Nguyên lý Hệ quản trị CSDL

Chương 2.1: Lưu trữ và Cấu trúc tập tin

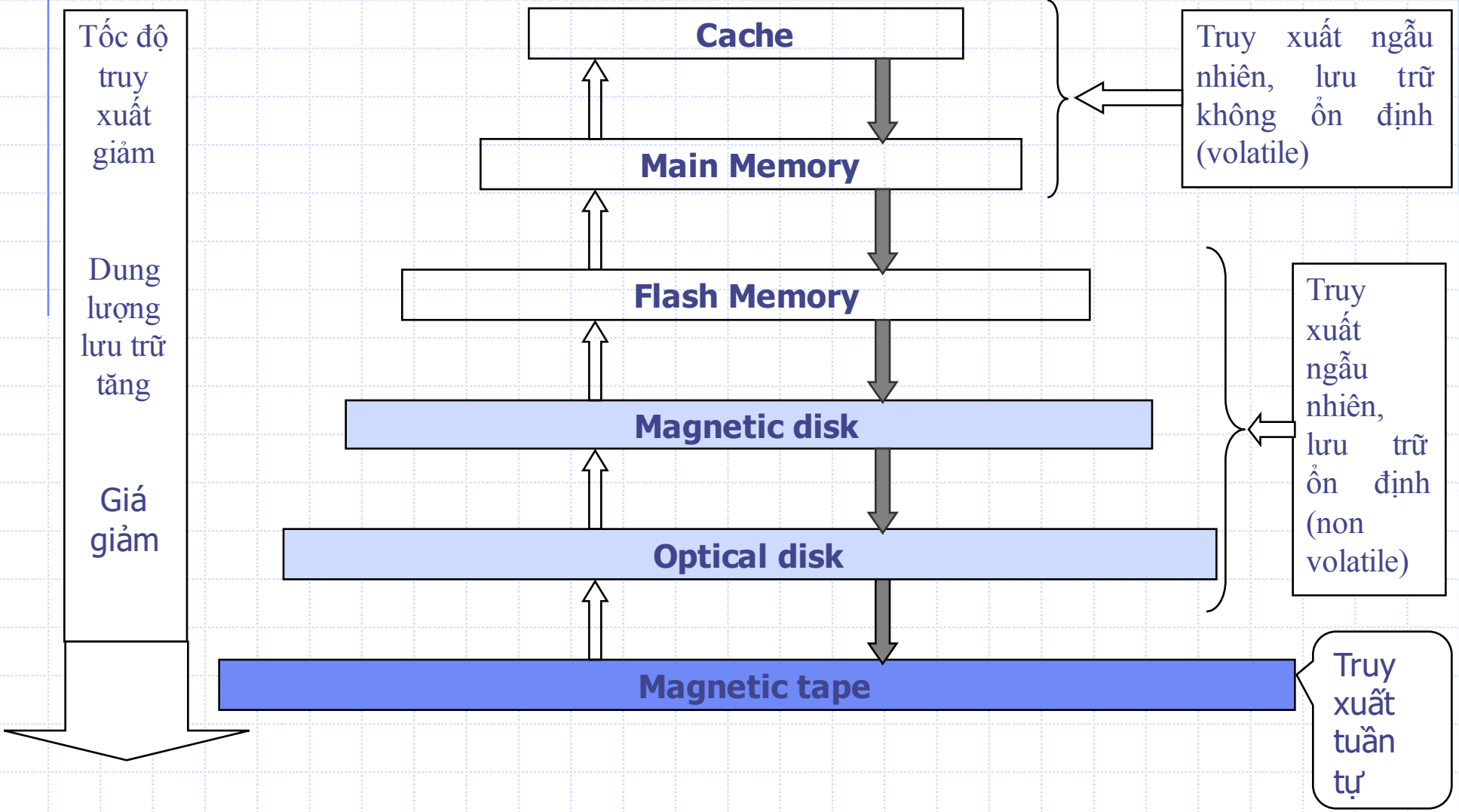
Mục tiêu

Nhằm giới thiệu các phương tiện lưu trữ, các phương pháp tổ chức tập tin và phương pháp tổ chức các mẩu tin trong tập tin

Nội dung

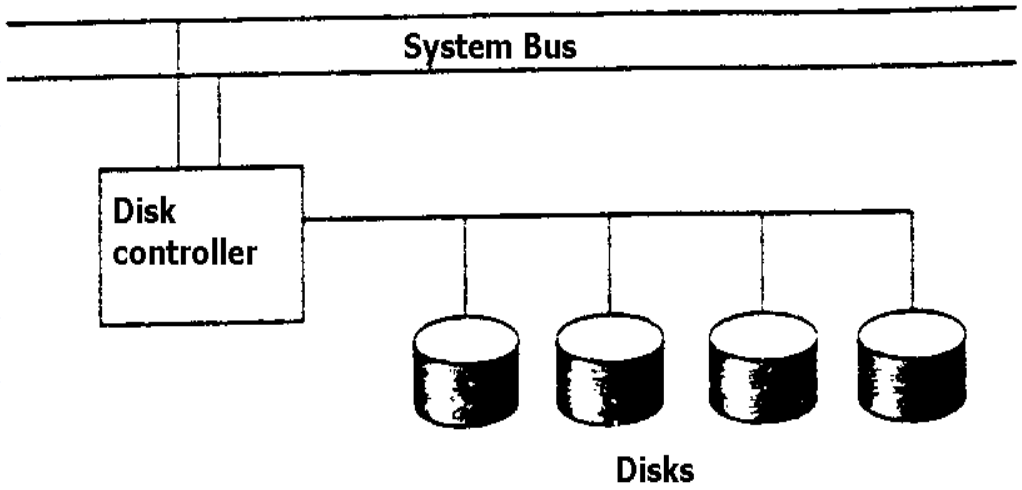
- Khái quát về phương tiện lưu trữ vật lý
- Đĩa từ và RAID
- Lưu trữ tam cấp
- Tổ chức file
- Tổ chức các mẫu tin trong file
- Lưu trữ từ điển dữ liệu

