732G16: Databaser; Design och programmering

Fö 4: Normalisering

2025

LINKÖPINGS UNIVERSITET

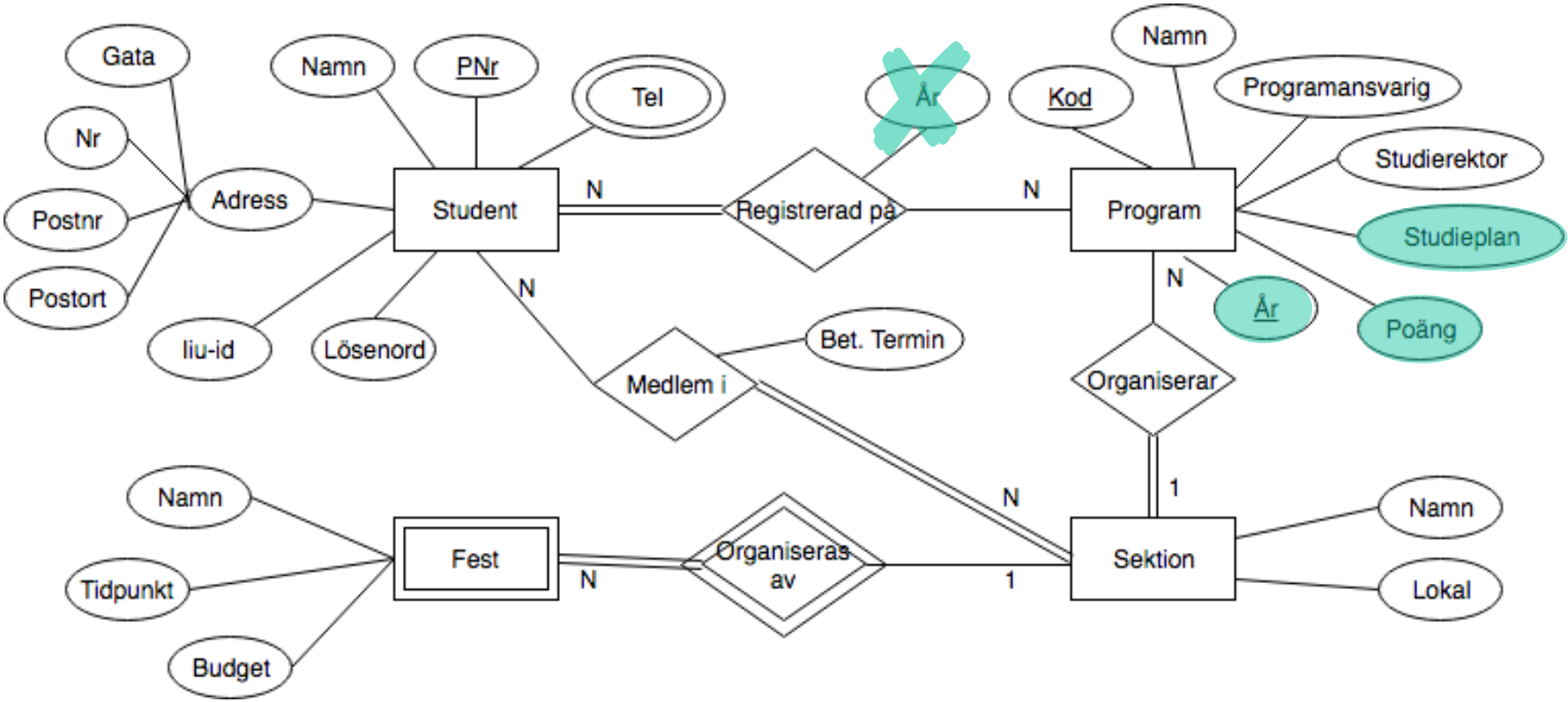
Innehåll

- Varför normalisera?
- Normalisering
  - Normalformer
  - Funktionella beroenden
  - Informationsbevarande relationsschemauppdelning

Varför normalisera?

