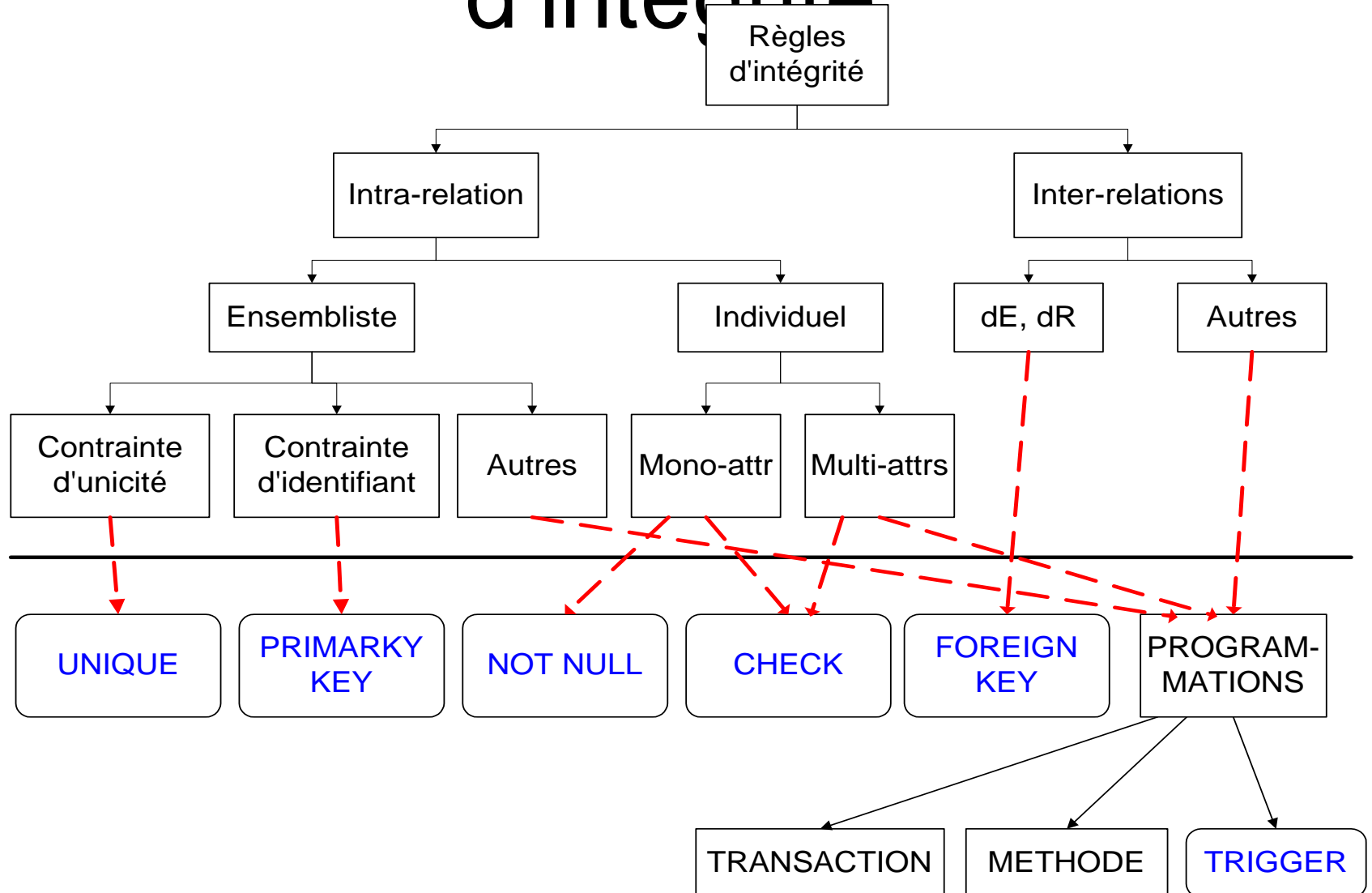


Validation des RI par des triggers

SES – Université de Genève

Taxonomie des règles d'intégrité



Etapes d'implémentation des RI

- Déterminer si une RI peut être « implicitement » validée grâce aux RIs standards supportées par SGBD
 - Clé primaire, clé étrangère, CHECK, NOT NULL
- Pour des RI « complexes »
 1. Définir la portée pour chaque RI.
 2. Établir le tableau de portées global.
 3. Spécifier les actions pour chaque cas de transgresser la RI.
 4. Implémenter des triggers.

