Author Information

- Name: 鍾博丞
- Student ID: 408410120
- E-mail: my072814638@csie.io

Part A

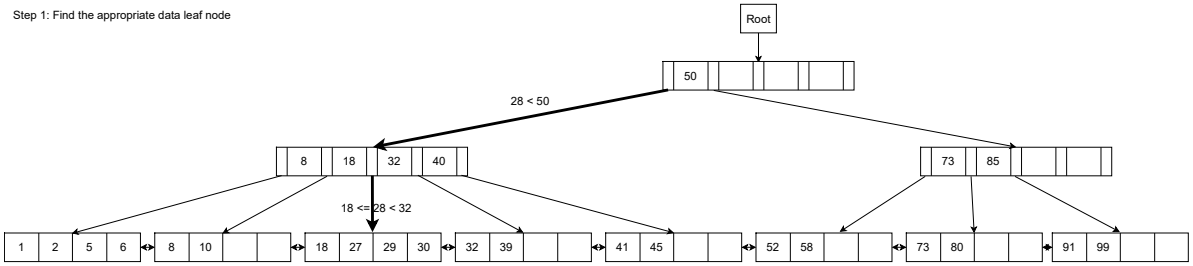
We **omit the doubly-linked list** between all nearby data leaf nodes. They indeed actually exist.

A1

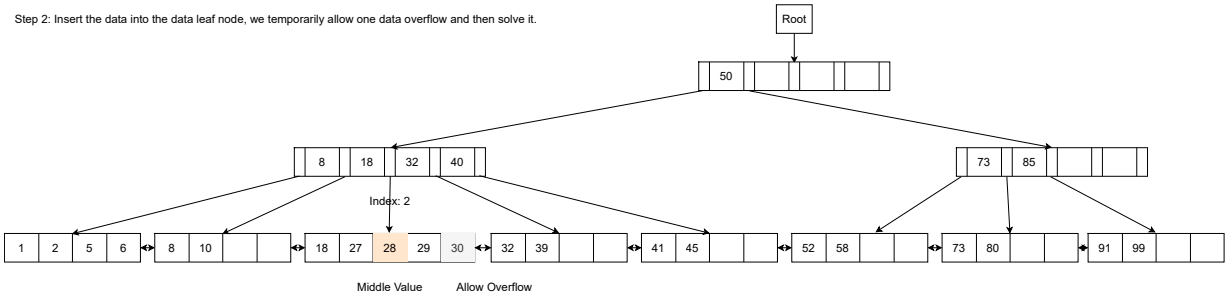
This is an SVG picture so please be free to zoom in.

Goal: Insert 28

Step 1: Find the appropriate data leaf node



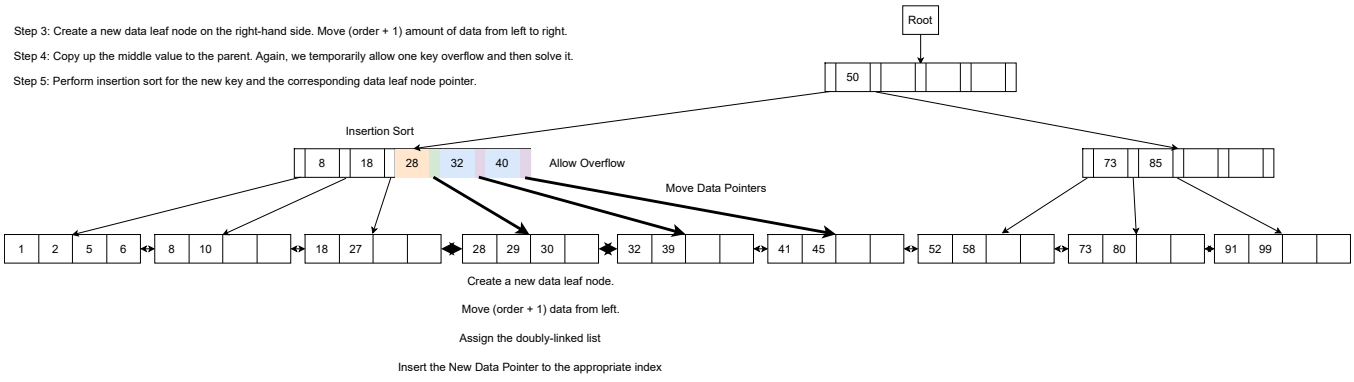
Step 2: Insert the data into the data leaf node, we temporarily allow one data overflow and then solve it.



Step 3: Create a new data leaf node on the right-hand side. Move (order + 1) amount of data from left to right.

Step 4: Copy up the middle value to the parent. Again, we temporarily allow one key overflow and then solve it.

Step 5: Perform insertion sort for the new key and the corresponding data leaf node pointer.



Create a new data leaf node.

Move (order + 1) data from left.

Assign the doubly-linked list

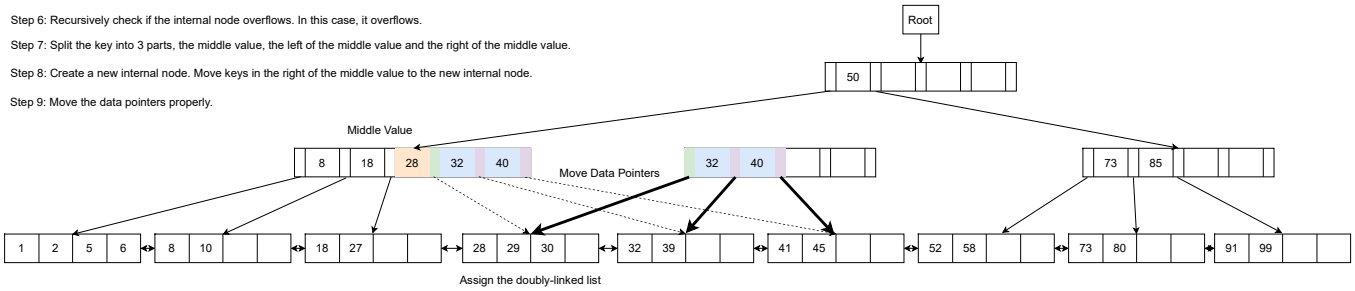
Insert the New Data Pointer to the appropriate index

Step 6: Recursively check if the internal node overflows. In this case, it overflows.

