Generell informasjon

- Faglærer: Torunn Gjester
- Nås på torunngj@oslomet.no
- Eksamensform: skriftlig, digital 3 timer uten hjelpemidler
- Undervisning mandag 10.30-12.15
  - 2 timer teori
  - 4 timer lab, mandag 12.30-16.15
  - Pensumlitteratur: Databasesystemer Bjørn Kristoffersen
- Alle ukeoppgaver er obligatoriske. Kan leveres som ett stort dokument i uke 6 12 og (18), eller godkjennes før innleveringsfrist på lab.
- Anbefaler å komme på lab gjøre oppgavene, få anledning til å stille spørsmål og få tilbakemelding ved godkjenning av hver ukeoppgave.

Innhold i kurset

- Få kunnskap om hva et database system er
- Lære SQL
- Lære ER-modellering
- Teori:
  - Indeksering
  - Transaksjoner
  - Lagrede prosedyrer
  - NoSQL databaser
- Verktøy:
  - MySql
  - MySQL Workbench

Hva kan dere fra før?

- Hva er data?
- Hva er informasjon?
- Hva er bestanddelene i et databasesystem?
- Hvilke databasesystemer kjenner dere til?

Dagens tema: Tabeller og enkle spørringer

- Database, relasjonsdatabase
- Databasehåndteringssystem (DBHS)
- Databasesystem

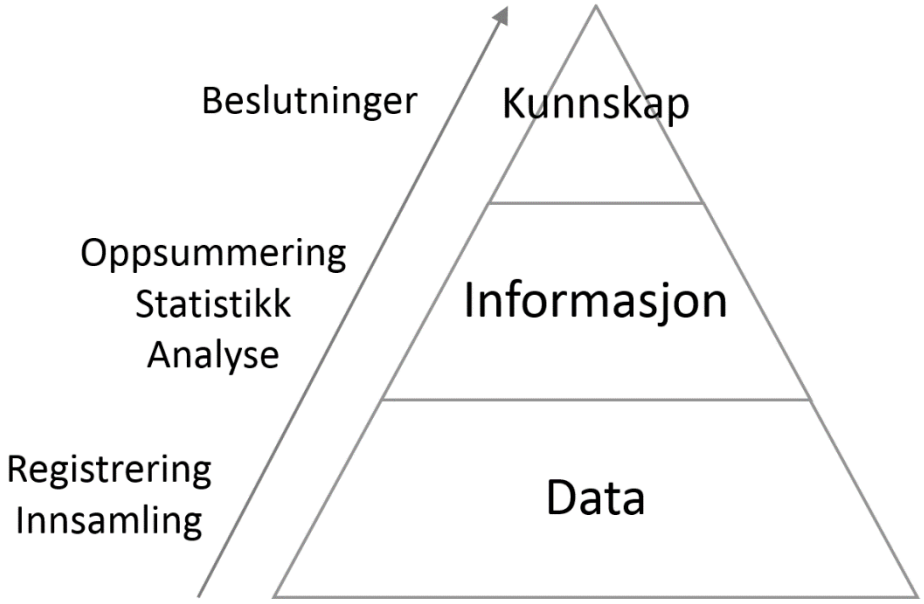
- Tabell, kolonne, rad, datatype, verdi, primærnøkkel

- Utvalgsspørringer i SQL
  - Velge ut rader
  - Velge ut kolonner
  - Kalkulerte kolonner
  - Sortering
  - Gruppering og mengdefunksjoner
  - Jokernotasjon og intervallsøk

