IN2090 – Databaser og datamodellering

10 – Mengdeoperatorer

Leif Harald Karlsen
leifhka@ifi.uio.no

Universitetet i Oslo

Mengdeoperatorer

- Vi har nå et relativt uttrykningskraftig språk for å hente ut informasjon fra en database
- Men det er noen elementære ting vi fortsatt ikke kan gjøre
- F.eks. kombinere svar fra to spørringer til én tabell
- Eller trekke svarene fra en spørring fra en annen
- Husk at vi kan se på svarene fra `SELECT` som en (multi-)mengde
- SQL tillater oss å bruke vanlige mengdeoperatorer (snitt, union, osv.)
- Ettersom SQLs tabeller er multimengder har vi to versjoner av hver operator:
  - én versjon som behandler resultatene som mengder (f.eks. `UNION`)
  - én versjon som behandler dem som multimengder (f.eks. `UNION ALL`)
- Disse mengdeoperatorene puttes *mellom to spørringer*

Mengdeoperatorene

- Vi har følgende mengdeoperatorer:
  - Union – `UNION`
  - Snitt – `INTERSECT`
  - Differanse – `EXCEPT`
- For alle disse har vi i tillegg en variant med `ALL` etter seg som behandler resultatene som multimengder
- Antall ganger en rad er med i resultatet av:
  - `q1 UNION ALL q2` er summen av antall ganger raden er med i `q1` og `q2`
  - `q1 INTERSECT ALL q2` er det minste antall ganger raden er med i `q1` og `q2`
  - `q1 EXCEPT ALL q2` er antall ganger raden er med i `q1` minus antallet ganger den er med i `q2`

Union-operatoren

persons

ID	Name	Phone	Email
1	Per	48123456	per@mail.no
2	Mari	NULL	mari@umail.net
3	Ola	NULL	NULL
4	Ida	98765432	NULL

```
(SELECT *
 FROM persons
 WHERE Phone IS NOT NULL)
UNION
(SELECT *
 FROM persons
 WHERE Email IS NOT NULL)
```

Resultat:

ID	Name	Phone	Email
1	Per	48123456	per@mail.no
4	Ida	98765432	NULL
2	Mari	NULL	mari@umail.net

```
(SELECT *
 FROM persons
 WHERE Phone IS NOT NULL)
UNION ALL
(SELECT *
 FROM persons
 WHERE Email IS NOT NULL)
```

Resultat:

ID	Name	Phone	Email
1	Per	48123456	per@mail.no
4	Ida	98765432	NULL
1	Per	48123456	per@mail.no
2	Mari	NULL	mari@umail.net

Union-kompatibilitet

- Hva skjer om vi tar unionen av to spørringer som returnerer forskjellig antall kolonner?

```
(SELECT Name, Phone
 FROM person
 WHERE Phone IS NOT NULL)
UNION
(SELECT Name, Phone, Email
 FROM person
 WHERE Email IS NOT NULL)
```

- Vi får en error! Spørringen gir ikke mening.
- For å ta unionen av to spørringer må de returnere like mange kolonner
- Kolonnene må også ha kompatible typer
- Kan f.eks. ta unionen av en kolonne med `integer` og `decimal`, får da en kolonne av typen `numeric`
- Alle mengdeoperatorer må ha union-kompatibilitet mellom delspørringene

Eksempel: Union

Finn navn og by på alle leverandør- (eng.: supplier) og kundefirmaer fra Tyskland

```
(SELECT company_name, city
 FROM customers
 WHERE country = 'Germany')
UNION
(SELECT company_name, city
 FROM suppliers
 WHERE country = 'Germany');
```

Snitt-operatoren

persons

ID	Name	Country
1	Per	UK
2	Mari	Norway
3	Ola	Norway
4	Ida	Italy
5	Carl	USA

companies

ID	Name	Country
1	Per's company	Germany
2	Fish'n trolls	Norway
3	Matpakke AS	Norway
4	Big Burgers	USA
5	Ysteriet	Norway

```
(SELECT Country
 FROM persons)
INTERSECT
(SELECT Country
 FROM companies)
```

