

# CS150A Database

Lu Sun

School of Information Science and Technology

ShanghaiTech University

Sept. 20, 2022

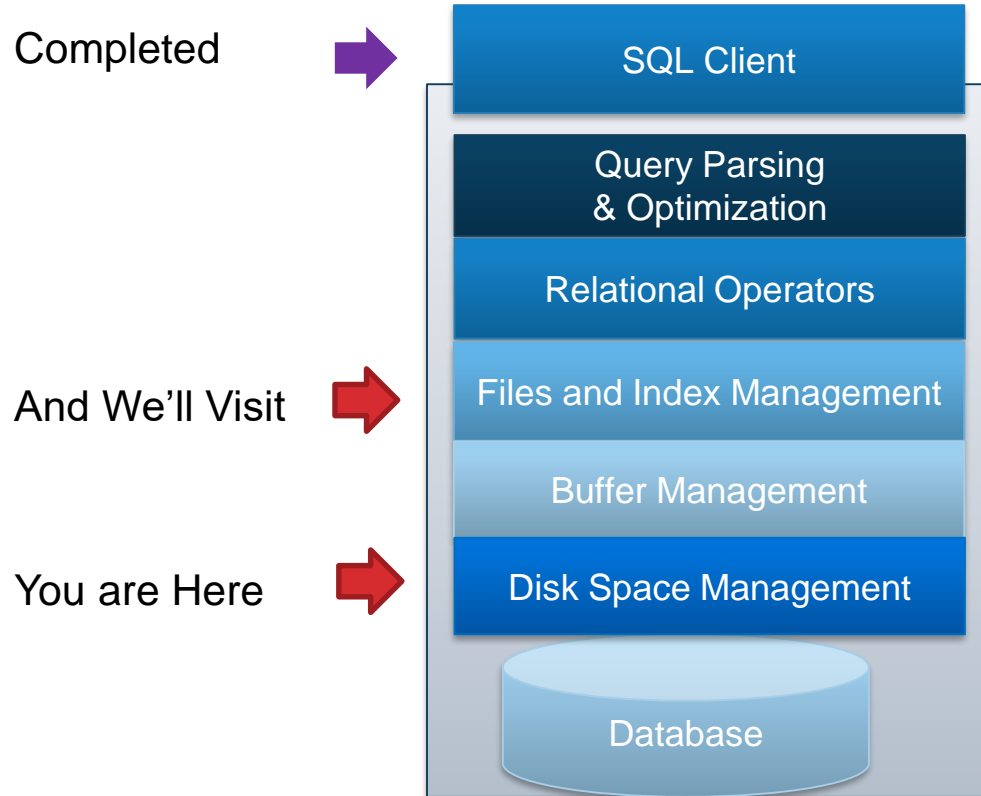
Today:

- Cost Model and Analysis:
  - In Heap File and Sorted File

Readings:

- Database Management Systems (DBMS), Chapters 8 and 9

# Architecture of a DBMS



# Recall: Heap Files

- Unordered collection of records
- Recall API for higher layers of the DBMS.  
Today we'll ask: "How? At what cost?"
  - Insert/delete/modify record
  - Fetch a particular record by ***record id*** ...
    - Record id is a pointer encoding pair of (**pageID**, **location** on page)
  - Scan all records
    - Possibly with some conditions on the records to be retrieved

# Recall: Multiple File Organizations

Many alternatives exist, each good in some situations and less so in others.

This is a theme in DB systems work!

- **Heap Files:** Suitable when typical access is a full scan of all records
- **Sorted Files:** Best for retrieval in order, or when a range of records is needed
- **Clustered Files & Indexes:** Group data into blocks to enable fast lookup *and* efficient modifications.
  - More on this soon ...

# Bigger Questions

- What is the “best” file organization?
  - Depends on access patterns ...
  - How? What are common access patterns anyway?
- Can we be quantitative about tradeoffs?
  - If one is better ... by how much?

