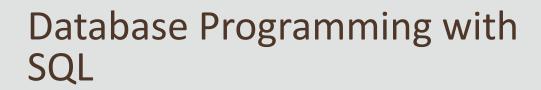
# ORACLE Academy



10-3

**Multiple-Row Subqueries** 





# **Objectives**

- This lesson covers the following objectives:
  - Correctly use the comparison operators IN, ANY, and ALL in multiple-row subqueries
  - Construct and execute a multiple-row subquery in the WHERE clause or HAVING clause
  - Describe what happens if a multiple-row subquery returns a null value
  - Understand when multiple-row subqueries should be used, and when it is safe to use a single-row subquery
  - Distinguish between pair-wise and non-pair-wise subqueries



DP 10-3 Multiple-Row Subqueries

#### Marin Marin

### Purpose

- A subquery is designed to find information you don't know so that you can find information you want to know
- However, single-row subqueries can return only one row. What if you need to find information based on several rows and several values?
- The subquery will need to return several rows
- We achieve this using multiple-row subqueries and the three comparison operators: IN, ANY, and ALL



DP 10-3 Multiple-Row Subqueries

# **Query Comparison**

- Whose salary is equal to the salary of an employee in department 20?
- This example returns an error because more than one employee exists in department 20, the subquery returns multiple rows
- We call this a multiple-row subquery

```
SELECT first_name, last_name
FROM employees
WHERE salary =
   (SELECT salary
   FROM employees
   WHERE department_id = 20);
```

LAST_NAME	DEPT_ID	SALARY
Hartstein	20	13000
Fay	20	6000



ORA-01427: single-row subquery returns more than one row



Academy

DP 10-3 Multiple-Row Subqueries

# **Query Comparison**

- The problem is the equal sign (=) in the WHERE clause of the outer query
- How can one value be equal to (or not equal to) more than one value at a time?
- It's a silly question, isn't it?

```
SELECT first name, last name
FROM employees
WHERE salary =
   (SELECT salary
    FROM employees
    WHERE department id = 20);
```





Academy

DP 10-3 Multiple-Row Subqueries

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

For the same reason, we cannot use <, > or <> in the WHERE clause condition.

It doesn't make sense to compare one value with several values. Is 10000 less than (13000,6000)?

#### Marin Dilla

# IN, ANY, and ALL

- Subqueries that return more than one value are called multiple-row subqueries
- Because we cannot use the single-row comparison operators (=, <, and so on), we need different comparison operators for multiple-row subqueries
- The multiple-row operators are:
  - -IN,
  - -ANY
  - -ALL
- The NOT operator can be used with any of these three operators

ORACLE Academy

DP 10-3 Multiple-Row Subqueries

#### IN

- The IN operator is used within the outer query WHERE clause to select only those rows which are IN the list of values returned from the inner query
- For example, we are interested in all the employees that were hired the same year as an employee in department 90

LAST_NAME	HIRE_DATE
King	17-Jun-1987
Kochhar	21-Sep-1989
De Haan	13-Jan-1993
Whalen	17-Sep-1987



Academy

DP 10-3 Multiple-Row Subqueries

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

THE EXTRACT function can be used to extract YEAR, MONTH or DAY fields from a DATE data type.

#### IN

- The inner query will return a list of the years that employees in department 90 were hired
- The outer query will then return any employee that was hired the same year as any year in the inner query list

```
SELECT last_name, hire_date

FROM employees

WHERE EXTRACT(YEAR FROM hire_date) IN

(SELECT EXTRACT(YEAR FROM hire_date)

FROM employees

WHERE department_id=90);
```

LAST_NAME	HIRE_DATE
King	17-Jun-1987
Kochhar	21-Sep-1989
De Haan	13-Jan-1993
Whalen	17-Sep-1987



