

# 数据库性能优化之道：Buffer Pool 深度剖析（二）

后端出路在何方 2025-01-13 75 阅读5分钟 专栏：MySQL

## 1. Buffer Pool 的组成

Buffer Pool 是一块被精心管理的内存区域，它的组成可以分为以下几个部分：

### 1.1 数据页（Data Pages）

- 数据页是 Buffer Pool 的核心组成部分。
- 每一个数据页存储的是磁盘上某部分数据的内容，例如：
  - 一个表的一部分行记录。
  - 一个索引的一部分节点。
- 数据页是数据库中数据的基本存储单位（如 InnoDB 中的数据页大小一般为 16KB）。

### 1.2 内存块（Blocks）

- Buffer Pool 的内存被分割成一个个固定大小的内存块，每个块可以存放一个数据页。
- 可以把内存块想象成一个“空房间”，装载不同的数据页。
- 内存块由 Buffer Pool 统一管理，负责分配和回收。

### 1.3 控制块（Control Blocks）

- 每个内存块都有一个对应的控制块，用来记录与这个内存块相关的元数据。
- 控制块的作用类似于“身份证”，它记录了数据页的信息，包括：
  - 数据页的 **Page ID**（页号）。
  - 引用计数**：记录数据页被访问的次数。
  - 数据页的状态：比如数据是否被修改、是否在链表中。
  - 链表指针：用于表示这个块在链表中的位置（比如自由链表、LRU 链表等）。

### 1.4 链表结构（Linked Lists）

Buffer Pool 中通过链表来管理数据页的状态，主要有以下三种链表：

#### 1. 自由链表（Free List）

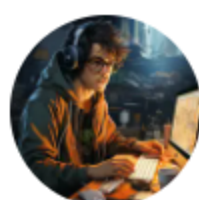
- 存储当前未分配的内存块（空闲块）。
- 当数据库需要加载新的数据页时，优先从自由链表中分配内存块。
- 如果自由链表为空，则需要从 LRU 链表中淘汰旧的数据页。

#### 2. 使用链表（LRU List）

- LRU（Least Recently Used）链表存储当前正在使用或最近使用过的数据页。
- 按照最近使用的时间排序，最近使用的页放在链表头部，最久未使用的页放在链表尾部。
- 当需要腾出空间时，优先从链表尾部淘汰最久未使用的页。

#### 3. 脏页链表（Flush List）

- 存储所有被修改过但尚未写回磁盘的数据页（即“脏页”）。
- 后台线程会定期扫描脏页链表，并将这些脏页写回磁盘以保持数据一致性。



后端出路在何方 LV.3  
破写代码的

79

文章

15k

阅读

62

粉丝

关注

私信

#### 目录

收起 ^

##### 1.4 链表结构（Linked Lists）

扩展知识

LRU链表的实现方式

#### 2. Buffer Pool 的运行机制

2.1 Buffer Pool 的初始化

2.2 Buffer Pool 的数据管理

#### 3. Buffer Pool 的组成与运行机制

#### 4. Buffer Pool 的运行流程图

#### 总结

#### 相关推荐

G1原理—5.G1垃圾回收过程之Mixed GC  
36阅读 · 1点赞

BeanNameGenerator解决mapper重名...  
45阅读 · 0点赞

Springboot实现CTWing接口对接  
59阅读 · 1点赞

🌟Nacos2🌟服务订阅与推送🌟  
1.3k阅读 · 4点赞

dubbo3 filter（过滤器）自定义过滤器  
72阅读 · 0点赞

#### 精选内容

掌握线程池先从源码注释出发  
稻草人2222 · 50阅读 · 2点赞

系统稳定性的基石：限流在 AutoMQ 中...  
AutoMQ · 15阅读 · 0点赞

鸿蒙轻内核A核源码分析系列七 进程管...  
别说我什么都不会 · 17阅读 · 1点赞

OpenHarmony（鸿蒙南向开发）——小...  
塞尔维亚大汉 · 31阅读 · 0点赞

哈啰出行Java 一面，我扛住了！！  
小林coding · 1.4k阅读 · 8点赞

#### 找对属于你的技术圈子

回复「进群」加入官方微信群



AI  
助手



扩展知识

LRU（Least Recently Used）链表是一种用于实现缓存机制的数据结构，它记录了数据页的使用顺序，以便能够快速找到并淘汰最久未使用的数据页。以下是LRU链表的实现方式及其在Buffer Pool中的应用的详细解释：

