Basi di dati

Maurizio Lenzerini

Dipartimento di Ingegneria Informatica
Automatica e Gestionale "Antonio Ruberti"
Università di Roma "La Sapienza"

Anno Accademico 2011/2012

http://www.dis.uniroma1.it/~lenzerini/?q=node/44

2. Il modello relazionale

2.1 Basi di dati relazionali

1. **basi di dati relazionali**
2. algebra relazionale

Il modello relazionale

- Proposto da E. F. Codd nel 1970 per favorire l'indipendenza fisica dei dati (ovvero per rendere il modo in cui si usano i dati a livello logico indipendente dalla loro memorizzazione fisica)
- Disponibile come modello logico in DBMS reali nel 1981 (10 anni di incubazione)
- Si basa sul concetto matematico di **relazione** (ma con importanti varianti)
- Le relazioni hanno una rappresentazione naturale per mezzo di tabelle
- Il modello è "**basato su valori**": anche i riferimenti fra dati in strutture (relazioni) diverse sono rappresentati per mezzo dei valori stessi

Relazione: tre accezioni

- **relazione matematica**: come nella teoria degli insiemi
- relazione (dall'inglese **relationship**) fra l'insieme delle istanze di due o più **entità**, nel modello **Entity-Relationship** (talvolta tradotto con **associazione** o **correlazione**)
- **relazione** secondo il modello relazionale dei dati: tabella

Relazione matematica

- Siano $D_1, D_2, \ldots, D_n$ n insiemi, non necessariamente distinti
- il **prodotto cartesiano** $D_1 \times D_2 \times \cdots \times D_n$, è l'insieme di tutte le n-uple ordinate $(d_1,d_2,\ldots,d_n)$ tali che $d_1 \in D_1$, $d_2 \in D_2$,…, $d_n \in D_n$
- una relazione matematica sugli insiemi $D_1, D_2, \ldots, D_n$ è un **sottoinsieme del prodotto cartesiano $D_1 \times D_2 \times \cdots \times D_n$**
- $D_1, D_2, \ldots, D_n$ sono i **domini** della relazione, anche detti **componenti** della relazione. Una relazione su n domini ha **grado** (o **arietà**) n
- il numero di n-uple è la **cardinalità** della relazione

Relazione matematica: esempio

- $D_1 = \{a,b\}$
- $D_2 = \{x,y,z\}$
- prodotto cartesiano $D_1 \times D_2$

a	x
a	y
a	z
b	x
b	y
b	z

- una relazione $r \subseteq D_1 \times D_2$

a	x
a	z
b	y
b	z

Relazione matematica: proprietà

Una **relazione matematica** è quindi un insieme di n-uple ordinate (dette anche **ennuple**, o **tuple**) su $D_1, D_2, \ldots, D_n$, ovvero n-ple della forma $(d_1, \ldots, d_n)$ tali che $d_1 \in D_1, \ldots, d_n \in D_n$

Si noti che una relazione è un **insieme** di tuple e quindi:

- non c'è ordinamento fra le sue n-uple
- le sue n-uple sono tutte distinte una dall'altra

Ciascuna n-upla di una relazione è **ordinata** e quindi

- il valore del primo elemento dell n-pla viene dal primo dominio
- il valore del secondo elemento dell n-pla viene dal secondo dominio
- …
- il valore del n-esimo elemento proviene dall'n-esimo dominio

Relazione matematica: esempio

$$Partita \subseteq string \times string \times integer \times integer$$

Juve	Lazio	3	1
Lazio	Milan	2	0
Juve	Roma	1	2
Roma	Milan	0	1

- Il primo ed il terzo dominio si riferiscono rispettivamente a nome e numero di reti della squadra ospitante; il secondo e il quarto a nome e numero reti della squadra ospitata

- La struttura è **posizionale**, nel senso che il primo ed il secondo dominio si riferiscono allo stesso insieme (string), ma sono distinguibili dalla posizione (analogamente per il terzo e quarto dominio, entrambi integer)

Relazione nel modello relazionale dei dati

- Una relazione nel modello relazionale è simile ad una relazione matematica, ma con le seguenti differenze:
  - le varie componenti sono dette **attributi**,
  - ogni attributo è caratterizzato da un nome ed un insieme di valori (detto dominio dell'attributo, da non confondere con il termine usato per indicare le componenti di una relazione matematica)

- Da ora in poi, quando parliamo di relazione intendiamo "relazione nel modello relazionale", e quando vogliamo parlare di relazione matematica useremo il termine "relazione matematica"

Relazione nel modello relazionale dei dati

- Una relazione si può quindi rappresentare come una **tabella** in cui gli attributi corrispondono alle colonne ed i nomi degli attributi sono usati come intestazioni delle colonne

- Poiché adesso ogni componente della relazione è identificata da un attributo, l'ordinamento fra gli attributi è irrilevante: la struttura, al contrario della relazione matematica, è **non posizionale**

Relazione nel modello relazionale dei dati

- Esempio:

attributo

ennupla
o tupla

Casa	Fuori	RetiCasa	RetiFuori
Juve	Lazio	3	1
Lazio	Milan	2	0
Juve	Roma	1	2
Roma	Milan	0	1

- In questa relazione gli attributi sono 4, e nella rappresentazione tabellare essi corrispondono alle colonne della tabella. Le ennuple, invece, corrispondono alle righe.
- I domini degli attributi sono *string* per Casa e Fuori, ed *integer* per RetiCasa e RetiFuori. Essi non vengono mostrati nella rappresentazione tabellare

Notazioni

- Sia X l'insieme degli attributi di una relazione R. Se t è una ennupla di R, cioè una ennupla su X, e A ∈ X, allora

  t[A] (oppure **t.A**)

  indica **il valore che la ennupla t ha in corrispondenza dell'attributo A**

- Se t è la ennupla che compare per prima nella tabella dell'esempio precedente, allora si ha che t[Fuori] = Lazio

- La stessa notazione è estesa anche ad insiemi di attributi:

  t[Fuori,RetiFuori] indica una ennupla sui due attributi Fuori e RetiFuori

- Riferendoci alla ennupla t vista prima, si ha che

  t[Fuori,RetiFuori] = <Lazio, 1>

Altra notazione

- La specifica del nome R di una relazione, degli attributi A1,A2,…,An e dei domini di tali attributi D1,D2,…,Dn forma il cosiddetto schema di relazione, che si denota come

  $$R(A1 : D1, A2 : D2, \ldots , An : Dn)$$

  oppure semplicemente come (nel caso non interessi esplicitare i domini degli attributi)

  $$R(A1, A2, \ldots , An)$$

- Di conseguenza, una tupla di R con t[A1] = a, t[A2] = b, …, t[An] = c si può denotare anche come <A1:a, A2:b,…, An:c>

- In altre parole, la tupla si può vedere come "tupla etichettata", in cui le etichette sono gli attributi della relazione, ed i valori associati alle etichette sono i valori che compongono la tupla

