局部性原理🡪即使所需取的值只有一个,innodb也会从数据库中取出其他相邻的值到内存中🡪也就是一页\*1,下次再拿时就可以直接从内存里去取

前驱节点，指非叶子节点的左子节点的最右叶子子树

后继节点，指非叶子节点的右子节点的最左叶子子树

\*1页🡪最大存储大小16384{

行\*2:最大大小🡪65535\*3

}

操作系统有一个🡪页单位🡪大概4kb的大小🡪Innodb也有一个页的单位,大概是16kb

\*2实际存储表格式🡪变长字段长度列表/null标志位/记录头信息/数据列表1/数据列表2...../隐藏主键/事务id\*4/回滚指针

🡪

变长字段实际储存的是varchar(20)中实际储存的字符串的长度🡪按照字段顺序排列,不需要用到key、value

🡪

null标记位🡪我们可以用一个特殊符号占位表示空,当数据库在头部写一个标记,确认每个位置是否为空

🡪当表中没有主键时,innodb会默认建立一个row\_id隐藏主键

\*3行溢出🡪当页储存不下时会被分页储存到不同页中

🡪存储格式:数据的部分内容+下一页的地址

🡪第二种方:行中不存数据,只存该数据在下一行中的地址,地址和数据分开储存

事务隔离级别:\*5

🡪读未提交:一个事务可以读其他还未提交数据的事务,会出现脏读

🡪读已提交:读一个事务提交后的数据,会出现不可重复读,幻读

🡪可重复读(mysql默认隔离级别):事务开启后,之后的事务修改不影响本次事务的结果

🡪串行化(用的少):串行条件下,当一个事务开启,其他事务的修改操作会被阻塞,读不影响

\*4transaction\_id事务id🡪负责保存最后一次事务修改过后的id也就是版本链的最顶端

版本链🡪每一个事务都有自己的id,版本链会保存事务的id,上一次事务修改记录的指针(包括提交和未提交)🡪他的保存位置undolog

🡪相当于,数据库的数据指向版本链的顶端,每本版本指向之前的事务版本:

🡪在每次查询时,当在读已提交事务隔离级别下,该语句会生成一个readview🡪m\_ids:[]它会保存事务在读取时哪些事务是活跃的并未提交,当该事务进行查询时,就会一次遍历到未活跃事务里没有的几率或者事务编号id等于自身事务id的记录

🡪可重复读事务隔离级别下,readview只生成一次,并且之后重复复用

\*5脏读,幻读,不可重复读

🡪脏读,读到了事务未提交的数据

🡪幻读,读到了事务已提交的新增数据

🡪不可重复读,读到了后续事务已提交的数据