BASES DE DONNEES RELATIONNELLES AVEC MYSQL OU MARIADB

CODE COLLIDERS

CHRISTINE GUNSENHEIMER

ROMAIN VIRMAUX

Version 1.0 – 22/02/2020

Textes & schémas : Licence Creative Commons

Illustration couverture : Licence Envato Elements

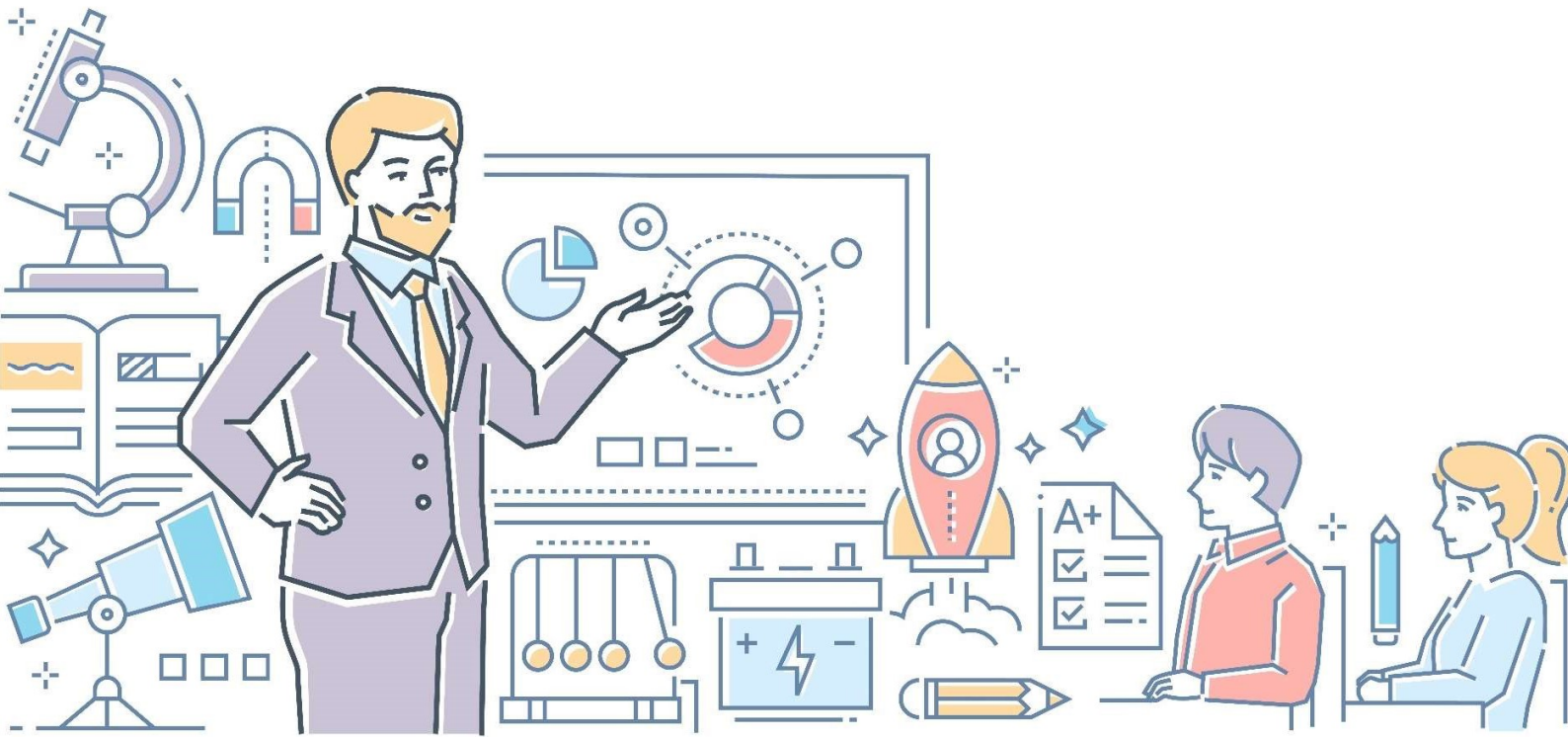


TABLE DES MATIERES

Licence Creative Commons

Sites utiles et intéressants :

https://sql.sh/

https://sqlbolt.com/ (en)

https://fr.wikibooks.org/wiki/MySQL

https://www.w3resource.com/mysql/mysql-tutorials.php (en)

LIEN AVEC LE REFERENTIEL

Activités type n°2 : Développer la partie backend d'une application web ou web mobile en intégrant les recommandations de sécurité.

Compétence n°5 : Créer une base de données (en annexe de ce document).

QU'EST CE QU'UNE BASE DE DONNEES ?

Une base de données est une collection organisée de données. Elle permet le stockage et la récupération de données brutes. Les données sont rangées de façon à pouvoir les retrouver facilement (et rapidement).

Il existe de nombreux types de base de données (hiérarchique, relationnel, objet, graph, ...), qui organisent et structurent les données selon différents modèles.

En savoir plus sur les types de bases de données :

https://www.digora.com/fr/blog/TOP-10-des-bases-de-donnees-partie-2

https://www.tutorialspoint.com/Types-of-databases

Exemples de cas d'utilisation d'une base de données :

- Enregistrement des informations en provenance d'un formulaire de contact sur un site Internet
- Enregistrement des informations d'un compte utilisateur

QU'EST-CE QU'UNE BASE DE DONNEE RELATIONELLE ?

Une base de données relationnelles est une base de données ou les données sont organisés dans des tableaux à deux dimensions.

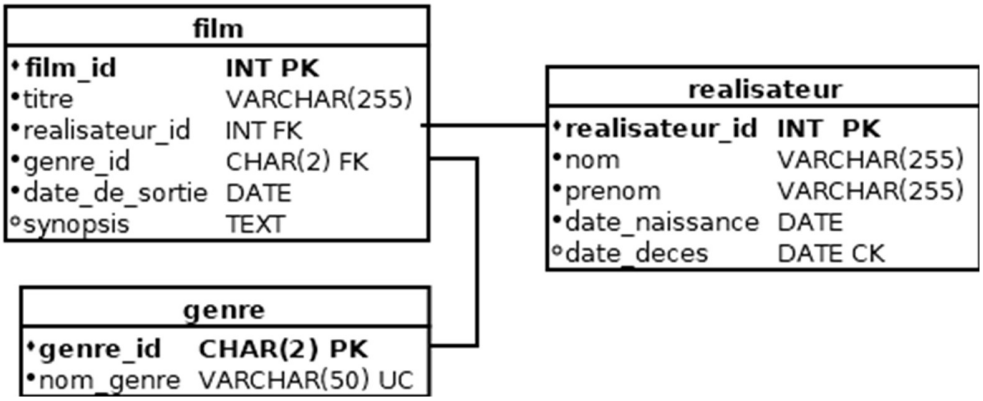
On appelle ces tableaux des tables. Une base de données contient plusieurs tables. Chaque table est constituée de colonnes.

