# Online upgrade of replication clusters without downtime

Hayato Kuroda

Vigneshwaran C

FUJITSU

# Speaker's profile

- Hayato Kuroda
  - Living in Japan
  - Working at Fujitsu

- Vigneshwaran C
  - Living in India
  - Working at Fujitsu Consulting India
  - Recognized contributor

# Agenda

- Introduction
- Upgrading streaming replication clusters
- Converting streaming clusters to logical ones
- Upgrading logical replication clusters

FUJITSU

FUJITSU

# Part 1 – Introduction

# Introduction – what is upgrade?

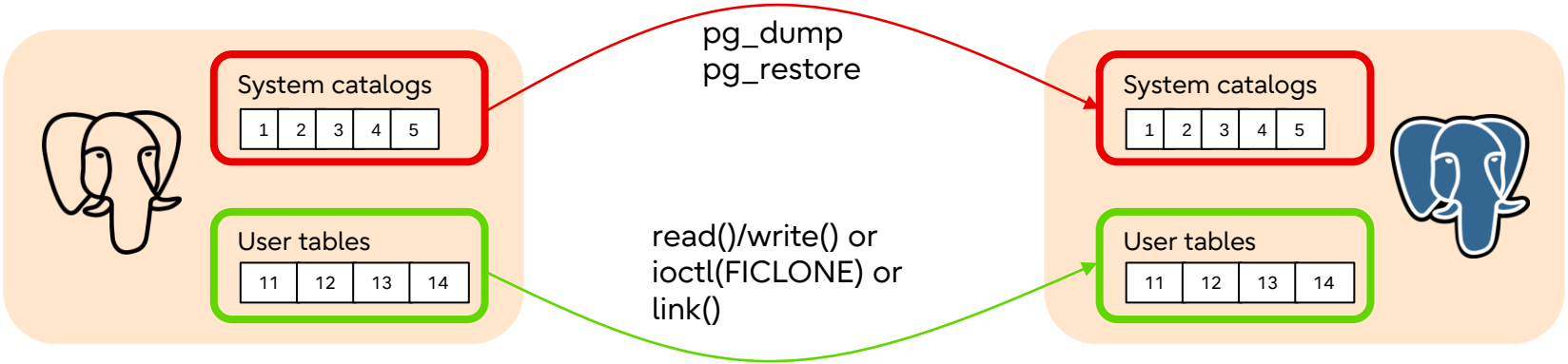
FUJITSU

- *to improve the quality or usefulness of something, ... – Cambridge Dictionary*
- In PostgreSQL, it means to use newer executables in your system
- Every major version adds a lot of features, tools, etc.
- **Major releases of PostgreSQL cannot understand old data directory**
  - System catalog may be changed
  - WAL format may be changed
  - etc.
- PostgreSQL community releases a new major version every year, but supports only for five years

# pg_upgrade – the way to upgrade your instance

FUJITSU

- The *built-in* upgrade tool
- 7x faster than normal dump and restore
- Avoids reading data by SQL commands
- Assumptions of this tool
  - System catalogs are changed for every releases
  - Table file format is preserved



# pg_upgrade - weaknesses

FUJITSU

- Requires the instance to be stopped
- Replication slot cannot be migrated, for PG16 and earlier
- **Breaks the streaming replication cluster**
  - Streaming replication requires major version of instances are the same

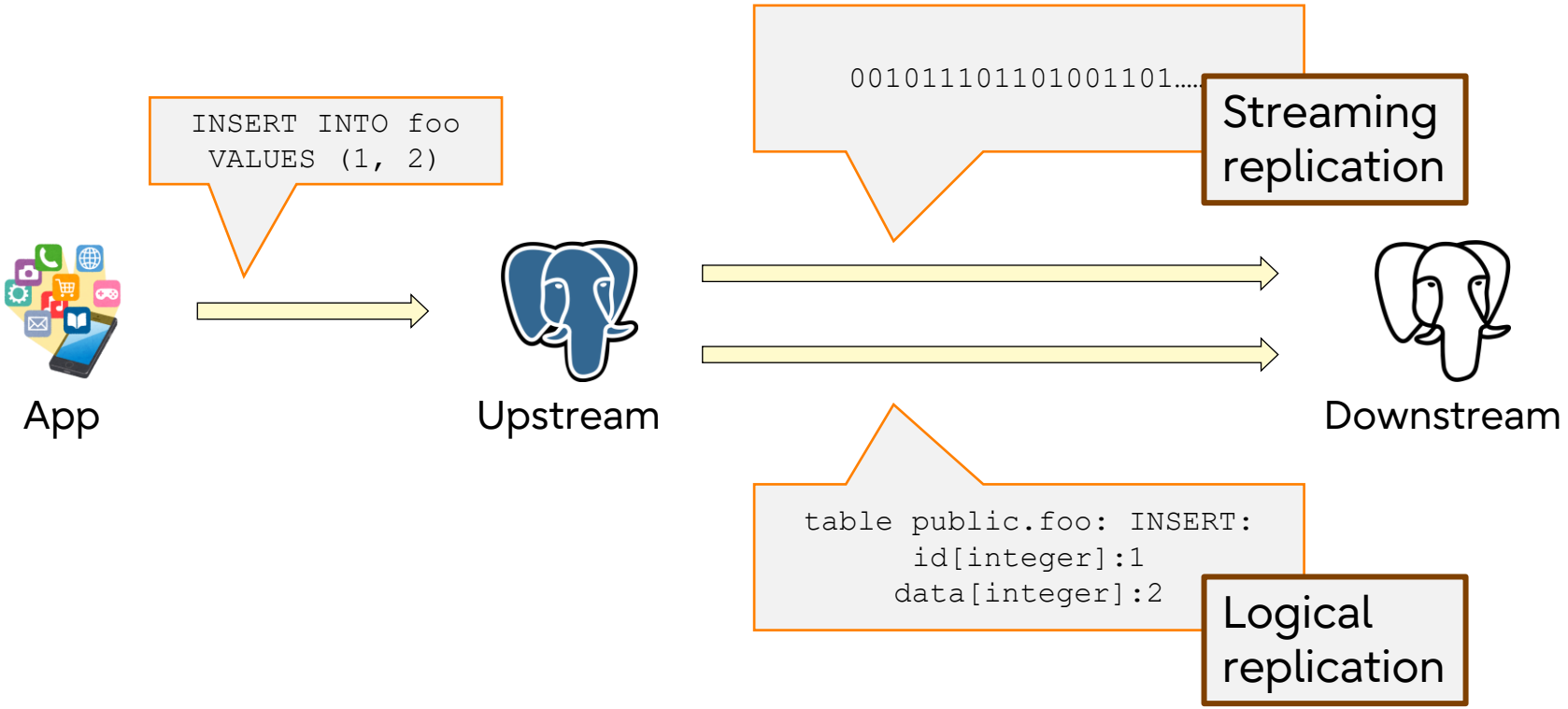
What should we do?