- Metod att skydda oss från ”dum” design
  - Lagra samma data flera ggr i onödan
  - Inte kunna lagra viss information
  - Otydlig betydelse av en tupel/rad – som kan komma från otydlig betydelse av entitetsinstans

Studentförening - nu med studieplaner och poäng



Relationsmodell Studentförening

Student (PNr, Namn, Gata, GatNr, PostNr, PostOrt, liu-id, Lösen)

StudentTel (Student, Tel)

Program (Kod, Namn, Programansv, Studierektor, Sektion)

Sektion (Namn, Lokal)

Fest(Sektion, Namn, tidpunkt, budget)

RegistreradPå (Student, Program, År)

MedlemI (Student, Sektion, BetTermin)

Relationsmodell Studentförening

Student (PNr, Namn, Gata, GatNr, PostNr, PostOrt, liu-id, Lösen)

StudentTel (Student, Tel)

Program (Kod, Namn, PgmAnsv, Studierektor, Sektion, Poäng, Studieplan, År)

Sektion (Namn, Lokal)

Fest(Sektion, Namn, tidpunkt, budget)

RegistreradPå (Student, Program)

MedlemI (Student, Sektion, BetTermin)

Relationen Program (exempel)

Program

Kod	Namn	PgmAnsv	Studie-rektor	Sektion	Poäng	Studieplan	År
6cddd	Datatek	DMnämnden	-	D-sektion	500	Dokument 5	2016
f7kko	Kogvet	Mattias	Jalal	Krogvet	300	Dokument2	2017
f7ksa	Statistik	Linda	Lotta	StaLin	300	Dokument1	2017
f7ksa	Statistik	Linda	Lotta	StaLin	300	Dokument3	2016
f7ksa	Statistik	Kalle	Lotta	StaLin	300	Dokument 4	2009

Problem med Program

- Samma namn, poäng och sektion återkommer:
  - Tar plats.
  - Uppdateringsanomali - kan bli fel
  - Insättningsanomali - vissa data kan ej lagras
  - Borttagningsanomali - vissa data kan ej lagras
  - Otydlig tolkning
- Hur undvika redundans?

Normalisering

www.liu.se

LINKÖPINGS UNIVERSITET

Normalisering

- **Metod:** Undersök om relationen uppfyller normaliseringsvillkoren, i ordning
- Normaliseringsvillkor = **normalformer**
- Begrepp:
  - atomära attribut, supernycklar, kandidatnycklar, primärnycklar och nyckelattribut
  - funktionella beroenden, determinant
  - informationsbevarande relationsschemauppdelning

1:a Normalform (1NF)

Definition: Alla attribut ska vara **atomära**

- Atomär = odelbar

- Notera: Relationsmodellen antar att alla attribut är odelbara.

- 2:a normalform, 3:e normalform, BCNF baseras på **Funktionella beroenden** i relationen.

Funktionellt beroende

X är en delmängd av attributen i en relation.

- Om X entydigt bestämmer värdet på ett attribut Y, kallas det att ”Y är **funktionellt beroende** av X” eller att ”X bestämmer Y”.
- $X \Rightarrow Y$
- X kallas **determinant** i det funktionella beroendet.

