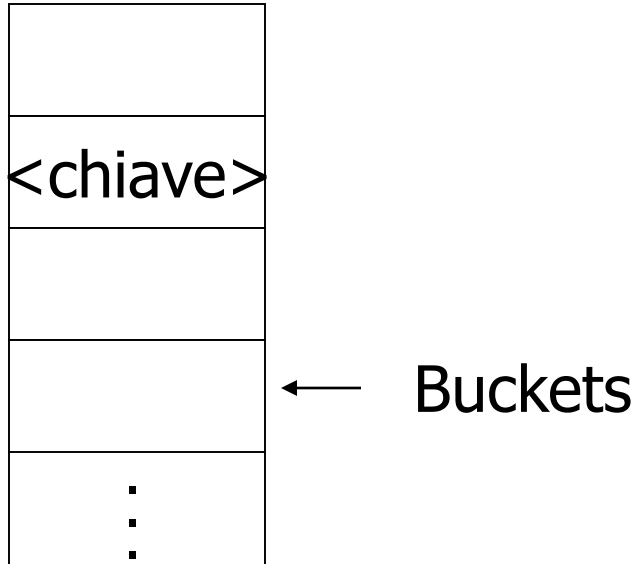
Hashing e indici multidimensionali

Leggere Cap 6 Riguzzi et al. Sistemi Informativi

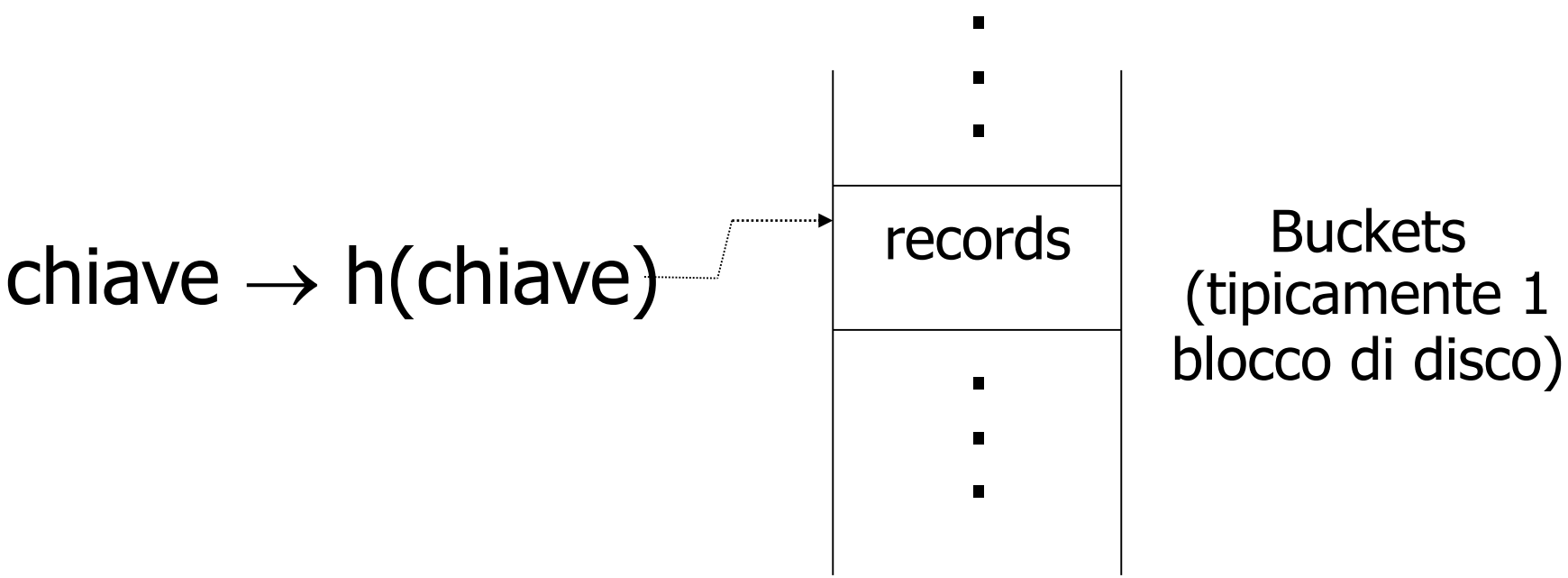
Lucidi derivati da quelli di Hector Garcia-Molina

Hashing

chiave $\rightarrow$ h(chiave)

