



# Développement Avancé d'Applications Web

– TP 3 –

## Contraintes & Triggers

**Dr Bouanaka Chafia**

NTIC

chafia.bouanaka@univ-constantine2.dz

### Etudiants concernés

Faculté/Institut	Département	Niveau	Spécialité
Nouvelles technologies	TLSI	Licence 3	Génie Logiciel (GL)

# Plan du TP

- Contraintes
- Contraintes de domaines
- Triggers

# **Section 1:**

## **Les Contraintes de domaines**

Les SGBD permettent de gérer plusieurs types de contraintes :

- **Les contraintes d'intégrité :**

- Clé primaire Primary Key
- Clé étrangère Foreign key
- Clé secondaire Unique
- Valeurs non nulles

- **Contraintes sur les attributs :**

- Contraintes sur les valeurs des attributs
- Contraintes sur les tuples d'une table

- **Les triggers**

# Contraintes

## Contraintes de domaine

### Définition

- Il s'agit de définir l'ensemble des valeurs que peut prendre un attribut.
- Ces contraintes sont décrites dans **la définition d'un attribut**, directement après **son type** et sa **longueur**.

### Exemples de contraintes de domaine

- **NOT NULL** : on impose que l'attribut possède une valeur
- **DEFAULT** : on spécifie une valeur par défaut dont le type doit correspondre au type de l'attribut
- **UNIQUE** : interdit qu'une colonne contienne deux valeurs identiques

# Contraintes

## Contrainte sur les attributs : check

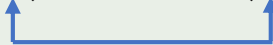
### Définition

- La Clause **check** permet de spécifier **une contrainte** qui doit être **vérifiée à tout moment** par les tuples de la table.
- cette clause permet de définir une contrainte lors de la déclaration d'un attribut pour
  - exprimer le domaine de valeurs d'un attribut
  - Vérifier une contrainte

### Exemple 1:

Contrainte pour exprimer une clé étrangère :

Soit  $R = (A, B)$        $S = (B, C)$



```
CREATE TABLE R (  
    A integer,  
    B integer Check ( B IN (Select B FROM S))  
)
```

# Contraintes

## Contrainte sur les attributs

### Clause CHECK

- Une contrainte peut définir le domaine de valeurs d'un attribut

### Exemple 2: Contrainte domaine de valeurs

Soit la table `Personne = (Nom, Sexe, Age)`

- On désire exprimer la contrainte que le domaine de valeurs de l'attribut `Sexe` soit `{'H','F'}`
- On tente ensuite d'insérer le tuple `('Rayene', 'D', '20')`

# Contraintes

## Contrainte sur les attributs

```
1 • CREATE TABLE Personne (  
2     Nom VARCHAR(50),  
3     Sexe CHAR(1) CHECK (Sexe in ('H', 'F')),  
4     Age INTEGER  
5 );  
6 • INSERT INTO Personne VALUES('Rayene', 'D', '20');
```

Output				
Action Output				
#	Time	Action	Message	
✓ 1	06:46:42	CREATE TABLE Personne ( Nom VARCHAR(50), Sexe CHAR(1) CHECK (Sexe i...	0 row(s) affected	
✗ 2	06:46:43	INSERT INTO Personne VALUES('Rayene', 'D', '20')	Error Code: 3819. Check constraint 'personne_chk_1' is violated.	

L'insertion n'est pas faite à cause de la violation de la contrainte sur les valeurs de l'attribut Sexe



# Contraintes

## Contraintes sur les Tuples

### Clause CHECK

- Une contrainte peut porter sur plusieurs attributs

### Exemple 3: contrainte sur plusieurs attributs

Soit la table Personne = (Nom, Sexe, Age)  
On désire exprimer la contrainte que l'attribut Sexe ait comme valeurs {'H','F'} et que le nom d'une femme ne commence pas par 'M.'

```
1 • ○ CREATE TABLE Personne(  
2     Nom VARCHAR(30),  
3     Sexe CHAR(1) CHECK (Sexe IN ('H', 'F')),  
4     Age Integer,  
5     CHECK ((Sexe = 'H') OR ( Nom NOT LIKE 'M.%'))  
6 )
```

