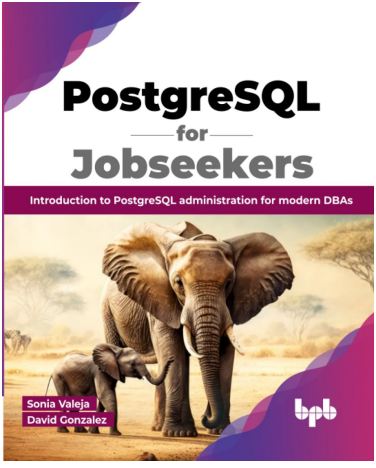
Database Administration

Lecture 07: Backup and Restore.

Valeja & Gonzales.

31 de agosto de 2025

Database Administration: Backup and Restore.



Content has been extracted from *PostgreSQL for Jobseekers (Chapter 7)*, by Sonia Valeja and David Gonzales, 2023. Visit https://bpbonline.com/products/postgresql-for-jobseekers.

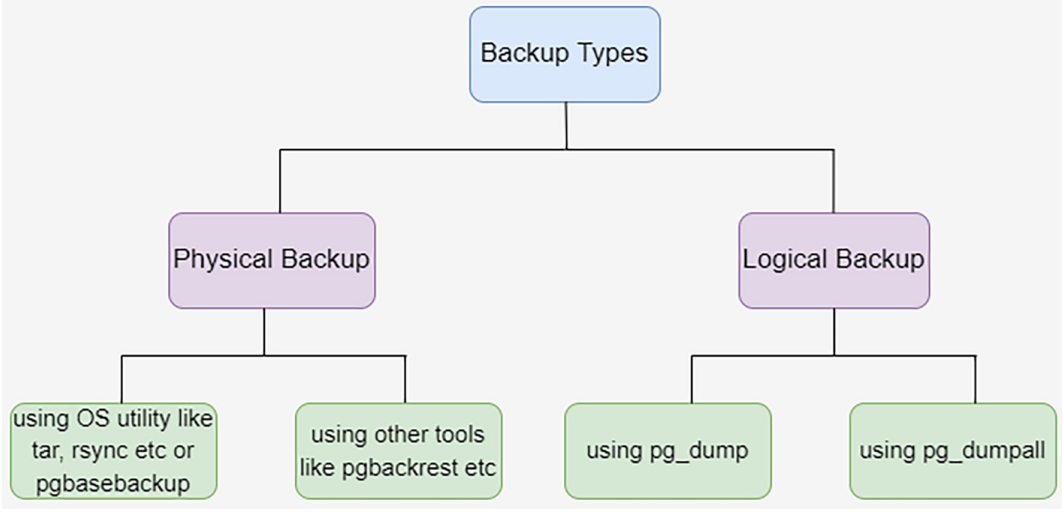
Introduction

- Regular backups are crucial for database administration.
- Backups enable recovery from accidental changes.
- PostgreSQL provides multiple backup and restore tools.

Types of Backups

- **Physical Backup**: Copies entire data directory.
- **Logical Backup**: Exports data in human-readable format.

Types of Backups



Physical Backup

- Uses filesystem-level copying.
- Steps:
  1. Create a checkpoint.
  2. Copy data directory (e.g., using `tar` or `rsync`).
  3. Stop backup process.
- Example tool: `pg_basebackup`.

pg_basebackup

```
postgres@ip-172-31-27-8:~$ pg_basebackup --help
pg_basebackup takes a base backup of a running PostgreSQL server.

Usage:
  pg_basebackup [OPTION]...

Options controlling the output:
  -D, --pgdata=DIRECTORY receive base backup into directory
  -F, --format=p|t       output format (plain (default), tar)
  -r, --max-rate=RATE    maximum transfer rate to transfer data directory
                         (in kB/s, or use suffix "k" or "M")
  -R, --write-recovery-conf
                         write configuration for replication
  -T, --tablespace-mapping=OLDDIR=NEWDIR
                         relocate tablespace in OLDDIR to NEWDIR
      --waldir=WALDIR    location for the write-ahead log directory
  -X, --wal-method=none|fetch|stream
                         include required WAL files with specified method
```

Point in Time Recovery (PITR) / Archival

PITR by enabling **archiving**, this allows restoring the database to a specific time frame, provided that WAL[1]/archives are restored until that point.

