GitLab Database Load Balancer

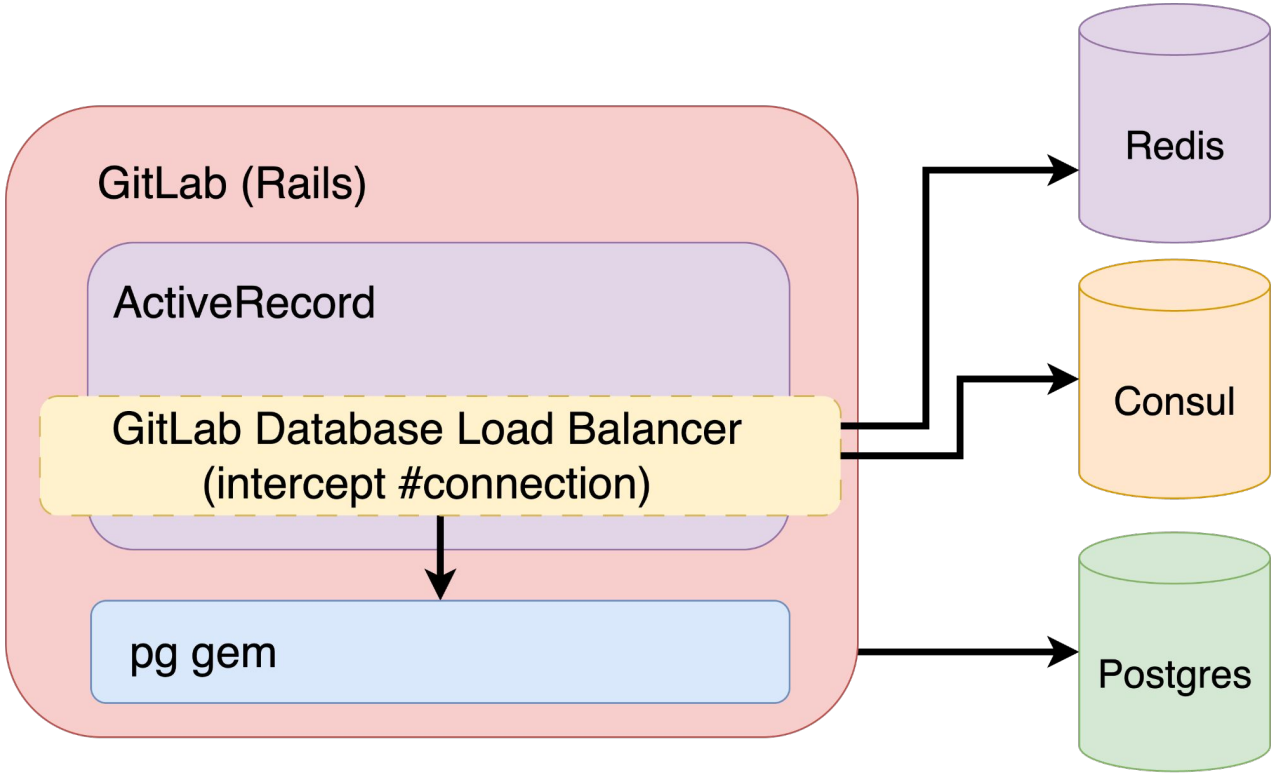
How to make efficient use of database replicas in a large scale web application.

Dylan Griffith, Principal Engineer GitLab
(gitlab.com/DylanGriffith)

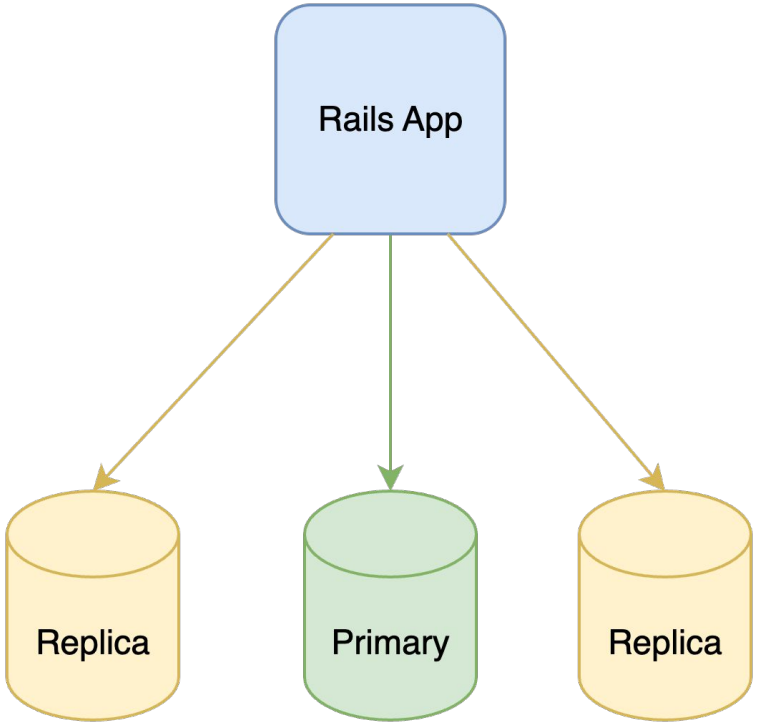
What will this talk cover?

- How GitLab uses replicas to serve read-only traffic
- How GitLab reads stale data without UX bugs
- How GitLab background jobs optimize use of replicas
- How GitLab database load balancer does service discovery
- How GitLab manages a fleet of replicas and connection poolers
- Mistakes we made along the way

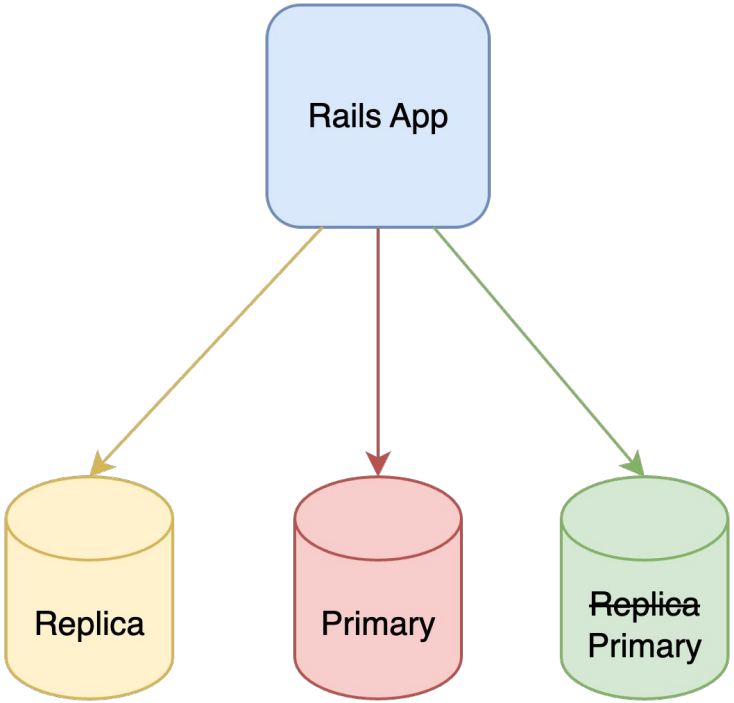
What is GitLab's Database Load Balancer?



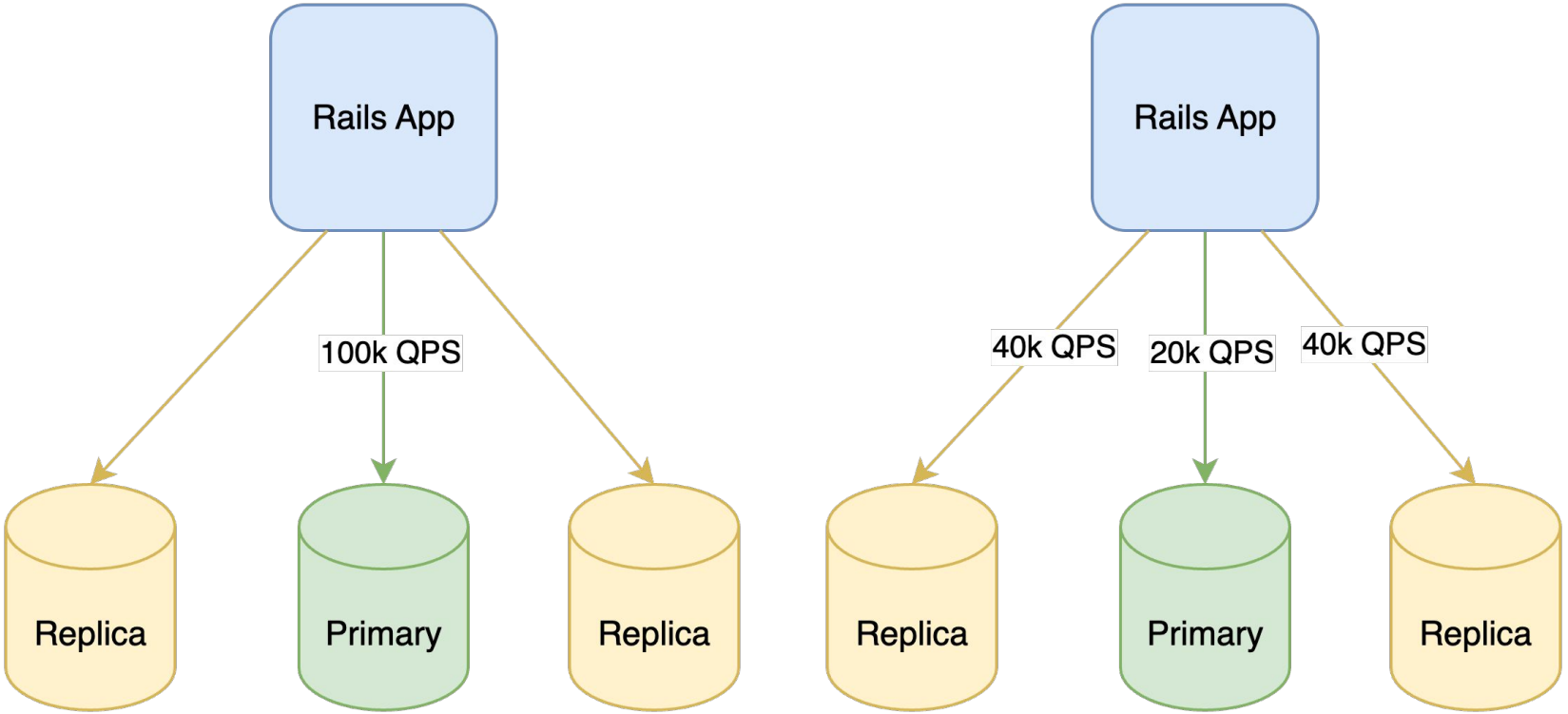
Why do we need replicas?



Why do we need replicas to have large CPUs?



Why do we want to use replicas?



Why can't we just send all read queries to replicas?

New issue

Type

Issue

Title (required)

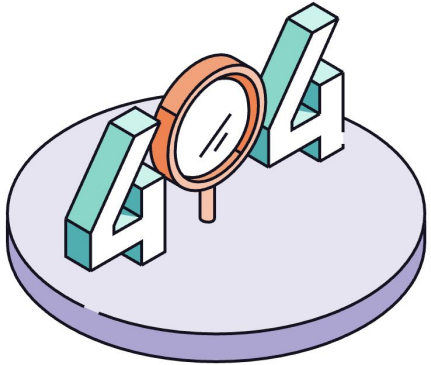
Make an awesome new feature

Description

Add description templates to help your contributors communicate effectively!

