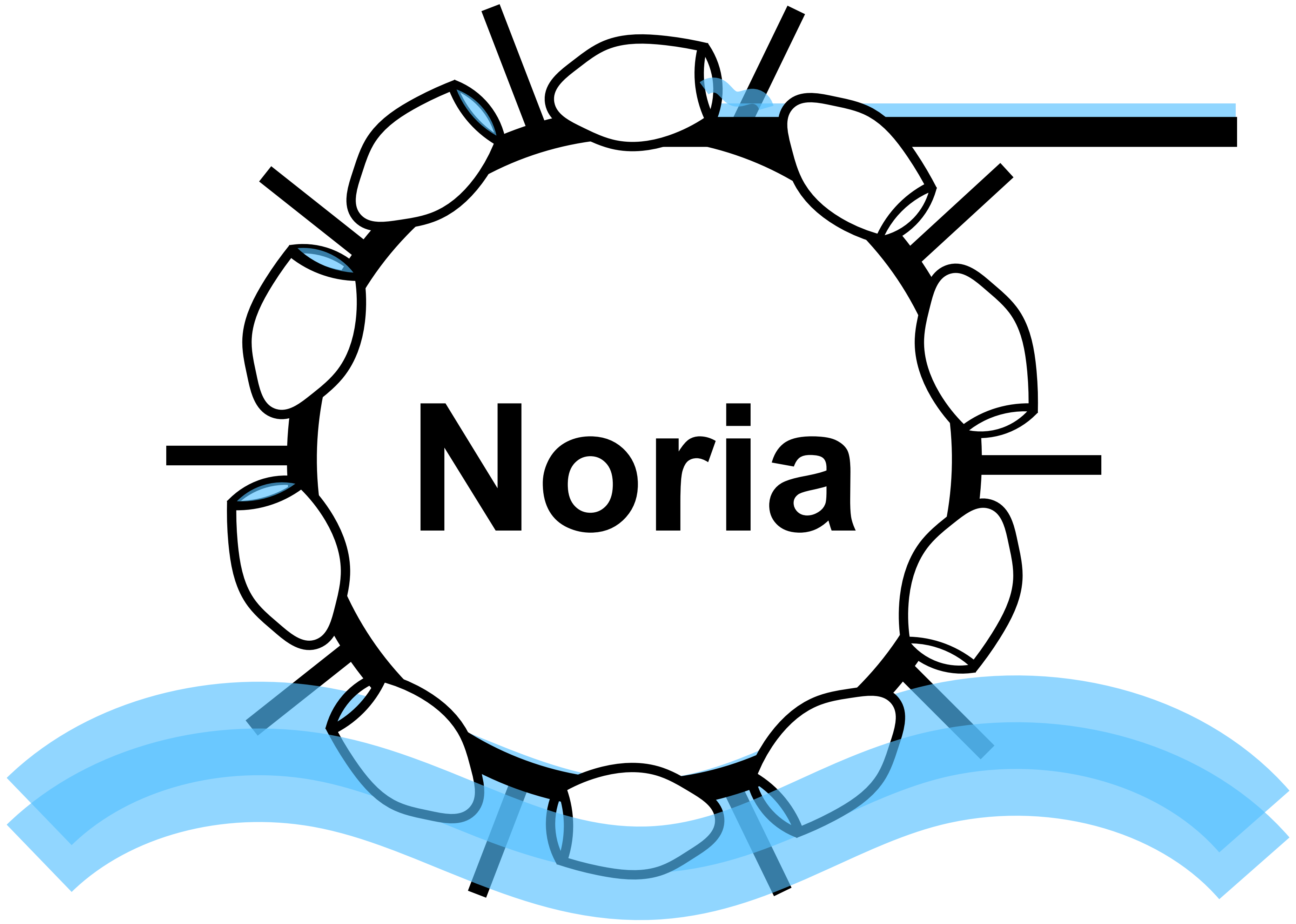


Dynamic, partially-stateful data-flow for high-performance Web applications

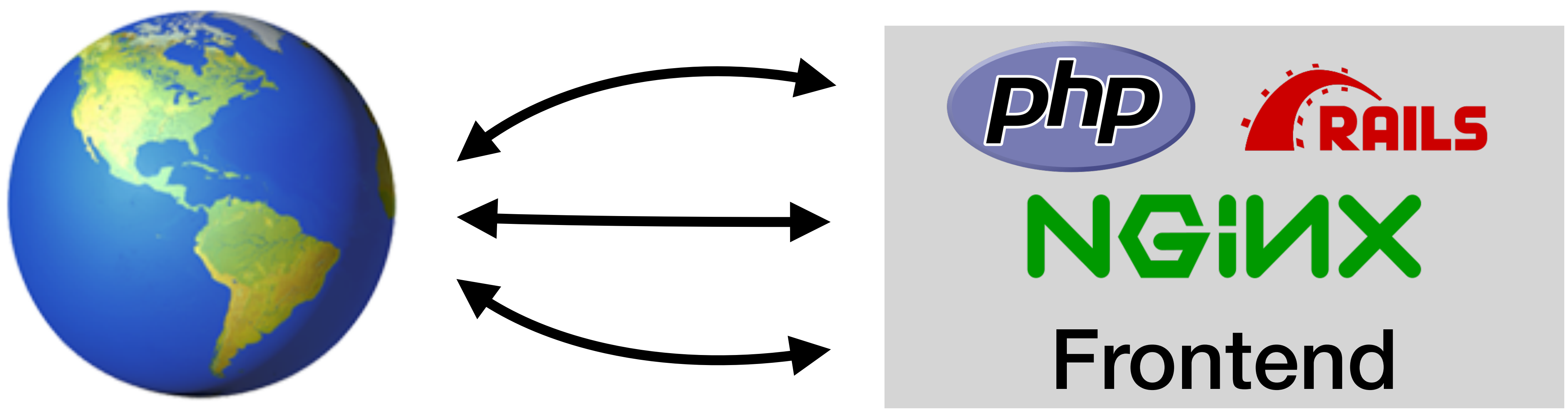
Jon Gjengset Malte Schwarzkopf

Jonathan Behrens Lara Timbó Araújo Martin Ek

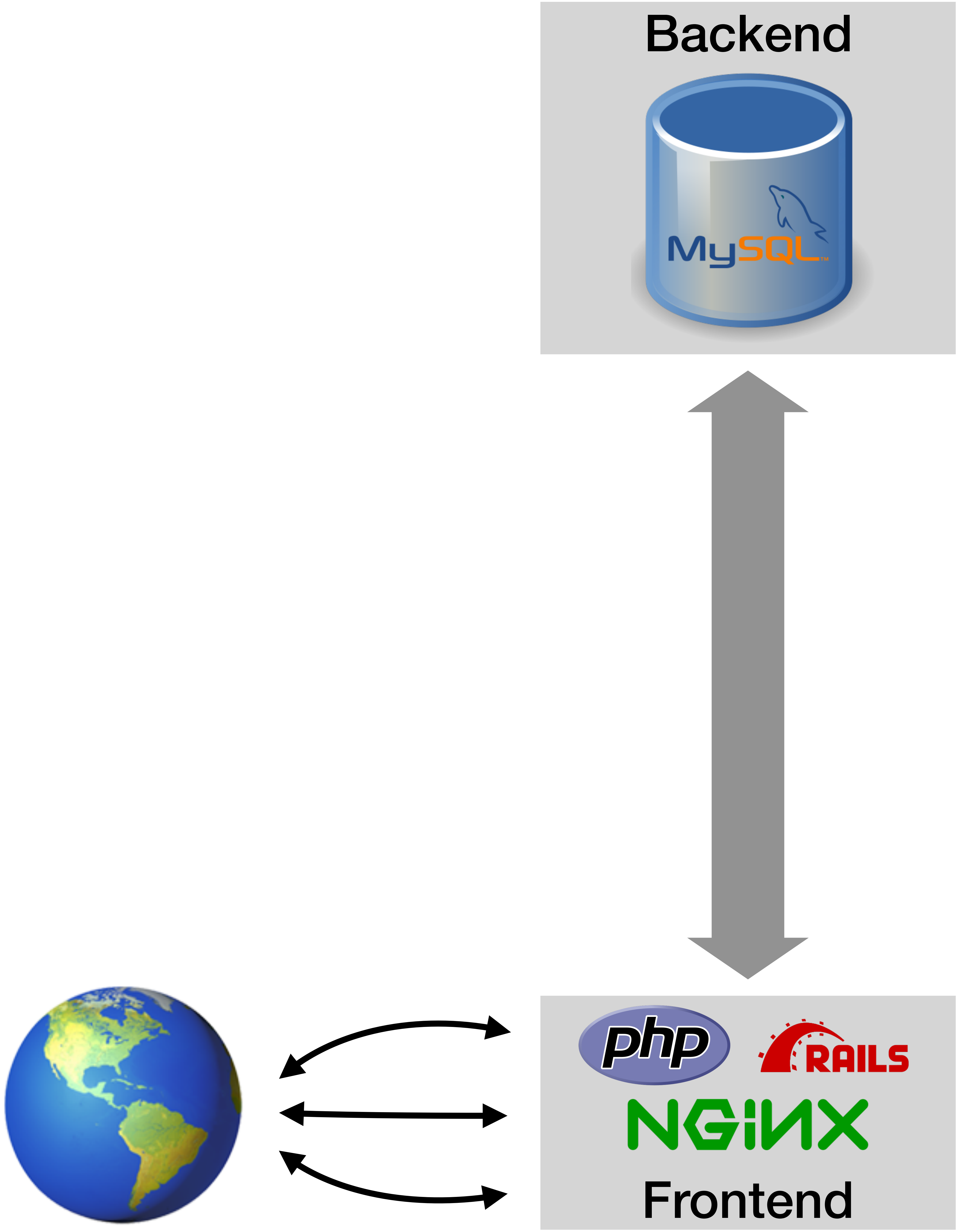
Eddie Kohler M. Frans Kaashoek Robert Morris

