Section 4: Databases

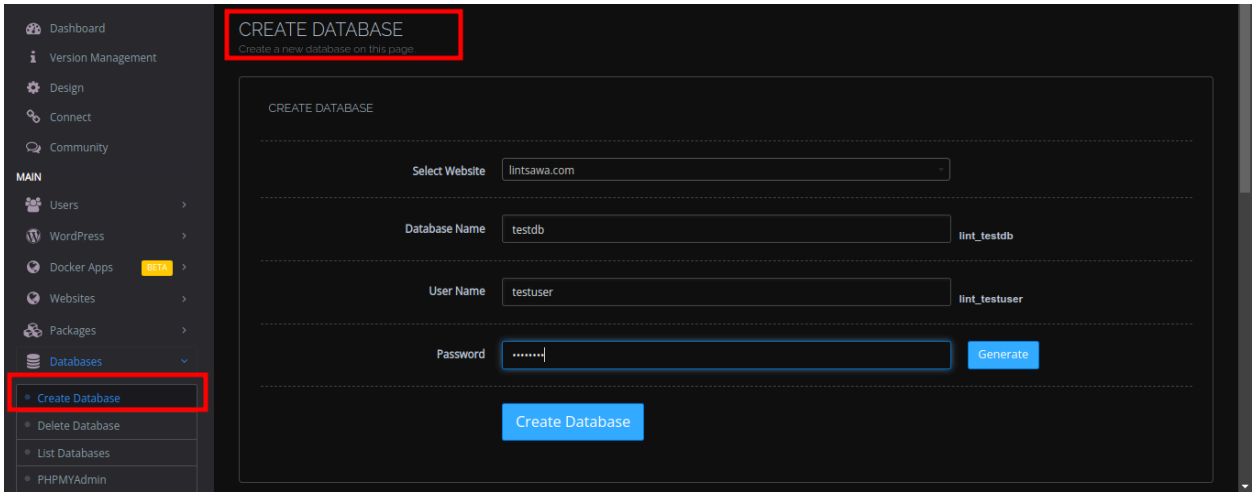
Managing databases is a crucial part of server administration.

---

1. Creating New Databases

Creating databases is a common task when setting up new applications or websites. CyberPanel simplifies this process through its user interface.

1. **Login to CyberPanel:**
    - Navigate to your CyberPanel dashboard.
2. **Go to Database Management:**
    - From the dashboard, go to **Databases > Create Database**.
3. **Enter Database Details:**
    - **Database Name**: Enter a name for your database.
    - **Database User**: Create a new user or select an existing one.
    - **Password**: Set a strong password for the database user.
    - **Select Website**: Choose the website to associate the database with
    - **Click Create Database:**
    - Once all fields are filled in, click the **Create Database** button.
4. **Verify Creation:**
    - After successful creation, you can verify the database by going to **Databases > List Databases**.



---

2. Exporting/Importing Databases

You might need to export or import databases for migrations, backups, or restoring from backups. This can be done through CyberPanel or via the terminal.

A. Exporting a Database:

1. **Login to CyberPanel:**
   - From the dashboard, go to **Databases**
   - Click on **phpMyAdmin** next to it.
2. **Export Database via phpMyAdmin:**
   - Inside phpMyAdmin, select the database and click **Export**.
   - Choose the **Quick** export method and format as .
   - Click **Go**, and the SQL file will be downloaded to your local machine.

B. Importing a Database:

1. **Login to phpMyAdmin via CyberPanel:**
   - Similar to the export process, go to **Databases > List Databases** and select **phpMyAdmin**.

2. **Select the Database:**
    - Choose the database where you want to import the data (or create a new database if necessary).
3. **Import the SQL File:**
    - Click on **Import** in phpMyAdmin.
    - Select the SQL file from your local machine and click **Go** to start the import process.