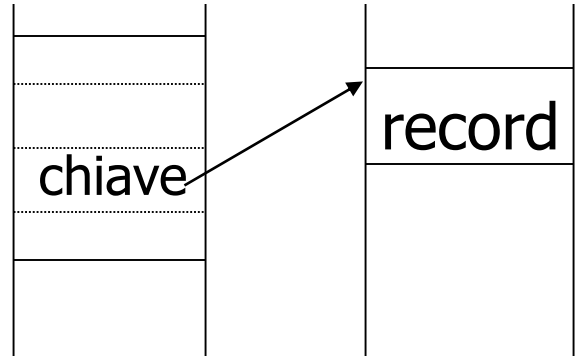


File hash


chiave → h(chiave)
records
Buckets
(tipicamente 1
blocco di disco)

Struttura hash su file non hash

(2) chiave $\rightarrow$ h(chiave)



Indice

La struttura hash diventa una struttura secondaria usata solo per l'accesso ai record

Esempio di funzione hash (1)

- Chiave: numero intero K
- Si abbiano *b* buckets
- h:
  - Calcola il resto della divisione intera di K per *b*
    - h(K)=K mod *b*

Esempio di funzione hash (2)

- Chiave = 'x_1 x_2 ... x_n' stringa di caratteri di *n* byte
- Si abbiano *b* buckets
- h:
  - Calcola somma=$x_1 + x_2 + \ldots\ldots x_n$
  - Calcola il resto della divisione intera di somma per *b*
    - h(K)=somma mod *b*

☒Vi sono molte funzioni hash, queste sono solo esempi

Buona funzione☞ Hash

Il numero di chiavi atteso in ogni bucket è lo stesso per ogni bucket

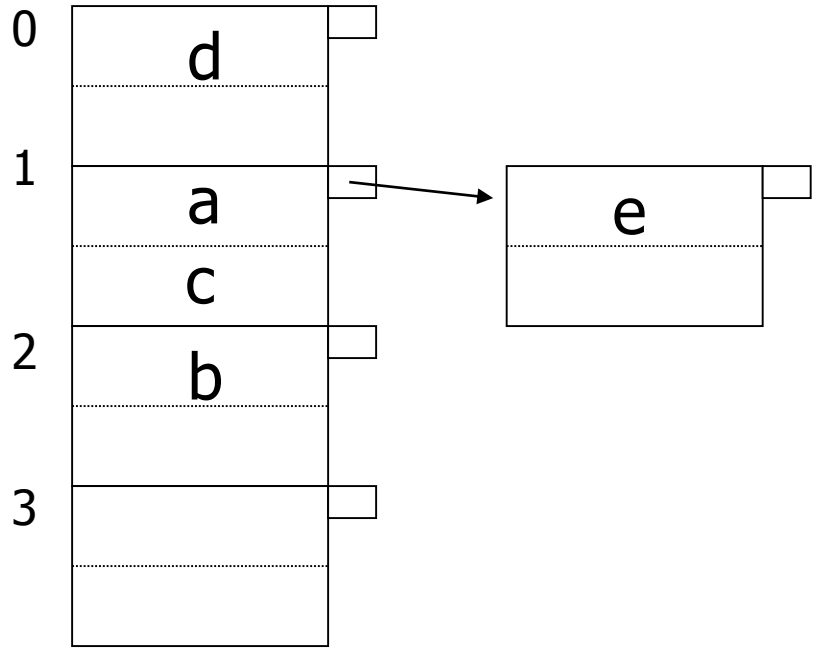
In un bucket:

- Si tengono le chiavi ordinate?

- Si', se il tempo di CPU e' critico e gli inserimenti e le cancellazioni non sono troppo frequenti

Inserimenti

- Se c'è posto nel bucket, non c'è problema, altrimenti si crea un bucket di oveflow
  - I bucket di overflow formano una catena



Cancellazioni

- Se si cancella un record da un bucket che aveva dei buckets di overflow, si possono spostare records da questi per ridurre la catena di oveflow.

Esempio 2 records/bucket, 4 buckets

Inserisci:

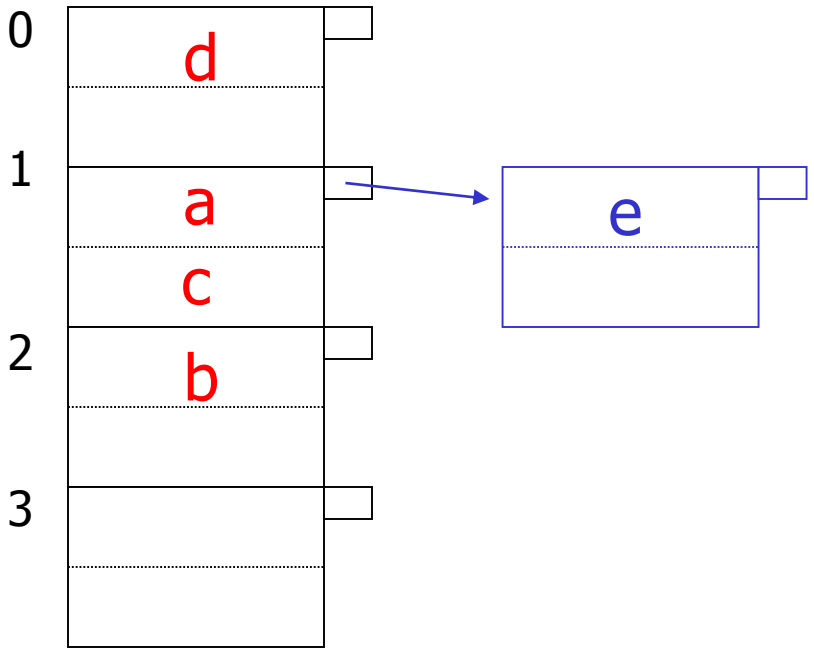
h(a) = 1

h(b) = 2

h(c) = 1

h(d) = 0

h(e) = 1



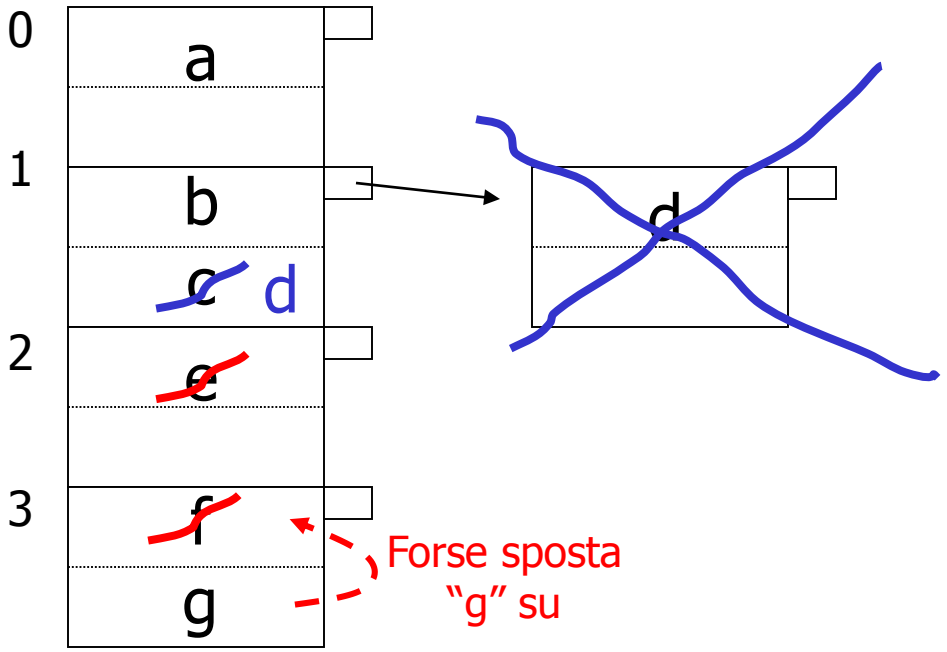
Esempio: cancellazione

Cancella:

e

f

c



Regola empirica:

- Cercare di tenere l'utilizzo tra il 50% e l'80%

$$\text{Utilizzo} = \frac{\#\text{ chiavi usate}}{\#\text{ massimo di chiavi nel file}}$$

- Se < 50%, si sta sprecando dello spazio
- Se > 80%, l'overflow e' significativo e dipende da quanto e' buona la funzione hash e dal numero di chiavi per bucket

Costo dell'accesso

- Accedere ad un record costa 1 se il file non ha catene di overflow
  - Più economico dell'indice
- Costa di più se ci sono catene di overflow
  - Costa 1 per ogni blocco della catena
- Svantaggi dei file hash rispetto ai B+-trees: non consentono query di range

Come gestire la crescita?

- Overflow e riorganizzazione
- Altre forme di hashing (hashing dinamico)

File hash, osservazioni

- È l'organizzazione più efficiente per l'accesso diretto basato su valori della chiave con condizioni di uguaglianza (accesso puntuale): costo medio di poco superiore all'unità (il caso peggiore è molto costoso ma talmente improbabile da poter essere ignorato)
- Non è efficiente per ricerche basate su intervalli
- I file hash "degenerano" se si riduce lo spazio sovrabbondante: funzionano solo con file la cui dimensione non varia molto nel tempo

Possiamo definire anche indici su piu' di un attributo, ad esempio:

CREATE INDEX foo ON R(A,B,C)

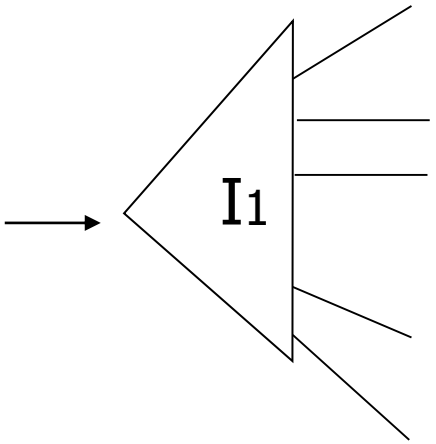
Indice Multichiave

Motivazione: trova i record per i quali

DEPT = “Toy” AND SAL > 50k

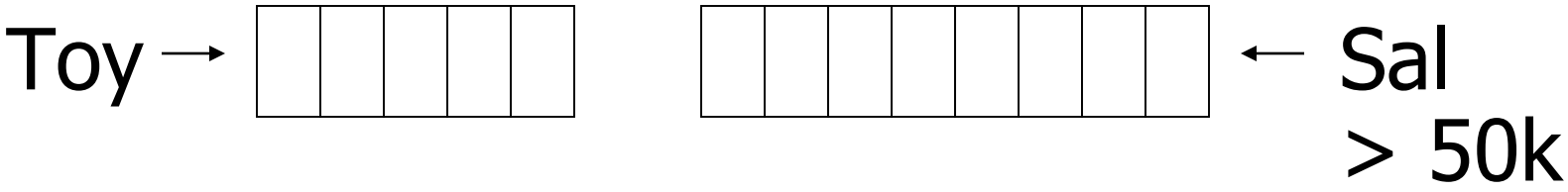
Strategia I:

- Usa un indice, ad esempio Dept.
- Ottieni tutti i record con Dept = “Toy” e verifica il loro salario



Strategia II:

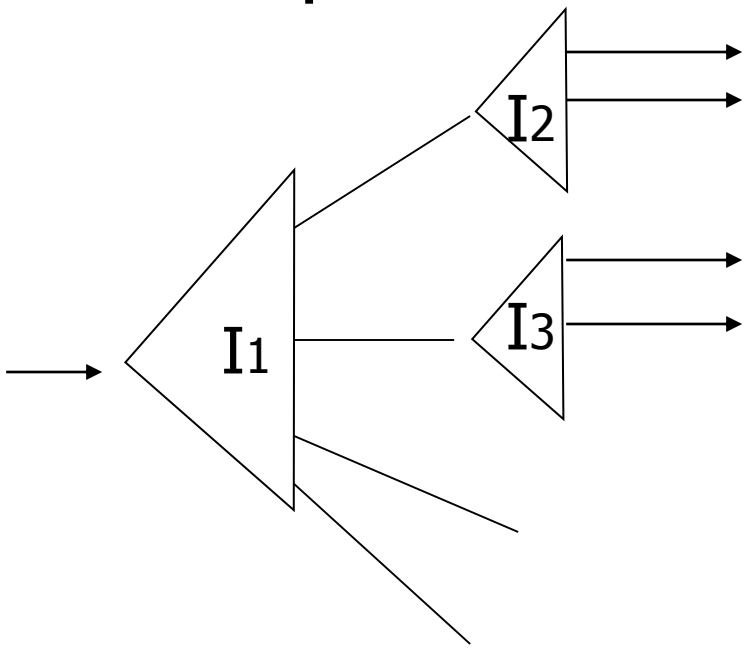
- Usa 2 indici; interseca i puntatori



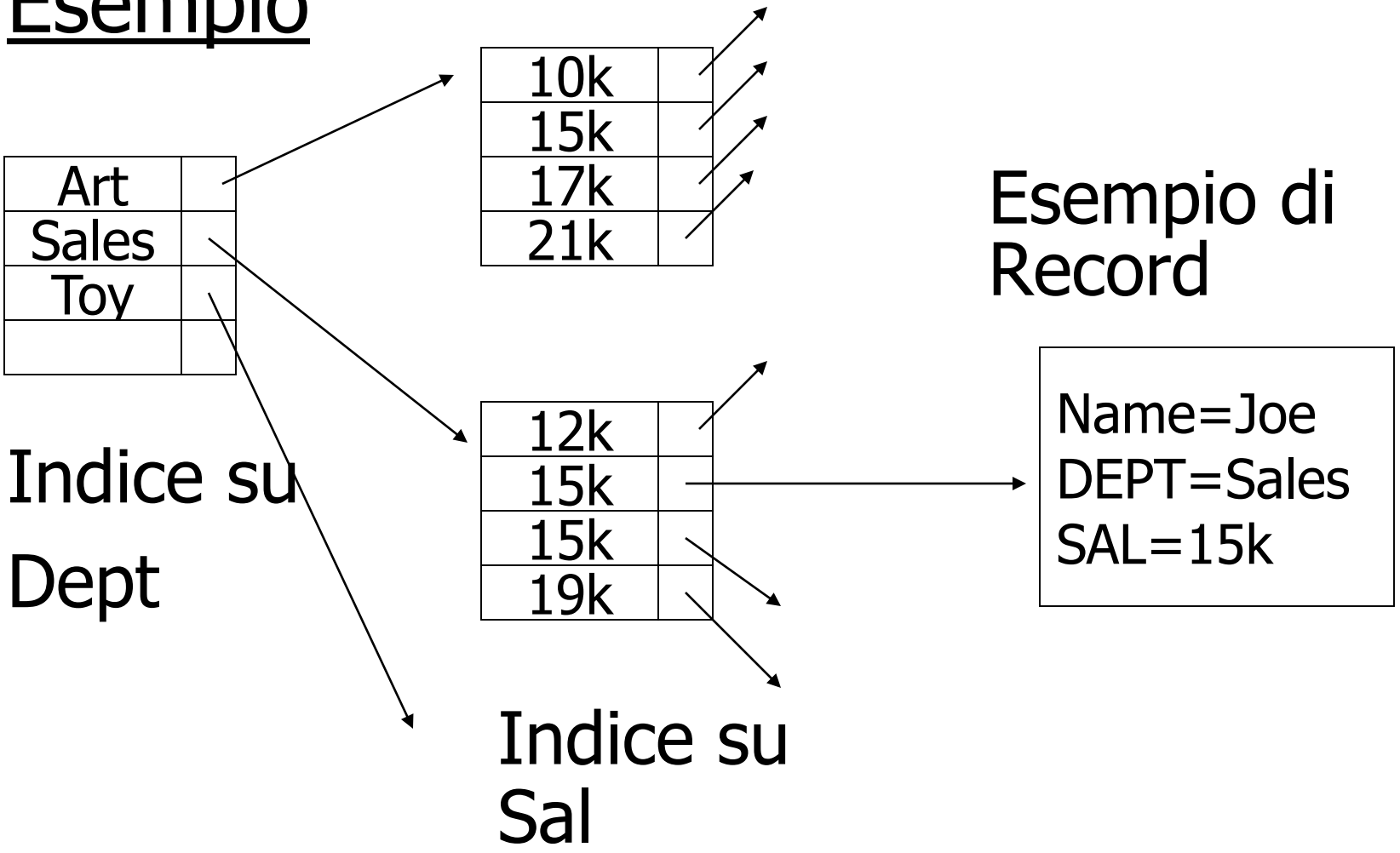
Strategia III:

- Indice su chiave multipla

Una possibilita':



Esempio

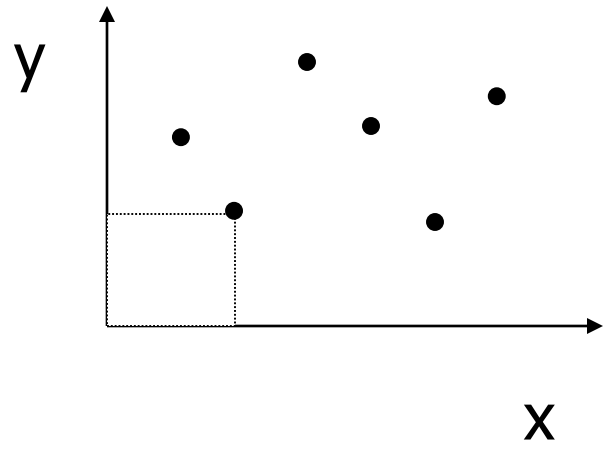

Art
Sales
Toy
10k
15k
17k
21k
12k
15k
15k
19k
Indice su Dept
Indice su Sal
Esempio di Record
Name=Joe
DEPT=Sales
SAL=15k

Per quali query quest'indice e' buono?

