

# *Raffinamento dello schema e forme normali*



# *Forme Normali*

- ❖ Le forme normali consentono di valutare la qualità delle relazioni
- ❖ Sono state proposte diverse forme normali che includono, in ordine di generalità:
  - Prima forma normale
  - Seconda forma normale
  - Forma normale di Boyce-Codd
  - Terza forma normale
- ❖ Il processo che consente di trasformare schemi in forma normale è detto di *normalizzazione*



## *I mali della ridondanza*

- ❖ La ridondanza è alla base di diversi problemi associati agli schemi relazionali:
  - memorizzazione ridondante, anomalie da inserimento/cancellazione/aggiornamento
- ❖ Vincoli di integrità, in particolare *dipendenze funzionali*, possono essere usate per identificare gli schemi con tali problemi e per suggerire raffinamenti
- ❖ Principale tecnica di raffinamento: *decomposizione* (sostituire ABCD con, diciamo, AB e BCD, oppure ACD e ABD)
- ❖ La decomposizione dovrebbe essere usata con giudizio:
  - C'è una ragione per decomporre una relazione?
  - Che problemi causa (se ne causa) una decomposizione?

# Dipendenze funzionali (DF)

- ❖ Una **dipendenza funzionale**  $X \rightarrow Y$  vale su una relazione  $R$  se, per ogni istanza ammissibile  $r$  di  $R$ :
  - $\forall t1 \in r, t2 \in r, \quad \Pi_x(t1) = \Pi_x(t2)$  implica  $\Pi_y(t1) = \Pi_y(t2)$  cioè, per ogni coppia di tuple in  $r$ , se i valori di  $X$  sono uguali, allora anche i valori di  $Y$  devono essere uguali ( $X$  e  $Y$  sono *insiemi* di attributi)
- ❖ Una DF è una affermazione su **tutte** le relazioni ammissibili
  - Deve essere identificata in base alla semantica dell'applicazione
  - Data una istanza ammissibile  $r1$  di  $R$ , possiamo controllare se essa viola qualche DF  $f$ , ma non possiamo dire se  $f$  vale su  $R$ !
- ❖ Dire che  $K$  è una chiave candidata per  $R$  significa che  $K \rightarrow E$ 
  - Però,  $K \rightarrow R$  non richiede che  $K$  sia *minimale*!



## *Dipendenze funzionali (DF)*

- ❖ Se  $K$  è una chiave candidata per  $R$ , esiste una dipendenza funzionale tra  $K$  ed un qualunque altro attributo di  $R$
- ❖ In altri termini le dipendenze funzionali *generalizzano* i vincoli di chiave
  - Però,  $K \rightarrow R$  non richiede che  $K$  sia *minimale*!

## Esempio: vincoli su un insieme di entità



- ❖ Consideriamo la relazione ottenuta da Imp\_Ad\_Ore
  - Imp\_Ad\_Ore(cf, nome, parcheggio, esp, paga\_oraria, ore\_lavorate)
- ❖ Notazione: denoteremo questo schema di relazione elencandone gli attributi: CNPEOL
  - Questo è realmente *l'insieme* di attributi {C, N, P, E, O, L}
  - A volte ci riferiremo a tutti gli attributi di una relazione usando il nome della relazione stessa (ad esempio, Imp\_Ad\_Ore per CNPEOL)
- ❖ Alcune DF su Imp\_Ad\_Ore:
  - *cf* è la chiave: C → CNPEOL
  - *esp* determina *paga\_oraria*: V → O



## Esempio (segue)

- ❖ Problemi dovuti a E -> O:
  - Anomalia da aggiornamento: possiamo cambiare O solo nella prima tupla di CNPEOL?
  - Anomalia da inserimento: che succede se vogliamo inserire un impiegato e non conosciamo la paga oraria per il suo livello?
  - Anomalia da aggiornamento: se cancelliamo tutti gli impiegati di livello 5 perdiamo le informazioni sulla paga per il livello 5!

