

## CORRECTIONS BD

### SERIE 2

1. select tablespace\_name, max\_blocks Quota from dba\_ts\_quotas where username='P1303195';

R	TABLESPACE_NAME	R	QUOTA
1	A		640

2. select privilege "Privilèges système" from dba\_sys\_privs where grantee = 'P1303195';

R	Privilèges système
1	CREATE SEQUENCE
2	UNLIMITED TABLESPACE
3	DEBUG CONNECT SESSION
4	CREATE TYPE
5	CREATE PROCEDURE
6	CREATE VIEW
7	CREATE ANY DIRECTORY
8	CREATE TRIGGER
9	DEBUG ANY PROCEDURE
10	SELECT ANY DICTIONARY
11	DROP PUBLIC SYNONYM
12	CREATE PUBLIC SYNONYM

3. select \* from dba\_tab\_privs where grantee = 'PUBLIC' and OWNER='SCOTT';

R	GRANTEE	R	OWNER	R	TABLE_NAME	R	GRANTOR	R	PRIVILEGE	R	GRANTABLE	R	HIERARCHY
1	PUBLIC		SCOTT		BONUS		SCOTT		SELECT		NO		NO
2	PUBLIC		SCOTT		SALGRADE		SCOTT		SELECT		NO		NO
3	PUBLIC		SCOTT		DEPT		SCOTT		SELECT		NO		NO
4	PUBLIC		SCOTT		EMP		SCOTT		SELECT		NO		NO

4. select \* from dba\_role\_privs where grantee = 'P1303195';

R	GRANTEE	R	GRANTED_ROLE	R	ADMIN_OPTION	R	DEFAULT_ROLE
1	P1303195		CONNECT		NO		YES
2	P1303195		RESOURCE		NO		YES

5. select limit "Nombre de minutes" from dba\_profiles where resource\_name = 'IDLE\_TIME' and profile = (select profile from dba\_users where username='P1303195');

R	Nombre de minutes
1	UNLIMITED

6. select count(\*) "Nombre d'utilisateurs" from dba\_users;

R	Nombre d'utilisateurs
	437

## SERIE 3

### Exercice 1

1.

```
set serveroutput on
DROP TABLE EMP;
DROP TABLE DEPT;
CREATE TABLE emp AS SELECT * FROM scott.emp;
CREATE TABLE dept AS SELECT * FROM SCOTT.dept;
```

```
DECLARE
    v_nb_emp number(3);
BEGIN
    select count(empno) into v_nb_emp from emp;
    dbms_output.put_line('Nombre d"employés ' || v_nb_emp);
END;
```

2.

```
ACCEPT num_emp prompt 'Entrez un numéro d'employé : '
-- suppression commentaires à l'exécution
SET verify OFF
```

```
DECLARE
    v_moy emp.sal%type;
    v_emp_sal emp.sal%type;
    v_emp_num emp.empno%type;

BEGIN
    /*génération de VALUE_ERROR (expression). Par exemple saisi de 'a' avec les quotes
    v_emp_num := to_number(&num_emp);
    génération de INVALID_NUMBER (ordre SQL). Par exemple saisi de 'a' avec les quotes */
    -- Question a) et b)
    SELECT sal INTO v_emp_sal FROM emp WHERE empno=&num_emp;
    DBMS_OUTPUT.put_line('Ancien salaire : ' || v_emp_sal);
    SELECT AVG(sal) INTO v_moy
    FROM emp
    WHERE job = (SELECT job FROM emp WHERE empno= &num_emp);
    DBMS_OUTPUT.put_line('Moyenne des salaires ' || v_moy);
    IF v_emp_sal < v_moy THEN
        v_emp_sal := v_moy;
    ELSE
        v_emp_sal := v_emp_sal*1.1;
    END IF;
    DBMS_OUTPUT.put_line('Nouveau salaire : ' || v_emp_sal);
    UPDATE emp SET sal = v_emp_sal WHERE empno=&num_emp;
    COMMIT;

EXCEPTION
/*WHEN VALUE_ERROR THEN
    RAISE_APPLICATION_ERROR(-20100,'ERREUR DE TYPE VALUE_ERROR : '|| SQLERRM || ' CODE
    ERREUR' || SQLCODE);*/
WHEN INVALID_NUMBER THEN
    RAISE_APPLICATION_ERROR(-20101,'ERREUR DE TYPE INVALID_NUMBER : '|| SQLERRM || ' CODE
    ERREUR' || SQLCODE);
WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.put_line('EMPLOYE INCONNU ' || SQLERRM);
    RAISE;
WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20111,'Erreur: '|| SQLERRM || ' Code erreur' || SQLCODE);
END;
```