Tabelle e relazioni

- Sottolineiamo che in una relazione del modello relazionale:
  - i valori di ciascuna colonna sono fra loro omogenei, cioè appartengono allo stesso dominio
  - le righe (cioè le tuple) tutte sono diverse fra loro
  - le intestazioni delle colonne (attributi) sono tutte diverse tra loro
- Inoltre, nella rappresentazione tabellare della relazione
  - l'ordinamento tra le righe è irrilevante
  - l'ordinamento tra le colonne è irrilevante
- Sottolineiamo anche che **il modello relazionale è basato sui valori**: i riferimenti fra due relazioni diverse sono espressi per mezzo di valori che compaiono nelle ennuple di entrambe le relazioni

Studente

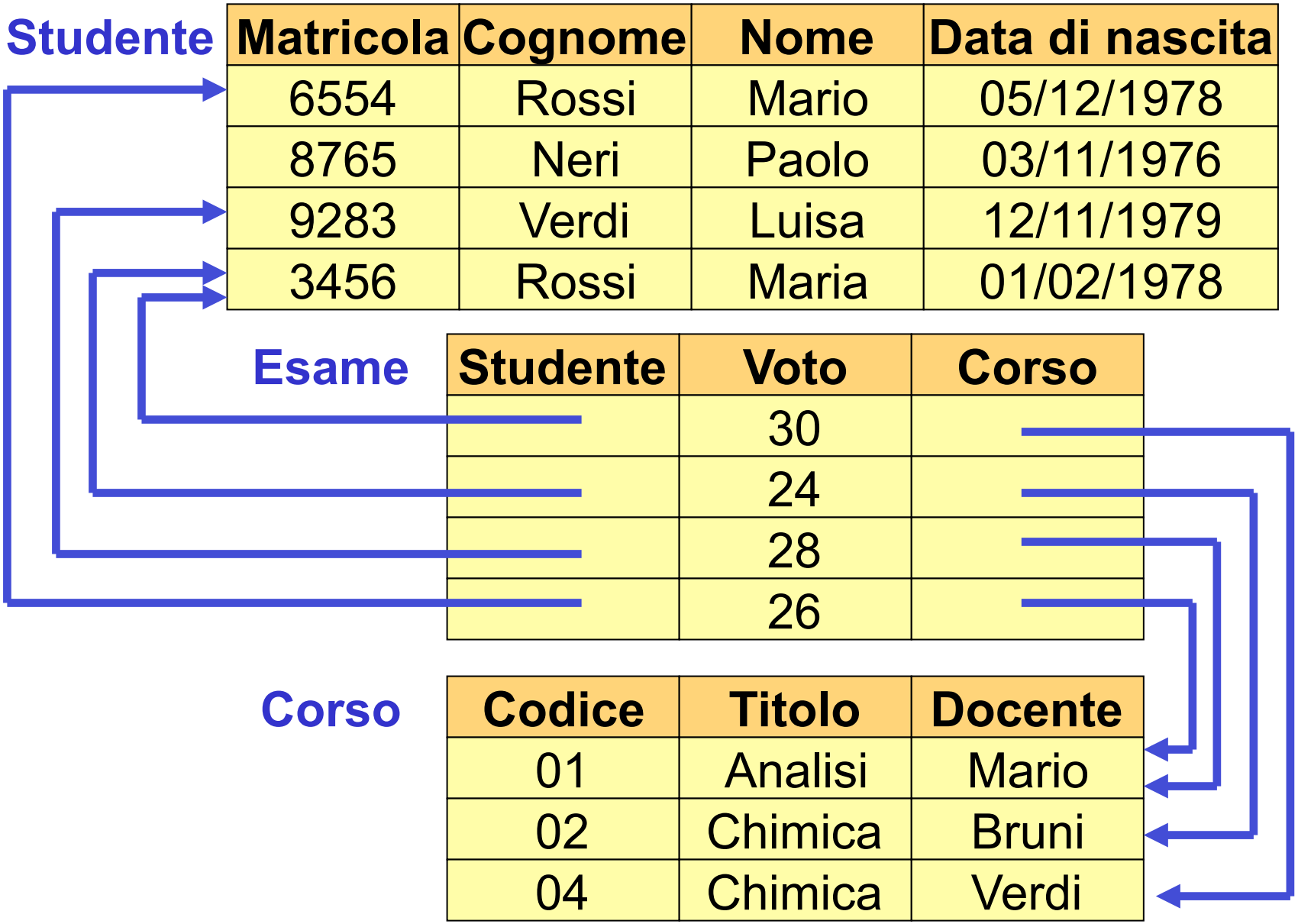
Matricola	Cognome	Nome	Data di nascita
6554	Rossi	Mario	05/12/1978
8765	Neri	Paolo	03/11/1976
9283	Verdi	Luisa	12/11/1979
3456	Rossi	Maria	01/02/1978

Esame

Studente	Voto	Corso
3456	30	04
3456	24	02
9283	28	01
6554	26	01

Corso

Codice	Titolo	Docente
01	Analisi	Mario
02	Chimica	Bruni
04	Chimica	Verdi


Studente
Matricola
Cognome
Nome
Data di nascita
6554
Rossi
Mario
05/12/1978
8765
Neri
Paolo
03/11/1976
9283
Verdi
Luisa
12/11/1979
3456
Rossi
Maria
01/02/1978
Esame
Studente
Voto
Corso
30
24
28
26
Corso
Codice
Titolo
Docente
01
Analisi
Mario
02
Chimica
Bruni
04
Chimica
Verdi

Vantaggi della struttura basata su valori

- **indipendenza dalle strutture fisiche**, che possono cambiare anche dinamicamente
- si rappresenta solo ciò che è **rilevante dal punto di vista dell'applicazione** (dell'utente); i puntatori sono meno comprensibili per l'utente finale
- i dati sono **portabili** più facilmente da un sistema ad un altro
- i valori consentono **bi-direzionalità**, mentre i puntatori sono direzionali

Nota: i puntatori possono essere usati a livello fisico dal sistema, se questo è vantaggioso per l'efficienza

Definizioni

Schema di relazione: un **nome di relazione** R con un insieme di **attributi** $A_1,..., A_n$ ed eventualmente anche i corrispondenti domini

$R(A_1, ... ,A_n)$ oppure $R(A_1 : D_1, A_2 : D_2, \ldots , A_n : D_n)$

Schema di base di dati: insieme di schemi di relazione con nomi diversi:

$$\mathbf{R} = \{R_1(X_1), ... ,R_n(X_n)\}$$

(Istanza di) relazione su uno schema R(X): insieme r di ennuple su X

(Istanza di) base di dati su uno schema **R**= $\{R_1(X_1), ... , R_n(X_n)\}$: insieme di relazioni **r** = $\{r_1,..., r_n\}$, con una relazione r_i sullo schema $R_i(X_i)$, per ogni i = 1,…,n

Esempio

Schema di base di dati R:

{ Studente(Matricola,Cognome,Nome,DataDiNascita),
StudenteLavoratore(Matricola) }

Istanza di base di dati sullo schema **R**:

Studente

Matricola	Cognome	Nome	DataDiNascita
6554	Rossi	Mario	05/12/1978
8765	Neri	Paolo	03/11/1976
9283	Verdi	Luisa	12/11/1979
3456	Rossi	Maria	01/02/1978

StudenteLavoratore

Matricola
6554
3456

Informazione incompleta

- Il modello relazionale impone ai dati una struttura rigida:
  - l'informazione è rappresentata per mezzo di ennuple
  - le ennuple ammesse sono dettate dagli schemi di relazione
- Nella pratica, però, i dati disponibili possono non corrispondere esattamente al formato previsto, per varie ragioni

  Esempio:
  - di Firenze non conosciamo l'indirizzo della prefettura
  - Tivoli non è provincia: non ha prefettura
  - Prato è "nuova" provincia: ha la prefettura?

