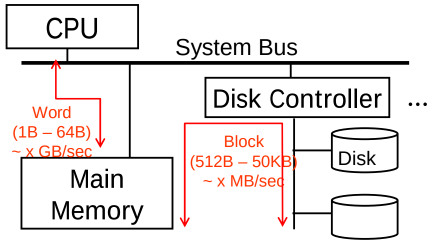
CS143: Disks and Files

System Architecture



CPU

System Bus

Disk Controller

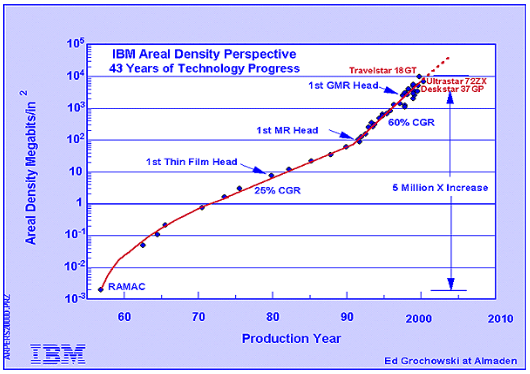
...

Word
(1B – 64B)
~ x GB/sec

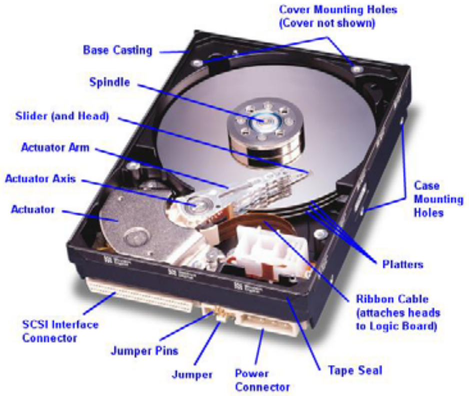
Main
Memory

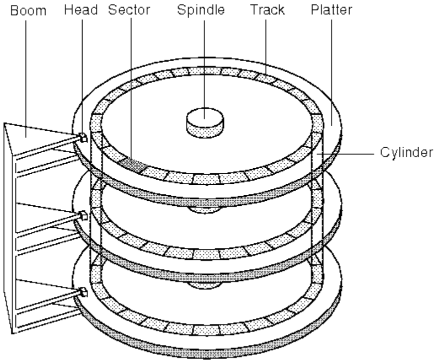
Block
(512B – 50KB)
~ x MB/sec

Disk

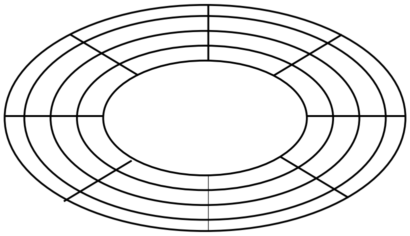


- Capacity keeps increasing, but speed does not





Structure of a Platter



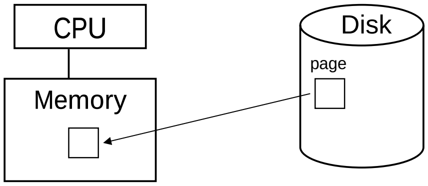
- Track, cylinder, sector (=block, page)

A Typical Disk

- Platter diameter: 1-5 in
- Cylinders: 100 – 2000
- Platters: 1 – 20
- Sectors per track: 200 – 500
- Sector size: 512 – 50K
- Rotation speed: 1000 – 15000 rpm
- Overall capacity: 1G – 300GB

- Q: 2 platters, 2 surfaces/platter, 500 tracks/surface, 200 sect/track, 1KB/sector. What is the overall capacity?

Access Time



CPU

Memory

Disk

page

- Q: How long does it take to read a page of a disk to memory?

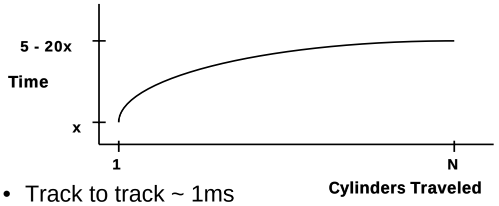
- Q: What needs to be done to read a page?

