TRƯỜNG ĐH CÔNG NGHỆ THÔNG TIN TP.HCM
KHOA HỆ THỐNG THÔNG TIN

Bài Thực hành số 4

Môn học: ***Hệ quản trị cơ sở dữ liệu***

I. Tóm tắt bài thực hành

1. Yêu cầu lý thuyết

Sinh viên đã được trang bị kiến thức:

- Các kiến thức cơ bản, nền tảng về CSDL, Cursor, Stored Procedure, Function, Trigger.
- Xây dựng các giao tác tường minh trên Oracle.

2. Nội dung

❖ **Tìm hiểu các phương thức khóa và các mức cô lập dữ liệu trong Oracle**

- Khái niệm
- Ý nghĩa.
- Cách sử dụng.

II. Các phương thức khóa

1. Tổng quan về giao tác trong Oracle

Một giao tác là một đơn vị thao tác luận lý bao gồm một hoặc nhiều câu lệnh SQL, được thực thi bởi một người dùng đơn. Trong Oracle, một giao tác bắt đầu bằng việc thực thi câu lệnh SQL đầu tiên của người dùng. Và kết thúc khi một trong những điều sau xảy ra:

- Người dùng sử dụng lệnh COMMIT hoặc ROLLBACK mà không có một điểm đánh dấu SAVEPOINT.
- Người dùng chạy một câu lệnh DDL chẳng hạn như CREATE, DROP, RENAME, hay ALTER.
- Người dùng ngắt kết nối với Oracle. Giao tác hiện tại sẽ được commit.
- Xử lý của người dùng bị ngắt một cách bất thường. Giao tác hiện tại sẽ bị rollback.

Sau khi một giao tác kết thúc, giao tác tiếp theo sẽ bắt đầu với câu lệnh SQL kế tiếp. Các câu lệnh kiểm soát giao tác:

- Ghi nhận vĩnh viễn những thay đổi được thực hiện trong giao tác (COMMIT).
- Quay ngược lại những thay đổi của giao tác, tình từ lúc giao tác bắt đầu hoặc từ một điểm savepoint (ROLLBACK).
- Đặt một điểm mà có thể rollback (SAVEPOINT)

- Thiết đặt thuộc tính cho giao tác (SET TRANSACTION)
  - `SET TRANSACTION ISOLATION LEVEL {READ COMMITTED|SERIALIZABLE};`
  - `SET TRANSACTION READ {ONLY|WRITE};`
- Điều chỉnh khi nào Oracle tiến hành commit (SET AUTOCOMMIT)
  - `SET AUTOCOMMIT ON` : commit ngay những thay đổi xuống cơ sở dữ liệu sau khi Oracle thực thi thành công các câu lệnh INSERT, UPDATE hoặc DELETE.
  - `SET AUTOCOMMIT OFF` (mặc định): ngăn không để Oracle commit tự động, người dùng phải commit thủ công.

2. Tại sao lại cần các phương thức khóa?

Giả sử có 2 giao tác đang truy xuất đồng thời trên 1 đơn vị dữ liệu. Có tất cả 4 trường hợp sau:

Trong connection C1, transaction thực hiện thao tác	Trong connection C2, transaction thực hiện thao tác	Nhận xét
Đọc	Đọc	Không có tranh chấp
Đọc	Ghi	Xảy ra tranh chấp
Ghi	Đọc	Xảy ra tranh chấp
Ghi	Ghi	HQT chỉ cho phép có đúng 1 giao tác được ghi trên đơn vị dữ liệu tại một thời điểm.

