Databaser-gr. 119: **Øvinger**

Tenker vi bruker dette dokumentet for å jobbe med øvinger, også innfører vi over til en pdf gjennom Word når vi leverer :))

Øving 1

Gruppe: **119**

- Andrine Kirstine Gudbrandsen, Endre Mork, Joakim Pettersen Vassbakk

Oppgave 1: Databasesystemer

a) **Database** (DB) er elektronisk lagret strukturert informasjon med data. Ofte ønsker man å ha data som er relatert med hverandre, som ofte er et designprinsipp man følger. Derimot et **databasehåndteringssystem** (DBMS) er et system som gjør det mulig å interagere med selve databasen. Det er et filhåndteringssystem som gjør det mulig å skrive til og få tilgang på det som allerede er på databasen. Hvis en som bruker interagerer med databasen, så interagerer man egentlig med databasehåndteringssystemet.

b) 1) **Program-data uavhengighet** handler om en uavhengighet mellom applikasjon eller program og databasesystemet. Dette er for å ha en ideell fungerende database uavhengig av programmet.

2) **Flerbrukersstøtte** er at databasehåndteringssystemet støtter å ha flere brukere som interagerer med databasen samtidig eller ikke. F.eks. med helsejournaler så vil man kunne ønske at to forskjellige kan kunne skrive til journalen samtidig uten at det kolliderer.

3) **Selvbeskrivende** handler om at strukturen og dataen i databasen er mulig/lett å forstå uten noe mer informasjon. Dette kan f.eks. være en tabell med username, epost, og telefonnummer, så vil det være lett å forstå hva som hører sammen og hva som skal i hvilken kolonne eller rad, som da vil være selvbeskrivende.

Oppgave 2: ER-modellen

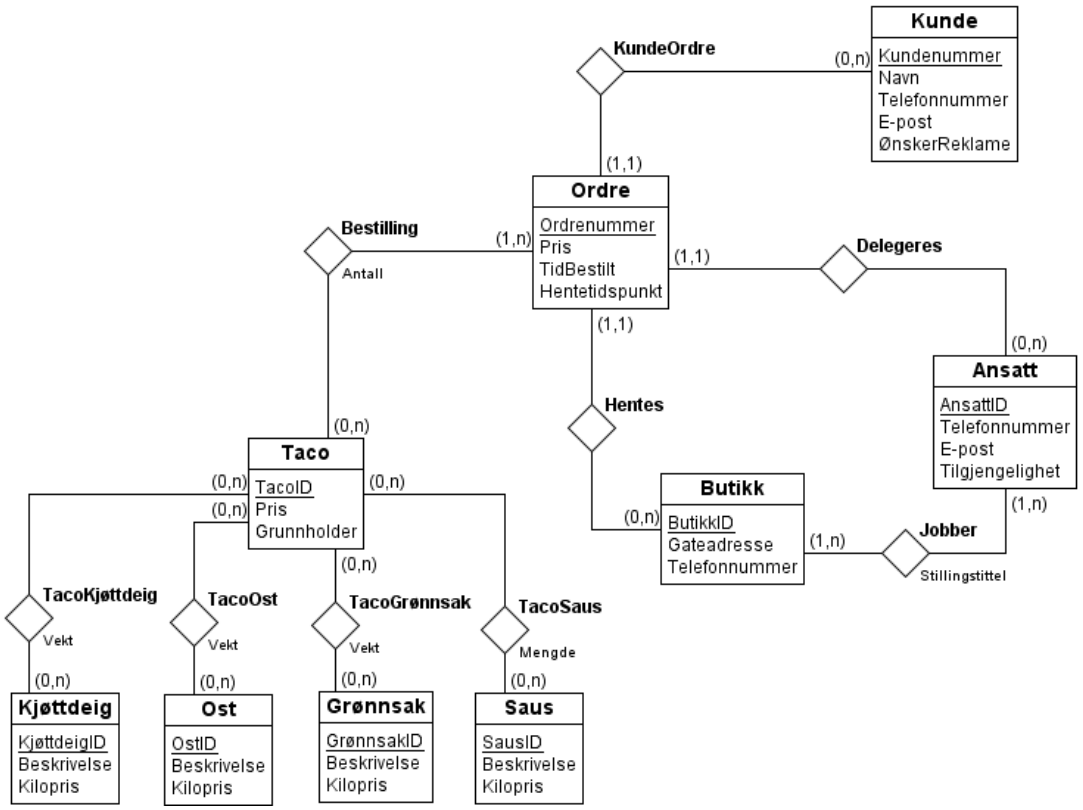
a)

1) En **entitet** er en representasjon av et objekt eller “ting” som har bestemte attributter som beskriver entitetens egenskaper. En **entitetsklasse** er en “kategorisering” av entiteter som har de samme attributtene. For å illustrere: man kan tenke på en entitet som en instans av en entitetsklasse.

