PERCONA®

When Logical Replication Goes Wrong

robert.bernier@percona.com

©2025 Percona

Databases run better with Percona

©2025 Percona

Overview

- whoami
- whoareu
- purpose of this talk
- talk breakdown

©2025 Percona

Who’s This Talk For …

DBA's and Developers creating and maintaining PostgreSQL Clusters implementing logical replication.

©2025 Percona

What's This Talk About ...

- Identify root causes to logical replication failures.
- Confirm the type of failure and its extent.
- Implement the optimal mitigation and resolution recovering from logical replication failures.
- Have a look at some different active-active configurations

©2025 Percona

Before You Start

What you need to know ...

- Linux
  - CLI
    - ssh (login using public keys)
    - netstat
    - su
    - sudo
  - bash scripting (basic stuff)

- DBMS knowledge
  - general administration i.e. create

    tables etc

- PostgreSQL (version 17.*)
  - Familiar administering postgres
    - User Account Creation
    - Creating
      - datacluster
      - database
      - tables
      - extensions
      - user accounts
      - assigning passwords
    - Configuration
      - authentication
      - permissions
      - basic tuning

©2025 Percona

Review

What is logical replication

- A method of replicating data objects and their changes, based upon their replication identity (usually a primary key).
- Logical replication uses a publish and subscribe model.
- Publishers “push” data to one or more subscribers.
- Subscribers “pull” data from one or more publishers.

- UPGRADE: Upgrade one version of PostgreSQL to another without downtime.
- SCALE OUT: Copies all or a selection of database tables.
- AGGREGATE: Accumulate changes from several sources
  ex: Data Warehouse
- INTEGRATE: Feeds database changes in real-time to other systems
- PROTECT: Provides backup or high availability for clusters

©2025 Percona

What You Need To Keep In Mind

Logical replication failures can occur …

- At Creation/Subscription time
- When performing DML operations
- When performing DDL operations

©2025 Percona

Monitoring/Debugging

Queries …

PUBLICATION SIDE

```
select * from pg_stat_activity;
select * from pg_publication_tables;
select * from pg_stat_replication;
select * from pg_get_replication_slots();
select * from pg_publication;
select * from pg_publication_tables;
```

SUBSCRIPTION SIDE

```
select * from pg_subscription;
select * from pg_stat_subscription;
```

Logging …

```
tail -f /var/log/postgresql/postgresql-17-pg1.log | grep -E 'ERROR|FATAL'
```

Zeroing …

```
> /var/log/postgresql/postgresql-17-pg1.log
```


©2025 Percona

Setup

The Servers: pg1, pg2

```
root@examples:~# pg_lsclusters
Ver Cluster Port Status Owner    Data directory              Log file
17  pg1     5432 online postgres /var/lib/postgresql/17/pg1 /var/log/postgresql/postgresql-17-pg1.log
17  pg2     5433 online postgres /var/lib/postgresql/17/pg2 /var/log/postgresql/postgresql-17-pg2.log
```

postgresql.auto.conf

```
psql -c "alter system set wal_level=logical;"
```


©2025 Percona

Scenarios

```
PART I
- Logical Replication Failures
    - Unsubscribed table
    - Duplicate key violation
    - Adding a new table
    - Updating postgres runtime parameters
    - Unique tuples
    - Adding a new column to a table
    - Skipping transactions (LSN)
    - Database copy failure

--------------------------
PART II
- Active-Active architectures
    - Daisy Chain
    - Star
    - Mesh
    - Xmas Tree
```


©2025 Percona

Example: Unsubscribed Table

©2025 Percona

Example: Unsubscribed Table

PG1

```
psql -p 5432 ex01 <<_eof1_
    create publication pub_ex01 for all tables;
_eof1_
```

PG2

```
psql -p 5433 ex01 <<_eof1_
    create subscription sub_ex01
        connection 'host=localhost port=5432 user=postgres password=postgres dbname=ex01'
        publication pub_ex01;
_eof1_
```

```
NOTICE:  database "ex01" does not exist, skipping
NOTICE:  database "ex01" does not exist, skipping
CREATE PUBLICATION
NOTICE:  created replication slot "sub_ex01" on publisher
CREATE SUBSCRIPTION
```


©2025 Percona

table t1 created on host pg1

PG1

```
psql -p 5432 ex01 <<_eof1_
    create table t1(c1 serial primary key,
                    c2 text default 'publication',
                    c3 timestamptz default clock_timestamp());

    insert into t1 values (default)
                          ,(default)
                          ,(default)
                  returning c1,c2,c3;
_eof1_
```

```
CREATE TABLE
 c1 |     c2      |              c3
----+-------------+-------------------------------
  1 | publication | 2025-09-15 20:09:58.804698+00
  2 | publication | 2025-09-15 20:09:58.804829+00
  3 | publication | 2025-09-15 20:09:58.804835+00
(3 rows)
```


©2025 Percona

table t1 missing on host pg2

```
2025-09-16 19:59:11.552 UTC [427] ERROR:  logical replication target relation "public.t1" does not exist
2025-09-16 19:59:11.552 UTC [427] CONTEXT:  processing remote data for replication origin "pg_16389" during message type "INSERT" in tran
saction 124135, finished at 0/911A4400
```


©2025 Percona

Mitigation;
table t1 created on host pg2

PG2

```
psql -p 5433 ex01 <<_eof1_
    create table t1(c1 serial primary key,
                    c2 text default 'subscription',
                    c3 timestamptz default clock_timestamp());
    table t1;
_eof1_
```

```
psql -p 5433 ex01 <<_eof1_
    alter subscription sub_ex01 refresh publication;
    select pg_sleep(2);
    table t1;
_eof1_
```

```
 c1 |     c2      |              c3
----+-------------+-------------------------------
  1 | publication | 2025-09-15 20:09:58.804698+00
  2 | publication | 2025-09-15 20:09:58.804829+00
  3 | publication | 2025-09-15 20:09:58.804835+00
(3 rows)
```


©2025 Percona

Example: Duplicate Key Violation

©2025 Percona

Example: Duplicate Key Violation

PG2

```
psql -p 5433 ex01 <<_eof1_
-- this fails
    insert into t1 values (default)
                          ,(default)
                          ,(default);
    table t1;
_eof1_
_eof_
```

```
ERROR:  duplicate key value violates unique constraint "t1_pkey"
DETAIL:  Key (c1)=(1) already exists.
 c1 |     c2      |              c3
----+-------------+-------------------------------
  1 | publication | 2025-09-15 20:09:58.804698+00
  2 | publication | 2025-09-15 20:09:58.804829+00
  3 | publication | 2025-09-15 20:09:58.804835+00
(3 rows)
```


©2025 Percona

Mitigation;
sequence updated

PG1

```
psql -p 5432 ex01 <<_eof1_
    select setval('t1_c1_seq', max(c1)+max(c1)%2) from t1;
    alter sequence t1_c1_seq increment by 2;

    insert into t1 values (default)
                          ,(default)
                          ,(default);
  \qecho ===== pg1: t1 =====
    table t1;
_eof1_
```

```
let A=\$(psql -Atp 5432 ex01 -c 'select max(c1) from t1')
let A=\$A+1
```

PG2

```
psql -p 5433 ex01 <<_eof1_
    alter sequence t1_c1_seq restart \$A increment by 2;

    insert into t1 values (default)
                          ,(default)
                          ,(default);
  \qecho ===== pg2: t1 =====
    table t1;
_eof1_
```

```
 setval
--------
      4
(1 row)

ALTER SEQUENCE
INSERT 0 3
===== pg1: t1 =====
 c1 |     c2      |              c3
----+-------------+-------------------------------
  1 | publication | 2025-09-15 20:09:58.804698+00
  2 | publication | 2025-09-15 20:09:58.804829+00
  3 | publication | 2025-09-15 20:09:58.804835+00
  6 | publication | 2025-09-15 20:23:24.377247+00
  8 | publication | 2025-09-15 20:23:24.377275+00
 10 | publication | 2025-09-15 20:23:24.377278+00
(6 rows)

ALTER SEQUENCE
INSERT 0 3
===== pg2: t1 =====
 c1 |      c2      |              c3
----+--------------+-------------------------------
  1 | publication  | 2025-09-15 20:09:58.804698+00
  2 | publication  | 2025-09-15 20:09:58.804829+00
  3 | publication  | 2025-09-15 20:09:58.804835+00
  6 | publication  | 2025-09-15 20:23:24.377247+00
  8 | publication  | 2025-09-15 20:23:24.377275+00
 10 | publication  | 2025-09-15 20:23:24.377278+00
 11 | subscription | 2025-09-15 20:23:24.449612+00
 13 | subscription | 2025-09-15 20:23:24.449648+00
 15 | subscription | 2025-09-15 20:23:24.44965+00
(9 rows)
```