Schéma physique de la table utilisateur

```
MariaDB [super_db]> SELECT * FROM utilisateur;
+----------------+---------+-----+-----------+-----------------------------+
| id_utilisateur | pseudo  | nom | prenom    | email                       |
+----------------+---------+-----+-----------+-----------------------------+
|              1 | niamor  | V.  | Romain    | romain@codecolliders.com    |
|              2 | noodle  | G.  | Christine | christine@codecolliders.com |
+----------------+---------+-----+-----------+-----------------------------+
2 rows in set (0.001 sec)
```

Affichage des données contenues dans la table « utilisateur »

Licence Creative Commons



Exemple de schéma physique de base de données relationnelle. Cette base est composée de 3 tables, et 2 relations y sont schématisées.

SGBD ET SGBDR

Le Système de Gestion de Base de Données (Relationnelle) est un logiciel qui permet aux utilisateurs de définir, créer, maintenir et contrôler l'accès à une base de données (relationnelle).

Le SGBD garanti la cohérence des données selon un modèle normalisé.

- Toutes les colonnes contiennent des valeurs simples (non multiples, non composées).

- Toutes les colonnes non clés dépendent de la clé primaire.

- Il n'y a pas de dépendance fonctionnelle entre deux colonnes non clé.

SQL - STRUCTURED QUERY LANGUAGE

Le SQL est un langage de requête structuré permettant de communiquer avec le SGBDR pour effectuer les actions suivantes :

- définition des données (DDL: data description language)

- requêtage des données (DQL: data query language)

- manipulation des données (DML: data manipulation language)

- contrôle des accès aux données (DCL: data control language)

Licence Creative Commons

Les bases du SQL

En SQL les commentaires peuvent s'écrire des façons suivantes :

```
-- Ceci est un commentaire sur une ligne (tout ce qui se trouve derrière est commenté)
# Ceci est également un commentaire sur une ligne (tout ce qui se trouve derrière est commenté)
/* Ceci est un commentaire sur une ou plusieurs lignes (tout ce qui se trouve à l'intérieur est commenté) */
```

Chaque requête SQL doit se terminer par ';'

Conventions de nommage (nom des bases, tables et colonnes) :

- Utiliser uniquement les caractères de l'alphabet anglais.
- Ne pas utiliser d'espaces et les remplacer par des _
- Eviter d'utiliser des nombres.
- Ne pas utiliser de caractères accentués.
- Utiliser de préférence des minuscules.
- Utiliser des noms compréhensibles et précis.

Installation de MariaDB en ligne de commande (dans le terminal)

Voici la commande permettant d'installer MariaDB sur Ubuntu 18.04 :

» apt install mariadb-server

Une fois l'installation effectuée il faut exécuter le script suivant et répondre aux questions afin de sécuriser l'installation :

» mysql_secure_installation

Pour se connecter au SGBD depuis un terminal, il suffit de taper la commande :

» mysql -u nom_utilisateur -p

DDL : Définition d'une base de données

Dans l'exemple ci-dessous, en orange et en majuscule sont ce qu'on appelle les « mot clés » du langage. Ils ont une signification, par exemple CREATE DATABASE veut dire « créer la base de données ». On écrit ensuite le nom de la base que l'on souhaite créer.

```
CREATE DATABASE nom_base_de_donnees;
```


Licence Creative Commons

Les noms des bases de données, des tables et des colonnes peuvent être encadrés du caractère ` (touche AltGr + 7). Il est important d'encadrer le nom des bases, tables et colonnes lorsque le nom peut être confondu avec un mot clé du langage par exemple si la base s'appelle « create ».

```
CREATE DATABASE `create`;
```

Il peut arriver que l'on créer une base de données qui porte le même nom qu'une base de données existante dans ce cas, MySQL retourne une erreur. Afin d'éviter cette erreur il est possible d'utiliser la requête suivante.

```
CREATE DATABASE IF NOT EXISTS nom_base_de_donnees;
```

Le mot clé USE permet de spécifier la base de données sur laquelle les requêtes suivantes vont s'exécuter.

```
USE nom_base_de_donnees;
```

SHOW DATABASES permet d'afficher la liste des bases des données existantes.

```
SHOW DATABASES;
```

DCL : Gestion des permissions

Création d'un utilisateur ayant accès au SGBD en local et définition de son mot de passe.

```
CREATE USER 'nom_utilisateur'@'localhost' IDENTIFIED BY
'mot_de_passe_sécurisé';
```

Définition du mot de passe pour un utilisateur.

```
SET PASSWORD FOR 'nom_utilisateur'@'localhost' =
PASSWORD('mot_de_passe_sécurisé');
```

Suppression d'un utilisateur.

```
DROP USER IF EXISTS 'nom_utilisateur'@'localhost';
```

Afficher les droits de tous les utilisateurs.

```
SHOW GRANTS;
```

Afficher les droits de l'utilisateur courant.

```
SHOW GRANTS FOR CURRENT_USER;
```

Donner tous les droits à un utilisateur sur une base de données (et spécifier son mot de passe).

Licence Creative Commons

```
GRANT ALL PRIVILEGES ON nom_base_de_donnees.* TO '
nom_utilisateur '@'localhost' IDENTIFIED BY
'mot_de_passe_sécurisé';
```

Donner les droits de sélection, insertion, modification et suppression à un utilisateur sur une base de données.

```
GRANT SELECT, INSERT, UPDATE, DELETE ON nom_base_de_donnees.* TO
'  nom_utilisateur '@'localhost';
```

Retire les droits à un utilisateur sur une base de données.

```
REVOKE ALL PRIVILEGES ON nom_base_de_donnees.* FROM '
nom_utilisateur '@'localhost';
```

Export de bases de données en ligne de commande

Pour exporter une base de données dans un fichier au format SQL, exécutez la commande suivante dans un terminal :

» mysqldump nom_base_de_donnees > fichier.sql

En fonction des droits, vous pouvez également spécifier le nom d'utilisateur à utiliser pour l'export.

