

Tietokantojen perusteet

SQL – Useita tauluja

Taulujen väliset riippuvuudet

- Viiteavain, viite-eheys

- Tietokannan kaavion graafinen esitys

Kyselyjä useista tauluista

- Karteesinen tulo, liitosehto

- Liitosehto ja valintaehto

- Liitosoperaatiot

- Taulun liittäminen itseensä

Useita tauluja

- SQL-tietokanta koostuu yleensä useista tauluista, joiden tiedot liittyvät toisiinsa.
- Luentojen esimerkkisovellusalueella on yrityksen työntekijöille, osastoille ja projekteille kuillekin oma taulunsa.
- Taulujen tiedot liittyvät toisiinsa:
 - Kukin työntekijä työskentelee jollakin osastolla.
 - Osa työntekijöistä toimii toisten työntekijöiden esimiehinä.
 - Osa työntekijöistä osallistuu projekteihin tietyllä viikkotuntimäärällä.
- Taulujen tietoja liitetään toisiinsa **viiteavainrakenteen** (vierasavainrakenteen) avulla.

Viiteavain

- **Viiteavain** (foreign key) on
 - sarake tai sarakeryhdistelmä,
 - jonka kaikki arvot ovat jonkin taulun pääavaimen arvoja tai tyhjääroja
- Viiteavain voi viitata toiseen tauluun.

tyontekija

ttnro	etunimi	sukunimi	...	osastonro	esimiesnro
88	Jukka	Susi		1	
33	Ville	Viima		5	88
12	Pekka	Puro		5	33
98	Jenni	Joki		4	88
99	Alli	Kivi		4	98

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

Osastonro-sarake viittaa osasto-aulun sarakkeeseen onro. Se liittää tyontekija-aulun rivejä osasto-aulun riveihin.

Esim. Alli Kivi työskentelee Hallinto-osastolla.

Viiteavain

- Viiteavain voi viitata tauluun itseensä.
 - Liittää saman taulun rivejä toisiinsa.

tyontekija

ttnro	etunimi	sukunimi	...	osastonro	esimiesnro
88	Jukka	Susi		1	
33	Ville	Viima		5	88
12	Pekka	Puro		5	33
98	Jenni	Joki		4	88
99	Alli	Kivi		4	98

Esim.
Jukka Sudella ei
ole esimiestä.

Ville Viiman
esimies on
Jukka Susi.

Esimiesnro-sarake viittaa tyontekija-taulun sarakkeeseen ttnro. Se liittää tyontekija-taulun rivejä toisiinsa.

Viiteavain

- Viiteavain määritellään taulun luontilauseessa **FOREIGN KEY** -avainsanoilla

```
FOREIGN KEY(sarake {,sarake}) REFERENCES taulu  
[(pääavainsarake {,pääavainsarake})]
```

- Viiteavaimen on oltava yhteensopiva viitattavan taulun pääavaimen kanssa.
 - sarakkeiden lukumäärän ja
 - tietotyyppienon vastattava toisiaan

Viiteavain

```
CREATE TABLE osasto (  
  onro INT,  
  onimi VARCHAR(15) NOT NULL,  
  PRIMARY KEY (onro),  
  UNIQUE (onimi));
```

Taulu, johon viitataan
(jostakin toisesta
taulusta), on luotava
ensin.

```
CREATE TABLE tyontekija (  
  ttnro INT,  
  etunimi VARCHAR(15) NOT NULL,  
  sukunimi VARCHAR(20) NOT NULL,  
  saika DATE NOT NULL,  
  kotikunta VARCHAR(20) NOT NULL,  
  palkka NUMERIC(8,2),  
  puhelin VARCHAR(15),  
  osastonro INT NOT NULL,  
  esimiesno INT,  
  PRIMARY KEY (ttnro),  
  FOREIGN KEY (osastonro) REFERENCES osasto(onro),  
  FOREIGN KEY (esimiesno) REFERENCES tyontekija(ttnro));
```

Viiteavain

tyontekija

ttnro	etunimi	...	esimiesnro
88	Jukka		
33	Ville		88
12	Pekka		33
98	Jenni		88
99	Alli		98

projekti

pnro	pnimi
1	Tuote X
2	Tuote Y
3	Tuote Z

osallistuu

ttnro	pnro	tunnit
12	1	32.5
12	2	7.5
33	2	10.0
33	3	10.0
99	3	30.0
99	1	10.0
98	2	15.0

- Taulu voi toimia kahden taulun tietojen yhdistäjänä.

Osallistuu-taulu yhdistää tyontekija- ja projekti-
taulujen rivejä toisiinsa.

Esim. Pekka osallistuu Tuote X -projektiin 32.5
viikkotunnin verran.

Viiteavain

```
CREATE TABLE projekti (  
  pnro INT,  
  pnimi VARCHAR(15) NOT NULL,  
  PRIMARY KEY (pnro),  
  UNIQUE (pnimi));
```

osallistuu-taulun pääavain muodostuu ttnro- ja pnro-sarakkeista.

Kukin (ttnro, pnro)-pari voi esiintyä taulussa vain yhdellä rivillä.

```
CREATE TABLE osallistuu (  
  ttnro INT,  
  pnro INT,  
  tunnit NUMERIC(3,1),  
  PRIMARY KEY (ttnro,pnro),  
  FOREIGN KEY (ttnro) REFERENCES tyontekija,  
  FOREIGN KEY (pnro) REFERENCES projekti);
```

projekti

pnro	pnimi
1	Tuote X
2	Tuote Y
3	Tuote Z

osallistuu

ttnro	pnro	tunnit
12	1	32.5
12	2	7.5
33	2	10.0
33	3	10.0
99	3	30.0
99	1	10.0
98	2	15.0

Viittauksen kohteena olevan pääavainsarakkeen voi jättää pois.

Viite-eheys

- Viiteavainrajoitteita ei ole välttämätöntä antaa taulua määriteltäessä, mutta ...
- kun ne on määritelty, tietokannanhallintajärjestelmä (yleensä) tarkistaa tietoja päivitettäessä viittausten johdonmukaisuuden.
 - PostgreSQL tarkistaa, SQLite ei.
- Voidaan viitata vain olemassa oleviin riveihin ja arvoihin.

Alla oleva rivin lisääminen ei onnistu, koska osasto-
taulussa ei ole riviä, jolla onro = 10.

```
INSERT INTO tyontekija  
VALUES (102, 'Nelli', 'Naakka', '1988-09-09',  
'Toijala', 2900.00, NULL, 10, 33);
```

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

PostgreSQL-tietokannanhallintajärjestelmän antama virheilmoitus:

```
=> INSERT INTO tyontekija  
-> VALUES (102, 'Nelli', 'Naakka', '1988-09-09', 'Toijala', 2900.00, NULL, 10, 33);
```

```
ERROR: insert or update on table "tyontekija" violates  
foreign key constraint "tyontekija_osastonro_fkey"  
DETAIL: Key (osastonro)=(10) is not present in table "osasto".
```

Viite-eheys

- Viiteavainmäärittelyn yhteydessä voidaan antaa toimintasääntö, minkä mukaan toimitaan, jos muutos- tai poisto-operaatio (UPDATE tai DELETE) uhkaa rikkoa viite-eheyden.
- Toimintasäännöt PostgreSQL:ssä
 - NO ACTION
 - oletussääntö
 - Jos viitteen kohde katoaisi operaation seurauksena, operaatio estetään (ei tehdä muutosta tai poistoa) *
 - RESTRICT
 - Jos viitteen kohde katoaa, operaatio estetään*
 - CASCADE
 - Muutokset vyörytetään viitanneisiin riveihin
 - SET NULL
 - Jos viitteen kohde katoaa, asetetaan viitannut arvo tyhjäärvoksi.
 - SET DEFAULT
 - Jos viitteen kohde katoaa, asetetaan viitannut arvo oletusarvoksi.

