Les contraintes d'intégrité avec MySQL

[sommaire]

1. **Objectifs**

- Connaitre les contraintes d'intégrité avec MySQL

- **Présentation**

  - Une contrainte d'intégrité est une règle qui définit la cohérence d'une donnée ou d'un ensemble de données de la BD.
  - Il existe deux types de contraintes :

  1. sur une colonne unique,
  2. ou sur une table lorsque la contrainte porte sur une ou plusieurs colonnes.

- Les contraintes sont définies au moment de la création des tables.

- **Contraintes d'intégrité sur une colonne**

  - Les contraintes d'intégrité sur une colonne sont :

  1. **PRIMARY KEY : définit l'attribut comme la clé primaire**

  - A chaque élément d'une table est associé un champ 'id', non null, qui identifie de façon unique l'enregistrement et qui nous sert de référence pour établir des liens avec d'autres tables. C'est ce que l'on appelle une **clé primaire**.
  - Cette contrainte définit une colonne ou une combinaison de colonnes qui identifie de manière unique chaque ligne de la table.
  - On pourra à la création de la table déclarer un champ (qu'il soit auto-incrémenté ou pas) comme étant une clé primaire par l'utilisation du mot clé PRIMARY KEY juste après la déclaration du type du champ. Comme dans l'exemple suivant:

    ```
    CREATE TABLE matable (id INTEGER NOT NULL PRIMARY KEY, ...)
    ```

  - **Syntaxe pour définir une clé primaire au niveau de la colonne : `column name datatype [CONSTRAINT constraint_name] PRIMARY KEY`**

apcpedagogie.com

- **Exemple :**
  - Pour créer une table d'employés avec une contrainte de clé primaire, la requête serait comme.
  - **Clé primaire au niveau de la colonne :**

```
CREATE TABLE employee
( id number(5) PRIMARY KEY,
name char(20),
dept char(10),
age number(2),
salary number(10),
location char(10)
);
```

- ou alors

```
CREATE TABLE employee
( id number(5) CONSTRAINT emp_id_pk PRIMARY KEY,
name char(20),
dept char(10),
age number(2),
salary number(10),
location char(10)
);
```

- **Clé primaire au niveau de la colonne :**

```
CREATE TABLE employee
( id number(5),
name char(20),
dept char(10),
age number(2),
salary number(10),
location char(10),
CONSTRAINT emp_id_pk PRIMARY KEY (id)
);
```

- **UNIQUE : interdit que deux tuples de la relation aient la même valeur pour l'attribut.**
  - Cette contrainte garantit qu'une colonne ou un groupe de colonnes de chaque ligne a une valeur distincte. Une ou plusieurs colonnes peuvent avoir une valeur nulle mais les valeurs ne peuvent pas être dupliquées.

apcpedagogie.com

- **Syntaxe pour définir une clé unique au niveau de la : `column name datatype [CONSTRAINT constraint_name] PRIMARY KEY`**
- **Exemple :**
  - pour créer une table d'employés avec une clé unique, la requête serait comme,
  - **Clé unique au niveau de la colonne :**

```
CREATE TABLE employee
( id number(5) PRIMARY KEY,
name char(20),
dept char(10),
age number(2),
salary number(10),
location char(10) UNIQUE
);
```

- ou alors

```
CREATE TABLE employee
( id number(5) PRIMARY KEY,
name char(20),
dept char(10),
age number(2),
salary number(10),
location char(10) CONSTRAINT loc_un UNIQUE
);
```

- **REFERENCES () : contrôle l'intégrité référentielle entre l'attribut et la table et ses colonnes spécifiées**
  - Cette contrainte identifie toute colonne référençant la PRIMARY KEY dans une autre table. Il établit une relation entre deux colonnes d'une même table ou entre des tables différentes. Pour qu'une colonne soit définie en tant que clé étrangère, elle doit être définie en tant que clé primaire dans la table à laquelle elle fait référence. Une ou plusieurs colonnes peuvent être définies comme clé étrangère.
  - **Syntaxe pour définir une clé étrangère au niveau de la colonne: : `[CONSTRAINT constraint_name] REFERENCES Referenced_Table_name(column_name)`**
  - **Exemple :**
    - Utilisons la table "product" et "order_items".
    - **Clé étrangère au niveau de la colonne :**

apcpedagogie.com

```
CREATE TABLE product
( product_id number(5) CONSTRAINT pd_id_pk PRIMARY KEY,
product_name char(20),
supplier_name char(20),
unit_price number(10)
);
```

```
CREATE TABLE order_items
( order_id number(5) CONSTRAINT od_id_pk PRIMARY KEY,
product_id number(5) CONSTRAINT pd_id_fk REFERENCES, product(product_id),
product_name char(20),
supplier_name char(20),
unit_price number(10)
);
```

