IN2090

GJENNOMGANG AV TIDLIGERE EKSAMENSOPPGAVER

EKSAMEN 2020 - SQL

poi

pid	navn	adresse	bid	type
1	Linken		3	Utsiktspunkt
2	Slottet	Karl Johansgate 1	1	Slott
3	Eiffeltårnet	5 avenue Anatole	6	Kulturminne
4	Lyse Kloster	Edne 121	3	Kulturminne
5	Berlinmuren		8	Kulturminne
6	Maemo	Dronning Eufemias gate 23	1	Restaurant
7	Kaf Kafe	Jacobsfjorden 4	3	Kafé
8	Kilden	Sjølystveien 2	2	Konserthus
9	London Bro	London SE1 9RA	13	Bro
10	Frihetsgudinnen	NY 10004	14	Kulturminne
11	Feffo	Via Delle Fornaci 2/4/6	4	Kafé
12	Frantzén	Klara Norra kyrkogata	11	Restaurant
13	Aker Brygge	Støperigata 2	1	Brygge
14	Caracallas Termer	Viale delle Terme di Caracalla	4	Kulturminne
15	Grefsenkollen Restaurant	Grefsenkollveien 100	1	Restaurant
16	Villa Bardini	Costa S. Giorgio, 2-4	4	Kulturminne

land

lid	navn
1	Norge
2	Frankrike
3	Italia
4	Tyskland
5	Sverige
6	Storbritannia
7	USA

by

bid	navn	lid
1	Oslo	1
2	Kristiansand	1
3	Bergen	1
4	Roma	3
5	Firenze	3
6	Paris	2
7	Lyon	2
8	Berlin	4
9	Bremen	4
10	Hamburg	4
11	Stockholm	5
12	Gøteborg	5
13	London	6
14	New York	7

værmelding

bid	dato	nedbør	vind
1	2020-12-17	1.1	2
1	2020-12-18	0	1
2	2020-12-17	3.4	5
2	2020-12-19	2.1	2
3	2020-12-17	10.7	8
3	2020-12-18	11.2	9
4	2020-12-17	0	1
4	2020-12-20	0	2
5	2020-12-17	1.3	6
5	2020-12-19	0.1	2
6	2020-12-17	0	2
8	2020-12-17	0.1	0
7	2020-12-21	5.4	6
3	2020-12-19	21.2	6
14	2020-12-24	5.1	4
14	2020-12-23	1.2	1
10	2020-12-17	3.1	0
9	2020-12-31	0	0
1	2020-12-19	0.1	3
13	2020-12-18	0.7	6

1 Flervalg

Dersom dataene i databasen er lik eksempeldataene i PDFen, hvilke av følgende tupler vil være med i resultatet på følgende spørring:

```
SELECT by.navn AS by, poi.navn AS land
FROM by INNER JOIN poi USING (bid)
WHERE by.navn LIKE '%r%i%' OR
      poi.navn LIKE '%er';
```

1. (Berlin, Berlinmuren)
2. (Oslo, Slottet)
3. (Kristiansand, Kilden)
4. (Paris, Eiffeltårnet)
5. (Oslo, Bentsebrua)
6. (London, London Bro)
7. (Berlin, Frihetsgudinnen)
8. (Oslo, Aker Brygge)
9. (Bergen, Lyse Kloster)
10. (Roma, Caracallas Termer)
11. (Firenze, Villa Bardini)
12. (Kristiansand, Maemo)

Løsningsforslag

1, 3, 4, 9, 10

by

bid	navn	lid
1	Oslo	1
2	Kristiansand	1
3	Bergen	1
4	Roma	3
5	Firenze	3
6	Paris	2
7	Lyon	2
8	Berlin	4
9	Bremen	4
10	Hamburg	4
11	Stockholm	5
12	Gøteborg	5
13	London	6
14	New York	7

poi

pid	navn	adresse	bid	type
1	Linken		3	Utsiktspunkt
2	Slottet	Karl Johansgate 1	1	Slott
3	Eiffeltårnet	5 avenue Anatole	6	Kulturminne
4	Lyse Kloster	Edne 121	3	Kulturminne
5	Berlinmuren		8	Kulturminne
6	Maemo	Dronning Eufemias gate 23	1	Restaurant
7	Kaf Kafe	Jacobsfjorden 4	3	Kafé
8	Kilden	Sjølystveien 2	2	Konserthus
9	London Bro	London SE1 9RA	13	Bro
10	Frihetsgudinnen	NY 10004	14	Kulturminne
11	Feffo	Via Delle Fornaci 2/4/6	4	Kafé
12	Frantzén	Klara Norra kyrkogata	11	Restaurant
13	Aker Brygge	Støperigata 2	1	Brygge
14	Caracallas Termer	Viale delle Terme di Caracalla	4	Kulturminne
15	Grefsenkollen Restaurant	Grefsenkollveien 100	1	Restaurant
16	Villa Bardini	Costa S. Giorgio, 2-4	4	Kulturminne

