6

Usage of 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 subqueries can solve
- List the types of subqueries
- Write single-row and multiple-row subqueries

Course RoadMap

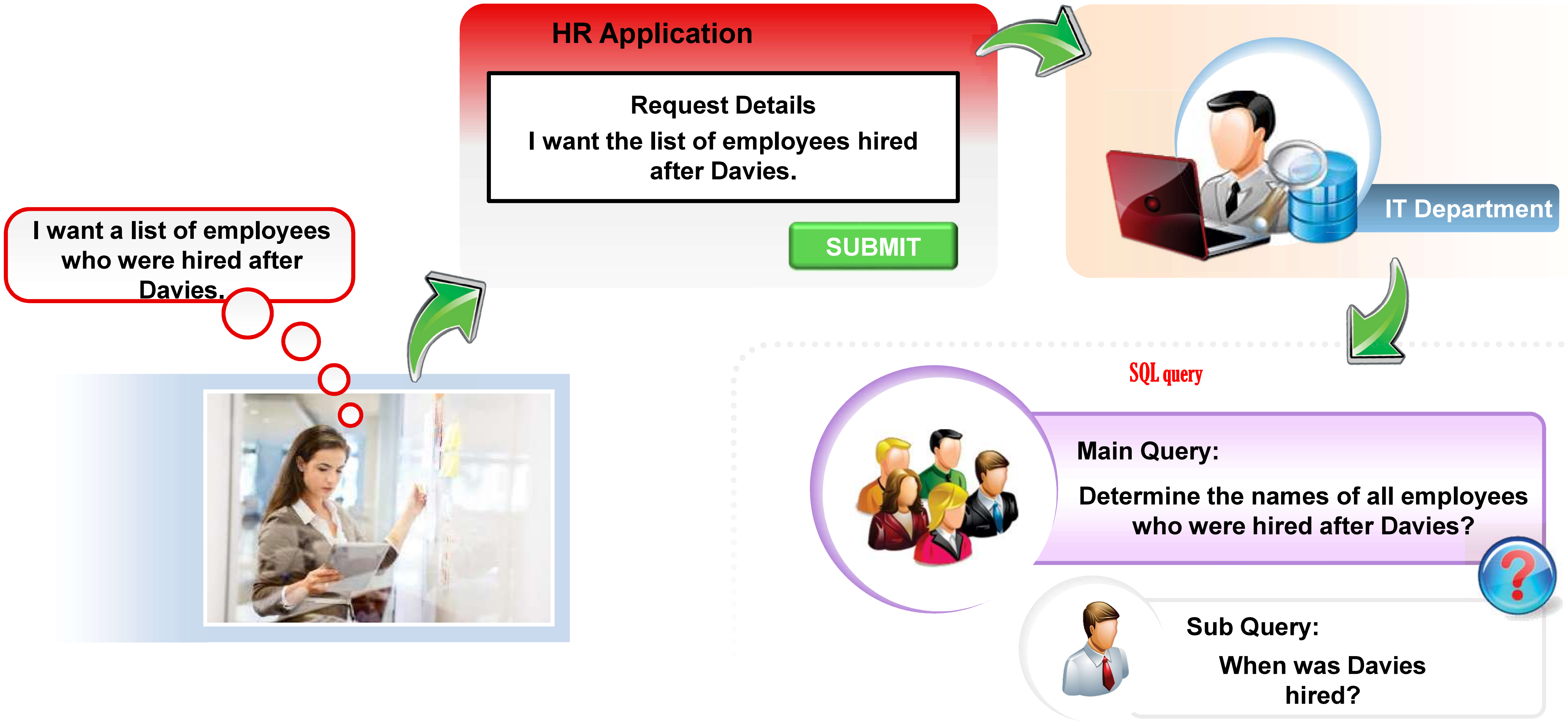
- Lesson 1: Introduction
- Unit 1: Retrieving, Restricting, and Sorting Data
- **Unit 2: Joins, Subqueries, and Set Operators**
- Unit 3: DML and DDL

- Lesson 6: Reporting Aggregated Data Using Group Functions
- Lesson 7: Displaying Data from Multiple Tables Using Joins
- **Lesson 8: Using Subqueries to Solve Queries** ◄ **You are here!**
- Lesson 9: Using Set Operators

