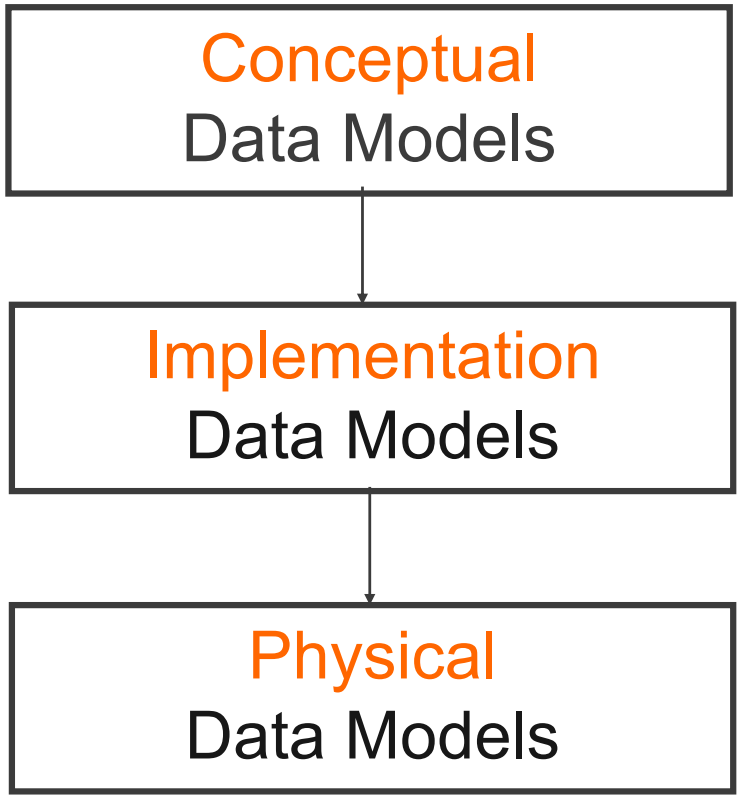
Disk Storage, Basic File Structures

政治大學
資訊科學系
沈錳坤

Content

- Secondary Storage Devices
- Placing File Records on Disk
- Operations on Files
- Primary File Organization

Data Models

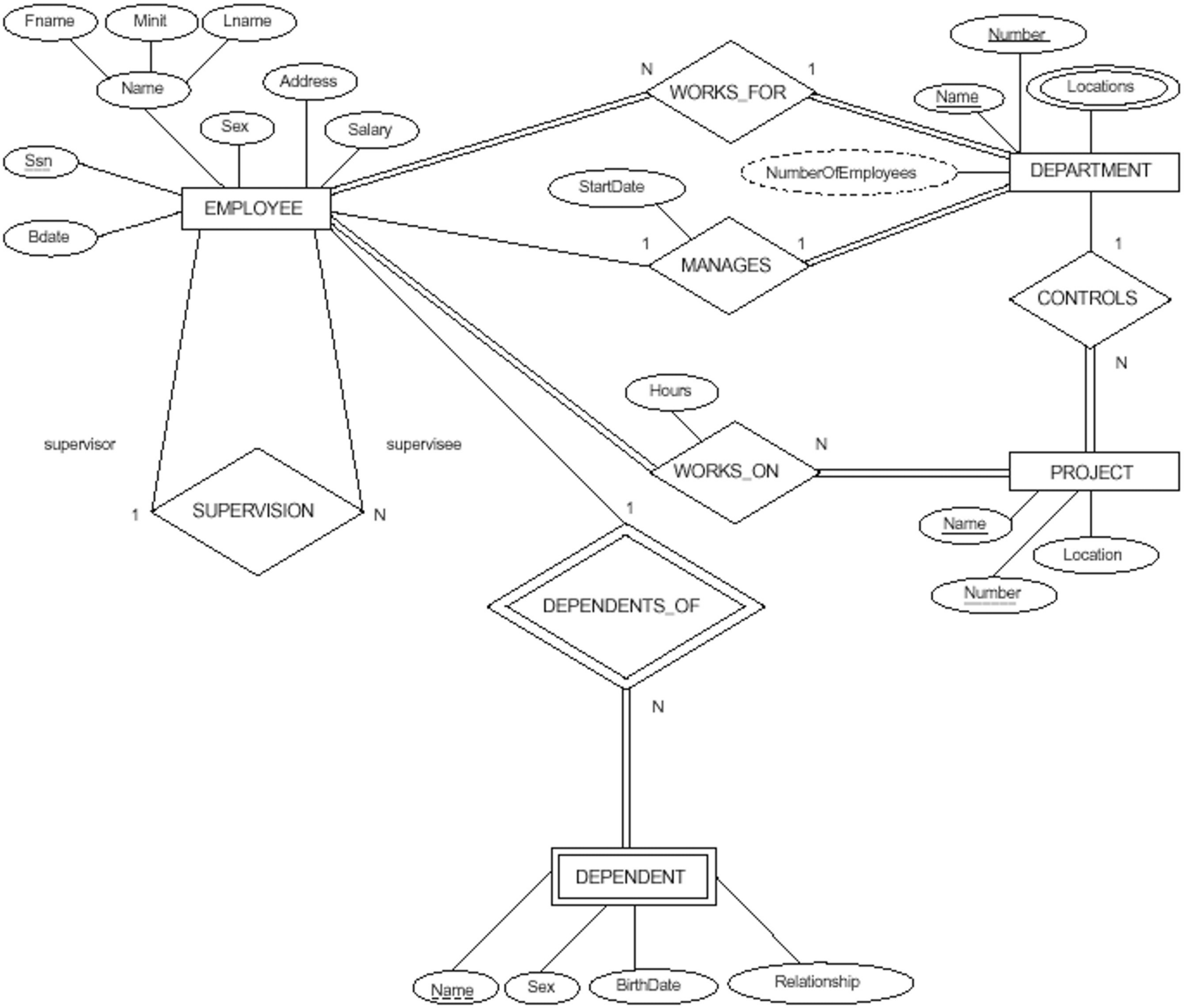


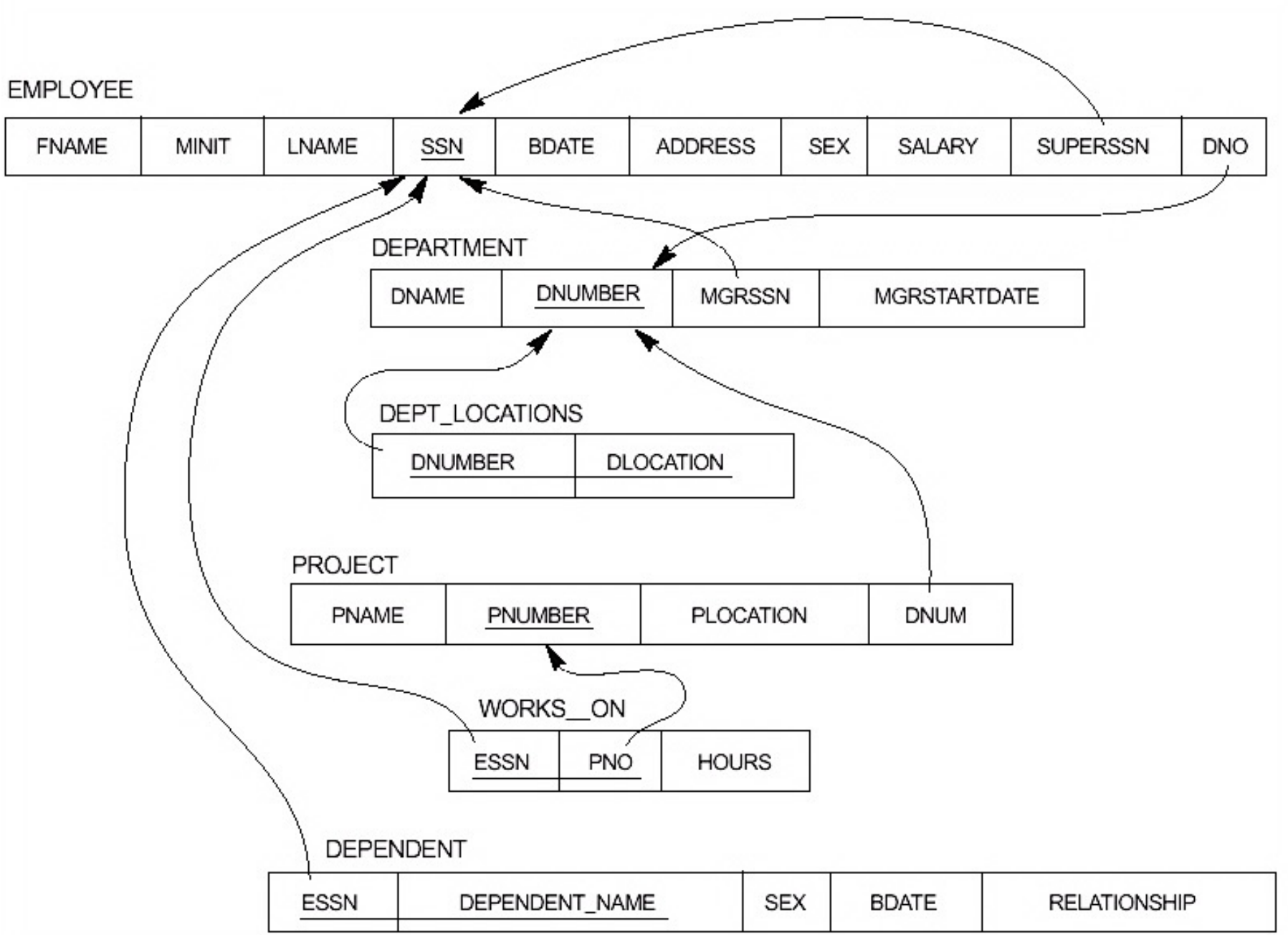
Entity-Relationship Model

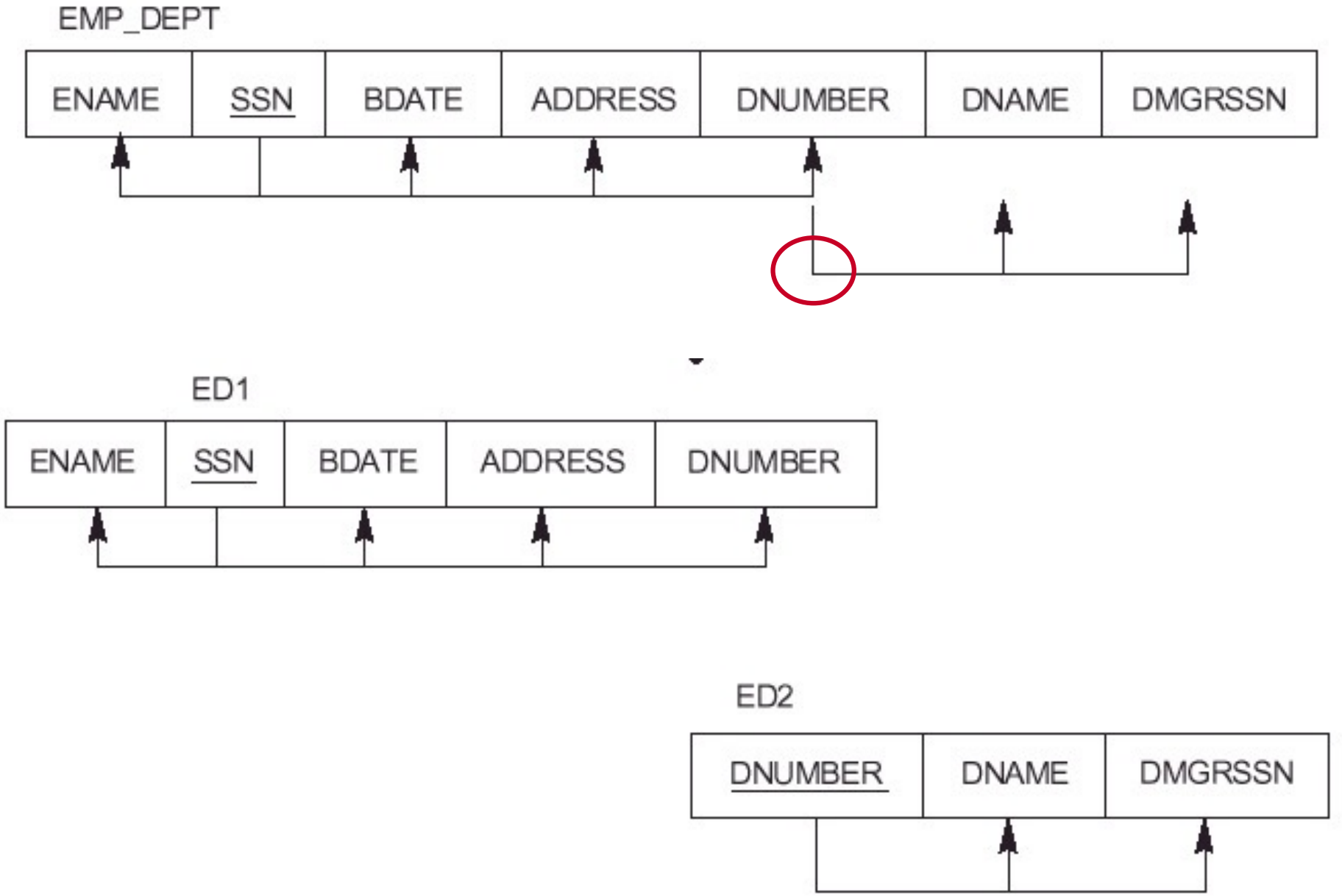
ER to Relational Mapping

Relational Data Model
SQL, Relational Algebra
Normalization

DBMS
Record Structure, Indexing
Query Processing
Query Optimization
Transaction Processing


Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
Bdate
EMPLOYEE
N
WORKS_FOR
1
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
1
MANAGES
1
1
CONTROLS
N
Hours
supervisor
supervisee
N
WORKS_ON
PROJECT
1
SUPERVISION
N
1
Name
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS_ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP


EMP_DEPT
ENAME
SSN
BDATE
ADDRESS
DNUMBER
DNAME
DMGRSSN
ED1
ENAME
SSN
BDATE
ADDRESS
DNUMBER
ED2
DNUMBER
DNAME
DMGRSSN

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

SQL

- Tuple relational calculus + Relational Algebra
- DDL, DML, View, Indexing, …
- Embedd SQL + Host language
- Syntax

SELECT <attribute list>

FROM <table list>

[**WHERE** <condition>]

[**GROUP BY** <grouping attribute(s)>]

[**HAVING** <group condition>]

[**ORDER BY** <attribute list>]

管理資料庫的軟體 ？

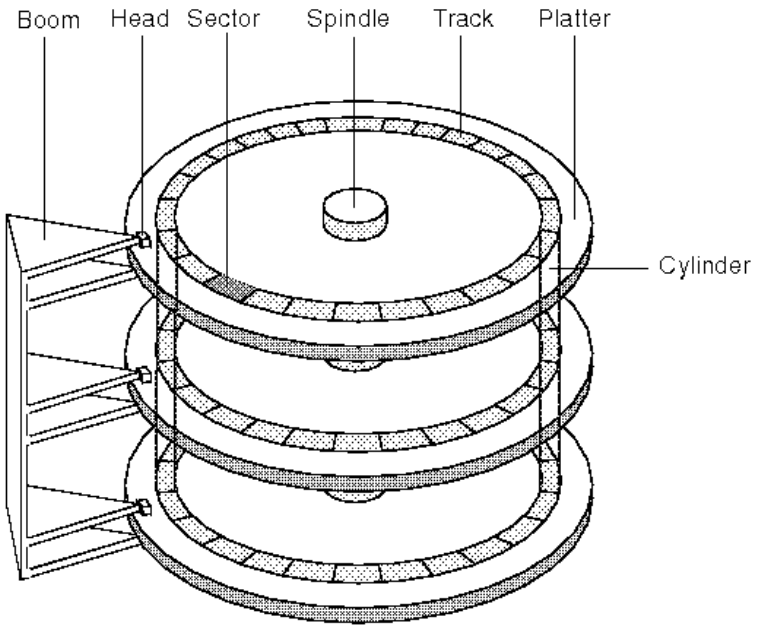
資料庫系統設計 (DBMS Design)

- 資料庫管理系統(DataBase Management Systems, DBMS)
- 資料庫管理系統 如何管理資料庫 (DB)？
  - 資料庫如何儲存在輔助記憶體 (Storage Management)？
  - DBMS 如何處理查詢 (Query Processing)？
  - 如何加快查詢速度 (Indexing, Query Optimization)？
  - 新增、讀取、修改、刪除 (CRUD) 對效能的影響為何？
  - DBMS 可以調校 (Database Tuning)，以提升效能嗎？
  - 資料儲存方式對於效能的影響為何？

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

organization
of data on disk



Physical Database Design

- Storage organization of data
- Process of physical DB design
  - choosing the particular data organization techniques that best suit the given application requirements
- Data stored in disk is organized as files of records
- Primary file organization: determine
  - How the records of a file are physically placed on the disk
  - How the records can be accessed
- Secondary file organization: determine
  - How to allow efficient access to the records of a primary file by Index File (secondary file)

Content

- Secondary Storage Devices
- Placing File Records on Disk
- Operations on Files
- Primary File Organization

Storage Hierarchy

- Primary storage (cache, main memory)
- Secondary storage (flash memory, disk)
- Tertiary storage (optical disk, tape)

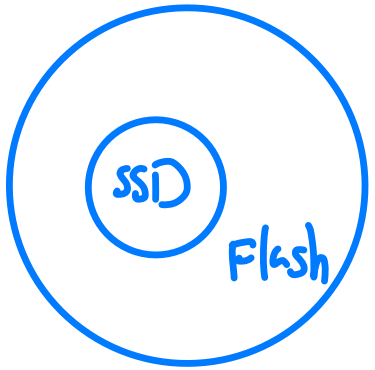
Storage Hierarchy (cont.)

- Primary storage
  - main memory (DRAM), cache(SRAM)
  - Storage media that can be operated on directly by CPU
  - Fast access
  - Of limited storage capacity
  - Contents are lost in case of power failure or system crash

Storage Hierarchy (cont.)

- Secondary storage
  - Disks, Flash Memory (SSD)
  - Cannot be processed directly by the CPU
  - Be copied into main memory
  - Slower access than primary storage
  - Larger capacity than primary storage
  - Less cost than primary storage

Hard Disk Drive (HDD)
≡ Hard Drive
≡ Disk Drive



Storage Hierarchy (cont.)

- Tertiary storage
  - Optical disks, Tapes
  - Removable media as offline storage for archiving DB
  - Be copied into main memory
  - Slower access than secondary storage
  - Larger capacity than secondary storage
  - Less cost than secondary storage

Memory Hierarchies in Modern Computer

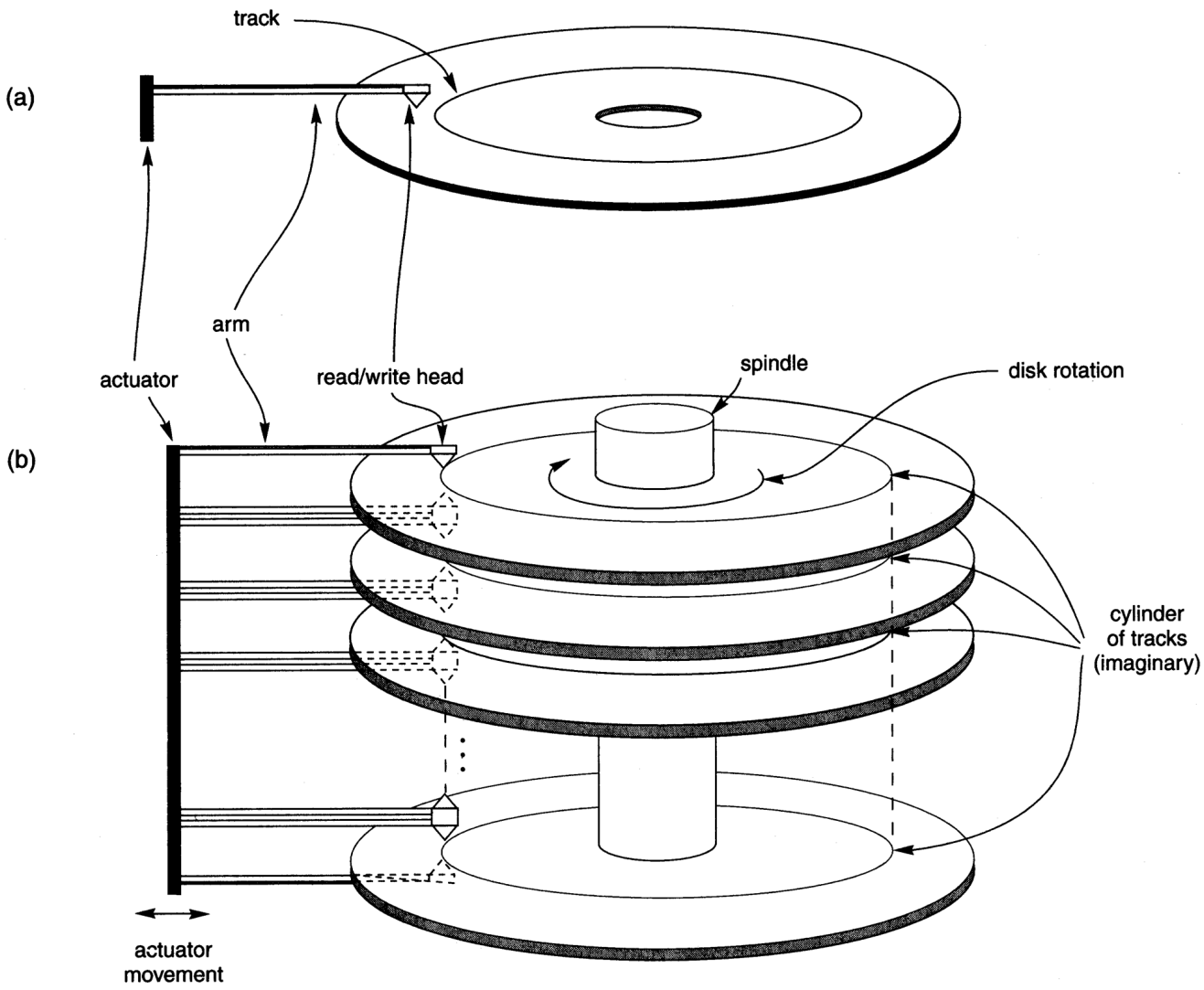
- Cache memory (static RAM)
  - used by CPU to speed up execution of programs
  - Pre-fectching, Pipelining
- Main memory (dynamic RAM)
- Flash memory(EEPROM, electrically erasable programmable ROM)
  - nonvolatile, fast access speed
  - An entire block must be erased & written over a time
- Magnetic disk
- CD-ROM, DVD, Juke box
- Tapes: archiving & backup

Type	Capacity*	Access Time	Max Bandwidth	Commodity Prices (2014)**
Main Memory- RAM	4GB–1TB	30ns	35GB/sec	$100–$20K
Flash Memory- SSD	64 GB–1TB	50μs	750MB/sec	$50–$600
Flash Memory- USB stick	4GB–512GB	100μs	50MB/sec	$2–$200
Magnetic Disk	400 GB–8TB	10ms	200MB/sec	$70–$500
Optical Storage	50GB–100GB	180ms	72MB/sec	$100
Magnetic Tape	2.5TB–8.5TB	10s–80s	40–250MB/sec	$2.5K–$30K
Tape jukebox	25TB–2,100,000TB	10s–80s	250MB/sec–1.2PB/sec	$3K–$1M+

Hardware Description of Disk Devices

- Disk pack
- Track
- Cylinder
- Sector
- Block
  - is set by OS during disk formatting
  - 512~4096 bytes
  - Blocks are separated by fixed-size inter-block gaps

Disk Architecture



Hardware Description of Disk Devices (cont.)

- Transfer of data between main memory & disk
  - in units of disk blocks
  - Hardware address of a block: a combination of a cylinder number, track number & block number
  - Logical block address (LBA): mapped to the hardware address automatically by disk controller
- Cluster: contiguous blocks transferred as a unit
- Disk controller
  - Controls the disk drive & interfaces it to the computer system
  - Standard interfaces
    - SCSI (Small Computer System Interface)
    - SATA (Serial AT Attachment)
    - SAS (Serial Attached SCSI)
  - Accepts high-level I/O commands, take action to position arm and read/write

Hardware Description of Disk Devices (cont.)

