Resultat Databastentan 2019

Jämfört med 2018 var det några färre som fick U (7% jämfört med 10%) och betydligt färre som fick VG (18% jämfört med 49%). Totalt var det 93% som klarade tentan (75% G respektive 18% VG).

2019-03-26 (minst 24/30 på G-delen krävdes för G, G + 12/30 på VG-delen för VG)

Betyg	Antal av 84 st.	%
U	**6**	**7.14%**
G	**63**	**75.00%**
VG	**15**	**17.86%**
Check	***84***	***100.00%***

2018-03-22 (50% krävdes för G, 75% för VG)

Betyg	Antal av 100 st.	%
U	**10**	**10.00%**
G	**41**	**41.00%**
VG	**49**	**49.00%**

Var tentan lättare än förra året?

Formellt sett så var det svårare att klara tentan 2019 eftersom vi delade in den i en G-del med frågor på grundläggande nivå där vi krävde 80% för att få minst G på tentan och en VG-del med lite svårare frågor där vi krävde 60% plus kravet för G för betyget VG.

I 2018 års tenta så fanns två frågor med om JDBC. Dessa frågor var det flest som hade problem med det året. Det årets fråga nummer 9, “JDBC basics” var det hela 20% som hade noll (0) poäng på (genom att inte svara alls eller svara helt fel), vilket naturligtvis drog ned poängen. Fråga 10, “JDBC other” från samma år var det ännu sämre ställt med. Där var det hela 63% som hade noll (0) poäng.

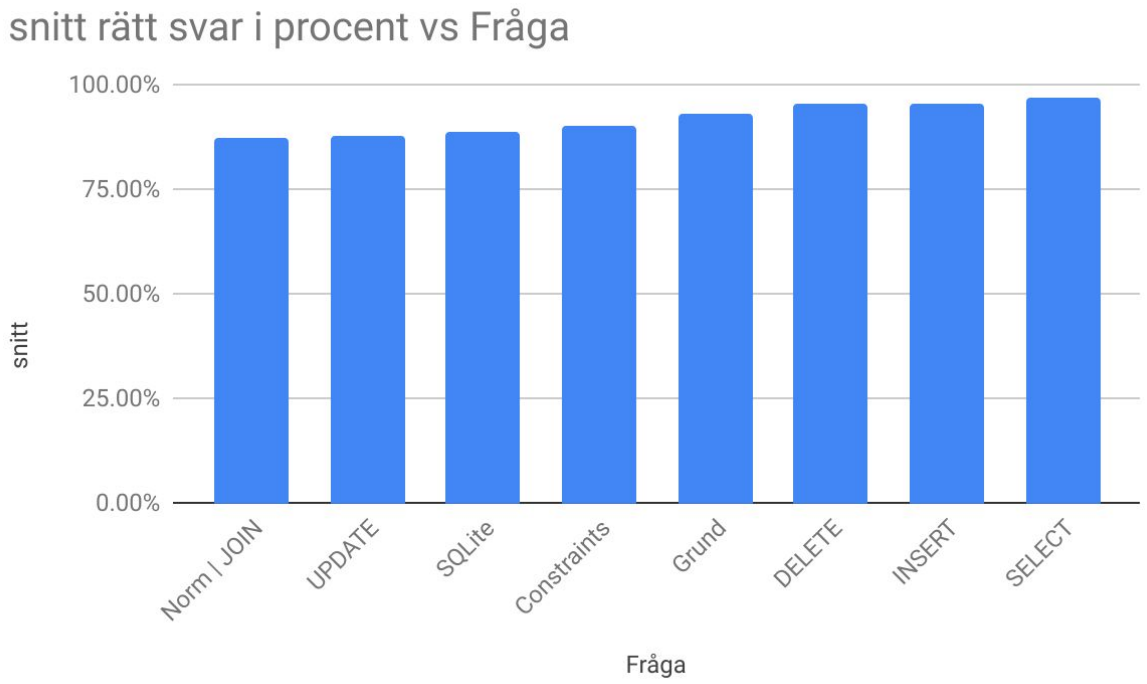
I 2019 års tentamen så strök vi alla frågor om JDBC vilket borde göra tentan lättare om vi förutsätter att JDBC fortfarande är det studenterna tycker är svårast.

I 2018 års tentamen så var den fråga som var svårast efter JDBC-frågorna, fråga 8 om Constraints. De skrivande hade det året drygt 54% rätt på den frågan i genomsnitt (2.71

poäng av 5 möjliga). Därefter var fråga 7 om JOIN och normalisering då den fjärde svåraste frågan med i snitt 65% rätt (3,37 poäng av 5 möjliga).
Eftersom vi strök frågorna om JDBC i år, så borde således (när vi tittar på G-delen) de två svåraste frågorna 2019 varit frågorna om Constraints (svårast) och JOIN (näst svårast) ifall tentan följt förra årets tenta innehålls- och svårighetsmässigt i övrigt.