Step 7: Split the key into 3 parts, the middle value, the left of the middle value and the right of the middle value.

Step 8: Create a new internal node. Move keys in the right of the middle value to the new internal node.

Step 9: Move the data pointers properly.

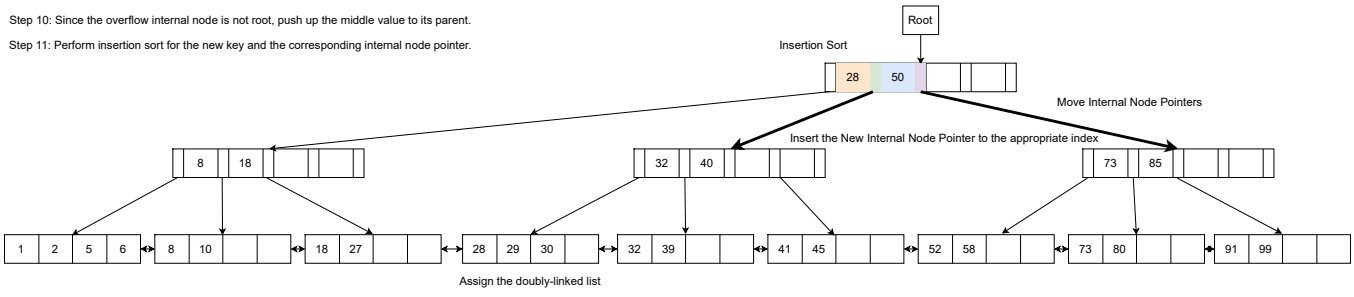


Assign the doubly-linked list

Insert the New Data Pointer to the appropriate index

Step 10: Since the overflow internal node is not root, push up the middle value to its parent.

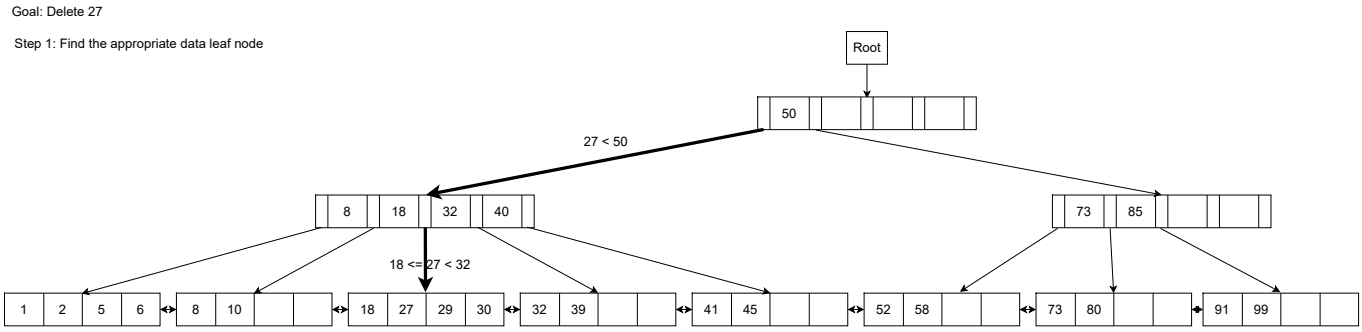
Step 11: Perform insertion sort for the new key and the corresponding internal node pointer.

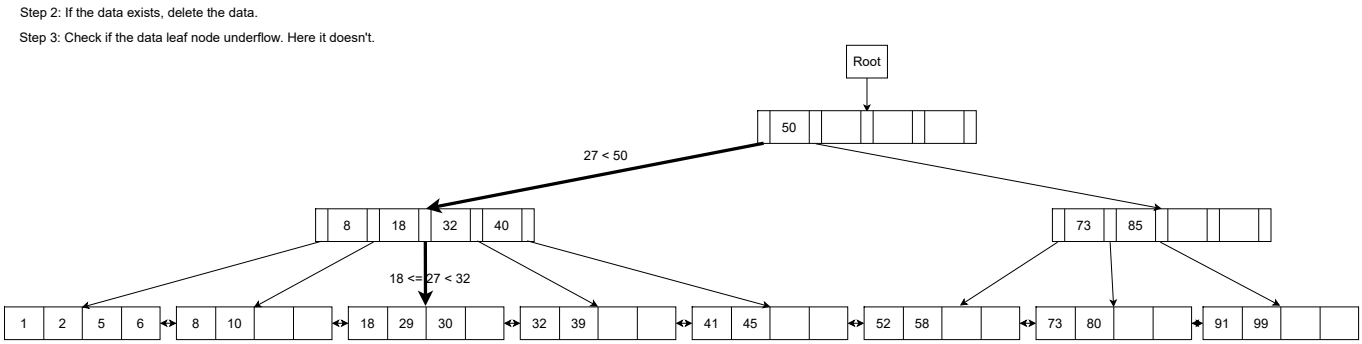


Assign the doubly-linked list

Insert the New Data Pointer to the appropriate index

A2





Part B

B1

Find the names of students who got grade 10 in some courses.

$$\Pi_{\text{sname}}(\sigma_{\text{grade}=10}(\text{Enrolled}) \bowtie_{\text{Students.sid}=\text{Enrolled.sid}} \text{Students})$$

B2

Find the ages of students who take some courses with 3 credits.

$$\begin{aligned}\Pi_{\text{age}}[&\text{Courses} \bowtie_{\text{Courses.cid}=\text{Enrolled.cid}} (\text{Enrolled}) \\ -&\sigma_{\text{credits}\neq 3}(\text{Courses}) \bowtie_{\text{Courses.cid}=\text{Enrolled.cid}} (\text{Enrolled})] \\ &\bowtie_{\text{Enrolled.sid}=\text{Students.sid}} (\text{Students})\end{aligned}$$

B3

Find the names of students who take a course named 'Calculus'.

