

Tietokantasuunnittelusta

- Tietokantasuunnittelun pääperiaatteena on tiedon toiston välttäminen.
- Tiedon toistumiseen liittyy monenlaisia ongelmia
 - toistuva tieto vie 'turhaa' tilaa
 - ylläpito muodostuu hankalaksi
kaikki 'kopiot' päivitettävä
 - ylläpito-operaatioilla voi olla odottamattomia sivuvaikutuksia.
- Toisaalta toistosta on etuakin – tiedon haku saattaa nopeutua

1

Tietokantasuunnittelusta

- Esimerkki taulusta, joka ei käyttyä 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

2

Ei tarpeetonta toistoa

E_no	E_name	E_bdate	D_no
1	M.Seppä	1.3.59	3
2	D.Leivo	4.10.40	3
3	K.Koivu	30.1.66	4
4	B.Oja	2.5.65	4
5	O.Itä	10.2.55	6

D_no	D_name	D_location
3	Myynti	Helsinki
4	Hallinto	Lahti
6	Tuotanto	Helsinki



Parempi ratkaisu

Tietokantasuunnittelusta

- Tietokannan suunnittelun vaiheita ovat
 - tietosisällön kartoitus
 - luokkakaaviona (Johdatus sovellussuunnitteluun)
 - kartoittamalla attributit ja niiden väliset riippuvuudet
 - loogisten rakenteiden suunnittelu ja
 - teknisten rakenteiden suunnittelu.
- **Loogisen rakenteen suunnittelun tavoite on sijoittaa yhteenkuuluvat tiedot samaan taulu**

Muunnos luokkakaaviosta relaatiokaavioon

- Järjestelmän määrittelyn yhteydessä järjestelmän tietosisältö voidaan määrittellä luokkakaavion avulla
- Luokkakaavio kuvaa tällöin 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
 - **edellyttää muunnosta oliorakenteista relaatorakenteiksi**