Use **logical replication**

# Understanding Replication in PostgreSQL



FUJITSU



# Understanding Replication in PostgreSQL



FUJITSU

| | Streaming replication | Logical replication |
|---|---|---|
| **Naming of instances** | Primary/Standby | Publisher/Subscriber |
| **What content do they send** | Exact WAL records | Replication messages,<br>extracted information from WAL |
| **Who initially synchronizes data** | pg_basebackup | Done automatically |
| **Replication target** | Whole of DB cluster | Per database |
| **What downstream can do** | Read-only queries | Both read and write queries |
| **Environments** | OS and major versions must be same | Can be different |

**Backup purpose**
Primary/standby becomes same state

**Backup and Other purposes**
Nodes can be different state

# Logical replication - Usage

FUJITSU

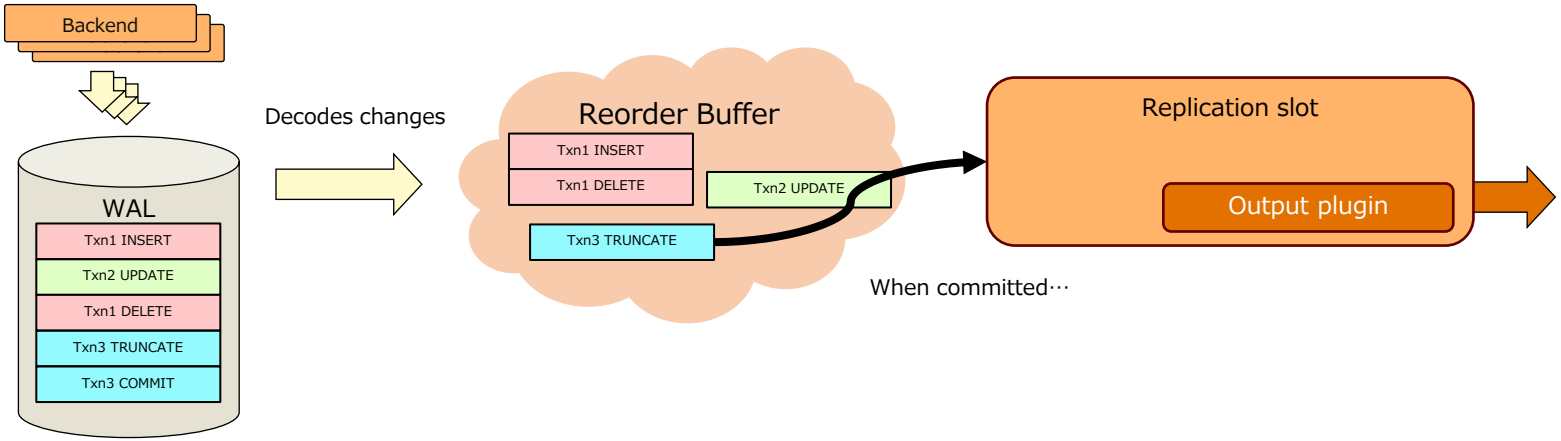
- The publication must be defined on an upstream node.

```
postgres=# CREATE PUBLICATION pub FOR ALL TABLES;
CREATE PUBLICATION
postgres=# SELECT * FROM pg_publication;
  oid   | pubname | pubowner | puballtables | pubinsert | pubupdate | pubdelete | pubtruncate | pubviaroot
-------+---------+----------+--------------+-----------+-----------+-----------+-------------+------------
 16396 | pub     |       10 | t            | t         | t         | t         | t           | f
(1 row)
```

- Then a downstream node subscription subscribes to the publication.

```
postgres=# CREATE SUBSCRIPTION sub CONNECTION 'user=postgres dbname=postgres port=5431' PUBLICATION pub;
NOTICE:  created replication slot "sub" on publisher
CREATE SUBSCRIPTION
postgres=# SELECT oid, subdbid, subname, subconninfo FROM pg_subscription;
  oid   | subdbid | subname |                subconninfo
-------+---------+---------+-----------------------------------------
 16402 |       5 | sub     | user=postgres dbname=postgres port=5431
(1 row)
```

