IN2090 – Databaser og datamodellering

Databasedesign og normalformer

Evgenij Thorstensen
evgenit@ifi.uio.no

Universitetet i Oslo

Oversikt

- Gode og dårlige skjemadesign (og litt historie)
- Funksjonelle avhengigheter og nøkler
- Normalformer
- Dekomponering

Et lite case

Vi trenger en database for å lagre data om studenter, kurs, og karakterer.

- For studenter: Brukernavn, navn, etternavn, adresse...
- For kurs: Kurskode, tittel, beskrivelse, ant. studiepoeng...
- Hvem har fått hvilken karakter i hva.

Naiv løsning: Alt i en tabell.

Skjema

Brnavn	Navn	Etternavn	Adresse	Kurskode	Tittel	Beskrivelse	AntSP	Kara
evgenit	Evgenij	Thorstensen	Addr1	IN2090	Databaser	EnBeskr...	10	B
peternl	Petter	Nilsen	Addr2	IN2090	Databaser	EnBeskr...	10	A
evgenit	Evgenij	Thorstensen	Addr1	IN2080	Beregn...	Descr...	10	A

Hvorfor er dette et dårlig design, annet enn at det er rotete?

Hvilke praktiske problemer har vi her?

Anomalier

I skjemaet over er det mange anomalier:

- Dataduplisering
- Kronglete å legge til studenter uten kurs, kurs uten studenter.
- Å slette et kurs kan slette en student, og omvendt.

Vi kan unngå disse med et bedre design.

Et bra skjema

Tabell Student:

Brnavn	Navn	Etternavn	Adresse
evgenit	Evgenij	Thorstensen	Addr1
peternl	Petter	Nilsen	Addr2

Tabell Kurs:

Kurskode	Tittel	Beskrivelse	AntSP
IN2090	Databaser	EnBeskr...	10
IN2080	Beregn...	Descr...	10

Tabell Karakterer:

Brnavn	Kurskode	Karakter
evgenit	IN2090	B
peternl	IN2090	A
evgenit	IN2080	B

Merk at vi har akkurat de samme attributtene!

Prinsipper for databasedesign

Data om entiteter i hver sin tabell.

Relasjoner mellom entiteter via referanser.

Etc... blah blah... Kan dette gjøres mer systematisk?

La oss se på skjemaet en gang til.

Et bra skjema, en gang til

Tabell Student:

Brnavn	Navn	Etternavn	Adresse
evgenit	Evgenij	Thorstensen	Addr1
peternl	Petter	Nilsen	Addr2

Tabell Kurs:

Kurskode	Tittel	Beskrivelse	AntSP
IN2090	Databaser	EnBeskr...	10
IN2080	Beregn...	Descr...	10

Tabell Karakterer:

Brnavn	Kurskode	Karakter
evgenit	IN2090	B
peternl	IN2090	A
evgenit	IN2080	B

Brnavn identifiserer alt om en student, Kurskode om et kurs.

Normalformer

Det går an å formelt definere kriterier som gjør at anomalier ikke forekommer.

Et skjema som oppfyller kriteriene, sies å være på en normalform.

Vi skal se på normalformene 1NF, 2NF, 3NF, og BCNF.

For å komme dit, trenger vi litt teori.

Funksjonell avhengighet

Et attributt A er **funksjonelt avhengig** av an mengde attributter X hvis det bare kan finnes en verdi av A for hver mengde verdier av attributtene i X.

Det skrives $X \rightarrow A$, og en slik formel kalles en funksjonell avhengighet (FD).

For eksempel er Karakter funksjonelt avhengig av $\{$Brnavn, Kurskode$\}$ i tabellen vi hadde:

Brnavn	Kurskode	Karakter
evgenit	IN2090	B
peternl	IN2090	A
evgenit	IN2080	B

Mer om FDer

FDer uttrykker det vi mener er sant om dataene våre. Kan fort bli et komplisert spørmål om verdens tilstand.

Mer om FDer

FDer uttrykker det vi mener er sant om dataene våre. Kan fort bli et komplisert spørmål om verdens tilstand.

