$\boxed{3.2.2}$ i) $S(A,B,C,D)$

$A \to B,\ A \to C,\ C \to D$

a) Ikketrivielle FDer (ett attributt i hs), i tillegg til de angitte

$A \to D$ $\quad AB \to C$ $\quad AC \to B$ $\quad AD \to B$ $\quad ABC \to D$

$AB \to D$ $\quad AC \to D$ $\quad AD \to C$ $\quad ABD \to C$

$ACD \to B$

b) Kandidatnøkler:

$A^+ = ABCD$

Ingen andre bestemmer A, så A må være med i alle supernøkler. A er eneste minimale supernøkkel og dermed eneste kandidatnøkkel.

c) Supernøkler som ikke er kandidatnøkler:

Alle mengder som inneholder A, utenom A alene:

$AB \quad ABC \quad ABCD$

$AC \quad ABD$

$AD \quad ACD$

ii) $T(A,B,C,D)$ $\quad A \to B,\ B \to C,\ C \to D,\ D \to A$

a) $A^+ = ABCD$ så $A \to C,\ A \to D$ og $AB \to C,\ AD \to C;\ AB \to D,\ AC \to D,\ ABD \to C,\ ABC \to D$

$B^+ = BCDA$ så $B \to D,\ B \to A$ osv.

$C^+ = CDAB$ så $C \to A,\ C \to B$

$D^+ = ABCD$ så $D \to B,\ D \to C$

b) A, B, C, D

c) alle 2-, 3- og 4-elements delmgd av $\{A,B,C,D\}$

iii) $U(A,B,C,D)$

$AB \to C,\ BC \to D,\ CD \to A,\ AD \to B$

a) $A^+ = A,\ B^+ = B,\ C^+ = C,\ D^+ = D$ [Regel: For å sjekke om $X \to Y$ er med, beregn X^+ og se om $Y \subseteq X^+$]

$AB^+ = ABCD$ så $\underline{AB \to D}$ (har alt $AB \to C$) (og dermed også $ABC \to D$)

$AC^+ = AC$

$AD^+ = ADBC$ (har alt $AD \to B$) så $\underline{AD \to C}$

$BC^+ = BCDA$ så $\underline{BC \to A}$ (og dermed også $BCD \to A$)

$BD^+ = BD$

$CD^+ = CDAB$ så $\underline{CD \to B}$ (og dermed også $ACD \to B$)

$ABC^+ = ABCD$ så $\underline{ABC \to D}$

$ABD^+ = ABCD$ så $\underline{ABD \to C}$

$BCD^+ = ABCD$ så $\underline{BCD \to A}$

$ACD^+ = ABCD$ så $\underline{ACD \to B}$

b) AB, BC, CD, AD

c) $ABC, ABD, BCD, ABCD, ACD$

$\boxed{3.3.1}$ a) $R(A, B, C, D)$

$B \to A,\ C \to B,\ D \to C,\ A \to D$

i) $A^+ = ABCD$

$B^+ = ABCD$

$C^+ = ABCD$

$D^+ = ABCD$

Kandidatnøkler: A, B, C, D

Brudd på BCNF: ingen

(og dermed heller ingen brudd på EKNF.)

3.3.3

$R(A,B,C,D)$

$F = \{A \to B, A \to C\}$

Kandidatnøkkel: AD

$A \to B$, $A \to C$ bryter BCNF og 3NF

Dekomposisjon ihht. $A \to B$:

$R_1(A,C,D)$ $R_2(A,B)$

$F_1 = \{A \to C\}$ $F_2 = \{A \to B\}$

Kand. nkl: AD Kand. nkl: A

$A \to C$ bryter BCNF, dekomponer til

$R_{11}(A,D)$ $R_{12}(A,C)$

$F_{11} = \{\}$ $F_{12} = \{A \to C\}$

Resultat:

$R_{11}(A,D)$ $F_{11} = \{\}$

$R_{12}(A,C)$ $F_{12} = \{A \to C\}$

$R_2(A,B)$ $F_2 = \{A \to B\}$

(FD-bevarende dekomposisjon)

Dekomposisjon ihht. $A \to BC$:

$R_1'(A,D)$ $R_2'(A,B,C)$

$F_1' = \{\}$ $F_2' = \{A \to BC\}$

(FD-bevarende)

Forskjellen er at vi ender opp med tre og to tabeller, henholdsvis. To tabeller sparer noe lagringsplass.