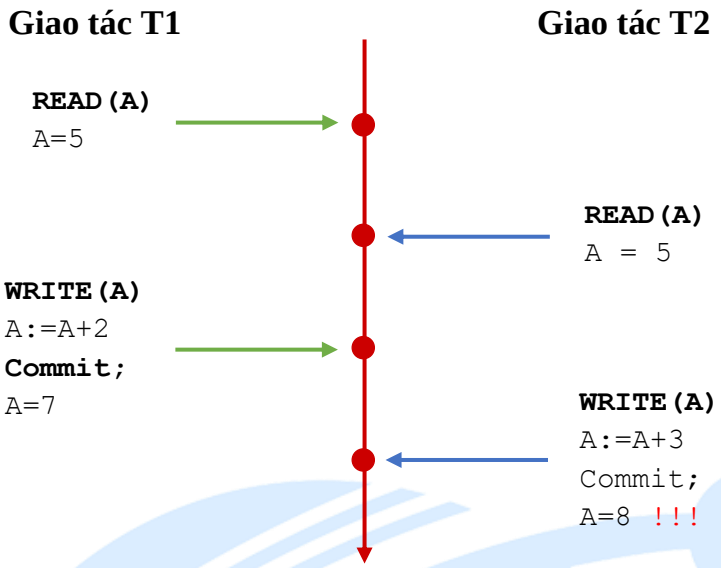
Như vậy khi có 2 giao tác (của 2 connection khác nhau) CÓ ÍT NHẤT MỘT THAO TÁC GHI trên cùng một đơn vị dữ liệu sẽ xảy ra tình trạng tranh chấp. Nếu để tình trạng tranh chấp này xảy ra sẽ dẫn đến những sai sót trên cơ sở dữ liệu.

Để giải quyết các vấn đề tranh chấp nêu trên, hệ quản trị cơ sở dữ liệu cần sử dụng các phương thức khóa, nhờ vậy mà khi có tranh chấp xảy ra hệ quản trị cơ sở dữ liệu có thể quyết định giao tác nào được thực hiện và giao tác nào phải chờ.

Trong môi trường truy xuất đồng thời, có thể xảy ra một số vấn đề như sau:

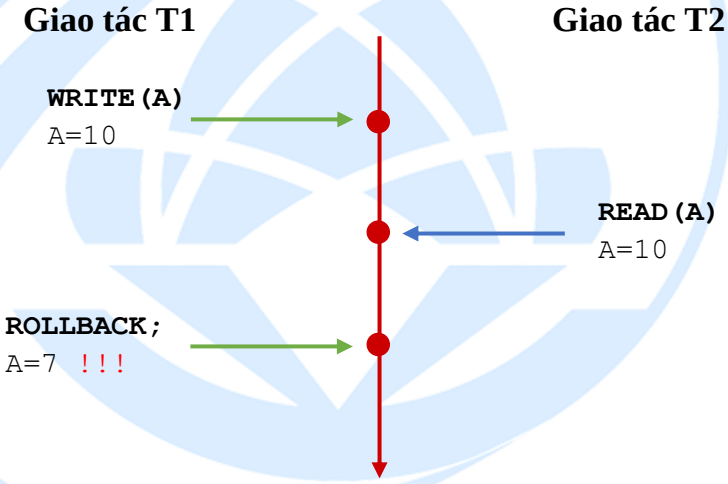
❖ **Mất dữ liệu cập nhật (Lost update)**

Tình trạng này xảy ra khi có nhiều hơn một giao tác cùng thực hiện cập nhật trên 1 đơn vị dữ liệu. Khi đó, tác dụng của giao tác cập nhật thực hiện sau sẽ đè lên tác dụng của thao tác cập nhật trước.



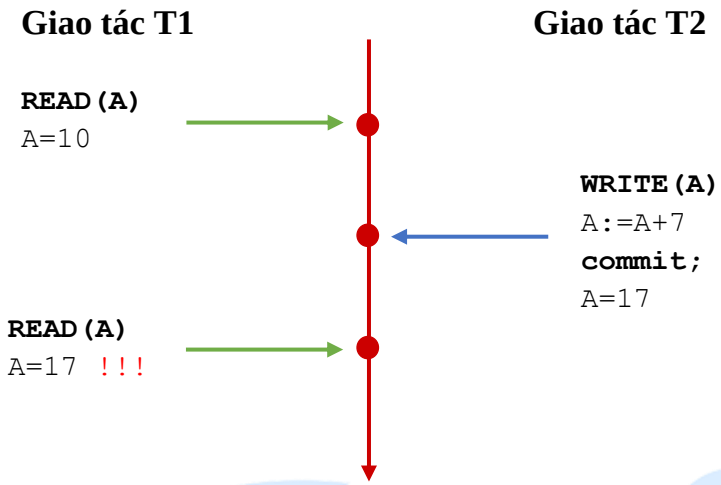
❖ **Đọc dữ liệu chưa commit (Uncommitted data, Dirty read)**

