

Algebra Relazionale

c.vallati@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

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**, 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 di input
 - generare delle tabelle 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)
 - **proiezione** (notazione Π)
 - **ridenominazione** (notazione ρ)
- Operatori di arità 2, derivati dagli operatori insiemistici tradizionali:
 - **unione** (notazione \cup)
 - **differenza insiemistica** (notazione $-$)
 - **intersezione** (notazione \cap)
 - **prodotto cartesiano** (notazione \times)
- Operazioni derivate dalle precedenti:
 - **Join** (notazione \bowtie)
- Gli operatori si possono comporre
- Gli operandi sono o (nomi di) relazioni del BD o espressioni

Selezione

Notazione (σ)

- 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
- lo schema della relazione risultato è lo stesso di quella di origine
- il predicato è costituito dal nome di un attributo, da un operatore, da un valore costante
- più predicati possono essere uniti con or (\vee) o and (\wedge)
- operatori sono
 - =, >, <, >=, <=, # (diverso)

σ Predicato (relazione)

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

Proiezione

Notazione (Π)

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

Π Lista Attributi (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:

- matricola e cognome
- cognome e filiale

Proiezione – Esempio

Esempio operazione di Proiezione:

- matricola e cognome di tutti gli impiegati

Π matricola, cognome (Impiegati)

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

- cognome e filiale di tutti gli impiegati

Π cognome, filiale (Impiegati)

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

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 se voglio sapere matricola e cognome degli impiegati che guadagnano più di 62, sottopongo la query

Π matricola, cognome (σ Stipendio>62 (Impiegati))

Ordine delle operazioni

Esempio: selezione e proiezione

Π matricola, cognome (σ stipendio>62 (Impiegati))

Otengo il risultato:

Matricola	Cognome
5998	Neri
5698	Bianchi

e se scambiassi l'ordine di selezione e proiezione?

σ stipendio>62 (Π matricola, cognome (Impiegati))

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)

- A volte in preparazione all'esecuzione ad un'applicazione o in seguito ad una sua esecuzione si ha bisogno di rinominare gli attributi di una relazione
- A tal fine un'operatore unario specifico è stato definito, 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

ρ NomeNuovo \leftarrow Nome(relazione)

Ridenominazione – Esempio

Persone

CF	Cognome	Nome
BNCGRG78F21A	Bianchi	Mario
RSSGRG78F23X	Rossi	Paolo

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

CodiceFiscale	Cognome	Nome
BNCGRG78F21A	Bianchi	Mario
RSSGRG78F23X	Rossi	Paolo

Unione, Intersezione e Differenza

- Le Relazioni sono insiemi, quindi possiamo applicare gli operatori tra insiemi
- Tuttavia, vogliamo che il risultato siano relazioni (cioè insiemi omogenei di ennuple)
 - 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, in B o in entrambe
 - 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

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

Π NomeCliente(σ NomeFiliale=Roma(Depositi))

U

Π NomeCliente(σ NomeFiliale=Roma(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 ($\sigma_{\text{Filiale}=\text{Roma}}$ (Depositi))

Cliente
Bianchi
Neri

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

Cliente
Rossi

Π Cliente ($\sigma_{\text{Filiale}=\text{Roma}}$ (Depositi)) \cup Π Cliente ($\sigma_{\text{Filiale}=\text{Roma}}$ (Prestiti))

Cliente
Rossi
Bianchi
Neri

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

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

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

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

Cliente
Bianchi
Neri

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

Cliente
Rossi
Neri

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

Cliente
Bianchi

Intersezione

- L' intersezione fra due tabelle $A \cap B$ è una tabella che contiene le tuple che sono presenti in A e in B
- **Esempio**: trovare i clienti che hanno sia un conto che un prestito

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

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

Cliente
Bianchi
Neri

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

Cliente
Neri

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

Cliente
Neri

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)

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

Banchieri(NomeCliente, NomeBanchiere)

NomeCliente	NomeBanchiere
Rossi	Cuccia
Bianchi	Visco

R1 X R2

Clienti x Banchieri

Clienti x Banchieri(C.NomeCliente, Via, Città, B.NomeCliente, NomeBanchiere)

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

NOTA: l'attributo NomeCliente 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!

Join

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

Join Naturale

- **Notazione** ▷ ◁
- Il Join naturale è un operatore binario (la sua definizione può essere facilmente estesa a più tabelle) che correla dati in relazioni diverse sulla base dei valori uguali in attributi con lo stesso nome
- La tabella 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 anche tramite un prodotto cartesiano e una selezione sugli 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 ▷◁ 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

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

Join Incompleto – Esempio

Join naturale incompleto:

R1

Impiegato	Reparto
Rossi	Vendite
Neri	Produzione
Bianchi	Produzione

R2

Capo	Reparto
Bruni	Acquisti
Mori	Produzione

R1 \bowtie 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 \bowtie LEFT R2

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

R1 \bowtie RIGHT R2

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

R1 \bowtie FULL 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 \bowtie 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