DP 10-3 Multiple-Row Subqueries

#### ANY

- The ANY operator is used when we want the outerquery WHERE clause to select the rows which match the criteria (<, >, =, etc.) of at least one value in the subquery result set
- The example shown will return any employee whose year hired is less than at least one year hired of employees in department 90

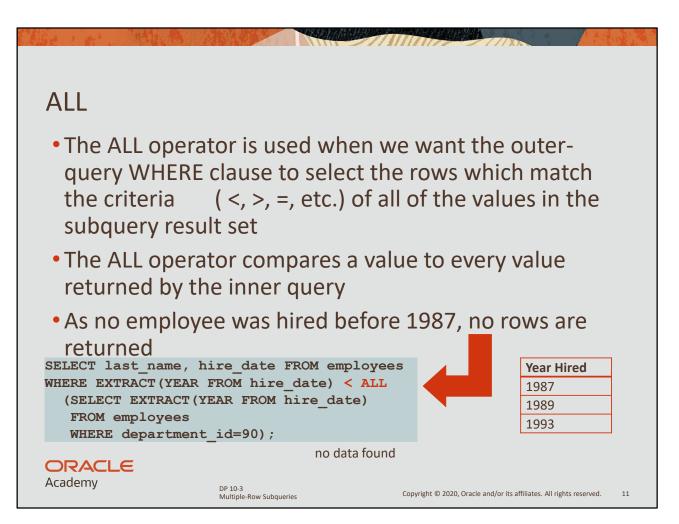
Year Hired
1987
1989
1993

SELECT last_name, hire_date			
FROM employees			
WHERE EXTRACT (YEAR FROM hire_date) < ANY			
(SELECT EXTRACT (YEAR FROM hire_date)			
FROM employees			
WHERE department id=90);			

LAST_NAME	HIRE_DATE
King	17-Jun-1987
Kochhar	21-Sep-1989
Whalen	17-Sep-1987
Hunold	03-Jan-1990
Ernst	21-May-1991

Academy

DP 10-3 Multiple-Row Subqueries



=ALL: How can one value equal every one of a set of values? For this reason, =ALL is rarely used.

#### **NULL Values**

- Suppose that one of the values returned by a multiple-row subquery is null, but other values are not
- If IN or ANY are used, the outer query will return rows which match the non-null values

N	IANAGER_ID
-	
10	00
10	00
10	01
10	01
20	05
10	00

SELECT last\_name,
employee\_id
FROM employees
WHERE employee\_id IN
 (SELECT manager\_id
 FROM employees);

LAST_NAME	EMPLOYEE_ID
King	100
Kochhar	101
De Haan	102
Higgins	205





Result of subquery

DP 10-3 Multiple-Row Subqueries

#### **NULL Values**

- If ALL is used, the outer query returns no rows because ALL compares the outer query row with every value returned by the subquery, including the null
- And comparing anything with null results in null

```
SELECT last_name, employee_id

FROM employees

WHERE employee_id <= ALL

(SELECT manager_id

FROM employees);
```

no data found



Academy

DP 10-3 Multiple-Row Subqueries

### **GROUP BY and HAVING**

- As you might suspect, the GROUP BY clause and the HAVING clause can also be used with multiple-row subqueries
- What if you wanted to find the departments whose minimum salary is less than the salary of any employee who works in department 10 or 20?

LAST_NAME	DEPT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000

DEPARTMENT_ID	MIN(SALARY)
10	4400
20	6000
50	2500
60	4200
80	8600
110	8300
(null)	7000



DP 10-3 Multiple-Row Subqueries

#### **GROUP BY and HAVING**

- We need a multiple-row subquery which returns the salaries of employees in departments 10 and 20
- The outer query will use a group function (MIN) so we need to GROUP the outer query BY department\_id

