Home Work #7

1. The Problem

Softy, Inc. is a software development firm with over a thousand programmers. A database is required to keep track of the programmers and the programming tasks to which they are assigned.

Programming tasks are distinguished by unique names (e.g., *Canyon*, *Driverless*). The starting date and the target and actual completion dates of each task must also be stored. A task typically has several programmers working for it but each programmer is assigned exactly one task at any given time. Typically, a programmer is assigned a different task when the current one is complete, but sometimes re-assignments occur in an active project.

Every programmer has a unique ID number assigned by the firm; we must also store his/her name, gender, yearly salary, date of birth, and job title (e.g., *Analyst*, *Trainer*). Each programmer has exactly one supervisor who is also a programmer. (A few supervise themselves.)

For each programmer, we need to know which programming language(s) (e.g., C, Python, Haskell, Oracle PL/SQL) he/she is known to be proficient in and also when he/she became proficient in that language. Softy, Inc. does not hire any programmer until he/she is demonstrably proficient in at least one language and also ensures that there is at least one proficient programmer per programming language used by the company.

Each programming language has a unique name. It is important to store whether or not it is a compiled or interpreted language, and also the name of the examination suite that programmers take in order to demonstrate their proficiency. Further, we must record exactly which of these languages a programmer uses in each task assigned to him or her.

Occasionally, tasks need to be revisited after completion. Hence, it is important to store information of completed tasks, the programmers who worked on them (even if they have since left the company), the number of days they worked, and the programming languages they used.

All IDs are numbers with at most 6 digits. All names are at most 15 characters in length. Monetary figures are all in US dollars and involve no more than nine digits before the decimal point. Let all character strings in the database be stored entirely in lowercase.

2. The Project

1. Design[1] an ER schema for Softy, Inc. based on the above problem statement; justify any need for null values.
2. Convert the above ER schema into a relation schema with constraints avoiding the possibility of null values as far as possible.;
3. Finally, implement the following queries, procedures, and triggers on Oracle (on the TCC) using `sqlplus`. Implement all constraints using declarative means whenever possible and by triggers otherwise.

2.1. Queries

Use a single standard `select` statement for each of the following. Use `group by` only if you must.

Q-1: For each task that is active today (the day the query is executed), find the number of days remaining (i.e., till the target completion date): display the task name, the target completion date, and number of remaining months in ascending order of completion date (the earliest will be first).

Q-2: Find the IDs and names of all programmers who are assigned to a task that is to terminate some time in the year 2017; present the result in ascending order of programmer ID.

Q-3: For each programming language used in at least one active task, display its name, the most senior programmer(s) in terms of proficiency, and the date that was achieved.

Q-4: For each programming language, display the id's and names of programmer-supervisor pairs whenever both the programmer and his/her supervisor are proficient in that language. Do not list a programmer who is his/her own supervisor.

2.2. Procedures

Implement the following stored procedures using cursors in PL/SQL.

P-1: Terminate a task. Take its name as input.

P-2: Retire a programming language. Take its name as input. (You must ensure that it is not used in any active project and that no programmer is left proficient in zero languages.)

[1] You must keep yourself informed about any clarification(s).

P-3: Handle a programmer who is leaving or retiring. Take his/her ID as input.

P-4: Given an already terminated task name as input, display the names of all programmers who were assigned to this task and the number of days they worked. Print a message if the task does not exist.

2.3. Triggers

Implement the following constraints through PL/SQL triggers.

T-1: All integrity constraints that were not declared in the schema. Use clear comments to specify which constraint a trigger is attempting to maintain.

T-2: A programmer cannot use a programming language (for a task) that he or she is not proficient in.

2.4. Extra credit

Implement the following queries using a single select statement if possible; use PL/SQL only if absolutely necessary.

EC-1: Find the id's and names of programmers who use all the programming languages in which they are proficient.

EC-2: Find the programming language (L) known by the maximum number (n) of programmers. In case of a tie, list all the tied pairs.

3. What to Submit

3.1. PDF file

1. Start with the following text provided you can honestly agree with it.
    - *I certify that every answer in this assignment is the result of my own work; that I have neither obtained my answers from the Internet nor from any one else; and I have not shared my answers or attempts at answers with anyone else.*
2. Include the ER diagram(s) plus any pertinent assumption(s) and constraint(s) not represent-able in the diagram(s). Clearly indicate which, if any, of these constraints you have implemented.

3.2 Code

Upload an `sqlplus` script file named YOURUSERNAME-proj.sql (an ASCII or text file created in a Unix environment containing no non-printable character other than `<LF>`) that the grader can execute to mimic your work. It should consist of

1. useful comments throughout the file beginning with a comment block giving your name, date, and a summary of the project;
2. statements specifying the relational schema, i.e., to create/alter the tables, including declarative integrity constraints;
3. definitions of procedures and triggers.

   If you have implemented any constraint using triggers, state the constraint that thetrigger is implementing;
4. statements to populate the tables,
5. statements to display the contents of all tables;
6. statements to invoke the various queries and stored procedures;
7. statements to invoke updates that display your trigger(s) in action; and
8. **a commented** block containing statements that clean up the data you have entered and the tables you have created.

   Note: This is important.

   - If this block is *missing*, the grader will have to spend time inferring how to destroy the data and code created.
   - If this block is *not commented*, the grader will be left with an empty database after executing your script, unable to experiment or verify!

3.3 Spool file

Upload a spool file (an ASCII or text file named YOURUSERNAME-proj.lst) recording the execution of your final script file; ensure that

1. it displays your schema and code (functions, procedures, triggers) and demonstrates that your code work correctly and also that violations of the constraints you have implemented are prevented.
2. there are comments that indicate to the grader what the following SQL or PL/SQL statement(s) demonstrates.