Pensum: Kapittel 1 og 2

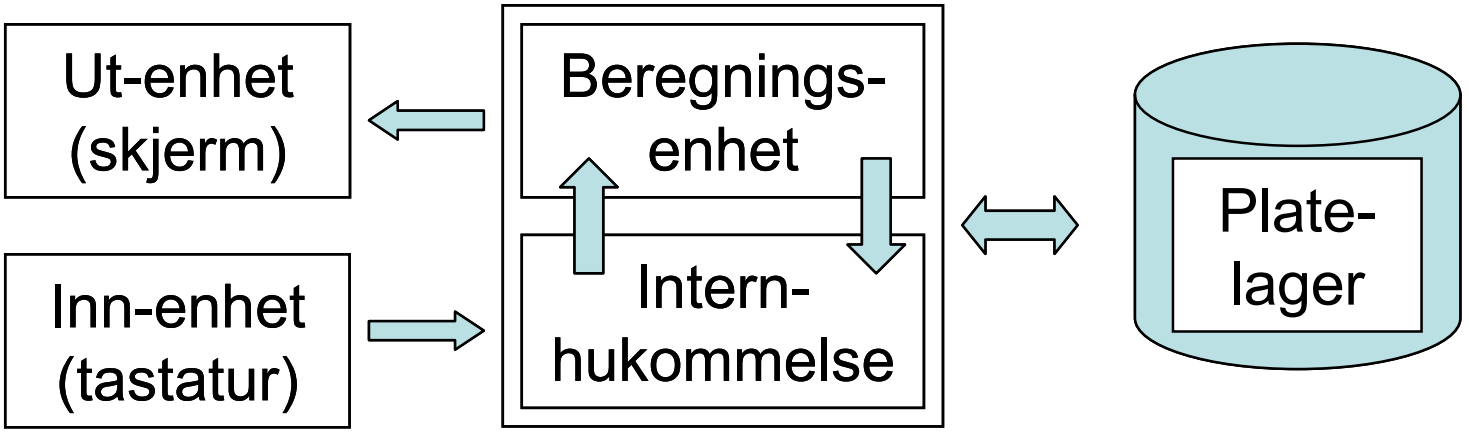
DIK(W)-pyramiden

- Databaser lagrer store mengder informasjon som er nødvendige i **daglig drift** av virksomheten.
- Men databaser brukes også mer **strategisk** som grunnlag for beslutninger.



Hvor starter man?

- Fysisk er en datamaskin komplisert. Logisk er den enkel:



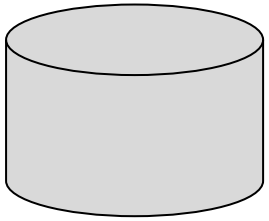
- En datamaskin kan to ting: Lagre data og utføre programmer.
- Vi tar utgangspunkt i datamaskinen som lagringsenhet.
- Hvilke data må en virksomhet lagre for å fungere?
- Eksempler: Varehandel, bibliotek, kommune, sykehus, skole,…

Måleenheter for datamengder

Måleenhet	Verdi	IT-bruk	Binær måleenhet	Verdi
kilobyte (kB)	10^3	2^{10}	kibibyte (KiB)	2^{10}
megabyte (MB)	10^6	2^{20}	mebibyte (MiB)	2^{20}
gigabyte (GB)	10^9	2^{30}	gibibyte (GiB)	2^{30}
terabyte (TB)	10^{12}	2^{40}	tebibyte (TiB)	2^{40}
petabyte (PB)	10^{15}	2^{50}	pebibyte (PiB)	2^{50}
exabyte (EB)	10^{18}	2^{60}	exbibyte (EiB)	2^{60}
zettabyte (ZB)	10^{21}	2^{70}	zebibyte (ZiB)	2^{70}
yottabyte (YB)	10^{24}	2^{80}	yobibyte (YiB)	2^{80}

Biter og byter

- Også nyttig å lage seg en forenklet **tankemodell** av lagringsmedier.
- Både **disk** og **minne** kan betraktes som en nummerert sekvens av byter.



- 1 **byte** = 8 **biter** som er 0 eller 1
- 1 bit kan lagre 2 alternative verdier
- 2 biter kan lagre $2^2 = 4$ verdier
- …
- 8 biter kan lagre $2^8 = 256$ verdier

1	1	0	0	1	1	0	1	0
2	0	0	1	1	1	0	1	1
3	1	1	1	0	1	1	1	0
4	0	0	0	0	0	1	1	0
5	1	0	0	0	0	0	1	1
6								

Representere tall

- 1 byte = 256 forskjellige **bitmønstre**
- 1 byte kan **tolkes** som heltallene [0..255]
- 2 byter kan tolkes som heltallene [0..65 535]

- Hvis vi bruker første bit som **fortegnsbit**, så kan vi representere [**-**32 768..**+**32 767]

- Ethvert **desimaltall** kan representeres som to heltall:
  - Tallet 486.229 kan skrives som 0.486229×10^3.
  - Lar seg representere ved heltallene 486229 og 3.
  - Samme teknikk kan brukes på alle desimaltall.

