Chapitre 3 : Le SGBD MySQL

I. Qu'est-ce que MySQL ?

I. 1. Introduction

MySQL est un Système de Gestion de Bases de Données Relationnelles (SGBDR) open source implémentant le langage SQL. Il propose une version open source gratuite (version communautaire : Open-Source Community Edition) qui permet à l'utilisateur d'accéder au code source et de le modifier. Cette version est disponible sur plus de 20 plateformes dont Linux, Unix, Mac Os, Windows, etc. L'autre version est une version commerciale (MySQL Entreprise Edition) permettant un accès aux dernières fonctionnalités du logiciel et au support fourni par Oracle, propriétaire et développeur actuel de MySQL.

Parmi les plus populaires au monde, il est connu pour délivrer de hautes performances dans le stockage de larges volumes de données (notamment dans le Big Data) ou la Business Intelligence. Fondé en 1994, MySQL a été racheté par Sun Microsystems en 2008 et appartient à Oracle Database depuis 2010. MySQL est utilisé par de nombreuses entreprises dans le monde, dont :

- ✓ Google, Yahoo!, YouTube, et Adobe, dans le digital ;
- ✓ Airbus, Alstom, et Alcatel-Lucent dans l'industrie ;
- ✓ Crédit agricole, dans le secteur de la banque et de l'assurance ;
- ✓ AFP, Reuters, BBC News, Ernst & Young dans le secteur des médias.

Il est développé par MySQL AB qui été acheté le 16 janvier 2008 par Sun Microsystems. En 2009, Sun Microsystems a été acquis par Oracle Corporation. C'est un serveur de BD rapide, fiable et facile à utiliser. Il fonctionne en mode client/serveur et en mode embarqué.

I. 2. Historique des différentes versions

La première version de MySQL est apparue le 23 mai 1995. Depuis lors, MySQL n'a cessé de s'améliorer. Ainsi, beaucoup de nouvelles versions sont apparues depuis 1995. De 2003 à 2018, il est passé de la version 5.0 à la version 8.0.

- ✓ Version 5.0 : première version en décembre 2003, stable depuis octobre 2005
- ✓ Version 5.1 : première version en novembre 2005,
- ✓ Version 5.2 : distribuée en avant-première en février 2007,
- ✓ Version 5.5 : Version stable depuis octobre 2010
- ✓ Version 5.6 : Version stable depuis février 2013

- ✓ Version 5.7 : Version stable depuis octobre 2015
- ✓ Version 6.0 : première version alpha en avril 2007, abandonnée depuis le rachat de MySQL par Oracle en décembre 2010
- ✓ Version 8.0 : Version stable depuis avril 2018

I. 3. Les avantages de MySQL

a. Portabilité : MySQL est développé avec les langages C et C++. Il tourne sur de nombreuses plates-formes. Un des outils qui font sa force est son système d'**API** (**A**pplication **P**rogramming **I**nterface). Il dispose d'API pour C, C++, Java, Eiffel, Perl, PHP, Python, Ruby et Tcl. Il peut être utilisé sur un serveur avec plusieurs processeurs. Il fournit des moteurs de tables transactionnels et non transactionnels.

b. Types de colonnes : Il existe plusieurs types de colonnes : FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, YEAR, SET et ENUM.

c. Commandes et fonctions : Il supporte les commandes et fonctions du SQL. Les tables et colonnes peuvent porter des noms de fonction. Possibilité d'utiliser deux tables se situant dans deux bases de données différentes.

d. Sécurité : Le système de droits et de mots de passe est très souple et sécuritaire. Les mots de passe sont chiffrés à chaque fois qu'ils doivent être envoyés, même lors des connexions.

