Storing Data: Disk & Files

CS 377: Database Systems

Recap: BCNF Normalization

- Relation R(A, B, C, D, E, F, G, H, I, J, K, L, M)
  - A —> B, C, D, E
  - E —> F, G, H
  - I —> J
  - A, I —> K
  - A, L —> M

Recap: BCNF Normalization (2)

- Step 1: Find all keys
  - Use heuristic #2 to find that A, I, L are not in RHS
  - $\{A, I, L\}^+ = A, I, L, B, C, D, E, F, G, H, J, K, M$
    - (A, I, L) is key
- Step 2: Check for BCNF condition
  - A —> B, C, D, E: violation FD as A is subset of key

Recap: BCNF Normalization (3)

- Step 2(c): Decompose using A^+
  - $\{A\}^+$ = A, B, C, D, E, F, G, H
  - R1(<u>A</u>, B, C, D, E, F, G, H)
    R2(<u>A</u>, <u>I</u>, J, K, <u>L</u>, M)
- Step 3: Check R1 for BCNF condition
  - E —> F, G, H: violation as E is not a super key

Recap: BCNF Normalization (4)

- Step 3(c): Decompose using E^+
  - $\{E\}^+$ = E, F, G, H
  - R11(<u>A</u>, B, C, D, E)
    R12(<u>E</u>, F, G, H)
- Step 4: Check R11 for BCNF condition
  - OK!
- Step 5: Check R12(<u>E</u>, F, G, H) for BCNF
  - OK!

Recap: BCNF Normalization (5)

- Step 6: Check R2(A, I, J, K, L, M) for BCNF
  - I —> J: violation as I is subset of key
  - Decompose using $\{I\}^+ = I, J$
    - R21(I, J)
      R2(A, I, K, L, M)
- Step 7: Check R21 for BCNF condition
  - OK!

Recap: BCNF Normalization (6)

- Step 8: Check R2(A, I, K, L, M) for BCNF condition
  - A, I —> K: violation as (A,I) is not superkey
  - Decompose using $\{A, I\}^+ = A, I, K$
    - R221(A, I, K)
      R222(A, I, L, M)
- Step 9: Check R221 for BCNF condition
  - OK!

Recap: BCNF Normalization (7)

- Step 10: Check R222(A, I, L, M) for BCNF condition
  - A, L —> M: violation as (A,L) is not superkey
  - Decompose using $\{A, L\}^+ = A, L, M$
    - R2221(A, L, M)
      R2222(A, I, L)
- Step 11: Check R2221 for BCNF condition
  - OK!

Recap: BCNF Normalization (8)

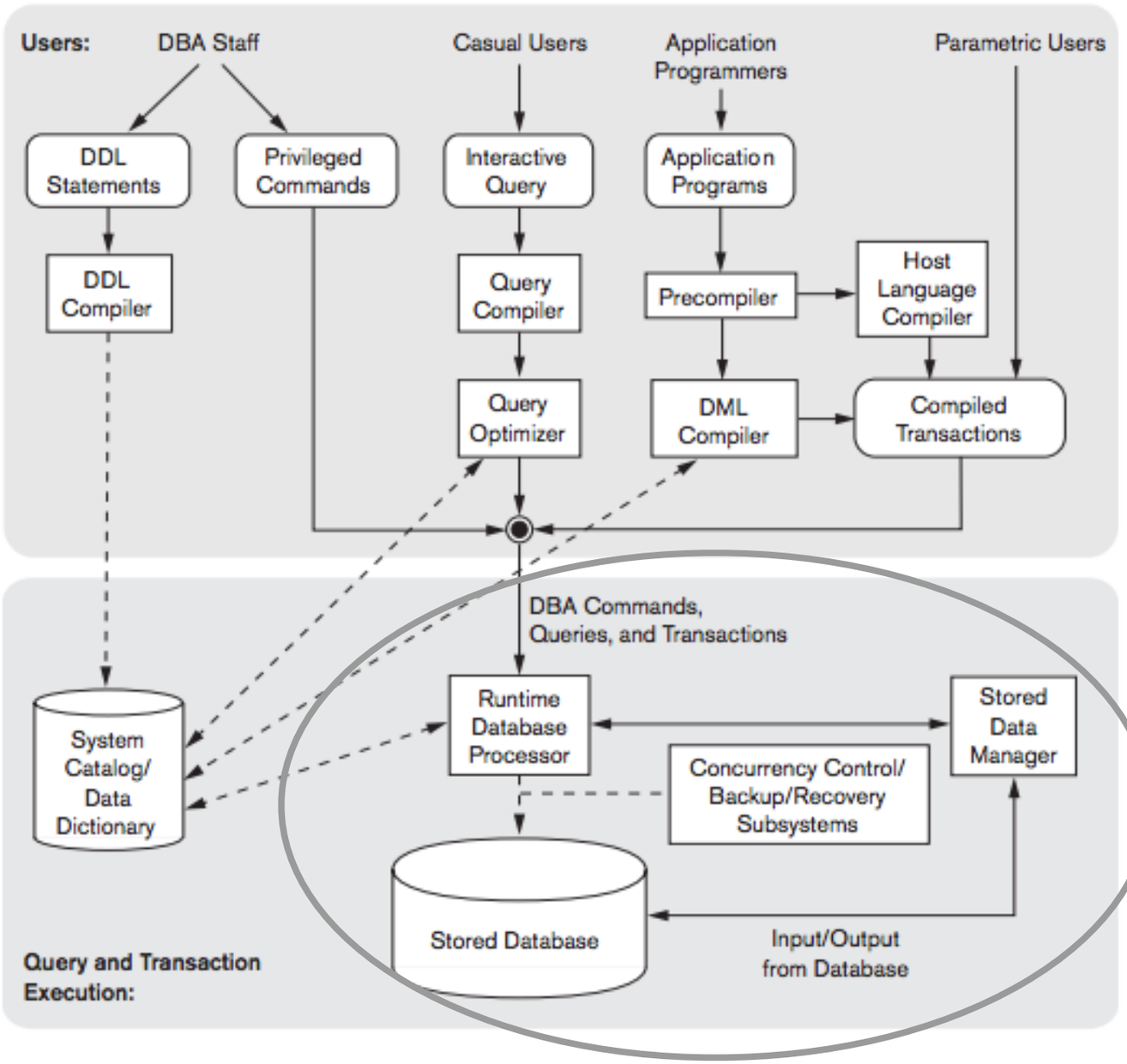
- Step 12: Check R2222(A, I, L) for BCNF condition
  - OK!
- Final Decomposition:

