

Databaser og SQL

Modul 9 (uke 7) innhold

Boolsk algebra

- Boolsk algebra (boolsk logikk) er sentralt i SQL spørringer (i WHERE delen).
- Kan dere dette fra før, eller skal vi ta kort om det nå?
 - AND logikk
 - OR logikk
 - NOT logikk

Datatyper

- Navn og syntaks for datatyper varierer litt fra database til database.
- PostgreSQL inneholder en rekke datatyper. Blant de vanligste er:
 - `character(size)`, `text`, `integer`, `numeric(size1,size2)` og `boolean`.
 - NB: Apostrof ('fnutt') rundt tekstverdier, *ikke* rundt tall!
- Fullstendig oversikt her:
 - [Documentation -> PostgreSQL 15 | Chapter 8: Data Types](#)

Verdien NULL (repetisjon)

- **NULL** representerer en kolonneverdi som ikke er satt for denne raden i tabellen.
- **MERK**
 - NULL er ikke det samme som tallverdien 0.
 - NULL er ikke det samme som et mellomrom/space.

	Code	Name	IndepYear
▶	ABW	Aruba	NULL
	AFG	Afghanistan	1919
	AGO	Angola	1975
	AIA	Anguilla	NULL
	ALB	Albania	1912
	AND	Andorra	1278
	ANT	Netherlands Antilles	NULL
	ARE	United Arab Emirates	1971
	ARG	Argentina	1816
	ARM	Armenia	1991
	ASM	American Samoa	NULL
	ATA	Antarctica	NULL
	ATF	French Southern territories	NULL

NULL kan fort spille oss et puss (1)

```
SELECT COUNT(*) AS AntLand  
FROM country;
```

	AntLand
▶	239

```
SELECT COUNT(*) AS AntLand  
FROM country  
WHERE IndepYear > 1814  
OR IndepYear <= 1814;
```

	AntLand
▶	192

NULL kan fort spille oss et puss (2)

```
SELECT COUNT(*) AS AntLand  
FROM country  
WHERE IndepYear > 1814  
OR IndepYear <= 1814  
OR IndepYear = NULL;
```

	AntLand
▶	192

```
SELECT COUNT(*) AS AntLand  
FROM country  
WHERE IndepYear > 1814  
OR IndepYear <= 1814  
OR IndepYear IS NULL;
```

	AntLand
▶	239

SQL – SELECT queries

- Hensikten med en SELECT query er å hente data fra en eller flere tabeller.
- Resultatet av en SELECT vises som en ny tabell.
- SELECT er den mest brukte SQL kommandoen.
- Syntaks/rekkefølge:

<code>select</code>	<code>kolonne [as navn]</code>
<code>from</code>	<code>tabell</code>
<code>[where</code>	<code>betingelse]</code>
<code>[group by</code>	<code>grupperingsuttrykk]</code>
<code>[having</code>	<code>betingelse]</code>
<code>[order by</code>	<code>kolonne]</code>

Nye ord,
lærer disse
etterpå.

SELECT DISTINCT

- Et select-utvalg for et begrenset antall kolonner kan gi like rader i svaret (fordi unike kolonner for disse radene er fjernet).
- For å fjerne evt. duplikater ved select:

```
SELECT DISTINCT CountryCode  
FROM city;  
ORDER BY CountryCode ASC;
```



Nytt ord!

WHERE

- Operatorer:

<code>=</code>	lik (<i>ikke</i> v/wildcards!)
<code><></code> eller <code>!=</code>	forskjellig fra
<code><</code>	mindre enn
<code>></code>	større enn
<code><=</code>	mindre eller lik
<code>>=</code>	større eller lik

<code>like</code>	lik, godtar wildcards
<code>in</code>	i gitt utvalg
<code>between</code>	utvalg, følges av <code>and</code>
<code>—</code>	wildcard, enkelt tegn
<code>%</code>	wildcard, flere tegn
<code>is null</code>	evt. <code>is not null</code>

Wildcards,
hva er det?

- Logiske operatorer – setter sammen kriterier:

<code>and</code>	og
<code>or</code>	eller
<code>not</code>	ikke

GROUP BY og HAVING

- GROUP BY lar oss gruppere summeringsresultater til mer enn én rad.
- Summeringsresultater får vi når vi bruker funksjoner som COUNT, SUM, AVG, ...
- Ønsker vi i tillegg å fjerne rader, bruker vi ikke WHERE, men HAVING.

GROUP BY og HAVING – forts.

```
SELECT COUNT(*), MIN(SurfaceArea), MAX(IndepYear),  
AVG(LifeExpectancy), SUM(GNP)  
FROM country;
```

```
SELECT Continent, COUNT(*), MIN(SurfaceArea),  
MAX(IndepYear), AVG(LifeExpectancy), SUM(GNP)  
FROM country  
GROUP BY Continent;
```

```
SELECT Continent, COUNT(*), MIN(SurfaceArea),  
MAX(IndepYear), AVG(LifeExpectancy), SUM(GNP)  
FROM country  
GROUP BY Continent  
HAVING COUNT(*) > 20 AND MIN(SurfaceArea) > 20;
```

GROUP BY og HAVING – forts.

- WHERE ekskluderer rader før gruppering.
- HAVING ekskluderer rader etter gruppering.
- SQL utføres nemlig i følgende rekkefølge:
 1. FROM
 2. WHERE
 3. GROUP BY
 4. HAVING
 5. SELECT
 6. ORDER BY(Altså ikke i kronologisk rekkefølge.)

GROUP BY og HAVING – forts.

```
SELECT Continent, COUNT(*), MIN(SurfaceArea),  
MAX(IndepYear), AVG(LifeExpectancy), SUM(GNP)  
FROM country  
WHERE IndepYear < 1950  
GROUP BY Continent  
HAVING COUNT(*) > 20 AND MIN(SurfaceArea) > 20;
```

- Hva er forskjellen på denne og den forrige spørringen vi kjørte?
- Hva er forskjellen i resultatet?
- Vi "mister" Nord-Amerika og Afrika, men vi "får" Asia!?
 - Dette skjer fordi WHERE fjerner en del rader fra resultatet *før* data grupperes og før HAVING-betingelsene så sjekkes.