I. 4. Les moteurs de stockage de données de MySQL

I. 4. 1. Qu'est-ce qu'un moteur de stockage ?

Un moteur de stockage de données est un ensemble d'algorithmes utilisés par un SGBDR pour stocker les données d'une base et y accéder au moyen d'une requête SQL. La plupart des SGBDR proposent un moteur unique, créé pour être le plus efficace possible dans tous les cas. MySQL et MariaDB, par contre, proposent à l'administrateur de la base de choisir pour chaque table de sa base quel moteur il désire utiliser. On se retrouve ainsi avec des bases où plusieurs moteurs peuvent coexister. C'est un choix de conception qui a ses avantages mais aussi ses inconvénients. Il existe plusieurs moteurs de stockages de données parmi lesquels :

- ✓ MyISAM ;
- ✓ InnoDB ;
- ✓ Merge ;
- ✓ Memory ;
- ✓ CSV.

Pour spécifier explicitement le moteur de stockage que l'on souhaite utiliser pour une table, on l'indique avec l'option ENGINE :

- ✓ lors de la création de la table avec la syntaxe suivante :

```
Create Table Nom_Table
(
    Att1    Domaine1,
    Att2    Domaine2
) ENGINE = MoteurDeStockage ;
```

- ✓ après création de la table avec la syntaxe suivante :

```
Alter Table Nom_Table Engine = MoteurDeStockage ;
```

Il est possible de définir un moteur par défaut en le spécifiant dans le fichier de configuration de MySQL soit définitivement, soit pour la session active seulement :

- ✓ Pour la session active seulement, on exécute la requete suivante :

```
Set Storage_engine = MoteurDeStockage ;
```

- ✓ Pour le faire définitivement, cela se fait au moyen de la directive suivante du fichier de configuration :

```
Default-storage-engine = MoteurDeStockage
```

Remarque :

- ✓ Pour les versions inférieures à MySQL 5.5 le moteur par défaut est MyISAM ;
- ✓ Pour les versions supérieures ou égales à 5.5, le moteur par défaut est InnoDB.

I. 4. 2. MyISAM

C'est le moteur par défaut des tables MySQL de la version 3.23.0 jusqu'à la version précéent MySQL 5.5. Chaque table **MyISAM** est stockée en trois fichiers. Les fichiers portent le nom de la table, et ont une extension qui spécifie le type de fichier. Le fichier **.frm** stocke la définition de la table. L'index est stocké dans un fichier avec l'extension **.MYI** (**MYIndex**), et les données sont stockées dans un fichier avec l'extension **.MYD** (**MYData**).

Il est possible de vérifier ou réparer une table **MyISAM** avec l'utilitaire ***myisamchk***. On peut aussi compresser les tables **MyISAM** avec l'utilitaire ***myisampack*** pour réduire leur taille sur le disque.

C'est un moteur non transactionnel assez rapide en écriture et très rapide en lecture. Il ne supporte pas les contraintes d'intégrité référentielle (clés étrangères). Ne gérant pas les

transactions ni les clés étrangères il n'y a donc pas beaucoup de contrôles qui sont généralement gourmands en ressources et prennent du temps. Il gère l'indexation des attributs et même l'index FULLTEXT sur les attributs de type TEXT et le verrouillage des données au niveau table. De plus, MyISAM garde en cache des métadonnées sur la table et ses index, comme le nombre de lignes, la taille perdue à cause de la fragmentation, etc.

