SERIE 5

```
create table jobs as select * from hr.jobs;
create table employees as select * from hr.employees;
create table departments as select * from hr.departments;
```

-- 1

```
CREATE OR REPLACE PROCEDURE ajout_job (p_id jobs.job_id%type, p_intitule jobs.job_title%type)
IS
BEGIN
insert into jobs (job_id,job_title) values (p_id, p_intitule);
commit;
EXCEPTION
WHEN others THEN
raise_application_error(-20000,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END ajout_job;
```

Tests :

```
execute ajout_job('IUT DI', 'Informatique');
```

19	PR_REP	Public Relations Representative	4500	10500
20	IUT	PROF	(null)	(null)
21	IUT DI	Informatique	(null)	(null)

-- 2

```
CREATE OR REPLACE PROCEDURE modif_job (p_id jobs.job_id%type, p_nouveauTitre
jobs.job_title%type)
IS
e_aucune_maj exception;
BEGIN
UPDATE jobs SET job_title=p_nouveauTitre WHERE upper(job_id)=upper(p_id);
IF SQL%ROWCOUNT = 0 THEN
RAISE e_aucune_maj;
END IF;
commit; -- à placer après la conditionnelle
EXCEPTION
WHEN e_aucune_maj THEN
raise_application_error(-20000,'Aucune mise à jour n''a eu lieu');
WHEN others THEN
raise_application_error(-20001,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END modif_job;
```

Tests :

```
exec modif_job('IUT DI', 'Dept Informatique');
```

19	PR_REP	Public Relations Representative	4500	10500
20	IUT	PROF	(null)	(null)
21	IUT DI	Dept Informatique	(null)	(null)

```
exec modif_job('IUT DP', 'Dept Informatique');

Erreur commençant à la ligne 1 de la commande :
exec modif_job('IUT DP', 'Dept Informatique')
Rapport d'erreur :
ORA-20000: Aucune mise à jour n'a eu lieu
ORA-06512: à "TESTLP.MODIF_JOB", ligne 12
```

-- 3

```
CREATE OR REPLACE PROCEDURE listemp
IS
BEGIN
DBMS_OUTPUT.PUT_LINE(rpad('Nom employé',30)  || 'Nom manager'|| chr(10) || rpad('-',50,'-'));
FOR nom_enrg IN (select e1.last_name Nom_emp, e1.first_name prenom_emp, e2.first_name
prenom_mgr, e2.last_name Nom_mgr -- alias obligatoire
from employees e1 LEFT JOIN employees e2 ON  e1.manager_id=e2.employee_id)
LOOP
DBMS_OUTPUT.PUT_LINE( rpad(nom_enrg.Nom_emp||' '||nom_enrg.prenom_emp,30)  ||
nom_enrg.prenom_mgr || ' ' ||nom_enrg.Nom_mgr);
END LOOP;
EXCEPTION
WHEN others THEN
raise_application_error(-20000,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Test :
set serveroutput on
exec listemp

```
Nom employé                    Nom manager
--------------------------------------------------
Fay Pat                        Michael Hartstein
Gietz William                  Shelley Higgins
Zlotkey Eleni                  Steven King
Cambrault Gerald               Steven King
Errazuriz Alberto              Steven King
Partners Karen                 Steven King
Russell John                   Steven King
Mourgos Kevin                  Steven King
Vollman Shanta                 Steven King
```

-- 4

```
CREATE OR REPLACE PROCEDURE gagneplus(p_nom_emp employees.last_name%type,
p_prenom_emp employees.first_name%type)
IS
v_sal employees.salary%type;
BEGIN
-- test existence de l'employé
select  salary*(1+nvl(COMMISSION_PCT,0)) into v_sal from employees where upper(last_name) =
upper(p_nom_emp) and upper(first_name)=upper(p_prenom_emp);
DBMS_OUTPUT.PUT_LINE('Prénom et Nom Employé' || chr(10)|| lpad('-',20,'-'));
FOR enrg IN (select first_name, last_name
from employees
where  salary*(1+nvl(COMMISSION_PCT,0)) > v_sal)
LOOP
DBMS_OUTPUT.PUT_LINE( enrg.first_name || ' ' ||enrg.last_name);
END LOOP;
EXCEPTION
WHEN NO_DATA_FOUND THEN
raise_application_error(-20000,'Erreur l''employé n''existe pas');
WHEN others THEN
raise_application_error(-20001,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Tests :
exec gagneplus('Geenberg','Nancy');

```
Erreur commençant à la ligne 6 de la commande :
exec gagneplus('Geenberg','Nancy')
Rapport d'erreur :
ORA-20000: Erreur l'employé n'existe pas
ORA-06512: à "TESTLP.GAGNEPLUS", ligne 19
ORA-06512: à ligne 1
20000. 00000 -  "%s"
*Cause:    The stored procedure 'raise_application_error'
```

