Santos Saenz Ferrero

Informatic Systems

UPDATE.SQL

After an analysys of the initial database I realize that there are missing foreign keys that would help with the erase and update in cascade necessary for the correct functioning of the database. Because of that I started adding external keys for the customers in its orders(table orders). Continuing with this task of adding external keys for a better functioning, I add in the imdb_actormovies table, this table relates the actors to the movies, external keys for both actorid and movieid.

It must be performed also the same action in imdb_directormovies, which is a similar table to the one before, that relates directors with movies (in this case I alter the autoincrement sequence of index because it increases the number of indexs of the table by tself).

We move on to the products table where I update and insert the Foreign Keys, for updating and deleting tables in cascade. The foreign key of the inventory table is added because this table contains the number of sales and the remaining stock of each product (all the information taken from the orderdetail table and inventory so we delete those orderdetail fields that we are using to avoid repetition on the table inventory after use).

The next table to point out is orderdetail which contained duplicate entries that they must be delated, but not before adding the quantities and modifying the prices of the orderdetails repeated (repeated means that they have the same orderid and same prod_id). After we add its primary index (helps for optimization).

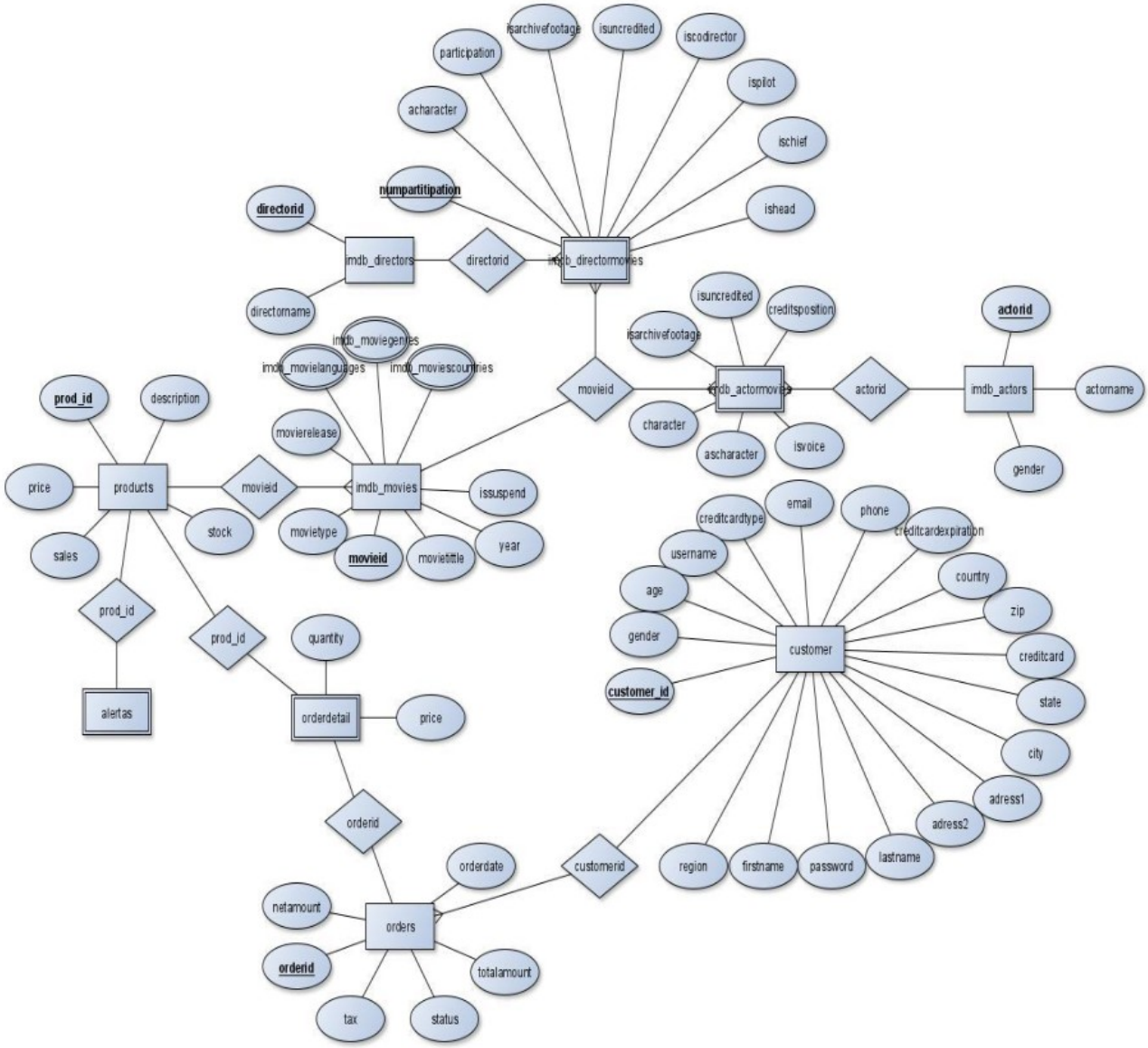
`To deal with the imdb_movielanguages table we create another table called languages that`
make it easy to index in two different tables and have clear information. I fill in the information of the new table and I correctly index both so that the deletion of movies is effective also in imdb_movielanguages.

The same idea is followed to deal with imdb_moviecountries and imdb_moviegenres since they have the same implementation.

The customers table is the “primary key of the database”, since it contains the customer data which is essential. From it, it is indexed the order tables to be able to generate simple deletions in cascade. In addition I delete the columns that I do not have implemented in the record and whose information is irrelevant.

Finally, I have created the alertas table, that when a product runs out of stock(inventory), is checked by the updInventory trigger.

Entity Relationship Diagram (ERD)



Functions, Triggers:

1. getTopSales:
   Function that uses a loop that goes from the first year entered as input to the second entered year. And within this loop it makes a selection of the films sold that year sorted by number of sales and selecting only the first, that it is, the largest.
2. `getTopMonths:`
   Function based on a select with two possible conditions, so we use an ‘or’ that joins them in the WHERE condition.
3. `updInventory:`
   Trigger that is executed when an order(order) goes from NULL(cart) to paid, because of that it is `needed to modify the inventory stock and the alertas table if the stock reaches 0`
4. `updOrders:`
   trigger that is fired when we delete, insert or update an orderdetail (we modify the products that are in the order) for which, we have to update as if outside of the requested query in the setOrderAmount file but in this case only for the order required. To do this, it is modified the totalamount and netamount of orders based on the price and the quantity of products of the order that are detailed in orderdetail.