Örebro universitet
Institutionen för naturvetenskap och teknik
Thomas Padron-McCarthy (thomas.padron-mccarthy@oru.se)

Tentamen i

Databasteknik

för D1 m fl

tisdag 14 mars 2017

Gäller som tentamen för:
DT105G Databasteknik, provkod 0100

Hjälpmedel:	Ordbok för översättning.
Poängkrav:	Maximal poäng är 40. För godkänt betyg krävs 24 poäng, varav minst fyra poäng på uppgift 1 och minst fem poäng på uppgift 2.
Resultat:	Meddelas på kursens hemsida eller via e-post senast tisdag 4 april 2017.
Återlämning av tentor:	Elektroniskt via Studentforum.
Examinator och jourhavande:	Thomas Padron-McCarthy, telefon 070 - 73 47 013.

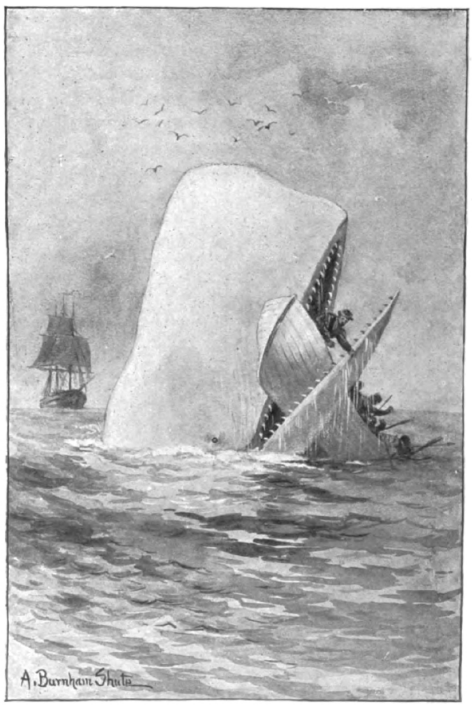
- Skriv tydligt och klart. Lösningar som inte går att läsa kan naturligtvis inte ge några poäng. Oklara formuleringar kommer att misstolkas.
- Skriv den personliga tentamenskoden på varje inlämnat blad. Skriv *inte* namn eller personnummer på bladen.
- Skriv bara på en sida av papperet. Använd inte röd skrift.
- Antaganden utöver de som står i uppgifterna måste anges. Gjorda antaganden får inte förändra den givna uppgiften.
- Skriv gärna förklaringar om hur du tänkt. Även ett svar som är fel kan ge poäng, om det finns med en förklaring som visar att huvudtankarna var rätt.

LYCKA TILL!

Scenario till uppgifterna

Vi behöver en databas med båtar och fiskar. Det vi ska lagra i databasen är följande:

- **Båtar.** Varje båt har ett unikt **nummer** och ett (inte nödvändigtvis unikt) **namn**. Båten har också ett uppskattat **värde** i kronor.
- **Personer** som **äger** båtarna. Varje båt ägs av en eller flera personer. En person kan också äga flera båtar. Vi antar att om det är flera personer som äger en båt tillsammans, så har de lika stor andel i båten, så vi behöver inte lagra hur stor andel varje person har. Varje person har ett unikt **nummer**, ett (inte nödvändigtvis unikt) **namn**, och bor på en (inte nödvändigtvis unik) **adress**.
- **Fiskarter.** Varje fiskart har ett unikt **namn**, till exempel **makrill**.
- **Fiskar.** Vi ska också lagra de enskilda fiskarna. Varje fisk tillhör en viss art. Varje fisk har också ett unikt **nummer**. (Vi hoppas kunna märka alla fiskarna med streckkoder.)
- Båtarna **fångar** fiskar. Varje fisk är antingen ofångad och simmar forfarande omkring, eller så har den fångats av en båt. När fisken väl fångats av en båt blir den uppäten, och kan inte fångas fler gånger, men en och samma båt kan förstås fånga många fiskar.
- Ibland lyckas någon ovanligt stor och arg fisk **sänka** en båt. Om en båt sänkts av en fisk har den sjunkit, och kan inte sänkas fler gånger, men en och samma fisk kan (om den är riktigt stor och arg) sänka flera båtar.



"Both jaws, like enormous shears, bit the craft completely in twain."

—Page 510.

Bilden ovan föreställer föreställer en båt och en val. Valar är egentligen inte fiskar, men bilden får duga som illustration.

Uppgift 1 (5 p)

Rita ett ER- eller EER-diagram för den beskrivna databasen. Använd informationen i scenariot ovan, men tänk också på att det ska gå att svara på frågorna i uppgift 3 nedan.

ER- och EER-diagram kan ritas på flera olika sätt. Om du använder en annan notation än kursboken, måste du förklara den notation som du använder.

Uppgift 2 (6 p)

Implementera den beskrivna databasen i relationsmodellen, dvs översätt ER-diagrammet till tabeller.