2) En **relasjon** representerer en sammenheng/assosiasjon mellom to eller flere entiteter. For eksempel: “bilEier” kan være en relasjon mellom en “Person” og en “Bil”. **Relasjonsklasser** er kategoriseringer av slike relasjoner med like egenskaper mellom like entitetsklasser, på samme måte som entiteter er instanser av entitetsklasser.

3) I en database er det nødvendig at alle entiteter har en unik **nøkkelattributt** slik at de spesifikke enhetene kan skilles fra hverandre/identifiseres til tross for å ellers ha like egenskaper. For eksempel kan to biler ha samme farge, modell, etc. men fortsatt være skillbare ved hjelp av en nøkkelattributt.

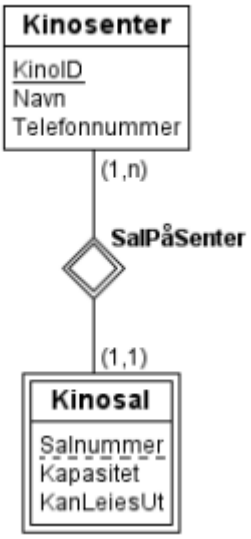
b)



1. True - Taco beskriver egenskapene til entiteter/instanser av taco, og notasjonen indikerer at TacoID vil være unik for taco instans
2. True - Kardinaliteten (0,n) indikerer at taco kan ha “fra null til mange” av hver enkelt entitetsklasse.
3. False - Relasjonsklassen “Bestilling” har en en-til-mange relasjon fra ordre, som betyr at hver instans av ordre må ha minst én taco.
4. True - (1,n)-relasjoner kan gå så høyt man vil.
5. False - “Ordre” har en (1,1) “Hentes”-relasjoner til “Butikk”, som betyr at hver enkelt ordre kan kun hentes med én spesifikk butikk.
6. True - Siden “Kunde” har (0,n) “KundeOrdre” relasjoner, som betyr at null KundreOrdre fra en gitt kunde er helt gyldig.

7. False - Ikke nødvendigvis. Utenom nøkkelen presiserer grønnsak-entitetsklassen kun beskrivelse og kilopris.
8. True - En Ansatt kan ha n antall Jobber (relasjoner) med hver sin Stillingstittel (relasjons-attributt).
9. False - Attributten “Tilgjengelighet” impliserer at ordre helst går til de som ikke allerede er opptatt med ordre.
10. True - Navn er en av attributtene til Kunde-klassen, som betyr at den må registreres.

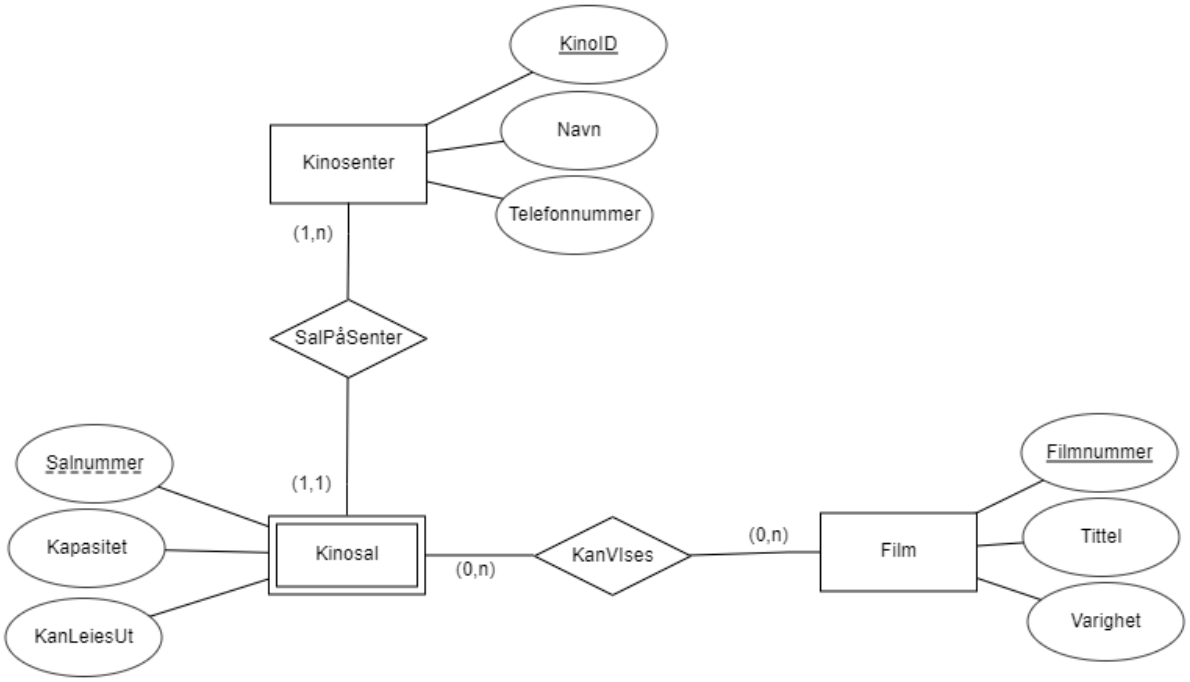
Oppgave 3: Svake klasser, forekomstdiagram og nye krav