Representere tekst

- Et **tegnsett** tilordner et tall til hvert symbol.
  - Bokstaver, siffer, spesialtegn kan representeres som tall.
  - Med 2 byter kan vi representere 65 535 symboler.
  - En **tekststreng** er en sekvens av tegn.
  - ASCII og **Unicode** er to eksempler på tegnsett.

Tegn	Kode	Tegn	Kode
A	65	æ	145
Z	90	Æ	146
a	97	!	33
z	122	=	61
0	60	?	63
9	71	@	64

- Hvordan lagres 97?
- Hvordan lagres HEI?
- Hvordan lagres et blått punkt i et bilde?

- Vi benytter 2 tall systemet.
- Her er plassene: 2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0
- Dvs: 64 32 16 8 4 2 1
- 97 vil være 64 +32 + 1 dvs: 1100001

- For å lagre bokstaver benyttes ASCII-koding eller UTF-8

4	4	EOT	20	14	DC4	36	24	$	52	34	4
5	5	ENQ	21	15	NAK	37	25	%	53	35	5
6	6	ACK	22	16	SYN	38	26	&	54	36	6
7	7	BEL	23	17	ETB	39	27	'	55	37	7
8	8	BS	24	18	CAN	40	28	(	56	38	8
9	9	TAB	25	19	EM	41	29	)	57	39	9
10	A	LF	26	1A	SUB	42	2A	*	58	3A	:
11	B	VT	27	1B	ESC	43	2B	+	59	3B	;
12	C	FF	28	1C	FS	44	2C	,	60	3C	<
13	D	CR	29	1D	GS	45	2D	-	61	3D	=
14	E	SO	30	1E	RS	46	2E	.	62	3E	>
15	F	SI	31	1F	US	47	2F	/	63	3F	?

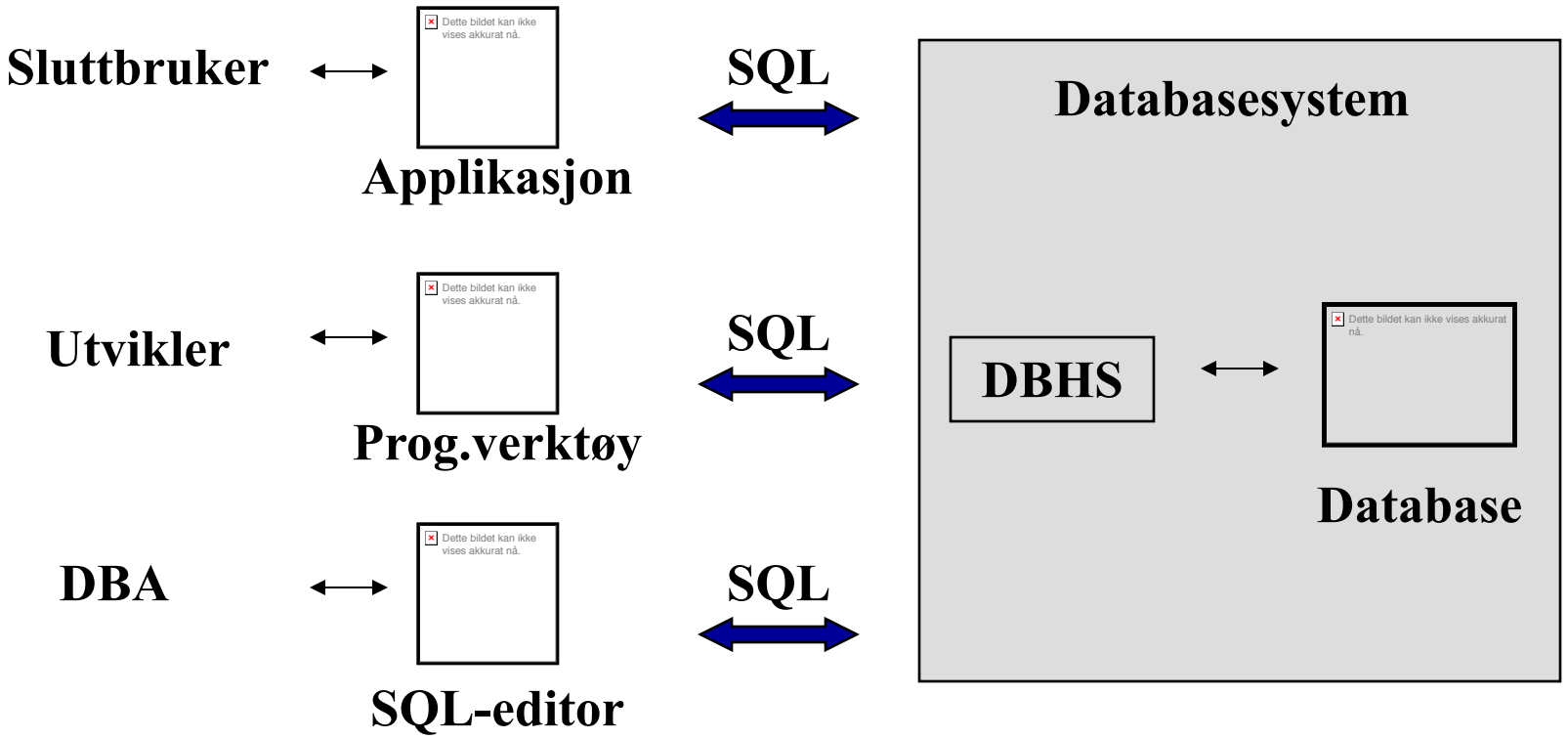
ASCII	Hex	Symbol	ASCII	Hex	Symbol	ASCII	Hex	Symbol	ASCII	Hex	Symbol
64	40	@	80	50	P	96	60	`	112	70	p
65	41	A	81	51	Q	97	61	a	113	71	q
66	42	B	82	52	R	98	62	b	114	72	r
67	43	C	83	53	S	99	63	c	115	73	s
68	44	D	84	54	T	100	64	d	116	74	t
69	45	E	85	55	U	101	65	e	117	75	u
70	46	F	86	56	V	102	66	f	118	76	v
71	47	G	87	57	W	103	67	g	119	77	w
72	48	H	88	58	X	104	68	h	120	78	x
73	49	I	89	59	Y	105	69	i	121	79	v

