
ORACLE
Academy

Database Foundations

6-7

Restricting Data Using WHERE

ORACLE
Academy

Objectives

- This lesson covers the following objectives:
  - Limit rows with:
    - WHERE clause
    - Comparison operators using =, <=,>=, <>,>,<, !=,^=, BETWEEN,IN,LIKE and NULL conditions
    - Logical conditions using AND,OR and NOT operators
  - Describe the rules of precedence for operators in an expression

Limiting Rows Using Selection (WHERE)

EMPLOYEES

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	JOB_ID	SALARY	DEPARTMENT_ID
100	Steven	King	AD_PRES	24000	90
101	Neena	Kochhar	AD_VP	17000	90
102	Lex	De Haan	AD_VP	17000	90
200	Jennifer	Whalen	AD_ASST	4400	10
205	Shelley	Higgins	AC_MGR	12000	110
206	William	Gietz	AC_ACCOUNT	8300	110
149	Eleni	Zlotkey	SA_MAN	10500	80
174	Ellen	Abel	SA_REP	11000	80

...

"retrieve all employees in department 90"

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	JOB_ID	SALARY	DEPARTMENT_ID
100	Steven	King	AD_PRES	24000	90
101	Neena	Kochhar	AD_VP	17000	90
102	Lex	De Haan	AD_VP	17000	90

Limiting the Rows That Are Selected

- Restrict the rows that are returned by using the WHERE clause:

```
SELECT *|{[DISTINCT] column|expression [alias],...}
FROM   table
[WHERE logical expression(s)];
```

- If the logical expression evaluates to true, the row meeting the condition is returned
- The WHERE clause follows the FROM clause

In the syntax:

- WHERE restricts the query to rows that meet a condition.
- *logical expression* consists of column names, constants, and a comparison operator. It specifies a combination of one or more expressions and Boolean operators, and returns a value of TRUE, FALSE, or UNKNOWN.

The WHERE clause can compare values in columns, literals, arithmetic expressions, or functions. It consists of three elements:

- Column name
- Comparison condition
- Column name, constant, or list of values

Using the WHERE Clause

- Retrieve all employees in department 90

```
SELECT employee_id, last_name, job_id, department_id
FROM   employees
WHERE  department_id = 90;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Note: You cannot use a column alias in the WHERE clause.

Character Strings and Dates

- Character strings and date values are enclosed in single quotation marks
- Character values are case-sensitive, and date values are format-sensitive

```
SELECT last_name, job_id, department_id
FROM   employees
WHERE  last_name = 'Whalen';
```

Character Strings and Dates

- The default date display format is DD-Mon-YYYY

```
SELECT last_name
FROM   employees
WHERE  hire_date = '29-Jan-2000';
```

Comparison Operators

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to
BETWEEN…AND…	Between two values (inclusive)
IN (set)	Match any of a list of values
LIKE	Match a character pattern
IS NULL	Is a null value

Note: The symbols != and ^= can also represent the not equal to condition.

Using Comparison Operators

- Retrieve records from the EMPLOYEES table where the salary is less than or equal to $3,000

```
SELECT last_name, salary
FROM   employees
WHERE  salary <= 3000;
```

LAST_NAME	SALARY
Matos	2600
Vargas	2500

Another example of using comparison operators :

SELECT last_name, salary
FROM employees
WHERE hire_date > '01-JAN-1998';

Note : when using date or character datatypes the literal must be in single quotes.

Range Conditions: BETWEEN Operator

- Use the BETWEEN operator to display rows based on a range of values:

```
SELECT last_name, salary
FROM   employees
WHERE  salary BETWEEN 2500 AND 3500 ;
```

Lower limit

Upper limit

LAST_NAME	SALARY
Rajs	3500
Davies	3100
Matos	2600
Vargas	2500

** Note – when using BETWEEN lower value must be specified first

Case Scenario: Retrieving Data

Faculty

```
SELECT *
FROM book_transactions
WHERE tran_date
BETWEEN '01-Jan-2017'
AND '28-Feb-2017';
```

ID	TRAN_DATE	TYPE	BOOK_ID	MEMBER_ID
0D0001	05-Jan-2017	out	1	111
0D0002	02-Feb-2017	out	2	111

Student

Successful retrieval of data

Membership Conditions: IN Operator

- Use the IN operator to test for values in a list:

```
SELECT employee_id, last_name, salary, manager_id
FROM   employees
WHERE  manager_id IN (100, 101, 201) ;
```

EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
101	Kochhar	17000	100
102	De Haan	17000	100
149	Zlotkey	10500	100
124	Mourgos	5800	100
201	Hartstein	13000	100

...

** Note: items in list can be in any order

Another example of using IN operator :

SELECT last_name, salary
FROM employees
WHERE job_id IN ('SA_MAN' , 'SA_REP');

Notes : When using date or character datatypes the literal must be in single quotes.

Column used to restrict rows does not have to be in SELECT clause.

Membership Conditions: NOT IN Operator

- Use the NOT IN operator to test for values not in a list:

```
SELECT employee_id, last_name, salary, manager_id
FROM   employees
WHERE  department_id NOT IN (60, 90, 100) ;
```

EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
200	Whalen	4400	101
205	Higgins	12000	101
206	Gietz	8300	205
149	Zlotkey	10500	100
174	Abel	11000	149
176	Taylor	8600	149

...

Another example of using NOT IN operator :

```
SELECT last_name, salary
FROM   employees
WHERE job_id NOT IN ('SA_MAN' , 'SA_REP');
```

Notes : When using date or character datatypes the literal must be in single quotes.

Column used to restrict rows does not have to be in SELECT clause.

Pattern Matching: LIKE Operator

- Use the LIKE operator to perform wildcard searches of valid search string values
- Search conditions can contain literal characters or numbers:
  - % denotes zero or more characters
  - _ denotes one character

```
SELECT   first_name
FROM     employees
WHERE    first_name LIKE 'S%' ;
```

FIRST_NAME
Shelley
Steven

In the slide example, the SELECT statement returns the first name from the EMPLOYEES table for any employee whose first name begins with the letter "S." (Notice the uppercase "S.“) Consequently, names beginning with a lowercase "s" are not returned.

The LIKE operator can be used as a shortcut for some BETWEEN comparisons. The following example displays the last names and hire dates of all employees who joined between January 2005 and December 2005:

```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date LIKE '%05';
```

Combining Wildcard Characters

- You can combine the two wildcard characters (%, _) with literal characters for pattern matching:

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

LAST_NAME
Kochhar
Lorentz
Mourgos

The ESCAPE identifier identifies the backslash (\) as the escape character. In the SQL statement, the escape character precedes the underscore (_) and causes the Oracle server to interpret the underscore literally.

Combining Wildcard Characters

- You can use the ESCAPE identifier to search for the actual % and _ symbols

```
SELECT employee_id, last_name, job_id
FROM   employees
WHERE  job_id LIKE '%SA\_%' ESCAPE '\';
```

- This will return records with SA_ in their job_id

EMPLOYEE_ID	LAST_NAME	JOB_ID
149	Zlotkey	SA_MAN
174	Abel	SA_REP
176	Taylor	SA_REP
178	Grant	SA_REP

The ESCAPE identifier identifies the backslash (\) as the escape character. In the SQL statement, the escape character precedes the underscore (_) and causes the Oracle server to interpret the underscore literally.

Project Exercise 1

- DFo_6_7_1_Project
  - Oracle Baseball League Store Database
  - Limit rows using WHERE:
    - BETWEEN, IN, NOT IN, LIKE, Wildcard Operators

Using the NULL Conditions

- Test for nulls with the IS NULL or IS NOT NULL operators:

```
SELECT last_name, manager_id
FROM   employees
WHERE  manager_id IS NULL;
```

LAST_NAME	MANAGER_ID
King	-

- You cannot test with = because a null cannot be equal or unequal to any value

Defining Conditions Using the Logical Operators

- A logical condition combines the result of two component conditions to produce a single result based on those conditions or if using NOT it inverts the result of a single condition

Operator	Meaning
AND	Returns TRUE if both component conditions are TRUE
OR	Returns TRUE if either component condition is TRUE
NOT	Returns TRUE if the condition is FALSE Returns FALSE if the condition is TRUE

Using the AND Operator

- AND requires both component conditions to be true:

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
AND    job_id LIKE '%MAN%' ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
149	Zlotkey	SA_MAN	10500
201	Hartstein	MK_MAN	13000

- Note: All character searches are case sensitive and must be enclosed in quotation marks

Using the OR Operator

- OR requires either component condition to be true:

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
OR     job_id LIKE '%MAN%' ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
100	King	AD_PRES	24000
101	Kochhar	AD_VP	17000
102	De Haan	AD_VP	17000
205	Higgins	AC_MGR	12000
149	Zlotkey	SA_MAN	10500

...

Using the NOT Operator

- NOT reverses the value of the condition:

```
SELECT last_name, job_id
FROM employees
WHERE job_id NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP') ;
```

LAST_NAME	JOB_ID
King	AD_PRES
Kochhar	AD_VP
De Haan	AD_VP
Whalen	AD_ASST
Higgins	AC_MGR
Gietz	AC_ACCOUNT
Zlotkey	SA_MAN

...

The NOT operator can also be used with other SQL operators, such as BETWEEN, LIKE, and NULL. Here are some examples:

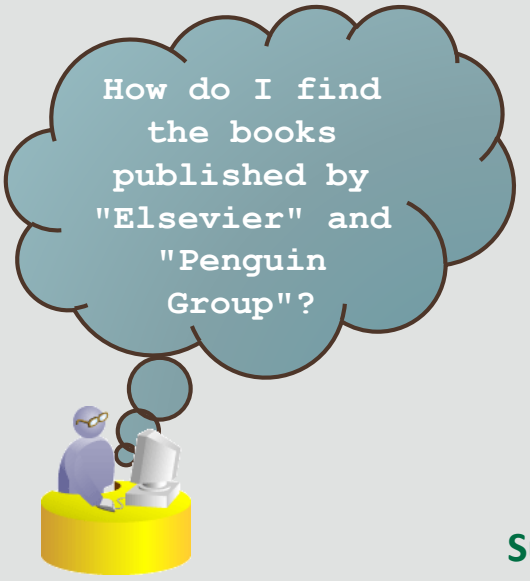
... WHERE job_id NOT IN ('AC_ACCOUNT', 'AD_VP')

... WHERE salary NOT BETWEEN 10000 AND 15000

... WHERE last_name NOT LIKE '%A%'

... WHERE commission_pct IS NOT NULL

Case Scenario: Retrieving Data Using Logical Operators

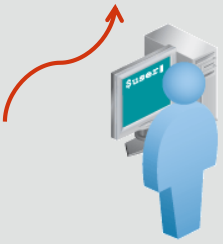


Faculty

```
SELECT ID, TITLE
FROM BOOKS
WHERE PUBLISHER_ID = '10'
OR PUBLISHER_ID = '30';
```

ID	TITLE
1	War and Peace
3	An Unsocial Socialist

Successful retrieval of the book details



Student

ORACLE Academy

Rules of Precedence

Precedence	Operator
1	Arithmetic operators
2	Concatenation operator
3	Comparison conditions
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	Not equal to
7	NOT logical operator
8	AND logical operator
9	OR logical operator

Use parentheses to override rules of precedence

Rules of Precedence

```
SELECT last_name, job_id, salary
FROM   employees
WHERE  job_id = 'SA_REP'
OR     job_id = 'AD_PRES'
AND    salary > 15000;
```

Precedence of the AND Operator

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000
Abel	SA_REP	11000
Taylor	SA_REP	8600
Grant	SA_REP	7000

There are two conditions in the slide example for precedence of the AND operator.

- The first condition is that the job ID is AD_PRES, and the salary is greater than $15,000.
- The second condition is that the job ID is SA_REP.

Therefore, the SELECT statement reads as "Select the row if an employee is a president and earns more than $15,000, *or* if the employee is a sales representative.“

Rules of Precedence

```
SELECT last_name, job_id, salary
FROM employees
WHERE (job_id = 'SA_REP'
OR     job_id = 'AD_PRES')
AND    salary > 15000;
```

Parentheses

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000

There are two conditions in the slide example for parentheses.

- The first condition is that the job ID is AD_PRES or SA_REP.
- The second condition is that the salary is greater than $15,000.

Therefore, the SELECT statement reads as "Select the row if an employee is a president or a sales representative, *and* if the employee earns more than $15,000."

Project Exercise 2

- DFo_6_7_2_Project
  - Oracle Baseball League Store Database
  - Limit rows using WHERE:
    - NULL, AND, OR and NOT Equal to Operators

Summary

- In this lesson, you should have learned how to:
  - Limit rows with:
    - WHERE clause
    - Comparison operators using =, <=,>=, <>,>,<, !=,^=, BETWEEN,IN,LIKE and NULL conditions
    - Logical conditions using AND,OR and NOT operators
  - Describe the rules of precedence for operators in an expression


ORACLE
Academy