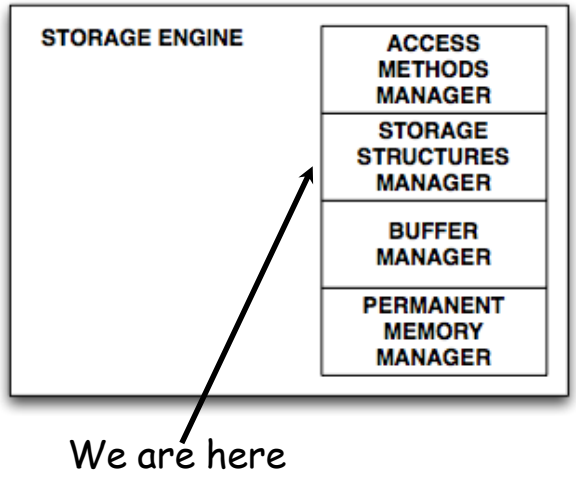
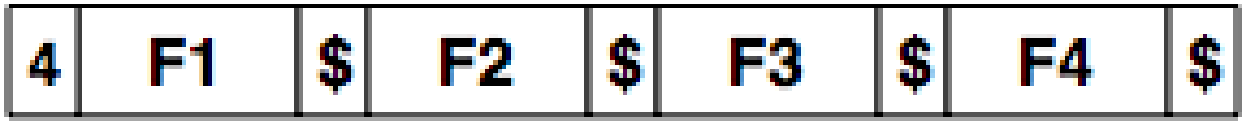
Data organizations

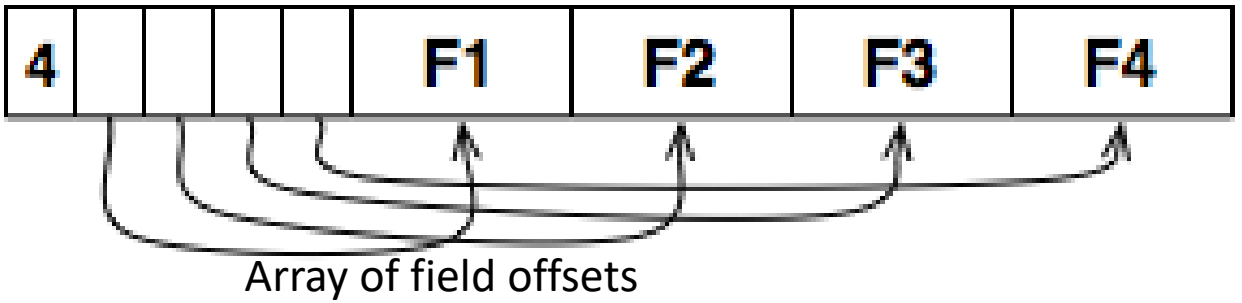


Representing a record

- Assume records are shorter than a page



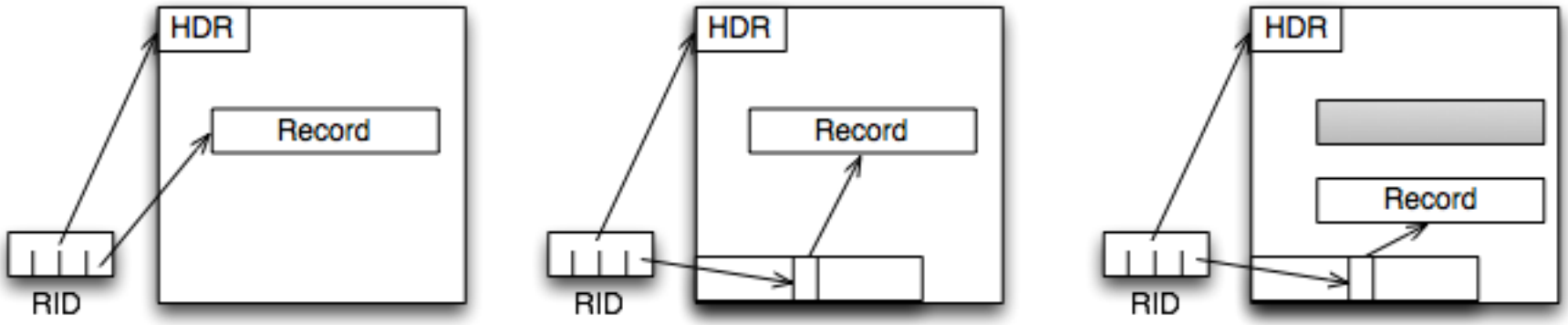
Fields delimited by special symbols



Array of field offsets

Storing records in a page

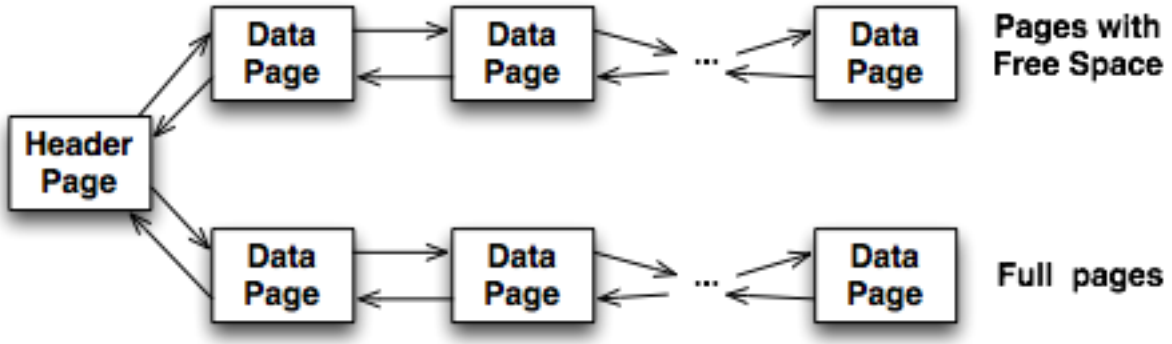
- Records are stored in *pages* of few KB.
- RID = (Page PID, position within page)
- If a record moves on a page, its RID must non change



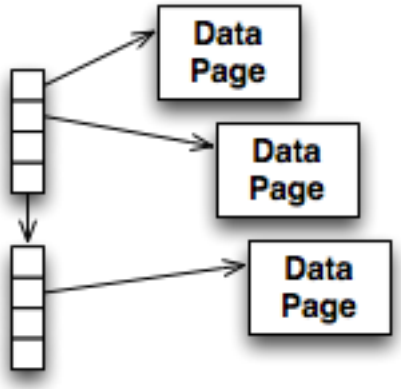
Storage layout of a page

Organizing a collection of pages

- Doubly Linked lists of pages



- Directory of pages



The directory entry for a page includes the number of free bytes on the page.

Heap organizations

- A new record is put at the end of the file
- Very simple, efficient in term of memory used
- The standard organization for every DBMS
- Ideal for:
  - Situations where insert is more common than search
  - Files where massive search is common - efficient equality search and range search need additional data structures
- Cost of memory
- Cost of search

Sequential Organizations

- Data are sorted on a “key” K
  - (Here *key* stands for *attribute*, not necessarily identifying a tuple)
- Useful for search on K – equality and range
- Problems to insert records
- Not commonly used
- Cost of memory
- Cost of search

Insertion on Sequential Organizations

- Differential file
- Keeping empty space in each page
  - Page splitting and page balancing

Heap vs. sequential

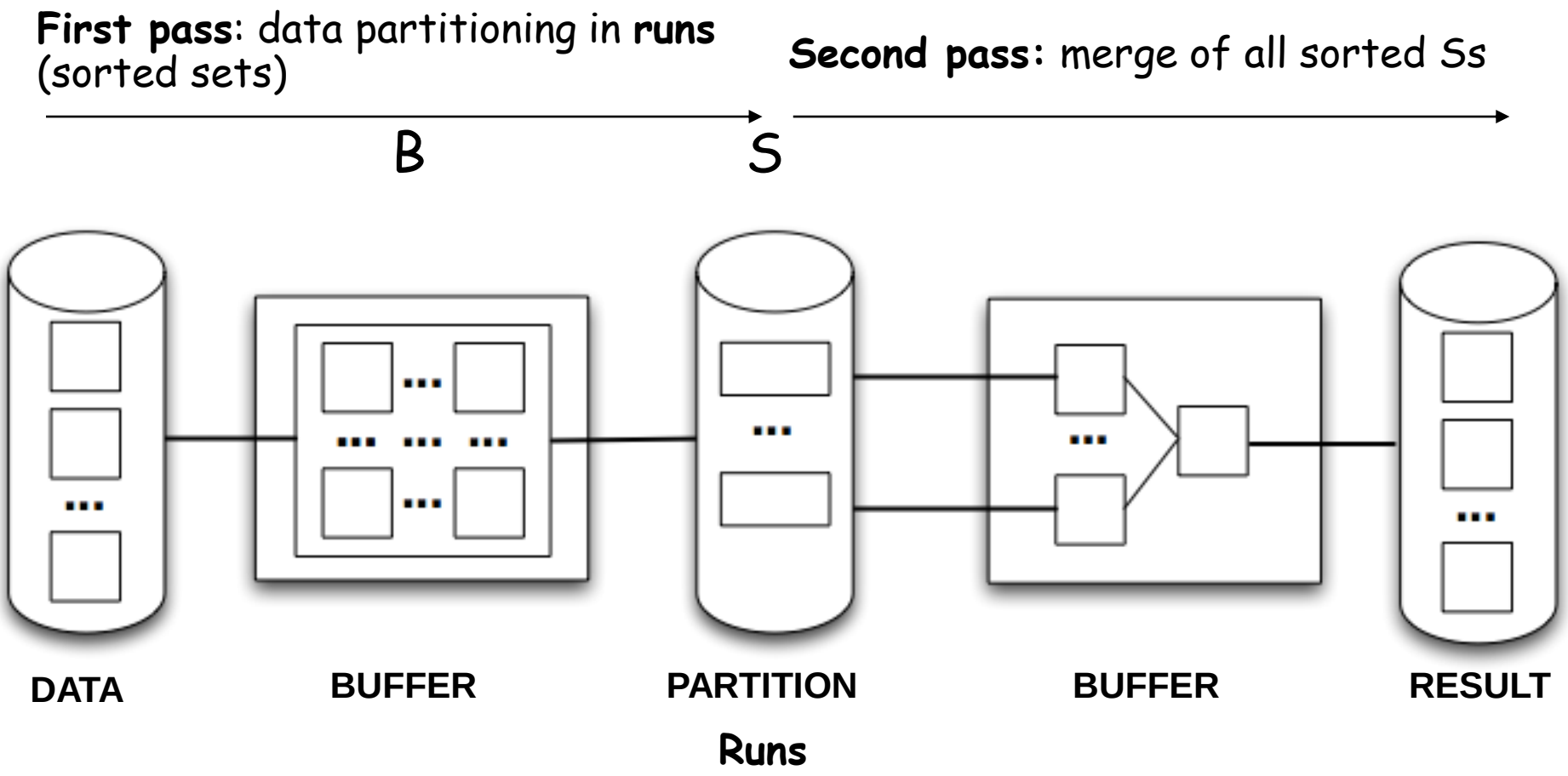
$s_f = (k_2 - k_1)/(k_{max} - k_{min})$

Type	Memory	Equality search C_s	Range search	Insertion	Deletion
Heap					
Sequential					

External sorting

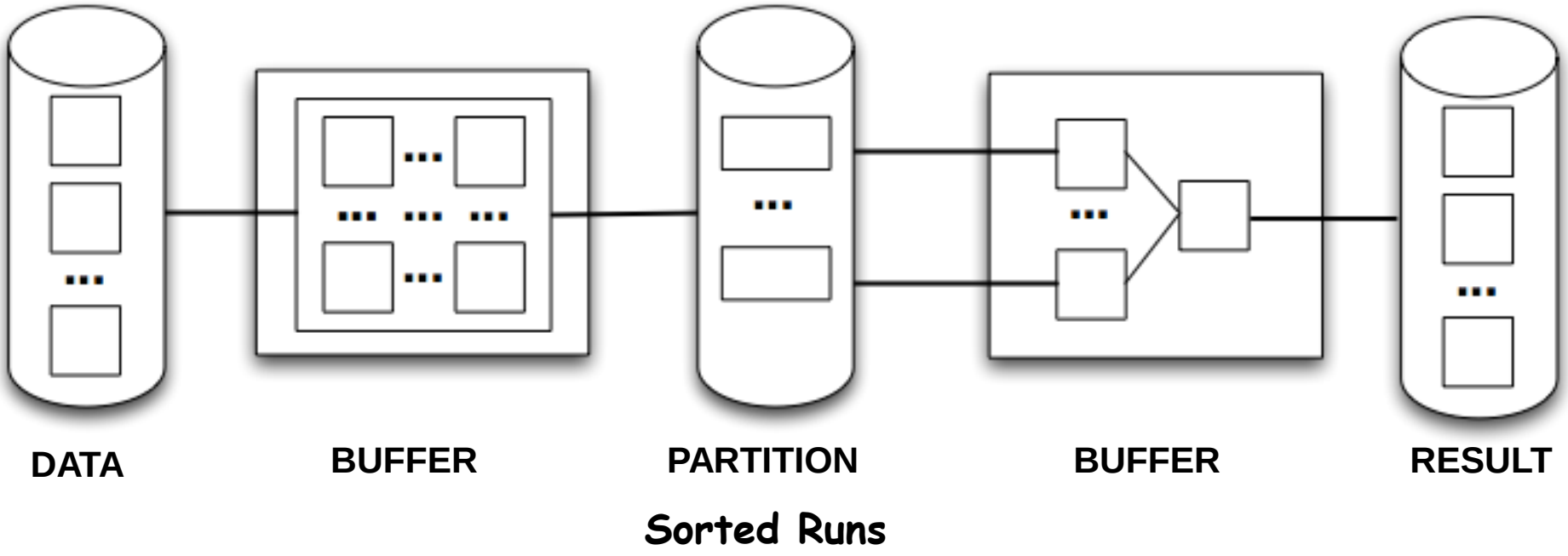
- Sorting is a classical problem in computer science
- Sorting is a common operation in a DBMS
- Merge-sort algorithm

