

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 käyttydy 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 t:n kohdalla

Jos Hallinto muuttua 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 periyymistä
 - 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

Yhteydet piilotettu

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

Kytkeä 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

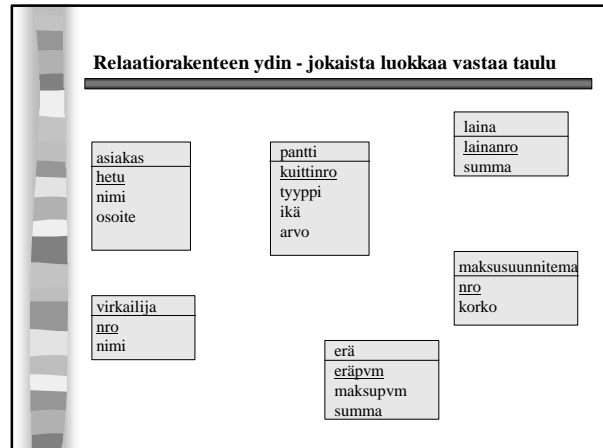
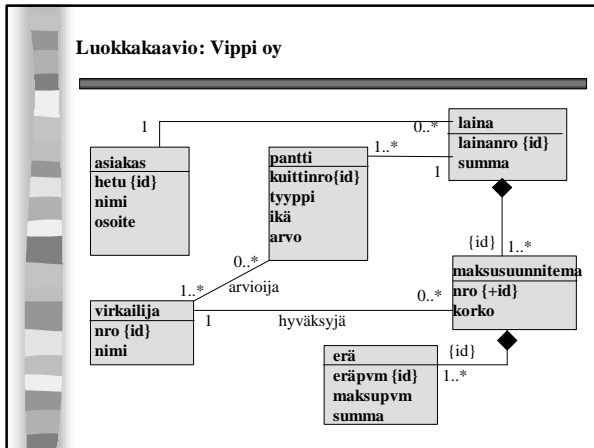
- tulisi selvittää mitkä tiedot ovat perustietoja ja mitkä voidaan päätellä (laskea) muiden tietojen perusteella
- attribuutin arvo tai yhteyden olemassaolo voi olla pääteltävissä

Ei esitetä johdettavissa olevaa tietoa

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

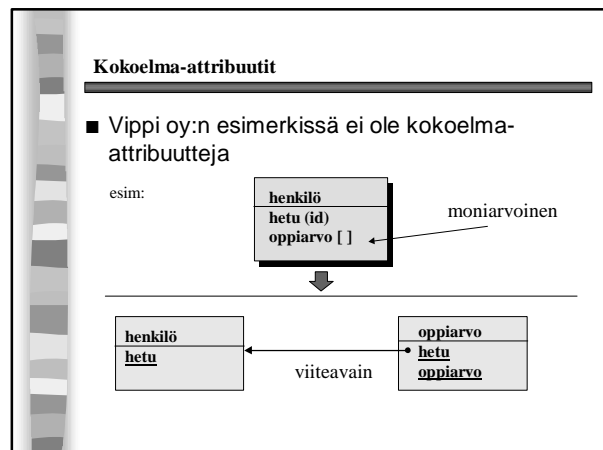
Muunnoksen perussäännöt:

- 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



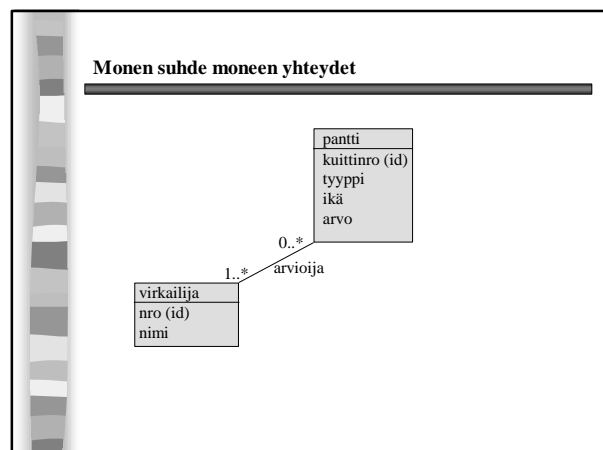
Kokoelma-attribuutit

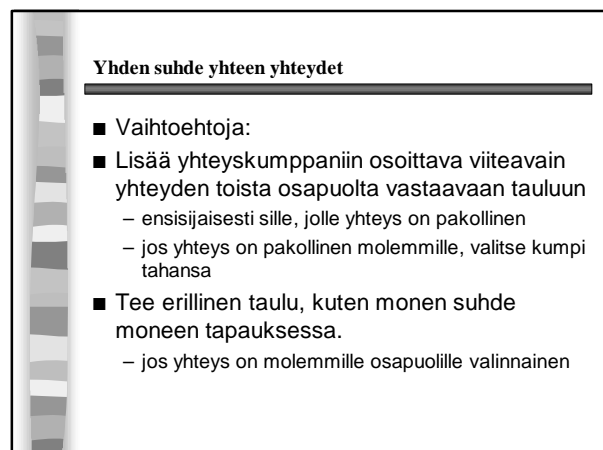
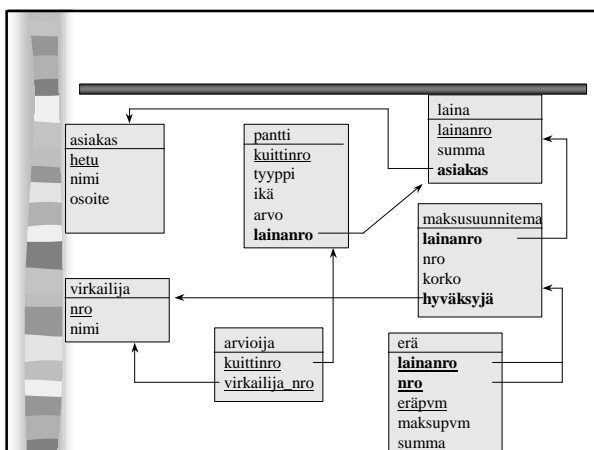
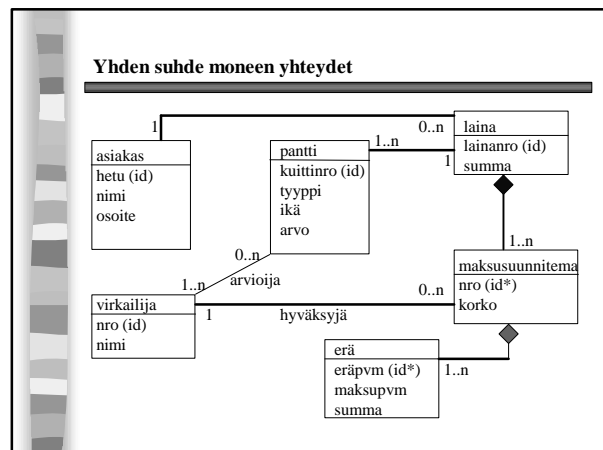
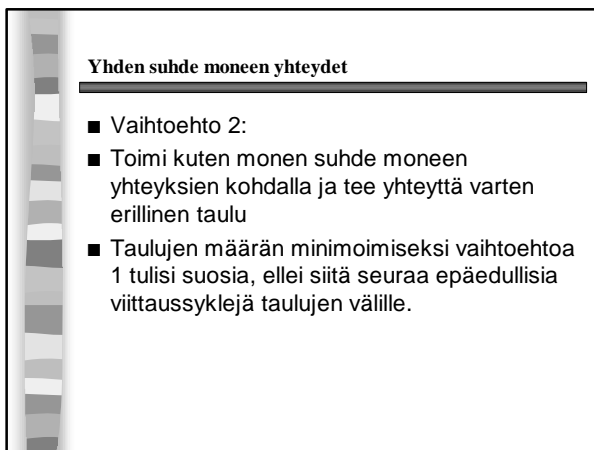
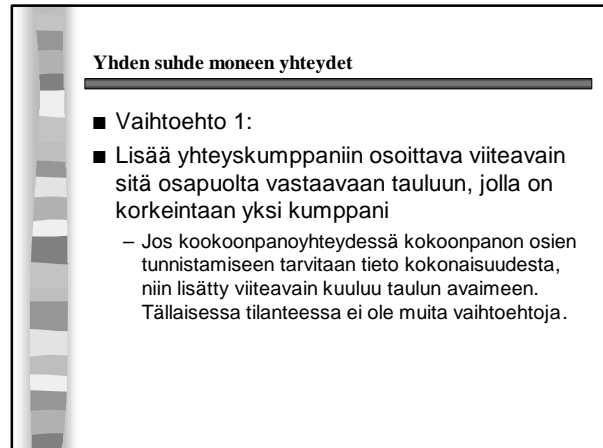
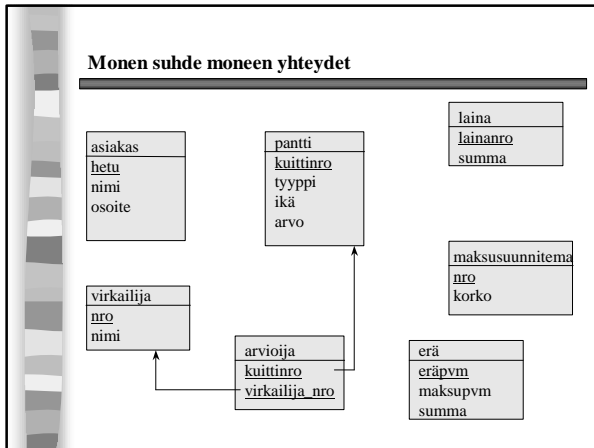
- Kutakin kokoelma-attribuuttia (moniarvoista) vastaa taulu, jonka sarakkeina ovat
 - viiteavain kokoelmatyyppisen attribuutin sisältävää luokkaa vastaavaan tauluun,
 - sarakke attribuutin arvoa varten.
- Taulun kaikki sarakkeet kuuluvat avaimen

