4.2 MySQL 엔진의 잠금

생성일	@2021년 5월 30일 오후 10:26
태그	

MySQL에서 사용되는 잠금

- 스토리지 엔진 레벨의 잠금
  - 스토리지 엔진 간 상호 영향을 미치지 않음
- MySQL 엔진 레벨의 잠금
  - 모든 스토리지 엔진에 영향을 미침

1. 글로벌 락

```
FLUSH TABLES WITH READ LOCK
```

- 글로벌 락은 위 명령으로만 획득 가능
- MySQL에서 제공하는 잠금 중 가장 범위가 크다. MySQL 서버 전체에 영향을 미침
- 한 세션에서 글로벌 락을 획득시, 다른 섹션 대부분의 DDL, DML이 대기 상태로 남음

2. 테이블 락

```
# 명시적 락 획득
LOCK TABLES table_name [READ | WRITE]
# 잠금 반남
UNLOCK TABLES
```

- 개별 테이블 단위로 설정되는 잠금 (명시적, 묵시적으로 획득 가능)
  - 명시적인 테이블 락은 특별한 상황이 아니면 애플리케이션에서 거의 사용할 필요 X (글로벌 락처럼 온라인 작업에 큰 영향 미침)
- 묵시적 테이블 락 : 쿼리가 실행되는 동안 자동적으로 획득 → 완료 후 자동 해제

- InnoDB의 경우 스토리지 엔진 차원에서 레코드 기반 잠금 제공 → 단순 데이터 변경 쿼리로 묵시적 테이블 락 설정 안됨
- InnoDB 테이블도 테이블 락 설정은 되지만 대부분 DML 쿼리에서는 무시되고 DDL에만 영향을 미침

3. 유저 락

```
GET_LOCK()
```

- 위 함수로 임의 잠금 설정 가능
- 잠금 대상이 테이블/레코드/AUTO_INCREMENT 같은 DB 객체가 아님
- 사용자가 지정한 String에 획득하고 반납하는 잠금임 (자주 사용되지는 않음)
  - 동시에 다수의 웹 서버가 정보 동기화를 해야하는 경우, 유저 락을 사용

```
-- // "mylock"이라는 문자열에 대해 잠금을 획득한다.
-- // 이미 잠금이 사용 중이면 2초 동안만 대기한다.
mysql> SELECT GET_LOCK('mylock', 2);
```

  - 많은 레코드를 한번에 변경하는 트랜잭션의 경우 (배치 프로그램) 데드락을 최소화 할 수 있음

4. 네임 락

```
RENAME TABLE tab_a TO tab_b
```

:DB 객체(테이블, 뷰)의 이름을 변경하는 경우 획득하는 잠금

- 명시적으로 획득/해제 불가능 → 테이블 이름 변경시 자동으로 획득하는 잠금
- 위 예시의 경우 원본 이름과 변경되는 이름 모두 한번에 잠금 설정

```
mysql> RENAME TABLE rank TO rank_backup;
mysql> RENAME TABLE rank_new TO rank;
```

→ 이 경우 잠깐이지만 table not found 'rank' 같은 에러가 뜰 수 있음

```
mysql> RENAME TABLE rank TO rank_backup , rank_new TO rank;
```

→ 요렇게 한 명령에 두개 RENAME을 하면 에러 안뜨고 적용 가능