42 Seoul & CUBRID

06 트랜잭션과 잠금

Team 3
jinbekim jseo bypark

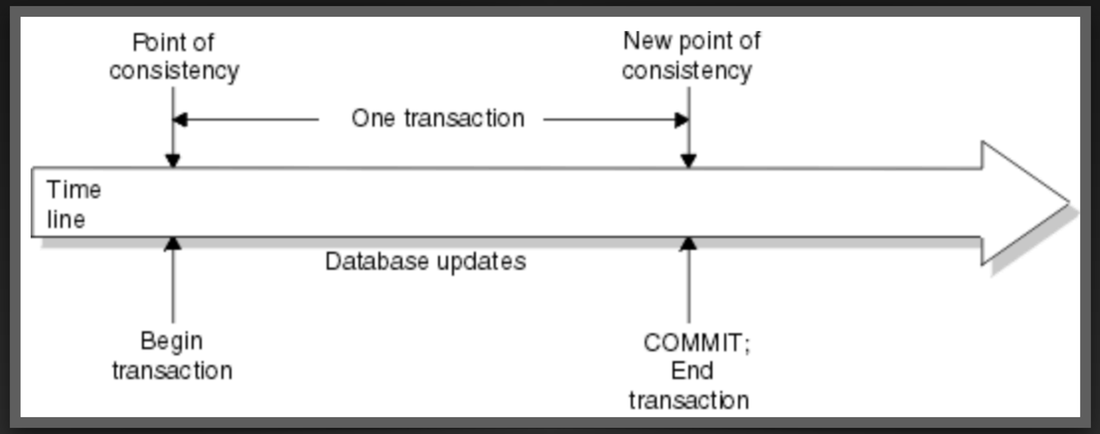
Transaction

→ 하나의 작업을 수행하기 위해 필요한 데이터베이스 연산들을 모아놓은 것

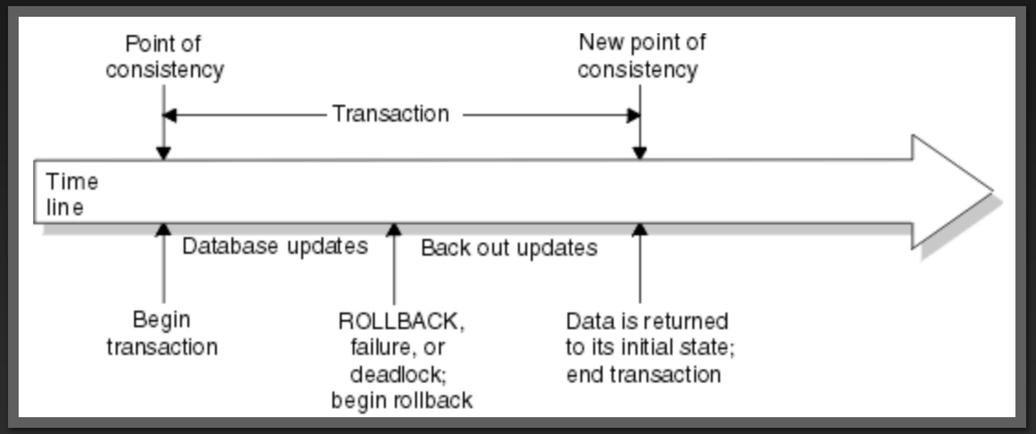
→ DB에서 논리적인 작업의 단위이며 장애가 발생했을 때 데이터를 복구하는 작업의 단위