LAST_NAME	DEPT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000

DEPARTMENT_ID	MIN(SALARY)
10	4400
20	6000
50	2500
60	4200
80	8600
110	8300
(null)	7000



DP 10-3 Multiple-Row Subqueries

# **GROUP BY and HAVING**

#### Here is the SQL statement:

```
SELECT department_id, MIN(salary)
FROM employees
GROUP BY department_id
HAVING MIN(salary) < ANY
(SELECT salary
FROM employees
WHERE department_id IN (10,20))
ORDER BY department_id;
```

DEPARTMENT_ID	MIN(SALARY)
10	4400
20	6000
50	2500
60	4200
80	8600
110	8300
-	7000

LAST_NAME	DEPT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000

**Result of subquery** 



Academy

DP 10-3 Multiple-Row Subqueries

- Subqueries can use one or more columns
- If they use more than one column, they are called multiple-column subqueries
- A multiple-column subquery can be either pair-wise comparisons or non-pair-wise comparisons

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
176	149	80

ORACLE

Academy

DP 10-3 Multiple-Row Subqueries

- The example below shows a multiple-column pair-wise subquery with the subquery highlighted in red and the result in the table below
- The query lists the employees whose manager and departments are the same as the manager and department of employees 149 or 174

ORACLE Academy 
 EMPLOYEE\_ID
 MANAGER\_ID
 DEPARTMENT\_ID

 176
 149
 80

DP 10-3 Multiple-Row Subqueries

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

First, the subquery to retrieve the MANAGER\_ID and DEPARTMENT\_ID values for the employees with EMPLOYEE ID 149 or 174 is executed.

These values are compared with the MANAGER\_ID column and the DEPARTMENT\_ID column of each row in the EMPLOYEES table. If the values match, the row is displayed.

In the output, the records of the employees with the EMPLOYEE\_ID 149 or 174 will not be displayed.

A non-pair-wise
 multiple-column
 subquery also uses
 more than one column
 in the subquery, but it
 compares them one at
 a time, so the
 comparisons take place
 in different subqueries

```
SELECT
        employee id,
        manager id,
        department id
FROM
        employees
WHERE
       manager id IN
              manager id
    (SELECT
    FROM
             employees
             employee id IN
    WHERE
               (149, 174))
       department id IN
AND
    (SELECT
              department id
    FROM
             employees
    WHERE
             employee id IN
               (149, 174))
AND employee id NOT IN(149,174);
 EMPLOYEE ID
           MANAGER ID
                       DEPARTMENT ID
```



DP 10-3 Multiple-Row Subqueries 176

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

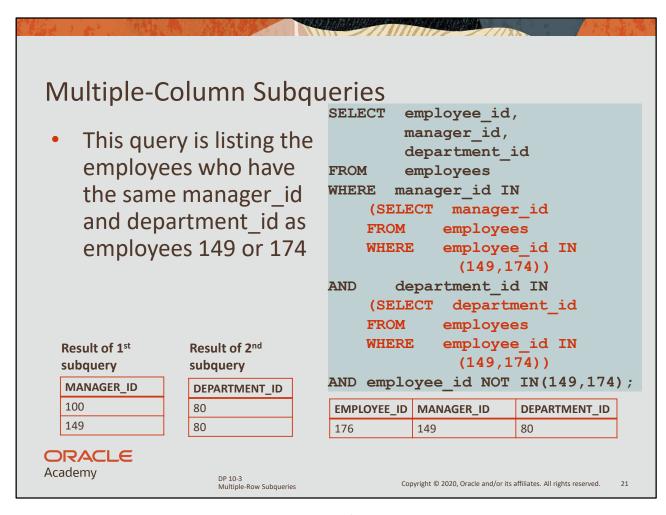
149

- You will need to write one subquery per column you want to compare against when performing non-pair-wise multiple column subqueries
- The example on the right shows a multiple-column non-pair-wise subquery with the subqueries highlighted in red

```
SELECT
         employee id,
         manager id,
         department id
FROM
         employees
WHERE
       manager id IN
    (SELECT
              manager id
             employees
    FROM
    WHERE
             employee id IN
               (149,174))
       department id IN
AND
    (SELECT
              department id
             employees
    FROM
    WHERE
             employee id IN
               (149, 174))
AND employee id NOT IN(149,174);
  EMPLOYEE_ID | MANAGER_ID
                        DEPARTMENT_ID
  176
            149
                        80
```