*NO ACTION ja RESTRICT –toimintasäännöillä on ero PostgreSQL:ssä (liittyy transaktioihin); tämä on tietokantojen jatkokurssien asioita.

Viite-eheys

```
CREATE TABLE tyontekija (  
  ttnro INT,  
  etunimi VARCHAR(15) NOT NULL,  
  sukunimi VARCHAR(20) NOT NULL,  
  saika DATE NOT NULL,  
  kotikunta VARCHAR(20) NOT NULL,  
  palkka NUMERIC(8,2),  
  puhelin VARCHAR(15),  
  osastonro INT NOT NULL,  
  esimiesnro INT,  
  PRIMARY KEY (ttnro),  
  FOREIGN KEY (osastonro) REFERENCES osasto(onro),  
  FOREIGN KEY (esimiesnro) REFERENCES tyontekija(ttnro));
```

Kun viiteavain määritellään, on NO ACTION -toimintasääntö oletusarvoisesti voimassa muutos- ja poisto-operaatioille.

Viite-eheys: PostgreSQL:n toimintasäännöt

NO ACTION

Table "public.tyontekija"

Column	Type	Modifiers
ttnro	integer	not null
etunimi	character varying(15)	not null
sukunimi	character varying(20)	not null
saika	date	not null
kotikunta	character varying(20)	not null
palkka	numeric(8,2)	
puhelin	character varying(15)	
osastonro	integer	not null
esimiesnro	integer	

Indexes:

"tyontekija_pkey" PRIMARY KEY, btree (ttnro)

Foreign-key constraints:

"tyontekija_esimiesnro_fkey" FOREIGN KEY (esimiesnro) REFERENCES tyontekija(ttnro)

"tyontekija_osastonro_fkey" FOREIGN KEY (osastonro) REFERENCES osasto(onro)

Referenced by:

TABLE "osallistuu" CONSTRAINT "osallistuu_ttnro_fkey" FOREIGN KEY (ttnro) REFERENCES tyontekija(ttnro)

TABLE "tyontekija" CONSTRAINT "tyontekija_esimiesnro_fkey" FOREIGN KEY (esimiesnro) REFERENCES tyontekija(ttnro)

Viite-eheys

NO ACTION

- Viittauksen kohteena olevaa arvoa ei voi muuttaa eikä riviä poistaa.
- Muita arvoja voidaan muuttaa.
- Rivi, joka ei ole viittauksen kohteena, voidaan poistaa.

NO ACTION

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

Muutosoperaatiota

```
UPDATE osasto  
SET onro = 500  
WHERE onro = 5;
```

ei suoriteta, vaan seurauksena on tietokannanhallintajärjestelmän antama virheilmoitus:

```
=> UPDATE osasto  
-> SET onro = 500  
-> WHERE onro = 5;
```

```
ERROR: update or delete on table "osasto" violates  
foreign key constraint "tyontekija_osastonro_fkey" on table "tyontekija"  
DETAIL: Key (onro)=(5) is still referenced from table "tyontekija".
```

NO ACTION

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

Poisto-operaatiota

```
DELETE FROM osasto  
WHERE onro = 5;
```

ei suoriteta, vaan seurauksena on tietokannanhallintajärjestelmän antama virheilmoitus:

```
=> DELETE FROM osasto  
-> WHERE onro = 5;
```

```
ERROR: update or delete on table "osasto" violates foreign key constraint  
"tyontekija_osastonro_fkey" on table "tyontekija"
```

```
DETAIL: Key (onro)=(5) is still referenced from table "tyontekija".
```

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

```
UPDATE osasto
SET onimi = 'Tutkimuskeskus'
WHERE onro = 5;
```

```
UPDATE tyontekija
SET osastonro = 5,
    esimiesnro = 33
WHERE ttnro = 99;
```

taulut muutosoperaatioiden jälkeen:

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimuskeskus

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		5	33

NO ACTION

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus
6	Markkinointi

Muutos

```
UPDATE osasto  
SET onro = 600  
WHERE onro = 6;
```

onnistuu, koska arvo 6 ei ole viittauksen kohteena.

taulut muutosoperaation jälkeen:

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus
600	Markkinointi

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

NO ACTION

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus
6	Markkinointi

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

Poisto

```
DELETE FROM osasto  
WHERE onro = 6;
```

onnistuu, koska poistettava rivi (rivillä oleva arvo 6) ei ole viittauksen kohteena.

taulut poisto-operaation jälkeen:

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

Viite-eheys: PostgreSQL:n toimintasäännöt

CASCADE

```
CREATE TABLE tyontekija (  
  ttnro INT,  
  etunimi VARCHAR(15) NOT NULL,  
  sukunimi VARCHAR(20) NOT NULL,  
  saika DATE NOT NULL,  
  kotikunta VARCHAR(20) NOT NULL,  
  palkka NUMERIC(8,2),  
  puhelin VARCHAR(15),  
  osastonro INT NOT NULL,  
  esimiesnro INT,  
  PRIMARY KEY (ttnro),  
  FOREIGN KEY(osastonro) REFERENCES osasto(onro)  
  ON DELETE CASCADE ON UPDATE CASCADE,  
  FOREIGN KEY (esimiesnro) REFERENCES tyontekija(ttnro)  
  ON DELETE CASCADE ON UPDATE CASCADE);
```

Oletetaan kalvoilla 20-22, että tietokannassa on vain osasto- ja tyontekija-taulut.

Viite-eheys: PostgreSQL:n toimintasäännöt

CASCADE

Table "public.tyontekija"		
Column	Type	Modifiers
ttnro	integer	not null
etunimi	character varying(15)	not null
sukunimi	character varying(20)	not null
saika	date	not null
kotikunta	character varying(20)	not null
palkka	numeric(8,2)	
puhelin	character varying(15)	
osastonro	integer	not null
esimiesnro	integer	

Indexes:

"tyontekija_pkey" PRIMARY KEY, btree (ttnro)

Foreign-key constraints:

"tyontekija_esimiesnro_fkey" FOREIGN KEY (esimiesnro) REFERENCES tyontekija(ttnro) ON UPDATE CASCADE ON DELETE CASCADE

"tyontekija_osastonro_fkey" FOREIGN KEY (osastonro) REFERENCES osasto(onro) ON UPDATE CASCADE ON DELETE CASCADE

Referenced by:

TABLE "tyontekija" CONSTRAINT "tyontekija_esimiesnro_fkey" FOREIGN KEY (esimiesnro) REFERENCES tyontekija(ttnro) ON UPDATE CASCADE ON DELETE CASCADE

Viite-eheys: PostgreSQL:n toimintasäännöt

ON UPDATE CASCADE

- Kun viittauksen kohteena olevaa arvoa muutetaan, tehdään vastaava muutos päivitettyyn riviin viittanneisiin riveihin.

