TIPOLOGIA A

1. Esercizi tipologia A

A. Data una query SQL/algebra relazionale:
(a) se SQL, fornire la rappresentazione canonica in algebra relazionale
(b) fornire una espressione algebrica equivalente a quella canonica o a quella data ma più/meno efficiente
(c) fornire l'espressione algebrica ottimale equivalente a quella canonica o a quella data
(d) individuare i fattori booleani nella condizione

ESERCIZIO 1

Si considerino le relazioni con schema:

PIZZA (CodP, *Nome*, Costo)
CLIENTE(TelC, NomeC, Via, NCiv, NInt)
ORDINE(TelC^{CLIENTE}, Data, CodP^{PIZZA}, Qta)

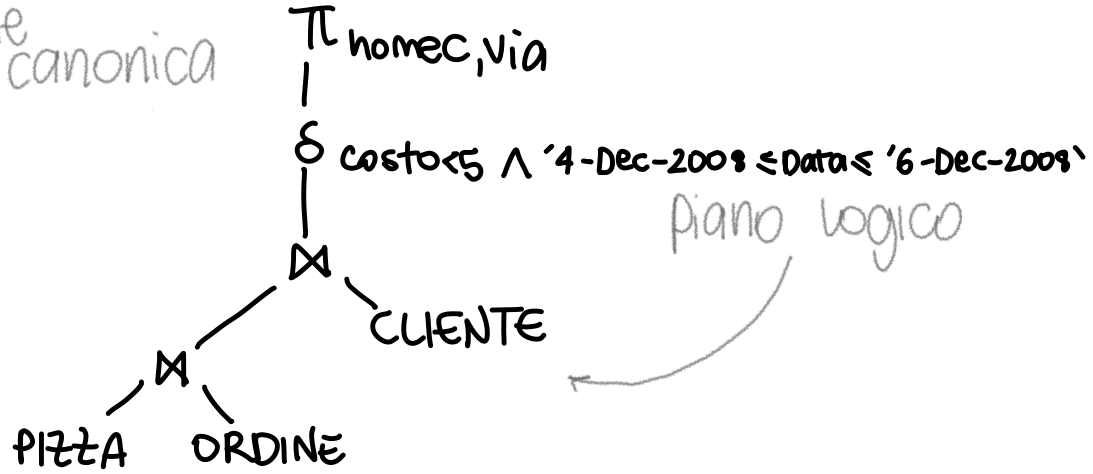
e la query

```
SELECT NomeC, Via
FROM PIZZA NATURAL JOIN ORDINE NATURAL JOIN CLIENTE
WHERE Costo < 5 AND Data  BETWEEN (DATE '4-Dec-2008' AND DATE '6-Dec-2008');
```

1. Presentare il piano logico corrispondente all'espressione algebrica canonica.

$\pi_{nomeC,via}(\delta_{costo<5 \wedge '4\text{-}Dec\text{-}2008' \leq Data \leq '6\text{-}Dec\text{-}2008'}(PIZZA \bowtie ORDINE \bowtie CLIENTE))$

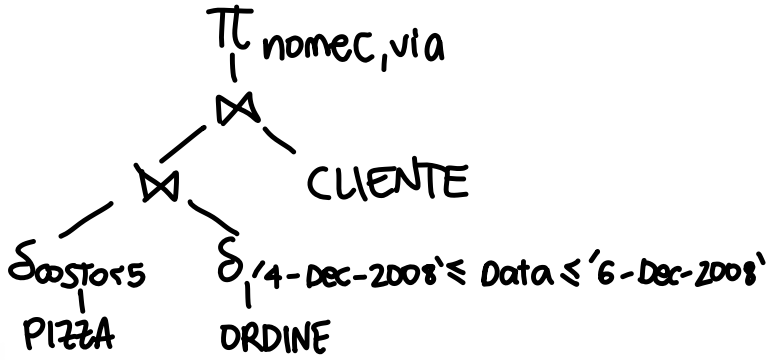
espressione algebrica canonica



piano logico

2. Individuare un piano logico equivalente al piano logico canonico ma più efficiente.

Ci sono due modi per renderlo più efficiente, anticipando o la selezione o la proiezione



3. Individuare i fattori booleani per la query.

Sono costo < 5 e '4-Dec-2008' ≤ Data ≤ '6 Dec 2008'

4. Cosa cambierebbe relativamente ai fattori booleani se la query fosse:

SELECT NomeC, Via
FROM PIZZA NATURAL JOIN ORDINE NATURAL JOIN CLIENTE
WHERE Costo < 5 OR Data BETWEEN (DATE '4-Dec-2008' AND DATE '6-Dec-2008');

non esistono fattori booleani

GIUGNO 2025

Esercizio III – Elaborazione delle interrogazioni

Si consideri il seguente schema relazionale:

IMPIEGATO(eid, did^{DIPARTIMENTO}, sal, hobby)
DIPARTIMENTO(did, nomedip, piano, telefono)
BILANCIO(did^{DIPARTIMENTO}, budget, uscite, entrate)

e la seguente interrogazione:

SELECT nomedip, budget
FROM IMPIEGATO NATURAL JOIN DIPARTIMENTO NATURAL JOIN BILANCIO
WHERE piano = 1 AND sal = 59000 AND hobby = "giardinaggio";

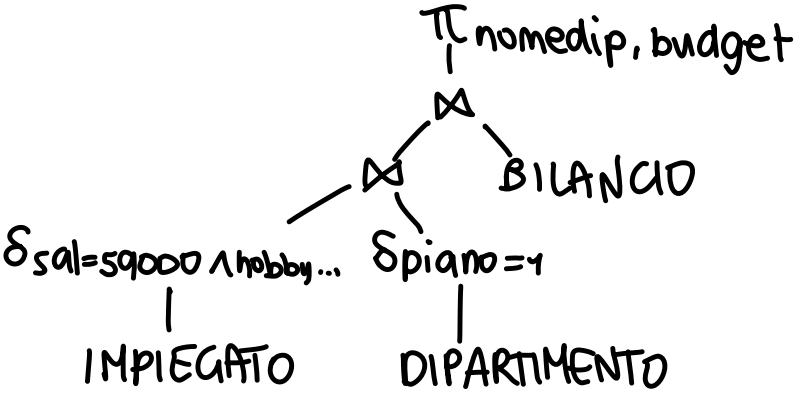
e le seguenti informazioni sullo schema fisico:

- indici ordinati (B+ alberi) su IMPIEGATO.eid, DIPARTIMENTO.piano, DIPARTIMENTO.did e BILANCIO.did; indice hash su IMPIEGATO.sal.
- i salari distinti degli impiegati sono 500, gli hobby diversi degli impiegati sono 200 e la compagnia occupa due piani dell'edificio;
- nella base di dati ci sono 50.000 impiegati e 5.000 dipartimenti (ognuno con le relative informazioni sul bilancio);

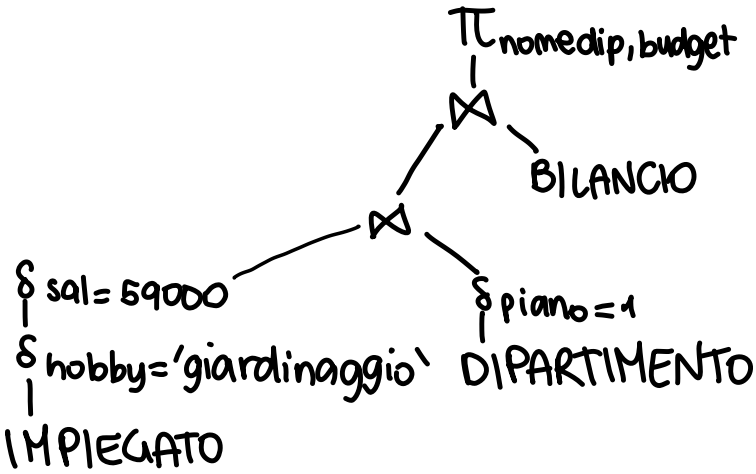
[Motivare tutte le risposte]

2. Presentare, se esiste, un piano logico di esecuzione più efficiente del piano canonico, evitando di inserire operazioni inutili.

$\pi_{nomedip, budget}(\delta_{piano=1 \wedge sal=59000 \wedge hobby='giardinaggio'}(IMPIEGATO \bowtie DIPARTIMENTO \bowtie BILANCIO))$



3. Presentare, se esiste, un piano logico di esecuzione meno efficiente del piano canonico, evitando di inserire operazioni inutili.



4. Supponendo che nel piano logico di esecuzione tutte le selezioni vengano eseguite prima del join mentre la proiezione venga eseguita dopo, stimare quante tuple verranno selezionate per la relazione IMPIEGATO.

La formula è F(C1 AND C2) = F(C1) * F(C2)

quindi:

F(sal = 59000 AND hobby = 'giardinaggio') =

F(sal = 59000) * F(hobby = 'giardinaggio')

Usiamo quindi la formula F(A=v) = 1/V(A,R)

Quindi: 1/500 * 1/200 = 1/100000

numero salari — numero hobby

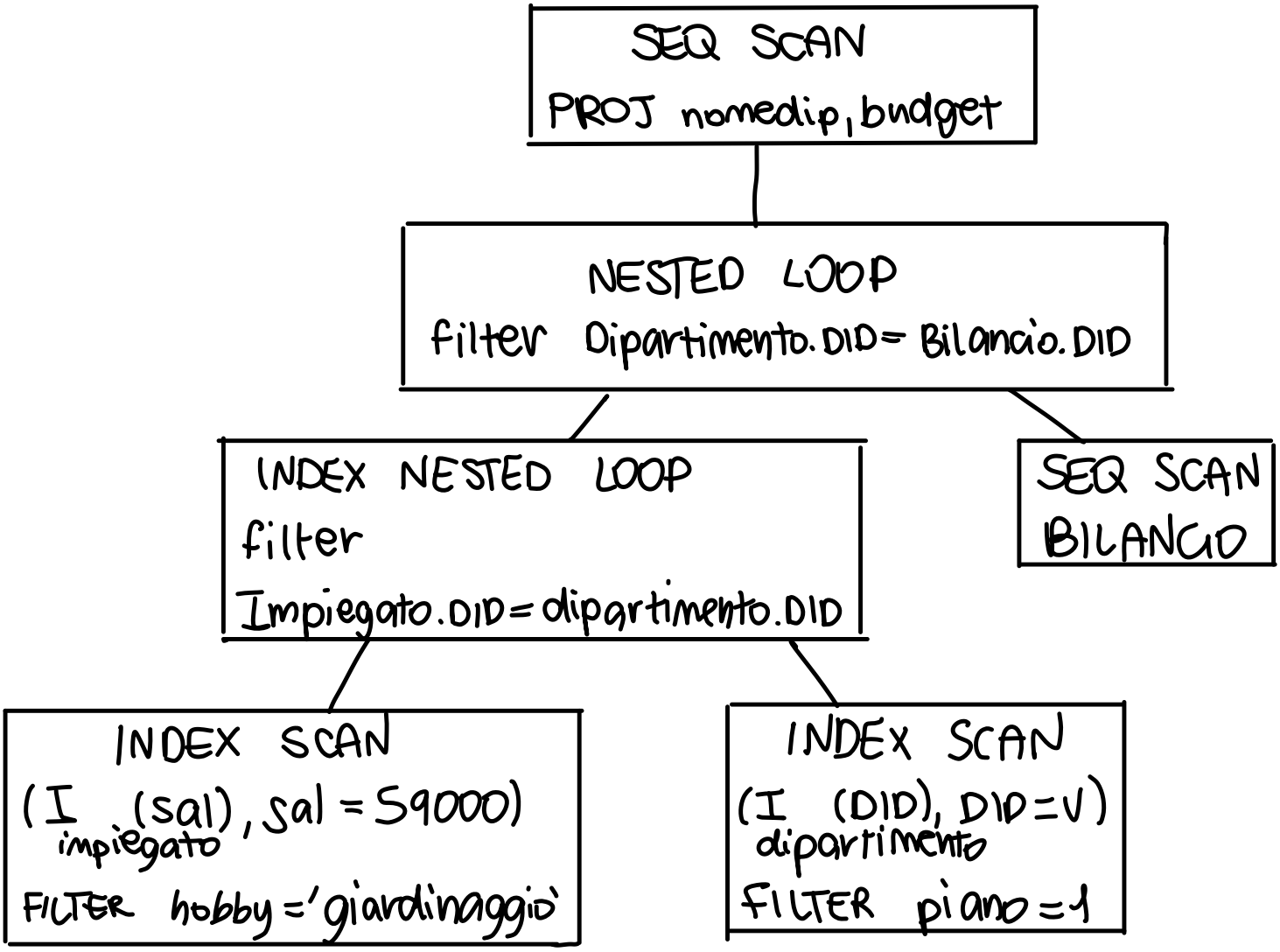
50.000 * 1/100000 = 1/2

numero dipendenti

STIMA DELLA DIMENSIONE DEL RISULTATO DELLA SELEZIONE

- F(A=v) = 1/V(A,R)
- F(A IN (v1, v2,.....,vN))= N * F(A=v)
- F(A>v) = (Max(A,R) - v)/(Max(A,R)-Min(A,R))
- F(A<v) = (v-Min(A,R))/(Max(A,R)-Min(A,R))
- F(A1 = A2) = 1/ MAX (V(A1,R), V(A2,R))
- F(C1 AND C2) = F(C1) *F(C2)
- F(C1 OR C2) = F(C1) +F(C2) -F(C1) *F(C2)
- F(NOT C) = 1 - F(C)

5. Presentare un piano fisico di esecuzione che utilizzi un cammino di accesso indicizzato per almeno una relazione di base.



2. Esercizi tipologia B

B. Data una query SQL/algebra relazionale e alcune informazioni relative allo schema fisico
(a) individuare i cammini di accesso a una relazione
(b) individuare e rappresentare graficamente un piano fisico che soddisfa determinate caratteristiche (se esiste)
(c) determinare se un certo piano fisico ammette elaborazione in pipelining
