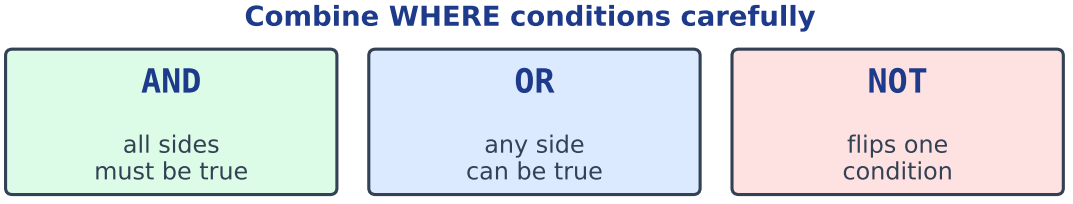
# 筛选与逻辑

SQL Reference

## AND、OR、NOT

用 AND、OR、NOT 组合条件。AND 需要各边都为真;OR 需要任意一边为真;NOT 把一个条件取反。用括号来设定优先级 (precedence)。

```
CREATE TABLE student (id INTEGER, name TEXT, form TEXT, score INTEGER);
INSERT INTO student VALUES
↪  (1,'Mei','11A',88),(2,'Sam','11B',71),(3,'Ana','11A',95);
SELECT name FROM student WHERE form = '11A' AND score >= 90;
```




Exam tip: use brackets for precedence; AND is stricter than OR

*AND* 要求全部为真;*OR* 只要任一为真;*NOT* 把一个条件反过来

## LIKE、IN 与 BETWEEN

LIKE 匹配一个文本模式 (pattern):% 代表任意文本,_ 代表一个字符 — 它们是通配符 (wildcard)。

```
CREATE TABLE student (id INTEGER, name TEXT, form TEXT, score INTEGER);
INSERT INTO student VALUES
↪  (1,'Mei','11A',88),(2,'Sam','11B',71),(3,'Ana','11A',95);
SELECT name FROM student WHERE name LIKE 'M%';        -- starts with M
```

| 模式 | 匹配 |
|---|---|
| 'M%' | 以 M 开头 |
| '%a' | 以 a 结尾 |
| '%an%' | 包含 “an” |
| 'M_i' | M, 接恰好一个字符, 再接 i |

IN 匹配一个值的集合 (set);BETWEEN 匹配一个范围 (range)(两端都包含)。

```
CREATE TABLE student (id INTEGER, name TEXT, form TEXT, score INTEGER);
INSERT INTO student VALUES
↪  (1,'Mei','11A',88),(2,'Sam','11B',71),(3,'Ana','11A',95);
SELECT name, score FROM student
WHERE form IN ('11A', '11C') AND score BETWEEN 80 AND 100;
```

## NULL 空值

NULL 表示缺失值 (missing value)— 这个单元格里什么都没存。NULL 不是 0, 也不是空字符串。用 = 或 <> 的比较永远匹配不到它; 要用 IS NULL 或 IS NOT NULL 来判断。

```
CREATE TABLE student (id INTEGER, name TEXT, score INTEGER);
INSERT INTO student VALUES (1, 'Mei', 88), (2, 'Sam', NULL), (3, 'Ana',
↪  95);
SELECT name FROM student WHERE score IS NULL;
```

```
CREATE TABLE student (id INTEGER, name TEXT, score INTEGER);
INSERT INTO student VALUES (1, 'Mei', 88), (2, 'Sam', NULL), (3, 'Ana',
↪  95);
SELECT name, score FROM student WHERE score IS NOT NULL;
```

- WHERE score = NULL 一行都不会返回 — 连 Sam 那一行也不会。
- NULL 参与算术仍是 NULL: 对 Sam 来说 score + 5 还是 NULL。

### 常见错误

- 判断空值用 IS NULL, 绝不能用 = NULL。
- 在 LIKE 中,% 匹配任意文本,_ 匹配一个字符:'A%' 表示 “ 以 A 开头 ”。
- IN (1, 2, 3) 比一串 OR 更简洁;BETWEEN a AND b 两端都包含。