Basi di dati II

Prova parziale — 25 marzo 2013 — Compito A

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

Run su disco	Buffer	Record prodotti dalle prime sei `next()`

111	...
421	...
722	...
211	...
521	...
322	...
942	...
871	...
601	...
174	...
496	...
575	...
145	...
635	...
946	...
855	...

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

R1(A,B,C), R2(D,E,F)

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from R1 join R2 on C=D`	Hash join

2.	`select A, B, C`
	`from R1 join R2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente L_1=1.000.000 ed L_2=2.000.000 ennuple, (con fattore di blocco rispettivamente f_1=10 e f_2=20)
- gli indici abbiano entrambi i=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.		
2.		

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 5$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 2$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 1000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

Basi di dati II

Prova parziale — 25 marzo 2013 — Compito B

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

Run su disco	Buffer	Record prodotti dalle prime sei `next()`

121	...
401	...
702	...
221	...
501	...
302	...
942	...
871	...
601	...
174	...
496	...
575	...
145	...
635	...
946	...
855	...

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

T1(A,B,C), T2(D,E,F)

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from T1 join T2 on C=D`	Hash join

2.	`select A, B, C`
	`from T1 join T2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente N_1=2.000.000 ed N_2=1.000.000 ennuple, (con fattore di blocco rispettivamente f_1=20 e f_2=10)
- gli indici abbiano entrambi p=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.		
2.		

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 4$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 3$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 2000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

Basi di dati II

Prova parziale — 25 marzo 2013 — Compito C

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

Run su disco	Buffer	Record prodotti dalle prime sei `next()`

101	...
421	...
722	...
201	...
521	...
322	...
942	...
871	...
601	...
174	...
496	...
575	...
145	...
635	...
946	...
855	...

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

T1(A,B,C), T2(D,E,F)

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from T1 join T2 on C=D`	Hash join

2.	`select A, B, C`
	`from T1 join T2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente N_1=2.000.000 ed N_2=2.000.000 ennuple, (con fattore di blocco rispettivamente f_1=20 e f_2=20)
- gli indici abbiano entrambi p=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.		
2.		

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 4$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 3$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 1000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

Basi di dati II

Prova parziale — 25 marzo 2013 — Compito D

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

Run su disco	Buffer	Record prodotti dalle prime sei `next()`

121	...
411	...
712	...
221	...
511	...
312	...
942	...
871	...
601	...
174	...
496	...
575	...
145	...
635	...
946	...
855	...

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

R1(A,B,C), R2(D,E,F)

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from R1 join R2 on C=D`	Hash join

2.	`select A, B, C`
	`from R1 join R2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente L_1=1.000.000 ed L_2=1.000.000 ennuple, (con fattore di blocco rispettivamente f_1=10 e f_2=10)
- gli indici abbiano entrambi i=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.		
2.		

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 5$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 2$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 2000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:

costo accesso diretto:

Basi di dati II

Prova parziale — 25 marzo 2013 — Compito A

Cenni sulle soluzioni

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

111 ...
421 ...
722 ...
211 ...
521 ...
322 ...
942 ...
871 ...
601 ...
174 ...
496 ...
575 ...
145 ...
635 ...
946 ...
855 ...

Run su disco

111 ...
211 ...
322 ...
421 ...
521 ...
722 ...

174 ...
496 ...
575 ...
601 ...
871 ...
942 ...

145 ...
635 ...
946 ...
855 ...

Buffer

521	... ←
722	...

174	...
496	... ←

145	...
635	... ←

Record prodotti dalle prime 5 `next()`

111 ...
145 ...
174 ...
211 ...
322 ...
421 ...

Il numero ideale di buffer (da utilizzare sia nella prima sia nella seconda passata) è pari alla radice quadrata del numero dei blocchi e quindi a tre.
Nella prima passata, si costruiscono quindi tre run ordinati, ciascuno a partire da tre blocchi del file originario. L'ordinamento di ciascun run può essere effettuato in memoria centrale, usando tre buffer. Poiché il risultato della prima passata viene materializzato, vengono mostrati i run ordinati.
Nella seconda passata, si carica un blocco per ciascuno dei run e si fa il merge usando nuovamente tre buffer. La seconda passata viene svolta in pipeline, e quindi i buffer sono mostrati con il contenuto che hanno quando è stato appena prodotto il sesto record.

