Tampereen yliopisto

Tietokantojen perusteet

ER-kaavio ja muunnos SQL-tietokannan kaavioksi:

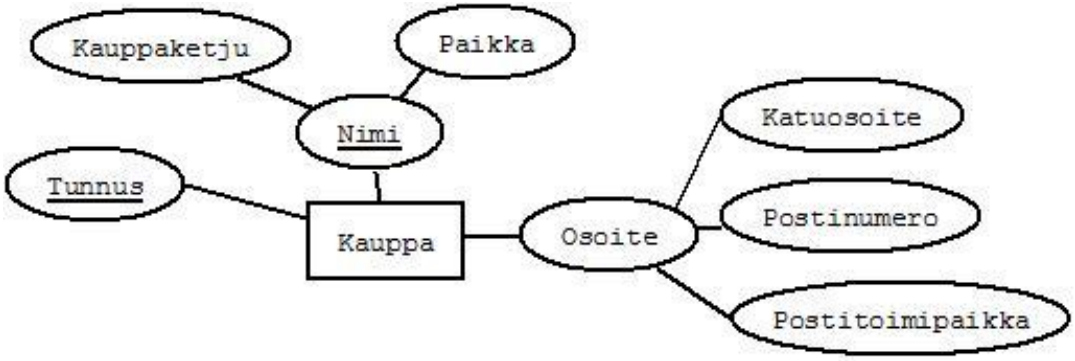
- Koottu attribuutti
- N-paikkaiset suhdetyypit
- Heikon entiteettityypin hyödyntämisestä mallintamisessa
- Avaimista
- Johdetuista attribuuteista
- Johdetuista suhdetyypeistä
- Suunnitteluperiaatteita

Kalvot perustuvat kirjaan Elmasri, Navathe: Fundamentals of Database Systems.

Koottu attribuutti

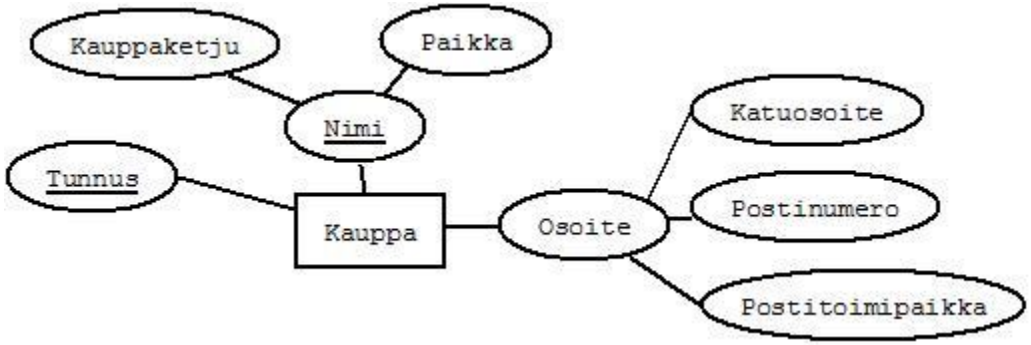
- Koottu (composite) attribuutti koostuu kahdesta tai useammasta yksinkertaisesta (simple, atomic) attribuutista.
  - Yksinkertaista attribuuttia käsitellään yhtenä kokonaisuutena, jota ei jaeta osiin.
  - Koottua attribuuttia voidaan käsitellä joko kokonaisuutena tai osina.

- Alla olevassa esimerkissä Kauppa-entiteettityypillä on kaksi koottua attribuuttia:
  - Nimi
    - Kauppaketju
    - Paikka
  - Osoite
    - Katuosoite
    - Postiosoite
    - Postitoimipaikka



ER_SQL: Koottu attribuutti

- Kun koottu attribuutti muunnetaan taulun sarakkeiksi, kootun attribuutin jokaiselle yksinkertaiselle attribuutille lisätään oma sarake.