Merge sort in two passes



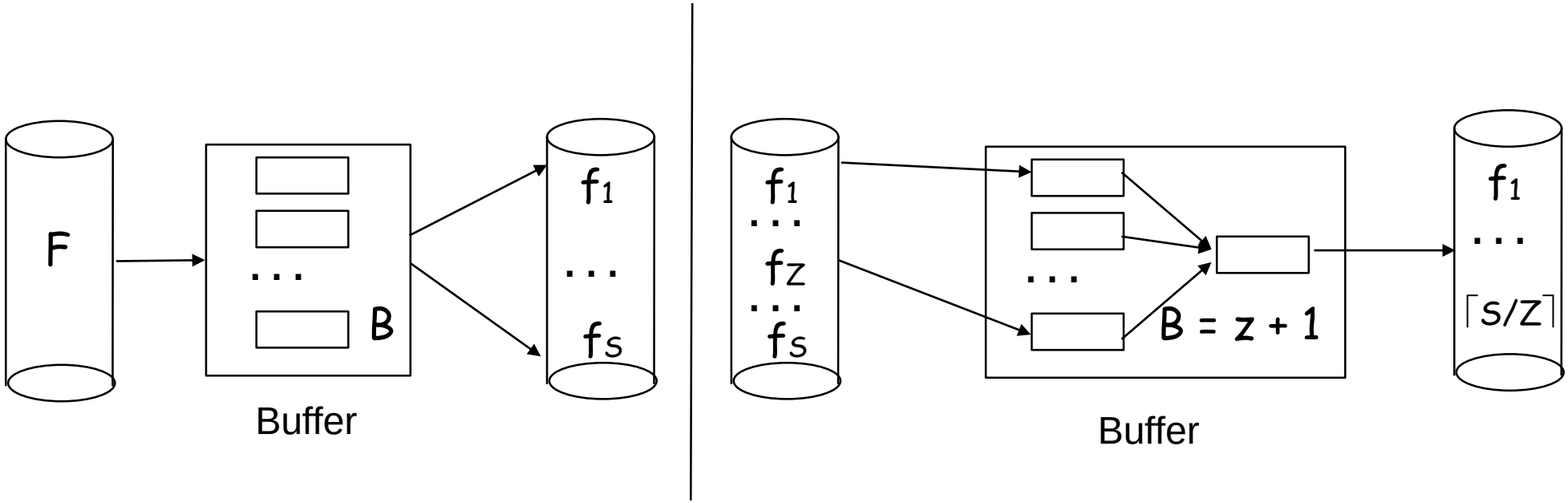
Merge sort in two passes: cost

- C = 4 × NPag(Data)
- How many buffer pages B are needed to sort NPag(Data) with one merge phase?