# Logical replication - Replication slots




- Provides a way to **ensure the instance does not remove WAL files**
- Two types:
  - streaming replication slot
  - logical replication slot
- Logical slots contain an "output plugin", used by logical decoding

FUJITSU

# Part 2 - Upgrading streaming replication clusters

# Challenges of upgrading replication clusters

FUJITSU

- Major releases can change the layout of the system catalogs (addition of columns, changed column type, etc).
- Major releases can change WAL records (addition of new WAL record, modification of WAL record, etc)
- Data files cannot be used by the upgraded instance
- Streaming replication clusters does not work after one of the instances is upgraded

# Why use Logical replication for upgrades?

FUJITSU

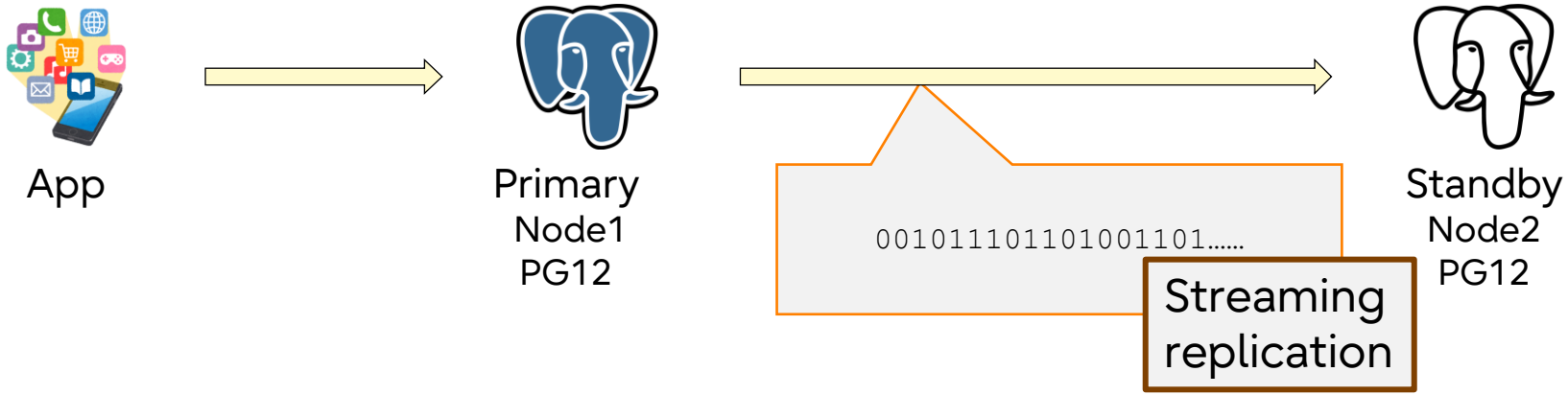
- Logical replication works across major versions, so even if one of the instances (Publisher or Subscriber) is upgraded, logical replication can continue
- WAL format changes do not affect logical replication
- Continues to identify and replicate changes even after the upgrade
- It helps in reducing the downtime

# Steps to upgrade streaming replication cluster (PG16)



FUJITSU

- We want to upgrade the cluster from PG12 to PG16
- Let's say primary is in node1 and standby is in node2
- Any concurrent activities are allowed, as much as possible
- Make sure `wal_level = logical` in primary



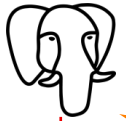
# Steps to upgrade streaming replication cluster (PG16)



FUJITSU

Primary
(Publisher)
PG12

Streaming replication
Logical replication



Another Primary
(Subscriber)
PG12

**1. CREATE replication slots for all databases**
`SELECT pg_create_logical_replication_slot…`

**2. Create publications for all tables**
`CREATE PUBLICATION FOR ALL TABLES`

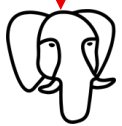
**4. Advance replication slots**
`SELECT pg_replication_slot_advance…`

