Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen

EIT:DB Databastmetodik

11/1 2013 kl. 13 – 17 + Lösningsförslag

Inga hjälpmedel är tillåtna (annat än ordbok). Kort syntaxsamling för delar av SQL samt lista med symboler för relationsalgebraiska operatorer finns sist i tentamen.

Skriv bara på en sida av pappret
Skriv namn på varje papper
Skriv läsligt, annars kan tentamen inte rättas

Lycka till!

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen EIT:DB 11 januari 2013

Uppgift 1 (svarar mot lärandemål 1 avseende 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?

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

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.

Rätt lösningsrad: b, c, a, a, a, b

Kommentar: Fråga 5 är möjligen öppen för tolkningsmöjligheter. Som rör tillståndet för UML-diagrammet, om man t ex använt modelleringsmönster av typen power-types och reifieringar så ökar normaliseringsgraden för den databas som blir resultatet om man översätter UML-schemat till ett logiskt databasschema – det är större sannorlikhet att hamna i högre normalform t ex 3NF. Men någon garanti är det inte, normalformer har ju att göra med funktionella beroenden mellan attributen/kolumnerna INOM det man modellerat som en klass eller tabell och alla sådana beroenden modelleras inte med nödvändighet bort via modelleringsmönster. Mao, det enda man är garanterad att hamna i är i 1NF om man följer reglerna för översättning från UML klasschema till ett logiskt databasschema, dvs en samling relationsscheman.

I del 7 och 8 ska giltigheten i följande utsagor 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.

7) Högsta möjliga normalform är alltid att föredra för alla tabeller.

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 alltså mer 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.
Kommentar: Både sant och falskt kan godtas här under förutsättning att man motiverat via något liknande texten ovan.

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen EIT:DB 11 januari 2013

8) Det är viktigt att en databas-transaktion är så kallat 'atomär'.

Att en databastransaktion är atomär innebär att den utförs i sin helhet eller inte alls. Om så inte vore fallet skulle databasen kunna hamna i ett inkonsistent tillstånd om tranaktionen avbryts oväntat mitt i. Anta t ex att en transaktion består av att först ändra ett värde i en tabell för att sen uppdatera en annan tabell som innehåller ett deriverat attribut som räknas fram just från det värde som ändrades i den första tabellen. Låt oss t ex ha en tabell som heter ART och som har ett attribut 'medelvikt', där medelvikten räknas fram baserat på värden i en annan tabell: DJUR som har ett attribut 'min_vikt'. Om vi nu ändrar några DJUR:s vikt så vill vi ju att ART:s medelvikt korrekt ska avspegla tillståndet i databasen – medelvikten för en ART är lika med att addera ihop alla DJUR:ens vikt och dela den med antalet DJUR. Uppdaterar vi då några DJUR:s vikt UTAN att uppdatera även ART:ens medelvikt (t ex pga ett ström-avbrott när vi gjort några DJUR-uppdateringar) så kommer vi att ha ett felaktigt värde på ART:ens medelvikt. Transaktions-begreppet ger oss en möjlighet att gruppera ihop operationer mot databasen till en odelbar enhet som antingen utförs i sin helhet eller inte alls. Kraschar databasen mitt i DJUR-uppdateringarna, dvs när bara en del av vår transaktion utförts (med andra ord innnan en så kallad COMMIT gjorts) så kommer dessa DJUR-uppdateringar att "rullas tillbaka" dsv återföras till det tillstånd som rådde innan man började köra transaktionen överhuvudtaget.