I. 4. 3. InnoDB

C'est le moteur par défaut des tables MySQL depuis la version MySQL 5.5. **InnoDB** fournit à MySQL un gestionnaire de table transactionnelle (compatible **ACID**), avec validation (**commits**), annulations (**rollback**) et capacités de restauration après crash. **InnoDB** utilise un verrouillage de lignes, et fournit des lectures cohérentes comme Oracle, sans verrous. Ces fonctionnalités accroissent les possibilités d'utilisation simultanées des tables, et les performances. Il n'y a pas de problème de queue de verrous avec **InnoDB**, car les verrous de lignes utilisent très peu de place. Les tables **InnoDB** sont les premières tables MySQL qui supportent les contraintes d'intégrité référentielle (**FOREIGN KEY**). **InnoDB** a été conçu pour maximiser les performances lors du traitement de grandes quantités de données. Son efficacité n'est égalée par aucun autre moteur de base de données de MySQL. Techniquement, **InnoDB** est un gestionnaire de table placé sous MySQL. **InnoDB** dispose de son propre buffer pour mettre en cache les données et les index en mémoire centrale. **InnoDB** stocke les tables et index dans un espace de table, qui peut être réparti dans plusieurs fichiers. Ceci diffère des tables comme, par exemple, MyISAM où chaque table est stockée dans un fichier différent. **InnoDB** crée un fichier .frm pour la structure de la table et un fichier .ibd pour les données et les index. Les tables **InnoDB** peuvent prendre n'importe quelle taille, même sur les systèmes d'exploitation dont la limite est de 2 Go par fichier. **InnoDB** est utilisé en production dans plusieurs sites où de grandes capacités de stockages et des performances accrues sont nécessaires. **InnoDB** est sous licence GNU GPL License Version 2 (de Juin 1991). Si vous distribuez MySQL et **InnoDB**, et que votre application ne satisfait pas les restrictions de la licence GPL, vous devez acheter une licence commerciale ***MySQL Pro*.**

I. 4. 4. Merge ou MRG_MyISAM

Une table **MERGE** est un groupe de tables **MyISAM** identiques et utilisées comme une seule. "Identique" signifie que toutes les tables ont la même structure de colonnes et d'index. Il est impossible de regrouper des tables qui ont des index dans un ordre différent. Toutefois, une ou plusieurs tables peuvent être compressées avec ***myisampack***.

Lorsque vous créez une table **MERGE**, MySQL crée deux fichiers sur le disque. Les fichiers ont pour nom celui de la table, et ont une extension qui indique le type de fichier. Le fichier **.frm** stocke la définition de la table, et le fichier **.MRG** contient les noms des tables qui doivent être utilisées. Originalement, toutes les tables utilisées dans la même table **MERGE** devaient être dans la même base que cette dernière. Cette restriction a été levée dans MySQL 4.1.1.

Pour le moment, vous avez simplement besoin des droits de **SELECT**, **UPDATE** et **DELETE** sur les tables que vous avez rassemblées dans la table **MERGE**. MERGE se contente de fournir une interface unique pour accéder en lecture à toutes les tables simultanément, et en écriture selon des règles que l'on aura fixées. Les tables peuvent provenir de plusieurs bases de données, si elles sont sur le même serveur physique. Il gère les index de la même manière que MyISAM, sauf Fulltext qu'il ne gère pas. Les tables fusionnées doivent respecter les critères suivants :

- ✓ Être sur le même serveur ;
- ✓ Même schéma ;
- ✓ Mêmes index (sauf FULLTEXT qui sera ignoré) ;
- ✓ Même ordre de déclaration des index.

Chaque table reste disponible, les modifications se répercuteront sur la table fusionnée. En créant une table avec MERGE, on ajoute deux paramètres à la syntaxe générique :

- ✓ UNION($Table_1$, $Table_2$, …, $Table_n$) : Permet de donner la liste des tables à fusionner ;
- ✓ Insert_Method = Valeur : Permet de définir où seront insérées les nouvelles lignes. Valeur prend les valeurs First ou Last

Exemple

```
Create Table Nom_Table
(
    Att1        Domaine1,
    Att2        Domaine2
) ENGINE = MERGE   Union(Table1, Table2)   Insert_Method = Last ;
```

Pour ajouter une nouvelle table à la fusion, il existe trois méthodes :

- ✓ Supprimer la table dans laquelle on a fait la fusion et la recréer avec la nouvelle table dans UNION ;
- ✓ Modifier la table avec la syntaxe :

Alter Table Nom_Table Union (Table1, Table2, …, Tablen) ;

- ✓ Modifier à la main le fichier .MRG créé dans le dossier de données de MySQL puis faire un Flush Tables dans le SGBD pour le forcer à relire les définitions

I. 4. 5. Memory

Le moteur de stockage **MEMORY** crée des tables dont le contenu est stocké en mémoire. Ceci en fait le moteur de stockage le plus rapide que propose MySQL, mais aussi le plus dangereux. Avant MySQL 4.1, les tables **MEMORY** étaient appelées des tables **HEAP**. Depuis 4.1, **HEAP** est un synonyme de **MEMORY**. **MEMORY** est le terme recommandé.

