BUỔI 8: TRANSACTION AND ISOLATION

I. CHỦ ĐỀ

- Tìm hiểu Transaction
- Tìm hiểu Isolation level
- Thực thi Transaction

II. MỤC ĐÍCH

- Biết cách áp dụng mức cô lập trong xử lý đồng thời.

III. CÔNG CỤ

- Microsoft SQL Server 2014 Express Edition/Management hoặc hơn.

IV. MÔI TRƯỜNG

- Window

V. CÁCH THỰC HIỆN

5.1 GIAO TÁC LÀ GÌ?

Giao tác là một đơn vị xử lý nguyên tố gồm một chuỗi các hành động tương tác lên CSDL. Một đơn vị xử lý nguyên tố phải thể hiện bốn tính chất: tính nguyên tố, tính nhất quán, tính cô lập, và tính bền vững.

Atomicity (Nguyên tố):

Một giao tác phải là một đơn vị nguyên tố (không thể phân chia được nữa) của công việc: hoặc là tất cả các thay đổi dữ liệu của nó được thực hiện, hoặc không ai trong số họ được thực hiện.

Ghi chú: Để đảm bảo tính Automicity 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.

Consistency (Nhất quán):

Khi hoàn tất, một giao tác cần phải trả về tập dữ liệu trong một khối thống nhất. Trong một CSDL quan hệ, tất cả các luật (rules) phải được áp dụng để sự thay đổi của giao tác vẫn duy trì tính toàn vẹn dữ liệu. Tất cả các cấu trúc dữ liệu nội bộ, chẳng hạn như chỉ mục B-cây hoặc danh sách liên kết đôi, phải được chính xác tại thời điểm kết thúc giao tác.

Isolation (Cô lập):

Cho dù nhiều giao tác có thể thực hiện đồng thời, hệ thống phải đảm bảo rằng đối với mỗi cặp giao tác Ti, Tj, hoặc Tj kết thúc thực hiện trước khi Ti khởi động hoặc Tj bắt đầu thực

hiện sau khi Ti kết thúc. Như vậy mỗi giao tác không cần biết đến các giao tác khác đang thực hiện đồng thời trong hệ thống.

Durability (Bền vững):

Sau khi giao tác hoàn thành, các thay đổi đã được tạo ra cho CSDL vẫn tồn tại ngay cả trong trường hợp hệ thống bị lỗi.

Các tính chất này thường được gọi là các tính chất ACID (Các chữ cái đầu của 4 tính chất).

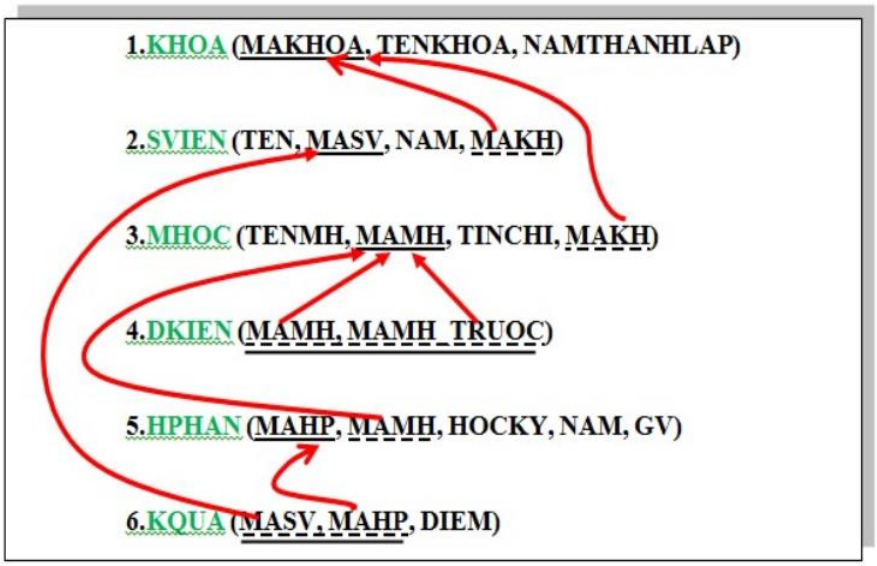
5.2 XỬ LÝ TRANSACTION:

Các lệnh sau đây được sử dụng để xử lý transaction:

COMMIT - để lưu các thay đổi.
ROLLBACK - để khôi phục lại các thay đổi.
SAVEPOINT - tạo ra các điểm trong transaction để ROLLBACK.
SAVEPOINT RELEASE – loại bỏ SAVEPOINT đã được tạo
SET TRANSACTION - thiết lập các thuộc tính cho transaction.