Paghe

E	O
8	10
5	7

Imp\_A\_Ore2

C	N	P	E	L
123-22-3666	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

C	N	P	E	O	L
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

# Ragionando sulle DF

- ❖ Date alcune DF, possiamo in genere inferire DF aggiuntive:
  - Sia lo schema  $Lavoratori(\underline{cf}, nome, pp, rid, dal)$
  - $cf \rightarrow rid, rid \rightarrow pp$  implica  $cf \rightarrow pp$
- ❖ Una DF  $f$  è implicata da un insieme di DF  $F$  se  $f$  vale ogni volta che valgono le DF in  $F$ 
  - $F^+$  = chiusura di  $F$  è l'insieme di tutte le DF che sono implicate da  $F$
- ❖ Assiomi di Armstrong ( $X, Y, Z$  sono insiemi di attributi):
  - Riflessività: Se  $X \subseteq Y$ , allora  $X \rightarrow Y$
  - Aumento: Se  $X \rightarrow Y$ , allora  $XZ \rightarrow YZ$  per ogni  $Z$
  - Transitività: Se  $X \rightarrow Y$  e  $Y \rightarrow Z$ , allora  $X \rightarrow Z$
- ❖ Queste sono regole di inferenza corrette e complete per le DF

## *Ragionando sulle DF (segue)*

- ❖ Coppia di regole addizionali (che seguono dagli Assiomi di Armstrong)
  - Unione: Se  $X \rightarrow Y$  e  $X \rightarrow Z$ , allora  $X \rightarrow YZ$
  - Decomposizione: Se  $X \rightarrow YZ$ , allora  $X \rightarrow Y$  e  $X \rightarrow Z$
- ❖ Esempio: Contratti(cid, fid, gid, did, pid, qta, val), e
  - C è la chiave:  $C \rightarrow CFGDPQV$
  - Un progetto acquista ciascun pezzo usando un singolo contratto:  $GP \rightarrow C$
  - Un reparto acquista al più un pezzo da un fornitore:  $FD \rightarrow P$
- ❖  $GP \rightarrow C$ ,  $C \rightarrow CFGDPQV$  implica  $GP \rightarrow CFGDPQV$
- ❖  $FD \rightarrow P$  implica  $FDG \rightarrow GP$
- ❖  $FDG \rightarrow GP$ ,  $GP \rightarrow CFGDPQV$  implica  $FDG \rightarrow CFGDPQV$



## *Ragionando sulle DF (segue)*

- ❖ Calcolare la chiusura di un insieme di DF può essere costoso (la dimensione della chiusura è esponenziale nel numero di attributi!)
- ❖ Tipicamente, vogliamo solo controllare se una data DF  $X \rightarrow Y$  è nella chiusura di un insieme di DF  $F$ . Un controllo efficiente:
  - calcolare la chiusura dell'attributo  $X$  (indicato con  $X^+$ ) rispetto  $F$ :
    - ◆ insieme di tutti gli attributi  $A$  tali che  $X \rightarrow A$  sia in  $F^+$
    - ◆ c'è un algoritmo in tempo lineare che lo calcola
  - Controllare se  $Y$  è in  $X^+$
- ❖  $F = \{A \rightarrow B, B \rightarrow C, CD \rightarrow E\}$  implica  $A \rightarrow E$ ?
  - Cioè,  $A \rightarrow E$  è nella chiusura  $F^+$ ? In maniera equivalente, è  $E$  in  $A^+$ ?



# *Forme normali*

- ❖ Ritornando al raffinamento degli schemi, la prima domanda da porsi è se un raffinamento sia necessario
- ❖ Se una relazione è in una certa forma normale (BCNF, 3NF, etc) si sa che certi tipi di problemi sono evitati/minimizzati. Ciò può aiutarci a decidere se decomporre la relazione sia utile.
- ❖ Ruolo delle DF nel rilevare la ridondanza:
  - Consideriamo una relazione R con 3 attributi, ABC.
    - ◆ Non ci sono DF: non c'è ridondanza
    - ◆ È data  $A \rightarrow B$ : diverse tuple potrebbero avere lo stesso valore A, e se ciò si verifica, esse avranno tutte lo stesso valore B!



# Forma normale di Boyce-Codd (BCNF)

- ❖ La relazione  $R$  con DF  $F$  è in BCNF se, per tutte le  $X \rightarrow A$  in  $F^+$ 
  - $A \in X$  (chiamata DF banale) oppure
  - $X$  contiene una chiave di  $R$
- ❖ In altre parole,  $R$  è in BCNF se le sole DF non banali che valgono su  $R$  sono vincoli di chiave
  - Non ci sono dipendenze in  $R$  che possano essere previste usando solo le DF
  - Se ci vengono mostrate due tuple con dei valori uguali, non possiamo inferire il valore  $A$  in una tupla dal valore  $A$  dell'altra
  - Se la relazione di esempio è in BCNF, le 2 tuple devono essere identiche (poiché  $X$  è una chiave)

X	Y	A
x	y1	a
x	y2	?




