TD/TP3 : CONTRAINTES D'INTEGRITE STATIQUES ET DYNAMIQUES

PARTIE 1 : CONTRAINTES D'INTEGRITE STATIQUES

1. Lister l'ensemble de contraintes d'intégrité pour chaque table de ce schéma.

```
SQL> select constraint_name from all_constraints where table_name='BIOLOGISTE';

CONSTRAINT_NAME
------------------------------------------------------------------------------
CK1
PK1
PK1
SQL> select constraint_name from all_constraints where table_name='PATIENT';

CONSTRAINT_NAME
------------------------------------------------------------------------------
PK2

SQL> select constraint_name from all_constraints where table_name='PRELEVEMENT';

CONSTRAINT_NAME
------------------------------------------------------------------------------
FK1
PK3
U2

SQL> select constraint_name from all_constraints where table_name='EFFECTUEPRELEVEMENT';

CONSTRAINT_NAME
------------------------------------------------------------------------------
FK2
FK3
FK4
PK4
```

2. Etendre la liste des rôles possibles avec la nouvelle fonction : « B-M à Domicile ».

```
SQL> alter table biologiste add constraint roleB check(roleB in ('Biologiste_Responsable','Biologiste_Medical','Ing_Qualité'
,'Aide_laboratoire','Secretaire','Technicien','Ing_Informatique','B-M à Domicile '));

Table modifi0e.

ERREUR α la ligne 1 :
ORA-02290: violation de contraintes (DBALAB.ROLEB) de v0rification
```

3. Exigez que la date de naissance des patients soit antérieure à la date d'aujourd'hui. Tester avec quelques insertions.

La fonction TRUNC() permet de récupérer la partie entière d'un nombre ou d'une date en éliminant les parties décimales ou les parties de temps.

```
SQL> CREATE OR REPLACE TRIGGER Date_naiss
  2  BEFORE INSERT OR UPDATE ON patient
  3  FOR EACH ROW
  4  BEGIN
  5    IF TRUNC(:NEW.date_naiss) >= TRUNC(SYSDATE) THEN
  6      RAISE_APPLICATION_ERROR(-20001, 'La date de naissance doit être antérieure à la date d''aujourd''hui');
  7    END IF;
  8  END;
  9  /

D0clencheur cr00.
```

```
SQL> insert into patient values(11,'AITALI','Bahia',TO_DATE('31/03/2023','DD/MM/YYYY'));
insert into patient values(11,'AITALI','Bahia',TO_DATE('31/03/2023','DD/MM/YYYY'))
            *
ERREUR α la ligne 1 :
ORA-20001: La date de naissance doit être antérieure à la date d'aujourd'hui
ORA-06512: α "DBALAB.DATE_NAISS", ligne 3
ORA-04088: erreur lors d'ex0cution du d0clencheur 'DBALAB.DATE_NAISS'
```

4. Un ingénieur ne peut exercer en tant que Bio-Medical. Ajouter la contrainte et effectuer les tests nécessaires.

```
SQL> ALTER TABLE biologiste
  2  ADD CONSTRAINT B1 CHECK(not(specialit='ingenieur' AND roleb='Biologiste_Medical'));

Table modifi0e.
```

5. Supprimer la table Biologiste. Que remarquez-vous ?

On ne peut pas

```
SQL> drop table biologiste;
drop table biologiste
           *
ERREUR α la ligne 1 :
ORA-02449: cl0s uniques/primaires de la table r0f0renc0es par des cl0s
0trang0res
```

- Désactiver la CI qui bloque la suppression et réessayer. Que remarquez-vous ?

