**HEAD -**

**Titre** : Continuons à travailler\_ *Synthèses des Tâches*

**Mots clés** : Drupal, Docker, MariaDB, Base de données, installation, connexion, Extensions, modules, tutoriels,

**BODY -**

**Header**

**logo** :

**Nav** :

**H1** :

Installation de Drupal dans un Conteneur Docker sur une VM Ubuntu

**P-0** :

*Vers une automatisation avancée des tâches à venir*

**H2-0** : Segments de la tâche

Menu burger

**ol-0**: point de factorisation

1- Préparation : l'environnement Docker

2- Installation : les images Docker de Drupal et MariaDB

3- Configuration : la connexion Drupal / MariaDB et la GUI administrateur

4- Développement : Drupal, un CMS très riche en potentialités

5- Sécurité : la persistance des données et la gestion des permissions

6- Troubleshooting : ajustements nécessaires aux problèmes de sécurité

7- Mémoire : ?

**Main**

**H2-1** : Bases de données et paramètres de connexion dans Docker

**H3-0** : les axes de travail

**ol-1**: un trio particulièrement interdépendant

1. Configuration des paramètres de connexion via Docker
2. Synchronisation des permissions entre Docker et la BDD MariaDB
3. Résolution des problèmes d’accès réseaux de la VM Ubuntu

**P-1** :

Ce qui rend la résolution très vite chronophage réside dans le fait que la cause d’un échec concret lors d’une étape peut provenir des trois axes, séparément comme simultanément. Une action de résolution d’un côté provoquait d’un autre côté une besoin d’ajustement dont je n’avais pas encore conscience. Si ce tâtonnement est nécessaire à l’apprentissage pour produire une conduite et une confiance procédurale, il convient, je crois, de finaliser les documents protocolaires au fur et à mesure du projet afin que la découverte consentie ne devienne pas juste une errance.

**H3-1** : les codes utiles

**ol-2**: le SGBD MariaDB du conteneur déployé exigeait sa propre commande

1. Accès : mariadb -u drupal -p drupal drupal
2. Manipulation : Lien\_001
3. Vérification : Lien\_002

**H3-2** : les objets d’étude

**ol-3**: le fichier « docker-compose.yml » borne l’administration des services

1. Accès : créér le fichier sur le niveau du répertoire Drupal et les images
2. ?
3. ?

**Article-0**

**Section-0**

**H2-2** : Préparation de la machine virtuelle et installation de Docker

**H3-3** : les codes utiles

**H4-0** : dépendances et signature des fichiers : un problème gris très sérieux

**P-2** : Jusqu’ici tout va bien…

Les contraintes matérielles et le rythme de la formation ne nous permettent pas forcément de faire face aux problèmes gris. Les formateurs eux même nous conseillent tantôt d’ignorer certains messages d’erreur. Nous n’avions pas le temps, certes, mais il faudra pourtant, dès que nous l’aurons regagné, trouver le courage de tout recommencer à « 0 » pour faire face aux problèmes gris car ce sont des symptômes de problèmes et d’échecs pour demain.

**ol-4** : implémentation et configuration des dépendances pour Docker

1. Accès : ouvrir le Terminal de la VM Ubuntu
2. Installation : Lien\_003 : reprendre le processus complet sans problème gris
3. Vérification : Lien\_004 : reprendre la liste et le protocole de vérification

**H3-4** : l’objet d’étude

**H4-1** : Docker : contextes, conteneur similaires et doublons d’images…

**P-3** : Profiter des avantages de Docker impose une gestion très rigoureuse

La multiplication des projets, les deux contextes d’environnements à ne pas confondre si Docker desktop est installé, le pluralité des conteneurs, la variété des images que l’on peut ou doit manipuler ainsi que la confusion entre celles qui sont déjà disponibles et celles qu’il faudra que le logiciel trouve et télécharge depuis le Hub produisent échecs et messages d’erreurs à répétition. Nettoyer l’espace de travail sans une vigilance accrue brise tantôt le fonctionnement des projets antérieurs, des fonctionnalités connexes. C’est un pas utile et indispensable mais qui nécessite une grande prudence.

Dans le cadre de ce projet, puisqu’il s’agit d’une livraison pour un client et qui plus est mon support de soutenance pour la validation de mon titre professionnel, j’ai préféré tous désinstaller, tout supprimer et tout réinstaller.

**ol-5** : installation et configuration de Docker

1. Accès : ouvrir le Terminal de la VM Ubuntu
2. Installation : Lien : reprendre le processus complet sans problème gris
3. Vérification : Lien : reprendre la liste et le protocole de vérification

Memo : sudo systemctl enable **--now** docker

**Section-1**

**H2-3** : Télécharger et Installer les images Docker de Drupal et MariaDB

**H3-5** : les codes utiles

**H4-1** : Faire face à un réseau très restreint par des mesures sécuritaires

**P-3** : Expérimenter les approches « hors-réseau » pour la machine cible

Ma VM Ubuntu avait été initialement conçue pour se situer en arrière-plan dans une infrastructure où, solidaire avec d’autres Debian et Ubuntu elle contribuerait à la redondance et à la « High Abilify » du site. A ce stade, je ne pouvais donc même pas encore accéder via internet aux dépôts de Git Hub ni du site officiel de Drupal. Il me fallait donc entreprendre un pas à pas différent de la « voie royale ». Dans le cas d’un Titre Professionnel de Développeur Web, ces objectifs peuvent être secondaires. Je devais donc les gérer parallèlement et à des rythmes différents. Pour bonifier le cumule d’initiatives, plutôt que de renoncer à la fermeture, pour expérimenter un cas de protection avancé, j’ai décidé réaliser l’installation avec une méthode plus « manuelle ».

La connexion SSH n’étant pas non plus disponible, le but était de télécharger les paquets via un autre ordinateur, de les convertir dans un format favorable à Ubuntu si besoin, puis, pour caractériser le cas d’école, de les transporter sur la VM avec une clé USB virtuelle. La finalité était de trouver, dès le début, par anticipation, des modus operandi qui me permettraient de terminer et de livrer le projet au client même si je rencontrais de graves difficultés de réseau.

