

9.2 to 15 and beyond



A case study of a tricky upgrade path

Nick Meyer @ Academia.edu

March 7 2025, SCaLE 22x


9.2 to 15 and beyond



A case study of a tricky upgrade path

Nick Meyer @ Academia.edu

March 7 2025, SCaLE 22x



Prelude: 2008
A database is born

A DB as an Archaeological Site

- “Written record” (slack? git? email?)
- Version: 8.3? Earlier?
- Eventually upgraded to 9.2
 - *At some point before 2018*

Mario Modesto Mata

Creative Commons Attribution-Share Alike 3.0 Unported

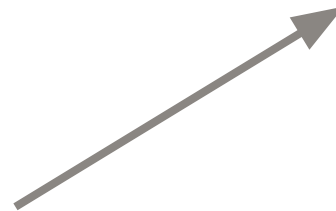




A A bit about me (Nick Meyer)

- [Academia.edu](https://academia.edu)
- <https://github.com/aristocrates>
- **Team lead of Platform Engineering**
- **Areas of focus**
 - Developer experience
 - Interface: application and infra
 - **Data layer**
 - **Postgres**





Slides: https://github.com/aristocrates/SCaLE22x_2025



A

What is the point?

A

What is the point of telling this story?

- **Empathy**
- **Emotion**
- **Technical details ignorable**
 - (but feel free to pay attention)
- **Are you:**
 - A contributor?
 - New community member?
 - Doing a similar upgrade?

A “Version-splaining”

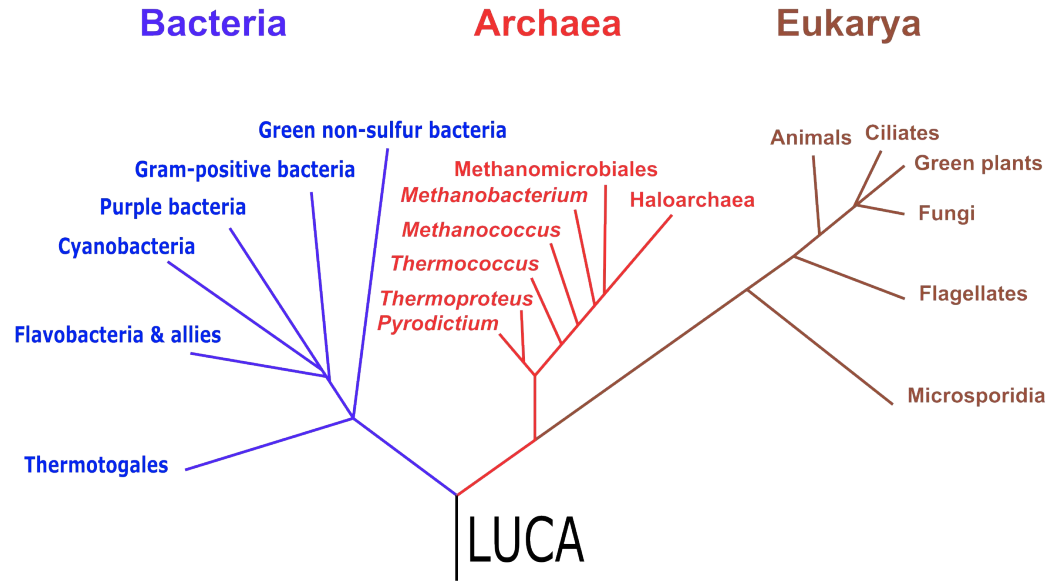
ver·sion·splain (verb)

/ 'vɜr ʒən ,splɛɪn /

To tell someone they should upgrade when they already said they are working on it.

A

“Vertical” sharding

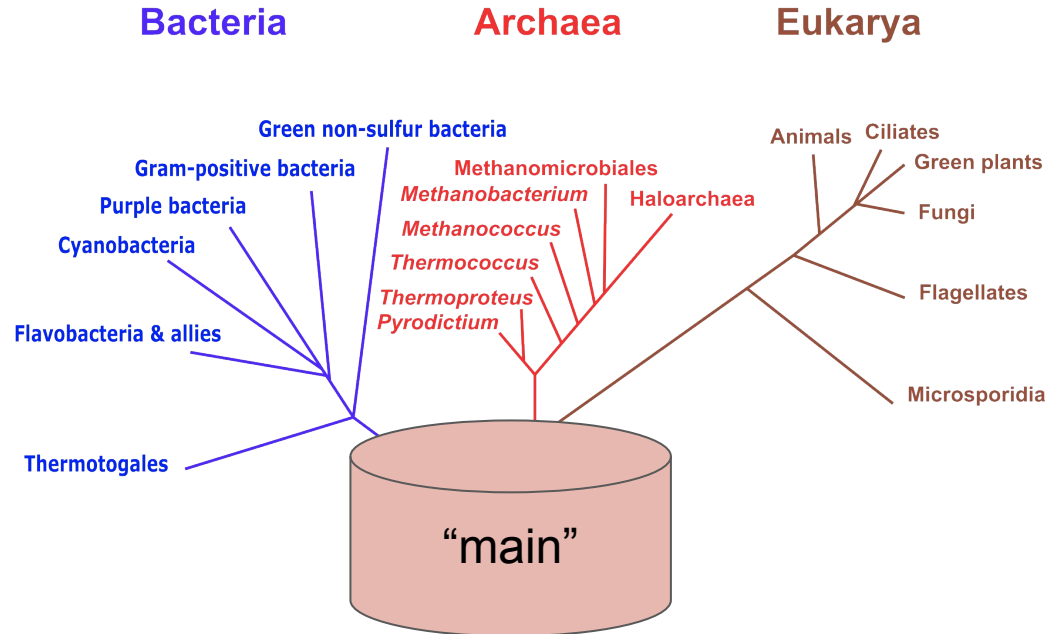


Chiswick Chap

Creative Commons Attribution-Share Alike 4.0 International

A

“Vertical” sharding

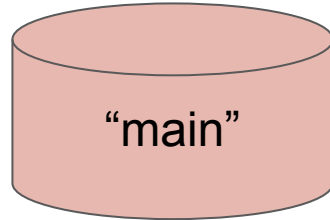


Chiswick Chap

Creative Commons Attribution-Share Alike 4.0 International

A

“main”

