

# CS 525: Advanced Database Organization

## 05: Hashing and More

Boris Glavic



Slides: adapted from a [course](#) taught by [Hector Garcia-Molina](#), Stanford InfoLab

CS 525



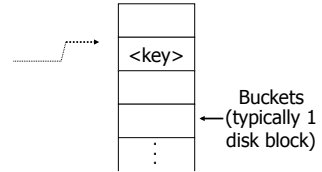
Notes 5 - Hashing

1

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### Hashing

key  $\rightarrow$  h(key)



CS 525



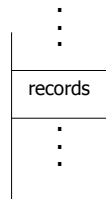
Notes 5 - Hashing

2

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### Two alternatives

(1) key  $\rightarrow$  h(key)



CS 525



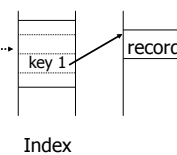
Notes 5 - Hashing

3

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### Two alternatives

(2) key  $\rightarrow$  h(key)



CS 525



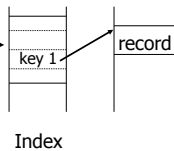
Notes 5 - Hashing

4

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### Two alternatives

(2) key  $\rightarrow$  h(key)



- Alt (2) for "secondary" search key

CS 525



Notes 5 - Hashing

5

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### Example hash function

- Key = 'x<sub>1</sub> x<sub>2</sub> ... x<sub>n</sub>'  $n$  byte character string
- Have  $b$  buckets
- h: add  $x_1 + x_2 + \dots + x_n$   
– compute sum modulo  $b$

CS 525



Notes 5 - Hashing

6

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

- This may not be best function ...
- Read Knuth Vol. 3 if you really need to select a good function.

CS 525



Notes 5 - Hashing

7

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

- This may not be best function ...
- Read Knuth Vol. 3 if you really need to select a good function.

Good hash function: ↗ Expected number of keys/bucket is the same for all buckets

CS 525



Notes 5 - Hashing

8

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### Within a bucket:

- Do we keep keys sorted?
- Yes, if CPU time critical & Inserts/Deletes not too frequent

CS 525

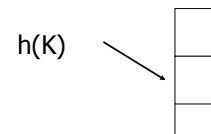


Notes 5 - Hashing

9

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

Next: example to illustrate inserts, overflows, deletes



CS 525



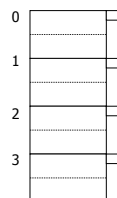
Notes 5 - Hashing

10

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### EXAMPLE 2 records/bucket

INSERT:  
 $h(a) = 1$   
 $h(b) = 2$   
 $h(c) = 1$   
 $h(d) = 0$



CS 525



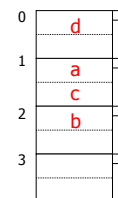
Notes 5 - Hashing

11

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### EXAMPLE 2 records/bucket

INSERT:  
 $h(a) = 1$   
 $h(b) = 2$   
 $h(c) = 1$   
 $h(d) = 0$   
 $h(e) = 1$



CS 525



Notes 5 - Hashing

12

IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

**EXAMPLE** 2 records/bucket

INSERT:

$h(a) = 1$   
 $h(b) = 2$   
 $h(c) = 1$   
 $h(d) = 0$   
 $h(e) = 1$

CS 525 Notes 5 - Hashing 13 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

**EXAMPLE:** deletion

Delete:

e  
f

CS 525 Notes 5 - Hashing 14 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

**EXAMPLE:** deletion

Delete:

e  
f  
c

CS 525 Notes 5 - Hashing 15 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

**EXAMPLE:** deletion

Delete:

e  
f  
c

CS 525 Notes 5 - Hashing 16 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

**Rule of thumb:**

- Try to keep space utilization between 50% and 80%

$$\text{Utilization} = \frac{\# \text{ keys used}}{\text{total } \# \text{ keys that fit}}$$

CS 525 Notes 5 - Hashing 17 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

**Rule of thumb:**

- Try to keep space utilization between 50% and 80%


$$\text{Utilization} = \frac{\# \text{ keys used}}{\text{total } \# \text{ keys that fit}}$$

- If < 50%, wasting space
- If > 80%, overflows significant
  - depends on how good hash function is & on # keys/bucket