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

7

Yhteydet piilotettu

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

Kurssi
Opettajan_nimi

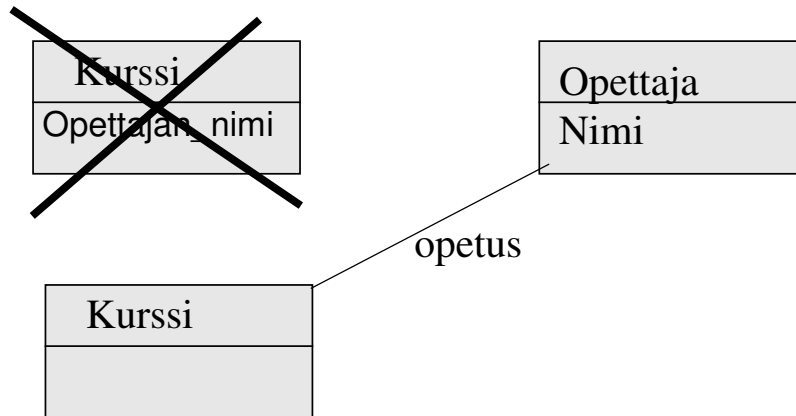
Opettaja
Nimi

Kytkenä on piilotettuna kurssin attribuuttiin opettajan_nimi

8

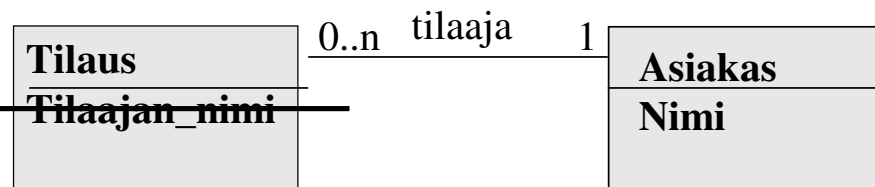
Yhteyden piilotus

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



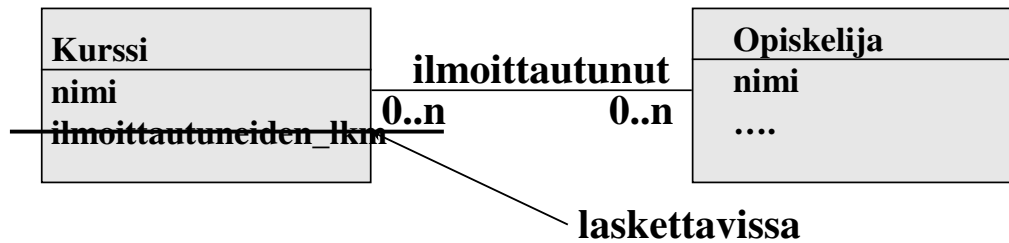
Asiat esitetään vain kertaalleen

- Samaa asiaa ei pidä esittää sekä yhteytenä että ominaisuutena



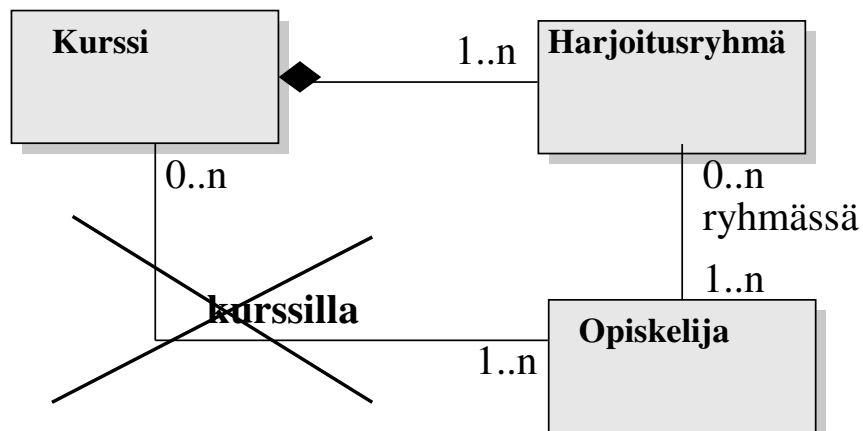
Ei esitetä johdettavissa olevaa tietoa

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



11

Ei esitetä johdettavissa olevaa tietoa



Kurssilla-yhteys on pääteltävissä, jos jokaisen kurssin osanottajan on oltava jossain ryhmässä

