병행성제어

트랜잭션

트랜잭션 개요

- 데이터베이스 객체들에 대한 일련의 판독 또는 기록
    - 판독시에는 객체의내용이 디스크에서 버퍼풀의 프레임으로 페이징되고 그 값이 프로그램 변수로 복사됨
    - 기록시에는 프레임의 객체 복사본이 수정된 후 디스크에 기록
- 데이터베이스 객체는 프로그램이 정보를 기록하거나 판독하는단위
    - 페이지 레코드등의 될 수 있으며 DBMS마다 다르게 구현
- ACID
    - 원자성 : 트래잭션과 관련된 작업들이 부분적으로 실행되다가 중단되지 않음을 보장
    - 일관성 : 실행을 완료하면 언제나 일관성 있는 데이터베이스 상태로 유지됨을 보장
    - 격리성 : 트랜잭션 수행시 다른 트랜잭션의 연산작업이 끼어들지 못하도록 보장
    - 지속성 : 성공적으로 수행된 트랜잭션은 영구적으로 데이터베이스에 반영됨을 보장

트랜잭션과 일관성

- 트랜잭션의 인터리빙과 관계없이 모든 트랜잭션은 순서대로 실행한 결과를 보장해야함
    - 두 트랜잭션 t1과 t2가 동시에 실행되더라도 실제 효과는 t1이 종료 된 후 t2가 실행된 결과와 같이야함
    - 격리성 만족
    - 무결성 제약조건을 집행할 메커니즘 제공
- DBMS는 서로 다른 트랜잭션 단위 작업을 인터리빙
    - DBMS는 인터리빙해서 실행한 실제 효과가 트랜잭션들을 선형으로 실행한 것과 동등하도록 보장
- 손상 복구
    - 트랜잭션이 정상적을 완료되지 못하는 경우
        - 트랜잭션을 철회하는 경우
        - 시스템 손상
    - 일관성 보장을 위해 미오나료 트랜잭션의 단위 작업을 무효화함.

트랜잭션과 스케줄

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

- DBMS 관점에서 트랜잭션 하나는 단위 작업 (action-read와 write)들의 나열
  - 트랜잭션은 부부적으로 순서화 된 집합으로 정의할 수 잇음
  - 단위작업의 리스트로 취급할 수 있음
- 트랜잭션 T에서 객체 O를 읽는 작업에서
  - 판독단위 작업Rt(O), 기록 단위작업 Wt(O), 철회작업 At, 완결작업 Ct
- 트랜잭션 스케줄
  - 어던 트랜잭션에 있는 단위작업 리스트
  - DBMS가 보는 트랜잭션 단위 작업들을 기술
  - 각 트랜잭션에 abort와 Commit연산을 넣은 스케줄을 완전한 스케줄로 부름
  - 다른 단위작업이 인터리빙 되지 않는 경우 직렬 스케줄로 부름

일관성

- 직렬 가능성
- 교차수행에 의한 이상
- 기록-판독 충돌(WR Conflict)
- 판독-기록 충돌(RW Conflict)
- 기록-기록 충돌(ww Conflict)

직렬가능성

- 직렬 가능 스케줄(Serializable Schedule)
  - 일관적인 데이터베이스 인스턴스에 대한 효과가 완결된 트랜잭션 집합 S에 대한 완전한 직렬 스케줄의 효과와 동등함이 보장되는 스케줄
  - 주어진 스케줄을 수행해서 나온 데이터베이스 인스턴스는 해당 트랜잭션들을 직렬 순서대로 수행해서 나 온 데이터베이스 인스턴스와 동등
- 트랜잭션들을 똑같이 직렬로 수행하더라도 순서를 바꾸면 결과도 달라질 수 있음- 이 결과는 수용 가능함
- Abort된 트랜잭션이 있는 스케줄까지 포함하지는 않음

