

MySQL : introduction

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

`elmouelhi.achref@gmail.com`



1 Environnement de travail

- MySQL Server
- Visual Studio Code

2 Type de données

- Nombres entiers
- Nombres décimaux
- Textes
- Dates
- Énumérations
- UUID
- BLOB
- Booléens
- Autres types
- Alias

MySQL

Téléchargement et installation

- Aller à <https://dev.mysql.com/downloads/mysql/?platform&os=3>
- Choisir la version à télécharger selon le système d'exploitation
- Lancer l'installation du fichier MSI sous **Windows** (fichier pkg de l'archive DMG sous MAC)

MySQL

Configuration : sous **Windows**, il faut ajouter le répertoire `bin` de **MySQL** au path de **Windows**

- Dans la barre de recherche, chercher `Système` ensuite cliquer `Paramètres système avancés`
- Choisir `Variables d'environnement` ensuite dans `Variables utilisateur` cliquer sur `Nouvelle`
- Saisir comme nom de variable `PATH` et comme valeur le chemin absolu du répertoire `bin` de **MySQL**

MySQL

Configuration : sous MAC, il faut exécuter

```
echo 'export PATH=/usr/local/mysql/bin:$PATH' >> ~/.profile
```

MySQL

Connexion à MySQL depuis une console

```
mysql --host=localhost --user=root --password=motdepasse
```

© Achref EL MOU

MySQL

Connexion à MySQL depuis une console

```
mysql --host=localhost --user=root --password=motdepasse
```

Ou aussi

```
mysql -h localhost -u root -p
```

MySQL

Propriété

- Le nom d'une base de données est sensible à la casse sur **Unix** mais pas sur **Windows**.
- Le noms d'une table est sensible à la casse sur **Unix** et pas sur **Windows**.

© Achref EL MOU

MySQL

Propriété

- Le nom d'une base de données est sensible à la casse sur **Unix** mais pas sur **Windows**.
- Le noms d'une table est sensible à la casse sur **Unix** et pas sur **Windows**.

On recommande de faire comme si **MySQL** est sensible à la casse

- Pour la portabilité du code,
- Pour éviter des éventuels problèmes avec les langages de programmation sensibles à la casse.

MySQL

Visual Studio Code (ou VSC)

- Gratuit.
- Extensible selon le langage de programmation.
- Recommandé par les développeurs front-end et même back-end.

MySQL

VSC : téléchargement

`code.visualstudio.com/download`

MySQL

VSC : extension pour **MySQL**

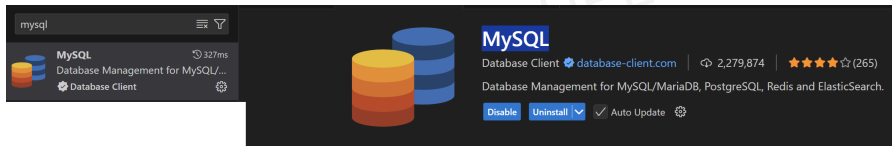
MySQL

© Achref EL MOUELHI ©

MySQL

VSC : extension pour MySQL

MySQL



mysql

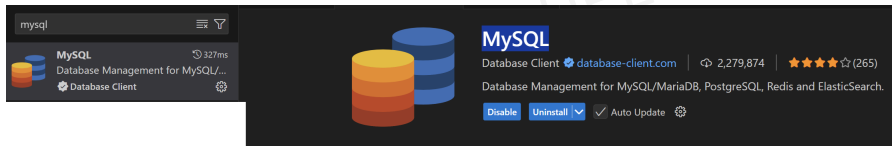
MySQL
Database Management for MySQL/...
Database Client

MySQL
Database Client database-client.com | 2,279,874 | ★★★★★ (265)
Database Management for MySQL/MariaDB, PostgreSQL, Redis and ElasticSearch.
Disable Uninstall Auto Update

MySQL

VSC : extension pour MySQL

MySQL



Première étape

Cliquez sur l'icône ajoutée à gauche et établissez la connexion avec le serveur.

