CUSTOMER
PK userID:INT(55)

name:VARCHAR(255)
email:VARCHAR(255)
address:VARCHAR(255)

CUSTOMER_HAS_SHOPPING_CART
FK, PK   userID:INT(55)
FK, PK   cartID:INT(55)

SHOPPING_CART
PK   cartID:INT(55)

numOfitems:INT(55)
creditCardNum:INT(12)

is part of
is made of
is made of
is part of
is part of
is made of
is part of
is made of

USER_LEAVES_REVIEW
FK,PK   reviewID:(55)
FK,PK   userID:INT(55)

REVIEW
PK reviewID:INT(55)
reviewText:VARCHAR(255)

is made of
is part of
is part of
is made of

SHOPPING_CART_IS_MADE_OF_PRODUCTS
FK,PK   cartID:INT(55)
FK,PK   productID:INT(55)
      numOfProducts:INT(55)

PRODUCT
PK productID:INT(55)
FK taxID:ID(55)
FK historyID:INT(55)
   inventory:INT(255)
   price:NUM(6,2)

TAX
PK taxID:INT(55)
    value:INT(2)

is made of
is part of
has
is part of
has
is part of

PRODUCT_HAS_REVIEW
FK,PK   reviewID:INT(55)
FK,PK   productID:INT(55)

has
is part of
is made of
is part of

PRICE_HITORY
PK historyID:INT(55)
    numOfclicks:INT(255)
    numOfsales:INT(55)
    stardDate:DATE
    endDate:DATE

PRODUCT_SUPPLIERS
FK,PK   supplierID:INT(55)
FK, PK   productID:INT(55)
    deliveryTIme

is part of

SUPPLIER
PK supplierID:INT(55)
    supAdress:VARCHAR(255)
    supPhone:INT(10)
    supEmail: VARCHAR(255)

is made of

Denormalization:
      In the 3NF model a new table was created in order to avoid multi value attributes (the productID in this case) in the SHOPPING_CART table, to solve that issue a SHOPPING_CART_IS_MADE_OF_PRODUCTS table was created but that also meant creating a new table for each product inside each cart created, that could increase a lot the access time because the system would have to spent a lot of time making those connections and wasting a lot of space with the new attributes created. We then denormalized the relationship getting it back to its previous state, eliminating the

SHOPPING_CART_IS_MADE_OF_PRODUCTS, and adding the productID attribute directly inside SHOPPING_CART. The denormalized part is now looking like this: