Subqueries

UNIT 7.

Objectives

- To become familiar with subqueries.
- To learn new operators used with subqueries:
  - IN subquery
  - ANY-ALL
  - EXISTS

Introduction

- A subquery is a query that appears in another query, in its select list or WHERE or HAVING clauses.
- The query that contains the subquery is named the outer query (consulta externa).

```
SELECT nombre
FROM empleados
WHERE cuota <= (SELECT SUM(importe)
                FROM pedidos
                WHERE rep = numemp);
```

- A subquery in the FROM clause is a derived table. Both have their own rules.

Subquery rules

- Enclosed in ( )
- Has no final ;
- ORDER BY can be included only if TOP is included.
- Subqueries can be nested, up to 32 levels or resources limits.
- The subquery is calculated for each outer row.
- A subquery can include external references (columns from the outer query) → correlated subquery.
- But the outer query cannot refer to the subquery columns.
- When performing the subquery, columns are looked for in the source tables of the subquery, if not found, then they are seeked in the outer query.

Introduction

- Sometimes we obtain the same result with an INNER JOIN.
- Generally a subquery uses less memory than an INNER JOIN, but sometimes more processing time.
- Depending on where the subquery is placed within the outer query, its result may be limited:
  - One column and at most one row (a scalar query): in the select list and with a standard comparison.
  - One column and several rows (list of values): IN – ANY - ALL
  - Several columns and several rows: EXISTS

Unique result subqueries

They return only one column and at the most one row.

Are named scalar queries.

They can appear in:

- The select list of the outer query.
- In a WHERE or HAVING clause in a standard comparison.

```
SELECT nombre
FROM empleados
WHERE cuota <= (SELECT SUM(importe)
                FROM pedidos
                WHERE rep = numemp);
```

List of values

Some subqueries return a list of values, one column and several rows (or one or none).

They can appear in the WHERE or HAVING clause with:

- IN – NOT IN
- ANY – SOME
- ALL

List of values with IN

<expression> IN subquery

- Values returned by the subquery must be compatible with *expression*.
- If the subquery returns no rows, IN is FALSE
- If *expression* is one of the values included in the list, IN is TRUE.
- If *expression* does not match any of the values included in the list and there is no NULL in the list , IN is FALSE.
- If *expression* does not match any of the values included in the list and there is a NULL in the list , IN is NULL.

List of values with IN

```
SELECT *
FROM empleados
WHERE oficina IN (SELECT oficina
                  FROM oficinas
                  WHERE region = 'Este');
```

IN	numemp	nombre	oficina
TRUE	101	Antonio Viguer	12
FALSE	102	Alvaro Jaumes	21
TRUE	103	Juan Rovira	12
TRUE	104	José González	12
TRUE	105	Vicente Pantalla	13
TRUE	106	Luis Antonio	11
FALSE	107	Jorge Gutiérrez	22
FALSE	108	Ana Bustamante	21
NULL	109	María Sunta	NULL
NULL	110	Juan Victor	NULL

oficina
11
12
13
28
29

List of values with NOT IN

<expression> NOT IN subquery

- If the subquery returns no rows, NOT IN is TRUE
- If *expression* is one of the values included in the list, NOT IN is FALSE.
- If *expression* does not match any of the values included in the list and there is no NULL in the list , NOT IN is TRUE.
- If *expression* does not match any of the values included in the list and there is a NULL in the list , NOT IN is NULL. Be careful!

List of values with NOT IN

```
SELECT *
FROM Oficinas
WHERE oficina NOT IN (SELECT oficina
                      FROM empleados
                      WHERE edad=40);
```

We want to obtain the offices that have no 40 year old employees.

If we have an employee with no office (oficina is null), in the returned list we'll have a NULL value, so the outer query will return no office.

List of values with ANY/SOME

<expression> {=|<>|!=|>|>=
|! >|<|<=|!<} {ANY|SOME} subquery

With a modified comparison operator the comparison is evaluated for each value returned by the subquery.

If comparison is TRUE for at least one value, ANY is TRUE.

If comparison is false with all the values, ANY is FALSE.

If comparison is false with some values and NULL with the others, ANY is NULL.

If the subquery returns no rows, ANY is FALSE even if expression is null.

= ANY and IN are equivalent.

List of values with ANY/SOME

```
SELECT *
FROM empleados e1
WHERE cuota > ANY (SELECT cuota
                   FROM empleados e2
                   where e1.oficina = e2.oficina);
```

Returns the employees whose *cuota* is greater than the *cuota* of any employee who works in the same office.

Ie, returns the employees who do not have the smallest *cuota* of their office.

List of values with ALL

<expression> {=|<>|!=|>|>=
|! >|<|<=|!<} ALL subquery

The comparison is evaluated for each value returned by the subquery.

If the comparison is TRUE for all the values, ALL is TRUE.

If the comparison is false with some values and true with the others, ALL is FALSE.

If the comparison is NULL with some values, ALL is NULL.

If the subquery returns no rows, ALL is TRUE.

<> ALL and NOT IN are equivalent.

List of values with ALL

```
SELECT *
FROM empleados e1
WHERE cuota >= ALL (SELECT cuota
                    FROM empleados e2
                    where e1.oficina = e2.oficina);
```

Returns the employees whose *cuota* is greater or equal to all the *cuotas* of the employees who work in the same office. The employees who have the greatest cuota of his/her office.

Employees with no office will also be returned.

subquerys with EXISTS

In this case the subquery may return several columns and several rows.

WHERE [NOT] EXISTS subquery

The EXISTS test returns TRUE if the subquery returns at least one row, and EXISTS returns FALSE if the subquery returns no rows.

We do not care about null values.

Usually, the subquery is a correlated subquery. Be careful!

subquerys with EXISTS

```
SELECT *
FROM empleados
WHERE EXISTS (SELECT *
                    FROM pedidos
                    WHERE fab ='ACI‘ AND numemp = rep);
```

The employee is returned if there exists an order by him for a product manufactured by ACI.

If we do not include the outer reference (numemp=rep), all the employes are returned if an order with ACI exists, regardless of the employee.