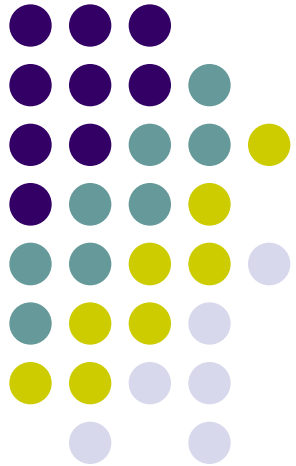
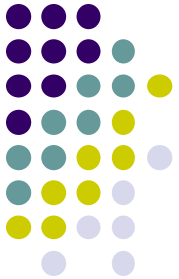
Databaser och databasdesign

Den relationella modellen, normalisering och modellering (2)





Varför databaser (DB)?

- Vi vill och måste kunna lagra data på sätt som motsvarar olika verksamheters behov

Vad är grundläggande?

- Grundläggande är att vi kan **lägga till**, **ändra** och **ta bort** data, och att lagrad data är korrekt

Spelar ”databasdesignen” nån roll?

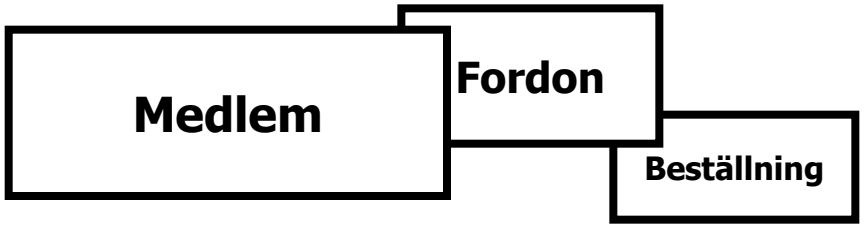
- Designen är kritisk för att en databas ska fungera som önskat; för att vi effektivt kunna söka, hitta och lista data**.**

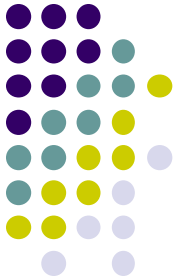
Databaser designas – vi har en designuppgift!

Modellering och design av relationsdatabaser bygger på:

- att vi korrekt analyserar vilka data som ska lagras
- att vi identifierar rätt *objekt* att lagra data om

Objekten kan beskrivas grafiskt med en rektangel och objektets namn. Namnet ska vara **beskrivande** och i singular:





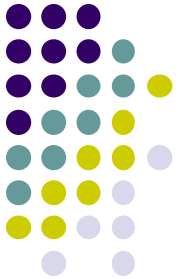
Objektsegenskaper

De *objekt* vi identifierar har *egenskaper*
(jfr. entiteter respektive attribut)

En analys kan leda till att vi tilldelar ett objekt **Medlem** egenskaper som:

- Namn
- Adress
- Telefon
- E-post

Relationer/samband (1)

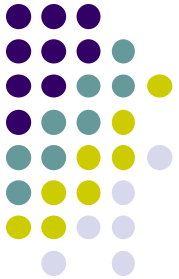


Objekten har *relationer* till varandra.

Vi kan identifiera relationer eller samband genom:

- **association** – t ex att *”Kund gör Beställning”* eller *”Kund lägger Order”*
- **struktur** – t ex att *”Order innehåller Artiklar”*

Relationer/samband (2)

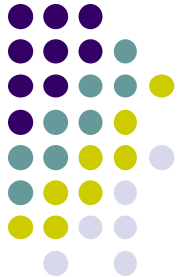


Vi kan behöva modellera relationer av typen *en-till-många* eller kort sagt **1:M**.

Ett exempel:

Om en leverantör levererar flera produkter råder en 1:M-relation mellan *Leverantör – Produkt*.

Utläses som: *En leverantör kan leverera många produkter*. Vi kan benämna relationen ”**Levererar”**



Relationer/samband (3)

Vi kan behöva modellera relationer av typen *många-till-många,* kort sagt **M:M**.

Ett exempel:

En student kan läsa flera kurser, och en kurs kan läsas av flera studenter. Det råder en M:M-relation mellan *Student – Kurs*.

Vi kan benämna relationen ”**Läser**”

På väg mot en E/R-graf…

Vi kan illustrera *objekt* och *relationer* med symboler:

E/R-grafen

En E/R-graf är en modell som visar alla viktiga objekt, objekts relationer och egenskaper (Se Chen notation).

