# 概念

Insert Buffer是InnoDB存储引擎所独有的功能，即使在今天看来，其涉及思想也是非常先进的。通过Insert Buffer，InnoDB存储引擎可以大幅度提升数据库中**非唯一约束**的辅助索引插入性能。

在B+树索引中，对于自增主键值的插入是顺序的，因此插入能有较高的插入性能。但在实际生产环境中，用户表中的主键仅有且只能有一个，然而表中可能存在多个辅助索引。辅助索引根据类型在插入时可以分为比较顺序的插入和随机的插入两种。比较顺序的插入可能是时间字段，随机的插入可能更为广泛，如用户的名字。

当导入大批量数据（bulk load）时，一般建议先不建立辅助索引，当数据导入完成后再一次性创建辅助索引。

InnoDB存储引擎Insert Buffer的设计思想是，在插入时首先判断插入的辅助索引叶子是否在缓冲池中。若在则直接插入，否则将插入的记录放到Insert Buffer中，然后根据一些算法将Insert Buffer缓存的记录通过后台线程慢慢地合并（merge）回辅助索引页中。

说明：在MySQL-Server层也有类似的优化，即先检查是否存在唯一索引或主键，存在则update，不存在则插入，底层实现跟这里一样都是基于bitmap实现。

这样做的好处是：

1. 减少磁盘的离散读取
2. 将多次插入合并为一次操作

此外，需要注意的是，Insert Buffer只能缓存非唯一性约束的索引。若是唯一索引，那么在插入时需要判断插入的记录是否是唯一，这需要读取辅助索引页，而Insert Buffer的设计就是避免读取Insert Buffer，这就导致失去Insert Buffer的设计意义。

# 源码

ibuf0ibuf.cc/ibuf0inuf.h/ibuf0ibuf.ic：Insert Buffer实现

# 原理

## 架构实现

Insert Buffer其实也是一棵B+树。这棵B+树的键值为待插入到辅助索引页的page\_no。B+树索引页的大小一样为16KB，其内存同样使用缓存池。宏IBUF\_POOL\_SIZE\_PER\_MAX\_SIZE定义了Insert Buffer最多占用缓存池中页的数量。由于该值默认为2，因此表示Insert Buffer最多可能占用二分之一的缓冲池大小。

## 相关数据结构

## 死锁

### latch顺序

### 并发控制

### 异步I/O线程

## 维护

### 记录合并

### 空间收缩

# 应用

手动实现一个insert buffer，基于bitmap。