Here we want to upgrade clusters from PG12 to PG16…

**3. Promote to primary**
```
$ pg_ctl -D node2 promote
```

**5. Create subscriptions for all databases**
`CREATE SUBSCRIPTION subXXX…`

**6. Stop and upgrade**
```
$ pg_ctl -D node2 stop
$ pg_upgrade -d node2 …
```



Primary
(Subscriber)
PG16

# Steps to upgrade streaming replication cluster (PG16)

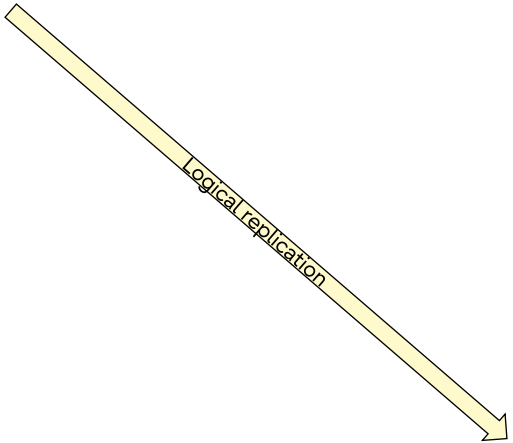


FUJITSU

Primary
(Publisher)
PG12

**9. Stop node1**

```
$ pg_ctl -D node1 stop
```




**7. Truncate all tables**

```
TRUNCATE XXX…
```

**8. Re-create subscriptions**

```
DROP SUBSCRIPTION subXX…
CREATE SUBSCRIPTION subXXX…
```

Primary
(Subscriber)
PG16

Standby
PG16

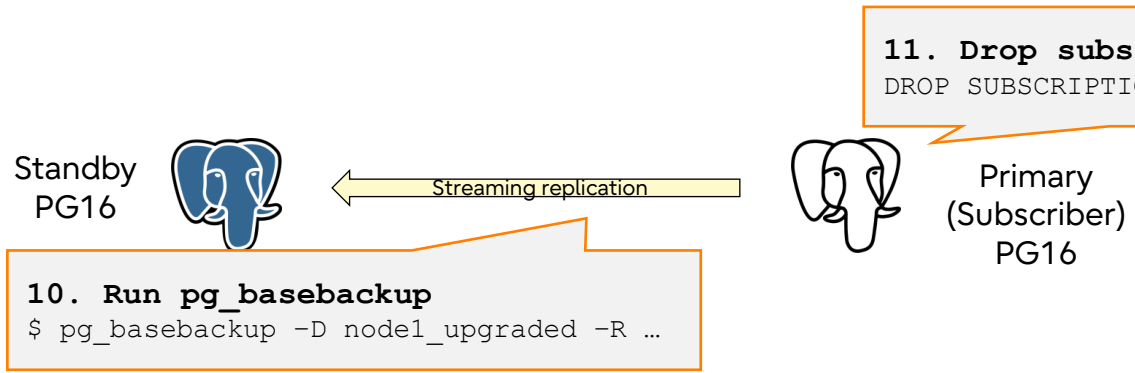
**10. Run pg_basebackup**

```
$ pg_basebackup -D node1_upgraded -R …
```

# Steps to upgrade streaming replication cluster (PG16)



FUJITSU



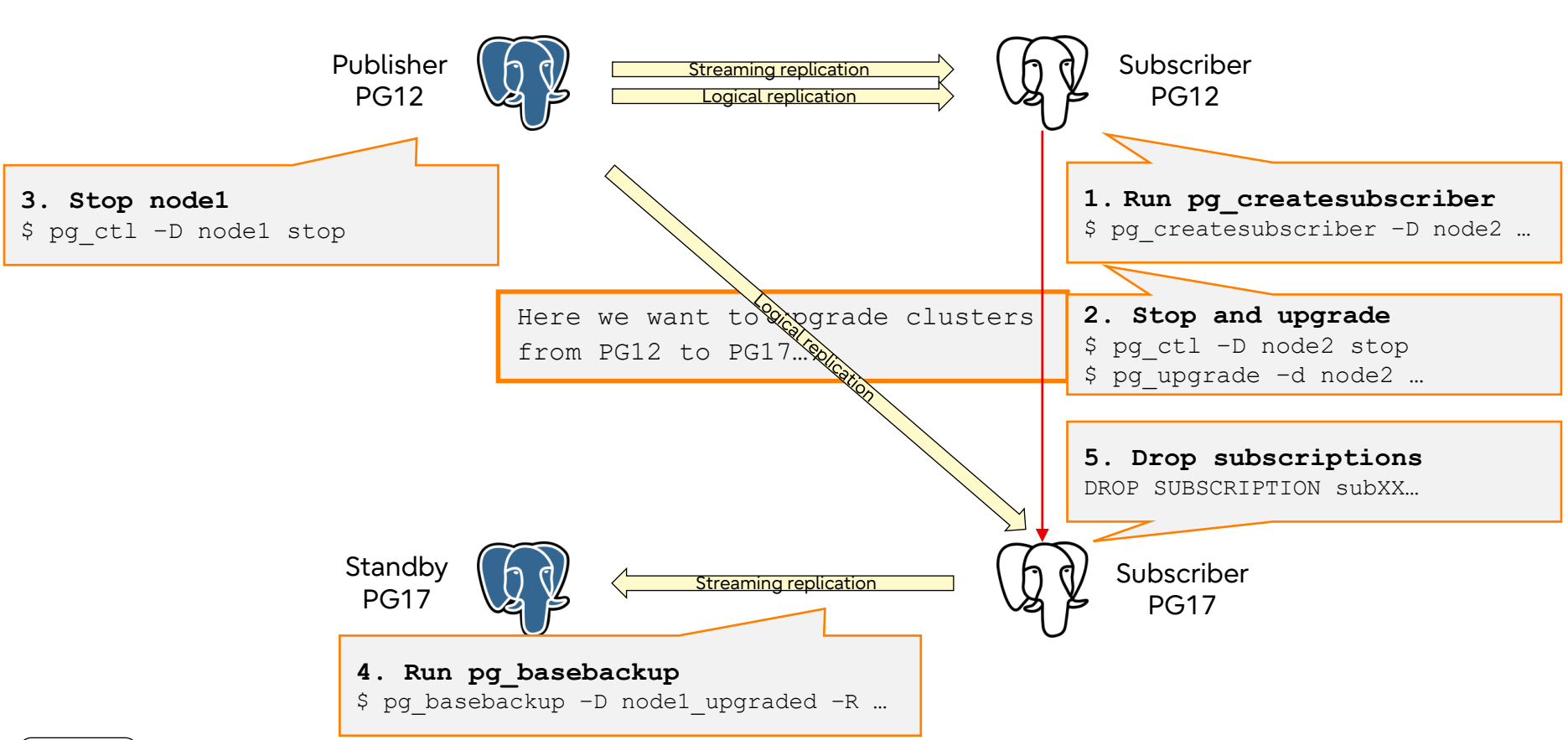
# The problems with PG16 upgrade

FUJITSU

- The logical replication slots must be re-created
- The replication slot LSN should be adjusted
- The subscription-related information will not be preserved
  - The subscriptions should be dropped
  - The table data should be truncated
  - The subscriptions should be re-created, depending on the data size
- This process is complex and can be time-consuming