©2025 Percona

An alternate solution ….

drop the PK/UNIQ constraint on the subscription side

©2025 Percona

Example: Adding New Table

©2025 Percona

Example: Adding New Table

PG1

```
create table t2(c1 serial primary key,
                c2 text default 'publication',
                c3 timestamptz default clock_timestamp());
```

```
select setval('t2_c1_seq', max(c1)+max(c1)%2) from t2;
alter sequence t2_c1_seq increment by 2;
```

```
insert into t1 values (default)
                      ,(default)
                      ,(default)
             returning c1,c2,c3;

insert into t2 values (default)
                      ,(default)
                      ,(default)
             returning c1,c2,c3;
```

```
 c1 |     c2      |              c3
----+-------------+-------------------------------
 18 | publication | 2025-09-15 20:33:03.069674+00
 20 | publication | 2025-09-15 20:33:03.069737+00
 22 | publication | 2025-09-15 20:33:03.069741+00
(3 rows)

INSERT 0 3
 c1 |     c2      |              c3
----+-------------+-------------------------------
  1 | publication | 2025-09-15 20:33:03.075535+00
  3 | publication | 2025-09-15 20:33:03.075625+00
  5 | publication | 2025-09-15 20:33:03.075631+00
(3 rows)
```


©2025 Percona

Mitigation;
add table,update sequence & subscribe

PG2

```
let A=\$(psql -Atp 5432 ex02 -c 'select max(c1) from t2')
let A=\$A+1

psql -p 5433 ex02 <<_eof1_
 \qecho ===== table t2 needs to be added and subscribed on pg2.ex02 =====
   create table t2(c1 serial primary key,
                   c2 text default 'subscription',
                   c3 timestamptz default clock_timestamp());

-- update the sequence on t2
   alter sequence t2_c1_seq restart \$A increment by 2;

-- replication resumes
    alter subscription sub_ex02 refresh publication;
    select pg_sleep(5);

    \qecho ===== table t1 =====
    table t1;
    \qecho ===== table t2 =====
    table t2;
_eof1_
```

```
===== table t1 =====
 c1 |      c2      |              c3
----+--------------+-------------------------------
  1 | publication  | 2025-09-15 20:09:58.804698+00
  2 | publication  | 2025-09-15 20:09:58.804829+00
  3 | publication  | 2025-09-15 20:09:58.804835+00
  6 | publication  | 2025-09-15 20:23:24.377247+00
  8 | publication  | 2025-09-15 20:23:24.377275+00
 10 | publication  | 2025-09-15 20:23:24.377278+00
 11 | subscription | 2025-09-15 20:23:24.449612+00
 13 | subscription | 2025-09-15 20:23:24.449648+00
 15 | subscription | 2025-09-15 20:23:24.44965+00
 12 | publication  | 2025-09-15 20:29:59.091371+00
 14 | publication  | 2025-09-15 20:29:59.091456+00
 16 | publication  | 2025-09-15 20:29:59.091458+00
 18 | publication  | 2025-09-15 20:33:03.069674+00
 20 | publication  | 2025-09-15 20:33:03.069737+00
 22 | publication  | 2025-09-15 20:33:03.069741+00
(15 rows)

===== table t2 =====
 c1 |     c2      |              c3
----+-------------+-------------------------------
  1 | publication | 2025-09-15 20:33:03.075535+00
  3 | publication | 2025-09-15 20:33:03.075625+00
  5 | publication | 2025-09-15 20:33:03.075631+00
(3 rows)
```


©2025 Percona

TIP ….

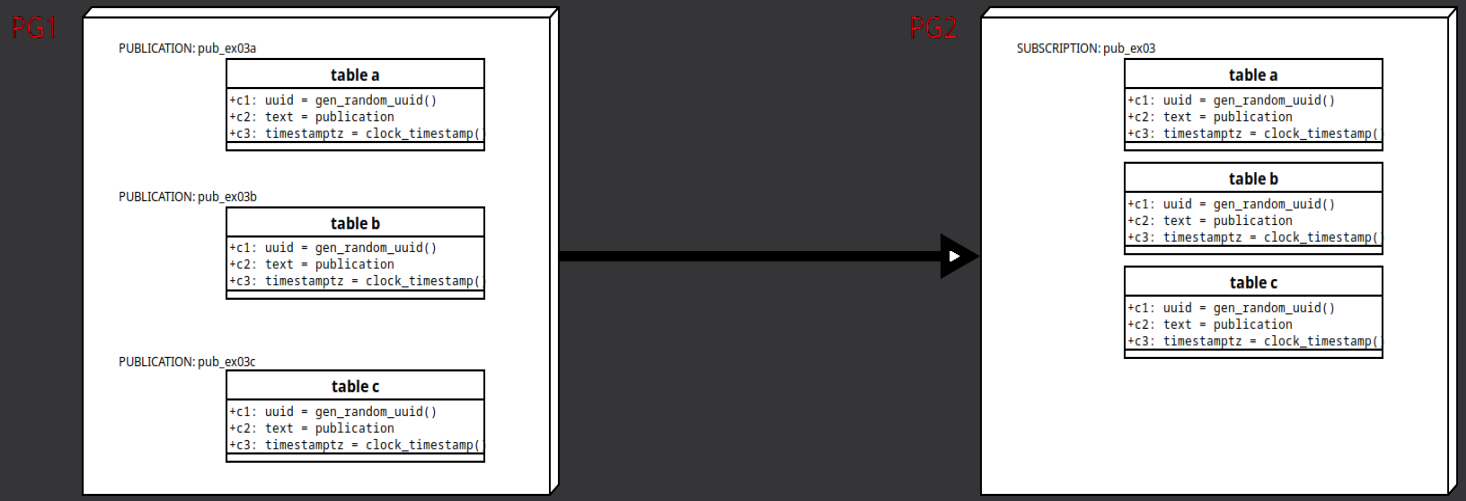
CREATE the relations on the SUBSCRIPTION 1st!

©2025 Percona

Example: Insufficient Workers

©2025 Percona

Example: Insufficient Workers



©2025 Percona

PERCONA

PG1

```
psql -p 5432 ex03 <<_eof1_

    create table a(c1 uuid default gen_random_uuid(),
                   c2 text default 'publication',
                   c3 timestamptz default clock_timestamp());

    create table b(like a including all);
    create table c(like a including all);

    \qecho ========= TABLE A =========
    insert into a values (default)
                         ,(default)
                         ,(default)
           returning c1,c2,c3;

    \qecho ========= TABLE B =========
    insert into b values (default)
                         ,(default)
                         ,(default)
            returning c1,c2,c3;

    \qecho ========= TABLE C =========
    insert into c values (default)
                         ,(default)
                         ,(default)
            returning c1,c2,c3;

    create publication pub_ex03a for table a;
    create publication pub_ex03b for table b;
    create publication pub_ex03c for table c;
_eof1_
```

```
========= TABLE A =========
                  c1                  |     c2      |              c3
--------------------------------------+-------------+-------------------------------
 90941226-a695-437d-8710-ba0532405512 | publication | 2025-09-15 20:37:31.472689+00
 34ecfc96-6182-47d4-9d0d-6f8ba57598e3 | publication | 2025-09-15 20:37:31.472731+00
 24e86a6e-63ce-4e27-8066-5a756d192460 | publication | 2025-09-15 20:37:31.472734+00
(3 rows)

INSERT 0 3
========= TABLE B =========
                  c1                  |     c2      |              c3
--------------------------------------+-------------+-------------------------------
 b2ca287c-341a-4027-9bcb-fbe48f512ec1 | publication | 2025-09-15 20:37:31.476285+00
 19eb5d96-08c2-4bb6-b151-6dd5f34f1eaa | publication | 2025-09-15 20:37:31.476324+00
 68fdb8fe-df2a-4f1d-ade3-6b4c496ffbfc | publication | 2025-09-15 20:37:31.476326+00
(3 rows)

INSERT 0 3
========= TABLE C =========
                  c1                  |     c2      |              c3
--------------------------------------+-------------+-------------------------------
 d066e270-0a60-409c-9714-e82fd5c92700 | publication | 2025-09-15 20:37:31.479752+00
 fd9e68c5-f505-4db2-b7a1-cd4862b14685 | publication | 2025-09-15 20:37:31.47982+00
 bcdc035b-1fac-472e-b4c5-3a4b404edcd0 | publication | 2025-09-15 20:37:31.479827+00
(3 rows)

INSERT 0 3
CREATE PUBLICATION
CREATE PUBLICATION
CREATE PUBLICATION
```