CS 525 Notes 5 - Hashing 18 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY


### How do we cope with growth?

- Overflows and reorganizations
- Dynamic hashing

CS 525  Notes 5 - Hashing 19 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

### How do we cope with growth?

- Overflows and reorganizations
- Dynamic hashing
  - Extensible
  - Linear

CS 525  Notes 5 - Hashing 20 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY


### Extensible hashing: two ideas

(a) Use  $i$  of  $b$  bits output by hash function

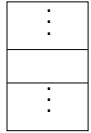
$h(k) \rightarrow$  00110101


←  $b$  →

use  $i \rightarrow$  grows over time....

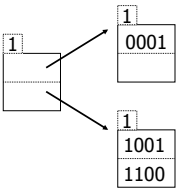
CS 525  Notes 5 - Hashing 21 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

(b) Use directory


$h(k)[i]$   to bucket

CS 525  Notes 5 - Hashing 22 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

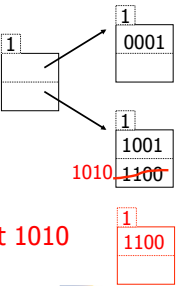
### Example: $h(k)$ is 4 bits; 2 keys/bucket

$i =$  1 


**Insert 1010**

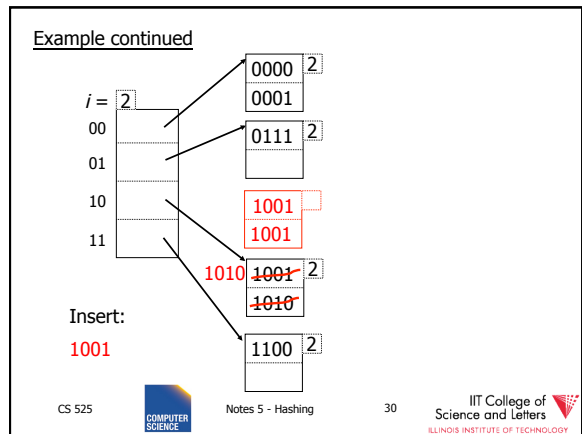
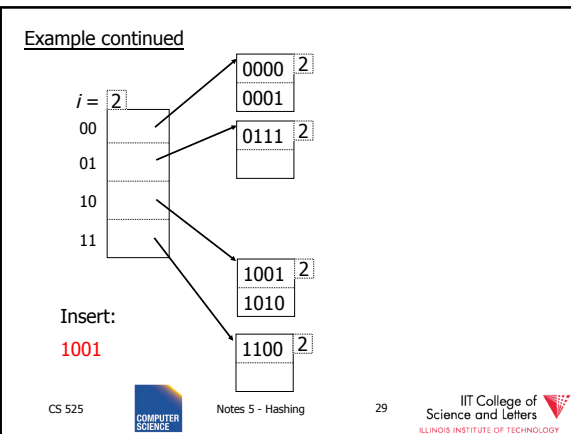
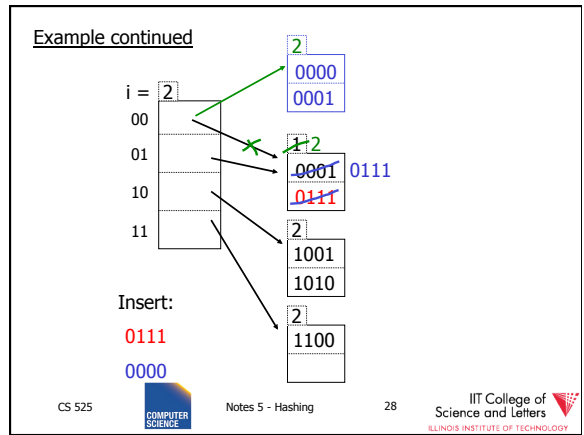
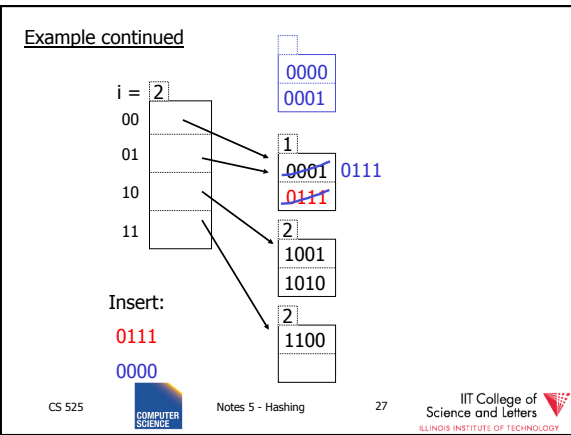
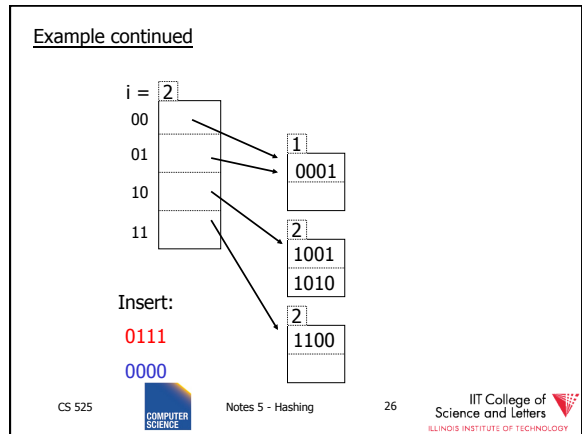
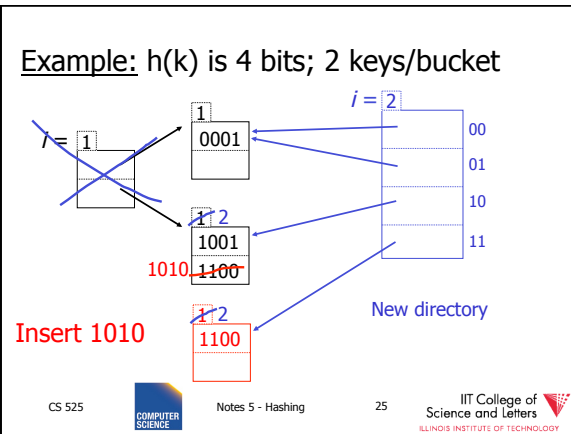
CS 525  Notes 5 - Hashing 23 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

