DDL – RESTRICCIONES DE TABLAS

CREACCIÓN DE TABLAS

Una tabla es una colección de datos estructurados en campos. Cada fila de la tabla, toma un único valor en cada campo.

Para crear una tabla con comandos SQL, debemos tener en cuenta el tipo de datos que van a contener los campos, y el tamaño máximo que van a ocupar.

Los tipos de datos más básicos en Oracle son:

PALABRA CLAVE	TIPO	EJEMPLOS
VARCHAR2(tamaño) VARCHAR(tamaño)	Variables alfanuméricas, el tamaño expresado será el número máximo de caracteres de ese campo.	124555 1234RTT Alfonso Rodríguez
NUMBER (precisión,escala) NUMBER NUMBER(precisión)	Variable numéricas. La precisión es el número máximo de dígitos a la izquierda de la coma decimal, y la escala es el número máximo de dígitos a la derecha de la coma decimal.	56 45,67 56E8 0,89E-7
DATE	Fechas	2006/12/31
TIMESTAMP	Fechas en Año,Mes y Día y horas, minutos y segundos	2006-10-03 12:09:24
BLOB	Un objeto binario de hasta 4 gigagytes	Imagen / Audio / Video
CLOB	Una cadena de caracteres de hasta 4 gigabytes	
BFILE	Un localizador a un fichero fuera de la BD	
LONG	Una cadena de caracteres muy larga de hasta 2 Gb.	Un capítulo de un libro.
RAW(tamaño)	Un objeto binario de hasta 2000 bytes	Una foto pequeña.

Oracle acepta más tipos de datos por compatibilidad con otras base de datos, pero son solo formas redundantes de definir los mismo tipos de datos.

Un **ejemplo** de creación de una tabla llamada empleados sería:

```
CREATE TABLE  Empleados(
DNI           VARCHAR(10),
Nombre        VARCHAR(20),
Edad          NUMBER(2,0)
);
```

RESTRICCIONES DE LOS CAMPOS

A los campos de una tabla se le pueden imponer ciertas restricciones de tal forma que los valores válidos en ese campo deban de cumplir ciertas propiedades además del tipo de dato.

CLAVE PRIMARIA

Toda tabla tiene una clave primaria, que es un campo o conjunto de campos, cuyos valores no se pueden repetir y no pueden tomar valor nulo.

La clave primaria puede estar formada por varios campos, cuya unión cumple la condición anterior.

La clave primaria en una tabla es única, es decir solo puede haber una clave primaria aunque como ya dijimos puede estar formada por más de un campo. En la definición de una tabla la restricción de PRIMARY KEY solo puede aparecer una vez.

Las restricciones en una tabla pueden ser impuestas de dos formas:

- A nivel de campo:

```
CREATE TABLE  Empleados(
DNI           VARCHAR(10)   PRIMARY KEY,
Nombre        VARCHAR(20),
Edad          NUMBER(2,0)
);
```

- a nivel de tabla:

```
CREATE TABLE  Empleados(
DNI           VARCHAR(10),
Nombre        VARCHAR(20),
Edad          NUMBER(2,0),
PRIMARY KEY (DNI)
);
```

Cuando la clave primaria está formada por dos o más campos, la restricción solo puede imponerse a nivel de tabla.

```
CREATE TABLE  Empleados_Puesto(
DNI           VARCHAR(10),
Cod_puesto    VARCHAR(5),
Hora          TIMESTAMP,
PRIMARY KEY (DNI,CodPuesto)
);
```

Toda restricción tiene un nombre puesto, con el fin de borrar la restricción sin borrar la tabla ni el campo ó campos que afecta. En los ejemplo anteriores como el usuario en el momento de imponer la restricción no especificó un nombre para ella, el propio sistema le impuso dicho nombre, por ejemplo si quisiéramos que en la tabla empleados DNI dejara de ser clave primaria, tendríamos que consultar como ha nombrado el sistema dicha restricción y luego borrarla. Para especificar en el momento de la creación de la tabla un nombre para la restricción de clave primaria sería:

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre          VARCHAR(20),
Edad            NUMBER(2,0)
);
```

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10),
Nombre          VARCHAR(20),
Edad            NUMBER(2,0),
CONSTRAINT ClaveEmpl PRIMARY KEY (DNI)
);
```

VALORES POR DEFECTO

Cuando una fila en un campo es normal que tenga siempre el mismo valor, conviene darle dicho valor por defecto. Esto quiere decir que al insertar una nueva fila, si no indicamos lo contrario el valor de la fila en ese campo será el predeterminado.

Por ejemplo:

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0),
Cargo           VARCHAR(20)  DEFAULT  'Administrativo'
);
```

Esto quiere decir que en todas las fias que insertemos se nos propondrá de forma automática el cargo de administrativo.

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0),
FehaAlta        DATE DEFAULT SYSDATE
);
```

La función SYSDATE asigna la fecha actual del sistema operativo.

