ALGEBRA RELAZIONALE

Linguaggi per basi di dati

- operazioni sullo schema
  - DDL
- operazioni sui dati
  - DML
    - interrogazione ("query")
    - aggiornamento

Linguaggi di interrogazione per basi di dati relazionali

- Dichiarativi ("che cosa")
  - specificano le proprietà del risultato
- Procedurali ("come")
  - costruiscono il risultato

Linguaggi di interrogazione

- Algebra relazionale: procedurale
- Calcolo relazionale:
  dichiarativo (teorico)
- SQL (Structured Query Language):
  parzialmente dichiarativo (reale)
- QBE (Query by Example):
  dichiarativo (reale)

Algebra relazionale

- Insieme di operatori
  - su relazioni, producono relazioni
  - quindi possono essere composti

Un servizio online per esercitazioni in algebra relazionale

- RelaX
  - http://dbis-uibk.github.io/relax/calc
- Verrà proposto un “homework” il cui svolgimento sarà necessario per partecipare alla prova parziale

- Oggi lo usiamo per vedere i risultati, chi ha il calcolatore può provare ad utilizzarlo subito

Operatori dell'algebra relazionale

- unione, intersezione, differenza
- ridenominazione
- selezione
- proiezione
- aggregazione
- join (join naturale, prodotto cartesiano, theta-join)

Operatori insiemistici

- le relazioni sono insiemi
- i risultati debbono essere relazioni
- quindi:
  - è possibile applicare unione, intersezione, differenza solo a relazioni definite sugli stessi attributi

- esempi alla lavagna e su Relax

Base di dati per l'esempio

Laureati

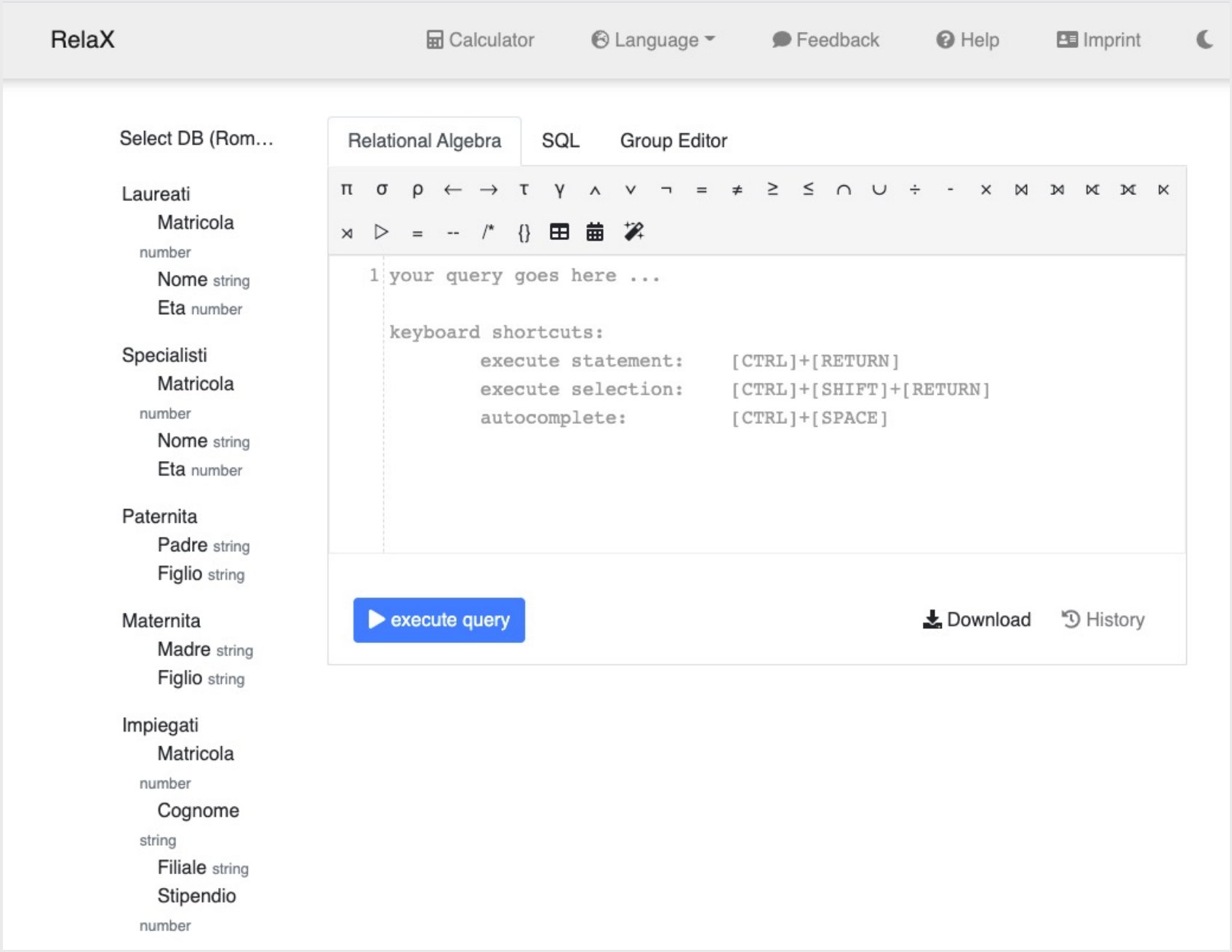
Matricola	Nome	Età
7274	Rossi	42
7432	Neri	54
9824	Verdi	45

Specialisti

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

- La carichiamo in Relax

http://dbis-uibk.github.io/relax/calc/gist/1a9dc6cd0f3478388fc177dfc9b5a314


RelaX
Calculator
Language
Feedback
Help
Imprint
Select DB (Rom…
Laureati
Matricola number
Nome string
Eta number
Specialisti
Matricola number
Nome string
Eta number
Paternita
Padre string
Figlio string
Maternita
Madre string
Figlio string
Impiegati
Matricola number
Cognome string
Filiale string
Stipendio number
Relational Algebra
SQL
Group Editor
π σ ρ ← → τ γ ∧ ∨ ¬ = ≠ ≥ ≤ ∩ ∪ ÷ - × ⋈ ⋈ ⋈ ⋈ ⋉
⋊ ▷ = -- /* {}
1 your query goes here ...
keyboard shortcuts:
execute statement: [CTRL]+[RETURN]
execute selection: [CTRL]+[SHIFT]+[RETURN]
autocomplete: [CTRL]+[SPACE]
execute query
Download
History

Un'unione sensata ma impossibile

Paternità

Padre	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

Maternità

Madre	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

Paternità $\cup$ Maternità

??

- Proviamo con RelaX

Ridenominazione

- operatore su una relazione
- "modifica lo schema" lasciando inalterata l'istanza dell'operando

- simbolo REN oppure RHO oppure ρ

- sintassi ed esempio alla lavagna e su RelaX

Paternità

Padre	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

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

Genitore	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

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

- "seleziona" da una relazione le ennuple 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

- Alla lavagna e su RelaX

- Simbolo
  - SEL
  - σ

- impiegati che guadagnano più di 50

$SEL_{Stipendio > 50}$ (Impiegati)

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

$SEL_{Stipendio > 50}$ (Impiegati)

- impiegati che guadagnano più di 50 e lavorano a Milano

$$SEL_{Stipendio > 50 \text{ AND } Filiale = 'Milano'}(Impiegati)$$

Matricola	Cognome	Filiale	Stipendio
5998	Neri	Milano	64

$$SEL_{Stipendio > 50 \text{ AND } Filiale = 'Milano'}(Impiegati)$$

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

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

Matricola	Cognome	Filiale	Stipendio
9553	Milano	Milano	44

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

selezione

Proiezione

- decompone "verticalmente":
  - "tutte" le ennuple, alcuni attributi

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

- alla lavagna

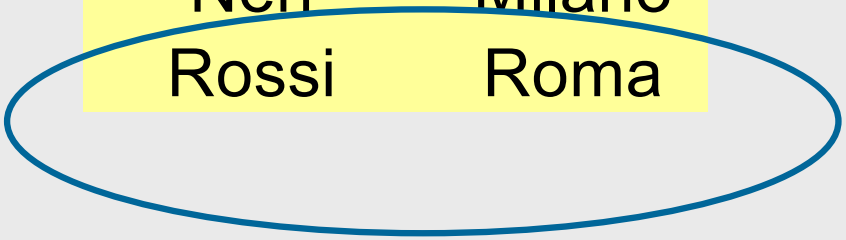
- matricola e cognome di tutti gli impiegati

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



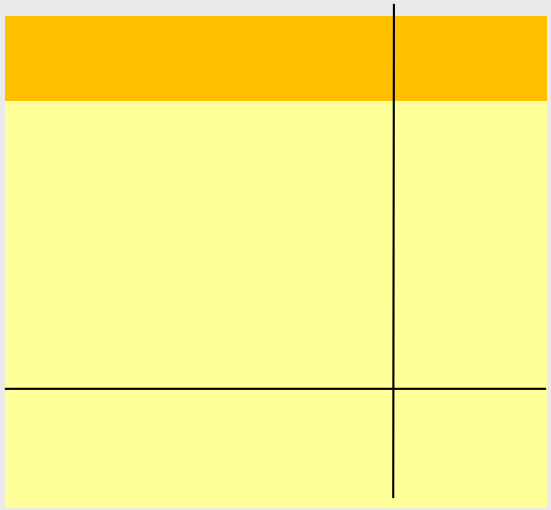
PROJ $_{\text{Cognome, Filiale}}$ (Impiegati)

Cardinalità delle proiezioni

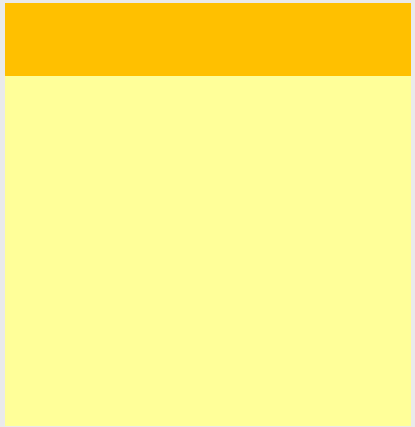
- Una proiezione
  - contiene al più tante ennuple quante l'operando
  - può contenerne di meno
- Quando possiamo essere sicuri che non siano di meno?

