Version Feb 11, 2011

Introduction to Database Systems CSE 444

Lecture 16: Data Storage and Indexes

cow book

cow books
cow bookends
cow books for kids
cow book database
cow book

About 7,140,000 results (0.18 seconds)

COW BOOKS
Have a look at our singular collection of selected books a
exceptional. STORE INFORMATION. **COW BOOKS**/Na
www.**cowbooks**.jp/english.html - Cached - Similar

COW BOOKS - [Translate this page]
2011/2/5. Weekly Highlight and more... \
www.**cowbooks**.jp/ - Cached - Similar

Amazon.com: Moo **Cow Book** (9780689876837):
★★★★☆ 13 reviews - $12.71 - In stock
Here's a spiffy plush **cow book** with a squeeze-it moo, a
dazzling satin teeth, and four padded cloth pages of jaun
www.amazon.com › Books › Children's Books › Animals

Comic **books** published by Top **Cow**
The comics you enter will be recorded into My Comic **Bo**
Top **Cow**. All Issues; In Stock. Display. Grouped by Title,
www.mycomicshop.com/search?pl=Top%20**Cow** - Cache

cow books | Flickr - Photo Sharing!
1 post - 1 author - Last post: Mar 30, 2007
This photo belongs to. pink_emmie_bat's photostream (6
books · Taco truck?! red bridge ...
www.flickr.com/photos/indrasarrow/439722560/ - Cached

Cash **Cow Books** - Richmond, Virginia (VA) | Com
Cash **Cow Books** company profile in Richmond, VA. Ou
Cash **Cow Books** includes business information such as
www.manta.com/c/mtm9l6j/cash-**cow-books** - Cached

Database Management Systems (Third Edition)
If you are using the **book** and have found bugs or typos,
presentation or content, please send email. ...
pages.cs.wisc.edu/~db**book**/ - Cached - Similar

Where we stand

- How to use a DBMS as a:
  - Data analyst: SQL, SQL, SQL, …
  - Application programmer: JDBC
  - Database admin: tuning, triggers, security
  - Massive-scale data analyst: Pig/MapReduce
- How DBMSs work:
  - Transactions
  - Data storage and indexing
  - Query execution
- Databases as a service

today

Feb 7	Transactions: Concurrency Control lecture 14-15 Midterm review on the board	Midterm	Data Storage and Indexing lecture 16 Homework 2
Feb 14	Database Tuning	Relational Algebra	Query Processing Overview Project 3 due
[illegible] 21	No class idents Day)	Operator Algorithms	Query Optimization
Feb 28	Query Optimization	Query Optimization	Parallel and Distributed DBMSs
Mar 7	Pig Latin	TBA	Wrap-up
Mar 14	Final Exam Thursday, March 17, 8:30am-10:20am, in class		

x

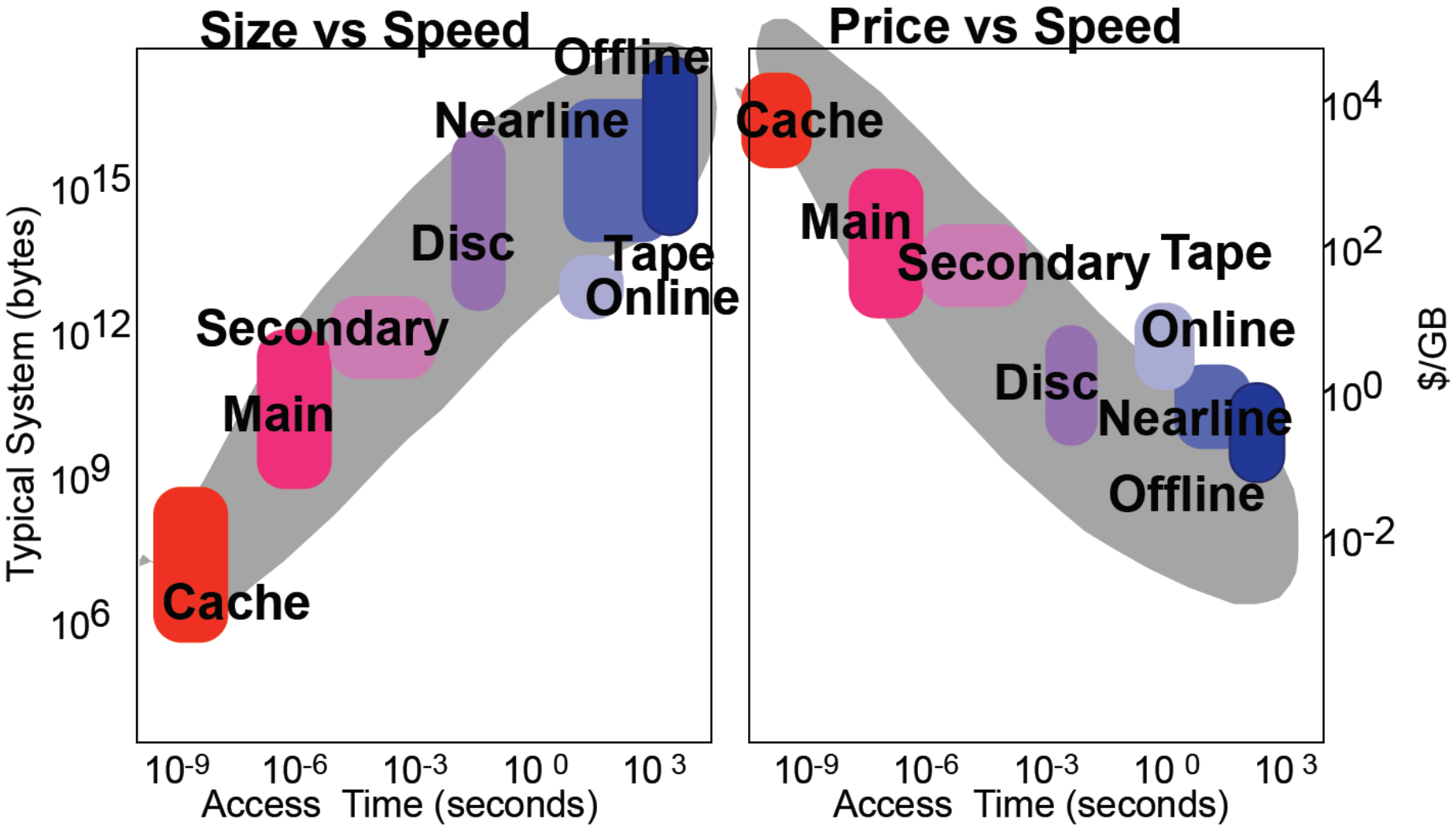
Outline

- Storage model

- Index structures (Section 14.1)
  - [Old edition: 13.1 and 13.2]

- B-trees (Section 14.2)
  - [Old edition: 13.3]

Memory hierarchy

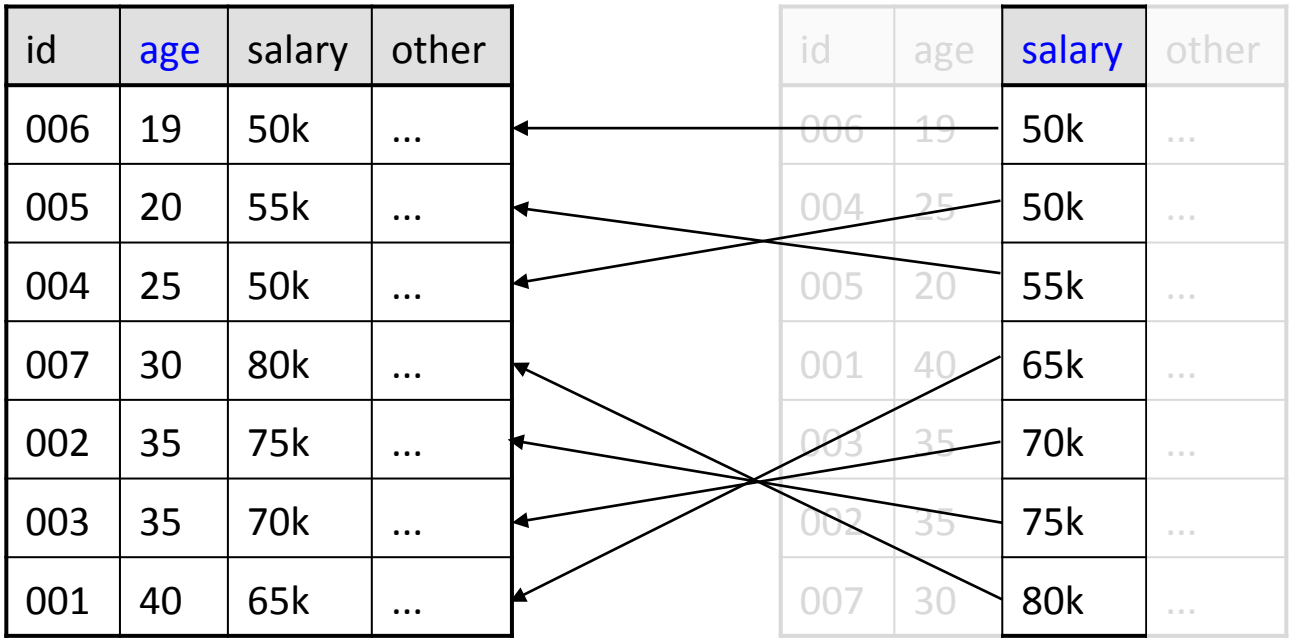


Source: "Long Term Storage Trends and You", Jim Gray, 2006: http://research.microsoft.com/en-us/um/people/gray/talks/

Outline

- Storage model
- Index structures (Section 14.1)
  - [Old edition: 13.1 and 13.2]
- B-trees (Section 14.2)
  - [Old edition: 13.3]

High-level overview: Indexes



data file = index file
clustered (primary) index

index file
unclustered (secondary) index

x

Database File Types

The data file can be one of:

- Heap file
  - Set of records, partitioned into blocks
  - Unsorted
- Sequential file
  - Sorted according to some attribute(s) called (sort) key

different from "primary key"!

x

Index

- A (possibly separate) file, that allows fast access to records in the data file given a search key
- The index contains (key, value) pairs:
  - The key = an attribute value
  - The value = either a pointer to the record, or the record itself

again different from "primary key"!

x

Index Classification

- Clustered/unclustered
  - Clustered = records close in index are close in data
  - Unclustered = records close in index may be far in data

- Primary/secondary
  - Meaning 1: (Cow book)
    - Primary = is over attributes that include the primary key
    - Secondary = otherwise
  - Meaning 2: means the same as clustered/unclustered
    (Stanford book)

