Øving 1

Oppgave 1: Databasesystemer

1. Forklar kort og overordnet hva en database (DB) og et databasehåndteringssystem (DBMS) er.

En database (DB) er en digital samling av data/informasjon som skal være strukturert slik at dataen er lett tilgjengelig, enkel å administrere og oppdatere.
Et databasehåndteringssystem (DBMS) er en programvaren som samhandler med brukerne slik at de kan administrere systemet og selve databasen.

2. En database og et databasehåndteringssystem kan man tenke seg til sammen utgjør et databasesystem. Det er mange fordeler ved å bruke et databasesystem i motsetning til tradisjonelle filsystemer der hver brukergruppe selv tar hånd om egne filer for å behandle lagring av data. Forklar nærmere hva som menes med følgende tre egenskaper og hvilke fordeler det medfører:

a. Program-data uavhengighet: Denne egenskapen går ut på at brukere kan benytte seg av databasen uten å være bekymret for å endre dataen, slik at det er en uavhengighet mellom program og data. I tillegg vil administratoren kunne endre databasen uten at brukerne mister tilgangen til den nye databasen. Fordelen er at vi unngår å måtte gjøre endringer i databasen ved nye applikasjoner, og endre applikasjoner når strukturen i databasen endres.

b. Flerbrukersstøtte: Går ut på at mange brukere skal kunne få tilgang til å endre de samme underliggende dataene, fra ulike maskiner på et nettverk. Fordelen med dette er at brukere ikke trenger å vente i kø for å få tilgang til databasen, i tillegg vil denne egenskapen hindre muligheten for async lagringsfeil.

c. Selvbeskrivende: Dette vil si at ved hjelp av navn og substruktur skal databasen beskrive hva som gjør/er/betyr hva, gjerne ved bruk av XML. Fordelen med dette er at man får en logisk og brukervennlig struktur som er lett å videreutvikle.

Oppgave 2: ER-modellen

1. Forklar:

a. Forskjellen på en entitet og en entitetsklasse: En entitetsklasse er ikke et objekt eller noe konkret. Det er et set eller en kategori som beskriver bl.a. egenskaper eller attributter til entiteter av dens klasse. Det vil si at entitetene er de konkrete objektene som faller innenfor deres entitetsklasse.

b. Forskjellen på en relasjon og en relasjonsklasse: Her er konseptet det samme som med entiteter og en entitetsklasser. En relasjonsklasse er en slags kategori som beskriver mulige konkrete relasjoner som kan eksistere innenfor den nevnte relasjonsklassen.

c. Hvorfor alle entiteter må ha et eller flere nøkkelattributt: Alle entiteter skal være tilgjengelige for brukeren. Dette er kun mulig dersom en entitet har en attribut som identifiserer den, og at denne attributten er unik. Denne attributten kalles nøkkelattributt. I noen tilfeller vil vi trenge flere nøkkelattributter for å lage en kompositt nøkkel siden et nøkkelattributt ikke alltid kan brukes som primærnøkkel.

2. Betrakt ER-modellen under og avgjør om følgende utsagn kan stemme. For hvert utsagn skal du svare "true", "false" eller "maybe" og gi en kort begrunnelse for svaret ditt. Fyll svarene dine inn i tabellen

1. Taco er en entitetsklasse og hver enkelt instans av en taco identifiseres ved hjelp av nøkkelattributtet TacoID.

True: Dette stemmer siden entitetsklassen er kategorien og en hver instans vil ha en unik nøkkelattributt som tacoen kan identifiseres med.

2. En taco kan inneholde et vilkårlig antall kjøttdeiger, oster, grønnsaker og sauser.

True: Vi ser at kardinalitetene er mang til mange (0, n) når det gjelder taco og de forskjellige tilbehørene.

3. En ordre gjort av en kunde trenger ikke inneholde noen taco.

False: En ordre må innehold (1,n) tacoer, men en taco trenger ikke å ha en ordre (0,n)

4. En ordre gjort av en kunde kan inneholde en million tacoer.

Maybe: Dette stemmer siden kardinaliteten (0, n) på taco, der n > 10^6, men bare dersom databasen har kapasitet til dette.