a)
Det er hensiktsmessig med svake klasser når det ikke er noen naturlig atributt som fungerer som nøkkelatributt. I den gitte modellen er kinosenter den identifiserende entitetsklassen. SalPåSenter er en identifiserende relasjonsklasse. Salnummer er er delvis nøkkel.

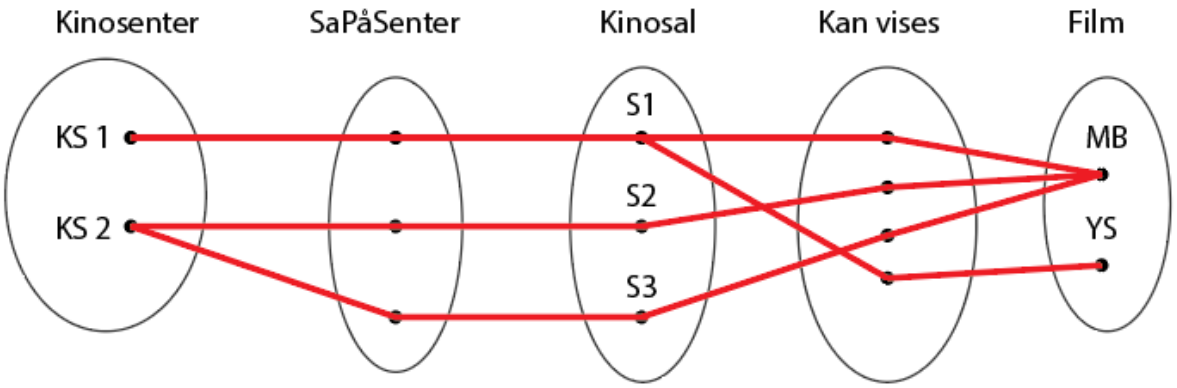
b)
Dersom kardinaliteten blir satt til (0,1) vil ikke Kinosal lenger kunne modelleres som svak. Det er fordi man kan få en Kinosal uten en identifiserende relasjon. Relasjonen (1,n) hadde ikke gitt mening siden en kinosal ikke kan høre til flere kinoer

c)