Sinh viên tạo Database **QUANLYDAOTAO_B8_MSSV** từ file đính kèm để thực hiện các ví dụ sau:

Lược đồ Cơ sở dữ liệu:



Thể hiên Cơ sở dữ liêu:

KHOA

MAKHOA	TENKHOA	NAMTHANHLAP
CNTT	Công Nghệ Thông tin	1995
VL	Vật lý	1976
TOAN	Toán	1976

SVIEN

TEN	MASV	NAM	MAKH
Sơn	17	1	CNTT
Bảo	8	2	CNTT
Trang	5	3	TOAN

MHOC

TENMH	MAMH	TINCHI	MAKH
Nhập môn Tin học	COSC1310	4	CNTT
Cấu trúc dữ liệu	COSC3320	4	CNTT
Toán rời rạc	MATH2410	3	TOAN
Cơ sở dữ liệu	COSC3380	3	CNTT
Vật lý đại cương	PHYS3332	3	VL

DKIEN

MAMH	MAMH_TRUOC
COSC3380	COSC3320
COSC3380	MATH2410
COSC3320	COSC1310

HPHAN

MAHP	MAMH	HOCKY	NAM	GV
85	MATH2410	1	1996	Kim
92	COSC1310	1	1996	An
102	COSC3320	2	1997	Nhiên
112	MATH2410	1	1997	Vân
119	COSC1310	1	1997	An
135	COSC3380	1	1997	Sơn

KQUA

MASV	MAHP	DIEM
17	102	8
17	119	6
5	85	10
8	92	10
8	102	8
8	135	10

a) COMMIT:

Lệnh COMMIT được sử dụng để lưu các thay đổi gọi bởi một transaction với cơ sở dữ liệu.

Cú pháp:

```
Statement 1
Statement 2
…
COMMIT
```

Ví dụ: Hãy thực hiện từng câu lệnh sau để kiểm tra kết quả

```
---Câu 1
SELECT * FROM KQUA
--- Câu 2
BEGIN TRAN
    DELETE KQUA WHERE MASV = 5
COMMIT
---Câu 3: Kiểm tra kết quả
SELECT * FROM KQUA
```

b) ROLLBACK:

Lệnh ROLLBACK được sử dụng để hoàn tác các transaction chưa được lưu vào cơ sở dữ liệu.

Cú pháp:

```
Statement 1
Statement 2
…
ROLLBACK
```

Ví dụ: Hãy thực hiện từng câu lệnh sau để kiểm tra kết quả

```
---Câu 1
SELECT * FROM KQUA
--- Câu 2
BEGIN TRAN
    DELETE KQUA WHERE MASV = 17
ROLLBACK
---Câu 3: Kiểm tra kết quả trước và sau
SELECT * FROM KQUA
```

c) SAVE POINT:

SAVEPOINT là một điểm trong một transaction khi bạn có thể cuộn transaction trở lại một điểm nhất định mà không quay trở lại toàn bộ transaction.

Cú pháp:

```
SAVE TRAN SAVEPOINT_NAME
Statement 1
Statement 2
…
ROLLBACK TRAN SAVEPOINT_NAME
```

Ví dụ 1: Hãy thực hiện từng câu lệnh sau để kiểm tra kết quả

```
---Câu 1
SELECT * FROM SVIEN
--- Câu 2
BEGIN TRAN
        SAVE TRAN SP1
                INSERT INTO SVIEN VALUES (N'Châu', 4, 1, 'CNTT')
                INSERT INTO SVIEN VALUES (N'Mi', 6, 1, 'CNTT')
        ROLLBACK TRAN SP1
COMMIT
---Câu 3: Kiểm tra kết quả trước và sau
SELECT * FROM SVIEN
```

