TRƯỜNG ĐẠI HỌC PHENIKAA

KHOA CÔNG NGHỆ THÔNG TIN

--❧•❧--Logo, company name

Description automatically generated

|  |  |
| --- | --- |
| **Họ và tên sinh viên :** | **Phạm Tiến Thành Công** |
| **Mã sinh viên:** | **20010886** |
| **Lớp:** | **CNTT4-K14** |
| **Năm học :** | **2020-2021** |

**• Giao tác (transaction**) là một tập hợp có thứ tự các thao tác. Tập hợp này được xem như một đơn vị công việc.

– i.e. tất cả thao tác trong giao tác phải được thực hiện thành công, hoặc không thao tác nào được thực hiện.

– nếu có một thao tác không hoàn thành được thì toàn bộ giao tác cũng không hoàn thành.

**Một số thuật ngữ liên quan đến giao tác :**

– Begin [transaction/tran] : bắt đầu một transaction

– Commit [transaction/tran] : hoàn tất một transaction

– Rollback [transaction/tran] : quay lui, hủy bỏ toàn bộ phần giao tác đã thực hiện trước đó

**Tính chất của giao tác: ACID:**

• Tính nguyên tố (Atomic) : tất cả thao tác trong giao tác phải được thực hiện thành công, hoặc không thao tác nào được thực hiện.

• Tính nhất quán (Consistency) : Chuyển CSDL từ tình trạng nhất quán này sang tình trạng nhất quán khác.

**Tính chất của giao tác (tt) :**

• Tính độc lập (Isolation) : xử lý của một giao tác phải độc lập với những tác động (thấy or thay đổi dữ liệu) của các giao tác khác thực hiện đồng thời

• Tính bền vững (Durability) : Sau khi giao tác commit thành công, tất cả những thay đổi trên CSDL mà giao tác đã thực hiện phải được ghi nhận chắc chắn (vào ổ cứng).

– HQT CSDL luôn phải có cơ chế phục hồi dữ liệu để đảm bảo điều này, thường dùng cơ chế ghi nhận bằng transaction log

Để đảm bảo tính chất A của giao tác Người sử dụng phải tường minh điều khiển sự rollback của giao tác.

• Cần kiểm tra lỗi sau khi thực hiện mỗi thao tác trong giao tác để có thể xử lý rollback kịp thời.

**Lịch trình**là một danh sách các thao tác (đọc, viết, từ chối thực hiện, hoặc thành công) của các giao dịch, và thứ tự hai thao tác trong giao dịch T xuất hiện trong lịch trình phải giống với thứ tự nó xuất hiện trong T. Lịch trình biểu diễn sự tuần tự thực hiện của các thao tác. Ví dụ, lịch trình trong Hình 2 chỉ ra thứ tự thực hiện của các thao tác trong hai giao dịch T1 và T2. Chúng tôi biểu diễn mỗi thao tác trong một dòng. Chúng tôi nhấn mạnh rằng lịch trình biểu diễn các thao tác của các giao dịch nhìn từ phía DBMS. Thêm vào những thao tác này, một giao dịch có thể thực hiện những thao tác khác, như đọc hoặc ghi từ các file hệ điều hành, đánh giá các biểu thức số học, vv…; tuy nhiên, chúng tôi giả sử rằng các thao tác này được thực hiện mà không ảnh hưởng đến những giao dịch khác.

|  |  |
| --- | --- |
| T1 | T2 |
| R(A) |  |
| W(A) |  |
|  | R(B) |
|  | W(B) |
| R(C) |  |
| W(C) |  |

**Hình1 : Lịch trình của hai giao dịch**

Lịch trình trong Hình 2 không chứa các thao tác hủy bỏ hoặc thành công của mỗi giao dịch. Lịch trình chứa cả những thao tác này được gọi là lịch trình đầy đủ. Một **lịch trình đầy đủ** phải chứa tất cả các thao tác của tất cả các giao dịch xuất hiện trong nó. Nếu các thao tác của các giao dịch khác nhau không được chèn vào, tức là, các giao dịch này được thực hiện từ bắt đầu tới kết thúc, tuần tự từng giao dịch- thì gọi lịch trình này là **lịch trình tuần tự** (serial schedule).

