Inst. för Data- och Systemvetenskap
SU

Övnings-Tentamen

(delar givna på EIT:DB Databastmetodik 22/2 2014 kl. 10 – 14)

Inga hjälpmedel är tillåtna (annat än ordbok).

Skall även användas som omtentamen för DB:SP/DVK/ATD/DSK/FK samt äldre databaskurser.

Med lösningsförslag (påbörjat) – klart nu.

Skriv bara på en sida av pappret

Skriv läsligt, annars är det omöjligt att rätta tentamen

Studenter som kompletterar FX-uppgift: Skriv att du gör FX-uppgift

Lycka till!

Uppgift 1 (Lärandemål: databasteknik och informationsadministration)

Del 1 till 6 är flervalsfrågor där rätt alternativ ska väljas (bara *ett* är rätt). Välj ***ett*** alternativ för var och en av flervalsfrågorna nedan.

1) Vad innebär redundans i databasen?
a) Att flera kolumner (i olika tabeller) heter samma sak.
b) Att samma information lagras på flera ställen.
c) Att en association/koppling går från och till samma tabell.

2) Vilket av följande påståenden är sant om en klass/entitet?
a) En entitet kan ha flera rekursiva associationer om avbildningsreglerna för båda rollerna i varje association är likadana.
b) En entitet kan ha högst en rekursiv association.
c) En entitet kan ha flera rekursiva associationer generellt (oavsett krav på avbildningsregler för rollerna i associationen).

3) Vilket av följande påståenden är sant om relationen mellan sub-klass och superklass i ett UML klasschema?
a) En subklass kan vara subklass till flera olika super-klasser.
b) En klass A kan vara subklass till en klass B som i sin tur är subklass till A.
c) En superklass måste ha minst två sub-klasser.

4) Vad av följande gäller för 2PL (Two-Phase Locking)?
a) En transaktion kan endast begära ett nytt lås om den inte redan har släppt ett annat lås.
b) En transaktion kan få ett nytt lås trots att den redan släppt ett lås, om den körs mot endast en tabell.
c) En transaktion kan begära ett nytt lås om den först släpper ett lås den fått tidigare.

5) Vilket av följande påståenden gäller för översättning av ett UML klassdiagram till ett relationsschema?
a) Om man översätter ett UML klass-diagram till ett relationsdatabasschema (via de översättningsregler som gäller för att översätta alla delar i klass-diagrammet till relationsscheman) så erhåller man ett antal relations-scheman som är i minst första normalform.
b) Om man översätter ett UML klass-diagram till ett relationsdatabasschema (via de översättningsregler som gäller för att översätta alla delar i klass-diagrammet till relationsscheman) så erhåller man ett antal relations-scheman som är i minst andra normalform.
c) Om man översätter ett UML klass-diagram till ett relationsdatabasschema (via de översättningsregler som gäller för att översätta alla delar i klass-diagrammet till relationsscheman) så erhåller man ett antal relations-scheman som är i minst tredje normalform.

6) Tabellen RIDNING nedan bryter mot en typ av regel, vilken?

HÄST

HästId	HästNamn
1	Man O' War
2	Ego Boy
3	Darley Arabian
4	Byerley Turk

RIDNING

Person	HästId	Datum
Lisa	1	120321
Lisa	1	120321
Lily	4	121231
John	4	121231
Lily	1	121231

RIDNING.HästId << HÄST.HästId

a) Referential integrity.
b) Entity integrity.
c) 2NF.

7) Följande funktionella beroenden gäller för tabellen HÄST nedan:

HästId → HästNamn, Mankhöjd, Färg
HästNamn → Färg

Nu vill man lägga till en rad till tabellen häst; vilken av INSERT-satserna nedan kan exekveras utan att bryta mot något funktionellt beroende?

HÄST

HästId	HästNamn	Mankhöjd	Färg
1	Man O' War	1.67	Röd
2	Ego Boy	1.72	Grå
3	Darley Arabian	1.67	Röd
4	Byerley Turk	1.45	Gyllene

a) INSERT INTO HÄST VALUES(5, 'Byerly Turk', 1.60, 'Röd')
b) INSERT INTO HÄST VALUES(5, 'Ego Boy, 1.65, 'Gyllene')
c) INSERT INTO HÄST VALUES(5, 'Byerly Turk', 1.60, 'Gyllene')

I del 8 ska giltigheten i följande utsaga diskuteras. Om du tycker påståendet är sant så skriv det och motivera sen varför, om du tycker påståendet är falskt så skriv det och motivera varför. Motivera utförligt.

