Démystifier les sauvegardes incrémentales dans l'écosystème PostgreSQL

data egret

Your remote PostgreSQL DBA team

Stefan FERCOT

contact@dataegret.com

Securing your PostgreSQL database availability and high performance.

- Performance audit
- Backup & restore
- Migration
- Cloud Cost Management
- Architecture review
- DataOps/ CDC projects
- 24/7 Incident support

on premises & cloud

EXPERTISE

Senior DBA team with **10+ years of PostgreSQL experience** each.

DEVELOPMENT

Involved in **new feature and extension development**.

TAILORED APPROACH

Dedicated DBA team that focused on success of your project.

COMMUNITY

Recognised significant **contributing sponsor to PostgreSQL**.

Stefan Fercot

- Senior PostgreSQL Expert **@Data Egret**
- pgBackRest fan & contributor
- aka. pgstef
- https://pgstef.github.io

Agenda

- Sauvegardes incrémentales dans PostgreSQL 17
  - Extraits des journaux de transaction (“WAL summaries”)
  - Copie incrémentale avec `pg_basebackup`
  - `pg_combinebackup`
- Limitations et autres considérations utiles
  - `pg_verifybackup`, *evergreen* backup
- L’écosystème
  - Exemple comparatif : `pg_basebackup` vs pgBackRest

PostgreSQL 17

PostgreSQL 17 Released on September 26, 2024.

https://www.postgresql.org/about/news/postgresql-17-released-2936/

Sauvegardes incrémentales

```
$ git show dc212340058b4e7ecfc5a7a81ec50e7a207bf288
Author: Robert Haas <rhaas@postgresql.org>
Date:   Wed Dec 20 09:49:12 2023 -0500
    Add support for incremental backup.

    An incremental backup is like a regular full backup except that some relation files are replaced
    with files with names like INCREMENTAL.${ORIGINAL_NAME}, and the backup_label file contains
    additional lines identifying it as an incremental backup.
    The new pg_combinebackup tool can be used to reconstruct a data directory from a full backup and
    a series of incremental backups.

    Patch by me. Reviewed by Matthias van de Meent, Dilip Kumar, Jakub Wartak, Peter Eisentraut,
    and Álvaro Herrera. Thanks especially to Jakub for incredibly helpful and extensive testing.
```

encore plus de travail réalisé après ça,
avec de nombreux hackers impliqués !

Toujours se référer à la doc !

https://www.postgresql.org/docs/current/continuous-archiving.html#BACKUP-INCREMENTAL-BACKUP

25.3.3. Making an Incremental Backup

You can use **pg_basebackup** to take an incremental backup by specifying the `--incremental` option. You must supply, as an argument to `--incremental`, the backup manifest to an earlier backup from the same server. In the resulting backup, non-relation files will be included in their entirety, but some relation files may be replaced by smaller incremental files which contain only the blocks which have been changed since the earlier backup and enough metadata to reconstruct the current version of the file.

To figure out which blocks need to be backed up, the server uses WAL summaries, which are stored in the data directory, inside the directory `pg_wal/summaries`. If the required summary files are not present, an attempt to take an incremental backup will fail. The summaries present in this directory must cover all LSNs from the start LSN of the prior backup to the start LSN of the current backup. Since the server looks for WAL summaries just after establishing the start LSN of the current backup, the necessary summary files probably won't be instantly present on disk, but the server will wait for any missing files to show up. This also helps if the WAL summarization process has fallen behind. However, if the necessary files have already been removed, or if the WAL summarizer doesn't catch up quickly enough, the incremental backup will fail.

When restoring an incremental backup, it will be necessary to have not only the incremental backup itself but also all earlier backups that are required to supply the blocks omitted from the incremental backup. See **pg_combinebackup** for further information about this requirement. Note that there are restrictions on the use of `pg_combinebackup` when the checksum status of the cluster has been changed; see **pg_combinebackup limitations**.

