

# SQL—Structured Query Language

- ▶ Hvorfor tabeller?
- ▶ Litt tabellterminologi
- ▶ Hente data fra tabeller
  - ▶ select-from-where
  - ▶ distinct
  - ▶ order by



Universe of discourse

Interesseområdet



Hva er vi interessert i å lagre informasjon om?

*Eksempler på informasjon:*

Ali har har tlfnr 97347777

100 g lettkokte havregryn inneholder 380 kcal

Michelle Bachelet er president i Amerika

En person med brukernavn mariulj tar IN2090 i 2015

Skjellettet består av ca. 206 enkeltknokler

Det er valgdag i Norge 14. september 2015



Jeg er interessert i å vite hvem som følger forelesningene i IN2090



Jeg er interessert i å vite hvem som følger forelesningene i IN2090



Hva menes med:

- følger forelesninger
- hvem

?

Jeg er interessert i navn på studenter som følger forelesningene i IN2090



Hva menes med:

- følger forelesninger ?

Jeg er interessert i navn på studenter som var på forelesningen onsdag  
12. september 2018



Ashar var på forelesningen i  
IN2090 20180912

Siri var på forelesningen i  
IN2090 20180912

Jeg er interessert i navn på studenter som har meldt seg til eksamen i IN2090



Finnes informasjonen noe sted?

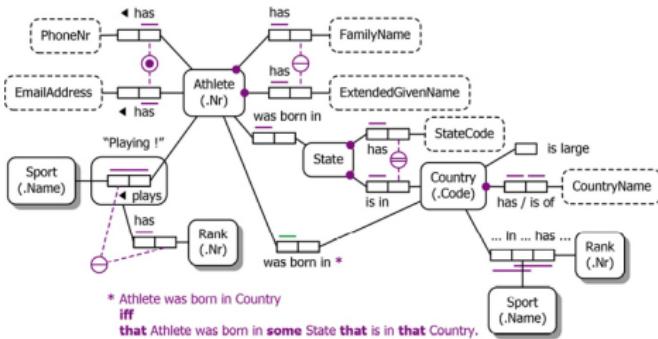
Kan den framskaffes?

Måles?

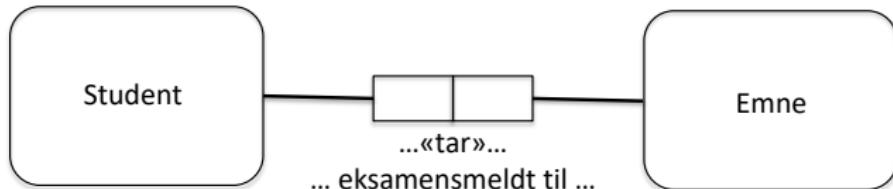
Beregnes?

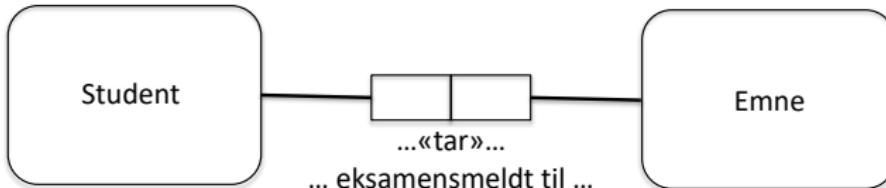
I IN2090 bruker vi ORM til å modellere fakta uttrykt som elementære setninger:

Ali tar INF1060  
Ali tar IN2090  
Ida tar IN2090  
Ida tar INF2220



En student med brukernavn gudbras er eksamensmeldt til et emne med emnekode IN2090





marisbru	IN2090
lindsay	IN2090
lindsay	INF1060
alexanrf	IN2090
alexanrb	IN2090
jonashag	IN2090
emilima	IN2090
mortensb	IN2090
erikbu	IN2090
kimmp	INF1060
kimmp	IN2090
kimmp	INF2220
oyvinfo	INF1500

*Forekomsttabell*

# Fakta i interesseområdet

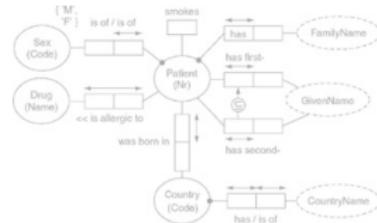
## Elementære setninger

# Fakta i interesseområdet

## Elementære setninger

### Generalisering

### Modellerer setningene med ORM



# Fakta i interesseområdet

# Elementære setninger

# Generalisering

# Modellerer setningene med ORM

## Lager tabeller

**... eller rettere  
sagt et TABELLSKJEMA**

StudentTarEmne ( brukernavn, emnekode )