Steps:

- Enable archiving in `postgresql.conf`.
- Configure the following parameters:

Configuration Example:

```
archive_mode = on
archive_command = 'test ! -f /mnt/server/archivedir/%f && \
    cp %p /mnt/server/archivedir/%f' # Linux
archive_command = 'copy "%p" "C:\\server\\archivedir\\%f"' # Windows
```

We will see in the restore section how to recover a database to a specific time frame.

[1] Write-Ahead Logging

Pros and Cons of Physical Backup

Pros:

- Contains a full filesystem-level backup of the data directory.
- Includes configuration files such as `postgresql.conf`, `pg_hba.conf`, etc.
- Simple process: copy the data directory and custom tablespaces to the destination server and start PostgreSQL.

Cons:

- Only cluster-level backups are possible (no single database or table backup).
- Consistency of the backup is difficult to guarantee.
- Backup/restore speed depends heavily on server hardware and network speed (for remote copies).

Logical Backup

- Exports data in SQL format.
- Allows individual table or database backups.
- Example tools:
  - `pg_dump` for single database.
  - `pg_dumpall` for full cluster backup.

pg_dump

```
postgres@ip-172-31-27-8:~$ pg_dump --help
pg_dump dumps a database as a text file or to other formats.

Usage:
  pg_dump [OPTION]... [DBNAME]

General options:
  -f, --file=FILENAME          output file or directory name
  -F, --format=c|d|t|p         output file format (custom, directory, tar,
                               plain text (default))
  -j, --jobs=NUM               use this many parallel jobs to dump
  -v, --verbose                verbose mode
  -V, --version                output version information, then exit
  -Z, --compress=0-9           compression level for compressed formats
  --lock-wait-timeout=TIMEOUT  fail after waiting TIMEOUT for a table lock
  --no-sync                    do not wait for changes to be written safely to disk
  -?, --help                   show this help, then exit

Options controlling the output content:
  -a, --data-only              dump only the data, not the schema
  -b, --blobs                  include large objects in dump
  -B, --no-blobs               exclude large objects in dump
  -c, --clean                  clean (drop) database objects before recreating
```

Pros and Cons of Logical Backup

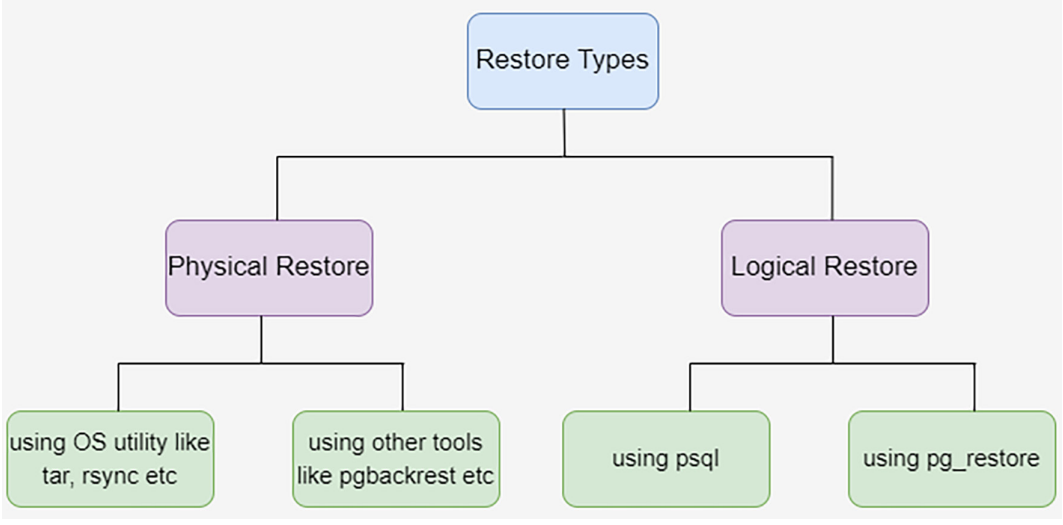
Pros:

- Can back up specific parts of the database (not just the entire cluster).
- Simple and human-readable format.
- Backups are stored as plain text/SQL files, making them portable across different operating systems.
- Faster than physical backup when only a single DB object is required.