Prefettura

città	indirizzo
Roma	Via IV Novembre
Firenze	
Tivoli	
Prato	

Informazione incompleta: soluzioni?

Spesso si utilizzare valori ordinari del dominio (0, stringa nulla, “99”, etc) per rappresentare informazione mancante. Questo però è un errore, per vari motivi:

- potrebbero non esistere valori “non utilizzati”
- valori “non utilizzati” fino ad un certo momento potrebbero diventare significativi in seguito
- in fase di utilizzo (ad esempio, nei programmi) è necessario ogni volta tener conto del “significato” di questi valori speciali e questo richiede di mettere d’accordo diversi programmatori, cosa non semplice nella pratica

Informazione incompleta nel modello relazionale

- Si adotta una tecnica rudimentale, ma per certi versi efficace:
  - Viene introdotto il cosiddetto **valore nullo**: esso denota l'assenza di un valore del dominio (e non è un valore del dominio, anche se può comparire come valore degli attributi definiti su quel dominio)

- Formalmente, è sufficiente estendere il concetto di ennupla: t[A], per ogni attributo A, è un valore del dominio dom(A) oppure il valore nullo **NULL**

- Al momento di definire uno schema di basi di dati, si possono poi imporre restrizioni sulla presenza di valori nulli nei vari attributi delle relazioni

Interpretazioni del valore nullo

- (almeno) tre casi differenti
  - valore **sconosciuto**: esiste un valore del dominio, ma non è noto (nell'esempio precedente: Firenze)
  - valore **inesistente**: non esiste un valore del dominio (nell'esempio precedente: Tivoli)
  - valore **senza informazione**: non è noto se esista o meno un valore del dominio (nell'esempio precedente: Prato)

- I DBMS non distinguono i tipi di valore nullo (e quindi implicitamente adottano l'interpretazione "**senza informazione**")

Vincoli di integrità: introduzione

Esistono istanze di basi di dati che, pur sintatticamente corrette, non rappresentano informazioni possibili per l'applicazione di interesse.

Studente

Matricola	Cognome	Nome	Nascita
276545	Rossi	Maria	23/04/1968
276545	Neri	Anna	23/04/1972
788854	Verdi	Fabio	12/02/1972

Esame

Studente	Voto	Lode	Corso
276545	28	e lode	01
276545	32		02
788854	23		03
200768	30	e lode	03

Corso

Codice	Titolo	Docente
01	Analisi	Giani
03	NULL	NULL
02	Chimica	Belli

Vincolo di integrità

Definizione di vincolo di integrità

- Un vincolo di integrità (o semplicemente vincolo) è una condizione che si esprime a livello di schema e che si intende debba essere soddisfatta da tutte le istanze della base di dati, perché individua una condizione necessaria per tutte quelle istanze della base di dati che rappresentano situazioni corrette per l'applicazione
- Ogni vincolo può essere visto come una funzione booleana (o un predicato) che associa ad ogni istanza della base di dati il valore **vero** (nel caso in cui il vincolo sia soddisfatto) o **falso** (altrimenti)

• Ad uno schema si associa un insieme di vincoli e si considerano **corrette** (diciamo anche lecite, legali, valide, ammissibili) solo le istanze che soddisfano tutti i vincoli

Vincoli di integrità: motivazioni

- risultano utili al fine di descrivere la realtà di interesse in modo più accurato di quanto le sole strutture permettano;
- forniscono un contributo verso la "qualità dei dati"
- costituiscono uno strumento di ausilio alla progettazione
- sono utilizzati dal sistema nella scelta della strategia di esecuzione delle interrogazioni

Nota:

- non tutte le proprietà di interesse sono rappresentabili per mezzo di vincoli esprimibili direttamente

Vincoli di integrità: classificazione

- Intrarelazionali
  - di ennupla
    - di dominio
  - di chiave
- Interrelazionali
  - di integrità referenziale (o di foreign key)

Vincoli intrarelazionali: vincoli di ennupla

- Esprimono **condizioni sui valori di ciascuna ennupla** di una relazione, indipendentemente dalle altre ennuple
- Un vincolo di ennupla su una relazione R si esprime come un'espressione booleana (con AND, OR e NOT) costruita su atomi che confrontano valori di attributi (della relazione R) o espressioni aritmetiche su di essi
- Un vincolo di ennupla che coinvolge un solo attributo di dice **vincolo di dominio**

Esempi di vincoli di dominio:

(Voto ≥ 18) AND (Voto ≤ 30)

(Voto = 30) OR NOT (Lode = "e lode")

Esempio di vincolo di ennupla:

Lordo = (Ritenute + Netto)

Vincoli intrarelazionali: vincoli di chiave

Matricola	Cognome	Nome	Corso	Nascita
27655	Rossi	Mario	Ing Inf	5/12/78
78763	Rossi	Mario	Ing Inf	3/11/76
65432	Neri	Piero	Ing Mecc	10/7/79
87654	Neri	Mario	Ing Inf	3/11/76
67653	Rossi	Piero	Ing Mecc	5/12/78

- il numero di matricola identifica gli studenti:
  - non ci sono due ennuple con lo stesso valore sull'attributo Matricola
- i dati anagrafici identificano gli studenti:
  - non ci sono due ennuple uguali su tutti e tre gli attributi Cognome, Nome e Nascita

Vincoli intrarelazionali: il concetto di chiave

Una chiave di una relazione è un insieme di attributi che identificano univocamente le ennuple di una relazione

Più precisamente:

- Sia R uno schema di relazione sull'insieme X di attributi, sia K un sottoinsieme di X, e sia r una istanza di R
- K è **superchiave** per r se r non contiene due ennuple distinte t_1 e t_2 tali che $t_1[K] = t_2[K]$
- K è **chiave** per r se è una superchiave minimale per r, ossia se K è una superchiave di r e se nessun sottoinsieme proprio di K è una superchiave per r

Il concetto di chiave: esempi

Matricola	Cognome	Nome	Corso	Nascita
27655	Rossi	Mario	Ing Inf	5/12/78
78763	Rossi	Mario	Ing Inf	3/11/76
65432	Neri	Piero	Ing Mecc	10/7/79
87654	Neri	Mario	Ing Inf	3/11/76
67653	Rossi	Piero	Ing Mecc	5/12/78

- **Matricola** è una chiave, infatti:
  - Matricola è superchiave
  - contiene un solo attributo e quindi è minimale
- **Cognome, Nome, Nascita** è un'altra chiave, infatti:
  - l'insieme Cognome, Nome, Nascita è superchiave
  - nessuno dei suoi sottoinsiemi è superchiave
- Cognome, Nome, Nascita, Corso è superchiave (non chiave)

Individuazione delle chiavi

Individuiamo le chiavi

- considerando le proprietà che i dati devono soddisfare nell'applicazione (il "frammento di mondo reale di interesse")
- notando quali insiemi di attributi permettono di identificare univocamente le ennuple, in qualunque istanza della base di dati
- e individuando i sottoinsiemi minimali di tali insiemi che conservano la capacità di identificare le ennuple

Esempio:

Studenti(Matricola, Cognome, Nome, Corso, Nascita) ha una chiave:

Matricola

Vincolo di chiave

- Un **vincolo di chiave** è un'asserzione che specifica che un insieme di attributi formano una chiave per una relazione.

