

Basi di Dati

Ottimizzazione logica

luca.anselma@unito.it

Contenuti - Roadmap

Lab (progettazione)	Corso di Teoria	Lab (SQL)
<ul style="list-style-type: none">• Metodologie e modello Entità Associazioni• Progettazione concettuale e logica	<ul style="list-style-type: none">• Modello Relazionale• Algebra relazionale• Ottimizzazione logica• Calcolo relazionale• La normalizzazione• Metodi di accesso e indici• Gestione della concorrenza• Gestione del ripristino	<ul style="list-style-type: none">• Linguaggio SQL

Outline

- Ottimizzazione e differenza tra ottimizzazione logica e ottimizzazione fisica
- Euristica dell'ottimizzatore logico
- Passi 1, ..., 6 dell'algoritmo di ottimizzazione
- Stima dei costi (selezione, join)
- Passo 7 dell'algoritmo di ottimizzazione

Base di dati «Ricoveri»

pazienti

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

reparti

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

ricoveri

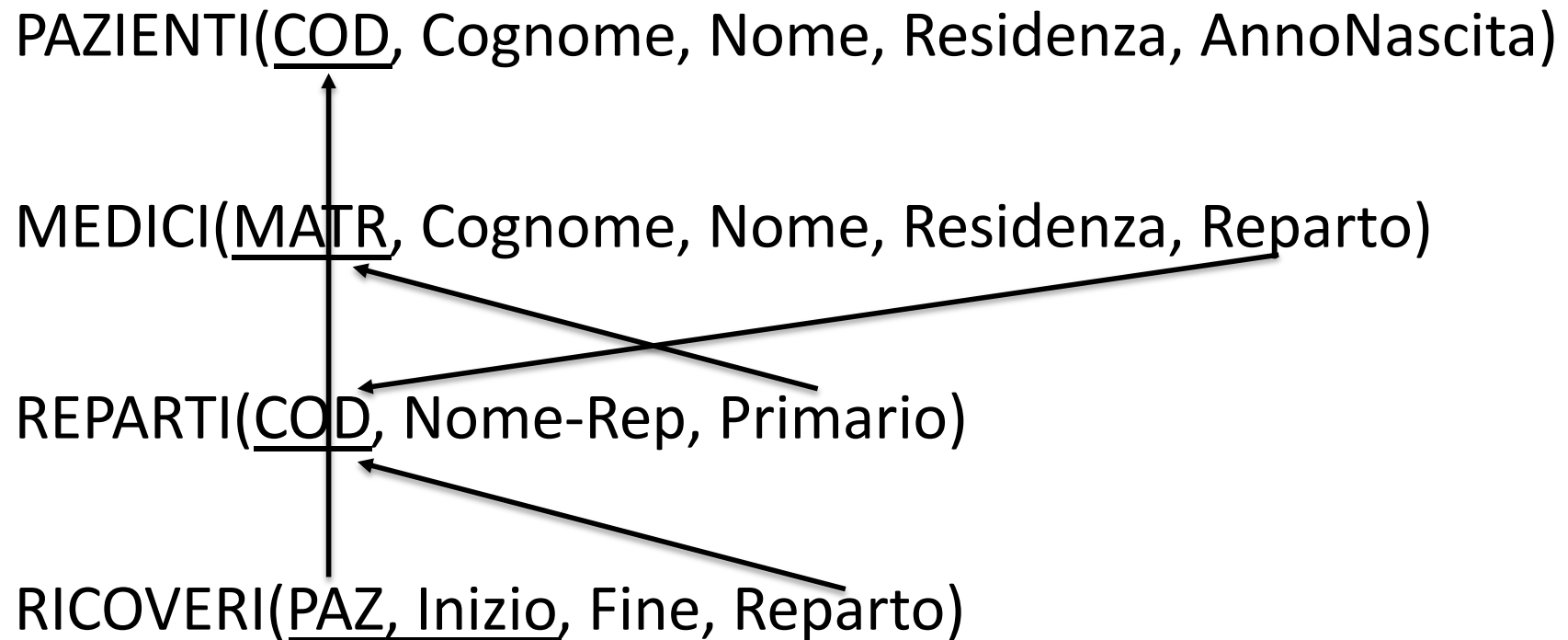
<u>PAZ</u>	Inizio	Fine	Reparto
A102	2/05/2014	9/05/2014	A
A102	2/12/2004	2/01/2005	A
S555	5/10/2014	3/12/2014	B
B444	1/12/2004	2/01/2005	B
S555	6/09/2015	1/11/2015	A

medici

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

Base di dati «Ricoveri»

- Schema relazionale con vincoli di integrità referenziale



Ottimizzazione delle interrogazioni

- Ottimizzare: fare in modo che l'implementazione dell'interrogazione risulti più efficiente possibile
- Ci occuperemo di ottimizzazione in tempo più che di ottimizzazione in memoria/spazio

Contesto

- Un'applicazione gira in memoria centrale e dialoga con il DBMS
- Il database è memorizzato in memoria secondaria
- Il DBMS «dialoga» con le periferiche di storage attraverso il gestore del buffer (che gestisce pagine di memoria)
- Le pagine sono mappate sul dispositivo di storage
- I tempi di accesso alla **memoria centrale** sono dell'ordine dei nanosecondi (**10^{-9} secondi**)
- I tempi di accesso alla **memoria secondaria** sono dell'ordine dei millisecondi (**10^{-3} secondi**)

Contesto

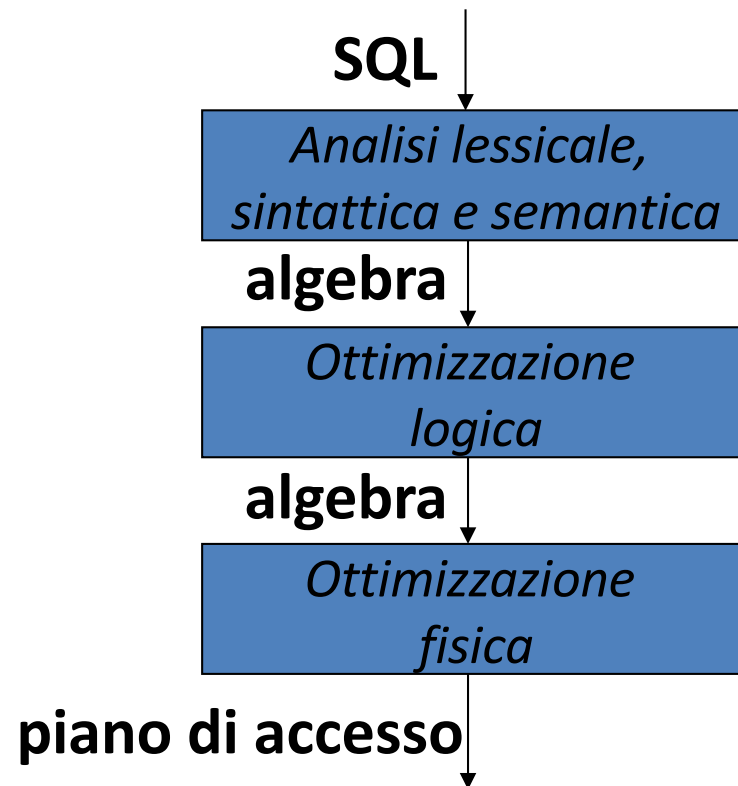
- Quando l'applicazione ha bisogno di una tupla chiede al DBMS di fornirgliela
- **Il DBMS deve trasferire una pagina dalla periferica di storage alla memoria centrale (buffer)**
- Quando la pagina con la tupla richiesta è nel buffer, l'applicazione può elaborarla

Contesto

La complessità computazionale delle attività "algorithmiche" del DBMS in memoria principale è contenuta.

Se i DBMS vogliono minimizzare i tempi di risposta devono **minimizzare soprattutto il numero di pagine trasferite da e verso memoria secondaria.**

Ottimizzazione in due fasi



Ottimizzazione *logica*

L'ottimizzazione logica è indipendente dalle strutture di memorizzazione

Prende in input l'albero sintattico dell'interrogazione e lo trasforma **sfruttando le proprietà dell'algebra relazionale**

L'albero sintattico in uscita è perfettamente equivalente all'interrogazione originale

Ottimizzazione *fisica*

- L'albero sintattico prodotto in output dall'ottimizzatore logico diventa l'input dell'ottimizzatore fisico
- L'ottimizzatore fisico prende in considerazione le strutture interne di memorizzazione
- L'ottimizzatore fisico entra nei nodi (operatori) dell'albero sintattico, li esamina e in **base alle strutture fisiche sceglie l'algoritmo ottimale per eseguire ogni nodo**
- Alla fine ogni nodo dell'albero sintattico avrà l'algoritmo più adatto per l'esecuzione ottimale dell'operatore algebrico corrispondente

Esempio di albero sintattico

Ricavo i dati dei pazienti ricoverati a Torino nel 2010 curati dal medico 405

$$\pi_{COD, PAZIENTI.Cognome, PAZIENTI.Nome, PAZIENTI.Residenza, Inizio, MATR}(\sigma_{PAZIENTI.Residenza='TO' \wedge Inizio=2010 \wedge MATR='405'}((pazienti \bowtie_{COD=PAZ} ricoveri) \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} medici))$$

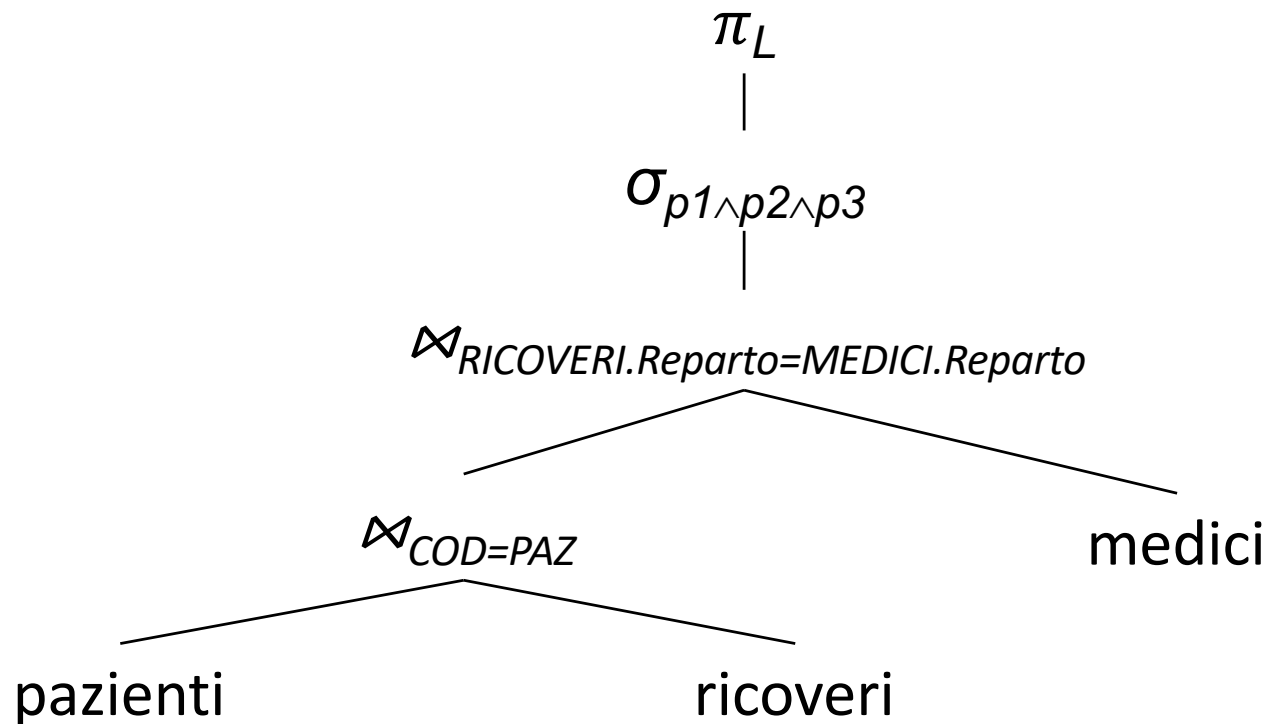
Chiamando

- $L: COD, PAZIENTI.Cognome, PAZIENTI.Nome, PAZIENTI.Residenza, Inizio, MATR$
- $p1: PAZIENTI.Residenza='TO'$
- $p2: Inizio=2010$
- $p3: MATR='405',$

$$\pi_L(\sigma_{p1 \wedge p2 \wedge p3}((pazienti \bowtie_{COD=PAZ} ricoveri) \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} medici))$$

Esempio di albero sintattico

$$\pi_L(\sigma_{p1 \wedge p2 \wedge p3}((pazienti \bowtie_{COD=PAZ} ricoveri) \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} medici))$$



Ottimizzazione logica

L'ottimizzatore è guidato da un semplice principio euristico:

Ridurre la massa di tuple concettualmente coinvolte dall'interrogazione

Infatti, quasi tutte le operazioni principali dei sistemi informativi che usiamo quotidianamente usano pochissimi dati per volta:

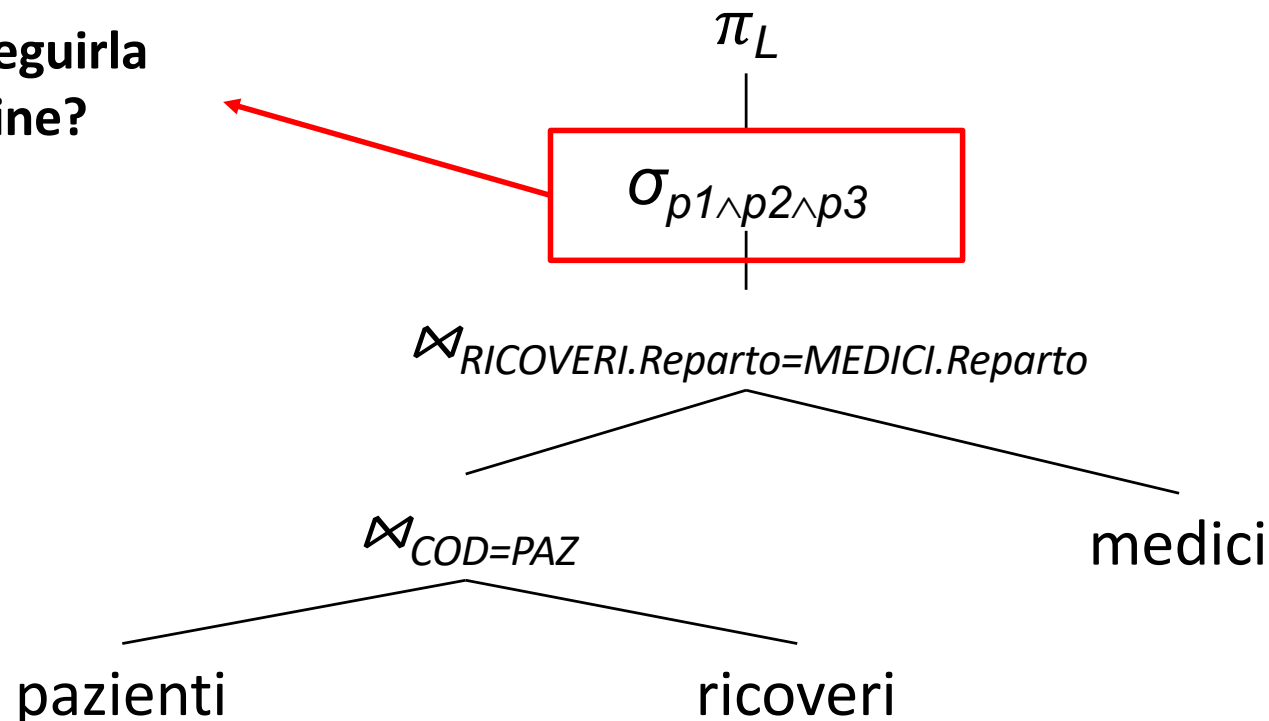
- lavoro sul mio piano di studi, non su tutti
- lavoro sui miei pazienti, non su tutti
- lavoro sulle operazioni del mio conto in banca, non su tutte

Idea: sfruttare le proprietà distributive della selezione in modo che sia uno dei primi operatori da eseguire, riducendo subito il numero di tuple coinvolte.

Esempio di albero

$$\pi_L(\sigma_{p1 \wedge p2 \wedge p3}((pazienti \bowtie_{COD=PAZ} ricoveri) \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} medici))$$

Perché eseguirla
solo alla fine?



Algoritmo di ottimizzazione logica

I predicati p della selezione sono in forma congiuntiva (non è una limitazione: si può ricondurre qualsiasi predicato a una congiunzione di disgiunzioni)

1. Decomporre gli AND

$$- \sigma_{p1 \wedge p2}(r) \rightarrow \sigma_{p1}(\sigma_{p2}(r))$$

2. Trasferire le **selezioni verso le foglie** finché è possibile con le proprietà distributive della selezione

3. Trasferire le **proiezioni verso le foglie** finché è possibile con le proprietà distributive della proiezione

Algoritmo di ottimizzazione logica

4. Ricondurre a un'**unica selezione** le selezioni multiple nello stesso nodo dell'albero

– $\sigma_{p1}(\sigma_{p2}(r)) \rightarrow \sigma_{p1 \wedge p2}(r)$

5. Riconoscere le **sequenze di join**

– $\sigma_{\theta}(r \times s) \rightarrow r \bowtie_{\theta} s$

6. Ricondurre a un'**unica proiezione** le proiezioni multiple

– $\pi_X(\pi_{X \cup Y}(r)) \rightarrow \pi_X(r)$

7. Esaminare le varianti dell'albero sintattico dovute alle proprietà associative scegliendo la **variante di costo minimo**

Spiegazione del passo 1

I predicati p della selezione sono in forma congiuntiva (non è una limitazione: si può ricondurre qualsiasi predicato a una congiunzione di disgiunzioni)

1. Decomporre gli AND

$$- \quad \sigma_{p1 \wedge p2}(r) \rightarrow \sigma_{p1}(\sigma_{p2}(r))$$

2. Trasferire le selezioni verso le foglie finché è possibile con le proprietà distributive della selezione
3. Trasferire le proiezioni verso le foglie finché è possibile con le proprietà distributive della proiezione

Spiegazione del passo 1

La proprietà distributiva della selezione rispetto al join:

$$\sigma_p(r(A) \bowtie_{\theta} s(B)) \rightarrow \sigma_p(r(A)) \bowtie_{\theta} s(B)$$

è valida **solo se** gli attributi coinvolti da p sono contenuti solo in A .

