CHƯƠNG 5

QUẢN TRỊ CÁC GIAO TÁC PHÂN TÁN

NỘI DUNG

5.1. TỔNG QUAN VỀ QUẢN LÝ GIAO TÁC

5.2. SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

5.3. SỰ PHỤC HỒI TRONG HỆ THỐNG TẬP TRUNG

5.4. CÁC SỰ CỐ TRUYỀN THÔNG TRONG HỆ PHÂN TÁN

5.5. KHÔI PHỤC CÁC GIAO TÁC PHÂN TÁN

5.6. GIAO THỨC UỶ THÁC HAI PHA

CHƯƠNG 5: QUẢN LÝ CÁC GIAO TÁC PHÂN TÁN

MỤC ĐÍCH

1. Nhằm quản lý một số vấn đề trong quá trình truyền thông của hệ phân tán như:

- *Độ tin cậy* (reliabity)
- *Điều khiển tương tranh* (concurrency control)
- *Hiệu quả sử dụng các tài nguyên của hệ thống*.

Và ...

1. Hiểu được việc quản lý giao tác phân tán là điều cần thiết để hiểu được sự liên quan giữa điều khiển tương tranh, cơ chế phục hồi và cấu trúc của hệ thống phân tán.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Giao tác phân tán?

Giao tác là một lần thực hiện của một chương trình.

Chương trình có thể là:

- một câu truy vấn
- một chương trình ngôn ngữ chủ với các lời gọi được gắn vào một ngôn ngữ vấn tin.

Ví dụ:

```
(T1): Begin
          read(a);
          a:=a+100;
          read(a); a:=a+2;
          write(a);
      end
```

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Hai giao tác cơ sở:

- Đọc dữ liệu từ CSDL : read(x)
- Ghi dữ liệu vào CSDL: write(x)

Chú ý:

Khi đọc hoặc ghi dữ liệu vào cơ sở dữ liệu các giao tác sẽ sử dụng một không gian làm việc riêng (private workspace) để thực hiện các thao tác tính toán.

Các thao tác tính toán này sẽ không ảnh hưởng đến cơ sở dữ liệu.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Ví dụ: xét 2 giao tác T1 và T2:

```
(T1):Begin
        read(a);
        a:=a+100;
        read(a);
        a:=a+2;
        write(a)
    end
```

```
(T2):Begin
        read(a);
        a:=a+100;
        write(a);
        read(a);
        a:=a+2;
        write(a)
    end
```

Nhận xét:

- Ở giao tác T1 giá trị của biến **a** chỉ được tăng lên 2 vì lệnh **a:=a+100** được thực hiện trong không gian riêng mà không ảnh hưởng đến cơ sở dữ liệu.
- Ở giao tác T2 giá trị của biến **a** chỉ được tăng thêm 102.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

5.1.1 Các tính chất của giao tác

-Tính nguyên tử (Atomicity)

-Tính bền vững (Durability)

-Tính tuần tự (Serializability)

-Tính biệt lập (Isolation)

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Tính nguyên tử

- Tính nguyên tử của một giao tác là sự thực hiện trọn vẹn mà không một giao tác nào được chen vào.
- Khi thực thi một giao tác thì hoặc là các hành động của giao tác đó được thực hiện hoặc là không một hành động nào được thực hiện cả.
- Tính nguyên tử đòi hỏi rằng nếu việc thực thi giao tác bị cắt ngang bởi một loại sự cố nào đó thì DBMS sẽ chịu trách nhiệm xác định những công việc của giao tác để khôi phục lại sau sự cố.
- Có 2 chiều hướng thực hiện:
  - hoặc nó sẽ được kết thúc bằng cách hoàn tất các hành động còn lại,
  - hoặc có thể kết thúc bằng cách hồi lại tất cả các hành động đã được thực hiện.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

02 lý do cơ bản làm cho giao tác không hoàn thành:

- giao tác tự huỷ bỏ (**transaction aborts)**
- hệ thống bị sự cố (**system crashes)**

. Tại sao giao tác tự hủy?

- Do yêu cầu của bản thân giao tác hoặc của người sử dụng nó.
- Do sự ép buộc của hệ thống:
  - quá tải hệ thống
  - bị kẹt trong một khoá gài (deadlock).