2 Italiensk værmelding

Skriv en SQL-spørring som finner morgendagens (17.12.2020) værmelding for alle byer i Italia. Spørringen skal skrive ut navnet på byen, antall millimeter regn og vindstyrken. Sorter resultatet alfabetisk på bynavn.

```sql
SELECT b.navn, v.nedbør, v.vind
FROM land AS l
     INNER JOIN by AS b USING (lid)
     INNER JOIN værmelding AS v USING (bid)
WHERE l.navn = 'Italia' AND v.dato = '2020-12-17'
ORDER BY b.navn;
```

land

lid	navn
1	Norge
2	Frankrike
3	Italia
4	Tyskland
5	Sverige
6	Storbritannia
7	USA

by

bid	navn	lid
1	Oslo	1
2	Kristiansand	1
3	Bergen	1
4	Roma	3
5	Firenze	3
6	Paris	2
7	Lyon	2
8	Berlin	4
9	Bremen	4
10	Hamburg	4
11	Stockholm	5
12	Gøteborg	5
13	London	6
14	New York	7

værmelding

bid	dato	nedbør	vind
1	2020-12-17	1.1	2
1	2020-12-18	0	1
2	2020-12-17	3.4	5
2	2020-12-19	2.1	2
3	2020-12-17	10.7	8
3	2020-12-18	11.2	9
4	2020-12-17	0	1
4	2020-12-20	0	2
5	2020-12-17	1.3	6
5	2020-12-19	0.1	2
6	2020-12-17	0	2
8	2020-12-17	0.1	0
7	2020-12-21	5.4	6
3	2020-12-19	21.2	6
14	2020-12-24	5.1	4
14	2020-12-23	1.2	1
10	2020-12-17	3.1	0
9	2020-12-31	0	0
1	2020-12-19	0.1	3
13	2020-12-18	0.7	6

3 Ukesmelding for jula

Skriv en SQL-spørring som finner navn, totalt antall millimeter nedbør og gjennomsnittlig vindstyrke i romjulsuka (altså fra og med dato 24.12.2020 til og med dato 31.12.2020) for hver by.

```sql
SELECT b.navn, sum(v.nedbør) AS sum_nedbør, avg(v.vind) AS avg_vind
FROM by AS b INNER JOIN værmelding AS v USING (bid)
WHERE v.dato >= '2020-12-24' AND v.dato <= '2020-12-31'
GROUP BY b.bid, b.navn;
```

værmelding

bid	dato	nedbør	vind
1	2020-12-17	1.1	2
1	2020-12-18	0	1
2	2020-12-17	3.4	5
2	2020-12-19	2.1	2
3	2020-12-17	10.7	8
3	2020-12-18	11.2	9
4	2020-12-17	0	1
4	2020-12-20	0	2
5	2020-12-17	1.3	6
5	2020-12-19	0.1	2
6	2020-12-17	0	2
8	2020-12-17	0.1	0
7	2020-12-21	5.4	6
3	2020-12-19	21.2	6
14	2020-12-24	5.1	4
14	2020-12-23	1.2	1
10	2020-12-17	3.1	0
9	2020-12-31	0	0
1	2020-12-19	0.1	3
13	2020-12-18	0.7	6

by

bid	navn	lid
1	Oslo	1
2	Kristiansand	1
3	Bergen	1
4	Roma	3
5	Firenze	3
6	Paris	2
7	Lyon	2
8	Berlin	4
9	Bremen	4
10	Hamburg	4
11	Stockholm	5
12	Gøteborg	5
13	London	6
14	New York	7

