# MySQL InnoDB表存储引擎

## 1 背景

**①**MySQL是最受欢迎的开源数据库。

**②**MySQL的客户范围很广泛，如FaceBook、Google、Adobe、LinkedIn、Wikipidea、YouTube等。

**③**一般来说，MySQl的两个特征——使用简单和性能优异——是MySQL受欢迎的主要原因。除此之外，支持众多的平台（包括Windows），和内嵌的复制支持，这为只读客户端提供了一个容易实现的支持横向扩展的解决方案。这也是MySQL受欢迎的重要原因。

**④**著名的MySQL存储引擎有MyISAM、InnoDB、NDB等。

**⑤**相比MyISAM，InnoDB支持事务以及事务的ACID特性，行锁、MVCC、自动恢复和数据污染检测。

**⑥**MySQL的缺陷是如果你的网站或论坛太受欢迎，数据库会成为性能的瓶颈，你需要学习一些特定的技能和工具来解决它。

**⑦**MySQL区别于SQL Server、Oracle等数据库的最大特征是其插件式存储引擎架构。

**⑧**InnoDB使得MySQL跃入了企业级数据库领域。

## 2 MySQL体系结构和存储引擎

### 1 数据库与实例

数据库：物理操作系统文件或其它形式文件的集合。

实例：MySQL数据库实例在操作系统中的表现就是一个进程，用于真正操作数据库文件，用户对数据库做的任何操作都是在数据库实例下进行的。

数据库实例的启动：当启动实例时，会去读取配置文件，配置文件中的datadir参数制订了数据库所在的路径。

## 3 InnoDB存储引擎

### 1 概述

InnoDB存储引擎时OLTP应用中核心表的首选引擎。

### 2 版本

**老版本**（静态编译的版本）：支持ACID、行锁、MVCC

**1.0版本**（Plugin版本）：支持compress和dynamic页格式

**1.1版本**：支持Linux AIO、多回滚段

**1.2版本**：支持全文索引、在线索引添加

### 3 体系结构

包括后台线程和内存池。后台线程包括Master、IO、Purge、Page Cleaner四种线程，内存池包括缓冲池（可有多个）、重做日志缓冲、额外内存池。

#### 1 后台线程

**Master Thread**：核心后台线程，主要负责将缓冲池中的数据异步刷新到磁盘，包括脏页的刷新、合并插入缓冲、undo页的回收等。

**IO Thread**：负责AIO请求的回调处理、分为write、read、insert buffer、log四种。

**Purge Thread**：事务被提交后，其所使用的undolog可能不再需要，需要Purge Thread来回收已经使用并分配的undo页。以前这个工作由Master Thread独立完成，Purge Thread的出现减轻了Master Thread的工作。

**Page Cleaner Thread**：刷新脏页到磁盘，以前这个工作由Master Thread独立完成，Page Cleaner Thread的出现减轻了Master Thread的工作，而且**不会再对用户查询线程进行阻塞**。

#### 2 内存池

**缓冲池**：

基于磁盘按页存储，缓冲池中页的类型包括：索引页、数据页、undo页、插入缓冲、自适应哈希索引、锁信息、数据字典信息等。允许有多个缓冲池实例，每个页根据哈希值平均分配到不同缓冲池实例中。

采用LRU算法管理缓存页的置换，淘汰页时置换**LRU列表**尾部的页。新读取到的页加入midpoint位置，midpoint之前的列表称为new列表，存储活跃的热点数据，之后的称为old列表。还可以指定新读取到的页经过多长时间才可以加入new列表。InnoDB需要保证LRU列表中有差不多100个空闲页可供使用。

数据库刚启动时，LRU列表为空，所有页都在**Free列表**中，当需要缓存读取的页时，从Free列表中删除一个页，加入到LRU列表中缓存读取到的页。若Free列表没有空闲页，则淘汰LRU列表尾部的页。LRU列表的页加上Free列表中的页不等于缓冲池总大小，因为有些页会分配给锁信息等页，这部分页不需要LRU、算法进行维护，因此不存在于LRU列表中。

**缓冲池命中率**通常不小于95%，如果小于95%，则用户需要观察是否是由于全表扫描引起的LRU列表被污染的问题。

InnoDB 1.0开始支持页压缩，对于压缩的页，通过**unzip\_LRU列表**来管理，LRU列表中包含了unzip\_LRU列表中的页。在unzip\_LRU列表中对不同大小的压缩页单独管理，通过伙伴算法进行内存的分配。

在LRU中页被修改后，称为脏页，数据库会通过CheckPoint机制将脏页刷新会磁盘。**Flush列表**中的页即为脏页，脏页既存在于LRU列表中又存在于Flush页表中，LRU列表用来管理缓冲池中页的可用性，Flush列表用来管理将脏页刷新会磁盘，二者互不影响。

**重做日志缓冲**：

三种情况重做日志缓冲将被刷新到磁盘（全部）：

①Master Thread每一秒刷新一次

②每个事务提交时刷新一次

③当重做日志缓冲剩余空间不足1/2时,刷新一次

**额外内存池**：

对一些InnoDB引擎本身使用的数据结构分配空间，如缓冲池的帧缓冲对应的缓冲控制对象，其记录了LRU、锁、等待等信息，因此需要申请很大的缓冲池时需要增大额外内存池。

### 4 CheckPoint技术

为了避免数据丢失问题（脏页中的数据为刷新回磁盘），事务数据库系统普遍采用了**Write Ahead Log**策略，即当事务提交时，先写重做日志，再修改页。InnoDB通过**LSN**来标记版本，每个页、重做日志、CheckPoint都有自己的LSN。

在InnoDB中，有两种CheckPoint策略：

**Sharp CheckPoint**：在数据库关闭时会将所有的脏页都刷新回磁盘。

**Fuzzy CheckPoint**：InnoDB中可能发生4中情况的Fuzzy CheckPoint。

**Master Thread CheckPoint**：Master Thread以每秒或每十秒的速度从缓冲池的脏页列表中**异步**刷新一定比例的页回磁盘。

**FLUSH\_LRU\_LIST CheckPoint**:如果LRU尾部的页被淘汰置换，且这个页为脏页，需要进行脏页刷新。

**Async/Sync Flush CheckPoint：保证循环使用重做日志文件的可用性，当必须的（脏页没有被刷新回磁盘的）重做日志大小超过重做日志文件大小的75%时，会异步刷新足够的脏页回磁盘。**

**Dirty Page too much CheckPoint**