**10 Real-Time Corporate Shell Scripts**

**🗂️ 1. Backup Script**

**✅ Script:**

#!/bin/bash

SOURCE="/home/ubuntu/aditya"

DESTINATION="/home/ubuntu/jaiswal/"

DATE=$(date +%Y-%m-%d\_%H-%M-%S)

mkdir -p $DESTINATION/$DATE

cp -r $SOURCE $DESTINATION/$DATE

echo "Backup completed on $DATE"

**🧠 Explanation:**

* SOURCE: Path to the directory you want to back up.
* DESTINATION: Where the backup should be stored.
* DATE: Captures the current time to create a timestamped backup folder.
* mkdir -p: Ensures the destination directory exists.
* cp -r: Recursively copies all files and folders.
* echo: Confirms the backup success with a timestamp.

**🕒 Automation with Cron:**

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

(Runs every minute. Modify for daily/hourly schedules.)

**💽 2. Disk Usage Check Script**

**✅ Script:**

#!/bin/bash

THRESHOLD=80

df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 }' |

while read output; do

usage=$(echo $output | awk '{ print $1}' | cut -d'%' -f1)

partition=$(echo $output | awk '{ print $2 }')

if [ $usage -ge $THRESHOLD ]; then

echo "Warning: Disk usage on $partition is at ${usage}%"

fi

done

**🧠 Explanation:**

* df -H: Shows disk usage in human-readable form.
* Filters out irrelevant lines (tmpfs, cdrom, etc.).
* Extracts %usage and device name.
* Checks if usage is above 80%.
* Sends a warning if threshold is crossed.

**🛠️ 3. Service Health Check Script**

**✅ Script:**

#!/bin/bash

SERVICE="nginx"

if systemctl is-active --quiet $SERVICE; then

echo "$SERVICE is running"

else

echo "$SERVICE is not running"

systemctl start $SERVICE

fi

**🧠 Explanation:**

* Checks if a service like nginx, mysql, etc., is running.
* If not, restarts the service automatically.
* Great for production uptime scripts.

**🌐 4. Network Connectivity Script**

**✅ Script:**

#!/bin/bash

HOST="google.com"

OUTPUT\_FILE="/home/ubuntu/output.txt"

if ping -c 1 $HOST &> /dev/null; then

echo "$HOST is reachable" >> $OUTPUT\_FILE

else

echo "$HOST is not reachable" >> $OUTPUT\_FILE

fi

**🧠 Explanation:**

* Checks if a host (e.g., Google) is reachable via ping.
* Redirects results to an output file.
* Can be used in alert systems.

**🛢️ 5. MySQL Database Backup Script**

**✅ Setup:**

sudo apt install mysql-server

**Login and set password:**

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'root';

FLUSH PRIVILEGES;

**✅ Script:**

#!/bin/bash

DB\_NAME="mydatabase"

BACKUP\_DIR="/path/to/backup"

DATE=$(date +%Y-%m-%d\_%H-%M-%S)

mysqldump -u root -p $DB\_NAME > $BACKUP\_DIR/$DB\_NAME-$DATE.sql

echo "Database backup completed: $BACKUP\_DIR/$DB\_NAME-$DATE.sql"

**🧠 Explanation:**

* Uses mysqldump to export the database to an SQL file.
* Timestamp helps in maintaining versioned backups.
* Add to cron to automate nightly backups.

**⏱️ 6. System Uptime Script**

**✅ Script:**

#!/bin/bash

uptime -p

**🧠 Explanation:**

* Displays how long the system has been running.
* Useful for tracking reboots and server reliability.

**📡 7. Listening Ports Script**

**✅ Setup:**

sudo apt install net-tools

**✅ Script:**

#!/bin/bash

netstat -tuln | grep LISTEN

**🧠 Explanation:**

* Shows active TCP/UDP ports and services.
* Helpful in debugging network issues or service status.

**♻️ 8. Automatic Package Updates Script**

**✅ Script:**

#!/bin/bash

apt-get update && apt-get upgrade -y && apt-get autoremove -y && apt-get clean

echo "System packages updated and cleaned up"

**🧠 Explanation:**

* Updates the system, removes unused packages, and clears cache.
* Ensures your server stays up-to-date and lightweight.

**🌐 9. HTTP Response Time Script**

**✅ Script:**

#!/bin/bash

URLS=("https://www.devopsshack.com/" "https://www.linkedin.com/")

for URL in "${URLS[@]}"; do

RESPONSE\_TIME=$(curl -o /dev/null -s -w '%{time\_total}\n' $URL)

echo "Response time for $URL: $RESPONSE\_TIME seconds"

done

**🧠 Explanation:**

* Measures how long it takes to load a website using curl.
* Ideal for performance monitoring and uptime checks.

**🧠 10. Top Memory-Consuming Processes**

**✅ Script:**

#!/bin/bash

ps aux --sort=-%mem | head -n 10

**🧠 Explanation:**

* Lists all processes sorted by memory usage.
* Top 10 most resource-hungry processes.
* Great for identifying performance bottlenecks.