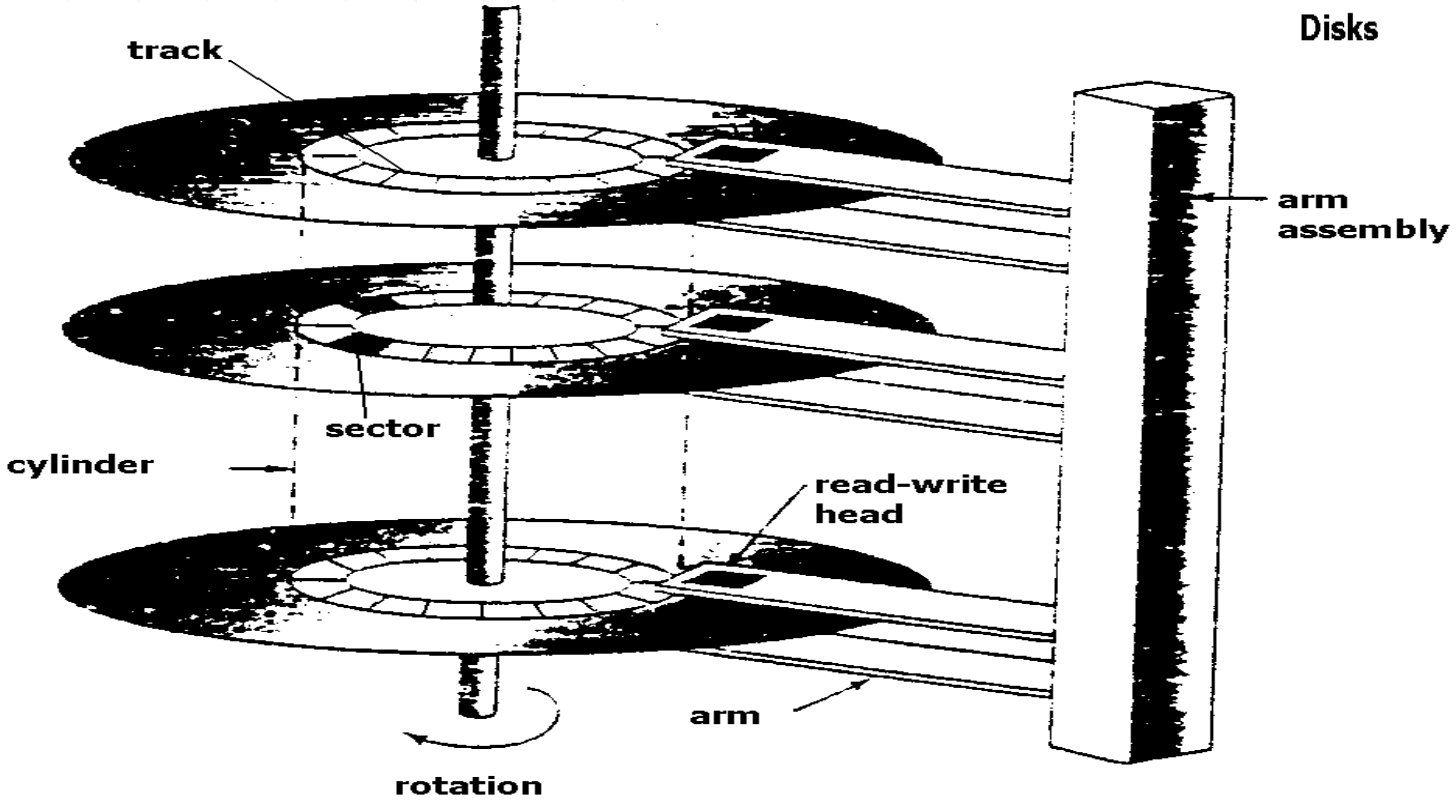
Khái quát về phương tiện lưu trữ vật lý



Đĩa từ (1)

- Cấu tạo





Đĩa từ (2)

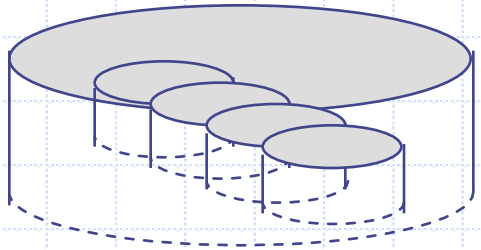
- Đo lường hiệu năng:
  - Thời gian truy xuất: là khoảng thời gian từ khi yêu cầu I/O đến khi dữ liệu bắt đầu được truyền.
    - Thời gian tìm kiếm: là khoảng thời gian chờ đầu đọc di chuyển đúng rãnh chứa sector cần truy xuất (trung bình là 1/3 thời gian trong trường hợp xấu nhất. Seagate: 8.5ms)
    - Thời gian tiềm ẩn luân chuyển: thời gian chờ sector xuất hiện dưới đầu đọc kể từ lúc đầu đọc định vị đúng rãnh (trung bình là thời gian đĩa quay ½ vòng).
  - Tốc độ truyền dữ liệu: là tốc độ dữ liệu có thể được lấy ra/ghi vào đĩa. ATA: 100→133Mbps, SATA: 150→600Mbps.
  - Thời gian trung bình không sự cố: thời gian TB hệ thống hoạt động liên tục mà không gặp bất kỳ sự cố nào, 3.5 → 91 năm.
  - Dung lượng.

Đĩa từ (3)

- Tối ưu hóa truy xuất khối đĩa:
  - Yêu cầu I/O sinh ra bởi hệ thống file và bộ quản trị bộ nhớ ảo
  - Đơn vị dữ liệu truyền giữa đĩa và bộ nhớ là khối, khối sẽ được tính ra số trụ của mặt + sectors.
  - Tốc độ truy xuất đĩa chậm hơn bộ nhớ → cần thiết có một chiến lược nâng cao tốc độ truy xuất khối đĩa:
    - Scheduling (lập lịch biểu).
    - Tổ chức tập tin.
    - Buffer.
    - Đĩa log.

RAID (1)

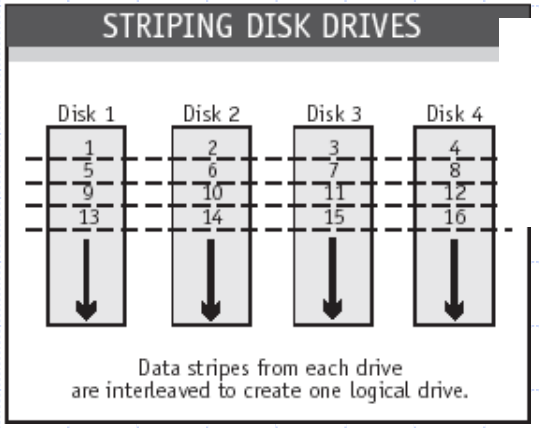
- **R**edundant **A**rray of **I**nexpensive **D**isks
- Ý tưởng:

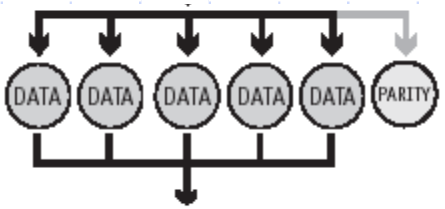


- Mục đích: đạt được hiệu năng và độ tin cậy cao hơn.
- Cách thức:

<u>Hiệu năng:</u> thông qua sự song song

- **Stripping**
- **Mirroring**





Mirror

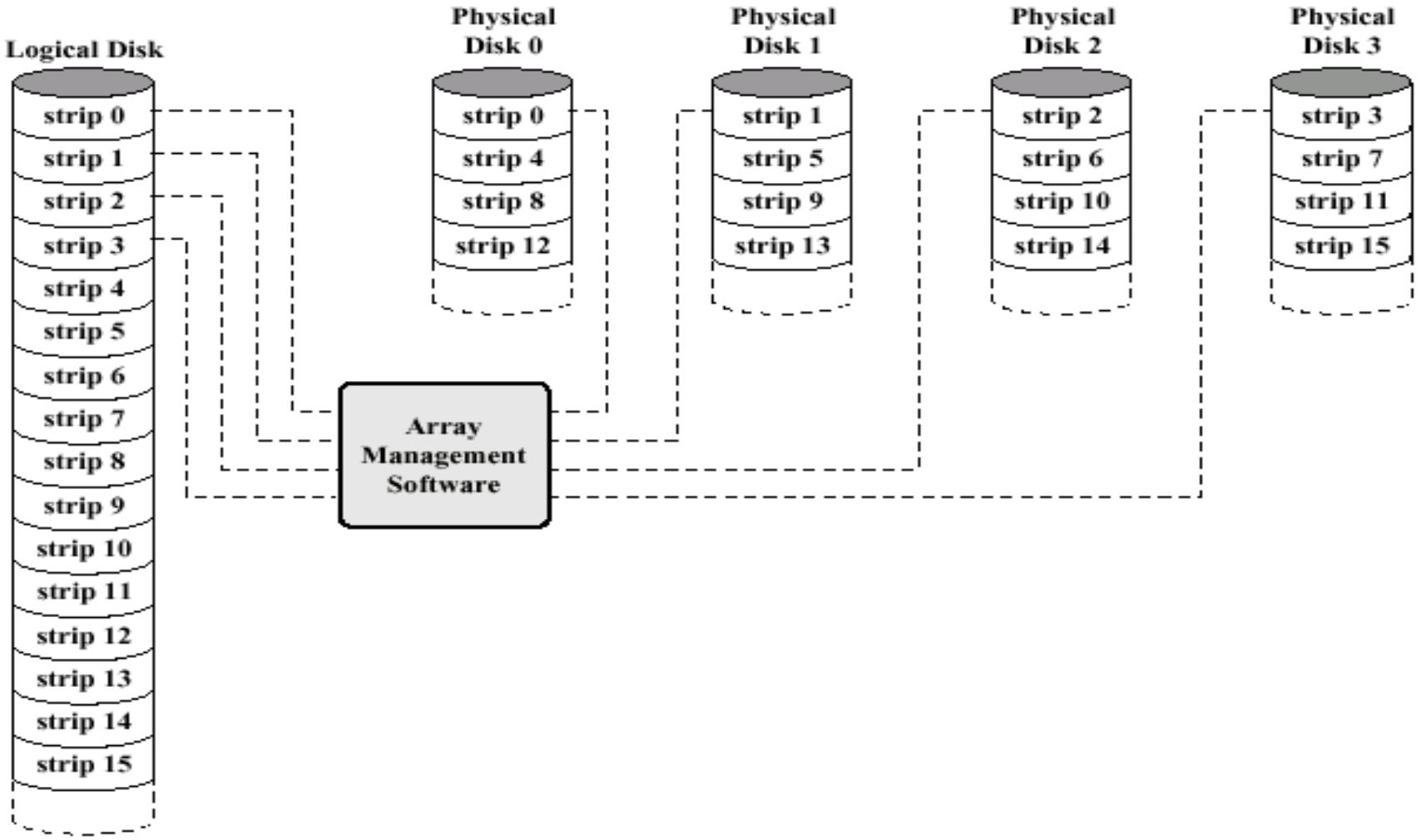
ECC

Parity

<u>Độ tin cậy:</u>

- **Sự dư thừa**

RAID (2)

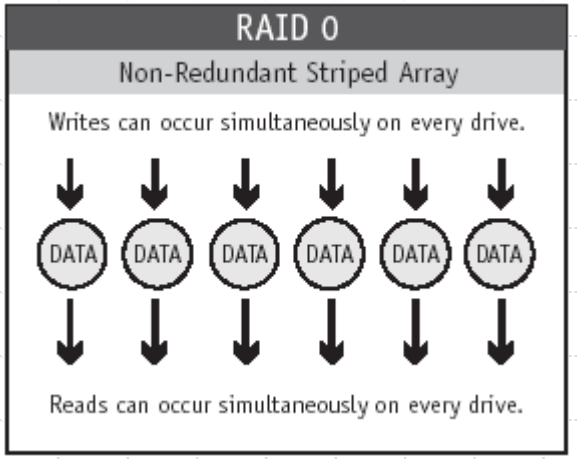

Logical Disk
strip 0
strip 1
strip 2
strip 3
strip 4
strip 5
strip 6
strip 7
strip 8
strip 9
strip 10
strip 11
strip 12
strip 13
strip 14
strip 15
Physical Disk 0
strip 0
strip 4
strip 8
strip 12
Physical Disk 1
strip 1
strip 5
strip 9
strip 13
Physical Disk 2
strip 2
strip 6
strip 10
strip 14
Physical Disk 3
strip 3
strip 7
strip 11
strip 15
Array Management Software

RAID (3)

- **RAID 0 - Stripping**
  - Liên quan đến các dàn đĩa với sự phân nhỏ mức khối.
  - Không có sự dư thừa nào.
  - Do DL được phân mảnh và phân bố trên các đĩa → song song.

  👍 Hiệu năng

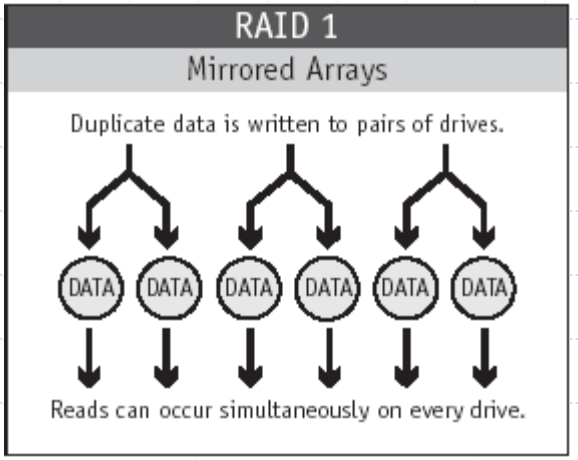
  👎 Độ tin cậy



- **RAID 1 - Mirroring**
  - Liên quan đến mirror đĩa.

  👍
    - Hiệu năng đọc (tăng gấp đôi)
    - Phục hồi

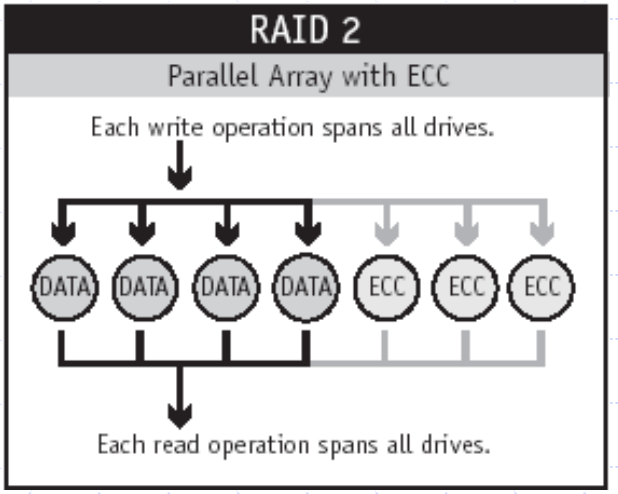
  👎
    - Giá thành



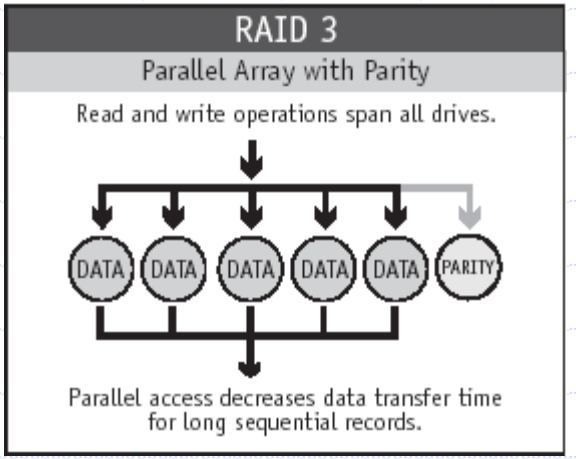
RAID (4)