- Organization: B+ tree or Hash table

x

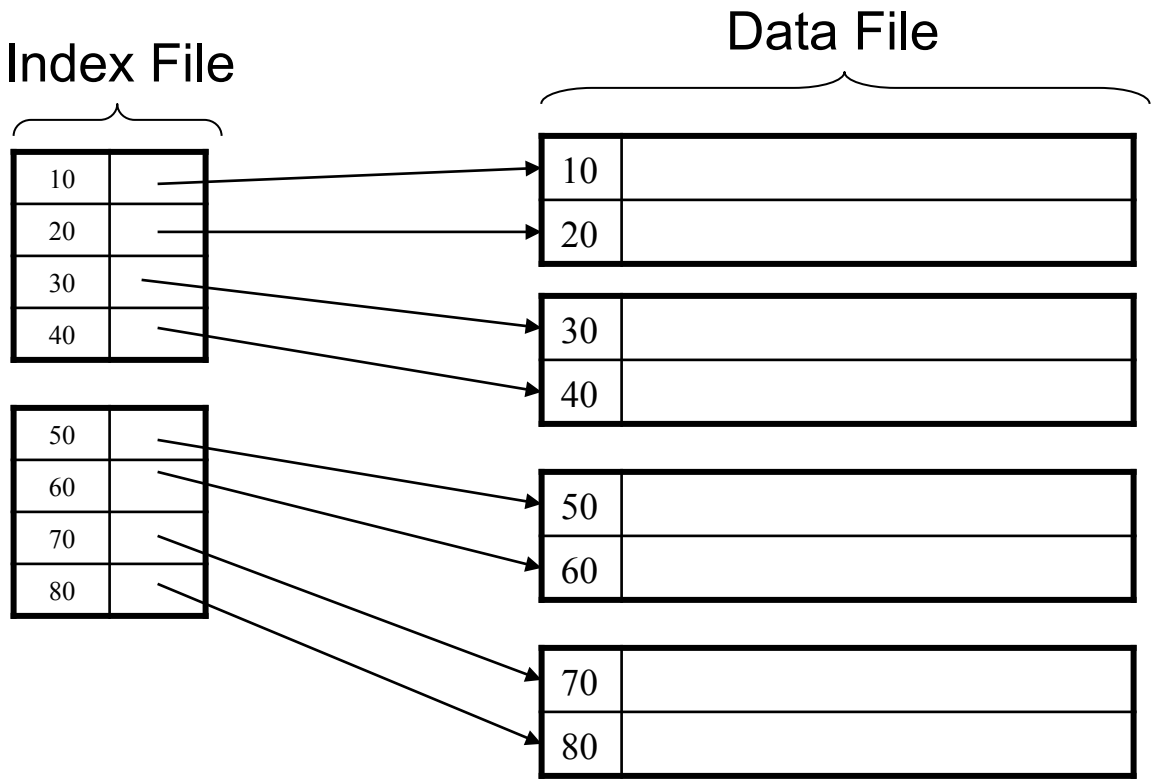
Clustered/Unclustered

- Clustered
  - Index determines the location of indexed records
  - Typically, clustered index is one where values are data records (but not necessary)

- Unclustered
  - Index cannot reorder data, does not determine data location
  - In these indexes: value = pointer to data record

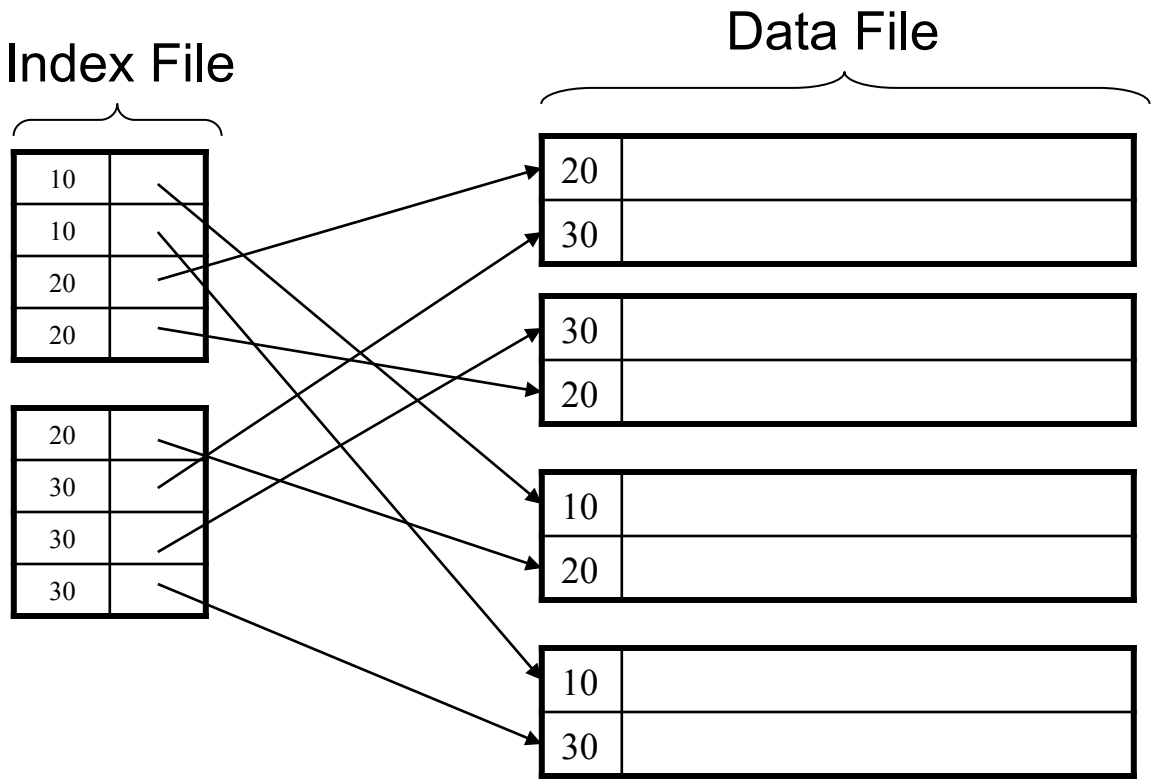
Clustered Index

- File is sorted on the index attribute
- Only one per table



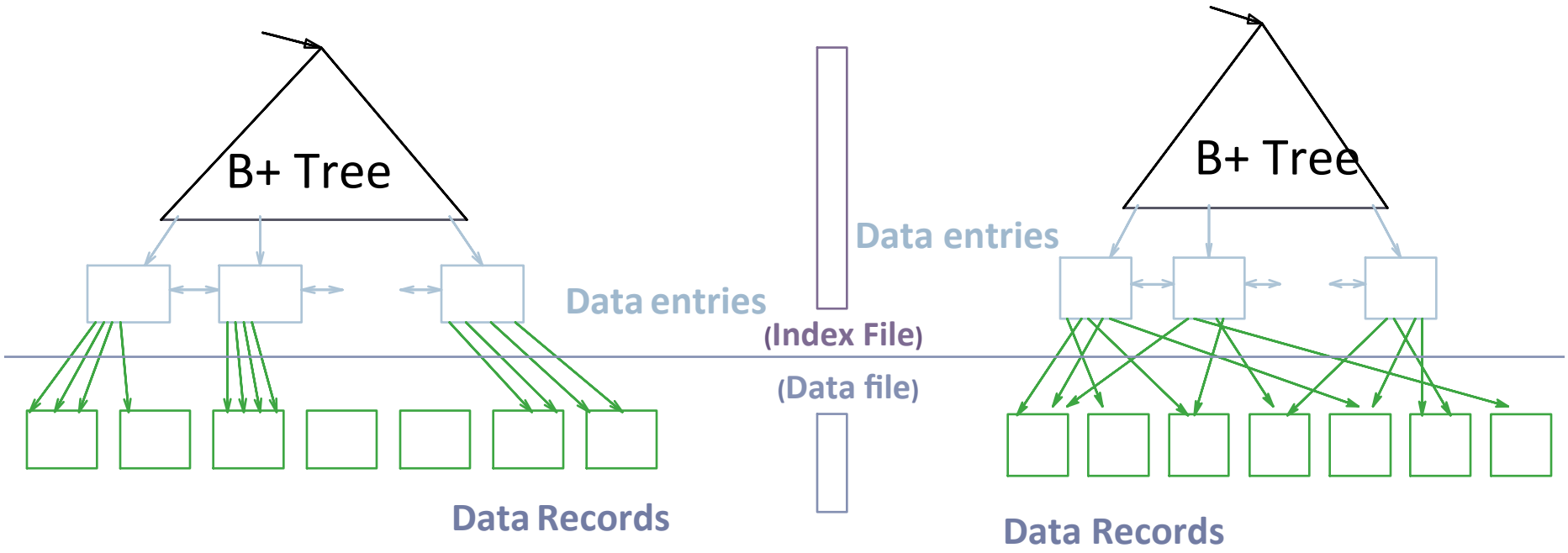
Unclustered Index

- Several per table



x

Clustered vs. Unclustered Index



More commonly, in a clustered B+ Tree index,
data entries are data records

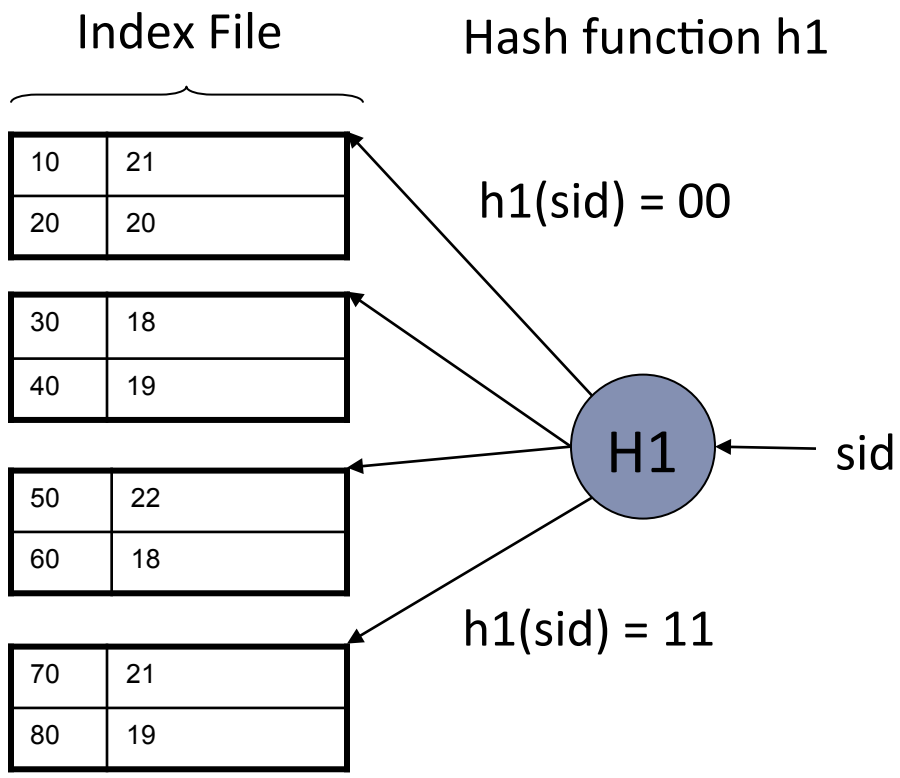
Hash-Based Index Example

Example hash-based index
on sid (student id)

