Série 3

/\*Laisser la ligne suivante : sert pour l'affichage de toutes les commandes\*/

SET SERVEROUTPUT ON;

/\* Ex1 \*/

DECLARE

v\_nb\_emp NUMBER(3);

BEGIN

SELECT COUNT(empno) INTO v\_nb\_emp FROM emp;

DBMS\_OUTPUT.PUT\_LINE('Le nombre total d''employés est : '||v\_nb\_emp);

END;

/\* Ex2 \*/

/\* a \*/

ACCEPT p\_nb PROMPT 'Entrer le numéro d''employé'

DECLARE

sal\_moy\_emp emp.sal%TYPE;

v\_job emp.job%TYPE;

BEGIN

SELECT job INTO v\_job FROM emp WHERE empno=&p\_nb;

SELECT avg(sal) INTO sal\_moy\_emp FROM emp WHERE job=v\_job;

DBMS\_OUTPUT.PUT\_LINE('Moyenne des salaires des '||v\_job||': '||sal\_moy\_emp||'€');

COMMIT;

END;

/\* b\*/

ACCEPT p\_nb PROMPT 'Entrer le numéro d''employé'

DECLARE

v\_moy\_sal emp.sal%TYPE;

v\_job emp.job%TYPE;

v\_sal emp.sal%TYPE;

BEGIN

SELECT job INTO v\_job FROM emp WHERE empno=&p\_nb;

SELECT avg(sal) INTO v\_moy\_sal FROM emp WHERE job=v\_job;

SELECT sal INTO v\_sal FROM emp WHERE empno=&p\_nb;

IF v\_sal<v\_moy\_sal THEN

UPDATE emp

SET sal=v\_moy\_sal

WHERE empno=&p\_nb;

ELSE

UPDATE emp

SET sal=v\_sal\*1.1

WHERE empno=&p\_nb;

END IF;

COMMIT;

END;

/\* Ex3 \*/

SET SERVEROUTPUT ON

SET VERIFY ON

Accept nb prompt ‘Entrez un nombre :’

DECLARE

e\_negatif exception;

e\_to\_big exception;

v\_fact number :1;

v\_nb\_saisi number(3);

BEGIN

V\_nb\_saisi := to\_number(&nb);

IF v\_nb\_saisi <0

THEN raise e\_negatif

ELSIF v\_nb\_saisi > 33

THEN

/\* Ex4 \*/

CREATE TABLE amorti (

Durée NUMBER(2) CONSTRAINT amorti\_pk PRIMARY KEY,

Somme\_due NUMBER(8),

Remb\_An NUMBER(6),

Intérêt NUMBER(5),

Annuité NUMBER(6) );

SET serveroutput on

SET verify off

ACCEPT somme prompt 'Entrez la somme empruntée :'

ACCEPT duree prompt 'Entrez la durée de l''emprunt en années :'

ACCEPT taux prompt 'Entrez le taux d''intérêt en % :'

DECLARE

v\_remb\_an amorti.somme\_due%type := &somme/&duree;

v\_interet amorti.intérêt%type;

v\_annuite amorti.annuité%type;

v\_som amorti.somme\_due%type;

BEGIN

v\_som := &somme;

FOR i IN 1..&duree LOOP

v\_interet := (v\_som \* &taux)/100;

v\_annuite := v\_remb\_an + v\_interet;

INSERT INTO amorti

VALUES (i, v\_som, v\_remb\_an, v\_interet, v\_annuite);

v\_som := v\_som - v\_remb\_an;

END LOOP

COMMIT;

END;

SELECT \* FROM amorti;

DROP TABLE amorti;

SET VERIFY ON

/\* Ex5.a-b \*/

SET SERVEROUTPUT ON;

ACCEPT p\_nb PROMPT 'Numéro à multiplier';

DECLARE

TYPE multi IS TABLE OF INTEGER(3) INDEX BY BINARY\_INTEGER;

v\_multi multi;

BEGIN

FOR i IN 1..9 LOOP

v\_multi(i) := &p\_nb \* i;

DBMS\_OUTPUT.PUT\_LINE(i || '\*&p\_nb=' || v\_multi(i));

END LOOP;

END;

/\* Ex6 \*/

--pas encore fonctionnel

CREATE TABLE amorti (

Durée NUMBER(2) CONSTRAINT amorti\_pk PRIMARY KEY,

Somme\_due NUMBER(8),

Remb\_An NUMBER(6),

Intérêt NUMBER(5),

Annuité NUMBER(6) );