Dock var 2019 fråga 7 om JOIN svårast (lägst poäng i snitt) med 87,5% rätt (3,5 av 4) tätt följt av fråga 2 om SQLite med 88,69% (2,7 av 3). Fråga 8 om Constraints gick bra 2019 med ett snitt på över 90%.

Nedan är en tabell över frågorna sorterat på svårighetsgrad utifrån hur många procent av maxpoäng studenterna hade 2019 (vad gäller G-delen).



De fyra “svåraste” frågorna bland U-studenterna var:

Norm | Join	43.75%
SQLite	44.44%
constraints	66.67%
INSERT	81.25%

I likhet med godkända studenter var resultatet svagast vad gäller normalisering och JOIN. Men här var antalet poäng ungefär hälften jämfört med de godkända. Ungefär lika dåligt gick det för U-studenterna på frågan om SQLite. Constraints-frågan gick lite bättre för U-studenterna men fortfarande låga 67%. Insert var den fjärde svåraste frågan för de underkända men med sina 81% så är det ändå lägre procent rätt än vad G-studenterna fick på sin svåraste fråga (JOIN, 87,5%).

Tittar man på G-delen respektive VG-delen per betygsgrupp (vi räknar bara snitt på VG-delen om man svarat på någon av frågorna) ser vi följande för 2019 års tenta:

Snitt bland G	27.88888889	5.59375
Snitt bland U	21.75	4
Snitt bland VG	28.76666667	14.16666667

Det är ganska liten skillnad i poäng på G-delen mellan de som fick G och VG. De underkända låg ganska nära G-gränsen men ändå betydligt under de som fick G respektive VG. Tittar vi på VG-delen (och där räknar bara de som fått åtminstone ett poäng, dvs svarat på någon av frågorna - vi såg ingen som svarat men ändå fått noll poäng) ser vi att det är ganska liten skillnad mellan G-studenterna (5,6 poäng av 20) och U-studenterna (4 poäng av 20). De som fick VG behövde minst 12 poäng (samt kraven för att få G) och snittet blev 14 poäng av 20, vilket får anses som bra.

Tittar vi på 2018 års tenta och bortser från fråga 9 och 10 om JDBC, så var det 60% som hade minst 80% av de 40 poäng som fråga 1-8 hade maximalt. Det vill säga, utan de två JDBC-frågorna så hade endast 60% blivit godkända med 80%-regeln förra åren. Det är dock svårt att jämföra dessa tentor rättvist, eftersom 2019 års tenta var uppdelad i en G-del och en VG-del. Man kan misstänka att många inte ens försökte på VG-delen (23% hade 0 poäng). Så en rättvis jämförelse skulle bara inkludera de delfrågor i 2018 års tenta som hade motsvarande nivå som G-delen på 2019 års tenta.

Det är också svårt att veta om årskullen varit bättre förberedda än förra årets studenter. Kanske studerade man hårdare i år med vetskapen om att det kommer krävas 80% på G-delen för att bli godkänd. I materialet för 2019 års undervisning fanns ett par nya moment för att kompensera för vissa brister i 2018 års resultat; NULL och WHERE gicks igenom i två nya föreläsningsfilmer. Samtliga kapitel i wikin försågs även med kodexempel tidigt i teorisidorna. Undervisningen ändrades också. till mer strikt Flipped-classroom - vi Live-kodade på föreläsningarna på databasdelen i stort sett uteslutande och lät studenterna se föreläsningsfilmerna i förväg (i stället för att genomföra traditionell föreläsning med samma innehåll som i filmerna).

Sammanfattningsvis tycker inte lärarna att det var en lättare tenta i år, trots att andelen underkända sjönk från 10% till 7,14%. Andelen VG sjönk också i år från 49% 2018 till 18% 2019. Eftersom hela 40% hade blivit underkända med 80%-krav på frågorna 1-8 (allt utom JDBC) från 2018 års tenta, så menar vi att modellen med 80% på G-delen och 60% på VG-delen är rättvisare i det att det krävs mer för att få G.