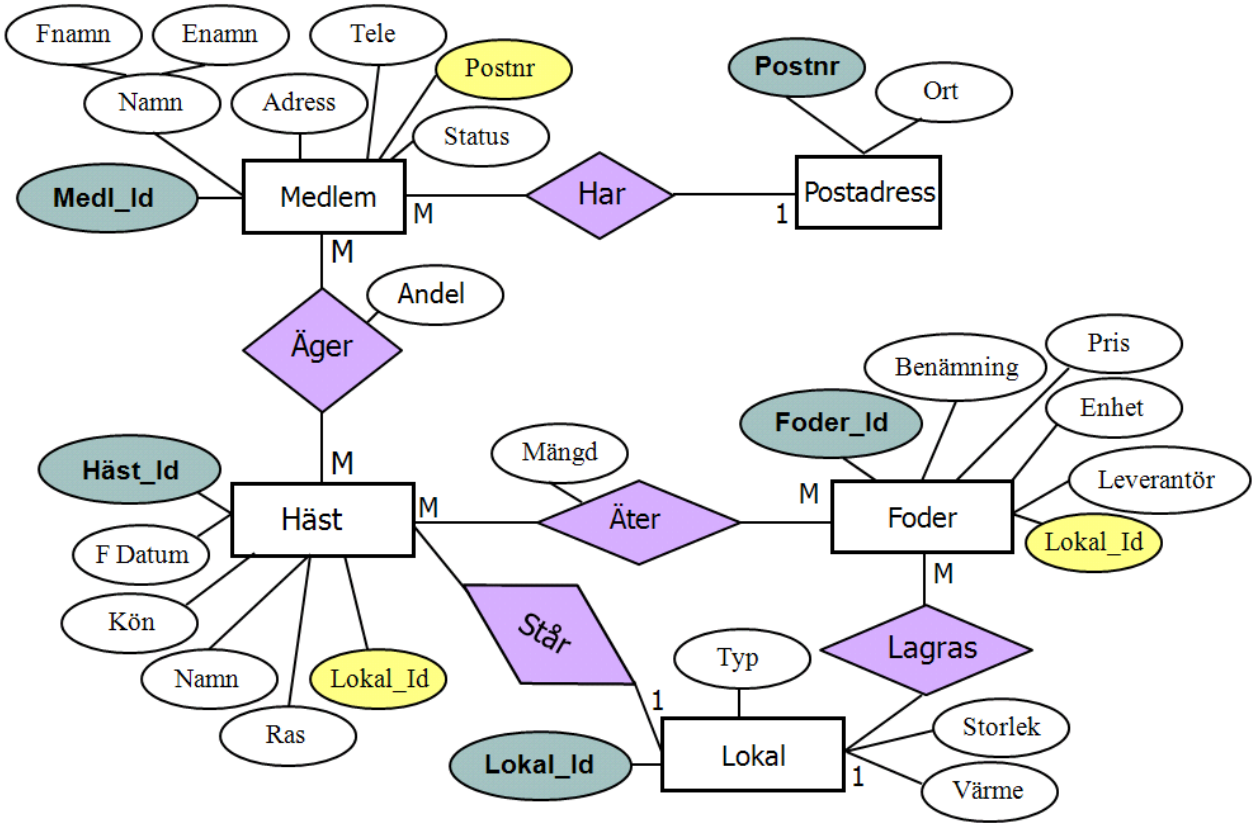
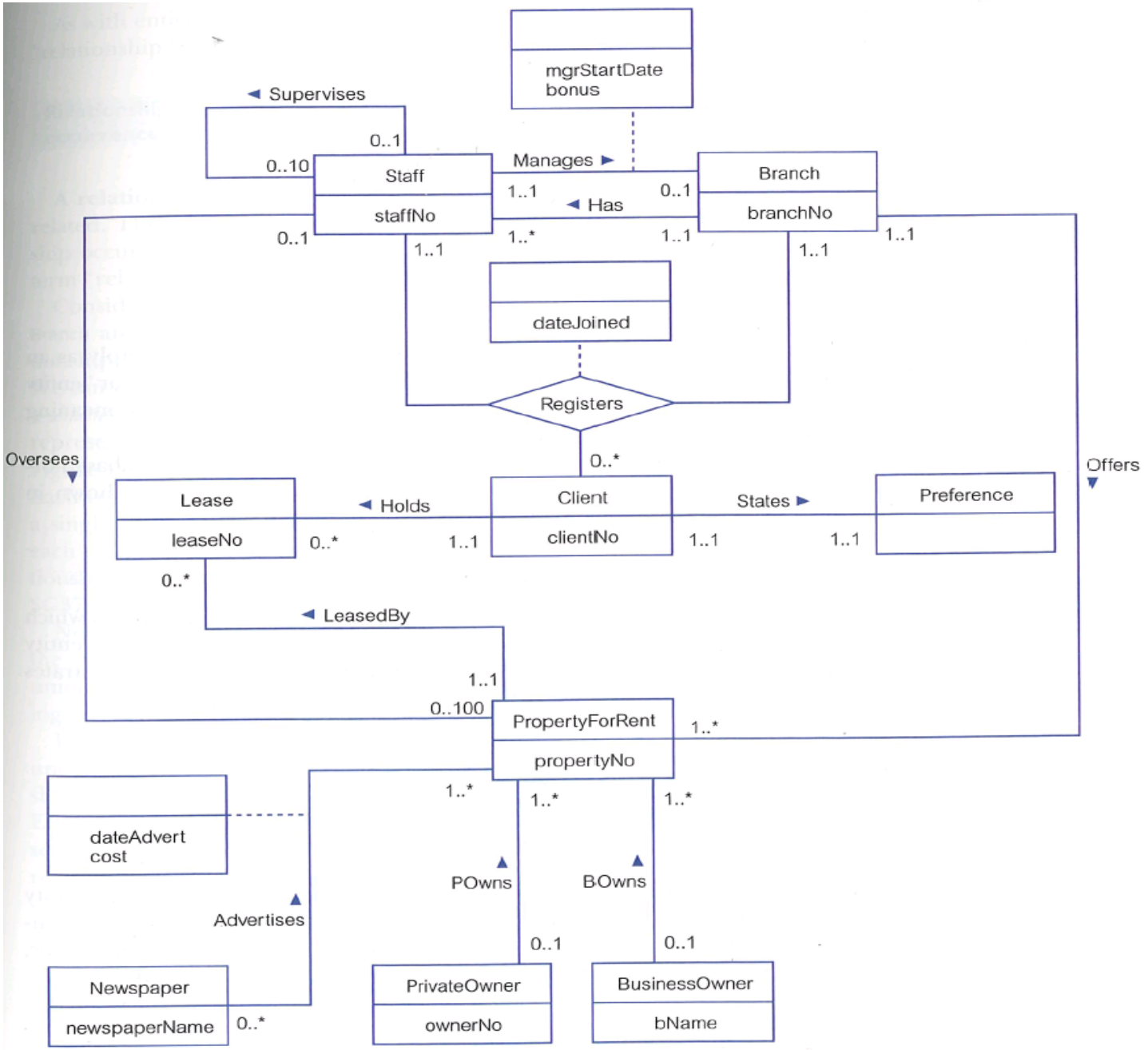
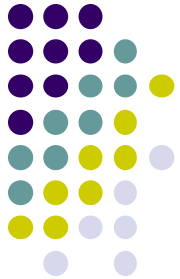