Se p coinvolge sia attributi di A che attributi di B , non si può applicare.

Quindi a una selezione con un predicato che pesca da più relazioni difficilmente può essere applicata la proprietà distributiva...

Spiegazione del passo 1

Infatti, nell'esempio ho

$$\sigma_{p1 \wedge p2 \wedge p3}$$

con

*p1: PAZIENTI.Residenza='TO' (che usa la relazione **pazienti**)*

*p2: Inizio=2010 (che usa la relazione **ricoveri**)*

*p3: MATR='405' (che usa la relazione **medici**)*

Questo predicato, preso insieme, non consente di muovere la selezione da dove è sfruttando la proprietà distributiva della selezione rispetto al join

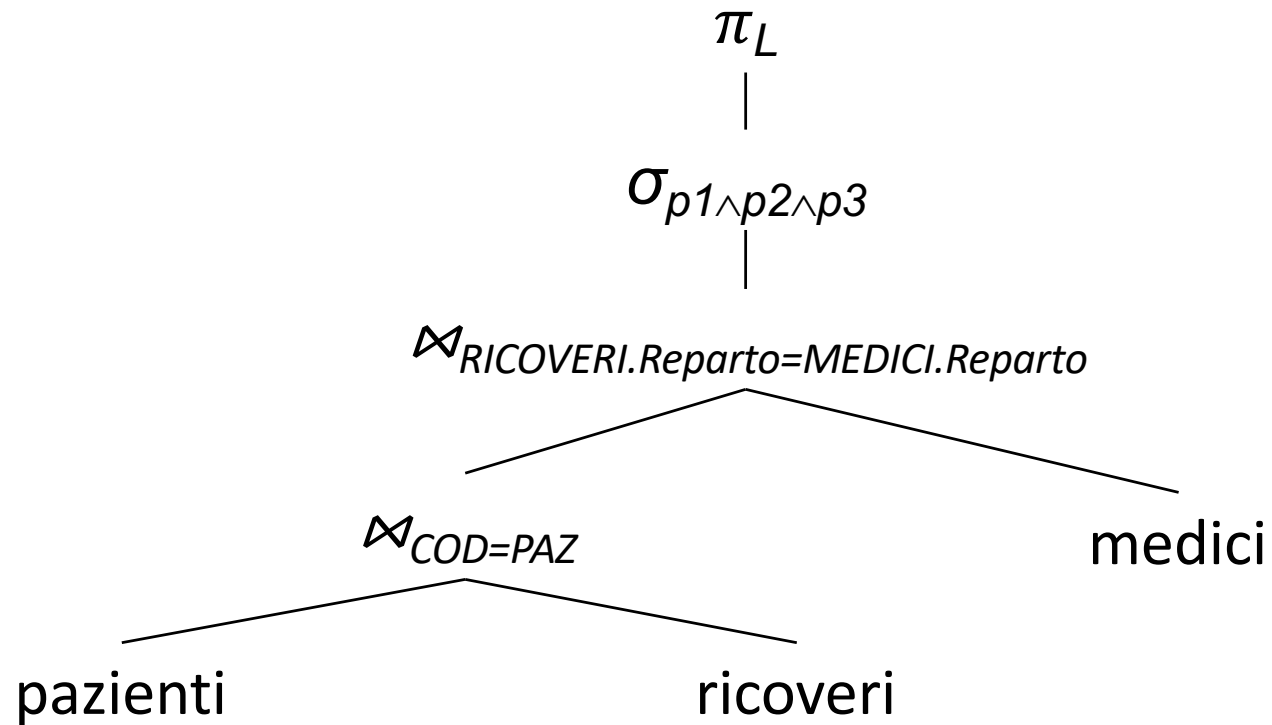
Spiegazione del passo 1

Ma se la spezzo

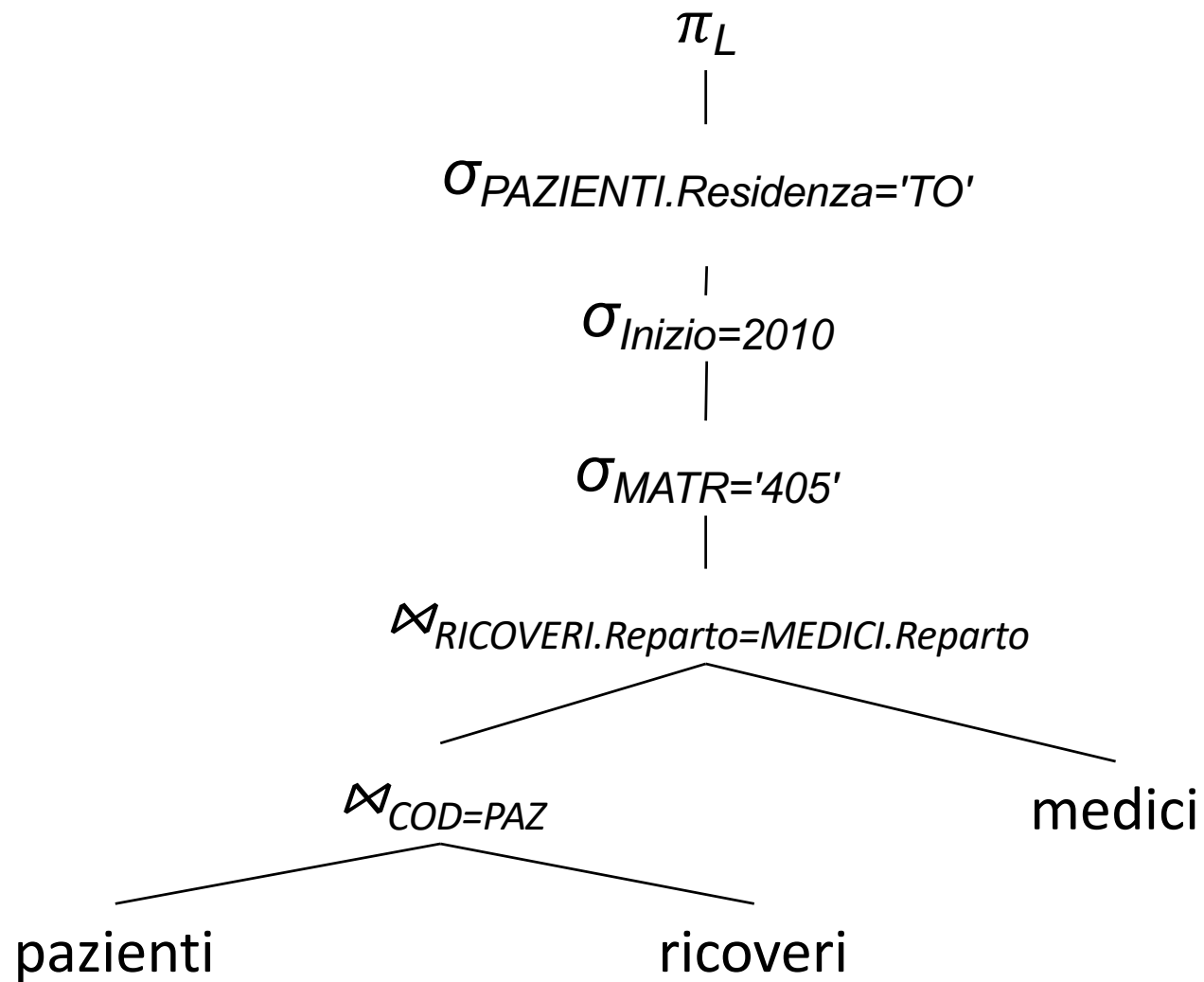
$$\sigma_{p1 \wedge p2 \wedge p3} = \sigma_{p1}(\sigma_{p2}(\sigma_{p3}(...)))$$

i singoli predicati sono più semplici e aumento la probabilità che essi soddisfino la condizione necessaria all'applicazione della proprietà distributiva

Esecuzione del passo 1



Esecuzione del passo 1



Spiegazione del passo 2

I predicati p della selezione sono in forma congiuntiva (non è una limitazione: si può ricondurre qualsiasi predicato a una congiunzione di disgiunzioni)

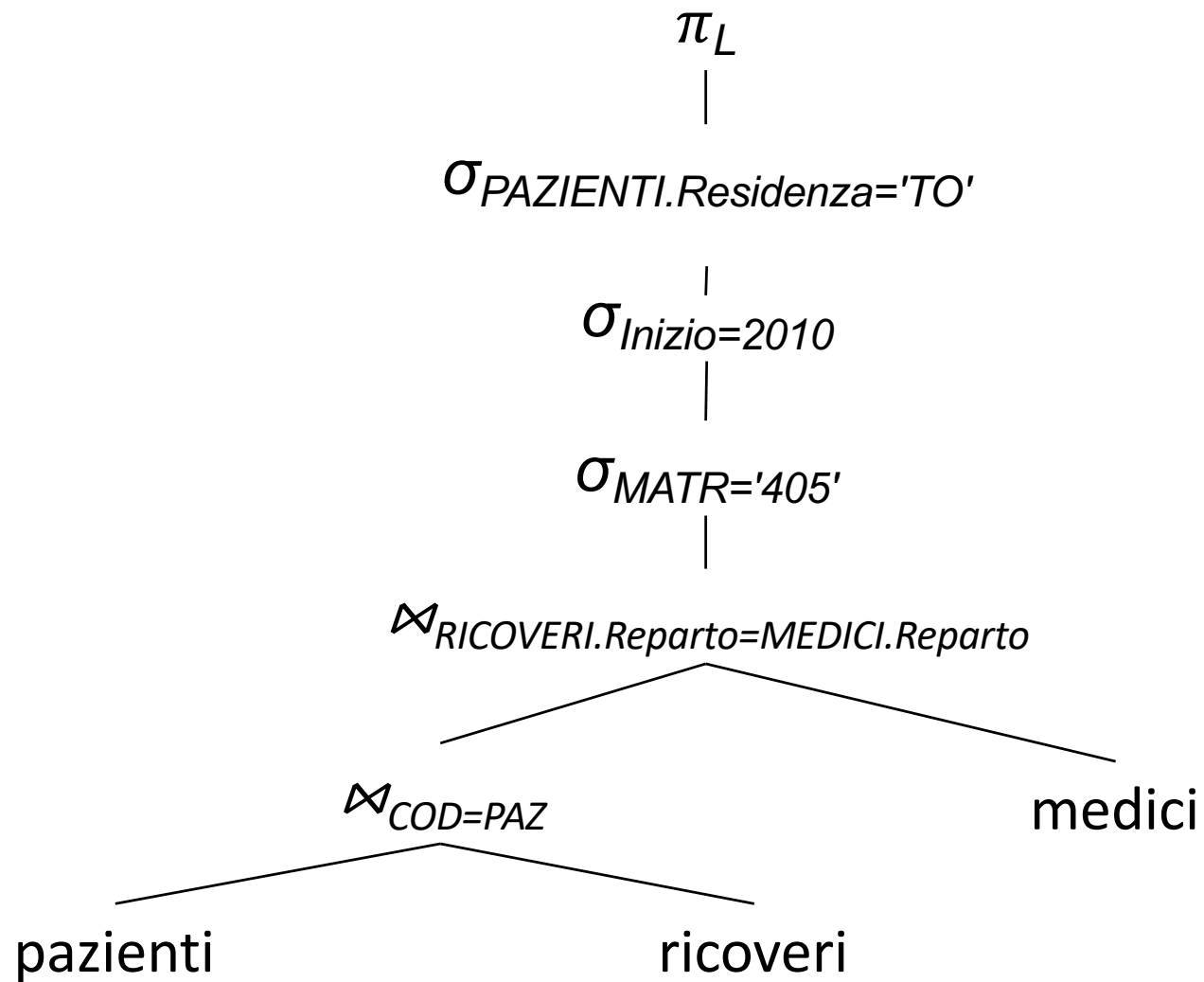
1. Decomporre gli AND

$$- \sigma_{p1 \wedge p2}(r) \rightarrow \sigma_{p1}(\sigma_{p2}(r))$$

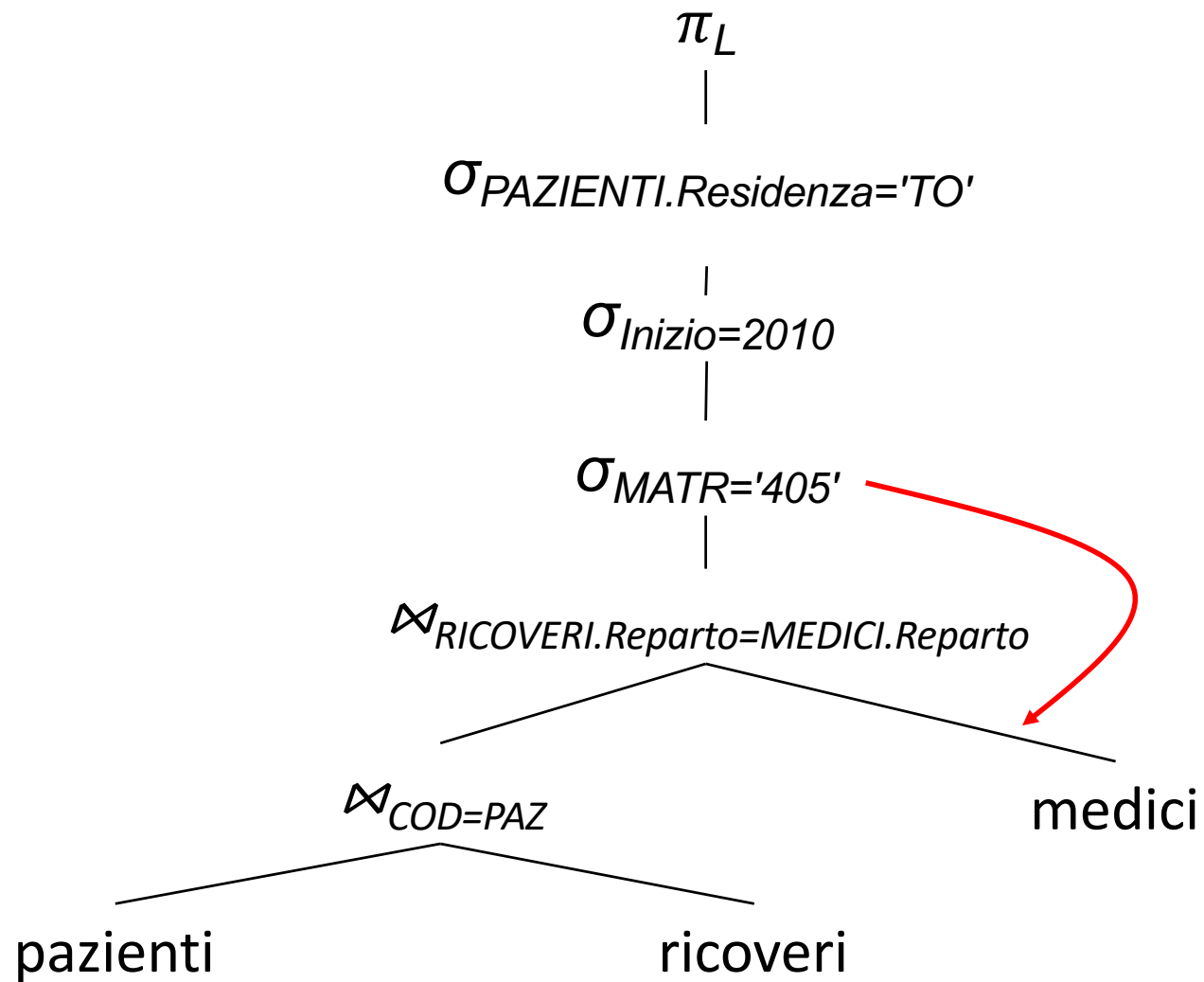
2. Trasferire le selezioni verso le foglie finché è possibile con le proprietà distributive della selezione