- ☐ Dept = "Sales" AND SAL=20k
- ☐ Dept = "Sales" AND SAL $\geq$ 20k
- ☐ Dept = "Sales"
- ☐ SAL = 20k

Dati multidimensionali:

- Dati geografici



DATI:

<X_1,Y_1, Attributi>

<X_2,Y_2, Attributi>

⋮

- Dati qualunque, ogni attributo e' una dimensione

Esempi di query su dati multidimensionali

- Match parziale: dati valori per una o piu' dimensioni trovare tutti i punti con quei valori
- Query di range: dati range per una o piu' dimensione trovare tutti i punti in quei range. Se sono rappresentate forme, trovare le forme che sono parzialmente o completamente incluse nei range
- Nearest neighbor query: dato un punto, trovare il punto piu' vicino

Esempi di query su dati multidimensionali

- Query "dove sono": dato un punto vogliamo sapere in quale forma, se ce n'è una, è collocato

Esempi

- Sia Points(x,y) una tabella che contiene tutti i punti
- Query di match parziale, tutti i punti con x=5:

  SELECT * FROM Points WHERE x=5
- Query di range, tutti in punti con x in [5,10]:

  SELECT * FROM Points WHERE x>=5 AND x<=10

Esempi

- Nearest neighbor query: punto più vicino a (10.0,20.0)

```
SELECT * FROM Points p WHERE NOT EXISTS(
    SELECT * FROM Points q WHERE
        (q.x-10.0)*(q.x-10.0)+(q.y-20.0)*(q.y-20.0)<
        (p.x-10.0)*(p.x-10.0)+(p.y-20.0)*(p.y-20.0)
);
```

Esempi

- Query "dove sono": trovare i rettangoli che contengono (10.0,20.0)
- Sia abbia una tabella

  Rectangles(id,xll,yll,xur,yur)
- Query

```
SELECT id FROM Rectangles WHERE
    xll<=10.0 AND yll<=20.0 AND
    xur>=10.0 AND yur>=20.0;
```

Trova il punto più vicino usando indici

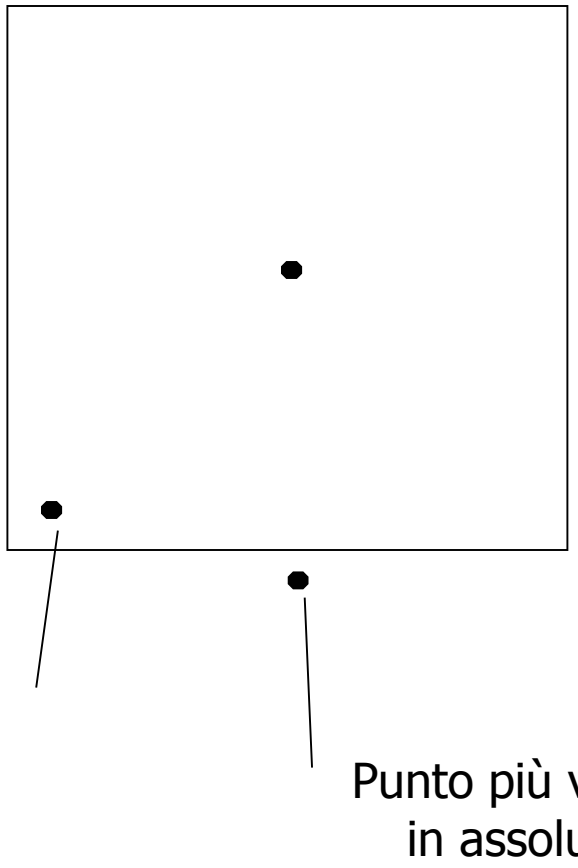
- Dato un punto (X,Y), si costruiscono due range: [X-d,X+d] e [Y-d,Y+d] con d scelto dall'utente
- Si esegue una query per restituire tutti i punti nel range usando i B+-trees su x e su y
- Si prende il punto più vicino tra quelli restituiti

Trova il punto più vicino usando indici

- Problemi:
  1. Non ci sono punti nel range
  2. Il punto più vicino nel range potrebbe non essere il più vicino complessivamente
- Problema 1.: si prova con un d più grande

