IN2090 - Databaser og datamodellering

05 - FROM-klausulen og joins

Leif Harald Karlsen leifhka@ifi.uio.no



Kombinere informasjon fra flere tabeller

• (Enkle) SELECT-spørringer har formen:

```
SELECT <kolonner>
FROM <tabeller>
WHERE <uttrykk>
```

- Frem til nå har vi bare sett på spørringer over én og én tabell
- Ofte ønsker vi å kombinere informasjon fra ulike tabeller
- Dette kan gjøres ved å legge til flere tabeller i FROM-klausulen

Spørringer over flere tabeller

Hva skjer dersom vi putter flere tabeller i FROM?

To tabeller i FROM

SELECT *

FROM products, orders

Resultat

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

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

Spørringer over flere tabeller

Hva skjer dersom vi putter flere tabeller i FROM?

To tabeller i FROM

```
SELECT *
  FROM products, orders
```

Resultat - Fargekodet

products

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

orders			
OrderID OrderedProduct Cust		Customer	
0	1	John Mill	
1	1	Peter Smith	
2	0	Anna Consuma	
3	1	Yvonne Potter	

Kryssprodukt

- Med flere tabeller i FROM-klausulen får vi alle mulige kombinasjoner av radene fra hver tabell
- Dette kalles kryssproduct eller Kartesisk produkt
- ◆ Altså, det som var × i relasjonsalgebraen

Kryssprodukt av to tabeller

Med to tabeller:

	T1	
C1	C2	C3
x1	x2	x3
y1	у2	уЗ



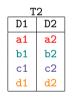


SELECT * FROM T1, T2				
C1	C2	СЗ	D1	D2
x1	x2	x3	a1	a1
x1	x2	x3	b1	b2
x1	x2	x3	c1	c2
x1	x2	x3	d1	d2
y1	у2	уЗ	a1	a2
y1	у2	уЗ	b1	b2
y1	у2	уЗ	c1	c2
у1	у2	уЗ	d1	d2

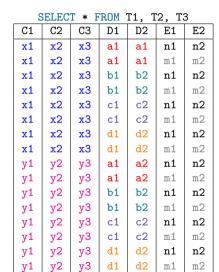
Kryssproduktet av tre tabeller

Med tre tabeller:

	T1	
C1	C2	C3
x1	x2	x3
y1	у2	уЗ







Hvorfor er dette nyttig?

- Kryssproduktet lar oss relatere en hvilken som helst verdi i en kolonne i en tabell til en hvilken som helst verdi i en kolonne i en annen tabell
- Ved å bruke WHERE -og SELECT-klausulene kan vi velge ut hva vi ønsker fra denne tabellen av alle mulige kombinasjoner

Eksempel spørring med flere tabeller

Hvilken kunde har kjøpt hvilket produkt?

```
SELECT ProductName, Customer
FROM products, orders
WHERE ProductID = OrderedProduct
```

Resultat

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

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

Joins

- Spørringer over flere tabeller kalles joins,
- Mange måter å relatere tabeller på, altså mange mulige joins, f.eks.
 - equi-join
 - theta-join
 - inner join
 - self join
 - anti join
 - semi join
 - outer join
 - natural join
 - cross join
- De er alle bare forskjellige måter å kombinere informasjon fra to eller flere tabeller
- Oftest (men ikke alltid) interesert i å "joine" på nøkler

Navn på joins

◆ Cross join mellom t1 og t2

```
SELECT * FROM t1, t2
```

◆ Equi-join mellom t1 og t2

```
SELECT * FROM t1, t2
WHERE t1.a = t2.b
```

◆ Theta-join mellom t1 og t2

```
SELECT * FROM t1, t2
WHERE <theta>(t1.a,t2.b)
```

hvor <theta> er en eller annen relasjon (f.eks. <, =, !=, LIKE) eller mer komplisert uttrykk

- ◆ Equi-join er en spesiell type Theta-join
- Alle disse formene for join (og et par til vi skal se etterpå) kalles indre joins (eng.: inner joins)

Problemer med spørring over flere tabeller

Hvilken kunde har kjøpt hvilket produkt?

```
SELECT ProductName, Customer
FROM products, orders
WHERE ProductID = ProductID -- ERROR!
```

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

Like kolonnenavn

- Når vi har flere tabeller i samme spørring kan vi få flere kolonner med likt navn
- For å fikse dette kan vi bruke tabellnavnet som prefiks
- ◆ F.eks. products.ProductID og orders.OrderID

Hvilken kunde har kjøpt hvilket produkt?

```
SELECT ProductName, Customer
FROM products, orders
WHERE products.ProductID = orders.ProductID
```

Navngi tabeller

- Det er ofte nyttig å kunne gi en tabell et nytt navn
- ◆ F.eks. dersom tabellnavnet er langt og gjentas ofte i ₩HERE-klausulen
- Eller dersom vi ønsker å gjøre en self-join (mer om dette om litt)
- ◆ Tabeller kan navngis med AS-nøkkelordet

Eksempel: Navngi tabeller

Finn produktnavnet og prisen til hver bestilling (2155 rader)

```
SELECT p.product_name, o.unit_price
FROM products AS p, order_details AS o
WHERE p.product_id = o.product_id;
```