**ol-5** : téléchargement des paquets pour un transport physique sécurisé

1. Accès : Terminal de Ubuntu ; Internet via un autre ordinateur ; clé USB
2. Installation : Lien : reprendre le processus complet sans problème gris
3. Vérification : Lien : reprendre la liste et le protocole de vérification

Memo : sudo systemctl enable **--now** docker

**H3-6** : l’objet d’étude

**H4-2** :  montage de disques externes et partage de dossier sur VirtualBox

**P-4** : ?

**ol-6** : ?

1. Accès : GUI de VirtualBox et Terminal de la VM
2. Installation : Lien : reprendre le processus complet sans problème gris
3. Vérification : Lien : reprendre la liste et le protocole de vérification

Memo : 🡪 docker pull drupal:11-apache ; 🡪 docker pull mariadb:latest

**Section-2**

**H2-4** : Configuration de la connexion entre Drupal et la BDD MariaDB

**H3-7** :  les codes utiles

**H4-3** : L’installation manuelle ne profite pas des setup automatiques…

**P-5** : Quand tout se passe bien au niveau des accès aux dépôt officiels par les commande du Terminal, on bénéficie souvent de processus de configuration automatisés dont on ignore même l’existence tant que la procédure manuelle ne nous met pas face aux messages d’erreurs. Dans ce cas précis, j’avais beaux renseigner les authentifications et mots de passes temporaires, la GUI web de Drupal ne m’ouvrait pas ses portes.

**ol-7** : Identifier les fichiers de configuration d’autorités supérieures

1. Accès : le fichier « settings.php »
2. Installation : Lien : reprendre le processus complet sans problème gris
3. Vérification : Lien : reprendre la liste et le protocole de vérification

**H3-8** : l’objet d’étude

**H4-4** : Retour au Dockerfile

**P-6** : Regards croisés sur le Dockerfile et docker-compose.yml

Si le fichier docker-compose.yml nous sert à configurer l’interopérabilité de plusieurs conteneurs, c’est le fichier dockerfile qui nous nous sera utile pour approfondir les besoin d’un conteneur spécifique. Son étude m’a permis de mieux comprendre les autres fichiers d’autorité supérieure du contexte et de me préparer à faire face à d’autres difficultés à venir pour d’autres projets.

**ol-8** : ?

1. Accès : Terminal de la VM Ubuntu
2. Installation : Lien : reprendre le processus complet sans problème gris
3. Vérification : Lien : reprendre la liste et le protocole de vérification

Memo : 🡪 sudo docker exec -it drupal\_db mysql -u drupal -p drupal drupal

**Section-3**

**H2-5** : ?

**H3-9** : les codes utiles

**H4-5** : ?

**P-7** : ?

?

**ol-9** : ?

1. Accès : ?
2. Installation : Lien : reprendre le processus complet sans problème gris
3. Vérification : Lien : reprendre la liste et le protocole de vérification

**H3-10** : l’objet d’étude

**H4-6** : ?

**P-8** : ?

?

**ol-10** : ?

1. Accès : ?
2. Installation : Lien : reprendre le processus complet sans problème gris
3. Vérification : Lien : reprendre la liste et le protocole de vérification

**Footer -**

**credit**  : Serge Hounhouayenou

**contacts** : serge@corpulence.fr

**Réseaux** : Liens

Lien\_001

1. Accès et authentification : se connecter à la BDD depuis le Terminal

🡪 mariadb -u drupal -p drupal drupal

// mariadb : commande principale : demander la connexion à la base de données

// -u : option : préciser l'utilisateur avec lequel on veut se connecter (drupal)

// -p : option : préciser le mot de passe de l'utilisateur en question (drupal)

// drupal (le dernier de la ligne de code) : valeur : nom de la base de données voulue

Lien\_001

1. Gestion des permissions : utilisateurs et rôles : sécuriser l'accès à la BDD

🡪 GRANT ALL PRIVILEGES ON drupal.\* TO 'drupal'@'localhost' IDENTIFIED BY 'password';

// GRANT ALL PRIVILEGES : commande pour accorder tous les privilèges.

// ON drupal.\* : spécifie la base de données et toutes ses tables.

// TO 'drupal'@'localhost' : utilisateur et hôte à qui les privilèges sont accordés.

// IDENTIFIED BY 'password' : mot de passe de l'utilisateur.

Bulle\_L1\_01 : explication pour "ON drupal.\*" :

Exemple de cas d'école pour donner des privilèges précis à une seule table :

GRANT SELECT, INSERT ON drupal.users TO 'drupal'@'localhost';

// GRANT SELECT, INSERT : accorde les privilèges de sélection et d'insertion.

// ON drupal.users : spécifie la table "users" dans la base de données "drupal".

// TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

Bulle-1 d'explication pour "TO 'drupal'@'localhost'" :

TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

// `localhost` : signifie que la base de données est accessible via le port du localhost, c'est-à-dire la machine locale.

// `remote\_host` : est la valeur pour autoriser n'importe quel hôte distant à effectuer une requête vers notre BDD. Cela peut être une autre machine sur le réseau ou un serveur accessible via Internet.

// `remote\_host` n'est pas un mot clé, il peut être remplacé par le nom d'une machine distante ou une adresse IP spécifique. Cela signifie que n'importe quelle machine distante peut effectuer une requête si elle est spécifiée.

// Importance : Définir des droits sur la base de données pour des hôtes spécifiques permet de contrôler et sécuriser l'accès, en limitant les connexions à des machines autorisées.

Bulle-2 d'explication pour "TO 'drupal'@'localhost'" :

// Exemple de cas d'école pour spécifier un utilisateur et un hôte différents :

GRANT SELECT ON drupal.\* TO 'user'@'192.168.1.100';

// TO 'user'@'192.168.1.100' : utilisateur "user" et adresse IP "192.168.1.100" auxquels les privilèges sont accordés.

GRANT SELECT ON drupal.\* TO 'user'@'nom\_machine';

// TO 'user'@'nom\_machine' : utilisateur "user" et nom de la machine "nom\_machine" auxquels les privilèges sont accordés.

GRANT SELECT ON drupal.\* TO 'user'@'serveur\_exemple.com';

// TO 'user'@'serveur\_exemple.com' : utilisateur "user" et serveur "serveur\_exemple.com" auxquels les privilèges sont accordés.

Lien\_001

1. Gestion de la connexion Drupal/MariaDB: paramètres, ports, et adresses IP

Accès : 🡪 settings.php 🡪 dans le répertoire sites/default du répertoire Drupal.

// Paramètres de connexion : définir les informations de connexion entre Drupal et MariaDB.

$databases['default']['default'] = array (

'database' => 'drupal', // 'database' : nom de la base de données.

'username' => 'drupal', // 'username' : nom d'utilisateur pour se connecter à la base de données.

