

ÉRICA PALOMINO

TEMA 4

LENGUAJES DE BASES DE DATOS. SQL

erica.palomino@escuelaartegranada.com

ESCUELAARTEGRANADA

EL LENGUAJE SQL

SQL: Structured Query Language (Lenguaje de consulta Estructurado)

Es un lenguaje **declarativo**: sólo hay que indicar qué se quiere hacer.

Incorpora instrucciones de DDL, DML y DCL

En SQL:

- Hablaremos de **tablas** en lugar de relaciones.
- Hablaremos de **columnas** en lugar de atributos.
- Hablaremos de **filas** en lugar de tuplas.

EL LENGUAJE SQL

Las operaciones de SQL reciben el nombre de sentencias.

Están formadas por diferentes partes que denominamos cláusulas.

Sentencia ◀ [SELECT codigo_producto, nombre_producto, tipo — Cláusula
FROM productos — Cláusula
WHERE precio > 1000; — Cláusula]

EL LENGUAJE SQL

CREATE : Para crear bases de datos, tablas, dominios, aserciones y vistas.

ALTER: Para modificar la estructura de tablas y cualquier otro elemento de la BD.

DROP: Para borrar bases de datos, tablas, dominios, aserciones y vistas.

EL LENGUAJE SQL

INSERT: permite insertar nuevas filas en las tablas.

DELETE: permite borrar filas completas de las tablas.

UPDATE: permite modificar algún dato que exista en una tabla.

SELECT: recuperar y mostrar por pantalla datos almacenados en una o varias tablas.



SENTENCIAS DDL

CREACIÓN DE TABLAS

CREATE TABLE: permite crear una tabla nueva en la BD.

La sintaxis genérica de CREATE TABLE es la siguiente:

```
CREATE TABLE nombre_tabla  
(  
    nombre_columna {tipo_datos|dominio} [def_defecto [restric_col]  
    [,nombre_columna {tipo_datos|dominio}[def_defecto][restric_col]]  
    [,nombre_columna {tipo_datos|dominio}[def_defecto][restric_col]]  
    ...  
);
```

CREATE TABLE. PASOS A SEGUIR

1. Decidir el nombre de la tabla (nombre_tabla).
2. Nombre de cada atributo (nombre_columna).
3. Asignar un tipo de datos a cada columna
 - a. También podremos dar definiciones por defecto y restricciones de columna.
4. Restricciones a nivel de tabla.

TIPOS DE DATOS

Los tipos de datos más utilizados son:

- varchar2 -> cadenas de texto
- number -> números
- date -> fechas

Tanto para las cadenas como para los números debemos indicar el tamaño máximo:

- varchar2(max)
- number(total, decimales)

Hay otros muchos tipos de datos

TIPOS DE DATOS

Tipos de datos predefinidos	
Tipos de datos	Descripción
CHARACTER (longitud)	Cadenas de caracteres de longitud fija.
CHARACTER VARYING (longitud)	Cadenas de caracteres de longitud variable.
BIT (longitud)	Cadenas de bits de longitud fija.
BIT VARYING (longitud)	Cadenas de bits de longitud variables.
NUMERIC (precisión, escala)	Número decimales con tantos dígitos como indique la precisión y tantos decimales como indique la escala.
DECIMAL (precisión, escala)	Número decimales con tantos dígitos como indique la precisión y tantos decimales como indique la escala.
INTEGER	Números enteros.
SMALLINT	Números enteros pequeños.
REAL	Números con coma flotante con precisión predefinida.
FLOAT (precisión)	Números con coma flotante con la precisión especificada.
DOUBLE PRECISION	Números con coma flotante con más precisión predefinida que la del tipo REAL.
DATE	Fechas. Están compuestas de: YEAR año, MONTH mes, DAY día.
TIME	Horas. Están compuestas de HOUR hora, MINUT minutos, SECOND segundos.
TIMESTAMP	Fechas y horas. Están compuestas de YEAR año, MONTH mes, DAY día, HOUR hora, MINUT minutos, SECOND segundos.

RESTRICCIONES DE COLUMNA

Una vez elegidos las columnas y asignados los tipos de datos, hay que decidir si las columnas tienen alguna restricción.

Las restricciones pueden ser del tipo:

- Debe ser **único** en toda la tabla o puede repetirse.
- Puede quedar **vacío** o debe estar siempre relleno.
- Si es o no clave **principal**.
- Si debe cumplir una determinada **condición**.
- Si tiene algún valor por **defecto**

RESTRICCIONES DE COLUMNA

Restricción	descripción
NOT NULL	En la columna no puede haber valores nulos
UNIQUE	En toda la columna no se pueden repetir valores
PRIMARY KEY	La columna es clave primaria: no puede tener valores nulos ni repetidos.
CHECK (condición)	Todos los valores de la columna deberán cumplir la condición
DEFAULT valor	Valor que tendrá por defecto la columna

RESTRICCIONES DE COLUMNA

Por último queda decidir que restricciones se van a incluir para toda la tabla en general.