**Test INVALID\_NUMBER Saisie: 'a'**

Rapport d'erreur :

ORA-20101: ERREUR DE TYPE INVALID\_NUMBER : ORA-01722: Nombre non valide CODE ERREUR-1722  
ORA-06512: à ligne 31

### Test VALUE\_ERROR Saisie : 'a'

Rapport d'erreur :

ORA-20100: ERREUR DE TYPE : ORA-06502: PL/SQL : erreur numérique ou erreur sur une valeur: erreur de conversion des caractères en chiffres CODE ERREUR-6502

ORA-06512: à ligne 29

### Test NO\_DATA\_FOUND Saisie : 5

EMPLOYE INCONNU ORA-01403: aucune donnée trouvée

### Test Saisie : 7369

```
bloc anonyme terminé
Ancien salaire : 800
Moyenne des salaires 1016,67
Nouveau salaire : 1016,67
```

### Saisie : 7499

```
bloc anonyme terminé
Ancien salaire : 1600
Moyenne des salaires 1400
Nouveau salaire : 1760
```

### --1-c

```
UPDATE emp x
set sal = (select decode(least(x.sal, avg(sal)),
                        avg(sal),x.sal*1.1,
                        avg(sal))
          from emp where job=x.job)
where empno=&num;
commit;
```

ou

```
UPDATE emp e SET sal = (CASE WHEN sal >= (SELECT AVG(sal) FROM emp WHERE job=e.job)
THEN sal*1.1 ELSE (SELECT AVG(sal) FROM emp WHERE job=e.job) END) WHERE empno=&num;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17/12/80	800	(null)	20
7499	ALLEN	SALESMAN	7698	20/02/81	1600	300	30
7521	WARD	SALESMAN	7698	22/02/81	1250	500	30
7566	JONES	MANAGER	7839	02/04/81	2975	(null)	20
7654	MARTIN	SALESMAN	7698	28/09/81	1250	1400	30
7698	BLAKE	MANAGER	7839	01/05/81	2850	(null)	30
7782	CLARK	MANAGER	7839	09/06/81	2450	(null)	10
7839	KING	PRESIDENT	(null)	17/11/81	5000	(null)	10
7844	TURNER	SALESMAN	7698	08/09/81	1500	0	30
7900	JAMES	CLERK	7698	03/12/81	950	(null)	30
7902	FORD	ANALYST	7566	03/12/81	3000	(null)	20
7934	MILLER	CLERK	7782	23/01/82	1300	(null)	10

### 3.

```
set serveroutput on
set verify off
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 > 83 THEN
        raise e_to_big;
    END IF;
    FOR i in 1.. v_nb_saisi LOOP
        v_fact := v_fact*i;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Valeur de la factorielle : '|| v_fact);
EXCEPTION
WHEN VALUE_ERROR THEN
    RAISE_APPLICATION_ERROR(-20110,'Erreur de type : '|| SQLERRM || ' Code erreur ' || SQLCODE);
WHEN e_negatif THEN
    RAISE_APPLICATION_ERROR(-20111,'Erreur - le nombre saisi ne doit pas être négatif '|| SQLERRM || ' Code
    erreur ' || SQLCODE);
WHEN e_to_big THEN
    RAISE_APPLICATION_ERROR(-20112,'Erreur - le nombre saisi doit être inférieur à 84 '|| SQLERRM || ' Code
    erreur ' || SQLCODE);
WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20113,'Erreur: '|| SQLERRM || ' Code erreur ' || SQLCODE);
END;
```

#### Tests :

Valeur saisie : -1

```
Rapport d'erreur -
ORA-20111: Erreur - le nombre saisi ne doit pas être négatif User-Defined Exception Code erreur 1
ORA-06512: à ligne 21
```

Valeur saisie : 84

```
Rapport d'erreur -
ORA-20112: Erreur - le nombre saisi doit être inférieur à 84 User-Defined Exception Code erreur 1
ORA-06512: à ligne 23
```

Valeur saisie : 5

```
bloc anonyme terminé
Valeur de la factorielle : 120
```

#### 4.