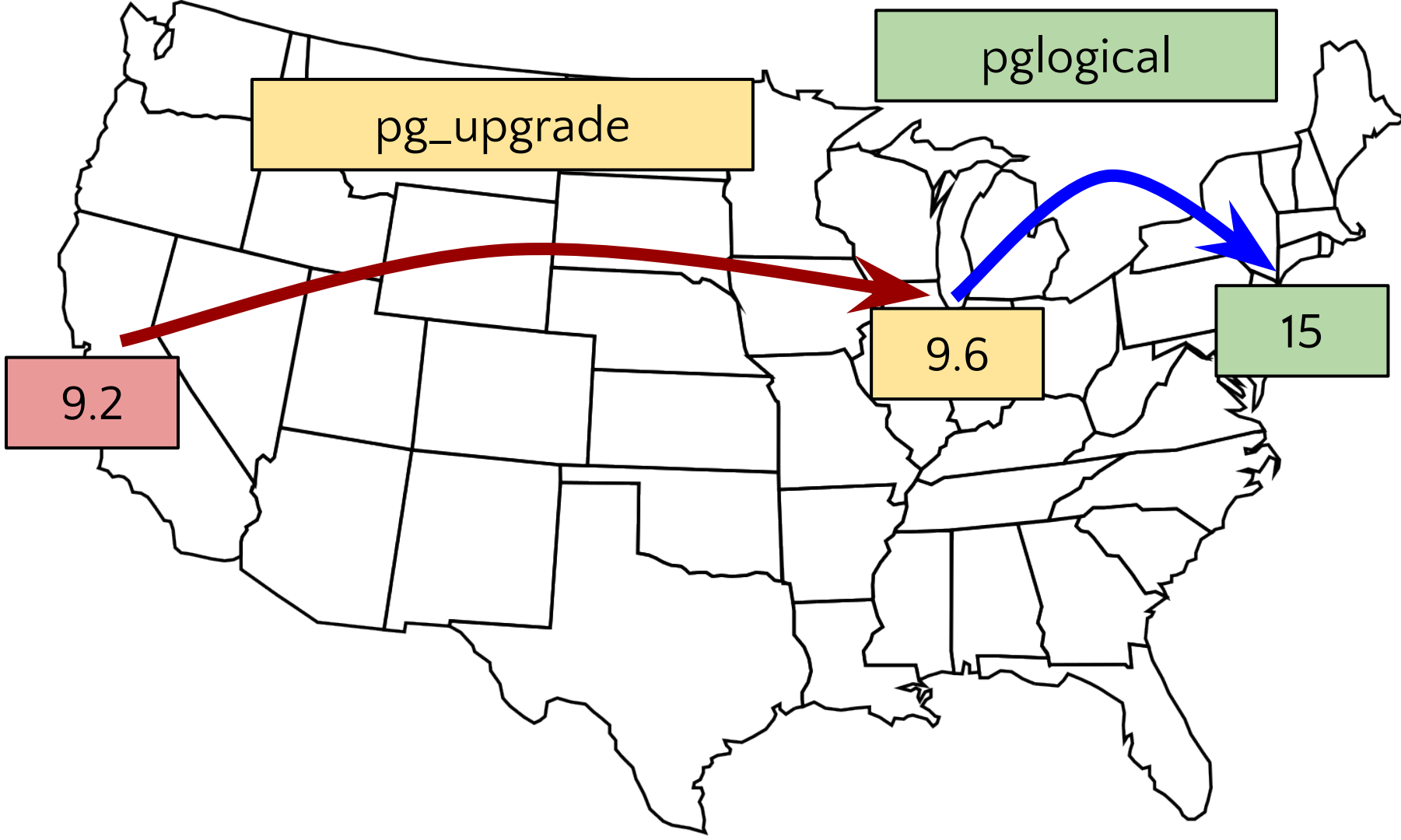


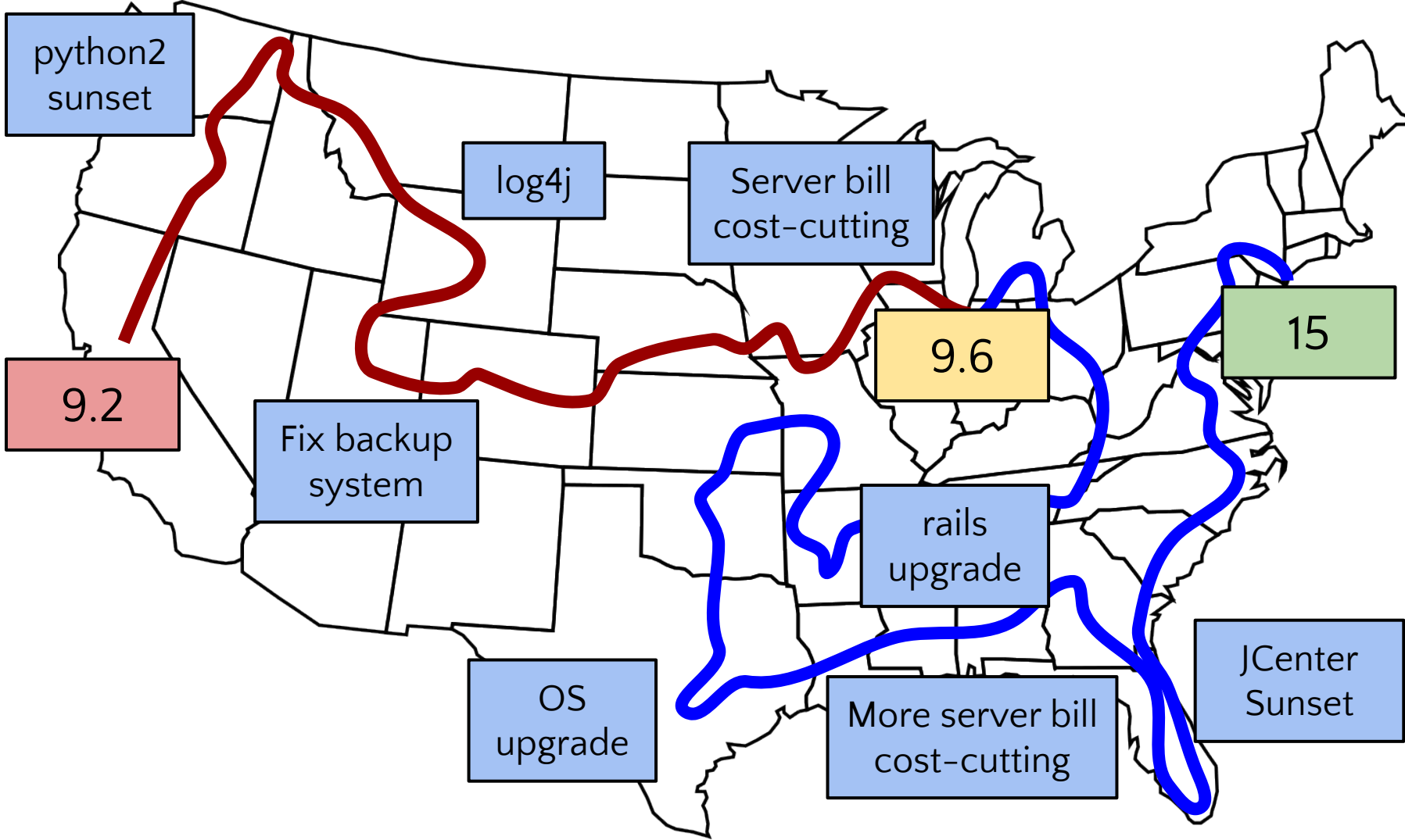
“main”

A

What is “main”?

- 600 tables
- 5 TB
- “Creative” use of PL/pgSQL functions
 - and custom types, constraints, etc





Part 1

9.2 -> 9.6

A

A

Why stop at 9.6?

- <https://www.postgresql.org/docs/release/8.4.0/>
- “Release Date: 2009-07-01”

```
commit 2169e42bef9db7e0bdd1bea00b81f44973ad83c8
Author: Neil Conway <neilc@samurai.com>
Date: Sun Mar 30 04:08:15 2008 +0000
```

Enable 64-bit integer datetimes by default, per previous discussion.