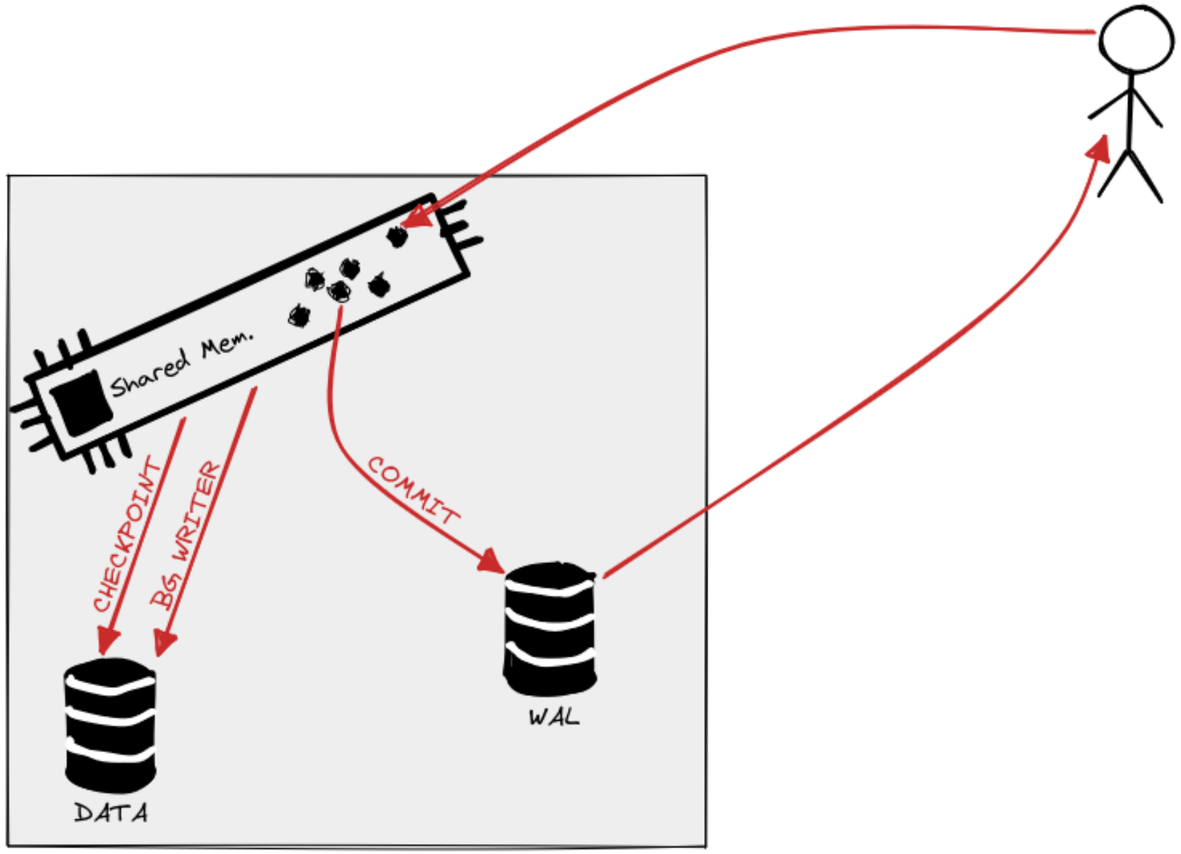
Note that all of the requirements for making use of a full backup also apply to an incremental backup. For instance, you still need all of the WAL segment files generated during and after the file system backup, and any relevant WAL history files. And you still need to create a `recovery.signal` (or `standby.signal`) and perform recovery, as described in **Section 25.3.5**. The requirement to have earlier backups available at restore time and to use `pg_combinebackup` is an additional requirement on top of everything else. Keep in mind that PostgreSQL has no built-in mechanism to figure out which backups are still needed as a basis for restoring later incremental backups. You must keep track of the relationships between your full and incremental backups on your own, and be certain not to remove earlier backups if they might be needed when restoring later incremental backups.

Incremental backups typically only make sense for relatively large databases where a significant portion of the data does not change, or only changes slowly. For a small database, it's simpler to ignore the existence of incremental backups and simply take full backups, which are simpler to manage. For a large database all of which is heavily modified, incremental backups won't be much smaller than full backups.

Qu’est-ce qu’une sauvegarde incrémentale ?

- plutôt que de copier l’instance complète…
  - ne copier que les données réellement modifiées
- ➕ plus rapide, moins d’espace disque nécessaire
- ➖ plus de travail nécessaire pour la restauration !
- ❓ comment identifier rapidement et de manière fiable quelles données ont changé ?

