Mysql

table drop 하려하는데 waiting for mata data lock 이라는 메세지만 있고 진행이안될때..

해결책 : <https://dba.stackexchange.com/questions/41141/unable-to-drop-the-table-since-its-showing-waiting-for-meta-data-lock>

<https://stackoverflow.com/questions/37306568/how-to-solve-mysql-innodb-waiting-for-table-metadata-lock-on-truncate-table>

=> SHOW ENGINE INNODB STATUS 해당 command를 던져서 내용을 확인해봐야함… idle(sleep)되어있는 프로세스가 있는데(정상인거같이보임..), 이를 죽여야지 drop table 이 가능..

위의 설명에 의하면 코드로인한 버그같다고하는데.. 일단 그렇다함…

야매해결책은, table이름 변경하고 truncate 하고, drop table하면 됨

No operations allowed after connection closed 오류 해결

-일정시간 mysql접속안해서 생기는문제.. 아래 사이트에서 해결

<https://do-dam.tistory.com/50>

max connection 될때 확인 & 수정해야할것

**show variables like '%max\_connect%'**;  
**show variables like 'wait\_timeout'**;  
  
**set global** max\_connections=1000;  
**set** wait\_timeout =60;  
  
**show status like '%CONNECT%'**;  
  
**show status like 'Threads\_connected'**;

datetime 자료형 빠른 검색 : <https://devhood.tistory.com/249>

Procedure

사용자정의변수(@variablr)의 scope는 session이다.. 다른 session에서는 접근할수없다.. 또한 session에서 작업이 이루어지는것은 하나의 스레드에서 이루어진다.. 즉 동시성문제는 일어나지않는다.. 다른 세션을 사용하면 같은 작업을 동시에할수있으나 사용자정의변수의 scope가 session 이므로 해당사항안댐!

Procedure 기본 작성방법 : <https://primary924.tistory.com/47>

LOCK

MySQL InnoDB 엔진의 경우 상황에 따른 여러가지 락(Lock)을 통해서 동시성을 제어하며 이를 통해서 사용자가 설정한 대로 원하는 수준의 트랜잭션 격리 수준(Transaction Isolation level)을 유지해준다.

innoDB는 locking read 를 할수있도록 두가지 쿼리 제공

-select .. lock in share mode (select할때 다른 세션 접근가능,, 수정은안댐)

-select .. for update (모든 접근 불가.. 배타락.. 읽지도못함

예) 특정 필드의 값을 읽어온 후 (SELECT), 해당 값을 1 증가시켜서 저장 (UPDATE)하고 싶을때 이 경우 SELECT .. FOR UPDATE를 사용하지 않으면, 여러 트랜잭션이 동시에 접근할 경우 정상 동작 하지 않을 수 있다. 즉, 트랜잭션 2개가 실행되었는데 값이 2만큼 증가되지 않고 1만 증가되어있을 가능성 존재.)

락 사용이유

:여러 트랜잭션이 경합하고잇을때 최대한의 성능을 위해서 여러방식의 락을 조합해서 사용함..

락의 종류

-Shared lock(s) : row level(행에 관한 lock) 이며 select를 위한 read lock임.. 다른 트랜잭션이 해당 Row에대해 X lock은 획득할수 없지만 S lock은 획득가능,, 즉, 다른놈들도 읽을수있다..

-Exclusive lock(x) : row level이며 update, delete를 위한 write lock임.. x lock 이 걸려있으면 다른 트랜잭션이 x, s lock 모두 획득 불가.. 대기!

-intention lock : table level이며 row에 대해서 나중에 row level의 lock을 걸것을 미리 알려주는 lock

Ex) select .. lock in share mode 가 실행되면 먼저 intetion shared lock(IS) 이 테이블에 걸림 그리고 rowlevel의 s lock

Slelct .. for update, insert, delete, update가 실행되면 먼저 intention exclusive lock(ix)가 테이블에 걸림 그리고 row level의 x lock

-IS, IX는 여러 트랜잭션에서 동시에 접근가능.. 하지만 동일한 row에 lock을 획득하는 과정에서 동시 접근을 막거나 허용하는 제어를 하게됨

- lock tables, alter talbe, drop table이 실행될때는 is, ix 모두 block하는 table-level 락이 걸림.. 즉 ,is, ix는 대기하게된다.. 반대도 마찬가지!

아래는 Share lock, Exclusive lock, Intention lock이 각각 다른 트랜잭션에서 사용될때, 충돌(Conflict, 대기상태 빠짐), 또는 호환(Compatible, 대기상태에 빠지지 않음)이 되는지에 대해 정리된 표이다.

|  | **X** | **IX** | **S** | **IS** |
| --- | --- | --- | --- | --- |
| X | Conflict | Conflict | Conflict | Conflict |
| IX | Conflict | Compatible | Conflict | Compatible |
| S | Conflict | Conflict | Compatible | Compatible |
| IS | Conflict | Compatible | Compatible | Compatible |

Record lock : pk, unique index로 조회해서 하나의 인덱스 레코드 에만 lock을 거는것.. 즉, 인덱스 레코드에 대해 락을 거는것을 의미!

