PARTE I
LENGUAJE SQL. GESTION DE DATOS

Tema 1. EL LENGUAJE DE GESTIÓN DE BASES DE DATOS 3

1 – Bases de datos 3
1.1 - Lenguaje de gestión de bases de datos 3
1.2 - ¿Qué es una Base de Datos? 3
1.3 - ¿Qué es un Sistema de Gestión de Bases de Datos? 4

2 – Modelos de datos y bases de datos relacionales 6
2.1 – Tipos de bases de datos 6
2.2 - El Modelo de Datos Relacional. Componentes. 6
2.3 – Bases de datos relacionales 7

3 – Lenguaje SQL 12
3.1 - ¿Qué podemos hacer con SQL? 12
3.2 - Tipos de sentencias SQL. 12
3.3 - Sentencias SQL 13

TEMA 2. ELEMENTOS DEL LENGUAJE 15

1 - Introducción. 15

2 – Elementos de SQL 15
2.1 – Identificadores 15
2.2 – Palabras reservadas 15

3 - Datos 16
3.1 – Datos constantes o literales 16
3.2 – Datos variables 16

4 - Operadores 19
4.1 - Operadores aritméticos 19
4.2- Operadores de comparación 19
4.3 - Operadores lógicos 20
4.4 – Precedencia o prioridad en los operadores 23

5 - Funciones predefinidas 25
5.1 - Funciones numéricas o aritméticas 25
5.2 - Funciones de caracteres 26
5.3 - Funciones de fecha 27
5.4 - Funciones de comparación 29
5. 5 - Otras funciones 30

6 – Valores Nulos (NULL) 31

7 – Expresiones y condiciones 33

Tema 3. CREACIÓN DE TABLAS 34

1 - Consideraciones previas a la creación de una tabla 34
1.1 – Definición de la tabla 34
1.2 - Restricciones en las tablas 34

2 - Formato genérico para la creación de tablas. 36
2.1 - Formato básico para la creación de tablas 36

2.2 – Ejemplos básicos de creación de tablas....36
2.3 – Formatos para la creación de tablas con la definición de restricciones....37
2.4 - Integridad referencial....42
2.5 - Formato completo de la creación de tablas....43
3 - Modificación de la definición de una tabla. 45
3.1 - Formato general para la modificación de tablas....45
3.2 – Ejemplos de modificaciones de tablas....46
4 - Eliminación de una tabla. 48
4.1 - Formato para eliminar una tabla. 48
4.2 – Ejemplos de borrado de una tabla 48
5 – Renombrado de una tabla....49
5.1 - Formato para renombrar una tabla 49
5.2 – Ejemplos de renombrado de una tabla 49
Tema 4. ACTUALIZACION DE TABLAS 50
1 - Introducción 50
2 - Inserción de nuevas filas en la base de datos....50
2.1 – Formato de la inserción una fila 50
2.2 – Ejemplos de inserción de filas....51
3 - Modificación de filas. 56
3.1 – Formato de la modificación de filas 56
3.2 – Ejemplos de modificación de filas 56
4 - Eliminación de filas. 58
4.1 – Formato de la eliminación de filas 58
4.2 – Ejemplos de la eliminación de filas 58
5 - Restricciones de integridad y actualizaciones. 60
5.1 - Control de las restricciones de integridad referencial....60
5.2 - Ejemplos de borrados y modificaciones en cascada....61
6 - Control de transacciones: COMMIT y ROLLBACK. 64

TEMA 1. EL LENGUAJE DE GESTIÓN DE BASES DE DATOS

Paulina Barthelemy

1 – Bases de datos

1.1 - Lenguaje de gestión de bases de datos.

SQL son las siglas de Structured Query Language que significa lenguaje estructurado de consulta

Se trata de un lenguaje definido por el estándar ISO/ANSI_SQL que utilizan para la gestión de las bases de datos los principales fabricantes de Sistemas de Gestión de Bases de Datos Relacionales.