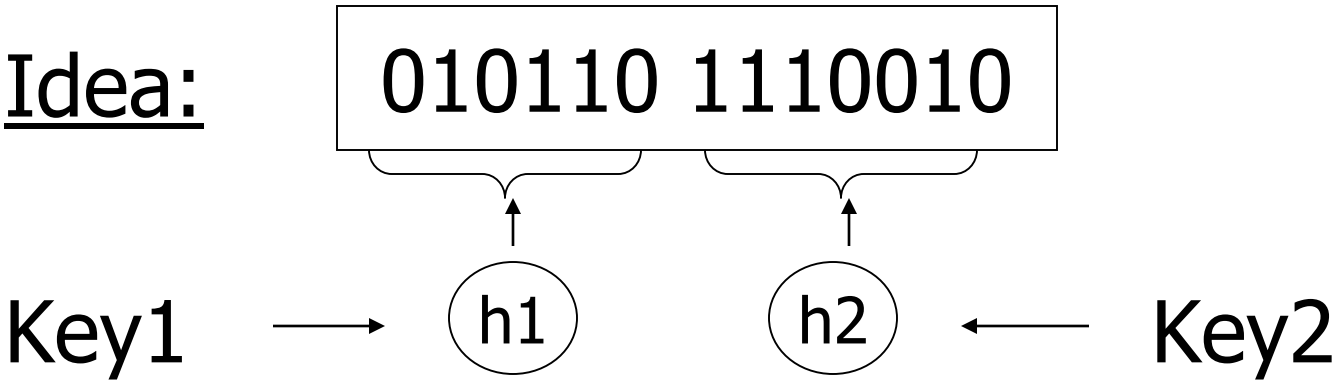
Problema 2.

- Se d’ è la distanza del punto più vicino nel range e d’>d, si riprova la query con d’
- Se c’è un punto con distanza < d’, con la seconda query lo si trova



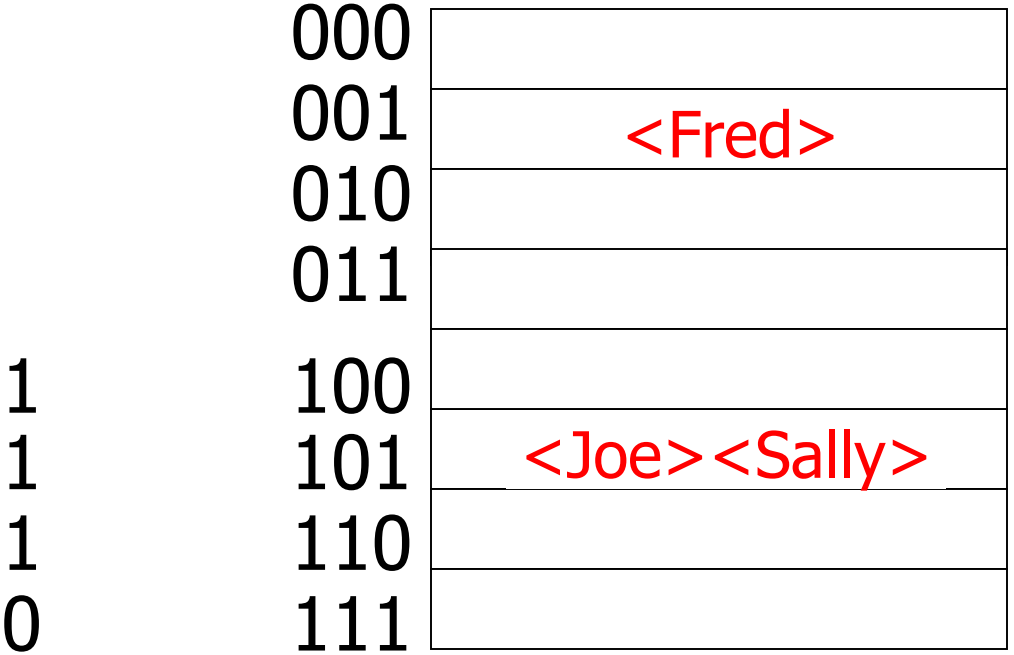
Funzione hash partizionate

<u>Idea:</u>



Es:

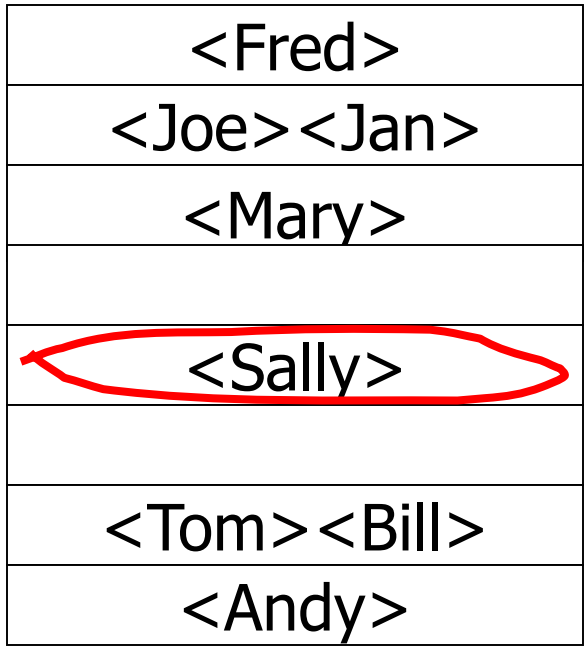
h1(toy) =0
h1(sales) =1
h1(art) =1
.
.
h2(10k) =01
h2(20k) =11
h2(30k) =01
h2(40k) =00
.
.