Cardinalità delle proiezioni

- Proprietà interessante:
  - se X è una superchiave di R, allora $PROJ_X(R)$ contiene esattamente tante ennuple quante R



proiezione

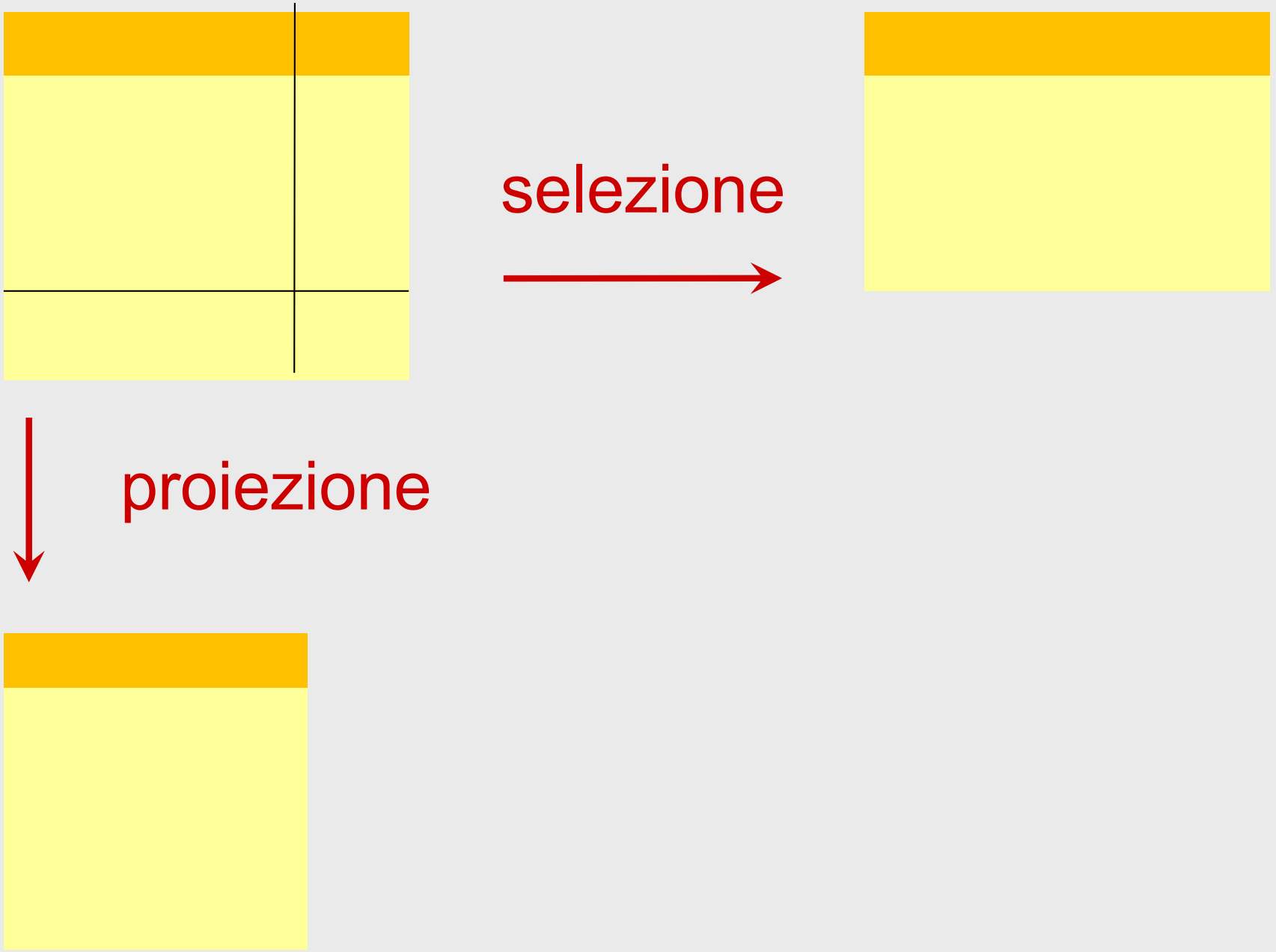


Selezione e proiezione

- operatori "ortogonali"
  - alla lavagna

Selezione e proiezione

- operatori "ortogonali"
- selezione:
  - decomposizione orizzontale
- proiezione:
  - decomposizione verticale


selezione
proiezione

Selezione con valori nulli

Impiegati

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

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

- la condizione atomica è vera solo per valori non nulli
- vediamo su Relax (relazione ImpiegatiTer)

http://dbis-uibk.github.io/relax/calc/gist/1a9dc6cd0f3478388fc177dfc9b5a314

Un risultato non desiderabile

$$\text{SEL}_{\text{Stipendio}>40}(\text{Persone}) \cup \text{SEL}_{\text{Stipendio}\leq 40}(\text{Persone}) \neq \text{Persone}$$

- Vediamo su Relax.
- Perché?
  - Perché le selezioni vengono valutate separatamente!
- Ma anche

  $$\text{SEL}_{\text{Stipendio}>40 \text{ OR } \text{Stipendio}\leq 40}(\text{Persone}) \neq \text{Persone}$$
- Perché?
  - Perché anche le condizioni atomiche vengono valutate separatamente!

Selezione con valori nulli: soluzione

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

- la condizione atomica è vera solo per valori non nulli
- per riferirsi ai valori nulli esistono forme apposite di condizioni:

IS NULL

IS NOT NULL

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

Nota bene

- RelaX usa

… = null

- Quindi:

$$\text{SEL}_{\text{Stipendio}>40}(\text{Persone}) \cup \text{SEL}_{\text{Stipendio}\leq 40}(\text{Persone}) \cup \text{SEL}_{\text{Stipendio IS NULL}}(\text{Persone})$$

$$=$$

$$\text{SEL}_{\text{Stipendio}>40 \text{ OR Stipendio}\leq 40 \text{ OR Stipendio IS NULL}}(\text{Persone})$$

$$=$$

$$\text{Persone}$$

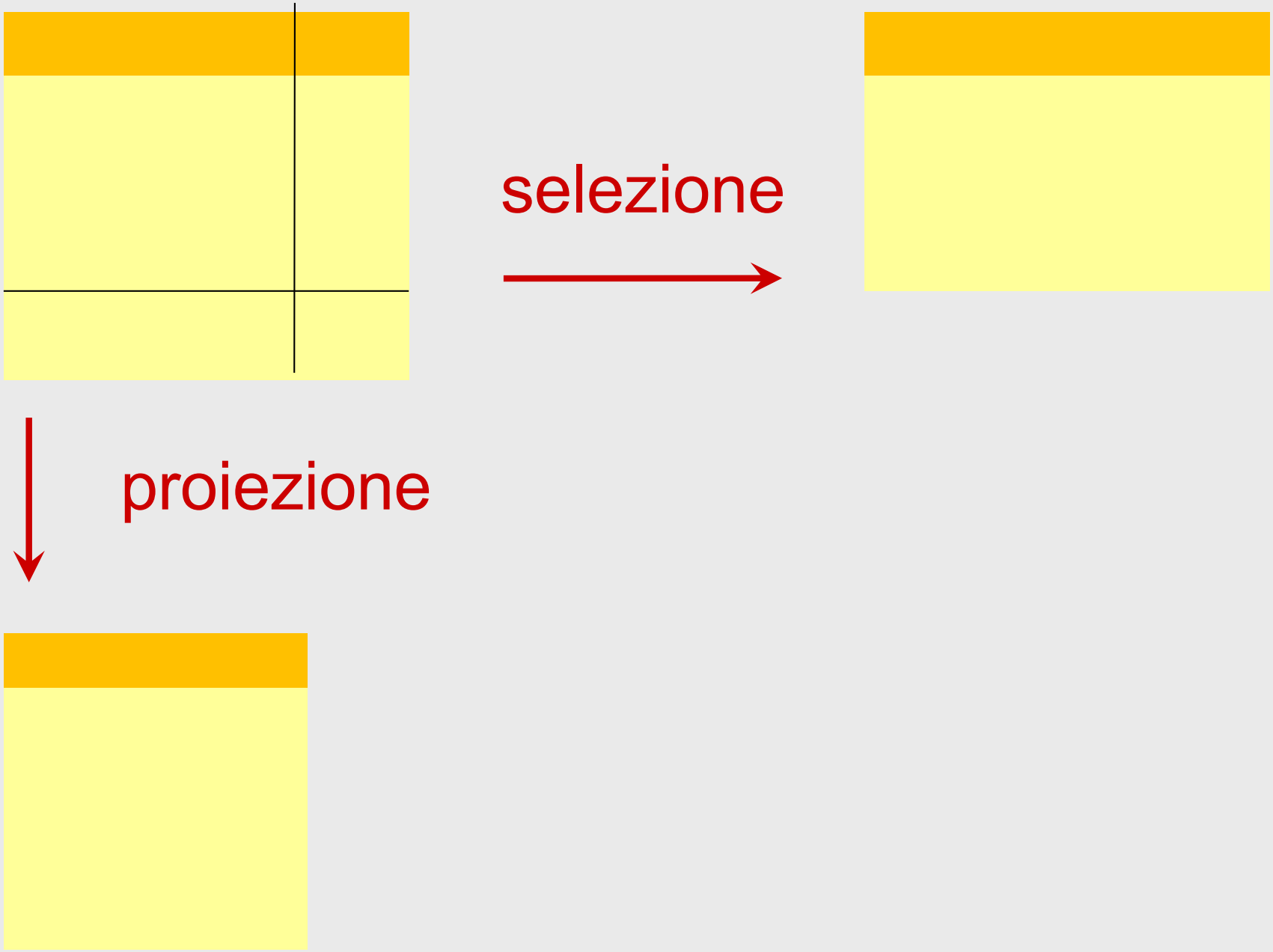
Impiegati

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

SEL (Stipendio > 40) OR (Stipendio IS NULL) (Impiegati)

Selezione e proiezione

- operatori "ortogonali"
- selezione:
  - decomposizione orizzontale
- proiezione:
  - decomposizione verticale


selezione
proiezione

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

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

- Combinando selezione e proiezione, possiamo estrarre informazioni da una relazione
- Invece non possiamo
  - calcolare informazioni derivate
  - correlare informazioni presenti in relazioni diverse, né informazioni in ennuple diverse di una stessa relazione
- Vediamo altri operatori che permettono queste attività

Join

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

Il solito esempio

(con nomi di attributi modificati in "esami")

studenti

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

esami

Matricola	Voto	Codice
3456	30	04
3456	24	02
9283	28	01
6554	26	01

Che voti ha preso **Maria Rossi** negli esami che ha superato?

corsi

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

studenti

Matricola	Cognome	Nome	Data di nascita
		…	
3456	Rossi	Maria	01/02/1978

esami

Matricola	Voto	Codice
3456	30	04
3456	24	02

corsi

Codice	Titolo	Docente
02	Chimica I	Bruni
04	Chimica II	Verdi

(studenti JOIN esami) JOIN corsi

Matricola	Cognome	Nome	Data di nascita	Voto	Codice	Titolo	Docente
3456	Rossi	Maria	01/02/1978	30	04	Chimica II	Bruni
3456	Rossi	Maria	01/02/1978	24	02	Chimica I	Verdi

Join naturale

- operatore binario (generalizzabile)
- produce un risultato
  - sull'**unione** degli attributi degli operandi
  - con ennuple costruite ciascuna a partire da una ennupla di ognuno degli operandi