©2025 Percona

WARNING;
out of logical replication slots

PG2

```
psql -p 5433 ex03 <<_eof1_

    create table a(c1 uuid default gen_random_uuid(),
                   c2 text default 'subscription',
                   c3 timestamptz default clock_timestamp());

    create table b(like a including all);
    create table c(like a including all);

    create subscription sub_ex03
        connection 'host=localhost port=5432 user=postgres password=postgres dbname=ex03'
        publication pub_ex03a, pub_ex03b, pub_ex03c;
_eof1_
```

```
CREATE TABLE
CREATE TABLE
CREATE TABLE
NOTICE:  created replication slot "sub_ex03" on publisher
CREATE SUBSCRIPTION
Press Enter to continue, WARNING:  out of logical replication worker slots ...
```


©2025 Percona

Mitigation;
update runtime parameters

Steps:

1) Update both publication and subscription
2) Restart both pg1, pg2 servers

Example parameters:

```
max_worker_processes = '30'
max_replication_slots = '20'
max_wal_senders = '20'
```


©2025 Percona

Example: Deletes, Updates fail

©2025 Percona

Example: Deletes, Updates fail

```
psql -p 5432 ex03 <<_eof1_
    delete from a;
_eof1_
```

```
ERROR:  cannot delete from table "a" because it does not have a replica identity and publishes deletes
HINT:  To enable deleting from the table, set REPLICA IDENTITY using ALTER TABLE.
```


©2025 Percona

Recall;

inserts on a subscribed table does NOT need a Primary Key constraint

```
CREATE PUBLICATION name
    [ FOR ALL TABLES
      | FOR publication_object [, ... ] ]
    [ WITH ( publication_parameter [= value] [, ... ] ) ]
```

```
WITH ( publication_parameter [= value] [, ... ] )
```

This clause specifies optional parameters for a publication. The following parameters are supported:

`publish (string)`

This parameter determines which DML operations will be published by the new publication to the subscribers. The value is comma-separated list of operations. The allowed operations are `insert`, `update`, `delete`, and `truncate`. The default is to publish all actions, and so the default value for this option is `'insert, update, delete, truncate'`.

©2025 Percona

Mitigation;
updates, deletes requires unique tuples

Mitigation #1: Add Replica Identity

```
psql -p 5432 ex03 <<_eof1_
    alter table a replica identity full;
    delete from a;

    \qecho ========= published TABLE A =========
    table a;

    \c 'port=5433 dbname=ex03'
    \qecho ========= subscribed TABLE A =========
    table a;
_eof1_
```

```
ALTER TABLE
DELETE 6
========= published TABLE A =========
 c1 | c2 | c3
----+----+----
(0 rows)

You are now connected to database "ex03" as user "postgres" on host "localhost" (address "::1") at port "5433".
========= subscribed TABLE A =========
 c1 | c2 | c3
----+----+----
(0 rows)
```

Mitigation #2: Add Primary Key

```
psql -p 5433 ex03 <<_eof1_
   \qecho ========= subscribed TABLE B, add PK and now it works ====
    alter table b add primary key(c1);

    \c 'port=5432 dbname=ex03'
    \qecho ========= published TABLE B, delete works =========
    delete from b;
    table b;
_eof1_

echo "sleeping ..." ; sleep 4s

psql -p 5433 ex03 <<_eof1_
   \qecho ========= subscribed TABLE B, delete validated =========
    table b;
_eof1_
_eof_
```

```
################## STAGE 9: repeat DELETE TABLE ##################
========= subscribed TABLE B, add PK and now it works =========
ALTER TABLE
You are now connected to database "ex03" as user "postgres" on host "localhost" (address "::1") at port "5432".
========= published TABLE B, delete works =========
DELETE 0
 c1 | c2 | c3
----+----+----
(0 rows)

sleeping ...
========= subscribed TABLE B, delete validated =========
 c1 | c2 | c3
----+----+----
(0 rows)
```


©2025 Percona

Example: Adding Column

©2025 Percona

Example: Adding Column

PG1

```
create table a(c1 uuid default gen_random_uuid(),
               c2 text default 'publication',
               c3 timestamptz default clock_timestamp());

\qecho ========= TABLE A =========
insert into a values (default)
                     ,(default)
                     ,(default);

create publication pub_ex05 for table a;
```

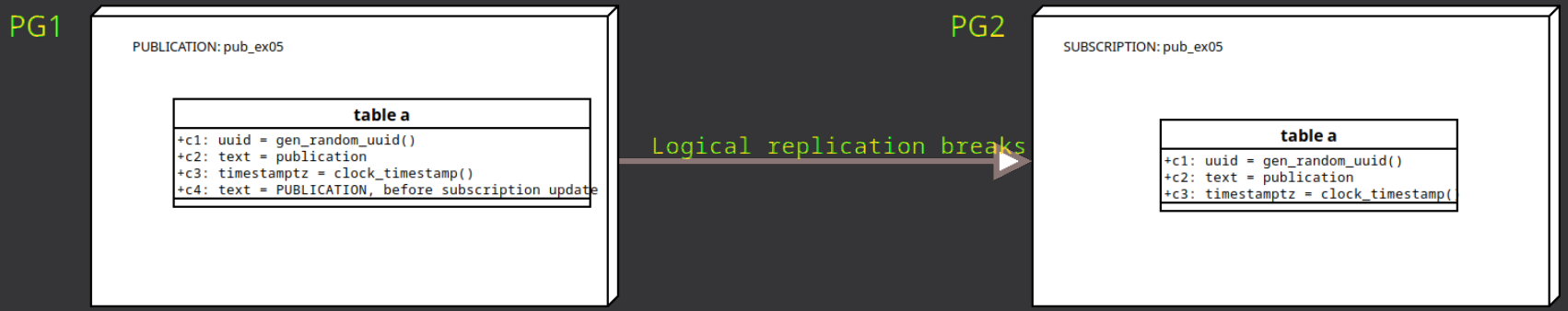
PG2

```
create table a(c1 uuid default gen_random_uuid(),
               c2 text default 'subscription',
               c3 timestamptz default clock_timestamp());

create subscription sub_ex05
    connection 'host=localhost port=5432 user=postgres password=postgres dbname=ex05'
    publication pub_ex05;

\qecho sleeping
select pg_sleep(5);

\qecho subscriber, port 5433
table a;
```

```
subscriber, port 5433
                  c1                  |     c2      |              c3
--------------------------------------+-------------+-------------------------------
 5e304346-8550-49b9-827c-b423765a2afb | publication | 2025-09-18 22:33:34.469708+00
 de807a9e-bcaf-41fb-b35d-a9382c61f7b9 | publication | 2025-09-18 22:33:34.46979+00
 6c821bea-72e0-4b8b-bf41-0e242861f128 | publication | 2025-09-18 22:33:34.469795+00
(3 rows)
```


©2025 Percona

Replication breaks



©2025 Percona

Mitigation;
adding column to subscriber restores replication

PG1

```
alter table a
    add column if not exists c4 text,
    alter column c4 set default 'PUBLICATION, before subscription update';

insert into a(c4) values (default)
                        ,(default)
                        ,(default)
                        ,(default);

\qecho publication, port 5432
table a;
```

PG2

```
alter table a
    add column if not exists c4 text,
    alter column c4 set default 'SUBSCRIBER, after subscription update';

insert into a(c4) values (default)
                        ,(default)
                        ,(default)
                        ,(default);
\qecho sleeping
select pg_sleep(10);
\qecho subscriber, port 5433, after adding column
table a;
```


