**索引**

MySQL索引总结（一 ~ 四）

<http://www.cnblogs.com/crazylqy/p/7615388.html>

# MySQL中的聚集索引和辅助索引知识详解

<https://blog.csdn.net/w892824196/article/details/82661446>

主索引 = 聚簇索引

辅助索引 = 二级索引 = 非聚簇索引

聚簇索引和非聚簇索引都是B+索引

主键 ≠ 聚簇索引

如果一张表定义了主键，那么主键作为聚集索引

如果没有定义主键，那么该表的第一个唯一非空索引被作为聚集索引

如果没有主键也没有唯一非空索引，那么innodb内部会生成一个隐藏的6字节的主键作为聚集索引，改列的值会随着数据的插入自增

除聚簇索引外的索引都是辅助索引

唯一约束就是唯一索引。唯一索引不允许具有索引值相同的行，从而禁止重复的索引或键值。系统在创建该索引时检查是否有重复的键值，并在每次使用 INSERT 或 UPDATE 语句添加数据时进行检查。

**MVCC**

MySql --MVCC实现原理（乐观锁实现原理）

<https://blog.csdn.net/ai_xiangjuan/article/details/78568337>

**快照读&当前读**

**快照读：**快照读读取的是快照版本，也就是历史版本。在事务中使用普通的select语句使用的就是快照读（select \* from table ...）。快照读也就是MVCC的select规则，即事务只能读取创建版本号早于当前事务版本号（包括等于）的行，事务读取的行要么是事务开始前已经提交的，要么是事务自身插入或修改的。行的删除版本号，要么未定义，要么大于当前事务版本号。这可以确保事务读取到的行在事务开始之前未被删除。

**当前读：**读取的是最新的数据，也就是忽略版本号的约束表中真正的数据。

在以下情况会使用当前读：

select \* from table where ? lock in share mode;

select \* from table where ? for update;

insert;

update;

delete;

其中第一个语句会加S锁，其他语句均加X锁（只有select语句才加lock in share mode 或 for update，insert、update和delete不加）

<https://www.cnblogs.com/cat-and-water/p/6427612.html>

SELECT ... LOCK IN SHARE MODE 行共享锁（事务结束自动释放）

SELECT ... FOR UPDATE 行排它锁（事务结束自动释放）

LOCK TABLE tb\_name READ 表共享锁（UNLOCK TABLES释放）

LOCK TABLE tb\_name WRITE 表排它锁（UNLOCK TABLES释放）

innodb 的行锁是在有索引的情况下，没有索引的表是锁定全表的。

**Record Lock & Gap Lock & Next-Key Locks**

Mysql加锁过程详解（7）-初步理解MySQL的gap锁

<http://www.cnblogs.com/crazylqy/p/7689447.html>

Mysql加锁过程详解（8）-理解innodb的锁(record,gap,Next-Key lock)

<http://www.cnblogs.com/crazylqy/p/7773492.html>

**MySQL如何解决幻读？**

MySQL使用MVCC + Next-Key Locks解决幻读。除非当前事务中只存在快照读，可以用MVCC解决幻读。但如果当前事务中存在当前读，就必须使用Next-Key Locks来解决幻读。Next-Key Locks对记录索引和索引间的空隙加了锁，使得其他事务无法插入或删除数据，真正避免了幻读的产生。

具体分析见：

MVCC能解决幻读吗？

<https://www.jianshu.com/p/cef49aeff36b>

**MySQL如何定位慢查询？**

# MySQL优化技术：定位慢查询

<https://blog.csdn.net/qzc70919700/article/details/76566846>