

PostgreSQL 17 highlight
In-core incremental backups,
and a lot of details

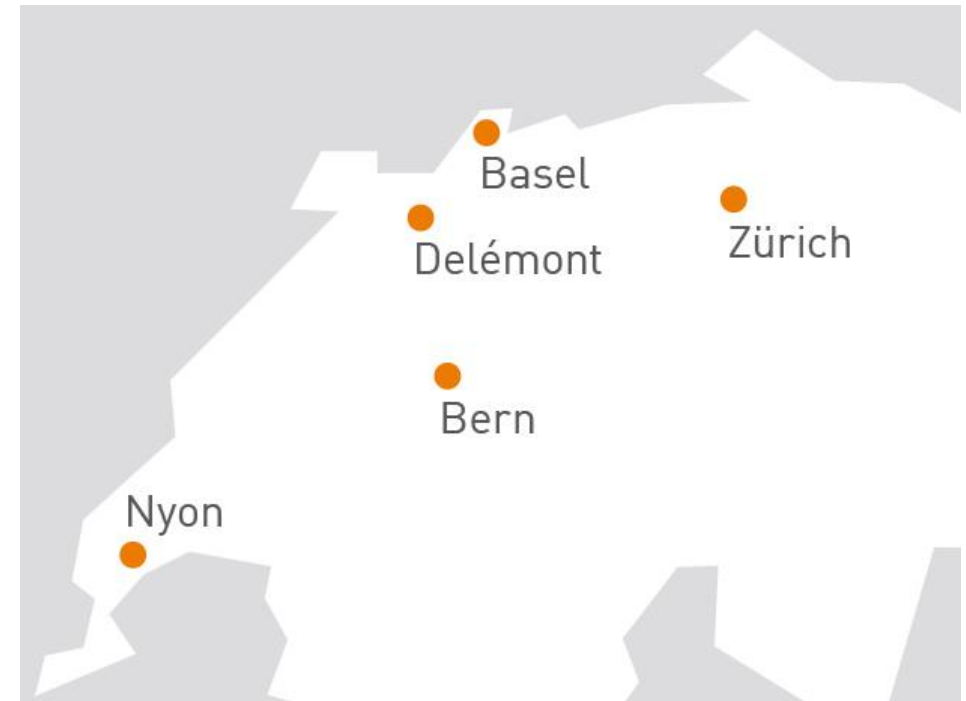
Who we are

The Company

- > Founded in 2010
- > More than 100 employees
- > Specialized in the Middleware Infrastructure
 - > The invisible part of IT
- > Customers in Switzerland and all over Europe

Our Offer

- > Consulting
- > Service Level Agreements (SLA)
- > Trainings
- > License Management



Daniel Westermann

Principal Consultant

Technology Leader Open Infrastructure

+41 79 927 2446

daniel.westermann[at]dbi-services.com



[Main page](#)[Discussion](#)

Swiss PostgreSQL Users Group

<https://swisspug.org>

Welcome

The Swiss PostgreSQL Users Group (SwissPUG) is a group of people sharing an interest in promoting the usage of the open source database system [PostgreSQL](#). SwissPUG is an association (ZGB Art. 60 ff). You will find more information about how the association is organized and which goals it follows, spread on various pages of this wiki. However, this all is still work in progress.

This wiki is the preferred way to centralize information and to coordinate activities between the contributors. If you wish to contribute, please send an E-Mail to our Administrator (admin@swisspug.org) with your name that you want to use. You will then receive a response with an initial password. After the first login you will be prompted to change the password.

To keep informed about activities consider subscribing to our [discussion](#) and [announce](#) mailing lists.

Switzerland is a multilanguage country. For that reason we encourage the usage of English for this wiki, instead of writing and synchronizing all texts in the national languages. The choice of English is due to the fact that most people working in IT use this language in their everyday business. In addition we want these pages to be open for an international audience. Of course, if you contribute, be aware that some people may not be able to understand it.

Willkommen

Die Swiss PostgreSQL Users Group (SwissPUG) ist eine Gruppe, die ein gemeinsames Interesse an der Verbreitung des Open Source Datenbank Systems [PostgreSQL](#) hat. Die SwissPUG Ziele sind auf diesem Wiki zu finden.

