Algebra Relazionale

foglia@iet.unipi.it

Sommario

- Introduzione all’algebra relazionale
- I costrutti principali dell’algebra relazionale
- Esempi
- Esercitazione

Basi di dati – Ciclo di vita

- Finita la fase di progettazione concettuale che produce un Diagramma E-R avviene la progettazione logica della base di dati che produce lo schema logico relazionale (descrizione delle relazioni/tabelle e attributi con i vincoli ad esse associati)
- Finita la progettazione, la base di dati con le sue tabelle è pronta per ospitare istanze di dati
- Dopo che la base di dati è stata riempita di dati, gli utenti (programmatori/utenti finali) sono pronti a fare interrogazioni e manipolare i dati

A tal fine gli utenti (o applicazioni) ricorrono ai linguaggi di interrogazione e manipolazione per poter accedere, manipolare, ricavare informazioni da tali dati

Interrogazioni e aggiornamento (DML)

- Interrogazione: data una istanza di una base di dati, produce una relazione
- Aggiornamento/manipolazione: data una istanza di una base di dati, produce una nuova istanza sullo stesso schema
- Si utilizzano
  - Direttamente al «terminale»
  - Tramite istruzioni «embedded» negli applicativi

- I DDL invece agiscono sullo schema della base di dati

Linguaggi di interrogazione per basi di dati relazionali

- Un linguaggio di manipolazione, o DML, permette di interrogare e modificare istanze di Basi di Dati
- Esistono due tipologie di linguaggi:
  - **Procedurali**, in cui si descrive al sistema il dato che l'utente vuole attraverso l'elenco delle procedure da eseguire per ottenerlo
  - **Non Procedurali (dichiaritivi)**, in cui invece vengono enunciate direttamente le proprietà del risultato senza alcun accenno alle procedure da eseguire per produrlo

- Il linguaggio DML più usato è il linguaggio **SQL** (Structured Query Language) che è a carattere **misto**, procedurale e non

Algebra Relazionale

- **Algebra relazionale** è un linguaggio procedurale formale di tipo algebrico i cui operandi sono relazioni
- Questo linguaggio non è usato nelle implementazioni dei vari DBMS ma definisce in maniera semplice tutte le operazioni tipiche dei diversi linguaggi di interrogazione
- Da un punto di vista didattico l'algebra relazionale è utile perché essendo svincolata dai "dettagli implementativi" dell'SQL (o di altri linguaggi), permette di comprendere rapidamente la tecnica d'uso dei linguaggi di interrogazione per basi di dati relazionali

Operatori

- L'Algebra Relazionale è basata su un **insieme di operatori** che accettano una o due tabelle (relazioni) come **input** e producono una tabella (relazione) in **output**
- Tramite gli operatori definiti si descrivono le procedure di **interrogazione** per :
  - prendere delle tabelle (relazioni) di input
  - generare delle tabelle (relazioni) di output secondo opportuni criteri
- **Tali operatori possono essere composti tra di loro al fine di definire interrogazioni complesse**

Operatori di base

- Operatori di arità 1, specifici sulle relazioni:
  - **selezione** (notazione σ o SEL)
  - **proiezione** (notazione Π o PROJ)
  - **ridenominazione** (notazione ρ o REN)
- Operatori di arità 2, derivati dagli operatori insiemistici tradizionali:
  - **unione** (notazione U)
  - **differenza insiemistica** (notazione -)
  - **intersezione** (notazione ∩ o Intersect)
  - **prodotto cartesiano** (notazione X*)*
- Operazioni derivate dalle precedenti:
  - **Join** (notazione >< oppure ⊳⊲ o JOIN)
- Gli operatori si possono comporre
- Gli operandi sono o (nomi di) relazioni del BD o espressioni

Selezione

Notazione (σ o SEL)

- la selezione è una operazione unaria, poiché accetta come argomento una sola tabella
- seleziona le tuple di una relazione che soddisfano un predicato producendo un sottoinsieme delle stesse
  - Formula proposizionale
- lo schema della relazione risultato è lo stesso di quella di origine
- il predicato è costituito dal nome di un attributo, da un operatore, e da un altro argomento che può essere un attributo o un valore costante
- più predicati possono essere uniti con OR, AND e NOT
- operatori sono
  - =, >, >, >=, <=, ≠ o in alternativa # (diverso) o <>

$$\sigma_{\text{Predicato}}(\text{relazione})$$

osservazione

- Affinché il predicato sia valido
  - Gli attributi del predicato devono appartenere allo schema della relazione
  - L'operatore deve avere senso nel domino degli attributi
  - Se si utilizzano costanti, queste devono essere compatibili con il dominio

Selezione – Esempio

Data la Tabella con schema:
Impiegati (Matricola, Cognome, Filiale, Stipendio)

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
9553	Milano	Napoli	44
5698	Bianchi	Torino	64
2568	Rossi	Roma	55

Istanza della Tabella Impiegati

Selezione (σ)
Vogliamo trovare gli impiegati che:

- guadagnano più di 62
- guadagnano più di 62 e lavorano a Milano

Selezione – Esempio

Esempio operazione di Selezione:

- guadagnano più di 62

  σ Stipendio>62 (Impiegati)

Matricola	Cognome	Filiale	Stipendio
5998	Neri	Milano	64
5698	Bianchi	Torino	64

- guadagnano più di 62 e lavorano a Milano

  σ Stipendio>62 AND Filiale = 'Milano' (Impiegati)

Matricola	Cognome	Filiale	Stipendio
5998	Neri	Milano	64

Altro esempio

Data la Tabella Cittadini schema:
Cittadini (Cognome, Nome, CittaDiNascita, Residenza)