- **RAID 2 - Hamming code**
  - Phân nhỏ dữ liệu mức bit
  - Lưu m bit sửa lỗi cho dãy n bit chính
  - Các đĩa thêm vào: đĩa overhead
    - Đắt tiền
    - Các ổ đĩa hiện đang sử dụng ECC

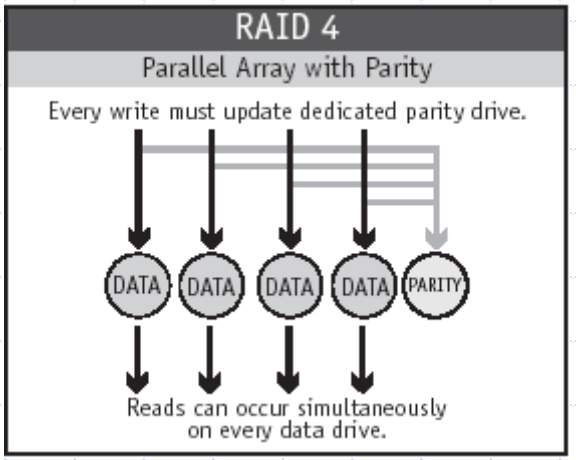
      → Không được hỗ trợ bởi Adapter RAID controllers



- **RAID 3 - Striping & Parity**
  - Phân nhỏ dữ liệu mức bit
  - Dùng 1 đĩa overhead để chứa bit parity
    - Bottleneck đĩa parity
    - Chỉ cho phép 1 yêu cầu IO tại 1 thời điểm

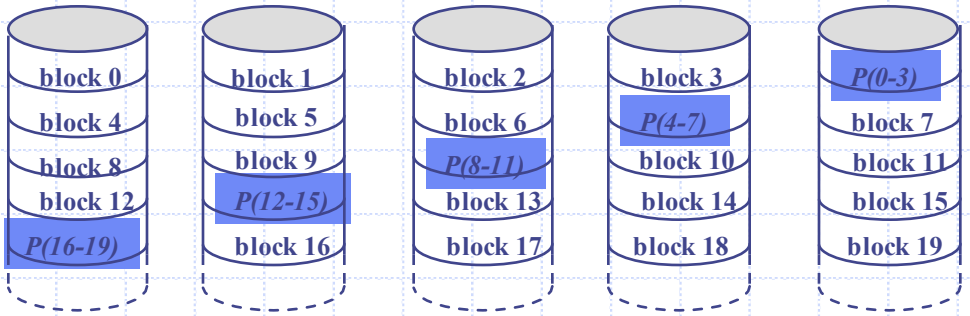


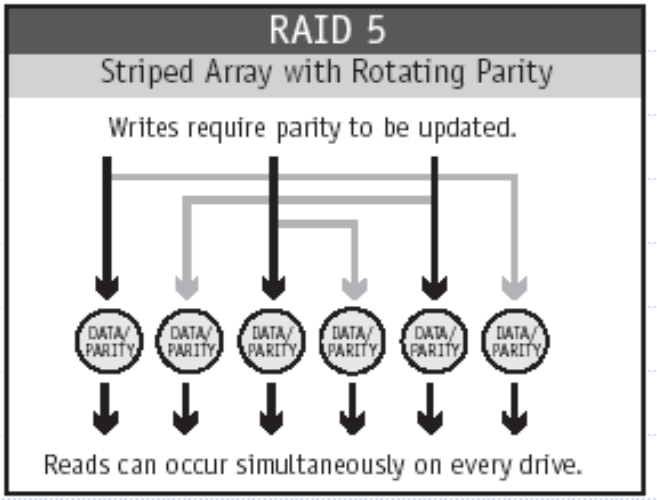
RAID (5)



- **RAID 4 - Striping & Parity**
  - Giống RAID 3
  - Phân nhỏ dữ liệu mức khối
    - Bottleneck đĩa parity

- **RAID 5 - Distributed Parity**
  - Cải tiến mức 4 bởi phân hoạch dữ liệu và parity giữa toàn bộ n+1 đĩa
    - Tránh bottleneck đĩa Parity



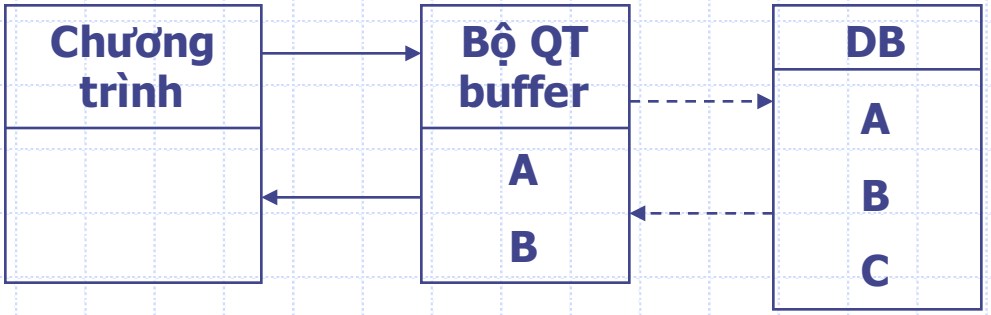


RAID (6)

- Các nhân tố ảnh hưởng đến việc Lựa chọn mức RAID:
  - Chi phí
  - Hiệu năng
  - Hiệu năng khi hệ thống bị sự cố
  - Hiệu năng khi phục hồi
- Lựa chọn:
  - RAID 0: không đòi hỏi tính an toàn của dữ liệu
  - RAID 2, 4: ít sử dụng, được thay thế bởi 3 và 5
  - RAID 3: hiện không còn được sử dụng vì bit-stripping có nhiều hạn chế so với block-stripping (RAID 5)
  - RAID 5: hiệu năng cao hơn RAID 4 (thay thế cho RAID 4) do tránh được nghẽn tại parity disk.

Truy xuất lưu trữ

- Mục tiêu nổi trội của DBMS là tối thiểu hóa số khối truyền giữa đĩa và bộ nhớ. Một cách thực hiện là trữ sẵn trong RAM
- Tuy nhiên, RAM không đủ để chứa tất cả DB → buffer.
- Hệ thống quản lý buffer được gọi là bộ quản trị buffer.
- Bộ quản trị buffer:
  - Chiến lược thay thế
  - Khối chốt
  - Xuất bắt buộc các khối



- Các đối sách thay thế:
  - LRU: thay thế khối đã được dùng lâu nhất.
  - MRU: thay thế khối được dùng gần đây nhất.
  - Các thông tin khác: xác suất sử dụng (tự điển DL, index), thông tin từ các thành phần khác (bộ điều khiển cạnh tranh, hệ thống PH)...

Tổ chức file (1)

- Một tập tin được tổ chức như một dãy các mẩu tin.
- Có 2 loại mẩu tin:
  - **Các mẫu tin có độ dài cố định**
  - **Các mẫu tin có độ dài thay đổi**
- Ta sẽ xét các phương pháp lưu trữ các mẩu tin vào trong file cho từng loại mẩu tin.

