



# Økt 2 (av 12)

DB1100 Databaser

( Tomas Sandnes / [tomas.sandnes@kristiania.no](mailto:tomas.sandnes@kristiania.no) )

# Dagens tema

Dagens tema: [Spørringer mot én tabell](#).

- Dagens pensum: [Læreboka, kapittel 2](#).



NB, husk, denne økten (og de kommende) følger **flipped classroom** modellen:

1. Dere starter med å lese forelesningsslides (disse) og dagens lærebokkapittel (se over). Deretter begynner dere på oppgavene (finnes på Canvas), gjerne i grupper. *Merk: Disse aktivitetene (punkt 1) skjer før dere møter til øvingstimer! Gjerne på formiddagen, samme dag som det er lærerstyrt aktivitet.*
2. I øvingstimene jobber dere videre med oppgavene, og snakker med veilederne og/eller foreleser om det som er vanskelig. Om dere vil se løsningsforslaget, publiseres det rett før siste øvingstime.
3. Dere møter så til forelesningen, som holdes etter øvingstimene. Her går foreleser gjennom temaene og problemstillingene dere møtte i dagens øvinger.

## Til info: SQLSaturday på Fjerdingen, lørdag!

Førstkommende lørdag (1. september) holdes **PASS SQLSaturday** på Campus Fjerdingen!

- Frokost 08:30
- Første foredrag 09:30
- GRATIS å delta! :-D

Se link og les mer her:

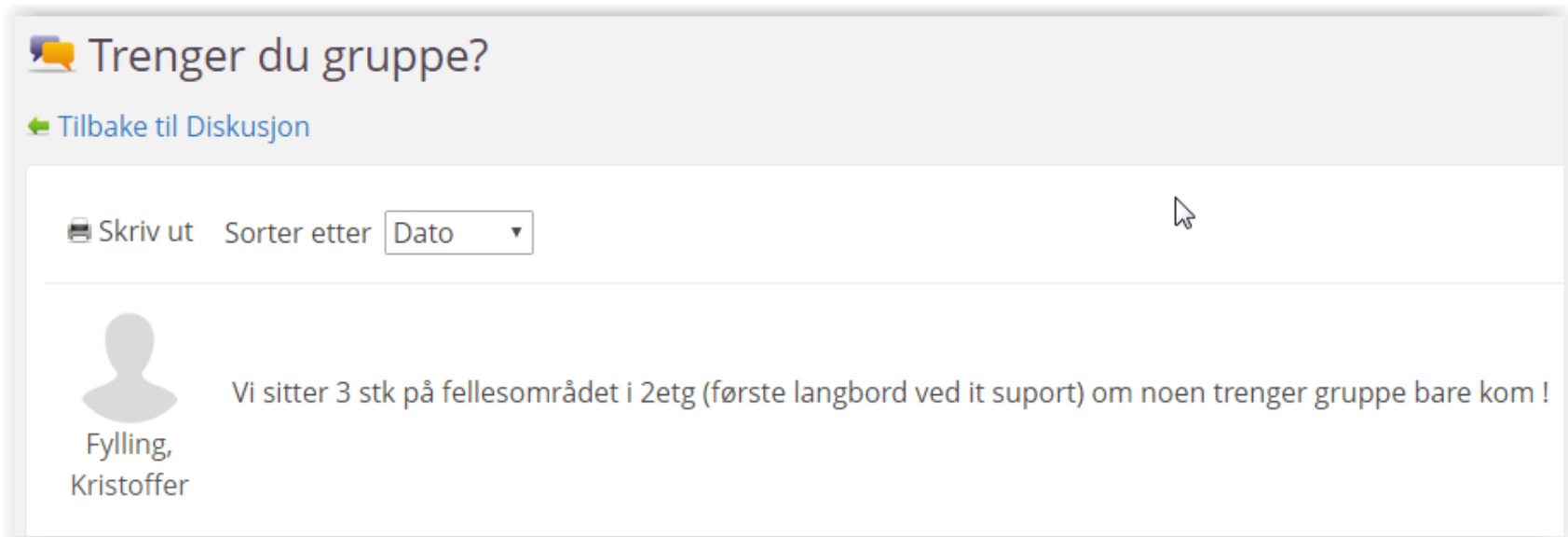
- [www.sqlsaturday.com/default.aspx](http://www.sqlsaturday.com/default.aspx)