'password' => 'password', // 'password' : mot de passe de l'utilisateur.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

);

Bulle d'explication pour 'default'

$databases['default']['default'] = array (

// 'default' : première valeur 'default' spécifie le groupe de base de données par défaut.

// 'default' : deuxième valeur 'default' spécifie la connexion par défaut pour ce groupe de base de données.

// Autres valeurs possibles : Vous pouvez définir plusieurs groupes de bases de données et connexions. Par exemple, $databases['default']['backup'] pour une connexion de sauvegarde.

// Tableau multidimensionnel : Utilisé pour organiser les connexions de manière structurée. Un tableau simple pourrait contenir les mêmes paires clé/valeur, mais un tableau multidimensionnel permet de gérer plusieurs connexions et groupes de bases de données.

// Exemple de déclaration de plusieurs groupes de connexion :

// Paramètres de connexion : définir les informations de connexion entre Drupal et plusieurs bases de données.

$databases = array (

'default' => array (

'default' => array (

…… content. Voir l’exemple plus haut……..

),

'backup' => array (

'database' => 'drupal\_backup', // 'database' : nom de la base de données de sauvegarde.

…… content. Voir l’exemple plus haut……..

),

'analytics' => array (

'default' => array (

'database' => 'drupal\_analytics', // 'database' : nom de la base de données d'analyse.

…… content. Voir l’exemple plus haut……..

);

Bulle d'explication pour 'host'

'host' => 'localhost',

// 'localhost' : signifie que la base de données est hébergée sur la machine locale.

// Exemple de cas d'école pour spécifier un hôte distant :

'host' => '192.168.1.100';

// '192.168.1.100' : adresse IP de l'hôte distant où la base de données est hébergée.

Bulle d'explication pour 'port'

'port' => '3306',

// '3306' : numéro de port par défaut utilisé par MariaDB pour les connexions.

// Exemple de cas d'école pour spécifier un port différent :

'port' => '3307';

// '3307' : numéro de port alternatif utilisé pour les connexions à la base de données.

Nous pouvons continuer avec les autres catégories

Autre exemple de groupes de connexion multiple dans settings.php

// Paramètres de connexion : définir les informations de connexion entre Drupal et plusieurs bases de données.

$databases['default']['default'] = array (

'database' => 'drupal', // 'database' : nom de la base de données.

'username' => 'drupal', // 'username' : nom d'utilisateur pour se connecter à la base de données.

'password' => 'password', // 'password' : mot de passe de l'utilisateur.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données est hébergée. [Survol pour plus d'explications]

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données. [Survol pour plus d'explications]

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

);

$databases['default']['backup'] = array (

'database' => 'drupal\_backup', // 'database' : nom de la base de données de sauvegarde.

'username' => 'drupal\_backup', // 'username' : nom d'utilisateur pour se connecter à la base de données de sauvegarde.

'password' => 'backup\_password', // 'password' : mot de passe de l'utilisateur de sauvegarde.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données de sauvegarde est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données de sauvegarde.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

);

$databases['analytics']['default'] = array (

'database' => 'drupal\_analytics', // 'database' : nom de la base de données d'analyse.

'username' => 'drupal\_analytics', // 'username' : nom d'utilisateur pour se connecter à la base de données d'analyse.

'password' => 'analytics\_password', // 'password' : mot de passe de l'utilisateur d'analyse.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données d'analyse est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données d'analyse.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

);

Bulle d'explication pour 'default'

$databases['default']['default'] = array (

// 'default' : première valeur 'default' spécifie le groupe de base de données par défaut.

// 'default' : deuxième valeur 'default' spécifie la connexion par défaut pour ce groupe de base de données.

// Autres valeurs possibles : Vous pouvez définir plusieurs groupes de bases de données et connexions. Par exemple, $databases['default']['backup'] pour une connexion de sauvegarde.

// Le fichier settings.php est crucial pour la configuration des connexions à la base de données dans Drupal. Voici comment cela fonctionne: Par défaut: Si vous ne spécifiez pas de groupe de connexion particulier, les valeurs par défaut définies dans $databases['default']['default'] seront utilisées1.

Groupes spécifiques: Vous pouvez nommer des groupes de connexion spécifiques comme backup ou analytics pour des connexions particulières1. Par exemple, si vous avez déclaré un groupe backup, vous pouvez l'utiliser pour vous connecter à une base de données de sauvegarde

// Tableau multidimensionnel : Utilisé pour organiser les connexions de manière structurée. Un tableau simple pourrait contenir les mêmes paires clé/valeur, mais un tableau multidimensionnel permet de gérer plusieurs connexions et groupes de bases de données.

Bulle d'explication pour 'host'

'host' => 'localhost',

// 'localhost' : signifie que la base de données est hébergée sur la machine locale.

// Exemple de cas d'école pour spécifier un hôte distant :

'host' => '192.168.1.100';

// '192.168.1.100' : adresse IP de l'hôte distant où la base de données est hébergée.

Bulle d'explication pour 'port'

'port' => '3306',

// '3306' : numéro de port par défaut utilisé par MariaDB pour les connexions.

// Exemple de cas d'école pour spécifier un port différent :

'port' => '3307';

// '3307' : numéro de port alternatif utilisé pour les connexions à la base de données.

Pour répondre à votre deuxième question, la valeur "default" dans $databases['default']['default'] ne désigne pas ce que nous avons renseigné dans le fichier docker-compose.yml. Elle spécifie le groupe de base de données par défaut et la connexion par défaut pour ce groupe de base de données dans le fichier settings.php. Le fichier docker-compose.yml est utilisé pour configurer les services Docker, tandis que le fichier settings.php est utilisé pour configurer les paramètres de connexion de Drupal à la base de données.

Lien\_001

1. Sécurité et sauvegarde : chiffrement des connexions et sauvegarde régulière des BDD

🡪 Chiffrement des connexions : activer SSL pour les connexions à la base de données.

$databases['default']['default'] = array (

'database' => 'drupal', // 'database' : nom de la base de données.

'username' => 'drupal', // 'username' : nom d'utilisateur pour se connecter à la base de données.

'password' => 'password', // 'password' : mot de passe de l'utilisateur.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

'pdo' => array(

PDO::MYSQL\_ATTR\_SSL\_CA => '/path/to/ca-cert.pem', // : chemin vers le certificat CA.

PDO::MYSQL\_ATTR\_SSL\_CERT => '/path/to/client-cert.pem', // : chemin vers le certificat client.

