MySQL 连接的使用：

使用 MySQL 的 JOIN 在两个或多个表中查询数据。可以在 SELECT, UPDATE 和 DELETE 语句中使用 Mysql 的 JOIN 来联合多表查询。

JOIN 按照功能大致分为如下三类：

INNER JOIN（内连接或等值连接）：获取两个表中字段匹配关系的记录；

LEFT JOIN（左连接）：获取左表所有记录，即使右表没有对应匹配的记录；

RIGHT JOIN（右连接）：与 LEFT JOIN 相反，用于获取右表所有记录，即使左表没有对应匹配的记录。

在命令提示符中使用 INNER JOIN（内连接或等值连接）：

使用 MySQL 的 INNER JOIN(也可以省略 INNER 使用 JOIN，效果一样)来连接以上两张表来读取 runoob_tbl 表中所有 runoob_author 字段在 tcount_tbl 表对应的 runoob_count 字段值：

```
mysql> SELECT a.runoob_id, a.runoob_author, b.runoob_count FROM runoob_tbl a
INNER JOIN tcount_tbl b ON a.runoob_author = b.runoob_author;
```

以上 SQL 语句等价于：

```
mysql> SELECT a.runoob_id, a.runoob_author, b.runoob_count FROM runoob_tbl a,
tcount_tbl b WHERE a.runoob_author = b.runoob_author;
```

在命令提示符中使用 LEFT JOIN（左连接）：

MySQL left join 与 join 有所不同。MySQL LEFT JOIN 会读取左边数据表的全部数据，即便右边表无对应数据。以 runoob_tbl 为左表，tcount_tbl 为右表，理解 MySQL LEFT JOIN 的应用：

```
mysql> SELECT a.runoob_id, a.runoob_author, b.runoob_count FROM runoob_tbl a
LEFT JOIN tcount_tbl b ON a.runoob_author = b.runoob_author;
```

在命令提示符中使用 RIGHT JOIN（右连接）：

MySQL RIGHT JOIN 会读取右边数据表的全部数据，即便左边边表无对应数据。以 runoob_tbl 为左表，tcount_tbl 为右表，理解 MySQL RIGHT JOIN 的应用：

```
mysql> SELECT a.runoob_id, a.runoob_author, b.runoob_count FROM runoob_tbl a
RIGHT JOIN tcount_tbl b ON a.runoob_author = b.runoob_author;
```

MySQL NULL 值的使用：

当提供的查询条件字段为 NULL 时，该命令可能就无法正常工作。为了处理这种情况，MySQL 提供了三大运算符：

（1）IS NULL: 当列的值是 NULL,此运算符返回 true；

（2）IS NOT NULL: 当列的值不为 NULL, 运算符返回 true；

（3）<=>：比较操作符（不同于 = 运算符），当比较的的两个值相等或者都为 NULL 时返回 true。

关于 NULL 的条件比较运算是比较特殊的。你不能使用 = NULL 或 != NULL 在列中查找 NULL 值 。在 MySQL 中，NULL 值与任何其它值的比较（即使是 NULL）永远返回 NULL，即 NULL = NULL 返回 NULL 。MySQL 中处理 NULL 使用 **IS NULL** 和 **IS NOT NULL** 运算符。

查找数据表中 runoob_test_tbl 列是否为 NULL，必须使用 IS NULL 和 IS NOT NULL，如下实例：

```
mysql> SELECT * from runoob_test_tbl WHERE runoob_count IS NULL;
mysql> SELECT * from runoob_test_tbl WHERE runoob_count IS NOT NULL;
```

MySQL 正则表达式：

MySQL 可以通过 LIKE ...% 来进行模糊匹配。MySQL 同样也支持其他正则表达式的匹配， MySQL 中使用 REGEXP 操作符来进行正则表达式匹配。

模式	描述
^	匹配输入字符串的开始位置。如果设置了 RegExp 对象的 Multiline 属性，^ 也匹配 '\n' 或 '\r' 之后的位置。
$	匹配输入字符串的结束位置。如果设置了RegExp 对象的 Multiline 属性，$ 也匹配 '\n' 或 '\r' 之前的位置。
.	匹配除 "\n" 之外的任何单个字符。要匹配包括 '\n' 在内的任何字符，请使用像 '[.\n]' 的模式。
[...]	字符集合。匹配所包含的任意一个字符。例如， '[abc]' 可以匹配 "plain" 中的 'a'。
[^...]	负值字符集合。匹配未包含的任意字符。例如， '[^abc]' 可以匹配 "plain" 中的'p'。
p1|p2|p3	匹配 p1 或 p2 或 p3。例如，'z|food' 能匹配 "z" 或 "food"。'(z|f)ood' 则匹配 "zood" 或 "food"。
*	匹配前面的子表达式零次或多次。例如，zo* 能匹配 "z" 以及 "zoo"。* 等价于{0,}。
+	匹配前面的子表达式一次或多次。例如，'zo+' 能匹配 "zo" 以及 "zoo"，但不能匹配 "z"。+ 等价于 {1,}。
{n}	n 是一个非负整数。匹配确定的 n 次。例如，'o{2}' 不能匹配 "Bob" 中的 'o'，但是能匹配 "food" 中的两个 o。
{n,m}	m 和 n 均为非负整数，其中n <= m。最少匹配 n 次且最多匹配 m 次。

实例：

查找 name 字段中以'st'为开头的所有数据：

```
mysql> SELECT name FROM person_tbl WHERE name REGEXP '^st';
```

查找 name 字段中以'ok'为结尾的所有数据：

```
mysql> SELECT name FROM person_tbl WHERE name REGEXP 'ok$';
```

查找 name 字段中包含'mar'字符串的所有数据：

```
mysql> SELECT name FROM person_tbl WHERE name REGEXP 'mar';
```

查找 name 字段中以元音字符开头或以'ok'字符串结尾的所有数据：

```
mysql> SELECT name FROM person_tbl WHERE name REGEXP '^[aeiou]|ok$';
```