4 Byer uten regn

Skriv en SQL-spørring som finner navn på alle byer hvor det ikke er meldt noe regn og ikke noe vind fra og med julaften (24.12.2020) og ut året. Spørringen skal skrive ut navnet på byene. Du kan anta at vi har værmelding (og både nedbør og vindstyrke) for alle byer hver dag i julen.

```sql
SELECT navn
FROM by
WHERE bid NOT IN (
  SELECT bid
  FROM værmelding
  WHERE dato >= '2020-12-24' AND
        dato < '2021-01-01' AND
        (nedbør > 0 OR vind > 0)
);
```

```sql
SELECT b.navn
FROM by AS b INNER JOIN værmelding AS v USING (bid)
WHERE dato >= '2020-12-24' AND
      dato < '2021-01-01'
GROUP BY b.bid, b.navn
HAVING sum(v.nedbør) = 0 AND sum(v.vind) = 0;
```

værmelding

bid	dato	nedbør	vind
1	2020-12-17	1.1	2
1	2020-12-18	0	1
2	2020-12-17	3.4	5
2	2020-12-19	2.1	2
3	2020-12-17	10.7	8
3	2020-12-18	11.2	9
4	2020-12-17	0	1
4	2020-12-20	0	2
5	2020-12-17	1.3	6
5	2020-12-19	0.1	2
6	2020-12-17	0	2
8	2020-12-17	0.1	0
7	2020-12-21	5.4	6
3	2020-12-19	21.2	6
14	2020-12-24	5.1	4
14	2020-12-23	1.2	1
10	2020-12-17	3.1	0
9	2020-12-31	0	0
1	2020-12-19	0.1	3
13	2020-12-18	0.7	6

by

bid	navn	lid
1	Oslo	1
2	Kristiansand	1
3	Bergen	1
4	Roma	3
5	Firenze	3
6	Paris	2
7	Lyon	2
8	Berlin	4
9	Bremen	4
10	Hamburg	4
11	Stockholm	5
12	Gøteborg	5
13	London	6
14	New York	7

5 Steder og vær

Skriv en SQL-kommando som lager et VIEW med navn Steder som viser dagens værmelding (nedbør i mm. og vindstyrke) for både byer og POIs i samme tabell.

Du kan anta at dagens dato finnes i variabelen current_date (slik som i PostgreSQL). VIEWet skal ha 4 kolonner, en med navnet på stedet, en med plassering som er landet dersom stedet er en by og adressen dersom stedet er en POI, samt nedbør og vindstyrke. For POIs er nedbør og vindstyrke lik byen den befinner seg i sin nedbør og vindstyrke. Vi er kun interessert i steder som faktisk har en posisjon, altså skal posisjon aldri være NULL. F.eks. kan innholdet i VIEWet se slik ut:

navn	posisjon	nedbør	vind
Oslo	Norge	1.1	2
Kilden	Sjølystveien 2	3.4	5
Eiffeltårnet	5 avenue Anatole	0	2
Grefsenkolloen Restaurant	Grefsenkollveien 100	1.1	2
Villa Bardini	Costa S. Giorgio, 2-4	0	1
Berlin	Tyskland	0.1	0
Kristiansand	Norge	3.4	5
Aker Brygge	Støperigata 2	1.1	2
Roma	Italia	0	1
Kaf Kafe	Jacobsfjorden 4	10.7	8
Feffo	Via Delle Fornaci 2/4/6	0	1
Maemo	Dronning Eufemias gate 23	1.1	2
Firenze	Italia	1.3	6
Bergen	Norge	10.7	8
Lyse Kloster	Edne 121	10.7	8
Paris	Frankrike	0	2
Caracallas Termer	Viale delle Terme di Caracalla	0	1
Hamburg	Tyskland	3.1	0
Slottet	Karl Johansgate 1	1.1	2

```sql
CREATE VIEW steder AS
SELECT b.navn, l.navn AS posisjon, v.nedbør, v.vind
FROM land AS l INNER JOIN by AS b USING (lid)
     INNER JOIN værmelding AS v USING (bid)
WHERE v.dato = current_date -- l.navn er markert med NOT NULL
UNION ALL -- også greit med UNION
SELECT p.navn, p.adresse AS posisjon, v.nedbør, v.vind
FROM poi AS p INNER JOIN by AS b USING (bid)
     INNER JOIN værmelding AS v USING (bid)
WHERE v.dato = current_date AND p.adresse IS NOT NULL;
```

