



Tietokantojen perusteet

Sisäkkäisistä kyselyistä

Sisäkkäisiä kyselyjä

- Alikysely on kysely, joka on upotettu toisen, "laajemman" kyselyn sisälle.
- Laajempaa kyselyä kutsutaan pääkyselyksi tai ulommaksi kyselyksi, alikyselyä kutsutaan myös sisemmäksi kyselyksi.
- Alikysely voi olla WHERE-, HAVING-, FROM- tai SELECT-osassa.
 - Tällä kurssilla käsitellään alikyselyjä, jotka ovat WHERE-, HAVING- tai FROM-osassa.

pääkysely →

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE ttnro IN
      (SELECT huoltajanro
       FROM huollettava);
```

← alikysely

Sisäkkäisiä kyselyjä

- Alikysely voi palauttaa **yhden tai useampia tulosrivejä**.
- Jos WHERE- tai HAVING-osan kysely voi palauttaa **useita rivejä**, on tällöin ulommassa kyselyssä käytettävä **joukko-operaatiota**.
 - IN
 - EXISTS
 - vertailuoperaattori ja ANY (tai SOME)
 - = ANY
 - > ANY
 - >= ANY
 - < ANY
 - <= ANY
 - <> ANY
 - vertailuoperaattori ja ALL
 - kuten ANY-määreen tapauksessa

HUOM. SQLite:ssä ei ole toteutettu ANY- ja ALL-avainsanoja.

Alikysely WHERE-osassa

- Sisäkkäisen kyselyn evaluointistrategia, kun alikysely on WHERE-osassa:
 - Muodosta ulomman kyselyn FROM-osan antama tulostaulu kuten aiemminkin (karteesinen tulo tai liitosoperaation tuottama taulu)
 - Kullekin edellisen vaiheen tulostaulun riville:
 - Testaa WHERE-osan ehto.
 - Ehdon testaamista varten (uudelleen)suorita alikysely.

Alikysely WHERE-osassa: IN

- IN-operaattorin avulla tutkitaan, kuuluuko arvo (tai arvojen yhdistelmä esim. PostgreSQL:ssä) alikyselyn tulokseen. IN palauttaa totuusarvon tosi, jos arvo kuuluu alikyselyn tulokseen.
 - Alikyselyn tulos on joukko tai monijoukko. (Nyt joukon alkioita ei luetella eksplisiittisesti.)

Kysely hakee tietoja työntekijöistä, joilla on huollettavia.

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE ttnro IN
      (SELECT huoltajanro
       FROM huollettava);
```

Ulommassa kyselyssä tutkitaan, kuuluuko työntekijänumero (ttnro) alikyselyn tuottamaan tulostauluun.

etunimi	sukunimi	ttnro
ville	viima	33
Pekka	Puro	12
Jenni	Joki	98

Alikyselyn tuloksena työntekijänumerot (huoltajanro) huollettava-taulusta. Tässä alikyselyn tulos on monijoukko.

Alikysely WHERE-osassa: IN

Haetaan niiden työntekijöiden etu- ja sukunimet ja työntekijänumerot, joilla on Aamu-niminen huollettava.

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE ttnro IN
      (SELECT huoltajanro
       FROM huollettava
       WHERE nimi = 'Aamu');
```

etunimi	sukunimi	ttnro
Ville	Viima	33
Pekka	Puro	12

Uloimmassa kyselyssä tutkitaan, kuuluuko ttnro alikyselyn tuottamaan tulostauluun.

Alikyselyn tuloksena niiden henkilöiden työntekijänumerot, joilla on Aamu-niminen huollettava.

(huollettava-aulusta löytyvät huoltajanro-sarakkeen arvot, joille pätee, että rivillä oleva nimi = 'Aamu').

Alikysely WHERE-osassa: NOT IN

- NOT IN palauttaa totuusarvon tosi, jos arvo (tai arvojen yhdistelmä) ei kuulu alikyselyn tuottamaan tulokseen.

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE ttnro NOT IN
      (SELECT huoltajanro
       FROM huollettava);
```

etunimi	sukunimi	ttnro
Jukka	Susi	88
Alli	Kivi	99

Kysely hakee niiden työntekijöiden tietoja, joilla ei ole huollettavia.

