Tietokantasuunnittelusta

- Tietokantasuunnittelun pääperiaatteena on tiedon toiston välttäminen.
- Tiedon toistumiseen liittyy monenlaisia ongelmia
  - toistuva tieto vie tilaa
  - ylläpito muodostuu hankalaksi
  - ylläpito-operaatioilla voi olla odottamattomia sivuvaikutuksia.

Tietokantasuunnittelusta

- Esimerkki taulusta, joka ei käyttäydy hyvin:

EMP_DEPT:

E_no	E_name	E_bdate	D_no	D_name	D_location
1	M.Seppä	1.3.59	3	Myynti	Helsinki
2	D.Leivo	4.10.40	3	Myynti	Helsinki
3	K.Koivu	30.1.66	4	Hallinto	Lahti
4	B.Oja	2.5.65	4	Hallinto	Lahti
5	O.Itä	10.2.55	6	Tuotanto	Helsinki

Avain: E_no

Jos O.Itä poistetaan, häviää tieto tuotanto-osastosta

Toistettava jokaisen osaston tt:n kohdalla

Jos Hallinto muuttaa Espooseen om muutettava useita rivejä

Tietokantasuunnittelusta

- Tietokannan suunnittelun vaiheita ovat
  - tietosisällön kartoitus
    - luokkakaaviona (Johdatus sovellussuunnitteluun)
    - kartoittamalla attribuutit ja niiden väliset riippuvuudet
  - loogisten rakenteiden suunnittelu ja
  - teknisten rakenteiden suunnittelu.

- **Loogisen rakenteen suunnittelun tavoite on sijoittaa yhteenkuuluvat tiedot samaan tauluun**

Muunnos luokkakaaviosta relaatiokaavioon

- Järjestelmän määrittelyn yhteydessä järjestelmän tietosisältö voidaan määrittellä luokkakaavion avulla
- Luokkakaavio kuvaa tällöin kuvaa järjestelmän pysyväisluonteisia (persistent) tietoja - tietojen on säilyttävä ohjelman suorituskertojen välillä
- Luonteva paikka säilyttää olioiden tila on tietokanta:
  - oliorakenteinen tietokanta (oliotietokanta)
  - relaatiotietokanta

Muunnos luokkakaaviosta relaatiokaavioon

- relaatiotietokanta
  - standardoitu SQL ja liittymät perinteisiin ohjelmointikieliin
  - ei suoraa tukea olioiden välisille yhteyksille
  - ei luokkahierarkiaa eikä periytymistä
  - vain data ei toimintoja
  - uusimmissa tkhj:ssä mukana oliopiirteitä
  - **edellyyttää muunnosta oliorakenteista relaatiorakenteiksi**

Muunnos luokkakaaviosta relaatiokaavioon

- Lähtökohtana normalisoitu luokkakaavio = **tiettyjä sääntöjä noudattava kaavio**
  - Yhteydet näkyviä, ei attribuuteiksi piilotettuja
  - Kukin asia esitetään vain kertaalleen
  - Ei johdettavissa / pääteltävissä olevaa tietoja
  - Yhteydet määritelty oikeiden osapuolten välillä

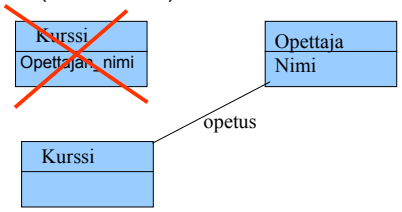
Yhteydet piilotettu

- Olioiden välinen kytkentä esitetään aina yhteytenä, sitä ei piiloteta ominaisuudeksi (attribuutiksi)

Kytkentä on piilotettuna kurssin attribuuttiin opettajan_nimi

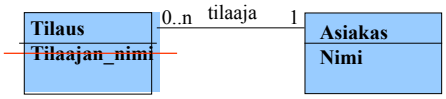
Yhteyden piilotus

- Olioiden välinen kytkentä esitetään aina yhteytenä, sitä ei piiloteta ominaisuudeksi (attribuutiksi)



