Gagnasafnsfræði

Páll Melsted

12. sept

Venslaalgebra

Við getum lýst öllum aðgerðum í SQL sem við höfum séð sem virkjum og föllum í venslaalgebru. Grunneiningin sem við vinnum með eru vensl, þ.e. mengi af n-dum.

UNION, INTERSECT og EXCEPT má lýsa með mengjavirkjunum $\cup$, $\cap$, $-$. Athugið samt að ólíkt venjulegum mengjum þá þurfum við að tryggja að í $R \cup S$ hafi R og S sömu eigindi og í sömu röð.

Dálkaval

- Ef R eru vensl má búa til ný vensl með því að velja suma dálka úr R, mögulega í annarri röð.
- Virkinn $\pi_{A_1,A_2,...,A_n}(R)$ velur dálka $A_1, A_2, \ldots, A_n$ úr R
- T.d. gefur $\pi_{title,year}(Movie)$

title	year
Pretty Woman	1990
The Man Who Wasn't There	2001
Logan's run	1976
Star Wars	1977
Empire Strikes Back	1980
Star Trek	1979
Star Trek: Nemesis	2002
Terms of Endearment	1983
The Usual Suspects	1995
Gone With the Wind	1938

Dálkaval (projection)

- Hins vegar gæfi $\pi_{length}(Movie)$

```
length
------
119
116
124
111
132
106
238
```

- Ástæðan er sú að algebran vinnur bara með mengi, endurtekin gildi detta út, ólíkt því sem gerist í SQL.

Val (selection)

- Valvirkinn tekur inn vensl og skilar nýjum venslum $\sigma_C(R)$, þar sem C er eitthvað skilyrði. Útkoman er með sömu dálkanöfn og R og aðeins þær n-dir sem uppfylla skilyrðið C.

- Útkoman úr $\sigma_{length \geq 100}(Movie)$ er

title	year	length	inColor	studioName	producerC
Pretty Woman	1990	119	1	Disney	999
The Man Who Wasn't There	2001	116	0	USA Entert	777
Star Wars	1977	124	1	Fox	555
Empire Strikes Back	1980	111	1	Fox	555
Star Trek	1979	132	1	Paramount	444
Star Trek: Nemesis	2002	116	1	Paramount	321
Terms of Endearment	1983	132	1	MGM	123
The Usual Suspects	1995	106	1	MGM	999
Gone With the Wind	1938	238	1	MGM	123

Tenging við SQL

- Einfaldar fyrirspurnir á forminu

```
SELECT A1,A2,...,AN
FROM R
WHERE C;
```

- Má skrifa á algebruformi sem

$$\pi_{A_1,\ldots,A_N}(\sigma_C(R))$$

- Hvað með mörg vensl?

Mörg vensl

- Til að vinna með meira en eina töflu í einu skilgreinum við margfeldi vensla

- $R \times S$ Útkoman er allar samsetningar af n-dum í R við n-dir í S.

- Dálkarnir í $R \times S$ halda nöfnunum sínum. Ef R og S hafa dálka með sama nafn, A, heita útkomurnar R.A og S.A.

Mörg vensl

- SQL fyrirspurnin

```
SELECT A1,A2,...,AN
FROM R1,R2,...,RN
WHERE C;
```

- Verður þá

$$\pi_{A_1,\ldots,A_N}(\sigma_C(R_1 \times R_2 \times \ldots \times R_N))$$

Endurnefndir dálkar

- Til að endurnefna dálka notum við ρ virkjann

$$\rho_{S(A_1,\ldots,A_N)}(R)$$

- Býr til ný vensl S með sömu gögn og R en dálkarnir heita $A_1, \ldots, A_N$ (miðað við að R hafi N dálka). Ef að við viljum bara endurnefna venslin þarf ekki að telja dálkana upp.

Join Dæmi

- SQL fyrirspurnina

```sql
SELECT name AS pname
FROM Movie, MovieExec
WHERE title = 'Star Wars' AND producerC = cert;
```

- má skrifa sem

$$\rho_{R(pname)}(\pi_{name}(\sigma_{title='StarWars' \ AND \ producerC=cert}(Movie \times MovieExec)))$$

Natural Join

- Í tengingunni sem við sáum þurftum við tengja á producerC og cert og báðir dálkar enduðu í niðurstöðunni. Oftast hafa báðir dálkar sama nafn og við viljum tengja þegar báðir eru eins

R:

A	B
1	2
3	4

S:

B	C	D
2	5	6
4	7	8
9	10	11

- $\sigma_{R.B=S.B}(R \times S)$

A	R.B	S.B	C	D
1	2	2	5	6
3	4	4	7	8

Natural Join

- Eðlileg tenging (natural join),$R \bowtie S$ tengir á milli R og S með því að velja á sameiginlegum dálkum og sleppa endurteknum gildum.

