CS-A1150 Tietokannat

5.3.2019

Oppimistavoitteet: tämän luennon jälkeen

- Osaat tehdä tietokantaa kuvaavan mallin UML-kaavion avulla. Tunnet esimerkiksi seuraavat UML-mallinnuksen keskeiset asiat:
  - luokka ja attribuutti
  - assosiaatio
  - assosiaatioluokka
  - aliluokka
  - kompositio ja aggregaatio.

UML-mallinnus

- *Ongelma:* mitä relaatioita luotavaan tietokantaan pitäisi määritellä?
- Usein on helpompi aloittaa suunnittelu korkeammalla tasolla kuin suoraan relaatiomallilla.
- Tässä esitetään yksi tapa esittää korkeamman tason suunnitelmia, UML-kaaviot (UML diagrams, UML = Unified Modeling Language).
- UML on alunperin kehitetty olio-ohjelmien suunnitteluun. Tietokantojen suunnittelussa UML:stä käytetään vain pientä osaa sen tarjoamista ominaisuuksista ja mahdollisuuksista.
- Muita yleisesti käytettyjä tapoja suunnitelmien esittelyyn:
  - ER-kaaviot (entity-relationship diagrams, E/R diagrams)
  - ODL (Object Description Language)

Eroja olio-ohjelmoinnin UML-suunnitteluun

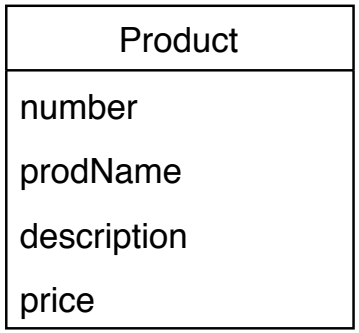
- (Jos et tiedä ennestään mitään UML:stä, voit hypätä tämän kalvon yli.)
- Toisin kuin olio-ohjelmia suunniteltaessa, tietokantoja suunnitellessa
  - UML-kaaviossa ei esitetä luokkien metodeita
  - UML-kaaviossa merkitään luokan avainattribuutit
  - UML-kaaviossa luokan sisälle ei merkitä niitä attribuutteja, joita vastaava tieto saadaan assosiaation avulla
  - luokan sisälle merkittyjen attribuuttien arvojen tulee olla atomisia
  - assosiaation valitsevuus (multiplisiteetti) on pakko merkitä näkyviin, jos se ei ole `1..1`
  - assosiaatioita kuvaavien viivojen päihin ei usein merkitä nuolia.

Luokka

- UML-kaavion luokka (class) vastaa suunnilleen olio-ohjelmointikielen luokkaa ilman luokassa määriteltyjä metodeita.
- Attribuutit kuvaavat luokan olioiden ominaisuuksia.

Esimerkki luokasta

- Luokka *Product* kuvaa verkkokaupan tuotteita.



- Jokaista verkkokaupassa tarjolla olevaa tuotetta vastaa yksi *Product*-olio.
- Luokan olion attribuutit on listattu luokan nimen alla. Attribuuttien arvojen tulee olla atomisia. Jokaisella *Product*-oliolla on omat arvot attribuuteille *number*, *prodName*, *description* ja *price*.
- Tietokantoja mallintaessa UML-kaaviossa ei lainkaan kuvata luokan olioille mahdollisia toimenpiteitä (metodeita).

Assosiaatiot

- Assosiaatiot (associations) yhdistävät kahta luokkaa.
- Esimerkiksi luokkien *Product* ja *Manufacturer* välille voidaan määritellä assosiaatio *Made-by*. Jos valmistaja *m* on valmistanut tuotteen *p*, *Product*-luokan olio *p* liittyy *Manufacturer*-luokan olioon *m* assosiaatiolla *Made-by*.
- Assosiaation nimi kirjoitetaan yleensä sitä kuvaavan viivan alapuolelle.

- Assosiaation valitsevuus (multiplisiteetti) merkitään assosiaatiota kuvavan viivan ylä- tai alapuolelle viivan päähän (tarkemmin seuraavalla kalvolla).

Assosiaation valitsevuus

- Assosiaatiota kuvaavan viivan ylä- tai alapuolelle viivan päähän merkitään, kuinka monta yhden luokan oliota voi liittyä yhteen toisen luokan olioon assosiaation kautta.