Access Time

- Access time =
  (seek time) + (rotational delay) +
  (transfer time)

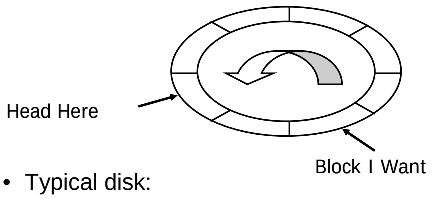
Seek Time

- Time to move a disk head between tracks



- Track to track ~ 1ms
- Average ~ 10 ms
- Full stroke ~ 20 ms

Rotational Delay

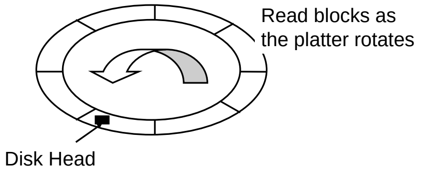


Head Here

Block I Want

- Typical disk:
  - 1000 rpm – 15000 rpm
- Q: For 6000 RPM, average rotational delay?

Transfer Rate



Read blocks as the platter rotates

Disk Head

6000 RPM, 400 sectors/track, 512B/sector

- Q: How long to read one block?

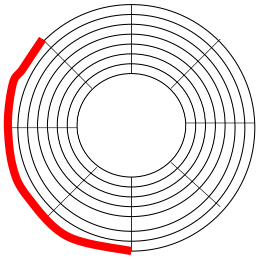
- Q: What is the transfer rate (bytes/sec)?

(Burst) Transfer Rate

- (Burst) Transfer rate =
  (RPM / 60) * (sectors/track) * (bytes/sector)

Sequential vs. Random I/O

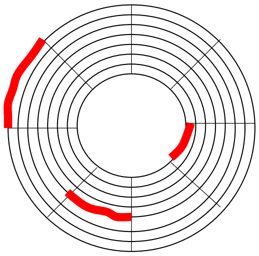
- Q: How long to read 3 sequential blocks?



□ 6000 RPM
□ 200 sectors/track
□ Assume the head is above the first block

Sequential vs. Random I/O

- Q: How long to read 3 random blocks?



□ 6000 RPM
□ 200 sectors/track
□ 10ms seek time
□ Assume the head is above the first block

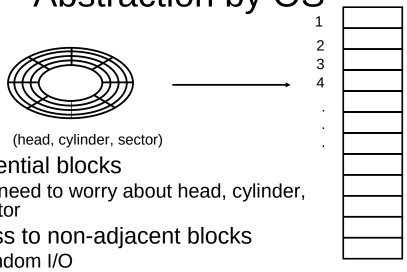
Important to remember

- Random I/O: VERY expensive
  - Compared to sequential I/O
- Avoid random I/O as much as we can

Data Modification

- Byte-level modification not allowed
  - Can be modified by blocks
- Q: How can we modify only a part of a block?

Abstraction by OS



- Sequential blocks
  - No need to worry about head, cylinder, sector
- Access to non-adjacent blocks
  - Random I/O
- Access to adjacent blocks
  - Sequential I/O

Buffers, Buffer pool

- Temporary main-memory “cache” for disk blocks
  - – Avoid future read
  - – Hide disk latency
  - – Most DBMS let users change buffer pool size

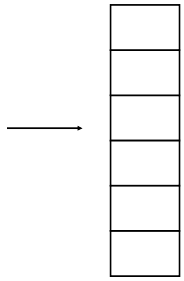
Reference

- Storage review disk guide
  - – http://www.storagereview.com/guide2000/ref/hdd/index.html

Files: Main Problem

- How to store tables into disks?

Jane	CS	3.7
Susan	ME	1.8
June	EE	2.6
Tony	CS	3.1

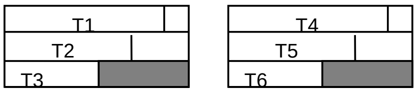


Spanned vs Unspanned

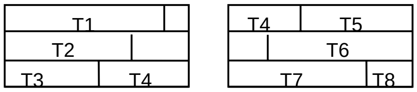
- Q: 512Byte block. 80Byte tuple. How to store?