## Terza Forma Normale (3NF)

- ❖ Una relazione  $R$  con DF  $F$  è in 3NF se, per tutte le  $X \rightarrow A$  in  $F^+$ 
  - $A \in X$  (chiamata DF banale) oppure
  - $X$  contiene una chiave di  $R$  oppure
  - $A$  è parte di qualche chiave di  $R$
- ❖ La *minimalità* di una chiave è cruciale nella terza condizione
- ❖ Se  $R$  è in BCNF, ovviamente è in 3NF
- ❖ Se  $R$  è in 3NF, qualche ridondanza è possibile. È un compromesso, usato quando la BCNF non è ottenibile (ad esempio, non c'è una decomposizione "buona", oppure per considerazioni relative alle prestazioni)
  - Una decomposizione di  $R$  in una collezione di relazioni 3NF *senza-perdita* (che conservi le dipendenze) è sempre possibile

# Decomposizione di uno schema di relazione

- ❖ Supponiamo che la relazione R contenga gli attributi  $A_1 \dots A_n$ . Una *decomposizione* di R consiste nella sostituzione di R con due o più relazioni tali che:
  - Ciascun nuovo schema di relazione contiene un sottoinsieme degli attributi di R (e nessun attributo che non appartiene a R), e
  - Ogni attributo di R appare come attributo di una delle nuove relazioni
- ❖ Intuitivamente, decomporre R significa che memorizzeremo istanze degli schemi di relazione prodotti dalla decomposizione, invece delle istanze di R
- ❖ Ad esempio, possiamo decomporre CNPEOL in CNPEL e EO



## *Decomposizione di esempio*

- ❖ Le decomposizioni dovrebbero essere usate solo quando necessario
  - CNPEOL ha DF  $C \rightarrow CNPEOL$  e  $E \rightarrow O$
  - La seconda DF causa la violazione della 3NF; valori  $O$  ripetutamente associati a valori di  $E$ . Il modo più facile per risolverla è creare una relazione  $EO$  per memorizzare queste associazioni, e rimuovere  $O$  dallo schema principale:
    - ◆ cioè, decomporre CNPEOL in CNPEL e EO

# Decomposizioni senza-perdita

- ❖ La decomposizione di  $R$  in  $X$  e  $Y$  è *senza-perdita* rispetto a un insieme di DF  $F$  se, per ogni istanza  $r$  che soddisfa  $F$ 
  - $\Pi_X(r) \bowtie \Pi_Y(r) = r$
- ❖ È sempre vero che  $r \subseteq \Pi_X(r) \bowtie \Pi_Y(r)$ 
  - In generale, il contrario non è vero! Se lo è, la decomposizione è *senza-perdita*
- ❖ La definizione si estende facilmente a decomposizioni in 3 o più relazioni
- ❖ È essenziale che tutte le decomposizioni usate per risolvere la ridondanza siano *senza-perdita*! (Si evita il problema (2))



## Ancora sulla proprietà senza-perdita

- ❖ La decomposizione di  $R$  in  $X$  e  $Y$  è *senza-perdita* rispetto  $F$  se e solo se la chiusura di  $F$  contiene
  - $X \cap Y \rightarrow X$ , oppure
  - $X \cap Y \rightarrow Y$
- ❖ In particolare:
  - Se  $X \cap Y = \emptyset$
  - Se  $X \rightarrow Y$  vale su  $R$
  - la decomposizione di  $R$  in  $XY$  e  $R - VY$  è *senza-perdita*

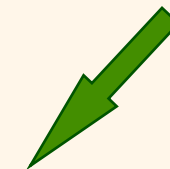
A	B	C
1	2	3
4	5	6
7	2	8



A	B
1	2
4	5
7	2

B	C
2	3
5	6
2	8

A	B	C
1	2	3
4	5	6
7	2	8
1	2	8
7	2	3



# Decomposizione in BCNF

- ❖ Consideriamo la relazione R con DF F. Se  $X \rightarrow Y$  viola la BCNF, si decompone R in R - Y e XY
  - Applicazioni ripetute di questa idea ci danno una collezione di relazioni che sono in BCNF; sono *senza-perdita*, e con garanzia di un termine
  - Ad esempio, CFGDPQV, chiave C,  $GP \rightarrow C$ ,  $FD \rightarrow P$ ,  $G \rightarrow F$
  - Per risolvere  $FD \rightarrow P$ , decomponiamo in FDP, CFGDQV
  - Per risolvere  $G \rightarrow F$ , decomponiamo CFGDQV in GF e CGDQV
- ❖ In generale, diverse dipendenze possono causare la violazione della BCNF. L'ordine in cui le analizziamo può portare a insiemi di relazioni completamente diversi!