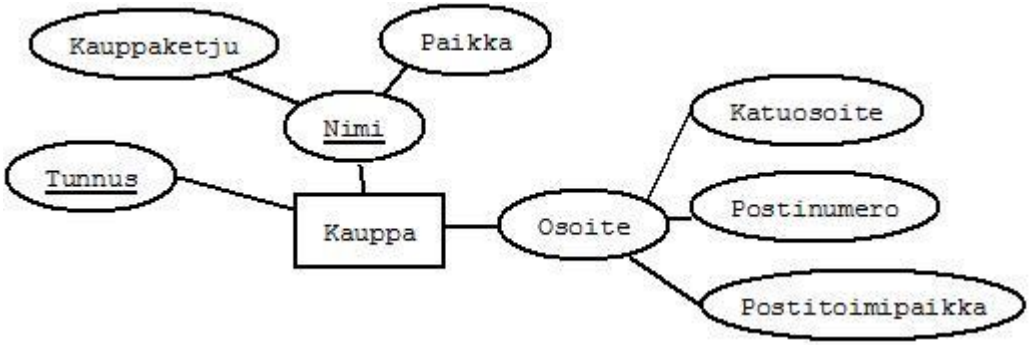
Kauppa-entiteettityypillä on kaksi tunnisteominaisuutta eli avainta. Näistä tunnus valitaan pääavaimeksi ja kauppaketju, paikka -yhdistelmästä tehdään avain.

kauppa
°tunnus PK
°kauppaketju U
°paikka U
°katuosoite
°postinumero
°postitoimipaikka

kauppa
°tunnus PK
°kauppaketju U1
°paikka U1
°katuosoite
°postinumero
°postitoimipaikka

Jos taulussa olisi useita avaimia, voitaisiin merkinnällä U#, missä # on numero, ilmaista eri avainten sarakkeet.

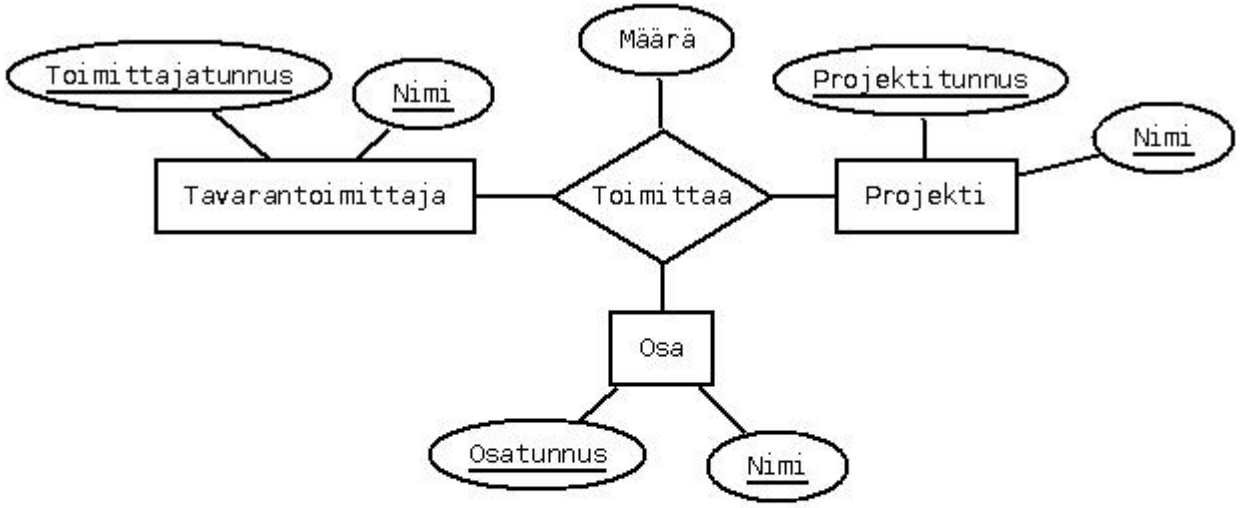
ER_SQL: Koottu attribuutti



kauppa

tunnus	kauppaketju	paikka	katuosoite	postinumero	postitoimipaikka
1	Ketju X	Kaleva	A-katu 22	33540	Tampere
2	Ketju Y	Kaleva	A-katu 22	33540	Tampere
3	Ketju Y	Lielahti	B-katu 1	33400	Tampere

