Una librería almacena los datos de sus libros en una tabla denominada "libros" y en una tabla "ofertas", algunos datos de los libros cuyo precio no supera los $30. Además, controla las inserciones que los empleados realizan sobre "ofertas", almacenando en la tabla "control" el nombre del usuario, la fecha y hora, cada vez que se ingresa un nuevo registro en la tabla "ofertas".

1.- Cree las tablas con las siguientes estructuras:

```
create table libros(
        codigo number(6),
        titulo varchar2(40),
        autor varchar2(30),
        editorial varchar2(20),
        precio number(6,2) );
create table ofertas(
        titulo varchar2(40),
        autor varchar2(30),
        precio number(6,2) );
create table control(
        usuario varchar2(30),
        fecha date );
```

2- Cree un disparador que se dispare cuando se ingrese un nuevo registro en "ofertas"; el trigger debe ingresar en la tabla "control", el nombre del usuario, la fecha y la hora en la cual se realizó un "insert" sobre "ofertas"

3- Ingrese los siguientes registros en "libros"

```
insert into libros values(100,'Uno','Richard Bach','Planeta',25,100);
insert into libros values(103,'El aleph','Borges','Emece',28,0);
insert into libros values(105,'Matematica estas ahi','Paenza','Nuevo siglo',12,50);
insert into libros values(120,'Aprenda PHP','Molina Mario','Nuevo siglo',55,200);
insert into libros values(145,'Alicia en el pais de las maravillas','Carroll','Planeta',35,10);
```

Crear un trigger a nivel de fila que se dispara "antes" que se ejecute un "insert" o un "update" sobre el campo "precio" de la tabla "libros". Se activa solamente si el nuevo precio que se ingresa o se modifica es superior a 50, en caso de serlo, se modifica el valor ingresado redondeándolo a entero

4- Cree un trigger a nivel de fila que se dispare "antes" que se ejecute un "update" sobre la tabla "libros". En el cuerpo del trigger se debe averiguar el campo que ha sido modificado. En caso de modificarse:

- el código, debe rechazarse la modificación con un mensaje de error.
- el "precio", se controla si es mayor o igual a cero, si lo es, debe dejarse el precio anterior y mostrar un mensaje de error.

- el stock, debe controlarse que no se ingrese un número negativo ni superior a 1000, en tal caso, debe rechazarse con un mensaje de error.

5.- Crear un disparador a nivel de sentencia, que se dispare cada vez que se ingrese, actualice o elimine un registro de la tabla "libros". El trigger ingresa en la tabla "control", el nombre del usuario, la fecha y la hora en la cual se realizó la modificación y el tipo de operación que se realizó:

- si se realizó una inserción (insert), se almacena "inserción";
- si se realizó una actualización (update), se almacena "actualización" y
- si se realizó una eliminación (delete) se almacena "borrado"

6.- El gerente permite:

- ingresar o borrar libros de la tabla "libros" únicamente los sábados de 8 a 12 hs.
- actualizar los precios de los libros de lunes a viernes de 8 a 18 hs. y sábados entre la 8 y 12 hs.

Cree un disparador para los tres eventos que controle la hora en que se realizan las operaciones sobre "libros". Si se intenta eliminar, ingresar o actualizar registros de "libros" fuera de los días y horarios permitidos, debe aparecer un mensaje de error. Si la operación de ingreso, borrado o actualización de registros se realiza, se debe almacenar en "control", el nombre del usuario, la fecha y el tipo de operación ejecutada

Un comercio almacena los datos de los artículos que tiene para la venta en una tabla denominada "articulos". En otra tabla denominada "ventas" almacena el código de cada artículo, la cantidad que se vende y la fecha.

1- Cree las tablas con las siguientes estructuras:

```
create table articulos(
        codigo number(4) not null,
        descripcion varchar2(40),
        precio number (6,2),
        stock number(4),
        constraint PK_articulos_codigo primary key (codigo) );
create table ventas(
        codigo number(4),
        cantidad number(4),
        fecha date,
        constraint FK_ventas_articulos foreign key (codigo) references articulos(codigo)
);
```

2- Cree una secuencia llamada "sec_codigoart", estableciendo que comience en 1, sus valores estén entre 1 y 9999 y se incrementen en 1. Antes elimínela por si existe

```
create sequence sec_codigoart start with 1 increment by 1;
```

3- Cree un trigger que coloque el siguiente valor de una secuencia para el código de "articulos" cada vez que se ingrese un nuevo artículo Podemos ingresar un nuevo registro en "articulos" sin incluir el código porque lo ingresará el disparador luego de calcularlo. Si al ingresar un registro en "articulos" incluimos un valor para código, será ignorado y reemplazado por el valor calculado por el disparador

4- Ingrese algunos registros en "articulos" sin incluir el código:

```
insert into articulos (descripcion, precio, stock) values ('cuaderno rayado 24h',4.5,100);
insert into articulos (descripcion, precio, stock) values ('cuaderno liso 12h',3.5,150);
insert into articulos (descripcion, precio, stock) values ('lapices color x6',8.4,60);
```

5- Ingrese algunos registros en "articulos" incluyendo el código:

```
insert into articulos values(160,'regla 20cm.',6.5,40);
insert into articulos values(173,'compas metal',14,35);
insert into articulos values(234,'goma lapiz',0.95,200);
```

6- Cuando se ingresa un registro en "ventas", se debe: - controlar que el código del artículo exista en "articulos" (lo hacemos con la restricción "foreign key" establecida en "ventas");

- controlar que exista stock, lo cual no puede controlarse con una restricción "foreign key" porque el campo "stock" no es clave primaria en la tabla "articulos";

Cree un trigger. Si existe stock, debe disminuirse en "articulos". Cree un trigger a nivel de fila sobre la tabla "ventas" para el evento se inserción. Cada vez que se realiza un

"insert" sobre "ventas", el disparador se ejecuta. El disparador controla que la cantidad que se intenta vender sea menor o igual al stock del artículo y actualiza el campo "stock" de "articulos", restando al valor anterior la cantidad vendida. Si la cantidad supera el stock, debe producirse un error, revertirse la acción y mostrar un mensaje

7- El comercio quiere que se realicen las ventas de lunes a viernes de 8 a 18 hs. Reemplace el trigger creado anteriormente "tr_insertar_ventas" para que No permita que se realicen ventas fuera de los días y horarios especificados y muestre un mensaje de error.

8- El comercio quiere que los registros de la tabla "articulos" puedan ser ingresados, modificados y/o eliminados únicamente los sábados de 8 a 12 hs. Cree un trigger "tr_articulos" que No permita que se realicen inserciones, actualizaciones ni eliminaciones en "articulos" fuera del horario especificado los días sábados, mostrando un mensaje de error. Recuerde que al ingresar un registro en "ventas", se actualiza el "stock" en "articulos"; el trigger debe permitir las actualizaciones del campo "stock" en "articulos" de lunes a viernes de 8 a 18 hs. (horario de ventas)

9. Considerando las siguientes dos tablas:

empleado: nss(numérico), nombre (10 caracteres), salario (numérico), num_dep (10 caracteres) departamento: num_dep (10 caracteres), nombre (10 caracteres), presupuesto (numérico)

- Diseñar un disparador que al insertar un nuevo empleado, automáticamente quede actualizado el presupuesto total del departamento al que el empleado pertenece, añadiéndole el salario asignado al nuevo empleado.
- Diseñar un disparador que al modificar el salario de un empleado, automáticamente quede actualizado el presupuesto total del departamento al que el empleado pertenece, en función del nuevo salario asignado al empleado.