$\boxed{3.5.2}$ $R(A,B,C,D,E)$

$S_1(A,B,C)$ $S_3(A,D)$ $S_4(A,B,E)$

Start-tabell:

	A	B	C	D	E
S_1	a	b	c	d_1	e_1
S_3	a	b_2	c_2	d	e_2
S_4	a	b	c_3	d_3	e

$AB \to C$ gir $c = c_3$ i C-kolonnen
$A \to D$ gir $d_1 = d = d_3$ i D-kolonnen

Tabellen ser nå slik ut:

	A	B	C	D	E
S_1	a	b	c	d	e_1
S_3	a	b_2	c_2	d	e_2
S_4	a	b	c	d	e

Siste rad er uten subskript-verdier, og vi har en tapsfri dekomposisjon.

$\boxed{3.6.1}$ $R(A,B,C)$

$B \twoheadrightarrow C$

(a_1, b, c_1) og (a_2, b, c_2) med i R betyr at også (a_1, b, c_2) og (a_2, b, c_1) må være med i R, ved å bytte om C-verdiene siden B-verdiene er like og $B \twoheadrightarrow C$.

Tilsvarende:

(a_1, b, c_1) og (a_3, b, c_3) med i R gir
(a_1, b, c_3) og (a_3, b, c_1) med i R

(a_2, b, c_2) og (a_3, b, c_3) med i R gir
(a_2, b, c_3) og (a_3, b, c_2) med i R.

3.3.1 e) $R(A, B, C, D, E)$

$AB \to C,\ C \to E,\ E \to A,\ E \to D$

- B må være med i samtlige kandidatnøkler

$B^+ = B$

$AB^+ = ABCED$ — AB kand. nøkkel. Nye FD: $AB \to D$, $AB \to E$

AB elementær kandidatnøkkel

$BC^+ = BCEAD$ — AC kand. nøkkel. Nye FD: $C \to A$, $C \to D$

AC ikke elem. kand. nøkkel

$BD^+ = BD$

$BE^+ = BEADC$ — BE kand. nøkkel. Nye FD: $BE \to C$

BE elementær kand. nøkkel

Brudd på BCNF: $C \to E,\ E \to A,\ E \to D,\ C \to D,\ C \to A$

Dekomposisjon til BCNF:

$$R(A, B, C, D, E)$$

$$\downarrow C \to ADE$$

$$R_1(A, \overline{C}, D, E)\ ,\quad R_2(\overline{B, C})$$

$$\downarrow \text{Brudd: } E \to A,\ E \to D$$

$$R_{11}(A, D, \overline{E})\ ,\ R_{12}(\overline{C}, E)$$

Brudd på EKNF: $E \to D,\ C \to D$

Dekomposisjon til EKNF:

1. Minimal overdekning

$AB \to C$
$C \to E$
$E \to A$
$E \to D$

Alle VS er minimale, ingen FDer er overflødige

2. $R_{AB}(\overline{A, B}, C)\ ,\ R_C(\overline{C}, E)\ ,\ R_E(A, D, \overline{E})$
3. $R_0 = \emptyset$
4. R_{AB} inneholder AB som er kand. nøkkel.

3.7.1 $R(A,B,C,D,E)$

$A \twoheadrightarrow BC$

$B \rightarrow E$

$C \twoheadrightarrow D$

a+b) $A \rightarrow D$ eller $A \twoheadrightarrow D$?

A	B	C	D	E
a	b_1	c_1	d	e_1
a	b	c	d_2	e
a	b	c	d	e_1
a	b_1	c_1	d_2	e

Rad 1–2: Startrader. $d = d_2$ gir $A \rightarrow D$, rad helt uten subskript gir $A \twoheadrightarrow D$

Rad 3–4: $A \twoheadrightarrow BC$

$B \rightarrow E$ gir $e = e_1$ (rad 1 og 4 / 2 og 3).
Rad 3 blir da uten subskript og $A \twoheadrightarrow D$ holder.
Derimot har vi ingen regel som kan gi $d = d_2$, slik at $A \rightarrow D$ holder ikke.

c+d) $A \rightarrow E$ eller $A \twoheadrightarrow E$?

A	B	C	D	E
a	b_1	c_1	d_1	e
a	b	c	d	e_2
a	b	c	d_1	e
a	b_1	c_1	d	e_2

Rad 1–2: Startrader

Rad 3–4: $A \twoheadrightarrow BC$

$B \rightarrow E$ gir $e = e_2$ (rad 1 og 4 / 2 og 3) slik at $A \rightarrow E$ holder. Siden enhver FD er en MVD holder også $A \twoheadrightarrow E$. (Dette kan evt. vises ved å bruke $C \twoheadrightarrow D$ på rad 2 og 3 i tabellen over.)

