mssql with Aws and Drive

Created	@August 29, 2025 3:30 PM
Class	DevOps
Last edited time	@August 29, 2025 3:46 PM

📘 MSSQL Backup Automation – Documentation

This repository contains scripts to **automate Microsoft SQL Server backups** and upload them to **Google Drive** or **AWS S3**.

🚀 Features

- Full and Differential MSSQL backups.
- Automatic compression into `.zip`.
- Upload to:
  - Google Drive (via `rclone` + Google Cloud Service Account)
  - AWS S3 (optional).
- Logs for every backup operation.
- Supports **multiple databases**.

🛠️ Prerequisites

1. **SQL Server Tools**

   Install `sqlcmd`:

   ```
   sudo apt-get install mssql-tools unixodbc-dev -y
   ```

2. **Compression Tool**

```
sudo apt-get install zip -y
```

3. **Cloud Tools**
   - rclone for Google Drive.
   - AWS CLI if using S3.
4. **Permissions**

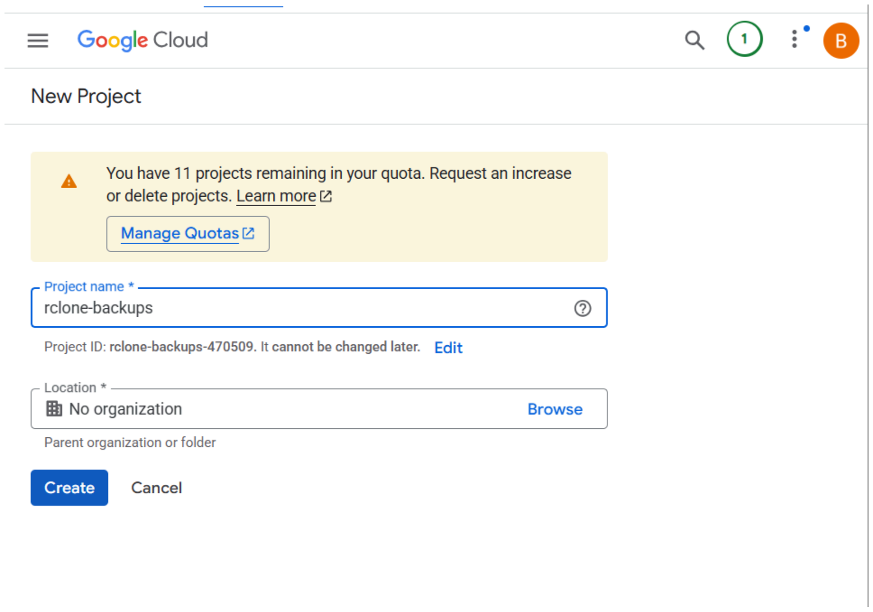
   Ensure `/var/opt/mssql/backups` is writable by `mssql` user:

```
sudo mkdir -p /var/opt/mssql/backups/{fullbackup,differentialbackup}
sudo chown -R mssql:mssql /var/opt/mssql/backups
sudo chmod -R 755 /var/opt/mssql/backups
```

⚙️ Google Cloud Service Account Setup

1. **Go to Google Cloud Console** → Google Cloud Console
   - Create a new project (or use an existing one).
2. **Enable Google Drive API**
   - Navigate to **APIs & Services > Library**.
   - Search for **Google Drive API** → Click **Enable**.
3. **Create a Service Account**
   - Go to **IAM & Admin > Service Accounts**.
   - Click **Create Service Account**.
   - Give it a name (e.g., `mssql-backup-service`).
   - Assign role → **Project > Editor**.
4. **Generate Credentials (JSON file)**

- Under your service account, go to **Keys > Add Key > Create New Key**.
- Choose **JSON** → download it.
- Save in your VM, e.g.