VALORES REQUERIDOS

Algunos campos, aunque no sean claves primarias, por el propio diseño de la base de datos, se requiere que no acepte valores nulos, es decir que cualquier registro o fila no puede quedar en blanco en dicho campo., aunque si se puede repetir.

La restricción es NOT NULL y se establece únicamente a nivel de campo.: Esta restricción no puede aplicarse al conjunto de dos campos, pero si a varios campos de una tabla.

Esta restricción acepta un nombre como en el caso de la clave primaria, pero no resulta muy útil porque para su modificación no se precisa un nombre de restricción.

```
CREATE TABLE  Empleados(
DNI           VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre        VARCHAR(20)   NOT NULL,
Edad          NUMBER(2,0)
);
```

```
CREATE TABLE  Empleados(
DNI           VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre        VARCHAR(20)   NOT NULL,
Apellidos     VARCHAR(30)   NOT NULL,
Edad          NUMBER(2,0)
);
```

```
CREATE TABLE  Empleados(
DNI           VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre        VARCHAR(20)   CONSTRAINT nn_nombre NOT NULL,
Apellidos     VARCHAR(30)   NOT NULL,
Edad          NUMBER(2,0)
);
```

Valores de la tabla válidos

DNI	NOMBRE	APELLIDOS	EDAD
1111111111U	Carlos	González	23
2222222222J	Carlos	González	
3333333333L	Juan	Pérez	

De esta forma el campo Nombre no podría quedar en blanco, al insertar una fila si no ponemos el nombre del empleado no nos deja guardar el resto de campos, pero puede haber dos empleados con el mismo nombre.

Si un campo es clave primaria, no es necesario que se le defina la restricción NOT NULL, ya que está exigencia ya nos la dá la clave primaria.

VALORES ÚNICOS

Si queremos que un campo que no sea clave primaria, no acepte 2 filas con el mismo valor, pero que sí pueden quedar en blanco,.se aplicaría la restricción UNIQUE, puede aplicarse a nivel de tabla, o a nivel de campo

La restricción puede aplicarse a un conjunto de campos como la clave primaria.

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0),
UNIQUE (Apellidos)
);
```

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0),
UNIQUE (Nombre,Apellidos)
);
```

También puede darse un nombre a la restricción, para luego modificarla sin borrar la tabla.

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30)   CONSTRAINT ApellidoUnico UNIQUE,
Edad            NUMBER(2,0),
);
```

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0),
CONSTRAINT ClaveApel   UNIQUE (Apellidos)
);
```

Valores de la tabla válidos:

DNI	NOMBRE	APELLIDOS	EDAD
1111111111U	Carlos	González	23
2222222222J	Carlos	García	45
3333333333L	Juan	Pérez	
4444444444G	Juan		
555555555M	María		

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0),
CONSTRAINT RestrUnicaNomApel   UNIQUE (Nombre,Apellidos)
);
```

Valores de la tabla válidos:

DNI	NOMBRE	APELLIDOS	EDAD
1111111111U	Carlos	González	23
2222222222J	Carlos	García	45
3333333333L	Juan	Pérez	
4444444444G	Juan		
555555555M	María		
666666666R	María	González	

hay que tener en cuenta que no es lo mismo aplicar la restricción a dos campos de forma separada que aplicar la restricción a dos campos de forma conjunta, por ejemplo la sentencia:

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   CONSTRAINT ClaveEmpl PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0),
UNIQUE (Nombre),
UNIQUE (Apellidos)
);
```

No adoptaría las filas de nombres, ni de apellidos repetidos, es decir la tabla anterior , solo tendría de valores válidos:

DNI	NOMBRE	APELLIDOS	EDAD
1111111111U	Carlos	González	23
3333333333L	Juan	Pérez	
555555555M	María		

CHEQUEO DE VALORES

Los valores en algunos campos, además del tipos de dato, deben estar limitados por alguna otra norma, por ejemplo, la edad es un número, pero como debe estar en edad laboral, deberiamos comprobar que la edad está entre los 18 y los 70 y no admitir ningún otro valor. Esto se conseguiría con:

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0) CHECK (Edad BETWEEN 18 AND 70),
FechaAlta       DATE DEFAULT SYSDATE
);
```

También puede establecerse a nivel de tabla:

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0) ,
FechaAlta       DATE DEFAULT SYSDATE,
CHECK (Edad BETWEEN 18 AND 70)
);
```

También puede darse un nombre a la restricción:

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0) CONSTRAINT  ck_edad CHECK (Edad BETWEEN 18 AND 70),
FechaAlta       DATE DEFAULT SYSDATE
);
```

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Edad            NUMBER(2,0) ,
FechaAlta       DATE DEFAULT SYSDATE,
CONSTRAINT ck_edad CHECK (Edad BETWEEN 18 AND 70)
);
```

CLAVES AJENAS

La restricción de clave ajena se establece sobre algún campo que necesariamente es clave primaria en otra tabla, ambos campos deben contener los mismos valores, es decir deben ser del mismo tipo aunque pueden llamarse de forma distinta.