Contrainte sur les valeurs  
d'attribut

Contrainte sur deux  
attributs

# Contraintes

## Contraintes sur les Tuples

```
1 • CREATE TABLE Personne(  
2     Nom VARCHAR(30),  
3     Sexe CHAR(1) CHECK (Sexe IN ('H', 'F')),  
4     Age Integer,  
5     CHECK ((Sexe = 'H') OR ( Nom NOT LIKE 'M.%'))  
6 );  
7 • INSERT INTO Personne VALUES('M. Mohamed', 'H', '20');  
8 • INSERT INTO Personne VALUES('M. Rayene', 'F', '20');
```

Result Grid		Filter Rows: <input type="text"/>	Export:	Wrap Cell Content:
	Nom	Sexe	Age	
▶	M. Mohamed	H	20	

✓	4	07:19:25	INSERT INTO Personne VALUES('M. Mohamed', 'H', '20')	1 row(s) affected
✗	5	07:19:26	INSERT INTO Personne VALUES('M. Rayene', 'F', '20')	Error Code: 3819. Check constraint 'personne_chk_2' is violated.

Le deuxième tuple a violé la contrainte sur le nom d'une femme

# Contraintes

## Contraintes sur les Tuples

### Nommer les contraintes

- pour donner un nom à une contrainte, il faut la précéder par le mot clé **CONSTRAINT**
- Le mot clé **CONSTRAINT** est placé avant la clause **CHECK**
- Nommer une contrainte permet de la référencer

```
1  CREATE TABLE Personne(  
2      Nom VARCHAR(30),  
3      Sexe CHAR(1) CONSTRAINT SexeVal CHECK (Sexe IN ('H', 'F')),  
4      Age Integer,  
5      CONSTRAINT NomF CHECK ((Sexe = 'H') OR ( Nom NOT LIKE 'M.%'))  
6  );
```

Nom de la contrainte

Nom de la contrainte

## **Section 2:**

# **Les Contraintes de domaines**

# Contraintes

## Contraintes d'intégrité

### Définition

- Elles spécifient la clé primaire d'une table via la clause PRIMARY KEY.
- Une clé primaire doit toujours avoir **une valeur** déterminée et **unique** pour la table.
- Quand une clé primaire est constituée de plusieurs attributs (**clé segmentée**), la clause **PRIMARY KEY** est placée après la définition des attributs, séparée par une virgule.

### Exemple de déclaration

```
1 • ○ CREATE TABLE Inscription(  
2     ID_Cours  Integer,  
3     ID_Etudiant INTEGER ,  
4     Primary key (ID_Cours, ID_Etudiant),  
5     Foreign key(ID_Cours) references Cours(ID),  
6     Foreign key(ID_Etudiant) references Etudiant(ID)  
7 );
```

Clé segmentée

### Remarques (Clés d'une table)

- Tous les attributs d'une clé segmentée doivent être spécifiés **NOT NULL**
- **PRIMARY KEY** peut aussi être séparée de la définition des attributs même s'il n'y a qu'un seul attribut
- Pour chaque table, il n'existe qu'une seule clé primaire
- Dans beaucoup de SGBD, un index est automatiquement construit sur la clé primaire.

