Compte rendu TP de développement de bases de données ZZ2 F2

Auteur

Sinaro LY

TP 1

Question 1

```
CREATE TABLE Auteurs (
    Num INT,
    Nom VARCHAR(100),
    Prenom VARCHAR(100),
    Pays VARCHAR(2),
    Tel VARCHAR(10)
);
```

```
table creee
```

```
CREATE TABLE Ouvrage (
    Code INT,
    Titre VARCHAR(100),
    Prix FLOAT
);
```

```
table creee
```

Pour charger le fichier, écrire dans le shell : `$ORACLE_HOME/bin/sqlldr sily/tiger tp1.ctl`

contenu de tp1.ctl :

```
LOAD DATA INFILE *
INTO TABLE Auteurs
fields terminated by ","
(Num, Nom, Prenom, Pays, Tel)
BEGINDATA
1,Dupont,Jacques,FR,0473151585
2,Durand,Marie,GB,NULL
```

```
3,Dupont,Pierre,NULL,NULL
3,Dupont,NULL,NULL,NULL
```

```
2 Rows successfully loaded. Check the log file: tp1.log for more information about
the load.
```

```
INSERT INTO Ouvrage VALUES (001, 'Intro aux BD', 260);
INSERT INTO Ouvrage VALUES (002, 'Journal de Bolivie', NULL);
INSERT INTO Ouvrage VALUES (003, 'L''homme aux sendales', NULL);
```

```
1 ligne creee
```

Question 2

Pour créer la table des exceptions, on utilise la commande : `@$ORACLE_HOME/rdbms/admin/utlexcpt.sql` La table créée se nomme `exceptions`, il faut de donc la renommer :

```
RENAME exceptions TO aut_violation;
```

```
ALTER TABLE Auteurs ADD CONSTRAINT Auteurs_primary_key PRIMARY KEY(num) EXCEPTIONS
INTO exceptions;
```

```
ERREUR a la ligne 1 :
ORA-02437: impossible de valider (SILY.AUTEURS_PRIMARY_KEY) - violation de la
cle primaire
```

```
SELECT * FROM aut_violation
```

```
ROW_ID
------------------
OWNER
--------------------------------------------------------------------------------
TABLE_NAME
--------------------------------------------------------------------------------
CONSTRAINT
```

```
--------------------------------------------------------------------------------
AABg/NAANAAAUlmAAA
SILY
AUTEURS
AUTEURS_PRIMARY_KEY
```

Question 3

```
ALTER TABLE Auteurs ADD CONSTRAINT AUTEURS_UPPER_NAME CHECK(UPPER(Nom)=Nom )
EXCEPTIONS INTO aut_violation;
```

```
ERREUR a la ligne 1 :
ORA-02293: impossible de valider (SILY.AUTEURS_UPPER_NAME) - violation d'une
contrainte de controle
```

Question 4

```
ALTER TABLE Auteurs DROP CONSTRAINT Auteurs_Upper_Name;
```

TP 2

Question 1

```
set serveroutput on;
DECLARE
e_nom emp.ename%type;
e_sal emp.sal%type;
e_comm emp.comm%type;
Cursor c IS SELECT ename, sal, comm FROM emp WHERE ename = 'MILLER';
BEGIN

    OPEN c;
    FETCH c INTO e_nom, e_sal, e_comm;
    dbms_output.put_line('nom : ' || e_nom || ' salaire : ' || e_sal || '
commission : ' || e_comm);
    CLOSE c;
END;
/
```

Résultat :

```
nom : MILLER salaire : 1300 commission :

Procedure PL/SQL terminee avec succes.
```

Question 2

```
DECLARE
i number(3);
BEGIN
    For i in 1..100 loop
        if (mod(i,2)=0) then
            INSERT INTO temp values(i, i*100, to_char(i) || ' est pair');
        else
            INSERT INTO temp values(i, i*100, to_char(i) || ' est impair');
        end if;
    end loop;
END;
/
```

Résultat :

```
Procedure PL/SQL terminee avec succes.
```

Question 3

```
DECLARE
e_sal emp.sal%type;
e_empno emp.empno%type;
e_nom emp.ename%type;
Cursor c IS SELECT * FROM (SELECT sal, empno, ename FROM emp ORDER BY sal DESC)
WHERE rownum <=5;
BEGIN
    FOR rec_c in c
    LOOP
        e_sal:=rec_c.sal;
        e_empno:=rec_c.empno;
        e_nom:=rec_c.ename;
        INSERT INTO temp values(e_sal, e_empno, e_nom);
    END LOOP;
END;
/
```

