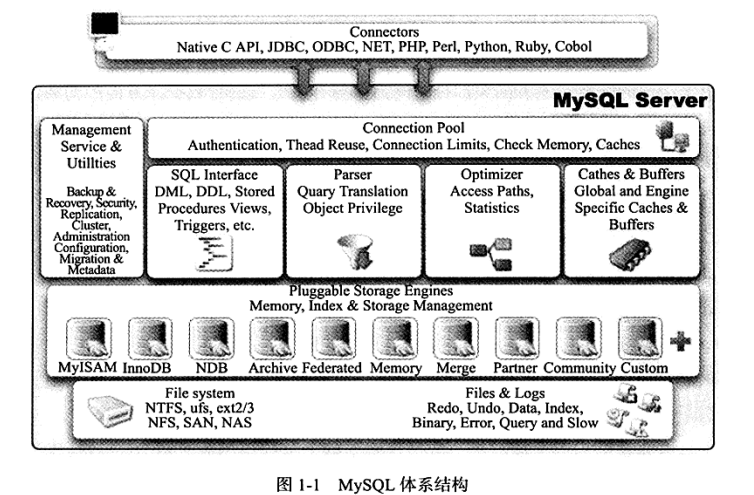
## **MySQL数据库的体系结构**：

由以下几部分组成：连接池组件、管理服务和工具组件、SQL接口组件、查询分析器组件、优化器组件、缓冲（Cache）组件、插件式存储引擎、物理文件。



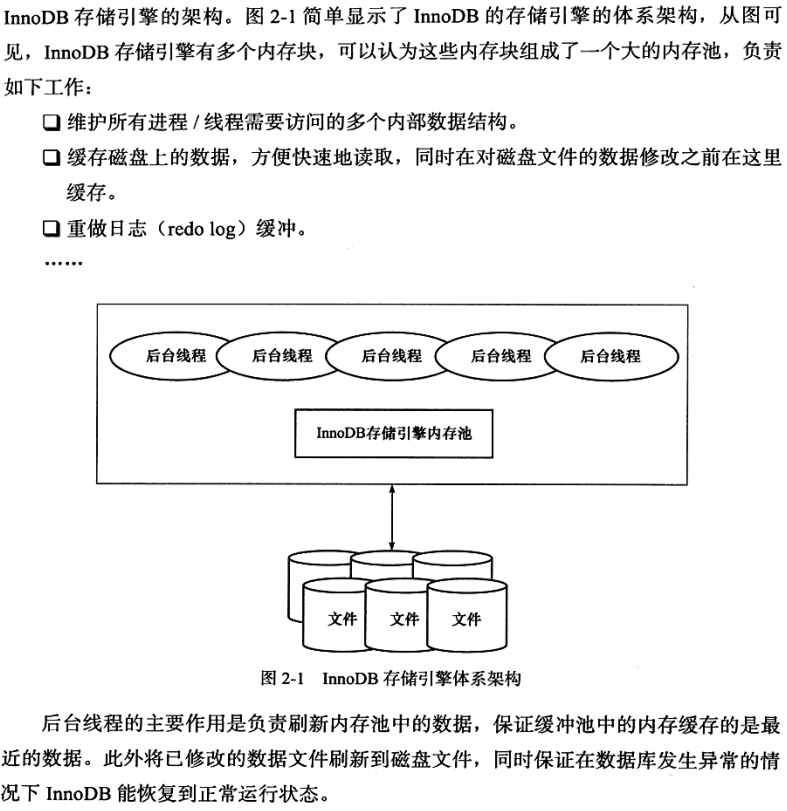
## 存储引擎

### InnoDB存储引擎

支持事务，其设计目标主要面向在线事务处理（OLTP）的应用。其特点是行锁设计、支持外键，即默认读取操作不会产生锁。 InnoDB通过使用多版本并发控制（MVCC）来获得高并发性，并且实现了sql标准的4种隔离级别，默认为repeatable级别。

