TEMA 7 (Parte 2)

LENGUAJE DE MANIPULACIÓN DE DATOS (DML)

Sentencia SELECT SIMPLE

CURSO 23-24

ÍNDICE

ÍNDICE

1.- INTRODUCCIÓN

SENTENCIA SELECT (hoja de ruta)

◎ **CONSULTA DE DATOS SIMPLE**:
1. Introducción a la sentencia SELECT — **TEMA 4**
2. Obtención de datos de varias tablas:
   - ◆Subconsultas.
   - ◆Inclusión de varias tablas en clausula FROM.

   TEMA 7 (PARTE 2)
3. Funciones agregadas.

◎ **CONSULTA DE DATOS AVANZADA**:
1. Comparación ALL/ANY.
2. Comparación EXISTS.
3. Consultas agrupadas.
4. Concatenación con operador JOIN.

TEMA 8

1.- INTRODUCCIÓN

OBTENCIÓN DE DATOS DE VARIAS TABLAS

- En todos los ejemplos vistos hasta ahora hemos obtenido los datos de una sola tabla.
- En la mayoría de las ocasiones para realizar una consulta se necesitan datos de varias tablas.
- Existen distintas formas de obtener datos de varias tablas en una consulta SQL:
  - **Utilizando SUBCONSULTAS.**
  - **Incluyendo varias tablas en el FROM**
    - Combinación o concatenación de tablas.
    - FROM compuesto.
- Se pueden incluso utilizar simultáneamente.

ÍNDICE

2.- SUBCONSULTAS

- En ocasiones para realizar una consulta se necesitan los datos devueltos por otra consulta.
  - Una subconsulta no es más que una sentencia select que se utiliza dentro de otra select[1].

    subconsulta::= (sentencia_ select)
    - Una subconsulta siempre va entre paréntesis
  - Las subconsultas dan potencia a las condiciones de la clausula WHERE. Es posible utilizar subconsultas en:
    - *comparacion_operador*
    - *comparacion_in*
    - *comparacion_between*
    - *comparacion_all_any*
    - *comparacion_exists*

Comprobad la sintaxis

1: o dentro de otras sentencias

2.- SUBCONSULTAS

- **EJEMPLO:** Dadas las tablas EMPLE y DEPART:
  1. Si nos plantean la siguiente consulta:

     Obtener el apellido de los empleados con oficio 'ANALISTA'

```
SELECT  APELLIDO
FROM    EMPLE
WHERE   OFICIO= 'ANALISTA'
```

  - Comparamos en el WHERE con un valor constante conocido previamente ('ANALISTA')
  - Es lo que hemos hecho hasta ahora.

2.- SUBCONSULTAS

◎ EJEMPLO (cont):

2. Si nos plantean ahora la siguiente consulta:

Obtener el apellido de los empleados con el mismo oficio que el empleado de número 7788.

- Hemos de buscar el valor con el que queremos comparar en la base de datos (se busca con otra consulta -SUBCONSULTA-)

CONSULTA PRINCIPAL

```
SELECT APELLIDO
FROM   EMPLE
WHERE  OFICIO=
```

SUBCONSULTA

```
(SELECT  OFICIO
FROM     EMPLE
WHERE    EMP_NO=7788)
```


2.- SUBCONSULTAS

CONSULTA PRINCIPAL

```
SELECT APELLIDO
FROM   EMPLE
WHERE  OFICIO=
```

SUBCONSULTA

```
(SELECT  OFICIO
FROM     EMPLE
WHERE    EMP_NO=7788)
```

◎ FUNCIONAMIENTO:

- La subconsulta forma parte del WHERE de la consulta principal
  - Recordad como funciona la sentencia SELECT
  - Hemos de pensar que la subconsulta se ejecuta para cada fila de la consulta principal
  - Como si fuera un bucle y la subconsulta fueran las instrucciones de dentro del bucle.