Implémentation des triggers

- Chaque risque dans la portée *peut être* un trigger.
 - *Implémenter toutes les RI concernant une opération d'une table dans un seul trigger.*
- Préciser la condition du déclenchement, spécifier pour optimiser la validation.
 - Colonnes concernées en cas de modification.
 - WHEN (<condition>)
- Le temps de déclenchement:
 - Si l'action de maintenir la RI est:
 - De modifier les nouvelles valeurs pour qu'elles soient satisfaites la RI => trigger de tuple BEFORE
 - D'autre cas, utiliser trigger avec le temps de déclenchement AFTER (raison de performance).

Exemple

- Sujet: Gestion d'un parc de véhicules (PARCVEH)
 - VOITURE (NOV,MV,KM,PSG)
 - CH (NCH,CHAUFFEUR)
 - V_CH (NOV,NCH,NKM)
 - REPARATION (NOREP,NOV,NOG,TYPREP,PX,KMCPT)
 - TRAJET (NOTRAJ,VILLEDEP,VILLEARR,DATETRAJET,NBKM)
 - TR_NOV (NOTRAJ,NOV,NCH,NBPERSTR)

Exemple - PARCVEH

- RIs de domaine (ex. PSG est de 1..5)
- RI1 : pour tout trajet, la ville de départ est différente de celle d'arrivée.
- RI2: le nombre de passagers transportés sur une voiture pour un trajet doit être inférieur au nombre de places disponibles de cette voiture.
- RI3 : les trajets ayant la même ville de départ et d'arrivée doivent avoir la même distance en km.
- RI4: le nombre de km au compteur KMCPT doit être inférieur ou égal à celui de KM parcourus.

Exemple - PARCVEH

- RIs de domaine \Rightarrow CHECK, NOT NULL
- RI1 \Rightarrow CHECK
- RI2 :
 - VOITURE : maj PSG
 - TR_NOV : créer, maj NBPERSTR
- RI3 :
 - TRAJET : créer, maj VILLEDEP, maj VILLEARR, maj NBKM
- RI4 :
 - VOITURE : maj KM
 - REPARATION : créer, maj KMCPT

Exemple - PARCVEH

Tableau de portée global

	VOITURE			REPARATION			TRAJET			TR_NOV		
	Cr	Maj	Sup	Cr	Maj	Sup	Cr	Maj	Sup	Cr	Maj	Sup
RI2		PSG								+	NBPERSTR	
RI3							+	VILLEDEP, VILLEARR, NBKM				
RI4		KM		+	KMCPT							

RI2: Actions

- *Maj PSG* : Refuser si la nouvelle valeur n'est pas satisfaite des valeurs de NBPERSTR stockées.
- *Créer (Insérer) un objet TR_NOV* : Si NBPERSTR est supérieur à celui de places disponibles, affecter la valeur actuelle de PSG à NBPERSTR.
- *Maj NBPERSTR* : Refuser si NBPERSTR est supérieur à PSG de la voiture correspondante

Exemple : Triggers dans TR_NOV

TR_NOV : Maj NBPERSTR	<pre>CREATE OR REPLACE TRIGGER maj_nbperstr AFTER UPDATE OF nbperstr ON tr_nov FOR EACH ROW WHEN (new.nbperstr>old.nbperstr)</pre>
Action : Refuser et afficher un message d'erreur si NBPERSTR est supérieur à PSG de la voiture correspondante.	<pre>DECLARE vt_psg voiture.psg%TYPE; BEGIN SELECT psg INTO vt_psg FROM voiture WHERE nov=:NEW.nov; IF vt_psg < :NEW.nbperstr THEN RAISE_APPLICATION_ERROR (-20001, 'MAJ est invalid'); END IF; END;</pre>