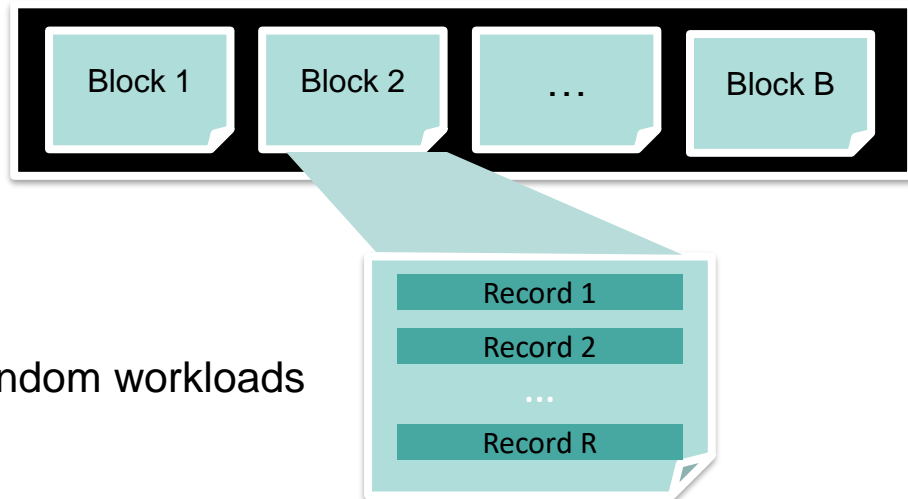
# Goals

- Big picture overheads for data access
  - We'll (overly) simplify performance models to provide insight, not to get perfect performance
  - Still, a bit of discipline:
    - **Clearly identify assumptions up front**
    - **Then estimate cost in a principled way**
- Foundation for query optimization
  - Can't choose the fastest scheme without an estimate of speed!

# **COST MODEL AND ANALYSIS**

# Cost Model for Analysis

- **B**: The number of data blocks in the file
- **R**: Number of records per block
- **D**: (Average) time to read/write disk block
- Focus: Average case analysis for uniform random workloads
- For now, we will ignore
  - Sequential vs Random I/O
  - Pre-fetching
  - Any in-memory costs
- Good enough to show the overall trends





# More Assumptions

- **Single record** insert and delete
- Equality selection – **exactly one match**
- For Heap Files:
  - Insert always **appends to end of file.**
- For Sorted Files:
  - **Packed:** Files compacted after deletions.
  - Sorted according to search key

# Extra Challenge

- After understanding these slides ...
  - You should question all these assumptions and rework
  - Good exercise to study for tests, and generate ideas

# Heap Files & Sorted Files

Heap File



Sorted File



For illustration, records are just integers

- **B:** The number of data blocks = 5
- **R:** Number of records per block = 2
- **D:** (Average) time to read/write disk block = 5ms

# Cost of Operations: Scan?

	Heap File	Sorted File
Scan all records		
Equality Search		
Range Search		
Insert		
Delete		

- **B:** The number of data blocks = 5
- **R:** Number of records per block = 2
- **D:** (Average) time to read/write disk block = 5ms

# Scan All Records

Heap File



Sorted File



- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block
- **Pages touched:** ?
- **Time to read the record:** ?

# Cost of Operations: Scan Cost

	Heap File	Sorted File
Scan all records	$B \cdot D$	$B \cdot D$
Equality Search		
Range Search		
Insert		
Delete		

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk bloc

# Cost of Operations: Equality Search?

	Heap File	Sorted File
Scan all records	$B \cdot D$	$B \cdot D$
Equality Search		
Range Search		
Insert		
Delete		

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block

# Find Key 8: Heap File

Heap File



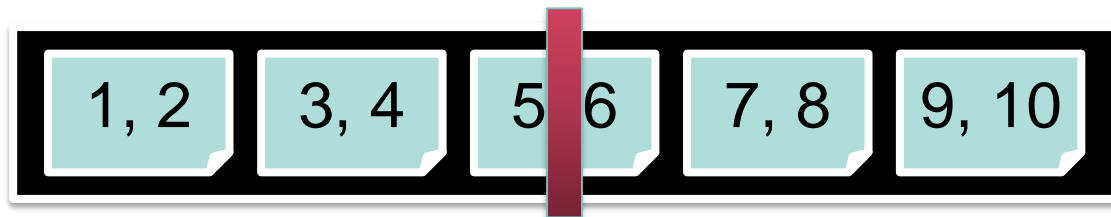
- **P(i)**: Probability that key is on page *i* is  $1/B$
- **T(i)**: Number of pages touched if key on page *i* is *i*
- Therefore the expected number of pages touched
- **Pages touched on average?**

$$\sum_{i=1}^B T(i)P(i) = \sum_{i=1}^B i \frac{1}{B} = \frac{B(B+1)}{2B} \approx \frac{B}{2}$$