Du behöver inte skriva **create table**-kommandon i SQL, men du ska ange vilka relationer som finns och vilka attribut varje relation innehåller. Ange också alla kandidatnycklar, vilken av dessa som är primärnyckel, samt vilka referensattribut som finns och vad de refererar till.

Implementationen ska vara bra.

Uppgift 3 (11 p)

Formulera följande frågor i SQL. Definiera gärna vyer om det underlättar, men skapa inte nya tabeller.

a) (1p) Det finns flera olika båtar som heter **Rudolfina**. Vilka nummer har de?

b) (2p) Här är båt nummer 4711. Vad heter ägarna?

c) (2p) Vad heter de fiskarter som har fångats av båt nummer 4711?

d) (3p) Man kan tänka sig att en båt blir sänkt av en fisk som fångats av den båten. (Fisken kanske var så stor och tung att båten sjönk.) Vad heter de båtar som råkat ut för detta?

e) (3p) Vilken art tillhör den fisk som sänkt flest båtar?

Uppgift 4 (3 p)

Databasen innehåller realistiska mängder data, vilket innebär tusentals fiskarter, miljoner båtar och miljarder fiskar. De tre SQL-frågorna a, b och c i uppgift 3 körs väldigt ofta (men kanske med andra konstanter, till exempel att man söker efter båtar med andra namn än just **Rudolfina**). De tar för lång tid att köra, och behöver snabbas upp. Vi märker att det inte finns några index alls i databasen, inte ens på nycklar.

Vilka index bör man skapa för att just SQL-frågorna a-c i uppgift 3 ska gå snabbt att köra?

Uppgift 5 (3 p)

Vi använder en databashanterare som har så kallad "auto-commit", vilket betyder att varje SQL-kommando räknas som en egen transaktion, så länge man inte uttryckligen startar en transaktion med kommandot "start transaction".

Vi startar två olika klientprogram som loggar in på samma databas, och ger följande SQL-kommandon, i den angivna ordningen, i de två klienterna.

Vad blir resultatet av var och en av de nio select-frågorna?

Klient 1	Klient 2
create table Fiskar (nummer integer not null primary key, vikt integer);	
insert into Fiskar values (1, 100);	
select * from Fiskar; -- *Fråga 1*	
	select * from Fiskar; -- *Fråga 2*
	start transaction;
select * from Fiskar; -- *Fråga 3*	
	insert into Fiskar values (2, 200);
	select * from Fiskar; -- *Fråga 4*
	commit;
	select * from Fiskar; -- *Fråga 5*
select * from Fiskar; -- *Fråga 6*	
start transaction; insert into Fiskar values (3, 300);	
select * from Fiskar; -- *Fråga 7*	
rollback;	
select * from Fiskar; -- *Fråga 8*	
	select * from Fiskar; -- *Fråga 9*

Uppgift 6 (3 p)

Här är tabellen **Fiskar** från uppgiften ovan, med några exempeldata:

Fiskar	
<u>Nummer</u>	**Vikt**
1	100
2	200
4	100

a) Vi vet redan att **Nummer** är primärnyckel i tabellen. Men finns det fler kandidatnycklar? I så fall vilka?

b) Vilka fullständiga funktionella beroenden finns i tabellen?

c) Vilken är den högsta normalform, av 1NF, 2NF, 3NF och BCNF, som tabellen uppfyller? Motivera svaret!

Uppgift 7 (9 p)

Ange för varje påstående om det är sant eller falskt! Lämna in denna sida tillsammans med resten av svaren. Fel svar ger inte minuspoäng.

Påstående	Sant	Falskt
De flesta personer som arbetar med data som finns i en databas söker i databasen genom att skriva SQL-frågor.		
Den så kallade tre-schema-arkitekturen innebär att samma databas kan betraktas på tre olika nivåer: SQL-nivån, transaktionsnivån och klientnivån.		
En svag entitetstyp är en entitetstyp där instanserna inte lagras i databasen, utan beräknas utifrån andra data som finns i databasen.		
"Relationerna" i en relationsdatabas är kopplingarna mellan tabeller med främmande nycklar ("foreign keys").		
Det finns en standard för frågespråket SQL, men de flesta vanliga databashanterarna följer inte SQL-standarden särskilt exakt, utan de har sina egna dialekter.		
Integritetsvillkor i databaser handlar om vilka data man får lagra om personer, för att inte kränka den personliga integriteten.		
En så kallad lagrad procedur är en programsnutt som man kan lagra i databasen, men den kan inte anropas och köras i databasen, utan måste först hämtas av klientprogrammet. Därefter kan den köras som en del av det programmet.		
Många webbplatser lagrar sina data i en databas. Det brukar fungera så att webbläsaren (till exempel Internet Explorer eller Safari) först hämtar den grundläggande HTML-koden för webbsidan, och sen kopplar webbläsaren upp sig mot databashanteraren och kör SQL-frågor. Svaren från SQL-frågorna visas sen av webbläsaren.		
Tre vanliga databashanterare är Microsoft MySQL Server, Microsoft Access Server och DbVisualizer.		