Nous pouvons conclure que la contrainte fk2 correspond à une clé étrangère (NumB) de la table effectueprelevement.

```
SQL> ALTER TABLE EffectuePrelevement DISABLE CONSTRAINT fk2;

Table modifi0e.

SQL> ALTER TABLE EffectuePrelevement DISABLE CONSTRAINT pk4;

Table modifi0e.
```

On ne peut pas supprimer la table biologiste même si on désactive les contraintes

```
SQL> drop table biologiste;
drop table biologiste
           *
ERREUR α la ligne 1 :
ORA-02449: cl0s uniques/primaires de la table r0f0renc0es par des cl0s
0trang0res
```

- Supprimer cette contrainte et réessayer. Conclure.

```
SQL> ALTER TABLE EffectuePrelevement Drop CONSTRAINT pk4;

Table modifi0e.

SQL> ALTER TABLE EffectuePrelevement DROP CONSTRAINT fk2;

Table modifi0e.

SQL> drop table biologiste;

Table supprim0e.

SQL>
```

En conclusion, il n'est pas possible de supprimer directement une table parente même en désactivant les contraintes qui empêchent la suppression. La suppression de la table parente n'est possible que si toutes les contraintes (clés primaires, clés étrangères) sont supprimées au préalable. Cependant, dans certains cas, il est possible de supprimer une table parente en désactivant les contraintes si la contrainte en question est une contrainte de vérification (check).

- Recréer la table Biologiste.

```
SQL> create table biologiste(NumB number(6),Nom varchar2(15),Prenom varchar2(15),Specialit varchar2(40),RoleB varchar2(40),
constraint pk1 primary key(NumB));

Table cr00e.

SQL> insert into biologiste values(1,'BADI','Salim','Microbio','Biologiste_Responsable');

1 ligne cr00e.
```

```
SQL> alter table effectueprelevement add constraint pk4 primary key(NumB,NumP,NumPr);

Table modifi0e.

SQL> alter table effectueprelevement add constraint fk2 foreign key (NumB)references biologiste(NumB);

Table modifi0e.

SQL>
```

PARTIE 2 : META-BASE (DICTIONNAIRE DE DONNEES)

1. Pour analyser l'organigramme du laboratoire, nous cherchons à afficher le personnel et leurs rôles.
Ecrire un code PLSQL qui permet d'afficher pour chaque rôle, le nombre de biologistes correspondants.
Exemple : Il y a 2 personne(s) qui exerce(nt) en tant que «Aide-laboratoire».

```
SQL> DECLARE
  2  nbr_role number(20);
  3  BEGIN
  4  FOR cr in (select DISTINCT roleB from biologiste) LOOP
  5  select count(*) into nbr_role from biologiste where roleB=cr.roleB;
  6  DBMS_OUTPUT.PUT_LINE('il y a: ' || nbr_role || ' qui exerce(nt) en tant que ' || cr.roleB );
  7  END loop;
  8  END;
  9  /
il y a: 1 qui exerce(nt) en tant que Biologiste_Responsable
il y a: 1 qui exerce(nt) en tant que Ing_Qualité
il y a: 2 qui exerce(nt) en tant que Aide_laboratoire
il y a: 1 qui exerce(nt) en tant que Secretaire
il y a: 1 qui exerce(nt) en tant que Technicien
il y a: 1 qui exerce(nt) en tant que Ing_Informatique
il y a: 3 qui exerce(nt) en tant que Biologiste_Medical

Proc0dure PL/SQL termin0e avec succ0s.
```

Il y a 2 personne(s) qui exerce(nt) en tant que «Aide-laboratoire».

2. Supposant que la norme mondiale pour le test antigénique Covid passe du seuil initial "0.5" à "0.8" pour que le résultat soit déclaré positif. Ajouter l'information, puis ajoutez la contrainte suivante : si le taux d'antigènes détectés est > 0.8 alors la conclusion doit être = "Positif".
Vérifiez après mis-à-j.

```
SQL> UPDATE resultat SET norme = '0.5 à 0.8' WHERE typeresultat = 'Antig-Covid';

2 lignes mises α jour.
```