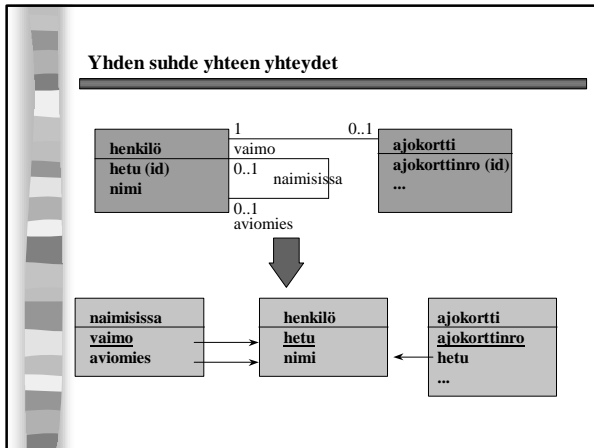


Monen suhde moneen yhteydet

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

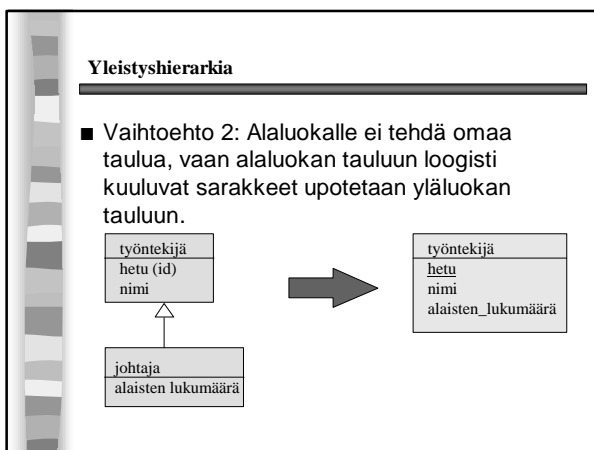
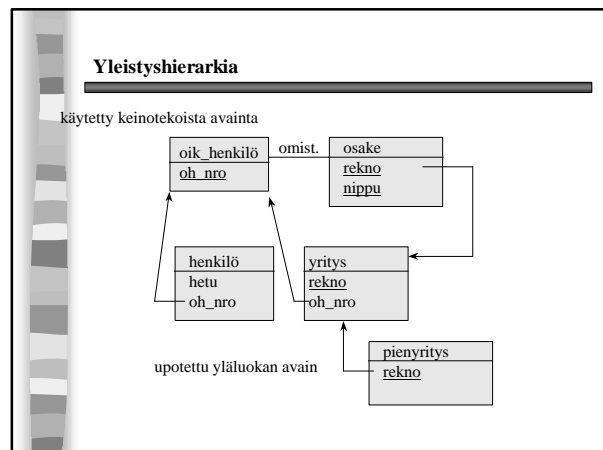
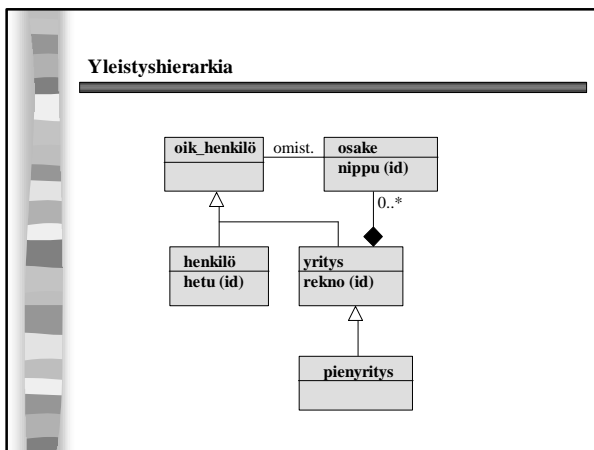






