IN2090 – Databaser og datamodellering

10 – Outer joins og mengdeoperatorer: Eksempler

Leif Harald Karlsen
leifhka@ifi.uio.no

Universitetet i Oslo

Oppgave 1: Løsning

Finn navn og antall bestillinger for alle kunder som har gjort færre enn 5 bestillinger

```
  SELECT c.company_name, count(o.order_id) AS num_orders
    FROM customers AS c
         LEFT OUTER JOIN orders AS o USING (customer_id)
GROUP BY c.company_name
  HAVING count(o.order_id) < 5;
```

Oppgave 2: Løsning

Finn ut antall ganger hver ansatt har håndtert en ordre fra hver kunde

```
WITH
  all_combinations AS (
    SELECT e.employee_id,
           e.first_name || ' ' || e.last_name AS fullname,
           c.customer_id,
           c.company_name
    FROM employees AS e, customers AS c -- Kryssprodukt, alle kombinasjoner
  )
SELECT ac.fullname, ac.company_name, count(o.order_id) AS num_transactions
FROM all_combinations AS ac
     LEFT OUTER JOIN orders AS o USING (employee_id, customer_id)
GROUP BY ac.company_name, ac.fullname;
```

Oppgave 3: Løsning

Finn ut for hvor mye penger hver kunde har kjøpt for, for de som har bestilt færre enn 100 produkter totalt

```
  SELECT c.company_name,
         sum(d.unit_price * d.quantity * (1 - d.quantity) AS sum_money
    FROM customers AS c
         LEFT OUTER JOIN orders AS o USING (customer_id)
         LEFT OUTER JOIN order_details AS d USING (order_id)
GROUP BY c.company_name
  HAVING sum(d.quantity) < 100 OR
         sum(d.quantity) IS NULL; -- MERK: NULL < 100 er NULL
```

Oppgave 4: Løsning

Finn alle filmer og serier som er laget i Norge

```
SELECT 'Serie' AS type, s.maintitle AS title
FROM series AS s INNER JOIN filmcountry AS c ON (s.seriesid = c.filmid)
WHERE c.country = 'Norway'
UNION
SELECT 'Film' AS type, f.title AS title
FROM film AS f INNER JOIN filmcountry AS c USING (filmid)
WHERE c.country = 'Norway';
```

Oppgave 5: Løsning

Finn ut hvor mange filmer og serier om ble laget hvert år, sorter etter antall

```
WITH
  years AS (
    SELECT prodyear AS year FROM film
    UNION ALL
    SELECT firstprodyear AS year FROM series
  )
SELECT year, count(*) AS nr
FROM years
GROUP BY year
ORDER BY nr;
```

Oppgave 6: Løsning

Finn ut hvor mange filmer og serier om ble laget hvert tiår, sorter etter antall (vanskelig!)

```
WITH
  years AS (
    SELECT prodyear AS year FROM film
    UNION ALL
    SELECT firstprodyear AS year FROM series
  )
SELECT year/10, count(*) AS nr
FROM years
GROUP BY year/10 -- Heltallsdivisjon her gir tiår
ORDER BY nr;
```

Oppgave 6: Løsning (penere output)

Finn ut hvor mange filmer og serier om ble laget hvert tiår, sorter etter antall (vanskelig!)

```
WITH
  years AS (
    SELECT prodyear AS year FROM film
    UNION ALL
    SELECT firstprodyear AS year FROM series
  )
SELECT ((year/10)*10)::text || ' - ' || (((year/10)*10)+9)::text AS tiår,
       count(*) AS nr
FROM years
GROUP BY year/10
ORDER BY nr;
```

Mengdeoperatorer – oppførsel

Gitt en tabell/spørring `t`. Er følgende riktig?

- `t UNION t = t`? Nei, `UNION` fjerner alle duplikater
- `t UNION ALL t = t`? Nei, vi får hver rad i `t` to ganger
- `t INTERSECT t = t`? Nei, samme som for `UNION`
- `t INTERSECT ALL t = t`? Ja!
- `t EXCEPT t` blir tomt? Ja!
- `t EXCEPT ALL t` blit tomt? Ja!
- `t LEFT OUTER JOIN t USING (col) = t INNER JOIN t USING (col)`? Nei, med mindre `col` er `NOT NULL`

Eksempler: Rekursive spørringer (Ikke pensum) (1)

Finn alle tall fra 1 til 100

```
WITH RECURSIVE
  numbers AS (
    (SELECT 1 AS n)
    UNION
    (SELECT n+1 AS n
     FROM numbers
     WHERE n < 100)
  )
SELECT * FROM numbers;
```

Eksempler: Rekursive spørringer (Ikke pensum) (2)

Finn alle Fibonacci-tall mindre enn 100000

```
WITH RECURSIVE
  fibs AS (
    (SELECT 1 AS n, 1 AS m)
    UNION
    (SELECT m AS n, n+m AS m
     FROM fibs
     WHERE m < 100000)
  )
SELECT n FROM fibs;
```

Eksempler: Rekursive spørringer (Ikke pensum) (3)

Finn ut alle *sjef-av*-relasjoner (hvor dersom `a` er sjef for `b` og `b` er sjef for `c` er også `a` sjef for `c`)

```
WITH RECURSIVE
  bossof AS (
    (SELECT employee_id, reports_to
     FROM employees)
    UNION
    (SELECT e.employee_id, b.reports_to
     FROM employees AS e INNER JOIN bossof AS b
          ON (e.reports_to = b.employee_id))
  )
SELECT * FROM bossof;
```