N-paikkaiset suhdetyypit



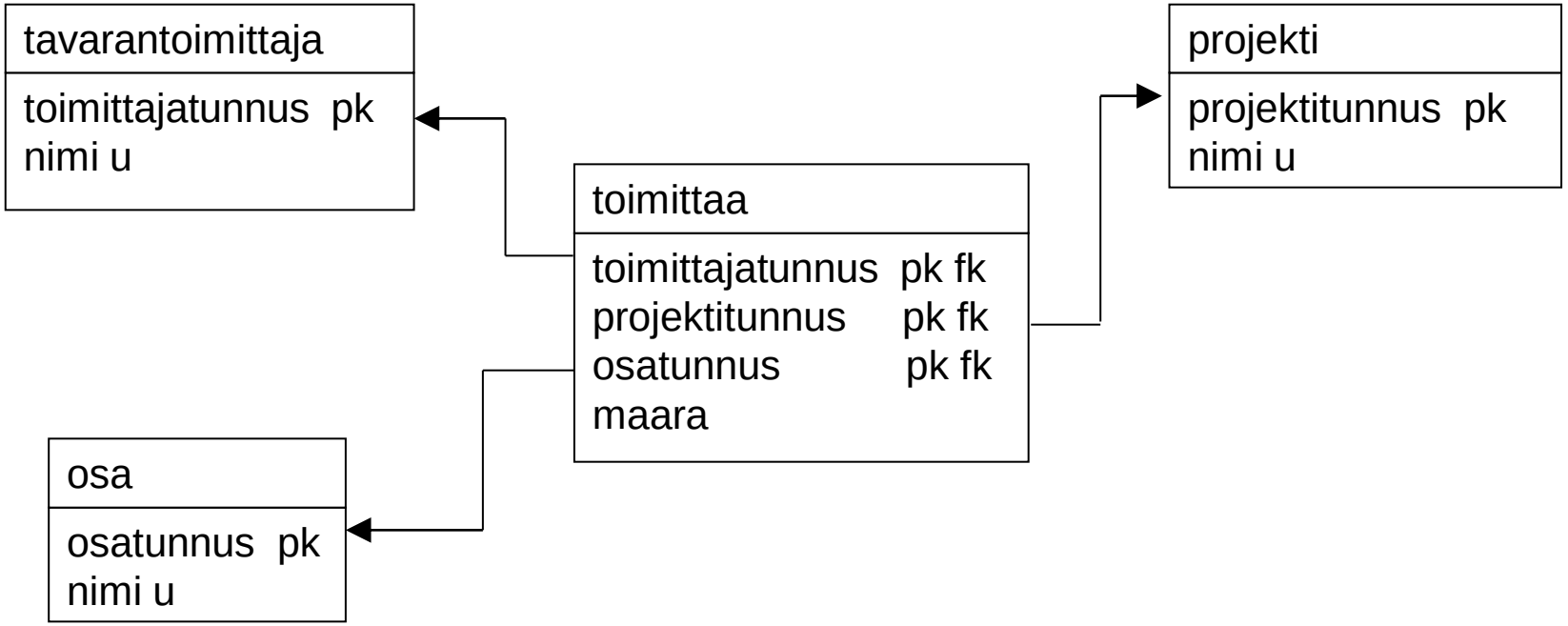
3-paikkainen (tertiäärinen) suhde: suhde yhdistää kolme entiteettiä toisiinsa.

Tietty tavarantoimittaja toimittaa tiettyä osaa tietylle projektille

ER_SQL: N-paikkaiset suhdetyypit

- Luodaan oma taulu suhdetyyppiä varten (“suhdetaulu”).
- Taulun nimeksi tulee suhdetyypin nimi.
- Tauluun sarakkeiksi
  - suhdetyyppiin liittyvien entiteettityyppien taulujen pääavainsarakkeet
    - joista tehdään viiteavaimet vastaaviin tauluihin
  - suhdetyypille mahdollisesti määritellyt attribuutit
- Suhdetaulun pääavaimeksi entiteettitauluilta saatujen sarakkeiden yhdistelmä.
  - PRIMARY KEY -määre