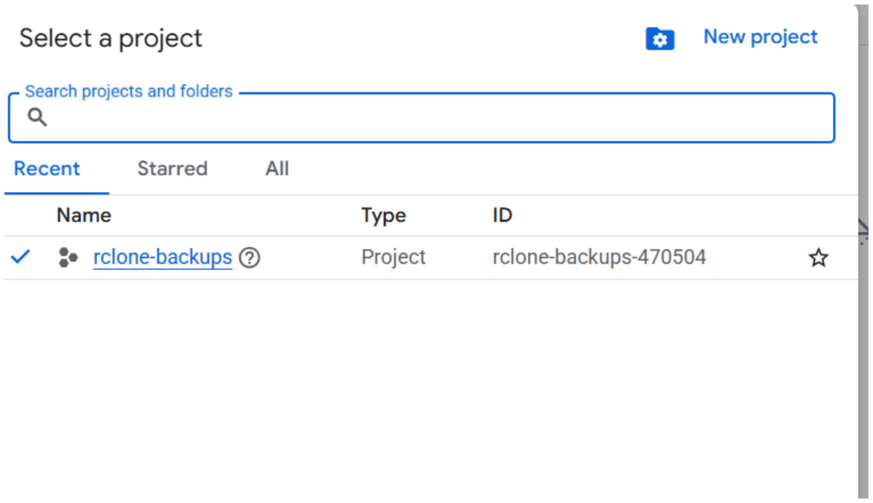
You have 11 projects remaining in your quota. Request an increase or delete projects. Learn more

Project name *
rclone-backups

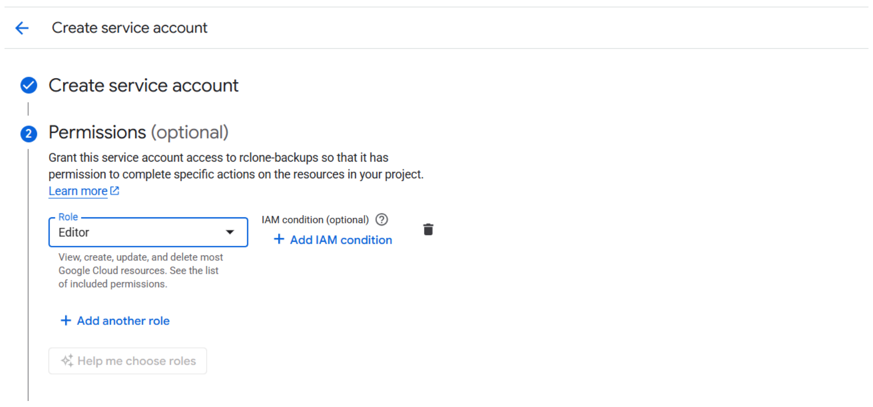
Project ID: rclone-backups-470509. It cannot be changed later. Edit

Location *
No organization

Parent organization or folder



Name	Type	ID
rclone-backups	Project	rclone-backups-470504


Create service account

Create service account

Permissions (optional)

Grant this service account access to rclone-backups so that it has permission to complete specific actions on the resources in your project.