La clave ajena señala una relación con la tabla donde dicho campo es clave primaria.

Además los valores de la tabla donde es clave ajena, deben existir previamente en la tabla donde es clave primaria. Esta condición se denomina **integridad referencial.**

Por ejemplo creamos un tabla DEPARTAMENTOS:

```
CREATE TABLE Departamentos(
Codigo_dep          VARCHAR(3) PRIMARY KEY,
Nombre_dep          VARCHAR(10)
);
```

En ella introducimos los siguientes valores:

Codigo_dep	**Nombre_dep**
ADM	Administración
FOR	Formación
DIS	Diseño

Ahora la tabla empleados , con un campo que representa el código del departamento en el que trabaja y exigimos integridad referencial códigos que se recogen como valores en la tabla anterior.

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Departamento            VARCHAR(3) ,
CONSTRAINT fk_dep_emp FOREIGN KEY (Departamento)  REFERENCES Departamentos
);
```

Esto quiere decir que los valores de las filas en el campo departamento, deben ser exclusivamente los valores contenidos en el codigo_dep (clave primaria) de la tabla Departamentos.

Es decir son válidos:

DNI	**NOMBRE**	**APELLIDOS**	**DEPARTAMENTO**
1111111111U	Carlos	González	ADM
3333333333L	Juan	Pérez	FOR
555555555M 444444444Y	María Pilar	 Lópes	 FOR

Es decir la clave ajena puede tomar valor nulo , pero no puede asignarse ningún empleado a un departamento que no exista.

La clave ajena siempre es interesante establecerse a nivel de tabla. Con el propósito de más adelante poderla activar o desactivar (Enable/Disable).

El campo clave ajena debe ser del mismo tipo que su referencia en la clave primaria, aunque como observamos en el ejemplo anterior, no tienen porque llamarse igual, aunque sí que es posible.

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Codigo_dep              VARCHAR(3) ,
FOREIGN KEY (Codigo_dep)  REFERENCES Departamentos
);
```

Un campo puede hacer referencia a una clave primaria formada por más de un campo, por ejemplo:

```
CREATE TABLA Viviendas (
Calle            VARCHAR(29),
Numero          NUMBER(4,0),
Piso_letra      VARCHAR(8),
PRIMARY KEY (Calle, Numero, Piso_letra)
);
```

```
CREATE TABLE Ciudadanos(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
Calle            VARCHAR(29),
Numero          NUMBER(4,0),
Piso_letra      VARCHAR(8),
CONSTRAINT ciu_viv FOREIGN KEY (Calle, Numero, Piso_letra) REFERENCES Viviendas
);
```

También puede darse el caso de un campo que sea clave primaria o parte de una clave primaria y clave ajena al mismo tiempo:

Supongamos que en una empresa un empleado puede ser de dos departamentos a la vez:

```
CREATE TABLE Departamentos(
Codigo_dep              VARCHAR(3) PRIMARY KEY,
Nombre_dep              VARCHAR(10)
);
```

```
CREATE TABLE  Empleados(
DNI             VARCHAR(10)   PRIMARY KEY,
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
);
```

No podemos insertar el campo codigo_dep en empleados pues cada empleado puede trabajar en más de un departamento, creamos una nueva tabla que recoja esa información:

```
CREATE TABLE Departamentos_Empleados (
Codigo_dep              VARCHAR(3) ,
DNI                     VARCHAR(10),
Función         VARCHAR(20),
PRIMARY KEY(Codigo_dep, DNI),
CONSTRAINT ca1 FOREIGN KEY (Codigo_dep) REFERENCES Departamentos,
CONSTRAINT ca2 FOREIGN KEY (DNI) REFERENCES Empleados
);
```

Otro ejemplo, supongamos que añadimos una tabla con los familiares del empleado, cuya clave primaria sea el DNI del familiar y un código interno que es:

C→Conyuge

H1→1ºHijo

H2→2ºHijo

Etc..

La tabla sería:

```
CREATE TABLE  Familiares(
DNI_empleado            VARCHAR(10),
Codigo_parentesco               VARCHAR(4),
Nombre          VARCHAR(20)   NOT NULL,
Apellidos       VARCHAR(30),
PRIMARY KEY (DNI_empleado, Codigo_parentesco),
CONSTRAINT fam_empl FOREIGN KEY (DNI_empleado) REFERENCES Empleados
);
```