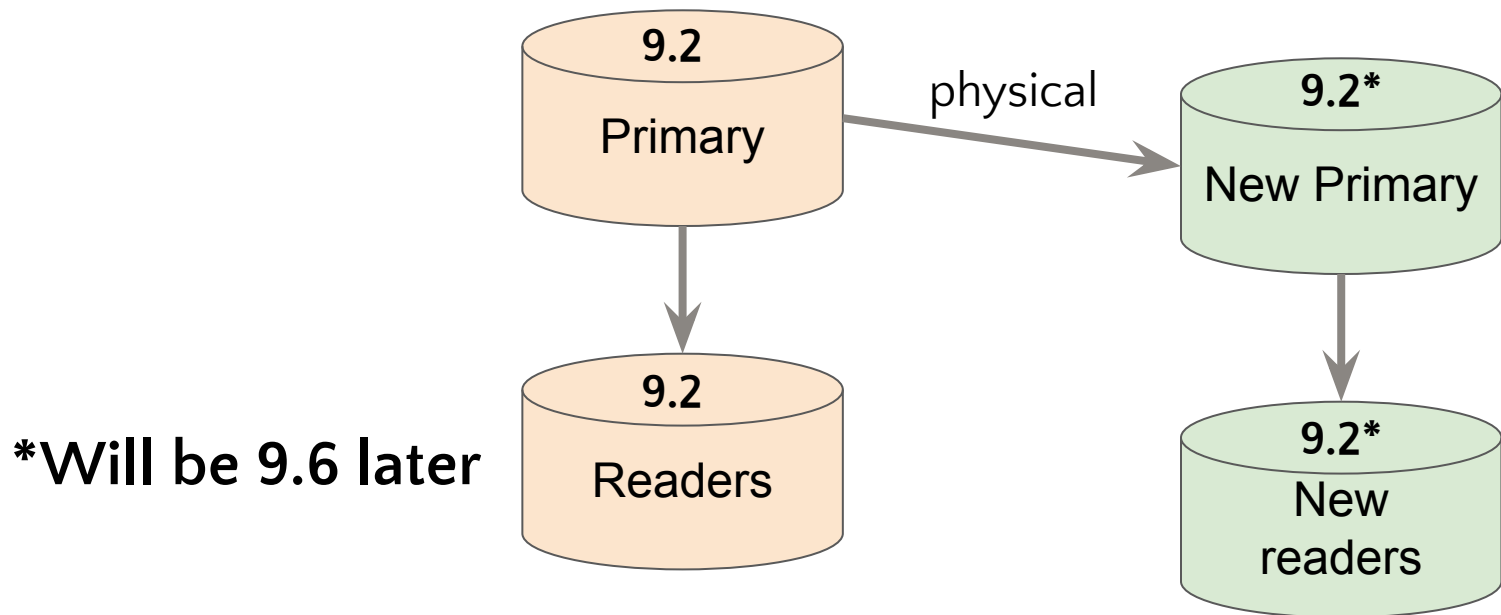
This requires a working 64-bit integer type. If such a type cannot be found, "--disable-integer-datetimes" can be used to switch back to the previous floating point-based datetime implementation.

A `--disable-integer-datetimes`

- `pg_upgrade --check` **complains**
- => **We built our own postgres binary**
 - And we did that all the way to 9.2
- **Postgres 10 removed this flag**