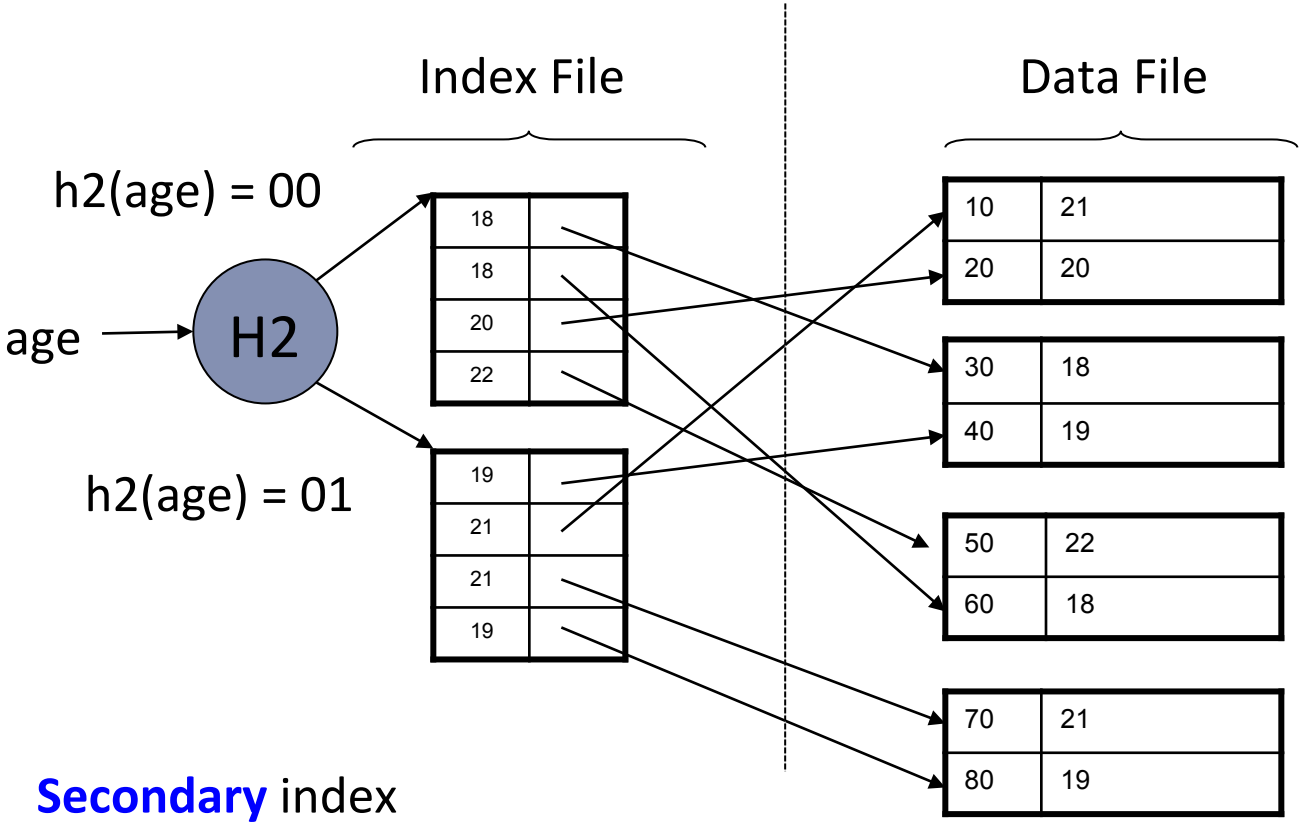
This is a **primary** index
because it determines the
order of indexed records

In this case, data entries in the index
are actual data records
There is no separate data file

This index is also **clustered**



Hash-Based Index Example 2



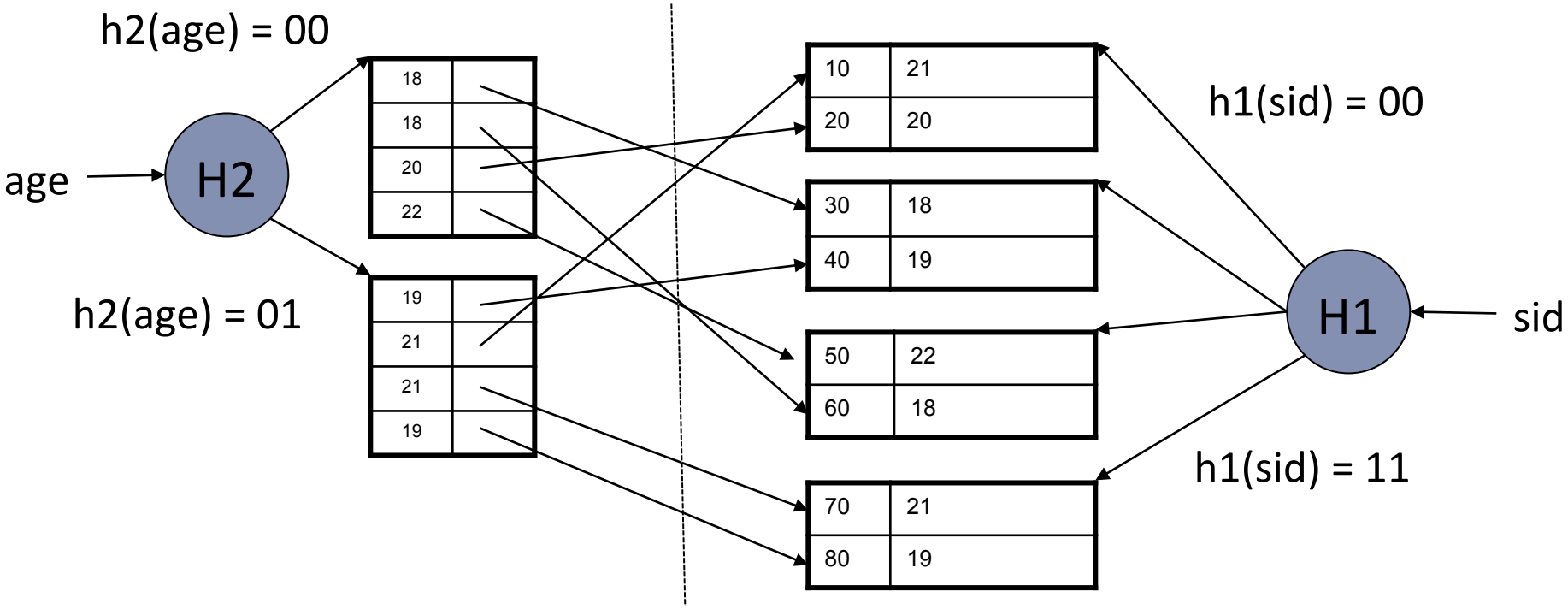
Secondary index

Data entries in index are (key,RID) pairs

Unclustered index

Hash-Based Index

Good for point queries but not range queries



Another example
of unclustered/secondary index

Another example of
clustered/primary index

Outline

- Storage model
- Index structures (Section 14.1)
  - [Old edition: 13.1 and 13.2]
- B-trees (Section 14.2)
  - [Old edition: 13.3]

B+ Trees

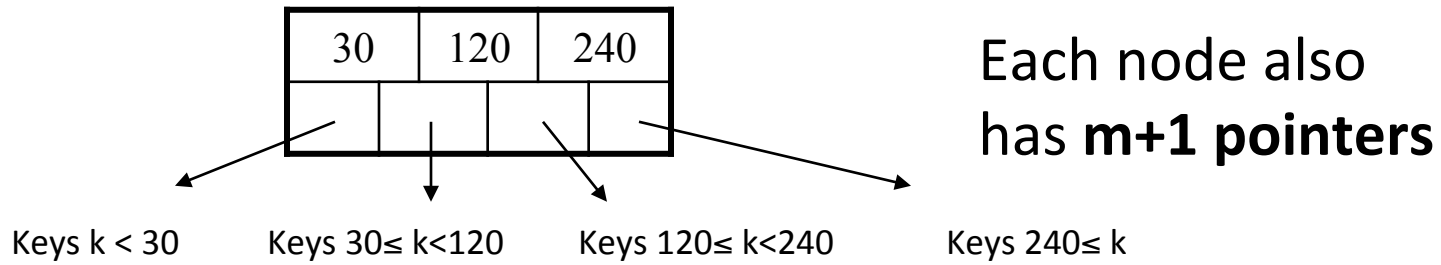
- Search trees

- Idea in B Trees
  - Make 1 node = 1 block
  - Keep tree balanced in height

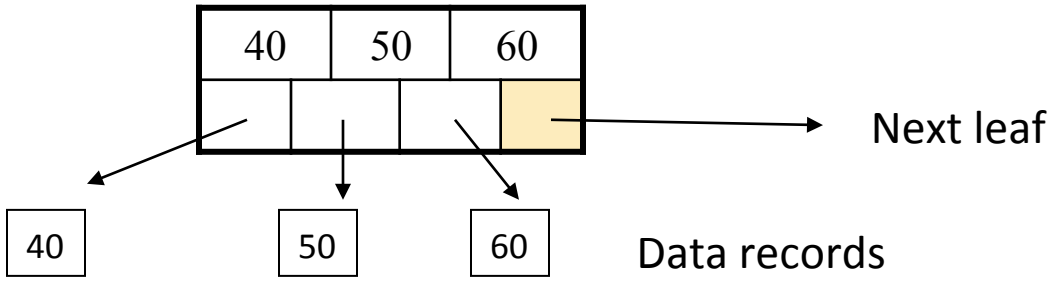
- Idea in B+ Trees
  - Make leaves into a linked list: facilitates range queries

B+ Trees Basics

- Parameter d = the degree
- Each interior node has **d ≤ m ≤ 2d keys** (except root)