Exempel

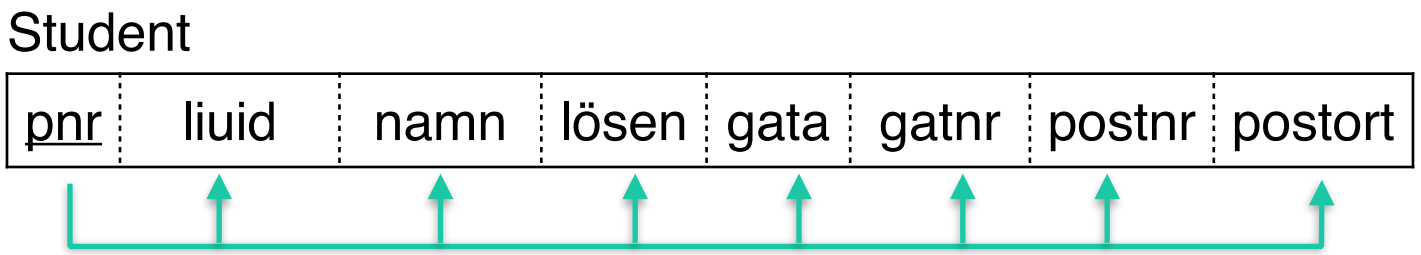
Ex: Student

{pnr}⇒namn, {pnr}⇒liuid, {pnr}⇒lösen, {pnr}⇒Gata, {pnr}⇒GatNr, {pnr}⇒PostNr, {pnr}⇒PostOrt, {pnr}⇒pnr

Kan också skrivas:

{pnr}⇒{pnr, namn, liuid, lösen, Gata, GatNr, PostNr, PostOrt}

Och ritas:



Funktionellt beroende, mer exempel

- Determinanten kan vara flera attribut
  ex Program:
  `{Kod,år}=>{Namn, PgmAnsv, Studierektor, Sektion, Poäng, Studieplan}`
- Determinanten får innehålla ”onödiga attribut”:
  ex Student:
  `{pnr, namn}⇒{pnr, namn, liuid, lösen, Gata, GatNr, PostNr, PostOrt}`
- Determinanten behöver inte vara en nyckel
  - I Student: `{PostNr} => {PostOrt}`

Funktionella beroenden, forts

- Funktionella beroenden baseras på relationens semantik. Ett fb ska gälla i alla möjliga instanser av databasen!

- Att det finns ett fb X ⇒ Y betyder **inte**

  att det finns ett fb Y ⇒ X

  – ex: pnr=>Namn betyder inte att Namn=>pnr

Funktionella beroenden kan härledas

- Transitivitet:
  - om $X \Rightarrow Y$ och $Y \Rightarrow Z$ så $X \Rightarrow Z$
- Reflexivitet:
  - $X \Rightarrow X$ alltid.
  - Om $Y \subseteq X$ så $X \Rightarrow Y$
- Augmentation:
  - Om $X \Rightarrow Y$ så har vi också $\{XZ\} \Rightarrow YZ$

Fullt funktionellt beroende

Givet: $X \Rightarrow Y$.

Om inget attribut kan tas bort ur X utan att vi förlorar det funktionella beroendet (dvs X är minimal), kallas det **fullt funktionellt beroende**, ffb.

Fullt funktionellt beroende, exemplet Program

Program

Pgmkod	Namn	PgmAnsv	Studierektor	Sektion	Poäng	Studieplan	År

- Nyckeln ger: {Kod, år} ⟹ {namn, PgmAnsv, Studierektor, Sektion, poäng, Studieplan}
- MEN: {kod, år} ⇒ {poäng} är **fb** men ej **ffb**
  - för ett program får inte ändra poäng hur som helst!
  - Likaså för {kod, år} => {Namn, Sektion} för ett program kan inte byta namn eller sektion.
- OBS att verkligheten styr!

Repetition: Nycklar

- Supernyckel
  - en attributmängd som identifierar hela tupeln/raden
- Kandidatnyckel
  - en minimal attributmängd som som identifierar hela tupeln
- Primärnyckel
  - den kandidatnyckel som väljs för att fungera som nyckel

Nyckelattribut

Attribut som ingår i **någon** kandidatnyckel till relationen.

- Exemplet Student:
  - pnr har ffb till alla attribut: **är nyckelattribut!**
  - **men** LiuID har det också, dvs
    - LiuID är också **kandidatnyckel!**
    - LiuID är också **nyckelattribut**

Nyckelattribut forts.

- Nyckelattribut markeras i relationsschemat med *
  - Student (pnr*, namn, liuid*, lösen, gata, gatnr …
  - Program (Kod*, Namn, PgmAnsv, Studierektor, Sektion, Studieplan, poäng, År*)

Student

pnr*	liuid*	namn	lösen	gata	gatnr	postnr	postort

Program

kod*	namn	PgmAnsv	Studierektor	Sektion	Poäng	Studieplan	År*

Andra normalformen - 2NF

Definition: En relation R är i 2NF

- **om** den är i 1NF
- **och** varje **icke-nyckelattribut** i R är **fullt funktionellt beroende** av varje **kandidatnyckel** i R.