Résultat :

```
Procedure PL/SQL terminee avec succes.


SQL> SELECT * FROM TEMP;

  NUM_COL1   NUM_COL2 CHAR_COL
---------- ---------- ----------
      5000   7839 KING
      3000   7902 FORD
      3000   7788 SCOTT
      2975   7566 JONES
      2850   7698 BLAKE
```

Question 4

```
DECLARE
e_sal emp.sal%type;
e_empno emp.empno%type;
e_nom emp.ename%type;
Cursor c IS SELECT sal, empno, ename FROM emp WHERE ((nvl(sal, 0) + nvl(comm, 0))
>= 2000);
BEGIN
    FOR rec_c in c
    LOOP
        e_sal:=rec_c.sal;
        e_empno:=rec_c.empno;
        e_nom:=rec_c.ename;
        INSERT INTO temp values(e_sal, e_empno, e_nom);
    END LOOP;
END;
/
```

Résultat :

```
Procedure PL/SQL terminee avec succes.


SQL> SELECT * FROM TEMP;

  NUM_COL1   NUM_COL2 CHAR_COL
---------- ---------- ---------
      2975   7566 JONES
      1250   7654 MARTIN
      2850   7698 BLAKE
      2450   7782 CLARK
      3000   7902 FORD
      3000   7788 SCOTT
```

```
       3500   7000 LY

8 lignes selectionnees.
```

Question 5

```
set serveroutput on;
DECLARE
    Cursor c IS SELECT * FROM emp start with empno=7902 CONNECT BY PRIOR
mgr=empno;
BEGIN
   FOR rec_c in c
   LOOP
        if rec_c.sal > 4000 or rec_c.mgr is null then
            INSERT INTO temp VALUES(rec_c.sal, rec_c.empno, rec_c.ename);
            exit;
        end if;
   END LOOP;
END;
/
```

Résultat :

```
Procedure PL/SQL terminee avec succes.
```

TP 3

Partie A - Question 1

```
CREATE OR REPLACE procedure createdept_ly
(x_numdept IN dept.deptno%type, x_nomdept IN dept.dname%type, x_locdept IN
dept.loc%type)
IS
Begin
    INSERT INTO  dept VALUES(x_numdept, x_nomdept, x_locdept);
    COMMIT;
End createdept_ly;
/
```

Résultat :

```
Procedure creee.
```

```
SQL> execute createdept_ly(21, 'essai1', 'Paris');

Procedure PL/SQL terminee avec succes.

SQL> execute createdept_ly(21, 'essai2', 'Paris');
BEGIN createdept_ly(21, 'essai2', 'Paris'); END;

*
ERREUR a la ligne 1 :
ORA-00001: violation de contrainte unique (SILY.SYS_C00111638)
ORA-06512: a "SILY.CREATEDEPT_LY", ligne 5
ORA-06512: a ligne 1
```

Question 2

```
create table SalIntervalle_ly (job varchar2(9), lsal number(7,2), hsal
number(7,2));
insert into SalIntervalle_ly values ('ANALYST', 2500, 3000) ;
insert into SalIntervalle_ly values ('CLERK', 900, 1300) ;
insert into SalIntervalle_ly values ('MANAGER', 2400, 3000) ;
insert into SalIntervalle_ly values ('PRESIDENT', 4500, 4900) ;
insert into SalIntervalle_ly values ('SALESMAN', 1200, 1700) ;

CREATE OR REPLACE function salok_ly
(x_job IN SalIntervalle_ly.job%type, x_sal IN SalIntervalle_ly.lsal%type)
RETURN number
IS
e_lsal SalIntervalle_ly.lsal%type;
e_hsal SalIntervalle_ly.hsal%type;
res number(8,2) :=0;
Cursor c IS SELECT lsal, hsal FROM SalIntervalle_ly WHERE job=x_job;
Begin
    OPEN c;
    FETCH c INTO e_lsal, e_hsal;
    CLOSE c;
    if x_sal >= e_lsal and x_sal<=e_hsal then
        res :=1;
    end if;
    return (res);
End salok_ly;
/
```