Exemple : Triggers dans TR_NOV

TR_NOV : Créer	<pre>CREATE OR REPLACE TRIGGER cre_tr_nov BEFORE INSERT ON tr_nov FOR EACH ROW WHEN (new.nbperstr>0)</pre>
Action : Affecter la valeur actuelle de PSG à NBPERSTR si la valeur d'entrée est supérieure à PSG.	<pre>DECLARE vt_psg voiture.psg%TYPE; BEGIN SELECT psg INTO vt_psg FROM voiture WHERE nov = :NEW.nov; IF vt_psg < :NEW.nbperstr THEN :NEW.nbperstr := vt_psg; END IF; END;</pre>

TRIGGER

Programmation avancée

SES – Université de Genève

Trigger : comment ça marche?

Décrivez le code du trigger
(Notepad, Wordpad, MSWord,...)

Compilez le trigger
(SQL*Plus : copiez le code du trigger et le compilez)

Trigger est créé

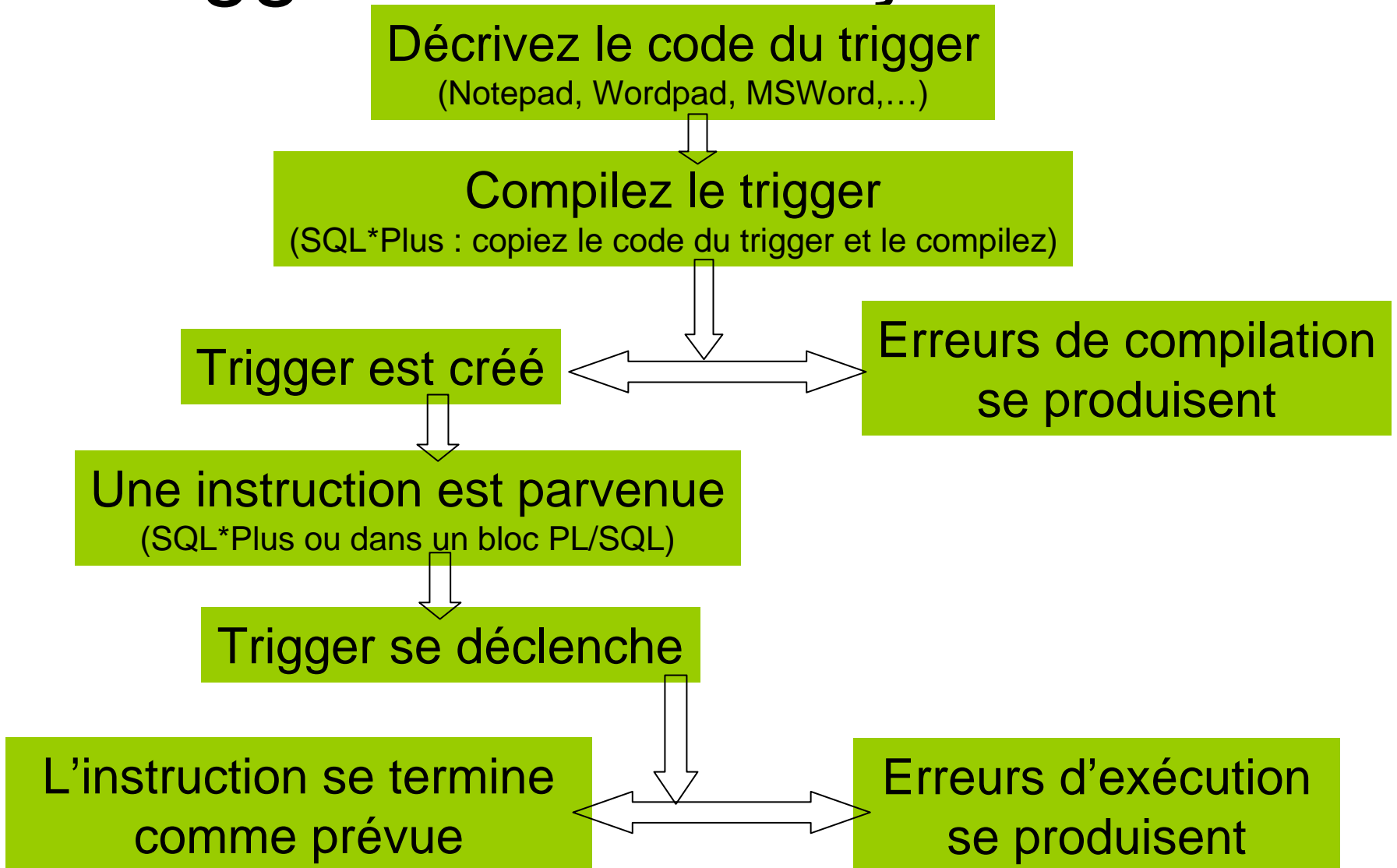
Erreurs de compilation
se produisent

Une instruction est parvenue
(SQL*Plus ou dans un bloc PL/SQL)

Trigger se déclenche

L'instruction se termine
comme prévue

Erreurs d'exécution
se produisent

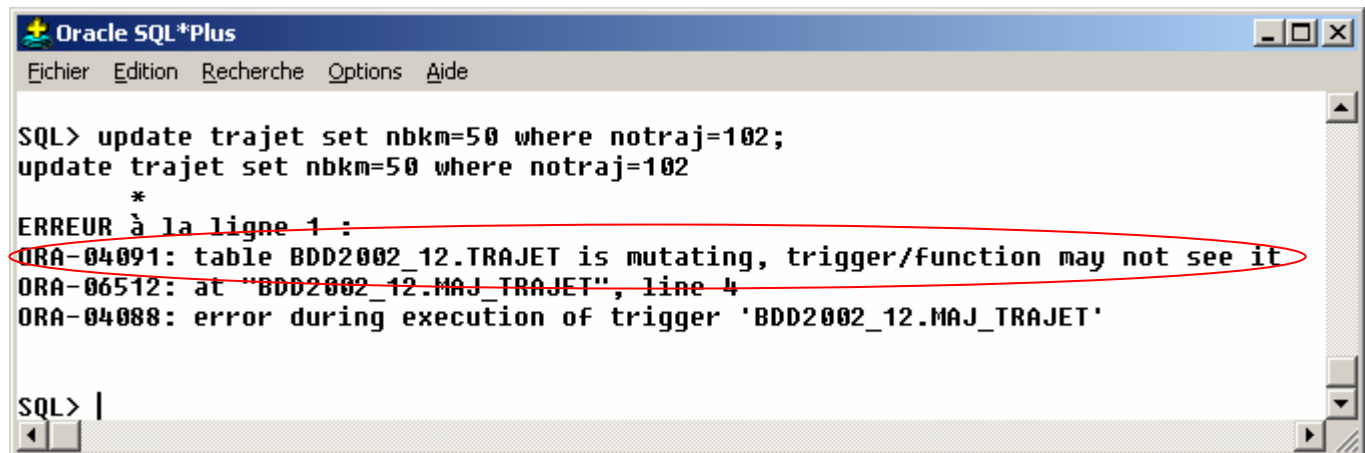


Erreur d'exécution «MutatingTable»

- Définition : Une table est en mutation (mutating) si
 - elle est en train d'être modifiée par une instruction UPDATE, INSERT, DELETE, *ou*
 - elle a besoin d'être mise à jour à cause des effets de l'action DELETE CASCADE.
- L'erreur « mutating table » se produit si on essaye de lire ou de modifier une table en mutation.
- Cette erreur n'a lieu qu'avec *un trigger au niveau de tuple*.