2.- SUBCONSULTAS

CONSULTA PRINCIPAL

```
SELECT APELLIDO
FROM   EMPLE
WHERE  OFICIO=
```

SUBCONSULTA

```
(SELECT  OFICIO
FROM     EMPLE
WHERE    EMP_NO=7788)
```

EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
7369	SÁNCHEZ	EMPLEADO	7902	17/12/1980	1040	(null)	20
7499	ARROYO	VENDEDOR	7698	20/02/1980	2080	390	30
7521	SALA	VENDEDOR	7698	22/02/1981	1625	650	30
7566	JIMÉNEZ	DIRECTOR	7839	02/04/1981	3867	(null)	20
7654	MARTÍN	VENDEDOR	7698	29/09/1981	1625	1820	30
7698	NEGRO	DIRECTOR	7839	01/05/1981	3705	(null)	30
7782	CEREZO	DIRECTOR	7839	09/06/1981	3185	(null)	10
7788	GIL	ANALISTA	7566	09/11/1981	3900	(null)	20
7839	REY	PRESIDENTE	(null)	17/11/1981	6500	(null)	10
7844	TOVAR	VENDEDOR	7698	08/09/1981	1950	0	30
7876	ALONSO	EMPLEADO	7788	23/09/1981	1430	(null)	20
7900	JIMENO	EMPLEADO	7698	03/12/1981	1235	(null)	30
7902	FERNÁNDEZ	ANALISTA	7566	03/12/1981	3900	(null)	20
7934	MUÑOZ	EMPLEADO	7782	23/01/1982	1690	(null)	10

2.- SUBCONSULTAS

CONSULTA PRINCIPAL

```
SELECT APELLIDO
FROM   EMPLE
WHERE  OFICIO=
```

SUBCONSULTA

```
(SELECT  OFICIO
FROM     EMPLE
WHERE    EMP_NO=7788)
```

◎ FUNCIONAMIENTO:

- ¿Pero si la subconsulta devuelve el mismo valor siempre por qué se ejecuta para cada fila?
  - En realidad los SGBD suelen optimizar la ejecución de las consultas con subconsultas.
    - Si la subconsulta devuelve el mismo valor lo suelen detectar y guardan el valor internamente.
  - **Consejo didáctico**: para entender bien el funcionamiento es mejor pensar que la subconsulta se ejecuta de nuevo.

2.- SUBCONSULTAS
2.1.- CONSIDERACIONES IMPORTANTES

◎ RESULTADO DE LA SUBCONSULTA:

- Dependiendo del tipo de comparación en la que aparecen están sometidas a ciertas restricciones.
- EN GENERAL el RESULTADO DE UNA SUBCONSULTA debe ser coherente con aquello que se espera comparar.

2.1.- CONSIDERACIONES IMPORTANTES

◎ RESULTADO DE LA SUBCONSULTA:

- **1**: si se espera comparar una columna con un valor la subconsulta debe devolver una sola columna y una sola fila (un solo valor)

```
SELECT  APELLIDO
FROM    EMPLE
WHERE   OFICIO=  (SELECT  OFICIO, APELLIDO
                  FROM    EMPLE
                  WHERE   EMP_NO=7788)
```

ERROR!

```
SELECT  APELLIDO
FROM    EMPLE
WHERE   OFICIO=  (SELECT  OFICIO
                  FROM    EMPLE
                  WHERE   EMP_NO=7788)
```

Solo puede devolver una fila (un valor) Sino dará ERROR!

