Tentamen

DB:SP/DVK/ATD Databastmetodik + Lösningsförslag

23/3 2013 kl. 10 – 14

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 läsligt, annars kan tentamen inte rättas

Lycka till!

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen DB:SP/DVK/ATD 23 mars 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 vyer?
a) En vy kan användas för att låta olika användar-grupper få tillgång till olika typer av information i databasen.
b) En vy kan aldrig uppdateras.
c) En vy kan inte användas för att förenkla SQL mot databasen eftersom vyn inte sparas när man lämnar databasen.

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 en alternativnyckel?
a) En alternativnyckel får inte innehålla NULL.
b) En alternativnyckel kan innehålla dubletter.
c) En alternativnyckel är en fd. kandidatnyckel som inte blivit vald till primärnyckel.

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 har en främmande nyckel mot HÄST (beskrivs nedan). Denna nyckel är deklarerad som ON DELETE CASCADE. Vad betyder det?

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen DB:SP/DVK/ATD 23 mars 2013

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	120101
Lily	4	121231
John	3	121231
Lily	1	121231

RIDNING.HästId << HÄST.HästId

a) Om jag tar bort en rad i tabellen HÄST så tas de rader i tabellen RIDNING som har en främmande nyckel mot den borttagna hästen också bort.
b) Om jag tar bort en rad i tabellen RIDNING så tas den rad i tabellen HÄST som innehåller motsvarande primärnyckel också bort.
c) Man kan inte ta bort rader i tabellen HÄST så länge det finns rader i tabellen RIDNING som har främmande nycklar mot de rader i HÄST som man vill ta bort.

I del 7 och 8 ska man besvara en fråga eller skriva en kortare förklaring som beskriver vad ett eller flera begrepp står för.

Rätt rad: baacaa

7) Varför behövs lås när man utför en transaktion mot en databas? Ge exempel på någon typ av princip för att tilldela lås till en transaktion och vad den innebär.

Här kan man fokusera på olika specifika problem eller bara skriva generellt om konkurrerande transaktioner, dvs om (minst) två olika transaktioner vill ha tillgång till (läsa eller skriva) resurser (hela tabeller, vissa rader, kolumner etc) i databasen uppstår konflikter om data ändras av någon av transaktionerna. Med konflikter menas här att databasen inte är konsistent med avseende på att båda transaktionerna hela tiden ser samma data och ändrar i samma data. För att omöjliggöra att transaktioner ser fel data eller att uppdateringar utförs fel eller förloras låser man olika typer av resurser som transaktionerna efterfrågar så att alla ändringar genomförs som skedde de i sekvens och inte parallellt. Alla lås-scheman går ut på att låsa så lite som möjligt med bibehållet krav på att data hela tiden är konsistent map konkurrerande användare. En typ av lås-schema är sk. Two-Phase-Locking där en trans antal begärda och erhållna lås först växer tills alla resurser som behövs för transen uppnåtts, därefter släpps låsen varefter de inte behövs.

8) Beskriv begreppen ROLLBACK och COMMIT och hur de relaterar till en atomär databas-transaktion.

Transaktionsbegreppet inom databashanteringssystem bygger på att transaktioner är atomära det vill säga att en transaktion antingen sker i sin helhet eller inte alls. Som vi såg ovan kunde t ex en

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

DELETE av en rad i en tabell leda till borttag av andra rader i andra tabeller. Det är ytterst viktigt att detta sker i sin helhet eller inte alls. Det vore t ex ytterst olyckligt om en rad i HÄST togs bort utan att motsvarande rader i t ex tabellen RIDNING också togs bort. Det skulle nämligen leda till brott mot referential integrity, dvs man riskerar att ha hästar i tabellen RIDNING som inte har någon motsvarighet i tabellen HÄST. Genom att definiera vad som utgör en transaktion (i detta fall DELETE av HÄST följt av DELETE av eventuella motsvarande rader i RIDNING, i just detta fall görs detta automatiskt eftersom vi deklarerat en främmande nyckel-regel som ON DELETE CASCADE) skapar man en enhet som antingen utförs i sin helhet eller inte alls. Dvs, först när båda dessa saker är gjorda görs en så kallad COMMIT, dvs signalering om att hela transaktionen är genomförd och att den ändrade informationen kan skrivas till disk. Först när detta är klart anses transaktionen genomförd. Vad händer om något fel inträffar mitt i transaktionen, tex. ett strömavbrott när en rad i tabellen HÄST tagits bort innan motsvarande förändringar inträffat i tabellen RIDNING? Här tillämpas begreppet ROLLBACK, dvs en påbörjad transaktion som endast delvis utförts (ingen commit har signalerat att transen är klar och skriven mot disk) 'rullas tillbaka', alla eventuella förändringar vad gäller tabellen HÄST (i detta fall) återförs till sitt tillstånd innan transaktionen påbörjades.