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

$$R1(\underline{A},B,C),\ R2(\underline{D},E,F)$$

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from R1 join R2 on C=D`	Hash join

2.	`select A, B, C`
	`from R1 join R2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente L_1=1.000.000 ed L_2=2.000.000 ennuple, (con fattore di blocco rispettivamente f_1=10 e f_2=20)
- gli indici abbiano entrambi i=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.	Con i buffer disponibili (che sono in numero maggiore della radice quadrata del numero di blocchi del più piccolo dei due file), l'hash-join si può eseguire in due passate, con un costo: $3 \times (\frac{L_1}{f_1} + \frac{L_2}{f_2}) = 600.000$	$\frac{L_1}{f_1} + L_1 \times (i - 2 + 1) = 3.100.000$ (nei compiti B e C) $= 6.100.000$ (i-2+1): i livelli dell'indice, meno quelli nel buffer (mediamente 2), più 1 per l'accesso al record
2.	sia $a = 5$, numero di valori di A selezionati dalla `where` Viene prima effettuata la selezione sulla relazione R1 e poi il join del risultato con l'altra relazione; la selezione viene effettuata sfruttando l'indice $i + a + \frac{L_2}{f_2} =$ ca 100.000 Alla relazione esterna si accede con l'indice: profondità più scansione, perché un intervallo; le poche ennuple vengono organizzate in memoria secondo la struttua hash e quindi basta poi una scansione della seconda relazione	$(i + a) + a \times (i - 1) =$ ca 25 Alla relazione esterna si accede con l'indice: $(i + a)$ per accedere ai record di interesse di R1; poi a accessi diretti a R2, con la sola radice nel buffer (e limitandosi all'indice, perché il record in effetti non serve, visti gli attributi nella target list)

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 5$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 2$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

La somma del tempo medio di seek, del tempo medio di latenza e del tempo minimo di lettura di un blocco

$$t_{tot} = t_S + t_L + t_B = 5 + 2 + 0,015 \text{ msec } = \text{ ca } 7 \text{ msec}$$

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

Il tempo medio necessario per leggere un blocco (il primo) più il tempo minimo di lettura per ciascuno degli altri

$$t_{tot} + (F - 1) \times t_B = \text{ ca } 8 \text{ msec}$$

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

Si può immaginare che al massimo la radice dell'indice si trovi nel buffer, ma non gli altri nodi. Quindi, tre accessi per l'indice e uno per il record del file, in posizioni non prevedibili:

$$4 \times t_{tot} = \text{ ca } 28 \text{ msec}$$

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 1000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Si può immaginare che la radice e un altro livello dell'indice restino nel buffer, dopo il primo caricamento, quindi, per ogni record, due accessi all'indice e uno al file, in posizioni non prevedibili (qualche accesso in più per il caricamento iniziale e qualcuno in meno per blocchi acceduti più volte):

$$3 \times m \times t_{tot} = \text{ ca } 21 \text{ sec}$$

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_1 = \approx 14$

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_2 + \ldots = \approx 100.000$

Basi di dati II

Prova parziale — 25 marzo 2013 — Compito B

Cenni sulle soluzioni

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

121 ...
401 ...
702 ...
221 ...
501 ...
302 ...
942 ...
871 ...
601 ...
174 ...
496 ...
575 ...
145 ...
635 ...
946 ...
855 ...

Run su disco

121 ...
221 ...
302 ...
401 ...
501 ...
702 ...

174 ...
496 ...
575 ...
601 ...
871 ...
942 ...

145 ...
635 ...
946 ...
855 ...

Buffer

501	... ←
702	...

174	... ←
496	...

145	... ←
635	...

Record prodotti dalle prime 5 `next()`

121 ...
145 ...
174 ...
221 ...
302 ...
401 ...

