Documentation Complète de la Configuration du Serveur de base de données

A. Installation et Configuration de Docker

- Mise à Jour des Paquets Debian avec les commandes

sudo apt update

sudo apt upgrade

- Installation de Docker sur Debian :

sudo apt install docker-ce

- Vérification de l'Installation de Docker :

docker --version

sudo systemctl status docker

B. Mise en Place du Serveur MySQL dans Docker

- Téléchargement de l'Image MySQL :

docker pull mysql

- Création et Lancement du Conteneur MySQL :

docker run --name mysql-container -e MYSQL_ROOT_PASSWORD=mon_mot_de_passe -p 3306:3306 -d mysql

- Configuration de Redémarrage Automatique du Conteneur :

docker update --restart always mysql-container

C. Configuration Initiale de MySQL

- Connexion au Conteneur MySQL :

docker exec -it mysql-container bash

- Connexion à MySQL :

mysql -u root -p

D. Création de la Base de Données et des Tables

- Exécution du Script SQL pour la Base de Données e_commerce :

- Copie du script SQL dans le conteneur :

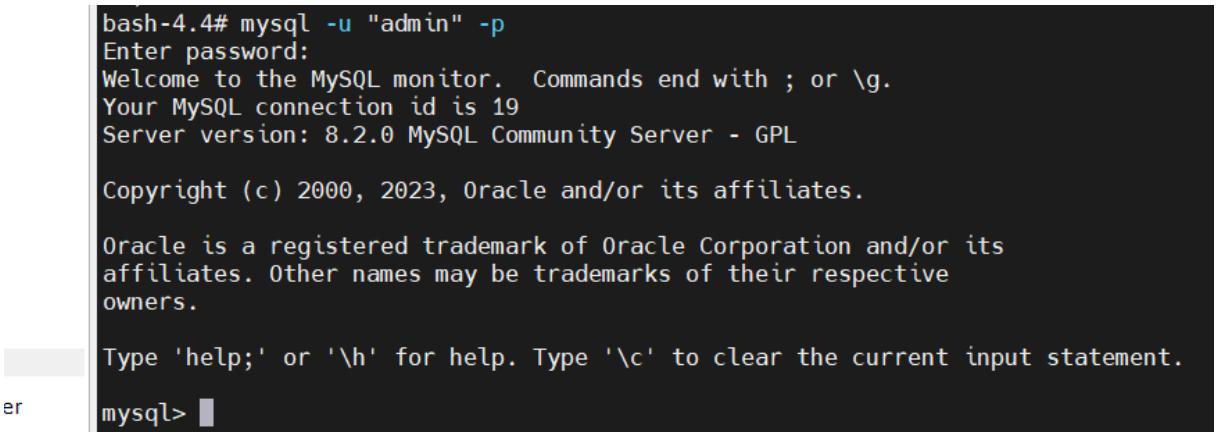
docker cp /chemin/vers/script.sql mysql-container:/chemin/dans/conteneur

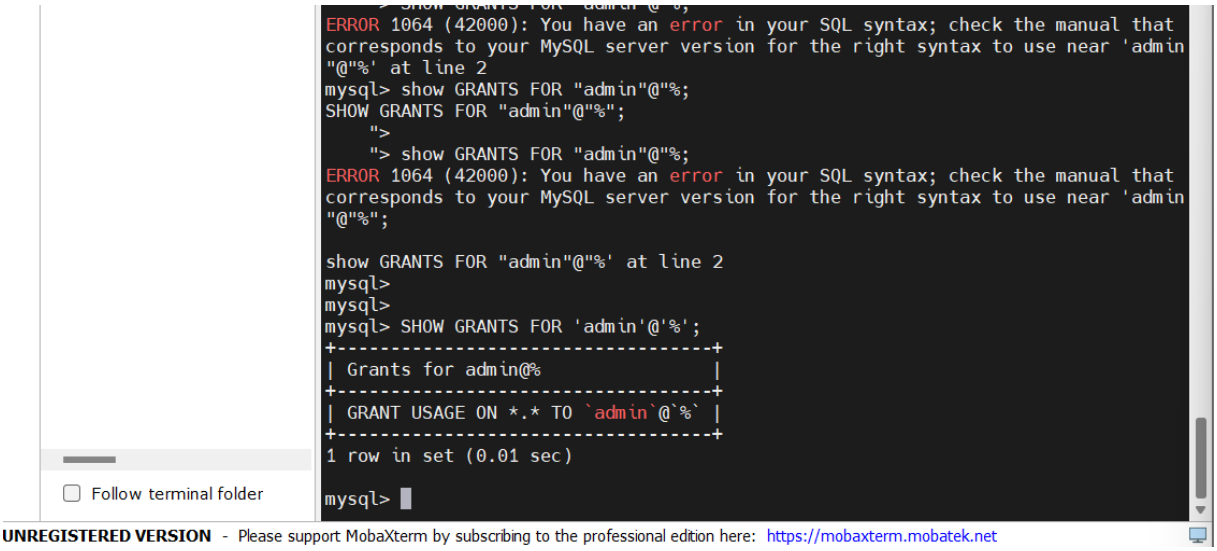
- Exécution du script dans MySQL :

