INF1300— Relasjonsalgebra

Et matematisk fundament for å forstå SQL-setninger

Innhold

- Relasjonsalgebraen
- Operatorene i relasjonsalgebraen
- Relasjonsalgebratolkning av select-setningen
- Kostbare operasjoner i SQL

Relasjonsalgebraen

- definerer en mengde av operasjoner på relasjoner
- utgjør det matematiske grunnlaget for prosessering av SQL-spørringer mot relasjonsdatabaser
- er et **prosedyralt** spørrespråk: Vi sier *hvordan svaret skal beregnes*. Til sammenlikning er SQL et **deklarativt** språk, det sier *hva svaret skal oppfylle*.

Bag/multiset vs. set

- Kommersielle DBMSer benytter *bag* (*multiset*) og ikke *set* som grunntype for å realisere relasjoner/tabeller
- SQL beregner bager (med noen få unntak, som bl.a. **select distinct**)
  - Hvis D er en *mengde*/*set*, så forekommer hvert element i D maksimalt én gang
    Rekkefølgen på elementene er likegyldig
  - Hvis D er en *bag*/*multiset*, så kan et element forekomme mer enn én gang i D
    Rekkefølgen på elementene er likegyldig
- Hvorfor *bag* og ikke *set*
  - *Bag* gir mer effektive beregninger for noen av operasjonene
  - Ved aggregering trenger vi bagfunksjonalitet
  - Men, *bag* er mer plasskrevende enn *set*

Operatorene i relasjonsalgebraen

Merk: Vi gjennomgår bare *noen* av operatorene i relasjonsalgebraen:

- seleksjon (σ)
- projeksjon (π)
- kartesisk produkt ($\times$)
- join ($\bowtie$)
- union ($\cup$)
- snitt ($\cap$)
- differanse ($\setminus$)

Seleksjon

- $\sigma_C(R)$ er relasjonen som fås fra R ved å velge ut de tuplene i R som tilfredsstiller betingelsen C
- C er et boolsk uttrykk bygget opp fra attributter i R

Seleksjon

Eksempel på seleksjon:

Ansatt

navn	**tlf**	**adr**
Ali	22465819	Lia 3
Lise	37485960	Bakken 1
Geir	25364758	Viken 2

$\sigma_{\text{tlf}>24000000}(\textbf{Ansatt})$

navn	**tlf**	**adr**
Lise	37485960	Bakken 1
Geir	25364758	Viken 2

Projeksjon

- $\pi_L(R)$ er relasjonen som fås fra R ved å velge ut kolonnene angitt av L
- L er en liste av attributter i R

Projeksjon

Eksempel på projeksjon:

Ansatt

navn	**tlf**	**adr**
Ali	22465819	Lia 3
Lise	37485960	Bakken 1
Geir	25364758	Viken 2

$\pi_{\text{navn,adr}}(\textbf{Ansatt})$

navn	**adr**
Ali	Lia 3
Lise	Bakken 1
Geir	Viken 2

Kartesisk produkt

- $R \times S$ er relasjonen som fås fra R og S ved å danne alle mulige sammensetninger av et tuppel fra R og et tuppel fra S
- Eventuell navnelikhet mellom attributter i R og S løses ved å kvalifisere navnene med opprinnelsesrelasjonen: $R.A$, $S.A$

Kartesisk produkt

Spisested

snavn	mkat
A	kosher
A	vegetabilsk
B	uten melk
B	hallal
B	glutenfri
B	kosher
C	glutenfri
C	hallal
C	kosher
D	vegetabilsk

Menykrav

navn	mkat
Ali	hallal
Liv	kosher
Lise	kosher
Geir	glutenfri

Spisested $\times$ Menykrav

snavn	Spisested.mkat	navn	Menykrav.mkat
A	kosher	Ali	hallal
A	vegetabilsk	Ali	hallal
B	uten melk	Ali	hallal
B	hallal	Ali	hallal
B	glutenfri	Ali	hallal
B	kosher	Ali	hallal
C	glutenfri	Ali	hallal
C	hallal	Ali	hallal
C	kosher	Ali	hallal
D	vegetabilsk	Ali	hallal
A	kosher	Liv	kosher
A	vegetabilsk	Liv	kosher
B	uten melk	Liv	kosher
B	hallal	Liv	kosher
B	glutenfri	Liv	kosher
B	kosher	Liv	kosher
C	glutenfri	Liv	kosher
C	hallal	Liv	kosher
C	kosher	Liv	kosher
D	vegetabilsk	Liv	kosher
A	kosher	Lise	kosher
A	vegetabilsk	Lise	kosher
B	uten melk	Lise	kosher
B	hallal	Lise	kosher
B	glutenfri	Lise	kosher
B	kosher	Lise	kosher
C	glutenfri	Lise	kosher
C	hallal	Lise	kosher
C	kosher	Lise	kosher
D	vegetabilsk	Lise	kosher
A	kosher	Geir	glutenfri
A	vegetabilsk	Geir	glutenfri
B	uten melk	Geir	glutenfri
B	hallal	Geir	glutenfri
B	glutenfri	Geir	glutenfri
B	kosher	Geir	glutenfri
C	glutenfri	Geir	glutenfri
C	hallal	Geir	glutenfri
C	kosher	Geir	glutenfri
D	vegetabilsk	Geir	glutenfri