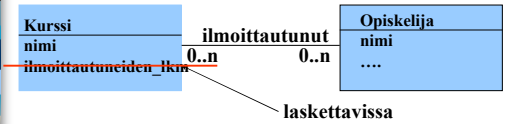
Asiat esitetään vain kertaalleen

- Samaa asiaa ei pidä esittää sekä yhteytenä että ominaisuutena

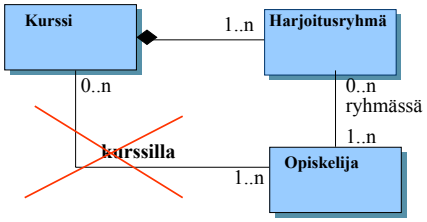


Ei esitetä johdettavissa olevaa tietoa

- tulisi selvittää mitkä tiedot ovat perustietoja ja mitkä voidaan päätellä (laskea) muiden tietojen perusteella
- attribuutin arvo tai yhteyden olemassaolo voi olla pääteltävissä



Ei esitetä johdettavissa olevaa tietoa

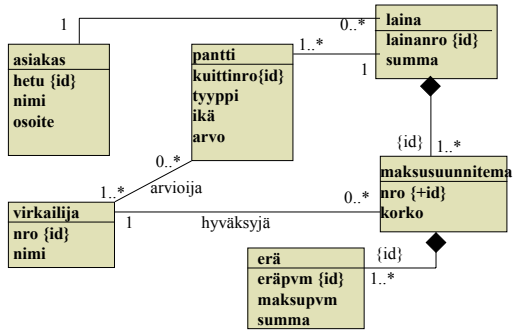


Kurssilla-yhteys on pääteltävissä, jos jokaisen kursillaolijan on oltava jossain ryhmässä

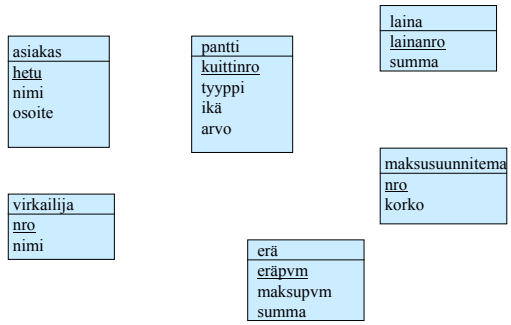
Muunnoksen perussäännöt:

- Kutakin olioluokkaa vastaa samanniminen taulu
- Kullakin luokan yksiarvoisella attribuutilla on samanniminen vastinsarake luokkaa vastaavassa taulussa
  - Luokan tunnistavia attribuutteja vastaavat sarakkeet kuuluvat luokkaa vastaavan taulun avaimeen

Luokkakaavio: Vippi oy



Relaatiorakenteen ydin - jokaista luokkaa vastaa taulu

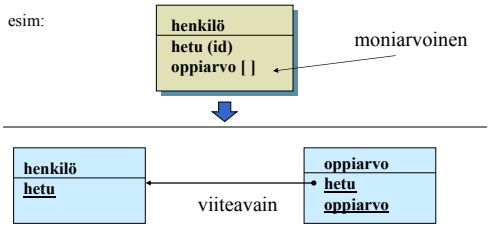


Kokoelma-attribuutit

- Kutakin kokoelma-attribuuttia (moniarvoista) vastaa taulu, jonka sarakkeina ovat
  - viiteavain kokoelmatyyppisen attribuutin sisältävää luokkaa vastaavaan tauluun,
  - sarake attribuutin arvoa varten.
- Taulun kaikki sarakkeet kuuluvat avaimeen

Kokoelma-attribuutit

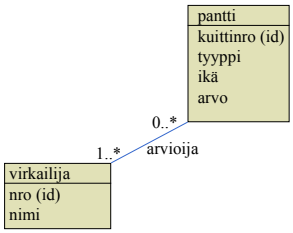
- Vippi oy:n esimerkissä ei ole kokoelma-attribuutteja



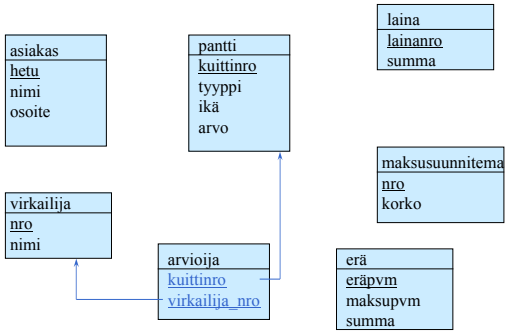
Monen suhde moneen yhteydet

- Kutakin monen suhde moneen yhteyttä vastaa taulu
  - Taulun nimi = yhteyden nimi
  - Taulun sarakkeina ovat yhteyden osapuoliin osoittavat viiteavaimet.
  - Taulun kaikki sarakkeet kuuluvat taulun pääavaimeen
  - monen suhde moneen yhteys on yhteys, jossa kummankin osapuolen maksimiosallistumisrajoite on suurempi kuin 1.