» mysqldump –u user –p nom_base_de_donnees > fichier.sql

Execution d'un fichier SQL en ligne de commande

Afin d'exécuter un fichier au format SQL en ligne de commande saisissez l'une des deux commandes ci-dessous (selon les droits dont vous disposez) :

» mysql nom_base_de_donnees < fichier.sql

» mysql –u user –p nom_base_de_donnees < fichier.sql

Typage des données

1. Types numériques entiers

Type	Description	Min	Max	Mémoire occupée
BOOLEAN	associe la valeur 0 à false et la valeur 1 à true	0	1	1
TINYINT	nombre entier	-128	127	1
SMALLINT	nombre entier	-32768	32767	2
MEDIUMINT	nombre entier	-8388608	8388607	3
INT **- ou INTEGER**	nombre entier	-2147483648	2147483647	4
BIGINT	nombre entier	-9223372036854775808	9223372036854775807	8

Licence Creative Commons

2. Types numériques décimaux

Type	Description	Valeur par défaut	Exemples
DECIMAL(p,e) **- ou NUMERIC(p,e)**	nombre à virgule où 'p' indique la 'précision' ou nombre de chiffres en tout (max 65) et 'e' l'échelle ou le nombre de chiffres après la virgule (max 30)	DECIMAL est équivalent à DECIMAL(10,0)	DECIMAL(5,3) permet de stocker les nombres de –99,999 à 99,999.
FLOAT(o),	nombre à virgule stocké sur le nb d'octet 'o'	FLOAT est équivalent à FLOAT(4),	FLOAT(4) permet de stocker des nombres de -3.402823466E+38 à -1.175494351E-38 et 0 et de 1.175494351E-38 à 3.402823466E+38
DOUBLE, **REAL**	nombre à virgule stocké sur 8 octets en MySQL	DOUBLE et REAL sont équivalents à FLOAT(8) en MySQL	FLOAT(8) permet de stocker des nombres de -1.7976931348623157E+308 à -2.2250738585072014E-308 et 0 et de 2.2250738585072014E-308 à 1.7976931348623157E+308

3. Types temporels

Nom	Description	Format	min	max
DATE	date	YYYY-MM-DD	'1000-01-01'	'9999-12-31'
TIME	heure	HH:MM:SS	'-838:59:59.99999'	'838:59:59.99999'
DATETIME	date et heure au format string	YYYY-MM-DD HH:MM:SS	'1000-01-01 00:00:00.00'	'9999-12-31 23:59:59.99'
YEAR	année	YYYY	1901	2155
TIMESTAMP	date et heure au format int (stocké sur 4octet)	YYYYMMDDHHMMSS	Attention, ce n'est pas un vrai timestamp (nb de secondes depuis le 1er janvier 1970, 0 h 0 m 0s)	

Licence Creative Commons

4. Types alphanumériques courts

Type	Description	Valeur par défaut	Mémoire occupée	Exemple
CHAR(n)	Chaîne de caractère courte (moins de 255 caractères) de 'n' caractères	CHAR est équivalent à CHAR(1)	'n' octets	CHAR(5) sert fréquemment à stocker les codes postaux
VARCHAR(n)	Chaîne de caractère courte (moins de 255 caractères) pouvant varier entre 0 et 'n' caratères		Nombre d'octets permettant de stocker le texte (max 'n' octets)	VARCHAR(255) sert fréquemment à stocker un nom

5. Types alphanumériques longs

Type	Description	Longueur max	Mémoire occupée
TINYTEXT	Chaîne de caractères	255 caractères	Longueur de la chaîne + 1octet (max 2^8 octets)
TEXT	Chaîne de caractères	65 535 caractères	Longueur de la chaîne + 2octets (max 2^{16} octets)
MEDIUMTEXT	Chaîne de caractères	16 777 215 caractères	Longueur de la chaîne + 3octets (max 2^{24} octets)
LONGTEXT	Chaîne de caractères	4 294 967 295 caractères	Longueur de la chaîne + 4octets (max 2^{32} octets)

6. Types alphanumériques stockés sous forme binaire

Type	Description	Mémoire occupée
BINARY(n)	Chaîne de caractère courte sous forme binaire de longueur 'n' bits. Equivalent à CHAR(n)	'n' octets
VARBINARY(n)	Chaîne de caractère courte sous forme binaire pouvant varier entre 0 et 'n' bits. Equivalent à VARCHAR(n)	Nombre d'octets permettant de stocker l'information (max 'n' octets)
TINYBLOB	Chaîne de caractère sous forme binaire de longueur max 255 bits. Equivalent à TINYTEXT	Longueur de la chaîne + 1octet (max 2^8 octets)
BLOB	Chaîne de caractère sous forme binaire de longueur max 65 535 bits. Equivalent à TEXT	Longueur de la chaîne + 2octets (max 2^{16} octets)
MEDIUMBLOB	Chaîne de caractère sous forme binaire de longueur max 16 777 215 bits. Equivalent à MEDIUMTEXT	Longueur de la chaîne + 3octets (max 2^{24} octets)
LONGBLOB	Chaîne de caractère sous forme binaire de longueur max 4 294 967 295 bits. Equivalent à LONGTEXT	Longueur de la chaîne + 4octets (max 2^{32} octets)

Licence Creative Commons

7. Types alphanumériques particuliers

Type	Description	Longueur max	Valeurs non autorisées	Exemple
ENUM	Chaîne de caractères où des valeurs autorisées sont définies	65 535 valeurs possibles	MySQL stockera une chaîne vide '' dans le champ	ENUM('monsieur', 'madame', 'mademoiselle') peut servir à stocker la civilité
SET	Liste de chaînes de caractères où des valeurs autorisées sont définies	64 valeurs possible	MySQL stockera un 0 pour chaque valeur possible dans le champ	

DDL : Définition d'une table

Requête de création d'une table et de définition de l'ensemble de ses colonnes.

```
CREATE TABLE nom_table (
   nom_colonne_1        VARCHAR(255)       NULL,
   nom_colonne_2        INT                NOT NULL
);
```

Requête permettant la suppression d'une table.

```
DROP TABLE nom_table;
```

Requête permettant de supprimer tous les enregistrements d'une table.

```
TRUNCATE TABLE nom_table;
```

Requêtes retournant la définition des colonnes d'une table.

```
SHOW COLUMNS FROM nom_table;
```

```
DESCRIBE nom_table;
```

Requête permettant d'ajouter une colonne à une table.

```
ALTER TABLE nom_table ADD COLUMN nom_colonne VARCHAR(255);
```

Requête permettant la suppression d'une colonne.

```
ALTER TABLE nom_table DROP COLUMN nom_colonne;
```

Requête permettant de modifier une colonne.

```
ALTER TABLE nom_table MODIFY COLUMN nom_colonne VARCHAR(255);
```


Licence Creative Commons

Contraintes

1. NOT NULL

La contrainte NOT NULL empêche une colonne d'accepter une valeur nulle. Si on tente d'insérer un enregistrement dont le champ a pour valeur nul, MySQL n'insérera pas l'enregistrement, génèrera une erreur et stoppera l'exécution du script.

Requête d'ajout de la contrainte NOT NULL sur une colonne :

```
ALTER TABLE nom_table MODIFY nom_colonne VARCHAR(255) NOT NULL;
```

Requête de suppression de la contrainte NOT NULL :

```
ALTER TABLE nom_table MODIFY nom_colonne VARCHAR(255) NULL;
```

Par défaut, les colonnes acceptent des valeurs nulles

2. UNIQUE

La contrainte UNIQUE garantit que chaque valeur stockée dans un champs de la ou des colonne(s) possède une valeur différente. Si on tente d'insérer un enregistrement dont le champs a pour valeur une valeur déjà présente dans un autre champs de la colonne, MySQL n'insérera pas l'enregistrement, génèrera une erreur et stoppera l'exécution du script.