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

```
UPDATE osasto  
SET onro = 500  
WHERE onro = 5;
```

Taulut UPDATE-lauseen suorituksen jälkeen

onro	onimi
1	Pääkonttori
4	Hallinto
500	Tutkimus

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		500	88
12	Pekka		500	33
98	Jenni		4	88
99	Alli		4	98

Viite-eheys: PostgreSQL:n toimintasäännöt

ON DELETE CASCADE

- Kun viittauksen kohteena oleva rivi poistetaan, poistetaan myös siihen viittanneet rivit

osasto

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

tyontekija

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
33	Ville		5	88
12	Pekka		5	33
98	Jenni		4	88
99	Alli		4	98

DELETE FROM osasto
WHERE onro = 5;

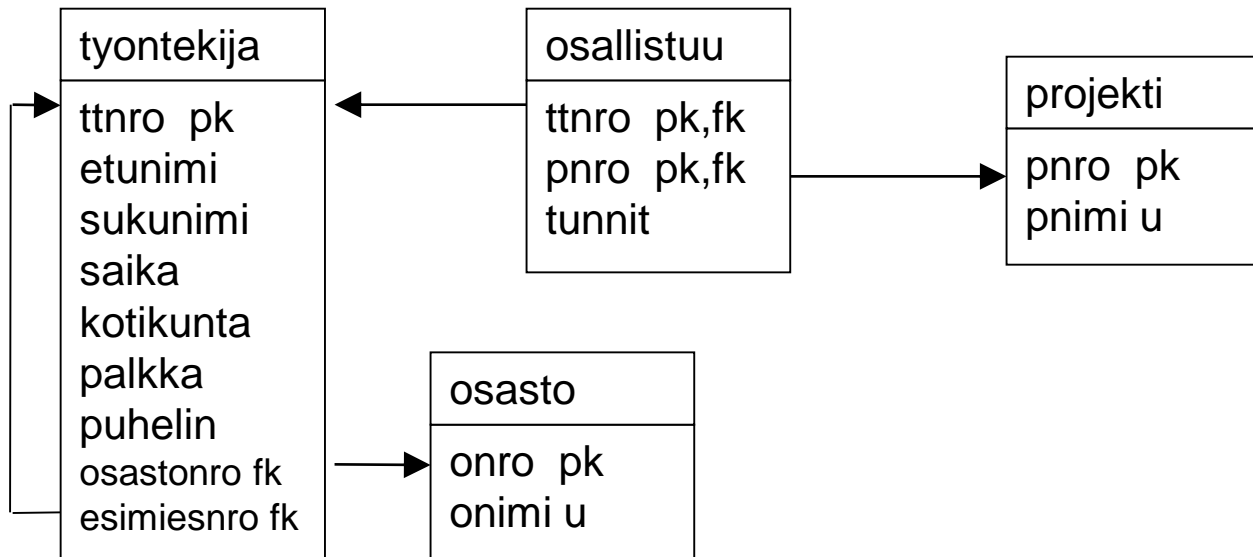
Taulut DELETE-lauseen suorituksen jälkeen

onro	onimi
1	Pääkonttori
4	Hallinto

ttnro	etunimi	...	osastonro	esimiesnro
88	Jukka		1	
98	Jenni		4	88
99	Alli		4	98

Pää- ja viiteavaimet tietokannan kaavion graafisessa esityksessä

- Eräs tietokannan kaavion graafinen esitystapa
 - taulut suorakaiteita, joissa sarakkeet lueteltu allekkain
 - pääavaimet merkitty pk-merkinnällä
 - avaimet merkitty u-merkinnällä
 - viiteavaimet merkitty fk-merkinnällä ja nuolilla: viiteavaimen kohdalta lähtee nuoli viittauksen kohteeseen



- Tietokannan kaavio on kokoelma tietokannan taulujen kaavioita.

Kyselyt useista tauluista

- SELECT-lauseen FROM-osassa voidaan luetella useita tauluja.
- Tulostauluna saadaan taulujen **karteellinen tulo** eli **ristitulo**.
 - Taulujen rivien kaikki mahdolliset kombinaatiot
- Tulostaulun rivit ovat yhdistettyjä rivejä, joilla on kaikki sarakkeet kaikista FROM-osan tauluista.

taulu1

s1	s2	s3
0	0	0
1	1	1
2	2	2

taulu2

sa	sb
1	aa
2	yy

FROM taulu1, taulu2

s1	s2	s3	sa	sb
0	0	0	1	aa
0	0	0	2	yy
1	1	1	1	aa
1	1	1	2	yy
2	2	2	1	aa
2	2	2	2	yy

FROM taulu1, taulu2, taulu3

s1	s2	s3	sa	sb	sx	sy
0	0	0	1	aa	1	ooo
0	0	0	1	aa	2	uuu
0	0	0	2	yy	1	ooo
0	0	0	2	yy	2	uuu
1	1	1	1	aa	1	ooo
1	1	1	1	aa	2	uuu
1	1	1	2	yy	1	ooo
1	1	1	2	yy	2	uuu
2	2	2	1	aa	1	ooo
2	2	2	1	aa	2	uuu
2	2	2	2	yy	1	ooo
2	2	2	2	yy	2	uuu

s1	s2	s3	sa	sb
0	0	0	1	aa
0	0	0	2	yy
1	1	1	1	aa
1	1	1	2	yy
2	2	2	1	aa
2	2	2	2	yy

sx	sy
1	ooo
2	uuu

s1	s2	s3
0	0	0
1	1	1
2	2	2

sa	sb
1	aa
2	yy

taulu1

s1	s2	s3
0	0	0
1	1	1
2	2	2

taulu2

sa	sb
1	aa
2	yy

taulu3

sx	sy
1	ooo
2	uuu

FROM taulu1, taulu2, taulu3

3 * 2 * 2 = 12 riviä

s1	s2	s3	sa	sb	sx	sy
0	0	0	1	aa	1	ooo
0	0	0	1	aa	2	uuu
0	0	0	2	yy	1	ooo
0	0	0	2	yy	2	uuu
1	1	1	1	aa	1	ooo
1	1	1	1	aa	2	uuu
1	1	1	2	yy	1	ooo
1	1	1	2	yy	2	uuu
2	2	2	1	aa	1	ooo
2	2	2	1	aa	2	uuu
2	2	2	2	yy	1	ooo
2	2	2	2	yy	2	uuu

