

# Sistemi per la Gestione delle Basi di Dati

## Quaderno #1

Sono date le relazioni seguenti (le chiavi primarie sono sottolineate):

FILIALE(CodFI, NomeFiliale, Città, Stato)  
DIPENDENTE(MatrD, NomeD, CognomeD, Ruolo, CodFI)  
PROGETTO(CodPR, NomeP, DataInizio, DataFine)  
RIUNIONE(CodRI, CodPR, Data, Sala, OraInizio, DurataInOre)  
PARTECIPA\_RIUNIONE(CodRI, MatrD)

Si ipotizzino le seguenti cardinalità per le tabelle:

- $\text{card}(\text{FILIALE}) \simeq 10^2$  tuple,  
Stato assume  $\simeq 10$  valori distinti,
- $\text{card}(\text{DIPENDENTE}) \simeq 10^3$  tuple,  
Ruolo assume  $\simeq 10$  valori distinti,
- $\text{card}(\text{PROGETTO}) \simeq 10^4$  tuple,  
 $\text{MIN}(\text{DataInizio}) = 1-1-1979$ ,  $\text{MAX}(\text{DataInizio}) = 31-12-2008$ ,
- $\text{card}(\text{RIUNIONE}) \simeq 10^5$  tuple,  
 $\text{MIN}(\text{Data}) = 1-1-1979$ ,  $\text{MAX}(\text{Data}) = 31-12-2008$ ,  
 $\text{MIN}(\text{DurataInOre}) = 2$ ,  $\text{MAX}(\text{DurataInOre}) = 6$ ,
- $\text{card}(\text{PARTECIPA\_RIUNIONE}) \simeq 5 \cdot 10^5$  tuple.

Si ipotizzino i seguenti fattori di riduzione per le condizioni di gruppo:

- nella query (a) fattore di riduzione di  $\text{having count}(\ast) > 10 \simeq \frac{1}{10}$ ,
- nella query (b) fattore di riduzione di  $\text{having sum}(\text{RI.DurataInOre}) > 500 \simeq \frac{1}{500}$ .

Si considerino le seguenti query SQL:

1. 

```
select DI.NomeD, RI.CodPR, SUM(DurataInOre)
from RIUNIONE RI, DIPENDENTE DI, PARTECIPA_RIUNIONE PAR
where PAR.CodRI=RI.CodRI and PAR.MatrD=DI.MatrD
and RI.Data ≥ 1-1-1990 and RI.Data ≤ 31-12-1992
and (DI.Ruolo='Dirigente' OR DI.Ruolo='Segretario')
group by DI.MatrD, DI.NomeD, RI.CodPR
having count(*) > 10
```
2. 

```
select NomeD
from DIPENDENTE DI, FILIALE FI
where DI.CodFI=FI.CodFI and FI.Stato='Italia'
and DI.MatrD not in (select PAR.MatrD
from RIUNIONE RI, PARTECIPA_RIUNIONE PAR, PROGETTO PR
where PAR.CodRI=RI.CodRI and RI.CodPR=PR.CodPR
and PR.DataInizio ≥ 1-1-2006
and RI.Data ≥ 1-1-2006 and RI.Data ≤ 31-12-2008
group by PAR.MatrD
having sum(RI.DurataInOre) > 500)
```

