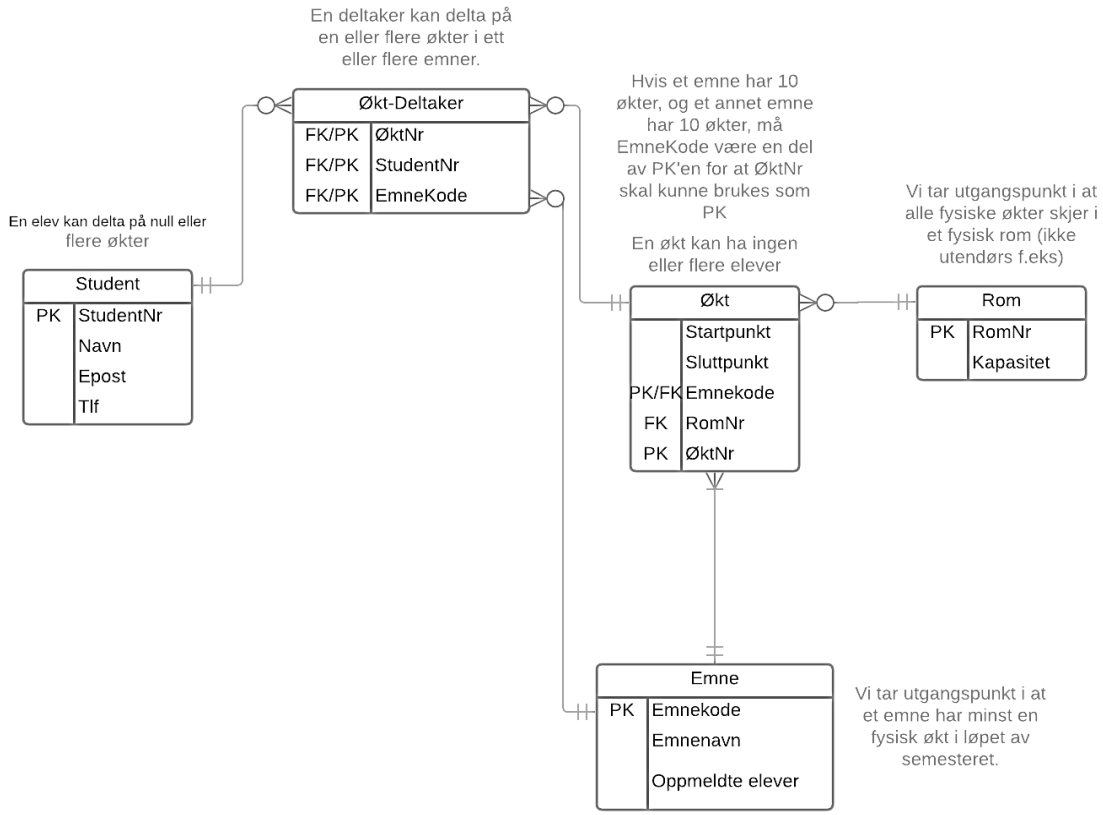
Oppgave 1


En deltaker kan delta på en eller flere økter i ett eller flere emner.
Økt-Deltaker
FK/PK ØktNr
FK/PK StudentNr
FK/PK EmneKode
Hvis et emne har 10 økter, og et annet emne har 10 økter, må EmneKode være en del av PK'en for at ØktNr skal kunne brukes som PK
En elev kan delta på null eller flere økter
Student
PK StudentNr
Navn
Epost
Tlf
En økt kan ha ingen eller flere elever
Vi tar utgangspunkt i at alle fysiske økter skjer i et fysisk rom (ikke utendørs f.eks)
Økt
Startpunkt
Sluttpunkt
PK/FK Emnekode
FK RomNr
PK ØktNr
Rom
PK RomNr
Kapasitet
Emne
PK Emnekode
Emnenavn
Oppmeldte elever
Vi tar utgangspunkt i at et emne har minst en fysisk økt i løpet av semesteret.

