Soit le schéma relationnel suivant :

- REGIONS(REGION_ID, REGION_NAME)
- COUNTRIES(COUNTRY_ID,COUNTRY_NAME, #REGION_ID) ;
- LOCATIONS(LOCATION_ID, STREET_ADDRESS, POSTAL_CODE, CITY, STATE_PROVINCE, #COUNTRY_ID) ;
- JOBS(JOB_ID, JOB_TITLE, MIN_SALARY, MAX_SALARY) ;
- EMPLOYEES(EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, #JOB_ID, SALARY, COMMISSION_PCT, #MANAGER_ID, #DEPARTMENT_ID) ;
- DEPARTMENTS(DEPARTMENT_ID, DEPARTMENT_NAME, #MANAGER_ID, LOCATION_ID) ;
- JOB_HISTORY(#EMPLOYEE_ID, START_DATE, END_DATE, #JOB_ID, #DEPARTMENT_ID) ;

Regions : Region_id :l'identifiant d'une région d'un employé. ;Region_name est le mon de cette région.

Countries : Country_id : l'identifiant d'un pays; country_name :le nom d'un pays ; Region_id : est le nom de la région d'un pays elle référence la table région pour le même attribut.

Locations est la table qui désigne l'adresse exacte : par Location_id : l'identifiant d'une adresse, Street_adresse est la rue, le postal_code est le code postal, state_province est la province (proche du gouvernorat), le prix de l'article; countriy_id référence la table countries (pays).

Jobs :ou emplois avec Job_id est l'identifiant d'un poste, Job_title : le titre d'un poste, min_salary et max_salary sont respectivement le salaire minimum et maximum pour un poste donné.

Employees : empolyee_id est l'identifiant d'un employé, First et last_name sont respectivement le prénom et le nom d'un employé. Les coordonnées d'un employé sont données par Email et phone_number. Hire_date est la date d'embauche, Job_id est l'identifiant du poste occupé par l'employé, il référence la table jobs, salary est le salaire touché par l'employé, commission_pct est le pourcentage d'une commission qui peut être éventuellement attribuée à l'employé, Manager_id est le superviseur des employés d'un département c'est une clé étrangère qui référence la même table employé sur son attribut employee_id (référence reflexive), Department_id l'identifiant du département dans lequel travaille l'employé, il référence la table département.

Departments : infos départements y compris : l'identifiant : par department_id, le nom : par department_name, le manager_id qui est le chef d'un département_id : c'est une clé étrangère

qui référence la table employees pour l'attribut employee_id et la location_id qui donne l'identifiant de la localisation d'un département et qui référence la table locations.

Job_History est une table d'archivage qui sauvegarde pour chaque employé embauché les différents postes qu'il a occupé au sein de l'entreprise en mentionnant son identifiant, la date du début de sa mission (Start_date) et la date de fin (end_date), le poste qu'il a occupé (job_id qui référence la table jobs) et au sein de quel département (department_id qui référence la table departments).

1- Commencer par créer un utilisateur pour ce TP (par exemple TP3MI2), donner lui les droits nécessaires pour administrer son tablespace.

2- Connecter vous en tant que cet utilisateur et effectuer tout le travail concernant le TP dans son espace dédié.

3- Exécuter le contenu du fichier TP3 pour créer et remplir les tables.

NB : Pour remplir un nouvel enregistrement, il faut considérer les séquences suivantes pour les nouvelles vleurs de clés primaires :

- locations_seq,
- departments_seq,
- employees_seq,
- locations_seq,
- departments_seq

Apporter des solutions en PL/SQL aux exercices listés ci-dessous :

Exercice 1

Ecrire un bloc anonyme PL/SQL qui permet d'afficher le nom et le revenu mensuel d'un employé dont l'id est saisi par l'utilisateur (revenu mensuel=salaire * (1+commission)).

NB n'oublier pas de bien configurer la console pour permettre :

- L'affichage,
- La non vérification des valeurs introduites au clavier.
- La demande de la saisie d'un identifiant au clavier
- Prévoir un traitement d'exception (utiliser les exceptions internes nommées).

Exercice 2

Ecrire un bloc anonyme PL/SQL qui permet d'afficher pour un département donné (dont l'id est saisi au clavier), le total du salaire et de la commission dépensée en faveur pour la totalité de ses employés et le salaire moyen. Ce dernier doit être faire appel à une fonction fctsalaireMoy (non-stockée).

Le programme doit permettre d'insérer dans une table provisoire temp que vous allez créer en dehors du bloc, le numéro du département, son manager, le total dépensé et le salaire moyen.

A la fin supprimer la table temp.

Exercice 3

1) Ecrire une procédure PL/SQL permettant d'afficher les 6 premiers employés ayant les salaires les plus élevés. Pour chacun de ces employés, afficher leur nom, leur job_id ainsi que leur salaire. Ordonner le résultat de telle sorte que le premier employé est celui ayant le salaire le plus élevé.

2) Exécuter la procédure dans un bloc PLSQL anonyme.

Exercice 4

En considérant la table DEPARTMENTS, on propose le code suivant executé sur console:

```
  ACCEPT p_dept_nom PROMPT 'Entrer un nom de département : '

BEGIN
  INSERT INTO departments(department_id, department_name, location_id)
  VALUES (departments_seq.NEXTVAL, '&p_dept_nom', 1700);
COMMIT;
END;
/
```

1) Expliquer le fonctionnement du programme PL/SQL ci-dessus :

2) Exécuter le code avec la valeur *'Health'*.

3) Afficher le nouveau numéro de département créé.

4) Créer un bloc PL/SQL pour supprimer un département. Pour cela créer un paramètre pour le numéro de département et faire afficher le nombre de lignes supprimées.