M3106 – BD Avancée

02/09/19

***Contraintes d’intégrité complexes : Triggers (did you just misbase me ?!)***

{} : répétition 1 ou n fois

[] : optionnel

**Contraintes d’intégrité – rappels :**

CREATE TABLE nom\_relation

( nom\_att1 type1 [DEFAULT val\_defaut ]

{ [CONSTRAINT nom\_cont] contrainte\_d\_attribut},

Nom\_att2 type2 …

…

{[CONSTRAINT nom\_cont] contrainte\_de\_relation}

)

**Contraintes d’attribut :**

* Primary key
* Unique
* Not null

----------------------------------------------------

* References nom\_relation(attribut)
* Check (condition)

**Contraintes de relation :**

* Primary key(liste\_attruibuts)
* Foreign key(liste\_attributs)
* References nom\_relation(liste\_attributs)

----------------------------------------------------

* Unique (liste\_attributs)
* Check (condition)

Triggers = déclencheurs :

* *Procédure stockée associée à une table* *et déclenchée automatiquement* par des *évènements liés à des actions* sur cette table.
* Les déclencheurs complètent les dispositifs de gestion des *contraintes d’intégrité* (voir transparents précédents) en permettant *l’application de règles plus complexes*. Ce sont des fonctionnalités de *Bases de Données Actives* (BDA).

Actions / évènements :

* Actions :

*Insertion – Suppression – MAJ* table

* Evènements :

*Avant – Après* action

* 6 Types de triggers
* Procédure stockée :

*A chaque trigger est associée une procédure* écrite *en langage PL/SQL*

PL : Procedure Language

* Déclenchement :

*La procédure est déclenchée* (exécutée) lorsque *l’évènement associé au trigger survient.*

*SYNTAXE SQL D’UN TRIGGER :*

**CREATE TRIGGER nom\_trigger**

**BEFORE | AFTER**

**INSERT | DELETE | UPDATE ON nom\_table**

**[FOR EACH ROW]**

*--Bloc PL/SQL contenant traitement à effectuer*

/

Variables « *spécifiques* » :

* :NEW.nom\_attribut :

Valeur d’un attribut (ici nom\_attribut) après mise à jour

* :OLD.nom\_attribut :

Valeur d’un attribut avant mise à jour

* Ex : INSERT INTO CLIENT (1, ‘Nouveau’)

:NEW.Numcli vaut 1

:NEW.Nom vaut ‘Nouveau’

* Ex : DELETE FROM CLIENT WHERE NumCli = 1

:OLD.NumCli vaut 1

***Exemple : Test de la clé primaire sur la table client***

CREATE TRIGGER trig\_cp

BEFORE INSERT ON Client

FOR EACH ROW

DECLARE

N INTEGER ;

Cle\_existe EXCEPTION ;

Cle\_nulle EXCEPTION ;

BEGIN

--Valeur nulle

IF :NEW.numcli IS NULL THEN RAISE cle\_nulle; END IF;

--Existence de la clé primaire

SELECT COUNT (numcli) INTO n FROM client

WHERE numcli = :NEW.numcli;

IF n>0 THEN RAISE cle\_existe; END IF;

EXCEPTION

WHEN cle\_existe THEN RAISE\_APPLICATION\_ERROR(-2050,’Clé primaire déjà utilisée !’) ;

WHEN cle\_nulle THEN RAISE\_APPLICATION\_ERROR(-20502, ‘La clé primaire doit avoir une valeur !’) ;

END ;

/

**La normalisation**

***But :***

* Garantir la qualité du schéma en contrôlant la redondance.
* Prévenir les anomalies de MAJ (insertion, suppression, modification).
* Normalisation = analyse des Dépendances Fonctionnelles (DF) & décomposition des relations.

***Plan :***

* Dépendances fonctionelles (DF)
* DF Elémentaires (DFE)
* Définitions
* Décomposition des relations
* Conclusion

***DF :***

Soit R(X,Y,Z) ou X,Y,Z sont des groupes d’attributs (avec Z eventuellement vide).

Il existe une **Dépendance Fonctionnelle (DF)** entre X et Y notée **X -> Y** **si chaque valeur de X détermine une valeur de Y quel que soit l’instant considéré.**

Axiomes d’Armstrong (1974)

* **Réflexivité :**

Si X est inclus dans Y alors Y -> X (DF triviale)

* **Augmentation :**

Si X -> Y alors X, Z -> Y, Z

* **Transitivité :**

Si X -> Y et Y -> Z alors X -> Z

* **Union :**

Si X -> Y et X -> Z alors X -> Y, Z

* **Décomposition :**

Si X -> Y, Z alors X -> Y et X -> Z

* **Composition :**

Si X -> Y et W -> Z alors X, W -> Y, Z

* **Pseudo-transitivité :**

Si X -> Y et Y, W -> Z alors X, W -> Z

***DFE***:

X -> Y est une **DF élémentaire** si **Y ne dépend pas d’un sous esemble de X** (X est la plus petite quantité d’information déterminant Y).

N.B :

Y est un attribut unique n’appartenant pas à X ( Y pas inclus dans X) et il n’existe pas X’ inclus dans X tel que X’ -> Y.

***Clé :***

Groupe minimal d’attributs qui détermine tous les autres

Dans R (X,Y,Z)

X est une clé

Si X -> Y, Z est une DFE

(il n’existe pas A inclus dans X tel que A -> Y,Z)

* **Attribut clé :**

Attribut appartenant au moins à une clé.

* **Attribut non clé :**

Attribut n’appartenant à aucune clé.

Graphe des DFE :

Soit la relation R(A, B, C, D, E) avec les DFE :

A -> B, C

A, D -> E

Le graphe des DFE est un graphe orienté dont les nœuds sont les attributs et les arcs sont les DFE.

A C

B

E

D

Le graphe des DFE permet permet de determiner la clé de la relation R (A,D, B, C, E)