$$\begin{aligned}\Pi_{\text{sname}}[&\sigma_{\text{cname}=\text{'Calculus'}}(\text{Courses}) \bowtie_{\text{Courses.cid}=\text{Enrolled.cid}} (\text{Enrolled}) \\ &\bowtie_{\text{Enrolled.sid}=\text{Students.sid}} (\text{Students})]\end{aligned}$$

B4

Find the names of students who obtained grades of at least 8 in some course that has less than 4 credits.

$$\Pi_{\text{sname}}[\sigma_{\text{grade} >= 8}\text{Enrolled} \bowtie \sigma_{\text{credits} < 4}\text{Courses} \bowtie \text{Students}]$$

B5

Find the names of students who obtained only grades of 10 (which implies that they took at least one course).

$$\Pi_{\text{sname}}\{\Pi_{\text{sid, sname}}[\text{Enrolled} \bowtie_{\text{Students.sid}=\text{Enrolled.sid}} \text{Students}] \\ -\Pi_{\text{sid, sname}}[\sigma_{\text{grade}\neq 10}(\text{Enrolled}) \bowtie_{\text{Students.sid}=\text{Enrolled.sid}} (\text{Students})]\}$$

B6

Find the names of students who took a course with three credits or who obtained grade 10 in some course.

$$\Pi_{\text{sname}}\{\{\Pi_{\text{sid}}[\sigma_{\text{grade}=10}(\text{Enrolled})] \\ \cup\Pi_{\text{sid}}[\text{Enrolled} \bowtie_{\text{Enrolled.cid}=\text{Courses.cid}\wedge\text{Courses.credits}=3} \text{Courses}]\} \\ \bowtie_{\text{sid}=\text{Students.sid}} \text{Students}\}$$

B7

Find the ages of students who attend 'Calculus' but never took any 4-credit course (assume there is a course 'Calculus' with 3 credits).

$$\Pi_{\text{age}}[(\sigma_{\text{cname}=\text{'Calculus'}}\text{Courses} \cap \sigma_{\text{credits}\neq 4}\text{Courses}) \bowtie_{\text{Enrolled.cid}=\text{Courses.cid}} \text{Enrolled} \\ \bowtie_{\text{Enrolled.sid}=\text{Students.sid}} \text{Students}]$$

B8

Find the names of students who have the lowest grades.

$$\Pi_{\text{sname}}[\sigma_{\text{Enrolled.grade}=\min(\text{Enrolled.grade})}(\text{Enrolled} \bowtie \text{Students})]$$

B9

Find the names of students who are enrolled in a **single** course.

$$\Pi_{\text{sname}}[\sigma_{\text{count(cid)}=1}(\text{Enrolled} \bowtie \text{Students}) \bowtie \text{Courses}]$$

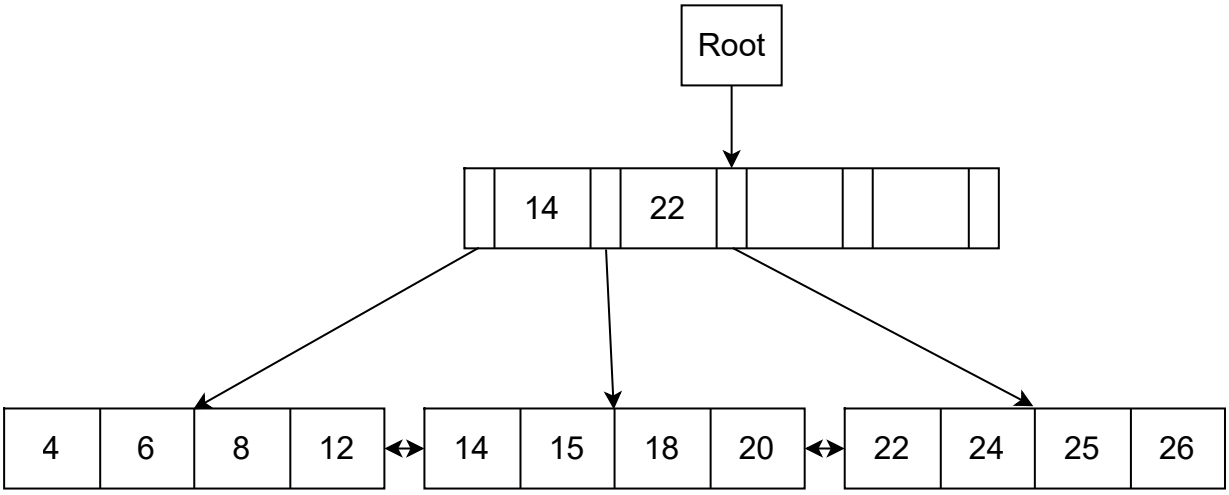
B10

Find the grades of students who are enrolled in the course(s) with the highest number of credits.

$$\Pi_{\text{grade}}[\text{Enrolled} \bowtie (\Pi_{\text{cid, credits}}\text{Courses}) \div \max_{\text{credits}}\text{Courses}] \bowtie \text{Students}$$

Part C

C1



C2

Every time when data leaf node overflows, the key in its parent will increase by 1 amount. In this case, the internal node already has 2 keys. If there happen 3 times of data leaf overflow, then the keys in this internal node will increase to 5 keys and thus overflow.

We observe that all these 3 data leaf nodes are full now. Therefore, we insert data into each data leaf node.

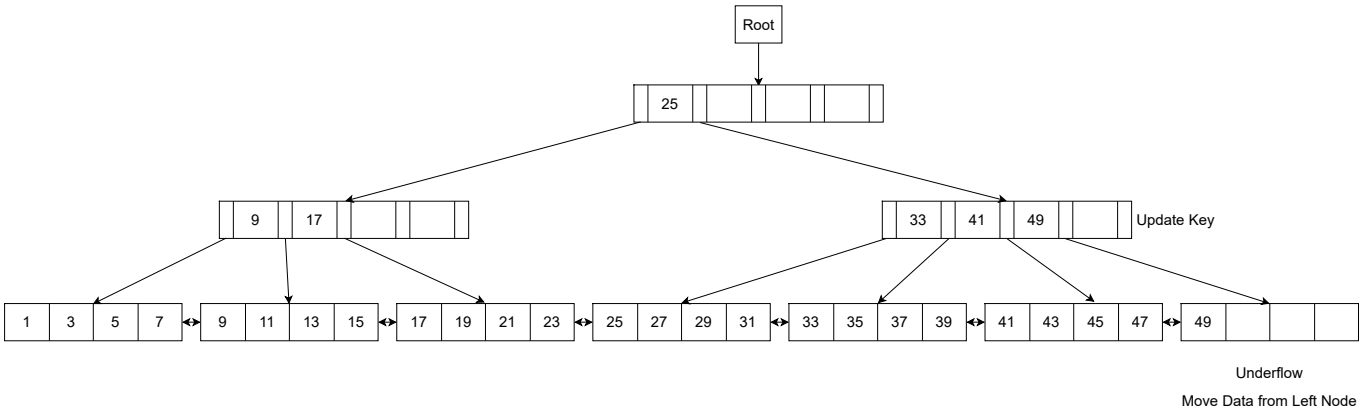
Insert the first data $x_1 < 14$, the second data $14 \leq x_2 < 22$, and the third data $x_3 \geq 22$.

For example, insert 7, 16, 23.

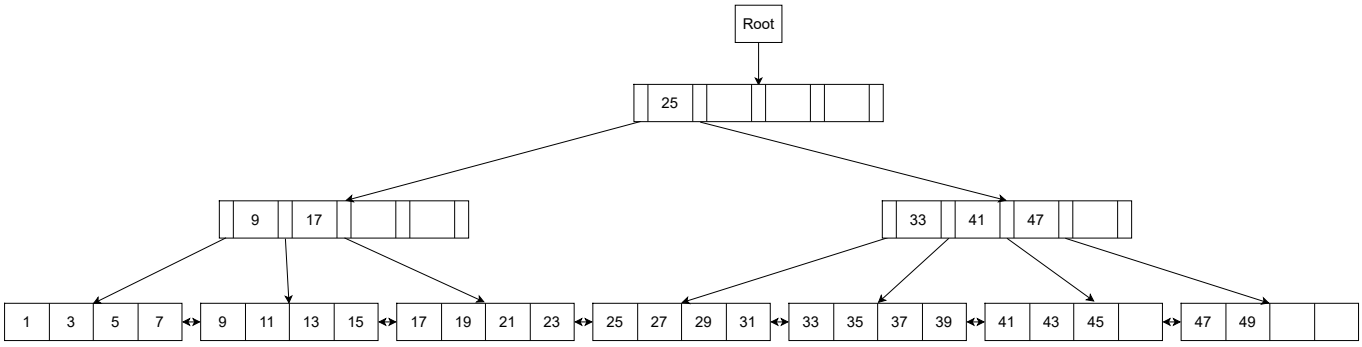
Part D

D1

The result of bulk loading is shown below.



The last data leaf node is underflow, we have to deal with this issue.



D2

The height of this tree is 3.

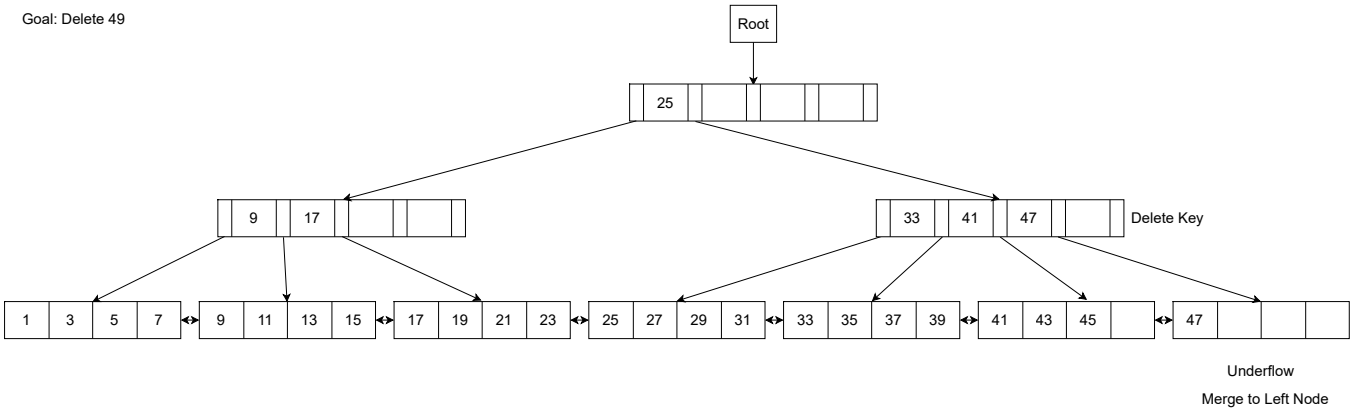
D3

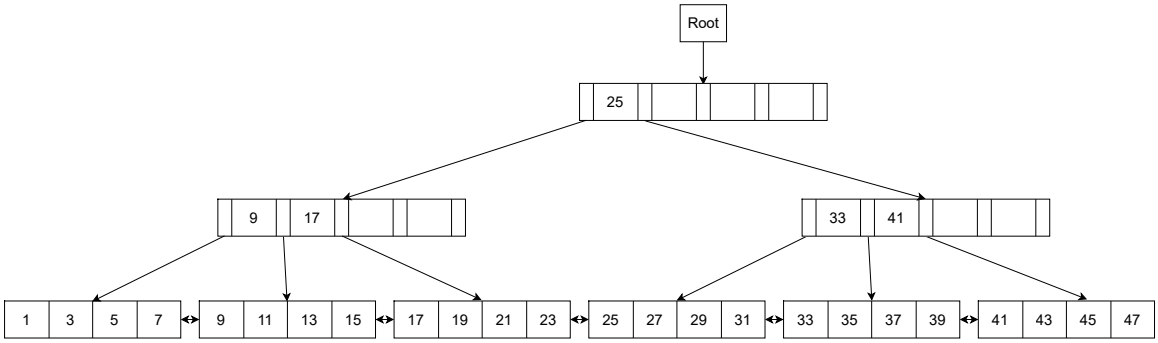
We need to delete **5** keys to make the tree decrease its height.