Alikysely WHERE-osassa: EXISTS

- EXISTS-operaattori testaa, sisältääkö alikyselyn tulos yhtään riviä. Operaattori palauttaa totuusarvon tosi, jos alikyselyn tulos ei ole tyhjä (ts. tulos sisältää yhden tai useamman rivin).

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE EXISTS
  (SELECT *
   FROM huollettava
   WHERE tyontekija.ttnro = huollettava.huoltajanro);
```

etunimi	sukunimi	ttnro
ville	viima	33
Pekka	Puro	12
Jenni	Joki	98

Kysely hakee niiden työntekijöiden tietoja, joilla on huollettavia.

Alikysely WHERE-osassa: EXISTS

- Aiemmissa kyselyissä (kalvot 5-7) alikysely voitiin suorittaa erillään pääkyselystä (ulommasta kyselystä).
- Nyt on kyseessä **kytketty alikysely**:
 - Alikyselyn WHERE-osan ehdossa viitataan ulommassa kyselyssä annetun taulun sarakkeeseen.
 - Alikysely on evaluoitava erikseen jokaista ulomman kyselyn FROM-osan tulostaulun riviä kohden.

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE EXISTS
  (SELECT *
   FROM huollettava
   WHERE tyontekija.ttnro = huollettava.huoltajanro);
```



Alikyselyn tulos riippuu ulomman kyselyn "nykyisestä" rivistä: ehto
tyontekija.ttnro = huollettava.huoltajanro

Alikysely WHERE-osassa: EXISTS

Alikyselyn tulos riippuu ulomman kyselyn
"nykyisestä" rivistä:
tyontekija.ttnro = huollettava.huoltajanro

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE EXISTS
  (SELECT *
   FROM huollettava
   WHERE tyontekija.ttnro = huollettava.huoltajanro AND
        nimi = 'Aamu');
```

Lisäksi mukana valintaehto.

etunimi	sukunimi	ttnro
-----+-----+-----		
ville	Viima	33
Pekka	Puro	12

Alikysely WHERE-osassa: NOT EXISTS

- NOT EXISTS palauttaa totuusarvon tosi, jos alikyselyn tulokseen ei sisälly yhtään riviä.

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE NOT EXISTS
  (SELECT *
   FROM huollettava
   WHERE tyontekija.ttnro = huollettava.huoltajanro);
```

etunimi	sukunimi	ttnro
Jukka	Susi	88
Alli	Kivi	99

Kysely hakee niiden työntekijöiden tietoja, joilla ei ole huollettavia.

Useita alikyselyjä WHERE-osassa

- WHERE-osassa voi esiintyä useita alikyselyjä.

Kysely hakee tietoja esimiehenä toimivista työntekijöistä, jotka eivät osallistu mihinkään projektiin.

```
SELECT etunimi, sukunimi, ttnro
FROM tyontekija
WHERE ttnro IN (SELECT esimiesnro
                FROM tyontekija)
      AND
      ttnro NOT IN
      (SELECT ttnro
       FROM osallistuu);
```

etunimi	sukunimi	ttnro
Jukka	Susi	88

Alikysely WHERE-osassa: Vertailuoperaattori ilman ANY- tai ALL-määrettä

- Kun alikysely palauttaa vain yhden tulosrivin, vertailuoperaattoria voidaan käyttää ilman ANY- tai ALL-määrettä.

