

SQL ORACLE

¿Qué es SQL Oracle?

Es un motor de base de datos desarrollado por Oracle Corporation. Está construido sobre lenguaje ensamblador, C y C++.

Lleva más de 40 años en el mercado. Actualmente se enfoca en servicios de base de datos en la nube.

Al igual que SQL SERVER, la BDD de Oracle trabaja con el lenguaje estándar SQL.

Introducción – Lenguaje SQL

SQL puede dividir su lenguaje general en dos tipos esenciales.

- Lenguaje de manipulación de datos (DML).
 - Lenguaje proporcionado por el gestor de base de datos para llevar a cabo tareas de consulta y manipulación de datos con tenidos en la base de datos.

Introducción – Lenguaje SQL

- Lenguaje de definición de datos (DDL).
 - Lenguaje proporcionado por el gestor de base de datos para llevar a cabo tareas de definición de la estructura en donde se almacenarán los datos.

CREATE TABLE

La sentencia **CREATE TABLE** representa la instrucción para la creación de una tabla y el conjunto de atributos que representa al objeto.

```
CREATE TABLE nombre_tabla
(
    columna1 tipo_de_dato(tamaño),
    columna2 tipo_de_dato(tamaño),
    columna3 tipo_de_dato(tamaño),
    ....
);
```

```
--numéricos
NUMBER
INT
FLOAT
LONG
--caracteres
CHAR(n)
VARCHAR2(n)
--fechas
DATE
```

Ejemplo CREATE TABLE

```
CREATE TABLE CLIENTE (  
    RUT VARCHAR2(10) NOT NULL,  
    NOMBRE VARCHAR2(20),  
    EDAD NUMBER CHECK(EDAD > 0),  
    CORREO VARCHAR2(40) UNIQUE,  
    TELEFONO NUMBER,  
    FECHA DATE,  
    DIRECCION VARCHAR2(50)  
);
```

DROP TABLE

La sentencia DROP TABLE representa la instrucción para el borrado de tablas de la base de datos.

Algunas consideraciones con esta instrucción:

- Elimina el objeto “tabla” de la base de datos.
- Elimina los datos que haya tenido el objeto.
- Una vez realizada la acción, no es posible recuperar ni la tabla, ni los datos.

Del ejemplo anterior, podríamos eliminar la tabla cliente ejecutando: DROP TABLE CLIENTE;

ALTER TABLE

La sentencia **ALTER TABLE** representa la instrucción que permite modificar la estructura de una tabla.

```
--ELIMINAR COLUMNA TELEFONO
ALTER TABLE CLIENTE DROP COLUMN TELEFONO;

--AGREGAR COLUMNA CELULAR TIPO NUMBER
ALTER TABLE CLIENTE ADD CELULAR NUMBER;

--MODIFICAR EL TIPO DE DATO DE UNA COLUMNA
--CAMBIAR LA FECHA DE "TIPO" DATE
--A "VARCHAR2(8)"
ALTER TABLE CLIENTE MODIFY FECHA VARCHAR2(10);
```


PRIMARY KEY

La sentencia `PRIMARY KEY` representa la instrucción para establecer la clave primaria de una tabla.

El atributo que representará la clave primaria debe poseer la restricción de “not null”.

En el caso del ejemplo sobre la tabla cliente, su clave primaria corresponde al atributo rut.

PRIMARY KEY (CLIENTE)

```
CREATE TABLE CLIENTE (  
    RUT VARCHAR2(10) NOT NULL,  
    NOMBRE VARCHAR2(20),  
    EDAD NUMBER CHECK(EDAD > 0),  
    CORREO VARCHAR2(40) UNIQUE,  
    TELEFONO NUMBER,  
    FECHA DATE,  
    DIRECCION VARCHAR2(50),  
    PRIMARY KEY(RUT)  
);
```

PRIMARY KEY CON ALTER TABLE

Otra forma de declarar la PRIMARY KEY es utilizando la sentencia ALTER TABLE. Realizando una declaración explícita de la clave primaria.

En esta forma de declaración, es necesario establecer una regla, la cual posee un “nombre” que tendrá la primary key que se creará.