# Gjennomgang øvingsoppgaver fra forrige økt

- Se løsningsforslag i Canvas.
  - Du finner alltid skriftlige løsningsforslag i Canvas.
- Til noen av løsningsforslagene finnes også videoer, laget av min kollega Per Lauvås.
  - Du finner dem i emnets [spilleliste](#).
- NB: I spillelista finner du også (av og til) forklaringer på nye temaer! :-)

# Om grupper

- Hvis du ønsker å delta i en gruppe, men ikke er på en gruppe nå; smisk deg inn på en eksisterende gruppe, eller kontakt en veileder i øvingstimen.
- Det er valgfritt å jobbe sammen i gruppe i DB1100, men det anbefales!
  - Tenk hvis flere gjorde som dette (fra tidligere år):



# Om bruk av forumet i forrige uke

- Mange gode oppgaver!
- Ikke få panikk hvis du syntes noe var vanskelig. :-)
- Nye muligheter i dag! Fortsett der dere slapp sist.
  - Å lage oppgaver gir god læring.
  - Oppgavene vil komme godt med når dere skal øve til eksamen.

# Hva er et db-script?

- Noen SQL-er i en fil som er kjørbare i en database.
- Jeg legger ut db-script til en del forelesninger, slik at dere enkelt kan prøve ut SQL-ene dere ser på slidene.
  - Alternativt kan dere skrive av SQL fra slides for hånd. (Best læring å skrive selv!)
  - Slike script ligger sammen med den aktuelle forelesningen, i Canvas.
- Kjør den relevante SQL-en for å se at det funker. Endre gjerne litt og se om resultatet blir som antatt.
  - Å kjøre en SQL selv gir bedre læring enn å se en SQL!
- Du finner mange db-script på pensumbokas [datasettside på nett!](#)

# Noen SQL funksjoner (rep. fra forrige økt)

- SQL har noen innebygde funksjoner, bl.a.:
  - COUNT(\*) → antall
  - AVG(kolonne\_navn) → gjennomsnitt
  - SUM(kolonne\_navn) → sum
  - MIN(kolonne\_navn) → minimum
  - MAX(kolonne\_navn) → maksimum
- For å få en meningsfull overskrift for slike kolonner kan vi gi resultatene egne navn. Dette kan gjøres ved å bruke det reserverte SQL ordet AS.

```
SELECT COUNT(*) AS 'Antall byer'  
FROM City;
```



Antall byer
4079



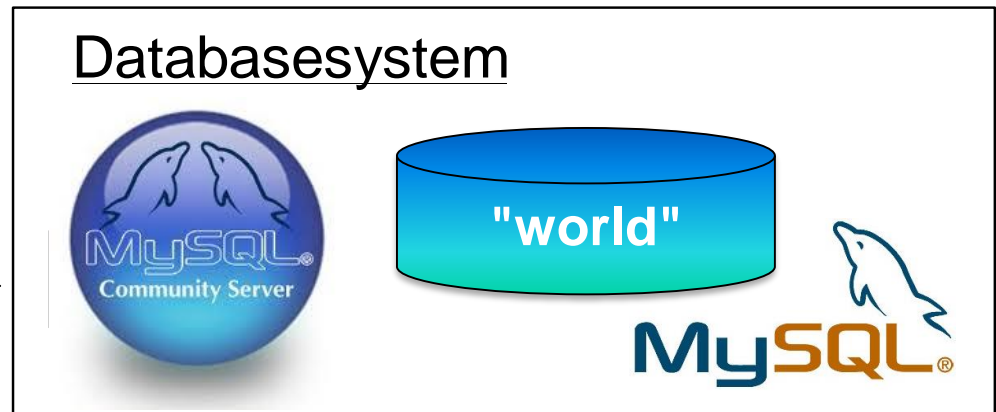
# Databasebegreper og MySQL

- Databasesystem = DBMS + database.
- DBMS = DataBase Management System:
- Vårt databasesystem: MySQL Community Server + Schema ("world").
- MySQL Workbench blir da ...?
  - Klienten (brukeren) som kommuniserer med DBMS.