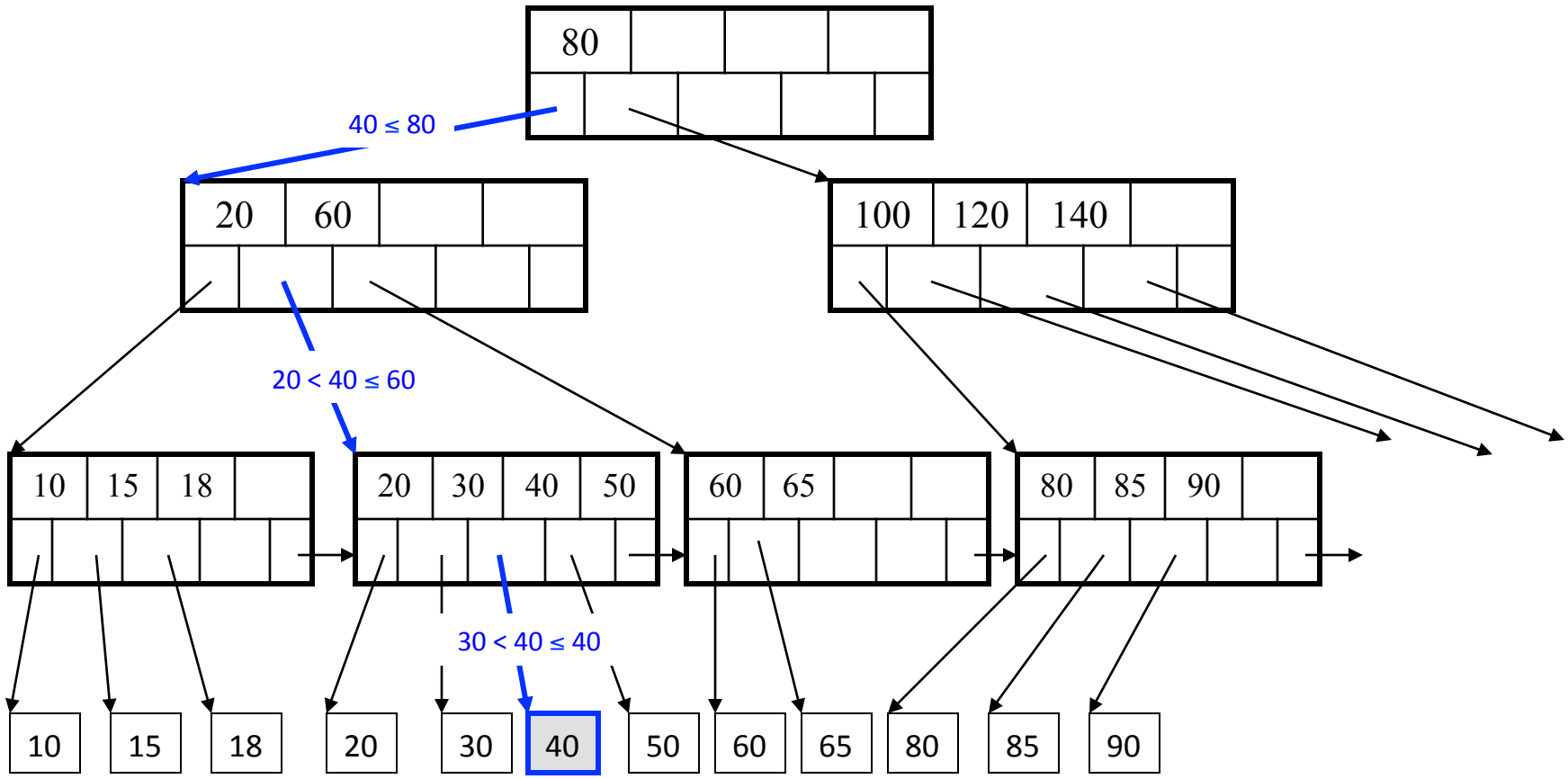
- Each leaf node has **d ≤ m ≤ 2d keys**



B+ Tree Example

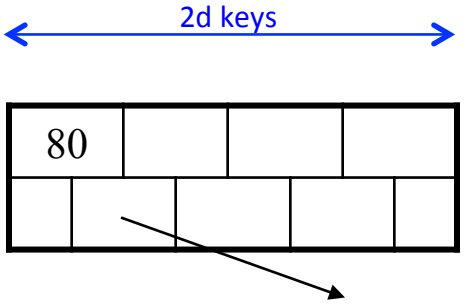
Find the key 40

d = 2



B+ Tree Design

- How large d ?
- Example:
  - Key size = 4 bytes
  - Pointer size = 8 bytes
  - Block size = 4096 bytes
- 2d x 4 + (2d+1) x 8 <= 4096
- d = 170



Searching a B+ Tree

- Exact key values:
  - Start at the root
  - Proceed down, to the leaf

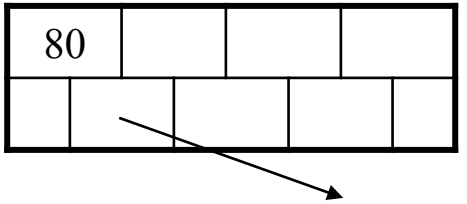
```
select  name
from    people
where   age = 25
```

- Range queries:
  - As above
  - Then sequential traversal

```
select  name
from    people
where   20 <= age
    and age <= 30
```

B+ Trees in Practice

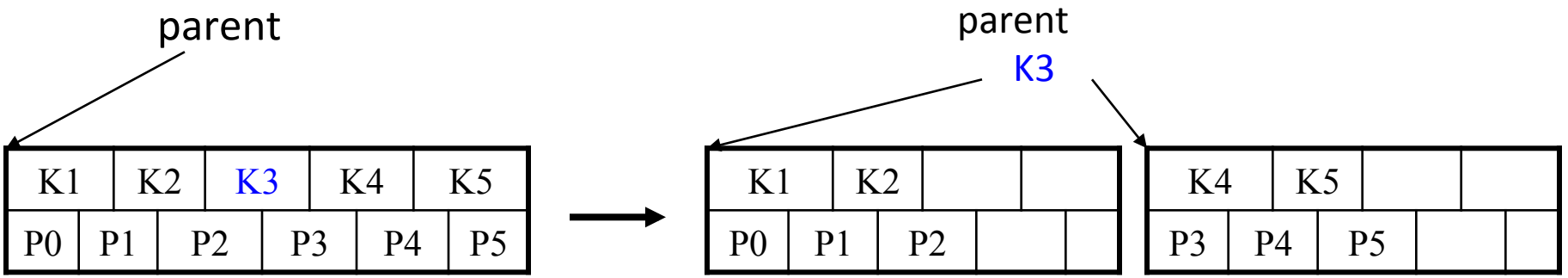
- Typical degree: 100. Typical fill-factor: 67%
  - average fanout = 133
- Typical capacities
  - [Height 1: 133^1 = 133 records]
  - Height 3: 133^3 = 2,352,637 records
  - Height 4: 133^4 = 312,900,700 records
- Can often hold top levels in buffer pool
  - Level 1 = 1 page = 8 Kbytes
  - Level 2 = 133 pages = 1 Mbyte
  - Level 3 = 17,689 pages = 133 Mbytes



Insertion in a B+ Tree

Insert (K, P)

- Find leaf where K belongs, insert
- If no overflow (2d keys or less), halt
- If overflow (2d+1 keys), split node, insert in parent:

