第一章：

1. InnoDB的体系结构包括哪两大部分？
2. InnoDB后台线程的作用？

一、刷新缓存内存池到最新的状态；二、刷新脏数据到数据库保证数据一致性；三、保证当数据库发生破坏时恢复数据库

* 1. 1个Main Thread：保证数据一致性（缓存和数据库），包括脏页刷新、合并插入缓存、undo页回收等
  2. 4个IO Thread：InnoDB中存在很多来自客户端的异步IO请求（AIO），那么InnoDB接收请求后由于是异步，所以不会立刻回调结果，此时InnoDB就需要有IO Thread去处理这些请求
  3. 1个Purge Thread：清除那些已经提交了事务的无用的undo log
  4. 1个Page Cleaner Thread：完成脏页的刷新（内存脏数据刷新到磁盘）

1. InnoDB内存池的作用？

一、维护存储各个InnoDB线程在使用的时候所需要的各个内存结构（）

二、存储缓存，加快客户端与数据库间读写的速度

三、保存重做日志缓存

4. 内存池的主要组成部分：

数据页、索引页、插入缓冲、锁信息、自适应哈希索引、重做日志缓冲

5. 内存池的其他组成部分：

LRU List（存储热点页的List，保证在页面置换的时候替换非热点页。midpoint技术）

Free List（存储空白页）

Flush List（存储脏页，checkpoint刷新脏页就是将Flush List中的脏页刷新到磁盘）

1. 什么是LRU midpoint技术？
2. 为什么LRU List 要使用LRU midpoint insertion而不是用朴素的LRU技术？
3. 重做日志缓冲（redo log buffer）作用？
4. 内存池的重做日志缓冲中的redo log刷新到磁盘中重做日志缓冲的3种情况？
5. 为何使用checkpoint技术？（缩短恢复数据库时间、刷新脏页腾出足够的缓冲池空闲区域、刷新redolog保证redolog缓冲区域足够存储新的redolog）
6. 有多少种checkpoint技术？
   1. Sharp checkpoint
   2. Fuzzy checkpoint
      1. Main thread checkpoint
      2. Flush LRU List checkpoint
      3. Async/Sync Flush checkpoint
      4. Dirty page too much checkpoint
7. Main thread 的工作原理
   1. 主循环（主线程每秒都会做的操作、主线程每10秒都会做的操作）
   2. 后台循环
8. 为何需要插入缓冲技术？（插入一个新的非聚集索引到非聚集索引页要进行离散读整个非聚集索引，不像聚集索引那样直接插入，速度慢）
9. 引入了插入缓冲技术能带来什么好处？（解决了因为仅仅一次非聚集索引插入就单独进行非聚集索引插入的弊端，等很多次插入操作混合，在一块进行非聚集索引插入）
10. 插入缓冲技术怎么实现？（先判断，再进行插入缓冲技术）
11. 使用插入缓冲技术要符合哪两个条件？（非聚集索引，非唯一的非聚集索引）

第二章

1. InnoDB的五个特性：插入缓冲、两次写、AHI自适应哈希索引、AIO异步IO、邻近页刷新
2. LRU List和自适应哈希AHI都是对热点页操作，不同之处：LRU List起到一个告知该替换哪个页的功能；而AHI起到一个直接定位搜索热点页访问热点页数据的功能
3. 插入索引的内部结构？
4. Insert buffer bitmap的作用？（记录每个辅助索引页的可用空间，当一个辅助索引页的bitmap显示可用空间快要满了，那么此时会立即进行插入缓冲的合并）
5. 插入缓冲的升级：change buffer，解决了插入缓冲宕机恢复数据慢以及写密集操作时占用内存空间大的问题，使得缓冲增加了insert buffer、delete buffer、purge buffer
6. Merge insert buffer发生的三个时间点？（索引页加载到缓冲池、bitmap空间不可用、master thread）
7. 非辅助索引的节点信息？（辅助索引-主键的映射）
8. 合并插入发生了什么？合并插入并非刷新到磁盘，而是将插入缓冲的内容插入到缓冲池对应的索引页上
9. 为什么需要两次写？（两次写和redo log共同保证了数据库宕机后数据的恢复问题，redo log仅仅是记录着脏页变化的物理信息）
10. AHI自适应哈希索引为什么叫自适应？（MySQL自动监控是否对某一些热点页新构建哈希索引）
11. AHI自适应哈希索引解决了什么问题？
12. AIO异步IO相比起同步IO的好处是什么（同步IO是阻塞式的IO，知道处理完同步IO系统才会接受下一个IO）
13. AIO具体做了什么（接受异步IO并不着急处理，而是等待多个IO的请求，进行合并再进行处理，数据库实例中的IO线程也是可以异步处理IO操作，只要后面回调处理结果就好）
14. 数据库何时使用异步IO？（read ahead操作时、脏页刷新也就是写入磁盘的时候）
15. 数据库的预读操作？

第三章