Haetaan niiden työntekijöiden etunimet, sukunimet ja palkat, joiden palkka on yhtä suuri kuin maksimipalkka.

```
SELECT etunimi, sukunimi, palkka  
FROM tyontekija  
WHERE palkka = (SELECT MAX(palkka)  
                FROM tyontekija);
```



Työntekijän palkan on oltava yhtä suuri kuin alikyselyn palauttaman maksimipalkan.

etunimi	sukunimi	palkka
Jukka	Susi	5500.00

Alikysely WHERE-osassa: Vertailuoperaattori ilman ANY- tai ALL-määrettä

Haetaan niiden työntekijöiden etunimet, sukunimet ja palkat, joiden palkka on suurempi kuin minimipalkka.

```
SELECT etunimi, sukunimi, palkka  
FROM tyontekija  
WHERE palkka > (SELECT MIN(palkka)  
                FROM tyontekija);
```



Työntekijän palkan on oltava suurempi kuin alikyselyn palauttaman minimipalkan.

etunimi	sukunimi	palkka
Jukka	Susi	5500.00
Ville	Viima	4000.50
Pekka	Puro	3000.00
Jenni	Joki	4300.00

Alikysely WHERE-osassa: Vertailuoperaattori ja ANY

- vertailuoperaattori ja ANY (SOME)
 - Joissakin järjestelmissä voidaan käyttää myös SOME-määrettä.
 - HUOM: Avainsanoja ANY ja SOME ei ole toteutettu SQLitessa.

Haetaan niiden työntekijöiden etunimet, sukunimet ja palkat, joiden palkka on suurempi kuin minimipalkka.

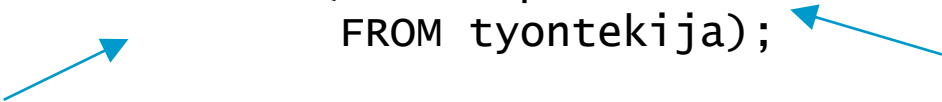
```
SELECT etunimi, sukunimi, palkka  
FROM tyontekija  
WHERE palkka > ANY (SELECT palkka  
                     FROM tyontekija);
```



ANY: vertailuehdon on oltava **voimassa jonkin** alikyselyn tuottaman tulostaulun **arvon suhteen**

Alikysely WHERE-osassa: Vertailuoperaattori ja ANY

```
SELECT etunimi, sukunimi, palkka  
FROM tyontekija  
WHERE palkka > ANY (SELECT palkka  
                     FROM tyontekija);
```



Testataan, onko palkka suurempi
kuin jokin alikyselyn tuloksen
palkoista.

Alikyselyn tuloksena
palkat tyontekija-
taulusta.

etunimi	sukunimi	palkka
Jukka	Susi	5500.00
Ville	Viima	4000.50
Pekka	Puro	3000.00
Jenni	Joki	4300.00

Alikysely WHERE-osassa: Vertailuoperaattori ja ALL

- vertailuoperaattori ja ALL

Haetaan niiden työntekijöiden etunimi, sukunimi ja palkka, joilla on korkein palkka.

```
SELECT etunimi, sukunimi, palkka  
FROM tyontekija  
WHERE palkka >= ALL (SELECT palkka  
                     FROM tyontekija);
```



ALL: vertailuehdon
on oltava **voimassa**
kaikkien alikyselyn
tulostaulun **arvojen**
suhteen

Alikysely WHERE-osassa: Vertailuoperaattori ja ALL

```
SELECT etunimi, sukunimi, palkka  
FROM tyontekija  
WHERE palkka >= ALL (SELECT palkka  
                     FROM tyontekija);
```

Testataan, onko palkka suurempi tai yhtä suuri kuin kaikki alikyselyn tuloksena saatavat palkat.

Alikyselyn tuloksena palkat tyontekija-taulusta.

etunimi	sukunimi	palkka
Jukka	Susi	5500.00

Tässä kannan tilassa korkeimman palkan omaavia työntekijöitä vain yksi, mutta jossakin muussa tilassa heitä voisi olla useita

Alikysely WHERE-osassa

Vertailuoperaattori ja ALL

HUOM!