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

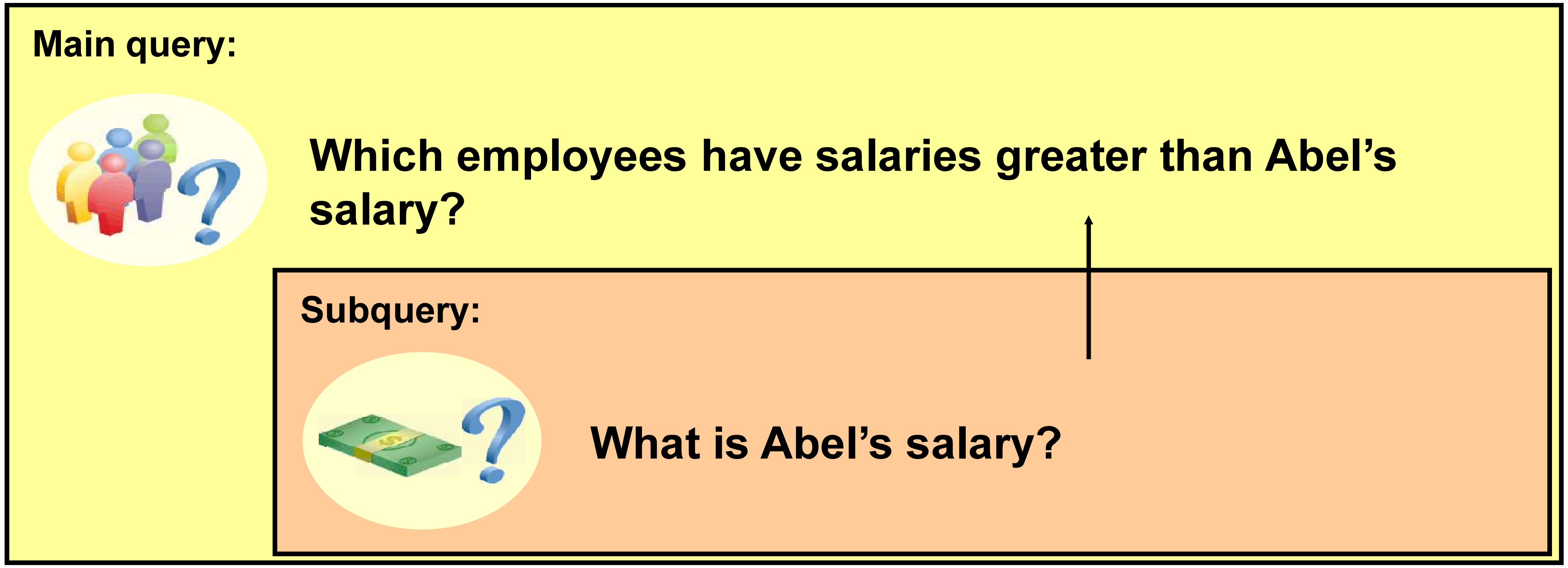
Using a Subquery to Solve a Problem



I want a list of employees who were hired after Davies.
HR Application
Request Details
I want the list of employees hired after Davies.
SUBMIT
IT Department
SQL query
Main Query:
Determine the names of all employees who were hired after Davies?
Sub Query:
When was Davies hired?

Using a Subquery

Who has a salary greater than Abel's?



Main query:

Which employees have salaries greater than Abel's salary?

Subquery:

What is Abel's salary?

Subquery Syntax

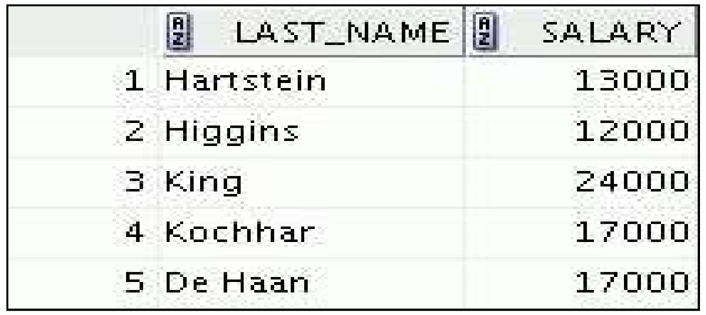
```
SELECT    select_list
FROM      table
WHERE     expr operator
                     (SELECT        select_list
                      FROM          table);
```