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

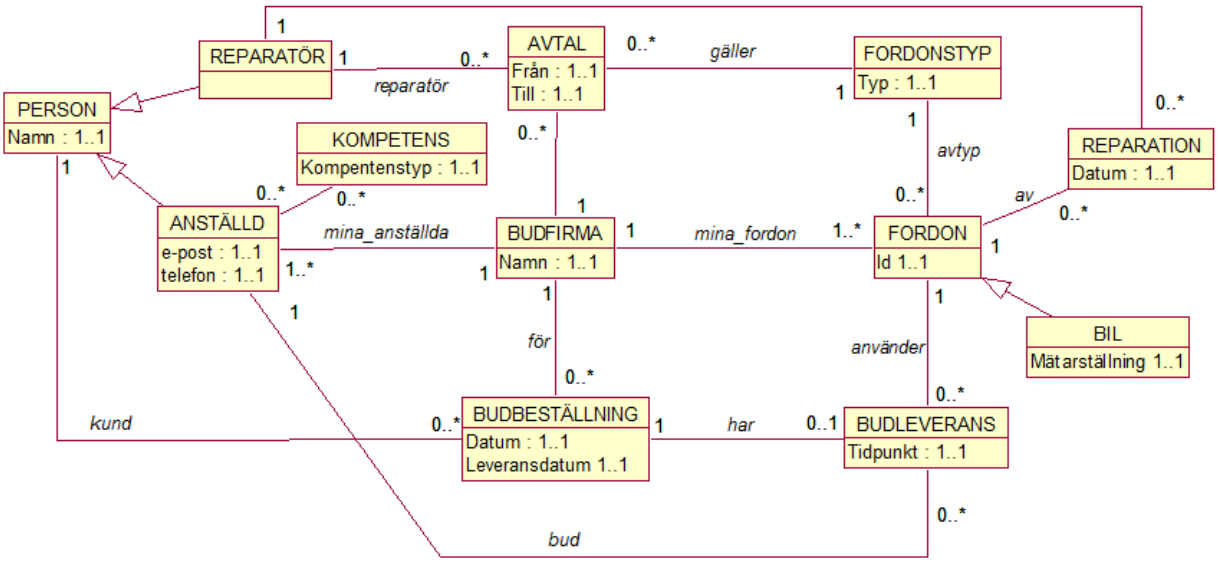
Konstruera ett konceptuellt schema i form av ett UML-klasschema som ger möjlighet att representera samtliga utsagor nedan. Ange avbildningsregler (min-max-värden) för samtliga attribut och för samtliga roller i associationer:

- Budfirmor har fordon att använda för sändning av bud av olika slag.
- Fordon är av typen lastbil, personbil, båt, motorcykel och cykel.
- För att sköta service av fordonen har budfirmorna avtal med reparatörer.
- Reparatörer kan laga olika typer av fordon och fordonstyper kan lagas av olika reparatörer.
- Budfirmorna har många anställda som budar produkter till kunder på olika sätt. Alla anställda förutsätts kunna cykla men för övriga typer av fordon krävs det specialkompetens för att använda fordonet, t ex förarbevis (båt) eller körkort.
- När fordon repareras vill man veta datum när reparationen skedde, vilket reparatör som utförde reparationen och, för bilar, vilken mätarställning bilen hade när det kom in för reparation.
- För anställda på budfirmorna vill man veta deras namn, e-postadress, telefon och eventuella specialkompetenser en viss anställd har.
- Budfirmorna håller reda på beställningar av bud och information som är av intresse är datum för beställningen, datum när leveransen ska ske och vem som är kund.
- När beställningen ska utföras kompletteras den med uppgifter om vilken anställd som budar och vilket fordon som används av budet.
- Varje beställning förutsätts utföras via en leverans, med andra ord: ett bud kan inte utföra flera beställningar i en och samma körning.

