Capitolo 9

REALIZZAZIONE DEI DBMS

Soluzione degli esercizi

1. a) Falso. Il primo piano esegue l'interrogazione con un algoritmo che recupera prima i record di *R*, poi li proietta su *A* e infine li ordina. Il secondo piano usa un algoritmo che prima ordina *R* poi recupera i record ordinati e infine li proietta su *A*, producendo lo stesso risultato.
   b) Falso. Un operatore logico è un operatore dell'algebra relazionale, mentre un operatore fisico è un algoritmo per realizzare un operatore logico sfruttando le strutture di memorizzazione disponibili nella macchina fisica.
   c) Vero. L'albero logico iniziale può essere trasformato in modi diversi applicando le regole di equivalenza dell'algebra relazionale.
   d) Vero. Un operatore logico può essere realizzato con algoritmi diversi.
   e) Falso. La tecnica del blocco dei dati può creare situazioni di stallo.
   f) Vero. Il metodo disfare–rifare segue le regole per *disfare* e per *rifare*: prima di modificare la BD scrive nel giornale la vecchia versione e la nuova versione dei dati modificati.
   g) Falso. Il metodo rifare non prevede che le modifiche delle transazioni siano riportate nella BD prima che il record di commit sia scritto nel giornale.

2. L'indice e il risultato sono:

A	TID
1	5
2	2
3	4, 6
6	7
7	1
8	3

A	*B*
7	7
2	2
2	2
3	3
3	3
1	1
3	3
3	3

3. *Albero logico iniziale* (Figura 9.1).
   Piano di accesso (Figura 9.2). Non occore raggruppare perché *A* è chiave. La soluzione non cambia se nella SELECT ci fosse DISTINCT.

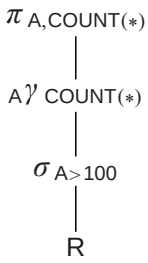

Figura 9.1. Albero logico iniziale

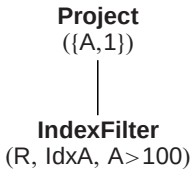

Figura 9.2. Piano di accesso

4. *Albero logico iniziale* (Figura 9.3).

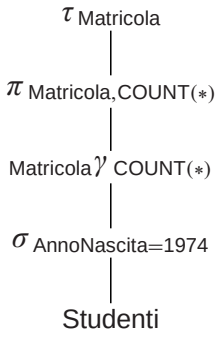

Figura 9.3. Albero logico iniziale

Il piano di accesso non produce il risultato dell'interrogazione.
Completiamo il piano di accesso aggiungendo le parti mancanti, che sono il filtro su AnnoNascita e la funzione di aggregazione COUNT(*) (Figura 9.4):

Adesso eliminiamo gli operatori superflui (Figura 9.5):

Partendo dal basso si eliminano:

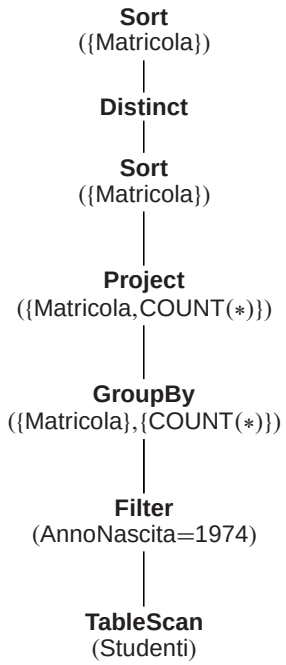

Figura 9.4. Piano completo

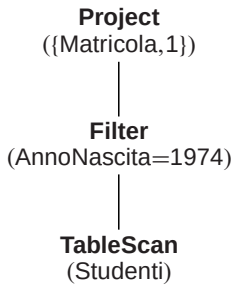

Figura 9.5. Piano semplificato

(a) GroupBy: l'attributo di raggruppamento è chiave e quindi non serve raggruppare.
(b) Project: l'operatore GroupBy restituisce record con campi Matricola, COUNT(*) e quindi il Project è inutile,
(c) Sort: la tabella è memorizzata ordinata sulla chiave Matricola e quindi non occorre ordinare,
(d) Distinct: tra gli attributi dei record che arrivano all'operatore Distinct è compresa una chiave, dunque non possono esistere record uguali,
(e) Sort: non serviva prima, non serve adesso.

Considerando l'indice su AnnoNascita il piano diventa (Figura 9.6):

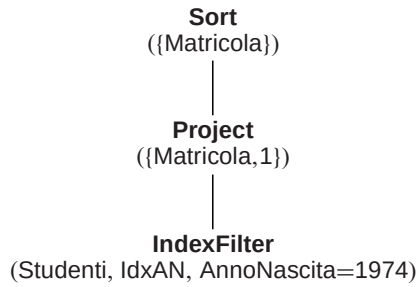

Figura 9.6. Piano finale

5. *Albero logico iniziale* (Figura 9.7).