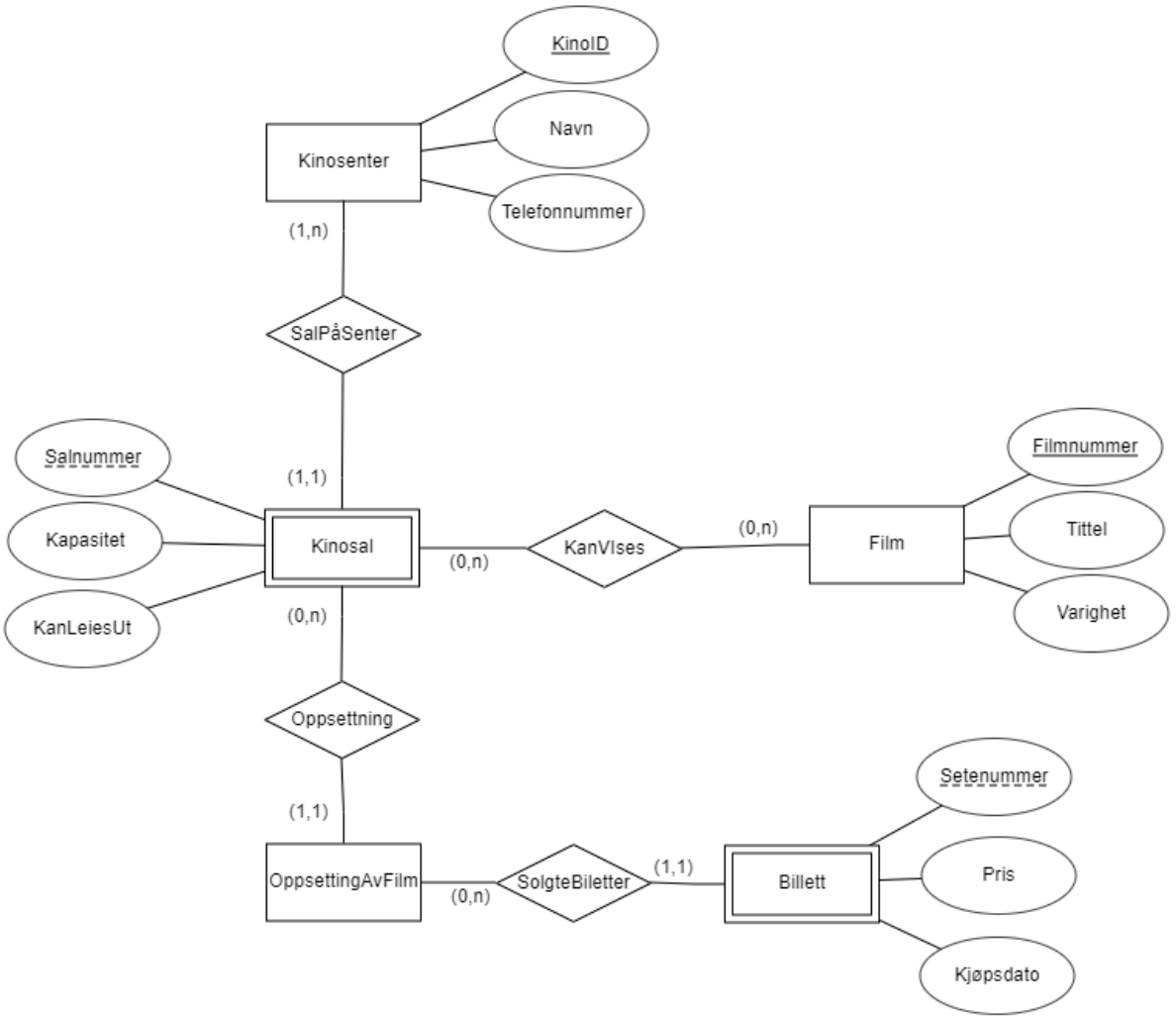


Antar at kinosaler ikke må kunne vise noen filmer, derfor (0,n)-relasjonen

d)



e)