5. Så fort en ordre er opprettet kan den hentes i en vilkårlig butikk.

Maybe: Så fort en ordre er opprettet vil dette bli lagt inn i databasen som er global pga. **Program-data uavhengighet.** Men selve prosessen som skal til i virkeligheten er usikker. dersom resturanten alltid har alle tacoer på lager vil dette rent hypotetisk sett være sant. Men vanligvis lages maten etter en ordre er opprettet og da bli påstande usant. I tillegg er det en mulighet for at ingen jobber og at maten dermed ikke kan hentes.

6. En kunde kan eksistere i systemet uten noensinne å ha bestilt noen tacoer.

True: Kunden kan ha gjort 0 (0, n) ordre -> 0 tacoer

7. Hver eneste grønnsak-entitet har en vekt.

Maybe: Vi ser at TacoGrønnsak relasjonen har en Vekt attributt, men det betyr ikke nødvendigvis at denne ikke kan være null.

8. En ansatt kan jobbe i ulike butikker med ulike stillingstitler.

True: En ansatt kan ha 1-n jobber-relasjoner hvor relasjons attributten Stillingstittel da også kan variere.

9. Alle ordre delegeres typisk til denne samme personen som er ansatt.

False: Vi ser at en ordre kan delegeres til 0-n forskellige ansatte.

10. En kunde må være registrert med et navn.

False: Siden navn ikke er en nøkkelattributt kan denne attributten bli satt til null.

Oppgave 3: Svake klasser, forekomstdiagram og nye krav:

a) Over ser du et eksempel på en svak klasse. Når kan det være hensiktsmessig med svake klasser? Forklar hva i modellen over som er identifiserende entitetsklasse, identifiserende relasjonsklasse og delvis nøkkel.

Det kan være hensiktsmessig med svake klasser hvis man ikke kan garantere at en primærnøkkel er unik. Alle kinoer vil ha en sal 1,2,3,4 for eksempel og man kan dermed ikke bruke salnummer som en primærnøkkel. Da er det greit å innføre en svak klasse og kombinere salnummeret med kinoen for å danne en primærnøkkel.

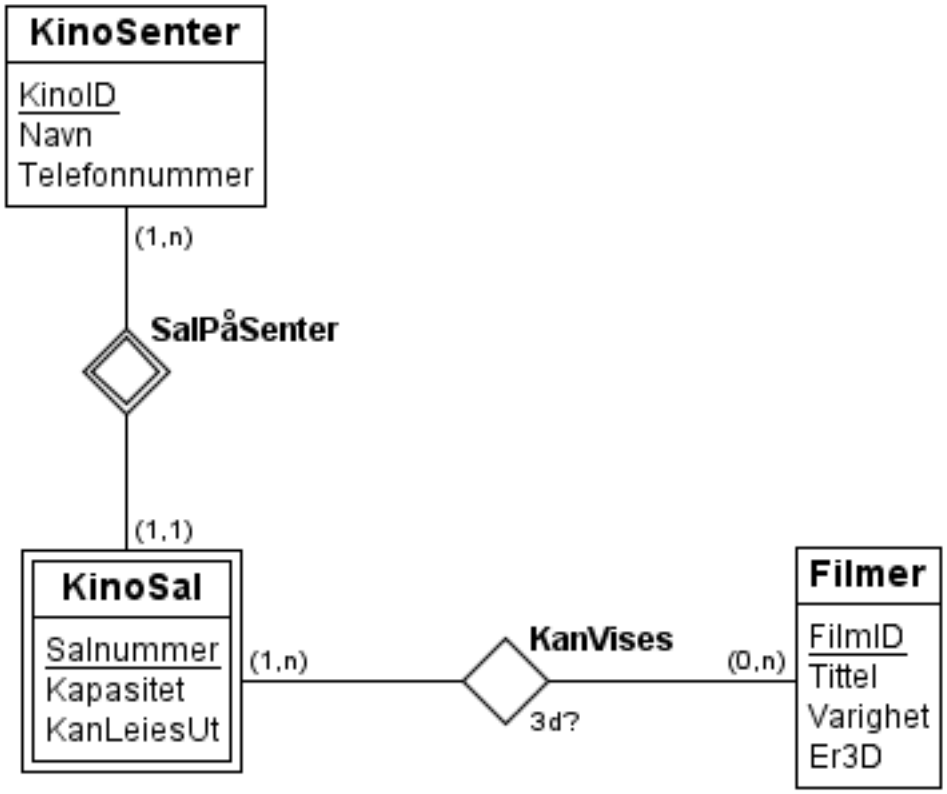
I modellen vil salnummer være den delvise nøkkelen, salPåSenter er da en identifiserende relasjonsklasse og Kinosenter blir da identifiserende entitetsklasse.