Yleistyshierarkia

- 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.



Tietokantasuunnittelusta

- Relatiomalliin 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 jokainen A:n arvo kuvautuu yhdelle B:n arvolla 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 selvitetävissä kun tiedetään attribuutin A arvo. Selvittäminen voisi tapahtua kyselyllä
 - `select distinct B from R where A=a;`

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 $Ax \rightarrow B$.

Tietokantasuunnittelusta

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

Tietokantasuunnittelusta

- **Hetu \rightarrow OpiskelijaNimi**
`select distinct OpiskelijaNimi from Kurssilainen where Hetu=a;`
(max 1 rivi)
- **Kurssikoodi \rightarrow KurssiNimi**
`select distinct KurssiNimi from Kurssilainen where Kurssikoodi= a;`
(max 1 rivi)
- **Hetu, Kurssikoodi \rightarrow TehtäväLkm**
`select distinct TehtavaLkm from Kurssilainen where Hetu=a;`
(monta riviä)
`select distinct TehtavaLkm from Kurssilainen where Kurssikoodi=b;`
(monta riviä)
`select distinct TehtavaLkm from Kurssilainen where Kurssikoodi=b and Hetu=a;`
(max 1 rivi)

Tietokantasuunnittelusta

- 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
- \Rightarrow Kaikki attribuutit eiväy kuulu yhteen koska
 - Hetu \rightarrow OpiskelijaNimi ja
 - Kurssikoodi \rightarrow KurssiNimi rikkovat sääntöä

Tietokantasuunnittelusta

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

Tietokantasuunnittelusta

- **Tilauslomaketta** analysoidessa löydettiin seuraavat attribuutit:
 - lomakenumero,
 - tilaajan tunnus,
 - tilaajan nimi,
 - tilaajan osoite,
 - tilaajan puhelinnumero,
 - toimitusosoite,
 - rivinumero,
 - tavarankoodi,
 - tavaranimi,
 - tilattumäärä, 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

- 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 tavarant määrän)
- lomakenumero → tilauspäivä

Tietokantasuunnittelusta

- **Jako** relaatioihin yhteisen määrääjän perusteella:
 - X(tilaajan tunnus, tilaajan nimi, tilaajan osoite, tilaajan puhelinnumero)
 - Y(tavarankoodi, tavaranimi)
 - Z(lomakenumero, tilauspäivä)
 - T(lomakenumero, rivinumero, tavarankoodi, tilattumäärä)

Tietokantasuunnittelusta

- Jos kaavioille löytyy kuvaava nimi jako on onnistunut:
 - tilaaja(tilaajan tunnus, tilaajan nimi, tilaajan osoite, tilaajan puhelinnumero)
 - tavara(tavarankoodi, tavaranimi)
 - tilaus(lomakenumero, tilauspäivä)
 - tilausrivi(lomakenumero, rivinumero, tavarankoodi, tilattumäärä)