I kursvärderingen (läst när 38 personer svarat) fick svårighetsgraden på examination följande betyg på skalan Svårighetsgrad 1 - 3 där 1 är “för låg” och 3 är “för hög”:

Alternativ	**För låg**	**Lagom**	**För hög**
Fördelning	5,3%	71,1%	23,7%
Antal	2	27	9

Då endast sex (6) personer fick U på tentan kan detta betyda att tentan upplevdes för svår men att studenterna kunde mer än de tror. Kanske är det rentav så att upplägget med krav på 80% på G-delen förmådde studenterna att studera hårdare än de skulle gjort annars, vad vet vi. Lärarnas åsikt är att tentan inte var för lätt men det får någon extern bedömare avgöra huruvida det är rätt. Vi kan ju här passa på att lista de lärandemål som rörde databasdelen:

- *beskriva grundläggande konstruktioner i SQL samt grundläggande kunna beskriva en relationsdatabas;*
- *självständigt använda grundläggande konstruktioner i språket SQL för att söka ut, ändra, lägga till och ta bort data från en relationsdatabas bestående av flera kopplade tabeller;*
- *resonera kring teknikerna för normalisering och dataintegritet från databasteorin utifrån när det är lämpligt att applicera dem, vilka effekter man vill uppnå samt konsekvensen av att inte applicera dem i olika grad.*

I fritextdelen finner vi följande kommentarer om tentan:

> “*Tentans upplägg var mindre bra. Sjukt svårt att bli godkänd, endast 6 fel på g-delen som var ca 80% av tentan (om inte mer?) och där VG frågorna inte räknades med för att bli godkänd. Lärarna förklarade dock gång på gång hur upplägget skulle vara på tentan och varför så förstår helt konceptet och var föreberedd på detta men det tar inte bort det faktum att tentan blev svårare för att bli godkänd och många struntade i VG för att bara sikta på G då nästan hela tentan skulle bestå av g-frågor och man inte fick ha nästan några fel och VG frågorna ej räknas med för att bli godkänd.*”

Lärarna håller med i princip men är övertygade om att en välförberedd student hade gott om tid att klara av även VG-frågorna. Det var 19 studenter som inte svarade på VG-delen (22%). Andra kommentarer:

> “*svår tenta med för hög G-gräns för vad som krävdes*”
>
> “*att behöva 80% på tentan för ett G var lite att ta i*”

En längre kommentar:

> “*Examination - vad var syfte med en examen som i G delen innehållde ganska många frågor som var direkt tagna från 2018 tentan? Vad examineras här, att man kan kolla på gamla tentor och lära sig exakta svar?*
>
> *VG delen - frågor 9 och 10 var exakt samma som exempler som vi hade tagit upp vid några övningar (nästan samma vid fråga 12, men där fanns lite variation). Skulle ha varit bättre om ni hade ändrat på de lite grann så att man är tvungen att fundera lite mer. Annars, tentan var jätte enkelt, känns inte som att man förtjänar en G eller VG alls.*

> *Ni borde fundera lite att så enkla tentor sänker ner utbildnings kvalitet, eftersom människor som studerar inte alls kan klara dem utan problem. Detta leder till att i nästa kurser i utbildning i stället för att lära sig nya saker, spenderar man tid på att upprepa igen och igen någonting från gamla kurser.*"

För det första så avråder vi alltid aktivt och vid upprepade tillfällen från att plugga genom att fokusera på gamla tentor. För det andra, så var några frågor väldigt lika tidigare års tenta men vi ändrade på detaljer eller ordning på delfrågor. Detta visade sig fungera delvis då många svarade fel på dessa frågor i samma utsträckning som på helt nya frågor. För det tredje så är det svårt att komma på helt unika frågor på grundnivå i till exempel SQL. Det blir alltid likheter.

En fråga som var ganska lik frågan från tidigare år handlade om UPDATE. Där skulle man ändra till nya värden (jämfört med förra året) och utifrån nya kriterier (jämfört med förra året). Däremot gjorde vi ena delfrågan lite svårare genom att kräva att två kolumner skulle ändras (förra året var det hela tiden en kolumn som skulle ändras). Trots detta var det högre snittpoäng på frågan som helhet jämfört med förra året. En annan skillnad mellan tidigare tentor och tentan i år var att JDBC som sagt helt tagits bort i år.

Den första frågan var enligt lärarna en gratispoäng på grund av att den var lätt. Det var en flervalsfråga som undersökte kunskaper om egenskaper hos en DBMS, vad "en organized collection of data" är, samt vad syftet med att använda en DBMS är. Frågorna var lika året innan men svarsalternativen var nya. 68 av 84 studenter hade alla rätt på den frågan. Men 78 (63 G plus 15 VG) studenter hade minst G på tentan. Så tio studenter av de godkända missade någon flervalsfråga.