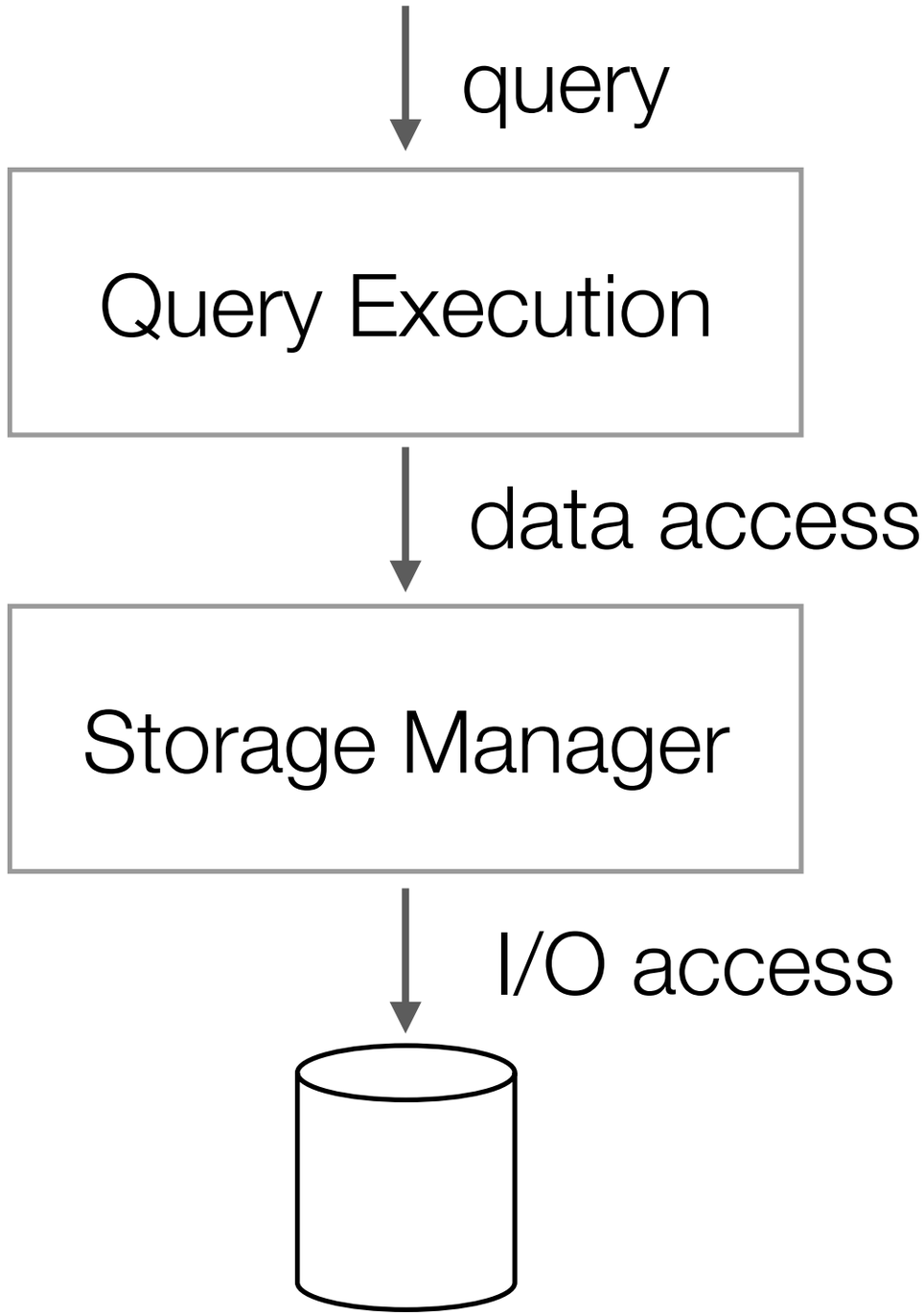
R11(A, B, C, D, E)
R12(E, F, G, H)
R21(I, J)
R221(A, I, K)
R2221(A, L, M)
R2222(A, I, L)