Vi kan samle opp høyresider i FDer, og skrive $X \to A, B$ dersom vi både har $X \to A$ og $X \to B$.

FDer er transitive: Hvis $X \to Y$ og $Y \to Z$, så har vi at $X \to Z$.

FDer forteller oss hvilke data hører sammen, og hva de hører til.

En FD $X \to Y$ hvor $Y \subseteq X$ kalles triviell, og vi ignorerer slike (hvorfor?)

Eksempel, FDer

Vi startet med en relasjon

$$R(Brnavn, Navn, Etternavn, Adresse, Kurskode, Tittel, Beskrivelse, AntSP, Karakter)$$

Eksempel, FDer

Vi startet med en relasjon

$$R(\mathit{Brnavn}, \mathit{Navn}, \mathit{Etternavn}, \mathit{Adresse}, \mathit{Kurskode}, \mathit{Tittel}, \mathit{Beskrivelse}, \mathit{AntSP}, \mathit{Karakter})$$

Jeg foreslår følgende FDer:

- Brnavn $\rightarrow$ Navn, Etternavn, Adresse
- Kurskode $\rightarrow$ Tittel, Beskrivelse, AntSP
- Brnavn, Kurskode $\rightarrow$ Karakter

Nøkler

En **supernøkkel** for en relasjon er enhver mengde attributter som entydig bestemmer resten av attributtene.

En **kandidatnøkkel** (ofte bare nøkkel) er en $\subseteq$-minimal supernøkkel.

La R være en relasjon med attributter X.

$Y \subseteq X$ er en supernøkkel for R hvis $Y \rightarrow X \setminus Y$, som er evivalent med $Y \rightarrow X$ (hvorfor?)

$Y \subseteq X$ er en kandidatnøkkel for R hvis Y er en supernøkkel og **ingen** $Z \subset Y$ er en supernøkkel.

Mer om nøkler

Anomalier oppstår når en relasjon inneholder data som ikke er direkte funksjonelt avhengige av en kandidatnøkkel.

Ved å bruke nøkler og FDer skal vi definere dette presist.

Men først, hvordan finner vi kandidatnøklene i en relasjon?

Tillukning

En kandidatnøkkel er en supernøkkel (bestemmer alle attributter), og er minimal.

For å sjekke om X er en supernøkkel, sjekk om alt er avhengig av X. Ekvivalent, bruk FDene og finn alle attributter som er avhengige av X, de som er avhengige av disse igjen, osv.

Tillukningen X^+ av X på en mengde FDer er mengden attributter som er funksjonelt avhengige av X. Hvis $A \notin X^+$, så er ikke $X \to A$ sant.

Tillukningen kan regnes ut ved å bruke FDene om og om igjen:

- sett $X^+ = X$
- sålenge X^+ forandres:
- finn en FD $Y \to Z$ med $Y \subseteq T$
- sett $X^+ = X^+ \cup Z$

Finne kandidatnøkler

Vi må sjekke alle delmengder av attributter, nedenfra. Men, følgende to regler hjelper oss:

- Hvis A ikke forekommer i noen høyreside, er A med i **alle** kandidatnøkler.
- Hvis A forekommer i minst en høyreside, men ingen venstresider, er A **ikke del** av noen kandidatnøkkel.

Så begynn med alle attributter som ikke forekommer på høyre side. Beregn tillukningen.

Hvis alle attributter er med, sjekk minimalitet. Hvis ikke, utvid i tur og orden med ett og ett nytt attributt.

