IN2090 – Databaser og datamodellering

09 – Avanserte eksempler

Leif Harald Karlsen
leifhka@ifi.uio.no

Universitetet i Oslo

Enklere syntaks for joins

- Man kan bruke `USING (<kolonne>)` fremfor `ON (a.<kolonne> = b.<kolonne>)`
- For eksempel:

```
SELECT p.product_name, c.category_name
  FROM products AS p
       INNER JOIN categories AS c USING (category_id);
```

- Merk: Må fortsatt bruke `ON` dersom kolonnene har ulikt navn

Eksempel: Variable i delspørringer (1)

Finn navnet på alle produkter som har lavere pris nå enn gjennomsnittsprisen den er solgt for tidligere [1 rad]

```
SELECT p.product_name
FROM products AS p
WHERE p.unit_price <
      (SELECT avg(d.unit_price)
       FROM order_details AS d
       WHERE p.product_id = d.product_id);
```

Merk: Man kan bruke variabler fra en spørring i dens delspørringer

Eksempel: Variable i delspørringer (2)

Finn antall produkter for hver kategori

```
SELECT c.category_name,
       (SELECT count(*)
        FROM products AS p
        WHERE p.category_id = c.category_id) AS nr_products
FROM categories AS c
```

Utlede informasjon om entiteter

- Aggregering i grupper, sortering og å begrense svaret er svært nyttig når man har store mengder data
- Når vi grupperer kan vi enten utlede implisitt informasjon om allerede eksisterende entiteter
- Eller lage nye entiteter fra attributter
- Sortering og begrensning lar oss hente ut de mest interessante objektene
- Dette gjør også at vi kan lage langt mer interessante views

Eksempel 1: Implisitt informasjon om kategorier

For hver kategori, finn høyeste, laveste, og gjennomsnittspris på produktene i kategorien, samt antall produkter

```
SELECT c.category_id, c.category_name,
       max(p.unit_price) AS highest,
       min(p.unit_price) AS lowest,
       avg(p.unit_price) AS average,
       count(*) AS nr_products
FROM categories AS c
     INNER JOIN products AS p USING (category_id)
GROUP BY c.category_id, c.category_name;
```

Eksempel 2: Implisitt informasjon om land

Finn de tre mest kjøpte produktene for hvert land

```
WITH
  bought_by_country AS (
    SELECT c.country, p.product_name, count(*) AS nr_bought
    FROM products AS p
         INNER JOIN order_details AS d USING (product_id)
         INNER JOIN orders AS o USING (order_id)
         INNER JOIN customers AS c USING (customer_id)
    GROUP BY c.country, p.product_name
    ORDER BY nr_bought DESC
  ),
  countries AS (
    SELECT DISTINCT country FROM customers
  )
SELECT c.country,
       (SELECT s.product_name FROM bought_by_country AS s
        WHERE s.country = c.country
        LIMIT 1) AS first_place,
       (SELECT s.product_name FROM bought_by_country AS s
        WHERE s.country = c.country
        OFFSET 1
        LIMIT 1) AS second_place,
       (SELECT s.product_name FROM bought_by_country AS s
        WHERE s.country = c.country
        OFFSET 2
        LIMIT 1) AS third_place
FROM countries AS c;
```

Anbefalingssystem (Komplisert eksempel! Utenfor pensum)

Vi vil lage en spørring som finner ut:

- hvilke produkter vi kan anbefale en kunde å kjøpe,
- basert på hva kunden har kjøpt,
- og hva andre kunder som har kjøpt det samme har kjøpt.

Anbefalingssystem (Komplisert eksempel! Utenfor pensum)

```sql
WITH
  bought AS ( -- Relaterer kunde-IDer til produkt-IDene til det de har kjøpt
    SELECT DISTINCT c.customer_id, d.product_id -- Vil ikke ha duplikater!
    FROM customers AS c
         INNER JOIN orders USING (customer_id)
         INNER JOIN order_details AS d USING (order_id)
  ),
  correspondences AS ( -- Relaterer par av produkter til antallet ganger disse er kjøpt av samme kunde
    SELECT b1.product_id AS prod1, b2.product_id AS prod2, count(*) AS correspondence
    FROM bought AS b1
         INNER JOIN bought b2 USING (customer_id)
    WHERE b1.product_id != b2.product_id -- Fjern par hvor produktene er like
    GROUP BY b1.product_id, b2.product_id -- Gruppér på par av produkter
    HAVING count(*) > 18 -- Antall korrespondanser bør være litt høyt
  ),
  reccomend AS ( -- Relaterer kunde-IDer til anbefalte produkters IDer
    SELECT DISTINCT b.customer_id, c.prod2 AS product_id -- Vil ikke ha duplikater!
    FROM correspondences AS c
         INNER JOIN bought AS b
         ON (b.product_id = c.prod1)
    WHERE NOT c.prod2 IN       -- Fjern produkter som kunden allerede har kjøpt
            (SELECT product_id
             FROM bought AS bi
             WHERE b.customer_id = bi.customer_id)
   )
-- Til slutt finn navn på både kunde og produkt og aggreger produktnavnene
-- for hver kunde i ett array med aggregatfunksjonen array_agg
SELECT c.company_name, array_agg(p.product_name) AS reccomended_products
FROM customers AS c INNER JOIN reccomend AS r USING (customer_id)
     INNER JOIN products AS p USING (product_id)
GROUP BY c.customer_id, c.company_name;
```

Takk for nå!

Neste uke handler om mengdeoperatorer og ytre joins.