Rätt rad: BCAAABC

8) En vy som innehåller aggregatfunktioner får uppdateras om vissa omständigheter är uppfyllda.

FALSKT. En vy som baseras på aggregatfunktioner kan inte uppdateras i det generella fallet (varför SQL-standarder inte heller tillåter det). Dvs man kan inte göra

UPDATE/INSERT/DELETE på en vy direkt om vy-definitionen innehåller aggregatfunktioner. Detta beror på att aggregatet baserar sig på flera uppgifter i en bas-tabell (eller ibland tom flera bastabeller), om vi ändrar i aggregatet måste denna ändring överföras till motsvarande bastabeller och detta kan ***inte alltid*** göras. Om t ex aggregatet baserar sig på ett genomsnitt av värdet i en viss kolumn för raderna i en bastabell så går det inte att spåra vilka ändringar i de enskilda radernas kolumner som skulle behöva göras för att motsvara andringen i aggregatet.
De vyer som _kan_ uppdateras ska uppfylla flera villkor, bland dem att vy-defintionen _inte_ innehåller aggregatfunktioner, att vydefinitionen baserar sig på en bastabell (innehåller _en_ tabell i FROM-klausulen) mm.

Uppgift 2 (Lärandemål: modellering)

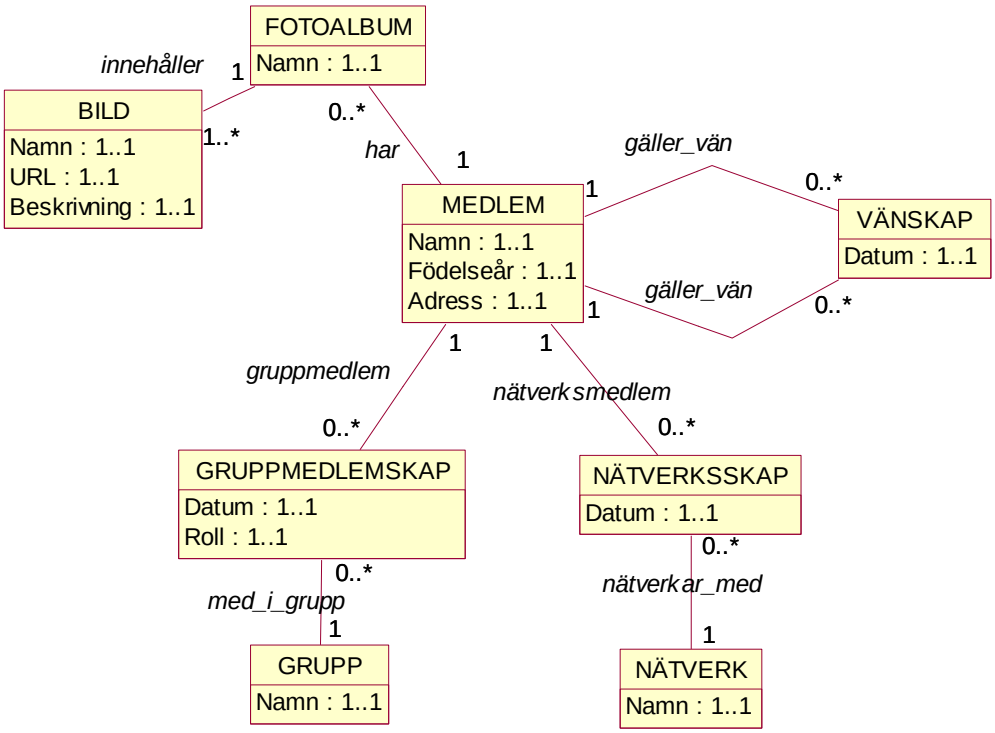
KONSTRUERA ETT KONCEPTUELLT SCHEMA som gör det möjligt att representera följande typer av fakta:

En av de mest besökta webbsajterna idag är Facebook, där deltagarna kan beskriva sig själva, delta i grupper och nätverk, relatera till andra användare, använda applikationer och utbyta meddelanden. Konstruera ett konceptuellt schema som gör det möjligt att representera följande fakta:

Per Persson är född 21 jan 1977 och bor i Stockholm
Per Persson gick med i nätverket "Sweden" 12 jan 2007
Eva Svensson är född 21 jan 1977 och bor i Oslo
Eva Svensson gick med i nätverket "Sweden" 12 jan 2006
Per Persson blev vän med Eva Svensson i Facebook 23 mars 2007
Per Persson blev vän med Olle Persson 7 januari 2014.
Per Persson har ett fotoalbum som heter "Födelsedag 2005"
Per Persson har i det fotoalbum som heter "Födelsedag 2005" en bild med bildtexten "Min födelsedagstårta"
Per Persson skapade en grupp som heter "Karlstads studenter" 20 juni 2005 och blev då administratör för den gruppen
Olle Persson är också administratör för gruppen "Karlstads studenter"
Eva Svensson är medlem av gruppen "Karlstads studenter"
Eva Persson gick med i nätverket ”DSV Alumni” 11 januari 2014.

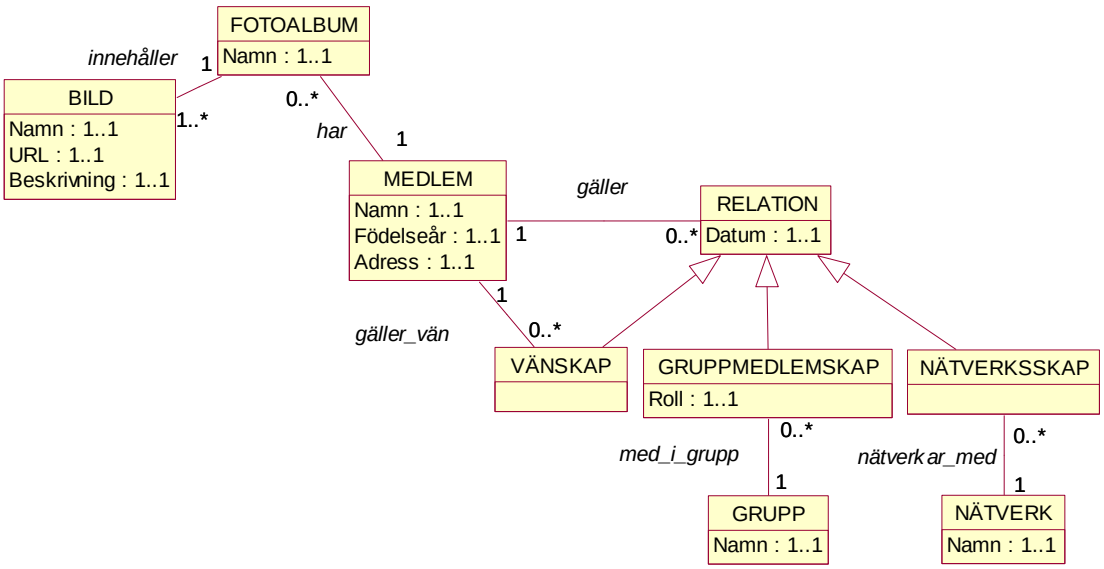
Modellera på ett sådant sätt att partiella attribut (attribut med min-värde = 0) undviks!

Lösningsförslag:



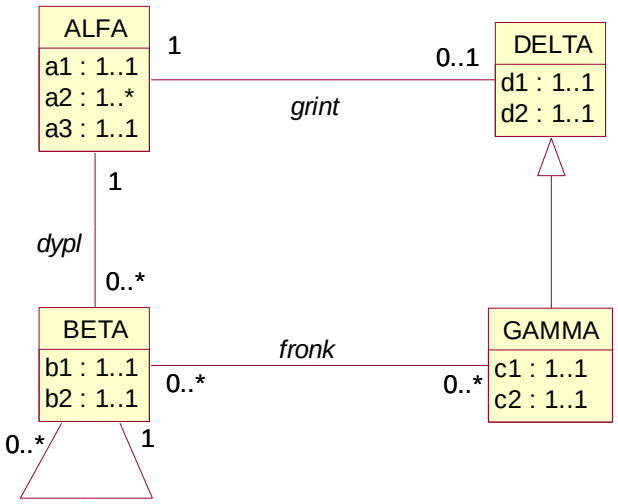
Måste man ha med både GRUPP-MEDLEMSKAP och NÄTVERKSSKAP ? – Räcker det inte med en av dem? Om man vill undvika partiella attribut behövs båda! I annat fall blir attributet 'Roll' som nu ligger i gruppmedlemskap partiellt (= minvärde är 0) eftersom man endast verkar ha roller i grupper – inte i nätverk. Av texten att döma. Man kan naturligtvis anse att det står väldigt lite om roller i grupper i texten och att detta inte är centralt – och det stämmer att detta inte är i fokus – det är inget jättefel att inte ta med både gruppmedlemskap och nätverksskap – men det är bättre att göra det.

Alternativ lösning med isa-relationer för att inte behöva upprepa alla datum för de olika "SKAP"-en (NÄTVERKSKAP, GRUPPMEDLEMSKAP, VÄNSKAP) och inte heller behöva upprepa den ena associationen mot MEDLEM:



Uppgift 3 (Lärandemål: Syntetisk databasdesign (översättning från konceptuell modell till logiskt relationsdatabasschema))

Betrakta följande konceptuella schema.



ALFA identifieras av attributen a1 och a3 tillsammans.
BETA identifieras av sin association till ALFA tillsammans med attributet b1.
GAMMA identifieras av sin association mot DELTA.
DELTA identifieras av attributet d1

ÖVERSÄTT DET KONCEPTUELLA SCHEMAT OVAN TILL ETT RELATIONSDATABASSCHEMA. Ange för varje relation vad som utgör primärnyckel samt vad som eventuellt utgör främmande nycklar (främmande nycklar måste specificeras med alla korrekta kolumner). I fallet främmande nycklar skall även specificeras mot vilken relation de utgör främmande nyckel. Översättningen *får ej* innebära att avsteg från den konceptuella modellen görs (annat än de avsteg som *måste* göras för att realisera relationsmodellen). **Surrogatnycklar får inte införas**.

Använd följande notation med understrykning av primärnycklar:

OMEGA(o1, o2, o3),
PI(p1, o1, o2, p2)

Tabellen PI innehåller två attribut o1 och o2 som utgör främmande nyckel mot tabellen OMEGA. Detta skrivs på följande sätt:

PI.(o1, o2) utgör FN mot OMEGA.(o1, o2)

Lösningsförslag:

ALFA(a1,a3)

A2(a1, a3, a2), där A2.(a1, a3) utgör FN mot ALFA.(a1, a3)

BETA(a1, a3, b1, b2, a11, a33, b11),
där BETA.(a1, a3) utgör FN mot ALFA.(a1, a3) och BETA.(a11, a33, b11) utgör FN mot BETA.(a1, a3, b1)

DELTA(d1, a1. a3, d2), där DELTA.(a1, a3) utgör FN mot ALFA.(a1, a3)

GAMMA(d1, c1, c2), där GAMMA.d1 utgör FN mot DELTA.d1

FRONK(a1, a3, b1, d1), där FRONK.(a1, a3, b1) utgör FN mot BETA.(a1, a3, b1) och FRONK.d1 utgör FN mot GAMMA.d1

Kommentarer:

A2 blir en egen tabell pga att attributet a2 i ALFA var flervärt, varje rad i denna nya tabell A2 innehåller då dels kolumnen a2 samt främmande-nyckel-kolumnerna mot ALFA dvs a1 och a3.

GAMMA har ju en isa-relation mot DELTA – detta blir en främmande nyckel mot DELTAs primärnyckel. I detta fall finns inget angivet om att GAMMA har någon egen identifierare – i så fall får GAMMA samma primärnyckel som DELTA, dvs främmande nyckeln mot DELTA får även tjäna som primärnyckel i GAMMA.

BETA har en rekursiv relation mot sig själv. Avbildningsreglerna för de båda rollerna i relationen är 0..* och 1..1, dvs vi kan låta BETA innehålla en främmande nyckel till sig själv. Det går även bra att bryta ut den rekursiva relationen till en egen tabell om man så vill:

B2(a1, a3, b1, a11, a33, b11), där B2.(a11, a33, b11) är FN mot BETA.(a1, a3, b1). I så fall kommer BETA att se ut så här:

BETA(a1, a3, b1, b2) och inte längre ha någon FN mot sig själv.