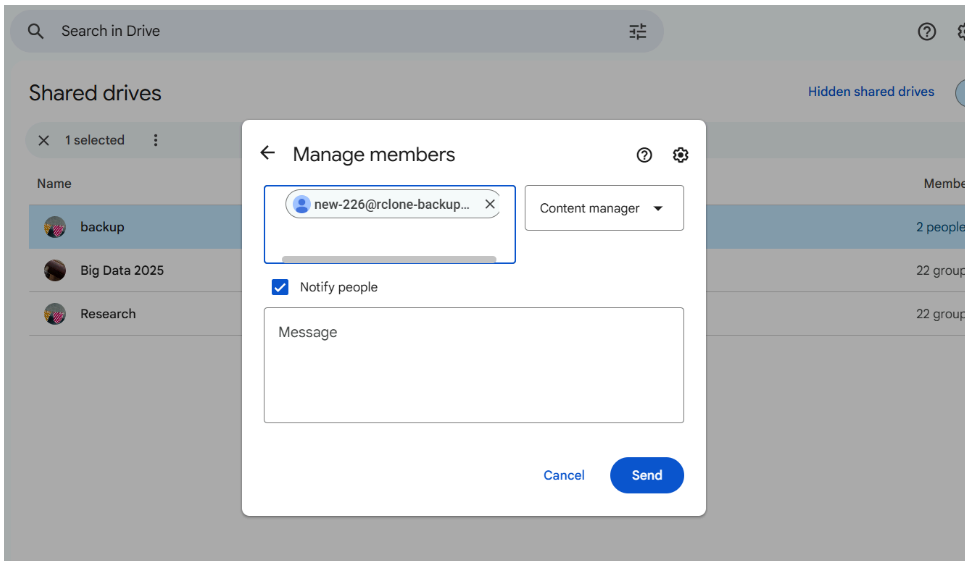
Role: Editor

View, create, update, and delete most Google Cloud resources. See the list of included permissions.

```
/home/username/.config/rclone/mssql-service.json
```

5. **Share Google Drive Folder with Service Account**

- Create a folder in your Google Drive (e.g., `mssql-backups`).
- Right-click → **Share**.
- Enter the service account email (looks like `service-account-name@project-id.iam.gserviceaccount.com`).
- Give **Editor** permission.


Manage members

new-226@rclone-backup...

Content manager

✓ Notify people

Name	Members
backup	2 people
Big Data 2025	22 group
Research	22 group

🔌 Configure rclone for Google Drive

Run:

```
rclone config
```

Steps:

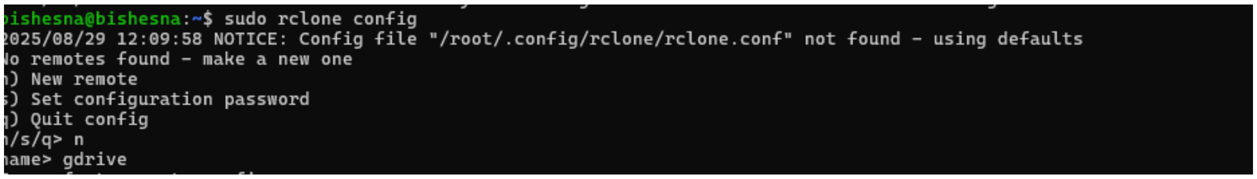
1. `n` → New remote
2. Name → `gdrive`
3. Storage type → `drive`
4. Choose `service_account_file` option
5. Enter path to your service account JSON:

   ```
   /home/username/.config/rclone/mssql-service.json
   ```

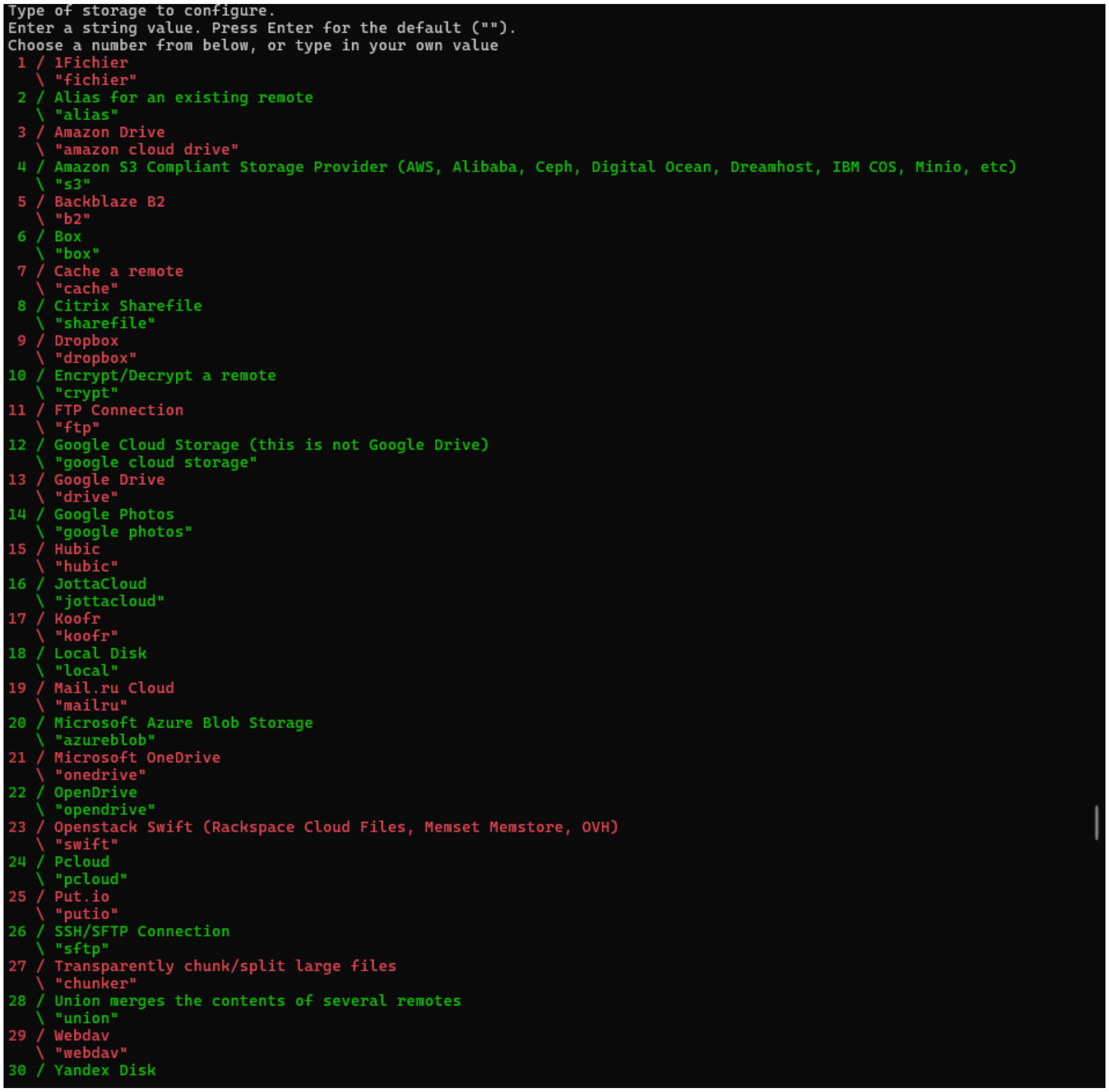
6. Save & quit

Verify setup:

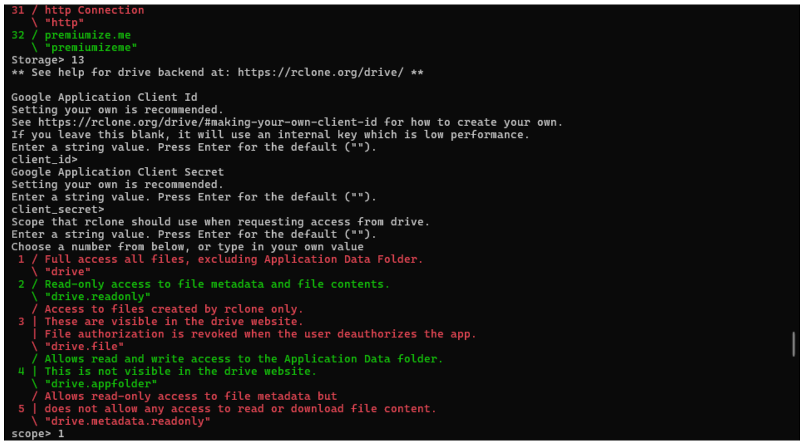
```
rclone ls gdrive:
```



```
ishesna@bishesna:~$ sudo rclone config
025/08/29 12:09:58 NOTICE: Config file "/root/.config/rclone/rclone.conf" not found - using defaults
o remotes found - make a new one
) New remote
) Set configuration password
) Quit config
/s/q> n
ame> gdrive
```

