SQL - DML

I. Introducción SQL

➔ *QUÉ ES*

- **SQL (Structured Query Language, *Lenguaje Estructurado de Consultas*)**: Lenguaje que permite expresar operaciones diversas (aritméticas, combinatorias, lógicas, selección y ordenación) con datos almacenados en **bases de datos relacionales**.
- Originalmente, SQL se llamaba SEQUEL y fue diseñado e implementado por IBM Research como interfaz para un SGBD relacional experimental denominado SYSTEM-R
- Actualmente es un estándar ANSI ISO (SQL:2003), pero:
  - Las compañías comerciales crean sus propios *dialectos*
- Ha sido aceptado como lenguaje *de facto*.
- Está basado en el álgebra relacional

I. Introducción SQL

➔ *CÓMO SE USA*

- Las peticiones para actuar sobre los datos se expresan mediante **sentencias**. Éstas deben escribirse de acuerdo con las reglas sintácticas y semánticas del lenguaje.
- **Sentencias SQL**: permiten enunciar operaciones sobre tablas. Existen varios tipos según el tipo de operación que expresan:
  - **DML:** Lenguaje de Manipulación de Datos (*Data Manipulation Language*)**:** Sentencias que permiten realizar **consultas** y **actualizaciones** de datos (inserción, borrado y modificación de filas):
    - ***SELECT, INSERT, UPDATE, DELETE***.
  - **DDL:** Lenguaje de Definición de Datos (*Data Definition Language*): Sentencias que permiten **definir** nuevos objetos (tablas, índices, claves, etc) o **destruir** los ya existentes:
    - ***CREATE, DROP, ALTER***.
  - **DCL:** Sentencias de Control de Datos (*Data Control Language*): Sentencias que permiten controlar aspectos varios, como, por ejemplo, la **autorización de acceso** a los datos.
    - ***GRANT, REVOKE***.

I. Introducción SQL

- En resumen, **SQL** es un lenguaje utilizado por SGBD relacionales que permite:
  - Consultar y actualizar datos (DML)
  - Definir y destruir objetos de la base de datos (DDL)
  - Conceder y denegar autorizaciones para usar estos objetos (DCL)

II. Elementos de Lenguaje SQL

- **Base de Datos Relacional**: el usuario la percibe como un conjunto de **tablas**.
- Correspondencia relación – tabla:

RELACIÓN	TABLA
Tupla	Fila
Atributo	Columna
Grado	Nº columnas
Cardinalidad	Nº filas

- Diferencia entre **Tabla** (SQL) y **Relación** (Modelo Relacional)
  - **Relación**: Conjunto de tuplas
  - **Tabla**: Multiconjunto de filas

II. SQL

➔ *Elementos de Lenguaje*

- **Componentes sintácticos de una sentencia SQL:**
  1. **Palabras reservadas:** Tienen un significado predefinido en SQL. Todas las sentencias comienzan con una de ellas. P.ej: ***SELECT, WHERE, INTO,...***
  2. **Nombres de tablas y columnas:** Identifican a las tablas y columnas. Se definen en la creación de las tablas con la sentencia ***CREATE***.
  3. **Constantes**: Sentencias de caracteres (letras, números, signos) que representan un valor determinado. Si no es numérico debe ir entre comillas. P.ej: 125, 'AB'
  4. **Signos delimitadores**: Signos especiales que aparecen en las sentencias y no se corresponden con los elementos anteriores. Limita y separan a estos últimos. P.ej: Espacio en blanco, paréntesis, comas, etc.

  - SQL **NO** es sensible al caso.
    - SELECT ≈ SeLEcT

III. SQL

➔ *Tipos de Datos*

- Los valores contenidos en una columna de una base de datos relacional deben ser homogéneos. Esto se consigue asignándole un **tipo de datos** en el momento de crear las columnas de las bases de datos.
- SQL proporciona tipos de datos predefinidos, aunque también permite definir nuevos tipos de datos conocidos como *Tipos de Datos Distintos definidos por el Usuario*.
- En el momento de asignar un tipo de datos a una columna, se está definiendo:
  - **Conjunto** de todos los **valores** posibles que puede tomar la columna.
  - Las **operaciones** que se pueden realizar con los valores de la columna.

III. SQL

➔ *Tipos de Datos*

- El estándar SQL ANSI/ISO especifica varios tipos de datos que pueden ser almacenados en una base de datos basada en SQL y manipulados por el lenguaje SQL.
- Estos tipos de datos son un conjunto mínimo.
- Casi todos los productos comerciales soporta este conjunto mínimo. En la mayoría de los casos definen un conjunto más amplio de tipos de datos.