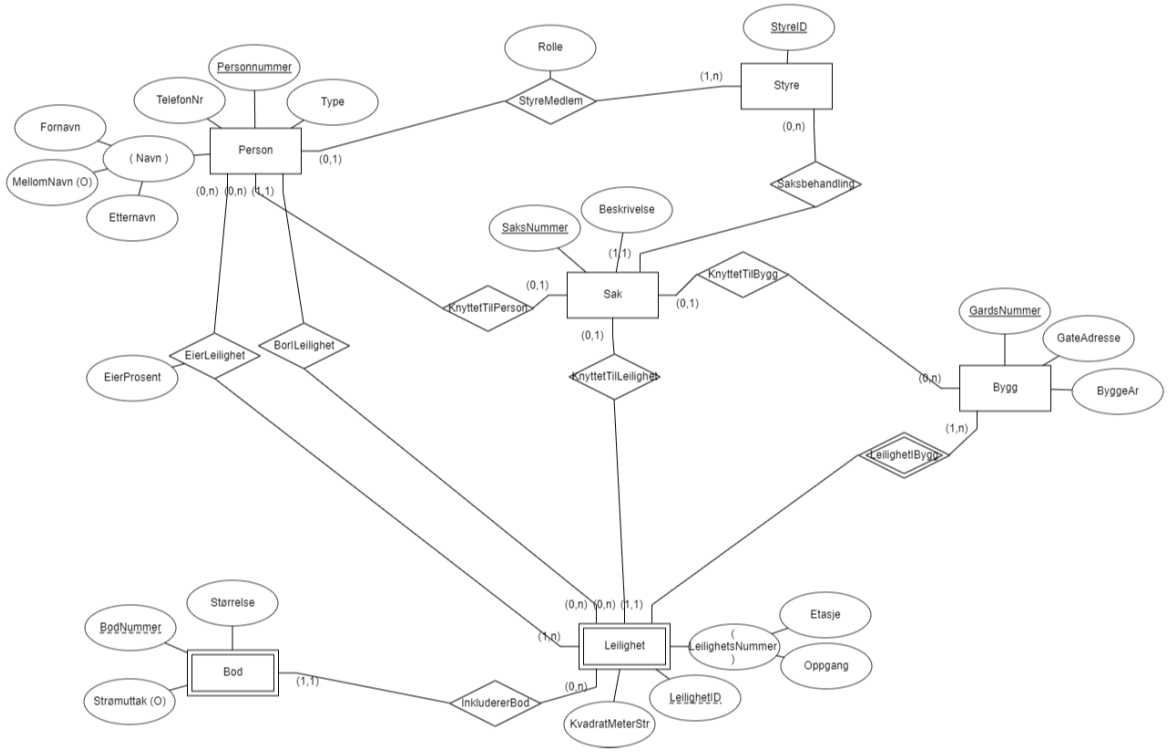


Antagelser: En kinosal kan kan kan ha mange eller ingen oppsetninger av en film. Alle filmer må ha en kinosal.

Oppgave 4: Fra miniverden til ER-modell

- Antar at personnummeret er gyldig å bruke som nøkkelattributt.
- Antar at personer bor i kun én leilighet av gangen (dermed (1,1)-relasjon).
- Antar at ethvert bygg har minst én leilighet (dermed (1,n)).

- Begrensning ved programmet: Vi brukte ERDPlus her, som førte til to problemer:
    - For relasjonsklassen “StyreMedlem” kan jeg ikke presisere at minst ett medlem må være leder.
    - For entitetsklassen “Sak” ønsket jeg å lage kun én relasjon mellom sak, person, leilighet og bygg, men ERDPlus lar en ikke trekke relasjoner mellom mer enn to entitetsklasser. Derfor har vi tre forskjellige relasjons klasser (teknisk sett fører til at en sak kan handle om én av hver).



Spent på det monstre av diagram du lager her. U got this ;)
> ty <3

Øving 2

Gruppe: **119**

- Andrine Kirstine Gudbrandsen, Endre Mork, Joakim Pettersen Vassbakk

Oppgave 1

a) **Total spesialisering:** Betyr at enhver entitet av en superklasse *må* være en instans av (nøyaktig) *én* av subklassene.
Disjunkte subklasser: Betyr at enhver entitet av en superklasse *kan* være en instans av (nøyaktig) *én* av subklassene - men må ikke nødvendigvis være det.

b) i) **Disjunkt og total:** Enhver instans av en superklasse *må* være instans av *nøyaktig én* subklasse.

- Eksempel: Miniverden skal beskrive biler av forskjellige typer, gjennom superklassen “bil” og subklassene “personbil”, “lastebil”, “bobil” og “varebil”. Med *disjunkt og total* begrensning så må hver entitet/instans av en bil være enten en personbil, lastebil, bobil eller varebil.

ii) **Disjunkt og delvis:** Enhver instans av en superklasse *kan* være instans av *nøyaktig én* subklasse.

- Eksempel: Miniverden skal beskrive mat, gjennom superklassen “måltid” og subklassene “frokost”, “lunsj” og “middag”. En gitt matrett kan altså være en av disse tre, men må ikke nødvendigvis være det.

iii) **Overlappende og total:** Enhver instans av en superklasse *må* være instans av *minst én* subklasse, *eventuelt flere*.

- Eksempel: Vi har superklassen “videogame” med subklassene “windowsgame”, “macgame”, “linuxgame”, “playstationgame”, “xboxgame” og “nintendogame”. Ethvert instans av videogame må være et spill på minst en plattform: altså det må være en av subklassene, men kan være flere.

iv) **Overlappende og delvis:** Enhver instans av en superklasse *kan* være instans av *minst én* subklasse, *eventuelt flere*.

- Eksempel: Miniverden skal beskrive personer med forskjellige yrker. Vi har superklassen “person” med subklassene “rørlegger”, “ingeniør”, “lærer”, “elektriker” (osv.). Enhver person kan altså være en instans av en eller flere av disse subklassene/yrkene.

c) - **Figur 1: Feil syntaks**.

Det skal ikke være en relasjon (her: “Foretrekker”) mellom en superklasse og sine subklasser, kun begrensningen mellom de (her: “disjunkt og total).

