8

Using Subqueries to Solve Queries

Objectives

After completing this lesson, you should be able to do the following:

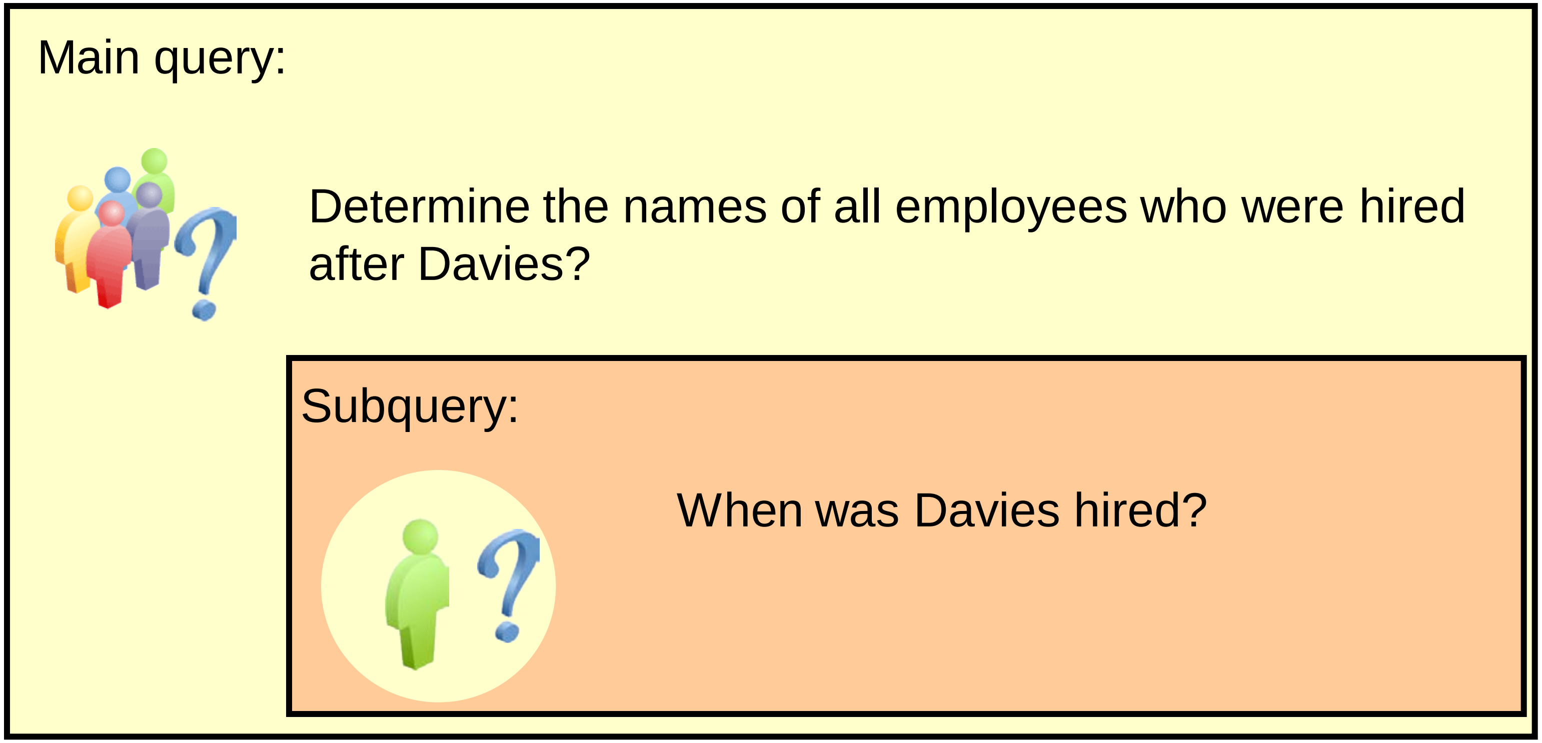
- Define subqueries
- Describe the types of problems that the subqueries can solve
- List the types of subqueries
- Write single-row, multiple-row, multiple-column subqueries

Lesson Agenda

- Subquery: Types, syntax, and guidelines
- Single-row subqueries:
  - Group functions in a subquery
  - `HAVING` clause with subqueries
- Multiple-row subqueries
  - Using `ALL` or `ANY` operator
- Multiple-column subqueries
- Null values in a subquery

Using a Subquery to Solve a Problem

Who is hired after Davies?



Main query:

Determine the names of all employees who were hired after Davies?

