Relación de Ejercicios Triggers

1.- EMPRESA

Una empresa almacena los datos de sus empleados en una tabla denominada "empleados" y en otra tabla llamada "secciones", el código de la sección y el sueldo máximo de cada una de ellas.

1. Elimine las tablas si existen
2. Cree las tablas
3. Ingrese algunos registros en ambas tablas
4. Cree un disparador para que se ejecute cada vez que una instrucción "insert" ingrese datos en "empleados"; el mismo debe verificar que el sueldo del empleado no sea mayor al sueldo máximo establecido para la sección, si lo es, debe mostrar un mensaje indicando tal situación y deshacer la transacción.
5. Ingrese un nuevo registro en "empleados" cuyo sueldo sea menor o igual al establecido para la sección.
6. Verifique que el disparador se ejecutó consultando la tabla "empleados": select *from empleados
7. Intente ingresar un nuevo registro en "empleados" cuyo sueldo sea mayor al establecido para la sección. El disparador se ejecutó mostrando un mensaje y la transacción se deshizo.
8. Intente ingresar un empleado con código de sección inexistente. Aparece un mensaje de error porque se viola la restricción "foreign key"; el trigger no llegó a ejecutarse.

```
create table secciones(
  codigo int identity,
  nombre varchar(30),
  sueldomaximo decimal(8,2),
  constraint PK_secciones primary key(codigo)
 );

 create table empleados(
  documento char(8) not null,
  nombre varchar(30) not null,
  domicilio varchar(30),
  codigoseccion int not null,
  sueldo decimal(8,2),
  constraint PK_empleados primary key(documento),
  constraint FK_empelados_seccion
   foreign key (codigoseccion) references secciones(codigo)
 );
 insert into secciones values('Administracion',1500);
 insert into secciones values('Sistemas',2000);
 insert into secciones values('Secretaria',1000);

 insert into empleados values('22222222','Ana Acosta','Avellaneda 88',1,1100);
 insert into empleados values('23333333','Bernardo Bustos','Bulnes 345',1,1200);
 insert into empleados values('24444444','Carlos Caseres','Colon 674',2,1800);
 insert into empleados values('25555555','Diana Duarte','Colon 873',3,1000);
```

Relación de Ejercicios Triggers

2.- Club de barrio.

Un club de barrio almacena los datos de sus socios en una tabla llamada "socios", los datos de las inscripciones en una tabla denominada "inscriptos" y en una tabla "morosos" almacena el documento de los socios inscriptos que deben matrícula.

1. Cree un disparador de inserción que no permita ingresar inscripciones si el socio es moroso.
2. Cree otro "insert trigger" para "inscriptos" que ingrese el socio en la tabla "morosos" si no paga la matrícula (si se ingresa 'n' para el campo "matricula"). Recuerde que podemos crear varios triggers para un mismo evento sobre una misma tabla
3. Creamos un disparador sobre la tabla "socios" para que no permita ingresar nuevos socios. El mismo debe mostrar un mensaje al dispararse y deshacer la transacción

Cree las tablas, con las siguientes estructuras:

```
create table socios(
 documento char(8) not null,
 nombre varchar(30),
 domicilio varchar(30),
 constraint PK_socios primary key (documento)
);

create table inscriptos(
 documento char(8) not null,
 deporte varchar(30) not null,
 matricula char(1),
 constraint CK_inscriptos check (matricula in ('s','n')),
 constraint PK_inscriptos primary key (documento,deporte),
 constraint FK_inscriptos_documento foreign key(documento)
  references socios (documento)
);

create table morosos(
 documento char(8) not null
);
insert into socios values ('22222222','Ana Acosta','Avellaneda 800');
insert into socios values ('23333333','Bernardo Bustos','Bulnes 234');
insert into socios values ('24444444','Carlos Caseros','Colon 321');
insert into socios values ('25555555','Mariana Morales','Maipu 483');
insert into inscriptos values ('22222222','tenis','s');
insert into inscriptos values ('22222222','natacion','n');
insert into inscriptos values ('23333333','tenis','n');
insert into inscriptos values ('24444444','tenis','s');
insert into inscriptos values ('24444444','futbol','s');
insert into morosos values ('22222222');
insert into morosos values ('23333333');
```

Relación de Ejercicios Triggers

3.- EMPLEADOS

```
CREATE TABLE empleados
        (dni char(4) PRIMARY KEY,
        nomemp varchar2(15),
        mgr char(4),
        salario integer DEFAULT 1000,
        usuario varchar2(15),
        fecha date );

ALTER TABLE EMPLEADOS ADD(FOREIGN KEY (mgr) REFERENCES empleados (DNI) );
```

3.1.- Crear un trigger sobre la tabla EMPLEADOS para que no se permita que un empleado sea jefe (MGR) de más de cinco empleados.

3.2.- Crear un trigger para impedir que se aumente el salario de un empleado en más de un 20%.

- Es necesario comparar los valores :old.salario y :new.salario cada vez que se modifica el atributo salario (BEFORE UPDATE).

3.3.- Crear una tabla empleados_baja con la siguiente estructura:

```
CREATE TABLE empleados_baja
(dni char(4) PRIMARY KEY,
nomemp varchar2(15),
mng char(4),
salario integer,
usuario varchar2(15),
fecha date );
```

- Crear un trigger que inserte una fila en la tabla empleados_baja cuando se borre una fila en la tabla empleados.
- Los datos que se insertan son los correspondientes al empleado que se da de baja en la tabla empleados, salvo en las columnas usuario y fecha se grabarán las variables del sistema USER y SYSDATE que almacenan el usuario y fecha actual.
- El comando que dispara el trigger es AFTER DELETE.

3.4.- Modificar la tabla empleados añadiendo el campo departamento (integer) (ALTER TABLE …).

- Crear un trigger para impedir que el salario total por departamento (suma de los salarios de los empleados por departamento) sea superior a 10.000.
- Ayuda:
  - – Será necesario distinguir si se trata de una modificación o de una inserción.
  - – Cuando se trate de una inserción (IF INSERTING...) se comprobará que el salario del empleado a insertar (:NEW.salario) más el salario total del departamento al que pertenece dicho empleado no es superior a 10.0000.
  - – Cuando se trate de una modificación (IF UPDATING...), al salario total del departamento se le sumará el :NEW.salario y se le restará el :OLD.salario.

3.5.- Crear un trigger para impedir que un empleado y su jefe pertenezcan a departamentos distintos.