- **Figur 2: Gyldig syntaks**.

Person må være enten smart eller dum, og en handling kan være smart og/eller dum.

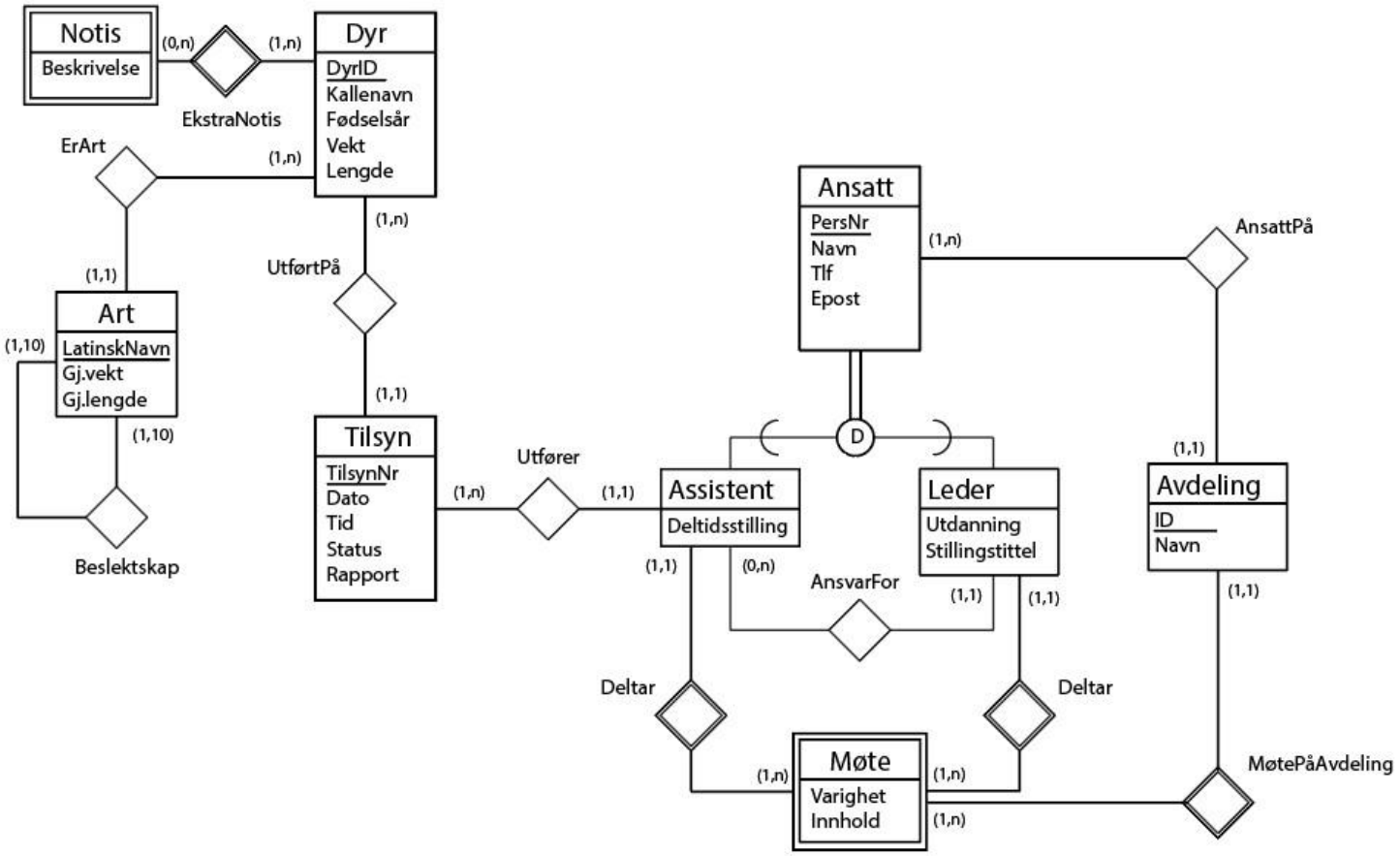
- **Figur 3: Gyldig syntaks**.

Bok kan være digital og/eller fysisk. *Men* det er litt ulogisk at begrensningen er delvis - altså en bok kan her være verken fysisk eller digital.

- **Figur 4: Feil syntaks**.

“Hjemme” og “Borte” er subklasser uten superklasse.