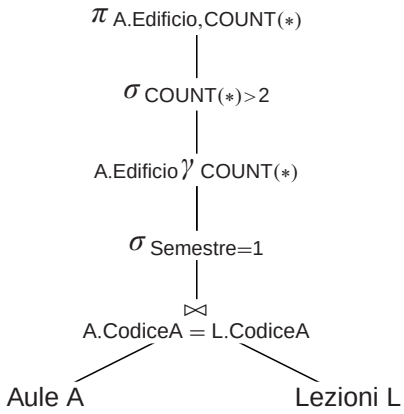

Figura 9.7. Albero logico iniziale

Piano di accesso con NestedLoop (Figura 9.8).

Piano di accesso con IndexNestedLoop (Figura 9.9)

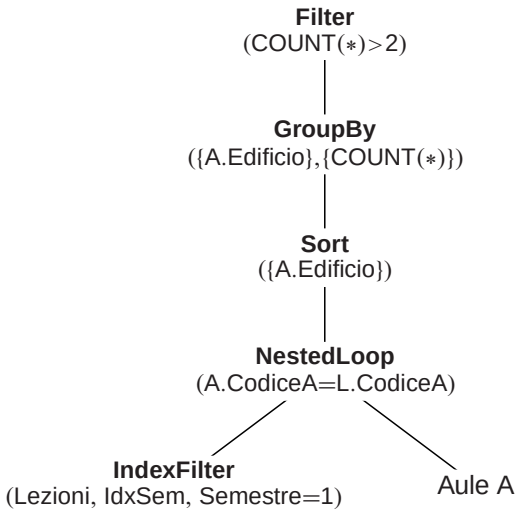

Figura 9.8. Piano di accesso con NestedLoop

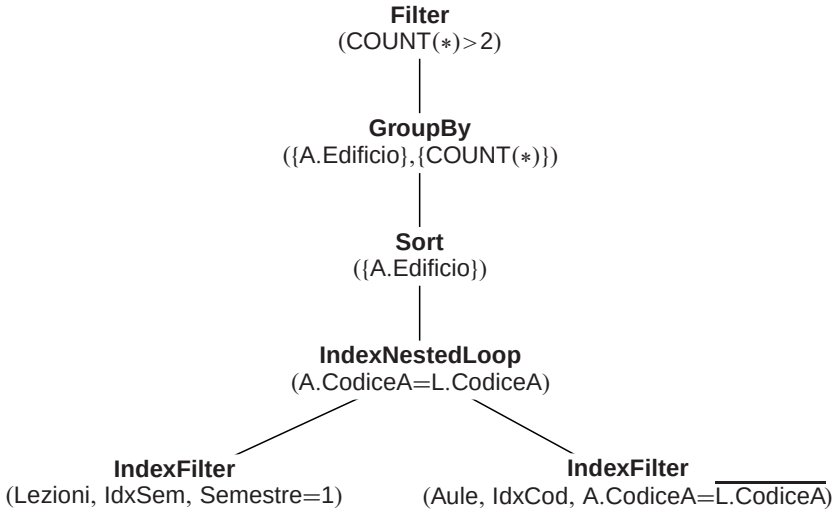

Figura 9.9. Piano di accesso con IndexNestedLoop

.

6. Il piano non è corretto perché mancano i filtri per AnnoNascita = 1974 e per HAVING, il GroupBy richiede i record ordinati su Codice, NomeCl, il SortScan deve essere un IndexFilter per la presenza dell'IndexNestedLoop, e quindi non

produce il risultato cercato. Un piano corretto è (Figura 9.10).

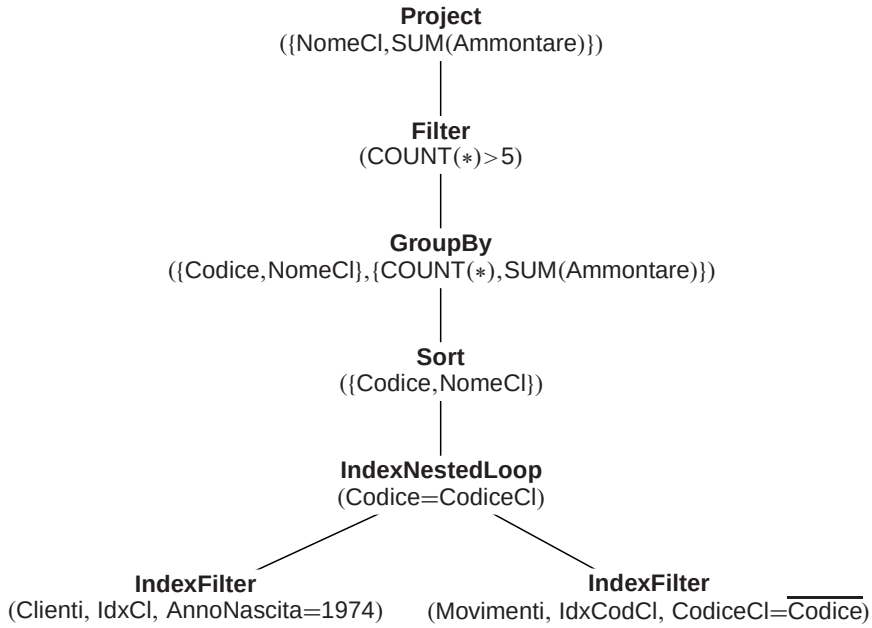

Figura 9.10. Piano di accesso corretto