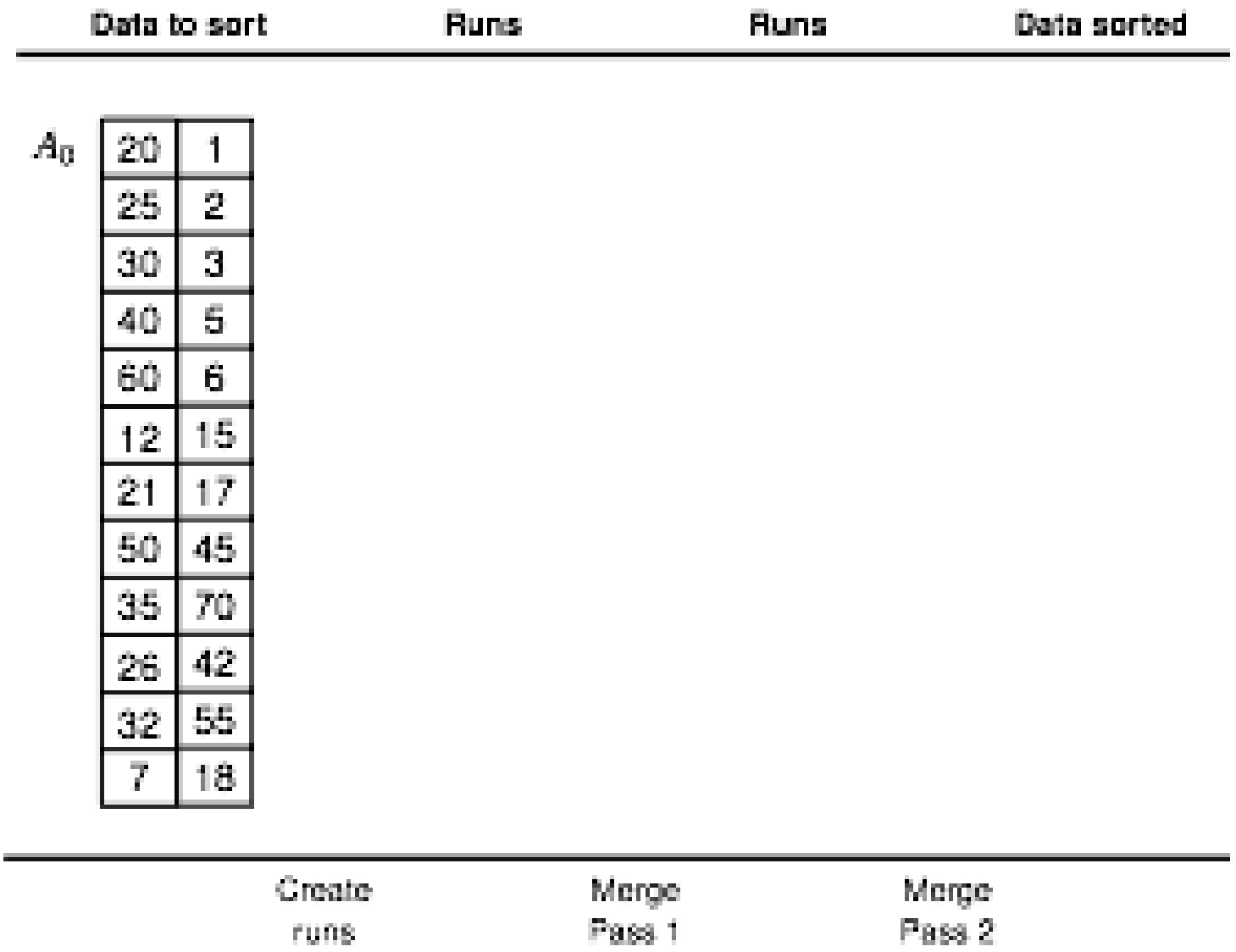


Merge-sort with several passes

- After the first pass of data partitioning in sorted runs, several merge passes of order Z are required...



Example with B=3



Merge sort: cost

- Buffer with B pages
- C = SortPhaseCost + MergePhaseCost

  $= 2\times N_{pag} + 2\times N_{pag}\times$NoMergePasses
- After each merge pass, the number of runs:
  - $\lceil S/Z \rceil, \lceil S/Z^2 \rceil, \ldots, \lceil S/Z^k \rceil = 1$
  - $k = \lceil \log_Z S \rceil$
  - $C = 2\times N_{pag} + 2\times N_{pag}\times \lceil \log_Z S \rceil$
  - $= 2 \times N_{pag} \times (1 + \lceil \log_Z (N_{pag}/B) \rceil)$
  - $\sim 2(N_{pag}\times \log_Z (N_{pag}))$
- Runs may be longer than B

Summary

- Heap organization as DBMSs default
- Sequential organization is not really used by DBMSs
- External sorting is important, we are still improving
- External merge-sort minimizes disk I/O cost
- Choice of internal sort algorithm matters:
  - Quick sort (quick)
  - Replacement sort, natural selection (2x slower, but 2x longer runs)