php
RAILS
NGINX

Frontend

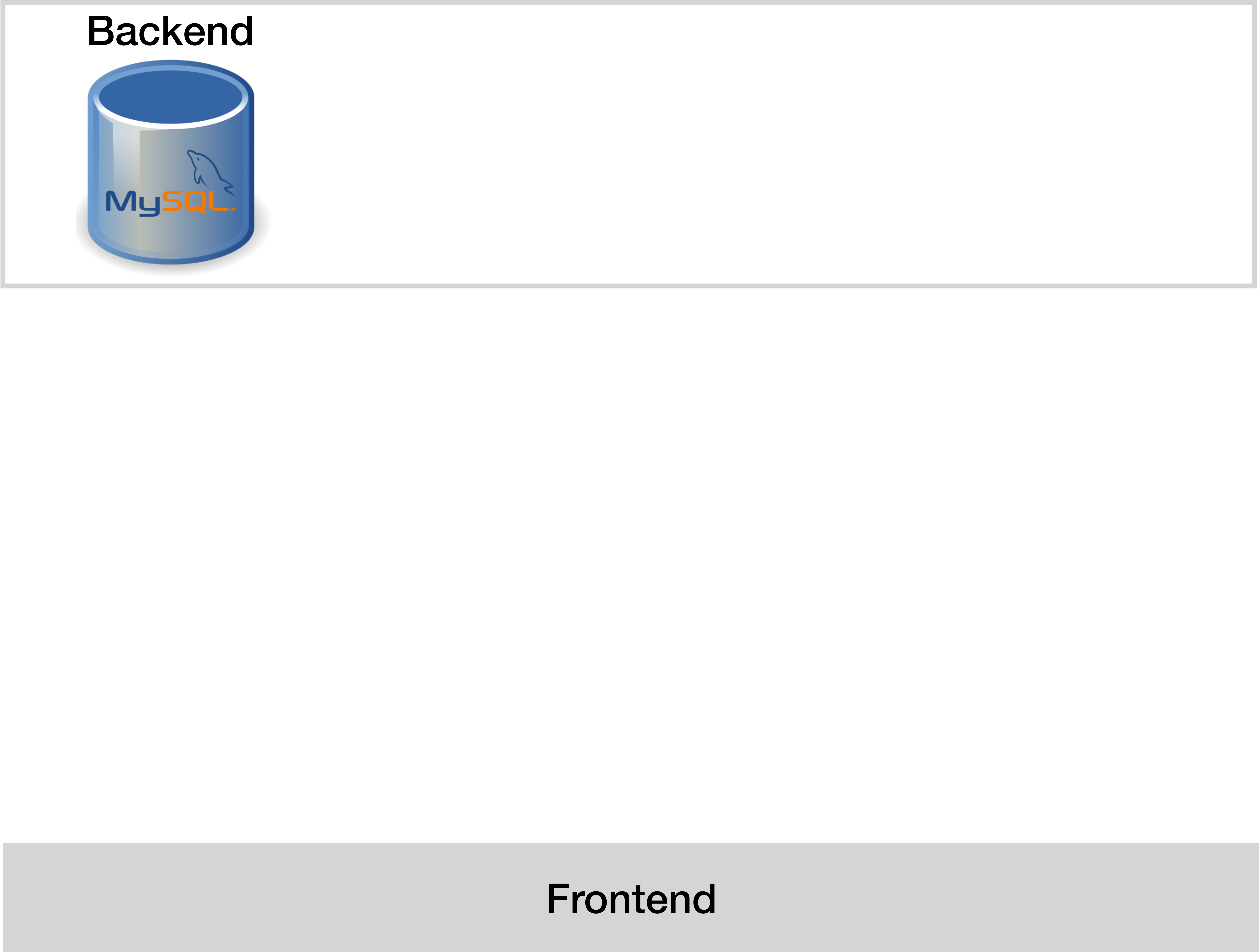


Frontend



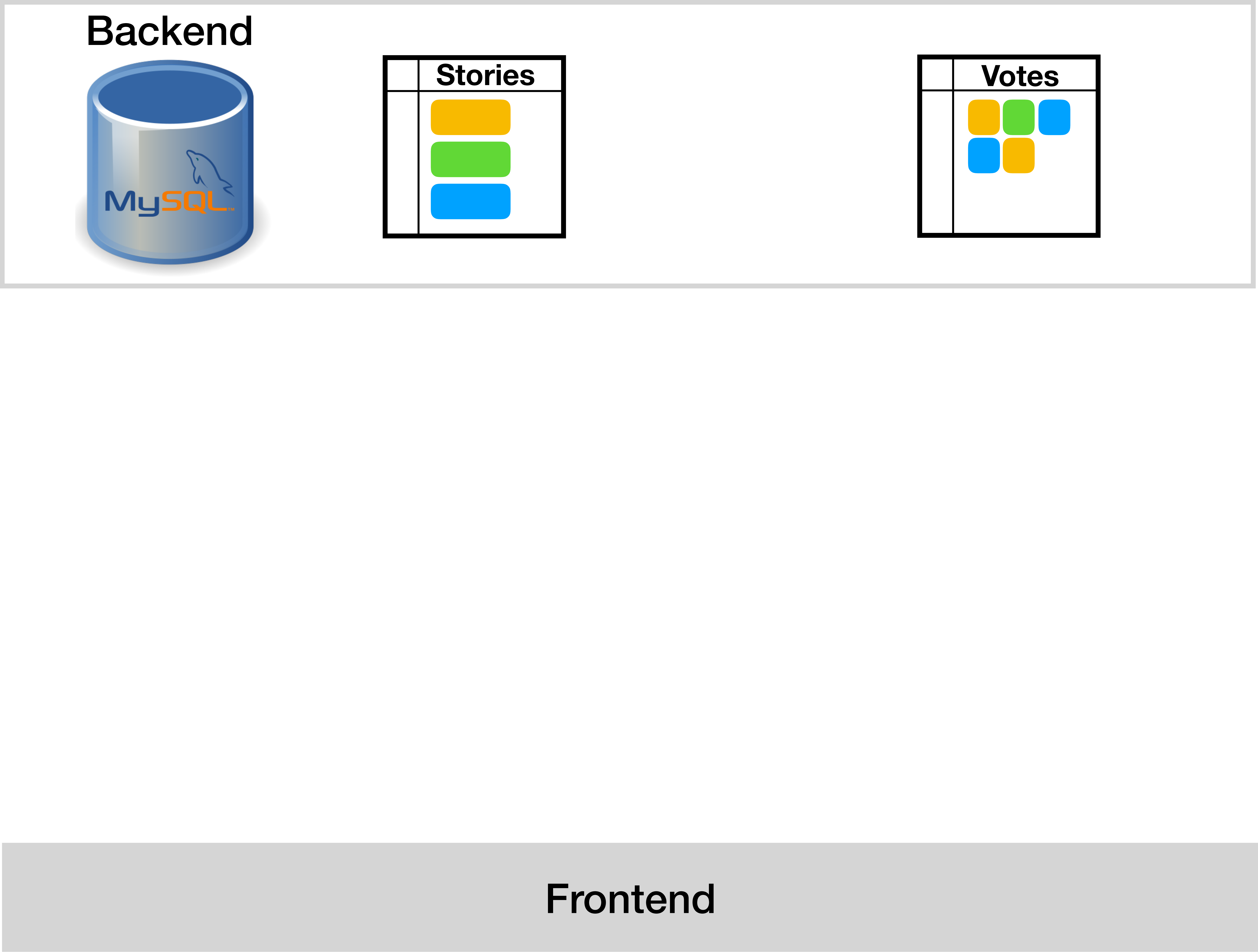
Backend

Frontend



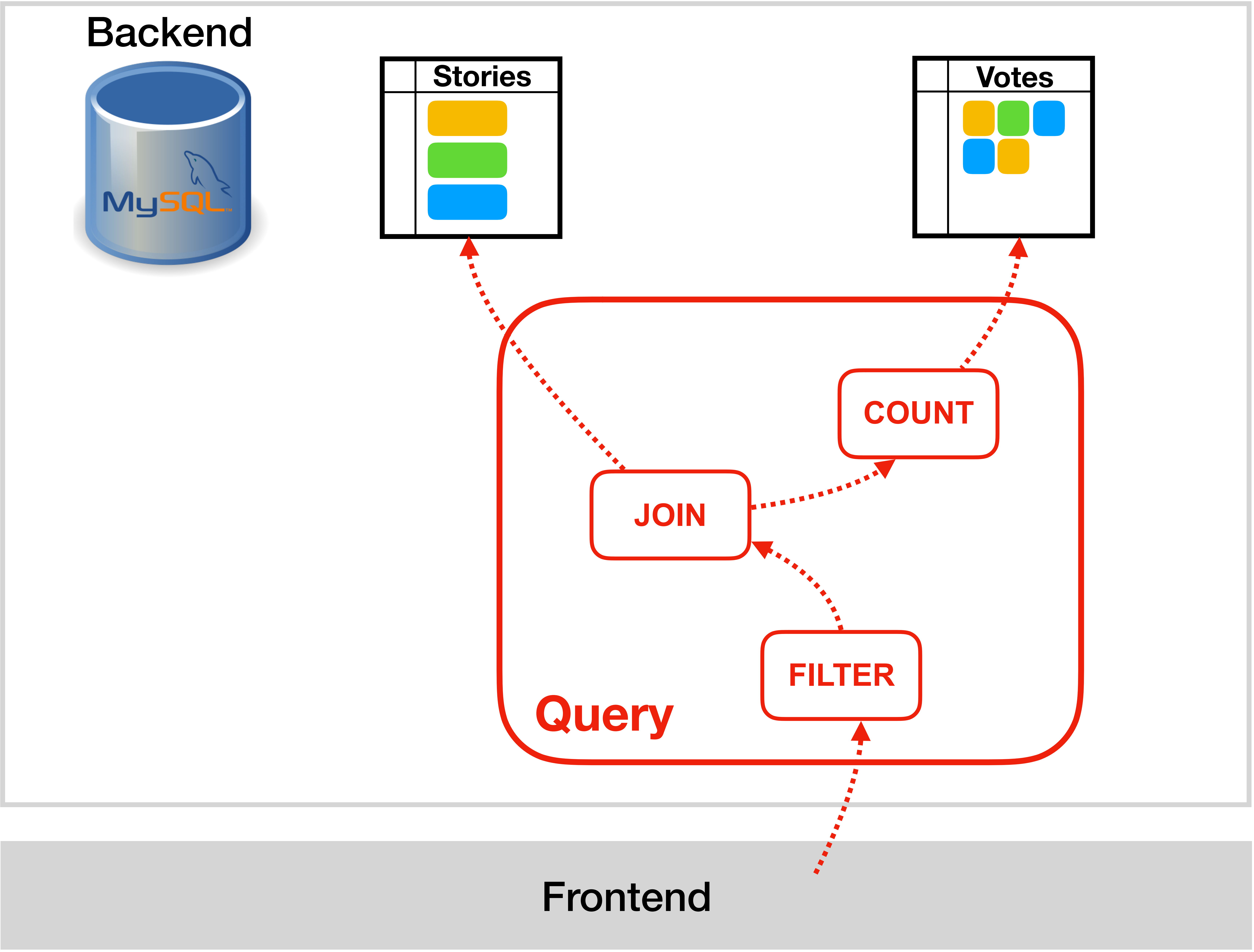
Backend

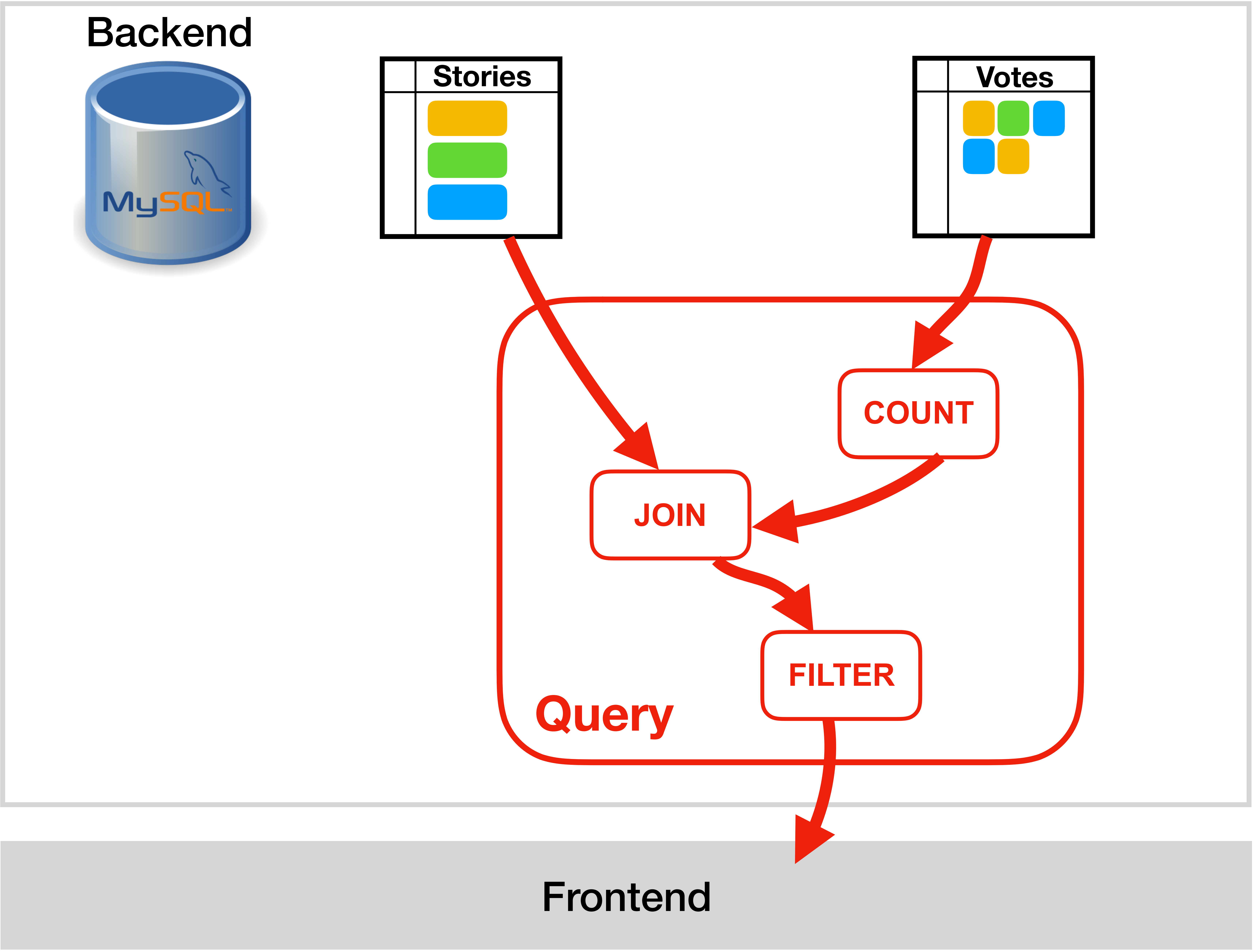
Frontend



Backend

Frontend


