# Sub Queries

## Learning Objective

After completing this lab the student should be able to:

- Define sub queries
- Describe the types of problems that sub quires can solve
- Syntax and rule for writing sub query
- Write single row and multiple-row sub queries.

## Tools and Technologies

- Oracle Database 11g Express Edition/Enterprise Edition.

## Oracle Credentials for Lab

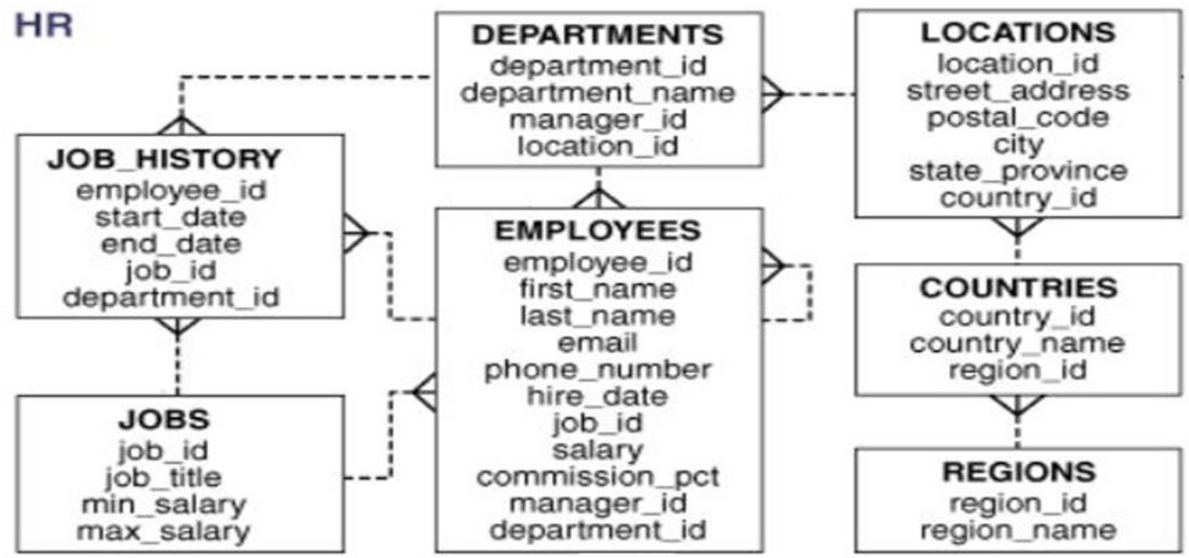
Enter the Url in your browser **http://172.17.10.114:8080/apex/**

Username **hr**

Password **hr**

## HR Schema

.




## HR Table Descriptions

Table **COUNTRIES**

| Name | Null? | Type |
|---|---|---|
| COUNTRY_ID | NOT NULL | CHAR(2) |
| COUNTRY_NAME | | VARCHAR2(40) |
| REGION ID | | NUMBER |

Table **DEPARTMENTS**

| Name | Null? | Type |
|---|---|---|
| DEPARTMENT_ID | NOT NULL | NUMBER(4) |
| DEPARTMENT_NAME | NOT NULL | VARCHAR2(30) |
| MANAGER ID | | NUMBER(6) |
| LOCATION ID | | NUMBER(4) |

Table **EMPLOYEES**

| Name | Null? | Type |
|---|---|---|
| EMPLOYEE_ID | NOT NULL | NUMBER(6) |
| FIRST_NAME | | VARCHAR2(20) |
| LAST NAME | NOT NULL | VARCHAR2(25) |
| EMAIL | NOT NULL | VARCHAR2(25) |
| PHONE_NUMBER | | VARCHAR2(20) |
| HIRE_DATE | | NOT NULL DATE |
| JOB_ID | | NOT NULL |
| VARCHAR2(10) | | |
| SALARY | | NUMBER(8,2) |
| COMMISSION PCT | | NUMBER(2,2) |
| MANAGER ID | | NUMBER(6) |
| DEPARTMENT_ID | | NUMBER(4) |

**Table JOBS**

| Name | Null? | Type |
|---|---|---|
| JOB ID | NOT NULL | VARCHAR2(10) |
| JOB_TITLE | NOT NULL | VARCHAR2(35) |
| MIN_SALARY | | NUMBER(6) |
| MAX SALARY | | NUMBER(6) |

**Table JOB_HISTORY**

| Name | Null? | Type |
|---|---|---|
| EMPLOYEE ID | NOT NULL | NUMBER(6) |
| START_DATE | NOT NULL | DATE |
| END_DATE | NOT NULL | DATE |
| JOB_ID | NOT NULL | VARCHAR2(10) |
| DEPARTMENT ID | | NUMBER(4) |