Commit & Rollback

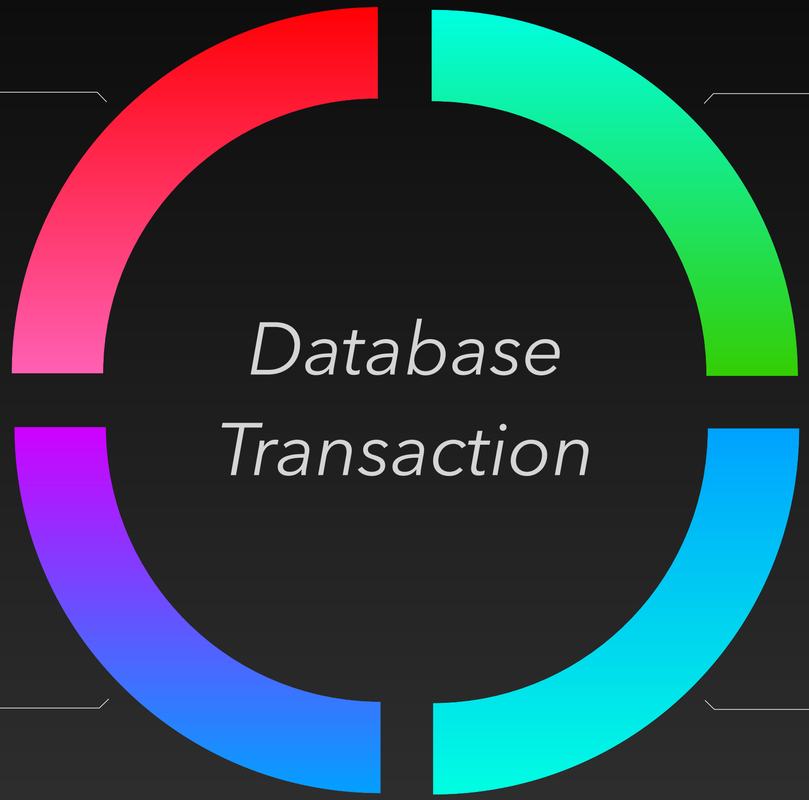
Commit



Rollback



ACID



Atomicity

Commit & Rollback

Durability

Recovery

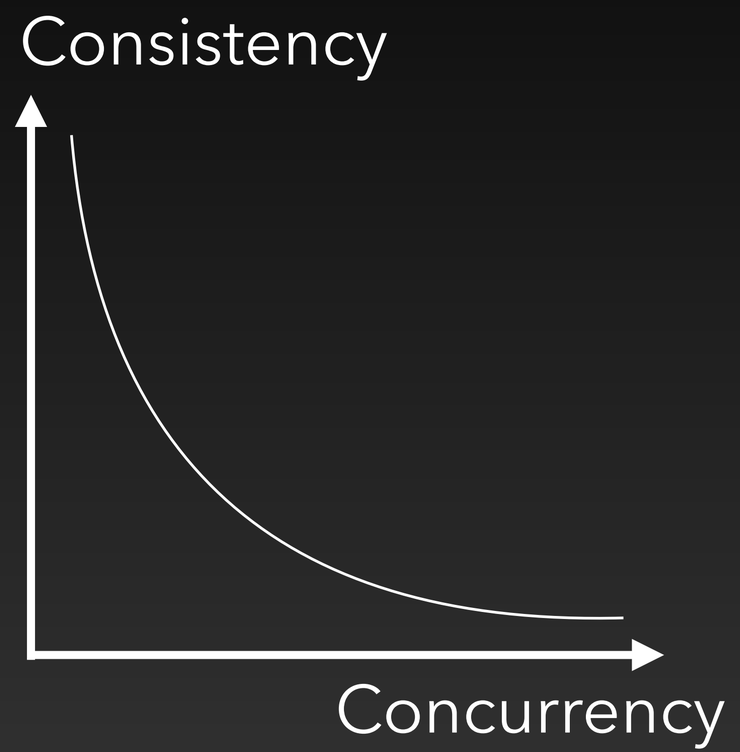
Consistency

Concurrency Control

Isolation

Isolation Level

Concurrency Control



Concurrency Control

→ 트랜잭션의 직렬성 보장, 동시 수행 트랜잭션 처리량 최대화

→ 공유도 최대, 응답 시간 최소, 시스템 활동의 최대 보장

→ 데이터의 무결성 및 일관성 보장

동시에 실행되는 트랜잭션 수를 최대화하면서 데이터의 무결성을 유지

Concurrency Control

Problem

- Lost Update
- Dirty Read
- Inconsistency
- Cascading Rollback

Concurrency Control

Problem

- Lost Update
- Dirty Read
- Inconsistency
- Cascading Rollback

→ 먼저 실행된 트랜잭션의 결과를 나중에 실행된 트랜잭션이 덮어쓸 때 발생하는 오류

Concurrency Control

Problem

Lost Update

Dirty Read

Inconsistency

Cascading Rollback

→ 트랜잭션의 중간 수행 결과를 다른 트랜잭션이 참조하여 발생하는 오류

Concurrency Control

Problem

