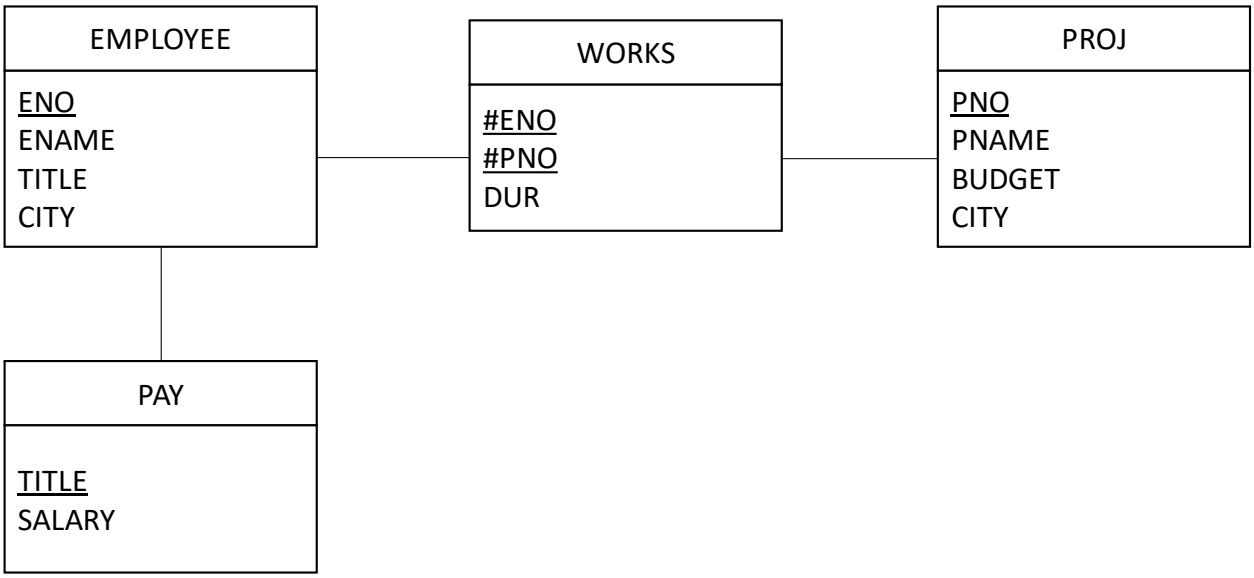
TD3 PLSQL – Concepts de base

1 – Télécharger le fichier emp_and_projects.sql et l'exécuter pour le chargement de la base de données emp_and_projects.

2 – Le schéma de emp_and_project est montré dans la figure suivante :



1 – Maitrise des concepts de base :

a) Exécuter : `set serveroutput on size 30000` pour permettre l'affichage avec la requête `DBMS_OUTPUT.put_line`

b) Ecrire le bloc PL/SQL qui permet d'initialiser deux variables de type varchar2 qui contiennent respectivement votre nom et votre prénom et de l'afficher.

c) Ajouter dans le bloc PL/SQL précédent l'affichage du nombre de caractères que contient votre nom et votre prénom (opérateur `length`) et de vérifier si votre nom ou votre prénom contient une lettre 'c' dans la troisième position. Utiliser deux variables pour réaliser ceci.

d) Ecrire le bloc PL/SQL qui permet de réaliser une boucle qui permet d'afficher les 10 premiers entiers avec 10 (utiliser les trois types de boucles, FOR, WHILE et LOOP simple).

2 – Manipulation basique des tables :

a) Ecrire le bloc PL/SQL qui permet d'afficher les informations de l'employé KRUNAL

b) Ecrire le bloc PL/SQL qui permet d'afficher les informations de l'employé qui est de la ville 'TORONTO'. Pourquoi ça ne marche pas ?

c) Ecrire le bloc PL/SQL qui permet de réaliser une boucle qui permet d'afficher les n premiers entiers avec n la taille de la table EMP

d) Ecrire le bloc PL/SQL qui permet d'afficher les employés de EMP en utilisant leurs identifiants

e) Ecrire le bloc PL/SQL qui permet d'afficher les durées de travail (table WORKS) **paires** de chaque employé sans utiliser **WHERE**.