Erreur – « Mutating Table »

```
CREATE OR REPLACE TRIGGER maj_trajet
  BEFORE UPDATE OF villedep, villearr, nbkm ON trajet
  FOR EACH ROW
  DECLARE      km trajet.nbkm%type;
BEGIN
  SELECT nbkm INTO km FROM trajet
  WHERE villedep=:new.villedep AND villearr=:new.villearr;
  IF km <> :new.nbkm THEN
    RAISE_APPLICATION_ERROR(-20010,'Erreur');
  END IF;
END;
```

A screenshot of the Oracle SQL*Plus command-line interface. The window title is "Oracle SQL*Plus". The menu bar includes "Fichier", "Edition", "Recherche", "Options", and "Aide". The command prompt shows the following sequence of commands and output:
SQL> update trajet set nbkm=50 where notraj=102;
update trajet set nbkm=50 where notraj=102
*
ERREUR à la ligne 1 :
ORA-04091: table BDD2002_12.TRAJET is mutating, trigger/function may not see it
ORA-06512: at "BDD2002_12.MAJ_TRAJET", line 4
ORA-04088: error during execution of trigger 'BDD2002_12.MAJ_TRAJET'
The error messages are circled in red. The prompt "SQL> |" is visible at the bottom.

```
Oracle SQL*Plus
Fichier Edition Recherche Options Aide

SQL> update trajet set nbkm=50 where notraj=102;
update trajet set nbkm=50 where notraj=102
*
ERREUR à la ligne 1 :
ORA-04091: table BDD2002_12.TRAJET is mutating, trigger/function may not see it
ORA-06512: at "BDD2002_12.MAJ_TRAJET", line 4
ORA-04088: error during execution of trigger 'BDD2002_12.MAJ_TRAJET'

SQL> |
```

Cas exceptionnel

Si le trigger est:

- BEFORE INSERT ou AFTER INSERT
- Déclenché par une insertion d'une seule ligne.

⇒ **La table associée du trigger n'est pas en mutation**

Exemple : cas exceptionnel

```
/*RI : un adhérent ne peut réserver que 5 livres au maximum*/  
CREATE OR REPLACE TRIGGER ins_reservation  
  AFTER INSERT ON reservation  
  FOR EACH ROW  
  DECLARE nb_re NUMBER(1);  
BEGIN  
  SELECT COUNT(*) INTO nb_re FROM reservation  
  WHERE numAdh = :new.numAdh;  
  IF nb_re>5 THEN  
    RAISE_APPLICATION_ERROR(-20010,'Trop de reservation');  
  END IF;  
END;
```

**SQL>Insert into Reservation
values ('00523', 'Base de données', '30/05/03')**

*Il n'y a pas d'erreur « mutating table » car
cette instruction n'insère qu'une seule ligne*