Gap lock(=range lcok) : 실제 존재하는 인덱스 레코드에 락을 거는것이 아니고 범위를 지정하기 위해 인덱스 레코드 사이의 범위(gap)에 락을 거는 것을 의미한다.

컬럼에 index가 걸려있지 않거나, index가 걸려있어도 unique 하지 않다면 gap lock이 필요하다.

-index가 걸려있다면 row를 찾기위해 스캔했던 index range에 대해서 gap lock 적용

-index가 걸려있지 않다면 결국 테이블 전체를 스캔해야 되기때문에 모든 row에대해 lock이 걸림.

Next-key lock : 범위를 지정한 쿼리를 실행하면, gap lock + record lock 이 복합적으로 사용된다..

Ex) select \* where pk > 99 for update

-Pk가 99보다 큰거를 찾기 직전의 레코드 부터 처음 발견한 record 까지 gap lock 적용

-Pk가 99보다 큰 모든 인덱스 레코드들의 사이에도 gap lock 적용

-pk가 99인 모든 인덱스 레코드들에 각각 record lock 적용

=> 이렇게 복합적으로 나타나는것을 next-key lock 이라함

Insert Intention lock : insert 구문이 실행될때 innodb에서 획득하는 특수한 형태의 Gap lock..

예) pk=3, pk=6의 레코드가 존재하는 테이블이 존재

A 트랜잭션에서 pk=5에 INSERT, B 트랜잭션에서 pk=4에 INSERT시도

-만약 일반적인 gap lock 사용한다면:

-A트랜잭션이 pk=5를 INSERT하는 과정에서 pk=3~5에는 gap lock 걸림

-B트랜잭션이 pk=4에 INSERT 시도시 pk=3~5에 gap lock이 걸려있기때문에 A가 트랜잭션이 완전히 -종료될 때 까지 기다려야 한다.

-대기시간 존재!

-Insert Intention Lock 사용시:

-A트랜잭션이 pk=5를 INSERT하는 과정에서 pk=3~5에는 insert intention lock 걸림

-B트랜잭션이 pk=4에 INSERT 시도시 pk=3~5에 insert intention lock이 걸려있더라도 pk가 겹치지 않기때문에 바로 진행 가능

-대기시간 없음!

-실제 InnoDB의 동작 방식

데드락 상태 확인 명령어

어떤 lock이 걸려있는지 상태를 확인하거나 최근에 발생했던 데드락에 대해 알고 싶으면 아래 명령어를 사용하면 된다.

“SHOW ENGINE INNODB STATUS;”

데드락 케이스

-upsert시 batch로 이루어질때..

A session : insert into table(pk, name) values(1,’a’), (2,’b’) on duplicate key update name=VALUES(name)

B session : insert into table(pk, name) values(2,’a’), (1,’b’) on duplicate key update name=VALUES(name)

=> A upsert가 pk1번을 x락과 동시에 B upsert가 pk2번을 x락..

=>A session은 pk2번 기다림, B session은 pk1번 기다림..

=>데드락..

-3개 이상의 세션에서 모두 insert를 하려하는데 제일 처음 시도하여 배타락을 획득했던 세션이 duplicate key error 가 발생하였을때..(원래 Insert를 하면 x lock 이 생기는데, 해당 에러가 발생하면 대기중이엿던 session이 shared lock을 먼저 가지게된다..)

=> insert를 하기위해 대기중이였던 두 세션 모두가 shared lock을 획득..

=> 두 세션 모두 배타락을 가지려하지만 shared lock을 가지고있어서 어느누구도 배타락을 획득하지못함..

=> 데드락..

위의 내용 출처 : https://www.letmecompile.com/mysql-innodb-lock-deadlock/

괜찮은 정리 : https://idea-sketch.tistory.com/m/47

잠금관련해서 예시 굿 : https://blog.appkr.dev/cheatsheet/understanding-mysql-db-lock/

트랜잭션 격리 수준 정리 : <https://www.letmecompile.com/database-transaction-isolation-level/>

Isolation과 트랜잭션 정리 굿(예시굿) : <https://suhwan.dev/2019/06/09/transaction-isolation-level-and-lock/>

데드락 확인해볼수있는 실제적인 쿼리들과 격리수준에 따른 차이점 정리 굿 : <http://labs.brandi.co.kr/2019/06/19/hansj.html>

Charset 과 collation

Charset은 텍스트를 저장하기위한 자료형을 나타냄.. ex) utf-8

- Mysql은 Emoji 문자열(Ex. 이모티콘.. 4바이트문자열임) 처리를 위해 utf8mb4가 추가.. 이것은 4바이트문자열까지 가능하도록 하기위해서 설계한것인데, mysql이 기존 utf-8을 3바이트 가변자료형으로 설계하여서 추가되었음

Collation은 정렬을 이야기함.. charset을 어떻게 정렬할것인가!? (utf\_bin, utf8\_general\_ci, utf8\_unicode\_ci)

Ex) a와 b중 무엇이 더 큰가..

A와 a중 무엇이 더 큰가 등

<https://sshkim.tistory.com/128>