Estas restricciones son del tipo:

- **Clave externa:** referencias a otras columnas de otra tabla.
- **Clave primaria múltiple:** cuando la clave principal está compuesta por varias columnas, se indicarán a nivel de tabla.

RESTRICCIONES DE COLUMNA

Sea la restricción que sea, siempre debe cumplir lo siguiente:

- Empieza por la palabra reservada **CONSTRAINT**.
- Tener un **nombre** único en toda la Base de Datos.
- **Tipo** de restricción.

Restricción	Descripción
FOREIGN KEY (columna) REFERENCES tabla(columna)	Clave externa
PRIMARY KEY (columnas)	Clave primaria múltiple

MANTENIMIENTO DE LA INTEGRIDAD

Cuando alguna de las restricciones es violada el sistema **genera un error**.

No se permitirá la realización de esos cambios y se nos **avisará del error** de algún modo

INTEGRIDAD: CLAVES EXTERNAS

El problema es algo más complejo.

El sistema no permite un valor en una clave externa que no exista en la tabla referenciada.

Si tratamos de eliminar o modificar una fila de la que existen referencias se rechazará la operación.

CREACIÓN DE TABLAS: EJERCICIOS

Estrella (nombre, dirección, sexo, f_nacimiento)

Estudio (nombre, dirección)

Película (título, año, duración, nombre_estudio)

Protagoniza (título_película, año_pel, nomb_estr)

CREACIÓN DE TABLAS: EJERCICIOS

Estrella (nombre, dirección, sexo, f_nacimiento)

- 1: Elegir nombre de la tabla
- 2: Elegir nombre de las columnas
- 3: Elegir tipos de datos de las columnas
- 4: Restricciones de columnas
- 5: Restricciones de tabla

CREACIÓN DE TABLAS: EJERCICIOS

Estudio (nombre, dirección)

- 1: Elegir nombre de la tabla
- 2: Elegir nombre de las columnas
- 3: Elegir tipos de datos de las columnas
- 4: Restricciones de columnas
- 5: Restricciones de tabla

CREACIÓN DE TABLAS: EJERCICIOS

Película (titulo, año, duración, nombre_estudio)

- 1: Elegir nombre de la tabla
- 2: Elegir nombre de las columnas
- 3: Elegir tipos de datos de las columnas
- 4: Restricciones de columnas
- 5: Restricciones de tabla

CREACIÓN DE TABLAS: EJERCICIOS

Protagoniza (título_película, año_pel, nomb_estr)

- 1: Elegir nombre de la tabla
- 2: Elegir nombre de las columnas
- 3: Elegir tipos de datos de las columnas
- 4: Restricciones de columnas
- 5: Restricciones de tabla

CREACIÓN DE TABLAS: EJERCICIOS

Persona (nombre, calle, ciudad)

Trabaja (nombre_persona, nombre_compañía, salario)

Compañía (nombre_compañía, ciudad)

Dirige (nombre_persona, nombre_director)

NOTA: El salario por defecto será de 1000€

CREACIÓN DE TABLAS: EJERCICIOS

Profesor (RFC, nombre, direcc, tlf, cod_depto, cod_centro)

Departamento (cod_depto, nombre)

Centro (cod_centro, nombre, teléfono, direcc)

Publico (cod_centro, cod_junta)

Privado (cod_centro, cuota)

Modulo (código, nombre, horas_semana)

Alumno (control, nombre, apellidos, control_del)

Matriculado (modulo, alumno)

Imparte (profesor, modulo, horas)

CREACIÓN DE TABLAS: EJERCICIOS

Departamento (cod_depto, nombre)

Familiar (DNI, nombre, parentesco)

Trabajador (DNI, nombre, dirección, cod_depto)

Proyecto (nombre_proyecto, presupuesto, cod_depto)

Trabaja (DNI, nombre_proyecto, n_horas)

- El código del departamento será numérico de 3 dígitos
- El parentesco solo puede ser «hijo» o «padre»
- El presupuesto por defecto será de 5000€
- Un trabajador hará un mínimo de 50 horas en un proyecto, pero podría no haber empezado y el campo estará vacío

CREACIÓN DE TABLAS: EJERCICIOS

Linea (nombre, inicio, fin)

Tren (id_tren, vagones, linea, cochera)

Cochera (id_cochera, metros, estacion)

Acceso (id_estacion, num_acceso, discapacitados)

Estacion (id_estacion, horario, direccion)

Recorridos (linea, estacion, orden)

- Un tren tendrá entre 1 y 3 vagones
- Los metros cuadrados de una cochera deben ser mayores de 100 y podrán tener 2 decimales
- El campo discapacitados almacenará «N» o «S»
- Horario será una cadena de caracteres con 5 caracteres máximo

CREACIÓN DE TABLAS: EJERCICIOS

Equipo (nombre, ciudad)

Jugador (DNI, nombre, dorsal, equipo)

Partido (id_partido, fecha, resultado)

Juega (DNI, partido, posición)

Rivales (id_partido, local, visitante)

- El dorsal de un jugador será un número entre 1 y 30
- El resultado de un partido puede estar vacío
- La posición de un jugador solo puede ser «D», «F», «P», «S»



CONJUNTO DE TABLAS PARA TRABAJAR

CONUNTO DE TABLAS

A partir de este momento trabajaremos con las siguientes tablas:

Clientes (dni, nombre, fecha_nac, direccion, sexo)

Productos (cod_prod, nombre, stock, precio, tipo)

Compras (cliente, producto, fecha, cantidad)

Trabajadores (codigo, nombre, categoria, area, tienda)

Tiendas (cod_tienda, metros)

Ofertas (cod_of, tienda, producto, trabajador, tipo, inicio, fin)

ELIMINACIÓN DE TABLAS

ELIMINACIÓN DE TABLAS

DROP TABLE : Borra una tabla junto con **todo** su contenido.

Su sintaxis es muy sencilla:

```
DROP TABLE <nombre_tabla> [CASCADE CONSTRAINTS];
```

Una tabla no se puede eliminar si existen otras tablas que la dependen de ella.

Forzamos el borrado con la opción CASCADE CONSTRAINTS.