```
CREATE TABLE amorti
(Duree number(2) CONSTRAINT amorti_pk primary key, Somme_due number(10,2), Remb_An number(8,2), Intérêt
number(8,2), Annuité number(8,2));
SET serveroutput on
SET verify off
```

```
ACCEPT somme prompt "Entrez la somme empruntée : "
ACCEPT duree prompt "Entrez la durée de l'emprunt : "
ACCEPT taux prompt "Entrez le taux d'intérêt : "
```

```
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
```

bloc anonyme terminé				
DUREE	SOMME_DUE	REMB_AN	INTÉRÊT	ANNUITÉ
1	12000	4000	480	4480
2	8000	4000	320	4320
3	4000	4000	160	4160

#### 5.

```
SET serveroutput on
SET verify off
ACCEPT num PROMPT 'Table de multiplication de : '
```

```
DECLARE
TYPE tabch_type IS TABLE OF number(2)
INDEX BY BINARY_INTEGER; -- obligatoire
tabch tabch_type;
```

```
BEGIN
    DBMS_OUTPUT.PUT_LINE('col1'||chr(9)||'col2');
    FOR i IN 1..10 LOOP
        tabch(i) := &num*i;
        DBMS_OUTPUT.PUT_LINE(TO_CHAR(tabch.last,'99') ||chr(9)||
        TO_CHAR(tabch(i),'99') );
    END LOOP;

END;
```

&num=8

col1	col2
1	8
2	16
3	24
4	32
5	40
6	48
7	56
8	64
9	72
10	80

## 6.

```
DROP TABLE AMORTI;
CREATE TABLE AMORTI
(Duree number(2) CONSTRAINT amorti_pk primary key, Somme_due number(10,2), Remb_An number(8,2), Intérêt
number(8,2), Annuité number(8,2));
SET serveroutput on
SET verify off
```

```
ACCEPT somme prompt "Entrez la somme empruntée : "
ACCEPT duree prompt "Entrez la durée de l'emprunt : "
ACCEPT taux prompt "Entrez le taux d'intérêt : "
```

```
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;
    DBMS_OUTPUT.PUT_LINE(rpad('Duree',7)||rpad('Intérêt',10)||'Annuité');
    FOR enr_amorti IN (SELECT * FROM amorti) LOOP
        DBMS_OUTPUT.PUT_LINE(rpad(enr_amorti.duree,7)||rpad(enr_amorti.intérêt,10)||enr_amorti.annuité);
    END LOOP;
END;
```

```
bloc anonyme terminé
Duree  Intérêt  Annuité
1      480     4480
2      320     4320
3      160     4160
```

## 7.

### 7 a Solution SQL

```
create table dept as select * from scott.dept;
create table emp as select * from scott.emp;
```

```
ALTER TABLE dept ADD (budget number(10,2) default 0);
/*default 0 : à l'ajout de la colonne 0 dans la colonne budget pour toutes les lignes. Pour les insertions futures de
départements, 0 pour le budget.
*/
```

```
update dept
set budget = (select nvl(sum(sal),0) from emp
where deptno = dept.deptno);
```

```
commit;
```

```
select * from dept;
drop table dept;
drop table emp;
```

```
table DEPT modifié(e).
4 lignes mis à jour.
validé (commit).
  DEPTNO DNAME          LOC          BUDGET
-----
      10 ACCOUNTING    NEW YORK      8750
      20 RESEARCH      DALLAS       6775
      30 SALES         CHICAGO      9400
      40 OPERATIONS    BOSTON         0

table DEPT supprimé(e).
table EMP supprimé(e).
```

### 7 a Solution PL/SQL

```
set serveroutput on
drop table dept;
drop table emp;
create table dept as select * from scott.dept;
create table emp as select * from scott.emp;
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;
    -- dept 40 non atteint, default 0 nécessaire
```

```
BEGIN
```

```
    FOR enr in curseur_budget LOOP
        DBMS_OUTPUT.PUT_LINE('Valeur de deptno '||enr.deptno||' Valeur de budget '|| enr.budget);
        update dept
        set budget = enr.budget
        where deptno = enr.deptno;
    END LOOP;
    COMMIT;
```

```
END;
```

```
Valeur de deptno 30 Valeur de budget 9400
Valeur de deptno 20 Valeur de budget 10875
Valeur de deptno 10 Valeur de budget 8750
```

## 7 b Solution SQL

```
drop table dept;
drop table emp;
create table dept as select * from scott.dept;
create table emp as select * from scott.emp;
```

