R3.07 - SQL dans un langage de programmation

TP3 - Banque (fin)

Apprentis - A. Ridard

I. Auto-incrémentation des clés primaires

AUTO_INCREMENT disponible sous MySQL n'avait pas d'équivalent sous Oracle 11g.
Heureusement, GENERATED BY DEFAULT AS IDENTITY est maintenant disponible sous Oracle 12c.
L'inconvénient est que l'on ne contrôle pas la séquence sous-jacente qui, par exemple, ne peut pas être réinitialisée.

1. A l'aide du code ci-dessous, implémenter l'auto-incrémentation de la clé primaire **numAgence**.

```
DROP SEQUENCE seq_Agence;
CREATE SEQUENCE seq_Agence;

CREATE OR REPLACE TRIGGER trig_seq_Agence
BEFORE INSERT ON Agence
FOR EACH ROW
WHEN (NEW.numAgence IS NULL)
BEGIN
    SELECT seq_Agence.NEXTVAL INTO :NEW.numAgence
    FROM DUAL;
END;
/
```

2. Procéder de la même manière pour les autres clés primaires simples.

II. Une demande de virement

Nous souhaitons enfin gérer une demande de virement (interne à la banque mais entre des clients éventuellement différents) :

- Si le compte émetteur est débiteur, le virement est refusé et un message d'erreur est envoyé.
- Sinon, le virement est réalisé et modifie la base de données en conséquence.

7. Écrire une fonction **fct_estDebiteur** définie par :
   - **Entrée** : le numéro de compte
   - **Sortie** : 1 si le compte est débiteur (solde négatif) et 0 sinon

8. Écrire une procédure **proc_ajoutOperation** définie par :
   - **Entrées** : le type d'opération, le montant, le client [a] et le compte
   - **Actions** : insère une ligne dans la table Opération et modifie le compte en conséquence

9. Programmer et tester la procédure de virement [b].

a. A quoi sert ce paramètre ?
b. **(Facultatif)** Si le montant dépasse 1000 euros, le virement est refusé et un message d'erreur est envoyé précisant le nom de l'agent qui gère le compte