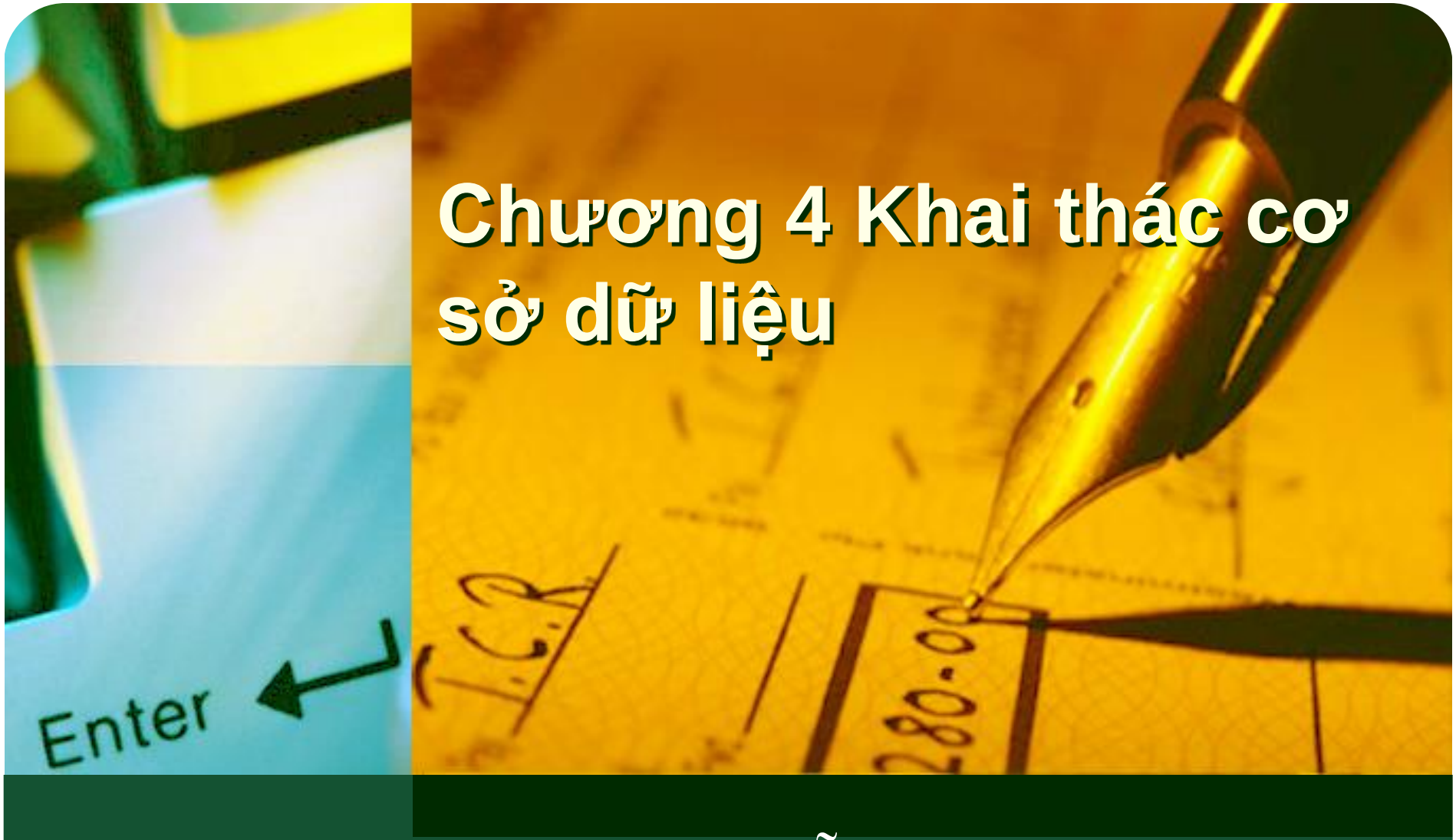


Chương 4 Khai thác cơ sở dữ liệu

Giáo viên: Nguyễn Mậu Uyên

Bộ môn: Hệ thống thông tin

Biên soạn: Đỗ Thị Mai Hường

Nội dung

- Giao dịch (Transaction)
  - Khái niệm giao dịch
  - Các tính chất ACID của giao dịch
  - Sử dụng giao dịch trong SQL server
- Các vấn đề của truy xuất đồng thời
- Lịch biểu (Schedule)
- Kỹ thuật khóa trong SQL Server