2.4.1

Product(maker, model, type)
PC(model, speed, ram, hd, price)
Laptop(model, speed, ram, hd, screen, price)
Printer(model, color, type, price)

a) De som selger printere, men ikke PCer:

$$\pi_{maker}(\sigma_{type='printer'}(Product) - \sigma_{type='pc'}(Product))$$

b) PCer med hastighet ≥ 2.50:

$$\pi_{model}(\sigma_{speed>=2.50}(PC))$$

c) De som selger laptoper med harddisk minst 120GB:

$$\pi_{maker}(\sigma_{hd>=120}(Product \bowtie Laptop))$$

Det er antakelig unødvendig med $\sigma_{type='laptop'}$, her, for Laptop skal bare matche de i Product med type='laptop'.

d) Modell og pris på produkter fra C:

$$\pi_{model,price}(\sigma_{maker='C'}(Product) \bowtie PC)$$
$$\cup$$
$$\pi_{model,price}(\sigma_{maker='C'}(Product) \bowtie Laptop)$$
$$\cup$$
$$\pi_{model,price}(\rho_{Product(maker,model,ptype)}(\sigma_{maker='C'}(Product)) \bowtie Printer)$$

NB! Må rename fordi type i Product ikke må matches mot type i Printer!

e) Alle sort-hvit laserskrivere:

$$\pi_{model}(\sigma_{\text{not color and } type='laser'}(Printer))$$

5.2.1

R:

A	B
1	2
3	4
1	2
3	5
4	5

S:

B	C
1	2
3	5
3	6
4	5
1	3
4	5

a) $\pi_{A^2, B^2, A+B}(R)$:

A^2	B^2	$A+B$
1	4	3
9	16	7
1	4	3
9	25	8
16	25	9

b) $\pi_{B-1, C+1}(S)$:

$B-1$	$C+1$
2	3
4	6
4	7
5	6
2	4
5	6

c) $\tau_{A,B}(R)$: [(1,2), (1,2), (3,4), (3,5), (4,5)]

d) $\tau_{C,B}(S)$: [(1,2), (1,3), (3,5), (4,5), (4,5), (3,6)]

e) $\delta(R)$:

A	B
1	2
3	4
3	5
4	5

2.4.3

a) Skip lansert før 1917:

$\pi_{name}(\sigma_{launched < 1917}(Ships))$

name
Haruna
Hiei
Kirishima
Kongo
Renown
Repulse
Resolution
Revenge
Royal oak
Royal Sovereign

b) Skip senket i slaget Surigao Strait:

ship
Fuso
Yamashiro

$\pi_{ship}(\sigma_{battle = 'Surigao\ Strait'\ and\ result = 'sunk'}(Outcomes))$

c) Skip som brøt Washington-avtalen:

$\pi_{name}(\sigma_{launched >= 1921}(Ships) \bowtie \sigma_{displacement > 35000}(Classes))$

name
Alabama
Iowa
Missouri
Musashi
New Jersey
South Dacota
Wisconsin
Yamato

d) Navn, vekt, antall våpen på skip som deltok i North Cape-slaget:

$\pi_{name, displacement, numGuns}\left(Classes \bowtie \left(Ships \bowtie_{name = ship} \sigma_{battle = 'North\ Cape'}(Outcomes)\right)\right)$

name	displacement	numGuns

(tom; verken Duke of York eller Scharnhorst står i Ships)

2.5.1

a) PC med hastighet < 3.00 må være priset ≤ $800.

$$\sigma_{speed < 3.00 \text{ and } price > 800}(PC) = \emptyset$$

[Brytes av eksempeldataene, se modell 1001: speed = 2.66, pris $2114.]

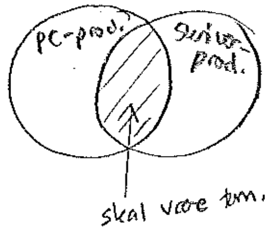
b) Laptop med skjerm mindre enn 15,4" må ha ≥ 120 Gb harddisk eller pris < $1000.

Det betyr at mengden av laptops med skjerm < 15,4", harddisk < 120Gb og pris ≥ $1000, skal være tom:

$$\sigma_{screen < 15.4 \text{ and } hd < 120 \text{ and } price >= 1000}(Laptop) = \emptyset$$

[Brytes av eksempeldataene, se modell 2004: screen = 13.3, hd = 60, price = 1150]

c) Ingen PC-produsenter får lage skrivere.



$$\sigma_{type='pc'}(Product) \cap \sigma_{type='printer'}(Product) = \emptyset$$

[Brytes av eksempeldataene; produsent D lager begge typer.]

2.5.1

d) Hvis laptop med større memory enn en PC, skal prisen også være høyere.

Idé: Finn par av laptoper og PCer hvor laptopen har størst memory. Finn par hvor laptopen har høyest pris. Førstnevnte mengde skal være inneholdt i sistnevnte.