Requêtes d'ajout d'une contrainte UNIQUE sur une ou plusieurs colonnes :

```
ALTER TABLE nom_table ADD CONSTRAINT UC_nom_contrainte UNIQUE
(nom_colonne);

ALTER TABLE nom_table ADD CONSTRAINT UC_nom_contrainte UNIQUE
(nom_colonne_1, nom_colonne_2, nom_colonne_3);
```

Requête de suppression d'une contrainte UNIQUE :

```
ALTER TABLE nom_table DROP INDEX UC_nom_contrainte;
```

Une table peut contenir plusieurs colonnes avec des contraintes d'unicité. Les contraintes d'unicité peuvent également être appelée 'clés secondaires'.

3. PRIMARY KEY

La contrainte PRIMARY KEY identifie de façon unique chaque enregistrement d'une table. Elle contient des valeurs uniques et non nulles. **Une table ne possède qu'une clé primaire au maximum**. La clé primaire peut être constituée d'une colonne (clé primaire simple) ou de plusieurs colonnes (clé primaire composée).

Licence Creative Commons

Requêtes d'ajout d'une clé primaire sur une ou plusieurs colonnes :

```
ALTER TABLE nom_table ADD CONSTRAINT PK_nom_table PRIMARY KEY
(nom_colonne);

ALTER TABLE nom_table ADD CONSTRAINT PK_nom_table PRIMARY KEY
(nom_colonne_1, nom_colonne_2, nom_colonne_3);
```

Requête de suppression d'une clé primaire :

```
ALTER TABLE nom_table DROP PRIMARY KEY;
```

4. FOREIGN KEY

La contrainte FOREIGN KEY permet de référencer une ou plusieurs colonnes d'une table par rapport à une ou plusieurs colonnes d'une autre table. Elle identifie de façon unique un enregistrement (ligne) d'une autre table. Cette contrainte garantit qu'aucune action d'enregistrement, modification ou suppression ne détruise le lien entre deux tables.

Requêtes d'ajout d'une clé étrangère sur une ou plusieurs colonnes :

```
ALTER TABLE nom_table_1
    ADD CONSTRAINT FK_nom_table_1_nom_table_2 FOREIGN KEY
(nom_colonne_1) REFERENCES nom_table_2 (nom_colonne_2);

ALTER TABLE nom_table_1
    ADD CONSTRAINT FK_nom_table_1_nom_table_2
    FOREIGN KEY (nom_colonne_1, nom_colonne_2, ...)
    REFERENCES nom_table_2 (nom_colonne_3, nom_colonne_4, ...);
```

Requête de suppression d'une clé étrangère :

```
ALTER TABLE nom_table_1 DROP FOREIGN KEY
FK_nom_table_1_nom_table_2;
```

5. CHECK

La contrainte CHECK s'assure que les valeurs stockées dans la colonne répondent à une condition particulière. Elle permet donc de restreindre les valeurs stockées dans la colonne. Si on tente d'insérer un enregistrement dont la valeur du champ ne satisfait pas la condition exprimée dans la contrainte CHECK, MySQL n'insérera pas l'enregistrement, génèrera une erreur et stoppera l'exécution du script. Cette contrainte peut se référer à des valeurs contenues dans une ou plusieurs colonnes

Requêtes d'ajout d'une contrainte CHECK à une ou plusieurs colonnes :

Licence Creative Commons

```sql
ALTER TABLE nom_table ADD CONSTRAINT CK_nom_contrainte CHECK
(colonne_1 > 0);

ALTER TABLE nom_table ADD CONSTRAINT CK_nom_contrainte CHECK
(colonne_1 > 0 AND colonne_2 = "valeur");
```

Requête de suppression de la contrainte CHECK :

```sql
ALTER TABLE nom_table DROP CHECK CK_nom_contrainte;
```

6. DEFAULT

La contrainte DEFAULT définit une valeur par défaut pour une colonne. Cette valeur sera ajoutée au champ de l'enregistrement uniquement si aucune valeur n'est spécifiée.

Ajout de la contrainte DEFAULT sur une colonne :

```sql
ALTER TABLE nom_table ALTER nom_colonne SET DEFAULT "valeur_par
défaut";
```

Suppression de la contrainte DEFAULT d'une colonne :

```sql
ALTER TABLE nom_table ALTER nom_colonne DROP DEFAULT;
```

AUTO_INCREMENT

AUTO_INCREMENT n'est pas à proprement parler une contrainte. Il s'agit plutôt d'une propriété d'une colonne qui permet de générer la valeur qui sera stockée dans le champ de la colonne lors d'un enregistrement.

Il ne peut y avoir qu'une seule colonne possédant la propriété AUTO_INCREMENT par table La propriété AUTO_INCREMENT ne peut être mis en place que sur une colonne composant soit une clé primaire, soit une clé secondaire.

Exemple de définition de la propriété AUTO_INCREMENT lors de la création de la table :

La propriété AUTO_INCREMENT est mise en place lors de la création de la table.

```sql
CREATE TABLE nom_table (
    nom_colonne_1    VARCHAR(255)   NULL      AUTO_INCREMENT,
    nom_colonne_2    INT            NOT NULL
);
```

Exemple de requête d'ajout d'un AUTO_INCREMENT et clé primaire sur une colonne existante :

```sql
ALTER TABLE nom_table CHANGE nom_colonne nom_colonne INT
AUTO_INCREMENT PRIMARY KEY;
```

DDL : Définition des index

Un index permet d'accéder aux valeurs contenues dans les champs d'une colonne de manière plus efficace et rapide. Les index sont automatiquement créés pour les clés primaires et les clés secondaires mais doivent être manuellement mis en place pour les clés étrangères ou lorsqu'un grand nombre de requête est effectué sur une ou plusieurs colonnes.

Attention, chaque index prend de la place sur le disque et en mémoire.