SET serveroutput on

SET verify off

ACCEPT somme prompt 'Entrez la somme empruntée :'

ACCEPT duree prompt 'Entrez la durée de l''emprunt en années :'

ACCEPT taux prompt 'Entrez le taux d''intérêt en % :'

DECLARE

v\_remb\_an amorti.somme\_due%type := &somme/&duree;

v\_interet amorti.intérêt%type;

v\_annuite amorti.annuité%type;

v\_som amorti.somme\_due%type;

BEGIN

v\_som := &somme;

FOR i IN 1..&duree LOOP

v\_interet := (v\_som \* &taux)/100;

v\_annuite := v\_remb\_an + v\_interet;

INSERT INTO amorti

VALUES (i, v\_som, v\_remb\_an, v\_interet, v\_annuite);

v\_som := v\_som - v\_remb\_an;

END LOOP

COMMIT;

--

DBSM\_OUTPUT.PUT\_LINE(rpad('Duree', 7) || rpad('Intérêt', 10) || 'Annuité');

FOR enr\_amorti IN (SELECT \* FROM amorti) LOOP

DBSM\_OUTPUT.PUT\_LINE(rpad(enr\_amorti.durée, 7) || rpad(enr\_amorti.intérêt, 10) || enr\_amorti.annuité);

END LOOP;

--

END;

SELECT \* FROM amorti;

DROP TABLE amorti;

SET VERIFY ON

/\* Ex7.a\*/

SET SERVEROUTPUT ON

ALTER TABLE dept ADD (budget NUMBER(10,2) default 0);

DECLARE

CURSOR curseur\_budget IS

SELECT deptno, SUM(sal) budget FROM emp

GROUP BY deptno;

BEGIN

FOR enr IN curseur\_budget LOOP

DBMS\_OUTPUT\_LINE('deptno' ||enr.deptno||'budjet'||enr.budget);

update dept

set budget = enr.budget

where deptno = enr.deptno;

END LOOP;

COMMIT;

END;

/\* Ex7.b\*/

Solution SQL :

UPDATE emp

SET sal = sal\*1.1

WHERE deptno IN (

SELECT deptno

FROM dept

WHERE UPPER(loc)

IN ('DALLAS', 'NEW YORK'));

Solution PL/SQL :

DECLARE

CURSOR curseur\_salaire IS

SELECT sal

FROM emp

WHERE deptno IN (

SELECT deptno

FROM dept

WHERE UPPER(loc) IN ('DALLAS', 'NEW YORK'))

FOR UPDATE OF SAL;

BEGIN

FOR i IN curseur\_salaire LOOP

UPDATE emp

SET sal = i.sal\*1.1

WHERE current of curseur\_salaire;

END LOOP;

COMMIT;

END;

/\* Ex7.c \*/

Solution SQL :

ACCEPT nbre\_salaire PROMPT 'Nombre de salaire?'

SELECT empno, ename, sal

FROM (

SELECT empno, ename, sal

FROM emp

ORDER BY sal desc)

WHERE rownum <= &nbre\_salaire;

Solution PL/SQL :

ACCEPT nbre\_salaire PROMPT 'Nombre de salaire?'

DECLARE

CURSOR curseur\_salaire\_max IS

SELECT empno, ename, sal

FROM emp

ORDER BY sal desc;

enregistre curseur\_salaire\_max%ROWTYPE;

BEGIN

OPEN curseur\_salaire\_max;

FETCH curseur\_salaire\_max INTO enregistre;

WHILE curseur\_salaire\_max%ROWCOUNT <= &nbre\_salaire

AND curseur\_salaire\_max%FOUND LOOP

DBMS\_OUTPUT.PUT\_LINE('NUMERO EMP:'||enregistre.empno||' NOM:'||enregistre.ename||' SALAIRE:1'||enregistre.sal);

FETCH curseur\_salaire\_max INTO enregistre;

END LOOP;

IF curseur\_salaire\_max%ROWCOUNT<&enregistre

THEN DBMS\_OUTPUT.PUT\_LINE('On ne peut afficher que '||curseur\_salaire\_max%ROWCOUNT||' employés');

END IF;

CLOSE curseur\_salaire\_max;

END;