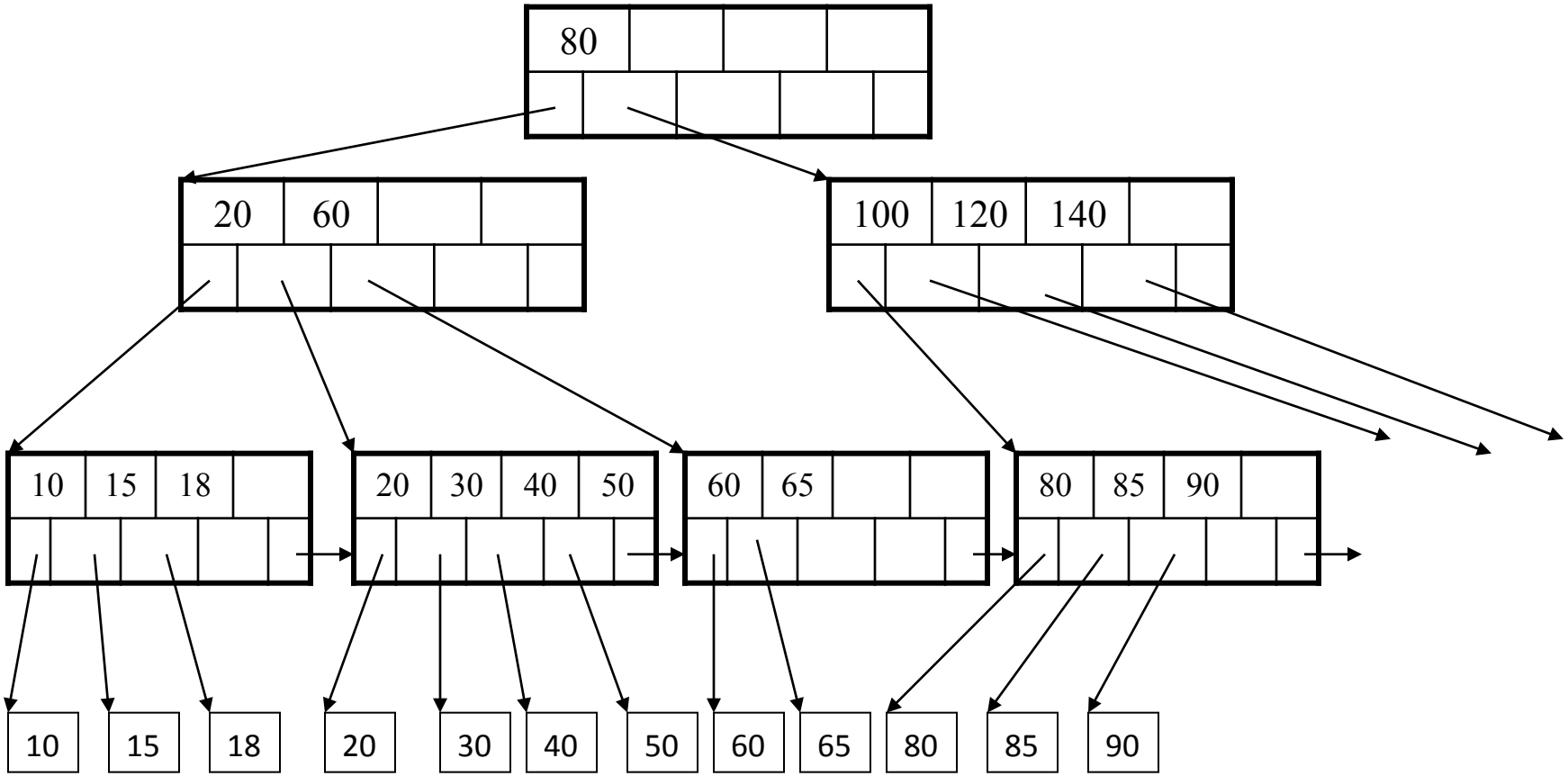


- If leaf, keep K3 too in right node
- When root splits, new root has 1 key only

Insertion in a B+ Tree

d=2

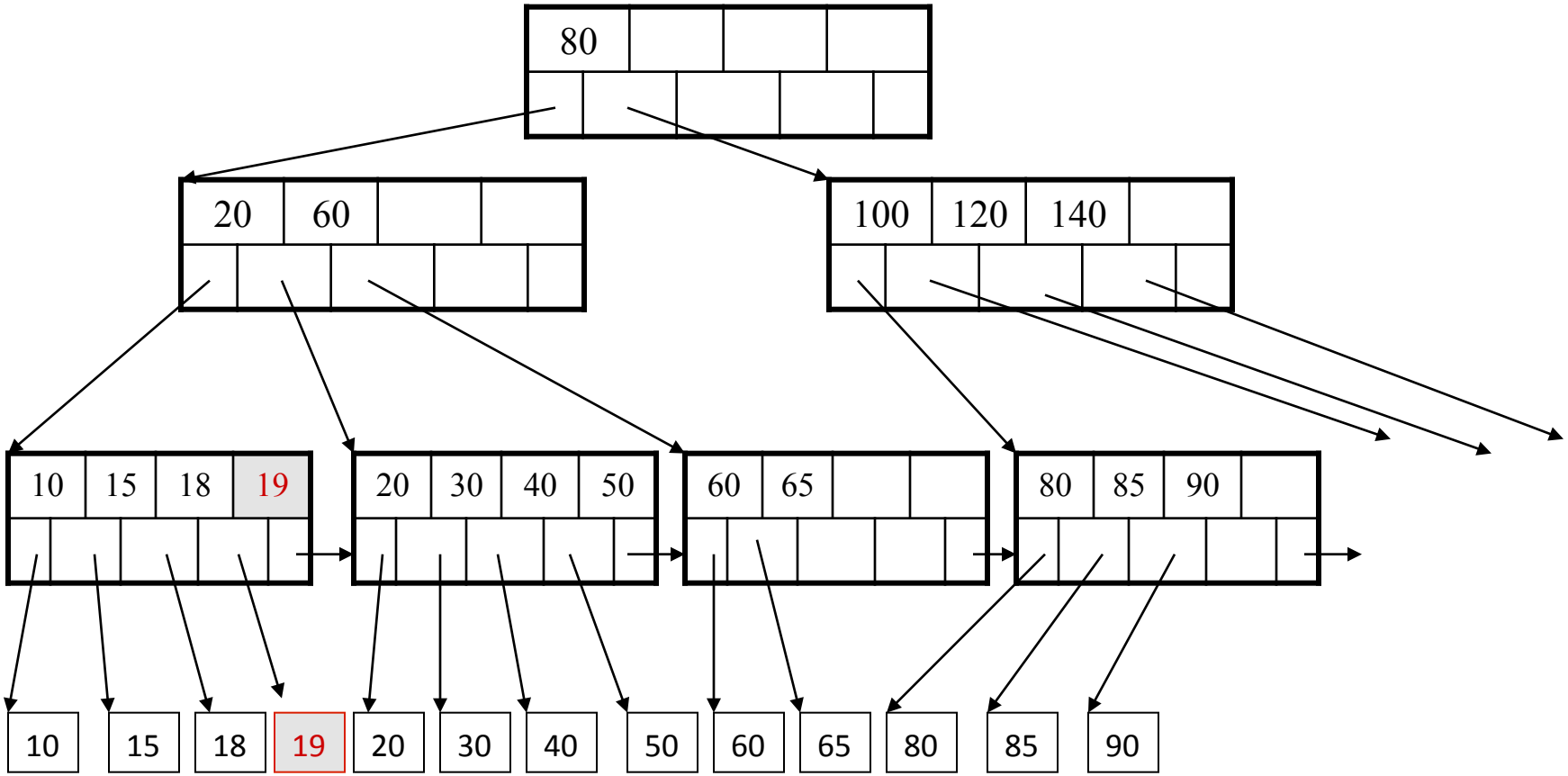
Insert K=19



Insertion in a B+ Tree

d=2

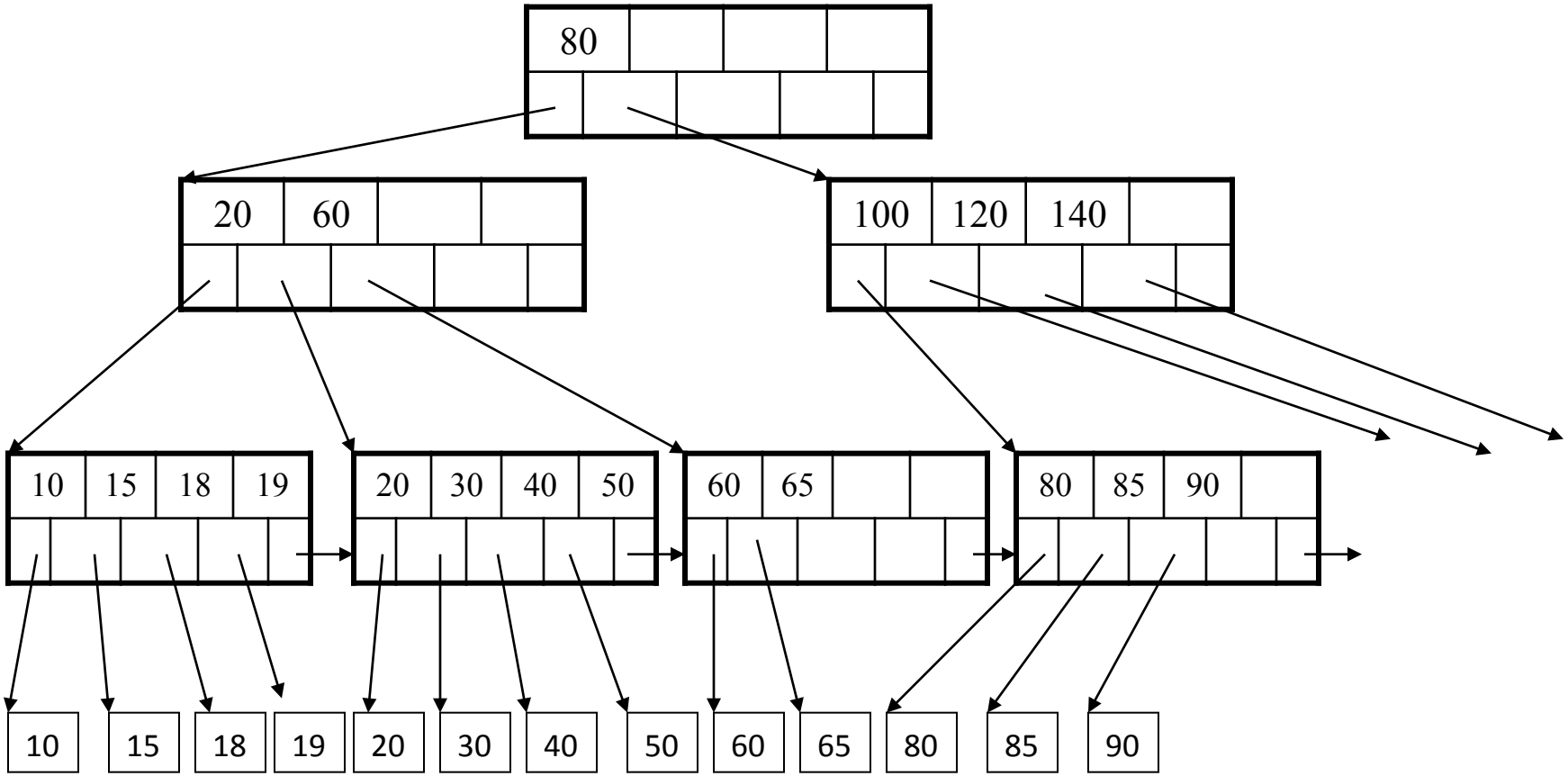