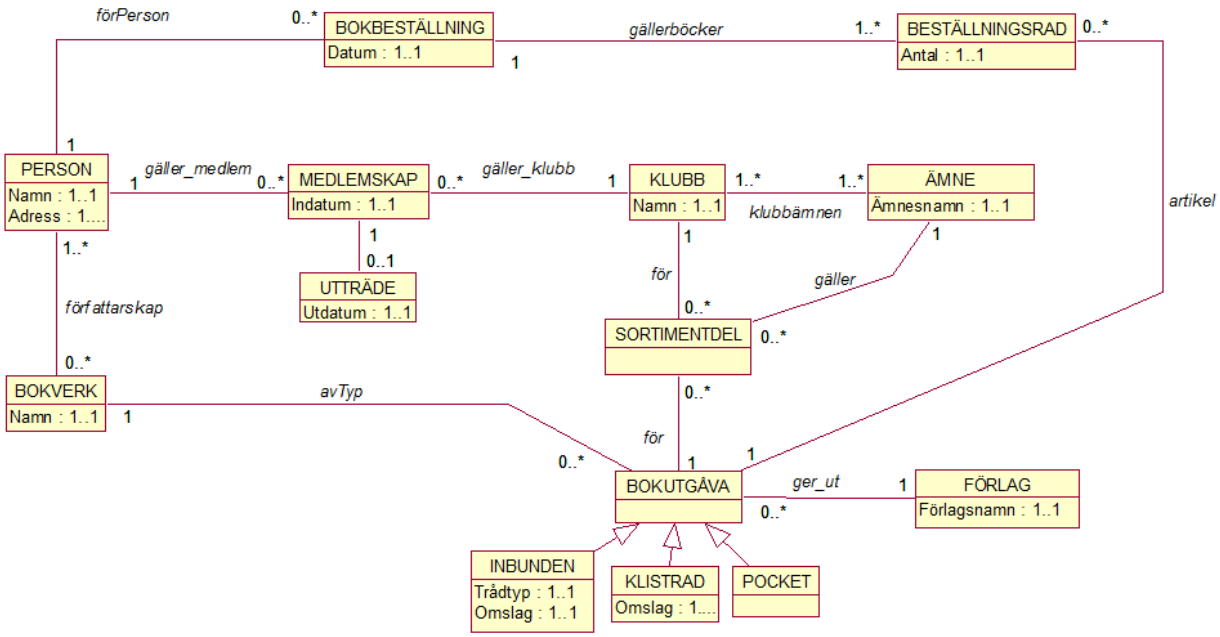
Uppgift 2(svarar mot lärandemål 2 avseende konceptuell modellering och databasmodellering)

Konstruera ett konceptuellt schema som ger möjlighet att representera samtliga utsagor nedan. Ange avbildningsregler för samtliga attribut och för samtliga roller i associationer:

- NYTTA & NÖJE är en bokklubb som har många medlemmar.
- Jane Marple gick in i NYTTA & NÖJE 21 mars 1965.
- Alexander Asimov gick inte in (i NYTTA & NÖJE) förrän 21 mars 1973.
- Varje bok som finns i NYTTA & NÖJE:s sortiment är klassad att tillhöra ett ämnesområde. Varje bok har dessutom en titel, en eller flera författare och ges ut av ett förlag. Böcker kan ges ut som inbundna, klistrade eller som pocket. För inbundna böcker håller man reda på vilken typ av tråd som använts till bindningen. För klistrade och inbundna lagrar man information om vilken typ av hårt omslag som använts.
- Hober Mallow är författare till boken "Databaser utan tårar" tillsammans med Jane Marple. Jane är även författare till boken "Conceptual Modelling- quick and easy".
- Gunvald Larsson har just beställt tre exemplar av "Databaser utan tårar" (pocket) och ett exemplar av "Allt du vill veta om rosor" (inbundet) från bokklubben NYTTA & NÖJE.
- Jane Marple bor på Drottninggatan 22 i Ängelholm.
- Jane Marple beställde ett exemplar av "Stora Svamp-boken", inbunden, från NYTTA & NÖJE 21 augusti 1996.
- Jean Valjean lämnade NYTTA & NÖJE 21 mars 1977.