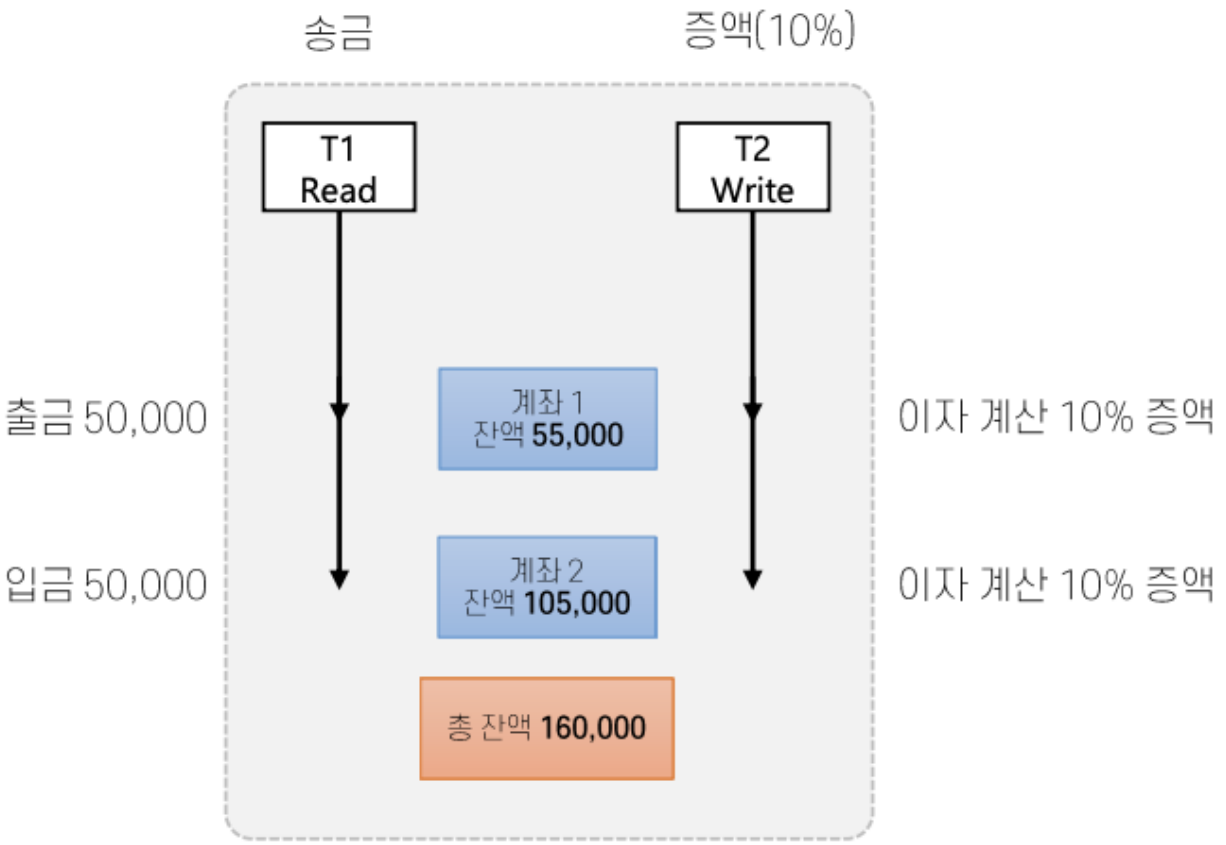
교차수행에 의한 일부 이상

- 트랜잭션 T1과 T2의 단위 작업이 충돌(conflict)하는 경우
    - 동일한 개체에 대해 두 개 이상의 단위 작업이 수행될 때 하나 이상의 트랜색션이 기록 연산일 경우
- 기록-판독 충돌(WR Conflict)
- 판독-기록 충돌(Rw Conflict)
- 기록-기록 충돌(ww Conflict)