Ví dụ 2: Hãy thực hiện từng câu lệnh sau để kiểm tra kết quả

```
---Câu 1
SELECT * FROM KQUA
--- Câu 2
BEGIN TRAN
        SAVE TRAN SP1
        DELETE KQUA WHERE MASV = 17
        SAVE TRAN SP2
        DELETE KQUA WHERE MASV = 8
        ROLLBACK TRAN SP2
COMMIT
---Câu 3: Kiểm tra kết quả trước và sau
SELECT * FROM KQUA
```

d) SET TRANSACTION:

SET TRANSACTION có thể được sử dụng để khởi tạo một Database Transaction. Lệnh này được sử dụng để chỉ định các đặc tính cho Transaction đó.

Cú pháp:

```
BEGIN TRAN
SET TRAN TRANSACTION_MODE
Statement 1
Statement 2
…
```

TRANSACTION_MODE: ISOLATION LEVEL.

Ví dụ: Hãy thực hiện từng câu lệnh sau để kiểm tra kết quả

```
---Câu 1
SELECT * FROM HPHAN
--- Câu 2: Kiểm tra kết quả thực hiện câu sau
BEGIN TRAN
SET TRAN ISOLATION LEVEL READ COMMITTED
        SELECT MAHP, MAMH, HOCKY, GV
        FROM HPHAN
        WHERE MAMH = 'MATH2410'
 COMMIT
```

Lưu ý: ví dụ với ISOLATION sẽ làm rõ ở mục 5.3.

5.3 CÁC VẤN DỀ LIÊN QUAN ĐẾN XỬ LÝ TRUY XUẤT ĐỒNG THỜI

Lost Update (mất dữ liệu cập nhật):

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, Dirtyread):

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 đã hoàn tất.

Thao tác đọc không thể lặp lại (Unrepeatable data):

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):

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

Cách giải quyết các vấn đề:

Dùng khái niệm giao tác và dùng cơ chế khóa:

- Write lock (exclusive lock)
- Read lock (shared lock)
- Các hệ quản trị cụ thể còn có những loại khoá mở rộng khác: updlock, holdlock…
- Đặt khóa vào đơn vị dữ liệu:
  - Tự động (đặt mức cô lập cho giao tác)

- Thủ công (đặt cấp độ khóa trong câu SELECT).

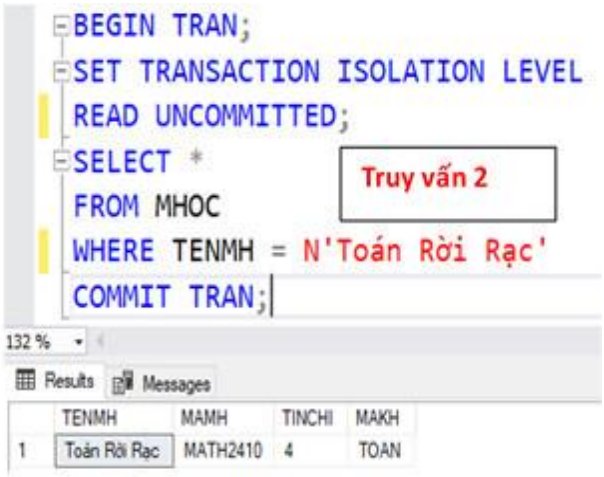
5.4 BÀI TẬP VÍ DỤ

Sinh viên sử dụng Database **QUANLYDAOTAO_B8_MSSV** đã tạo ở trên và thực hiện các yêu cầu sau:

Yêu cầu:

1. **Read Uncommitted + vấn đề Dirty Read**

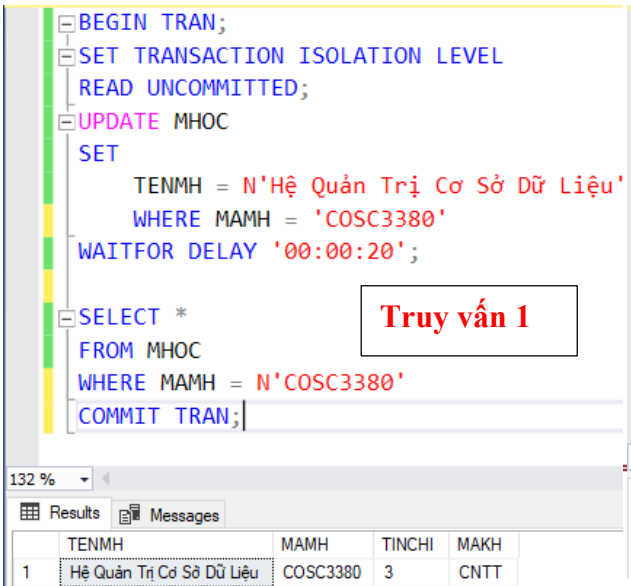
Bây giờ chúng ta sẽ thực hiện các truy vấn sau, Truy vấn-1 cập nhật số tín chỉ của môn học "Toán Rời Rạc" thành 4 tín chỉ, sau đó nó sẽ đợi 20 giây và khôi phục sửa đổi dữ liệu. Tại thời điểm này, chúng tôi sẽ ngay lập tức thực thi Truy vấn-2 và truy vấn này đọc dữ liệu đã sửa đổi nhưng việc cập nhật chưa được xác nhận đã hoàn tất.

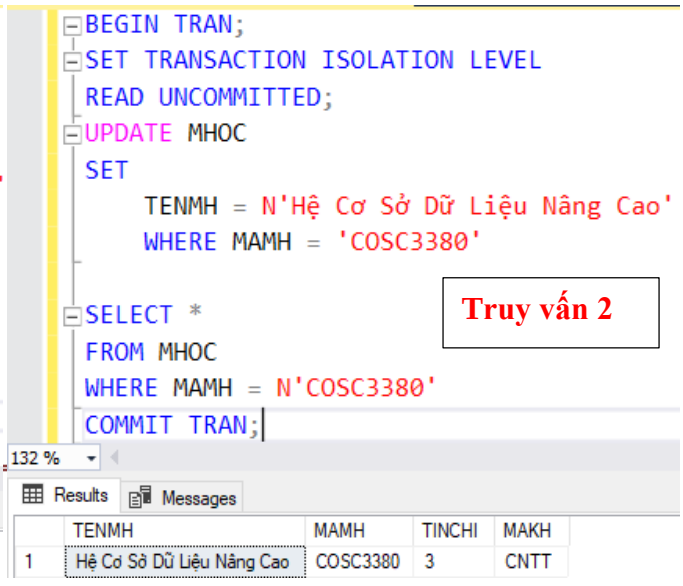


Kết quả là, dữ liệu được đọc bởi Truy vấn-2 là ***dữ liệu rác***. Ở Truy vấn-1 dữ liệu được trả về trạng thái đầu tiên do quá trình khôi phục.

2. **Read Uncommitted + vấn đề Lost Updated**

Trường hợp 2, chúng ta sẽ thực hiện các truy vấn sau, trên giao tác 1, tạo Truy vấn-1 cập nhật tên môn học thành "Hệ Quản Trị Cơ Sở Dữ Liệu", sau đó nó sẽ đợi 20 giây, tiếp tục trên giao tác 2, ta tạo Truy vấn-2 cập nhật tên môn học thành "Hệ Cơ Sở Dữ Liệu Nâng Cao".