Monen suhde moneen yhteydet



Monen suhde moneen yhteydet



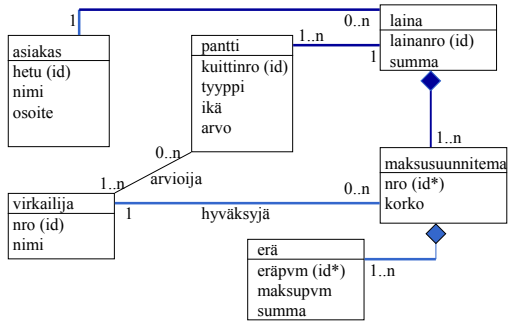
Yhden suhde moneen yhteydet

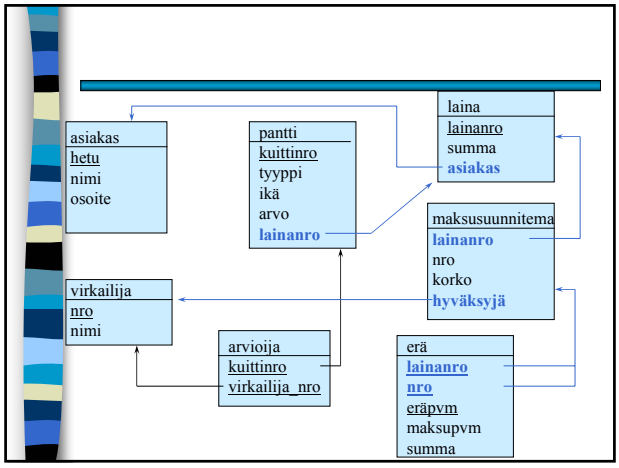
- Vaihtoehto 1:
- Lisää yhteyskumppaniin osoittava viiteavain sitä osapuolta vastaavaan tauluun, jolla on korkeintaan yksi kumppani
  - Jos kookoonpanoyhteydessä kokoonpanon osien tunnistamiseen tarvitaan tieto kokonaisuudesta, niin lisätty viiteavain kuuluu taulun avaimeen. Tällaisessa tilanteessa ei ole muita vaihtoehtoja.

Yhden suhde moneen yhteydet

- Vaihtoehto 2:
- Toimi kuten monen suhde moneen yhteyksien kohdalla ja tee yhteyttä varten erillinen taulu
- Taulujen määrän minimoimiseksi vaihtoehtoa 1 tulisi suosia, ellei siitä seuraa epäedullisia viittaussyklejä taulujen välille.

Yhden suhde moneen yhteydet

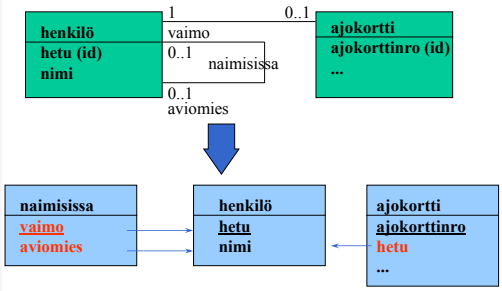




Yhden suhde yhteen yhteydet

- Vaihtoehtoja:
- Lisää yhteyskumppaniin osoittava viiteavain yhteyden toista osapuolta vastaavaan tauluun
  - ensisijaisesti sille, jolle yhteys on pakollinen
  - jos yhteys on pakollinen molemmille, valitse kumpi tahansa
- Tee erillinen taulu, kuten monen suhde moneen tapauksessa.
  - jos yhteys on molemmille osapuolille valinnainen