Backend
MySQL
Stories
Votes
COUNT
JOIN
FILTER
Query
Frontend



Backend

Stories

Votes

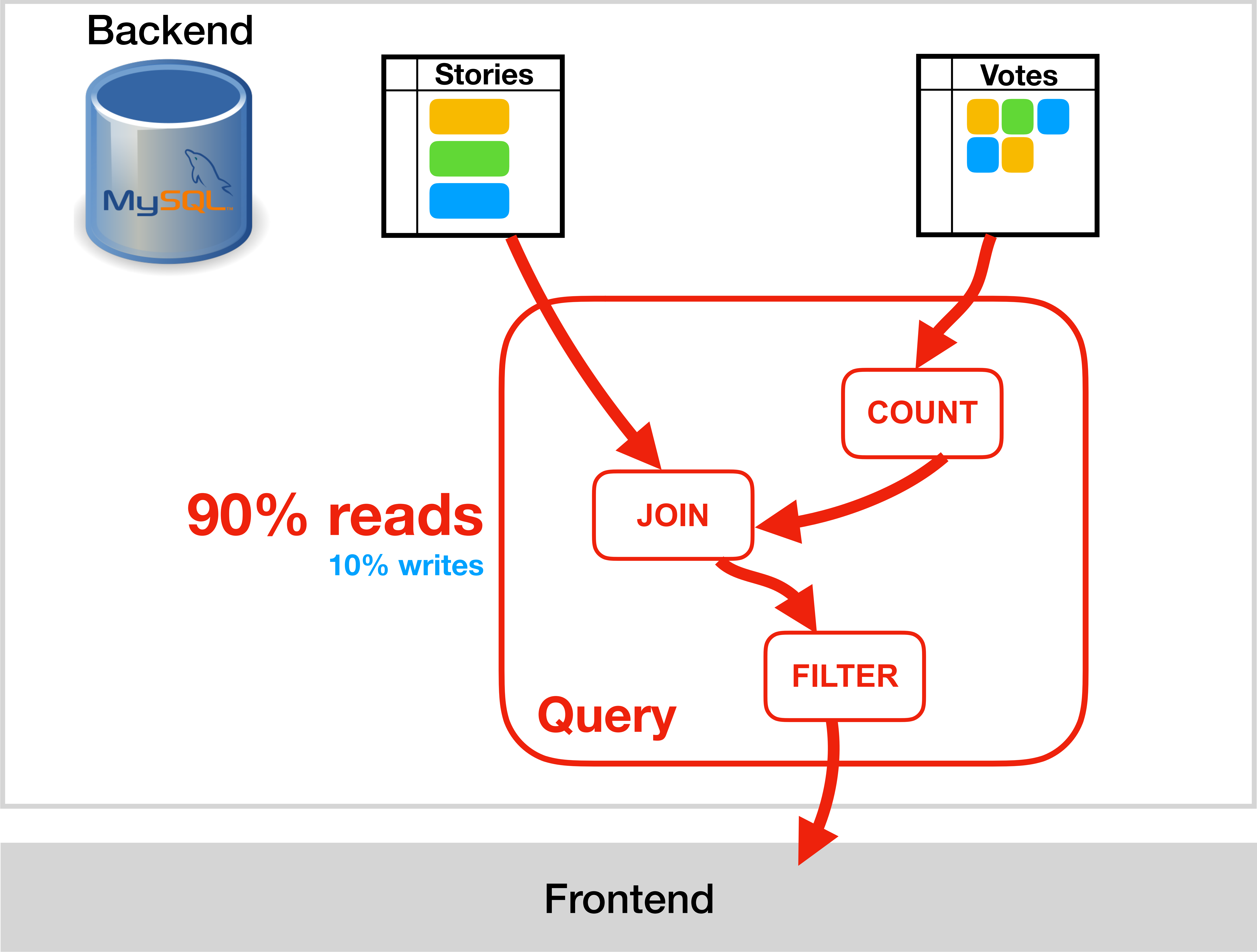
COUNT

JOIN

FILTER

Query

Frontend

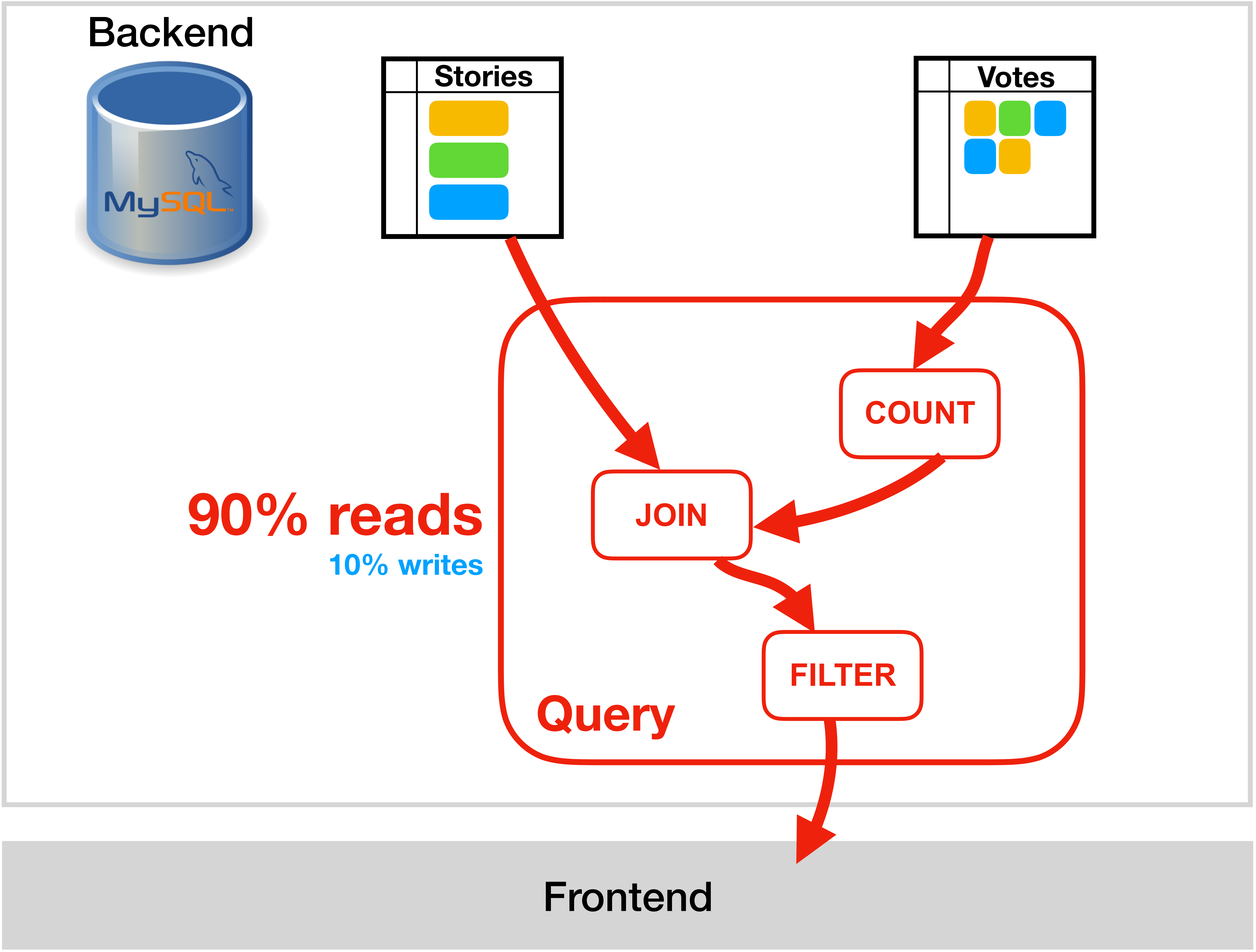


90% reads

10% writes

Slow reads, repeated work!

