TRABAJO PRÁCTICO ESPECIAL - BASES DE DATOS 2019.
MAZZA, ELOY - SEGURA, EMANUEL
Grupo 17
Fecha entrega: 2019-06-06

WAREHOUSE MANAGEMENT SYSTEM

Introducción

Se propuso implementar la Base de Datos y dos servicios que realizan consultas sobre la misma para la empresa "**WMS TANDIL".** Para ello se hizo uso de el motor de bases de datos de **Postreges SQL** y el lenguaje de backend **PHP** para servir el sitio web.
Se crearon los tres archivos solicitados: **GR17_Creacion.sql** (con algunos campos agregados para cumplir con los requerimientos) , **GR17_Cambios.sql** en donde se implementaron las restricciones, funciones, y vistas requeridas, y el archivo **GR17_Borrado.sql** el cual provee las sentencias necesarias para realizar la eliminación de las tablas, vistas y funciones.
Sumado a lo anterior se agrega el siguiente repositorio de github: https://github.com/eloymazza/tpeBases.

En él se haya el código del sitio, el cual solo se debe instalar y correr en XAMPP. Solo debe proporcionar su password para acceder a la Base de Datos.

1. Interpretación e implementación de los requerimientos

a) Para la implementación de la restricción de que las posiciones deben tener una unica posicion global,
se creo una restriccion de **"UNIQUE"** en la tabla **GR17_POSICIONES**. Este campo es un entero que contiene un número con el formato EEE_FFF_PPP(001001001) , donde EEE es el numero de estanteria, FFF es el número de fila, y PPP es el numero de posición. Este número de 9 dígitos no se puede repetir. De esta forma se puede identificar donde esta la posición sin tener que consultar la tabla fila o estanteria. No se crearon restricciones para controlar el cumplimiento de este formato, solo que a la hora de insertar una posicion o actualizarla, se tiene en cuenta el mismo.

b) Se agrego el campo alto_mts en la tabla fila, de tipo numeric(4,2), para dar soporte al requerimiento no obligatorio "Cada fila tiene un alto".

c) Para tener registro de cuáles filas están ocupadas o no, se optó por mantener actualizado el campo ESTADO de la tabla **GR17_ALQUILER_POSICIONES** ante cada inserción en las tablas **GR17_MOV_ENTRADA, GR17_MOV_INTERNO,** y **GR17_MOV_SALIDA.** La inserción de un Movimiento de Entrada activa el trigger **TR_GR17_ACTUALIZAR_ESTADO_MOV_ENTRADA,** el cual ejecuta un procedimiento que coloca en "true" el estado relativo a esa posicion.
La insercion de un Movimiento interno activa el trigger **TR_GR17_ACTUALIZAR_ESTADO_MOV_INTERNO,** el cual ejecuta un procedimiento que coloca "false" en el estado de la posicion desde la cual se esta moviendo el pallet y un "true" en el estado de la posicion a la cual se esta moviendo el pallet.
La insercion de un Movimiento de salida activa el trigger **TR_GR17_ACTUALIZAR_ESTADO_MOV_SALIDA,** el cual ejecuta un procedimiento que coloca "false" en el estado de la posición indicada en el movimiento de salida.

d) Para la restriccion "**También se registran los movimientos de egreso de mercadería, que *siempre deben hacer referencia a un movimiento de entrada"*** se agregó un campo nuevo a la tabla llamado "**id_movimiento_entrada"**, el cual no puede ser nulo y

debe referenciar (esto se checkea mediante una foreing key hacia la tabla GR17_MOV_ENTRADA) a un movimiento de entrada.

e) Para la restricción "***también deben hacer referencia a un movimiento de entrada* o a otro movimiento de cambio",** se agregaron dos campos: **"id_movimiento_entrada"** y "**id_movimiento_interno".** Estos campos son FK y referencian a las tablas **GR17_MOV_ENTRADA** Y **GR17_MOV_ENTRADA** respectivamente. Ademas se agregó en el archivo **Cambios.sql** el checkeo **CHK_GR17_MOVIMIENTO_INTERNO_REFERENCIAS,** el cual checkea que al menos uno sea nulo y el otro no nulo. Y que no puedan ser ambos no nulos o ambos nulos al mismo tiempo, ya que un movimiento interno solo debe referenciar a un movimiento de entrada o a un movimiento interno para la misma tupla.

2.- Ajustes del esquema

a) Se crearon en el archivo **Creación.sql** los datos requeridos para tener una tabla de ejemplo. Todos las inserciones cumplen con las restricciones implementadas para evitar la generación de errores.

b) Como se indicó en el punto anterior, se agregaron campos en las tablas adecuadas para cumplir con las restricciones solicitadas.

