Tema 3. SQL (DDL)

Para MySQL

Índice

Introducción

En un sistema de gestión de bases de datos las operaciones sobre los datos se realizan a través de consultas formuladas con un lenguaje específico declarativo. SQL se considera el lenguaje estándar de base de datos relacional.

SQL (Structured Query Language) es un Lenguaje de Consulta Estructurado. Pero es algo más que un lenguaje de consulta, puesto que no solo ofrece funciones de recuperación, sino también se permite operaciones de actualización y manipulación de información y estructura de los datos.

- Fue desarrollado por IBM a mediados de los 70.
- En 1976 Oracle Corporation introduce la primera implementación comercial disponible de este lenguaje la cual continua vigente.
- SQL se considera estándar de la industria como lenguaje de acceso a base de datos.

Características de SQL

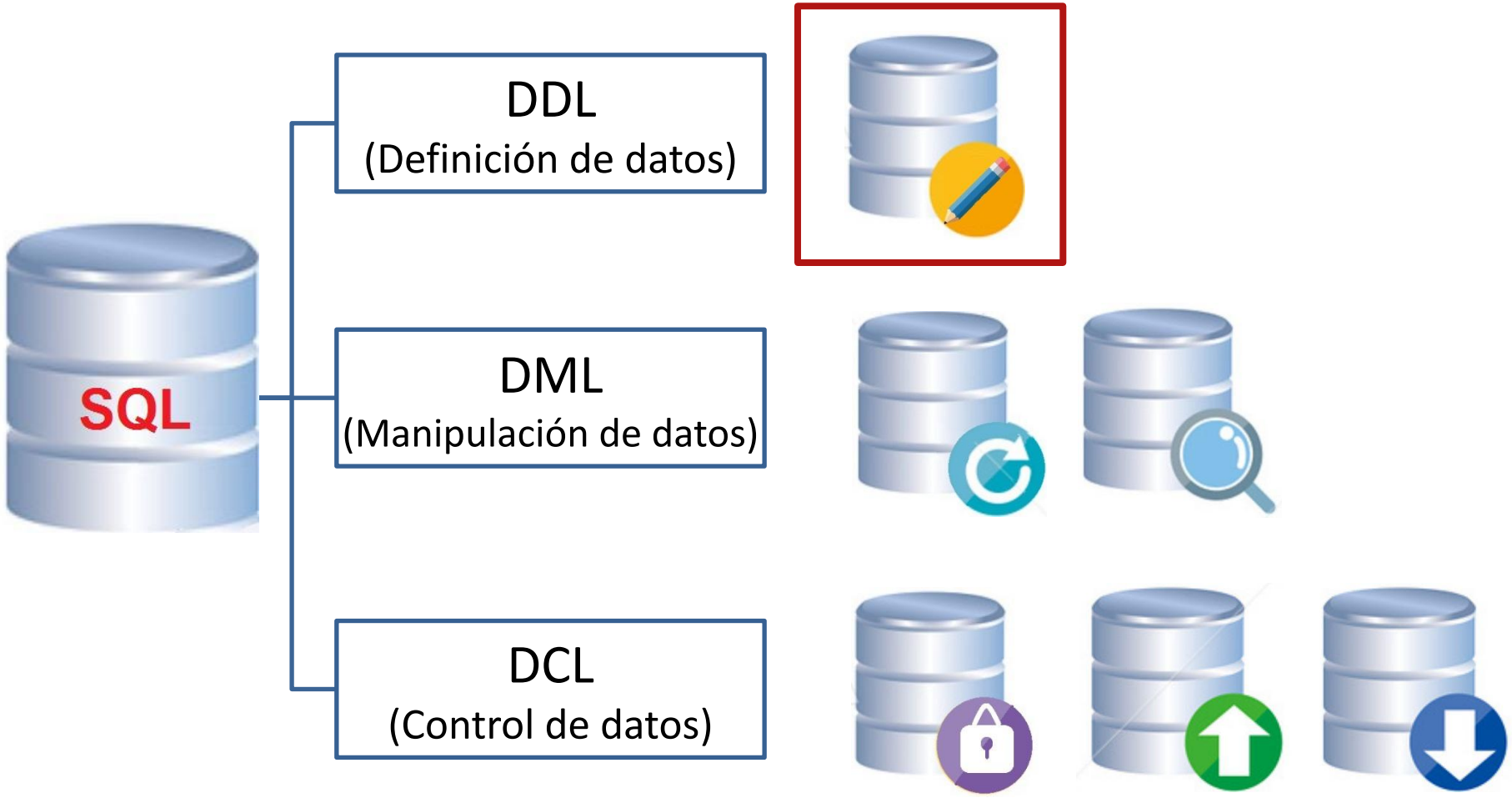
- Lenguaje **unificado** para todo tipo de tareas y usuarios , incluyendo:
  - Consulta, actualización, inserción, borrado de filas en una tabla.
  - Creación, modificación, reemplazamiento y borrado de objetos.
  - Permite garantizar la integridad y consistencia de los datos.
- Es un lenguaje **no procedimental o declarativo**
  - Cuando se realice una consulta, se describe cuál es el conjunto de datos que se quiere obtener, sin tener que especificar cuál es la estrategia de recuperación de esos datos. Especificamos QUÉ queremos, sin decir CÓMO conseguirlo.
  - Procesa varias filas a la vez
  - La navegación para buscar los datos es automática, utilizando unas rutinas de accesos llamadas “Optimizador”