Journaux de transaction (Write Ahead Log)



Extraits des journaux de transaction - WAL summarizer

```
$ git show 174c480508ac25568561443e6d4a82d5c1103487
Author: Robert Haas <rhaas@postgresql.org>
Date:   Wed Dec 20 08:41:09 2023 -0500
    Add a new WAL summarizer process.
```

- `summarize_wal` active ou désactive ce nouveau *background process*
- crée des extraits des journaux de transaction
  - dans `pg_wal/summaries`
  - un fichier par cycle de *checkpoint*

Optionnel : configurer `wal_summary_keep_time`

- *WAL summarizer* supprime automatiquement les fichiers
  - plus vieux que `wal_summarize_keep_time` (10 jours par défaut)
- lorsque vous effectuez une sauvegarde incrémentale :
  - à quand remonte la sauvegarde précédente ?
  - `wal_summary_keep_time` doit être supérieur à cette durée

Exemple

```
postgres=# ALTER SYSTEM SET summarize_wal TO on;
ALTER SYSTEM
postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)

postgres=# SELECT * FROM pg_settings WHERE name='summarize_wal';
name             | summarize_wal
setting          | on
category         | Write-Ahead Log / Summarization
short_desc       | Starts the WAL summarizer process to enable incremental backup.
context          | sighup
```

Exemple (2)

```
$ pgbench -i -s 60 pgbench
$ du -hs 17/data/pg_wal/summaries/*
28K 17/data/pg_wal/summaries/000000010000000017813F800000000223FEF00.summary
20K 17/data/pg_wal/summaries/000000010000000223FEF00000000002FC4CA18.summary

$ psql -c "select * from pg_available_wal_summaries();"
 tli |  start_lsn  |  end_lsn
-----+-------------+------------
   1 | 0/17813F8   | 0/223FEF00
   1 | 0/223FEF00  | 0/2FC4CA18
```

- pour une analyse plus approfondie : client `pg_walsummary`
- fonctions SQL
  - `pg_get_wal_summarizer_state()`
  - `pg_available_wal_summaries()`, `pg_wal_summary_contents()`

pg_basebackup

```
$ pg_basebackup --help|grep incremental
  -i, --incremental=OLDMANIFEST
                           take incremental backup
```

- ne fonctionne qu'avec le format `plain`
- il suffit de fournir le manifeste de la sauvegarde précédente (référence)

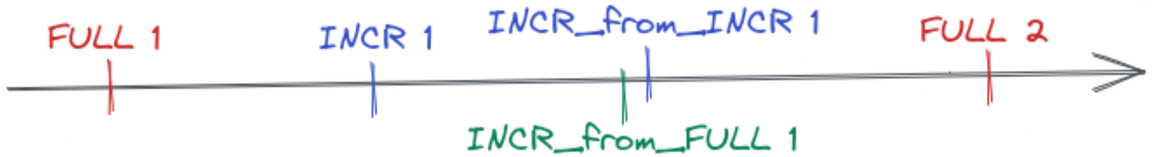
Qu’est-ce qu’un manifeste de sauvegarde ?

```
$ pg_basebackup -D full_1 --checkpoint=fast
$ cat full_1/backup_manifest
"System-Identifier": 7473522949340343009,
"Files": [
...
{ "Path": "base/1/1255", "Size": 811008,
    "Last-Modified": "2025-02-20 15:25:58 GMT",
    "Checksum-Algorithm": "CRC32C", "Checksum": "c33fb063" },
],
"WAL-Ranges": [
{ "Timeline": 1, "Start-LSN": "0/30000028", "End-LSN": "0/30000120" }
],
```

Exemple

```
$ pgbench -i -s 10 pgbench
$ pg_basebackup -D incr_1 --incremental=full_1/backup_manifest -c fast

$ pgbench -i -s 10 pgbench
$ pg_basebackup -D incr_from_full1 --incremental=full_1/backup_manifest -c fast
$ pg_basebackup -D incr_from_incr1 --incremental=incr_1/backup_manifest -c fast
$ pg_basebackup -D full_2 --checkpoint=fast
```



pg_combinebackup

- reconstruit un répertoire de données (PGDATA)
  - à partir d'une sauvegarde incrémentale
  - et de toutes les sauvegardes dont elle dépend

```
$ pg_combinebackup --output=restore_1 full_1 incr_1 incr_from_incr1

$ pg_combinebackup --output=restore_2 full_1 incr_from_full1

$ pg_combinebackup --output=restore_3 full_1 incr_from_incr1
error: backup at "full_1" starts at LSN 0/30000028, but expected 0/39000028
```

- pas d'historique des dépendances entre les sauvegardes !