3. Trasferire le proiezioni verso le foglie finché è possibile con le proprietà distributive della proiezione

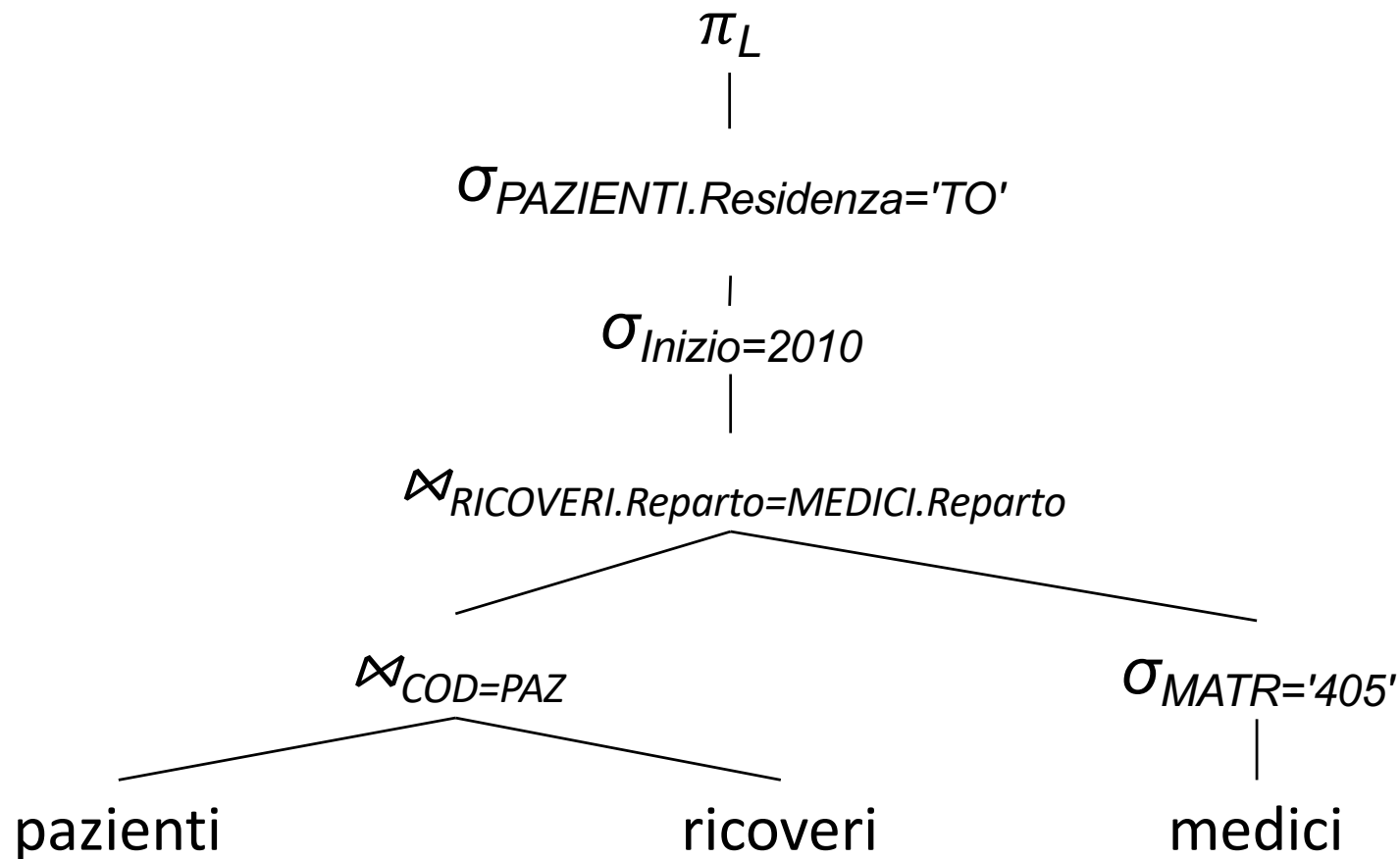
Esecuzione del passo 2



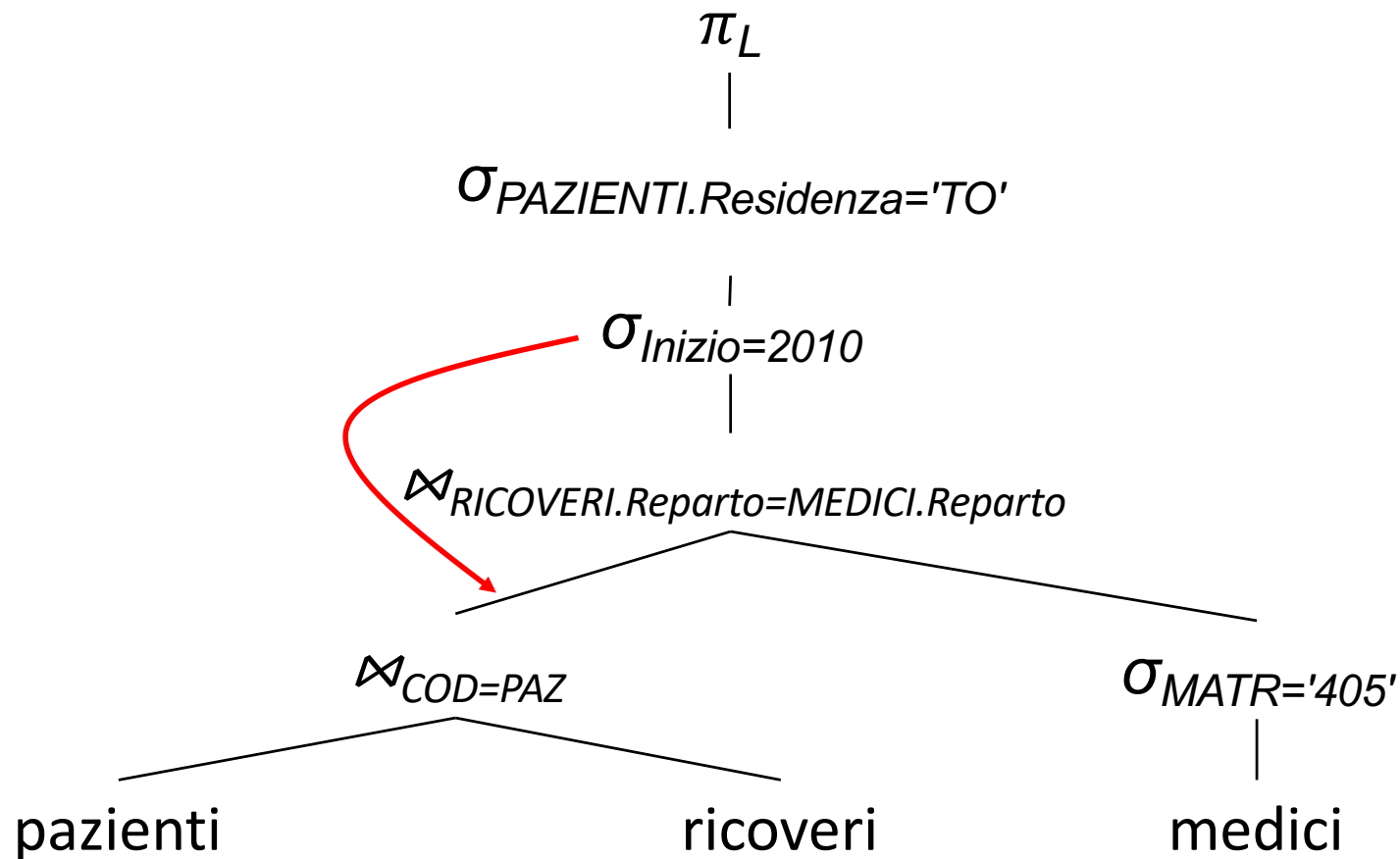
Esecuzione del passo 2



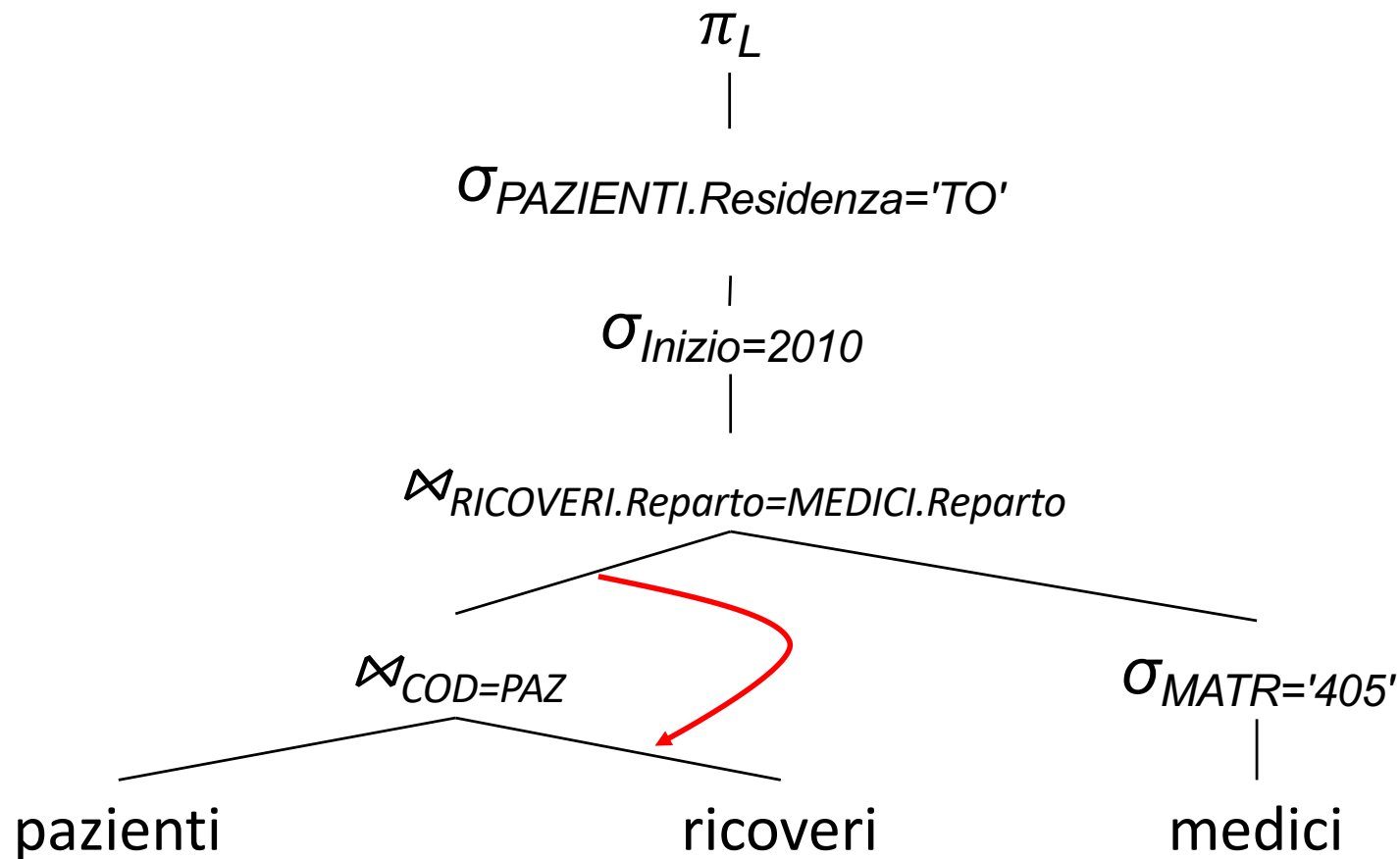
Esecuzione del passo 2



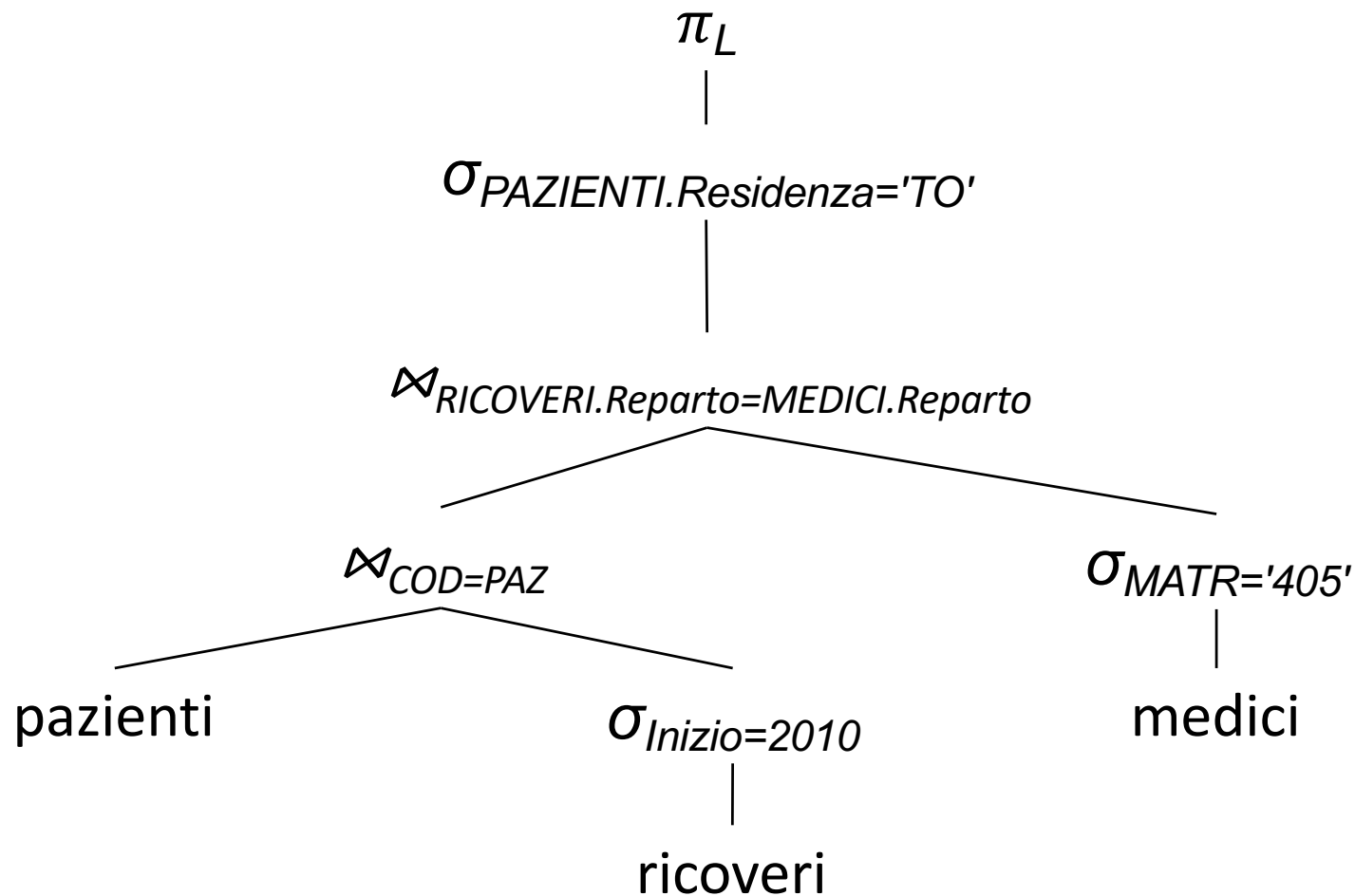
Esecuzione del passo 2



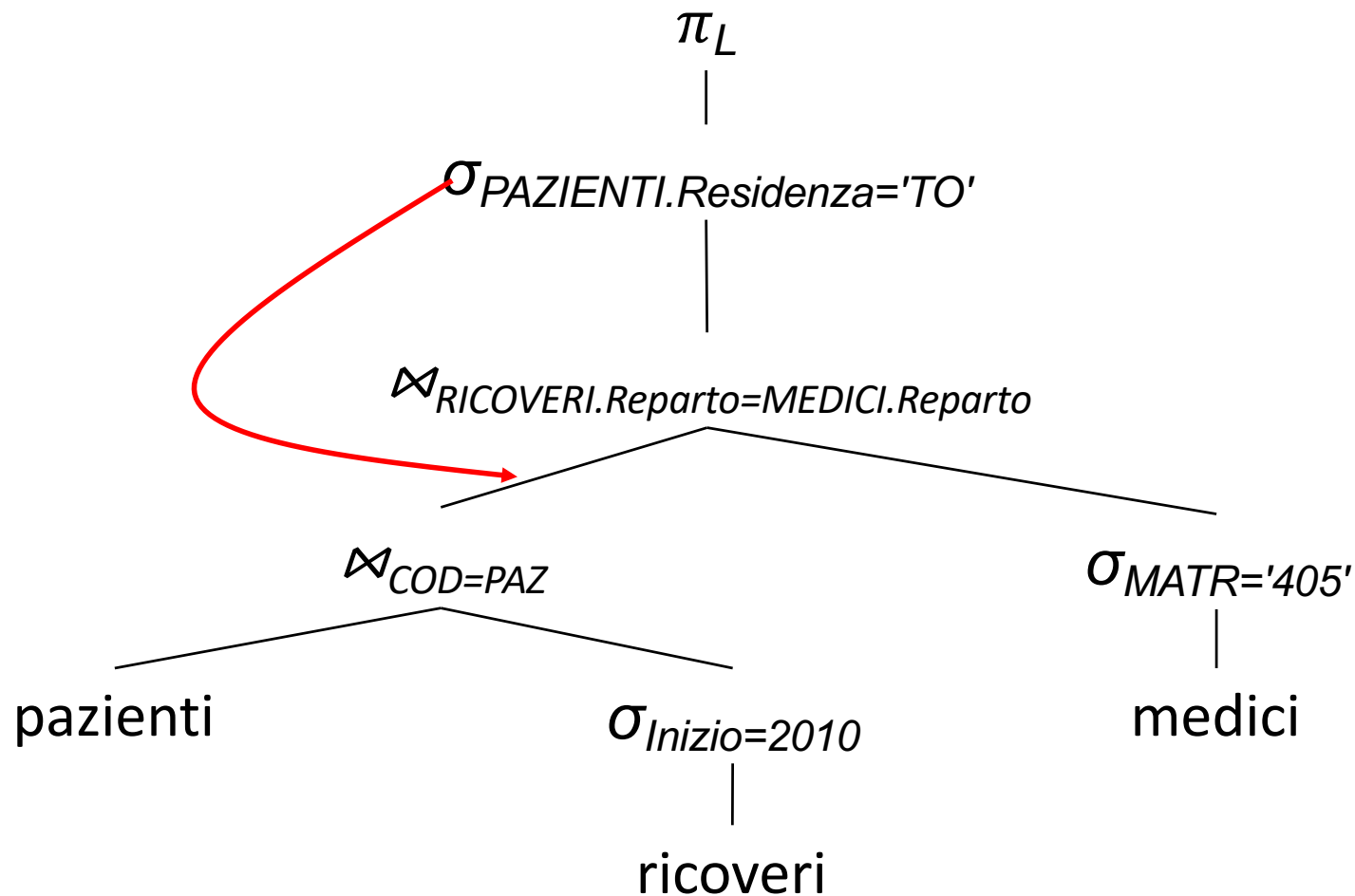
Esecuzione del passo 2



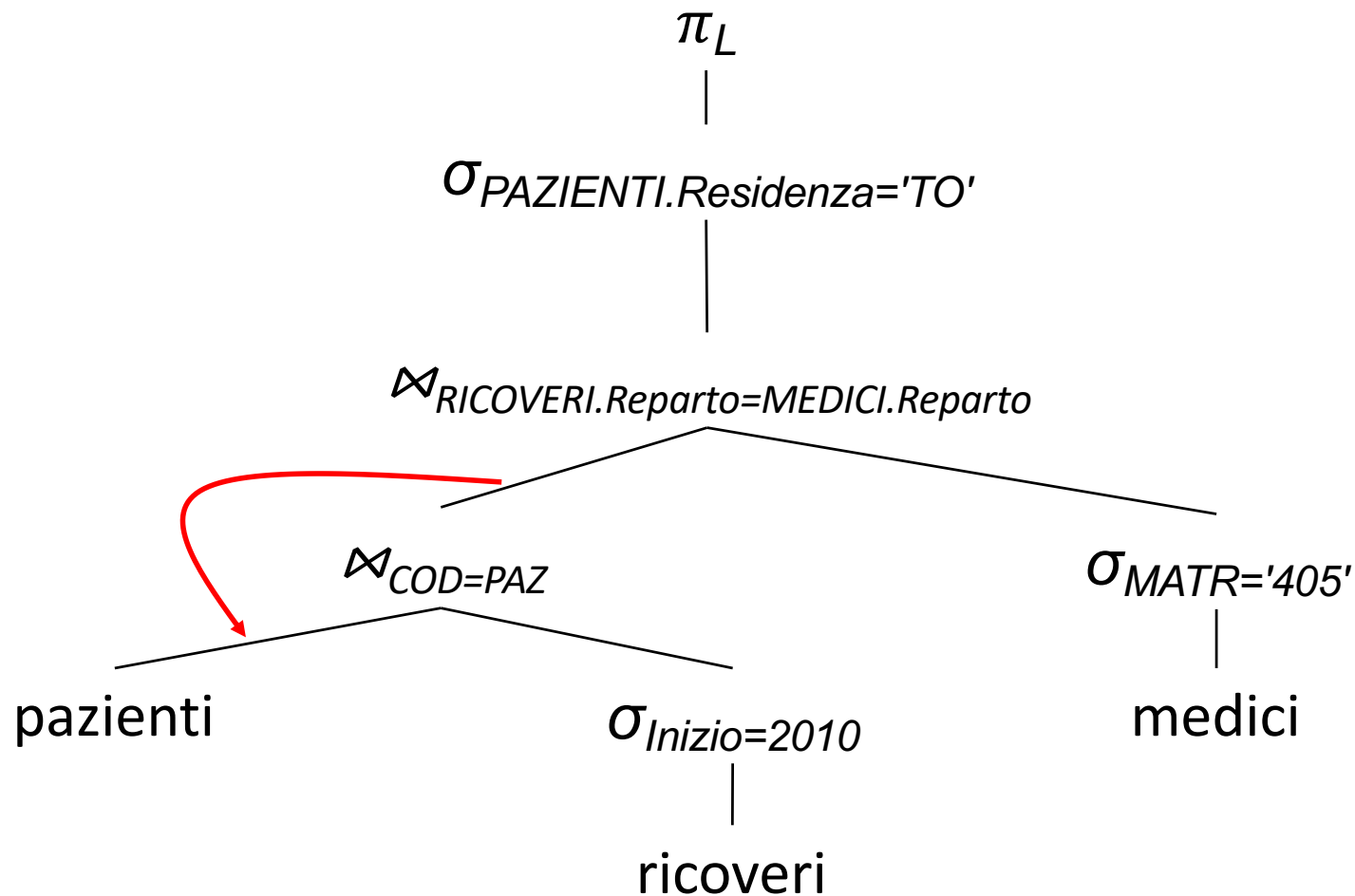
Esecuzione del passo 2



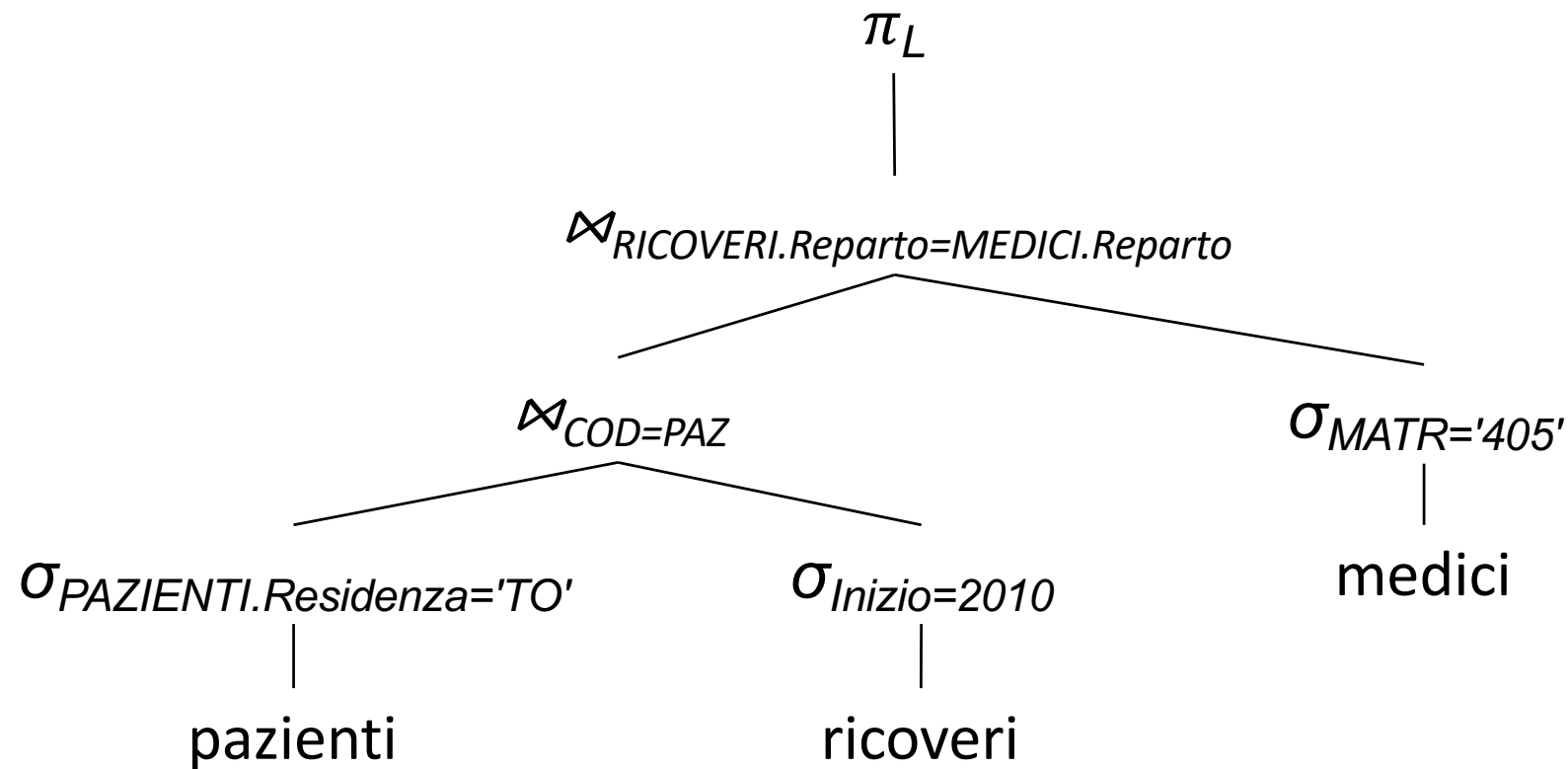
Esecuzione del passo 2



Esecuzione del passo 2



Esecuzione del passo 2



Spiegazione del passo 3

I predicati p della selezione sono in forma congiuntiva (non è una limitazione: si può ricondurre qualsiasi predicato a una congiunzione di disgiunzioni)

1. Decomporre gli AND

$$- \sigma_{p1 \wedge p2}(r) \rightarrow \sigma_{p1}(\sigma_{p2}(r))$$

2. Trasferire le selezioni verso le foglie finché è possibile con le proprietà distributive della selezione

3. Trasferire le proiezioni verso le foglie finché è possibile con le proprietà distributive della proiezione

Spiegazione del passo 3

- Simile al passo 2

Spiegazione del passo 4

4. Ricondurre a un'unica selezione le selezioni multiple

- $\sigma_{p1}(\sigma_{p2}(r)) \rightarrow \sigma_{p1 \wedge p2}(r)$

5. Riconoscere le sequenze di join

- $\sigma_{\theta}(r \times s) \rightarrow r \bowtie_{\theta} s$

6. Ricondurre a un'unica proiezione le proiezioni multiple

- $\pi_X(\pi_{X \cup Y}(r)) \rightarrow \pi_X(r)$

7. Esaminare le varianti dell'albero sintattico dovute alle proprietà associative scegliendo la variante di costo minimo

Spiegazione del passo 4

- A forza di trasferire le selezioni verso le foglie, le selezioni si fermano sui sottoalberi di competenza
- Potremmo quindi trovare delle selezioni in cascata (selezioni multiple)
- L'ottimizzatore ricompone insieme le selezioni multiple, applicando una sola selezione con una congiunzione di predicati

Spiegazione del passo 5

4. Ricondurre a un'unica selezione le selezioni multiple

$$- \sigma_{p1}(\sigma_{p2}(r)) \rightarrow \sigma_{p1 \wedge p2}(r)$$

5. Riconoscere le sequenze di join

$$- \sigma_{\theta}(r \times s) \rightarrow r \bowtie_{\theta} s$$

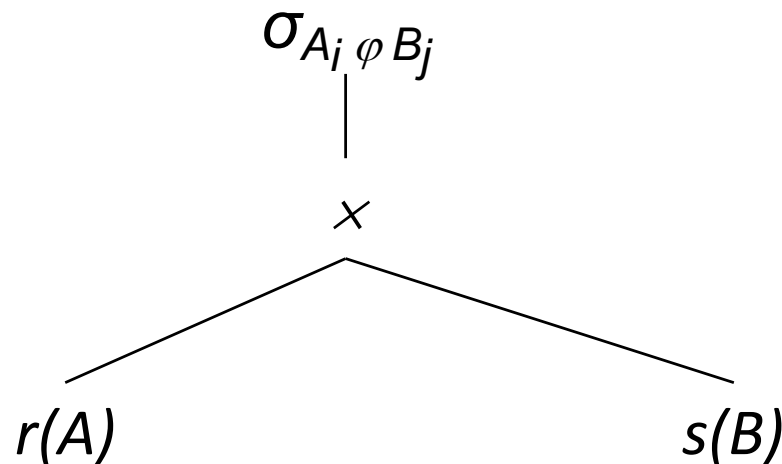
6. Ricondurre a un'unica proiezione le proiezioni multiple

$$- \pi_X(\pi_{X \cup Y}(r)) \rightarrow \pi_X(r)$$

7. Esaminare le varianti dell'albero sintattico dovute alle proprietà associative scegliendo la variante di costo minimo

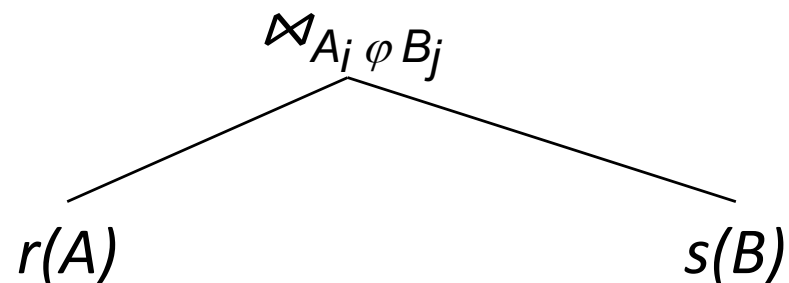
Spiegazione del passo 5

Capita spesso, anche in SQL, di trovarsi con delle selezioni che hanno come sottoalbero un prodotto cartesiano



Spiegazione del passo 5

Il passo 5 trasforma quindi il la selezione+prodotto cartesiano in un theta-join (tutti i DBMS hanno infatti algoritmi ottimizzati per eseguire i join)



Spiegazione del passo 6

4. Ricondurre a un'unica selezione le selezioni multiple
 - $\sigma_{p1}(\sigma_{p2}(r)) \rightarrow \sigma_{p1 \wedge p2}(r)$
5. Riconoscere le sequenze di join
 - $\sigma_{\theta}(r \times s) \rightarrow r \bowtie_{\theta} s$