Separatamente per ciascuna delle due interrogazioni SQL

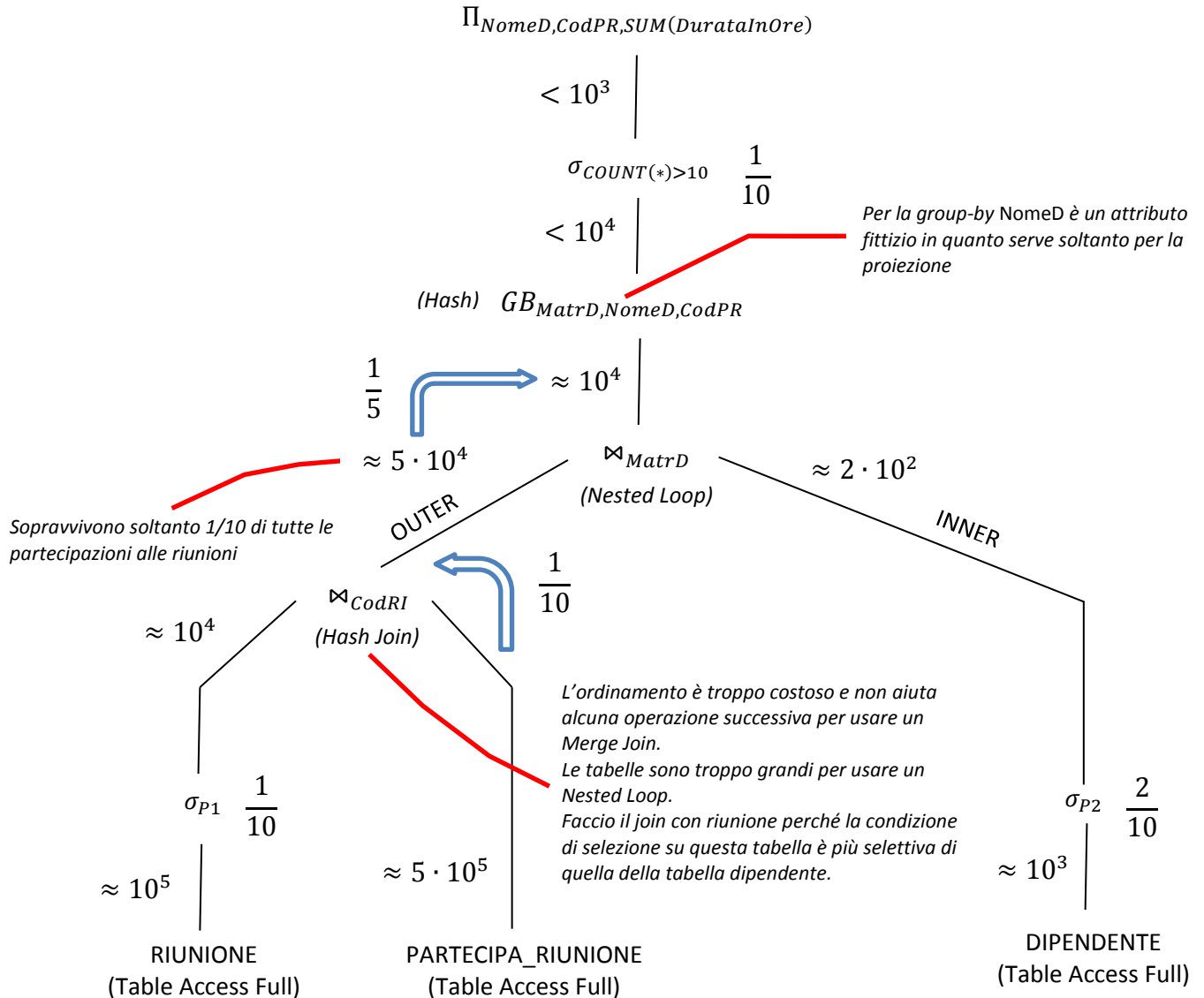
1. Si scriva l'espressione algebrica corrispondente.
2. Si scriva il piano di esecuzione che potrebbe essere scelto dall'ottimizzatore in assenza di strutture fisiche accessorie (ordine e tipo dei join, metodi di accesso alle tabelle, operazioni di filtro, proiezione e raggruppamento).
3. Si scelgano le strutture fisiche accessorie per migliorare le prestazioni dell'interrogazione. Si motivi la scelta e si scriva il nuovo piano di esecuzione ipotizzato utilizzando le strutture fisiche aggiuntive.
4. Dove necessario, si ipotizzi la distribuzione dei dati.