# Steps to upgrade streaming replication cluster (PG17)

FUJITSU



FUJITSU

# Part 3 – Converting streaming clusters to logical ones

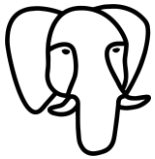
# Motivation

FUJITSU

```
CREATE PUBLICATION pub
FOR ALL TABLES;
```

App

Primary



Standby


I want to create a logical replica...

```
CREATE SUBSCRIPTION SUB
CONNECTION ……
WITH (COPY_DATA = ON);
```

New node

# Difficulties preparing the new Subscriber

FUJITSU

- **Takes a long time**
  - Initial synchronization runs COPY command, per table
  - Estimated execution time is proportional to the number of tables

- **Requires additional disk resources**
  - Replication slots will be created while copying data
  - Generated WAL files are preserved
  - They may fill up disk - *PANIC!*

# pg_createsubscriber – a new tool in PG17

FUJITSU

- **Converts physical standby into logical Subscriber**
  - Confirms the standby is caught up at the certain point,
  - Then defines subscriptions on the standby

- How? – introduces a **new server application**
  - Must be executed on the standby server

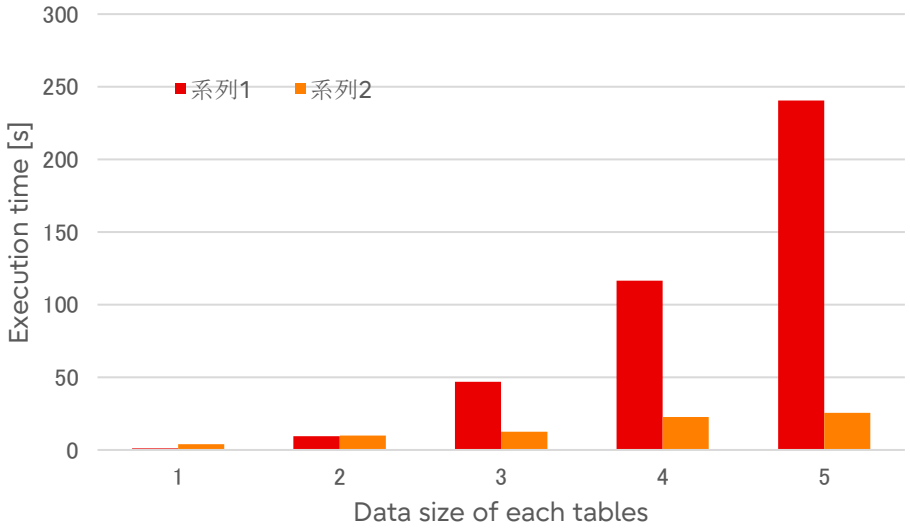
```
$ pg_createsubscriber [option...] { -d | --database } dbname
                                  { -D | --pgdata } datadir
                                  { -P | --publisher-server } connstr
```

- Pushed on HEAD!

# pg_createsubscriber - performance comparison

FUJITSU

- Compares the elapsed time while synchronizing 10 tables
  - Logical replication: elapsed time from CREATE SUBSCRIPTION to end of synchronization
  - pg_createsubscriber: command execution time




```
wal_level = logical
shared_buffers = 40GB
max_worker_processes = 32
max_parallel_maintenance_workers = 24
max_parallel_workers = 32
synchronous_commit = off
checkpoint_timeout = 1d
max_wal_size = 24GB
min_wal_size = 15GB
autovacuum = off
```

```
$ cat /proc/meminfo | grep MemTotal
MemTotal:       792237412 kB

$ grep processor /proc/cpuinfo | wc -l
120
```

# pg_createsubscriber - how it works



FUJITSU

**1. Verify this can be a Publisher**

```
wal_level == logical
max_walsender >=
   no. of target databases…
```

**2. Verify this can be a Subscriber**

```
max_worker_processes >
   no. of target databases…
```

**3. Create publications and replication slots for all tables**

```
CREATE PUBLICATION FOR ALL TABLES
SELECT
pg_create_logical_replication_slot…
```

**9. Update system_identifier**

```
$ pg_resetwal -D node1 …
$ pg_resetwal -D node2 …
```

Primary

Streaming replication

Standby

**4. Set recovery_target_lsn and recovery_target_action**

```
recovery_target_lsn = 'YYY'
recovery_target_action = promote
```

