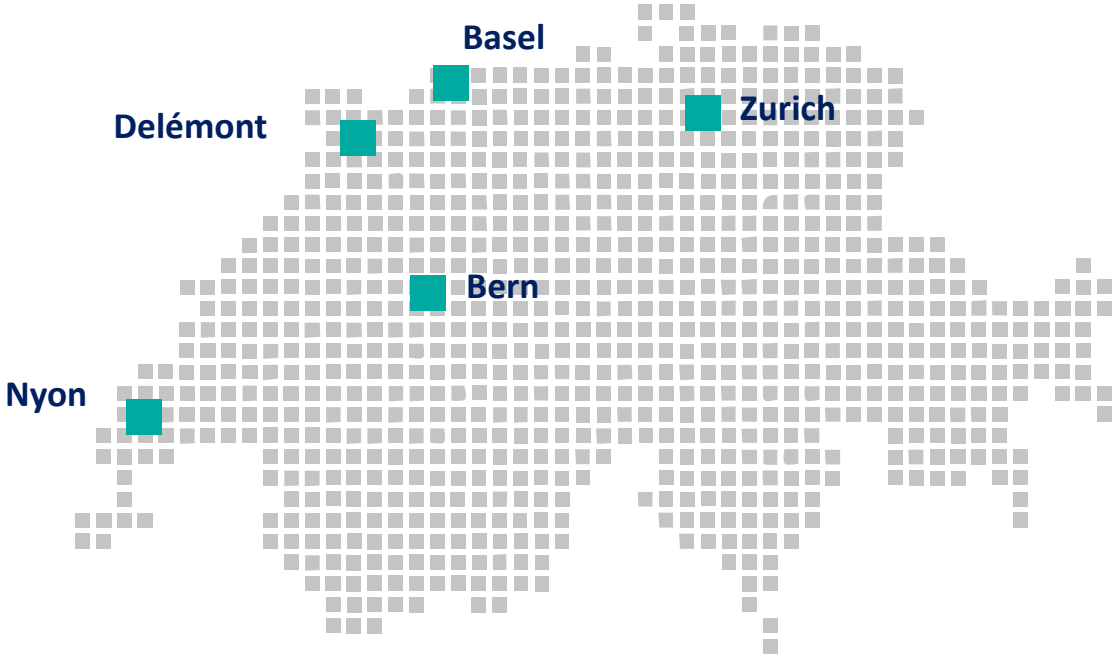
dbi services
by Sequotech

What we already know about PostgreSQL 19

WHO WE ARE

The company

- Founded in 2010
- 80 employees
- Specialized in data infrastructure and platforms
- Customers in Switzerland and all over Europe



Our offer

- Consulting
- Service Level Agreements
- Trainings
- License Management
- Products

ABOUT ME

Daniel Westermann

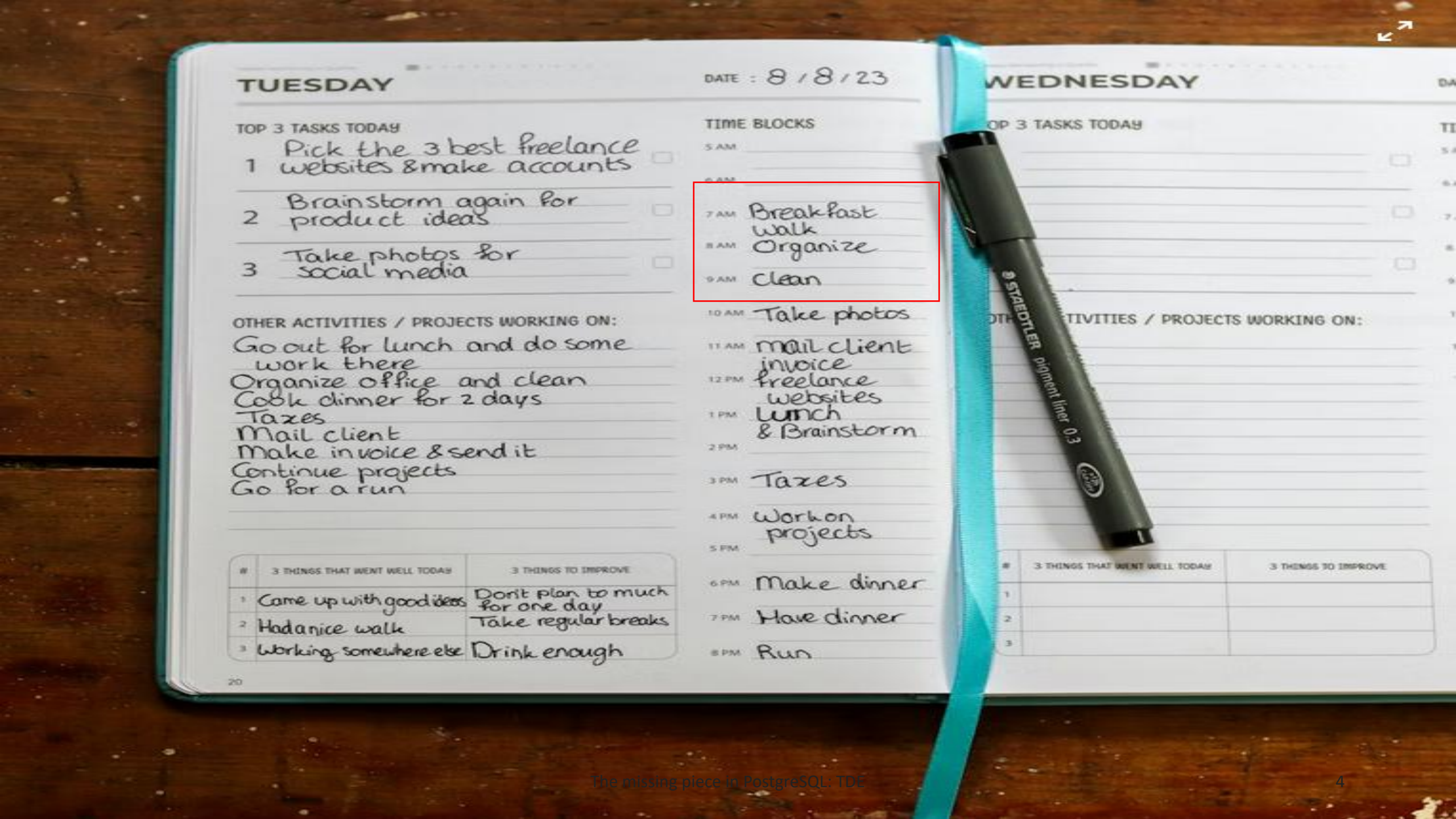
Principal Consultant

Technology Leader Open Infrastructure

I help with all things PostgreSQL

+41 79 927 2446

daniel.westermann@dbi-services.com


TUESDAY
DATE : 8 / 8 / 23
TOP 3 TASKS TODAY
1 Pick the 3 best freelance websites & make accounts
2 Brainstorm again for product ideas
3 Take photos for social media
OTHER ACTIVITIES / PROJECTS WORKING ON:
Go out for lunch and do some work there
Organize office and clean
Cook dinner for 2 days
Taxes
Mail client
Make invoice & send it
Continue projects
Go for a run
| 3 THINGS THAT WENT WELL TODAY | 3 THINGS TO IMPROVE
1 | Came up with good ideas | Don't plan to much for one day
2 | Had a nice walk | Take regular breaks
3 | Working somewhere else | Drink enough
20
TIME BLOCKS
5 AM
7 AM Breakfast
Walk
8 AM Organize
9 AM Clean
10 AM Take photos
11 AM mail client
invoice
12 PM freelance
websites
1 PM Lunch
& Brainstorm
2 PM
3 PM Taxes
4 PM Work on
projects
5 PM
6 PM Make dinner
7 PM Have dinner
8 PM Run
WEDNESDAY
STAEDTLER pigment liner 0.3

Quick Links

- About
- Governance
- Policies
- Feature Matrix
- Donate
- History
- Sponsors
  - Contributing
  - Financial
  - Servers
- Latest News
- Upcoming Events
  - Past events
- Press
- Licence

PostgreSQL 19 Beta 1 Released!

Posted on **2026-06-04** by PostgreSQL Global Development Group

PostgreSQL Project

The PostgreSQL Global Development Group announces that the first beta release of Postg
feature previews ahead of general availability, though some details of the release can chan

You can find information about all of the PostgreSQL 19 features and changes in the releas

https://www.postgresql.org/docs/19/release-19.html

In the spirit of the open source PostgreSQL community, we strongly encourage you to test
other issues. While we do not advise you to run beta versions in production environments,
this beta release.

Your testing and feedback help the community ensure that PostgreSQL 19 upholds our sta
source relational database. Please read more about our beta testing process and how you

https://www.postgresql.org/developer/beta/

E.1.3. Changes

Below you will find a detailed account of the changes between PostgreSQL 19 and the previous major release.

E.1.3.1. Server

E.1.3.1.1. Optimizer

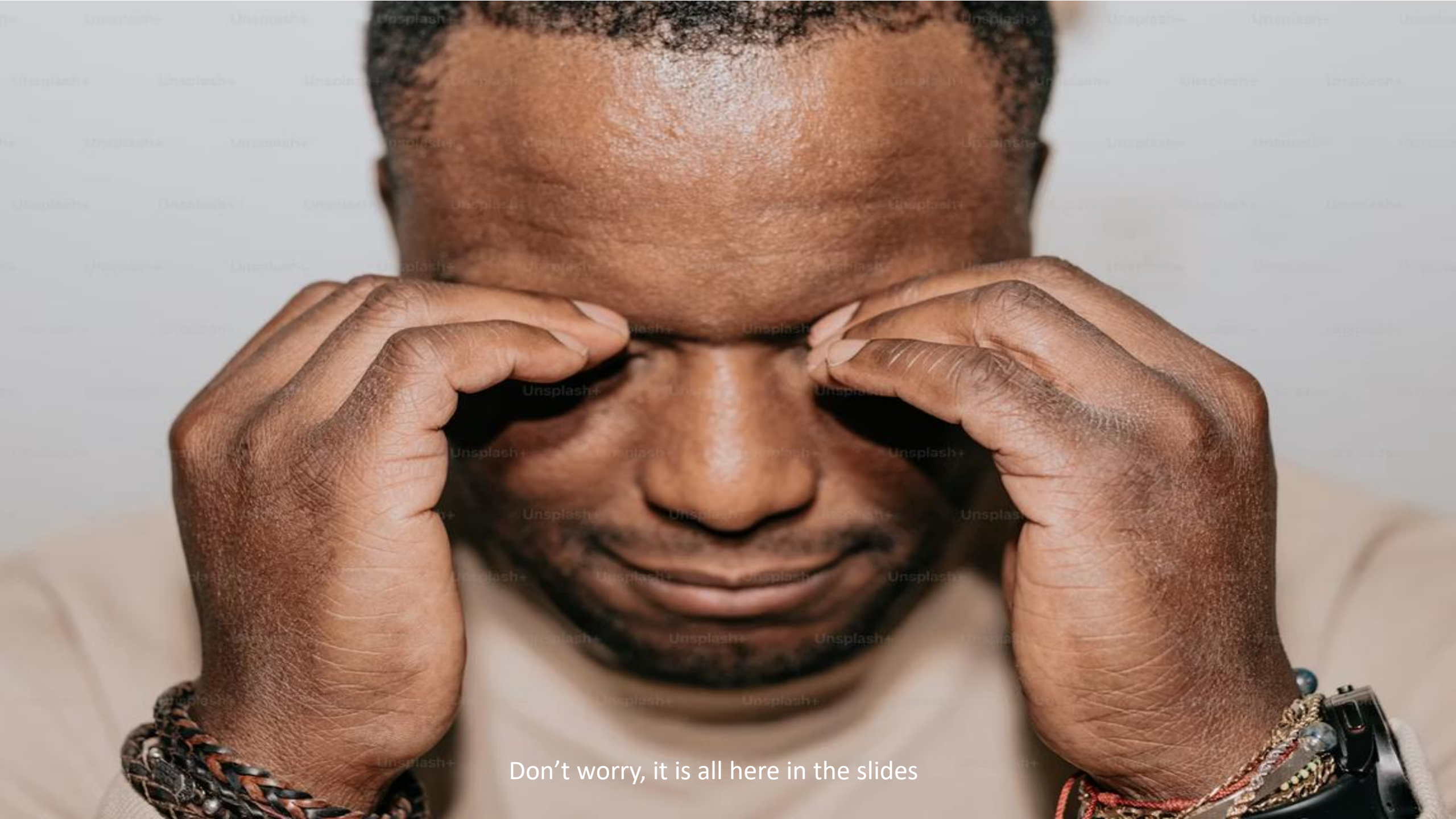
- Allow `NOT IN` clauses to be converted to more efficient anti-joins when NULLs are not present (Richard Guo) §
- Allow more `LEFT JOIN`s to be converted to `ANTI JOIN`s (Tender Wang, Richard Guo) §
- Allow use of Memoize for `ANTI JOIN`s with unique inner sides (Richard Guo) §
- Improve the planning of semijoins (Richard Guo) §
- Improve hash join's handling of tuples with `NULL` join keys (Tom Lane) §
- Convert `IS [NOT] DISTINCT FROM NULL` to `IS [NOT] NULL` during constant folding (Richard Guo) §

  The latter form is more easily optimized.
- Perform earlier constant folding of Var `IS [NOT] NULL` in the optimizer (Richard Guo) §

  This allows for later optimizations.
- Allow Append and MergeAppend to consider explicit incremental sorts (Richard Guo) §
- Allow some aggregate processing to be performed before joins (Richard Guo, Antonin Houska) § § §

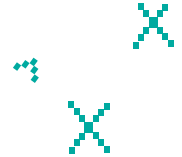
  This can reduce the number of rows needed to be processed.
- Allow negative values of `pg_aggregate.aggtransspace` to indicate unbounded memory usage (Richard Guo) §


No slides – Just demo


Don’t worry, it is all here in the slides

Last slide
(the next one)

What we already know about PostgreSQL 19



dbi services
by Sequotech

Wasn’t there something about hints? (https://wiki.postgresql.org/wiki/OptimizerHintsDiscussion)

> We are not interested in implementing hints in the exact ways they are commonly implemented on other databases. Proposals based on "because they've got them" will not be welcomed. If you have an idea that avoids the problems that have been observed with other hint systems, that could lead to valuable discussion.

Problems with existing Hint systems

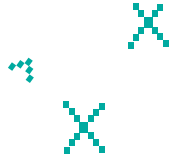
- Poor application code maintainability: hints in queries require massive refactoring.
- Interference with upgrades: today's helpful hints become anti-performance after an upgrade.
- Encouraging bad DBA habits slap a hint on instead of figuring out the real issue.
- Does not scale with data size: the hint that's right when a table is small is likely to be wrong when it gets larger.
- Failure to actually improve query performance: most of the time, the optimizer is actually right.
- Interfering with improving the query planner: people who use hints seldom report the query problem to the project.

Where Existing Hint Systems Benefit

- "One-shot" issues, such as annual or one-time reports, for which maintainability is not a concern
- Ability to "test" various execution paths in detail and see how the optimizer is working (or not)
- Optimizer failure
  - Implementation failure (known issues)
  - Theoretical failure (estimation limits, n^2 correlation problem)


WELL
DONE!

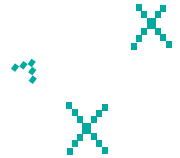
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

```
postgres@:/home/postgres/ $ export PGPORT=8888
postgres@:/home/postgres/ $ export PGDATA=/var/tmp/dummy
postgres@:/home/postgres/ $ export PATH=/u01/app/postgres/product/19/db_0/bin:$PATH
postgres@:/home/postgres/ $ pg_ctl -D /var/tmp/dummy stop
postgres@:/home/postgres/ $ rm -rf /var/tmp/dummy
postgres@:/home/postgres/ $ initdb -D /var/tmp/dummy
postgres@:/home/postgres/ $ pg_ctl -D /var/tmp/dummy start -l /dev/null

postgres@:/home/postgres/ $ psql -c "select version()"
postgres@:/home/postgres/ $ pgbench -i -s 10
                                        version
-----------------------------------------------------------------------------
 PostgreSQL 19beta1 on x86_64-linux, compiled by gcc-15.2.0, 64-bit
(1 row)
```

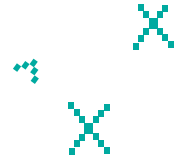
What we already know about PostgreSQL 19



dbi services
by Sequotech

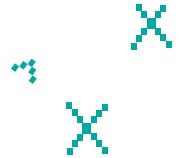

```
postgres@:/home/postgres/ $ psql -c "alter system set shared_preload_libraries=pg_plan_advice"
postgres@:/home/postgres/ $ pg_ctl restart -l /dev/null

postgres@:/home/postgres/ $ psql -c "explain (plan_advice)
                                     select * from pgbench_accounts a
                                     join pgbench_branches b on a.bid = b.bid;"
                                      QUERY PLAN
------------------------------------------------------------------------------
 Hash Join  (cost=1.23..30132.72 rows=1000000 width=461)
   Hash Cond: (a.bid = b.bid)
   ->  Seq Scan on pgbench_accounts a  (cost=0.00..26394.00 rows=1000000 width=97)
   ->  Hash  (cost=1.10..1.10 rows=10 width=364)
         ->  Seq Scan on pgbench_branches b  (cost=0.00..1.10 rows=10 width=364)
 Generated Plan Advice:
   JOIN_ORDER(a b)
   HASH_JOIN(b)
   SEQ_SCAN(a b)
   NO_GATHER(a b)
(10 rows)
```

What we already know about PostgreSQL 19




```
postgres@:/home/postgres/ $ psql -c "set pg_plan_advice.advice='JOIN_ORDER(b a)';
                                     explain (plan_advice)
                                     select * from pgbench_accounts a
                                     join pgbench_branches b on a.bid = b.bid;"
                                          QUERY PLAN
-------------------------------------------------------------------------------------------
 Merge Join  (cost=235428.11..252928.16 rows=1000000 width=461)
   Merge Cond: (b.bid = a.bid)
   ->  Sort  (cost=1.27..1.29 rows=10 width=364)
         Sort Key: b.bid
         ->  Seq Scan on pgbench_branches b  (cost=0.00..1.10 rows=10 width=364)
   ->  Materialize  (cost=235426.84..240426.84 rows=1000000 width=97)
         ->  Sort  (cost=235426.84..237926.84 rows=1000000 width=97)
               Sort Key: a.bid
               ->  Seq Scan on pgbench_accounts a  (cost=0.00..26394.00 rows=1000000 width=97)
 Supplied Plan Advice:
   JOIN_ORDER(b a) /* matched */
 Generated Plan Advice:
   JOIN_ORDER(b a)
   MERGE_JOIN_MATERIALIZE(a)
   SEQ_SCAN(b a)
   NO_GATHER(a b)
```

What we already know about PostgreSQL 19



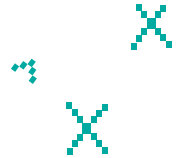
dbi services
by Sequotech
Confidentiality: Restricted

```
postgres@:/home/postgres/ $ psql -c "set pg_plan_advice.advice='HASH_JOIN(a)';
                                     explain (plan_advice)
                                     select * from pgbench_accounts a
                                     join pgbench_branches b on a.bid = b.bid;"
                                      QUERY PLAN
------------------------------------------------------------------------------------
 Hash Join  (cost=10000054519.00..10000081424.21 rows=1000000 width=461)
   Hash Cond: (b.bid = a.bid)
   ->  Seq Scan on pgbench_branches b  (cost=0.00..1.10 rows=10 width=364)
   ->  Hash  (cost=26394.00..26394.00 rows=1000000 width=97)
         ->  Seq Scan on pgbench_accounts a  (cost=0.00..26394.00 rows=1000000 width=97)
 Supplied Plan Advice:
   HASH_JOIN(a) /* matched */
 Generated Plan Advice:
   JOIN_ORDER(b a)
   HASH_JOIN(a)
   SEQ_SCAN(b a)
   NO_GATHER(a b)
(12 rows)
```

What we already know about PostgreSQL 19


```
postgres@:/home/postgres/ $ psql -c "set pg_plan_advice.advice='JOIN_ORDER(b x)';
                            explain (plan_advice)
                            select * from pgbench_accounts a
                            join pgbench_branches b on a.bid = b.bid;"
                                      QUERY PLAN
------------------------------------------------------------------------------
 Nested Loop  (cost=0.14..177655.02 rows=1000000 width=461)
   Disabled: true
   ->  Seq Scan on pgbench_accounts a  (cost=0.00..26394.00 rows=1000000 width=97)
   ->  Index Scan using pgbench_branches_pkey on pgbench_branches b
         Index Cond: (bid = a.bid)
 Supplied Plan Advice:
   JOIN_ORDER(b x) /* partially matched */
 Generated Plan Advice:
   JOIN_ORDER(a b)
   NESTED_LOOP_PLAIN(b)
   SEQ_SCAN(a)
   INDEX_SCAN(b public.pgbench_branches_pkey)
   NO_GATHER(a b)
```

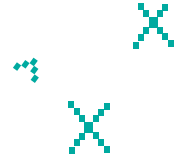
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

```
postgres@:/home/postgres/ $ psql -c "select name
                                     from pg_available_extensions where name like '%advice%'"
      name
-----------------
 pg_stash_advice
(1 row)
postgres@:/home/postgres/ $ psql -c "create extension pg_stash_advice"
postgres@:/home/postgres/ $ psql -c "\dx"
                              List of installed extensions
      Name       | Version |                  Description
-----------------+---------+-----------------------------------------
 pg_stash_advice | 1.0     | store and automatically apply plan advice
 plpgsql         | 1.0     | PL/pgSQL procedural language
```

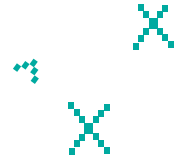
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

```
postgres@:/home/postgres/ $ psql -c "show compute_query_id"
 compute_query_id
------------------
 auto
(1 row)
postgres@:/home/postgres/ $ psql -c "alter system set
                                     shared_preload_libraries=pg_plan_advice,pg_stash_advice"
postgres@:/home/postgres/ $ pg_ctl restart -l /dev/null
postgres@:/home/postgres/ $ psql -c "show pg_stash_advice.persist"
pg_stash_advice.persist
------------------------
 on
(1 row)
postgres@:/home/postgres/ $ psql -c "\dfS pg_create_advice_stash"
                                List of functions
 Schema |          Name           | Result data type | Argument data types | Type
--------+-------------------------+------------------+---------------------+------
 public | pg_create_advice_stash | void             | stash_name text     | func
(1 row)
```

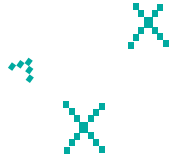
What we already know about PostgreSQL 19



dbi services
by Sequotech

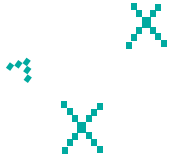

```
postgres@:/home/postgres/ $ psql -c "select * from pg_create_advice_stash('dummy');"
 pg_create_advice_stash
------------------------

(1 row)
postgres@:/home/postgres/ $ psql -c "select * from pg_get_advice_stashes()"
 stash_name | num_entries
------------+-------------
 dummy      |           0
(1 row)
```

What we already know about PostgreSQL 19




```
postgres@:/home/postgres/ $ psql -c "explain (plan_advice, verbose)
                                     select * from pgbench_accounts a
                                     join pgbench_branches b on a.bid = b.bid;"

 Hash Join  (cost=1.23..30132.72 rows=1000000 width=461)
   Output: a.aid, a.bid, a.abalance, a.filler, b.bid, b.bbalance, b.filler
   Inner Unique: true
   Hash Cond: (a.bid = b.bid)
   ->  Seq Scan on public.pgbench_accounts a  (cost=0.00..26394.00 rows=1000000 width=97)
         Output: a.aid, a.bid, a.abalance, a.filler
   ->  Hash  (cost=1.10..1.10 rows=10 width=364)
         Output: b.bid, b.bbalance, b.filler
         ->  Seq Scan on public.pgbench_branches b  (cost=0.00..1.10 rows=10 width=364)
               Output: b.bid, b.bbalance, b.filler
 Query Identifier: 5572457633751549O9
 Generated Plan Advice:
   JOIN_ORDER(a b)
   HASH_JOIN(b)
   SEQ_SCAN(a b)
   NO_GATHER(a b)
```

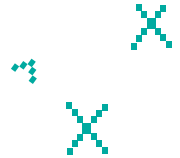
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "select * from
                            pg_set_stashed_advice('dummy'
                                                  , 55724576337515490 9, 'JOIN_ORDER(b a)');"
 pg_set_stashed_advice
-----------------------

(1 row)

postgres@:/home/postgres/ $ psql -c "select * from pg_get_advice_stash_contents('dummy');"
 stash_name |       query_id       |  advice_string
------------+----------------------+-----------------
 dummy      | 5572457633751549 09 | JOIN_ORDER(b a)
(1 row)
```

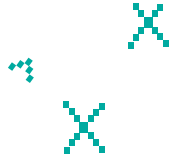
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

```
postgres@:/home/postgres/ $ psql -c "explain (plan_advice, verbose) select * from
                            pgbench_accounts a join pgbench_branches b on a.bid = b.bid;"
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Hash Join  (cost=1.23..30132.72 rows=1000000 width=461)
   Output: a.aid, a.bid, a.abalance, a.filler, b.bid, b.bbalance, b.filler
   Inner Unique: true
   Hash Cond: (a.bid = b.bid)
   ->  Seq Scan on public.pgbench_accounts a  (cost=0.00..26394.00 rows=1000000 width=97)
         Output: a.aid, a.bid, a.abalance, a.filler
   ->  Hash  (cost=1.10..1.10 rows=10 width=364)
         Output: b.bid, b.bbalance, b.filler
         ->  Seq Scan on public.pgbench_branches b  (cost=0.00..1.10 rows=10 width=364)
               Output: b.bid, b.bbalance, b.filler
 Query Identifier: 5572457633751549O9
 Generated Plan Advice:
   JOIN_ORDER(a b)
   HASH_JOIN(b)
   SEQ_SCAN(a b)
   NO_GATHER(a b)
```

What we already know about PostgreSQL 19



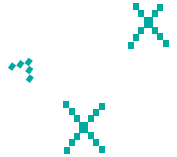
dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "show pg_stash_advice.stash_name"
 pg_stash_advice.stash_name
----------------------------

(1 row)

postgres@:/home/postgres/ $ psql -c "alter database postgres
                              set pg_stash_advice.stash_name='dummy';"
ALTER DATABASE
postgres@:/home/postgres/ $ psql -c "select * from pg_db_role_setting;"
 setdatabase | setrole |                setconfig
-------------+---------+-----------------------------------
           5 |       0 | {pg_stash_advice.stash_name=dummy}
(1 row)
```

What we already know about PostgreSQL 19


```
postgres@:/home/postgres/ $ psql -c "explain (plan_advice, verbose) select * from
                            pgbench_accounts a join pgbench_branches b on a.bid = b.bid;"
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Merge Join  (cost=235428.11..252928.16 rows=1000000 width=461)
…
     Sort  (cost=235426.84..237926.84 rows=1000000 width=97)
               Output: a.aid, a.bid, a.abalance, a.filler
               Sort Key: a.bid
               ->  Seq Scan on public.pgbench_accounts a  (cost=0.00..26394.00 rows=1000000
width=97)
                     Output: a.aid, a.bid, a.abalance, a.filler
 Query Identifier: 557245763375154909
 Supplied Plan Advice:
   JOIN_ORDER(b a) /* matched */
 Generated Plan Advice:
   JOIN_ORDER(b a)
   MERGE_JOIN_MATERIALIZE(a)
   SEQ_SCAN(b a)
   NO_GATHER(a b)
```

What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

```
postgres@:/home/postgres/ $ cat $PGDATA/pg_stash_advice.tsv
stash   dummy
entry   dummy   55724576337515490 9      JOIN_ORDER(b a)
postgres@:/home/postgres/ $ psql -c "select name,total_bytes,free_bytes
                                     from pg_catalog.pg_backend_memory_contexts
                                    where name like '%stash%';"
      name        | total_bytes | free_bytes
------------------+-------------+------------
 pg_stash_advice |       49232 |       7680
(1 row)
postgres@:/home/postgres/ $ psql -c "\dconfig *stash*"
     List of configuration parameters
            Parameter              | Value
-----------------------------------+-------
 pg_stash_advice.persist           | on
 pg_stash_advice.persist_interval | 30s
 pg_stash_advice.stash_name        | dummy
(3 rows)
```

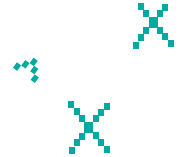
What we already know about PostgreSQL 19


```
postgres@:/home/postgres/ $ cd postgresql/; git show
5883ff30b02ceed3c5eabba4d9c09a7766f9a8fc; cd -

Author: Robert Haas <rhaas@postgresql.org>
Date:   Thu Mar 12 12:59:52 2026 -0400

    Add pg_plan_advice contrib module.

    Provide a facility that (1) can be used to stabilize certain plan choices
…
    Reviewed-by: Lukas Fittl <lukas@fittl.com>
    Reviewed-by: Jakub Wartak <jakub.wartak@enterprisedb.com>
    Reviewed-by: Greg Burd <greg@burd.me>
    Reviewed-by: Jacob Champion <jacob.champion@enterprisedb.com>
    Reviewed-by: Haibo Yan <tristan.yim@gmail.com>
    Reviewed-by: Dian Fay <di@nmfay.com>
    Reviewed-by: Ajay Pal <ajay.pal.k@gmail.com>
    Reviewed-by: John Naylor <johncnaylorls@gmail.com>
    Reviewed-by: Alexandra Wang <alexandra.wang.oss@gmail.com>
```

What we already know about PostgreSQL 19

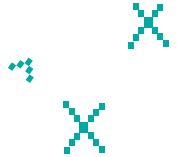

```
postgres@:/home/postgres/ $ cd postgresql/; git show
c10edb102ada607eb054bc9e23690109d86849ef; cd -

Author: Robert Haas <rhaas@postgresql.org>
Date:   Tue Apr 7 10:11:25 2026 -0400

    pg_stash_advice: Allow stashed advice to be persisted to disk.

    If pg_stash_advice.persist = true, stashed advice will be written to
    pg_stash_advice.tsv in the data directory, periodically and at
    shutdown. On restart, stash modifications are locked out until this
    file has been reloaded, but queries will not be, so there may be a
    short window after startup during which previously-stashed advice is
    not automatically applied.

    Author: Robert Haas <rhaas@postgresql.org>
    Co-authored-by: Lukas Fittl <lukas@fittl.com>
    Discussion: https://postgr.es/m/CA+Tgmob87qsWa-VugofU6epuV0H5XjWZGMbQas4Q-
ADKmvSyBg@mail.gmail.com
```

What we already know about PostgreSQL 19




```
postgres@:/home/postgres/ $ psql -c "\dconfig io_*work*"
 List of configuration parameters
         Parameter          | Value
----------------------------+-------
 io_max_workers             | 8
 io_min_workers             | 2
 io_worker_idle_timeout     | 1min
 io_worker_launch_interval | 100ms
(4 rows)
```

What we already know about PostgreSQL 19



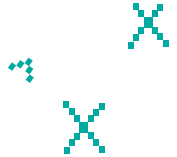
dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "\dconfig io_*work*"
 List of configuration parameters
         Parameter          | Value
----------------------------+-------
 io_max_workers             | 8
 io_min_workers             | 2
 io_worker_idle_timeout     | 1min
 io_worker_launch_interval | 100ms
(4 rows)

postgres@:/home/postgres/ $ ps -ef | grep postgres | grep worker | grep io | grep -v grep
postgres   26036   26035  0 08:58 ?        00:00:00 postgres: io worker 1
postgres   26037   26035  0 08:58 ?        00:00:00 postgres: io worker 0
```

What we already know about PostgreSQL 19


```
postgres@:/home/postgres/ $ ## second session
postgres@:/home/postgres/ $ watch "ps -ef | grep postgres | grep worker | grep -v grep"

postgres   26036   26035  0 08:58 ?        00:00:00 postgres: io worker 1
postgres   26037   26035  0 08:58 ?        00:00:00 postgres: io worker 0

postgres@:/home/postgres/ $ ## first session
postgres@:/home/postgres/ $ psql -c "insert into t select i, i::text, now() from
generate_series(1,20000000) i;"
postgres@:/home/postgres/ $ psql -c "select count(*) from t;"
postgres@:/home/postgres/ $ ## second session
postgres   26036   26035  0 08:58 ?        00:00:00 postgres: io worker 1
postgres   26037   26035  0 08:58 ?        00:00:00 postgres: io worker 0
postgres   26043   26035  0 08:58 ?        00:00:00 postgres: pg_stash_advice worker
postgres   27225   26035  0 09:59 ?        00:00:00 postgres: io worker 2
postgres   27226   26035  0 09:59 ?        00:00:00 postgres: io worker 3
```

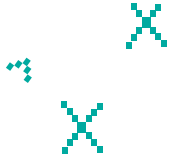
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "\dconfig io_*work*"
 List of configuration parameters
         Parameter          | Value
----------------------------+-------
 io_max_workers             | 8
 io_min_workers             | 2
 io_worker_idle_timeout     | 1min
 io_worker_launch_interval | 100ms
(4 rows)
```

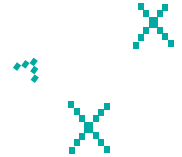
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

```
postgres@:/home/postgres/ $ cd postgresql; git show
d1c01b79d4ae90e52bf9db9c05c9de17b7313e85; cd -

Author: Thomas Munro <tmunro@postgresql.org>
Date:   Wed Apr 8 19:06:14 2026 +1200

    aio: Adjust I/O worker pool automatically.

    The size of the I/O worker pool used to implement io_method=worker was

      io_min_workers=2
      io_max_workers=8 (up to 32)
      io_worker_idle_timeout=60s
      io_worker_launch_interval=100ms

    Reviewed-by: Andres Freund <andres@anarazel.de>
    Reviewed-by: Dmitry Dolgov <9erthalion6@gmail.com>
    Reviewed-by: Nazir Bilal Yavuz <byavuz81@gmail.com>
```

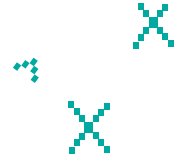
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "show data_checksums"
 data_checksums
----------------
 on
(1 row)

postgres@:/home/postgres/ $ pg_controldata | grep -i checksum
Latest checkpoint's data_checksum_version:1
Data page checksum version:               1
```

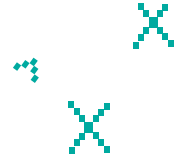
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ pg_checksums --pgdata=$PGDATA --disable
pg_checksums: error: cluster must be shut down

postgres@:/home/postgres/ $ psql -c "\dfS *checksums*"
                                List of functions
   Schema   |            Name             |                        Argument data types
------------+-----------------------------+-----------------------------------------------------
 pg_catalog | pg_disable_data_checksums   |
 pg_catalog | pg_enable_data_checksums    | cost_delay integer DEFAULT 0, cost_limit integer
DEFAULT 100
```

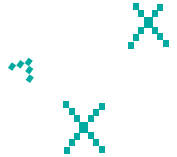
What we already know about PostgreSQL 19



dbi services
by Sequotech

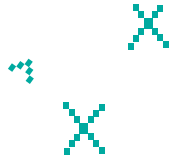

```
postgres@:/home/postgres/ $ pg_checksums --pgdata=$PGDATA --disable
pg_checksums: error: cluster must be shut down

postgres@:/home/postgres/ $ psql -c "\dfS *checksums*"
                                List of functions
   Schema   |            Name             |                        Argument data types
------------+-----------------------------+-----------------------------------------------------
 pg_catalog | pg_disable_data_checksums   |
 pg_catalog | pg_enable_data_checksums    | cost_delay integer DEFAULT 0, cost_limit integer
DEFAULT 100
```

What we already know about PostgreSQL 19




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

(1 row)

postgres@:/home/postgres/ $ psql -c "show data_checksums"
 data_checksums
----------------
 off
(1 row)

postgres@:/home/postgres/ $ pg_controldata | grep -i checksum
Latest checkpoint's data_checksum_version:0
Data page checksum version:               0
```

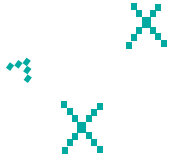
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ ## first session
postgres@:/home/postgres/ $ watch "ps -ef | grep checksum | grep -v watch"

postgres@:/home/postgres/ $ ## second session
postgres@:/home/postgres/ $ psql -c "select
                                     pg_enable_data_checksums(cost_delay=>1
                                                              ,cost_limit=>3000);"
postgres@:/home/postgres/ $ ## first session
…
postgres   28519   26035  0 10:23 ?        00:00:00 postgres: datachecksums launcher
postgres   28520   26035 26 10:23 ?        00:00:00 postgres: datachecksums worker
```

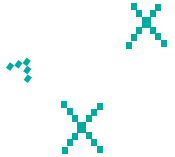
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ cd postgresql; git show
f19c0eccae9680f5785b11cdc58ef571998caec9; cd -

Author: Daniel Gustafsson <dgustafsson@postgresql.org>
Date:   Fri Apr 3 22:58:51 2026 +0200

    Online enabling and disabling of data checksums
…
    Author: Daniel Gustafsson <daniel@yesql.se>
    Author: Magnus Hagander <magnus@hagander.net>
    Co-authored-by: Tomas Vondra <tomas@vondra.me>
    Reviewed-by: Tomas Vondra <tomas@vondra.me>
    Reviewed-by: Andres Freund <andres@anarazel.de>
    Reviewed-by: Heikki Linnakangas <hlinnaka@iki.fi>
```

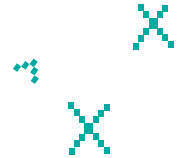
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ mkdir /var/tmp/tbs1
postgres@:/home/postgres/ $ psql -c "create tablespace tbs1 location '/var/tmp/tbs1'"
postgres@:/home/postgres/ $ psql -c "drop table t, tt«
postgres@:/home/postgres/ $ psql -c "create user u with login password 'u’”
postgres@:/home/postgres/ $ psql -c "create tablespace tbs1 owner u location
'/var/tmp/tbs1'"
postgres@:/home/postgres/ $ psql -c "create table t ( a int primary key generated always as
identity, b text ) tablespace tbs1;"
postgres@:/home/postgres/ $ psql -c "create table tt ( a int, b int references t(a))
tablespace tbs1"
postgres@:/home/postgres/ $ psql -c "\dfS *pg_get_*_ddl*"
postgres@:/home/postgres/ $ psql -c "select pg_get_database_ddl('postgres’)”
postgres@:/home/postgres/ $ psql -c "select pg_get_database_ddl('postgres','pretty','true')"
postgres@:/home/postgres/ $ psql -c "select pg_get_role_ddl('u')"
postgres@:/home/postgres/ $ psql -c "select pg_get_tablespace_ddl('tbs1')"
```

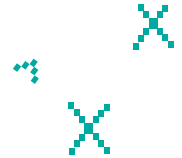
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "create table x ( a int primary key, b text );"
postgres@:/home/postgres/ $ psql -c "insert into x select i, md5(i::text) from
generate_series(1,1000000) i;"
postgres@:/home/postgres/ $ psql -c "copy x to '/var/tmp/x';"
postgres@:/home/postgres/ $ head /var/tmp/x
postgres@:/home/postgres/ $ psql -c "copy x to '/var/tmp/x' with (format json);"
postgres@:/home/postgres/ $ head /var/tmp/x
postgres@:/home/postgres/ $ psql -c "copy (select a from x) to '/var/tmp/x' with (format
json);"
postgres@:/home/postgres/ $ head /var/tmp/x
```

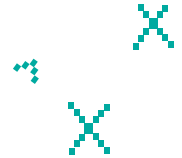
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "create table y ( a int primary key, b text );"
postgres@:/home/postgres/ $ psql -c "insert into y select i, md5(i::text) from
generate_series(1,10000000) i;"
postgres@:/home/postgres/ $ ## second session
postgres@:/home/postgres/ $ psql
select count(*) from y;
\watch
## first session
postgres@:/home/postgres/ $ psql -c "vacuum full y;"
postgres@:/home/postgres/ $ psql -c "cluster y using y_pkey;"
postgres@:/home/postgres/ $ psql -c "repack (concurrently) t;"
postgres@:/home/postgres/ $ psql -c "repack (concurrently) y using index y_pkey;"
postgres@:/home/postgres/ $ cd postgresql; git show
ac58465e0618941842439eb3f5a2cf8bebd5a3f1; cd -
    Author: Antonin Houska <ah@cybertec.at>
    Reviewed-by: Mihail Nikalayeu <mihailnikalayeu@gmail.com>
    Reviewed-by: Álvaro Herrera <alvherre@kurilemu.de>
    Reviewed-by: Robert Treat <rob@xzilla.net>
    Reviewed-by: Euler Taveira <euler@eulerto.com>
    Reviewed-by: Matheus Alcantara <matheusssilv97@gmail.com>
    Reviewed-by: Junwang Zhao <zhjwpku@gmail.com>
    Reviewed-by: jian he <jian.universality@gmail.com>
```

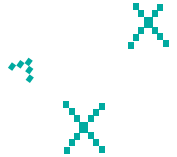
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ pg_basebackup --pgdata=/var/tmp/dummy2 --write-recovery-conf --checkpoint=fast
postgres@:/home/postgres/ $ echo "port=9999" >> /var/tmp/dummy2/postgresql.auto.conf
postgres@:/home/postgres/ $ chmod 700 /var/tmp/dummy2/
postgres@:/home/postgres/ $ pg_ctl -D /var/tmp/dummy2/ start -l /dev/null
postgres@:/home/postgres/ $ psql -p 9999 -c "select pg_is_in_recovery()"
postgres@:/home/postgres/ $ psql -c "select usename,sent_lsn,write_lsn,flush_lsn,replay_lsn from pg_stat_replication;"
postgres@:/home/postgres/ $ psql -p 9999 -c "select * from pg_wal_replay_pause();"
postgres@:/home/postgres/ $ psql -p 9999 -c "select * from pg_is_wal_replay_paused();"
postgres@:/home/postgres/ $ psql -c "create table t(a int)"
postgres@:/home/postgres/ $ psql -c "insert into t values(1)"
postgres@:/home/postgres/ $ psql -c "select pg_current_wal_insert_lsn();"
postgres@:/home/postgres/ $ psql -p 9999 -c "WAIT FOR LSN '3/2B0195E8'"
## second session
postgres@:/home/postgres/ $ psql -p 9999 -c "select * from pg_wal_replay_resume();"
```

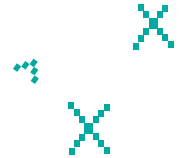
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql
show search_path
\set PROMPT1 '%/%R%x%..%S..# ';
set search_path='xxxxx';
\q
```

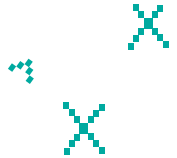
What we already know about PostgreSQL 19



dbi services
by Sequotech


```
postgres@:/home/postgres/ $ psql -c "alter system set logging_collector=on"
postgres@:/home/postgres/ $ psql -c "alter system set log_autoanalyze_min_duration = '1ms';"
postgres@:/home/postgres/ $ pg_ctl restart -l /dev/null
postgres@:/home/postgres/ $ psql -c "create table o ( a int , b text );"
## second session
postgres@:/home/postgres/ $ tail -f $PGDATA/log/...
## first session
postgres@:/home/postgres/ $ psql -c "insert into o select i, i::text from
generate_series(1,1000000) i;"
```

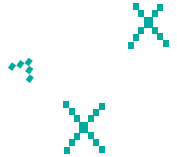
What we already know about PostgreSQL 19



dbi services
by Sequotech

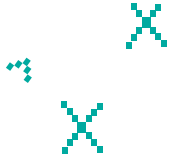

```
postgres@:/home/postgres/ $ psql -c "create table q ( a int primary key generated always as identity, b text)"
postgres@:/home/postgres/ $ psql -c "insert into q (b) values ('aaaa')"
postgres@:/home/postgres/ $ psql -c "insert into q (b) values ('bbbb')"
postgres@:/home/postgres/ $ pg_basebackup --pgdata=/var/tmp/dummy3 --write-recovery-conf --checkpoint=fast
postgres@:/home/postgres/ $ echo "port=9999" >> /var/tmp/dummy3/postgresql.auto.conf
postgres@:/home/postgres/ $ pg_createsubscriber --all --pgdata=/var/tmp/dummy3 --subscriber-port=9999 --publisher-server="host=localhost,port=8888"
postgres@:/home/postgres/ $ pg_ctl --pgdata=/var/tmp/dummy3 start
postgres@:/home/postgres/ $ psql -p 9999 -c "select * from q"
postgres@:/home/postgres/ $ psql -p 8888 -c "insert into q (b) values('cccc');"
postgres@:/home/postgres/ $ psql -p 9999 -c "select * from q"
postgres@:/home/postgres/ $ psql -p 8888 -c "\x" -c "select * from pg_sequences;"
postgres@:/home/postgres/ $ psql -p 9999 -c "\x" -c "select * from pg_sequences;"
postgres@:/home/postgres/ $ psql -p 8888 -c "\x" -c "select * from pg_publication;"
```

What we already know about PostgreSQL 19




```
postgres@:/home/postgres/ $ psql -p 8888 -c "create publication pubseq for all sequences;"
postgres@:/home/postgres/ $ psql -p 8888 -c "\x" -c "select * from pg_publication;"
postgres@:/home/postgres/ $ psql -p 9999 -c "create subscription subseq connection
'host=localhost port=8888' publication pubseq"
postgres@:/home/postgres/ $ psql -p 9999 -c "select * from pg_subscription_rel;"
postgres@:/home/postgres/ $ psql -p 9999 -c "\x" -c "select * from pg_sequences"
postgres@:/home/postgres/ $ psql -p 8888 -c "insert into q (b) values ('eeee')"
postgres@:/home/postgres/ $ psql -p 8888 -c "insert into q (b) values ('ffff')"
postgres@:/home/postgres/ $ psql -p 8888 -c "\x" -c "select * from pg_sequences"
postgres@:/home/postgres/ $ psql -p 9999 -c "\x" -c "select * from pg_sequences"
postgres@:/home/postgres/ $ psql -p 9999 -c "\x" -c "alter subscription subseq refresh
sequences"
postgres@:/home/postgres/ $ psql -p 9999 -c "\x" -c "select * from pg_sequences"
```

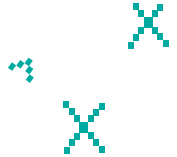
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

```
## https://www.cybertec-postgresql.com/en/handling-graphs-with-sql-pgq-in-postgresql/
postgres@:/home/postgres/ $ psql -c "\h create property graph"
postgres@:/home/postgres/ $ psql -c "create table person ( id int  primary key
                              , name text not null
                              , age int  not null
                              , city text not null);"
postgres@:/home/postgres/ $ psql -c "create Table knows ( a int NOT NULL REFERENCES
person(id)
                             , b int NOT NULL REFERENCES person(id)
                             , since int NOT NULL
                             , PRIMARY KEY (a, b));"
postgres@:/home/postgres/ $ psql -c "insert into person values (1, 'Alice', 30, 'Berlin')
                                    ,(2, 'Bob',   25, 'Berlin')
                                    ,(3, 'Carol', 35, 'Paris')
                                    ,(4, 'Dan',   28, 'Paris')
                                    ,(5, 'Eve',   40, 'London')
                                    ,(6, 'Frank', 33, 'London');"
```

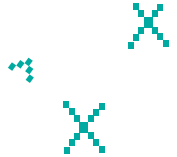
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

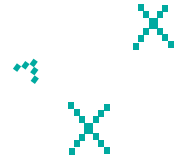
```
postgres@:/home/postgres/ $ psql -c "insert into knows VALUES  (1, 2, 2018), (2, 1, 2019)
                                    ,(1, 3, 2020), (2, 3, 2020), (3, 2, 2021)
                                    ,(3, 4, 2021), (4, 5, 2022)
                                    , (5, 6, 2019), (6, 1, 2023);"
postgres@:/home/postgres/ $ psql -c "create property graph social
        vertex tables ( person KEY (id) label person
                        properties (id, name, age, city)
                      )
        edge tables (
        knows
            source      KEY (a) references person (id)
            destination KEY (b) references person (id)
            label knows properties (since)
        );"
postgres@:/home/postgres/ $ psql -c "select name from graph_table (social
         match (p IS person)
         columns (p.name))
       order by name;"
```

What we already know about PostgreSQL 19




```
postgres@:/home/postgres/ $ psql -c "select * from graph_table (social
          match (p IS person )
        columns (p.id, p.name)
        );"
## Who knows whom?
postgres@:/home/postgres/ $ psql -c "select *
           from graph_table (social
          match (p is person )-[is knows]->(p2 is person)
         columns (p.id, p.name, p2.id, p2.name)
         )
         order by 1, 2, 3;"
```

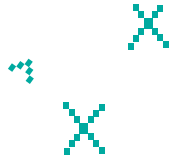
What we already know about PostgreSQL 19



dbi services
by Sequotech
Confidentiality: Restricted

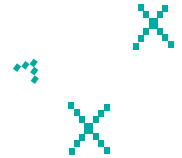
```
postgres@:/home/postgres/ $ psql -c "drop table t"
postgres@:/home/postgres/ $ psql -c "create table t ( a int, b text ) partition by list
(b);"
postgres@:/home/postgres/ $ psql -c "\d+ t"
postgres@:/home/postgres/ $ psql -c "create table t_p1 partition of t for values in ('a');"
postgres@:/home/postgres/ $ psql -c "create table t_p2 partition of t for values in ('b');"
postgres@:/home/postgres/ $ psql -c "create table t_p3 partition of t for values in ('c');"
postgres@:/home/postgres/ $ psql -c "create table t_p4 partition of t for values in ('d');"
postgres@:/home/postgres/ $ psql -c "\d+ t"
postgres@:/home/postgres/ $ psql -c "insert into t select i, 'a' from generate_series(1,100)
i;"
postgres@:/home/postgres/ $ psql -c "insert into t select i, 'b' from
generate_series(101,200) i;"
postgres@:/home/postgres/ $ psql -c "insert into t select i, 'c' from
generate_series(201,300) i;"
postgres@:/home/postgres/ $ psql -c "insert into t select i, 'd' from
generate_series(301,400) i;"
```

What we already know about PostgreSQL 19




```
postgres@:/home/postgres/ $ psql -c "select count(*) from t_p1;"
postgres@:/home/postgres/ $ psql -c "select count(*) from t_p2;"
postgres@:/home/postgres/ $ psql -c "select count(*) from t_p3;"
postgres@:/home/postgres/ $ psql -c "select count(*) from t_p4;"
postgres@:/home/postgres/ $ psql -c "alter table t merge partitions (t_p1, t_p2) into
t_p12;"
postgres@:/home/postgres/ $ psql -c "\d+ t"
postgres@:/home/postgres/ $ psql -c "alter table t split partition t_p12 into ( partition
t_p1 for values in ('a'), partition t_p2 for values in ('b'));"
postgres@:/home/postgres/ $ psql -c "\d+ t"
```

What we already know about PostgreSQL 19



dbi services
by Sequotech


```
## https://neon.com/postgresql/postgresql-19/temporal-data-operations
postgres@:/home/postgres/ $ psql -c "create extension if not exists btree_gist;"
postgres@:/home/postgres/ $ psql -c "create table employee_salaries ( employee_id int not
null
                                     , valid_range daterange not null
                                     , salary decimal(10,2) not null
                                     , department varchar(50)
                                     , primary Key (employee_id, valid_range without
overlaps)
                                    );"
postgres@:/home/postgres/ $ psql -c "insert into employee_salaries values (101,
daterange('2024-01-01', '2026-01-01'), 85000, 'Engineering');"
## Promotion effective July 2025
postgres@:/home/postgres/ $ psql -c "update employee_salaries for portion Of valid_range
FROM '2025-07-01' to '2026-01-01'
         set salary = 95000
         where employee_id = 101;"
postgres@:/home/postgres/ $ psql -c "select * from employee_salaries"
```

dbi services
by Sequotech

Confidentiality: Restricted

THANK YOU!

ANY QUESTIONS? PLEASE, DO ASK

- info@dbi-services.com
- www.dbi-services.com

OUR OFFICES

Basel | Bern | Delémont | Nyon | Zurich