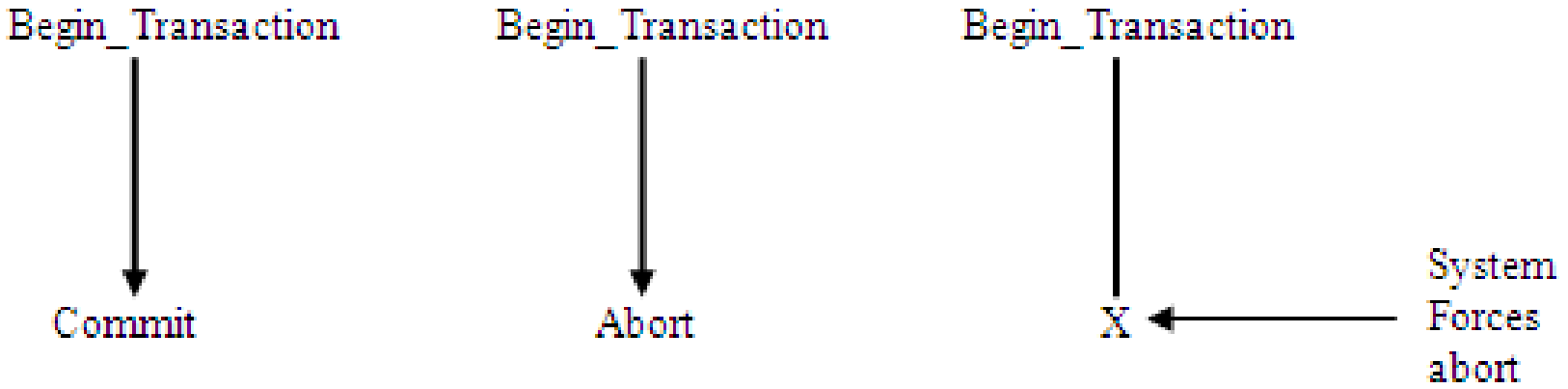
5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Một số khái niệm:

- ***Khôi phục giao tác*** **(transaction recovery)**: duy trì được tính nguyên tử khi có sự cố mà giao tác tự huỷ bỏ.
- ***Khắc phục sự cố*** **(crash recovery).**

duy trì được tính nguyên tử khi có sự cố hệ thống

- ***Uỷ thác*** (**commitment):** Sự hoàn thành một giao tác



Hình 5.1. Các dạng hoàn thành giao tác

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Tính bền vững

Mục đích:

Để bảo đảm rằng mỗi khi giao tác uỷ thác, kết quả của nó sẽ được duy trì và không bị xoá ra khỏi CSDL.

DDBMS có trách nhiệm bảo đảm kết quả của giao tác và ghi vào CSDL.

Tính bền vững được sử dụng như là một điều kiện để *khôi phục dữ liệu* **(database recovery**), nghĩa là cách khôi phục CSDL về trạng thái nhất quán mà ở đó mọi hành động đã uỷ thác đều được phản ánh.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Tính tuần tự (Serializability):

Mục đích:
Quản lý vấn đề *thực hiện đồng thời của hai hoặc nhiều giao tác*.

Yêu cầu:
Nếu có nhiều giao tác thực hiện đồng thời, thì kết quả phải như nhau nếu nó được thực hiện tuần tự trong cùng một thứ tự đó.

Nếu một hệ thống có tính điều khiển đồng thời, người lập trình có thể ghi lại giao tác như khi nó thực hiện một mình.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Tính biệt lập

Mục đích:

Tính chất này để ngăn ngừa sự *hủy bỏ dây chuyền* (cascading abort - Còn gọi là hiệu ứng domino).

Chú ý:

- Một giao tác đang thực thi không thể đưa ra các kết quả của nó cho những giao tác khác đang cùng hoạt động trước khi nó uỷ thác.
- Nếu một giao tác cho phép những giao tác khác sử dụng những kết quả chưa hoàn tất của mình trước khi uỷ thác, rồi sau đó nó quyết định huỷ bỏ, thì mọi giao tác đã đọc những giá trị chưa hoàn tất đó cũng sẽ phải được huỷ bỏ nếu không xâu mắt xích này dễ dàng tăng nhanh và gây ra những phí tổn đáng kể cho DDBMS.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Ví dụ: Xét 2 giao tác đồng thời T_1 và T_2 cùng truy xuất đến mục dữ liệu *x*, Giả sử giá trị của *x* trước khi bắt đầu thực hiện là 50.

```
T1:  Read(x)
     x:= x + 1
     Write(x)
     Commit
```

```
T2:  Read(x)
     x:= x + 1
     Write(x)
     Commit
```