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

Using a Subquery

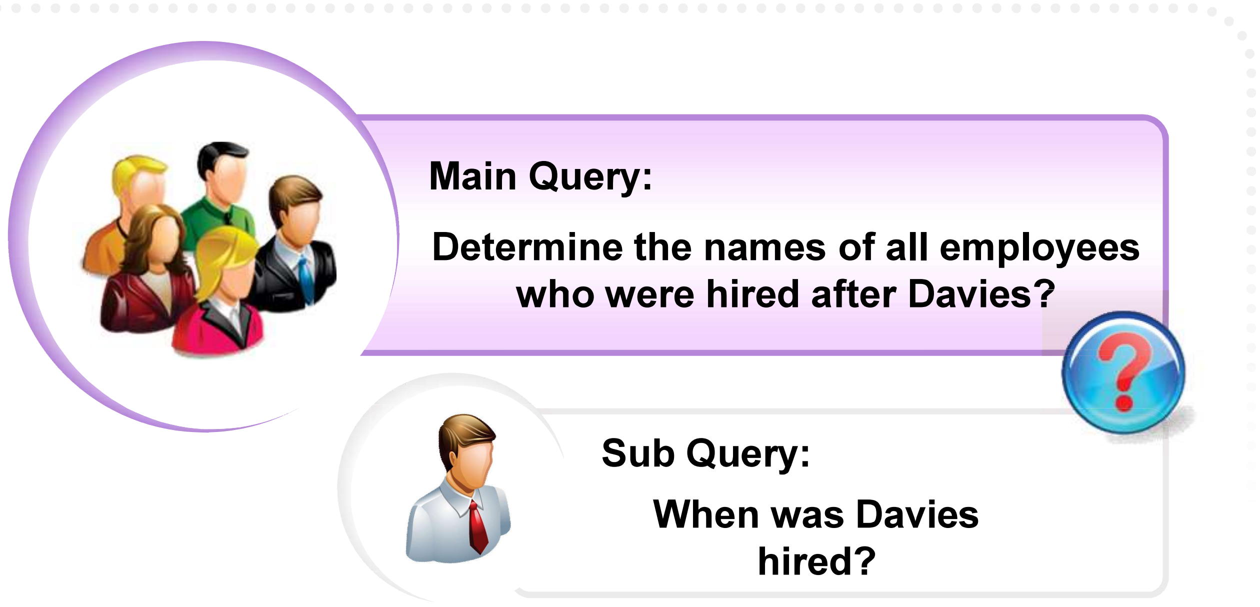
```
SELECT last_name, salary
FROM   employees
WHERE  salary >
               (SELECT salary
                FROM   employees
                WHERE  last_name = 'Abel');
```

11000



```
     LAST_NAME   SALARY
  1  Hartstein    13000
  2  Higgins      12000
  3  King         24000
  4  Kochhar      17000
  5  De Haan      17000
```

Using a Subquery


Main Query:
Determine the names of all employees who were hired after Davies?
Sub Query:
When was Davies hired?

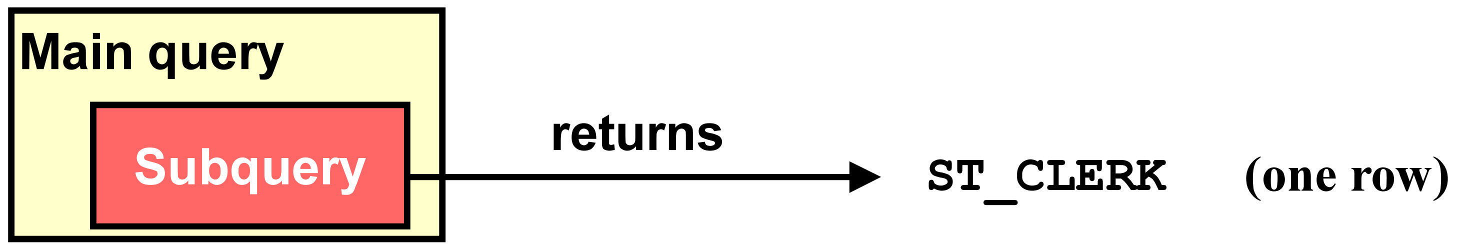
```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date > (SELECT hire_date
                    FROM   employees
                    WHERE  last_name = 'Davies');
```