MySQL

Quelques raccourcis VSC

- Pour activer la sauvegarde automatique : aller dans `File > AutoSave`
- Pour indenter son code : `Alt` `Shift` `f`
- Pour commenter/décommenter : `Ctrl` `:`
- Pour sélectionner toutes les occurrences : `Ctrl` `f2`
- Pour sélectionner l'occurrence suivante : `Ctrl` `d`
- Pour placer le curseur dans plusieurs endroits différents : `Alt`

MySQL

Nombres entiers

- `INT` : 4 octets (de -2 147 483 648 à 2 147 483 647)
- Autres types : `TINYINT` (1 octet), `SMALLINT` (2 octets), `MEDIUMINT` (3 octets), `BIGINT` (8 octets)
- L'attribut `UNSIGNED` supprime les valeurs négatives et double la plage positive.
- L'attribut `ZEROFILL` complète l'affichage avec des zéros en tête.
Exemple : `INT(3) ZEROFILL` avec la valeur 15 affiche 015.
- `ZEROFILL` impose `UNSIGNED` automatiquement.
- Depuis **MySQL 8.0.17**, `ZEROFILL` et la largeur d'affichage (exemple : `INT(3)`) sont dépréciés et n'affectent plus la présentation.

MySQL

Types entiers : plages de valeurs

Type	Octets	Plage (signé)
TINYINT	1	-128 à 127
SMALLINT	2	-32 768 à 32 767
MEDIUMINT	3	-8 388 608 à 8 388 607
INT / INTEGER	4	-2 147 483 648 à 2 147 483 647
BIGINT	8	-9 223 372 036 854 775 808 à 9 223 372 036 854 775 807

UNSIGNED : valeurs uniquement positives \Rightarrow plage doublée.

MySQL

Nombres décimaux

- **NUMERIC** et **DECIMAL** acceptent deux paramètres (n , m) : n étant le nombre total de chiffres, dont m après la virgule. En cas de précision d'un seul paramètre, la valeur de m sera 0.
- **REAL** et **DOUBLE** peuvent être écrits avec des paramètres, mais **MySQL** les ignore.
- **FLOAT** peut être utilisé avec ou sans paramètre.
- **NUMERIC** et **DECIMAL** sont stockés sous forme de chaînes de caractères.
- Pour les autres, une valeur approchée sera enregistrée (attention aux erreurs d'arrondi lors des comparaisons).

MySQL

NUMERIC, FLOAT, REAL, DOUBLE, FIXED : différences

Type	Stockage	Précision	Usage typique
FLOAT	4 octets	Approximative	Calculs rapides
DOUBLE / REAL	8 octets	Approximative	Calculs scientifiques
NUMERIC / DECIMAL / FIXED	Variable	Exacte	Finance, montants précis

MySQL

Chaînes de caractères

- CHAR et VARCHAR : limités à 255 caractères. Il faut indiquer la longueur maximale.
- TEXT (jusqu'à 2^{16} octets)
- TINYTEXT (jusqu'à 2^8 octets)
- MEDIUMTEXT (jusqu'à 2^{24} octets)
- LONGTEXT (jusqu'à 2^{32} octets)

Différences entre CHAR et VARCHAR

- CHAR (n) :
 - Stocke une longueur fixe de caractères (complété par des espaces).
 - Plus rapide pour des chaînes de longueur constante.
 - Idéal pour des données comme des codes postaux, numéros de tél...
- VARCHAR (n) :
 - Stocke une longueur variable avec 2 octets supplémentaires pour la taille.
 - Plus économe en espace pour des longueurs de chaîne variables.
 - Préféré pour des textes comme les noms ou les adresses e-mail.
- **Remarque :** CHAR est plus rapide sur des chaînes courtes de longueur fixe, tandis que VARCHAR est plus efficace pour des données de longueur variable.

MySQL

Les dates

- `DATE` : date (format : année-mois-jour).
- `TIME` : heure (format : heure-minute-seconde).
- `DATETIME` : date et heure.
- `TIMESTAMP` : nombre de secondes écoulées depuis le 01/01/1970.
- **Attention** : `TIMESTAMP` est sensible au fuseau horaire, contrairement à `DATETIME`.

MySQL

Quand utiliser ?