Preview | B I S | </> | Write a comment or drag your files here…

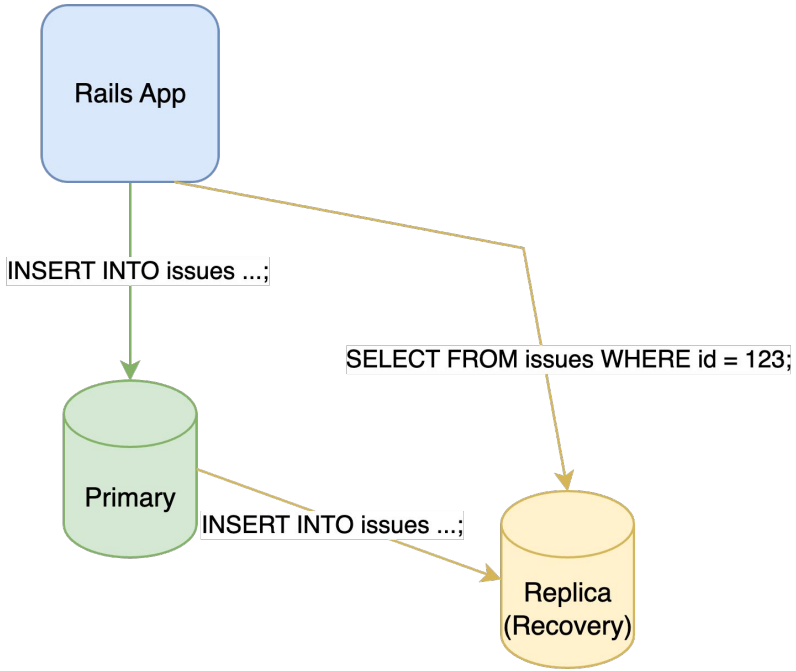
Create issue | Cancel

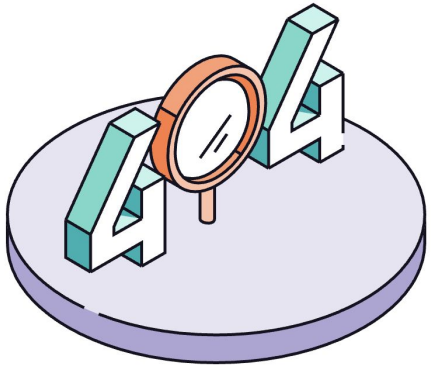


404: Page not found

POST /issues → 302 /issues/123 → GET /issues/123

Why can't we just send all read queries to replicas?

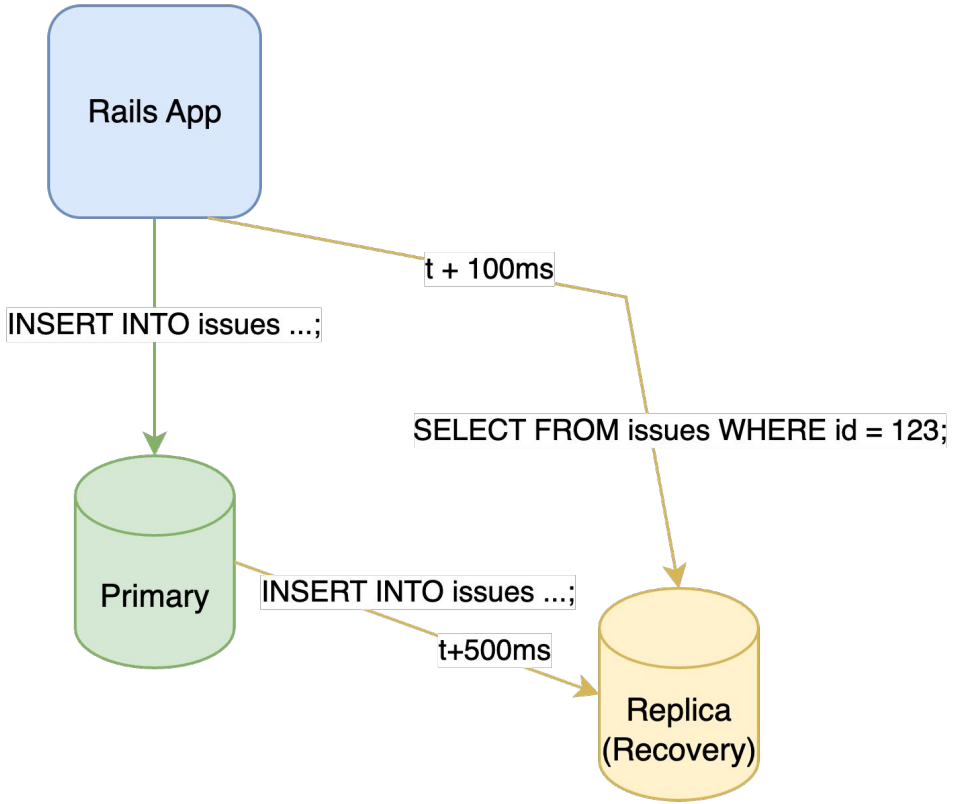




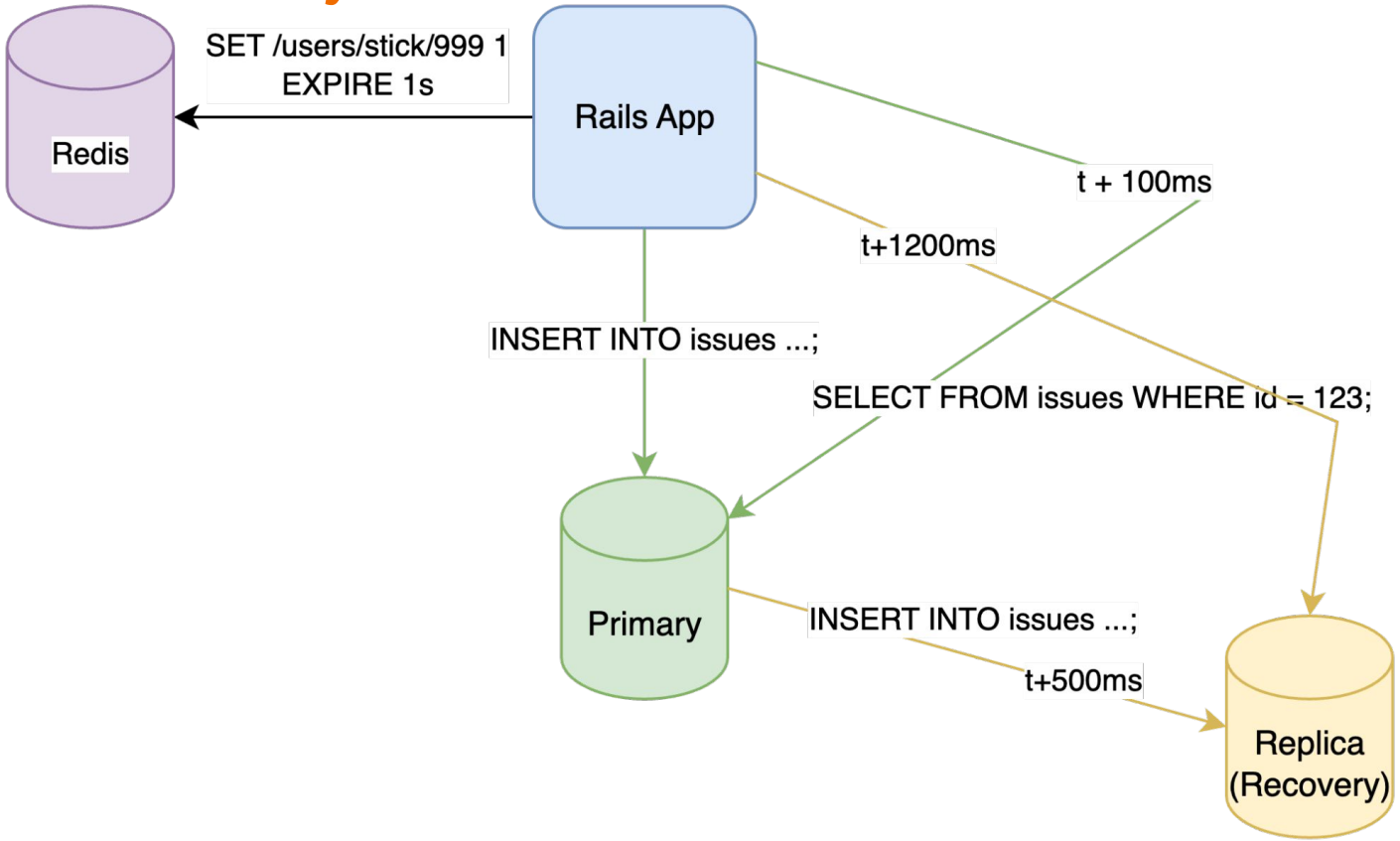
404: Page not found

POST /issues → 302 /issues/123 → GET /issues/123

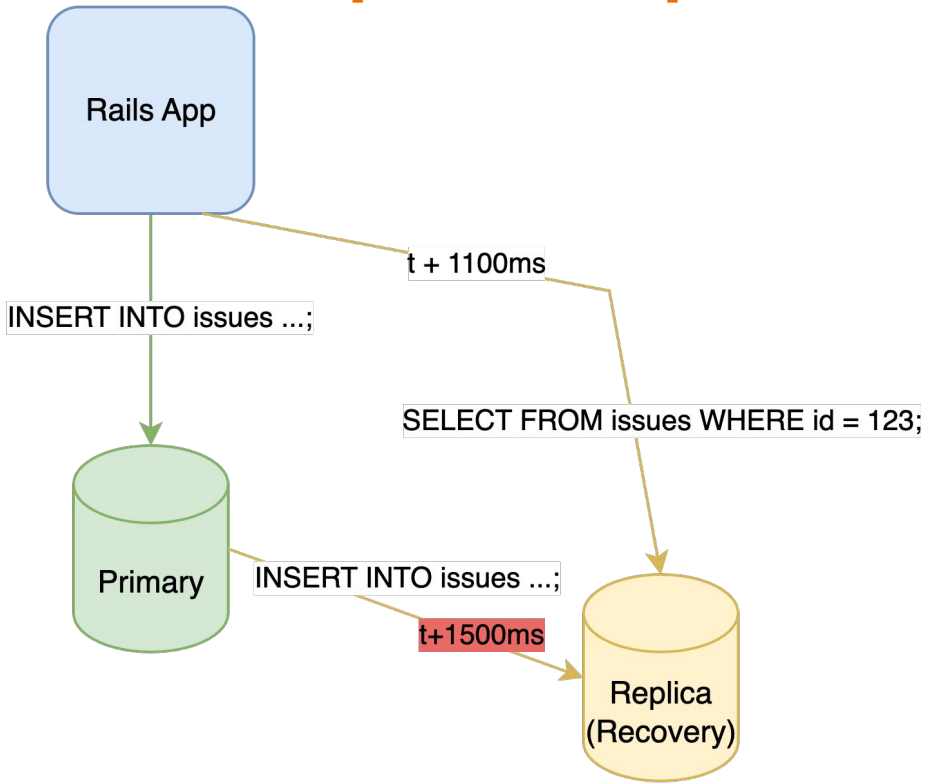
Why can’t we just check if the replica is “near enough”?

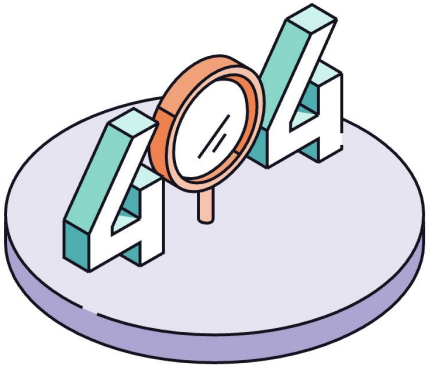


User based sticky sessions



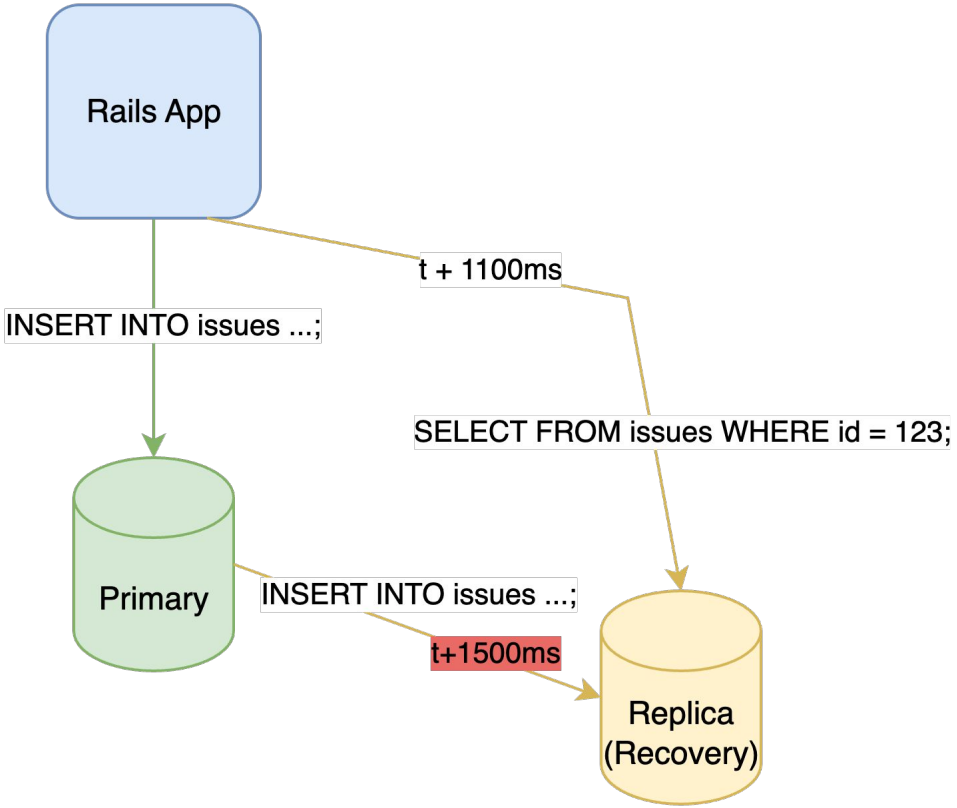
What about replication spikes?