Điều gì sẽ xảy ra nếu:

a. Hai giao tác thực hiện tuần tự
b. Hai giao tác thực hiện đồng thời

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

a. **2 giao tác thực hiện tuần tự:**

dãy thực thi cho các hành động của 2 giao tác là:

```
T1: x:= x + 1
T1: Write(x)
T1: Commit
T2: Read(x)
T2: x:= x + 1
T2: Write(x)
```

Nhận xét:

- **Giá trị ban đầu của x là 50**
- **Giá trị của x sau khi T1 uỷ thác là 51**
- **Giá trị của x sau khi T2 uỷ thác là 52**

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

b. **Các giao tác thực thi đồng thời**

dãy hành động được thực hiện như sau có thể xảy ra:

```
T1:  Read(x)
T1: x:=x + 1
T2: Read(x)
T1: Write(x)
T2: x:=x + 1
T2: Write(x)
T1: Commit
T2: Commit
```

Nhận xét:

- **Giá trị ban đầu của x là 50**
- **Giá trị của x sau khi T1 uỷ thác là 51**
- **Giá trị của x sau khi T2 uỷ thác là 51**

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Thí dụ (về sự quan trọng của tính biệt lập)

Xét 02 giao tác được thực hiện lần lượt ở một hệ thống kế toán ngân hàng như sau:

1.Giao tác T1 chuyển $1000 vào một tài khoản hiện có $0.

2.Giao tác T2 đọc quyết toán $1000 được ghi bởi T1 trước khi T1 hoàn tất và ghi nợ $1000 vào cùng một tài khoản.

3.T2 hoàn tất, và tiền mặt $1000 được chuyển cho người dùng người như đã yêu cầu thực hiện T2.

4.T1 bị huỷ bỏ vì lý do một thao tác nào đó bất hợp lệ.

5.Việc hủy bỏ T1 yêu cầu phải hủy bỏ T2, bởi vì thao tác thực hiện bởi T2 dựa vào các thao tác thực hiện bởi T1.

6.Tuy nhiên, việc hủy bỏ T2 là không thể được vì hậu quả của T2 trong thế giới thực không thể được hoàn lại bởi hệ thống.

5.1.2 Quản lý khoá trong giao tác

Mục dữ liệu (**data item**):

- là những đơn vị dữ liệu cần được truy xuất có điều khiển
- Bản chất và kích thước của mỗi mục dữ liệu được lựa chọn tuỳ theo mục tiêu của bài toán
- Trong điều khiển đồng thời phân tán, CSDL phải được phân nhỏ thành các mục dữ liệu
- Phương pháp thông dụng nhất để điều khiển việc truy xuất các mục là sử dụng ***khoá chốt*** (**lock**).

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Khoá chốt

• Khoá chốt là một đặc quyền truy xuất trên một mục dữ liệu mà mà *bộ quản lý khoá chốt* (lock manager) uỷ quyền cho một giao tác nào đó hoặc thu hồi lại.

• Bộ quản lý khoá chốt là một thành phần cơ bản của DDBMS, chịu trách nhiệm theo dõi xem một mục dữ liệu nào đó hiện đang chịu sự đọc/ghi của một hoặc nhiều giao tác.

Ví dụ dưới đây cho chúng ta thấy vai trò quan trọng của khoá chốt.

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

Ví dụ: Tại cùng một thời điểm, hai đại lý bán vé của Việt nam Airline đang bán cho 2 khách hàng 2 vé đi HCMC-HUẾ. Hai thao tác này được mô tả qua 2 giao tác T_1 và T_2 có chung một chương trình P như dưới đây. Các giao tác này cùng truy xuất đến mục dữ liệu *x (chẳng hạn, x là số vé đã bán được của chuyến bay)*. Giả sử, giá trị của *x* trước khi bắt đầu thực hiện các giao tác là 50 (đã bán được 50 vé).

```
T1:  Read(x)
     x:=x+1
     Write(x)
     Commit
```

```
T2:  Read(x)
     x:=x+1
     Write(x)
     Commit
```

