# 背景

缓冲池的设计目的是为了协调CPU速度与磁盘速度的鸿沟。页的操作都在在缓冲池中完成的。如果一条DML语句，如Update或者Delete改变了页中的记录，那么此时页是脏的，即缓冲池中的页的版本比磁盘的要新。数据库需要将最新版本的页刷新到磁盘。

InnoDB采用Write Ahead Log策略来防止宕机数据丢失，即事务提交时，先写重做日志，再修改内存数据页，这样就产生了脏页。既然有重做日志保证数据持久性，查询时也可以直接从缓冲池页中取数据，那为什么还要刷新脏页到磁盘呢？如果重做日志可以无限增大，同时缓冲池足够大，能够缓存所有数据，那么是不需要将缓冲池中的脏页刷新到磁盘。但是，通常会有以下几个问题：

1. 服务器内存有限，缓冲池不够用，无法缓存全部数据
2. 重做日志无限增大成本要求太高
3. 宕机时如果重做全部日志恢复时间过长

事实上，当数据库宕机时，数据库不需要重做所有的日志，只需要执行上次刷入点之后的日志。这个点就叫做Checkpoint，它解决了以上的问题：

1. 缩短数据库恢复时间
2. 缓冲池不够用时，将脏页刷新到磁盘
3. 重做日志不可用时，刷新脏页

重做日志被设计成可循环使用，当日志文件写满时，重做日志中对应数据已经被刷新到磁盘的那部分不再需要的日志可以被覆盖重用。

# 原理

## LSN

InnoDB引擎通过LSN(Log Sequence Number)来标记版本，LSN是日志空间中每条日志的结束点，用字节偏移量来表示。每个page有LSN，redo log也有LSN，Checkpoint也有LSN。可以通过命令show engine innodb status来观察：

---

LOG

---

Log sequence number 11102619599

Log flushed up to 11102618636

Last checkpoint at 11102606319

0 pending log writes, 0 pending chkp writes

15416290 log i/o's done, 12.32 log i/o's/second

## 作用

1、缩短数据库的恢复时间

当数据库宕机时，数据库不需要重做所有日志，因为CheckPoint之前的页都已经刷新回磁盘。只需对CheckPoint后的重做日志进行恢复，从而缩短恢复时间。

2、缓冲池不够用时，将脏页刷新到磁盘

当缓存池不够用时，LRU算法会溢出最近最少使用的页，若此页为脏页，会强制执行CheckPoint，将该脏页刷回磁盘。

3、重做日志不可用时，刷新脏页

不可用是因为对重做日志的设计是循环使用的。重做日志可以被重用的部分，是指当数据库进行恢复操作时不需要的部分。若此时这部分重做日志还有用，将强制执行CheckPoint，将缓冲池的页至少刷新到当前重做日志的位置。

# 分类

## Sharp CheckPoint

发生在数据库关闭时，会将所有的脏页刷回磁盘。

## Fuzzy CheckPoint

为提高性能，数据库运行时使用Fuzzy CheckPoint进行页的刷新，即只刷新一部分脏页。

**Fuzzy CheckPoint触发时机：**

1、Master Thread CheckPoint

差不多以每秒或每十秒的速度，从缓存池脏页列表中刷新一定比例的页，且此过程是异步的，因此不会阻塞其他操作。

2、FLUSH\_LRU\_LIST CheckPoint

因为InnoDB需要保证LRU列表中有一定数量的空闲页可使用，倘若不满足该条件，则会将LRU列表尾端的页移除，若这些页中有脏页，则会进行CheckPoint。该检查被放在一个单独的Page Cleaner线程中进行。

用户可以通过innodb\_lru\_scan\_depth控制LRU列表的可用页数量，默认为1024。

3、Async/Sync Flush CheckPoint

当重做日志文件不可用的情况下，会强制将一些页刷回磁盘。Async/Sync Flush CheckPoint是为了重做日志的循环使用的可用性。

简单来说，Async发生在要刷回磁盘的脏页较少的情况下，Sync发生在要刷回磁盘的脏页很多时。

这部分操作放入到了Page Cleaner线程中执行，不会阻塞用户操作。

4、Dirty Page too much CheckPoint

是指当脏页比例太多，会导致InnoDB存储引擎强制执行CheckPoint。目的根本上还是为了保证缓冲池中有足够可用的页。

比例可由参数innodb\_max\_dirty\_pages\_pct控制。若该值为75，表示当缓冲池中脏页占据75%时，强制CheckPoint。