Uke 8 - Mer normalformer :)

Av Katrine :3

Gjennomgang av oblig 1

Algoritmen for å finne NF

- Først: Skrive om FDer og fjerne redundans
- Sist sa vi at A -> B og A -> C kunne skrives A -> B,C
- Men her ønsker vi at det bare står én på høyresiden
- Det skal ikke gjøres på venstre, husk at den siden sier hvilke attributter som kreves for å finne høyresiden
- Hvis X -> B, så vil X,Y -> B for alle Y også, men det er overføldig, så da fjerner vi Y

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

A → D
B → C
B → E
AC → F

Bare høyre: D, E
Bare venstre: A, B
Begge: C

{A, B}+ = A, B, D, C, E, F

Fordi AC er i tillukningen

Trinn 1: Finn kandidatnøkler

- Om dere husker, så vil alle attributter som opptrer på bare venstre være en del av kandidatnøkkelen
- Og de som bare er på høyre vil ikke være det
- Og de som er på begge vil kunne være del av forskjellige kandidatnøkler

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

A → D
B → C
B → E
AC → F

Bare høyre: D, E
Bare venstre: A, B
Begge: C

{A, B}+ = A, B, D, C, E, F

Fordi AC er i tillukningen

Trinn 2: Finn normalformen til FDene

- FD 1: A != {AB}, ikke en supernøkkel, og bryter med BCNF
- FD 1: D er ikke i {AB}, og bryter med 3NF
- FD 1: A er i {AB}, bryter med 2NF
- Konklusjon, R er på 1NF
- Relasjonen er alltid på den laveste NF

Tipsfri dekomponering

- Vi har nå sett på NF til en gitt relasjon med sine FDer og resulterende kandidatnøkler
- For å gjøre tabeller som ikke er BCNF til det, så dekomponerer vi tapsfritt
- Det vil si at vi deler data som ikke hører sammen opp i mindre tabeller på en måte som lar oss natural joine dem sammen igjen til det vi hadde
- Det viktigste når du dekomponerer tapsfritt er å holde på forholdet mellom de nye tabellene

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

$A \rightarrow D$
$B \rightarrow C$
$B \rightarrow E$
$AC \rightarrow F$

Bare høyre: D, E
Bare venstre: A, B
Begge: C

{A, B}

Tapsfri dekomponering av $R(X)$ med FDer F:

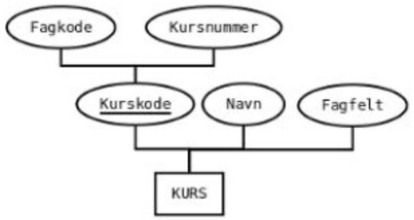
1. Beregn nøklene til R
2. For hver FD $Y \rightarrow A \in F$, hvis FDen er et brudd på BCNF:
   2.1 beregn Y^+,
   2.2 og dekomponer R til $S_1(Y^+)$ og $S_2(Y, X/Y^+)$.
3. Fortsett rekursivt (over S_1 og S_2) til ingen brudd på BCNF

(Hvis en FD inneholder attributter fra ulike tabeller kan den ignoreres)

Eksempel på tapsfri dekomponering

- 1. Nøkkelen AB
- 2. AC -> F er et brudd på BCNF (Vi starter med den FDen med flest avhengigheter)
- 2.1 Dekomponer R på AC -> F => AC+ = ACFD
- 2.2 S1(ACFD) og S2(AC, ABCDEF/ACFD) = S2(ABCE)
- 2.1. Dekomponer S1 på A -> D => 2.1 A+ = AD
- 2.2 S11(AD) og S12(A, ACFD/AD) = S21(ACF). S11 og S12 er nå på BCNF
- 2.1 Dekomponer S2 på B -> C => B+ = BCE (Siden B også finner E)
- 2.2 S21(BCE) og S22(B, ABCE/BCE) = S222(AB). S21 og S22 er nå på BCNF

- ER-modell:



- FDer:
  - $Fagkode, Kursnummer \rightarrow Navn$
  - $Fagkode \rightarrow Fagfelt$ ← Kan ikke uttrykkes i ER!
- Realiseres til:

$$\texttt{Kurs}(\underline{\texttt{Fagkode}, \texttt{Kursnummer}}, \texttt{Navn}, \texttt{Fagfelt})$$

- $Fagkode \rightarrow Fagfelt$ bryter med 2NF fordi $Fagkode$ kun er en del av kandidatnøkkelen.