Spanned vs Unspanned

- Unspanned



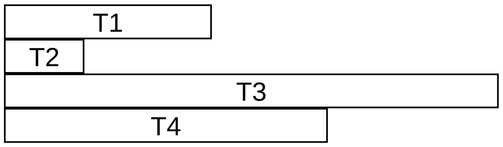
- Spanned



- Q: Maximum space waste for unspanned?

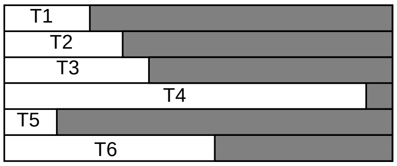
Variable-Length Tuples

- How do we store them?



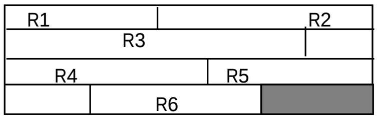
Reserved Space

- Reserve the maximum space for each tuple



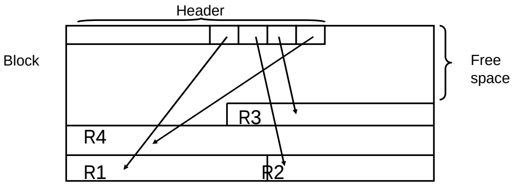
- Q: Any problem?

Variable-Length Space



- Pack tuples tightly
- Q: How do we know the end of a record?
- Q: What to do for delete/update?
- Q: How can we “point to” to a tuple?

Slotted Page



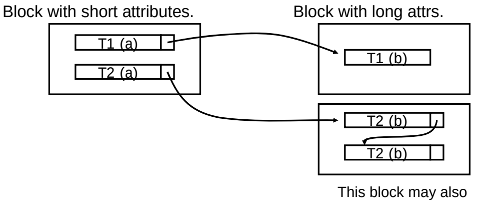
Q: How can we point to a tuple?

Long Tuples

- ProductReview(
  pid INT,
  reviewer VARCHAR(50),
  date DATE,
  rating INT,
  comments VARCHAR(1000))
- Block size 512B
- How should we store it?

Long Tuples

- Spanning
- Splitting tuples



This block may also have fixed-length slots.

Sequential File

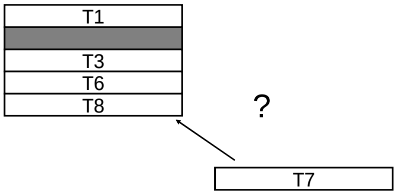
- Tuples are ordered by certain attribute(s) (search key)

Elaine	CS	3.7
James	ME	2.8
John	EE	1.8
Peter	EE	3.9
Susan	CS	1.0
Tony	EE	2.4

– Search key: Name

Sequencing Tuples

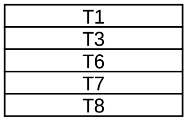
- Inserting a new tuple
  - – Easy case



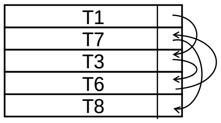
Sequencing Tuples

Two options

1) Rearrange

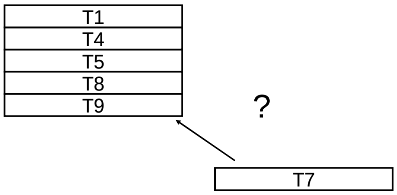


2) Linked list



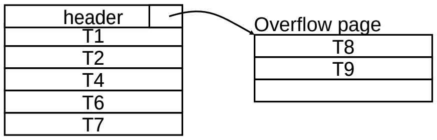
Sequencing Tuples

- Inserting a new tuple
  - – Difficult case



Sequencing Tuples

- Overflow page



- Reserving free space to avoid overflow
  - – PCTFREE in DBMS
    CREATE TABLE R(a int) PCTFREE 40

Sequencing Tuples Initially

- CREATE TABLE T …;
  INSERT INTO T
  (SELECT * … ORDER BY key);
- Future insertions will gradually destroy the order
  - – Periodic reordering may be necessary
- Other possibilities discussed later

Things to Remember

- Disk
  - Platter, track, cylinder, sector, block
  - Seek time, rotational delay, transfer time
  - Random I/O vs Sequential I/O
- Files
  - Spanned/unspanned tuples
  - Variable-length tuples (slotted page)
  - Long tuples
  - Sequential file and search key
    - Problems with insertion (overflow page)
    - PCTFREE