Join, sintassi e semantica

- alla lavagna

JOIN $\bowtie$

JOIN NATURALE

$R_1(X_1)$ $R_2(X_2)$

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

Esempi

- alla lavagna
  - join completo, non completo, vuoto, mxn

R_1

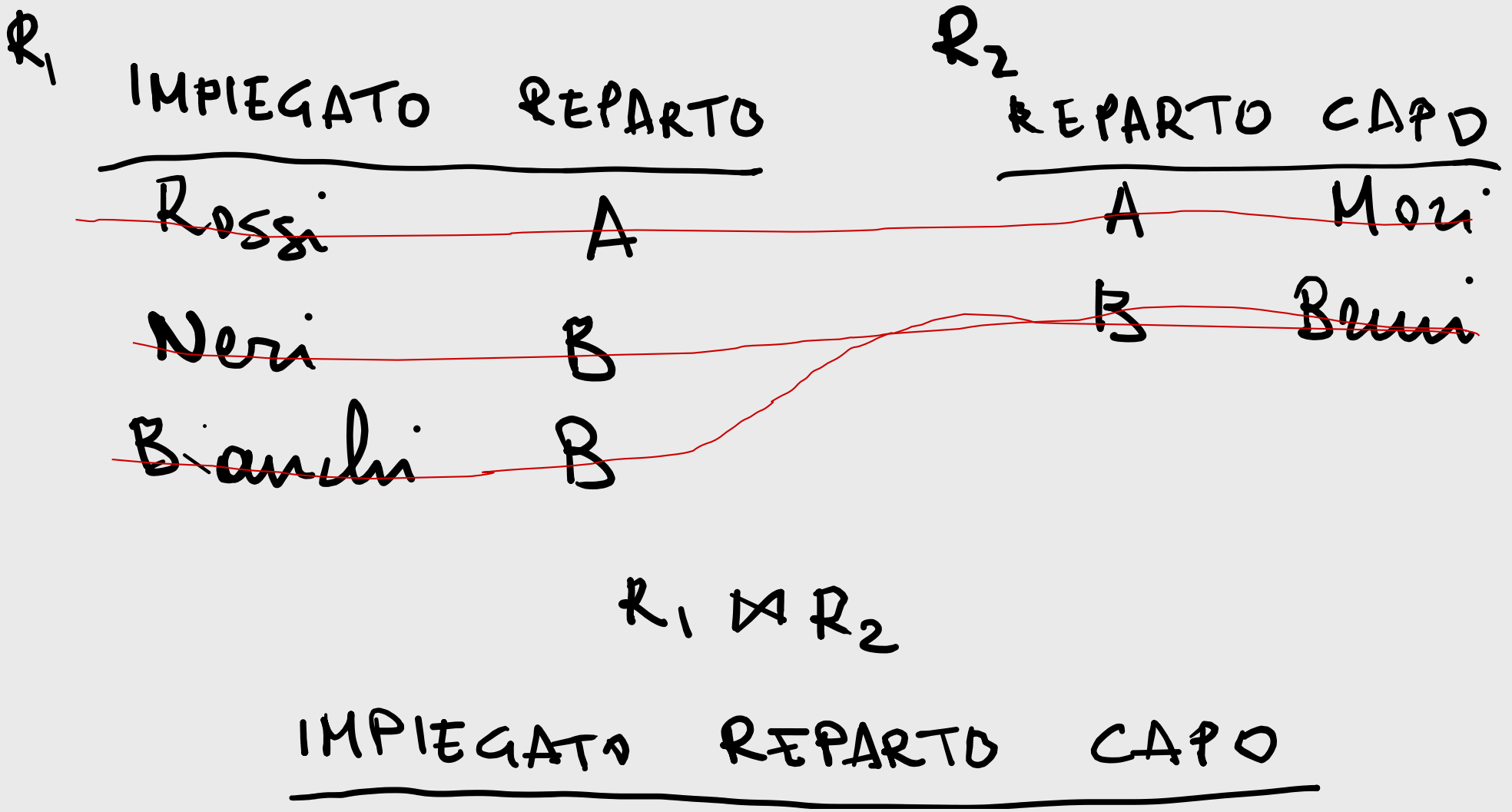
IMPIEGATO	REPARTO
Rossi	A
Neri	B
Bianchi	B

R_2

REPARTO	CAPO
A	Mori
B	Bruni

$R_1 \bowtie R_2$

IMPIEGATO	REPARTO	CAPO
Rossi	A	Mori
Neri	B	Bruni
Bianchi	B	Bruni


R1
IMPIEGATO REPARTO
Rossi A
Neri B
Bianchi B
R2
REPARTO CAPO
A Mori
B Bruni
R1 ⋈ R2
IMPIEGATO REPARTO CAPO

R_1

IMPIEGATO	REPARTO
Rossi	A
Neri	B
Bianchi	B

R_2

REPARTO	CAPO
B	Mori
C	Bruni

$R_1 \bowtie R_2$

IMPIEGATO	REPARTO	CAPO

R_1

IMPIEGATO	REPARTO
Rossi	A
Neri	B
Bianchi	B

R_2

REPARTO	CAPO
C	Mori
D	Bruni

$R_1 \bowtie R_2$

IMPIEGATO	REPARTO	CAPO

R_1

IMPIEGATO	REPARTO
Rossi	A
Neri	A
Bianchi	A

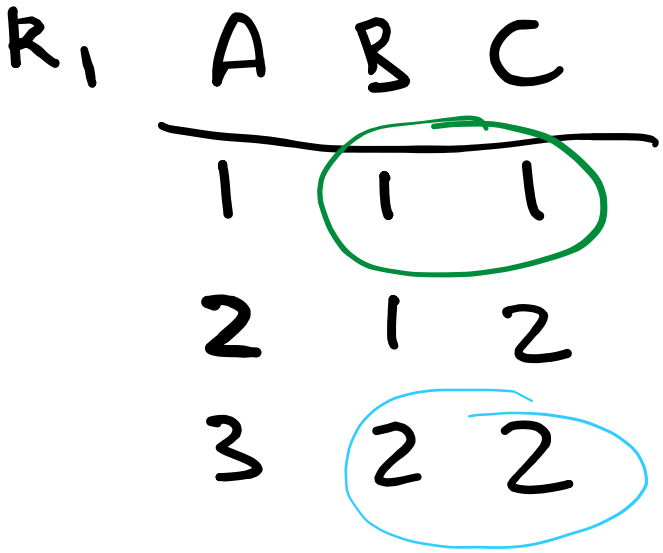
R_2

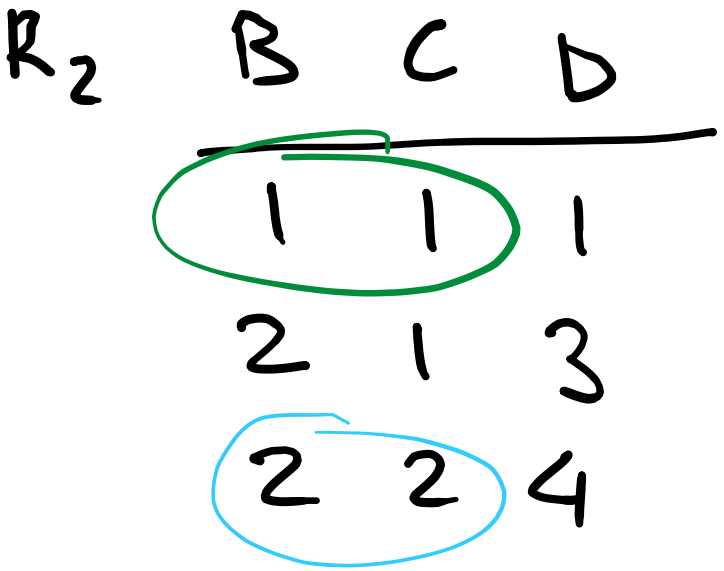
REPARTO	CAPO
A	Mori
A	Bruni

$R_1 \bowtie R_2$

IMPIEGATO	REPARTO	CAPO
⋮		

Esempio!





$R_1 \bowtie R_2$

A	B	C	D

Cardinalità del join

- Il join di R_1 e R_2 contiene un numero di ennuple …
  - alla lavagna (chiavi, vincoli integrità referenziale)

$R_1 \quad |R_1|$

$R_2 \quad |R_2|$

$$0 \leq |R_1 \bowtie R_2| \leq |R_1| \cdot |R_2|$$

R_1

A	B
1	0
2	4
3	4

R_2

B	C
1	x
2	y
3	2
4	w
...	

$$0 \leq |R_1 \bowtie R_2| \leq |R_1|$$

R_1

A	B
1	1
2	4
3	4

R_2

B	C
1	x
2	y
3	2
4	w
...	

VINCOLO DI INTEGRITÀ RIF.

FRA $R_1(B)$ E LA CHIAVE B DI R_2

$$|R_1 \not\ltimes R_2| = |R_1|$$

Cardinalità del join

- Il join di R_1 e R_2 contiene un numero di ennuple
  - compreso fra zero e il prodotto di $|R_1|$ e $|R_2|$
- se il join coinvolge una chiave di R_2, allora il numero di ennuple è
  - compreso fra zero e $|R_1|$
- se il join coinvolge una chiave di R_2 e un vincolo di integrità referenziale verso di essa, allora il numero di ennuple è
  - pari a $|R_1|$

Cardinalità del join, 2

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

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

- se B è chiave in R_2

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

- se B è chiave in R_2 ed esiste vincolo di integrità referenziale fra B (in R_1) e R_2:

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

Join, una difficoltà

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
B	Mori
C	Bruni

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori

- alcune ennuple non contribuiscono al risultato: 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, destro, completo

Join, esterno

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
B	Mori
C	Bruni

- Vedere anche su Relax (nella base di dati mostrata per gli esempi, usare R1 e R2)

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*

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

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Reparto	Capo
B	Mori
C	Bruni

Impiegati $JOIN_{FULL}$ Reparti

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

- Su Relax il risultato ha una forma leggermente diversa, ma il concetto è lo stesso

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

Join e proiezioni

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

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

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
Neri	B	Bruni
Bianchi	B	Mori
Bianchi	B	Bruni
Verdi	A	Bini

Proiezioni e join

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

Impiegato	Reparto
Neri	B
Bianchi	B
Verdi	A

Reparto	Capo
B	Mori
A	Bini

Join e proiezioni

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

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

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

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

Join e proiezioni

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

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

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

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

- Quando c'è uguaglianza?