Oppgave 2: Dyrehage



Oppgave 3

Sammenhengen mellom **primærnøkkel og entitetsintegritet**, dreier seg i hovedsak om at entitetsintegritet setter en begrensning på verdiene slik at vi ikke lagrer samme rad eller tuppel flere ganger. Derfor har vi en primærnøkkel som er unike verdier for hver rad. Denne kan ikke være null verdier og gjør at vi kan ha unike id-er for hver tuppel.

Sammenhengen mellom **fremmednøkkel og referanseintegritet**, dreier seg i hovedsak om at vi ønsker å ha noen verdier som går igjen i forskjellige tabeller eller referere til andre tabeller slik at vi får en sammenheng. Derfor har vi en fremmednøkkel som er en primærnøkkel hos en annen tabell slik at man får en kryssreferanse, altså at det er en tabell som har en integritet basert på referanseverdier. Fremmednøkkel er altså en nøkkel som ikke "hører til" og er veldig praktisk når man har flere tabeller med ulike verdier. Fremmednøkler kan, i motsetning til primærnøkler, være null.

Oppgave 4

a) **HasExam** (ExamNo, StudentNo)
- ExamNo er fremmednøkkel mot Exam
- StudntNo er fremmednøkkel mot Student

Exam (ExamNo, CourseCode, ExaminationAids)

Student (StudentNo, Name)

SetUp (Date, StudentPlacement, ExamNo, StudentNo, RoomNo)
- ExamNo er fremmednøkkel mot Exam
- StudntNo er fremmednøkkel mot Student
- RoomNo er fremmednøkkel mot ExaminationLocation

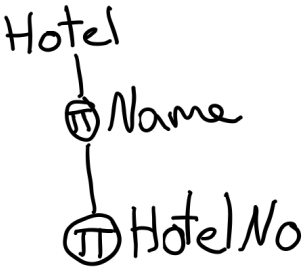
ExaminationLocation (RoomNo, Name, Capacity)

Table (TableNo, Type, RoomNo)
- RoomNo er fremmednøkkel mot ExaminationLocation

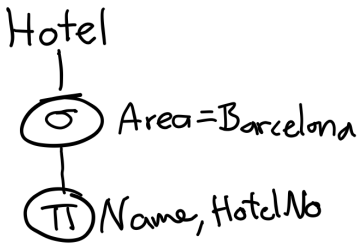
Chair (ChairNo, Type, RoomNo)
- RoomNo er fremmednøkkel mot ExaminationLocation

b)

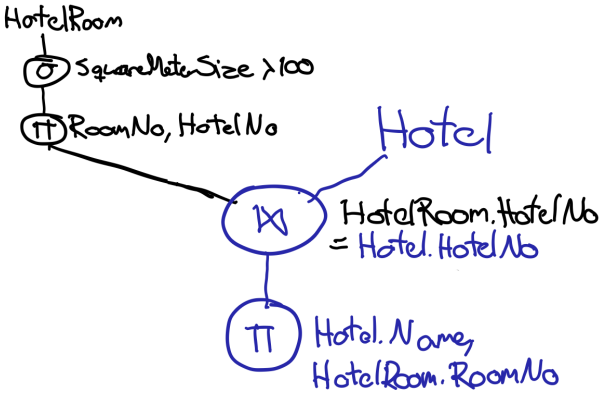
1)


Hotel
π Name
π HotelNo

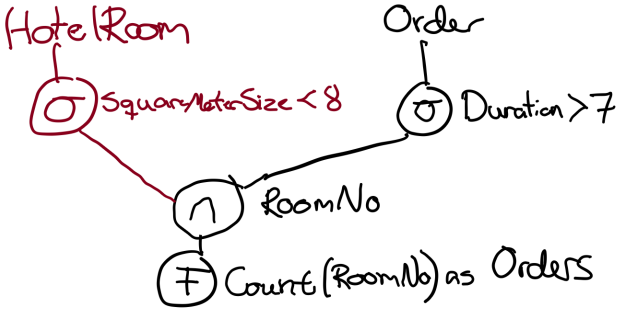
2)