Fra algoritme til design

- En tabell per entitetstype (fornavn skal i fornavn, og etternavn skal i etternavn)
- Relasjoner mellom entiteter representeres ved fremmednøkkelreferanser (1:1 og 1:N) eller via egen tabell (N:M)
- Her ser dere at det som logsik gir mening, fagkode og kursnummer som nøkkel, gir en dårlig FD

$$R(\texttt{Brukernavn}, \texttt{Navn}, \texttt{Etternavn}, \texttt{KursKode}, \texttt{KursNavn})$$

FDer:

1. $\texttt{Brukernavn} \rightarrow \texttt{Navn}$
2. $\texttt{Brukernavn} \rightarrow \texttt{Etternavn}$
3. $\texttt{KursKode} \rightarrow \texttt{KursNavn}$

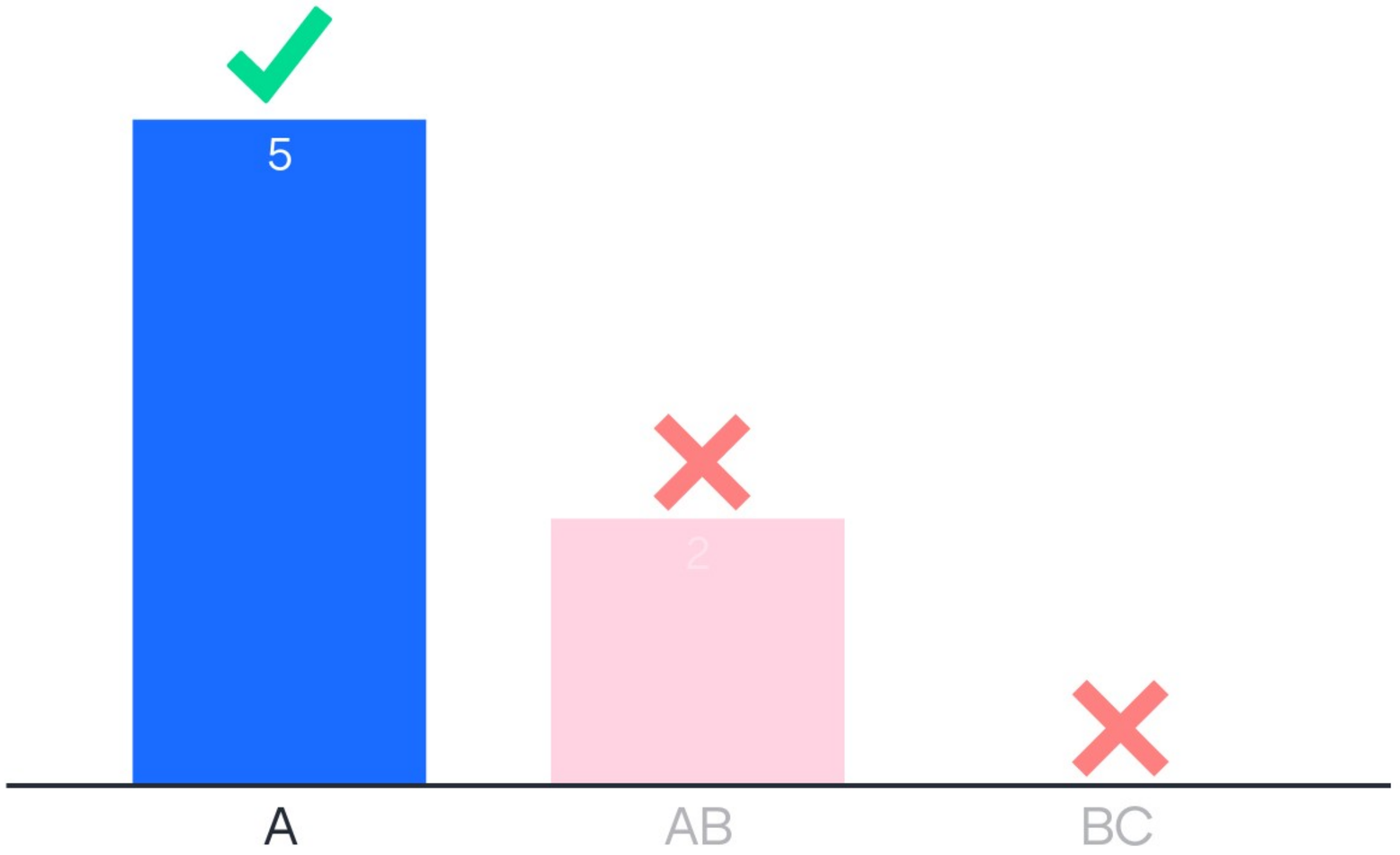
- Kan dekomponeres til:
  - $S(\texttt{Brukernavn}, \texttt{Navn})$
  - $T(\texttt{Brukernavn}, \texttt{Etternavn})$
  - $U(\texttt{Brukernavn}, \texttt{KursKode})$
  - $V(\texttt{KursKode}, \texttt{KursNavn})$