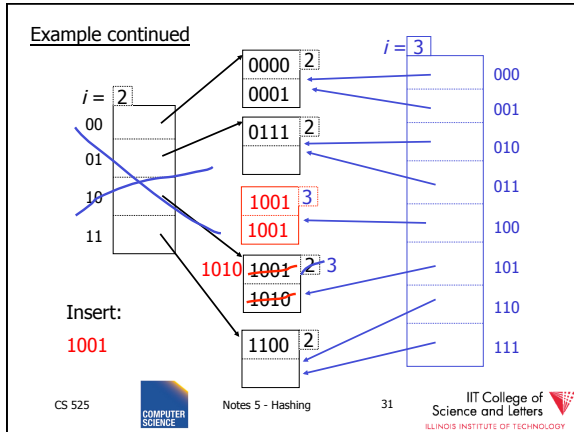
### Example: $h(k)$ is 4 bits; 2 keys/bucket

$i =$  1 

**Insert 1010**

CS 525  Notes 5 - Hashing 24 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY





**Extensible hashing: deletion**

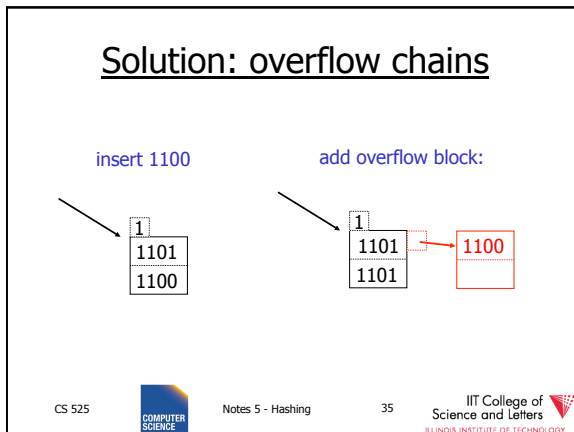
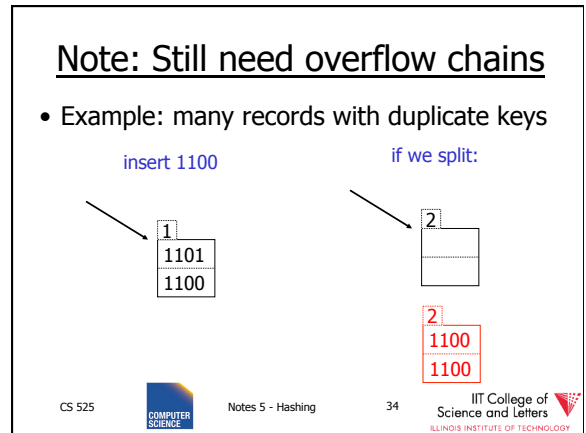
- No merging of blocks
- Merge blocks and cut directory if possible (Reverse insert procedure)