Declaración de PK con ALTER TABLE

```
CREATE TABLE CLIENTE (  
    RUT VARCHAR2(10) NOT NULL,  
    NOMBRE VARCHAR2(20),  
    EDAD NUMBER CHECK(EDAD > 0),  
    CORREO VARCHAR2(40) UNIQUE,  
    TELEFONO NUMBER,  
    FECHA DATE,  
    DIRECCION VARCHAR2(50),  
    PRIMARY KEY(RUT)  
);
```

```
--PK_CLIENTE REPRESENTA EL NOMBRE DE LA PK DE CLIENTE  
ALTER TABLE CLIENTE ADD CONSTRAINT PK_CLIENTE  
PRIMARY KEY(RUT);  
  
--SE ELIMINA LA PK APODADA PK_CLIENTE  
ALTER TABLE CLIENTE DROP CONSTRAINT PK_CLIENTE;
```


FOREIGN KEY

La sentencia FOREIGN KEY representa la instrucción para establecer la clave foránea de una tabla que referencia a otra.

La conexión de acuerdo a cada modelo, se realiza entre los atributos que representan la foránea y primaria entre las tablas respectivas.

FOREIGN KEY (EJEMPLO)

```
CREATE TABLE CLIENTE (  
    RUT VARCHAR2(10) NOT NULL,  
    NOMBRE VARCHAR2(20),  
    EDAD NUMBER CHECK(EDAD > 0),  
    CORREO VARCHAR2(40) UNIQUE,  
    TELEFONO NUMBER,  
    FECHA DATE,  
    DIRECCION VARCHAR2(50),  
    PRIMARY KEY(RUT)  
);  
  
CREATE TABLE BOLETA(  
    CODIGOBOLETA NUMBER NOT NULL,  
    RUT VARCHAR2(10) NOT NULL,  
    FECHA DATE,  
    TOTALBOLETA NUMBER,  
    PRIMARY KEY(CODIGOBOLETA),  
    FOREIGN KEY(RUT) REFERENCES  
    CLIENTE(RUT)  
);
```



FOREIGN KEY CON ALTER TABLE

De la misma forma en como declaramos la primary key con ALTER TABLE, podemos realizar la declaración de la foreign key.

Es necesario establecer una regla, la cual representa el “nombre” que tendrá la foreign key que se creará.