- 6. Ricondurre a un'unica proiezione le proiezioni multiple**
 - $\pi_X(\pi_{X \cup Y}(r)) \rightarrow \pi_X(r)$
7. Esaminare le varianti dell'albero sintattico dovute alle proprietà associative scegliendo la variante di costo minimo

Spiegazione del passo 6

La spiegazione è ovvia!

Spiegazione del passo 7

4. Ricondurre a un'unica selezione le selezioni multiple
 - $\sigma_{p1}(\sigma_{p2}(r)) \rightarrow \sigma_{p1 \wedge p2}(r)$
5. Riconoscere le sequenze di join
 - $\sigma_{\theta}(r \times s) \rightarrow r \bowtie_{\theta} s$
6. Ricondurre a un'unica proiezione le proiezioni multiple
 - $\pi_X(\pi_{X \cup Y}(r)) \rightarrow \pi_X(r)$
7. **Esaminare le varianti dell'albero sintattico dovute alle proprietà associative scegliendo la variante di costo minimo**

Spiegazione del passo 7

Nel nostro caso di studio l'ottimizzatore esplora due alternative:

- $\pi_L((\sigma_{p1}(\textit{pazienti}) \bowtie_{\textit{COD=PAZ}} \sigma_{p2}(\textit{ricoveri})) \bowtie_{\textit{RICOVERI.Reparto=MEDICI.Reparto}} \sigma_{p3}(\textit{medici}))$
- $\pi_L(\sigma_{p1}(\textit{pazienti}) \bowtie_{\textit{COD=PAZ}} (\sigma_{p2}(\textit{ricoveri}) \bowtie_{\textit{RICOVERI.Reparto=MEDICI.Reparto}} \sigma_{p3}(\textit{medici})))$

Come scegliere?

Introduciamo un approccio alternativo che stima **quantitativamente** i costi delle varie alternative

Aspetti quantitativi delle interrogazioni

I DBMS mantengono nel dizionario dei dati una serie di informazioni di tipo statistico su ogni tabella r , in particolare:

- **$CARD(r)$** $= |r|$ cardinalità della relazione
- **$SIZE(t)$** ampiezza della tupla in byte
- **$VAL(A_i, r)$** numero di valori distinti che appaiono nella colonna A_i all'interno della tabella r , ovvero

$$VAL(A_i, r) = |\pi_{A_i}(r)|$$

- Ad esempio $VAL(Reparto, ricoveri) = 2$
- Se A_i è chiave, $VAL(A_i, r) = CARD(r)$

Aspetti quantitativi delle interrogazioni

I DBMS mantengono nel dizionario dati una serie di informazioni di tipo statistico su ogni tabella r , in particolare:

- **$MIN(A_i, r)$** valore minimo di A_i contenuto in r
- **$MAX(A_i, r)$** valore massimo di A_i contenuto in r
- **$NPAGE(r)$** numero di pagine occupate da r
 $NPAGE(r) = CARD(r) / \text{fattore_di_bloccaggio}$
 - ***fattore_di_bloccaggio** è il numero massimo di tuple contenute in una pagina*

Analisi dei costi delle interrogazioni

L'analisi quantitativa dell'interrogazione permette di predire *ex-ante* il risultato della cardinalità della relazione risultato senza eseguirla

Stima del costo della selezione

Data la selezione $\sigma_p(r)$, conoscendo l'intervallo di variabilità della selezione $\sigma_p(r)$

$$0 \leq |\sigma_p(r)| \leq |r|,$$

si può modellare la cardinalità della selezione $\sigma_p(r)$ con un **fattore di selettività** f_p per la cardinalità di r

$$|\sigma_p(r)| = f_p \cdot |r|$$

Il fattore di selettività f_p è legato al solo predicato p di selezione e varia tra 0 e 1

Fattore di selettività

Il fattore di selettività f_p può essere interpretato come la probabilità che una tupla in r soddisfi il predicato di selezione p , ovvero la stima della percentuale di tuple che soddisfano il predicato di selezione

Come possiamo stimare f_p ?