Características de SQL

- Es **relacionalmente completo**, es decir, permite la realización de cualquier consulta de datos.
- Lenguaje **sencillo**, similar al inglés y sus comandos son fáciles de aprender.
- **Carácter estándar**. Existe una especificación estándar de este lenguaje, ANSI SQL. Sin embargo, cada fabricante refleja las peculiaridades propias de su SGBD modificando su SQL.

Sublenguajes de SQL

Dependiendo del tipo de operación SQL se divide en:



Sublenguajes de SQL
DDL (Data Definition Language)

Proporciona órdenes para definir, modificar o eliminar los distintos objetos de la base de datos (tablas, vistas, índices...).

CREATE	→	Permite al usuario definir un objeto. Por ejemplo una tabla.
ALTER	→	Permite al usuario modificar las características de un objeto ya existente. Por ejemplo añadir una columna a una tabla.
REPLACE	→	Permite reemplazar el objeto por uno nuevo si existiese previamente (*)
DROP	→	Permite al usuario borrar un objeto. Por ejemplo una tabla completa, no solo las filas.
RENAME	→	Permite renombrar un objeto. Por ejemplo una tabla, vista, procedimientos o sinónimo.
TRUNCATE	→	Borra las filas de una tabla o índice sin borrar su estructura. La tabla permanece pero vacía.

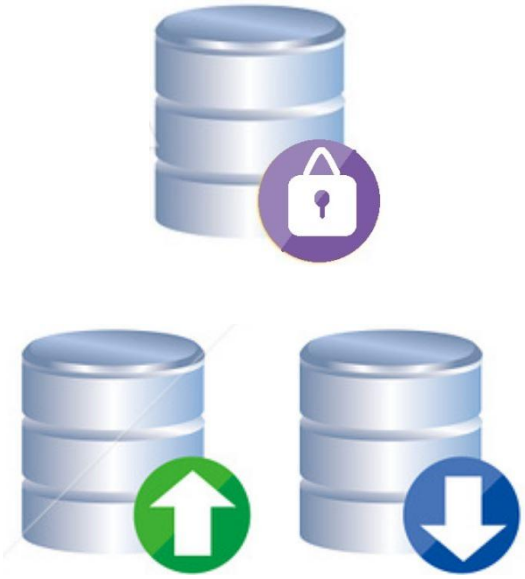
Sublenguajes de SQL
DML (Data Manipulation Language)

Agrupa los comandos SQL que permiten realizar consultas y modificaciones, inserciones y borrados.

SELECT→ Consultar datos (sin modificarlos)
UPDATE → Actualizar datos de tablas
INSERT→ Introducir nuevos datos en las tablas
DELETE → Borrar datos de las tablas

Sublenguajes de SQL
DCL (Data Control Language)

Permite establecer derechos de acceso a los usuarios, comprobaciones de integridad y control de transacciones. Incluye ordenes para dar y quitar privilegios.