Figure 12.1 An Entity–Relationship (ER) diagram of the Branch view of *DreamHome*.



I relationsdatabasen knyts objekt (tabeller) ihop till en logisk helhet genom ett **nyckel-system** och **matchning av värden**



Primärnyckel (PN)
unik, identifierare

Främmande nyckel (FN) - referens till PN i en primärtabell

StudentID	Fnamn	Enamn	*ProgramID*
8201139901	Anna	Ek	VIP07-10
8611221113	Hanna	Ström	IT106-09
8403200278	Bo	Ahlberg	VIP07-10

PN

ProgramID	Programbeskrivning
VIP07-10	Programmet syftar till.....
IT106-09	Programmet syftar.....
WEB05-08	Programmet.....

1:M relation

”En leverantör kan leverera många (flera) produkter.”

Leverantör

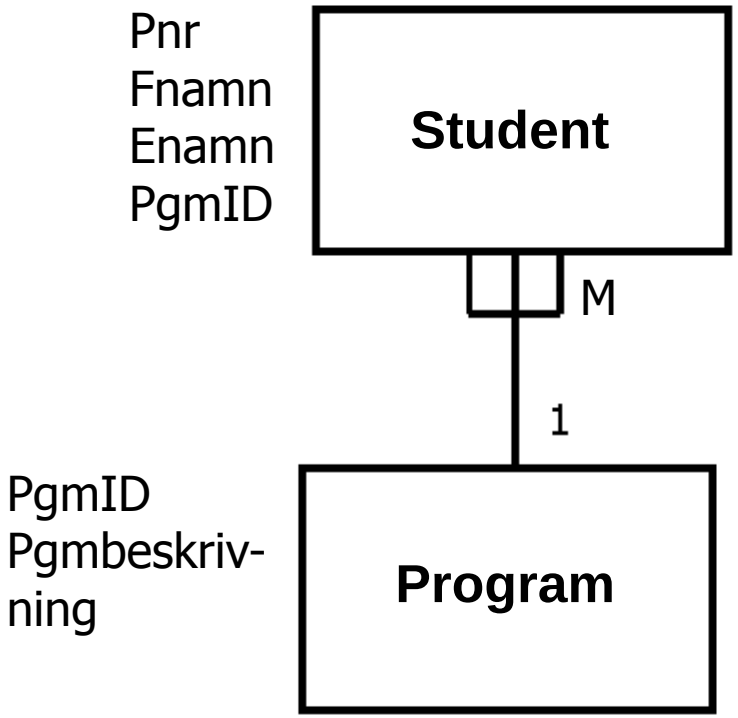
LevID	Leverantörsnamn	Adress	Tel	Postnr
1001	Stål & Rost AB	Sveavägen 1	629752	116 32
1002	Handelskompaniet	Brogatan 12	173210	305 44
1033	ICA	Flygstaden 1	700009	311 66

Produkt

ProduktID	Produktnamn	Mått	*LevID*
12-B-105	Fjäderbricka	6	1001
12-B-339	Bult	6	1001
10-A-991	Mutter	12	1002

Modelleringsregler – 1:M

I en 1:M-relation relation "lånar" objektet på M-sidan den identifierande egenskapen(**PgmId)** från objektet på 1-sidan, som blir en FN (referens) till primärtabellen



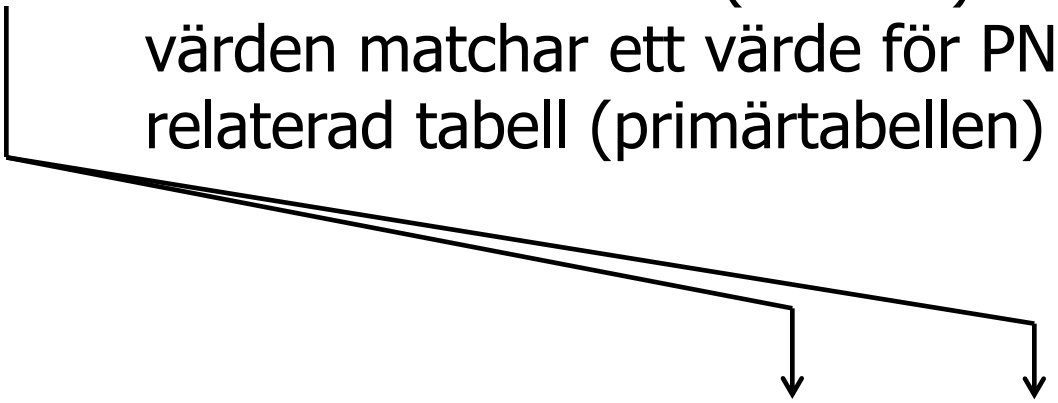
Student

Pnr	Fnamn	Enamn	*PgmId*
1001	Anna	Ek	1
1002	Hanna	Ström	**2**
1006	Bo	Al	**2**

Program

PgmId	Pgmbeskrivning
1	Programmet syftar till …
2	Programmet syftar …

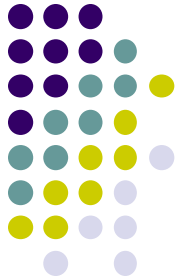
FN – ett eller fler attribut (kolumn) vars värden matchar ett värde för PN i relaterad tabell (primärtabellen)



StudentID	Fnamn	Enamn	Adress	*Postnr*	*ProgramID*
8201139901	Ann	Ek	Stigen 2	305 72	VIP07-10
8611221113	Eva	Ström	Ekvägen 1	119 34	IT106-09
8403200278	Bo	Ahlberg	Urgränd 9	305 72	VIP07-10