Résultat :

```
set serveroutput on;
DECLARE
resultat number(8,2);
BEGIN
```

```
    resultat := salok_ly('ANALYST', 2600);
    dbms_output.put_line(resultat);

    resultat := salok_ly('ANALYST', 1);
    dbms_output.put_line(resultat);

    resultat := salok_ly('DQZD', 1);
    dbms_output.put_line(resultat);

END;
/

1
0
0

Procedure PL/SQL terminee avec succes.
```

Question 4

```
CREATE OR REPLACE procedure raisesalary_ly
(x_emp_id IN emp.empno%type, x_amount IN emp.sal%type)
IS
new_sal emp.sal%type;
empjob emp.job%type;
Cursor c IS SELECT job, sal FROM emp WHERE empno=x_emp_id;
Begin
    OPEN c;
    FETCH c INTO empjob, new_sal;
    CLOSE c;
    new_sal := new_sal + x_amount;
    if (salok_ly(empjob, new_sal) = 1) then
        UPDATE emp
        SET sal = new_sal
        WHERE empno = x_emp_id;
        COMMIT;
    else
        raise_application_error(-20001,'salary is too high');
    end if;
End raisesalary_ly;
/
```

```
Procedure creee.

SQL> select * from emp where empno = 7900;

     EMPNO ENAME      JOB          MGR HIREDATE    SAL       COMM     DEPTNO
---------- ---------- --------- ---------- -------- ---------- ---------- --------
--
```

```
      7900 JAMES      CLERK       7698 03/12/82   900                 30

SQL> execute raisesalary_ly(7900, 50);

Procedure PL/SQL terminee avec succes.

SQL> select * from emp where empno = 7900;

     EMPNO ENAME      JOB          MGR HIREDATE    SAL       COMM     DEPTNO
---------- ---------- --------- ---------- -------- ---------- ---------- --------
--
      7900 JAMES      CLERK       7698 03/12/82   950                 30

SQL> execute raisesalary_ly(7900, 9999);
BEGIN raisesalary_ly(7900, 9999); END;

*
ERREUR a la ligne 1 :
ORA-20001: salary is too high
ORA-06512: a "SILY.RAISESALARY_LY", ligne 18
ORA-06512: a ligne 1
```

Partie B

```
DECLARE
    Cursor c IS SELECT table_name FROM user_tables WHERE table_name NOT LIKE
'%_OLD';
    Cursor c_old IS SELECT table_name FROM user_tables WHERE table_name LIKE
'%_OLD';
BEGIN
    FOR rec_c_old in c_old
    LOOP
        EXECUTE IMMEDIATE ('DROP TABLE ' || rec_c_old.table_name);
    END LOOP;
    COMMIT;
    FOR rec_c in c
    LOOP
        EXECUTE IMMEDIATE ('CREATE TABLE '|| rec_c.table_name || '_OLD AS SELECT *
FROM ' || rec_c.table_name);
    END LOOP;
    COMMIT;
END;
/

Create table ly (id number(5)) ;
insert into ly Values (1) ;
insert into ly Values (2) ;
insert into ly Values (3) ;
```

TP 4

Package

```
CREATE OR REPLACE Package package_tp4_ly AS
   TYPE EmpType IS RECORD (empno emp.empno%type, ename emp.ename%type);
   Cursor emp_par_dep_ly (x_deptno emp.deptno%type) return EmpType;
   Procedure raisesalary_ly (x_emp_id emp.empno%type, x_amount emp.sal%type);
   Procedure afficher_emp_ly (x_deptno emp.deptno%type);
END package_tp4_ly;
/

CREATE OR REPLACE Package Body package_tp4_ly AS
   Cursor emp_par_dep_ly (x_deptno emp.deptno%type) return EmpType IS
       SELECT empno, ename FROM emp WHERE deptno = x_deptno;
   Procedure raisesalary_ly (x_emp_id emp.empno%type, x_amount emp.sal%type) IS
       new_sal emp.sal%type;
       empjob emp.job%type;
       Cursor c IS SELECT job, sal FROM emp WHERE empno=x_emp_id;
       Begin
           OPEN c;
           FETCH c INTO empjob, new_sal;
           CLOSE c;
           new_sal := new_sal + x_amount;
           if (salok_ly(empjob, new_sal) = 1) then
               UPDATE emp
               SET sal = new_sal
               WHERE empno = x_emp_id;
           else
               raise_application_error(-20001,'salary is too high');
           end if;
       End raisesalary_ly;
   Procedure afficher_emp_ly (x_deptno emp.deptno%type) IS
       Cursor c IS SELECT empno, ename FROM emp WHERE deptno = x_deptno;
       Begin
           FOR rec_c in c
           LOOP
               dbms_output.put_line('numero employe : '|| rec_c.empno || '     nom
employe : ' || rec_c.ename);
           END LOOP;
       end afficher_emp_ly;

END package_tp4_ly;
/
```