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen EIT:DB 11 januari 2013



Kommentarer: Många tolkningsmöjligheter och korrekta scheman finns. Några exempel: ska man modellera bokklubbar generellt för att kunna lägga in många olika klubbar eller ska man betrakta uppgiften som att modellera ett informationssystem för just NYTTA & NÖJE? Båda delar är möjligt eftersom texten bara ger en klubb som exempel. I schemat ovan är den generella lösningen vald, dsv en klass KLUBB finns angiven. Begreppet BOK i texten ovan är en homonym med minst två (ev. fler) olika betydelser som båda behövs i schemat. Dels finns det BOKVERK som representerar det litterära verket, t ex verket 'Stiftelsen och Imperiet' av Isaac Asimov eller 'Les Miserables' av Victor Hugo. Böcker kan ges ut som utgåvor, därav klassen BOKUTGÅVA som har egna egenskaper (skilda från det litterära verket), t ex kan samma bokverk ges ut av olika förlag eller ges ut som inbunden etc. Om bokklubbar ska anses ha litterära verk eller utgåvor i sitt sortiment kan man välja, här valde jag att koppla KLUBB till BOKUTGÅVA. Vilket i sin tur fick till följd att det behövdes ett relationsobjekt mellan KLUBB, ÄMNE och BOKUTGÅVA för att knyta de böcker som klubben har listat till olika ämnen. Här kan man lika gärna (eller tom bättre) ha relations-objektet kopplat till KLUBB, ÄMNE och BOKVERK. Personer kan, vidare, beställa böcker från bokklubben. Här finns en tredje typ av bok, den beställda, som en klass BESTÄLLNINGSRAD kopplat till en BESTÄLLNING. Texten antyder att man kan beställa flera böcker samtidigt i en beställning och eventuellt i olika antal, därför behövs både en BESTÄLLNING-klass plus en BESTÄLLNINGSRAD (aka beställd-bok) för att kunna ange hur många exemplar av viss bok man vill ha i en viss beställning. Slutligen kan PERSON:er författa böcker, jag bara relaterade PERSON till BOKVERK för att modellera detta. Alternativt kan man ha en sub-klass till PERSON som heter FÖRFATTARE, jag fann dock inga egna egenskaper för denna klass jämfört med PERSOn (annat än relationen till BOKVERK) så denna klass är inte helt nödvändig. Några har modellerat FÖRFATTARSKAP som en klass och då lagt till datum för skapandet av boken, bra men inte helt nödvändigt med utgångspunkt från de uppgifter som står i texten.

Uppgift 3 (svarar mot lärandemål 4 avseende analytisk databasdesign)

Betrakta följande relationsschema:

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen EIT:DB 11 januari 2013

R(A, B, C, D, E, F, G)

Följande funktionella beroenden gäller:

AD → EF

A → F

F → G

a) Bestäm primärnyckel för R.

En primärnyckel ska bestämma alla övriga attribut funktionellt (direkt eller indirekt). Vi ser att E och F är bestämda av AD, G är i sin tur bestämd av F, dvs G är också bestämd av AD eftersom AD bestämde F. B och C i sin tur är inte bestämda av något annat attribut. En möjlig primärnyckel är därför ABCD.

b) Normalisera R till 3NF. Motivera dina svar.

Vi förutsätter att tabellen är i minst 1NF. För att tabellen även ska vara i 2NF måste alla icke nyckel-attribut vara fullständigt funktionellt bestämda av hela primärnyckeln, dvs inga partiella beroenden mellan nyckeln och icke-nyckel attribut får existera. Här finns dock flera sådana, vi ser att både E och F bestäms av bara en del av nyckeln, dvs F och E är funktionellt bestämd dels av {ABCD}, men enbart även av mindre del av nyckeln, dvs {AD} dsv vi har ett partiellt funktionellt beroende mellan primärnyckeln och EF, dvs vi bryter mot 2NF. Även G är funktionellt bestämd av både nyckeln och AD vilket man kan se genom att tillämpa transitiva lagen (ett av Armstrong's axiom):

Vi har att AD → EF, detta kan vi (genom dekomponeringslagen) skriva om som att
AD → E OCH AD → F

Sen har vi att F bestämmer G (givet i uppgiftstexten). Nu har vi alltså att AD → F och F → G, dvs, via tillämpning av transitiva lagen, så gäller att AD→ G

Summa: Både E, F och G är bestämda av bara en del av nyckeln, nämligen AD:

Lösningen är att lägga AD och EFG i en egen tabell, då får vi:

R(ABCD), där R.(A, D) utgör främmande nyckel mot R'.(A, D)
R'(ADEFG)

Nu är R i 2 NF (och även i 3NF eftersom vi inte har något beroende mellan icke-nyckelattribut) men vi måste fortsätta att analysera R' mot övriga funktionella beroenden. A → F och både A och F ingår ju i R', dsv A är en determinant map denna tabell. Eftersom F är bestämd av bara en del av nyckeln så är inte R' i 2NF. Även G är (av samma skäl som anges ovan) beroende av bara en del av nyckeln eftersom F → G gäller. Vi bryter ut A och F och G till en egen tabell, nu har vi:

R(ABCD), där R.(A, D) utgör främmande nyckel mot R'.(A, D)
R'(ADE), där R'.A « R''.A
R''(AFG)

Nu är även R' i både 2 och 3NF eftersom vi bara har ett icke nyckelattribut, dsv inga beroenden mellan icke-nyckelattribut kan ge upphov till transitiva beroenden mellan icke-nyckelattribut och nyckeln. Vi måste dock fortsätta att analysera R'', finns det något beroende mellan icke-nyckel attribut som kan ge upphov till ett transitivt beroende mellan nyckeln och icke-nyckelattribut? Ja det gör det ju, icke-nyckelattributet F bestämmer icke-nyckelattributet G. Då har vi alltså att nyckeln bestämmer G och nyckeln bestämmer F och F bestämmer G (mao A → F och F → G), vi har alltså ett transitivt beroende mellan nyckeln och ett icke nyckel-attribut, dvs vi bryter mot 3NF. Lösningen är att lägga F och G i en egen tabell, vi har nu:

R(ABCD), där R.(A, D) utgör främmande nyckel mot R'.(A, D)
R'(ADE), där R'.A utgör främmande nyckel mot R''.A
R''(AF), där R''.F utgör främmande nyckel mot R'''.F
R'''(FG)

Kommentar: uppgiften kan ge upphov till följdfel om man bestämmer en felaktig primärnyckel i första delen, detta tas hänsyn till vid rättningen, dsv om man motiverat sina normaliseringar korrekt map den felaktiga nyckeln vägs det in i bedömningen. Notera dock att detta potentiellt innebär en kraftig förenkling av uppgiften, även detta kan komma att vägas in i bedömningen.

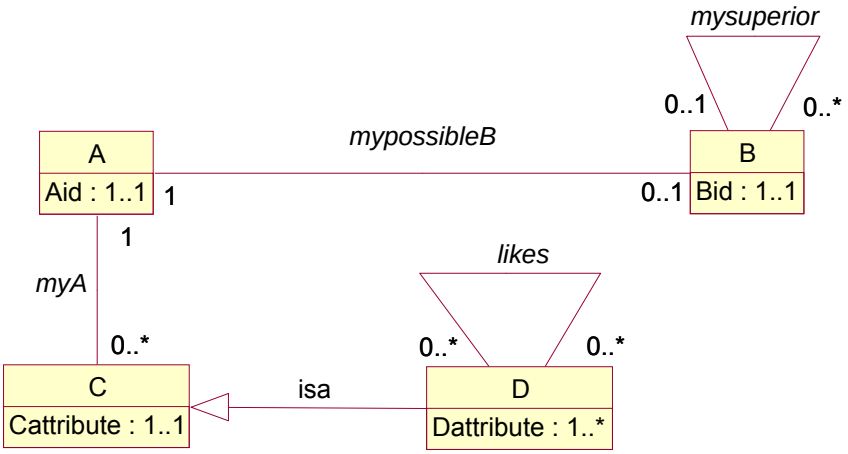
Uppgift 4 (svarar mot lärandemål 3 avseende syntetisk databasdesign)

Översätt det konceptuella schemat nedan till ett relationsdatabasschema. Ange alla primärnycklar och främmande nycklar. Vid översättningen får inga förändringar av det konceptuella schemat

införas (annat än de som innebär att en översättningsregel tillämpas). Surrogatnycklar får inte införas (om de inte redan finns i schemat).
Använd följande notation:

TABELL1(Prim1, Prim2, Kolumn1), där (Prim1, Prim2) är primärnyckel för TABELL1
TABELL2(Prim, For1, For2), där Prim är primärnyckel för TABELL2 och där (For1, For2) utgör främmande nyckel mot TABELL1.

