DAT1000 Database 1 – løsningsforslag eksamen vår 2019

Oppgave 1

1a

```
SELECT *
FROM LagerRom
WHERE LNr = 1
AND Kvm BETWEEN 5 AND 10;
```

1b

```
SELECT A.*, UPPER(CONCAT(Fornavn, ' ', Etternavn)) AS Kundenavn
FROM Avtale AS A, Kunde AS K
WHERE A.KNr = K.KNr
AND YEAR(FraDato) = 2019;
```

1c

```
SELECT LNr, COUNT(*) AS Antall
FROM LagerRom
GROUP BY LNr
HAVING COUNT(*) > 5
ORDER BY COUNT(*) DESC;
```

1d

```
UPDATE Lager
SET KvmPris = KvmPris * 1.1
WHERE PostNr = '3200';
```

1e

```
CREATE TABLE LagerRom
(
  LNr     INTEGER,
  RNr     INTEGER,
  Kvm     DOUBLE(4, 1) NOT NULL,
  PRIMARY KEY (LNr, RNr),
  FOREIGN KEY (LNr) REFERENCES Lager (LNr)
);

INSERT INTO
  LagerRom(LNr, RNr, Kvm)
VALUES
  (3, 5, 15);
```

1f

```
SELECT A.*, KvmPris*Kvm*DATEDIFF(TilDato, FraDato) AS Beløp
FROM Avtale AS A, LagerRom AS R, Lager AS L
WHERE A.LNr = R.LNr
AND A.RNr = R.RNr
AND R.LNr = L.LNr
AND L.LNr = 1;
```

1g

```
CREATE VIEW LedigeRom AS
SELECT *
FROM LagerRom AS R
WHERE R.RNr NOT IN
 (
        SELECT A.RNr
        FROM Avtale AS A
        WHERE A.LNr = R.LNr
        AND CURDATE() BETWEEN FraDato AND TilDato
 );
```