Il numero ideale di buffer (da utilizzare sia nella prima sia nella seconda passata) è pari alla radice quadrata del numero dei blocchi e quindi a tre.
Nella prima passata, si costruiscono quindi tre run ordinati, ciascuno a partire da tre blocchi del file originario. L'ordinamento di ciascun run può essere effettuato in memoria centrale, usando tre buffer. Poiché il risultato della prima passata viene materializzato, vengono mostrati i run ordinati.
Nella seconda passata, si carica un blocco per ciascuno dei run e si fa il merge usando nuovamente tre buffer. La seconda passata viene svolta in pipeline, e quindi i buffer sono mostrati con il contenuto che hanno quando è stato appena prodotto il sesto record.

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

$$T1(\underline{A},B,C),\ T2(\underline{D},E,F)$$

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from T1 join T2 on C=D`	Hash join

2.	`select A, B, C`
	`from T1 join T2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente N_1=2.000.000 ed N_2=1.000.000 ennuple, (con fattore di blocco rispettivamente f_1=20 e f_2=10)
- gli indici abbiano entrambi p=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.	Con i buffer disponibili (che sono in numero maggiore della radice quadrata del numero di blocchi del più piccolo dei due file), l'hash-join si può eseguire in due passate, con un costo: $3 \times (\frac{N_1}{f_1} + \frac{N_2}{f_2}) = 600.000$	$\frac{N_1}{f_1} + N_1 \times (p - 2 + 1) = 3.100.000$ (nei compiti B e C) $= 6.100.000$ (p-2+1): i livelli dell'indice, meno quelli nel buffer (mediamente 2), più 1 per l'accesso al record
2.	sia $a = 5$, numero di valori di A selezionati dalla `where` Viene prima effettuata la selezione sulla relazione T1 e poi il join del risultato con l'altra relazione; la selezione viene effettuata sfruttando l'indice $p + a + \frac{N_2}{f_2} =$ ca 100.000 Alla relazione esterna si accede con l'indice: profondità più scansione, perché un intervallo; le poche ennuple vengono organizzate in memoria secondo la struttua hash e quindi basta poi una scansione della seconda relazione	$(p + a) + a \times (p - 1) =$ ca 25 Alla relazione esterna si accede con l'indice: $(p + a)$ per accedere ai record di interesse di T1; poi a accessi diretti a T2, con la sola radice nel buffer (e limitandosi all'indice, perché il record in effetti non serve, visti gli attributi nella target list)

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 4$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 3$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

La somma del tempo medio di seek, del tempo medio di latenza e del tempo minimo di lettura di un blocco

$$t_{tot} = t_S + t_L + t_B = 4 + 3 + 0,015 \text{ msec} = \text{ ca } 7 \text{ msec}$$

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

Il tempo medio necessario per leggere un blocco (il primo) più il tempo minimo di lettura per ciascuno degli altri

$$t_{tot} + (F - 1) \times t_B = \text{ ca } 8 \text{ msec}$$

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

Si può immaginare che al massimo la radice dell'indice si trovi nel buffer, ma non gli altri nodi. Quindi, tre accessi per l'indice e uno per il record del file, in posizioni non prevedibili:

$$4 \times t_{tot} = \text{ ca } 28 \text{ msec}$$

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 2000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Si può immaginare che la radice e un altro livello dell'indice restino nel buffer, dopo il primo caricamento, quindi, per ogni record, due accessi all'indice e uno al file, in posizioni non prevedibili (qualche accesso in più per il caricamento iniziale e qualcuno in meno per blocchi acceduti più volte):

$$3 \times m \times t_{tot} = \text{ ca } 42 \text{ sec}$$

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_1 = \approx 14$

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_2 + \ldots = \approx 100.000$

Basi di dati II

Prova parziale — 25 marzo 2013 — Compito C

Cenni sulle soluzioni

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

101 ...
421 ...
722 ...
201 ...
521 ...
322 ...
942 ...
871 ...
601 ...
174 ...
496 ...
575 ...
145 ...
635 ...
946 ...
855 ...