ER_SQL: N-paikkaiset suhdetyypit



ER → SQL: N-paikkaiset suhdetyypit

```
CREATE TABLE tavarantoimittaja(
toimittajatunnus INT,
nimi VARCHAR(30) NOT NULL,
PRIMARY KEY (toimittajatunnus),
UNIQUE (nimi));
```

```
CREATE TABLE projekti(
projektitunnus INT,
nimi VARCHAR(30) NOT NULL,
PRIMARY KEY (projektitunnus),
UNIQUE (nimi));
```

```
CREATE TABLE osa(
osatunnus INT,
nimi VARCHAR(30) NOT NULL,
PRIMARY KEY (osatunnus),
UNIQUE (nimi));
```

```
CREATE TABLE toimittaa(
toimittajatunnus INT,
projektitunnus INT,
osatunnus INT,
maara INT NOT NULL,
PRIMARY KEY (toimittajatunnus, projektitunnus, osatunnus),
FOREIGN KEY (toimittajatunnus) REFERENCES tavarantoimittaja(toimittajatunnus),
FOREIGN KEY (projektitunnus) REFERENCES projekti(projektitunnus),
FOREIGN KEY (osatunnus) REFERENCES osa(osatunnus));
```

ER → SQL: N-paikkaiset suhdetyypit

tavarantoimittaja

toimittajatunnus	nimi
1	Pulttipaja
2	Maken metalli
3	Terästakomo

osa

osatunnus	nimi
1	Pultti 1
2	Pultti 2
3	Pikkuruuvi
4	Jättiruuvi

projekti

projektitunnus	nimi
1	Projekti A
2	Projekti B

toimittaa

toimittajatunnus	projektitunnus	osatunnus	maara
1	1	1	500
2	1	1	2000
3	2	4	1000

Esimerkki: Miksi tarvitaan 3-asteinen suhdetyyppi?

- On rekkoja, tuotteita ja kauppoja sekä 3-asteinen suhdetyyppi toimittaa.
  - Esimerkiksi rekka 1 toimittaa tuotetta 1 kaupalle 1 (r1,t1,k1)

3-asteinen
(r1,t1,k1)
(r2,t1,k2)
(r2,t2,k1)

2-asteinen
(r1,t1) **(r2,t1)**
(r2,t2)

(r1,k1) **(r2,k1)**
(r2,k2)

(t1,k1) (t2,k1)
(t1,k2)