6 Ferieplanlegging

La oss si at du skal reise på ferie til Frankrike og ønsker å finne ut hvilken by du skal reise til for å gå tur og se på museer. Skriv derfor en SQL-spørring som finner den byen i Frankrike me opphold (0 mm. nedbør) i morgen (17.12.2020), som har minst 3 kaféer og som har flest museer.

```sql
WITH
  opphold_og_kafeer AS (
    SELECT b.bid, b.navn
    FROM land AS l
         INNER JOIN by AS b USING (lid)
         INNER JOIN værmelding AS v USING (bid)
         INNER JOIN poi AS p USING (bid)
    WHERE l.navn = 'Frankrike' AND
          v.dato = '2020-12-17' AND
          v.nedbør = 0 AND
          p.type = 'Kafé'
    GROUP BY b.bid, b.navn
    HAVING count(*) >= 3
  )
SELECT o.navn, count(*) AS antall_museer
FROM opphold_og_kafeer AS o INNER JOIN poi AS p USING (bid)
WHERE p.type = 'Museum'
GROUP BY o.bid, o.navn
ORDER BY antall_museer DESC
LIMIT 1;
```

```sql
WITH
  opphold_og_kafeer AS (
    SELECT b.bid, b.navn
    FROM land AS l
         INNER JOIN by AS b USING (lid)
         INNER JOIN værmelding AS v USING (bid)
         INNER JOIN poi AS p USING (bid)
    WHERE l.navn = 'Frankrike' AND
          v.dato = '2020-12-17' AND
          v.nedbør = 0 AND
          p.type = 'Kafé'
    GROUP BY b.bid, b.navn
    HAVING count(*) >= 3
  ),
  antall_museer AS (
    SELECT b.bid, count(*) AS antall_museer
    FROM by AS b
         INNER JOIN land AS l USING (lid)
         INNER JOIN poi AS p USING (bid)
    WHERE p.type = 'Museum' AND l.navn = 'Frankrike'
    GROUP BY b.bid, b.navn
  ),
  flest_museer AS (
    SELECT *
    FROM antall_museer
    WHERE antall_museer = (SELECT max(antall_museer) FROM antall_museer)

  )
SELECT ok.navn, fm.antall_museer
FROM oppholde_og_kafeer AS ok
     INNER JOIN flest_museer AS fm USING (bid);
```

EKSAMEN 2020 - Tapsfri dekomposisjon

Gitt følgende relasjon **Student**(id, navn, veileder, oppgave, institutt)

med kandidatnøkler {id, oppgave} og FDer

1. id -> navn
2. oppgave -> veileder
3. oppgave -> institutt

Dekomponer relasjonen tapsfritt til BCNF. Beskriv hvordan du kommer frem til resultatet, og list opp nøkler og FDer for alle tabeller underveis.

Tapsfri dekomposisjon

Student(id, navn, veileder, oppgave, institutt)
kandidatnøkel {id, oppgave}

FDer
1. id -> navn
2. oppgave -> veileder
3. oppgave -> institutt

- Dekomponerer med hensyn på
  id -> navn(id ikke supernøkkel så brudd på BCNF) først og får
  - **T1(id, navn)** og
  - **T2(id, veileder, oppgave, institutt).**
- For T1
  holder kun den første FDen, og har dermed kandidatnøkkel id, og er dermed på BCNF.
- For T2 holder oppgave -> veileder og oppgave -> institutt
  og har dermed kandidatnøkkel {id, oppgave}.
- Den første av de to FDene bryter med BCNF. Dekomponerer T2 videre og får
  - **T21(oppgave, veileder, institutt)** og
  - **T22(id, oppgave).**
- For T21 holder FDene 2. og 3. og har da kandidatnøkkel oppgave og er den dermed på BCNF.
- For T22 holder ingen av FDene, og den er dermed også på BCNF.
  Student kan dermed dekomponeres tapsfritt til
  T1(id, navn), T21(oppgave, veileder, institutt) og **T22(id, oppgave)**.

