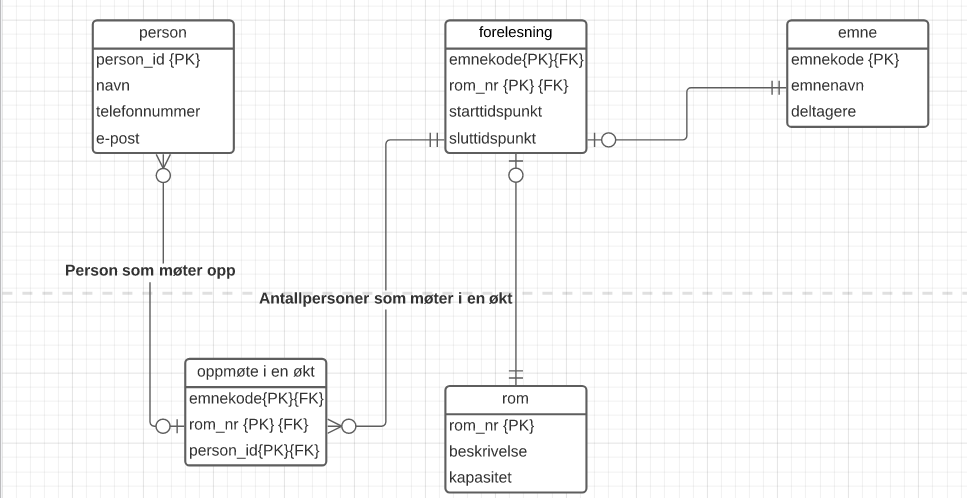
Eksamen DB1102 H2020

**Oppgave 1 – modellering**



**Beskrivelse av modellen:**

Ender opp med 5 entiteter, person forelesning, emne, rom og oppmøte i en økt (koplings-entitet) uitifra oppgave teksten. Kopling entiteten en økt’ tar utgangspunkt i at en person kan kun møte opp i en økt om gangen, mens en forelesning kan ha flere oppmøter per forelesning.

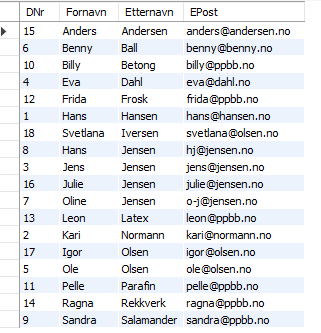
Valgte å bruke en rullerende surigatnøkkel som primærnøkkel for person tabellen i stedet for en naturlignøkkel telefonnummer, synes det er en ryddigere møte å gjøre det på.

Har også satt opp enda en surigatnøkkel (rom\_nr) for rom tabellen, siden det et rom lan ha lik beskrivelse og samme kapasitet, da kan disse attributtene brukes som primærnøkkel.

**Oppgave 2 – SQL**

**a)**

SELECT \* FROM deltaker ORDER BY etternavn ASC, fornavn ASC;



**b)**

SELECT fornavn, etternavn FROM deltaker WHERE epost LIKE '%@ppbb.no';



**c)**

SELECT dagnr AS Konferansedag,

SUM(måltidpris) AS Måltidpris

FROM måltid

GROUP BY dagnr;

**d)**

Velger å bruke subquery her.

SELECT \* FROM deltaker WHERE dnr NOT IN(SELECT dnr FROM forfatter);

**e)**

SELECT fornavn, etternavn, COUNT (temanr) AS temaer FROM deltaker

LEFT JOIN deltakertema ON deltaker.DNr = deltakertema.DNR

GROUP BY deltaker.DNr

ORDER BY temaer DESC;



**f)**

**Legger inn måltidene**

INSERT INTO måltidbestilling(DNr,Måltidtype,Dagnr) VALUES (4,'Middag',1);

INSERT INTO måltidbestilling(DNr,Måltidtype,Dagnr) VALUES (6,'Lunsj',2);

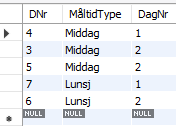
INSERT INTO måltidbestilling(DNr,Måltidtype,Dagnr) VALUES (5,'Middag',2);

INSERT INTO måltidbestilling(DNr,Måltidtype,Dagnr) VALUES (7,'Lunsj',1);

INSERT INTO måltidbestilling(DNr,Måltidtype,Dagnr) VALUES (3,'Middag',2);

**Sjekker at alt er lagt inn**

SELECT \* FROM måltidbestilling;



**g)**

UPDATE deltaker SET EPost = 'svetlana@iversen.no' WHERE Fornavn = 'Svetlana';

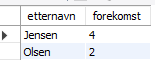
SELECT EPost FROM deltaker WHERE fornavn = 'Svetlana';



**h)**

SELECT etternavn, COUNT(Etternavn) AS forekomst

FROM deltaker GROUP BY etternavn HAVING forekomst > 1;



