Tampereen yliopisto
Tampere University

Tietokantajärjestelmät

Esimerkki ER-kaavion evoluutiosta

Tässä esityksessä tarkastellaan yksinkertaisen esimerkin avulla, että kuinka sama asia voidaan mallintaa erilailla ja kuinka ER-malli kehittyy, kun otetaan uusia asioita huomioon.

Tampereen yliopisto
Tampere University

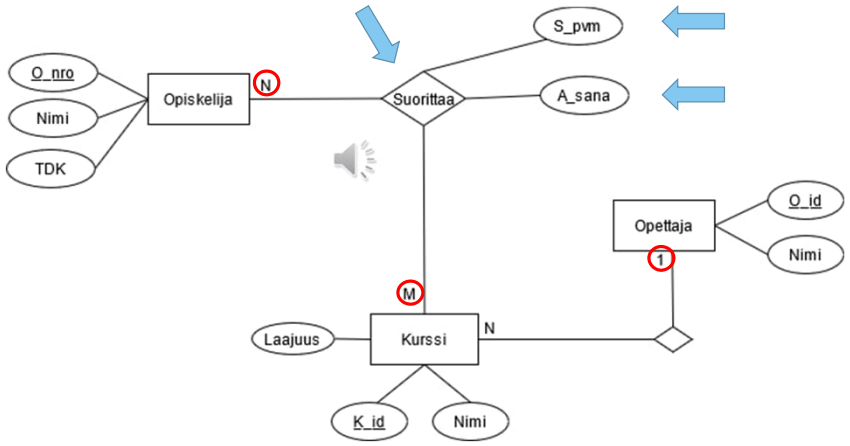
Opiskelijat suorittavat kursseja, joita eri opettajat järjestävät. Opiskelija saa suorittamastaan kurssista arvosanan. Oletetaan aluksi, että kurssilla on vain yksi opettaja.

- Miten kaavio muuttuu, kun
  - Kurssilla on enemmän kuin yksi opettaja
  - Kurssille pitää myös ilmoittautua
  - Erotellaan kurssi ja sen toteutus
  - Poistetaan redundanttisuutta
- Tarkastellaan askel kerrallaan
- Esitetty malli ei ole ’ainoa oikea’

Tehtävänä on mallintaa opiskelijoiden kurssisuorituksia. Oletetaan aluksi, että kurssilla on yksi opettaja. Tehdään tästä yksinkertainen malli ja katsotaan, että mitä muutoksia malliin pitää tehdä, kun lähtöoletukset laajentuvat. Tarkastellaan muutoksia askel kerrallaan.

Tampereen yliopisto
Tampere University

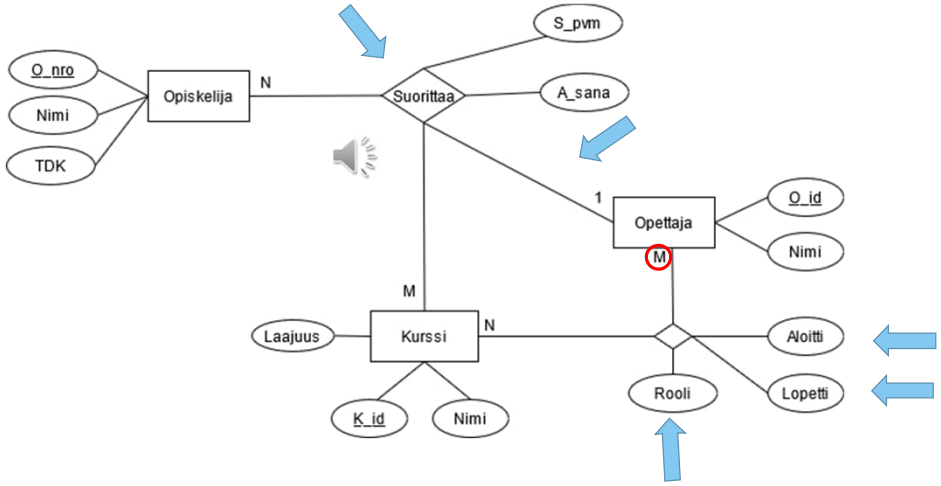
Lähtötilanne



Lähtökohtaisesti kurssin suoritus voidaan toteuttaa suhteena kurssin ja opiskelijan välillä. Opiskelija voi suorittaa monta kurssia ja samalla kurssilla on voi olla useita opiskelijoita. Suoritukseen liitetään attribuutit päivämäärä ja arvosana. Laajennetaan mallia siten, että kurssilla voi olla useampi opettaja.