Proiezioni e join

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

Impiegato	Reparto
Neri	B
Bianchi	B
Verdi	A

Reparto	Capo
B	Mori
A	Bini

Join e proiezioni

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

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

- Quando c'è uguaglianza?
  - quando gli attributi comuni alle due proiezioni $X_1 \cap X_2$ sono chiave (in effetti, basta superchiave) per una delle due proiezioni)

Prodotto cartesiano

- un join naturale su relazioni senza attributi in comune
- 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

- Il prodotto cartesiano, in pratica, ha senso (quasi) solo se seguito da selezione:

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

- L'operazione viene chiamata theta-join e indicata con

$$R_1\ JOIN_{Condizione}\ R_2$$

Equi-join

- Se l'operatore di confronto nel theta-join è sempre l'uguaglianza (=) allora si parla di equi-join

Nota: ci interessa davvero l'equi-join, non il theta-join più generale

- Equi-join: prodotto cartesiano seguito da selezione di uguaglianza

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

Join naturale ed equi-join

- In pratica, ciò che ci interessa è l'equi-join
- Il join naturale lo abbiamo usato solo a fini didattici, perché i concetti sono più semplici
- Nelle interrogazioni "pratiche" useremo l'equi-join

Equivalenza di espressioni

- Due espressioni sono equivalenti se producono risultati uguali fra loro su ogni istanza della base di dati
- L'equivalenza è importante in pratica perché i DBMS cercano di eseguire espressioni equivalenti a quelle date, ma meno "costose"

Un'equivalenza importante

- Push selections (se A è attributo di R_1)

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

Nota

- In questo corso, ci preoccupiamo poco dell'efficienza:
  - L'obiettivo è di scrivere interrogazioni corrette e leggibili
- Motivazione:
  - I DBMS si preoccupano di scegliere le strategie realizzative efficienti

Aggregazione

- Il contenuto delle basi di dati viene spesso aggregato:
  - Il voto massimo nell'esame di basi di dati
  - Il voto medio per ciascun esame
  - Il numero di CFU conseguiti da ciascuno studente

Aggregazione, sintassi e semantica

- Sintassi

  AGG *AttributiRaggruppamento; Funzione (Attributo), …* (*Operando*)

  - *Funzione*: count, sum, avg, max, min

- Semantica
  - Il risultato contiene la proiezione sugli attributi di raggruppamento e il valore delle funzioni in corrispondenza al sottoinsieme raggruppato
- simbolo più compatto … γ (da GROUP BY)

Versione semplice

- Senza attributo di raggruppamento
- Calcolo sull'intera relazione

  $AGG_{Funzione\ (Attributo)}\ (Operando)$

  o meglio

  $AGG_{Funzione\ (Attributo) \rightarrow Nome}\ (Operando)$

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Il numero totale di esami

AGG $_{\text{count(*)} \rightarrow \text{NumeroEsami}}$ (esami)

NumeroEsami
5

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Il numero totale di esami

AGG $_{\text{count(*)} \rightarrow \text{NumeroEsami}}$ (esami)

NumeroEsami
5

- In RelaX (con o senza parentesi)

$$\gamma\ ;\ \text{COUNT(*)} \rightarrow \text{NumeroEsami}\ \ \text{Esami}$$

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Il voto medio complessivo

$$AGG_{avg(Voto) \rightarrow MediaGenerale}(esami)$$

MediaGenerale
27

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Il voto medio complessivo

$$AGG_{avg(Voto) \rightarrow MediaGenerale}(esami)$$

MediaGenerale
27

$$\gamma_{; AVG(Voto) \rightarrow MediaGenerale}(Esami)$$

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Il numero di esami e il voto medio complessivo

NumeroEsami	MediaGenerale
5	27

$$AGG_{count(*) \rightarrow NumeroEsami,\ avg(Voto) \rightarrow MediaGenerale}(esami)$$

$$\gamma_{count(*) \rightarrow NumeroEsami,\ avg(Voto) \rightarrow Media}(Esami)$$

Versione completa

- Con attributi di raggruppamento
- Calcolo sulle partizioni

  AGG *AttributiRaggruppamento; Funzione (Attributo), …* *(Operando)*

  o meglio

  AGG *AttributiRaggruppamento; Funzione (Attributo) → Nome, …* *(Operando)*

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Il voto medio per ciascuno studente

AGG Matricola; avg(Voto) → Media (esami)

Matricola	Media
3456	28
9283	27
6554	26

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Il voto medio per ciascuno studente

$$\mathcal{AGG}_{\text{Matricola; avg(Voto)} \rightarrow \text{Media}}(\text{esami})$$

Matricola	Media
3456	28
9283	27
6554	26

$$\gamma_{\text{Matricola; AVG(Voto)} \rightarrow \text{MediaGenerale}}(\text{Esami})$$

esami

Matricola	Voto	Codice
3456	30	04
3456	26	02
9283	27	01
6554	26	01
6554	26	05

- Numero esami e voto medio per ciascuno studente

AGG Matricola; avg(Voto) → Media, count(*) → NumEsami (esami)

Matricola	Media	NumEsami
3456	28	2
9283	27	1
6554	26	2

Aggregazioni e valori nulli

- vengono trattati correttamente (cioè ignorati nelle medie, nelle somme e nei conteggi)

count

- count(*) conta le ennuple
- count(A) conta le ennuple che hanno il valore di A
  - non i valori distinti di A
  - quindi in assenza di valori il risultato è lo stesso, in presenza di valori nulli per A
    - count(A) è il numero di ennuple che non hanno il valore nullo

esami

Matricola	Voto	Codice
3456	30	04
3456	NULL	02
9283	27	01
6554	26	01
6554	26	05

- Numero esami e voto medio per ciascuno studente

AGG Matricola; avg(Voto) → Media, count(*) → NumEsami (esami)

Matricola	Media	NumEsami
3456	30	2
9283	27	1
6554	26	2

- Numero voti e voto medio per ciascuno studente

AGG Matricola; avg(Voto) → Media, count(Voto) → NumVoti (esami)

Matricola	Media	NumVoti
3456	30	1
9283	27	1
6554	26	2

Viste (relazioni derivate)

- Relazioni di base: contenuto autonomo, le relazioni nella base di dati
- Relazioni derivate:
  - relazioni il cui contenuto è funzione del contenuto di altre relazioni (definito per mezzo di interrogazioni)
- Le relazioni derivate possono essere definite su altre derivate, ma …

Viste, esempio

Afferenza

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Direzione

Reparto	Capo
A	Mori
B	Bruni

- una vista:

Supervisione =

PROJ $_{\text{Impiegato, Capo}}$ (Afferenza JOIN Direzione)

Interrogazioni sulle viste

- Sono eseguite sostituendo alla vista la sua definizione:

$$SEL_{Capo='Leoni'}\ (Supervisione)$$

viene eseguita come

$$SEL_{Capo='Leoni'}(\ PROJ_{Impiegato,\ Capo}\ (Afferenza\ JOIN\ Direzione))$$

Viste, motivazioni

Nota bene:

- L'utilizzo di viste non influisce sull'efficienza delle interrogazioni

Vantaggi:

- Soprattutto:
  - **Strumento di programmazione**:
    - si può semplificare la scrittura di interrogazioni: espressioni complesse e sottoespressioni ripetute
- Ogni utente vede solo
  - ciò che gli interessa e nel modo in cui gli interessa, senza essere distratto dal resto
  - ciò che autorizzato a vedere (autorizzazioni)
- Utilizzo di programmi esistenti su schemi ristrutturati

Viste come strumento di programmazione

- Trovare gli impiegati che hanno lo stesso capo di Rossi (vedremo meglio il concetto più avanti)
- Senza vista:

$$\text{PROJ}_{\text{Impiegato}}((\text{Afferenza JOIN Direzione}) \text{ JOIN } \text{REN}_{\text{ImpR,RepR} \leftarrow \text{Imp,Reparto}}(\text{SEL}_{\text{Impiegato='Rossi'}}(\text{Afferenza JOIN Direzione})))$$

- Con la vista:

$$\text{PROJ}_{\text{Impiegato}}(\text{Supervisione JOIN } \text{REN}_{\text{ImpR} \leftarrow \text{Imp}}(\text{SEL}_{\text{Impiegato='Rossi'}}(\text{Supervisione})))$$

Un servizio online per esercitazioni in algebra relazionale

(lo abbiamo già visto ma ripetiamo i dettagli per comodità – ora è necessario usare lo strumento)

- RelaX
  - http://dbis-uibk.github.io/relax/calc
- Verrà proposto un “homework” il cui svolgimento sarà necessario per partecipare alla prova parziale

RelaX

- Utilizza una sintassi molto simile a quella vista a lezione e sul libro
- L'editor aiuta nella scrittura degli operatori e dei nomi di relazione e di attributo (basta cliccare sul simbolo desiderato)
- Talvolta è utile scrivere direttamente – allora attenzione a maiuscole e minuscole (è "case-sensitive")
- Le espressioni sono talvolta di lettura non semplice, perché tutto su una linea, senza "pedici":
  - scriviamo σ Stipendio>40 (Impiegati) invece di

    $\sigma_{\text{Stipendio>40}}$ (Impiegati)
- Attenzione agli spazi (talvolta lo strumento si confonde) e spesso è utile qualche parentesi in più
- Una differenza nella "assegnazione"; serve una "ridenominazione" esplicita della relazione; invece di

  Capi := Impiegati

  dobbiamo scrivere

  Capi = ρ Capi (Impiegati)

Rappresentazione grafica

- RelaX fornisce anche una rappresentazione grafica delle espressioni sotto forma di albero, molto espressiva
- Ogni operatore è un nodo, con uno o due nodi discendenti (a seconda che abbia uno o due operandi) e le foglie sono relazioni nella base di dati
- Nei lucidi seguenti sono mostrate le interrogazioni discusse in aula e per ciascuna è mostrata la formulazione mostrata in aula, quelle in RelaX (molto simile) e l'albero generato da RelaX

Dati

- Accedendo al servizio si possono specificare interrogazioni su una base di dati
  - fra quelle disponibili sul servizio, oppure
  - su una “caricata” dall’utente
- Per i primi esempi (in questa presentazione), le basi di dati sono state predisposte e possono essere caricate selezionando il link “Select DB ..” (in alto a sinistra) e inserendo nel campo “Load dataset stored in a gist” il relativo link

  1a9dc6cd0f3478388fc177dfc9b5a314 (prima bd)
  b7a8eac38317e0d6a7f0b904a9a10bd3 (seconda bd)
- oppure, più semplicemente richiamando RelaX con l’url:

  http://dbis-uibk.github.io/relax/calc/gist/1a9dc6cd0f3478388fc177dfc9b5a314
  http://dbis-uibk.github.io/relax/calc/gist/b7a8eac38317e0d6a7f0b904a9a10bd3