Cognome	Nome	CittaDiNascita	Residenza
Rossi	Mario	Roma	Milano
Neri	Luca	Roma	Roma
Verdi	Nico	Firenze	Firenze
Rossi	Marco	Napoli	Firenze

Trovare i cittadini che risiedono nella città di nascita:

soluzione

Trovare i cittadini che risiedono nella città di nascita:

σ CittaDiNascita=Residenza (Cittadini)

Cognome	Nome	CittaDiNascita	Residenza
Neri	Luca	Roma	Roma
Verdi	Nico	Firenze	Firenze

Proiezione

Notazione (Π o PROJ)

- anche la proiezione è una operazione unaria
- data una relazione, la sua proiezione su un dato insieme di attributi è costituita dalla tabella generata dagli attributi specificati, contenente tutte le tuple della tabella di partenza
- eventuali tuple (righe) duplicate vengono ignorate

$$\Pi \text{ Lista Attibuti } (\text{relazione})$$

Proiezione – Esempio

Data la Tabella con schema:
Impiegati (Matricola, Cognome, Filiale, Stipendio)

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
9553	Milano	Napoli	44
5698	Bianchi	Torino	64
2568	Rossi	Roma	55

Istanza della Tabella Impiegati

Proiezione (Π)
per tutti gli impiegati trovare:

- matricola e cognome
- cognome e filiale

Proiezione – Esempio

Esempio operazione di Proiezione:

- matricola e cognome di tutti gli impiegati

$$\Pi_{\text{matricola, cognome}}(\text{Impiegati})$$

Matricola	Cognome
7309	Rossi
5998	Neri
9553	Milano
5698	Bianchi
2568	Rossi

- cognome e filiale di tutti gli impiegati

$$\Pi_{\text{cognome, filiale}}(\text{Impiegati})$$

- la tupla in rosso non fa parte del risultato

Cognome	Filiale
Rossi	Roma
Neri	Milano
Milano	Napoli
Bianchi	Torino
Rossi	Roma

Cardinalità delle proiezioni

- una proiezione
  - contiene al più tante ennuple quante l'operando
  - può contenerne di meno

- se X è una superchiave di R, allora $\Pi_X(R)$ contiene esattamente tante ennuple quante R

Combinazione Selezione – Proiezione

- Dal momento che selezione e proiezione richiedono come argomenti una tabella (una relazione) e che producono come risultato una tabella, posso combinare le due operazioni per ottenere query complesse (e più espressive)

- Per esempio voglio sapere matricola e cognome degli impiegati che guadagnano più di 62, nella relazione Impiegati di schema (quello precedente)

$$\Pi_{\text{matricola, cognome}}\left(\sigma_{\text{Stipendio>62}}(\text{Impiegati})\right)$$

esempio

Π matricola, cognome $(\sigma$ Stipendio>62 (Impiegati)$)$

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
9553	Milano	Napoli	44
5698	Bianchi	Torino	64
2568	Rossi	Roma	55

Ottengo il risultato:

Matricola	Cognome
5998	Neri
5698	Bianchi

Ordine delle operazioni

Esempio: selezione e proiezione

$$\Pi_{\text{matricola, cognome}}\left(\sigma_{\text{stipendio>62}}(\text{Impiegati})\right)$$

Ottengo il risultato:

Matricola	Cognome
5998	Neri
5698	Bianchi

e se scambiassi l'ordine di selezione e proiezione?

$$\sigma_{\text{stipendio>62}}\left(\Pi_{\text{matricola, cognome}}(\text{Impiegati})\right)$$

È un **errore**, poiché la tabella risultante dall'operazione di proiezione ha come schema gli attributi matricola e cognome, e su di esso non posso effettuare la selezione con predicato sull'attributo stipendio

Ridenominazione

Notazione (ρ o REN)

- A volte in preparazione all'esecuzione di una interrogazione o in seguito ad una sua esecuzione si ha bisogno di rinominare gli attributi di una relazione
- A tal fine l'operatore di ridenominazione che permette di ottenere una nuova tabella con i nuovi nomi per gli attributi modificati e che ha le stesse tuple della tabella originale

$$\rho\ \text{NomeNuovo} \leftarrow \text{Nome}(\text{relazione})$$

Ridenominazione – Esempio

Persone

CF	Cognome	Nome
BNCGRG78F21A	Bianchi	Mario
RSSGRG78F23X	Rossi	Paolo

$\rho_{(CodiceFiscale \leftarrow CF)}(Persone)$

CodiceFiscale	Cognome	Nome
BNCGRG78F21A	Bianchi	Mario
RSSGRG78F23X	Rossi	Paolo

Ridenominazione per più attributi

Impiegati

Cognome	Ufficio	Stipendio
Rossi	Roma	55
Neri	Milano	64

ρ (Sede, Retribuzione ← Ufficio, Stipendio) (Impiegati)

Impiegati

Cognome	Sede	Retribuzione
Rossi	Roma	55
Neri	Milano	64

Unione, Intersezione e Differenza

- Le Relazioni sono insiemi, quindi possiamo applicare gli operatori tra insiemi

- Tuttavia, vogliamo che il risultato siano relazioni (cioè insiemi di tuple omogenee, ossia definite sugli stessi attributi)
  - È quindi significativo applicare l'unione, intersezione e differenza solo a coppie di relazioni definite su gli stessi attributi
  - Nel caso in cui alcuni attributi siano omogenei ma abbiano nomi diversi l'operazione di ridenominazione può essere applicata

