

みなさん、こんにちは!

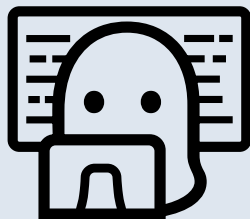
Hi, I'm Rosa



`github.com/rosa`



`rosa.codes`



`programmer at 37signals`

Solid Queue: two years on Rails

Kaigi on Rails, 17 September 2026

v1.0.0

Really battle FunTM-tested stuff

**RAILS
WORLD**
NOV 20-22, 2013
SAN FRANCISCO, CA

v1.0.0

Really battle Fun™-tested stuff

I didn't know how much
Fun™ was yet to come! 😄

RAILS
WORLD
2007 JAN 10, TORONTO CA

Solid Queue 1.0

Solid Queue 1.0

- Active Job backend
- Based on Active Record (relational DB)
- MySQL, PostgreSQL, SQLite
- No need for Redis

Solid Queue 1.0

```
# Jobs
gem "resque"
gem "resque_supervised_fork"
gem "resque-pool"
gem "resque-scheduler"
gem "resque-pause"
gem "sequential_jobs"
gem "scheduled_job"
```



```
# Jobs
gem "solid_queue"
```



Solid Queue 1.0

Solid Queue 1.0



FOR UPDATE SKIP LOCKED

Solid Queue 1.0



FOR UPDATE SKIP LOCKED

Regular SELECT .. FOR UPDATE

jobs

id	queue	priority
1	default	0
2	default	1