III. SQL

➔ *Tipos de Datos*

- **NUMÉRICOS**
  - *Números enteros*
    - INT[EGER]: Normalmente 32bits (-2147483648 a +2147483647)
    - SMALLINT: Normalmente 16bits (–32768 y +32767)
  - *Números en coma flotante*:
    - FLOAT (precis)
    - REAL
    - DOUBLE PRECISION
  - *Números con formato*:
    - DECIMAL (precis[,decim])
    - NUMERIC (precis[,decim]), donde *precis* es el número total de dígitos decimales y *decim* es la escala, es decir, el número de dígitos después del punto decimal.
    - NUMBER (precis [,decim]) es un tipo básico equivalente a los anteriores.

III. SQL

➔ *Tipos de Datos*

- **TEXTO (caracteres)**
  - *CHAR(n)*: admite letras, números o caracteres especiales. Internamente, se almacena en formato de longitud fija. Su longitud máxima es de 2000 caracteres. CHAR es equivalente a CHAR(1).
  - *VARCHAR2(n)*: admite letras, números o caracteres especiales. Se almacena en un formato de longitud variable. Su longitud máxima es de 4000 caracteres.
  - Las cadenas se representan entre comillas simples.

III. SQL

➔ *Tipos de Datos*

- **DE TIEMPO**
  - *DATE*: campo de longitud fija de 7 bytes, que se utiliza para almacenar datos temporales, lo que incluye la fecha (día del mes, mes y año) y hora (hora, minutos y segundos, e incluso fracciones de segundo). Es importante tener esto en cuenta cuando se desea comparar dos fechas.
    El formato predeterminado de fecha es DD-MON-YY, donde DD es el día, MON es el mes e YY son los dos últimos dígitos del año.

    Permite almacenar fechas desde el 1 de enero del 4712 a.C. hasta el 31 de diciembre del 4712 d.C.

III. SQL

➔ *Tipos de Datos*

- **BINARIOS**
  - *BLOB:* Es un objeto binario de gran tamaño, siendo el tamaño máximo 4 GB (gigabytes). Normalmente un blob se utiliza para almacenar una imagen, datos de voz, o cualquier otro bloque de datos grande no estructurado.