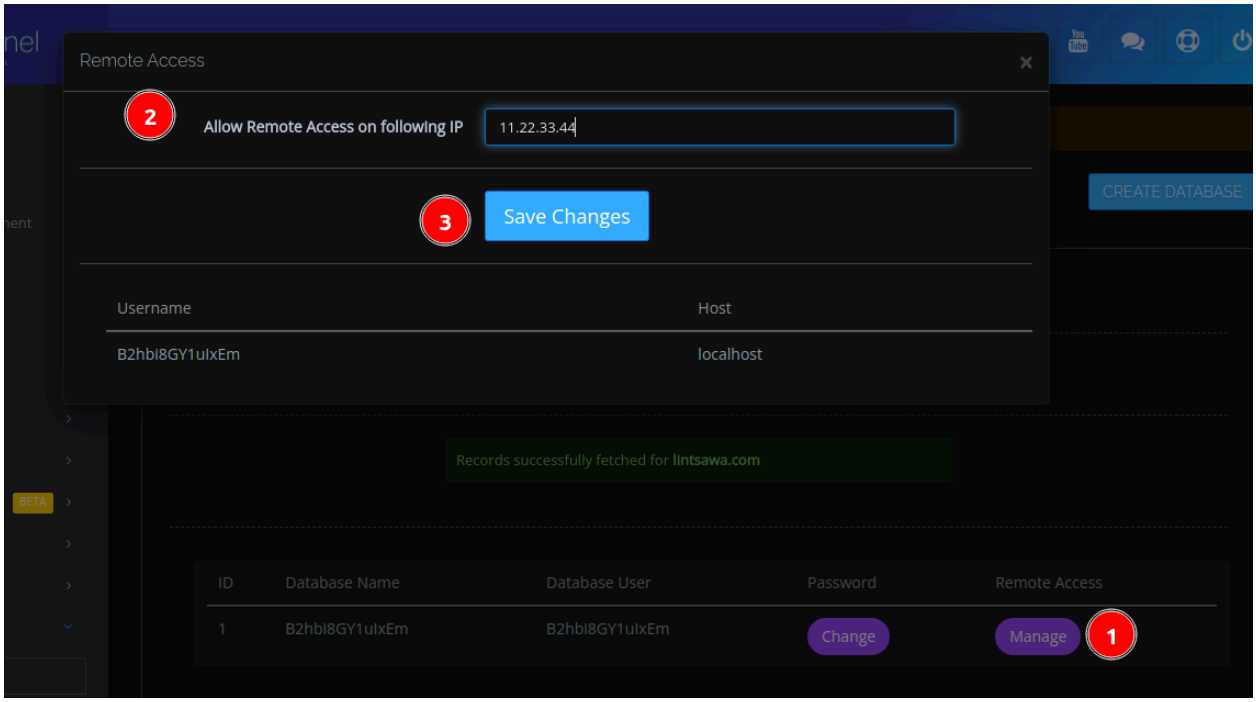
Use Case: Database export/import is often used during site migrations or when backing up and restoring databases for maintenance purposes.

---

3. Enabling Remote Database Access

Remote database access allows you to connect to your server's databases from an external machine. This can be useful for managing databases using external tools like MySQL Workbench, Dbeaver or for applications that require access to a central database.

1. **Login to CyberPanel:**
    - From the dashboard, go to **Databases > List Databases - Manage Access**
2. **Add Remote IP Address:**
    - In the **Remote MySQL** section, add the IP address of the machine you want to allow access to the database.
3. **Enable Remote Access:**
    - Once you've added the IP address, the server will allow connections from this machine to the databases.



4. **Configure MySQL to Listen on All Interfaces (Optional):**

By default, MySQL may only listen on localhost (127.0.0.1). To allow remote access, you need to modify the MySQL configuration file.

sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf

-

Look for the following line and comment it out or change it to:

bind-address = 0.0.0.0

-

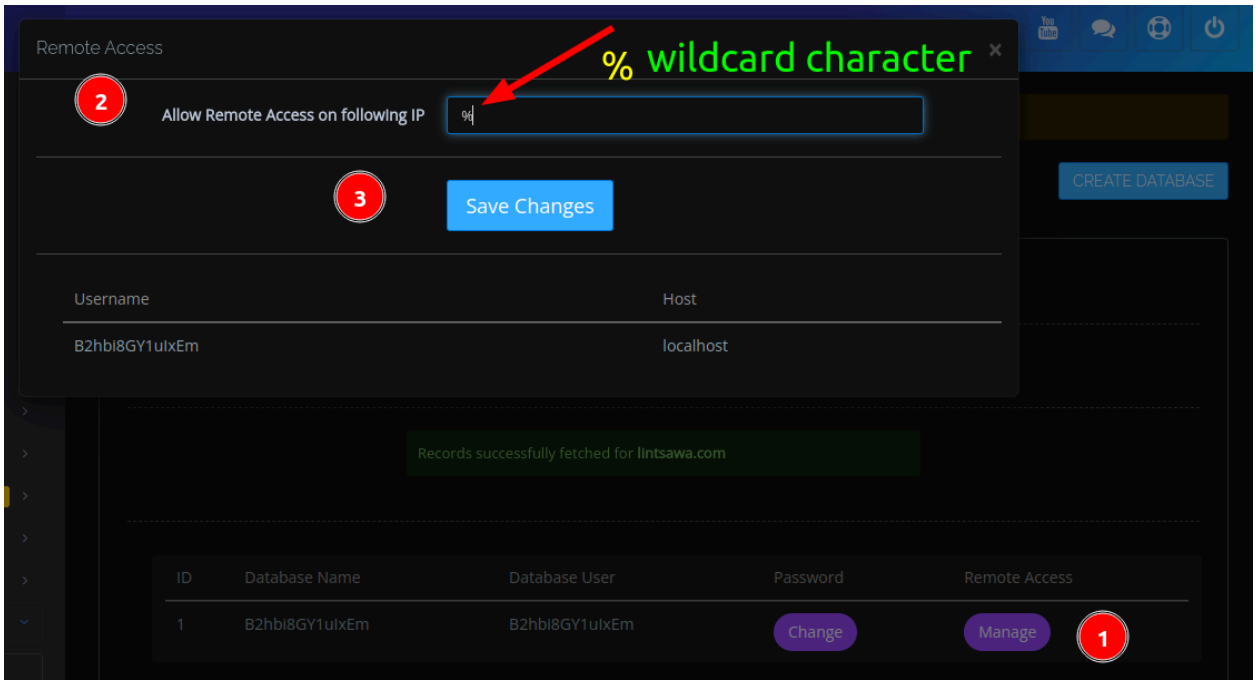
5. **Restart MySQL:**

Restart MySQL for changes to take effect:

sudo systemctl restart mariadb

Then From the cyberpanel dashboard, go to **Databases > List Databases - Manage Access**

- In the **Remote MySQL** section, add the wildcard character **(%)** to allow access to the database from all IP addresses.