Final Note on Normalization

- Is 3NF or BCNF better?
  - 3NF can be lossless and preserves all functional dependencies
  - BCNF is guaranteed to be lossless but may not preserve all functional dependencies
- Ultimate goal:
  - BCNF, lossless, preserves all functional dependencies
- Next ultimate goal:
  - 3NF, lossless, preserves all functional dependencies

DMBS Architecture

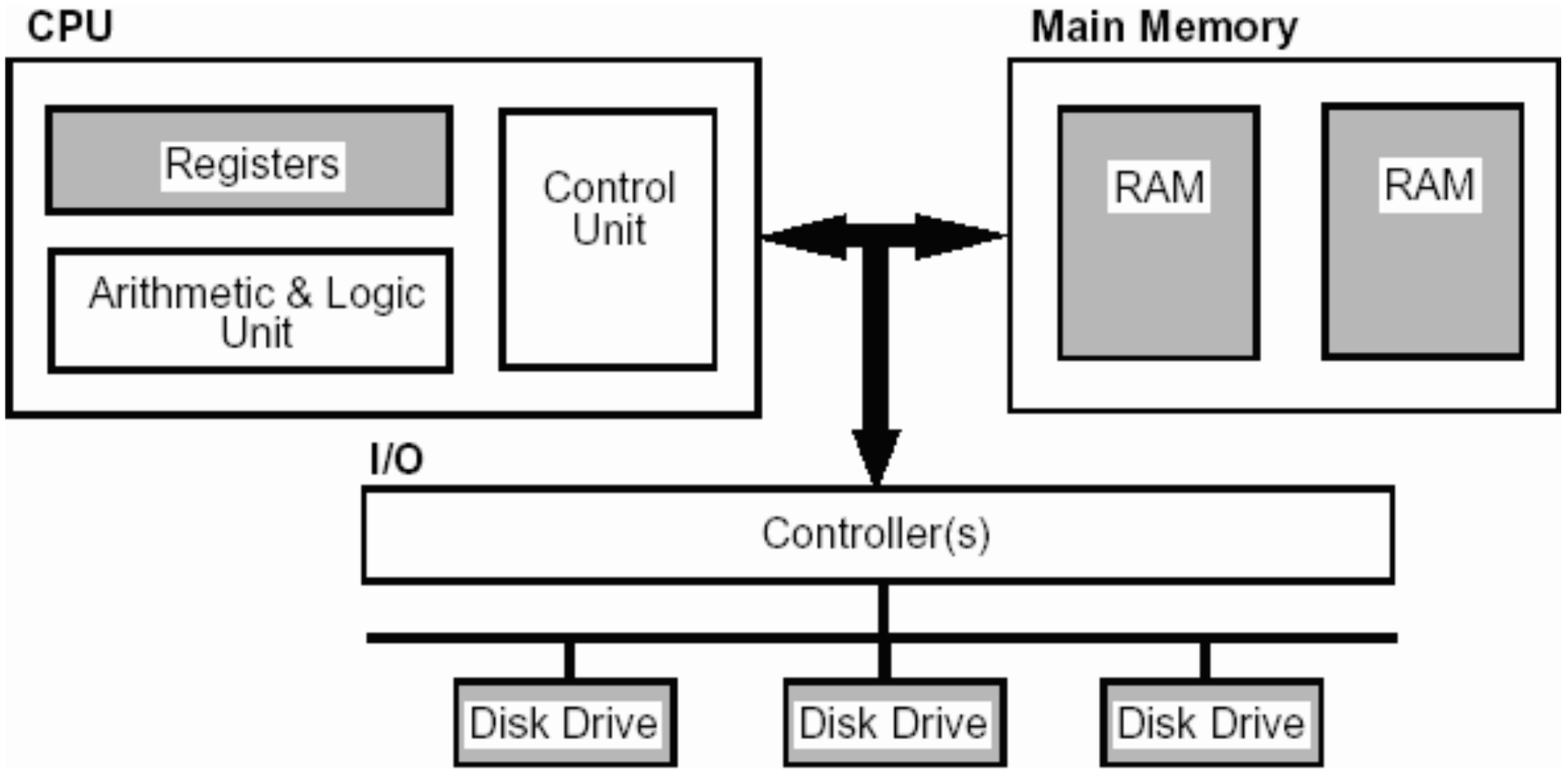




Data Storage

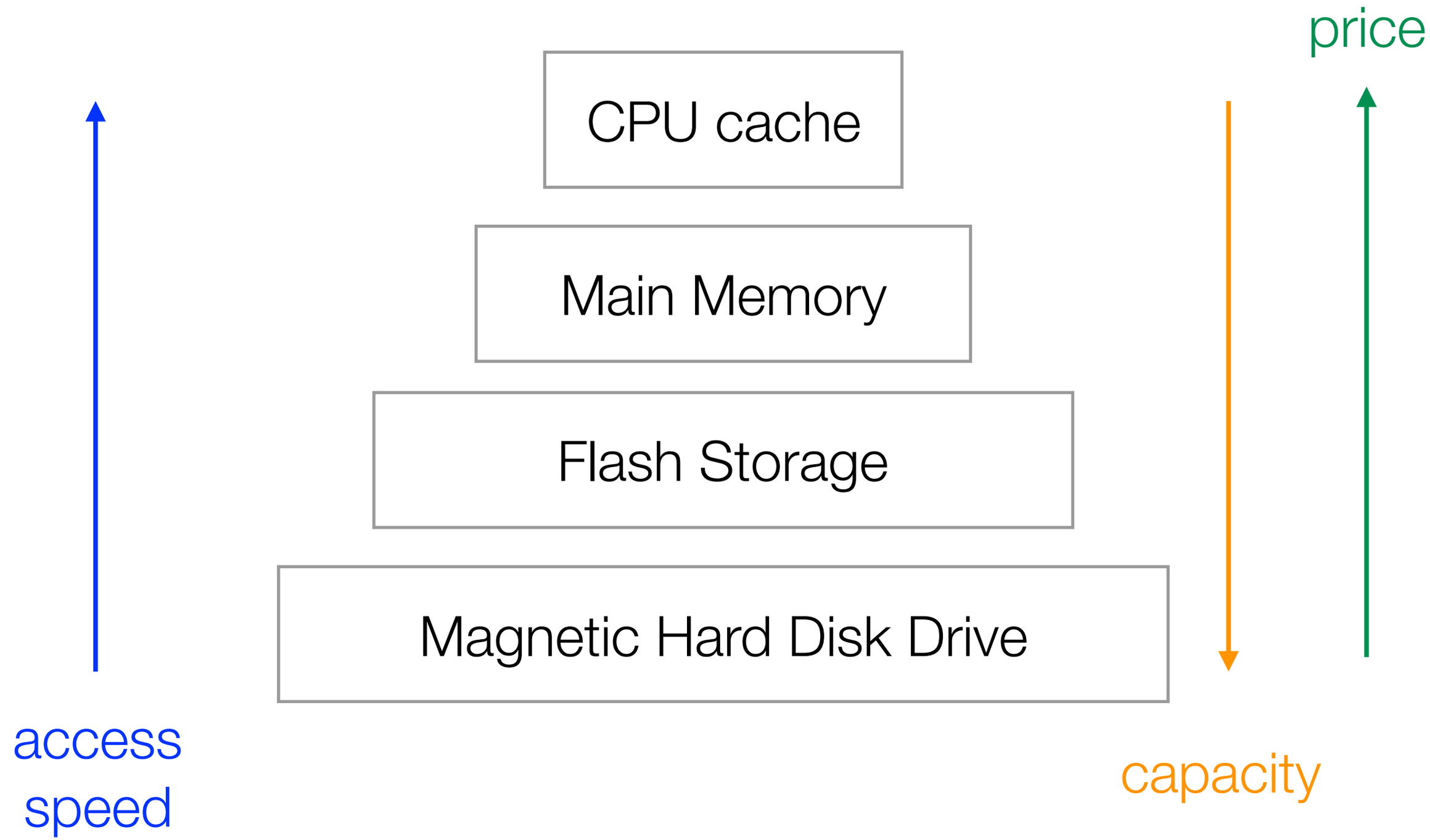
- How does a DBMS store and access data?
  - Disk
  - Main memory
- How do we move data from disk to main memory?
  - Blocks
- How do we organize relational data into files?