# Find Key 8: Sorted File

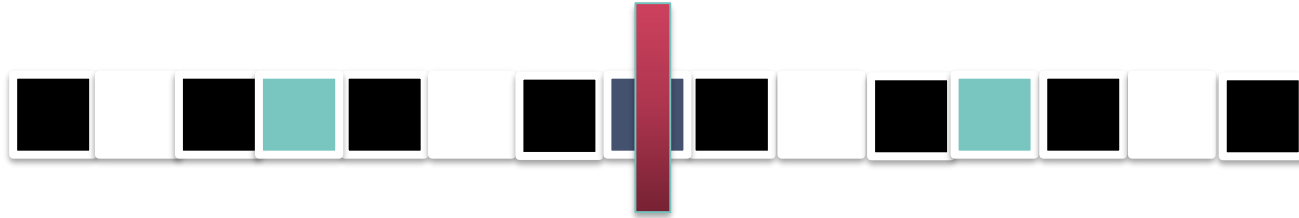
Sorted File



- **Worst-case:** Pages touched in binary search
  - $\log_2 B$
- **Average-case:** Pages touched in binary search
  - $\log_2 B?$

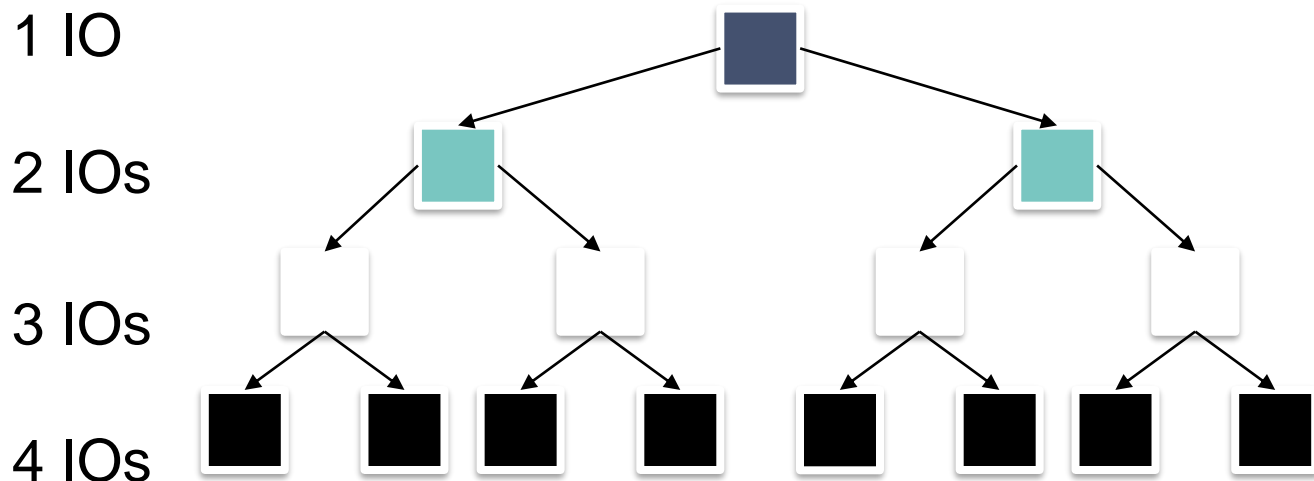
# Average Case Binary Search

**Expected Number of Reads:  $1 (1 / B) + 2 (2 / B) + 3 (4 / B) + 4 (8 / B)$**



# Average Case Binary Search cont

**Expected Number of Reads:  $1 (1 / B) + 2 (2 / B) + 3 (4 / B) + 4 (8 / B)$**



$$\sum_{i=1}^{\log_2 B} i \frac{2^{i-1}}{B} = \frac{1}{B} \sum_{i=1}^{\log_2 B} i 2^{i-1} = \log_2 B - \frac{B-1}{B}$$

# Cost of Operations: Equation Search Cost

	Heap File	Sorted File
Scan all records	$B \cdot D$	$B \cdot D$
Equality Search	$0.5 \cdot B \cdot D$	$(\log_2 B) \cdot D$
Range Search		
Insert		
Delete		

- **B**: The number of data blocks
- **R**: Number of records per block
- **D**: Average time to read/write disk block

# Cost of Operations: Range Search?

	Heap File	Sorted File
Scan all records	$B * D$	$B * D$
Equality Search	$0.5 * B * D$	$(\log_2 B) * D$
Range Search		
Insert		
Delete		