## QUERY 1

### PIANO D'ESECUZIONE SENZA INDICI

$P1 := Data \geq 1-1-1990 \text{ AND } Data \leq 31-12-1992$

$P2 := Ruolo = 'Dirigente' \text{ OR } Ruolo = 'Segretario'$

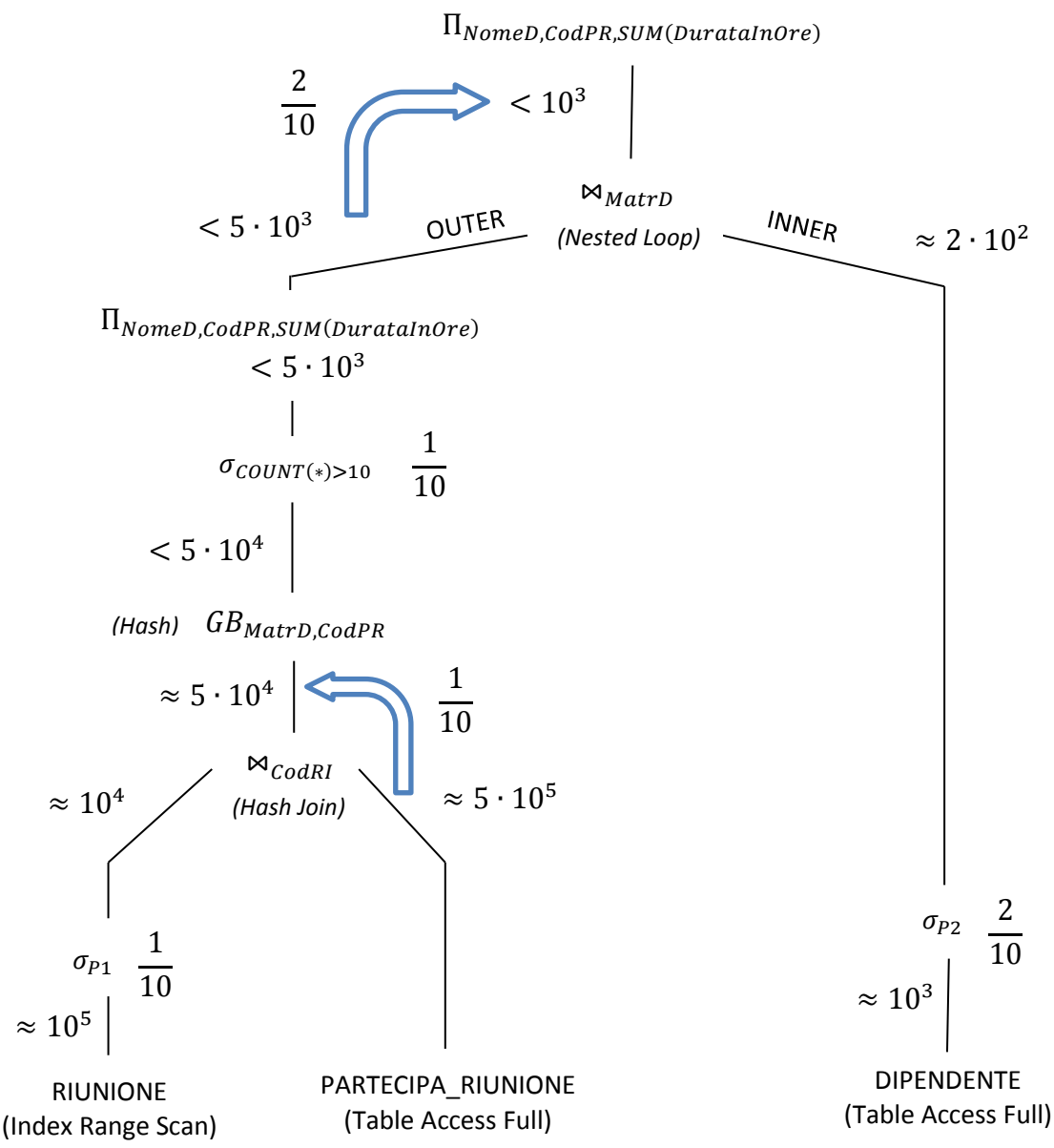


## CONSIDERAZIONI SUGLI INDICI

<ul style="list-style-type: none"> <li>▪ Dipendente(Ruolo) <ul style="list-style-type: none"> <li>↳ Poco selettivo ✗</li> <li>↳ Non coprente ✗</li> <li>↳ Non aiuta operazioni costose successive ✗</li> </ul> </li> </ul>		<ul style="list-style-type: none"> <li>▪ Riunione(Data) <ul style="list-style-type: none"> <li>↳ Selettivo ✓</li> <li>↳ Non coprente ✗</li> <li>↳ Non aiuta operazioni costose successive ✗</li> </ul> </li> </ul>
<ul style="list-style-type: none"> <li>▪ Dipendente(Matrd) <ul style="list-style-type: none"> <li>↳ Poco selettivo ✗</li> <li>↳ Non coprente ✗</li> </ul> </li> </ul>		<ul style="list-style-type: none"> <li>▪ Riunione(CodRI) <ul style="list-style-type: none"> <li>↳ Poco selettivo ✗</li> <li>↳ Non coprente ✗</li> </ul> </li> </ul>