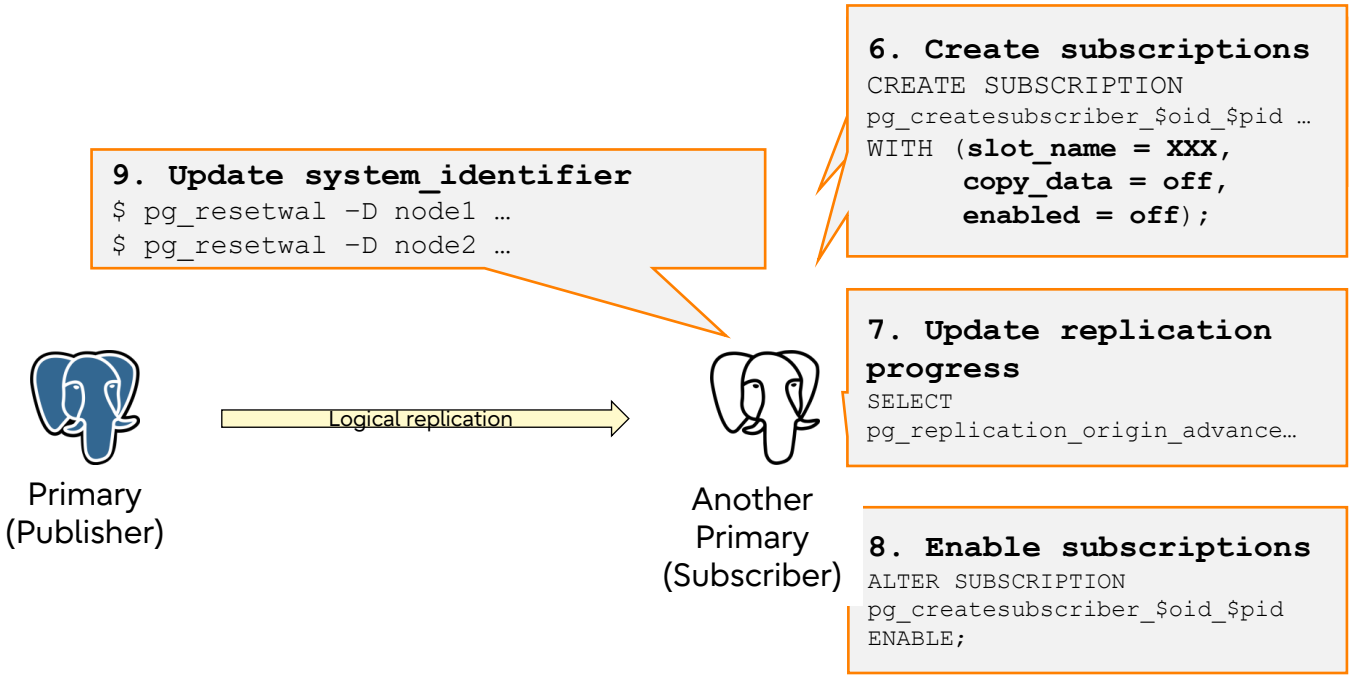
**5. Restart a standby**

```
$ pg_ctl -D standby restart
```

