Trabajo Practico sobre Triggers

create database triggers

go

use triggers

**Ejercicio A**

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.

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('11111111','Carla Gomez','Rivadavia 1234',1,1100);

insert into empleados values('23333333','Juan Garcia','Bulnes 238',1,1200);

insert into empleados values('24444444','Pedro Perez','Cordoba 765',2,1800);

insert into empleados values('25555555','Maria Duarte','Belgrano 345',3,1000);

1- Cree un disparador para que se ejecute cada vez que una instrucción "insert" ingrese datos "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.

2- Ingrese un nuevo registro en "empleados" cuyo sueldo sea menor o igual al establecido para la sección.

3- Verifique que el disparador se ejecutó consultando la tabla "empleados":

4- Intente ingresar un nuevo registro en "empleados" cuyo sueldo sea mayor al establecido para la sección.

5- Verifique que el registro no se agregó en "empleados"

6- Intente ingresar un empleado con código de sección inexistente.

**Ejercicio B**

Un comercio que vende artículos de informática almacena los datos de sus artículos en una tabla denominada "articulos".

create table articulos(

codigo int identity,

tipo varchar(30),

descripcion varchar(40),

precio decimal(8,2),

stock int,

constraint PK\_articulos primary key (codigo)

);

insert into articulos values ('impresora','Epson Stylus C45',400,100);

insert into articulos values ('impresora','Brother',500,200);

insert into articulos values ('impresora','Canon Color 600',400,0);

insert into articulos values ('monitor','Samsung 23',900,0);

insert into articulos values ('monitor','Samsung 19',1200,0);

insert into articulos values ('monitor','xxx 15',1500,0);

insert into articulos values ('monitor','yyy 17',1600,0);

insert into articulos values ('monitor','zzz 15',1300,0);

1- Cree un disparador para controlar que no se elimine un artículo si hay stock. El disparador se activará cada vez que se ejecuta un "delete" sobre "articulos", controlando el stock, si se está eliminando un artículo cuyo stock sea mayor a 0, el disparador debe retornar un mensaje de error y deshacer la transacción.

2- Solicite la eliminación de un artículo que no tenga stock.

3- Intente eliminar un artículo para el cual haya stock.

4- Solicite la eliminación de varios artículos que no tengan stock.

5- Intente eliminar varios artículos, algunos con stock y otros sin stock.

6- Cree un trigger para evitar que se elimine más de 1 artículo.

7- Solicite la eliminación de 1 artículo para el cual no haya stock.

8- Solicite la eliminación de 1 artículo que tenga stock.

9- Solicite la eliminación de 2 artículos para los cuales no haya stock.

10- Solicite la eliminación de 2 artículos para los que haya stock.

**Ejercicio C**

Un club almacena los datos de sus socios en una tabla denominada "socios", las inscripciones en "inscriptos" y en otra tabla "morosos" guarda los documentos de los socios que deben matrículas.

create table socios(

documento char(8) not null,

nombre varchar(30),

domicilio varchar(30),

constraint PK\_socios primary key(documento)

);

create table inscriptos(

numero int identity,

documento char(8) not null,

deporte varchar(20),

matricula char(1),

constraint FK\_inscriptos\_documento

foreign key (documento)

references socios(documento),

constraint CK\_inscriptos\_matricula check (matricula in ('s','n')),

constraint PK\_inscriptos primary key(documento,deporte)

);

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 345');

insert into socios values('24444444','Carlos Caseros','Colon 382');

insert into socios values('25555555','Mariana Morales','Maipu 234');

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','futbol','s');

insert into inscriptos values('24444444','natacion','s');

insert into morosos values('22222222');

insert into morosos values('23333333');

1- Cree un disparador para la tabla "inscriptos" que se active ante una sentencia "update" y no permita actualizar más de un registro.

2- Cree otro disparador para la tabla "inscriptos" que se active ante una sentencia "update". Si se actualiza el pago de la matrícula a 's', el socio debe eliminarse de la tabla "morosos"; no debe permitir modificar a 'n' una matrícula paga.

3- Actualice cualquier campo (diferente de "matricula") de un registro de la tabla "inscriptos".

4- Actualice cualquier campo (diferente de "matricula") de varios registros de la tabla "inscriptos".

5- Actualice el campo "matricula" a 's' de un inscripto que deba la matrícula.

6- Verifique que el campo se actualizó y que el socio ya no está en "morosos":

7-Actualice el campo "matricula" a 'n' de un inscripto que tenga la matrícula paga.