III. SQL

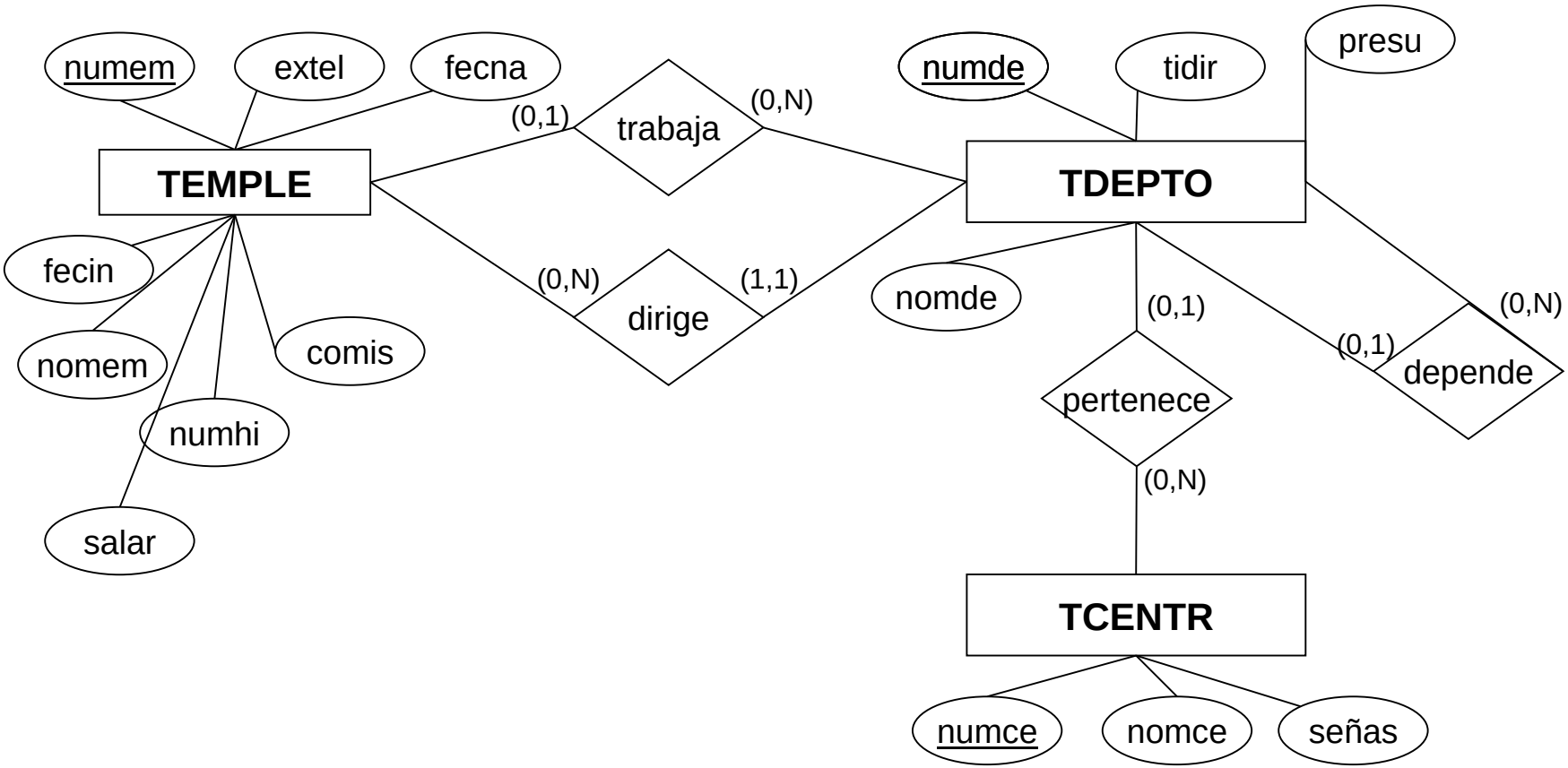
➔ *Tipos de Datos*

- **IMPORTANTE**: Todos los dominios incluyen el valor **nulo**. En el momento de crear una columna de una tabla se puede indicar si se desea permitir o no ese valor para la columna.

- **Operaciones de COMPARACIÓN:**
  - **Operadores**: =, <,>,<>,<=, >=
    - Si uno de los valores que se compara es Nulo, el resultado de la comparación también lo es.
    - Sólo se pueden comparar valores homogéneos.
    - Todos los tipos de datos numéricos pueden ser comparados unos con otros. Lo mismo para tipos de datos de texto.
    - Dos cadenas vacías se consideran iguales.

IV. SQL como DML
➔ *BD*

- Para los ejemplos y ejercicios se utilizará una base de datos de empleados. A continuación se indica el MER correspondiente:



IV. SQL como DML

➔ *BD*

- Las tablas de la BD son las siguientes:

TDEPTO							
	NUMDE	**NUMCE**	**DIREC**	**TIDIR**	**PRESU**	**DEPDE**	**NOMDE**
	PK						
		FK.TCENTR	FK.TEMPLE			FK.TDEPTO	

SIGNIFICADO DE LAS COLUMNAS	
NUMDE	Número identificador del departamento (Integer)
NUMCE	Número del centro de trabajo donde está el departamento (Integer)
DIREC	Número del empleado que es director del departamento (Integer)
TIDIR	Tipo de director (Char) (P: en Propiedad, F: en Funciones)
PRESU	Presupuesto anual del departamento (Decimal), en MILES de EUROS
DEPDE	Número del departamento del que depende (Integer)
NOMDE	Nombre del departamento (Varchar2)

IV. SQL como DML

➔ *BD*

- Las tablas de la BD son las siguientes:

TEMPLE									
	NUMEM	**NUMDE**	**EXTEL**	**FECNA**	**FECIN**	**SALAR**	**COMIS**	**NUMHI**	**NOMEM**
	PK								
		FK.TDEPTO							

SIGNIFICADO DE LAS COLUMNAS	
NUMEM	Número identificador del empleado (Integer)
NUMDE	Número del departamento al que está asignado (Integer)
EXTEL	Extensión telefónica correspondiente al empleado (Smallint)
FECNA	Fecha de nacimiento (Date)
FECIN	Fecha de ingreso (Date)
SALAR	Salario mensual (Decimal), en euros
COMIS	Comisión mensual (Decimal), en euros
NUMHI	Número de hijos (Smallint)
NOMEM	Nombre del empleado (Varchar2)

IV. SQL como DML

➔ *BD*

- Las tablas de la BD son las siguientes:

TCENTR			
	NUMCE	**NOMCE**	**SEÑAS**
	PK		

SIGNIFICADO DE LAS COLUMNAS	
NUMCE	Número identificador del centro (Integer)
NOMCE	Nombre del centro (Varchar2)
SEÑAS	Dirección del local (Varchar2)

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

- Permite ***extraer*** y ***visualizar*** datos de una o más tablas. No modifica el contenido de la tabla.
- Consta básicamente de las cláusulas `SELECT` y `FROM` como obligatorias y de otras varias cláusulas opcionales:

```
SELECT [DISTINCT|ALL] {*|[exprColumna[AS nuevoNombre]][,…]}
FROM nombreTabla [alias][,…]
[WHERE predicado]
[GROUP BY listaColumnas] [HAVING condición]
[ORDER BY {[nombreColumna[DESC]][,…]}]
```

  - ***exprColumna***: representa un nombre de columna o una expresión
  - ***nombreTabla***: Es el nombre de tabla
  - ***alias***: abreviatura opcional para *nombreTabla*
  - ***predicado***: Condición cuyo valor es Verdadero o Falso

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

- Cláusulas
  - **FROM**: especifica la tabla o tablas que hay que usar. Operador Producto cartesiano (**X**)
  - **WHERE**: filtra las filas (una a una) de acuerdo al predicado. Operador de selección (σ)
  - **SELECT**: especifica qué columnas deben aparecer en la salida. Operador de proyección (Π)
  - **ORDER BY**: especifica el orden de salida.
  - **GROUP BY**: forma grupos de filas que tengan el mismo valor de columna. Operador de Agregación (**G**)
  - **HAVING**: filtra los grupos de acuerdo con alguna condición

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

TCENTR			
	NUMCE	**NOMCE**	**SEÑAS**
	PK		

- Ejemplos:
  - *Listar los datos de los centros*

```
SELECT * FROM TCENTR
```

  - *Listar los nombres de los centros*

```
SELECT NOMCE FROM TCENTR
```

```
SELECT NOMCE AS NombreCentro FROM TCENTR
```

  - *Listar los datos de los centros cuyo nombre es UVIGO*

```
SELECT NOMCE
FROM TCENTR
WHERE NOMCE='UVIGO'
```

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

- **ORDER BY**: especifica el orden de salida.
  - En caso de no indicar esta cláusula => orden impredecible
  - Si se indica:
    - Las filas resultantes se presentan ordenadas por valores crecientes en la(s) columna(s) de ordenación. Si se desea ordenación decreciente es preciso aplicar la cláusula `DESC`.
    - Si hay valores **nulos**, se presentan después de todos los demás.
    - Si dos filas tienen valores iguales en la columna de ordenación => orden impredecible.
    - Es posible indicar el número de columna, en lugar de su nombre.

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

- **ORDER BY**: Ejemplos
  - *Listar los nombres y señas de los centros ordenados por nombre*

```
SELECT NOMCE,SEÑAS FROM TCENTR ORDER BY NOMCE
```

```
SELECT NOMCE,SEÑAS FROM TCENTR ORDER BY 1
```

  - *Listar los nombres y señas de los centros ordenados por nombre de forma descendente*

```
SELECT NOMCE,SEÑAS FROM TCENTR ORDER BY NOMCE DESC
```

```
SELECT NOMCE,SEÑAS FROM TCENTR ORDER BY 1 DESC
```

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

- *Ejercicios I*:
  1. Obtener los nombres de los empleados que trabajan en el departamento 121.
  2. Extraer todos los datos del departamento 121.
  3. Obtener los nombres y sueldos de los empleados con más de 3 hijos por orden alfabético.
  4. Obtener la comisión, nº de departamento y nombre, de los empleados cuyo salario es inferior a 1900 euros, ordenándolos por departamento en orden creciente, y por comisión en orden decreciente dentro de cada de cada departamento.
  5. Igual que la anterior, pero las columnas resultantes han de llamarse: COMISION, DEPTO y EMPLEADO.
  6. Obtener por orden alfabético los nombres de los departamentos cuyo presupuesto sea superior a 20000 euros.

