IN2090

Aggregering og sortering

Sortering

- Sorterer radene i resultatet fra en SELECT-spørring

```
>> ORDER BY <kolonner>
```

- Sorteringen er gjort i henhold til typens naturlige ordning
  - Tall: verdi
  - Tekst: alfabetisk
  - Tidspunkter: kronologisk
- ORDER BY-klausulen kommer alltid etter WHERE-klausulen

Sortere på flere kolonner og reversering

- Standard-ordningen er fra minst til størst, legger man til DESC reverserer vi ordningen, til størst til minst.
- Med flere kolonner i ORDER BY vil radene ordnes først ihht. Den første kolonnen, så ihht. den andre kolonnen for de med like verdier i den første, osv.

LIMIT og OFFSET

- For å begrense antall rader kan vi bruke LIMIT
- LIMIT-klausulen kommer alltid til sist
- OFFSET brukes til å hoppe over rader
- Dette er nyttig dersom man ønsker å presentere resultater i grupper

GROUP BY

- GROUP BY tar en liste med kolonner, og grupperer dem i henhold til likhet på verdiene i disse kolonnene
- Vi kan så bruke aggregeringsfunksjoner på hver gruppe i SELECT-klausulen
- Vi kan da også ha de grupperende kolonnene sammen med aggregatet i SELECT-klausulen
- Kun de grupperte kolonnene gir mening å ha utenfor et aggregat i SELECT
- Vi kan også gruppere på flere kolonner
- Da vil hver gruppe bestå av de radene med like verdier på alle kolonnene vi grupperer på

EKSEMPEL

- Hvilke verdier forekommer i attributtet filmtype i relasjonen filmitem? Lag en oversikt over filmtypene og hvor mange filmer innen hver type (7).

```
SELECT filmtype, COUNT(*) ant
FROM filmitem
GROUP BY filmtype
ORDER BY ant DESC;
```

- Bruker DESC for å få den filtypen med flest filmer først.

EKSEMPEL

- Skriv ut serietittel, produksjonsår og antall episoder for de 15 eldste TV-seriene i filmdatabasen (sortert stigende etter produksjonsår).

```
SELECT s.seriesid, maintitle, firstprodyear, count(e.episodeid)
FROM series AS s
     INNER JOIN episode e ON s.seriesid=e.seriesid
GROUP BY s.seriesid, maintitle, firstprodyear
ORDER BY firstprodyear ASC
LIMIT 15;
```

HAVING

- HAVING blir altså evaluert på hver gruppe
- Fungerer som en slags WHERE for grupper

EKSEMPEL

- Mange titler har vært brukt i flere filmer. Skriv ut en oversikt over titler som har vært brukt i mer enn 30 filmer. Bak hver tittel skriv antall ganger den er brukt. Ordne linjene med hyppigst forekommende tittel først. (12 eller 26)

```
SELECT title, COUNT(*) AS ant
FROM film
GROUP BY title
HAVING COUNT(*) > 30
ORDER BY ant DESC;
```

EKSEMPEL

- Finn de “Pirates of the Caribbean”-filmene som er med i flere enn 3 genre (4)

```
SELECT title, count(*) as antall_genre
FROM film AS f JOIN filmgenre AS fg USING (filmid)
WHERE f.title LIKE 'Pirates of the Caribbean%'
GROUP BY f.filmid, title
HAVING count(*) > 3;
```

Oversikt

- Generelt ser våre SQL-spørringer nå slik ut:

```
>> WITH
>> SELECT
>> FROM
>> WHERE
>> GROUP BY
>> HAVING
>> ORDER BY [DESC]
>> LIMIT
>> OFFSET
```

- I denne rekkefølgen (LIMIT og OFFSET kan bytte plass)
- Kan selvfølgelig droppe klausuler, men må ha GROUP BY for å ha HAVING

Bruk av ulike navn

- Navnene vi lager med AS i WITH-klausulen kan brukes i alle de etterfølgende spørringene
- Navnene fra SELECT kan brukes i ORDER BY-klausulen og alle ytre spørringer
- Navnene fra FROM kan brukes i alle klausuler utenom samme FROM-klausul

EKSMEPEL

- Finn filmid, tittel og antall medregissører (parttype ’director’) (0 der han har regissert alene) for filmer som Ingmar Bergman har regissert (62).

```
WITH ingmarbergmanmovies AS (
  SELECT fp.filmid
  FROM filmparticipation AS fp
       INNER JOIN person AS p ON fp.personid = p.personid
  WHERE fp.parttype = 'director' AND
        p.firstname = 'Ingmar' AND
        p.lastname = 'Bergman'
),
ant_regissorer AS (
  SELECT fp.filmid, COUNT(*) ant
  FROM filmparticipation AS fp
  WHERE fp.filmid IN (SELECT * FROM ingmarbergmanmovies)
  AND fp.parttype = 'director'
  GROUP BY fp.filmid
)
SELECT f.filmid, f.title, (ar.ant - 1) AS ant_medregissorer
FROM film AS f INNER JOIN ant_regissorer AS ar ON f.filmid = ar.filmid;
```

Jobb med ukesoppgaver/Innlevering 3

- Innlevering 3 (Normalformer):
  innlevering3.pdf (uio.no)
- Frist for Innlevering 3 (Normalformer): 26 Oktober kl 23.59!
- Ukesoppgaver (uke 9: SQL: Aggregering og sortering)
  IN2090-ukesoppgaver: Uke 9 – Universitetet i Oslo (uio.no)
- Utsettelse på Innlevering? send mail til camilldb@uio.no