GRANT → Concede privilegio
REVOKE → Suprime Privilegios
COMMIT
ROLLBACK
SAVEPOINT

Reglas sintácticas en SQL

- En una sentencia SQL se pueden incluir: tabuladores, retornos de carro, espacios y comentarios
- Son indiferentes las mayúsculas o minúsculas en palabras reservadas, identificadores y parámetros. Pero no en los literales de texto (constantes)
- SQL emplea los siguientes términos:
  - **TABLE**: tabla en lugar de relación
  - **ROW**: fila en lugar de tupla
  - **COLUMN**: columna en lugar de atributo

Reglas sintácticas en SQL

Reglas para nombrar objetos

- El nombre del objeto debe tener una longitud entre 1 (mínimo) y 30 caracteres (máximo).
- El nombre de un objeto no puede estar entre paréntesis, ni entrecomillado.
- El nombre de un objeto solo contiene caracteres alfanuméricos y se permiten también el carácter “_”

Reglas sintácticas en SQL

Reglas para referenciar objetos

- Sintaxis general
  [esquema.]nombre_objeto[.parte_objeto]
- Ejemplo: El usuario CARMEN ejecuta:

SELECT * FROM emp;

Suponemos que la tabla emp es propiedad del usuario que quiere acceder a los datos, de lo contrario se produce un error. Si la tabla es propiedad de un usuario distinto al que la consulta, por ejemplo LUIS es el propietario; CARMEN puede consultarla, siempre y cuando LUIS le haya concedido permiso.

- El usuario CARMEN ejecuta:

SELECT * FROM LUIS.emp;

Reglas sintácticas en SQL

Comentarios

Para introducir comentarios en varias líneas se introduce:

/* TEXTO COMENTADO */

Para introducir comentarios en una sola línea se usa:

-- A partir de aquí es comentario

Ejemplo:

```
/* Comentario
Largo */
-- Esto también es un comentario
```

Tipos de datos y conversiones entre tipos

Principales tipos

CHAR(n): Cadena de caracteres de longitud fija.
Se puede especificar el número de caracteres que tendrá (n).
VARCHAR(n): Cadena de caracteres de longitud variable.
Se debe especificar el número de caracteres que tendrá (n).
INT(n): Dato de tipo numérico entre -2147483648 a 2147483647.
Números enteros.
Se puede especificar el número de dígitos (n).
DECIMAL(size,d): Números reales. Donde “size” especifica el número total de dígitos y “d” el número total de decimales.
Por ejemplo DECIMAL(4,2) tiene como máximo valor 99.99.
DATE: El tipo DATE permite almacenar fechas.

Tipos de datos y conversiones entre tipos

Cadenas de caracteres: CHAR(n) y VARCHAR(n)

Las cadenas de caracteres se delimitan utilizando comillas simples.

Por ejemplo: ‘Hola’, ‘Una cadena’.

Conviene poner suficiente espacio para almacenar los valores. En el caso de los VARCHAR, Oracle no malgasta espacio por poner más espacio del deseado ya que si el texto es más pequeño que el tamaño indicado, el resto del espacio se ocupa.

Además de los operadores de igualdad (=, !=, ...) otras funciones útiles para trabajar con cadenas son:

CONCAT(cad1, cad2): concatena dos cadenas.

LENGTH(cad): devuelve la longitud de la cadena.

LOWER(cad): convierte todas las letras de la cadena a minúsculas.

UPPER(cad): ídem a mayúsculas.

Tipos de datos y conversiones entre tipos

Números

INT(n): Representa números enteros.

N especifica el número de cifras, su valor máximo es 255.

Su rango es entre -2147483648 a 2147483647. Fuera de estos rangos devuelve un error.

DECIMAL(size,d): Representa números reales.

size es la precisión máxima y **d** es el número de decimales.

Como máximo size vale 65 y d 30.

Ejemplo: DECIMAL (8,3) son números de 8 cifras con 3 decimales. 12345.678

Los decimales se presenta con el punto y no con la coma.

Para números mayores se utiliza float o double

Tipos de datos y conversiones entre tipos

Números: NUMBER y operaciones

Además de las operaciones típicas con valores numéricos (+, -, *, /), otras funciones útiles son:

- **ABS(num)**: devuelve el valor absoluto.
- **SQRT(num)**: devuelve la raíz cuadrada.
- **POWER(b,e)**: devuelve la potencia de b elevado a e.
- Existen otras funciones para grupos de valores (suma, media, máximo, ...) que se verán en apartados posteriores.