Unione

- l'unione fra due tabelle è rappresentata da una tabella costituita dall'unione "matematica" delle due tabelle
  - L' unione fra due tabelle A e B è una tabella che contiene le tuple che sono presenti in A, e in B
  - tuple replicate sono considerate un'unica tupla e fuse in un'unica tupla
- affinché l'unione abbia senso, è necessario che
  - le due tabelle abbiano lo **stesso numero di attributi**
  - i tipi degli attributi corrispondenti siano uguali

- Se le relazioni hanno **nomi di attributo diversi**, nella relazione risultato per convenzione si usano i nomi della prima relazione (in questo caso A), a meno di opportune ridenominazioni
- Il **grado** della relazione risultato è uguale al grado delle relazioni operandi
- Se il numero degli attributi delle due relazioni (tabelle) non è uguale, si genera un errore

Unione – Esempio

- si hanno le seguenti tabelle:
  - depositi(NomeFiliale, NumeroDeposito, NomeCliente, Saldo)
  - prestiti(NomeFiliale, NumeroPrestito, NomeCliente, Saldo)

- si vogliono trovare tutti i clienti della filiale di Roma che hanno un deposito o un prestito o entrambi

$$\Pi_{\text{NomeCliente}}(\sigma_{\text{NomeFiliale=Roma}}(\text{Depositi}))$$

$$\cup$$

$$\Pi_{\text{NomeCliente}}(\sigma_{\text{NomeFiliale=Roma}}(\text{Prestiti}))$$

Unione – Esempio

Depositi

Filiale	NumeroDeposito	Cliente	Saldo
Milano	010025	Rossi	1000
Roma	012025	Bianchi	15000
Roma	010525	Neri	200
Pisa	013025	Torino	100

Prestiti

Filiale	NumeroPrestito	Cliente	Saldo
Roma	010625	Rossi	1500
Milano	017025	Bianchi	3000

Π Cliente (σ Filiale=Roma(Depositi))

Cliente
Bianchi
Neri

Π Cliente (σ Filiale=Roma(Prestiti))

Cliente
Rossi

Π Cliente (σ Filiale=Roma(Depositi)) **U** Π Cliente (σ Filiale=Roma(Prestiti))

Cliente
Rossi
Bianchi
Neri

Unione – altro esempio

Trovare le città in cui i clienti hanno depositi e/o prestiti

Depositi

Filiale	NumeroDeposito	Cliente	Saldo
Milano	010025	Rossi	1000
Roma	012025	Bianchi	15000
Roma	010525	Neri	200
Pisa	013025	Torino	100

Prestiti

Filiale	NumeroPrestito	Cliente	Saldo
Roma	010625	Rossi	1500
Milano	017025	Bianchi	3000

Π Filiale (Depositi)

Filiale
Milano
Roma
Pisa

Roma non è duplicata

Π Filiale (Prestiti)

Filiale
Roma
Milano

Π Filiale Depositi) **U** Π Filiale Prestiti)

Filiale
Milano
Roma
Pisa

Roma non è duplicata

Differenza

- La differenza fra due tabelle A e B è una tabella che contiene le tuple che sono presenti in A ma non in B
- La differenza (come l'unione) può essere eseguita solo se le relazioni hanno lo stesso grado e gli attributi sono compatibili
- Il grado della relazione risultato è uguale al grado delle relazioni operandi

Differenza – Esempio

Esempio: Trovare i clienti che hanno un deposito ma non un prestito nella filiale di Roma

$$\Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Depositi})) - \Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Prestiti}))$$

Depositi

Filiale	NumeroDeposito	Cliente	Saldo
Milano	010025	Rossi	1000
Roma	012025	Bianchi	15000
Roma	010525	Neri	200
Pisa	013025	Torino	100

Prestiti

Filiale	NumeroPrestito	Cliente	Saldo
Roma	010625	Rossi	1500
Milano	017025	Bianchi	3000
Roma	018221	Neri	10000

$$\Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Depositi}))$$

Cliente
Bianchi
Neri

$$\Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Prestiti}))$$

Cliente
Rossi
Neri

$$\Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Depositi})) - \Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Prestiti}))$$

Cliente
Bianchi

Intersezione

- L' intersezione fra due tabelle A∩B è una tabella che contiene le tuple che sono presenti in A e in B
  - **Valgono per numero e tipo degli attributi, le stesse considerazioni di unione e differenza**
- **Esempio**: trovare i clienti che hanno sia un conto che un prestito

$$\Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Depositi})) \cap \Pi_{\text{Cliente}}(\sigma_{\text{Filiale=Roma}}(\text{Prestiti}))$$

Intersezione – Esempio

Depositi

Filiale	NumeroDeposito	Cliente	Saldo
Milano	010025	Rossi	1000
Roma	012025	Bianchi	15000
Roma	010525	Neri	200
Pisa	013025	Torino	100

Prestiti

Filiale	NumeroPrestito	Cliente	Saldo
Roma	010625	Neri	1500
Milano	017025	Bianchi	3000

Π Cliente (σ Filiale=Roma(Depositi))

Cliente
Bianchi
Neri

Π Cliente (σ Filiale=Roma(Prestiti))

Cliente
Neri

Π Cliente (σ Filiale=Roma(Depositi)) $\cap$ Π Cliente (σ Filiale=Roma(Prestiti))