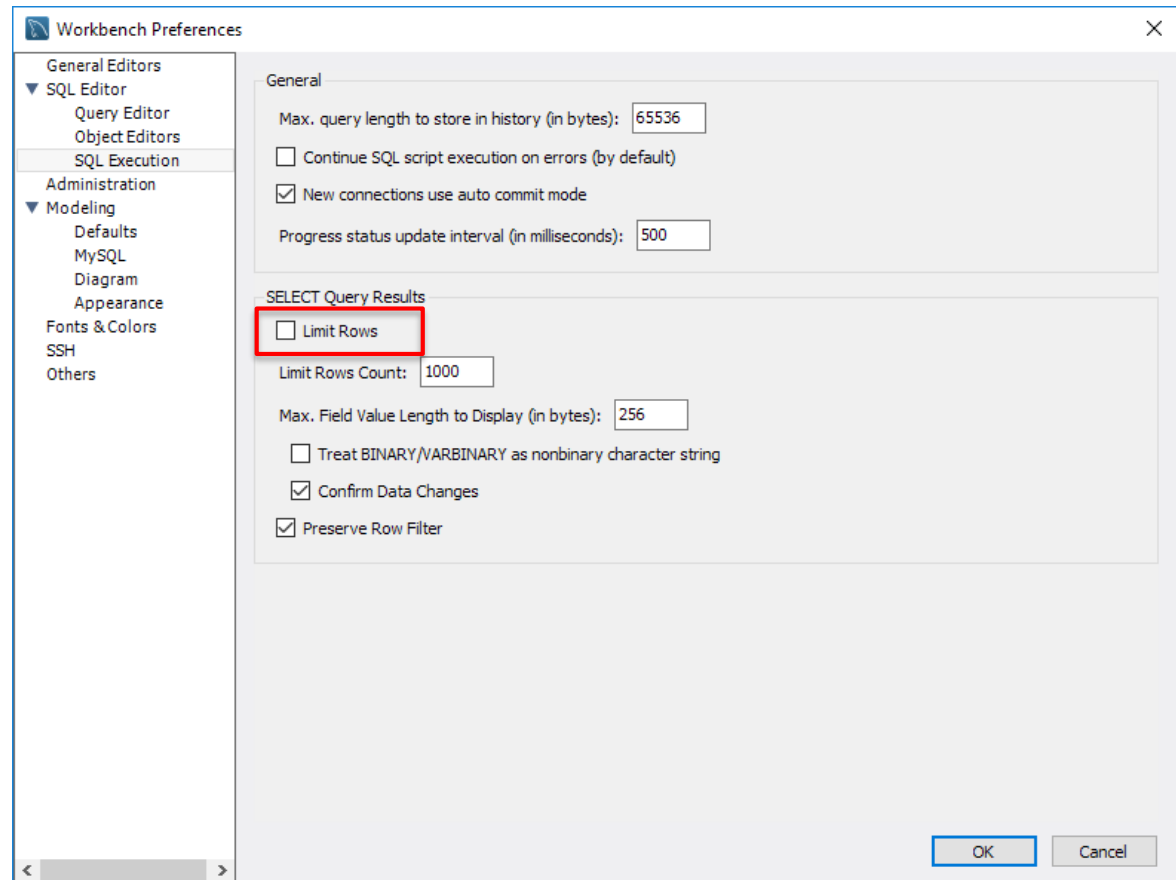
Bruker

SQL  
↔



# Oppsett, MySQL Workbench

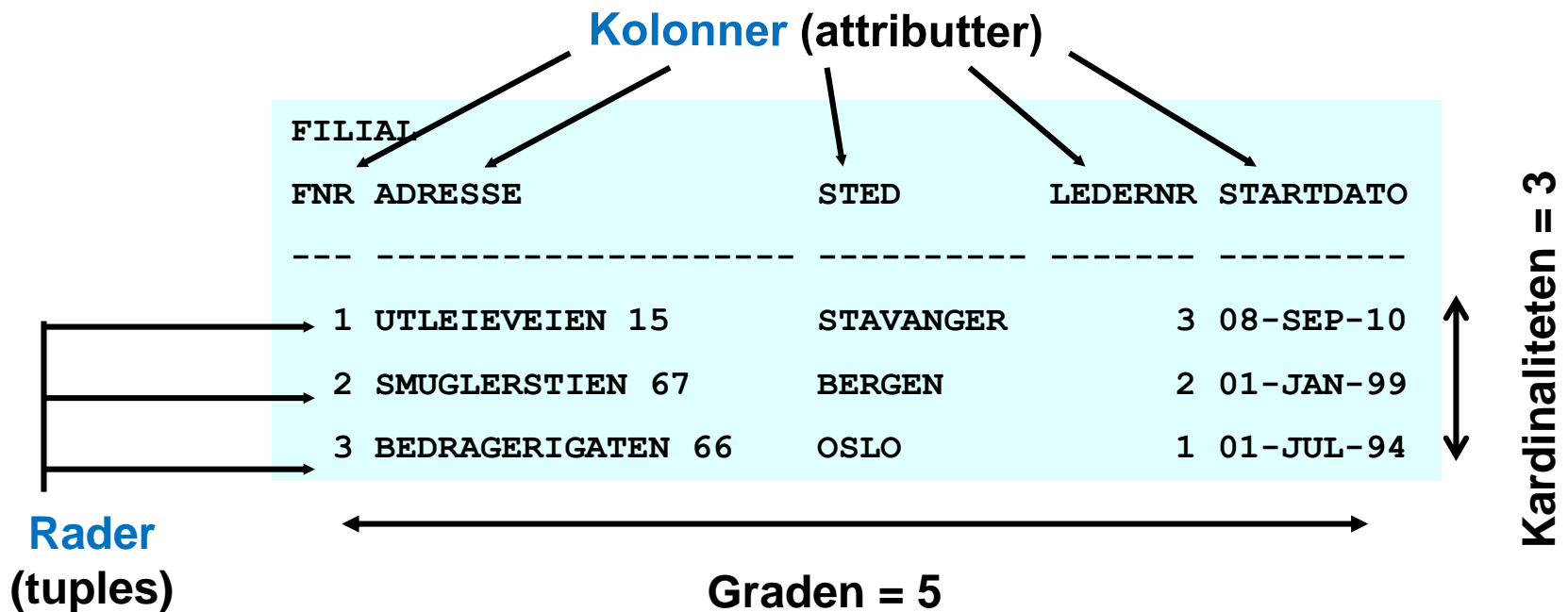
- Edit->Preferences i menyen.
- "SQL Execution" raden.
- Ta vekk kryss fra "Limit Rows".



# Relasjonsmodellen: Terminologi

- En **relasjon** er en **tabell** med kolonner og rader.
  - En **tuppel** er et annet navn for en **rad** i tabellen.
  - Et **attributt** er en navngitt **kolonne** i tabellen.
  - Et **domene** er mengden **tillatte verdier** for et eller flere attributter.
- Vi sier noe om størrelsen til en tabell (relasjon) ut i fra:
  - **Graden** til en tabell: **antall kolonner** den inneholder.
  - **Kardinaliteten** til en tabell: **antall rader** den inneholder.
- En **relasjonsdatabase** er en samling relasjoner.
  - Nivået av strukturering angir normaliseringen (kommer tilbake til normalisering på en senere forelesning).