$L(m, s, r, h, sc, p) :=$ Laptop; (renaming)

Answer(, . . .) :=

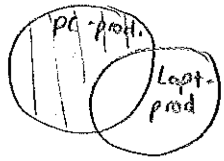
$$\sigma_{r > ram}(L \times PC) \subseteq \sigma_{p > price}(L \times PC)$$

[Moteksempel i eksempeldataene:

Laptop 2002 har ram = 1024 og price = 949
PC 1002 har ram = 512 og price = 995]

6.2.2

a) Produsenter av PCer, men ikke Laptops:




```
(select maker
 from Product
 where type = 'pc')
except
(select maker
 from Product
 where type = 'laptop');
```

← må ha mengdeoperator her (altså except, ikke except all) eller ha select distinct kombinert med except all

maker
C
D

b) Produsent, hastighet for laptops med hd >= 100 GB:

```
select distinct p.maker, l.speed
from Product p, Laptop l
where p.model = l.model and l.hd >= 100;
```

maker	speed
E	2,00
A	2,16
B	1,83
F	1,60
G	2,00

6.2.2

c) Modell, pris for produkter fra C:

```
(select q.model, q.price
 from Product p, PC q
 where p.model = q.model and p.maker = 'C')
union all
(select q.model, q.price
 from Product p, Laptop q
 where p.model = q.model and p.maker = 'C')
union all
(select q.model, q.price
 from Product p, Printer q
 where p.model = q.model and p.maker = 'C');
```

union all gir en mengde som svar fordi hver av selectene gir en mengde ut (og ikke en bag). Det skyldes at model er primærnøkkel i hver av relasjonene PC, Laptop og Printer.

model	price
1007	510

6.3.1 [Subqueries, minst to ekte forskjellige fremgangsmåter:]

a) Alle som lager laptop med hastighet minst 2.0:

```
select distinct p.maker
from Product p
where p.model in (select L.model
                  from Laptop L
                  where L.speed >= 2.0);
```

```
select distinct p.maker
from Product p
where exists ( select *
               from Laptop L
               where L.model = p.model and L.speed >= 2.0);
```

b) Skriverne med høyest pris:

```
select p.model
from Printer p
where p.price >= all (select q.price
                      from Printer q);
```

```
select p.model
from Printer p
where p.price = (select max(q.price)
                 from Printer q);
```

6.3.1

c) Laptops med lavere hastighet enn den raskeste PCen:

```
select L.model
from Laptop L
where L.speed < (select max(p.speed)
                 from PC p);
```

```
select L.model
from Laptop L
where L.speed < any (select p.speed
                     from PC p);
```

d) Modellene med lavest pris:

```
select u.model
from ((select model, price from PC)
      union all
      (select model, price from Laptop)
      union all
      (select model, price from Printer)) as u
where u.price >= all ((select price from PC)
                      union all
                      (select price from Laptop)
                      union all
                      (select price from Printer));
```

6.3.1

d) (2. variant) (I like seeing elegant, men ...)

```
(select model
 from PC
 where price <= all ( (select min(price) from PC)
                      union all
                      (select min(price) from Laptop)
                      union all
                      (select min(price) from Printer)))
 union all
(select model
 from Laptop
 where price <= all ((select min(price) from PC)
                      union all
                      (select min(price) from Laptop)
                      union all
                      (select min(price) from Printer)))

 union all
(select model
 from Printer
 where price <= all ((select min(price) from PC)
                      union all
                      (select min(price) from Laptop)
                      union all
                      (select min(price) from Printer)))
```

Oppgave 1

$F = \{A \to C,\ AC \to D,\ E \to AD,\ E \to H\}$

$G = \{A \to CD,\ E \to AH\}$

Ekvivalens hvis FDene i den ene følger fra den andre, og omvendt.

F gir $A \to CD$?

A	B	C	D	E	H
A	b_1	C	D	e_1	h_1
A	B	$\cancel{c_2}$ C	$\cancel{d_2}$ D	E	H

} $A \to CD$?

($A \to C$, $AC \to D$)

Bruker FDene i F, får at radene er like i ACD. Så F gir $A \to CD$.
Alternativt: Lukk A i hht. F:

$A^+ = ACD$ i F

Så $A \to CD$ holder i F

F gir $E \to AH$?

A	B	C	D	E	H
A	b_1	c_1	$\cancel{d_1}$ D	E	H
$\cancel{a_2}$ A	B	C	D	E	$\cancel{h_2}$ H

} $E \to AH$?

Ja. ($E \to AD$, $E \to H$)
Alternativt:

$E^+ = EADHC$ i F

G gir $A \to C$?

A	B	C	D	E	H
A	b_1	C	d_1	e_1	h_1
A	B	c_2	D	E	H

} $A \to C$?

Ja. ($A \to CD$)
Alternativt:

$A^+ = ACD$ i G

G gir $AC \to D$?