🙁



Backend

Stories

Votes

COUNT

JOIN

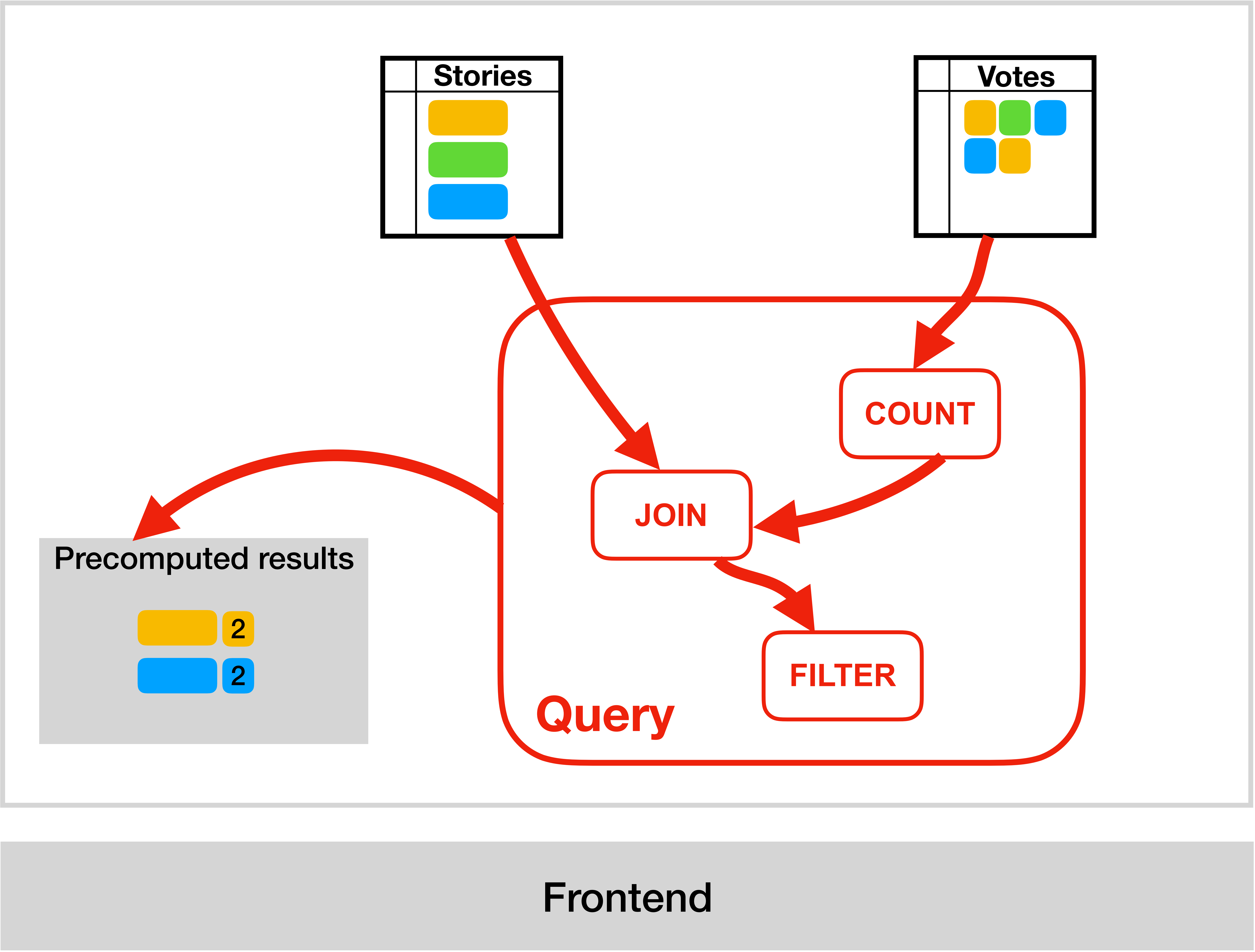
FILTER

Query

90% reads

10% writes

Frontend



Stories

Votes

Query

JOIN

COUNT

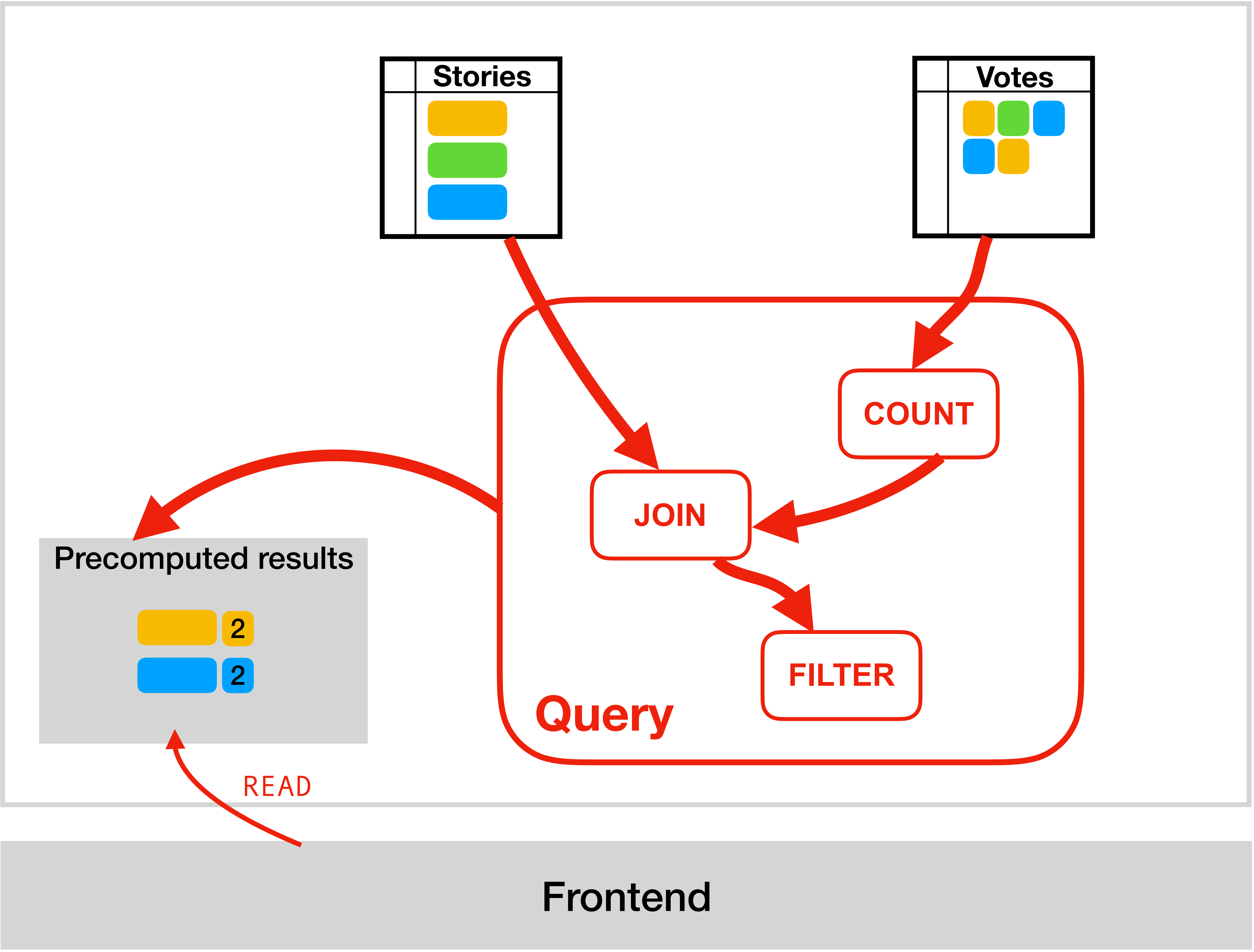
FILTER

Precomputed results

2

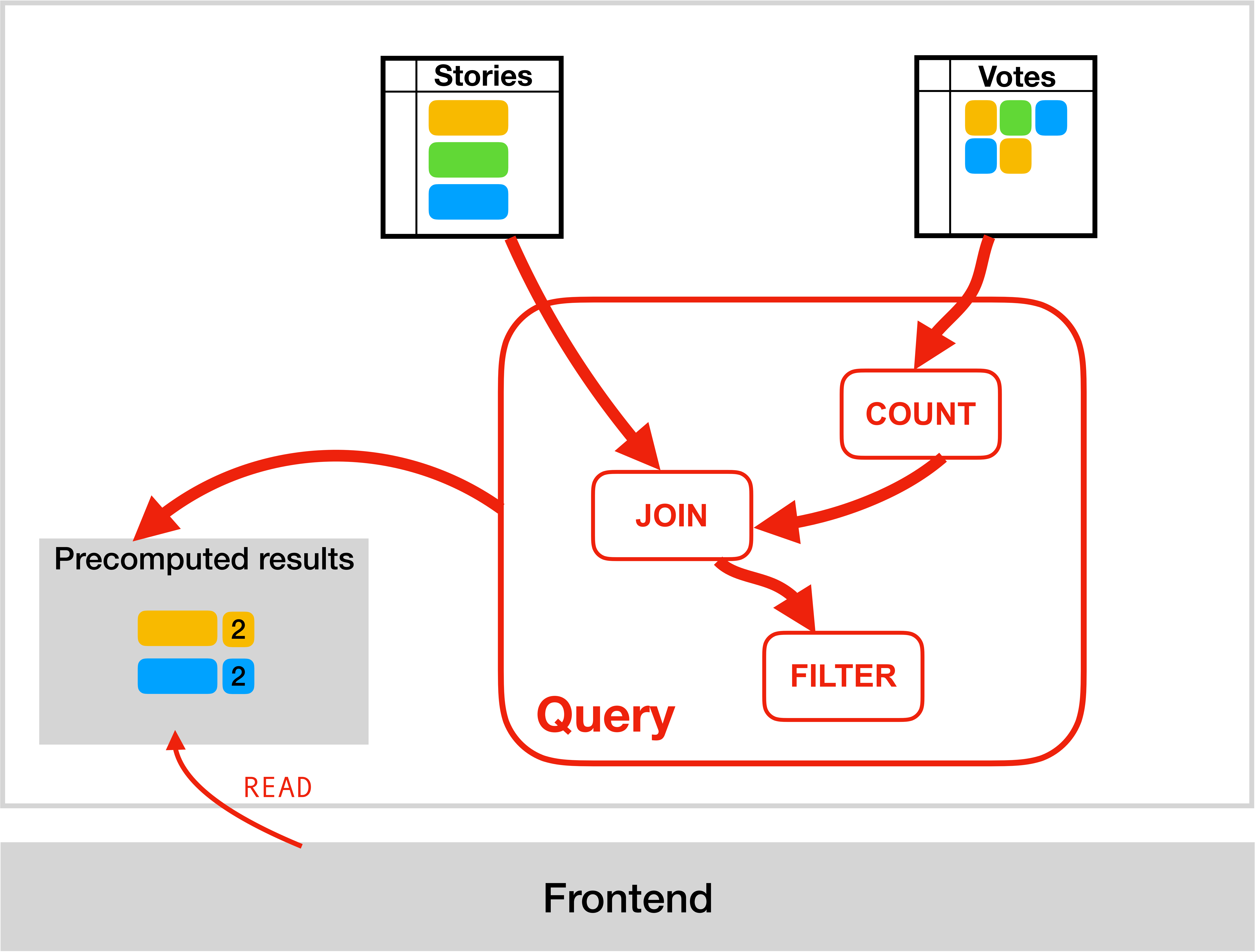
2

Frontend


Stories
Votes
COUNT
JOIN
FILTER
Query
Precomputed results
2
2
READ
Frontend

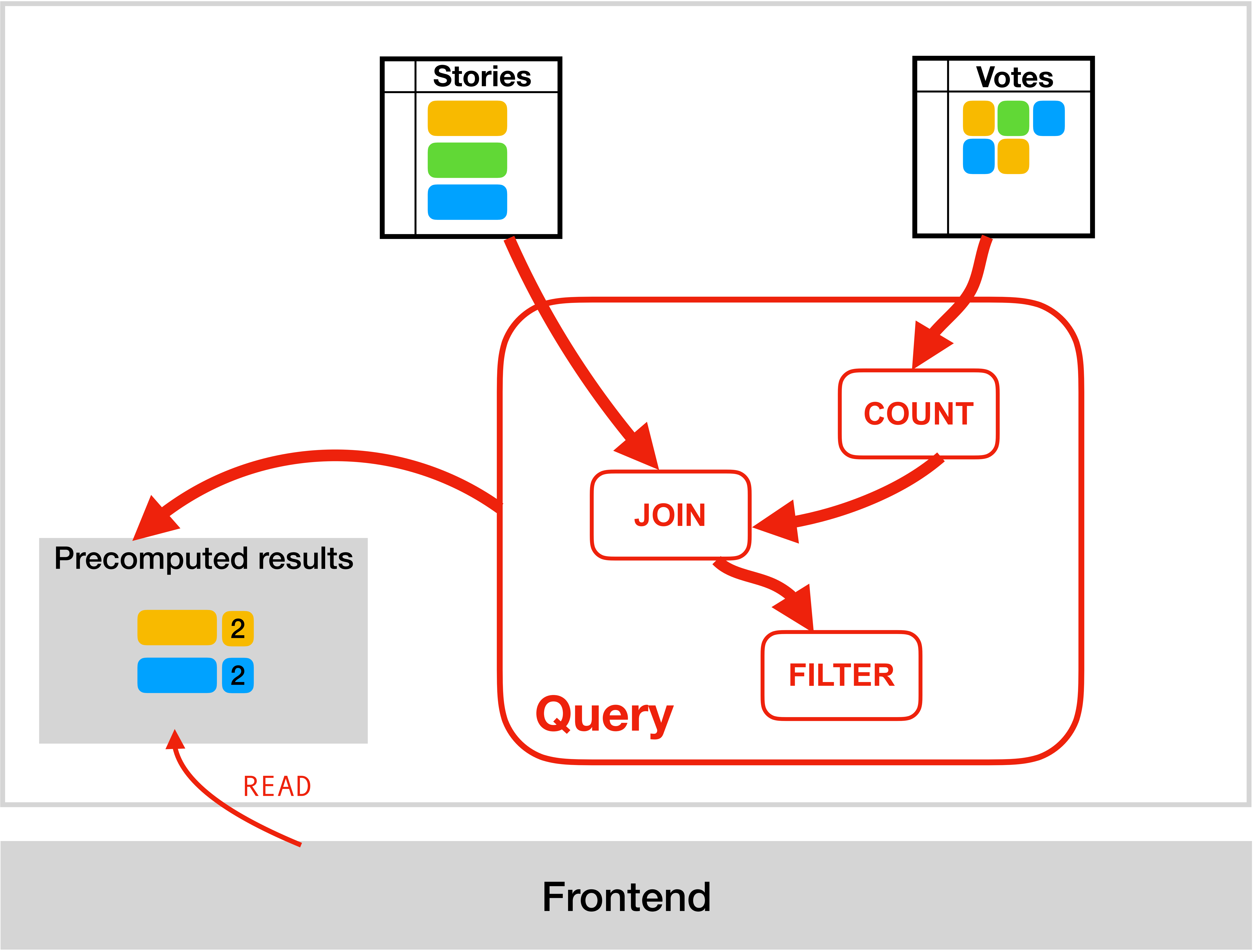
Store in base table?

— manual, slow.



Store in base table?
— **manual, slow**.

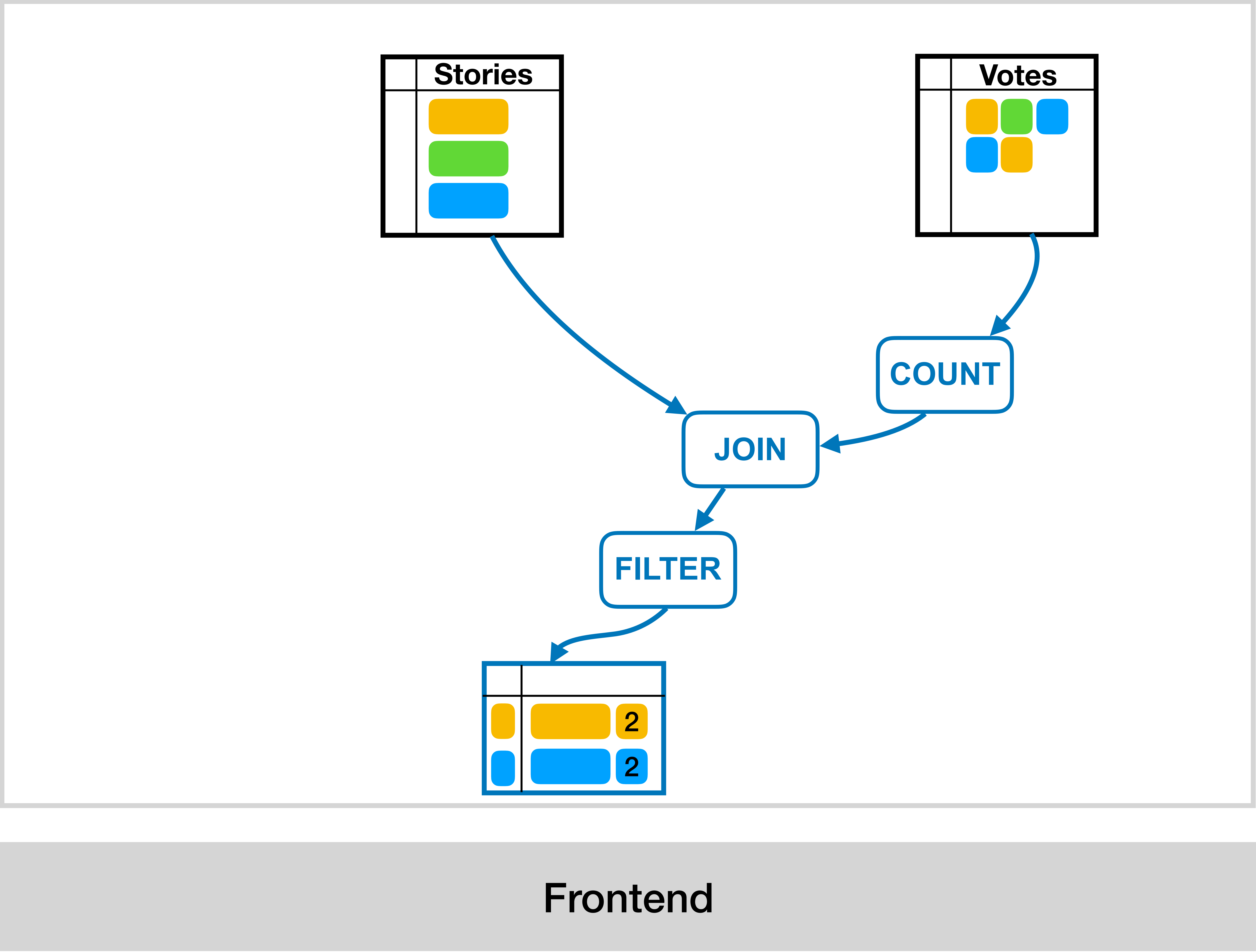
memcached?
— **complex** [Facebook NSDI'13].