Fattore di selettività

I DBMS hanno a disposizione un formulario anche molto avanzato per il calcolo del fattore di selettività f_p

Noi ricaviamo una stima di f_p in una versione più grossolana, ma sufficiente ai nostri scopi assumendo:

- una **distribuzione uniforme** dei valori all'interno delle varie colonne, cioè ignoriamo il fatto che alcuni valori possono essere più probabili di altri
- **assenza di correlazione** tra attributi diversi

Fattore di selettività (predicati atomici)

Predicato p	Stima di f_p
$A_i = v$	$\frac{1}{VAL(A_i, r)}$
$A_i \leq v$	$\frac{v - MIN(A_i, r)}{MAX(A_i, r) - MIN(A_i, r)}$
$A_i \geq v$	$\frac{MAX(A_i, r) - v}{MAX(A_i, r) - MIN(A_i, r)}$
$v_1 \leq A_i \leq v_2$	$\frac{v_2 - v_1}{MAX(A_i, r) - MIN(A_i, r)}$

- Con l'assunzione di uniformità su tutti i valori, abbiamo che $VAL(A_i, r) = MAX(A_i, r) - MIN(A_i, r) + 1$ (perché non abbiamo «buchi») e quindi le prime tre righe risultano essere un caso particolare dell'ultima riga. Tuttavia, quando ci sono buchi, le stime fornite dalle formule nella tabella sono più precise.
- Poiché stiamo ricavando una stima, trascuriamo alcuni termini (ad es. l'ultima riga più precisamente sarebbe $f_p = (v_2 - v_1 + 1) / (MAX(A_i, r) - MIN(A_i, r) + 1)$).
- Quando v , v_1 e v_2 non sono compresi tra $MIN(A_i, r)$ e $MAX(A_i, r)$, le formule devono essere aggiustate (ad es. nella prima riga, invece di $f_p = 1/VAL(A_i, r)$, bisogna avere $f_p = 0$).

Fattore di selettività (predicati composti)

Predicato p	Stima di f_p
$p_1 \wedge p_2 \wedge \dots \wedge p_n$	$f_{p_1} \cdot f_{p_2} \cdot \dots \cdot f_{p_n}$
$\neg p$	$1 - f_p$
$p_1 \vee p_2 \vee \dots \vee p_n$	$1 - ((1 - f_{p_1}) \cdot (1 - f_{p_2}) \cdot \dots \cdot (1 - f_{p_n}))$

L'OR si dimostra da AND e NOT con De Morgan:

$$\neg\neg(p_1 \vee p_2 \vee \dots \vee p_n) = \neg(\neg p_1 \wedge \neg p_2 \wedge \dots \wedge \neg p_n)$$

Stima della cardinalità del join

Per i nostri scopi, ci limiteremo a studiare l'equi-join

$$|r(A) \bowtie_{A_i=B_j} s(B)|$$

- Iniziamo notando che per il prodotto cartesiano abbiamo

$$|r(A) \times s(B)| = \text{CARD}(r) \cdot \text{CARD}(s)$$

- Se ci accontentiamo di una stima grezza possiamo dire che

$$0 \leq |r(A) \bowtie_{A_i=B_j} s(B)| \leq \text{CARD}(r) \cdot \text{CARD}(s)$$

- Ma possiamo ricavare una stima più utile con il fattore di selettività...

Stima della cardinalità del join

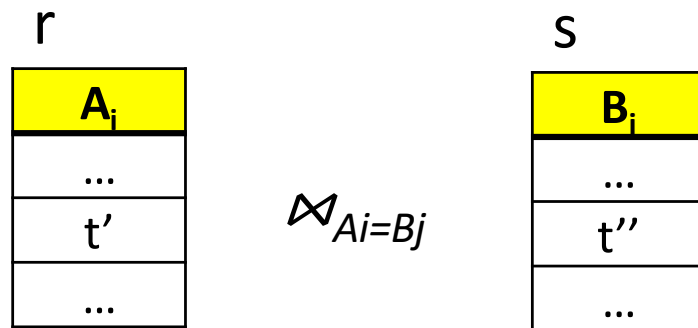
Per stimare $|r(A) \bowtie_{A_i=B_j} s(B)|$, consideriamo prima **una singola tupla $t' \in r$** per cui $t'[A_i]=v$ (v è una costante):

- la probabilità che **una singola tupla $t'' \in s$** sia tale che $t''[B_j]=v$ è $1/VAL(B_j, s)$
- considerando **tutta la relazione s** , ci saranno $(1/VAL(B_j, s)) \cdot CARD(s)$ tuple t'' per cui $t''[B_j]=v$.

Se esistesse un vincolo di integrità referenziale « $r(A_i)$ *referenzia* $s(B_j)$ », dato che B_j sarebbe chiave di s , B_j non avrebbe valori ripetuti e $VAL(B_j, s) = CARD(s)$, cioè $1/VAL(B_j, s) \cdot CARD(s) = 1$, cioè avremmo una e una sola tupla t'' per ogni tupla t' .

Se non esiste il vincolo di integrità referenziale, sappiamo che la stima precedente è pessimistica perché potrebbe non esserci nessuna tupla in s tale che $t''[B_j]=v$.

Stima della cardinalità del join



Stima della cardinalità del join

Consideriamo ora **tutte le tuple** $t' \in r$:

La stima pessimistica della cardinalità del join diventa quindi:

$$|r(A) \bowtie_{A_i=B_j} s(B)| \leq (1/VAL(B_j, s)) \cdot CARD(s) \cdot CARD(r)$$

Non ho ancora finito!

Stima della cardinalità del join

Dato che vale la proprietà commutativa del join, posso ripetere il ragionamento che ha portato a

$$|r(A) \bowtie_{A_i=B_j} s(B)| \leq (1/VAL(B_j, s)) \cdot CARD(s) \cdot CARD(r)$$

partendo da s invece che da r e ottengo:

$$|s(B) \bowtie_{B_j=A_i} r(A)| \leq (1/VAL(A_i, r)) \cdot CARD(r) \cdot CARD(s)$$

Rispetto al caso precedente cambia solo il fattore di selettività:

$$1/VAL(A_i, r) \quad \text{invece di} \quad 1/VAL(B_j, s)$$

Stima della cardinalità del join

Entrambe le formule sono delle stime pessimistiche, cioè delle sovrastime. Se consideriamo il minimo delle due, abbiamo una sovrastima migliore.

La cardinalità del join diventa:

$$|r(A) \bowtie_{A_i=B_j} s(B)| \leq \min\{1/\text{VAL}(A_i, r) \cdot \text{CARD}(r) \cdot \text{CARD}(s), \\ 1/\text{VAL}(B_j, s) \cdot \text{CARD}(s) \cdot \text{CARD}(r)\}$$

Cioè

$$|r(A) \bowtie_{A_i=B_j} s(B)| \leq \min\{1/\text{VAL}(A_i, r), 1/\text{VAL}(B_j, s)\} \cdot \text{CARD}(r) \cdot \text{CARD}(s)$$

Caso di studio

$$\pi_L(\sigma_{PAZIENTI.Residenza='TO' \wedge Inizio=2010 \wedge MATR='405'}($$

(pazienti $\bowtie_{COD=PAZ}$ *ricoveri)* $\bowtie_{RICOVERI.Reparto=MEDICI.Reparto$
medici}))

Con

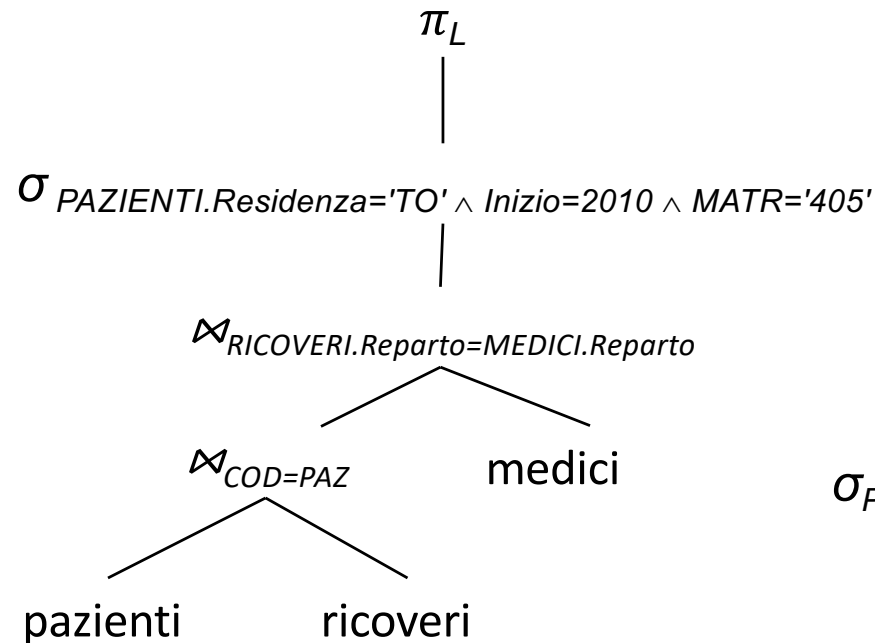
- *p1: PAZIENTI.Residenza='TO'*
- *p2: Inizio=2010*
- *p3: MATR='405'*

Caso di studio

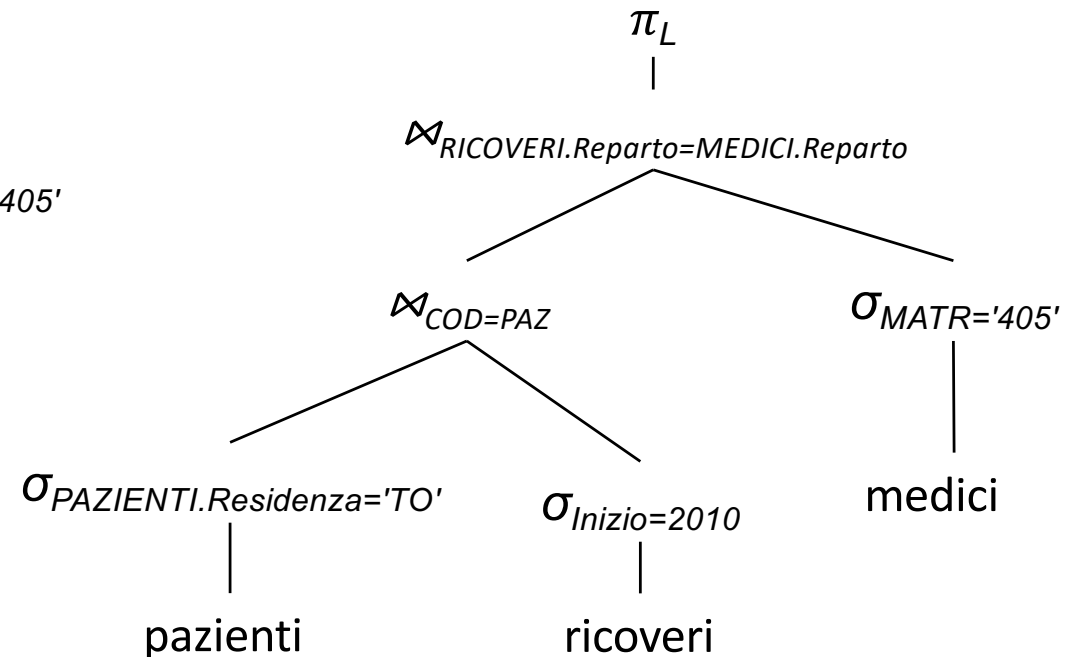
$\pi_L(\sigma_{PAZIENTI.Residenza='TO' \wedge Inizio=2010 \wedge MATR='405'}((pazienti \bowtie_{COD=PAZ} ricoveri) \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} medici))$

Confrontiamo i costi dell'interrogazione prima e dopo i passi 1-6 dell'algoritmo di ottimizzazione logica

Prima



Dopo



Caso di studio

Dizionario dati:

- $\text{CARD}(\text{pazienti}) = 10^5$
- $\text{CARD}(\text{ricoveri}) = 10^5$
- $\text{CARD}(\text{medici}) = 100$
- $\text{CARD}(\text{reparti}) = 10$
- $\text{VAL}(\text{Inizio}, \text{ricoveri}) = 10$ (supponiamo di memorizzare solo gli anni (dieci))
- $\text{VAL}(\text{reparto}, \text{ricoveri}) = 10$
- $\text{VAL}(\text{residenza}, \text{pazienti}) = 100$
- $\text{VAL}(\text{PAZ}, \text{ricoveri}) = 10^5$
- $\text{VAL}(\text{reparto}, \text{medici}) = 10$

Caso di studio

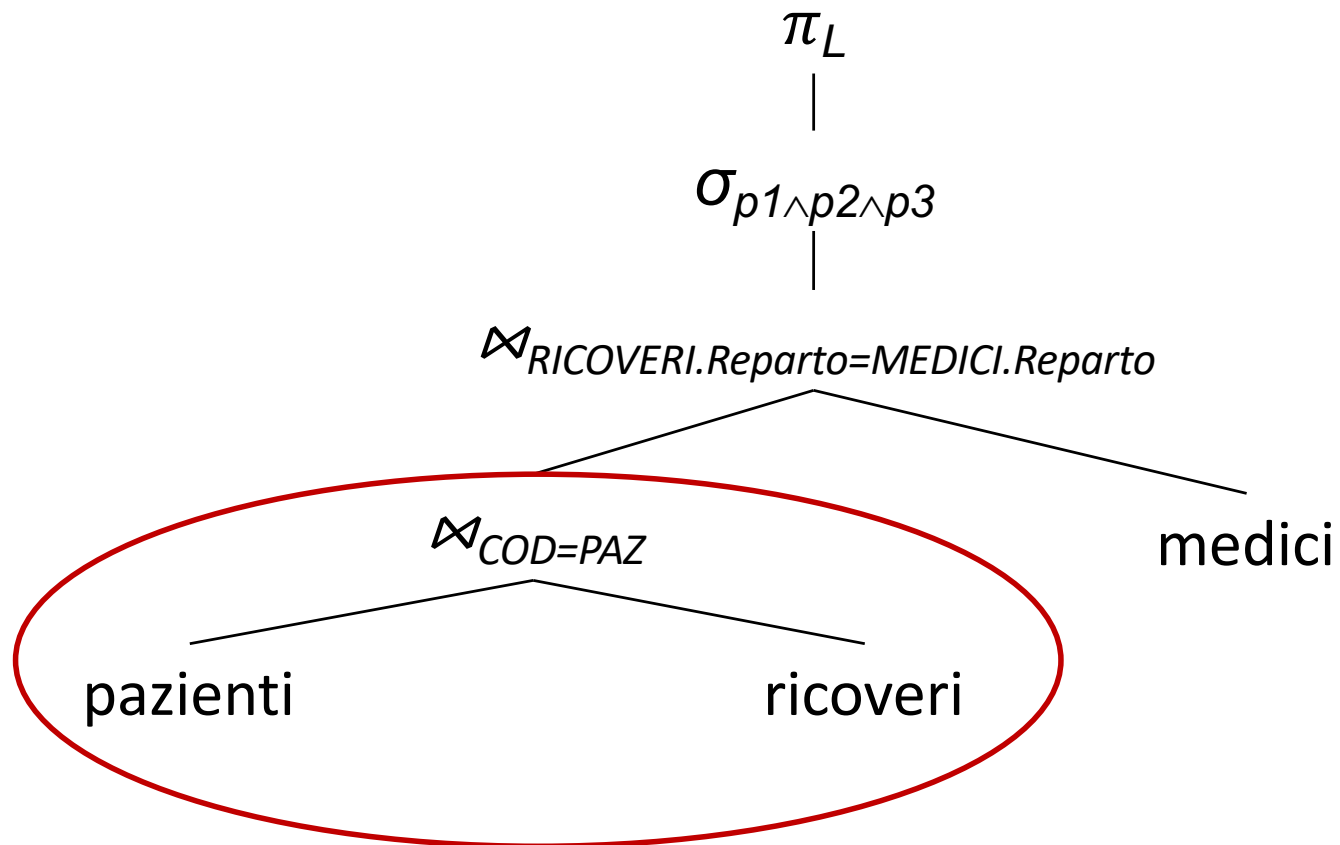
Posso già calcolare i fattori di selettività dei predicati

- $p1: PAZIENTI.Residenza='TO'$
- $p2: Inizio=2010$
- $p3: MATR='405'$

cioè

- $f_{p1} = 1/VAL(residenza, pazienti) = 1/100$
- $f_{p2} = 1/VAL(Inizio, ricoveri) = 1/10$
- $f_{p3} = 1/VAL(MATR, medici) = \mathbf{1/CARD(medici)} = 1/100$

Caso di studio (prima)



Caso di studio (prima)

Partendo dalle foglie e andando verso la radice, per ogni operatore, calcolo in base a quante tuple riceve in input:

- 1) quante tuple «muove» (cioè quante tuple elabora) e
- 2) quante tuple produce in output (che verrà usato come input per l'operatore successivo)

Caso di studio (prima)