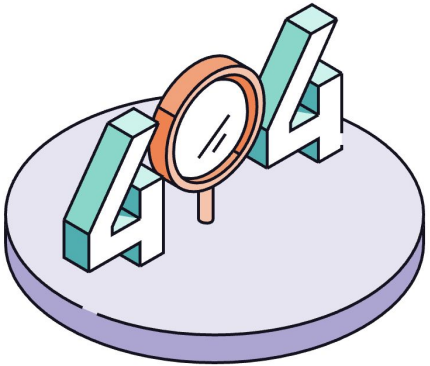




404: Page not found

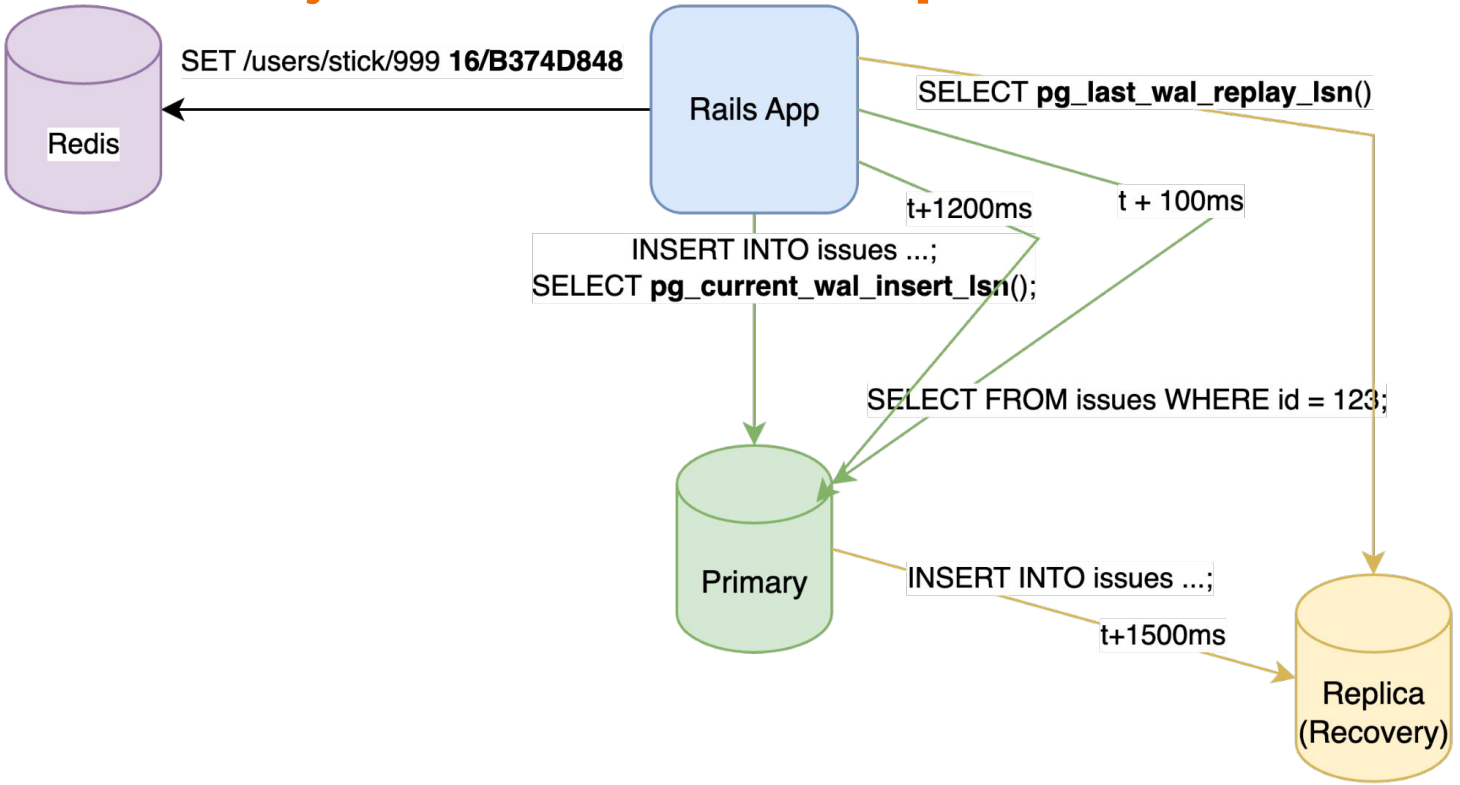
Can’t we just increase the sticking time to 10s?



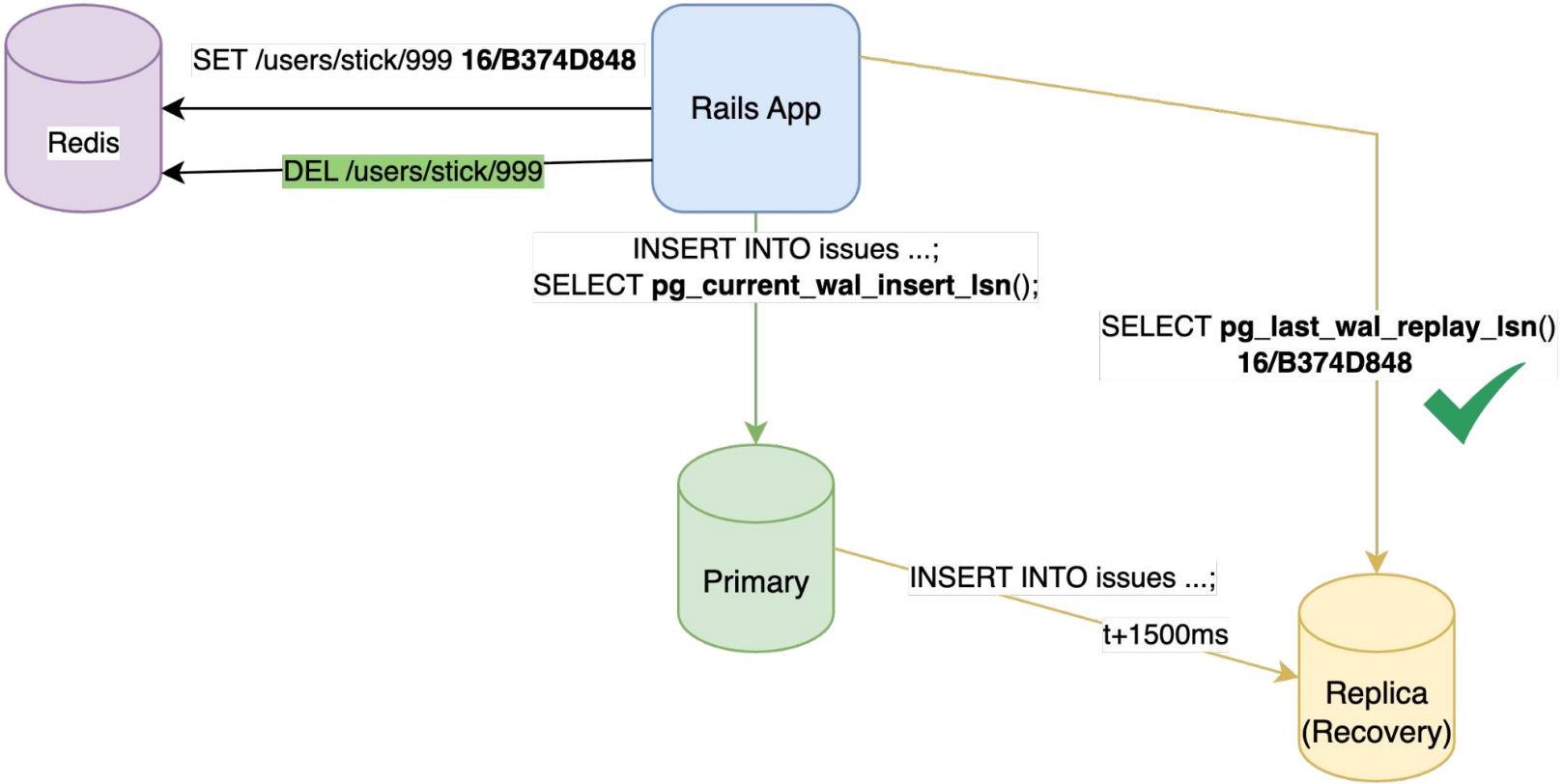


404: Page not found

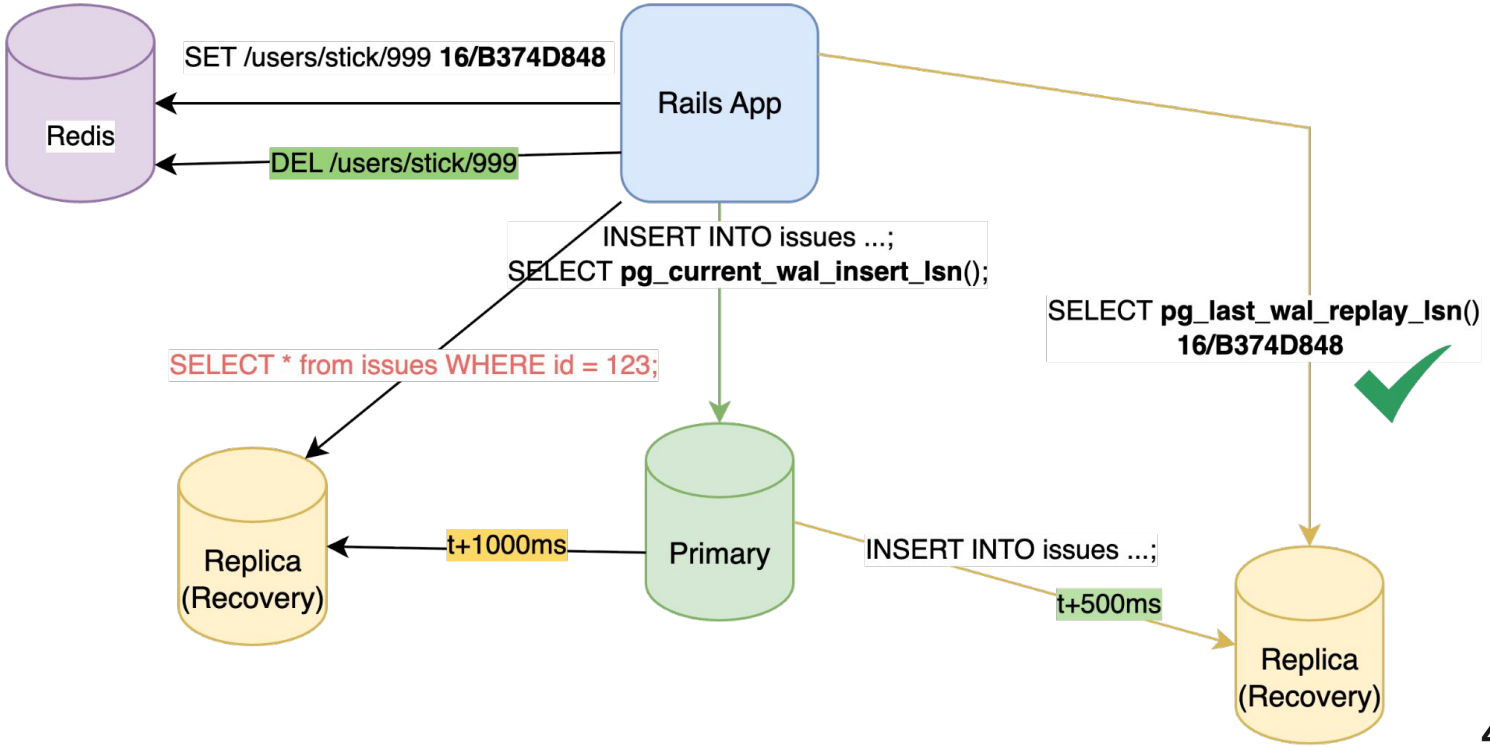
User based sticky sessions with LSN position