PDO::MYSQL\_ATTR\_SSL\_KEY => '/path/to/client-key.pem', // : chemin vers la clé privée du client.

),

);

🡪 Sauvegarde régulière des BDD dans un nouveau fichier :

# Script de sauvegarde de la base de données Drupal.

# Sauvegarde quotidienne à 2h du matin.

0 2 \* \* \* /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_$(date +\%F).sql

### Sauvegarde régulière des BDD :

### 0 2 \* \* \* : planification de la tâche cron pour s'exécuter tous les jours à 02h00

### /usr/bin/mysqldump : commande pour effectuer une sauvegarde de la BDD

### -u drupal : option pour spécifier l'utilisateur de la base de données

### -p'password' : option pour spécifier le mot de passe de l'utilisateur

### drupal : nom de la base de données à sauvegarder

### > /path/to/backup/drupal\_backup\_$(date +\%F).sql : chemin et nom + date du jour

🡪 Sauvegarde différentielle :

La sauvegarde différentielle consiste à sauvegarder uniquement les données qui ont changé depuis la dernière sauvegarde complète. Voici un exemple de script de sauvegarde différentielle pour une base de données MySQL:

# Script de sauvegarde différentielle de la base de données Drupal.

# Sauvegarde complète hebdomadaire à 2h du matin le dimanche.

0 2 \* \* 0 /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_full\_$(date +\%F).sql

# Sauvegarde différentielle quotidienne à 2h du matin.

0 2 \* \* 1-6 /usr/bin/mysqldump -u drupal -p'password' --single-transaction --quick --lock-tables=false --incremental --incremental-base=history:LAST\_BACKUP drupal > /path/to/backup/drupal\_backup\_diff\_$(date +\%F).sql

Explications :

Sauvegarde complète: La sauvegarde complète est effectuée une fois par semaine, le dimanche à 2h du matin (0 2 \* \* 0). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_full\_$(date +\%F).sql).

Sauvegarde différentielle: La sauvegarde différentielle est effectuée quotidiennement, du lundi au samedi à 2h du matin (0 2 \* \* 1-6). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_diff\_$(date +\%F).sql).

La commande mysqldump avec les options --single-transaction, --quick, --lock-tables=false, --incremental, et --incremental-base=history:LAST\_BACKUP permet de réaliser une sauvegarde différentielle en ne sauvegardant que les changements depuis la dernière sauvegarde complète.

🡪 Sauvegarde incrémentielle:

La sauvegarde incrémentielle consiste à sauvegarder uniquement les fichiers qui ont été modifiés depuis la dernière sauvegarde complète ou incrémentielle. Cela permet de réduire la taille des fichiers de sauvegarde et de minimiser l'espace de stockage nécessaire. Voici un exemple de script de sauvegarde incrémentielle pour une base de données MySQL:

# Script de sauvegarde incrémentielle de la base de données Drupal.

# Sauvegarde complète hebdomadaire à 2h du matin le dimanche.

0 2 \* \* 0 /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_full\_$(date +\%F).sql

# Sauvegarde incrémentielle quotidienne à 2h du matin.

0 2 \* \* 1-6 /usr/bin/mysqldump -u drupal -p'password' --single-transaction --quick --lock-tables=false --incremental --incremental-base=history:LAST\_BACKUP drupal > /path/to/backup/drupal\_backup\_inc\_$(date +\%F).sql

Explications

Sauvegarde complète: La sauvegarde complète est effectuée une fois par semaine, le dimanche à 2h du matin (0 2 \* \* 0). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_full\_$(date +\%F).sql).

Sauvegarde incrémentielle: La sauvegarde incrémentielle est effectuée quotidiennement, du lundi au samedi à 2h du matin (0 2 \* \* 1-6). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_inc\_$(date +\%F).sql).

La commande mysqldump avec les options --single-transaction, --quick, --lock-tables=false, --incremental, et --incremental-base=history:LAST\_BACKUP permet de réaliser une sauvegarde incrémentielle en ne sauvegardant que les changements depuis la dernière sauvegarde complète ou incrémentielle

🡪 Créeation des scripts:

Sauvegarde complète et incrémentielle

Ouvrez un éditeur de texte (comme VS Code ou Nano).

Créez un nouveau fichier et nommez-le backup\_script.sh.

Ajoutez le contenu du script de sauvegarde complet et incrémentiel:

#!/bin/bash

# Script de sauvegarde complète hebdomadaire à 2h du matin le dimanche.

0 2 \* \* 0 /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_full\_$(date +\%F).sql

# Script de sauvegarde incrémentielle quotidienne à 2h du matin.

0 2 \* \* 1-6 /usr/bin/mysqldump -u drupal -p'password' --single-transaction --quick --lock-tables=false --incremental --incremental-base=history:LAST\_BACKUP drupal > /path/to/backup/drupal\_backup\_inc\_$(date +\%F).sql

🡪 Chiffrement des connexions :

Ouvrez un éditeur de texte (comme VS Code ou Nano).

Créez un nouveau fichier et nommez-le settings.php.

Ajoutez le contenu du script de chiffrement des connexions:

// Activer le chiffrement SSL pour les connexions à la base de données.

$databases['default']['default'] = array (

'database' => 'drupal',

'username' => 'drupal',

'password' => 'password',

'host' => 'localhost',

'port' => '3306',

'driver' => 'mysql',

'prefix' => '',

'pdo' => array(

PDO::MYSQL\_ATTR\_SSL\_CA => '/path/to/ca-cert.pem',

PDO::MYSQL\_ATTR\_SSL\_CERT => '/path/to/client-cert.pem',

PDO::MYSQL\_ATTR\_SSL\_KEY => '/path/to/client-key.pem',

),

);

🡪 Enregistrement des scripts :

Enregistrez le fichier backup\_script.sh dans un répertoire approprié sur votre serveur, par exemple /path/to/scripts/.

Enregistrez le fichier settings.php dans le répertoire sites/default du dossier Drupal.

🡪 Activation des scripts pour les rendre exécutables

Ouvrez le terminal et naviguez jusqu'au répertoire où vous avez enregistré backup\_script.sh.

