Contents

1 Introduction

We talk about backup System, we see a different backup strategy, and we talk about Grandfather–Father–Son backup strategy. we use tools like rsync, tar, and cron to implement backup strategy. and we dont forget linux distribution like ubuntu, debian, and centos. in this tutorial, we se manual backup and automatic backup, and we talk about advantages and disadvantages of each strategy.

2 Backup Strategy

2.1 Manual Backup

Because we are using tools like Docker, we have the container and volume. we just run the command below to backup the volume. just create a backups directory in the host machine, and run the command below to backup the volume.

```
docker exec postgres_db pg_dump -U <username> -d <database> \
 > backup.sql
```

2.1.1 Restore

```
cat backup.sql | docker exec -i postgres_db \
psql -U <username> -d <database>
```

and thi is the simple way to backup and restore the database, but we need to do it manually, and we need to remember to do it regularly. and dont forget to compress the backup file to save space, and encryption with gpg to secure the backup file.

2.2 Automatic Backup

2.2.1 Host cron job (pro production)

we can use a script to backup the database, and we can use cron to schedule the script to run regularly. the file name is backup.sh, and the content is below. you can specifie a time to run the backup, for example, every day at 2 am.

```
CONTAINER=container_name
DB_USER=username
DB_NAME=database_name
BACKUP_ROOT="$HOME/backups"

TIMESTAMP=$(date +"%Y-%m-%d_%H-%M")
BACKUP_DIR="$BACKUP_ROOT/$TIMESTAMP"
```

```
mkdir -p "$BACKUP_DIR"

/usr/bin/docker exec -i $CONTAINER \
pg_dump -U $DB_USER $DB_NAME \
| gzip > "$BACKUP_DIR/backup.sql.gz"

echo "Backup created: $BACKUP_DIR/backup.sql.gz"
```

dont forget make the script executable with

```
chmod +x backup.sh
```

now we can add cron job to run the script , because we have deffrents mode in cron we some jobs done every day or week or month in specified time, for example, we want to run the backup every day at 2 am, ather with dellay like every 2 hours or every 30 minutes. for the simullation we test with dellay .
this called **Interval scheduling**

2.2.2 Interval scheduling

we open the cron file with the command below

```
crontab -e
```

and add role to run the backup script every 3 minutes

```
*/3 * * * * /full/path/to/backup.sh
```

2.2.3 Time-based scheduling

we can use time-based scheduling to run the backup script at a specific time. for example, we want to run the backup every day at 2 am. we can add the following line to the cron file

```
0 2 * * * /full/path/to/backup.sh
```

2.2.4 Restore from compressed backup

if we have a compressed backup file, we can restore it with the following command

```
gunzip -c backup.sql.gz | docker exec -i postgres_db \
psql -U <username> -d <database>
```

but those are the simple way to backup and restore the database, waste of space and time . finally you found your self have 1 to 9999 backup files, and you need to manage them, and you need to delete the old backup files to save space, and you need to remember to do it regularly. becuase of this problem we need to use backup strategy to manage the backup files.

2.3 Backup Manager

we use backup manager to manage files , daily backup, weekly backup, and monthly backup, and we can set the retention policy to delete old backup files, and we can set the schedule to run the backup script. we create 3 directories in the host machine, daily, weekly, and monthly, and we set the retention policy to keep 7 daily backup files, 4 weekly backup files, and 12 monthly backup files.

2.4 Grandfather–Father–Son backup strategy

this strategy is a common backup strategy that uses three levels of backups: daily (son), weekly (father), and monthly (grandfather). The idea is to have a hierarchy of backups that allows for both short-term and long-term recovery options.

- Daily (Son): These are the most recent backups, typically created every day.
- Weekly (Father): These are created less frequently, usually once a week, and are kept for a longer period than daily backups.
- Monthly (Grandfather): These are the least frequent backups, created once a month and kept for the longest period, often several months or even years

2.5 Types of this strategy

- Seperated backup: we dont have relation between the backup files, we just create backup files and store them in the directories, and we set the retention policy to delete old backup files.
- Hierarchical time-based backup: backups are organized in a tiered structure (daily, weekly, monthly) and triggered by time schedules rather than direct dependency between backup files. Each backup remains independent while following a hierarchical retention policy.

2.5.1 Single backup pipeline architecture

the ather level of backup files weekly and monthly backup files are created from the daily backup files, and we set the retention policy to delete old backup files, and we can set the schedule to run the backup script. for example, we can set the schedule to run the daily backup script every day at 2 am we use and counter to count the number of daily backup files, and we set the retention policy to keep 4 weekly backup files,

and this code is backup script for the single backup pipeline architecture

```
    #!/bin/bash

CONTAINER=postgres_db_test
DB_USER=admin
DB_NAME=mydb
BACKUP_ROOT="$HOME/backups"

TIMESTAMP=$(date +"%Y-%m-%d_%H-%M-%S")

DAILY_DIR="$BACKUP_ROOT/daily"
WEEKLY_DIR="$BACKUP_ROOT/weekly"
MONTHLY_DIR="$BACKUP_ROOT/monthly"

COUNTER_FILE="$BACKUP_ROOT/counter.txt"

mkdir -p "$DAILY_DIR" "$WEEKLY_DIR" "$MONTHLY_DIR"
```

```
DAILY_FILE="$DAILY_DIR/$TIMESTAMP.sql.gz"

/usr/bin/docker exec -i $CONTAINER \
pg_dump -U $DB_USER $DB_NAME | gzip > "$DAILY_FILE"


COUNT=$(cat "$COUNTER_FILE" 2>/dev/null || echo 0)
COUNT=$((COUNT + 1))
echo "$COUNT" > "$COUNTER_FILE"


if (( COUNT % 7 == 0 )); then
    WEEK_NUM=$((COUNT / 7))
    cp "$DAILY_FILE" "$WEEKLY_DIR/week_$WEEK_NUM.sql.gz"
fi


if (( COUNT % 28 == 0 )); then
    MONTH_NUM=$((COUNT / 28))
    cp "$DAILY_FILE" "$MONTHLY_DIR/month_$MONTH_NUM.sql.gz"
fi

ls -t "$DAILY_DIR" | tail -n +8 |
    xargs -I {} rm -f "$DAILY_DIR/{}" 2>/dev/null
ls -t "$WEEKLY_DIR" | tail -n +5 |
    xargs -I {} rm -f "$WEEKLY_DIR/{}" 2>/dev/null
ls -t "$MONTHLY_DIR" | tail -n +13 |
    xargs -I {} rm -f "$MONTHLY_DIR/{}" 2>/dev/null
```

and just enter the command below to run the backup script

```
./backup.sh
```

for test and make it executable after enter to crontab file with the command below

```
crontab -e
```

and add the following line to run the backup script every day at 2 am

```
0 2 * * * /full/path/to/backup.sh
```

this is the simple way to implement the Grandfather–Father–Son backup strategy, but we need to manage the backup files, and we need to delete old backup files to save space, and we need to remember to do it regularly.