Delete 49, 47, 45, 43, 41

Try to let underflow occurs as many times as possible when deletion.

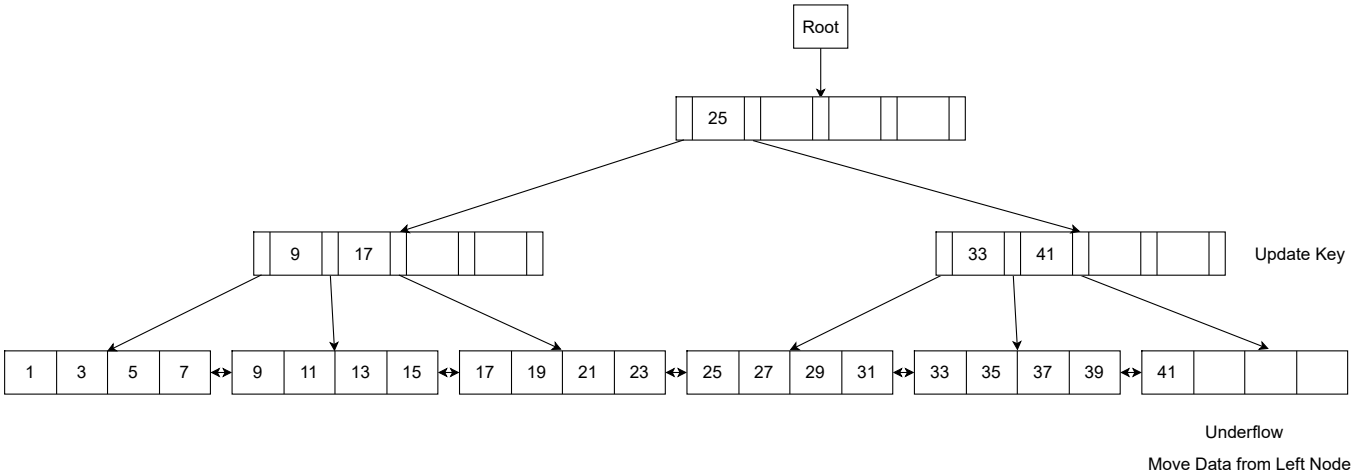
1. Delete 49

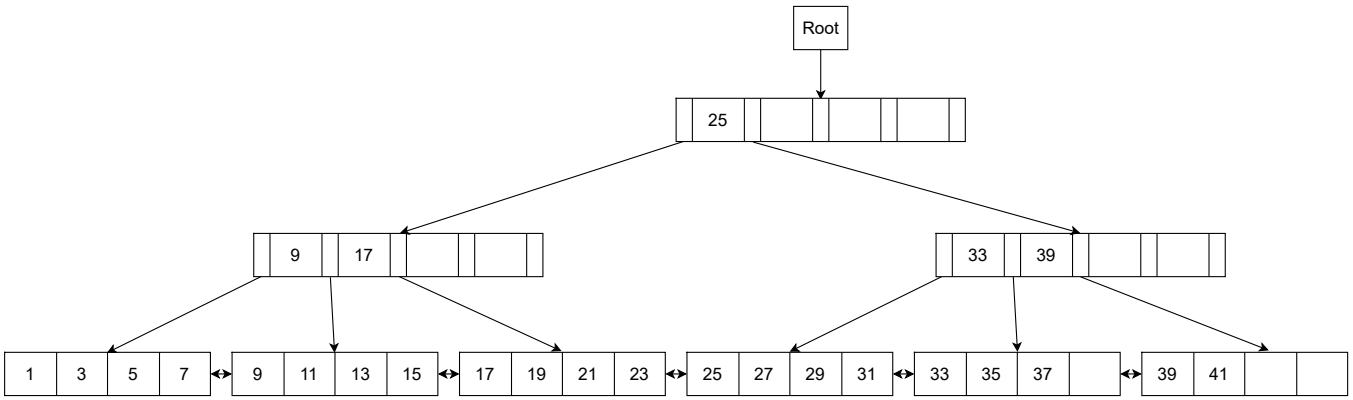