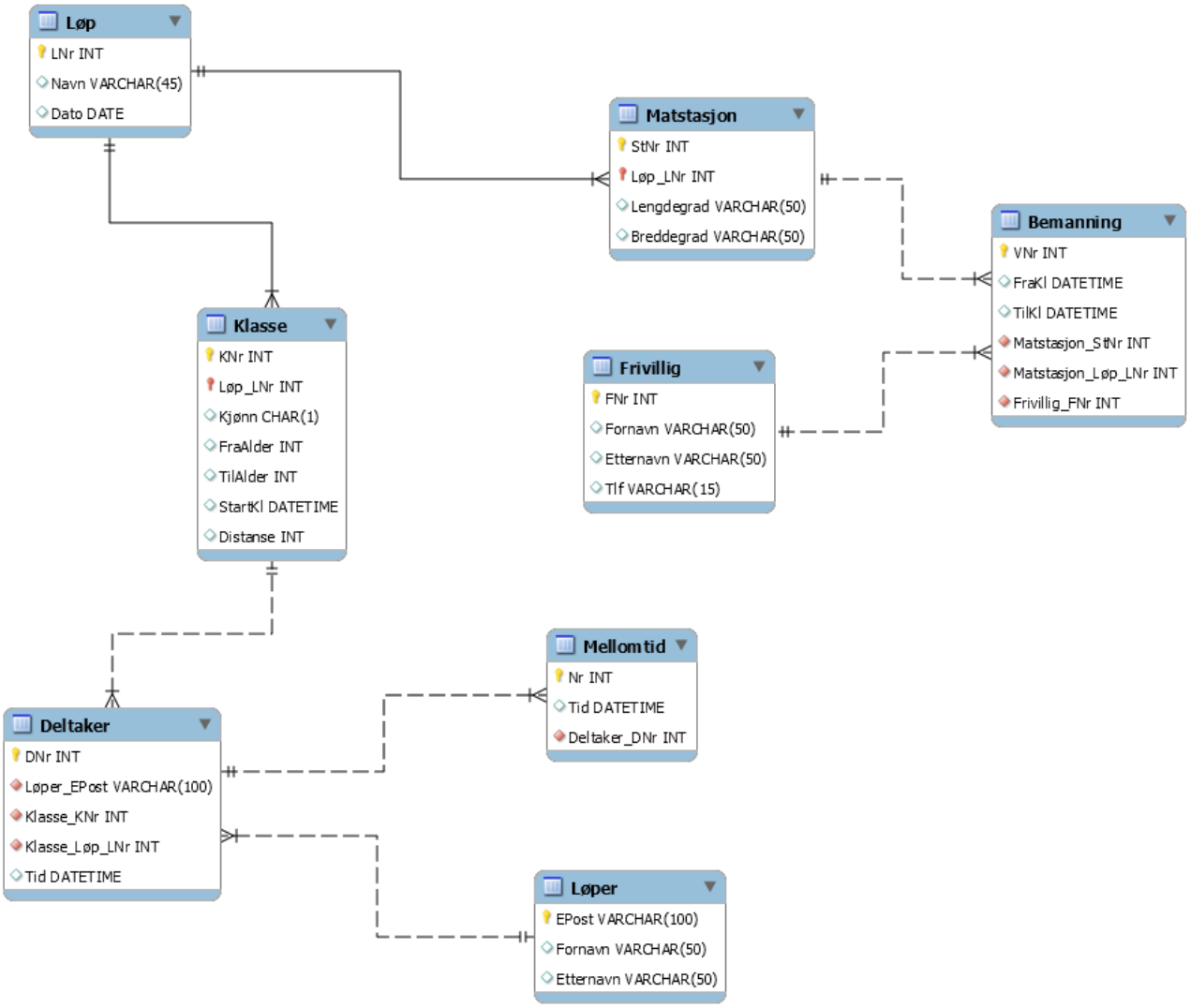
Oppgave 2

2-a

Svaret skal inneholde en logisk datamodell:

- Det skal opprettes en entitet for hver av tabellene i oppgave 1.
- Lager, Kunde og Avtale får enkle primærnøkler, hhv. LNr, KNr og ANr.
- Det skal være ikke-identifiserende en-til-mange forhold mellom LagerRom og Avtale, samt mellom Kunde og Avtale.
- Det skal være et identifiserende en-til-mange forhold mellom Lager og LagerRom.
- Primærnøkkel i LagerRom skal være både LNr og RNr.
- Alle kolonner merket med stjerne, skal merkes med FK i datamodellen.
- Alle en-til-mange forhold kan ha «nøyaktig 1» på én-siden og «0 eller mange» på mange-siden.

2-b



Oppgave 3

3a

Ulempen med den foreslåtte tabellen er at både sonenavn og pris blir gjentatt for hver forekomst av et gitt sonenummer. Dette skaper redundans (dobbeltlagring) og medfører at oppdatering av tabellen blir mer komplisert. For å endre navn eller pris for en sone, så må man passe på at samtlige forekomster blir behandlet.

Funksjonelle avhengigheter:

- LNr -> ALLE KOLONNER
- SoneNr -> SoneNavn, KvmPris

Kandidatnøkkel: LNr

Resultattabeller:

- Lager(LNr, Gateadresse, PostNr, SoneNr*)
- Sone(SoneNr, SoneNavn, KvmPris)

3b

Stikkord som bør være med i en god besvarelse: GRANT og REVOKE er SQL-kommandoer som brukes i forbindelse med brukeradministrasjon. Etter at man har opprettet brukere, og eventuelt roller, kan disse kommandoene brukes for å tildele og frata rettigheter mot tabeller i databasen (objektrettigheter). GRANT og REVOKE kan også brukes for å tildele og frata mer generelle systemrettigheter mot databasen, som f.eks. rettigheten til å starte og stoppe databasen.

Eksempler på bruk mot tabellene i oppgave 1 følger, der vi antar at brukere og roller som blir referert er opprettet på forhånd.

Gi bruker per leserettighet på tabellen Kunde:
GRANT SELECT ON Kunde TO peder;

Gi bruker kari oppdateringsrettighet på tabellen Lager:
GRANT UPDATE ON Lager TO kari;

Frata bruker kari oppdateringsrettighet på tabellen Lager:
REVOKE UPDATE ON Lager FROM kari;

Gi *rollen* selgere leserettighet på tabellen Avtale:
GRANT SELECT ON Avtale TO selgere;

Oppgave 4

Korrekte alternativer:

- bcbcd dacbb cbaac