Chaque table **MEMORY** est associée à un fichier sur le disque. Le fichier a le nom de la table, et pour extension **.frm** pour indiquer la définition de la table.

- ✓ Il ne gère pas les champs TEXT ni BLOB.
- ✓ Il ne gère pas les transactions ;
- ✓ Il ne gère pas les contraintes d'intégrité référentielles ;
- ✓ Comme il ne gère pas les champs TEXT, il ne gère pas non plus les index FULLTEXT.

MEMORY est parfait pour stocker des données purement temporaires qui ont besoin d'être traitées rapidement et surtout dont la perte n'engendre pas une incohérence de la base. Un arrêt anormal du serveur engendre une perte des données.

I. 4. 6. CSV

Le moteur CSV a été ajouté à partir de MySQL 4.1.4. Ce moteur stocke les données dans un fichier texte, avec le format CSV dont valeurs séparées par des virgules. Lorsqu'on crée une table CSV, le serveur créé un fichier de définition de table et un fichier de données qui porte le nom de la table :

- ➢ Le fichier de définition de table a le même nom que de table avec l'extension **.frm** ;
- ➢ Le fichier de données a aussi le même nom que la table avec l'extension **.CSV** ;

Le fichier de données est un fichier texte simple. Lorsqu'on stocke des données dans la table, le moteur les écrit au format CSV dans ce fichier. Les valeurs sont séparées par des virgules, les lignes sont séparées par des sauts de ligne. Le moteur ne gère ni l'intégrité référentielle, ni les transactions, ni les index. Il permet une grande interopérabilité entre des systèmes externes à MySQL, car le format CSV est pratiquement universel et est reconnu par tous les tableurs et de nombreux logiciels qui importent et exportent des données dans ce format. Il est en outre extrêmement pratique lorsqu'on désire exporter une table au format CSV compatible avec Excel, il suffit de faire une copie de la table avec ce moteur pour la nouvelle table.// Tous les attributs doivent être Not Null et pas de contrainte Check.

II. Les types de données

L'administrateur d'une base de données devant spécifier le type de chaque attribut de toutes les relations lors de leur création, MySQL prévoit un certain nombre de type qu'il est possible de donner. C'est ainsi que nous avons des types numériques (entiers et décimaux), des types pour les chaines, des types pour les dates et heures etc.

II. 1. Les types numériques

II. 1. 1. Les entiers

Nom	Taille	Borne inferieure	Borne supérieure
TINYINT	2^8	-128	127
TINYINT UNSIGNED	2^8	0	255
SMALLINT	2^{16}	-32768	32767
SMALLINT UNSIGNED	2^{16}	0	65535
MEDIUMINT	2^{24}	-8388608	8388607
MEDIUMINT UNSIGNED	2^{24}	0	16777215
INT*	2^{32}	-2147483648	214748647
INT* UNSIGNED	2^{32}	0	4294967295
BIGINT	2^{64}	-9223372036854775808	9223372036854775807
BIGINT UNSIGNED	2^{64}	0	18446744073709551615

II. 1. 2. Les réels

Nom	Borne inferieure	Borne supérieure
FLOAT	-3.402823466E+38 -1.175494351E-38	1.175494351E-38 3.402823466E+38
DOUBLE*	-1.7976931348623157E+308 -2.2250738585072014E-308	2.2250738585072014E-308 1.7976931348623157E+308

II. 2. Les chaines de caractères

Nom	Longueur
CHAR(M)	Chaîne de taille fixée à M, où 1<M<255, complétée avec des espaces si nécessaire.
CHAR(M) BINARY	Idem, mais insensible à la casse lors des tris et recherches.
VARCHAR(M)	Chaîne de taille variable, de taille maximum M, où 1<M<255, complété avec des espaces si nécessaire.
VARCHAR(M) BINARY	Idem, mais insensible à la casse lors des tris et recherches.
TINYTEXT	Longueur maximale de 255 caractères.
TEXT	Longueur maximale de 65535 caractères.
MEDIUM TEXT	Longueur maximale de 16777215 caractères.
LONGTEXT	Longueur maximale de 4294967295 caractères.
DECIMAL (M, D)*	Simule un nombre flottant de D chiffres après la virgule et de M chiffres au maximum. Chaque chiffre ainsi que la

	virgule et le signe moins (pas le plus) occupe un caractère.