Xảy ra khi một giao tác thực hiện đọc trên một đơn vị dữ liệu mà đơn vị dữ liệu này đang bị cập nhật bởi một giao tác khác nhưng việc cập nhật chưa được xác nhận.



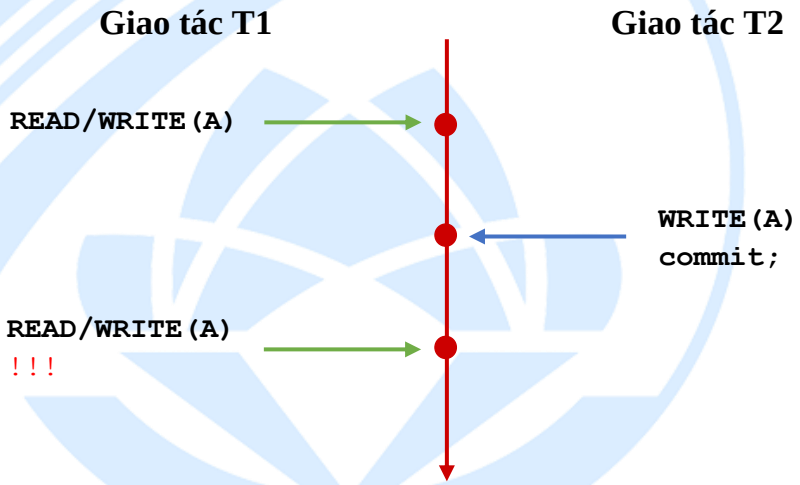
❖ **Giao tác đọc không thể lặp lại (Non-repeatable read)**

Tình trạng này xảy ra khi một giao tác T1 vừa thực hiện xong thao tác đọc trên một đơn vị dữ liệu (nhưng chưa commit) thì giao tác khác (T2) lại thay đổi (ghi) trên đơn vị dữ liệu này. Điều này làm cho lần đọc sau đó của T1 không còn nhìn thấy dữ liệu ban đầu nữa.



❖ **Bóng ma (Phantom read)**

Là tình trạng mà một giao tác đang thao tác trên một tập dữ liệu nhưng giao tác khác lại chèn thêm các dòng dữ liệu vào tập dữ liệu mà giao tác kia quan tâm.



Phantom read khác với Non-repeatable read ở chỗ nó không thay đổi dữ liệu cũ mà thêm dữ liệu mới vào.

3. Các phương thức khóa cơ bản

3.1. Shared Locks (S)

- Shared Lock ⇔ Read Lock
- Khi đọc 1 đơn vị dữ liệu, hệ quản trị tự động thiết lập Shared Lock trên đơn vị dữ liệu đó (trừ trường hợp sử dụng No Lock)
- Shared Lock có thể được thiết lập trên 1 bảng, 1 trang, 1 khóa hay trên 1 dòng dữ liệu.
- Nhiều giao tác có thể đồng thời giữ Shared Lock trên cùng 1 đơn vị dữ liệu.
- Không thể thiết lập Exclusive Lock trên đơn vị dữ liệu đang có Shared Lock.

- Shared Lock thường được giải phóng ngay sau khi sử dụng xong dữ liệu được đọc, trừ khi có thiết lập giữ shared lock cho đến hết giao tác.