Oppgave 2

a)
SELECT * **FROM** deltaker
ORDER BY Etternavn **ASC**, Fornavn **ASC**;

DNr	Fornavn	Etternavn	EPost
15	Anders	Andersen	anders@andersen.no
6	Benny	Ball	benny@benny.no
10	Billy	Betong	billy@ppbb.no
4	Eva	Dahl	eva@dahl.no
12	Frida	Frosk	frida@ppbb.no
1	Hans	Hansen	hans@hansen.no
18	Svetlana	Iversen	svetlana@olsen.no
8	Hans	Jensen	hj@jensen.no
3	Jens	Jensen	jens@jensen.no
16	Julie	Jensen	julie@jensen.no
7	Oline	Jensen	o-j@jensen.no
13	Leon	Latex	leon@ppbb.no
2	Kari	Normann	kari@normann.no
17	Igor	Olsen	igor@olsen.no
5	Ole	Olsen	ole@olsen.no
11	Pelle	Parafin	pelle@ppbb.no
14	Ragna	Rekkverk	ragna@ppbb.no
9	Sandra	Salamander	sandra@ppbb.no
NULL	NULL	NULL	NULL

b)
SELECT Fornavn, Etternavn **FROM** deltaker
WHERE Epost **LIKE** '%@ppbb.no';

Fornavn	Etternavn
Sandra	Salamander
Billy	Betong
Pelle	Parafin
Frida	Frosk
Leon	Latex
Ragna	Rekkverk

c)
SELECT SUM(MåltidPris) **as** Totalpris_Måltid, DagNr **as** Dagsnummer **FROM** måltid
GROUP BY Dagsnummer

Totalpris_Måltid	Dagsnummer
278	1
278	2

d)

```
SELECT * FROM deltaker
LEFT JOIN forfatter
ON deltaker.DNr = forfatter.DNr
WHERE forfatter.PresNR IS NULL;
```

	DNr	Fornavn	Etternavn	EPost	DNr	PresNr
▶	15	Anders	Andersen	anders@andersen.no	NULL	NULL
	16	Julie	Jensen	julie@jensen.no	NULL	NULL
	17	Igor	Olsen	igor@olsen.no	NULL	NULL
	18	Svetlana	Iversen	svetlana@olsen.no	NULL	NULL

e)

```
SELECT deltaker.fornavn as fornavn, deltaker.etternavn as etternavn, COUNT(*) as antall_temaer
FROM deltaker
LEFT JOIN deltakertema
ON deltaker.dnr = deltakertema.dnr
GROUP BY (deltaker.dnr)
ORDER BY antall_temaer DESC;
```

	fornavn	etternavn	antall_temaer
▶	Hans	Hansen	5
	Billy	Betong	5
	Oline	Jensen	4
	Ole	Olsen	3
	Hans	Jensen	3
	Pelle	Parafin	3
	Frida	Frosk	3
	Kari	Normann	2
	Eva	Dahl	2
	Leon	Latex	2
	Jens	Jensen	1
	Benny	Ball	1
	Sandra	Salamander	1
	Ragna	Rekkverk	1
	Anders	Andersen	1
	Julie	Jensen	1
	Igor	Olsen	1
	Svetlana	Iversen	1

f)

```
INSERT INTO måltidbestilling(DNr, MåltidType, DagNr)
VALUES
(1, 'Middag', 1),
(2, 'Lunsj', 1),
(2, 'Middag', 1),
(3, 'Middag', 2),
(4, 'Lunsj', 1);
```

DNr	MåltidType	DagNr
1	Middag	1
2	Middag	1
3	Middag	2
2	Lunsj	1
4	Lunsj	1
NULL	NULL	NULL