Kyselyt useista tauluista: Karteesinen tulo

- Olkoon joukko $D_1 = \{A,B,C\}$ ja joukko $D_2 = \{1,2\}$.
- Tällöin joukkojen D_1 ja D_2 karteesinen tulo (ristitulo) $D_1 \times D_2$ on kaikkien niiden järjestettyjen parien (x,y) joukko, joissa x kuuluu joukkoon D_1 ja y kuuluu joukkoon D_2 .
 - $D_1 \times D_2 = \{(A,1), (A,2), (B,1), (B,2), (C,1), (C,2)\}$
- Olkoon $|D_i|$ joukon D_i alkioden lukumäärä. Tällöin karteesisen tulon (ristitulon) $D_1 \times D_2$ alkioden lukumäärä on
 - $|D_1| * |D_2| = 3 * 2 = 6$
- Olkoon joukot D_1 , D_2 ja D_3 . Tällöin $D_1 \times D_2 \times D_3$ on kaikkien niiden järjestettyjen kolmikkojen (x,y,z) joukko, joissa x kuuluu joukkoon D_1 , y kuuluu joukkoon D_2 ja z kuuluu joukkoon D_3
- Huom: SQL-tietokannan tauluissa voi olla duplikaattirivejä (taulut monijoukkoja).

```
SELECT *
FROM tyontekija, osasto;
```

tyontekija-taulun
sarakkeet

osasto-taulun
sarakkeet

tyontekija-taulussa
5 riviä (kalvo 3)

osasto-taulussa 3
riviä (kalvo 3)

karteesisessa
tulossa 15 riviä

ttnro	etunimi	...	osastonro	esimiesnro	onro	onimi
88	Jukka		1		1	Pääkonttori
33	Ville		5	88	1	Pääkonttori
12	Pekka		5	33	1	Pääkonttori
98	Jenni		4	88	1	Pääkonttori
99	Alli		4	98	1	Pääkonttori
88	Jukka		1		4	Hallinto
33	Ville		5	88	4	Hallinto
12	Pekka		5	33	4	Hallinto
98	Jenni		4	88	4	Hallinto
99	Alli		4	98	4	Hallinto
88	Jukka		1		5	Tutkimus
33	Ville		5	88	5	Tutkimus
12	Pekka		5	33	5	Tutkimus
98	Jenni		4	88	5	Tutkimus
99	Alli		4	98	5	Tutkimus

Kyselyt useista tauluista: **Liitosehto**

- Karteesisen tulon sijaan kyselyn tulokseksi halutaan yleensä **toisiinsa liittyviä tietoja**.
- Rajataan tulostauluun tulevia rivejä WHERE-osassa annettavalla **liitosehdolla**

Haetaan työntekijät osastoineen.

```
SELECT sukunimi, etunimi, onimi  
FROM tyontekija, osasto  
WHERE osastonro = onro;
```

liitosehto

sukunimi	etunimi	onimi
-----+-----+-----		
Susi	Jukka	Pääkonttori
Viima	Ville	Tutkimus
Puro	Pekka	Tutkimus
Joki	Jenni	Hallinto
Kivi	Alli	Hallinto

FROM tyontekija, osasto
WHERE osastonro = onro

ttnro	etunimi	...	osastonro	esimiesnro	onro	onimi
88	Jukka		1		1	Pääkonttori
33	Ville		5	88	1	Pääkonttori
12	Pekka		5	33	1	Pääkonttori
98	Jenni		4	88	1	Pääkonttori
99	Alli		4	98	1	Pääkonttori
88	Jukka		1		4	Hallinto
33	Ville		5	88	4	Hallinto
12	Pekka		5	33	4	Hallinto
98	Jenni		4	88	4	Hallinto
99	Alli		4	98	4	Hallinto
88	Jukka		1		5	Tutkimus
33	Ville		5	88	5	Tutkimus
12	Pekka		5	33	5	Tutkimus
98	Jenni		4	88	5	Tutkimus
99	Alli		4	98	5	Tutkimus

Kyselyt useista tauluista

Haetaan työntekijät osastoineen.

```
SELECT tyontekija.sukunimi, tyontekija.etunimi, osasto.onimi  
FROM tyontekija, osasto  
WHERE tyontekija.osastonro = osasto.onro;
```

sukunimi	etunimi	onimi
Susi	Jukka	Pääkonttori
Viima	Ville	Tutkimus
Puro	Pekka	Tutkimus
Joki	Jenni	Hallinto
Kivi	Alli	Hallinto

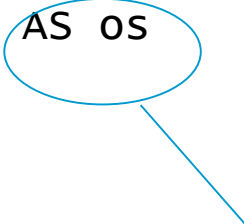
Taulun nimeä voidaan käyttää sarakkeen **tarkenteena**:

taulunimi.sarakenimi

Kyselyt useista tauluista

Haetaan työntekijät osastoineen.

```
SELECT tt.sukunimi, tt.etunimi, os.onimi  
FROM tyontekija AS tt, osasto AS os  
WHERE tt.osastonro = os.onro;
```



sukunimi	etunimi	onimi
Susi	Jukka	Pääkonttori
Viima	Ville	Tutkimus
Puro	Pekka	Tutkimus
Joki	Jenni	Hallinto
Kivi	Alli	Hallinto

Taululle voidaan antaa FROM-osassa **uusi, tilapäinen nimi**.

Lyhyttä tilapäistä nimeä käyttämällä pääsee pienemmällä kirjoittamisen vaivalla, kun taulujen nimiä käytetään tarkenteina.

Kyselyt useista tauluista: Kyselyn evaluointialgoritmi

Algoritmi, jonka tarkoituksena on auttaa ymmärtämään, miten kyselyn tulos muodostuu:

Vaihe 1. FROM-osa

- Muodostetaan FROM-osassa annettujen **taulujen karteellinen tulo eli ristitulo**, joka on tämän vaiheen tulostaulu.

Vaihe 2. WHERE-osa

- Poistetaan edellisen vaiheen tulostaulusta ne rivit, jotka eivät täytä **liitos-** ja/tai **valintaehtoja**.

Vaihe 3. SELECT-osa

- Poistetaan edellisen vaiheen tulostaulusta sarakkeet, jotka eivät esiinny SELECT-listassa.

4. ORDER BY -osa

- Järjestetään rivit järjestystekijöiden (sarakkeiden) mukaisesti.

Tämän algoritmin tarkoituksena on toimia ajattelun apuvälineenä.

Kyselyt useista tauluista: **Liitosehto ja valintaehto**

- Yhdistetään WHERE-osassa **liitosehto** ja **valintaehto AND**-operaattorilla.

Haetaan Hallinto-osastolla työskentelevät työntekijät.

```
SELECT sukunimi, etunimi, kotikunta
FROM tyontekija, osasto
WHERE osastonro = onro AND
      onimi = 'Hallinto';
```

liitosehto
valintaehto

sukunimi		etunimi		kotikunta
Joki		Jenni		Lempäälä
Kivi		Alli		Nokia

Tässä kyselyssä WHERE-osan ehto koostuu kahdesta alkeisehdosta, jotka on yhdistetty AND-operaattorilla (JA-operaattorilla). Jotta yhdistetty ehto tuottaa totuusarvon tosi (true), on molempien alkeisehtojen tuotettava totuusarvo tosi. Loogiset operaatiot käydään läpi myöhemmin.