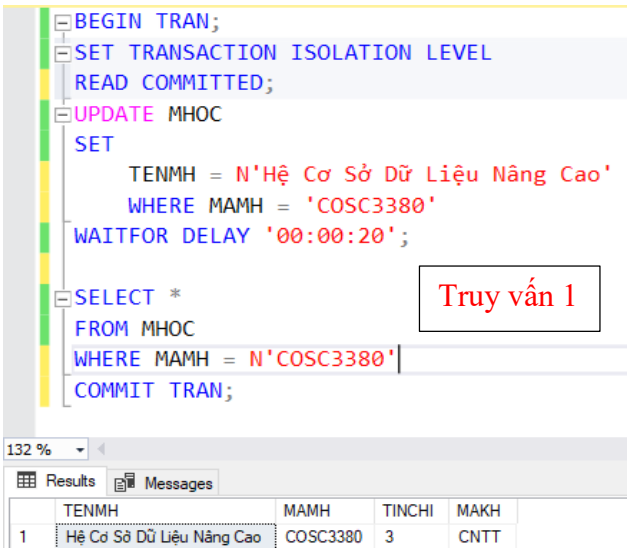


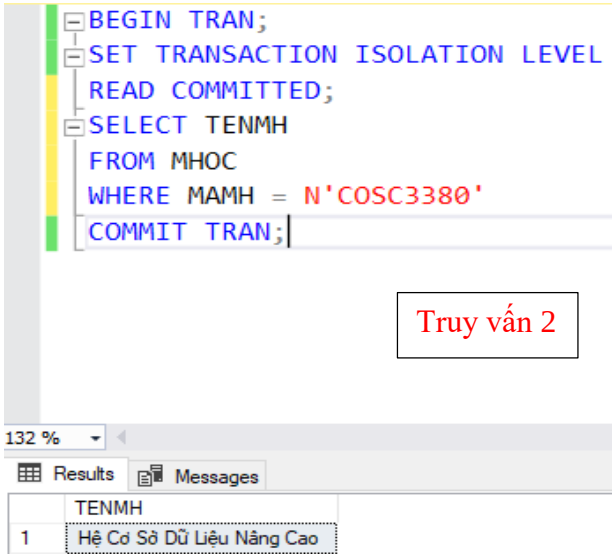


Kết quả là, dữ liệu được cập nhập ở bởi Truy vấn-2 đè lên tác dụng của thao tác cập nhật trước của Truy vấn-1.

3. Read Commited + vấn đề Dirty Read

Trường hợp 3, chúng ta sẽ thực hiện các truy vấn sau, tên môn học thành "Hệ Quản Trị Cơ Sở Dữ Liệu Nâng Cao", sau đó nó sẽ đợi 20 giây. Tại thời điểm này, chúng tôi sẽ ngay lập tức thực thi Truy vấn-2 và truy vấn này đọc dữ liệu đã sửa đổi nhưng việc cập nhật chưa được xác nhận đã hoàn tất.





Tại level này thì Transaction sẽ không thể đọc dữ liệu từ một Transaction đang trong quá trình cập nhật hay sửa đổi mà phải đợi transaction đó hoàn tất. Như vậy thì chúng ta có thể tránh được Dirty Read

Kết quả là, dữ liệu được đọc bởi Truy vấn-2 là dữ liệu đã được cập nhập ở Truy vấn-1.

VI. BÀI TẬP TẠI LỚP:

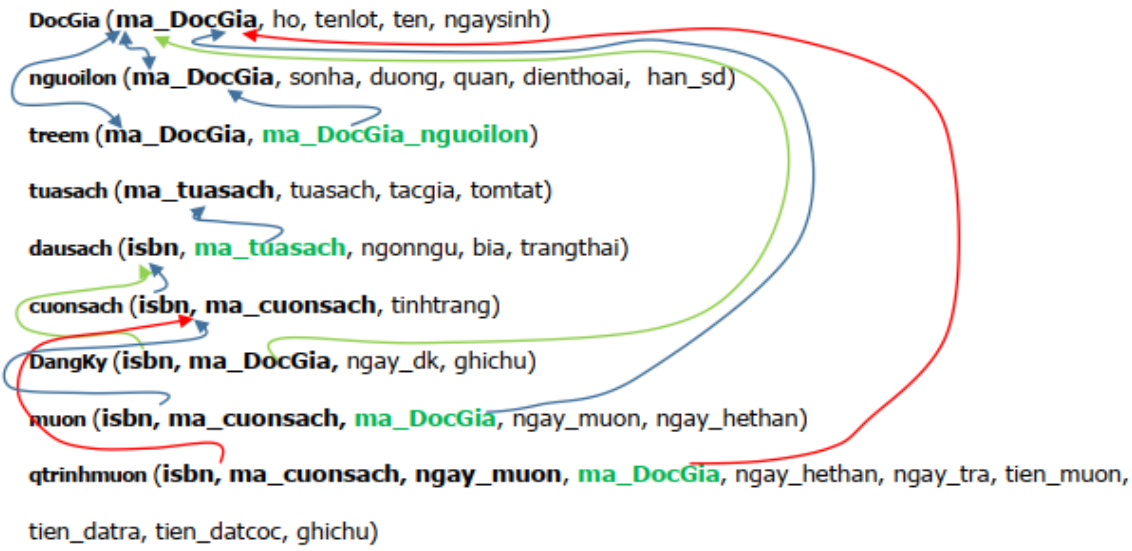
- Sinh viên tạo Database **QUANLYTHUVIEN_B8_MSSV** với hai file đính kèm (CREATDATABASE.sql, USEDATABASE.sql) và hoàn thành các câu hỏi trong bài tập sau bằng cách bổ sung kết quả phân tích và hình vào file Word (đính kèm), sau đó đổi tên như sau và nộp bài lên học trực tuyến.
    - **Assignment - Session 8 - Submission**