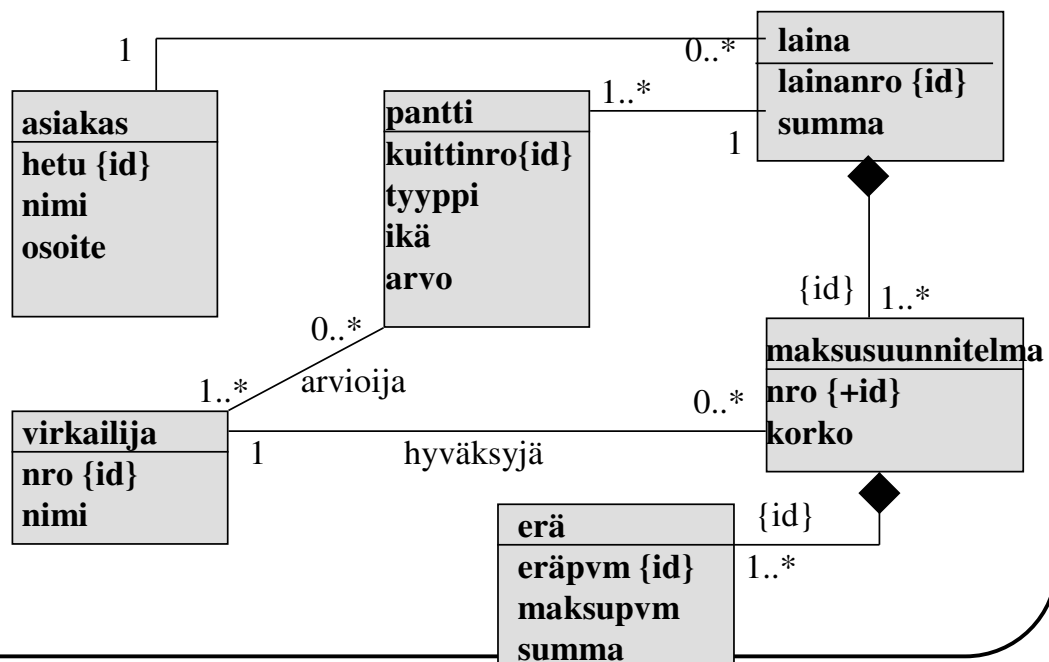
12

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 avaimiin

13

Luokkakaavio: Vippi oy



14

Relaatorakenteen ydin - jokaista luokkaa vastaa taulu

asiakas
<u>hetu</u>
nimi
osoite

pantti
<u>kuittinro</u>
tyyppi
ikä
arvo

laina
<u>lainanro</u>
summa

virkaileija
<u>nro</u>
nimi

maksusuunnitelma
<u>nro</u>
korke

erä
<u>eräpvm</u>
maksupvm
summa

15

Kokoelma-attribuutit

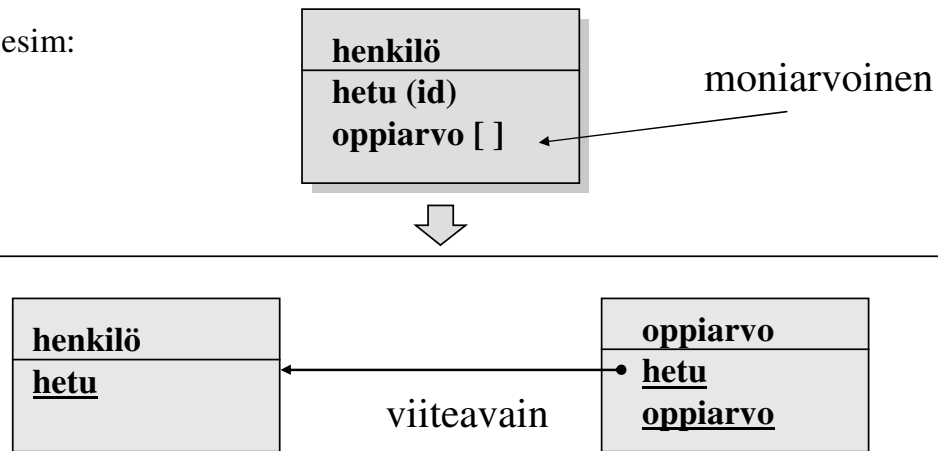
- Kutakin kokoelma-attribuuttia (moniarvoista) vastaa taulu, jonka sarakkeina ovat
 - viiteavain kokoelmatyyppisen attribuutin sisältävää luokkaa vastaavaan tauluun,
 - sarake attribuutin arvoa varten.
- Taulun kaikki sarakkeet kuuluvat avaimeen

16

Kokoelma-attribuutit

- Vippi oy:n esimerkissä ei ole kokoelma-attribuutteja

esim:



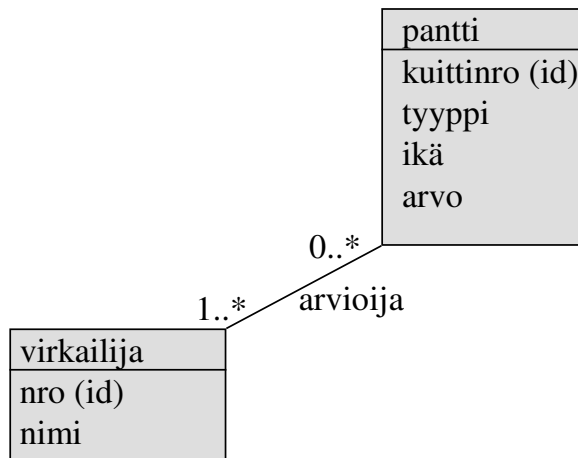
17

Monen suhde moneen yhteydet

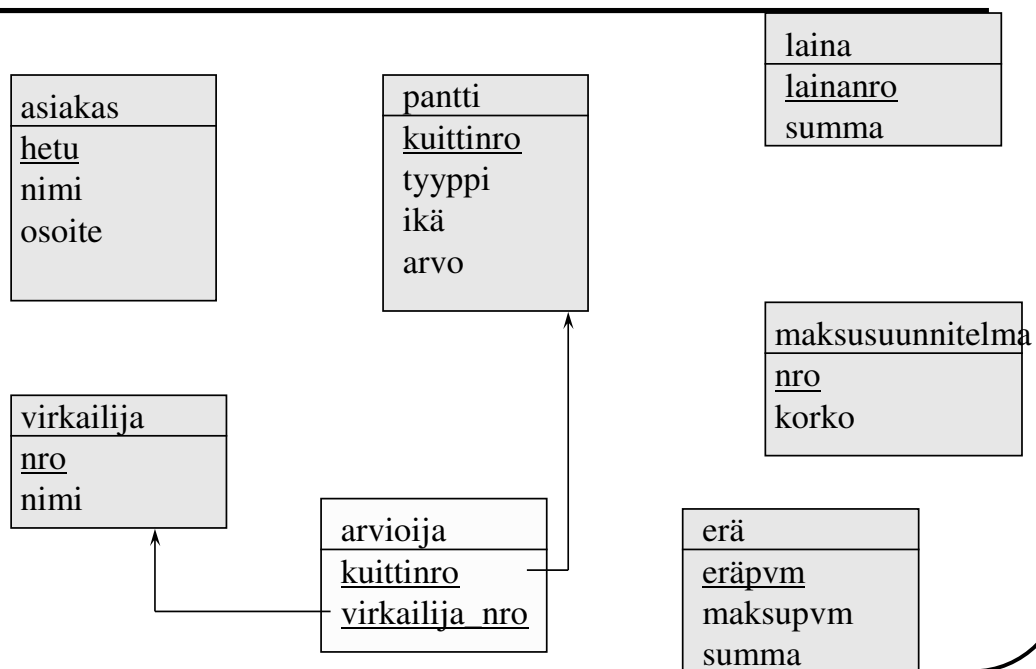
- Kutakin monen suhde moneen yhteyttä vastaa taulu
 - Taulun nimi = yhteyden nimi
 - Taulun sarakkeina ovat yhteyden osapuoliin osoittavat viiteavaimet.
 - Taulun kaikki sarakkeet kuuluvat taulun pääavaimeen
 - monen suhde moneen yhteys on yhteys, jossa kummankin osapuolen maksimiosallistumisrajoite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