Kommentar: Man må ikke ha (DNr, MåltidType, DagNr) etter måltidbestilling hvis man legger det inn I rekkefølgen som tabellen blir opprettet I, men det er ofte greit for å ha oversikt over hvilke verdier som legges inn hvor. Tabellen som er lagt med for å vise spørringen er ***SELECT * FROM*** *måltidbestilling* , Da selve INSERT INTO spørringen ikke returnerer noen tabell.

g)

```
UPDATE deltaker
SET EPost = 'svetlana@iversen.no'
WHERE DNr = 18;
```

Alternativt:

```
UPDATE deltaker
SET EPost = 'svetlana@iversen.no'
WHERE fornavn = 'Svetlana' AND etternavn = 'Iversen';
```

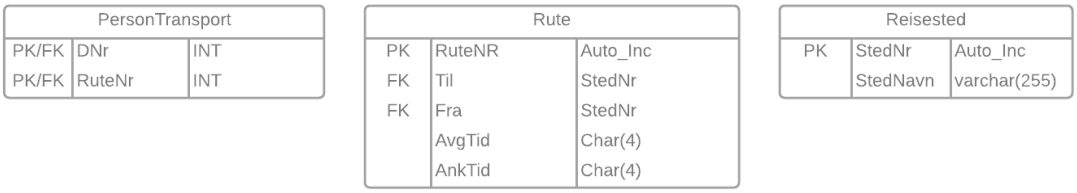
DNr	Fornavn	Etternavn	EPost
1	Hans	Hansen	hans@hansen.no
2	Kari	Normann	kari@normann.no
3	Jens	Jensen	jens@jensen.no
4	Eva	Dahl	eva@dahl.no
5	Ole	Olsen	ole@olsen.no
6	Benny	Ball	benny@benny.no
7	Oline	Jensen	o-j@jensen.no
8	Hans	Jensen	hj@jensen.no
9	Sandra	Salamander	sandra@ppbb.no
10	Billy	Betong	billy@ppbb.no
11	Pelle	Parafin	pelle@ppbb.no
12	Frida	Frosk	frida@ppbb.no
13	Leon	Latex	leon@ppbb.no
14	Ragna	Rekkverk	ragna@ppbb.no
15	Anders	Andersen	anders@andersen.no
16	Julie	Jensen	julie@jensen.no
17	Igor	Olsen	igor@olsen.no
18	Svetlana	Iversen	svetlana@iversen.no
NULL	NULL	NULL	NULL

Kommentar: Den første metoden tar utgangspunkt I at du vet DNr til Svetlana Iversen, noe man kan finne ut ved en enkel spørring. Det er tryggere å endre bruker ved å bruke en PK, da du kan ha brukere med samme fornavn og etternavn. Man kan alternativt endre ved å bruke flere spørringer, men en spørring uten PK er ikke like sikker. Tabellen som er lagt med for å vise spørringen er ***SELECT * FROM*** *deltaker*, Da selve UPDATE spørringen ikke returnerer noen tabell.

h)
SELECT Etternavn, **COUNT**(*) as Antall_Med_Etternavn **FROM** deltaker
GROUP BY Etternavn
HAVING Antall_Med_Etternavn > 1

	Etternavn	Antall_Med_Etternavn
▶	Jensen	4
	Olsen	2

i)
PLAN:



SQL, CREATE:

CREATE TABLE persontransport(
DNr **INT NOT NULL**,
RuteNR **INT NOT NULL**,
FOREIGN KEY (DNr) **REFERENCES** deltaker(DNr),
FOREIGN KEY (RuteNR) **REFERENCES** rute(RuteNR)
);

CREATE TABLE rute(
RuteNR **INT NOT NULL AUTO_INCREMENT**,
Til **INT**,
Fra **INT**,
AvgDato **DATE NOT NULL,**
AvgTid **char(4) NOT NULL**,
AnkDato **DATE NOT NULL,**
AnkTid **char(4) NOT NULL**,
PRIMARY KEY (RuteNR),
FOREIGN KEY (Fra) **REFERENCES** sted(StedNr),
FOREIGN KEY (Til) **REFERENCES** sted(StedNr)
);