- Lost Update
- Dirty Read
- Inconsistency
- Cascading Rollback

→ 두 트랙잭션이 동시에 실행되어 DB가 일관성이 결여되는 오류

Concurrency Control

Problem

Lost Update

Dirty Read

Inconsistency

Cascading Rollback

→ 복수의 트랜잭션이 데이터 공유 시
특정 트랜잭션이 처리를 취소할 경우
다른 트랜잭션이 처리한 부분을 취소하지 못하는 오류

Concurrency Control

구분	제어 기법	내용
Locking	Shared Lock	데이터 항목에 대해 읽기(read)만 가능
	Exclusive Lock	데이터 항목에 대해 읽기와 기록(입력/삭제)가 모두 불가능
2 Phase Locking		모든 트랜잭션들이 lock과 unlock연산을 확장 단계와 수축 단계로 구분하여 수행
Timestamp Ordering		DB시스템에 들어오는 트랜잭션 순서대로 System Clock/Logical Counter 할당하고 순서를 부여하여 동시성 제어의 기준으로 사용
Validation		트랜잭션 수행 동안은 어떠한 검사도 하지 않고, 트랜잭션 종료 시 일괄적 검사 기법
MVCC		트랜잭션의 타임스탬프와 접근 데이터의 여러 버전 타임스탬프를 비교하여 직렬 가능성이 보장되는 버전 선택

Concurrency Control

Locking

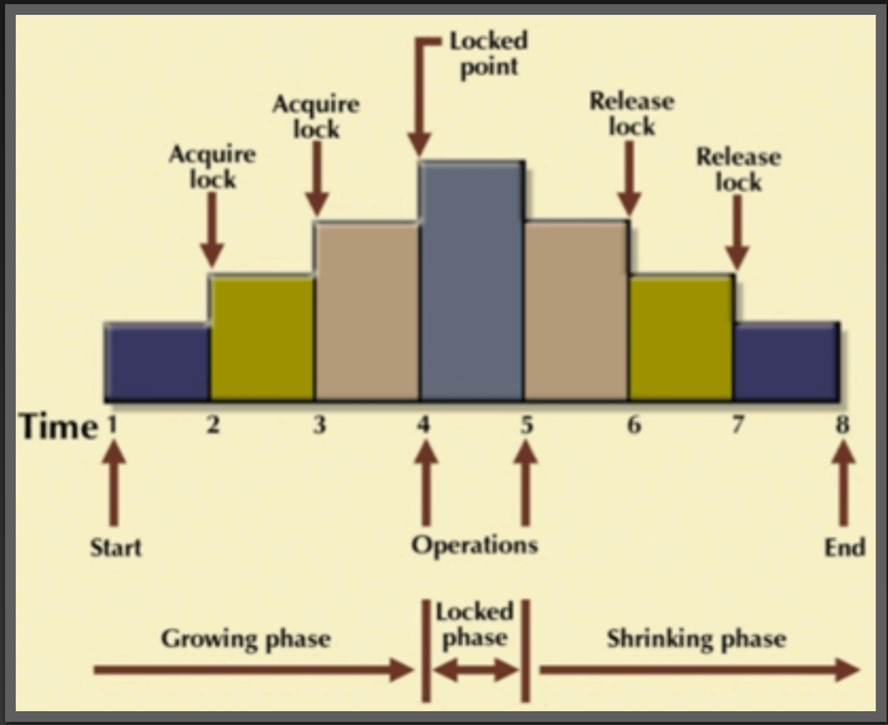
→ 병행 수행되는 트랜잭션들이 동일한 데이터에 접근하지 못하도록 lock과 unlock이라는 두개의 연산을 이용해 제어

Lock 연산 : 트랜잭션이 사용할 데이터에 대한 독점권을 가지기 위해 사용

Unlock 연산 : 트랜잭션이 데이터에 대한 독점권을 반납하기 위해 사용

Concurrency Control

2 Phase Locking



Lock 확장 단계 (Growing Phase - 1단계)
→ 새로운 Lock 연산만 수행 가능

Lock 수축 단계 (Shrinking Phase - 2단계)
→ Unlock 연산만 수행 가능

Concurrency Control

MVCC

→ Lock 을 사용하지 않고 데이터 읽기의 일관성을 보장해주는 방법

→ 데이터에 접근하는 사용자는 접근한 시점에서 데이터베이스의 Snapshot을 읽는다

1. 일반적인 RDBMS 보다 매우 빠르게 작동
2. 사용하지 않는 데이터가 계속 쌓이게 되므로 데이터를 정리하는 시스템 필요
3. 데이터 버전이 충돌하면 애플리케이션 영역에서 이러한 문제를 해결해야 함

Concurrency Control

MVCC

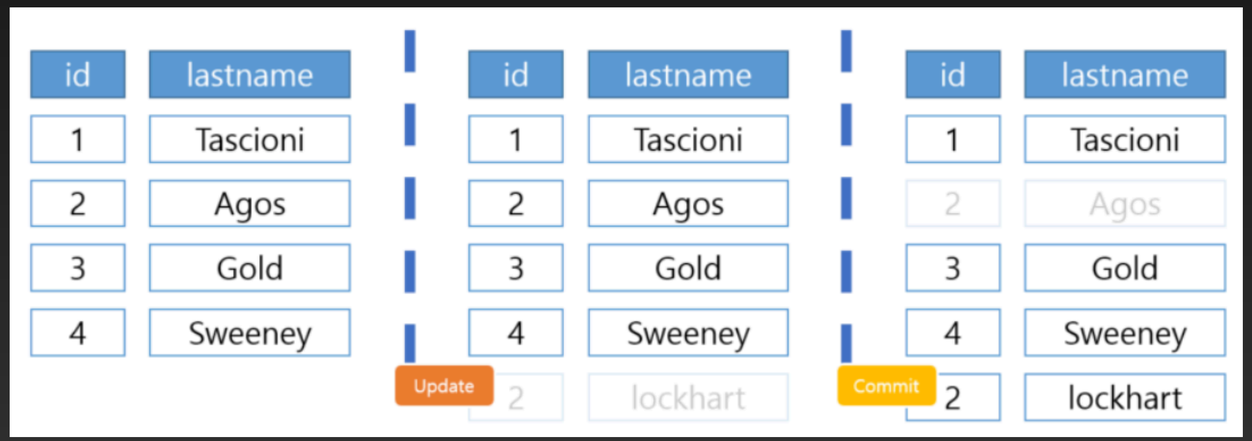
MGA (Multi Generation Architecture)

→ PostgreSQL, SQL Server DB

→ 데이터베이스 내에 다중 버전의 데이터 저장

→ 더 이상 필요하지 않을 때 레코드(데이터) 정리

→ Vacuum



Concurrency Control

MVCC

Rollback Segment

→ Oracle, MySQL DB

→ 최신 버전의 데이터만 데이터베이스 내에 저장

→ SCN(System Commit Number)