18

Monen suhde moneen yhteydet



Monen suhde moneen yhteydet



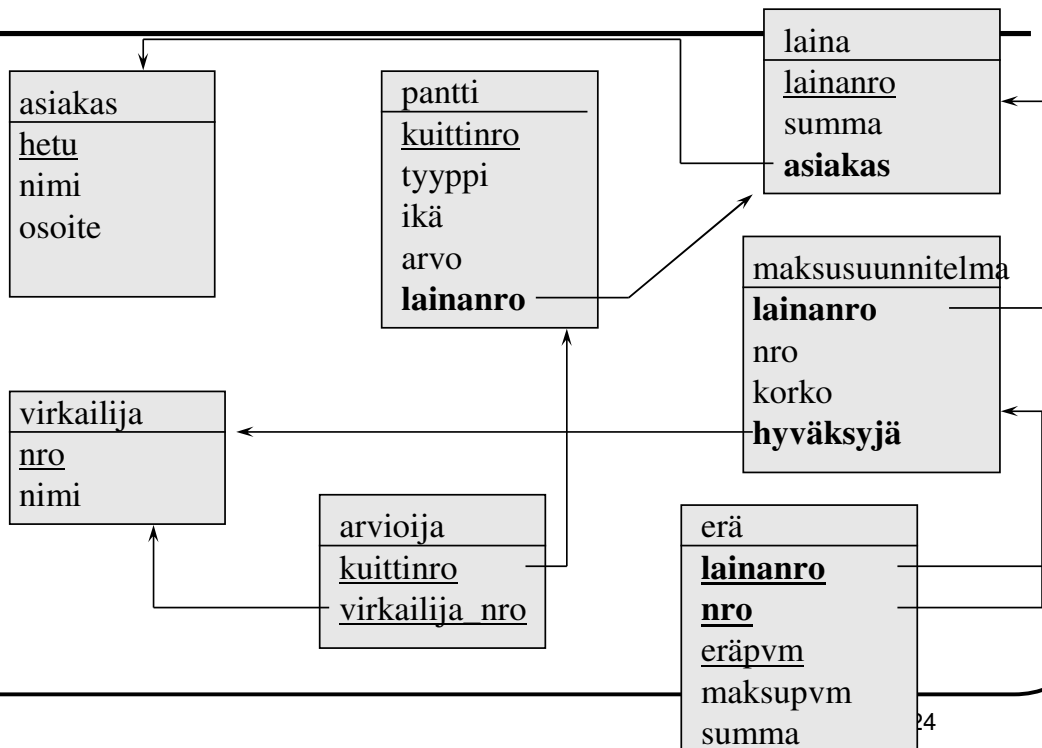
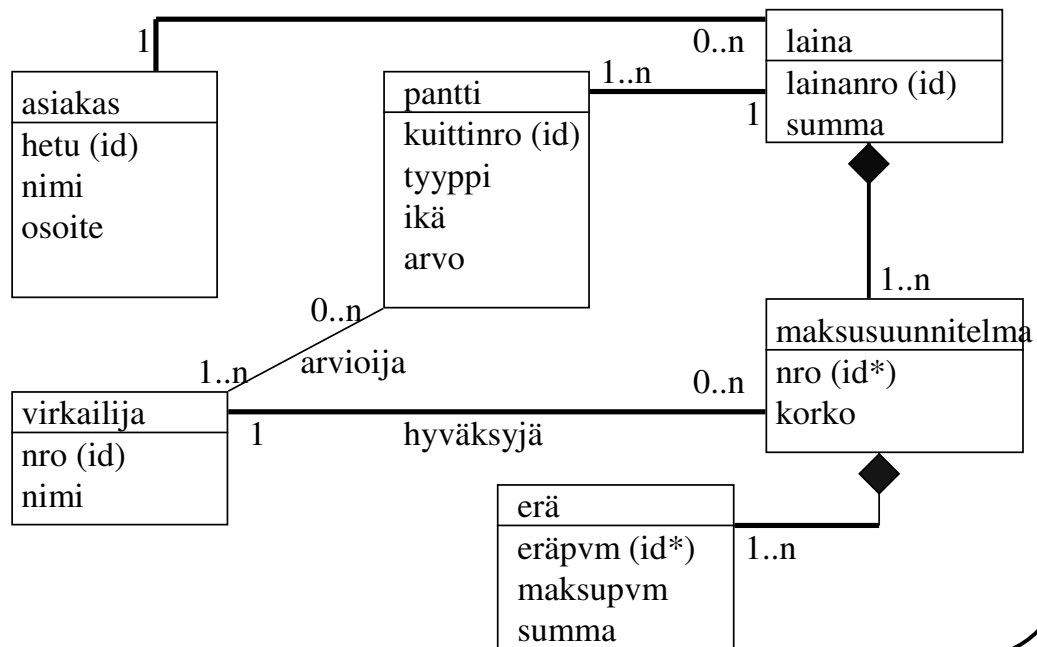
Yhden suhde moneen yhteydet