Exécutez la commande suivante pour rendre le script exécutable:

chmod +x /path/to/scripts/backup\_script.sh

🡪 Planification de l'exécution des scripts:

Ouvrez le fichier de configuration des tâches cron:

crontab -e

Ajoutez les lignes suivantes pour planifier l'exécution des scripts de sauvegarde:

0 2 \* \* 0 /path/to/scripts/backup\_script.sh

0 2 \* \* 1-6 /path/to/scripts/backup\_script.sh

🡪 Vérification et tester les scripts:

Exécutez manuellement les scripts pour vérifier qu'ils fonctionnent correctement:

/path/to/scripts/backup\_script.sh

Lien\_001

1. Maintenance et optimisation : Exécution des tâches de maintenance de la BDD

🡪 quelques étapes importantes à suivre 🡪 liste non-exhaustive

Vérification de l'intégrité des tables de la BDD

// Vérifie l'intégrité de la table users dans la base de données drupal

CHECK TABLE drupal.users;

// Vérifie l'intégrité de la table nodes dans la base de

CHECK TABLE drupal.nodes; données drupal

Optimisation des tables de BDD pour améliorer les performances

// Optimise la t able users pour améliorer les performances

OPTIMIZE TABLE drupal.users;

// Optimise la table nodes pour améliorer les performances

OPTIMIZE TABLE drupal.nodes;

Résolution des problèmes de la BDD :

// Répare la table users en cas de problèmes

REPAIR TABLE drupal.users;

// Répare la table nodes en cas de problèmes.

REPAIR TABLE drupal.nodes;

Identification des autres besoins spécifiques de maintenance :

// Analyse la table users pour collecter des statistiques

ANALYZE TABLE drupal.users;

// Analyse la table nodes pour collecter des statistiques

ANALYZE TABLE drupal.nodes;

Lien\_001

1. Dépannage : Identification et résolution de la BDD

🡪 les problèmes de base les plus courants 🡪 liste non-exhaustive

les erreurs de connexion

// Vérifier les paramètres de connexion dans le fichier settings.php.

$databases['default']['default'] = array (

'database' => 'drupal', // 'database' : nom de la base de données.

'username' => 'drupal', // 'username' : nom d'utilisateur pour se connecter à la base de données.

'password' => 'password', // 'password' : mot de passe de l'utilisateur.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

);

Bulle d'explication pour 'host'

'host' => 'localhost',

// 'localhost' : signifie que la base de données est hébergée sur la machine locale.

// Exemple de cas d'école pour spécifier un hôte distant :

'host' => '192.168.1.100';

// '192.168.1.100' : adresse IP de l'hôte distant où la base de données est hébergée.

// Vérifier les privilèges de l'utilisateur dans la base de données.

GRANT ALL PRIVILEGES ON drupal.\* TO 'drupal'@'localhost' IDENTIFIED BY 'password';

// GRANT ALL PRIVILEGES : commande pour accorder tous les privilèges.

// ON drupal.\* : spécifie la base de données et toutes ses tables.

// TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

// IDENTIFIED BY 'password' : mot de passe de l'utilisateur.

Bulle d'explication pour 'TO 'drupal'@'localhost''

TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

// `localhost` : signifie que la base de données est accessible via le port du localhost, c'est-à-dire la machine locale.

// `remote\_host` : est la valeur pour autoriser n'importe quel hôte distant à effectuer une requête vers notre BDD. Cela peut être une autre machine sur le réseau ou un serveur accessible via Internet.

// `remote\_host` n'est pas un mot clé, il peut être remplacé par le nom d'une machine distante ou une adresse IP spécifique. Cela signifie que n'importe quelle machine distante peut effectuer une requête si elle est spécifiée.

// Importance : Définir des droits sur la base de données pour des hôtes spécifiques permet de contrôler et sécuriser l'accès, en limitant les connexions à des machines autorisées.

Est-ce que cela répond à vos attentes pour le Problème de base N°1 ? Si vous avez besoin de plus de

les problèmes de performance

EXPLAIN SELECT \* FROM drupal.users;

// EXPLAIN : commande pour analyser et expliquer l'exécution d'une requête SQL.

// SELECT \* FROM drupal.users : requête pour sélectionner toutes les lignes de la table 'users' dans la base de données 'drupal'.

-- Optimiser les requêtes SQL pour améliorer les performances.

OPTIMIZE TABLE drupal.users;

// OPTIMIZE TABLE : commande pour optimiser une table de la base de données.

// drupal.users : spécifie la table 'users' dans la base de données 'drupal'.

Bulle d'explication pour 'EXPLAIN'

EXPLAIN : commande pour analyser et expliquer l'exécution d'une requête SQL.

// Utilisée pour identifier les problèmes de performance et optimiser les requêtes.

les conflits de permissions.

SHOW GRANTS FOR 'drupal'@'localhost';

// SHOW GRANTS : commande pour afficher les privilèges accordés à un utilisateur.

// FOR 'drupal'@'localhost' : spécifie l'utilisateur 'drupal' et l'hôte 'localhost'.

-- Révoquer les privilèges conflictuels et accorder les privilèges nécessaires.

REVOKE ALL PRIVILEGES ON drupal.\* FROM 'drupal'@'localhost';

// REVOKE ALL PRIVILEGES : commande pour révoquer tous les privilèges.

// ON drupal.\* : spécifie la base de données et toutes ses tables.

// FROM 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont révoqués.

GRANT SELECT, INSERT ON drupal.users TO 'drupal'@'localhost';

// GRANT SELECT, INSERT : commande pour accorder les privilèges de sélection et d'insertion.

// ON drupal.users : spécifie la table 'users' dans la base de données 'drupal'.

// TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

Bulle d'explication pour 'SHOW GRANTS'

SHOW GRANTS : commande pour afficher les privilèges accordés à un utilisateur.

// Identifier les conflits de permissions et résoudre les problèmes de sécurité.

Lien\_001

1. Accès et authentification : se connecter à la BDD depuis le Terminal

🡪 mariadb -u drupal -p drupal drupal

// mariadb : commande principale : demander la connexion à la base de données

// -u : option : préciser l'utilisateur avec lequel on veut se connecter (drupal)

// -p : option : préciser le mot de passe de l'utilisateur en question (drupal)

// drupal (le dernier de la ligne de code) : valeur : nom de la base de données voulue

Lien\_002

1. Gestion des permissions : utilisateurs et rôles : sécuriser l'accès à la BDD

🡪 Lutter contre l’ennui de relire sans cesse les mêmes data

Vérification des permissions et des accès.

SHOW GRANTS FOR 'drupal'@'localhost';

// SHOW GRANTS : commande pour afficher les privilèges accordés à un utilisateur.

