Basi di Dati
Algebra relazionale
- seconda parte -

Outline

- Interrogazioni con negazione
- Interrogazioni con quantificazione universale
- Operatore quoziente
- Semantica di Codd del valore nullo e logica a tre valori
- Join interni e join esterni
- Proprietà degli operatori algebrici

Base di dati «Ricoveri»

pazienti

COD	Cognome	Nome	Residenza	AnnoNascita
A102	Necchi	Luca	TO	1950
B372	Rossigni	Piero	NO	1940
B543	Missoni	Nadia	TO	1960
B444	Missoni	Luigi	VC	2000
S555	Rossetti	Gino	AT	2010

reparti

COD	Nome-Rep	Primario
A	Chirurgia	203
B	Pediatria	574
C	Medicina	530
L	Lab-Analisi	530
R	Radiologia	405

ricoveri

PAZ	Inizio	Fine	Reparto
A102	2/05/2014	9/05/2014	A
A102	2/12/2004	2/01/2005	A
S555	5/10/2014	3/12/2014	B
B444	1/12/2004	2/01/2005	B
S555	6/09/2015	1/11/2015	A

medici

MATR	Cognome	Nome	Residenza	Reparto
203	Neri	Piero	AL	A
574	Bisi	Mario	MI	B
461	Bargio	Sergio	TO	B
530	Belli	Nicola	TO	C
405	Mizzi	Nicola	AT	R
501	Monti	Mario	VC	A

Base di dati «Ricoveri»

- Schema relazionale con vincoli di integrità referenziale

PAZIENTI(<u>COD</u>, Cognome, Nome, Residenza, AnnoNascita)

MEDICI(<u>MATR</u>, Cognome, Nome, Residenza, Reparto)

REPARTI(<u>COD</u>, Nome-Rep, Primario)

RICOVERI(<u>PAZ, Inizio</u>, Fine, Reparto)

Base di Dati "Impiegati"

impiegati

MATR	Cognome	Nome	Età	Stipendio
203	Neri	Piero	50	40
574	Bisi	Mario	60	60
461	Bargio	Sergio	30	61
530	Belli	Nicola	40	38
405	Mizzi	Nicola	55	60
501	Monti	Mario	25	35

organigramma

Capo	Impiegato
203	405
203	501
574	203
574	530
405	461

IMPIEGATI(<u>MATR</u>, Cognome, Nome, Età, Stipendio)

ORGANIGRAMMA(<u>Capo,Impiegato</u>)

~~Interrogazioni con negazione~~

~~Vediamo i seguenti esempi~~

- ~~Esempio 1: elencare i pazienti **non residenti** a Torino~~
- ~~Esempio 2: elencare i medici **non primari**~~

Interrogazioni con negazione

Esempio 1: elencare i pazienti **non residenti** a Torino

Si tratta di un'interrogazione in cui la negazione non è essenziale (cioè è *inessenziale*)

- Il contesto del sistema informativo mi suggerisce che è possibile risolvere l'interrogazione con un'interrogazione affermativa, ad esempio:

Elencare i pazienti con residenza **diversa** da Torino

Negazione non essenziale

"elencare i pazienti **non residenti** a Torino"

=

"elencare i pazienti con residenza **diversa** da Torino"

$$\sigma_{\text{Residenza} \neq \textit{'Torino'}}(\text{pazienti})$$

Interrogazioni con negazione

Esempio 2: elencare i medici **non primari**

Non riesco a scrivere un'espressione algebrica con appropriati confronti che risolva questa interrogazione

Si tratta di un'interrogazione con negazione *essenziale*, non possiamo risolverla in maniera affermativa

Premessa

- Nelle basi di dati si lavora **assumendo un «mondo chiuso»** (*Closed-World Assumption*)
- Ovvero assumiamo che i fatti contenuti nella base di dati siano una descrizione **completa** del mondo di interesse, cioè **i soli fatti veri del mondo sono quelli presenti nella base di dati**
- Perciò, se un fatto **non è descritto** nella base di dati, è **falso**. Per es. nell'ospedale non esistono medici al di fuori di quelli inclusi nella relazione *medici*
- Quindi posso cercare i medici non primari per complemento **usando la relazione *medici***

Negazione essenziale

Elencare i medici **non primari**

Schema generale di soluzione:

1. Si definisce l'universo del discorso *U* (qui si sta sfruttando l'assunzione di mondo chiuso)
   – Nel nostro esempio: *U := medici*

Negazione essenziale

Elencare i medici **non primari**

Schema generale di soluzione:

1. Si definisce l'universo del discorso *U* (qui si sta sfruttando l'assunzione di mondo chiuso)
   - Nel nostro esempio: *U := medici*
2. Si risponde alla domanda in forma positiva *P*
   - Nel nostro esempio: *elencare i medici **primari***

   $P := \pi_{MATR,Cognome,Nome,Residenza,Reparto}(medici \bowtie_{MATR=Primario} reparti)$

Negazione essenziale

Elencare i medici **non primari**

Schema generale di soluzione:

1. Si definisce l'universo del discorso *U* (qui si sta sfruttando l'assunzione di mondo chiuso)
   - Nel nostro esempio: $U := medici$
2. Si risponde alla domanda in forma positiva *P*
   - Nel nostro esempio: *elencare i medici* ***primari***

     $P := \pi_{MATR,Cognome,Nome,Residenza,Reparto}(medici \bowtie_{MATR=Primario} reparti)$
3. Si trova la risposta all'interrogazione originale con il complemento

   $R := U - P$
   - Nel nostro esempio:

$$medici - \pi_{MATR,Cognome,Nome,Residenza,Reparto}(medici \bowtie_{MATR=Primario} reparti)$$

