Databaser — design och programmering

Transaktionshantering och säkerhet

- säkerhetsproblem
- fleranvändarproblem
    - transaktioner
    - låsning

LINKÖPINGS UNIVERSITET

Säkerhetsproblem

- Informationen i databasen
  - måste vara pålitlig (inte kunna ändras, bli fel eller försvinna av misstag)
  - får inte spridas till obehöriga
  - måste vara åtkomlig när den behövs
- Men: om datorn hänger sig, hårddisken kraschar, obehöriga försöker ta sig in… eller någon är bara klantig?

Databashanteraren

… har rutiner för många av dessa problem

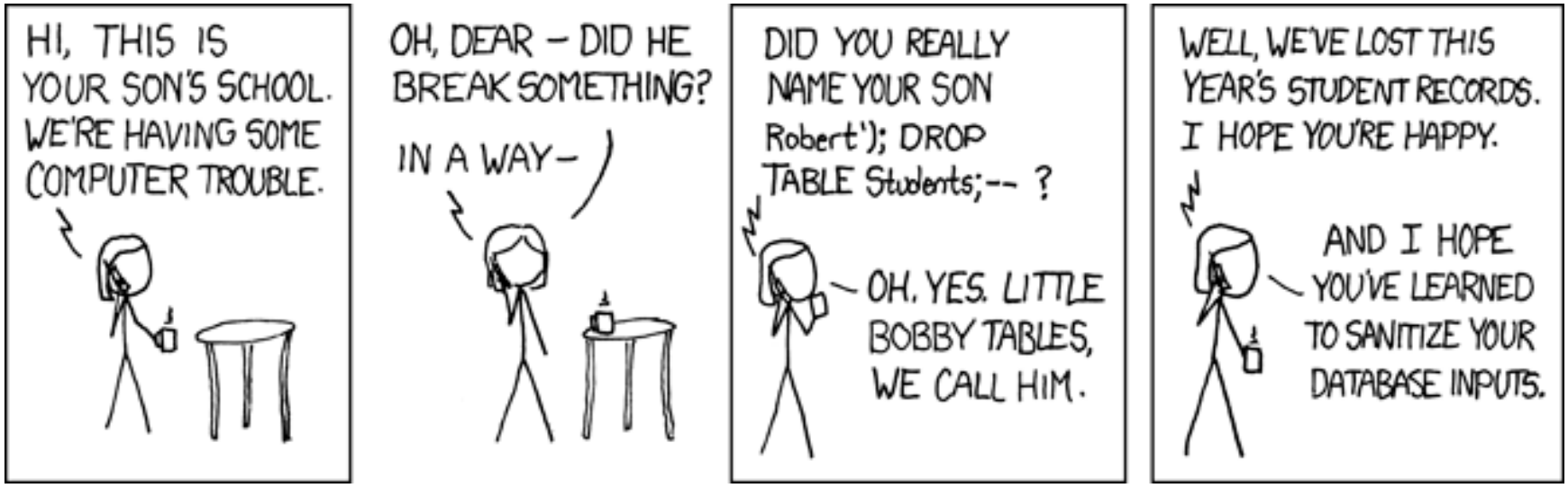
- Inloggning
- Säkerhetsnivåer (olika användare olika behörighet)
- SQL-kommandon: grant och revoke

  grant select **on** employee **to** user-id
- vyer eller begränsning av kommandon

Statistiska databaser och integritet

- Statistiska data som samlas in för analys
- Integritetsproblem: kunna analysera men inte identifiera individer
  - ex politisk åsikt - utan att kunna identifiera vad grannen röstade på
  - vyer?
  - bara tillåta aggregerade frågor?

SQL-injektion



Källa: xkcd.com/327/

Korrekthet: Exempel

Du ska betala en räkning genom att överföra pengar från ditt konto till mottagarens.

- Begär överföring från ett konto till annat.
- Tryck “bekräfta”
  - beloppet dras från det ena kontot
  - sedan hänger sig systemet.
- insättningen registrerades inte!

Transaktion

Definition: Logiskt sammanhängande serie interaktioner med databasen, som är:

- Atomära (**A**tomic) : odelbara
- Konsistenta (**C**onsistent): integritetsbevarande
- Isolerade (**I**solated): oberoende av annat som görs i db.
- Bestående (**D**urable): inte kunna försvinna.

Transaktion

Överföring mellan två konton (X och Y)

Läsning och skrivning - till databasen!

Pseudokod:

```
Läs(X)
X = X - uttag
Skriv(X)
Läs(Y)
Y = Y + uttag
Skriv(Y)
```

Transaktion

En helhet: markera start och slut!