Bienvenue

Le Swiss PostgreSQL Users Group (SwissPUG) est un groupe dont le but est de promouvoir la diffusion du système de gestion de bases de données Open Source [PostgreSQL](#) à travers la Suisse. Vous trouverez de plus amples informations à propos de l'association et de ses buts sur cette page Wiki.

Benvenuti

Il Swiss PostgreSQL Users Group (SwissPUG) è un gruppo di persone interessate alla propagazione del sistema di banca dati open source [PostgreSQL](#) in Svizzera. Il SwissPUG è un'associazione e sugli obiettivi dell'associazione si trovano sulle pagine di questo wiki.

Bainvegni

La Swiss PostgreSQL Users Group (SwissPUG) è ina gruppia che ha in interess cuminaivel vi da la derasaziun dal sistem da banca da datas Open Source [PostgreSQL](#) en Svizra. La SwissPUG ha per l'ur finamiras chattais Vus sin quest Wiki.

[Welcome](#)[Agenda](#)[Bylaws](#)[Board](#)[Members](#)[Contact](#)[Related Organizations](#)[Apply for Membership](#)[Members Self-Care](#)[Additional](#)[Promotion](#)[Recent changes](#)[Random page](#)[Help](#)[Code of Conduct](#)[Tools](#)[What links here](#)[Related changes](#)[Special pages](#)[Printable version](#)[Permanent link](#)[Page information](#)

As promised...

No Agenda
No Slides



Before we start, all credits go to

```
postgres@pgbox:/home/postgres/$ git show dc212340058b4e7ecfc5a7a81ec50e7a207bf288
Author: Robert Haas <rhaas@postgresql.org>
Date:   Wed Dec 20 09:49:12 2023 -0500
```

Add support for incremental backup.

To take an incremental backup, you use the new replication command `UPLOAD_MANIFEST` to upload the manifest for the prior backup. This prior backup could either be a full backup or another incremental backup. You then use `BASE_BACKUP` with the `INCREMENTAL` option to take the backup. `pg_basebackup` now has an `--incremental=PATH_TO_MANIFEST` option to trigger this behavior.

...

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.

Basic usage



PostgreSQL 17 highlight - In-core incremental backups

Basic usage

We'll do brute force, start from scratch, and won't read the documentation

```
postgres@pgbox:/home/postgres/ [pgxxx] initdb --version
initdb (PostgreSQL) 17devel
postgres@pgbox:/home/postgres/ [pgxxx] export PGDATA=/var/tmp/pgday.ch
postgres@pgbox:/home/postgres/ [pgxxx] export PGPORT=8888
postgres@pgbox:/home/postgres/ [pgxxx] initdb --pgdata=$PGDATA
postgres@pgbox:/home/postgres/ [pgxxx] pg_ctl --pgdata=$PGDATA start -l /dev/null
postgres@pgbox:/home/postgres/ [pgxxx] psql -c "select version()"
                                version
```

```
-----
 PostgreSQL 17devel on x86_64-linux, compiled by gcc-7.5.0, 64-bit
(1 row)
```


PostgreSQL 17 highlight - In-core incremental backups

Basic usage



Before we can do incremental backups, we need a full backup

```
postgres@pgbox:/home/postgres/ [pgxxx] mkdir -p backups/bkpfull_1  
postgres@pgbox:/home/postgres/ [pgxxx] pg_basebackup --pgdata=backups/bkpfull_1  
postgres@pgbox:/home/postgres/ [pgxxx] echo $?  
0
```

Having a full backup, we should be able to do an incremental backup

```
postgres@pgbox:/home/postgres/ [pgxxx] pg_basebackup --help | grep incremental  
-i, --incremental=OLDMANIFEST  
take incremental backup
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

What is a backup manifest?