Giao dịch

- Giao dịch là 1 *đơn vị xử lý nguyên tố* gồm 1 chuỗi các hành động tương tác lên CSDL.
- Khi thực hiện một giao dịch hoặc phải *thực hiện tất cả các hành động* của nó *hoặc không thực hiện hành động nào* hết.
- Một số thuật ngữ liên quan đến giao dịch:
  - 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 dịch đã thực hiện trước đó

Giao dịch (tt)

LOGO

Tính chất ACID của giao dịch

- **Tính nguyên tố (Atomicity):** Không thể chia nhỏ được nữa
- **Tính nhất quán (Consistancy)**
  - Giao dịch không phá vỡ trạng thái nhất quán của CSDL
- **Tính độc lập (Isolation)**
  - Giao dịch không ảnh hưởng/ chịu ảnh hưởng của giao dịch khác

Giao dịch (tt)

LOGO

Tính chất ACID của giao dịch

❖ Tính bền vững (Durability)

- Sau khi giao dịch commit thành công, tất cả những thay đổi trên CSDL mà giao dịch 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*

Giao dịch (tt)

LOGO

Để đảm bảo tính Atomicity

- Để đảm bảo tính chất A của giao dịch, người sử dụng phải điều khiển tường minh sự rollback của giao dịch.
- Cần kiểm tra lỗi sau khi thực hiện mỗi thao tác trong giao dịch để có thể xử lý rollback kịp thời
  - Kiểm tra lỗi: dùng try…catch hoặc @@error
- Ví dụ:

TAIKHOAN (MaTK, ChuTK, SoDuTK)

Viết thủ tục để chuyển khoản một số tiền từ tài khoản này sang tài khoản khác

```
create proc usp_ChuyenKhoan
  @tkdi char(10), @tkden char(10), @sotien int
as
begin
    begin try
        BEGIN TRAN
            SET XACT_ABORT ON
            update TaiKhoan
            set SoDuTK=SoDuTK-@sotien
            where MaTK= @tkdi

            update TaiKhoan
            set SoDuTK=SoDuTK+@sotien
            where MaTK=@tkden
        COMMIT TRAN
    end try
    begin catch
        declare @loi nvarchar(100)
        set @loi=N'Lỗi:'+ Error_message()
        raiserror (@loi,16,1)
        ROLLBACK TRAN
        return
    end catch
end
```

Truy xuất đồng thời

- Giao dịch (Transaction)
- Các vấn đề của truy xuất đồng thời
  - Mất dữ liệu đã cập nhật (Lost Updated)
  - Đọc phải dữ liệu rác (Dirty Read)
  - Không thể đọc lại (Unrepeatable Read)
  - “Bóng ma” dữ liệu (Phantom)
- Lịch biểu (Schedule)
- Kỹ thuật khóa trong SQL Server

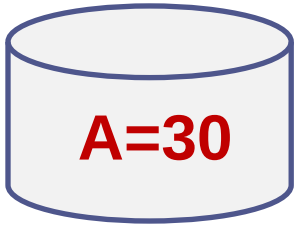
Truy xuất đồng thời

- Xử lý truy xuất đồng thời: các yêu cầu truy xuất xảy ra cùng một lúc (thực hiện đan xen với nhau)
- Có 2 loại:
  - Không tranh chấp: n yêu cầu truy xuất trên n đơn vị dữ liệu.
  - Có tranh chấp:

Ví dụ: trong tài khoản còn có 120, 2 người truy xuất cùng một tài khoản: 1 người rút 100, người kia rút 80.

Mất dữ liệu đã cập nhật (Lost Updated)

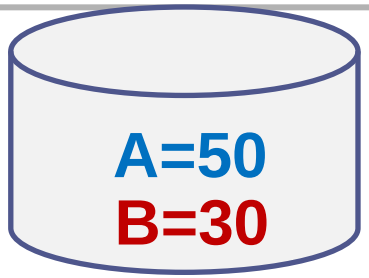
- P_1 và P_2 xử lý đồng thời:



	P_1		P_2	
t_1	Read(A)	A=20		
t_2			Read(A)	A=20
t_3	A=A-5	A=15		
t_4			A=A+10	A=30
t_5	Write(A)	A=15		
t_6			Write(A)	A=30
t_7	Read(A)	A=30		

Đọc phải dữ liệu rác (Dirty Read)

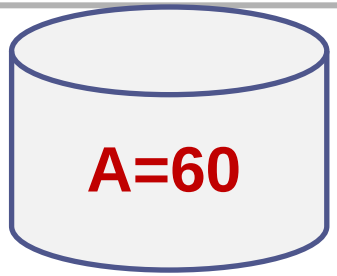
- P_1 và P_2 xử lý đồng thời:



	P_1	P_2
t_1	Read(A) A=50	
t_2		Read(B) B=30
t_3		B=B+ 10 B=40
t_4		Write(B) B=40
t_5	Read(B) B=40	
t_6	C=A+B C=90	
t_7	Print(C) C=90	
t_8		Rollback

Không thể đọc lại (Unrepeatable Read)

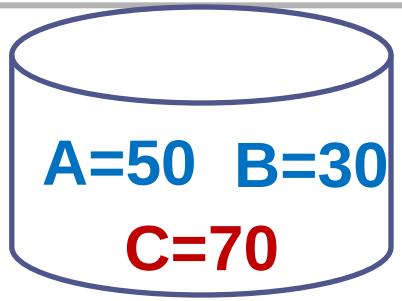
- P_1 và P_2 xử lý đồng thời:



	P_1	P_2
t_1	Read(A) A=50	
t_2	Print(A) A=50	
t_3		Read(A) A=50
t_4		A=A+10 A=60
t_5		Write(A) A=60
t_6	Read(A) A=60	

Bóng ma dữ liệu (Phantom)

- P_1 và P_2 xử lý đồng thời:



	P_1	P_2
t_1	Read(>40) A=50	
t_2		C=70
t_3		Write(C) C=70
t_4	Read(>40) A=50 C=70	

Lịch biểu (Schedule)

- Giao dịch (Transaction)
- Các vấn đề của truy xuất đồng thời
- Lịch biểu (Schedule)
- Kỹ thuật khóa trong SQL Server

Lịch biểu (Schedule)

- **Định nghĩa:** 1 lịch biểu được lập từ n giao dịch xử lý đồng thời T1,T2,Tn là một thứ tự thực hiện các hành động của n giao dịch đó.
- **Ví dụ:** Giả sử chỉ có 2 giao tác T1,T2 xử lý đồng thời thì có thể hình thành lịch biểu: S1 thực hiện 2 lần hành động thứ nhất của T1, sau đó thực hiện 3 lần hành động thứ nhất của T2

Lịch biểu (Schedule)

- Lịch tuần tự: lịch thực hiện hết giao tác này mới đến giao tác kia
- Định nghĩa: một lịch S được lập từ các giao tác T1,T2,..Tn được gọi là lịch tuần tự nếu các hành động của mỗi Ti được thực hiện liên tiếp nhau.
- Không có hệ quản trị nào lại thực hiện các giao tác đồng thời tuần tự.

Lịch biểu (Schedule)

Lịch khả tuần tự: Serializable schedule

Một lịch S được lập từ n giao tác xử lý đồng thời T1,T2, ..Tn được gọi là một lịch khả tuần tự nếu nó cho kết quả giống như một lịch tuần tự nào đó được lập từ n giao tác đã cho.

Chú ý: chỉ cần cho kết quả giống với một lịch tuần tự nào đó.

Bộ lập lịch (Scheduler)

- Bộ phận của HQTCSDL nhận vào n giao tác đồng thời thì đưa ra một lịch khả tuần tự S để thực hiện n giao tác đó.
- Ví dụ:
  - Hàng năm thanh niên đi khám nghĩa vụ quân sự, có n thanh niên cùng vào khám một lúc. Nếu các sắp xếp tuần tự là n thanh niên xếp hàng và lần lượt vào các phòng khám.
  - Giải pháp khác là chia nhóm lần lượt khám ở các phòng khác nhau.

Bộ lập lịch (Scheduler)

- Để bảo đảm cho việc bộ lập lịch có thể lập được một lịch khả tuần tự thì các giao tác Ti phải được viết tuân thủ một số nghi thức cho trước.
- Ví dụ: Xét 2 giao tác xử lý đồng thời

T1	T2
Read(A)	Read(A)
A:=A-N	A=A+M
Write(A)	Write(A)
Read(B)	
B=B+N	
Write(B)	

Bộ lập lịch (Scheduler)

Lịch S1:

T1	T2
Read(A)	
A:=A-N	
Write(A)	
Read(B)	
B=B+N	
Write(B)	
	Read(A)
	A=A+M
	Write(A)

Bộ lập lịch (Scheduler)

LOGO

Lịch S2:

T1	T2
	Read(A)
	A=A+M
	Write(A)
Read(A)	
A:=A-N	
Write(A)	
Read(B)	
B=B+N	
Write(B)	

Bộ lập lịch (Scheduler)

Lịch S3:

T1	T2
Read(A)	
A:=A-N	
	Read(A)
	A=A+M
Write(A)	
Read(B)	
	Write(A)
B=B+N	
Write(B)	

Bộ lập lịch (Scheduler)

Lịch S4:

T1	T2
Read(A)	
A:=A-N	
Write(A)	
	Read(A)
	A=A+M
	Write(A)
Read(B)	
B=B+N	
Write(B)	