Tampereen yliopisto
Tampere University

Kurssilla monta opettajaa; vain yksi antaa merkinnän



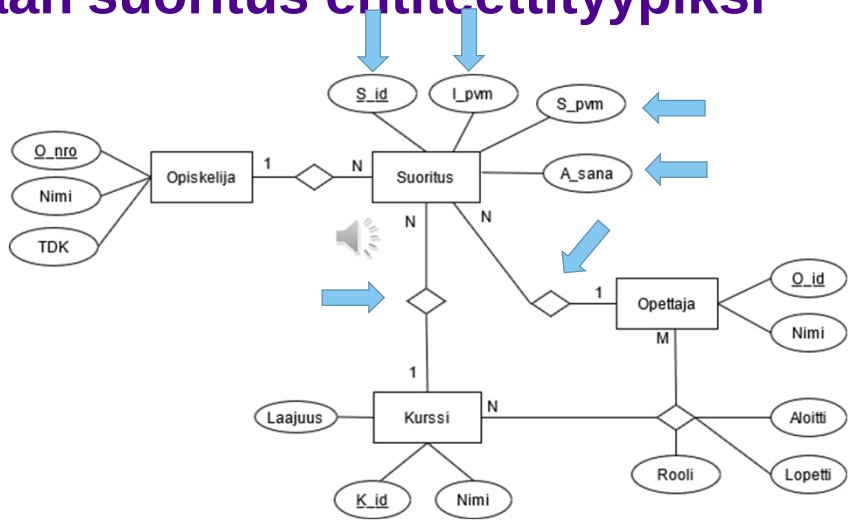
Muutoksessa ei riitä, että vain vaihdetaan lukumääräsuhde, vaan myös muita lisäyksiä on tehtävä. Kun opettaja oli vain yksi, niin oli selvää, että kuka kirjasi suorituksen. Nyt suorittaa-suhteeseen on liitettävä opettaja, joka tekee kirjauksen. Lisäksi kurssin ja opettajan väliseen suhteeseen on syytä lisätä attribuutit, että missä roolissa opettaja kurssilla on sekä vastuun aloitus- ja lopetusajankohdat.

Mietitään seuraavaksi, että mitä vaikuttaa, jos järjestelmään halutaan tallentaa myös kurssille ilmoittuminen. Tämä olisi mahdollista toteuttaa omana suhteena kurssin opiskelijan väliin, jolloin tietokantaan tulisi oma taulu. Säästetään kuitenkin tilaa ja tallennetaan ilmoittautuminen samaan yhteyteen kuin kurssin suoritus, eli että tietokannasta löytyisi samasta taulusta sekä ilmoittuminen että kurssin suoritus.

Ilmoittautuminen voitaisiin siis liittää suorittaa-suhteen yhteyteen. Suorittaa-suhde ei kuitenkaan ole luonteva paikka lisätä ilmoittumistietoja, koska suhde syntyy vasta suorituksen yhteydessä. Korvataan siis suhde entiteettityypillä.

Tampereen yliopisto
Tampere University

Muunnetaan suoritus entiteettityypiksi

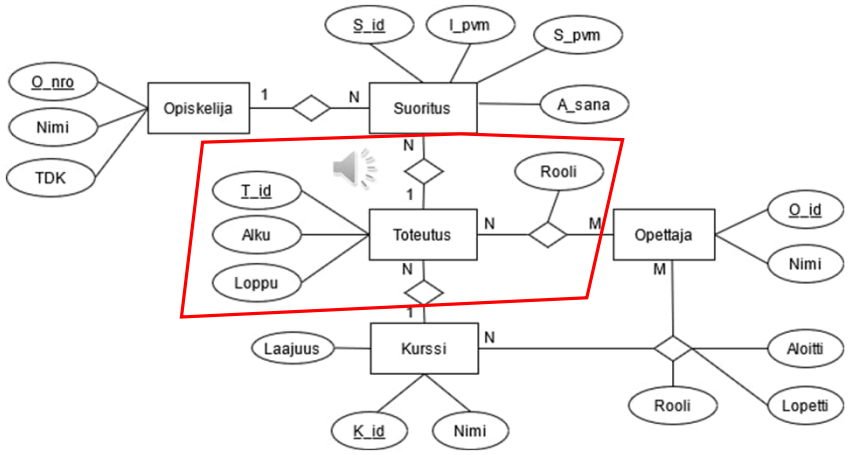


Nyt ajatus on, että suoritus-entiteetti voidaan luoda ja identifioida jo ilmoittumisen yhteydessä. Kun kurssi suoritetaan, niin suorituspäivä ja arvosana arvotetaan. Jos ne jäävät tyhjiksi, niin suoritus on jäänyt kesken.

Malli on jo aika pitkällä, mutta se ei ole SISU-yhteensopiva. Opiskelija ei ilmoittaudu SISUssa kurssille vaan toteutukselle. Tehdään siis ero kurssille ja sen toteutukselle.