Limitations

- ne peut fonctionner qu'avec des sauvegardes stockées localement
- ne recalcule pas les sommes de contrôle des pages (*page checksums*)
  - prendre une nouvelle sauvegarde complète après avoir activé/désactivé les sommes de contrôle
- ne vérifie que si les sauvegardes sont correctement ordonnées…
  - pas que chaque sauvegarde est correcte/intacte
  - pour ça, utilisez `pg_verifybackup` !

pg_verifybackup

orthogonalité : un outil doit faire une seule tâche et la faire aussi bien que possible

```
$ cp -rfP incr_from_incr1 incr_from_incr1_corrupted
$ rm incr_from_incr1_corrupted/base/1/INCREMENTAL.2675
$ pg_verifybackup incr_from_incr1_corrupted
error: "base/1/INCREMENTAL.2675" is present in the manifest but not on disk

$ pg_combinebackup --output=restore_3 full_1 incr_1 incr_from_incr1_corrupted
$ pg_verifybackup restore_3
backup successfully verified
```

The “evergreen” backup

- `pg_combinebackup` va produire…
  - une sauvegarde complète *“artificielle”* !
  - qui peut être utilisée comme source pour de nouvelles sauvegardes
  - ou pour d’autres invocations de `pg_combinebackup`

```
# idée initiale par Robert Haas
$ pg_basebackup -D sunday_full -c fast
$ pg_basebackup -D monday_incr --incremental=sunday_full/backup_manifest -c fast
$ pg_combinebackup -o monday_full sunday_full monday_incr
$ rm -rf sunday_full monday_incr
```

L’écosystème

- pgBarman
- WAL-G
- pgBackRest

pgBarman

- version 3.13.0, sortie le 20 Février 2025
- https://github.com/EnterpriseDB/barman
- support des sauvegardes incrémentales
  - *file-level* via `rsync`
  - *block-level* via `pg_basebackup`
    - depuis la version 3.11.0, sortie le 22 Août 2024

WAL-G

- version 3.0.5, sortie le 10 Janvier 2025
- https://github.com/wal-g/wal-g
- supporte PostgreSQL, MS SQL Server, et Greenplum
  - *unstable* pour MongoDB, MySQL/MariaDB et Redis
- delta backup

Delta computation is based on ModTime of file system and LSN number of pages in datafiles

WAL-G (2)

```
WALG_DELTA_MAX_STEPS
Delta-backup is the difference between previously taken backup and present state.
WALG_DELTA_MAX_STEPS determines how many delta backups can be between full backups.
Delta computation is based on ModTime of file system and LSN number of pages in datafiles.
```

```
WALG_DELTA_FROM_NAME
To configure base for next delta backup.
WALG_DELTA_ORIGIN can be LATEST (chaining increments),
LATEST_FULL (for bases where volatile part is compact and chaining has no meaning).
```

pgBackRest

- version 2.54.2, sortie le 20 Janvier 2025
- https://github.com/pgbackrest/pgbackrest
- utilisation de `repo-block` recommandée à partie de la v2.52.1

> *Block incremental backups save space by only storing the parts of a file that have changed since the prior backup rather than storing the entire file.*

Exemple comparatif : pg_basebackup vs pgBackRest

```
$ createdb pgbench
$ pgbench -i -s 600 pgbench

$ cat /var/lib/pgsql/17/data/postgresql.auto.conf
summarize_wal = 'on'
archive_mode = on
archive_command = 'pgbackrest --stanza=demo archive-push %p'
```

pgBackRest config

```
$ cat /etc/pgbackrest.conf
[global]
repo1-path=/var/lib/pgbackrest/1
repo1-retention-full=2

repo2-path=/var/lib/pgbackrest/2
repo2-retention-full=2
repo2-bundle=y
repo2-block=y

compress-type=zst
start-fast=y
[demo]
pg1-path=/var/lib/pgsql/17/data
```

Sauvegarde complète (full)

```
$ pg_basebackup -D /var/lib/pgsql/17/backups/full_1 \
  --checkpoint=fast --wal-method=none --progress
$ du -hs /var/lib/pgsql/17/backups/full_1
8.8G  /var/lib/pgsql/17/backups/full_1
$ find /var/lib/pgsql/17/backups/full_1 -type f | wc -l
1291
```

```
$ pgbackrest --stanza=demo --type=full --repo=1 backup
$ du -hs /var/lib/pgbackrest/1/backup/demo/latest/
415M  /var/lib/pgbackrest/1/backup/demo/latest/
$ find /var/lib/pgbackrest/1/backup/demo/latest/ -type f |wc -l
1385
```

```
$ pgbackrest --stanza=demo --type=full --repo=2 backup
$ du -hs /var/lib/pgbackrest/2/backup/demo/latest/
393M  /var/lib/pgbackrest/2/backup/demo/latest/
$ find /var/lib/pgbackrest/2/backup/demo/latest/ -type f |wc -l
15
```