Bộ lập lịch (Scheduler)

Lich S1:

T1	T2
Read(A)	
A:=A-N	
Write(A)	
Read(B)	
B=B+N	
Write(B)	
	Read(A)
	A=A+M
	Write(A)

Lich S2:

T1	T2
	Read(A)
	A=A+M
	Write(A)
Read(A)	
A:=A-N	
Write(A)	
Read(B)	
B=B+N	
Write(B)	

Lich S3:

T1	T2
Read(A)	
A:=A-N	
	Read(A)
	A=A+M
Write(A)	
Read(B)	
	Write(A)
B=B+N	
Write(B)	

Lich S4:

T1	T2
Read(A)	
A:=A-N	
Write(A)	
	Read(A)
	A=A+M
	Write(A)
Read(B)	
B=B+N	
Write(B)	

Giả sử A=10, B=20,N=30,M=40. Tính A, B trong các lịch:
S1: A=20,B=50; S2: A=20, B=50; S3: A=50;B=50; S4: A=20,B=50;

Bộ lập lịch (Scheduler)

- S4 là lịch khả tuần tự
- Bộ lập lịch SQLServer cố gắng thực hiện

❖ Nhận xét:

- Các hành động thực hiện trên cùng một đơn vị dữ liệu thì 2 hành động Read có tính chất giao hoán

Bộ lập lịch (Scheduler)

Thuật toán kiểm tra tính khả tuần tự của lịch

- Input: một lịch S được lập từ n giao tác xử lý đồng thời
- Output: lịch S khả tuần tự hay không?

❖ **Thuật toán:**

Xây dựng một đồ thị có hướng G như sau:

- Mỗi giao tác Ti là một đỉnh của đồ thị G
- Nếu có một Tj phát ra một yêu cầu Write(A) sau một giao tác Ti đã phát ra yêu cầu Read(A) thì vẽ cung từ Ti đến Tj.

Bộ lập lịch (Scheduler)

Thuật toán kiểm tra tính khả tuần tự của lịch

Xây dựng một đồ thị có hướng G như sau:

- Nếu có một Tj phát ra một yêu cầu Read(A) sau một giao tác Ti đã phát ra yêu cầu Write(A) thì vẽ cung từ đỉnh Ti đến Tj
- Nếu có một Tj phát ra một yêu cầu Write(A) sau một giao tác Ti đã phát ra yêu cầu Write(A) thì vẽ cung từ đỉnh Ti đến Tj

Kết luận: Nếu G mà có chu trình thì S không khả tuần tự

Bộ lập lịch (Scheduler)

Ví dụ: Xây dựng đồ thị kiểm tra bộ lập lịch S1,S2,S3,S4

Nhận xét: một lịch khả tuần tự sẽ tương đương với lịch tuần tự nào: chỉ cần chỉ ra một đường đi trong đồ thị G thì đấy chính là lịch tuần tự cần tìm.

Kỹ thuật khóa trong SQLServer

- Giao dịch (Transaction)
- Các vấn đề của truy xuất đồng thời
- Lịch biểu (Schedule)
- Kỹ thuật khóa trong SQL Server
  - Kỹ thuật khóa (Locking)
  - Các mức độ cô lập
  - Khóa trực tiếp trong câu lệnh
  - Deadlock

Kỹ thuật khóa

- Một giao dịch P trước khi muốn thao tác (read/write) lên một đơn vị dữ liệu A phải phát ra một yêu cầu xin khóa A: lock(A)
- Nếu yêu cầu được chấp thuận thì giao dịch P mới được phép thao tác lên đơn vị dữ liệu A
- Sau khi thao tác xong, giao dịch P phải phát ra lệnh giải phóng A: unlock(A)

Ví dụ kỹ thuật khóa

	T_1	T_2
t_1	Lock(A), A=5	
t_2	0	Lock(A), A=5
t_3	A=2	0
t_4	0	A=8
t_5	A=20	0
t_6	0	A=8
t_7	, A=80	Unlock(A 0
	Unlock(A 0	)

	T_1	T_2
t_1	Lock(A),	
t_2		
t_3		
t_4	,	Lock(A),
t_5	Unlock(A)	
t_6		
t_7		, Unlock(A)

Khóa đọc + khóa ghi

LOGO

- **Read lock**
  - Giao dịch giữ Slock được phép **ĐỌC** dữ liệu, nhưng không được phép ghi.
  - ***Nhiều giao dịch có thể đồng thời giữ Slock*** trên cùng 1 đơn vị dữ liệu

- **Write lock**
  - Giao dịch giữ Xlock được phép **GHI + ĐỌC** dữ liệu
  - Tại 1 thời điểm chỉ có tối đa ***1 giao dịch được quyền giữ Xlock*** trên 1 đơn dữ liệu.
  - Không thể thiết lập Slock trên đơn vị dữ liệu đang có dạng Xlock.