II. 3. Les autres types

II. 3. 1. Dates et heures

Nom	Description
DATE	Date au format anglophone AAAA-MM-JJ
DATETIME	Date et heure au format anglophone AAAA-MM-JJ HH:MM:SS
TIMESTAMP	Affiche la date et l'heure sans séparateur AAAAMMJJHHMMSS
TIMESTAMP (M)	Idem, mais M vaut un entier pair entre 2 et 14. Affiche les M premiers caractères de TIMESTAMP
TIME	Heure au format HH:MM:SS
YEAR	Année au format AAAA

II. 3. 2. Les ensembles

Un attribut de type **SET** peut prendre pour valeur la chaîne vide, **NULL** ou une chaîne contenant une liste de valeurs qui doivent être déclarées lors de la définition de l'attribut pendant la création de la table.

Par exemple, un attribut déclaré comme suit :

SET('Voiture', 'Moto', 'Vélo') NOT NULL ; peut prendre les valeurs suivantes :

- **''** (chaîne vide)
- **'Voiture,Moto'**
- **'Vélo,Voiture,Moto'**
- Et autres combinaisons de listes des trois valeurs définies plus haut.

Un attribut déclaré comme suit :

SET('Voiture', 'Moto', 'Vélo') ; peut prendre la valeur **NULL**, en plus des autres.

On ne peut donner que 64 éléments maximum.

II. 3. 3. Les énumérations

Un attribut de type **ENUM** peut prendre une valeur parmi celles définies lors de la création de la table plus la chaîne vide ainsi que NULL si la définition le permet. Ces valeurs sont exclusivement des chaînes de caractères. Une énumération peut contenir 65535 éléments au maximum.

Syntaxe :

Nom_Attribut ENUM('valeur 1', 'valeur 2', …) NOT NULL ;

Nom_Attribut ENUM('valeur 1', 'valeur 2', …) ;

A chaque valeur est associée un index allant de 0 à n si n valeurs ont été définies. L'index 0 est associé à la chaîne vide, l'index 1 à la première valeur… L'index NULL est associé à la valeur NULL. Si une sélection (SELECT ou WHERE) est faite dans un contexte numérique, l'index est renvoyé. Sinon, c'est la valeur qui est retournée. On peut définir jusqu'à 65535 valeurs distinctes insensibles à la casse.

Exemple

Jour Enum('Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi', 'Samedi', 'Dimanche') ;

Valeur	Index
Null	Null
''	0
'Lundi'	1
'Mardi'	2
'Mercredi'	3
'Jeudi'	4
'Vendredi'	5
'Samedi'	6
'Dimanche'	7

III. Les bases de données natives de MySQL

III. 1. La base de données Information_Schema

C'est une base de données incluse dans le SGBD dans laquelle il y a toutes les informations concernant les bases de données du SGBD. C'est ainsi qu'on peut y trouver entre autres :

- ✓ Les noms de toutes les bases de données se trouvant dans le serveur,
- ✓ Les noms de toutes les tables dans le serveur,
- ✓ Le type de chaque attribut dans le serveur,
- ✓ Les privilèges qui existent,
- ✓ Etc.

