

Back up every database on several PostgreSQL versions

On RHEL, Rocky and Alma, PGDG packages let PostgreSQL 16 and 17 (for example) run side by side, each on its own port with its own binaries in `/usr/pgsql-<ver>/bin`. One script can back up both.

Describe your instances

```
PG_INSTANCES=(  
  "5432|usr/pgsql-16/bin|pg16"  
  "5433|usr/pgsql-17/bin|pg17"  
)
```

Use each instance's own `pg_dump`. A newer `pg_dump` can dump an older server, but not the other way round.

The loop

Run it as the `postgres` user so it connects over the local socket with peer authentication. No password is needed.

```
for inst in "${PG_INSTANCES[@]}"; do  
  IFS='|' read -r port bin name <<<"$inst"  
  out=/mnt/backup/$(hostname -s)/$name; mkdir -p "$out"  
  "$bin/pg_dumpall" -p "$port" --globals-only | gzip > "$out/$(date +%F_%H).globals.sql.gz"  
  while read -r db; do  
    "$bin/pg_dump" -p "$port" "$db" | gzip > "$out/$(date +%F_%H).$db.sql.gz"  
  done < <("$bin/psql" -p "$port" -Atc \  
    "SELECT datname FROM pg_database WHERE NOT datistemplate AND datname <> 'postgres'")  
done  
find /mnt/backup/$(hostname -s) -type f -name '*.sql.gz' -mtime +7 -delete
```

- **Back up the globals.** `pg_dump` doesn't include roles, passwords or grants. Without `pg_dumpall --globals-only`, a restore to a new server has no logins.
- Check that the backup share is mounted (`mountpoint -q`), and that the `postgres` user can write to it.
- Use `flock` to stop runs from overlapping.

Schedule

`/etc/cron.d/backup-postgres`:

```
0 * * * * postgres /usr/local/bin/backup_postgres.sh >/dev/null 2>&1
```

Restore

```
gunzip -c 2026-01-01_03.globals.sql.gz | psql -p 5433 postgres
createdb -p 5433 mydb && gunzip -c 2026-01-01_03.mydb.sql.gz | psql -p 5433 mydb
```

Revision #1

Created 24 October 2026 13:29:32 by Admin

Updated 24 October 2026 13:29:32 by Admin