**Lịch trình tuần tự** trên một tập S các giao dịch là một lịch trình mà kết quả thực hiện của nó tương đương với kết quả thực hiện các giao dịch này theo các lịch trình khác nhau.

Ví dụ, lịch trình trong Hình 2 là lịch trình tuần tự. Mặc dù các thao tác trong T1 và T2 xen kẽ nhau, nhưng kết quả của lịch trình này tương đương với việc thực hiện T1 (toàn bộ) và sau đó thực hiện T2. Như ta quan sát được, việc đọc và ghi đối tượng A của giao dịch T1 không ảnh hưởng đến việc đọc và ghi đối tượng A của giao dịch T2, và chúng ta có thể ‘đảo’ thứ tự T1, T2.

|  |  |
| --- | --- |
| T1 | T2 |
| R(A) |  |
| W(A) |  |
|  | R(A) |
|  | W(A) |
| R(B) |  |
| W(B) |  |
|  | R(B) |
|  | W(B) |
|  | Commit |
| Commit |  |

**Hình2: Lịch trình tuần tự**

Việc thực hiện các giao dịch này theo các thứ tự khác nhau có thể đưa đến những kết quả khác nhau nhưng giả sử tất cả chúng đều được chấp nhận. Để nhìn thấy điều này, lưu ý rằng hai giao dịch trong Hình có thể được thực hiện như Hình: Lịch trình tuần tự khác. Lịch trình này cũng là lịch trình tuần tự, tương đương với lịch trình T2; T1. Nếu T1 và T2 được gửi tới DBMS đồng thời, cả hai lịch trình này có thể được lựa chọn.

Định nghĩa về lịch trình tuần tự phía trên không bao gồm trường hợp các lịch trình chứa các giao dịch bị hủy bỏ. Chúng ta mở rộng định nghĩa này để nó bao hàm cả các lịch trình bị hủy bỏ trong 4.

|  |  |
| --- | --- |
| T1 | T2 |
|  | R(A) |
|  | W(A) |
| R(A) |  |
|  | R(B) |
|  | W(B) |
| R(A) |  |
| R(B) |  |
| W(B) |  |
|  | Commit |
| Commit |  |

**Hình3: Lịch trình tuần tự khác**

Cuối cùng, chúng ta ghi nhớ rằng một DBMS có thể thực hiện các giao dịch có lịch trình không tuần tự. Điều này có thể xảy ra vì hai lý do. Thứ nhất, DBMS này có lẽ sử dụng một cơ chế điều khiển tương tranh để đảm bảo các lịch trình không tuần tự được thực hiện như các lịch trình tuần tự (ví dụ xem trong . Thứ hai, SQL cung cấp cho người lập trình ứng dụng khả năng chỉ dẫn cho DBMS để nó lựa chọn các lịch trình không tuần tự.

**Lịch khả tuần tự:**

Có 2 loại lịch khả tuần tự:

**Conflict Serializable**: Dựa trên ý tưởng hoán vị các hành động không xung đột để chuyển một lịch đồng thời S về một lịch tuần tự S’. Nếu có một cách biến đổi như vậy thì S là một lịch conflict serializable.

**View Serializable** :Dựa trên ý tưởng lịch đồng thời S và lịch tuần tự S’ đọc và ghi những giá trị dữ liệu giống nhau. Nếu có một lịch S’ như vậy thì S là một lịch view serializab

|  |  |
| --- | --- |
| T1 | T2 |
| R(A,T) |  |
| T:=T+100 |  |
| W(A,T) |  |
|  | R(A,S) |
|  | S:S\*2 |
|  | W(A,S) |
| R(R,T) |  |
| T:=T+100 |  |
| W(B,T) | R(B,S) |
|  | S:S\*2 |
|  | W(B,S) |