TDT4145 - Øving 3

Sander Lindberg

1 Oppgave 1

1.1 a

Dette kan gjøres med ON DELETE CASCADE

1.2 b

```
USE film;

CREATE TABLE regissør(
regissørID INTEGER NOT NULL,
navn VARCHAR(255),
 CONSTRAINT regissør_PK PRIMARY KEY (regissørID)
);

CREATE TABLE film (
filmID INTEGER NOT NULL,
title VARCHAR(30),
prodaar INTEGER,
regissorID INTEGER NOT NULL,
CONSTRAINT film_PK PRIMARY KEY (filmID),
CONSTRAINT film_FK FOREIGN KEY (regissorID)
REFERENCES regissør (regissørID),
CONSTRAINT aarsjekk CHECK (prodaar < YEAR(GETDATE()))
);

CREATE TABLE sjanger (
sjangerID INTEGER NOT NULL,
navn VARCHAR(20) NOT NULL,
beskrivelse VARCHAR(255),
CONSTRAINT sjanger_PK PRIMARY KEY (sjangerID)
```

```
);

CREATE TABLE sjangerforfilm (
filmID INTEGER NOT NULL,
sjangerID INTEGER NOT NULL,
CONSTRAINT sjangerforfilm_PK PRIMARY KEY (filmID, sjangerID),
CONSTRAINT sjangerforfilm_FK1 FOREIGN KEY (filmID) REFERENCES film (filmID)
ON DELETE CASCADE,
CONSTRAINT sjangerforfilm_FK2 FOREIGN KEY (sjangerID) REFERENCES sjanger (sjangerID)
);

CREATE TABLE skuespiller (
skuespillerID INTEGER NOT NULL,
navn VARCHAR(255),
faar INTEGER NOT NULL,
CONSTRAINT skuespiller_PK PRIMARY KEY (skuespillerID),
CONSTRAINT aarsjekk CHECK (faar < YEAR(getdate()))
);

CREATE TABLE skuespillerifIlm (
filmID INTEGER NOT NULL,
skuespillerID INTEGER NOT NULL,
rolle VARCHAR(255),
CONSTRAINT skuespillerifilm_PK PRIMARY KEY (filmID, skuespillerID),
CONSTRAINT skuespillerifilm_FK1 FOREIGN KEY (filmID) REFERENCES film (filmID)
ON DELETE CASCADE,
CONSTRAINT skuespillerifilm_FK2 FOREIGN KEY (skuespillerID) REFERENCES skuespiller
(skuespillerID)
);
```

1.3 c

```
INSERT INTO regissør VALUES (1, "Python Reed"), (2, "Tom Shadyac");

INSERT INTO film VALUES (1, "Yes Man", 2008, 1);

INSERT INTO skuespiller VALUES (1 , "Jim Carrey", 1962);

INSERT INTO skuespillerifilm VALUES (1, 1, "Carl")
```

1.4 d

```
UPDATE skuespiller SET navn = "James Eugene Carrey" WHERE skuespillerID = 1;
```

1.5 e

```
DELETE FROM regissør where regissørID = 2;
```

2 Oppgave 2

2.1 a

```
SELECT * FROM film;
```

2.2 b

```
SELECT navn FROM skuespiller where faar > 1960;
```

2.3 c

```
SELECT * FROM skuespiller WHERE faar BETWEEN 1980 AND 1989 ORDER BY navn ASC;
```

2.4 d

```
SELECT film.title, skuespillerifilm.rolle FROM film
JOIN skuespillerifilm ON film.filmID = skuespillerifilm.filmID
JOIN skuespiller ON skuespiller.skuespillerID = skuespillerifIlm.skuespillerID
WHERE navn = "Morgan Freeman";
```

2.5 e

```
SELECT DISTINCT  title
FROM film
JOIN regissør ON film.regissorID = regissør.regissørID J
OIN skuespillerifIlm ON film.filmID = skuespillerifilm.filmID
JOIN skuespiller ON skuespillerifilm.skuespillerID = skuespiller.skuespillerID
WHERE skuespiller.navn = regissør.navn;
```

2.6 f

```
SELECT DISTINCT COUNT(navn) as 'Antall_navn_c' FROM skuespiller WHERE navn
LIKE "C%";
```

2.7 g

```
SELECT DISTINCT sjanger.navn, COUNT(film.filmID)
FROM sjangerforfilm
JOIN film ON sjangerforfilm.filmID = film.filmID
JOIN sjanger ON sjangerforfilm.sjangerID = sjanger.sjangerID
GROUP BY sjanger.navn;
```