Valutiamo il join

$$r1 := \text{pazienti} \bowtie_{\text{COD}=\text{PAZ}} \text{ricoveri}$$

Muove $|\text{pazienti}| \cdot |\text{ricoveri}|$ tuple

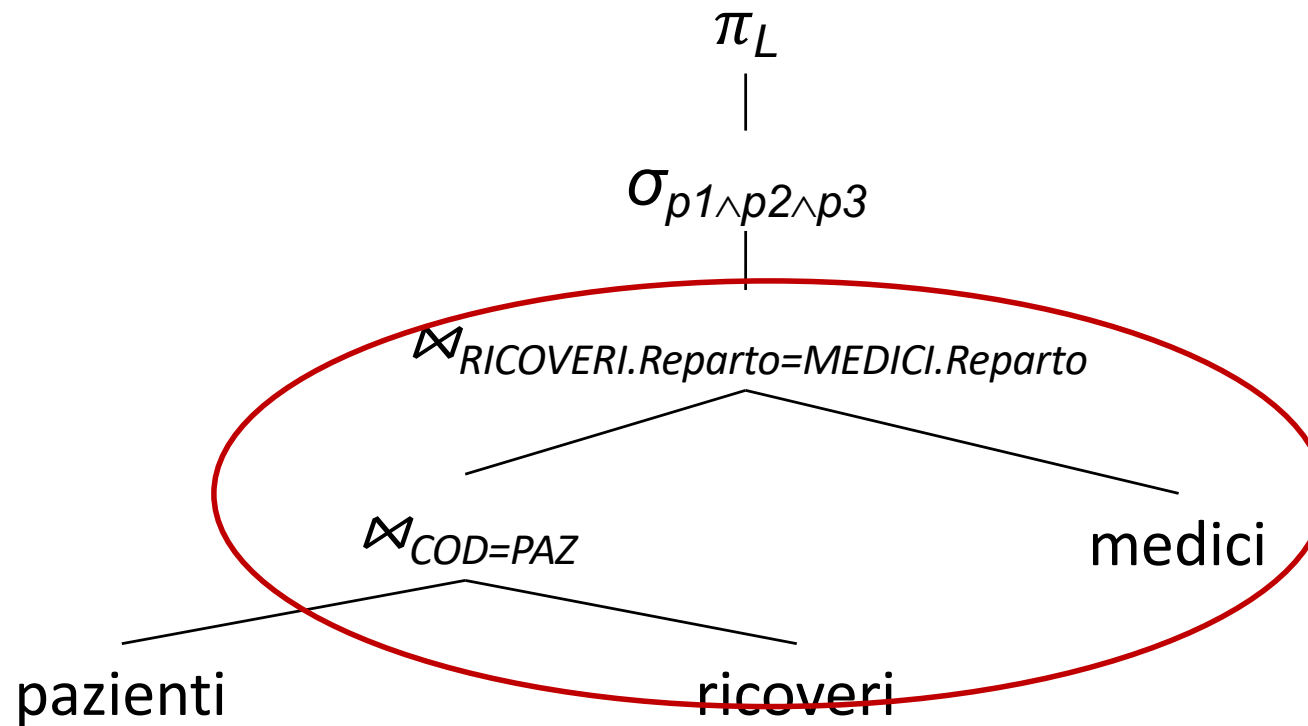
Per la cardinalità dell'output applichiamo la stima della cardinalità del join:

$$\begin{aligned} |\text{pazienti} \bowtie_{\text{COD}=\text{PAZ}} \text{ricoveri}| = \\ \min\{1/\text{VAL}(\text{COD}, \text{pazienti}) \cdot \text{CARD}(\text{pazienti}) \cdot \text{CARD}(\text{ricoveri}), \\ 1/\text{VAL}(\text{PAZ}, \text{ricoveri}) \cdot \text{CARD}(\text{pazienti}) \cdot \text{CARD}(\text{ricoveri})\} \end{aligned}$$

COD è chiave di *pazienti* quindi $\text{VAL}(\text{COD}, \text{pazienti}) = \text{CARD}(\text{pazienti})$ e

$$\begin{aligned} |\text{pazienti} \bowtie_{\text{COD}=\text{PAZ}} \text{ricoveri}| = \\ \min\{\text{CARD}(\text{ricoveri}), 1/\text{VAL}(\text{PAZ}, \text{ricoveri}) \cdot \text{CARD}(\text{pazienti}) \cdot \text{CARD}(\text{ricoveri})\} = \\ \min\{10^5, 1/10^5 \cdot 10^5 \cdot 10^5\} = 10^5 \end{aligned}$$

Caso di studio (prima)



Caso di studio (prima)

Se $r1$ è il risultato del join precedente, dobbiamo valutare il join

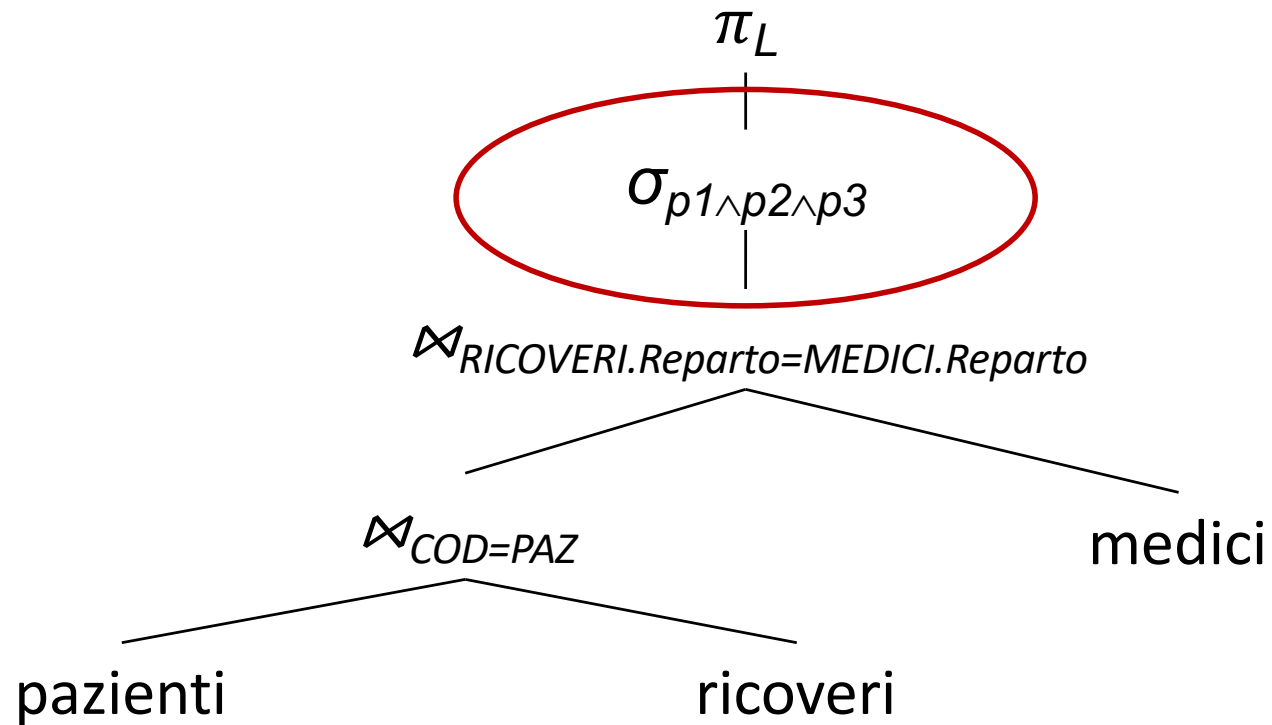
$$r2 := r1 \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} medici$$

$r2$ muove $|r1| \cdot |medici|$ tuple

e produce, applicando la stima della cardinalità del join:

$$\begin{aligned} |r1 \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} medici| &= \\ \min\{1/VAL(Reparto, r1), 1/VAL(Reparto, medici)\} \cdot CARD(r1) \cdot \\ & \quad CARD(medici) \\ &= \min(1/10, 1/10) \cdot 10^5 \cdot 10^2 \\ &= 1/10 \cdot 10^5 \cdot 10^2 = 10^6 \end{aligned}$$

Caso di studio (prima)



Caso di studio (prima)

A questo punto abbiamo la selezione $\sigma_{p1 \wedge p2 \wedge p3}$ che elabora le 10^6 tuple del risultato precedente (r2).

Dati i fattori di selettività $f_{p1} = 1/100$, $f_{p2} = 1/10$, $f_{p3} = 1/100$, produce:

$$\begin{aligned} r3 &:= |\sigma_{p1 \wedge p2 \wedge p3}(r2)| = \\ f_{p1} \cdot f_{p2} \cdot f_{p3} \cdot \text{CARD}(r2) &= 1/100 \cdot 1/10 \cdot 1/100 \cdot 10^6 \\ &= 10 \end{aligned}$$

Caso di studio (prima)

Qual è la massa di tuple coinvolte dall'interrogazione prima dell'ottimizzazione (ignorando per semplicità la proiezione)?

Consideriamo le tuple coinvolte da ogni operazione:

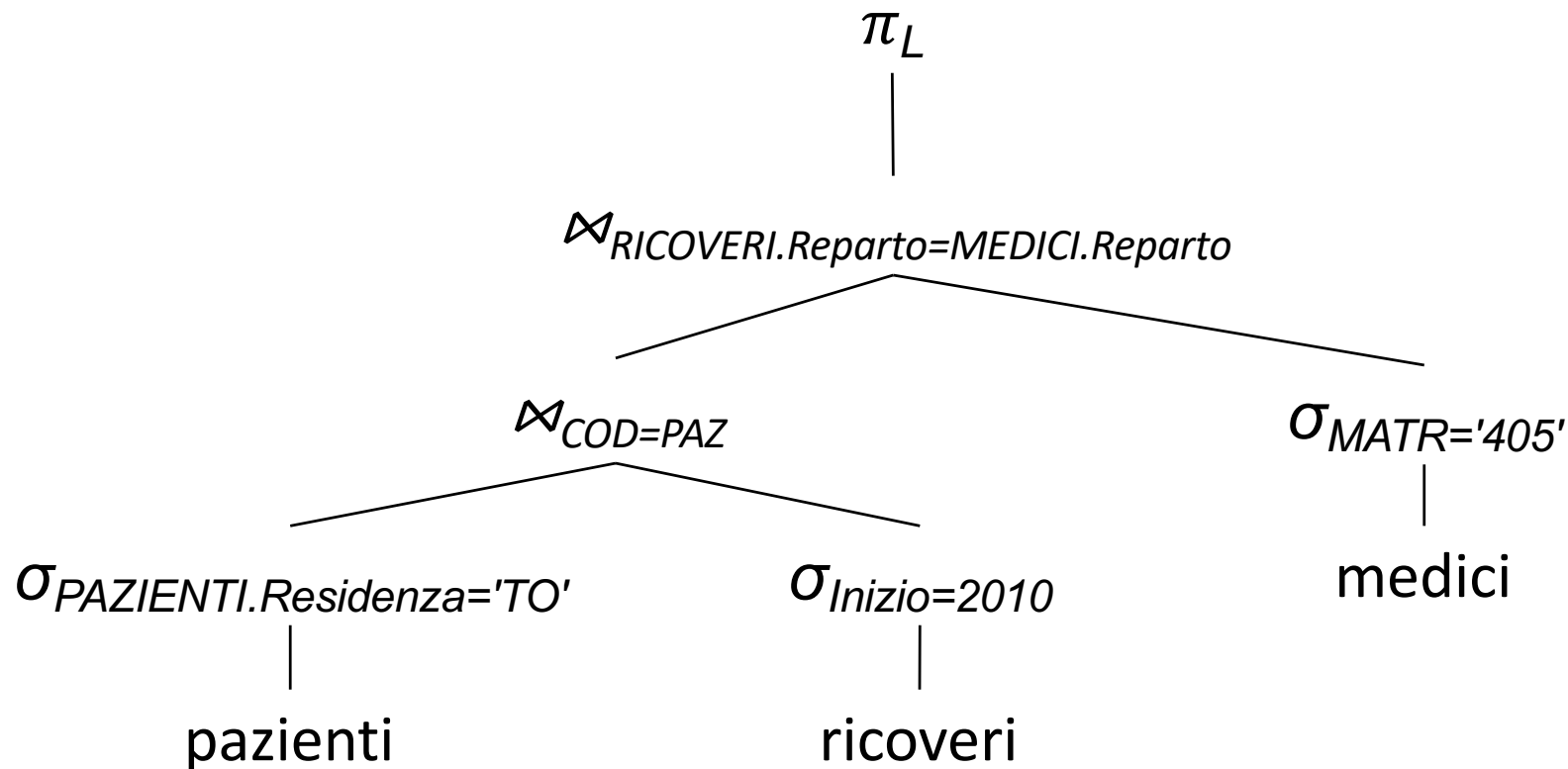
- $r1 := \text{pazienti} \bowtie_{\text{COD}=\text{PAZ}} \text{ricoveri}$
combina 10^5 tuple di *medici* con 10^5 tuple di *ricoveri* per restituire 10^5 tuple
- $r2 := r1 \bowtie_{\text{RICOVERI.Reparto}=\text{MEDICI.Reparto}} \text{medici}$
combina 10^5 tuple di *r1* con 10^2 tuple di *medici* per restituire 10^6 tuple
- $r3 := \sigma_{p1 \wedge p2 \wedge p3}(r2)$ valuta 10^6 tuple di *r2* per restituire 10 tuple

Quindi sono coinvolte $10^5 \cdot 10^5 + 10^5 \cdot 10^2 + 10^6$ tuple

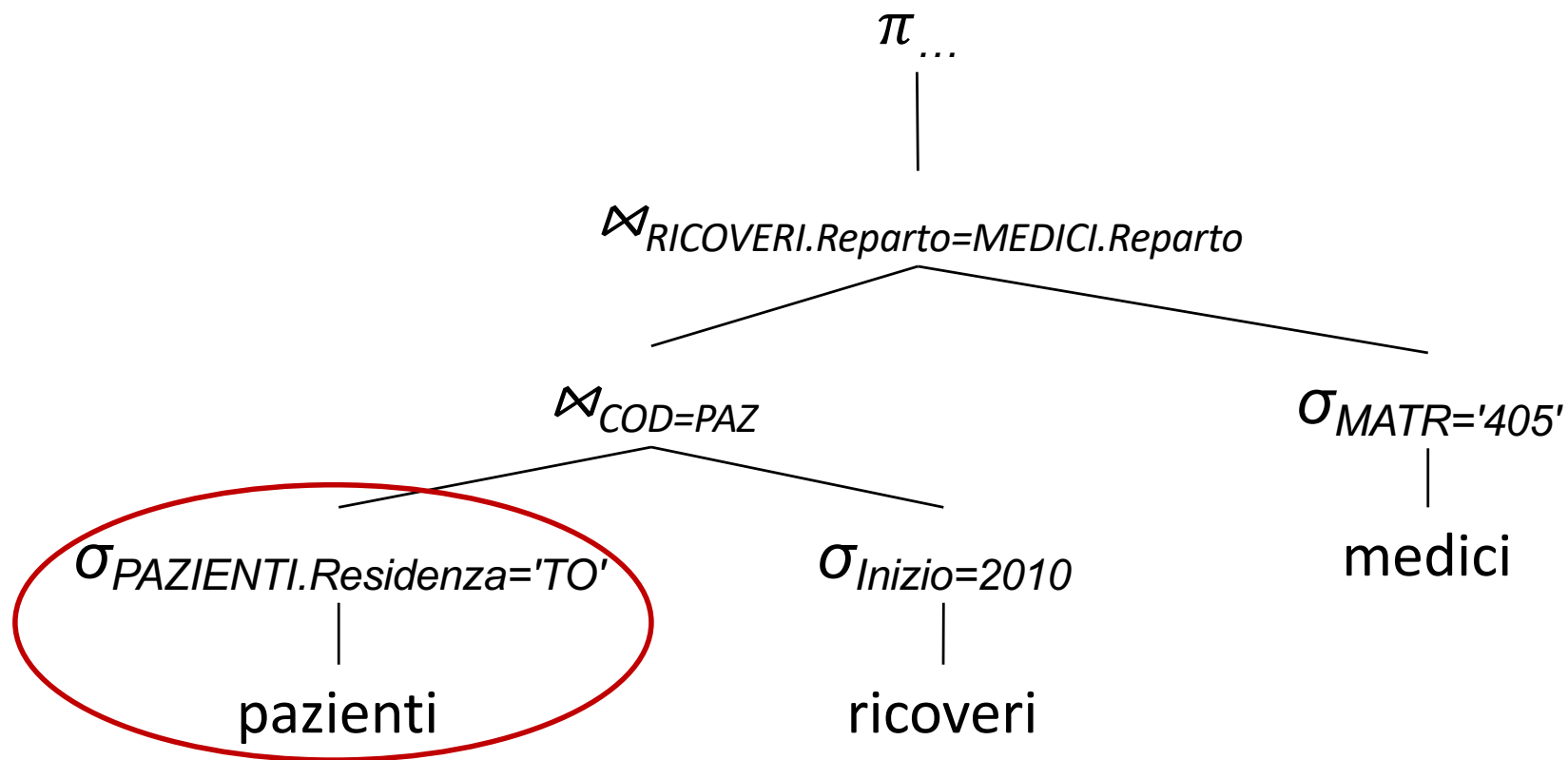
Considerando il valore dominante, $10^5 \cdot 10^5 = 10^{10}$, «muoviamo» 10^{10} tuple per avere approssimativamente 10 tuple

Caso di studio (dopo)

Consideriamo ora il costo dopo i passi 1-6 dell'algoritmo di ottimizzazione logica



Caso di studio (dopo)



Caso di studio (dopo)