Cons:

- Configuration files are not included (must be backed up separately).
- Point-in-Time Recovery (PITR) is not supported.

Types of Restore



Restore Methods

- `psql`: Restores backups in plain SQL format.
- `pg_restore`: Restores backups from compressed/custom format.

pg_restore

```
postgres@ip-172-31-27-8:~$ pg_restore --help
pg_restore restores a PostgreSQL database from an archive created by pg_dump.

Usage:
  pg_restore [OPTION]... [FILE]

General options:
  -d, --dbname=NAME        connect to database name
  -f, --file=FILENAME      output file name (- for stdout)
  -F, --format=c|d|t       backup file format (should be automatic)
  -l, --list               print summarized TOC of the archive
  -v, --verbose            verbose mode
  -V, --version            output version information, then exit
  -?, --help               show this help, then exit

Options controlling the restore:
  -a, --data-only              restore only the data, no schema
  -c, --clean                  clean (drop) database objects before recreating
  -C, --create                 create the target database
  -e, --exit-on-error          exit on error, default is to continue
  -I, --index=NAME             restore named index
```

Logical Backup and Restore example

Description	Command
To dump a database called mydb into a SQL-script file:	`pg_dump mydb > "/opt/mydb_dump.sql"`
To reload such a script into a (freshly created) database named newdb:	`psql -d newdb -f "/opt/mydb_dump.sql"`
To dump a database into a custom-format archive file:	`pg_dump -Fc mydb > "/opt/mydb_dump.dump"`
To reload an archive file into a (freshly created) database named newdb:	`pg_restore -d newdb "/opt/mydb_dump.dump"`
To dump a single table named mytab	`pg_dump -t mytab mydb > "/opt/mydb_dump.sql"`
To dump the database with the log file using user test	`pg_dump -U test -v -f "/opt/mydb_dump.sql" mydb 2>> "/opt/mydb_log.log"` `password for user test :`
To dump only structure without data	`pg_dump -sU test -v -f "/opt/mydb_dump.sql" mydb 2>> "/opt/mydb_log.log"` `password for user test :`
To take insert scripts of particular table	`pg_dump --column-inserts -a -t mytab -U test mydb > "/opt/mytab_inserts.sql"`
To dump only specific tables with data	`pg_dump -U test -n schema1 -t BCL_* -f "/opt/BCL_TABLES.sql" mydb`

Useful Backup and Restore Tools

Community-developed tools:

- Support full, incremental, and differential backups.
- Provide backup compression and checksum.
- Allow multiple repositories: on-premise or cloud.
- Enable parallelism for faster operations.
- Maintain a backup catalog.
- Offer alerting and notification features.

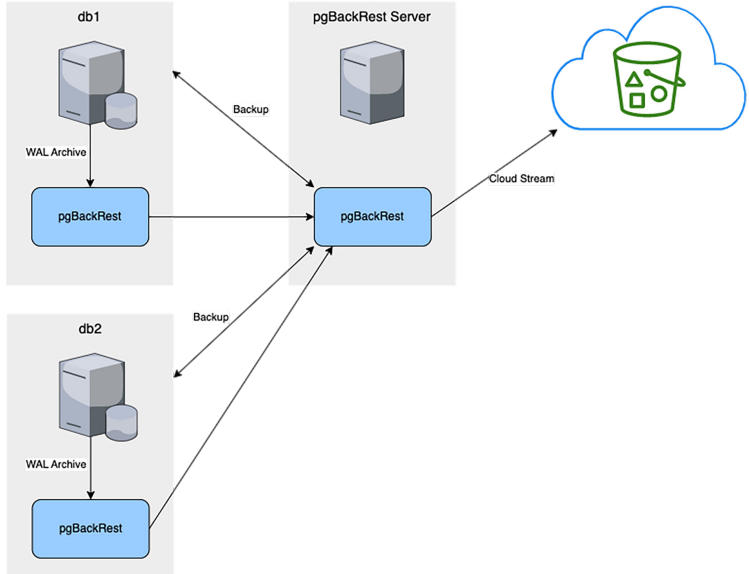
We will next review some of the most popular open-source tools, their options, and how to perform a backup and restore.