3	default	2
4	default	3

Solid Queue 1.0



FOR UPDATE SKIP LOCKED

SELECT .. FOR UPDATE SKIP LOCKED

jobs

id	queue	priority
1	default	0
2	default	1
3	default	2
4	default	3

Solid Queue 1.0

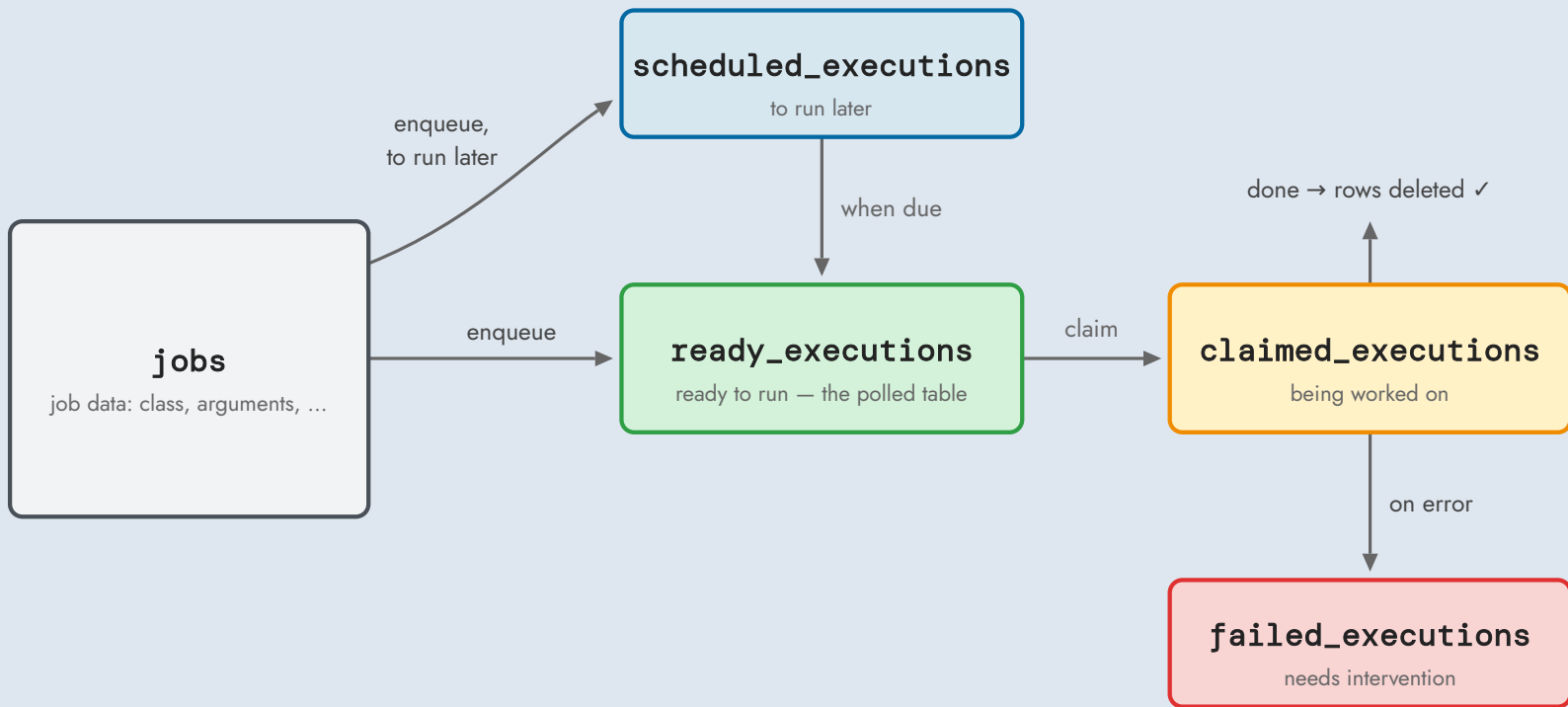


Small polling table

Solid Queue 1.0



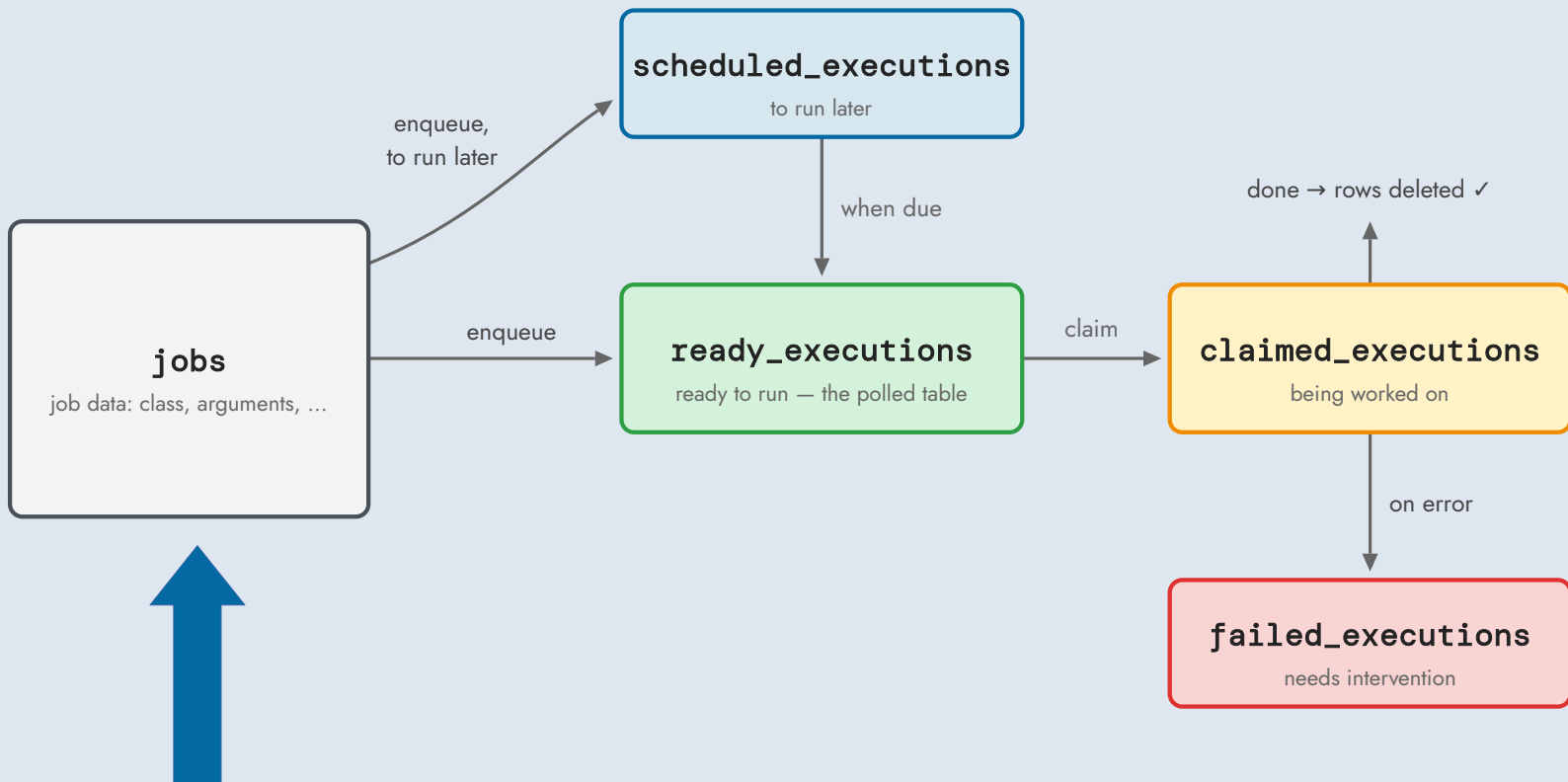
Small polling table



Solid Queue 1.0



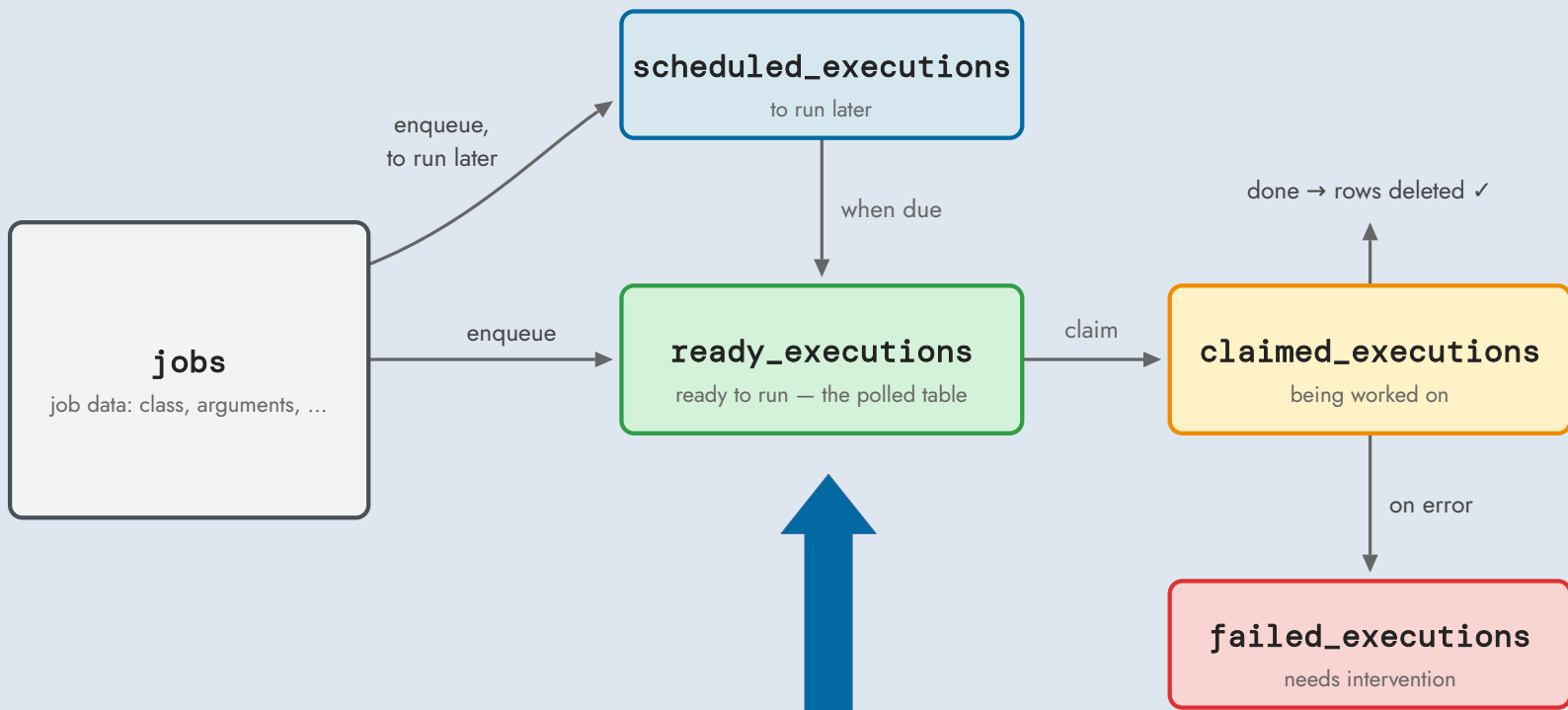
Small polling table



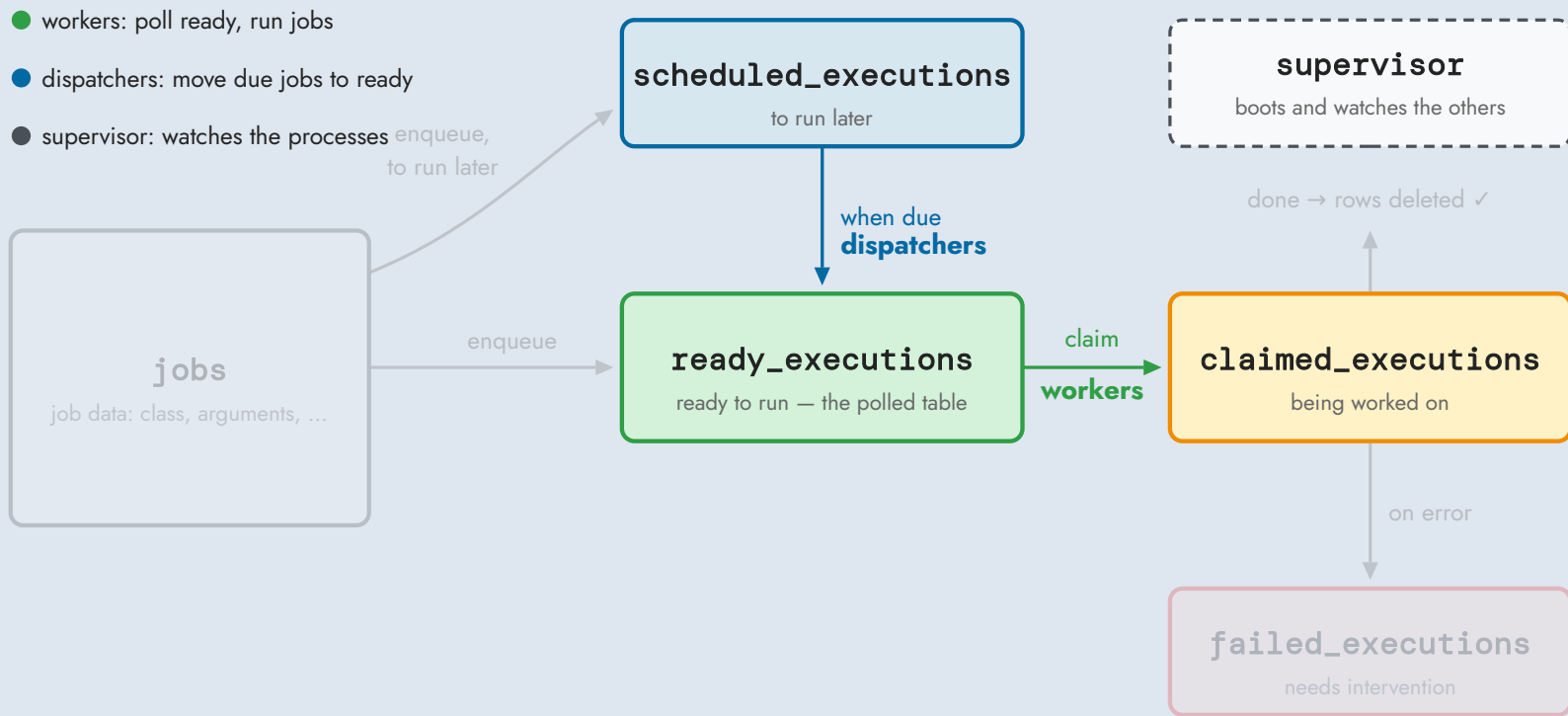
Solid Queue 1.0



Small polling table



Solid Queue 1.0



Solid Queue 1.0

- Enqueue, poll and run jobs ✓
- Keep failed jobs for manual retry ✓
- Allow scheduling jobs in the future ✓

Solid Queue 1.0

- Enqueue, poll and run jobs ✓
- Keep failed jobs for manual retry ✓
- Allow scheduling jobs in the future ✓
- Sequential jobs?

Solid Queue 1.0

- Enqueue, poll and run jobs ✓
- Keep failed jobs for manual retry ✓
- Allow scheduling jobs in the future ✓
- ~~Sequential jobs?~~ Concurrency controls

