

PostgreSQL Backup & Recovery Reference

Production guide to pg_dump, pg_basebackup, WAL archiving, pgBackRest, PITR and disaster scenarios

PostgreSQL 15-18

Updated: October 2026 · Production-grade



PG

RPO
How much data? · acceptable loss

RTO
How fast? · time to restore

RULE
3-2-1-1-0 · 3 copies, 2 media, 1 off-site

+1 / +0
immutable · 0 restore-test errors

GOLDEN RULE
An untested backup = no backup

01 Fundamentals & Strategy



CONCEPT	DESCRIPTION
Logical	SQL / object-level dump. Version & platform independent, easy selective restore; slow restore on large data (indexes are rebuilt).
Physical	Block copy of the data files. Fast restore , fits TB scale; requires same major version + platform.
Snapshot	LVM / ZFS / EBS / SAN snapshot. Crash-consistent; safe if WAL/redo is on the same volume or backup mode is used.
Full / Incr / Diff	Everything / changes since last backup / since last full. The longer the incremental chain, the higher restore time and risk .
Hot / Warm / Cold	Reads+writes allowed / reads only / server stopped. Production target: hot .
Continuous Archive	Base backup + continuous WAL stream → PITR : return to any moment. Brings RPO down to seconds.

CHOOSING A METHOD

SCENARIO	RECOMMENDATION
< 100 GB, version migration, single schema	pg_dump -Fd -j N
TB scale, short RTO	pgBackRest / pg_basebackup + WAL
RPO in seconds	Streaming replica + pg_receivewal
"I deleted it by mistake" protection	PITR + recovery_min_apply_delay

TOOL MAP

Logical	pg_dump · pg_dumpall · pg_restore
Physical	pg_basebackup · pg_combinebackup (17+)
WAL	archive_command · pg_receivewal · pg_waldump
Enterprise	pgBackRest · Barman · WAL-G

Replication is not a backup: DROP TABLE or a bad UPDATE reaches every standby within milliseconds. HA ≠ DR.

02 Logical Backup



PG_DUMP FORMAT	RESTORE TOOL	PARALLEL	SELECTIVE
-Fp plain	psql	x	x
-Fc custom	pg_restore	restore	✓
-Fd directory	pg_restore	dump + restore	✓
-Ft tar	pg_restore	x	✓

PARALLEL DUMP & GLOBAL OBJECTS

```
# 8 parallel jobs, zstd compression (PG16+)
$ pg_dump -h db1 -U backup -Fd -j 8 -Z zstd:3 \
  -f /bk/shop_$(date +%F) shop
# Roles, tablespaces, settings
$ pg_dumpall -h db1 -U postgres --globals-only \
  > /bk/globals_$(date +%F).sql
```

PARALLEL & SELECTIVE RESTORE

```
$ psql -f globals.sql postgres
$ createdb -O app shop
$ pg_restore -j 8 -d shop --no-owner /bk/shop_dir
# Single table only:
$ pg_restore -l /bk/shop_dir > list.txt # edit it
$ pg_restore -L list.txt -d shop /bk/shop_dir
$ pg_restore -t orders -d shop /bk/shop_dir
```

Consistency Single snapshot (REPEATABLE READ) — does not block writes; a long dump **holds back vacuum**.

Version Upgrade Always use the **target version's** pg_dump binary.

Caution: pg_dump does not include roles or tablespaces. Always pair it with pg_dumpall --globals-only — otherwise restore fails with "role does not exist".

03 Physical Backup & WAL Archiving



PARAMETER	VALUE / NOTE
wal_level	replica (default) — sufficient for PITR
archive_mode	on — requires restart
archive_command	Must return 0 only on success; or archive_library (15+)
archive_timeout	60s — RPO ceiling under low traffic
summarize_wal	on — for incremental backup (17+)

BASE BACKUP + VERIFICATION

```
archive_command = 'test ! -f /arch/%f && cp %p /arch/%f'

$ pg_basebackup -h db1 -U repl -D /bk/base_$(date +%F) \
  -Ft -z -Xs -P -c fast
$ pg_verifybackup /bk/base_2026-10-13 # manifest (13+)
```

NATIVE INCREMENTAL (PG 17+)

```
$ pg_basebackup -D /bk/incr1 \
  --incremental=/bk/full/backup_manifest
# combine the chain before restore:
$ pg_combinebackup /bk/full /bk/incr1 /bk/incr2 -o /restore
```

TOOL	HIGHLIGHTS
pgBackRest	Full/diff/incr, parallel, compression + encryption, S3/GCS/Azure, verify
Barman	Central backup server, streaming WAL, retention policy
WAL-G	Cloud-focused, delta backup, lightweight setup

Disk-Full Risk: archive_command keeps failing → WAL piles up in pg_wal → disk fills → **server stops**. pg_stat_archiver.failed_count needs an alert.