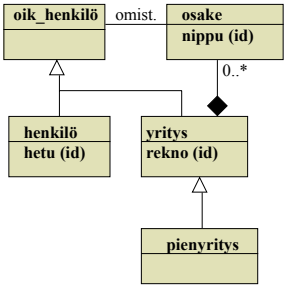
Yhden suhde yhteen yhteydet



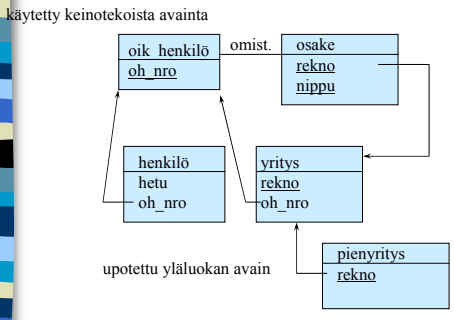
Yleistyshierarkia

- Vaihtoehto 1:
- Taulu sekä ala- että yläluokalle. Alaluokan tauluun sijoitetaan yläluokan tauluun osoittava viiteavain.
  - Jos yläluokalla on käyttäjän määrittelemiä tunnistavia attribuutteja otetaan viiteavain alaluokan taulun avaimeksi.
  - Jos yläluokan taululla ei ole käyttäjän määrittämää tunnistavaa attribuuttia, joudutaan sille luomaan keinotekoinen avain. Tätä ei oteta alaluokan taulun avaimeksi, jos alaluokalle on määritelty tunnistavia attribuutteja.

Yleistyshierarkia

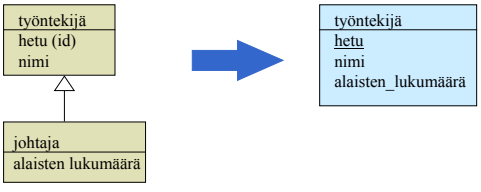


Yleistyshierarkia



Yleistyshierarkia

- Vaihtoehto 2: Alaluokalle ei tehdä omaa taulua, vaan alaluokan tauluun loogisti kuuluvat sarakkeet upotetaan yläluokan tauluun.



Tietokantasuunnittelusta

- Relaatiomalliin liittyvässä suunnitteluteoriassa yhteenkuuluvuus määritellään **riippuvuuksien** avulla
- Keskeinen riippuvuus: **funktionaalinen riippuvuus**
- Attribuutti B on funktionaalisesti riippuva attribuutista A (A määrää funktionaalisesti B:n), jos ja vain jos kaikissa relaatiokaavion R ilmentymissä kuvaus A:n arvojoukolta B:n arvojoukolle on funktionaalinen.

Tietokantasuunnittelusta

- Kuvaus f:v(A)->v(B) on funktionaalinen, jos kaikissa relaatiokaavion R ilmentymissä jokainen A:n arvo kuvautuu yhdelle B:n arvolle eli,
  - jos riveillä r ja s attribuutilla A on sama arvo (r.A=s.A), niin näillä riveillä täytyy myös B-attribuuteilla olla keskenään sama arvo (r.B=s.B).
  - Funktionaalinen riippuvuus tarkoittaa sitä, että attribuutin B arvo on yksikäsitteisesti selvitettävissä kun tiedetään attribuutin A arvo. Selvittäminen voisi tapahtua kyselyllä
  - select distinct B from R where A=a; (tuloksena olisi enintään yksi rivi)

Tietokantasuunnittelusta

- Funktionaalista riippuvuutta, jossa A määrää B:n merkitään A→B. Attribuuttia A kutsutaan määrääjäksi.
- Yksittäisen attribuutin A tilalla voi olla myös attribuuttiyhdistelmä. Tavoitteena on kuitenkin löytää yhdistelmät, joissa on minimaalinen määrä attribuutteja,
- Jos A→B, voidaan määrääjään lisätä mikä tahansa attribuutti x ja pätee Ax→B.

Tietokantasuunnittelusta

- Tarkastellaan relaatiokaaviota
- Kurssilainen(
- Kurssikoodi,
- Hetu,
- OpiskelijaNimi,
- KurssiNimi,
- TehtavaLkm).
- Oletetaan että relaatio sisältää tietoja useista opiskelijoista ja useista kursseista ja kuvaa opiskelijoiden osallistumista kursseille.