A	B	C	D	E	H
A	b_1	C	D	e_1	h_1
A	B	$\cancel{c_2}$ C	$\cancel{d_2}$ D	E	H

} $AC \to D$? Ja.
Alternativt: $AC^+ = ACD$ i G

G gir $E \to AD$?

A	B	C	D	E	H
A	b_1	c_1	D	E	$\cancel{h_1}$ H
$\cancel{a_2}$ A	B	C	d_2	E	H

} $E \to AD$?

Nei. Radene er ikke like i EAD
Alternativt:
$E^+ = EAH$ i G
Herav følger ikke $E \to AD$.

G gir $E \to H$?

A	B	C	D	E	H
$\cancel{a_1}$ A	b_1	$\cancel{c_1}$ C	$\cancel{d_1}$ D	E	H
A	B	C	D	E	$\cancel{h_2}$ H

} $E \to H$? Ja.
Alternativt: $E^+ = EAHCD$ i G.

Siden $E \to AD$ ikke følger av G, er F og G ikke ekvivalente.
Men alle i G følger av FDene i F, så G følger av F.

Oppgave 2

$AB \to CD$, $B \to D$

A må være med i alle kandidatnøkler fordi den kun er i venstresiden.

$A^+ = A$

$AB^+ = ABCD$ – AB er en kandidatnøkkel.

$AC^+ = AC$

$AD^+ = AD$

AB er eneste kandidatnøkkel.

$AB \to CD$: AB er kandidatnøkkel og supernøkkel, så er på BCNF.

$B \to D$:

÷ B er ikke en supernøkkel.
÷ D er ikke et nøkkelattributt
÷ B er ikke inneholdt i en kandidatnøkkel

brudd på 2NF.
$B \to D$ er på 1NF.

Relasjonen er på 1NF, men bryter 2NF.

Dekomposisjon i henhold til FDen som bryter 2NF:

R(A,B,C,D)

S(B,D)
$B \to D$
På BCNF

T(A,B,C)
$AB \to C$
På BCNF

	A	B	C	D
ABC	A	B	C	~~d_1~~ D
BD	a_2	B	c_2	D

Tapsfri fordi første rad er uten indekserte forekomster.

$A \twoheadrightarrow D$ innføres.

A	B	C	D
A	b_1	c_1	D
A	B	C	~~d_2~~ D
A	B	C	D
A	b_1	c_1	~~d_2~~ D

Rad 1–2: $A \to D$?
Rad 3–4: $A \twoheadrightarrow D$

Ja, for A og D i rad 1 og 2.

7.4.2

a)
```
create assertion max3ships
  check ( 3 >= all ( select count(*)
                     from Classes c, Ships s
                     where s.class = c.class
                     group by s.class ));
```

Hvis vi skal ta alvorlig at group by-attributtene alltid skal være med i select, må vi gjøre det f.eks. slik:

```
create assertion max3ships
  check ( 3 >= all ( select nships
                     from ( select s.class, count(*) as nships
                            from Classes c, Ships s
                            where s.class = c.class
                            group by s.class ) as t ));
```

b)
```
create assertion shipwithclassname
  check ( not exists ( ( select class as name from Classes )
                       except
                       ( select name from Ship where name = class )));
```

(Kunne vært løst uten check assertion ved en enkel fremmednøkkel)

2.5.2

Classes(class, type, country, numGuns, bore, displacement)
Ships(name, class, launched)
Battles(name, date)
Outcomes(ship, battle, result)

a) Ingen klasse med 'bore' >18":

$$\pi_{class}(\sigma_{bore>18}(Classes)) \subseteq \emptyset$$

($\pi_{class}(\ldots)$ er unødvendig)

b) Hvis mer enn 10 våpen, må 'bore' <=15":

$$\pi_{class}(\sigma_{numGuns>10}(Classes)) \subseteq \pi_{class}(\sigma_{bore<=15}(Classes))$$

Alternativt:

$$\pi_{class}(\sigma_{numGuns>10 \text{ and } bore>15}(Classes)) \subseteq \emptyset$$

($\pi_{class}(\ldots)$ er i begge tilfellene unødvendig)

c) Ingen klasse mer enn tre skip:

$$\sigma_{noships>3}(\gamma_{class,\, count(name) \rightarrow noships}(Ships \bowtie Classes)) \subseteq \emptyset$$

d) Intet land både battleship og battlecruiser:

$$\pi_{country}(\sigma_{type='bb'}(Classes)) \cap \pi_{country}(\sigma_{type='bc'}(Classes)) \subseteq \emptyset$$

e) Intet skip med mer enn 10 våpen i slag med skip med færre enn 9 våpen og som ble senket:

skip med mer enn 10 våpen $\quad \rho_{s10(name1)}(\pi_{name}(Ships \bowtie \sigma_{numGuns>10}(Classes)))$