**Table LOCATIONS**

| Name | Null? | Type |
|---|---|---|
| LOCATION ID | NOT NULL | NUMBER(4) |
| STREET_ADDRESS | | VARCHAR2(40) |
| POSTAL_CODE | | VARCHAR2(12) |
| CITY | NOT NULL | VARCHAR2(30) |
| STATE_PROVINCE | | VARCHAR2(25) |
| COUNTRY_ID | | CHAR(2) |

**Table REGIONS**

| Name | Null? | Type |
|---|---|---|
| REGION_ID | NOT NULL | NUMBER |
| REGION_NAME | | VARCHAR2(25) |

## SUBQUERIES

### Using a Subquery to solve a Problem

Suppose you want to write a query to find out who earns a salary greater than Jones salary. To solve this problem, you need two queries: one query to find what Jones earns and second query to find who earns more than that amount.

You can solve this problem by combining the two queries, placing one query inside the other query. The inner query or the subquery returns a value that is used by the outer query or the main query. Using a subquery is equivalent to performing two sequential queries and using the result of the first query as the search value in the second query

### What is a Subquery?

A subquery is a SELECT statement that is embedded in a clause of another SELECT statement. You can build powerful statements out of simple ones by using subqueries. They can be very useful when you need to select rows from a table with a condition that depends on the data in the table itself.

You can place the subquery in a number of SQL clauses:

- WHERE clause
- HAVING clause
- FROM clause

### 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.

### Example

Which employees salary greater than Jones salary?

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

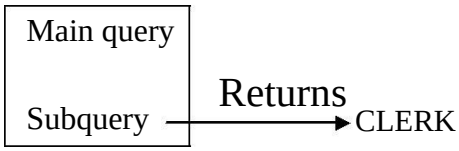
Display all employees name whose salary is equal to empoyeed_id 143.

## Guideline For using Sub Quires

- Enclose sub queries in parentheses.
- Place sub queries on the right side of the comparison condition.
- The ORDER BY clause in the sub query is not needed unless you are performing Top-N analysis.
- Use single-row operators with single-row sub queries, and use multiple-row operators with multiple-row sub queries.

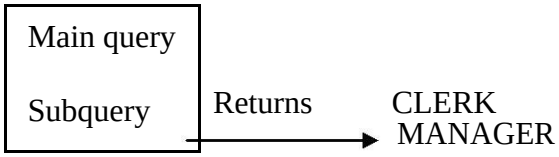
## Types of Subquery

- **Single-row subquery**
  Queries that return only one row from the inner SELECT statement.




- **Multiple-row subqueries:**

  Queries that returns more than one row from the inner SELECT statement




## Single Row Queries

- 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 first name ,job id whose job title is the same as that of employee 177.

```
Select first_name,job_id
From employees
Where job_id=(select job_id
From employees
Where employee_id=177);
```

**Example**

Display all employee records whose job title is the same as that of employee 141 and whose salary is greater than that of employee 143.

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

**Multiple-row subqueries:**

Queries that returns more than one row from the inner SELECT statement.

Subqueries that return more than one row are called multiple row subqueries. You use a multiple row operator (i.e IN) instead of a single row operator, with a multiple row subquery. The multiple row operator expects one or more values.

| operator | Meanings |
|---|---|
| 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 |

**Example**

Any operator example

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

All operator Example

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

**Using group functions in a subquery**

You can display data from a main query by using a group function in a subquery to return a single row. The subquery is in parentheses and is placed after the comparison operator.

**Example**

Display the employee name, job title and salary of all employees whose salary is equal to the minimum salary.

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

The MIN group function returns a single value (2100) to the outer query.

## Having clause with subqueries

You can use subqueries not only in the WHERE clause, but also in the HAVING clause. The Oracle Server executes the subquery, and the results are returned into the HAVING clause of the main query.

## Example

Display all the departments that have a minimum salary greater than that of department 20.

```
SELECT department_id, MIN(salary)
FROM Employees
GROUP BY department_id
HAVING MIN (salary)>
(SELECT MIN (salary)
FROM employees
WHERE department_id=20);
```

## Lab Exercise

1. Display the report of all those employees whose income is less than those who work in department numbers 50, 20, and 10. (3 Marks)
2. Write a query that displays the employee number and last name of all employees who work in a department with any employee whose last name contains ‘a’ and ‘u’. (3 Marks)
3. The HR department needs a report that displays the last name, department number, and job ID of all employees whose department location ID is 1700. (3 Marks)
4. Display the job number and job title of those employees whose maximum salary is greater than 'ST_MAN'. (3 Marks)
5. Show a report that displays the employee number, first name, and salary of all employees who earn more than the average salary. Sort the results according to salary in ascending order. (3 Marks).

6. Display all the departments that have a minimum salary greater than that of department 50 (3 marks).

## Lab Instructor:

Qazi Shuja ud Din (Riphah International University)
Email: qazi.shujauddin@riphah.edu.pk