CS 525 | COMPUTER SCIENCE | Notes 5 - Hashing | 32 | IIT College of Science and Letters | ILLINOIS INSTITUTE OF TECHNOLOGY

**Deletion example:**

- Run thru insert example in reverse!

CS 525 | COMPUTER SCIENCE | Notes 5 - Hashing | 33 | IIT College of Science and Letters | ILLINOIS INSTITUTE OF TECHNOLOGY




**Summary**     Extensible hashing

- ⊕ Can handle growing files
  - with less wasted space
  - with no full reorganizations

CS 525 | COMPUTER SCIENCE | Notes 5 - Hashing | 36 | IIT College of Science and Letters | ILLINOIS INSTITUTE OF TECHNOLOGY

**Summary** Extensible hashing

- ⊕ Can handle growing files
  - with less wasted space
  - with no full reorganizations
- ⊖ Indirection  
(Not bad if directory in memory)
- ⊖ Directory doubles in size  
(Now it fits, now it does not)

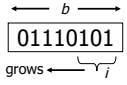
CS 525  Notes 5 - Hashing 37 IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY


**Linear hashing**

- Another dynamic hashing scheme

Two ideas:

(a) Use  $i$  low order bits of hash



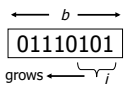
CS 525  Notes 5 - Hashing 38 IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

**Linear hashing**


- Another dynamic hashing scheme


Two ideas:

(a) Use  $i$  low order bits of hash



(b) File grows linearly



CS 525  Notes 5 - Hashing 39 IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY


**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

0000	0101		
1010	1111		

00      01      10      11

$m = 01$  (max used block)

← Future growth buckets

CS 525  Notes 5 - Hashing 40 IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY


**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

0000	0101		
1010	1111		

00      01      10      11

$m = 01$  (max used block)

**Rule** If  $h(k)[i] \leq m$ , then  
look at bucket  $h(k)[i]$   
else, look at bucket  $h(k)[i] - 2^{i-1}$

CS 525  Notes 5 - Hashing 41 IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket


- insert 0101

0000	0101		
1010	1111		

00      01      10      11

$m = 01$  (max used block)

**Rule** If  $h(k)[i] \leq m$ , then  
look at bucket  $h(k)[i]$   
else, look at bucket  $h(k)[i] - 2^{i-1}$

CS 525  Notes 5 - Hashing 42 IIT College of Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

• insert 0101  
• can have overflow chains!

0000	0101		
1010	1111		

00      01      10      11

$m = 01$  (max used block)

**Rule** If  $h(k)[i] \leq m$ , then  
look at bucket  $h(k)[i]$   
else, look at bucket  $h(k)[i] - 2^{i-1}$

CS 525    COMPUTER SCIENCE    Notes 5 - Hashing    43    IIT College of Science and Letters    ILLINOIS INSTITUTE OF TECHNOLOGY

**Note**

- In textbook,  $n$  is used instead of  $m$
- $n = m + 1$

0000	0101		
1010	1111		

00      01      10      11

$m = 01$  (max used block)

CS 525    COMPUTER SCIENCE    Notes 5 - Hashing    44    IIT College of Science and Letters    ILLINOIS INSTITUTE OF TECHNOLOGY

**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

0000	0101		
1010	1111		

00      01      10      11

$m = 01$  (max used block)