Khóa dự định ghi

- **Update lock = Intent - to - update lock (Ulock): Khóa dự định ghi**
  - Ulock sử dụng khi đọc dữ liệu với dự định ghi trở lại trên dữ liệu này.
  - Ulock là chế độ khoá trung gian giữa Slock và Xlock
  - Khi thực hiện thao tác ghi lên dữ liệu thì bắt buộc Ulock phải tự động chuyển thành Xlock
  - Giao dịch giữ Ulock được phép **GHI + ĐỌC** dữ liệu
  - Tại 1 thời điểm chỉ có tối đa ***1 giao dịch được quyền giữ Ulock*** trên 1 đơn dữ liệu.
  - Có thể thiết lập Slock trên đơn vị dữ liệu đang có dạng Ulock

Bảng tương thích giữa các chế độ khóa

gt T vs *gt* T'	S	U	X
S	✓	✓	✗
U	✓	✗	✗
X	✗	✗	✗

Tương thích: T' không phải chờ T
Không tương thích: T' phải chờ T giải phóng khóa

Truy xuất đồng thời

- Giao dịch
- Các vấn đề của truy xuất đồng thời
- Lịch biểu
- Kỹ thuật khóa trong SQL Server
  - Kỹ thuật khóa (Locking)
  - Mức cô lập trong giao dịch
  - Khóa trực tiếp trong câu lệnh
  - Deallock

Mức độ cô lập của giao dịch

LOGO

- **Mục đích:**
  - Tự động đặt khóa cho các thao tác (đọc) trong kết nối dữ liệu hiện hành.
- **Các mức độ cô lập**
  - Read Uncommitted
  - Read Committed
  - Repeatable Read
  - Serializable

Read Uncommitted

- **Đặc điểm:**
  - **Đọc dữ liệu**: không cần phải thiết lập SLock
  - **Ghi dữ liệu**: SQL Server tự động thiết lập XLock trên đơn vị dữ liệu được ghi, ***XLock được giữ cho đến hết giao dịch***

Read Uncommitted + vấn đề Lost Updated

LOGO

KHACHHAN

MaKH	TenKH	DiaChi	DienThoai
KH001	XYZXYZ	Đà Nẵng	0511.3246135
KH002	Cửa hàng Hoàng Gia	Quảng Nam	0510.6333444
KH003	Nguyễn Lan Anh	Huế	0988.148248
KH004	Công ty TNHH An Phước	Đà Nẵng	0511.6987789
KH005	Huỳnh Ngọc Trung	Quảng Nam	0905.888555
KH006	Cửa hàng Trung Tín	Đà Nẵng	*NULL*

Xlock

Xlock

T1	T2
BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED* UPDATE KhachHang SET TenKH= 'ABC' WHERE MaKH= 'KH001' WAITFOR DELAY '00:00:05' SELECT TenKH FROM KhachHang WHERE MaKH= 'KH001' COMMIT TRAN	BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED* UPDATE KhachHang SET TenKH= 'XYZ' WHERE MaKH= 'KH001' COMMIT TRAN

Read Uncommitted + vấn đề Dirty Read

LOGO

KHACHHAN

MaKH	TenKH	DiaChi	DienThoai
KH001	ABC	Đà Nẵng	0511.3246135
KH002	Cửa hàng Hoàng Gia	Quảng Nam	0510.6333444
KH003	Nguyễn Lan Anh	Huế	0988.148248
KH004	Công ty TNHH An Phước	Đà Nẵng	0511.6987789
KH005	Huỳnh Ngọc Trung	Quảng Nam	0905.888555
KH006	Cửa hàng Trung Tín	Đà Nẵng	*NULL*

Xlock

T1	T2
BEGIN TRAN ***SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED*** UPDATE KhachHang SET TenKH= 'ABC' WHERE MaKH= 'KH001' WAITFOR DELAY '00:00:05' ROLLBACK TRAN	BEGIN TRAN ***SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED*** SELECT TenKH FROM KhachHang WHERE MaKH= 'KH001' COMMIT TRAN

Read Uncommitted

- **Ưu điểm:**
  - Giải quyết vấn đề Lost Updated
  - Không cần thiết lập Slock khi đọc=> không cản trở giao dịch khác giữ khóa Xlock.
- **Hạn chế:**
  - Có khả năng xảy ra 3 vấn đề của truy xuất đồng thời: Dirty Read, Unrepeatable Read, Phantom