Quante tuple «muove» la selezione

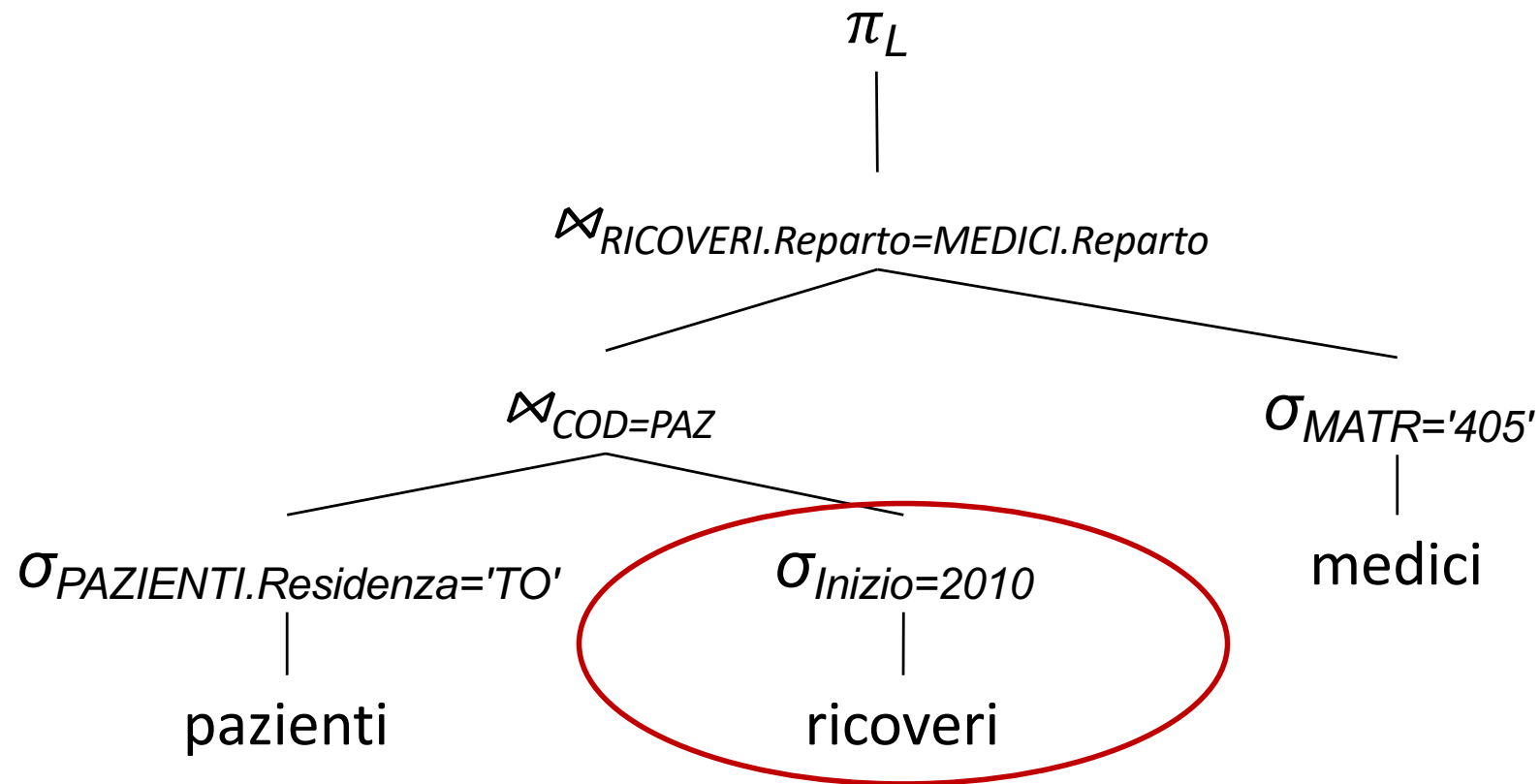
$\sigma_{PAZIENTI.Residenza='TO'}(pazienti)$?

Sono esattamente le tuple della relazione *pazienti*,
ovvero 10^5

Quante tuple produce?

$$|\sigma_{PAZIENTI.Residenza='TO'}(pazienti)| = f_{p1} \cdot CARD(pazienti) = \\ 1/100 \cdot 10^5 = 10^3$$

Caso di studio (dopo)



Caso di studio (dopo)

Quante sono le tuple coinvolte dalla selezione

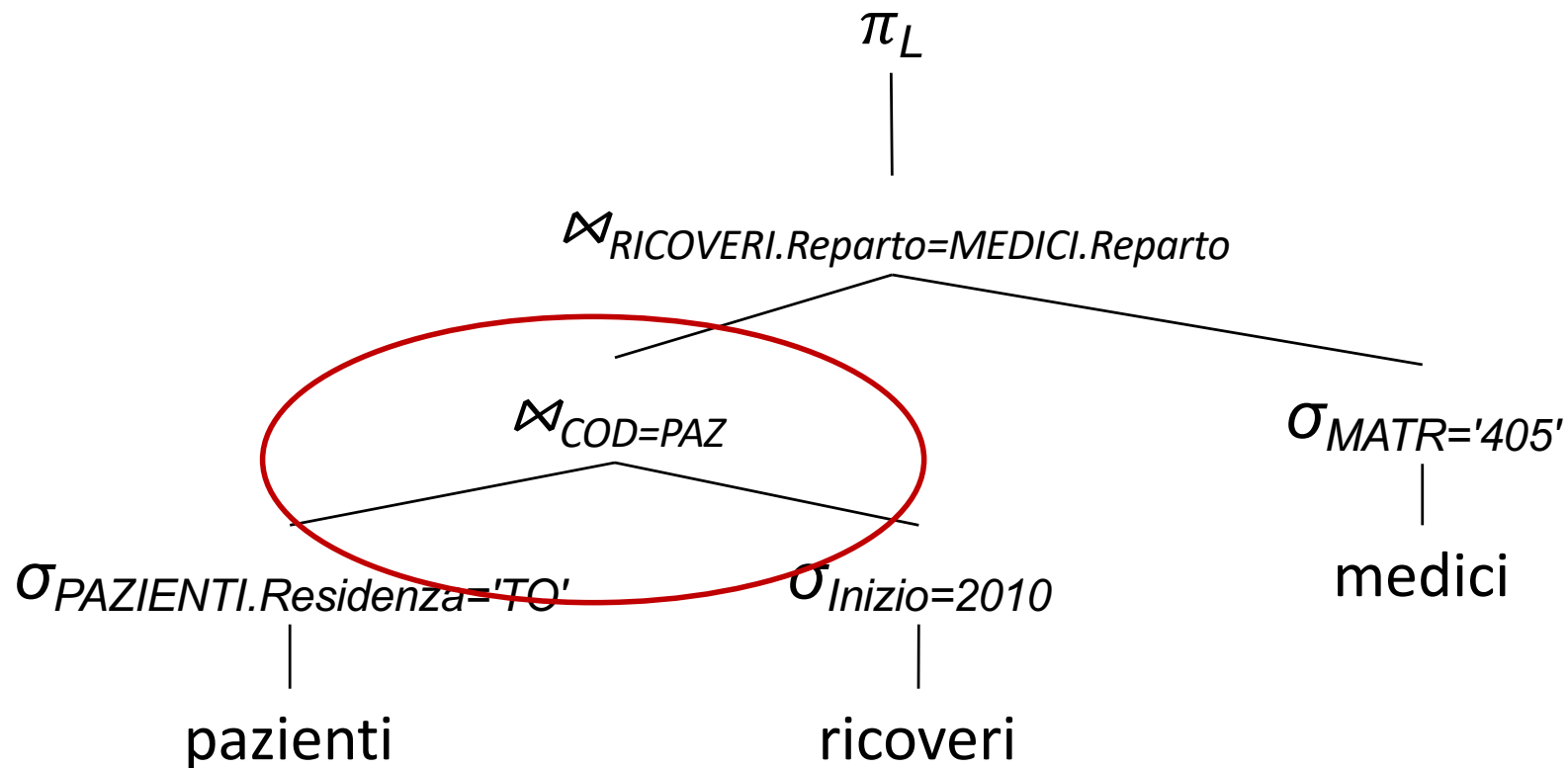
$\sigma_{Inizio=2010}(ricoveri)$?

Sono esattamente le tuple della relazione *ricoveri*,
ovvero 10^5 .

Quante tuple produce?

$$|\sigma_{Inizio=2010}(ricoveri)| = f_{p2} \cdot CARD(ricoveri) = 1/10 \cdot 10^5 = 10^4$$

Caso di studio (dopo)



Caso di studio (dopo)

Quante sono le tuple coinvolte dal join

$$r1' := \sigma_{p1}(\text{pazienti}) \bowtie_{\text{COD}=\text{PAZ}} \sigma_{p2}(\text{ricoveri}) ?$$

Questa volta il join lavora su $10^3 \cdot 10^4 = 10^7$ tuple.

Quante tuple produce?

Caso di studio (dopo)

Quante tuple produce $r1' := \sigma_{p1}(\text{pazienti}) \bowtie_{\text{COD}=\text{PAZ}} \sigma_{p2}(\text{ricoveri})$?

Stimo la cardinalità dell'equi-join

$$\begin{aligned} & |\sigma_{p1}(\text{pazienti}) \bowtie_{\text{COD}=\text{PAZ}} \sigma_{p2}(\text{ricoveri})| = \\ & \min\{1/\text{VAL}(\text{COD}, \sigma_{p1}(\text{pazienti})), 1/\text{VAL}(\text{PAZ}, \sigma_{p2}(\text{ricoveri}))\} \cdot \\ & \text{CARD}(\sigma_{p1}(\text{pazienti})) \cdot \text{CARD}(\sigma_{p2}(\text{ricoveri})) \end{aligned}$$

Non conosco però $\text{VAL}(\text{COD}, \sigma_{p1}(\text{pazienti}))$ e $\text{VAL}(\text{PAZ}, \sigma_{p2}(\text{ricoveri}))$.

Euristica del DBMS per la stima

Posso stimare $VAL(COD, \sigma_{p1}(pazienti))$ partendo dai dati che il DBMS conosce.

Il numero da stimare non può essere maggiore di:

- $VAL(COD, pazienti) = 10^5$
(numero di tuple in pazienti con valori distinti per COD)
- $CARD(\sigma_{p1}(pazienti)) = f_{p1} \cdot CARD(pazienti) = 1/100 \cdot 10^5 = 10^3$
(numero di tuple nel risultato della selezione σ_{p1})

Quindi:

$$\begin{aligned} VAL(COD, \sigma_{p1}(pazienti)) &= \\ \min\{VAL(COD, pazienti), CARD(\sigma_{p1}(pazienti))\} &= \\ \min\{10^5, 10^3\} &= 10^3 \end{aligned}$$

Euristica del DBMS per la stima

Formula generale:

$$\begin{aligned} |\sigma_{p1}(r(A)) \bowtie_{A_i=B_j} \sigma_{p2}(s(B))| = \\ \min\{1/VAL(A_i, \sigma_{p1}(r)), 1/VAL(B_j, \sigma_{p2}(s))\} \cdot \\ CARD(\sigma_{p1}(r)) \cdot CARD(\sigma_{p2}(s)) \end{aligned}$$

con

- $VAL(A_i, \sigma_{p1}(r)) = \min\{VAL(A_i, r), CARD(\sigma_{p1}(r))\}$
- $VAL(B_j, \sigma_{p2}(s)) = \min\{VAL(B_j, s), CARD(\sigma_{p2}(s))\}$

quando A_i e B_j non sono coinvolti nei predicati di selezione

Caso di studio (dopo)

Devo ancora ricavare $VAL(PAZ, \sigma_{p_2}(ricoveri))$.

Il DBMS conosce questi dati:

- $VAL(PAZ, ricoveri) = 10^5$
- $CARD(\sigma_{p_2}(ricoveri)) = f_{p_2} \cdot CARD(ricoveri) = 1/10 \cdot 10^5 = 10^4$

Di conseguenza

$$\begin{aligned} VAL(PAZ, \sigma_{p_2}(ricoveri)) &= \\ \min\{VAL(PAZ, ricoveri), CARD(\sigma_{p_2}(ricoveri))\} &= \\ \min\{10^5, 10^4\} &= 10^4 \end{aligned}$$

Caso di studio (dopo)

Quante tuple produce $r1' := \sigma_{p1}(\text{pazienti}) \bowtie_{\text{COD}=\text{PAZ}} \sigma_{p2}(\text{ricoveri})$?

$$|\sigma_{p1}(\text{pazienti}) \bowtie_{\text{COD}=\text{PAZ}} \sigma_{p2}(\text{ricoveri})|$$

=

$$\min\{1/\text{VAL}(\text{COD}, \sigma_{p1}(\text{pazienti})), \\ 1/\text{VAL}(\text{PAZ}, \sigma_{p2}(\text{ricoveri}))\} \cdot$$

$$\text{CARD}(\sigma_{p1}(\text{pazienti})) \cdot \text{CARD}(\sigma_{p2}(\text{ricoveri}))$$

=

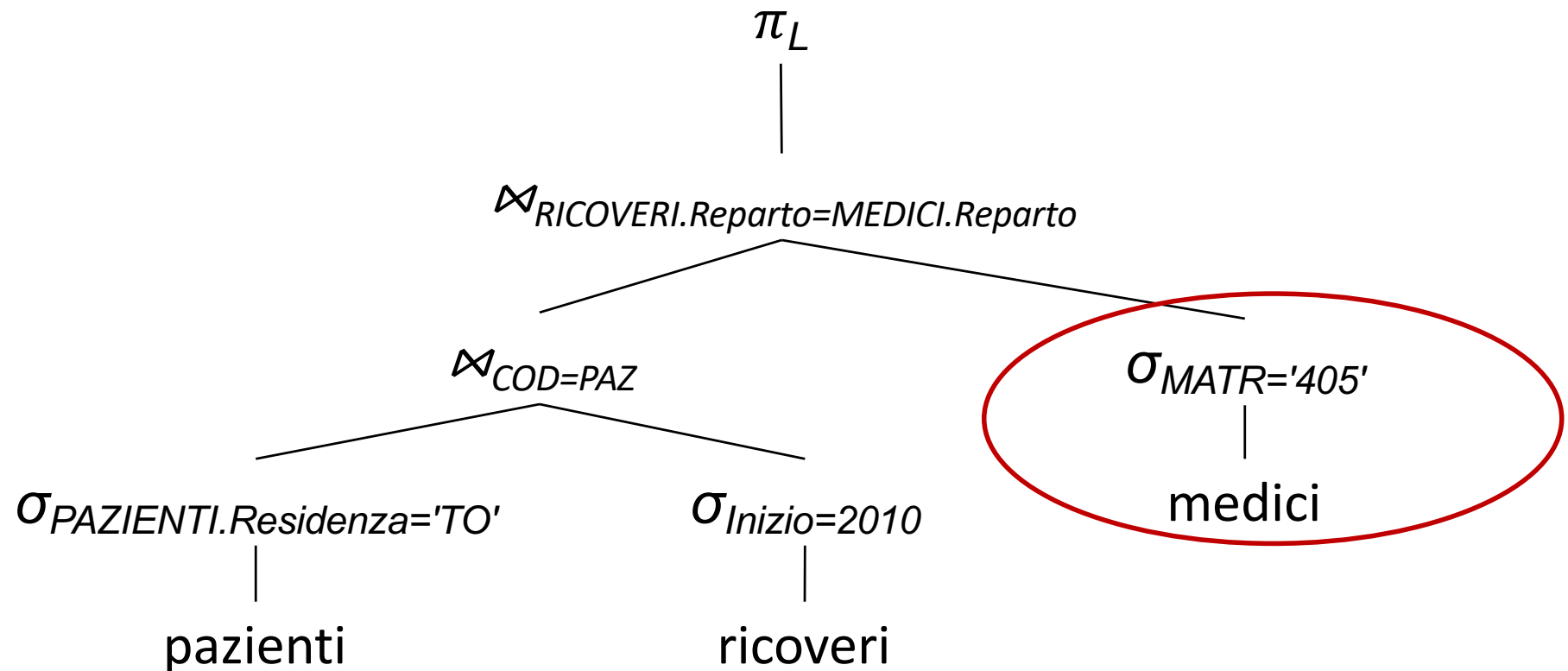
$$\min\{1/\min\{\text{VAL}(\text{COD}, \text{pazienti}), \text{CARD}(\sigma_{p1}(\text{pazienti}))\}, \\ 1/\min\{\text{VAL}(\text{PAZ}, \text{ricoveri}), \text{CARD}(\sigma_{p2}(\text{ricoveri}))\}\} \cdot$$

$$\text{CARD}(\sigma_{p1}(\text{pazienti})) \cdot \text{CARD}(\sigma_{p2}(\text{ricoveri}))$$

=

$$\min\{1/10^3, 1/10^4\} \cdot 10^3 \cdot 10^4 = 1/10^4 \cdot 10^3 \cdot 10^4 = \mathbf{10^3}$$

Caso di studio (dopo)



Caso di studio (dopo)