(d) calcolare una stima della cardinalità del risultato di una query di selezione
(e) determinare quali indici creare per favorire la scelta di un piano di esecuzione fisico ritenuto ottimale
(f) discutere l'impatto della clusterizzazione in un certo piano fisico

ESERCIZIO 1

Si considerino le relazioni con schema:

PIZZA (CodP, *Nome*, Costo)
CLIENTE(TelC, NomeC, Via, NCiv, NInt)
ORDINE(TelC^{CLIENTE}, Data, CodP^{PIZZA}, Qta)

si assuma inoltre che i cataloghi di sistema includano questo contenuto:

Relazioni

R	T(R)	B(R)
PIZZA	30	5
CLIENTE	500	100
ORDINE	10000	2500

Cluster

NomeC	Tipo	Su
C1	Index	PIZZA.Costo
C2	Hash	CLIENTE.TelC
C3	Hash	ORDINE.CodP

Indici

Chiave di ricerca	Tipo	H(I)	L(I)	CARD(I)
PIZZA.Costo	B+	2	5	20
PIZZA.Nome	B+	2	5	30
ORDINE.CodP	Hash			
ORDINE.Data	B+	2	30	300
ORDINE.TelC	B+	2	50	500
CLIENTE.TelC	Hash			

Attributi

A	R	V(A,R)	Min(A,R)	Max(A,R)
CodP	PIZZA	30		
Nome	PIZZA	30		
Costo	PIZZA	20	4	14
TelC	CLIENTE	500		
TelC	ORDINE	500		
Data	ORDINE	300	'01-Jan-2008'	'09-Jan-2009'
CodP	ORDINE	30		
Qta	ORDINE	15	1	20

Considerare l'interrogazione

```
SELECT NomeC, Via
FROM PIZZA NATURAL JOIN ORDINE NATURAL JOIN CLIENTE
WHERE Costo < 5 AND Data  BETWEEN (DATE '4-Dec-2008' AND DATE '6-Dec-2008');
```

1. Determinare tutti i possibili cammini di accesso per le relazioni PIZZA, ORDINE e CLIENTE (considerare a questo proposito sia le condizioni di selezione sia le condizioni di join)

PIZZA

- Scansione sequenziale (vale sempre)
- I_{pizza} (COSTO), COSTO < 5
- l'attributo CodP è anche attributo di Join con ORDINE. Verifico se esiste indice (ordinato o hash) su CodP. Non esiste. Quindi non ci sono altri cammini di accesso.

