트랜잭션 : 사용자 질의처리 한꺼번에 하는 묶음 단위

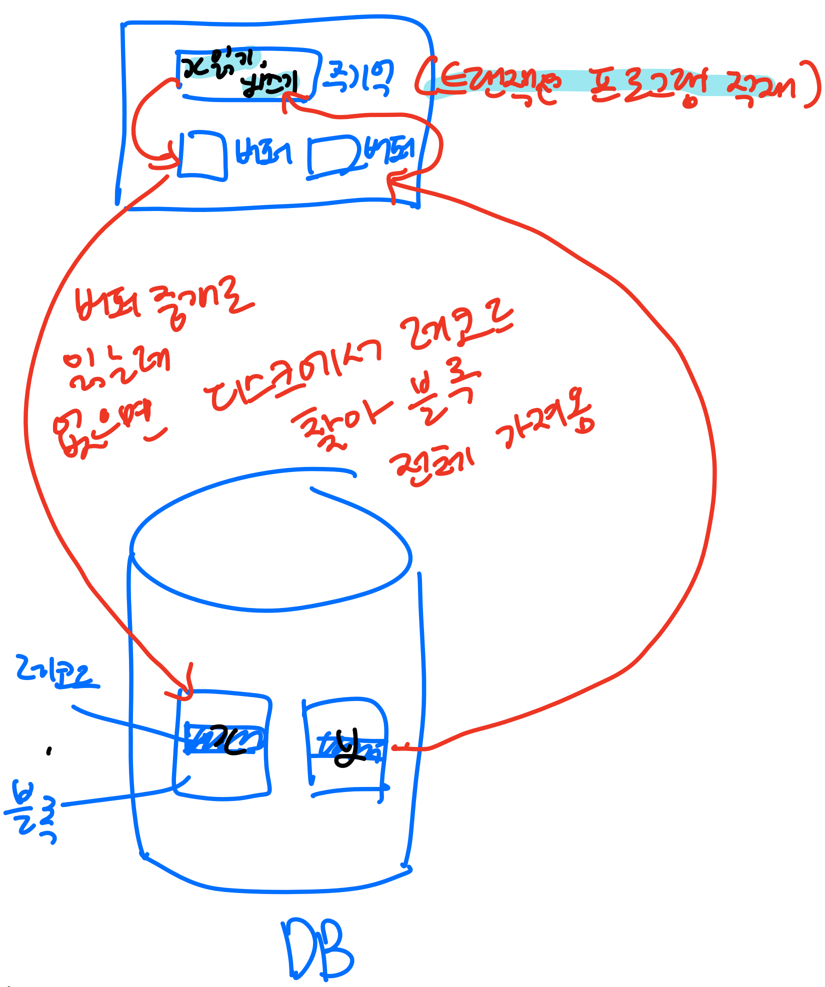
- 트랜잭션 → must be **ACID**
  - [원자성 / 일관성 / 고립성 / 지속성]
  - 고립성 ← [Lv0. 잘못쓰기 ㄴㄴ (오손읽기한번 안됨) / Lv1. 갱신 ㄴㄴ (Read uncommit) / Lv2. 갱신+오손 ㄴㄴ (R com) / Lv3. 갱신+오손+비반복 ㄴㄴ (serinlized)]
- 이게 여러개!? → 동시성 제어 필요
- 여러 sql을.. 고장!? → 회복 가능 필요
- 동시성 제어 필요, 회복 가능 필요 → ACID

* 다중 사용자 시스템 현실...

인터리빙 or 다중프로세스로 병렬처리.
(시분할.. 일정기간에선 동시에 하는거와 같음)
트랜잭션 동시!

트랜잭션 작업 종류
- read
- write

같은 데이터 동시참조하면 버퍼.블록 구조상 문제 생길수 있는 이유.



동시성 제어 안하면 생기는 문제

1. 갱신손실 (잘못 쓰는 문제.. 제일 심각)

T_1	T_2
x : 100 → 50	
	x : 50 → 250 커밋
y : x + 500 = 550	
롤백	

이런 문제로 인해 롤백해서 처음 시점을 돌아가면 x는 100으로 돌아가여. 사이에 T_2가 x를 250으로 올려준 사실이 없어짐.

→ T_2의 입장에선 쓰기가 (write) 손해 발생

2. 오손 읽기 (잘못 쓰는건 아닌데 잘못 읽음)

T_1	T_2
x: 100→50	
	print(x) 커밋
y: x+500 = 550	
롤백	

위와 같은 상황 가정

T1이 실패하면 x의 값은 100으로 되돌아가는데 돌아가기전 임시의 값 50을 T2가 읽어버림.

읽기에 문제 발생 (read).

위와 다른부분

3. 부정확한 요약 (잘못 쓴거도, 잘못 읽은거도 아닌데 시점때문에 통계 잘못됨)

||

오손, 비반복적 읽기.

T_1	T_2
x: 100→50	
	S: x+y =50
y: x+500 = 550	

한 트랜잭션 안에서 보면 (커밋 기준)

커밋

2. 2와 다르게 우시히 커밋.

S:450;
600

커밋.

St50은 100이어야 하는데 650...
통계에 일관성이 없음.

4. 유령읽기 (잘못 읽기인데 없던걸 읽게되는게 더욱 문제!)

T_1	T_2
$R(x_1 \sim x_3)$ $= x_1, x_3$.	
	$w(x_2)$
Commit. (x_1, x_2, x_3).	

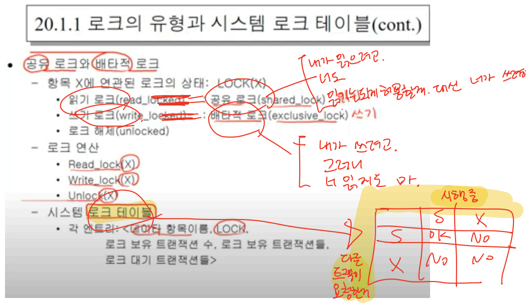
* 1~4 다 비슷한 문제인데
어떤 행위에 영향을 주는지를
구분하는 듯하다.

동시성 제어 방법?

* 직렬 스케줄
* 비직렬 스케줄.

(직렬가능성 따지는거 생각)

• 로킹 으로 구현. →

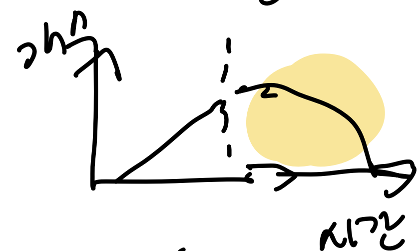


20.1.1 로크의 유형과 시스템 로크 테이블(cont.)

- 공유 로크와 배타적 로크
  - 항목 X에 연관된 로크의 상태: LOCK(X)
    - 읽기 로크(read_locked) : 공유 로크(shared_lock)
    - 쓰기 로크(write_locked) : 배타적 로크(exclusive_lock)
    - 로크 해제(unlocked)
  - 로크 연산
    - Read_lock(X)
    - Write_lock(X)
    - Unlock(X)
  - 시스템 로크 테이블
    - 각 엔트리: <데이타 항목이름, LOCK, 로크 보유 트랜잭션 수, 로크 보유 트랜잭션들, 로크 대기 트랜잭션들>

사용하기전 잠그고 시작하겠다는거.

- 2단계: 잠금/해제하면 연쇄복귀등 발생가능하므로.
  - 확장 (새 락을 획득만...
  - 축소 (락들을 풀기만...



- 낙관적 (생각)

* 락 단위크기
  - 레코드 필드 < ..... < DB전체.
  - 성능영향 O
  - 작은단위일수록 병렬가능해서 동시성↑ 성능↑
  - 대신 락테이블 체크 많아짐.

  요즘은 동적으로 크기조정!

* 교착상태 발생가능 ㅠ ㅠ

* 타임스탬프 (생각)

* 다중버전 (생각)