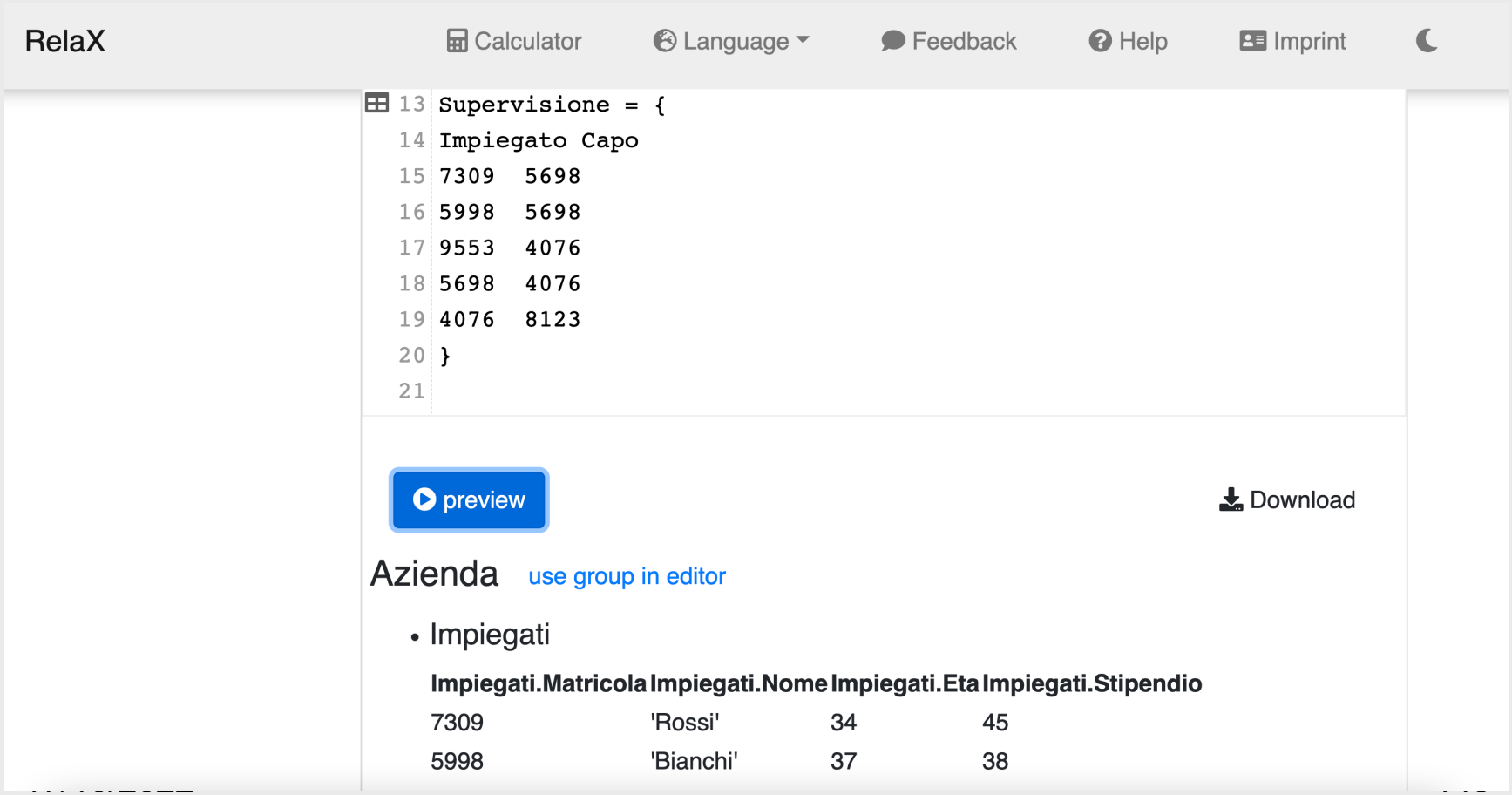
- Ulteriori basi di dati (data-set nella terminologia di RelaX) possono essere predisposti con una sintassi molto semplice e caricati su github (vedere l’help)

```
Impiegati = {
  Matricola, Nome, Eta:number, Stipendio:number
             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
}
```

Metodo alternativo per caricare i dati

- Con il Group Editor: "Preview" e "use group in Editor"



group: Azienda

Impiegati = {

Matricola	Nome	Eta	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

}

Istruzioni per caricare i dati con il group editor

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

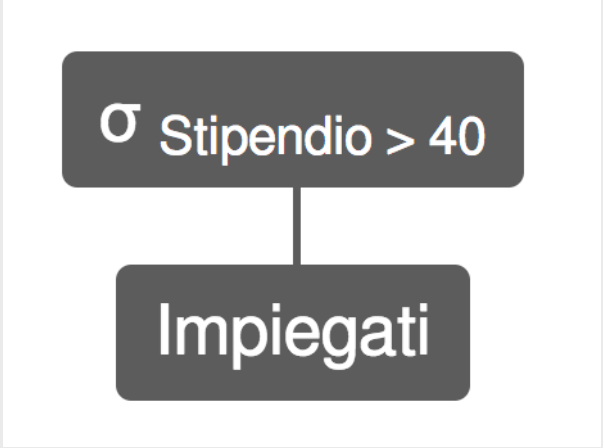
Supervisione

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

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

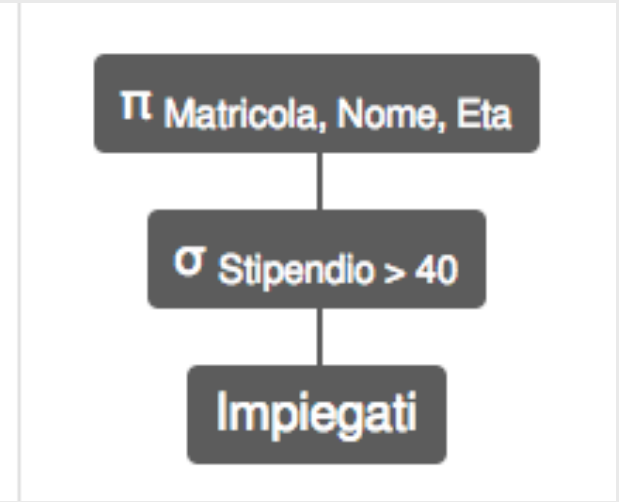
$$\sigma\ Stipendio>40\ (Impiegati))$$



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

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

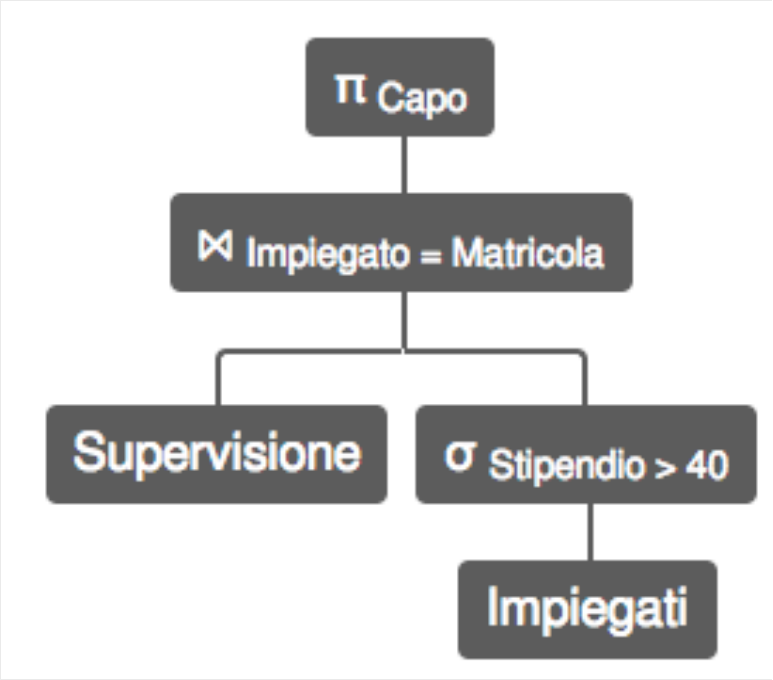
$$\pi\ Matricola,\ Nome,\ Eta\ (\sigma\ Stipendio>40\ (Impiegati))$$



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

$$PROJ_{Capo}\,(Supervisione\ JOIN_{Impiegato=Matricola}\ (SEL_{Stipendio>40}(Impiegati)))$$

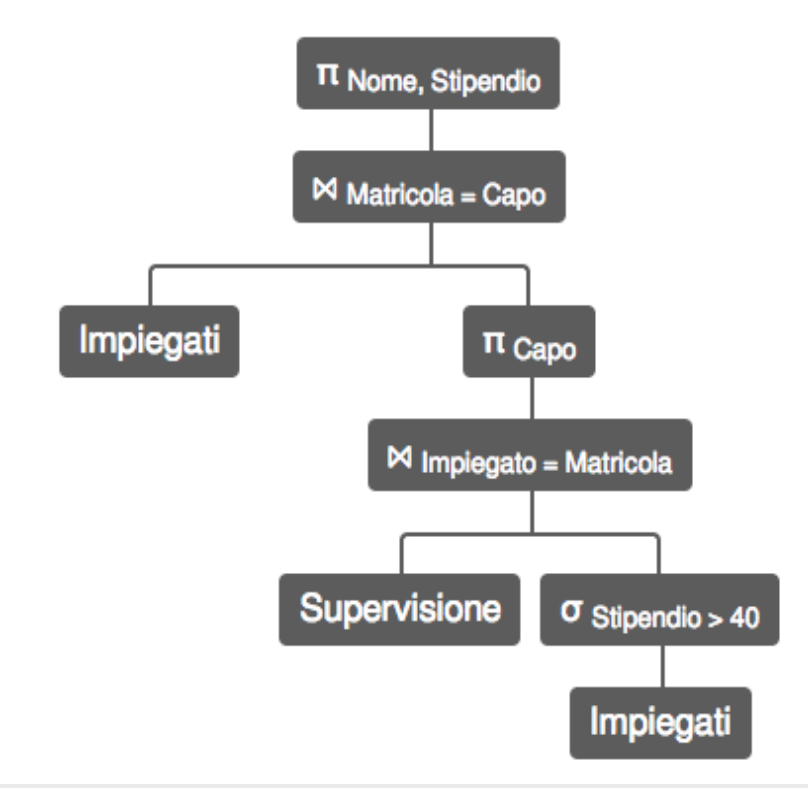
$$\pi_{Capo}\ ((Supervisione) \bowtie_{Impiegato=Matricola} (\sigma_{Stipendio>40}\ (Impiegati)))$$



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

$$PROJ_{Nome,Stipendio}\ (\ Impiegati\ JOIN_{Matricola=Capo}\ PROJ_{Capo}(Supervisione\ JOIN_{Impiegato=Matricola}\ (SEL_{Stipendio>40}(Impiegati))))$$

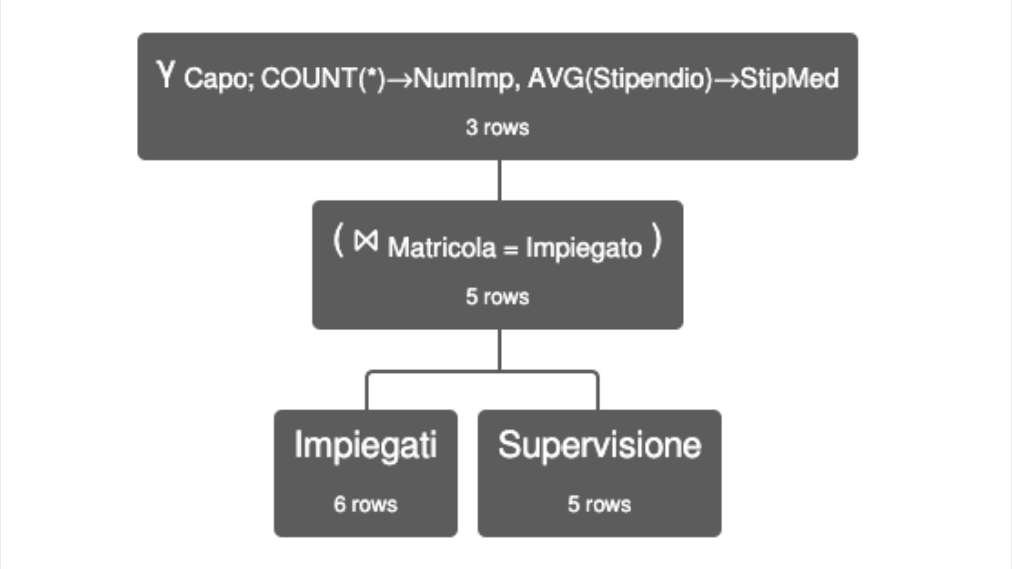
$$\pi_{Nome,Stipendio}\ (\ Impiegati \bowtie_{Matricola = Capo}\ (\pi_{Capo}\ ((Supervisione) \bowtie_{Impiegato=Matricola}\ (\sigma_{Stipendio>40}\ (Impiegati)))))$$