CS 525    COMPUTER SCIENCE    Notes 5 - Hashing    45    IIT College of Science and Letters    ILLINOIS INSTITUTE OF TECHNOLOGY

**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

0000	0101	1010	
<del>1010</del>	1111		

00      01      10      11

$m = 10$  (max used block)

CS 525    COMPUTER SCIENCE    Notes 5 - Hashing    46    IIT College of Science and Letters    ILLINOIS INSTITUTE OF TECHNOLOGY

**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

• insert 0101

0000	0101	1010	
<del>1010</del>	1111		

00      01      10      11

$m = 10$  (max used block)

CS 525    COMPUTER SCIENCE    Notes 5 - Hashing    47    IIT College of Science and Letters    ILLINOIS INSTITUTE OF TECHNOLOGY

**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

• insert 0101

0000	0101	1010	
<del>1010</del>	1111		

00      01      10      11

$m = 11$  (max used block)

CS 525    COMPUTER SCIENCE    Notes 5 - Hashing    48    IIT College of Science and Letters    ILLINOIS INSTITUTE OF TECHNOLOGY



**Example**  $b=4$  bits,  $i=2$ , 2 keys/bucket

• insert 0101

Future growth buckets

$m = 01$  (max used block)

CS 525    IIT College of Science and Letters

**Example Continued: How to grow beyond this?**

$i = 2$

0000	0101	1010	1111
	0101		

00      01      10      11      ...

$m = 11$  (max used block)

CS 525    IIT College of Science and Letters

**Example Continued: How to grow beyond this?**

$i = 2, 3$

0000	0101	1010	1111		
	0101				

000      001      010      011      ...

100      101      110      111

$m = 11$  (max used block)

CS 525    IIT College of Science and Letters

**Example Continued: How to grow beyond this?**

$i = 2, 3$

0000	0101	1010	1111		
	0101				

000      001      010      011      100      ...

~~100~~      101      110      111

$m = 11$  (max used block)

100

CS 525    IIT College of Science and Letters

**Example Continued: How to grow beyond this?**

$i = 2, 3$

0000	0101	1010	1111		0101
	0101				0101

000      001      010      011      100      101      ...

~~100~~      ~~101~~      110      111

$m = 11$  (max used block)

100

101

CS 525    IIT College of Science and Letters



• When do we expand file?

• Keep track of:  $\frac{\# \text{ used slots}}{\text{total \# of slots}} = U$

CS 525    IIT College of Science and Letters



☛ When do we expand file?

- Keep track of:  $\frac{\text{\# used slots}}{\text{total \# of slots}} = U$
- If  $U > \text{threshold}$  then increase  $m$  (and maybe  $i$ )

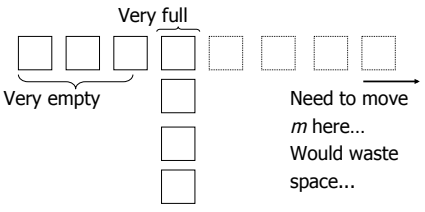
CS 525  Notes 5 - Hashing 55 

**Summary** Linear Hashing



- ⊕ Can handle growing files
  - with less wasted space
  - with no full reorganizations
- ⊕ No indirection like extensible hashing
- Can still have overflow chains

CS 525  Notes 5 - Hashing 56 

Example: BAD CASE





Very empty      Very full      Need to move  $m$  here...  
Would waste space...

CS 525  Notes 5 - Hashing 57 

**Summary**



Hashing

- How it works
- Dynamic hashing
  - Extensible
  - Linear

CS 525  Notes 5 - Hashing 58 

Next:



- Indexing vs Hashing
- Index definition in SQL
- Multiple key access

CS 525  Notes 5 - Hashing 59 

**Indexing vs Hashing**

- Hashing good for probes given key
 

e.g.,  
SELECT ...  
FROM R  
WHERE R.A = 5

CS 525  Notes 5 - Hashing 60 

## Indexing vs Hashing

- INDEXING (Including B Trees) good for Range Searches:  
e.g.,  
SELECT  
FROM R  
WHERE R.A > 5

CS 525



Notes 5 - Hashing

61

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

## Index definition in SQL

- CREATE index name on rel (attr)
- CREATE unique index name on rel (attr)  
└── defines candidate key
- DROP INDEX name

CS 525



Notes 5 - Hashing

62

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

**Note** CANNOT SPECIFY TYPE OF INDEX  
(e.g. B-tree, Hashing, ...)  
OR PARAMETERS  
(e.g. Load Factor, Size of Hash,...)