LRU链表的实现方式

1. 数据结构：
- 双向链表：每个节点包含数据页的指针和两个指针（前驱和后继），用于维护节点的顺序。

◦ 哈希表：用于快速查找数据页是否在缓存中，哈希表的值是双向链表的节点指针。
2. 操作：
- 访问数据页：

▪ 如果数据页在缓存中（哈希表命中），将该节点移动到链表的头部。

▪ 如果数据页不在缓存中（哈希表未命中），从磁盘加载数据页，创建新节点，插入到链表的头部，并更新哈希表。

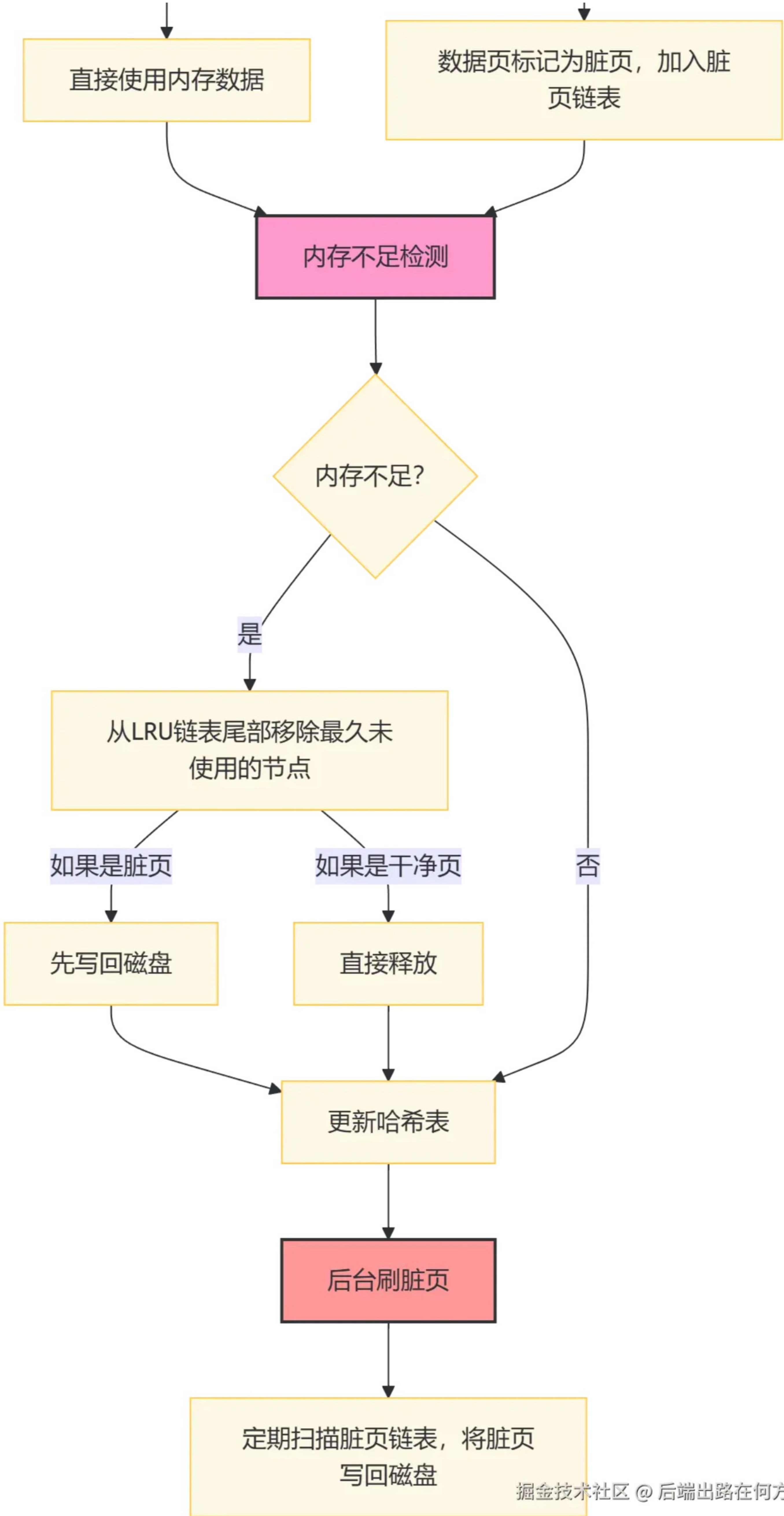
◦ 淘汰数据页：

▪ 当缓存满时，从链表的尾部移除最久未使用的节点，并从哈希表中删除对应的条目。

▪ 如果移除的节点是脏页，先将其写回磁盘。
- 以下是LRU链表在Buffer Pool中的实现方式的流程图：
- ```
graph TD; A[系统启动] --> B[分配内存，初始化自由链表和LRU链表]; B --> C[数据访问请求]; C --> D{检查数据页是否已在 Buffer Pool}; D -- "是 (命中缓存)" --> E[将节点移动到LRU链表头部]; D -- "否 (未命中缓存)" --> F[从磁盘加载到内存块]; F --> G[创建新节点，插入到LRU链表头部]; E --> H[数据页操作]; G --> H; H -- "读取操作" --> I[ ]; H -- "写入操作" --> J[ ]
```

The flowchart illustrates the LRU implementation in a Buffer Pool. It begins with '系统启动' (System Start), leading to '分配内存，初始化自由链表和LRU链表' (Allocate memory, initialize free list and LRU list). This is followed by '数据访问请求' (Data access request). A decision diamond asks '检查数据页是否已在 Buffer Pool' (Check if data page is in Buffer Pool). If '是 (命中缓存)' (Yes, hit cache), the flow goes to '将节点移动到LRU链表头部' (Move node to LRU list head). If '否 (未命中缓存)' (No, miss cache), it goes to '从磁盘加载到内存块' (Load from disk to memory block), then to '创建新节点，插入到LRU链表头部' (Create new node, insert to LRU list head). Both paths lead to '数据页操作' (Data page operation). From here, the flow branches into '读取操作' (Read operation) and '写入操作' (Write operation).
- Captured by FireShot Pro: 19 2月 2025, 16:26:52  
https://getfireshot.com





掘金技术社区 @ 后端出路在何方

## 2. Buffer Pool 的运行机制

Buffer Pool 的运行机制可以分为以下几个过程：

### 2.1 Buffer Pool 的初始化

- 系统启动时：
  - 数据库会为 Buffer Pool 分配一块指定大小的内存（可以通过配置参数设置大小，比如 InnoDB 的 `innodb_buffer_pool_size`）。
  - 这块内存被划分成多个固定大小的内存块。
  - 每个内存块都与一个控制块相关联，控制块记录内存块的元信息。
  - 所有未使用的内存块会加入到自由链表中。

### 2.2 Buffer Pool 的数据管理

Buffer Pool 的核心任务是高效管理数据页的加载、使用和释放。具体过程如下：

#### 1. 加载数据页

- 当数据库需要访问某个数据页时，会先检查这个数据页是否已经存在于 Buffer Pool 中。
  - 如果存在（命中缓存），直接从内存读取数据。
  - 如果不存在（未命中缓存），则需要从磁盘加载。
- 加载时会从自由链表中分配一个空闲块，如果自由链表为空，则从 LRU 链表中淘汰最久未使用的页。