Requisito

Gli schemi di *U* e *P* devono essere **uguali** per essere compatibili con l'operatore di differenza

Alternativa

Lavoro sulle matricole e ottengo solo alla fine le altre informazioni dei medici

1. $U := \pi_{MATR}(medici)$

2. $P := \pi_{MATR}(medici \bowtie_{MATR=Primario} reparti)$

3. $R := U - P =$
   $\pi_{MATR}(medici) - \pi_{MATR}(medici \bowtie_{MATR=Primario} reparti)$

4. $R \bowtie medici$
   - *per ottenere le informazioni dei medici*

Esempio

Elencare i pazienti **non residenti** in città in cui risiede qualche medico

Esempio

Elencare i pazienti **non residenti** in città in cui risiede qualche medico

1. *(tutti i pazienti)*

   $U := pazienti$

2. *(pazienti che risiedono in città in cui risiede un medico)*

   $P := \pi_{COD,PAZIENTI.Cognome,PAZIENTI.Nome,PAZIENTI.Residenza,AnnoNascita}$
   $(pazienti \bowtie_{PAZIENTI.Residenza=MEDICI.Residenza} medici)$

3. *(pazienti non residenti in città in cui risiede un medico)*

   $R := U - P =$

   $pazienti - \pi_{COD,PAZIENTI.Cognome,PAZIENTI.Nome,PAZIENTI.Residenza,AnnoNascita}$
   $(pazienti \bowtie_{PAZIENTI.Residenza=MEDICI.Residenza} medici)$

Esercizio

Elencare i reparti in cui non avvengono ricoveri

Esercizio

Elencare i reparti in cui non avvengono ricoveri

1. *(tutti i reparti)*

   $U := \pi_{COD,Nome\text{-}Rep}(reparti)$

2. *(reparti in cui ci sono ricoveri)*

   $P := \pi_{COD,Nome\text{-}Rep}(reparti \bowtie_{REPARTI.COD=RICOVERI.Reparto} ricoveri)$

3. *(reparti in cui non avvengono ricoveri)*

   $R := U - P =$

   $\pi_{COD,Nome\text{-}Rep}(reparti) -$

   $\pi_{COD,Nome\text{-}Rep}(reparti \bowtie_{REPARTI.COD=RICOVERI.Reparto} ricoveri)$

Generalità dell'approccio

Esempio 1: elencare i pazienti **non residenti** a Torino (negazione inessenziale)

Posso comunque usare la procedura studiata:

1. $U := pazienti$
2. $P := \sigma_{Residenza='Torino'}(pazienti)$
3. $R := U - P = pazienti - \sigma_{Residenza='Torino'}(pazienti)$

È però una query inutilmente complessa rispetto a $\sigma_{Residenza \neq 'Torino'}(pazienti)$!

es. comp

Esempio più complesso

Elencare i pazienti di Torino mai curati dal primario 203

Esempio più complesso

Elencare i pazienti di Torino mai curati dal primario 203

1. *(pazienti di Torino) U := $\sigma_{Residenza='Torino'}$(pazienti)*

Esempio più complesso

Elencare i pazienti di Torino mai curati dal primario 203

1. *(pazienti di Torino)* $U := \sigma_{Residenza='Torino'}(pazienti)$

2. *(pazienti curati dal primario 203)*
   $P := \pi_{PAZIENTI.COD,Cognome,Nome,Residenza,AnnoNascita}$
   $(pazienti \bowtie_{COD=PAZ} ricoveri \bowtie_{Reparto=REPARTI.COD} \sigma_{Primario='203'}(reparti))$

Esempio più complesso

Elencare i pazienti di Torino mai curati dal primario 203

1. *(pazienti di Torino)* $U := \sigma_{Residenza='Torino'}(pazienti)$

2. *(pazienti curati dal primario 203)*
   $P := \pi_{PAZIENTI.COD,Cognome,Nome,Residenza,AnnoNascita}$
   $(pazienti \bowtie_{COD=PAZ} ricoveri \bowtie_{Reparto=REPARTI.COD} \sigma_{Primario='203'}(reparti))$

3. *(pazienti di Torino mai curati dal primario 203)*
   $R := U - P = \sigma_{Residenza='Torino'}(pazienti) -$
   $\pi_{PAZIENTI.COD,Cognome,Nome,Residenza,AnnoNascita}$
   $(pazienti \bowtie_{COD=PAZ} ricoveri \bowtie_{Reparto=REPARTI.COD} \sigma_{Primario='203'}(reparti))$

ambigua

Ambiguità nelle interrogazioni

«Elencare i pazienti di Torino mai curati dal primario 203»

può essere letta come:

«Elencare i pazienti di Torino *ricoverati* ma mai presi in cura dal primario 203»

Cambia l'universo del discorso

$$U := \pi_{PAZIENTI.COD,Cognome,Nome,Residenza,AnnoNascita}(\sigma_{Residenza='Torino'}(pazienti) \bowtie_{COD=PAZ} ricoveri)$$

~~Esempio (negazione essenziale nascosta)~~

~~Elencare i pazienti con un solo ricovero~~

neg nascosta

Esempio (negazione essenziale nascosta)

«Elencare i pazienti con un solo ricovero»

equivalente a:

«Elencare i pazienti ricoverati almeno una volta e che non hanno avuto due o più ricoveri»

Esempio (negazione essenziale nascosta)

Elencare i pazienti ricoverati almeno una volta ma che non hanno avuto due o più ricoveri