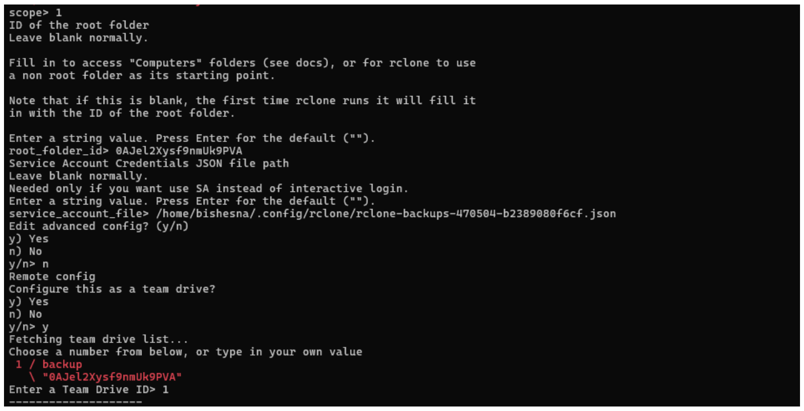


```
Type of storage to configure.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value
 1 / 1Fichier
   \ "fichier"
 2 / Alias for an existing remote
   \ "alias"
 3 / Amazon Drive
   \ "amazon cloud drive"
 4 / Amazon S3 Compliant Storage Provider (AWS, Alibaba, Ceph, Digital Ocean, Dreamhost, IBM COS, Minio, etc)
   \ "s3"
 5 / Backblaze B2
   \ "b2"
 6 / Box
   \ "box"
 7 / Cache a remote
   \ "cache"
 8 / Citrix Sharefile
   \ "sharefile"
 9 / Dropbox
   \ "dropbox"
10 / Encrypt/Decrypt a remote
   \ "crypt"
11 / FTP Connection
   \ "ftp"
12 / Google Cloud Storage (this is not Google Drive)
   \ "google cloud storage"
13 / Google Drive
   \ "drive"
14 / Google Photos
   \ "google photos"
15 / Hubic
   \ "hubic"
16 / JottaCloud
   \ "jottacloud"
17 / Koofr
   \ "koofr"
18 / Local Disk
   \ "local"
19 / Mail.ru Cloud
   \ "mailru"
20 / Microsoft Azure Blob Storage
   \ "azureblob"
21 / Microsoft OneDrive
   \ "onedrive"
22 / OpenDrive
   \ "opendrive"
23 / Openstack Swift (Rackspace Cloud Files, Memset Memstore, OVH)
   \ "swift"
24 / Pcloud
   \ "pcloud"
25 / Put.io
   \ "putio"
26 / SSH/SFTP Connection
   \ "sftp"
27 / Transparently chunk/split large files
   \ "chunker"
28 / Union merges the contents of several remotes
   \ "union"
29 / Webdav
   \ "webdav"
30 / Yandex Disk
```



```
31 / http Connection
   \ "http"
32 / premiumize.me
   \ "premiumizeme"
Storage> 13
** See help for drive backend at: https://rclone.org/drive/ **

Google Application Client Id
Setting your own is recommended.
See https://rclone.org/drive/#making-your-own-client-id for how to create your own.
If you leave this blank, it will use an internal key which is low performance.
Enter a string value. Press Enter for the default ("").
client_id>
Google Application Client Secret
Setting your own is recommended.
Enter a string value. Press Enter for the default ("").
client_secret>
Scope that rclone should use when requesting access from drive.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value
 1 / Full access all files, excluding Application Data Folder.
   \ "drive"
 2 / Read-only access to file metadata and file contents.
   \ "drive.readonly"
   / Access to files created by rclone only.
 3 | These are visible in the drive website.
   | File authorization is revoked when the user deauthorizes the app.
   \ "drive.file"
   / Allows read and write access to the Application Data folder.
 4 | This is not visible in the drive website.
   \ "drive.appfolder"
   / Allows read-only access to file metadata but
 5 | does not allow any access to read or download file content.
   \ "drive.metadata.readonly"
scope> 1
```

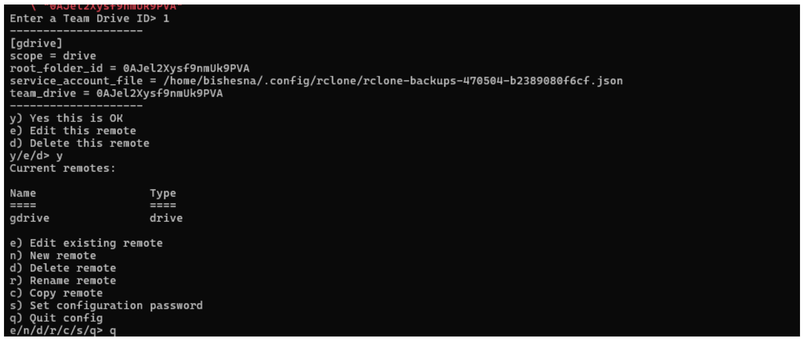


```
scope> 1
ID of the root folder
Leave blank normally.

Fill in to access "Computers" folders (see docs), or for rclone to use
a non root folder as its starting point.

Note that if this is blank, the first time rclone runs it will fill it
in with the ID of the root folder.

Enter a string value. Press Enter for the default ("").
root_folder_id> 0AJel2Xysf9nmUk9PVA
Service Account Credentials JSON file path
Leave blank normally.
Needed only if you want use SA instead of interactive login.
Enter a string value. Press Enter for the default ("").
service_account_file> /home/bishesna/.config/rclone/rclone-backups-470504-b2389080f6cf.json
Edit advanced config? (y/n)
y) Yes
n) No
y/n> n
Remote config
Configure this as a team drive?
y) Yes
n) No
y/n> y
Fetching team drive list...
Choose a number from below, or type in your own value
 1 / backup
   \ "0AJel2Xysf9nmUk9PVA"
Enter a Team Drive ID> 1
--------------------
```



```
Enter a Team Drive ID> 1
--------------------
[gdrive]
scope = drive
root_folder_id = 0AJel2Xysf9nmUk9PVA
service_account_file = /home/bishesna/.config/rclone/rclone-backups-470504-b2389080f6cf.json
team_drive = 0AJel2Xysf9nmUk9PVA
--------------------
y) Yes this is OK
e) Edit this remote
d) Delete this remote
y/e/d> y
Current remotes:

Name                 Type
====                 ====
gdrive               drive

e) Edit existing remote
n) New remote
d) Delete remote
r) Rename remote
c) Copy remote
s) Set configuration password
q) Quit config
e/n/d/r/c/s/q> q
```

🖥️ Script Configuration

Edit `mssql_aws_drive4.sh`:

- **SA password**

```
SA_PASSWORD="YourStrongSAPasswordHere"
```

- **Databases**

```
DATABASES=("DB_NAME_1" "DB_NAME_2")
```

- **Google Drive remote name** (same as `rclone config`):

```
GDRIVE_REMOTE="gdrive"
```

- **AWS S3 path** (optional):

```
AWS_PATH="s3://mybucket/mssql-backups/"
```

▶️ Usage

Full Backup (default)

```
./mssql_aws_drive4.sh
```

Differential Backup

```
./mssql_aws_drive4.sh differential
```

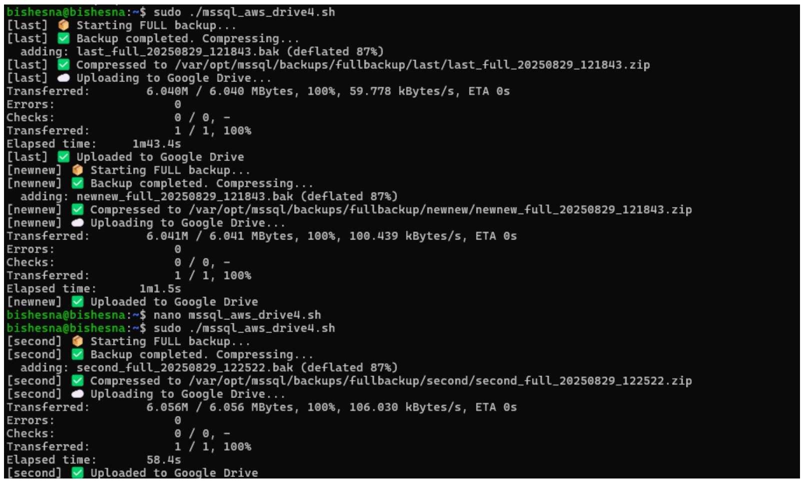
Backup Specific Databases

```
./mssql_aws_drive4.sh full MyDatabase1 MyDatabase2
```