Esimerkki: Kyselyn evaluointi

FROM tyontekija, osasto
Vaiheen 1 tulostaulu

Sovelletaan kyselyn
evaluointialgoritmia
edellisen kalvon kyselyyn.

ttnro	etunimi	...	osastonro	esimiesnro	onro	onimi
88	Jukka		1		1	Pääkonttori
33	Ville		5	88	1	Pääkonttori
12	Pekka		5	33	1	Pääkonttori
98	Jenni		4	88	1	Pääkonttori
99	Alli		4	98	1	Pääkonttori
88	Jukka		1		4	Hallinto
33	Ville		5	88	4	Hallinto
12	Pekka		5	33	4	Hallinto
98	Jenni		4	88	4	Hallinto
99	Alli		4	98	4	Hallinto
88	Jukka		1		5	Tutkimus
33	Ville		5	88	5	Tutkimus
12	Pekka		5	33	5	Tutkimus
98	Jenni		4	88	5	Tutkimus
99	Alli		4	98	5	Tutkimus

Esimerkki: Kyselyn evaluointi

```
WHERE osastonro = onro AND  
      onimi = 'Hallinto'
```

Vaiheen 2 tulostaulu:

karteesisesta tulosta poistettu ne rivit, jotka eivät täytä liitos- ja valintaehtoja

ttnro	etunimi	...	osastonro	esimiesnro	onro	onimi
98	Jenni		4	88	4	Hallinto
99	Alli		4	98	4	Hallinto

Esimerkki: Kyselyn evaluointi

```
SELECT sukunimi, etunimi, kotikunta
```

Vaiheen 3 tulostaulu:

Poistettu kaikki sarakkeet, jotka eivät esiinny SELECT-listassa.
(Sarakkeet järjestetty SELECT-listan mukaiseen järjestykseen.)

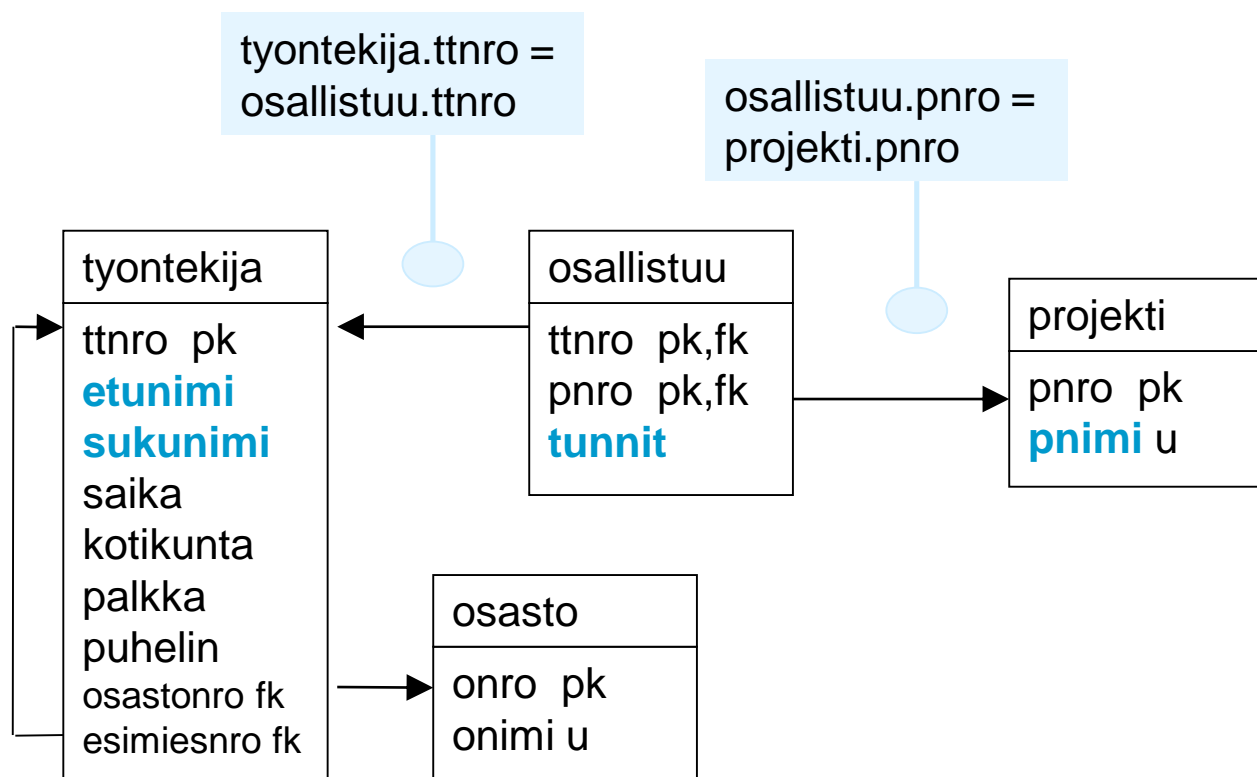
sukunimi	etunimi	kotikunta
Joki	Jenni	Lempäälä
Kivi	Alli	Nokia

Kyselyt useista tauluista

- Kyselyn rakentaminen:
 - Mistä tauluista otetaan tietoja mukaan tulostauluun?
 - Miten tiedot liitetään toisiinsa?
 - Jos FROM-osassa on annettu useita tauluja, tarvitaan yleensä liitosehto(ja) taulujen tietojen yhdistämiseksi.
 - Jos tauluja on N kappaletta, tarvitaan yleensä vähintään $N-1$ liitosehtoa.
 - Liitosehdossa testataan tyypillisesti pääavaimen ja viiteavaimen yhtäsuuruutta. Nämä voivat koostua useasta sarakkeesta → liitosehto muodostuu useasta alkeisehdosta
 - Tarvitaanko jotakin taulua tietojen liittämiseksi toisiinsa?
 - Tarvitaanko jotakin taulua tulostulosten rajaamista varten?