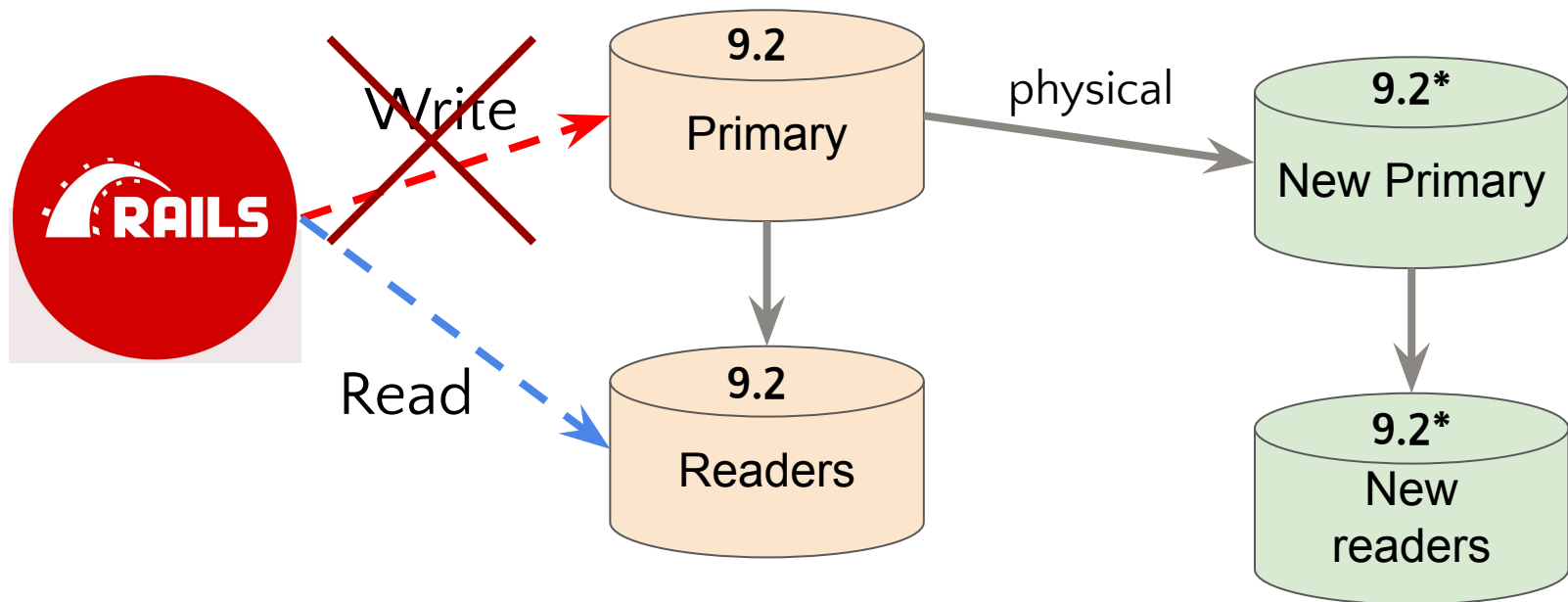
A pg_upgrade with streaming replicas

Step 1: Bring up “new tree” replicas



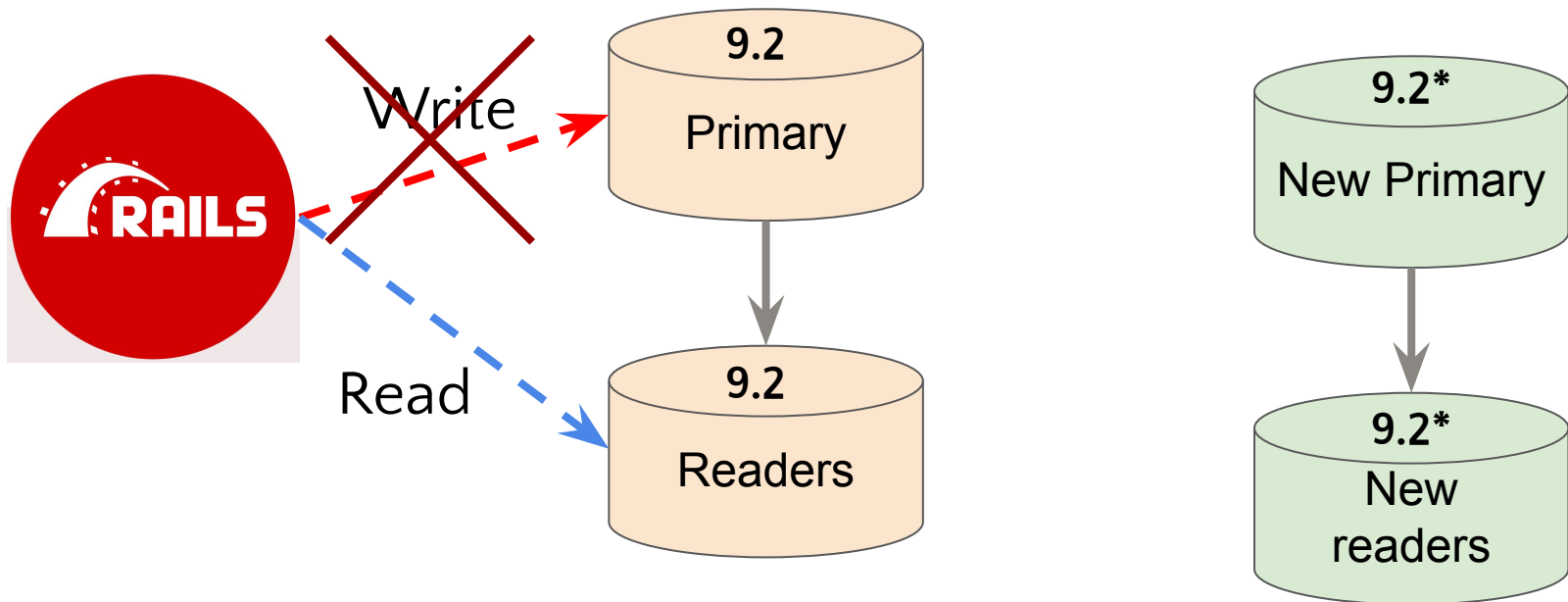
A pg_upgrade with streaming replicas

Step 2: Block writes



A pg_upgrade with streaming replicas

Step 3: Promote new primary



A pg_upgrade with streaming replicas

Step 4: Follow the simple 17 step guide

<https://www.postgresql.org/docs/9.6/pgupgrade.html>

(But also reference the latest version of those docs:

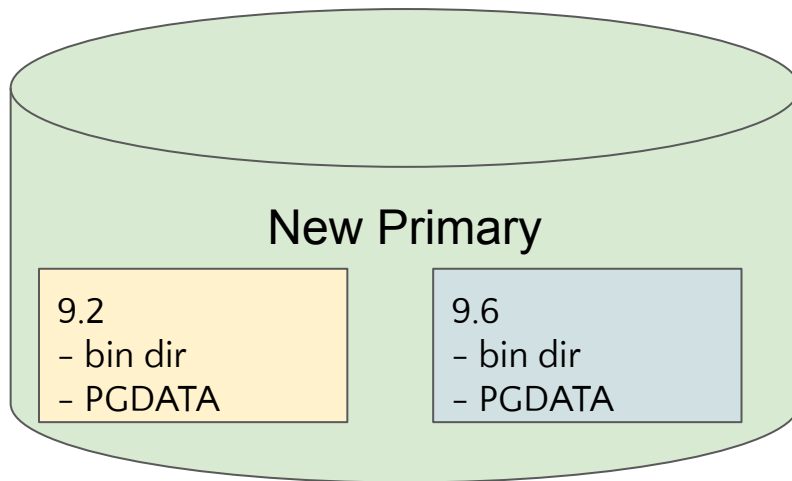
<https://www.postgresql.org/docs/17/pgupgrade.html>)

Diving into the details

A

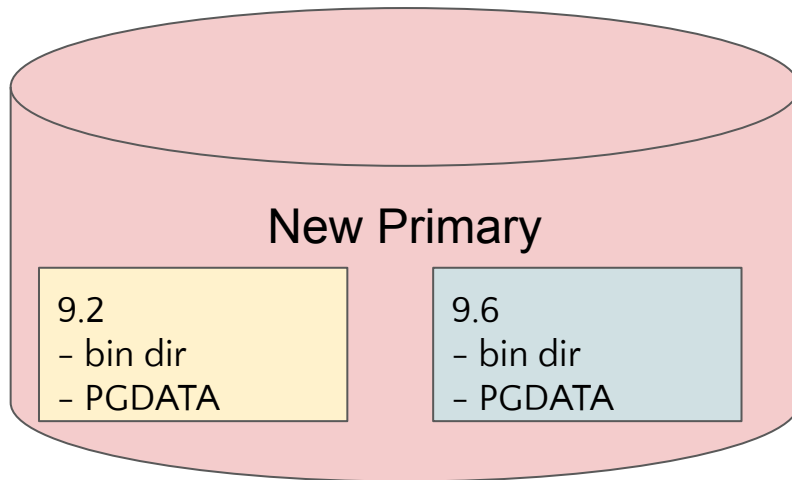
A pg_upgrade with streaming replicas

Step 4a: Install newer version, and initdb



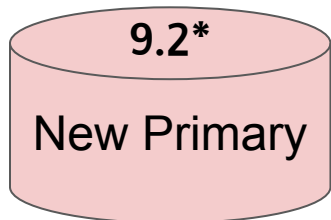
A pg_upgrade with streaming replicas

Step 4b: Stop postgres on new primary
(but keep it running on the replicas)

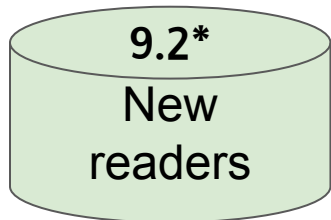


A pg_upgrade with streaming replicas

Step 4c: Check that pg_controldata (checkpoint) matches



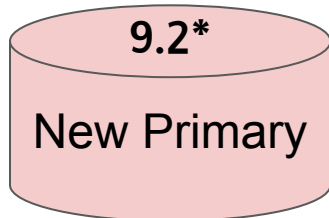
```
$ pg_controldata $PGDATA | grep 'Latest checkpoint location'  
Latest checkpoint location: 0/41933E90
```



```
$ pg_controldata $PGDATA | grep 'Latest checkpoint location'  
Latest checkpoint location: 0/41933E90
```

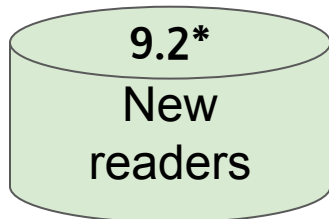

A pg_upgrade with streaming replicas

Step 4c: Check that pg_controldata (checkpoint) matches



```
$ pg_controldata $PGDATA | grep 'Latest checkpoint location'  
Latest checkpoint location: 0/4193BE90
```

Do not proceed unless they match

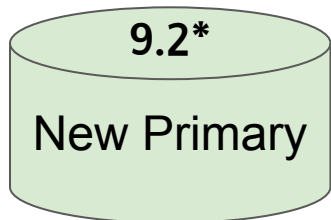


```
$ pg_controldata $PGDATA | grep 'Latest checkpoint location'  
Latest checkpoint location: 0/4193BE90
```

A

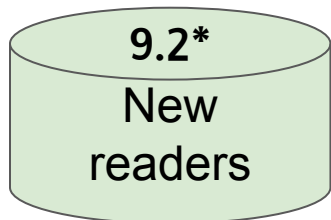
Aside: what we did before step 4b

Before stopping postgres on new primary: checkpoint in loop



```
postgres=# checkpoint;  
$ pg_controldata $PGDATA | grep 'Latest checkpoint location'  
Latest checkpoint location: 0/41933E90
```