Solid Queue 1.0

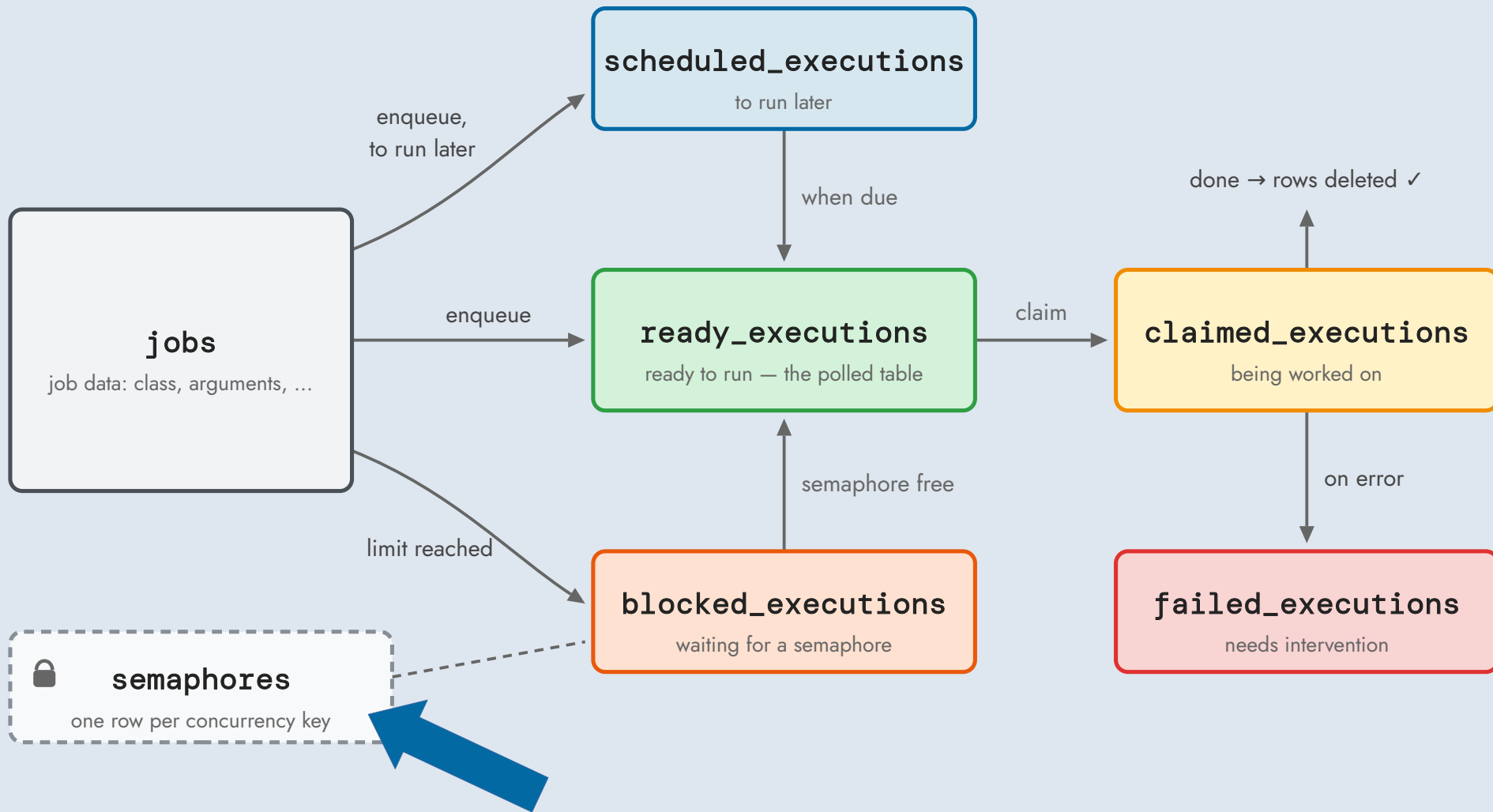
Concurrency controls

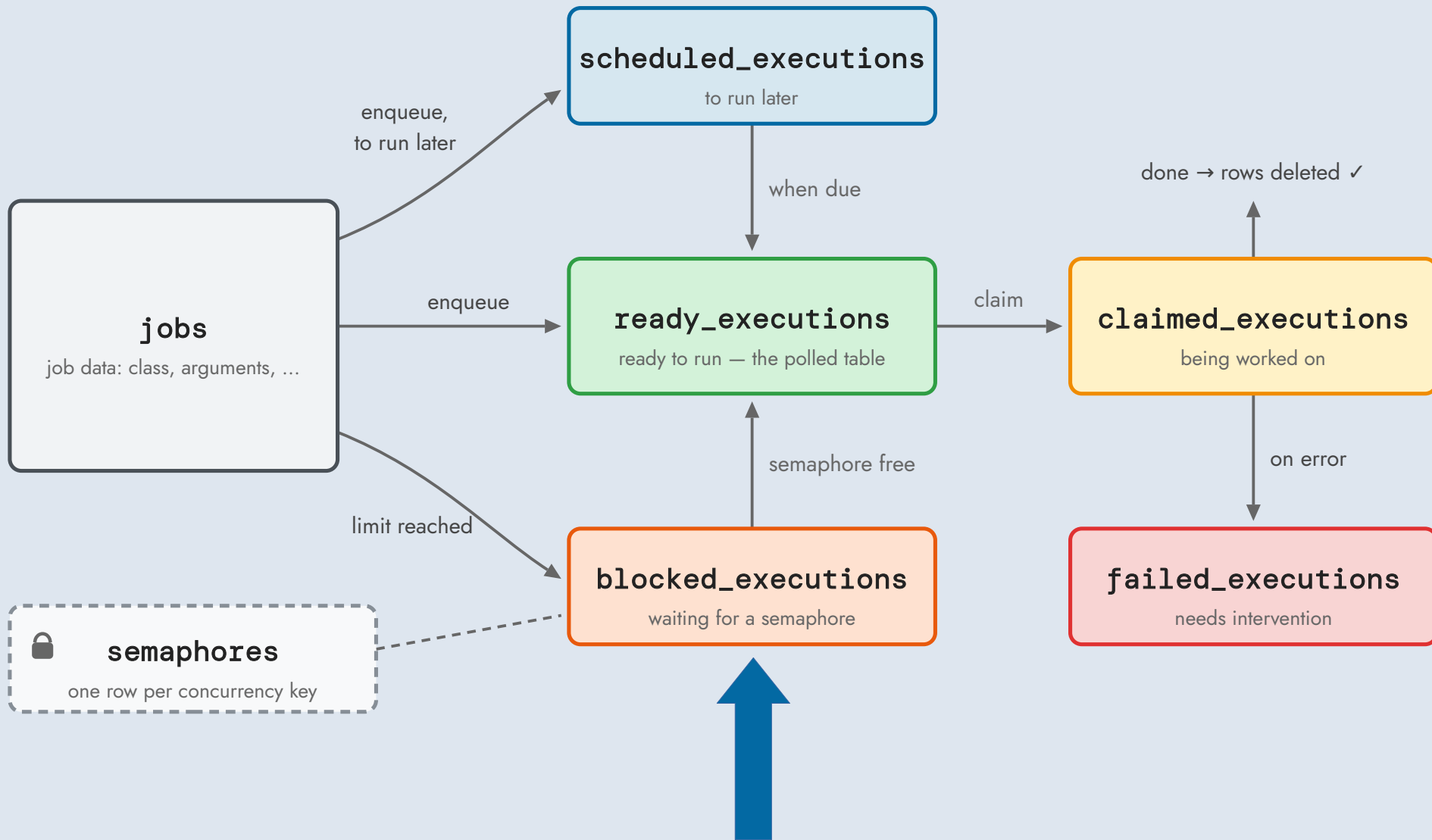
```
class Recording::StatusPropagationJob
  limits_concurrency limit: 1,
                    key: ->(recording, _, _) { recording.bucket_id },
                    duration: 1.day

  # ...
```

Solid Queue 1.0

Concurrency controls





Solid Queue 1.0

Concurrency controls



Solid Queue 1.0

- Enqueue, poll and run jobs ✓
- Keep failed jobs for manual retry ✓
- Allow scheduling jobs in the future ✓
- Concurrency controls ✓

Solid Queue 1.0

- Enqueue, poll and run jobs ✓
- Keep failed jobs for manual retry ✓
- Allow scheduling jobs in the future ✓
- Concurrency controls ✓
- Recurring jobs