ORDINE

- Scansione sequenziale
- I_{ordine} (DATA), '4-Dec-2008' ≤ Data ≤ '6-Dec-2008'
- I_{ordine} (CODP), codP = V con V attributo di Join
- I_{ordine} (TELC), telC = V con V attributo di Join

CLIENTE

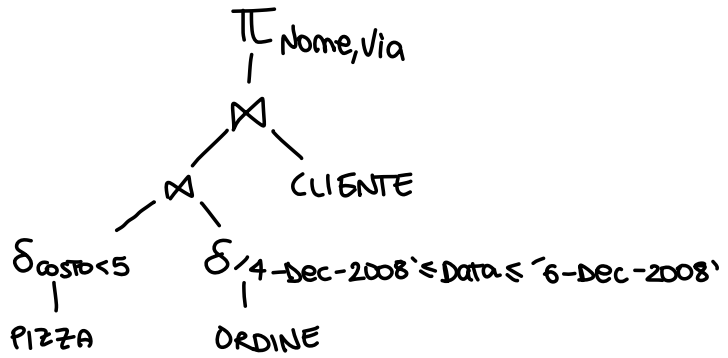
- Scansione sequenziale
- $I_{cliente}$(TELC), telC=V con V attributo di Join

2. Presentare un esempio di piano di esecuzione fisico per l'interrogazione, supponendo di utilizzare un cammino di accesso sequenziale per tutte le relazioni, partendo dal piano di esecuzione logico in cui le selezioni vengono eseguite prima del join e la proiezione dopo.

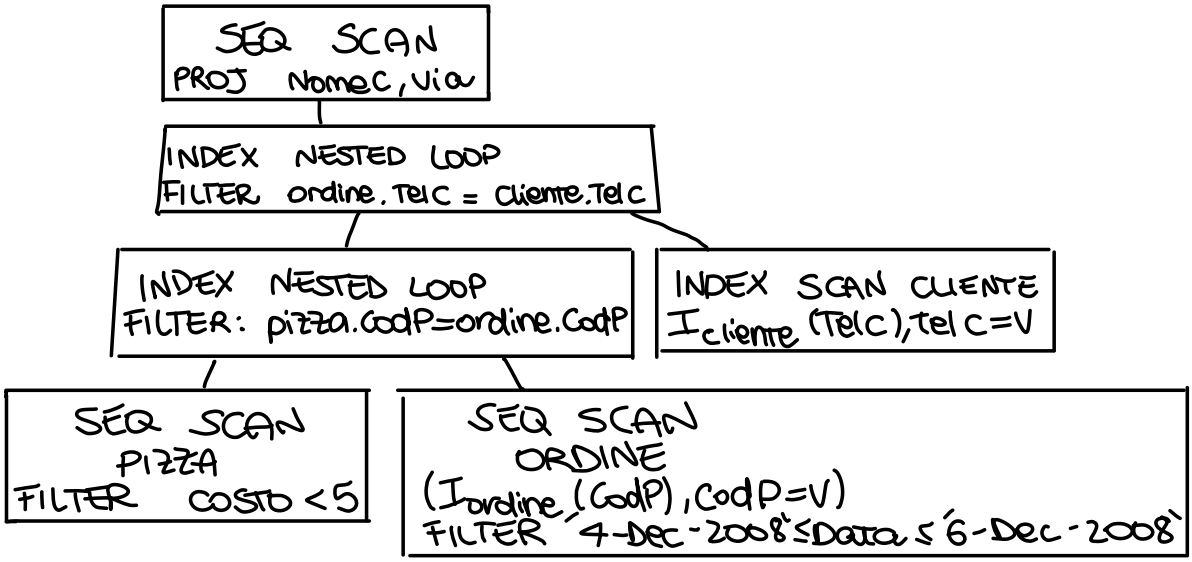
Riscrivo il codice SQL tradotto in Algebra Relazionale:

$\pi_{Nome,Via}(\delta_{Costo<5 \wedge '4\text{-}Dec\text{-}2008' \leq Date \leq '6\text{-}Dec\text{-}2008'}(PIZZA \bowtie ORDINE \bowtie CLIENTE))$

piano di esecuzione logico:



piano fisico:



3. Presentare, se esiste, un esempio di piano di esecuzione fisico per l'interrogazione, supponendo di utilizzare l'operatore di join index nested loop per entrambi i join.

(uguale al piano fisico sopra)

4. Presentare, se esiste, un esempio di piano di esecuzione fisico per l'interrogazione, supponendo di utilizzare l'operatore di merge join per entrambi i join senza aggiungere ulteriori nodi rispetto al piano logico.

Il merge join funziona solo se entrambe le tabelle sono ordinate rispetto all'attributo di join. L'indice I_{pizza} (CodP) non esiste e l'indice I_{ordine} (TelC) non è clusterizzato.

5. Determinare se i piani presentati ai punti (3) e (4) ammettono elaborazione in pipelining.

SINTESI ELABORAZIONE DEGLI OPERATORI RELAZIONALI TRAMITE OPERATORI FISICI

_____= ammette elaborazione in pipeline (nel join su relazione outer)

Operatore fisico			
Operatore logico	Iterazione	Uso di indici	Partizionamento
ordinamento		x	merge sort a due fasi
σ	x	x	
π^D	x		
π (con eliminazione duplicati)		x	Ordinamento (uso di funzione hash, non presentato)
$\bowtie$	nested loop semplice e a blocchi	index join	merge join hash join

Il piano al punto 3 ammette elaborazioni in pipelining (la scansione sequenziale, l'index nested loop join e la proiezione senza eliminazione di duplicati ammettono elaborazione in pipelining).

6. I piani presentati ai punti (3) e (4) sono left deep?

Si, i piani sono left deep perchè la relazione inner di ogni operatore di join (a destra per convenzione nel piano) è sempre una relazione di base

ESERCIZIO 2

Si considerino le relazioni e lo schema fisico presentato nell'esercizio 1.
Considerare l'interrogazione

```
SELECT TelC
FROM ORDINE
WHERE Qta = 5 AND CodP = 2
```

1. Calcolare una stima del numero di tuple restituite dall'interrogazione.

Per rispondere a questa domanda bisogna usare questa tabella ↘

STIMA DELLA DIMENSIONE DEL RISULTATO DELLA SELEZIONE

- F(A=v) = 1/V(A,R)
- F(A IN (v1, v2,.....,vN))= N * F(A=v)
- F(A>v) = (Max(A,R) - v)/(Max(A,R)-Min(A,R))
- F(A<v) = (v-Min(A,R))/(Max(A,R)-Min(A,R))
- F(A1 = A2) = 1/ MAX (V(A1,R), V(A2,R))
- F(C1 AND C2) = F(C1) *F(C2)
- F(C1 OR C2) = F(C1) +F(C2) -F(C1) *F(C2)
- F(NOT C) = 1 - F(C)

64

Per prima cosa calcoliamo il fattore di selettività della condizione.

$F(Qta=5) = 1/V(Qta, ORDINE) = 1/15$

$F(CodP=2) = 1/V(CodP, ORDINE) = 1/30$

$F(Qta=5 \wedge CodP=2) = F(Qta=5) * F(CodP=2) = 1/15 * 1/30 = 1/450$

Per ottenere il numero stimato di tuple restituite, moltiplichiamo il numero di tuple della relazione ORDINE per il valore di selettività:

$T(ORDINE) * F(Qta=5 \wedge F(CodP=2) = 10.000/450 = 22,22$ quindi circa 23 tuple

2. Determinare i fattori booleani.

I fattori booleani sono Qta=5 e CodP=2

3. Supponendo di non avere ancora creato alcun indice e di poterne creare uno solo, su quale attributo converrebbe crearlo? Discutere l'impatto della clusterizzazione su questo indice. [la parte in blu è un chiarimento rispetto al testo discusso a lezione]

Converrebbe crearlo sulla condizione più selettiva, quindi quella che restituisce meno tuple, ossia: CodP=2.