Tipos de datos y conversiones entre tipos

Fechas

Las fechas se escriben entre comillas simples y el separador puede ser casi cualquier símbolo. Rango desde 01/01/4712 AC hasta 31/12/9999 DC.

Los tipos de datos mas usados son:

- DATE - format YYYY-MM-DD. Ej: ’2022-01-01
- DATETIME - format: YYYY-MM-DD HH:MI:SS
- TIMESTAMP - format: YYYY-MM-DD HH:MI:SS
- YEAR - format YYYY or YY

El formato se puede cambiar con:

- DATE_FORMAT(fecha, "%d-%m-%Y") -> ’01-01-2022’

OJO: Asegurate que el formato que insertas es el que usa la base de datos.

Tipos de datos y conversiones entre tipos

Fechas

Las fechas se pueden comparar con los operadores típicos de comparación (<, >, !=, =, ...).

- **NOW() devuelve la fecha actual** (fecha y hora).
- **CURDATE() devuelve la fecha actual** (fecha).

Con las fechas es posible realizar operaciones aritméticas como sumas y restas de fechas, teniendo en cuenta que a una fecha se le suman días y que la diferencia entre dos fechas se devuelve también en días.

Por ejemplo: NOW() + 1 devuelve la fecha de mañana.

Tipos de datos y conversiones entre tipos

Datos de gran tamaño

Son tipos pensados para almacenar datos de tamaño muy grande. No pueden poseer índices ni ser parte de claves.

- **BLOB** (Binary Large OBject)

  Utilizado para guardar datos binarios de hasta 128 TB de tamaño. Se utilizan para almacenar datos binarios, típicamente imágenes, vídeos, documentos con formato como PDF o similares, ...

- **TEXT** (Character Large OBject)

  Utilizado para almacenar datos de texto de gran tamaño (hasta 128 TB texto)

Tipos datos Oracle vs MySQL

Tipo de dato	Naturaleza	Tamaño/formato	Oracle	MySQL
TINYINT [UNSIGNED]	Entero	1 byte	X	X
SMALLNT [UNSIGNED]	Entero	2 bytes	X	X
MEDIUMINT [UNSIGNED]	Entero	3 bytes	X	X
INT [UNSIGNED]	Entero	4 bytes	X	X
BIGINT [UNSIGNED]	Entero	8 bytes	X	X
INTEGER [UNSIGNED]	Entero	2 bytes	X	X
DOUBLE [UNSIGNED]	Real Aproximado	4 bytes	X	X
FLOAT [UNSIGNED]	Real Aproximado	8 bytes	X	X
DECIMAL(long,decimal)	Real Exacto	Variable		X
NUMERIC(long,decimal)	Real Exacto	Variable		X
NUMBER(long,decimal)	Real Exacto	Variable	X	
DATE	Fecha	‘aaaa-mm-dd’	X	X
TIME	Hora	‘hh:mm:ss’		X
TIMESTAMP	Fecha y hora	‘aaaa-mm-dd hh:mm:ss’	X	X
DATESTIME	Fecha y hora	‘aaaa-mm-dd hh:mm:ss’		X
CHAR(longitud)	Caracteres	Longitud Fija	X	X
VARCHAR(longitud)	Caracteres	Longitud Variable	X	X
VAR2CHAR(longitud)	Caracteres	Longitud Variable	X	
BLOB	Objetos binarios	Longitud Variable	X	X
TEXT	Campos Memo	Longitud Variable		X
CLOB	Campos Memo	Longitud Variable	X	

Conversiones de tipo

ORACLE soporta conversiones implícitas y explicitas de valores de un tipo a otro.

Implícitas: Hechas por ORACLE de forma automática.