... at least in SQL...

CS 525



Notes 5 - Hashing

63

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

**Note** ATTRIBUTE LIST ⇒ MULTIKEY INDEX  
(next)

e.g., CREATE INDEX foo ON R(A,B,C)

CS 525



Notes 5 - Hashing

64

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

## Multi-key Index

Motivation: Find records where  
DEPT = "Toy" AND SAL > 50k

CS 525



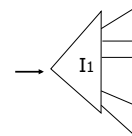
Notes 5 - Hashing

65

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

## Strategy I:

- Use one index, say Dept.
- Get all Dept = "Toy" records  
and check their salary



CS 525



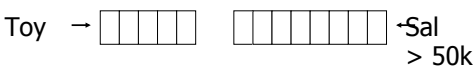
Notes 5 - Hashing


66

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

### Strategy II:

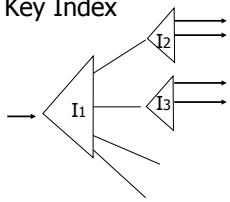
- Use 2 Indexes; Manipulate Pointers


Toy → 

CS 525  Notes 5 - Hashing 67 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

### Strategy III:

- Multiple Key Index

One idea: 

CS 525  Notes 5 - Hashing 68 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

### Example

Art
Sales
Toy

Dept Index


10k
15k
17k
21k

Salary Index

12k
15k
15k
19k


Example Record

Name=Joe
DEPT=Sales
SAL=15k

CS 525  Notes 5 - Hashing 69 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

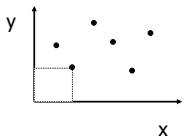
### For which queries is this index good?

- Find RECs Dept = "Sales"  $\wedge$  SAL=20k
- Find RECs Dept = "Sales"  $\wedge$  SAL  $\geq$  20k
- Find RECs Dept = "Sales"
- Find RECs SAL = 20k

CS 525  Notes 5 - Hashing 70 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY


### Interesting application:

- Geographic Data




DATA:

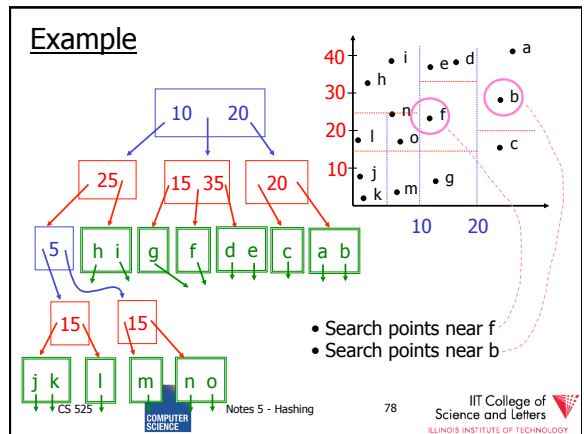
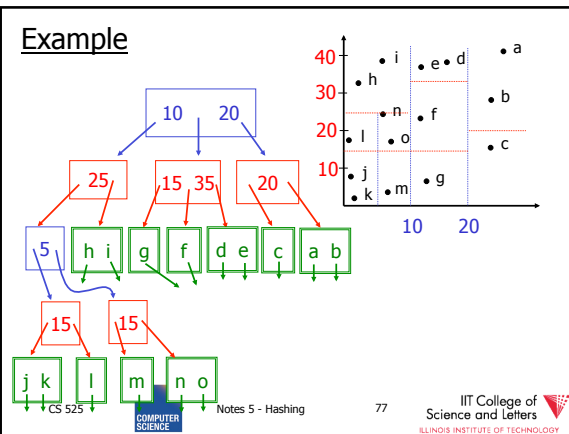
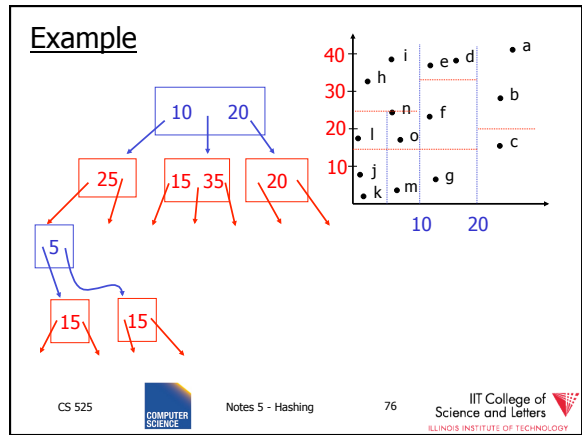
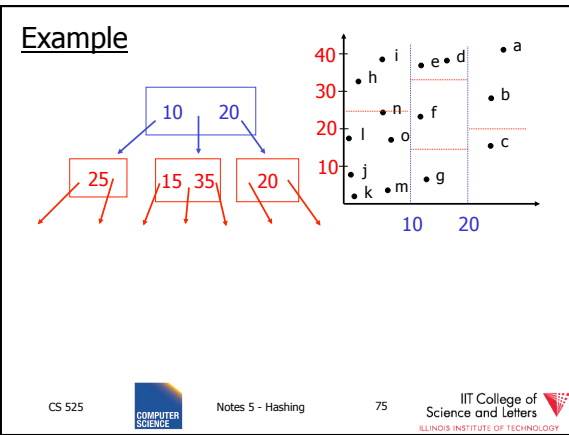
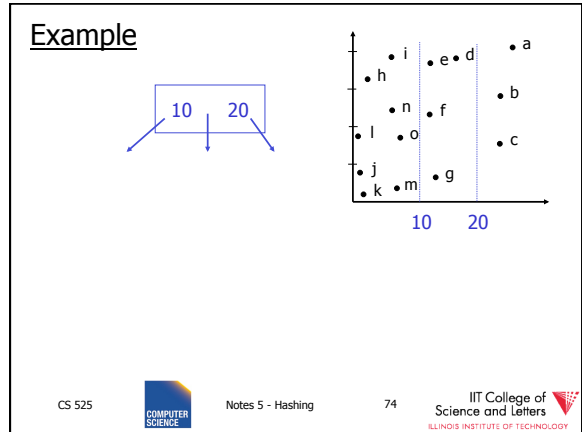
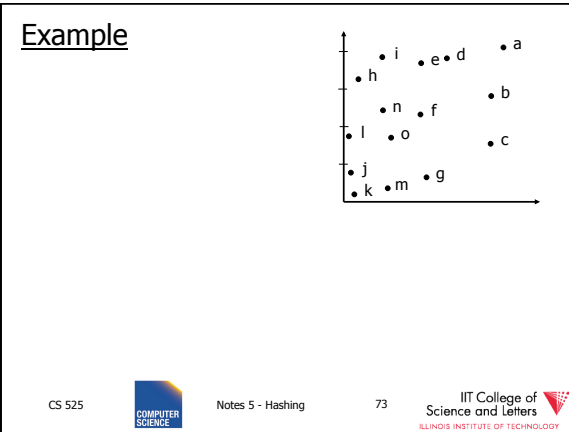
- <X1,Y1, Attributes>
- <X2,Y2, Attributes>
- ⋮

CS 525  Notes 5 - Hashing 71 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY

### Queries:

- What city is at  $\langle X_i, Y_i \rangle$ ?
- What is within 5 miles from  $\langle X_i, Y_i \rangle$ ?
- Which is closest point to  $\langle X_i, Y_i \rangle$ ?

CS 525  Notes 5 - Hashing 72 IIT College of Science and Letters ILLINOIS INSTITUTE OF TECHNOLOGY



## Queries

- Find points with  $Y_i > 20$
- Find points with  $X_i < 5$
- Find points “close” to  $i = \langle 12, 38 \rangle$
- Find points “close” to  $b = \langle 7, 24 \rangle$

CS 525



Notes 5 - Hashing

79

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY

## Next

- Even more index structures ☺

CS 525



Notes 5 - Hashing

80

IIT College of  
Science and Letters  
ILLINOIS INSTITUTE OF TECHNOLOGY