+ Tên file: StudentID-FullName-Assignment-Session8.doc

Ví dụ: 217000000044-NguyenVanA-Assignment-Session8.doc

+ Hạn nộp: theo lịch học của lớp

+ Lược đồ Cơ sở dữ liệu:



+ Yêu cầu:

Đề bài Lớp 221_71ITIS30203_01 (01, 02, 03):

1. Sử dụng **Read Uncommitted + vấn đề Dirty Read,** các bạn sẽ thực hiện các truy vấn sau:
   - ✓ query-1: cập nhật lại ngôn ngữ thành "Tiếng Nga" cho isbn là 1 trong bảng DAUSACH, sau đó nó sẽ đợi 10 giây và ***khôi phục*** sửa đổi dữ liệu.
   - ✓ query-2: chế độ đọc ***Read Uncommited***, truy vấn này đọc (xem) dữ liệu đã sửa đổi bảng DAUSACH và ***hoàn tất***.
   - ✓ Sinh viên thực hiện query-1 trước, query-2 sau, sau đó phân tích kết quả của query-1, query-2 và chụp hình kết quả của 2 query vào file Word.

2. Viết một thủ tục (store procedure) mang tên *SP_ThemDauSach*, thủ tục này dùng để insert dữ liệu cho bảng DAUSACH có chèn lệnh waitfor delay "00:00:10", với các mức cô lập: **READ UNCOMMITTED**, sau đó giả lập 2 giao dịch cùng thực hiện stored procedure này, lần lượt với các mức cô lập: **READ UNCOMMITTED.**

   Cú pháp mẫu:

```
CREATE PROCEDURE SP_TEST
@i_TEST1 VARCHAR(50)
AS
BEGIN TRAN
        SET NOCOUNT ON
        SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

        WAITFOR DELAY '00:00:10'
        INSERT INTO TEST00(TEST1) VALUES(@i_TEST1)

        SET NOCOUNT OFF
COMMIT TRAN
-----------------------------------------------------------------------
EXEC SP_TEST 1
SELECT * FROM TEST00   --Trường hợp 3
```

Sinh viên phân tích kết quả và chụp hình kết quả của hai giao dịch vào file Word theo các trường hợp sau:

- ✓ Trường hợp 1: Tương tự cấu trúc trên
- ✓ Trường hợp 2: Đưa câu lệnh WAITFOR DELAY xuống dưới câu INSERT (ở cả 2 giao dịch)
- ✓ Trường hợp 3: Đưa câu SELECT từ bên ngoài vào dưới câu WAITFOR DELAY (của trường hợp 2).

Đề bài Lớp 221_71ITIS30203_02 (01, 02, 03):

1. Sử dụng **Read Commited + vấn đề Dirty Read,** các bạn sẽ thực hiện các truy vấn sau:
   - ✓ query-1: chế độ đọc ***Read Commited,*** cập nhật lại ngôn ngữ thành "Tiếng Lào" cho isbn là 1 trong bảng DAUSACH, sau đó nó sẽ đợi 10 giây, đọc (xem) dữ liệu đã sửa đổi bảng DAUSACH và ***hoàn tất.***
   - ✓ query-2: chế độ đọc ***Read Commited***, truy vấn này đọc (xem) dữ liệu đã sửa đổi bảng DAUSACH và ***hoàn tất***.
   - ✓ Sinh viên thực hiện query-1 trước, query-2 sau, sau đó phân tích kết quả của query-1, query-2 và chụp hình kết quả của 2 query vào file Word (*gợi ý giống ví dụ 3*).