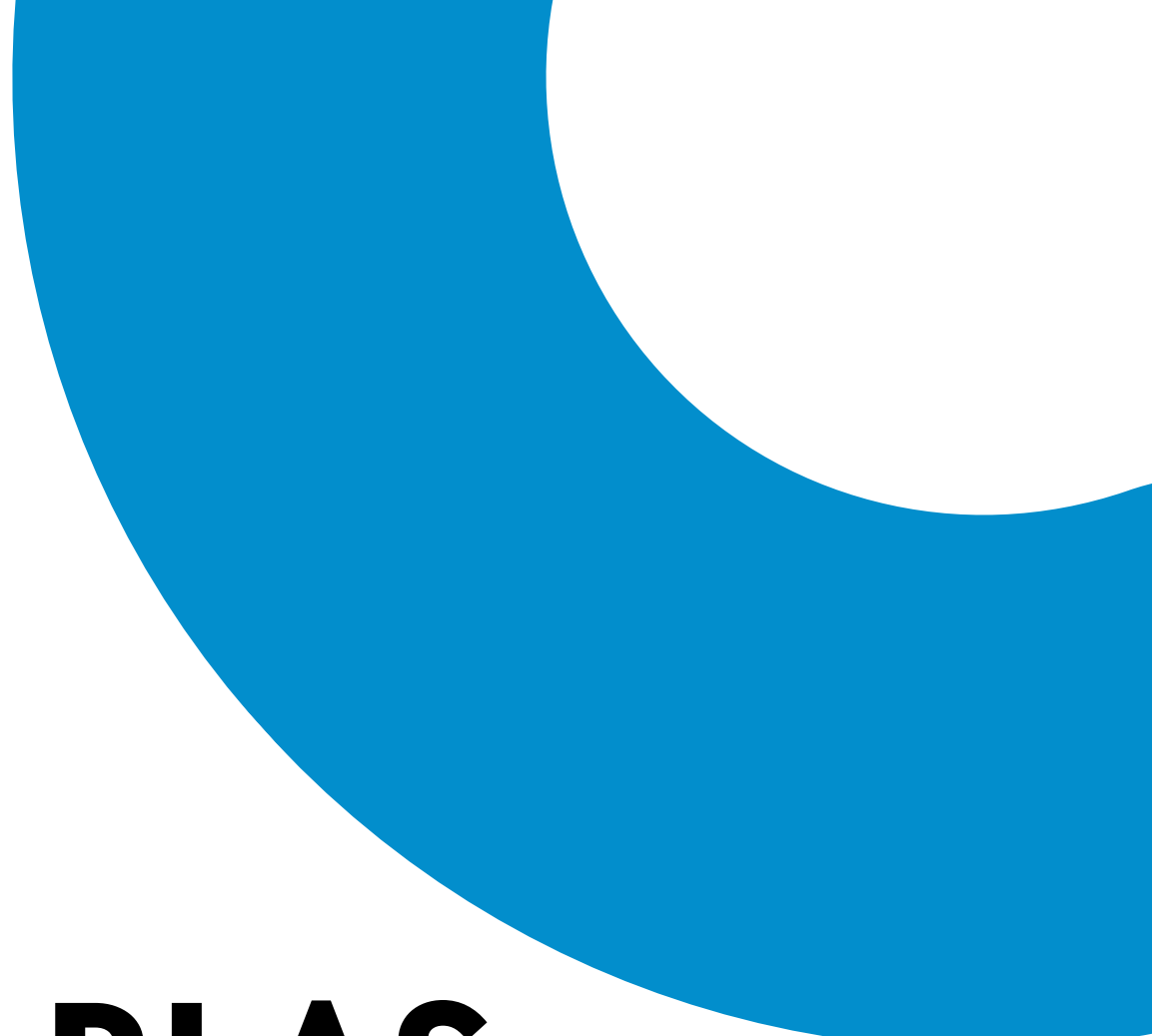
ELIMINACIÓN DE TABLAS

Ejemplo: Borrar la tabla Clientes de la base de datos.

DROP TABLE clientes;

Ejemplo: Borrar la tabla Clientes de la base de datos, aunque haya claves externas que apunten a ella:

DROP TABLE clientes CASCADE CONSTRAINTS;



MODIFICACIÓN DE TABLAS

MODIFICACIÓN DE TABLAS

ALTER TABLE : permite modificar la estructura de una tabla existente.

Los cambios que se pueden hacer son:

- **Añadir** columnas nuevas
- **Eliminar** una columna existente
- **Cambiar el nombre** a una columna existente
- Modificar las **restricciones** de una columna existente
- Cambiar el **nombre de la tabla** completa.

Su sintaxis es muy sencilla:

ALTER TABLE <nombre_tabla> <cambio a realizar>

MODIFICACIÓN DE TABLAS

Añadir nuevas columnas:

```
ALTER TABLE <nombre_tabla>
```

```
ADD <nombre_columna> <restricciones_columna>;
```

Se permite añadir varias columnas a la vez:

```
ALTER TABLE <nombre_tabla> ADD
```

```
(
```

```
<nombre_columna> <restricciones_columna>,
```

```
<nombre_columna> <restricciones_columna>,
```

```
<nombre_columna> <restricciones_columna>
```

```
);
```

MODIFICACIÓN DE TABLAS

Eliminar una columna existente:

```
ALTER TABLE <nombre_tabla>  
DROP COLUMN <nombre_columna>
```

Cambiar el nombre a una columna existente:

```
ALTER TABLE <nombre_tabla>  
RENAME COLUMN <nombre_viejo> TO <nombre_nuevo>
```

Cambiar el nombre de la tabla completa:

```
ALTER TABLE <nombre_tabla>  
RENAME TO <nuevo_nombre>
```

MODIFICACIÓN DE TABLAS

Modificar restricciones de una columna existente:

```
ALTER TABLE <nombre_tabla>  
MODIFY <nombre_column><nuevo_tipo><nuevas_restri>;
```

Se permite modificar varias columnas a la vez:

```
ALTER TABLE <nombre_tabla> MODIFY  
(  
  <nombre_column> <nuevo_tipo> <nuevas_restri>,  
  <nombre_column> <nuevo_tipo> <nuevas_restri>,  
  <nombre_column> <nuevo_tipo> <nuevas_restri>  
);
```

MODIFICACIÓN DE TABLAS: EJERCICIOS

Teniendo en cuenta el conjunto de tablas:

- Añadir a los clientes y a los trabajadores el teléfono.
- Quitar de los productos el stock.
- Asegurarnos que el teléfono de los clientes y de los trabajadores empieza por 6 como mínimo.
- El precio por defecto será de 0.
- Añadir a las tiendas un nombre y una localización
- Asegurarnos que el stock no sea menor de 0 y que el precio esté entre 0 y 10

INSERCIÓN DE FILAS

INSERCIÓN DE FILAS

Para insertar datos en una tabla, la sintaxis es sencilla:

```
INSERT INTO <nombre_tabla> VALUES (<valor1>, ..... <valorN>);
```

Ejemplo: Insertar un nuevo cliente en la base de datos con los siguientes datos.

- DNI = 44444444
- Nombre: Carmen Garrido
- Fecha de nacimiento: 28/09/75
- Dirección: Granada
- Sexo: Mujer

```
INSERT INTO clientes VALUES ('44444444', 'Carmen Garrido',  
'28/09/75', 'Granada', 'M');
```

INSERCIÓN DE FILAS - ACLARACIONES

Datos de tipo **cadena** y **fecha** van entrecomillados.

Los valores de tipo fecha suelen expresarse como cadenas, pero el formato específico depende de SGBD concreto.

Los valores han de suministrarse en el mismo orden en el que se crearon al definir la tabla con CREATE TABLE.