3.- Restricciones

a) **Fechas de alquileres consistentes:** Se implementó el checkeo de fila **CHK_GR17_ALQUILER_CONSISTENTE_FECHA**, sobre la tabla **GR17_ALQUILER,** el cual requiere que el campo fecha_desde sea menos que el campo fecha_hasta.

b) **El peso de los pallets de una fila no debe superar al máximo de la fila.** Esta restricción es un **ASSERTION** ya que su control implica más de una tabla. Fue implementado mediante triggers que llaman a la funcion **TRFN_GR17_VERIFICARPESO().** Esta restricción se controla cada vez que se realiza un movimiento de entrada, un movimiento interno, o se cambia el PESO MAXIMO soportado por una fila.

c) Valores posibles del campo "tipo" en la tabla Posicion: Esta restricción es una restriccion de campo (**tipo**) sobre la tabla **GR17_POSICION.** Se implemento en el checkeo llamado **CHK_GR17_POSICION_TIPO**.

3.- Servicios

a) Para una fecha determinada dar la lista de las posiciones libres: Se interpretó que las posiciones libres son tanto las que no estan ocupadas para la fecha dada como todas las que aparecen en la tabla **GR17_POSICION** pero que a la vez no estan en la tabla **GR17_ALQUILER_POSICIONES.** Se implemento mediante la funcion **FN_GR17_GETPOSICIONESLIBRES(date).** La idea de la funcion es un UNION entre dos selects, el primero las posiciones de **GR17_ALQUILER_POSICIONES** que no estan siendo alquiladas en la fecha indicada junto con un segundo select que retorna las posiciones que aparecen en la tabla **GR17_POSICION** pero no en **GR17_ALQUILER_POSICIONES**.
El retorno de esta funcion es un SETOF **GR17_POSICION.**
La fecha se pasa desde el sitio web mediante un input date. PHP ejecuta la funcion getPosicionesLibres(date) y muestra el resultado en una tabla.

b) Dar la lista de los clientes que en una cierta cantidad de días (configurable) se les debe avisar que se vence su alquiler: Esta funcionalidad se desarrolló haciendo una resta entre la fecha_hasta de la tabla **GR17_ALQUILER** un integer (que representa una cantidad de dias) pasado como parámetro desde php. Si dicha operacion retorna la fecha actual (CURRENT_DATE) entonces se busca el ID cliente de ese alquiler y se retornan susdatos.
La cantidad de dias se pasa como parametro desde el sitio web mediante un input number.
PHP ejecuta la funcion getClientesANotificar() y muestra el resultado en una tabla.
Para poder probar esta función se recomienda insertar un cliente cuya fecha_hasta de alquiler sea dentro de x dias y pasar esos x dias como parámetro en el form del sitio web.
Para realizar esto se encuentra un INSERT comentado en el archivo Cambios.slq (linea 214)

4- Definición de vistas.

a) Ver Posiciones libres y días que quedan de alquiler para las posiciones ocupadas: Se implementó mediante la vista **GR17_ESTADO_POSICIONES.**

Esta vista posee 3 selects unidos mediante dos **UNION.**

El primer select obtiene todas las posiciones que estan en falso en **G17_ALQUILER_POSICIONES**, ya que como se mantiene el estado segun se comento anteriormente, estas posiciones estarían libres para la fecha actual. Se

setea NULL en el campo dias_restantes_alquiler ya que al no estar alquiladas en ese momento no hay dias restantes de alquiler.

El segundo select retorna las posiciones que estan en la tabla **POSICION** pero que no aparecen en la tabla **ALQUILER_POSICIONES** y por esto no estan ocupadas y no se puede calcular los dias restantes de alquiler, entonces el campo queda en null.

El ultimo select trae las posiciones que se encuentran alquiladas en el momento actual, es decir que su estado en **ALQUILER_POSICIONES** es 'true' , y luego se calculan los dias restates restando el campo **fecha_hasta** de la fecha actual (**CURRENT_DATE**).

b) Realizar una vista que liste los 10 clientes que más dinero han invertido en el último año (tomar el momento en el que se ejecuta la consulta hacia atrás): Esto fue implementado mediante la vista **GR17_CLIENTES_MAS_VALIOSOS.** Esta vista se ayuda de la funcion getCantDias(date,date), la cual retorna los dias en los cuales cada cliente pago su alquiler en el ultimo año. Es decir si su alquiler es de 5$ y alquilo 10 dias, el cliente invirtio 50$. Este valor se almacena en la columna "IMPORTE". Luego se ordenan los resultados ordenandolos por esta nueva columna IMPORTE.
La funcion getCantDias() hace algunas validaciones extra como por ejemplo retornar 0 cuando la fecha_desde supera la fecha actual (Por lo cual el cliente aun no invirtio) o cuando la fecha_hasta es anterior a CURRENT_DATE - 1 YEAR (un año atras), que tambien indicaria que el cliente no invirtio en este año.

FIN DEL INFORME