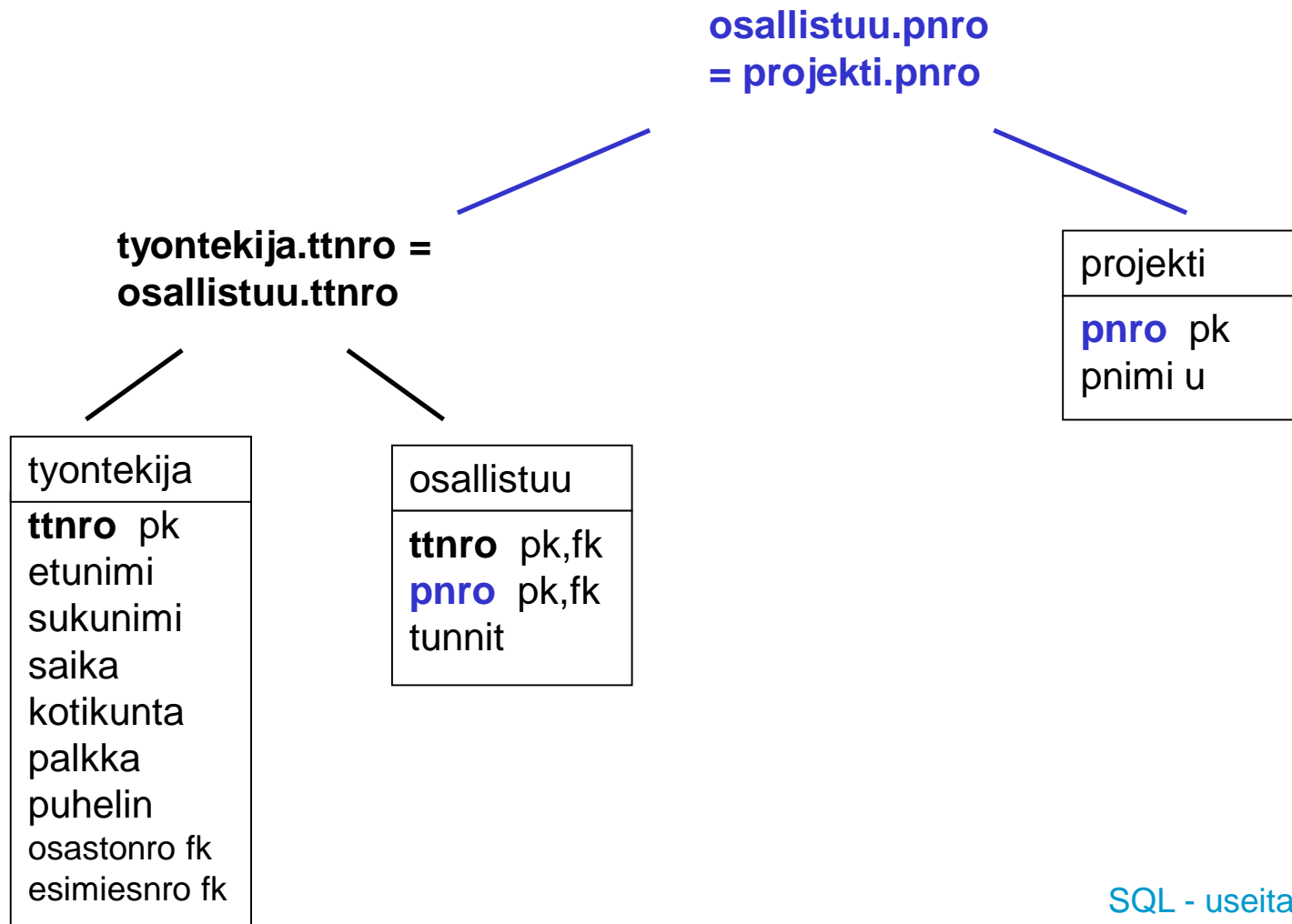
Kyselyt useista tauluista

Haetaan työntekijöiden etu- ja sukunimet, heidän projektinsa nimet ja projekteissa tehtävät tuntimäärät.



Kyselyt useista tauluista

Haetaan työntekijöiden etu- ja sukunimet, heidän projektinsa nimet ja projekteissa tehtävät tuntimäärät.



Kyselyt useista tauluista

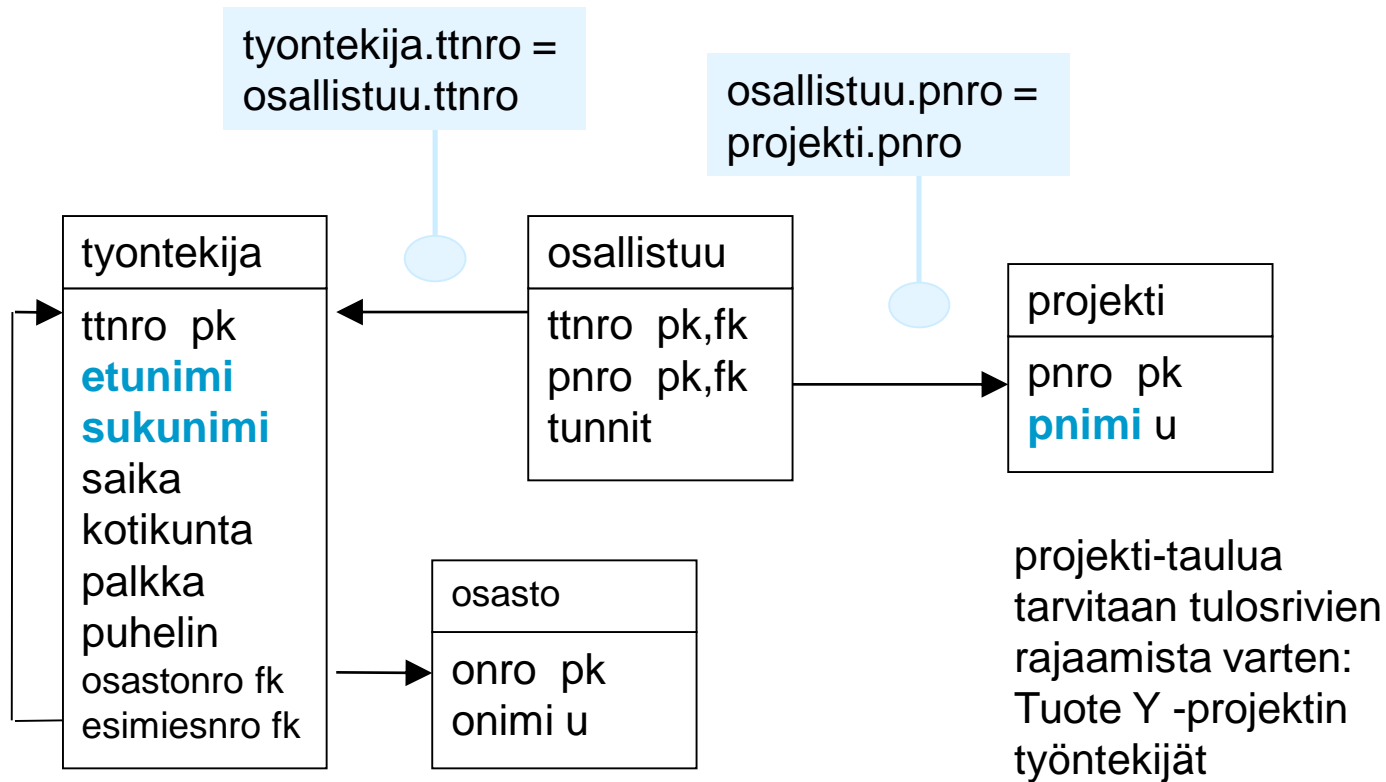
Tässä AS-sidesana on jätetty pois annettaessa taululle uutta, tilapäistä nimeä.

```
SELECT etunimi, sukunimi, pnimi, tunnit
FROM tyontekija tt, osallistuu o, projekti p
WHERE tt.ttnro = o.ttnro AND
      o.pnro = p.pnro;
```

etunimi	sukunimi	pnimi	tunnit
Pekka	Puro	Tuote X	32.5
Pekka	Puro	Tuote Y	7.5
Ville	Viima	Tuote Y	10.0
Ville	Viima	Tuote Z	10.0
Alli	Kivi	Tuote Z	30.0
Alli	Kivi	Tuote X	10.0
Jenni	Joki	Tuote Y	15.0

