b- Les valeurs des numpers de la table PERSONNE forment une séquence de 1 à 21. Utilisez une boucle dans laquelle vous placerez une requête pour recopier les couples nom/prénom de la table personne dans la table CLIENT. Réexécuter de nouveau votre code. Que se passe-t-il ?
Vider la table personne puis lancer votre script. Qu'est-ce que vous obtenez ?
- **Créer une procédure :** CopyFromPersonneToClient **pour refaire le même travail en utilisant un curseur.**

NB. Vous rattraperez les exceptions NO DATA FOUND et TOO MANY ROWS, chaque fois qu'un SELECT ... INTO ... sera effectué et l'exception DUP VAL ON INDEX après chaque insertion.
(Pensez à mettre : SET AUTOCOMMIT OFF)

c- Ecrire un script PL/SQL récupérant le client de clé primaire la plus élevée, et injectant ce client dans la table PERSONNEL.

d- Ecrire en PL/SQL une fonction demi-frères prenant deux numéros de personnes en paramètre et retournant vrai si et seulement si ces deux personnes ont un parent en commun.

La fonction return Boolean et échoue sur l'erreur suivante : "l'expression est de type incorrect".
Au fait, Pur SQL (ex : select ... From ...) ne permet pas de reconnaître un type booléen, bien que le cas pour PL/SQL

e- Ecrire en PL/SQL une procédure récursive affichant les noms des ascendants de sexe masculin de la personne dont le numéro est passé en paramètre

f- Soit la contrainte suivante : « Le nouveau salaire doit être toujours supérieur à l'ancien salaire ».
Ecrire en code PL/SQL un trigger qui permet de stopper toute mise à jour qui viole cette contrainte sur la table « Personnel » puis tester votre code.
N.B : Le trigger sera déclenché après toute màj qui viole cette contrainte.