5.1 TỔNG QUAN VỀ QUẢN LÝ CÁC GIAO TÁC

- Điều gì sẽ xảy ra nếu 2 giao tác T1 và T2 thực hiện đồng thời với các mã lệnh của P được xen kẻ như sau:

```
T1: Read(x)
T2: Read(x)
T1: x:=x+1
T2: x:=x+1
T1: Write(x)
T2: Write(x)
T1: Commit
T2: Commit
```

- **Đã bán thêm 2 vé nhưng trong CSDL chỉ ghi 1 vé.**

- **Làm thế nào để giải quyết ?**

- Sử dụng khoá chốt.
- Nghĩa là, trước khi T1 đọc x phải khoá x lại (lock x), ngăn không cho các giao tác khác truy xuất x cho đến khi T1 hoàn thành.

P: Lock(x); Read(x); x:=x + 1; Write(x); Unlock(x);

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

5.2.1 Các sự cố trong các CSDL tập trung

Phân loại các lỗi:

a.Các sự cố không làm mất thông tin: <dễ phục hồi>

•Các thông tin được cất giữ trong bộ nhớ và sẵn sàng cho sự phục hồi.

•Chẳng hạn như, việc hủy các giao tác vì một điều kiện lỗi bị phát hiện, như lỗi tràn số hoặc lỗi chia cho zero.

b. Các sự cố có mất thông tin tạm thời:

•Nội dung của bộ nhớ chính bị mất; tuy nhiên

•Các thông tin được ghi lại trên đĩa không bị ảnh hưởng bởi lỗi.

c. Các sự cố có mất thông tin lưu trữ: <ít xảy ra>

•Được gọi là các lỗi môi trường

•Nội dung của bộ đĩa lưu trữ cũng bị mất.

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

Nhận xét:

Sự cố có mất thông tin lưu trữ ít xảy ra

Cách giải quyết:

⇒tạo bản sao thông tin trên vài đĩa độc lập với các phương thức lỗi.

⇒*Sử dụng bộ lưu trữ ổn định: một bộ nhớ* trung gian để phục hồi dữ liệu.

d. *Các sự cố làm mất dữ liệu lưu trữ ổn định*:

Ít xảy ra.

- *Backup ổ đĩa*
- *Backup server*

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

5.2.2 Nhật ký (logs):

- Một ***nhật ký*** lưu trữ các thông tin về việc **hủy** hoặc **làm lại** tất cả các hành động được thực hiện bởi các giao tác.

- **Hủy** các hành động của một giao tác có nghĩa là xây dựng lại CSDL trước sự thực hiện của nó.

- **Làm lại** các hành động của một giao tác nghĩa là thực hiện lại các hành động của nó.

- Sự cần thiết của việc hủy các hành động của một giao tác là sự thất bại trước khi sự ủy thác xảy ra;

Các hoạt động hủy và làm lại phải được thay đổi giá trị, tức là:

UNDO(UNDO(UNDO(…(hành động)…))) = UNDO(hành động)

REDO(REDO(REDO(…(hành động)…))) = REDO(hành động)

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

5.2.2 Nhật ký (logs):

Một bản ghi nhật ký bao gồm thông tin được yêu cầu cho việc hủy hoặc việc làm lại các hành động:

- Định danh của giao tác.
- Định danh của bản ghi.
- Kiểu hoạt động của giao tác (chèn, xóa, sửa đổi).
- Giá trị bản ghi cũ (dùng để hoàn lại thao tác).
- Giá trị bản ghi mới (dùng để thực hiện lại thao tác).
- Thông tin phụ dùng cho thủ tục phục hồi.