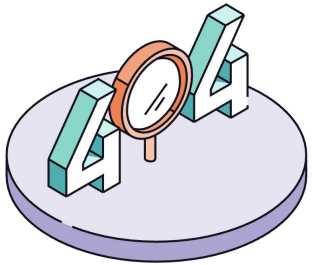


How do we quickly remove user sessions from Redis



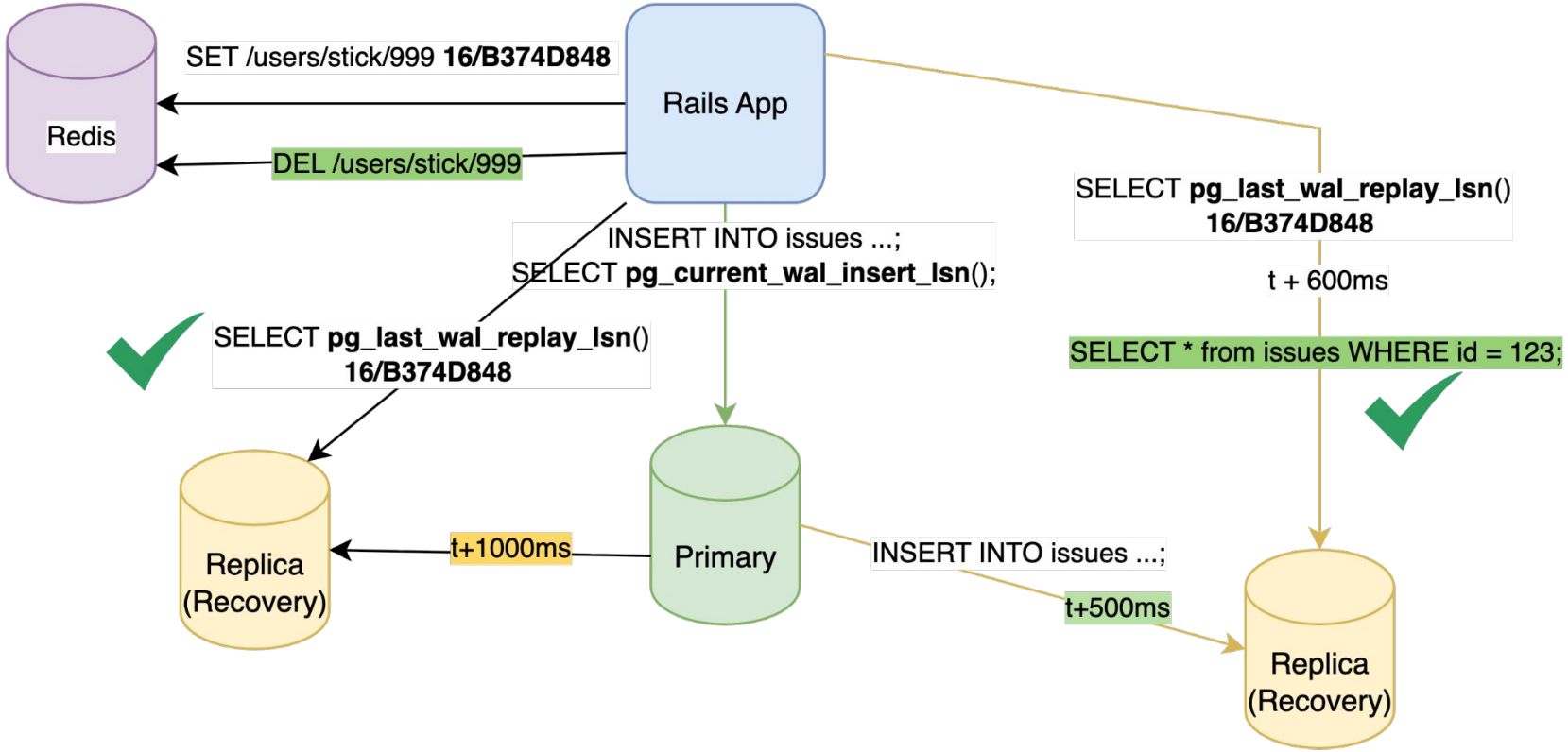
But not all replicas have the same lag...



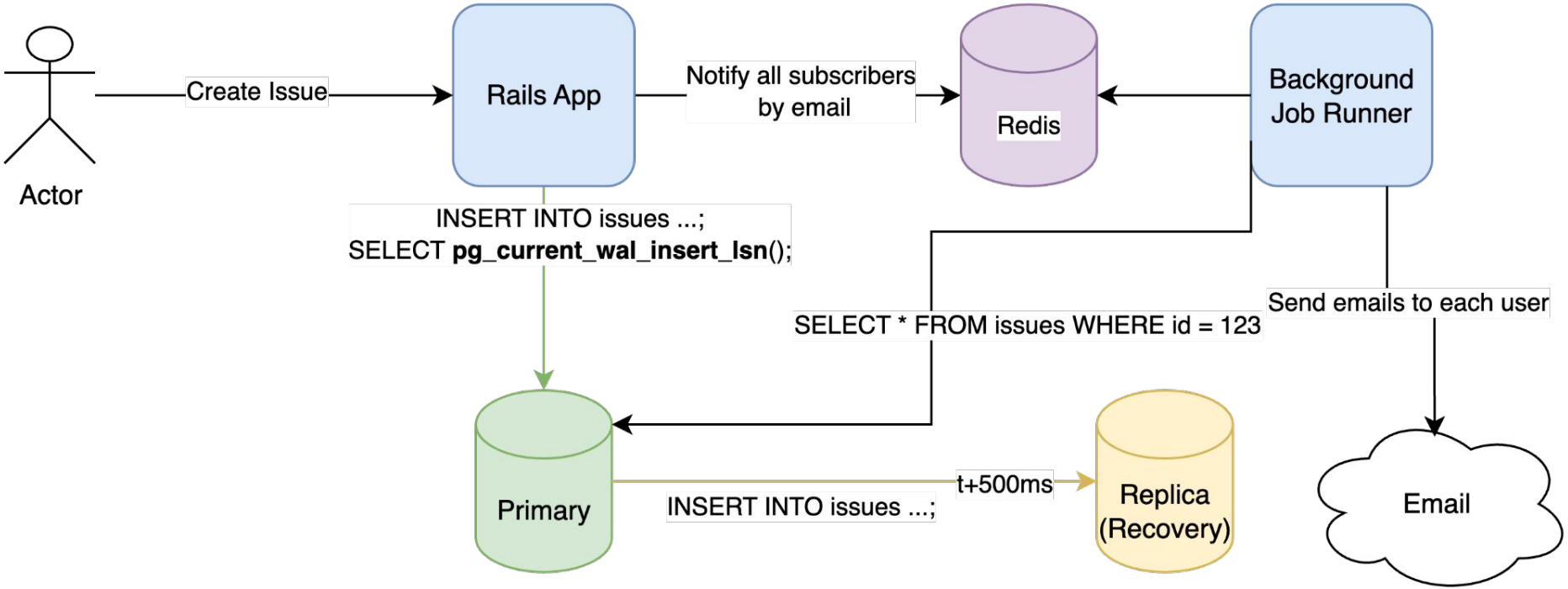


404: Page not found

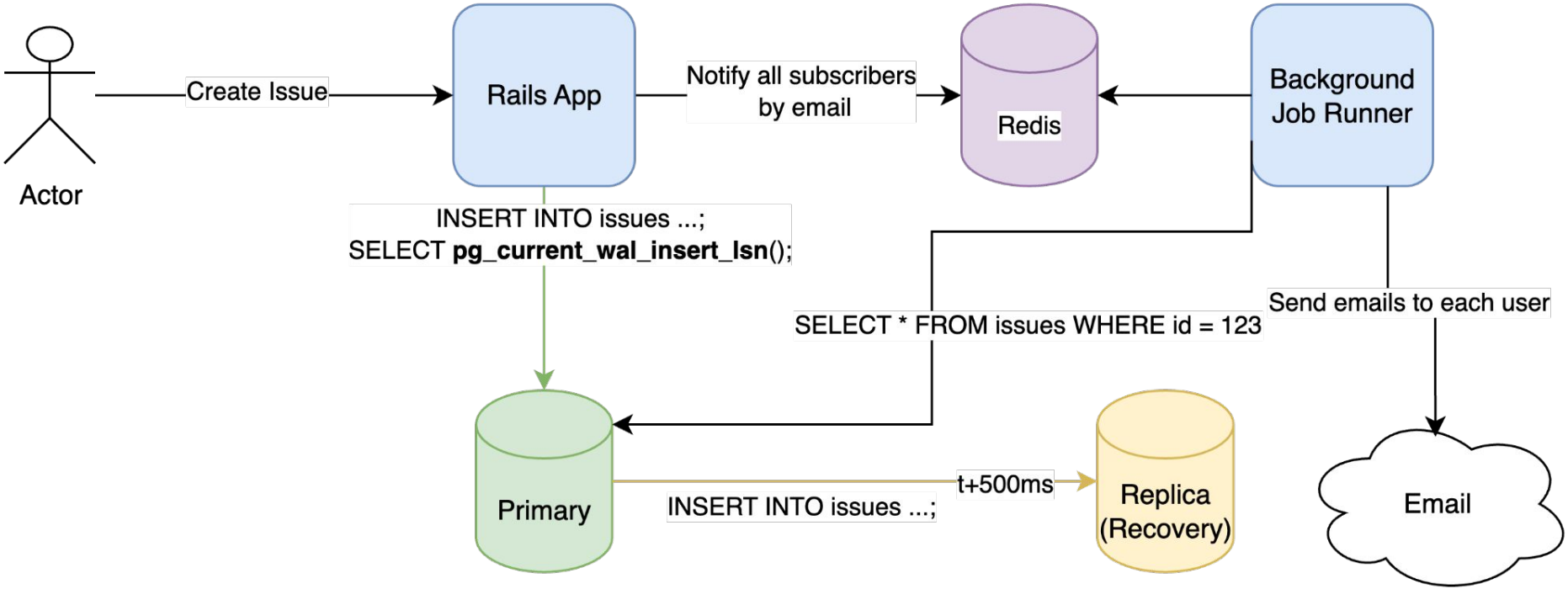
Only expire when all replicas have caught up



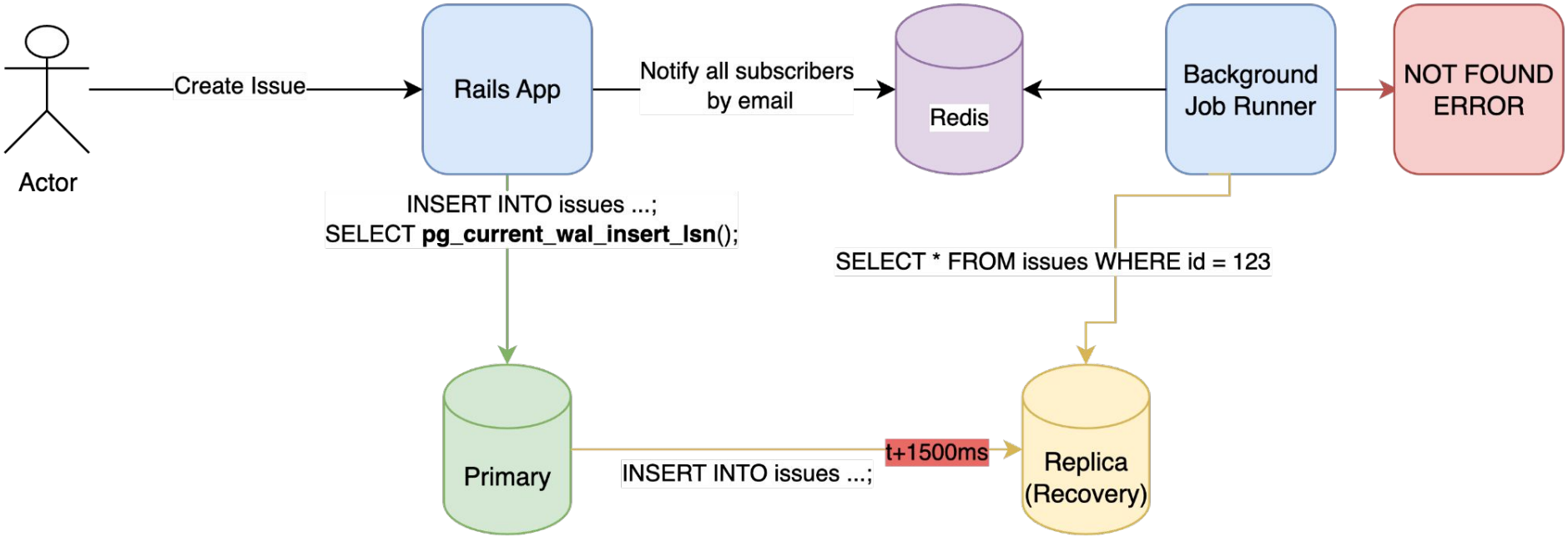
What are background jobs?