Run su disco

101 ...
201 ...
322 ...
421 ...
521 ...
722 ...

174 ...
496 ...
575 ...
601 ...
871 ...
942 ...

145 ...
635 ...
946 ...
855 ...

Buffer

521	...
722	...

←

174	...
496	...

←

145	...
635	...

←

Record prodotti dalle prime 5 `next()`

101 ...
145 ...
174 ...
201 ...
322 ...
421 ...

Il numero ideale di buffer (da utilizzare sia nella prima sia nella seconda passata) è pari alla radice quadrata del numero dei blocchi e quindi a tre.
Nella prima passata, si costruiscono quindi tre run ordinati, ciascuno a partire da tre blocchi del file originario. L'ordinamento di ciascun run può essere effettuato in memoria centrale, usando tre buffer. Poiché il risultato della prima passata viene materializzato, vengono mostrati i run ordinati.
Nella seconda passata, si carica un blocco per ciascuno dei run e si fa il merge usando nuovamente tre buffer. La seconda passata viene svolta in pipeline, e quindi i buffer sono mostrati con il contenuto che hanno quando è stato appena prodotto il sesto record.

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

$$T1(\underline{A},B,C),\ T2(\underline{D},E,F)$$

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from T1 join T2 on C=D`	Hash join

2.	`select A, B, C`
	`from T1 join T2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente N_1=2.000.000 ed N_2=2.000.000 ennuple, (con fattore di blocco rispettivamente f_1=20 e f_2=20)
- gli indici abbiano entrambi p=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.	Con i buffer disponibili (che sono in numero maggiore della radice quadrata del numero di blocchi del più piccolo dei due file), l'hash-join si può eseguire in due passate, con un costo: $3 \times (\frac{N_1}{f_1} + \frac{N_2}{f_2}) = 600.000$	$\frac{N_1}{f_1} + N_1 \times (p - 2 + 1) = 3.100.000$ (nei compiti B e C) $= 6.100.000$ (p-2+1): i livelli dell'indice, meno quelli nel buffer (mediamente 2), più 1 per l'accesso al record
2.	sia $a = 5$, numero di valori di A selezionati dalla `where` Viene prima effettuata la selezione sulla relazione T1 e poi il join del risultato con l'altra relazione; la selezione viene effettuata sfruttando l'indice $p + a + \frac{N_2}{f_2} =$ ca 100.000 Alla relazione esterna si accede con l'indice: profondità più scansione, perché un intervallo; le poche ennuple vengono organizzate in memoria secondo la struttua hash e quindi basta poi una scansione della seconda relazione	$(p + a) + a \times (p - 1) =$ ca 25 Alla relazione esterna si accede con l'indice: $(p + a)$ per accedere ai record di interesse di T1; poi a accessi diretti a T2, con la sola radice nel buffer (e limitandosi all'indice, perché il record in effetti non serve, visti gli attributi nella target list)

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 4$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 3$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

La somma del tempo medio di seek, del tempo medio di latenza e del tempo minimo di lettura di un blocco

$$t_{tot} = t_S + t_L + t_B = 4 + 3 + 0,015 \text{ msec } = \text{ ca } 7 \text{ msec}$$

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

Il tempo medio necessario per leggere un blocco (il primo) più il tempo minimo di lettura per ciascuno degli altri

$$t_{tot} + (F - 1) \times t_B = \text{ ca } 8 \text{ msec}$$

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

Si può immaginare che al massimo la radice dell'indice si trovi nel buffer, ma non gli altri nodi. Quindi, tre accessi per l'indice e uno per il record del file, in posizioni non prevedibili:

$$4 \times t_{tot} = \text{ ca } 28 \text{ msec}$$

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 1000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Si può immaginare che la radice e un altro livello dell'indice restino nel buffer, dopo il primo caricamento, quindi, per ogni record, due accessi all'indice e uno al file, in posizioni non prevedibili (qualche accesso in più per il caricamento iniziale e qualcuno in meno per blocchi acceduti più volte):

$$3 \times m \times t_{tot} = \text{ ca } 21 \text{ sec}$$

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_1 = \approx 14$

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_2 + \ldots = \approx 100.000$