PN – identifierande attribut (kolumn) med unika värden

M:M relationen involverar ett relationsobjekt



Komposit/sammansatt PN {Ordernr, Artnr}

Orderspec

Ordernr	Artnr	Antal
1001	12-B-105	75
1001	12-B-339	50
1009	10-A-991	300

PN

Order

Ordernr	Datum	Kundnr
1001	110603	A-2137
1009	110722	B-4359

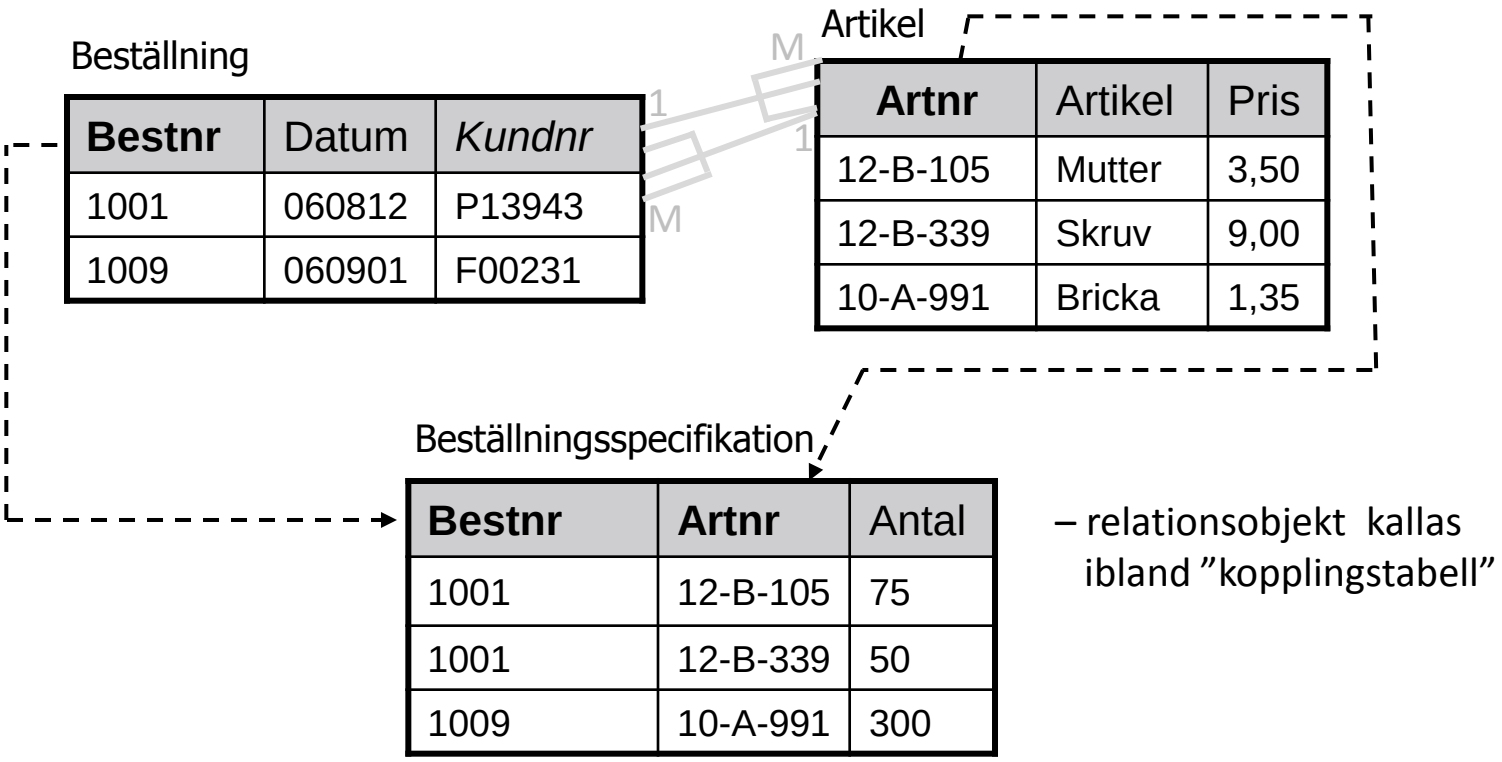
PN

Artikel

Artnr	Beskrivning	Pris
12-B-105	SonyPS3	3990
12-B-339	PS3CoD	549
10-A-991	PS3PoP	650

Modelleringsregler – M:M ex. (2)

En M:M-relation kan ses som två 1:M-relationer.
När en relation är M:M- skapar vi ett tredje objekt vars ”inlånade” nycklar blir PN **och** FN.