Advanced Backup Tools

pgBackRest

- Supports full, incremental, and differential backups.
- Uses parallelism for faster operations.
- Supports cloud storage.

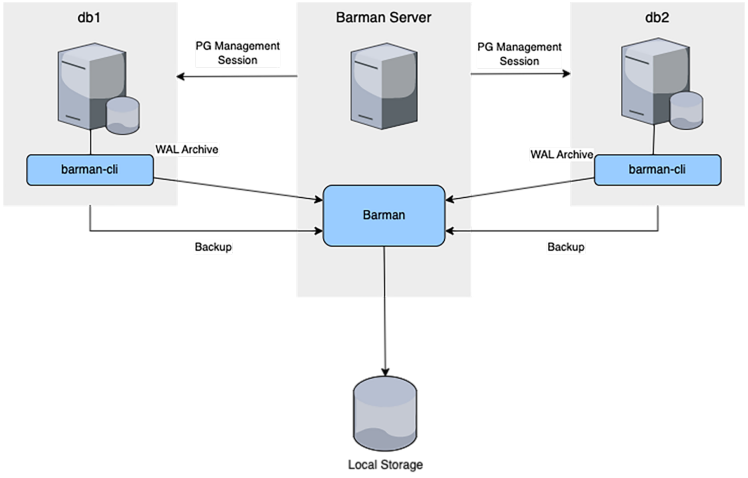
pgBackRest


db1
pgBackRest Server
Backup
WAL Archive
pgBackRest
pgBackRest
Cloud Stream
Backup
db2
WAL Archive
pgBackRest

Barman

- Used for remote backups.
- Provides incremental backup and deduplication.
- Maintains a backup catalog.

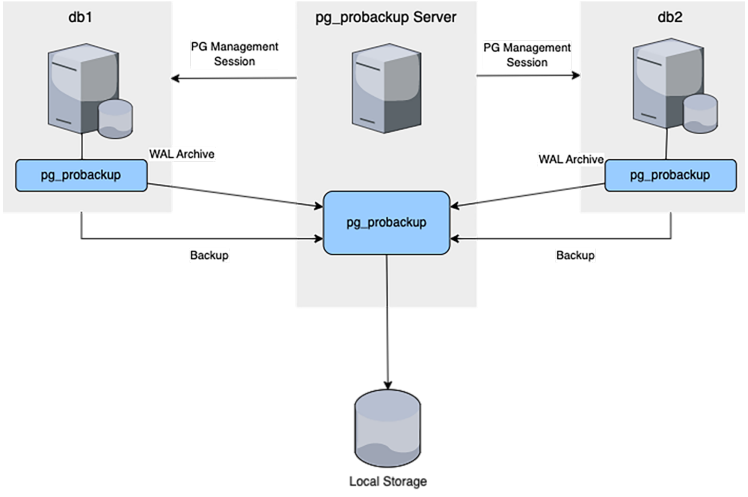
Barman


db1
Barman Server
db2
PG Management Session
PG Management Session
WAL Archive
WAL Archive
barman-cli
barman-cli
Barman
Backup
Backup
Local Storage