Rules and Guidelines for Using Subqueries

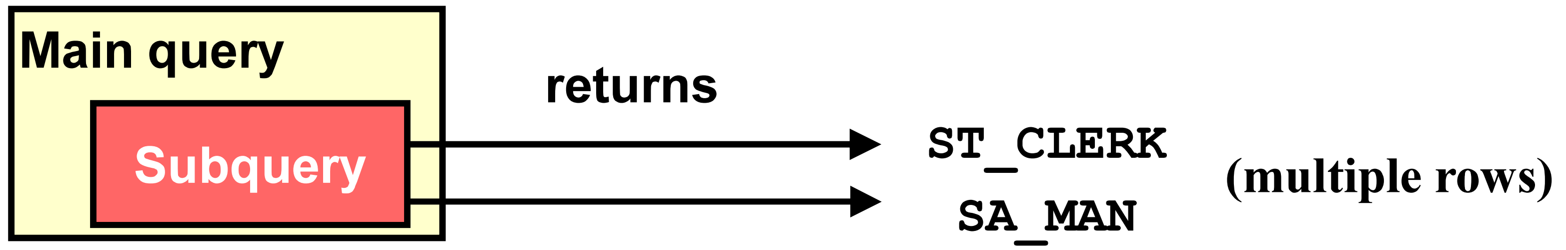
- Enclose subqueries in parentheses.
- Place subqueries on the right side of the comparison condition for readability. (However, the subquery can appear on either side of the comparison operator.)
- Use single-row operators with single-row subqueries and multiple-row operators with multiple-row subqueries.

Types of Subqueries

- Single-row subquery



- Multiple-row subquery



Lesson Agenda

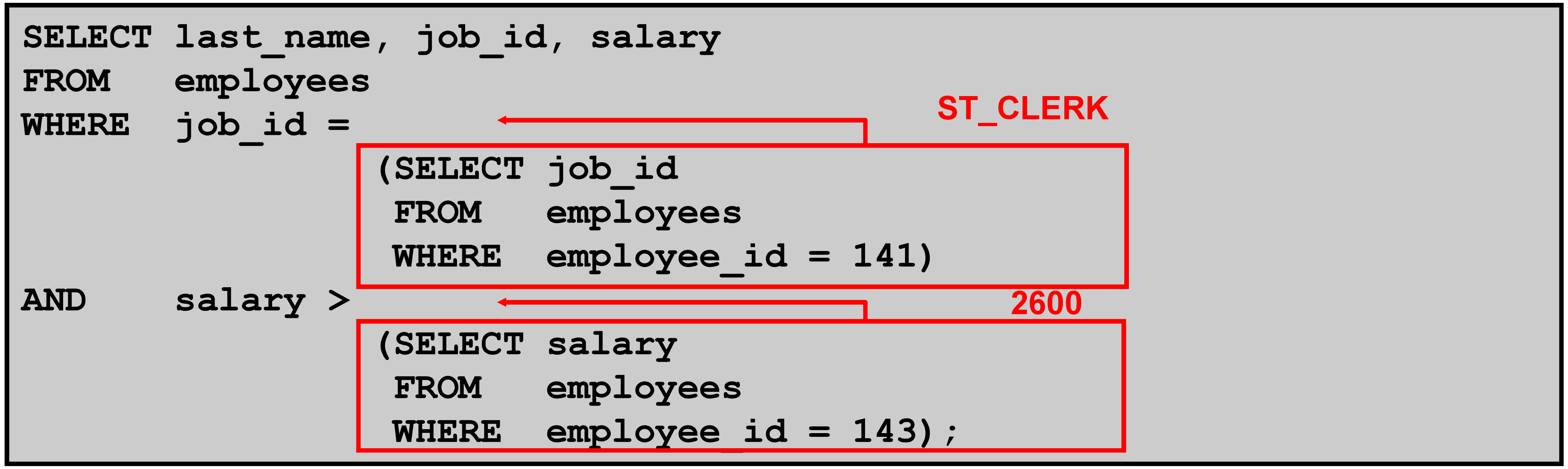
- Subquery: Types, syntax, and guidelines
- Single-row subqueries:
  - Group functions in a subquery
  - `HAVING` clause with subqueries
- Multiple-row subqueries
  - Using `ALL` or `ANY` operator
- Multiple-column subqueries
- Null values in a subquery