# Rel.mod.: Terminologi – forts.



Domene

Attributt	Domene navn	Betydning	Domene definisjon
Fnr	Filialnummer	Mengden av alle mulige filialnummer	Number (2)
Adresse	Gatenavn	Mengden av alle gater i Norge	Streng (25)
Startdato	Startdato	Mengden av alle datoer	Date, fra 1- JAN-1985 format dd-mon-yy

# Egenskaper for å være en tabell

- Hver tabell må ha et unikt navn (innenfor denne database / schema).
- Hver kolonne må ha et unikt navn (innenfor denne tabell).
  - Verdiene til en kolonne må være fra samme domene.
  - Rekkefølgen på kolonnene kan ikke ha betydning.
- Det skal ikke finnes like rader (alle er unike).
  - Rekkefølgen på radene kan ikke ha betydning.
- Hver celle skal inneholde én gyldig verdi fra kolonnens domene (eller NULL – hvis tillatt).

# Dat typer

- Navn og syntaks for datatyper varierer litt fra database til database.
- MySQL inneholder en rekke datatyper. Blant de vanligere er: char, varchar, int, float, date og enum.
- Fullstendig oversikt (MySQL med fler) finner dere her:
  - [SQL datatypes @ w3schools.com](http://www.w3schools.com/sql/sql_datatypes.asp).

# Verdien NULL

- **NULL** representerer en kolonneverdi som ikke er satt for denne raden i tabellen.
- **MERK:**
  - NULL er ikke det samme som tallet 0.
  - NULL er ikke det samme som en blank/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

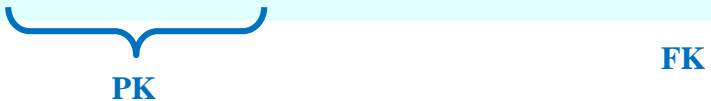
```
SELECT COUNT(*) AS AntLand
FROM country
WHERE IndepYear > 1814
OR IndepYear <= 1814
OR IndepYear IS NULL;
```

	AntLand
▶	239

# Nøkler

- Primærnøkkel (PK – primary key) er den unike identifikatoren for radene i en tabell. PK kan være sammensatt av flere kolonner.
- Fremmednøkkel (FK – foreign key) er en (eller flere) kolonne(r) i en tabell som viser til (har samme verdi som) en primærnøkkel i en annen (evt. samme) tabell.

LEIEKONTRAKT				
EIENDOMNR	FRADATO	TILDATO	LNR	LEIE
-----				
1	01-JUN-05	31-MAY-10	3	6500
1	01-JUN-08		3	7500
2	01-AUG-98	31-JAN-06	4	11000



PK

FK

# Integritet

- En primærnøkkels kolonner kan ikke inneholde NULL.
- En fremmednøkkel må ha samme verdi som primærnøkkelen i tabellen den refererer til, eller være NULL.

# Nøkler, oppgaver – skriv ned svarene dine!

## Eier:

Id	Navn	Adresse	Tlf	Epost
1	Ola Olsen	Liksomveien 2	22222222	ola@online.no
2	Ola Olsen	Januarveien 2	33333333	ola@is.com
3	Ina Jensen	Juliveien 3	44444444	ina@is.com

## Husdyr:

Id	Dyr	Navn	Eier
1	Katt	Mia	2
2	Hund	Passop	2
3	Papegøye	Polly	1
4	Katt	Kitty	3

1. Hvilke tabeller har vi?
2. Hvilken kardinalitet og grad har tabellene?
3. Hvilke kolonner kan være primærnøkler?
4. Hvilke kolonner kan være fremmednøkler?

# Nøkler - oppgaver

## City

ID	Name	CountryCode	District	Population
1	Kabul	AFG	Kabul	1780000
2	Qandahar	AFG	Qandahar	237500
3	Herat	AFG	Herat	186800
4	Mazar-e...	AFG	Balkh	127800
5	Amster...	NLD	Noord-H...	731200
6	Rotterd...	NLD	Zuid-Holl...	593321
7	Haag	NLD	Zuid-Holl...	440900
8	Utrecht	NLD	Utrecht	234323

## Countrylanguage

CountryCode	Language	IsOfficial	Percentage
ABW	Dutch	T	5.3
ABW	English	F	9.5
ABW	Papiamentu	F	76.7
ABW	Spanish	F	7.4
AFG	Balochi	F	0.9
AFG	Dari	T	32.1
AFG	Pashto	T	52.4
AFG	Turkmenian	F	1.9