Tổ chức file (2) - Mẫu tin độ dài cố định

- Xét một file các mẫu tin account trong CSDL ngân hàng:

```
type deposit=record
        branch_name: char(20);        //tên chi nhánh
        account_number: char(10);     //số tài khoản
        balance: real;                //số dư
end;
```

- Các mẫu tin được lưu liên tiếp trong tập tin CSDL

- Tập tin chứa các mẫu tin account

0	Perryridge	A-102	400
1	Round hill	A-305	350
2	Mianus	A-215	700
3	Downtown	A-101	500
4	Redwood	A-222	700
5	Perryridge	A-201	900
6	Brighton	A-217	750
7	Downtown	A-110	600
8	Perryridge	A-218	700

Tổ chức file (3) - Mẫu tin độ dài cố định

0	Perryridge	A-102	400
1	Round hill	A-305	350
3	Downtown	A-101	500
4	Redwood	A-222	700
5	Perryridge	A-201	900
6	Brighton	A-217	750
7	Downtown	A-110	600
8	Perryridge	A-218	700

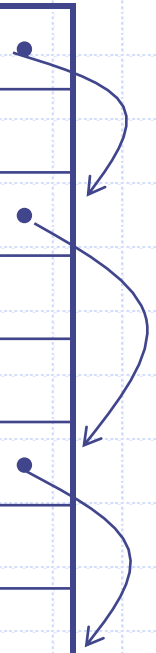
Xoá mẫu tin 2 và chuyển tất cả mẫu tin còn lại lên

0	Perryridge	A-102	400
1	Round hill	A-305	350
8	Perryridge	A-218	700
3	Downtown	A-101	500
4	Redwood	A-222	700
5	Perryridge	A-201	900
6	Brighton	A-217	750
7	Downtown	A-110	600

Xoá mẫu tin 2 và chuyển mẫu tin cuối cùng lên

Tổ chức file (4) - Mẫu tin độ dài cố định

Header				
0	Perryridge	A-102	400	
1				
2	Mianus	A-215	700	
3	Downtown	A-101	500	
4				
5	Perryridge	A-201	900	
6				
7	Downtown	A-110	600	
8	Perryridge	A-218	700	



- Dùng con trỏ để quản lý các mẫu tin bị xoá

Tổ chức file (5) - Mẫu tin độ dài thay đổi

- Các mẫu tin có độ dài thay đổi tồn tại trong CSDL theo nhiều cách:
  - Lưu nhiều kiểu mẫu tin trong cùng một file
  - Các kiểu mẫu tin cho phép các trường có độ dài thay đổi
  - Các kiểu mẫu tin cho phép lặp lại các trường
- Ví dụ:

```
type account-list=record
        branch-name: char(22);
        account-info: array[1..∞] of
                record
                        account_number: char(10);
                        balance: real;
                        end;
        end;
```

Tổ chức file (6) - Mẫu tin độ dài thay đổi

- Biểu diễn theo chuỗi byte:

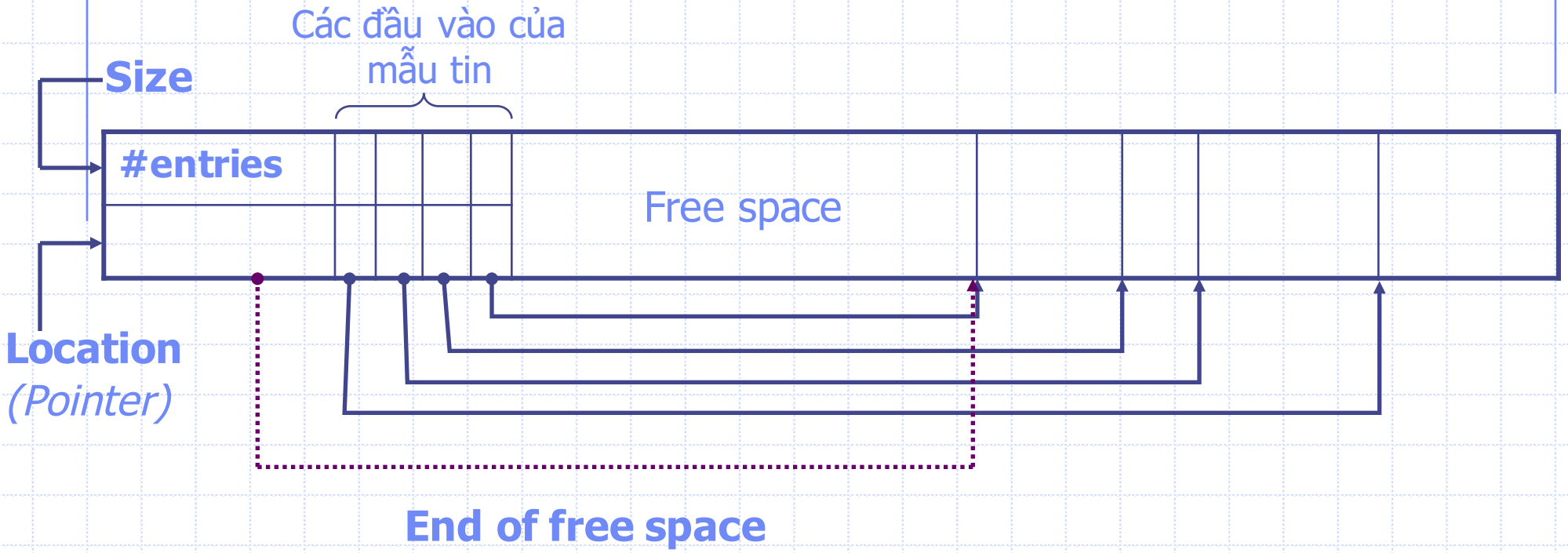
0	Perryridge	A-102	400	A-201	900	A-210	700	⊥
1	Round Hill	A-310	350	⊥				
2	Mianus	A-110	800	⊥				
3	Downtown	A-211	500	A-222	600	⊥		
4	Redwood	A-300	650	A-200	1200	A-255	950	⊥
5	Brighton	A-111	750	⊥				

End of record

- Khó sử dụng lại không gian trống.
- Không có không gian phát triển cho mẫu tin.