source /chemin/dans/conteneur/script.sql

E. Gestion des Utilisateurs et des Privilèges(contrôles ACL) :

- Création de l'Utilisateur admin et Attribution de Privilèges :

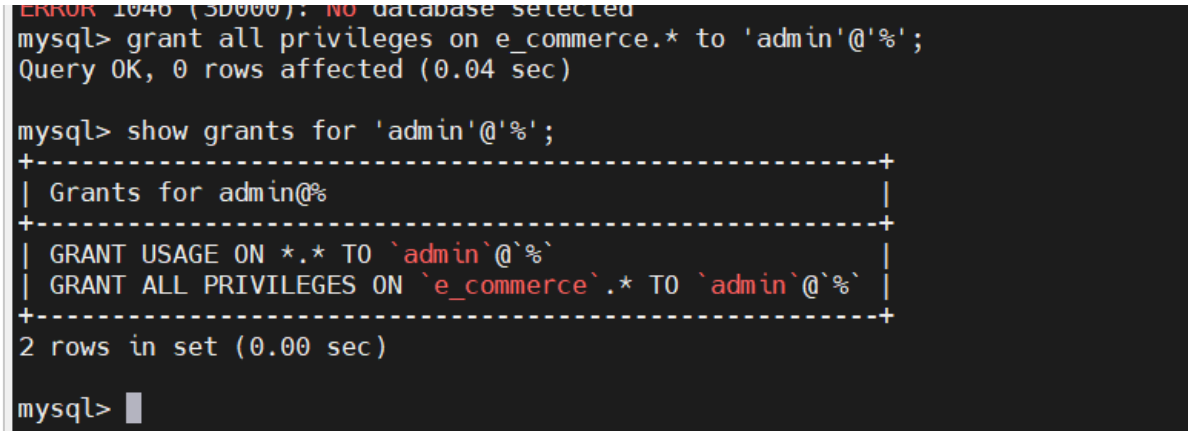




CREATE USER 'admin'@'%' IDENTIFIED BY 'password';

GRANT ALL PRIVILEGES ON e_commerce.* TO 'admin'@'%';

FLUSH PRIVILEGES;



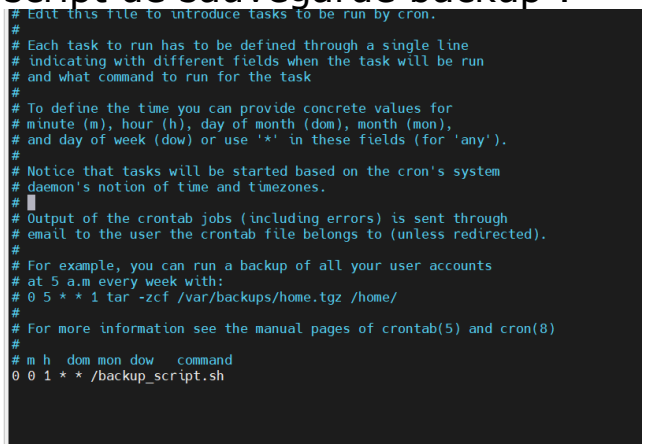
F. Mise en Place des Sauvegardes Automatisées

- Création d'un Script de Sauvegarde avec mysqldump :