Es un lenguaje estándar no procedimental que se utiliza para definir, gestionar y manipular la información contenida en una Base de Datos Relacional.

En los lenguajes procedimentales se deben especificar todos los pasos que hay que dar para conseguir el resultado. Sin embargo, como ya hemos dicho, SQL es un lenguaje no procedimental en el que tan solo deberemos indicar al sistema qué es lo que queremos obtener, y el sistema decidirá cómo obtenerlo.

Es un lenguaje sencillo y potente que se emplea para la gestión de la base de datos a distintos niveles de utilización: usuarios, programadores y administradores de la base de datos.

1.2 - ¿Qué es una Base de Datos?

Una base de datos está constituida por un conjunto de información relevante para una empresa o entidad, junto con los procedimientos para almacenar, controlar, gestionar y administrar esa información.

Además, la información contenida en una base de datos cumple una serie de requisitos o características:

- Los datos están interrelacionados, sin redundancias innecesarias.
- Los datos son independientes de los programas que los usan.
- Se emplean métodos determinados para recuperar los datos almacenados o para incluir datos nuevos y borrar o modificar los existentes

Una base de datos estará organizada de forma que se cumplan los requisitos para que la información se almacene con las mínimas redundancias, con capacidad de acceso para diferentes usuarios pero con un control de seguridad y privacidad. Debe tener mecanismos que permitan recuperar la información en caso de pérdida y la capacidad de adaptarse fácilmente a nuevas necesidades de

almacenamiento.

1.3 - ¿Qué es un Sistema de Gestión de Bases de Datos?

Un Sistema de Gestión de Bases de Datos (SGBD) es una aplicación formada por un conjunto de programas que permite construir y gestionar bases de datos. Proporciona al usuario de la base de datos las herramientas necesarias para realizar, al menos, las siguientes tareas:

- Definir las estructuras de los datos.
- Manipular los datos. Es decir, insertar nuevos datos, así como modificar, borrar y consultar los datos existentes.
- Mantener la integridad de la información.
- Proporcionar control de la privacidad y seguridad de los datos en la Base de Datos, permitiendo sólo el acceso a los mismos a los usuarios autorizados.

Para realizar las funciones que acabamos de describir, el Sistema Gestor de Bases de Datos necesita un conjunto de programas que gestionen el almacenamiento y la recuperación de dichos datos y un personal informático que maneje dichos programas.

Los componentes principales de un SGBD son:

- **GESTOR DE LA BASE DE DATOS**
  Es un conjunto de programas transparentes al usuario que se encargan de gestionar la seguridad de los datos y el acceso a ellos. Interacciona con el sistema operativo proporcionando una interfaz entre el usuario y los datos. Cualquier operación que se realice ha de estar procesada por este gestor.

- **DICCIONARIO DE LA BASE DE DATOS**
  Es donde se almacena toda la descripción de los diferentes objetos de la base de datos. Esta información se almacena con la misma estructura que los datos de los usuarios. El almacenamiento de esta información lo realiza el sistema gestor y cualquier usuario puede acceder a su contenido con el mismo lenguaje que al resto de los datos almacenados (SQL)

- **LENGUAJES**
  El sistema gestor ha de proporcionar lenguajes que permitan definir la estructura de los datos, almacenar la información y recuperarla. Podrán utilizar estos lenguajes los usuarios y lo administradores de la base de datos.
  Estos lenguajes son:
    - **Lenguaje de definición de datos (DDL)**
      Para definir la estructura con la que almacenaremos los datos.
    - **Lenguaje de manipulación de datos (DML)**
      Para añadir, modificar o eliminar datos, así como recuperar la información almacenada.
    - **Lenguaje de control de datos (DCL)**
      Para controlar el acceso a la información y la seguridad de los datos. Permiten limitar y controlar los accesos a la información almacenada. De esta tarea se ocupa el administrador de la base de datos.

- **ADMINISTRADOR DE LA BASE DE DATOS**
  Es una persona o grupo de personas responsables de la seguridad y la eficiencia de todos los componentes del sistema de bases de datos. Deben conocer el sistema tanto a nivel físico como lógico y a todos los usuarios que interaccionan con él.