- For å skrive hei skriver man 72 69 73

- Et bilde består av pixler. Hver pixel har en farge som er representert med hvor mye rødt, grønt og blått som er I den. Benyttes 3 byte til å lagre en pixel vil et blått punkt være representert slik:

- 00000000 00000000 11111111

SQL og databasesystemer



DBHS = DataBaseHåndteringsSystem

(engelsk: DBMS = DataBase Management System)

Viktige begreper

- En database er en samling strukturerte data som blir brukt til et bestemt formål.
- Et databasehåndteringssystem (DBHS) er et verktøy for å lagre og gjenfinne store mengder delte data over lang tid på en sikker og effektiv måte for mange samtidige brukere.
  - Eksempler: MySQL, Access, SQL Server, Oracle, DB2, …
- Databasesystem = DBHS + database
- SQL er et språk for å kommunisere med databasesystemer.
- Databaseadministrator (DBA) har ansvaret for den daglige driften av et databasesystem.

Et papirskjema

Etternavn: Hansen **Fornavn**: Hans

Ansatt dato: 23.08.2003

Stilling: Programmerer **Lønn**: 325.000

Prosjektdeltakelse siste år:

Prosjektkode	**Timer**
P1002	44
P1007	25
P1012	10

Tabellen Ansatt

AnsattNr	Etternavn	Fornavn	AnsattDato	Stilling	Lønn
1	Veum	Varg	01.01.1989	Løpegutt	kr 183 000.00
2	Stein	Trude	10.10.1997	DBA	kr 270 700.00
3	Dudal	Inger-Lise	24.12.1985	Sekretær	kr 299 000.00
4	Hansen	Hans	23.08.2003	Programmerer	kr 325 000.00
5	Bjørnsen	Henrik	01.01.1997	Tekstforfatter	kr 375 000.00
6	Gredelin	Sofie	18.05.1995	Underdirektør	kr 625 850.00
7	Zimmermann	Robert	17.05.1992	Regnskapsfører	kr 375 000.00
8	Nilsen	Lise	03.04.1999	Direktør	kr 675 340.00
11	Fosheim	Katinka	13.09.1998	Selger	kr 420 000.00
13	Lovløs	Ada	12.08.2002	Programmerer	kr 384 250.00
16	Ibsen	Bjørnstjerne	02.01.2005	Tekstforfatter	kr 346 000.00
17	Fleksnes	Marve	17.05.2006	Lagerleder	kr 320 120.00
20	Felgen	Reodor	12.12.1998	Sykkelreparatør	kr 279 500.00
23	Karius	Jens	13.12.1998	Salgssekretær	kr 280 390.00
29	Wirkola	Gabriel	21.04.2006	Sekretær	kr 255 000.00