Solid Queue 1.0

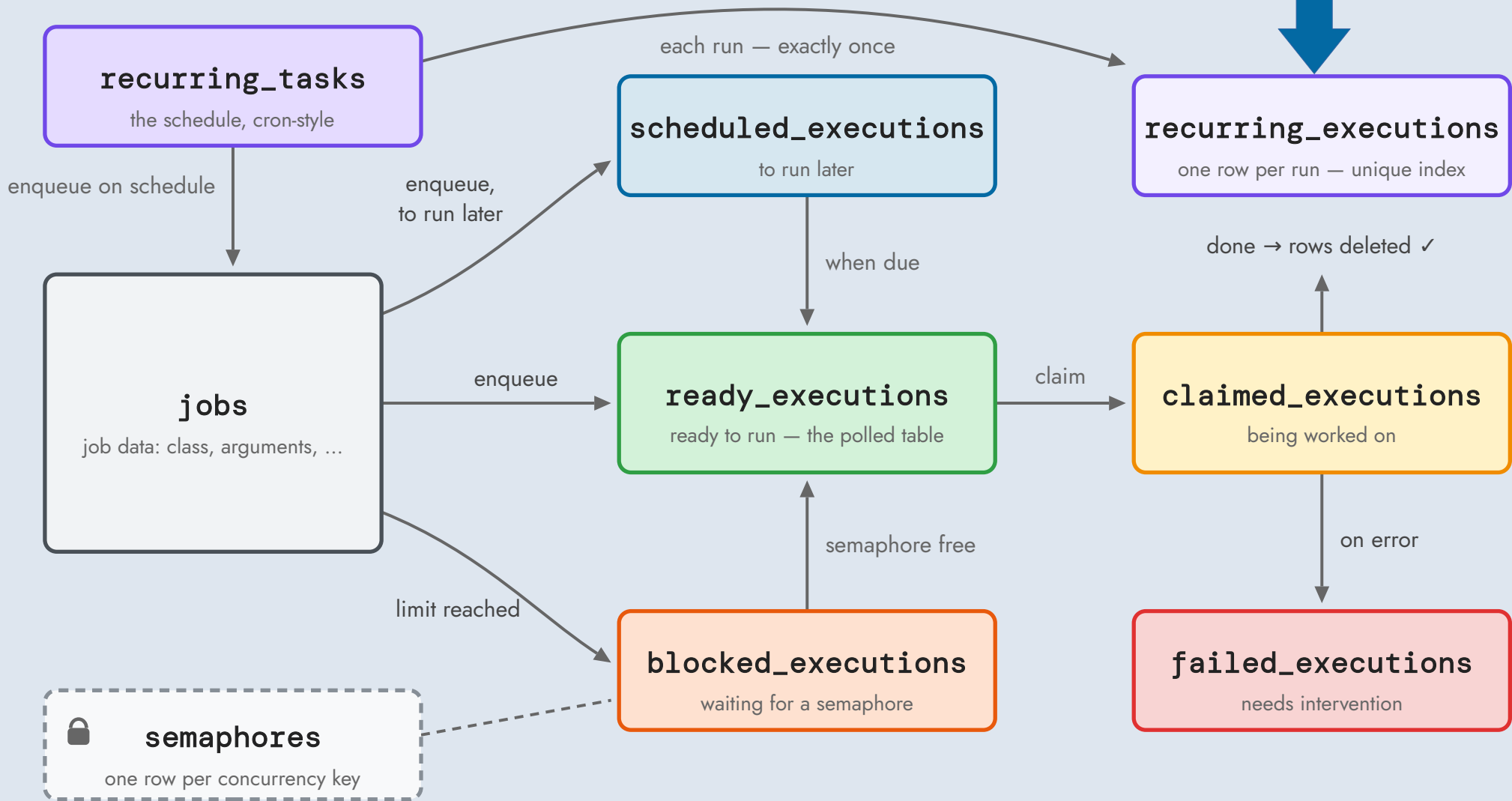
Recurring jobs

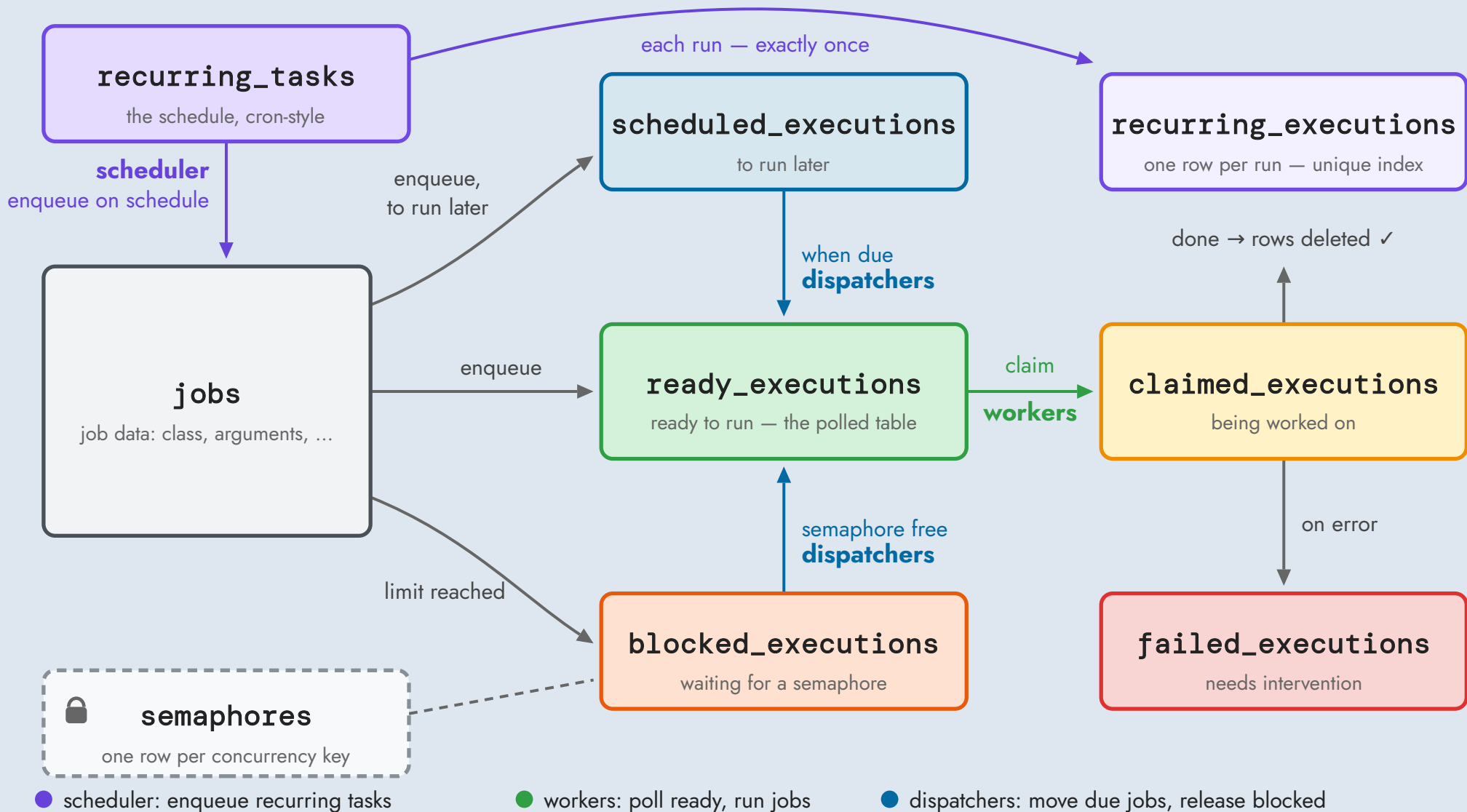
```
# config/recurring.yml
production:
  deliver_bundled_notifications:
    command: "Notification::Bundle.deliver_all_later"
    schedule: every 30 minutes

  delete_unused_tags:
    class: DeleteUnusedTagsJob
    schedule: every day at 04:02

  clear_solid_queue_finished_jobs:
    command: "SolidQueue::Job.clear_finished_in_batches"
    schedule: every hour at minute 12

# ...
```



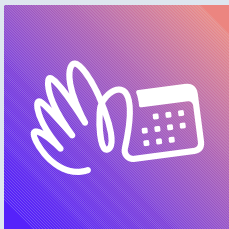


Solid Queue 1.0

- Enqueue, poll and run jobs ✓
- Keep failed jobs for manual retry ✓
- Allow scheduling jobs in the future ✓
- Concurrency controls ✓
- Recurring jobs ✓

Solid Queue 1.0

built for one app...



- ~20M jobs / day
- ~800 workers, 74 supervisors
- Big, dedicated MySQL instance

... but adopted by others

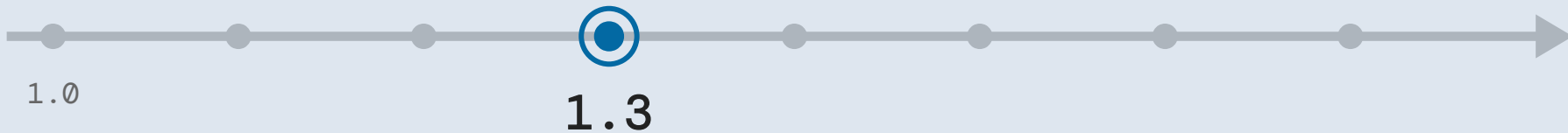
- PostgreSQL at a large scale
- Single server, SQLite
- LLM-heavy jobs

From 1.0 to 2.0

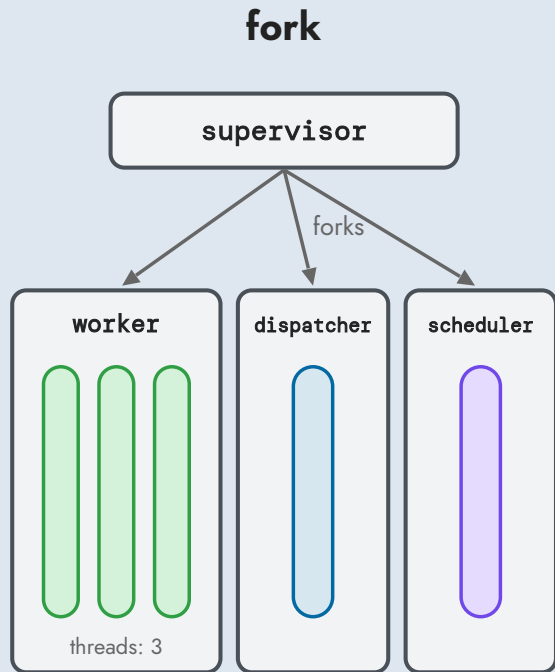
- 21 releases
- 163 PRs merged
- 119 contributors, ~70 new since 1.0
- ~26M downloads