### MyISAM存储引擎

不支持事务、表锁设计，支持全文索引，主要面向一些OLAP（联机分析处理）数据库应用。



## InnoDB存储引擎

### InnoDB体系架构

#### 后台线程

1. Master Thread

主要负责将缓冲池中的数据异步刷新到磁盘，保证数据的一致性，包括脏页的刷新、合并插入缓冲、UNDO页的回收等。

1. IO Thread

在InnoDB存储引擎中大量使用了AIO来处理写IO请求，这样可以极大提高数据库的性能。而IO Thread的工作主要是负责这些IO请求的回调处理。

SHOW VARIABLES LIKE 'innodb\_version'

SHOW VARIABLES LIKE 'innodb\_%\_io\_threads'

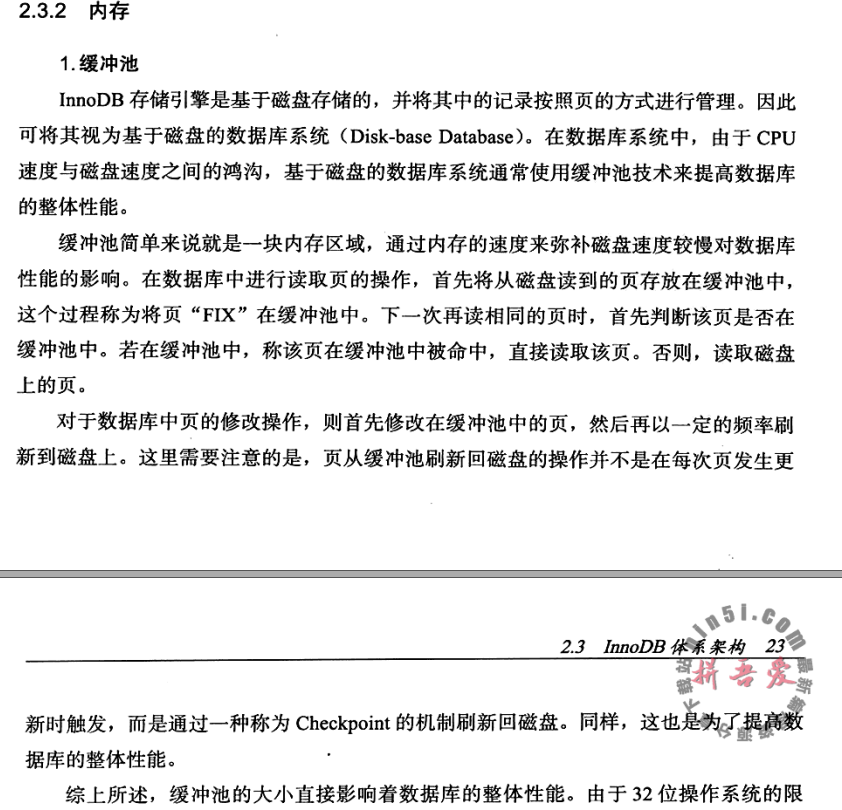
可以通过命令SHOW ENGINE INNODB STATUS来观察InnoDB中的IO Thread

1. Purge Thread

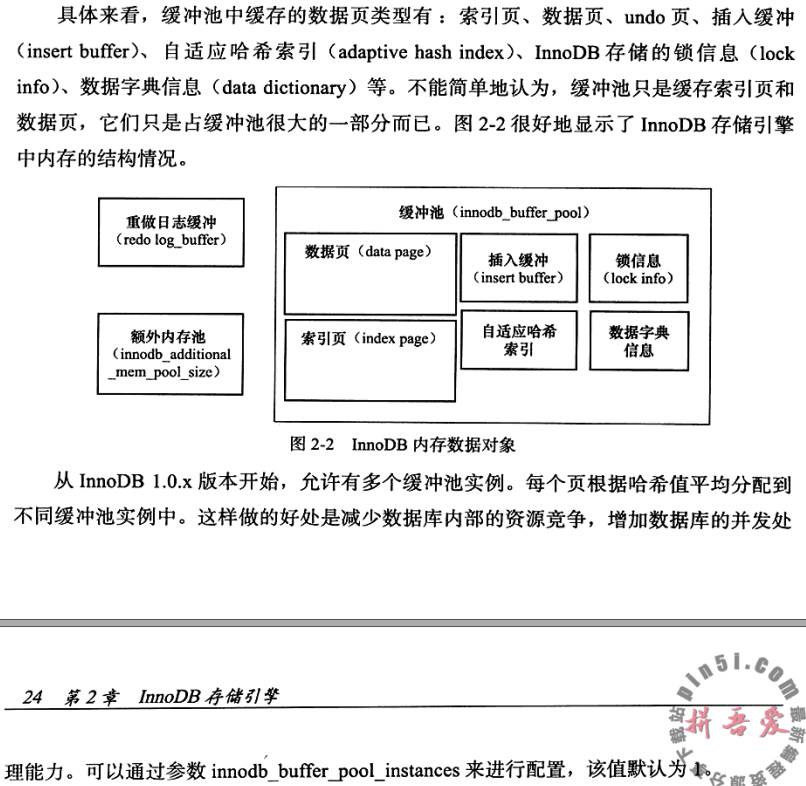
事务被提交后，其所使用的undolog可能不再需要，因此需要PurgeThead来回收已经使用并分配的undo页