Resultat:

Country
Norway
USA

```
(SELECT Country
 FROM person)
INTERSECT ALL
(SELECT Country
 FROM companies)
```

Resultat:

Country
Norway
Norway
USA

Differanse-operatoren

persons

ID	Name	Country
1	Per	UK
2	Mari	Norway
3	Ola	Norway
4	Ida	Italy
5	Carl	USA

companies

ID	Name	Country
1	Per's company	Germany
2	Fish'n trolls	Norway
3	Matpakke AS	Norway
4	Big Burgers	USA
5	Ysteriet	Norway

```
(SELECT Country
 FROM companies)
EXCEPT
(SELECT Country
 FROM persons)
```

Resultat:

Country
Germany

```
(SELECT Country
 FROM companies)
EXCEPT ALL
(SELECT Country
 FROM persons)
```

Resultat:

Country
Germnay
Norway

EXISTS

- Av og til er vi kun interessert i om en del spørring *har et svar*, og ikke svaret i seg selv
- Typisk er dette når vi er interessert i å hente ut objekter med en bestemt egenskap, men hvor egenskapen kan avgjøres med en delspørring
- I slike tilfeller kan vi bruke `EXISTS` før en delspørring i `WHERE`-klausulen
- `EXISTS q` er sann for en spørring `q` dersom `q` har minst ett svar
- Kan også bruke `NOT EXISTS q` for å finne ut om `q` ikke har noen svar

EXISTS-nøkkelordet

companies

ID	Name	Country
1	Per's company	Germany
2	Fish'n trolls	Norway
3	Matpakke AS	Norway
4	Big Burgers	USA
5	Ysteriet	Norway

persons

ID	Name	Country
1	Per	UK
2	Mari	Norway
3	Ola	Norway
4	Ida	Italy
5	Carl	USA

```sql
SELECT p.Name
FROM persons AS p
WHERE NOT EXISTS (
        SELECT * -- Kan bruke hva som helst her
        FROM companies AS c
        WHERE c.country = p.country
      );
```

Resultat:

p.Name
Per
Ida

Mange måter å gjøre det samme på

Finn ID på alle kunder som ikke har bestilt noe:

Med `EXCEPT`

```
(SELECT customer_id
 FROM customers)
EXCEPT
(SELECT customer_id
 FROM orders);
```

Med `NOT IN`

```
SELECT customer_id
FROM customers
WHERE customer_id NOT IN (
  SELECT customer_id
  FROM orders);
```

Med `NOT EXISTS`

```
SELECT c.customer_id
FROM customers AS c
WHERE NOT EXISTS (
 SELECT * FROM orders AS o
 WHERE o.customer_id = c.customer_id
);
```

Med `LEFT OUTER JOIN`

```
SELECT c.customer_id
FROM customers AS c
     LEFT OUTER JOIN orders AS o
     USING (customer_id)
WHERE o.customer_id IS NULL;
```

Mye vi ikke har sett på

Følgende er nyttige ting vi ikke har sett på (og ikke del av pensum):

- Viduspørringer
  `https://www.postgresql.org/docs/current/tutorial-window.html`
- Rekursive spørringer
  `https://www.postgresql.org/docs/current/queries-with.html`
- Lateral join
  Sek. 7.2.1.5 i
  `https://www.postgresql.org/docs/current/queries-table-expressions.html`
- Triggere
  `https://www.postgresql.org/docs/current/plpgsql-trigger.html`
- Lage egne SQL-funksjoner og typer
  `https://www.postgresql.org/docs/current/xfunc.html`
  `https://www.postgresql.org/docs/current/xtypes.html`

Takk for nå!

Neste uke handler om programmering med SQL.