b) Hva skjer hvis kardinaliteten fra Kinosal til Kinosenter er (0,1) eller (1,n)? Kan vi fortsatt modelle Kinosal som svak? Hvorfor eller hvorfor ikke?

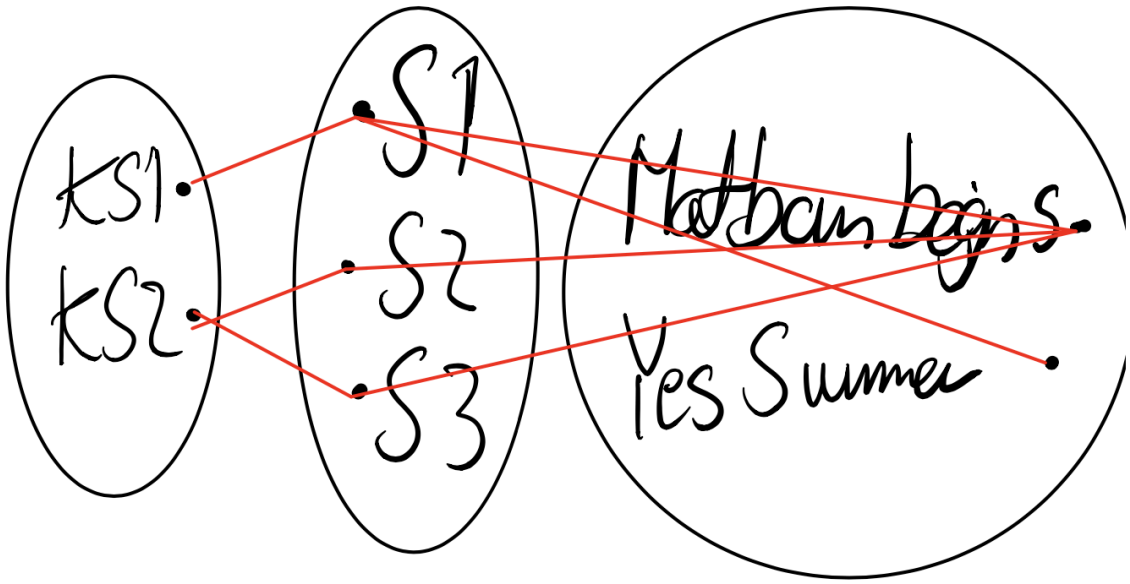
Hvis vi bytter kardinaliteten til (1,n) kan hver kinosal «eksistere» på flere sentere, og en kinosal kan ha ulike salnummer. Dermed vil ikke salnummer fungere som en delvis nøkkel og kinosal-klassen kan ikke settes til å være en svak klasse lenger.

Svake klasser må ha en kardinalitet av (1,1), dersom den har (0, 1) vil det oppstå tilfeller der ScreenRoom objektet eksisterer alene og man dermed ikke har noen måte å finne objektet på.