Tổ chức file (7) - Mẫu tin độ dài thay đổi

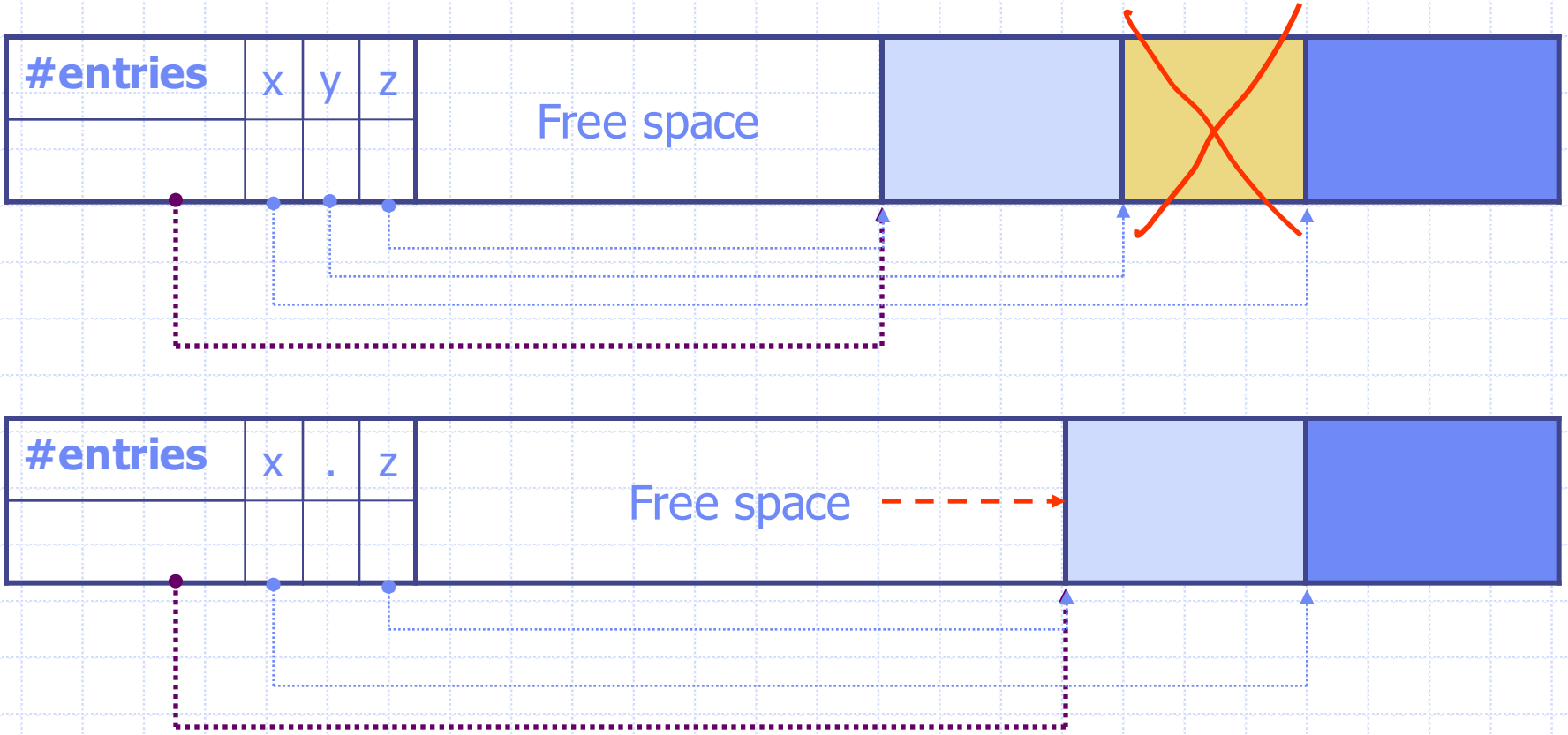
- Dùng cấu trúc khe trang:



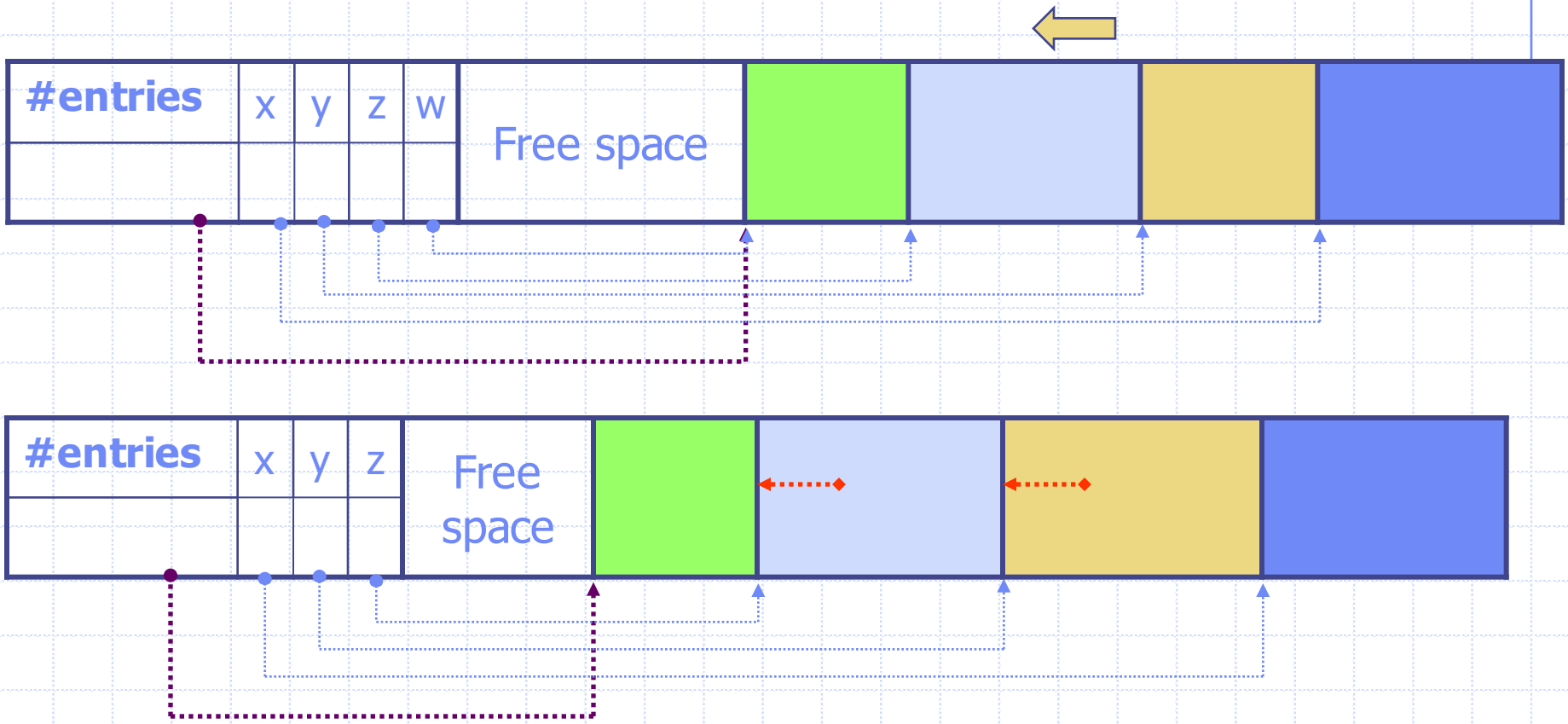
- Giá phải trả cho sự di chuyển các mẩu tin không quá cao.