1. $U := \pi_{PAZ}(ricoveri)$
2. $P := \pi_{RICOVERI1.PAZ}\,(\rho_{RICOVERI1 \leftarrow RICOVERI}(ricoveri)$
   $\bowtie_{\mathrm{RICOVERI1.PAZ=RICOVERI2.PAZ} \wedge \mathrm{RICOVERI1.Inizio} \neq \mathrm{RICOVERI2.Inizio}}$
   $\rho_{RICOVERI2 \leftarrow RICOVERI}(ricoveri))$
   – *interrogazione già vista*
3. $R := U - P = \ldots$

Esempio (negazione essenziale nascosta)

Elencare i pazienti con un solo ricovero

$$\pi_{PAZ}(ricoveri) - \pi_{RICOVERI1.PAZ}(\rho_{RICOVERI1 \leftarrow RICOVERI}(ricoveri)$$

$$\bowtie_{\mathrm{RICOVERI1.PAZ=RICOVERI2.PAZ} \wedge \mathrm{RICOVERI1.Inizio \neq RICOVERI2.Inizio}}$$

$$\rho_{RICOVERI2 \leftarrow RICOVERI}(ricoveri))$$

Esempio: Calcolo del massimo

Consideriamo una relazione *r(A)* composta da un unico attributo di tipo intero *NUM*

<u>**NUM**</u>
3
5
2

Pur non usando operatori aggregati, in algebra relazionale è possibile estrarre il massimo!

Esempio: Calcolo del massimo

Qui la forma con negazione essenziale è ancora più nascosta

Possiamo però ricordare che il massimo di un insieme di valori è il valore che è maggiore o uguale di *tutti* gli altri → non posso → non esiste valore maggiore

Purtroppo non posso esprimere direttamente questa definizione in algebra relazionale

Calcolare il massimo

So però decidere se un valore risulta essere minore di un altro.

Calcolo il prodotto cartesiano di r con se stesso ricavando l'insieme di tutte le coppie:

$$\rho_{NUM1 \leftarrow NUM}(r) \times \rho_{NUM2 \leftarrow NUM}(r)$$

NUM1	NUM2
3	3
3	5
3	2
5	3
5	5
5	2
2	3
2	5
2	2

Calcolare il massimo

So però decidere se un valore risulta essere minore di un altro.

Calcolo il prodotto cartesiano di r con se stesso ricavando l'insieme di tutte le coppie:

$$\rho_{NUM1 \leftarrow NUM}(r) \times \rho_{NUM2 \leftarrow NUM}(r)$$

NUM1	NUM2
3	3
3	5
3	2
5	3
5	5
5	2
2	3
2	5
2	2

Calcolare il massimo

So però decidere se un numero risulta essere minore di un altro.

Seleziono le tuple per cui *NUM1 < NUM2*:

$$\sigma_{NUM1<NUM2}(\rho_{NUM1 \leftarrow NUM}(r) \times \rho_{NUM2 \leftarrow NUM}(r))$$

NUM1	NUM2
3	5
2	3
2	5

Cos'è la proiezione su NUM1?

$$\pi_{NUM1}(\sigma_{NUM1<NUM2}(\rho_{NUM1 \leftarrow NUM}(r) \times \rho_{NUM2 \leftarrow NUM}(r)))$$

Calcolare il massimo

Cos'è la proiezione su NUM1?

$$\pi_{NUM1}(\sigma_{NUM1<NUM2}(\rho_{NUM1 \leftarrow NUM}(r) \times \rho_{NUM2 \leftarrow NUM}(r)))$$

È l'elenco dei valori (2 e 3) ognuno dei quali risulta essere minore di **qualche** altro numero.

NUM1
3
2

Calcolare il massimo

Quindi se chiamo

- U l'insieme di tutti i valori:
  $U := \rho_{NUM1 \leftarrow NUM}(r)$
- P i valori che sono minori di qualche valore:
  $P := \pi_{NUM1}(\sigma_{NUM1<NUM2}(\rho_{NUM1 \leftarrow NUM}(r) \times \rho_{NUM2 \leftarrow NUM}(r)))$

$R := U - P$ è l'elenco dei valori **che non sono minori** di nessun valore, cioè dei valori maggiori o uguali a tutti!

NUM1
5

Interrogazioni con quantificazione universale

«L’elenco dei valori maggiori o uguali a **tutti gli altri**» è un esempio di **quantificazione universale**, come tutti i casi in cui troviamo le parole "**tutti**", "**ogni**", "**sempre**" nell’interrogazione.

L’algebra relazionale non gestisce direttamente – come invece fa il calcolo relazionale – la quantificazione universale, ma…

Quantificazione universale

…possiamo ricondurci alla quantificazione esistenziale ricordandoci che la quantificazione universale è la negazione di una quantificazione esistenziale, cioè

$$\{t \mid \forall t'(p(t,t'))\} \equiv \{t \mid \nexists t'(\neg p(t,t'))\}$$

Ad es.: un valore t è «maggiore o uguale a tutti i valori t'» se e solo se «non esiste un valore t' di cui è minore».

es

Esempio

Consideriamo questo esempio

E = esami, P = piano di studi

E

MATR	Corso
2	Programmazione
3	Algebra
2	Basi di dati
3	Programmazione
2	Algebra

P

Corso
Programmazione
Basi di dati
Algebra

Elencare gli studenti che hanno superato ***tutti i corsi*** *previsti dal piano di studi*

Procedimento di soluzione

*Elencare gli studenti che hanno superato **tutti i corsi** previsti dal piano di studi*

1. Individuo l'universo di riferimento

$$U := \pi_{MATR}(E)$$

MATR
2
3

Procedimento di soluzione

*Elencare gli studenti che hanno superato **tutti i corsi** previsti dal piano di studi*

