## 十五、Comparison and Logical Operators（比较与逻辑运算符）

### 1. Comparison Operators（比较运算符）

Comparison operators are used to compare values in SQL statements.  
比较运算符用于在 SQL 查询中比较不同的值。  
  
= Equal to  
!= or <> Not equal to  
> Greater than  
< Less than  
>= Greater than or equal to  
<= Less than or equal to  
  
Examples:  
SELECT \* FROM books WHERE released\_year = 2010;  
SELECT \* FROM books WHERE released\_year > 2010;  
SELECT \* FROM books WHERE author\_lname != 'Smith';  
  
= 检查是否相等，!= 或 <> 检查不相等，> < >= <= 分别用于比较大小。

### 2. BETWEEN（范围比较）

BETWEEN checks if a value is within a specific range (inclusive).  
BETWEEN 用于判断某个值是否在给定的范围内（包含边界值）。  
  
Example:  
SELECT \* FROM books WHERE released\_year BETWEEN 2000 AND 2010;  
  
相当于 released\_year >= 2000 AND released\_year <= 2010。

### 3. IN（集合比较）

IN checks if a value matches any value in a list.  
IN 用于匹配多个可能的值。  
  
Example:  
SELECT \* FROM books WHERE author\_lname IN ('Smith', 'Brown', 'Lee');  
  
IN 适合用于多个特定匹配，而不是多重 OR 条件。

### 4. IS NULL / IS NOT NULL

IS NULL tests if a column contains NULL values.  
IS NOT NULL tests if the column is not NULL.  
  
Example:  
SELECT \* FROM books WHERE author\_lname IS NULL;  
SELECT \* FROM books WHERE author\_lname IS NOT NULL;  
  
NULL 表示缺失或未知值，不能用 = NULL 比较，必须用 IS NULL。

### 5. LIKE（模糊匹配）

LIKE allows pattern matching using wildcards.  
LIKE 用于模糊匹配字符串。  
  
Patterns:  
% — any number of characters  
\_ — one single character  
  
Example:  
SELECT \* FROM books WHERE title LIKE '%love%';  
SELECT \* FROM books WHERE author\_fname LIKE 'A%';  
SELECT \* FROM books WHERE title LIKE '\_at';  
  
LIKE 常用于搜索包含特定模式的文本。

### 6. Logical Operators（逻辑运算符）

Logical operators combine multiple conditions.  
逻辑运算符用于组合多个查询条件。  
  
AND — Returns true if all conditions are true.  
OR — Returns true if at least one condition is true.  
NOT — Negates a condition.  
  
Examples:  
SELECT \* FROM books WHERE released\_year > 2010 AND author\_lname = 'Smith';  
SELECT \* FROM books WHERE released\_year < 2000 OR pages > 500;  
SELECT \* FROM books WHERE NOT author\_lname = 'King';  
  
AND 同时满足多个条件，OR 满足其中之一，NOT 表示取反。

### 7. Combining Conditions（组合条件）

You can combine multiple comparison and logical operators for complex filtering.  
可以将多个比较运算符与逻辑运算符结合，实现复杂筛选。  
  
Example:  
SELECT \* FROM books  
WHERE (released\_year > 2000 AND released\_year < 2020)  
AND (author\_lname IN ('Smith', 'Lee'))  
AND pages BETWEEN 200 AND 800;  
  
使用括号 () 来明确条件优先级。

### 8. ORDER of Evaluation（逻辑运算符优先级）

SQL evaluates logical operators in this order:  
SQL 逻辑运算符的执行优先级如下：  
  
1. NOT  
2. AND  
3. OR  
  
Example:  
SELECT \* FROM books WHERE NOT author\_lname = 'King' AND released\_year > 2000;  
  
NOT 优先级最高，其次是 AND，最后是 OR。

### 9. CASE Expression（条件表达式）

CASE returns a value based on conditional logic, similar to IF/ELSE.  
CASE 根据条件返回不同结果，类似 IF/ELSE。  
  
-- Searched CASE（常用写法）  
SELECT title,  
 CASE  
 WHEN pages >= 800 THEN 'Long'  
 WHEN pages >= 400 THEN 'Medium'  
 ELSE 'Short'  
 END AS length\_label  
FROM books;  
  
上述查询按照页数区分“Long/Medium/Short”。  
  
-- Simple CASE（等值匹配）  
SELECT title,  
 CASE author\_lname  
 WHEN 'King' THEN 'Famous'  
 WHEN 'Lee' THEN 'Popular'  
 ELSE 'Other'  
 END AS author\_tag  
FROM books;  
  
等值匹配根据列值直接映射标签。  
  
-- CASE in ORDER BY（自定义排序）  
SELECT title, released\_year  
FROM books  
ORDER BY CASE  
 WHEN released\_year >= 2020 THEN 1  
 WHEN released\_year >= 2010 THEN 2  
 ELSE 3  
 END, title;  
  
在 ORDER BY 中使用 CASE 可实现自定义排序优先级。  
  
-- CASE with aggregates（与聚合结合）  
SELECT  
 SUM(CASE WHEN released\_year >= 2015 THEN 1 ELSE 0 END) AS books\_since\_2015,  
 SUM(CASE WHEN pages >= 500 THEN 1 ELSE 0 END) AS big\_books  
FROM books;  
  
通过在 SUM/COUNT 中嵌套 CASE 可实现条件汇总统计。  
  
要点：  
- 以 END 结束；可选 ELSE（未匹配时返回 NULL 或指定默认值）。  
- 优先使用 searched CASE（WHEN 条件）更灵活；simple CASE（等值匹配）用于精确等值场景。