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen DB:SP/DVK/ATD 23 mars 2013

Lösningsförslag:



Kommentarer: I denna uppgift talas det om olika budfirmor så klassen budfirma behövs – informationssystemet ska inte bara stödja en budfirma (i vilket fall denna klass kunde uteslutas). Anställda har specialkompetenser, eventuellt flera och eventuellt ingen (utöver förmåga att cykla som inte betraktas som en specialkompetens eftersom alla anställda har den) – om vi inte ska få ett partiellt attribut behöver vi klassen KOMPETENS (eller specialkompetens) relaterad till klass ANSTÄLLD. Anställda och reparatörer kan eventuellt utgöra subklasser till en klass PERSON så slipper man upprepa namnet på dem. Budfirmor har avtal med reparatörer om att reparera vissa fordonstyper – man behöver förmodligen både klassen fordon och fordonstyp, den sista för att åskådliggöra att avtalet med reparatörer gäller fordonstyper (inte enskilda fordon) och den förra för att kunna modellera att olika typer av budningar görs med enskilda fordon. Man modellerar med fördel både klassen budbeställning och budleverans för att kunna skilja på planerade och utförda budningar. Alternativet med en enda klass för både beställning och leverans är möjlig men mindre utbyggbar, man vet t ex inte vilket fordon som ska användas förrän leveransen sker. Jag modellerade ingen klass KUND utan använde PERSON-klassen med en association som heter kund istället. Man kan dock utmärkt väl ha med KUND om man vill. Mätarställning var viktigt att hålla reda på när reparationer utfördes på bilar men eftersom bilar alltid har en mätarställning modellerades det hela som att BIL är en sub-klass till fordon och BIL har ett attribut mätarställning. Det innebär att man alltid håller reda på mätarställningen på BIL:ar. Ett alternativ är att istället sub-klassa REPARATION med en klass BIL-reparation och lägga attributet mätarställning där.
Viktigt att man modellerar relationer mellan klasser via associationer. Man får _inte_ modellera sådana relationer genom att ange attribut i klasserna (liknande främmande nycklar).

Uppgift 3 (svarar mot lärandemål 4 avseende analytisk databasdesign)

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen DB:SP/DVK/ATD 23 mars 2013

Betrakta följande relationsschema:

PERSON(Personnummer, Namn, Adress, Telefonnr, e-postadress, rumsnummer, rumsstorlek)

Följande funktionella beroenden gäller:

Personnummer → Namn, Adress, Telefonnr, e-postadress, rumsnummer
Rumsnummer → rumsstorlek

a) Bestäm primärnyckel för tabellen PERSON

b) PERSON förutsätts vara i 1NF. Normalisera PERSON till 3NF. Motivera dina svar.

Lösningsförslag:

a) Primärnyckel för PERSON blir Personnummer. Detta eftersom Personnummer bestämmer övriga attribut funktionellt (rumsstorlek är också bestämd av Personnummer efersom Personnummer bestämmer Rumsnummer funktionellt och Rumsnummer bestämmer i sin tur rumsstorlek, använder vi transitiva lagen får vi att Personnummer bestämmer rumsstorlek.)

b) I uppgiftstexten anges att tabellen är i 1NF. Vi ska nu överföra tabellen till högre normalform. (det är dock inte givet att den inte redan är i högre normalform, uppgiften om 1NF är given pga att vi inga rader eller domän-beskrivningar har att inspektera, vi kan alltså inte av den tabelldefinition vi har här avgöra om tabellen är i 1NF, detta måste anges i uppgiftstexten). Vi har in icke sammansatt primärnnyckel – det betyder att vi inte kan ha några partiella funktionella beroenden med avseende på primärnyckeln – således är vi i 2NF. Vi har dock ett transitivt beroende mellan primärnyckeln och attributet rumsstorlek (se diskussion i a)). Vi är alltså inte i 3NF. Vi bryter därför ut Rumssnummer och rumsstorlek (alltså de två kolumner som gav upphov till det transitiva beroendet) till en egen tabell:

RUM (**Rumsnummer,** rumsstorlek) där allts rumsnummer är primärnyckel, kvar i den gamla tabellen blir:
PERSON(**Personnummer,** Namn, Adress, Telefonnr, e-postadress, rumsnummer) där PERSON.rumsnummer << RUM.rumsnummer

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

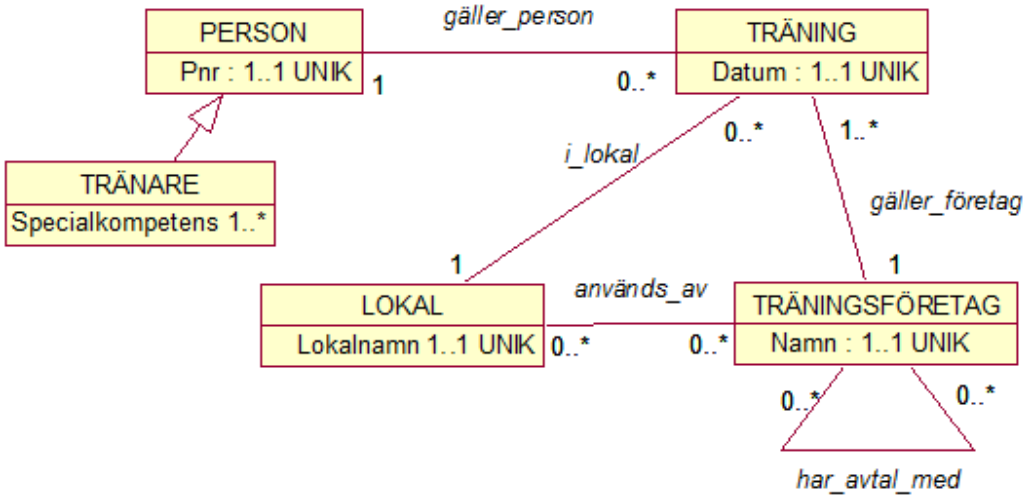
Tentamen DB:SP/DVK/ATD 23 mars 2013

Uppgift 4 (svarar mot lärandemål 3 avseende syntetisk databasdesign)

Översätt det konceptuella schemat nedan till ett relationsdatabasschema. Vad som är identifierare för alla klasser framgår av schemat. 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)



Lösningsförslag:

PERSON(Pnr)
TRÄNARE (Pnr), där TRÄNARE.Pnr är FN mot PERSON.Pnr
SPECIALKOMPETENS(Pnr, Specialkompetens) där SPECIALKOMPETENS.Pnr är FN MOT TRÄNARE.Pnr
LOKAL(Lokalnamn)
TRÄNINGSFÖRETAG(Namn)
ANVÄNDSNING(Lnamn, TFnamn) där ANVÄNDNING.Lnamn är FN mot LOKAL.Lokalnamn och ANVÄNDNING.TFnamn är FN mot TRÄNINGSFÖRETAG.Namn
AVTAL(Namn1, Namn2), där AVTAL.Namn1 är FN mot TRÄNINGSFÖRETAG:Namn och AVTAL.Namn2 är FN mot TRÄNINGSFÖRETAG.Namn
TRÄNING(Datum, Pnr, Företag, Lokal), där TRÄNING.Pnr är FN mot PERSON.Pnr och TRÄNING.Företag är FN mot TRÄNINGSFÖRETAG.Namn och TRÄNING.Lokal är FN mot LOKAL.Lokalnamn

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen DB:SP/DVK/ATD 23 mars 2013

Kommentar: Den sista tabellens primärnyckel är given i texten på ett mkt dumt sätt, tabellen TRÄNING borde naturligtvis identifieras av både Datum, Personen som tränar och träningsföretaget. Men nu var det fel angivet i tenta-texten och då blir översättningen som jag gjort ovan. Stor hänsyn tas till de som rent reflexmässigt gjort den andra översättningen, den är i och sig fel givet uppgiftstexten men mycket rimligare för domänen.