Cliente
Neri

Esempio con l'operatore di renaming

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

Si vuole sapere il cognome, la sede e la retribuzione di tutti i dipendenti

Prodotto Cartesiano

- Il prodotto cartesiano fra due tabelle è una tabella con schema la somma degli schemi, se due attributi sono uguali questi sono ripetuti
- Le tuple della tabella sono il risultato del prodotto cartesiano dei suoi elementi, ossia da tutte le coppie possibili composte dagli elementi appartenenti alle due relazioni

$$R1 \times R2$$

Prodotto Cartesiano – Esempio

Clienti(NomeCliente, Via, Città, ID),

NomeCliente	Via	Città	ID
Rossi	Pardi	Pisa	1
Bianchi	Roma	Pisa	2
Blu	Italia	Napoli	2
Verdi	Foscolo	Torino	2

Banchieri(ID, NomeBanchiere)

ID	NomeBanchiere
1	Cuccia
2	Visco

R1 X R2
Clienti x Banchieri
Clienti x Banchieri(NomeCliente,Via,Città, ID, B.ID,NomeBanchiere)

NomeCliente	Via	Città	ID	B.ID	NomeBanchiere
Rossi	Pardi	Pisa	1	1	Cuccia
Rossi	Pardi	Pisa	1	2	Visco
Bianchi	Roma	Pisa	2	1	Cuccia
Bianchi	Roma	Pisa	2	2	Visco
Blu	Italia	Napoli	2	1	Cuccia
Blu	Italia	Napoli	2	2	Visco
Verdi	Foscolo	Torino	2	1	Cuccia
Verdi	Foscolo	Torino	2	2	Visco

NOTA: l'attributo ID **è ripetuto due volte**. Per questo l'operazione produce una "unione non matematica" degli schemi.
NOTA: Chiaramente non tutte le tuple risultanti hanno un significato reale!
NOTA: B.id viene usato per discriminare il valore da id -> si usa la convenzione nome_relazione.nome_attr o il risultato di un rename

Esempi di utilizzo del prodotto cartesiano

- supponiamo, dati gli schemi clienti e banchieri precedenti, di voler saper la via e la città in cui abitano tutti i clienti di Cuccia

- per ottenere tale informazione non basta solo la tabella banchieri, perché associa ad un cliente il relativo banchiere, né la tabella clienti da sola, perché non ha il banchiere

- occorre allora combinare le due tabelle tramite il prodotto cartesiano, perché nell'insieme delle tabelle è contenuta l'informazione che serve

Clienti x Banchieri

Esempi di utilizzo del prodotto cartesiano

- A questo punto, nella tabella risultante, si ha tutta l'informazione che serve (o meglio, di più)
- occorre estrarre solo le tuple significative
  - (clienti.ID=banchieri.ID)
- si selezionano le tuple che hanno NomeBanchiere=Cuccia
- infine si fa una proiezione per avere il risultato

$$\sigma_{\text{Clienti.ID=Banchieri.ID}} (\text{Clienti} \times \text{Banchieri})$$

$$\Pi_{\text{via, città}} (\sigma_{\text{NomeBanchiere=Cuccia}} (\sigma_{\text{Clienti.ID=Banchieri.ID}} (\text{Clienti} \times \text{Banchieri})))$$

Cosa succede:

Clienti X Banchieri

NomeCliente	Via	Città	ID	B.ID	NomeBanchiere
Rossi	Pardi	Pisa	1	1	Cuccia
Rossi	Pardi	Pisa	1	2	Visco
Bianchi	Roma	Pisa	2	1	Cuccia
Bianchi	Roma	Pisa	2	2	Visco
Blu	Italia	Napoli	2	1	Cuccia
Blu	Italia	Napoli	2	2	Visco
Verdi	Foscolo	Torino	2	1	Cuccia
Verdi	Foscolo	Torino	2	2	Visco

$\sigma_{\text{Clienti.ID=Banchieri.ID}}$ (Clienti X Banchieri)

NomeCliente	Via	Città	ID	B.ID	NomeBanchiere
Rossi	Pardi	Pisa	1	1	Cuccia
Bianchi	Roma	Pisa	2	2	Visco
Blu	Italia	Napoli	2	2	Visco
Verdi	Foscolo	Torino	2	2	Visco

$\Pi_{\text{via, città}}$ ($\sigma_{\text{NomeBanchiere=Cuccia}}$($\sigma_{\text{Clienti.ID=Banchieri.ID}}$ (Clienti X Banchieri)))

Via	Città
Pardi	Pisa

Ricapitolando: scoprire dati correlati in più query

Una volta formulata la query:

- si cercano le tabelle (relazioni) in cui è contenuta l'informazione che serve
- si effettua l'operazione di prodotto cartesiano se l'informazione è presente in più tabelle (relazioni) che vanno concatenate
- si scartano le tuple non significative con una selezione sugli attributi a comune
- si fanno selezioni e proiezioni per rispondere alla query

osservazioni

- Schema di R1XR2
  - Se X è lo schema di R1, Y lo schema di R2
  - R1 X R2 ha schema la giustapposizione degli schemi
  - Attributi presenti in entrambi gli schemi vengono ripetuti
- Cardinalità
  - Se R1 ed R2 hanno cardinalità N1 ed N2
  - R1 X R2 ha cardinalità N1*N2
  - In generale è una operazione molto dispendiosa

Join

- E’ l’operatore più caratteristico dell’algebra relazionale, in quanto è quello che **permette di correlare dati contenuti in relazioni diverse confrontando i valori comuni contenuti in esse**
- Esistono diverse varianti di tale operatore comunque riconducibili l’una con l’altra:
  - Join Naturale (Natural Join)
  - Join Esterni (Outer Join)
  - Theta Join ed Equi Join (inner Join)