- Esimerkissä yhteen *Product*-olioon voi liittyä 0 tai 1 *Manufacturer*-oliota.
- Yhteen *Manufacturer*-olioon voi liittyä mielivaltainen määrä *Product*-olioita.
- **Huomaa, kumpaan päähän merkinnät tulevat!**

Assosiaation valitsevuus, merkinnät tarkemmin

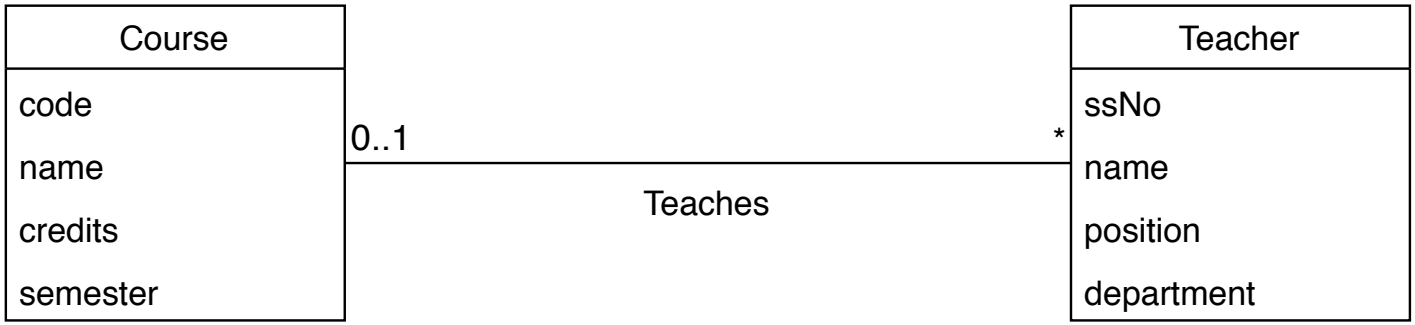
- Luokkien *A* ja *B* välistä assosiaatiota kuvaavan viivan ylä- tai alapuolella **luokan *B* puoleisessa päässä** oleva merkintä `m..n` tarkoittaa, että jokaiseen *A*-luokan olioon liittyy vähintään *m* ja enintään *n* kappaletta *B*-luokan olioita.
- Esimerkkejä:
  - `0..1` korkeintaan 1, mutta ei välttämättä yhtään
  - `0..5` korkeintaan 5, mutta ei välttämättä yhtään
  - `1..2` vähintään 1, korkeintaan 2
  - `1..1` tämälleen 1 (voidaan merkitä myös `1`)
  - `1..*` vähintään 1, muuten mielivaltainen määrä
  - `0..*` mielivaltainen määrä (voidaan esittää myös pelkällä `*`:llä).
- Jos merkintä puuttuu, se on tietokantojen mallinnuksessa sama kuin `1..1`.

Assosiaation valitsevuus, termejä

- *Monesta moneen*-assosiaatio (many-many): yhteen ensimmäisen luokan olioon voi liittyä mielivaltainen määrä toisen luokan olioita ja yhteen toisen luokan olioon voi liittyä mielivaltainen määrä ensimmäisen luokan olioita assosiaation kautta.
- *Monesta yhteen*-assosiaatio (many-one): yhteen ensimmäisen luokan olioon voi liittyä korkeintaan yksi toisen luokan olio.
- *Monesta täsmälleen yhteen*-assosiaatio (many-exactly one): yhteen ensimmäisen luokan olioon liittyy täsmälleen yksi toisen luokan olio.

Välitehtävä 1

- Mikä seuraavista välitteistä pätee alla olevaan UML-kaavioon, jossa kuvataan opettajia ja kursseja?



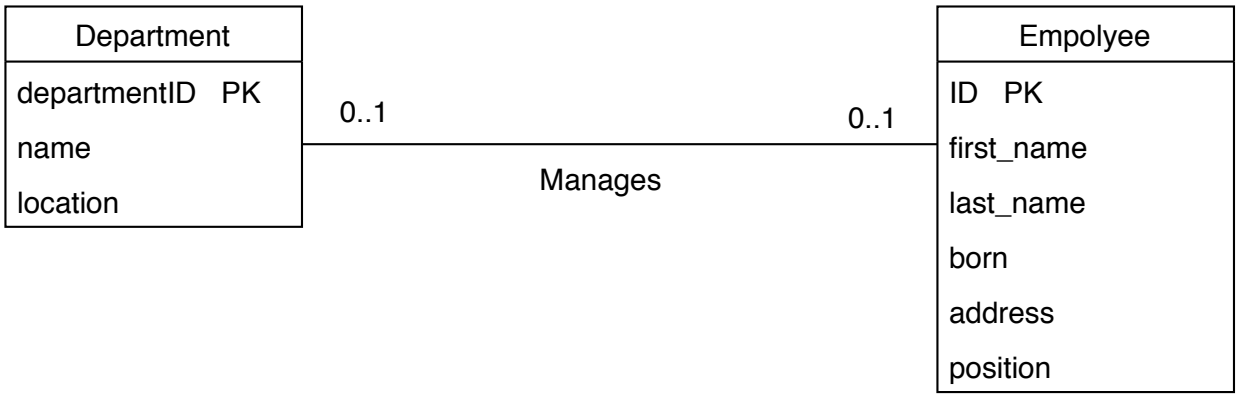
a Yksi opettaja voi opettaa mielivaltaista määrää kursseja, ja yhdellä kurssilla voi olla mielivaltainen määrä opettajia.
b Yksi opettaja voi opettaa korkeintaan yhtä kurssia, mutta yhdellä kurssilla voi olla mielivaltainen määrä opettajia.
c Yksi opettaja voi opettaa mielivaltaista määrää kursseja, mutta yhdellä kurssilla voi olla korkeintaan yksi opettaja.
d Yksi opettaja voi opettaa korkeintaan yhtä kurssia, ja yhdellä kurssilla voi olla korkeintaan yksi opettaja.