show VARIABLES like 'innodb\_purge\_threads'

#### 内存

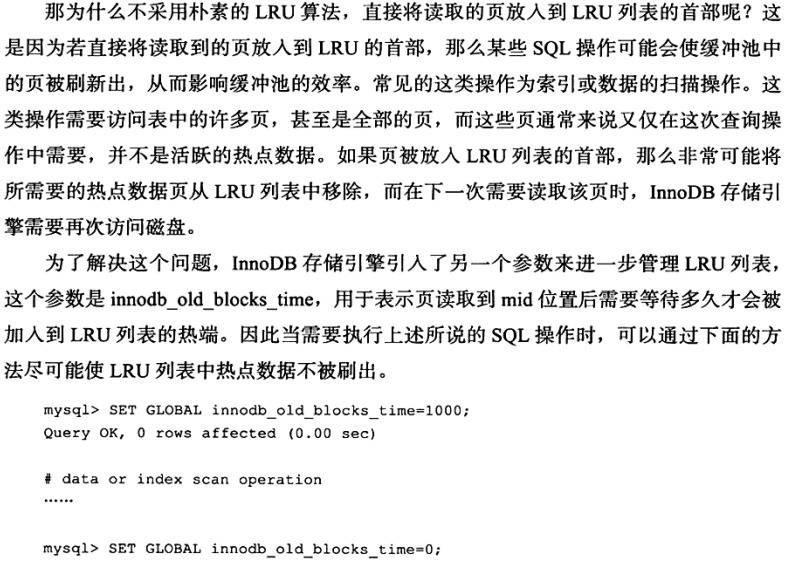


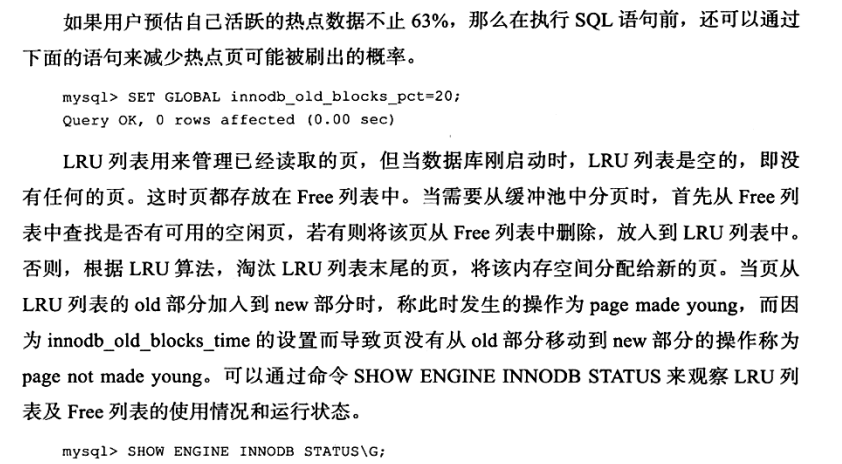
show VARIABLES like 'innodb\_buffer\_pool\_size'