Basi di dati II

Prova parziale — 25 marzo 2013 — Compito D

Cenni sulle soluzioni

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (25%)
Considerare la relazione sotto schematizzata, definita su vari attributi, uno dei quali è la chiave, i cui valori sono mostrati. Supponendo una disponibilità di buffer abbastanza ampia, ma non sufficiente a caricare in memoria l'intera relazione (supporre ad esempio una disponibilità di 3 buffer, con un fattore di blocco pari a 2 e quindi uno spazio occupato dalla relazione pari a 8 blocchi), considerare l'esecuzione di un mergesort a più vie (e due passate) sulla relazione e mostrare lo stato delle strutture in memoria centrale e secondaria dopo l'esecuzione di sei chiamate al metodo `next()` sullo scan che implementa il mergesort. In particolare, mostrare i "run" (cioè le porzioni di file ordinate durante prima passata) memorizzati su disco e i buffer in memoria centrale, evidenziando per ciascun buffer il record corrente. Mostrare anche i record prodotti dalle prime sei chiamate di `next()`.

121 ...
411 ...
712 ...
221 ...
511 ...
312 ...
942 ...
871 ...
601 ...
174 ...
496 ...
575 ...
145 ...
635 ...
946 ...
855 ...

Run su disco

121 ...
221 ...
312 ...
411 ...
511 ...
712 ...

174 ...
496 ...
575 ...
601 ...
871 ...
942 ...

145 ...
635 ...
946 ...
855 ...

Buffer

511	...
712	...

←

174	...
496	...

←

145	...
635	...

←

Record prodotti dalle prime 5 `next()`

121 ...
145 ...
174 ...
221 ...
312 ...
411 ...

Il numero ideale di buffer (da utilizzare sia nella prima sia nella seconda passata) è pari alla radice quadrata del numero dei blocchi e quindi a tre.
Nella prima passata, si costruiscono quindi tre run ordinati, ciascuno a partire da tre blocchi del file originario. L'ordinamento di ciascun run può essere effettuato in memoria centrale, usando tre buffer. Poiché il risultato della prima passata viene materializzato, vengono mostrati i run ordinati.
Nella seconda passata, si carica un blocco per ciascuno dei run e si fa il merge usando nuovamente tre buffer. La seconda passata viene svolta in pipeline, e quindi i buffer sono mostrati con il contenuto che hanno quando è stato appena prodotto il sesto record.

Domanda 2 (25%)
Si consideri una base di dati con le relazioni (entrambe con indice sulla chiave primaria, costituita, in entrambi i casi, da valori interi positivi consecutivi, a partire da 1)

$$R1(\underline{A},B,C),\ R2(\underline{D},E,F)$$

Eseguendo le interrogazioni seguenti, su Postgres (ma il comportamento su altri sistemi è probabilmente lo stesso), si rilevano le seguenti scelte per l'operatore di join:

1.	`select *` `from R1 join R2 on C=D`	Hash join

2.	`select A, B, C`
	`from R1 join R2 on C=D`
	`where A>= 21 AND A<=25`
		Nested loop join

Motivare ciò, valutando, per ciascuna delle due interrogazioni, il costo di un piano di esecuzione con hash join e uno con nested loop, e supponendo che

- le relazioni abbiano rispettivamente L_1=1.000.000 ed L_2=1.000.000 ennuple, (con fattore di blocco rispettivamente f_1=10 e f_2=10)
- gli indici abbiano entrambi i=4 livelli (contando anche radice e foglie) e il fattore di blocco massimo dell'indice sia, in entrambi i casi, f_i=100
- l'operazione possa contare su un numero di pagine di buffer pari a circa q=500.