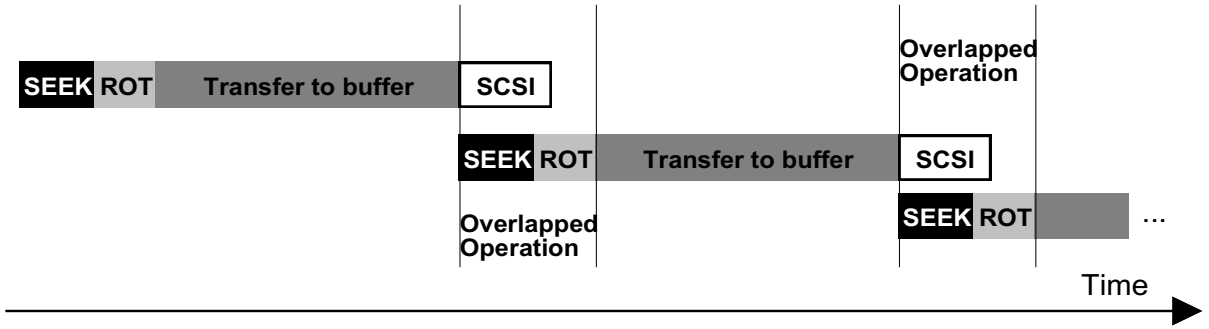
- Seek time
  - Position read/write head on the correct track
  - 5~10 msec. on desktops & 3~8 msec. on servers
    (12~14 msec. 10 years ago)
- Latency time (rotation delay or latency)
  - Rotates the desired block into position under read/write head
  - 2~3 msec. (10000~15000 rpm)
- Block transfer time
  - Transfer the data

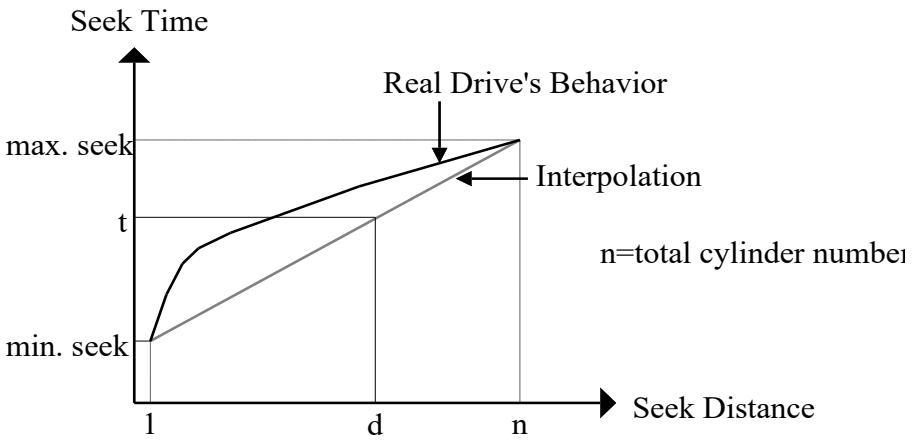
- Locating data on disk is a major bottleneck in DB applications
- Transfer of multiple blocks: transfer consecutive blocks on the same track or cylinder
  - e.g. contiguous blocks
    - (seek+latency) time 9~60 msec., transfer blocks 0.4~2 msec./block

9~60 >> 0.4~2

Disk Access

Disk access time = seek time+ latency time + transfer time



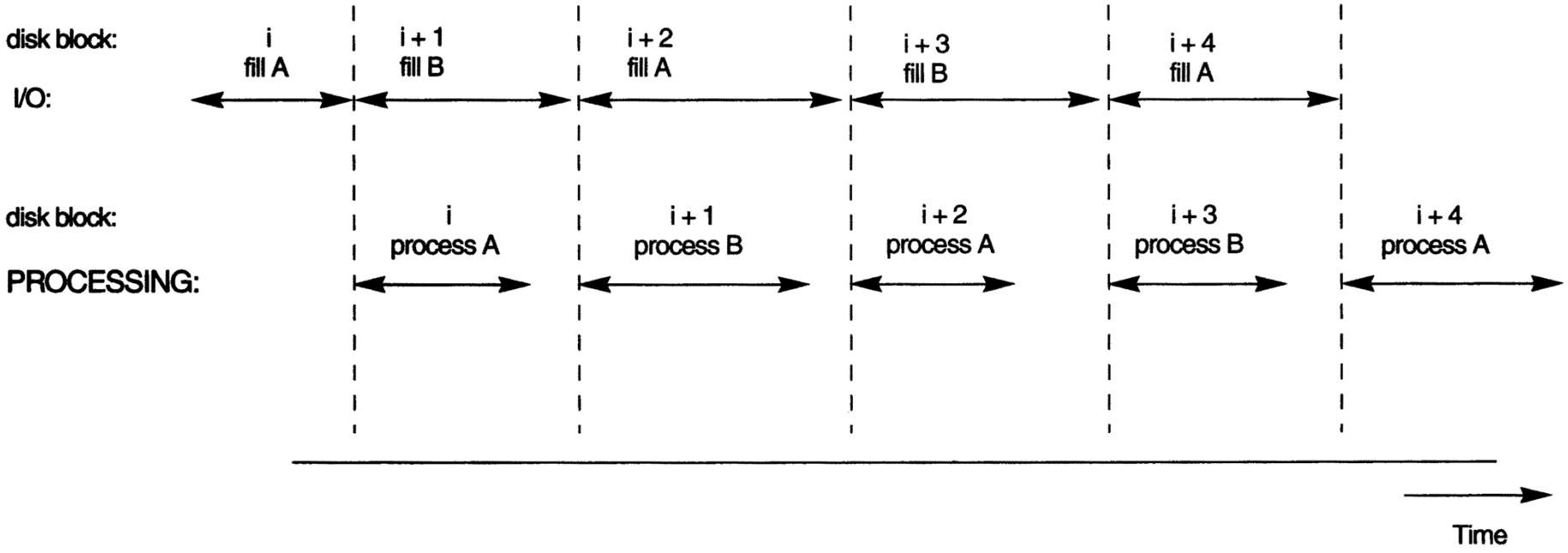


Making Data Access More Efficient on Disk

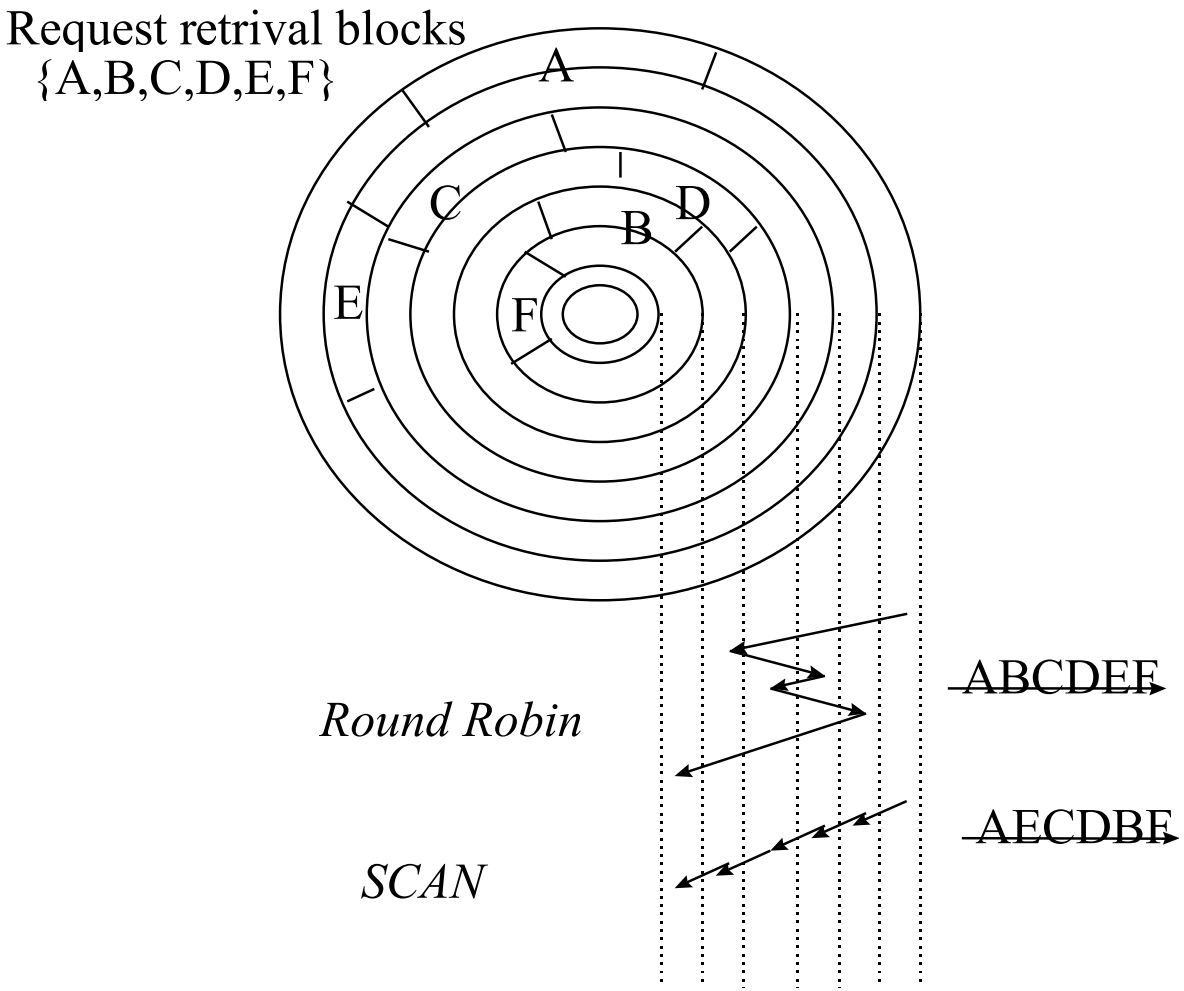
- Buffering of data
- Organization of data on disk
- Disk scheduling
- Prefetch
- Use of log disks to temporarily hold writes
- Use of SSDs for recovery purpose

Buffering of Blocks

- When several blocks need to be transferred from disk to memory & all the block address are known
  - Buffers can be served in memory to speed up the transfer
  - While one buffer is being accessed, CPU can process data in the other buffer
  - Double buffering : minimizes idle time of CPU & disk, throughput ↑



Disk Scheduling



RAID

- RAID: Redundancy Array of Inexpensive Disks
- Dimensions of Disk arrays
  - addressing space
  - fault tolerance
    - data replication
    - data encoding