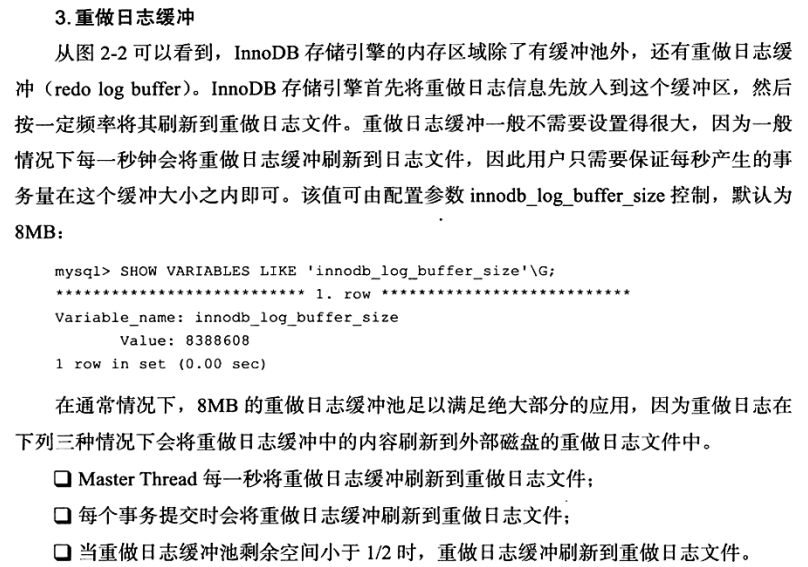


LRU List、Free List和Flush List

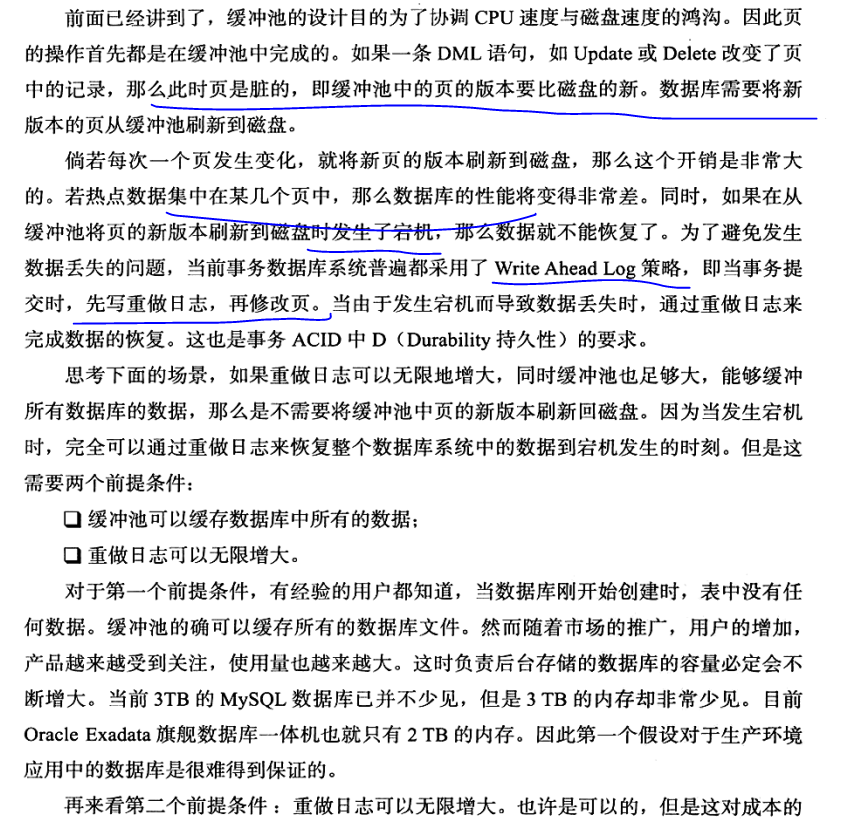
InnoDB对传统的LRU算法做了一些优化，加入了midpoint位置。新读取的页，虽然是最新访问的页，但并不是直接放入到LRU列表的首部，而是放入到LRU列表的midpoint位置。Midpoint位置可由参数innodb\_old\_blocks\_pct控制