En algoritme gir ikke nødvendigvis godt design

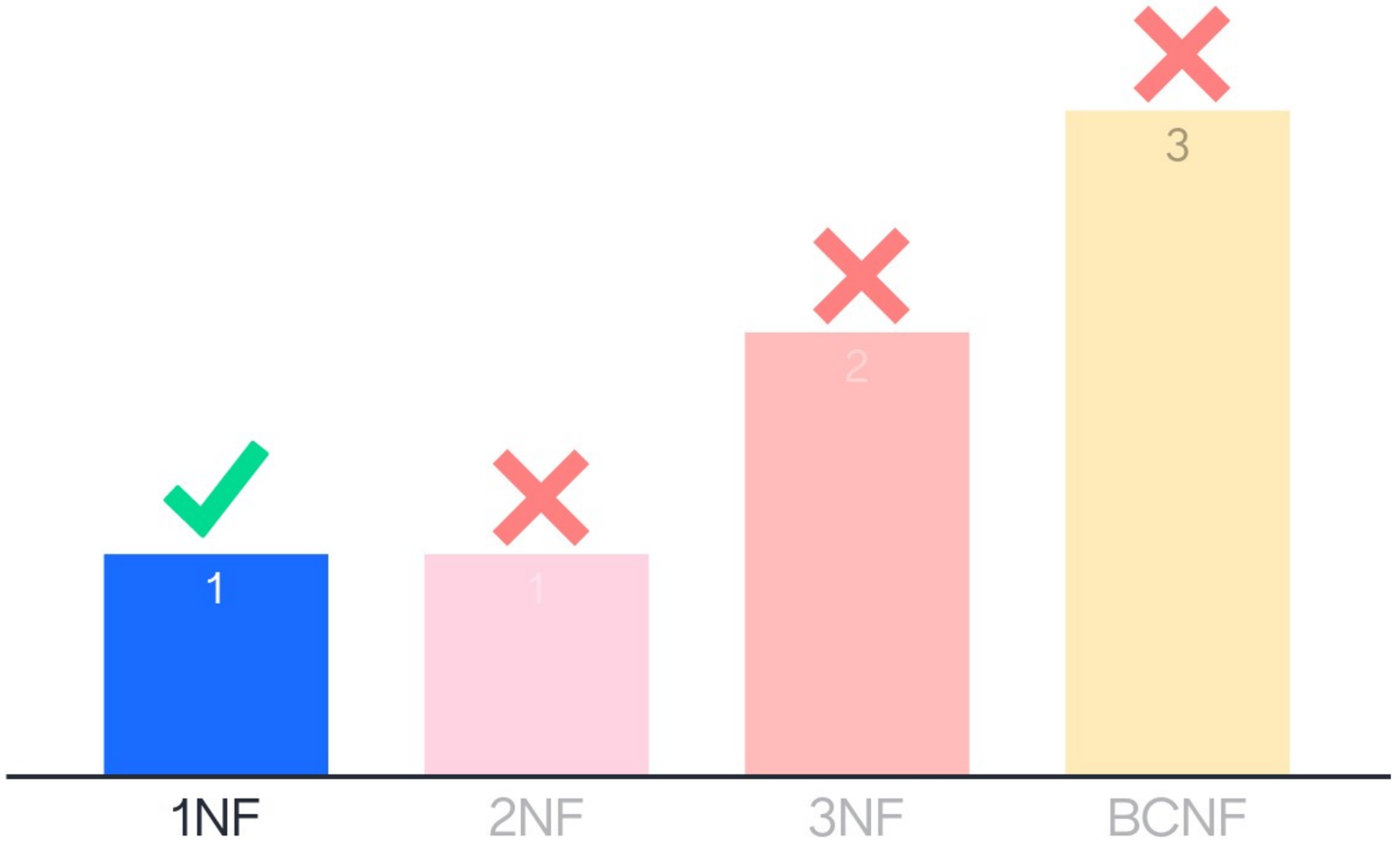
- Som nevnt forrige gang, er høy NF = få anomaliteter, men også mange relasjoner
- Eksempelet på bildet gir oss 4 relasjoner med dekomponering
- Men det gir ikke noe praktisk mening å ha navn og etternavn som egne relasjoner, når brukernavn peker på det fulle navnet til brukeren!
- Derfor er godt design en kombinasjon av algoritmen for å luke ut så mange anomaliteter som mulig, og sunt vett for hva databasen skal brukes til :)

QUIZ!