After insertion



Insertion in a B+ Tree

d=2

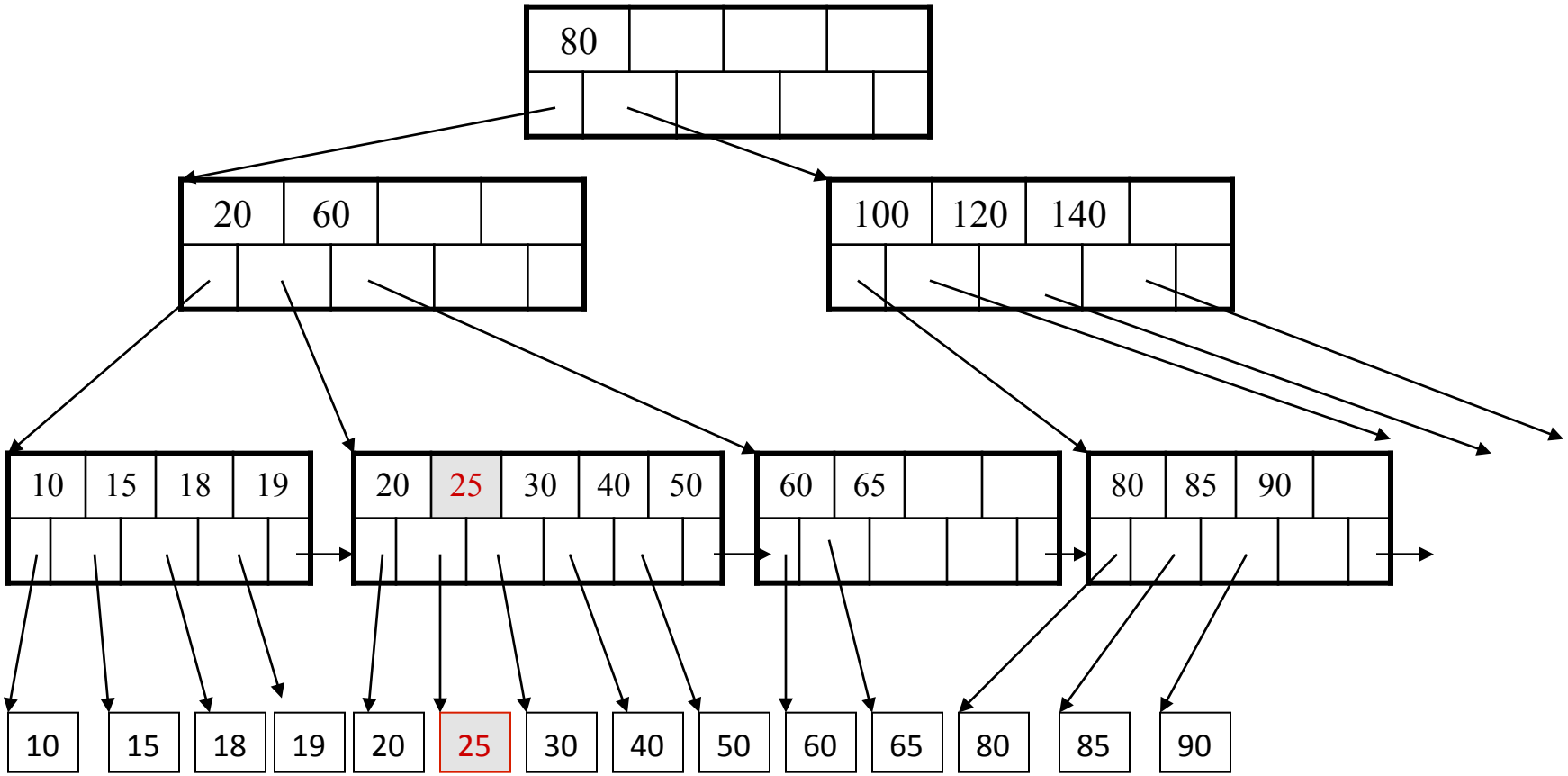
Now insert 25



Insertion in a B+ Tree

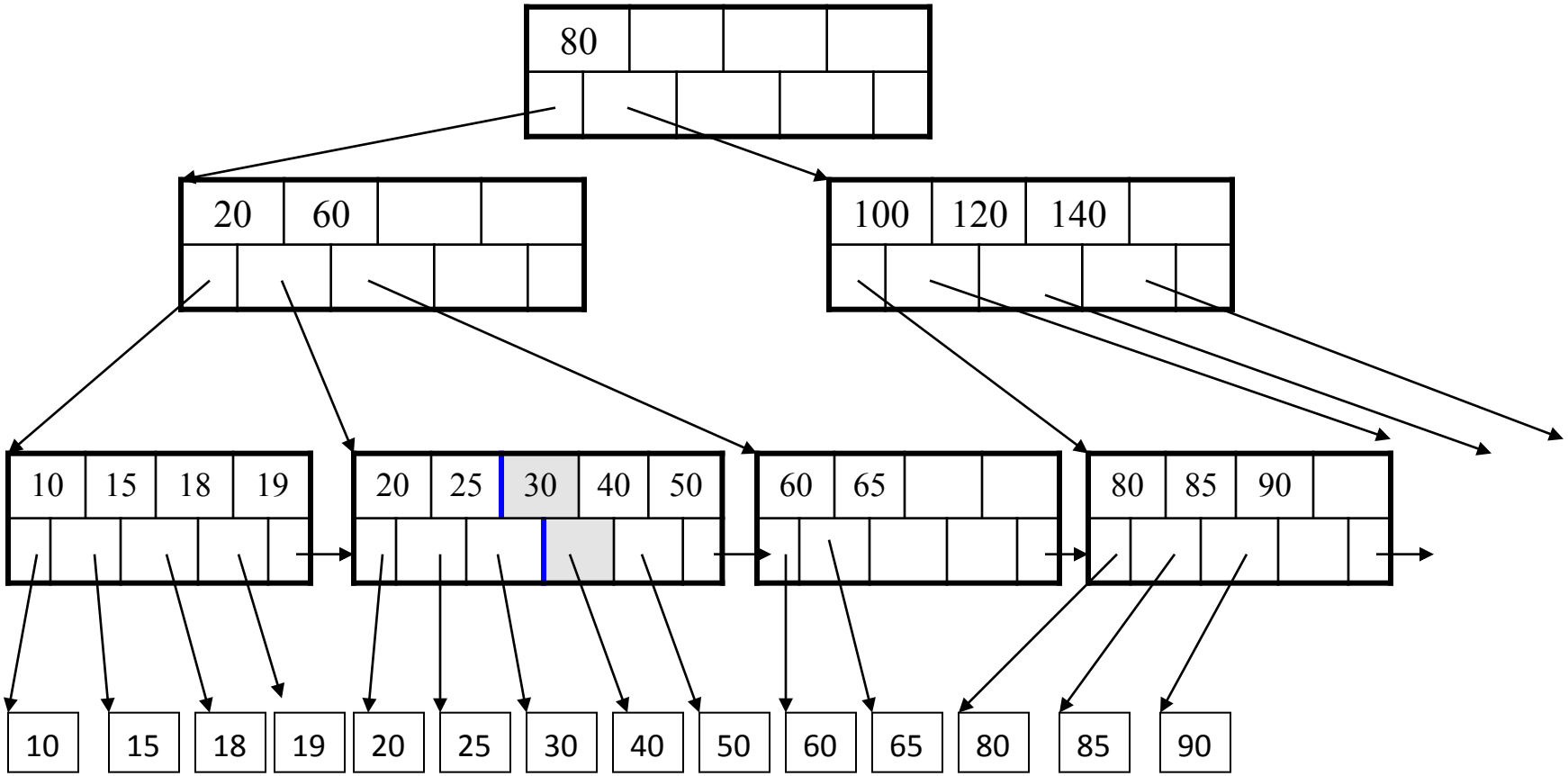
d=2

After insertion



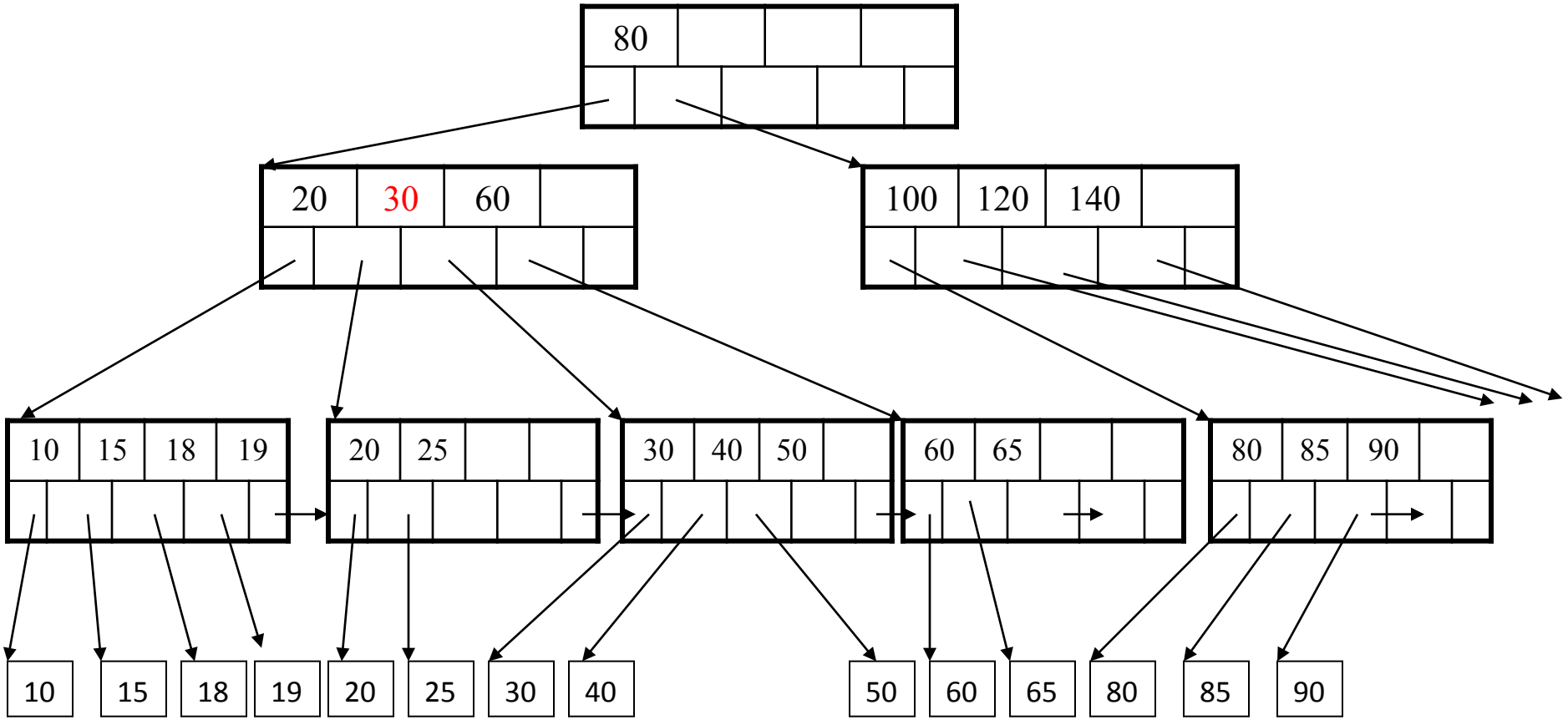
Insertion in a B+ Tree

But now have to split !