Generell join (også kalt theta-join)

- Resultatet av $R \underset{C}{\bowtie} S$ er en sammenskjøting av tupler fra R og S i henhold til joinbetingelsen C

- Intuitivt:
  - 1. Beregn $R \times S$
  - 2. Velg ut de tuplene som tilfredsstiller C

Join

Bistro

bn	mkat
A	kosher
A	vegetabilsk
B	uten melk
B	hallal
B	glutenfri
B	kosher
C	glutenfri
C	hallal
C	kosher
D	vegetabilsk

Krav

navn	mkat
Ali	hallal
Liv	kosher
Lise	kosher
Geir	glutenfri

Bistro $\bowtie_{\text{Bistro.mkat=Krav.mkat}}$ **Krav**

bn	Bistro.mkat	navn	Krav.mkat
B	hallal	Ali	hallal
C	hallal	Ali	hallal
A	kosher	Liv	kosher
B	kosher	Liv	kosher
C	kosher	Liv	kosher
A	kosher	Lise	kosher
B	kosher	Lise	kosher
C	kosher	Lise	kosher
B	glutenfri	Geir	glutenfri
C	glutenfri	Geir	glutenfri

Naturlig join

- $R \bowtie S$ er relasjonen som fås fra R og S ved å danne alle mulige sammensmeltinger av et tuppel fra R med et fra S der tuplene skal stemme overens i samtlige attributter med sammenfallende navn
- Fellesattributtene til R og S tas bare med én gang i svarrelasjonen

Naturlig join

Bistro

bn	mkat
A	kosher
A	vegetabilsk
B	uten melk
B	hallal
B	glutenfri
B	kosher
C	glutenfri
C	hallal
C	kosher
D	vegetabilsk

Krav

navn	mkat
Ali	hallal
Liv	kosher
Lise	kosher
Geir	glutenfri

Bistro $\bowtie$ Krav

bn	mkat	navn
B	hallal	Ali
C	hallal	Ali
A	kosher	Liv
B	kosher	Liv
C	kosher	Liv
A	kosher	Lise
B	kosher	Lise
C	kosher	Lise
B	glutenfri	Geir
C	glutenfri	Geir

Union, snitt og differanse

- Union: $R \cup S$
- Snitt: $R \cap S$
- Differanse: $R \setminus S$

Unionkompabilitet: For alle disse operatorene gjelder at R og S må ha like mange attributter og attributtene må parvis ha identiske domener. *Eks:*

Ansatt(navn,tlf,adr,id) $\cup$ Medlem(id,navn,adr,tlf)

Før operasjonen utføres, ordnes S slik at attributtene kommer i samme rekkefølge som i R:

Ansatt(navn,tlf,adr,id) $\cup$ Medlem(navn,tlf,adr,id).

Union

- Hvis t er et tuppel som forekommer n ganger i R og m ganger i S, så forekommer t $n + m$ ganger i $R \cup S$.

Union

- Eksempel på union av relasjoner:

Ansatt

navn	tlf	adr
Ali	22465819	Lia 3
Lise	37485960	Bakken 1
Geir	25364758	Viken 2

Medlem

navn	adr	tlf
Lise	Bakken 1	37485960
Liv	Lia 3	19203142
Jan	Heia 7	26374859
Geir	Viken 2	25364758

Ansatt $\cup$ Medlem

navn	tlf	adr
Ali	22465819	Lia 3
Lise	37485960	Bakken 1
Lise	37485960	Bakken 1
Liv	19203142	Lia 3
Jan	26374859	Heia 7
Geir	25364758	Viken 2
Geir	25364758	Viken 2

Eksempel på union i SQL

Prosjektplan(pnr, anr, timer)

```
create view Innsats as
            select   anr, sum(timer) as sumtimer
            from     Prosjektplan
            group by anr;


create view Bonus(anr, bonusiNOK) as
            (select  anr, 3000
             from     Innsats
             where    sumtimer >= 500)
       union all
            (select  anr, 1500
             from    Innsats
             where   sumtimer >= 300 and sumtimer < 500);
```

Snitt

- Hvis t er et tuppel som forekommer n ganger i R og m ganger i S, så forekommer t $\min(n, m)$ ganger i $R \cap S$

Snitt

Eksempel på snitt mellom relasjoner:

Ansatt

navn	tlf	adr
Ali	22465819	Lia 3
Lise	37485960	Bakken 1
Lise	37485960	Bakken 1
Geir	25364758	Viken 2
Geir	25364758	Viken 2

Medlem

navn	adr	tlf
Lise	Bakken 1	37485960
Lise	Bakken 1	37485960
Liv	Lia 3	19203142
Jan	Heia 7	26374859
Geir	Viken 2	25364758