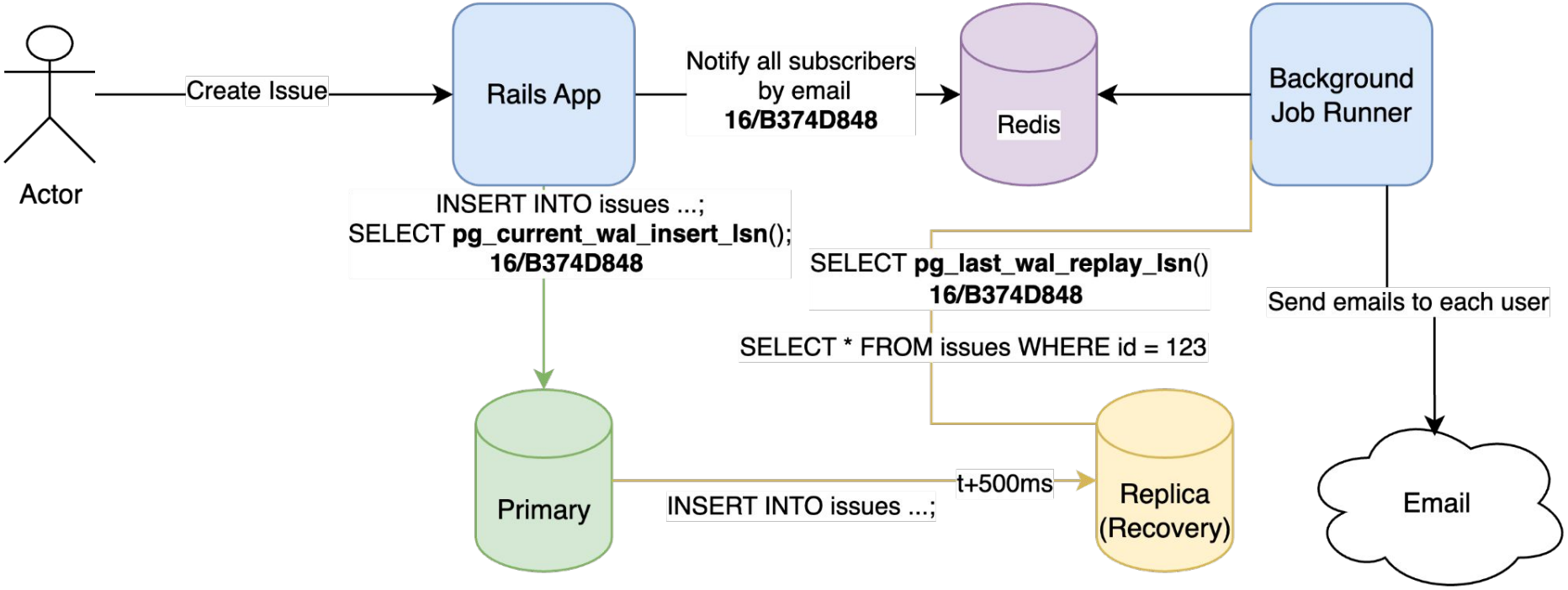
Why not always stick for the user?



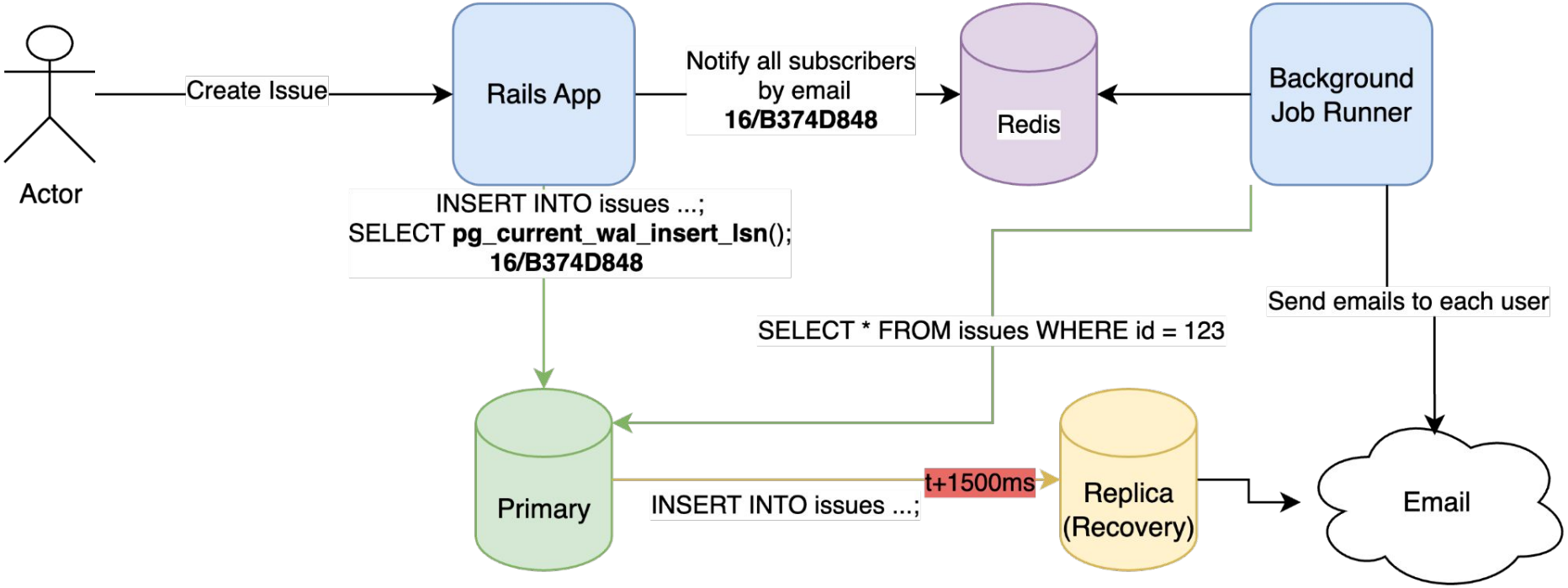
Why not always use replicas as they are run “later”?



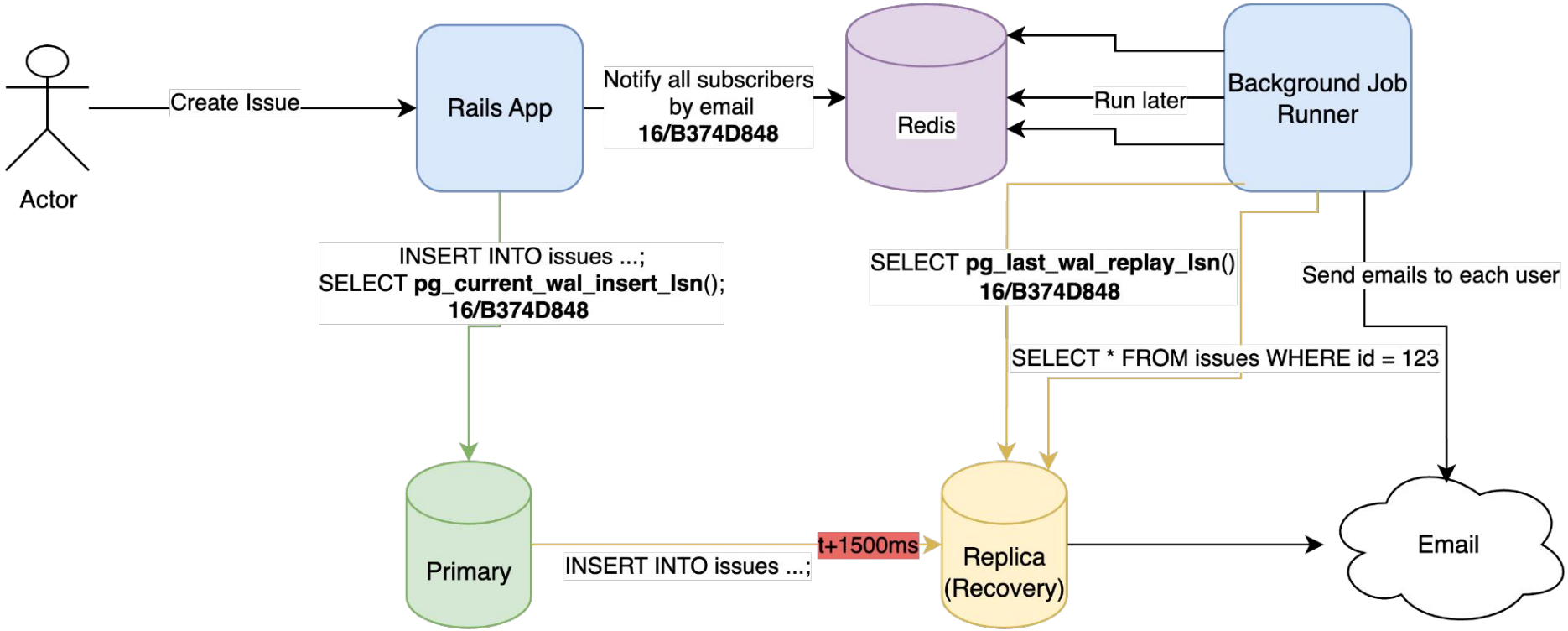
Solution: Store the LSN position with the job in Redis



