**Trabajo 1**

**Bolkar Leimar Ocampo Montoya**

**José David Rueda Torres**

**Sergio Alejandro Bermúdez Gómez**

**Universidad Nacional de Colombia**

**Sede Medellín**

**Bases de datos 2**

**Francisco Javier Moreno**

**22 de septiembre de 2022**



**Trigger inserción en Cooperativa (3%)**

Se realizó un Trigger de before insert, para la fila que se está ingresando se hace que su *c\_acumulado* sea 0 antes de ingresar a la tabla.

**Trigger inserción en Socio (3%)**

Se realizó un Trigger de before insert, para la fila que se está ingresando se hace que su *s\_acumulado* sea 0 antes de ingresar a la tabla.

**Trigger inserción en Coopexsocio** **(3%)**

Se realizó un Trigger de before insert, para la fila que se está ingresando se hace que su *sc\_acumulado* sea 0 antes de ingresar a la tabla.

**Trigger actualización tabla cooperativa** **(25%)**

Este trigger es disparado por cualquier actualización (evento update) que se realice sobre la columna *c\_acumulado*de la tabla *cooperativa*, su momento es *before* y su tipo *for each row*. Haciendo uso de un cursor (*sociosCoop*), se guarda la información extraída de las tablas *socio* y *coopexsocio* sobre los socios vinculados a la cooperativa modificada y sus respectivos acumulados. Por medio de una consulta se obtiene la cantidad de socios pertenecientes a dicha cooperativa (*cantSocios*).

Luego, para calcular el incremento que tendrá la fila en cuestión es su columna *c\_acumulado* se realiza la resta entre el nuevo valor y el anterior, esta diferencia se guarda en la variable *incremento*. En este punto se verifica que dicho incremento sea positivo, pues no se permiten incrementos negativos, y en caso de ser negativo la columna no cambiará su valor, rechazando la actualización. Además, se lanza la excepción *incrementoNegativo* la cual le comunica al usuario que solo se permiten incrementos mayores a cero.

Por último, en el caso de que el incremento sea positivo, se recorre el cursor mediante un ciclo *for* para realizar la modificación de los respectivos acumulados de los socios, tanto en la tabla *socio* como en *coopexsocio,* pertenecientes a la cooperativa que se está actualizando.

Otra excepción adicional que se tiene en cuenta es la de ZERO\_DIVIDE en caso de que la cooperativa no tenga socios vinculados.

**Trigger borrado tabla cooperativa** **(15%)**

Es un Trigger de before delete.

Se realiza un cursor, el cual obtiene el socio y el acumulado de *coopexsocio*, donde la cooperativa es la que se quiere borrar.

Se recorre el cursor, para cada fila se obtiene el socio asociado a *coopexsocio* y a su *s\_acumulado* actual se le resta el *cs\_acumulado* que tiene en *coopexsocio*.

Luego se eliminan los registros de *coopexsocio* donde la cooperativa que se quiere eliminar está asociada.

**Trigger borrado tabla socio** **(10%)**

Es un Trigger de before delete.

Se elimina mediante una consulta los valores en *coopexsocio* que están asociados al socio que se desea eliminar.

**Programa mostrar cooperativa (20%)**

Para este primer programa, donde se recibe el código de una cooperativa como parámetro, se hace uso de un cursor (*curSocios*) en el cual se guarda la información presente en las filas de las tablas *socio* y *coopexsocio* donde la cooperativa sea la pasada como parámetro. Haciendo uso del código de la cooperativa se obtiene, por medio de una consulta, el nombre y el acumulado de dicha cooperativa, al igual que la cantidad de socios presentes en esta. Esta información es la que primero se imprime, excepto la cantidad de socios mencionada anteriormente.

Luego, recorriendo el cursor por medio de un ciclo *for*, se obtiene e imprime por cada fila el nombre de cada socio con su respectivo acumulado correspondiente al de la cooperativa a la que pertenece. Por último, se imprime la cantidad de socios vinculados a la cooperativa sobre la cual se realiza la consulta.

Por el lado de las excepciones se controla el caso en el que se ingrese el código de una cooperativa que no existe, pues esto genera un NO\_DATA\_FOUND.

**Programa mostrar socio (21%)**

Se utilizaron las siguientes estructuras de datos:

Arreglo llamado *arreglo\_contiene*, el cual guarda las filas de *coopexsocio* que están asignadas al socio pasado por parámetro.

Arreglo llamado *total\_cooperativas*, el cual guarda el total de cooperativas presentes en la base de datos.

Para mostrar las cooperativas en las que está asociado el socio, se recorre *arreglo\_contiene* y de cada posición del arreglo se obtiene el nombre de la cooperativa y el acumulado del socio en dicha cooperativa.

Para mostrar las cooperativas en las que no está asociado el socio, se recorre *total\_cooperativas* y para cada cooperativa existente en la base de datos se busca en *arreglo\_contiene*. Si no se encuentra es porque el socio pasado por parámetro no está asociado a la cooperativa, y se obtiene el nombre de la cooperativa. Si se encuentra simplemente se omite esa cooperativa

Excepciones:

Se captura la excepción de NO\_DATA\_FOUND cuando el socio pasado por parámetro no se encuentra en la base de datos