Real MySQL - 트랜잭션

잠금은 동시성을 제어하기 위한 기능이고 트랜잭션은 데이터의 정합성을 보장하기 위한 기능이다. 격리 수준이라는 것은 하나의 트랜잭션 내에서 또는 여러 트랜잭션 간의 작업 내용을 어떻게 공유하고 차단할 것인지를 결정하는 레벨을 의미한다.

트랜잭션

MyISAM은 트랜잭션을 지원하지 않기 때문에 쿼리가 부분적으로 커밋되는 Partial Update 현상이 일어날 수 있고, 이러한 현상은 테이블 데이터의 정합성으 맞추는데 상당히 어려운 문제를 만들어 낸다.

MySQL 엔진의 잠금

MySQL 엔진 레벨의 잠금은 모든 스토리지 엔진에 영향을 미친다.

글로벌 락

MySQL에서 제공하는 잠금 가운데 가장 범위가 크다. 글로벌 락이 영향을 미치는 범위는 MySQL 서버 전체이며, 작업 대상 테이블이나 데이터베이스가 다르더라도 동일하게 영향을 미친다. 주로 MyISAM이나 MEMORY 데이터베이스에 존재하는 테이블에 대해 일관된 백업을 받아야 할 때 사용된다.

특정 세션에서 백업 락을 획득하면 모든 세션에서 다음과 같이 테이블의 스키마나 사용자의 인증 관련 정보를 변경할 수 없게 된다.

- 데이터베이스 및 테이블 등 모든 객체 생성 및 변경, 삭제
- REPAIR TABLE과 OPTIMIZE TABLE 명령
- 사용자 관리 및 비밀번호 변경

하지만 백업 락은 일반적인 테이블의 데이터 변경은 허용된다.

MySQL 서버의 구성은 소스 서버와 레플리카 서버로 구성되는데, 백업은 레플리카 서버에서 실행된다.

테이블 락

테이블락은 명시적으로 획득할 수 있으며, 묵시적인 테이블 락은 테이블에 데이터를 변경하는 쿼리를 실행하면 발생한다. 하지만 InnoDB 테이블의 경우 스토리지 엔진 차원에서 레코드 기반의 잠금을 제공하기 때문에 단순 데이터 변경 쿼리로 인해 묵시적인 테이블 락이 설정되지는 않는다.

네임드 락 (분산 락)

네임드 락은 임의의 문자열에 대해 잠금을 설정할 수 있다, 즉 대상이 테이블이나 레코드 또는 AUTO_INCREMENT와 같은 데이터베이스 객체가 아니라는 것이다. 네임드 락은 단순히 사용자가 지정한 문자열에 대해 획득하고 반납하는 잠금이다.

메타데이터 락

데이터베이스 객체의 이름이나 구조를 변경하는 경우에 획득하는 잠금이다. 메타데이터 락은 명시적으로 획득하거나 해제할 수 있는 것이 아니고 테이블의 이름을 변경하는 경우 자동으로 획득하는 잠금이다.

테이블의 구조를 변경해야 할 때 Online DDL을 이용할 수도 있지만, 시간이 너무 오래 걸리는 경우라면 새로운 구조의 테이블을 생성하고 여러 개의 스레드로 빠르게 복사하는 것이 좋다. Online DDL은 단일 스레드로 작동하기 때문에 상당히 많은 시간이 소모되기 때문이다.

InnoDB 스토리지 엔진 잠금

InnoDB 스토리지 엔진은 MySQL에서 제공하는 잠금과는 별개로 스토리지 엔진 내부에서 레코드 기반의 잠금 방식을 탑재하고 있다.

InnoDB 스토리지 엔진은 레코드 기반의 잠금 기능을 제공하며, 잠금 정보가 상당히 작은 공간으로 관리되기 때문에 레코드 락이 페이지 락으로, 또는 테이블 락으로 escalate 되는 경우는 없다.