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

- Cláusula ***DISTINCT***
  - *Obtener los números de los departamentos donde trabajan empleados cuyo salario sea inferior a 2500 euros.*

```
SELECT NUMDE FROM TEMPLE WHERE SALAR<2500 ORDER BY NUMDE
```

EXISTEN FILAS REPETIDAS EN EL RESULTADO

RECORDAR:

Diferencia entre Tabla (SQL) y Relación (Modelo Relacional)

Relación: Conjunto de tuplas

Tabla: Multiconjunto de filas

NUMDE
110
111
111
121

- La palabra reservada ***DISTINCT*** antes de la lista de columnas del SELECT, eliminará las filas duplicadas antes de devolver el resultado.

```
SELECT DISTINCT NUMDE FROM TEMPLE WHERE SALAR<2500 ORDER BY NUMDE
```

IV. SQL como DML

➔ *Consultas Simples:* ***SELECT***

- *Ejercicios II*:
  7. Obtener los números de los departamentos donde trabajan empleados cuyo salario sea inferior a 2500 euros.
  8. Obtener los valores diferentes de comisiones que hay en el departamento 110.
  9. Hallar las combinaciones diferentes de valores de salario y comisión en el departamento 111, por orden de salario y comisión crecientes.

IV. SQL como DML

➔ *Expresiones*

- Permite realizar operaciones:
  - En las condiciones de selección de las filas
  - En los valores de columna que aparecen en los resultados
- Devuelve un único valor como resultado
- Se formula como una combinación de operadores, operandos (sus valores han de ser homogéneos) y paréntesis
- *Ejemplos*

```
3+2
SALAR * 1,5
0,5 * COMIS
SALAR + COMIS
(SELECT COMIS FROM TEMPLE WHERE NUMEM=120)
(SELECT COMIS FROM TEMPLE WHERE NUMEM=120) + 50
(SELECT COMIS FROM TEMPLE WHERE NUMEM=100)
```

SELECT *Escalar*

- Si el resultado es una tabla vacía, a efectos de considerarla una expresión, es como si devolviera el valor *Nulo*.

IV. SQL como DML

➔ *Expresiones*

- En una sentencia **SELECT** se puede especificar una expresión de cualquier tipo donde sea válido especificar un valor, por ejemplo:
  - En la lista de **SELECT**, en lugar de los nombres de columna
  - En la cláusula **WHERE**, donde uno o ambos de los valores a comparar pueden ser el resultado de evaluar una expresión.
- *Ejemplo*: *Obtener los nombres y sueldos anuales expresados en euros de los empleados del departamento 100. Presentarlos por orden decreciente de sueldos.*

```
SELECT NOMEN, SALAR*12
FROM TEMPLE
WHERE NUMDE=100
ORDER BY 2 DESC
```

IV. SQL como DML

➔ *Expresiones*

- *Ejercicios III*:
  10. Obtener los nombres de los empleados cuya comisión es superior o igual al 50% de su salario, por orden alfabético.
  11. En una campaña de ayuda familiar se ha decidido dar a los empleados una paga extra de 30 euros por hijo a partir del cuarto inclusive. Obtener por orden alfabético para estos empleados: nombre y salario resultante que van a cobrar incluyendo esta paga extra.
  12. Obtener una relación por orden alfabético de los departamentos cuyo presupuesto es inferior a 30.000 euros. El nombre de los departamentos vendrá precedido de las palabras "*departamento de*".
  13. Llamemos presupuesto medio mensual de un departamento al resultado de dividir su presupuesto anual por 12. Supongamos que se decide aumentar los presupuestos medios mensuales de todos los departamentos en un 10% a partir del mes de octubre inclusive. Para los departamentos cuyo presupuesto medio de los meses anteriores a octubre es de más de 3.000 euros, hallar por orden alfabético el nombre del departamento y su presupuesto anual total después del incremento.

IV. SQL como DML

➔ *Expresiones*

- *Ejercicios III*:

14. Suponiendo que en los próximos 3 años el coste de vida va a aumentar un 6% anual y que se suben los salarios en la misma proporción, hallar para los empleados con más de 4 hijos su nombre y su sueldo anual actual y para cada uno de los próximos tres años, clasificándolos por orden alfabético.
15. Hallar por orden alfabético los nombres de los empleados tales que si se les da una gratificación de 60 euros por hijo, el total de esta gratificación no supera a la décima parte del salario.
16. Para los empleados del departamento 112, hallar el nombre y el salario total de cada uno (salario más comisión), por orden de salario total descendente, y por orden alfabético dentro del salario total.
17. Hallar por orden de número de empleado el nombre y salario total (salario más comisión) de los empleados cuyo salario total supera a 1.800 euros mensuales.
18. Obtener los números de los departamentos en los que hay algún empleado cuya comisión supere al 20% de su salario.