📦 Example Output

```
[MyDatabase1] 📦 Starting FULL backup...
[MyDatabase1] ✅ Backup completed. Compressing...
[MyDatabase1] ✅ Compressed to /var/opt/mssql/backups/fullbackup/MyDatabase1/MyDatabase1_full_20250829_121843.zip
```

```
[MyDatabase1] ☁️ Uploading to Google Drive...
[MyDatabase1] ✅ Uploaded to Google Drive
```



```
bishesna@bishesna:~$ sudo ./mssql_aws_drive4.sh
[last] 📦 Starting FULL backup...
[last] ✅ Backup completed. Compressing...
  adding: last_full_20250829_121843.bak (deflated 87%)
[last] ✅ Compressed to /var/opt/mssql/backups/fullbackup/last/last_full_20250829_121843.zip
[last] ☁️ Uploading to Google Drive...
Transferred:        6.040M / 6.040 MBytes, 100%, 59.778 kBytes/s, ETA 0s
Errors:                 0
Checks:                 0 / 0, -
Transferred:            1 / 1, 100%
Elapsed time:     1m43.4s
[last] ✅ Uploaded to Google Drive
[newnew] 📦 Starting FULL backup...
[newnew] ✅ Backup completed. Compressing...
  adding: newnew_full_20250829_121843.bak (deflated 87%)
[newnew] ✅ Compressed to /var/opt/mssql/backups/fullbackup/newnew/newnew_full_20250829_121843.zip
[newnew] ☁️ Uploading to Google Drive...
Transferred:        6.041M / 6.041 MBytes, 100%, 100.439 kBytes/s, ETA 0s
Errors:                 0
Checks:                 0 / 0, -
Transferred:            1 / 1, 100%
Elapsed time:      1m1.5s
[newnew] ✅ Uploaded to Google Drive
bishesna@bishesna:~$ nano mssql_aws_drive4.sh
bishesna@bishesna:~$ sudo ./mssql_aws_drive4.sh
[second] 📦 Starting FULL backup...
[second] ✅ Backup completed. Compressing...
  adding: second_full_20250829_122522.bak (deflated 87%)
[second] ✅ Compressed to /var/opt/mssql/backups/fullbackup/second/second_full_20250829_122522.zip
[second] ☁️ Uploading to Google Drive...
Transferred:        6.056M / 6.056 MBytes, 100%, 106.030 kBytes/s, ETA 0s
Errors:                 0
Checks:                 0 / 0, -
Transferred:            1 / 1, 100%
Elapsed time:       58.4s
[second] ✅ Uploaded to Google Drive
```

🔒 Security Notes

- Do not hardcode passwords in production.
- Store `SA_PASSWORD` in an environment variable:

  ```
  export SA_PASSWORD="YourStrongPassword"
  ```

  Then update the script:

  ```
  SA_PASSWORD="${SA_PASSWORD:?Missing SA password}"
  ```

- Add these to `.gitignore`:
  - `.json` (service account file)
  - `.config/rclone/`
  - Backup `.zip` files

🌟 Future Improvements

- Incremental backups.
- Centralized logging system.
- Notifications on failure (Slack/Email).

✅ With this setup:

- Your SQL backups are automatically stored locally.
- They are compressed to save space.
- They are securely uploaded to Google Drive using a **service account**.
- Optionally, they can also be pushed to AWS S3.