2. Delete 47, 45, 43

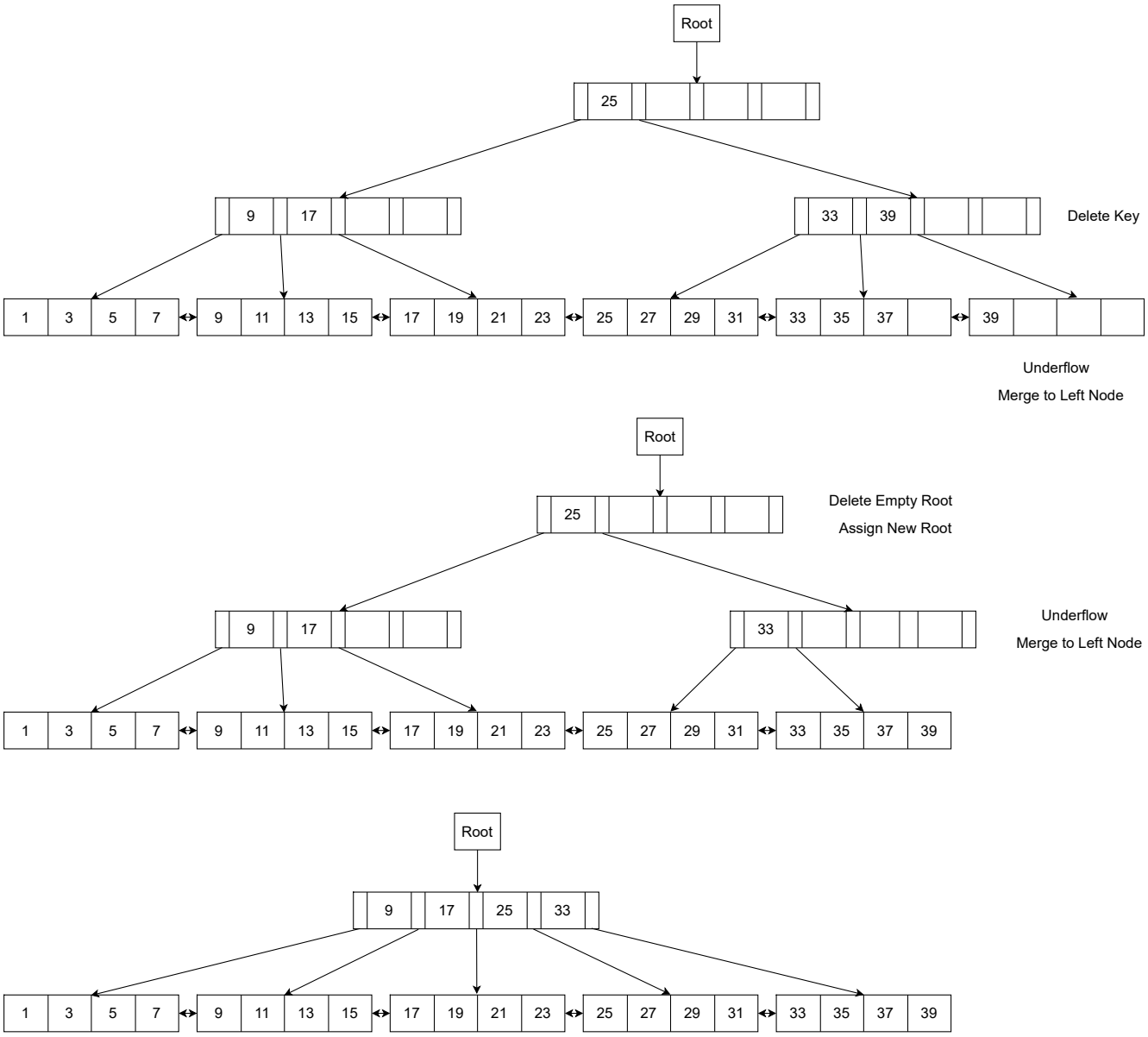
Goal: Delete 47, 45, 43





3. Delete 41

Goal: Delete 41

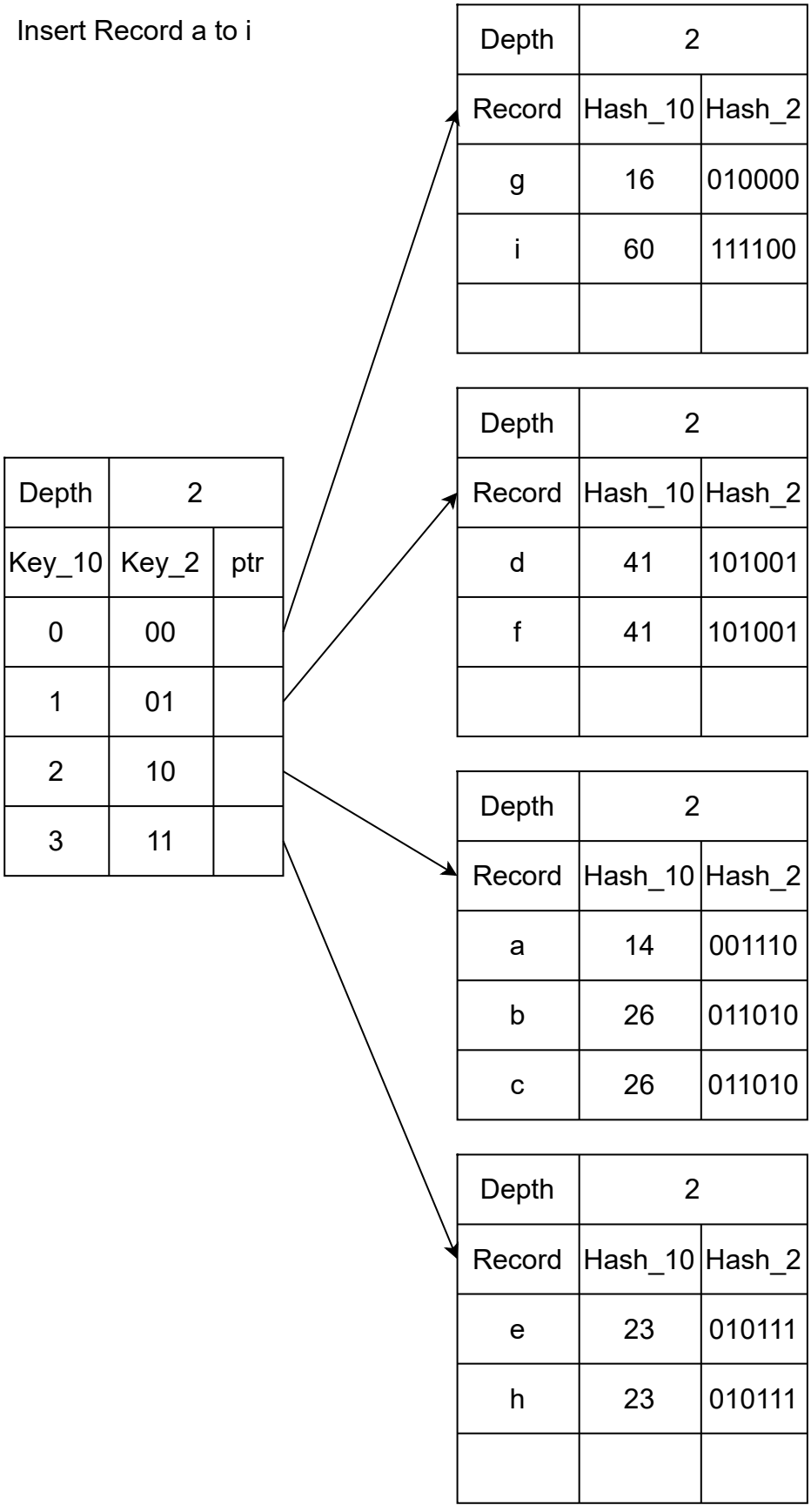


Part E

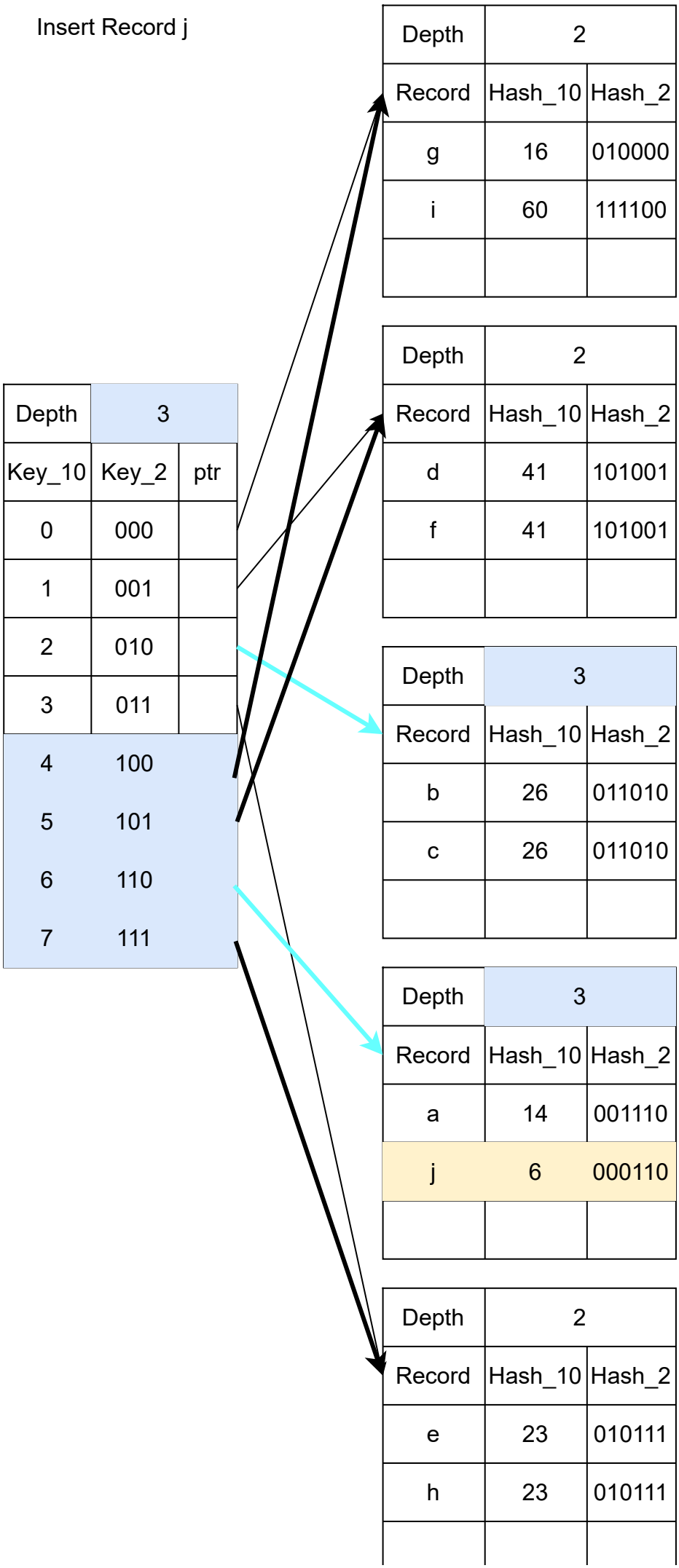