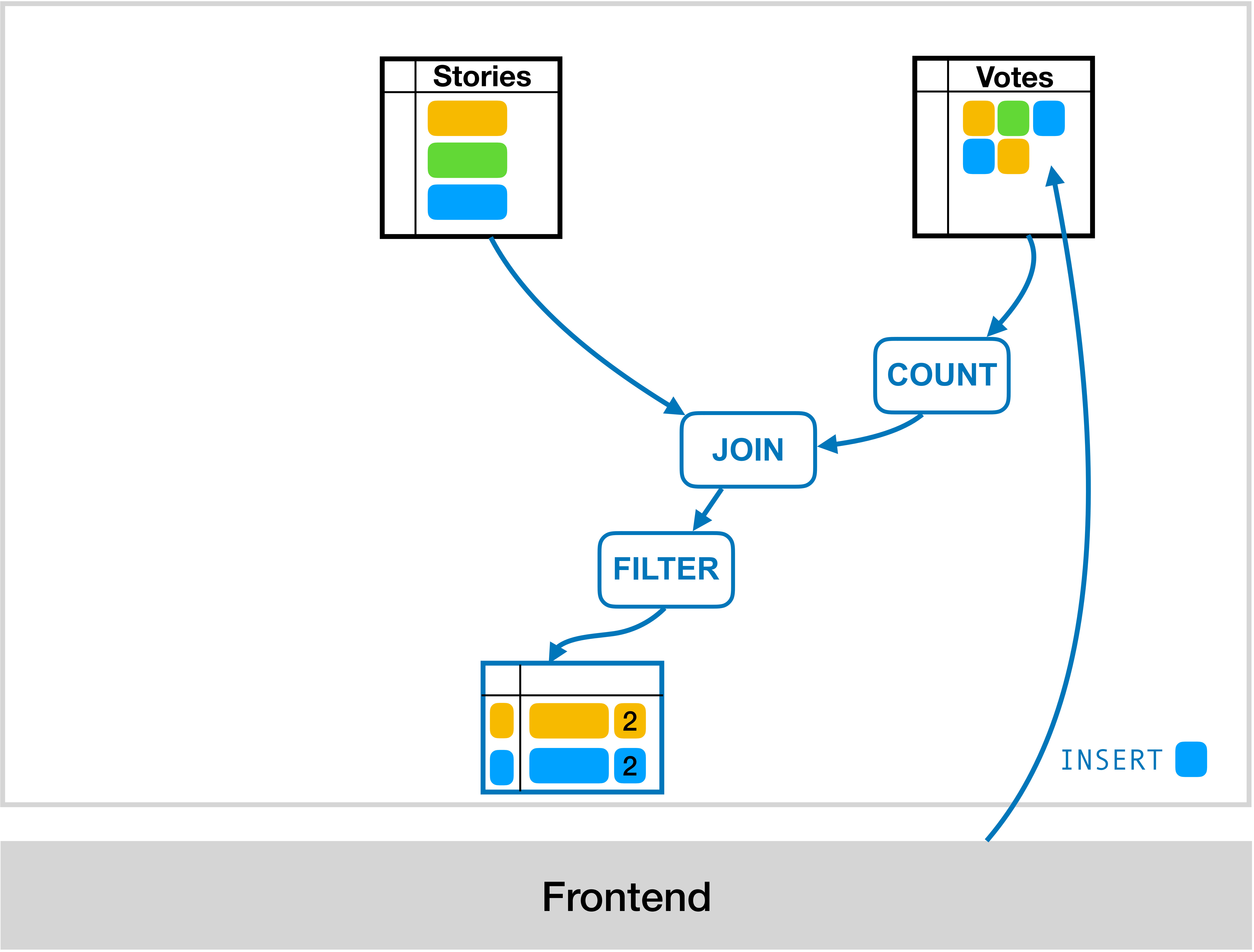
Store in base table?
— manual, slow.

memcached?
— complex [Facebook NSDI'13].

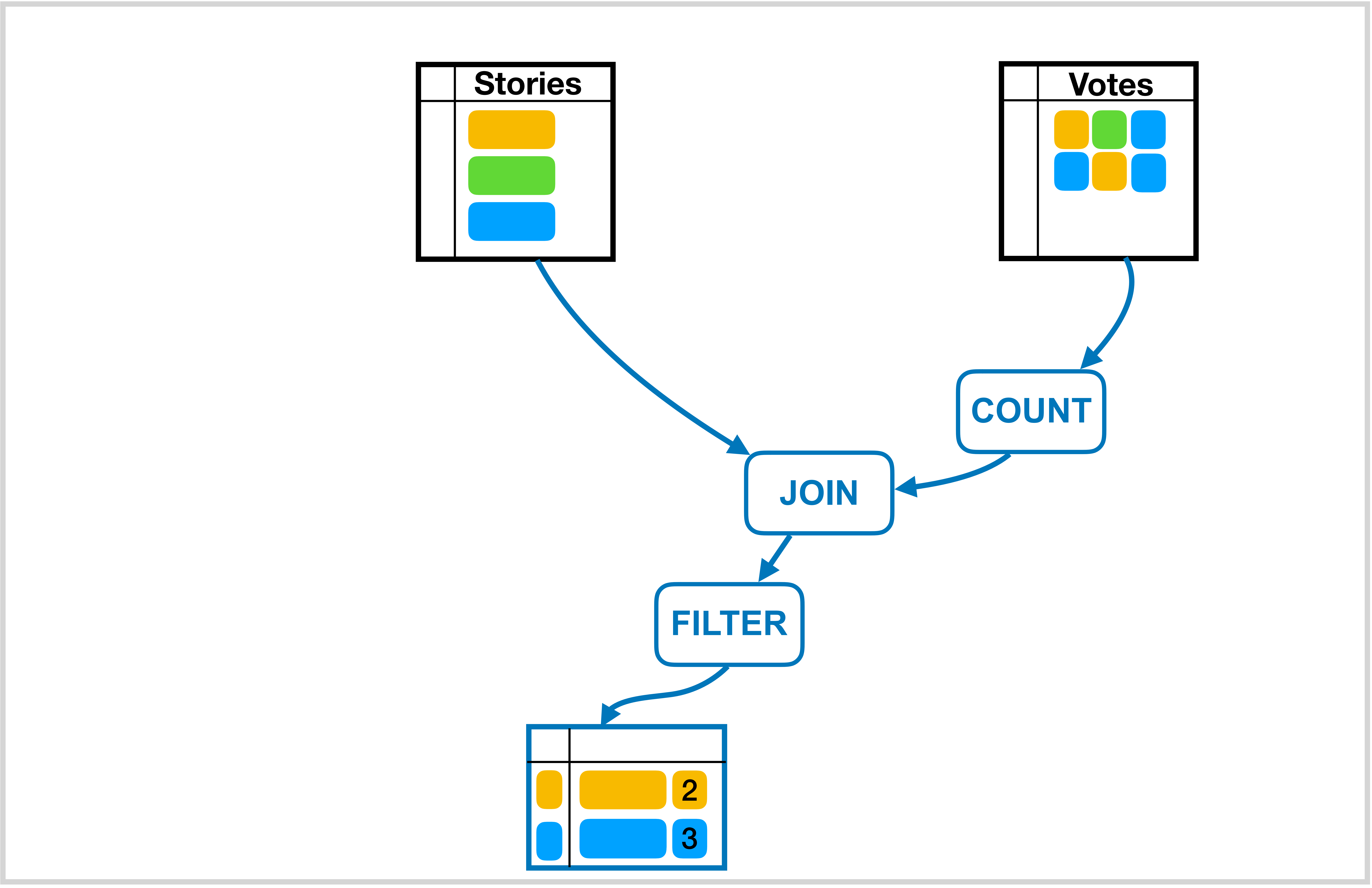
Streaming data-flow?



Streaming data-flow?

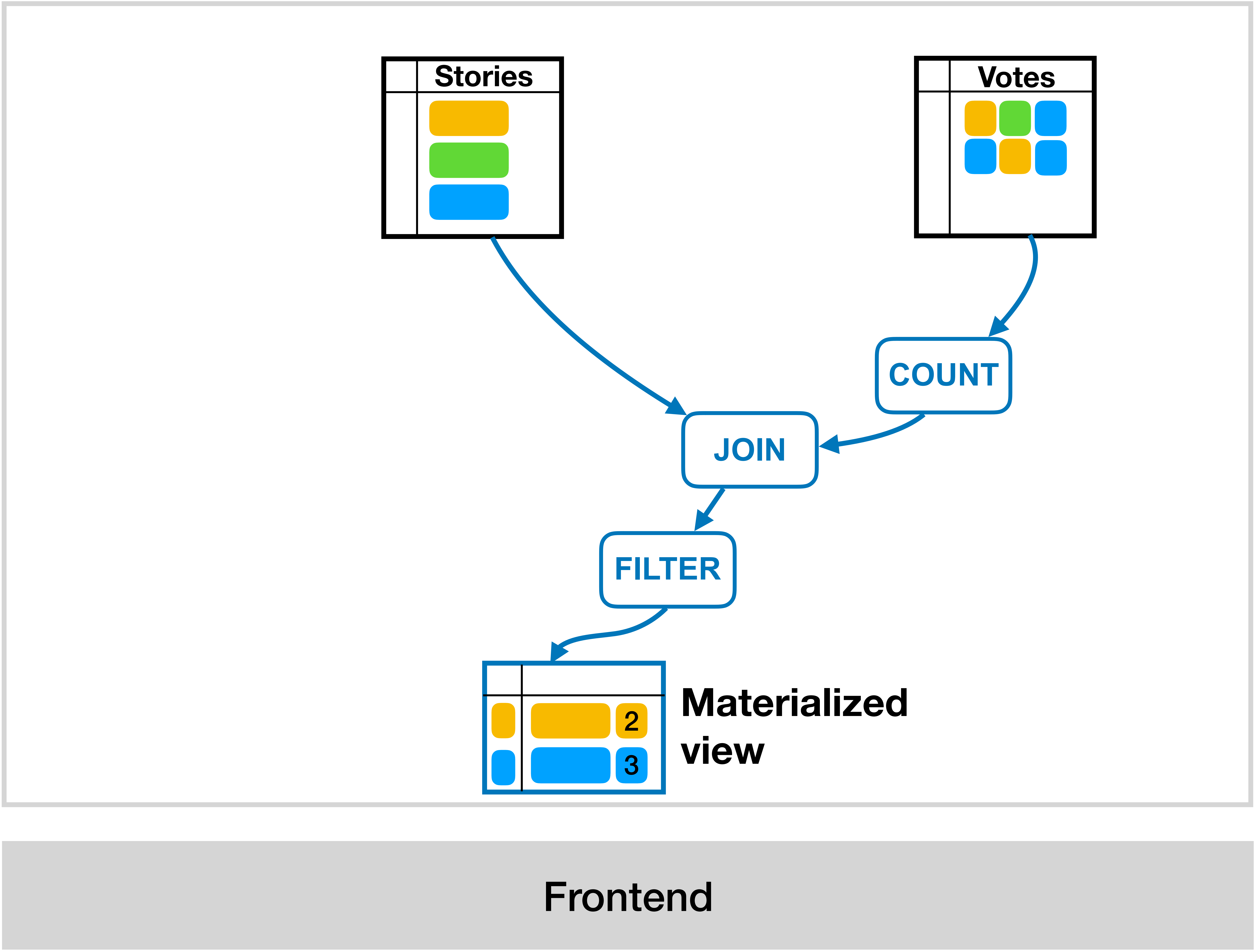


Streaming data-flow?



Frontend

Streaming data-flow?