f) Ecrire le bloc PL/SQL qui permet d'insérer 5 employés de E9 à E13 qui sont des 'PROGRAMMER' à 'NEW YORK'. Les noms sont identiques à leurs identifiants. (Utilisation d'une boucle).

g) Ecrire le bloc PL/SQL qui permet de modifier l'emploie des 5 employés de NEW YORK de PROGRAMMER à ''SUPPORT STAFF'

3 – Utilisation des curseurs :

a) Ecrire le bloc PL/SQL qui permet de parcourir et afficher tous les projets avec la boucle FOR

b) Ecrire le bloc PL/SQL qui permet de parcourir et afficher tous les projets avec la boucle WHILE

c) Ecrire le bloc PL/SQL qui permet d'afficher les employés avec leur salaire en utilisant une jointure

d) Ecrire le bloc PL/SQL qui permet de modifier les salaires des employés (table PAY) de la manière suivante :

- Si le salaire dépasse la moyenne, le diminuer par 10%
- Sinon l'augmenter par 10%

e) Ecrire le bloc PL/SQL qui permet de calculer la moyenne des salaires pour chaque ville (avec l'utilisation de group by)

f) Ecrire le bloc PL/SQL qui permet de calculer la moyenne des salaires pour chaque ville (sans l'utilisation de group by)

4 – Gestion des exceptions :

a) Ecrire le bloc PL/SQL qui permet d'afficher les informations de l'employé BUTTERS. Gérer ensuite l'exception remontée par ce bloc.

b) Dans la question **2.b**, le bloc PL/SQL remonte une exception puisque la requête **select … into** retourne plusieurs valeurs. Modifier la procédure de telle manière à ce que le bloc affiche le nombre de lignes retournées en utilisant un traitement d'exceptions.

c) Ecrire un bloc PL/SQL qui parcours la table des salaires des employés (PAY) en utilisant les curseurs. Ce bloc doit générer une exception dès qu'il trouve un salaire dépassant le seuil de 10000. L'exception doit être déclarée d'une manière explicite.

d) Ecrire un bloc PL/SQL qui permet de générer une exception s'il existe au moins un employé qui n'est pas affecté a aucun projet.

5 – Fonctions et procédures :

a) Ecrire une fonction PL/SQL qui permet de retourner le maximum entre deux entiers.

b) Ecrire une fonction qui permet de retourner le nombre d'employés.

c) Ecrire une fonction qui permet de retourner le nombre de programmeurs qui travaillent sur un projet P_i.

d) En supposant que le salaire perçu par un employé durant tout un projet est égal le salaire correspondant à son titre multiplié par la durée sur le projet. Ecrire une fonction qui permet de déduire l'argent qui doit être ajouté au budget d'un projet donné pour subvenir à la totalité du coût du projet en termes de salaires.

e) Utiliser la fonction précédemment déclarée pour écrire une procédure PL/SQL qui permet de rétablir les budgets nécessaires pour chaque projet.

f) Ecrire une procédure PL/SQL qui permet d'ajouter un MANAGER sur les projets qui ne possèdent pas un manager.

g) Rétablir la base de données en utilisant procédure écrite a **'e'** (*Ce rétablissement doit être fait automatiquement après le cours des Triggers*)

6 – Les déclencheurs :

a) Ecrire un trigger qui s'exécute avant une insertion ou une modification dans la table « Employee », afin de vérifier si le nom de l'employé concerné est écrit en majuscule. Si ce n'est pas le cas, le trigger génère une exception.

b) Modifier le trigger de la question précédente. Au lieu de générer une exception quand le nom n'est pas en majuscule, le trigger le rétablit en majuscule.

c) Ecrire un trigger qui met à jour le budget total d'un projet quand il y a un ajout ou une suppression d'une ligne dans la table « works », afin d'assurer que le budget du projet est égal à la somme des salaires perçus par les employés concernés pendant la durée du projet.

d) Ecrire un trigger qui met à jour les budgets des projets si un salaire est modifié au niveau de la table « Pay ».

e) Ecrire un trigger qui empêche d'avoir un salaire inférieur au SMIC (1480), et un autre trigger qui vérifie que la durée d'une tâche dans « works » est toujours comprise entre 0 et 24. Ces triggers génèrent une erreur s'il y a une action qui viole ces contraintes. Comment on peut assurer ces contraintes sans utiliser les triggers ?