PRØVEEKSAMEN 2020 - SQL

barn

bid	navn	snill
0	Ola	t
1	Kari	t
2	Per	f
3	Nils	t
4	Mari	f
5	Kine	f
6	Jasmin	t

gave

gid	navn	nyttig
0	Lekebil	f
1	Sokker	t
2	Sykkel	t
3	Lekepistol	f
4	Dataspill	f
5	Fuglebrett	t
6	Dukke	f
7	Bok	t

ønskeliste

barn	gave
1	0
1	6
0	6
0	5
3	2
5	1
5	3
5	4
4	3
4	0
6	7

5 Snille barn

Skriv en SQL-kommando som oppdaterer barnet med bid lik 0 sin snill-verdi til false.

```
UPDATE barn
SET snill = false
WHERE bid = 0;
```

6 Nyttige gaver

Skriv en spørring som finner alle barn som ønsker seg nyttige gaver som har navn som starter med strengen 'Sokker'. Spørringen skal returnere navnet på barnet og navnet på gaven.

```
SELECT b.navn AS barn, g.navn AS gave
FROM barn AS b
     INNER JOIN ønskeliste AS ø ON (b.bid = ø.barn)
     INNER JOIN gave AS g ON (ø.gave = g.gid)
WHERE g.nyttig AND g.navn LIKE 'Sokker%';
```

7 Oversikts-VIEW

Skriv en SQL-kommando som lager et VIEW som heter oversikt og inneholder én rad for hvert ønske med bid og navnet på barnet som har ønsket, gid og navnet på gaven som er ønsket, hvorvidt barnet har vært snill, og hvorvidt gaven er nyttig. VIEWet skal være sortert først på barnets navn, og så på gavens navn i alfabetisk rekkefølge. Innholdet i VIEWet kan f.eks. se slik ut:

bid	barn	gid	gave	snill	nyttig
6	Jasmin	7	Bok	t	t
1	Kari	6	Dukke	t	f
1	Kari	0	Lekebil	t	f
5	Kine	4	Dataspill	f	f
5	Kine	3	Lekepistol	f	f
5	Kine	1	Sokker	f	t
4	Mari	0	Lekebil	f	f
4	Mari	3	Lekepistol	f	f
3	Nils	2	Sykkel	t	t
0	Ola	6	Dukke	t	f
0	Ola	5	Fuglebrett	t	t

```sql
CREATE VIEW oversikt AS
SELECT b.bid, b.navn AS barn, g.gid, g.navn AS gave, b.snill, g.nyttig
FROM barn AS b
     INNER JOIN ønskeliste AS ø ON (b.bid = ø.barn)
     INNER JOIN gave AS g ON (ø.gave = g.gid)
ORDER BY barn, gave;
```

8 Like ønkser

Skriv en SQL-spørring som finner alle par av ulike barn som har ønsket seg det samme. Svaret skal kun inneholde unike rader. Du kan benytte VIEWet fra oppgave 7 om du ønsker.

```
WITH
  likt_ønske AS (
    SELECT b1.barn AS barn1, b2.barn AS barn2
    FROM ønskeliste AS b1
         INNER JOIN ønskeliste AS b2 USING (gave)
    WHERE b1.barn != b2.barn
  )
SELECT DISTINCT b1.navn, b2.navn
FROM likt_ønske AS l
     INNER JOIN barn AS b1 ON (l.barn1 = b1.bid)
     INNER JOIN barn AS b2 ON (l.barn2 = b2.bid);
```

eller med VIEWet fra oppgave 7:

```
SELECT DISTINCT b1.barn, b2.barn
FROM oversikt AS b1
     INNER JOIN oversikt AS b2 USING (gid)
WHERE b1.bid != b2.bid;
```

```
CREATE VIEW oversikt AS
SELECT b.bid, b.navn AS barn, g.gid, g.navn AS gave, b.snill, g.nyttig
FROM barn AS b
     INNER JOIN ønskeliste AS ø ON (b.bid = ø.barn)
     INNER JOIN gave AS g ON (ø.gave = g.gid)
ORDER BY barn, gave;

SELECT DISTINCT b1.barn, b2.barn
FROM oversikt AS b1
     INNER JOIN oversikt AS b2 USING (gid)
WHERE b1.bid != b2.bid;
```

9 Populære gaver

Skriv en spørring som finner de tre mest populære nyttige gavene, og de tre mest populære unyttige gavene. Resultatet skal inneholde navn på gaven, antall barn som ønsker gaven, samt hvorvidt den er nyttig eller ikke. Du kan benytte view'et fra oppgave 7.

```sql
SELECT gave, count(*) AS antall_ønsker, true AS nyttig
FROM oversikt
WHERE nyttig
GROUP BY gave
ORDER BY antall_ønsker DESC
LIMIT 3
UNION
SELECT gave, count(*) as antall_ønsker, false AS nyttig
FROM oversikt
WHERE NOT nyttig
GROUP BY gave
ORDER BY antall_ønsker DESC
LIMIT 3;
```

10 Gaveliste

Skriv en spørring som tilordner gaver til barn i henhold til følgende regler:

- Snille barn får alt de ønsker seg
- Usnille får kun nyttige ting de ønsker seg. Om de ikke ønsker seg noen nyttige ting får de gaven med navn 'Genser'.