Alla olevat kyselyt tuottavat esimerkkietokannassa saman tuloksen, koska työntekija-taulussa on jokaisella rivillä palkka-sarakkeessa ei-tyhjä arvo.

```
SELECT etunimi, sukunimi, palkka
FROM tyontekija
WHERE palkka >= ALL (SELECT palkka
                     FROM tyontekija);
```

Ylempi kysely ei toimi halutulla tavalla (haetaan suuripalkkaisin heistä, joiden palkka tiedetään), jos palkka-sarakkeessa on tyhjäarvo(ja).

Mutta sen alikyselyä voi muuttaa niin, että se (siis alikysely) palauttaa ei-tyhjäarvot ...

```
SELECT etunimi, sukunimi, palkka
FROM tyontekija
WHERE palkka = (SELECT MAX(palkka)
               FROM tyontekija);
```

Alempi kysely sen sijaan toimii kaikissa tietokannan tiloissa halutulla tavalla.

Alikysely HAVING-osassa

- Alikyselyä voidaan hyödyntää, kun asetetaan HAVING-osassa ehtoja GROUP BY -osassa tuotetuille ryhmille.
- HAVING-osassa voi olla useita alikyselyjä.

Hae nimet ja maksimipalkat niille osastoille, joilla on ainakin yksi työntekijä, jonka palkka on yhtä suuri kuin kaikkien työntekijöiden maksimipalkka.

```
SELECT onimi, MAX(palkka) max_palkka
FROM osasto o INNER JOIN tyontekija t
    ON o.onro=t.osastonro
GROUP BY onimi
HAVING MAX(palkka) = (SELECT MAX(palkka)
                      FROM tyontekija);
```

onimi	max_palkka
Pääkonttori	5500.00

Alikysely HAVING-osassa

- SQL:ssä ei voi käyttää aggregointifunktioita sisäkkäin.
 - Esim. `max(count(...))` ei ole mahdollista.
- Tällöin voidaan ottaa avuksi HAVING-osassa oleva alikysely.
 - Huom: ALL-avainsanaa ei ole toteutettu SQLitessa.

Haetaan niiden osastojen nimet, joilla on eniten työntekijöitä.

```
SELECT onimi
FROM osasto, tyontekija
WHERE onro = osastonro
GROUP BY onimi
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
                        FROM tyontekija
                        GROUP BY osastonro);
```

```
onimi
-----
Hallinto
Tutkimus
```

Alikysely FROM-osassa

- Alikysely (tai useita alikyselyjä) voi esiintyä kyselyn FROM-osassa.
- Tällä tavalla voidaan yhdistää yksittäisillä riveillä olevia tietoja (rivikohtaisia tietoja) ja eri tavoin tuotettuja yhteenvetotietoja.
- Esimerkkikysely:

Haetaan kullekin työntekijälle

- työntekijänumero
- sukunimi
- etunimi
- palkka
- osasto, jolla työntekijä työskentelee
- osaston työntekijöiden lukumäärä
- osaston työntekijöiden minimipalkka
- osaston työntekijöiden maksimipalkka

Alikysely FROM-osassa

```
SELECT ttnro, sukunimi, etunimi, palkka, onimi,  
       tyontekija_lkm, min_palkka, max_palkka  
FROM   tyontekija, osasto,  
       (SELECT osastonro,  
              COUNT(*) AS tyontekija_lkm,  
              MIN(palkka) AS min_palkka,  
              MAX(palkka) AS max_palkka  
        FROM tyontekija  
        GROUP BY osastonro) AS osastotietoja  
WHERE  tyontekija.osastonro = osasto.onro AND  
       tyontekija.osastonro = osastotietoja.osastonro;
```

osastokohtaisia
yhteenvetotietoja, jotka on
laskettu yhden tai useamman
tyontekija-taulun rivin perusteella

ttnro	sukunimi	etunimi	palkka	onimi	tyontekija_lkm	min_palkka	max_palkka
88	Susi	Jukka	5500.00	Pääkonttori	1	5500.00	5500.00
33	Viima	Ville	4000.50	Tutkimus	2	3000.00	4000.50
12	Puro	Pekka	3000.00	Tutkimus	2	3000.00	4000.50
98	Joki	Jenni	4300.00	Hallinto	2	2500.00	4300.00
99	Kivi	Alli	2500.00	Hallinto	2	2500.00	4300.00