- **CHECK () : contrôle la validité de la valeur de l'attribut spécifié dans la condition dans le cadre d'une restriction de domaine**
    - Cette contrainte définit une règle métier sur une colonne. Toutes les lignes doivent satisfaire à cette règle. La contrainte peut être appliquée pour une seule colonne ou un groupe de colonnes.
    - **Syntaxe pour définir une contrainte de vérification : : `[CONSTRAINT constraint_name] CHECK (condition)`**
    - **Exemple :**
        - Dans la table des employés pour sélectionner le sexe d'une personne, la requête serait comme
        - **Vérifiez la contrainte au niveau de la colonne :**

```
CREATE TABLE employee
( id number(5) PRIMARY KEY,
name char(20),
dept char(10),
age number(2),
gender char(1) CHECK (gender in ('M','F')),
salary number(10),
location char(10)
);
```

apcpedagogie.com

- ## Contraintes d'intégrité sur une table

  - Les contraintes d'intégrité sur une table sont :

1. **PRIMARY KEY () : définit les attributs de la liste comme la clé primaire**

- La **clé primaire** pourra être déclarée à la fin de la requête de création de table, notamment si elle implique plusieurs champs comme dans le schéma suivant (si l'on considère une table où les couples (nom/prenom) sont uniques)

```
CREATE TABLE matable (nom VARCHAR(64) NOT NULL, prenom VARCHAR(64) NOT NULL, ..., PRIMARY KEY(nom,prenom))
```

- **Syntaxe pour définir une clé primaire au niveau de la table : `[CONSTRAINT constraint_name] PRIMARY KEY (column_name1,column_name2,..)`**
- **Exemple :**
  - **Clé primaire au niveau de la colonne :**

```
CREATE TABLE employee
( id number(5), NOT NULL,
name char(20),
dept char(10),
age number(2),
salary number(10),
location char(10),
ALTER TABLE employee ADD CONSTRAINT PK_EMPLOYEE_ID PRIMARY KEY (id)
);
```

- **UNIQUE () : interdit que deux tuples de la relation aient les mêmes valeurs pour l'ensemble des attributs de la liste.**

  - **Syntaxe pour définir une clé unique au niveau de la table : `[CONSTRAINT constraint_name] UNIQUE(column_name)`**
  - **Exemple :**
    - **Clé unique au niveau de la table**

```
CREATE TABLE employee
( id number(5) PRIMARY KEY,
name char(20),
dept char(10),
```

apcpedagogie.com

```
age number(2),
salary number(10),
location char(10),
CONSTRAINT loc_un UNIQUE(location)
);
```

- **FOREIGN KEY () REFERENCES () : contrôle l'intégrité référentielle entre les attributs de la liste et la table et ses colonnes spécifiées**
    - Cette contrainte identifie toute colonne référençant la PRIMARY KEY dans une autre table. Il établit une relation entre deux colonnes d'une même table ou entre des tables différentes. Pour qu'une colonne soit définie en tant que clé étrangère, elle doit être définie en tant que clé primaire dans la table à laquelle elle fait référence. Une ou plusieurs colonnes peuvent être définies comme clé étrangère.
    - **Syntaxe pour définir une clé étrangère au niveau de la table: : `[CONSTRAINT constraint_name] FOREIGN KEY(column_name) REFERENCES referenced_table_name(column_name);`**
    - **Exemple :**
    - **Clé étrangère au niveau de la table : :**

```
CREATE TABLE order_items
( order_id number(5) ,
product_id number(5),
product_name char(20),
supplier_name char(20),
unit_price number(10)
CONSTRAINT od_id_pk PRIMARY KEY(order_id),
CONSTRAINT pd_id_fk FOREIGN KEY(product_id) REFERENCES product(product_id)
);
```

- **CHECK () : contrôle la validité de la valeur des attributs spécifiés dans la condition dans le cadre d'une restriction de domaine**
    - **Exemple :**
        - **Vérifier la contrainte au niveau de la table :**

```
CREATE TABLE employee
( id number(5) PRIMARY KEY,
name char(20),
dept char(10),
age number(2),
```

apcpedagogie.com

```
gender char(1),
salary number(10),
location char(10),
CONSTRAINT gender_ck CHECK (gender in ('M','F'))
);
```

- ## Application

  - Soit le schéma de la base de données Bibliothèque suivante :

```
Etudiant(NumEtd,NomEtd,AdresseEtd)
Livre(NumLivre,TitreLivre,NumAuteur,NumEditeur,NumTheme,AnneeEdition)
Auteur(NumAuteur,NomAuteur,AdresseAuteur)
Editeur(NumEditeur,NomEditeur,AdresseEditeur)
Theme(NumTheme,IntituléTheme)
Prêt(NumEtd,NumLivre,DatePret,DateRetour)
```

- Un étudiant peut emprunter plusieurs livres à la fois. Chaque fois qu'un livre est retourné, la date de retour est mise à jour. Sinon sa valeur reste **null**. Pour des raisons de statistique, on conserve dans la table Prêt tous les tuples les livres retournés.

  - Un livre ne peut pas être emprunté le même jour ou il est retourné !
  - Donner les clés de ces relations. Justifier.
  - Donner toutes les contraintes d'intégrités référentielles qui apparaissent dans ce schéma

---

Sommaire du cours MySQL

Source : https://apcpedagogie.com/les-contraintes-dintegrite-avec-mysql · téléchargé le 03/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com