**SQL优化（系列）- 增删改语句优化**

**一、优化原则**

     SQL 中的增删改（DML）语句，优化是非常重要的，可以*提高查询的性能和效率*。以下是常见的 DML 语句优化技巧：

**1.1、减少影响行数**

*尽量避免不必要的数据更新，只更新必要的数据。*

    使用 WHERE 子句*限制更新或删除的行数，避免影响整个表。*

**1.2、批量操作**

    将*多个更新或删除操作合并为一个批量操作，减少数据库的调用次数*。

**1.3、使用事务**

    在执行多个 DML 操作时，尽量将它们放入一个事务中，*确保数据的完整性。*

**1.4、索引优化**

*在更新或删除操作中，尽量使用索引来定位要影响的行，减少全表扫描的开销。*

**1.5、避免触发器的过度使用**

    触发器可以导致数据更新的级联操作，可能影响性能，尽量避免过多的触发器。

**1.6、避免表锁**

    在更新或删除数据时，*尽量避免锁定整个表*，可以使用适当的事务隔离级别来控制锁定范围。

**1.7、合理使用索引**

    对*经常被更新或删除的列适量考虑是否需要添加索引*，同时要避免过多的索引导致更新或删除操作变慢。

**1.8、分批次处理**

*当需要删除大量数据时，考虑分批次进行删除操作，以避免一次性删除过多数据而导致性能问题。*

    具体的优化策略会根据数据库系统的特点、表结构和数据量等因素而有所不同，建议根据实际情况选择合适的优化方法。

**二、实践场景**

**2.1、大批量插入数据**

    如果同时执行大量的插入，建议使用多个值的INSERT语句(方法二)。这比使用分开INSERT语句快（方法一），一般情况下批量插入效率有几倍的差别。

方法一：

insert into T values(1,2); insert into T values(1,3); insert into T values(1,4);

方法二：

Insert into T values(1,2),(1,3),(1,4);

选择后方法二的原因：

* *减少SQL语句解析的操作，MySQL没有类似Oracle的share pool，采用方法二，只需要解析一次就能进行数据的插入操作；*
* *在特定场景可以减少对DB连接次数*
* *SQL语句较短，可以减少网络传输的IO。*

**2.2、适当使用commit**

    适当使用commit可以释放事务占用的资源而减少消耗，commit后能释放的资源如下：

* 事务占用的undo数据块；
* 事务在redo log中记录的数据块；
* 释放事务施加的，减少锁争用影响性能。特别是在需要使用delete删除大量数据的时候，必须分解删除量并定期commit。

**2.3、避免重复查询更新的数据**

    针对业务中经常出现的更新行同时又希望获得改行信息的需求，MySQL并不支持PostgreSQL那样的UPDATE RETURNING语法，在MySQL中可以通过变量实现。

例如，更新一行记录的时间戳，同时希望查询当前记录中存放的时间戳是什么，简单方法实现：

Update t1 set time=now() where col1=1; Select time from t1 where id =1;

使用变量，可以重写为以下方式：

Update t1 set time=now () where col1=1 and @now: = now ();   
Select @now;

前后二者都需要两次网络来回，但使用变量避免了再次访问数据表，特别是当t1表数据量较大时，后者比前者快很多。

**2.4、查询优先还是更新（insert、update、delete）优先**

    我们首先应该确定应用的类型，判断应用是以查询为主还是以更新为主的，是确保查询效率还是确保更新的效率，决定是查询优先还是更新优先。

*1）写入操作优先于读取操作。*

*2）对某张数据表的写入操作某一时刻只能发生一次，写入请求按照它们到达的次序来处理。*

*3）对某张数据表的多个读取操作可以同时地进行。MySQL 提供了几个语句调节符，允许你修改它的调度策略：*

* *LOW\_PRIORITY关键字应用于DELETE、INSERT、LOAD DATA、REPLACE和UPDATE；*
* *HIGH\_PRIORITY关键字应用于SELECT和INSERT语句；*
* *DELAYED关键字应用于INSERT和REPLACE语句。*