04 Point-in-Time Recovery CRITICAL

STEP BY STEP PITR

- 1
- Stop the server: `systemctl stop postgresql`
- 2
- Do **not delete**, **move** the old data directory — unarchived WAL in `pg_wal` may be needed.
- 3
- Extract the base backup (for tar: `base.tar.gz` + `pg_wal.tar.gz`), owner: `postgres`.
- 4
- Add `restore_command` + target parameters to `postgresql.conf`.
- 5
- `touch $PGDATA/recovery.signal` → start, watch the log.
- 6
- Check the data at the target → `SELECT pg_wal_replay_resume();` to promote.

📄 POSTGRES.CONF (PG 12+)

```
restore_command      = 'cp /arch/%f %p'
recovery_target_time  = '2026-10-13 09:59:59+03'
recovery_target_inclusive = off
recovery_target_action = 'pause' # pause|promote|shutdown
recovery_target_timeline = 'latest' # default
```

TARGET	WHEN?
recovery_target_time	Incident time is known (include the time zone!)
recovery_target_xid	The bad transaction ID is known
recovery_target_lsn	pg_waldump exact WAL position found with
recovery_target_name	pg_create_restore_point() marked point
recovery_target	'immediate' — stop as soon as consistent

⚡ PRE-DEPLOY MARKER & PGBACKREST PITR

```
SELECT pg_create_restore_point('before_deploy_v42');
$ pgbackrest --stanza=main --delta --type=time \
  --target="2026-10-13 09:59:59+03" \
  --target-action=promote restore
```

⚠ **Caution:** Promotion starts a **new timeline**. Take a **new base backup** from the recovered server right away; the old chain does not cover the new timeline.

05 Verification, Testing & Monitoring

CHECK	TOOL / METHOD
Backup Integrity	pg_verifybackup (manifest), pgbackrest verify
Data Integrity	Data checksums, pg_amcheck (14+), pg_checksums --check
Archive Health	pg_stat_archiver, count of .ready files
Restore Test	Automated, at least weekly; restore on a separate server → validate with queries → measure RTO and record it.

💖 WAL ARCHIVE HEALTH CHECK

```
SELECT archived_count, last_archived_wal,
       now() - last_archived_time AS since_last,
       failed_count, last_failed_wal, last_failed_time
FROM   pg_stat_archiver;
-- WAL files waiting to be archived:
SELECT count(*) FROM pg_ls_archive_statusdir()
WHERE  name LIKE '%.ready';
```

🔗 PGBACKREST & INTEGRITY CHECKS

```
$ pgbackrest --stanza=main check # archive + config test
$ pgbackrest --stanza=main info  # backups & WAL range
$ pgbackrest --stanza=main verify
$ pg_amcheck -d shop --heapallindexed -j 4
```

⬆ SET UP ALERTS

Age of last successful backup > 26 hours	CRITICAL
failed_count increase · .ready pile-up	CRITICAL
now() - last_archived_time > RPO	CRITICAL
Sudden ±30% change in backup size	IMPORTANT
pg_wal / backup disk usage > 80%	IMPORTANT

📌 **PG 18:** `initdb` now enables data checksums **by default**. On older clusters: `pg_checksums --enable` (with the server stopped).

06 Disaster Scenarios & Checklist

SCENARIO	SOLUTION
Accidental DROP / DELETE	PITR (<code>recovery_target_xid</code> / <code>_time</code>) or from a delayed standby via <code>pg_dump -t</code>
Disk / server loss	Promote standby (<code>pg_promote()</code>) → new standby: <code>pgbackrest --type=standby restore</code>
Ransomware	Immutable repo (S3 object lock) + <code>repo-cipher-type</code> + separate credentials
Corrupt page	Checksum / <code>pg_amcheck</code> detects it → restore the affected object from backup
Region loss	pgBackRest multi-repo (<code>repo2</code> in another region) + DR runbook

🕒 DELAYED REPLICA (1 HOUR)

```
# standby: postgresql.conf
recovery_min_apply_delay = '1h'
-- during an incident: pause replay, recover the data
SELECT pg_wal_replay_pause();
```

✓ RPO / RTO documented and signed off by the business SLA → method choice	CRITICAL
✓ WAL archiving enabled and monitored archive_mode · pg_stat_archiver	CRITICAL
✓ Automated restore test weekly · RTO measurement	CRITICAL
✓ Off-site + encrypted copy repo-cipher-type · S3 object lock	CRITICAL
✓ Retention policy enforced repo1-retention-full · repo1-retention-diff	IMPORTANT
✓ Global objects backed up pg_dumpall --globals-only	IMPORTANT
✓ Config files version-controlled postgresql.conf · pg_hba.conf · pgbackrest.conf	RECOMMENDED
✓ DR runbook current and rehearsed 2 drills per year	RECOMMENDED
CRITICAL now IMPORTANT this quarter RECOMMENDED plan it	