<ul style="list-style-type: none"> <li>▪ Partecipa_Riunione(CodRI, MatrD) <ul style="list-style-type: none"> <li>↳ Coprente ✓</li> <li>↳ Non aiuta il join. Poiché debbo comunque accedere a tutte le tuple della tabella Partecipa_Riunione, un Full Table Scan è più efficiente. ✗</li> </ul> </li> </ul>		<ul style="list-style-type: none"> <li>▪ Partecipa_Riunione(CodRi) <ul style="list-style-type: none"> <li>↳ Non coprente ✗</li> <li>↳ Non aiuta il join (accedo comunque a tutto le tuple di Partecipa_Riunione) ✗</li> </ul> </li> </ul>
<ul style="list-style-type: none"> <li>▪ Partecipa_Riunione(CodRI, MatrD) <ul style="list-style-type: none"> <li>↳ Coprente ✓</li> <li>↳ Non aiuta il join. Poiché debbo comunque accedere a tutte le tuple della tabella Partecipa_Riunione, un Full Table Scan è più efficiente. ✗</li> </ul> </li> </ul>		<ul style="list-style-type: none"> <li>▪ Partecipa_Riunione(CodRi) <ul style="list-style-type: none"> <li>↳ Non coprente ✗</li> <li>↳ Non aiuta il join (accedo comunque a tutto le tuple di Partecipa_Riunione) ✗</li> </ul> </li> </ul>

Alla luce delle considerazioni fatte, si crea un indice B<sup>+</sup>-Tree Unclustered su Riunione(Data). Inoltre, è possibile anticipare l'operazione di Group-by sul ramo sinistro del join, cosicché quest'ultimo sarà fatto su tabelle di dimensioni inferiori.

