> 索引是快速搜索的关键。MySQL索引的建立对于MySQL的高效运行是很重要的。下面介绍几种常见的MySQL索引类型。

> 在数据库表中，对字段建立索引可以大大提高查询速度。假如我们创建了一个 mytable表：

```
CREATE TABLE mytable(   ID INT NOT NULL,    username VARCHAR(16) NOT NULL  );
我们随机向里面插入了10000条记录，其中有一条：5555，admin。
在查找username="admin"的记录 SELECT * FROM mytable WHERE username='admin';时，
如果在username上已经建立了索引，MySQL无须任何扫描，即准确可找到该记录。相反，MySQL会扫描
所有记录，即要查询10000条记录。
```

> 索引分单列索引和组合索引。单列索引，即一个索引只包含单个列，一个表可以有多个单列索引，但这不是组合索引。组合索引，即一个索包含多个列。

MySQL索引类型包括：

（1）普通索引

这是最基本的索引，它没有任何限制。普通索引(由关键字KEY或INDEX定义的索引)的唯一任务是加快对数据的访问速度。因此，应该只为那些最经常出现在查询条件(WHERE column = ...)或排序条件(ORDER BY column)中的数据列创建索引。只要有可能，就应该选择一个数据最整齐、最紧凑的数据列(如一个整数类型的数据列)来创建索引。

创建方式：

创建索引

```
CREATE INDEX indexName ON mytable(username(length)); 如果是CHAR，VARCHAR类型，
length可以小于字段实际长度；如果是BLOB和TEXT类型，必须指定 length，下同。
```

修改表结构

```
ALTER mytable ADD INDEX [indexName] ON (username(length))
```

创建表的时候直接指定

```
CREATE TABLE mytable(   ID INT NOT NULL,    username VARCHAR(16) NOT NULL,   INDEX
[indexName] (username(length))   );
```

删除索引的语法：

```
DROP INDEX [indexName] ON mytable;
```

(2）唯一索引

它与前面的普通索引类似，不同的就是：普通索引允许被索引的数据列包含重复的值。而唯一索引列的值必须唯一，但允许有空值。如果是组合索引，则列值的组合必须唯一。

创建方式：

创建索引

```
CREATE UNIQUE INDEX indexName ON mytable(username(length))
```

修改表结构

```
ALTER mytable ADD UNIQUE [indexName] ON (username(length))
```

创建表的时候直接指定

```
CREATE TABLE mytable(   ID INT NOT NULL,    username VARCHAR(16) NOT NULL,
UNIQUE [indexName] (username(length))   );
```

（3）主键索引

```
它是一种特殊的唯一索引，不允许有空值。一般是在建表的时候同时创建主键索引：
```

```
CREATE TABLE mytable(   ID INT NOT NULL,    username VARCHAR(16) NOT NULL,
PRIMARY KEY(ID)   );
```

当然也可以用 ALTER 命令。记住：一个表只能有一个主键。

与之类似的，外键索引 如果为某个外键字段定义了一个外键约束条件，MySQL就会定义一个内部索引来帮助自己以最有效率的方式去管理和使用外键约束条件。

（4）组合索引

> 为了形象地对比单列索引和组合索引，为表添加多个字段：

```
CREATE TABLE mytable(   ID INT NOT NULL,    username VARCHAR(16) NOT NULL,   city
VARCHAR(50) NOT NULL,   age INT NOT NULL  );
```

> 为了进一步榨取MySQL的效率，就要考虑建立组合索引。就是将 name, city, age建到一个索引里：

```
ALTER TABLE mytable ADD INDEX name_city_age (name(10),city,age);
```

建表时，usernname长度为 16，这里用 10。这是因为一般情况下名字的长度不会超过10，这样会加速索引查询速度，还会减少索引文件的大小，提高INSERT的更新速度。

如果分别在 usernname，city，age上建立单列索引，让该表有3个单列索引，查询时和上述的组合索引效率也会大不一样，远远低于我们的组合索引。虽然此时有了三个索引，但MySQL只能用到其中的那个它认为似乎是最有效率的单列索引。

建立这样的组合索引，其实是相当于分别建立了下面三组组合索引： usernname,city,age usernname,city usernname 为什么没有 city，age这样的组合索引呢？这是因为MySQL组合索引“最左前缀”的结果。简单的理解就是只从最左面的开始组合。并不是只要包含这三列的查询都会用到该组合索引，

下面的几个SQL就会用到这个组合索引：

```
SELECT * FROM mytable WHREE username="admin" AND city="郑州"
SELECT * FROM mytable WHREE username="admin"
而下面几个则不会用到：
SELECT * FROM mytable WHREE age=20 AND city="郑州"
SELECT * FROM mytable WHREE city="郑州"
```