Join Naturale

- **Notazione** $\triangleright\triangleleft$ **oppure** $\rhd\lhd$ oppure JOIN
- Il Join naturale è un operatore binario (la sua definizione può essere facilmente estesa a più relazioni) che correla dati in relazioni diverse sulla base dei valori uguali in attributi con lo stesso nome
- La relazione risultante è una tabella che ha come attributi l'unione degli attributi delle tabelle iniziali e contiene solamente le tuple che hanno valori uguali negli attributi a comune
- Il Join può essere ottenuto tramite un prodotto cartesiano e una selezione imponendo l'uguaglianza su tutti gli attributi in comune:

$$\sigma_{C.AttrComune=B.AttrComune}\ (C \times B)$$

Join Naturale – Esempio

Join naturale:

Voli(Codice, Data, Comandante)

Codice	Data	Comandante
AZ427	21/07/2001	Bianchi
AZ427	23/07/2001	Rossi
TW056	21/07/2001	Smith

Linee(Codice, Partenza, Arrivo)

Codice	Partenza	Arrivo
AZ427	FCO	JFK
TW056	LAX	FCO

Voli $\triangleright\!\triangleleft$ Linee

Codice	Data	Comandante	Partenza	Arrivo
AZ427	21/07/2001	Bianchi	FCO	JFK
AZ427	23/07/2001	Rossi	FCO	JFK
TW056	21/07/2001	Smith	LAX	FCO

Altro esempio

Clienti(NomeCliente, Via, Città, ID),

NomeCliente	Via	Città	ID
Rossi	Pardi	Pisa	1
Bianchi	Roma	Pisa	2
Blu	Italia	Napoli	2
Verdi	Foscolo	Torino	2

Banchieri(ID, NomeBanchiere)

ID	NomeBanchiere
1	Cuccia
2	Visco

Clienti ⊳⊲ Banchieri

NomeCliente	Via	Città	ID	NomeBanchiere
Rossi	Pardi	Pisa	1	Cuccia
Bianchi	Roma	Pisa	2	Visco
Blu	Italia	Napoli	2	Visco
Verdi	Foscolo	Torino	2	Visco

Altro esempio: una query

- come rispondere ora alle query «trovare la via e la città in cui abitano tutti i clienti di Cuccia?»

Clienti ⊳⊲ Banchieri

NomeCliente	Via	Città	ID	NomeBanchiere
Rossi	Pardi	Pisa	1	Cuccia
Bianchi	Roma	Pisa	2	Visco
Blu	Italia	Napoli	2	Visco
Verdi	Foscolo	Torino	2	Visco

$$\Pi_{\text{via, città}} (\sigma_{\text{NomeBanchiere=Cuccia}} (\text{Clienti} \rhd\lhd \text{Banchieri}))$$

Join completi e incompleti

- Nell'esempio considerato ciascuna tupla degli operandi contribuisce ad almeno una tupla del risultato (perché ciascuna tupla di una delle tabelle ha almeno una tupla corrispondente nell'altra con gli attributi a comune uguali)
- Se questa condizione è verificata il Join si dice ***completo***
- Questa proprietà non è sempre verificata, in tal caso le tuple per cui l'altra relazione non contiene tuple con gli stessi valore sugli attributi in comune non verranno riportate nel risultato finale
- Queste tuple si chiamano tuple dangling (dondolanti) e il Join è di tipo incompleto

Un join naturale completo con NxM tuple (come il prodotto cartesiano)

Impiegato	Reparto
Rossi	B
Verdi	B

Reparto	Capo
B	Mori
B	Tori

Impiegato	Reparto	Capo
Rossi	B	Mori
Rossi	B	Tori
Verdi	B	Mori
Verdi	B	Tori

Join Incompleto – Esempio

Join naturale incompleto:

R1

Impiegato	Reparto
Rossi	Vendite
Neri	Produzione
Bianchi	Produzione

R2

Capo	Reparto
Bruni	Acquisti
Mori	Produzione

$R1 \rhd\lhd R2$

Impiegato	Reparto	Capo
Neri	Produzione	Mori
Bianchi	Produzione	Mori

Join Esterno

- La caratteristica di eliminare le tuple senza corrispondenza nel Join naturale *può essere un comportamento indesiderato* in alcuni casi dato che questa pratica porta ad omettere informazioni potenzialmente rilevanti
- Al fine di ovviare a tale problema è stata definita una variante del Join chiamata **Join Esterno** (o outer join) che *prevede che tutte le tuple diano un contributo al risultato eventualmente estese con valori null ove non vi siano controparti opportune*
- Esistono tre sottovarianti, il join esterno sinistro, destro o completo a seconda che vengano estese tutte le tuple del primo operando, del secondo o di entrambi

Join Esterno – Esempio

R1

Impiegato	Reparto
Rossi	Vendite
Neri	Produzione
Bianchi	Produzione

R2

Capo	Reparto
Bruni	Acquisti
Mori	Produzione

$R1 \; ⟕ \; R2$

Impiegato	Reparto	Capo
Rossi	Vendite	NULL
Neri	Produzione	Mori
Bianchi	Produzione	Mori

$R1 \; ⟖ \; R2$

Impiegato	Reparto	Capo
Neri	Produzione	Mori
Bianchi	Produzione	Mori
NULL	Acquisti	Bruni

$R1 \; ⟗ \; R2$

Impiegato	Reparto	Capo
Rossi	Vendite	NULL
Neri	Produzione	Mori
Bianchi	Produzione	Mori
NULL	Acquisti	Bruni

Prodotto Cartesiano

- Nel caso in cui gli *insiemi di attributi degli operandi siano disgiunti* il Join Naturale diventa un prodotto cartesiano

- Tale operazione produce una tabella con schema la somma degli schemi e un insieme di tuple prodotto dalla concatenazione delle tuple della prima relazione e tuple della seconda