**... eller rettere  
sagt et TABELLSKJEMA**

StudentTarEmne ( brukernavn, emnekode )

navn på tabellen  
antall kolonner (attributter)  
attributter (kolonnenavn)

## StudentTarEmne

brukernavn	emnekode
------------	----------

## StudentTarEmne

brukernavn	emnekode
marisbru	IN2090
lindasy	IN2090
lindasy	INF1060
alexanrf	IN2090
alexanrb	IN2090
jonashag	IN2090
emilima	IN2090
mortensb	IN2090
erikbu	IN2090
kimmp	INF1060
kimmp	IN2090
kimmp	INF2220
oyvinfo	INF1500

StudentTarEmne

**tabellnavn**

**attributt**

**tabellskjema (eller bare skjema)**

**tuppel/forekomst**

**instans/forekomster**

brukernavn	emnekode
marisbru	IN2090
lindsay	IN2090
lindsay	INF1060
alexanrf	IN2090
alexanrb	IN2090
jonashag	IN2090
emilima	IN2090
mortensb	IN2090
erikbu	IN2090
kimmp	INF1060
kimmp	IN2090
kimmp	INF2220
oyvinfv	INF1500

**tabellnavn**

**attributt**

**tabellskjema (eller bare skjema)**

**tuppel/forekomst**

**instans/forekomster**

StudentTarEmne

brukernavn	emnekode
marisbru	IN2090
lindsay	IN2090
lindsay	INF1060
alexanrf	IN2090
alexanrb	IN2090
jonashag	IN2090
emilima	IN2090
mortensb	IN2090
erikbu	IN2090
kimmp	INF1060
kimmp	IN2090
kimmp	INF2220
oyvinfv	INF1500

