PL/SQL Basics – Day 5 Assignments

Consider the below tables Customer, Product of an online store database.
Customer(Cust_ID Number, Cust_Name Varchar2(50), Join_Date Date, Last_Visit_Date Date)
Product(PROD_ID NUMBER, PROD_NAME VARCHAR2(50), COST NUMBER, STOCK_QUANTITY NUMBER, RED_FLAG VARCHAR2(10), LAST_SOLD_DATE DATE)

1) Create a procedure bill_calc(CUST_ID, PROD_ID,QUANTITY) which takes CUST_ID, PROD_ID,QUANTITY as input and prints total cost of the purchase.
    i) The procedure should print "OUT OF STOCK" if the quantity requested is more than that in stock and set the RED_FLAG as 'TRUE' for that product.
    ii) If a customer is a new customer (he/she has joined within the past 30 days) then they get a 5% discount.
    iii) If a customer's last visit was within the past 10 days he/she gets a 5% discount.
    iv) If the product has not been sold in the last 20 days then give an additional discount of 10%.
2) Create a new procedure Avg_calc(CUST_ID, PROD_ID,QUANTITY), to find the average cost of all the products and check if the product being bought is above or below the average cost (Products having RED_FLAG = TRUE should not be considered for calculating average).
    i) The costs is above the average cost then give the customer a discount, charge them 50% of the cost price for one of the units regardless of the quantity of products bought.
    ii) The costs is below the average cost then give the customer a discount, charge them 25% of the cost price for one of the units only if the quantity being bought is more than 2.
    iii) Find out the product which has not been bought for the longest period of time and doesn't have a RED_FLAG set to TRUE (in case of multiple products having the same last_sold_date choose the one with the highest quantity) and suggest the customer to buy it at a 25% discount.

3) Create a package to have the following functionality
    i) Inserting customers
    ii) Inserting Products
    iii) Deleting Customers
    iv) Deleting Products
    v) Calculating Bill (Use the proc in assingment1)
    vi) Editing Customer details
    vii) Editing Product details

4)
    a) Edit the Package created in assignment 3, create a function called compute_star_prod in the package which returns the star product_name in the Product table. The criteria for finding the star product is,
        - The product should have least amount of stock_quantity.
        - The most recent LAST_SOLD_DATE(Give priority to LAST_SOLD_DATE in calculating Star product).
        - Should not have RED_FLAG = FALSE.
    b) Create a function check_duplicate_customer which takes customer_name as an input and finds out if the same customer has multiple entries in the DB. If so then return 'TRUE' else return 'FALSE'.
5) Create a procedure that finds out the oldest customer in the store for that customer check the last visited date
    i) If the date is older than 60 days, delete his/her record.
    ii) If the date is 30-60 days old then append '_consider_deleting' to the end of his name.
    iii) If the date is 15-30 days old then add 'Call_back_' at the beginning of their name.
    iv) If the date is 0-15 days old then add '_TRUSTED_' after the first 2 characters of their name.
    v) After performing these additions to his/her name check the length of the name string. If the name is longer than 20 characters then check for white spaces in the string and remove them.

6) Write a procedure that accepts a string as input and sends another string as output. Business Rules to be followed.
    i) The string should not be blank. Raise an user exception “Please do not enter blank values”.
    ii) The string should be at least 10 characters long. Else throw user defined exception “Please enter a string greater than 10 characters”.
    iii) The last 4 characters entered should be numeric and the last but fourth character cannot be zero. If this is so, please raise user exception.
    iv) The last 4 characters of the string should be a multiple of 8. If not, throw an error “Please enter a string whose last 4 digits are a multiple of 8”.

7) Write a function that accepts member_id and delimiter as input, returns the member’s name, the smallest attribute_id and attribute_name mapped to him in this format.
    i) All strings to be quoted with single quotes. I have taken ~as an example of the delimiter. ‘John55’~’Smith55’~1024~’runidcorrect’.
    ii) ‘firstname’ ~ ‘Last name’~attribute_id ~’ attribute name’. Use this in a query and fetch all the member_ids and the transformed function output.

    Note: Use Vendor_Members, MT_Members tables.