$\bowtie$ ← ② begrenser første skip til de med mer enn 10 våpen

slag $\quad \rho_{out10(name1,battle,result1)}(Outcomes)$

$\bowtie$ ← ① kobler sammen to og to skip i samme slag hvor ett skip er senket

senkede skip i slag $\quad \rho_{out9(name2,battle,result2)}(\sigma_{result='sunk'}(Outcomes))$

$\bowtie$ ← ③ begrenser andre skip til de med mindre enn 9 våpen

skip med mindre enn 9 våpen $\quad \rho_{s9(name2)}(\pi_{name}(Ships \bowtie \sigma_{numGuns<9}(Classes))) \subseteq \emptyset$ (ingen slike)

6.4.7

a) Antall bc-klasser:

```
select count(*)
from Classes
where type = 'bc';
```

b) Gjennomsnittlig bore for bb:

```
select sum(bore)/count(bore) as avgbore
from Classes
where type = 'bb';
```

(Denne går bare bra hvis count(bore) ≠ 0.)

c) Som b, men vekter med antall skip i klassen:

```
select sum(weight)/sum(noships) as avgweightedbore
from (select c.class, count(s.name) as noships, count(s.name)*max(bore) as weight
      from Ships s, Classes c
      where c.type = 'bb' and s.class = c.class
      group by c.class) as d;
```

NB! Det er ikke lov å ha beregning inni en aggregeringsfunksjon!
Ellers kunne vi gjort det slik (men det går altså ikke):

```
select sum(noships * bore)/sum(noships)
from (select c.class, count(s.name) as noships, max(bore) as bore
      from Ships s, Classes c
      where c.type = 'bb' and s.class = c.class
      group by c.class) as d;
```

NB2! Det er ikke lov å referere 'bore' uten å aggregere i en group by, selv om alle i hver gruppe har samme bore.
Et triks er å bruke max(bore) eller min(bore) for å få ut verdien.

6.4.7

d) Antall skip pr. klasse senket i slag:

```
select s.class, count(s.name) as nosunk
from Ships s, Outcomes o
where o.result = 'sunk' and o.ship = s.name
group by s.class;
```

e) Siste lanseringsår pr. klasse:

```
select class, max(launched) as lastlaunch
from Ships
group by class;
```

f) For hver klasse med minst to skip, antall senkede skip:

```
select s.class, count(s.name) as nosunk
from (select class
      from Ships
      group by class
      having count(name) >= 2) as c,
      Ships s, Outcomes o
where o.result = 'sunk' and o.ship = s.name and s.class = c.class
group by s.class;
```

(Burde hatt med 0 for de klassene med minst to skip og ingen senkede; det blir litt mer komplisert.)

Oppgave 2

$F = \{ DE \to AD, AB \to BC, BC \to CD, CD \to ABE \}$

1. Minimal overdekning:

a Splitter høyresidene (og fjerner trivielle FDer)

$DE \to A$

triviell $\cancel{DE \to D}$

-"- $\cancel{AB \to B}$

$AB \to C$

-"- $\cancel{BC \to C}$

$BC \to D$

$CD \to A$

$CD \to B$

$CD \to E$

b Minimale venstresider:

Alle venstresider er minimale:
siden det ikke er noen FD med
bare ett attributt i venstresiden.
F.eks. for $DE \to A$:

$D^+ = D$, $E^+ = E$,

så verken $D \to A$ eller $E \to A$
holder.

c Fjern overflødige FDer:

$DE^+ = DE$ i $F - \{DE \to A\}$

$AB^+ = AB$ i $F - \{AB \to C\}$

$BC^+ = BC$ i $F - \{BC \to D\}$

$CD^+ = CDBEA$ i $F - \{CD \to A\}$ $CD \to A$ er overflødig

$CD^+ = CDAE$ i $F - \{CD \to B\}$

$CD^+ = CDAB$ i $F - \{CD \to E\}$

Resultat:

$DE \to A$

$AB \to C$

$BC \to D$

$CD \to B$

$CD \to E$

2,

Elementær FD: $X \to A$ der

- A er ett attributt
- $X \to A$ er ikketriviell (dvs. $A \notin X$)
- X minimal (dvs. har ikke $Y \to A$ for noen ekte delmengde Y av X)

Alle FDene i den minimale overdekningen er elementære:

$DE \to A$

$AB \to C$

$BC \to D$

$CD \to B$

$CD \to E$

($A^+ = A$, $B^+ = B$, $C^+ = C$, $D^+ = D$, $E^+ = E$, så ingen venstreside kan gjøres mindre og fortsatt utgjøre en FD.)