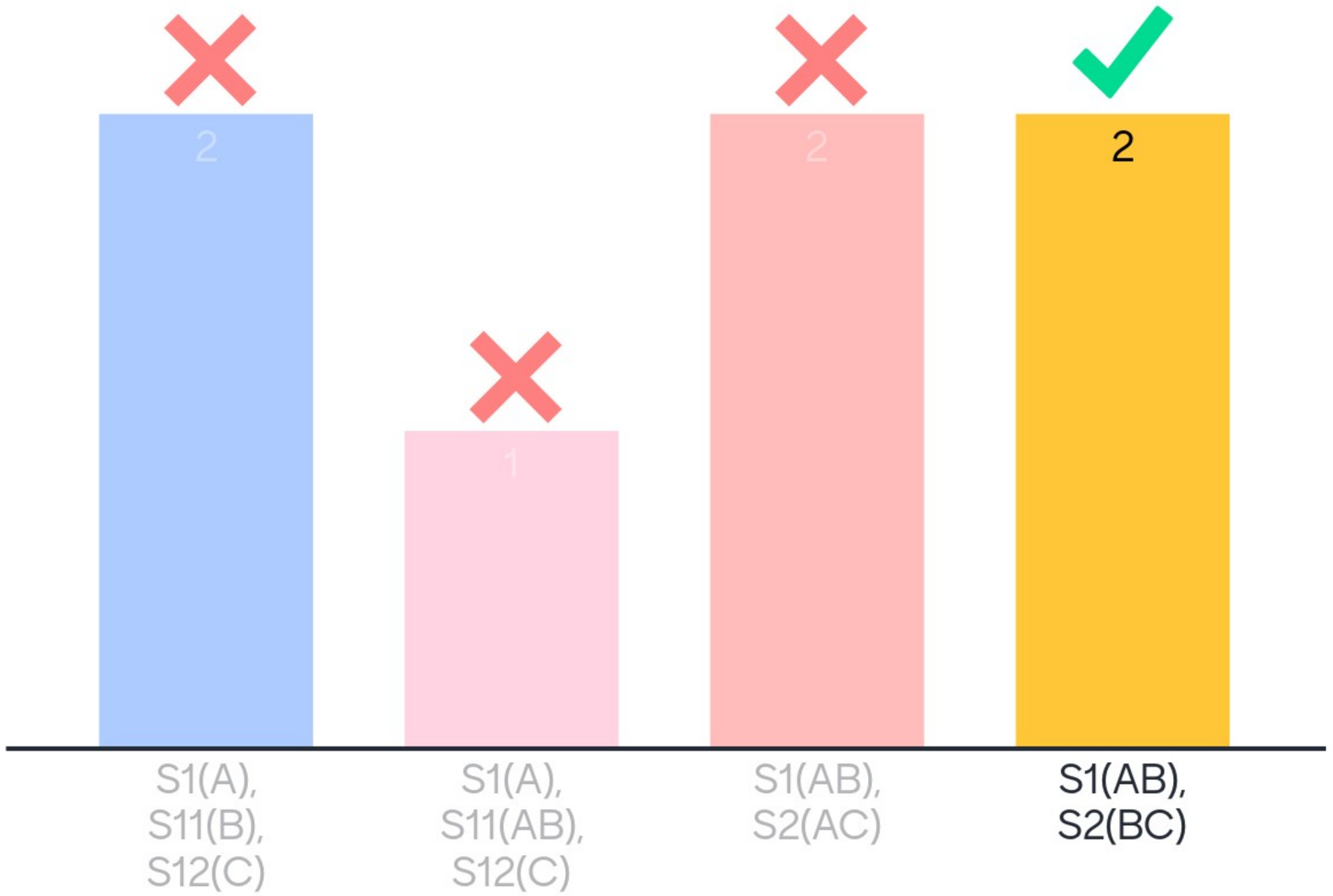
Select Answer



Select Answer



Select Answer



Leaderboard

Mentimeter

1749 p		Mars *fastest*
1726 p		Tom
1644 p		Frosty
962 p		m
822 p		ingen
0 p		Puggie
0 p		Maverick

7