Spørringen skal skrive ut navnet på barnet og navnet på gaven. Merk, du kan bruke view'et fra oppgave 7.

```
WITH
  usnille_med_ønsker AS (
    SELECT bid, barn, gave
    FROM oversikt
    WHERE snill = false AND
          nyttig = true
  ),
  usnille_uten_ønkser AS (
    SELECT navn AS barn, 'Genser' AS gave
    FROM barn
    WHERE snill = false AND
          bid NOT IN (SELECT bid FROM usnille_med_ønsker)
  )
SELECT barn, gave
FROM oversikt
WHERE snill = true
UNION ALL
SELECT barn, gave
FROM usnille_med_ønsker
UNION ALL
SELECT barn, gave
FROM usnille_uten_ønsker;
```

PRØVEEKSAMEN 2020 - Relasjonsalgebra

I denne oppgaven skal du bruke samme databaseskjame som oppgavene i SQL. Sriv et uttrykk i relasjonsalgebraen som finner navn på alle barn som ønsker seg hoppestokk.

$$\pi_{\text{navn}}(\text{barn} \bowtie_{\text{bid}} \text{ønskeliste} \bowtie_{\text{gid}} \pi_{\text{gid}}(\sigma_{\text{navn='hoppestokk'}}(\text{gave})))$$

eller

$$\pi_{\text{navn}}(\text{barn} \bowtie_{\text{bid}} \text{ønskeliste} \bowtie_{\text{gid}} \rho_{\text{navn}\to\text{gavenavn}}(\sigma_{\text{navn='hoppestokk'}}(\text{gave})))$$

PRØVEEKSAMEN 2020 - Tapsfri dekomponering

Gitt følgende relasjon R(A, B, C, D, E, F)
med kandidatnøkler AB og CD, og FDer:

1. $AB \rightarrow C$
2. $AB \rightarrow D$
3. $CD \rightarrow A$
4. $CD \rightarrow B$
5. $A \rightarrow E$
6. $C \rightarrow F$

Dekomponer relasjonen tapsfritt til BCNF. Vis hvordan du kommer frem til svaret. For hver relasjon du får underveis i dekomponeringen, list opp hvilke FDer som holder, og hvilke kandidatnøkler relasjonen har

Tapsfri dekomponering

R(A, B, C, D, E, F)
kandidatnøkler AB og CD, og FDer:

1. AB → C
2. AB → D
3. CD → A
4. CD → B
5. A → E
6. C → F

- Vi går igjennom hver FD og sjekker om de bryter med BCNF:
  1. AB → C: AB er supernøkkel så bryter ikke med BCNF.
  2. AB → D: AB er supernøkkel så bryter ikke med BCNF.
  3. CD → A: CD er supernøkkel så bryter ikke med BCNF.
  4. CD → B: CD er supernøkkel så bryter ikke med BCNF.
  5. A → E: A er ikke en supernøkkel så bryter med BCNF.
- Bruker FD nummer 5. Dekomponerer R til
  - **S1(A, E)** og
  - **S2(A, B, C, D, F)**
- For S1 holder kun FDen A → E og har dermed kandidatnøkkel A, og er dermed på BCNF.
- For S2 holder alle de andre FDene og får dermed kandidatnøkler AB og CD. Må derfor (muligens) dekomponere S2 videre. Sjekker defor S2 videre med C → F: C, er for S2 ikke en supernøkkel, må derfor dekomponere S2 videre. Dekomponerer S2 til
  - **S21(C, F)** og
  - **S22(C, A, B, D).**
- For S21 holder kun C → F og har da kandidatnøkkel C og er derfor på BCNF.
- For S22 holder FDene 1. til 4., og har derfor kandidatnøkler AB og CD. Av samme grunn som over bryter ingen av disse BCNF, og S22 er på BCNF.
- R kan altså dekomponeres tapsfritt til **S1(A, E)**, **S21(C, F)** og **S22(A, B, C, D)**.

LYKKE TIL PÅ EKSAMEN!