Oavsett huruvida en tenta är lik en tidigare tenta eller inte så anser lärarna att det är en dålig strategi att lära sig svar utantill. För att motverka utantillkunskaper så upplyser vi alltid studenterna om att det kommer finnas ett appendix med manual för SQLite samt att vi är generösa när vi rättar (till exempel straffar vi småfel endast en gång och bortser från upprepade identiska småfel).

Plan för dig som fått U på tentan

Gör övningarna, så att du åtminstone vet hur man skapar en databas utifrån en SQL-fil och vet vad det är för skillnad på SQL-filen och databasfilen.

```
sqlite3 students.db < students.sql
```

Läs även på här: https://sqlite.org/cli.html

Ingen av dem som fick U kunde svara på denna fråga med full poäng (2+1 poäng):

> ***2-G. SQLite-databashanteraren (3p)***
> *2.1 Du har en fil, t ex students.sql, med SQL-kommandon. Skriv en (1) kommandorad för bash för att sqlite ska utföra alla dessa kommandon för att skapa en databas, t ex students.db. (2p)*
>
> *2.2 Beskriv skillnaden i syfte och innehåll mellan filerna students.sql respektive students.db efter kommandoraden körts. (1p)*

Läs på om normalisering och JOIN. Det var den fråga U-studenterna hade minst antal poäng på i snitt(44% av poängen på frågan i snitt). Här är några länkar:

- http://wiki.juneday.se/mediawiki/index.php/Database:Combining_rows_of_data_from_related_tables_-_SQL_JOIN
- http://wiki.juneday.se/mediawiki/index.php/Database:Combining_rows_of_data_from_related_tables_-_SQL_JOIN#Decomposing_a_table_-_English_videos
- http://wiki.juneday.se/mediawiki/index.php/Database:Combining_rows_of_data_from_related_tables_-_SQL_JOIN#Extra_lecture_on_JOINs_-_English_videos
- http://databasteknik.se/webbkursen/normalisering/index.html
- http://databasteknik.se/webbkursen/sql/index.html
- http://www.databasteknik.se/webbkursen/relalg-lecture/index.html
- https://www.w3schools.com/sql/sql_join.asp

Läs på om Constraints, där var U-studenterna också svaga:

- https://sqlite.org/lang_createtable.html#constraints
- http://wiki.juneday.se/mediawiki/index.php/Database:Adding_constraints_to_prevent_garbage_data
- https://sqlite.org/foreignkeys.html
- https://www.w3schools.com/sql/sql_constraints.asp
    - https://www.w3schools.com/sql/sql_notnull.asp
    - https://www.w3schools.com/sql/sql_unique.asp
    - https://www.w3schools.com/sql/sql_primarykey.asp
    - https://www.w3schools.com/sql/sql_foreignkey.asp
    - https://www.w3schools.com/sql/sql_check.asp

Läs även på om SELECT, WHERE, LIKE och ORDER BY. ***Gör övningarna***.

Gör check your progress-testerna i wikin.

Saker som förvånat lärarna

Det var förvånande att så pass många hade problem med frågan om SQLite som löd:

> *2.1 Du har en fil, t ex `students.sql`, med SQL-kommandon. Skriv en (1) kommandorad för bash för att sqlite ska utföra alla dessa kommandon för att skapa en databas, t ex `students.db`. (2p)*
>
> *2.2 Beskriv skillnaden i syfte och innehåll mellan filerna `students.sql` respektive `students.db` efter kommandoraden körts. (1p)*

En oroväckande stor andel hade fel på första frågan och svarade något i stil med:

```
students.db < students.sql
```

Det vill säga utelämnade kommandot `sqlite3` . Rätt svar skulle vara:

```
sqlite3 students.db < students.sql
```

Ovan är `sqlite3` det kommando vi använt några hundra gånger under kursen för att starta den databashanterare kursen använt.

Vi har inte statistik på delfrågenivå men lade märke till att ganska många missade denna fråga, vilket enligt lärarna var mer eller mindre “gratispoäng”. Det konstiga var att detta är något man gör i stort sett i varje övningsdel till varje kapitel i kursmaterialet. Och detta var något som gjordes i varje live-föreläsning tillsammans med en repetition om innebörden.

Det var i synnerhet underkända studenter som hade problem med frågan men hela 21 studenter saknade full poäng på denna simpla fråga. Lärarna misstänker starkt att detta beror på två saker:

1. Studenterna i fråga har inte gjort tillräckligt många övningar
2. Studenterna i fråga har inte deltagit i tillräckligt många live-föreläsningar

Detta med filer och omdirigering är något vi tror kommer bli bättre med en nya delkursen i TIG015 som har premiär under höstterminen 2019. Att förstå filer, filformat och omdirigering är enligt lärarna en grundläggande färdighet för alla typer av arbete inom IT, i synnerhet när det gäller programmering och databaser.