Requête d'ajout d'un INDEX sur une colonne.

```
CREATE INDEX nom_index ON nom_table( nom_colonne ASC );
```

Requête d'ajout d'un INDEX sur plusieurs colonnes.

```
CREATE INDEX nom_index ON nom_table( nom_colonne_1,
nom_colonne_2, … DESC );
```

Requête d'ajout d'une contrainte UNIQUE sur une colonne.

```
CREATE UNIQUE INDEX nom_index ON nom_table (nom_colonne_1,
nom_colonne_2, … ASC);
```

Requête de suppression d'une contrainte INDEX sur une colonne.

```
DROP INDEX nom_index ON nom_table;
```

Requête permettant de reconstruire tous les index d'une table.

```
ALTER INDEX ALL ON nom_table REBUILD;
```


Licence Creative Commons

BASES DE DONNEES RELATIONNELLES AVEC MYSQL OU MARIADB

CODE COLLIDERS

CHRISTINE GUNSENHEIMER

ROMAIN VIRMAUX

Version 1.0 – 22/02/2020

Textes & schémas : Licence Creative Commons

Illustration couverture : Licence Envato Elements

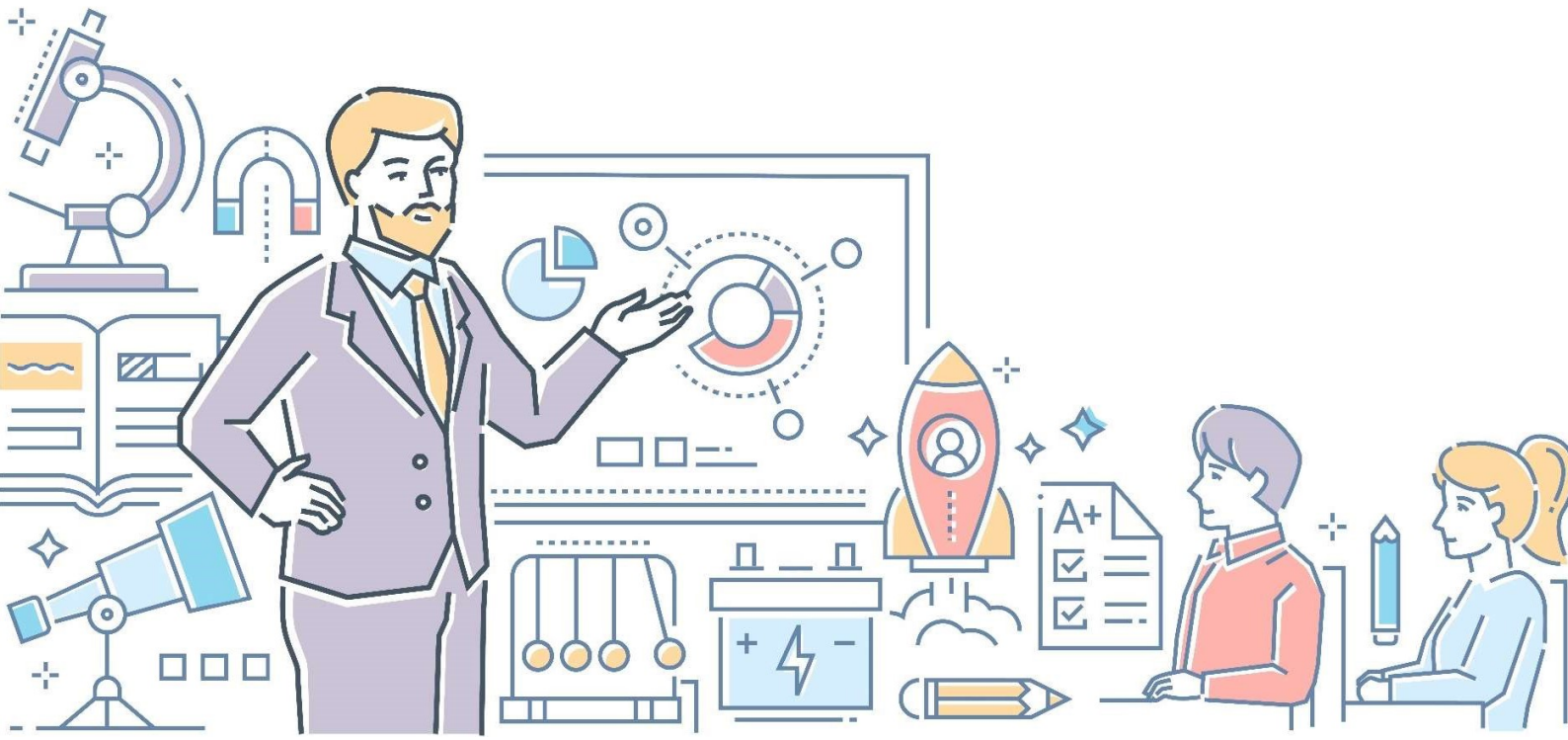


TABLE DES MATIERES

Licence Creative Commons

Sites utiles et intéressants :

https://sql.sh/

https://sqlbolt.com/ (en)

https://fr.wikibooks.org/wiki/MySQL

https://www.w3resource.com/mysql/mysql-tutorials.php (en)

LIEN AVEC LE REFERENTIEL

Activités type n°2 : Développer la partie backend d'une application web ou web mobile en intégrant les recommandations de sécurité.

Compétence n°5 : Créer une base de données (en annexe de ce document).

QU'EST CE QU'UNE BASE DE DONNEES ?

Une base de données est une collection organisée de données. Elle permet le stockage et la récupération de données brutes. Les données sont rangées de façon à pouvoir les retrouver facilement (et rapidement).

Il existe de nombreux types de base de données (hiérarchique, relationnel, objet, graph, ...), qui organisent et structurent les données selon différents modèles.

En savoir plus sur les types de bases de données :

https://www.digora.com/fr/blog/TOP-10-des-bases-de-donnees-partie-2

https://www.tutorialspoint.com/Types-of-databases

Exemples de cas d'utilisation d'une base de données :

- Enregistrement des informations en provenance d'un formulaire de contact sur un site Internet
- Enregistrement des informations d'un compte utilisateur

QU'EST-CE QU'UNE BASE DE DONNEE RELATIONELLE ?

Une base de données relationnelles est une base de données ou les données sont organisés dans des tableaux à deux dimensions.

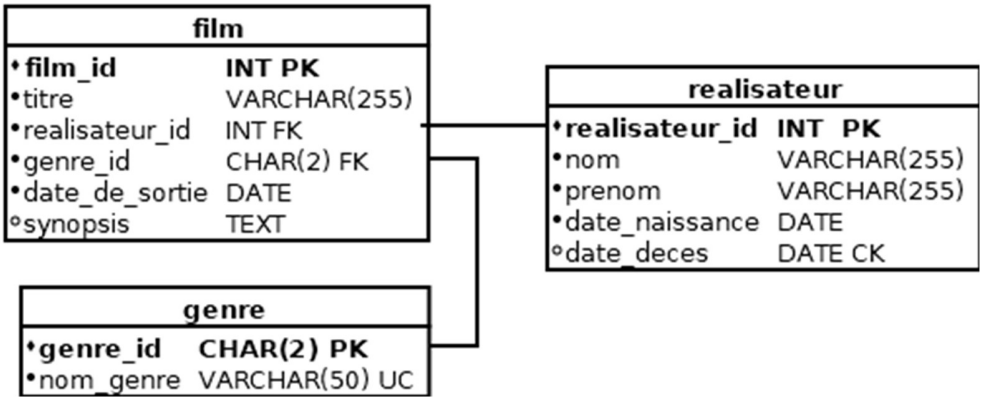
On appelle ces tableaux des tables. Une base de données contient plusieurs tables. Chaque table est constituée de colonnes.