2.1.- CONSIDERACIONES IMPORTANTES

◎ RESULTADO DE LA SUBCONSULTA:

- **2**: si se espera comparar una columna con un conjunto de valores la subconsulta debe devolver una sola columna, pero puede devolver varias filas (valores)

```
SELECT  APELLIDO
FROM    EMPLE
WHERE   OFICIO IN (SELECT  OFICIO, APELLIDO
                   FROM    EMPLE
                   WHERE   APELLIDO='GIL')
```

ERROR!

```
SELECT  APELLIDO
FROM    EMPLE
WHERE   OFICIO IN (SELECT  OFICIO
                   FROM    EMPLE
                   WHERE   APELLIDO='GIL')
```

Puede devolver mas de una fila (varios valores)

2.1.- CONSIDERACIONES IMPORTANTES

◎ RESULTADO DE LA SUBCONSULTA:

- **3**: es posible comparar varias columnas a la vez[2]
  - La subconsulta debe devolver tantas columnas como se quiera comparar.
  - El poder devolver varias filas o una sola depende del tipo de comparación que se realice.

```
SELECT  APELLIDO
FROM    EMPLE
WHERE   (OFICIO, SALARIO) = (SELECT OFICIO
                             FROM    EMPLE
                             WHERE   EMP_NO=7788)
```

ERROR!

```
SELECT  APELLIDO
FROM    EMPLE
WHERE   (OFICIO, SALARIO) = (SELECT OFICIO, SALARIO
                             FROM    EMPLE
                             WHERE   EMP_NO=7788)
```

Sólo puede devolver una fila

2: SOLO EN ORACLE. No estándar

2.1.- CONSIDERACIONES IMPORTANTES

SUBCONSULTAS VS CONSULTA PRINCIPAL:

- Una consulta (principal) no puede hacer referencia a los datos obtenidos en una subconsulta (NO TIENE ACCESO).

```
SELECT   APELLIDO, E2.OFICIO
FROM     EMPLE E1
WHERE    OFICIO IN (SELECT  OFICIO
                    FROM    EMPLE E2
                    WHERE   APELLIDO='GIL')
```

- Sin embargo si es posible hacer referencia a los campos de la consulta principal en una subconsulta.

```
SELECT   E1.APELLIDO, E1.SALARIO, E1.DEPT_NO
FROM     EMPLE E1                                  OBSERVAD!!!
WHERE    SALARIO IN  (SELECT   SALARIO
                      FROM     EMPLE E2
                      WHERE    E2.DEPT_NO<>E1.DEPT_NO)
```


2.1.- CONSIDERACIONES IMPORTANTES

SUBCONSULTAS VS CONSULTA PRINCIPAL:

- Este mecanismo es muy potente ya que se evalúa o ejecuta la subconsulta para cada valor de fila de la consulta principal
  - Para cada FILA DE EMPLEADO de E1 se evalúa la subconsulta tomando su E1.DEPT_NO

```
SELECT   E1.APELLIDO, E1.SALARIO, E1.DEPT_NO
FROM     EMPLE E1
WHERE    SALARIO IN  (SELECT   SALARIO
                      FROM     EMPLE E2
                      WHERE    E2.DEPT_NO<>E1.DEPT_NO)
```

Analiza lo que obtiene esta consulta...

- **SE SUELE UTILIZAR**:
  - Combinado con funciones agregadas
    - LO VEREMOS AL FINAL DEL TEMA.
  - Al utilizar los comparadores ALL/ANY y EXISTS (Tema 8)

ÍNDICE

3.- COMBINACIÓN DE TABLAS

Inclusión de varias tablas en la clausula FROM

```
SELECT [ALL|DISTINCT]
       [comalista_ref_columna | *]
FROM   comalista_ref_tabla ←
[WHERE condicion]
[ORDER BY comalista_ref_columna [DESC|ASC]]
```

```
ref_tabla::= tabla [alias]
```

- Se indican todas las tablas de las que se desea obtener información en la cláusula FROM separadas por comas

```
SELECT   *
FROM     R, S
```

3.- COMBINACIÓN DE TABLAS

- Al incluir varias tablas en la clausula FROM se realiza un **PRODUCTO CARTESIANO** entre las tablas:

```
SELECT   *
FROM     R, S
```