Tổ chức file (8) - Mẫu tin độ dài thay đổi

Tổ chức file (9) - Mẫu tin độ dài thay đổi



Tổ chức file (10) - Mẫu tin độ dài thay đổi



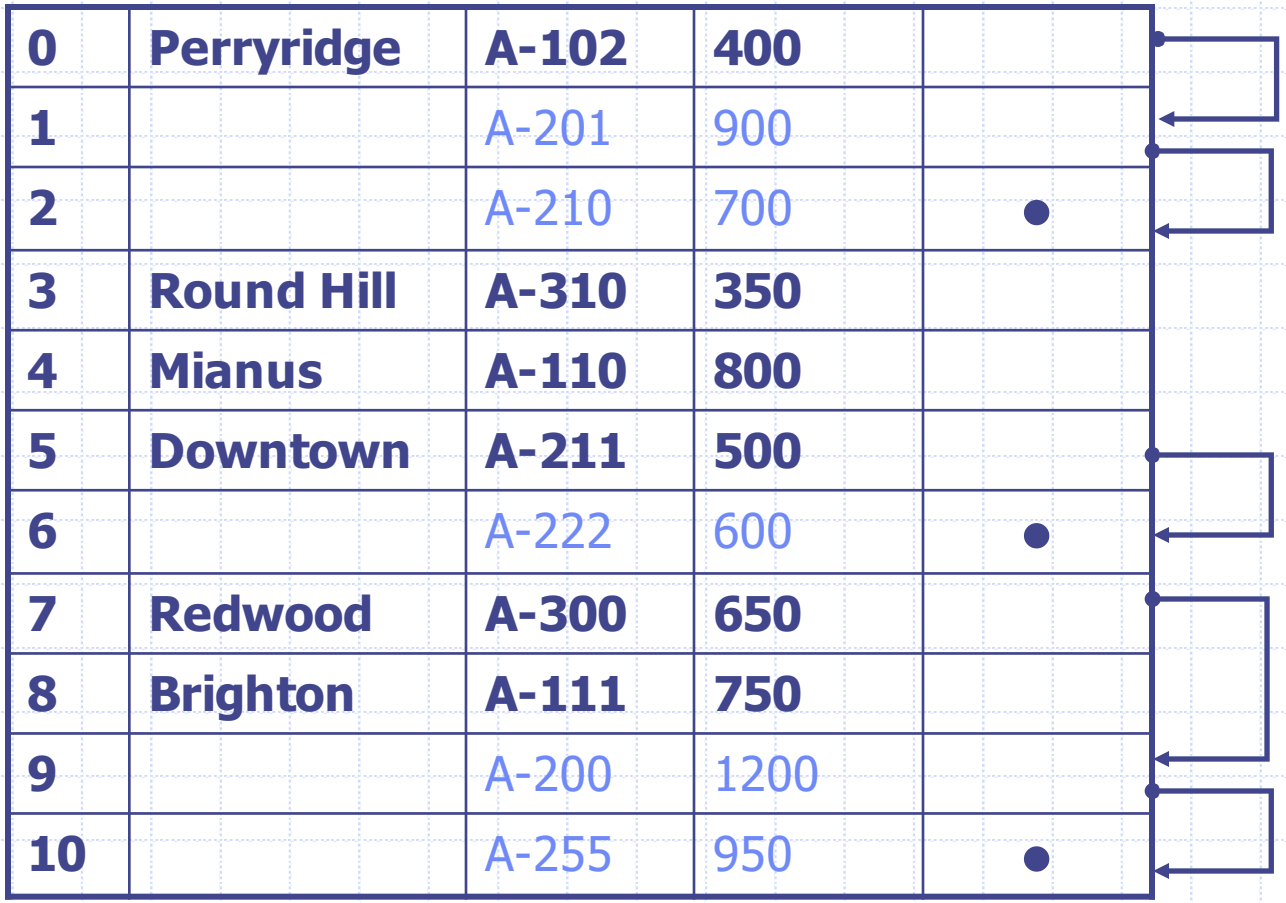
Tổ chức file (11) - Mẫu tin độ dài thay đổi

- Dùng không gian dự trữ:

0	Perryridge	A-102	400	A-201	900	A-210	700	⊥
1	Round Hill	A-310	350	⊥	⊥	⊥	⊥	⊥
2	Mianus	A-110	800	⊥	⊥	⊥	⊥	⊥
3	Downtown	A-211	500	A-222	600	⊥	⊥	⊥
4	Redwood	A-300	650	A-200	1200	A-255	950	⊥
5	Brighton	A-111	750	⊥	⊥	⊥	⊥	⊥

Tổ chức file (12) - Mẫu tin độ dài thay đổi

- Dùng phương pháp con trỏ:



0	**Perryridge**	**A-102**	**400**	
1		A-201	900	
2		A-210	700	●
3	**Round Hill**	**A-310**	**350**	
4	**Mianus**	**A-110**	**800**	
5	**Downtown**	**A-211**	**500**	
6		A-222	600	●
7	**Redwood**	**A-300**	**650**	
8	**Brighton**	**A-111**	**750**	
9		A-200	1200	
10		A-255	950	●

Tổ chức file (13) - Mẫu tin độ dài thay đổi

- Dùng phương pháp khối neo và khối tràn:

Khối neo

Perryridge	A-102	400	
Round Hill	A-310	350	
Mianus	A-110	800	
Downtown	A-211	500	
Redwood	A-300	650	
Brighton	A-111	750	

Khối tràn

A-201	900	
A-210	700	•
A-222	600	•
A-200	1200	
A-255	950	•

Tổ chức các mẫu tin trong file (1)

- Tổ chức file đống (heap):
  - Một mẫu tin có thể được lưu ở bất kỳ vị trí nào trong file, ở đó có không gian cho nó
  - Không có thứ tự nào cho các mẫu tin.
  - Một file cho một quan hệ
- Tổ chức file tuần tự (sequential):
  - Các mẫu tin được lưu trữ tuần tự dựa trên giá trị của khoá tìm kiếm của mỗi mẫu tin

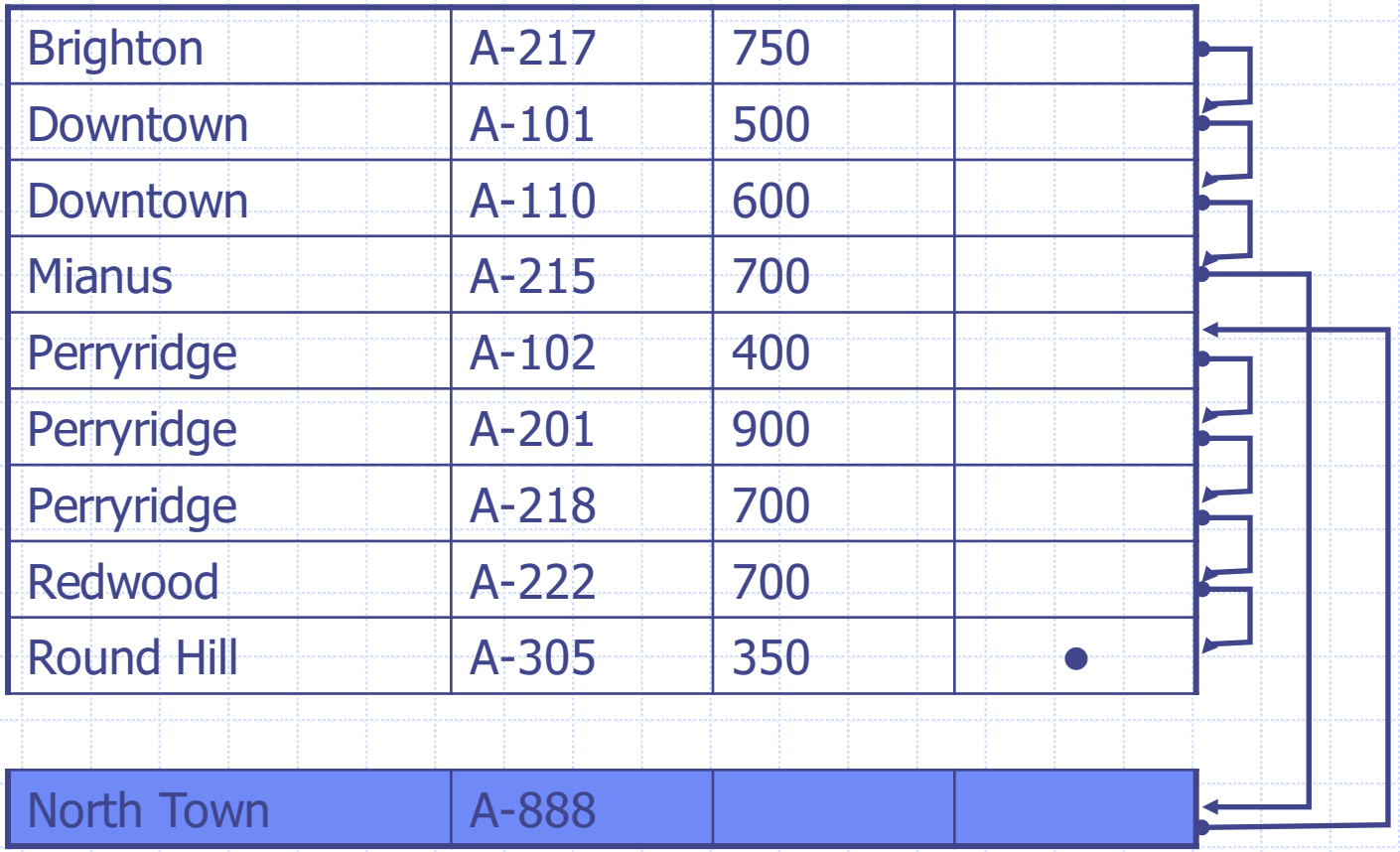
Tổ chức các mẫu tin trong file (1)