Subquery:

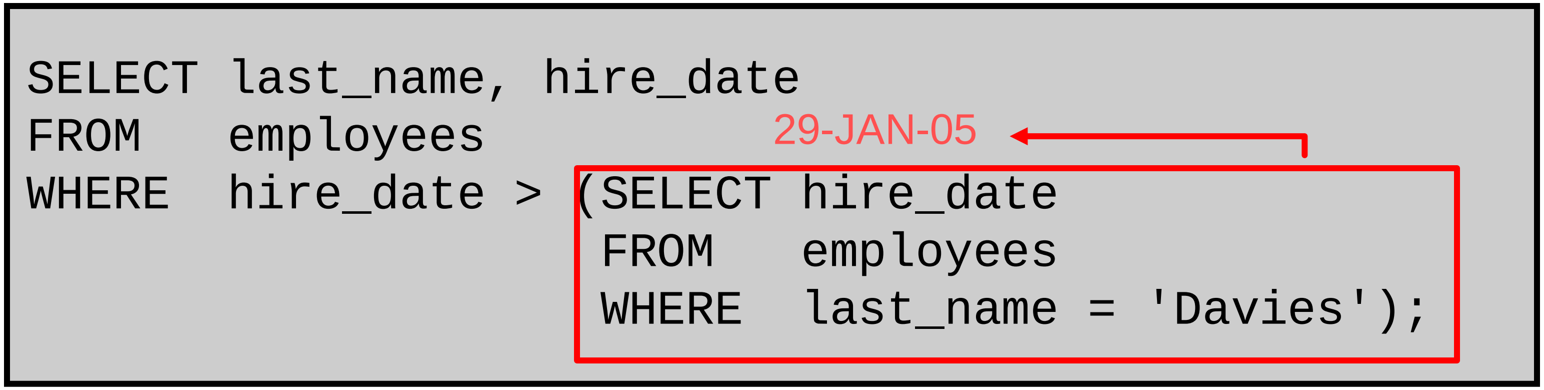
When was Davies hired?

Subquery Syntax

- The subquery (inner query) executes *before* the main query (outer query).
- The result of the subquery is used by the main query.

```
SELECT   select_list
FROM     table
WHERE    expr operator
                    (SELECT      select_list
                    FROM         table);
```

Using a Subquery



```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date > (SELECT hire_date
                    FROM   employees
                    WHERE  last_name = 'Davies');
```

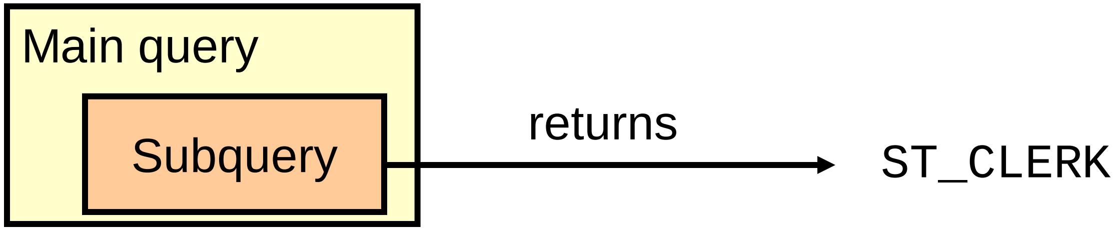
29-JAN-05

Rules and Guidelines for Using Subqueries

- Enclose subqueries in parentheses.
- Place subqueries on the right side of the comparison condition for readability. (However, the subquery can appear on either side of the comparison operator.)
- Use single-row operators with single-row subqueries and multiple-row operators with multiple-row subqueries.