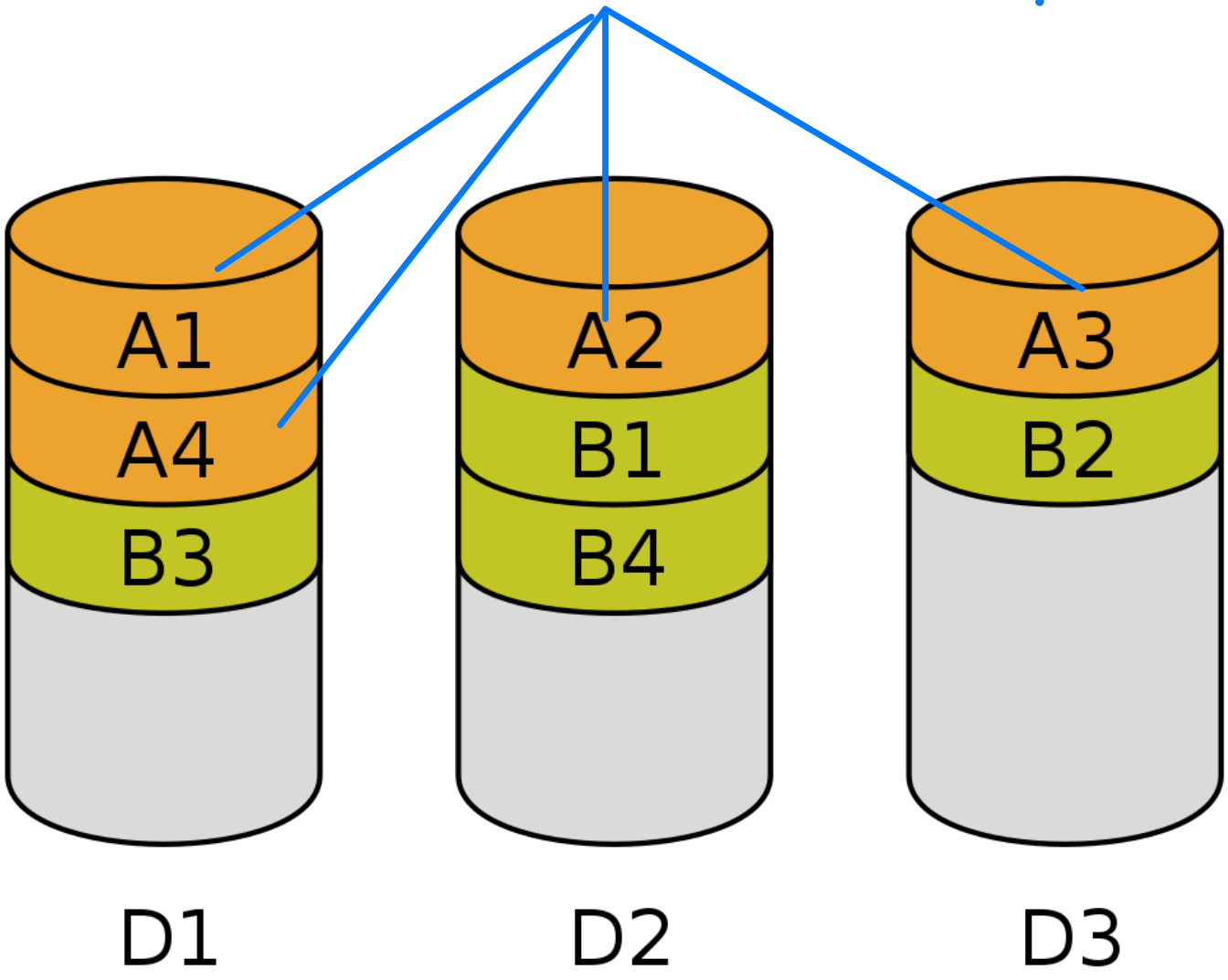
many disks

↓

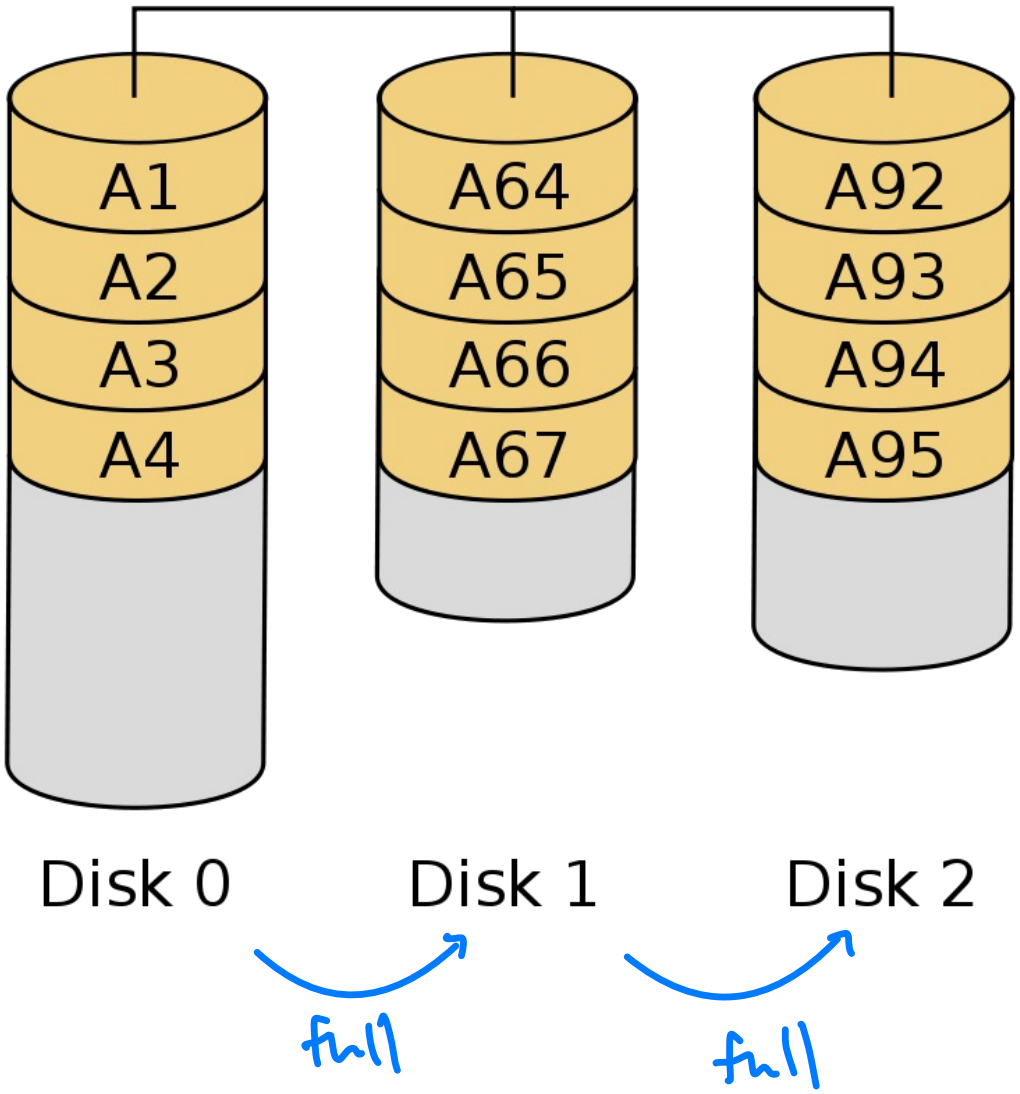
1 logical disk

Striping

data is broken down into smaller pieces

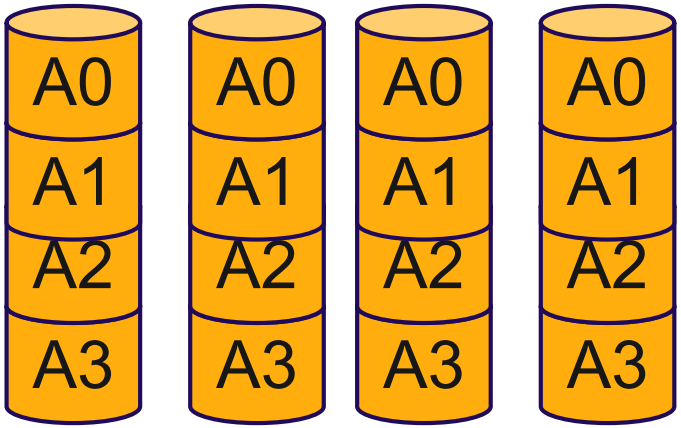


JBOD
Just a Bunch Of Disk



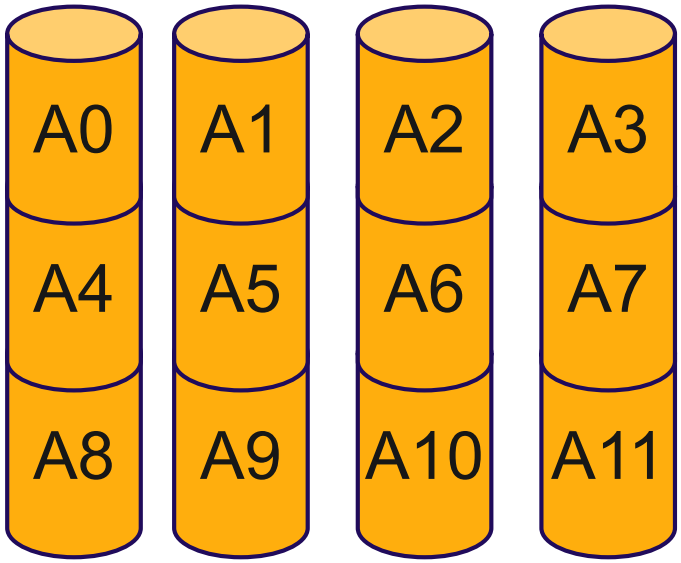
Addressing Space of Disk Array

bit-interleaving



parallelism: 4
concurrency: 1

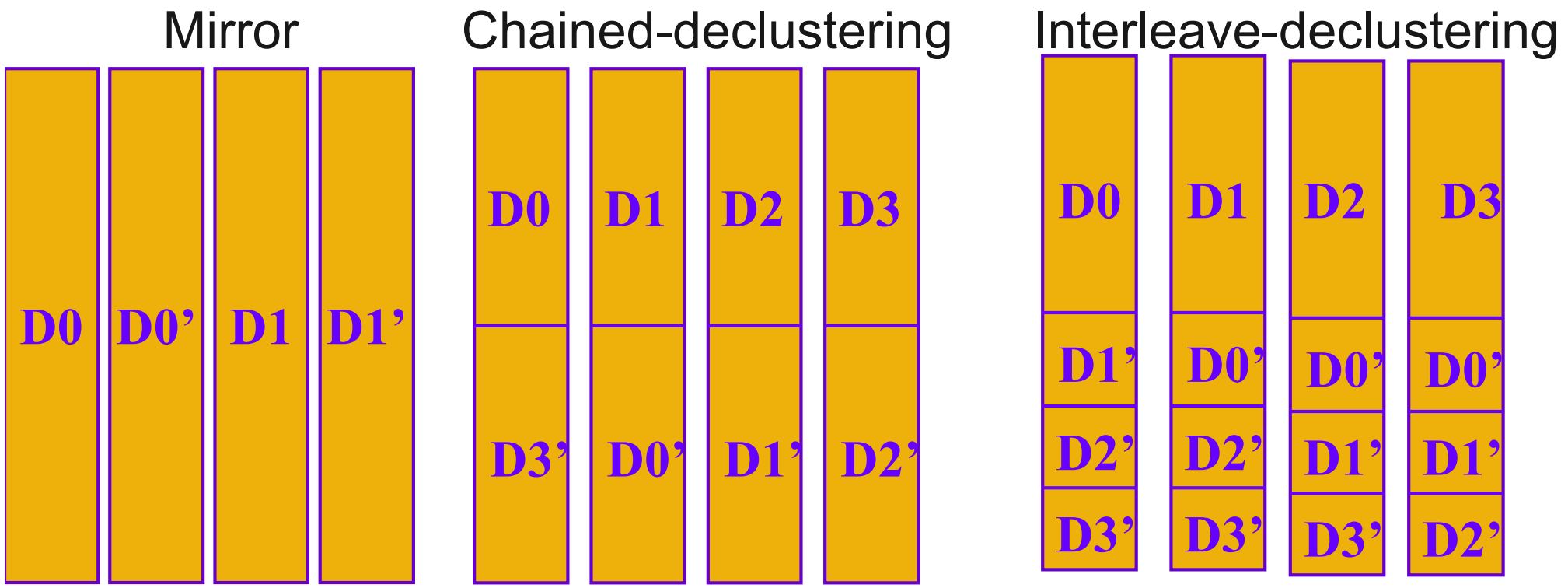
block-interleaving



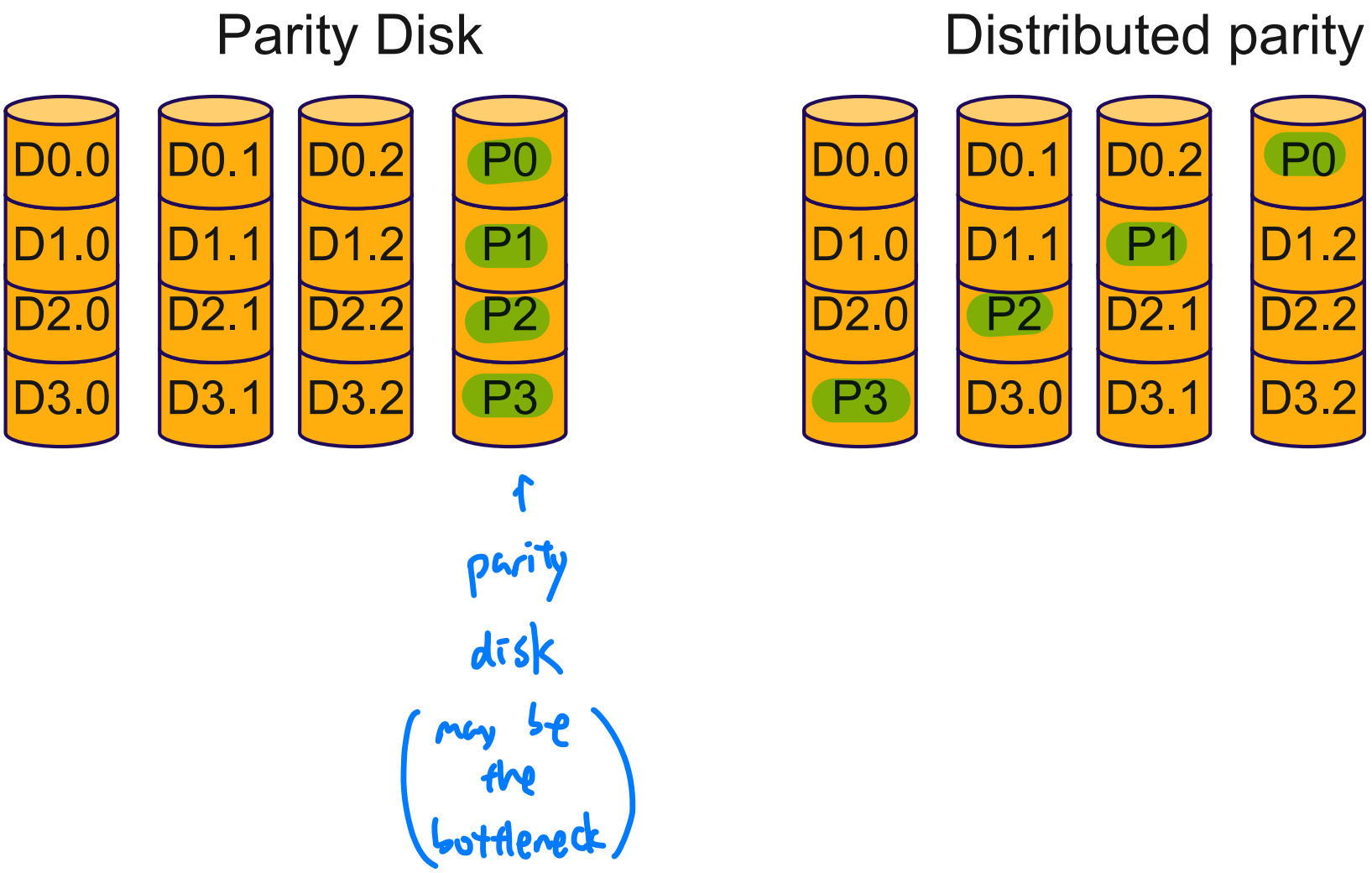
R1: (A0, A1), R2:(A6, A7)
parallelism: 2
concurrency: 2