- **USUARIOS DE LA BASE DE DATOS**
  Tradicionalmente considerados como un componente más de los sistemas gestores de bases de datos, debido a que estos fueron los primeros en considerarlos una parte importante para el correcto funcionamiento del sistema. Pueden ser usuarios terminales (usuarios no especializados que interaccionan con la base de datos), usuarios técnicos (usuarios que desarrollan programas de aplicación para ser utilizados por otros) y usuarios especializados (usuarios que utilizan el sistema gestor de la base de datos como una herramienta de desarrollo dentro de otros sistemas más complejos).

Algunos de los productos comerciales más difundidos son:

- **ORACLE** de Oracle Corporation.
- **DB2** de I.B.M. Corporation
- **Informix** de Informix Software Inc.
- **SQL Server** de Microsoft Corporation.
- **MySQL** producto Open Source (código abierto)

2 – Modelos de datos y bases de datos relacionales

Un modelo de datos es una filosofía de trabajo que permite realizar una abstracción de la realidad y representarla en el mundo de los datos.

2.1 – Tipos de bases de datos

Existen, tradicionalmente, varios tipos de bases de datos:

Bases de Datos Jerárquicas
Bases de Datos en Red
Bases de Datos Relacionales
Bases de Datos Objeto-Relacionales

Estas dos últimas son, con diferencia, las más difundidas y utilizadas en la actualidad debido a su potencia, versatilidad y facilidad de utilización. Se basan en el Modelo Relacional cuyas principales características veremos a continuación. Para gestionarlas se utiliza el lenguaje SQL.

2.2 - El Modelo de Datos Relacional. Componentes.

Un modelo de datos es un conjunto de reglas y convenciones que nos permiten describir, con los elementos del modelo, los datos y las relaciones entre ellos.

Analizaremos el modelo de datos relacional, en él se basan las bases de datos relacionales.
Sus principales componentes son:

Entidad.
Es un objeto acerca del cual se recoge información relevante.
Ejemplo de entidades: EMPLEADO, CLIENTE, PRODUCTO.

Atributo
Es una propiedad o característica de la entidad.
Por ejemplo pueden ser atributos de la entidad PERSONA los siguientes: DNI, NOMBRE, EDAD, etc.

Relación o Inter-Relación
Representa la relación que puede haber entre dos entidades.

Por ejemplo, una relación **compra** representa la relación entre las entidades CLIENTE y PRODUCTO

CLIENTE -> **compra** -> PRODUCTO..... "Un cliente **compra** un producto"

Y también, una relación **pertenece a** representa la relación entre las entidades EMPLEADO Y DEPARTAMENTO

EMPLEADO -> **pertenece a** -> DEPARTAMENTO....."Un empleado **pertenece** a un departamento"

Con este modelo podemos representar, utilizando los componentes anteriores, la información que deseamos almacenar en la base de datos. Posteriormente este modelo se va a convertir en una base de datos relacional.

2.3 – Bases de datos relacionales

La definición de Bases de Datos Relacionales fue enunciada por *E.F. Codd.* En 1972 estableciendo las reglas que debía cumplir cualquier base de datos para ser una base de datos relacional. Son 12 reglas, llamadas reglas de Codd, más una teoría matemática, llamada cálculo y álgebra relacional, que marcan las características de los sistemas relacionales.

Es un modelo basado en la teoría de las relaciones, álgebra y calculo relacional, con las siguientes características:

- Hay una parte de definición de datos, llamada estática, que nos da la estructura del modelo, donde los datos se encuentran almacenados en forma de relaciones, llamadas generalmente tablas, ya que su estructura es muy similar a las tablas convencionales. Estas tablas son independientes de la forma física de almacenamiento. A estos datos se añaden unas restricciones que son unas reglas que limitan los valores que podemos almacenar en las tablas de la base de datos y nos permiten implementar las relaciones entre las tablas.
- A esta parte se añade otra, llamada dinámica, con las operaciones que se pueden realizar sobre las tablas, anteriormente definidas, para gestionar los datos almacenados

El modelo de datos relacional que acabamos de exponer se almacena en una base de datos relacional. Al realizar la conversión de un modelo relacional a una base de datos relacional las entidades y las relaciones del modelo se transforman en tablas y restricciones de la base de datos. Estas tablas junto con las restricciones forman la clave de las bases de datos relacionales.