- Vaihtoehto 1:
- Lisää yhteyskumppaniin osoittava viiteavain sitä osapuolta vastaavaan tauluun, jolla on korkeintaan yksi kumppani
 - Jos kokoonpanoyhteydessä 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

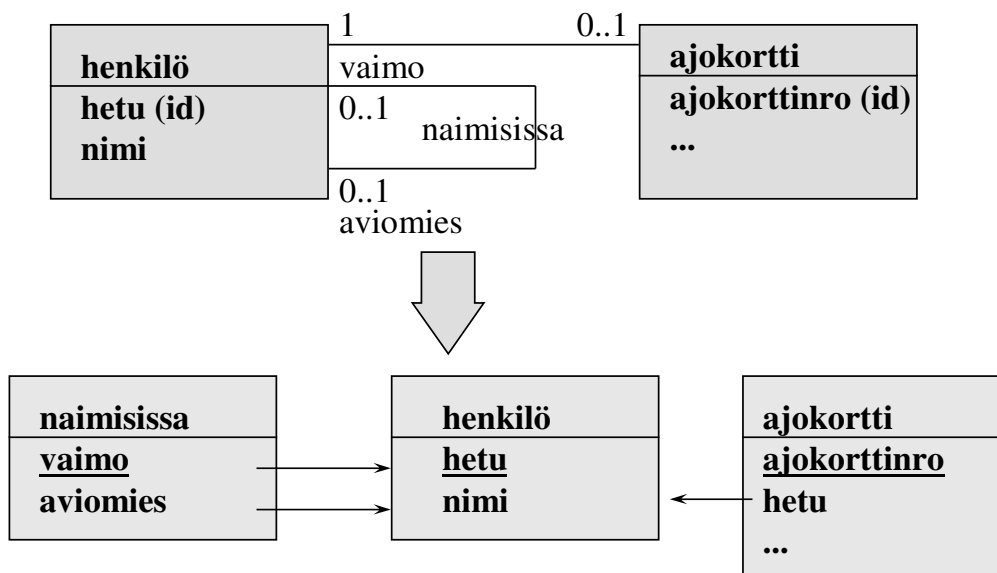


Yhden suhde yhteen yhteydet

- 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

25

Yhden suhde yhteen yhteydet



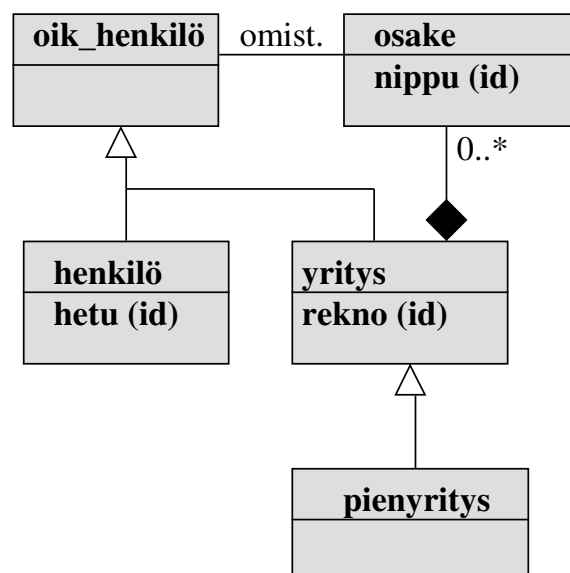
26

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.