Async/single-process mode



Async/single-process mode

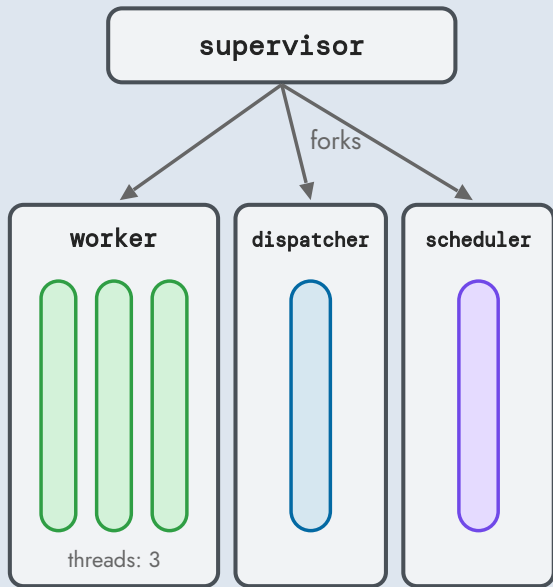


a separate process for each one —
good isolation, more memory

Async/single-process mode

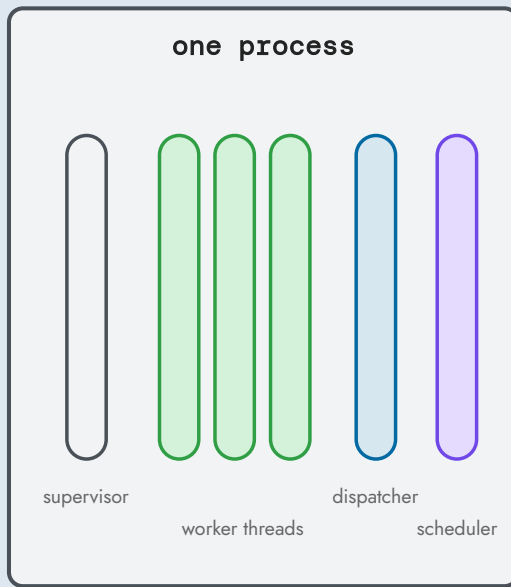


fork



a separate process for each one —
good isolation, more memory

async

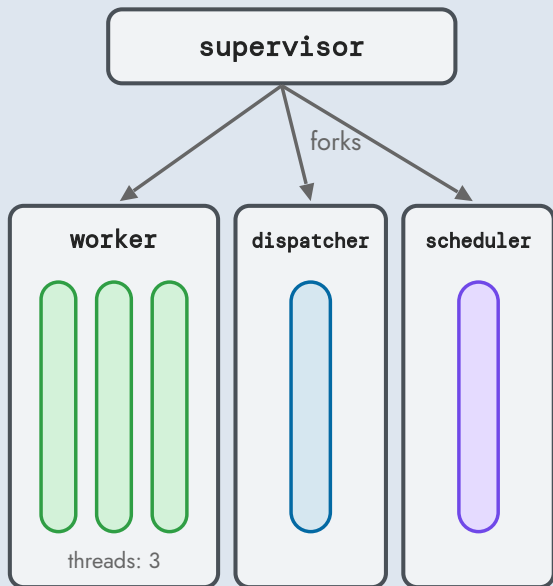


everything runs on threads,
in a single process

Async/single-process mode

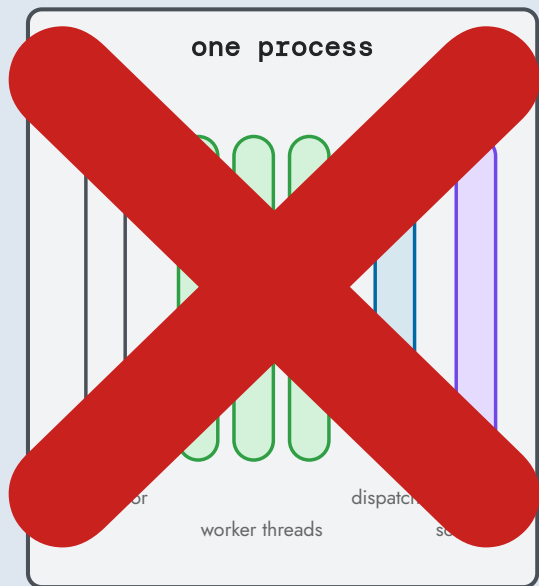


fork



a separate process for each one —
good isolation, more memory

async



everything runs on threads,
in a single process

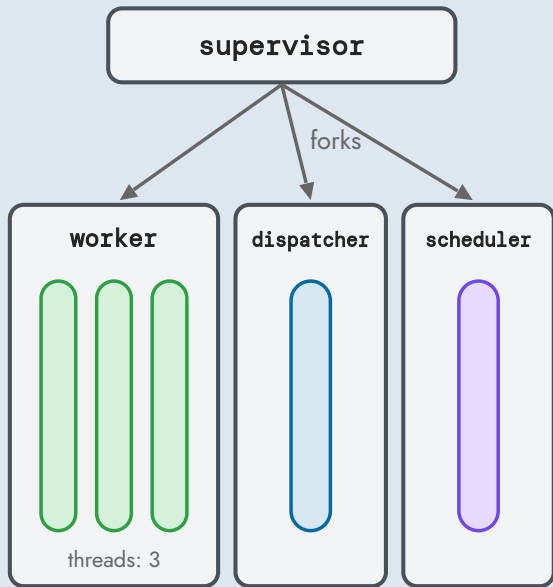


@bensheldon

Async/single-process mode

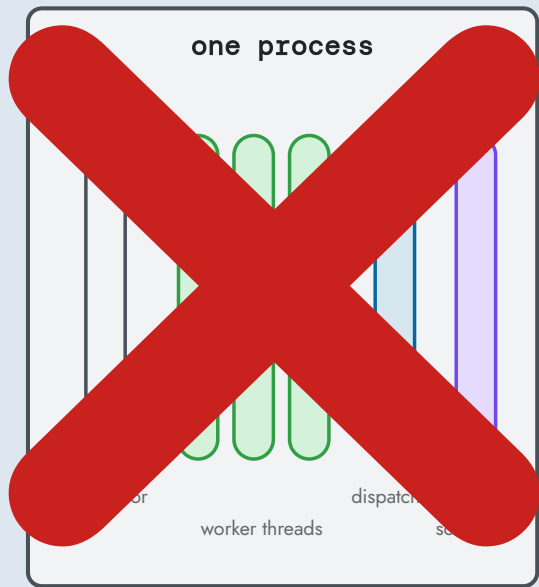


fork



a separate process for each one —
good isolation, more memory

async



everything runs on threads,
in a single process



@bensheldon



Async/single-process mode 🖥️

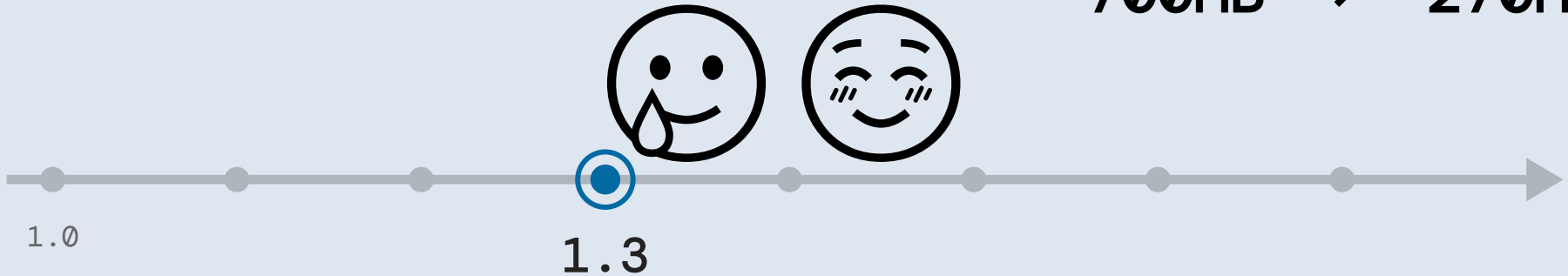
```
# single-process mode
```

```
$ bin/jobs --mode async
```

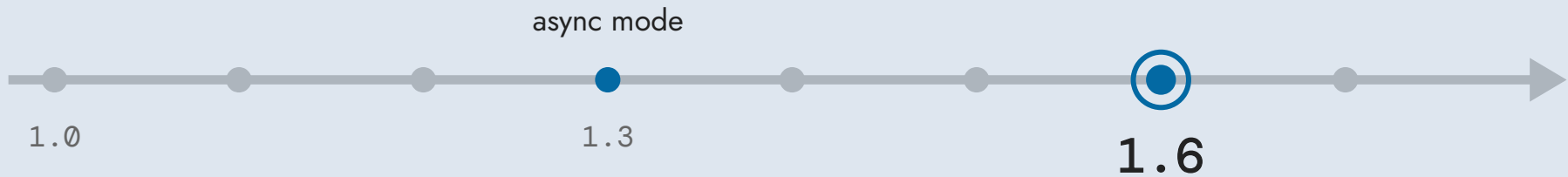


@joshleblanc

~700MB → ~270MB

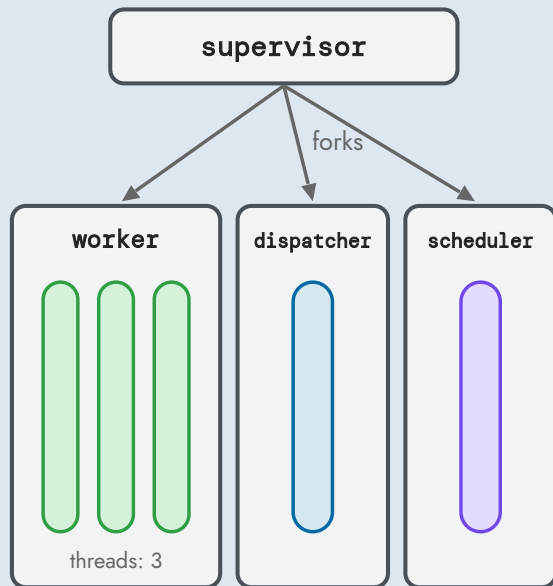


Fibers



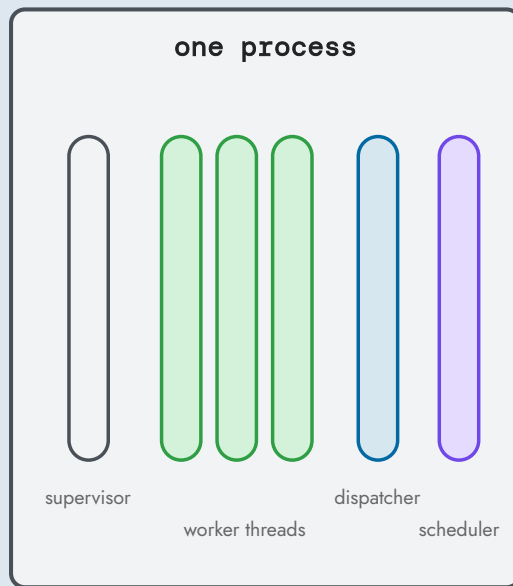
Fibers

fork



a separate process for each one —
good isolation, more memory

async



everything runs on threads,
in a single process



**THE HEART
OF CODE**

EURUKO 2025 | PORTUGAL
September 18-19, Viana do Castelo

@crmne



Fibers

```
# config/queue.yml
workers:
  - queues: "api*"
    threads: 3
    polling_interval: 0.05
```



@crmne

Fibers

```
# config/queue.yml
workers:
  - queues: "api*"
    fibers: 100
    polling_interval: 0.05
```

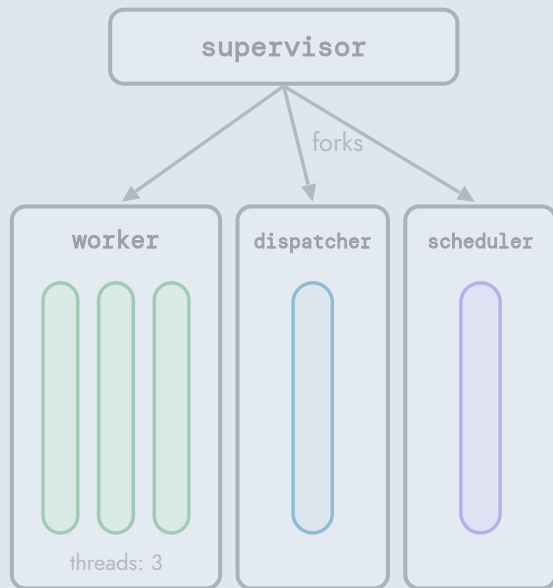


@crmne

- Built on the **Async** gem
- Fork or *async* mode

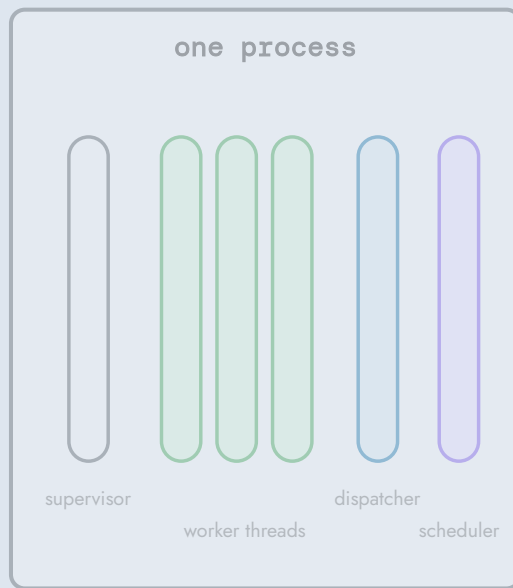
Fibers

fork



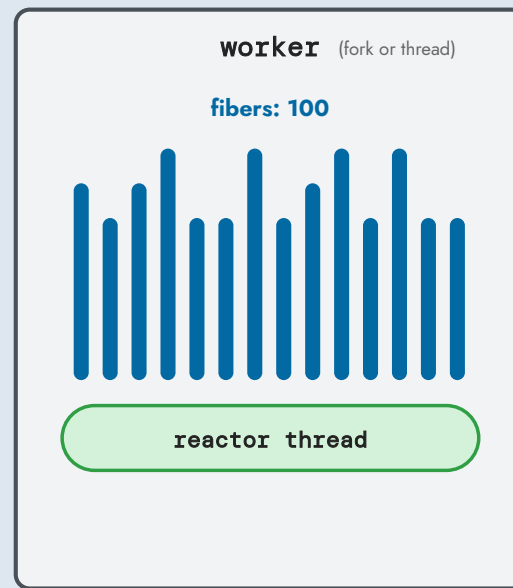
a separate process for each one —
good isolation, more memory