2.3.1 - Tablas

Son los objetos de la Base de Datos donde se almacenan los datos. Tienen la forma de una tabla tradicional, de ahí su nombre. Normalmente una tabla representa una entidad aunque también puede representar una asociación de entidades. Cada fila representa una ocurrencia de la entidad y cada columna representa un atributo o característica de la entidad.
Las tablas tienen un nombre que las identifica y están formadas por atributos representados en las columnas, y por tuplas representadas en las filas.

Ejemplos de atributos: para la tabla `departamentos` las columnas con el número de departamento, el nombre del departamento y la localidad donde se encuentra.
La tabla `empleados` puede tener como columnas o atributos: numero de empleado, nombre, fecha de alta, salario,...

Ejemplos de tuplas son: los datos de un empleado si es una tabla de `empleados`, de un departamento si es una tabla de `departamentos`, de un cliente si se trata de una tabla de `clientes`, o de un producto si es una tabla de `productos`.

Ejemplos de tablas:
Tabla ***DEPARTAMENTOS:***

	Columna 1	*Columna 2*	*Columna 3*
	DEP_NO	**DNOMBRE**	**LOCALIDAD**
Fila 1 ->	10	CONTABILIDAD	BARCELONA
Fila 2 ->	20	INVESTIGACION	VALENCIA
Fila 3 ->	30	VENTAS	MADRID
Fila 4 ->	40	PRODUCCION	SEVILLA

Tabla de ***EMPLEADOS:***

EMP_NO	APELLIDO	OFICIO	DIRECTOR	FECHA_ALTA	SALARIO	COMISION	DEP_NO
7499	ALONSO	VENDEDOR	7698	20/02/81	1400.00	400.00	30
7521	LOPEZ	EMPLEADO	7782	08/05/81	1350.00	NUL0	10
7654	MARTIN	VENDEDOR	7698	28/09/81	1500.00	1600.00	30
7698	GARRIDO	DIRECTOR	7839	01/05/81	3850.00	NULO	30
7782	MARTINEZ	DIRECTOR	7839	09/06/81	2450.00	NUL0	10
7839	REY	PRESIDENTE	NULO	17/11/81	6000.00	NULO	10
7844	CALVO	VENDEDOR	7698	08/09/81	1800.00	0.00	30
7876	GIL	ANALISTA	7782	06/05/82	3350.00	NULO	20
7900	JIMENEZ	EMPLEADO	7782	24/03/82	1400.00	NULO	20

2.3.2 - Tablas del curso

A lo largo de este curso utilizaremos, además de las tablas `EMPLEADOS` y `DEPARTAMENTOS` cuya forma y contenido ya hemos visto, las tablas de `CLIENTES`, `PRODUCTOS` y `PEDIDOS` cuya forma y contenido es el siguiente:

TABLA DE CLIENTES

```
mysql> SELECT *
    -> FROM clientes;
+----------+-----------------------+-----------+-----------+-----+------+---------------+
|CLIENTE_NO| NOMBRE                | LOCALIDAD |VENDEDOR_NO|DEBE | HABER| LIMITE_CREDITO|
+----------+-----------------------+-----------+-----------+-----+------+---------------+
|      101 | DISTRIBUCIONES GOMEZ  | MADRID    |      7499 |0.00 |  0.00|       5000.00 |
|      102 | LOGITRONICA S.L       | BARCELONA |      7654 |0.00 |  0.00|       5000.00 |
|      103 | INDUSTRIAS LACTEAS S.A.| LAS ROZAS |      7844 |0.00 |  0.00|      10000.00 |
```

```
|      104 | TALLERES ESTESO S.A.    | SEVILLA   |      7654 |0.00 |  0.00|       5000.00 |
|      105 | EDICIONES SANZ          | BARCELONA |      7499 |0.00 |  0.00|       5000.00 |
|      106 | SIGNOLOGIC S.A.         | MADRID    |      7654 |0.00 |  0.00|       5000.00 |
|      107 | MARTIN Y ASOCIADOS S.L.| ARAVACA   |      7844 |0.00 |  0.00|      10000.00 |
|      108 | MANUFACTURAS ALI S.A.   | SEVILLA   |      7654 |0.00 |  0.00|       5000.00 |
+----------+-------------------------+-----------+-----------+-----+------+---------------
8 rows in set (0.00 sec)
```

TABLA PRODUCTOS

```
mysql> SELECT *
    -> FROM productos;
+-------------+-----------------------------+---------------+------------------+
| PRODUCTO_NO | DESCRIPCION                 | PRECIO_ACTUAL | STOCK_DISPONIBLE |
+-------------+-----------------------------+---------------+------------------+
|          10 | MESA DESPACHO MOD. GAVIOTA  |        550.00 |               50 |
|          20 | SILLA DIRECTOR MOD. BUFALO  |        670.00 |               25 |
|          30 | ARMARIO NOGAL DOS PUERTAS   |        460.00 |               20 |
|          40 | MESA MODELO UNIÓN           |        340.00 |               15 |
|          50 | ARCHIVADOR CEREZO           |       1050.00 |               20 |
|          60 | CAJA SEGURIDAD MOD B222     |        252.00 |               15 |
|          70 | DESTRUCTORA DE PAPEL A3     |        450.00 |               25 |
|          80 | MODULO ORDENADOR MOD. ERGOS |        495.00 |               25 |
+-------------+-----------------------------+---------------+------------------+
8 rows in set (0.03 sec)
```

TABLA PEDIDOS

```
mysql> SELECT *
    -> FROM pedidos;
+-----------+-------------+------------+----------+--------------+
| PEDIDO_NO | PRODUCTO_NO | CLIENTE_NO | UNIDADES | FECHA_PEDIDO |
+-----------+-------------+------------+----------+--------------+
|      1000 |          20 |        103 |        3 | 1999-10-06   |
|      1001 |          50 |        106 |        2 | 1999-10-06   |
|      1002 |          10 |        101 |        4 | 1999-10-07   |
|      1003 |          20 |        105 |        4 | 1999-10-16   |
|      1004 |          40 |        106 |        8 | 1999-10-20   |
|      1005 |          30 |        105 |        2 | 1999-10-20   |
|      1006 |          70 |        103 |        3 | 1999-11-03   |
|      1007 |          50 |        101 |        2 | 1999-11-06   |
|      1008 |          10 |        106 |        6 | 1999-11-16   |
|      1009 |          20 |        105 |        2 | 1999-11-26   |
|      1010 |          40 |        102 |        3 | 1999-12-08   |
|      1011 |          30 |        106 |        2 | 1999-12-15   |
|      1012 |          10 |        105 |        3 | 1999-12-06   |
|      1013 |          30 |        106 |        2 | 1999-12-06   |
|      1014 |          20 |        101 |        4 | 2000-01-07   |
|      1015 |          70 |        105 |        4 | 2000-01-16   |
+-----------+-------------+------------+----------+--------------+
16 rows in set (0.02 sec)
```

2.3.3 – Restricciones

Son una parte importante para la implementación del modelo relacional. Restringen los valores que pueden tomar los datos en cada una de las columnas.

Estas restricciones son:

- **Clave primaria (PRIMARY KEY)**
  Una de las características del modelo relacional es que cada fila debe ser única. Ello obliga a la existencia de un identificativo que permita y controle esta unicidad. Este identificativo es la clave primaria. Estará formada por una columna o grupo de columnas y se elegirá de tal forma que su valor en cada fila sea único.
  En una base de datos relacional es obligatoria su existencia en cada una de las tablas.