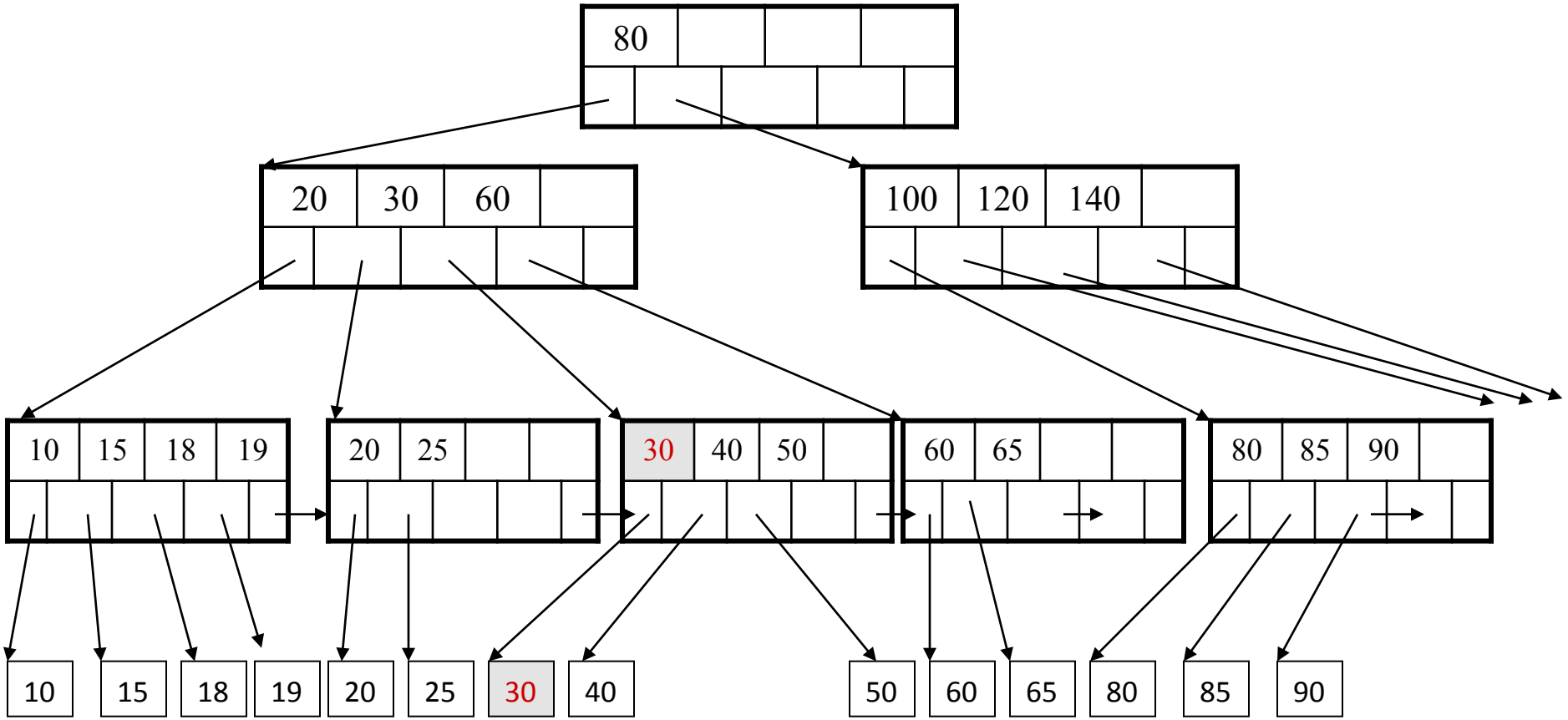
Insertion in a B+ Tree

After the split

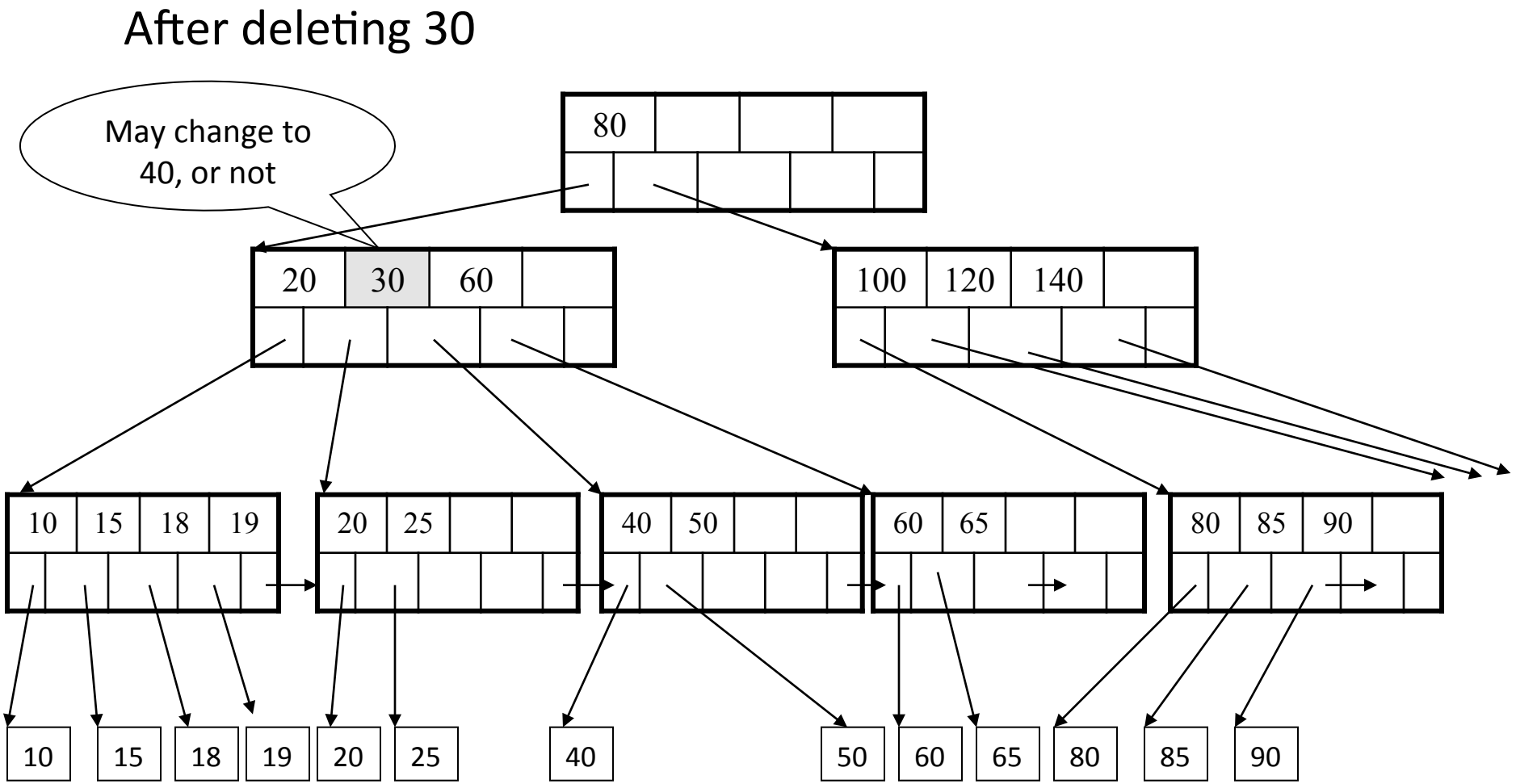


Deletion from a B+ Tree

Delete 30

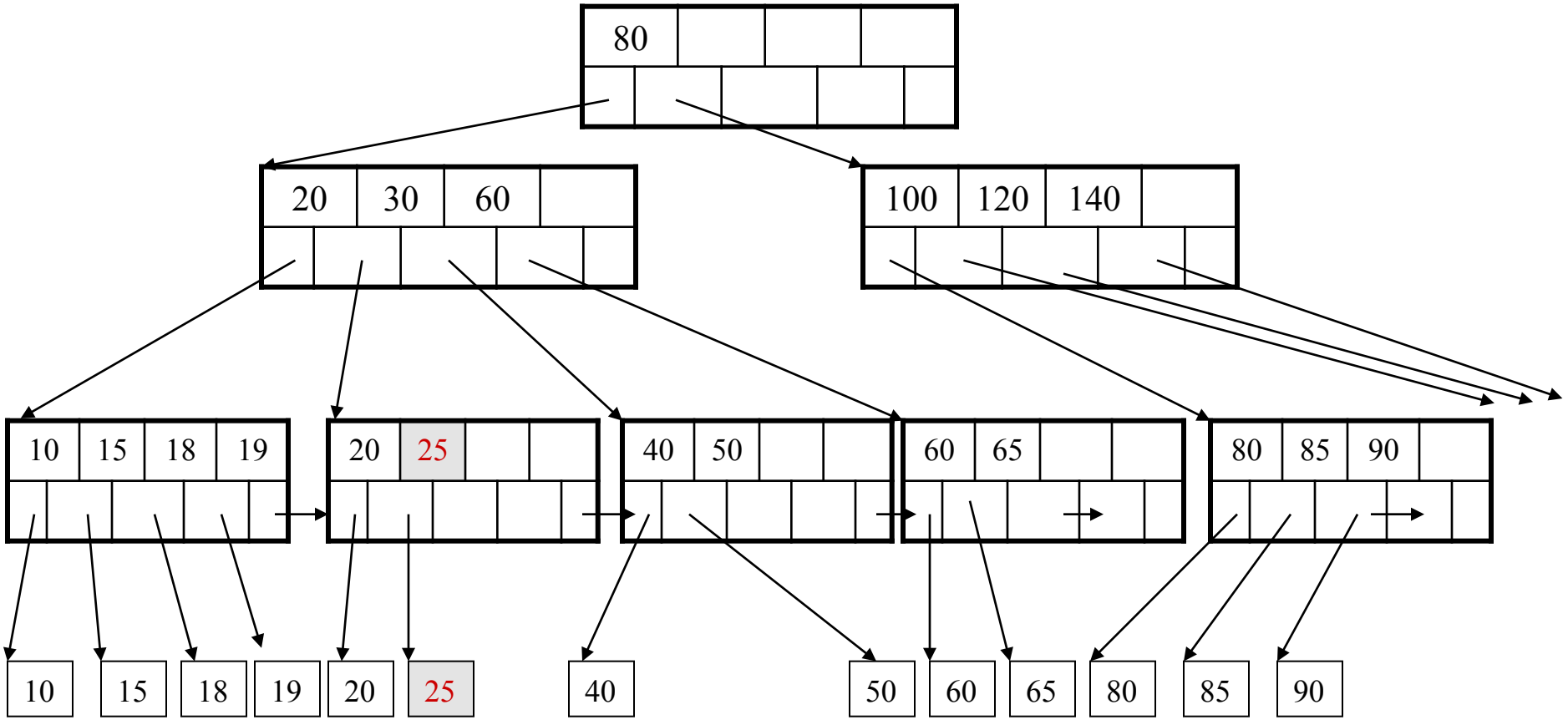


Deletion from a B+ Tree



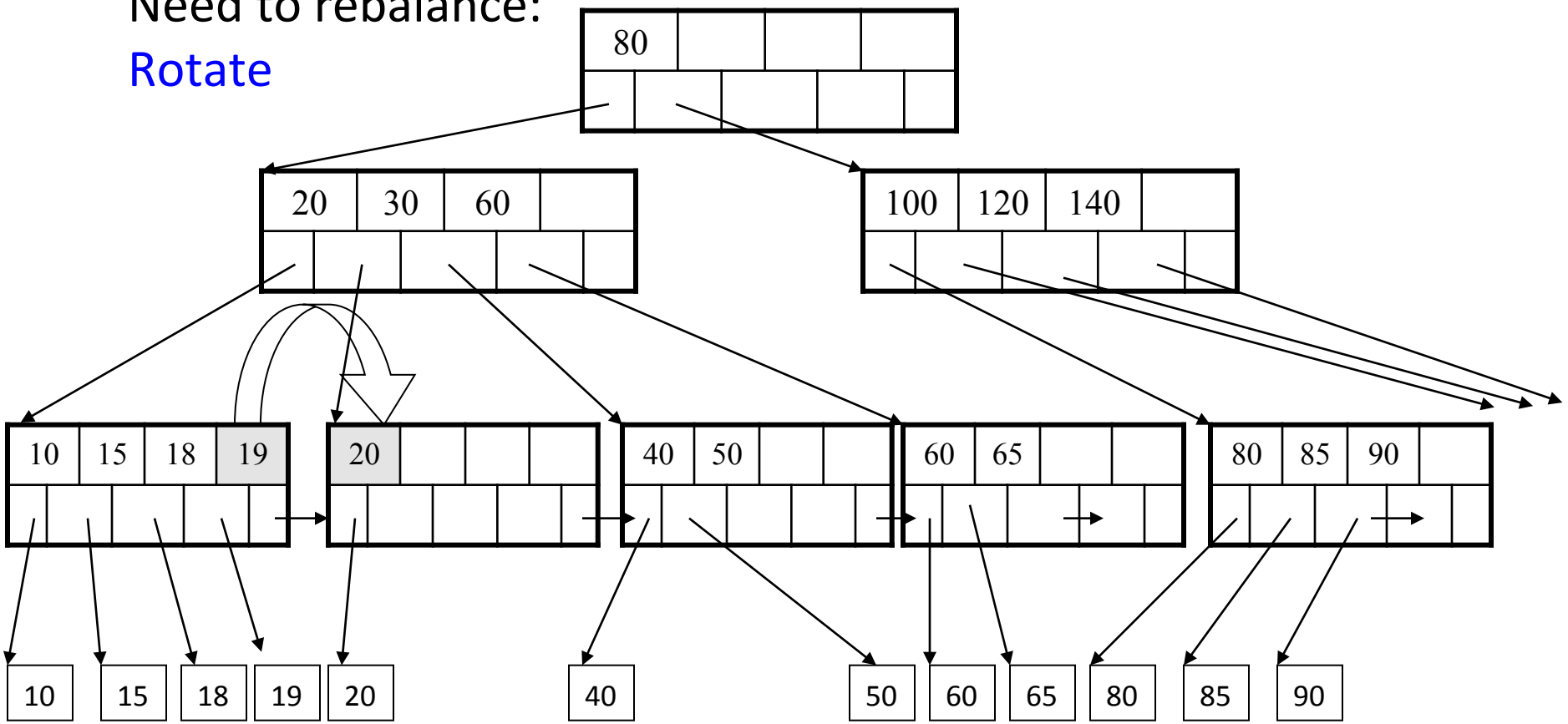
Deletion from a B+ Tree

Now delete 25



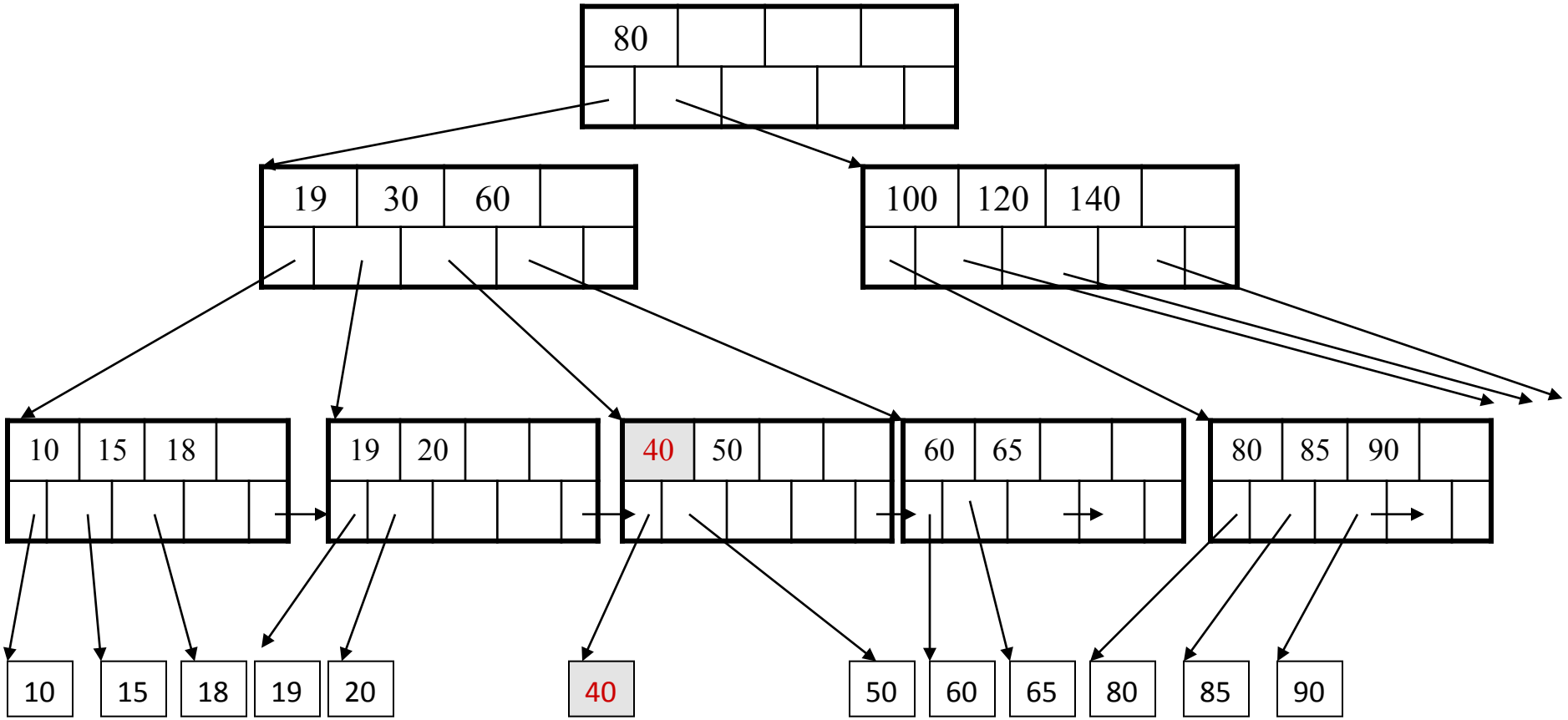
Deletion from a B+ Tree

After deleting 25
Need to rebalance:
Rotate



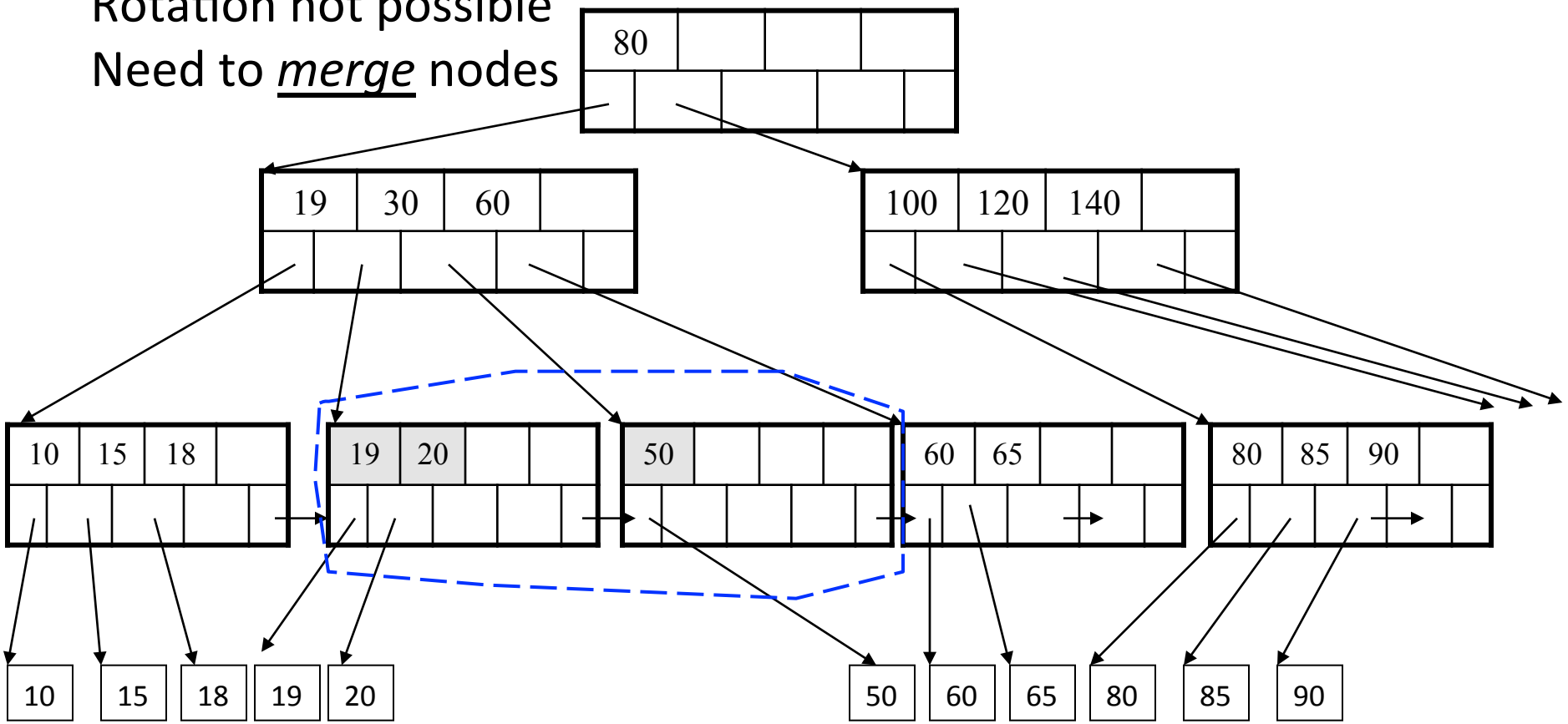
Deletion from a B+ Tree

Now delete 40



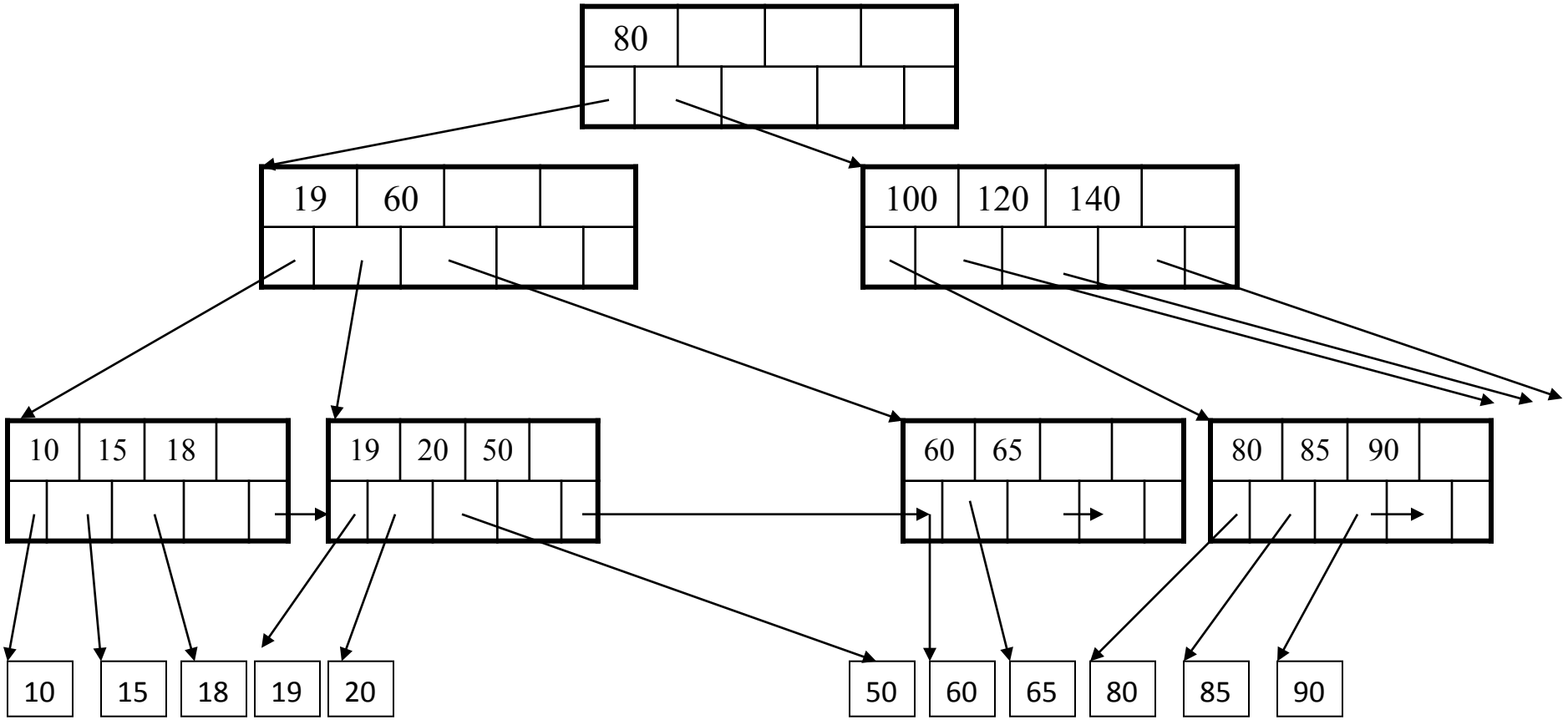
Deletion from a B+ Tree

After deleting 40
Rotation not possible
Need to *merge* nodes



Deletion from a B+ Tree

Final tree



Summary of B+ Trees

- Default index structure on most DBMS
- Very effective at answering 'point' queries:
  productName = 'gizmo'
- Effective for range queries:
  50 < price AND price < 100
- Less effective for multirange:
  50 < price < 100 AND 2 < quant < 20

Indexes in PostgreSQL

```
CREATE TABLE V(M int, N varchar(20), P int);
```

```
CREATE INDEX V1_N ON V(N)
```

```
CREATE INDEX V2 ON V(P, M)
```

```
CREATE  NDEX VVV ON V(M, N)
```

```
CLUSTER V USING V2
```

Makes V2 clustered