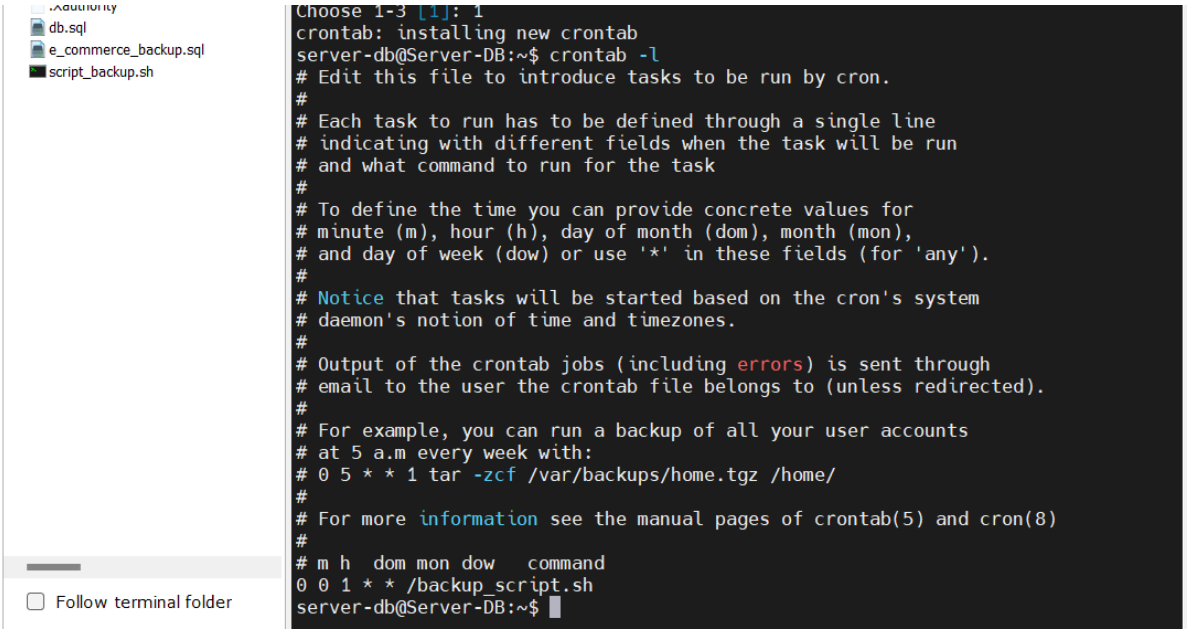
Contenu du script :

```
#!/bin/bash

# Paramètres de connexion MySQL
DB_CONTAINER_NAME="mysql-container"
DB_USERNAME="admin"
DB_PASSWORD="ynovparis"
DB_NAME="e_commerce"

# Emplacement et nom de fichier pour la sauvegarde
BACKUP_PATH="e_commerce_backup"
BACKUP_FILENAME="backup_$DB_NAME_$(date
+%Y%m%d_%H%M%S).sql"

# Commande de sauvegarde
docker exec $DB_CONTAINER_NAME /usr/bin/mysqldump -u
$DB_USERNAME --password=$DB_PASSWORD $DB_NAME >
$BACKUP_PATH/$BACKUP_FILENAME

# Afficher un message
echo "La sauvegarde de la base de données a été effectuée avec succès :
$BACKUP_PATH/$BACKUP_FILENAME"
```

- Mise en place d'une tâche **cron** pour exécuter automatiquement à minuit le premier jour de chaque mois le script de sauvegarde backup :



`0 0 1 * * /chemin/vers/backup_script.sh`


.Xauthority
db.sql
e_commerce_backup.sql
script_backup.sh
Follow terminal folder
Choose 1-3 [1]: 1
crontab: installing new crontab
server-db@Server-DB:~$ crontab -l
Edit this file to introduce tasks to be run by cron.
#
Each task to run has to be defined through a single line
indicating with different fields when the task will be run
and what command to run for the task
#
To define the time you can provide concrete values for
minute (m), hour (h), day of month (dom), month (mon),
and day of week (dow) or use '*' in these fields (for 'any').
#
Notice that tasks will be started based on the cron's system
daemon's notion of time and timezones.
#
Output of the crontab jobs (including errors) is sent through
email to the user the crontab file belongs to (unless redirected).
#
For example, you can run a backup of all your user accounts
at 5 a.m every week with:
0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
For more information see the manual pages of crontab(5) and cron(8)
#
m h dom mon dow   command
0 0 1 * * /backup_script.sh
server-db@Server-DB:~$