27

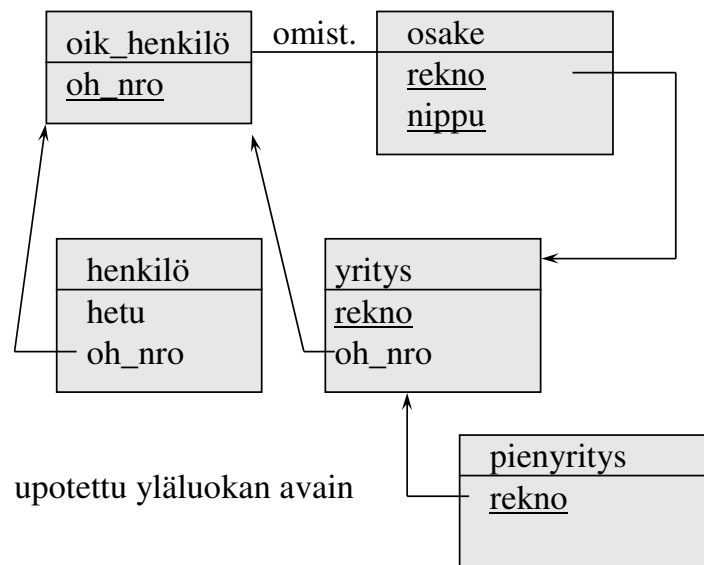
Yleistyshierarkia



28

Yleistyshierarkia

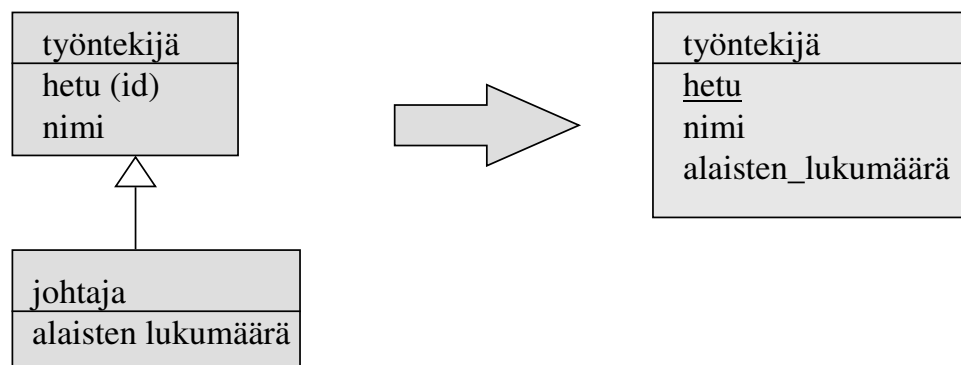
käytetty keinotekoisista avainta



29

Yleistyshierarkia

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



30

Tietokantasuunnittelusta

- Loogisen rakenteen suunnittelun tavoite on sijoittaa yhteenkuuluvat tiedot samaan tauluun
- 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 $f: v(A) \rightarrow v(B)$ 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 $A \rightarrow B$. Attribuuttia A kutsutaan määrääjäksi.
- Yksittäisen attribuutin A tilalla voi olla myös attribuuttiyhdistelmä. Tavoitteena on kuitenkin löytää yhdistelmät, joissa on minimaalinen määrä attribuutteja,
- Jos $A \rightarrow B$, voidaan määrääjään lisätä mikä tahansa attribuutti x ja pätee $Ax \rightarrow B$.
- Avain määrää funktionaalisesti kaikki relaatiokaavan attribuutit
- Riippuvuudet ilmaisevat jonkin kuvattavaan todellisuuteen liittyvän säännön

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
Henkilötunnuksen perusteella on selvítettävissä opiskelijan nimi
- **Kurssikoodi→KurssiNimi**
Yhtä kurssikoodia kohti on vain yksi kurssinimi
- **Hetu, Kurssikoodi --> TehtäväLkm**
 - Yhtä henkilötunnus – kurssikoodi yhdistelmää kohti on vain yksi tehtävälukumäärä = Henkilön tekemien tehtävien lukumäärä tehtävämäärä on kurssikohtainen
 - 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