5.2.3 Các quy trình phục hồi:

If có sự cố có mất thông tin tạm thời xảy ra **then**

Begin

- Đọc file *nhật ký*
- Xác định tất cả các giao tác chưa hoàn tất cần phải hoàn lại {bằng cách tìm các giao tác có begin_transaction nhưng không có commit hoặc abort}
- Xác định tất cả các giao tác cần thực hiện lại.
- Hoàn lại và thực hiện lại các giao tác trên

End

Xem thêm mục 5.2.3. Khôi phục các giao tác phân tán, trang 115-118 Giáo trình

5.2.4 Giao thức ủy thác 2 pha (2-Phase Commitment Protocol)

Giới thiệu về ủy thác phân tán:

- Giả sử có một giao tác T khởi hoạt tại một vị trí và sinh ra nhiều giao tác tại các vị trí khác.
- Thành phần của giao tác T thực hiện tại vị trí gốc được gọi là ***điều phối viên*** (coordinator)
- Các giao tác thực hiện tại các vị trí khác được gọi là các ***thành viên*** (participants)
- *Điều phối viên* chịu trách nhiệm cuối cùng về quyết định ủy thác hoặc hủy bỏ
- Nếu một *thành viên* không thể ủy thác cục bộ giao tác con của nó thì tất cả các *thành viên* khác phải hủy bỏ cục bộ.

5.2.4 Giao thức ủy thác 2 pha (2-Phase Commitment Protocol)

Ý nghĩa: xác định một quyết định duy nhất cho tất cả *thành viên* về việc ủy thác hoặc hủy bỏ tất cả các giao tác con cục bộ.

Pha 1 –Pha biểu quyết (Voting phase)

- *Điều phối viên* yêu cầu tất cả *thành viên* chuẩn bị ủy thác (prepare for commitment**)**;
- Mỗi *thành viên* trả lời READY nếu ở tình trạng sẵn sàng thực hiện các giao tác con cục bộ.
- Trước khi gửi sự chuẩn bị về thông tin ủy thác, *điều phối viên* lưu lại an toàn bằng việc ghi nhật ký (*log record*) của một dạng mới, được gọi là ghi nhật ký **"prepare"**

Điều phối viên: Ghi "prepare" vào nhật ký

Gửi thông điệp PREPARE và kích hoạt thời gian quá hạn

Thành viên

Chờ thông điệp PREPARE

If thành viên sẵn sàng ủy thác **then**

begin

Ghi giao tác con vào nhật ký

Ghi mẫu tin "ready" vào nhật ký

Gửi thông điệp trả lời READY đến điều phối viên

end

Else

begin

Ghi mẫu tin "abort" vào nhật ký

Gửi thông điệp trả lời ABORT đến điều phối viên

end

Điều phối viên

Chờ thông điệp trả lời (READY hoặc ABORT) từ tất cả thành viên hoặc thời gian quá hạn
If thời gian quá hạn đã hết hoặc có vài thông điệp trả lời là ABORT **then**
    Begin
        Ghi mẫu tin "global_abort" vào nhật ký
        Gửi thông điệp ABORT đến tất cả thành viên
    End
Else
    Begin
        Ghi mẫu tin "global_commit" vào nhật ký;
        Gửi thông điệp COMMIT đến tất cả thành viên
    End

Thành viên

Chờ thông điệp
Ghi mẫu tin "abort" hoặc "commit" vào nhật ký
Gửi thông điệp ACK đến điều phối viên
Thi hành lệnh

Điều phối viên

Chờ thông điệp ACK từ tất cả thành viên
Ghi mẫu tin "complete" vào nhật ký

Pha 2 –Pha quyết định (Decision phase)

Điều phối viên: Ghi vào nhật ký các quyết định

Khai báo tất cả *thành viên* đã quyết định, bằng cách gửi cho họ thông điệp.

Thành viên :

Các *thành viên* ghi lệnh **commit** hoặc **abort** vào nhật ký

thành viên gửi thông điệp xác nhận (ACK) đến *điều phối viên*,

Thực hiện những hành động cần thiết cho việc ủy thác hoặc hủy bỏ giao tác con.

Điều phối viên: *điều phối viên* chờ cho đến khi thu được một thông điệp ACK báo hoàn thành ủy thác từ tất cả các *thành viên*

Ghi vào nhật ký dưới một dạng mới, mẫu tin "complete"

Thuật toán mô tả giao thức ủy thác 2 pha

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

Cách xử lý giao tác khi có các sự cố

1. Sự cố về vị trí *(site failures)*

a. Một thành viên gặp sự cố trước khi ghi mẫu tin "ready" vào nhật ký

- Thời gian quá hạn của ***điều phối viên*** đã hết
- ***Điều phối viên*** quyết định hủy bỏ giao tác
- Tất cả các ***thành viên*** trực thuộc hủy bỏ những giao tác con của nó.
- ***thành viên bị sự cố*** được phục hồi, quy trình sẽ bắt đầu lại *(restart)* mà không cần phải tập hợp thông tin từ những vị trí khác.