Poiché CodP non è chiave primaria, creare l'indice clusterizzato potrebbe ridurre il numero di blocchi di organizzazione primaria acceduti.

TIPOLOGIA C

3. Esercizi tipologia C

ESERCIZIO 1

Data la seguente relazione:

MODELLO(codice, capacita', peso, anno_p, cod_produttore)

su cui sono definiti un indice clusterizzato su codice, un indice non clusterizzato su anno_p e un indice non clusterizzato su cod_produttore:

1. fornire un esempio di interrogazione che è velocizzata dall'avere l'indice su anno_p implementato come indice hash; → uguaglianze

```
SELECT *
FROM MODELLO
WHERE anno_p = 2004
```

2. fornire un esempio di interrogazione che è velocizzata dall'avere l'indice su anno_p implementato come indice ordinato; B+

```
SELECT *
FROM MODELLO
WHERE anno_p BETWEEN 2003 AND 2004
```

o anno_p > 2004

3. fornire un esempio di aggiornamento che è velocizzato dalla presenza di tali indici;

```
UPDATE MODELLO
SET peso = 40
WHERE anno_p > 2007 ∧ codice = 111
```

uso peso perché nel testo non è indicato come indice clusterizzato

Un aggiornamento viene velocizzato dalla presenza di indici se la clausola di qualificazione (WHERE) include almeno una condizione di selezione rispetto a un attributo indicizzato e la clausola SET non contiene attributi indicizzati.

4. fornire un esempio di aggiornamento che è rallentato dalla presenza di tali indici.

```
UPDATE MODELLO
SET cod_produttore = 'ABC'
WHERE peso > 70
```

Un aggiornamento viene rallentato dalla presenza di indici se la clausola SET riferisce almeno un attributo indicizzato e la clausola di qualificazione (WHERE) non contiene attributi indicizzati.

ESERCIZIO 2

Data la seguente relazione:

IMPIEGATO(imp#, nome, stip, età, dip#)

con un indice clusterizzato su imp# e un indice non clusterizzato su età:

1. fornire un esempio di interrogazione che è velocizzata dall'avere l'indice su imp# implementato come indice hash;
2. fornire un esempio di interrogazione che è velocizzata dall'avere l'indice su imp# implementato come indice ordinato;
3. fornire un esempio di interrogazione che è velocizzata dall'avere l'indice su età implementato come indice hash;
4. fornire un esempio di interrogazione che è velocizzata dall'avere l'indice su età implementato come indice ordinato;
5. la presenza degli indici velocizza il controllo del vincolo di chiave? in che casi?
6. fornire un esempio di operazione di aggiornamento rallentata dall'indice;
7. fornire un esempio di operazione di aggiornamento velocizzata dall'indice.

1)
```
SELECT *
FROM IMPIEGATO
WHERE IMP# = 12345
```

→ indice hash : uguaglianze

2) SELECT *
FROM IMPIEGATO
WHERE IMP# > 12345

3) SELECT *
FROM IMPIEGATO
WHERE ETÀ = 18

4) SELECT *
FROM IMPIEGATO
WHERE ETÀ BETWEEN 18 AND 20

5) La presenza di un indice su un attributo che è anche una chiave velocizza il controllo dell'unicità del valore. Nei casi in cui si inserisce un nuovo record o quando si aggiorna un record esistente, il database deve verificare che il valore per la chiave non sia già presente. Se esiste un indice sulla chiave, questa ricerca è molto più rapida (grazie all'indice, la ricerca del valore da inserire/modificare è molto più veloce rispetto a una scansione completa di tutta la tabella)

6) UPDATE IMPIEGATO
SET ETÀ = ETÀ + 1
WHERE STIP > 1000

7) UPDATE IMPIEGATO
SET STIP = 2000
WHERE ETÀ > 30