- La vérification :

```
SQL> select norme ,typeresultat from resultat;

NORME                                    TYPERESULTAT
---------------------------------------- ------------------------
12 à 16g/dL                              Hemoglobine
150K à 400K/mm3                          Plaquettes
4K à 10K/mm3                             Leucocytes
1.5K à 4K/mm3                            Lymphocytes
0.5 à 0.8                                Antig-Covid
A, B, AB, O -+                           Groupage
-                                        Culture
-                                        Sens. Antibiotique
12 à 16g/dL                              Hemoglobine
150k à 400k/mm3                          Plaquettes
4k à 10k/mm3                             Leucocytes

NORME                                    TYPERESULTAT
```

```
SQL> ALTER TABLE resultat
  2  ADD CONSTRAINT Pos CHECK (CAST (resul AS NUMBER) > 0.8 AND conclusion = 'Positif' and typeresultat = 'Antig-Covid')
;
ADD CONSTRAINT Pos CHECK (CAST (resul AS NUMBER) > 0.8 AND conclusion = 'Positif' and typeresultat = 'Antig-Covid')
            *
ERREUR α la ligne 2 :
ORA-02293: impossible de valider (DBALAB.POS) - violation d'une contrainte de
contr⌈le
```

3. Ecrire une procédure CasPositifs qui affiche « Le patient "i" a été testé positif. » si le patient en question a un résultat positif aux tests Covid de différents types de prélèvement.

```
SQL> CREATE OR REPLACE PROCEDURE Cas_postif
  2  IS
  3    CURSOR cr IS
  4      SELECT numP, TypePr
  5      FROM resultat R, prelevement P
  6      WHERE R.numPr = P.numPr AND UPPER(conclusion) LIKE '%POSITIF%' AND UPPER(typeresultat) LIKE '%COVID%'
  7      GROUP BY TypePr, numP;
  8    found BOOLEAN := FALSE;
  9  BEGIN
 10    FOR cr_rec IN cr LOOP
 11      dbms_output.put_line('Le Patient N°' || cr_rec.numP || ' est testé positif');
 12      found := TRUE;
 13    END LOOP;
 14
 15    IF NOT found THEN
 16      DBMS_OUTPUT.PUT_LINE('Il n''existe pas de patients qui sont positifs');
 17    END IF;
 18  END;
 19  /

Proc0dure cr00e.

SQL> EXECUTE Cas_postif;
Le Patient N°4 est testé positif
Le Patient N°5 est testé positif

Proc0dure PL/SQL termin0e avec succ0s.

SQL>
```

4. Ecrire une fonction qui retourne, pour un Biologiste donné, le nombre de prélèvements effectués.
Exécuter la fonction pour plusieurs biologistes.
Exemple : Le Biologiste BADI a effectué 3 prélèvements.

```
SQL> CREATE OR REPLACE FUNCTION NBR_PRELEVEMENT(biologiste IN VARCHAR2)
  2  RETURN NUMBER
  3  IS
  4    nbr_pr NUMBER := 0;
  5  BEGIN
  6    SELECT COUNT(*) INTO nbr_pr
  7    FROM biologiste B, Effectueprelevement E
  8    WHERE B.numB = E.numB AND B.nom = biologiste;
  9    RETURN nbr_pr;
 10  END;
 11  /

Fonction cr00e.
```

Le Biologiste BADI a effectué 3 prélèvements.

- Les tests sur tous les biologistes :

```
SQL> SELECT 'Le biologiste ' || UPPER(biologiste.nom) || ' a effectué ' || NBR_PRELEVEMENT(biologiste.nom) || ' prélèvem
ents.' AS RESULTAT
  2  FROM biologiste;

RESULTAT
--------------------------------------------------------------------------------
Le biologiste BADI a effectué 3 prélèvements.
Le biologiste SAHLI a effectué 0 prélèvements.
Le biologiste NADIR a effectué 2 prélèvements.
Le biologiste BENMIHOUB a effectué 0 prélèvements.
Le biologiste CHERGUI a effectué 0 prélèvements.
Le biologiste BOUSALEM a effectué 3 prélèvements.
Le biologiste KADI a effectué 0 prélèvements.
Le biologiste SMATI a effectué 2 prélèvements.
Le biologiste NAILI a effectué 2 prélèvements.
Le biologiste AMRAN a effectué 3 prélèvements.

10 lignes s0lectionn0es.

SQL>
```

