Database Programming with SQL

2-3
Comparison Operators
Objectives

This lesson covers the following objectives:

• Apply the proper comparison operator to return a desired result

• Demonstrate proper use of BETWEEN, IN, and LIKE conditions to return a desired result

• Distinguish between zero and NULL, the latter of which is unavailable, unassigned, unknown, or inapplicable

• Explain the use of comparison conditions and NULL
Purpose

• We use comparisons in everyday conversation without really thinking about it.
  – "I can meet you BETWEEN 10:00 a.m. and 11:00 a.m."
  – "I'm looking for a pair of jeans LIKE the ones you are wearing."
  – "If I remember correctly, the best concert seats are IN rows 100, 200, and 300."
Purpose

• The need to express these types of comparisons also exists in SQL.
• Comparison conditions are used to find data in a table meeting certain conditions.
• Being able to formulate a SELECT clause to return specific data is a powerful feature of SQL.
Comparison Operators

• You are already familiar with the comparison operators such as equal to (=), less than (<), and greater than (>).

• SQL has other operators that add functionality for retrieving specific sets of data.

• These include:
  – BETWEEN...AND
  – IN
  – LIKE
BETWEEN...AND

• The BETWEEN...AND operator is used to select and display rows based on a range of values.

• When used with the WHERE clause, the BETWEEN...AND condition will return a range of values between and inclusive of the specified lower and upper limits.
BETWEEN...AND

• Note in the example from the Employees database, the values returned include the lower-limit value and the upper-limit value.

• Values specified with the BETWEEN condition are said to be inclusive.

• Note also that the lower-limit value must be listed first.

```sql
SELECT last_name, salary
FROM employees
WHERE salary BETWEEN 9000 AND 11000;
```

• Note that the output included the lower-limit and upper-limit values.

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zlotkey</td>
<td>10500</td>
</tr>
<tr>
<td>Abel</td>
<td>11000</td>
</tr>
<tr>
<td>Hunold</td>
<td>9000</td>
</tr>
</tbody>
</table>
BETWEEN...AND

• Using BETWEEN...AND is the same as using the following expression:

  \[
  \text{WHERE} \quad \text{salary} \geq 9000 \quad \text{AND} \quad \text{salary} \leq 11000;
  \]

• In fact, there is no performance benefit in using one expression over the other.

• We use BETWEEN...AND for simplicity in reading the code.
IN

- The IN condition is also known as the "membership condition."
- It is used to test whether a value is IN a specified set of values.
- For example, IN could be used to identify students whose identification numbers are 2349, 7354, or 4333 or people whose international phone calling code is 1735, 82, or 10.

```sql
SELECT city, state_province, country_id
FROM locations
WHERE country_id IN('UK', 'CA');
```

<table>
<thead>
<tr>
<th>CITY</th>
<th>STATE_PROVINCE</th>
<th>COUNTRY_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toronto</td>
<td>Ontario</td>
<td>CA</td>
</tr>
<tr>
<td>Oxford</td>
<td>Oxford</td>
<td>UK</td>
</tr>
</tbody>
</table>
IN

- In this example, the WHERE clause could also be written as a set of OR conditions:

```sql
SELECT city, state_province, country_id
FROM locations
WHERE country_id IN('UK', 'CA');
...
WHERE country_id = 'UK' OR country_id = 'CA';
```

- As with BETWEEN...AND, the IN condition can be written using either syntax just as efficiently.
LIKE

• Have you ever gone shopping to look for something that you saw in a magazine or on television but you weren't sure of the exact item?

• It's much the same with database searches.

• A manager may know that an employee's last name starts with "S" but doesn't know the employee's entire name.

• Fortunately, in SQL, the LIKE condition allows you to select rows that match either characters, dates, or number patterns.

• Two symbols -- the (%) and the underscore (_) -- called wildcard characters, can be used to construct a search string.
LIKE

• The percent (%) symbol is used to represent any sequence of zero or more characters.
• The underscore (_) symbol is used to represent a single character.
• In the example shown below, all employees with last names beginning with any letter followed by an "o" and then followed by any other number of letters will be returned.

```sql
SELECT last_name
FROM employees
WHERE last_name LIKE '_o%';
```

<table>
<thead>
<tr>
<th>LAST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kochhar</td>
</tr>
<tr>
<td>Lorentz</td>
</tr>
<tr>
<td>Mourgos</td>
</tr>
</tbody>
</table>
LIKE

```sql
SELECT last_name
FROM employees
WHERE last_name LIKE '_o%';
```

• Which of the following last names could have been returned from the above query?
  1. Sommersmith
  2. Oog
  3. Fong
  4. Mo
LIKE

• One additional option that is important:
  – When you need to have an exact match for a string that has a % or _ character in it, you will need to indicate that the % or the _ is not a wildcard but is part of the item you're searching for.
LIKE

• The ESCAPE option can be used to indicate that the _ or % is part of the name, not a wildcard value.

• For example, if we wanted to retrieve an employee JOB_ID from the employees table containing the pattern _R, we would need to use an escape character to say we are searching for an underscore, and not just any one character.

```sql
SELECT last_name, job_id
FROM EMPLOYEES
WHERE job_id LIKE '%
R%' ESCAPE '\'
```

• This example uses the backslash '\' as the escape character, but any character can be used.
LIKE

• Without the ESCAPE option, all employees that have an R in their JOB_ID would be returned.

```
SELECT last_name, job_id
FROM EMPLOYEES
WHERE job_id LIKE '%%_R%'
```
IS NULL, IS NOT NULL

• Remember NULL?
• It is the value that is unavailable, unassigned, unknown, or inapplicable.
• Being able to test for NULL is often desirable.
• You may want to know all the dates in June that, right now, do not have a concert scheduled.
• You may want to know all of the clients who do not have phone numbers recorded in your database.
IS NULL, IS NOT NULL

• The IS NULL condition tests for unavailable, unassigned, or unknown data.

• IS NOT NULL tests for data that is available in the database.

• In the example on the next slide, the WHERE clause is written to retrieve all the last names of those employees who do not have a manager.
IS NULL, IS NOT NULL

• Employee King is the President of the company, so has no manager.

    SELECT last_name, commission_pct
    FROM employees
    WHERE commission_pct IS NOT NULL;

    | LAST_NAME | COMMISSION_PCT |
    |-----------|----------------|
    | Zlotkey   | .2             |
    | Abel      | .3             |
    | Taylor    | .2             |
    | Grant     | .15            |

• IS NOT NULL returns the rows that have a value in the commission_pct column.
Terminology

Key terms used in this lesson included:

- BETWEEN...AND
- IN
- LIKE
- IS NULL
- IS NOT NULL
Summary

In this lesson, you should have learned how to:

• Apply the proper comparison operator to return a desired result

• Demonstrate proper use of BETWEEN, IN, and LIKE conditions to return a desired result

• Distinguish between zero and NULL, the latter of which is unavailable, unassigned, unknown, or inapplicable

• Explain the use of comparison conditions and NULL