Tabellen Prosjekt

ProsjektNr	Budsjett	Leder	Start	Slutt
1001	kr 15 000.00	20	12.01.2005	12.03.2005
1002	kr 750 000.00	8	23.06.2005	23.07.2005
1007	kr 125 000.00	2	12.06.2006	
1009	kr 500 000.00	20	01.01.2006	
1012	kr 10 000.00	4	10.07.2006	
1020	kr 900 000.00	8	23.07.2005	01.09.2006

Tabellen ProsjektDeltakelse

Hvilke ansatte har jobbet på hvilke prosjekter – og hvor mange timer har de jobbet?

ProsjektDeltakelse er en koblingstabell.

ProsjektNr	AnsattNr	AntTimer
1001	1	12
1002	4	44
1002	8	20
1002	13	125
1002	20	2
1007	4	25
1007	11	20
1009	2	5
1009	17	10
1009	20	23
1012	4	10
1020	1	20
1020	8	35
1020	17	125

Nullmerker

- Legg merke til at det mangler noen verdier i tabellen Prosjekt. Vi kaller dette for nullmerker.

- Nullmerker kan skyldes at vi har glemt å registrere data, at vi ennå ikke kjenner til den korrekte verdien, eller at det ikke gir mening å registrere data.

- Nullmerker er *ikke* verdier.

- Nullmerker kan skape problemer!

Relasjonsdatabase

- En databasetabell kan betraktes som en matematisk relasjon (mer om dette i kapittel 6).
- En relasjonsdatabase består (logisk sett) av en samling tabeller (relasjoner).

- Andre typer av databaser:
  - Hierarkiske databaser
  - Nettverksdatabaser
  - Objektorienterte databaser
  - Objektrelasjonelle databaser
  - Logiske databaser

SQL

- SQL er et «abstrakt» programmeringsspråk - vi sier hva vi ønsker mer enn hvordan det skal beregnes. En utvalgsspørring:

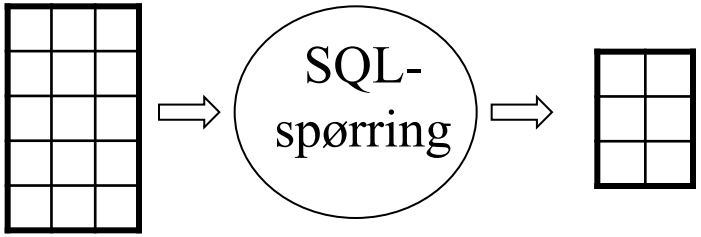
```
SELECT AnsattNr, Etternavn, Lønn
FROM Ansatt
WHERE Lønn < 280000
```

- Hva må DBHS gjøre? Hva må DBHS vite?
- **SELECT**, **FROM** og **WHERE** er reserverte ord i SQL: De har en spesiell betydning.

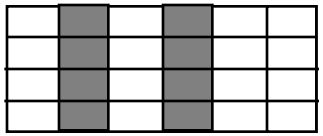
Spørreresultat

AnsattNr	Etternavn	Lønn
1	Veum	kr 183 000.00
2	Stein	kr 270 700.00
20	Felgen	kr 279 500.00
29	Wirkola	kr 255 000.00

- En utvalgsspørring tar tabeller som ”inndata” og gir som ”utdata” et spørreresultat som også er på ”tabellform”.