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block

# Find Keys Between 7 and 9: Heap File

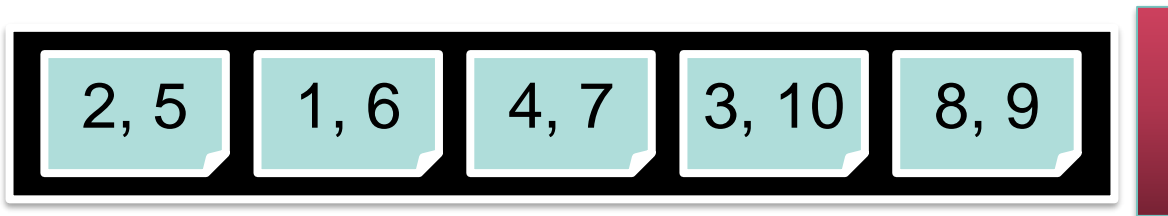
Heap File



- Always touch all blocks. Why?

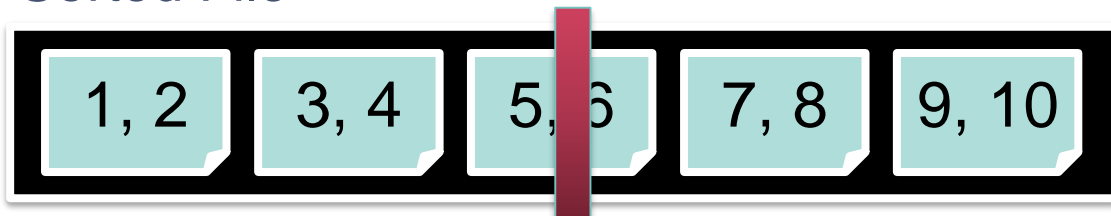
# Find Keys Between 7 and 9: Comparison

## Heap File



- Find beginning of range

## Sorted File



- Search for start of range
- Scan right

# Cost of Operations: Range Search Cost

	Heap File	Sorted File
Scan all records	$B * D$	$B * D$
Equality Search	$0.5 * B * D$	$(\log_2 B) * D$
Range Search	$B * D$	$((\log_2 B) + \text{pages}) * D$
Insert		
Delete		

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block



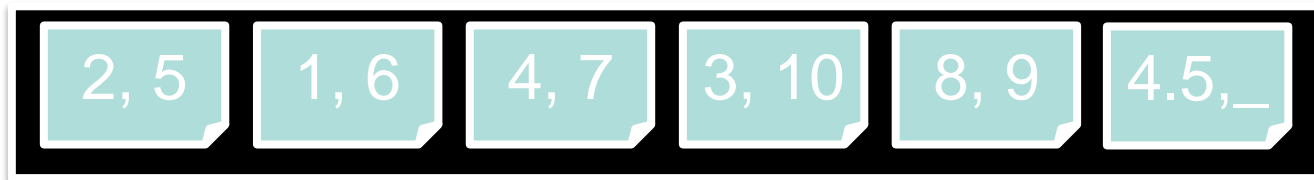
# Cost of Operations: Insert?

	Heap File	Sorted File
Scan all records	$B * D$	$B * D$
Equality Search	$0.5 * B * D$	$(\log_2 B) * D$
Range Search	$B * D$	$((\log_2 B) + \text{pages}) * D$
Insert		
Delete		

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block

# Insert 4.5: Heap File

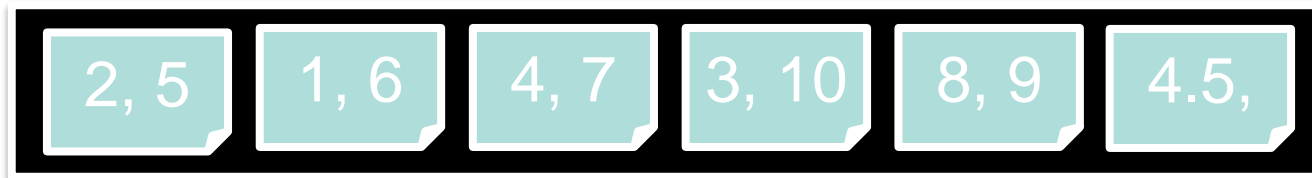
Heap File



- Stick at end of file
- Cost =  $2 \cdot D$
- Why 2?