Types of Subqueries

- Single-row subquery



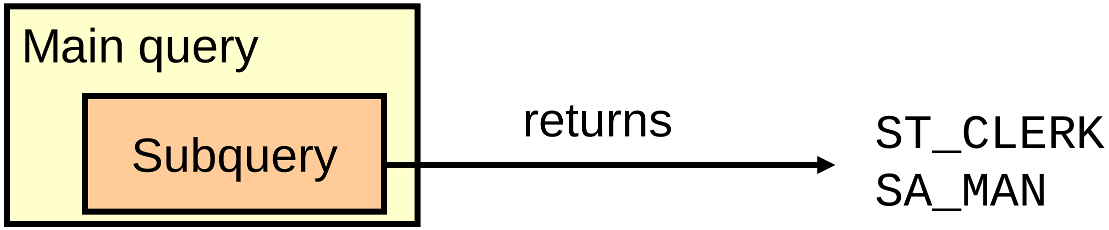
Main query

Subquery

returns

ST_CLERK

- Multiple-row subquery



Main query

Subquery

returns

ST_CLERK
SA_MAN

Lesson Agenda

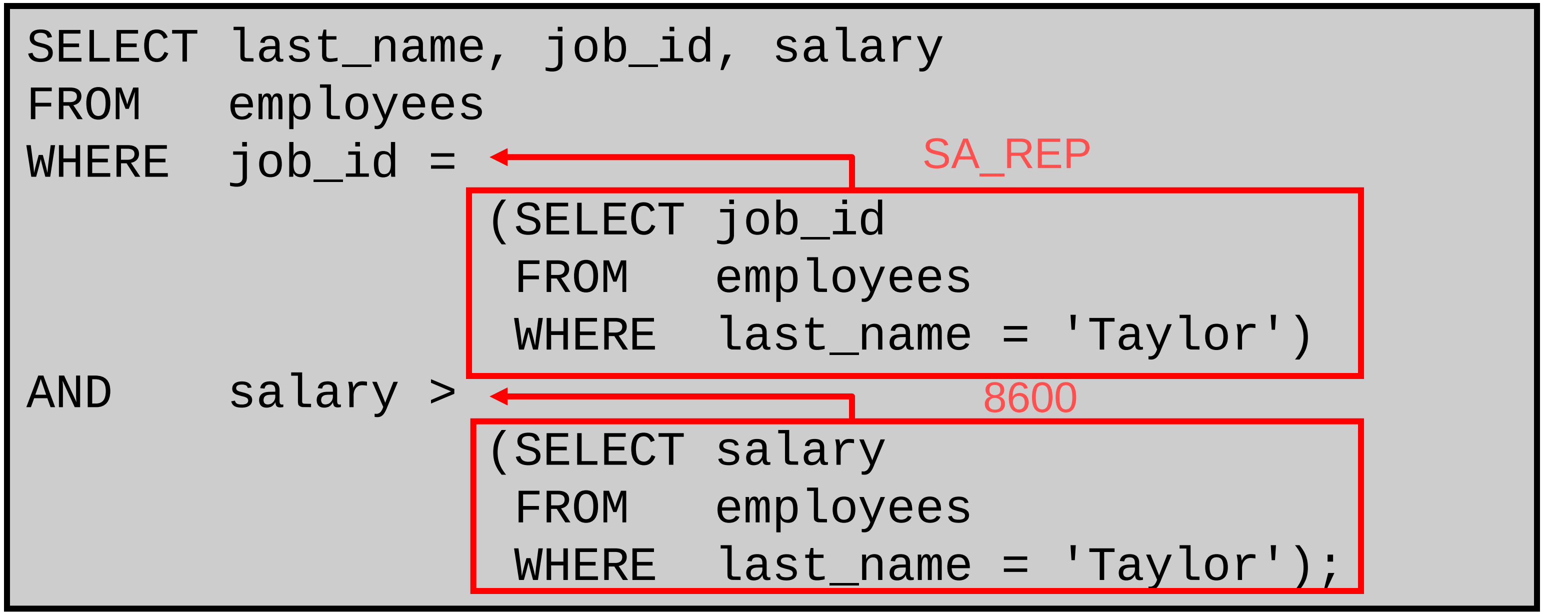
- Subquery: Types, syntax, and guidelines
- Single-row subqueries:
  - Group functions in a subquery
  - `HAVING` clause with subqueries
- Multiple-row subqueries
  - Using `ALL` or `ANY` operator
- Multiple-column subqueries
- Null values in a subquery

Single-Row Subqueries

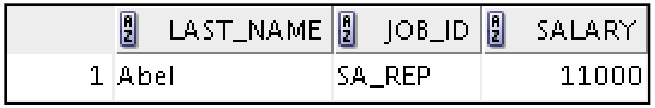
- Return only one row
- Use single-row comparison operators

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to

Executing Single-Row Subqueries



```
SELECT last_name, job_id, salary
FROM   employees
WHERE  job_id =
                (SELECT job_id
                 FROM   employees
                 WHERE  last_name = 'Taylor')
AND    salary >
                (SELECT salary
                 FROM   employees
                 WHERE  last_name = 'Taylor');
```