Stories

Votes

COUNT

JOIN

FILTER

2

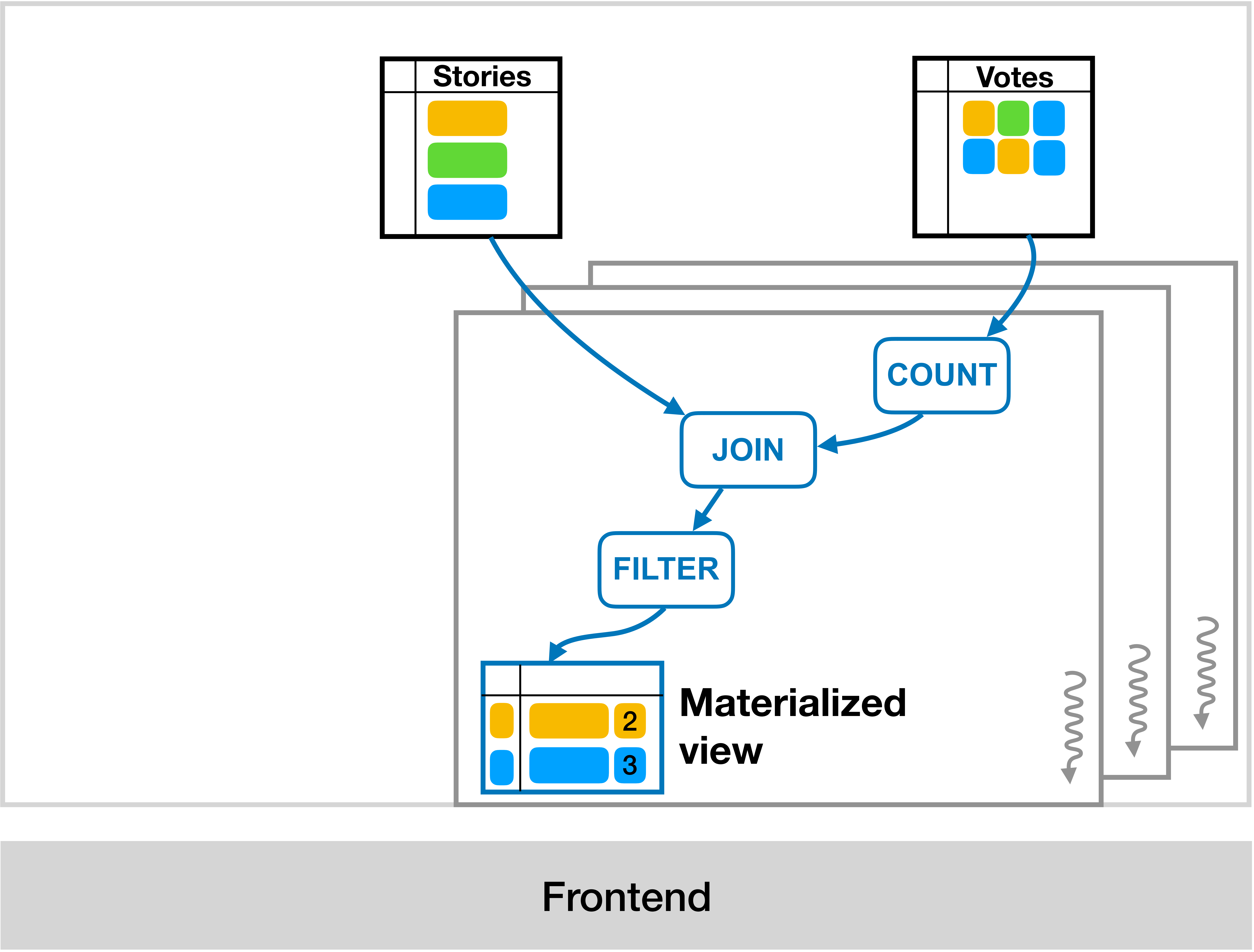
3

Materialized view

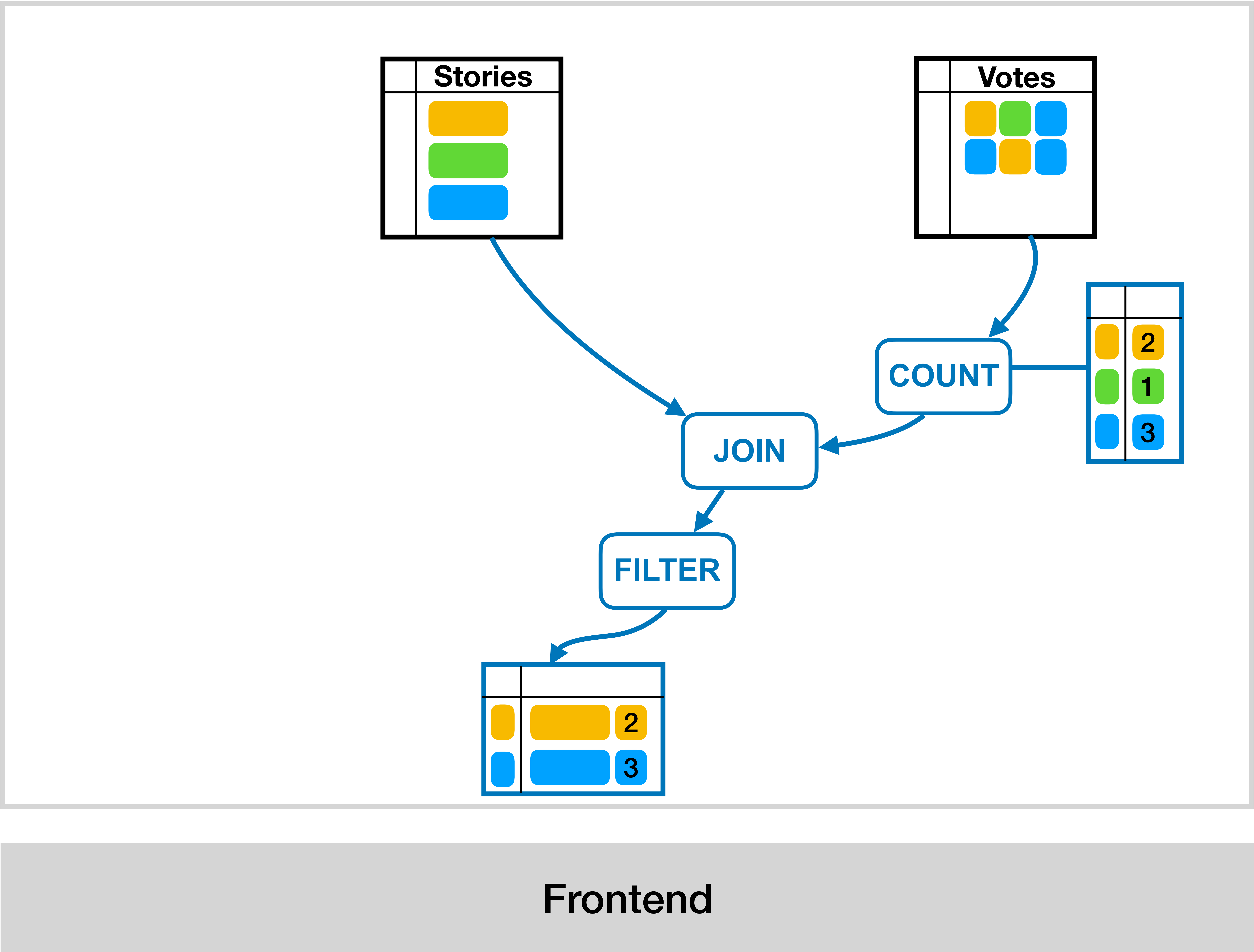
Frontend

Streaming data-flow?

Fast reads.
Efficient writes.
Parallelizes well.



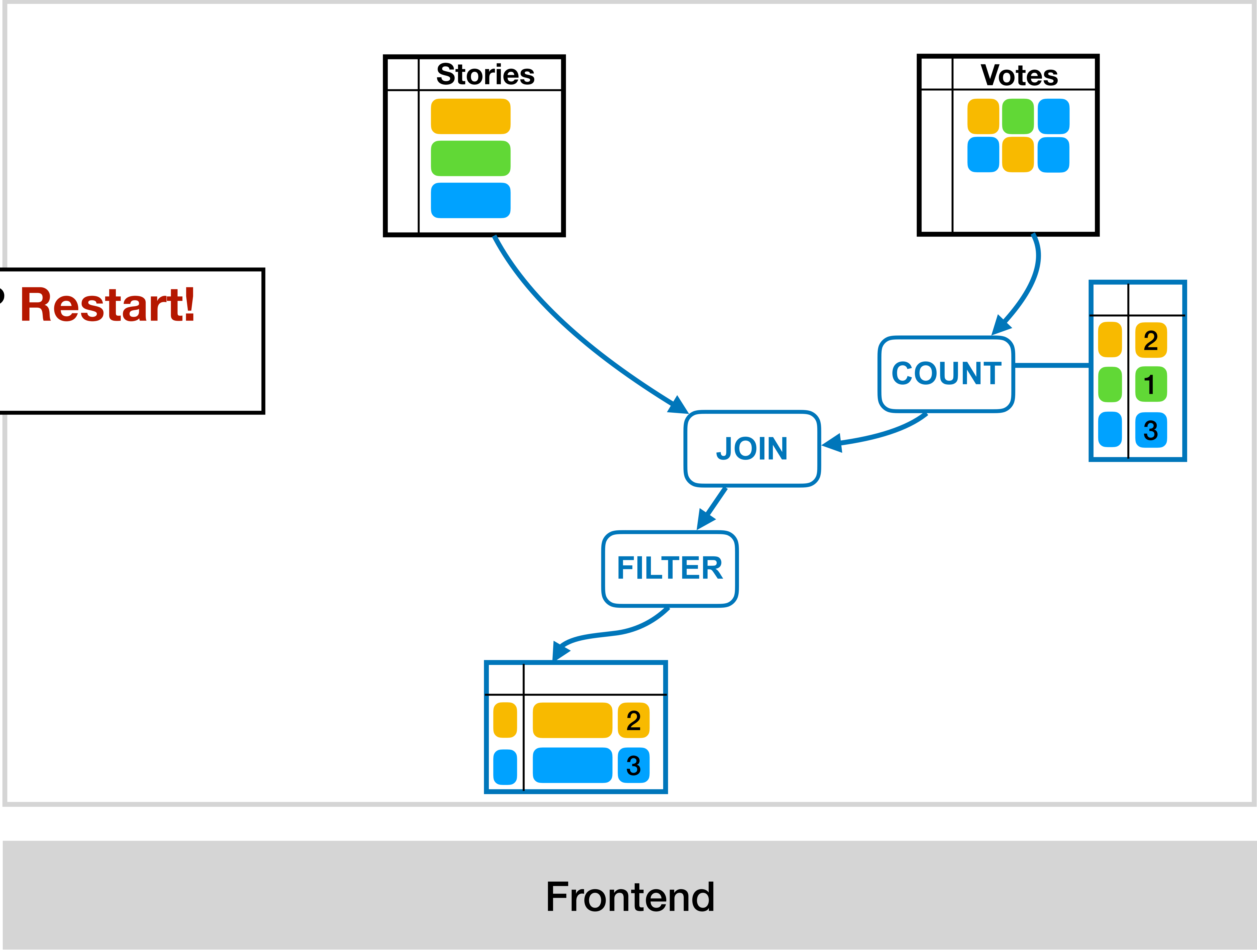
Challenges



Challenges

State-of-the-art data-flow systems:

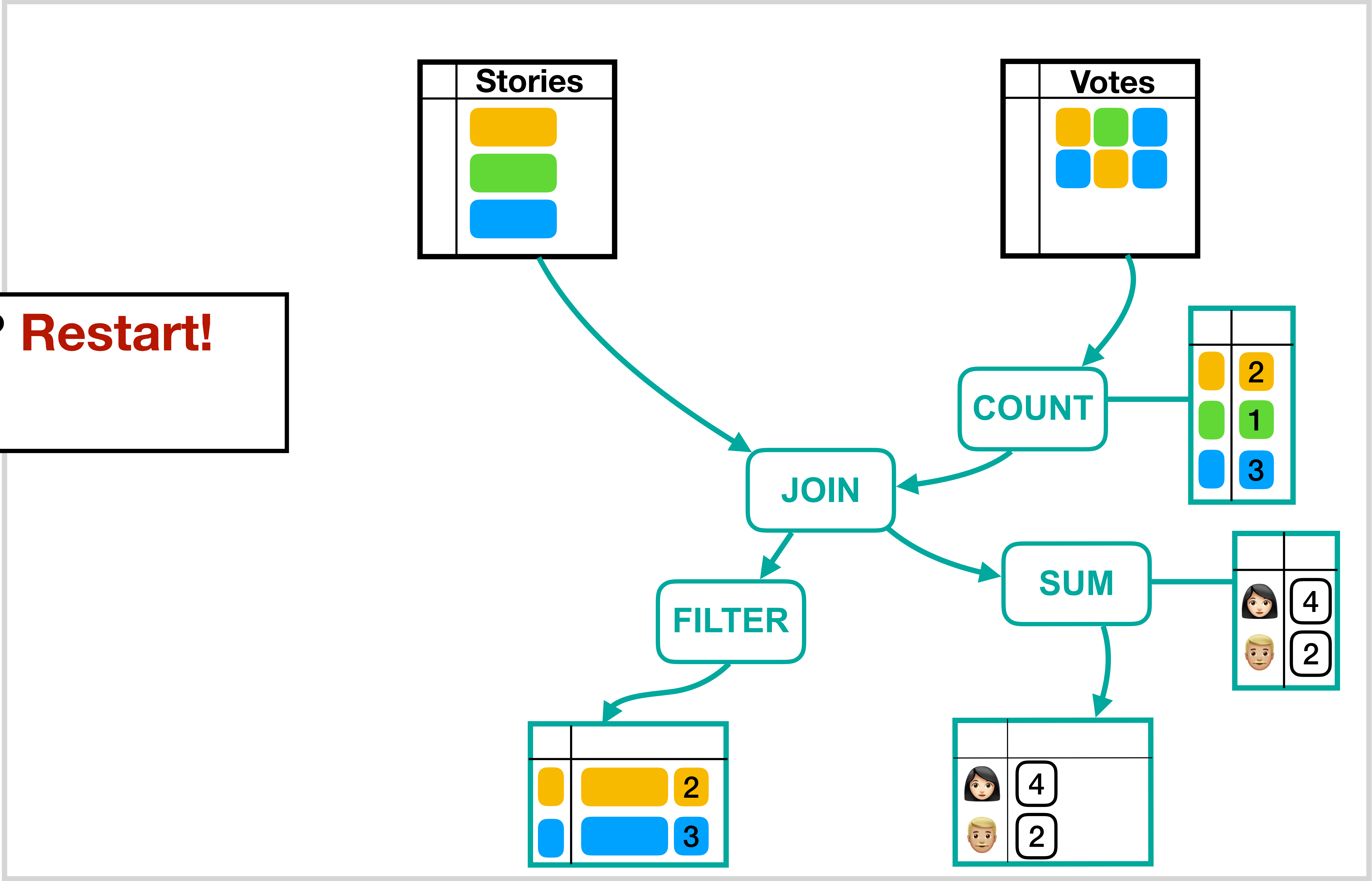
- **Change queries? Restart!**



Challenges

State-of-the-art data-flow systems:

- **Change queries? Restart!**

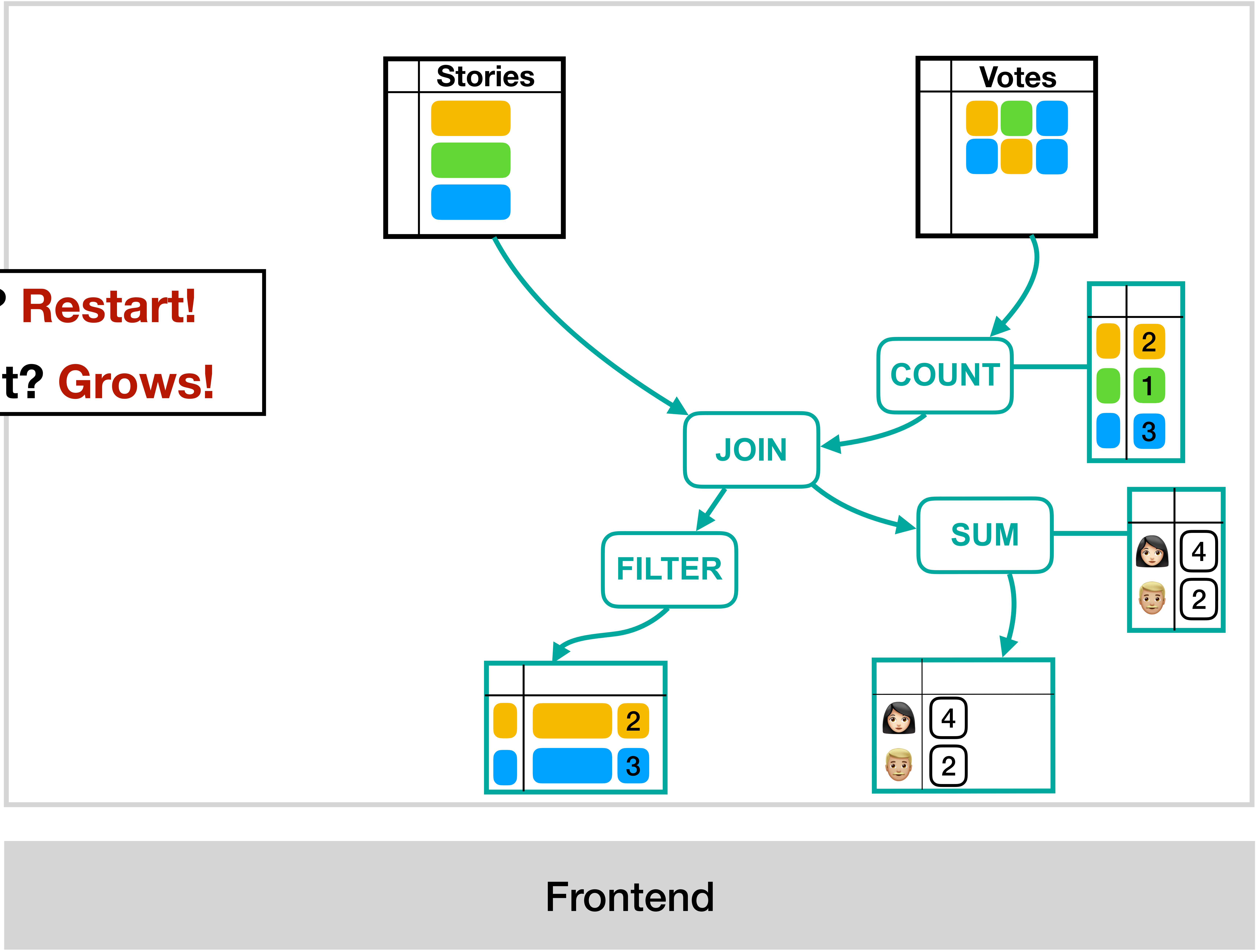


Frontend

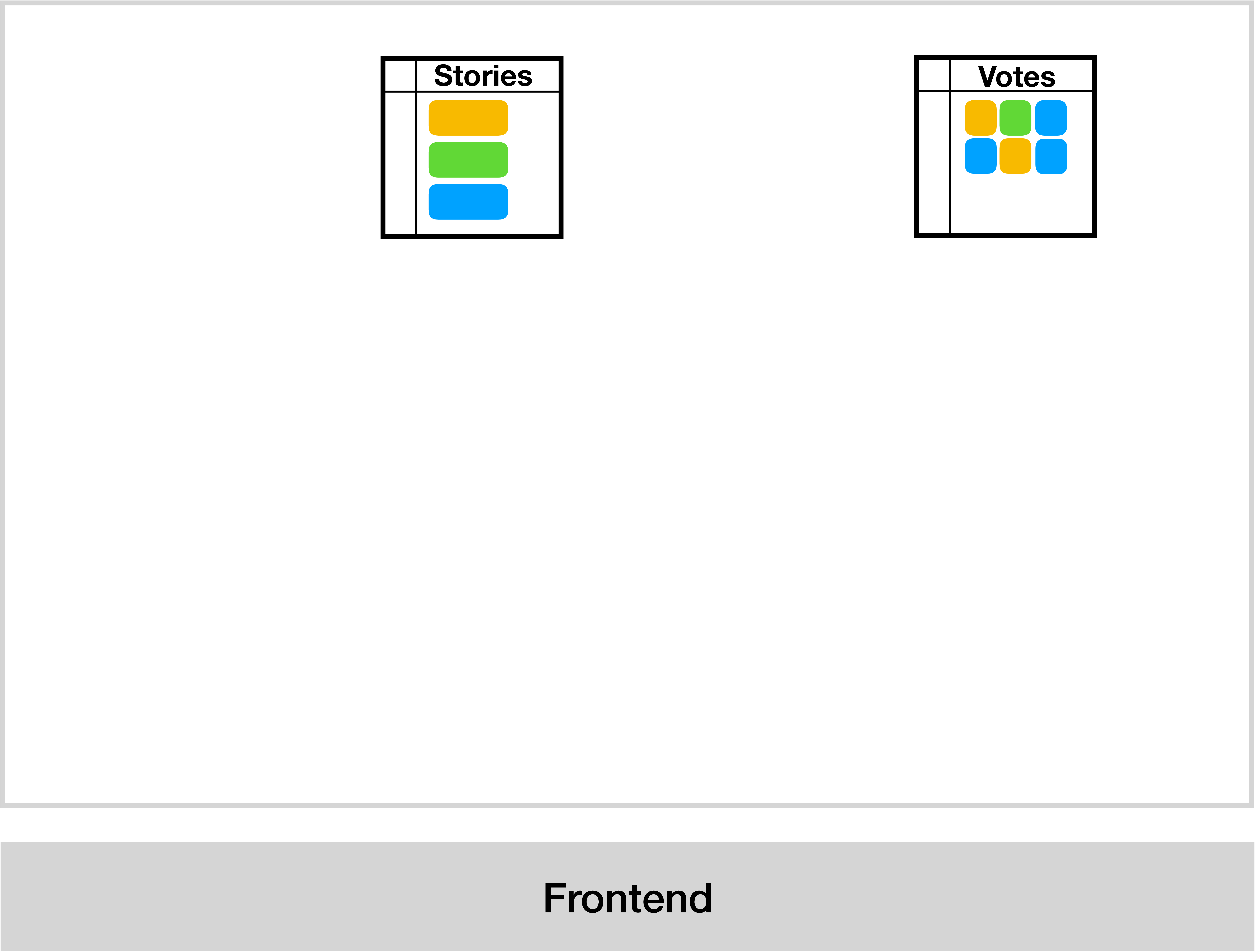
Challenges

State-of-the-art
data-flow systems:

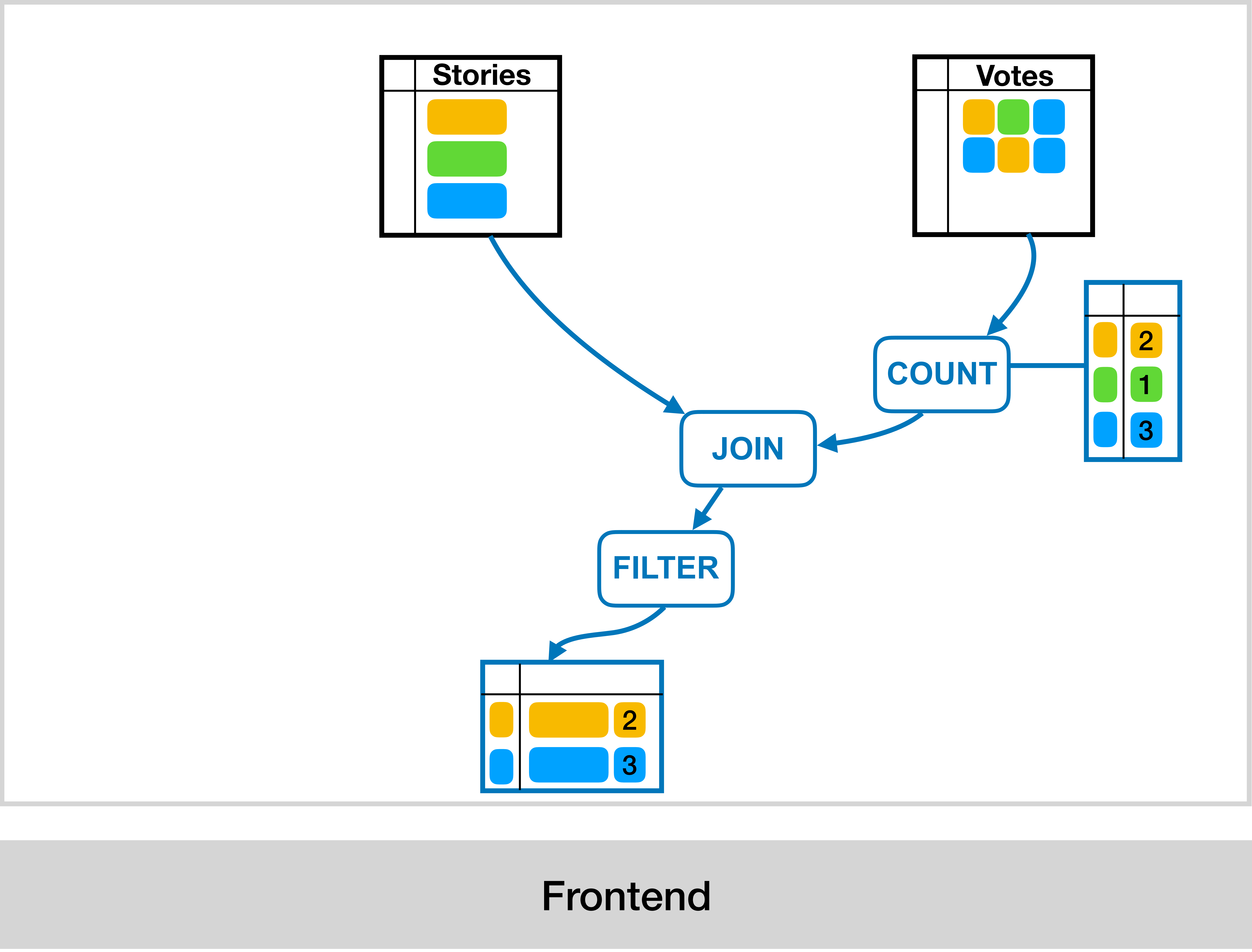
- **Change queries? Restart!**
- **Memory footprint? Grows!**