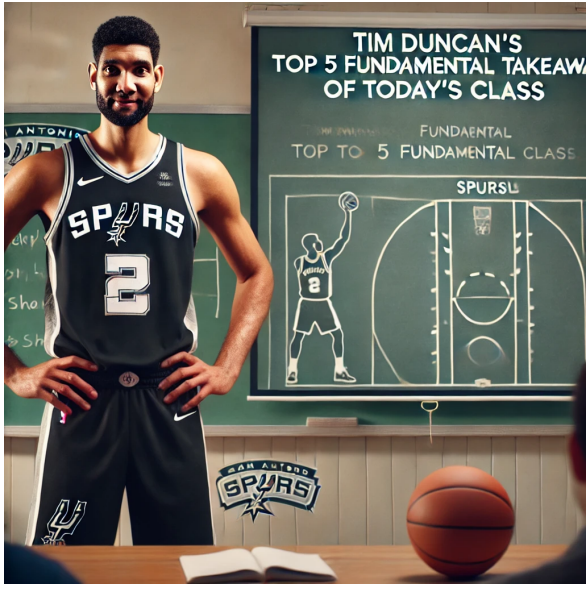
pg_probackup

- Provides incremental restore to save time.
- Supports backup merging and deduplication.
- Offers parallelism and remote operations.

pg_probackup



TDT5FTOTC


TIM DUNCAN'S
TOP 5 FUNDAMENTAL TAKEAW
OF TODAY'S CLASS
FUNDAMENTAL
TOP TO 5 FUNDAMENTAL CLASS
SPURS
SPURS
2
SPURS

TDT5FTOTC

TDT5FTOTC

5 **Regular backups** are essential for ensuring high availability, enabling point-in-time recovery, and recovering from accidental changes.

TDT5FTOTC

5 **Regular backups** are essential for ensuring high availability, enabling point-in-time recovery, and recovering from accidental changes.

4 PostgreSQL supports **two main backup types**: physical (cluster-level, filesystem copy) and logical (human-readable, object-level, portable).

TDT5FTOTC

5 **Regular backups** are essential for ensuring high availability, enabling point-in-time recovery, and recovering from accidental changes.

4 PostgreSQL supports **two main backup types**: physical (cluster-level, filesystem copy) and logical (human-readable, object-level, portable).

3 **Built-in tools** like pg_dump, pg_dumpall, pg_basebackup, psql, and pg_restore allow creating and restoring different types of backups.

TDT5FTOTC

5 **Regular backups** are essential for ensuring high availability, enabling point-in-time recovery, and recovering from accidental changes.

4 PostgreSQL supports **two main backup types**: physical (cluster-level, filesystem copy) and logical (human-readable, object-level, portable).

3 **Built-in tools** like pg_dump, pg_dumpall, pg_basebackup, psql, and pg_restore allow creating and restoring different types of backups.

2 **Point-in-Time Recovery** (PITR) requires WAL archiving and is only supported with physical backups, not logical ones.

TDT5FTOTC

5 **Regular backups** are essential for ensuring high availability, enabling point-in-time recovery, and recovering from accidental changes.

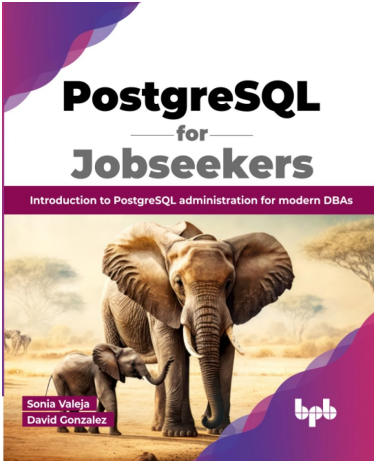
4 PostgreSQL supports **two main backup types**: physical (cluster-level, filesystem copy) and logical (human-readable, object-level, portable).

3 **Built-in tools** like pg_dump, pg_dumpall, pg_basebackup, psql, and pg_restore allow creating and restoring different types of backups.

2 **Point-in-Time Recovery** (PITR) requires WAL archiving and is only supported with physical backups, not logical ones.

1 **Community tools** such as pgBackRest, Barman, and pg_probackup provide advanced features like incremental backups, compression, cloud storage, and catalogs.

Database Administration: Backup and Restore.



Content has been extracted from *PostgreSQL for Jobseekers (Chapter 7)*, by Sonia Valeja and David Gonzales, 2023. Visit https://bpbonline.com/products/postgresql-for-jobseekers.