3. Kandidatnøkler: Enhver kandidatnøkkel må bestå av minst to attributter siden alle venstresider har to attributter.

$DE^+ = DEA$

$AB^+ = ABCDE$ Kandidatnøkkel: AB

$BC^+ = BCDEA$ Kandidatnøkkel: BC

$CD^+ = CDBEA$ Kandidatnøkkel: CD

$AC^+ = AC$, $AD^+ = AD$, $AE^+ = AE$, $BD^+ = BD$, $BE^+ = BE$, $CE^+ = CE$, $ACE^+ = ACE$

Så

$AB \to C$, $BC \to D$, $CD \to B$, $CD \to E$

oppfyller kravet til BCNF. For

$DE \to A$

er venstresiden ikke en supernøkkel, så kravet til BCNF er brutt. Høyreside er et nøkkelattributt (nøkkelattributtene er $ABCD$), så kravet til 3NF er oppfylt. Nå er $AB \to C$ en elementær FD, så AB er en elementær kandidatnøkkel. Følgelig oppfyller $DE \to A$ kravet til EKNF.

Totalt er R derfor på EKNF.

4. Tapsfri dekomposisjon til BCNF:

$$R(A,B,C,D,E)$$

$$\swarrow \qquad \searrow \quad DE \to A \text{ bryter BCNF}$$

$$R1(A,D,E) \qquad R2(B,C,D,E)$$

5. Det fins ingen FD-bevarende dekomposisjon til BCNF. Hvis vi bruker algoritmen for FD-bevarende dekomposisjon (til EKNF eller bedre), får vi følgende relasjoner

$R_{AB} = \{\underline{A,B},C\}$ fra $AB \to C$

$R_{BC} = \{\underline{B,C},D\}$ fra $BC \to D$

$R_{CD} = \{B,\underline{C,D},E\}$ fra $CD \to B$, $CD \to E$. R_{CD} inneholder også FDen $BC \to D$.

$R_{DE} = \{A,\underline{D,E}\}$ fra $DE \to A$

$BC \to D$ i R_{CD} bryter BCNF. Så denne er på EKNF, men ikke BCNF. Hvis det hadde vært mulig å dekomponere tapsfritt og FD-bevarende til BCNF, ville denne algoritmen som vi her har brukt, gitt en slik dekomposisjon.

10.4.2

```
create type NameType as (
    lastname      char(40),
    middlename    char(40),
    firstname     char(20),
    title         char(20)
);

create type PersonType (
    name NameType,
    mother ref(PersonType),
    father ref(PersonType)
);

create type MarriageType (
    date char(6),
    husband ref(PersonType),
    wife ref(PersonType)
);
```

10.5.1

a) Navn på skuespillerne i Bride and Prejudice:

```
select s.star() -> name
from   StarsIn s
where  s.movie() -> title = 'Bride and Prejudice';
```

Jeg tror formelt sett parentesene skal være med i det ovenstående, men ifølge boka vil DBMSene stort sett godta at parentesene sløyfes. Dette gjøres derfor i det videre.

b) Alle filmer med Priety Zinta:

```
select s.movie -> title, s.movie -> year
from   StarsIn s
where  s.star -> name = 'Priety Zinta';
```

c) Tittel og år for filmer med minst seks skuespillere:

Jeg tror følgende burde gå:

```
select m.title, m.year
from   Movies m, StarsIn s
where  s.movie -> title = m.title and s.movie -> year = m.year
group by m.title, m.year
having count(*) > 6;
```

(Knytter Movies-objekter opp mot alle tilhørende StarsIn-objekter; grupperer og teller.)

10.5.1

d) Tittel og år for filmer hvor minst én skuespiller bor i Hyderabad:

```
select m.title, m.year
from Movies m
where exists ( select
               from StarsIn s
               where s.star -> address.city = 'Hyderabad'
                     and s.movie -> title = m.title
                     and s.movie -> year = m.year );
```

13.2.1

a) Diskkapasitet:

- 8 overflater
- 100 000 spor pr. overflate
- hvert spor i snitt 2000 sektorer à 1024 bytes

Så $8 \cdot 100\,000 = 800\,000$ spor, hvert i snitt $2000 \cdot 1024 = 2048000$ bytes, totalt $800\,000 \cdot 2048000$ bytes $= 1,6$ terabytes.

b) Maksimal søketid: Det er når diskhodene må flyttes på tvers av samtlige spor. Det er oppgitt at tiden det tar å flytte hodene n spor, er

$1 + 0,0003n$ millisekunder

Her er det 100 000 spor, så må flytte hodene 99999 spor; det gir

$1 + 0,0003 \cdot 99999 = 31$ ms

c) Maksimal rotasjonsforsinkelse: Det er hvis platene må rotere en gang før rett sektor er under diskhodet (hodet mistet akkurat såvidt den aktuelle sektoren).

