**Exercice 1 :**

**Parmi les déclarations de variables suivantes, déterminer celles qui sont incorrectes :**   
    
A -    DECLARE   
          v\_id     NUMBER(4);   
  
B -    DECLARE   
         v\_x,v\_y,v\_z    VARCHAR2(10);   
  
C -   DECLARE   
         v\_date\_naissance DATE NOT NULL;   
  
D -    DECLARE   
          v\_en\_stock    BOOLEAN := 1;   
  
E -    DECLARE   
         emp\_record    emp\_record\_type;   
  
F -    DECLARE   
         TYPE      type\_table\_nom IS TABLE OF VARCHAR2(20)   
         INDEX BY BINARY\_INTEGER;   
         dept\_table\_nom     type\_table\_nom;

**Exercice 2 :**

**1 - Créer un bloc PL/SQL pour insérer un nouveau département dans la table DEPT**

a) Créer la séquence DEPT\_ID\_SEQ et utilisez-la pour générer un numéro de département. Créer un paramètre pour  le nom du département. Laisser la ville à NULL.

b) Exécuter le bloc PL/SQL avec la valeur “REC” pour le nom du département.

c) Afficher le nouveau département créé

**2 - Créer un bloc PL/SQL pour supprimer le département créé précédemment**

a) Créer un paramètre pour le numéro de département. Faire afficher à l’écran le nombre de lignes affectées.

b) Tester le bloc. Que se passe-t-il si on saisit un numéro de département qui n’existe pas?

Si on saisit un numéro de département qui a des employés?

c) Tester le bloc avec le département REC(50)

d) Vérifier que le département n’existe plus

**Exercice 3 :**

**1 - Créer un bloc PL/SQL permettant de mettre à jour le pourcentage de commission d’un employé en fonction du total des commissions**

a) Créer un paramètre qui reçoit un numéro d’employé   
  
     Trouver la moyenne de toutes les commissions traitées   
- si la comm de l’employé est inférieure à la moyenne passer la commission à 1000   
- si la comm est supérieure ou égale à la moyenne passer la commission à 500   
- si la comm est nulle pour cet employé, mettre la commission à 0   
     Valider la modification (commit)   
  
b) Tester le bloc et visualiser les résultats (on teste avec les employés 7369,7654)

**2 - Créer un bloc PL/SQL qui boucle pour chaque département afin de modifier  la commission de tous les employés. Ne pas valider (pas de commit).**

     Si  le numéro de département est pair mettre  la commission à 500 (même si elle  l’est déjà), sinon mettre à 1000 pour les numéros de département impairs.

     Une fois les lignes modifiées, trouvez combien de lignes ont été mises à jour.

**Afficher les résultats suivants en fonction du nombre de lignes modifiées :**

* **Si  moins  de  trois  lignes  ont  été  modifiées,  afficher  :**  ‘Moins  de  trois  lignes  ont  été  modifiées pour le DEPT x’ (x étant le numéro du département).
* **Sinon afficher  :** ‘y  lignes ont été modifiées pour  le  DEPT x’(y étant  le nombre de  lignes modifiées).

     Annuler les modifications  (rollback).

**Exercice 4 :**

Créer un bloc PL/SQL qui détermine les employés de plus haut salaire.   
  
a) Créer pour cet exercice une nouvelle table pour stocker les employés et leurs salaires

b) Utiliser un paramètre pour prendre une valeur n en entrée pour identifier les n meilleurs.   
  
 Ecrire une boucle WHILE  avec  curseur pour  récupérer  le nom  et  salaire des n meilleurs  employés  selon  leur salaire dans la table EMP   
  
    Enregistrer les noms et salaires dans la table MEILLEURS.   
    
    On suppose qu’aucun employé n’a le même salaire qu’un autre.   
  
c) Tester le bloc avec différents cas tels que n=4 ou n=0 ou n supérieur au nombre total d’employés (14).   
  
     Vider  la table MEILLEURS après chaque test.   
  