©2025 Percona

Validation

```
publication, port 5432
                  c1                  |     c2      |             c3             |                    c4
--------------------------------------+-------------+----------------------------+-------------------------------------------
 5e304346-8550-49b9-827c-b423765a2afb | publication | 2025-09-18 22:33:34.469708+00 |
 de807a9e-bcaf-41fb-b35d-a9382c61f7b9 | publication | 2025-09-18 22:33:34.46979+00  |
 6c821bea-72e0-4b8b-bf41-0e242861f128 | publication | 2025-09-18 22:33:34.469795+00 |
 1e840c88-e016-4b8a-9397-d5c8e0c53dbf | publication | 2025-09-18 22:33:41.761953+00 | PUBLICATION, before subscription update
 31b2b14f-9560-4ad3-b11f-936847254aba | publication | 2025-09-18 22:33:41.76206+00  | PUBLICATION, before subscription update
 dfb98dab-2dd7-48c0-8338-d69643a9ac36 | publication | 2025-09-18 22:33:41.762208+00 | PUBLICATION, before subscription update
 93751a74-30de-401c-b127-b865a36ab01c | publication | 2025-09-18 22:33:41.762211+00 | PUBLICATION, before subscription update
(7 rows)
```

```
subscriber, port 5433, before adding column
                  c1                  |     c2      |             c3
--------------------------------------+-------------+----------------------------
 5e304346-8550-49b9-827c-b423765a2afb | publication | 2025-09-18 22:33:34.469708+00
 de807a9e-bcaf-41fb-b35d-a9382c61f7b9 | publication | 2025-09-18 22:33:34.46979+00
 6c821bea-72e0-4b8b-bf41-0e242861f128 | publication | 2025-09-18 22:33:34.469795+00
(3 rows)
```

```
subscriber, port 5433, after adding column
                  c1                  |     c2      |             c3             |                    c4
--------------------------------------+-------------+----------------------------+-------------------------------------------
 5e304346-8550-49b9-827c-b423765a2afb | publication | 2025-09-18 22:33:34.469708+00 |
 de807a9e-bcaf-41fb-b35d-a9382c61f7b9 | publication | 2025-09-18 22:33:34.46979+00  |
 6c821bea-72e0-4b8b-bf41-0e242861f128 | publication | 2025-09-18 22:33:34.469795+00 |
 1e840c88-e016-4b8a-9397-d5c8e0c53dbf | publication | 2025-09-18 22:33:41.761953+00 | PUBLICATION, before subscription update
 31b2b14f-9560-4ad3-b11f-936847254aba | publication | 2025-09-18 22:33:41.76206+00  | PUBLICATION, before subscription update
 dfb98dab-2dd7-48c0-8338-d69643a9ac36 | publication | 2025-09-18 22:33:41.762208+00 | PUBLICATION, before subscription update
 93751a74-30de-401c-b127-b865a36ab01c | publication | 2025-09-18 22:33:41.762211+00 | PUBLICATION, before subscription update
 e344cca6-9515-4055-8b42-b7ea85b19455 | subscription | 2025-09-18 22:33:46.791814+00 | SUBSCRIBER, after subscription update
 091ab42a-8aeb-4098-b79e-6256ecc35668 | subscription | 2025-09-18 22:33:46.791851+00 | SUBSCRIBER, after subscription update
 29fac31a-7208-444f-90b4-8e3a6e8cd20c | subscription | 2025-09-18 22:33:46.791856+00 | SUBSCRIBER, after subscription update
 eb7d818b-7d80-4464-8830-5d754eefb965 | subscription | 2025-09-18 22:33:46.79186+00  | SUBSCRIBER, after subscription update
```


©2025 Percona

TIP ...

Add column to subscribed table first!

©2025 Percona

Example: Skipping Transactions, Working with LSN’s

©2025 Percona

Example: Skipping Transactions, Working with LSN’s

Logical Sequence Number (LSN)

- Points to specific locations within the WAL stream.
- Plays a fundamental role in replication, recovery, and data consistency.
- Tracks all DML and DDL operations as database changes over time.
- Subscribers
  - Use LSNs to track how far they have processed the WAL stream.
  - Guarantees all necessary changes to stay synchronized with the Publication.

©2025 Percona

Working with LSN's

PG1

```
create table a(c1 uuid PRIMARY KEY default gen_random_uuid(),
               c2 text not null,
               c3 timestamptz default clock_timestamp(),
               c4 text default null);

insert into a(c1,c2,c3,c4)
    values (default,'PG1: record1',default,'before conflict')
    returning c1,c2,c3;

create publication pub_ex07 for table a;
select * from pg_publication_tables;
```

PG2

```
create table a(c1 uuid PRIMARY KEY default gen_random_uuid(),
               c2 text UNIQUE not null,
               c3 timestamptz default clock_timestamp(),
               c4 text default null);

create subscription sub_ex07
    connection 'host=localhost port=5432 user=postgres password=postgres dbname=ex07'
    publication pub_ex07;

\qecho sleep for a few seconds and wait for COPY to complete ...
select pg_sleep(2);

insert into a(c1,c2,c3) values (default,'PG2: record1',default)
                              ,(default,'PG2: record2',default)
                              ,(default,'PG2: record3',default);
```


©2025 Percona

PG1

```
========= PG1: TABLE A =========
                  c1                  |      c2      |              c3
--------------------------------------+--------------+-------------------------------
 f3909798-e01a-4ca3-9e80-cdda7dce91c6 | PG1: record1 | 2025-09-26 14:12:49.203805+00
(1 row)

INSERT 0 1
CREATE PUBLICATION
 pubname   | schemaname | tablename |   attnames    | rowfilter
-----------+------------+-----------+---------------+-----------
 pub_ex07 | public     | a         | {c1,c2,c3,c4} |
(1 row)
```

PG2

```
################## STAGE 2, PG2: SUBSCRIBE AND POPULATE TABLE a ##################
CREATE TABLE
NOTICE:  created replication slot "sub_ex07" on publisher
CREATE SUBSCRIPTION
sleep for a few seconds and wait for COPY to complete ...
 pg_sleep
----------

(1 row)

INSERT 0 3
                  c1                  |      c2      |              c3               |       c4
--------------------------------------+--------------+-------------------------------+----------------
 f3909798-e01a-4ca3-9e80-cdda7dce91c6 | PG1: record1 | 2025-09-26 14:12:49.203805+00 | before conflict
 a4dc8b00-859a-4ec0-8219-910387ea27ee | PG2: record1 | 2025-09-26 14:16:49.585359+00 |
 92ddf416-54ae-4e94-9328-eb74da62c78f | PG2: record2 | 2025-09-26 14:16:49.585433+00 |
 cff8f5b9-e605-40bc-8308-5d96b68abcf5 | PG2: record3 | 2025-09-26 14:16:49.585436+00 |
(4 rows)
```


©2025 Percona

Replication Stalls

PG1: Add conflicting records

```
insert into a(c1,c2,c3,c4) values (default,'PG1: record2',default,'before conflict');
--
insert into a(c1,c2,c3,c4) values (default,'PG1: record2',default, 'conflict 1/3');
insert into a(c1,c2,c3,c4) values (default,'PG1: record2',default, 'conflict 2/3');
insert into a(c1,c2,c3,c4) values (default,'PG1: record2',default, 'conflict 3/3');
--
insert into a(c1,c2,c3,c4) values (default,'PG1: record3',default, 'after conflict');
insert into a(c1,c2,c3,c4) values (default,'PG1: record4',default, 'after conflict');
```

PG2: logical replication is stalled due to UNIQUE constraint on column "c2"

```
                c1                |      c2      |              c3               |       c4
----------------------------------+--------------+-------------------------------+----------------
f3909798-e01a-4ca3-9e80-cdda7dce91c6 | PG1: record1 | 2025-09-26 14:12:49.203805+00 | before conflict
62799f9c-5f0e-4bed-80aa-dd8d8806a2d4 | PG1: record2 | 2025-09-26 14:23:43.438212+00 | before conflict
a4dc8b00-859a-4ec0-8219-910387ea27ee | PG2: record1 | 2025-09-26 14:16:49.585359+00 |
92ddf416-54ae-4e94-9328-eb74da62c78f | PG2: record2 | 2025-09-26 14:16:49.585433+00 |
cff8f5b9-e605-40bc-8308-5d96b68abcf5 | PG2: record3 | 2025-09-26 14:16:49.585436+00 |
```


©2025 Percona

PG2: Conflicting LSN flagged

```
2025-09-26 14:46:49.568 UTC [243] LOG:  background worker "logical replication apply worker" (PID 19944) exited with exit code 1
2025-09-26 14:46:54.558 UTC [19952] LOG:  logical replication apply worker for subscription "sub_ex07" has started
2025-09-26 14:46:54.571 UTC [19952] ERROR:  duplicate key value violates unique constraint "a_c2_key"
2025-09-26 14:46:54.571 UTC [19952] DETAIL:  Key (c2)=(PG1: record2) already exists.
2025-09-26 14:46:54.571 UTC [19952] CONTEXT:  processing remote data for replication origin "pg_16901" during message type "INSERT" for replica
tion target relation "public.a" in transaction 124471, finished at 0/9B16E3B0
2025-09-26 14:46:54.573 UTC [243] LOG:  background worker "logical replication apply worker" (PID 19952) exited with exit code 1
```

```
==== LSN: 0/9B16E3B0 ====
```


©2025 Percona

Conflicting LSN is Skipped

PG2: Subscription skips LSN

```
alter subscription sub_ex07 skip (lsn='0/9B16E3B0');
```


©2025 Percona

but there’s a problem ….