Computer System Overview



http://www.doc.ic.ac.uk/~eedwards/compsys/memory/memory.gif

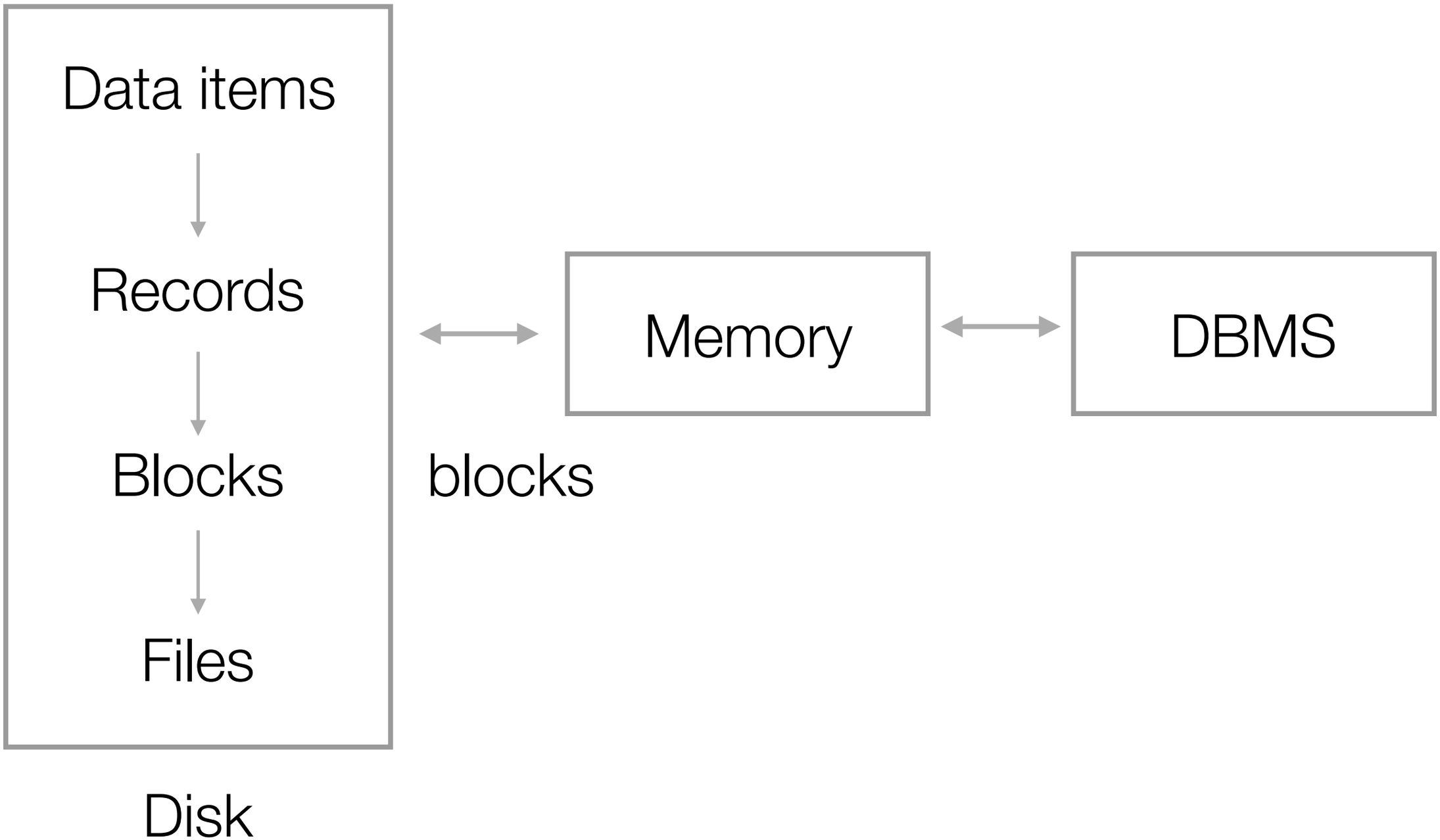
Memory Hierarchy



Typical Storage Hierarchy

- Main memory (RAM) for currently used data
  - Not persistent
  - Relatively high cost
- Disk for main database (secondary storage)
- Tapes for archiving older versions of the data (tertiary storage)

Data Store Overview



Hard Disks

- Data is stored and retrieved in units called disk blocks or pages
  - Typical numbers these days are 64 KB per block
- Retrieval time depends upon the location of the disk
  - Placement of blocks on disk has major impact on DBMS performance

Disks are a mechanical anachronism!

