6- Subqueries

Objectives

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

- **Describe the types of problem that subqueries can solve**
- **Define subqueries**
- **List the types of subqueries**
- **Write single-row and multiple-row subqueries**

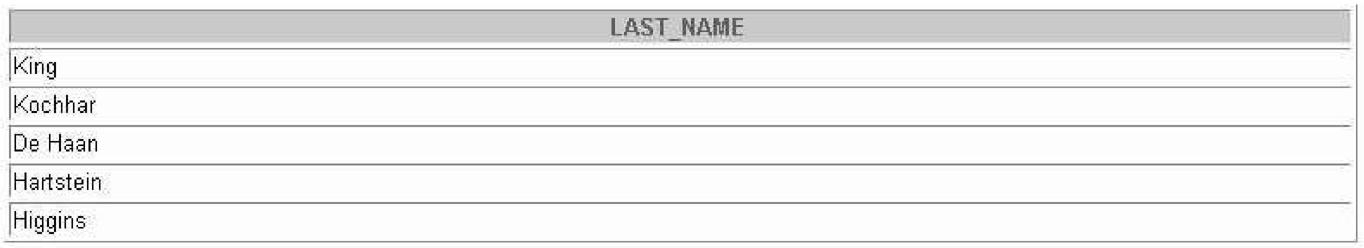
Using a Subquery to Solve a Problem

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.
•The result of the subquery is used by the main



LAST_NAME
King
Kochhar
De Haan
Hartstein
Higgins

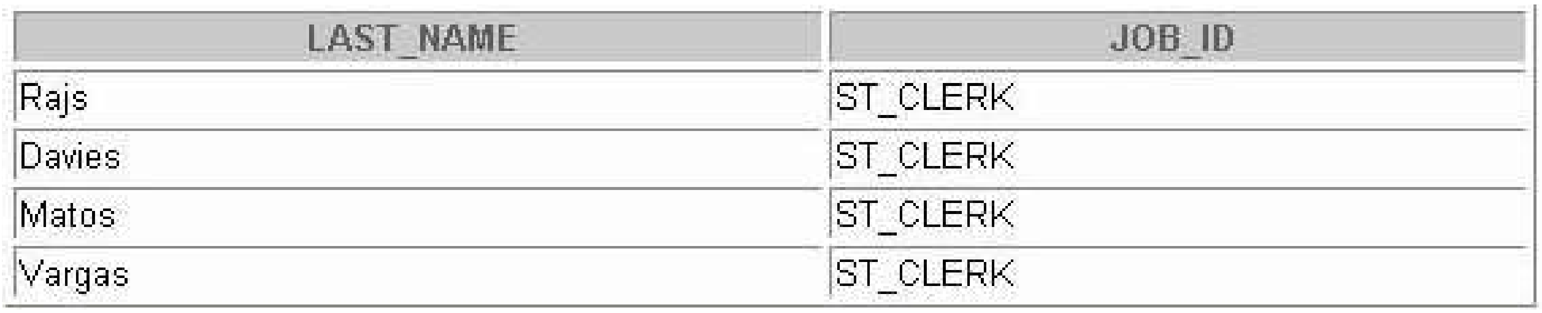
output is used to complete the query condition for the main or outer query.

Using a Subquery

```
SELECT last_name
FROM   employees
11000
WHERE  salary >
(SELECT salary
FROM employees
WHERE  last_name = 'Abel');
```

Types of Subqueries

• Single-row subquery Main query returns
ST_CLERK Subquery
• Multiple-row subquery Main query
ST_CLERK returns Subquery



LAST_NAME	JOB_ID
Rajs	ST_CLERK
Davies	ST_CLERK
Matos	ST_CLERK
Vargas	ST_CLERK

SA_MAN

Single-Row Subqueries

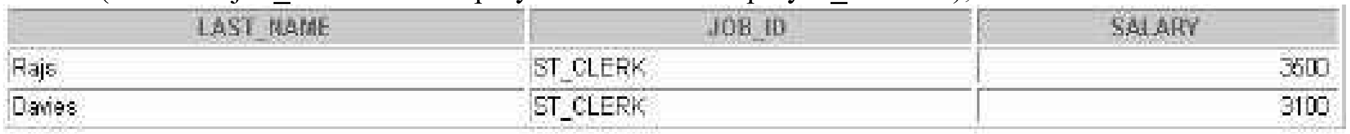
•Return only one row
•Use single-row comparison operators

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

Example

Display the employees whose job ID is the same as that of employee 141.

```
SELECT last_name, job_id FROM   employees WHERE  job_id =
(SELECT job_id FROM   employees WHERE  employee_id = 141);
```

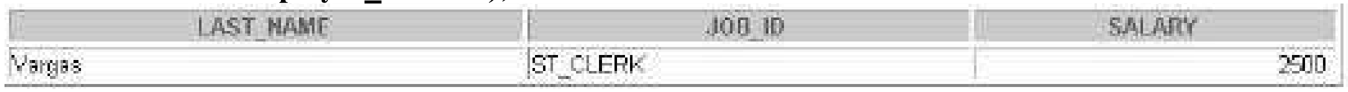


LAST_NAME	JOB_ID	SALARY
Rajs	ST_CLERK	3500
Davies	ST_CLERK	3100

Executing Single-Row Subqueries

```
SELECT last_name, job_id, salary
FROM   employees
WHERE  job_id =
ST_CLERK
(SELECT job_id
FROM   employees
WHERE  employee_id = 141)
AND    salary >
2600
```