FRONK, slutligen, är en relationstabell som infördes eftersom BETA och GAMMA hade en många-många relation mellan sig. Denna relationstabell innehåller enbart nyckel-kolumnerna för BETA och GAMMA tillsammans.

Uppgift 4 (Lärandemål: Analytisk databasdesign (normalisering))

Relationsschemat PLISP har 5 kolumner:

PLISP(**wogf, greng**, winini, meeg, klosp)

(primärnyckeln är understruken och tryckt i fetstil)

FÖLJANDE FUNKTIONELLA BEROENDEN RÅDER:

wogf, greng → winini, meeg, klosp
greng →winini
meeg →klosp

a) PLISP är i 1 NF. Normalisera till 3NF. Kommentera varje steg, d.v.s. ange vad som är skälet till att en nedbrytning gjorts från t ex 1NF till 2NF. Hoppa inte över några steg utan gör bara den nedbrytning som behöver för att komma från 1NF till 2NF och sedan (i nästa steg) det som behövs för nästa högre normalform o.s.v. Om tabellen redan är i 2NF eller 3NF så ange varför detta gäller.
b) Motivera varför en högre normalform är att föredra framför en lägre. Finns det några skäl att inge normalisera?

Lösningsförslag:

Vi har ett partiellt funktionellt beroende mellan kolumnen ’winini’ och primärnyckeln ’wogf, greng’. Detta pga att följande gäller:

{wogf, greng} → {winini, meeg, klosp} (var givet i uppgiftstexten)

Armstrong’s dekomponeringslag ger att följande också gäller:

{wogf, greng}→ {winini}

Vi har dessutom att

{greng} → {winini}. Eftersom {greng}är en delmängd av {wogf, greng} så gäller att kolumnen ’winini’ är partiellt bestämd av primärnyckeln (dvs ’wogf, greng’). Den är även direkt bestämd av primärnyckeln – precis som alla andra icke-nyckel-attribut.

Eftersom vi har ett partiellt funktionellt beroende mellan kolumnen ’winini’ och primärnyckeln ’wogf, greng’ så är tabellen inte i 2NF.

Vi skapar en ny tabell i 2NF:

PLISP’(**greng**, winini)

och det som är kvar i den gamla tabellen (också 2NF nu) ser ut så här:

PLISP(**wogf, greng**, meeg ,klosp) där PLISP.greng är FN mot PLISP'.greng

Nu ska båda tabellerna analyseras med avseende på om de är i 3NF eller ej:

PLISP' är i 3NF eftersom vi bara har ett icke nyckel attribut – det kan inte finnas några transitiva beroenden mellan icke nyckel-attributet och primärnyckeln!

PLISP är däremot inte i 3NF pga att vi har ett transitivt beroende mellan kolumnen 'klosp' och primärnycklen. Detta pga att följande gäller:

{wogf, greng} → {winini, meeg, klosp} (var givet i uppgiftstexten)

Armstrong's dekomponeringslag ger att följande också gäller:

{wogf, greng}→ {meeg}

Vi har dessutom att

{meeg} → {klosp}. Vi ser då (genom att tillämpa Armstrongs transitiva lag) att kolumnen 'klosp' är transitivt bestämd av primärnyckeln. (Den är också direkt bestämd av primärnyckeln – precis som alla icke nyckel-attribut).
Med andra ord, PLISP är inte i 3NF eftersom det finns ett icke nyckel-attribut som är transitivt bestämd av primärnyckeln.

Vi skapar en ny tabell i 3NF:

PLISP''(**meeg**, klosp)

och det som är kvar i den gamla tabellen (också 3NF nu) ser ut så här:

PLISP(**wogf, greng**, meeg) där PLISP.meeg är FN mot PLISP''.meeg

b)
Man normaliserar för att undvika redundans, varje faktum ska lagras på ett ställe, inte spridas över fler tabeller (med undantag av kontrollerad redundans i form av främmande nycklar). Detta för att undvika uppdateringsanomalier men också för att spara plats då ennormaliserad databas tar oftast mindre plats (pga att redundansen minskar). Bryr man sig bara om att minska redundansen så är en högre normalform bättre än en lägre. Ett normaliserat databasschema kommer dock att innehålla fler tabeller än ett onormaliserat, det blir alltsåmer komplext och eventuellt svåröverskådligt. Hög normaliseringsgrad kan också leda till att transaktioner tar längre tid att genomföra pga att fler joins mellan de många nya tabellerna måste göras. Är det en ofta förekommande transaktion (-styp) som tar längre tid är detta en stor nackdel och man kan i optimeringssyfte välja att de-normalisera.