Ansatt $\cap$ Medlem

navn	tlf	adr
Lise	37485960	Bakken 1
Lise	37485960	Bakken 1
Geir	25364758	Viken 2

Eksempel på snitt i SQL

```
-- antall felles filmid-er (snitt)
-- mellom film og filmparticipation

select count(snitt.*)
from (
        (select distinct filmid from filmparticipation)
             intersect all
        (select filmid from film)
      )
        as snitt
;
```

```
 count
--------
 479529
(1 row)
```

Differanse

- Hvis t er et tuppel som forekommer n ganger i R og m ganger i S, så forekommer t $\max(0, n - m)$ ganger i $R \setminus S$

Differanse

Eksempel på differanse mellom relasjoner:

Ansatt

navn	tlf	adr
Ali	22465819	Lia 3
Lise	37485960	Bakken 1
Geir	25364758	Viken 2
Geir	25364758	Viken 2

Medlem

navn	adr	tlf
Lise	Bakken 1	37485960
Liv	Lia 3	19203142
Jan	Heia 7	26374859
Geir	Viken 2	25364758

Ansatt $\setminus$ Medlem

navn	tlf	adr
Ali	22465819	Lia 3
Geir	Viken 2	25364758

Eksempel på differanse i SQL

```
-- antall filmer i filmparticipation som
-- ikke er i film

select count(diff.*)
from (
        (select distinct filmid from filmparticipation)
             except all
        (select filmid from film)
      )
        as diff
;
```

```
 count
--------
 455223
(1 row)
```

Relasjonsalgebratolkning av select-setningen—1

1. Ta det kartesiske produktet av relasjonene i **from**
2. Selekter ifølge betingelsene i **where**
3. Projiser ifølge betingelsene i **select**
4. Fjern flerforekomster hvis **select distinct**
5. Sorter i henhold til **order by**

Eksempel

- Skjema:
  Spisested(snavn, matkat)
  Menykrav(navn, matkat)

```
select distinct snavn
from Spisested, Menykrav
where
    (navn='Ali' or navn='Liv')
    and Spisested.matkat =
    Menykrav.matkat
order by snavn;
```

- Query:
- $\tau_{\text{snavn}}(\delta(\pi_{\text{snavn}}(\sigma_c(\text{Spisested} \times \text{Menykrav}))))$
- Betingelsen c: (navn = 'Ali' or navn = 'Liv') and Spisested.matkat = Menykrav.matkat
- $\delta(R)$ fjerner flerforekomster av tupler fra bagrelasjonen R
- $\tau_{A_1,A_2,\ldots,A_n}(R)$ sorterer tuplene i R etter verdiene i A_1, for like verdier i A_1 sorteres tuplene etter verdiene i A_2, osv.

Relasjonsalgebratolkning av select-setningen—2

- Hvis vi ser nøyere på vanlige select-setninger, er det ofte to vesensforskjellige typer setningsledd i where-klausulen:
  - *Seleksjon* av tupler i en relasjon
  - *Join* av tupler på tvers av relasjoner
- Selv om den forenklede tolkningen av select-setningen er riktig matematisk sett, vil databasesystemet velge en tolkning som er mer effektivt beregnbar

Relasjonsalgebratolkning av select-setningen—2 (forts.)

Derfor vil databasesystemet oversette til et uttrykk som likner det vi får ved å gjøre følgende:

1. Selekter ifølge *seleksjonsbetingelsene* i **where** (gir færre tupler i relasjonene før join)
2. Join relasjonene i **from** i henhold til *joinbetingelsene* i **where**
3. Projiser ifølge betingelsene i **select**
4. Fjern flerforekomster hvis **select distinct**
5. Sorter i henhold til **order by**

Eksempel

- Skjema:
  Spisested(snavn, matkat)
  Menykrav(navn, matkat)

```
select distinct snavn
from Spisested, Menykrav
where
    (navn='Ali' or navn='Liv')
    and Spisested.matkat =
    Menykrav.matkat
order by snavn;
```

- Query:
- $\tau_{\text{snavn}}(\delta(\pi_{\text{snavn}}(\text{Spisested} \underset{\text{jb}}{\bowtie} \sigma_{\text{sb}}(\text{Menykrav}))))$
- Seleksjonsbetingelsen sb: navn = 'Ali' or navn = 'Liv'
  Joinbetingelsen jb: Spisested.matkat = Menykrav.matkat
- I dette tilfellet kunne joinbetingelsen alternativt vært oversatt til naturlig join

Kostbare operasjoner i SQL

- **distinct**:
  - Sortering er generelt kostbart
  - Bør brukes med forsiktighet
- **union**, **intersect**, **except**:
  - Når vi benytter **union**, **intersect**, **except**, skal resultatet være en mengde – da må flerforekomster fjernes
  - Vurder å bruke **union all**, **intersect all**, **except all** som er bag-variantene