#### 2. 标记脏页

- 如果对数据页进行了修改，数据库不会立刻把修改写入磁盘，而是将内存中的数据标记为“脏页”。
- 脏页会被加入到脏页链表中，等待后台线程（Flush Thread）将其写回磁盘。

#### 3. 淘汰数据页

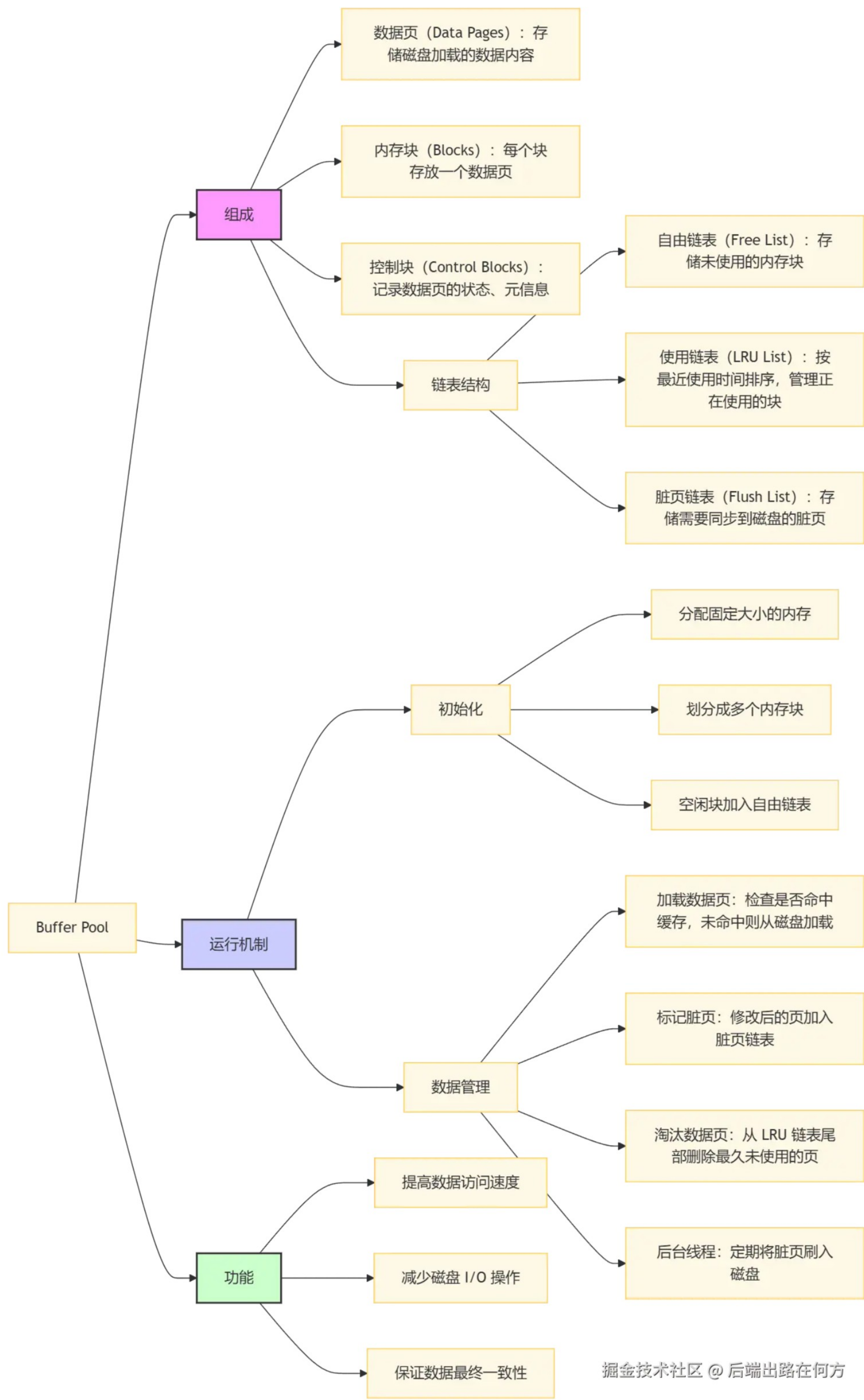


- 如果 Buffer Pool 中没有空闲块，需要从 LRU 链表中淘汰最久未使用的数据页。
- 如果被淘汰的数据页是脏页，必须先写回磁盘。

4. 后台线程刷脏页

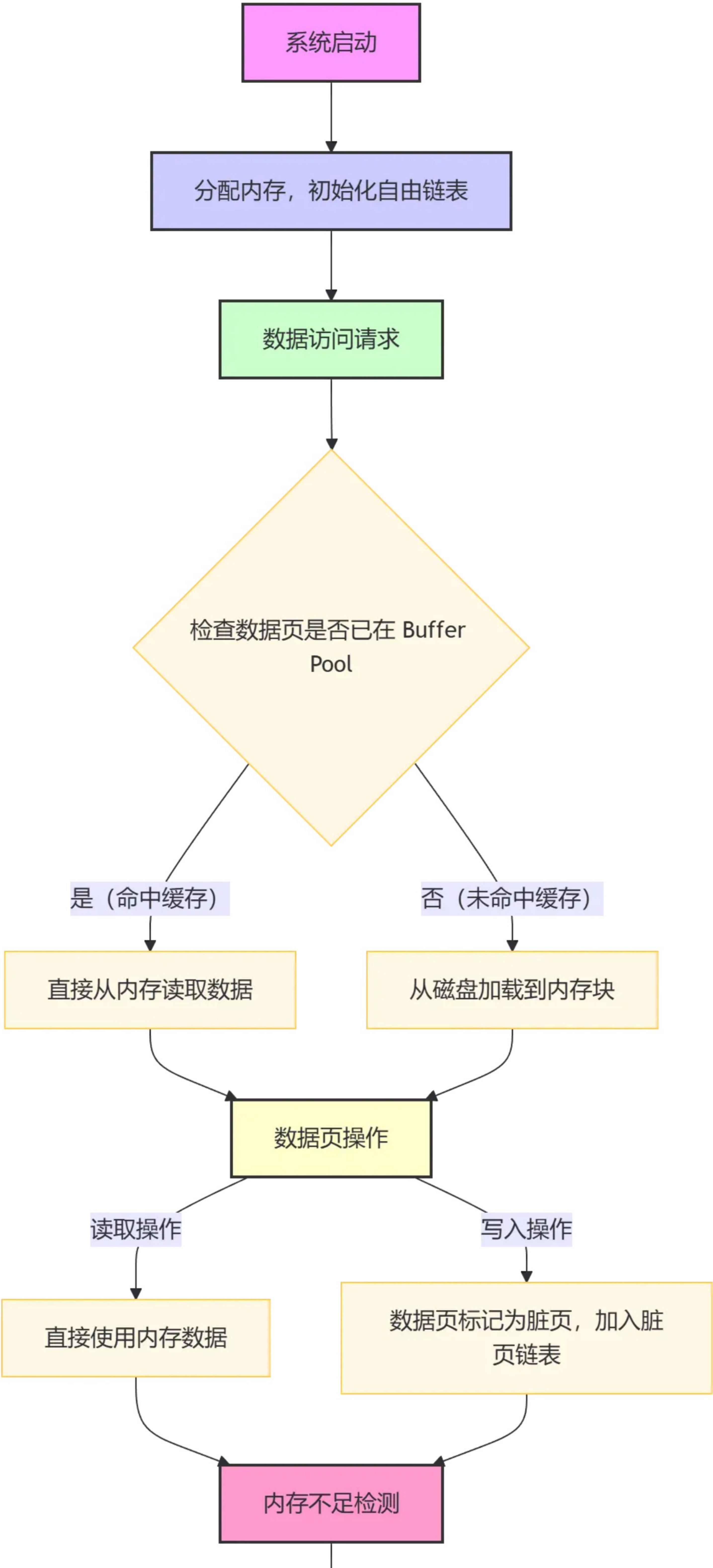
- 数据库系统会通过后台线程定期扫描脏页链表，将脏页写回磁盘，确保数据一致性。

3. Buffer Pool 的组成与运行机制

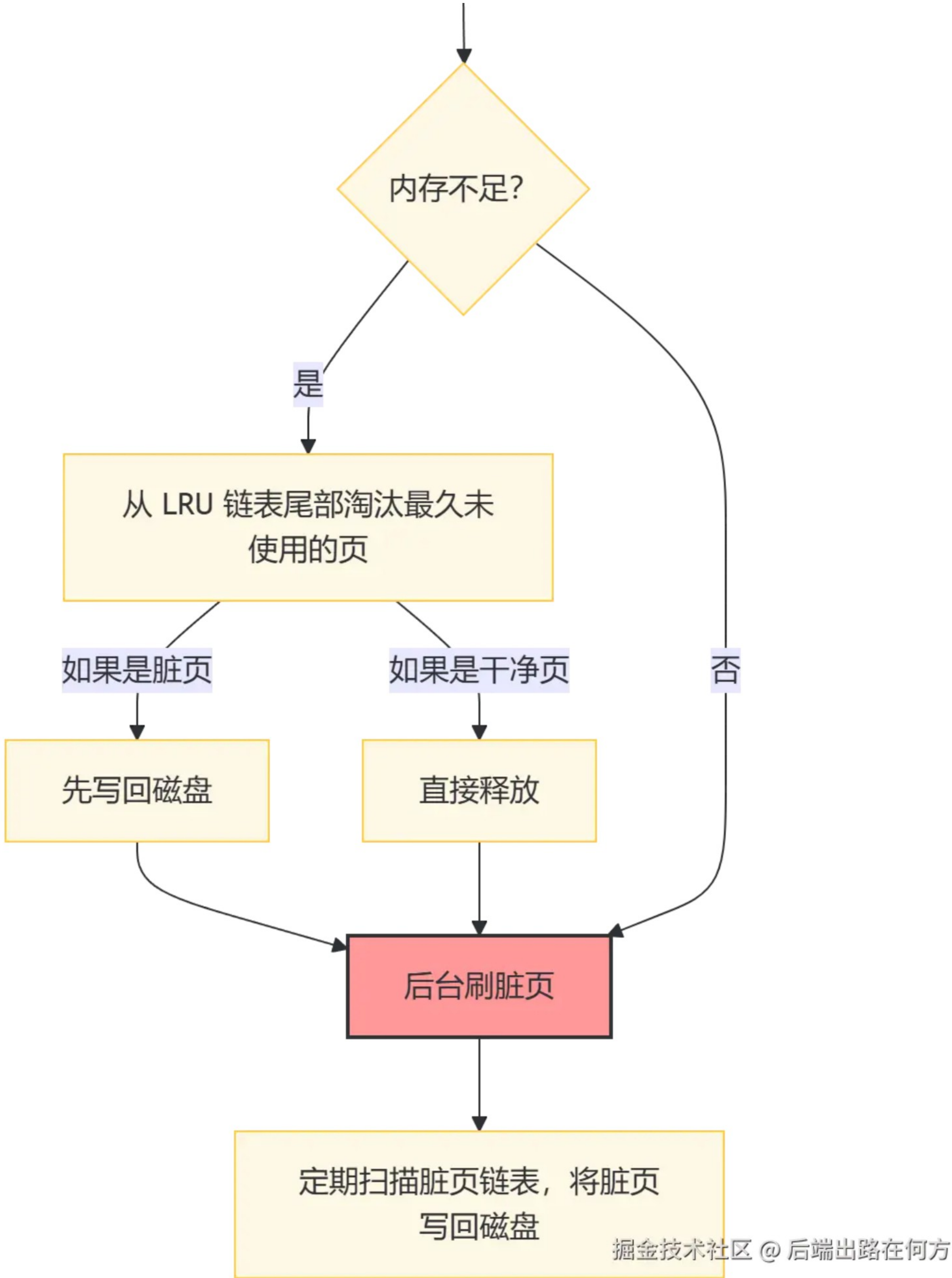


4. Buffer Pool 的运行流程图










### 总结

Buffer Pool 的组成与运行机制是数据库性能优化的核心所在：