Schéma physique de la table utilisateur

```
MariaDB [super_db]> SELECT * FROM utilisateur;
+----------------+---------+-----+-----------+-----------------------------+
| id_utilisateur | pseudo  | nom | prenom    | email                       |
+----------------+---------+-----+-----------+-----------------------------+
|              1 | niamor  | V.  | Romain    | romain@codecolliders.com    |
|              2 | noodle  | G.  | Christine | christine@codecolliders.com |
+----------------+---------+-----+-----------+-----------------------------+
2 rows in set (0.001 sec)
```

Affichage des données contenues dans la table « utilisateur »

Licence Creative Commons



Exemple de schéma physique de base de données relationnelle. Cette base est composée de 3 tables, et 2 relations y sont schématisées.

SGBD ET SGBDR

Le Système de Gestion de Base de Données (Relationnelle) est un logiciel qui permet aux utilisateurs de définir, créer, maintenir et contrôler l'accès à une base de données (relationnelle).

Le SGBD garanti la cohérence des données selon un modèle normalisé.

- Toutes les colonnes contiennent des valeurs simples (non multiples, non composées).

- Toutes les colonnes non clés dépendent de la clé primaire.

- Il n'y a pas de dépendance fonctionnelle entre deux colonnes non clé.

SQL - STRUCTURED QUERY LANGUAGE

Le SQL est un langage de requête structuré permettant de communiquer avec le SGBDR pour effectuer les actions suivantes :

- définition des données (DDL: data description language)

- requêtage des données (DQL: data query language)

- manipulation des données (DML: data manipulation language)

- contrôle des accès aux données (DCL: data control language)

Licence Creative Commons

Les bases du SQL

En SQL les commentaires peuvent s'écrire des façons suivantes :

```
-- Ceci est un commentaire sur une ligne (tout ce qui se trouve derrière est commenté)
# Ceci est également un commentaire sur une ligne (tout ce qui se trouve derrière est commenté)
/* Ceci est un commentaire sur une ou plusieurs lignes (tout ce qui se trouve à l'intérieur est commenté) */
```

Chaque requête SQL doit se terminer par ';'

Conventions de nommage (nom des bases, tables et colonnes) :

- Utiliser uniquement les caractères de l'alphabet anglais.
- Ne pas utiliser d'espaces et les remplacer par des _
- Eviter d'utiliser des nombres.
- Ne pas utiliser de caractères accentués.
- Utiliser de préférence des minuscules.
- Utiliser des noms compréhensibles et précis.

Installation de MariaDB en ligne de commande (dans le terminal)

Voici la commande permettant d'installer MariaDB sur Ubuntu 18.04 :

» apt install mariadb-server

Une fois l'installation effectuée il faut exécuter le script suivant et répondre aux questions afin de sécuriser l'installation :

» mysql_secure_installation

Pour se connecter au SGBD depuis un terminal, il suffit de taper la commande :

» mysql -u nom_utilisateur -p

DDL : Définition d'une base de données

Dans l'exemple ci-dessous, en orange et en majuscule sont ce qu'on appelle les « mot clés » du langage. Ils ont une signification, par exemple CREATE DATABASE veut dire « créer la base de données ». On écrit ensuite le nom de la base que l'on souhaite créer.

```
CREATE DATABASE nom_base_de_donnees;
```


Licence Creative Commons

Les noms des bases de données, des tables et des colonnes peuvent être encadrés du caractère ` (touche AltGr + 7). Il est important d'encadrer le nom des bases, tables et colonnes lorsque le nom peut être confondu avec un mot clé du langage par exemple si la base s'appelle « create ».

```
CREATE DATABASE `create`;
```

Il peut arriver que l'on créer une base de données qui porte le même nom qu'une base de données existante dans ce cas, MySQL retourne une erreur. Afin d'éviter cette erreur il est possible d'utiliser la requête suivante.

```
CREATE DATABASE IF NOT EXISTS nom_base_de_donnees;
```

Le mot clé USE permet de spécifier la base de données sur laquelle les requêtes suivantes vont s'exécuter.

```
USE nom_base_de_donnees;
```

SHOW DATABASES permet d'afficher la liste des bases des données existantes.

```
SHOW DATABASES;
```

DCL : Gestion des permissions

Création d'un utilisateur ayant accès au SGBD en local et définition de son mot de passe.

```
CREATE USER 'nom_utilisateur'@'localhost' IDENTIFIED BY
'mot_de_passe_sécurisé';
```

Définition du mot de passe pour un utilisateur.

```
SET PASSWORD FOR 'nom_utilisateur'@'localhost' =
PASSWORD('mot_de_passe_sécurisé');
```

Suppression d'un utilisateur.

```
DROP USER IF EXISTS 'nom_utilisateur'@'localhost';
```

Afficher les droits de tous les utilisateurs.

```
SHOW GRANTS;
```

Afficher les droits de l'utilisateur courant.

```
SHOW GRANTS FOR CURRENT_USER;
```

Donner tous les droits à un utilisateur sur une base de données (et spécifier son mot de passe).

Licence Creative Commons

```
GRANT ALL PRIVILEGES ON nom_base_de_donnees.* TO '
nom_utilisateur '@'localhost' IDENTIFIED BY
'mot_de_passe_sécurisé';
```

Donner les droits de sélection, insertion, modification et suppression à un utilisateur sur une base de données.

```
GRANT SELECT, INSERT, UPDATE, DELETE ON nom_base_de_donnees.* TO
'  nom_utilisateur '@'localhost';
```

Retire les droits à un utilisateur sur une base de données.

```
REVOKE ALL PRIVILEGES ON nom_base_de_donnees.* FROM '
nom_utilisateur '@'localhost';
```

Export de bases de données en ligne de commande

Pour exporter une base de données dans un fichier au format SQL, exécutez la commande suivante dans un terminal :

» mysqldump nom_base_de_donnees > fichier.sql

En fonction des droits, vous pouvez également spécifier le nom d'utilisateur à utiliser pour l'export.