# Tabell = Relasjon

**relasjonsnavn**

**attributt**

**relasjonsskjema (eller bare skjema)**

**tuppel/forekomst**

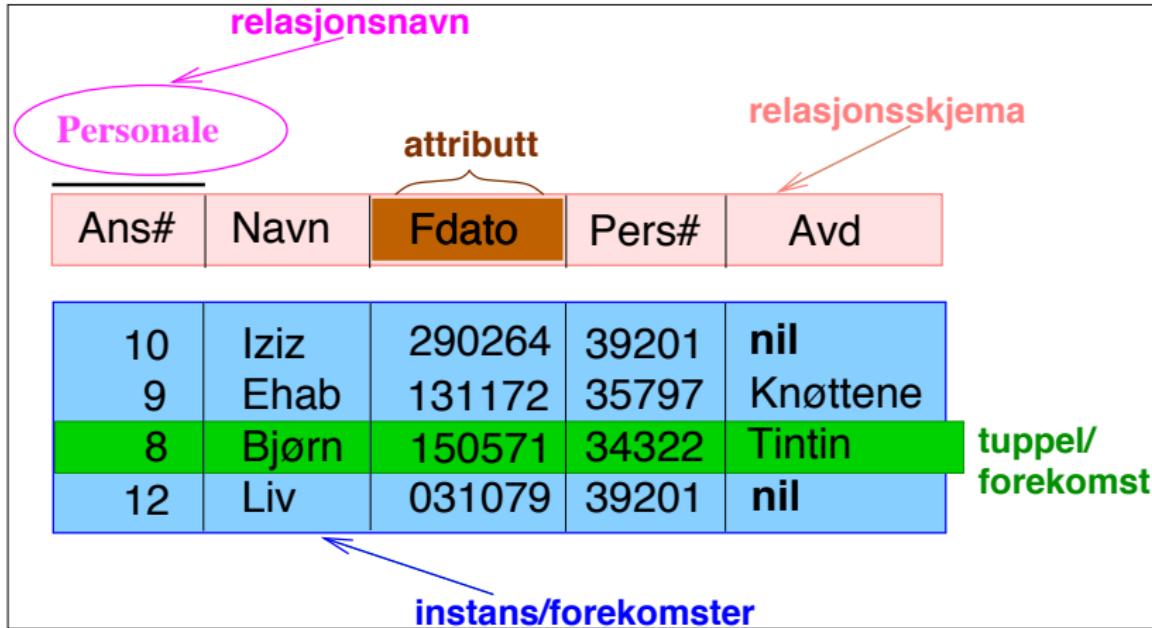
**instans/forekomster**

StudentTarEmne

brukernavn	emnekode
marisbru	IN2090
lindsay	IN2090
lindsay	INF1060
alexanrf	IN2090
alexanrb	IN2090
jonashag	IN2090
emilima	IN2090
mortensb	IN2090
erikbu	IN2090
kimmp	INF1060
kimmp	IN2090
kimmp	INF2220
oyvinfv	INF1500

# Tabell = Relasjon

# Relasjoner—terminologi



**tabell = relasjon**

# Hente/skrive ut data fra tabeller

```
select [distinct] ATTRIBUTLISTE  
from NAVNELISTE  
[where WHERE-BETINGELSE]  
[group by GRUPPERINGSATTRIBUTTER  
[having HAVING-BETINGELSE ] ]  
[order by ATTRIBUTT [asc | desc]  
[, ATTRIBUTT [asc | desc] ] ... ];
```

[ ] betyr at dette leddet er en valgfri del av setningen,  
så i dag skal vi bare se på denne delen:

# Hente/skrive ut data fra tabeller

```
select [distinct] ATTRIBUTLISTE  
from NAVNELISTE  
[where WHERE-BETINGELSE]  
[order by ATTRIBUTT [asc | desc]  
[, ATTRIBUTT [asc | desc] ... ];
```

[ ] betyr at dette leddet er en valgfri del av setningen,  
så i sin enkleste form ser setningen slik ut:

# Hente/skrive ut data fra tabeller

```
select ATTRIBUTLISTE  
from NAVNELISTE  
;
```

Eksempel:

```
select bnavn, snr, navn, stprog  
from Student  
;
```

samme som:

```
select * from Student ;
```

Student (bnavn, snr, navn, stprog)

# Hente/skrive ut data fra tabeller

```
select ATTRIBUTLISTE  
from NAVNELISTE  
;
```

Noen attributter:

```
select bnavn, navn  
from Student  
;
```

annen rekkefølge:

```
select snr, navn, bnavn, stprog  
from Student ;
```

Student (bnavn, snr, navn, stprog)

# Select-setningens enkeltdeler

## ► **select**

Angir hvilke attributter som skal vises i svaret

## ► **distinct**

Fjerner flerforekomster (duplikater) av svartuplene

## ► **from**

Navn på de relasjonene spørringen refererer til

## ► **where**

Seleksjonsbetingelse (kan inneholde en eller flere join-betingelser)

## ► **order by** Ordner tuplene i henhold til angitte kriterier

# Select-setningen

Typisk utseende:

```
select [distinct] A1, A2, ..., Aj  
from R1, R2, ..., Rk  
where C;
```

hvor

$R_1, R_2, \dots, R_k$  er relasjonsnavn

$A_1, \dots, A_j$  er attributter fra  $R_1, R_2, \dots, R_k$

$C$  er en betingelse

# select—eksempel 1

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, StartDato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PId, Timer)

Kunde(KId, Knavn, Adresse)

- **Oppgave:** Finn navn på de ansatte som er ansatt etter 2003. (Det kan være flere ansatte som har samme navn.)

# select—eksempel 1

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, StartDato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PlId, Timer)

Kunde(KId, Knavn, Adresse)

- ▶ **Oppgave:** Finn navn på de ansatte som er ansatt etter 2003. (Det kan være flere ansatte som har samme navn.)
- ▶ **Løsning**

```
select distinct Navn  
from Ansatt  
where AnsDato > date '2003-12-31' ;
```

# Merknader til *select*

- ▶ **select** (SQL) skiller ikke mellom store og små bokstaver, unntatt i tekststrenger
- ▶ **select** beregner *bager* (med unntak av noen av operatorene)

En *bag* er en *samling* med tupler der samme tuppel kan forekomme flere ganger. (Til forskjell fra en *mengde* tupler/forekomster, der samme tuppel ikke kan forekomme flere ganger.) Like tupler fjernes eventuelt ved å bruke **distinct**.