©2025 Percona

PERCONA

PG2: There’s another blocking record!

```
skip LSN=0/9B16E3B0
ALTER SUBSCRIPTION
sleeping ...
==== NEW LSN: 0/9B16E498 ====
```


©2025 Percona

Mitigation;

ID conflicting records

```
pg_logical_slot_peek_binary_changes ( slot_name name, upto_lsn pg_lsn, upto_nchanges integer, VARIADIC options text[] ) → setof
      record ( lsn pg_lsn, xid xid, data bytea )
```

PG1:

```
with aa as (select distinct on(lsn) lsn::pg_lsn, xid
            from pg_logical_slot_peek_binary_changes
            ( 'sub_ex07', NULL, NULL
             ,'proto_version','1'
             ,'publication_names','pub_ex07')
            )
select lsn, c1, c2, c3, c4
from a, aa
where xid=a.xmin
and c4 ~ '^conflict'
order by lsn,c3;
```

Attention: the slot must NOT be active for this query to work, which is default behaviour with conflicts!

lsn	c1	c2	c3	c4
0/9B16E3E0	54d04e91-63c7-4da3-b880-0f6f6b36509b	PG1: record2	2025-09-26 14:23:43.444962+00	conflict 2/3
0/9B16E4C8	54d04e91-63c7-4da3-b880-0f6f6b36509b	PG1: record2	2025-09-26 14:23:43.444962+00	conflict 2/3
0/9B16E5B0	c7aef88c-2c68-4b8f-9cdc-ca41e0593dbc	PG1: record2	2025-09-26 14:23:43.450882+00	conflict 3/3

©2025 Percona

Mitigation;

LSNs are Skipped

pg_replication_slot_advance (*slot_name* name, *upto_lsn* pg_lsn) → record (*slot_name* name, *end_lsn* pg_lsn)

Advances the current confirmed position of a replication slot named *slot_name*. The slot will not be moved backwards, and it will not be moved beyond the current insert location. Returns the name of the slot and the actual position that it was advanced to. The updated slot position information is written out at the next checkpoint if any advancing is done. So in the event of a crash, the slot may return to an earlier position. If the specified slot is a logical failover slot then the function will not return until all physical slots specified in `synchronized_standby_slots` have confirmed WAL receipt.

PG1: Skip remaining conflicting records

```
with aa as (select distinct on(lsn) lsn::pg_lsn, xid
            from pg_logical_slot_peek_binary_changes
            ( 'sub_ex07', NULL, NULL
             ,'proto_version','1'
             ,'publication_names','pub_ex07')
            ),
     bb as (select lsn, c1, c2, c3, c4
            from a, aa
            where xid=a.xmin
            and c4 ~ '^conflict'
            order by lsn,c3)
select pg_replication_slot_advance('sub_ex07',lsn)
from bb
order by lsn desc limit 1;
```

```
    lsn     |                  c1                  |
------------+--------------------------------------+
 0/9B16E3E0 | 54d04e91-63c7-4da3-b880-0f6f6b36509b |
 0/9B16E4C8 | 54d04e91-63c7-4da3-b880-0f6f6b36509b |
 0/9B16E5B0 | c7aef88c-2c68-4b8f-9cdc-ca41e0593dbc |
```

```
pg_replication_slot_advance
----------------------------
(sub_ex07,0/9B16E5B0)
```


©2025 Percona

Mitigation;

validation

```
====== PUBLICATION ======
                  c1                  |      c2      |              c3               |       c4
--------------------------------------+--------------+-------------------------------+----------------
 f3909798-e01a-4ca3-9e80-cdda7dce91c6 | PG1: record1 | 2025-09-26 14:12:49.203805+00 | before conflict
 62799f9c-5f0e-4bed-80aa-dd8d8806a2d4 | PG1: record2 | 2025-09-26 14:23:43.438212+00 | before conflict
 9fc263df-dd3f-423b-825c-88e4a6f9c0cc | PG1: record2 | 2025-09-26 14:23:43.441515+00 | conflict 1/3
 54d04e91-63c7-4da3-b880-0f6f6b36509b | PG1: record2 | 2025-09-26 14:23:43.444962+00 | conflict 2/3
 c7aef88c-2c68-4b8f-9cdc-ca41e0593dbc | PG1: record2 | 2025-09-26 14:23:43.450882+00 | conflict 3/3
 12e597b4-3681-4888-9800-ac764b5b9724 | PG1: record3 | 2025-09-26 14:23:43.454349+00 | after conflict
 7dff4aeb-f9b0-41c0-a967-14e8d3a5145a | PG1: record4 | 2025-09-26 14:23:43.457738+00 | after conflict
(7 rows)

====== SUBSCRIPTION ======
                  c1                  |      c2      |              c3               |       c4
--------------------------------------+--------------+-------------------------------+----------------
 f3909798-e01a-4ca3-9e80-cdda7dce91c6 | PG1: record1 | 2025-09-26 14:12:49.203805+00 | before conflict
 62799f9c-5f0e-4bed-80aa-dd8d8806a2d4 | PG1: record2 | 2025-09-26 14:23:43.438212+00 | before conflict
 12e597b4-3681-4888-9800-ac764b5b9724 | PG1: record3 | 2025-09-26 14:23:43.454349+00 | after conflict
 7dff4aeb-f9b0-41c0-a967-14e8d3a5145a | PG1: record4 | 2025-09-26 14:23:43.457738+00 | after conflict
 a4dc8b00-859a-4ec0-8219-910387ea27ee | PG2: record1 | 2025-09-26 14:16:49.585359+00 |
 92ddf416-54ae-4e94-9328-eb74da62c78f | PG2: record2 | 2025-09-26 14:16:49.585433+00 |
 cff8f5b9-e605-40bc-8308-5d96b68abcf5 | PG2: record3 | 2025-09-26 14:16:49.585436+00 |
(7 rows)
```


©2025 Percona

Example: Database Copy Failure

©2025 Percona

Example: Database Copy Failure

```
# pg1
createdb -p 5432 ex02 --template=ex01

# pg2
createdb -p 5433 ex02 --template=ex01
_eof_
```

Because of the enabled subscription on pg2, the replication slot on pg1 is still active!

PG1 `select * from pg_get_replication_slots()`

```
postgres@examples:/root$ tail -f /var/log/postgresql/postgresql-17-pg2.log | grep -A 3 -E 'ERROR'
2025-09-16 20:04:41.979 UTC [561] postgres@postgres ERROR:  source database "ex01" is being accessed by other users
2025-09-16 20:04:41.979 UTC [561] postgres@postgres DETAIL:  There is 1 other session using the database.
2025-09-16 20:04:41.979 UTC [561] postgres@postgres STATEMENT:  CREATE DATABASE ex02 TEMPLATE ex01;
```


©2025 Percona

Resolving a database copy failure

Steps:

1) Disable subscription
2) Copy databases
3) Rename publication (?)
4) Update subscription conninfo
5) Enable subscription (Refresh: `copy=on vs copy=off`)

©2025 Percona

Mitigation;
subscriptions are disabled

PG2

```
psql -p 5433 ex01 <<_eof1_
    \qecho ===== disable subscription =====
    alter subscription sub_ex01 disable;
_eof1_
```

PG1

```
createdb -p 5432 ex02 --template=ex01
```

PG2

```
createdb -p 5433 ex02 --template=ex01

psql -p 5433 ex01 <<_eof1_
    \qecho ===== pg2.ex01 re-enable subscription =====
    alter subscription sub_ex01 enable;
_eof1_
```