- Per ogni capo, trovare il numero di impiegati e lo stipendio medio

AGG Capo; count(*)→NumImp, avg(Stipendio)→StipMed (
Impiegati JOIN Matricola=Impiegato Supervisione)

γ Capo; count(*)→NumImp, avg(Stipendio)→StipMed
(Impiegati ⋈ Matricola=Impiegato Supervisione)

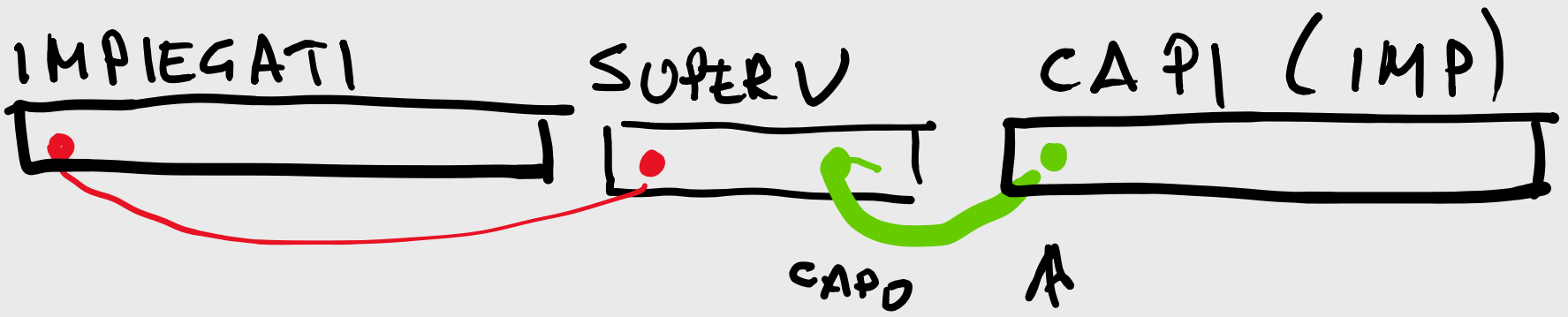


- Trovare gli impiegati che guadagnano più del proprio capo, mostrando matricola, nome e stipendio dell'impiegato e del capo

- un po' complessa, vediamo prima un'altra interrogazione con caratteristiche simili, ma più semplice

- Trovare matricola, nome e stipendio dei capi degli impiegati che guadagnano più di 40; per ciascuno, mostrare, matricola, nome e stipendio anche dell'impiegato

- Il problema:
  - ci interessano, insieme, valori di uno stesso attributo, ma di ennuple diverse



7309 ROSSI 45 7309 5698 5698 BRUNI 42

$$(\text{IMPIEGATI} \bowtie \text{SUPERV}) \bowtie \text{IMPIEGATI}$$

MATR

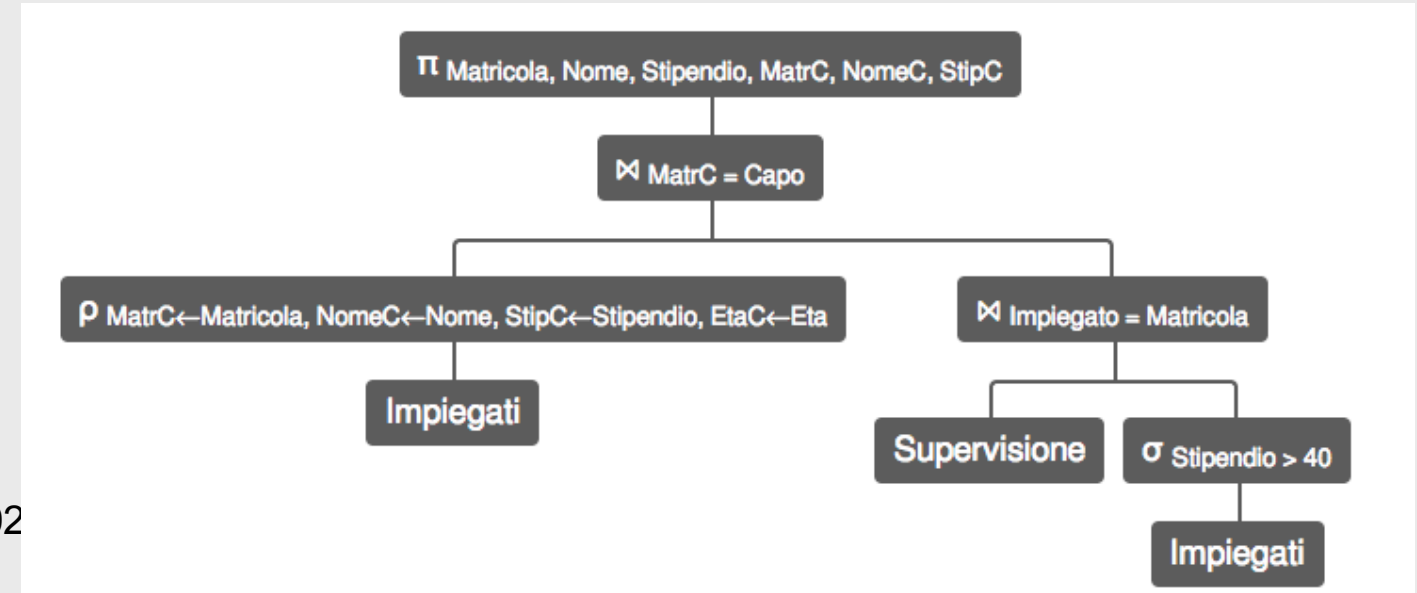
CAPO = MATR

?

- Trovare matricola, nome e stipendio dei capi degli impiegati che guadagnano più di 40; per ciascuno, mostrare, matricola, nome e stipendio anche dell'impiegato

$$\text{PROJ}_{\text{Matr,Nome,Stip,MatrC,NomeC,StipC}}$$
$$(\text{REN}_{\text{MatrC,NomeC,StipC,EtàC} \leftarrow \text{ Matr,Nome,Stip,Età}}(\text{Impiegati})$$
$$\text{JOIN }_{\text{MatrC=Capo}}$$
$$(\text{Supervisione JOIN }_{\text{Impiegato=Matricola}} \text{ SEL}_{\text{Stipendio>40}}(\text{Impiegati})))$$

$$\pi \text{ Matricola, Nome, Stipendio, MatrC, NomeC, StipC}$$
$$(\rho \text{ MatrC} \leftarrow \text{Matricola, NomeC} \leftarrow \text{Nome, StipC} \leftarrow \text{Stipendio, EtaC} \leftarrow \text{Eta (Impiegati)}$$
$$\bowtie \text{ MatrC = Capo}$$
$$(((\text{Supervisione}) \bowtie \text{ Impiegato=Matricola } (\sigma \text{ Stipendio>40 (Impiegati)})))$$



- La notazione con le ridenominazioni, pur corretta, è un po’ troppo “verbosa"
- Ne vediamo un’altra, basata sulle viste

Una convenzione e notazione alternativa per i join

- Nota: è sostanzialmente l'approccio usato in SQL
- Ignoriamo il join naturale (cioè non consideriamo implicitamente condizioni su attributi con nomi uguali)
- Per "riconoscere" attributi con lo stesso nome gli premettiamo il nome della relazione
- Usiamo viste (o "assegnazioni") per ridenominare le relazioni
  - (ridenominiamo gli attributi solo quando serve per l'unione o per dare nomi significativi nel riusltato)

- Trovare matricola, nome e stipendio dei capi degli impiegati che guadagnano più di 40; per ciascuno, mostrare, matricola, nome e stipendio anche dell'impiegato

$$\text{PROJ}_{\text{Matr,Nome,Stip,MatrC,NomeC,StipC}}$$
$$(\text{REN}_{\text{MatrC,NomeC,StipC,EtàC} \leftarrow \text{ Matr,Nome,Stip,Età}}(\text{Impiegati})$$
$$\text{JOIN }_{\text{MatrC=Capo}}$$
$$(\text{Supervisione JOIN }_{\text{Impiegato=Matricola}} \text{ SEL}_{\text{Stipendio>40}}(\text{Impiegati})))$$

Capi := Imp

$$\text{PROJ}_{\text{Imp.Matr, Imp.Nome, Imp.Stip,Capi.Matr,Capi.Nome, Capi.Stip}} (\text{Capi JOIN}_{\text{Capi.Matr=Capo}} (\text{Sup JOIN}_{\text{Imp=Imp.Matr}} \text{SEL}_{\text{Stipendio>40}}(\text{Imp})))$$

.

RelaX

- Utilizza una sintassi molto simile a quella vista a lezione e sul libro
- L'editor aiuta nella scrittura degli operatori e dei nomi di relazione e di attributo (basta cliccare sul simbolo desiderato)
- Talvolta è utile scrivere direttamente – allora attenzione a maiuscole e minuscole (è "case-sensitive")
- Le espressioni sono talvolta di lettura non semplice, perché tutto su una linea, senza "pedici":
  - scriviamo σ Stipendio>40 (Impiegati) invece di $\sigma_{Stipendio>40}$ (Impiegati)
- Attenzione agli spazi (talvolta il parser si confonde) e spesso è utile qualche parentesi in più
- Una differenza nella "assegnazione"; serve una "ridenominazione" esplicita della relazione; invece di

  Capi := Impiegati

  dobbiamo scrivere

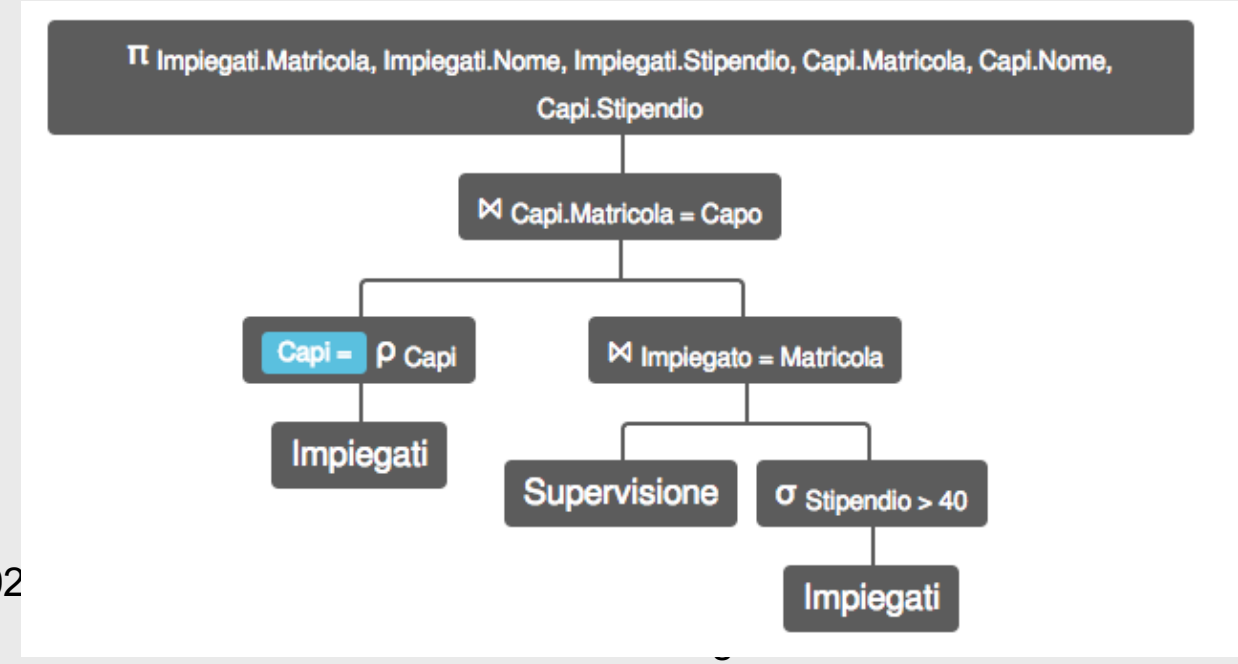
  Capi = ρ Capi (Impiegati)

