Major Features: Postgres 16

BRUCE MOMJIAN

POSTGRESQL is an open-source, full-featured relational database. This presentation gives an overview of the Postgres 16 release.

https://momjian.us/presentations

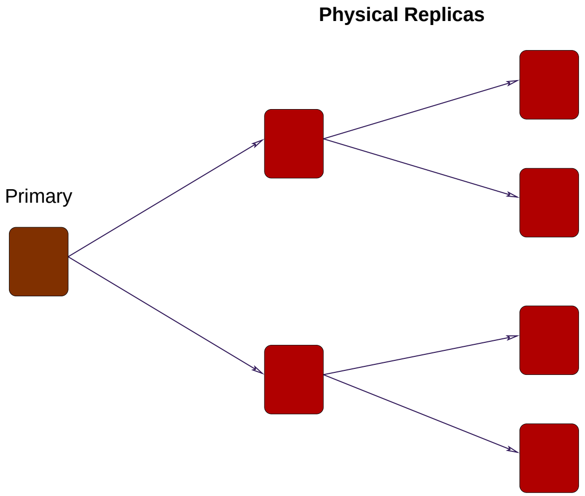
Last updated: August 2023

Postgres 16 Feature Outline

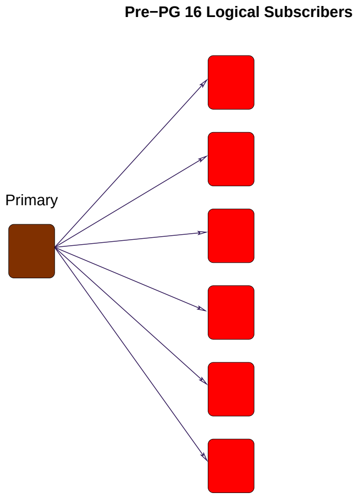
1. Logical replication from standby servers
2. Allow logical replication cycles
3. Role membership control
4. Allow indexes to use *date_trunc()*
5. Record statistics on the last sequential and index scans
6. Allow monitoring of I/O statistics
7. CPU vectorization
8. Allow libpq load balancing and control of authentication

Full item list at `https://momjian.us/pgsql_docs/release-16.html` and `https://www.postgresql.org/docs/16/release-16.html`.

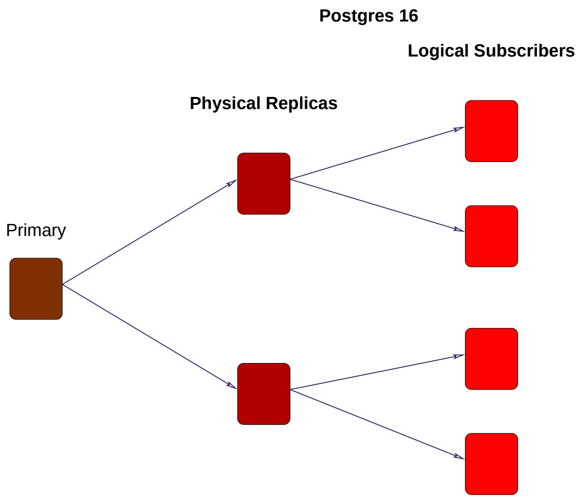
1. Logical Replication from Standby Servers



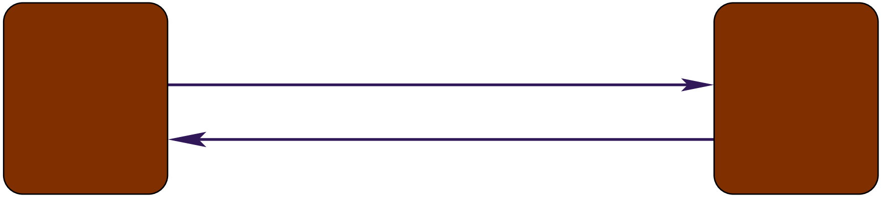
Pre-Postgres 16 Supports Only Primary-Sourced Logical Subscribers



Postgres 16 Supports Standby-Sourced Logical Subscribers



PG 16 Logical Subscribers on the Same Table



This is done by creating subscriptions with *slotname = none*, which causes only logical replication records with no slotname to be sent. This prevents records that arrived from other servers from being sent. There is no conflict resolution.