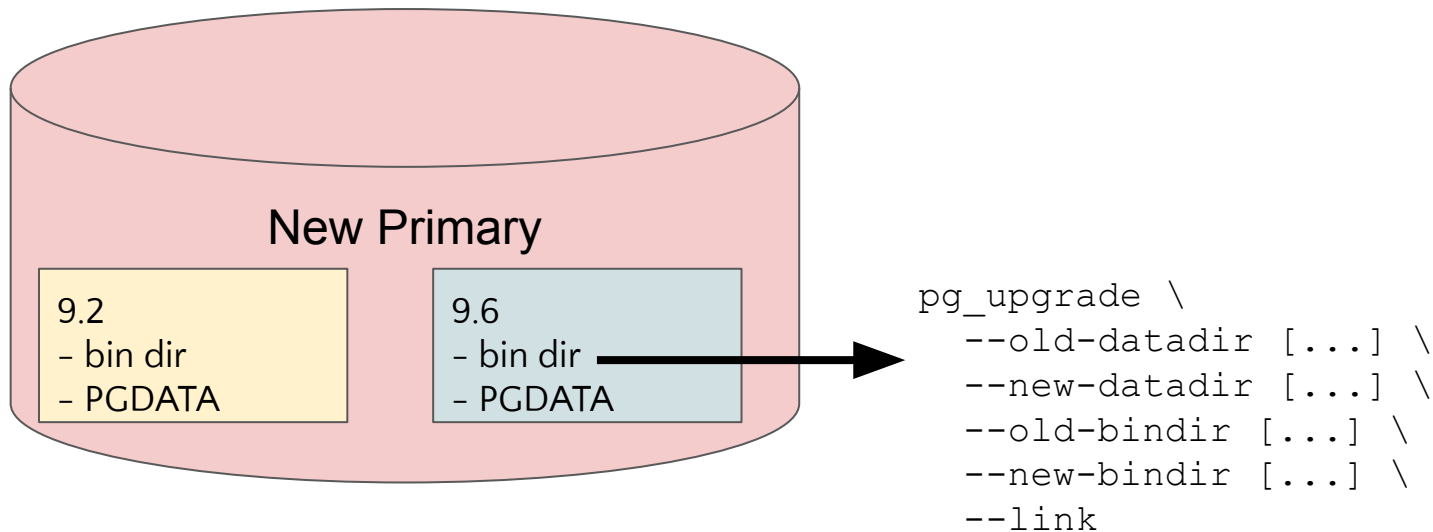
Do not proceed unless they match



```
postgres=# checkpoint;  
$ pg_controldata $PGDATA | grep 'Latest checkpoint location'  
Latest checkpoint location: 0/41933E90
```

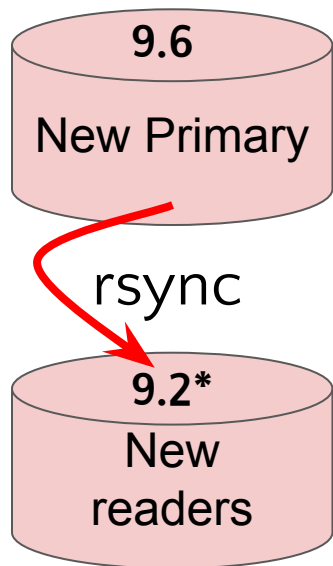
A pg_upgrade with streaming replicas

Step 4d: Run pg_upgrade with --link (ensure no errors)



A pg_upgrade with streaming replicas

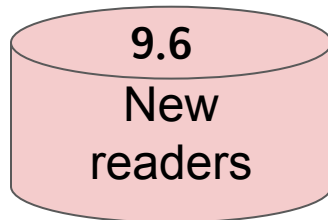
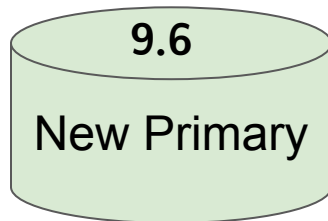
Step 4e: Stop replicas, install new pg and... run rsync (?)



```
rsync \
  --archive \
  --delete \
  --hard-links \
  --size-only \
  --no-inc-recursive \
  /opt/PostgreSQL/9.2 \
  /opt/PostgreSQL/9.6 \
  standby.example.com:/opt/PostgreSQL
```

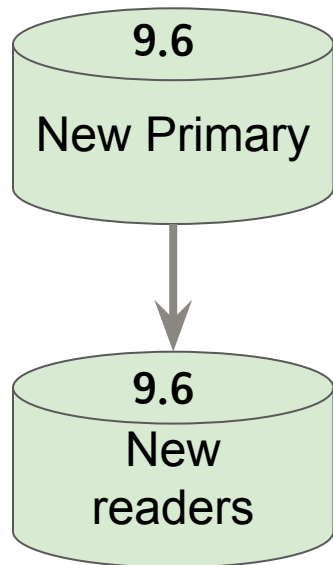
A pg_upgrade with streaming replicas

Step 4f: Start new primary



A pg_upgrade with streaming replicas

Step 4g: Then start streaming replicas



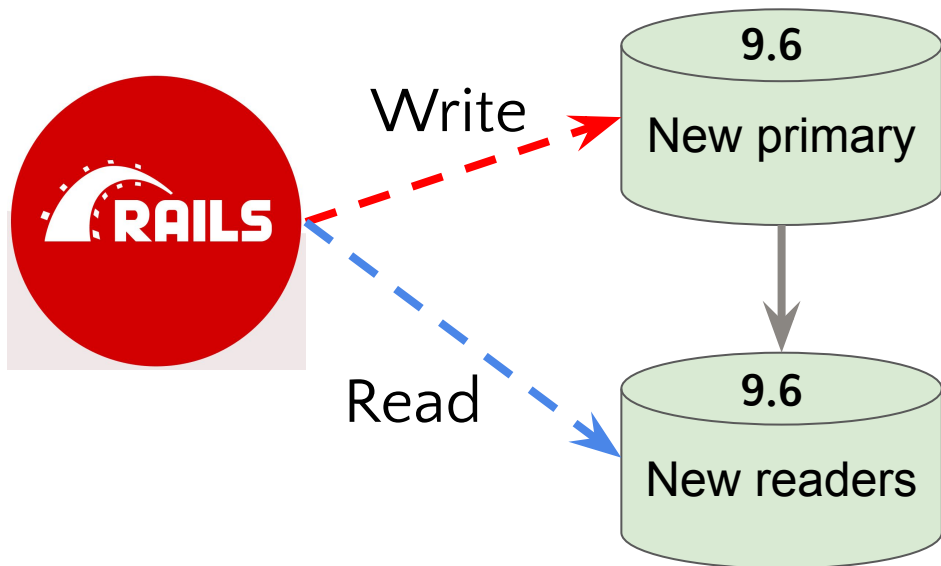
A pg_upgrade with streaming replicas

Step 5: Run ANALYZE on all tables

- Academia-specific: run *before* resuming reads/writes
- We have caught corruption at this stage
- This extends the required maintenance window

A pg_upgrade with streaming replicas

Step 6: Point application at new nodes, resume writes



A

Should I do this?

No

A

Should I do this?

~~No~~

It depends...

A

What could go wrong?

- Standby corruption
- Postgres FM | pg_upgrade: the tricky and dangerous parts
- pgsql-hackers: pg_upgrade instructions involving "rsync --size-only" might lead to standby corruption?
- Adyen: Database corruption in PostgreSQL: our journey to improving our upgrade process

A

Why might you want to do this anyway?