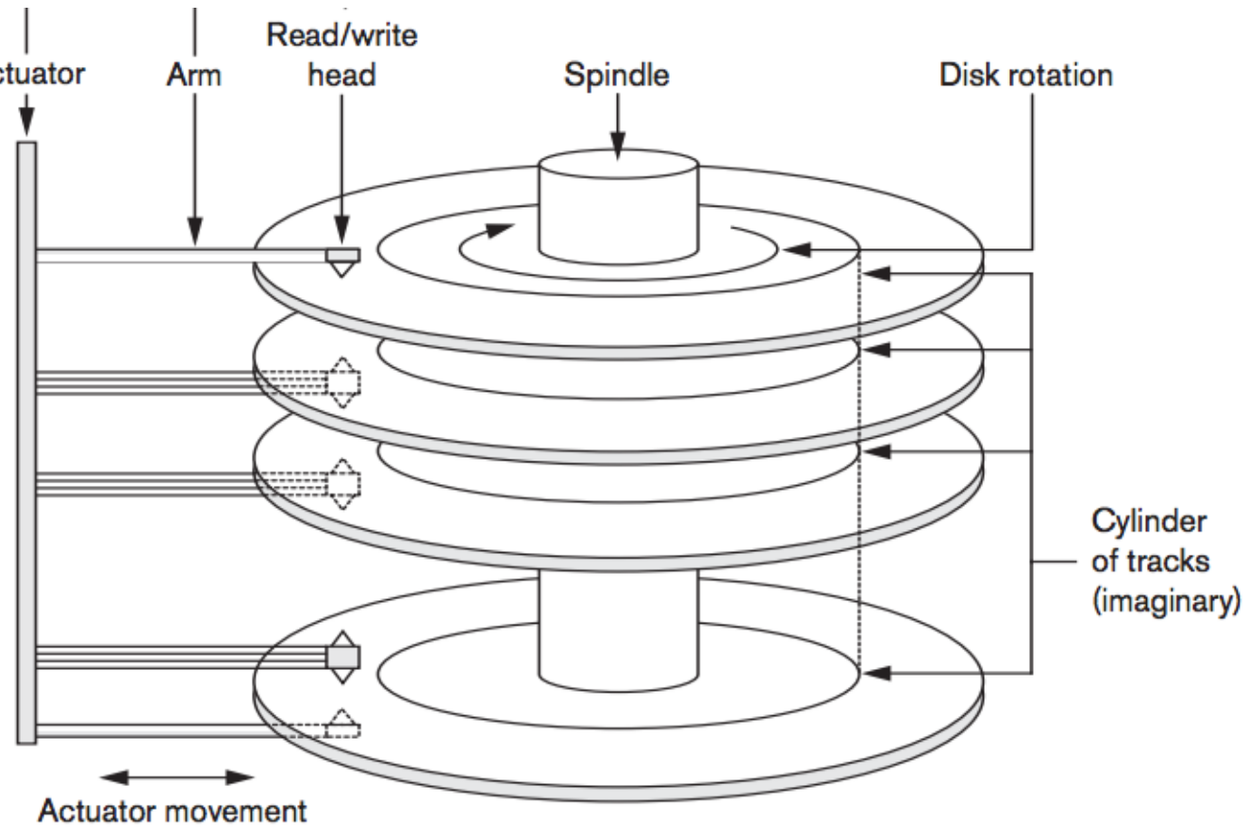
Components of Disk

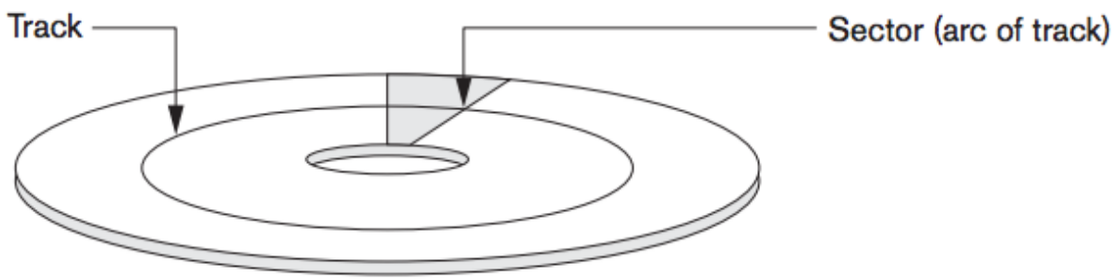
- Data is encoded in concentric circles of sectors called tracks

- Disk head: mechanism to read / write data

- Boom (disk arm) moves to position disk head on the desired track

- Exactly one head reads/ writes at any time





Block Access

- Time to access (read/write) a page
  - Seek time: move arms to position disk head on track
  - Rotational delay: wait for page to rotate under head
  - Transfer time: move data to/from disk surface
- Seek time and rotational delay are dominant factors
  - Seek time ~ 0 to 10 ms
  - Rotational delay ~ 0 to 10 ms
  - Transfer rate: ~100 MB / s

Disk Access Situations

- Random access: collection of short processes that execute in parallel, share the same disk, and cannot be predicted in advance
  - Very expensive I/O
- Sequential access: blocks are accessed in a sequence that can be predicted (e.g., accessing all the records in a single relation)
  - Much less expensive I/O

Example: Disk Specifications

	Seagate HDD
Capacity	3TB
RPM	7,200
Average Seek Time	9 ms
Max Transfer Rate	210 MB/s
# Platters	3

- What are I/O rates for block size of 4 KB?
  - Random workload: ~0.3 MB/s
  - Sequential workload: ~210 MB/s

Speeding up Disk Access

- Blocks in a file should be arranged sequentially on disk to minimize seek and rotational delay
- 'Next' block concept
  - Blocks on same track
  - Blocks on same cylinder
  - Blocks on adjacent cylinder
- For sequential scan, pre-fetch several blocks at a time!