- **DATETIME** : pour stocker des dates fixes, indépendantes du serveur
 - Exemple : date de naissance, échéance, réunion prévue
- **TIMESTAMP** : pour suivre des événements liés au système
 - Exemple : date de création ou modification automatique d'une ligne

MySQL

Les énumérations

- `ENUM` et `SET` permettent de définir une liste de valeurs.
- Ces types sont propres à **MySQL**.
- Exemple : `ENUM("foot", "tennis", "hockey")`.
- `ENUM` accepte une seule valeur parmi celles définies, tandis que `SET` peut en accepter plusieurs simultanément.
- `SET` est limité à 64 éléments max.
- `SET` est plus complexe à manipuler (notamment en `SELECT`, car il faut gérer des chaînes séparées par des virgules).

MySQL

Différences principales

- **ENUM** : permet de choisir **une seule** valeur parmi une liste fixe.
- **SET** : permet de choisir **plusieurs** valeurs parmi une liste fixe.

© Achref EL MOUELHI

MySQL

Différences principales

- **ENUM** : permet de choisir **une seule** valeur parmi une liste fixe.
- **SET** : permet de choisir **plusieurs** valeurs parmi une liste fixe.

Exemples

- ENUM :
 - couleur `ENUM('rouge', 'vert', 'bleu')`
 - Une seule couleur possible : 'rouge' ou 'vert' ou 'bleu'.
- SET :
 - options `SET('wifi', 'climatisation', 'balcon')`
 - Plusieurs options possibles : 'wifi', 'wifi,balcon', 'climatisation,balcon'...

MySQL

Les **UUID** (Universally Unique Identifier)

- un identifiant unique de 128 bits
- généralement représenté sous forme de chaîne hexadécimale de 36 caractères
- Exemple : 550e8400-e29b-41d4-a716-446655440000

© Achref EL MOU

MySQL

Les **UUID** (Universally Unique Identifier)

- un identifiant unique de 128 bits
- généralement représenté sous forme de chaîne hexadécimale de 36 caractères
- Exemple : 550e8400-e29b-41d4-a716-446655440000

Les différentes versions de **UUID**

- **UUID v1** : basé sur l'horodatage et l'adresse **MAC**.
- **UUID v4** : basé sur des nombres aléatoires (le plus courant).
- **UUID v7** : basé sur l'horodatage avec une disposition optimisée pour le tri et les bases de données (recommandé dans **MySQL** $\geq 8.0.31$).

UUID v7

- **Basé sur un horodatage** : permet un tri naturel des enregistrements par date.
- **Optimisé pour les index** : recommandé pour les bases de données.
- **Performant en lecture/écriture** : surtout sur de grandes tables.
- **Structure d'un UUID v7 (128 bits)** : Unix timestamp (48 bits) + portion aléatoire (80 bits)

Types Binaires **BLOB** : (Binary Large Object)

Pour stocker des données binaires (images, vidéos, fichiers)

- TINYBLOB : jusqu'à 255 octets
- BLOB : jusqu'à 65 535 octets
- MEDIUMBLOB : jusqu'à 16 Mo
- LONGBLOB : jusqu'à 4 Go

Particularité du type `BOOLEAN` en **MySQL**

- `BOOLEAN` est un alias de **`TINYINT(1)`**
- `0 = FALSE`
- `1 = TRUE`
- Aucune contrainte automatique : **autres valeurs possibles** (2, 100...)

MySQL

Type **JSON**

stocke des structures au format **JSON**

© Achref EL MOUELHI ©

MySQL

Type **JSON**

stocke des structures au format **JSON**

Types Géospatiaux (**GIS**)

POINT, LINESTRING, POLYGON, GEOMETRY : pour les données géospatiales

MySQL

Type **JSON**

stocke des structures au format **JSON**

Types Géospatiaux (**GIS**)

POINT, LINESTRING, POLYGON, GEOMETRY : pour les données géospatiales

Types Spéciaux

- BIT (n) : stocke des valeurs binaires (BIT (8) pour 8 bits)
- YEAR : stocke des années (de 1901 à 2155)

MySQL

Alias de types courants en MySQL

Alias	Type réel
BOOL, BOOLEAN	TINYINT(1)
INT, INTEGER	INT(11)
DEC, DECIMAL	DECIMAL
NUMERIC	DECIMAL
FIXED	DECIMAL
DOUBLE PRECISION	DOUBLE
REAL	DOUBLE
CHARACTER	CHAR
CHARACTER VARYING	VARCHAR
NCHAR	CHAR (Unicode)
NVARCHAR	VARCHAR (Unicode)
BINARY VARYING	VARBINARY

MySQL

Documentation en français

<http://sql.sh/>