Práctica 2

Sistemas Informáticos

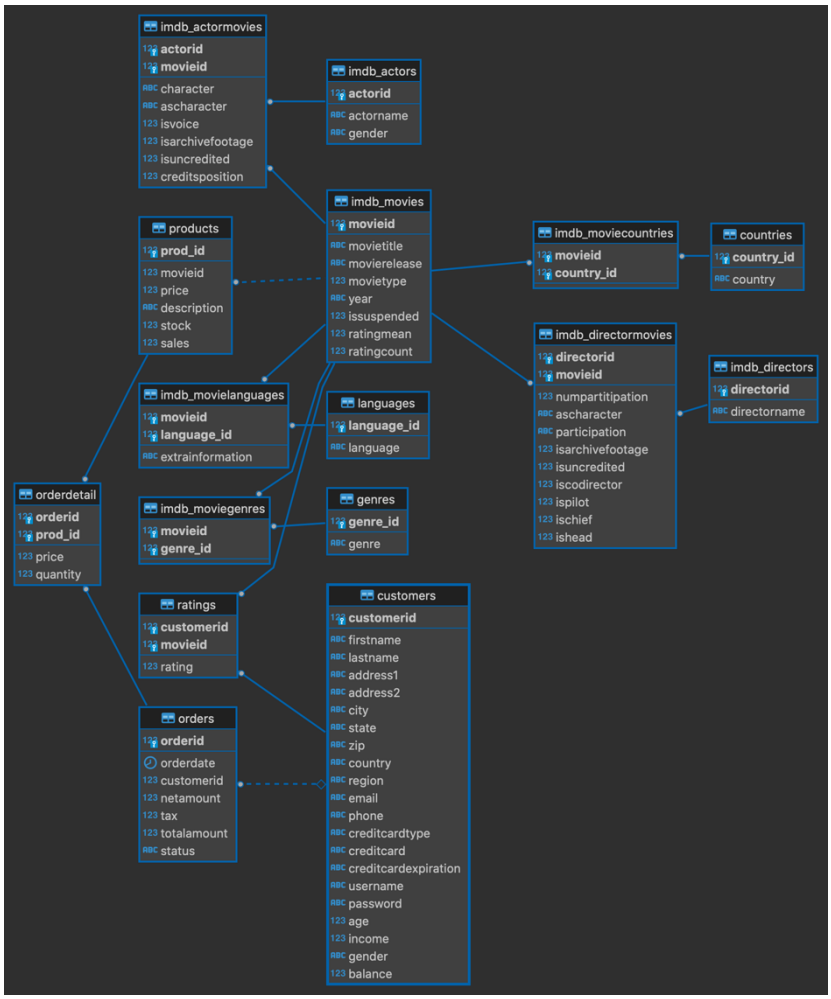
Giancarlo Lazo
Pablo Izaguirre
Grupo 1321

Contenido

1. Introducción

Nos hemos centrado en la implementación de la base de datos (en PostgreSQL y con ayuda de phppgadmin) que dará soporte al backend diseñado en la práctica 1 y donde se guardarán los datos de productos, usuarios y compras.

2. Diagrama de E-R



3. Consultas, triggers y funciones postgresql

Ficheros:

- actualiza.sql: fichero que actualiza la base de datos proporcionada.

Explicado en el apartado mejoras

- setPrice.sql: fichero que completa la columna “price” de la tabla orderdetail.

Se obtiene el precio del producto correspondente a cada entrada de la tabla orderdetail y se calcula el precio en el momento de compra teniendo en cuenta que el precio ha ido aumentando un 2% anual. La formula para calcular ese valor es: precio = precio_actual / (1.02^(año_actual – año_compra))

- setOrderAmount.sql: fichero que completa las columnas netamount y totalamount.

Se han actualizado netamount y totalamount. Netamount crea una tabla auxiliar S, para sacar la suma de los precios totales agrupados por el orderid. Después, con la tabla actualizada, igualas el totalamount (netamount más impuestos) con otra tabla auxiliar T.

- getTopSales.sql: función que recibe dos argumentos, para recibir las peliculas que tuvieron las mejores ventas por año.

Se han usado dos tablas una que se llama AUX, con todas las id de las peliculas, el año que le corresponde, venta por año, y su prod_id. AUX2 es igual a AUX, pero sin prod_id.

MaxSalesPerYearTable usa AUX2 teniendo como resultado el año y la venta máxima ese año.