Vastaus välitehtävään

- Kaaviossa assosiaation luokan *Teacher* puoleisessa päässä on tähti. Tämä tarkoittaa sitä, että yhteen *Course*-olioon voi liittyä mielivaltainen määrä *Teacher*-olioita. Toisin sanoen yhdellä kurssilla voi olla mielivaltainen määrä opettajia.
- Assosiaation luokan *Course* puoleisessa päässä on merkintä `0..1`. Näin olleen yhteen *Teacher*-olioon voi liittyä 0 tai 1 *Course*-oliota. Yhdellä opettajalla voi siis olla korkeintaan yksi kurssi.
- Vaihtoehto b on siis oikein.

Assosiaation valitsevuus, lisää termejä

- Jos assosiaatio on monesta yhteen kumppankin suuntaan, on se *yhdestä yhteen* -assosiaatio.
- Esimerkki: yrityksessä jokaisellla osastolla on korkeintaan yksi johtaja ja kukin työntekijä voi johtaa korkeintaan yhtä osastoa.

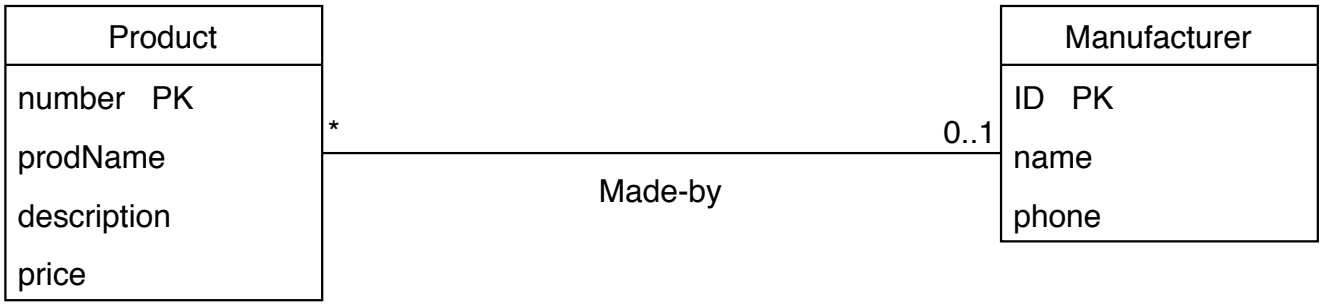


- Yhdestä yhteen -assosiaatio on monesta yhteen -assosiaation erikoistapaus.

Avaimet

- Olio-ohjelmia suunnitellessa avaimia ei tarvita, koska olio-ohjelmissa ajatellaan jokaisella oliolla olevan yksilöllinen identiteetti. Tietokannoissa avaimet ovat kuitenkin olennaisia, jotta tietokannan taulun eri rivit (relaation monikot) voitaisiin yksilöidä. Siksi tietokantoja suunnitellessa avainattribuutit merkitään UML-kaavioon.
- Luokan E avain on attribuutti tai attribuuttijoukko siten, että millään kahdella E:n oliolla ei ole samaa avainattribuutin arvoa tai avainattribuuttijoukon arvoyhdistelmää.
- Avainattribuutti merkitään merkinnällä PK.
- Avainattribuuttien valintaan voi olla useita eri vaihtoehtoja, mutta vain yksi niistä merkitään avaimeksi. Jos saman luokan useampi attribuutti on merkitty PK-merkinnällä, muodostaa tämä attribuuttijoukko yhdessä avaimen.

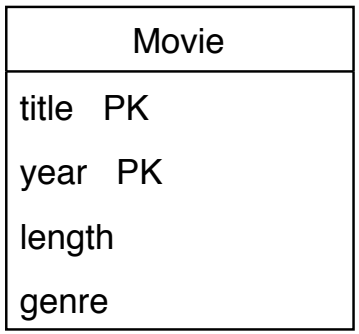
Esimerkki avainattribuuttien merkitsemisestä



- Tuote tunnistetaan sen tuotenumeron (attribuutti *number*) ja valmistaja sen tunnuksen (attribuutti *ID*) perusteella.

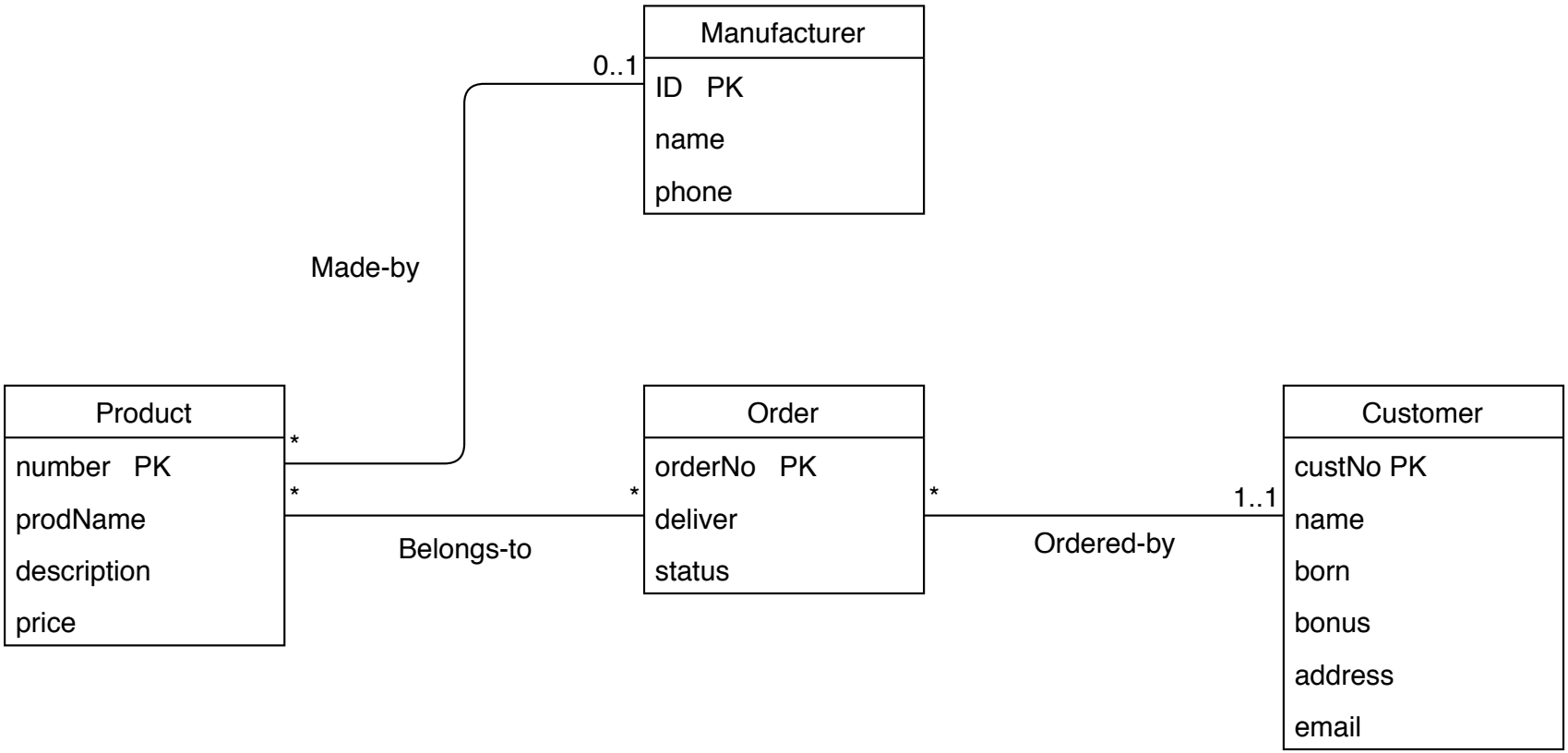
Toinen esimerkki avaimista

- Oppikirjan esimerkissä luokan *Movie* avaimina toimivat elokuvan nimi ja vuosiluku yhdessä. (Ajatuksena on, että mikään elokuvastudio ei julkaise samana vuonna samannimistä elokuvaa kuin jollain kilpailevalla studiolla on. Sen sijaan elokuvasta voidaan myöhempänä vuonna tehdä uusi versio samalla nimellä, esim. Tuntematon sotilas.)



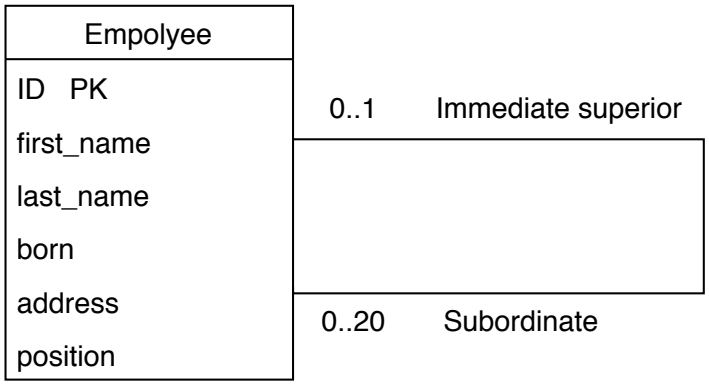
Esimerkkikaavio

- Verkkokauppaesimerkkiin on lisätty yksilöjoukot *Orders* ja *Customers* sekä assosiaatiot *Ordered-by* ja *Belongs-to*.
- Malli sallii tietokantaan lisättävän tuotteita, joiden valmistajan tietoja ei ole tietokannassa. Sen sijaan tietokantaan ei voi lisätä tilauksia, joilla ei ole asiakasta.



Itseassosiaatio

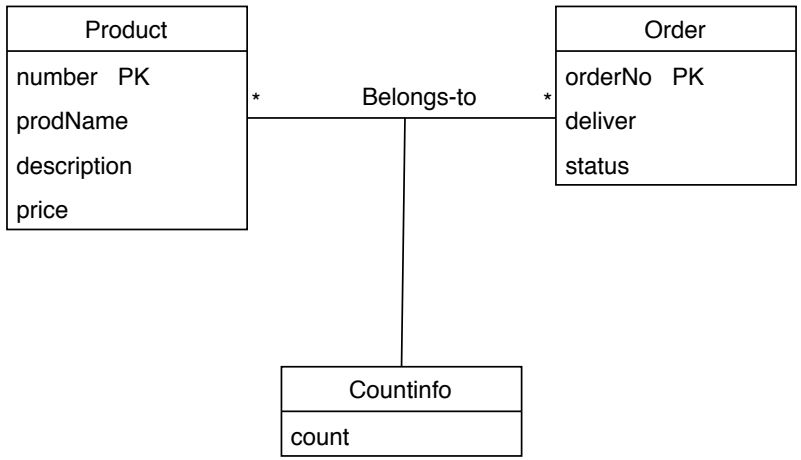
- *Ongelma:* Miten kuvataan UML-kaaviossa assosiaatioita, joissa sama luokka esiintyy kahteen kertaan?
- Esimerkki: määritellään, että joku työntekijä on toisen työntekijän lähiesimies.



- Assosiaatioon kirjoitetaan molemmat roolit. Assosiaation valitsevuus kirjoitetaan oikean roolin päähän (tässä työntekijällä voi olla korkeintaan yksi lähiesimies ja 0–20 alaista).