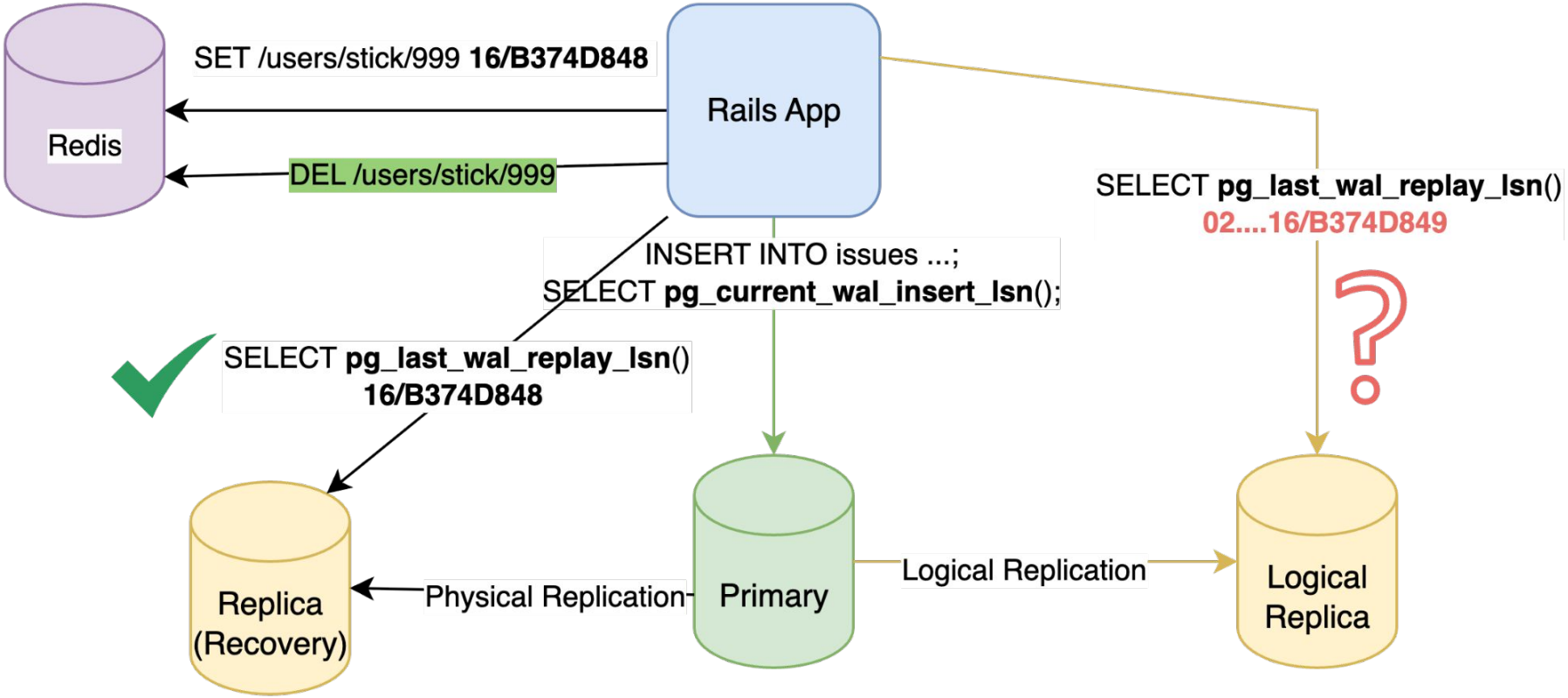
But can we do better?



Solution: Delayed strategy



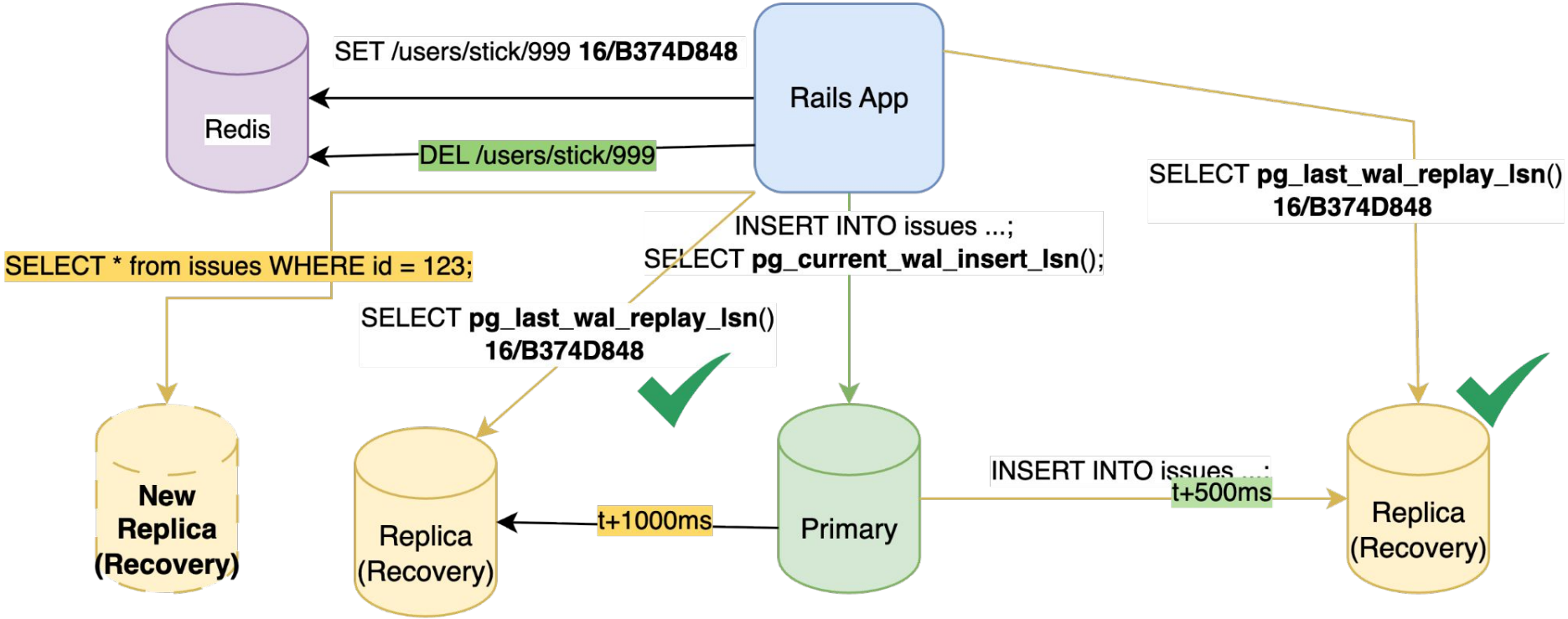
The problem with logical replication



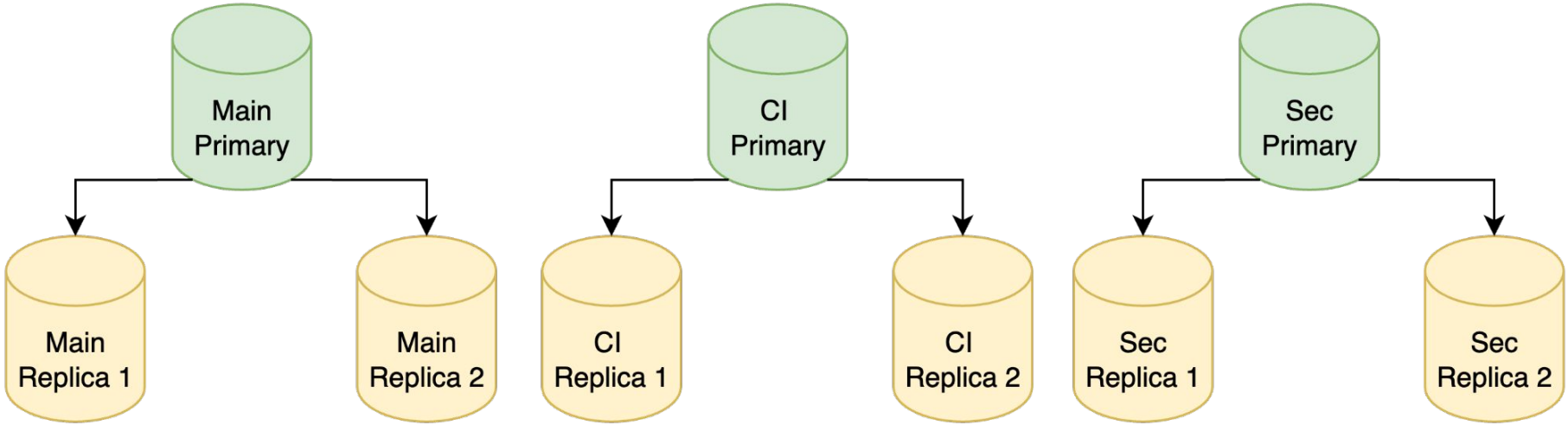
Solution: pg_is_in_recovery and pg_last_wal_replay_lsn

```
CASE
WHEN (SELECT TRUE FROM pg_replication_origin_status) THEN
  (SELECT remote_lsn FROM pg_replication_origin_status)
WHEN pg_is_in_recovery() THEN
  pg_last_wal_replay_lsn()
ELSE
  pg_current_wal_insert_lsn()
END
```

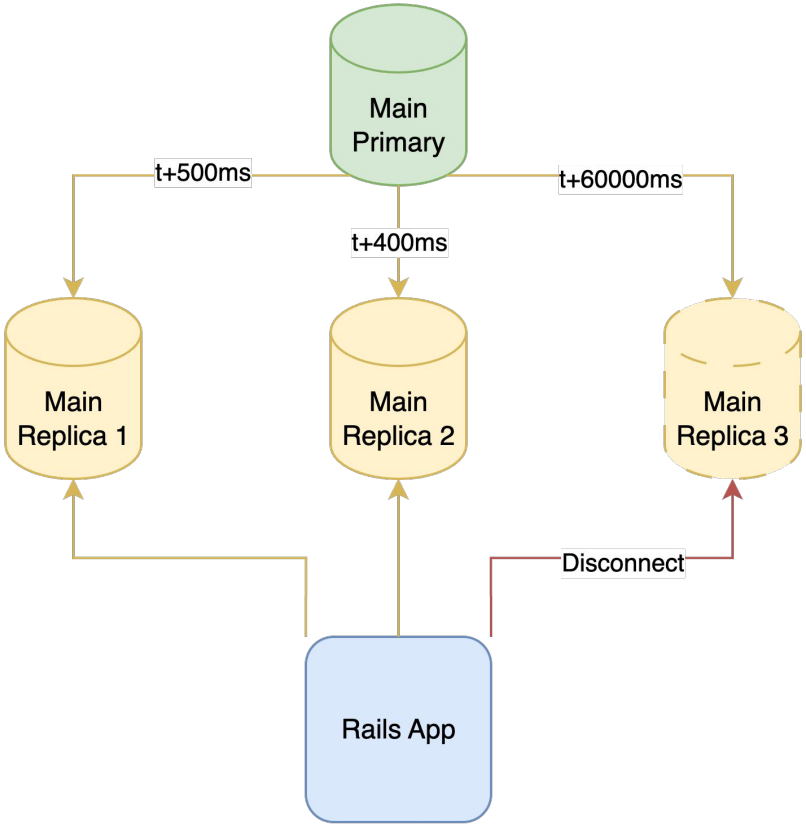
What about removing from Redis and bringing a replica back in that is delayed?



Multiple databases with different LSNs



What about removing unhealthy hosts



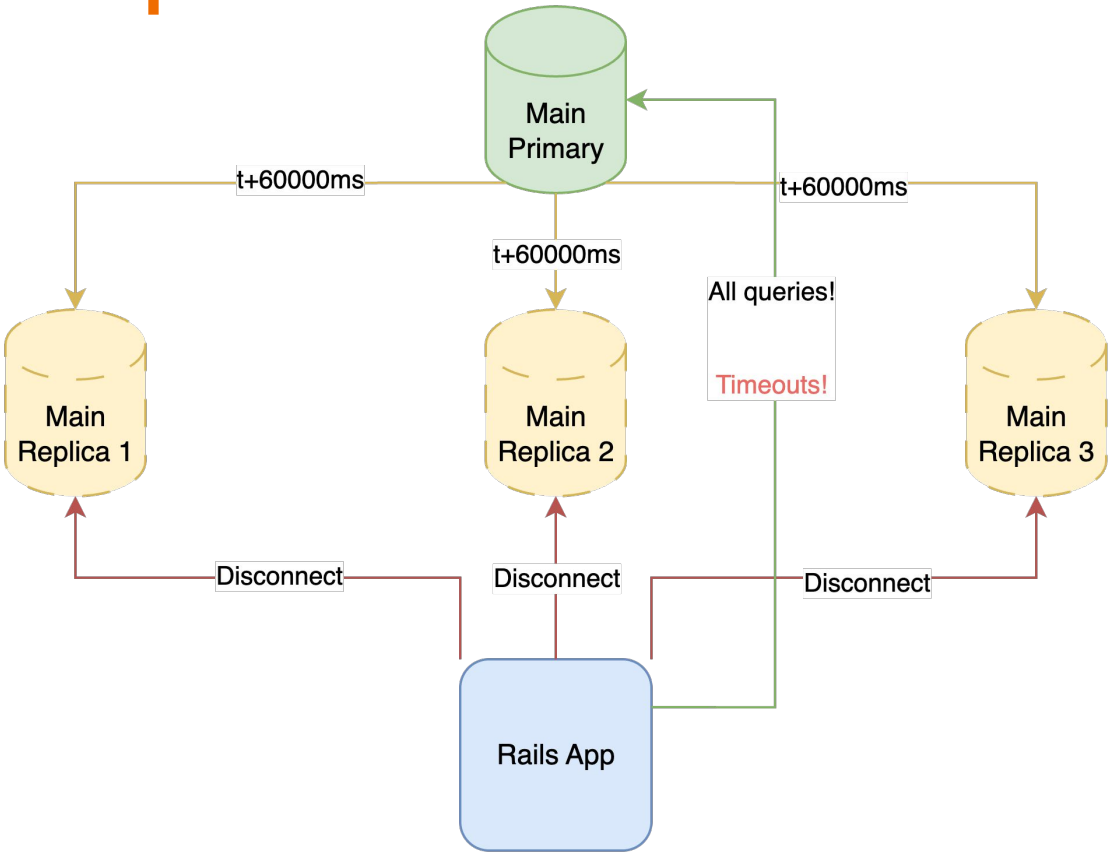
Lag Time:

```
SELECT
EXTRACT(
  EPOCH FROM (now() - pg_last_xact_replay_timestamp())
)::float as lag
```