Si nos interesa dar un valor nulo a un campo concreto, podemos utilizar la palabra reservada *null*.

INSERCIÓN DE FILAS - ACLARACIONES

Se puede alterar el orden en el que se suministran los valores de las columnas.

Para ello hay que utilizar una variante de la sintaxis,

```
INSERT INTO <nombre_tabla> (<nombre_columna>,.....  
<nombre_columna>) VALUES (<valor1>, .....,<valorN>);
```

```
INSERT INTO clientes (DNI, fecha_nac, nombre, sexo, direccion)  
VALUES ('44444444', '28/09/75', 'Carmen Garrido', 'M', 'Granada');
```

INSERCIÓN DE FILAS - EJERCICIOS

Clientes:

DNI	nombre	Fecha_nac	Direccion	Sexo
11111111Z	Lucía	12/05/2002	Granada	M
22222222B	Mónica	18/12/2008	Jaén	M
12345678C	Luis	03/01/2005	Granada	H
33333333R	César	08/09/2003	Granada	H
55555555T	Roberto	24/11/2008	Málaga	H

INSERCIÓN DE FILAS - EJERCICIOS

Productos:

cod_prod	Nombre	Stock	Precio	Tipo
1	Lápiz negro	100	0.75	1
2	Bolígrafo azul	85		1
3	Libreta A4	60	1.75	2
4	Cuaderno rubio	50	3.25	2
5	Corrector	86		3
6	Goma borrar	150	0.35	3

INSERCIÓN DE FILAS - EJERCICIOS

Compras:

Cliente	Producto	Fecha	Cantidad
11111111Z	1	25/10/2023	2
12345678C	1	26/10/2023	1
11111111Z	2	25/10/2023	4
55555555T	2	26/10/2023	3
11111111Z	3	26/10/2023	1
12345678C	4	26/10/2023	1
33333333R	2	25/10/2023	6
11111111Z	1	30/10/2023	5
12345678C	3	02/11/2023	2
12345678C	4	30/10/2023	1
33333333R	1	25/10/2023	4
55555555T	2	30/10/2023	3
55555555T	3	30/10/2023	1
55555555T	3	02/11/2023	2
12345678C	4	02/11/2023	2

ELIMINACIÓN DE FILAS

ELIMINACIÓN DE FILAS

El borrado de filas es muy sencillo en SQL. La sentencia es:

```
DELETE FROM <nombre_tabla>;
```

Ejemplo: Borrar todas las filas de la tabla clientes.

```
DELETE FROM clientes;
```

ELIMINACIÓN DE FILAS

No siempre nos interesa borrar todas las filas de una tabla.

SQL incluye una segunda cláusula dentro de la sentencia

DELETE

WHERE permite incluir la condición que deben cumplir las filas que se borrarán:

```
DELETE FROM <nombre_tabla> WHERE condición;
```

Ejemplo: Borrar todas las filas de los clientes que sean mujeres.

```
DELETE FROM clientes WHERE sexo='M';
```

ELIMINACIÓN DE FILAS

Eliminar de la base de datos los siguientes elementos:

- Eliminar todas las compras
- Clientes de Málaga
- Todos los productos de menos de 1€.
- Todos los clientes hombre
- Todos los clientes
- Todos los productos

RESTAURAR TODOS LOS DATOS

- Eliminar las compras con cantidad inferior a 3

ACTUALIZACIÓN DE FILAS

ACTUALIZACIÓN DE FILAS

Se realiza mediante el uso de la sentencia **UPDATE**.

La sintaxis es la siguiente:

```
UPDATE <nombre_tabla>  
SET <nombre_columna>=<nuevo_valor>  
[{,<nombre_columna>=<nuevo_valor>}]  
[WHERE condicion];
```

ACTUALIZACIÓN DE FILAS

Ejemplo: Cambiar el nombre del cliente con DNI=44444444 por el de JUAN LOPEZ

```
UPDATE clientes SET nombre='Juan López'  
WHERE DNI = '44444444';
```

Ejemplo: Subir el precio en 0,25 € y el stock a 150 del bolígrafo azul.

```
UPDATE productos SET precio = precio + 0.25,  
stock=150 WHERE nombre = 'Bolígrafo azul';
```

ACTUALIZACIÓN DE FILAS - EJERCICIOS

Cambiar de la base de datos los siguientes datos:

- Mónica se muda a Málaga
- Todos los productos valdrán 0,25€ más
- A todas las compras del producto 2 hay que subirles en 1 la cantidad.
- Los productos de tipo 1 y 2 tendrán un precio de 1,25€ cada uno.