Read Committed

LOGO

- **Đặc điểm:**
  - **Đọc dữ liệu**: SQL server tự động thiết lập SLock trên đơn vị dữ liệu được đọc, SLock được giải phóng ngay sau khi đọc xong
  - **Ghi dữ liệu**: SQL server tự động thiết lập XLock trên đơn vị dữ liệu được ghi, XLock được giữ cho đến hết giao dịch

Read Commited + vấn đề Dirty Read

KHACHHAN

MaKH	TenKH	DiaChi	DienThoai
KH001	ABC Văn Tuyến	Đà Nẵng	0511.3246135
KH002	Cửa hàng Hoàng Gia	Quảng Nam	0510.6333444
KH003	Nguyễn Lan Anh	Huế	0988.148248
KH004	Công ty TNHH An Phước	Đà Nẵng	0511.6987789
KH005	Huỳnh Ngọc Trung	Quảng Nam	0905.888555
KH006	Cửa hàng Trung Tín	Đà Nẵng	*NULL*

Xlock

Slock

T1	T2
BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL READ COMMITTED* UPDATE KhachHang SET TenKH= 'ABC' WHERE MaKH= 'KH001' WAITFOR DELAY '00:00:05' ROLLBACK TRAN	BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL READ COMMITTED* SELECT TenKH FROM KhachHang WHERE MaKH= 'KH001' COMMIT TRAN

Read Commited + vấn đề Unrepeatable Read

MaKH	TenKH	DiaChi	DienThoai
KH001	ABC	Đà Nẵng	0511.3246135
KH002	Cửa hàng Hoàng Gia	Quảng Nam	0510.6333444
KH003	Nguyễn Lan Anh	Huế	0988.148248
KH004	Công ty TNHH An Phước	Đà Nẵng	0511.6987789
KH005	Huỳnh Ngọc Trung	Quảng Nam	0905.888555
KH006	Cửa hàng Trung Tín	Đà Nẵng	*NULL*

Slock

Tuyên

Xlock

T1	T2
BEGIN TRAN SET TRANSACTION ISOLATION LEVEL READ COMMITTED SELECT TenKH FROM KhachHang WHERE MaKH = 'KH001' WAITFOR DELAY '00:00:05' SELECT TenKH FROM KhachHang WHERE MaKH = 'KH001' COMMIT TRAN	BEGIN TRAN SET TRANSACTION ISOLATION LEVEL READ COMMITTED UPDATE KhachHang SET TenKH= 'ABC' WHERE MaKH= 'KH001' COMMIT TRAN

Read Committed

- **Ưu điểm:**
  - Giải quyết vấn đề Dirty Read, Lost Updated
  - SLock được giải phóng ngay ==> không cản trở nhiều đến thao tác ghi dữ liệu của các giao dịch khác.

- **Hạn chế:**
  - Chưa giải quyết được vấn đề Unrepeatable Read, Phantom

Repeatable Read

- **Đặc điểm:**
  - **Đọc dữ liệu**: SQL server tự động thiết lập Slock trên đơn vị dữ liệu được đọc và giữ Slock đến hết giao dịch.
  - **Ghi dữ liệu**: SQL server tự động thiết lập XLock trên đơn vị dữ liệu được ghi, XLock được giữ cho đến hết giao dịch.

Repeatable Read + vấn đề Unrepeated Read

MaKH	TenKH	DiaChi	DienThoai
KH001	ABC	Đà Nẵng	0511.3246135
KH002	Cửa hàng Hoàng Gia	Quảng Nam	0510.6333444
KH003	Nguyễn Lan Anh	Huế	0988.148248
KH004	Công ty TNHH An Phước	Đà Nẵng	0511.6987789
KH005	Huỳnh Ngọc Trung	Quảng Nam	0905.888555
KH006	Cửa hàng Trung Tín	Đà Nẵng	*NULL*

Slock

Tuyen

Xlock

T1	T2
BEGIN TRAN ***SET TRANSACTION ISOLATION LEVEL REPEATABLE READ*** SELECT TenKH FROM KhachHang WHERE MaKH = 'KH001' WAITFOR DELAY '00:00:05' SELECT TenKH FROM KhachHang WHERE MaKH = 'KH001' COMMIT TRAN	BEGIN TRAN ***SET TRANSACTION ISOLATION LEVEL REPEATABLE READ*** UPDATE KhachHang SET TenKH= 'ABC' WHERE MaKH= 'KH001' COMMIT TRAN