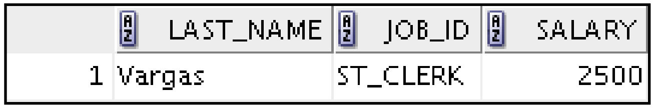


	LAST_NAME	JOB_ID	SALARY
1	Abel	SA_REP	11000

Using Group Functions in a Subquery

```
SELECT last_name, job_id, salary
FROM   employees
WHERE  salary =
                (SELECT MIN(salary)
                 FROM   employees);
```

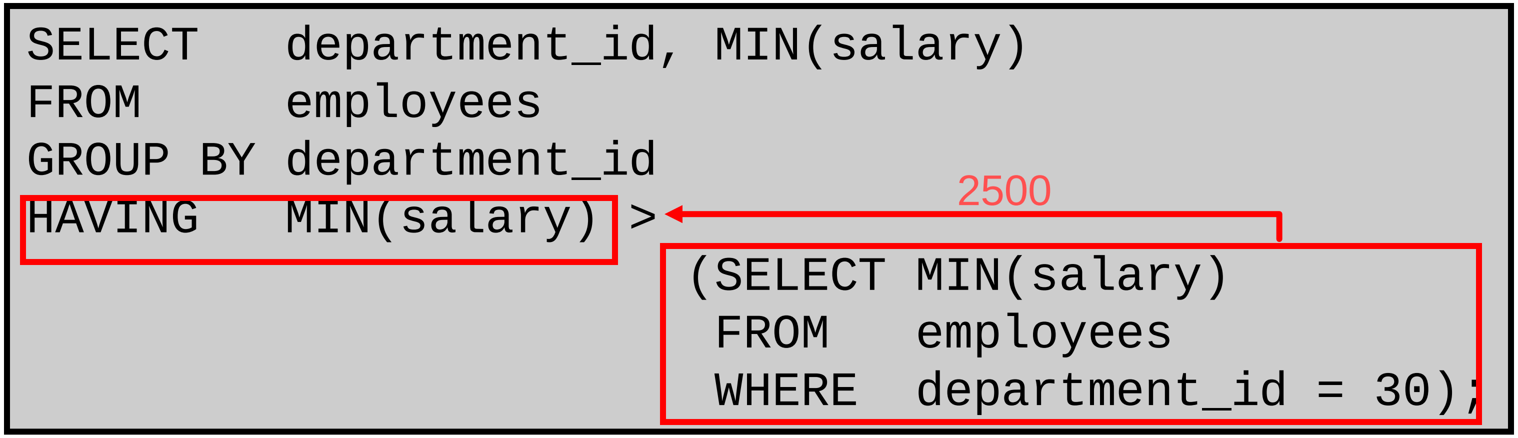
2500



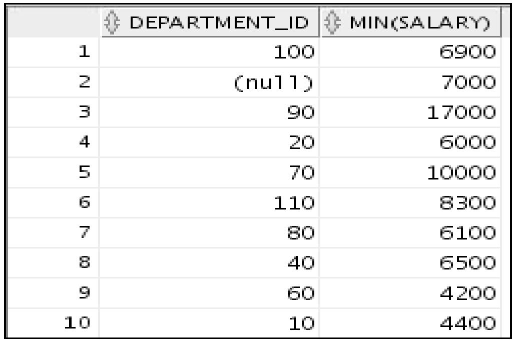
	LAST_NAME	JOB_ID	SALARY
1	Vargas	ST_CLERK	2500

HAVING Clause with Subqueries

- The Oracle server executes the subqueries first.
- The Oracle server returns results into the HAVING clause of the main query.



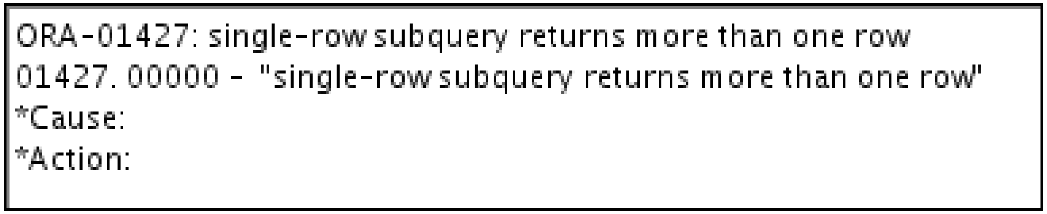
```
SELECT   department_id, MIN(salary)
FROM     employees
GROUP BY department_id
HAVING   MIN(salary) >
                       (SELECT MIN(salary)
                        FROM   employees
                        WHERE  department_id = 30);
```