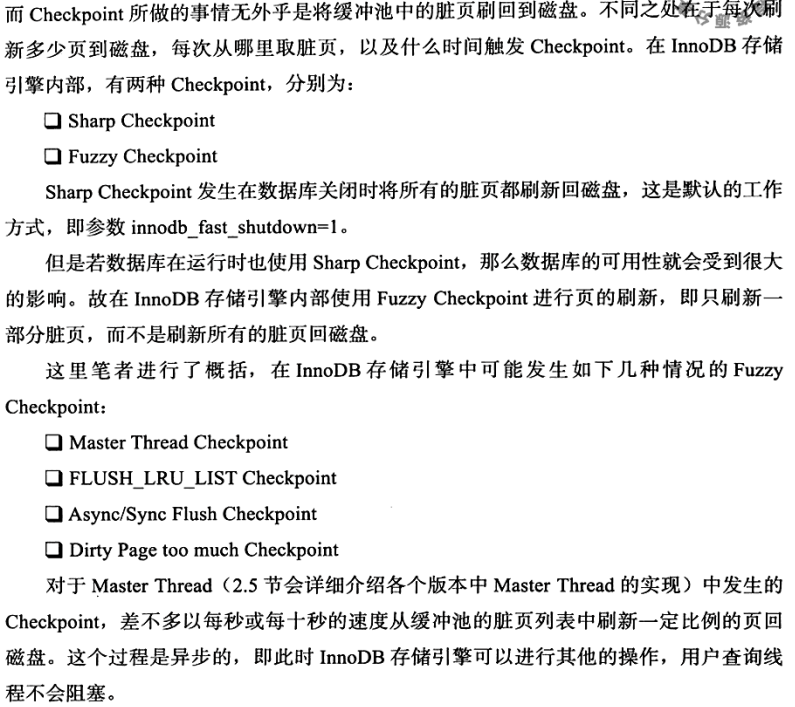
### Checkpoint技术



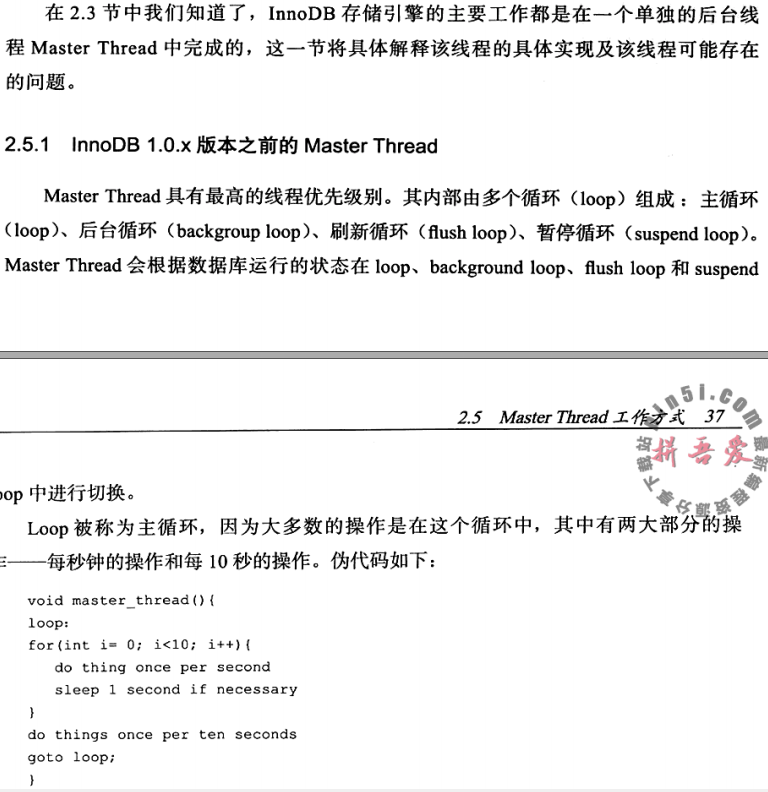
Checkpoint（检查点）技术的目的是解决以下几个问题：

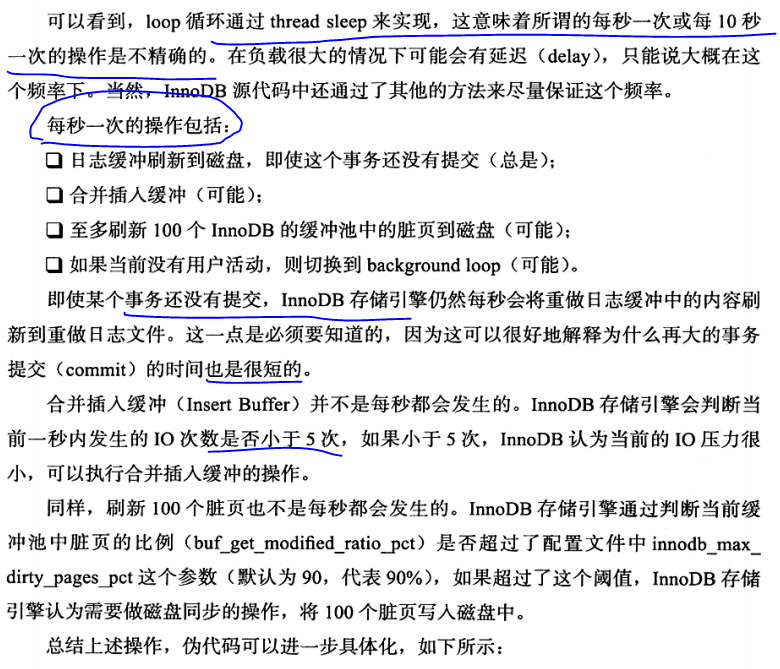
缩短数据库的恢复时间；缓冲池不够用时，将脏页刷新到磁盘；重做日志不可用时，刷新脏页