- 清晰的内存块管理（通过控制块和链表）。
- 高效的数据页操作（缓存命中与延迟写回）。
- 动态的内存调度（加载、淘汰、刷盘）。

标签： MySQL 数据库 话题： 每天一个知识点

### 本文收录于以下专栏



MySQL

专栏目录


数据库、事物、存储、IO、锁机制、ACID、MVCC、持久化

3 订阅 · 15 篇文章

订阅

上一篇 数据库性能优化之道：Buffer Pool ... 下一篇 数据库性能优化之道：Buffer Pool ...

### 评论 0



登录 / 注册

即可发布评论!



暂无评论数据



为你推荐

谈谈InnoDB核心组件--Buffer Pool

Colors | 3年前 | 👁 534 | 👍 点赞 | 💬 评论

MySQL

MySQL内存架构和索引说明

编程学习网 | 3年前 | 👁 353 | 👍 点赞 | 💬 评论

MySQL | 后端

【MySQL】InnoDB - Buffer Pool 数据页管理

敞开的大门 | 3年前 | 👁 1.5k | 👍 点赞 | 💬 评论

MySQL

Mysql内存组件Buffer-Pool分析（1）

小熙 | 4年前 | 👁 305 | 👍 2 | 💬 评论

MySQL

MySQL 的 BufferPool 是个啥？

PandarSkr | 2年前 | 👁 254 | 👍 1 | 💬 评论

后端

一文详解InnoDB最核心组件Buffer Pool（二）

南山的架构笔记 | 3年前 | 👁 146 | 👍 点赞 | 💬 评论

MySQL

MySQL十六：36张图理解Buffer Pool

星河之码 | 2年前 | 👁 1.6k | 👍 9 | 💬 评论

数据库

【MySQL 02】InnoDB 的 Buffer Pool 学习

晚\_风 | 1年前 | 👁 1.0k | 👍 3 | 💬 评论

后端 | 数据库 | MySQL

MySQL - InnoDB 内存结构解析

政采云技术团队已转... | 1年前 | 👁 1.4k | 👍 29 | 💬 2

MySQL

下饭，深入理解Buffer Pool原理

肖说一下 | 3年前 | 👁 903 | 👍 3 | 💬 评论

MySQL

数据库缓冲池（Buffer Pool）你真的了解吗？

Sma2tArt | 2年前 | 👁 1.6k | 👍 11 | 💬 4

MySQL | 后端 | 面试

Mysql成神之路----InnoDB 的 Buffer Pool

彭阿三 | 4年前 | 👁 423 | 👍 1 | 💬 评论

MySQL | 后端

MySQL如何加速读写速度？来看看Buffer Pool

Mr\_Yao | 1月前 | 👁 63 | 👍 点赞 | 💬 评论

MySQL

Buffer pool详解

聪明小不懂 | 2年前 | 👁 489 | 👍 1 | 💬 评论

后端

MySQL数据库缓冲池——Buffer Pool

来碗鱼粥 | 3年前 | 👁 1.1k | 👍 3 | 💬 评论

数据库