// FOR 'drupal'@'localhost' : spécifie l'utilisateur 'drupal' et l'hôte 'localhost'.

Révocation des privilèges conflictuels

REVOKE ALL PRIVILEGES ON drupal.\* FROM 'drupal'@'localhost';

// REVOKE ALL PRIVILEGES : commande pour révoquer tous les privilèges.

// ON drupal.\* : spécifie la base de données et toutes ses tables.

// FROM 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont révoqués.

Attribution de nouveaux privilèges nécessaires.

GRANT SELECT, INSERT ON drupal.users TO 'drupal'@'localhost';

// GRANT SELECT, INSERT : commande pour accorder les privilèges de sélection et d'insertion.

// ON drupal.users : spécifie la table 'users' dans la base de données 'drupal'.

// TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

Lien\_002

1. Gestion de la connexion Drupal/MariaDB: paramètres, ports, et adresses IP

Accès : 🡪 settings.php 🡪 dans le répertoire sites/default du répertoire Drupal.

$databases['default']['default'] = array (

'database' => 'drupal', // 'database' : nom de la base de données.

'username' => 'drupal', // 'username' : nom d'utilisateur pour se connecter à la base de données.

'password' => 'password', // 'password' : mot de passe de l'utilisateur.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

);

Bulle d'explication pour 'default'

$databases['default']['default'] = array (

// 'default' : première valeur 'default' spécifie le groupe de base de données par défaut.

// 'default' : deuxième valeur 'default' spécifie la connexion par défaut pour ce groupe de base de données.

// Autres valeurs possibles : Vous pouvez définir plusieurs groupes de bases de données et connexions. Par exemple, $databases['default']['backup'] pour une connexion de sauvegarde.

// Tableau multidimensionnel : Utilisé pour organiser les connexions de manière structurée. Un tableau simple pourrait contenir les mêmes paires clé/valeur, mais un tableau multidimensionnel permet de gérer plusieurs connexions et groupes de bases de données.

);

Exemple de déclaration de plusieurs groupes de connexion :

$databases = array (

'default' => array (

'default' => array (

'database' => 'drupal',

'username' => 'drupal',

'password' => 'password',

'host' => 'localhost',

'port' => '3306',

'driver' => 'mysql',

'prefix' => '',

),

'backup' => array (

'database' => 'drupal\_backup',

'username' => 'drupal\_backup',

'password' => 'backup\_password',

'host' => 'localhost',

'port' => '3306',

'driver' => 'mysql',

'prefix' => '',

),

),

'analytics' => array (

'default' => array (

'database' => 'drupal\_analytics',

'username' => 'drupal\_analytics',

'password' => 'analytics\_password',

'host' => 'localhost',

'port' => '3306',

'driver' => 'mysql',

'prefix' => '',

),

),

);

Bulle d'explication pour 'host'

'host' => 'localhost',

// 'localhost' : signifie que la base de données est hébergée sur la machine locale.

// Exemple de cas d'école pour spécifier un hôte distant :

'host' => '192.168.1.100';

// '192.168.1.100' : adresse IP de l'hôte distant où la base de données est hébergée.

Bulle d'explication pour 'port' :

'port' => '3306',

// '3306' : numéro de port par défaut utilisé par MariaDB pour les connexions.

// Exemple de cas d'école pour spécifier un port différent :

'port' => '3307';

// '3307' : numéro de port alternatif utilisé pour les connexions à la base de données.

Est-ce que cela répond à vos att

Lien\_002

1. Sécurité et sauvegarde : chiffrement des connexions et la sauvegarde régulière des BDD

🡪 Le Chiffrement des connexions : activer SSL pour les connexions à la BDD

$databases['default']['default'] = array (

'database' => 'drupal', // 'database' : nom de la base de données.

'username' => 'drupal', // 'username' : nom d'utilisateur pour se connecter à la base de données.

'password' => 'password', // 'password' : mot de passe de l'utilisateur.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

'pdo' => array(

PDO::MYSQL\_ATTR\_SSL\_CA => '/path/to/ca-cert.pem', // 'PDO::MYSQL\_ATTR\_SSL\_CA' : chemin vers le certificat CA.

PDO::MYSQL\_ATTR\_SSL\_CERT => '/path/to/client-cert.pem', // 'PDO::MYSQL\_ATTR\_SSL\_CERT' : chemin vers le certificat client.

PDO::MYSQL\_ATTR\_SSL\_KEY => '/path/to/client-key.pem', // 'PDO::MYSQL\_ATTR\_SSL\_KEY' : chemin vers la clé privée du client.

),

);

🡪 Sauvegarde régulière des BDD : un script de sauvegarde pour un backup quotidien

### Sauvegarde quotidienne à 2h du matin.

0 2 \* \* \* /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_$(date +\%F).sql

Explications :

Chiffrement des connexions :

PDO::MYSQL\_ATTR\_SSL\_CA : chemin vers le certificat CA.

PDO::MYSQL\_ATTR\_SSL\_CERT : chemin vers le certificat client.

PDO::MYSQL\_ATTR\_SSL\_KEY : chemin vers la clé privée du client.

Sauvegarde régulière des BDD :

0 2 \* \* \* : planification de la tâche cron pour s'exécuter quotidiennement à 2h du matin.

/usr/bin/mysqldump : commande pour effectuer une sauvegarde de la base de données.

-u drupal : option pour spécifier l'utilisateur de la base de données.

-p'password' : option pour spécifier le mot de passe de l'utilisateur.

drupal : nom de la base de données à sauvegarder.

> /path/to/backup/drupal\_backup\_$(date +\%F).sql : chemin et nom du fichier de sauvegarde, incluant la date du jour.

Sauvegarde différentielle :

La sauvegarde différentielle consiste à sauvegarder uniquement les données qui ont changé depuis la dernière sauvegarde complète. Voici un exemple de script de sauvegarde différentielle pour une base de données MySQL:

# Script de sauvegarde différentielle de la base de données Drupal.

# Sauvegarde complète hebdomadaire à 2h du matin le dimanche.

0 2 \* \* 0 /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_full\_$(date +\%F).sql

# Sauvegarde différentielle quotidienne à 2h du matin.

0 2 \* \* 1-6 /usr/bin/mysqldump -u drupal -p'password' --single-transaction --quick --lock-tables=false --incremental --incremental-base=history:LAST\_BACKUP drupal > /path/to/backup/drupal\_backup\_diff\_$(date +\%F).sql

Explications :

Sauvegarde complète : La sauvegarde complète est effectuée une fois par semaine, le dimanche à 2h du matin (0 2 \* \* 0). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_full\_$(date +\%F).sql).