Prodotto Cartesiano – Esempio

Impiegati

Impiegato	Progetto
Rossi	A
Neri	A
Neri	B

Progetti

Codice	Nome
A	Venere
B	Marte

R1 ⋈ R2

Impiegato	Progetto	Codice	Nome
Rossi	A	A	Venere
Neri	A	A	Venere
Neri	B	A	Venere
Rossi	A	B	Marte
Neri	A	B	Marte
Neri	B	B	Marte

Theta-join e Equi-join (inner join)

- Nel caso in cui il Join si riduca ad un prodotto cartesiano questo solitamente viene seguito da una selezione (solitamente per eliminare le tuple che non hanno un significato)
- Per questa ragione è definito un operatore derivato dal prodotto cartesiano, il **theta-join** che permette l'esecuzione del prodotto cartesiano e della selezione sul risultato con un'unica operazione:

$$R1 \ \triangleright\!\triangleleft_{OP} R2 = \sigma_{OP}(R1 \times R2)$$

- OP è una condizione di selezione
- Nel caso in cui OP sia composta solo da uguaglianze l'operazione prende il nome di **Equi-join**

Equi-join – Esempio

Impiegati

Impiegato	Progetto
Rossi	A
Neri	A
Neri	B

Progetti

Codice	Nome
A	Venere
B	Marte

$$\text{Impiegati} \triangleright\!\triangleleft_{\text{Progetto=Codice}} \text{Progetti}$$

Impiegato	Progetto	Codice	Nome
Rossi	A	A	Venere
Neri	A	A	Venere
Neri	B	B	Marte

Esempio: trovare gli impiegati che partecipano al progetto venere

$$\pi_{\text{impiegato}}(\sigma_{\text{nome = venere}}(\text{Impiegati} \triangleright\!\triangleleft_{\text{Progetto=Codice}} \text{Progetti}))$$

Impiegato
Rossi
Neri

osservazione

- In generale, il mettere in relazione due tabelle avviene tramite valori, e non tramite il nome degli attributi
  - Gli attributi non devono avere lo stesso nome
  - Ma se hanno valori uguali, esprimono la relazione

- Per questo, in generale si usa il **theta-join** o **l' Equi-join,** più che il natural join
  - Alcuni DBMS relazionali moderni però supportano entrambi

viste

- In algebra relazionale il risultato di una interrogazione è a sua volta una relazione

- A tale relazione può essere assegnato un nome
  - E' una relazione derivata, in quanto il suo contenuto dipende dalle relazioni da cui è stata generata

- A tale relazione derivata o virtuale si da il nome di vista

- Hanno diversi vantaggi, fra cui la possibilità di semplificare la scrittura di interrogazioni complesse

Impiegati

id	impiegato	progetto
1	Rossi	A
2	Neri	A
3	Neri	B
4	Bianchi	B

esempio

- Trovare il nome dei progetti a cui sono assegnati gli impegati di nome Neri e Bianchi

$$\boldsymbol{\Pi}_{progetto}\left(\sigma_{impiegato=neri}\ (impiegati)\right) \cup \boldsymbol{\Pi}_{progetto}\left(\sigma_{impiegato=bianchi}\ (impiegati)\right)$$

- In alternativa con le viste

$$S1 = \boldsymbol{\Pi}_{progetto}\left(\sigma_{impiegato=neri}\ (impiegati)\right)$$

$$T1 = \boldsymbol{\Pi}_{progetto}\left(\sigma_{impiegato=bianchi}\ (impiegati)\right)$$

$$S1 \cup T1$$

Join e ridenominazione

- L’operazione di join e quella di ridenominazione possono essere usate per svolgere operazioni di ricerca complesse all’interno di una stessa tabella
- In particolare la ridenominazione permette di realizzare il cosiddetto **self-join**, il join di una tabella su se stessa
- Senza ridenominazione il join (se naturale) di una tabella su se stessa produrrebbe la tabella stessa

- se prima del join si applica la ridenominazione, *si possono distinguere le colonne delle due tabelle ed effettuare interrogazioni particolari*

Esempio di self-join

- Data la relazione impiegati, trovare tutti gli impiegati che lavorano allo stesso progetto cui lavora Rossi

Impiegati

id	impiegato	progetto
1	Rossi	A
2	Neri	A
3	Neri	B
4	Bianchi	B

impiegatiXimpiegati

id	impiegato	progetto	id	impiegato	progetto
1	Rossi	A	1	Rossi	A
2	Neri	A	1	Rossi	A
3	Neri	B	1	Rossi	A
4	Bianchi	B	1	Rossi	A
1	Rossi	A	2	Neri	A
2	Neri	A	2	Neri	A
3	Neri	B	2	Neri	A
4	Bianchi	B	2	Neri	A
1	Rossi	A	3	Neri	B
2	Neri	A	3	Neri	B
3	Neri	B	3	Neri	B
4	Bianchi	B	3	Neri	B
1	Rossi	A	4	Bianchi	B
2	Neri	A	4	Bianchi	B
3	Neri	B	4	Bianchi	B
4	Bianchi	B	4	Bianchi	B

- se i progetti nelle due relazioni sono uguali, si ottengono gli impiegati che lavorano allo stesso progetto
  - Evidenziati solo quelli per rossi

Risultato del join

- S1=((impiegati) $><_{progetto=prog1}$ ρ(id1,imp1,prog1← id,impiegato,progetto)(impiegati))

Id	Impiegato	Progetto	id1	imp1	prog1
4	Bianchi	B	4	Bianchi	B
4	Bianchi	B	3	Neri	B
3	Neri	B	4	Bianchi	B
3	Neri	B	3	Neri	B
2	Neri	A	2	Neri	A
2	Neri	A	1	Rossi	A
1	Rossi	A	2	Neri	A
1	Rossi	A	1	Rossi	A

- se imp1=rossi, si ottengono gli impiegati che lavorano al progetto cui lavora rossi

$\sigma_{imp1=rossi}(S1)$

- selezione

$\boldsymbol{\Pi}_{impiegato}\left(\sigma_{imp1=rossi}(S1)\right)$

- proiezione per ottenere il risultato

$\boldsymbol{\Pi}_{impiegato}\left(\sigma_{impiegato<>rossi\ AND\ imp1=rossi}(S1)\right)$

- Per eliminare rossi dal risultato?

Un altro esempio di self-join

- Data la relazione impiegati, trovare tutti gli impiegati (il cognome) che guadagnano più di rossi

Impiegati

id	Cognome	Sipendio
5	rossi	100
6	verdi	200
7	blu	50

impiegatiXimpiegati

id	cognome	Stipendio	id1	Cogn1	Stip1
5	Rossi	100	5	rossi	100
6	verdi	200	5	rossi	100
7	blu	50	5	rossi	100
5	rossi	100	6	verdi	200
6	verdi	200	6	verdi	200
7	blu	50	6	verdi	200
5	rossi	100	7	blu	50
6	verdi	200	7	blu	50
7	blu	50	7	blu	50

- La seconda tabella è stata rinominata
- Le tuple con cogn1= rossi contengono contengono le informazioni sullo stipensio di Rossi e degli altri impiegati
- Di queste, quelle cercate sono con Stipendio>Stip1

Risultato del join

- S1=ρ(id1,cogn1,stip1← id,cognome,stipendio)(impiegati))
- S2= $\sigma_{cogn1=rossi}(S1)$
- S3=((impiegati) $><_{stipendio>Stip1}$ (S2)

id	cognome	Stipendio	id1	Cogn1	Stip1
5	Rossi	100	5	rossi	100
6	verdi	200	5	rossi	100
7	blu	50	5	rossi	100

- **Nota: non è un equi-join**
- risultato del prodotto cartesiano (il join è un operatore derivato)

id	cognome	Stipendio	id1	Cogn1	Stip1
5	Rossi	100	5	rossi	100
6	verdi	200	5	rossi	100
7	blu	50	5	rossi	100

- risultato dell'applicazione della condizione di join

$\boldsymbol{\Pi}_{cognome}(S3)$

- risultato finale

osservazione

- Anche la selezione iniziale può far parte del join (il risultato non cambia)

- S1=ρ(id1,cogn1,stip1← id,cognome,stipendio)(impiegati))
- S2=((impiegati) $><_{\sigma_{cogn1=rossi\ AND\ stipendio>Stip1}}$ (S1)
- $\boldsymbol{\Pi}_{cognome}(S2)$

id	cognome	Stipendio	id1	Cogn1	Stip1
5	Rossi	100	5	rossi	100
6	verdi	200	5	rossi	100
7	blu	50	5	rossi	100
5	rossi	100	6	verdi	200
6	verdi	200	6	verdi	200
7	blu	50	6	verdi	200
5	rossi	100	7	blu	50
6	verdi	200	7	blu	50
7	blu	50	7	blu	50

- risultato del prodotto cartesiano (il join è un operatore derivato)

id	cognome	Stipendio	id1	Cogn1	Stip1
5	Rossi	100	5	rossi	100
6	verdi	200	5	rossi	100
7	blu	50	5	rossi	100
5	rossi	100	6	verdi	200
6	verdi	200	6	verdi	200
7	blu	50	6	verdi	200
5	rossi	100	7	blu	50
6	verdi	200	7	blu	50
7	blu	50	7	blu	50

- risultato dell'applicazione della condizione di join

Un altro esempio

- Senza ridenominazione il join di una tabella su se stessa produrrebbe la tabella stessa
  - Se un natural join o se un inner join su tutti gli attributi

- se prima del join si applica la ridenominazione *si può ottenere una nuova tabella con proprietà particolari*
  - *Utilizzando condizioni di join particolari*