**Exercice 5 :**  
Modifier le bloc PL/SQL fourni pour gérer les exceptions.       
  
a) Le traitement essaie de mettre à jour les villes pour des départements existants.   
  
b) Exécuter le bloc avec comme valeur 40 pour le numéro de département et Brest pour le nom de la ville.

c) Ecrire un  traitement d’exception pour l’anomalie constatée afin de passer un message à l’utilisateur lorsque le département spécifiée n’existe pas.

d) Exécuter le bloc avec comme valeur 33 pour le numéro de département et Bordeaux pour la ville.

e) Ecrire un  traitement d’exception pour  l’anomalie constatée afin de passer un message à  l’utilisateur  lorsque  la ville spécifiée est déjà référencée pour ce département

f) Exécuter le bloc avec comme valeur 40 pour le numéro de département et BREST pour la ville

**Corrigé Exercice 1 :**

A **Correcte**

B **Incorrecte : un seul identifiant par ligne**

C **Incorrecte : une valeur NOT NULL doit être initialisée**

D **Incorrecte : 1 n’est pas une valeur booléenne**

E **Incorrecte : EMP\_RECORD\_TYPE doit être déclaré**

F **Correcte**

**Corrigé Exercice 2 :**

1-a

create sequence dept\_id\_seq minvalue 50 increment by 10;

ACCEPT p\_dept\_nom PROMPT 'Entrer un nom de département : '

BEGIN

INSERT INTO dept(deptno, dname, loc)

VALUES (dept\_id\_seq.NEXTVAL, '&p\_dept\_nom', NULL);

COMMIT;

END;

/

1-b  
Entrer un nom de département : REC   
PL/SQL procedure successfully completed.

1-c

SELECT \* FROM dept WHERE dname = 'REC';

2-a

ACCEPT p\_dept\_id PROMPT 'Entrer un numéro de département : '

VARIABLE g\_mess VARCHAR2(30)

DECLARE

v\_resultat NUMBER(2);

BEGIN

DELETE FROM dept WHERE deptno = &p\_dept\_id;

v\_resultat := SQL%ROWCOUNT;

:g\_mess := TO\_CHAR(v\_resultat)||' ligne(s) supprimée(s).';

COMMIT;

END;

/

PRINT g\_mess

2-b

O lignes supprimées

ORA-02292: violation de contrainte (JP.FK\_DEPT) d'intégrité - enregistrement fils existant

**Corrigé Exercice 3 :**

1-a

ACCEPT p\_id PROMPT 'Entrer un numéro employé : '

DECLARE

v\_comm\_moy NUMBER(11,2);

v\_comm emp.comm%TYPE;

BEGIN

SELECT AVG(nvl(comm,0)) INTO v\_comm\_moy from emp;

select comm into v\_comm from emp where empno = &p\_id;

IF v\_comm < v\_comm\_moy THEN v\_comm := 1000;

ELSIF v\_comm >= v\_comm\_moy THEN v\_comm := 500;

ELSE v\_comm := 0;

END IF;

UPDATE emp SET comm = v\_comm WHERE empno = &p\_id;

COMMIT;

END;

/

2

VARIABLE g\_mess VARCHAR2(1000)

DECLARE

v\_sortie VARCHAR2(1000);

v\_nb\_modif NUMBER(2);

v\_comm emp.comm%type;

c\_peu CONSTANT VARCHAR2(1000)

:= 'Moins de 3 lignes ont été modifiées pour la région ';

cursor c\_dep is select deptno from dept;

BEGIN

FOR i IN c\_dep LOOP

IF MOD(substr(i.deptno,1,1),2) <> 0 THEN

v\_comm := 1000;

ELSE

v\_comm := 500;

END IF;

UPDATE emp SET comm = v\_comm WHERE deptno = i.deptno;

v\_nb\_modif := SQL%ROWCOUNT;

IF v\_nb\_modif < 3 THEN v\_sortie := v\_sortie||c\_peu||TO\_CHAR(i.deptno)||chr(10);