Tampereen yliopisto
Tampere University

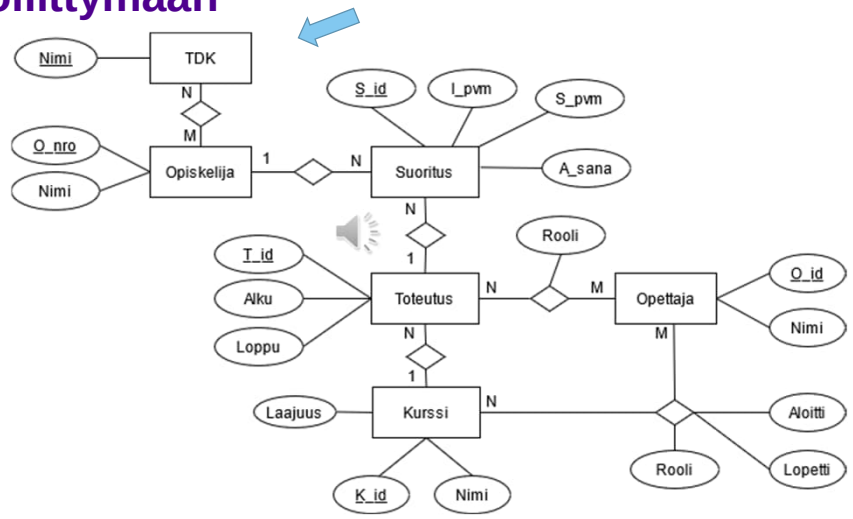
Erotellaan kurssi ja sen toteutus



Kaavio muuttuu siten, että suoritus ja ilmoittautuminen on linkitetty toteutukseen, jolla on alku- ja loppuajankohdat. Opettaja on suhteessa toteutukseen ja nyt oletetaan, että opettajan rooli kertoo kirjausvastuun. Opettaja on edelleen suhteessa myös itse kurssiin, eli opettaja saattaa olla vastuussa kurssista, vaikka ei osallistu sen toteutukseen.

Tampereen yliopisto
Tampere University

Tiedekunta omaksi tyypiksi: Poistaa redundanttisuutta ja mahdollistaa valikon käyttöliittymään



Tehdään lopuksi pieni muunnos, eli muodostetaan tiedekunnasta oma tyyppinsä. Aina ei ole selvää, että milloin joku asia pitäisi ilmaista attribuuttina ja milloin omana tyyppinään. Tähän voi antaa kaksi sääntöä. Tarkastellaan näitä tämän esimerkin kautta. Ensinnäkin, jos tiedekunnasta halutaan tallentaa muitakin ominaisuuksia kuin pelkkä nimi, niin siitä pitää tehdä oma entiteettityyppi. Tämähän on mallintamisen perusteita. Toinen asia tulee käytännön tarpeesta. Jos tiedekunta on opiskelijan attribuutti se pitää kirjoittaa aina kun uusi opiskelija lisätään. Jos sen sijaan käyttäjälle halutaan tarjota valikko, jossa näkyy kaikki tiedekunnat, niin silloin tiedekunta pitää toteuttaa omana entiteettityyppinä, jota vastaavaan tauluun arvot voidaan etukäteen tallentaa tietokantaan.

Tampereen yliopisto
Tampere University

Kuinka homma jatkuisi

- Jos halutaan määritellä, että mitä rooleja opettajalla voi olla, niin roolista pitää tehdä oma entiteettityyppi.
- Jos halutaan, että käytössä on arvosanavalikoima, niin siitä pitää tehdä oma entiteettityyppinsä.
- Jos suoritus voi poiketa kurssin opintopisteistä, niin suoritukseen pitää liittää attribuutti laajuus.
- Kaavio laajenee, kun siiten liitetään oppiaineen tiedot tai edeltävyystiedot.

Mietitään lopuksi, että kuinka homma laajenisi, jos mallintamista jatkettaisi. Ensinnäkin, opettajalla on attribuuttina rooli. Käytännössä eri roolit on etukäteen määritelty, eli ne pitäisi tallentaa tietokantaan. Mallintamisessa tämä tarkoittaa, että roolista pitäisi tehdä oma entiteettityyppi.

Vähän vastaavasti: jos halutaan antaa arvosanavalikoima etukäteen, niin arvosanasta voidaan tehdä oma tyyppinsä. Tässä tosin pitää miettiä, että jos arvosanat ovat yhdestä viiteen, niin tämä voidaan toteuttaa helposti käyttöliittymässä ilman, että lukuja haetaan tietokannasta. Jos sen sijaan arvosanat ovat impropaatturi, apropaatturi, lupentteri ja niin edelleen, ne on syytä tallentaa etukäteen tietokantaan.

Esimerkkimallissa opintopistelaajuus oli sidottu suoritukseen. Jos käytännön tilanne olisi kuitenkin sellainen, että kurssista voi saada erisuuruisia suorituksia, niin laajuus pitäisi liittääkin suoritukseen.

Lopuksi voidaan todeta, että kaavio yhä laajenee, kun siihen integroidaan oppiaineen tiedot ja edeltävyystiedot. Edeltävyydet voivat olla suositeltuja tai pakollisia. Oppaine voi liittyä opiskelijaan, kurssiin, opettajaan ja toteutukseen. Jos näitä ei mallinneta huolellisesti, niin vaarana on, että opintotietojärjestelmästä tulee hankalakäyttöinen. Näinhän ei onneksi tänä päivänä pääse käymään -ainakaan Tampereen yliopistossa.