exec gagneplus('Greenberg','Nancy');

```
Prénom et Nom Employé
--------------------
Michael Hartstein
Steven King
Neena Kochhar
Lex De Haan
John Russell
Karen Partners
Alberto Errazuriz
Gerald Cambrault
Eleni Zlotkey
Peter Tucker
Janette King
Patrick Sully
Allan McEwen
Clara Vishney
Lisa Ozer
Ellen Abel
```

exec gagneplus('King','Steven');

```
Prénom et Nom Employé
--------------------
```

-- 5

```
CREATE OR REPLACE PROCEDURE jobgagneplus(p_id_job employees.employee_id%type, p_id_sal
employees.employee_id%type)
IS
v_sal employees.salary%type;
v_job_id employees.job_id%type;
BEGIN
-- test existence de l'employé
select  salary*(1+nvl(COMMISSION_PCT,0)) into v_sal  from employees where employee_id = p_id_sal;
select job_id into v_job_id from employees where employee_id=p_id_job;
DBMS_OUTPUT.PUT_LINE('Nom Employé gagnant plus que '|| p_id_sal || ' ou ayant le même travail que
'||p_id_job  || chr(10)||  lpad('-',50,'-'));
FOR enrg IN (select first_name, last_name
from employees
where  salary*(1+nvl(COMMISSION_PCT,0)) > v_sal OR job_id= v_job_id AND employee_id != p_id_job)
LOOP
DBMS_OUTPUT.PUT_LINE( enrg.first_name || ' ' || enrg.last_name);
END LOOP;
EXCEPTION
WHEN NO_DATA_FOUND THEN
raise_application_error(-20000,'Erreur l''employé n''existe pas');
WHEN others THEN
raise_application_error(-20001,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Test :
exec jobgagneplus(149,174);

```
Nom Employé gagnant plus que 174 ou ayant le même travail que 149
------------------------------------------------
Steven King
Neena Kochhar
Lex De Haan
John Russell
Karen Partners
Alberto Errazuriz
Gerald Cambrault
Lisa Ozer
```

exec jobgagneplus(14,174);

```
Erreur commençant à la ligne 9 de la commande :
exec jobgagneplus(14,174)
Rapport d'erreur :
ORA-20000: Erreur l'employé n'existe pas
ORA-06512: à "TESTLP.JOBGAGNEPLUS", ligne 18
ORA-06512: à ligne 1
20000. 00000 -  "%s"
*Cause:    The stored procedure 'raise application error'
```

-- 6

```
CREATE OR REPLACE procedure nsalaires(p_n number) IS
BEGIN
DBMS_OUTPUT.PUT_LINE('Employés percevant les plus gros salaires'|| chr(10) || lpad('-',40,'-'));
FOR enrg IN (SELECT first_name prenom, last_name nom from (SELECT first_name, last_name from
employees order by salary desc) where rownum <= p_n) LOOP
DBMS_OUTPUT.PUT_LINE(enrg.prenom || ' ' || enrg.nom);
END LOOP;
EXCEPTION
WHEN others THEN
raise_application_error(-20000,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Test :
exec nsalaires(5);

```
Employés percevant les plus gros salaires
----------------------------------------
Steven King
Neena Kochhar
Lex De Haan
John Russell
Karen Partners
```

-- 7

```
CREATE OR REPLACE PROCEDURE DEPT_SANS_EMP
IS
BEGIN
DBMS_OUTPUT.PUT_LINE('Départements sans employé'|| chr(10) || lpad('-',26,'-'));
FOR enrg IN (select department_name from departments where department_id not in (select
distinct(department_id) from employees where department_id is not null) order by 1) LOOP
DBMS_OUTPUT.PUT_LINE( enrg.department_name);
END LOOP;
EXCEPTION
WHEN others THEN
raise_application_error(-20000,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Test :
exec dept_sans_emp

```
Départements sans employé
-------------------------
Benefits
Construction
Contracting
Control And Credit
Corporate Tax
Government Sales
IT Helpdesk
IT Support
Manufacturing
NOC
Operations
Payroll
Recruiting
Retail Sales
```

-- 8