©2025 Percona

Mitigation;
publication is renamed & new slot created

PG1

```
psql -p 5432 ex02 <<_eof1_
   \qecho ===== rename publication from pub_ex01 to pub_ex02 =====
    alter publication pub_ex01 rename to pub_ex02;

    \qecho ===== create a new logical slot for dbname=ex02 =====
    select * from pg_create_logical_replication_slot('sub_ex02','pgoutput');

   \qecho ===== validate pg1.ex02.t1 =====
    insert into t1 values (default)
                          ,(default)
                          ,(default);
    table t1;
_eof1_
```

```
===== rename publication from pub_ex01 to pub_ex02 =====
ALTER PUBLICATION
===== create a new logical slot for dbname=ex02 =====
 slot_name |    lsn
-----------+------------
 sub_ex02  | 0/915D95B0
(1 row)

===== validate pg1.ex02.t1 =====
INSERT 0 3
 c1 |     c2      |              c3
----+-------------+-------------------------------
  1 | publication | 2025-09-15 20:09:58.804698+00
  2 | publication | 2025-09-15 20:09:58.804829+00
  3 | publication | 2025-09-15 20:09:58.804835+00
  6 | publication | 2025-09-15 20:23:24.377247+00
  8 | publication | 2025-09-15 20:23:24.377275+00
 10 | publication | 2025-09-15 20:23:24.377278+00
 12 | publication | 2025-09-15 20:29:59.091371+00
 14 | publication | 2025-09-15 20:29:59.091456+00
 16 | publication | 2025-09-15 20:29:59.091458+00
(9 rows)
```


©2025 Percona

Mitigation;
subscription is edited/updated

PG2

```
psql -p 5433 ex02 <<_eof1_

    \qecho ===== validate pg2.ex02.t1 BEFORE =====
    table t1;

    create subscription sub_ex02
    connection 'host=localhost port=5432 user=postgres password=postgres dbname=ex02'
    publication pub_ex02
    with (enabled=true, copy_data=false, create_slot=false, slot_name=sub_ex02);

    select pg_sleep(2);

   \qecho ===== validate pg2.ex02.t1 AFTER =====
    table t1;
_eof1_
_eof_
```

```
===== validate pg2.ex02.t1 BEFORE =====
 c1 |      c2       |              c3
----+---------------+-------------------------------
  1 | publication   | 2025-09-15 20:09:58.804698+00
  2 | publication   | 2025-09-15 20:09:58.804829+00
  3 | publication   | 2025-09-15 20:09:58.804835+00
  6 | publication   | 2025-09-15 20:23:24.377247+00
  8 | publication   | 2025-09-15 20:23:24.377275+00
 10 | publication   | 2025-09-15 20:23:24.377278+00
 11 | subscription  | 2025-09-15 20:23:24.449612+00
 13 | subscription  | 2025-09-15 20:23:24.449648+00
 15 | subscription  | 2025-09-15 20:23:24.44965+00
(9 rows)

CREATE SUBSCRIPTION
 pg_sleep
----------

(1 row)

===== validate pg2.ex02.t1 AFTER =====
 c1 |      c2       |              c3
----+---------------+-------------------------------
  1 | publication   | 2025-09-15 20:09:58.804698+00
  2 | publication   | 2025-09-15 20:09:58.804829+00
  3 | publication   | 2025-09-15 20:09:58.804835+00
  6 | publication   | 2025-09-15 20:23:24.377247+00
  8 | publication   | 2025-09-15 20:23:24.377275+00
 10 | publication   | 2025-09-15 20:23:24.377278+00
 11 | subscription  | 2025-09-15 20:23:24.449612+00
 13 | subscription  | 2025-09-15 20:23:24.449648+00
 15 | subscription  | 2025-09-15 20:23:24.44965+00
 12 | publication   | 2025-09-15 20:29:59.091371+00
 14 | publication   | 2025-09-15 20:29:59.091456+00
 16 | publication   | 2025-09-15 20:29:59.091458+00
(12 rows)
```


©2025 Percona

we can keep on doing this all day long ….

©2025 Percona

Caveat

- Permissions requires for remote access by subscriber
- Logical replication declarations are database specific
- Schemas must be named the same between databases
- Tuples must be unique for deletes, updates etc
- One unique index/constraint/pk per table
- DDLs not supported
- Sequences (doesn’t increment on subscribers)
- Triggers (doesn’t fire on subscribers)
- Unlogged/temporary tables not supported
- Logical replication from replicas only available on versions 16+
- No Replication Queue Flush (Failover is problematic)

©2025 Percona

Let’s move onto something else ...

©2025 Percona

Active-Active Architectures

©2025 Percona

Understanding Active-Active Architectures

Key Concerns

- DML conflicts when the subscribed data conflicts with data already present in the relation.
- The "echo effect" where data, initially propagated from the published table, returns to its original source host server as a DML operation.

Mitigation Techniques

- Maintaining consistent table definitions i.e. table columns and their constraints between PUBLISHED and SUBSCRIBED tables.
- Filtering undesired rows when creating the PUBLICATION using the WHERE clause
- Setting the ORIGIN parameter to "none" at SUBSCRIPTION creation.

©2025 Percona

Active-Active Caveat

All tuples (records) must be unique in a table i.e. possess a PRIMARY, or UNIQUE, key..

Where primary or unique keys do not exist in a table, this SQL command add the necessary information to the WAL.

ALTER TABLE … REPLICA IDENTITY

©2025 Percona

Active-Active: 2 Node example

Step 1: Create Environment

```
-- common to both primary nodes
create role active_active with login replication password 'active';
create database ex08;
```


©2025 Percona

Step 1: cont'd

```
-- pg1
\c ex08
drop table if exists t1;
create table t1(
     id uuid default gen_random_uuid()
    ,comments text default 'pg1'
    ,t_stamp timestamptz default clock_timestamp()
    ,PRIMARY KEY (t_stamp,id)
);

grant all on all tables in schema public to active_active
grant insert, update, delete on table t1 TO active_active;
```

```
-- pg2
\c db01
drop table if exists t1;
create table t1(
     id uuid default gen_random_uuid()
    ,comments text default 'pg2'
    ,t_stamp timestamptz default clock_timestamp()
    ,PRIMARY KEY (t_stamp,id)
);


grant all on all tables in schema public to active_active;
grant insert, update, delete on table t1 TO active_active;
```


©2025 Percona

Step 2: Start Replication

```
-- PUBLICATION
--
-- pg1
create publication pg1_ex08 for table t1;

-- pg2
create publication pg2_ex08 for table t1;
```

```
-- SUBSCRIPTION
--
-- pg1
create subscription pg1_sub_ex08
    connection 'host=pg2 port=5433 dbname=db01 user=active_active password=active'
    publication pg2_pub_ex08
    with (origin=none, copy_data=false);

-- pg2
create subscription pg2_sub_ex08
    connection 'host=pg1 port=5432 dbname=db01 user=active_active password=active'
    publication pg1_pub_ex08
    with (origin=none, copy_data=true);
```


©2025 Percona

Step 3: Validation, DML Operations

```
-- pg1
insert into t1 values
    (default,default,default),
    (default,default,default),
    (default,default,default);
```

```
-- pg2
insert into t1 values
    (default,default,default),
    (default,default,default),
    (default,default,default);
```

```
################## HOST PG1, VALIDATION ##################
                  id                  | comments |            t_stamp
--------------------------------------+----------+-------------------------------
 2f118fcf-4e54-43b8-8693-cc04982a045d | pg1      | 2025-09-26 22:25:35.68061+00
 75b98f52-93e0-47b5-a5d9-c88f95fbb6fb | pg1      | 2025-09-26 22:25:35.680738+00
 f6b57d0d-dbfe-4ecd-9c23-6cfa408c5352 | pg1      | 2025-09-26 22:25:35.680741+00
 786b1b5f-00d9-4388-abe7-1dce8a1ad56a | pg2      | 2025-09-26 22:25:35.728702+00
 28185a9a-e4d0-4034-b275-40d51b2bea44 | pg2      | 2025-09-26 22:25:35.728773+00
 8c300385-947a-47f4-ae54-97a02c3bbfdf | pg2      | 2025-09-26 22:25:35.728775+00
(6 rows)


################## HOST PG2, VALIDATION ##################
                  id                  | comments |            t_stamp
--------------------------------------+----------+-------------------------------
 2f118fcf-4e54-43b8-8693-cc04982a045d | pg1      | 2025-09-26 22:25:35.68061+00
 75b98f52-93e0-47b5-a5d9-c88f95fbb6fb | pg1      | 2025-09-26 22:25:35.680738+00
 f6b57d0d-dbfe-4ecd-9c23-6cfa408c5352 | pg1      | 2025-09-26 22:25:35.680741+00
 786b1b5f-00d9-4388-abe7-1dce8a1ad56a | pg2      | 2025-09-26 22:25:35.728702+00
 28185a9a-e4d0-4034-b275-40d51b2bea44 | pg2      | 2025-09-26 22:25:35.728773+00
 8c300385-947a-47f4-ae54-97a02c3bbfdf | pg2      | 2025-09-26 22:25:35.728775+00
(6 rows)
```


