurssille CS-A1150

```
SELECT O.opiskelijanumero, O.nimi
FROM Opiskelijat O
JOIN OpiskelijaHarjoitusryhmät OH ON O.opiskelijanumero = OH.opiskelijanumero
WHERE OH.kurssikoodi = 'CS-A1150';
```

	opiskelijanumero	nimi
1	10002001	Aamos Rex
2	10002002	Gabriella Tolvanen
3	10002003	Kalle Berg

Käyttötapaus 10: Eniten varauksia tehnyt työntekijä

Etsitään se yliopiston työntekijä, joka on tehnyt kaikista eniten varauksia.

```
SELECT V.varauksen_tekijän_tunnus, COUNT(V.varaustunnus) AS varausten_määrä
FROM Varaus V
JOIN Työntekijät T ON V.varauksen_tekijän_tunnus = T.tunnus
GROUP BY V.varauksen_tekijän_tunnus
ORDER BY varausten_määrä DESC
LIMIT 1;
```

	varauksen_tekijän_tunnus	varausten_määrä
1	1002	3

Käyttötapaukset

Käyttötapaus 11: Kurssi, jolla eniten kurssikertoja

Haetaan kaikki kurssit ja järjestetään ne sen mukaan, kuinka monta kertaa niitä on pidetty.

```
SELECT k.nimi, k.kurssikoodi, COUNT(k.kurssikoodi) AS kertoja_pidetty
FROM Kurssit AS k
JOIN Kurssikerrat AS kk ON k.kurssikoodi = kk.kurssikoodi
GROUP BY k.kurssikoodi
ORDER BY kertoja_pidetty DESC;
```

	nimi	kurssikoodi	kertoja_pidetty
1	Tietokannat	CS-A1150	1
2	Johdatus opiskeluun Sähkötekniikan kandidaattiohjelmassa	ELEC-A0110	1
3	Matriisilaskenta	MS-A0003	1

Käyttötapaus 12: Tutkinto-ohjelmat järjestykseen

Laitetaan tutkinto-ohjelmat laskevaan järjestykseen tutkinto-ohjelma opiskelijoiden keskiarvon kurssilta CS-A1150 perusteella. Näytetään tutkinto-ohjelmat, että niiden opiskelijoiden keskiarvo.

```
SELECT O.tutkinto_ohjelma, AVG(T.arvosana) AS Keskiarvo
FROM Opiskelijat O
JOIN Tentti_suoritukset T ON O.opiskelijanumero = T.opiskelijanumero
WHERE T.kurssikoodi = 'CS-A1150'
GROUP BY O.tutkinto_ohjelma
ORDER BY Keskiarvo DESC
;
```

	tutkinto_ohjelma	Keskiarvo
1	Tietotekniikka	5
2	Informaatioverkostot	3
3	Bioinformaatioteknologia	3

Käyttötapaukset

Käyttötapaus 13: Tutkinto-ohjelmien opiskelijamäärät

Etsitään jokaisen tutkinto-ohjelman opiskelijamäärä.

```
SELECT O.tutkinto_ohjelma, COUNT(*) AS Määrä
FROM Opiskelijat O
GROUP BY O.tutkinto_ohjelma
ORDER BY Määrä DESC;
```

	tutkinto_ohjelma	Määrä
1	Tietotekniikka	2
2	Informaatioverkostot	1
3	Bioinformaatioteknologia	1

Käyttötapaus 14: Yliopiston vaikein kurssi

Etsitään yliopiston vaikein kurssi, eli toisin sanoen kurssi, jonka arvosanojen keskiarvot on matalin.

```
WITH KurssiKeskiarvot AS (
    SELECT T.kurssikoodi, AVG(T.arvosana) AS Keskiarvosana
    FROM Tentti_suoritukset T
    GROUP BY T.kurssikoodi
)
SELECT kurssikoodi, Keskiarvosana
FROM KurssiKeskiarvot
WHERE Keskiarvosana = (SELECT MIN(Keskiarvosana) FROM KurssiKeskiarvot);
```

	kurssikoodi	Keskiarvosana
1	ELEC-A0110	3.33333333333333

Käyttötapaukset

Käyttötapaus 15: Tietyt ehdot täyttävät opiskelijat

Etsitään opiskelijat, joilla on keskiarvo yli tai yhtäsuuri kuin 3 sekä yli 10 opintopistettä. Näytetään opiskelijan opiskelijanumero, tutkinto-ohjelma, keskiarvo sekä opintopistemäärä.

```
WITH Keskiarvot AS (
    SELECT T.opiskelijanumero, AVG(T.arvosana) AS Keskiarvosana
    FROM Tentti_suoritukset T
    WHERE T.arvosana > 0
    GROUP BY T.opiskelijanumero
), Opintopisteet AS (
    SELECT T.opiskelijanumero, SUM(K.opintopistemäärä) AS Opintopistemäärä
    FROM Tentti_suoritukset T
        JOIN Kurssit K ON T.kurssikoodi = K.kurssikoodi
    WHERE T.arvosana > 0
    GROUP BY T.opiskelijanumero
)
SELECT O.opiskelijanumero, O.tutkinto_ohjelma,
Opintopisteet.Opintopistemäärä, Keskiarvot.Keskiarvosana
FROM Opiskelijat O
    JOIN Keskiarvot ON O.opiskelijanumero = Keskiarvot.opiskelijanumero
    JOIN Opintopisteet ON O.opiskelijanumero = Opintopisteet.opiskelijanumero
WHERE Opintopisteet.Opintopistemäärä > 10 AND Keskiarvot.Keskiarvosana >=
3;
```

	opiskelijanumero	tutkinto_ohjelma	Opintopistemäärä	Keskiarvosana
1	10002001	Informaatioverkostot	12	4.33333333333333
2	10002003	Tietotekniikka	12	4.33333333333333

Tekijät:
Tatu Rouhiainen - tatu.rouhiainen@aalto.fi
Axel Andersson - axel.w.andersson@aalto.fi
Benjamin Hadaya -benjamin.hadaya@aalto.fi

Aalto Yliopisto