Hotel
σ Area=Barcelona
π Name, HotelNo

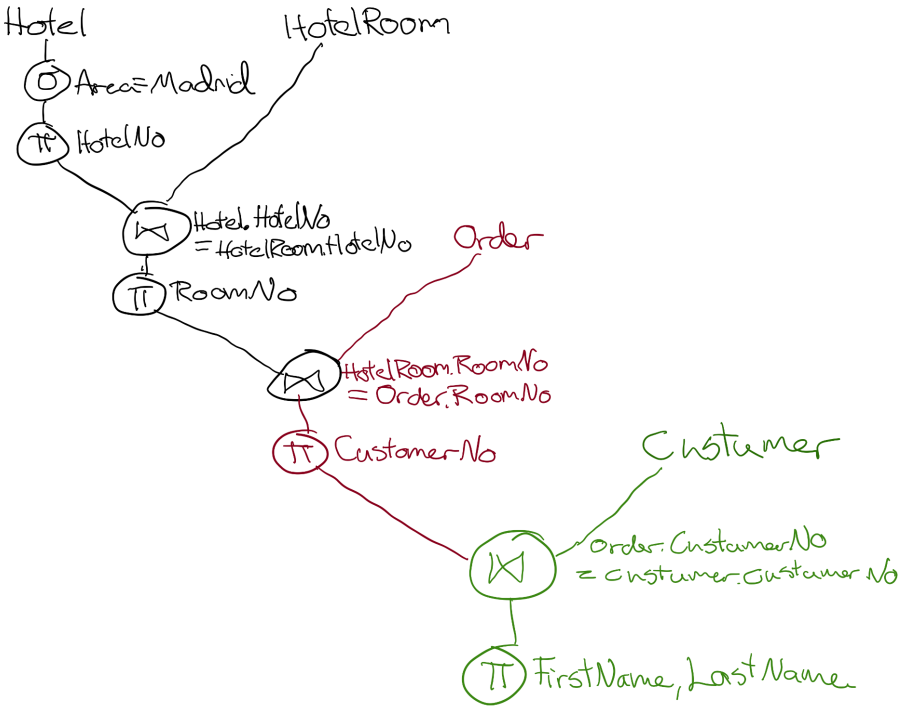
3)


HotelRoom
σ SquareMeterSize >100
π RoomNo, HotelNo
Hotel
⋈ HotelRoom.HotelNo = Hotel.HotelNo
π Hotel.Name, HotelRoom.RoomNo

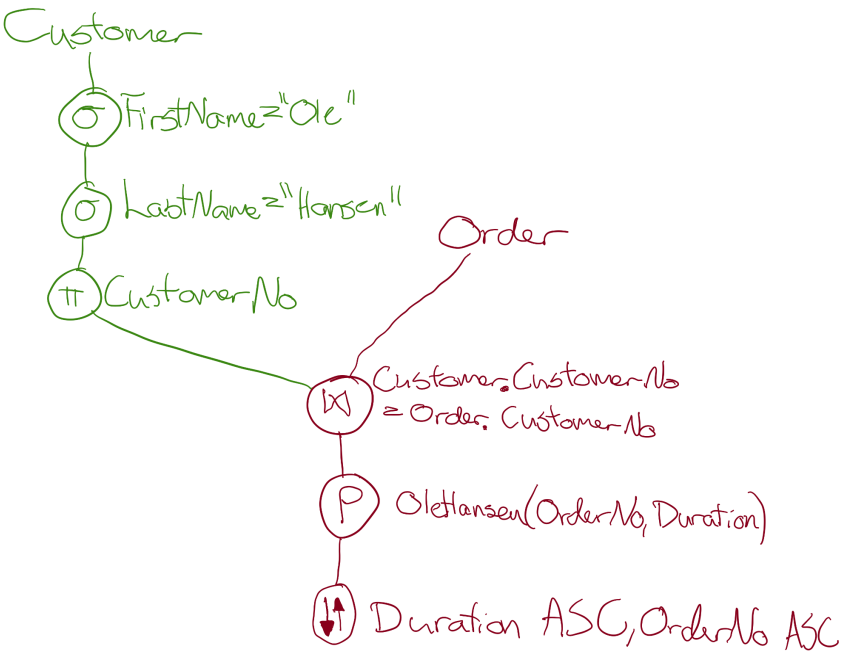
4)


HotelRoom
σ SquareMeterSize < 8
Order
σ Duration > 7
∩ RoomNo
F Count(RoomNo) as Orders

5)


Hotel
HotelRoom
Area=Madrid
HotelNo
Hotel.HotelNo
=HotelRoom.HotelNo
RoomNo
Order
HotelRoom.RoomNo
=Order.RoomNo
CustomerNo
Customer
Order.CustomerNo
=Customer.CustomerNo
FirstName, LastName

6)


Customer
FirstName="Ole"
LastName="Hansen"
CustomerNo
Order
Customer.CustomerNo
=Order.CustomerNo
OleHansen(OrderNo, Duration)
Duration ASC, OrderNo ASC