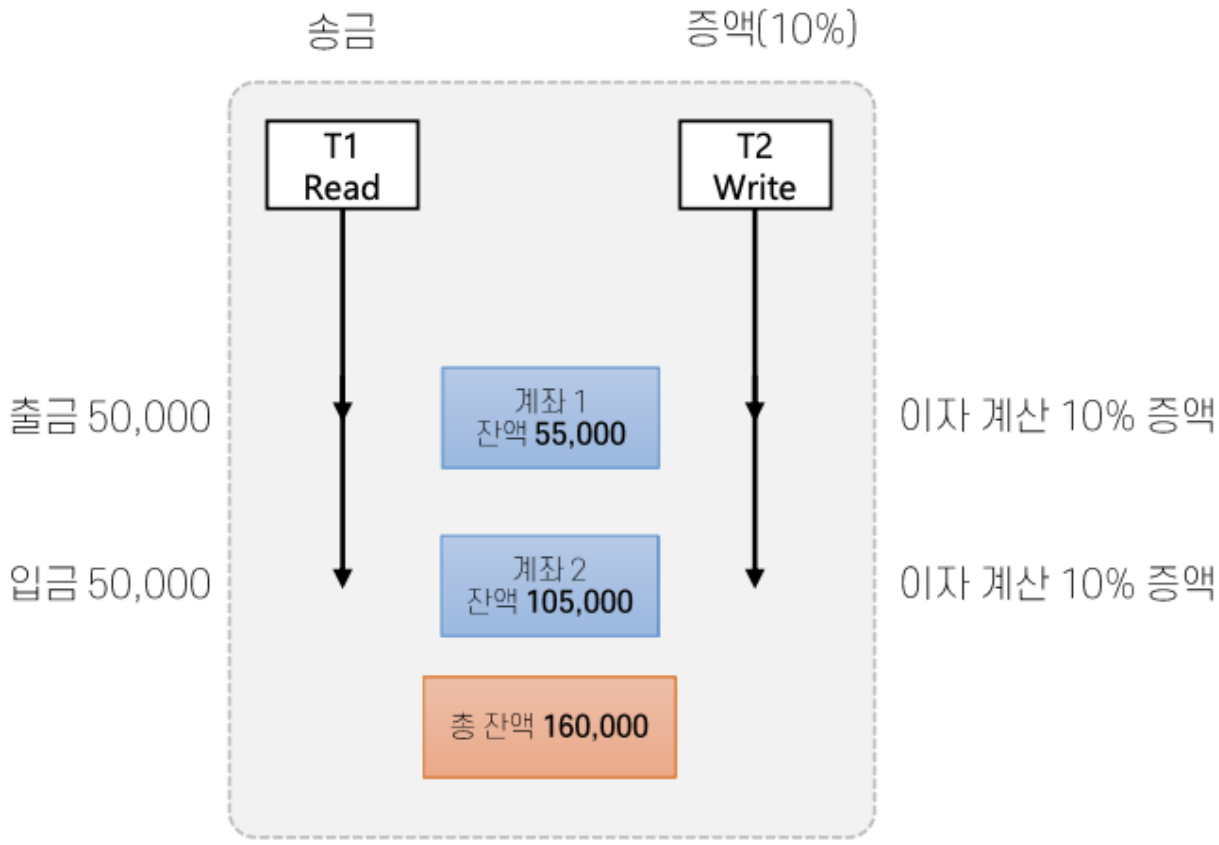
기록-판독 충돌(WR 충돌)

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

- 미 완결된 데이터를 읽는 경우
- 기록-판독 충돌(WR 충돌)
    - 트랜잭션 T이 수정한 데이터베이스 객체 A를, T10 Commit되기 전에 트랜잭션 T2가 판독(Read)하는 경우
    - 이러한 판독을 오손 판독(Dirty Read)라고 부름