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen DB:SP/DVK/ATD 23 mars 2013

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

Betrakta följande relationer:

ANSTÄLLD

Anstnr	Namn	Telefon	E-post
1	Hober Mallow	12345	hober@term.gov
2	Hari Seldon	NULL	seldon@sf.trant
3	Bayta Darrell	23456	baytad@ff.term
4	Ebling Mis	NULL	eblingm@ff.term
5	Mulan	11111	firstcit@calg.gov

PROJEKTDELTAGANDE

Person	Projekt	Från	Till
1	1	130421	130521
2	1	130401	140101
2	2	130101	140101
2	3	120530	121231
2	4	130321	131231
3	3	120530	121231
4	3	120530	121001
5	3	120530	121101

PROJEKT

Projekt	Start	Slut	Ledare
1	130421	140101	2
2	130101	140101	2
3	120530	121231	5
4	130321	131231	2

PROJEKTDELTAGANDE har två främmande nycklar:
En mot PROJEKT där PROJEKTDELTAGANDE.Projekt är FN mot PROJEKT.Projekt
En mot ANSTÄLLD där PROJEKTDELTAGANDE.Person är FN mot ANSTÄLLD.Anstnr

PROJEKT har en främmande nyckel mot ANSTÄLLD dvs PROJEKT.Ledare är FN mot ANSTÄLLD.Anstnr

a) Skriv SQL för att ta fram namn och e-postadresser för alla som arbetar i ett projekt som ’Hari Seldon’ leder.

```
SELECT A1.Namn, A1.E-post
FROM ANSTÄLLD A1, ANSTÄLLD A2, PROJEKTDELTAGANDE PD, PROJEKT P
```

```
WHERE A2.Namn = 'Hari Seldon'
AND A2.Anstnr = P.Ledare
AND P.Projekt = PD.Projekt
AND PD.Person = A1.Anstnr
AND A1.Namn <> 'Hari Seldon'

/* sista raden tolkningsbar, vill man ha med att Hari arbetar i det projekt Hari leder tar man
bort raden, om inte behåller man den */
```

b) Skriv SQL för att ta fram den som arbetat (hans eller hennes namn) i flest projekt.

En hjälpsam vy först:

```
CREATE VIEW jobbat AS
        (SELECT Person, COUNT(*) AS Antaljobb
        FROM PROJEKTDELTAGANDE
        GROUP BY Person)

SELECT Namn, Antaljobb
FROM ANSTÄLLD, jobbat
WHERE Anstnr=Person
AND Antaljobb = (SELECT MAX(Antaljobb) FROM jobbat)
```

c) Vad blir resultatet, rita ut raderna, om följande SQL-sats exekveras:

```
SELECT Namn
FROM ANSTÄLLD, PROJEKTDELTAGANDE
WHERE Anstnr = Person
AND Från > 121231
```

Namn
Hober Mallow
Hari Seldon
Hari Seldon
Hari Seldon

d) Vad blir resultatet, rita ut raderna, om följande relationsalgebraiska uttryck exekveras:

$\pi_{Namn}(\text{ANSTÄLLD} \; |X|_{Anstnr=Person} (\sigma_{Från>121231} \; \text{PROJEKTDELTAGANDE}))$

Namn
Hober Mallow
Hari Seldon

e) Skriv relationsalgebra för att ta fram anställda (deras namn) som är projektledare.

Inst. för Data- och Systemvetenskap
Stockholms Universitet
Maria Bergholtz

Tentamen DB:SP/DVK/ATD 23 mars 2013

Projektledare ← π_{Ledare} (PROJEKT)

π_{Namn} (ANSTÄLLD $|X|_{Anstnr=Ledare}$ Projektledare)

Eller bara en rad med join mellan ANSTÄLLD och PROJEKT

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>]
```