5. Créer une procédure qui permet d'ajouter un prélèvement à partir de tous les attributs nécessaires. N'oublier pas de vérifier l'unicité de la clé et l'existence de clé étrangère vers
Patient et Biologiste. Affichez les messages d'erreurs en cas de problèmes.

```
SQL> CREATE OR REPLACE PROCEDURE AJOUTER_PRELEVEMENT(
  2    IN_NUM_PRELEVEMENT IN PRELEVEMENT.NumPr%TYPE,
  3    IN_NUM_PATIENT IN PATIENT.NumP%TYPE,
  4    IN_DATE_PRELEVEMENT IN PRELEVEMENT.DatePr%TYPE,
  5    IN_TYPE_PRELEVEMENT IN PRELEVEMENT.TypePr%TYPE
  6  ) AS
  7    V_NUM_PATIENT_EXISTE NUMBER;
  8    V_NUM_PRELEV_EXISTE NUMBER;
  9  BEGIN
 10    -- Vérifier que le patient existe
 11    SELECT COUNT(*) INTO V_NUM_PATIENT_EXISTE FROM PATIENT WHERE NumP = IN_NUM_PATIENT;
 12    IF V_NUM_PATIENT_EXISTE = 0 THEN
 13      dbms_output.put_line('Erreur: Le patient n''existe pas.');
 14
 15    END IF;
 16
 17    -- Vérifier que le numéro de prélèvement n'existe pas déjà
 18    SELECT COUNT(*) INTO V_NUM_PRELEV_EXISTE FROM PRELEVEMENT WHERE NumPr = IN_NUM_PRELEVEMENT;
 19    IF V_NUM_PRELEV_EXISTE > 0 THEN
 20      dbms_output.put_line('Erreur: Le numéro de prélèvement existe déjà.');
 21     END IF;
 22    -- Insérer les données dans la table Prelevement
 23    INSERT INTO PRELEVEMENT(NumPr, NumP, DatePr, TypePr)
 24    VALUES (IN_NUM_PRELEVEMENT, IN_NUM_PATIENT, IN_DATE_PRELEVEMENT, IN_TYPE_PRELEVEMENT);
 25
 26    dbms_output.put_line('Le prélèvement a été ajouté avec succès.');
 27  EXCEPTION
 28    WHEN OTHERS THEN
 29      dbms_output.put_line('Erreur: Une erreur est survenue lors de l''ajout du prélèvement.');
 30  END;
 31  /

Proc0dure cr00e.
```

Les exemples possibles d'exécution :

```
SQL> execute  AJOUTER_PRELEVEMENT(1, 2, SYSDATE, 'Type de prélèvement');
Erreur: Le numéro de prélèvement existe déjà.
Erreur: Une erreur est survenue lors de l'ajout du prélèvement.

Proc0dure PL/SQL termin0e avec succ0s.

SQL>  execute  AJOUTER_PRELEVEMENT(1,11, SYSDATE, 'Type de prélèvement');
Erreur: Le patient n'existe pas.
Erreur: Le numéro de prélèvement existe déjà.
Erreur: Une erreur est survenue lors de l'ajout du prélèvement.

Proc0dure PL/SQL termin0e avec succ0s.

SQL>  execute  AJOUTER_PRELEVEMENT(15,3, SYSDATE, 'Type de prélèvement');
Le prélèvement a été ajouté avec succès.

Proc0dure PL/SQL termin0e avec succ0s.

SQL> select * from prelevement where numPr=15;

     NUMPR       NUMP DATEPR   TYPEPR
---------- ---------- -------- ----------------------------------------
        15          3 01/04/23 Type de prélèvement

SQL>
```