# pg_createsubscriber - how it works

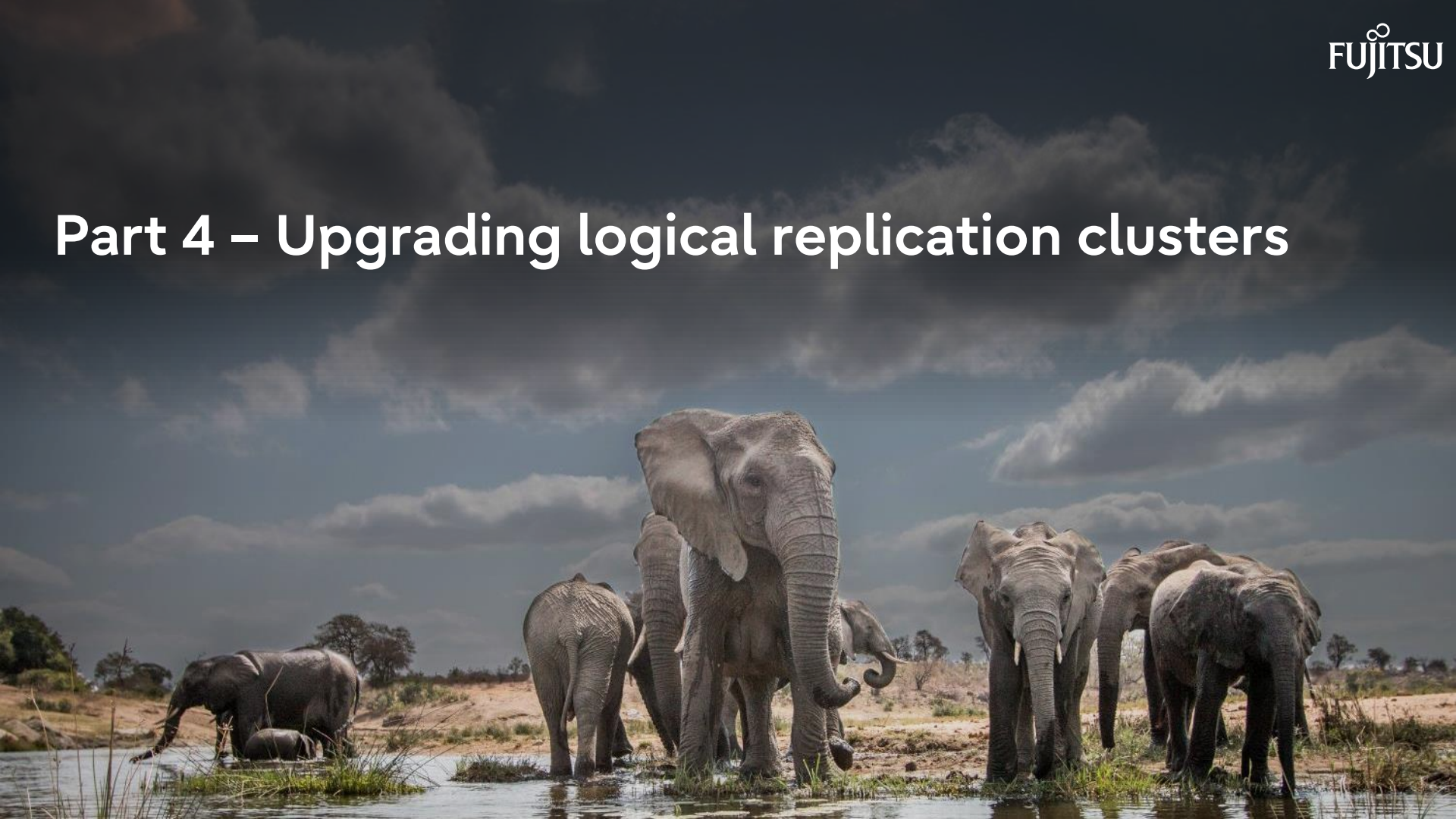


FUJITSU



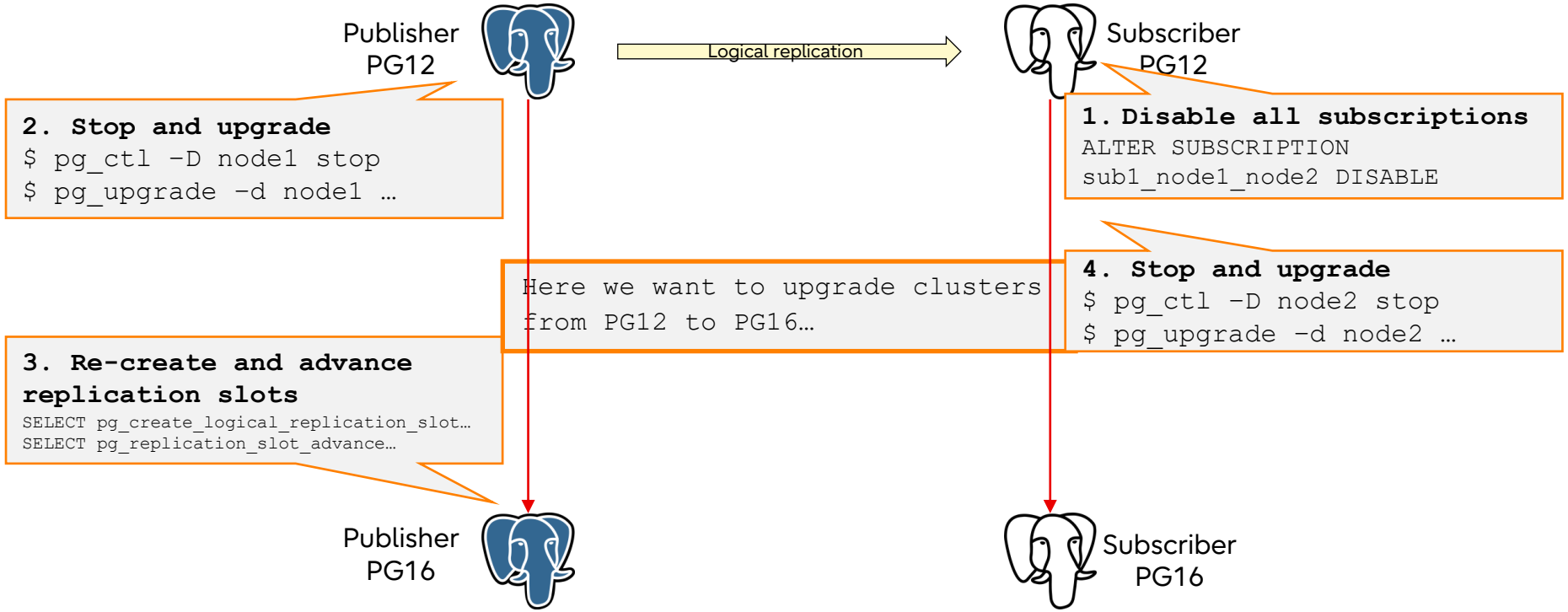
FUJITSU

# Part 4 – Upgrading logical replication clusters



# Upgrading logical replication clusters (PG16)

FUJITSU



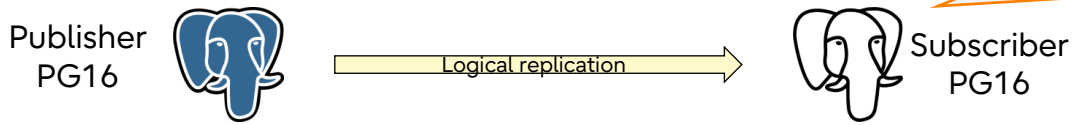
# Upgrading logical replication clusters (PG16)

FUJITSU

**5. Define tables that were created in node1 during upgrade**
```
CREATE TABLE … (if needed)
```

**6. Truncate all tables**
```
TRUNCATE XXX…
```

**7. Re-create subscriptions**
```
DROP SUBSCRIPTION sub1_node1_node2
CREATE SUBSCRIPTION sub1_node1_node2
```



# New features added in PostgreSQL 17



FUJITSU

- Logical replication slots are migrated:
  - Logical replication slot information will be copied from the old cluster
  - After the upgrade, just the subscription connection strings should be updated to point to the updated Publisher instance
  - Logical replication can continue seamlessly