Trigger - Question 1

```
CREATE OR REPLACE Trigger raise_ly
BEFORE UPDATE of sal on emp
For each row
Begin
   if (:old.sal > :new.sal) then
```

```
        raise_application_error(-20120, 'le salaire ne peut pas diminuer');
    End if;
End;
/
```

Résultat :

```
Declencheur cree.
SQL> UPDATE emp SET sal = sal - 100 WHERE ename = 'ADAMS';
       *
ERREUR a la ligne 1 :
ORA-20120: le salaire ne peut pas diminuer
ORA-06512: a "SILY.RAISE_LY", ligne 3
ORA-04088: erreur lors d'execution du declencheur 'SILY.RAISE_LY'


SQL> UPDATE emp SET sal = sal - 100;
       *
ERREUR a la ligne 1 :
ORA-20120: le salaire ne peut pas diminuer
ORA-06512: a "SILY.RAISE_LY", ligne 3
ORA-04088: erreur lors d'execution du declencheur 'SILY.RAISE_LY'
```

Question 2

```
CREATE OR REPLACE Trigger numdept_ly
BEFORE UPDATE of deptno on emp
For each row
Begin
    if (:new.deptno > 69 OR :new.deptno < 61) then
        raise_application_error(-20121, 'le departement doit etre dans [61,69]');
    End if;
End;
/
```

Résultat :

```
Declencheur cree.
SQL> UPDATE emp SET deptno = 70 WHERE ename = 'ADAMS';
       *
ERREUR a la ligne 1 :
ORA-20121: le departement doit etre dans [61,69]
ORA-06512: a "SILY.NUMDEPT_LY", ligne 3
ORA-04088: erreur lors d'execution du declencheur 'SILY.NUMDEPT_LY'
```

Question 3

```
CREATE OR REPLACE Trigger dept_ly
BEFORE INSERT OR UPDATE of deptno on emp
For each row
DECLARE
    c number;
Begin
    SELECT COUNT(*) INTO c FROM DEPT WHERE DEPTNO = :new.deptno;
    if c = 0  then
        INSERT INTO DEPT VALUES(:new.deptno, 'A SAISIR', 'A SAISIR');
    End if;
End;
/
```

Résultat:

```
Declencheur cree.

SQL> INSERT INTO EMP VALUES(6969, 'Longinus', 'CLERK', 7839, TO_DATE('20/12/2020',
'DD/MM/YYYY'), 1000, 200, 48);

1 ligne creee

SQL> SELECT * FROM DEPT;

    DEPTNO DNAME      LOC
---------- -------------- -------------
    21 essai1     Paris
    10 ACCOUNTING     NEW YORK
    20 RESEARCH   DALLAS
    30 SALES      CHICAGO
    40 OPERATIONS     BOSTON
    48 A SAISIR   A SAISIR
```

Question 4

```
CREATE OR REPLACE Trigger noweek_ly
BEFORE UPDATE OR INSERT OR DELETE on emp
For each row
Begin
    if (TO_CHAR(SYSDATE, 'D') = '7' OR TO_CHAR(SYSDATE, 'D') = '6')   then
        RAISE_APPLICATION_ERROR(-20021, 'Cannot insert record on weekends');
    End if;
End;
/
```

Résultat:

```
Declencheur cree.

SQL> UPDATE emp SET sal = sal + 100 WHERE ename = 'ADAMS';
UPDATE emp SET sal = sal + 100 WHERE ename = 'ADAMS'
       *
ERREUR a la ligne 1 :
ORA-20021: Cannot insert record on weekends
ORA-06512: a "SILY.NOWEEK_LY", ligne 3
ORA-04088: erreur lors d'execution du declencheur 'SILY.NOWEEK_LY'
```

Question 5

```
ALTER TRIGGER noweek_ly DISABLE;
```

Résultat:

```
Declencheur modifie.

SQL> UPDATE emp SET sal = sal + 100 WHERE ename = 'ADAMS';

1 ligne mise a jour.
```