- ¿Qué resultado devuelve esta consulta?
  - **COLUMNAS**:
    - Todas las columnas de R y las de S
    - Si existen columnas en R y S con el mismo nombre se diferencian y referencian de la forma:

```
nom_tabla. nom_columna
```

    - **Por ejemplo**:

```
R.codigo
S.codigo
```


3.- COMBINACIÓN DE TABLAS

- ¿Qué resultado devuelve esta consulta?
  - **FILAS**:
    - Todas las filas resultantes de combinar o concatenar cada fila de R con cada fila de S
    - Intuitivamente puede verse:

Filas de R	Filas de S	Filas resultado
a	x	(a, x)
b	y	(a, y)
c	z	(a, z)
d		(b, x)
		(b, y)
		(b, z)
		(c, x)
		(c, y)
		(c, z)
		(d, x)
		(d, y)
		(d, z)

3.- COMBINACIÓN DE TABLAS

- **Ejemplo**:

```
SELECT   *
FROM     PACIENTE, HOSPITAL
```

PACIENTE

dni	nombre	dir	numss	cod_hosp
20450120A	Juan Pérez	Cuenca 20	46123456	H003
12904569P	José Rodríguez	Mar 35		
35778843S	María Gutiérrez		46555789	H000

HOSPITAL

cod_hosp	nombre	dir
H000	La salud	
H003	La Fe	

RESULTADO

dni	nombre	dir	numss	cod_hosp	cod_hosp	nombre	dir
20450120A	Juan Pérez	Cuenca 20	46123456	H003	H000	La salud	
20450120A	Juan Pérez	Cuenca 20	46123456	H003	H003	La Fe	
12904569P	José Rodríguez	Mar 35			H000	La salud	
12904569P	José Rodríguez	Mar 35			H003	La Fe	
35778843S	María Gutiérrez		46555789	H000	H000	La salud	
35778843S	María Gutiérrez		46555789	H000	H003	La Fe	

3.- COMBINACIÓN DE TABLAS

- ¿Qué sentido tiene realizar esta consulta?
  - A primera vista no sirve de nada combinar todas la filas de R con todas las de S.
  - En realidad sólo tiene sentido cuando ADEMÁS las tablas se ligan adecuadamente en el WHERE.

```
SELECT   *
FROM     PACIENTE, HOSPITAL
WHERE    PACIENTE.cod_hosp=HOSPITAL.cod_hosp
```

  - Este mecanismo se suele utilizar para:
    - Buscar información relacionada en otra tabla.
      - Normalmente a través de una clave ajena.
    - Comparar información entre 2 tablas

3.- COMBINACIÓN DE TABLAS

PACIENTE

Dni	Nombre	Dir	NumSS	Cod_Hosp
20450120A	Juan Pérez	Cuenca 20	46123456	H003
12904569P	José Rodríguez	Mar 35		
35778843S	María Gutiérrez		46555789	H000

HOSPITAL

Cod_Hosp	Nombre	Dir
H000	La salud	
H003	La Fe	

```
SELECT   *
FROM     PACIENTE, HOSPITAL
```

Dni	Nombre	Dir	NumSS	Cod_Hosp	Cod_Hosp	Nombre	Dir
20450120A	Juan Pérez	Cuenca 20	46123456	H003	H000	La salud	
20450120A	Juan Pérez	Cuenca 20	46123456	H003	H003	La Fe	
12904569P	José Rodríguez	Mar 35			H000	La salud	
12904569P	José Rodríguez	Mar 35			H003	La Fe	
35778843S	María Gutiérrez		46555789	H000	H000	La salud	
35778843S	María Gutiérrez		46555789	H000	H003	La Fe	

```
SELECT   *
FROM     PACIENTE, HOSPITAL
WHERE    PACIENTE.Cod_Hosp=HOSPITAL.Cod_Hosp
```

Dni	Nombre	Dir	NumSS	Cod_Hosp	Cod_Hosp	Nombre	Dir
20450120A	Juan Pérez	Cuenca 20	46123456	H003	H003	La Fe	
35778843S	María Gutiérrez		46555789	H000	H000	La salud	

3.- COMBINACIÓN DE TABLAS

- ¿Qué hemos conseguido?