Eksempel (lett)

$R(\mathit{Brnavn}, \mathit{Navn}, \mathit{Etternavn}, \mathit{Adresse}, \mathit{Kurskode}, \mathit{Tittel}, \mathit{Beskrivelse}, \mathit{AntSP}, \mathit{Karakter})$

- Brnavn $\rightarrow$ Navn, Etternavn, Adresse
- Kurskode $\rightarrow$ Tittel, Beskrivelse, AntSP
- Brnavn, Kurskode $\rightarrow$ Karakter

Eksempel (lett)

$R(\textit{Brnavn}, \textit{Navn}, \textit{Etternavn}, \textit{Adresse}, \textit{Kurskode}, \textit{Tittel}, \textit{Beskrivelse}, \textit{AntSP}, \textit{Karakter})$

- Brnavn $\rightarrow$ Navn, Etternavn, Adresse
- Kurskode $\rightarrow$ Tittel, Beskrivelse, AntSP
- Brnavn, Kurskode $\rightarrow$ Karakter

Attributter som ikke er på høyresider: Brnavn, Kurskode

Attributter som er i høyresider, men ikke venstre: Alle andre!

Ergo er $\{$ Brnavn, Kurskode $\}$ eneste kandidatnøkkel.

Eksempel (lett, ordentlig)

$R(\mathit{Brnavn}, \mathit{Navn}, \mathit{Etternavn}, \mathit{Adresse}, \mathit{Kurskode}, \mathit{Tittel}, \mathit{Beskrivelse}, \mathit{AntSP}, \mathit{Karakter})$

- Brnavn $\rightarrow$ Navn, Etternavn, Adresse
- Kurskode $\rightarrow$ Tittel, Beskrivelse, AntSP
- Brnavn, Kurskode $\rightarrow$ Karakter

Eksempel (lett, ordentlig)

$R(\mathit{Brnavn}, \mathit{Navn}, \mathit{Etternavn}, \mathit{Adresse}, \mathit{Kurskode}, \mathit{Tittel}, \mathit{Beskrivelse}, \mathit{AntSP}, \mathit{Karakter})$

- Brnavn $\rightarrow$ Navn, Etternavn, Adresse
- Kurskode $\rightarrow$ Tittel, Beskrivelse, AntSP
- Brnavn, Kurskode $\rightarrow$ Karakter

Attributter som ikke er på høyresider: Brnavn, Kurskode

Ta tillukning, sjekk at vi har alle attributter. Minimalitet har vi automatisk her.

Eksempel (ikke lett)

Gitt relasjonen $R(A, B, C, D, E, F)$ med FDene $AB \to DE$, $C \to A$, $BD \to E$, $AE \to BF$, finn alle nøkler.

Attributter som ikke forekommer i noen høyreside: C
Attributter som bare forekommer i høyresider: F

Begynn med C og utvid med A, B, D, E.

1. $X = C$. $C^+ = CA$. C er ikke en kandidatnøkkel.
2. Prøv å utvide X med B, D, E. (Siden A allerede er i C^+, er det ikke noe poeng å utvide C med A.)
   - 2.1 $X = BC$. $BC^+ = BCADEF$. BC er en kandidatnøkkel.
   - 2.2 $X = CD$. $CD^+ = CDA$. CD er ikke en kandidatnøkkel.
   - 2.3 $X = CE$. $CE^+ = BCADEF$. CE er en kandidatnøkkel.
   - 2.4 Fortsett med $X = CD$. Prøv å utvide med B, E.
     - 2.4.1 $X = BCD$. $BCD^+ = BCADEF$. Men BC er en kandidatnøkkel, så BCD er ikke minimal, og ikke kandidatnøkkel.
     - 2.4.2 $X = CDE$. $CDE^+ = BCADEF$. Men CE er en kandidatnøkkel, så CDE er ikke minimal, og ikke kandidatnøkkel.

Oppsummering så langt

Skjemaer som er dårlig designet inneholder anomalier som kan unngås.

Som regel skyldes dette at ikke-relatert informasjon er i samme tabell.

Vi skal definere normalformer som gjør at anomalier ikke forekommer.

For å gjøre det trenger vi FDer (kunnskap om hva som avhenger av hva).

FDer tillater oss å identifisere nøkler, som er unike identifikatorer for tupler.

Med dette kan vi definere normalformer.

Normalformene 1NF-BCNF

Normalformene vi skal se på danner et hierarki:

$$BCNF \subseteq 3NF \subseteq 2NF \subseteq 1NF$$

Det vil si: Hvis et skjema oppfyller 3NF, oppfyller det også 2NF og 1NF.