Elle est appelée "dictionnaire de données" ou "catalogue système". Elle contient un ensemble de vues systèmes et nom des tables physiques. Chaque utilisateur peut accéder aux données de cette base concernant les tables dont il peut accéder. Parmi ses vues nous avons :

a. SCHEMATA : Elle contient les informations concernant toutes les bases de données crées dans le SGBD. Ses attributs les plus importants sont :

- SCHEMA_NAME qui contient les noms des bases de données,
- DEFAULT_CHARACTER_SET_NAME qui contient le type langue utilisée

Ses enregistrements sont de la forme :

SCHEMA_NAME	DEFAULT_CHARACTER_SET_NAME
Information_schema	utf8
Mysql	Latin1
TP1	Latin1

Pour obtenir les informations concernant une base on exécute la requête suivante :

```
Select * From SCHEMATA Where SCHEMA_NAME = 'nom_base' ;
```

b. TABLES : Elle contient les informations concernant toutes les tables dans le SGBD. Ses colonnes les plus importantes sont :

- TABLE_NAME qui contient les noms des tables dans le système ;
- TABLE_SCHEMA qui contient les noms des bases dans lesquelles se trouvent les tables
- TABLE_TYPE qui contient les types d'objet (table, vues, temporary etc.)
- TABLE_COMMENT qui donne une information sur la table en question.

Ses enregistrements sont de la forme :

TABLE_NAME	TABLE_SCHEMA	TABLE_TYPE	TABLE_COMMENT
Views	Information_schema	Système View	
Host	Mysql	Base Table	Host privilèges
Cours	TP1	Base Table	

Pour obtenir toutes les tables d'une base de données :

```
Select TABLE_NAME From INFORMATION_SCHEMA.Tables Where TABLE_SCHEMA = 'nom_base' ;
```

c. COLUMNS : Elle contient les informations concernant les colonnes des tables. Ses attributs les plus importants sont :

- TABLE_SCHEMA qui contient les noms des bases de données,
- TABLE_NAME qui donne les noms des tables,
- COLUMN_NAME qui donne les noms des colonnes,
- ORDINAL_POSITION qui donne la position de la colonne dans la table,

- IS_NULLABLE prend les valeurs "YES" et "NO" selon que la colonne peut prendre la valeur NULL ou non,
- DATA_TYPE qui donne le domaine de la colonne,
- COLUMN_TYPE qui donne le type de la colonne avec plus de précision,
- COLUMN_KEY qui indique si la colonne est une clé primaire ou non,
- EXTRA qui donne les actions qui porte sur la colonne,
- PRIVILEGES qui donne les privilèges sur la colonne.

Ses enregistrements sont de la forme :

TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	ORDINAL_POSITION	DATA_TYPE	COLUMN_TYPE
TP1	Cours	Libelle	1	varchar	varchar(30)
TP1	Cours	VolHoraire	2	int	int(11)
TP1	Cour	Coefficient	3	int	int(11)
Mysql	User	Host	1	char	char(60)
Mysql	User	User	2	char	char(16)
Mysql	User	Password	3	char	char(41)

Pour obtenir toutes les colonnes d'une table dans une base de données, on exécute la requête :

Select COLUMN_NAME From COLUMNS Where TABLE_SCHEMA = 'nom_base' and TABLE_NAME = 'nom_table' ;

d. USER_PRIVILEGES

- GRANTEE contient les différents utilisateurs avec leur hot
- PRIVILEGE_TYPE contient tous les privilèges qui existent
- IS_GRANTABLE a pour valeur "YES" ou "NO" pour signifier que l'utilisateur a oui ou non le privilège

Ses enregistrements sont de la forme :

GRANTEE	PRIVILEGE_TYPE	IS_GRANTABLE
root@localhost	Select	YES
login@localhost	Delete	NO

e. TABLE_PRIVILEGES

- GRANTEE contient l'utilisateur à qui on a donné le privilège ;
- TABLE_SCHEMA contient la base de données dans laquelle se trouve la table ;

- TABLE_NAME contient la table sur laquelle porte le privilège ;
- PRIVILEGE_TYPE il s'agit du privilège attribué.