# Insert 4.5: Heap VS Sorted File

## Heap File



- Read last page, append, write. Cost =  $2 \cdot D$

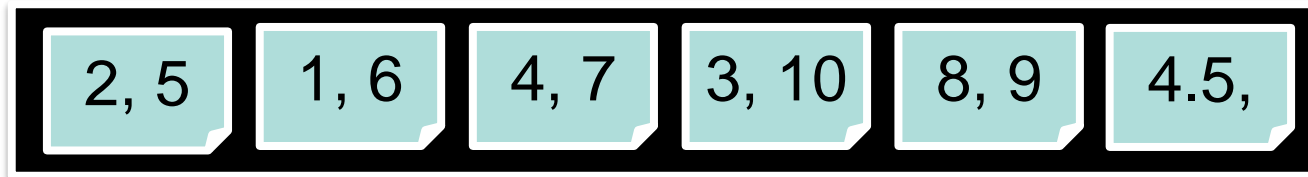
## Sorted File



- Find location for record. Cost =  $\log_2 BD$

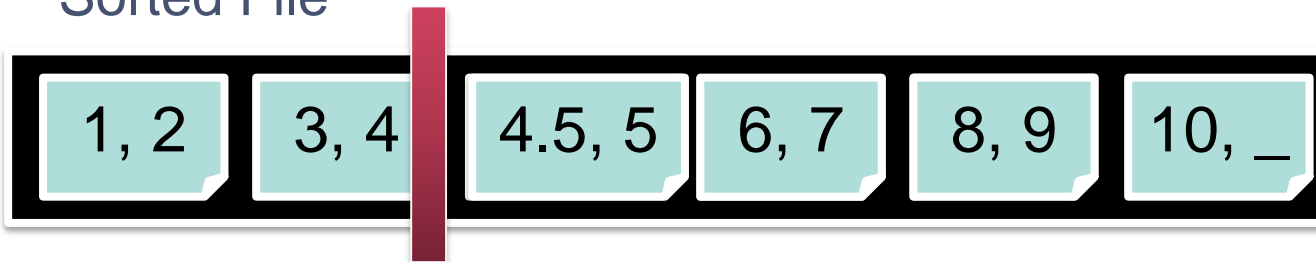
# Insert 4.5: Heap Vs Sorted Pt 2

## Heap File



- Read last page, append, write. Cost =  $2 \cdot D$

## Sorted File



- Find location for record. Cost =  $\log_2 BD$
- Insert and shift rest of file

# Cost of Operations: Insert Cost

	Heap File	Sorted File
Scan all records	$B * D$	$B * D$
Equality Search	$0.5 * B * D$	$(\log_2 B) * D$
Range Search	$B * D$	$((\log_2 B) + \text{pages}) * D$
Insert	$2 * D$	$((\log_2 B) + B) * D$
Delete		

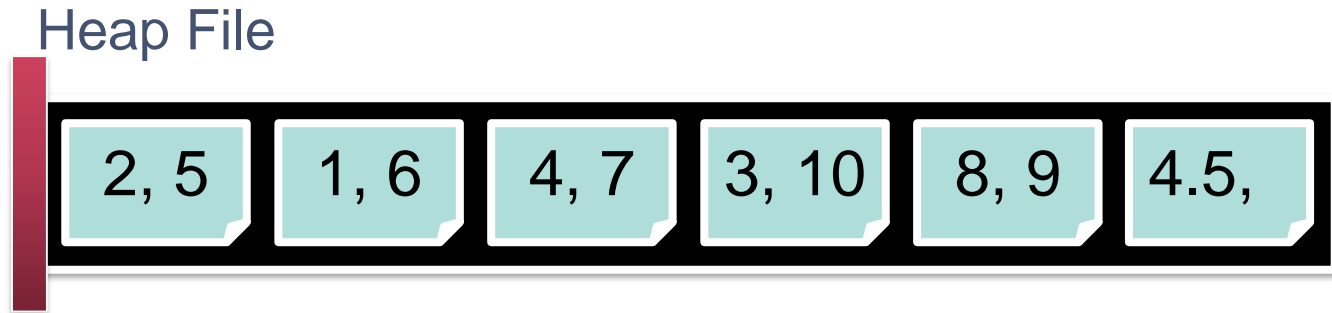
- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block

# Cost of Operations: Delete?

	Heap File	Sorted File
Scan all records	$B * D$	$B * D$
Equality Search	$0.5 * B * D$	$(\log_2 B) * D$
Range Search	$B * D$	$\frac{((\log_2 B) + \text{pages}) * D}{D}$
Insert	$2 * D$	$((\log_2 B) + B) * D$
Delete		