async



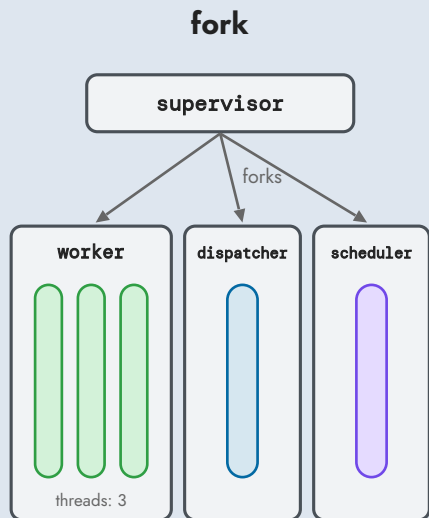
everything runs on threads,
in a single process

fibers

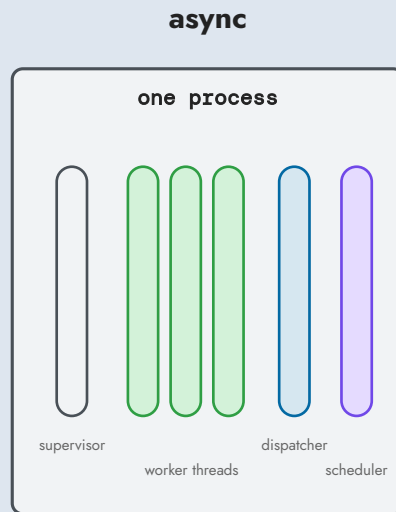


one thread runs many fibers (Async) —
cheap waiting on I/O

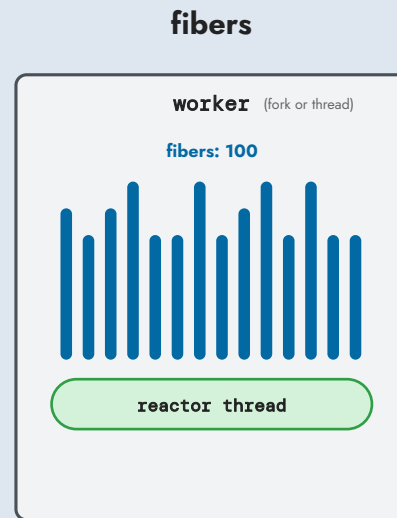
Fibers



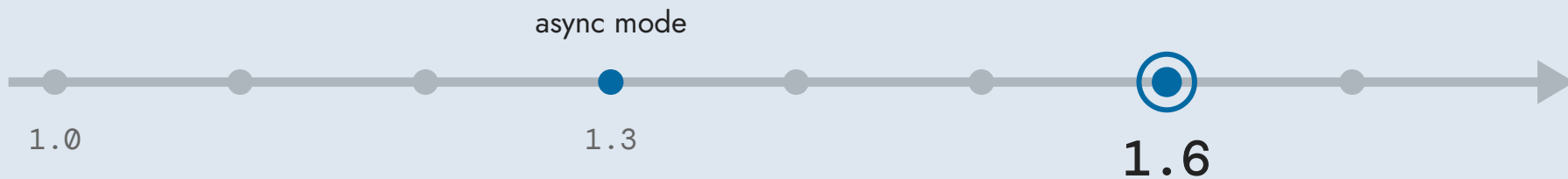
a separate process for each one —
good isolation, more memory



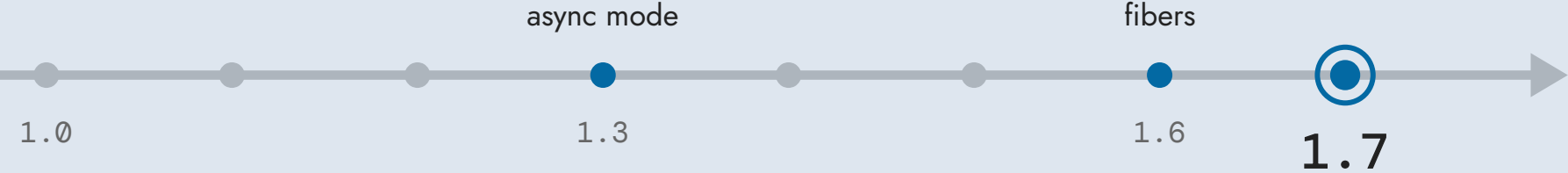
everything runs on threads,
in a single process



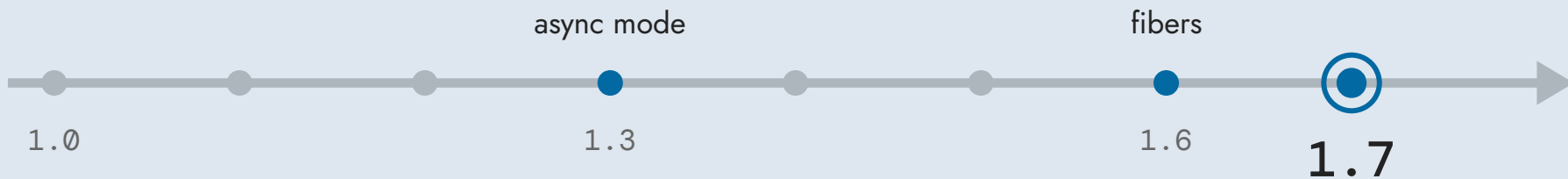
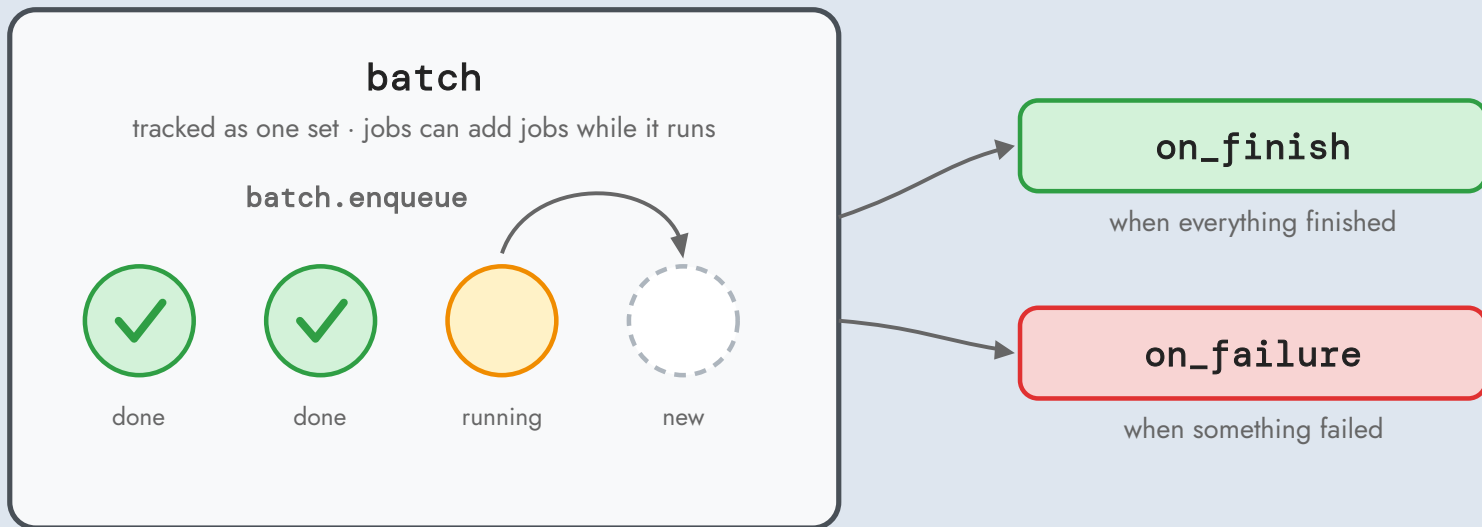
one thread runs many fibers (Async) —
cheap waiting on I/O