Høyere NF gir færre anomalier, men flere tabeller og flere joins.

Gitt et skjema, så finnes det en algoritme som lager et ekvivalent skjema på hvilken NF man ønsker.

Vi skal se på algoritmen for å lage BCNF, samt hvordan man sjekker hvilken NF et skjema er på.

For **alle** NF er det slik at et skjema oppfyller en gitt NF hvis alle tabeller oppfyller kravene.

1NF

En tabell er på 1NF hvis alle attributter er **atomære**.

Ingen egendefinerte lister/JSON/whatever i tabellfeltene.

Av og til ullent begrep (postgres støtter JSON som datatype).

Vi antar derfor at 1NF alltid er oppfylt.

1NF, eksempler på brudd

Tabell Student:

Brnavn	...	Veiledere
evgenit	...	[arild, martingi]
peternl	...	[abc]

Tabell Ansatt:

Brnavn	...	Studenter
arild	...	[evgenit]
abc	...	[peternl]

Nesten alltid lurere å lage en egen tabell Veiledning(Student, Veileder).

2NF

En tabell oppfyller 2NF hvis den oppfyller 1NF og hvis alle attributter *A* som ikke er nøkkelattributter, **ikke** er funksjonelt avhengige av en delmengde av en kandidatnøkkel.

Nøkkelattributt: Attributt som er med i en kandidatnøkkel.

Alternativt: En tabell **bryter** 2NF hvis det finnes et ikke-nøkkelattributt *A* som **er avhengig** av en delmengde av en kandidatnøkkel.

2NF, eksempel

$R(Brnavn, Navn, Etternavn, Adresse, Kurskode, Tittel, Beskrivelse, AntSP, Karakter)$

- Brnavn $\rightarrow$ Navn, Etternavn, Adresse
- Kurskode $\rightarrow$ Tittel, Beskrivelse, AntSP
- Brnavn, Kurskode $\rightarrow$ Karakter

Kandidatnøkkel: Brnavn, Kurskode.

2NF, eksempel

$R(Brnavn, Navn, Etternavn, Adresse, Kurskode, Tittel, Beskrivelse, AntSP, Karakter)$

- Brnavn $\rightarrow$ Navn, Etternavn, Adresse
- Kurskode $\rightarrow$ Tittel, Beskrivelse, AntSP
- Brnavn, Kurskode $\rightarrow$ Karakter

Kandidatnøkkel: Brnavn, Kurskode.

Navn er avhengig av Brnavn, Brnavn er en del av nøkkelen. Brudd på 2NF.

3NF

En tabell oppfyller 3NF hvis den oppfyller 2NF og alle ikke-nøkkelattributter **kun** er avhengige av kandidatnøkler.

Alternativt: En tabell bryter 3NF hvis det finnes et ikke-nøkkelattributt som **er avhengig** av noe som **ikke** er en kandidatnøkkel.

Hvordan kan et slikt brudd oppstå (uten å bryte 2NF)?

3NF, eksempel

Ansatt(Id, Navn, Avdeling, AvdelingsKode)

- Id $\rightarrow$ Navn, AvdelingsKode
- AvdelingsKode $\rightarrow$ Avdeling

Id er en kandidatnøkkel (transitivitet), men Avdeling er avhengig av AvdelingsKode, som ikke er en nøkkel.

Avdeling dobbeltlagres for hver ansatt (burde skilles ut i egen tabell).

BCNF

En tabell oppfyller BCNF hvis alle attributter kun er avhengige av en kandidatnøkkel.

Samme som 3NF, men unntaket for nøkkelattributter er **fjernet**.

Unntaket blir sjeldent brukt, og som regel er tabeller på 3NF også på BCNF.

BCNF

En tabell oppfyller BCNF hvis alle attributter kun er avhengige av en kandidatnøkkel.

Samme som 3NF, men unntaket for nøkkelattributter er **fjernet**.

Unntaket blir sjeldent brukt, og som regel er tabeller på 3NF også på BCNF.