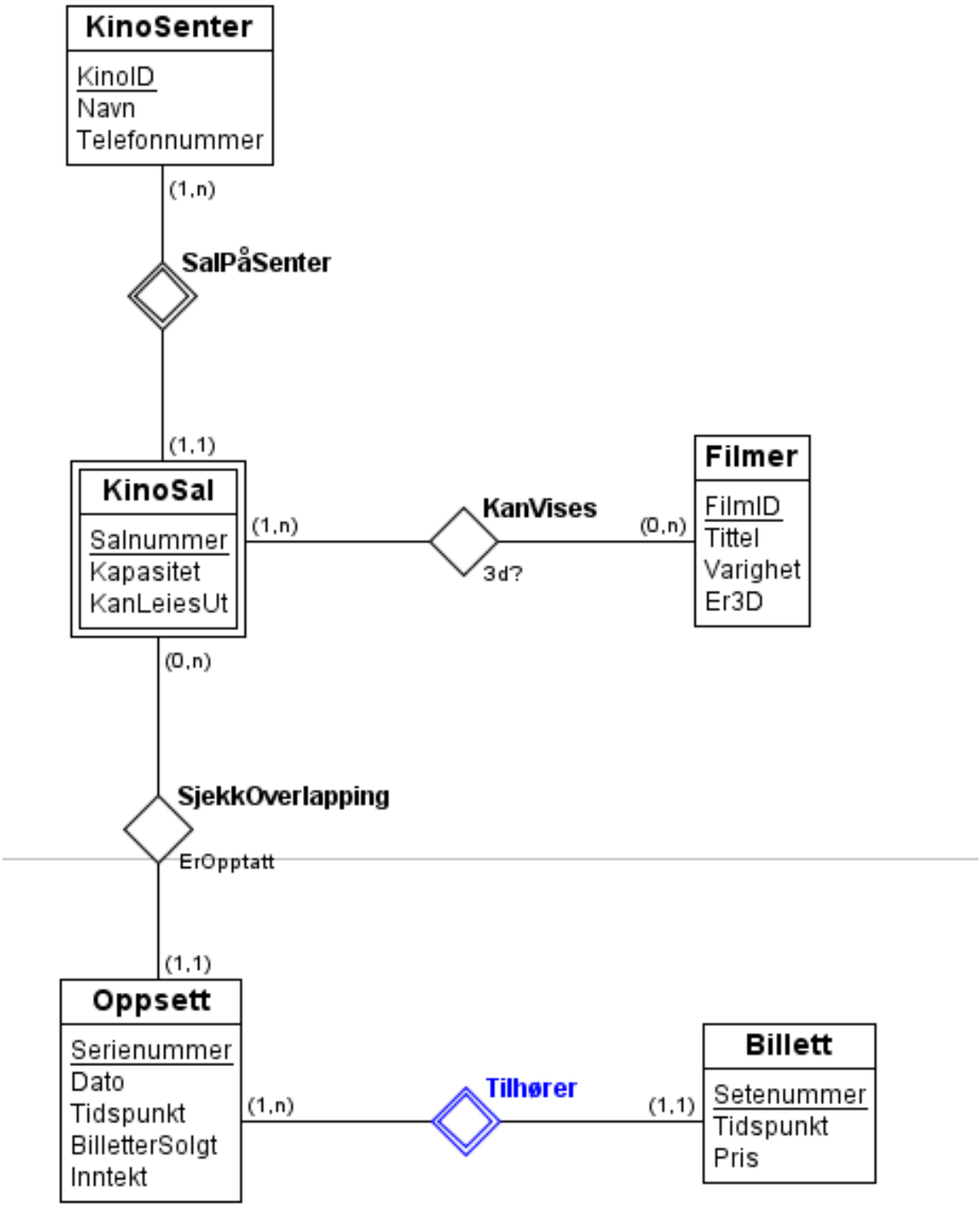
c)


KinoSenter
KinoID
Navn
Telefonnummer
(1,n)
SalPåSenter
(1,1)
KinoSal
Salnummer
Kapasitet
KanLeiesUt
(1,n)
KanVises
3d?
(0,n)
Filmer
FilmID
Tittel
Varighet
Er3D

d)


KS1
KS2
S1
S2
S3
Yes Sumer

e)


KinoSenter
KinoID
Navn
Telefonnummer
(1,n)
SalPåSenter
(1,1)
KinoSal
Salnummer
Kapasitet
KanLeiesUt
(1,n)
KanVises
3d?
(0,n)
Filmer
FilmID
Tittel
Varighet
Er3D
(0,n)
SjekkOverlapping
ErOpptatt
(1,1)
Oppsett
Serienummer
Dato
Tidspunkt
BilletterSolgt
Inntekt
(1,n)
Tilhører
(1,1)
Billett
Setenummer
Tidspunkt
Pris

Oppgave 4: Fra miniverden til ER-modell