Sauvegarde différentielle : La sauvegarde différentielle est effectuée quotidiennement, du lundi au samedi à 2h du matin (0 2 \* \* 1-6). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_diff\_$(date +\%F).sql).

La commande mysqldump avec les options --single-transaction, --quick, --lock-tables=false, --incremental, et --incremental-base=history:LAST\_BACKUP permet de réaliser une sauvegarde différentielle en ne sauvegardant que les changements depuis la dernière sauvegarde complète.

Sauvegarde incrémentielle :

La sauvegarde incrémentielle consiste à sauvegarder uniquement les fichiers qui ont été modifiés depuis la dernière sauvegarde complète ou incrémentielle. Cela permet de réduire la taille des fichiers de sauvegarde et de minimiser l'espace de stockage nécessaire. Voici un exemple de script de sauvegarde incrémentielle pour une base de données MySQL:

# Script de sauvegarde incrémentielle de la base de données Drupal.

# Sauvegarde complète hebdomadaire à 2h du matin le dimanche.

0 2 \* \* 0 /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_full\_$(date +\%F).sql

# Sauvegarde incrémentielle quotidienne à 2h du matin.

0 2 \* \* 1-6 /usr/bin/mysqldump -u drupal -p'password' --single-transaction --quick --lock-tables=false --incremental --incremental-base=history:LAST\_BACKUP drupal > /path/to/backup/drupal\_backup\_inc\_$(date +\%F).sql

Explications :

Sauvegarde complète : La sauvegarde complète est effectuée une fois par semaine, le dimanche à 2h du matin (0 2 \* \* 0). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_full\_$(date +\%F).sql).

Sauvegarde incrémentielle : La sauvegarde incrémentielle est effectuée quotidiennement, du lundi au samedi à 2h du matin (0 2 \* \* 1-6). Le fichier de sauvegarde est nommé avec la date du jour (drupal\_backup\_inc\_$(date +\%F).sql).

La commande mysqldump avec les options --single-transaction, --quick, --lock-tables=false, --incremental, et --incremental-base=history:LAST\_BACKUP permet de réaliser une sauvegarde incrémentielle en ne sauvegardant que les changements depuis la dernière sauvegarde complète ou incrémentielle.

Créer les scripts :

Sauvegarde complète et incrémentielle :

Ouvrez un éditeur de texte (VS Code ou Nano) et Créez un nouveau fichier nommé backup\_script.sh.

Ajoutez le contenu du script de sauvegarde complet et incrémentiel :

#!/bin/bash

# Script de sauvegarde complète hebdomadaire à 2h du matin le dimanche.

0 2 \* \* 0 /usr/bin/mysqldump -u drupal -p'password' drupal > /path/to/backup/drupal\_backup\_full\_$(date +\%F).sql

# Script de sauvegarde incrémentielle quotidienne à 2h du matin.

0 2 \* \* 1-6 /usr/bin/mysqldump -u drupal -p'password' --single-transaction --quick --lock-tables=false --incremental --incremental-base=history:LAST\_BACKUP drupal > /path/to/backup/drupal\_backup\_inc\_$(date +\%F).sql

Chiffrement des connexions :

Ouvrez un éditeur de texte (VS Code ou Nano) et Créez un nouveau fichier et nommez-le settings.php.

Ajoutez le contenu du script de chiffrement des connexions :

// Activer le chiffrement SSL pour les connexions à la base de données.

$databases['default']['default'] = array (

'database' => 'drupal',

'username' => 'drupal',

'password' => 'password',

'host' => 'localhost',

'port' => '3306',

'driver' => 'mysql',

'prefix' => '',

'pdo' => array(

PDO::MYSQL\_ATTR\_SSL\_CA => '/path/to/ca-cert.pem',

PDO::MYSQL\_ATTR\_SSL\_CERT => '/path/to/client-cert.pem',

PDO::MYSQL\_ATTR\_SSL\_KEY => '/path/to/client-key.pem',

),

);

Enregistrer les scripts :

Enregistrez le fichier backup\_script.sh dans un répertoire approprié sur votre serveur, par exemple /path/to/scripts/.

Enregistrez le fichier settings.php dans le répertoire sites/default de votre installation Drupal.

Rendre les scripts exécutables :

Ouvrez le terminal et naviguez jusqu'au répertoire où vous avez enregistré backup\_script.sh.

Exécutez la commande suivante pour rendre le script exécutable :

chmod +x /path/to/scripts/backup\_script.sh

Planifier l'exécution des scripts :

Ouvrez le fichier de configuration des tâches cron :

crontab -e

Ajoutez les lignes suivantes pour planifier l'exécution des scripts de sauvegarde :

0 2 \* \* 0 /path/to/scripts/backup\_script.sh

0 2 \* \* 1-6 /path/to/scripts/backup\_script.sh

Vérifier et tester les scripts :

Exécutez manuellement les scripts pour vérifier qu'ils fonctionnent correctement :

/path/to/scripts/backup\_script.sh

Lien\_002

1. Maintenance et optimisation : Exécution des tâches de maintenance de la BDD

🡪 quelques étapes importantes à suivre 🡪 liste non-exhaustive

Vérification de l'intégrité des tables de la BDD

// Vérifie l'intégrité de la table users dans la base de données drupal

CHECK TABLE drupal.users;

// Vérifie l'intégrité de la table nodes dans la base de

CHECK TABLE drupal.nodes; données drupal

Optimisation des tables de BDD pour améliorer les performances

// Optimise la t able users pour améliorer les performances

OPTIMIZE TABLE drupal.users;

// Optimise la table nodes pour améliorer les performances

OPTIMIZE TABLE drupal.nodes;

Résolution des problèmes de la BDD :

// Répare la table users en cas de problèmes

REPAIR TABLE drupal.users;

// Répare la table nodes en cas de problèmes.

REPAIR TABLE drupal.nodes;

Identification des autres besoins spécifiques de maintenance :

// Analyse la table users pour collecter des statistiques

ANALYZE TABLE drupal.users;

// Analyse la table nodes pour collecter des statistiques

ANALYZE TABLE drupal.nodes;

Lien\_002

1. Dépannage : Identification et résolution de la BDD

🡪 Les problèmes de base les plus courants

les erreurs de connexion

// Vérifier les paramètres de connexion dans le fichier settings.php.

