IN2090 – Databaser og datamodellering

10 – Ytre joins

Leif Harald Karlsen
leifhka@ifi.uio.no

Universitetet i Oslo

Aggregering og NULL

- Aggregering med `sum`, `min`, `max` og `avg` ignorerer `NULL`-verdier
- Det betyr også at dersom det kun er `NULL`-verdier i en kolonne blir resultatet av disse `NULL`
- `count(*)` teller med `NULL`-verdier
- Men dersom vi oppgir en konkret kolonne, f.eks. `count(product_name)` vil den kun telle verdiene som ikke er `NULL`
- For eksempel:

Person

Name	Age
Per	2
Kari	4
Mari	NULL

```
SELECT min(Age) FROM Person;    --> 2
SELECT avg(Age) FROM Person;    --> 3
SELECT count(Age) FROM Person; --> 2
SELECT count(*) FROM Person;    --> 3

SELECT sum(Age) FROM Person
WHERE Name = 'Mari';            --> NULL

SELECT count(Age) FROM Person
WHERE Name = 'Mari';            --> 0
```

Repetisjon: Inner joins

Hvilken kunde har kjøpt hvilket produkt?

```
SELECT ProductName, Customer
FROM products AS p INNER JOIN orders AS o
     ON p.ProductID = o.ProductID
```

Resultat

products

ProductID	Name	Price
0	TV 50 inch	8999
1	Laptop 2.5GHz	7499

orders

OrderID	ProductID	Customer
0	1	John Mill
1	1	Peter Smith
2	0	Anna Consuma
3	1	Yvonne Potter

Inner joins og manglende verdier

Hvilken kunde har kjøpt hvilket produkt?

```
SELECT ProductName, Customer
FROM products AS p INNER JOIN orders AS o
     ON p.ProductID = o.ProductID
```

Resultat

products

ProductID	Name	Price
0	TV 50 inch	8999
1	Laptop 2.5GHz	7499
2	Noise-amplifying Headphones	9999

orders

OrderID	ProductID	Customer
0	1	John Mill
1	1	Peter Smith
2	0	Anna Consuma
3	1	Yvonne Potter

Inner joins og manglende verdier med aggregater

Hvor mange har kjøpt hvert produkt?

```
SELECT p.ProductName, count(o.Customer) AS num
FROM products AS p INNER JOIN orders AS o
     ON p.ProductID = o.ProductID
GROUP BY p.ProductName
```

Resultat

products

ProductID	Name	Price
0	TV 50 inch	8999
1	Laptop 2.5GHz	7499
2	Noise-amplifying Headphones	9999

orders

OrderID	ProductID	Customer
0	1	John Mill
1	1	Peter Smith
2	0	Anna Consuma
3	1	Yvonne Potter

Problemer med Indre joins

- I forige spørring fikk vi ikke opp at 0 kunder har kjøpt Noise-amplifying Headset
- Årsaken er at den ikke joiner med noe, og derfor forsvinner fra svaret
- For å få ønsket resultat trenger vi altså en ny type join
- De nye joinene som løser problemet vårt heter ytre joins, eller *outer join* på engelsk

Outer Joins

- Vi har flere varianter av ytre joins, nemlig
  - `left outer join`
  - `right outer join`
  - `full outer join`
- Brukes ved å bytte ut `INNER JOIN` med f.eks. `LEFT OUTER JOIN`
- Hovedidéen bak denne typen join er å bevare alle rader fra en eller begge tabellene i joinen
- Og så fylle inn med `NULL` hvor vi ikke har noen match

Left Outer Join

- I en *left outer join* vil alle rader i den venstre tabellen bli med i svaret
- Resultatet av `a LEFT OUTER JOIN b ON (a.c1 = b.c2)` blir
  - samme som `a INNER JOIN b ON (a.c1 = b.c2)`,
  - men hvor alle rader fra `a` som ikke matcher noen i `b`
  - (altså hvor `a.c1` ikke er lik noen `b.c2`)
  - blir lagt til resultatet, med `NULL` for alle `b`s kolonner

Eksempel: Left Outer Join

Left outer join mellom `products` og `orders`

```
SELECT *
FROM products AS p LEFT OUTER JOIN orders AS o
     ON p.ProductID = o.ProductID;
```

Resultat

products

ProductID	Name	Price
0	TV 50 inch	8999
1	Laptop 2.5GHz	7499
2	Noise-amplifying Headphones	9999

orders

OrderID	ProductID	Customer
0	1	John Mill
1	1	Peter Smith
2	0	Anna Consuma
3	1	Yvonne Potter

