Base de Datos 2 - Obligatorio 2016

- El obligatorio será realizado en forma individual.

- La entrega se hará de manera electrónica a fer200417@gmail.com.

- Se deberá entregar un archivo .zip el cual debe contener los 3 archivos mencionados posteriormente con asunto Base de Datos 2 – Obligatorio 2.
En el mail poner nombre apellido y cédula de identidad.

- El plazo para la entrega es hasta el **Domingo 03/07/2016** hasta las 23:59 hs. Una vez culminado el plazo estipulado, **no** se aceptarán más obligatorios por parte del docente.

Video Club

Se pide:

PARTE 1. Uso de triggers

Objetivo: Experimentar en el uso de **triggers** como mecanismo de control del cumplimiento de restricciones de integridad.

1. Totalidades.

Un mecanismo adecuado para controlar el cumplimiento de las restricciones de totalidad existentes en el MER son los **triggers**. Estos permiten disparar la ejecución de procedimientos ante eventos sobre tablas (por ejemplo inserciones o borrados). Combinando estos procedimientos con el uso de **tablas de control** podemos detectar y almacenar aquellas tuplas que violan las restricciones de totalidad planteadas.
Tomemos por ejemplo la restricción de totalidad existente en la relación idiomas_de_pelicula entre p*eliculas* e i*diomas*, la cual exige que para toda película se conozca al menos un idioma. Una estrategia para detectar aquellas tuplas que no cumplen la restricción es la siguiente:

i) Definir un trigger que se ejecutará cada vez que se **inserte** una tupla en la tabla *peliculas*. Este trigger registrará en una tabla de control que la inserción se realizó.

ii) Definir un trigger que se ejecutará cada vez que se **inserte** una tupla en la tabla i*diomas_de_peliculas*. Este trigger borrará la tupla correspondiente a la película de la tabla de control.

De esta forma podemos asegurar que la tabla de control sólo contiene aquellas películas que violan la restricción de totalidad
A continuación especificamos en detalle los triggers requeridos para resolver el problema.

Nombre Trigger: tot_peliculas
Se ejecuta Cuando se inserta una tupla en peliculas
Acción Inserta una tupla en la tabla: **inconsist_tot_pel (id_alerta,id_pelicula)**

Nombre Trigger: tot_peliculas_borrar
Se ejecuta Cuando se inserta una tupla en idiomas_de_peliculas
Acción Borra la tupla correspondiente de la tabla: **inconsist_tot_pel(id_alerta,id_pelicula)**
Importante: - El atributo **id_alerta** debe ser de tipo **SERIAL**.
- El atributo **id_pelicula** debe ser del mismo tipo que su correspondiente en la tabla *idiomas_de_peliculas* (integer)
- Utilizar los nombre de los triggers mencionados anteriormente.
Se pide: Implementar mediante triggers el control de la restricción de totalidad entre las entidades Películas e Idiomas, tal como se sugiere.

Se debe entregar: un archivo llamado parte1_1.sql que contenga la creación de las tablas y la creación de los triggers necesarios para esta parte

2. Restricciones No Estructurales.

Los **triggers** también pueden ser usados para asegurar el cumplimiento de restricciones no estructurales (RNE), por ejemplo impidiendo la inserción de tuplas que las violen. Teniendo en cuenta las restricción podemos imaginarnos que el procedimiento de creación de una sucursal y asignación de su encargado podría ser el siguiente:

1- Se inserta en la tabla *sucursales* la sucursal s, con el atributo encargado en NULL
2- Se insertan en la tabla *personal* los empleados que trabajan en la sucursal s.
3- Se actualiza el atributo encargado de la tupla de la tabla *sucursales* que representa a la sucursal s con el identificador de alguno de los empleados de s.

Para controlar que el empleado que se indica como encargado efectivamente trabaje en dicha sucursal se puede implementar un trigger como se indica a continuación:

Nombre trigger: r3_encargados
Se ejecuta cuando se inserta/actualiza una tupla en *sucursales*.
Acción Sea *<s,p,-,-,-,->* la tupla que se desea insertar/actualizar en la tabla *Sucursales*
Si p no es NULL se controla que:
    Exista una tupla en *personal* de la forma *<p,s,-,-,-,-,-,-,true,-,->*
    Si no se cumple:
        No se permite la inserción/actualización
        Se despliega el mensaje que informe el error, mencionando la sucursal y el identificador de personal.
        Se inserta una tupla en la tabla: **inconsist_r3 (id_sucursal,id_personal)**
Si p es NULL no hace nada
Importante: - Los atributos de la tabla **inconsist_r3** deben ser del mismo tipo a sus correspondientes en otras tablas.
- Utilizar el nombre del trigger dado anteriormente.

Se debe entregar: un archivo llamado parte1_2.sql que contenga la creación de las tablas y la creación de los triggers necesarios para esta parte.

PARTE 2. Uso de procedimientos almacenados

Objetivo: Experimentar en el desarrollo de **procedimientos almacenados** para resolver problemas sencillos directamente en la base de datos.

Se pide: Implementar procedimientos almacenados que permitan calcular el estado de cuenta de los clientes del Video Club.

Nombre Función: balance_cliente
Parámetros: Entrada: id_cliente (integer)
fecha (timestamp)
Salida: saldo (numeric)
Acción Dado un cliente (id_cliente) y una fecha (fecha) calcula el saldo de cuenta del cliente a esa fecha.
El estado de cuenta de un cliente se compone de:
1- La suma de los costos de alquiler de todos los alquileres realizados ANTES de la fecha de cálculo.
2- Un recargo de $5 por cada dia de atraso[1] de los alquileres considerados en el punto 1.
3- Si el atraso es mayor al triple de la duracion_alquiler se le debe cobrar el costo de reemplazo de la película.
4 - Descontar todos los pagos que el cliente haya realizado ANTES de la fecha de cálculo.
Nombre Función: balance_clientes
Parámetros: Entrada: fecha (timestamp)

Acción Recorre la tabla *clientes* y para cada uno de ellos invoca la función *balance_cliente.*
Por cada invocación inserta una tupla en la tabla *balances* con el cliente (id_cliente), la fecha de cálculo (fecha) y el saldo calculado.
Importante:
- La tabla **balance** debe tener los atributos id_cliente, fecha_calculo, saldo
- Utilizar los nombres de funciones dados anteriormente.

Se debe entregar: un archivo llamado parte2.sql que contenga la creación de las tablas y la creación de los procedimientos almacenados necesarios para resolver esta parte.

1 Recordar que para cada película se conoce el costo_alquiler que corresponde a un plazo igual a duracion_alquiler. Si la película no fue devuelta transcurrido ese plazo (en dias) se considera que está atrasado.