## Country

Code	Name	Continent	Region	SurfaceArea	IndepYear	Population	LifeExpectancy	GNP	GNPOld	LocalName	GovernmentForm	HeadOfState	Capital	Code2
ABW	Aruba	North America	Caribbean	193.00	NULL	103000	78.4	828.00	793.00	Aruba	Nonmetropolitan T...	Beatrix	129	AW
AFG	Afghani...	Asia	Souther...	652090.00	1919	22720000	45.9	5976.00	NULL	Afganistan/A...	Islamic Emirate	Mohammad Omar	1	AF
AGO	Angola	Africa	Central ...	1246700.00	1975	12878000	38.3	6648.00	7984.00	Angola	Republic	JosÃ© Eduard...	56	AO
AIA	Anguilla	North America	Caribbean	96.00	NULL	8000	76.1	63.20	NULL	Anguilla	Dependent Territor...	Elisabeth II	62	AI
ALB	Albania	Europe	Souther...	28748.00	1912	3401200	71.6	3205.00	2500.00	ShqipÃ«ria	Republic	Rexhep Mejdani	34	AL
AND	Andorra	Europe	Souther...	468.00	1278	78000	83.5	1630.00	NULL	Andorra	Parliamentary Copri...		55	AD
ANT	Netherl...	North America	Caribbean	800.00	NULL	217000	74.7	1941.00	NULL	Nederlandse...	Nonmetropolitan T...	Beatrix	33	AN

1. Hvilke kolonner kan være primærnøkler?
2. Hvilke kolonner kan være fremmednøkler?

# 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:

```
select    kolonne [as navn]
from      tabell
[where    betingelse]
[group by grupperingsuttrykk] [having betingelse]
[order by kolonne]
```

Nye idag!

# SELECT

- \* velger alle kolonner.
- Et utvalg kolonner velges ved å navngi de, kommaseparert.
- Merk: Matematiske operatorer (+ - \* /) kan benyttes som en del av utvelgelsen.
- En del funksjoner gir summeringer: COUNT, AVG, ...
- Kolonner i resultatsettet/-tabellen kan gis nytt navn ved å bruke AS.

# 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
```



Ny idag!



# WHERE

- Operatorer:

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

- Logiske operatorer – setter sammen kriterier:

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

# ORDER BY

- Sorterer utvalget, basert på kolonnenavn.
  - Kan også benytte kolonnenummer, men dette er "deprecated". (Dvs. utgått standard, men fortsatt støttet. Vi bruker det altså IKKE.)
- Kan sortere flere kolonner (kommaseparert).
- Rekkefølge styres med ASC (ascending - stigende) og DESC (descending - synkende).
  - Ved ORDER BY uten ASC eller DESC blir rekkefølgen stigende.
  - (Uten ORDER BY er rekkefølgen vilkårlig/opp til systemet.)

# 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:
  - FROM
  - WHERE
  - GROUP BY
  - HAVING
  - SELECT
  - 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 fordi vi mister rader ved bruk av WHERE, noe som medfører at HAVING-betingelsen ikke lenger oppfylles.

# Øving

- Før øvingen skal du ha:
  - Lest kapittel 2 i pensumboka.
  - Lest igjennom denne slideserien.
  - Sett på SQL til slideserien. (Husk at det ofte ligger et db-script i Canvas for forelesningen.) Du lærer mest av å kjøre SQL spørringer selv, ikke bare lese de. Gjør gjerne små endringer også, og se om du får forventet resultat.
  - Gjerne ha begynt på øvingsoppgavene.
- Lurt å jobbe sammen i grupper før øvingen!
- Sjekk TimeEdit for rom til øving.
  - Ser ut til å bli: FGR-U113, FUN-207, FUN-305, FUN-309
- Våre flinke studentveiledere er på plass i øvingsrommene. :-)
  - Jeg sirkulerer mellom rommene. (Med mindre noen har en klonemaskin?)