$R \bowtie S$

A	B	C	D
1	2	5	6
3	4	7	8

Theta Join

- Fyrra formið á tengingu sem við sáum, er almennara þar sem C getur verið hvaða skilyrði sem er. Stundum kallað θ-join, táknað $R \bowtie_\theta S$

- Natural join má skrifa sem

$$R \bowtie S = \pi_L(\sigma_C(R \times S))$$

- þar sem C er skilyrðið sem velur dálkana eins og L eru dálkarnir án endurtekninga.

Nauðsynlegir virkjar

- Til að vinna með venslaalgebru þarf eftirfarandi 6 virkja: $\cup, -, \sigma, \pi, \times, \rho$
- Alla aðra virkja má skrifa út frá þessum
- SQL gagnagrunnar geta notað venslaalgebru til að einfalda fyrirspurnir út frá algebrureglum
- T.d. gildir

$$\sigma_C(R \times S) = \sigma_C(R) \times S$$

- ef engir dálkar í S koma fyrir í C og

$$\sigma_C(R \times S) = \sigma_C(R) \times \sigma_C(S)$$

- ef dálkarnir í C koma fyrir í bæði R og S.

Hlutfyrirspurnir

- Í SQL skila allar fyrirspurnir nýjum venslum. Þessi vensl má svo nota í aðrar SQL fyrirspurnir og koma þá fyrir sem hlutfyrirspurn í stærra samhengi.

- Þrjár meginleiðir til að nota hlutfyrirspurnir

1. Fyrirspurn skilar einu gildi, þetta gildi er hægt að bera saman við önnur í WHERE hluta

2. Fyrirspurn skilar venslum, hægt er að nota þessi vensl eins og önnur í WHERE hluta

3. Fyrirspurn getur komið fyrir í FROM hluta með nafni til að nota fyrir niðurstöðu.

Hlutfyrirspurnir með gildi

```
SELECT name
FROM MovieExec
WHERE cert =
     (SELECT producerC
     FROM Movies
     WHERE title = 'Star Wars'
     );
```

- Hér skilar hlutfyrirspurnin gildinu 12345. Niðurstaðan er því eins og við hefðum keyrt

```
SELECT name
FROM MovieExec
WHERE cert = 12345
```

Hlutfyrirspurnir með venslum

- Niðurstaðan úr fyrirspurn er yfirleitt vensl. Þegar venslin eru með einn dálk má nota eftirfarandi virkja

1. EXISTS IN R - er niðurstaðan ekki tóm?

2. s IN R- er gildið s í R?

3. s > ALL R- er s stærra en **öll** gildin í R

4. S > ANY R - er s stærra en **eitthvert** gildi í R.

Hlutfyrirspurnir með n-dum

- Það er hægt að nota = og <> með ANY og ALL

- þegar vensl eru með meira en einn dálk.

```
SELECT name
FROM MovieExec
WHERE cert IN
    (SELECT producerC
    FROM Movie
    WHERE (title,year) IN
        (select movieTitle, movieYear
        FROM StarsIn
        WHERE starName = 'Harrison Ford')
    );
```

- úbbs! SQLite leyfir ekki IN með mörgum dálkum.

Hægt að einfalda með venjulegu Join

```
SELECT name
FROM MovieExec, Movie, StarsIn
WHERE cert = producerC AND
    title = movieTitle AND
    year = movieYear AND
    starName = 'Harrison Ford';
```

- Reyndar kemur sama nafnið fyrir tvisvar.

Þurfum við hlutfyrirspurnir?

- Já (tekið úr cloudcoder kerfinu)

```
select
    uu . *, best . *
from
    cc_users as uu
        left join
    (select
        u.id as the_user_id, e . *, sr . *
    from
        cc_users as u, cc_events as e, cc_submission_receipts as sr,
                (select
        i_u.id as user_id,
            best.max_tests_passed as max_tests_passed,
            MIN(i_e.timestamp) as timestamp
    from
        cc_users as i_u, cc_events as i_e, cc_submission_receipts as i_sr, (select
        ii_u.id as user_id,
            MAX(ii_sr.num_tests_passed) as max_tests_passed
    from
        cc_users as ii_u, cc_events as ii_e, cc_submission_receipts as ii_sr
    where
        ii_u.id = ii_e.user_id
            and ii_e.id = ii_sr.event_id
            and ii_e.problem_id = %d
        and timestamp < %d
    group by ii_u.id) as best
    where
        i_u.id = i_e.user_id
            and i_e.id = i_sr.event_id
            and i_e.problem_id = %d
            and i_u.id = best.user_id
            and i_sr.num_tests_passed = best.max_tests_passed
    group by i_u.id , best.max_tests_passed) as earliest_and_best
    where
        u.id = e.user_id and e.id = sr.event_id
            and e.problem_id = %d
            and u.id = earliest_and_best.user_id
```

Háðar hlutfyrirspurnir

- Í fyrri dæmum hefur niðurstaðan úr innri fyrirspurn verið föst. Við getum líka látið niðurstöðuna vera háða ytri fyrirspurn.