3.2. Exclusive Locks (X)

- Exclusive Lock ⇔Write Lock
- Khi thực hiện thao tác ghi (insert, update, delete) trên 1 đơn vị dữ liệu, hệ quản trị tự động thiết lập Exclusive Lock trên đơn vị dữ liệu đó.
- Exclusive Lock luôn được giữ đến hết giao tác.
- Tại 1 thời điểm, chỉ có tối đa 1 giao tác được quyền giữ Exclusive Lock trên 1 đơn vị dữ liệu.
- Không thể thiết lập Exclusive Lock trên đơn vị dữ liệu đang có Shared Lock.

III. Mức cô lập

1. Các mức cô lập

1.1. Read Uncommitted

❖ **Đặc điểm:**

- Không thiết lập Shared Lock trên những đơn vị dữ liệu cần đọc. Do đó không phải chờ khi đọc dữ liệu (kể cả khi dữ liệu đang bị lock bởi giao tác khác)
- (Vẫn tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác)

❖ **Ưu điểm:**

- Tốc độ xử lý rất nhanh
- Không cản trở những giao tác khác thực hiện việc cập nhật dữ liệu

❖ **Khuyết điểm:**

Có khả năng xảy ra mọi vấn đề khi xử lý đồng thời :

- Dirty Read
- Non-repeatable Read
- Phantom Read
- Lost Update

1.2. Read Committed

❖ **Đặc điểm:**

- Đây là mức độ cô lập mặc định của Oracle/SQL Server
- Tạo Shared Lock trên đơn vị dữ liệu được đọc, Shared Lock được giải phóng ngay sau khi đọc xong dữ liệu

- Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác.

❖ **Ưu điểm:**

- Giải quyết vấn đề Dirty Reads
- Shared Lock được giải phóng ngay, không cần phải giữ cho đến hết giao tác nên không cản trở nhiều đến thao tác cập nhật của các giao tác khác.

❖ **Khuyết điểm:**

- Chưa giải quyết được vấn đề Non-repeatable Read, Phantom Read, Lost Update
- Phải chờ nếu đơn vị dữ liệu cần đọc đang được giữ khoá ghi (xlock)

1.3. Repeatable Read

❖ **Đặc điểm:**

- Tạo Shared Lock trên đơn vị dữ liệu được đọc và giữ shared lock này đến hết giao tác ➔ Các giao tác khác phải chờ đến khi giao tác này kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này.
- (Repeatable Read = Read Committed + Giải quyết Non-repeatable Read)
- Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác.

❖ **Ưu điểm:**

- Giải quyết vấn đề Dirty Read và Non-repeatable Read

❖ **Khuyết điểm:**

- Chưa giải quyết được vấn đề Phantom Read, do vẫn cho phép insert những dòng dữ liệu thỏa điều kiện thiết lập shared lock
- Phải chờ nếu đơn vị dữ liệu cần đọc đang được giữ khoá ghi (xlock)
- Shared lock được giữ đến hết giao tác ➔ cản trở việc cập nhật dữ liệu của các giao tác khác

1.4. Serializable

❖ **Đặc điểm:**

- Tạo Shared Lock trên đơn vị dữ liệu được đọc và giữ shared lock này đến hết giao tác ➔ Các giao tác khác phải chờ đến khi giao tác này kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này.
- Không cho phép Insert những dòng dữ liệu thỏa mãn điều kiện thiết lập Shared Lock (sử dụng Key Range Lock) ➔ Serializable = Repeatable Read + Giải quyết Phantom Read
- Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác.