Batches



Batches



Batches



@jpcamara

PR #142



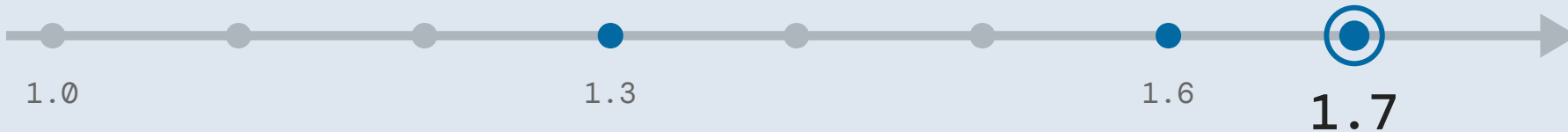
opened 2 Feb 2024 — @jpcamara

2025

2026

async mode

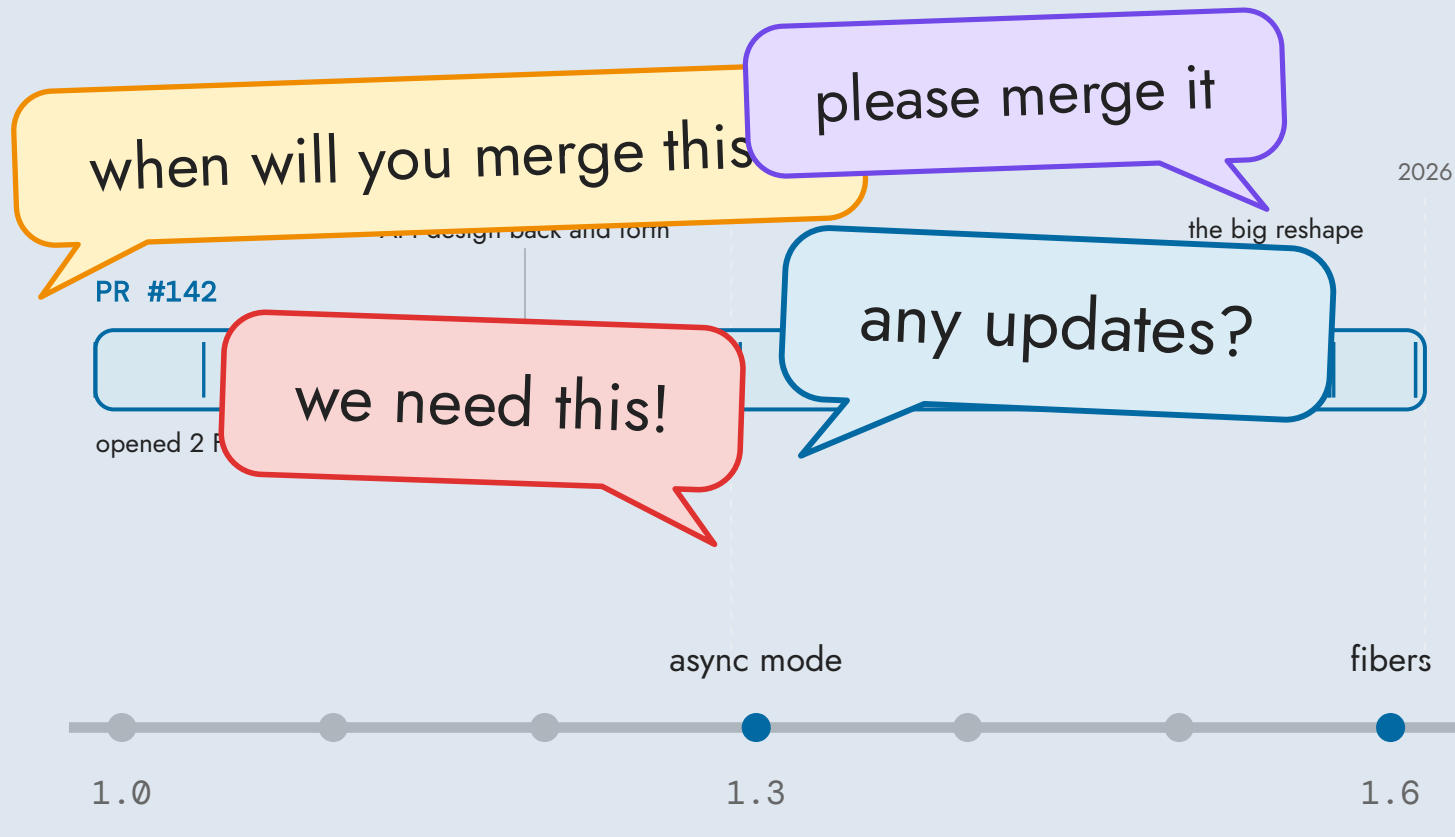
fibers



Batches



@jpcamara



Batches



@jpcamara

PR #142



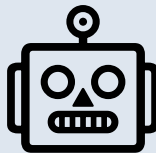
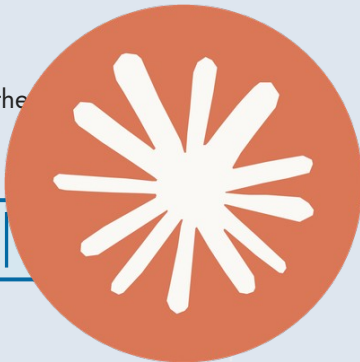
opened 2 Feb 2024 — @jpcamara

API design back and forth

2025

"is this ready yet?"

the



async mode

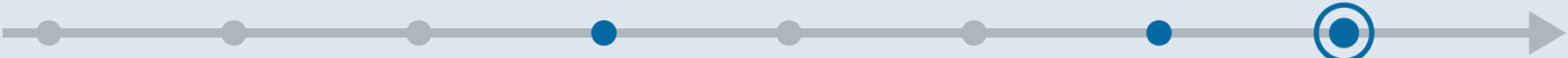
fibers

1.0

1.3

1.6

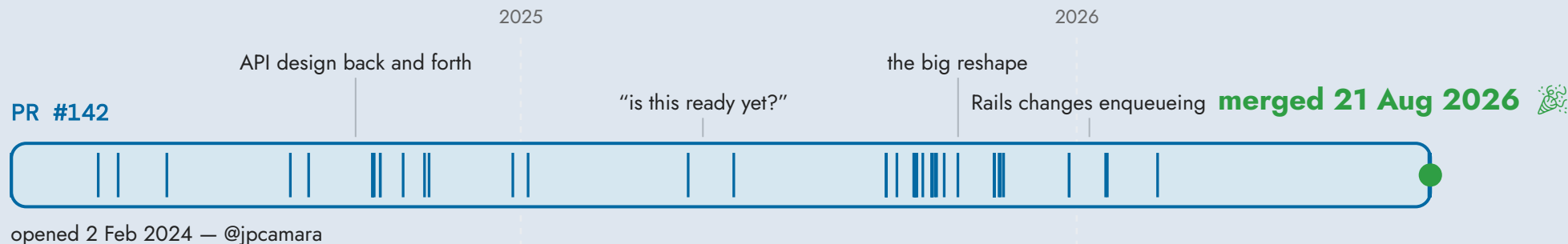
1.7



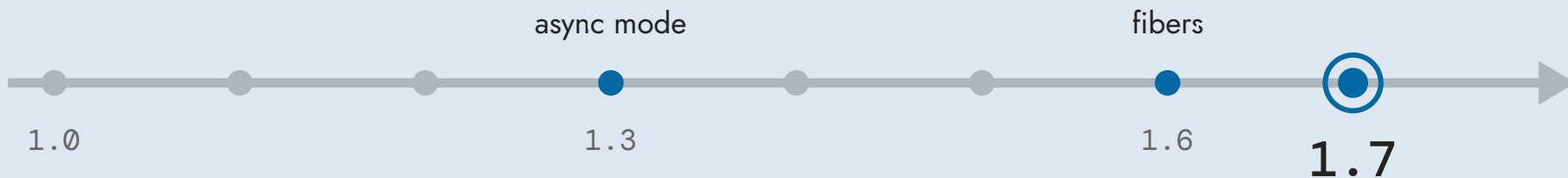
Batches



@jpcamara



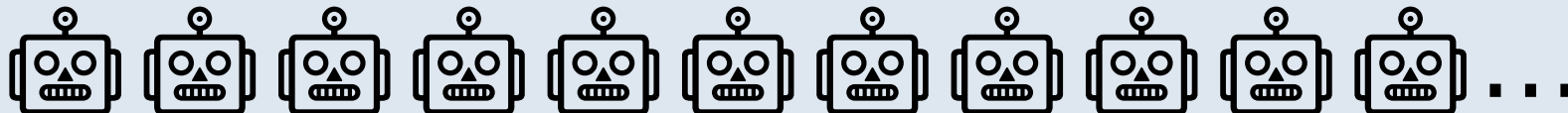
2 years, 6 months and 19 days later 😊



Oh, the new reality...



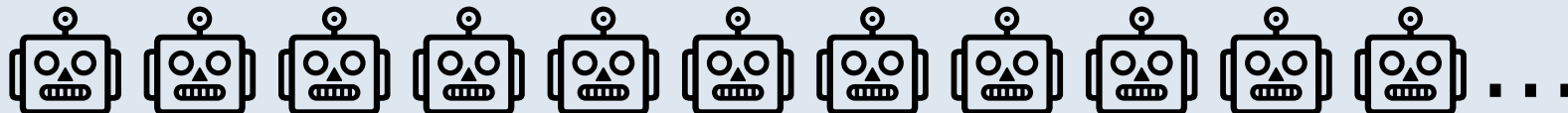
- Taste still matters



Oh, the new reality...



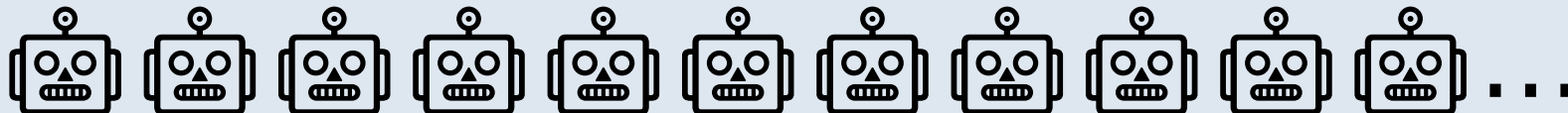
- Taste still matters
- Shipping and maintaining are still the definitive test



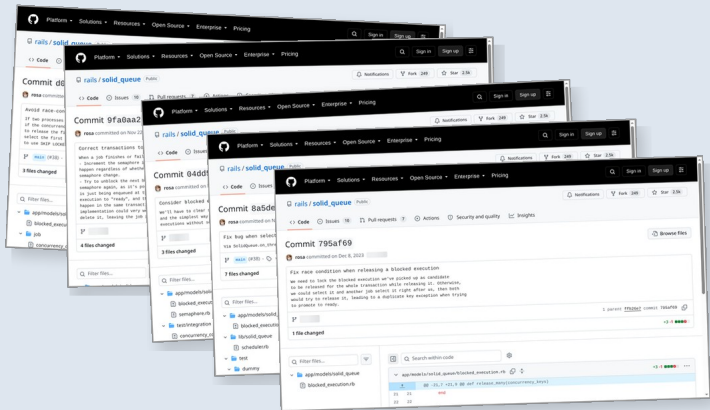
Oh, the new reality...



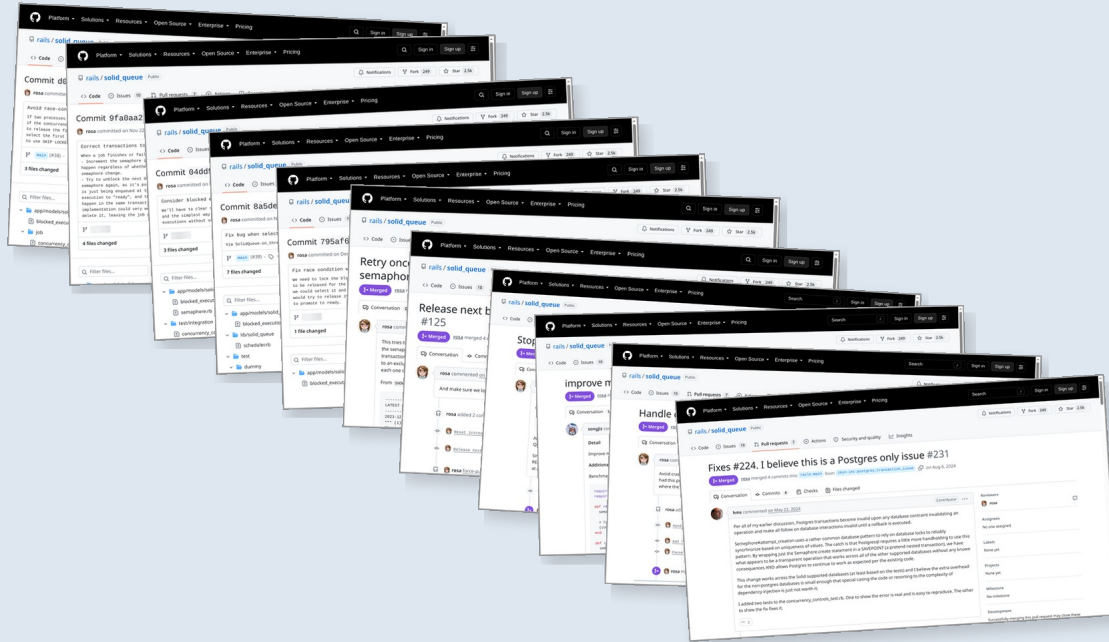
- Taste still matters
- Shipping and maintaining are still the definitive test
- Understanding your code still matters