Se le hace un inner join con AUX para recibir el movieid de las ventas máximas. Se intento devolver la salida con los out que decía la práctica, pero al usar select into y las variables out, solo retornaba la fila de más arriba y no todas las peliculas entre los dos años.

- getTopActors.sql: recibe un parámetro “género” y devuelve los actores o actrices que más veces han actuado en dicho género.

Esta función primero realiza una query para obtener los actores que más han actuado en el género correspondiente junto con el número de películas que han realizado en dicho género. Luego hacemos otra query utilizando dichos actores y encontrando el año de su debut en el género y la película o películas en las que participó en ese año. Hacemos uso de natural join que hace join usando las columnas con el mismo nombre, lo cual resulta muy cómodo para programar.

- updOrders.sql: actualiza la tabla de orders cuando se modifica el carrito, es decir, si se modifica la tabla orderdetail. Este trigger modifica los campos orderdate, introduciendo la fecha del momento del cambio, y las columnas que indican los distintos precios totales del pedido correspondiente.

- updRatings.sql: actualiza la información de la tabla 'imdb_movies'
cuando se añade, actualiza o elimina una valoración, modificando los campos ratingmean y ratingcount.

Cuando se borra una valoración hacemos uso de la fórmula: ratingmean = (ratingmean * ratingcount - old.rating) / (ratingcount - 1).

Cuando se inserta una nueva valoración, actualizamos la media con: ratingmean = (ratingmean*ratingcount + new.rating)/(ratingcount + 1)

Cuando se actualiza una valoración, utilizamos: ratingmean = (ratingmean*ratingcount - old.rating + new.rating)/ratingcount

- updInventoryAndCustomer.sql: actualiza la tabla inventory (products en nuestro caso) y descuenta el balance de un costumer id con el precio de su compra correspondiente.

Se ha declarado el rec como la union del old (orders), products y orderdetail para luego, con esa tabla actualizar las tablas products y customer. A productos se modifica su stock y sales siempre que prod_id sea igual a rec.prod_id, y a customer se le resta el balance de la compra siempre que su id sea igual al customer id de la tabla rec donde cada customer id tiene un producto que ha comprado dicho customer.

4. Mejoras

Se ha añadido la tabla ratings, como lo pedía la práctica, junto con los atributos customerid, movieid y rating; su clave principal es customerid junto a movieid.

Se han agregado las columnas en imdb_movie, como ratingmean y ratingcount y también se cambió el atributo password en customers para que soporte 96 caracteres.

En customers, el campo email es único.

La tabla orderdetail no incluía clave primaria, por lo que asignamos como clave primaria a la pareja (orderid, prod_id), que identifican de forma única un producto de un pedido.

Había dos tablas antes del actualiza (inventory y products) que las terminamos uniendo debido a que ambas se referían a lo mismo, un conjunto de productos con su precio, stock… por lo que es más sencillo tenerlo en una sola tabla.

Se han creado las tablas genres, languages y countries para garantizar la integridad de datos, como se pedía en la práctica; también se ha añadido el campo balance a la tabla customers que sea no null y por defecto tenga el valor devuelto por la función, que está en el actualiza.sql, dentro de un rango de 0 a N=100.

En el campo orders se le ha puesto por defecto tax = 15.

El orden de ejecución debe ser igual al especificado en el enunciado, en caso contrario puede que no salgan los datos correctamente, ocurran errores y la salida puede que no sea la esperada.

Algunas películas tenían más de un año en la columna year. Esto producía problemas a la hora de obtener el año de debut de los top actores por género, por lo que modificamos estos valores para que se establezca como fecha de estreno el segundo valor entre las dos posibles (Por ejemplo pasamos de 1998-1999 a solamente 1999).

Se han añadido todas las foreign keys necesarias en las columnas de las tablas que hacen referencia a otras tablas.

La tabla orderdetails tenía algunos valores duplicados para las duplas (orderid, prod_id) lo que impedía utilizar estas columnas como clave principal. Para arreglar esto hemos agrupado los valores repetidos y sumado las cantidades de productos.

Hemos añadido cambios en cascada a algunas foreign keys para que en caso de que se eliminen valores de las tablas padre, se borren también las dependencias.

En el futuro sería interesante introducir en los detalles de cada pelicula sus principales actores, y que al hacer click en ellos te lleve a una página donde se muestran las principales películas donde aparecen y más información.