Reglas de conversión implícitas:

1. Con INSERT y UPDATE. Se convierte el valor al tipo de la columna afectada
2. Al comparar un CHAR con un NUMBER se convierte CHAR a NUMBER
3. Al comparar un CHAR con DATE se convierte a DATE
4. En asignaciones ORACLE convierte el tipo de la derecha al de la izquierda
5. En la concatenación se convierte los tipos no CHAR a CHAR o NCHAR.
6. En operaciones aritméticas o de comparación entre tipos carácter y no carácter los convierte a NUMBER, DATE o ROWID, según sea apropiado en casa caso
7. En operaciones aritméticas entre CHAR/VARCHAR2 y NCHAR/NVARCHAR2 se convierte a número.

Ejemplo: Se compara un CHAR con un NUMBER y se convierte el char a number.

```
Select nombre from emp where numemp = '7936' ;
```

Conversiones de tipo

Explícitas: Especificadas por el usuario a través de funciones.

Las más utilizadas son:

- Conversión número-cadena: **TO_CHAR** (número [, formato]).
- Conversión cadena-número: **TO_NUMBER** (cadena [,formato]).
- Conversión fecha-cadena: **TO_CHAR** (fecha [, formato]).
- Conversión cadena-fecha: **TO_DATE** (cadena [, formato]).

Expresiones y operadores condicionales

Las **condiciones son expresiones lógicas** (devuelven verdadero o falso) que se sitúan normalmente junto a una cláusula SQL que utilizan muchos comandos.

Dentro del **DDL** se utilizarán con la cláusula **CHECK** que sirve para establecer las **condiciones** que deben **cumplir** sobre los **valores** que se almacenarán **en** una **tabla**.

Las condiciones se construyen utilizando los operadores de comparación y los operadores lógicos.

Operadores de comparación

Comparan un valor de X con otro de Y.

X operador Y

- =
- < >, ^=, !=, -=
- <
- <=
- >
- >=

Ejemplos:

horas >= 10.5

nombre = 'PEPE'

fecha < '1-ene-93'

Sentencias DDL

DDL tiene 3 instrucciones básicas:

- **CREATE** tipo_objeto Nombre Definición

  Crea un objeto de un tipo determinado con un nombre.

  Tipos de objetos: DATABASE, TABLE, INDEX, etc.

  Ejemplo: *CREATE TABLE Actores (Nombre CHAR(50) PRIMARY KEY);*

- **ALTER** tipo_objeto Nombre Modificación.

  Modifica la definición de un objeto.

  Ejemplo: *ALTER TABLE Actores DROP COLUMN Fecha*, eliminaría la columna Fecha de la tabla Actores.

- **DROP** tipo_objeto Nombre.

  Elimina un tipo de objeto especificado mediante un nombre.

  Ejemplo: *DROP TABLE Actores*, borraría de la base de datos la tabla Actores junto con todos sus datos.

Base de Datos: *CREACIÓN*

Sintaxis para crear una base de datos, teniendo privilegios **DBA** (DataBase Administrator) en Oracle es **SYSDBA**:

CREATE DATABASE nombre_db;

Se pueden incluir más opciones, con todas las opciones sería:

CREATE {**DATABASE** | SCHEMA} [IF NOT EXISTS] **nombre_db** [especificación [, especificación] ...] especificación;

Donde especificación se refiera a: [DEFAULT] CHARACTER SET juego_caracteres

| [DEFAULT] COLLATE nombre_colación

Ejemplo: crear la base de datos Tienda , en MySQL:

- **CREATE DATABASE Tienda;**
- **CREATE DATABASE** prueba LOGFILE prueba**.**log MAXLOGFILES **25** MAXINSTANCES **10** ARCHIVELOG CHARACTER **SET** AL32UTF8 **NATIONAL** CHARACTER **SET** UTF8 DATAFILE prueba1**.**dbf AUTOEXTEND **ON** MAXSIZE **500**MB**;**

Base de Datos: *VISUALIZACIÓN Y MODIFICACIÓN*

VISUALIZACIÓN

Puedes comprobar si se crea, visualizando la lista de bases de datos con:

SINTAXIS: **SHOW DATABASES;**

MODIFICACIÓN

Podemos cambiar el juego de caracteres y su colación de una base de datos.

SINTAXIS: **ALTER DATABASE nombre_db** [modificacion];

Ejemplo: crear la base de datos Tienda :

ALTER DATABASE Tienda

CHARACTER SET utf8 COLLATE utf8_unicode_ci;

Base de Datos: *ELIMINACIÓN*

Ten cuidado al borrar la base de datos, se perderá de toda la información almacenada en ella .