Velge ut kolonner



- Når vi kun er interessert i noen av kolonnene:

```
SELECT AnsattNr, Etternavn
FROM Ansatt
```

- Når vi vil ha alle kolonnene:

```
SELECT *
FROM Ansatt
```

- Utplukk av kolonner kan gi like rader. For å fjerne duplikater:

```
SELECT DISTINCT Stilling
FROM Ansatt
```

Velge ut rader

- Når vi vil plukke ut rader som oppfyller en gitt betingelse.

```
SELECT *
FROM Ansatt
WHERE Lønn < 280000
```

- En betingelse er et uttrykk som er sant eller galt.

Sortering

- Sortert navneliste:

```
SELECT AnsattNr, Etternavn
FROM Ansatt
ORDER BY Etternavn
```

- Flere sorteringskriterier:

```
SELECT Etternavn, Stilling, Lønn
FROM Ansatt
ORDER BY Stilling ASC, Lønn DESC
```

- **ASC** gir stigende sortering (standard) og **DESC** synkende.

Bruk parenteser!

- Vi ønsker å finne sekretærer/selgere som tjener mer enn 280.000 kroner i året.

```
SELECT *
FROM Ansatt
WHERE Lønn > 280000 AND
Stilling = 'Sekretær' OR Stilling = 'Selger'
```

- Kan spørringen tolkes på flere måter?
- Oppnår vi det vi vil?

Logiske operatorer AND, OR og NOT

- Brukes for å bygge sammensatte betingelser:

```
(Lønn > 280000) AND
(Stilling = 'Sekretær')

(Lønn < 300000) OR
(Lønn > 500000)

NOT (Lønn <= 300000)
```

AND	true	false
true	true	false
false	false	false

OR	true	false
true	true	true
false	true	false

NOT	true	false
	false	true

Operatorprioritet

1. – (unær minus, f.eks. –3)
2. * / %
3. + – (binære operatorer, f.eks. 2+2)
4. < <= > >= = <>
5. **NOT**
6. **AND**
7. **OR**

Jokernotasjon og intervallsøk

- Finn alle med etternavn som begynner på F:

```
SELECT *
FROM Ansatt
WHERE Etternavn>='F'AND Etternavn<'G'
```

- Jokernotasjon er mer elegant her:

```
SELECT *
FROM Ansatt
WHERE Etternavn LIKE 'F%'
```

- Hva med **Etternavn = 'F%'** ?
- Finn alle som har etternavn som slutter på 'sen' !

Kalkulerte kolonner

- Det er lov å bruke uttrykk i SELECT-delen:

```
SELECT AnsattNr, Etternavn,
       Lønn/12 AS [Lønn pr måned]
FROM Ansatt
```

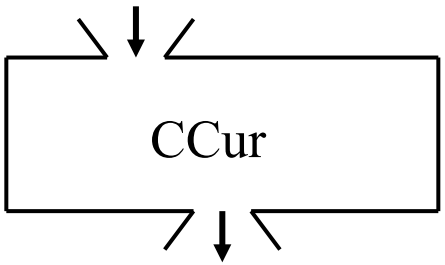
- Det er ikke nødvendig å lagre både årslønn og månedslønn!
- Lønn/12 er et uttrykk. For å få en meningsfull overskrift i utskriften gir vi denne kolonnen et navn.

Funksjoner

- Kommatall kan konverteres til valuta:

```
SELECT AnsattNr, Etternavn,
    CCur( Lønn/12 ) AS [Lønn pr måned]
FROM Ansatt
```

- CCur (Convert to Currency) er et eksempel på en funksjon (Access)



- En funksjon kan ha flere argumenter, men bare én returverdi.

- Til hver datatype finnes det mange funksjoner.

Dato og tid

- Varighet i ansettelsesforhold (antall dager):

```
SELECT Etternavn,
  YEAR(AnsattDato),
  AnsattDato - CURDATE() AS AntDager
FROM Ansatt
```

- Nå-tid: **CURDATE**
- Trekke ut deler av en dato: **YEAR**, **MONTH**, **HOUR**
- Sammenligne datoer: <, >, <=, >=, =

- Navn på funksjoner varierer noe fra system til system.