	DEPARTMENT_ID	MIN(SALARY)
1	100	6900
2	(null)	7000
3	90	17000
4	20	6000
5	70	10000
6	110	8300
7	80	6100
8	40	6500
9	60	4200
10	10	4400

What Is Wrong with This Statement?

```
SELECT employee_id, last_name
FROM   employees
WHERE  salary =
                (SELECT   MIN(salary)
                 FROM     employees
                 GROUP BY department_id);
```

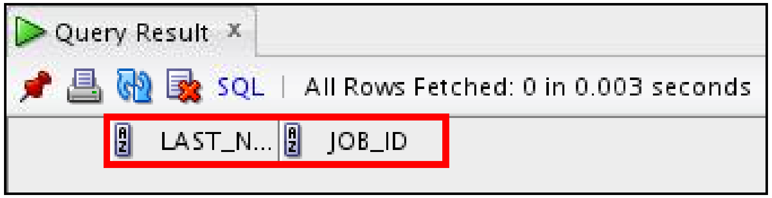


```
ORA-01427: single-row subquery returns more than one row
01427. 00000 -  "single-row subquery returns more than one row"
*Cause:
*Action:
```

Single-row operator with multiple-row subquery

No Rows Returned by the Inner Query

```
SELECT last_name, job_id
FROM   employees
WHERE  job_id =
                (SELECT job_id
                 FROM   employees
                 WHERE  last_name = 'Haas');
```



All Rows Fetched: 0 in 0.003 seconds

LAST_N...	JOB_ID

Subquery returns no rows because there is no employee named “Haas.”

Lesson Agenda

- Subquery: Types, syntax, and guidelines
- Single-row subqueries:
  - Group functions in a subquery
  - `HAVING` clause with subqueries
- Multiple-row subqueries
  - Use `IN`, `ALL`, or `ANY`
- Multiple-column subqueries
- Null values in a subquery

Multiple-Row Subqueries

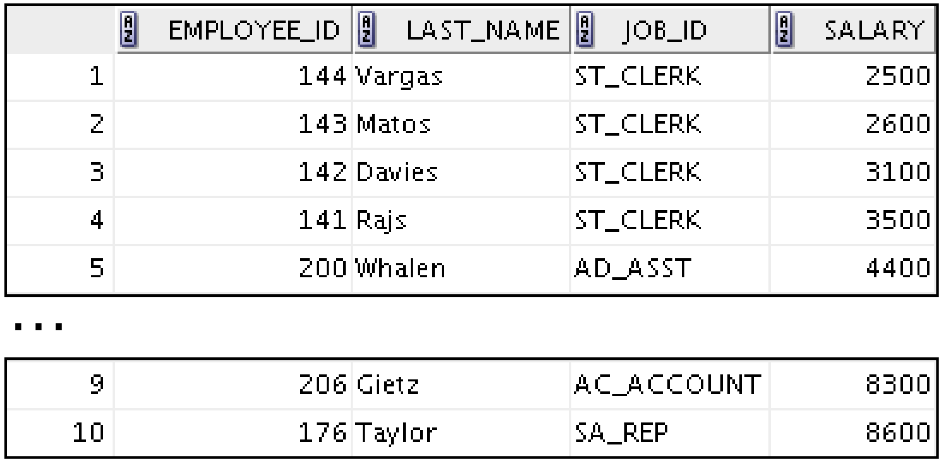
- Return more than one row
- Use multiple-row comparison operators

Operator	Meaning
`IN`	Equal to any member in the list
`ANY`	Must be preceded by =, !=, >, <, <=, >=. Returns TRUE if at least one element exists in the result set of the subquery for which the relation is TRUE.
`ALL`	Must be preceded by =, !=, >, <, <=, >=. Returns TRUE if the relation is TRUE for all elements in the result set of the subquery.