SINTAXIS: **DROP DATABASE nombre_db**;

Ejemplo: borrado de la base de datos Tienda :

```
DROP DATABASE Tienda;
```

Consejo: asegúrese de tener privilegios de administrador antes de eliminar cualquier base de datos. Una vez que se borre una base de datos, se puede comprobar en la lista de bases de datos con:

SHOW DATABASES;

Esquema: *CREACIÓN*

En Oracle para crear un esquema o usuario. La forma más sencilla es:

CREATE USER nombre **IDENTIFIED BY** contraseña;

Se suelen añadir diversas cláusulas. Una sentencia más detallada es:

CREATE USER nombre
IDENTIFIED BY clave
DEFAULT TABLESPACE users
QUOTA 10M **ON** users
TEMPORARY TABLESPACE temp
QUOTA 5M **ON** temp
PASSWORD EXPIRE**;**

Esquema: *MODIFICACIÓN*

Cada parámetro indicado en la creación del esquema puede modificarse mediante la instrucción **ALTER USER**, que se utiliza igual que CREATE USER.

Ejemplo:

- **ALTER USER** nombre IDENTIFIED **BY** "nuevaclave"**;**
- **ALTER USER** nombre QUOTA UNLIMITED **ON** users**;**

Esquema: *ELIMINACIÓN*

Sintaxis:

DROP USER usuario **[CASCADE];**

La opción **CASCADE** elimina los objetos del esquema del usuario antes de eliminar al propio usuario.

Es obligatorio si el esquema contiene objetos.

Tabla: Creación

El nombre de las tablas debe cumplir las siguientes reglas:

- Deben comenzar con una letra
- No deben tener más de 30 caracteres
- Sólo se permiten utilizar letras del alfabeto (inglés), números o el signo de subrayado (también el signo $ y #, pero se utilizan de manera especial, no son recomendados)
- No puede haber dos tablas con el mismo nombre para el mismo usuario (pueden coincidir los nombres si están en distintos esquemas)
- No puede coincidir con el nombre de una palabra reservada de SQL

Tabla: Creación

La creación de tablas tiene una sintaxis compleja, empezaremos por lo básico. SINTAXIS:

```
CREATE TABLE nombre_tabla (
    columna1 tipo_dato [ restricciones de columna1 ], ...
    [ restricciones de tabla ]
);
```

- Definición de columnas (nombre, tipos de datos, longitud).

  Una tabla puede tener hasta 254 columnas
- Restricciones de integridad
- Características de almacenamiento de objeto donde se almacena y cuanto ocupa

Tabla: Creación

Sintaxis completa

```
CREATE TABLE nombre_tabla (
    campo1 tipo_dato [longitud] [NOT NULL] [UNIQUE] [PRIMARY KEY]
     [CHECK condición ] [DEFAULT valor] ,
    ....
    campoN tipo_dato [longitud] [NOT NULL] [UNIQUE] [PRIMARY KEY]
     [CHECK condición ] [DEFAULT valor]

    [, CONSTRAINT nombre_restriccion PRIMARY KEY (campo, [campo])]

    [, CONSTRAINT nombre_restriccion FOREIGN KEY (campo, [campo])
     REFERENCES nombre_tabla (campo, [campo])
        [ON UPDATE [NOT ACTION | SET DEFAULT | SET NULL | CASCADE]
        [ON DELETE [NOT ACTION | SET DEFAULT | SET NULL | CASCADE] ]
    [, FOREIGN KEY ...]
);
```

Tabla: Creación, opciones

Es obligatorio crear una restricción de tabla cuando una misma restricción afecte a varias columnas. Por ejemplo si tenemos una clave primaria compuesta por varios campos, debemos establecer una restricción de tabla, no de columna.

El significado de las distintas opciones de CREATE TABLE es:

PRIMARY KEY: establece un atributo o conjunto como la clave primaria de la tabla. Esta restricción ya implica las restricciones UNIQUE y NOT NULL.

UNIQUE: impide que se introduzcan valores repetidos para ese atributo o conjunto de atributos. No se puede utilizar junto con PRIMARY KEY. Se utiliza para claves alternativas.