Rivikohtaisia tietoja

Sisäkkäisiä kyselyjä: Vaihtoehto ulkoiselle liitosoperaatiolle

- Ulkoisten liitosoperaatioiden avulla voidaan yhdistää tietoja kahdesta taulusta, vaikkei kaikille riveille löytyisikään liitosehdon mukaista paria.
 - LEFT OUTER JOIN
 - RIGHT OUTER JOIN
 - FULL OUTER JOIN
- Tämä onnistuu myös WHERE-osan liitosehtoa, UNION-operaatiota ja sisäkkäistä kyselyä käyttämällä.

Sisäkkäisiä kyselyjä: Vaihtoehto ulkoiselle liitosoperaatiolle

Haetaan kullekin työntekijälle työntekijänumero, etunimi, sukunimi ja huollettavien lukumäärä.

```
SELECT ttnro, etunimi, sukunimi, COUNT(nimi) AS huollettavia
FROM tyontekija LEFT OUTER JOIN huollettava
    ON ttnro = huoltajanro
GROUP BY ttnro, etunimi, sukunimi
ORDER BY sukunimi, etunimi;
```

ttnro	etunimi	sukunimi	huollettavia
98	Jenni	Joki	1
99	Alli	Kivi	0
12	Pekka	Puro	2
88	Jukka	Susi	0
33	Ville	Viima	3

Sisäkkäisiä kyselyjä: Vaihtoehto ulkoiselle liitosoperaatiolle

```
SELECT ttnro, etunimi, sukunimi, COUNT(nimi) AS huollettavia
FROM tyontekija, huollettava
WHERE ttnro = huoltajanro
GROUP BY ttnro, etunimi, sukunimi;
```

ttnro	etunimi	sukunimi	huollettavia
12	Pekka	Puro	2
33	Ville	Viima	3
98	Jenni	Joki	1

```
SELECT ttnro, etunimi, sukunimi, 0 AS huollettavia
FROM tyontekija
WHERE ttnro NOT IN
      (SELECT huoltajanro
       FROM huollettava);
```

ttnro	etunimi	sukunimi	huollettavia
88	Jukka	Susi	0
99	Alli	Kivi	0

Sisäkkäisiä kyselyjä: Vaihtoehto ulkoiselle liitosoperaatiolle

```
SELECT ttnro, etunimi, sukunimi, COUNT(nimi) AS huollettavia
FROM tyontekija, huollettava
WHERE ttnro = huoltajanro
GROUP BY ttnro, etunimi, sukunimi
UNION
SELECT ttnro, etunimi, sukunimi, 0 AS huollettavia
FROM tyontekija
WHERE ttnro NOT IN
      (SELECT huoltajanro
       FROM huollettava)
ORDER BY sukunimi, etunimi; ← Järjestetään tulosrivit
```

ttnro	etunimi	sukunimi	huollettavia
98	Jenni	Joki	1
99	Alli	Kivi	0
12	Pekka	Puro	2
88	Jukka	Susi	0
33	Ville	Viima	3

Sisäkkäisiä kyselyjä

- Sisäkkäinen kysely pystytään usein korvaamaan yksinkertaisella (ts. ei-sisäkkäisellä) kyselyllä.
- Sisäkkäisiä kyselyjä kannattaa käyttää ”säästeliäästi”; sisäkkäiset kyselyt tulevat helposti monimutkaisiksi niin kyselyn myöhemmän lukijan kuin tietokannanhallintajärjestelmänkin kannalta.