1. Individuo l'universo di riferimento

$$U := \pi_{MATR}(E)$$

2. Ricavo tutte le combinazioni possibili di matricola con i corsi previsti:

$$\pi_{MATR}(E) \times P$$

MATR	Corso
2	Programmazione
2	Basi di dati
2	Algebra
3	Programmazione
3	Basi di dati
3	Algebra

Procedimento di soluzione

2. Ricavo tutte le combinazioni possibili di matricola con i corsi previsti:

$$\pi_{MATR}(E) \times P$$

3. Trovo la differenza con *E* (hanno lo stesso schema)

$$(\pi_{MATR}(E) \times P) - E$$

MATR	Corso
2	Programmazione
2	Basi di dati
2	Algebra
3	Programmazione
3	Basi di dati
3	Algebra

–

MATR	Corso
2	Programmazione
3	Algebra
2	Basi di dati
3	Programmazione
2	Algebra

=

MATR	Corso
3	Basi di dati

Ho trovato l'elenco degli esami non superati

Procedimento di soluzione

4. Proietto su *MATR*

$$\pi_{MATR}((\pi_{MATR}(E) \times P) - E)$$

MATR
3

Ho ottenuto l'elenco delle matricole che non hanno ancora superato **qualche** esame

Ora riconsidero l'universo di partenza $U := \pi_{MATR}(E)$

Procedimento di soluzione

5. Sottraggo il risultato dall'universo di partenza

$$\pi_{MATR}(E) - \pi_{MATR}((\pi_{MATR}(E) \times P) - E)$$

MATR
2

Ho finalmente l'elenco di studenti che hanno superato **tutti** i corsi!

Non è altro che la definizione dell'**operatore quoziente**

Operatore quoziente o divisione

Date due relazioni r(A,B) e s(B) con *A* e *B* disgiunti, l'operatore **quoziente o divisione *r(A,B) ÷ s(B)*** produce una relazione *u(A)* che contiene le tuple che in *r(A,B)* compaiono in combinazione con **ogni tupla in s(B)**

Il risultato di *r(A,B) ÷ s(B)* ha schema A

Nell'esempio precedente abbiamo calcolato
E(Matr, Corso) ÷ P(Corso)

Operatore derivato quoziente

Si definisce come:

$$r(A,B) \div s(B) := \pi_A(r) - \pi_A((\pi_A(r) \times s) - r)$$

$\pi_A(r) \times s$	ha schema (A,B) e dà tutte le combinazioni possibili
$(\pi_A(r) \times s) - r$	ricava le combinazioni *non* presenti in r
$\pi_A(r) - \pi_A((\pi_A(r) \times s) - r)$	dà tutte le *altre* tuple di r, cioè quelle che compaiono in ogni combinazione

La cardinalità del quoziente è

$$0 \leq |r(A,B) \div s(B)| \leq |\pi_A(r)| \leq |r|$$

Il quoziente non ha un corrispettivo diretto in SQL, ma si può esprimere traducendo la definizione relazionale oppure usando sottoquery correlate

Esempio

Elencare i pazienti che sono stati ricoverati **in ogni** reparto

Esempio

Elencare i pazienti che sono stati ricoverati **in ogni** reparto

Applichiamo il quoziente:

$$r(A,B) \div s(B)$$

dove

- $r(A,B) = \pi_{PAZ,Reparto}(ricoveri)$
- $S(B) = \pi_{Reparto}(ricoveri)$

cioè: $\pi_{PAZ,Reparto}(ricoveri) \div \pi_{Reparto}(ricoveri)$

Esempio

Elencare i pazienti che sono stati ricoverati **in ogni** reparto

Espandiamo il quoziente:

$$r(A,B) \div s(B) = \pi_A(r) - \pi_A((\pi_A(r) \times s) - r)$$

$$\pi_{PAZ,Reparto}(ricoveri) \div \pi_{Reparto}(ricoveri) =$$
$$\pi_{PAZ}(\pi_{PAZ,Reparto}(ricoveri)) -$$
$$\pi_{PAZ}((\pi_{PAZ}(\pi_{PAZ,Reparto}(ricoveri)) \times \pi_{Reparto}(ricoveri)) -$$
$$\pi_{PAZ,Reparto}(ricoveri))$$

Esercizio

La soluzione precedente ha il problema di considerare solo i reparti in cui è già stato ricoverato qualcuno. Potrei pensare di ricavare i reparti dalla relazione *reparti.*

Se avessi scelto come relazione *s* la relazione

$$\rho_{\text{Reparto} \leftarrow \text{COD}}(\pi_{COD}(reparti))$$

e quindi, invece di $\pi_{PAZ,Reparto}(ricoveri) \div \pi_{Reparto}(ricoveri)$, avessi calcolato

$$\pi_{PAZ,Reparto}(ricoveri) \div \rho_{\text{Reparto} \leftarrow \text{COD}}(\pi_{COD}(reparti))$$

nell'ipotesi che ci siano reparti non adibiti a ricoveri, quale sarebbe stata la risposta del quoziente?

Sarebbe equivalente alla prima formulazione?

Semantica di Codd del valore nullo

Abbiamo fin qui assunto che non compaiano valori nulli. Ora invece prendiamo in considerazione questo caso.

Un valore nullo è un valore specifico che può appartenere a ogni dominio.

Semantica di Codd del valore nullo

Vari significati:

1. Informazione esistente, ma *non nota* (ad es. data di nascita)
2. Informazione *inesistente* (ad es. data di morte per un vivo)
3. *Assenza* di informazione (ad es. numero di telefono – potrebbe essere ignoto (caso 1.) o inesistente (caso 2.))