- In altre parole, se in una relazione R(A,B,C,D) dichiaro un vincolo di chiave su {A,B}, sto asserendo che in tutte le istanze della basi di dati, non esistono due tuple della relazione R che coincidono negli attributi A e B e sto anche asserendo che nessun sottoinsieme proprio di {A,B} è una chiave.

- Non ci sono limitazioni per il numero di vincoli di chiave che si definiscono per una relazione (a parte il limite derivante dal numero di attributi)

Esistenza delle chiavi

- poiché le relazioni sono insiemi, una relazione non può contenere ennuple uguali fra loro:
  - ne segue che ogni relazione ha come superchiave l'insieme degli attributi su cui è definita
- poiché l'insieme di tutti gli attributi è una superchiave per ogni relazione, ogni schema di relazione ha almeno una superchiave
- ne segue che **ogni schema di relazione ha (almeno) una chiave**

Importanza delle chiavi

- L'esistenza delle chiavi garantisce l'accessibilità a ciascun dato della base di dati

- Ogni singolo valore è univocamente accessibile tramite:
  - nome della relazione
  - valore della chiave (che indica al massimo una ennupla della relazione)
  - nome dell'attributo in corrispondenza del quale è presente il valore da accedere

- Come vedremo più avanti, le chiavi sono lo strumento principale attraverso il quale vengono correlati i dati in relazioni diverse ("il modello relazionale è basato su valori")

Chiavi e valori nulli

- In presenza di valori nulli, i valori degli attributi che formano la chiave:
  - non permettono di identificare le ennuple come desiderato
  - né permettono di realizzare facilmente i riferimenti da altre relazioni

Matricola	Cognome	Nome	Corso	Nascita
NULL	NULL	Mario	Ing Inf	5/12/78
78763	Rossi	Mario	Ing Civile	3/11/76
65432	Neri	Piero	Ing Mecc	10/7/79
87654	Neri	Mario	Ing Inf	NULL
NULL	Neri	Mario	NULL	5/12/78

Chiave primaria

- La presenza di valori nulli nelle chiavi deve essere limitata
- Soluzione pratica: per ogni relazione scegliamo una chiave (la **chiave primaria**) su cui non ammettiamo valori nulli.
- Notazione per la chiave primaria: gli attributi che la compongono sono sottolineati

Matricola	Cognome	Nome	Corso	Nascita
27655	Rossi	Mario	Ing Inf	5/12/78
78763	Rossi	Mario	Ing Civile	3/11/76
65432	Neri	Piero	Ing Mecc	10/7/79
87654	Neri	Mario	Ing Inf	NULL
67653	Rossi	Piero	NULL	5/12/78

Vincolo di chiave primaria

- Un **vincolo di chiave primaria** è un'asserzione che specifica che
  - un insieme di attributi formano una chiave per una relazione e
  - non si ammettono per tali attributi i valori nulli.

- Un solo vincolo di chiave primaria è ammessa per una relazione.

Vincoli interrelazionali: integrità referenziale

- Informazioni in relazioni diverse sono correlate attraverso valori comuni, in particolare, attraverso valori delle chiavi (primarie, di solito)

- Un **vincolo di integrità referenziale** (detto anche vincolo di "**foreign key**") fra un insieme di attributi X di una relazione R_1 e un'altra relazione R_2 impone ai valori su X di ciascuna ennupla dell'istanza di R_1 di comparire come valori della chiave (primaria) dell'istanza di R_2

- Giocano un ruolo fondamentale nel concetto di "modello basato su valori"

Vincoli di integrità referenziale: esempio

Infrazioni

Codice	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	E39548
64521	5/4/96	3295	PR	839548
73321	5/2/98	9345	PR	839548

Vigili

Matricola	Cognome	Nome
3987	Rossi	Luca
3295	Neri	Piero
9345	Neri	Mario
7543	Mori	Gino

Esempio (cont.)

Infrazioni

Codice	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	E39548
64521	5/4/96	3295	PR	839548
73321	5/2/98	9345	PR	839548

Auto

Prov	Numero	Cognome	Nome
MI	39548K	Rossi	Mario
TO	E39548	Rossi	Mario
PR	839548	Neri	Luca

Vincoli di integrità referenziale: esempio

- Nell'esempio, i vincoli di integrità referenziale sussistono fra:
  - l'attributo **Vigile** della relazione **Infrazioni** e la relazione **Vigili**
  - gli attributi **Prov** e **Numero** di **Infrazioni** e gli omonimi attributi della relazione **Auto**

Violazione di vincolo di integrità referenziale

Infrazioni

Codice	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	E39548
64521	5/4/96	2468	PR	839548
73321	5/2/98	9345	PR	839548

Vigili

Matricola	Cognome	Nome
3987	Rossi	Luca
3295	Neri	Piero
9345	Neri	Mario
7543	Mori	Gino

Violazione di vincolo di integrità ref. (cont.)

Infrazioni

Codice	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	39548K
64521	5/4/96	3295	PR	839548
73321	5/2/98	9345	PR	839548

Auto

Prov	Numero	Cognome	Nome
MI	39548K	Rossi	Mario
TO	E39548	Rossi	Mario
PR	839548	Neri	Luca

Integrità referenziale e valori nulli

In presenza di valori nulli i vincoli possono essere resi meno restrittivi

Impiegati

Matricola	Cognome	Progetto
34321	Rossi	IDEA
53524	Neri	XYZ
64521	Verdi	NULL
73032	Bianchi	IDEA

Progetti

Codice	Inizio	Durata	Costo
IDEA	01/2000	36	200
XYZ	07/2001	24	120
BOH	09/2001	24	150

Integrità referenziale: azioni compensative

Sono possibili meccanismi per il supporto alla gestione dei vincoli di integrità ("azioni" compensative a seguito di violazioni)

Ad esempio, se viene eliminata una ennupla causando una violazione:

- comportamento “standard”: rifiuto dell'operazione
- azioni compensative:
  - eliminazione in cascata
  - introduzione di valori nulli

Eliminazione in cascata

Impiegati

Matricola	Cognome	Progetto
34321	Rossi	IDEA
53524	Neri	XYZ
64521	Verdi	NULL
73032	Bianchi	IDEA

Progetti

Codice	Inizio	Durata	Costo
IDEA	01/2000	36	200
XYZ	07/2001	24	120
BOH	09/2001	24	150

Introduzione di valori nulli

Impiegati

Matricola	Cognome	Progetto
34321	Rossi	IDEA
53524	Neri	NULL
64521	Verdi	NULL
73032	Bianchi	IDEA