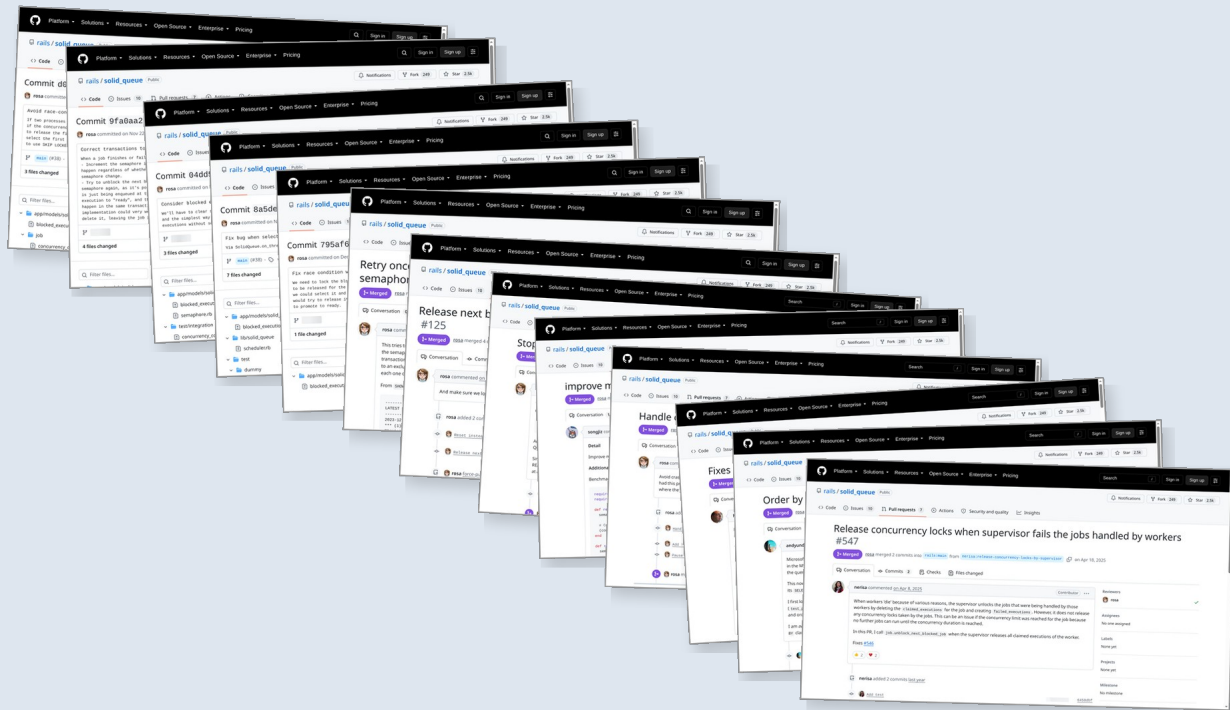
My most regretted feature



My most regretted feature



My most regretted feature



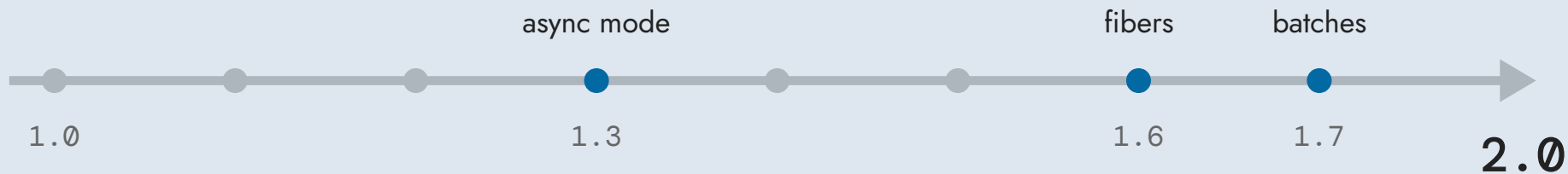
My most regretted feature



is also one of the
most Fun™ tested ones

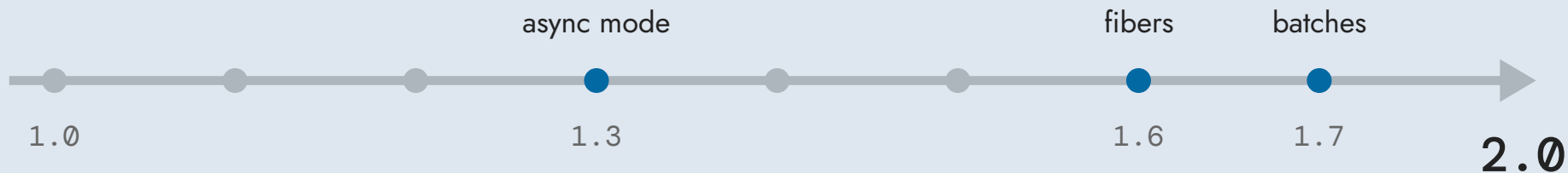


The path to 2.0



Sharding 🍔

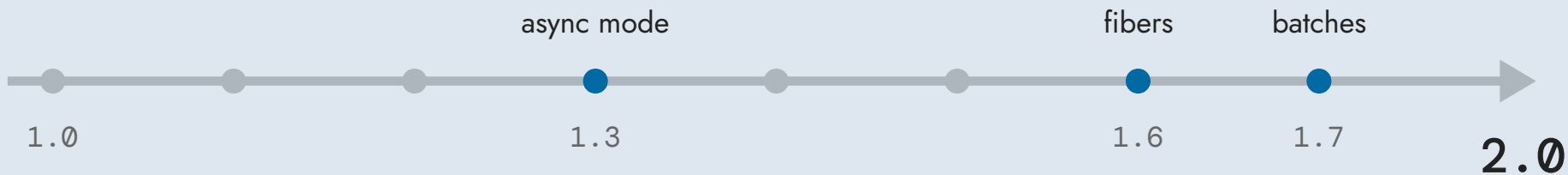
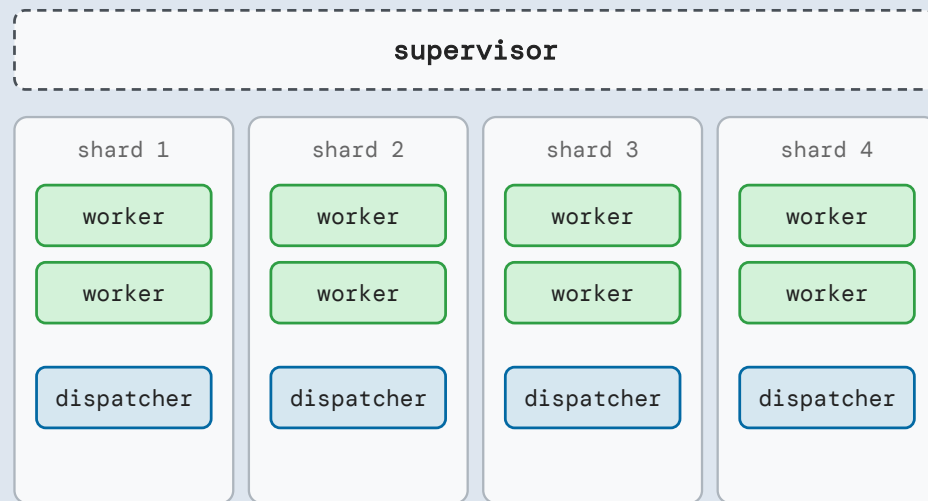
- Horizontal scaling
- A single DB has a ceiling
- Different from multi-tenancy



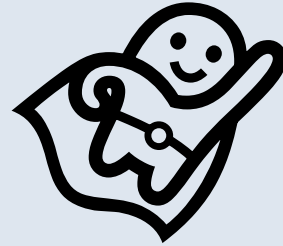
Sharding



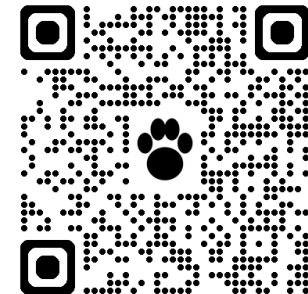
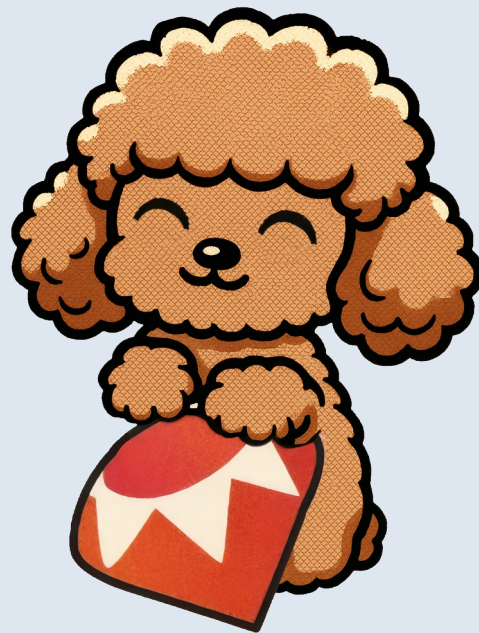
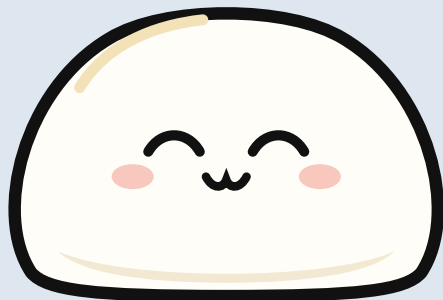
```
config.solid_queue.connects_to = {  
  shards: {  
    shard1: { writing: :solid_queue_shard1 },  
    shard2: { writing: :solid_queue_shard2 },  
    ...  
  }  
}
```



Onward



もち



Slides & more

みなさん、ありがとうございました!