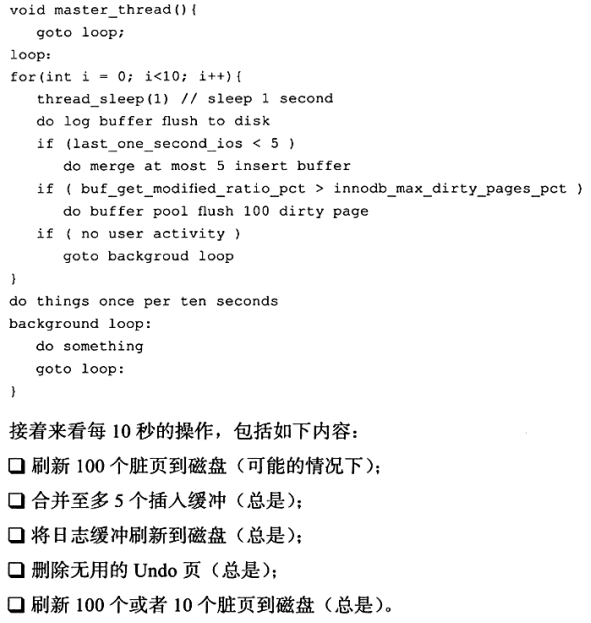
对于InnoDB存储引擎而言，其是通过LSN（Log Sequence Number）来标记版本的。而LSN是8字节的的数字，其单位是字节。每个页有LSN，重做日志中也有LSN，checkpoint也有LSN。可以通过命令show engine innodb status来观察