Progetti

Codice	Inizio	Durata	Costo
IDEA	01/2000	36	200
XYZ	07/2001	24	120
BOH	09/2001	24	150

Vincoli multipli su più attributi

Incidenti

Codice	Data	ProvA	NumeroA	ProvB	NumeroB
34321	1/2/95	TO	E39548	MI	39548K
64521	5/4/96	PR	839548	TO	E39548

Auto

Prov	Numero	Cognome	Nome
MI	39548K	Rossi	Mario
TO	E39548	Rossi	Mario
PR	839548	Neri	Luca

- Vincolo di integrità referenziale da ProvA,NumeroA di Incidenti a Prov,Numero di Auto
- Vincolo di integrità referenziale da ProvB,NumeroB di Incidenti a Prov,Numero di Auto

2. Il modello relazionale

2.2 Algebra relazionale

1. basi di dati relazionali
2. **algebra relazionale**

Linguaggi per basi di dati

- Operazioni sullo schema:

  DDL: data definition language

- Operazioni sui dati:

  DML: data manipulation language

  - interrogazioni ("query language")
  - aggiornamenti

Linguaggi di interrogazione per basi di dati relazionali

Tipologia:

- **Dichiarativi**: specificano le proprietà del risultato ("*che cosa*")
- **Procedurali**: specificano le modalità di generazione del risultato ("*come*")

Rappresentanti più significativi:

- **Algebra relazionale**: procedurale
- **Calcolo relazionale**: dichiarativo
- **SQL** (Structured Query Language): parzialmente dichiarativo (linguaggio ormai standard)
- **QBE** (Query by Example): dichiarativo (reale)

Algebra relazionale

Costituita da un insieme di operatori

- definiti su relazioni
- che producono relazioni
- e possono essere composti

Operatori dell'algebra relazionale:

- unione, intersezione, differenza
- ridenominazione
- selezione
- proiezione
- Join in diverse versioni: join naturale, prodotto cartesiano, theta-join

Operatori insiemistici

- a livello estensionale, le relazioni sono insiemi di tuple, e quindi è sensato definire per essi gli operatori insiemistici
- i risultati di tali operatori debbono essere a loro volta relazioni (proprietà di chiusura delle algebre)
- è possibile applicare **unione**, **intersezione**, **differenza** solo a relazioni definite sugli stessi attributi

Unione

Laureati

Matricola	Nome	Età
7274	Rossi	42
7432	Neri	54
9824	Verdi	45

Quadri

Matricola	Nome	Età
9297	Neri	33
7432	Neri	54
9824	Verdi	45

Laureati ∪ Quadri

Matricola	Nome	Età
7274	Rossi	42
7432	Neri	54
9824	Verdi	45
9297	Neri	33

Intersezione

Laureati

Matricola	Nome	Età
7274	Rossi	42
7432	Neri	54
9824	Verdi	45

Quadri

Matricola	Nome	Età
9297	Neri	33
7432	Neri	54
9824	Verdi	45

Laureati ∩ Quadri

Matricola	Nome	Età
7432	Neri	54
9824	Verdi	45

Differenza

Laureati

Matricola	Nome	Età
7274	Rossi	42
7432	Neri	54
9824	Verdi	45

Quadri

Matricola	Nome	Età
9297	Neri	33
7432	Neri	54
9824	Verdi	45

Laureati – Quadri

Matricola	Nome	Età
7274	Rossi	42

Un'unione sensata ma impossibile

Paternità

Padre	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

Maternità

Madre	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

Paternità ∪ Maternità

??

Ridenominazione

- operatore monadico (con un argomento)
- "modifica lo schema" lasciando inalterata l'istanza dell'operando

Paternità

Padre	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

$\text{REN}_{\text{Genitore} \leftarrow \text{Padre}}$ (Paternità)

Genitore	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

Ridenominazione

Sintassi

$$\text{REN}_{A1 \leftarrow B1,\ A2 \leftarrow B2,\ldots,\ An \leftarrow Bn}\ (\textit{Operando})$$

Semantica

La schema della relazione rappresentata da “Operando” viene modificato sostituendo al nome di attributo B1 il nome A1, al nome di attributo B2 il nome A2, … , e al nome di attributo Bn il nome An.

Nota

non ci devono essere duplicati negli attributi risultanti dalla ridenominazione

Paternità

Padre	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

$REN_{Genitore \leftarrow Padre}$ (Paternità)

Genitore	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

Maternità

Madre	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

$REN_{Genitore \leftarrow Madre}$ (Maternità)

Genitore	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

$REN_{Genitore \leftarrow Padre}$ **(Paternità)**

Genitore	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

$REN_{Genitore \leftarrow Madre}$ **(Maternità)**

Genitore	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

$REN_{Genitore \leftarrow Padre}$ **(Paternità)**

$\cup$

$REN_{Genitore \leftarrow Madre}$ **(Maternità)**

Genitore	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco
Eva	Abele
Eva	Set
Sara	Isacco

Impiegati

Cognome	Ufficio	Stipendio
Rossi	Roma	55
Neri	Milano	64

Operai

Cognome	Fabbrica	Salario
Bruni	Monza	45
Verdi	Latina	55

$$\text{REN}_{\text{Sede, Retribuzione} \leftarrow \text{Ufficio, Stipendio}}(\text{Impiegati}) \cup \text{REN}_{\text{Sede, Retribuzione} \leftarrow \text{Fabbrica, Salario}}(\text{Operai})$$

Cognome	Sede	Retribuzione
Rossi	Roma	55
Neri	Milano	64
Bruni	Monza	45
Verdi	Latina	55

Selezione

- operatore monadico
- produce un risultato che
  - ha lo stesso schema dell'operando
  - contiene un sottoinsieme delle ennuple dell'operando, quelle che soddisfano una condizione

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
9553	Milano	Milano	44
5698	Neri	Napoli	64

- impiegati che
  - guadagnano più di 50
  - guadagnano più di 50 e lavorano a Milano
  - hanno lo stesso nome della filiale presso cui lavorano

Selezione, sintassi e semantica

Sintassi:

SEL $_{Condizione}$ (*Operando*)

Condizione: espressione booleana (come quelle dei vincoli di ennupla)

Semantica

la relazione risultato ha gli stessi attributi dell'operando e contiene le ennuple dell'operando che soddisfano la condizione specificata

– impiegati che guadagnano più di 50

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
5698	Neri	Napoli	64

$$\mathbf{SEL_{Stipendio > 50}\ (Impiegati)}$$

– impiegati che guadagnano più di 50 e lavorano a Milano

Impiegati

Matricola	Cognome	Filiale	Stipendio
5998	Neri	Milano	64

$$\mathbf{SEL}_{\mathbf{Stipendio > 50\ AND\ Filiale = 'Milano'}}\ \mathbf{(Impiegati)}$$