Tietokantasuunnittelusta

- **Hetu-->OpiskelijaNimi**
  Yhtä henkilötunnusta kohden on vain yksi Opiskelijanimi
- **Kurssikoodi→KurssiNimi**
  Yhtä kurssikoodia kohti on vain yksi kurssinimi
- **Hetu, Kurssikoodi --> TehtäväLkm**
  - Yhtä henkilötunnus – kurssikoodi yhdistelmää kohti on vain yksi tehtävälukumäärä = Henkilöllä on kurssikohtainen tehtävämäärä
    - Yhtä henkilötunnusta kohti voi olla useita tehtävämääriä (opiskelija on monella kurssilla)
    - Yhtä kurssikoodia kohti voi olla monta tehtävämäärää (kurssilla on monta opiskelijaa)

Tietokantasuunnittelusta

- **Mitkä sarakkeet samaan relaatiokaavioon?**
- **Erilaisia kriteerejä**
- Yhteenkuuluvuussääntönä, ns. Boyce-Codd normaalimuodon sääntö:
  - relaatiokaavion R attribuutit kuuluvat yhteen, jos ja vain jos relaatiokaavioon R ei liity yhtään sellaista funktionaalista riippuvuutta, jossa määrääjä ei sisältäisi relaation avainta.
- Kurssilainen relaation avain on pari Kurssikoodi, Hetu
- **=> Kaikki attribuutit eivät BC-säännön mukaan kuulu yhteen koska**
  - **Hetu-->OpiskelijaNimi ja**
  - **Kurssikoodi→KurssiNimi rikkovat BC-sääntöä**

Tietokantasuunnittelusta

- Uudelleenjärjesteltävä relaatiokaavioiksi, joissa yhteenkuuluvuussäännöt ovat voimassa.
- päädytään relaatiokaavioihin
  - Opiskelija(Hetu, OpiskelijaNimi)
  - Kurssi(Kurssikoodi, KurssiNimi)
  - Osallistuminen(Hetu, Kurssikoodi, TehtäväLkm).

Jako Boyce-Codd normaalimuotoon

- Miten jaetaan:
  - Karsi riippuvuusjoukko minimaaliseksi – poista johdettavissa olevat riippuvuudet
  - Ryhmitä riippuvuudet yhteisen määrääjän (vasen puoli) perusteella.
  - Muodosta jokaista ryhmää kohti oma relaatio
  - Jos alkuperäisen kaavion yksikään avain ei sisälly muodostuneisiin relaatiokaavioihin tee sille oma kaavio
  - Anna relaatiokaavioille nimet

Tietokantasuunnittelusta

- **Tilauslomaketta analysoitaessa löydettiin seuraavat attribuutit:**
- lomakenumero,
- tilaajan tunnus,
- tilaajan nimi,
- tilaajan osoite,
- tilaajan puhelinnumero,
- toimitusosoite,
- rivinumero,
- tavaran koodi,
- tavaran nimi,
- tilattu määrä, ja
- tilauspäivä.

Tietokantasuunnittelusta

- lomakenumero → tilaajan tunnus (lomakkeella voidaan ilmoittaa vain yksi tilaaja)
- tilaajan tunnus → tilaajan nimi (tunnus identifioi tilaajan, joten sen perusteella saamme selville kaikki tilaajaan liittyvät tiedot)
- tilaajan tunnus → tilaajan osoite,
- tilaajan tunnus → tilaajan puhelinnumero,
- lomakenumero → toimitusosoite (lomakkeella voidaan tilata tavaroita vain yhteen paikkaan ja sama tilaaja voi tilata eri paikkoihin)

Tietokantasuunnittelusta

- tavaran koodi → tavaran nimi (tavaran koodi on tavaran tunniste, jonka kautta päästään kaikkiin tavaran tietoihin)
- lomakenumero, rivinumero → tavaran koodi (lomakkeen rivillä voi ilmoittaa yhden tilattavan tavaran koodin)
- lomakenumero, rivinumero → tilattu määrä (lomakkeen rivillä voi ilmoittaa yhden tilattavan tavaran määrän)
- lomakenumero → tilauspäivä

Tietokantasuunnittelusta

- Jako relaatioihin yhteisen määrääjän perusteella:

- X(tilaajan tunnus, tilaajan nimi, tilaajan osoite, tilaajan puhelinumero)
- Y(tavaran koodi, tavaran nimi)
- Z(lomakenumero,tilauspäivä,tilaajan_tunnus)
- T(lomakenumero, rivinumero, tavaran koodi, tilattu määrä)

Tietokantasuunnittelusta

- Jos kaavioille löytyy kuvaava nimi jako on onnistunut:

- tilaaja(tilaajan tunnus, tilaajan nimi, tilaajan osoite, tilaajan puhelinumero)
- tavara(tavaran koodi, tavaran nimi)
- tilaus(lomakenumero,tilauspäivä,tilaajan_tunnus)
- tilausrivi(lomakenumero, rivinumero, tavaran koodi, tilattu määrä)

Tietokantasuunnittelusta