Inserisci

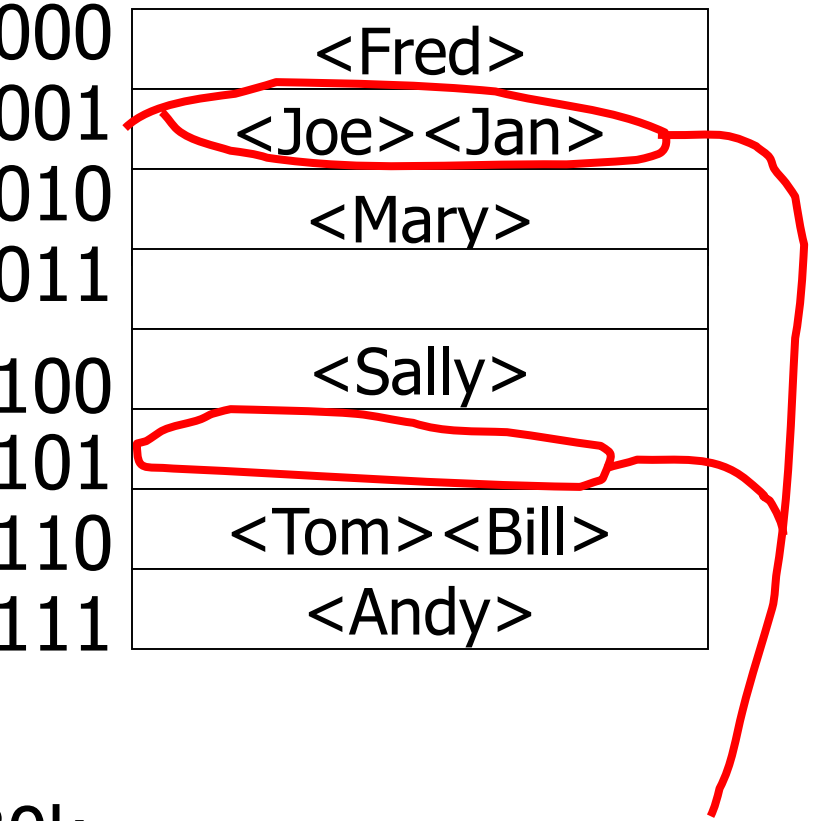
<Fred,toy,10k>,<Joe,sales,10k>
<Sally,art,30k>

h1(toy) =0
h1(sales) =1
h1(art) =1
.
.
.
h2(10k) =01
h2(20k) =11
h2(30k) =01
h2(40k) =00
.
.
.



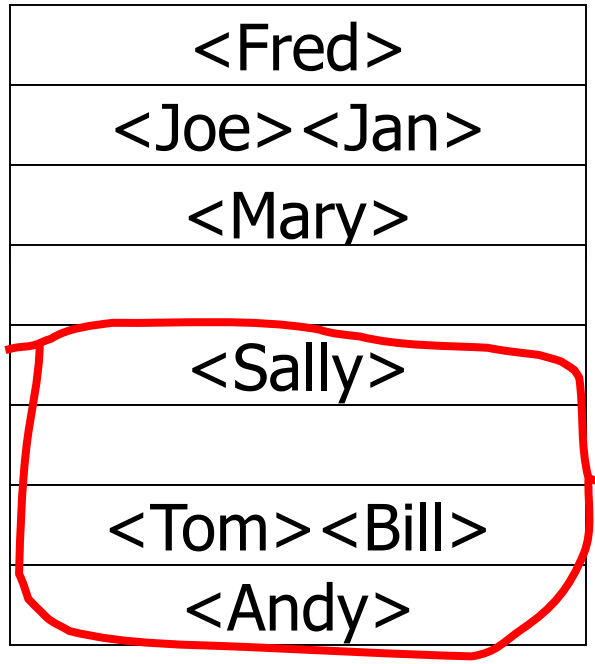
- Trova Imp. Con Dept. = Sales $\wedge$ Sal=40k

h1(toy) =0
h1(sales) =1
h1(art) =1
.
.
.
h2(10k) =01
h2(20k) =11
h2(30k) =01
h2(40k) =00
.
.
.



- Trova Imp. con Sal=30k

h1(toy) =0
h1(sales) =1
h1(art) =1
.
.
.
h2(10k) =01
h2(20k) =11
h2(30k) =01
h2(40k) =00
.
.
.



- Find Emp. with Dept. = Sales

Guarda qui