Kan også droppe AS-nøkkelordet, og f.eks. kun skrive

```
FROM products p, order_details o
```

Eksempeler på joins: Northwind-databasen

Finn alle unike par av (fulle) navn på kunde og ansatte som har inngått en handel med last (eng.: freight) over 500kg(13 rader)

Relasjonell algebra og SQL

- SQL-spørringene med joins kan også oversettes til relasjonsalgebra
- For eksempel kan de enkle SQL-spørringene vi nå har sett oversettes slik:

Egen notasjon for joins

- SQL har en egen notasjon for joins
- ◆ For den typen joins vi har gjort hittil har man INNER JOIN -og ON-nøkkelordene
- Fremfor å skrive:

kan man skrive

- De to spørringene er ekvivalente
- Øverste kalles implisitt join, nederste kalles eksplisitt join
- Skal senere se at enkelte joins ikke kan skrives på den øverste formen
- ◆ Den nederste formen gjør det lettere å se hvordan tabellene er "joinet"

Flere join-eksempler (Northwind-DB)

Finn ut hvilke drikkevarer som er kjøpt og av hvem [404 rader]

Self-joins

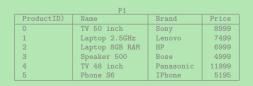
- Av og til ønsker man å kombinere informasjon fra rader i samme tabell
- Dette kalles en self-join
 - Dette gjøres ved å bruke den samme tabellen to eller flere ganger i FROM-klausulen
- Må da gi dem forskjellige navn

Self-join-eksempel

Finn navn og pris på alle produkter som er dyrere enn produktet Laptop 2.5GHz?

```
SELECT P2.Name, P2.Price
FROM Product AS P1, Product AS P2
WHERE P1.Name = 'Laptop 2.5GHz' AND P1.Price < P2.Price</pre>
```

Resultat



P2			
ProductID	Name	Brand	Price
0	TV 50 inch	Sony	8999
1	Laptop 2.5GHz	Lenovo	7499
2	Laptop 8GB RAM	HP	6999
3	Speaker 500	Bose	4999
4	TV 48 inch	Panasonic	11999
5	Phone S6	IPhone	5195

Self-join-eksempel

Finn navn og pris på alle produkter som er dyrere enn produktet Laptop 2.5GHz?

```
SELECT P2.Name, P2.Price
FROM Product AS P1, Product AS P2
WHERE P1.Name = 'Laptop 2.5GHz' AND P1.Price < P2.Price
```

Resultat

P1.ProductID	P1.Name	P1.Brand	P1.Price	P2.ProductID	P2.Name	P2.Brand	P2.Price
A							
0	TV 50 inch	Sony	8999	5	Phone S6	IPhone	5195
1	Laptop 2.5GHz	Lenovo	7499	0	TV 50 inch	Sony	8999
1	Laptop 2.5GHz	Lenovo	7499	1	Laptop 2.5GHz	Lenovo	7499
1	Laptop 2.5GHz	Lenovo	7499	2	Laptop 8GB RAM	HP	6999
1	Laptop 2.5GHz	Lenovo	7499	3	Speaker 500	Bose	4999
1	Laptop 2.5GHz	Lenovo	7499	4	TV 48 inch	Panasonic	11999
1	Laptop 2.5GHz	Lenovo	7499	5	Phone S6	IPhone	5195
2	Laptop 8GB RAM	HP	6999	0	TV 50 inch	Sony	8999
4							

Naturlig Join

- Vi joiner ofte på de kolonnene som har likt navn
- ◆ F.eks. categories.category_id med products.category_id
- Dette kan gjøres enklere med naturlig join
- Naturlig join joiner (med likhet) automatisk på alle kolonner med likt navn
- I tillegg projiserer den vekk de dupliserte kolonnene
- Trenger derfor aldri gi tabellene navn (i resultatet av en naturlig join vil det aldri finnes kolonner med likt navn)
- Merk: Må være sikker på at vi ønsker å joine på ALLE kolonnene med likt navn!

Naturlig Join: Eksempel

Finn navnet på alle drikkevarer [12 rader]

```
SELECT product_name
  FROM categories NATURAL JOIN products
WHERE category_name = 'Beverages';
```

Enkle SELECT-spørringer i et nøtteskall

- FROM-klausulen sier hvilke tabell(er) som skal brukes for å besvare spørringen og joiner dem sammen
- ◆ WHERE-klausulen velger ut hvilke rader som skal være med i svaret
 - Kolonnenavn brukes som variable som instansieres med radenes verdier
 - ◆ Kan sammenlikne kolonner og verdier med f.eks. =, !=, <, <=, LIKE
 - ◆ Bruker AND, OR og NOT på uttrykk
 - ◆ Evaluerer til enten TRUE, FALSE eller NULL for hver rad
 - Kun de som evaluerer til TRUE blir med i svaret
- SELECT-klausulen velger hvilke verdier/kolonner som skal være med i svaret
 - Kan også endre rekkefølgen på kolonner, bruke dem i uttrykk, osv.
 - ◆ Bruk * for å velge alle kolonnene
- SQL bryr seg ikke om mellomrom og linjeskift, eller store og små bokstaver

Mye mer kan gjøres i hver klausul og det finnes flere klausuler, mer om dette senere i kurset!

Takk for nå!

Neste video vil se på nestede spørringer.