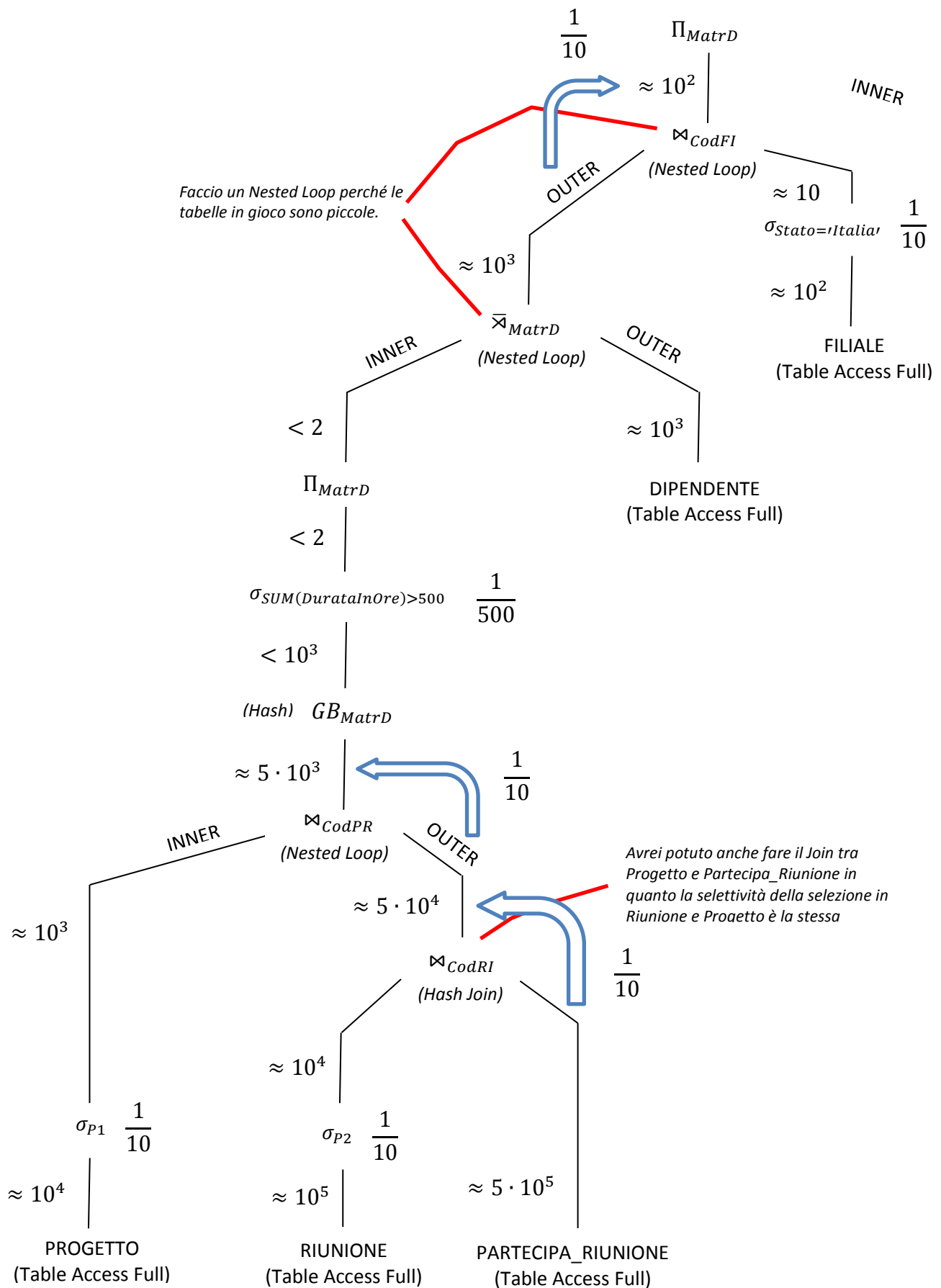
PIANO D'ESECUZIONE CON INDICI



## PIANO D'ESECUZIONE SENZA INDICI

$P1 := DataInizio \geq 1-1-2006$

*P2 := Data ≥ 1-1-2006 AND Data ≤ 31-12-2008*



## CONSIDERAZIONI SUGLI INDICI

▪ Riunione(Data)		▪ Progetto(DataInizio)	
↳ Selettivo	✓	↳ Selettivo	✓
↳ Non coprente	✗	↳ Non coprente	✗
↳ Non aiuta operazioni costose successive	✗	↳ Non aiuta operazioni costose successive	✗
▪ Dipendente(MatrD)		▪ Filiale(Stato)	
↳ Aiuta l'anti-join, che però non è costoso, essendo la cardinalità della INNER < 10. Non conviene usarlo considerando il costo del suo mantenimento	✗	↳ Selettivo	✓
		↳ La tabella è piccola	✗
		↳ Non aiuta operazioni costose successive	✗

Alla luce delle considerazioni fatte, si crea un indice B<sup>+</sup>-Tree Unclustered su Riunione(Data) e Progetto(DataInizio).

Inoltre, è possibile anticipare l'operazione di Group-by sul ramo destro del join, cosicché quest'ultimo sarà fatto su tabelle di dimensioni inferiori.

PIANO D'ESECUZIONE CON INDICI