ORACLE Academy

DP 10-3 Multiple-Row Subqueries



First, the subquery to retrieve the MANAGER\_ID values for the employees with the EMPLOYEE\_ID 149 or 174 is executed. Similarly, the second subquery to retrieve the DEPARTMENT\_ID values for the employees with the EMPLOYEE\_ID 149 or 174 is executed.

The retrieved values are compared with the MANAGER\_ID and DEPARTMENT\_ID column for each row in the EMPLOYEES table.

# One Last Point About Subqueries

- Some subqueries may return a single row or multiple rows, depending on the data values in the rows
- If even the slightest possibility exists of returning multiple rows, make sure you write a multiple-row subquery

```
SELECT first_name, last_name,
job_id
FROM employees
WHERE job_id =
   (SELECT job_id
   FROM employees
   WHERE last name = 'Ernst');
```

FIRST_NAME	LAST_NAME	JOB_ID
Alexander	Hunold	IT_PROG
Bruce	Ernst	IT_PROG
Diana	Lorentz	IT_PROG



**Result of subquery** 

FIRST_NAME	LAST_NAME	JOB_ID
Bruce	Ernst	IT_PROG

DP 10-3 Multiple-Row Subqueries

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

22

# One Last Point About Subqueries

- For example: Who has the same job\_id as Ernst?
- This single-row subquery works correctly because there is only one Ernst in the table
- But what if later, the business hires a new employee named Susan Ernst?

```
SELECT first_name, last_name,
job_id
FROM employees
WHERE job_id =
   (SELECT job_id
   FROM employees
   WHERE last name = 'Ernst');
```

FIRST_NAME	LAST_NAME	JOB_ID
Alexander	Hunold	IT_PROG
Bruce	Ernst	IT_PROG
Diana	Lorentz	IT_PROG



**Result of subquery** 

FIRST_NAME	LAST_NAME	JOB_ID
Bruce	Ernst	IT_PROG

DP 10-3 Multiple-Row Subqueries

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

23

# One Last Point About Subqueries

- It would be better to write a multiple-row subquery
- The multiple-row subquery syntax will still work even if the subquery returns a single row
- If in doubt, write a multiple-row subquery!

```
SELECT first_name, last_name,
job_id
FROM employees
WHERE job_id IN
   (SELECT job_id
   FROM employees
   WHERE last_name = 'Ernst');
```

FIRST_NAME	LAST_NAME	JOB_ID
Alexander	Hunold	IT_PROG
Bruce	Ernst	IT_PROG
Diana	Lorentz	IT_PROG

FIRST_NAME	LAST_NAME		Result of subquery
Bruce	Ernst	IT_PROG	There are 2 people with last name 'Ernst'

ORACLE

Academy

DP 10-3
Multiple-Row Subqueries

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

When we use IN, ANY, or ALL to compare with a list of values, they will still work even if there is only one value in the list.

# **Terminology**

- Key terms used in this lesson included:
  - -Non-pair-wise multiple column subquery
  - -Pair-wise multiple column subquery



DP 10-3 Multiple-Row Subqueries

# Summary

- In this lesson, you should have learned how to:
  - Correctly use the comparison operators IN, ANY, and ALL in multiple-row subqueries
  - Construct and execute a multiple-row subquery in the WHERE clause or HAVING clause
  - Describe what happens if a multiple-row subquery returns a null value
  - Understand when multiple-row subqueries should be used, and when it is safe to use a single-row subquery
  - Distinguish between pair-wise and non-pair-wise subqueries



DP 10-3 Multiple-Row Subqueries

# ORACLE Academy