- **If you have no other choice**
- **People run this in production, and (some) say it works**
 - This is what matters

Part 2

9.6 -> 15

A

A

How to get from 9.6 to 15

- **✗ pg_dump + pg_restore**
 - Too slow, we can't shut the site down for 2 days


A

How to get from 9.6 to 15

- **✗ pg_dump + pg_restore**
 - Too slow, we can't shut the site down for 2 days
- **✗ pg_upgrade**
 - `--disable-integer-datetime`
 - Postgres 10 removed support for that compile flag

A

How to get from 9.6 to 15

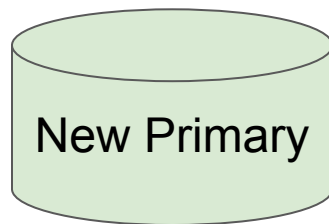
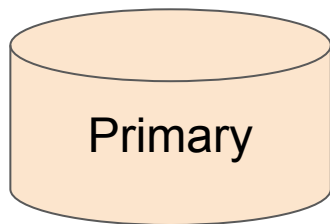
-  **logical replication**
 - “No” downtime
 - Keep logical replica in sync real-time
 - Built-in needs postgres 10 or higher...
 - ...but pglogical extension works on 9.6

A**Recommended reading****PostgreSQL 12 High Availability Cookbook, Shaun Thomas**

- **Chapter 7: PostgreSQL Replication -> pglogical**
 - (if you still need pglogical)
- **Chapter 15: Zero-downtime Upgrades**

A Logical upgrades (high level)

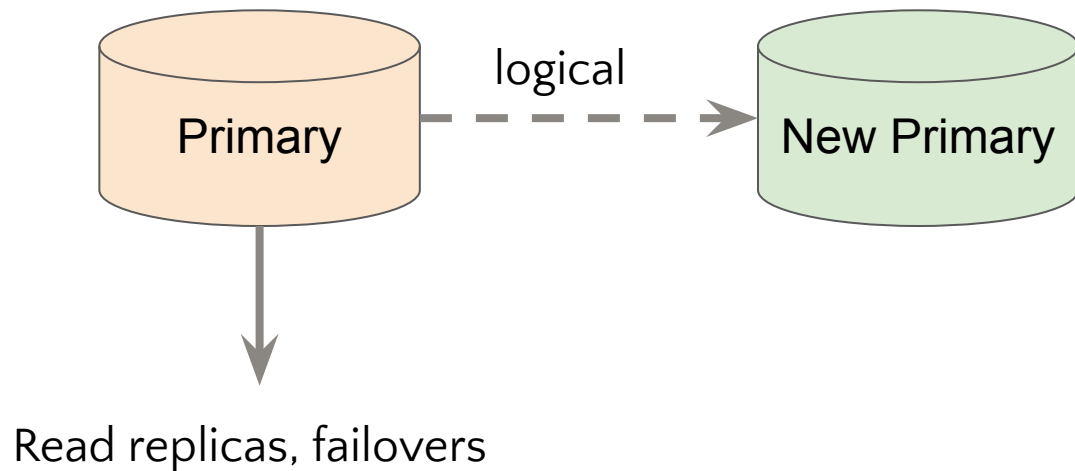
Step 1: Make a new main 15 DB, with the same schema



Read replicas, failovers

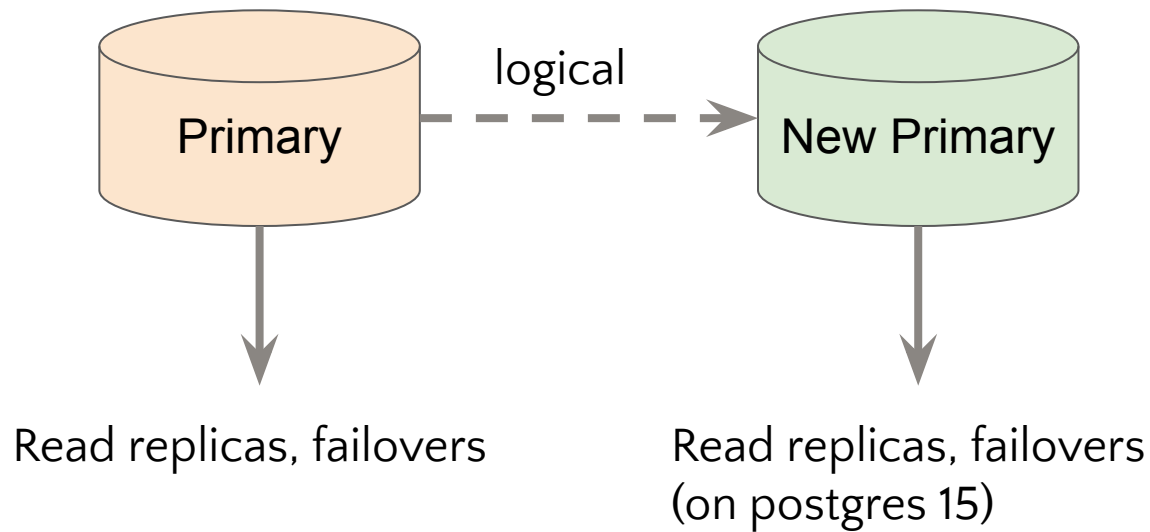
A Logical upgrades (high level)

Step 2: Copy data, then stay in sync with changes



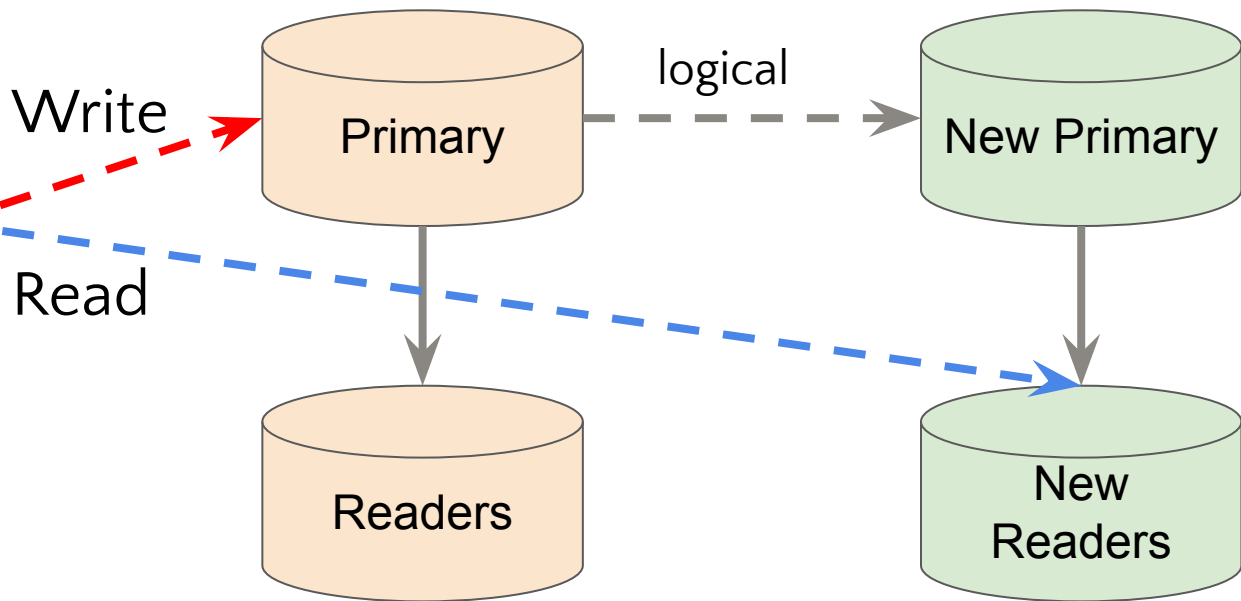
A Logical upgrades (high level)

Step 3: Bring up new tree (streaming replication)