Operatorer og uttrykk

- Operatorer:

Aritmetiske:	`*, /, +, -`
Sammenligning:	`>,<, =, >=, <=, <, <>`
Jokernotasjon:	`LIKE`
Test for nullmerke:	`IS NULL`
Boolske:	`NOT, AND, OR`
Intervalltest:	`BETWEEN … AND …`

- Vi kan bygge opp uttrykk fra literaler (konstanter), kolonnenavn, funksjoner og operatorer:

```
(Stilling LIKE 'S*') AND ((Lønn/12)>15000)
```

Sammensatte navn

- Vi har ofte kolonner med samme navn i to tabeller.
  - Eksempel: AnsattNr i tabellen Ansatt og AnsattNr i tabellen ProsjektDeltakelse.

- Når vi jobber med flere tabeller *må* vi av og til bruke sammensatte navn. Det er *alltid* lov å bruke sammensatte navn:

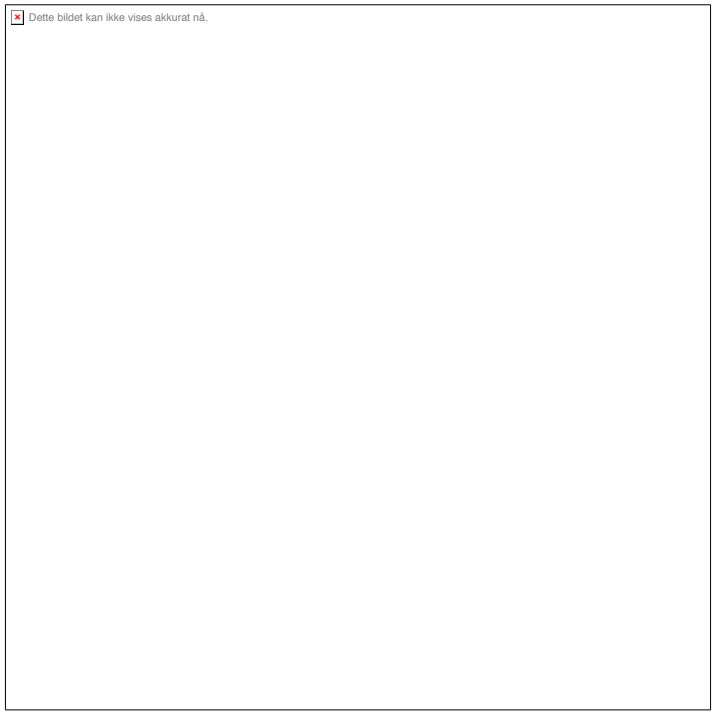
```
SELECT Ansatt.AnsattNr
FROM Ansatt
WHERE Ansatt.Lønn > 280000
```

Zooming og aggregerte data

Ola Nordmann bor i Hansegata 3, Andebu og tjener 320.000 kr. pr. år.

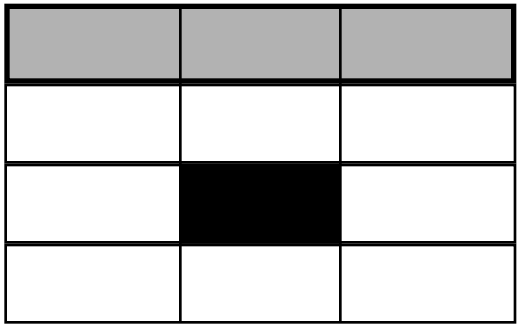
- Zoomer vi langt nok inn i kartet kan vi vise egenskaper ved Ola.

- Zoomer vi ut ønsker vi aggregerte data:
AVG, MIN, MAX, SUM, …

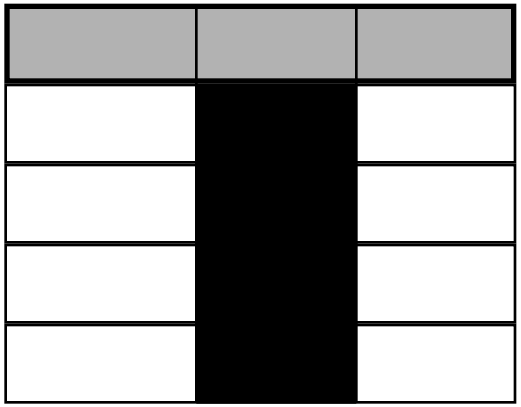


Funksjoner

- Operasjoner på enkeltverdier:
  - Eksempel: strengfunksjoner



- Operasjoner på verdisamlinger:
  - **AVG**(kolnavn) gjennomsnitt
  - **SUM**(kolnavn) sum
  - **COUNT**(*) antall
  - **MIN**(kolnavn) minimum
  - **MAX**(kolnavn) maksimum



Mengdefunksjoner

- Gjennomsnittslønn:

```
SELECT AVG(Lønn) AS Gjennomsnittslønn
FROM Ansatt
```

- Hvordan finner vi samlede lønnsutgifter?
- Hvordan finner vi gjennomsnittslønn for sekretærer?