NOT NULL: evita que se introduzcan filas en la tabla con valor NULL para ese atributo. No se utiliza con PRIMARY KEY.

DEFAULT valor_por_defecto: permite asignar un valor por defecto al campo que se está definiendo.

Tabla: Creación, opciones

CHECK (condición): permite establecer condiciones que deben cumplir los valores de la tabla que se introducirán en dicha columna.

Si un CHECK se especifica como una restricción de columna, la condición sólo se puede referir a esa columna.

Si el CHECK se especifica como restricción de tabla, la condición puede afectar a todas las columnas de la tabla.

Sólo se permiten condiciones simples, por ejemplo, no está permitido referirse a columnas de otras tablas o formular subconsulas dentro de un CHECK.

Además las funciones SYSDATE y USER no se pueden utilizar dentro de la condición. En principio están permitidas comparaciones simples de atributos y operadores lógicos (AND, OR y NOT).

Tabla: Creación, opciones

FOREIGN KEY: define una clave externa de la tabla respecto de otra tabla.
Esta restricción especifica una columna o lista de columnas como clave externa de una tabla referenciada. No se puede definir una restricción de integridad referencial que se refiere a una tabla antes de que dicha tabla haya sido creada.
Una clave externa referencia a una clave primaria completa de la tabla padre.

ON DELETE elimina

ON UPDATE actualiza el valor

CASCADE: especifica que se mantenga automáticamente la integridad referencial en los valores de la llave externa correspondientes a un valor borrado o modificado de la tabla referenciada (tabla padre).
Si se omite esta opción no se permitirá borrar o modificar valores de una tabla que sean referenciados como llave externa en otras tablas.

SET NULL: especifica que se ponga a NULL los valores de la llave externa correspondientes a un valor borrado o modificado de la tabla referenciada.

NOT ACTION: no hace nada

SET DEFAULT: pone el valor por defecto

Tabla: Creación

En la definición de una tabla pueden aparecer **varias** cláusulas **FOREIGN KEY**, tantas como llaves externas tenga la tabla, sin embargo **sólo** puede existir **una llave primaria**, si bien esta llave primaria puede estar formada por varios atributos.

La cláusula **CONSTRAINT nombre_restricción** establece un nombre determinado para la restricción de integridad, lo cual permite buscar en el Diccionario de Datos de la base de datos con posterioridad y futuras modificaciones.

Tabla: Creación, Constraints

- Asigne un nombre a la restricción, en caso contrario el Server generará uno usando el formato SYS_Cn.
- Se puede crear una restricción
  - En el momento de crear la tabla.
  - Después que la tabla ha sido creada.
- Se puede definir un restricción a nivel de columna o a nivel de tabla.

Tabla: Creación

Ejemplos

```
CREATE TABLE TPais(
      nPaisID NUMBER(3,0) NOT NULL PRIMARY KEY,
      cNombre VARCHAR(30) NOT NULL
);

CREATE TABLE TEditorial (
      nEditorialID NUMBER(3,0) PRIMARY KEY,
      cNombre VARCHAR(40) NOT NULL,
      nPaisID NUMBER(3,0) DEFAULT 724,
      CONSTRAINT FK_PAIS FOREIGN KEY ( nPaisID) REFERENCES Tpais(nPaisID)
            ON UPDATE CASCADE
            ON DELETE NO ACTION
);
```

Tabla: Creación

Ejemplos

```
CREATE TABLE Socio (
        NIF CHAR(9)NOT NULL UNIQUE PRIMARY KEY,
        Nombre VARCHAR(40) NOT NULL,
        Apellidos VARCHAR(60) NOT NULL,
        Direccion VARCHAR(100),
        Telefono CHAR(12) NOT NULL,
        Nacimiento DATE NOT NULL,
        Alta DATE NOT NULL CHECK Alta >= “2003-09-01"
);

CREATE TABLE Prestamo (
        Signatura VARCHAR(15) NOT NULL,
        NIF CHAR(9) NOT NULL,
        dPrestamo Date NOT NULL,
        CONSTRAINT PK_PREST PRIMARY KEY (Signatura, NIF, dPrestamos),
        FOREIGN KEY (Signatura) REFERENCES Tejemplar (Signatura) ,        FOREIGN KEY
(NIF) REFERENCES Socio (NIF)
);
```

Tabla: Creación

A partir de otra tabla

Se puede crear una copia de una tabla existente.