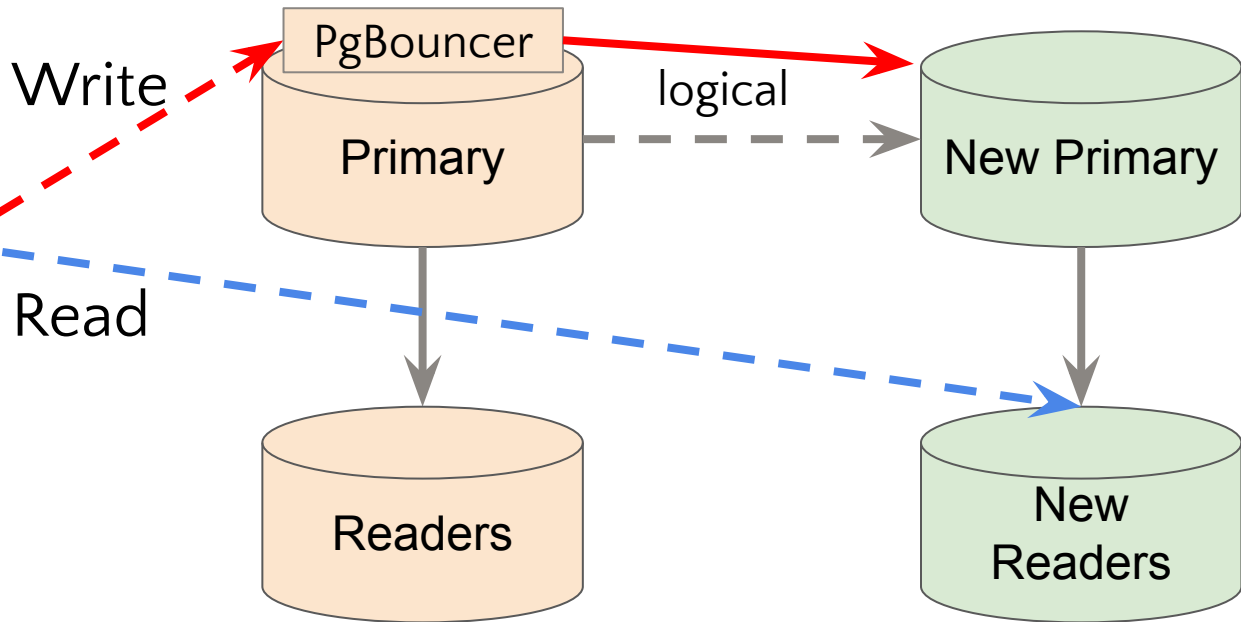
A Logical upgrades (high level)

Step 4: Test reads



A Logical upgrades (high level)

Step 5: Switch writes



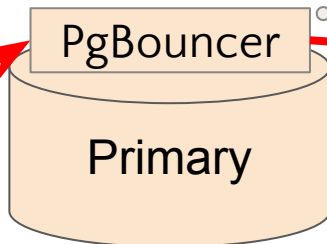
A Logical upgrades (high level)

Step 5: Switch writes

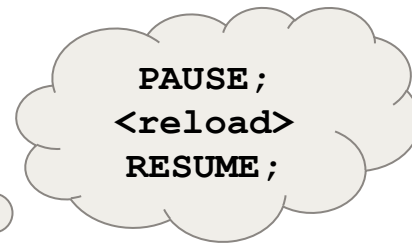
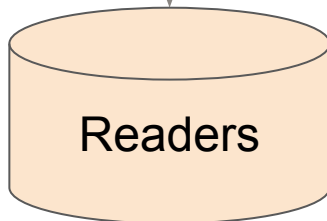
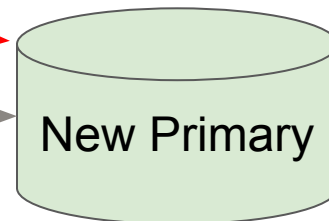


Write

Read



logical



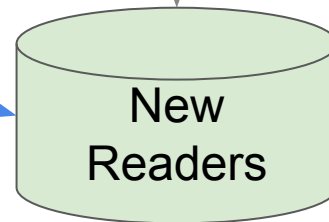
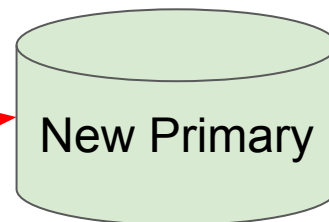
A Logical upgrades (high level)

Step 5: Switch writes



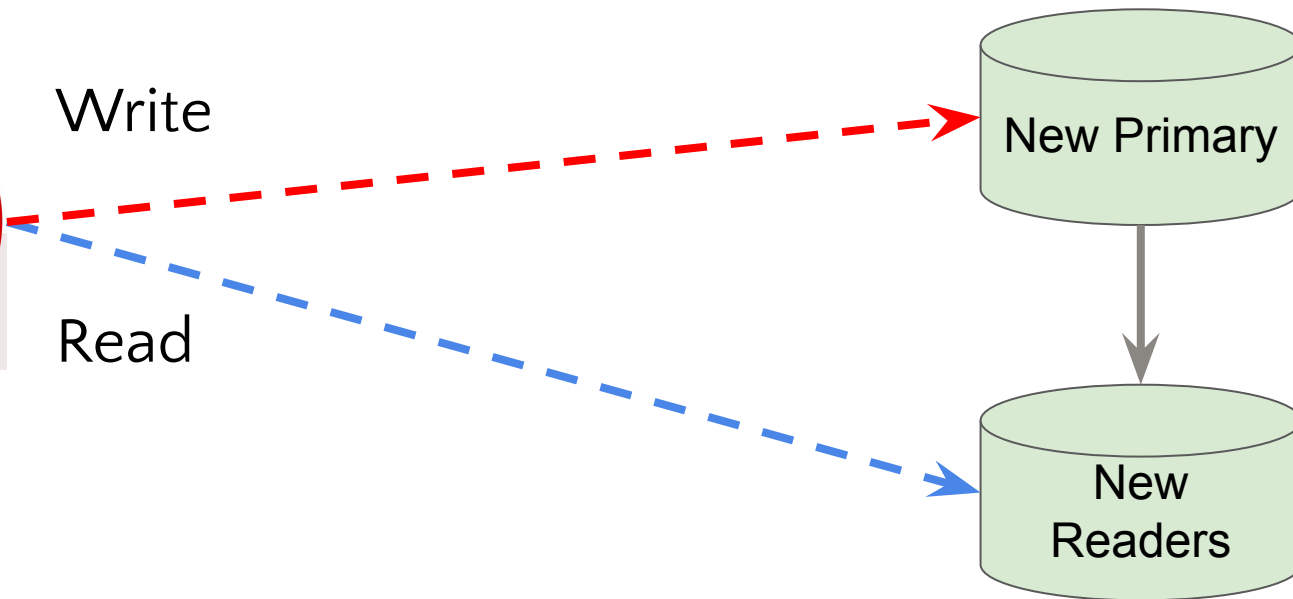
Write

Read



New Primary

New Readers



Diving into the details

A

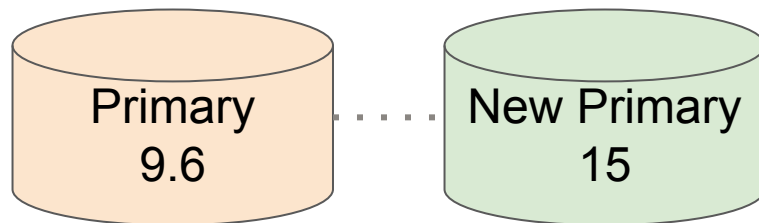
A

“Solved problems”

- **Breaking changes**
 - Parsing pg_dump output
 - Tons of surgical fixes
- **Creating indexes and constraints *after* data load**
- **Long-running initial syncs**
 - WAL backlog
- **Smooth backup solution transition**

A Monitoring initial syncs

On the destination (e.g. postgres 15)



```
SELECT
```

```
    pg_stat_progress_copy.tuples_processed,
```

```
FROM
```