Capi := Imp

$$PROJ_{Imp.Matr,\ Imp.Nome,\ Imp.Stip,Capi.Matr,Capi.Nome,\ Capi.Stip}$$
$$(Capi\ JOIN_{Capi.Matr=Capo}$$
$$(Supervisione\ JOIN_{Imp=Imp.Matr}\ SEL_{Stipendio>40}(Imp)))$$

Capi = ρ Capi (Impiegati)

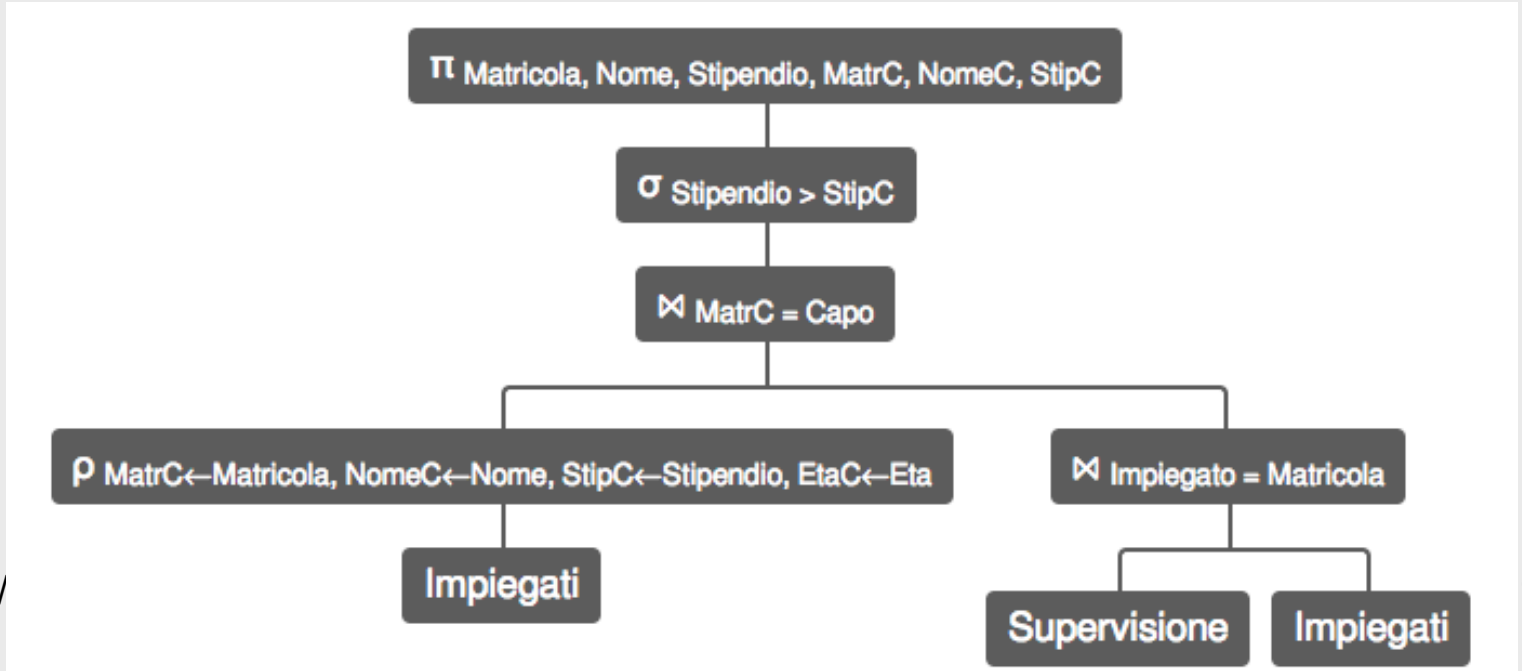
π Impiegati.Matricola, Impiegati.Nome, Impiegati.Stipendio,
Capi.Matricola, Capi.Nome, Capi.Stipendio (Capi⋈ Capi.Matricola = Capo
(((Supervisione) ⋈ Impiegato=Matricola (σ Stipendio>40 (Impiegati)))))



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

$$PROJ_{Matr,Nome,Stip,MatrC,NomeC,StipC}$$
$$(SEL_{Stipendio>StipC}($$
$$REN_{MatrC,NomeC,StipC,EtàC \leftarrow Matr,Nome,Stip,Età}(Impiegati)$$
$$JOIN_{MatrC=Capo}$$
$$(Supervisione\ JOIN_{Impiegato=Matricola}\ Impiegati)))$$

$$\pi\ Matricola, Nome, Stipendio, MatrC, NomeC, StipC$$
$$(\sigma\ Stipendio>StipC$$
$$(\rho\ MatrC \leftarrow Matricola, NomeC \leftarrow Nome, StipC \leftarrow Stipendio, EtaC \leftarrow Eta\ (Impiegati)$$
$$\bowtie\ MatrC = Capo$$
$$((Supervisione) \bowtie Impiegato=Matricola\ (Impiegati))))$$



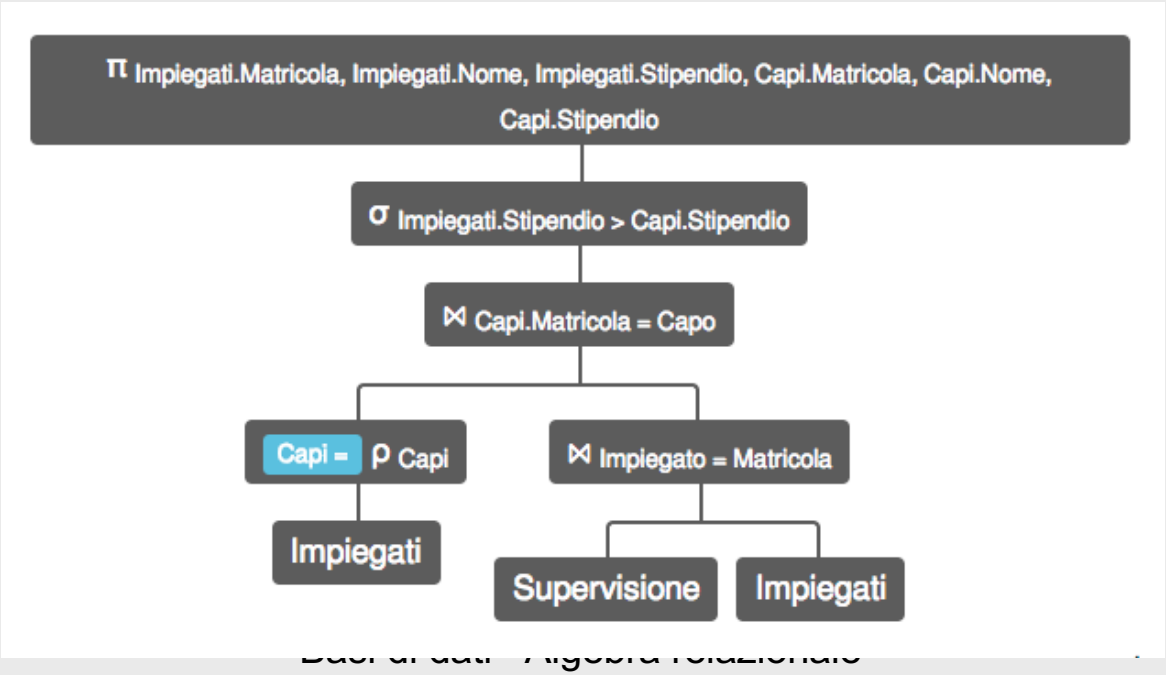
- Trovare gli impiegati che guadagnano più del proprio capo, mostrando matricola, nome e stipendio dell'impiegato e del capo

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

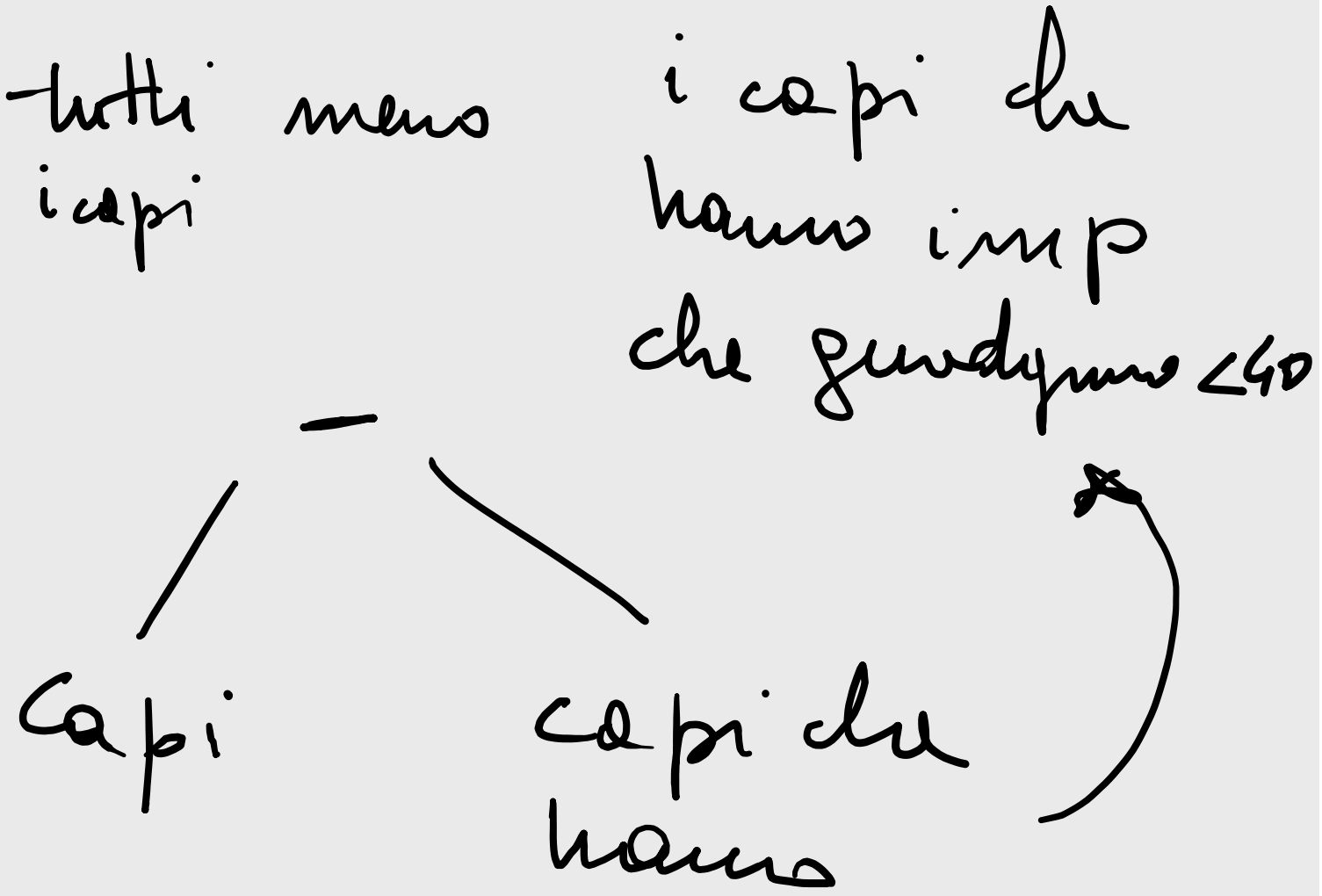
$$PROJ_{Matr,Nome,Stip,MatrC,NomeC,StipC}(SEL_{Stip>StipC}(REN_{MatrC,NomeC,StipC,EtàC \leftarrow Matr,Nome,Stip,Età}(Impiegati) \; JOIN_{MatrC=Capo} (Supervisione \; JOIN_{Imp=Matr} \; Impiegati)))$$

