# MySQL调优

## 数据库调优的目标是什么？

如何确定调优的目标 1. 用户的反馈 2. 日志的分析 3. 服务器资源使用监控 4. 数据库内部状况监控

## 如果要进行调优，都有哪些维度可以选择？

调优对象是整个数据库管理系统，而不仅包括SQL查询

1. 使用适合的DBMS

RDBMS中，如果对事务性处理以及安全性要求高的话，可以选择商业的数据库产品。

采用开源的 MySQL 进行存储，有很多存储引擎可以选择，如果进行事务处理的话可以选择 InnoDB，非事务处理可以选择 MyISAM。

InnoDB支持事务和行级锁，是MySQL默认的存储引擎 MyISAM只支持表级锁，非事务安全性，更适合读取数据库的情况， 小型的应用，如果需要大量的SELECT查询，可以考虑MyISAM。如果是事务处理应用需要选择InnoDB。 两种引擎各有特点，另外在MySQL中不同的数据表也可以选择不同的存储引擎。

1. 优化表设计

* 表结构要尽量遵循第三范式的原则（关于第三范式，我在后面章节会讲）。这样可以让数据结构更加清晰规范，减少冗余字段，同时也减少了在更新，插入和删除数据时等异常情况的发生。
* 如果分析查询应用比较多，尤其是需要进行多表联查的时候，可以采用反范式进行优化。反范式采用空间换时间的方式，通过增加冗余字段提高查询的效率。
* 表字段的数据类型选择，关系到了查询效率的高低以及存储空间的大小。一般来说，如果字段可以采用数值类型就不要采用字符类型；字符长度要尽可能设计得短一些。针对字符类型来说，当确定字符长度固定时，就可以采用 CHAR 类型；当长度不固定时，通常采用 VARCHAR 类型。

数据表的结构设计很基础，也很关键。好的表结构可以在业务发展和用户量增加的情况下依然发挥作用，不好的表结构设计会让数据表变得非常臃肿，查询效率也会降低。

1. 优化逻辑查询

SQL 查询优化，可以分为逻辑查询优化和物理查询优化。逻辑查询优化就是通过改变 SQL 语句的内容让 SQL 执行效率更高效，采用的方式是对 SQL 语句进行等价变换，对查询进行重写。重写查询的数学基础就是关系代数。

SQL 的查询重写包括了子查询优化、等价谓词重写、视图重写、条件简化、连接消除和嵌套连接消除等。

比如我们在讲解 EXISTS 子查询和 IN 子查询的时候，会根据小表驱动大表的原则选择适合的子查询。在 WHERE 子句中会尽量避免对字段进行函数运算，它们会让字段的索引失效。

1. 优化物理查询

物理查询优化是将逻辑查询的内容变成可以被执行的物理操作符，从而为后续执行器的执行提供准备。它的核心是高效地建立索引，并通过这些索引来做各种优化。

但你要知道索引不是万能的，我们需要根据实际情况来创建索引。那么都有哪些情况需要考虑呢？

* 如果数据重复度高，就不需要创建索引。通常在重复度超过 10% 的情况下，可以不创建这个字段的索引。比如性别这个字段（取值为男和女）。
* 要注意索引列的位置对索引使用的影响。比如我们在 WHERE 子句中对索引字段进行了表达式的计算，会造成这个字段的索引失效。
* 要注意联合索引对索引使用的影响。我们在创建联合索引的时候会对多个字段创建索引，这时索引的顺序就很重要了。比如我们对字段 x, y, z 创建了索引，那么顺序是 (x,y,z) 还是 (z,y,x)，在执行的时候就会存在差别。
* 要注意多个索引对索引使用的影响。索引不是越多越好，因为每个索引都需要存储空间，索引多也就意味着需要更多的存储空间。此外，过多的索引也会导致优化器在进行评估的时候增加了筛选出索引的计算时间，影响评估的效率。

查询优化器在对 SQL 语句进行等价变换之后，还需要根据数据表的索引情况和数据情况确定访问路径，这就决定了执行 SQL 时所需要消耗的资源。SQL 查询时需要对不同的数据表进行查询，因此在物理查询优化阶段也需要确定这些查询所采用的路径，具体的情况包括：

* 单表扫描：对于单表扫描来说，我们可以全表扫描所有的数据，也可以局部扫描。
* 两张表的连接：常用的连接方式包括了嵌套循环连接、HASH 连接和合并连接。
* 张表的连接：多张数据表进行连接的时候，顺序很重要，因为不同的连接路径查询的效率不同，搜索空间也会不同。我们在进行多表连接的时候，搜索空间可能会达到很高的数据量级，巨大的搜索空间显然会占用更多的资源，因此我们需要通过调整连接顺序，将搜索空间调整在一个可接收的范围内。

物理查询优化是在确定了逻辑查询优化之后，采用物理优化技术（比如索引等），通过计算代价模型对各种可能的访问路径进行估算，从而找到执行方式中代价最小的作为执行计划。在这个部分中，我们需要掌握的重点是对索引的创建和使用。

1. 使用 Redis 或 Memcached 作为缓存

因为数据都是存放到数据库中，我们需要从数据库层中取出数据放到内存中进行业务逻辑的操作，当用户量增大的时候，如果频繁地进行数据查询，会消耗数据库的很多资源。如果我们将常用的数据直接放到内存中，就会大幅提升查询的效率。

1. 库级优化

如果读和写的业务量都很大，并且它们都在同一个数据库服务器中进行操作，那么数据库的性能就会出现瓶颈，这时为了提升系统的性能，优化用户体验，我们可以采用读写分离的方式降低主数据库的负载，比如用主数据库（master）完成写操作，用从数据库（slave）完成读操作。

除此以外，我们还可以对数据库分库分表。当数据量级达到亿级以上时，有时候我们需要把一个数据库切成多份，放到不同的数据库服务器上，减少对单一数据库服务器的访问压力。如果你使用的是 MySQL，就可以使用 MySQL 自带的分区表功能，当然你也可以考虑自己做垂直切分和水平切分。

采用垂直分表的形式，就是将一张数据表分拆成多张表，采用水平拆分的方式，就是将单张数据量大的表按照某个属性维度分成不同的小表。

但需要注意的是，分拆在提升数据库性能的同时，也会增加维护和使用成本。

## 如何思考和分析数据库调优这件事？

做任何事情之前，我们都需要确认目标。在数据库调优中，我们的目标就是响应时间更快，吞吐量更大。利用宏观的监控工具和微观的日志分析可以帮我们快速找到调优的思路和方式。

虽然每个人的情况都不一样，但我们同样需要对数据库调优这件事有一个整体的认知。在思考数据库调优的时候，可以从三个维度进行考虑。

1. 选择比努力更重要

在进行 SQL 调优之前，可以先选择 DBMS 和数据表的设计方式。

1. 你可以把 SQL 查询优化分成两个部分，逻辑查询优化和物理查询优化

虽然 SQL 查询优化的技术有很多，但是大方向上完全可以分成逻辑查询优化和物理查询优化两大块。逻辑查询优化就是通过 SQL 等价变换提升查询效率，直白一点就是说，换一种查询写法执行效率可能更高。物理查询优化则是通过索引和表连接方式等技术来进行优化，这里重点需要掌握索引的使用。

1. 我们可以通过外援来增强数据库的性能