La nueva tabla obtiene las mismas definiciones de columna.

Se pueden seleccionar todas las columnas o específicas.

La nueva tabla se completará con los valores existentes de la tabla anterior.

Sintaxis

```
CREATE TABLE new_table_name AS
    SELECT column1, column2,...
    FROM existing_table_name
    WHERE ....;
);
```

Creación de una tabla

Realizar SQL del siguiente ejemplo

Tenemos que almacenar los datos del módulo que estamos cursando y el profesor que los imparte.

Del módulo necesitamos conocer el código del modulo (Texto max 5 caracteres), su nombre, ciclo, curso (Numérico). No se puede repetir ningún modulo con el mismo código.

De los profesores debemos conocer el código del profesor (Texto max 5 caracteres) y su nombre.

Por otra parte debemos conocer cuales módulos imparten cuales profesores, conociendo que un modulo puede ser impartido por muchos profesores y un profesor puede impartir muchos módulos.

Tabla: Modificación

Cambiar de nombre una tabla

La orden RENAME permite el cambio de nombre de cualquier objeto. Sintaxis:

RENAME nombre **TO** nombre_nuevo**;**

Ejemplo:

RENAME COCHES **TO** AUTOMOVILES**;**

Cambia el nombre de la tabla COCHES y a partir de ese momento se llamará AUTOMOVILES

Tabla: Modificación

ALTER TABLE permite hacer cambios en la estructura de una tabla: añadir columna, borrar columna, modificar columna.

Añadir Columnas

ALTER TABLE nombre **ADD (**
    columna1 tipo **[** restricciones **]**
    [, columna2 tipo **[** restricciones **] …]);**

Ejemplos:

- Añadir la columna “fechaMatric” a la tabla VEHÍCULOS:

ALTER TABLE VEHICULOS **ADD (** fechaMatric DATE **);**

- Añadir las columnas “fechaMatric” y “tipoFaros” a la tabla VEHÍCULOS:
- **ALTER TABLE** VEHICULOS **ADD (** fechaMatric DATE**,** tipoFaros VARCHAR2**(20) NOT NULL):**

Tabla: Modificación

- Eliminar una columna

ALTER TABLE nombre_tabla **DROP (**nombre_campo**);**

Ejemplo: **ALTER TABLE** VEHICULOS **DROP (**tipoFaros**);**

- Modificar una columna

ALTER TABLE nombre_tabla **MODIFY (**
campo1 tipo_dato [restricciones] [,
campo2 tipo_dato [restricciones]
…]
);

Ejemplo: **ALTER TABLE** AUTOMOVILES MODIFY **(**
color VARCHAR2**(20) NOT NULL,** codTaller VARCHAR2**(15));**

Tabla: Modificación

- Añadir una restricción

ALTER TABLE <tabla> ADD CONSTRAINT <nombre> [CHECK] [PRIMARY KEY] [FOREIGN KEY];

- Eliminar una restricción

ALTER TABLE <tabla> DROP CONSTRAINT <nombre>;

Tabla: Eliminación

Sintaxis:

DROP TABLE nombre_tabla **[CASCADE CONSTRAINTS];**

CASCADE CONSTRAINTS permite eliminar una tabla que contenga atributos referenciados por otras tablas, eliminando también todas esas referencias.

Si la clave principal de la tabla es una clave foránea en otra tabla y no utiliza la opción CASCADE CONSTRAINTS, entonces no se podrá eliminar la tabla.

El borrado de una tabla es irreversible y no hay ninguna petición de confirmación, por lo que conviene ser muy cuidadoso con esta operación. Al borrar una tabla se borran todos los datos que contiene.

Tabla: Eliminación

Ejemplos

Ejemplos:

- **DROP TABLE** COCHES**;**

  Se eliminará la tabla COCHES, siempre que su clave principal no sea clave foránea de ninguna tabla de la BD.

- **DROP TABLE** COCHES **CASCADE CONSTRAINTS;**

  Se eliminará la tabla COCHES aunque su clave principal sea clave foránea de alguna tabla de la BD.

  Automáticamente se borrará la restricción de clave foránea asociada.