2.8 h

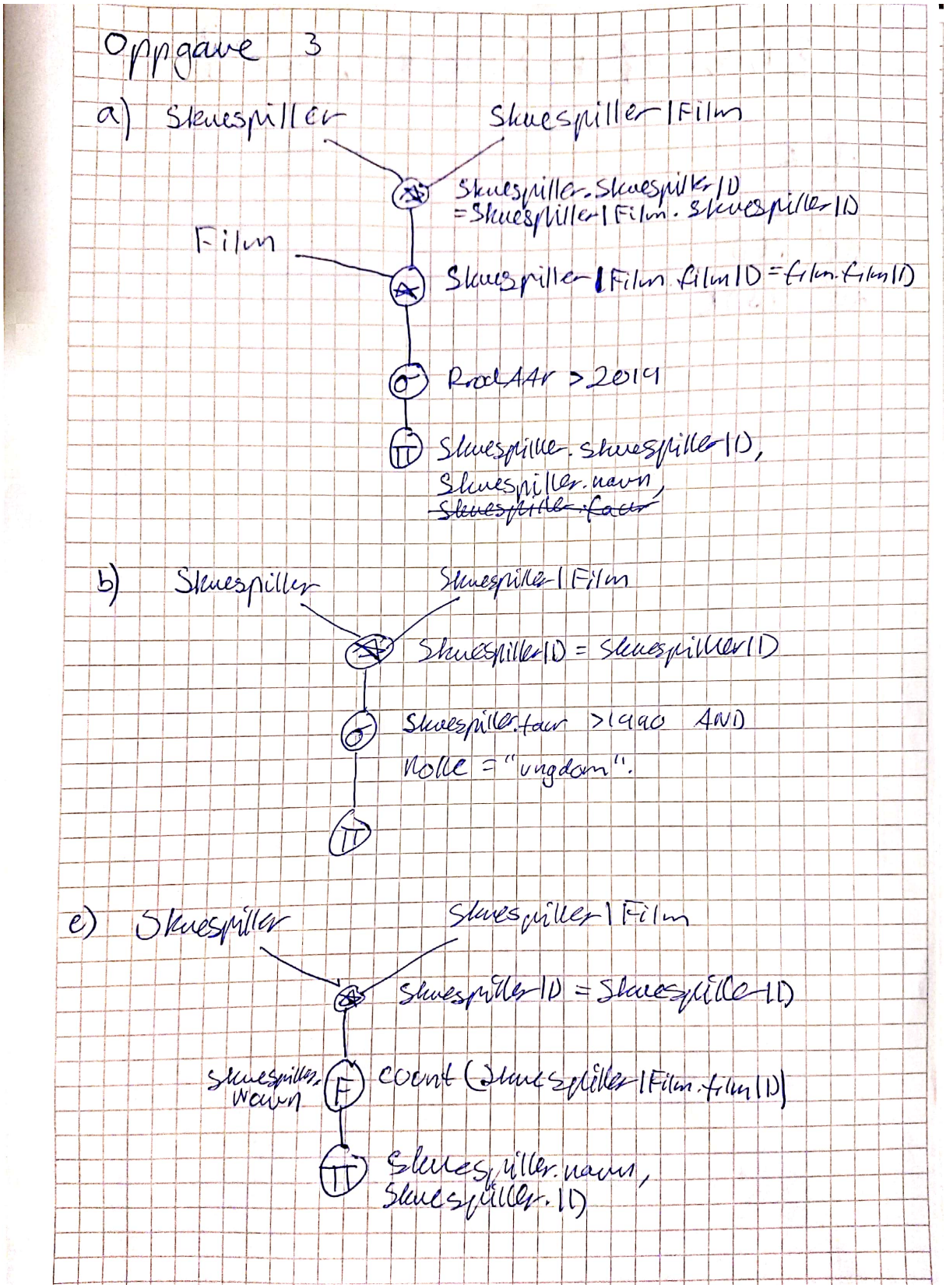
```
SELECT skuespiller.navn
```

```
FROM skuespiller JOIN skuespillerifilm ON skuespiller.skuespillerID
= skuespillerifilm.skuespillerID
JOIN film ON film.filmID = skuespillerifilm.filmID
WHERE film.title = "Ace Ventura: Pet Detective"
AND skuespiller.skuespillerID NOT IN (
SELECT skuespiller.skuespillerID
FROM skuespiller JOIN skuespillerifilm ON skuespiller.skuespillerID
= skuespillerifilm.skuespillerID
JOIN film ON film.filmID = skuespillerifilm.filmID
WHERE film.title = "Ace Ventura: When Nature Calls");
```

2.9 i

```
SELECT film.title, AVG(skuespiller.faar) AS Average_age_movie
FROM film JOIN skuespillerifilm ON skuespillerifilm.filmID = film.filmID
JOIN skuespiller ON skuespiller.skuespillerID = skuespillerifilm.skuespillerID
GROUP BY film.filmID
HAVING Average_age_movie >
(SELECT AVG(skuespiller.faar) FROM film JOIN skuespillerifilm
ON skuespillerifilm.filmID = film.filmID
JOIN skuespiller ON skuespiller.skuespillerID = skuespillerifilm.skuespillerID);
```

3 Oppgave 3



Figur 1: Oppgave 3 - Relasjonsalgebra

4 Oppgave 4

4.1 a

3 rader, 15 kolonner.

4.2 b

Ville satt fakultet i en egen tabell:

Fakultet(fakultetkode, fakultetsnavn, fakultetsbygg)

Og ansatt i en annen:

Ansatt(PersonID, Navn, Telefonnr, fakultetkode)

Her har jeg satt fakultetkode i ansatttabellen, da en ansatt kan kun ha et fakultet, mens fakulteter kan ha flere ansatte.

5 Oppgave 4

5.1 a

1. Ja
2. Kan kanskje stemme, da alle a'er peker på unike verdier av $b1$. Altså, a_1 peker kun på b_1, a_2 peker kun på b_1 osv...Vet ikke om dette er tilfelle i "resten" av tabellen.
3. Nei. a_1 peker på både c_1 og c_2.
4. Nei, $a_1 b_1$ peker på c_1 og c_2.
5. Kanskje, samme som 2)
6. Nei. d_2 peker både på c_2 og c_4
7. Ja $a \rightarrow a$, $b \rightarrow b$, $c \rightarrow c$, $d \rightarrow d$
8. Kanskje. Gitt at $c \rightarrow d$ stemmer.

9. Nei, a_1 f.eks identfiserer to rader.

10. Kanskje, vi kan komme oss til $ABCD$ ved hjelp av AC.

5.2 b

$R = \{A, B, C, D\}$, $F = \{A \rightarrow C, B \rightarrow D, ABC \rightarrow D\}$:

$A^+ = AC$

$D^+ = D$

$BC^+ = BCD$

$AB^+ = ABCD$