Samannimiset sarakkeet on erotettava toisistaan käyttämällä taulun nimeä tarkenteena:
taulunimi.sarakenimi

Haetaan Tuote Y -projektiin osallistuvien työntekijöiden etu- ja sukunimet.



```
SELECT etunimi,sukunimi
FROM tyontekija tt, osallistuu o, projekti p
WHERE tt.ttnro = o.ttnro AND
      o.pnro = p.pnro AND
      pnimi = 'Tuote Y';
```

etunimi	sukunimi
Pekka	Puro
Ville	Viima
Jenni	Joki

Kyselyt useista tauluista: **Liitosoperaatiot**

- FROM-osassa käytettävät **liitosoperaatiot (JOIN)** yhdistävät kaksi taulua yhdeksi tauluksi.

```
SELECT sarake {, sarake}  
FROM taulu1  
      liittostyyppi JOIN taulu2  
      ON liitosehto  
[WHERE ehto]
```

- Liitosehto
 - millä perusteella kahden taulun rivejä liitetään toisiinsa
- Liitostyyppi
 - INNER, LEFT OUTER, RIGHT OUTER, FULL OUTER
 - miten kohdellaan rivejä, joille liitosehto ei toteudu
- Huom. SQLitessa on toteutettu vain INNER JOIN ja LEFT OUTER JOIN, PostgreSQL:ssä kaikki liittostyytit.

Kyselyt useista tauluista: Liitosoperaatiot

- **INNER JOIN – liitos**
 - tuottaa samat rivit kuin karteesisen tulon ja WHERE-osan liitosehdon käyttäminen
- **OUTER JOIN - ulkoliitos**
 - **LEFT OUTER JOIN - vasen ulkoliitos**
 - liitoksen antamat rivit ja
 - myös ne vasemmanpuoleisen taulun rivit, joille liitosehto ei toteudu.
 - **RIGHT OUTER JOIN - oikea ulkoliitos**
 - liitoksen antamat rivit ja
 - myös ne oikeanpuoleisen taulun rivit, joille liitosehto ei toteudu.
 - **FULL OUTER JOIN - täysi ulkoliitos**
 - liitoksen antamat rivit ja
 - myös ne vasemmanpuoleisen taulun ja oikeanpuoleisen taulun rivit, joille liitosehto ei toteudu.

Kyselyt useista tauluista: Liitosoperaatiot

taulu1

nimi	laji
Ressu	koira
Kaustinen	lintu
Wagner	sika

taulu2

nimi	sarjakuva
Kaustinen	Tenavat
Wagner	Viivi ja Wagner
Lassi	Lassi ja Leevi

```
SELECT *  
FROM taulu1, taulu2  
WHERE taulu1.nimi = taulu2.nimi;
```

```
SELECT *  
FROM taulu1 INNER JOIN taulu2  
ON taulu1.nimi = taulu2.nimi;
```

nimi	laji	nimi	sarjakuva
Kaustinen	lintu	Kaustinen	Tenavat
Wagner	sika	Wagner	Viivi ja Wagner

Kyselyt useista tauluista: Liitoperaatiot

Haetaan Hallinto-osaston työntekijät.

```
SELECT sukunimi, etunimi, kotikunta
FROM tyontekija INNER JOIN osasto
      ON osastonro = onro
WHERE onimi = 'Hallinto';
```

liitoperaatio

liitosehto

valintaehto

sukunimi	etunimi	kotikunta
Joki	Jenni	Lempäälä
Kivi	Alli	Nokia

Kyselyt useista tauluista: Liitosoperaatiot

Haetaan työntekijöiden etu- ja sukunimet, heidän projektiensa nimet ja projekteissa tehtävät tuntimäärät.

Tehdään kysely liitosoperaatioiden avulla:

```
SELECT etunimi,sukunimi, pnimi, tunnit
FROM ((tyontekija tt
      INNER JOIN osallistuu o ON tt.ttnro = o.ttnro)
      INNER JOIN projekti p ON o.pnro = p.pnro);
```

```
SELECT etunimi,sukunimi, pnimi, tunnit
FROM tyontekija tt
      INNER JOIN osallistuu o ON tt.ttnro = o.ttnro
      INNER JOIN projekti p ON o.pnro = p.pnro;
```

Liitosoperaatioita voidaan ketjuttaa: edetään sisimmistä suluista uloimpiin.

Sulut voi jättää pois. Liitokset tehdään annetussa järjestyksessä vasemmalta oikealle edeten.

etunimi	sukunimi	pnimi	tunnit
Pekka	Puro	Tuote X	32.5
Pekka	Puro	Tuote Y	7.5
Ville	Viima	Tuote Y	10.0
Ville	Viima	Tuote Z	10.0
Alli	Kivi	Tuote Z	30.0
Alli	Kivi	Tuote X	10.0
Jenni	Joki	Tuote Y	15.0

Kyselyt useista tauluista: Liitosoperaatiot

```
SELECT *  
FROM taulu1 LEFT OUTER JOIN taulu2  
    ON taulu1.nimi = taulu2.nimi;
```

nimi	laji	nimi	sarjakuva
Kaustinen	lintu	Kaustinen	Tenavat
Ressu	koira		
Wagner	sika	Wagner	Viivi ja Wagner

```
SELECT *  
FROM taulu1 RIGHT OUTER JOIN taulu2  
    ON taulu1.nimi = taulu2.nimi;
```

nimi	laji	nimi	sarjakuva
Kaustinen	lintu	Kaustinen	Tenavat
		Lassi	Lassi ja Leevi
Wagner	sika	Wagner	Viivi ja Wagner

Kyselyt useista tauluista: Liitosoperaatiot

```
SELECT *  
FROM taulu1 FULL OUTER JOIN taulu2  
    ON taulu1.nimi = taulu2.nimi;
```

nimi	laji	nimi	sarjakuva
Kaustinen	lintu	Kaustinen	Tenavat
		Lassi	Lassi ja Leevi
Ressu	koira		
Wagner	sika	Wagner	Viivi ja Wagner

- HUOM!
 - FROM-osassa käytettäviä liitosoperaatioita (JOIN) ei ole toteutettu kaikissa järjestelmissä ja niiden syntaksi voi vaihdella järjestelmittäin.
 - SQLitessa on toteutettu vain INNER JOIN ja LEFT OUTER JOIN, PostgreSQL:ssä kaikki liitostyypit.