1. **Bit-interleaving:**
   - **Description**: Data is distributed across the disks at a very fine granularity, often at the bit level. This means that every consecutive bit of data is stored on a different disk.
   - **Parallelism**: High (4 in the example), as each disk can be accessed simultaneously for different bits.
   - **Concurrency**: Low (1 in the example), as data related to a single operation (such as a word or byte) is spread across all disks, so each operation can only handle one piece of data at a time across all disks.
2. **Block-interleaving:**
   - **Description**: Data is divided into larger blocks (than bits), and these blocks are then distributed across the disks. Each disk holds a complete block of data before moving to the next disk for the subsequent block.
   - **Parallelism**: Lower than bit-interleaving (2 in the example), as fewer disks can be accessed simultaneously for different blocks.
   - **Concurrency**: Higher (2 in the example), as multiple blocks can be accessed or processed concurrently, allowing for more operations to be performed at the same time.

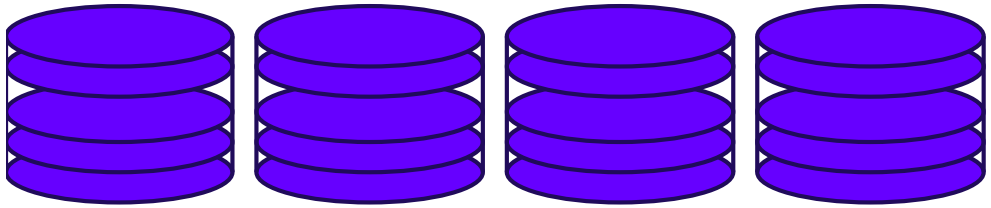
Data Replication of Disk Array



Placement of Data Encoding in Disk Array



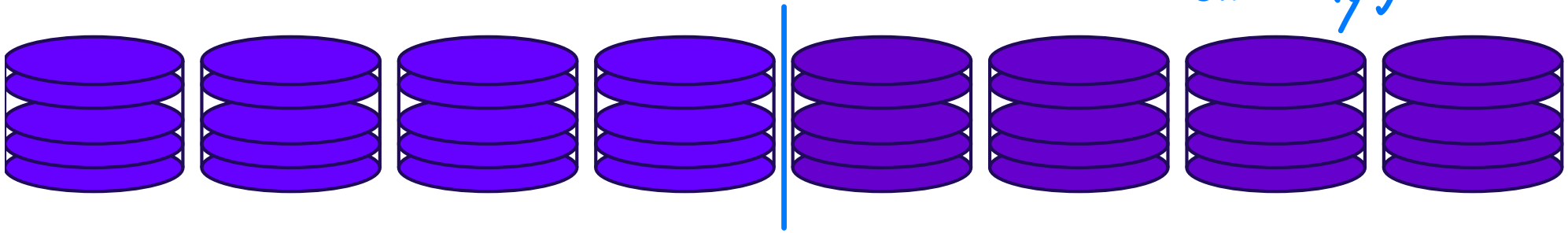
RAID 0, 1, 2



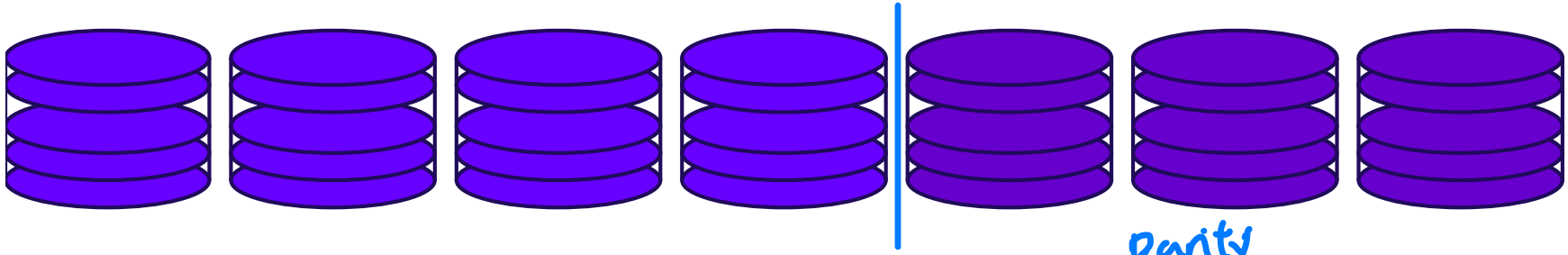
Speed ↑

RAID 0: independent-addressing

Reliability ↑



RAID 1: block interleaving + mirroring

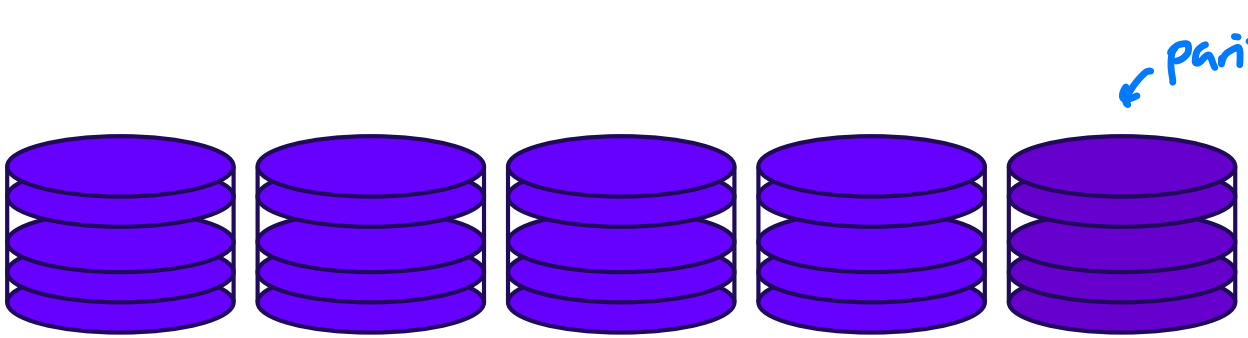


≥ 3 drives

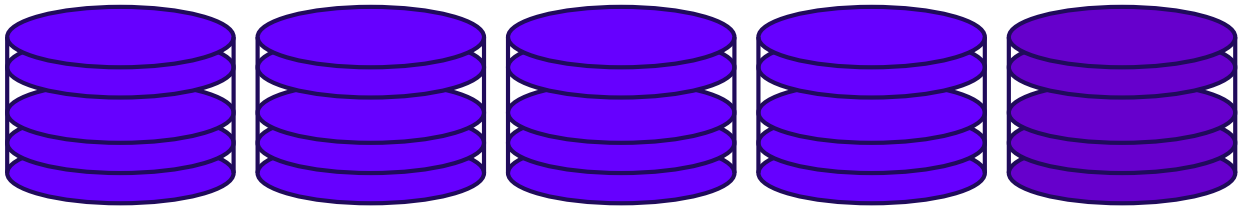
parity

RAID 2: bit interleaving + memory-style ECC

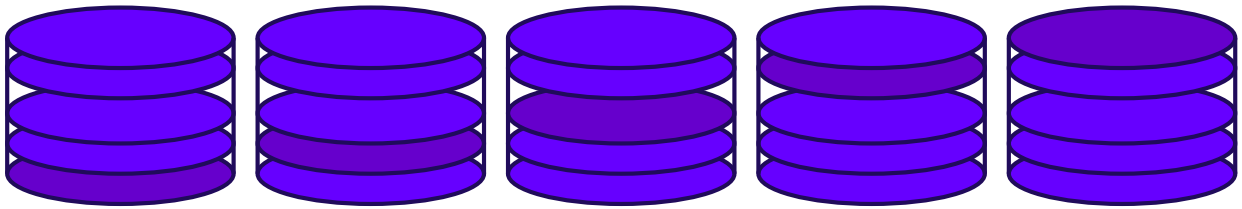
RAID 3, 4, 5



RAID 3: byte-interleaving + parity check



RAID 4: block interleaving + parity check



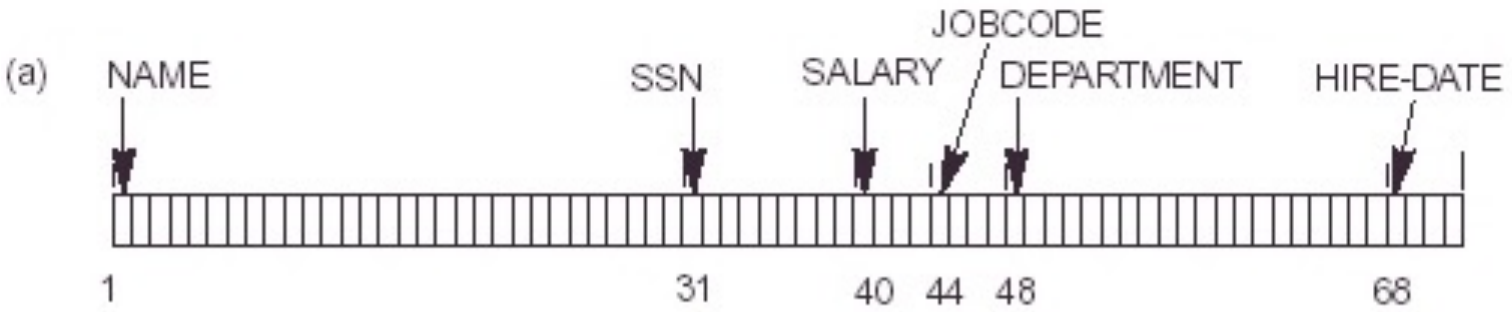
RAID 5: block interleaving + distributed parity check

Content

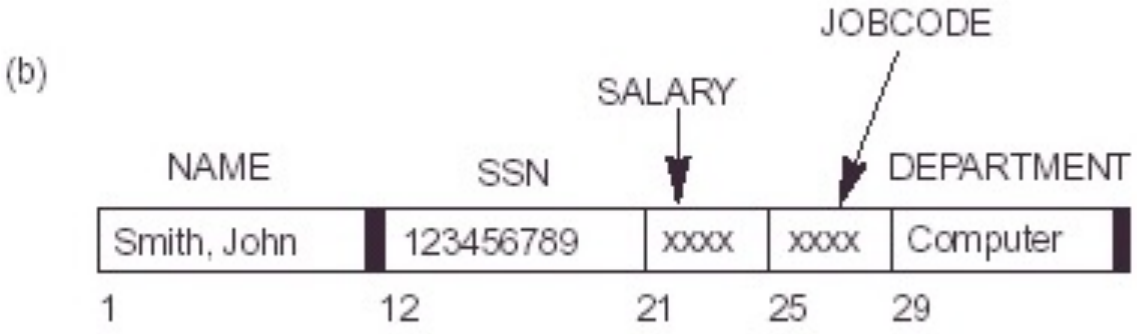
- Secondary Storage Devices
- Placing File Records on Disk
- Operations on Files
- Primary File Organization