©2025 Percona

Understanding CREATE SUBSCRIPTION is key!

```
CREATE SUBSCRIPTION subscription_name
    CONNECTION 'conninfo'
    PUBLICATION publication_name [, ...]
    [ WITH ( subscription_parameter [= value] [, ... ] ) ]
```

Subscription Parameters (ver 17):

- connect
- create_slot
- enabled
- slot_name
- binary
- **copy_data**
- streaming
- synchronous_commit
- two_phase
- disable_on_error
- password_required
- run_as_owner
- **origin**
- failover

copy_data: Specifies whether to copy pre-existing data from the publication(s) i.e. true/false

origin:

- Setting to **none**: subscription requests publisher to send changes without an origin
- Setting to **any**: publisher sends changes regardless of origin i.e. *echo effect*

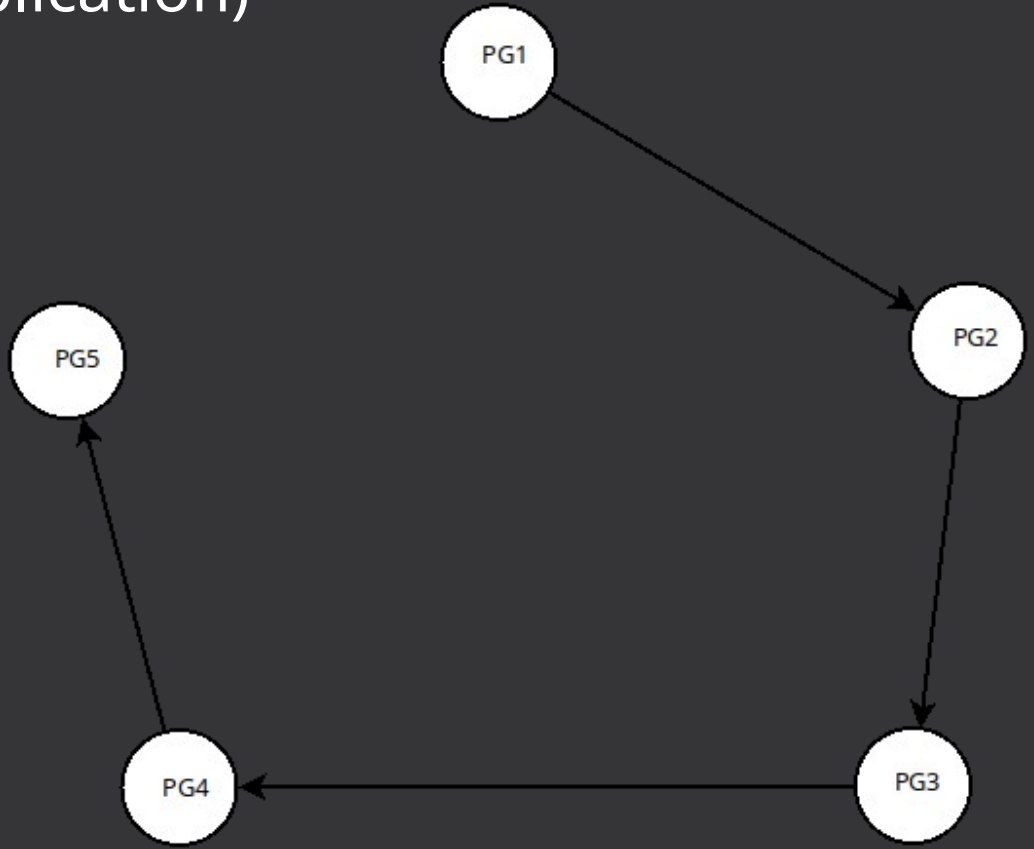
©2025 Percona

Architectural Types

- Daisy Chain
- Star
- Mesh

©2025 Percona

Daisy Chain
(1 way-replication)



©2025 Percona

```
for u in 1 2 3 4 5
do
    echo "========= pg$u ========="
    psql -q "host=pg$u dbname=db01 user=postgres password=postgres" <<_eof_
        drop publication if exists pg$u;
        drop table if exists t1;
        create table t1(
            id uuid default gen_random_uuid()
            ,comments text default 'pg$u-openai'
            ,t_stamp timestamptz default clock_timestamp()
            ,PRIMARY KEY (t_stamp,id)
        );
        grant all on all tables in schema public to active_active;
        create publication pg$u for table t1;
_eof_
done
```


©2025 Percona

```
for u in 2 3 4 5
do
    let x=$u-1
    echo "========= pg$x -> pg$u ========="
    psql -q "host=pg$u dbname=db01 user=postgres password=postgres" <<_eof_
        create subscription pg$x
            connection 'host=pg$x port=5432 dbname=db01 user=active_active password=active'
            publication pg$x
            with (origin=any, copy_data=false, slot_name=pg$u);
_eof_
done
```

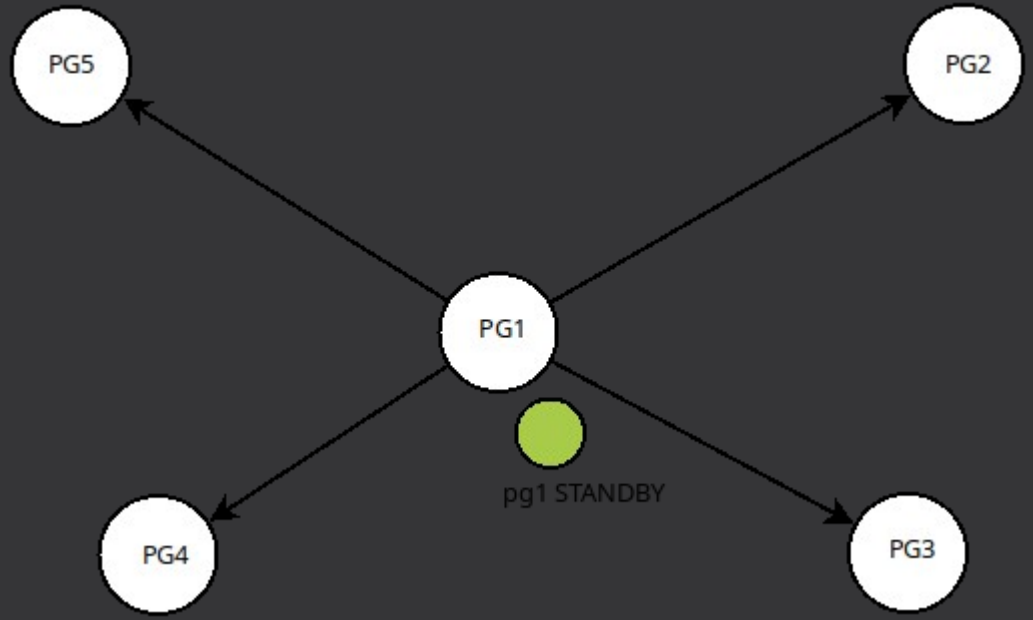

©2025 Percona

Daisy Chain

- Advantages
- Disadvantages

©2025 Percona

Star Topology
(1 way-replication)



©2025 Percona

```
psql "host=pg1 dbname=db01 user=postgres password=postgres" <<_eof_
    drop publication if exists pg1;
    drop table if exists t1;
    create table t1(
        id uuid default gen_random_uuid()
        ,comments text default 'pg1-openai'
        ,t_stamp timestamptz default clock_timestamp()
        ,PRIMARY KEY (t_stamp,id)
    );
    create publication pg1 for table t1;
    grant all on all tables in schema public to active_active;
_eof_
```