Repeatable Read + vấn đề Phantom

HANGHOA

MaHH	TenHH	DVT	SLCon	DonGiaHH
BU	Bàn ủi Philip	Cái	60	400000
CD	Nồi cơm điện Sharp	Cái	100	350000
DM	Đầu máy Sharp	Cái	75	350000
MG	Máy giặt SanYo	Cái	10	350000
MQ	Máy quạt ASIA	cái	40	350000
TL	Tủ lạnh Hitachi	Cái	50	350000
TV	TiVi JVC 14WS	Cái	33	350000
IP	IPad	IPad	Cái	100	10000000

Nồi cơm điện Sharp

Slock

Slock

T1	T2
BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL REPEATABLE READ* SELECT * FROM HangHoa WHERE SLCon = 100 WAITFOR DELAY '00:00:05' SELECT * FROM HangHoa WHERE SLCon = 100 COMMIT TRAN	BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL REPEATABLE READ* INSERT INTO HangHoa VALUES ('IP','Ipad','Cái',100,10000000) COMMIT TRAN

Repeatable Read

- **Ưu điểm:**
  - Giải quyết được 3 vấn đề: Lost Updated, Dirty Read và Unrepeatable Read

- **Khuyết điểm:**
  - Chưa giải quyết được vấn đề Phantom, do vẫn cho phép insert những dòng dữ liệu thỏa điều kiện thiết lập Slock
  - Slock được giữ đến hết giao dịch ==> cản trở việc cập nhật dữ liệu của các giao dịch khác

Serializable

LOGO

- **Đặc điểm:**
  - **Đọc dữ liệu**: SQL server tự động thiết lập SLock trên đơn vị dữ liệu được đọc và giữ Slock này đến hết giao dịch
  - Không cho phép thêm những dòng dữ liệu thỏa mãn điều kiện thiết lập Slock
  - **Ghi dữ liệu**: SQL server tự động thiết lập XLock trên đơn vị dữ liệu được ghi, ELock được giữ cho đến hết giao dịch

Serializable + vấn đề Phantom

HANGHOA

MaHH	TenHH	DVT	SLCon	DonGiaHH
BU	Bàn ủi Philip	Cái	60	400000
CD	Nồi cơm điện Sharp	Cái	100	350000
DM	Đầu máy Sharp	Cái	75	350000
MG	Máy giặt SanYo	Cái	10	350000
MQ	Máy quạt ASIA	cái	40	350000
TL	Tủ lạnh Hitachi	Cái	50	350000
TV	TiVi JVC 14WS	Cái	33	350000
IP	IPad	Cái	100	10000000

Nồi cơm điện Sharp

Slock

T1	T2
BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL REPEATABLE READ* SELECT * FROM HangHoa WHERE SLCon = 100 WAITFOR DELAY '00:00:05' SELECT * FROM HangHoa WHERE SLCon = 100 COMMIT TRAN	BEGIN TRAN *SET TRANSACTION ISOLATION LEVEL REPEATABLE READ* INSERT INTO HangHoa VALUES ('IP','Ipad','Cái',100,10000000) COMMIT TRAN

Serializable

LOGO

- **Ưu điểm:**
  - Giải quyết được 4 vấn đề: Lost Updated, Dirty Read, Unrepeatable Read và Phantom

- **Khuyết điểm:**
  - Slock được giữ đến hết giao dịch ==> cản trở việc cập nhật dữ liệu của các giao dịch khác
  - Không cho phép Insert những dòng dữ liệu thỏa mãn điều kiện thiết lập Slock ==> cản trở việc thêm mới dữ liệu của các giao dịch khác

Kỹ thuật khóa trong SQL Server

LOGO

- **Lưu ý:**
  - Mức cô lập mặc định trong SQL Server là Read Commited
  - Mức cô lập không quan tâm khóa Ulock
  - Tầm vực của Isolation level là ở mức connection chứ không phải mức transaction.
    - Khi 1 connection N được đặt mức cô lập X thì X sẽ phát huy hiệu lực trên tất cả các transaction Ti chạy trên N

LOGO

Kỹ thuật khóa trong SQL Server

- Kỹ thuật khóa (Locking)
- Mức cô lập trong giao dịch
- **Khóa trực tiếp trong câu lệnh**
- Deadlock

Kỹ thuật khóa trong SQL Server

LOGO

- Mức cô lập quyết định cách phát và giữ khóa S trong một transaction và có hiệu lực trên tất cả các thao tác đọc trong transaction đó.
- Thực tế, ta cần phát và giữ khóa Slock theo các cách khác nhau cho các thao tác đọc khác nhau trong cùng một transaction