– impiegati che hanno lo stesso nome della filiale presso cui lavorano

Impiegati

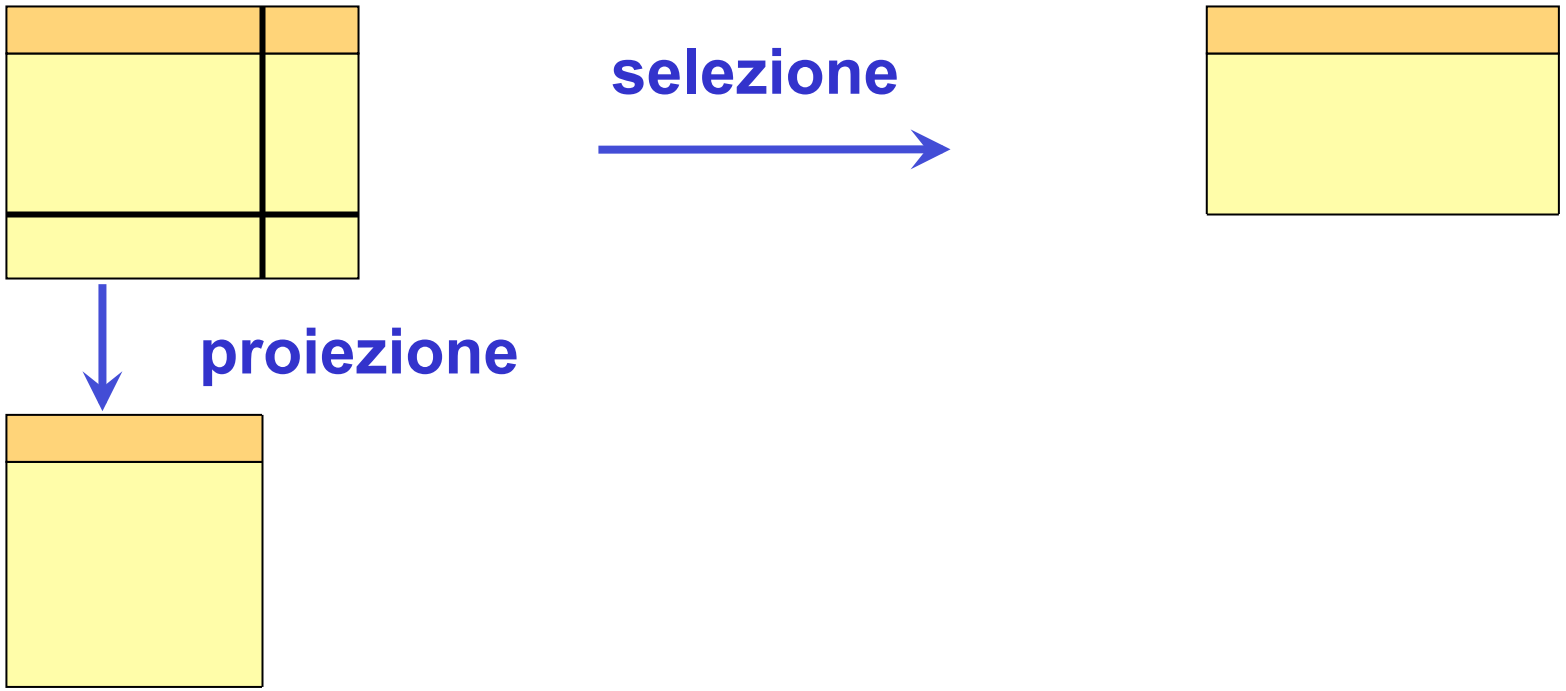
Matricola	Cognome	Filiale	Stipendio
9553	Milano	Milano	44

$$\mathbf{SEL}_{\mathbf{Cognome = Filiale}}\mathbf{(Impiegati)}$$

Selezione e proiezione

Sono due operatori "ortogonali"

- selezione:
  - decomposizione orizzontale
- proiezione:
  - decomposizione verticale



Proiezione

- operatore monadico
- produce un risultato che
  - ha parte degli attributi dell'operando
  - contiene ennuple cui contribuiscono tutte le ennuple dell'operando

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Neri	Napoli	55
5998	Neri	Milano	64
9553	Rossi	Roma	44
5698	Rossi	Roma	64

– per tutti gli impiegati:
  - matricola e cognome
  - cognome e filiale

Proiezione, sintassi e semantica

Sintassi

$$\text{PROJ}_{\text{ListaAttributi}}\ (\text{Operando})$$

Semantica

- la relazione risultato ha i soli attributi contenuti in ListaAttributi, e contiene le ennuple ottenute da tutte le ennuple dell'operando ristrette agli attributi nella lista

– matricola e cognome di tutti gli impiegati

Matricola	Cognome
7309	Neri
5998	Neri
9553	Rossi
5698	Rossi

PROJ $_{\text{Matricola, Cognome}}$ (Impiegati)

– cognome e filiale di tutti gli impiegati

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma

PROJ $_{\text{Cognome, Filiale}}$ (Impiegati)

Cardinalità delle proiezioni

- una proiezione
  - contiene al più tante ennuple quante l'operando
  - può contenerne di meno, a causa di eliminazione di duplicati

- Nota:

  se X è una superchiave di R, allora $\text{PROJ}_X(R)$ contiene esattamente tante ennuple quante R

Selezione e proiezione

- Combinando selezione e proiezione, possiamo estrarre interessanti informazioni da una relazione

matricola e cognome degli impiegati che guadagnano più di 50

Matricola	Cognome
7309	Rossi
5998	Neri
5698	Neri

$$\mathbf{PROJ_{Matricola,Cognome}\ (SEL_{Stipendio > 50}\ (Impiegati))}$$

Join

- combinando selezione e proiezione, possiamo estrarre informazioni da **una** relazione
- non possiamo però correlare informazioni presenti in relazioni diverse

- il **join** è l'operatore più interessante dell'algebra relazionale
- permette di correlare dati in relazioni diverse

Corsi e linguaggi di programmazione

- Ogni docente insegna uno o più corsi
- Nei corsi si insegnano i linguaggi di programmazione

1	Basi di dati
3	Reti
1	Prog. sw
2	Prog. sw

Basi di dati	SQL
Basi di dati	Java
Prog. sw	UML
Prog. sw	Java

- Quali docenti insegnano quali linguaggi?

1	SQL
1	UML
1	Java
2	UML
2	Java

Join naturale

- operatore binario (generalizzabile secondo quanto specificato più avanti)

- produce un risultato
  - sull'unione degli attributi degli operandi
  - con ennuple costruite ciascuna a partire da una ennupla di ognuno degli operandi

Join naturale: sintassi e semantica

- Ricordiamo che se X_1 e X_2 sono due insiemi, l'espressione X_1X_2 denota la loro unione

- Siano $R_1(X_1)$, $R_2(X_2)$ due schemi di relazioni

- R_1 JOIN R_2 è una relazione su X_1X_2 il cui insieme di ennuple è:

$$\{ t \text{ su } X_1X_2 \mid \text{esistono } t_1 \in R_1 \text{ e } t_2 \in R_2 \text{ tali che } t[X_1] = t_1 \text{ e } t[X_2] = t_2 \}$$

Join naturale: sintassi e semantica

- R_1 JOIN R_2 è una relazione su X_1X_2 il cui insieme di ennuple è:

  $\{ t \text{ su } X_1X_2 \mid \text{esistono } t_1 \in R_1 \text{ e } t_2 \in R_2$
  $\text{tali che } t[X_1] = t_1 \text{ e } t[X_2] = t_2 \}$

- Diciamo che $t_1 \in R_1$ e $t_2 \in R_2$ sono combinabili dal join naturale se $t[X_1 \cap X_2] = t_2[X_1 \cap X_2]$. Ne segue che ogni ennupla nel join tra R_1 ed R_2 proviene da due ennuple combinabili dal join. Ed in effetti $t_1 \in R_1$ e $t_2 \in R_2$ tali che $t[X_1 \cap X_2] = t_2[X_1 \cap X_2]$ si combinano per ottenere la ennupla t tale che $t[X_1] = t_1$ e $t[X_2] = t_2$

Join naturale: esempio

Docente	Corso
1	BD
2	PS
3	Reti
1	PS

Corso	Ling
BD	SQL
BD	Java
PS	Java
PS	UML

Docente	Corso	Ling
1	BD	SQL
1	BD	Java
1	PS	Java
1	PS	UML
2	PS	Java
2	PS	UML

Join completo

Un **join completo** è un join in cui ogni ennupla contribuisce al risultato. Questo è un esempio di join naturale completo:

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
A	Mori
B	Bruni

Impiegato	Reparto	Capo
Rossi	A	Mori
Neri	B	Bruni
Bianchi	B	Bruni

Un join naturale non completo

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
B	Mori
C	Bruni

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori

Un join vuoto

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
D	Mori
C	Bruni

Impiegato	Reparto	Capo

Un join completo, con (n x m) ennuple

Impiegato	Reparto
Rossi	B
Neri	B

Reparto	Capo
B	Mori
B	Bruni

Impiegato	Reparto	Capo
Rossi	B	Mori
Rossi	B	Bruni
Neri	B	Mori
Neri	B	Bruni

Cardinalità del join naturale

- $R_1(A,B)$, $R_2(B,C)$

- In generale, il join di R_1 e R_2 contiene un numero di ennuple compreso fra zero ed il prodotto di $|R_1|$ e $|R_2|$:

$$0 \leq |R_1 \text{ JOIN } R_2| \leq |R_1| \times |R_2|$$

- se il join coinvolge una chiave di R_2 (ovvero, B è chiave in R_2), allora il numero di ennuple è compreso fra zero e $|R_1|$:

$$0 \leq |R_1 \text{ JOIN } R_2| \leq |R_1|$$

- se il join coinvolge una chiave di R_2 (ovvero, B è chiave in R_2) ed esiste un vincolo di integrità referenziale fra B (in R_1) e R_2, allora il numero di ennuple è pari a $|R_1|$:

$$|R_1 \text{ JOIN } R_2| = |R_1|$$

Join e proiezioni

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
B	Mori
C	Bruni

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori

Impiegato	Reparto
Neri	B
Bianchi	B

Reparto	Capo
B	Mori

Proiezioni e join

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Bruni
Verdi	A	Bini

Impiegato	Reparto
Neri	B
Bianchi	B
Verdi	A

Reparto	Capo
B	Mori
B	Bruni
A	Bini

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Bruni
Neri	B	Bruni
Bianchi	B	Mori
Verdi	A	Bini

Join e proiezioni

- $R_1(X_1)$, $R_2(X_2)$

$$\text{PROJ}_{X_1}(R_1 \text{ JOIN } R_2) \subseteq R_1$$

- $R(X)$, $X = X_1 \cup X_2$

$$(\text{PROJ}_{X_1}(R)) \text{ JOIN } (\text{PROJ}_{X_2}(R)) \supseteq R$$

Prodotto cartesiano

- Ricordiamo la definizione di join naturale:

  R_1 JOIN R_2 è una relazione su X_1X_2 il cui insieme di ennuple è:

  $$\{ t \text{ su } X_1X_2 \mid \text{esistono } t_1 \in R_1 \text{ e } t_2 \in R_2 \text{ con } t[X_1] = t_1 \text{ e } t[X_2] = t_2 \}$$

  Da essa si evince che il join naturale su relazioni senza attributi in comune si riduce al prodotto cartesiano

- Il risultato contiene sempre un numero di ennuple pari al prodotto delle cardinalità degli operandi (le ennuple sono tutte combinabili)

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Codice	Capo
A	Mori
B	Bruni

Impiegati JOIN Reparti

Impiegato	Reparto	Codice	Capo
Rossi	A	A	Mori
Rossi	A	B	Bruni
Neri	B	A	Mori
Neri	B	B	Bruni
Bianchi	B	A	Mori
Bianchi	B	B	Bruni

Theta-join

- Supponiamo che R_1, R_2 non abbiano attributi in comune. Il prodotto cartesiano, in pratica, ha senso (quasi) solo se combinato con una selezione:

$$SEL_{condizione}\ (R_1\ JOIN\ R_2)$$

- La combinazione delle due operazioni viene chiamata **theta-join**, richiede che R_1 ed R_2 non abbiano attributi in comune, e viene indicata con

$$R_1\ JOIN_{condizione}\ R_2$$

- La condizione di selezione è spesso una congiunzione (AND) di atomi di confronto $A_1 \vartheta A_2$ dove ϑ (da cui deriva il nome) è uno degli operatori di confronto (=, >, <, …).
- Se il solo operatore di confronto usato nella condizione ϑ è l'uguaglianza (=), allora si parla di **equi-join**.

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Codice	Capo
A	Mori
B	Bruni

Impiegati $JOIN_{Reparto=Codice}$ Reparti

Impiegato	Reparto	Codice	Capo
Rossi	A	A	Mori
Neri	B	B	Bruni
Bianchi	B	B	Bruni

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Reparto	Capo
A	Mori
B	Bruni

Impiegati JOIN Reparti

Join naturale ed equi-join

Possiamo riesprimere un join naturale usando un equi-join.

Impiegati

Impiegato	Reparto

Reparti

Reparto	Capo

Join naturale: Impiegati JOIN Reparti

Equi-join:

$$\text{PROJ}_{\text{Impiegato,Reparto,Capo}}\left(\text{SEL}_{\text{Reparto=Codice}}\left(\text{Impiegati JOIN REN}_{\text{Codice} \leftarrow \text{Reparto}}(\text{Reparti})\right)\right)$$

Equi-join

prodotto cartesiano

Esempi

Impiegati

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45
5998	Bianchi	37	38
9553	Neri	42	35
5698	Bruni	43	42
4076	Mori	45	50
8123	Lupi	46	60

Supervisione

Impiegato	Capo
7309	5698
5998	5698
9553	4076
5698	4076
4076	8123

Trovare matricola, nome, età e stipendio degli impiegati che guadagnano più di 40 milioni.

$$\mathbf{SEL_{Stipendio>40}(Impiegati)}$$

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45
5698	Bruni	43	42
4076	Mori	45	50
8123	Lupi	46	60

Trovare matricola, nome ed età degli impiegati che guadagnano più di 40 milioni.

$$\mathbf{PROJ_{Matricola,\ Nome,\ Età}\ (SEL_{Stipendio>40}(Impiegati)\ )}$$

Matricola	Nome	Età
7309	Rossi	34
5698	Bruni	43
4076	Mori	45
8123	Lupi	46

Trovare le matricole dei capi degli impiegati che guadagnano più di 40 milioni.