2. Viết một thủ tục (store procedure) mang tên *SP_ThemDocGia*, thủ tục này dùng để insert dữ liệu cho bảng DOCGIA có chèn lệnh waitfor delay "00:00:10", với các mức cô lập: **READ COMMITTED**, sau đó giả lập 2 giao dịch cùng thực hiện stored procedure này, lần lượt với các mức cô lập: **READ COMMITTED.**

   Cú pháp mẫu:

```
CREATE PROCEDURE SP_TEST
@i_TEST1 VARCHAR(50)
AS
BEGIN TRAN
        SET NOCOUNT ON
        SET TRANSACTION ISOLATION LEVEL READ COMMITTED

        WAITFOR DELAY '00:00:10'
        INSERT INTO TEST00(TEST1) VALUES(@i_TEST1)

        SET NOCOUNT OFF
COMMIT TRAN
-----------------------------------------------------------------------
EXEC SP_TEST 1
SELECT * FROM TEST00   --Trường hợp 3
```

   Sinh viên phân tích kết quả và chụp hình kết quả của hai giao dịch vào file Word theo các trường hợp sau:

   - ✓ Trường hợp 1: Tương tự cấu trúc trên
   - ✓ Trường hợp 2: Đưa câu lệnh WAITFOR DELAY xuống dưới câu INSERT (ở cả 2 giao dịch)
   - ✓ Trường hợp 3: Đưa câu SELECT từ bên ngoài vào dưới câu WAITFOR DELAY (của trường hợp 2).

Đề bài Lớp 221_71ITIS30203_03 (01, 02):

1. Sử dụng **Read Uncommitted + vấn đề Lost Updated,** các bạn sẽ thực hiện các truy vấn sau:
   - ✓ query-1: chế độ đọc ***Read Uncommited***, cập nhật lại trạng thái thành "N" của isbn là 1 trong bảng DAUSACH, sau đó nó sẽ đợi 10 giây, xem dữ liệu của bảng DAUSACH và ***hoàn tất***.
   - ✓ query-2: chế độ đọc ***Read Uncommited***, cập nhật lại trạng thái thành "Y" của isbn là 1 trong bảng DAUSACH, xem dữ liệu của bảng DAUSACH và ***hoàn tất***.
   - ✓ Sinh viên thực hiện query-1 trước, query-2 sau, sau đó phân tích kết quả của query-1, query-2 và chụp hình kết quả của 2 query vào file Word.

2. Viết một thủ tục (store procedure) mang tên *SP_ThemTuaSach*, thủ tục này dùng để insert dữ liệu cho bảng TUASACH có chèn lệnh waitfor delay "00:00:10", với các mức cô lập: **READ UNCOMMITTED**, sau đó giả lập 2 giao dịch cùng thực hiện stored procedure này, lần lượt với các mức cô lập: **READ UNCOMMITTED.**

   Cú pháp mẫu:

```
CREATE PROCEDURE SP_TEST
@i_TEST1 VARCHAR(50)
AS
BEGIN TRAN
        SET NOCOUNT ON
        SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

        WAITFOR DELAY '00:00:10'
        INSERT INTO TEST00(TEST1) VALUES(@i_TEST1)

        SET NOCOUNT OFF
COMMIT TRAN
-----------------------------------------------------------------------
EXEC SP_TEST 1
SELECT * FROM TEST00   --Trường hợp 3
```

   Sinh viên phân tích kết quả và chụp hình kết quả của hai giao dịch vào file Word theo các trường hợp sau:
   - ✓ Trường hợp 1: Tương tự cấu trúc trên
   - ✓ Trường hợp 2: Đưa câu lệnh WAITFOR DELAY xuống dưới câu INSERT (ở cả 2 giao dịch)
   - ✓ Trường hợp 3: Đưa câu SELECT từ bên ngoài vào dưới câu WAITFOR DELAY (của trường hợp 2).

----------o0o----------