# **Section 3: Les Triggers**

# Les Triggers(Déclencheurs)

## Définition

### Définition : Trigger (Déclencheur)

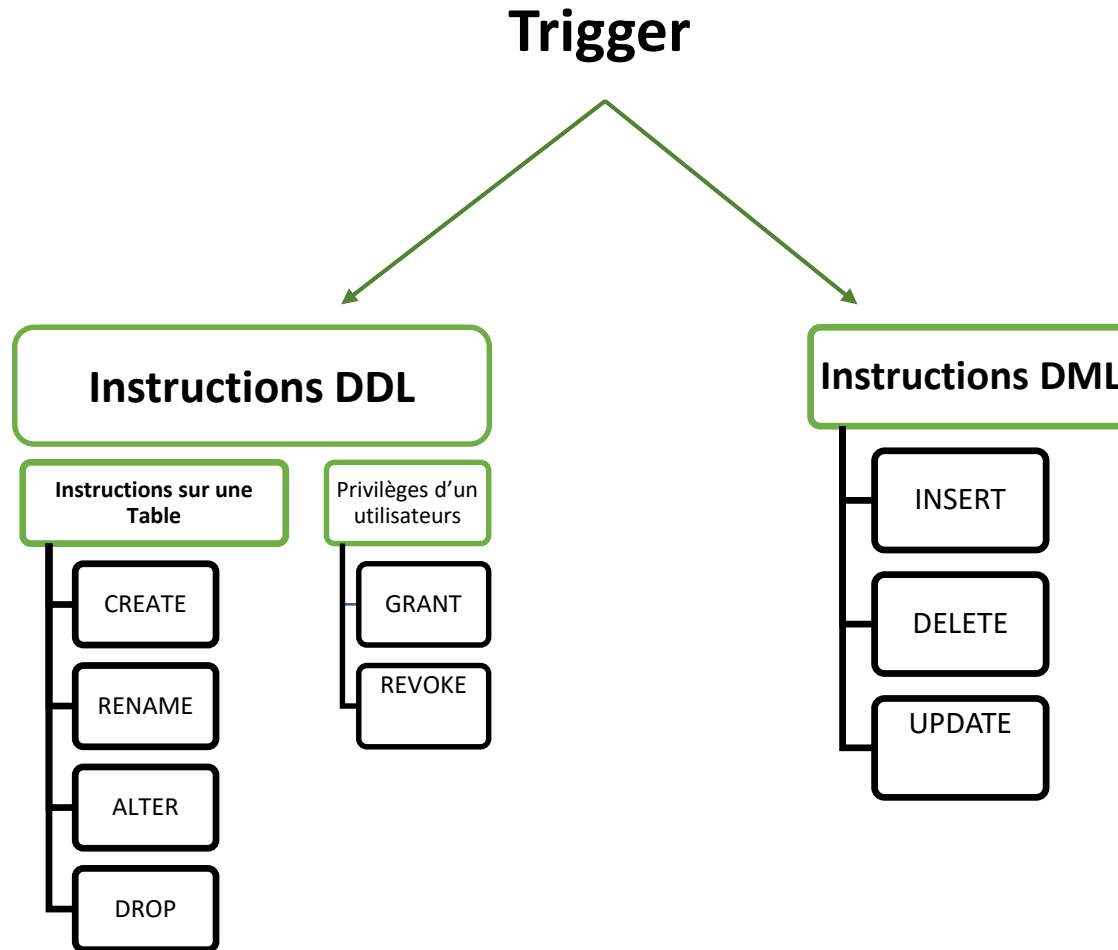
- Un trigger est :
  - Un programme déclenché par un événement
  - N'est pas appelé explicitement par une application
  - s'exécute automatiquement
- Événements déclencheurs :
  - Instruction LMD : INSERT, UPDATE, DELETE
  - Instruction LDD : CREATE, ALTER, DROP
  - Démarrage ou arrêt de la base
  - Connexion ou déconnexion d'utilisateur
  - Erreur d'exécution
- Usage fréquent
  - Contraintes non exprimables sur les tables
  - Modification des dépendances



# Triggers

## Définition

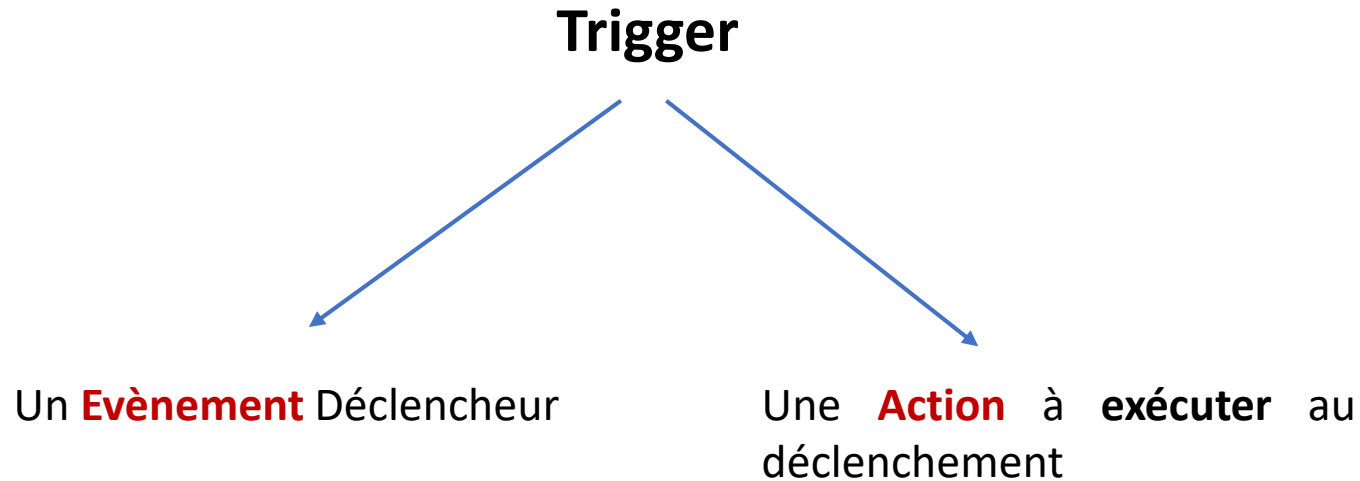
Les triggers peuvent être associés à des instructions DDL(Data Definition Language) ou DML(Data Manipulation Language):