Kyselyt useista tauluista: Liitosoperaatiot

Haetaan kaikki työntekijät, heidän mahdolliset projektinsa ja niihin tehtävät tuntimäärät.

```
SELECT etunimi, sukunimi, pnimi, tunnit
FROM ((tyontekija tt
      LEFT OUTER JOIN osallistuu o ON tt.ttnro = o.ttnro)
      LEFT OUTER JOIN projekti p ON o.pnro = p.pnro);
```

etunimi	sukunimi	pnimi	tunnit
Pekka	Puro	Tuote X	32.5
Pekka	Puro	Tuote Y	7.5
Ville	Viima	Tuote Y	10.0
Ville	Viima	Tuote Z	10.0
Jukka	Susi		
Jenni	Joki	Tuote Y	15.0
Alli	Kivi	Tuote X	10.0
Alli	Kivi	Tuote Z	30.0

Kyselyt useista tauluista: Liitosoperaatiot

Haetaan kaikki työntekijät, joiden kotikunta on Tampere, ja heidän mahdollisten projektinsa numerot.

```
SELECT tt.ttnro, sukunimi, etunimi, kotikunta, pnro AS projektinnumero
FROM tyontekija tt LEFT OUTER JOIN osallistuu o
  ON tt.ttnro = o.ttnro AND kotikunta = 'Tampere';
```

Ei näin.

ttnro	sukunimi	etunimi	kotikunta	projektinnumero
12	Puro	Pekka	Tampere	1
12	Puro	Pekka	Tampere	2
33	Viima	Ville	Nokia	
98	Joki	Jenni	Lempäälä	
99	Kivi	Alli	Nokia	
88	Susi	Jukka	Tampere	

```
SELECT tt.ttnro, sukunimi, etunimi, kotikunta, pnro AS projektinnumero
FROM tyontekija tt LEFT OUTER JOIN osallistuu o
  ON tt.ttnro = o.ttnro
WHERE kotikunta = 'Tampere';
```

**Vaan
näin.**

ttnro	sukunimi	etunimi	kotikunta	projektinnumero
88	Susi	Jukka	Tampere	
12	Puro	Pekka	Tampere	1
12	Puro	Pekka	Tampere	2

Kyselyt useista tauluista: Kyselyn evaluointi

Vaihe 1. FROM-osa

- Jos FROM-osassa ei ole liitosoperaatioita,
 - muodostetaan annettujen **taulujen karteesinen tulo eli ristitulo**, joka on tämän vaiheen tulostaulu.
- Jos FROM-osassa on liitosoperaatio(ita),
 - muodostetaan liitoksen (karteesinen tulo ja liitosehto) antama tulostaulu ja
 - lisätään tulostauluun mahdollisen ulkoliitoksen antamat ”lisärivit”.
 - (Käydään läpi kaikki liitosoperaatiot vastaavalla tavalla.)

Vaihe 2. WHERE-osa

- Poistetaan edellisen vaiheen tulostaulusta ne rivit, jotka eivät täytä **liitos-** ja/tai **valintaehtoja**.

Vaihe 3. SELECT-osa

- Poistetaan edellisen vaiheen tulostaulusta sarakkeet, jotka eivät esiinny SELECT-listassa.

4. ORDER BY -osa

- Järjestetään rivit järjestystekijöiden (sarakkeiden) mukaisesti.

Kyselyt useista tauluista: Taulun liittäminen itseensä

- Kyselyssä voidaan liittää saman taulun rivejä toisiinsa antamalla sama taulu useaan kertaan FROM-osassa.
- Taululle on annettava tällöin FROM-osassa uusi, tilapäinen nimi.

Haetaan työntekijät ja heidän esimiehensä.

```
SELECT tt.etunimi, tt.sukunimi, em.sukunimi  
FROM tyontekija AS tt, tyontekija AS em  
WHERE tt.esimiesnro = em.ttnro;
```

Tulostaulun
sarakkeelle voidaan
antaa uusi nimi.

AS pomo

AS em

taululle uusi,
tilapäinen nimi

ttnro	etunimi	...	esimiesnro	ttnro	etunimi	...	esimiesnro
88	Jukka			88	Jukka		
88	Jukka			33	Ville		88
88	Jukka			12	Pekka		33
88	Jukka			98	Jenni		88
88	Jukka			99	Alli		98
33	Ville		88	88	Jukka		
33	Ville		88	33	Ville		88
33	Ville		88	12	Pekka		33
33	Ville		88	98	Jenni		88
33	Ville		88	99	Alli		98
12	Pekka		33	88	Jukka		
12	Pekka		33	33	Ville		88
12	Pekka		33	12	Pekka		33
12	Pekka		33	98	Jenni		88
12	Pekka		33	99	Alli		98
98	Jenni		88	88	Jukka		
98	Jenni		88	33	Ville		88
98	Jenni		88	12	Pekka		33
98	Jenni		88	98	Jenni		88
98	Jenni		88	99	Alli		98
99	Alli		98	88	Jukka		
99	Alli		98	33	Ville		88
99	Alli		98	12	Pekka		33
99	Alli		98	98	Jenni		88
99	Alli		98	99	Alli		98

```
FROM tyontekija AS tt,
      tyontekija AS em
WHERE tt.esimiesnro =
      em.ttnro;
```

Kyselyt useista tauluista: Taulun liittäminen itseensä

```
SELECT tt.etunimi, tt.sukunimi, em.sukunimi AS pomo
FROM tyontekija AS tt, tyontekija AS em
WHERE tt.esimiesnro = em.ttnro;
```

etunimi	sukunimi	pomo
Ville	Viima	Susi
Pekka	Puro	Viima
Jenni	Joki	Susi
Alli	Kivi	Joki

Kyselyt useista tauluista: Taulun liittäminen itseensä

```
SELECT em.etunimi AS esimies
FROM tyontekija tt, tyontekija em
WHERE tt.esimiesnro = em.ttnro AND
      tt.ttnro = 12;
```

Haetaan
työntekijän nro
12 lähin esimies.

```
esimies
-----
ville
```

```
SELECT eem.etunimi AS esimiehen_esimies
FROM tyontekija tt, tyontekija em, tyontekija eem
WHERE tt.esimiesnro = em.ttnro AND
      em.esimiesnro = eem.ttnro AND
      tt.ttnro = 12;
```

Haetaan
työntekijän nro
12 lähimmän
esimiehen lähin
esimies

```
esimiehen_esimies
-----
Jukka
```


Kyselyt useista tauluista: Taulun liittäminen itseensä

```
SELECT tt.etunimi, tt.sukunimi, em.sukunimi AS pomo
FROM tyontekija AS tt INNER JOIN tyontekija AS em
      ON tt.esimiesnro = em.ttnro;
```

etunimi	sukunimi	pomo
Ville	Viima	Susi
Pekka	Puro	Viima
Jenni	Joki	Susi
Alli	Kivi	Joki

```
SELECT tt.etunimi, tt.sukunimi, em.sukunimi AS pomo
FROM tyontekija AS tt LEFT OUTER JOIN tyontekija AS em
      ON tt.esimiesnro = em.ttnro;
```

etunimi	sukunimi	pomo
Jukka	Susi	
Ville	Viima	Susi
Pekka	Puro	Viima
Jenni	Joki	Susi
Alli	Kivi	Joki

Haetaan kaikki työntekijät ja heidän mahdolliset esimiehensä.