Using the ANY Operator in Multiple-Row Subqueries

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary < ANY
                    (SELECT salary
                     FROM   employees
                     WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```

9000, 6000, 4200

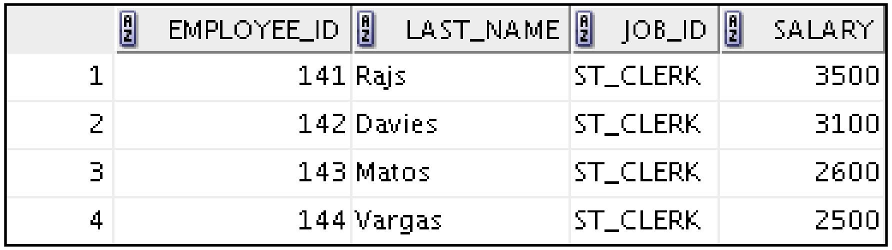


	EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
1	144	Vargas	ST_CLERK	2500
2	143	Matos	ST_CLERK	2600
3	142	Davies	ST_CLERK	3100
4	141	Rajs	ST_CLERK	3500
5	200	Whalen	AD_ASST	4400
...				
9	206	Gietz	AC_ACCOUNT	8300
10	176	Taylor	SA_REP	8600

Using the ALL Operator in Multiple-Row Subqueries

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary < ALL
                   (SELECT salary
                    FROM   employees
                    WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```

9000, 6000, 4200



	EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
1	141	Rajs	ST_CLERK	3500
2	142	Davies	ST_CLERK	3100
3	143	Matos	ST_CLERK	2600
4	144	Vargas	ST_CLERK	2500

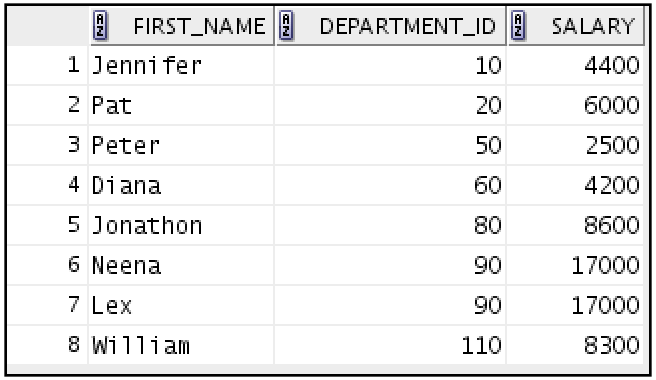
Multiple-Column Subqueries

- A multiple-column subquery returns more than one column to the outer query.
- Column comparisons in multiple column comparisons can be pairwise or nonpairwise.
- A multiple-column subquery can also be used in the `FROM` clause of a `SELECT` statement.

Multiple-Column Subquery: Example

Display all the employees with the lowest salary in each department

```
SELECT first_name, department_id, salary
FROM employees
WHERE (salary, department_id) IN
      (SELECT min(salary), department_id
       FROM employees
       GROUP BY department_id)
ORDER BY department_id;
```



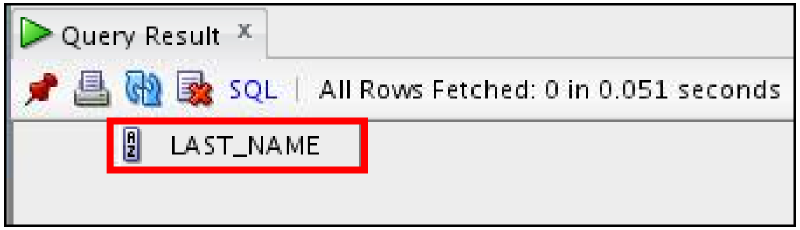
	FIRST_NAME	DEPARTMENT_ID	SALARY
1	Jennifer	10	4400
2	Pat	20	6000
3	Peter	50	2500
4	Diana	60	4200
5	Jonathon	80	8600
6	Neena	90	17000
7	Lex	90	17000
8	William	110	8300

Lesson Agenda

- Subquery: Types, syntax, and guidelines
- Single-row subqueries:
  - Group functions in a subquery
  - `HAVING` clause with subqueries
- Multiple-row subqueries
  - Using `ALL` or ANY operator
- Multiple-column subqueries
- Null values in a subquery

Null Values in a Subquery

```
SELECT emp.last_name
FROM   employees emp
WHERE  emp.employee_id NOT IN
                           (SELECT mgr.manager_id
                            FROM   employees mgr);
```



All Rows Fetched: 0 in 0.051 seconds

LAST_NAME

Subquery returns no rows because one of the values returned by a subquery is null.

Quiz

Using a subquery is equivalent to performing two sequential queries and using the result of the first query as the search values in the second query.

a. True

b. False

Summary

In this lesson, you should have learned how to:

- Define subqueries
- Identify the types of problems that the subqueries can solve
- Write single-row, multiple-row, multiple-column subqueries

Practice 8: Overview

This practice covers the following topics:

- Creating subqueries to query values based on unknown criteria
- Using subqueries to find out the values that exist in one set of data and not in another