**14.8.3.2 配置多个缓冲池实例**Configuring Buffer Pool Flushing

对于缓冲池在数千兆字节范围内的系统，将缓冲池划分为多个单独的实例可以通过减少争用不同线程读取和写入缓存页面的争用来提高并发性。此功能通常用于[buffer pool](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html#glos_buffer_pool)大小在千兆字节范围内的系统。使用[innodb\_buffer\_pool\_instances](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_buffer_pool_instances)配置选项配置了多个缓冲池实例，您还可以调整[innodb\_buffer\_pool\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_buffer_pool_size)值。

当InnoDB缓冲池很大时，可以通过从内存中检索来满足许多数据请求。您可能会遇到多个线程尝试立即访问缓冲池的瓶颈。您可以启用多个缓冲池以最小化此争用。使用散列函数将存储在缓冲池中或从缓冲池读取的每个页面随机分配给其中一个缓冲池。每个缓冲池 Management 自己的空闲列 table，刷新列 table，LRU 和连接到缓冲池的所有其他数据结构，并受其自己的缓冲池互斥量保护。

要启用多个缓冲池实例，请将innodb\_buffer\_pool\_instances配置选项设置为大于 1(默认)的值，最大为 64(最大)。仅当将innodb\_buffer\_pool\_size设置为 1GB 或更大时，此选项才生效。您指定的总大小将分配给所有缓冲池。为了获得最佳效率，请指定[innodb\_buffer\_pool\_instances](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_buffer_pool_instances)和[innodb\_buffer\_pool\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_buffer_pool_size)的组合，以便每个缓冲池实例至少为 1GB。

**14.8.3.3 防止缓冲池扫描**Making the Buffer Pool Scan Resistant

InnoDB而不是使用严格的[LRU](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html" \l "glos_lru)算法，而是使用一种技术来最小化进入[buffer pool](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html" \l "glos_buffer_pool)且再也不会访问的数据量。目的是确保即使[read-ahead](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html" \l "glos_read_ahead)和[全 table 扫描](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html" \l "glos_full_table_scan)引入了以后可能会访问或可能不会访问的新块，缓冲池中仍会保留频繁访问的(“热”)页。

新读取的块将插入到 LRU 列 table 的中间。默认情况下，所有新读取的页面都插入到 LRU 列 table 末尾的3/8的位置。当页面第一次在缓冲池中访问时，它们将移到列 table 的最前面(最近使用的末尾)。因此，从未访问过的页面永远不会进入 LRU 列 table 的最前面，并且比严格的 LRU 方法更早地“老化”。这种安排将 LRU 列 table 分为两个部分，其中插入点下游的页面被认为是“旧的”，是 LRU 驱逐的理想受害者。

**14.8.3.4 配置 InnoDB 缓冲池预取(预读)** Configuring InnoDB Buffer Pool Prefetching (Read-Ahead)

[read-ahead](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html#glos_read_ahead)请求是一个 I/O 请求，用于异步预取[buffer pool](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html" \l "glos_buffer_pool)中的多个页面，因为预计不久将需要这些页面。请求将所有页面都放在[extent](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html" \l "glos_extent)中。 InnoDB使用两种预读算法来提高 I/O 性能：

**线性** 预读是一项技术，该技术根据 Sequences 访问的缓冲池中的页面来预测不久将需要哪些页面。您可以使用配置参数[innodb\_read\_ahead\_threshold](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_read_ahead_threshold)来调整触发异步读取请求所需的 Sequences 页面访问数，从而控制InnoDB何时执行预读操作。在添加此参数之前，InnoDB仅在读取当前扩展区的最后一页时，才计算是否对整个下一个扩展区发出异步预取请求。

**随机** 预读是一项技术，它根据缓冲池中已有的页面来预测何时可能需要该页面，而不管这些页面的读取 Sequences 如何。如果在缓冲池中找到来自同一扩展区的 13 个连续页面，则InnoDB异步发出请求以预取该扩展区的其余页面。要启用此功能，请将配置变量[innodb\_random\_read\_ahead](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_random_read_ahead)设置为ON。