Eksempel: Left Outer Join

Hvor mange har kjøpt hvert produkt?

```
    SELECT p.ProductName, count(o.Customer) AS num
    FROM products AS p LEFT OUTER JOIN orders AS o
        ON p.ProductID = o.ProductID
GROUP BY p.ProductName
```

Resultat

products

ProductID	Name	Price
0	TV 50 inch	8999
1	Laptop 2.5GHz	7499
2	Noise-amplifying Headphones	9999

orders

OrderID	ProductID	Customer
0	1	John Mill
1	1	Peter Smith
2	0	Anna Consuma
3	1	Yvonne Potter

Andre nyttige bruksområder for ytre joins

- Som vi ser er ytre joins nyttige når vi aggregerer, for å ikke miste resultater underveis
- Ytre joins kan også være nyttige for å kombinere ufullstendig informasjon fra flere tabeller
- For eksempel:

Persons

ID	Name
1	Per
2	Mari
3	Ida

Numbers

ID	Phone
1	48123456
3	98765432

Emails

ID	Email
1	per@mail.no
2	mari@umail.net

```
SELECT p.Name, n.Phone, e.Email
FROM Persons AS p
     LEFT OUTER JOIN Numbers AS n
       ON (p.ID = n.ID)
     LEFT OUTER JOIN Emails AS e
       ON (p.ID = e.ID);
```

p.Name	n.Phone	e.Email
Per	48123456	per@mail.no
Mari	NULL	mari@umail.net
Ida	98765432	NULL

Andre ytre joins

- `a RIGHT OUTER JOIN b ON (a.c1 = b.c2)` er akkurat det samme som `b LEFT OUTER JOIN a ON (b.c2 = a.c1)`
- Altså, i en *right outer join* vil alle radene i den høyre tabellen være med i resultatet
- Vi har også en `FULL OUTER JOIN` som er en slags kombinasjon, her vil ALLE rader være med i svaret
- For eksempel:

Persons

ID	Name
1	Per
2	Mari

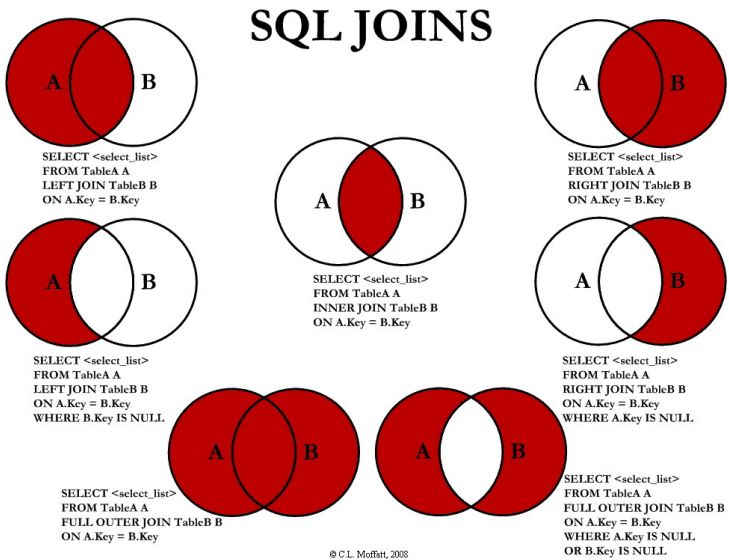
Numbers

ID	Phone
1	48123456
3	98765432

```
SELECT p.Name, n.Phone
FROM Persons AS p
     FULL OUTER JOIN Numbers AS n
       ON (p.ID = n.ID);
```

p.Name	n.Phone
Per	48123456
Mari	NULL
NULL	98765432

Oversikt over joins



Ytre join-eksempel (1)

Finn navn på alle kunder som har gjort 2 eller færre 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) <= 2;
```

Ytre join-eksempel (2)

Finn ut for hvor mange produkter i hver kategori firmaet Leka Trading supplier

```
WITH
  supplies AS (
    SELECT category_id
    FROM suppliers INNER JOIN products USING (supplier_id)
    WHERE company_name = 'Leka Trading'
  )
SELECT c.category_name, count(s.category_id) AS nr_products
FROM categories AS c
     LEFT OUTER JOIN supplies AS s USING (category_id)
GROUP BY c.category_name;
```

Syntaks for joins

I stedet for

- `LEFT OUTER JOIN` kan man skrive `LEFT JOIN`
- `RIGHT OUTER JOIN` kan man skrive `RIGHT JOIN`
- `FULL OUTER JOIN` kan man skrive `FULL JOIN`
- `INNER JOIN` kan man skrive `JOIN`

Takk for nå!

Neste video handler om mengdeoperatorer.