```
postgres@pgbox:/home/postgres/ [pgxxx] cat backups/bkpfull_1/backup_manifest
{ "PostgreSQL-Backup-Manifest-Version": 2,
  "System-Identifier": 7350636235843627793,
  "Files": [
    { "Path": "backup_label", "Size": 225, "Last-Modified": "2024-03-26 11:48:13 GMT",
      "Checksum-Algorithm": "CRC32C", "Checksum": "b92de9a2" },
    { "Path": "global/1262", "Size": 8192, "Last-Modified": "2024-03-26 11:42:46 GMT",
      "Checksum-Algorithm": "CRC32C", "Checksum": "b0545bb5" },
    { "Path": "global/2964", "Size": 0, "Last-Modified": "2024-03-26 11:42:46 GMT", "Checksum-
Algorithm": "CRC32C", "Checksum": "00000000" },
    ...
    { "Path": "global/pg_control", "Size": 8192, "Last-Modified": "2024-03-26 11:48:13 GMT",
      "Checksum-Algorithm": "CRC32C", "Checksum": "43872087" }
  ],
  "WAL-Ranges": [
    { "Timeline": 1, "Start-LSN": "0/20000D8", "End-LSN": "0/20001D0" }
  ],
  "Manifest-Checksum": "6cedb1657cd68e79c962fc8c46a1d963dfd1c41c161fcc7b0cceb5acf17d8488" }
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

Let's go for our first incremental backup

```
postgres@pgbox:/home/postgres/ [pgxxx] mkdir backups/bkpinchr_1
postgres@pgbox:/home/postgres/ [pgxxx] pg_basebackup --pgdata=backups/bkpinchr_1 \
                                         --incremental=backups/bkpfull_1/backup_manifest
pg_basebackup: error: could not initiate base backup: ERROR: incremental backups cannot
be taken unless WAL summarization is enabled
pg_basebackup: removing contents of data directory "backups/bkpinchr_1"
```

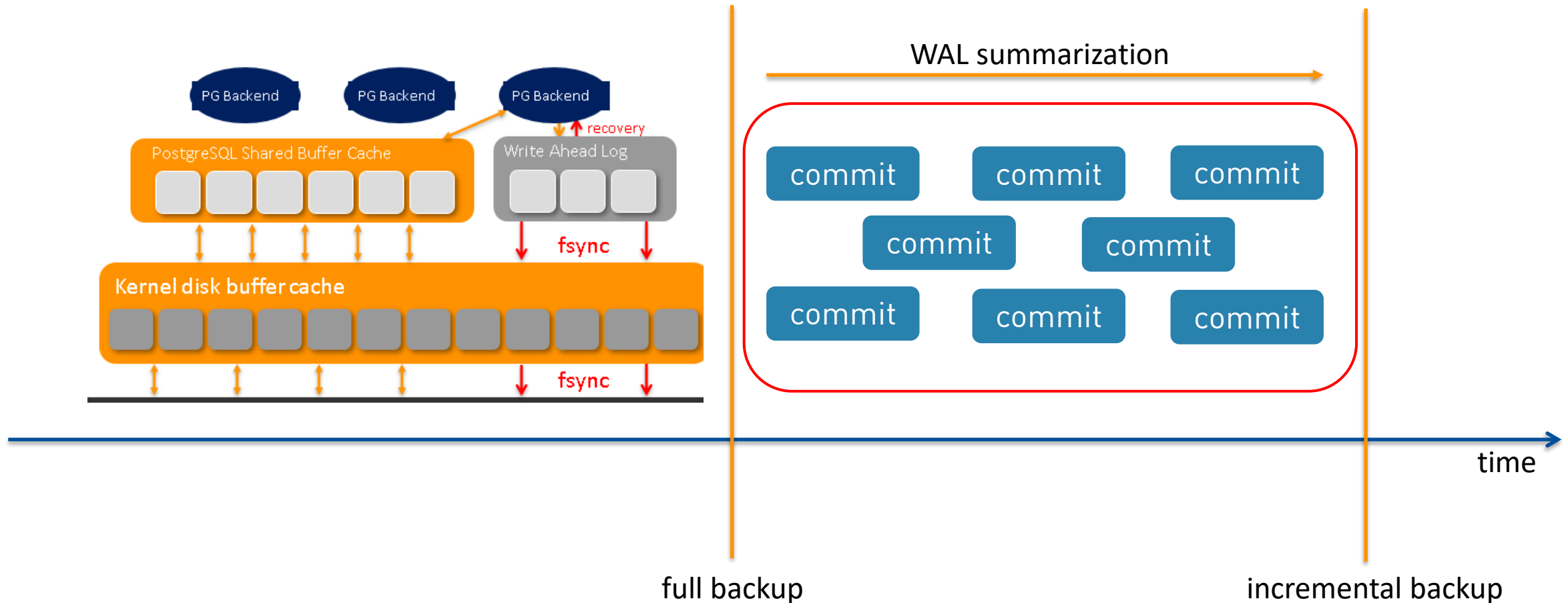
What is WAL summarization?

```
postgres@pgbox:/home/postgres/ [pgxxx] psql -c "\dconfig *summa*"
List of configuration parameters
Parameter | Value
-----+-----
summarize_wal | off
wal_summary_keep_time | 10d
(2 rows)
```


PostgreSQL 17 highlight - In-core incremental backups

Basic usage

What is WAL summarization?



PostgreSQL 17 highlight - In-core incremental backups

Basic usage

Turning on WAL summarization

```
postgres@pgbox:/home/postgres/$ echo "summarize_wal=on" \  
                                >> $PGDATA/postgresql.auto.conf  
postgres@pgbox:/home/postgres/$ psql -c "select context  
                                from pg_settings  
                                where name = 'summarize_wal'"  
  
context  
-----  
sighup  
(1 row)  
postgres@pgbox:/home/postgres/$ pg_ctl restart  
postgres@pgbox:/home/postgres/$ ps -ef | grep summa | grep -v grep  
postgres 32097 32092  0 13:19 ?          00:00:00 postgres: walsummarizer
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

WAL summary files will go to \$PGDATA/pg_wal/summaries

```
postgres@pgbox:/home/postgres/$ ls $PGDATA/pg_wal/
00000001000000000000000003 00000001000000000000000004 archive_status summaries
postgres@pgbox:/home/postgres/$ ls -l $PGDATA/pg_wal/summaries/
total 12
-rw----- 1 postgres 32 Mar 26 13:19 0000000100000000030000280000000003000060.summary
-rw----- 1 postgres 32 Mar 26 13:19 0000000100000000030000600000000003000168.summary
-rw----- 1 postgres 32 Mar 26 13:24 0000000100000000030001680000000003000218.summary
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

Now that we have WAL summaries, what about a new incremental backup?

```
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpincr_1 \  
                                --incremental=backups/bkpfull_1/backup_manifest  
...  
2024-03-26 13:34:01.404 CET [32745] ERROR:  WAL summaries are required on timeline 1 from  
0/20000D8 to 0/4000028, but the summaries for that timeline and LSN range are incomplete  
2024-03-26 13:34:01.404 CET [32745] DETAIL:  The first unsummarized LSN in this range is  
0/20000D8.  
2024-03-26 13:34:01.404 CET [32745] STATEMENT:  BASE_BACKUP ( INCREMENTAL, LABEL  
'pg_basebackup base backup', PROGRESS, WAIT 0, MANIFEST 'yes', TARGET 'client')  
pg_basebackup: error: could not initiate base backup: ERROR:  WAL summaries are required  
on timeline 1 from 0/20000D8 to 0/4000028, but the summaries for that timeline and LSN  
range are incomplete  
DETAIL:  The first unsummarized LSN in this range is 0/20000D8.  
pg_basebackup: removing contents of data directory "backups/bkpincr_1"  
...
```

> This makes sense, as we do not have WAL summaries starting from our first full backup

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

Starting from scratch

```
postgres@pgbox: /home/postgres/$ rm -rf backups/bkpfull_1/*
postgres@pgbox: /home/postgres/$ pg_basebackup --pgdata=backups/bkpfull_1
postgres@pgbox: /home/postgres/$ pg_basebackup --pgdata=backups/bkpincr_1 \
                                --incremental=backups/bkpfull_1/backup_manifest
postgres@pgbox: /home/postgres/$ du -sh backups/bkpfull_1/
39M      backups/bkpfull_1/
postgres@pgbox: /home/postgres/$ du -sh backups/bkpincr_1/
22M      backups/bkpincr_1/
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

Generating some data and compare once more

```
postgres@pgbox:/home/postgres/$ pgbench -i -s 10 postgres
postgres@pgbox:/home/postgres/$ mkdir backups/bkpincr_2
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpincr_2 \
--incremental=backups/bkpfull_1/backup_manifest \
--checkpoint=fast
postgres@pgbox:/home/postgres/$ du -sh backups/bkpfull_1/
39M      backups/bkpfull_1/
postgres@pgbox:/home/postgres/$ du -sh backups/bkpincr_1/
22M      backups/bkpincr_1/
postgres@pgbox:/home/postgres/$ du -sh backups/bkpincr_2/
172M     backups/bkpincr_2/
```


PostgreSQL 17 highlight - In-core incremental backups

Basic usage

Incremental backups can take another incremental backup as a start

```
postgres@pgbox:/home/postgres/$ psql -c "create database d"
CREATE DATABASE
postgres@pgbox:/home/postgres/$ pgbench -i -s 1 d
postgres@pgbox:/home/postgres/$ mkdir backups/bkpinchr_3
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpinchr_3 \
--incremental=backups/bkpinchr_2/backup_manifest \
--checkpoint=fast
postgres@pgbox:/home/postgres/$ du -sh backups/bkpfull_1/
39M      backups/bkpfull_1/
postgres@pgbox:/home/postgres/$ du -sh backups/bkpinchr_1/
22M      backups/bkpinchr_1/
postgres@pgbox:/home/postgres/$ du -sh backups/bkpinchr_2/
172M     backups/bkpinchr_2/
postgres@pgbox:/home/postgres/$ du -sh backups/bkpinchr_3/
45M      backups/bkpinchr_3/
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

The referenced incremental backup can be any existing incremental or full backup

```
postgres@pgbox:/home/postgres/$ mkdir backups/bkpinchr_{5..7}
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpinchr_4 \
--incremental=backups/bkpinchr_1/backup_manifest \
--checkpoint=fast
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpinchr_5 \
--incremental=backups/bkpinchr_2/backup_manifest \
--checkpoint=fast
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpinchr_6 \
--incremental=backups/bkpinchr_3/backup_manifest \
--checkpoint=fast
...
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

Now that we have incremental backups, how can we restore those?

> Say hello to "pg_combinebackup" (output truncated for readability)

```
postgres@pgbox:/home/postgres/$ pg_combinebackup --help
pg_combinebackup reconstructs full backups from incrementals.
```

Usage:

```
pg_combinebackup [OPTION]... DIRECTORY...
```

Options:

<code>-n, --dry-run</code>	don't actually do anything
<code>-o, --output</code>	output directory
<code>-T, --tablespace-mapping=OLDDIR=NEWDIR</code>	relocate tablespace in OLDDIR to NEWDIR
<code>--manifest-checksums=SHA{224,256,384,512} CRC32C NONE</code>	use algorithm for manifest checksums
<code>--no-manifest</code>	suppress generation of backup manifest
<code>--sync-method=METHOD</code>	set method for syncing files to disk
<code>--clone</code>	clone (reflink) instead of copying files
<code>--copy-file-range</code>	copy using copy_file_range() syscall

Restoring from incremental backups, starting again from scratch

```
postgres@pgbox:/home/postgres/$ rm -rf backups/*
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpfull_1 --checkpoint=fast
postgres@pgbox:/home/postgres/$ pgbench -i -s 10 postgres
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpincr_1 \
    --incremental=backups/bkpfull_1/backup_manifest \
    --checkpoint=fast
postgres@pgbox:/home/postgres/$ psql -c "create database d"
postgres@pgbox:/home/postgres/$ pgbench -i -s 1 d
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/bkpincr_2 \
    --incremental=backups/bkpincr_1/backup_manifest \
    --checkpoint=fast
postgres@pgbox:/home/postgres/$ mkdir /var/tmp/dummy
postgres@pgbox:/home/postgres/$ pg_combinebackup -o /var/tmp/dummy \
    backups/bkpfull_1/ backups/bkpincr_1/ backups/bkpincr_2/
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

The result is consistent and can be started right away

```
postgres@pgbox:/home/postgres/$ echo "port=8889" >> /var/tmp/dummy/postgresql.auto.conf
postgres@pgbox:/home/postgres/$ chmod 700 /var/tmp/dummy/
postgres@pgbox:/home/postgres/$ pg_ctl -D /var/tmp/dummy/ start -l /dev/null
postgres@pgbox:/home/postgres/$ psql -p 8889 -c "\l+"
  Name      | Owner   | Encoding | Locale Provider | Collate   | Ctype     |>
-----+-----+-----+-----+-----+-----+>
 d          | postgres | UTF8     | libc            | en_US.utf8 | en_US.utf8 |>
 postgres   | postgres | UTF8     | libc            | en_US.utf8 | en_US.utf8 |>
 template0  | postgres | UTF8     | libc            | en_US.utf8 | en_US.utf8 |>
            |          |          |                  |             |             |>
 template1  | postgres | UTF8     | libc            | en_US.utf8 | en_US.utf8 |>
            |          |          |                  |             |             |>
(4 rows)
postgres@pgbox:/home/postgres/$ pg_ctl -D /var/tmp/dummy/ stop
postgres@pgbox:/home/postgres/$ rm -rf /var/tmp/dummy/
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage



You need to define your backup strategy

> e.g. a full backup every Sunday and incremental backups during the week

```
postgres@pgbox:/home/postgres/$ rm -rf backups/*
postgres@pgbox:/home/postgres/$ mkdir backups/sunday
postgres@pgbox:/home/postgres/$ pg_basebackup --pgdata=backups/sunday --checkpoint=fast
postgres@pgbox:/home/postgres/$ weekdays=('mon' 'tue' 'wed' 'thu' 'fri' 'sat')
postgres@pgbox:/home/postgres/$ for i in "${weekdays[@]}"
do
    mkdir -p backups/${i}
    pg_basebackup --incremental=backups/sunday/backup_manifest --pgdata=backups/${i}
done
postgres@pgbox:/home/postgres/$ ls -l backups/
postgres@pgbox:/home/postgres/$ du -sh backups/*
```

PostgreSQL 17 highlight - In-core incremental backups

Basic usage

If you want a full consistent backup for Wednesday

> Combine the full backup from Sunday with the incremental backup from Wednesday

```
postgres@pgbox:/home/postgres/$ pg_combinebackup -o /var/tmp/dummy backups/sunday
backups/wed
postgres@pgbox:/home/postgres/$ echo "port=8889" >> /var/tmp/dummy/postgresql.auto.conf
postgres@pgbox:/home/postgres/$ pg_ctl -D /var/tmp/dummy start -l /dev/null
postgres@pgbox:/home/postgres/$ psql -p 8889 -c "\1"
postgres@pgbox:/home/postgres/$ pg_ctl -D /var/tmp/dummy stop
postgres@pgbox:/home/postgres/$ rm -rf /var/tmp/dummy
```


WAL summaries



PostgreSQL 17 highlight - In-core incremental backups

WAL summaries

Short recap: WAL summaries are created in \$PGDATA/pg_wal

> The file system and the internal function give of course the same picture

```
postgres@pgbox:/home/postgres/$ ls -la $PGDATA/pg_wal/summaries
-rw----- 1 postgres postgres 9 17:36 0000000100000000050000280000000005040230.summary
-rw----- 1 postgres postgres 9 17:37 0000000100000000050402300000000006000028.summary
-rw----- 1 postgres postgres 9 17:37 0000000100000000060000280000000007000028.summary
-rw----- 1 postgres postgres 9 17:37 0000000100000000070000280000000008000028.summary
-rw----- 1 postgres postgres 9 17:38 0000000100000000080000280000000011000028.summary
...
postgres@pgbox:/home/postgres/$ psql -c "select * from pg_available_wal_summaries ();"
tli | start_lsn | end_lsn
-----+-----+-----
1 | 0/5000028 | 0/5040230
1 | 0/5040230 | 0/6000028
1 | 0/6000028 | 0/7000028
```

tli = timeline

WAL summaries can be inspected with "pg_wal_summary_contents"

```
postgres@pgbox:/home/postgres/$ psql -c "select * from pg_wal_summary_contents ( 1, '0/1C000028', '0/25000028' );"
```

relfilenode	reltablespace	reldatabase	relforknumber	relblocknumber	is_limit_block
1247	1663	5	0	14	f
1249	1663	5	0	54	f
1249	1663	5	0	16	f
1249	1663	5	0	55	f

> is_limit_block:

- > true indicates either that
 - > (a) the relation fork was truncated to the length given by relblocknumber within the relevant range of WAL records or
 - > (b) that the relation fork was created or dropped within the relevant range of WAL records
- > in such cases, relblocknumber will be zero

PostgreSQL 17 highlight - In-core incremental backups

WAL summaries

WAL summaries can be inspected with "pg_wal_summary_contents"

> This was the creation of the table

```
postgres@pgbox:/home/postgres/$ psql -c "select * from pg_wal_summary_contents ( 1, '0/330018A8', '0/33444398' ) where reldatabase = 24580 and relfilenode = 24581;"
```

relfilenode	reltablespace	reldatabase	relforknumber	relblocknumber	is_limit_block
24581	1663	24580	0	0	f
24581	1663	24580	0	0	t

(2 rows)

PostgreSQL 17 highlight - In-core incremental backups

WAL summaries

WAL summaries can be inspected with "pg_wal_summary_contents"

> This were the 1000 rows we inserted

```
postgres@pgbox:/home/postgres/$ psql -c "insert into t select * from
generate_series(1,1000)" x
postgres@pgbox:/home/postgres/$ psql -c "checkpoint"
postgres@pgbox:/home/postgres/$ psql -c "select * from pg_available_wal_summaries ();"
postgres@pgbox:/home/postgres/$ psql -c "select * from pg_wal_summary_contents ( 1,
'0/33444398', '0/33454018' ) where reldatabase = 24580 and relfilenode = 24581;"
relfilenode| reltablespace | reldatabase | relforknumber | relblocknumber | is_limit_block
-----+-----+-----+-----+-----+-----
      24581 |           1663 |       24580 |              0 |              0 | f
      24581 |           1663 |       24580 |              0 |              1 | f
      24581 |           1663 |       24580 |              0 |              2 | f
      24581 |           1663 |       24580 |              0 |              3 | f
      24581 |           1663 |       24580 |              0 |              4 | f
```

PostgreSQL 17 highlight - In-core incremental backups

WAL summaries

Get the state of the summarizer process

```
postgres@pgbox:/home/postgres/$ psql -c "select * from pg_get_wal_summarizer_state ();"
```

summarized_tli	summarized_lsn	pending_lsn	summarizer_pid
1	0/33456E60	0/33456F68	18380

(1 row)

Finally ...



PostgreSQL 17 highlight - In-core incremental backups

Finally

<https://www.postgresql.org/message-id/CA%2BTgmoZ8ZJPHmt67ijq7fWc%3D27xE1tfsakRwd%2B3SGjzAW0FpuA%40mail.gmail.com>

From: Robert Haas <robertmhaas(at)gmail(dot)com>

To: Tomas Vondra <tomas(dot)vondra(at)enterprisedb(dot)com>

Cc: Michael Paquier <michael(at)paquier(dot)xyz>, "pgsql-hackers(at)postgresql(dot)org" <pgsql-hackers(at)postgresql(dot)org>

Subject: Re: post-freeze damage control

> Anyway, I'm still not worried about this particular feature, and I'll keep doing the stress testing.

In all sincerity, I appreciate the endorsement. Basically what's been

scaring me about this feature is the possibility that there's some

incurable design flaw that I've managed to completely miss.

If it has some more garden-variety bugs, that's still pretty bad: people will

potentially lose data and be unable to get it back.



Please help testing!

Any questions?

Please do ask!



We would love to boost
your IT-Infrastructure
How about you?