**i)**

CREATE TABLE transport (

DNr int(11) NOT NULL,

Dag\_nr int(11) NOT NULL,

Fra\_plass ENUM('konferanse','flyplass') NOT NULL,

Til\_plass ENUM('konferanse','flyplass') NOT NULL,

CONSTRAINT transport\_pk PRIMARY KEY (DNr,Dag\_nr),

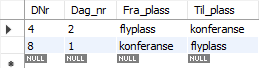
CONSTRAINT transport\_fk FOREIGN KEY (DNr) REFERENCES deltaker(DNr)

);

INSERT INTO transport(DNr, Dag\_nr,Fra\_plass,Til\_plass) VALUES(8,1,'konferanse','flyplass');

INSERT INTO transport(DNr, Dag\_nr,Fra\_plass,Til\_plass) VALUES(4,2,'flyplass','konferanse');

SELECT \* FROM transport;



**j)**

**Lager spørringen først:**

SELECT p.RomNr,

CONCAT(DATE\_FORMAT(starttid, "%d. %M kl %H.%i-"), DATE\_FORMAT(DATE\_ADD(starttid, INTERVAL 20 MINUTE),"%H.%i")) AS tidspunkt,

p.tittel,

CONCAT( d.fornavn,' ', d.etternavn) AS navn,

t.temanavn, r.antplasser

FROM presentasjon p

LEFT JOIN deltaker d ON p.DNr = d.DNr

LEFT JOIN tema t ON p.TemaNr = t.TemaNr

LEFT JOIN rom r ON p.RomNr = r.RomNr

ORDER BY r.Romnr ASC, tidspunkt ASC;

**Lager viewet:**

CREATE VIEW presentasjoner

AS

SELECT p.RomNr,

CONCAT(DATE\_FORMAT(starttid, "%d. %M kl %H.%i-"), DATE\_FORMAT(DATE\_ADD(starttid, INTERVAL 20 MINUTE),"%H.%i")) AS tidspunkt,

p.tittel,

CONCAT( d.fornavn,' ', d.etternavn) AS navn,

t.temanavn, r.antplasser

FROM presentasjon p

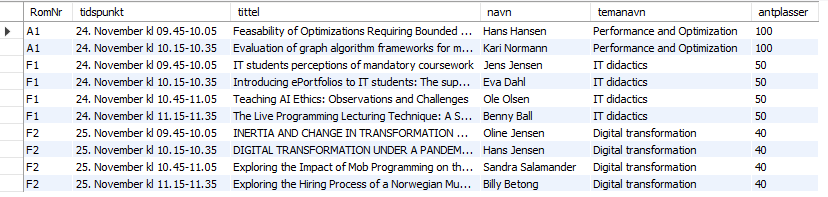
LEFT JOIN deltaker d ON p.DNr = d.DNr

LEFT JOIN tema t ON p.TemaNr = t.TemaNr

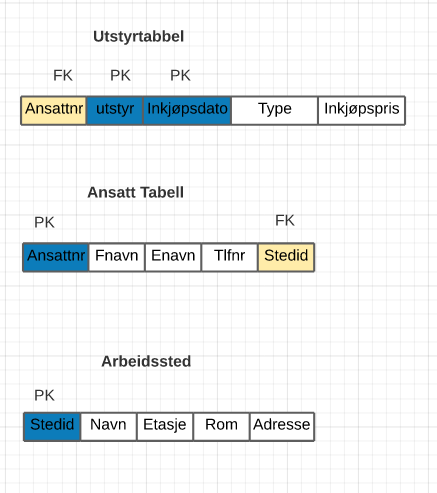
LEFT JOIN rom r ON p.RomNr = r.RomNr

ORDER BY r.Romnr ASC, tidspunkt ASC;

**Tester view:**

SELECT \* FROM presentasjoner;

**Oppgave 3 – Normalisering**

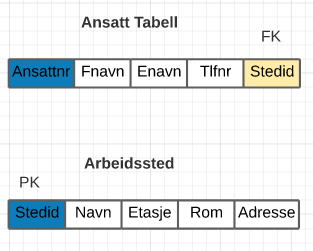
**Tabell 1:**

**Forklaring til tabell 1:**

Tabellen ovenfor er på 3NF fordi attributtene innkjøpspris og type er avhengig av begge primærnøklene. Eks, for å finne innkjøpspris på type så må man vite både dato og utstyr.

Har tatt ut informasjon om ansatt og lagt dette inn i en egen tabell, siden de attributtene kun er avhengig av ansatnr attributt.

Tillegg har jeg laget en tabell for arbeidssted tabell som har fremmednøkkel Stedid i ansatt tabellen, dette for å gjøre tabellene mer oversiktlige.

 **Tabell 2:**

**Oppgave 4 – Diverse**

* For å optimalisere databasen kan vi bruke noe som heter for *indekser,* indekser er brukes for å kjapt hente ut kolonner med gitt kolonne verdier. Uten hjelp av indekser så leter spørringen gjennom hele tabellen, noe som kan ta vesentlig lang tid hvis tabellen inneholder mye data.

* Minimere bruken av «krevende» views, siden et view kun er en pre-definert spørringen, så kan den bli tung å kjøre avhengig av kompleksiteten til spørringen.