**SQL>Insert into Reservation
select '00523', titre, '30/05/03'
from livre where nom='Adiba';**

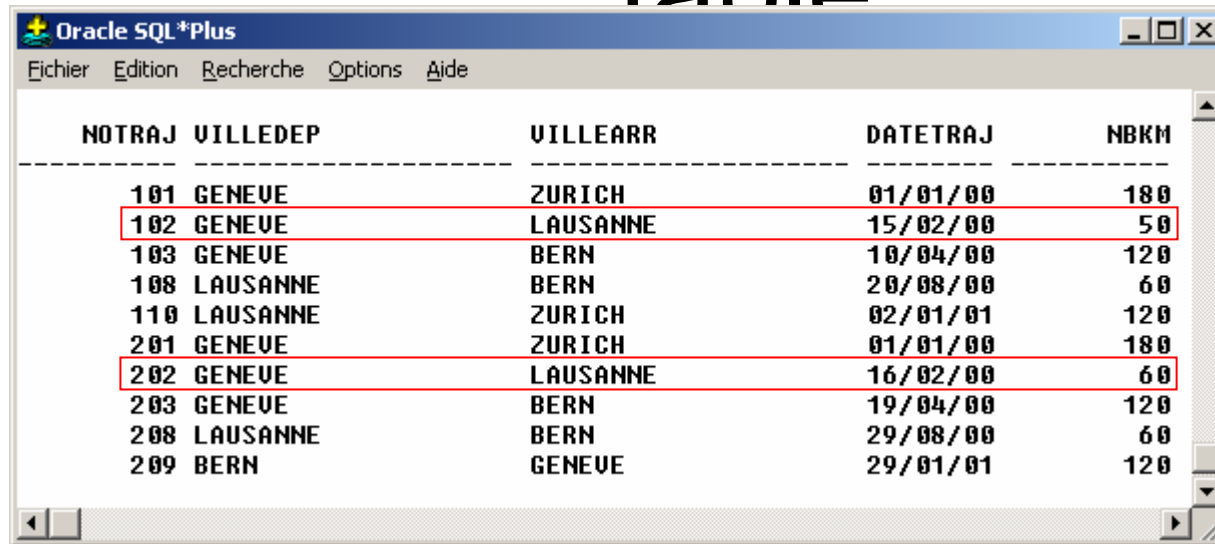
L'erreur « mutating table » se produit

Solutions pour « Mutating Table »

- Solution généralement utilisée afin d'éviter l'erreur « mutating table »

Transformer un trigger au niveau de tuple à celui au niveau d'instruction

Correction d'erreur : Mutating table



NOTRAJ	VILLEDEP	VILLEARR	DATETRAJ	NBKM
101	GENEVE	ZURICH	01/01/00	180
102	GENEVE	LAUSANNE	15/02/00	50
103	GENEVE	BERN	10/04/00	120
108	LAUSANNE	BERN	20/08/00	60
110	LAUSANNE	ZURICH	02/01/01	120
201	GENEVE	ZURICH	01/01/00	180
202	GENEVE	LAUSANNE	16/02/00	60
203	GENEVE	BERN	19/04/00	120
208	LAUSANNE	BERN	29/08/00	60
209	BERN	GENEVE	29/01/01	120

- Utiliser un trigger au niveau d'instruction => ne peut plus consulter les valeurs des lignes affectées.
Nouvelle action: Vérifier s'il existe deux tuples (deux trajets) dont les villes de départ et d'arrivée sont les mêmes mais les distances en KM sont différentes.

101	GENEVE	ZURICH	180
102	GENEVE	LAUSANNE	50
...			
209	BERN	GENEVE	120

101	GENEVE	ZURICH	180
102	GENEVE	LAUSANNE	50
...			
202	GENEVE	LAUSANNE	60

Mutating Table – Solution (1)

```
CREATE OR REPLACE TRIGGER maj_sur_trajet
AFTER UPDATE OF villeddep, villearr, nbkm ON trajet
DECLARE
    CURSOR cur_trajet_1 IS
        SELECT villeddep, villearr, nbkm FROM trajet
            GROUP BY villeddep, villearr, nbkm;
    CURSOR cur_trajet_2 IS
        SELECT villeddep, villearr, nbkm FROM trajet
            GROUP BY villeddep, villearr, nbkm;

    vd1 trajet.villeddep%TYPE;
    va1 trajet.villearr%TYPE;
    nbkm1 trajet.nbkm%TYPE;
    vd2 trajet.villeddep%TYPE;
    va2 trajet.villearr%TYPE;
    nbkm2 trajet.nbkm%TYPE;

    var_exit NUMBER(1) :=0;
```

Transformer
le trigger de
niveau tuple
au niveau
instruction

Mutating Table – Solution (1)

```
BEGIN
  OPEN cur_trajet_1;
  LOOP
    FETCH cur_trajet_1 INTO vd1, va1, nbkm1;
    OPEN cur_trajet_2;
    LOOP
      FETCH cur_trajet_2 INTO vd2, va2, nbkm2;
      IF vd1=vd2 and va1=va2 and nbkm1<>nbkm2 THEN
        var_exit:=1;
      END IF;
      EXIT WHEN (cur_trajet_2%NOTFOUND OR var_exit=1);
    END LOOP;
    CLOSE cur_trajet_2;
    EXIT WHEN (cur_trajet_1%NOTFOUND OR var_exit=1);
  END LOOP;
  CLOSE cur_trajet_1;
  IF var_exit=1 THEN
    RAISE_APPLICATION_ERROR(-20101,'Invalid update');
  END IF;
END;
```

Si deux
tuples ne
satisfont pas
RI, lancer
une erreur