Det er oppgitt at disken roterer 6000 rpm, så tiden for én runde er

$60/6000 = 0,01$ s $= 10$ ms

mente de $65536 = 1024 \cdot 64$?

d) Én blokk er 64 sektorer eller 65546 bytes. Overføringstid er tiden det tar før de ønskede data å passere under diskhodet.

Hvis sektorene ligger samlet langs ett spor, utgjør de i snitt

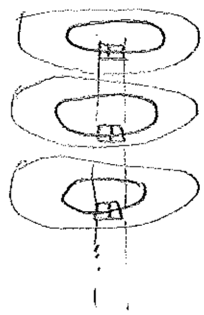
$64/2000$

av det totale sporet, så tiden det tar er

$(64/2000) \cdot 10\text{ms} = 0,32$ ms

Hvis sektorene ligger fordelt på samme sylinder, men forøvrig så tett som mulig, kan vi lese fra alle 8 overflatene i parallell. På hvert enkelt spor ligger i såfall $64/8 = 8$ sektorer, og tiden det tar er

$(8/2000) \cdot 10$ ms $= 0,04$ ms



Alternativ fremgangsmåte, alle bytene ligger "tett" langs ett spor: Kan lese

$$\underbrace{2000 \cdot \overbrace{1024}^{=1\text{Kb}}}_{\text{antall bytes pr. spor}} \cdot \underbrace{(6000/60)}_{\text{antall runder pr. sekund}} = 200\,000 \text{ Kb/s} = 200 \text{ Mb/s}.$$

Siden datamengden i det gitte eksempelet er $65546/1024 = 64$ Kb, tar det $64/200\,000 = 0,00032$ s å lese blokken.

13.4.5

RAID 4

a)
$$
\begin{array}{r}
01010110 \\
\oplus\ 11000000 \\
\oplus\ 00101011 \\
\oplus\ 10111011 \\
\hline
= 00000110
\end{array}
$$

b)
$$
\begin{array}{r}
11110000 \\
\oplus\ 11111000 \\
\oplus\ 00111100 \\
\oplus\ 01000001 \\
\hline
= 01110101
\end{array}
$$

Oppgave 2

d1	0	0	1
d2	0	1	0
d4	1	0	0
d3	0	1	1
d5	1	0	1
d6	1	1	0
d7	1	1	1

a)
$$
d1 = d3 \oplus d5 \oplus d7 = \begin{array}{r}
00001111 \\
\oplus\ 00000100 \\
\oplus\ 11000101 \\
\hline
= 11001110
\end{array}
$$

$$
d2 = d3 \oplus d6 \oplus d7 = \begin{array}{r}
00001111 \\
\oplus\ 10011011 \\
\oplus\ 11000101 \\
\hline
= 01010001
\end{array}
$$

$$
d4 = d5 \oplus d6 \oplus d7 = \begin{array}{r}
00000100 \\
\oplus\ 10011011 \\
\oplus\ 11000101 \\
\hline
= 01011010
\end{array}
$$

b) Siden d2 og d4 er konstruert på grunnlag av d6, må – i tillegg til endringen på d6 – disse diskene endres. Nytt innhold:

$$
\text{ny } d2 = \text{gammel } d2 \oplus \text{gammel } d6 \oplus \text{ny } d6 = \begin{array}{r}
01010001 \\
\oplus\ 10011011 \\
\oplus\ 10000100 \\
\hline
= 01001110
\end{array}
$$

$$
\text{ny } d4 = \text{gammel } d4 \oplus \text{gammel } d6 \oplus \text{ny } d6 = \begin{array}{r}
01011010 \\
\oplus\ 10011011 \\
\oplus\ 10000100 \\
\hline
= 01000101
\end{array}
$$

I praksis kan man først beregne

gammel d6 $\oplus$ ny d6

og legge denne til både d2 og d4.

Oppgave 2 (forts)

c) d5 kan rekonstrueres på ett av følgende vis:

$d4 \oplus d6 \oplus d7$

$d1 \oplus d3 \oplus d7$

d) d6 kan rekonstrueres fra $d4 \oplus d5 \oplus d7$ eller fra $d2 \oplus d3 \oplus d7$.
Men siden d7 også er korrupt, går ikke dette.

d7 kan rekonstrueres fra $d4 \oplus d5 \oplus d6$, $d2 \oplus d3 \oplus d6$ eller $d1 \oplus d3 \oplus d5$.
Siden d6 er korrupt, bruker vi den siste: $d1 \oplus d3 \oplus d5$.

Når d7 er rekonstruert, kan vi rekonstruere også d6, fra f.eks. $d4 \oplus d5 \oplus d7$.