❖ **Ưu điểm:**
- Giải quyết thêm được vấn đề Phantom Read

❖ **Khuyết điểm:**
- Phải chờ nếu đơn vị dữ liệu cần đọc đang được giữ khoá ghi (xlock)
- Cản trở nhiều đến việc cập nhật dữ liệu của các giao tác khác

2. Sử dụng các mức cô lập

Trong Oracle, để ngăn ngừa các vấn đề về truy xuất đồng thời, ta xem xét sử dụng các mức cô lập như sau:

	Dirty Read	Lost Update	Non-repeatable Read	Phantom Read
Read Uncommitted	Cho phép xảy ra	Cho phép xảy ra	Cho phép xảy ra	Cho phép xảy ra
Read Committed (Oracle)	Không được	Cho phép xảy ra	Cho phép xảy ra	Cho phép xảy ra
Repeatable Read	Không được	Không được	Không được	Cho phép xảy ra
Serializable (Oracle)	Không được	Không được	Không được	Không được

IV. Bài tập

1. Bài tập 1

1.1. Chuẩn bị

Cần mở hai cửa sổ SQL * Plus trên màn hình. Chúng ta sẽ giả sử rằng có hai người dùng, mỗi cửa sổ cho một người dùng. Đặt hai cửa sổ sao cho một cửa sổ ở phía trên màn hình và cửa sổ kia ở dưới màn hình. Đặt tên cửa sổ SQL * Plus trên là TOP và cửa sổ dưới là BOT. SET AUTOCOMMIT OFF ở cả hai cửa sổ.

1.2. Tạo bảng và dữ liệu

Tạo các bảng theo dõi số lượng hàng trong kho Các bộ phận của búp bê (DollParts) và Búp bê đã lắp ráp (Dolls). Thực hiện các câu lệnh SQL sau trong TOP hoặc BOT:

```
drop table DollParts;
create table DollParts (
    name  varchar2(10) primary key,
    cnt   number check (cnt >= 0));
```

```
insert into DollParts(name, cnt) values('HEAD', 17);
insert into DollParts(name, cnt) values('BODY', 17);
insert into DollParts(name, cnt) values('ARM', 17);
insert into DollParts(name, cnt) values('LEG', 17);
commit;

create table Dolls (
    name  varchar2(20) primary key,
    cnt   number);

insert into Dolls(name, cnt) values('Barbie', 0);
insert into Dolls(name, cnt) values('Ken', 0);
commit;
```

1.3. Sử dụng mức cô lập mặc định và thực hiện giao tác

Giao tác thực hiện là khấu trừ trong kho (DollParts) những thành phần cần thiết để tạo ra một con búp bê và tăng số lượng của búp bê đã lắp ráp lên 1. Vấn đề mà chúng ta có thể gặp phải là: nếu hai người dùng làm điều này cùng một lúc?

Bây giờ hãy chạy các câu lệnh SQL sau trong các cửa sổ TOP và BOT tương ứng:

```
TOP:  select * from DollParts;
BOT:  select * from DollParts;
TOP:  update DollParts set cnt=cnt-1 where name='HEAD';
TOP:  update DollParts set cnt=cnt-1 where name='BODY';
BOT:  select * from DollParts;
TOP:  select * from DollParts;
TOP:  commit;
BOT:  select * from DollParts;
```

Hãy trả lời các câu hỏi sau:

- Số lượng ‘HEAD’ và ‘BODY’ có được từ câu lệnh truy vấn BOT thứ hai là bao nhiêu?
- Số lượng ‘HEAD’ và ‘BODY’ có được từ câu lệnh truy vấn TOP thứ hai là bao nhiêu?

- Số lượng 'HEAD' và 'BODY' có được từ câu lệnh truy vấn BOT thứ ba là bao nhiêu?

1.4. Rollback giao tác

Hãy chạy các câu lệnh SQL sau trong cửa sổ TOP và BOT:

```
select * from Dolls;
update Dolls set cnt=cnt+1 where name='Barbie';
update Dolls set cnt=cnt+1 where name='Ken';
select * from Dolls;
rollback;
select * from Dolls;
```

1.5. Các vấn đề truy xuất đồng thời 1

Hãy chạy các câu lệnh SQL sau trong cửa sổ TOP hoặc BOT:

```
create table Account (
    AcctName   varchar2(30) primary key,
    AcctBal    number);

insert into Account values('John', 100);
insert into Account values('David', 200);
insert into Account values('Mary', 300);
insert into Account values('Cathy', 100);
commit;
```