Huskeregel for BCNF: The key, the whole key, and nothng but the key, (so help me Codd)

Hvordan sjekke normalformer

For å sjekke hvilken NF et skjema oppfyller, kan vi sjekke alle tabeller opp mot alle FDer.

FDene må skrives på formen $X \to A$ (splitt høyresidene), og man må passe seg for redundans.

Redundans: Hvis $X \to B$, så har vi også $X \cup Y \to B$ for alle Y.

Algoritme for å sjekke NF

Først, finn alle kandidatnøkler.

For hver tabell og hver FD $X \to A$:

1. Er X en supernøkkel? Hvis ja, BCNF sålangt, gå til neste FD. Hvis nei, brudd på BCNF.
2. Er A et nøkkelattributt? Hvis ja, 3NF sålangt, gå til neste FD. Hvis nei, brudd på 3NF.
3. Er X del av en kandidatnøkkel? Hvis nei, 2NF sålangt, gå til neste FD. Hvis ja, brudd på 2NF, stopp.

Tabellen er på den laveste normalformen vi får ut av denne algoritmen.

Skjemaet er på den laveste normalformen av tabellenes.

Med andre ord: Hvis jeg har en tabell og en FD som bryter 2NF, er skjemaet på 1NF.

Sjekke NF, eksempel

$$R(Brnavn, Navn, Etternavn, Kurskode, KursTittel, Karakter)$$

1. Brnavn, Kurskode $\rightarrow$ Karakter
2. Brnavn $\rightarrow$ Navn
3. Brnavn $\rightarrow$ Etternavn
4. Kurskode $\rightarrow$ KursTittel

Kandidatnøkkel: $\{$ Brnavn, Kurskode $\}$. Vi kan begynne med hvilken FD vi vil.

Sjekke NF, eksempel

$$R(Brnavn, Navn, Etternavn, Kurskode, KursTittel, Karakter)$$

1. Brnavn, Kurskode $\rightarrow$ Karakter
2. Brnavn $\rightarrow$ Navn
3. Brnavn $\rightarrow$ Etternavn
4. Kurskode $\rightarrow$ KursTittel

Kandidatnøkkel: $\{$ Brnavn, Kurskode $\}$. Vi kan begynne med hvilken FD vi vil.

FD 1: Brnavn, Kurskode er en supernøkkel, så BCNF sålangt.

Sjekke NF, eksempel

$$R(Brnavn, Navn, Etternavn, Kurskode, KursTittel, Karakter)$$

1. Brnavn, Kurskode $\rightarrow$ Karakter
2. Brnavn $\rightarrow$ Navn
3. Brnavn $\rightarrow$ Etternavn
4. Kurskode $\rightarrow$ KursTittel

Kandidatnøkkel: $\{$ Brnavn, Kurskode $\}$. Vi kan begynne med hvilken FD vi vil.

FD 1: Brnavn, Kurskode er en supernøkkel, så BCNF sålangt.

FD 2: Brnavn ikke supernøkkel, brudd på BCNF.

Sjekke NF, eksempel

$$R(Brnavn, Navn, Etternavn, Kurskode, KursTittel, Karakter)$$

1. Brnavn, Kurskode $\rightarrow$ Karakter
2. Brnavn $\rightarrow$ Navn
3. Brnavn $\rightarrow$ Etternavn
4. Kurskode $\rightarrow$ KursTittel

Kandidatnøkkel: $\{$ Brnavn, Kurskode $\}$. Vi kan begynne med hvilken FD vi vil.

FD 1: Brnavn, Kurskode er en supernøkkel, så BCNF sålangt.

FD 2: Brnavn ikke supernøkkel, brudd på BCNF.

FD 2: Navn ikke nøkkelattributt, brudd på 3NF.

Sjekke NF, eksempel

$$R(Brnavn, Navn, Etternavn, Kurskode, KursTittel, Karakter)$$

1. Brnavn, Kurskode $\rightarrow$ Karakter
2. Brnavn $\rightarrow$ Navn
3. Brnavn $\rightarrow$ Etternavn
4. Kurskode $\rightarrow$ KursTittel

Kandidatnøkkel: $\{$ Brnavn, Kurskode $\}$. Vi kan begynne med hvilken FD vi vil.