SHOW ENGINE INNODB STATUS命令显示统计信息，以帮助您评估预读算法的有效性。

**14.8.3.5 配置缓冲池刷新**Configuring Buffer Pool Flushing

InnoDB在后台执行某些任务，包括从缓冲池中清除脏页。脏页是指已被修改但尚未写入磁盘上数据文件的页。

在 MySQL 5.7 中，缓冲池刷新是由页面清理程序线程执行的。页面清理器线程的数量由[innodb\_page\_cleaners](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_page_cleaners)变量控制，该变量的默认值为 4.但是，如果页面清理器线程的数量超过缓冲池实例的数量，则[innodb\_page\_cleaners](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_page_cleaners)会自动设置为与[innodb\_buffer\_pool\_instances](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_buffer_pool_instances)相同的值。

当脏页的百分比达到[innodb\_max\_dirty\_pages\_pct\_lwm](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_max_dirty_pages_pct_lwm)变量定义的低水位标记值时，将启动缓冲池刷新。默认的低水位标记为 0，这会禁用此早期冲洗行为。

[innodb\_max\_dirty\_pages\_pct\_lwm](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_max_dirty_pages_pct_lwm)阈值的目的是控制缓冲池中脏页的百分比，并防止脏页数量达到[innodb\_max\_dirty\_pages\_pct](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_max_dirty_pages_pct)变量(默认值为 75)所定义的阈值。InnoDB积极刷新缓冲池页如果缓冲池中的脏页百分比达到[innodb\_max\_dirty\_pages\_pct](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_max_dirty_pages_pct)阈值。

**14.8.3.6 保存和还原缓冲池状态Saving and Restoring the Buffer Pool State**

为了减少重新启动服务器后的[warmup](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/glossary.html#glos_warm_up)时间，InnoDB在服务器关闭时为每个缓冲池保存了最近使用页面的百分比，并在服务器启动时恢复了这些页面。 [innodb\_buffer\_pool\_dump\_pct](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html" \l "sysvar_innodb_buffer_pool_dump_pct)配置选项定义了最近使用页面的存储百分比。

重新启动繁忙的服务器后，通常会有一个预热期，吞吐量不断增加，这是因为缓冲池中的磁盘页被带回内存中(查询，更新相同数据等)。通过在重新启动之前重新加载缓冲池中的磁盘页面，而不是 awaitDML 操作访问相应的行，可以在启动时还原缓冲池，从而缩短了预热时间。另外，可以批量执行 I/O 请求，从而使整体 I/O 更快。页面加载发生在后台，并且不会延迟数据库启动

**14.8.3.1 配置 InnoDB 缓冲池大小**Configuring InnoDB Buffer Pool Size

您可以在服务器运行时离线(启动时)或在线配置InnoDB缓冲池大小。本节中描述的行为适用于两种方法。有关在线配置缓冲池大小的其他信息，请参阅[在线配置 InnoDB 缓冲池大小](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-buffer-pool-resize.html#innodb-buffer-pool-online-resize)。

当增大或减小[innodb\_buffer\_pool\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_size)时，将按块执行操作。块大小由[innodb\_buffer\_pool\_chunk\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_chunk_size)配置选项定义，该选项的默认值为128M。有关更多信息，请参见[配置 InnoDB 缓冲池块大小](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-buffer-pool-resize.html#innodb-buffer-pool-chunk-size)。

缓冲池大小必须始终等于[innodb\_buffer\_pool\_chunk\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_chunk_size) \* [innodb\_buffer\_pool\_instances](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_instances)或倍数。如果将[innodb\_buffer\_pool\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_size)配置为不等于[innodb\_buffer\_pool\_chunk\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_chunk_size) \* [innodb\_buffer\_pool\_instances](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_instances)的整数倍，则缓冲池大小会自动调整为等于[innodb\_buffer\_pool\_chunk\_size](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_chunk_size) \* [innodb\_buffer\_pool\_instances](https://www.docs4dev.com/docs/zh/mysql/5.7/reference/innodb-parameters.html#sysvar_innodb_buffer_pool_instances)的整数倍。