# Triggers

## Définition

Un Trigger est constitué de deux parties :



### Important

- Le code de la partie Action est écrit dans le langage PL/SQL
- Le langage PL/SQL sert à définir des procédures stockées et des triggers
- Le langage PL/SQL sera présenté après la section Triggers

# Triggers

## Structure d'un Trigger

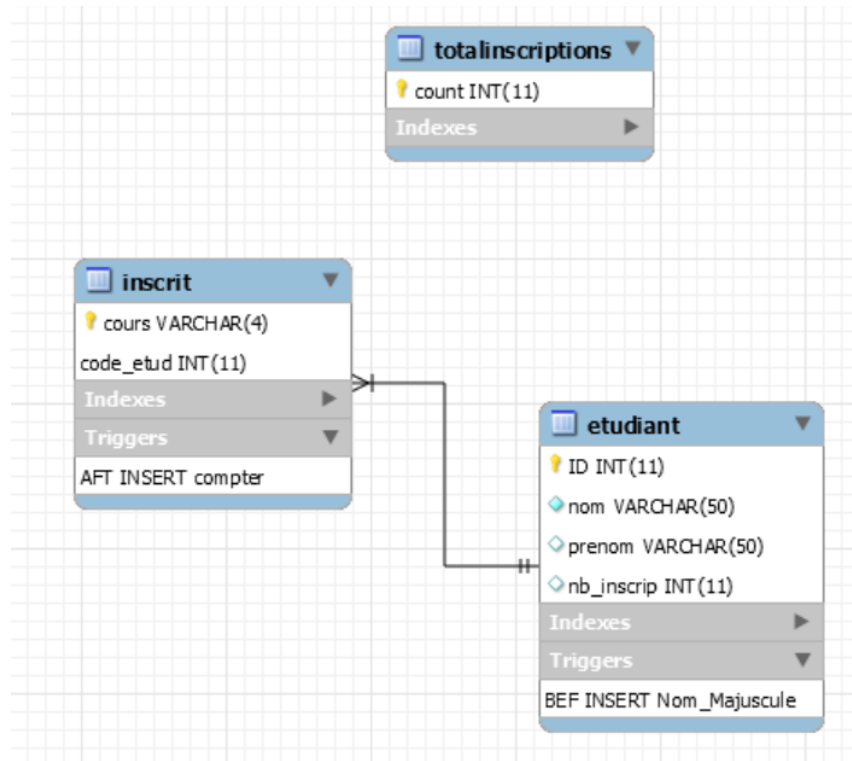
```
CREATE [OR REPLACE] TRIGGER <nom_trigger>
{BEFORE|AFTER}
{INSERT|DELETE|UPDATE [OF colonnes]}
ON <nom_table>
[FOR EACH ROW] ← row trigger si présent
[DECLARE]
-- déclaration de variables, exceptions
-- curseurs
BEGIN
-- bloc action
-- ordres SQL et PL/SQL
END;
/
```

# Triggers

## Exemples de Triggers

Soit le schema de la BD Enseignementbd, défini par :

- **etudiant** (ID : int, nom : varchar(50), prenom : varchar(50), nb\_inscript : int)
- **inscrit**(cours : varchar(4), Code\_etud : int)
- Totalinscription (count : int)