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

Jako Boyce-Codd normaalimuotoon

- Miten järjestellään:
 - Karsi riippuvuusjoukko minimaaliseksi – poista johdettavissa olevat riippuvuudet (materiaalissa sääntöjä)
 - 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,
 - tavarankoodi,
 - tavaranimi,
 - 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)
- lomakenumero → tilauspäivä (lomakkeeseen liittyy yksikäsitteinen tilauspäivä)

Tietokantasuunnittelusta

- tavarankoodi → tavaranimi (tavarankoodi on tavarantunniste, jonka kautta päästään kaikkiin tavarantietoihin)
- lomakenumero, rivinumero → tavarankoodi (lomakkeen rivillä voi ilmoittaa yhden tilattavan tavarankoodin)
- lomakenumero, rivinumero → tilattumäärä (lomakkeen rivillä voi ilmoittaa yhden tilattavan tavarantilattamäärän)

Tietokantasuunnittelusta

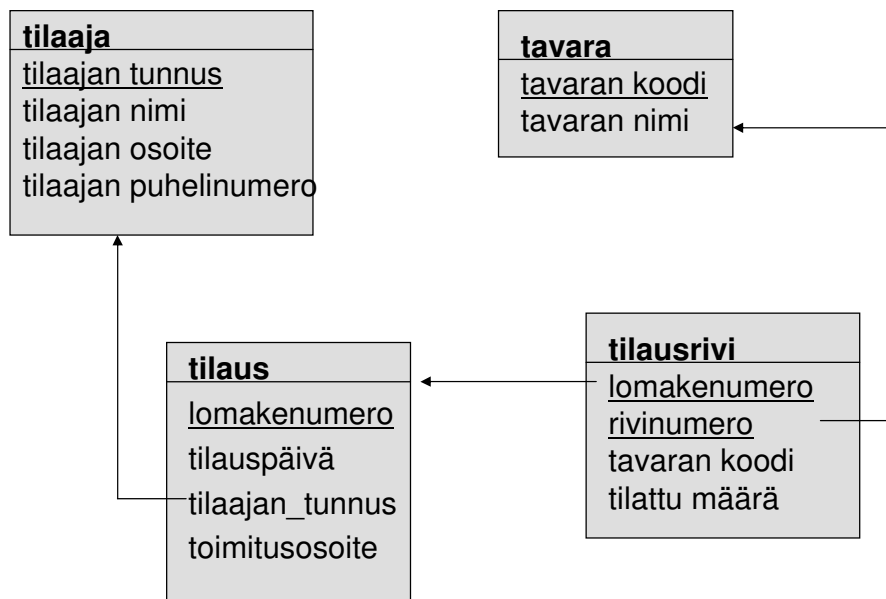
- Jako relaatioihin yhteisen määrääjän perusteella (jos riippuvuuksissa sama määrääjä niin niissä esiintyvät attribuutit samaan relaatioon)
- X(tilaajan tunnus, tilaajan nimi, tilaajan osoite, tilaajan puhelinnumero)
- Y(tavarankoodi, tavaranimi)
- Z(lomakenumero, tilauspäivä, tilaajan_tunnus, toimitusosoite)
- T(lomakenumero, rivinumero, tavarankoodi, tilattumäärä)

Tietokantasuunnittelusta

- Jos kaavioille löytyy kuvaava nimi, jako on onnistunut:
 - tilaaja(tilaajan tunnus, tilaajan nimi, tilaajan osoite, tilaajan puhelinnumero)
 - tavara(tavaran koodi, tavaran nimi)
 - tilaus(lomakenumero, tilauspäivä, tilaajan_tunnus, toimitusosoite)
 - tilausrivi(lomakenumero, rivinumero, tavaran koodi, tilattu määrä)

43

Tietokantasuunnittelusta



44