FD 1: Brnavn, Kurskode er en supernøkkel, så BCNF sålangt.

FD 2: Brnavn ikke supernøkkel, brudd på BCNF.

FD 2: Navn ikke nøkkelattributt, brudd på 3NF.

FD 2: Brnavn del av en kandidatnøkkel, brudd på 2NF.

Sjekke NF, eksempel

$$R(Brnavn, Navn, Etternavn, Kurskode, KursTittel, Karakter)$$

1. Brnavn, Kurskode $\rightarrow$ Karakter
2. Brnavn $\rightarrow$ Navn
3. Brnavn $\rightarrow$ Etternavn
4. Kurskode $\rightarrow$ KursTittel

Kandidatnøkkel: $\{$ Brnavn, Kurskode $\}$. Vi kan begynne med hvilken FD vi vil.

FD 1: Brnavn, Kurskode er en supernøkkel, så BCNF sålangt.

FD 2: Brnavn ikke supernøkkel, brudd på BCNF.

FD 2: Navn ikke nøkkelattributt, brudd på 3NF.

FD 2: Brnavn del av en kandidatnøkkel, brudd på 2NF.

Relasjonen er derfor på 1NF (som vi antar alltid er tilfelle).

Oppsummering sålangt

1NF-BCNF, basert på hvilke avhengigheter utenfor kandidatnøkler vi forbyr.

Algoritme for å sjekke NF.

Huskeregel “The key (1NF), the whole key (2NF), and nothing but the key (3NF/BCNF)” (og 1-3 har unntak for nøkkelattributter).

Hvordan oppnå BCNF?

Vår opprinnelige relasjon var på 1NF. Hvordan lage et nytt skjema på BCNF?

Problemet er, grovt sett, at data som ikke hører sammen er i samme tabell.

Kan løses ved å **dekomponere** tabellen til mindre tabeller.

Kan ikke dekomponere som vi vil — dekomposisjonen må være **tapsfri**.

Tapsfri dekomponering

La $R(X)$ være en relasjon. En dekomponering av R er en mengde nye relasjoner $\{S_1(Y_1), \ldots, S_n(Y_n)\}$ slik at

1. $Y_i \subseteq X$
2. $\bigcup_{i=1}^{n} Y_i = X$

En dekomponering er tapsfri hvis vi alltid kan ta en instans IR av R, projisere ned til instanser IS_i av S_i, og så rekonstruere IR via naturlig join.

Ikke tapsfri dekomponering

Opprinnelig tabell: Ansatte(AvdID, AvdNavn, AnsattID, Navn, Etternavn)

Dekomponert til:

- Avdeling(AvdID, AvdNavn)
- Ansatt(AnsattId, Navn, Etternavn)

Alle atttributter er med, men vi har mistet noe viktig.

Tapsfri dekomponering, eksempel

Brnavn	Navn	Etternavn	Adresse	Kurskode	Tittel	Beskrivelse	AntSP	Kara
evgenit	Evgenij	Thorstensen	Addr1	IN2090	Databaser	EnBeskr...	10	B
peternl	Petter	Nilsen	Addr2	IN2090	Databaser	EnBeskr...	10	A
evgenit	Evgenij	Thorstensen	Addr1	IN2080	Beregn...	Descr...	10	A

Tabell Student:

Brnavn	Navn	Etternavn	Adresse
evgenit	Evgenij	Thorstensen	Addr1
peternl	Petter	Nilsen	Addr2

Tabell Kurs:

Kurskode	Tittel	Beskrivelse	AntSP
IN2090	Databaser	EnBeskr...	10
IN2080	Beregn...	Descr...	10

Tabell Karakterer:

Brnavn	Kurskode	Karakter
evgenit	IN2090	B
peternl	IN2090	A
evgenit	IN2080	B

Alle attributter er med, og naturlig join gir opprinnelig tabell.

Hvordan garantere tapsfri dekomponering?