Pour la table effectueprelevement :

```
SQL> CREATE OR REPLACE PROCEDURE AJOUTER_EffectuePrelevement(
  2    NumB1 IN BIOLOGISTE.Numb%TYPE,
  3    NumP1 IN PATIENT.NumP%TYPE,
  4    NumPr1 IN PRELEVEMENT.NumPr%TYPE
  5
  6  ) AS
  7    V_NUM_PATIENT_EXISTE NUMBER;
  8    V_NUM_PRELEV_EXISTE NUMBER;
  9    V_NUM_BIOLOGISTE_EXISTE NUMBER;
 10    V_NUM_EFFECTUEPRELEV_EXISTE NUMBER;
 11  BEGIN
 12    -- Vérifier que le patient existe
 13    SELECT COUNT(*) INTO V_NUM_PATIENT_EXISTE FROM PATIENT WHERE NumP =NumP1;
 14    IF V_NUM_PATIENT_EXISTE = 0 THEN
 15      dbms_output.put_line('Erreur: Le patient n''existe pas.');
 16
 17    END IF;
 18  -- Vérifier que le biologiste existe
 19    SELECT COUNT(*) INTO V_NUM_BIOLOGISTE_EXISTE FROM biologiste WHERE NumB = NumB1;
 20    IF V_NUM_BIOLOGISTE_EXISTE = 0 THEN
 21      dbms_output.put_line('Erreur: Le biologiste n''existe pas.');
 22
 23    END IF;
 24    -- Vérifier que le prelevement existe
 25    SELECT COUNT(*) INTO V_NUM_PRELEV_EXISTE FROM prelevement WHERE NumPr= NumPr1;
 26    IF V_NUM_PRELEV_EXISTE = 0 THEN
 27      dbms_output.put_line('Erreur: Le prelevement n''existe pas.');
 28
 29    END IF;
 30    -- Vérifier que le numéro de prélèvement n'existe pas déjà
 31    SELECT COUNT(*) INTO V_NUM_EFFECTUEPRELEV_EXISTE FROM effectueprelevement  WHERE NumPr =NumPr1 AND NumB=NumB1 AND
NumP=NumP1;
 32    IF V_NUM_EFFECTUEPRELEV_EXISTE> 0 THEN
 33      dbms_output.put_line('Erreur: Le prélèvement(NUMB,NUMPR,NUMP) existe déjà.');
 34     END IF;
 35    -- Insérer les données dans la table  effectue Prelevement
 36    INSERT INTO EFFECTUEPRELEVEMENT(NumB, NumP, NumPr)
 37    VALUES (NumB1, NumP1, NumPr1);
 38
 39    dbms_output.put_line('Le prélèvement a été ajouté avec succès dans la table effectue prelevement.');
 40  EXCEPTION
 41    WHEN OTHERS THEN
 42      dbms_output.put_line('Erreur: Une erreur est survenue lors de l''ajout du prélèvement dans la table effectue pr
elevement.');
 43  END;
 44  /

Proc0dure cr00e.
```

Les exemples possibles de l'exécution :

```
SQL> set serveroutput on;
SQL> execute AJOUTER_EffectuePrelevement(1,1,1);
Erreur: Le prélèvement(NUMB,NUMPR,NUMP) existe déjà.
Erreur: Une erreur est survenue lors de l'ajout du prélèvement dans la table
effectue prelevement.

Proc0dure PL/SQL termin0e avec succ0s.

SQL> execute AJOUTER_EffectuePrelevement(14,1,1);
Erreur: Le biologiste n'existe pas.
Erreur: Une erreur est survenue lors de l'ajout du prélèvement dans la table
effectue prelevement.

Proc0dure PL/SQL termin0e avec succ0s.

SQL> execute AJOUTER_EffectuePrelevement(1,1,1);
Erreur: Le prélèvement(NUMB,NUMPR,NUMP) existe déjà.
Erreur: Une erreur est survenue lors de l'ajout du prélèvement dans la table
effectue prelevement.

Proc0dure PL/SQL termin0e avec succ0s.

SQL> execute AJOUTER_EffectuePrelevement(14,14,2);
Erreur: Le patient n'existe pas.
Erreur: Le biologiste n'existe pas.
Erreur: Une erreur est survenue lors de l'ajout du prélèvement dans la table
effectue prelevement.

Proc0dure PL/SQL termin0e avec succ0s.

SQL>
```