- Þá látum við innri fyrirspurn vísa í dálk sem er ekki til í því samhengi, t.d.

```
(SELECT year
 FROM Movie
 WHERE title = Old.title
 );
```

- en myndi skila niðurstöðu ef við vissum gildið á Old.title

Háðar hlutfyrirspurnir

- Í ytri fyrirspurn verður innri fyrirspurnin keyrð þegar við vitum gildið og getur verið ólíkt á milli n-da í venslunum.

```
SELECT title
FROM Movie AS Old
WHERE year <= ANY
    (SELECT year
     FROM Movie
     WHERE title = Old.title
    );
```

- Þegar við notum háðar hlutfyrirspurnir þá verðum við að huga að gildissviði nafna (scope).

Hlutfyrirspurnir í FROM

- Hlutfyrirspurnir geta líka komið fyrir í FROM hluta, niðurstaðan virkar eins og tafla með niðurstöðunni hefði verið notuð, en það þarf að nefna töfluna.

```
SELECT name
FROM MovieExec,
  (SELECT producerC
   FROM Movie, StarsIn
   WHERE title = movieTitle AND
   year = MovieYear AND
   starName = 'Harrison Ford'
   ) AS Prod
WHERE cert = producerC;
```

Tengingar (Join) í SQL

- Við getum notað tengingar í SQL beint í stað þess að telja upp venslin

```
SELECT *
FROM Movie CROSS JOIN StarsIn;
```

- Skilar sama eins og FROM Movie, StarsIn

Join

- Í stað þessa að velja úr mörgum venslum og tengja saman í WHERE getum viðfært tenginuna nær töflunni.

```
SELECT *
From Movie JOIN StarsIn ON title=movieTitle AND year=movieYear
```

- Þetta er í raun θ-join, $\bowtie_C$ virkinn.

Natural Join

- SQL er líka með eðlilega tengingu (natural join) með NATURAL JOIN virkjanum.

```
SELECT *
FROM MovieStar NATURAL JOIN MovieExec;
```

- Það er ekki hægt að nota ON með NATURAL JOIN

Breytingar á gögnum

- Þrjú tilfelli um það hvernig gögnum er breytt

1. Nýjar raðir settar inn í töflu
2. Röðum eytt úr töflu
3. Gildum (hluta úr röð) breytt í töflu

Innsetning

- Innsetning inn í töflu er framkvæmd með

```
INSERT INTO R(A1,...,AN) VALUES (v1,..,vN)
```

- Ef venslin hafa fleiri dálka sem við teljum ekki upp þá fá þau sjálfgefin gildi.

```
INSERT INTO StarsIn (movieTitle, movieYear, starName)
VALUES ('The Terminator', 1984, 'Arnold Schwarzenegger');
```

- Ef við teljum ekki upp dálka þá **verða** gildin að vera í sömu röð og í skilgreiningunni á töflunni.

Flóknari innsetning

- Við getum líka sett gögn inn í töflu sem niðurstöðu úr fyrirspurn.

```
INSERT INTO Studio (name)
    SELECT DISTINCT studioName
    FROM Movie
    WHERE studioName NOT IN
        (SELECT name
         FROM Studio);
```

Eyðing gagna

- Til að eyða röð úr töflu R notum við

```
DELETE FROM R WHERE <...>;
```

- WHERE skilyrðið er eins og í SELECT. Yfirleitt viljum við aðeins eyða einni röð og þá notum við lykla í töflu til að hjálpa til við það.

```
DELETE FROM Movie WHERE title = 'Gone With the Wind';
```

Uppfærsla

- Til að breyta gildum á röðum sem eru nú þegar í gagnagrunni notum við UPDATE

```
UPDATE R SET <...> WHERE <...>;
```

- WHERE hlutinn velur þær raðir sem á að uppfæra, yfirleitt er hægt að nota lykla til að velja einstaka raðir. Í sqlite er sér dálkur rowid sem er alltaf til og hægt að nota sem lykil, t.d.

```
SELECT rowid,*
FROM Movie
WHERE length IS NULL;
```

```
UPDATE Movie
SET length=0
WHERE rowid=3
```

- Reyndar væri einfaldara að gera

```
UPDATE Movie SET length=0 WHERE length IS NULL;
```

Fyrir næstu viku

- Lesa kafla 2.5, 6.4 og 6.5

- SELECT fyrirspurnirnar verða flóknari eftir því sem á líður og mikilvægt að kunna þær vel

- Þær gilda 35% af lokaprófinu!