BUSCAR INFORMACIÓN ASOCIADA EN OTRA TABLA.

- Obtener EN UNA SOLA CONSULTA para cada fila de la tabla PACIENTE su información de la tabla HOSPITAL asociada.
- Es decir, hemos obtenido para cada paciente la información completa de su hospital asociado (no solo el código).

```
SELECT   *
FROM     PACIENTE, HOSPITAL
WHERE    PACIENTE.cod_hosp=HOSPITAL.cod_hosp
```

Dni	Nombre	Dir	NumSS	Cod_Hosp	Cod_Hosp	Nombre	Dir
20450120	Juan Pérez	Cuenca 20	46123456	H003	H003	La Fe	
35778843	María Gutiérrez		46555789	H000	H000	La salud	

3.- COMBINACIÓN DE TABLAS

- **EJEMPLO**: Dadas las tablas EMPLE Y DEPART:
  - Obtener para cada empleado sus datos y los del departamento al que pertenecen.

```
SELECT   *
FROM     EMPLE E, DEPART D
WHERE    E.DEPT_NO=D.DEPT_NO
```

  - Prueba la consulta en SQL Developer:
    - ¿Qué COLUMNAS se muestran como resultado?
    - ¿Qué filas?

3.- COMBINACIÓN DE TABLAS

- **CONSIDERACIONES FINALES**:
  - Conviene observar que las consultas siguientes son equivalentes:

```
SELECT   *
FROM     PACIENTE, HOSPITAL
WHERE    PACIENTE.cod_hosp=HOSPITAL.cod_hosp
```

```
SELECT   *
FROM     HOSPITAL, PACIENTE
WHERE    HOSPITAL.cod_hosp=PACIENTE.cod_hosp
```

  - Simplemente se mostrarían los campos en distinto orden.

3.- COMBINACIÓN DE TABLAS

- **CONSIDERACIONES FINALES**:
  - Este mecanismo puede repetirse cuantas veces lo necesitemos en una consulta.
  - De esta forma podemos encontrarnos con numerosas tablas en la clausula FROM de una consulta.

```
SELECT  *
FROM    EMPLE E1, DEPART D, EMPLE E2
WHERE   E1.DEPT_NO=D.DEPT_NO AND
        E1.DIR=E2.EMP_NO
```

  - Prueba la consulta en SQL Developer
    - ¿Qué COLUMNAS se muestran como resultado?
    - ¿Qué filas?

ÍNDICE

4.- FUNCIONES AGREGADAS

```
ref_columna::= atributo|
               constante|
               expresion_aritmetica|
               ref_funcion ←
```

```
ref_funcion::=  COUNT (*)|  ← Funciones Agregadas (SQL Estándar)
                {SUM| AVG| MAX| MIN| COUNT} ([ALL|DISTINCT] ref_columna)|
                funcion_Oracle  ← TEMA 5
```

- Una función agregada devuelve **un ÚNICO valor** que se calcula a partir de TODAS LAS FILAS resultado de una SELECT:
  - COUNT (*): devuelve el número total de filas (CASO ESPECIAL)

```
SELECT   COUNT(*)
FROM     EMPLE
```

  - {SUM| AVG| MAX| MIN| COUNT} ([ALL|DISTINCT] *ref_columna*): aplica sobre ref_columna una de las funciones siguientes (sigue →)