Rispondere riempiendo la tabella sottostante, indicando il costo in modo sia simbolico sia numerico (considerare, ovviamente, anche l'eventuale selezione)

	Hash join	Nested loop join
1.	Con i buffer disponibili (che sono in numero maggiore della radice quadrata del numero di blocchi del più piccolo dei due file), l'hash-join si può eseguire in due passate, con un costo: $3 \times (\frac{L_1}{f_1} + \frac{L_2}{f_2}) = 600.000$	$\frac{L_1}{f_1} + L_1 \times (i - 2 + 1) = 3.100.000$ (nei compiti B e C) $= 6.100.000$ (i-2+1): i livelli dell'indice, meno quelli nel buffer (mediamente 2), più 1 per l'accesso al record
2.	sia $a = 5$, numero di valori di A selezionati dalla `where` Viene prima effettuata la selezione sulla relazione R1 e poi il join del risultato con l'altra relazione; la selezione viene effettuata sfruttando l'indice $i + a + \frac{L_2}{f_2} =$ ca 100.000 Alla relazione esterna si accede con l'indice: profondità più scansione, perché un intervallo; le poche ennuple vengono organizzate in memoria secondo la struttua hash e quindi basta poi una scansione della seconda relazione	$(i + a) + a \times (i - 1) =$ ca 25 Alla relazione esterna si accede con l'indice: $(i + a)$ per accedere ai record di interesse di R1; poi a accessi diretti a R2, con la sola radice nel buffer (e limitandosi all'indice, perché il record in effetti non serve, visti gli attributi nella target list)

Domanda 3 (25%) Considerare un sistema con dischi con $N = 400$ blocchi per traccia e con

- tempo medio di posizionamento della testina (tempo di seek) $t_S = 5$ msec
- tempo medio di latenza (attesa dovuta alla rotazione) $t_L = 2$ msec
- tempo minimo di lettura di un blocco $t_B = 10$ μsec

Rispondere alle seguenti domande mostrando formula e valore numerico (N.B. non servono calcolatrici, i calcoli sono semplici; **se si ritengono le informazioni imprecise, dare risposte approssimate, usando il buon senso**).

1. Qual è il tempo medio necessario per leggere un blocco del quale sia dato l'indirizzo fisico?

La somma del tempo medio di seek, del tempo medio di latenza e del tempo minimo di lettura di un blocco

$$t_{tot} = t_S + t_L + t_B = 5 + 2 + 0,015 \text{ msec } = \text{ ca } 7 \text{ msec}$$

2. Qual è il tempo medio necessario per la scansione sequenziale di un file costituito da $F = 100$ blocchi contigui, non letti di recente?

Il tempo medio necessario per leggere un blocco (il primo) più il tempo minimo di lettura per ciascuno degli altri

$$t_{tot} + (F - 1) \times t_B = \text{ ca } 8 \text{ msec}$$

3. Qual è il tempo che si può ipotizzare necessario per eseguire un accesso diretto ad un record di un file attraverso un indice che abbia profondità $p = 4$ e fan-out (fattore di blocco dell'indice) $f_I = 100$ e che sia stato usato di recente, ma in modo non molto intenso?

Si può immaginare che al massimo la radice dell'indice si trovi nel buffer, ma non gli altri nodi. Quindi, tre accessi per l'indice e uno per il record del file, in posizioni non prevedibili:

$$4 \times t_{tot} = \text{ ca } 28 \text{ msec}$$

4. Qual è il tempo che si può ipotizzare necessario per eseguire $m = 2000$ accessi diretti in tempi ravvicinati a record di un file (molto grande) attraverso un indice che abbia profondità $p = 4$, fan-out $f_I = 100$, con disponibilità di circa $P = 150$ pagine di buffer?

Si può immaginare che la radice e un altro livello dell'indice restino nel buffer, dopo il primo caricamento, quindi, per ogni record, due accessi all'indice e uno al file, in posizioni non prevedibili (qualche accesso in più per il caricamento iniziale e qualcuno in meno per blocchi acceduti più volte):

$$3 \times m \times t_{tot} = \text{ ca } 42 \text{ sec}$$

Domanda 4 (25%) Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_1 = \approx 14$

2. campo poco selettivo ($v_2 = 10$ valori diversi per *Categoria*)

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_2 + \ldots = \approx 100.000$