# 1. – PROBLEM DESCRIPTION

On the relational database already designed for the coffee products e-commerce company (*CAFFEINEMANIACS*™ Inc.), it is necessary to develop some operative elements. Specifically queries, views, procedures, and triggers. The description of these elements is as follows:

## 1.1.- Queries (2 points, one point each)

a) Bestsellers Geographic Report: for the last year (current year minus one) provide the best sold varietal per country (in number of buyers); along with total of buyer, provide the total units sold, total income (money), average units sold per reference, number of countries who are potential consumers of the varietal (buying more than 1% of total units sold, trademarks (product names) of that varietal.

b) Business way of life: For last twelve months, provide a monthly report (twelve rows) with the best sold reference (#units sold) each month, along with its number of purchases, its number of units sold, total income regarding that reference, and total benefit (income-cost) due to that reference (totally, 6 columns). Months for which there's no data (no purchases) can be excluded from the report (so you get less than 12 rows).

## 1.2.- Operativity (package 'caffeine' containing these two public procedures) (2 points, distributed as 0,5+1.5 respectively)

- Procedure Set Replacement Orders: convert draft orders into placed orders
- Report on a provider: receives a CIF (a provider), and displays some statistics: number of orders placed/fulfilled in the last year, average delivery period for already fulfilled offers; and the detail of their offers: for each reference they offer, specify current cost, minimum and maximum cost (during last year), difference of current cost minus the average of costs of all offers (regarding that reference), and its difference regarding the best offer for the product (in case the best offer is that of the reported provided, the second best offer will be taken).

Testing: you have to insert directly some data for testing this section. Summarize the data entered in your report.

## 1.3.- External Design: "client" user profile (2 points, respectively … 0.5+0.5+1). For this section you need to define the concept of *'current user'*. You can decide either to take the value provided by nullary function USER, or create a package with a public variable (for example, current_user VARCHAR2(30) ).

- **`my_purchases`** (*read only*) view: shows all current users purchases.
- **`my_profile`** (*read only*) view: displays my personal data, addresses and credit cards.
- **`my_posts`** (*full operativity*) view: lists current user posts. Allows inserting new posts, removing posts, or changing the text of a post (only if its likes are zero; the rest of attributes shouldn't be changed). Posts can't be removed when their *likes* are greater than zero.

**1.4.- Active Databases/Triggering (4 points,** one point each**)**
Design, Implement, and Test the following triggers:

a) When a post is added (or edited), calculate the attribute endorsed (if the user has purchased that product or reference previously, endorsed is 'Y', and otherwise 'N').
b) When a client is removed, move his/her purchases to the anonymous purchases. Move his/her posts as well. Optional: notice that two anonymous comments could show the same date and time, and any solution is actually limited; comment (in the written report) the limit of your solution, and how to extend that limit (and to what extent).
c) Prevent the insertion of anonymous purchase using a credit card already stored for a registered user.
d) Update stocks: when a new purchase is set, update current stock and, in case, insert a draft replacement order.

## 2. – STARTING POINT

A whole new database, already populated with the data from the previously available database. For you to replicate such DB, both creation and load scripts will be provided, along with a brief documentation on that design (relational graph, semantic comments). First step is, therefore, running those scripts for replicating the development environment.

## 3. – SUPPORTING MATERIALS

Apart from classes and tutoring sessions, students can count on the following resources:

- Documents:
  - assignment statement (this doc);
  - class slides;
  - solution to the first assignment (design and comments)
  - template for writing the assignment report (.docx format).
- Audiovisual resources: video classes to acquire specific knowledge about the use of the tools that will be used in the laboratories (pl/sql syntax) in the "inverted class" style.
- Sw Resources: user account on RDBMS Oracle (accessible from all computer rooms in the University, and from Aula Virtual), with enough privileges for all required operations and reading privileges on the obsolete DB's tables. Scripts for replicating environment:
  - Script for creating the tables that implement the new Database.
  - Data migration script from the obsolete DB to the new DB.

## 4. – TO DO

All the results of this ASSIGNMENT (designs, code, tests) will be collected in a single document (*assignment report*, in PDF format). For each section of the first item in the statement (queries, package, external design, triggers) include a chapter in your assignment's report. Within each chapter, include a separate section for each element required in the statement. These sections must follow the same order that has been established in the statement.

Each of these sections will be presented with the description of the element to be developed (statement), and its resolution with three subsections, as described below: **design**, **implementation**, and **testing**. Notice that those subsections are worth **30%**, **40%** and **30%** of the **score** (respectively), so neglecting any of them can involve severe score loss.

A. **Design**: queries should be described in relational algebra (views and subqueries must also be documented); as for code blocks, you have to describe their logic (working); finally, the parameterization of the ECA rules (triggers) must be justified. The design will be accompanied by relevant comments about the implicit semantics incorporated or the explicit not reflected (wherever necessary). In case, new/modified elements (such as, tables) will be documented.

B. **Implementation**: PL/SQL code that implements that element. The code must be collected in text format (so that it can be easily transferred by copying from the .pdf document and pasting onto the sql*plus console for eventual running checks). Besides, the code has to be properly indented to improve its readability, and commented in any aspect not included in the former section (design).

C. **Tests**: description of the actions to be carried out to verify the correct working of the implemented element, description of the expected result, and description of the result obtained (accompany by screenshots to illustrate that result). When the result obtained differs from that expected, explain the deviation, the problem detected (if any) and the actions necessary to correct.

Examples:

- A query can be verified by running it several times on as many different states of the DB, and establishing in advance the differences between the results that should be obtained. Note that a simple execution is not a test (it is not known whether its result is correct or not). But by changing the state of the DB is changed (inserting, deleting and/or updating data) so that the result of the query change, and so that the result can be either predicted or characterized in advance, then useful test cases can be established. Exhaustive testing is not requested in this work (just include some examples of testing).
- Checking views is analogous to checking queries.
- Blocks (procedures) can be tested by running with different parameters (checking results).
- A trigger can be checked by causing it to be activated in different cases (forcing the triggering event) and checking its effect. If there is a risk of a mutating table error, it is convenient to check it by means of activations that involve more than one row.

Document all the work carried out by means of the pertinent *Labwork Report*, for which writing a template is provided. Apart from including the requested elements with their development stages (design, implementation, test), make sure that all design decisions are conveniently

justified and thus reflected in the report. Save the document as **.pdf** file (portable document format), name it as ***nia1_nia2_nia3_LW2.pdf*** , and submit it through Aula Global.

Just one student from each team should deliver.

Deadline: April 12, 2024