레코드 락

레코드 자체만을 잠그는 것을 레코드 락이라고 하며, InnoDB 스토리지 엔진은 레코드 자체가 아니라 인덱스의 레코드를 잠근다. 인덱스가 하나도 없는 테이블이더라도 내부적으로 자동 생성된 클러스터 인덱스를 이용해 잠금을 설정한다.

갭 락

레코드 자체가 아니라 레코드와 바로 인접한 레코드 사이의 간격만을 잠그는 것이다. 레코드와 레코드 사이의 간격에 새로운 레코드가 생성되는 것을 제어한다.

넥스트 키 락

레코드 락과 갭 락을 합쳐 놓은 형태의 잠금을 넥스트 키 락이라고 한다. InnoDB의 갭 락이나 넥스트 키 락은 바이너리 로그에 기록되는 쿼리가 레플리카 서버에서 실행될 때 소스 서버에서 만들어 낸 결과와 동일한 결과를 만들어내도록 보장하는 것이 주목적이다. 레플리카 서버에서 소스 서버에게 받은 바이너리 로그를 저장해둔 파일을 릴레이 로그라고 한다.

자동 증가 락

AUTO_INCREMENT 칼럼이 사용된 테이블에 동시에 여러 레코드가 INSERT 되는 경우, 저장되는 각 레코드는 중복되지 않고 저장된 순서대로 증가하는 일련번호 값을 가져야 한다. 이를 위해 AUTO_INCREMENT 락이라고 하는 테이블 수준의 잠금이 사용된다. INSERT와 REPLACE 쿼리 문장과 같이 새로운 레코드를 저장하는 쿼리에서만 필요하며, UPDATE나 DELETE 등의 쿼리에서는 걸리지 않는다.

다른 잠금과는 달리 트랜잭션과 관계없이 INSERT나 REPLACE 문장에서 AUTO_INCREMENT 값을 가져오는 순간만 락이 걸렸다가 즉시 해제된다.

8.0 부터 innodb_autoinc_lock_mode의 기본값이 2로 바뀌었다. 이 설정은 자동 증가 락을 걸지 않고 경량화된 뮤텍스를 사용한다. 하지만 이 설정에서는 하나의 INSERT 문장으로 삽입되는 레코드라고 하더라도 연속된 자동 증가 값을 보장하지는 않는다. 그래서 이 설정 모드를 인터리빙 모드(interleaved mode)라고도 한다.

인덱스와 잠금

InnoDB의 잠금은 레코드를 잠그는 것이 아니라 인덱스를 잠그는 방식으로 처리된다. 즉, 변경해야 할 레코드를 차기 위해 검색한 인덱스의 레코드를 모두 락을 걸어야 한다.

MySQL의 격리 수준

트랜잭션의 격리 수준이란 여러 트랜잭션이 동시에 처리될 때 특정 트랜잭션이 다른 트랜잭션에서 변경하거나 조회하는 데이터를 볼 수 있게 허용할지 말지를 결정하는 것이다.

격리 수준은 크게

- READ UNCOMMITTED
- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE

의 4가지로 나뉜다.

4개의 격리 수준에서 순서대로 뒤로 갈수록 각 트랜잭션 간의 데이터 격리 정도가 높아지며, 동시 처리 성능도 떨어진다. 하지만 SERIALIZABLE 격리 수준이 아니라면 크게 성능의 개선이나 저하는 발생하지 않는다.

데이터베이스의 격리 수준을 이야기하면 항상 함께 언급되는 세 가지 부정합의 문제점이 있다.

- DIRTY READ: 어떤 트랜잭션에서 처리한 작업이 완료되지 않았는데도 다른 트랜잭션에서 볼 수 있는 현상
- NON-REPEATABLE READ: 하나의 트랜잭션 내에서 똑같은 SELECT 쿼리를 실행했을 때는 항상 같은 결과를 가져와야 하는 REPEATABLE READ 정합성을 만족하지 못하는 경우.
- PHANTOM READ: 다른 트랜잭션에서 수행한 변경 작업에 의해 레코드가 보였다 안 보였다 하는 현상.