Eller som fråga: Finns det vanliga attribut som beror av **en del av** en kandidatnyckel?

2NF Exemplet program

Program (Kod, Namn, PgmAnsv, Studierektor, Sektion, Poäng, Studieplan, År)

Kandidatnyckel: {Kod, år}

{Kod, år} ⇒ {Studierektor} FFB
{Kod, år} ⇒ {Programansv} FFB
{Kod, år} ⇒ {Studieplan} FFB
{Kod} ⇒ {namn} FFB
{Kod} ⇒ {poäng} FFB
{Kod} ⇒ {sektion} FFB

- Uppfyller **ej** 2NF

Relationsschemauppdelning

- Lyft ut det ”problematiska” funktionella beroendet till en egen tabell
  - ny tabell: determinant plus de attribut som bestäms
  - gamla tabellen: stryk de attribut som bestäms, lämna kvar determinanten.

- Hur vet vi att vi inte tappar bort något?

Informationsbevarande relationsschemauppdelning

Defintion:

- Om relationen R delas upp i R1 och R2 så att
- R1 * R2 = R
- så är uppdelningen informationsbevarande!

- Där * är Naturlig Sammansättning

Relationsalgebra:

- Naturlig sammansättning
- R * S
- Definition: givet två relationer R och S där ett (eller flera) attribut har samma namn (dvs samma domän, t.ex. a)
  - så är R * S $\subseteq_V R \times S$
  - där V är villkoret R.a = S.a

Informationsbevarande relationsschemauppdelning - inte

- Person(Pnr,Namn,Adress)
  med funktionella beroenden:
  Pnr ⇒ Namn, Pnr ⇒ Adress,
  inte Namn ⇒ Adress

pnr	namn	Adress
7908101234	Anna	Ågatan 3
8112237890	Anna	Rydsv.12

pnr	namn
7908101234	Anna
8112237890	Anna

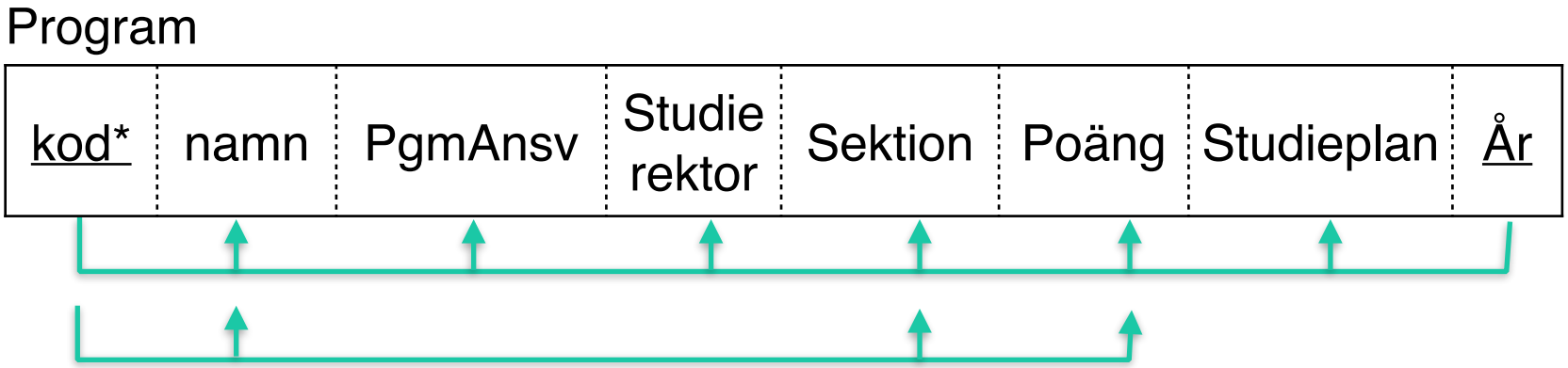
namn	Adress
Anna	Ågatan 3
Anna	Rydsv.12

Informationsbevarande relationsschemauppdelning

Tips:

- Bryt inte något fullt funktionellt beroende!
- Ett funktionellt beroende a=>x som finns i R: skapa en ny relation R1(a, x) och revidera den gamla R2=R-x
  - R1.a blir nyckel till R1
  - R2.a blir främmande nyckel till R1

Exemplet Program



Ger:

ProgramÅr (Kod, År, PgmAnsv, Studierektor, Studieplan)

ProgramAlltid (Kod, namn, Sektion, Poäng)

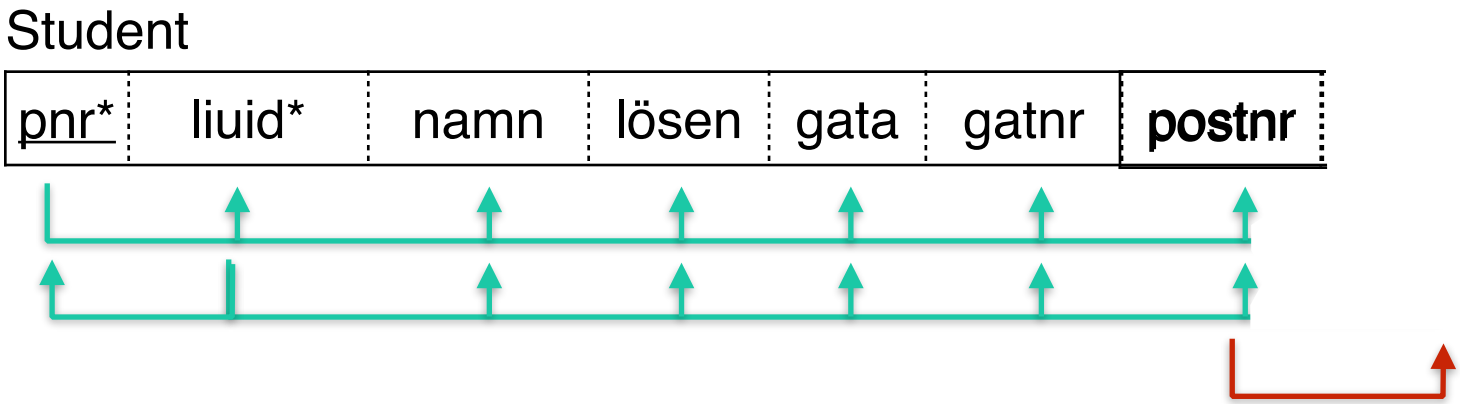
Tredje normalform - 3NF

Definition: En relation R är i 3NF

- **om** den är i 2NF
- **och** inget icke-nyckel-attribut är fullt funktionellt beroende (ffb) av något annat icke-nyckel-attribut.

- Som en fråga: finns det något vanligt attribut som är beroende av något annat vanligt attribut?

Är Student i 3NF?



- Nej.
  postnr => postort och postnr är INTE nyckelattribut!
- Dela upp: (postnr blir kvar i Student men ej postort)

Boyce-Codd normalform - BCNF

Definition: En relation R är i BCNF

- **om** det är i 3NF
- **och** alla determinanter är kandidatnycklar.

- Frågan för BCNF: är alla determinanter kandidatnycklar?

3NF/BCNF

- Inte alltid möjligt att transformera ett schema till BCNF och behålla beroendena.
- 3NF har de flesta av BCNF's fördelar.
- Det är inte självklart att man måste uppfylla BCNF.
- OBS: 3NF tillåter någon sorts redundans som BCNF inte gör (funktionella beroenden mellan nyckelattribut).

Resultat:

- Student' (pnr, liuid, namn, lösen, gata, postnr)
- PostAdress (PostNr, PostOrt)
- ProgramÅr (Kod, År, PgmAnsv, Studierektor, Studieplan)
- ProgramAlltid(Kod, namn, poäng, Sektion)

Frågor?

LINKÖPINGS UNIVERSITET