$databases['default']['default'] = array (

'database' => 'drupal', // 'database' : nom de la base de données.

'username' => 'drupal', // 'username' : nom d'utilisateur pour se connecter à la base de données.

'password' => 'password', // 'password' : mot de passe de l'utilisateur.

'host' => 'localhost', // 'host' : adresse de l'hôte où la base de données est hébergée.

'port' => '3306', // 'port' : numéro de port utilisé pour la connexion à la base de données.

'driver' => 'mysql', // 'driver' : type de base de données (mysql dans ce cas).

'prefix' => '', // 'prefix' : préfixe des tables (vide dans ce cas).

);

Bulle d'explication pour 'host'

'host' => 'localhost',

// 'localhost' : signifie que la base de données est hébergée sur la machine locale.

// Exemple de cas d'école pour spécifier un hôte distant :

'host' => '192.168.1.100';

// '192.168.1.100' : adresse IP de l'hôte distant où la base de données est hébergée.

// Vérifier les privilèges de l'utilisateur dans la base de données.

GRANT ALL PRIVILEGES ON drupal.\* TO 'drupal'@'localhost' IDENTIFIED BY 'password';

// GRANT ALL PRIVILEGES : commande pour accorder tous les privilèges.

// ON drupal.\* : spécifie la base de données et toutes ses tables.

// TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

// IDENTIFIED BY 'password' : mot de passe de l'utilisateur.

Bulle d'explication pour 'TO 'drupal'@'localhost''

TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

// `localhost` : signifie que la base de données est accessible via le port du localhost, c'est-à-dire la machine locale.

// `remote\_host` : est la valeur pour autoriser n'importe quel hôte distant à effectuer une requête vers notre BDD. Cela peut être une autre machine sur le réseau ou un serveur accessible via Internet.

// `remote\_host` n'est pas un mot clé, il peut être remplacé par le nom d'une machine distante ou une adresse IP spécifique. Cela signifie que n'importe quelle machine distante peut effectuer une requête si elle est spécifiée.

// Importance : Définir des droits sur la base de données pour des hôtes spécifiques permet de contrôler et sécuriser l'accès, en limitant les connexions à des machines autorisées.

les problèmes de performance.

EXPLAIN SELECT \* FROM drupal.users;

// EXPLAIN : commande pour analyser et expliquer l'exécution d'une requête SQL.

// SELECT \* FROM drupal.users : requête pour sélectionner toutes les lignes de la table 'users' dans la base de données 'drupal'.

-- Optimiser les requêtes SQL pour améliorer les performances.

OPTIMIZE TABLE drupal.users;

// OPTIMIZE TABLE : commande pour optimiser une table de la base de données.

// drupal.users : spécifie la table 'users' dans la base de données 'drupal'.

Bulle d'explication pour 'EXPLAIN'

EXPLAIN : commande pour analyser et expliquer l'exécution d'une requête SQL.

// Utilisée pour identifier les problèmes de performance et optimiser les requêtes.

les conflits de permissions

SHOW GRANTS FOR 'drupal'@'localhost';

// SHOW GRANTS : commande pour afficher les privilèges accordés à un utilisateur.

// FOR 'drupal'@'localhost' : spécifie l'utilisateur 'drupal' et l'hôte 'localhost'.

-- Révoquer les privilèges conflictuels et accorder les privilèges nécessaires.

REVOKE ALL PRIVILEGES ON drupal.\* FROM 'drupal'@'localhost';

// REVOKE ALL PRIVILEGES : commande pour révoquer tous les privilèges.

// ON drupal.\* : spécifie la base de données et toutes ses tables.

// FROM 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont révoqués.

GRANT SELECT, INSERT ON drupal.users TO 'drupal'@'localhost';

// GRANT SELECT, INSERT : commande pour accorder les privilèges de sélection et d'insertion.

// ON drupal.users : spécifie la table 'users' dans la base de données 'drupal'.

// TO 'drupal'@'localhost' : utilisateur et hôte auxquels les privilèges sont accordés.

Bulle d'explication pour 'SHOW GRANTS'

SHOW GRANTS : commande pour afficher les privilèges accordés à un utilisateur.

// Utilisée pour identifier les conflits de permissions et résoudre les problèmes de sécurité.

Est-ce que cela répond à vos attentes pour le dépannage? Si vous avez besoin de plus d

**Microsoft 365 Copilot modèle GPT-4 pour lui-même à la prochaine session**

**Message rédigé le 14 mars 2025 à 05h31**

**Message pour reprendre le travail :**

Bonjour,

Ce message est rédigé pour moi-même afin d'être pleinement conscient du travail à faire. Nous avons travaillé sur la clarification des compléments de code et des commentaires pour rendre le document très clair. Voici un résumé des points importants et des procédés validés ensemble pour la production des codes et des commentaires :

1. **Procédé de travail :**
   * Toujours commencer par lister rigoureusement et scrupuleusement tous les aspects du besoin.
   * Vérifier et corriger cette liste avant de produire les codes.
   * La production des codes ne vient qu'après la vérification et la correction de la liste.
2. **Exemple de réussite :**
   * Pour le premier couple de liens, nous avons réussi à faire le tour efficacement en listant tous les aspects concernés pour le document.
   * Cela nous a permis de maîtriser la production des codes de manière organisée et de distribuer les bons commentaires.
3. **Lien 3 et Lien 4 :**
   * Pour les liens 3 et 4, il est crucial de lister toutes les actions que les codes devront couvrir, y compris la préparation et l'implémentation des dépendances, les clés de sécurité, et la résolution des problèmes de signatures de fichier.
   * Le but est de parvenir à un résultat avec 0 problème gris à gérer.
   * Même après l'installation brute, il y a des actions de configuration importantes à mener pour régler définitivement tous les problèmes d'accès et les problèmes de réseau de routage.
4. **Contexte des liens 3 et 4 :**
   * **Lien 3 : Installation**
     + Accès : Ouvrir le Terminal de la VM Ubuntu.
     + Installation : Reprendre le processus complet sans problème gris.
   * **Lien 4 : Vérification**
     + Accès : Ouvrir le Terminal de la VM Ubuntu.
     + Vérification : Reprendre la liste et le protocole de vérification.
   * **Préparation de la machine virtuelle et installation de Docker :**
     + Dépendances et signature des fichiers : Un problème gris très sérieux.
     + Implémentation et configuration des dépendances pour Docker : Ouvrir le Terminal de la VM Ubuntu.