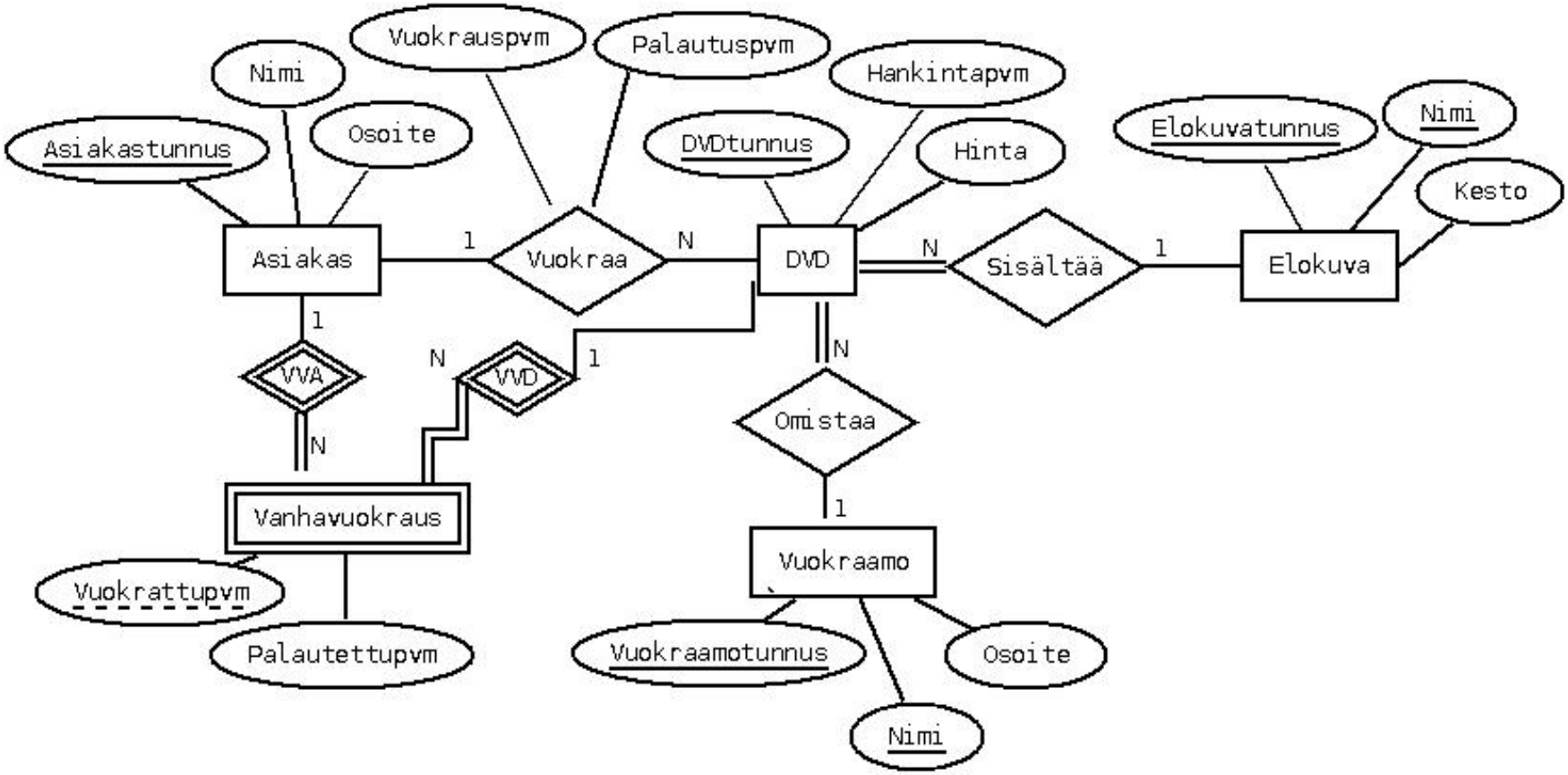
Binäärisuhteiden perusteella voitaisiin virheellisesti päätellä, että rekka 2 toimittaa tuotetta 1 kaupalle 1

(r2,t1), (r2,k1), (t1,k1)

Näin ei kuitenkaan ole – tarvitaan siis 3-asteinen suhde.

Heikon entiteettityypin hyödyntäminen mallintamisessa

- Suhdetyypeillä ei ole avainattribuutteja eikä suhdetyypin attribuutteja käytetä suhteiden tunnistamisessa.
  - Tietty suhde yksilöidään siihen osallistuvien entiteettien perusteella.
- Tietokantaan voidaan haluta tallentaa esim. ”historiatietoja”, esim.
  - kirjastokirjojen lainaus
  - kurssien suorittaminen (kurssiarvosanan korottaminen)
  - DVD-levyjen vuokraus
- Historiatietojen mallintamisessa voidaan käyttää heikkoa entiteettityyppiä:

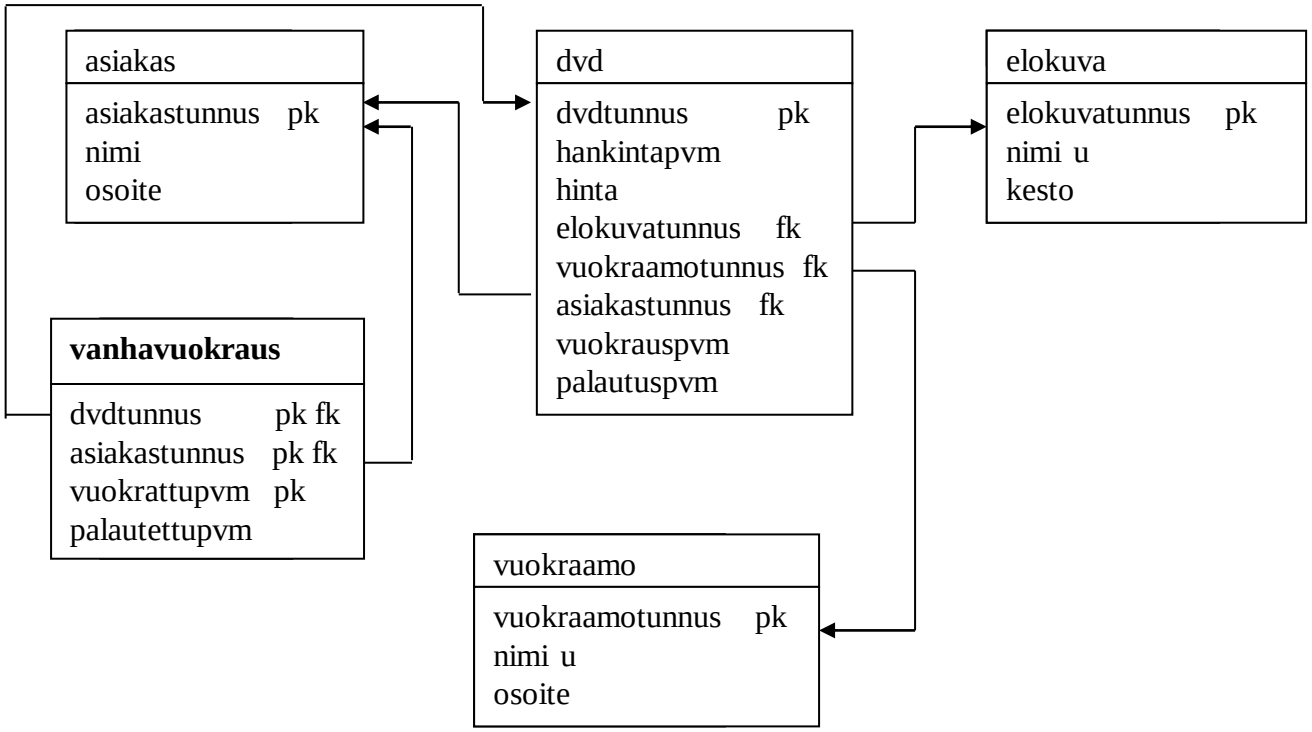


Mallinnetaan ”vanhat” vuokraamiset heikkona entiteettityyppinä, jonka osittaisena avaimena on vuokrauspäivämäärä.

Entiteettityypeille, suhdetyypeille ja attribuuteille pitäisi (yleensä) löytyä luontevat, kuvaavat nimet. Suhdetyypin voi tarvittaessa nimetä myös siihen osallistuvien entiteettityyppien nimien yhdistelmällä.

- yllä olevassa kaaviossa VVA- ja VVD-nimet lyhennetty entiteettityyppien nimien yhdistelmistä.

Heikon entiteettityypin hyödyntäminen mallintamisessa



Avaimet

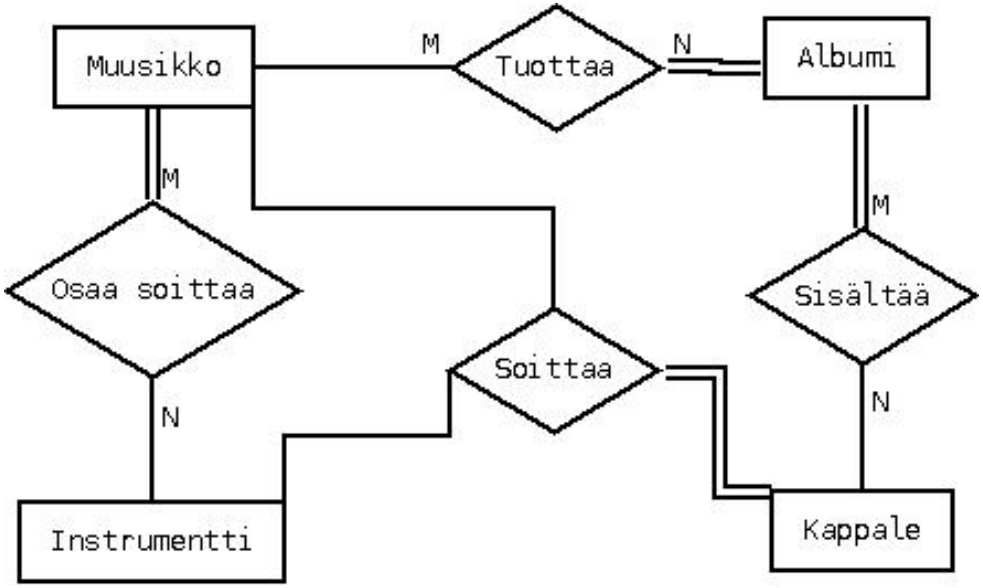
- Hyvä pääavain on attribuutti tai attribuuttiyhdistelmä, jonka arvo ei muutu (tai muuttuu hyvin harvoin).
  - Pääavaimen arvon muutos voi johtaa suureen määrään päivityksiä
- Joskus ”luonnollinen” attribuutti on arvoltaan muuttumaton (tai harvoin muuttuva), esim.
  - henkilötunnus
  - valmistus- tai sarjanumero (esim. autot, kännykät)
- Usein käytetään keinotekoisia pääavaimia. Monesti nämä ovat kokonaislukutyyppisiä.
  - Esimerkiksi työntekijänumero pääavain, henkilötunnus avain
- ER-kaavion on vastattava SQL-kannan kaaviota. Siksi ER-kaavioon laitetaan mukaan myös keinotekoiset avaimet.