Records

- Records contain fields which have values of a particular type (e.g., amount, date, time, age)

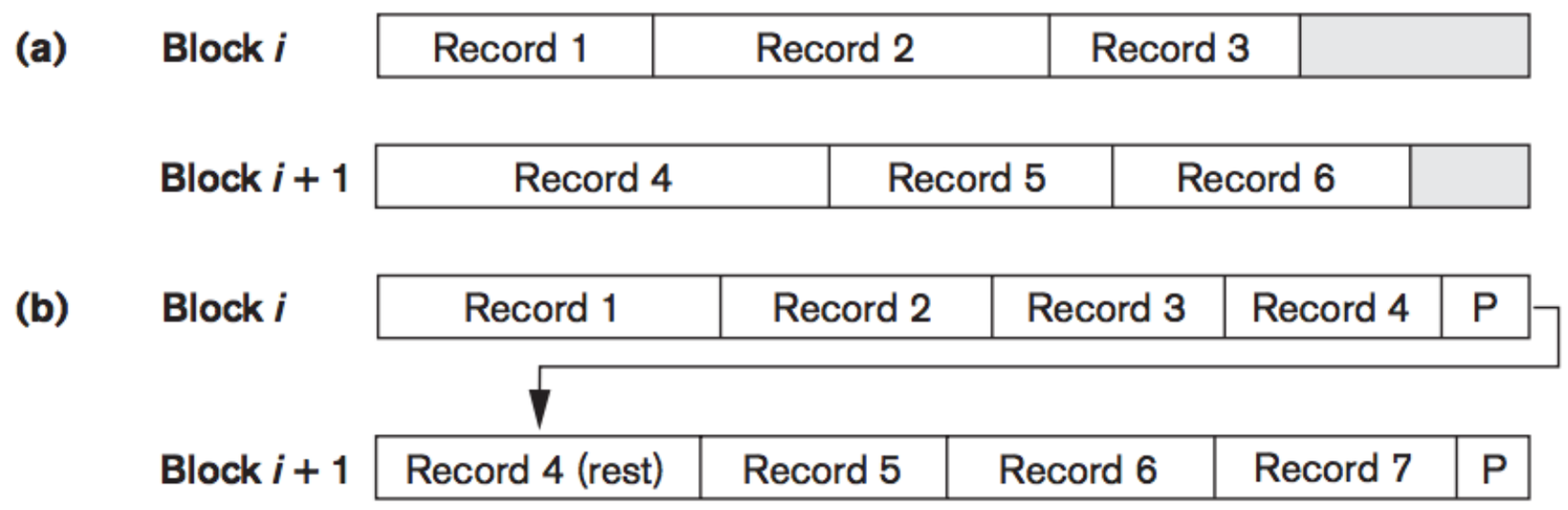
- Fields themselves may be fixed length or variable length

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

Blocks

Blocks contain records

- Unspanned: records must be within one block, simple but can lead to unused space
- Spanned: record size can be larger than block size, pointer to rest of record



Files

- Disk space is organized into files
- Files consist of blocks (pages)
- Blocks consist of records
- Organization of records in files
  - Heap
  - Ordered (sequential)

Unordered (Heap) Files

- Contains records in no particular order
- New records are inserted at the end of the file
- Insert: very efficient, last disk block of file is copied into buffer, add new record, and rewrite back onto disk
- Linear search: O(b)
- Reading the records in order of a particular filed requires sorting the file records

Ordered (Sequential) File

- File whose records are sorted by some attribute (usually its primary key)
- Search: binary search in $O(\log_2(b))$
- Insert: more expensive to keep records in ordered file
- Reading the records in order of the ordering field is quite efficient

	Name	Ssn	Birth_date	Job	Salary	Sex
Block 1	Aaron, Ed					
	Abbott, Diane					
	⋮					
	Acosta, Marc					
Block 2	Adams, John					
	Adams, Robin					
	⋮					
	Akers, Jan					
Block 3	Alexander, Ed					
	Alfred, Bob					
	⋮					
	Allen, Sam					
Block 4	Allen, Troy					
	Anders, Keith					
	⋮					
	Anderson, Rob					
Block 5	Anderson, Zach					
	Angeli, Joe					
	⋮					
	Archer, Sue					
Block 6	Arnold, Mack					
	Arnold, Steven					
	⋮					
	Atkins, Timothy					
	⋮					
Block *n*–1	Wong, James					
	Wood, Donald					
	⋮					
	Woods, Manny					
Block *n*	Wright, Pam					
	Wyatt, Charles					
	⋮					
	Zimmer, Byron					