Khóa trực tiếp trong câu lệnh

- **Cú pháp:**

```
SELECT ...
FROM table1 WITH (lock1 [, lock2,…] )
WHERE ...

DELETE FROM table1 WITH (lock1 [, lock2, …])
WHERE ...

UPDATE table1 WITH  (lock1 [, lock2, …])
SET ...
WHERE ...
```

Các chế độ khóa trực tiếp (lock mode)

1	READUNCOMITTED/ NOLOCK	- Không thiết lập shared lock khi đọc (tương tự mức cô lập read uncommitted)
2	READCOMMITTED	- Đây là chế độ mặc định (tương tự mức cô lập read committed) - Chỉ đọc những dữ liệu đã được commit - Thiết lập shared lock trên đơn vị dữ liệu cần đọc và mở lock ra ngay sau khi đọc xong
3	REPEATABLEREAD	Thiết lập shared lock khi select và giữ shared lock đến hết giao tác (tương tự mức cô lập repeatable read)
4	SERIALIZABLE/ HOLDLOCK	- Thiết lập shared lock khi đọc và giữ đến hết giao tác - Tương tự như sử dụng Isolation Level là Serializable
5	UPDLOCK	- Sử dụng Updatelock thay vì Shared lock.

6	XLOCK	- khoá độc quyền
7	READPAST	- Chỉ có thể sử dụng trong lệnh Select và chỉ áp dụng trên khóa của dòng dữ liệu (row-lock). Những dòng bị khóa sẽ được bỏ qua.
8	ROWLOCK	- Khóa chỉ những dòng cần thao tác
9	TABLOCK	- Khóa toàn bộ bảng trong CSDL. - Các thao tác cập nhật (insert/ delete/ update) của những giao tác khác không thể thực hiện trên bảng này trong khi khóa vẫn đang được giữ.
10	TABLOCKX	- xlock+tablock

Kết hợp Mức cô lập + Khóa trực tiếp

- Trong transaction luôn có các thao tác yêu cầu bảo vệ nghiêm ngặt và các thao tác ít yêu cầu bảo vệ nghiêm ngặt
- Dùng mức cô lập ứng với yêu cầu bảo vệ ít nghiêm ngặt nhất
- Bổ sung lock trực tiếp vào các thao tác yêu cầu bảo vệ nghiêm ngặt hơn mức mà mức cô lập đó cung cấp.

Kỹ thuật khóa trong SQL Server

LOGO

Kỹ thuật khóa trong SQL Server

- Kỹ thuật khóa (Locking)
- Mức cô lập trong giao dịch
- Khóa trực tiếp trong câu lệnh
- **Deadlock**

Deadlock

- Khi xử lý đồng thời, không tránh khỏi việc transaction này phải chờ đợi transaction khác
- Nếu vì lý do gì đó mà hai transaction lại chờ lẫn nhau vĩnh viễn, không cái nào trong hai có thể hoàn thành được thì ta gọi đó là hiện tượng Dead Lock

T_1	T_2	*Ghi chú*
Lock(A)		*T_1 đợi T_2 unlock(B) trong khi T_2 lại đợi T_1 unlock(A) để tiếp tục thực hiện. Hai giao dịch này cứ chờ nhau và cả hai đều không chạy được.*
	Lock(B)	
Lock(B) *(Chờ)*		
	Lock(A) *(chờ)*	

Xử lý Deadlock trong SQL Server

LOGO

- SQL Server sẽ chọn 1 trong 2 transaction gây deadlock để hủy bỏ, khi đó transaction còn lại sẽ được tiếp tục thực hiện cho đến khi hoàn tất
- Transaction bị chọn hủy bỏ là transaction mà SQL ước tính chi phí cho phần việc đã làm được ít hơn transaction còn lại.

Xử lý DeadLock trong SQL Server

LOGO

T1	T2
BEGIN TRAN SET ANSI_WARNINGS ON UPDATE KhachHang with (XLOCK) SET TenKH = 'ABC' WHERE MAKH = 'KH001' WAITFOR DELAY '00:00:05'	BEGIN TRAN SET ANSI_WARNINGS ON UPDATE KhachHang with (XLOCK) SET TenKH = '123' WHERE MAKH = 'KH002' UPDATE KhachHang with (XLOCK) SET TenKH = '456' WHERE MAKH = 'KH001' COMMIT TRAN
UPDATE KhachHang with (XLOCK) SET TenKH = 'XYZ' WHERE MAKH = 'KH002' COMMIT TRAN	

Bài tập

- Bài 1: Xác định các cập nhật trong hệ thống cần phải thực hiện theo phiên trên «Hệ thống quản lý bán sách online»
- Bài 2: Xác định các cập nhật trong hệ thống cần phải thực hiện theo phiên trên «Hệ thống quản lý đào tạo LQDUNI»