3. Role Membership Control

Role membership give two advantages:

- The ability to automatically INHERIT permissions of member roles
- The ability to execute queries as the member role via SET ROLE
- `https://www.postgresql.org/docs/current/role-membership.html`

Postgres 16 adds:

- Allow roles to INHERIT from only some members
- Allow members to be added to roles without giving them SET ROLE permission via SET ROLE FALSE
- Allow members to be added to roles if they have ADMIN permission on the role; previously create role permission was required
- Allow users who create roles to be automatically given INHERIT or SET ROLE abilities to the new role via *createrole_self_grant*
- Add *psql \drg* command to show role members and their INHERIT and SET ROLE permissions

Inheritance in Pre-Postgres 16

```
-- create role
CREATE ROLE a1;
CREATE ROLE b INHERIT;
GRANT a1 TO b;

-- create table
GRANT CREATE ON SCHEMA public TO a1;
SET ROLE a1;
CREATE TABLE a1_test (x INTEGER);
RESET ROLE;

-- SELECT table as 'b'
SET ROLE b;
SELECT * FROM a1_test;
 x
---

RESET ROLE;

-- change role inheritance status
ALTER USER b NOINHERIT;

-- SELECT table as 'b'
SET ROLE b;
SELECT * FROM a1_test;
ERROR:  permission denied for table a1_test
RESET ROLE;
```

Inheritance in Posgres 16

```
-- create role
CREATE ROLE a1;
CREATE ROLE b INHERIT;
GRANT a1 TO b;

-- create table
GRANT CREATE ON SCHEMA public TO a1;
SET ROLE a1;
CREATE TABLE a1_test (x INTEGER);
RESET ROLE;

-- SELECT table as 'b'
SET ROLE b;
SELECT * FROM a1_test;
 x
---

RESET ROLE;

-- change role inheritance status
ALTER USER b NOINHERIT;

-- SELECT table as 'b'
SET ROLE b;
SELECT * FROM a1_test;
 x
---

RESET ROLE;
```

Inheritance in Postgres 16

```
-- create roles
CREATE ROLE a2;
CREATE ROLE a3;
CREATE ROLE a4;

-- grant membership
GRANT a2 TO b;
GRANT a3 TO b WITH INHERIT TRUE;
GRANT a4 TO b WITH INHERIT FALSE;

\drg
              List of role grants
 Role name | Member of |    Options     | Grantor
-----------+-----------+----------------+----------
 b         | a1        | INHERIT, SET | postgres
 b         | a2        | SET          | postgres
 b         | a3        | INHERIT, SET | postgres
 b         | a4        | SET          | postgres
```

4. Allow Indexes to Use *date_trunc()*

```
CREATE TABLE trunc_test (x TIMESTAMP WITH TIME ZONE);

CREATE INDEX i_trunc_test ON trunc_test (date_trunc('month', x));
ERROR:  functions in index expression must be marked IMMUTABLE

CREATE INDEX i_trunc_test ON trunc_test (date_trunc('month', x, 'America/New_York'));

\d trunc_test
                     Table "public.trunc_test"
 Column |           Type           | Collation | Nullable | Default
--------+--------------------------+-----------+----------+---------
 x      | timestamp with time zone |           |          |
Indexes:
    "i_trunc_test" btree (date_trunc('month'::text, x, 'America/New_York'::text))
```

5. Record Statistics on the Last Sequential and Index Scans

```
\d pg_stat_user_tables
                      View "pg_catalog.pg_stat_user_tables"
       Column        |           Type           | Collation | Nullable | Default
---------------------+--------------------------+-----------+----------+---------
 relid               | oid                      |           |          |
 schemaname          | name                     |           |          |
 relname             | name                     |           |          |
 seq_scan            | bigint                   |           |          |
 last_seq_scan       | timestamp with time zone |           |          |
 seq_tup_read        | bigint                   |           |          |
 idx_scan            | bigint                   |           |          |
 last_idx_scan       | timestamp with time zone |           |          |
 idx_tup_fetch       | bigint                   |           |          |
 n_tup_ins           | bigint                   |           |          |
 n_tup_upd           | bigint                   |           |          |
 n_tup_del           | bigint                   |           |          |
 n_tup_hot_upd       | bigint                   |           |          |
 n_tup_newpage_upd   | bigint                   |           |          |
 n_live_tup          | bigint                   |           |          |
 n_dead_tup          | bigint                   |           |          |
 n_mod_since_analyze | bigint                   |           |          |
 n_ins_since_vacuum  | bigint                   |           |          |
 last_vacuum         | timestamp with time zone |           |          |
 last_autovacuum     | timestamp with time zone |           |          |
 last_analyze        | timestamp with time zone |           |          |
 last_autoanalyze    | timestamp with time zone |           |          |
 vacuum_count        | bigint                   |           |          |
 autovacuum_count    | bigint                   |           |          |
 analyze_count       | bigint                   |           |          |
 autoanalyze_count   | bigint                   |           |          |
```