```
CREATE OR REPLACE PROCEDURE hierarchie (p_num_emp employees.employee_id%type)
IS
BEGIN
DBMS_OUTPUT.PUT_LINE('Hierarchie');
FOR enrg IN (select first_name, last_name, LEVEL from employees
connect by prior employee_id=manager_id
start with manager_id=p_num_emp) LOOP
DBMS_OUTPUT.PUT_LINE( rpad(' ', enrg.level+1,'*') ||enrg.LEVEL ||'-'|| enrg.first_name||' '
||enrg.last_name);
END LOOP;
EXCEPTION
WHEN others THEN
raise_application_error(-20000,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Test :
exec hierarchie(100)

```
Hierarchie
 *1-Neena Kochhar
 **2-Nancy Greenberg
 ***3-Daniel Faviet
 ***3-John Chen
 ***3-Ismael Sciarra
 ***3-Jose Manuel Urman
 ***3-Luis Popp
 **2-Jennifer Whalen
 **2-Susan Mavris
 **2-Hermann Baer
 **2-Shelley Higgins
 ***3-William Gietz
 *1-Lex De Haan
 **2-Alexander Hunold
 ***3-Bruce Ernst
```

-- 9

```
CREATE VIEW sal_par_dept as
select d.department_id, nvl(sum(salary),0) somme from employees e right join departments d on
d.department_id=e.department_id
group by d.department_id;
```

```
CREATE OR REPLACE PROCEDURE dept_somme_sal(p_somme_sal employees.salary%type)
IS
BEGIN
DBMS_OUTPUT.PUT_LINE('Département où somme des salaires > ' || p_somme_sal || chr(10) || lpad('-
',42,'-') );
FOR enrg IN (select department_id, somme  from sal_par_dept
where somme > p_somme_sal) LOOP
DBMS_OUTPUT.PUT_LINE( rpad(enrg.department_id,10) || enrg.somme);
END LOOP;
EXCEPTION
WHEN others THEN
raise_application_error(-20000,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Test :
exec dept_somme_sal(50000);

```
Département où somme des salaires > 50000
------------------------------------------
100       51608
90        58000
50        156400
80        304500
```

-- 10

```
CREATE OR REPLACE PROCEDURE rapport_employes
IS
BEGIN
DBMS_OUTPUT.PUT_LINE('Employés de salaire supérieur au salaire moyen de leur département' ||
chr(10) || lpad('-',50,'-'));
FOR enrg IN (select first_name, last_name FROM EMPLOYEES e where SALARY > (select avg(salary)
from employees where department_id=e.department_id)) LOOP
DBMS_OUTPUT.PUT_LINE( enrg.first_name || ' ' || enrg.last_name);
END LOOP;
EXCEPTION
WHEN others THEN
raise_application_error(-20000,'Erreur ' || SQLERRM || 'de numéro :' || SQLCODE);
END;
```

Test :
exec rapport_employes

```
Employés de salaire supérieur au salaire moyen de leur département
------------------------------------------------------------------
Michael Hartstein
Shelley Higgins
Steven King
Alexander Hunold
Bruce Ernst
Nancy Greenberg
Daniel Faviet
Den Raphaely
```

-- 11

```
CREATE OR REPLACE FUNCTION  check_sal(p_num_emp employees.employee_id%type)
RETURN boolean IS
v_dept_id employees.department_id%type;
v_sal employees.salary%type;
v_avg_sal employees.salary%type;
BEGIN
SELECT salary, department_id into v_sal, v_dept_id from employees where employee_id=p_num_emp;
SELECT avg(salary) into v_avg_sal from employees where department_id=v_dept_id;
IF v_sal > v_avg_sal THEN
RETURN TRUE;
ELSE
RETURN FALSE;
END IF;
EXCEPTION
WHEN NO_DATA_FOUND THEN
RETURN NULL;
END;
```

Test :

```
set verify off
BEGIN
CASE check_sal(&p_num)
WHEN TRUE THEN
dbms_output.put_line( 'Salaire > Moyenne des salaires');
WHEN FALSE THEN
dbms_output.put_line( 'Salaire < Moyenne des salaires');
ELSE
dbms_output.put_line('La fonction a renvoyé NULL à cause d''une exception');
END CASE;
END;
```

p_num = 100
p_num = 200
p_num = 1

```
Salaire > Moyenne des salaires

Salaire < Moyenne des salaires

La fonction a renvoyé NULL à cause d'une exception
```