Funktionella beroenden (1)

Funktionellt beroende (FD) definition:

*Låt **R** vara en relation och **X** och **Y** attribut i **R**. Då kan vi säga att **Y** är funktionellt beroende av **X** om, och endast om, varje X-värde i R är associerat med precis ett Y-värde i **R**.*

R

X	Y
1001	A
1002	B

X är en determinant, bestämmer Y

Funktionellt beroende (2)

FD – ett attribut bestämmer ett annat

Tabellen **Kund** innehåller icke-nyckel attributen Kund, Adress Postnr och Ort. Ort är funktionellt beroende av Postnr – *värdet på postnummer bestämmer värdet för Ort; postnummer är* ***determinant****.*

Kund

KundNr	Kund	Adress	Postnr	Ort
1001	Willys	Storgatan 1	961 67 →	Boden
1002	Maxi	Grågränd 3	961 67 →	Boden
1003	Arla	Byholmen	961 98 →	Boden

Funktionellt beroende (3)

Vilka funktionella beroenden som råder mellan attribut i en tabell bestäms av verksamhetsregler
- *inte av vilka data som lagras vid ett visst tillfälle!*

Vi kan se är vilka FD:s som ***ev. kan finnas***; genom att de inte motsägs av de data som finns lagrade.

Vilka FD:s *kan* respektive *kan inte* finnas i tabellen nedan?

A	B	C	D
1	4	10	100
2	5	20	50
3	6	20	200
1	4	10	200
2	6	20	2
3	6	20	300
1	4	10	3
2	6	20	50
3	6	20	50

Det kan inte finnas ett FD: A → B, då det för ett och samma värde på A finns olika värden på B.

A	B	C	D
1	4	10	100
2	5	20	50
3	6	20	200
1	4	10	200
2	6	20	2
3	6	20	300
1	4	10	3
2	6	20	50
3	6	20	50

Det kan t ex finnas ett FD: A → C och B → C.

Fullständigt funktionellt beroende (FFD)

Attributet *Antal* i tabell Orderspec nedan är FFD av PN {OrderNr, ArtikelNr} – ***av hela PN inte av en del av PN***

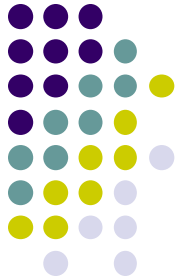
Det går inte att reducera PN (komposit), utan att PN förlorar sin unikt identifierande egenskap – OrderNr **och** ArtikelNr behövs för att fastställa hur många artiklar som ingår i en viss order

OrderNr	ArtikelNr	Antal
1001	101	2
1001	103	1
1001	105	3
1002	101	2

OrderNr# → Antal? (motsägs av data)

ArtikelNr# → Antal? (ej tidsoberoende)

{OrderNr#, ArtikelNr#} → Antal (gäller)

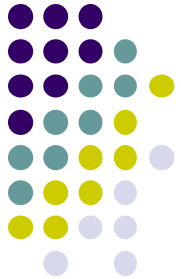


Normalisering (1)

Normalisering är att tillämpa ett antal regler för att uppnå vissa centrala designmål – en formell metod för att ge DB önskvärda egenskaper

Normalisering av en relation/tabell kan innebära att den delas upp i flera tabeller, med vissa gemensamma data (jfr. PN och FN)

Normalisering (2)

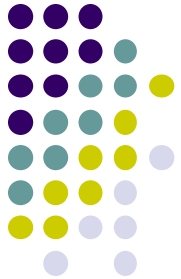


- formell grund för analys av DB utifrån ”nycklar” och funktionella beroenden mellan attribut

- konkreta riktlinjer för hur DB/tabeller bör designas för att undvika bl a **redundans** (onödig dubbel-lagring av data) och **uppdateringsproblem**

- ett antal tester som kan göras för att kontrollera om en DB uppfyller villkoren för en viss normalform

Normalisering (3)