Quante sono le tuple coinvolte da

$r2' := \sigma_{MATR='405'}(medici)?$

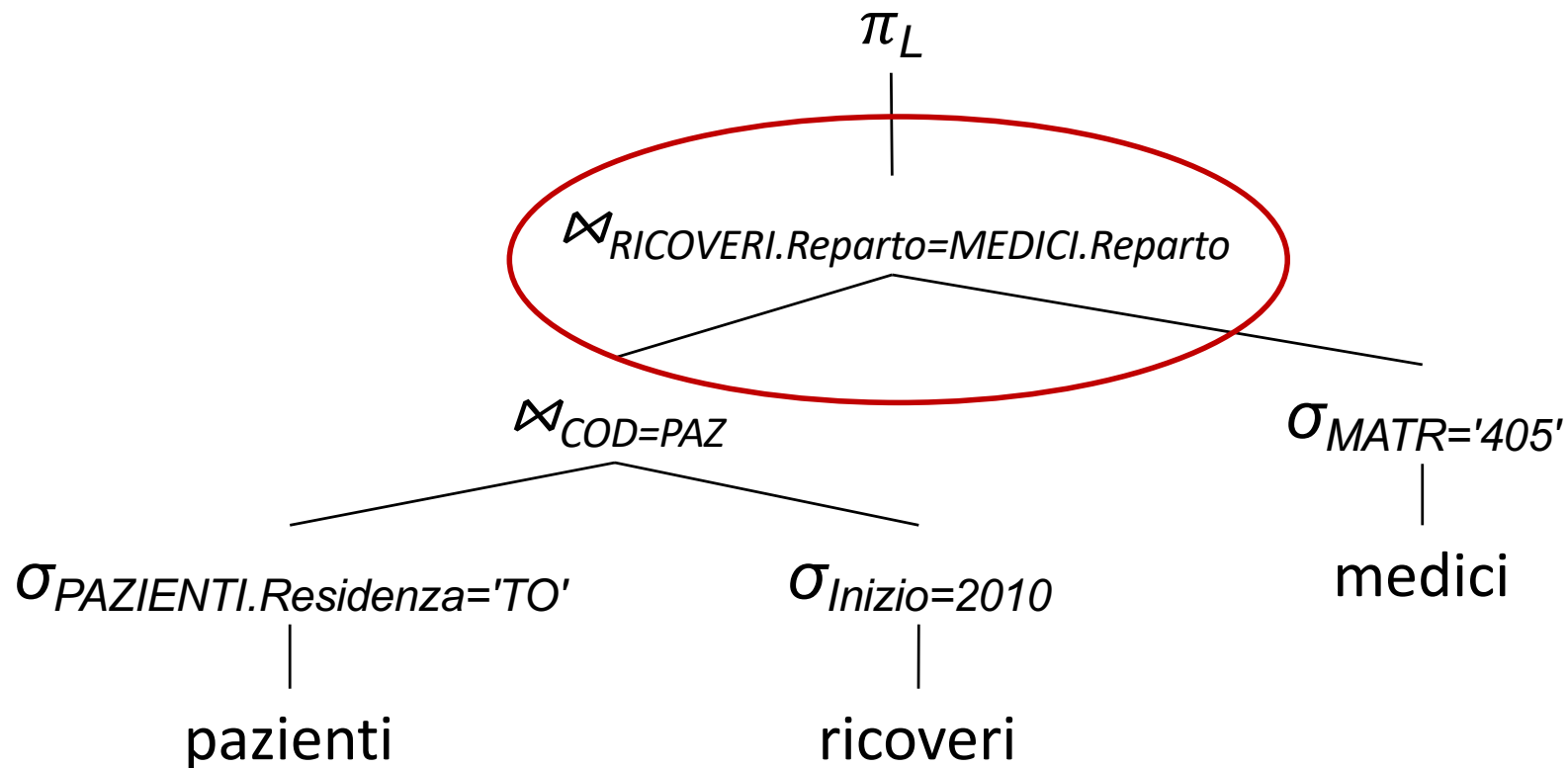
Sono esattamente le tuple della relazione *medici*,
ovvero 10^2 .

Quante tuple produce?

$$|\sigma_{MATR='405'}(medici)| = 1$$

(è una selezione su MATR che è chiave, quindi produce
una sola tupla)

Caso di studio (dopo)



Caso di studio (dopo)

Quante sono le tuple coinvolte da

$$r3' := r1' \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} r2'?$$

Questa volta il join lavora su $10^3 \cdot 1 = 10^3$ tuple

Quante tuple produce?

Non è utile calcolarlo perché, dato che questo albero sintattico è equivalente al precedente, dovrà produrre lo stesso numero di tuple.

Caso di studio (dopo)

Qual è la massa di tuple coinvolte dall'interrogazione dopo i passi 1-6 dell'ottimizzazione (ignorando per semplicità la proiezione)?

- $\sigma_{p1}(\text{pazienti})$
considera 10^5 tuple per produrre 10^3 tuple
- $\sigma_{p2}(\text{ricoveri})$
considera 10^5 tuple per produrre 10^4 tuple
- $r1' := \sigma_{p1}(\text{pazienti}) \bowtie_{\text{COD}=\text{PAZ}} \sigma_{p2}(\text{ricoveri})$
combina 10^3 tuple di $\sigma_{p1}(\text{pazienti})$ con 10^4 tuple di $\sigma_{p2}(\text{ricoveri})$
per restituire 10^3 tuple
- $r2' := \sigma_{\text{MATR}='405'}(\text{medici})$
valuta 10^2 tuple di medici per restituire 1 tupla
- $r3' := r1' \bowtie_{\text{RICOVERI.Reparto}=\text{MEDICI.Reparto}} r2'$
combina 10^3 tuple di $r1'$ con 1 tupla di $r2'$ per restituire 10 tuple

Quindi sono coinvolte $10^5 + 10^5 + 10^3 \cdot 10^4 + 10^2 + 10^3 \cdot 1$ tuple

Considerando il valore dominante, $10^3 \cdot 10^4 = 10^7$, «muoviamo» **mille volte meno** della massa di tuple coinvolte senza ottimizzazione logica!

Spiegazione del passo 7

Esaminare le varianti dell'albero sintattico dovute alle proprietà associative scegliendo la variante di costo minimo

- L'ottimizzatore cerca una configurazione dell'albero sintattico che riduca ulteriormente la massa di tuple concettualmente elaborate
- Principalmente questo passo consiste nella ricerca di parentesizzazioni alternative – e meno costose – di join multipli

Spiegazione del passo 7

Questo passo ha delle difficoltà computazionali non indifferenti
Consideriamo una sequenza di join:

$$r_1 \bowtie r_2 \bowtie \dots \bowtie r_n$$

- Le configurazioni possibili dovute alle proprietà associative crescono esponenzialmente con n
 - $n=3$, 2 configurazioni
 - $n=4$, 5 configurazioni
 - $n=5$, 14 configurazioni
 - $n=10$, 4862 configurazioni

$$\text{configurazioni} = \frac{1}{n} \binom{2(n-1)}{n-1}$$

Esempio con 4 relazioni

$$((r_1 \bowtie r_2) \bowtie r_3) \bowtie r_4$$

$$(r_1 \bowtie (r_2 \bowtie r_3)) \bowtie r_4$$

$$r_1 \bowtie ((r_2 \bowtie r_3) \bowtie r_4)$$

$$r_1 \bowtie (r_2 \bowtie (r_3 \bowtie r_4))$$

$$(r_1 \bowtie r_2) \bowtie (r_3 \bowtie r_4)$$

Spiegazione del passo 7

- Gli ottimizzatori non esplorano tutte le possibili configurazioni e si accontentano di una configurazione eventualmente subottimale.
- L'euristica applicata dell'ottimizzatore è quella di anticipare il prima possibile i join tra relazioni con cardinalità bassa.
- Si cerca il join a cardinalità minima dei sottoalberi e si esegue per primo e così via.
- Anche se il join lavora su una massa di tuple enorme, solitamente, produce un numero di tuple limitato grazie alla selettività del suo predicato θ .

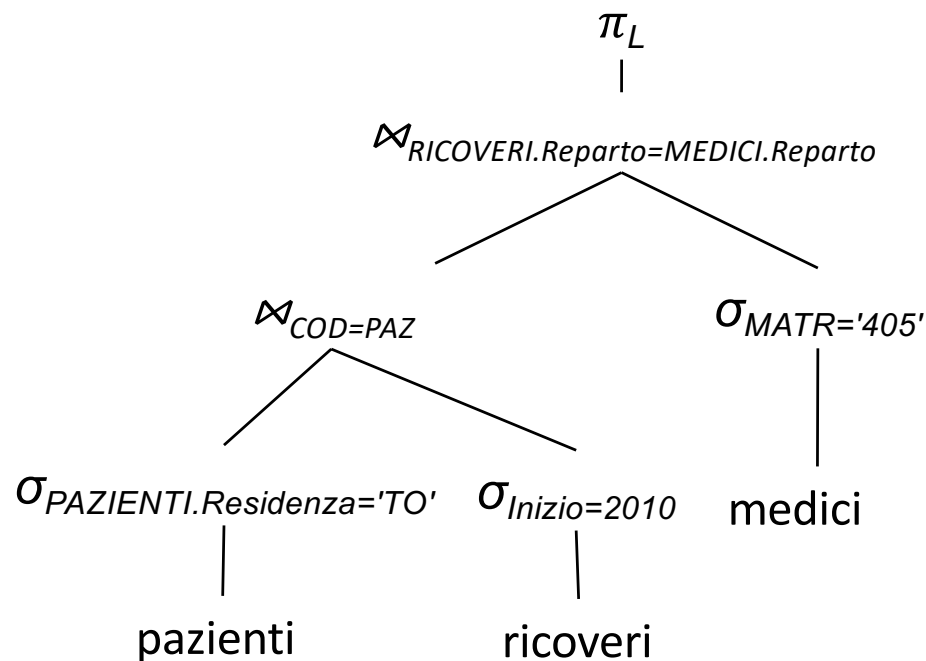
Caso di studio

- I join della query operano su tre selezioni:
 - $\sigma_{p1}(\text{pazienti})$ elabora 10^5 tuple e $|\sigma_{p1}| = 10^3$
 - $\sigma_{p2}(\text{ricoveri})$ elabora 10^5 tuple e $|\sigma_{p2}| = 10^4$
 - $\sigma_{p3}(\text{medici})$ elabora 10^2 tuple e $|\sigma_{p3}| = 1$
- Quindi l'ottimizzatore cercherà di eseguire prima un join che coinvolge $\sigma_{p3}(\text{medici})$.
- Tra gli altri due sottoalberi, quello con la cardinalità minore (10^3) è $\sigma_{p1}(\text{pazienti})$.
- Ma *medici* non può entrare in join direttamente con *pazienti*, quindi l'ottimizzatore è costretto a scegliere il join con $\sigma_{p2}(\text{ricoveri})$.

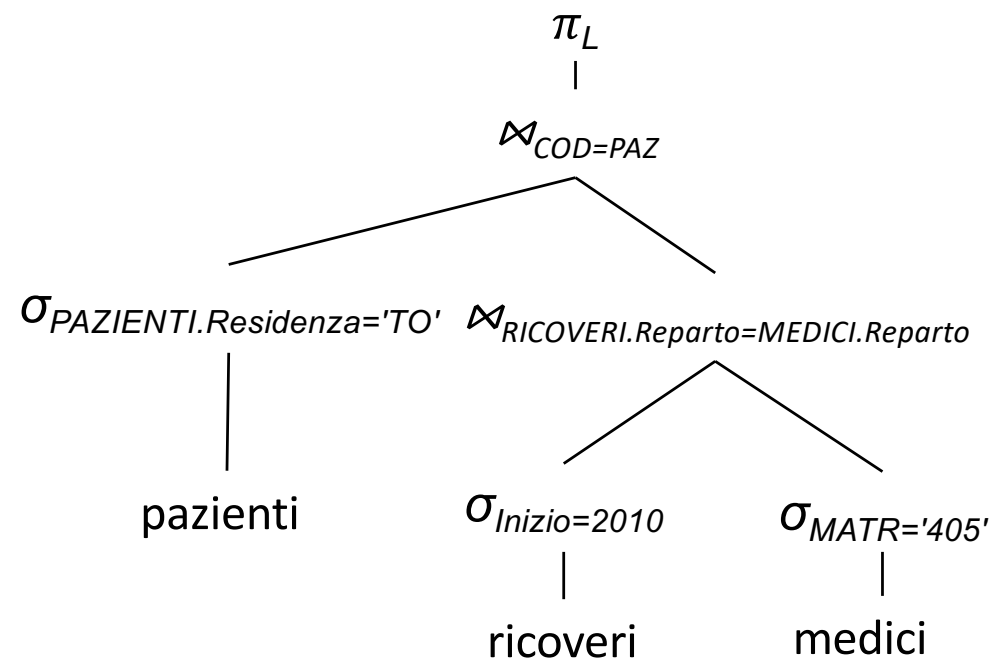
Spiegazione del passo 7

Abbiamo due alternative per i join date dalla proprietà associativa:

(A)

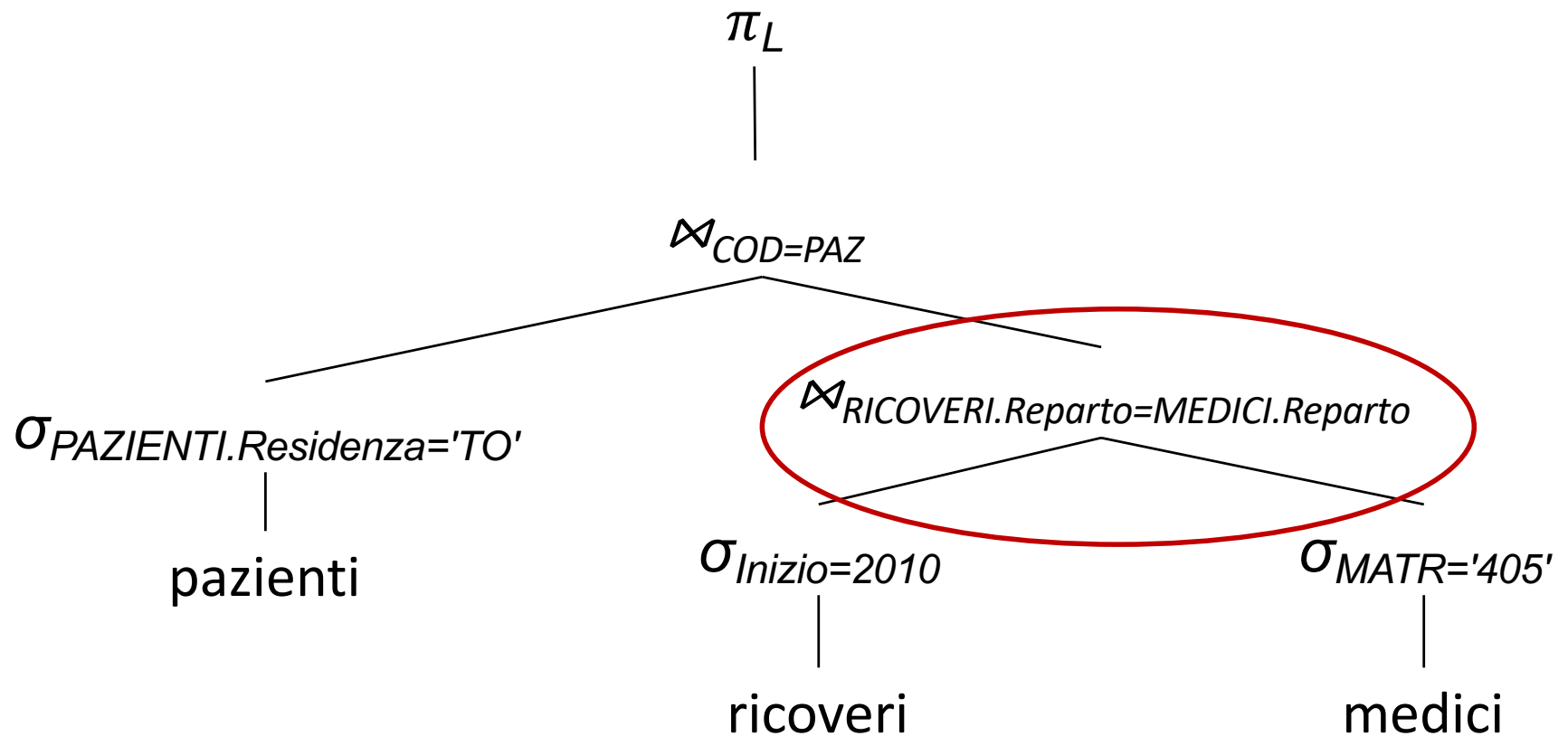


(B)



Il costo di (A) è già stato stimato. Ora stimiamo il costo di (B)

Spiegazione del passo 7



Analisi quantitativa

Il join $\sigma_{p2} \bowtie_{RICOVERI.Reparto=MEDICI.Reparto} \sigma_{p3}$ dovrà elaborare $10^4 \cdot 1 = 10^4$ tuple e produrrà

$$\begin{aligned} & \min(1/VAL(Reparto, \sigma_{p3}(medici)), 1/VAL(Reparto, \sigma_{p2}(ricoveri))) \cdot \\ & \quad CARD(\sigma_{p3}(medici)) \cdot CARD(\sigma_{p2}(ricoveri)) = \\ & \quad \min(1/1, 1/10) \cdot 1 \cdot 10^4 = \\ & \quad 1/10 \cdot 10^4 = 10^3 \end{aligned}$$

Analisi quantitativa

Per l'ultimo join tra $\sigma_{p1}(pazienti)$ e il risultato del join precedente
 $10^3 \cdot 10^3 = 10^6$ tuple elaborate

Qual è la massa di tuple coinvolte dall'interrogazione?

Bisogna sommare le tuple coinvolte da:
tre selezioni + primo join + secondo join:
 $10^2 + 10^5 + 10^5 + 10^4 + 10^6$

È sufficiente considerare il valore dominante, ovvero:

10^6

Guadagniamo un ordine di grandezza rispetto all'alternativa precedente

Cosa succede dopo?

- Quando l'ottimizzatore logico termina il suo lavoro e invia l'albero sintattico all'ottimizzatore fisico, quest'ultimo:
 - cambia i nodi foglia con nodi che tengono conto delle strutture fisiche di accesso ai dati e
 - cambia i nodi intermedi con specifici algoritmi per l'esecuzione degli operatori che sfruttino le strutture fisiche (scansioni sequenziali, ordinamenti, accessi diretti, varie modalità di esecuzione del join, ...)