Uppgift 5 (Lärandemål: frågespråk (SQL och relationsalgebra))

Betrakta databasschemat nedan som innehåller två tabeller: FASTIGHET och ARRENDE
Följande främmande nyckel-reglel gäller:

ARRENDE.Fid << FASTIGHET.FastighetsId
(det vill säga kolumnen Fid i tabellen ARRENDE utgör främmande nyckel mot primärnyckeln FastighetsId i tabellen FASTIGHET)

(Primärnycklarna i tabellerna nedan är i fetstil och understrukna).

a) Översätt följande till SQL och relationsalgebra: Vilka fastigheter har arrenderats av personen med pnr = 11111? ?Ange fastigheternas id och deras areal.

b) Översätt följande till SQL och relationsalgebra: Vilka personer har arrenderat alla fastigheter?

c) Vad blir resultatet (rita tabell med rader) SQL- uttryck exekveras (basera svaret på relationerna nedan)?:

```
SELECT Pnr, count(Fid) AS Antal
FROM ARRENDE
GROUP BY Pnr
HAVING count(Fid) > 1
```

ARRENDE

Pnr	**Från**	Till	**Fid**
11111	00-01-01	04-06-21	'Hanstavik 2:4'
11111	07-03-21	04-12-31	'Ekeby 1:1'
22222	00-01-01	04-06-21	'Hanstavik 2:5'
11111	05-01-01	07-12-31	'Hanstavik 2:6'
33333	00-01-01	07-12-31	'Stjärna 1:1'
55555	98-01-01	99-12-31	'Hanstavik 2:4'

FASTIGHET

FastighetsId	Areal
'Hanstavik 2:4'	100
'Ekeby 1:1'	50
'Hanstavik 2:5'	100
'Hanstavik 2:6'	50
'Stjärna 1:1'	200
'Tuna: 4:8'	50

Relationsalgebraiska operatorer:

Selektion σ , Projektion π , Theta-join θ , Natural join ⋈ , Kartesisk produkt □

Division ÷, Snitt ∩ , Union ∪ , Differens -, Aggregering/gruppering G $_{\mathcal{F}}$

Tilldelning ←, Namnändring ρ

SQL-syntax:

```
SELECT [DISTINCT] <attributlista>
FROM <tabellista>
[WHERE <villkorsuttryck>]
[GROUP BY <kolumnlista>
        [HAVING <villkorsuttryck>]]
[ORDER BY <kolumnlista>]
```

Lösningsförslag:

a)

```
SELECT FastighetsId, areal
FROM FASTIGHET, ARRENDE
WHERE FastighetsId = Fid
AND Pnr = '11111'
```

$\Pi_{\text{FastighetsId, Areal}}\, \sigma_{\text{Pnr='11111'}}\, (\text{FASTIGHET} \,|X|_{\text{FastighetsId=Fid}}\, \text{ARRENDE})$

b)

```
SELECT A1.Pnr                                   /* Start på täljaren */
FROM ARRENDE A1
WHERE NOT EXISTS
        (SELECT FastighetsId                    /* Nämnaren, alla fastigheter */
        FROM FASTIGHET
        WHERE FastighetsId NOT IN
                (SELECT A2.Fid                  /* Återkopplning till täljaren */
                FROM ARRENDE A2                 /* dvs alla fastigheter som A1.Pnr */
                WHERE A2.Pnr=A1.Pnr))           /* arrenderat */
```

Täljare ← $\Pi_{\text{Pnr, Fid}}$ ARRENDE

Nämnare (Fid)← $\Pi_{\text{FastighetsId}}$ FASTIGHET

SVAR ← Täljare KVOT NÄMNARE

c)

Pnr	Antal
11111	3

(dvs två kolumner i tabellen och en enda rad som instansierar tabellen).