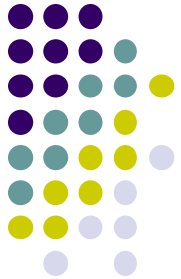


- fem normalformer: 1NF – 5 NF

- varje normalform har vissa villkor som måste uppfyllas

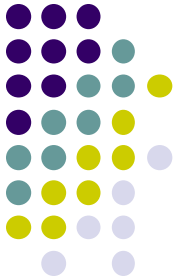
- en relation ***R*** sägs vara i en viss normalform om den uppfyller kraven som definierats för normalformen

Normalformer - **1NF**



- endast ett värde i varje rad/kolumnposition (fält)
- *tabellen ska ha en primärnyckel*

127763-A1	Karl	Larsson	035-173064, 070-353243	Haverdal



Normalformer - **2NF**

- Tabellen uppfyller kraven för 1NF plus…
- alla icke-nyckelattribut fullständigt funktionellt (FFD) beroende av **hela** primärnyckeln

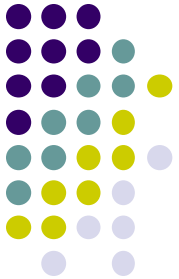
Normalformen gäller tabeller med komposit PN.

Om inte alla icke-nyckelattribut är FFD av hela PN, så skapa en ny tabell med de beroende attributen och den del av PN de bestäms av

Ex. 2 NF och normalisering

OrderNr	ArtikelNr	Antal	Beskrivning	Spec	Vikt
1001	101	2	Mutter	M6	20
1001	103	1	Skruv	M12	40
1001	105	3	Bricka	M12	3
1002	101	2	Mutter	M6	20

ArtikelNr	Beskrivning	Spec	Vikt
101	Mutter	M6	20
103	Skruv	M12	40
105	Bricka	M12	3
101	Mutter	M6	20



Normalformer - **3NF**

- Tabellen uppfyller kraven för 2NF plus…
- innehåller ej transitiva beroenden som t ex **Kundnr → Postnr → Ort.**

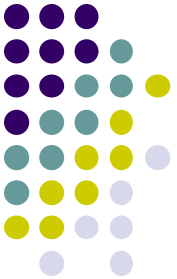
Relationer i 3NF får ej innehålla icke nyckel-attribut som är funktionellt beroende av varandra

KundNr	Kund	Adress	*Postnr*	Ort
1001	Willys	Storgatan 1	961 67	Boden
1002	Maxi	Grågränd 3	961 67	Boden

Postnr	Ort
961 67	Boden
961 98	Boden

För att en DB skall vara i 3NF, måste alla tabeller uppfylla kraven för den normalformen!

Varför normalisering?



Om en DB uppfyller villkoren för 3NF fungerar ofta grundläggande operationer (tillägg, uppdatering, borttagning på postnivå) utan problem – **vi blir även av med omotiverad redundans**

Normalisering underlättar uppdateringar, men ökar utsökningstiden (det omvända kan gälla för redundans)

Redundans och inkonsistens?

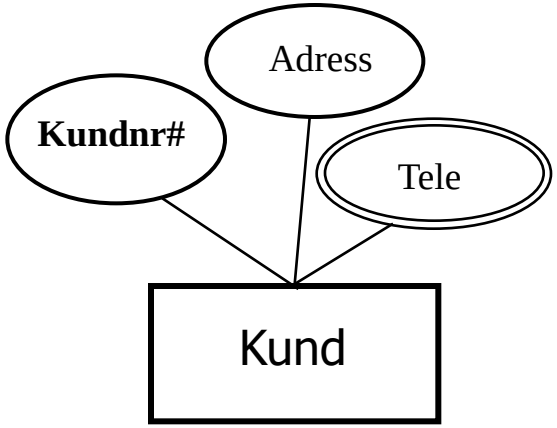
Förlustlös dekomposition?

En relation (tabell) som normaliserats måste kunna återskapas utan att data förloras; återskapas som den var innan normalisering

Ex. multivärda attribut…

(a)