CREATE TABLE sted(
StedNr **INT NOT NULL AUTO_INCREMENT**,
StedNavn **varchar(255) NOT NULL**,
PRIMARY KEY (StedNr)
)

SQL, INSERT:

INSERT INTO sted
VALUES (**NULL**, 'Flyplass'), (**NULL**, 'konferansesenter');

	StedNr	StedNavn
▶	1	Flyplass
	2	konferansesenter
*	NULL	NULL

INSERT INTO rute(RuteNR, Til, Fra, AvgTid, AnkTid, AvgDato, AnkDato)
VALUES (**NULL**, 1, 2, '0900', '1030', '2020-11-24', '2020-11-24'), (**NULL**, 2, 1, '1800', '1930', '2020-11-25', '2020-11-25'**);**

	RuteNR	Til	Fra	AvgTid	AnkTid	AvgDato	AnkDato
▶	1	1	2	0900	1030	2020-11-24	2020-11-24
	2	2	1	1800	1930	2020-11-25	2020-11-25
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Kommentar: Jeg fant ut at rutene kan være på forskjellige dager, så jeg oppdaterte tabellen til å reflektere dette etter jeg var ferdig med planen.

INSERT INTO persontransport(DNr, RuteNR)
VALUES (1, 1), (1, 2);

	DNr	RuteNR
▶	1	1
	1	2

j)

```
CREATE OR REPLACE VIEW presentasjonsinformasjon
AS
SELECT p.RomNr as Rom_nummer, r.AntPlasser as Antall_plasser, p.Tittel as Tittel_På_Presentasjon,
t.TemaNavn as Tema, CONCAT(d.fornavn, ' ', d.etternavn) as Presentasjonsholder,
CONCAT(DATE_FORMAT(p.StartTid, '%d. %M kl. %H.%i' ), '-' ,DATE_FORMAT(DATE_ADD(p.starttid,
INTERVAL p.varighet MINUTE), '%H.%i')) as Tidspunkt
FROM Presentasjon p
LEFT JOIN rom r
ON p.RomNr = r.RomNr
LEFT JOIN deltaker d
ON p.DNr = d.DNr
LEFT JOIN tema t
ON p.TemaNr = t.TemaNr
ORDER BY Rom_nummer ASC, Tidspunkt ASC;
```

SELECT * FROM presentasjonsinformasjon

Rom_nummer	Antall_p	Tittel_På_Presentasjon	Tema	Presentasjonsholder	Tidspunkt
A1	100	Feasability of Optimizations Requiring Bounded Treed within a Data Flow Centric Intermediate Representation	Performance and Optimization	Hans Hansen	24. November kl. 09.45-10.05
A1	100	Evaluation of graph algorithm frameworks for multi-core systems	Performance and Optimization	Kari Normann	24. November kl. 10.15-10.35
F1	50	IT students perceptions of mandatory coursework	IT didactics	Jens Jensen	24. November kl. 09.45-10.05
F1	50	Introducing ePortfolios to IT students: The support process	IT didactics	Eva Dahl	24. November kl. 10.15-10.35
F1	50	Teaching AI Ethics: Observations and Challenges	IT didactics	Ole Olsen	24. November kl. 10.45-11.05
F1	50	The Live Programming Lecturing Technique: A Study of the Student Experience in Introductory and Advanced Programming Courses	IT didactics	Benny Ball	24. November kl. 11.15-11.35
F2	40	INERTIA AND CHANGE IN TRANSFORMATION OF THE IT-FUNCTION IN LARGE ORGANIZATIONS: A PATH THEORY LENS	Digital transformation	Oline Jensen	25. November kl. 09.45-10.05
F2	40	DIGITAL TRANSFORMATION UNDER A PANDEMIC: A CASE STUDY OF COVID-19 CONTACT TRACING IN NORWAY	Digital transformation	Hans Jensen	25. November kl. 10.15-10.35
F2	40	Exploring the Impact of Mob Programming on the Well-Being of Developers: Insights from a Software Company	Digital transformation	Sandra Salamander	25. November kl. 10.45-11.05
F2	40	Exploring the Hiring Process of a Norwegian Municipality using Process Mining	Digital transformation	Billy Betong	25. November kl. 11.15-11.35