Question 6

```
ALTER TRIGGER noweek_ly ENABLE;
```

Résultat:

```
Declencheur modifie.

SQL> UPDATE emp SET sal = sal + 100 WHERE ename = 'ADAMS';
UPDATE emp SET sal = sal + 100 WHERE ename = 'ADAMS'
       *
ERREUR a la ligne 1 :
ORA-20021: Cannot insert record on weekends
ORA-06512: a "SILY.NOWEEK_LY", ligne 3
ORA-04088: erreur lors d'execution du declencheur 'SILY.NOWEEK_LY'
```

Question 7-A

```
CREATE TABLE STATS_ly (TypeMaj varchar2(10), NbMaj number, Date_derniere_Maj
date);

insert into STATS_ly values ('INSERT', 0, NULL);
insert into STATS_ly values ('DELETE', 0, NULL);
insert into STATS_ly values ('UPDATE', 0, NULL);

CREATE OR REPLACE Trigger update_stats_ly
AFTER UPDATE OR INSERT OR DELETE on emp
For each row
Begin
    if INSERTING then
    UPDATE STATS_ly SET NbMaj = NbMaj+1, Date_derniere_Maj = SYSDATE WHERE
TypeMaj='INSERT';
    End if;
    if UPDATING then
    UPDATE STATS_ly SET NbMaj = NbMaj+1, Date_derniere_Maj = SYSDATE WHERE
TypeMaj='UPDATE';
    End if;
    if DELETING then
    UPDATE STATS_ly SET NbMaj = NbMaj+1, Date_derniere_Maj = SYSDATE WHERE
TypeMaj='DELETE';
    End if;
End;
/
```

Résultat:

```
Declencheur modifie.

SQL> UPDATE emp SET sal = sal + 100 WHERE ename = 'ADAMS';
UPDATE emp SET sal = sal + 100 WHERE ename = 'ADAMS'
       *
ERREUR a la ligne 1 :
ORA-20021: Cannot insert record on weekends
ORA-06512: a "SILY.NOWEEK_LY", ligne 3
ORA-04088: erreur lors d'execution du declencheur 'SILY.NOWEEK_LY'

SQL> UPDATE emp SET sal = sal + 100 WHERE ename = 'Longinus';

1 ligne mise a jour.

SQL> select * from stats_ly;

TYPEMAJ     NBMAJ DATE_DER
---------- ---------- --------
INSERT          1 20/03/22
DELETE          0
UPDATE          1 20/03/22

SQL> DELETE FROM emp WHERE ename = 'Longinus';
```

```
1 ligne supprimee.

SQL> select * from stats_ly;

TYPEMAJ     NBMAJ DATE_DER
---------- ---------- --------
INSERT          1 20/03/22
DELETE          1 20/03/22
UPDATE          1 20/03/22
```

Question 7-B

On constate en utilisant par exemple `UPDATE emp SET sal = sal * 1.05;` le résultat suivant :

```
SQL> select * from stats_ly;

TYPEMAJ     NBMAJ DATE_DER
---------- ---------- --------
INSERT          0
DELETE          0
UPDATE         15 20/03/22
```

On en déduit donc que l'on incrémente autant que le nombre de n-uplets modifiés.

Question 8

```
CREATE OR REPLACE Trigger checksal_ly
BEFORE UPDATE of job on emp
For each row
DECLARE
    min_sal emp.sal%type;
    max_sal emp.sal%type;
Begin
    if (:old.job != 'PRESIDENT') then
        SELECT lsal, hsal INTO min_sal, max_sal FROM SalIntervalle_ly WHERE job =
:new.job;
        :new.sal := GREATEST(min_sal, LEAST(max_sal, :old.sal+100));
    End if;
End;
/
```

Résultat:

```
SQL> select * from emp where empno = 7499;

     EMPNO ENAME      JOB          MGR HIREDATE     SAL       COMM     DEPTNO
```

```
---------- ---------- --------- ---------- -------- ---------- ---------- --------
--
      7499 ALLEN      SALESMAN        7698 20/02/81   1680        300     30

SQL> update emp set job='MANAGER' WHERE empno = 7499;

1 ligne mise a jour.

SQL> select * from emp where empno = 7499;

     EMPNO ENAME      JOB          MGR HIREDATE    SAL       COMM     DEPTNO
---------- ---------- --------- ---------- -------- ---------- ---------- --------
--
      7499 ALLEN      MANAGER         7698 20/02/81   2400        300     30
```