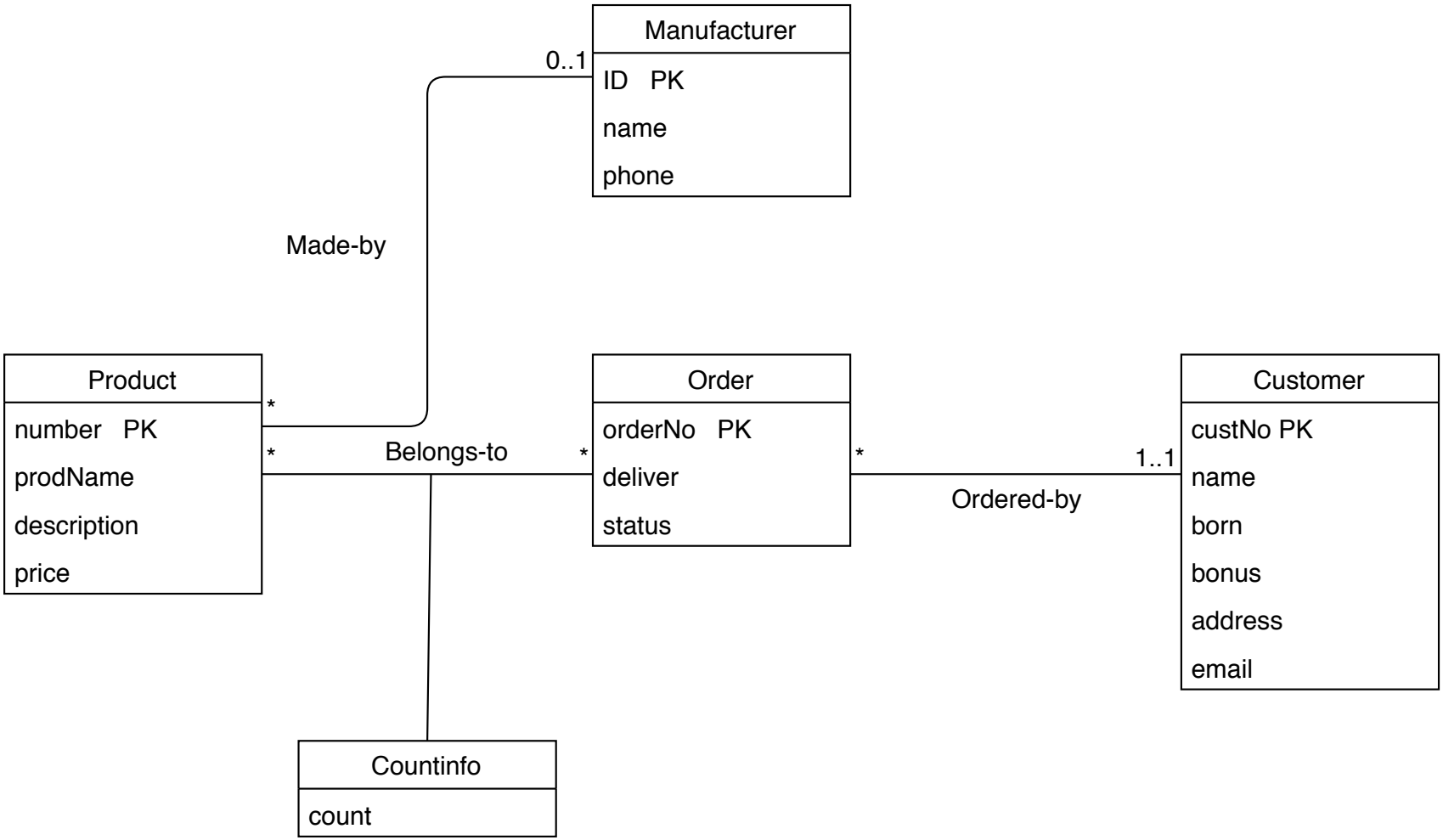
Assosiaatioluokka

- Halutaan, että verkkokauppaesimerkissä asiakas voi lisätä yhteen tilaukseen monta kappaletta samaa tuotetta.
- Mihin tieto kappalemäärästä pitäisi liittää?
- Tehdään assosiaatioon *Belongs-to* asssosiaatioluokka (association class), jonka attribuutiksi lisätään tieto lukumäärästä.



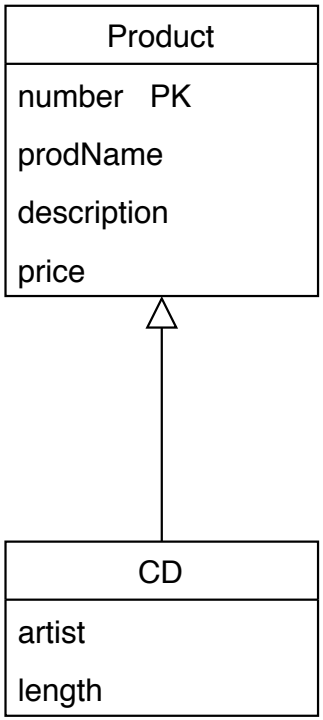
- Assosiaatioluokalla ei ole koskaan avainattribuutteja.

Verkkokauppaesimerkki assosiaatioluokan kanssa



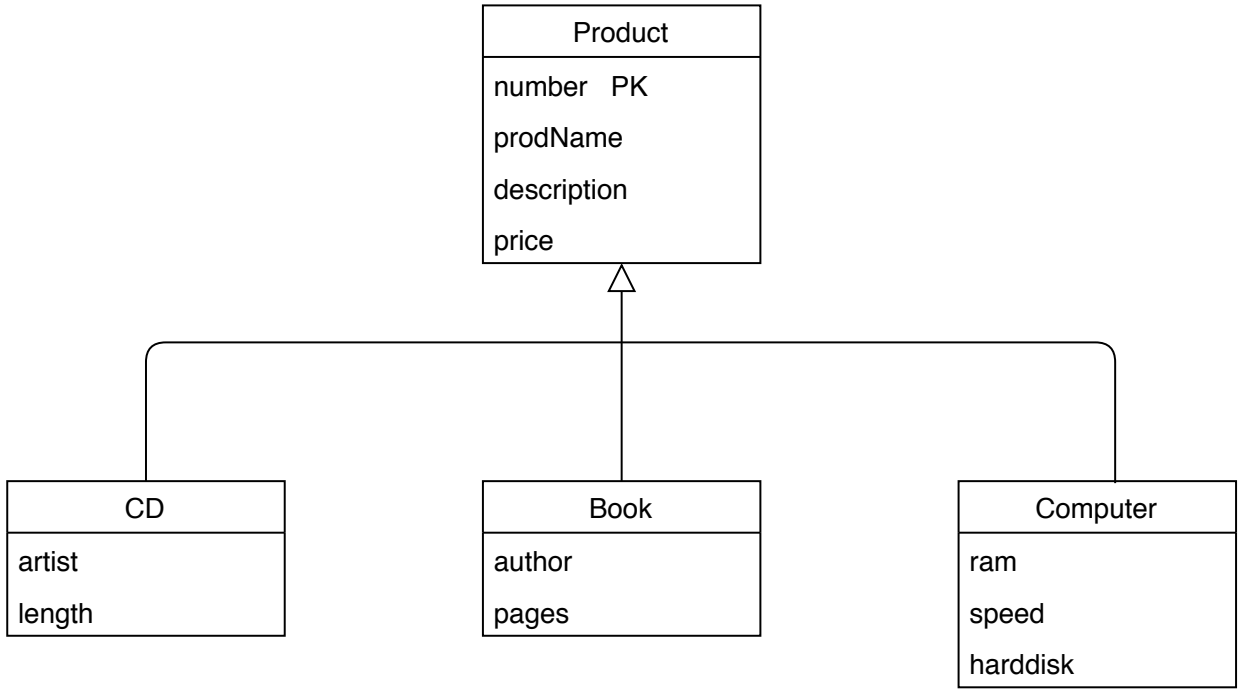
Perintä ja aliluokat

- Joillakin luokan olioilla on ominaisuuksia, joita ei ole kaikilla luokan olioilla. Näitä olioita varten voidaan määritellä *aliluokka* (subclass).
- Aliluokan olioilla on kaikki sen yliluokan (superclass) atttribuutit ja assosiaatiot sekä lisäksi aliluokan omat attribuutit ja assosiaatiot
- Suhteeseen aliluokasta yliluokkaan merkitään pieni kolmio (kärki yläluokkaan päin).



