MySQL Automation - Switch between 2 cluster sets

Overview

This Python script is designed to manage MySQL database servers in a clustered environment. It checks the status of servers, connects to them, and handles promotion and demotion based on server availability. It performs failover and failback operations for two database clusters.

Dependencies

- `subprocess`: To execute shell commands (e.g., `ping`).
- `time`: For sleep intervals between checks.
- `socket`: To resolve IP addresses to hostnames.
- `mysql.connector`: To connect to MySQL servers.
- `os`: For creating directories and handling file paths.

User Inputs

- **Production Database Server IP**: IP address of the primary production database server.
- **High Availability Database IP**: IP address of the high availability (HA) server.
- **Near DR Database IP**: IP address of the near disaster recovery (DR) server.
- **DR Database IP**: IP address of the disaster recovery (DR) server.

Variables

- `database_cluster_1`: List of IPs for the production, high availability, and near DR servers.
- `database_cluster_2`: List containing the DR server IP.
- `check_interval`: Time interval (in seconds) between checks.

Functions

`setup_log_file()`

- **Purpose**: Creates a log directory and file if they do not already exist.
- **Returns**: Path to the log file.

`log_message(log_file, message)`

- **Purpose**: Writes a log entry to the log file with a timestamp.
- **Parameters**:
  - `log_file` (string): Path to the log file.
  - `message` (string): Message to log.
- **Returns**: None.

`get_hostname_ip(ip_address)`

- **Purpose**: Retrieves the hostname for a given IP address.
- **Parameters**:
  - `ip_address` (string): IP address to resolve.
- **Returns**: Hostname or error message.

`connect_mysql_production_database_server(host, port, user, password)`

- **Purpose**: Connects to the MySQL production server and logs the connection status.
- **Parameters**:
  - `host` (string): Hostname or IP address of the MySQL server.
  - `port` (int): Port number of the MySQL server.
  - `user` (string): Username for MySQL authentication.
  - `password` (string): Password for MySQL authentication.
- **Returns**: None. Prints connection status to the screen and logs to the file.

`connect_mysql_near_dr_database_server(host, port, user, password)`

- **Purpose**: Connects to the MySQL near DR server and logs the connection status.
- **Parameters**:
  - `host` (string): Hostname or IP address of the MySQL server.
  - `port` (int): Port number of the MySQL server.
  - `user` (string): Username for MySQL authentication.
  - `password` (string): Password for MySQL authentication.
- **Returns**: None. Prints connection status to the screen and logs to the file.

`connect_mysql_dr_database_server(host, port, user, password)`

- **Purpose**: Connects to the MySQL DR server and logs the connection status.
- **Parameters**:
  - `host` (string): Hostname or IP address of the MySQL server.
  - `port` (int): Port number of the MySQL server.
  - `user` (string): Username for MySQL authentication.
  - `password` (string): Password for MySQL authentication.
- **Returns**: None. Prints connection status to the screen and logs to the file.

`ping_database_cluster_1(server)`

- **Purpose**: Pings servers in cluster 1 to check their availability and logs the result.
- **Parameters**:
  - `server` (string): IP address of the server to ping.
- **Returns**: Success status and output of the ping command.

`ping_database_cluster_2(server)`

- **Purpose**: Pings servers in cluster 2 to check their availability and logs the result.
- **Parameters**:
  - `server` (string): IP address of the server to ping.
- **Returns**: Success status and output of the ping command.

`promotion_commands(connection)`

- **Purpose**: Executes commands to promote a server to primary.
- **Parameters**:
  - `connection` (MySQL connection object): Connection to the MySQL server.
- **Returns**: None. Executes SQL commands to modify server role.

`demotion_commands(connection)`

- **Purpose**: Executes commands to demote a server to secondary.
- **Parameters**:
  - `connection` (MySQL connection object): Connection to the MySQL server.
- **Returns**: None. Executes SQL commands to modify server role.

`failover_to_secondary_production_database_server()`

- **Purpose**: Demotes the production database server to secondary.
- **Returns**: None.

`failover_to_secondary_near_dr_database_server()`

- **Purpose**: Demotes the near DR database server to secondary.
- **Returns**: None.

`failover_to_secondary_dr_database_server()`

- **Purpose**: Demotes the DR database server to secondary.
- **Returns**: None.

`failback_to_primary_production_database_server()`

- **Purpose**: Promotes the production database server to primary.
- **Returns**: None.

`failback_to_primary_near_dr_database_server()`

- **Purpose**: Promotes the near DR database server to primary.
- **Returns**: None.

`failback_to_primary_dr_database_server()`

- **Purpose**: Promotes the DR database server to primary.
- **Returns**: None.

`main_1()`

- **Purpose**: Monitors and manages Database Cluster 1. If all servers are down, it demotes the production and near DR servers and promotes the DR server. If any server is up, it performs related actions.
- **Returns**: None.

`main_2()`

- **Purpose**: Monitors and manages Database Cluster 2. If all servers are down, it demotes the DR server and promotes production and near DR servers. If any server is up, it performs related actions.
- **Returns**: None.

Execution Loop

- The script continuously runs `main_1()` and `main_2()` in a loop, with a sleep interval defined by `check_interval`.