（5）建立索引的时机

> 到这里我们已经学会了建立索引，那么我们需要在什么情况下建立索引呢？一般来说，在WHERE和JOIN中出现的列需要建立索引，但也不完全如此，因为MySQL只对<，<=，=，>，>=，BETWEEN，IN，以及某些时候的LIKE才会使用索引。 例如：

```
SELECT t.Name  FROM mytable t LEFT JOIN mytable m    ON t.Name=m.username WHERE
m.age=20 AND m.city='郑州'
```

> 此时就需要对city和age建立索引，由于mytable表的userame也出现在了JOIN子句中，也有对它建立索引的必要。

刚才提到只有某些时候的LIKE才需建立索引。因为在以通配符%和_开头作查询时，MySQL不会使用索引。like操作一般在全文索引中会用到（InnoDB数据表不支持全文索引）。

```
例如下句会使用索引：
SELECT * FROM mytable WHERE username like'admin%'
而下句就不会使用：
SELECT * FROM mytable WHEREt Name like'%admin' 因此，在使用LIKE时应注意以上的区别。
```

（6）索引的不足之处

> 上面都在说使用索引的好处，但过多的使用索引将会造成滥用。因此索引也会有它的缺点：

```
虽然索引大大提高了查询速度，同时却会降低更新表的速度，如对表进行INSERT、UPDATE和DELETE。
因为更新表时，MySQL不仅要保存数据，还要保存索引文件。
```

建立索引会占用磁盘空间的索引文件。一般情况这个问题不太严重，但如果你在一个大表上创建了多种组合索引，索引文件的会膨胀很快。

索引只是提高效率的一个因素，如果你的MySQL有大数据量的表，就需要花时间研究建立最优秀的索引，或优化查询语句。
因此应该只为最经常查询和最经常排序的数据列建立索引。注意，如果某个数据列包含许多重复的内容，为它建立索引就没有太大的实际效果。

从理论上讲，完全可以为数据表里的每个字段分别建一个索引，但MySQL把同一个数据表里的索引总数限制为16个。

（7）使用索引的注意事项

使用索引时，有以下一些技巧和注意事项：

索引不会包含有NULL值的列

只要列中包含有NULL值都将不会被包含在索引中，复合索引中只要有一列含有NULL值，那么这一列对于此复合索引就是无效的。所以我们在数据库设计时不要让字段的默认值为NULL。

使用短索引

对串列进行索引，如果可能应该指定一个前缀长度。例如，如果有一个CHAR(255)的列，如果在前10个或20个字符内，多数值是惟一的，那么就不要对整个列进行索引。

短索引不仅可以提高查询速度而且可以节省磁盘空间和I/O操作。在绝大多数应用里，数据库中的字符串数据大都以各种各样的名字为主，把索引的长度设置为10~15个字符已经足以把搜索范围缩小到很少的几条数据记录了。

索引列排序

MySQL查询只使用一个索引，因此如果where子句中已经使用了索引的话，那么order by中的列是不会使用索引的。因此数据库默认排序可以符合要求的情况下不要使用排序操作；尽量不要包含多个列的排序，如果需要最好给这些列创建复合索引。

like语句操作

一般情况下不鼓励使用like操作，如果非使用不可，如何使用也是一个问题。like “%aaa%” 不会使用索引而like “aaa%”可以使用索引。

不要在列上进行运算

select * from users where YEAR(adddate)<2007; 将在每个行上进行运算，这将导致索引失效而进行全表扫描，因此我们可以改成 select * from users where adddate<‘2007-01-01’;

不使用NOT IN和<>操作

> 对于not in，可以用not exists或者（外联结+判断为空）来代替； 对于<>，用其它相同功能的操作运算代替，如a<>0 改为 a>0 or a<0

（8）查询和索引的优化

> 只有当数据库里已经有了足够多的测试数据时，它的性能测试结果才有实际参考价值。如果在测试数据库里只有几百条数据记录，它们往往在执行完第一条查询命令之后就被全部加载到内存里，这将使后续的查询命令都执行得非常快–不管有没有使用索引。
>
> 只有当数据库里的记录超过了1000条、数据总量也超过了 MySQL服务器上的内存总量时，数据库的性能测试结果才有意义。
>
> 在不确定应该在哪些数据列上创建索引的时候，人们从EXPLAIN SELECT命令那里往往可以获得一些帮助。这其实只是简单地给一条普通的SELECT命令加一个EXPLAIN关键字作为前缀而已。
>
> 有了这个关键字，MySQL将不是去执行那条SELECT命令，而是去对它进行分析。MySQL将以表格的形式把查询的执行过程和用到的索引(如果有的话)等信息列出来。