**EJERCICIO 1.**

--TABLA FAMILIA. Contine las familias a las que perteneces los productos.

CREATE TABLE FAMILIA(

Codfamilia NUMBER(3,0) PRIMARY KEY,

Denofamilia VARCHAR2(50) NOT NULL UNIQUE

);

--TABLA PRODUCTO. Contine información general sobre los productos que distribuye la empresa a las tiendas.

CREATE TABLE PRODUCTO(

Codproducto NUMBER(5,0) PRIMARY KEY,

Denoproducto VARCHAR2(20) NOT NULL,

Descripcion VARCHAR2(100),

PrecioBase NUMBER (6,2) NOT NULL,

PorcReposicion NUMBER(3,0),

UnidadesMinimas NUMBER(4,0) NOT NULL,

CodFamilia NUMBER(3,0) NOT NULL CONSTRAINT pro\_cod\_FK REFERENCES FAMILIA,

CONSTRAINT pro\_ck CHECK(PrecioBase > 0 AND PorcReposicion > 0 AND UnidadesMinimas > 0)

);

Cada restricción de check debe de ir aparte, es decir, cada check con su campo, no mezclar en una misma restricción varias cosas que afecten a distintos campos, porque queda mucho mas claro y de cara al mantenimiento está mucho mejor:

CREATE TABLE PRODUCTO(

 Codproducto number(5) NOT NULL,

 Denoproducto varchar2(20) not null,

 Descripcion varchar2(100),

 PrecioBase number(8,2) not null,

 PorcReposicion number(3),

 UnidadesMinimas number(4) not null,

 UnidadesActuales number(4) not null,

 Codfamilia number(3) not null,

 CONSTRAINT pk\_codproducto PRIMARY KEY(Codproducto),

 CONSTRAINT chk\_preciobase check (PrecioBase >0),

 CONSTRAINT fk\_codfamilia FOREIGN KEY (Codfamilia) REFERENCES FAMILIA(Codfamilia),

 CONSTRAINT chk\_PorcReposicion CHECK (PorcReposicion >0),

 CONSTRAINT chk\_UnidadesMinimas CHECK (UnidadesMinimas>0),

 CONSTRAINT chk\_UnidadesActuales CHECK (UnidadesActuales >0));

--TABLA TIENDA. Contine información básica sobre las tiendas que distribuyen los productos

CREATE TABLE TIENDA(

Codtienda NUMBER(3,0) PRIMARY KEY,

Denotienda VARCHAR2(20) NOT NULL,

Telefono VARCHAR2(11),

CodigoPostal VARCHAR2(5) NOT NULL,

Provincia VARCHAR2(5) NOT NULL

);

--TABLA STOCK. Contiene, para cada tienda, el número disponible de cada producto.

CREATE TABLE STOCK(

Codtienda NUMBER(3,0) CONSTRAINT sto\_tie\_fk REFERENCES TIENDA,

Codproducto NUMBER(5,0) CONSTRAINT sto\_pro\_fk REFERENCES PRODUCTO,

Unidades NUMBER(6,0) NOT NULL CONSTRAINT sto\_uni\_ck CHECK(Unidades > 0),

PRIMARY KEY (Codtienda, Codproducto)

);

**EJERCICIO 2.**

--AÑADIR A LA TABLA STOCK

--Una columna de tipo fecha, llamada FechaUltimaEntrada, que por defecto tome el valor de la fecha actual

ALTER TABLE STOCK

ADD FechaUltimaEntrada DATE DEFAULT SYSDATE;

--Una columna llamada Beneficio, que contendrá el tipo de beneficio qu esa tienda aplica en ese produco.

--Se debe controlar que el valor que almacene sea 1, 2, 3, 4 o 5.

ALTER TABLE STOCK

ADD Beneficio NUMBER(1,0) CONSTRAINT sto\_ben\_ck CHECK (Beneficio IN (1,2,3,4));

Beneficio mejor utilizar CHAR. Porque sabemas que ocupan un numero de caracteres fijos

--EN LA TABLA PRODUCTO

--Eliminar de la tabla producto la columna Descripción.

ALTER TABLE PRODUCTO

DROP COLUMN Descripcion;

--Añadir una columna llamada perecedero, que únicamente acepte los valores: S o N.

ALTER TABLE PRODUCTO

ADD perecedero VARCHAR2(1) CONSTRAINT pro\_per\_ck CHECK (perecedero IN ('S', 'N'));

--Modificar el tamaño de la columna Denoproducto a 50.

ALTER TABLE PRODUCTO

MODIFY Denoproducto VARCHAR2(50);

--EN LA TABLA FAMILIA

--Añadir una columna llamada IVA, que represente el porcentaje de IVA y únicamente pueda contener los valores 21,10,ó 4.

ALTER TABLE FAMILIA

ADD IVA NUMBER(2,0) CONSTRAINT fam\_iva\_ck CHECK (IVA IN (21,10,4));

--EN LA TABLA TIENDA

--La empresa desea restringir el número de tiendas con las que trabaja, de forma que no pueda haber más de una tienda

--en una misma zona (la zona se identifica por el código postal). Definir mediante DDL las restricciones necesarias para que

--se cumpla en el campo correspondiente.

ALTER TABLE TIENDA

ADD CONSTRAINT tie\_cod\_uk UNIQUE (CodigoPostal);

--Renombra la tabla STOCK por PRODXTIENDAS.

ALTER TABLE STOCK

RENAME TO PRODXTIENDAS;

--Elimina la tabla FAMILIA y su contenido si lo tuviera.

ALTER TABLE PRODUCTO

DROP CONSTRAINT pro\_cod\_FK;

DROP TABLE FAMILIA [CASCADE CONSTRAINT];

--Crea un usuario llamado C##INVITADO siguiendo los pasos de la unidad 1 y dale todos los privilegios sobre la tabla PRODUCTO.

DISCONNECT

CONNECT SYSTEM

CREATE USER c##invitado IDENTIFIED BY invitado;

GRANT CONNECT TO c#invitado;

GRANT ALL c##bea.Producto TO c##invitado;

--Retira los permisos de modificar la estructura de la tabla y borrar contenido de la tabla PRODUCTO al usuario anterior.

REVOKE ALL c##bea.PRODUCTO FROM c##invitado;