6. Créez un trigger qui affiche « un nouveau patient est ajouté » après chaque insertion d'un patient.

```
SQL> set serveroutput on;
SQL> CREATE OR REPLACE TRIGGER INSERTION_PATIENT
  2  AFTER INSERT
  3  on Patient
  4  BEGIN
  5  dbms_output.put_line('un nouveau patient est ajouté ');
  6  END;
  7  /

D0clencheur cr00.
```

- Répétez la même chose pour la modification ou la suppression.

```
SQL> CREATE OR REPLACE TRIGGER Update_Suppression_PATIENT
  2  AFTER UPDATE OR DELETE
  3  on Patient
  4  BEGIN
  5  dbms_output.put_line('un patient est modifié/supprimé ');
  6  END;
  7  /

D0clencheur cr00.

SQL> delete from patient where numP=20;
un patient est modifié/supprimé

1 ligne supprim0e.

SQL>
```

7. Créer un trigger qui empêche la modification du numéro de biologiste de la table des prélèvements.

```
SQL> CREATE OR REPLACE TRIGGER EMPECHE_MODIFICATION_NUMB_PRELEVEMENT
  2  BEFORE UPDATE
  3  OF numB
  4  on effectueprelevement
  5  BEGIN
  6  RAISE_APPLICATION_ERROR(-20000,'erreur vous ne pouvez pas modifier le numB dans la table effectueprelevement');
  7  END;
  8  /

D0clencheur cr00.

SQL> update effectueprelevement set numB=7 where numB=1;
update effectueprelevement set numB=7 where numB=1
       *
ERREUR α la ligne 1 :
ORA-20000: erreur vous ne pouvez pas modifier le numB dans la table
effectueprelevement
```

8. L'administrateur veut, pour un besoin interne, avoir le nombre total des prélèvements par biologistes. Pour cela, il ajoute un attribut : Nb_Pr dans la table Biologistes.
Ajoutez l'attribut.

```
SQL> alter table biologiste add Nb_Pr number(4);

Table modifi0e.

SQL> desc biologiste;
 Nom                                       NULL ?   Type
 ----------------------------------------- -------- --------------------
 NUMB                                      NOT NULL NUMBER(6)
 NOM                                                VARCHAR2(15)
 PRENOM                                             VARCHAR2(15)
 SPECIALIT                                          VARCHAR2(40)
 ROLEB                                              VARCHAR2(40)
 NB_PR                                              NUMBER(4)

SQL>
```

-l'initialisation de NB_Pr :

```
SQL> CREATE OR REPLACE PROCEDURE Initialisation_Nb_Pr
  2  IS
  3    cursor cr1 is select numB from biologiste;
  4    v_nbr number;
  5  BEGIN
  6    FOR i IN cr1 loop
  7      select count(NumPr) into v_nbr from effectueprelevement where numB=i.numB;
  8      update biologiste set Nb_Pr=v_nbr where numB=i.numB;
  9    END loop;
 10  END;
 11  /

Proc0dure cr00e.
```

La vérification :

```
SQL> execute Initialisation_Nb_Pr;

Proc0dure PL/SQL termin0e avec succ0s.
```

```
SQL> select Nb_Pr, numB from biologiste ;

     NB_PR        NUMB
---------- ----------
         3           1
         0           3
         2           4
         0           5
         0           6
         3           7
         0           8
         2           9
         2          10
         3           2

10 lignes s0lectionn0es.

SQL>
```

- Créez un trigger T1 qui permet de màj automatiquement l'attribut Nb_Pr

```
SQL> CREATE OR REPLACE TRIGGER ALLIMENTER_Nb_Pr
  2  AFTER INSERT OR DELETE OR UPDATE
  3  ON effectueprelevement
  4  BEGIN
  5  update biologiste set Nb_Pr=(select count(distinct(effectueprelevement.NumPr)) from effectueprelevement whe
re effectueprelevement.numB=biologiste.numB);
  6  END;
  7  /

D0clencheur cr00.
```

```
SQL> select Nb_Pr from biologiste where numB=1;

     NB_PR
----------
         3

SQL> insert into effectueprelevement values(1,1,1);

1 ligne cr00e.

SQL> select Nb_Pr from biologiste where numB=1;

     NB_PR
----------
         4

SQL>
```