Use Case: Remote database access is used when you need to connect from a development machine or when multiple applications hosted on different servers share the same database.

---

4. Upgrading the Database (MariaDB)

Regularly updating MariaDB ensures security, stability, and access to the latest features. Here's how to upgrade MariaDB.

1. **Backup Databases:**

Before upgrading, it's crucial to backup all databases:

```
mysqldump --all-databases > /home/db-backup-$(date +%F).sql
```

2. **Check Current MariaDB Version:**

To check the current version of MariaDB, use the following command:

```
mysql -V
```

3. **Stop MariaDB Service:**

Stop MariaDB before upgrading:

```
systemctl stop mariadb
```

4. **Add MariaDB Repository:**

Add the repository for the newer MariaDB version. For example, to upgrade to MariaDB 10.6:

```
curl -LsS
https://r.mariadb.com/downloads/mariadb_repo_setup |
sudo bash -s -- --mariadb-server-version="mariadb-10.6"
```

5. **Upgrade MariaDB:**

After adding the repository, upgrade MariaDB:

apt install mariadb-server

   -

6. **Restart MariaDB and Verify:**

Restart MariaDB and check the version:

systemctl start mariadb

For a full guide, see the documentation below.

https://truehost.com/support/knowledge-base/how-to-upgrade-mariadb-on-cyberpanel-from-10-3-to-10-6-on-ubuntu-20-04/

Use Case: Upgrading ensures that your database system is up-to-date with the latest performance improvements and security patches.

---

5. Any Other Database-Related Actions

- **Managing Database Users and Permissions:**
  - You can create, delete, and modify database users and set specific permissions for each database. This can be done via CyberPanel under **Databases > Create Database** by assigning specific users to the databases.
- **Backup and Restore:**

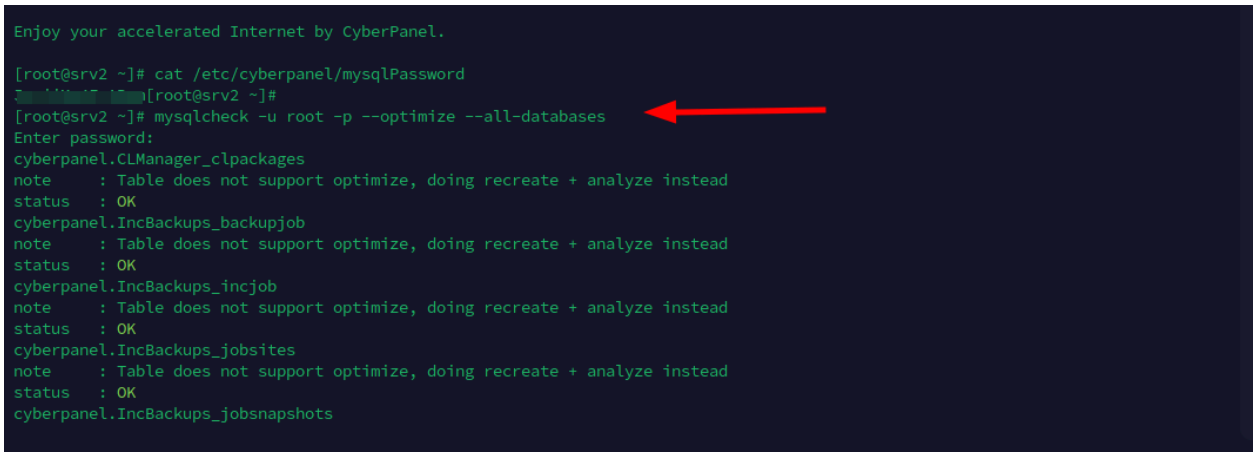
Regular backups are crucial for disaster recovery. Apart from manual exports through phpMyAdmin, you can automate backups using cron jobs to regularly export databases:

mysqldump --all-databases > /path/to/backup/db-backup-$(date +%F).sql

- **Repairing and Optimizing Databases:**

You can use the **mysqlcheck** command to check, repair, and optimize your databases:

mysqlcheck -u root -p --optimize --all-databases



- **Modifying Database Configuration:**
  - If needed, you can tweak the database configuration in the MySQL configuration file

(**etc/mysql/mariadb.conf.d/50-server.cnf)** to adjust settings like buffer sizes and cache limits for performance optimization.