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

1. Sự cố về vị trí (site failures)

b. Một thành viên gặp sự cố sau khi ghi mẫu tin "ready" vào nhật ký

- Những vị trí đúng sẽ kết thúc giao tác (commit hoặc abort).
- Vị trí có sự cố phục hồi bằng cách:
  - Truy cập thông tin phục hồi từ xa
  - *Điều phối viên* và tất cả những *thành viên* trực thuộc khác sẽ thực hiện như phần (1a), trong khi thành viên bị sự cố thì sẽ thực hiện bắt đầu lại như đã mô tả.

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

1. ***Sự cố về vị trí*** *(site failures)*

   c. Điều phối viên gặp sự cố sau khi ghi mẫu tin "prepare" nhưng trước khi ghi mẫu tin "global_commit" hoặc "global_abort" vào nhật ký.

- Các *thành viên* đã trả lời READY phải chờ sự phục hồi của *điều phối viên*.
- Quy trình bắt đầu lại của *điều phối viên* thừa nhận giao thức ủy thác từ lúc khởi đầu
- Giải thích tính đồng nhất của những *thành viên* từ việc ghi mẫu tin "prepare" vào nhật ký, và gửi lại thông điệp PREPARE cho họ.
- Mỗi *thành viên* thừa nhận rằng thông điệp PREPARE mới là một sự lặp lại của thông điệp trước.

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

1. ***Sự cố về vị trí*** *(site failures)*

   d. Điều phối viên gặp sự cố sau khi ghi mẫu tin "global_commit" hoặc "global_abort" nhưng trước khi ghi mẫu tin "complete" vào nhật ký.

- *Điều phối viên* phải gửi lại cho tất cả các *thành viên* những quyết định.
- Tất cả những *thành viên* không nhận được lệnh phải chờ cho đến khi *điều phối viên* phục hồi.
- Như phần trước, những *thành viên* không nên bị ảnh hưởng bởi việc nhận thông điệp thứ hai.

Cách xử lý giao tác khi có các sự cố

1.Sự cố về vị trí *(site failures)*

e. Điều phối viên gặp sự cố sau khi ghi mẫu tin "complete" vào nhật ký.

Trong trường hợp này, giao tác đã được kết thúc và không có hành động cần thiết để thực hiện lại.

5.2 SỰ HỖ TRỢ NGUYÊN TỬ CỦA CÁC GIAO TÁC PHÂN TÁN

2. Thông điệp bị mất

a. Một thông điệp trả lời (READY hoặc ABORT) từ một thành viên bị mất.

- Thời gian quá hạn của *điều phối viên* đã hết
- Giao tác bị hủy bỏ.

b. Một thông điệp PREPARE bị mất.

- Trong trường hợp này, *các thành viên* vẫn đợi.
- Giải quyết như trường hợp a.

2. Thông điệp bị mất

c. Một thông điệp lệnh (COMMIT hoặc ABORT) bị mất từ điều phối viên.

- *Thành viên* không nhận được lệnh *COMMIT hoặc ABORT từ điều phối viên và thành viên* vẫn không chắc chắn về các quyết định của mình.
- Giải quyết vấn đề này bằng cách mở đầu thời gian quá hạn trong log của *thành viên*;
- Nếu không có lệnh nào được nhận sau khoảng thời gian quá hạn thì khi trả lời, thì thành viên yêu cầu nhắc lại lệnh được gửi đến *điều phối viên*.

2. Thông điệp bị mất

d. Một thông điệp ACK bị mất.

- Điều phối viên vẫn không rõ *thành viên* đã nhận được thông điệp lệnh hay chưa.
- Giải quyết vấn đề này bằng cách xem thời gian quá hạn trong log của *điều phối viên*;
- Nếu không có thông điệp ACK được nhận sau khoảng thời gian quá hạn từ lúc phát lệnh thì điều phối viên sẽ gửi lại lệnh đó.
- Phương án tốt nhất để xử lý trường hợp này là gửi lại thông điệp ACK tại vị trí *thành viên* bị mất, mắc dù giao tác con đã được hoàn thành trong thời gian chờ và không hoạt động lâu hơn.

HẾT CHƯƠNG 5