Placing File Records on Disk

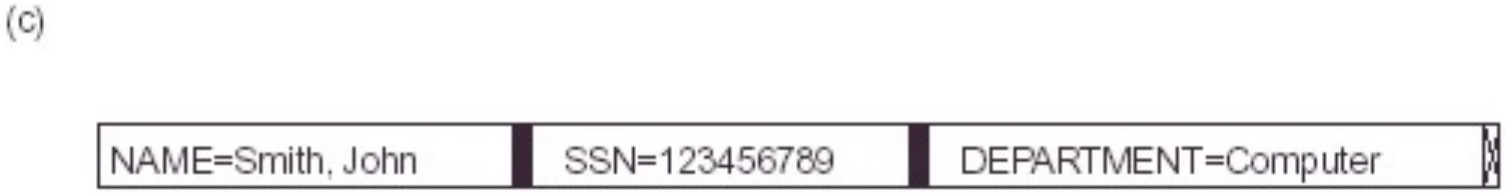
- Fixed-length records
- Variable-length records
  - Variable-length fields
    - Special separator characters
    - Store the length of the field, preceding the field value
  - Repeating fields (multiple values)
    - Allocate as many spaces as the maximum number of values
    - Special separator characters
  - Optional fields
    - <field name, field value> if #(actually appear fields) is small
    - Field type code, rather than field name
  - Mixed file: records of different record type
    - Each record is preceded by a record type indicator



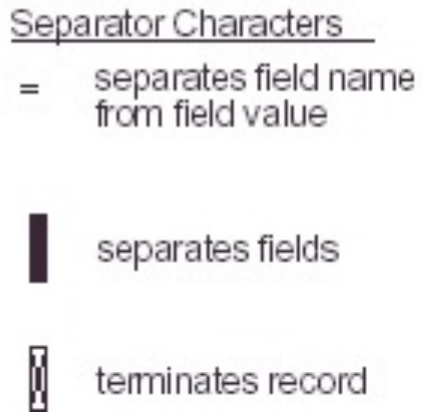
Fixed length record



Variable length record (name, department variable length)



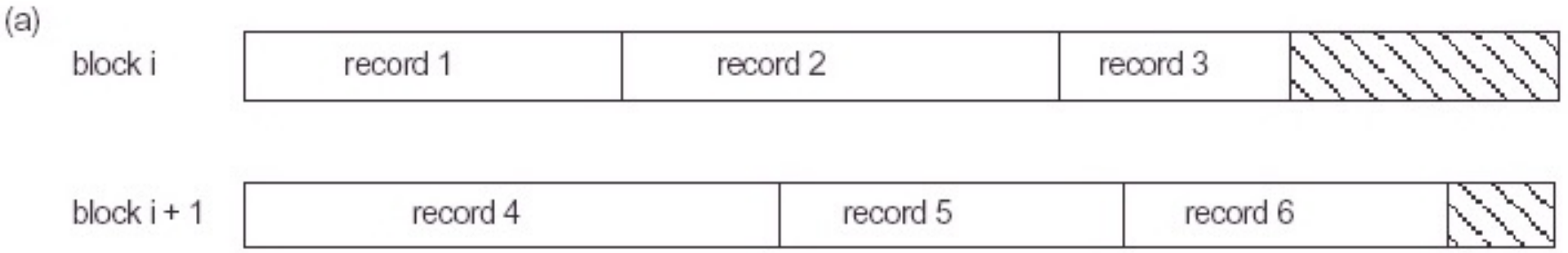
Variable length with optional field

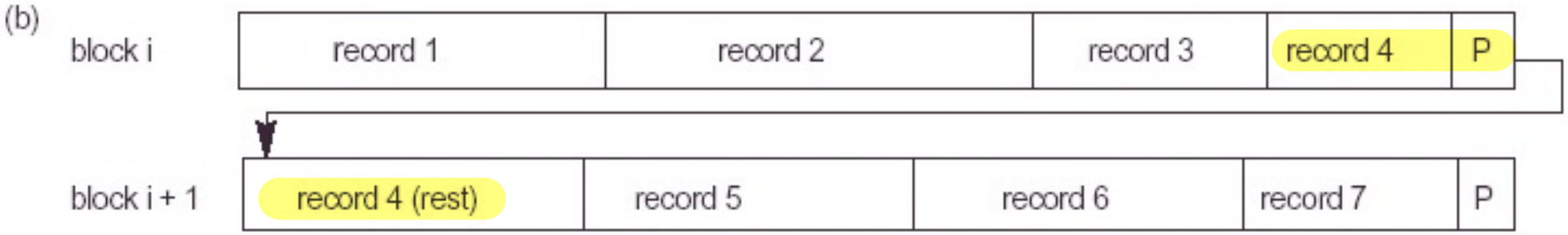


Record Blocking

- Blocking factor: # of records in a disk block
- Utilization of unused space in disk block
  - Store part of a record on one block and the rest on another
  - A pointer at the end of the first block points to the second block
  - Spanned: records span more than one block
  - Unspanned: records not allowed to cross block boundaries

Unspanned



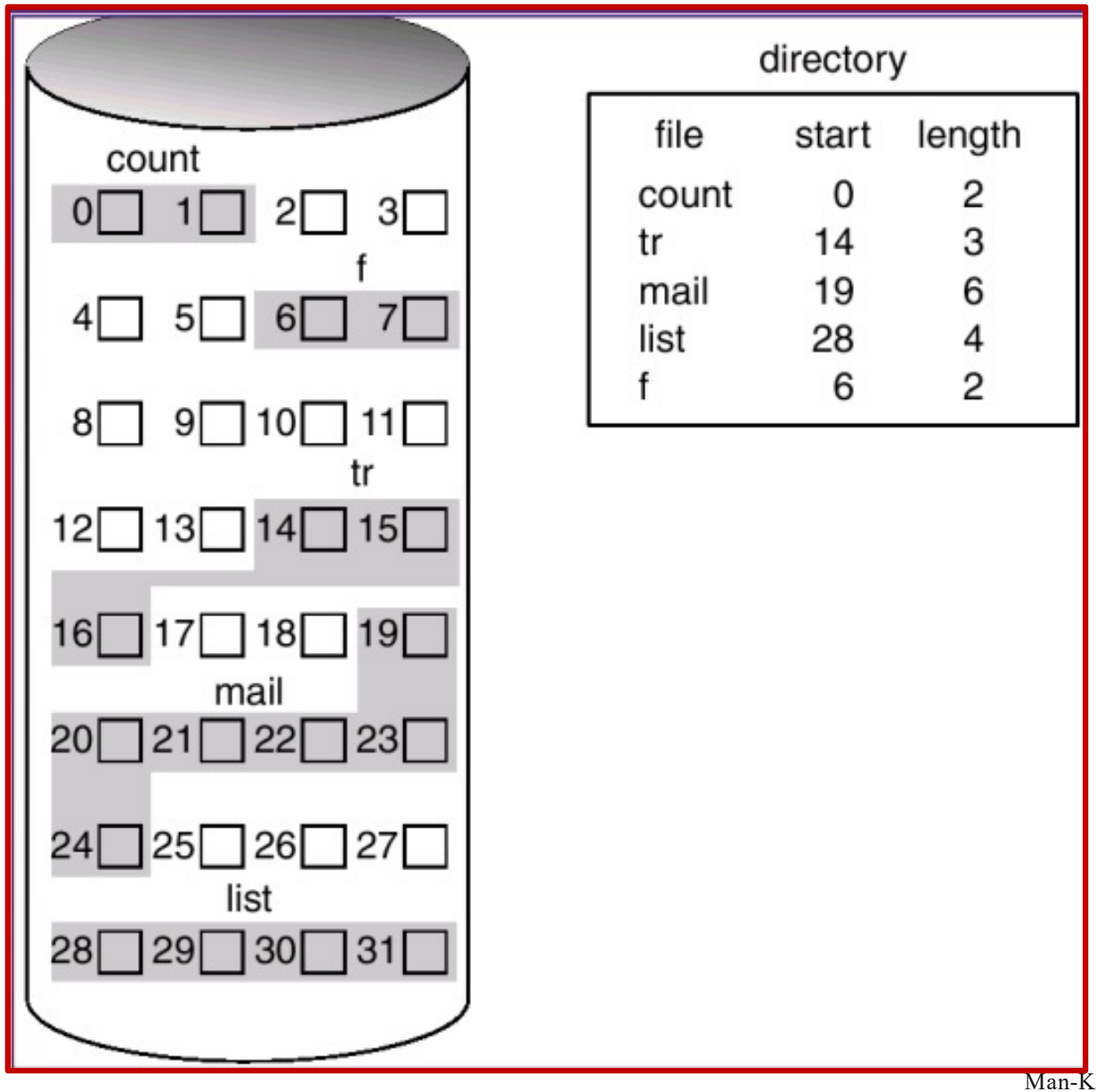


Spanned

Allocating File Blocks on Disk

- Contiguous allocation
- Linked allocation
- Linked clusters allocation
- Index allocation