# New features added in PostgreSQL 17

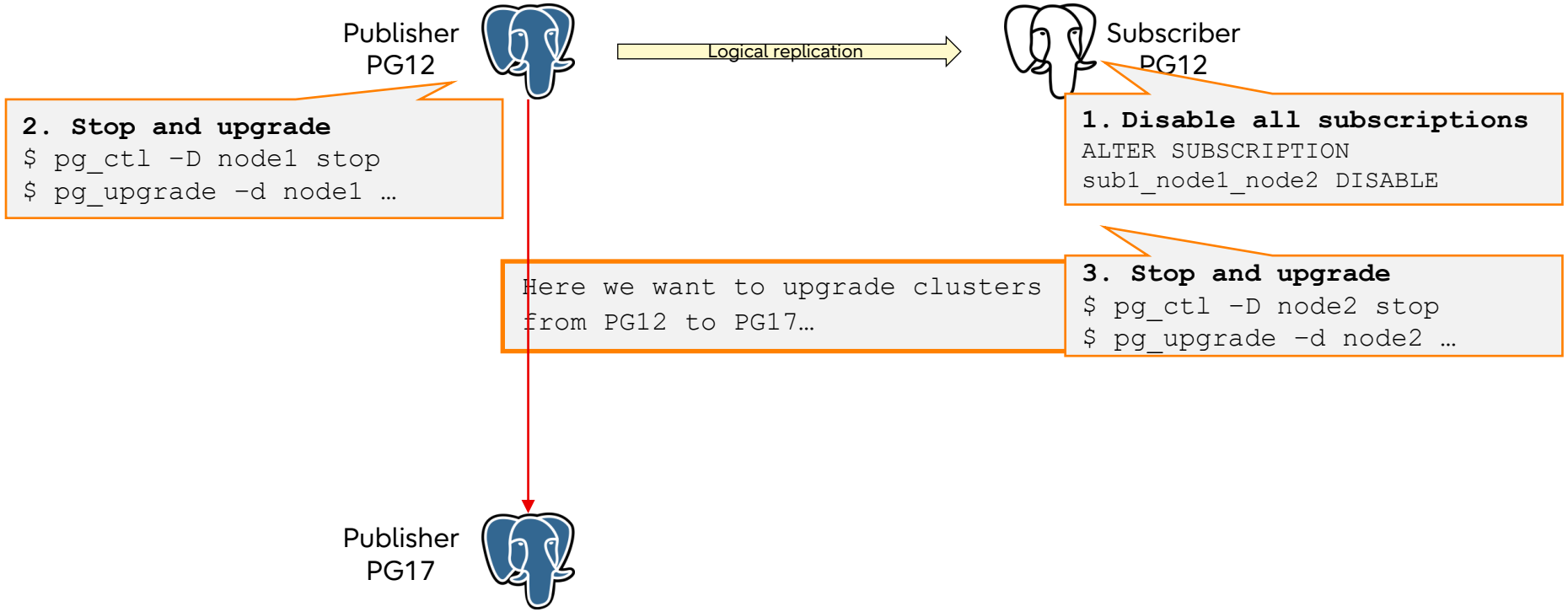


FUJITSU

- Subscription-related information is preserved:
  - Previously, only the subscription metadata information was preserved
  - Without the list of relations and their state, it's impossible to re-enable the subscriptions without missing some records
  - Now the `pg_subscription_rel` information will be preserved
  - Now replication origin will be preserved

# Upgrading logical replication clusters (PG17)

FUJITSU



# Upgrading logical replication clusters (PG17)

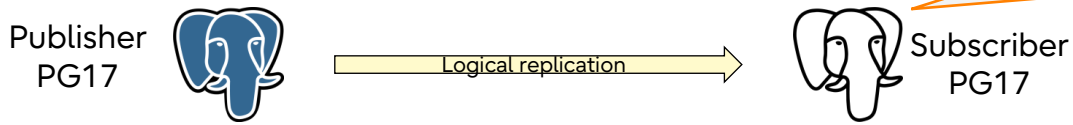
FUJITSU

**4. Define tables that were created in node1 during upgrade**

```
CREATE TABLE … (if needed)
```

**5. Enable and refresh subscriptions for all tables**

```
ALTER SUBSCRIPTIONS sub_node1_node2 ENABLE
ALTER SUBSCRIPTIONS sub_node1_node2 REFRESH PUBLICATION
```



# Summary

FUJITSU

- Upgrading replication clusters had many challenges
- Some features have been committed:
  - Preserving logical replication slots information during upgrade
  - Preserving subscription information during upgrade
  - **pg_createsubscriber,** which converts the streaming replication cluster to a logical replication cluster
- Together, these features remove most downtime while upgrading the streaming replication cluster
- Logical replication clusters can be upgraded now without the need to copy the table data again

# Appendix

FUJITSU

- https://www.postgresql.org/docs/16/pgupgrade.html
- https://www.postgresql.org/docs/current/protocol-replication.html
- https://www.postgresql.org/docs/current/logical-replication.html
- You can send any questions to:
  - kuroda.hayato@fujitsu.com

# Thank you

## Online upgrade of replication clusters without downtime

Hayato Kuroda

Vigneshwaran C