Single-Row Subqueries

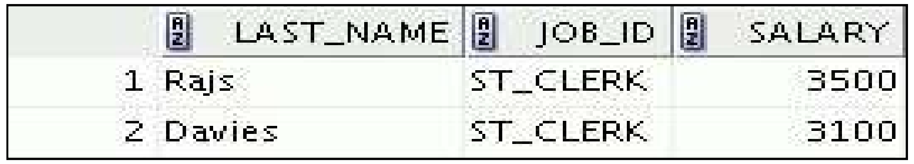
- Return only one row
- Use single-row comparison operators

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  employee_id = 141)
AND    salary >
                (SELECT salary
                 FROM   employees
                 WHERE  employee_id = 143);
```

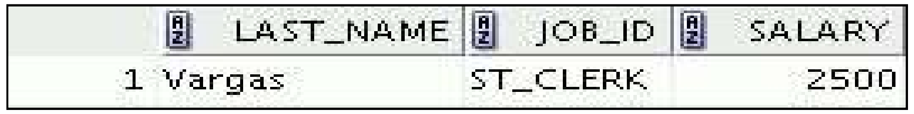


	LAST_NAME	JOB_ID	SALARY
1	Rajs	ST_CLERK	3500
2	Davies	ST_CLERK	3100

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

The HAVING Clause with Subqueries

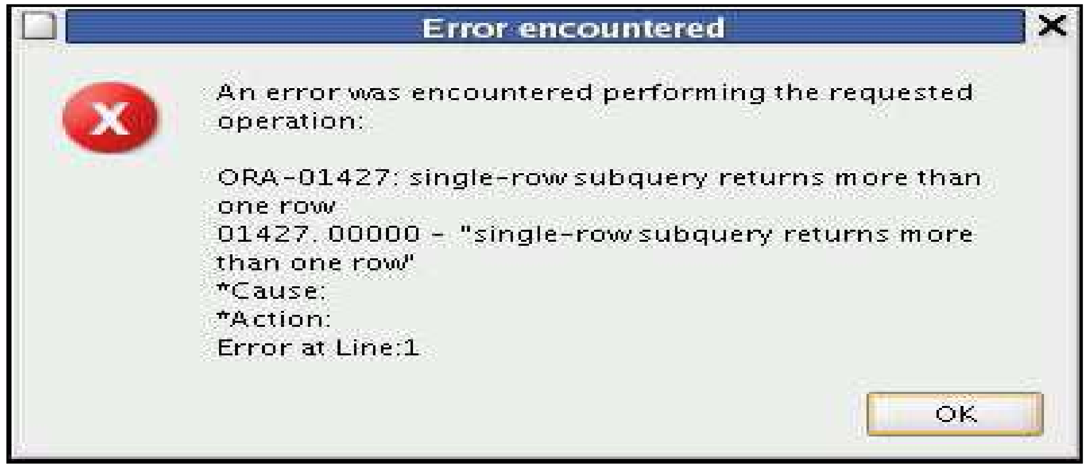
- The Oracle server executes subqueries first.
- The Oracle server returns results into the HAVING clause of the main query.

```
SELECT   department_id, MIN(salary)
FROM     employees
GROUP BY department_id
HAVING   MIN(salary) >
                       (SELECT MIN(salary)
                        FROM   employees
                        WHERE  department_id = 50);
```

2500

What Is Wrong with This Statement?

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



Error encountered

An error was encountered performing the requested operation:

```
ORA-01427: single-row subquery returns more than
one row
01427. 00000 - "single-row subquery returns more
than one row"
*Cause:
*Action:
Error at Line:1
```

Single-row operator with multiple-row subquery

Will This Statement Return Rows?

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

0 rows selected
```

Subquery returns no values.