- **Clave Ajena (FOREIGN KEY)**
  Será la forma de implementar las relaciones entre las tablas. Una columna o grupo de columnas que sea clave ajena referenciará a la tabla con la que está relacionada. Los valores que podrá tomar la clave ajena serán valores que ya existan en la tabla relacionada, o en su defecto un valor nulo.
  La importancia de su definición radica en que será la encargada de implementar las relaciones entre las tablas, para ajustarnos al Modelo Relacional
  Por ejemplo la tabla EMPLEADOS está relacionada con la tabla DEPARTAMENTOS a través de la columna DEP_NO (numero de departamento) que se encuentra en ambas tablas. Esta columna es clave primaria de la tabla DEPARTAMENTOS y en la tabla EMPLEADOS es la clave ajena. Esto quiere decir que los valores que tome el campo DEP_NO en la tabla EMPLEADOS solo pueden ser valores que ya existan en el campo DEP_NO de la tabla DEPARTAMENTOS.

- **Unicidad (UNIQUE)**
  Es una restricción que obliga a que una columna o conjunto de columnas tenga un valor único o un valor nulo. También admite valores nulos.

- **Restricción de valores permitidos (CHECK)**
  Es una restricción que nos permitirá controlar el conjunto de valores que serán válidos en una columna. Solo serán validos los valores que cumplan la condición especificada.
  Nota: MySQL lo acepta dentro del formato pero no lo implementa en la versión actual.

- **Obligación de valor (NOT NULL)**
  Una de las características de las bases de datos relacionales es que se permite la ausencia de valor en los datos. Esta ausencia puede ser debida a que en el momento de introducir los datos se desconoce el valor de un atributo para esa fila o a que ese atributo es inaplicable para esa fila. El valor que se le asigna a ese atributo en esa fila se llama valor nulo (`NULL`). Esta restricción obliga a que una columna tenga que tener siempre valor no permitiéndose que tome el valor nulo.
  En el siguiente capítulo hablaremos de forma más extensa de estos valores nulos.

2.3.4 - Restricciones en las tablas del curso

Para implementar el modelo relacional con las restricciones propias (cada fila debe ser única) y las restricciones de nuestro modelo necesitamos, al menos, las siguientes restricciones en las tablas:

TABLA DEPARTAMENTOS

dep_no	**CLAVE PRIMARIA**

TABLA EMPLEADOS

emp_no	**CLAVE PRIMARIA**
dep_no	**CLAVE AJENA QUE REFERENCIA A dep_no DE DEPARTAMENTOS**
dir	**CLAVE AJENA QUE REFERENCIA A emp_no DE EMPLEADOS**

La tabla `EMPLEADOS` está relacionada con la tabla `DEPARTAMENTOS` a través de la columna DEP_NO (numero de departamento) que se encuentra en ambas tablas. De esta forma podemos saber, por ejemplo que el empleado GIL pertenece al departamento 20. Y si vamos a la tabla departamentos comprobaremos que el departamento 20 es INVESTIGACION y se encuentra en VALENCIA. Por tanto, el empleado GIL pertenece al departamento de INVESTIGACION que está en VALENCIA.
La tabla `EMPLEADOS` también se relaciona consigo misma mediante las columnas EMP_NO y DIRECTOR. Cada empleado tiene un número de empleado (EMP_NO) y suele tener también un DIRECTOR. Esta última columna contiene un número de empleado que, suponemos, es el director del empleado en cuestión. Así podemos saber que REY es el director de GARRIDO y de MARTINEZ; y que el director de JIMENEZ es MARTINEZ, etcétera. El director de un empleado debe ser a su vez empleado de la empresa, de ahí la existencia de esta clave ajena. La columna DIRECTOR deberá contener un valor que se corresponda con un valor de EMP_NO o tener el valor nulo.