Aliluokat UML-kaaviossa, esimerkki

- Esimerkissä luokalle *Product* on määritelty aliluokkia erilaisia myytäviä tuotteita varten, tässä *CD*, *Book* ja *Computer*.

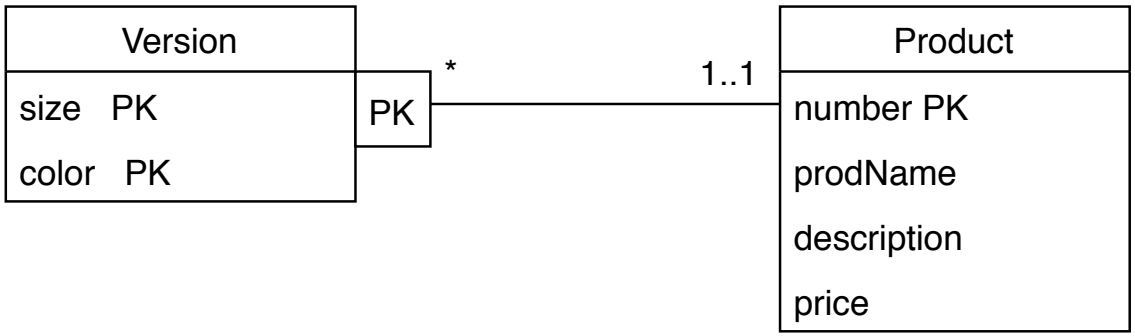


- Aliluokilla voi olla omia assosiaatioita, joita ei ole yliluokalla tai muilla perintähierarkian luokilla. Esim. luokalla *CD* voisi olla assosiaatio luokkaan *Track*, joka kuvaa yhtä levyllä olevaa kappaletta.

Kun attribuutit eivät riitä avaimeksi

- Oletetaan, että verkkokaupan tuotteesta voi olla erilaisia versioita, esimerkiksi samaa vaatemallia erivärisenä ja -kokoisena.
- Määritellään luokka *Version* kuvaamaan tuotteen eri versioita.
- Luokan attribuutteina on version väri ja koko. Ne eivät kuitenkaan kelpaa edes yhdessä yksilöimään *Version*-oliota, siis avaimeksi.
- Ratkaisu: muodostetaan *Version*-luokan avain käyttämällä sekä *Version*-luokan omia attribuutteja että *Product*-luokan avainattribuuttia *number*.
- *Version*-luokkaan merkitään tällöin vain sen omat attribuutit.

Kun attribuutit eivät riitä avaimeksi, esimerkki



Aggregaatio

- Aggregaation (aggregation) avulla kerrotaan, että jonkin luokan olio koostuu toisen luokan olioista, esimerkiksi verkkokaupassa on tuoteryhmiä, jotka koostuvat tuotteista.
- Aggregaatiota merkitään avoimella vinoneliöllä sen luokan päässä, jonka olio koostuu toisen luokan olioista.

Kompositio

- Kompostio (composition) on kuin aggregaatio, mutta tiukempi vaatimus. Siinä vaaditaan, että luokan olion on pakko kuulua johonkin mustalla vinoneliöllä merkityn luokan olioon.

UML-kaavion laatiminen

- Miten voi lähteä laatimaan UML-kaaviota?
- Yksinkertainen, mutta usein toimiva menetelmä:
  1. Kirjoita kuvaus mallinnettavasta tietokannasta.
  2. Alleviivaa kuvauksesta kaikki substantiivit.
  3. Etsi ehdokkaita luokiksi ja attribuuteiksi substantiivien joukosta.
  4. Kaikista substantiiveista ei tule kumpaakaan.
  5. Kun luokat ja attribuutit ovat valmiina, mieti, millaisia suhteita luokkien olioiden välillä on. Tee niistä assosiaatioita.

Esimerkki: verkkokauppa

- Suunnittele tietokanta verkkokaupalle, jossa on tuotteita ja asiakkaita. Asiakkaat tekevät tilauksia, joihin voi kuulua useita tuotteita. Tuotteista tiedetään tuotenumero, nimi, kuvaus, hinta ja valmistaja. Valmistajasta tallennetaan tunnus, nimi ja puhelinnumero. Asiakkaasta tiedetään asiakasnumero, nimi, syntymävuosi, bonuspisteet, osoite ja sähköpostiosoite. Jokaisella tilauksella on yksikäsitteinen tilausnumero. Tilauksesta tiedetään sen toimitustapa, tila, mitä tuotteita siihen kuuluu ja tilauksen tehnyt asiakas.