E1

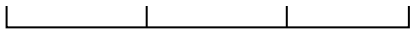
First, we insert records a to i.

Insert Record a to i



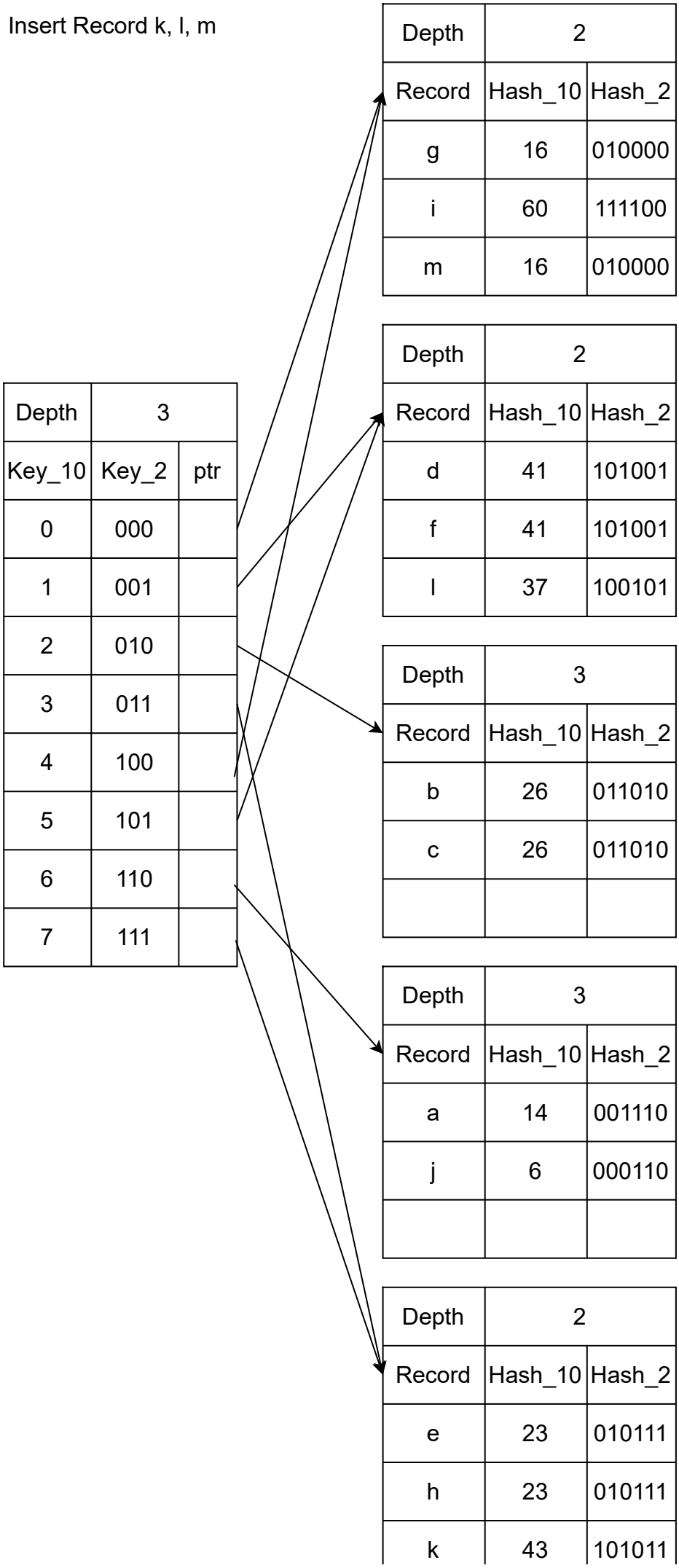
We then insert record j, and this will cause overflow.


