

# 索引优化

---

## 1. 独立的列

在进行查询时，索引列不能是表达式的一部分，也不能是函数的参数，否则无法使用索引。

例如下面的查询不能使用 actor\_id 列的索引：

```
SELECT actor_id FROM sakila.actor WHERE actor_id + 1 = 5;
```

## 2. 多列索引

在需要使用多个列作为条件进行查询时，使用多列索引比使用多个单列索引性能更好。例如下面的语句中，最好把 actor\_id 和 film\_id 设置为多列索引。

```
SELECT film_id, actor_id FROM sakila.film_actor  
WHERE actor_id = 1 AND film_id = 1;
```

## 3. 索引列的顺序

让选择性最强的索引列放在前面。

索引的选择性是指：不重复的索引值和记录总数的比值。最大值为 1，此时每个记录都有唯一的索引与其对应。选择性越高，每个记录的区分度越高，查询效率也越高。

例如下面显示的结果中 customer\_id 的选择性比 staff\_id 更高，因此最好把 customer\_id 列放在多列索引的前面。

```
SELECT COUNT(DISTINCT staff_id)/COUNT(*) AS staff_id_selectivity,  
COUNT(DISTINCT customer_id)/COUNT(*) AS customer_id_selectivity,  
COUNT(*)  
FROM payment;  
staff_id_selectivity: 0.0001  
customer_id_selectivity: 0.0373  
COUNT(*): 16049
```

## 4. 前缀索引

对于 BLOB、TEXT 和 VARCHAR 类型的列，必须使用前缀索引，只索引开始的部分字符。

前缀长度的选取需要根据索引选择性来确定。

## 5. 覆盖索引

索引包含所有需要查询的字段的价值。

具有以下优点：

- 索引通常远小于数据行的大小，只读取索引能大大减少数据访问量。

- 一些存储引擎（例如 MyISAM）在内存中只缓存索引，而数据依赖于操作系统来缓存。因此，只访问索引可以不使用系统调用（通常比较费时）。
- 对于 InnoDB 引擎，若辅助索引能够覆盖查询，则无需访问主索引。