Oppgave 3

Ansattnr	Fnavn	Enavn	Tlfnr	Utstyr	Innkjøpspris	Innkjøpsdato	Type
123456	Jens	Jensen	55555555	PC	10000	2019-01-02	Lenovo
123456	Jens	Jensen	55555555	Mobil	9000	2019-01-02	iPhone
123456	Jens	Jensen	55555555	Stol	3490	2018-12-12	Zareto
234567	Kari	Normann	66666666	Mac	13900	2017-05-05	MacBook Pro
234567	Kari	Normann	66666666	Mobil	9900	2019-05-05	Samsung
234567	Kari	Normann	66666666	Stol	3900	2017-07-05	Watford
234567	Kari	Normann	66666666	Headset	2900	2017-08-05	Boss

Hvis vi tar utgangspunkt i at én ansatt kun kan kjøpe én av en type utstyr per dag

Ansattnr	Fnavn	Enavn	Tlfnr	Rom	Sted	Etasje	Adresse
123456	Jens	Jensen	55555555	12	Oslo	1	Smalveien 1
234567	Kari	Normann	66666666	12	Oslo	1	Smalveien 1
345678	Ole	Olsen	77777777	22	Oslo	1	Smalveien 1
445544	Lise	Olsen	88888888	Gløtt	Bergen	5	Brygga 2
554455	Per	Persen	88668866	Gløtt	Bergen	5	Brygga 2
989898	Eva	Jensen	45454545	Regn	Bergen	5	Brygga 2
323232	Nils	Nilsen	23343223	Regn	Bergen	5	Brygga 2

Eksamen, DB1102 H2020 – Kandidatnr 10426

PK	FK	FK/PK

ANSATTE

AnsattNr	Fnavn	Enavn	Tlfnr	Arbeidsrom
123456	Jens	Jensen	55555555	12
234567	Kari	Normann	66666666	12
345678	Ole	Olsen	77777777	22
445544	Lise	Olsen	88888888	Gløtt
554455	Per	Persen	88668866	Gløtt
989898	Eva	Jensen	45454545	Regn
323232	Nils	Nilsen	23343223	Regn

UTSTYR, EIER

Ansattnr	UtstyrID
123456	5
123456	6
123456	4
234567	1
234567	7
234567	2
234567	3

INNKJØP

UtstyrID	ProduktID	Dato
5	1	2019-01-02
6	2	2019-01-02
4	3	2018-12-12
1	4	2017-05-05
7	5	2019-05-05
2	6	2017-07-05
3	7	2017-08-05

PRODUKT

ProduktID	Utstyr	Type	Pris
1	PC	Lenovo	10000
2	Mobil	iPhone	9000
3	Stol	Zareto	3490
4	Mac	MacBook Pro	13900
5	Mobil	Samsung	9900
6	Stol	Watford	3900
7	Headset	Boss	2900

AVDELING

AvdNr	Etasje	Adresse	Sted
1	1	Smalveien 1	Oslo
2	5	Brygga 2	Bergen

ROM

AvdNr	Rom
1	12
1	22
2	Gløtt
2	Regn

Vi tar utgangspunkt i at alle rom-navn er unike, også på tvers av kontorer. Bergen-Kontoret kan ikke ha to rom som heter "Gløtt", eller et rom som heter "12"

For at databasen skal oppnå 3NF må den ha oppfylt følgende kriterier:

- Hver tabell inneholder kun atomære verdier (**1NF**)
    - Alle tabell-celler skal kun inneholde én verdi
- Ingen tabeller inneholder partielle avhengigheter (**2NF**)
    - Alle kolonner skal være knyttet til hele PK, det vil si at hvis man har flere PK, skal alle kolonner være direkte knyttet til alle PK'ene. Hvis de bare er delvis knyttet til PK, skal de skilles ut I egen tabell.
- Ingen tabeller inneholder transitive avhengigheter (**3NF**)
    - Alle kolonner skal være direkte knyttet til PK. Hvis de er knyttet til en annen kolonne, som igjen er knyttet til PK, skal det skilles ut I egen kolonne.

ANSATTE:

- Navn, etternavn, telefonnummer, og arbeidsrom er alle direkte knyttet til ansattnr. Det er ingen partielle eller transitive avhengigheter mot PK.

UTSTYR, EIER

- Ansattnr og UtstyrID er begge FK'er fra andre kolonner som sammen danner en PK. Dermed er det ingen partielle eller transitive avhengigheter.

INNKJØP

- Her tar vi utgangspunkt I at man kun kjøper én enhet per kjøp, hvis ikke kunne man laget en egen "bestilling"-tabell der man kan ha flere enheter, og linke det til en innkjøpsID.
- ProduktID og Dato får en unik UstyrID internt I bedriften. Det gjør at det ikke er noen partielle eller transitive avhengigheter mot PK.

PRODUKT

- Her tar vi utgangspunkt I at prisen aldri endrer seg, hvis prisen skalendres kunne man hatt pris under innkjøpstabellen.
- Utstyr, Type, og Pris er direkte knyttet til ProduktID, og har ingen partielle eller transitive avhengigheter mot PK.

AVDELING:

- Her tar vi utgangspunkt I at en avdeling kun har én etasje.
- Både etasje, adresse, og sted er direkte knyttet mot PK, og har ingen partielle eller transitiv avhengighet

ROM:

- Her tar vi utgangspunkt I at rom-navnene er unike på tvers av alle avdelinger. Hvis de ikke hadde vært det, ville AvdNr og Rom samlet vært PK.

Siden det kun er atomære verdier, ingen partielle avhengigheter, og ingen transitive avhengigheter, er tabellen på 3NF.

Oppgave 4

Grunner til at spørringer kan gå treigt:

- Det kan være at man joiner mange tabeller, noe som gjør at spørringen må jobbe en del med å få de riktige radene og kolonnene på riktig sted. Dette er spesielt for views, der man ikke nødvendigvis ser spørringene som ligger bak.
- Jo mer komplisert en spørring er, jo lenger tid vil den ta. Dette er spesielt tydelig på steder der datamaskinen gjør mye av jobben for deg (f.eks joins, subqueries osv). Har man flere av disse, kanskje til og med nested, vil det fort ta lang tid.
- Man spør for bredt – hvis man må hente inn veldig store mengder data kan det ta lang tid.
- Lite RAM på PC'en. SQL cacher data I RAM for å optimalisere tidsbruken på spørringer. Hvis man har for lite, eller dedikert for lite, så kan de ha en effekt på farten.

Løsninger:

- Indexes: Hvis man lager en index på noe man kjører ofte, så kan det være at spørringen går mye kjappere. Index gjør at det man legger inn I indexen blir sortert og lagret, så spørringen som benytter seg av den ikke trenger å gå gjennom samme liste og sortere

den selv hver gang. Hvis man f.eks har en liste med studenter som en database sjekker mot flere ganger, kan man lage en index på student(navn) for å kjappere finne navnene på studentene.

- Smartere spørringer: Ved å ikke bruke unødvendige eller uptimaliserte løsninger kan man spare tid, spesielt om man kan unngå unødvendige joins og subqueries.
- Mer RAM / øk mengden RAM databasen får bruke.