Insert Record j
Depth 3
Key_10 Key_2 ptr
0 000
1 001
2 010
3 011
4 100
5 101
6 110
7 111
Depth 2
Record Hash_10 Hash_2
g 16 010000
i 60 111100
Depth 2
Record Hash_10 Hash_2
d 41 101001
f 41 101001
Depth 3
Record Hash_10 Hash_2
b 26 011010
c 26 011010
Depth 3
Record Hash_10 Hash_2
a 14 001110
j 6 000110
Depth 2
Record Hash_10 Hash_2
e 23 010111
h 23 010111



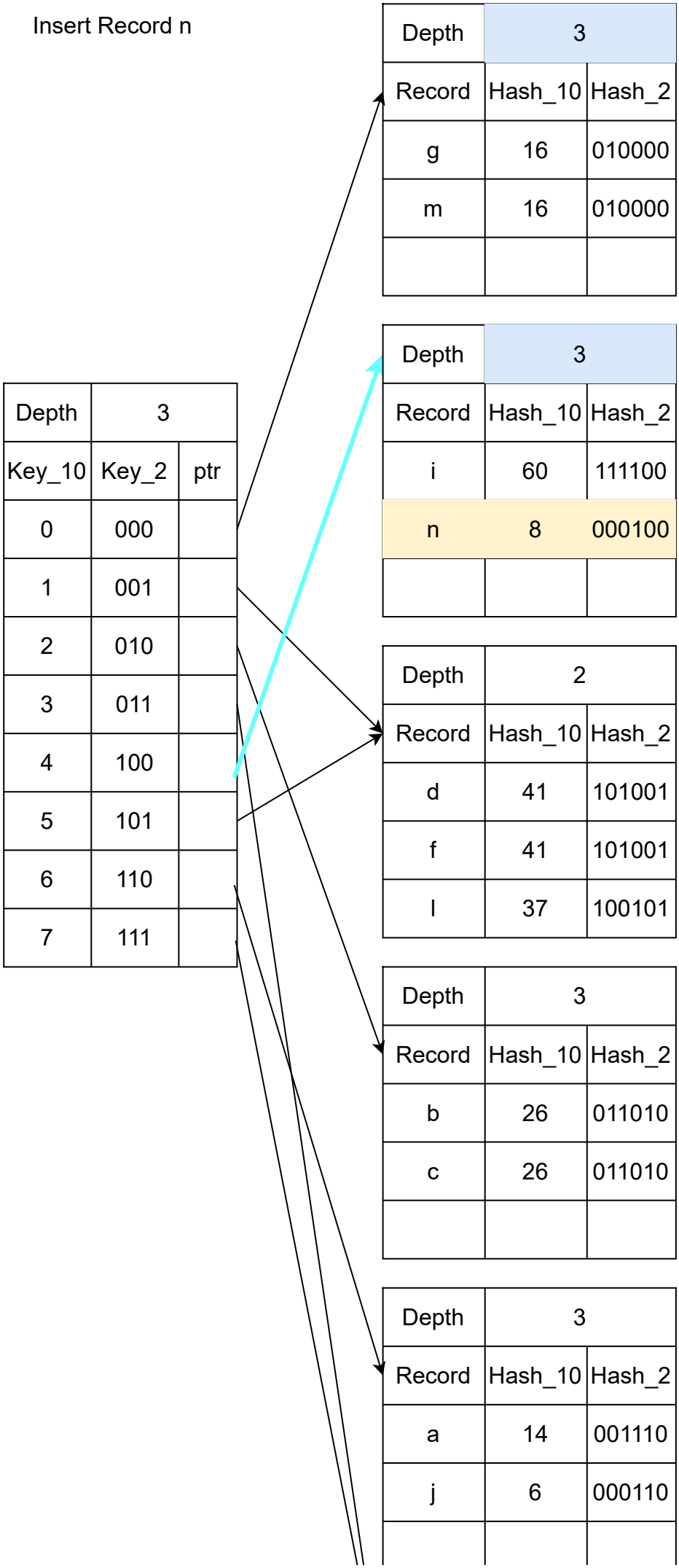
Now we insert records k, l, and m.

Insert Record k, l, m



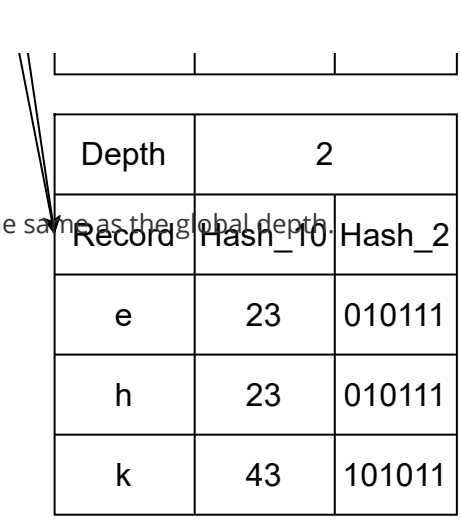
We then insert record n, and this will cause overflow.

Insert Record n



E2

There are **4** buckets that have local depth 3, which are the same as the global depth.



Depth	2	
Record	Hash_10	Hash_2
e	23	010111
h	23	010111
k	43	101011