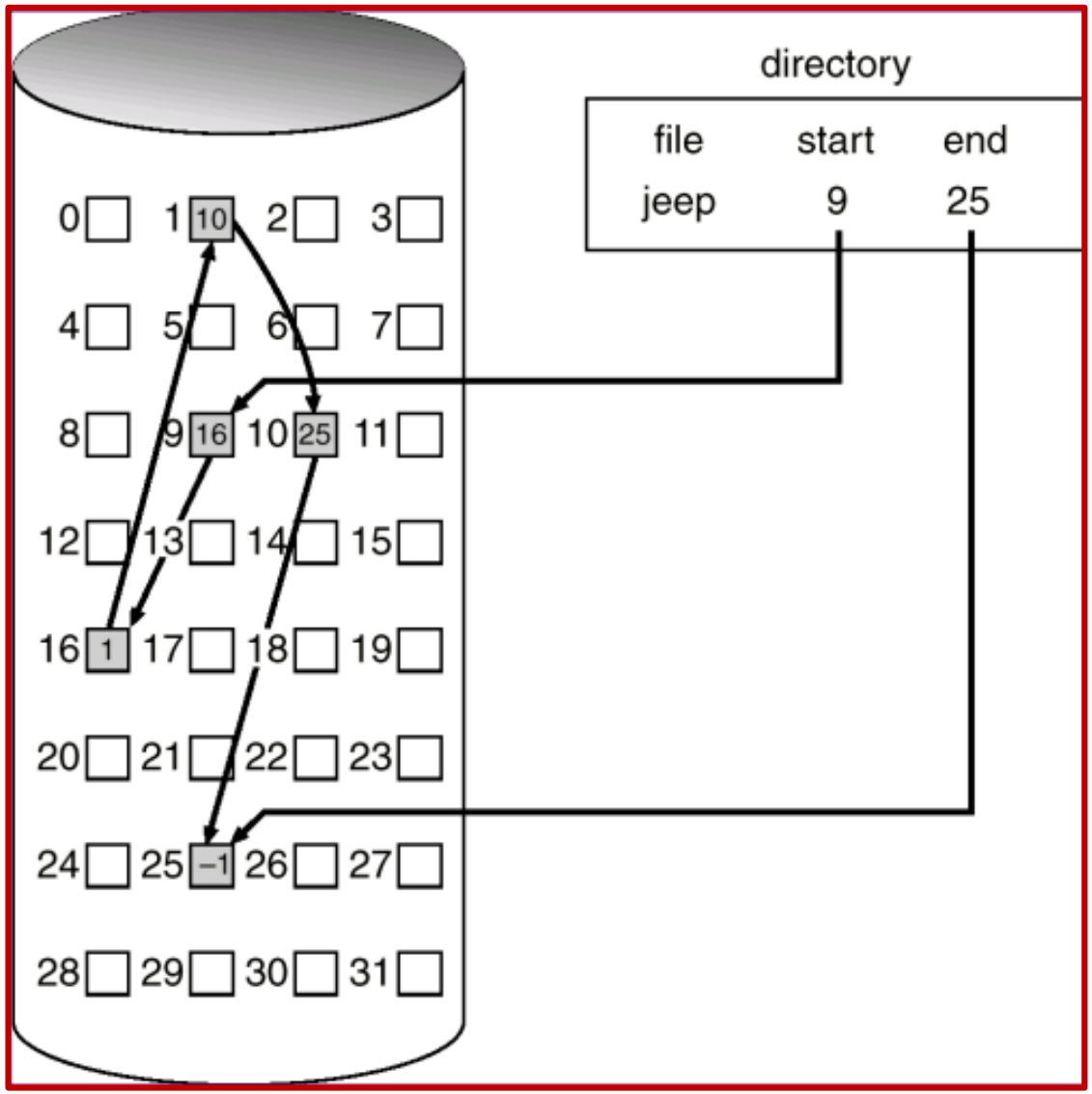
Contiguous Allocation



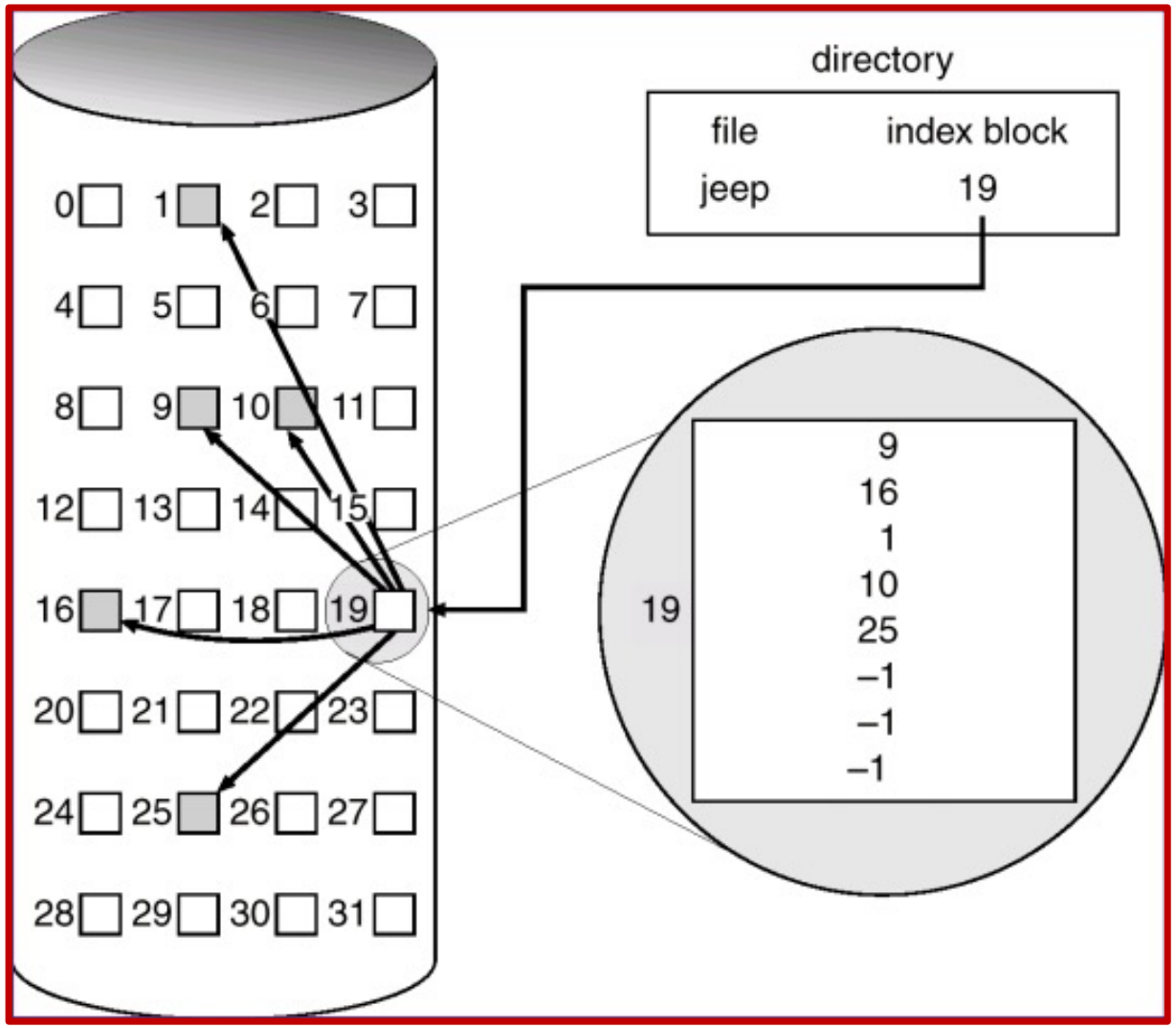
directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Linked Allocation



Index Allocation



Content

- Secondary Storage Devices
- Placing File Records on Disk
- Operations on Files
- Primary File Organization

Operation on Files (*)

- Operations on files are grouped into
  - Retrieval operations
  - Update operations
- Common operations
  - Open: allocate buffers, access file header, set file pointer
  - Reset: set file pointer
  - Find (locate): search for the first record, transfer into buffer, file pointer to the record
  - Read: copy from buffer to program variable, advance file pointer
  - Find-next
  - Delete: delete current record & update file on disk
  - Modify: modify current record & update file on disk
  - Insert: locate block, transfer into buffer, insert & write buffer to disk
  - Close: release buffer

Operations on Files (cont.)

- Set-at-A-Time operations
  - FindAll
  - FindOrdered
  - Reorganize

Content

- Secondary Storage Devices
- Buffering of Blocks
- Placing File Records on Disk
- Operations on Files
- Primary File Organization

File Organization

- File organization
  - Organization of data of a file into records, blocks & access structure

- Primary file organization: data file
- Secondary file organization: index file

Static vs. Dynamic Files

- Static files: files where update operations are rarely performed
- Dynamic files: files where update operations are constantly applied.

Primary File Organization

- Heap files (unordered files, pipe files, sequential files)
  - Records are placed in the inserted order
  - New records are inserted at the end of the file
  - Insertion: efficient,
  - Deletion: inefficient, lazy deletion
- Sorted files
  - Ordered files, sequential files
  - Binary search on ordering key field
  - Insertion, deletion cost: master file + overflow file(transaction file)
  - Modifying cost
  - Are rarely used in DB application unless primary index is used
- Hashing files
  - Internal hashing
  - External hashing

records are marked as deleted but not physically removed immediately. These records are skipped during normal operations and removed during a periodic cleanup process.

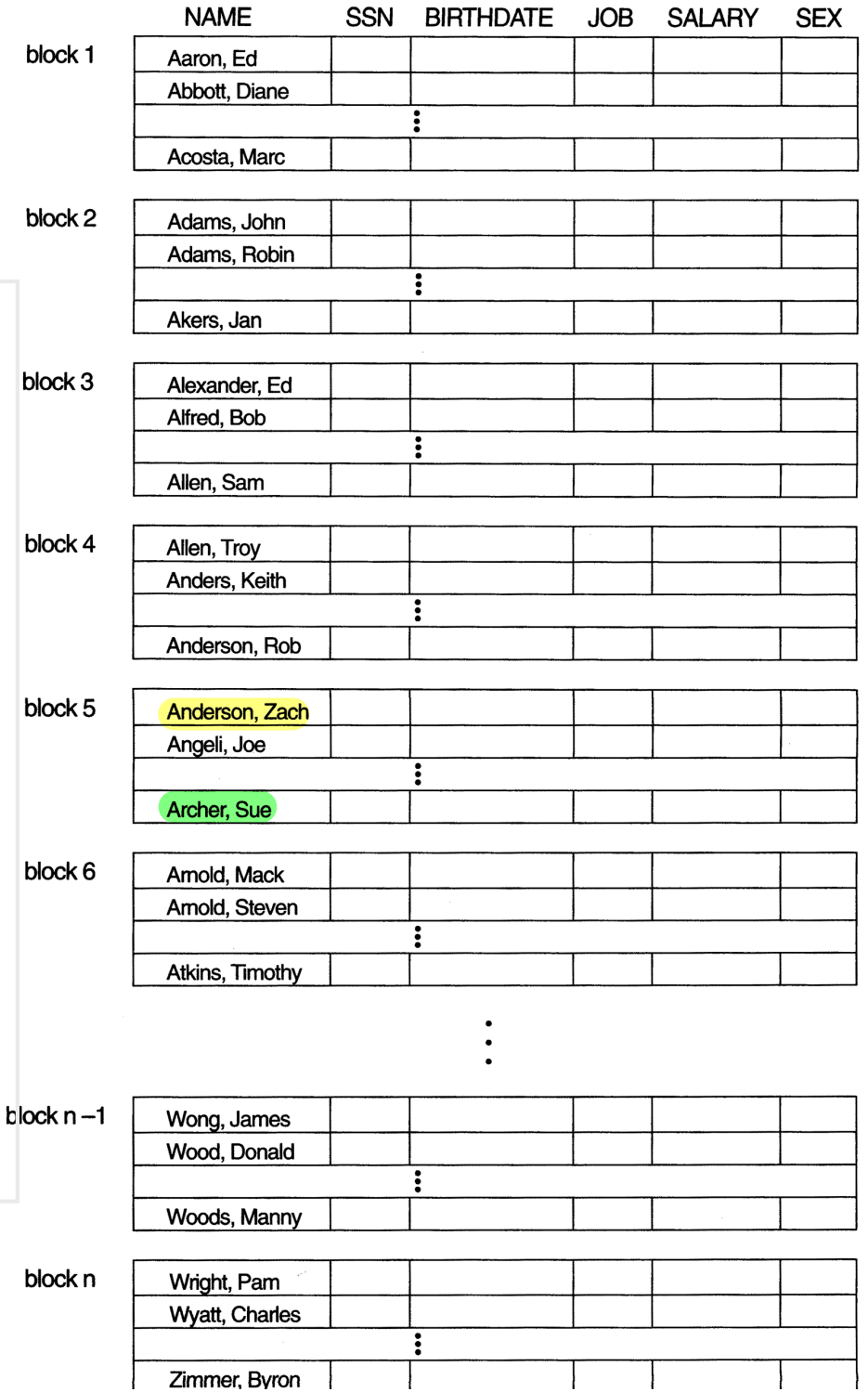
Sorted File

key field

	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					

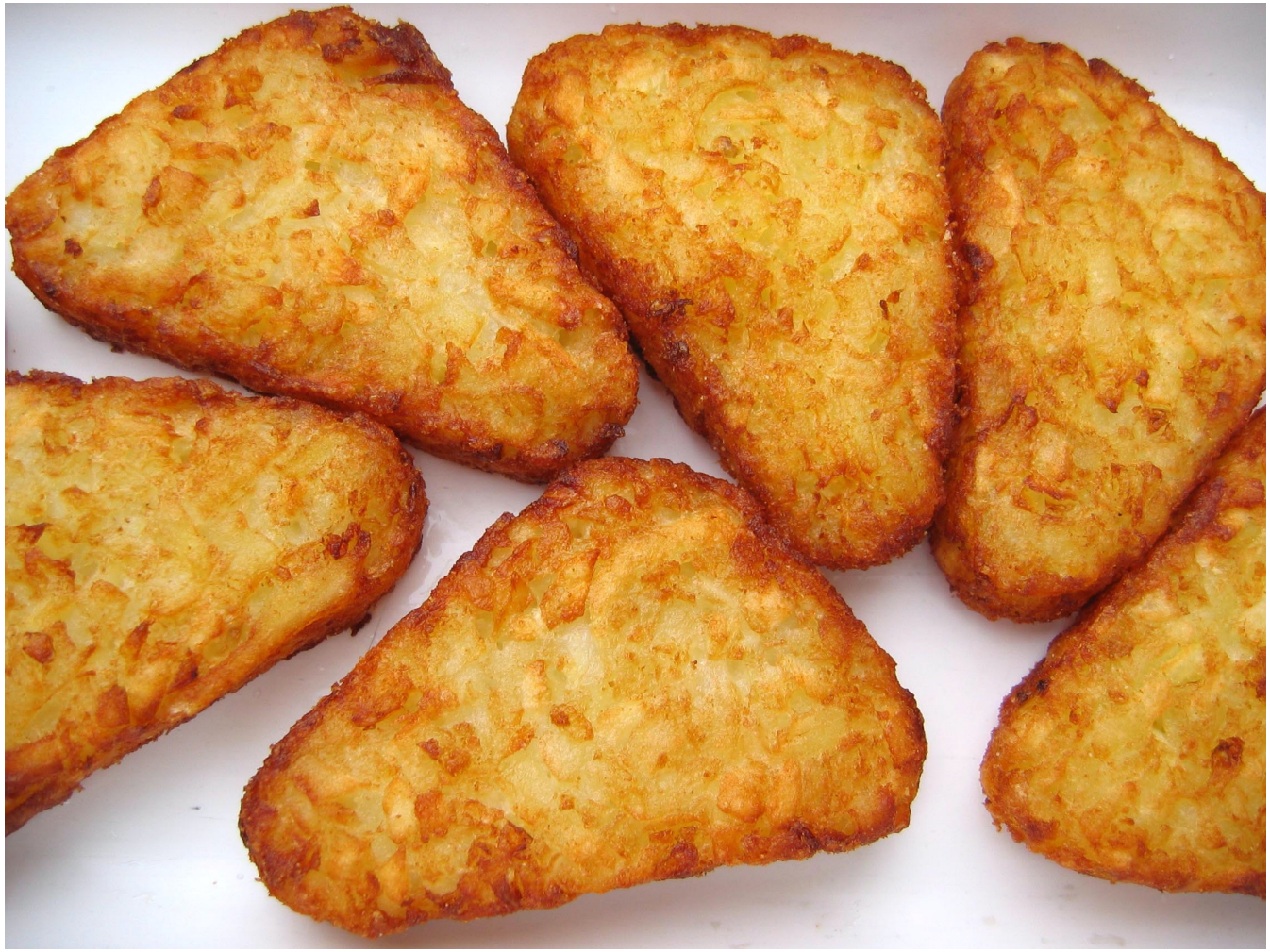
Binary search *K* on sorted file

$l = 1,\ u = \text{n}$
While $(l \leq u)$ do
    $i = (l + u)\ /2$
    read block i into buffer
    if $K <$ first record of block i
        $u = i - 1$
    else if $K >$ last record of block i
        $l = i + 1$
    else if K is in the buffer
        found
    else not found