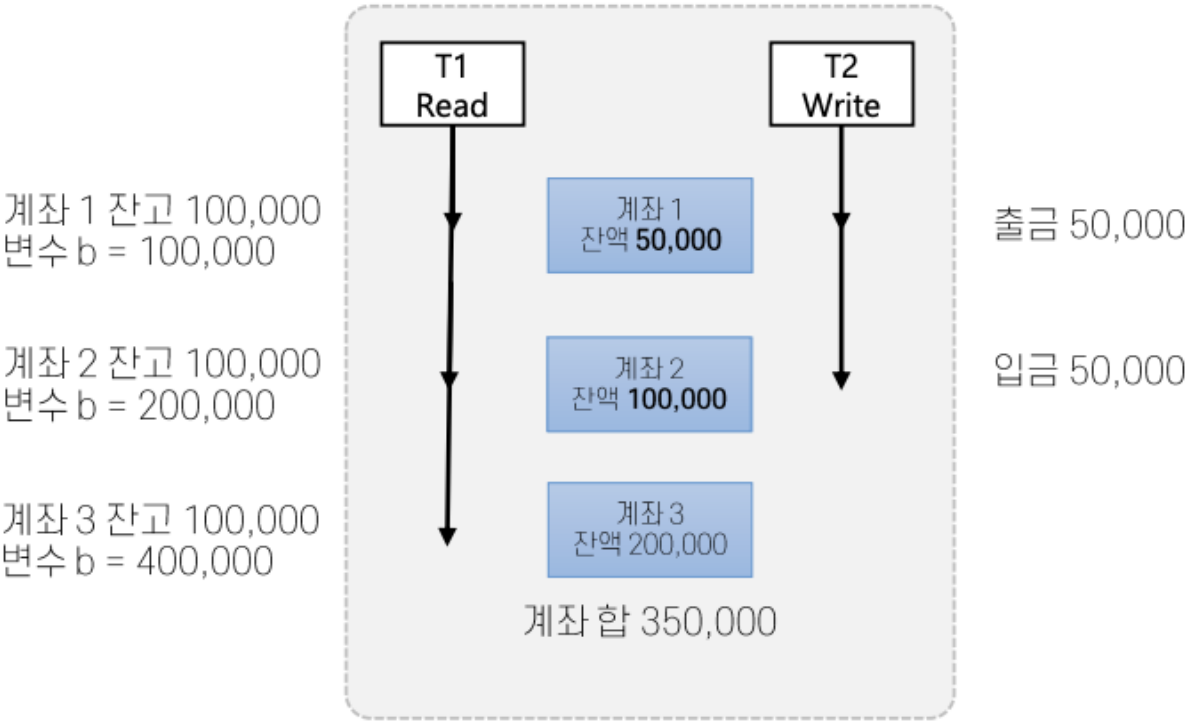

송금
증액(10%)
T1
Read
T2
Write
출금 50,000
입금 50,000
계좌 1
잔액 55,000
계좌 2
잔액 105,000
총 잔액 160,000
이자 계산 10% 증액
이자 계산 10% 증액

순차적 트랜잭션


송금
증액(10%)
T1
Read
T2
Write
출금 50,000
입금 50,000
계좌 1
잔액 55,000
계좌 2
잔액 105,000
총 잔액 160,000
이자 계산 10% 증액
이자 계산 10% 증액

판독-기록 충돌(RW 충돌)

- 반복 불가능한 읽기를 수행하는 경우
- 판독-기록 충돌(RW 충돌)
    - T이 객체 A의 값을 판독중에 T2가 객체 A의 값을 변경하는 경우
    - 이런 판독을 반복 불가능한 판독(Unrepeatable Read)라고 부름



기록-기록 충돌(WW 충돌)

- 미 완결된 데이터를 덮어 쓰는 경우
- 기록-기록 충돌(WW충돌)
- TOI 어떤 객체 A의 값을 수정하고, 진행 중인 상태에서 R2가 값을 덮어 쓰는 경우
- 이런 기록을 맹목 기록(Blind Write)라고 부름

Lock

- Lock Mode
- Shared Lock Mode
- Exclusive Lock Mode
- Strict 2 Phase Lock

Lock Mode

- 직렬성을 보장하기 위한 방법으로, 데이터 객체들이 상호 배타적으로 액세스 되도록 함
- 한 트랜잭션이 한 데이터 객체에 액세스 했을 때 다른 트랜잭션이 해당 객체를 수정하지 못하도록 함
- 다중 트랜잭션 환경에서 데이터베이스의 일관성과 무결성을 유지하기 위한 순차적 진행을 보장