Il **modello relazionale** non distingue i tre casi: l'interpretazione del significato di un valore nullo è esterna al modello relazionale

Semantica di Codd del valore nullo

La semantica di Codd del valore nullo è il comportamento formale di una interrogazione a fronte di tuple che contengono valori nulli.

Ci riferiamo in particolare alla selezione σ_p con predicato *p* che è una combinazione di confronti del tipo:

- A_i φ *costante*
- A_i φ A_j

Semantica di Codd del valore nullo

A_i φ *costante*

Quale valore di verità assume il confronto quando una tupla ha valore nullo in A_i?

- Es. se t[DataNascita] è non nota, cioè NULL, qual è il valore di verità del confronto t[DataNascita]=1/1/1990?
- Se il confronto fosse vero, non sarebbe coerente nel caso in cui la data di nascita effettiva risultasse essere diversa da 1/1/1990
- Se il confronto fosse falso, non sarebbe coerente nel caso in cui la data di nascita effettiva fosse 1/1/1990

Logica a tre valori

Codd propone di usare una logica a tre valori:

- *Falso (F, False),*
- *Vero (T, True),*
- *Sconosciuto (U, Unknown)*

Quindi se $t[A_i]$ è nullo, il valore di verità del confronto

$$t[A_i] \ \varphi \ costante$$

sarà sconosciuto *(U)*

Logica a tre valori

Caso del confronto tra attributi $A_i \varphi A_j$

- Ad esempio, join tra due tabelle

Quando si confrontano due valori, ci sono diversi casi possibili (assumiamo che v_i sia una costante non nulla):

1. *$v_i \varphi v_j$ (assenza di valori nulli)*
2. *$v_i \varphi$ NULL*
3. *NULL φv_j*
4. *NULL φ NULL*

Logica a tre valori

Caso del confronto tra attributi $A_i \varphi A_j$

1. *$v_i \varphi v_j$ (assenza di valori nulli)*
2. *$v_i \varphi$ NULL*
3. *NULL φv_j*
4. *NULL φ NULL*

Codd propone di assegnare **sempre**, negli ultimi tre casi, il valore di verità *Sconosciuto (U)*

Valori nulli ed espressioni algebriche

Le espressioni algebriche contengono combinazioni di predicati semplici tramite $\wedge$, $\vee$ e $\neg$

Dobbiamo estendere le tabelle di verità dei tre operatori

Operatori e logica a tre valori

∧	F	U	T
F	F	F	F
U	F	U	U
T	F	U	T

∨	F	U	T
F	F	U	T
U	U	U	T
T	T	T	T

¬	
F	T
U	U
T	F

Se consideriamo F=0, U=1 e T=2, l’and si calcola facilmente con il **minimo**, l’or con il **massimo**, ¬p come 2–p

Espressioni e logica a tre valori

Date le tabelle di verità a tre valori, ogni tupla presa in esame genera un valore di verità dell'intera espressione che può essere F, T o U

La selezione dà come risultato tutte le tuple per cui il valore di verità è **T**, escludendo quelle per cui è F o U.

Ricerca di valori nulli

- Per cercare i valori nulli non posso usare gli usuali operatori di confronto = e <>
- Infatti in corrispondenza della selezione avrei sempre valore di verità sconosciuto nel confronto con un valore nullo
- Per potere cercare una tupla che **ha (o non ha) un valore nullo** in corrispondenza dell'attributo A_i, uso un predicato speciale:
  - *A_i IS NULL* (dà T se A_i è NULL, F altrimenti)
  - *A_i IS NOT NULL* (dà T se A_i non è NULL, F altrimenti)
- Né IS NULL né IS NOT NULL sono mai valutati U

Esempi

clienti

COD	Residenza
100	TO
200	NULL
300	TO
400	NULL

- $\sigma_{\text{Residenza IS NULL}}(\text{clienti})$

COD	Residenza
200	NULL
400	NULL

- $\sigma_{\text{Residenza IS NOT NULL}}(\text{clienti})$

COD	Residenza
100	TO
300	TO

Logica a tre valori e logica classica

1 • La logica a tre valori vìola il principio di **non contraddizione** della logica classica

$$(p \wedge \neg p) = F$$

• Infatti se p è U, abbiamo $(p \wedge \neg p) = U$

2 • La logica a tre valori vìola anche il principio del **terzo escluso**

$$(p \vee \neg p) = T$$

Infatti se p è U, abbiamo $(p \vee \neg p) = U$

3 • Vale invece la doppia negazione anche quando $p = U$

$$\neg\neg p = p$$

Esempio

Ipotizziamo che *pazienti* abbia tuple con valore nullo in corrispondenza dell'attributo *AnnoNascita.*

La selezione

$$\sigma_{AnnoNascita < 1980 \vee AnnoNascita \geq 1980}(pazienti)$$

contiene un predicato della forma $(p \vee \neg p)$, ma, se in pazienti ci sono tuple con AnnoNascita *NULL*, il predicato non è T, quindi:

$$|\sigma_{AnnoNascita<1980 \vee AnnoNascita>1980}(pazienti)| < |pazienti|$$

Esempio

L'interrogazione

$$\sigma_{AnnoNascita < 1980 \ \vee AnnoNascita \geq 1980}(pazienti)$$

è equivalente a

$$\sigma_{AnnoNascita\ IS\ NOT\ NULL}(pazienti)$$

Join interni (inner join)

I join che abbiamo visto finora sono tutti join interni (inner join): fanno parte del risultato solo le (combinazioni di) tuple per cui la condizione θ è vera.