En annan sak som förvånade oss är att trots de nya föreäsningsfilmerna om WHERE och NULL så var det ganska många (även här saknas statistik på delfrågenivå) som missade hur man skapar ett sammansatt WHERE-villkor, till exempel med AND eller OR. Ganska många skrev:

```
WHERE color LIKE ‘g%’ OR ‘G%’
```

För att ovanstående ska ha rätt syntax borde det i så fall ha varit:

```
WHERE color LIKE ‘g%’ OR color LIKE ‘G%’
```

Ett förtydligande om hur man använder AND, OR och så vidare har tillfogats kapitlen om SELECT och UPDATE. I frågan ovan så har studenterna även missat att LIKE-operatorn i SQLite-SQL är "*case insensitive*" så OR är helt överflödigt. Vi satte ändå rätt på de olika varianterna så länge som `color LIKE 'g%'` var med eftersom det är svårt att upptäcka små syntaxfel när man skriver med penna och papper och dessutom för att det finns databaser, t ex PostgreSQL, som har en LIKE som är case sensitive. Samma problem (eller åtminstone ett närbesläktat problem) hade ganska många studenter med syntaxen för UPDATE när det gäller att uppdatera mer än en kolumn. Rätt syntax är t ex:

```
UPDATE cars SET color = 'Blue', make = 'Volvo' WHERE...
```

Här *ska det vara kommatecken* men en hel del studenter använde AND vilket inte alls passar in mellan kolumnuppdateringar. AND, OR osv har ju med logiska värden att göra och därmed något som torde återfinnas i WHERE-klausulen. Ändå skrev rätt många något i stil med:

```
set color = 'Blue' AND make = 'Volvo'
```

och detta trots att vi bifogade syntaxdiagram för UPDATE-satsen. Detta är som sagt tillfogat kapitlet om UPDATE och SELECT som en följd av detta mysterium. Även här misstänker vi att de studenter som har problem med detta kanske borde gjort lite fler övningar. Åtminstone kunde de tänka på att läsa syntaxdiagrammen där det i fallet UPDATE är tydligt att man kan ha fler kolumner uppdaterade och att man då ska separera dem med kommatecken.

När det gäller felaktigt användande av AND i UPDATE så var vi hårdare och drog av något halvt poäng för det (första gången och lät efterföljande fel av samma karaktär bero). Här fanns ju syntaxdiagrammet att tillgå men också för att vi tycker det är ett allvarligare fel att tro att man kan använda AND som en separator i en lista av kolumner att uppdatera, än att glömma att upprepa "color LIKE" vid en sammansatt WHERE-sats.

Vanliga misstag vid JOIN

Även om JOIN-frågorna (och frågan om att bryta upp en tabell i flera kopplade tabeller) gick betydligt bättre i år (utan att vara lättare än förra årets motsvarande frågor), så var det ganska många som missade att utföra JOIN mellan två av tabellerna. T ex kunde det se ut ungefär så här:

```
SELECT title, genre, artist FROM track NATURAL JOIN genre,artist;
```

Rätt svar hade i så fall varit:

```
SELECT title, genre, artist FROM track NATURAL JOIN genre NATURAL JOIN artist;
```

Här får vi lägga till material och övningar som förklarar vad som händer när man gör en SELECT från två tabeller med kommatecken (så kallad Cartesisk produkt).

När det gäller VG-frågan om “Vilken färg är det ingen bil som har” så var vi ute efter LEFT OUTER JOIN (eller LEFT JOIN):

```
SELECT color
  FROM color LEFT JOIN car
    ON car.color_id = color.color_id
 WHERE car.license IS NULL;
```

Eller:

```
SELECT color FROM color
        LEFT JOIN car USING(color_id)
            WHERE car.license IS NULL;
```

Här såg vi många fantasifulla försök men mycket få studenter hade rätt på denna fråga. Detta tror vi beror på att få studenter såg filmerna om “More JOINs” och få studenter gick på repetitionspasset där vi gick igenom LEFT JOIN. Vi hade även live-föreläsning där vi visade LEFT JOIN men det var långt ifrån fullsatt vid det tillfället. Det var dock en VG-fråga och LEFT JOIN var inte med förra året, så det fungerade ändå relativt rättvist att ha detta som en VG-fråga anser vi lärare. Men synd att så få kommer på föreläsningar och repetitioner.