- 잠금 모드
    - Shared Lock Mode(Read Lock)
        - 트랜잭션 T가 데이터 객체 A에 Shared Lock을 가지고 있다면 는 A를 읽을 수 있지만 갱신할 수 없음
    - Exclusive Lock ModeWrite Lock - X로 표시
        - 트랜색션 T가 데이터 객체 A에 Exclusive Lock을 가지고 있다면 는 A를 읽을 수도, 갱신할 수도 있음

Shared Lock Mode

- Shared Lock 또는 Read Lock- S로 표시함
- 다른 트랜잭션에서 데이터를 읽을 수 있는 LOCk Mode
    - Shared Lock은 다른 트랜잭션에 대한 읽기 시도를 허용
- 동일 데이터베이스 객체에 여러 Shared Lock이 적용될 수 있음
    - 트랜잭션 T1이 객체 A에 Shared Lock을 가지고 있으면, 트랜잭션 Ti에 대해 Shared Lock을 허용
    - 트랜색션 T1이 객체 A에 Shared Lock을 가지고 있으면, 트랜잭션 Ti에 대해 Exclusive Lock을 허용하지 않음
- SELECT 쿼리에 적용됨

Exclusive Lock Mode

- Write Lock - X로 표시함
- 다른 트랜잭션에서 데이터를 읽을 수도, 쓸 수도 없는 잠금 모드
    - Exclusive Lock은 다른 트랜잭션의 접근을 허용하지 않음
- 트랜잭션 T1이 객체 A에 Exclusive Lock을 가지고 있을 경우, Ti는 A에 서 Shared Lock을 가질 수 없음
- 트랜잭션 T1이 객체 A에 Exclusive Lock을 가지고 있을 경우, Ti는 A에 서 Exclusive Lock을 가질 수 없음
- UPDATE, DELETE, INSERT 등의 갱신 쿼리에 적용됨

Strict 2 Phase Lock

- 가장 널리 사용되는 잠금 규약으로, 두 가지 규칙을 사용
    1. 트랜잭션 T가 어떤 객체를 판독(수정)하려면, 그 객체에 대해 공유(배타적) 잠금을 요청
    2. 트랜잭션이 종료될 때 가지고 있던 모든 잠금을 풀어줌
- 잠금 규약은 안전한 인터리빙만을 이용함
    - 두 트랜잭션이 동일한 객체에 접근하며, 한 트랜잭션이 객체를 수정하려 할 때, 단위 작업이 직렬 순서로 수행한 호과를 얻도록 함

SQL의 트랜잭션 지원

잠금없는 병행제어

잠금 관리

- DBMS는 잠금 관리자(LoCk Manager)를 제공
    - 잠금 관리자는 잠금 테이블(Lock Table)과 트랜잭션 테이블(Transaction Table)을 유지
    - 잠금 테이블 엔트리(Lock Table Entry)로 객체에 대한 잠금 정보 관리
- 잠금 요청과 잠금 해제 요청 구현
    - Shared Lock을 요청하고, 요청 큐는 비어 있으며(객체에 대한 Lock을 가진 트랜잭션이 없음), 해당 객체가 현재 Exclusive로 LOck되어 있지 않은 경우에는, Lock Manager는 이 LOck을 허가하고 해당 객체에 대한 Lock Entry를 갱신

- Exclusive Lock을 요청하고, 현재 그 객체에 대한 Lock을 가진 트랜잭션이 없는 경우에는 Lock Manager는 이 Lock을 허가하고 Lock Table Entry를 갱신
- 이 밖의 경우에는 요청된 Lock이 바로 허가될 수 없으며, 해당 객체에 대한 Lock Request Queue에 요청을 추가

교착 상태

교착 상태 예방

교착상태 탐지