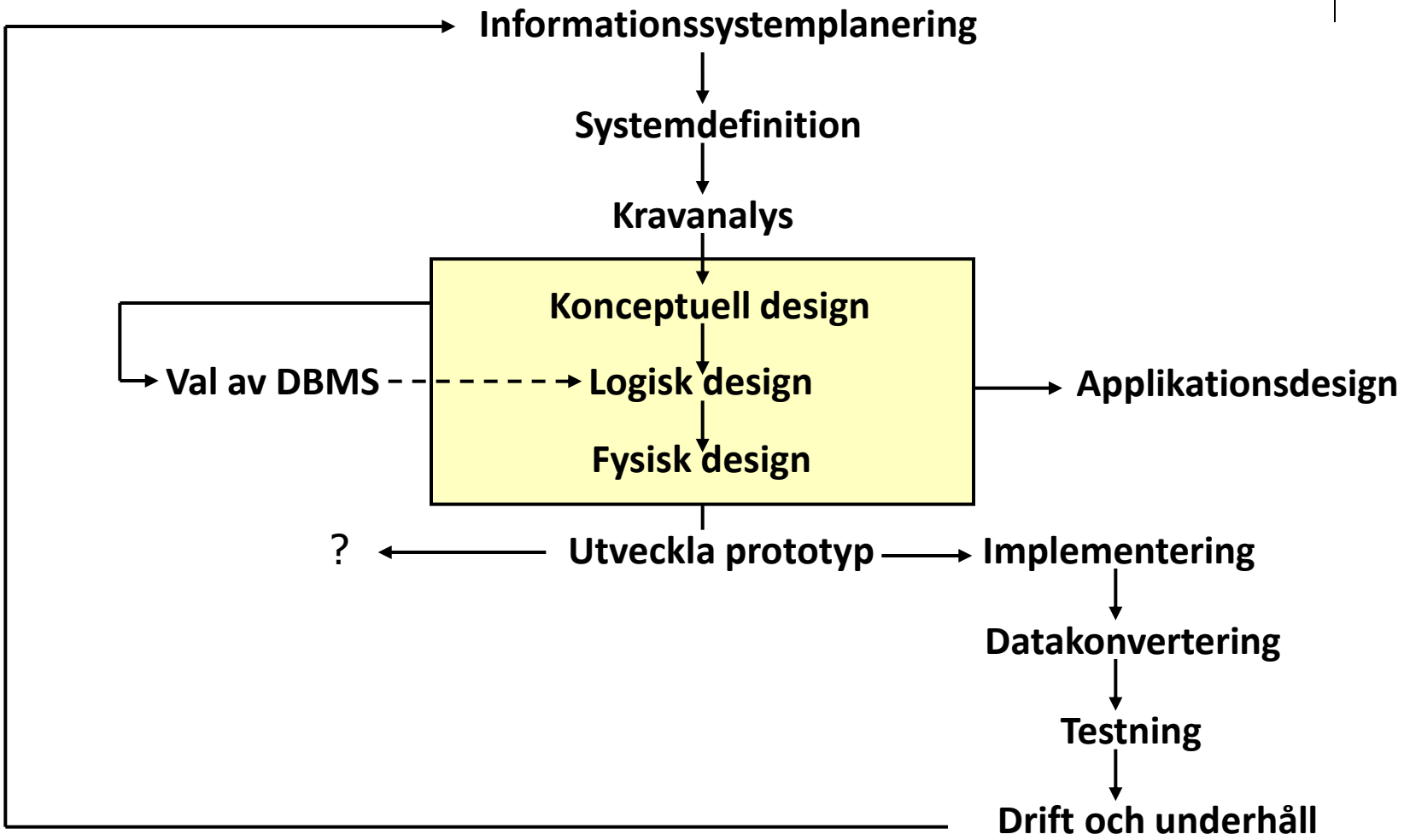
Kund
Kundnr {PN} Adress Tel [1-3]

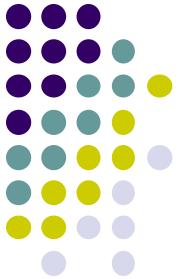


(b)

Kund	Har	**Kundtele**
Kundnr {PN} Adress	1 .. 1 — 1 .. 3	Kundtel {PN} Typ

Informationssystem (IS) – livscykel och databasdesign

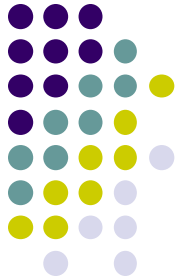




Konceptuell databasdesign?
Logisk databasdesign?
Fysisk databasdesign

(se C & B s.417, 436, 440-ff)

Framgångsfaktorer DB-design (ex.)



- arbeta interaktivt med användare

- välj ett datadrivet angreppssätt

- använd en strukturerad metod vid datamodellering

- inkludera eller bilägg struktur- och integritetsvillkor i modellen

- validera konceptualisering, normalisering, transaktioner

- komplettera modellen med DD (data dictionary)