# 概述

# AUTO\_INCREMENT

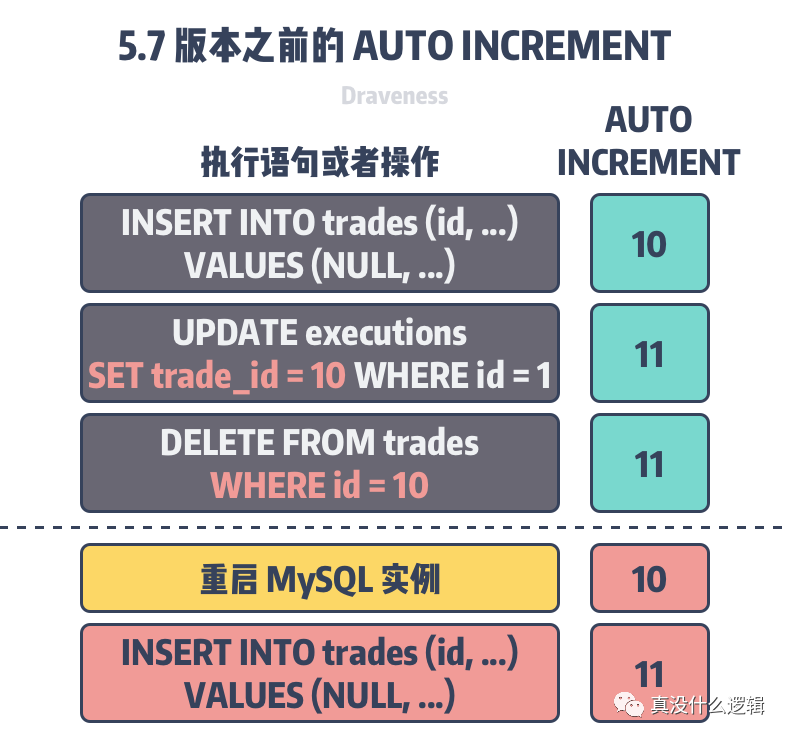
AUTO\_INCREMENT属性虽然在MySQL中十分常见，但是在较早的MySQL版本中，它的实现还比较简陋，InnoDB引擎会在内存中存储一个整数表示下一个被分配到的ID，当客户端向表中插入数据时会获取 AUTO\_INCREMENT 值并将其加一。



因为该值存储在内存中，所以在每次MySQL实例重新启动后，当客户端第一次向table\_name表中插入记录时，MySQL会使用如下所示的SQL语句查找当前表中id 的最大值，将其加一后作为待插入记录的主键，并作为当前表中AUTO\_INCREMENT计数器的初始值。

SELECT MAX(ai\_col) FROM table\_name FOR UPDATE;

如果让作者实现 AUTO\_INCREMENT，在最开始也会使用这种方法。不过这种实现虽然非常简单，但是如果使用者不严格遵循关系型数据库的设计规范，就会出现如下所示的数据不一致的问题：



因为重启了MySQL的实例，所以内存中的 AUTO\_INCREMENT 计数器会被重置成表中的最大值，当我们再向表中插入新的 trades 记录时会重新使用10作为主键，主键也就不是单调的了。在新的trades记录插入之后，executions表中的记录就错误的引用了新的trades，这其实是一个比较严重的错误。

然而这也不完全是MySQL的问题，如果我们严格遵循关系型数据库的设计规范，使用外键处理不同表之间的联系，就可以避免上述问题，因为当前trades记录仍然有外部的引用，所以外键会禁止trades记录的删除，不过多数公司内部的DBA都不推荐或者禁止使用外键，所以确实存在出现这种问题的可能。

然而在MySQL 8.0中，AUTO\_INCREMENT计数器的初始化行为发生了改变，每次计数器的变化都会写入到系统的重做日志（Redo log）并在每个检查点存储在引擎私有的系统表中。

In MySQL 8.0, this behavior is changed. The current maximum auto-increment counter value is written to the redo log each time it changes and is saved to an engine-private system table on each checkpoint. These changes make the current maximum auto-increment counter value persistent across server restarts.

当MySQL服务被重启或者处于崩溃恢复时，它可以从持久化的检查点和重做日志中恢复出最新的AUTO\_INCREMENT计数器，避免出现不单调的主键也解决了这里提到的问题。

# 操作