Esimerkki jatkuu: alleviivaa attribuutit

- Suunnittele tietokanta verkkokaupalle, jossa on tuotteita ja asiakkaita. Asiakkaat tekevät tilauksia, joihin voi kuulua useita tuotteita. Tuotteista tiedetään tuotenumero, nimi, kuvaus, hinta ja valmistaja. Valmistajasta tallennetaan tunnus, nimi ja puhelinnumero. Asiakkaasta tiedetään asiakasnumero, nimi, syntymävuosi, bonuspisteet, osoite ja sähköpostiosoite. Jokaisella tilauksella on yksikäsitteinen tilausnumero. Tilauksesta tiedetään sen toimitustapa, tila, mitä tuotteita siihen kuuluu ja tilauksen tehnyt asiakas.
- Yllä "tietokanta"on yleinen termi, joka ei suoraan liity mallinnettavaan kohteeseen. Siitä ei siis tehdä luokkaa tai attribuuttia. "Verkkokauppa"taas kuvaa sitä kokonaisuutta, jota ollaan mallintamassa (koko UML-kaavio), joten siitäkään ei tehdä luokkaa tai attribuuttia.

Suunnitteluperiaatteita

- Mitä pitäisi ottaa huomioon, kun lähdetään laatimaan UML-mallia jostain tosielämän asiakokonaisuudesta?
- Seuraavilla kalvoilla käydään läpi joitakin keskeisiä suunnitteluperiaatteita.

Tarkkuus

- Luokkien ja niiden attribuuttien tulee vastata kuvattavaa reaalimaailmaa.
- Esimerkiksi assosiaation valitsevuus pitää valita sen mukaan, voiko samaan olioon liittyä todellisuudessa yksi vai useampi toisen luokan olio.
- Verkkokaupassa samaan tilaukseen voi kuulua useita tuotteita, mutta tilauksella voi olla täsmälleen yksi asiakas. Tämä näkyy UML-kaaviossa.

Ylimäärän (redundancy) välttäminen

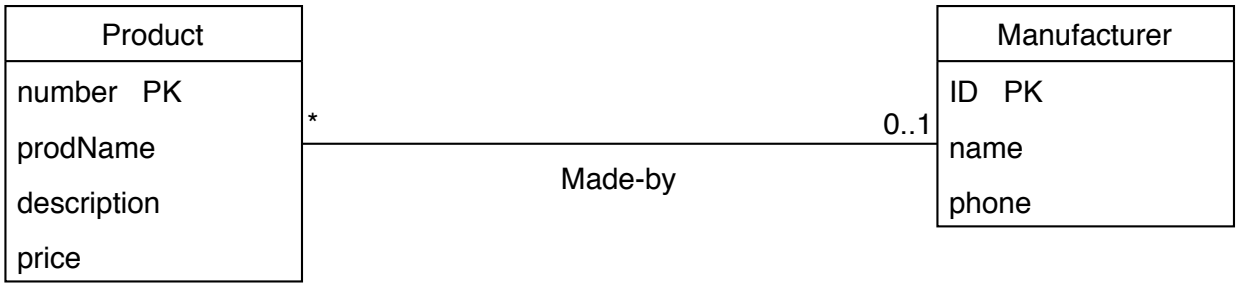
- Kukin asia pitäisi sanoa vain yhteen kertaan.
- Esimerkiksi luokalle *Product* ei kannata laittaa attribuuksi valmistajan tunnusta, koska assosiaatio *Made-by* jo ilmaisee tuotteen valmistajan.
- Miksi ylimäärä on haitallista?
  - Saman tiedon toistaminen vie turhaa tilaa.
  - Tiedon toistaminen aiheuttaa ongelmia päivitysten yhteydessä.
- Olio-ohjelmia suunnitellessa toimitaan toisella tavalla, koska UML-kaavioon halutaan saada suoraan näkyviin, mitä attribuutteja olio-ohjelman luokilla on. Tietokantoja suunnitellessa tilanne on eri, koska luokat eivät edusta suoraan tietokantaan tulevia relaatioita.

Yksikertaisuuteen pyrkiminen

- Malliin ei kannata ottaa mukaan ylimääräisiä elementtejä.
- Esimerkiksi luokkaa ei kannata jakaa kahdeksi luokaksi ja niiden väliseksi assosiaatioksi ilman hyvää syytä.

Oikeiden elementtien käyttö

- Koska jonkin asian kuvaamiseen on syytä käyttää mallissa luokkaa, koska attribuutteja?
- Tarkastellaan alla olevaa mallia. Voisiko luokan *Manufacturers* ja assosiaation *Made-by* korvata lisäämällä luokkaan *Products* sopivat attribuutit?



Oikeiden elementtien käyttö, jatkoa

- Luokalle *Products* voisi periaatteessa määritellä attribuutit *manufacturerID*, *manufacturerName* ja *phone*, jotka korvaisivat luokan *Manufacturers* ja assosiaation *Made-by*.
- Mutta silloin saman valmistajan nimi ja puhelinnumero toistuisivat useassa eri tuotteessa.
- *Yleisperiaate:* jos jostain reaalimaailman käsitteestä mallinnetaan muutakin kuin pelkkä nimi tai lukuarvo, tulisi se mallintaa luokkana eikä attribuuttina.