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block

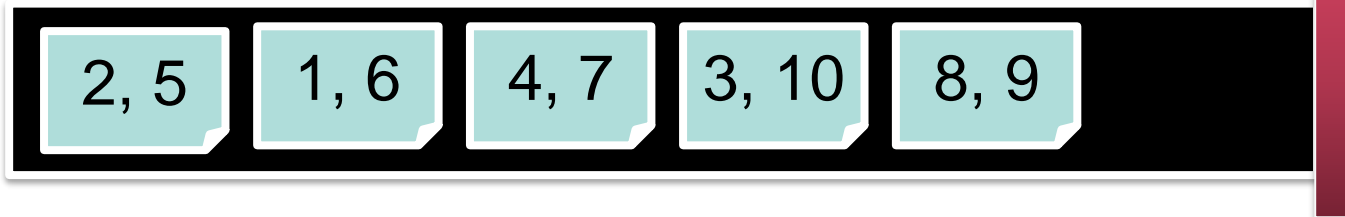
# Delete 4.5: Heap File



- Average case to find the record:  **$B/2$  reads**
- Delete record from page
- Cost =  $(B/2 + 1) * D$ 
  - Why + 1?

# Delete 4.5: Heap File Vs Sorted File

## Heap File



- Average case runtime:  $(B/2+1) * D$

## Sorted File

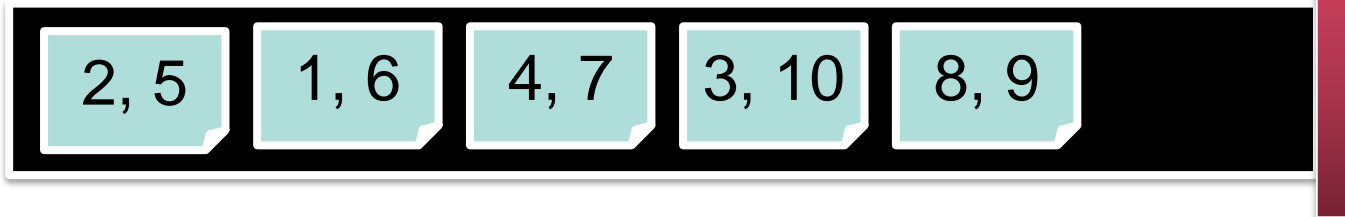


- Find location for record:  $\log_2 B$
- Delete record in page  $\rightarrow$  Gap



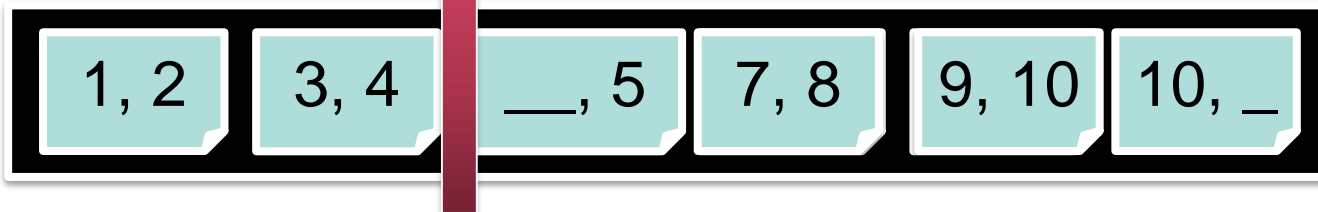
# Delete 4.5: Heap File Vs Sorted File Pt 2

## Heap File



- Average case runtime:  $(B/2+1) * D$

## Sorted File



- Find location for record:  $\log_2 B$
- Shift the rest by 1 record  $2 * (B/2)$

# Cost of Operations Complete

	Heap File	Sorted File
Scan all records	$B * D$	$B * D$
Equality Search	$0.5 * B * D$	$(\log_2 B) * D$
Range Search	$B * D$	$((\log_2 B) + \text{pages}) * D$
Insert	$2 * D$	$((\log_2 B) + B) * D$
Delete	$(0.5 * B + 1) * D$	$((\log_2 B) + B) * D$

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block

# Cost of Operations Complete Pt 2

	Heap File	Sorted File
Scan all records	$B * D$	$B * D$
Equality Search	$0.5 * B * D$	$(\log_2 B) * D$
Range Search	$B * D$	$\frac{((\log_2 B) + \text{pages}) * D}{D}$
Insert	$2 * D$	$((\log_2 B) + B) * D$
Delete	$(0.5 * B + 1) * D$	$((\log_2 B) + B) * D$

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** Average time to read/write disk block
- Can we do better?
  - Indexes!