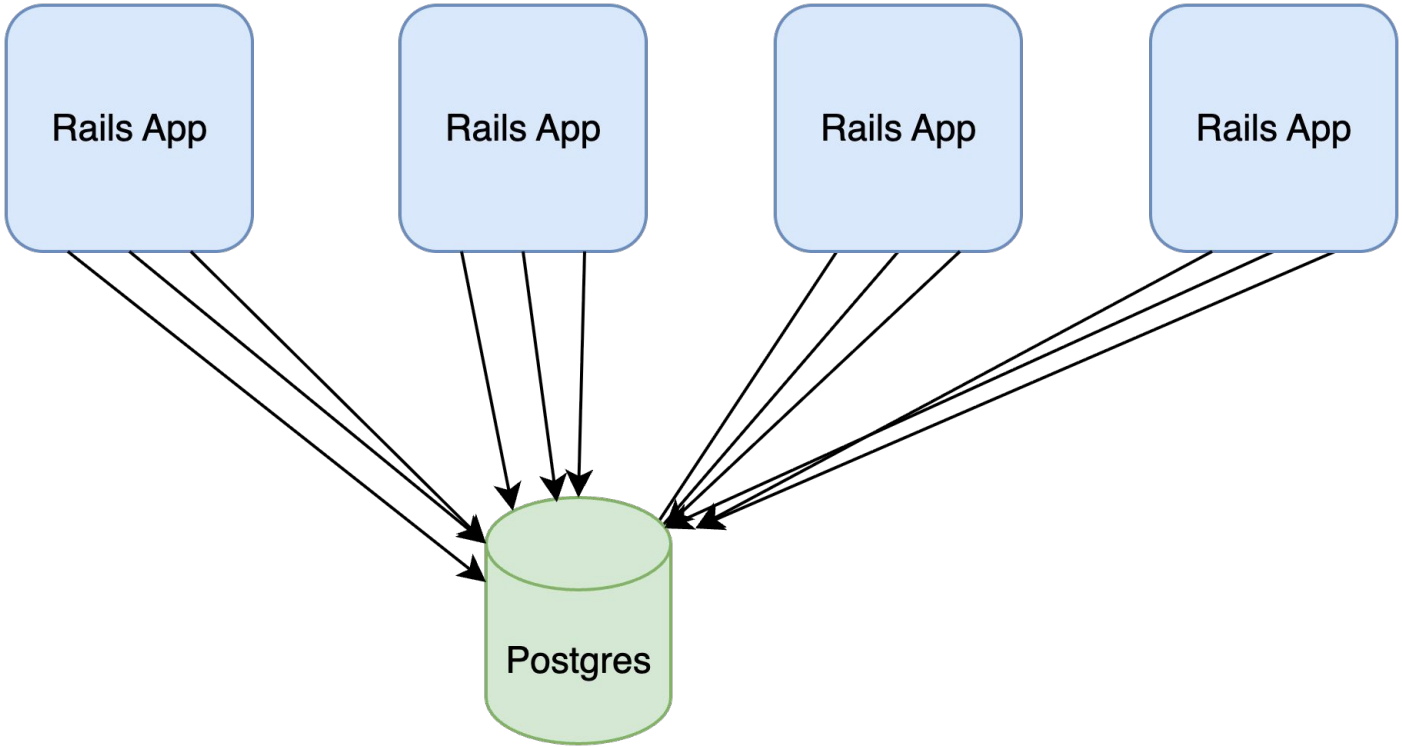
Lag Size (in bytes):

```
SELECT pg_wal_lsn_diff(
  #{location}, (#{latest_lsn_query})
)::float AS diff
```

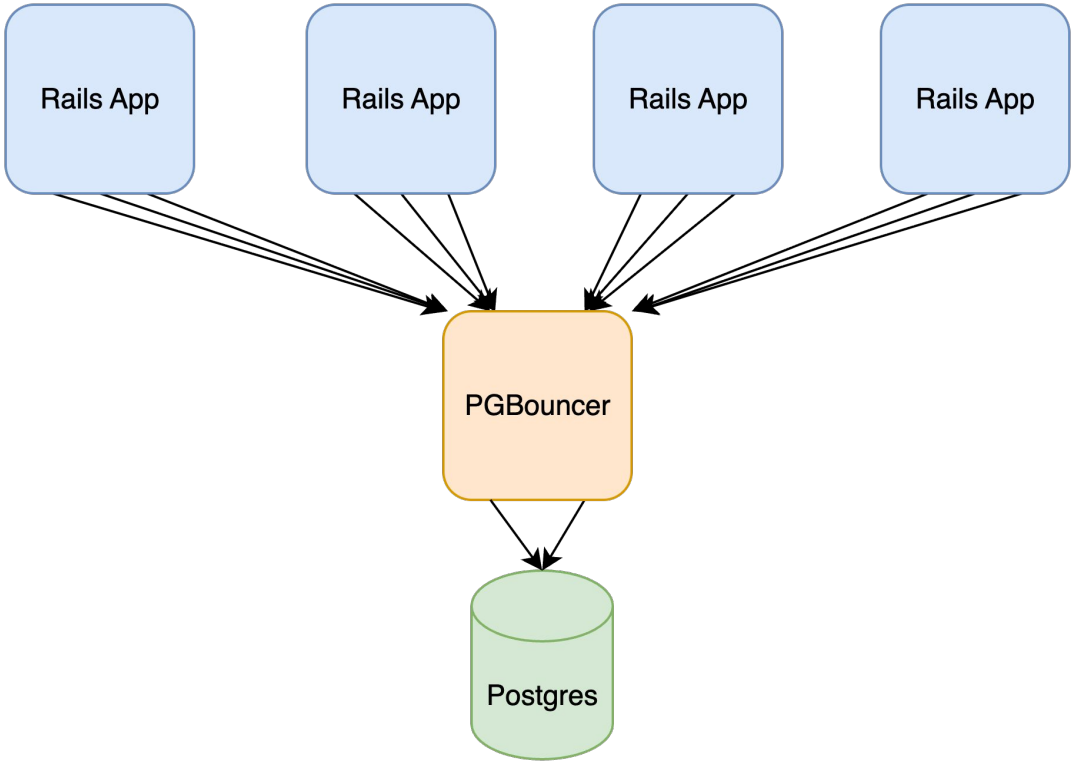
What if all replicas are too far behind?



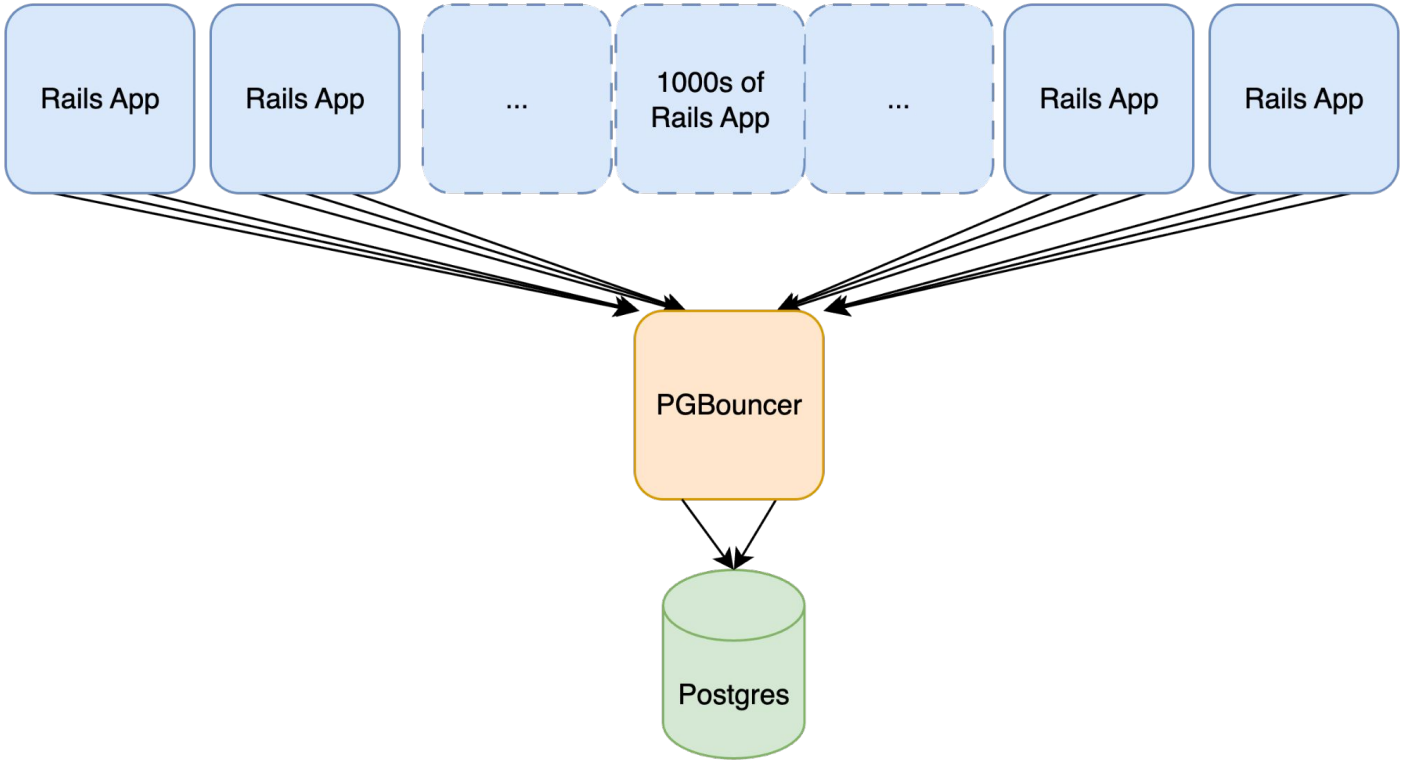
Why do we need a connection pooler?