» mysqldump –u user –p nom_base_de_donnees > fichier.sql

Execution d'un fichier SQL en ligne de commande

Afin d'exécuter un fichier au format SQL en ligne de commande saisissez l'une des deux commandes ci-dessous (selon les droits dont vous disposez) :

» mysql nom_base_de_donnees < fichier.sql

» mysql –u user –p nom_base_de_donnees < fichier.sql

Typage des données

1. Types numériques entiers

Type	Description	Min	Max	Mémoire occupée
BOOLEAN	associe la valeur 0 à false et la valeur 1 à true	0	1	1
TINYINT	nombre entier	-128	127	1
SMALLINT	nombre entier	-32768	32767	2
MEDIUMINT	nombre entier	-8388608	8388607	3
INT **- ou INTEGER**	nombre entier	-2147483648	2147483647	4
BIGINT	nombre entier	-9223372036854775808	9223372036854775807	8

Licence Creative Commons

2. Types numériques décimaux

Type	Description	Valeur par défaut	Exemples
DECIMAL(p,e) **- ou NUMERIC(p,e)**	nombre à virgule où 'p' indique la 'précision' ou nombre de chiffres en tout (max 65) et 'e' l'échelle ou le nombre de chiffres après la virgule (max 30)	DECIMAL est équivalent à DECIMAL(10,0)	DECIMAL(5,3) permet de stocker les nombres de –99,999 à 99,999.
FLOAT(o),	nombre à virgule stocké sur le nb d'octet 'o'	FLOAT est équivalent à FLOAT(4),	FLOAT(4) permet de stocker des nombres de -3.402823466E+38 à -1.175494351E-38 et 0 et de 1.175494351E-38 à 3.402823466E+38
DOUBLE, **REAL**	nombre à virgule stocké sur 8 octets en MySQL	DOUBLE et REAL sont équivalents à FLOAT(8) en MySQL	FLOAT(8) permet de stocker des nombres de -1.7976931348623157E+308 à -2.2250738585072014E-308 et 0 et de 2.2250738585072014E-308 à 1.7976931348623157E+308

3. Types temporels

Nom	Description	Format	min	max
DATE	date	YYYY-MM-DD	'1000-01-01'	'9999-12-31'
TIME	heure	HH:MM:SS	'-838:59:59.99999'	'838:59:59.99999'
DATETIME	date et heure au format string	YYYY-MM-DD HH:MM:SS	'1000-01-01 00:00:00.00'	'9999-12-31 23:59:59.99'
YEAR	année	YYYY	1901	2155
TIMESTAMP	date et heure au format int (stocké sur 4octet)	YYYYMMDDHHMMSS	Attention, ce n'est pas un vrai timestamp (nb de secondes depuis le 1er janvier 1970, 0 h 0 m 0s)	

Licence Creative Commons

4. Types alphanumériques courts

Type	Description	Valeur par défaut	Mémoire occupée	Exemple
CHAR(n)	Chaîne de caractère courte (moins de 255 caractères) de 'n' caractères	CHAR est équivalent à CHAR(1)	'n' octets	CHAR(5) sert fréquemment à stocker les codes postaux
VARCHAR(n)	Chaîne de caractère courte (moins de 255 caractères) pouvant varier entre 0 et 'n' caratères		Nombre d'octets permettant de stocker le texte (max 'n' octets)	VARCHAR(255) sert fréquemment à stocker un nom

5. Types alphanumériques longs

Type	Description	Longueur max	Mémoire occupée
TINYTEXT	Chaîne de caractères	255 caractères	Longueur de la chaîne + 1octet (max 2^8 octets)
TEXT	Chaîne de caractères	65 535 caractères	Longueur de la chaîne + 2octets (max 2^{16} octets)
MEDIUMTEXT	Chaîne de caractères	16 777 215 caractères	Longueur de la chaîne + 3octets (max 2^{24} octets)
LONGTEXT	Chaîne de caractères	4 294 967 295 caractères	Longueur de la chaîne + 4octets (max 2^{32} octets)

6. Types alphanumériques stockés sous forme binaire

Type	Description	Mémoire occupée
BINARY(n)	Chaîne de caractère courte sous forme binaire de longueur 'n' bits. Equivalent à CHAR(n)	'n' octets
VARBINARY(n)	Chaîne de caractère courte sous forme binaire pouvant varier entre 0 et 'n' bits. Equivalent à VARCHAR(n)	Nombre d'octets permettant de stocker l'information (max 'n' octets)
TINYBLOB	Chaîne de caractère sous forme binaire de longueur max 255 bits. Equivalent à TINYTEXT	Longueur de la chaîne + 1octet (max 2^8 octets)
BLOB	Chaîne de caractère sous forme binaire de longueur max 65 535 bits. Equivalent à TEXT	Longueur de la chaîne + 2octets (max 2^{16} octets)
MEDIUMBLOB	Chaîne de caractère sous forme binaire de longueur max 16 777 215 bits. Equivalent à MEDIUMTEXT	Longueur de la chaîne + 3octets (max 2^{24} octets)
LONGBLOB	Chaîne de caractère sous forme binaire de longueur max 4 294 967 295 bits. Equivalent à LONGTEXT	Longueur de la chaîne + 4octets (max 2^{32} octets)

Licence Creative Commons

7. Types alphanumériques particuliers

Type	Description	Longueur max	Valeurs non autorisées	Exemple
ENUM	Chaîne de caractères où des valeurs autorisées sont définies	65 535 valeurs possibles	MySQL stockera une chaîne vide '' dans le champ	ENUM('monsieur', 'madame', 'mademoiselle') peut servir à stocker la civilité
SET	Liste de chaînes de caractères où des valeurs autorisées sont définies	64 valeurs possible	MySQL stockera un 0 pour chaque valeur possible dans le champ	

DDL : Définition d'une table

Requête de création d'une table et de définition de l'ensemble de ses colonnes.

```
CREATE TABLE nom_table (
   nom_colonne_1        VARCHAR(255)       NULL,
   nom_colonne_2        INT                NOT NULL
);
```

Requête permettant la suppression d'une table.

```
DROP TABLE nom_table;
```

Requête permettant de supprimer tous les enregistrements d'une table.

```
TRUNCATE TABLE nom_table;
```

Requêtes retournant la définition des colonnes d'une table.

```
SHOW COLUMNS FROM nom_table;
```

```
DESCRIBE nom_table;
```

Requête permettant d'ajouter une colonne à une table.

```
ALTER TABLE nom_table ADD COLUMN nom_colonne VARCHAR(255);
```

Requête permettant la suppression d'une colonne.

```
ALTER TABLE nom_table DROP COLUMN nom_colonne;
```

Requête permettant de modifier une colonne.

```
ALTER TABLE nom_table MODIFY COLUMN nom_colonne VARCHAR(255);
```


Licence Creative Commons

Contraintes

1. NOT NULL

La contrainte NOT NULL empêche une colonne d'accepter une valeur nulle. Si on tente d'insérer un enregistrement dont le champ a pour valeur nul, MySQL n'insérera pas l'enregistrement, génèrera une erreur et stoppera l'exécution du script.

Requête d'ajout de la contrainte NOT NULL sur une colonne :

```
ALTER TABLE nom_table MODIFY nom_colonne VARCHAR(255) NOT NULL;
```

Requête de suppression de la contrainte NOT NULL :

```
ALTER TABLE nom_table MODIFY nom_colonne VARCHAR(255) NULL;
```

Par défaut, les colonnes acceptent des valeurs nulles

2. UNIQUE

La contrainte UNIQUE garantit que chaque valeur stockée dans un champs de la ou des colonne(s) possède une valeur différente. Si on tente d'insérer un enregistrement dont le champs a pour valeur une valeur déjà présente dans un autre champs de la colonne, MySQL n'insérera pas l'enregistrement, génèrera une erreur et stoppera l'exécution du script.