# select—eksempel 2

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, StartDato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PId, Timer)

Kunde(KId, Knavn, Adresse)

### ► Oppgave

Finn navn og startdato for alle prosjekter bestilt av kunden «Pust og pes AS». Sorter dem slik at det nyeste prosjektet kommer først.

# select—eksempel 2

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, Startdato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PId, Timer)

Kunde(KId, Knavn, Adresse)

### ► Oppgave

Finn navn og startdato for alle prosjekter bestilt av kunden «Pust og pes AS». Sorter dem slik at det nyeste prosjektet kommer først.

### ► Løsning

```
select    Pnavn, Startdato
from      Kunde K, Prosjekt P
where     Knavn = 'Pust og pes AS' and K.KId = P.KId
order by  Startdato desc;
```

# Seleksjons- og join-betingelser

La oss se nærmere på løsningen fra forrige lysark:

```
select Pnavn, Startdato  
from Kunde K, Prosjekt P  
where Knavn = 'Pust og pes AS' and K.KId = P.KId  
order by Startdato desc;
```

where-betingelsen består av to deler:

- ▶ Knavn = 'Pust og pes AS'

Dette leddet kalles en **seleksjonsbetingelse**

Det plukker ut forekomster i Kunde (her trolig bare en)

- ▶ K.KId = P.KId

Dette leddet kalles en **join-betingelse**

Det kobler sammen forekomster fra Kunde med forekomster i

Prosjekt forutsatt at verdiene i attributtene KId og Kid er like

# Uttrykk i betingelser — 1

**where**-betingelsen er et **boolsk uttrykk** hvor atomene har en av følgende former:

- ▶ Verdisammenlikning:  $P \text{ op } Q$ 
  - ▶  $P$  og  $Q$  må ha samme domene, minst en av dem må være et attributt, den andre kan være en konstant
  - ▶  $\text{op} \in \{=, <, >, \leq, \geq, \neq, \text{like}\}$   
(**like** er bare lov når  $Q$  er en konstant tekststreng)
- ▶ null-test:  $P \text{ is null}$  eller  $P \text{ is not null}$
- ▶ Relasjonssammenlikning: **exists**, **in**, **all**, **any**  
(Disse tar vi for oss i en senere forelesning)

# Uttrykk i betingelser — 2

Spesialregler for sammenlikning av **strenger** :

- ▶ Leksikografisk ordning:  $s < t$ ,  $s > t$ ,  $s \leq t$ ,  $s \geq t$
- ▶ Sammenlikning:  $s = t$ ,  $s \neq t$
- ▶ Mønstergjenkjenning:  $s$  **like**  $p$   
 $p$  er et mønster hvor
  - % matcher en vilkårlig sekvens (null eller flere tegn)
  - \_ matcher ett vilkårlig tegn

# Tekstmønstre

- I SQL kan vi bruke **like** for å sammenligne et tekst-attributt med et tekstmønster

Eksempel 1:

```
select firstname from person  
where firstname like '0_a';
```

passer med Oda og Ola og O4a, men  
ikke med Olga

Eksempel 2:

```
select firstname from person  
where firstname like '0%a';
```

passer med alle navn som begynner  
med «O» og slutter med «a», som  
Ola, Olga, Othilia, Oda,  
Ofjhwskjfhkxxa

# Tekstmønstre

- ▶ I SQL kan vi bruke **like** for å sammenligne et tekst-attributt med et tekstmønster
- ▶ Et **tekstmønster** er en **tekstkonstant** hvor to tegn, kalt jokertegn, har spesiell betydning:
  - ▶ **\_** (understrekning) passer med **ett vilkårlig tegn**
  - ▶ **%** passer med en **vilkårlig tekststreng (null eller flere tegn)**

Eksempel 1:

```
select firstname from person  
where firstname like '0_a';
```

passer med Oda og Ola og O4a, men ikke med Olga

Eksempel 2:

```
select firstname from person  
where firstname like '0%a';
```

passer med alle navn som begynner med «O» og slutter med «a», som Ola, Olga, Othilia, Oda, Ofjhwskjfhkxxa

# Uttrykk i betingelser — 3

Datoer og tidspunkter:

- ▶ Dato: **date** 'yyyy-mm-dd'
- ▶ Tidspunkt: **time** 'hh:mm', **time** 'hh:mm:ss'
- ▶ Tidspunkt med finere gradering enn sekund:  
**time** 'hh:mm:ss.ccc'
- ▶ Tidspunkt før GMT: **time** 'hh:mm:ss+hh'
- ▶ Tidspunkt etter GMT: **time** 'hh:mm:ss-hh'
- ▶ Dato og tid: **timestamp** 'yyyy-mm-dd hh:mm:ss'

## SELECT beregner *bager*

- ▶ **select** (SQL) skiller ikke mellom store og små bokstaver, unntatt i tekststrenger
- ▶ **select** beregner *bager* (med unntak av noen av operatorene)

En *bag* er en *samling* med tupler der samme tuppel kan forekomme flere ganger. (Like tupler fjernes eventuelt ved å bruke **distinct**.)

# select—eksempel 3

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, StartDato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PId, Timer)

Kunde(KId, Knavn, Adresse)

- *Oppgave:* Finn navn og tittel på alle som har arbeidet på prosjektet «Vintersalg»

# select—eksempel 3

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, StartDato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PId, Timer)

Kunde(KId, Knavn, Adresse)

- ▶ *Oppgave:* Finn navn og tittel på alle som har arbeidet på prosjektet «Vintersalg»
- ▶ *Løsning*

```
select distinct Navn, Tittel  
from      Ansatt A, Timeliste T, Prosjekt P  
where     Pnavn = 'Vintersalg' and  
           P.PId = T.PId and T.AId = A.AId;
```

# select—eksempel 3

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, StartDato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PId, Timer)

Kunde(KId, Knavn, Adresse)

- ▶ *Oppgave:* Finn navn og tittel på alle som har arbeidet på prosjektet «Vintersalg»

- ▶ *Løsning*

```
select distinct Navn, Tittel  
from      Ansatt A, Timeliste T, Prosjekt P  
where     Pnavn = 'Vintersalg' and  
           P.PId = T.PId and T.AId = A.AId;
```

- ▶ Her består join-betingelsen av to ledd. Den binder sammen en forekomst fra hver av de tre tabellene Ansatt, Timeliste og Prosjekt

# select—eksempel 3

## Skjema

Prosjekt(PId, Pnavn, KId, Pleder, StartDato)

Ansatt(AId, Navn, Tittel, Fdato, Pnr, AnsDato)

Timeliste(AId, Dato, PId, Timer)

Kunde(KId, Knavn, Adresse)

- ▶ *Oppgave:* Finn navn og tittel på alle som har arbeidet på prosjektet «Vintersalg»

- ▶ *Løsning*

```
select distinct Navn, Tittel  
from      Ansatt A, Timeliste T, Prosjekt P  
where     Pnavn = 'Vintersalg' and  
           P.PId = T.PId and T.AId = A.AId;
```

- ▶ Her består join-betingelsen av to ledd. Den binder sammen en forekomst fra hver av de tre tabellene Ansatt, Timeliste og Prosjekt
- ▶ At join-attributtene parvis har samme navn, er tilfeldig. Det holder at de har samme domene

# select—navnekonflikter

- Kvalifiser attributter med relasjonsnavn: R.A

## select—navnekonflikter

- ▶ Kvalifiser attributter med relasjonsnavn: R.A
- ▶ Navngi relasjoner med aliaser:  
...**from** R **as** S...    (**as** kan sløyfes)  
S blir en kopi av R med nytt relasjonsnavn

# select—navnekonflikter

- ▶ Kvalifiser attributter med relasjonsnavn: R.A
- ▶ Navngi relasjoner med aliaser:  
**...from R as S...**    (**as** kan sløyfes)  
S blir en kopi av R med nytt relasjonsnavn
- ▶ Gi attributter nytt navn:  
**select A as B from...**  
A omnavnes til B i resultatrelasjonen

# PostgreSQL

- ▶ For å bruke PostgreSQL: Fra Linux-promptet (...>), gi kommandoen
  - > `psql -h dbpg-ififi-kurs -U <brukernavn>`
  - og du blir bedt om å oppgi passordet ditt.
- ▶ Dersom du vil lage egne tabeller, skriver du ditt eget brukernavn i stedet for fdb
- ▶ For å kjøre en kommandofil, skriv `\i <filnavn>`
- ▶ For å avslutte, skriv `\q`
- ▶ Les forøvrig dokumentet om filmdatabasen og postgres som er tilgjengelig via lenke fra kursets semesterside.