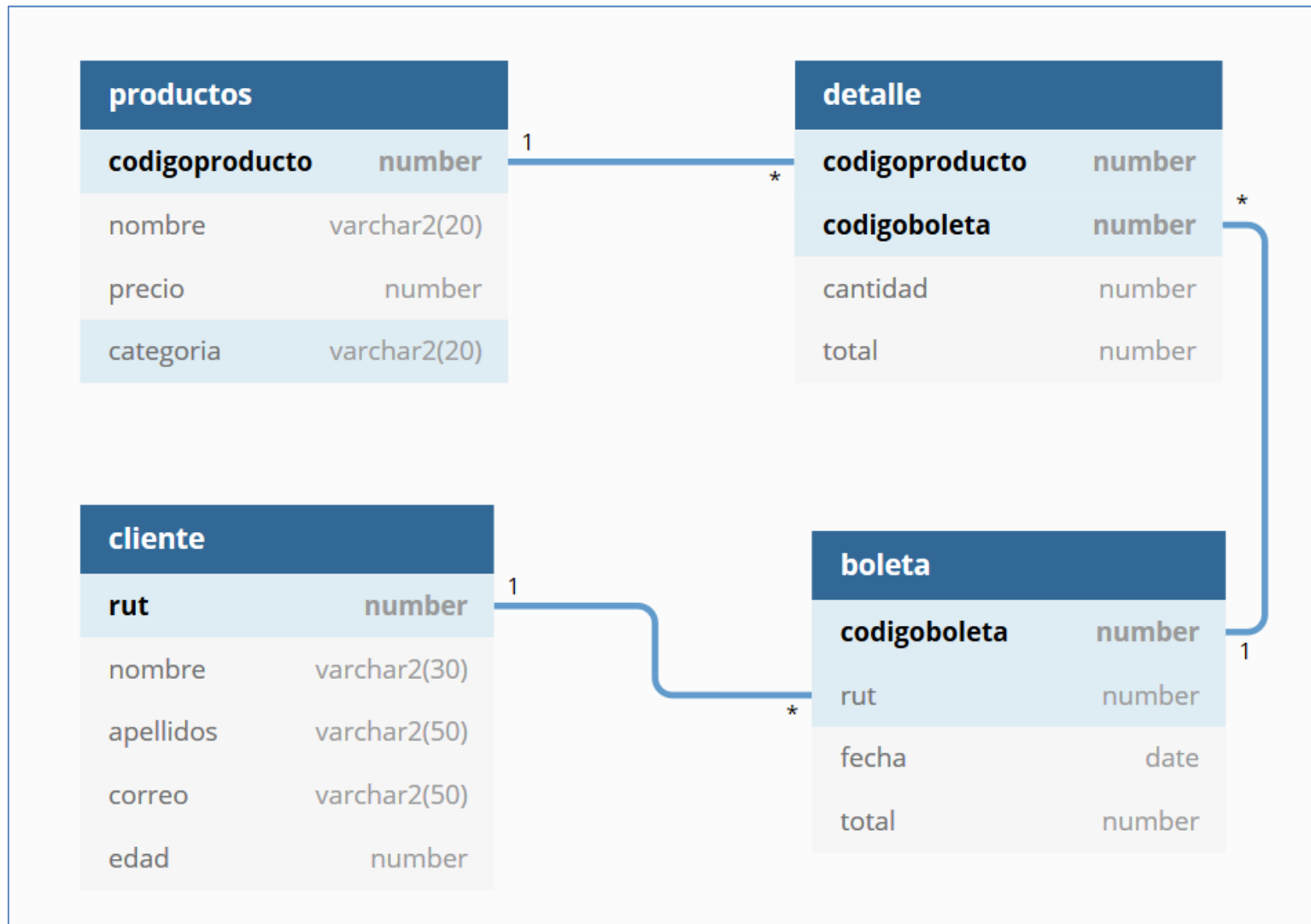
Declaración de FK con ALTER TABLE

```
CREATE TABLE BOLETA(  
  CODIGOBOLETA NUMBER NOT NULL,  
  RUT VARCHAR2(10) NOT NULL,  
  FECHA DATE,  
  TOTALBOLETA NUMBER  
);
```

```
CREATE TABLE CLIENTE(  
  RUT VARCHAR2(10) NOT NULL,  
  NOMBRE VARCHAR2(20),  
  EDAD NUMBER CHECK(EDAD > 0),  
  CORREO VARCHAR2(40) UNIQUE,  
  TELEFONO NUMBER,  
  FECHA DATE,  
  DIRECCION VARCHAR2(50)
```

```
--PK_CLIENTE REPRESENTA EL NOMBRE DE LA PK DE CLIENTE  
ALTER TABLE CLIENTE ADD CONSTRAINT PK_CLIENTE PRIMARY KEY(RUT);  
  
--PK_BOLETA REPRESENTA EL NOMBRE DE LA PK DE BOLETA  
ALTER TABLE BOLETA ADD CONSTRAINT PK_BOLETA PRIMARY KEY(CODIGOBOLETA);  
  
--FK_RUT_BOLETA REPRESENTA EL NOMBRE DE LA FK DE RUT EN BOLETA  
ALTER TABLE BOLETA ADD CONSTRAINT FK_RUT_BOLETA FOREIGN KEY(RUT)  
REFERENCES CLIENTE(RUT) ON DELETE CASCADE;
```


Modelo del Semestre



Funcionamiento del Laboratorio

- El laboratorio está destinado a realizar trabajo práctico (programación).
- Cada laboratorio tendrá una introducción teórica de la clase antes de comenzar con lo práctico.
- El material de los laboratorios estará disponible en el LMS en la pestaña “Laboratorios”.

Programas a utilizar

- Oracle Database Express Edition (XE) Release 11.2.0.2.0 (11gR2).
 - Disponible: [Oracle Express](#)
- SQL Developer 19.2
 - Disponible: [SQL Developer](#)

Programa global del curso

- Consultas
- Procedimientos
- Funciones
- Cursores
- Triggers
- Reglas ECA (Triggers)
- Base de datos (Objetos - No SQL)
- Conexión Cliente – Servidor (Proyecto Final)

Proyecto Final (Cliente – Servidor)

- Usar cliente web (HTML – CSS)
- Conexión entre el cliente y servidor.
 - PHP
 - Python (Por confirmar)
- Cliente flaco
 - Solo envía información al servidor (BDD)
- Servidor gordo
 - Realizar toda la lógica del sistema (BDD)

Atención a estudiantes

- Virtual:
 - [Facebook](#)
 - Correo: ftapia46@gmail.com
- Presencial:
 - Por acuerdo.