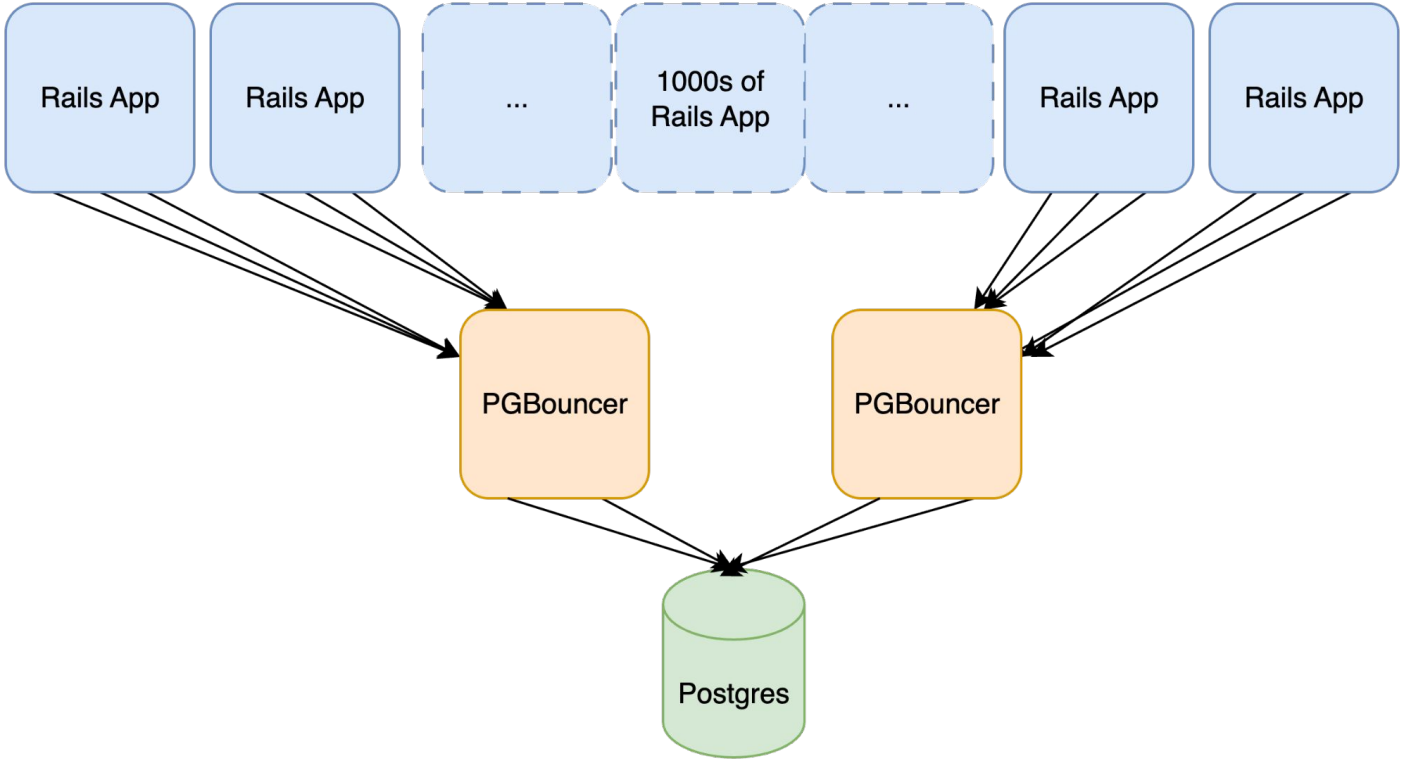
PGBouncer Reduces Connections to Postgres



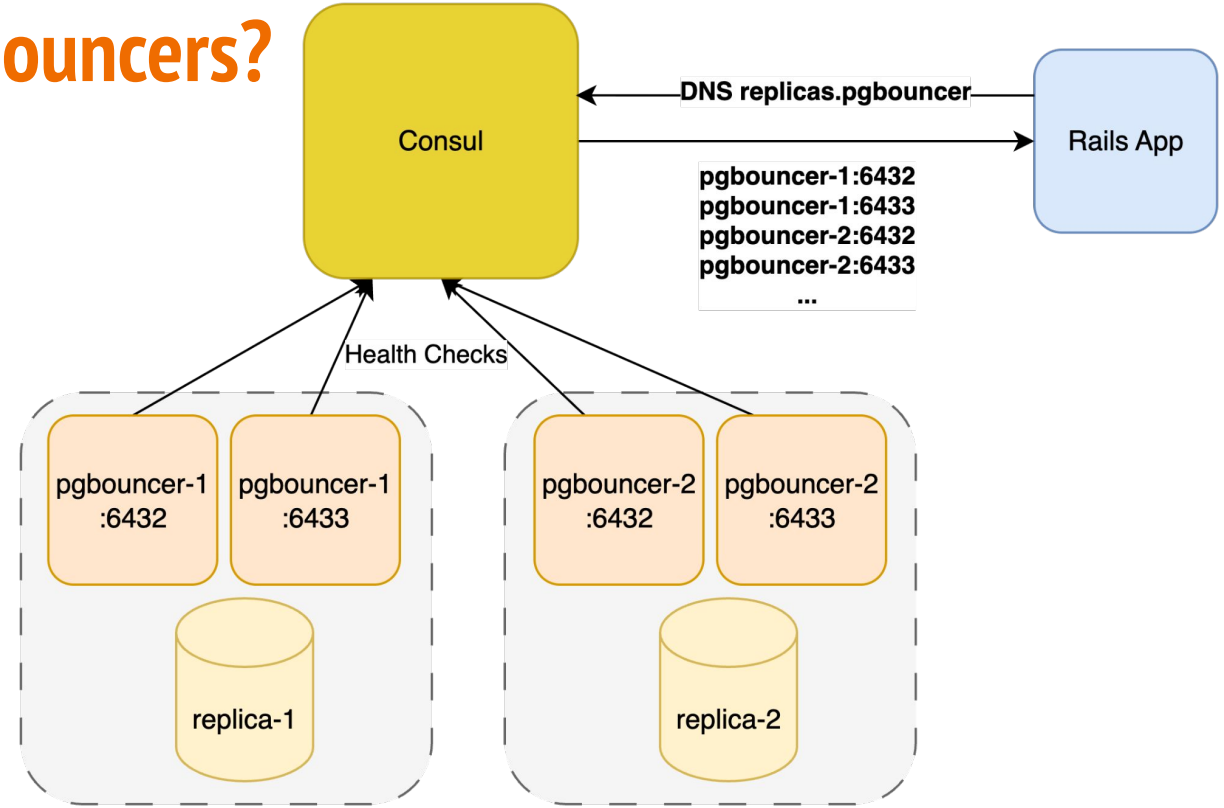
Why do we need multiple connection poolers per PG host?



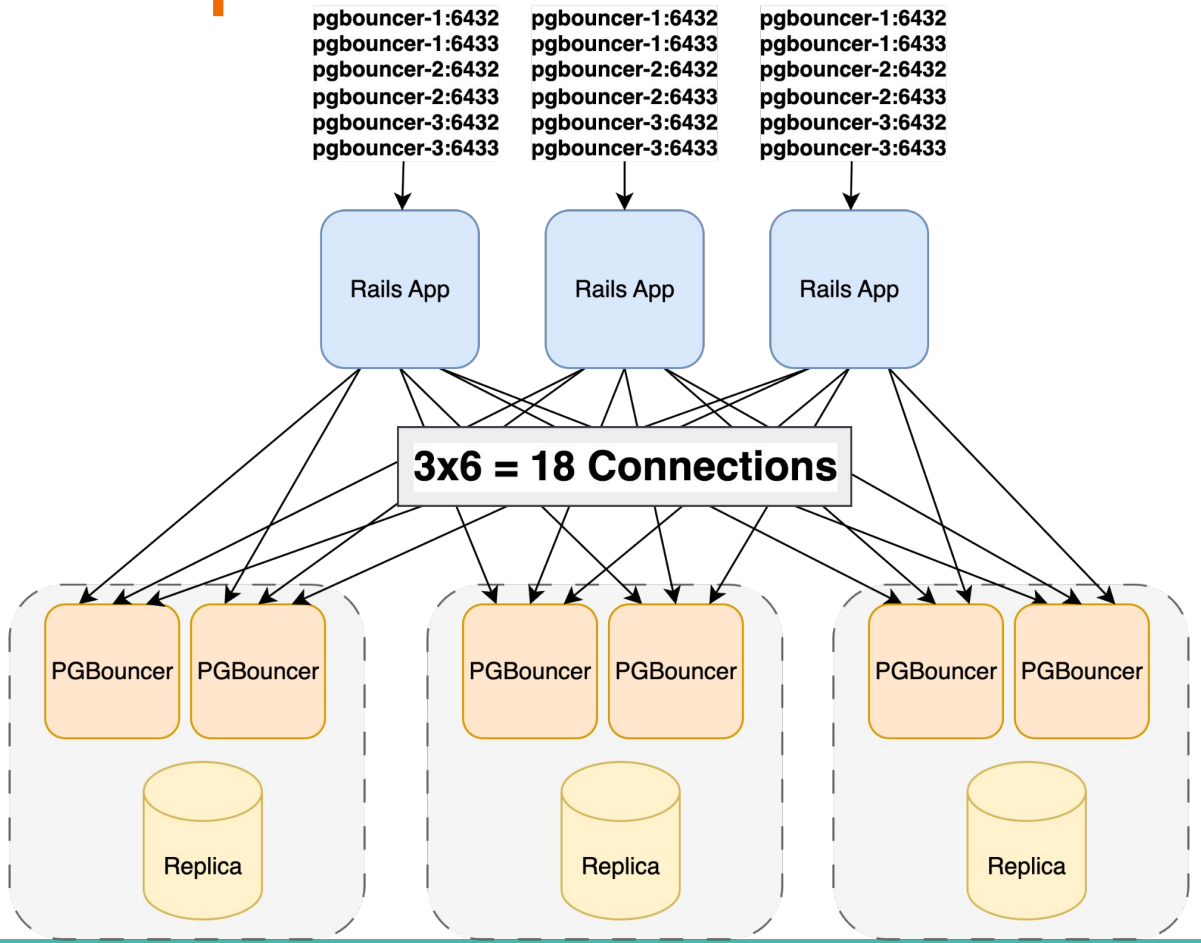
Run multiple PGBouncers per Postgres server



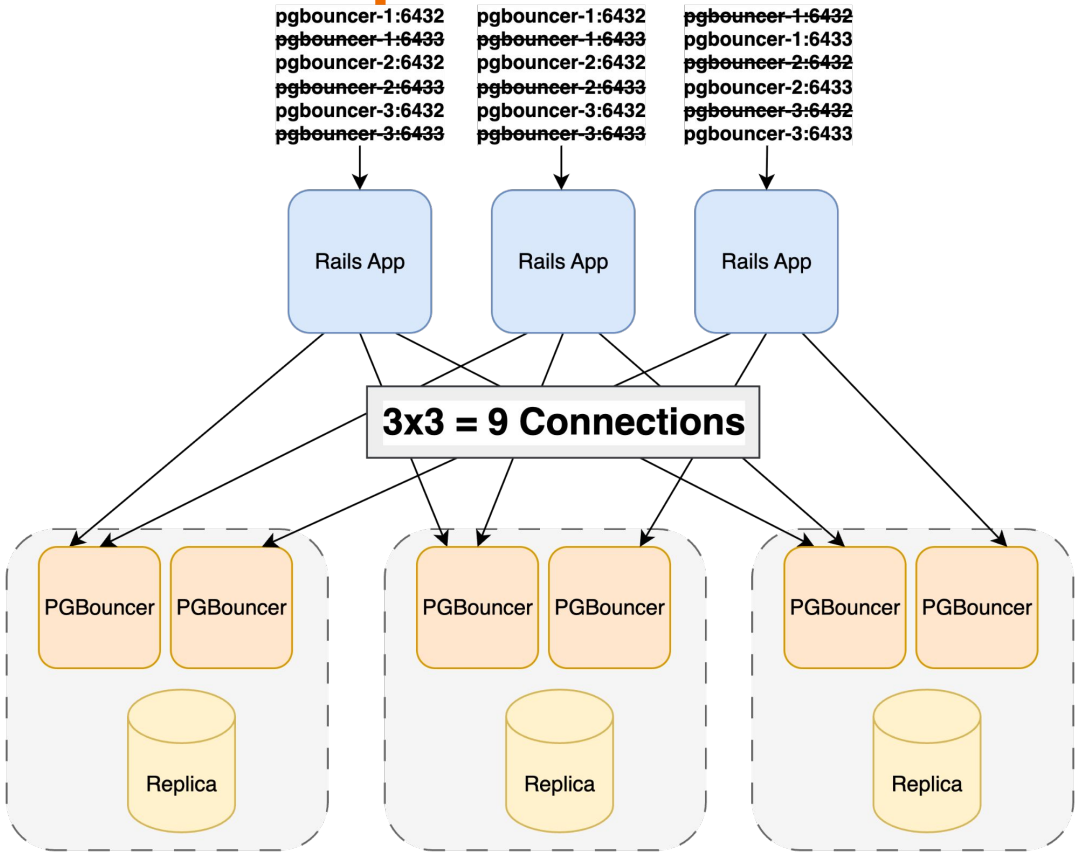
How does GitLab discover and manage all our replicas and their PGBouncers?



N*M connection problem



Solution: 1 connection per hostname



Where to learn more

1. https://gitlab.com/gitlab-org/gitlab/-/tree/master/lib/gitlab/database/load_balancing => MIT Expat license
2. https://gitlab.com/gitlab-org/gitlab/-/tree/master/gems/gitlab-database-load_balancing => Hopefully a gem someday
3. More docs: https://docs.gitlab.com/development/database/

Dylan Griffith, Principal Engineer GitLab
(gitlab.com/DylanGriffith)