Det sista kan även skrivas TABELL2.(For1, For2) << TABELL1.(Prim1, Prim2)



A identifieras av sitt AId
B identifieras av sitt Bid.
C identifieras av sitt Cattribute plus sin relation mot A.
D identifieras av identifieraren för dess supertyp.

A(**Aid**)
B(**Bid**, mittA), där B.mittA << A.Aid
MySuperior(**B1, B2**), där mySuperior.B1 << B.Bid och mySuperior.B2 << B.Bid
C(**Cattribute, myA**), där C.myA << A.Aid
D(**C, A**), där D.(C, A) << C.(Cattribute, myA)
Dattributetable(**C, A,** Dattribute), där Dattributetable.(C, A) << D.(C, A)
Likes(**C1, A1, C2, A2**), där Likes.(C1, A1) << D.(C, A) och Likes.(C2, A2) << D.(C.A)

7 tabeller totalt alltså. Man kan dock få 6 istället om man väljer att översätta associationen my-Superior som en främmande nyckel i tabellen B. Det fungerar men är något sämre eftersom denna främmande nyckel i detta fall kommer att kunna vara NULL. Noteras bör också att vi låter B relatera till A via en främmande nyckel istället för tvärtom för att undvika NULL.

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Uppgift 5 (svarar mot lärandemål 5, frågespråk mot relationsmodellen)

Betrakta följande relationsscheman:

BIOGRAF(**namn**, adress)
FILM(**namn,** regissör, inspelningsår)
FÖRESTÄLLNING(**bio, film, datum, klockslag**)

Primärnycklar är angivna i fetstil.
Följande främmande nyckel förhållanden råder:

FÖRESTÄLLNING.bio << BIO.namn (dvs kolumnen ’bio’ i tabellen FÖRESTÄLLNING utgör främmande nyckel tabellen BIO).
FÖRESTÄLLNING.film << FILM.namn (dvs kolumnen ’film’ i tabellen FÖRESTÄLLNING utgör främmande nyckel mot tabellen FILM).

a) Formulera följande fråga i SQL: Vilka datum har filmerna som har ’Akira Kurosawa’ som regissör’ visats?

```
SELECT Datum
FROM FILM, FÖRESTÄLLNING
WHERE film=namn
AND regissör = ’Akira Kurosawa’
```

b) Formulera följande fråga i SQL: Vilken är den mest visade filmen?

```
CREATE VIEW antalvisningar AS
        (SELECT film as Film, count(*) as Antal
        FROM FÖRESTÄLLNING
        GROUP by film)

SELECT Film, Antal
FROM antalvisningar
WHERE Antal = (SELECT Max(Antal) FROM antalvisningar)
```

c) Formulera följande fråga i relationsalgebra: Vilka datum har filmerna som har ’Akira Kurosawa’ som regissör’ visats?

π_{Datum}(FÖRESTÄLLNING $|X|_{namn=film}$ ($\sigma_{regissör = ‘Akira\ Kurosawa’}$ FILM))

d) Formulera följande fråga i relationsalgebra: Vilken film har aldrig visats på någon biograf?

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen EIT:DB 11 januari 2013

π_{namn}(FILM) MINUS π_{film}(FÖRESTÄLLNING)

Kommentar: Här kan man först döpa om kolumen 'film' i tabellen FÖRESTÄLLNING till 'namn', dock ej nödvändigt i relationsalgebra (men väl i SQL om vi skulle vilja utföra motsvarande operation där).

Relationsalgebraiska operatorer:

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

Division ÷, Snitt ∩ , Union ∪ , Differens -, Aggregering/gruppering G $_{f}$ Tilldelning ←, Namnändring ρ

SQL-syntax:

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