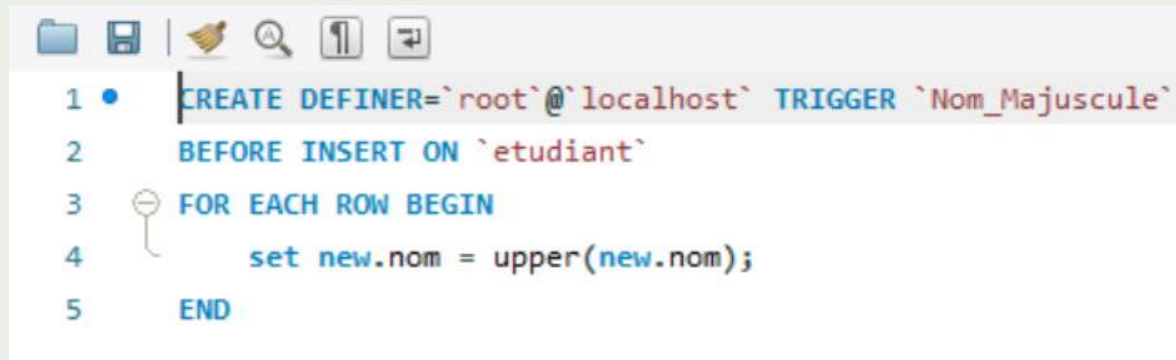
# Triggers

## Exemples de Trigger

### Triggers : Exemple 1

On désire ajouter un trigger qui transforme les noms des étudiants en majuscule.

Le trigger a la structure suivante :

A screenshot of a SQL IDE window. The window has a toolbar at the top with icons for file operations (folder, save, print), search (magnifying glass), and editing (undo, redo). Below the toolbar, the SQL code is displayed on a light background with line numbers on the left. The code is as follows:

```
1 CREATE DEFINER='root'@'localhost' TRIGGER `Nom_Majuscule`  
2 BEFORE INSERT ON `etudiant`  
3 FOR EACH ROW BEGIN  
4     set new.nom = upper(new.nom);  
5 END
```

# Triggers

## Exemples de Trigger

### Triggers : Exemple 1

On désire ajouter un trigger qui transforme les noms des étudiants en majuscule.

Le trigger a la structure suivante :

- ```
create Trigger Majuscule
before insert on Etudiant
for each row
set new.nom = upper(new.nom), new.prenom = upper(new.prenom);
```

# Triggers

## Exemples de Trigger

### Triggers : Exemple 2

On désire définir un trigger permettant de :

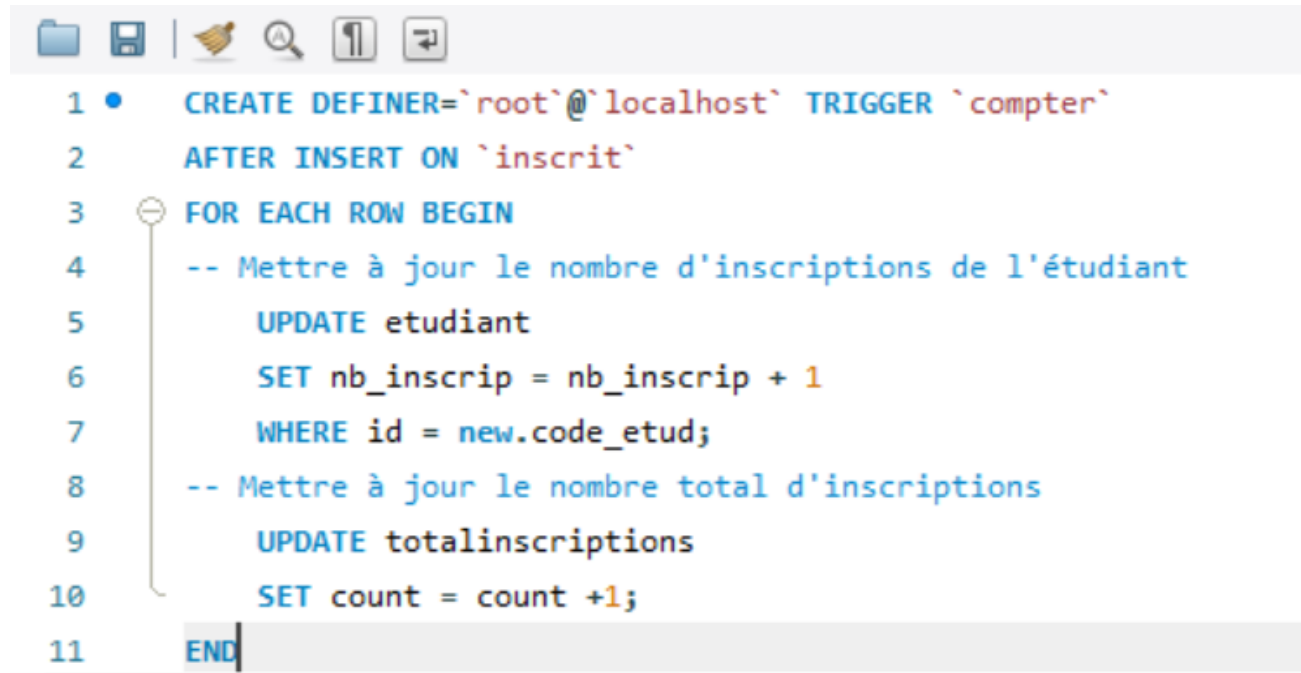
- Compter le nombre d'étudiants inscrits à tous les cours
- Compter le nombre d'inscriptions de chaque étudiant