Pseudokod:

```
Start transaction
Läs(X)
X = X - uttag
Skriv(X)
Läs(Y)
Y = Y + uttag
Skriv(Y)
Commit
```

Spara databasoperationer i Loggfil

```
start transaction
Läs(X)
X = X - uttag
Skriv(X)
Läs(Y)
Y = Y + uttag
Skriv(Y)
commit transaction
```

```
Loggfil:
start (t234)
read (t234, X)

write (t234, X, 360, 260)
read (t234, Y)

write (t234, Y, 120, 220)
commit (t234)
```

Återställa transaktion: Rollback

Om transaktionen avbryts innan commit:

Återställ de tidigare värdena med hjälp av loggfilen

```
Loggfilen

start (t234)

read (t234, X)

write (t234, X, 360, 260)

read (t234, Y)
```

Återställning mer generellt

Efter krasch är läget odefinierat. Måste säkra databasen!

- Hämta från backup om hårddiskproblem
- Transaktioner som ej slutförts: rollback
- Transaktioner som slutförts efter senaste backup repeteras

Hur långt tillbaka som helst?

- Checkpoint

OBS: Skriv till loggfilen först, sedan till databasen!

Transaktion + Loggfil + Backup => ACID?

A (atomär)	-	Start/Commit/Rollback/loggfil
C (konsistent)	-	A + kontroll av integritetsvillkor i transaktionen
I (oberoende)	-	?
D (bestående)	-	backup + loggfil

Problem med parallella transaktioner

Varor

Artikelnr	Namn	i lager	Pris
2398475	Rosor, 10-p	3	79,9
9834576	Tulpan, 10-p	15	49,9

Dagens försäljning

Artikelnr	Antal	Dag
2398475	3	180506
9834576	1	180507
2398475	2	180507

Ex: Uppdatering av varutabellen

Kassaförsäljning:

```
start transaction
Läs(X) (varan ros)
X = X - 1
Skriv(X)
Läs(Y)(dagens förs, ros)
Y = Y + 1
Skriv(Y)
commit transaction
```

Lagerhantering:

```
start transaction
Läs(X) (varan ros)
X = X + levererat ant
Skriv(X) (varan ros)
commit transaction
```

Tid	Försäljning	Lagerhantering
1		Start transaction
2		Läs(X)
3		X = X + Levererat
4	Start Transaction	
5	Läs (X)	
6	X = x-1	
7		Skriv (X)
8		Commit Transaction
9	Skriv (X)	
10	Läs (Y)	
11	Y = Y +1	
12	Skriv (Y)	
13	Commit Transaction	

Bortkastad uppdatering

Tid	Försäljning	Lagerhantering
1	Start Transaction	
2	Läs (X)	
3	X = x-1	
4	Skriv (X)	
5		Start transaction
6		Läs(X)
7		X = X + Levererat
8		Skriv(X)
9		Commit Transaction
10	Rollback	
11	~~Läs (Y)~~	
12	~~Y = Y +1~~	
13	~~Skriv (Y)~~	

Smutsig läsning

Tid	Överföring	Summering saldo
1		Start transaction
2		Sum=0
3		Läs (X)
4		Sum = Sum + X
5	Start Transaction	
6	Läs (X)	
7	X = x-m	
8	Skriv (X)	
9	Läs (Y)	
10	Y = Y +m	
11	Skriv (Y)	
12	Commit Transaction	
13		Läs (Y)
14		Sum = Sum + Y

Felaktig summering

Parallella transaktioner: lås

Problem med **I**soleringen (och **K**onsistensen)

Lösning: reservera åtkomsträttigheten för en artikel till en transaktion: **lås**

```
Lås (X), LåsUpp (X)
```

Vad X är avgörs av låsets granularitet (tabell/rad/cell)

(Alternativ: t.ex. serialiserbarhet och tidsstämpling, studeras ej)

Låsning - binära lås

Två tillstånd: **Låst, Olåst**.

Protokoll för binära lås (måste följas!):

1. `Lås(X)` måste utföras innan någon `Läs(X)` eller `Skriv(X)`-operation utförs.

2. `LåsUpp(X)` måste utföras när läsning och skrivning av X är klar.

3. man får inte göra `Lås(X)` om man redan har låst X.

4. man får inte göra `LåsUpp(X)` om man inte har låst X för tillfället.