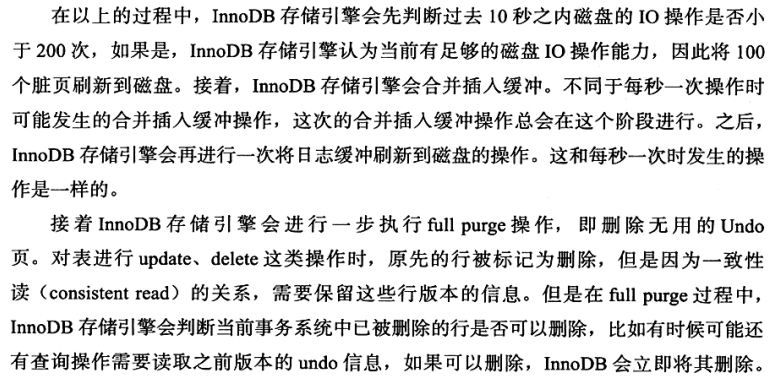


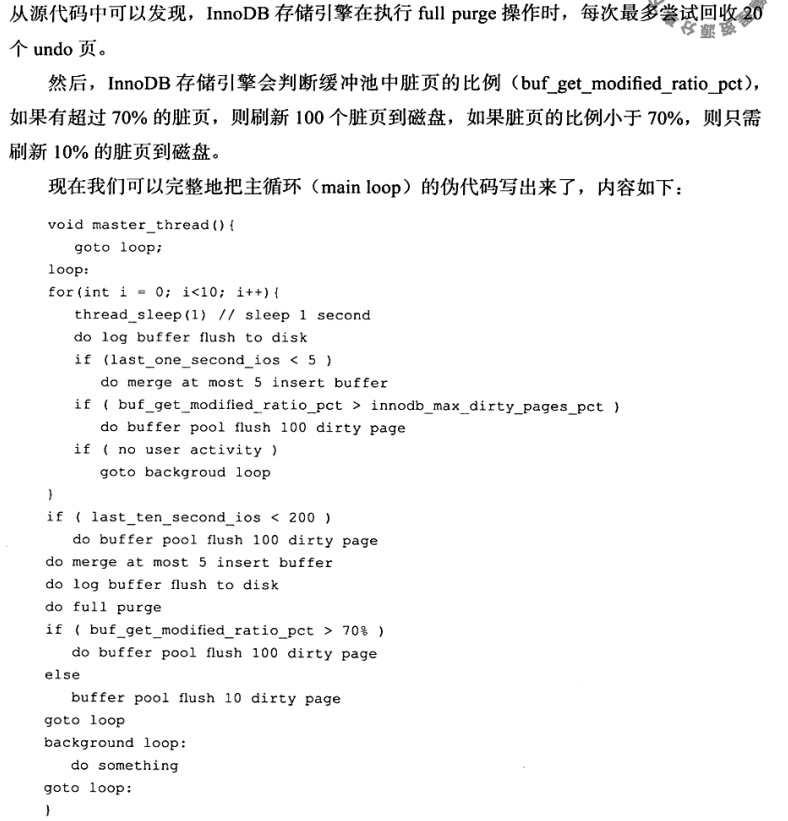
### Master Thread工作方式











对于backgroud loop，若当前没有用户活动（数据库空闲时）或者数据库关闭（shutdown），就会切换到这个循环。Background loop会执行以下操作：

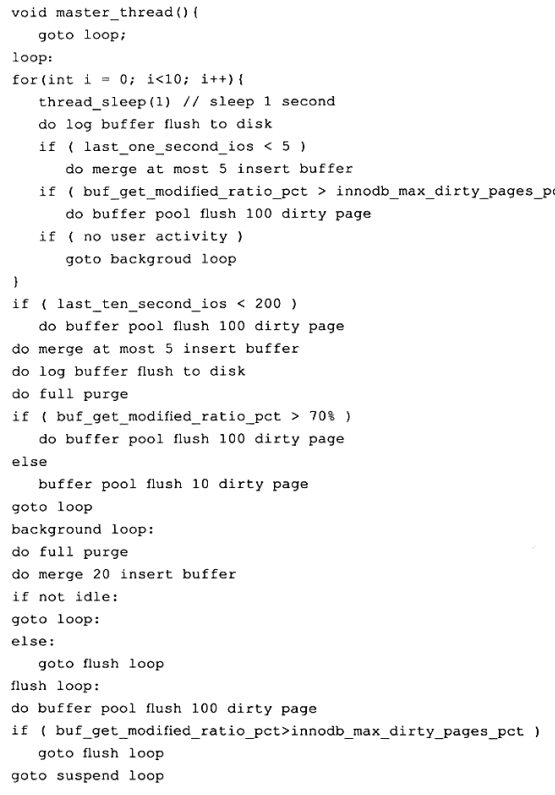
删除无用的undo页（总是）；

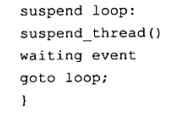
合并20个插入缓冲（总是）；

跳回到主循环（总是）；

不断刷新100个页直到符合条件（可能，跳转到flush loop中完成）

若flush loop中也没有什么事情可以做了，innoDB存储引擎会切换到suspend\_loop，将Master Thread挂起，等待事件的发生。若用户启用（enable）了InnoDB存储引擎，却没有使用任何InnoDB存储引擎的表，那么Master Thread总是处于挂起的状态。