Cài đặt lại mức cô lập ở cả TOP và BOT:

```
set transaction isolation level read uncommitted;
```

Chạy các câu lệnh sau theo đúng thứ tự, ghi nhận kết quả và giải thích:

TOP	BOT
-- steps 1 & 2 var x number; begin select AcctBal into :x from Account where AcctName='John'; end; print x;	
	-- steps 1' & 2' var x number; begin select AcctBal into :x from Account where AcctName='John'; end; print x;
-- step 3	

begin :x := :x + 10; end; print x;	
	-- step 3' begin :x := :x + 20; end; print x;
-- step 4 begin update Account set AcctBal=:x where AcctName='John'; end;	
	-- step 4' begin update Account set AcctBal=:x where AcctName='John'; end;
-- step 5 commit; or rollback;	
	-- step 5' commit;

1.6. Các vấn đề truy xuất đồng thời 2

Lần lượt thay đổi mức cô lập ở cả TOP và BOT thành READ COMMITTED, REPEATABLE READ, SERIALIZABLE. Thực hiện lại 1.5, ghi nhận và giải thích kết quả đạt được.

1.7. Các vấn đề truy xuất đồng thời 3

Chạy câu lệnh sau để khôi phục lại dữ liệu:

```
update Account set AcctBal=100 where AcctName='John';
commit;
```

Chạy lại 1.5 theo thứ tự sau: 1, 2, 3, 4, 1', 2', 5r, 3', 4', 5'c

Ghi nhận kết quả vào bảng bên dưới:

	Kỳ vọng giá trị của John account	Giao tác có chạy hoàn tất?	Nếu không, hãy giải thích tại sao	Giá trị account thực tế, nếu giao tác hoàn tất
RC				
RR				
SR				

1.8. Các vấn đề truy xuất đồng thời 4

Khôi phục dữ liệu và chạy lại 1.5 theo thứ tự sau: 1, 2, 1', 2', 3, 4, 5c, 3', 4', 5'c

Ghi nhận kết quả vào bảng như phần 1.7.

1.9. Các vấn đề truy xuất đồng thời 5

Sửa lại giao tác BOT như sau:

```
BOT:
-- step 1'
set serveroutput on;
var x number;

-- step 2'
begin select AcctBal into :x from Account where
AcctName='John'; end;
print x;

-- step 3'
declare bal number;
begin
  if :x=100  then
    select AcctBal into bal from Account where
AcctName='John';
    dbms_output.put_line(bal);
  end if;
end;

-- step 4
commit;
```

Khôi phục dữ liệu và chạy lại theo thứ tự sau: 1', 2', 1, 2, 3, 4, 5c, 3', 4'c

Ghi nhận kết quả vào bảng như phần 1.7.

1.10. Các vấn đề truy xuất đồng thời 6

Sửa lại giao tác BOT như sau:

```
BOT:

-- step 1'
```

```
set serveroutput on
var x number;


-- step 2'
begin select AVG(AcctBal) into :x from Account where
AcctBal>100; end;
print x;


-- step 3'
declare
  avgBal number;
begin
  if :x=250 then
    select AVG(AcctBal) into avgBal from Account where
AcctBal>100;
    dbms_output.put_line(avgBal);
  end if;
end;
print x;


-- step 4'
commit;
```

Khôi phục dữ liệu và chạy lại theo thứ tự sau: 1', 2', 1, 2, 3, 4, 5c, 3', 4'c
Ghi nhận kết quả vào bảng như phần 1.7.

2. Bài tập 2

Yêu cầu sinh viên phát hiện tất cả các trường hợp xử lý đồng thời trong cơ sở dữ liệu Quản lý thư viện và đề nghị cách giải quyết.

3. Bài tập 3

Yêu cầu sinh viên phát hiện tất cả các trường hợp xử lý đồng thời trong đồ án môn học và đề nghị cách giải quyết.

~ HẾT ~