Requêtes d'ajout d'une contrainte UNIQUE sur une ou plusieurs colonnes :

```
ALTER TABLE nom_table ADD CONSTRAINT UC_nom_contrainte UNIQUE
(nom_colonne);

ALTER TABLE nom_table ADD CONSTRAINT UC_nom_contrainte UNIQUE
(nom_colonne_1, nom_colonne_2, nom_colonne_3);
```

Requête de suppression d'une contrainte UNIQUE :

```
ALTER TABLE nom_table DROP INDEX UC_nom_contrainte;
```

Une table peut contenir plusieurs colonnes avec des contraintes d'unicité. Les contraintes d'unicité peuvent également être appelée 'clés secondaires'.

3. PRIMARY KEY

La contrainte PRIMARY KEY identifie de façon unique chaque enregistrement d'une table. Elle contient des valeurs uniques et non nulles. **Une table ne possède qu'une clé primaire au maximum**. La clé primaire peut être constituée d'une colonne (clé primaire simple) ou de plusieurs colonnes (clé primaire composée).

Licence Creative Commons

Requêtes d'ajout d'une clé primaire sur une ou plusieurs colonnes :

```
ALTER TABLE nom_table ADD CONSTRAINT PK_nom_table PRIMARY KEY
(nom_colonne);

ALTER TABLE nom_table ADD CONSTRAINT PK_nom_table PRIMARY KEY
(nom_colonne_1, nom_colonne_2, nom_colonne_3);
```

Requête de suppression d'une clé primaire :

```
ALTER TABLE nom_table DROP PRIMARY KEY;
```

4. FOREIGN KEY

La contrainte FOREIGN KEY permet de référencer une ou plusieurs colonnes d'une table par rapport à une ou plusieurs colonnes d'une autre table. Elle identifie de façon unique un enregistrement (ligne) d'une autre table. Cette contrainte garantit qu'aucune action d'enregistrement, modification ou suppression ne détruise le lien entre deux tables.

Requêtes d'ajout d'une clé étrangère sur une ou plusieurs colonnes :

```
ALTER TABLE nom_table_1
    ADD CONSTRAINT FK_nom_table_1_nom_table_2 FOREIGN KEY
(nom_colonne_1) REFERENCES nom_table_2 (nom_colonne_2);

ALTER TABLE nom_table_1
    ADD CONSTRAINT FK_nom_table_1_nom_table_2
    FOREIGN KEY (nom_colonne_1, nom_colonne_2, ...)
    REFERENCES nom_table_2 (nom_colonne_3, nom_colonne_4, ...);
```

Requête de suppression d'une clé étrangère :

```
ALTER TABLE nom_table_1 DROP FOREIGN KEY
FK_nom_table_1_nom_table_2;
```

5. CHECK

La contrainte CHECK s'assure que les valeurs stockées dans la colonne répondent à une condition particulière. Elle permet donc de restreindre les valeurs stockées dans la colonne. Si on tente d'insérer un enregistrement dont la valeur du champ ne satisfait pas la condition exprimée dans la contrainte CHECK, MySQL n'insérera pas l'enregistrement, génèrera une erreur et stoppera l'exécution du script. Cette contrainte peut se référer à des valeurs contenues dans une ou plusieurs colonnes

Requêtes d'ajout d'une contrainte CHECK à une ou plusieurs colonnes :

Licence Creative Commons

```sql
ALTER TABLE nom_table ADD CONSTRAINT CK_nom_contrainte CHECK
(colonne_1 > 0);

ALTER TABLE nom_table ADD CONSTRAINT CK_nom_contrainte CHECK
(colonne_1 > 0 AND colonne_2 = "valeur");
```

Requête de suppression de la contrainte CHECK :

```sql
ALTER TABLE nom_table DROP CHECK CK_nom_contrainte;
```

6. DEFAULT

La contrainte DEFAULT définit une valeur par défaut pour une colonne. Cette valeur sera ajoutée au champ de l'enregistrement uniquement si aucune valeur n'est spécifiée.

Ajout de la contrainte DEFAULT sur une colonne :

```sql
ALTER TABLE nom_table ALTER nom_colonne SET DEFAULT "valeur_par
défaut";
```

Suppression de la contrainte DEFAULT d'une colonne :

```sql
ALTER TABLE nom_table ALTER nom_colonne DROP DEFAULT;
```

AUTO_INCREMENT

AUTO_INCREMENT n'est pas à proprement parler une contrainte. Il s'agit plutôt d'une propriété d'une colonne qui permet de générer la valeur qui sera stockée dans le champ de la colonne lors d'un enregistrement.

Il ne peut y avoir qu'une seule colonne possédant la propriété AUTO_INCREMENT par table La propriété AUTO_INCREMENT ne peut être mis en place que sur une colonne composant soit une clé primaire, soit une clé secondaire.

Exemple de définition de la propriété AUTO_INCREMENT lors de la création de la table :

La propriété AUTO_INCREMENT est mise en place lors de la création de la table.

```sql
CREATE TABLE nom_table (
    nom_colonne_1    VARCHAR(255)   NULL      AUTO_INCREMENT,
    nom_colonne_2    INT            NOT NULL
);
```

Exemple de requête d'ajout d'un AUTO_INCREMENT et clé primaire sur une colonne existante :

```sql
ALTER TABLE nom_table CHANGE nom_colonne nom_colonne INT
AUTO_INCREMENT PRIMARY KEY;
```

DDL : Définition des index

Un index permet d'accéder aux valeurs contenues dans les champs d'une colonne de manière plus efficace et rapide. Les index sont automatiquement créés pour les clés primaires et les clés secondaires mais doivent être manuellement mis en place pour les clés étrangères ou lorsqu'un grand nombre de requête est effectué sur une ou plusieurs colonnes.

Attention, chaque index prend de la place sur le disque et en mémoire.

Requête d'ajout d'un INDEX sur une colonne.

```
CREATE INDEX nom_index ON nom_table( nom_colonne ASC );
```

Requête d'ajout d'un INDEX sur plusieurs colonnes.

```
CREATE INDEX nom_index ON nom_table( nom_colonne_1,
nom_colonne_2, … DESC );
```

Requête d'ajout d'une contrainte UNIQUE sur une colonne.

```
CREATE UNIQUE INDEX nom_index ON nom_table (nom_colonne_1,
nom_colonne_2, … ASC);
```

Requête de suppression d'une contrainte INDEX sur une colonne.

```
DROP INDEX nom_index ON nom_table;
```

Requête permettant de reconstruire tous les index d'une table.

```
ALTER INDEX ALL ON nom_table REBUILD;
```


Licence Creative Commons