In alcuni casi vogliamo avere la garanzia di non tralasciare informazioni, cioè di avere tutte le tuple di una relazione anche se non c'è un corrispettivo nell'altra relazione.

Per questo motivo sono stati introdotti i join esterni.

Esempio: reparti

dipendenti

Impiegato	Reparto
Rossi	vendite
Neri	produzione
Bianchi	produzione

capireparto

Reparto	Capo
produzione	Mori
acquisti	Bruni

Supponiamo che non ci siano vincoli di integrità referenziale

Join esterni (outer join)

Distinguiamo tre tipi di outer join

- Left join
- Right join
- Full join

L'outer join può essere applicato in generale a un theta-join e prevedere una condizione, ma negli esempi per semplicità useremo il natural join

Left join: $⟕_\theta$

Left join tra *dipendenti* e *capireparto:*

dipendenti ⟕ *capireparto*

- Il risultato comprende tutte le tuple della relazione *dipendenti*, nessuna esclusa
- Si verifica se le tuple della relazione *dipendenti* sono in join con tuple della relazione *capireparto*
- Qualora una tupla di *dipendenti* non faccia join con nessuna tupla della relazione *capireparto*, si inseriscono valori nulli in corrispondenza degli attributi della **seconda** relazione

Left join

Left join tra *dipendenti* e *capireparto*

$$dipendenti \; ⟕ \; capireparto$$

Impiegato	Reparto	Capo
Rossi	vendite	NULL
Neri	produzione	Mori
Bianchi	produzione	Mori

DIPENDENTI

Impiegato	Reparto
Rossi	vendite
Neri	produzione
Bianchi	produzione

CAPIREPARTO

Reparto	Capo
produzione	Mori
acquisti	Bruni

Left join: $⟕_\theta$

Definibile tramite gli altri operatori:

$r_1(A) ⟕ r_2(B) :=$

$(r_1 \bowtie r_2) \cup$ *(inner join unito alle…)*

$((r_1 - \pi_A(r_1(A) \bowtie r_2(B))$ *(tuple in r_1 assenti dall’inner join)*

$\times \{(NULL, ..., NULL)\}$ *(accoppiate con la relazione costantemente NULL con schema B-A)*

Right join: $⟖_{\theta}$

Right join tra *clienti* e *capireparto:*

dipendenti ⟖ *capireparto*

- Il risultato comprende tutte le tuple della relazione *capireparto*, nessuna esclusa
- Si verifica se le tuple della relazione *capireparto* sono in join con tuple della relazione *dipendenti*
- Qualora una tupla di *capireparto* non faccia join con nessuna tupla della relazione *dipendenti*, si inseriscono valori nulli in corrispondenza degli attributi della **prima** relazione

Right join

Right join tra *dipendenti* e *capireparto*

dipendenti ⟖ *capireparto*

Impiegato	Reparto	Capo
Neri	produzione	Mori
Bianchi	produzione	Mori
NULL	acquisti	Bruni

DIPENDENTI

Impiegato	Reparto
Rossi	vendite
Neri	produzione
Bianchi	produzione

CAPIREPARTO

Reparto	Capo
produzione	Mori
acquisti	Bruni

Full join: ⟗

Il full join è l'unione del left join e del right join

dipendenti ⟗ *capireparto*

equivale a

(dipendenti ⟕ *capireparto)*

∪

(dipendenti ⟖ *capireparto)*

Full join

Full join tra *clienti* e *agenzie*

dipendenti ⟗ *capireparto*

Impiegato	Reparto	Capo
Rossi	vendite	NULL
Neri	produzione	Mori
Bianchi	produzione	Mori
NULL	acquisti	Bruni

DIPENDENTI

<u>Impiegato</u>	Reparto
Rossi	vendite
Neri	produzione
Bianchi	produzione

CAPIREPARTO

<u>Reparto</u>	Capo
produzione	Mori
acquisti	Bruni

Proprietà degli operatori algebrici

Obiettivi:

- Conoscere bene le proprietà permette di districarsi nelle espressioni algebriche complesse per manipolarle (ad esempio per semplificarle, per sapere se due espressioni sono equivalenti, ecc.)
- È importante conoscere le proprietà perché sono effettivamente utilizzate dal DBMS per ottimizzare le interrogazioni

Gruppi di proprietà

- *Commutative*
  - del prodotto cartesiano $r(A) \times s(B) = s(B) \times r(A)$
  - del theta-join $r(A) \bowtie_\Theta s(B) = s(B) \bowtie_\Theta r(A)$
- *Associative*
  - del prodotto cartesiano $(r(A) \times s(B)) \times u(C) = r(A) \times (s(B) \times u(C))$
  - ristretta del theta-join

    $r(A) \bowtie_{\theta(A,B \cup C)} (s(B,C) \bowtie_{\theta(B \cup C, D \cup E)} u(D,E)) =$

    $(r(A) \bowtie_{\theta(A,B \cup C)} s(B,C)) \bowtie_{\theta(B \cup C, D \cup E)} u(D,E)$
- *Selezione multipla* $\sigma_{p \wedge q}(r(A)) = \sigma_p(\sigma_q(r(A)))$
- *Sostituzione di operatori*

  1) $\sigma_{p \wedge q}(r(A)) = \sigma_p(r(A)) \cap \sigma_q(r(A))),$ 2) $\sigma_{p \vee q}(r(A)) = \sigma_p(r(A)) \cup \sigma_q(r(A))),$