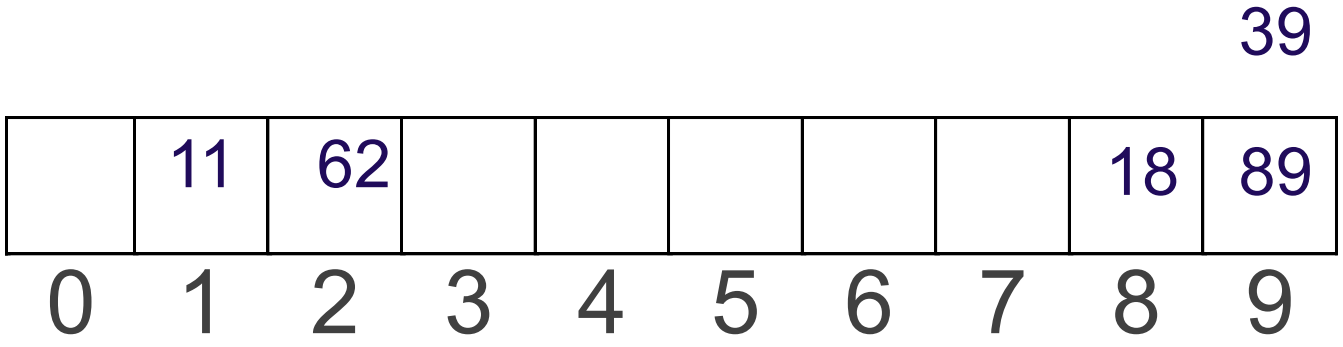
Hashing File

- Implementation of hash tables: hashing
- Hashing: insertions, deletions & finds in constant time.
- General data structure of hashing
  - array of fixed size (table size) containing the keys
  - hash function



Hashing

- ***H*(*K*) = *K* % *TS*,**
- **Example: *H*(*K*) = *K* % 10**



*** insert 89, 18, 11, 62**

Collision & Overflow

- Collision: two keys hash to the same value.
- Overflow: collision and the cell is full.

Hashing

- **Static hashing**
  - perfect hashing without collision
  - hashing with collision
    - separate chaining
    - open addressing
      - linear probing
      - quadratic probing
      - double hashing
    - rehashing

hash value + x

hash value + x^2

(2 different hash functions
e.hashfunc1() + e.hashfunc2())

- **Dynamic hashing (for external hashing)**
  - extensible hashing
  - linear hashing

Rehashing

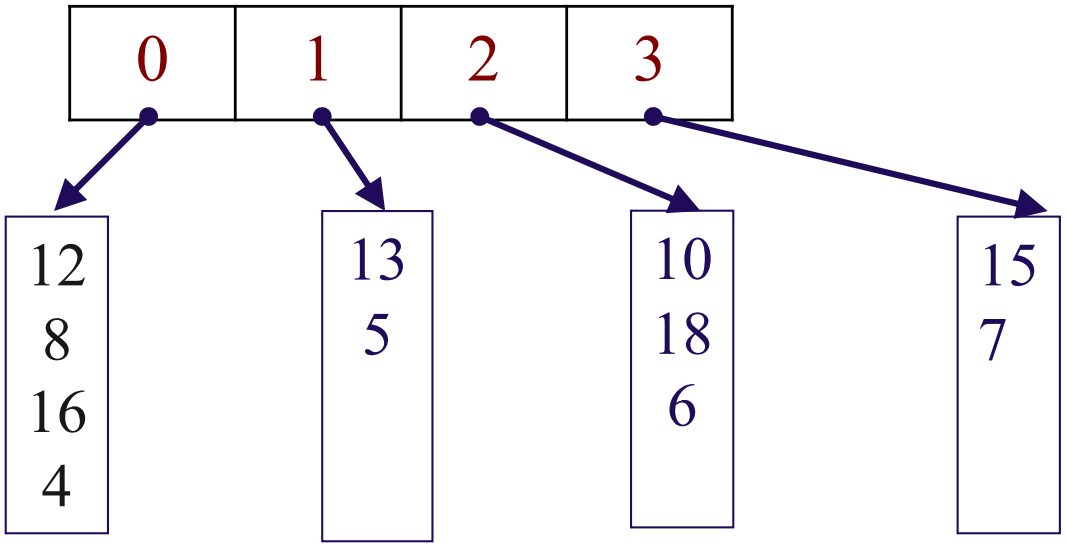
	29	37	17		12	
0	1	2	3	4	5	6

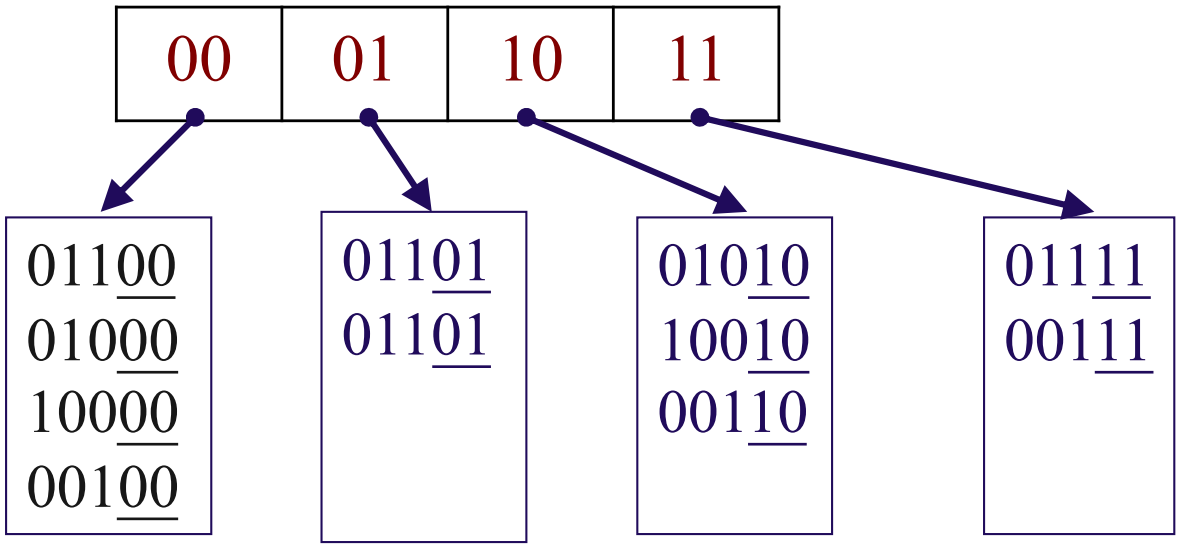
h(k) = k % 7

	12			37		17	29			
0	1	2	3	4	5	6	7	8	9	10

h(k) = k % 11

External Hashing



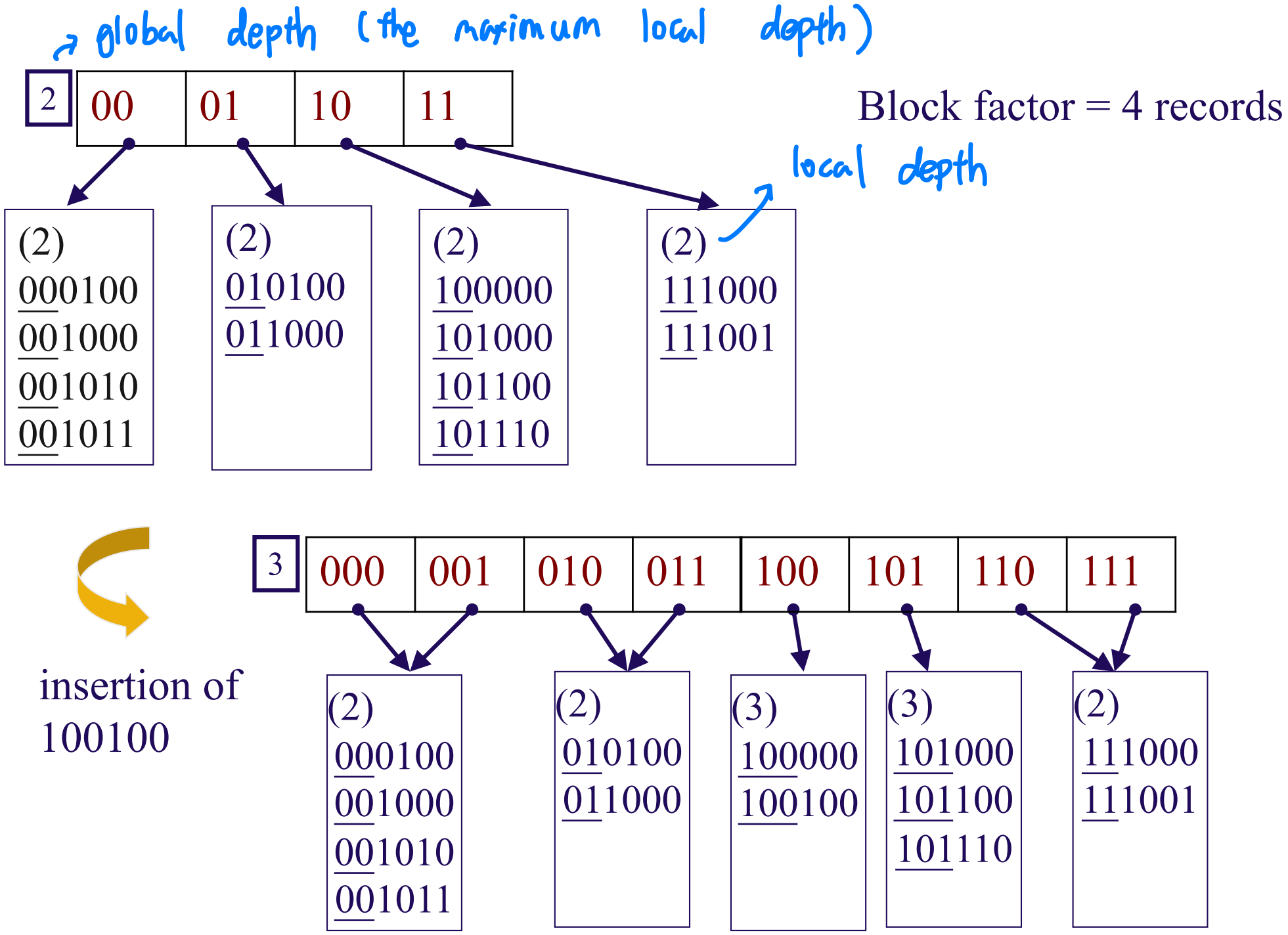


Dynamic Hashing

- Hashing function is changed dynamically.
- Widely used in external hashing
- Two popular dynamic hashing scheme
  - Extendible hashing
  - Linear hashing

Extendible Hashing

- Hashing for searching of large amount of data.
- Main consideration: #(disk access)
- Data structure
  - Directory (memory):
    - hashing function
    - global size
  - Data bucket (disk block)
    - local size


global depth (the maximum local depth)
2
00
01
10
11
Block factor = 4 records
local depth
(2)
000100
001000
001010
001011
(2)
010100
011000
(2)
100000
101000
101100
101110
(2)
111000
111001
insertion of
100100
3
000
001
010
011
100
101
110
111
(2)
000100
001000
001010
001011
(2)
010100
011000
(3)
100000
100100
(3)
101000
101100
101110
(2)
111000
111001

Conclusions

- Disk access unit: block
- Disk access time= seek time + rotation latency time + block transfer time
- Data organization on disk
  - Fixed vs. Variable length record
  - Un-spanned vs. Spanned record blocking
  - Block allocation
  - Primary file organization
    - Heap (unordered file)
    - Sorted file (ordered file, binary search)
    - Hashed file (dynamic hash)
  - Secondary file organization (index file, next chapter)