Johdetut attribuutit

- Johdetut attribuutit jätetään yleensä pois tietokannasta, koska niiden arvot voidaan johtaa tietokantaan tallennettavien tietojen perusteella.
    - esim. osaston työntekijöiden lukumäärä
- Joskus johdettu attribuutti halutaan tietokannan tauluun mukaan tehokkuus- ja/tai käyttömukavuussyistä. (Ei tämän kurssin asioita)

Johdettavissa olevat suhdetyypit

- Johdettavissa olevat suhdetyypit jätetään (yleensä) ER-kaaviosta pois.



Muusikon ja albumin välille ei tarvita albumilla esiintymisestä kertovaa suhdetyyppiä, koska yhteys on johdettavissa polusta Muusikko – Soittaa – Kappale – Sisältää – Albumi

Osaa soittaa - ja Soittaa -suhdetyypit ovat molemmat tarpeellisia: Osaa soittaa -suhdetyyppi ilmaisee, mitä instrumentteja muusikko osaa soittaa. Soittaa-suhdetyyppi ilmaisee, mitä instrumentteja muusikko soittaa milläkin kappaleella.

Suunnitteluperiaatteita

- Tietokannan suunnittelun tärkeimpiä periaatteita ovat toistuvan tiedon välttäminen ja puuttuvien arvojen välttäminen.
  - Tiedon toistumiseen l. redundanssiin liittyy seuraavia ongelmia:
    - toistuva tieto vie tilaa turhaan
      - (Toistuvaa tietoa voidaan tosin sisällyttää tietokantaan tarkoituksellisestikin tehokkuussyistä. Ei tämä kurssin asioita.)
    - toistuvan tiedon ylläpitäminen kuluttaa turhaan resursseja ja on virhealtista
  - Puuttuviin arvoihin liittyy mm. seuraavia ongelmia:
    - mahdollinen tilan tuhlaaminen
    - puuttuvat arvot osattava huomioida kyselyissä
    - taulun rivien merkityksien tulkinta voi vaikeutua

Suunnitteluperiaatteita

- Kaikki tietokannan tiedot voitaisiin tallentaa yhteen tauluun, mutta tällainen taulu olisi ongelmallinen tietojen redundanssin ja puuttuvien arvojen vuoksi
- Pyritään suunnittelemaan ”sopivan” kokoisia tauluja ja välttämään redundantteja tietoja ja puuttuvia arvoja.
- Kun ER-kaavio tehdään ja muunnetaan SQL-tietokannaksi annettujen periaatteiden mukaisesti, pitäisi tuloksena olla tietokanta, jossa **ei ole turhaa redundanssia eikä turhia puuttuvia arvoja**.
  - Joskus tietokantaan kuitenkin halutaan **hallittua redundanssia** tehokkuuden ja/tai käyttömukavuuden vuoksi.
  - Tällöin on osattava valvoa, että tietojen johdonmukaisuus säilyy ts. ettei kantaan voi tallentaa ristiriitaista tietoa.
    - (Ei tämän kurssin asioita.)