6. Allow Monitoring of I/O Statistics

```
\d pg_stat_io
                          View "pg_catalog.pg_stat_io"
     Column      |           Type           | Collation | Nullable | Default
-----------------+--------------------------+-----------+----------+---------
 backend_type    | text                     |           |          |
 object          | text                     |           |          |
 context         | text                     |           |          |
 reads           | bigint                   |           |          |
 read_time       | double precision         |           |          |
 writes          | bigint                   |           |          |
 write_time      | double precision         |           |          |
 writebacks      | bigint                   |           |          |
 writeback_time  | double precision         |           |          |
 extends         | bigint                   |           |          |
 extend_time     | double precision         |           |          |
 op_bytes        | bigint                   |           |          |
 hits            | bigint                   |           |          |
 evictions       | bigint                   |           |          |
 reuses          | bigint                   |           |          |
 fsyncs          | bigint                   |           |          |
 fsync_time      | double precision         |           |          |
 stats_reset     | timestamp with time zone |           |          |
```

Types of Recorded Statistics

```
SELECT backend_type, object, context FROM pg_stat_io;
    backend_type     |    object     |  context
---------------------+---------------+-----------
 autovacuum launcher | relation      | bulkread
 autovacuum launcher | relation      | normal
 autovacuum worker   | relation      | bulkread
 autovacuum worker   | relation      | normal
 autovacuum worker   | relation      | vacuum
 client backend      | relation      | bulkread
 client backend      | relation      | bulkwrite
 client backend      | relation      | normal
 client backend      | relation      | vacuum
 client backend      | temp relation | normal
 background worker   | relation      | bulkread
 background worker   | relation      | bulkwrite
 background worker   | relation      | normal
 background worker   | relation      | vacuum
 background worker   | temp relation | normal
 background writer   | relation      | normal
 checkpointer        | relation      | normal
 standalone backend  | relation      | bulkread
 standalone backend  | relation      | bulkwrite
 standalone backend  | relation      | normal
 standalone backend  | relation      | vacuum
 startup             | relation      | bulkread
 startup             | relation      | bulkwrite
 startup             | relation      | normal
 startup             | relation      | vacuum
 walsender           | relation      | bulkread
 walsender           | relation      | bulkwrite
 …
```

Sample Statistics

```
SELECT *
FROM pg_stat_io
WHERE backend_type = 'client backend' AND
      object = 'relation' AND
      context = 'normal';
-[ RECORD 1 ]--+------------------------------
backend_type   | client backend
object         | relation
context        | normal
reads          | 3705962
read_time      | 0
writes         | 2216017
write_time     | 0
writebacks     | 0
writeback_time | 0
extends        | 20125
extend_time    | 0
op_bytes       | 8192
hits           | 81950274
evictions      | 3714568
reuses         | (null)
fsyncs         | 0
fsync_time     | 0
stats_reset    | 2023-08-23 21:30:36.252786-04
```

7. CPU Vectorization

- Allow JSON string parsing to use vector operations
- Add support for SSE2 (Streaming SIMD Extensions 2) vector operations on x86-64 architectures
- Add support for Advanced SIMD (Single Instruction Multiple Data) (NEON) instructions on ARM architectures

8. Allow libpq Load Balancing and Control of Authentication

- libpq connection option *load_balance_hosts=random* now randomly chooses a listed host
    - combined with *target_session_attrs=standby*, this allows load balancing among standby servers
- libpq option *require_auth* allows client to specify acceptable authentication methods
    - for example, *require_auth=scram-sha-256* prevents other password methods from being used

Conclusion

https://momjian.us/presentations

https://www.flickr.com/photos/thomasletholsen/