- Antall ansatte: `SELECT COUNT(*) FROM Ansatt`

- Hva blir resultatet av denne spørringen:

```
SELECT SUM(Lønn) / COUNT(*) FROM Ansatt
```

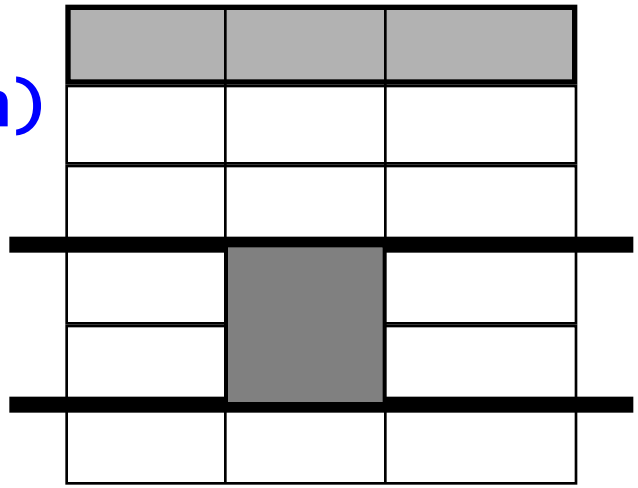
- Hva om vi har nullmerker i kolonnen Lønn?

```
SELECT SUM(Lønn) / COUNT(Lønn) FROM Ansatt
```

Gruppering

- Finn gjennomsnittlønn for hver stillingskategori.

```
SELECT Stilling, AVG(Lønn)
FROM Ansatt
GROUP BY Stilling
```



- Stilling har kun få ulike verdier!
- Grupperingskolonner må være med i resultatet.
- Gruppering brukes ofte sammen med mengdefunksjoner.

Spørreresultat

- For å finne **gjennomsnittslønn fordelt på stilling** må vi først danne **grupper**: selgerne, programmererne, …
- Så må vi beregne gjennomsnittslønn for hver gruppe.
- Resultatet får 4 rader fordi det er 4 forskjellige verdier i Stilling.

Stilling	Lønn
Sekretær	300 000
Programmerer	500 000
Selger	400 000
Direktør	800 000

Gruppebetingelse (HAVING)

- Hva blir resultatet her?

```
SELECT Stilling, AVG(Lønn), COUNT(*)
FROM Ansatt
GROUP BY Stilling
HAVING AVG(Lønn) > 280000
```

- **HAVING** på grupper svarer til **WHERE** på rader.

- Merk: Det er ikke lov å bruke mengdefunksjoner i **WHERE**-betingelser.

Gruppebetingelser og radbetingelser

- Betingelser på rader og på grupper:

```
SELECT Stilling, AVG(Lønn), COUNT(*)
FROM Ansatt
WHERE Year(AnsattDato) > 1990
GROUP BY Stilling
HAVING AVG(Lønn) > 280000
```

- Hvorfor kan ikke de to betingelsene slås sammen?
- Hvordan kan DBHS utføre spørringen?

Oppbygging av utvalgsspørringer

- Utvalgsspørringer (**SELECT**-spørringer) mot **1 tabell** følger dette «mønsteret»:

```
SELECT     Hvilke kolonner skal med?
FROM       Hva heter tabellen?
WHERE      Hvilke rader skal med?
GROUP BY   Gruppere på hvilke kolonner?
HAVING     Hvilke grupper skal med?
ORDER BY   Sortere på hvilke kolonner?
```

- Ikke alle delene må være med i alle spørringer.

- Vi skal legge på flere reserverte ord.
- **SELECT** er én blant mange SQL-kommandoer…

Les kap 1 og 2 og ta quiz

- http://dbsys.info/Databasesystemer/quiz/