ELSE

v\_sortie := v\_sortie||TO\_CHAR(v\_nb\_modif)||

' lignes ont été modifiées pour la région '||TO\_CHAR(i.deptno)||chr(10);

END IF;

END LOOP;

:g\_mess := v\_sortie;

END;

/

PRINT g\_mess

**Corrigé Exercice 4 :**

4-a

CREATE TABLE meilleurs (nom VARCHAR2(25),salaire NUMBER(11,2));

4-b

ACCEPT p\_n PROMPT 'Entrer une valeur numérique : '

DECLARE

CURSOR emp\_cursor IS SELECT ename, sal FROM emp

WHERE sal IS NOT NULL

ORDER BY sal DESC;

emp\_record emp\_cursor%ROWTYPE;

BEGIN

OPEN emp\_cursor;

FOR i IN 1..&p\_n LOOP

FETCH emp\_cursor INTO emp\_record;

INSERT INTO meilleurs(nom, salaire)

VALUES (emp\_record.ename,emp\_record.sal);

END LOOP;

CLOSE emp\_cursor;

COMMIT;

END;

/

SELECT nom,TO\_CHAR(salaire,'fm$9,999,999') salaire FROM meilleurs;

**Corrigé Exercice 5 :**

5-a

ACCEPT p\_dept\_id PROMPT 'Numéro de département : '

ACCEPT p\_ville PROMPT 'Nom de ville : '

VARIABLE g\_mess VARCHAR2(1000)

DECLARE

v\_dep dept.deptno%TYPE;

BEGIN

UPDATE dept SET loc=UPPER('&p\_ville')

WHERE deptno = &p\_dept\_id;

:g\_mess := 'Le département : '||TO\_CHAR(&p\_dept\_id)||

' est affecté à la ville '||TO\_CHAR('&p\_ville');

-- COMMIT;

END;

/

PRINT g\_mess

5-c

ACCEPT p\_dept\_id PROMPT 'Numéro de département : '

ACCEPT p\_ville PROMPT 'Nom de ville : '

VARIABLE g\_mess VARCHAR2(1000)

DECLARE

v\_dep dept.deptno%TYPE;

BEGIN

select deptno into v\_dep from dept where deptno=&p\_dept\_id;

UPDATE dept SET loc=UPPER('&p\_ville')

WHERE deptno = &p\_dept\_id;

:g\_mess := 'Le département : '||TO\_CHAR(&p\_dept\_id)||

' est affecté à la ville '||TO\_CHAR('&p\_ville');

COMMIT;

EXCEPTION

when no\_data\_found then

:g\_mess := 'Le département '||TO\_CHAR(&p\_dept\_id)||

' n''existe pas ';

END;

/

PRINT g\_mess

5-e

ACCEPT p\_dept\_id PROMPT 'Numéro de département : '

ACCEPT p\_ville PROMPT 'Nom de ville : '

VARIABLE g\_mess VARCHAR2(1000)

DECLARE

v\_dep dept.deptno%TYPE;

v\_ville dept.loc%type;

ville\_idem exception;

BEGIN

select deptno into v\_dep from dept where deptno=&p\_dept\_id;

select loc into v\_ville from dept where deptno=&p\_dept\_id;

if v\_ville = TO\_CHAR('&p\_ville') then

raise ville\_idem;

end if;

UPDATE dept SET loc=UPPER('&p\_ville')

WHERE deptno = &p\_dept\_id;

:g\_mess := 'Le département : '||TO\_CHAR(&p\_dept\_id)||

' est affecté à la ville '||TO\_CHAR('&p\_ville');

COMMIT;

EXCEPTION

when no\_data\_found then

:g\_mess := 'Le département '||TO\_CHAR(&p\_dept\_id)||

' n''existe pas ';

when ville\_idem then

:g\_mess := 'Le département '||TO\_CHAR(&p\_dept\_id)||

' est déjà dans la ville '||TO\_CHAR('&p\_ville');

END;

/

PRINT g\_mess