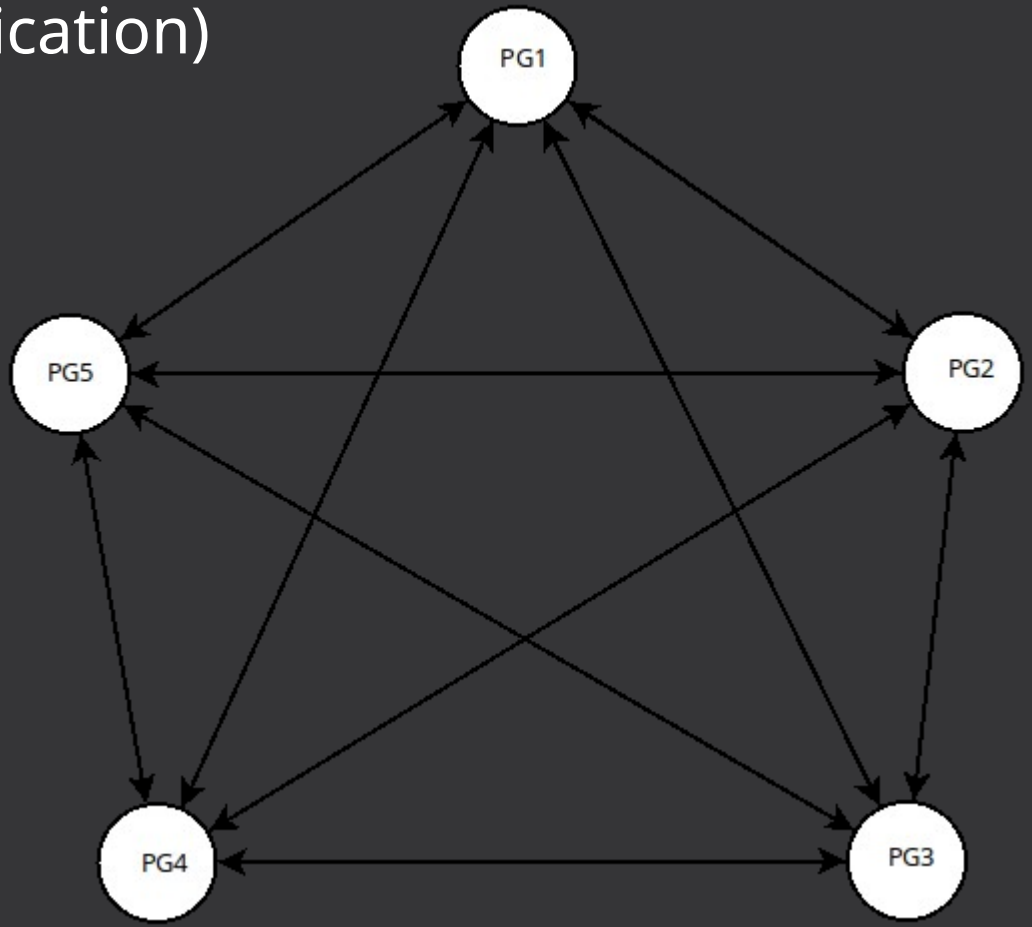
```
psql "host=pg1 dbname=db01 user=postgres password=postgres" <<_eof_
    drop publication if exists pg1;
    drop table if exists t1;
    create table t1(
        id uuid default gen_random_uuid()
        ,comments text default 'pg1-openai'
        ,t_stamp timestamptz default clock_timestamp()
        ,PRIMARY KEY (t_stamp,id)
    );
_eof_
```


©2025 Percona

```
for u in 2 3 4 5
do

    echo "========= pg1 -> pg$u ========="
psql -q "host=pg$u dbname=db01 user=postgres password=postgres" <<_eof_
create subscription pg1
        connection 'host=pg1 port=5432 dbname=db01 user=active_active password=active'
        publication pg1
        with (copy_data=false, slot_name=pg$u);
_eof_

done
```


©2025 Percona

Mesh
(2 way-replication)



©2025 Percona

```
for u in 1 2 3 4 5
do
echo "========= pg${u} ========="
psql "host=pg${u} dbname=db01 user=postgres password=postgres" <<_eof_
    drop publication if exists pg${u};
    drop table if exists t1;
    create table t1(
        id uuid default gen_random_uuid()
        ,comments text default 'pg${u}-openai'
        ,t_stamp timestamptz default clock_timestamp()
        ,PRIMARY KEY (t_stamp,id)
    );
    grant all on all tables in schema public to active_active;
    create publication pg${u} for table t1;
_eof_
done
```


©2025 Percona

```
for u in 1 2 3 4 5
do
    for v in 1 2 3 4 5
    do
    if [ $u -ne $v ]
    then
        echo "========= pg$u -> pg$v ========="
        psql -q "host=pg$u dbname=db01 user=postgres password=postgres" <<_eof_
          create subscription pg$v
            connection 'host=pg$v port=5432 dbname=db01 user=active_active password=active'
            publication pg$v
            with (origin=none, copy_data=false, slot_name=pg$u);
_eof_
    fi
    done
done
```


©2025 Percona

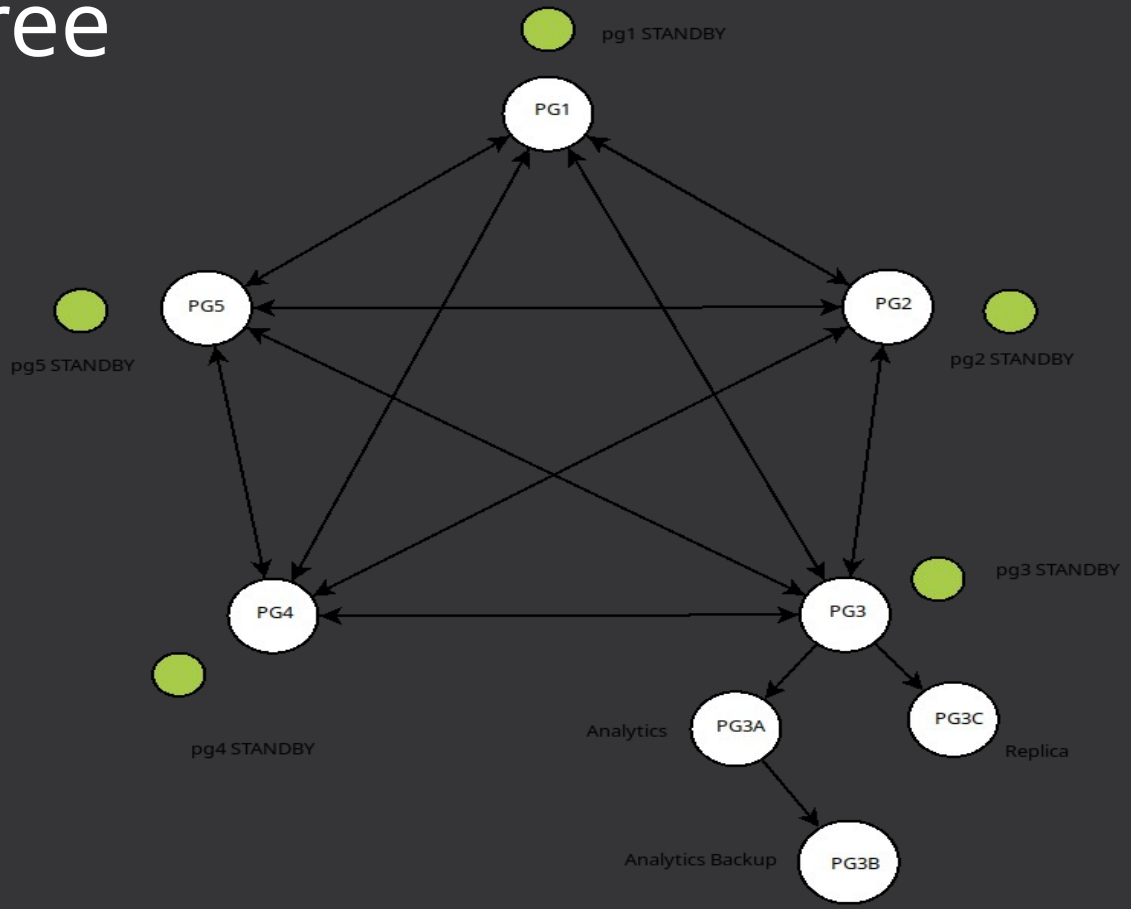
- Advantages: Not only can multiple DML operations be executed on each host but one can failover to any one of the PRIMARY read-write nodes without downtime or SUBSCRIPTION configuration updates.
- Disadvantages: Topology complexity and network overhead increases as the system scales.

©2025 Percona

Number Of Nodes	Total Number Slots req’d
2	2
3	6
4	12
5	20
6	30

©2025 Percona

XMAS Tree



©2025 Percona

THANK YOU!

©2025 Percona

QUESTIONS?

©2025 Percona