# Raffinare uno schema ER

Prima:

- ❖ Prima traduzione dello schema ER:

Lavoratori(C, N, P, R, D)

Reparto(R, B)

- Parcheggi associati ai lavoratori

- ❖ Supponiamo che a tutti i lavoratori di un reparto sia assegnato lo stesso parcheggio:  $R \rightarrow P$

Ridondanza; risolta da:

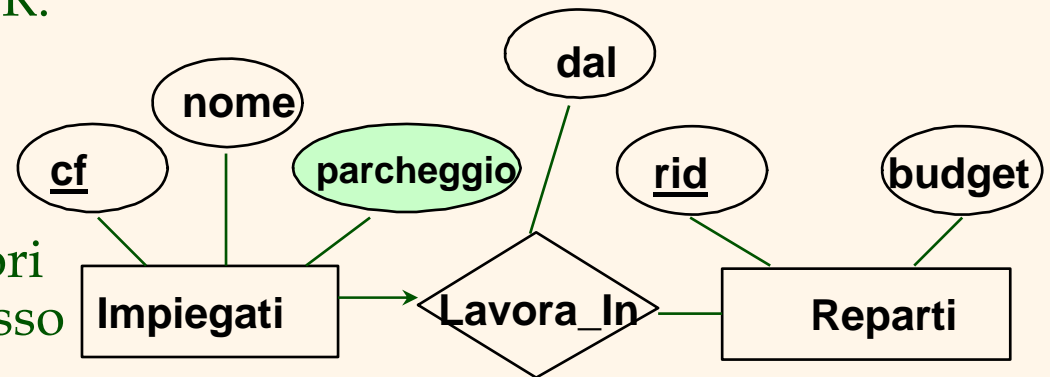
Lavoratori2(C, N, R, D)

Parcheggi\_Rep(R, P)

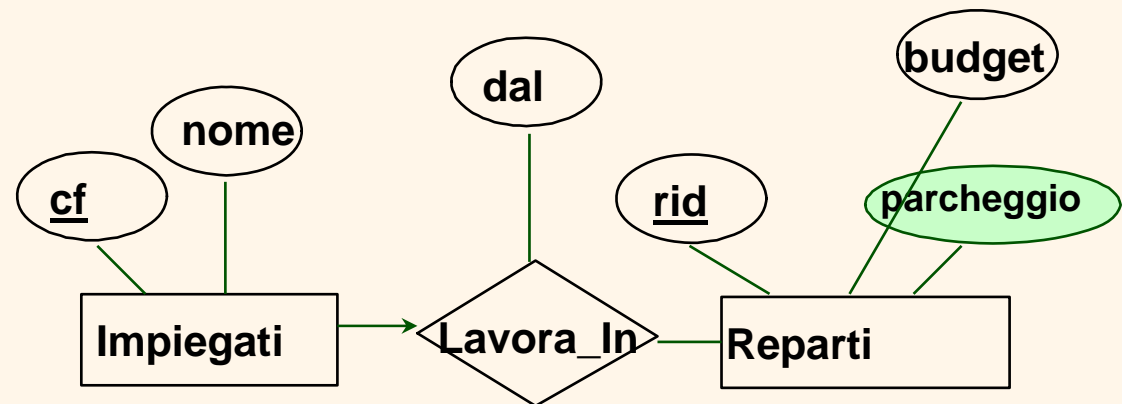
- ❖ Miglioramento:

Lavoratori2(C, N, R, D)

Reparti(R, B, P)



Dopo:





## *Riassunto del raffinamento di schema*

- ❖ Se una relazione è in BCNF, è priva di ridondanze che possono essere determinate usando le DF. Quindi, cercare di garantire che tutte le relazioni siano in BCNF è una buona euristica
- ❖ Se una relazione non è in BCNF, possiamo provare a decomporla in una collezione di relazioni BCNF
  - Dobbiamo considerare se tutte le DF sono conservate. Se una decomposizione in BCNF *senza-perdita* che conserva le dipendenze non è possibile (o non adatta, date le interrogazioni tipiche), dovremmo considerare una decomposizione in 3NF
  - Le decomposizioni dovrebbero essere eseguite e/o riesaminate tenendo in conto i *requisiti sulle prestazioni*