Impiegati

Matricola	Nome	Età	Stipendio

Supervisione

Impiegato	Capo

$$\text{PROJ}_{\text{Capo}}\,(\text{Supervisione}\ \text{JOIN}_{\text{Impiegato=Matricola}}\,(\text{SEL}_{\text{Stipendio>40}}\,(\text{Impiegati})))$$

Trovare nome e stipendio dei capi degli impiegati che guadagnano più di 40 milioni.

Impiegati

Matricola	Nome	Età	Stipendio

Supervisione

Impiegato	Capo

$$\text{PROJ}_{\text{Nome,Stipendio}} (\text{Impiegati JOIN}_{\text{Matricola=Capo}} \text{PROJ}_{\text{Capo}}(\text{Supervisione JOIN}_{\text{Impiegato=Matricola}} (\text{SEL}_{\text{Stipendio>40}}(\text{Impiegati}))))$$

Trovare gli impiegati che guadagnano più del proprio capo, mostrando matricola, nome e stipendio dell'impiegato e del capo.

Impiegati

Matricola	Nome	Età	Stipendio

Supervisione

Impiegato	Capo

$$\text{PROJ}_{\text{Matr,Nome,Stip,MatrC,NomeC,StipC}}(\text{SEL}_{\text{Stipendio}>\text{StipC}}(\text{REN}_{\text{MatrC,NomeC,StipC,EtàC} \leftarrow \text{Matr,Nome,Stip,Età}}(\text{Impiegati}) \text{ JOIN}_{\text{MatrC=Capo}} (\text{Supervisione JOIN}_{\text{Impiegato=Matricola}} \text{Impiegati}))))$$

Trovare la matricola degli impiegati che **non** hanno un capo

Impiegati	Matricola	Nome	Età	Stipendio

Supervisione	Impiegato	Capo

$$\text{PROJ}_{\text{Matricola}}(\text{Impiegati}) - \text{REN}_{\text{Matricola} \leftarrow \text{Impiegato}}(\text{PROJ}_{\text{Impiegato}}(\text{Supervisione}))$$

Trovare matricola ed età degli impiegati che **non** hanno un capo

Impiegati

Matricola	Nome	Età	Stipendio

Supervisione

Impiegato	Capo

$$\text{PROJ}_{\text{Matricola, Età}}(\text{Impiegati} \text{ JOIN } (\text{PROJ}_{\text{Matricola}}(\text{Impiegati}) - \text{REN}_{\text{Matricola} \leftarrow \text{Impiegato}}(\text{PROJ}_{\text{Impiegato}}(\text{Supervisione}))))$$

Trovare le matricole dei capi i cui impiegati guadagnano **tutti** più di 40 milioni.

Impiegati

Matricola	Nome	Età	Stipendio

Supervisione

Impiegato	Capo

$$\text{PROJ}_{\text{Capo}}(\text{Supervisione}) - \text{PROJ}_{\text{Capo}}(\text{Supervisione JOIN}_{\text{Impiegato=Matricola}}(\text{SEL}_{\text{Stipendio} \leq 40}(\text{Impiegati})))$$

Il valore nullo nell'algebra relazionale

Una condizione in un'espressione nell'algebra è vera **solo per valori non nulli**

Esempio sulla selezione:

Impiegati

Matricola	Cognome	Filiale	Età
7309	Rossi	Roma	32
5998	Neri	Milano	45
9553	Bruni	Milano	NULL

$$\text{SEL}_{\text{Età} > 40}\ (\text{Impiegati})$$

Un risultato non desiderabile

$$\mathbf{SEL}_{\text{Età}>30}(\text{Persone}) \cup \mathbf{SEL}_{\text{Età}\leq 30}(\text{Persone}) \neq \text{Persone}$$

Perché? Perché le selezioni vengono valutate separatamente!

Ma anche

$$\mathbf{SEL}_{\text{Età}>30 \vee \text{Età}\leq 30}(\text{Persone}) \neq \text{Persone}$$

Perché? Perché anche le condizioni atomiche vengono valutate separatamente!

Soluzione

- per riferirsi ai valori nulli esistono forme apposite di condizioni:

  IS NULL

  IS NOT NULL

- Esempio sulla selezione:

  $$SEL_{Età > 40}(Impiegati)$$

  la condizione atomica è vera solo per valori non nulli

- Si potrebbe usare (ma non serve) una "logica a tre valori" (vero, falso, sconosciuto)

Quindi:

$$\mathbf{SEL}_{\text{Età}>30}(\text{Persone}) \cup \mathbf{SEL}_{\text{Età}\leq 30}(\text{Persone}) \cup \mathbf{SEL}_{\text{Età IS NULL}}(\text{Persone})$$

$$=$$

$$\mathbf{SEL}_{\text{Età}>30 \vee \text{Età}\leq 30 \vee \text{Età IS NULL}}(\text{Persone})$$

$$=$$

$$\text{Persone}$$

Impiegati

Matricola	Cognome	Filiale	Età
5998	Neri	Milano	45
9553	Bruni	Milano	NULL

SEL $_{\text{(Età > 40) OR (Età IS NULL)}}$ **(Impiegati)**

Join: un'osservazione

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
B	Mori
C	Bruni

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori

Come visto prima, alcune ennuple possono non contribuire al risultato nel join: vengono "tagliate fuori"

Join esterno

- Il **join esterno** estende, con valori nulli, le ennuple che verrebbero tagliate fuori da un join (interno)

- esiste in tre versioni:
  - **sinistro**: mantiene tutte le ennuple del primo operando, estendendole con valori nulli, se necessario
  - **destro**: ... del secondo operando ...
  - **completo**: … di entrambi gli operandi ...

Join esterno sinistro

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Reparto	Capo
B	Mori
C	Bruni

Impiegati $\text{JOIN}_{\text{LEFT}}$ Reparti

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori
Rossi	A	NULL

Join esterno destro

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Reparto	Capo
B	Mori
C	Bruni

Impiegati $\text{JOIN}_{\text{RIGHT}}$ Reparti

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori
NULL	C	Bruni

Join esterno completo

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Reparto	Capo
B	Mori
C	Bruni

Impiegati $\text{JOIN}_{\text{FULL}}$ Reparti

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori
Rossi	A	NULL
NULL	C	Bruni

Equivalenza di espressioni

- Due espressioni sono **equivalenti** se producono lo stesso risultato qualunque sia l'istanza della base di dati sulla quale vengono valutate

- L'equivalenza è importante in pratica perché i DBMS cercano di eseguire espressioni equivalenti a quelle date, ma meno "costose" della espressione orginaria

Un'equivalenza importante

- Effettuare le selezioni il prima possibile (**push selections**)

  Esempio: se A è attributo di R_2

  $$\text{SEL}_{A=10}(R_1 \text{ JOIN } R_2) = R_1 \text{ JOIN } \text{SEL}_{A=10}(R_2)$$

- Le selezioni tipicamente riducono in modo significativo la dimensione del risultato intermedio (e quindi il costo dell'operazione)