```
UPDATE emp
SET sal = sal*1.1
WHERE deptno IN (select deptno from dept where upper(loc) IN ('DALLAS','NEW YORK'));
COMMIT;
```

### column ename heading 'Nom|Salarié'

```
select ename, loc localité, sal salaire
from emp JOIN dept ON emp.deptno=dept.deptno
where upper(loc) IN ('DALLAS','NEW YORK');
```

Nom		
Salarié	LOCALITÉ	SALAIRE
SMITH	DALLAS	880
JONES	DALLAS	3272,5
CLARK	NEW YORK	2695
SCOTT	DALLAS	3300
KING	NEW YORK	5500

## 7 b Solution PL/SQL

```
set serveroutput on
drop table dept;
drop table emp;
create table dept as select * from scott.dept;
create table emp as select * from scott.emp;
```

```
DECLARE
    CURSOR curseur_sal IS
        select sal*1.1 nvscal from emp
        where deptno IN (select deptno from dept where
            upper(loc) IN ('DALLAS','NEW YORK'))
    FOR UPDATE OF SAL; /*verrouillage des lignes de EMP correspondant aux employés travaillant à Dallas ou à
    New York*/
```

```
BEGIN
    FOR enr in curseur_sal LOOP
        update emp
        set sal = enr.nvscal
        where current of curseur_sal;
    END LOOP;
    COMMIT;
```

```
END;
/
select ename Nom, loc localité, sal salaire
from emp JOIN dept ON emp.deptno=dept.deptno
where upper(loc) IN ('DALLAS','NEW YORK');
```

NOM	LOCALITÉ	SALAIRE
SMITH	DALLAS	880
JONES	DALLAS	3272,5
CLARK	NEW YORK	2695
SCOTT	DALLAS	3300
KING	NEW YORK	5500
ADAMS	DALLAS	1210
FORD	DALLAS	3300
MILLER	NEW YORK	1430



## 7 c Solution SQL

```
drop table dept;
drop table emp;
create table dept as select * from scott.dept;
create table emp as select * from scott.emp;
```

ACCEPT num PROMPT 'Nombre maximum de salaires à afficher : '

```
select empno, ename, sal from (select empno, ename, sal
from emp order by sal desc)
where rownum <= &num;
```

/\* avec num=4 \*/

EMPNO	ENAME	SAL
7839	KING	5000
7788	SCOTT	3000
7902	FORD	3000
7566	JONES	2975

## 7 c Solution PL/SQL

set verify off

ACCEPT num PROMPT 'Nombre maximum de salaires à afficher : '

DECLARE

```
CURSOR curseur_max_sal IS
select empno, ename, sal from emp
order by sal DESC;
v_enrg curseur_max_sal%ROWTYPE;
```

BEGIN

```
OPEN curseur_max_sal;
FETCH curseur_max_sal INTO v_enrg;
WHILE curseur_max_sal%ROWCOUNT <= &num AND curseur_max_sal%FOUND LOOP
    DBMS_OUTPUT.PUT_LINE('N° EMP ' || v_enrg.empno || ' NOM ' || v_enrg.ename || ' Salaire ' || v_enrg.sal);
    FETCH curseur_max_sal INTO v_enrg;
END LOOP;
IF curseur_max_sal%ROWCOUNT < &num THEN
    DBMS_OUTPUT.PUT_LINE ('Nous ne pouvons afficher que ' || curseur_max_sal%ROWCOUNT || ' employé(s)');
END IF;
CLOSE curseur_max_sal;
END;
```

/\* Test avec 16 \*/

```
bloc anonyme terminé
N° EMP 7839 NOM KING Salaire 5000
N° EMP 7902 NOM FORD Salaire 3000
N° EMP 7566 NOM JONES Salaire 2975
N° EMP 7698 NOM BLAKE Salaire 2850
N° EMP 7782 NOM CLARK Salaire 2450
N° EMP 7499 NOM ALLEN Salaire 1600
N° EMP 7844 NOM TURNER Salaire 1500
N° EMP 7934 NOM MILLER Salaire 1300
N° EMP 7654 NOM MARTIN Salaire 1250
N° EMP 7521 NOM WARD Salaire 1250
N° EMP 7900 NOM JAMES Salaire 950
N° EMP 7369 NOM SMITH Salaire 800
Nous ne pouvons afficher que 12 employé(s)
```