MANIPULACIÓN DE DATOS CON SQL

ACTUALIZACIÓN DE FILAS

Para recuperar los datos de las tablas de nuestra base de datos, SQL dispone de la sentencia **SELECT**. La sintaxis básica de esta sentencia tiene la siguiente forma:

```
SELECT <nombre_coluna> [{,nombre_columna}]  
FROM <nombre_tabla>  
[WHERE <condicion>];
```

MANIPULACIÓN DE DATOS

```
SELECT <nombre_coluna> [{,nombre_columna}]  
FROM <nombre_tabla>  
[WHERE <condicion>];
```

Hay tres cláusulas principales:

- **SELECT** contiene las columnas que queremos mostrar.
- **FROM** indica sobre que tabla queremos consultar.
- **WHERE** impone una condición booleana que deben cumplir las tuplas para ser recuperadas.

MANIPULACIÓN DE DATOS

Mostrar sólo los nombres de los clientes.

El proceso a seguir para conseguirlo:

Decidir qué tabla o tablas necesitamos:

- Clientes

Decidir qué columna o columnas queremos mostrar:

- nombre

Construir la sentencia SQL:

```
SELECT nombre  
FROM clientes;
```


MANIPULACIÓN DE DATOS

Mostrar la dirección de los clientes

Decidir qué tabla o tablas necesitamos:

- clientes

Decidir qué columna o columnas queremos mostrar:

- direccion

Construir la sentencia SQL:

```
SELECT direccion  
FROM clientes;
```

MANIPULACIÓN DE DATOS

Mostrar el DNI y el nombre de los clientes

MANIPULACIÓN DE DATOS

Mostrar el DNI y el nombre de los clientes

Decidir qué tabla o tablas necesitamos:

- clientes

Decidir qué columna o columnas queremos mostrar:

- DNI, nombre

Construir la sentencia SQL:

```
SELECT DNI, nombre  
FROM clientes;
```

MANIPULACIÓN DE DATOS

Mostrar todos los datos de los clientes

MANIPULACIÓN DE DATOS

Mostrar todos los datos de los clientes

Decidir qué tabla o tablas necesitamos:

- clientes

Decidir qué columna o columnas queremos mostrar:

- Todas

Construir la sentencia SQL:

```
SELECT DNI, nombre, fecha_nac, direccion, sexo  
FROM clientes;
```

```
SELECT * FROM clientes;
```

MANIPULACIÓN DE DATOS

Las condiciones de la cláusula WHERE pueden ser tan complicadas como el usuario necesite.

- Identificadores de columnas
- Literales
- Operadores de comparación (<, >, >=, <=, <>, =)
- Operadores lógicos (AND, OR, NOT)

MANIPULACIÓN DE DATOS

Mostrar el nombre de los clientes que son de Granada

MANIPULACIÓN DE DATOS

Mostrar el nombre de los clientes que son de Granada

Decidir qué tabla o tablas necesitamos:

- clientes

Decidir qué columna o columnas queremos mostrar:

- nombre

Decidir qué condición o condiciones deben cumplir las filas mostradas:

- Direccion = 'Granada'

Construir la sentencia SQL:

```
SELECT nombre
```

```
FROM clientes
```

```
WHERE direccion = 'Granada';
```


MANIPULACIÓN DE DATOS

Mostrar el nombre de los clientes que son mujeres

MANIPULACIÓN DE DATOS

Mostrar el nombre de los clientes que son mujeres

Decidir qué tabla o tablas necesitamos:

- clientes

Decidir qué columna o columnas queremos mostrar:

- nombre

Decidir qué condición o condiciones deben cumplir las filas mostradas:

- sexo = 'M'

Construir la sentencia SQL:

```
SELECT nombre
```

```
FROM clientes
```

```
WHERE sexo= 'M';
```

MANIPULACIÓN DE DATOS

Mostrar el nombre y DNI de los clientes que NO son de Granada

MANIPULACIÓN DE DATOS

Mostrar el nombre y DNI de los clientes que NO son de Granada

Decidir qué tabla o tablas necesitamos:

- clientes

Decidir qué columna o columnas queremos mostrar:

- nombre, DNI

Decidir qué condición o condiciones deben cumplir las filas mostradas:

- Direccion <> 'Granada'

Construir la sentencia SQL:

```
SELECT nombre, DNI
```

```
FROM clientes
```

```
WHERE direccion <> 'Granada';
```

MANIPULACIÓN DE DATOS

Mostrar todos los datos del cliente llamado Lucía

MANIPULACIÓN DE DATOS

Mostrar todos los datos del cliente llamado Lucía

Decidir qué tabla o tablas necesitamos:

- clientes

Decidir qué columna o columnas queremos mostrar:

- Todos

Decidir qué condición o condiciones deben cumplir las filas mostradas:

- nombre= 'Lucía'

Construir la sentencia SQL:

```
SELECT * FROM clientes WHERE nombre= 'Lucía';
```

MANIPULACIÓN DE DATOS

Realicemos las siguientes consultas:

Mostrar el nombre de todos los productos.