  3) $\sigma_{p \wedge \neg q}(r(A)) = \sigma_p(r(A)) - \sigma_q (r(A)))$

Gruppi di proprietà

- *Distributive della selezione* (ristrette a opportuni predicati p!) rispetto a:
  - proiezione $\sigma_p(\pi_X(r(A))) \rightarrow \pi_X(\sigma_p(r(A)))$
  - prodotto cartesiano $\sigma_p(r(A) \times s(B)) = \sigma_p(r(A)) \times s(B)$
  - join $r(A) \bowtie_{\Theta(A,B\cup C)} (s(B,C) \bowtie_{\Theta(B\cup C,D\cup E)} u(D,E)) =$
    $(r(A) \bowtie_{\Theta(A,B\cup C)} s(B,C)) \bowtie_{\Theta(B\cup C,D\cup E)} u(D,E)$
  - unione, differenza, intersezione
- *Proiezione multipla* $\pi_X(\pi_{X,Y}(r(A))) = \pi_X(r(A))$
- *Distributive della proiezione* rispetto a:
  - prodotto cartesiano $\pi_X(r(A) \times s(B)) = \pi_{X\cap A}(r(A)) \times \pi_{X\cap B}(s(B))$
  - join $\pi_X(r(A) \bowtie_\Theta s(B)) = \pi_{X\cap A}(r(A)) \bowtie_\Theta \pi_{X\cap B}(s(B))$
  - unione $\pi_X(r(A) \cup s(A)) = \pi_X(r(A)) \cup \pi_X(s(A))$

Proprietà commutativa del prodotto cartesiano

Ricordandoci che

- lo schema risultato del prodotto cartesiano è l'unione degli schemi
- l'ordine degli attributi in una relazione di Codd è irrilevante,
- il prodotto cartesiano combina ogni tupla di una relazione con ogni tupla dell’altra relazione

possiamo concludere che:

$$\boldsymbol{r(A) \times s(B) = s(B) \times r(A)}$$

Proprietà commutativa del Θ-join

Ricordandoci che

$$r(A) \bowtie_{\theta} s(B) := \sigma_{\theta}(r(A) \times s(B))$$

e che, come abbiamo dimostrato, vale la commutatività del prodotto cartesiano, possiamo scrivere

$$\sigma_{\theta}(r(A) \times s(B)) = \sigma_{\theta}(s(B) \times r(A))$$

quindi

$$\sigma_{\theta}(s(B) \times r(A)) = s(B) \bowtie_{\theta} r(A)$$

cioè

$$\boldsymbol{r(A) \bowtie_{\theta} s(B) = s(B) \bowtie_{\theta} r(A)}$$

Proprietà associativa del prodotto cartesiano

$$\boldsymbol{(}r(A) \times s(B)\boldsymbol{)} \times u(C) = r(A) \times \boldsymbol{(}s(B) \times u(C)\boldsymbol{)}$$

- lo schema di $r(A) \times s(B)$ *è* $A \cup B$
- lo schema di $(r(A) \times s(B)) \times u(C)$ è $(A \cup B) \cup C$
- lo schema di $(A \cup B) \cup C$ è uguale a $A \cup (B \cup C)$ per la proprietà associativa dell'unione
- di conseguenza i due schemi coincidono
- il prodotto cartesiano combina ogni tupla di una relazione con ogni tupla dell'altra relazione

$$\boldsymbol{(r(A) \times s(B)) \times u(C) = r(A) \times (s(B) \times u(C))}$$

Conseguenza della proprietà associativa

Non è ambiguo scrivere

$$r(A) \times s(B) \times u(C)$$

senza parentesizzazione e l'ottimizzatore è libero di scegliere se eseguire prima $r(A) \times s(B)$ o prima $s(B) \times u(C)$

Proprietà associativa ristretta del Θ-join

Il Θ-join è associativo a patto che gli attributi di Θ possano essere suddivisi in questo modo:

$$r(A) \bowtie_{\Theta(A,B \cup C)} (s(B,C) \bowtie_{\Theta(B \cup C, D \cup E)} u(D,E))$$

$$=$$

$$(r(A) \bowtie_{\Theta(A,B \cup C)} s(B,C)) \bowtie_{\Theta(B \cup C, D \cup E)} u(D,E)$$

dove Θ(X, Y) = formula con confronti tra attributi in X e in Y, cioè confronti $X_i \varphi Y_j$

Proprietà associativa ristretta del Θ-join

Per esempio:

$$r(A) \bowtie_{A=B} (s(B,C) \bowtie_{C=D} u(D,E))$$

$$=$$

$$(r(A) \bowtie_{A=B} s(B,C)) \bowtie_{C=D} u(D,E)$$

Però

$$r(A) \bowtie_{A=E} (s(B,C) \bowtie_{C=D} u(D,E))$$

$$\neq$$

$$(r(A) \bowtie_{\underline{A=E}} s(B,C)) \bowtie_{C=D} u(D,E)$$

Proprietà della selezione multipla

$$\sigma_{p \wedge q}(r(A)) = \sigma_p(\sigma_q(r(A)))$$

La selezione con predicato $p \wedge q$ sceglie le tuple che soddisfano sia il predicato p che il predicato q

La selezione con predicato p applicata alle tuple selezionate con predicato q seleziona le tuple che soddisfano contemporaneamente p e q

Proprietà della sostituzione di operatori

$$\sigma_{p \wedge q}(r(A)) = \sigma_p(r(A)) \cap \sigma_q(r(A)))$$
$$\sigma_{p \vee q}(r(A)) = \sigma_p(r(A)) \cup \sigma_q(r(A)))$$
$$\sigma_{p \wedge \neg q}(r(A)) = \sigma_p(r(A)) - \sigma_q(r(A)))$$

Dimostrazione simile alla precedente derivata dalla semantica degli operatori logici