- 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 il **theta-join** che permette l'esecuzione del prodotto cartesiano e della selezione sul risultato con un'unica operazione: $R1 \bowtie_{OP} R2 = \sigma_{OP}(R1 \bowtie R2)$
- 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

Impiegati ▷◁ Progetto=Codice Progetti

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

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 di una tabella su se stessa produrrebbe la tabella stessa, se prima del join si applica la ridenominazione *si può ottenere una nuova tabella con proprietà particolari*

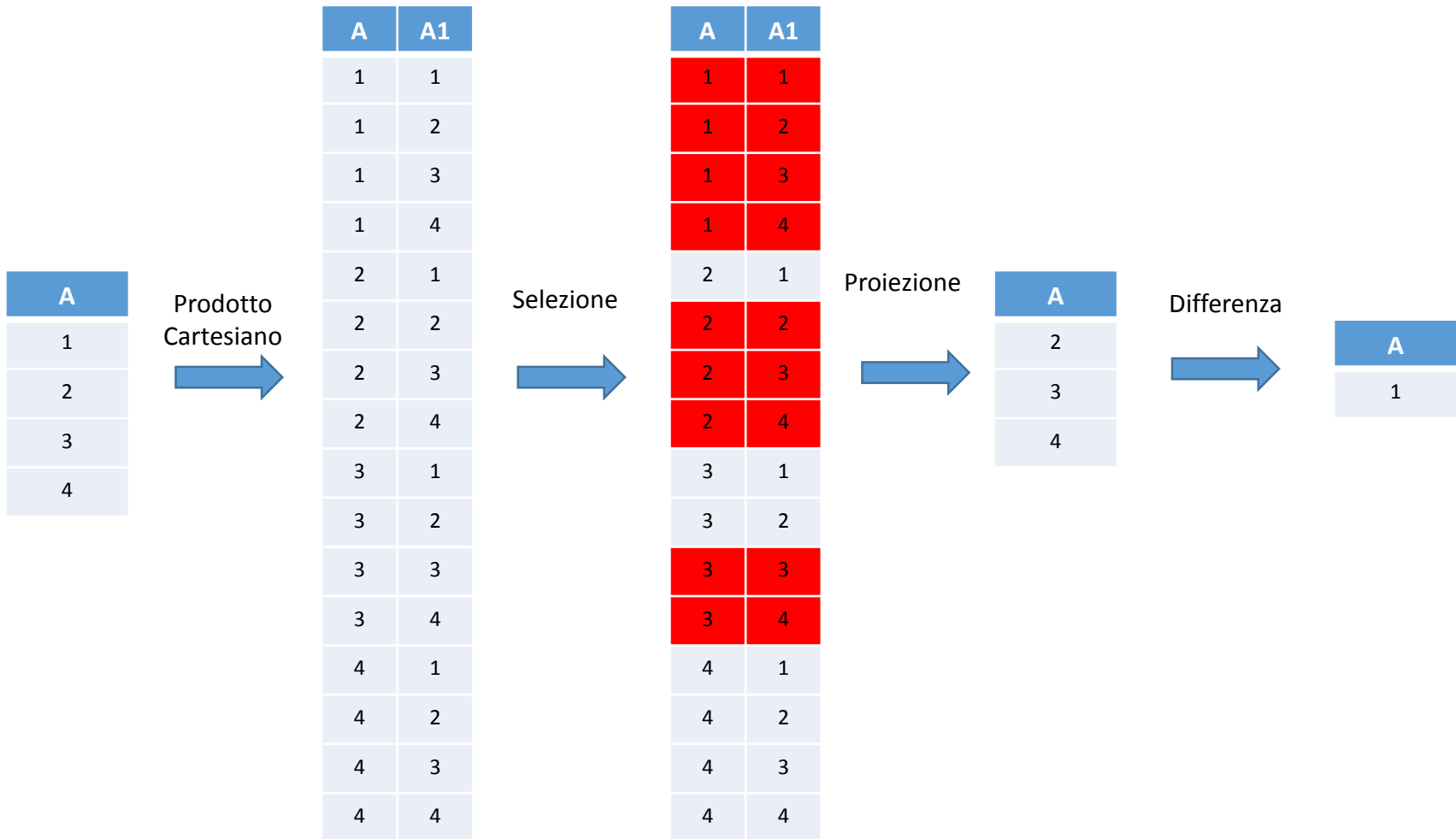
Ricerca del minimo/massimo

- 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 stessa 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 complementarità 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:
- $\sigma_{F_1 \text{ AND } F_2}(E) \equiv \sigma_{F_2}(\sigma_{F_1}(E))$
- 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 non sono coinvolti nel join

- **Trasformazioni basate sulla corrispondenza tra operatori insiemistici e selezioni:**

- $\sigma_{F1 \text{ OR } F2}(R) \equiv \sigma_{F1}(R) \cup \sigma_{F2}(R)$
 $\sigma_{F1 \text{ AND } F2}(R) \equiv \sigma_{F1}(R) \cap \sigma_{F2}(R)$