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

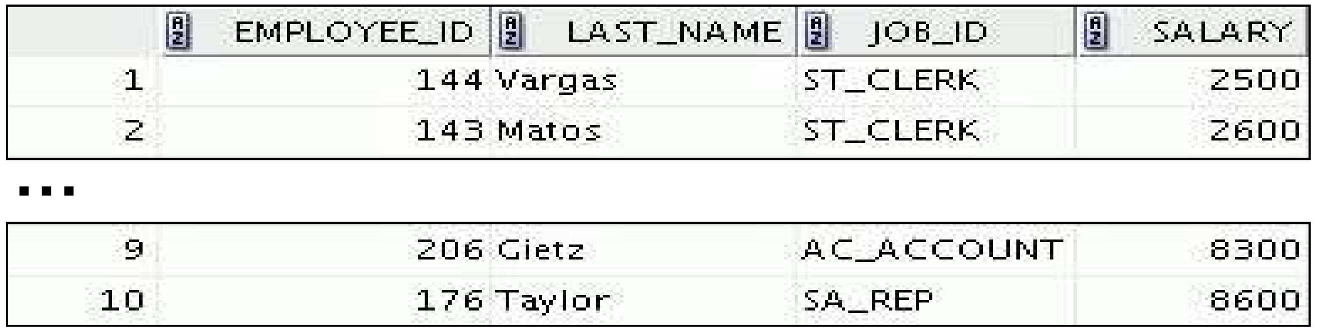
Multiple-Row Subqueries

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

Operator	Meaning
`IN`	Equal to any member in the list
`ANY`	Compare value to each value returned by the subquery
`ALL`	Compare value to every value returned by the subquery

Using the ANY Operator

```
SELECT employee_id, last_name, job_id, salary
FROM   employees                          9000, 6000, 4200
WHERE  salary < ANY
                (SELECT salary
                 FROM   employees
                 WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```



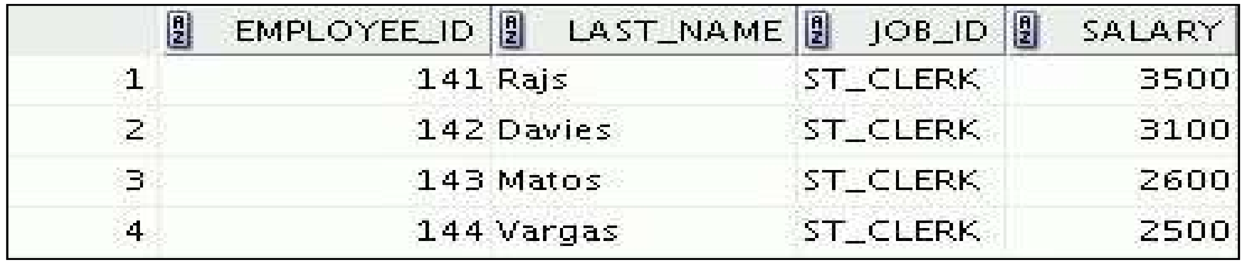
	EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
1	144	Vargas	ST_CLERK	2500
2	143	Matos	ST_CLERK	2600

...

9	206	Gietz	AC_ACCOUNT	8300
10	176	Taylor	SA_REP	8600

Using the `ALL` Operator

```
SELECT employee_id, last_name, job_id, salary
FROM   employees                              9000, 6000, 4200
WHERE  salary < ALL
                    (SELECT salary
                     FROM   employees
                     WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```



	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

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

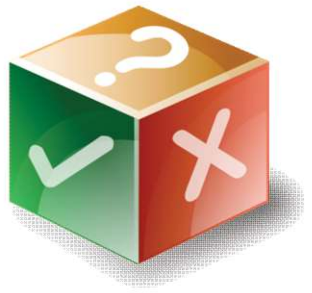
```
SELECT emp.last_name
FROM   employees emp
WHERE  emp.employee_id NOT IN
                             (SELECT mgr.manager_id
                              FROM   employees mgr);

0 rows selected
```

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:

- Identify when a subquery can help solve a question
- Write subqueries when a query is based on unknown values

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

Practice 6: Overview

This practice covers the following topics:

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