(al DBMS costano molto meno le forme a sinistra)

Proprietà distributiva della selezione rispetto alla proiezione

$$\sigma_p(\pi_X(r(A))) = \pi_X(\sigma_p(r(A)))$$

Perché questa proprietà sia valida, bisogna prestare attenzione agli attributi su cui è definita *p*…

Proprietà distributiva della selezione rispetto alla proiezione

Tutte le proprietà viste finora (distributiva esclusa) valgono come **regole di riscrittura** in **entrambi i versi**

Esempio:

- se trovo $\sigma_{p \wedge q}(r(A))$
- posso scrivere $\sigma_p(r(A)) \cap \sigma_q(r(A)))$

ma vale anche il **viceversa**:

- se trovo $\sigma_p(r(A)) \cap \sigma_q (r(A)))$
- posso scrivere $\sigma_{p \wedge q}(r(A))$

Proprietà distributiva della selezione rispetto alla proiezione

Nel caso della proprietà distributiva della selezione

- se trovo: $\sigma_p(\pi_X(r(A)))$
- posso sempre riscrivere $\pi_X(\sigma_p(r(A)))$

ma il **viceversa non è sempre valido**, ovvero

- se trovo: $\pi_X(\sigma_p(r(A)))$
- affinché possa scrivere $\sigma_p(\pi_X(r(A)))$ è necessario che il predicato *p* sia definito solo su attributi di *X*

Esempi

$$\pi_{COD,AnnoNascita}(\sigma_{AnnoNascita>1980}(pazienti))$$

posso riscriverla come

$$\sigma_{AnnoNascita>1980}(\pi_{COD,AnnoNascita}(pazienti))$$

Invece:

$$\pi_{COD}(\sigma_{AnnoNascita>1980}(pazienti))$$

non può essere riscritta come

$$\sigma_{AnnoNascita>1980}(\pi_{COD}(pazienti))$$

Proprietà distributiva della selezione rispetto al prodotto cartesiano

$$\sigma_p(r(A) \times s(B))$$

Quando *p* coinvolge **sia attributi di *A* che attributi di *B* non** c'è nessuna possibilità di applicare proprietà distributive

Se invece *p* coinvolge solo attributi contenuti nello schema di **una** delle due relazioni (ad esempio *A*), è possibile scrivere

$$\sigma_p(r(A) \times s(B)) = \sigma_p(r(A)) \times s(B)$$

Proprietà distributiva della selezione rispetto al join

$$\sigma_p(r(A) \bowtie_\theta s(B))$$

Stesse considerazioni di quelle sul prodotto cartesiano

Quando *p* coinvolge sia attributi di *A* che attributi di *B* non c'è nessuna possibilità di applicare proprietà distributive

Se invece *p* coinvolge solo attributi contenuti nello schema di una delle due relazioni (ad esempio *A*), è possibile scrivere

$$\sigma_p(r(A) \bowtie_\theta s(B)) = \sigma_p(r(A)) \bowtie_\theta s(B)$$

Proprietà distributiva della selezione rispetto all'unione, all'intersezione e alla differenza

$$\sigma_p(r(A) \cup s(A)) = \sigma_p(r(A)) \cup \sigma_p(s(A))$$
$$\sigma_p(r(A) \cap s(A)) = \sigma_p(r(A)) \cap \sigma_p(s(A))$$
$$\sigma_p(r(A) - s(A)) = \sigma_p(r(A)) - \sigma_p(s(A))$$

La dimostrazione è immediata (si vede con i diagrammi di Venn)

Per la differenza (e intersezione) vale anche:

$$\sigma_p(r(A) - s(A)) = \boldsymbol{\sigma_p(r(A))} - s(A)$$
$$\sigma_p(r(A) \cap s(A)) = \boldsymbol{\sigma_p(r(A))} \cap s(A)$$

Proprietà della proiezione multipla

$$\pi_X(\pi_{X,Y}(r(A))) = \pi_X(r(A))$$

dove X e Y sono sottoinsiemi di A

La dimostrazione è immediata dalla definizione di proiezione

Proprietà distributiva della proiezione rispetto al prodotto cartesiano

$$\pi_X(r(A) \times s(B)) = \pi_{X \cap A}(r(A)) \times \pi_{X \cap B}(s(B))$$

La dimostrazione è immediata

Proprietà distributiva della proiezione rispetto al join

$$\pi_X(r(A) \bowtie_\Theta s(B)) = \pi_{X \cap A}(r(A)) \bowtie_\Theta \pi_{X \cap B}(s(B))$$

È necessario che gli attributi coinvolti nel predicato Θ siano contenuti in $X \cap A$ e in $X \cap B$

Se X non comprende gli attributi usati da Θ, la proprietà non vale

La dimostrazione è immediata

Proprietà distributiva della proiezione rispetto all'unione

$$\pi_X(r(A) \cup s(A)) = \pi_X(r(A)) \cup \pi_X(s(A))$$

La dimostrazione è immediata

N.B.: le proprietà distributive della proiezione rispetto alla differenza e all'intersezione **non valgono**.

Vediamo un controesempio con la differenza…

Proprietà non valida su differenza

$\pi_X(r(A) - s(A)) \neq \pi_X(r(A)) - \pi_X(s(A))$

Immaginiamo due relazioni con due soli attributi *A* e *B*

r

A	B
a	b_1

s

A	B
a	b_2

con $b_1 \neq b_2$

$\pi_A(r - s) = \pi_A($

A	B
a	b_1

$) =$

A
a

$\pi_A(r) - \pi_A(s) =$

A
a

$-$

A
a

$= \emptyset$