Mostrar el precio de todos los productos.

Mostrar código y precio de todos los productos.

Mostrar el precio de los productos de tipo 2.

Mostrar todos los datos del producto con código 4

Mostrar todos los datos de aquellos productos que valen más de 0,75€.

Mostrar los nombres de los productos que valen entre 0,75€ y 1,25€.

MANIPULACIÓN DE DATOS

Realicemos las siguientes consultas:

Mostrar el DNI de los clientes que han comprado algo.

Mostrar el DNI de los clientes que han comprado el producto 3.

Mostrar el DNI de los clientes que han comprado más de 3 unidades del producto con código 1.

Mostrar el DNI y el código de producto de los clientes que hayan de lo que sea más de 3 unidades.

MANIPULACIÓN DE DATOS - ANOTACIONES

SELECT pueden aparecer los atributos propios de la tabla que aparecen la cláusula FROM.

Se pueden construir expresiones que parezcan en las columnas proyectadas para que el sistema muestre el resultado de aplicar esas expresiones a cada una de las filas.

MANIPULACIÓN DE DATOS - ANOTACIONES

Ejemplo: Mostrar el precio con IVA de los productos. En estos productos el IVA es del 21%.

```
SELECT nombre, precio*1.21 FROM productos;
```

El resultado de esta consulta mostrará, para cada producto, su nombre y una columna adicional con el resultado de multiplicar su precio por 1,21.

MANIPULACIÓN DE DATOS - ANOTACIONES

Ejemplo: Mostrar el precio con IVA de los productos. En estos productos el IVA es del 21%.

```
SELECT nombre, precio*1.21 FROM productos;
```

El resultado de esta consulta mostrará, para cada producto, su nombre y una columna adicional con el resultado de multiplicar su precio por 1,21.

MANIPULACIÓN DE DATOS - ANOTACIONES

Ejemplo: Mostrar los productos cuyo precio es desconocido:

```
SELECT cod_prod, nombre  
FROM productos  
WHERE precio IS NULL;
```

NULL es un estado «**vacío**» no es un valor, por tanto se comprueba con **IS** o **IS NOT**, no con igual o distinto

MANIPULACIÓN DE DATOS - ANOTACIONES

Con SQL, si queremos eliminar las filas duplicadas del resultado debemos solicitarlo mediante **DISTINCT**.

```
SELECT DISTINCT direccion  
FROM clientes;
```

MANIPULACIÓN DE DATOS - ANOTACIONES

Al ejecutar una consulta, obtenemos los datos en el mismo orden en el que se encuentran en la tabla original.

Podemos indicar que queremos que los resultados se muestren en otro orden.

Utilizamos para ello la cláusula **ORDER BY**.

ORDER BY admite además indicar el tipo de ordenación.

- **ASC** • **DESC**

MANIPULACIÓN DE DATOS - ANOTACIONES

Ejemplo: Mostrar la lista de los hombres ordenados en orden alfabético.

```
SELECT * FROM clientes WHERE sexo='H'  
ORDER BY nombre ASC;
```

MANIPULACIÓN DE DATOS - ANOTACIONES

Ejemplo: Mostrar la lista de los clientes ordenados por su direccion de procedencia de forma descendente y dentro de cada direccion, ordenados alfabéticamente.

```
SELECT direccion, nombre FROM clientes  
ORDER BY direccion DESC, nombre ASC;
```




CONSULTAS SOBRE VARIAS TABLAS

CONSULTAS SOBRE VARIAS TABLAS

Hay veces que es necesario utilizar datos que están repartidos en distintas tablas.

En SQL, se pueden incluir varias tablas en la cláusula FROM.

En este caso el sistema hace el **producto cartesiano** de todas las tablas incluidas y luego realiza la consulta sobre la tabla resultado de ese producto cartesiano.

Esto quiere decir que tenemos que eliminar las filas **NO REALES**.

CONSULTAS SOBRE VARIAS TABLAS

Para explicar esto, vamos a utilizar dos nuevas tablas:

Dueños:

Perros:

DNI	Nom_dueño
111111S	Rocio
333333E	Paloma
666666B	Victor

Num_perro	Nom_perro	Dni_dueño
1	Ali	111111S
2	Buda	333333E
3	Pico	111111S
4	Rufo	666666B

CONSULTAS SOBRE VARIAS TABLAS

Supongamos que nos piden mostrar los perros de cada uno de los dueños.

La consulta:

```
Select * From dueños, perros;
```

CONSULTAS SOBRE VARIAS TABLAS

Supongamos que nos piden mostrar los perros de cada uno de los dueños.

La consulta:

```
Select * From dueños, perros;
```

Dentro de esta tabla, hay muchas filas que no son "REALES" ya que no unen de verdad a cada perro con su dueño

CONSULTAS SOBRE VARIAS TABLAS

¿Qué pasa con las filas que SI son reales? ¿Qué tienen en común todas ellas?

- DNI = DNI_dueño

Por tanto, una vez unidas las tablas, deberemos quedarnos solo con las filas que nos interesan:

```
Select * From dueños, perros  
Where dni = dni_dueño;
```

CONSULTAS SOBRE VARIAS TABLAS

Volvamos ahora a las tablas que teníamos:

Clientes (DNI, nombre, fecha_nac, direccion, sexo)

Productos (cod_prod, nombre, stock, precio, tipo)

Compras (cliente, producto, fecha, cantidad)

CONSULTAS SOBRE VARIAS TABLAS

Mostrar para cada compra, el nombre del producto al que corresponde.

Tablas que necesitamos:

- Compras, productos.

Columnas que queremos visualizar:

- Todas las de compras, nombre de productos

Condición que deben cumplir las filas visualizadas:

- producto = cod_prod

CONSULTAS SOBRE VARIAS TABLAS

Mostrar para cada compra, el nombre del producto al que corresponde.

Consulta:

```
Select compras.*, productos.nombre  
From compras, productos  
Where producto = cod_prod;
```

CONSULTAS SOBRE VARIAS TABLAS

Mostrar la tabla compras, incluyendo el nombre de cada cliente que ha comprado.

Mostrar el nombre de aquellos clientes que hayan comprado más de 4 unidades del producto con código 2.

Visualizar el nombre de los productos que han comprado menos de 3 unidades.

Mostrar todos los datos de los productos que ha comprado el cliente llamado "Mónica"

CONSULTAS SOBRE VARIAS TABLAS

Supongamos el contenido de la tabla TRABAJADORES:

Código	Nombre	Categoría	Área	Tienda
18	Pedro	Encargado	Cajas	T1
21	Elena	Encargado	Reposición	T1
35	Manuel	Suplente	Reposición	T3

•El contenido de TIENDAS podría ser:

Cod_tienda	metros
T1	500
T2	800
T3	250

CONSULTAS SOBRE VARIAS TABLAS

Y el contenido de OFERTAS podría ser algo como:

cod_of	Tienda	Product o	Trabajador	Tipo_of	Inicio	Fin
O1	T1	2	18	1	Septiembre	Octubre
O2	T2	6		1	Octubre	Noviembre
O3	T3	6	35	2	Julio	Agosto
O4	T3	3	35	3	Octubre	Diciembre

CONSULTAS SOBRE VARIAS TABLAS

Mostrar el nombre de los productos ofertados en la tienda T3.

Mostrar los códigos de los ofertas de Bolígrafo Azul.

Mostrar los códigos de los ofertas de Goma de borrar.

Mostrar los metros de las tiendas en las que se oferta Lápiz Negro.

Listar el nombre y la categoría de los trabajadores que trabajan en la tienda T1.

Visualizar todos los datos de las ofertas de la tienda en la que trabaja Elena

CONSULTAS SOBRE VARIAS TABLAS

Visualizar los nombres de los productos que se ofertan en septiembre.

Mostrar los nombres de los productos que se ofertan en tiendas más grandes de 500 metros.

Mostrar los nombres de los productos que han comprado los clientes de Granada.

Listar las provincias de las que vienen los clientes que han comprado más de 7 Bolígrafos azules.

CONSULTAS SOBRE VARIAS TABLAS

Mostrar los nombres y las compras de todos los clientes de Granada, ordenados alfabéticamente.

Mostrar las provincias de los clientes que han comprado 8 Bolígrafos Azules.

Mostrar las tiendas en las que se ofertan productos que aun no tienen asignado precio.

Mostrar los nombre y los precio de los productos que se han ofertado con ofertas de tipo 2.

CONSULTAS SOBRE VARIAS TABLAS

Listar los datos de los clientes que han comprado Bolígrafo Azul, ordenados por su fecha de nacimiento. Mostrar por pantalla los nombres de los clientes que han comprado productos que sí tienen el precio marcado.

Suponiendo que el IVA es de 21%, mostrar para cada cliente de Málaga, lo que ha tenido que pagar en sus compras

CONSULTAS SOBRE VARIAS TABLAS

Mostrar todas las ofertas que hay en las tiendas de más de 500 metros cuadrados. Se deberá mostrar el nombre de los productos y de los trabajadores.

Mostrar las provincias de las que vienen los clientes que han comprado productos con precios asignados.

Sacar los nombres y los precios de los productos que aparecen en ofertas que no se sabe qué trabajador ha creado.

Mostrar los nombres de los productos que están asociados a ofertas que empiezan en abril o en junio