TABLA CLIENTES

cliente_no	**CLAVE PRIMARIA**
vendedor_no	**CLAVE AJENA QUE REFERENCIA emp_no DE EMPLEADOS**

La tabla `CLIENTES` se relaciona con `EMPLEADOS` por medio de la columna VENDEDOR_NO de la primera que hace referencia a la columna EMPLEADO_NO de la segunda. Así cada cliente tendrá asignado un vendedor, que será un empleado de la empresa existente en la tabla `EMPLEADOS`.

TABLA PRODUCTOS

producto_no	**CLAVE PRIMARIA**

TABLA PEDIDOS

pedido_no	**CLAVE PRIMARIA**
producto_no	**CLAVE AJENA QUE REFERENCIA A producto_no DE PRODUCTOS**
cliente_no	**CLAVE AJENA QUE REFERENCIA A cliemte_no DE CLIENTES**

La tabla `PEDIDOS` se relaciona con `PRODUCTOS` mediante la columna PRODUCTO_NO y con CLIENTES mediante la columna CLIENTE_NO. De esta forma sabemos que el pedido número 1000 lo ha realizado el cliente INDUSTRIAS LACTEAS S.A. y que el producto solicitado es SILLA DIRECTOR MOD. BUFALO a un precio de 670.00, etcétera.

El SGBD velará porque todas las operaciones que se realicen respeten estas restricciones manteniendo así la integridad de la información (`integridad referencial`)

El resto de las restricciones las estudiaremos en el tema 3 de creación de tablas.

3 – Lenguaje SQL

Como ya hemos dicho al inicio del tema, SQL significa lenguaje estructurado de consulta *(Structured Query Language).*
Es un lenguaje desarrollado sobre un prototipo de gestor de bases de datos relacionales con su primera implementación en el año 1979.
Posteriormente, en 1986, fue adoptado como estándar por el instituto ANSI (*American National Standard Institute*) como estándar y en 1987 lo adopta ISO (*Internacional Standardization Organization).* Aparece así el ISO/ANSI SQL que utilizan los principales fabricantes de Sistemas de Gestión de Bases de Datos Relacionales.
El lenguaje SQL es un lenguaje relacional que opera sobre relaciones (tablas) y da como resultado otra relación.

3.1 - ¿Qué podemos hacer con SQL?

Todos los principales SGBDR incorporan un motor SQL en el Servidor de Base Datos, así como herramientas de cliente que permiten enviar comandos SQL para que sean procesadas por el motor del servidor. De esta forma, todas las tareas de gestión de la Base de Datos (BD) pueden realizarse utilizando sentencias SQL.
Lo que podemos hacer con este lenguaje SQL es:

- Consultar datos de la Base de Datos.
- Insertar, modificar y borrar datos.
- Crear, modificar y borrar objetos de la Base de Datos.
- Controlar el acceso a la información.
- Garantizar la consistencia de los datos.

Actualmente se ha impuesto el almacenamiento de la información en bases de datos. El SQL es, por tanto, un lenguaje muy extendido y muchos lenguajes de programación incorporan sentencias SQL como parte de su repertorio o permiten la comunicación con los motores de SQL.

3.2 - Tipos de sentencias SQL.

Entre los trabajos que se pueden realizar en una base de datos podemos distinguir tres tipos: definición, manipulación y control de datos. Por ello se distinguen tres tipos de sentencias SQL:

- Sentencias de definición de datos. (Lenguaje de Definición de Datos **DDL**)
  Se utilizan para:
  Crear objetos de base de datos ---------- **SENTENCIA CREATE**
  Eliminar objetos de base de datos ---------- **SENTENCIA DROP**
  Modificar objetos de base de datos ---------- **SENTENCIA ALTER**

- Sentencias de manipulación de datos. (Lenguaje de Manipulación de Datos **DML**)
  Se utilizan para:
  Recuperar información ---------- **SENTENCIA SELECT**
  Actualizar la información:
  Añadir filas ---------- **SENTENCIA INSERT**
  Eliminar filas ---------- **SENTENCIA DELETE**
  Modificar filas ---------- **SENTENCIA UPDATE**