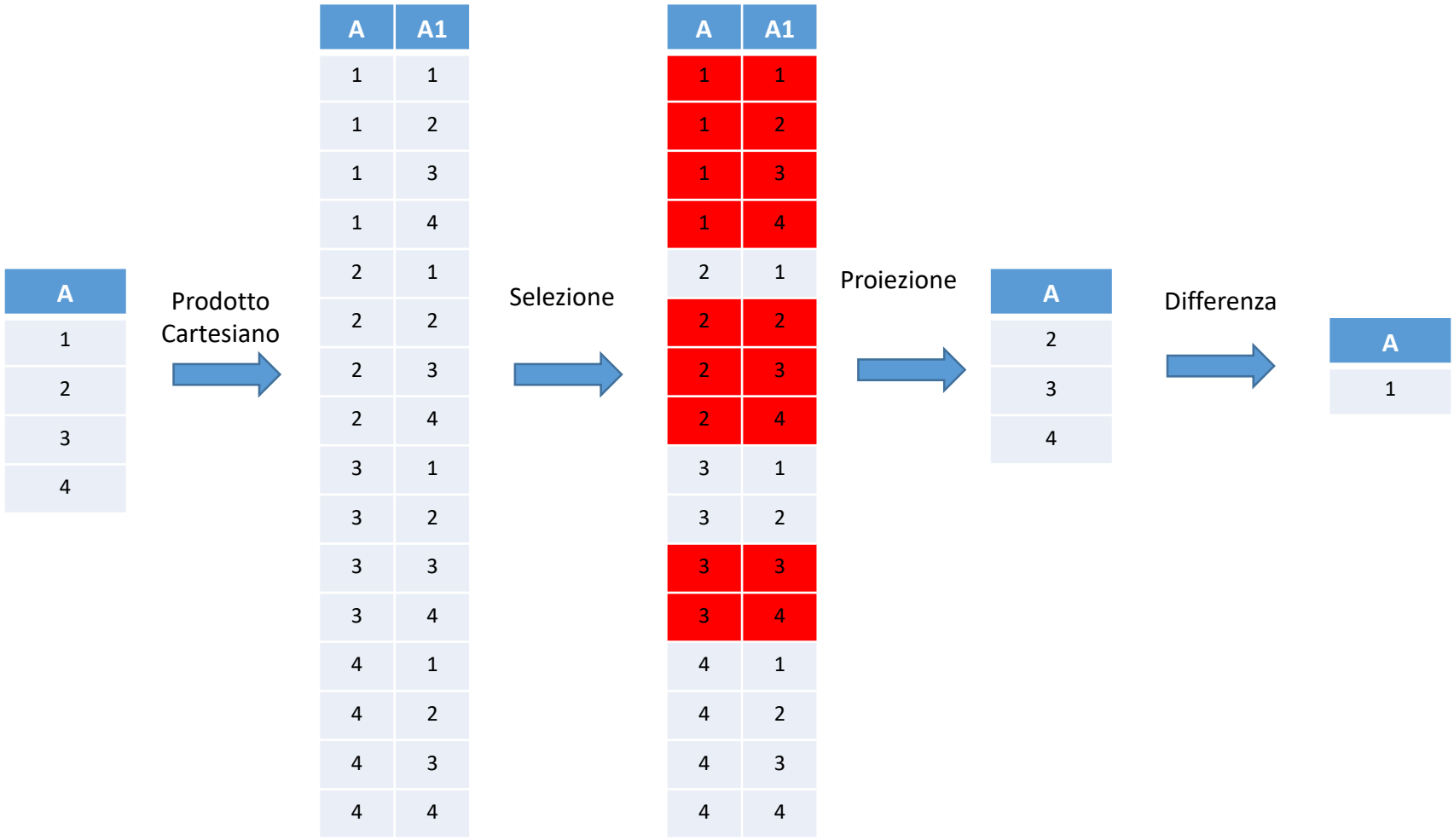
Ricerca del minimo/massimo

- Esempio dato uno schema R(A,B) trovare il minimo/massimo B in R
- Questo può essere fatto facendo un join di R su se stesa dopo aver rinominato tutti gli attributi nel seguente modo:

$$\pi_B (R) - \pi_B (R \bowtie_{B>B1} (\rho(A1,B1 \leftarrow A,B)(R)))$$

- Nella seconda parte vengono trovati tutti quei valori che non sono il minimo. Per far questo viene fatto un join tra la relazione R e se stessa, con però tutti gli attributi rinominati. La condizione di join dice che ogni attributo B deve essere maggiore degli stessi attributi rinominati. In questo modo vengono tenute tutte le tuple **tranne quella in cui l' attributo B assume il valore minore**. Quindi per il principio di complementarietà sottraendo dall' insieme iniziale, l' insieme delle tuple dove B non è il minimo, otteniamo proprio il valore minimo che cercavamo.

Ricerca Minimo - Esempio



Equivalenza di espressioni algebriche

- L’algebra relazionale permette di formulare espressioni fra loro equivalenti, cioè che producono lo stesso risultato
- L’equivalenza di espressioni dell’algebra risulta particolarmente importante in quanto può essere sfruttata dal DBMS in fase di esecuzione delle interrogazioni per ottimizzarne l’esecuzione
- A tal fine il DBMS solitamente utilizza delle trasformazioni di equivalenza cioè operazioni che sostituiscono un’espressione con un’altra equivalente

Trasformazioni equivalenti

- **Atomizzazione delle Selezioni**
  - Una congiunzione di selezioni può essere sostituita da una sequenza di selezioni atomiche:
  - $\sigma_{F1 \text{ AND } F2}(E) \equiv \sigma_{F2}(\sigma_{F1}(E))$
  - E è un espressione qualsiasi
- **Idempotenza delle Proiezioni**
  - Una proiezione può essere trasformata in una sequenza di proiezioni che eliminano i vari attributi in varie fasi
  - $\pi_A(E) \equiv \pi_A(\pi_{A,B}(E))$
  - E espressione definita su un insieme di attributi che contiene A e B

Trasformazioni equivalenti

- **Anticipazione della Selezione Rispetto al Join**
  - $\sigma_F (E1 \bowtie E2) \equiv E1 \bowtie \sigma_F (E2)$
  - Se la condizione F coinvolge solo attributi della sottoespressione E2.
- **Anticipazione della Proiezione Rispetto al Join**
  - $\pi_{Y2}(E1 \bowtie E2) \equiv E1 \bowtie \pi_{Y2}(E2)$
  - Se Y2 sono attributi di E2 e I suoi attributi sono coinvolti nel join
- **Trasformazioni basate sulla corrispondenza tra operatori insiemistici e selezioni:**
  - $\sigma_{F1 \text{ OR } F2}(R) \equiv \sigma_{F1}(R) \cup \sigma_{F2}(R)$
    $\sigma_{F1 \text{ AND } F2}(R) \equiv \sigma_{F1}(R) \cap \sigma_{F2}(R)$