Name: ______________________________ ID: __________ CS 480
Database Systems
Fall 2019

Homework #7

Complete By: Wednesday, November 20th @ 5:00pm
Submission: submitted digitally through codio

Writing SQL (100 points)

Use the tables in the populateSale.sql file included in the codio environment as the data to answer the following set of questions. This data has been preloaded for you, and may be reloaded by executing the populateSale.sql.
For each of the questions in this section, you will write SQL code as requested by the question. Put your answer into the submission folder in codio and mark the assignment as completed when done.

1. Write a procedure with the name prc_cust_add to add a new customer to the CUSTOMER table. Take the first name, last name, initial as parameters. Generate an id equal to the old maximum plus one. Leave the area code and Phone number as null. Set the balance to 0.
2. Invoke the procedure from question 1 to add the customer Peter C Rabbit to the database.
3. Create a stored procedure named prc_new_product to insert new rows in the PRODUCT table. The procedure should satisfy the following conditions:
    a. The vendor number will be provided as a parameter, along with the product code (not an int) and description. Set the INDATE to the current time, the Quantity On Hand to 0, the Minimum amount to keep on hand to 0, the price to 1.00, and the discount to 0.
    b. Check that there is a vendor matching that number which exists in the database. If it does not exist, then a message should be displayed that the membership does not exist, and no data should be written to the database. Signaling an exception state is acceptable means of generating a message, alternatively you could produce an output and then let the foreign key requirement cause the operation to fail.

c. If the vendor does exist, then retrieve the vendor name into a table as part of the execution of the procedure.

4. Invoke the procedure from question 1 to attempt to add the product with code ACME, description "Real Product", and vendor code 1 to the database.
   Invoke it again to add the product with code "EGG/TO", description "Egg Parachute", and vendor code 24004 to the database.

5. Write a function named TAX that computes the tax on an input value by multiplying it by 8%. Truncate the result (don't round up) to 2 decimal places.
   Test your function by invoking it in a select statement producing one row of two columns, the value 150.00, and the output of your TAX function when given an input of 150.00.

6. Write a procedure that when invoked, has a customer order a product. This procedure, named prc_place_order, should have the following properties.

   a. The parameters to the function include the customer code, the product code, and the quantity (number) of that product to be ordered.

   b. If either the customer code or product code reference entities that do not exist, the procedure should fail with no updates performed on the database.

   c. If the quantity is more than the Quantity on Hand of the product, the procedure should fail with no updates performed on the database.

   d. The subtotal should be calculated by multiplying the price of the product by the number ordered.

   e. This function generates a Line item, an entry in the LINE table. The new entry should have the following properties.

      I. A newly generated Invoice Number from a newly generated Invoice object, with an automatically generated Invoice Number, the customer code which was given as input, the invoice date from the current datetime, the subtotal as computed before, the tax should be computed by multiplying the subtotal by 8%, and the total should be computed by adding the subtotal and tax.

      II. An automatically generated Line Number.

      III. The product code given as input

      IV. The units is the quantity given as input

      V. The price is taken from the product table for the given product.

      VI. The total is the subtotal computed before.

Once the line item has been created, return the invoice number, line number, and total cost of the purchase.

7. Create a procedure named prc_add_to_invoice which adds a purchase to an existing invoice that takes an invoice number, a product code, and a quantity as inputs, and adds that line item to the invoice.

   The newly added line item should follow the rules from question 6, with the exception of creating a new invoice entity. Instead, the invoice number should match the one given in the input, and the invoice should be updated with a new subtotal, tax, and total based on the new subtotal produced by adding the total from the line item.

8. Create a function named prc_cus_balance_update that will take the invoice number as a parameter and update the customer balance by subtracting the total from the customer balance. If the balance would become negative, signal an exception with the output "Insufficient balance: Has X needs Y" where X and Y are the customers current balance and total from the invoice respectively, and do not apply the change. If the customer does have sufficient funds, return the new value of the customer's balance.

9. Write a procedure named prc_restock which takes no inputs and produces a table containing for each product which has less quantity on hand than the minimum specified for that product:
   The product codes, the number by which the quantity on hand is less than the minimum specified for that product, and the vendor code for that product.

10. Assuming that the foreign key CUS_CODE in the INVOICE table does not support "ON DELETE CASCADE" (it doesn't in the database we've given you) Create a procedure named prc_expunge_cust that will take the customer ID (i.e., CUS_CODE) as a parameter and remove the customer from the CUSTOMER table and remove all rows in the INVOICE table that are associated with this customer, and all rows from the LINE table associated with those INVOICE.