- Sentencias de control de datos. (Lenguaje de Control de datos DCL)

Se utilizan para:
Crear privilegios de acceso a los datos ---------- **SENTENCIA GRANT**
Quitar privilegios de acceso a los datos ---------- **SENTENCIA REVOKE**

3.3 - Sentencias SQL

Realizaremos algunas consideraciones sobre las notaciones y formatos utilizados.

a) Formatos de las instrucciones
Estos formatos están recuadrados. Se escriben utilizando una notación que recordamos a continuación:

- Las palabras reservadas de SQL aparecen en mayúsculas.

- Los nombres de objetos (tablas, columnas, etcétera) aparecen en el formato TipoTítulo (las iniciales de las palabras en mayúsculas)

- Las llaves **{ }** indican la elección obligatoria entre varios elementos.

- La barra vertical **|** separa los elementos en una elección.

- Los corchetes **[ ]** encierran un elemento opcional.

- El punto y coma **;** que aparece al final de cada comando es el separador de instrucciones y en realidad no forma parte de la sintaxis del lenguaje SQL, pero suele ser un elemento requerido por las herramientas de cliente para determinar el final del comando SQL y enviar la orden (sin él ;) al servidor.

Los formatos de las instrucciones, cuando sean complejos, se irán viendo por partes. Cada vez que añadamos algo nuevo lo remarcaremos en negrita.

b) Consulta de los datos.
Realizar una consulta en SQL consiste en recuperar u obtener aquellos datos que, almacenados en filas y columnas de una o varias tablas de una base de datos, cumplen unas determinadas especificaciones. Para realizar cualquier consulta se utiliza la sentencia SELECT.

Aunque la sentencia de consulta de datos en las tablas, SELECT, se tratará con profundidad en los temas del 5 al 9, necesitaremos hacer alguna pequeña consulta para poder verificar los datos que tenemos en las tablas. Las primeras consultas van a ser escritas con un formato inicial de la sentencia SELECT, que se completará en el bloque siguiente.

SELECT { * | [*NombreColumna* [, *NombreColumna*]....] }
FROM *NombreTabla*
[WHERE *Condicion] ;

*Notación: hay que escoger obligatoriamente una de las opciones, entre indicar los nombres de las columnas y el asterisco ***** (por eso aparecen las posibles opciones entre llaves y separadas por una barra). En caso de escoger la segunda opción se pueden indicar una o varias columnas (por eso aparece entre corchetes y seguido de puntos suspensivos). La cláusula WHERE es opcional (por eso aparece entre corchetes).*

El funcionamiento de esta sentencia es el siguiente: visualizará las correspondientes filas de la tabla. Si hemos escrito los nombres de las columnas separados por comas visualizará solo los valores de esas columnas y si hemos escrito el signo * visualizará todas las columnas. Si además, hemos escrito una condición con la cláusula `WHERE` visualizará solo las filas que cumplan esa condición.

c) Notación de los ejemplos

Otras directrices importantes de notación. Vamos a escribir cada cláusula de la instrucción en una línea, como los espacios en blanco y saltos de línea dentro de la instrucción son ignorados, hasta que encuentra el ; que es el fin de la instrucción. Ello facilita enormemente la lectura de las instrucciones de selección.

Vamos a ver un ejemplo aunque todavía no conozcamos el significado.

Supongamos que escribimos:

select apellido,(salario+ifnull(comisio,0)) “salario total”,
dept_no from empleados where oficio = ’analista’ order by dept_no;

O bien que escribimos:

```
SELECT apellido, salario+IFNULL(comision,0) “Salario total”, dept_no
FROM empleados
WHERE oficio = ’ANALISTA’
ORDER BY dept_no;
```

Este segundo formato es más claro y visual. Cada cláusula, comenzando con una palabra reservada, aparece en una línea y se han introducido algunos espacios en blancos y saltos de línea para separar.
Utilizaremos este formato en todos los ejemplos y ejercicios.