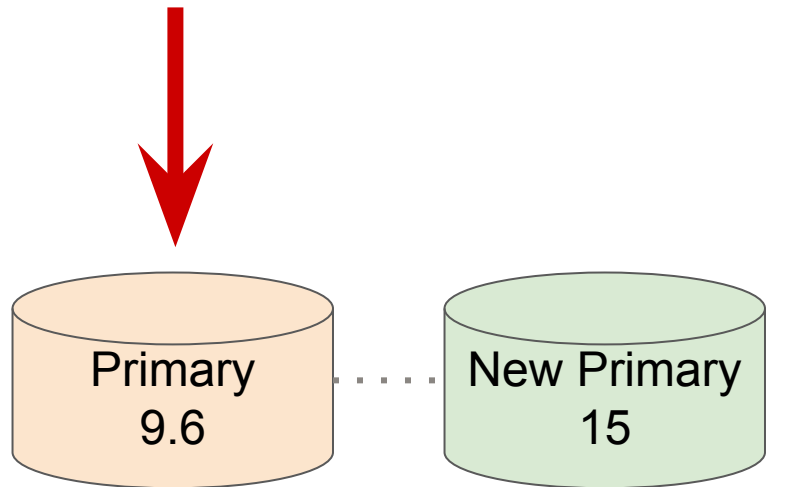
```
    pg_stat_progress_copy;
```

tuples_processed

A Monitoring initial syncs

On the source (e.g. postgres 9.6)

```
SELECT
    c.reltuples AS row_estimate
FROM
    pg_class c
    LEFT JOIN pg_namespace n
    ON n.oid = c.relnamespace
WHERE
    n.nspname = 'public'
    AND c.relname = 'followings';
```



tuples_processed
row_estimate

A

Monitoring initial syncs

$$\text{approx percent done}^* = \frac{\text{tuples_processed}}{\text{row_estimate}}$$

*(per table)

A Complications (the highlights)

1. Schema changes (migrations)
2. Duplicate strings, in spite of a **UNIQUE** index...
3. pglogical bugs

A

1. Schema changes (migrations)

- 3x-5x per week on average
- Logical replication has no schema change support
- pgl_ddl_deploy extension
 - “Transparent Logical DDL Replication”
- **Magic?**

A

pgl_ddl_deploy: the devil is in the corner cases

- **No CREATE INDEX support (by design)**
 - => manually detect new indexes and create them on 15
 - pg_query ruby gem helped
- **Not all DDL was guaranteed to work**
- **We just didn't trust it to not break**
 - => cumbersome QA + sign-off process

A

2. Duplicate strings, in spite of a UNIQUE index

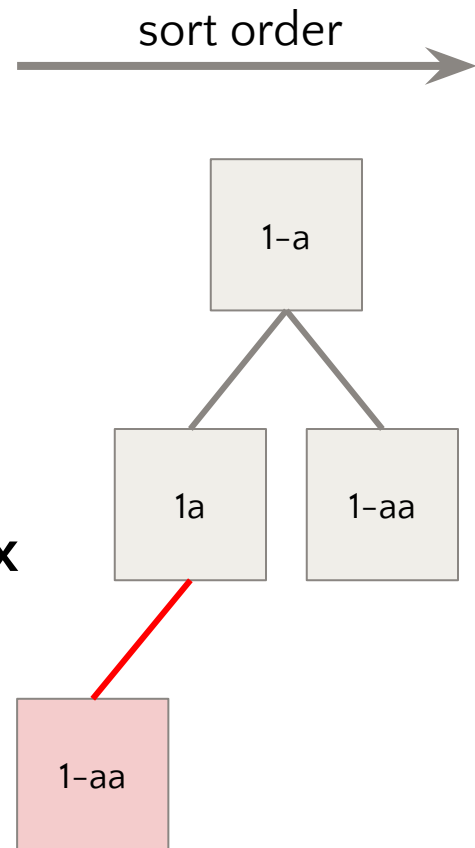
```
SELECT a.*  
FROM table_name a  
JOIN table_name b ON a.name = b.name  
WHERE a.id = 12345;
```

- There is a unique index on table_name.name
- But two rows were returned

A

Likely culprit: “in-place” OS upgrade

- Past: ubuntu 14.04 -> 16.04
- Physical replication to a 16.04 replica
- OS upgrade = “glibc” upgrade
 - What postgres uses to sort strings
 - Every version changes its mind about sorting
- String uniqueness enforced by btree index
 - Index persists the old sorting
 - So ever since: index (slightly) wrong



A

3. pglogical bugs

- **Confusing error messages**
- **Sometimes crashes -> data loss**
 - From incorrect replication slot handling
 - Had to monitor for this
- **CREATE INDEX CONCURRENTLY considered harmful?**
 - <https://github.com/2ndQuadrant/pglogical/issues/469>

Favorite new features



A

Nick Meyer's esoteric list of favorite features (9.3-9.6)

- **lock_timeout**
- **pg_stat_wal_receiver**
- **pg_stat_progress_vacuum**
- **VACUUM improvements**

A

Things we probably already know about (10-15)

- **Built-in logical replication**
- **Declarative partitioning**
- **MERGE**

A

Nick Meyer's esoteric list of favorite features (10-15)

- **pg_stat_statements_info**

A

Nick Meyer's esoteric list of favorite features (10-15)

- **pg_stat_statements_info**
- **pg_stat_progress_***

A

Nick Meyer's esoteric list of favorite features (10-15)

- `pg_stat_statements_info`
- `pg_stat_progress_*`
 - `pg_stat_progress_create_index`
 - `pg_stat_progress_copy`
 - ~~`pg_stat_progress_basebackup`~~ (`pgbackrest info`)

A

Nick Meyer's esoteric list of favorite features (10-15)

- **pg_stat_statements_info**
- **pg_stat_progress_***
 - pg_stat_progress_create_index
 - pg_stat_progress_copy
 - ~~pg_stat_progress_basebackup~~ (pgbackrest info)
- **pg_sequences view**

A

Nick Meyer's esoteric list of favorite features (10-15)

- **pg_stat_statements_info**
- **pg_stat_progress_***
 - pg_stat_progress_create_index
 - pg_stat_progress_copy
 - ~~pg_stat_progress_basebackup~~ (pgbackrest info)
- **pg_sequences view**
- **pg_hba_file_rules**

A

Nick Meyer's esoteric list of favorite features (10-15)

- **pg_stat_statements_info**
- **pg_stat_progress_***
 - pg_stat_progress_create_index
 - pg_stat_progress_copy
 - ~~pg_stat_progress_basebackup~~ (pgbackrest info)
- **pg_sequences view**
- **pg_hba_file_rules**
- **psql --csv**

A

Nick Meyer's esoteric list of favorite features (10-15)

- **pg_stat_statements_info**
- **pg_stat_progress_***
 - pg_stat_progress_create_index
 - pg_stat_progress_copy
 - ~~pg_stat_progress_basebackup~~ (pgbackrest info)
- **pg_sequences view**
- **pg_hba_file_rules**
- **psql --csv**
- **max_slot_wal_keep_size**

Takeaways



A

1. Be both patient and impatient

- The DB doesn't care about your deadline
- Never “fire and forget” a long running operation
 - Build your own progress bar
 - `pg_stat_progress_copy`
 - `pg_stat_progress_create_index`
 - `df -h`

A

2. Learning vs doing

- **Expect uncertainty in time estimates**
 - But still remember “Be both patient and impatient”
- **DEFERRABLE constraints**
- **ALTER TYPE can run in a transaction**
 - But only in postgres 12+
- **glibc**

A

3. Celebrate with low-hanging fruit

- **Add a new column with a default value**
 - Thank you postgres 11
- **Teach someone about `pg_stat_progress_copy`**
 - (or `pg_stat_progress_create_index`)

A

4. Let's seek to understand one another

- Solution-splaining
- Can we make upgrades easier for users?



Thanks for listening!

Questions?