Exempel lås

Försäljning:

```
Start transaction
Läs(X)
X = X-1
Skriv(X)

Läs(Y)
Y = Y + 1
Skriv(Y)
Commit
```

Lagerhantering:

```
Start trans
Läs(X)
X = X + Levererat
Skriv(X)
Commit
```

Exempel lås

Försäljning:

```
Start transaction
Lås(X)
Läs(X)
X = X-1
Skriv(X)
LåsUpp(X)
Lås(Y)
Läs(Y)
Y = Y + 1
Skriv(Y)
LåsUpp(Y)
Commit
```

Lagerhantering:

```
Start trans
Lås(X)
Läs(X)
X = X + Levererat
Skriv(X)
LåsUpp(X)
Commit
```

Läs- och skrivlås

Parallell läsning fungerar bra: skilj på läsning och skrivning.

```
LäsLås(X)
```

```
SkrivLås (X)
```

```
LåsUpp(X)
```

LäsLås går bara igenom om inget SkrivLås är satt.

SkrivLås går bara igenom om inget lås alls är satt.

Läs- och skrivlås: protokoll:

1. Innan någon `Läs(X)`-operation utförs måste `LäsLås(X)` **eller** `SkrivLås(X)` utföras.
2. Innan någon `Skriv(X)`-operation utförs måste `SkrivLås(X)` utföras.
3. `LåsUpp(X)` måste utföras när läsning och skrivning av X är klar.
4. man får inte göra `LäsLås(X)` om man redan har låst X.
5. man får inte göra `SkrivLås(X)` om man redan har låst X för skrivning.
6. man får inte göra `LåsUpp(X)` om man inte har låst X för tillfället.

Men...

Tid	Överför	Summer
1		Sum=0
2		LäLås(X)
3		Läs(X)
4		Sum=Sum+x
5		LåsUpp(X
6	SkLås(X)	
7	Läs(X)	
8	x=x-m	
9	Skriv(x)	
10	LåsUpp(X	
11		

Tid	Överför	Summer
12	SkLås(Y)	
13	Läs Y	
14	y=y+m	
15	Skriv(y)	
16	LåsUpp(Y	
17		LäLås(Y)
18		Läs(y)
19		Sum=Sum+
20		y
21		LåsUpp(y
22		

Felaktig summering

Tvåfaslåsning

Protokoll som tidigare plus:

Två faser:

1. Lås

2. Lås upp

->Inte släppa något lås förrän allt som behövs för transaktionen har låsts!

Men: om problem uppstår så någon transaktion får avbrytas?

Men…

Tid	**Försäljn**	**Lagerh**
1	Start t	
2	**Lås(X)**	
3	Läs(X)	
4	x=x-1	
5	Skriv(x)	
6	**Lås(Y)**	
7	**LåsUpp(X**	
8		Start t
9		Lås(X)
10		Läs(X)
11		x=x+lev

Tid	**Försäljn**	**Lagerh**
12		Skriv(x)
13		LåsUpp(X
14		Commit
15		
16	ROLLBACK	
17	~~Läs Y~~	
18	~~y=y+1~~	
19	~~Skriv(y)~~	
20	~~LåsUpp(Y~~	

Read uncommited

Kaskad-rollback?

Om en committad transaktion beror av en annan som gör Roll-back?

Rulla tillbaka även den commit-ade transaktionen.

Men då förlorar vi D (Durable, bestående)

Tvåfaslåsning inte tillräcklig!

Förbättrad tvåfaslåsning

- Rigorös tvåfaslåsning
  - Släpp inte något lås förrän hela transaktionen är commitad!
- Strikt tvåfaslåsning
  - Läslås kan släppas i upplåsningsfasen men skrivlås släpps inte förrän hela transaktionen är committad

Nu är våra transaktioner ACID!

Men...

```
Tid: Trans 1:                Trans 2:
1    SkrivLås(X)
2                            SkrivLås(Y)
3    …                       …
4    SkrivLås(Y)
5    VÄNTAR!                 SkrivLås(X)
6                            VÄNTAR!
.
.    LåsUpp(X)               LåsUpp(Y)
.    LåsUpp(y)               LåsUpp(X)
```

Deadlock!

Dödläge (deadlock):

Definition:

Korsvis (eller cirkulär, om flera transaktioner är inblandade) låsning av objekt i databasen,

sådan att ingen kan släppa en artikel förrän den fått låsa en artikel som är låst av någon som väntar på den artikel man redan låst.

Deadlock, strategier

Förebyggande:

1. Konservativ tvåfaslåsning (lås allt på en gång)
2. Dataobjekten låses alltid i en viss ordning

Upptäckande

1. Time-out
2. undersöka wait-for-grafen

Transaktioner och säkerhet - begrepp

Säkerhet (inloggning, rättigheter, back-up)

ACID: Transaktioner är odelbara, konsistensbevarande, isolerade och bestående.

- Backup, återställning
- Loggning (loggfil), start/commit,
- Rollback, kaskad-rollback
- Låsningssystem: Binäralås, Läs/Skrivlås
  - Protokoll: tvåfaslåsning - strikt/rigorös
  - Deadlock

Frågor?

www.liu.se

LINKÖPINGS UNIVERSITET