Fagins teorem: En dekomponering $R(X, Y), S(X, Z)$ er tapsfri hvis og bare hvis $X \rightarrow Y$.

Med andre ord, vi kan skille ut noen attributter og det de alle er avhengige av.

Dette gir opphav til dekomponeringsalgoritmen for BCNF.

Tapsfri dekomponering til BCNF

Gitt $R(X)$ og en mengde FDer F. Hvis $Y \to A \in F$ er et brudd på BCNF,

- beregn Y^+,
- og dekomponer R til $S_1(Y^+)$ og $S_2(Y, X/Y^+)$.

Fortsett rekursivt til det ikke lengre er noen brudd på BCNF.

Hvis en FD ender opp på tvers av flere tabeller, kan den ignoreres (skjer sjeldent).

Dekomponering, eksempel

La $R(A, B, C)$ ha FDer $F = \{A, B \to C, C \to A\}$.

Dekomponering, eksempel

La $R(A, B, C)$ ha FDer $F = \{A, B \to C, C \to A\}$.

Nøkkel: B forekomer ikke på høyresider, så B er med i alle nøkler. $\{A, B\}$ er en nøkkel, og $\{B, C\}$ er **også** en nøkkel.

Dekomponering, eksempel

La $R(A, B, C)$ ha FDer $F = \{A, B \to C, C \to A\}$.

Nøkkel: B forekomer ikke på høyresider, så B er med i alle nøkler. $\{A, B\}$ er en nøkkel, og $\{B, C\}$ er **også** en nøkkel.

$A, B \to C$ er ikke brudd på BCNF, siden A, B er en supernøkkel.

$C \to A$ er brudd på BCNF, men ikke på 3NF, siden A er et nøkkelattributt.

Dekomponering, eksempel

La $R(A, B, C)$ ha FDer $F = \{A, B \to C, C \to A\}$.

Nøkkel: B forekomer ikke på høyresider, så B er med i alle nøkler. $\{A, B\}$ er en nøkkel, og $\{B, C\}$ er **også** en nøkkel.

$A, B \to C$ er ikke brudd på BCNF, siden A, B er en supernøkkel.

$C \to A$ er brudd på BCNF, men ikke på 3NF, siden A er et nøkkelattributt.

Siden $C^+ = \{C, A\}$, dekomponerer vi til $S_1(A, C)$ og $S_2(B, C)$.

Dårlig dekomponering til BCNF

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

FDer:

- Brukernavn $\rightarrow$ Navn
- Brukernavn $\rightarrow$ Etternavn
- KursKode $\rightarrow$ KursNavn

Dårlig dekomponering til BCNF

$$R(Brukernavn, Navn, Etternavn, KursKode, KursNavn)$$

FDer:

- Brukernavn $\rightarrow$ Navn
- Brukernavn $\rightarrow$ Etternavn
- KursKode $\rightarrow$ KursNavn

Dekomp til:

- S(Brukernavn, Navn)
- T(Brukernavn, Etternavn)
- U(Brukernavn, KursKode)
- V(KursKode, KursNavn)

Tapsfri og BCNF, men vi har to tabeller som burde vært en. Husk å ta tillukning.

Oppsummering dekomponering

Tapsfri dekomponering via Fagins teorem: Ingen tap av koblinger.

Dekomponering til BCNF ved å skille ut attributter som bryter BCNF.

Kan gjøre dekomponering uten å ta tillukning, men det gir mange ekstra tabeller.

Hvordan designe til ca. BCNF?

Hovedprinsipp: Ett tuppel = en entitet eller en oppføring.

Relasjonstyper: En til en, en til mange, mange til mange.

En til en i samme tabell (på en nøkkel, kanskje kunstig).

En til mange via fremmednøkkel til ny tabell (Student og studieprogram)

Mange til mange (Studenter og kurs) via egen tabell.

Andre designtips

Vær rause med ant. tegn i strenger!

Tenk over nytten av historisk informasjon ved endringer.

Tidsstempling er et gode.

Validering av input er aldri så enkelt som man tror.

Færre begrensninger er ofte et gode.