회복

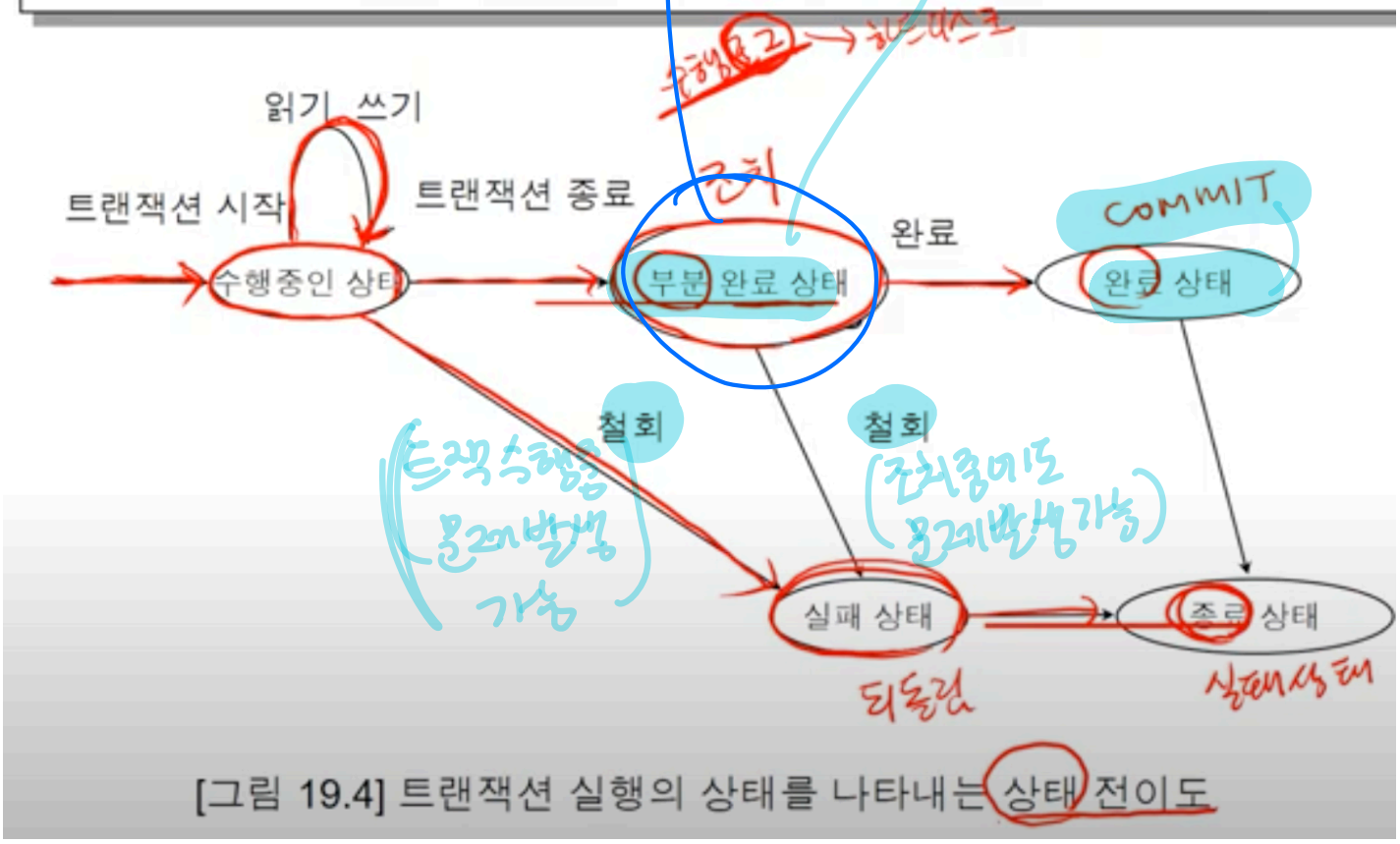
- 고장에 대한 것..
  - 트랜잭션 (실행 불가능 오류) → 트랜잭션 회복가능.
  - 시스템 (프로그램 망감)
  - 매체 (PC재부팅해도 하드자체가 망감)
    - → 미리 백업필요
- 목표... 완전 수행 or 완전 미수행
- how?

일단 상태를 구분해서 대응.

트랜잭션이 끝났더라도 추가조치를 해줘야함...

(끝나고 고장나도 돌아갈 수 있게 ... 로그 기록!)

19.2.1 트랜잭션의 상태와 추가적인 연산들(cont.)



[그림 19.4] 트랜잭션 실행의 상태를 나타내는 상태 전이도

* 로그

디스크에 작성.

백업 필요 (안그럼)

redo (새것)

undo

* 회복가능성 있는 스케줄 인지?

+ SQL 트랜잭션 자동: 묵시적으로 시작해주니
명시적으로 end만 써주면 됨
(commit 등)

: 옵션지정?

SET TRANSACTION

ISOLATION [|]

READ
UN