Ses enregistrements sont de la forme :

GRANTEE	TABLE_SCHEMA	TABLE_NAME	PRIVILEGE_TYPE
login1@localhost	TP1	Cours	Select
login1@localhost	TP1	Etudiant	Update
login2@localhost	TP1	Etudiant	Select
login2@localhost	TP2	Salle	Delete

f. TABLE_CONSTRAINTS

- CONSTRAINT_NAME donne le nom de la contrainte si elle en a,
- TABLE_SCHEMA donne le nom de la base contenant la table dans laquelle se trouve la contrainte,
- TABLE_NAME donne le nom de la table dans laquelle se trouve la contrainte,
- CONSTRAINT_TYPE donne le type contrainte

Ses enregistrements sont de la forme :

CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE
PRIMARY	TP1	Cours	PRIMARY KEY
PRIMARY	TP1	Etudiant	PRIMARY KEY
PRIMARY	TP1	Suivre	PRIMARY KEY
FK_Suivre_Cours	TP1	Suivre	FOREIGN KEY
FK_Suivre_Etudiant	TP1	Suivre	FOREIGN KEY

g. VIEWS

- TABLE_SCHEMA contient le nom de la base dans laquelle est créée la vue
- TABLE_NAME contient le nom de la vue
- CHECK_OPTION
- IS_UPDATABLE permet de connaitre si le contenu de la vue est modifiable
- DEFINER donne l'utilisateur qui a créé la vue

Ses enregistrements sont de la forme :

TABLE_SCHEMA	TABLE_NAME	CHECK_OPTION	IS_UPDTABLE	DEFINER
TP1	Cours_1	CASCADED	YES	root@localhost
TP1	Etu_Cours	NONE	YES	root@localhost
TP1	Etudiant_1	NONE	YES	root@localhost

h. REFERENTIAL_CONSTRAINTS

- CONSTRAINT_NAME contient le nom de la contraint
- CONSTRAINT_SCHEMA contient la base de données dans laquelle se trouve la contraint
- TABLE_NAME contient la table dans laquelle se trouve la contrainte,
- REFERENCED_TABLE_NAME contient la table dans laquelle se trouve la clé primaire dont elle fait référence.

Ses enregistrements dont de la forme :

CONSTRAINT_NAME	CONSTRAINT_SCHEMA	TABLE_NAME	REFERENCED_TABLE_NAME
FK_AvLieu_Cours	TP1	AvoirLieu	Cours
FK_AvLieu_Salle	TP1	AvoirLieu	Salle
FK_Suivre_Cours	TP1	Suivre	Cours
FK_Suivre_Etudiant	TP1	Suivre	Etudiant

i. TRIGGERS

- TRIGGER_SCHEMA contient la base de données dans laquelle le trigger est créé
- TRIGGER_NAME contient le nom du trigger
- EVENT_MANIPULATION contient l'événement du trigger (Insert ou Update ou Delete)
- ACTION_TIMING contient le moment de déclenchement du trigger (Before ou After)
- ACTION_STATEMENT contient le code du trigger

Ses enregistrements dont de la forme :

TRIGGER_SCHEMA	TRIGGER_NAME	EVENT_MANIPULATION	ACTION_TIMING
Etudiant	Creer_Matricule	Insert	Before
Cours	Creer_Vol_Horaire	Update	After

Remarque

Il existe d'autres tables qui peuvent être très importantes pour un administrateur. Parmi ces tables on a : ROUTINES, ENGINES.

III. 2. La base de données mysql

C'est une base de données native de MySQL qui permet d'administrer les bases de données des utilisateurs. Son rôle essentiel est la gestion de la sécurité des accès à ces bases. Seul l'utilisateur ***root*** doit avoir accès à cette base. Elle utilise essentiellement 3 tables pour gérer les droits d'accès des utilisateurs.

- ✓ La table ***user*** contient les informations de sécurité qui s'appliquent à tout le serveur ;
- ✓ La table ***Proc*** contient les fonctions et procédures créées dans le serveur ;
- ✓ La table ***InnoDB_Table_Stats*** contient les statistiques sur les tables ;
- ✓ La table ***InnoDB_Index_Stats*** contient les statistiques sur les index.