Sauvegarde incrémentale

```
$ pgbench -c 4 -n -b simple-update -t 100 pgbench

$ pg_basebackup -D /var/lib/pgsql/17/backups/incr_1 \
  --incremental=/var/lib/pgsql/17/backups/full_1/backup_manifest \
  --checkpoint=fast --wal-method=none --progress

$ du -hs /var/lib/pgsql/17/backups/incr_1
17M /var/lib/pgsql/17/backups/incr_1
```

```
$ pgbackrest --stanza=demo --type=incr --repo=1 backup
$ du -hs /var/lib/pgbackrest/1/backup/demo/latest/
407M  /var/lib/pgbackrest/1/backup/demo/latest/

$ pgbackrest --stanza=demo --type=incr --repo=2 backup
$ du -hs /var/lib/pgbackrest/2/backup/demo/latest/
7.3M  /var/lib/pgbackrest/2/backup/demo/latest/
```

Sauvegarde incrémentale chaînée - pg_basebackup

```
$ pgbench -c 4 -n -b simple-update -t 100 pgbench

$ pg_basebackup -D /var/lib/pgsql/17/backups/incr_from_incr_1 \
  --incremental=/var/lib/pgsql/17/backups/incr_1/backup_manifest \
  --checkpoint=fast --wal-method=none --progress
$ du -hs /var/lib/pgsql/17/backups/incr_from_incr_1
17M /var/lib/pgsql/17/backups/incr_from_incr_1

$ pg_basebackup -D /var/lib/pgsql/17/backups/incr_from_full_1 \
  --incremental=/var/lib/pgsql/17/backups/full_1/backup_manifest \
  --checkpoint=fast --wal-method=none --progress
$ du -hs /var/lib/pgsql/17/backups/incr_from_full_1
24M /var/lib/pgsql/17/backups/incr_from_full_1
```

Sauvegarde incrémentale chaînée - pgBackRest

```
$ pgbackrest --stanza=demo --type=incr --repo=2 backup
$ pgbackrest info --stanza=demo --repo=2
incr backup: 20250311-080722F_20250311-081157I
    timestamp start/stop: 2025-03-11 08:11:57+00 / 2025-03-11 08:12:03+00
    wal start/stop: 000000010000000500000EC / 000000010000000500000ED
    database size: 8.8GB, database backup size: 8.8GB
    repo2: backup size: 6.6MB
    backup reference total: 1 full, 1 incr
```

```
$ pgbackrest --stanza=demo --type=diff --repo=2 backup
$ pgbackrest info --stanza=demo --repo=2
diff backup: 20250311-080722F_20250311-081241D
    timestamp start/stop: 2025-03-11 08:12:41+00 / 2025-03-11 08:12:47+00
    wal start/stop: 000000010000000500000EF / 000000010000000500000EF
    database size: 8.8GB, database backup size: 8.8GB
    repo2: backup size: 12.5MB
    backup reference total: 1 full
```

Performance de la restauration - pg_basebackup

```
$ time pg_combinebackup -o new_full full_1 incr_1 incr_from_incr_1
real  0m11.778s
user  0m1.513s
sys 0m4.858s

$ time pg_combinebackup -o new_full full_1 incr_from_full_1
real  0m10.582s
user  0m1.442s
sys 0m4.701s
```

Performance de la restauration - pgBackRest

```
$ time pgbackrest --stanza=demo restore --repo=2 \
  --set=20250311-080722F_20250311-081157I --process-max=1
real  0m16.919s
user  0m7.698s
sys 0m3.622s

$ time pgbackrest --stanza=demo restore --repo=2 \
  --set=20250311-080722F_20250311-081157I --process-max=2
real  0m9.862s
user  0m8.044s
sys 0m3.768s
```

```
$ time pgbackrest --stanza=demo restore --repo=2 \
  --set=20250311-080722F_20250311-081241D --process-max=2
real  0m9.659s
user  0m8.124s
sys 0m3.544s
```

Résumé

- nouvelle fonctionnalité nécessitant de l'orchestration
- réfléchir attentivement à sa politique de rétention
  - utiliser régulièrement `pg_verifybackup`
  - garder une liste des inter-connexions entre les sauvegardes
  - ne pas trop compliquer les choses !
- tenter régulièrement de restaurer les sauvegardes

La suite !

Jeudi prochain :-)

https://2025.pgday.paris

20 Mars 2025

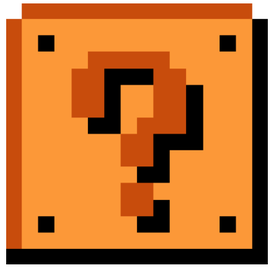
PGDay FR

3 et 4 Juin 2025

Appel à orateurs·rices ouvert ;-)

https://pgday.fr/appel

Questions ?



Merci pour votre attention !