4.- FUNCIONES AGREGADAS

- **Continúa…**
  - {SUM| AVG| MAX| MIN| COUNT} ([ALL|DISTINCT] *ref_columna*):
    - SUM: SUMA de los valores de *ref_columna*.
      - *ref_columna* debe ser numérico
    - AVG: MEDIA de los valores de *ref_columna*.
      - *ref_columna* debe ser numérico
    - MAX/MIN: MÁXIMO/MÍNIMO de los valores de *ref_columna*.
      - *ref_columna* NO necesariamente numérico
    - COUNT: numero de valores que toma *ref_columna*.
      - *ref_columna* NO necesariamente numérico
  - ALL: opción por defecto.
    - Se realiza el cálculo con todos los valores.
  - DISTINCT: indica que los valores repetidos sean eliminados antes de que se realice el cálculo correspondiente.

4.- FUNCIONES AGREGADAS

FUNCIONAMIENTO BÁSICO:

1. Los cálculos se realizan después de aplicar las condiciones del WHERE.
2. Los valores nulos son eliminados antes de realizar los cálculos.
3. Si el número de filas resultantes de la selección es 0:
   - La función COUNT devuelve el valor 0
   - El resto de funciones devuelve el valor nulo.

- **Ver ejemplos básicos**
  - Esquema EMPLE Y DEPART.

4.- FUNCIONES AGREGADAS

FUNCIONAMIENTO BÁSICO:

4. Cuando una sentencia SELECT no contiene la cláusula GROUP BY, en la selección no pueden aparecer a la vez referencias a atributos y a funciones agregadas.
   - Ya que las funciones van a devolver un único valor.

```
SELECT  APELLIDO, AVG (SALARIO)
FROM    EMPLE
```
→ ERROR!

```
SELECT  MIN(SALARIO), MAX(SALARIO), AVG (SALARIO)
FROM    EMPLE
```
→ OK!

4.- FUNCIONES AGREGADAS

CONSIDERACIONES IMPORTANTES:

- Una función agregada toma su significado porque actúa con un grupo o conjunto de filas (varios valores).
- **No tiene sentido utilizar una función agregada en la cláusula WHERE** de consultas sin la cláusula GROUP BY
  - En este contexto se dispone de una sola fila (un solo valor).

```
SELECT  EMP_NO, SALARIO
FROM    EMPLE
WHERE   SALARIO=MAX(SALARIO)
```
→ ERROR!

```
SELECT  EMP_NO, SALARIO
FROM    EMPLE
WHERE   SALARIO = (SELECT  MAX(SALARIO)
                   FROM    EMPLE)
```
→ OK!

¿y si sólo quisiéramos el salario?

- Lo mismo se aplica al anidar funciones agregadas

```
SELECT  SUM (MAX(SALARIO))
FROM    EMPLE
```
→ ¿Qué devuelve?

4.- FUNCIONES AGREGADAS

CONSIDERACIONES IMPORTANTES:

- En resumen, son errores comunes:

```
SELECT  APELLIDO, AVG (SALARIO)
FROM    EMPLE
```

```
SELECT  EMP_NO, SALARIO
FROM    EMPLE
WHERE   SALARIO=MAX(SALARIO)
```

```
SELECT  SUM (MAX(SALARIO))
FROM    EMPLE
```

- **Sin la clausula GROUP BY**
  - Veremos que al utilizar GROUP BY la cosa cambiará

4.- FUNCIONES AGREGADAS

CONSIDERACIONES IMPORTANTES:

- Es bastante común combinar las funciones agregadas con subconsultas para realizar consultas avanzadas

```
SELECT DNOMBRE
FROM   DEPART D  <-------------- OBSERVAD!!!
WHERE 2=  (SELECT COUNT(*)          |
           FROM   EMPLE E           |
           WHERE  E.DEPT_NO=D.DEPT_NO AND
                  OFICIO='ANALISTA')
```

- Prueba la consulta en SQL Developer
  - ¿Qué COLUMNAS se muestran como resultado?
  - ¿Qué filas?
- Aunque algunas de estas consultas las haremos más fácilmente con GROUP BY (tema 8)