Pour se faire, il faut :

- Définir le trigger '**compter**' sur la table **inscrire**, il est déclenché après chaque **insert** dans cette table
- Mettre à jour l'attribut **nb\_inscrip** de la table **etudiant** après chaque insertion dans la table inscrit

# Triggers

## Exemples de Triggers

A screenshot of a SQL IDE window. The title bar shows icons for file operations (folder, save, print, search, undo, redo). The editor contains a SQL script for creating a trigger named 'compter'. The script is as follows:




```
1 CREATE DEFINER='root'@'localhost' TRIGGER `compter`  
2 AFTER INSERT ON `inscrit`  
3 FOR EACH ROW BEGIN  
4   -- Mettre à jour le nombre d'inscriptions de l'étudiant  
5   UPDATE etudiant  
6     SET nb_inscrip = nb_inscrip + 1  
7     WHERE id = new.code_etud;  
8   -- Mettre à jour le nombre total d'inscriptions  
9   UPDATE totalinscriptions  
10  SET count = count +1;  
11 END
```

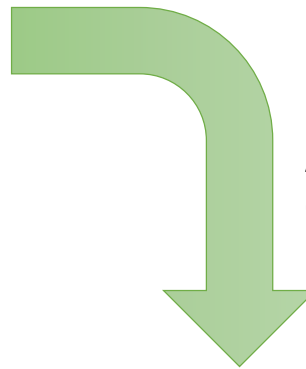


# Triggers

## Exemples de Trigger


Table étudiant : Données avant exécution de l'insertion

|                                                                                 |      |                                                                                   |                                                                                                                     |            |
|---------------------------------------------------------------------------------|------|-----------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|------------|
| Result Grid                                                                     |      |  |  Filter Rows: <input type="text"/> |            |
|                                                                                 | ID   | nom                                                                               | prenom                                                                                                              | nb_inscrip |
|                                                                                 | 1    | Benali                                                                            | Cherif                                                                                                              | 0          |
|                                                                                 | 2    | Benmimi                                                                           | Mohamed                                                                                                             | 0          |
|                                                                                 | 3    | Sari                                                                              | Ali                                                                                                                 | 0          |
|                                                                                 | 4    | Selami                                                                            | Ali                                                                                                                 | 0          |
|                                                                                 | 5    | Saadi                                                                             | Mohamed                                                                                                             | 0          |
|  | NULL | NULL                                                                              | NULL                                                                                                                |            |



Après exécution de l'insertion  
et du **trigger Nom\_Majuscule**

Result Grid



|   | ID | nom     | prenom  | nb_inscrip |
|---|----|---------|---------|------------|
|   | 1  | BENALI  | Cherif  | 0          |
|   | 2  | BENMIMI | Mohamed | 0          |
|   | 3  | SARI    | Ali     | 0          |
|   | 4  | SELAMI  | Ali     | 0          |
|   | 5  | SAADI   | Mohamed | 0          |
| » |    | NULL    | NULL    | NULL       |