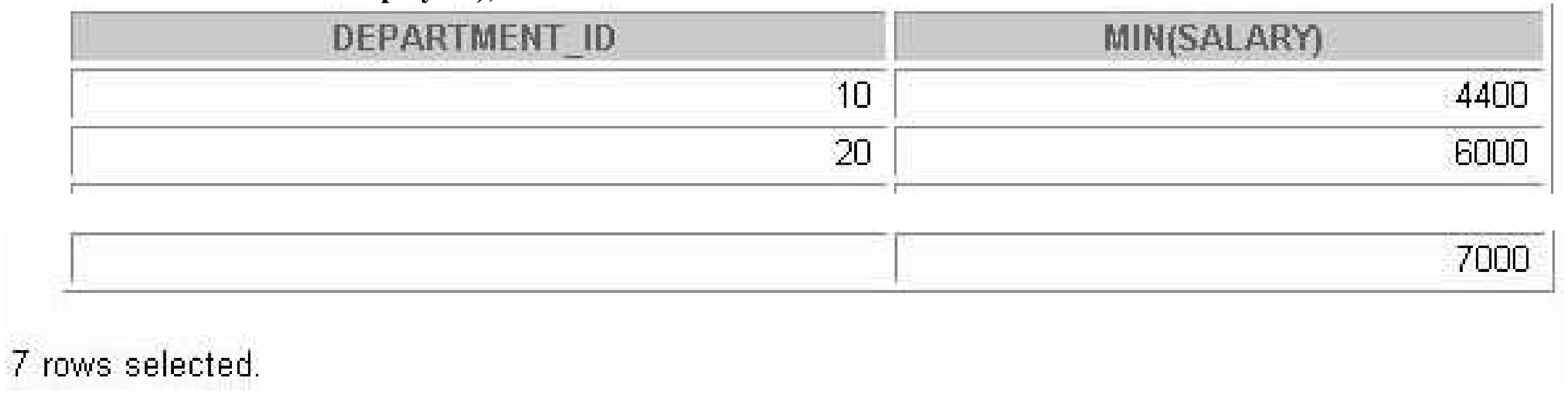
```
(SELECT salary
FROM   employees
WHERE  employee_id = 143);
```



LAST_NAME	JOB_ID	SALARY
Vargas	ST_CLERK	2500

Using Group Functions in a Subquery

```
SELECT last_name, job_id, salary
FROM   employees
2500
WHERE  salary =
(SELECT MIN(salary)
FROM   employees);
```



DEPARTMENT_ID	MIN(SALARY)
10	4400
20	6000
	7000

7 rows selected.

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
2500
HAVING   MIN(salary) >
(SELECT MIN(salary)
FROM employees
WHERE department_id = 50);
```

Example

Find the job with the lowest average salary.

```
SELECT   job_id, AVG(salary)
FROM     employees
GROUP BY job_id
HAVING   AVG(salary) = (SELECT   MIN(AVG(salary))
FROM     employees
GROUP BY job_id);
```

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 at line 4:
ORA-01427: single-row subquery returns more than
one row
```

Will This Statement Return Rows?

```
SELECT last_name, job_id
FROM   employees
WHERE  job_id =
(SELECT job_id
FROM   employees
WHERE  last_name = 'Haas');
no rows selected
```

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**

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

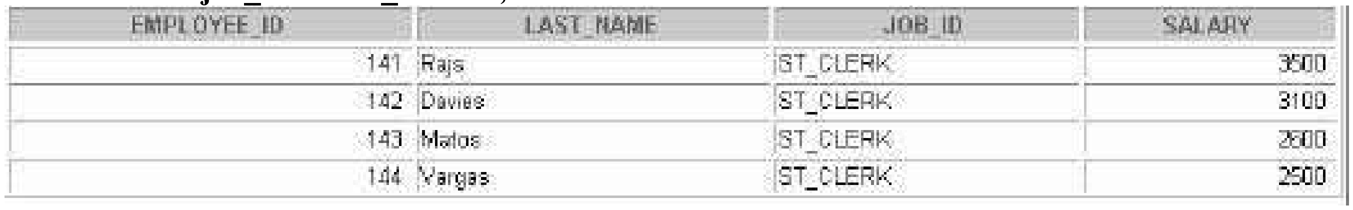
Example

Find the employees who earn the same salary as the minimum salar y for each department. The inner query is executed first, producing a query result. The main query block is then processed and uses the values returned by the inner query to complete its sear ch condition. In fact, the main query would
look like the following to the Oracle Server:

```
SELECT last_name, salary, department_id
FROM   employees
WHERE  salary IN (2500, 4200, 4400, 6000, 7000, 8300, 8600, 1 7000);
```

Using the ANY Operator in Multiple-Row Subqueries

```
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
141	Rajs	ST_CLERK	3500
142	Davies	ST_CLERK	3100
143	Matos	ST_CLERK	2600
144	Vargas	ST_CLERK	2500

Using the ALL Operator in Multiple-Row Subqueries

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary < ALL
9 00 0 ,  6 0 00 ,   42 0 0
(SELECT salary
FROM   employees
WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```

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);
no rows selected
```

Practice 5

1. Write a query to display the last name and hire date of any employee in the same department as Zlotkey. Exclude Zlotkey.
2. Create a query to display the employee numbers and last names of all employees who earn more than the average salary. Sort the results in ascending order of salary.
3. Write a query that displays the employee numbers and last names of all employees who work in a department with any employee whose last name contains a *u*. Place your SQL statement in a text file named
lab6_3.sql. Run your query.
4. Display the last name, department number, and job ID of all employees whose department location ID is 1700.
5. Display the last name and salary of every employee who reports to King.
6. Display the department number, last name, and job ID for every employee in the Executive department.

If you have time, complete the following exercises:

7. Modify the query in lab6_3.sql to display the employee numbers, last names, and salaries of all employees who earn more than the average salary and who work in a department with any employee with a *u* in their name. Resave lab6_3.sql to lab6_7.sql . Run the statement in lab6_7.sql.