- Tổ chức file băm (hashing):
  - Tập tin được chia thành nhiều "ngăn" (block)
  - Một hàm băm trên mẩu tin được sử dụng để xác định vị trí của mẫu tin sẽ được lưu trữ trong ngăn nào
- Tổ chức file cụm (clustering):
  - Các mẫu tin của một vài quan hệ được lưu trữ trong cùng một file
  - Các mẫu tin có liên hệ của vài quan hệ khác nhau được lưu trên cùng một khối sao cho hoạt động I/O đem lại các mẫu tin có liên hệ từ tất cả các quan hệ

Tổ chức file tuần tự

Brighton	A-217	750	
Downtown	A-101	500	
Downtown	A-110	600	
Mianus	A-215	700	
Perryridge	A-102	400	
Perryridge	A-201	900	
Perryridge	A-218	700	
Redwood	A-222	700	
Round Hill	A-305	350	•

Tổ chức file tuần tự



- Thêm một record vào khối tràn

Tổ chức file cụm

- Quan hệ ***depositor***

Customer-name	***Account-number***
Hayes	A-102
Hayes	A-220
Hayes	A-503
Turner	A-305

- Quan hệ ***customer***

Customer-name	***Customer-street***	***Customer-city***
Hayes	Main	Brooklyn
Turner	Putnam	Standford

Tổ chức file cụm

- Giả sử người dùng đặt ra câu vấn tin:

```
SELECT account-number, customer-name,
       customer-street, customer-city
FROM depositor d, customer c
WHERE d.customer-name = c.customer-name
```

- Câu vấn tin này chính là phép nối của các quan hệ *customer và depositor*

- Ta sẽ trình bày một cấu trúc file được thiết kế để thực hiện hiệu quả các câu vấn tin trên

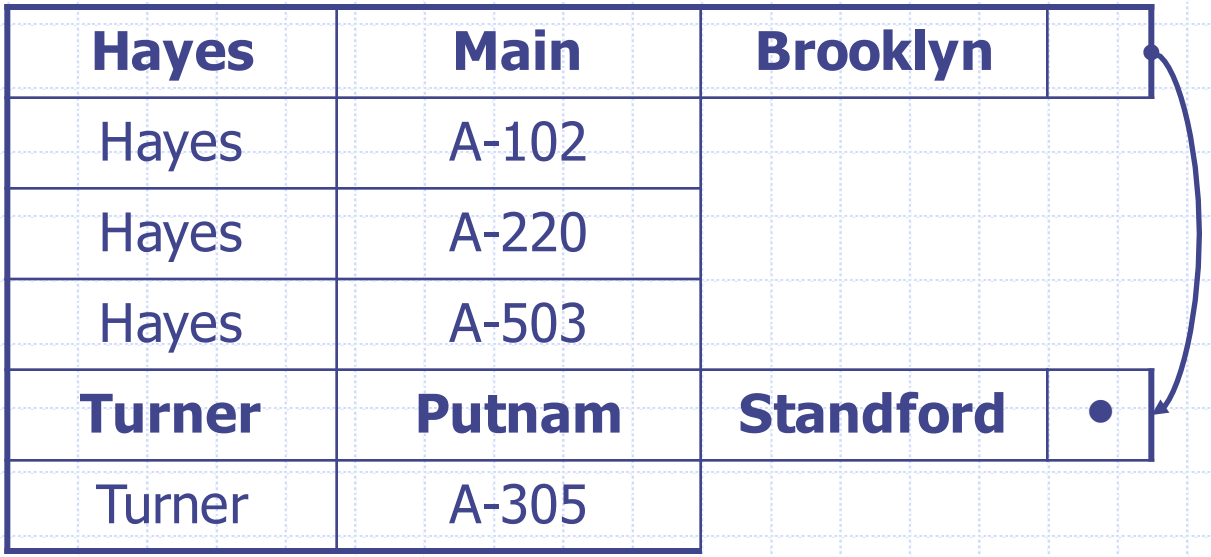
Tổ chức file cụm

- Cấu trúc file cụm cho hai quan hệ *customer* và *depositor*

Hayes	**Main**	**Brooklyn**
Hayes	A-102	
Hayes	A-220	
Hayes	A-503	
Turner	**Putnam**	**Standford**
Turner	A-305	

Tổ chức file cụm

- Cấu trúc file cụm với dây chuyền con trỏ



Hayes	**Main**	**Brooklyn**	
Hayes	A-102		
Hayes	A-220		
Hayes	A-503		
Turner	**Putnam**	**Standford**	•
Turner	A-305		

Lưu trữ từ điển dữ liệu

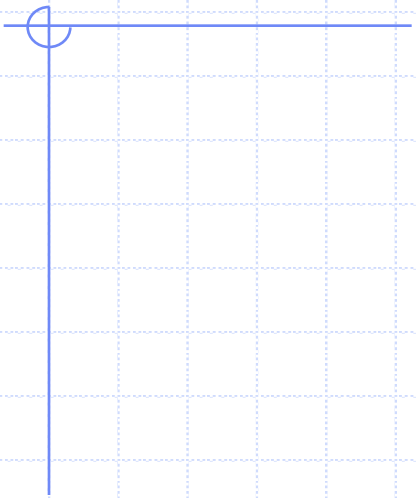
System_catalog_schema=(relation_name, number_of_attributes)

Attribute_schema=(attribute_name, relation_name, domain_type, position, length)

User_schema=(user_name, encrypted_password, group)

Index_chema=(index_name, relation_name, index_type, index_attributes)

View_schema=(view_name, definition)



Questions?