Capi := Impiegati

$$PROJ_{Impiegati.Matr,\ Impiegati.Nome,\ Impiegati.Stip,Capi.Matr,Capi.Nome,\ Capi.Stip}(SEL_{Impiegati.Stip>Capi.Stip}(Capi \; JOIN_{Capi.Matr=Capo} (Supervisione \; JOIN_{Impiegato=Impiegati.Matr} \; Impiegati)))$$



- Trovare le matricole dei capi i cui impiegati guadagnano tutti più di 40

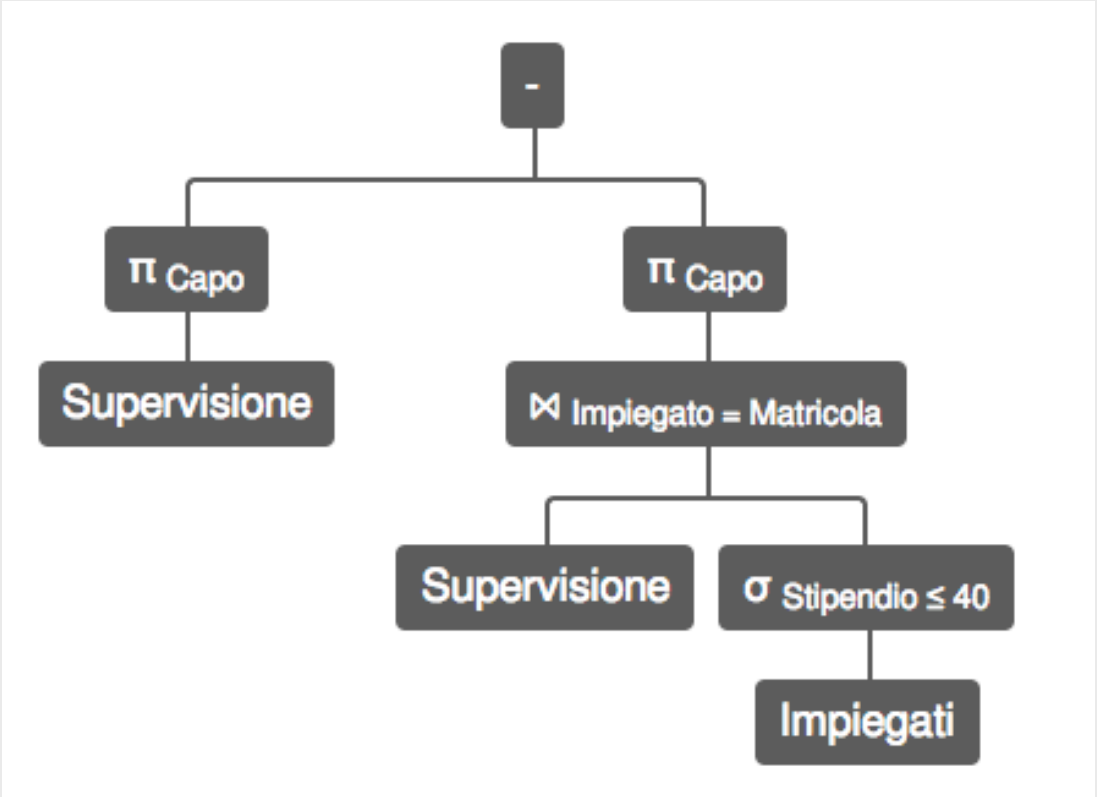


- Trovare le matricole dei capi i cui impiegati guadagnano tutti più di 40
  - tutti i capi, esclusi quelli che hanno impiegati che guadagnano non più di 40
  - con la differenza

- Trovare le matricole dei capi i cui impiegati guadagnano tutti più di 40

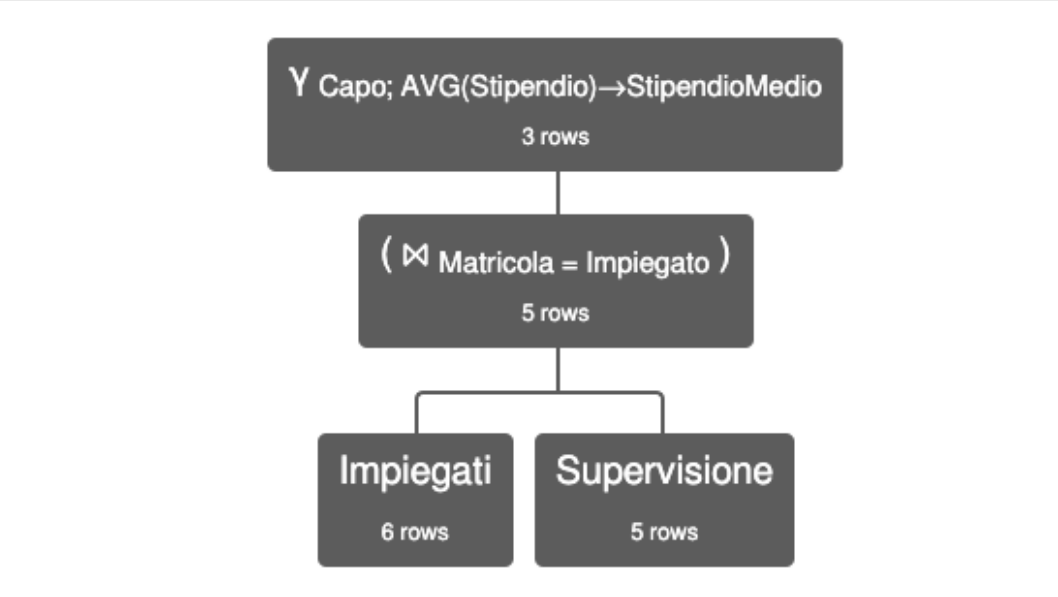
$$PROJ_{Capo}\,(Supervisione) - PROJ_{Capo}\,(Supervisione\ JOIN_{Impiegato=Matricola}\ (SEL_{Stipendio \leq 40}(Impiegati)))$$

$$\pi\ Capo\ (Supervisione) - \pi\ Capo\ (Supervisione \bowtie_{Impiegato=Matricola} (\ \sigma\ Stipendio \leq 40\ (Impiegati)))$$

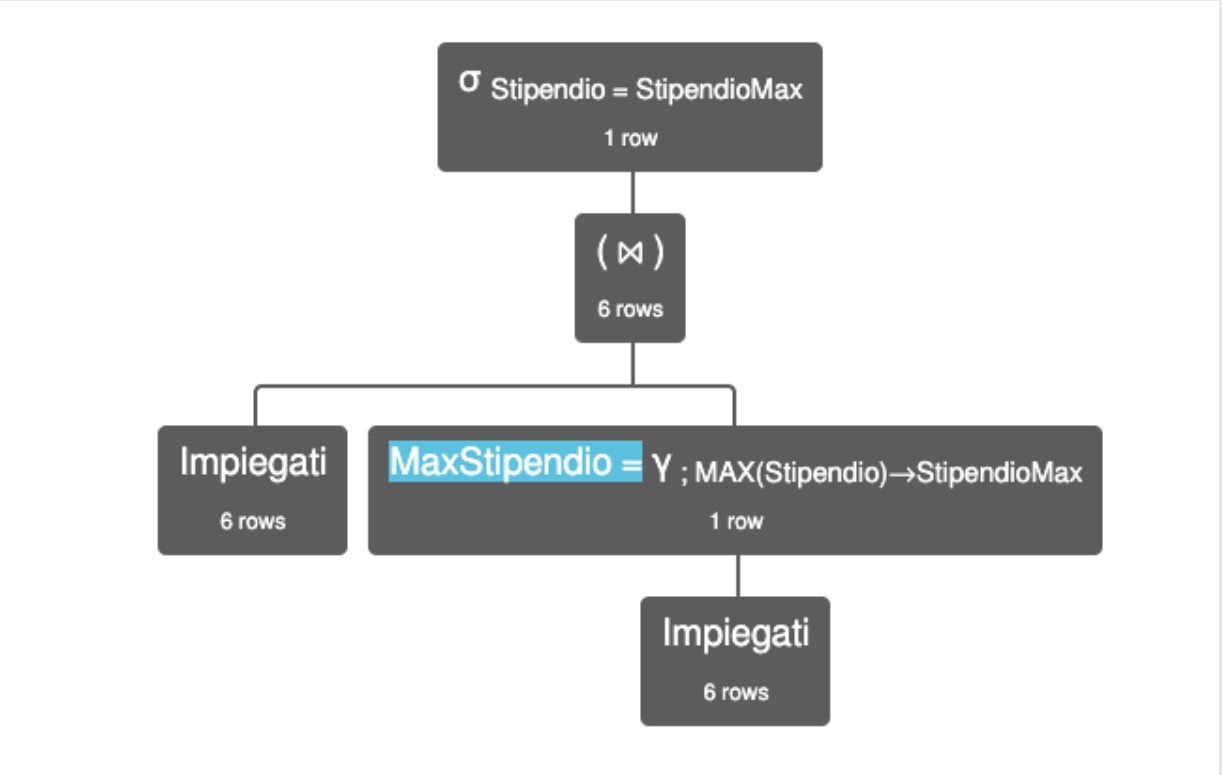


- Trovare, per ciascun capo, la media degli stipendi dei relativi impiegati

  γ Capo; avg(Stipendio)→StipendioMedio (Impiegati ⋈ Matricola=Impiegato Supervisione)



- Trovare l'impiegato (o gli impiegati, se più di uno) con lo stipendio massimo
  - serve una vista

MaxStipendio = γ max(Stipendio)→StipendioMax (Impiegati)

σ Stipendio=StipendioMax (Impiegati ⋈ MaxStipendio)