Average Access Times

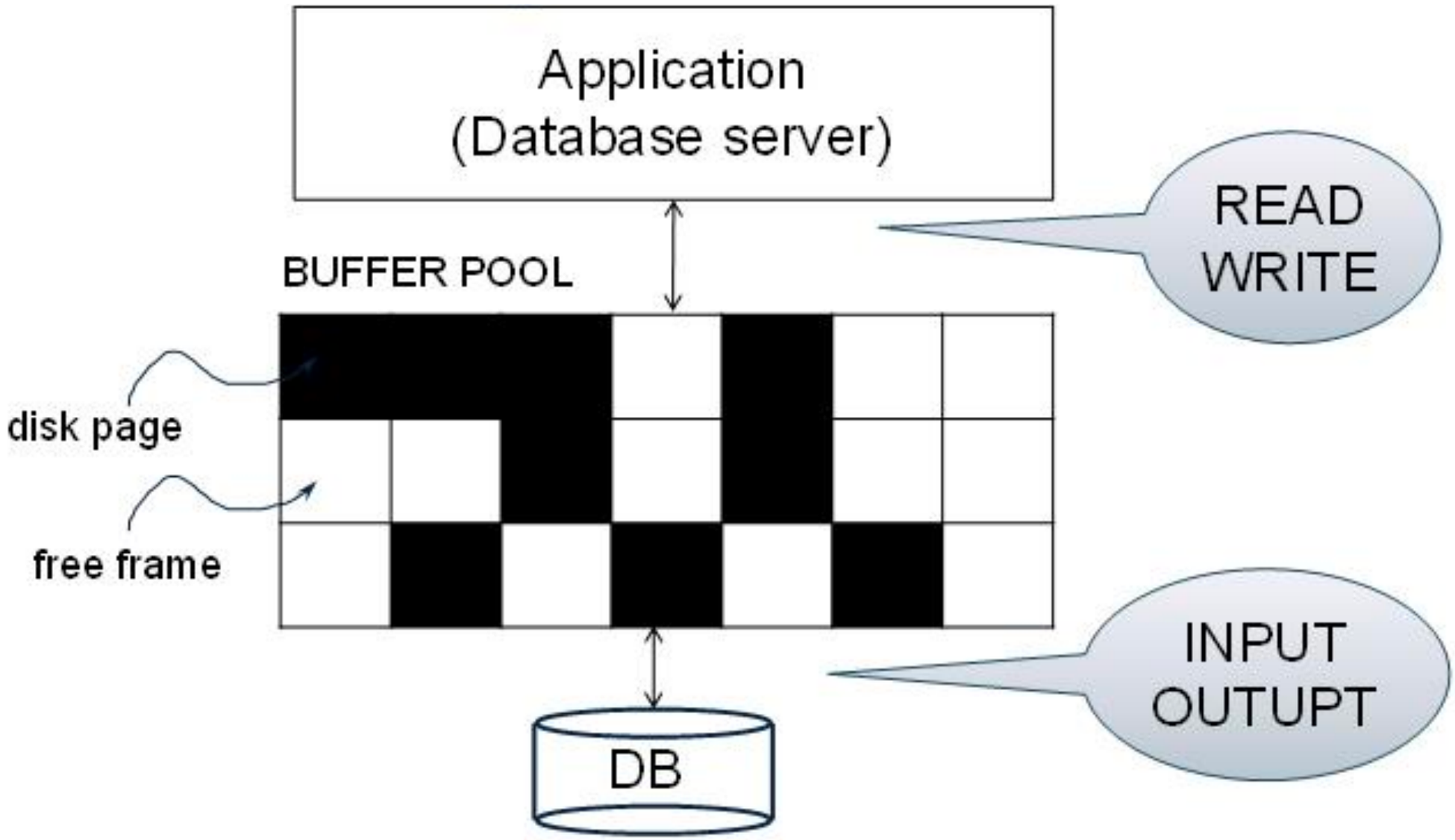
Table 17.2 Average Access Times for a File of *b* Blocks under Basic File Organizations

Type of Organization	Access/Search Method	Average Blocks to Access a Specific Record
Heap (unordered)	Sequential scan (linear search)	$b/2$
Ordered	Sequential scan	$b/2$
Ordered	Binary search	$\log_2 b$

Buffer Manager

- Data should be in RAM for DBMS to operate on it efficiently
- All pages may not fit into main memory
- Buffer manager is responsible for bringing blocks from disk to main memory as needed
  - Allocate space in the buffer if not exist (replace some other block to make space for new block)
  - Reads the block from disk to buffer

Buffer Manager (Pictorially)



http://courses.cs.washington.edu/courses/csep544/14wi/video/archive/html5/csep544_14wi_6/slide487.jpg

DBMS vs. OS File System

Why not let OS handle disk management and buffer management?

- DBMS better at predicting reference patterns
- Buffer management is necessary to implement concurrency control and recovery
- More control of the overlap of I/O with computation
- Leverage multiple disks more effectively

Data Storage: Recap

- How DBMS stores data
  - Disk, main memory
  - Files, blocks, records
    - Organization of records in files
- Buffer manager