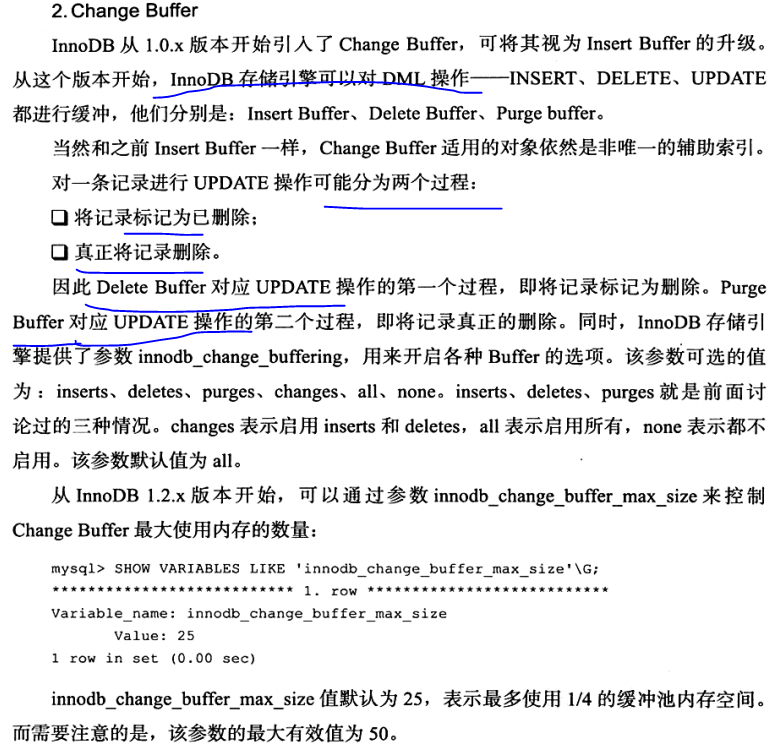




### InnoDB关键特性

InnoDB存储引擎的关键特性包括：

插入缓冲（insert buffer）；两次写（double write）；自适应哈希索引（adaptive hash index）；异步io（async io）；刷新邻接页（flush neighbor page）



在默认情况下，mysql数据库并不启动慢查询日志，用户需要手动将这个参数设为on：

**Show** Variables **like** "long\_query\_time"

**show** variables **like** "log\_slow\_queries"

是否使用索引

**Show** Variables **like** "log\_queries\_not\_using\_indexes"

当越来越多的sql查询被记录到了慢查询日志文件中，此时要分析该文件就显得不是那么简单和直观的了。这时可以使用mysqldumpslow命令

Mysqldumpslow nh122-190-slow.log

如果用户希望得到执行时间最长的10条sql语句，可以执行如下命令

Mysqldumpslow –s al –n 10 david.log

Msyql5.1开始可以将慢查询的日志记录放入一张表中，这使得用户的查询更加方便和直观，慢查询表在mysql架构下，名为slow\_log，其表结构定义如下：

Show create table mysql.slow\_log

**show** variables **like** "log\_output"

**set** **global** log\_output="table"

**select** \* **from** mysql.slow\_log

## 第四章 表

在innoDB存储引擎中，表都是根据主键顺序组织存放的，这种存储方式的表称为索引组织表。在innoDB存储引擎表中，每张表都有个主键，如果在创建表时没有显示地定义主键，则innoDB存储引擎会按如下方式选择或创建主键：

首先判断表中是否有非空的唯一索引，如果有，则该列即为主键。

如果不符合上述条件，innoDB存储引擎自动创建一个6字节大小的指针。