Noria

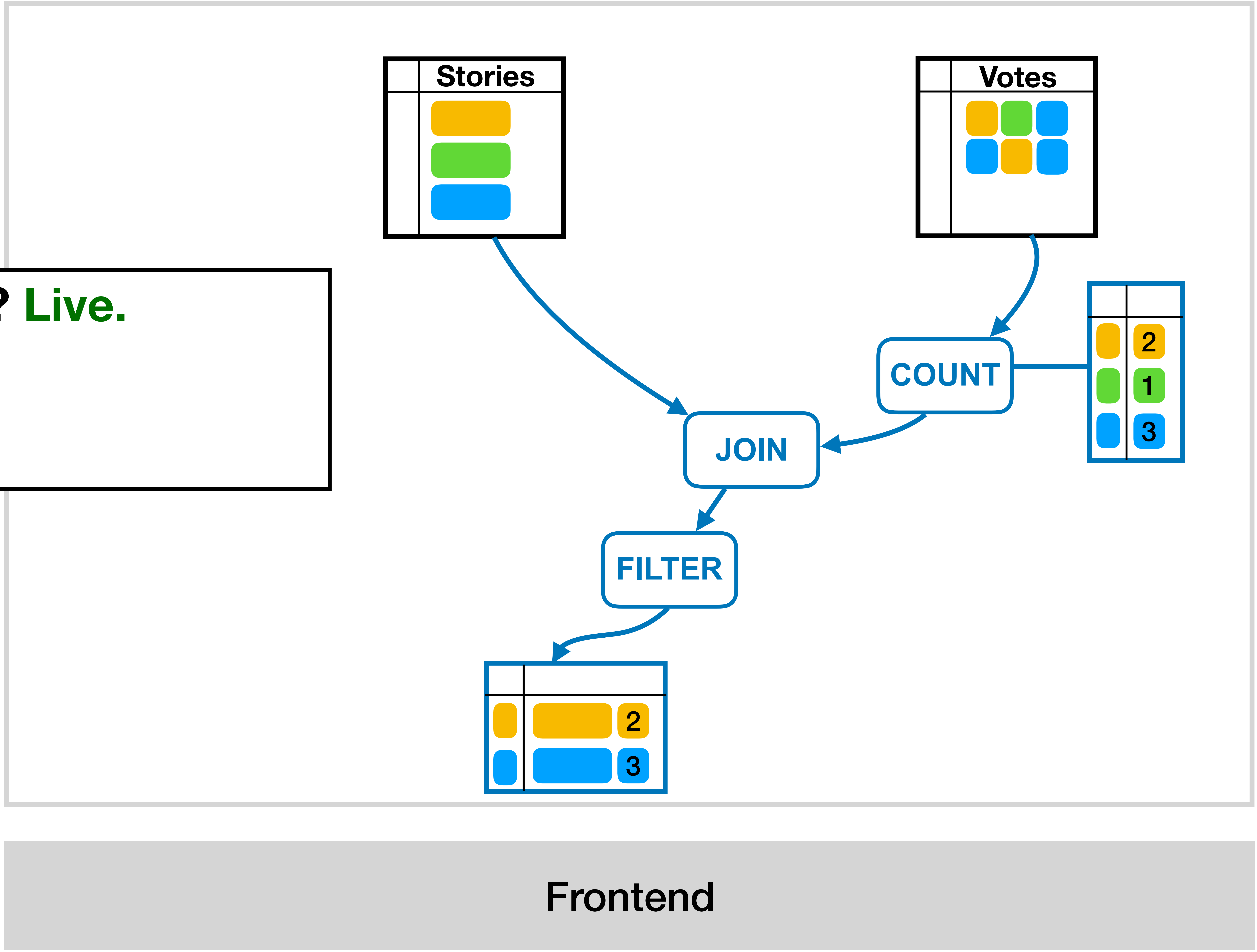


Noria



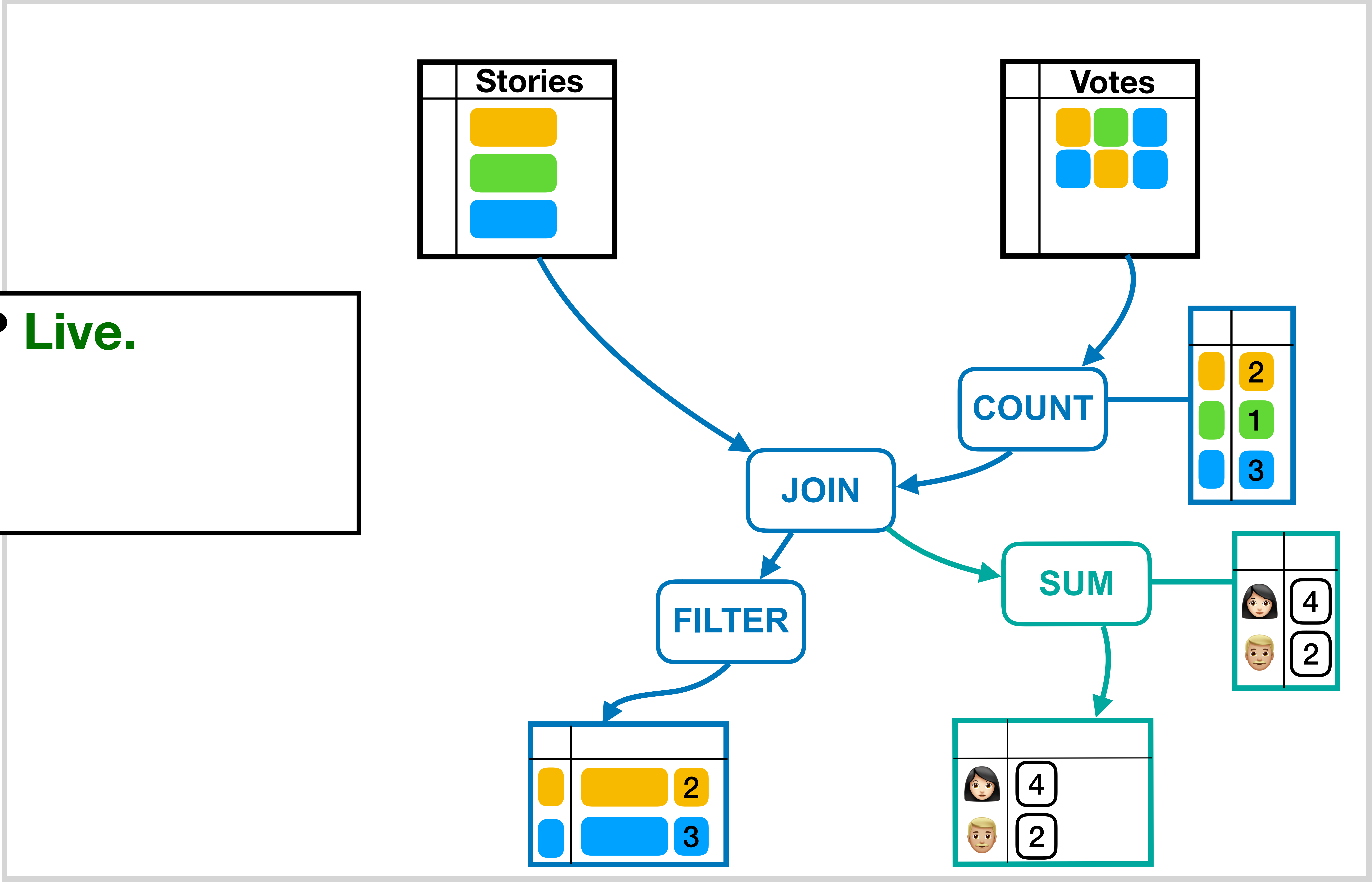
Noria

- **Change queries? Live.**



Noria

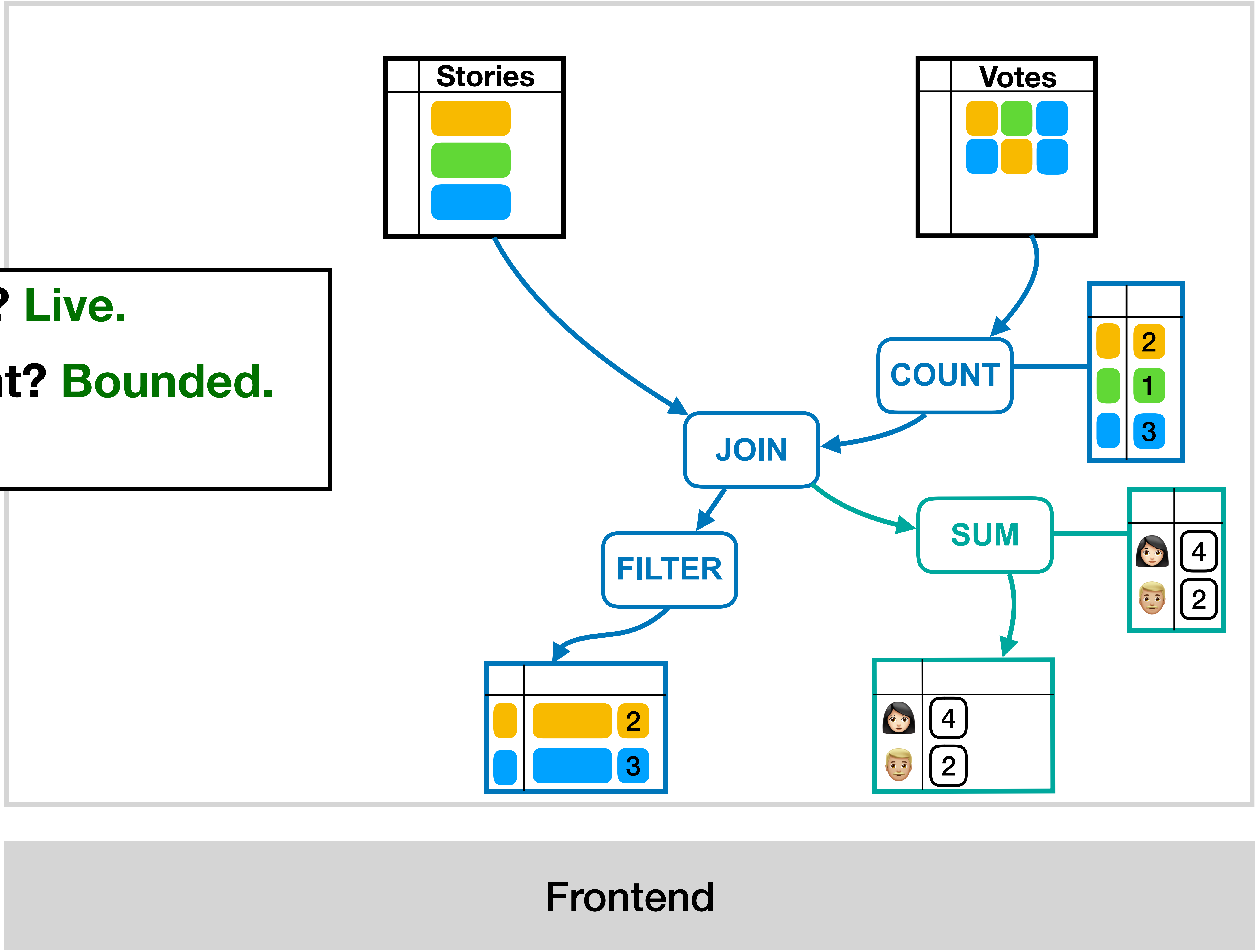
- **Change queries? Live.**



Frontend

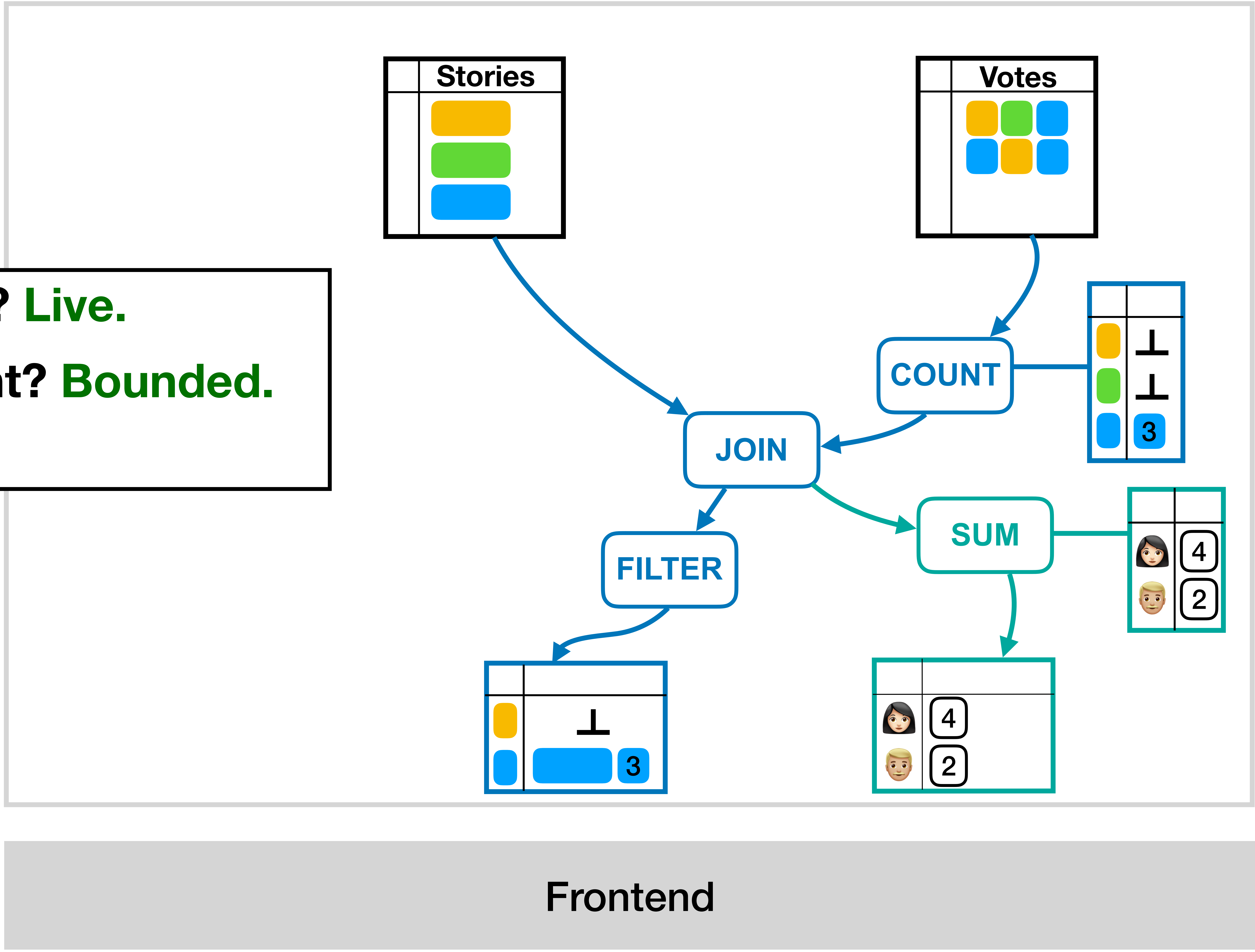
Noria

- Change queries? **Live.**
- Memory footprint? **Bounded.**



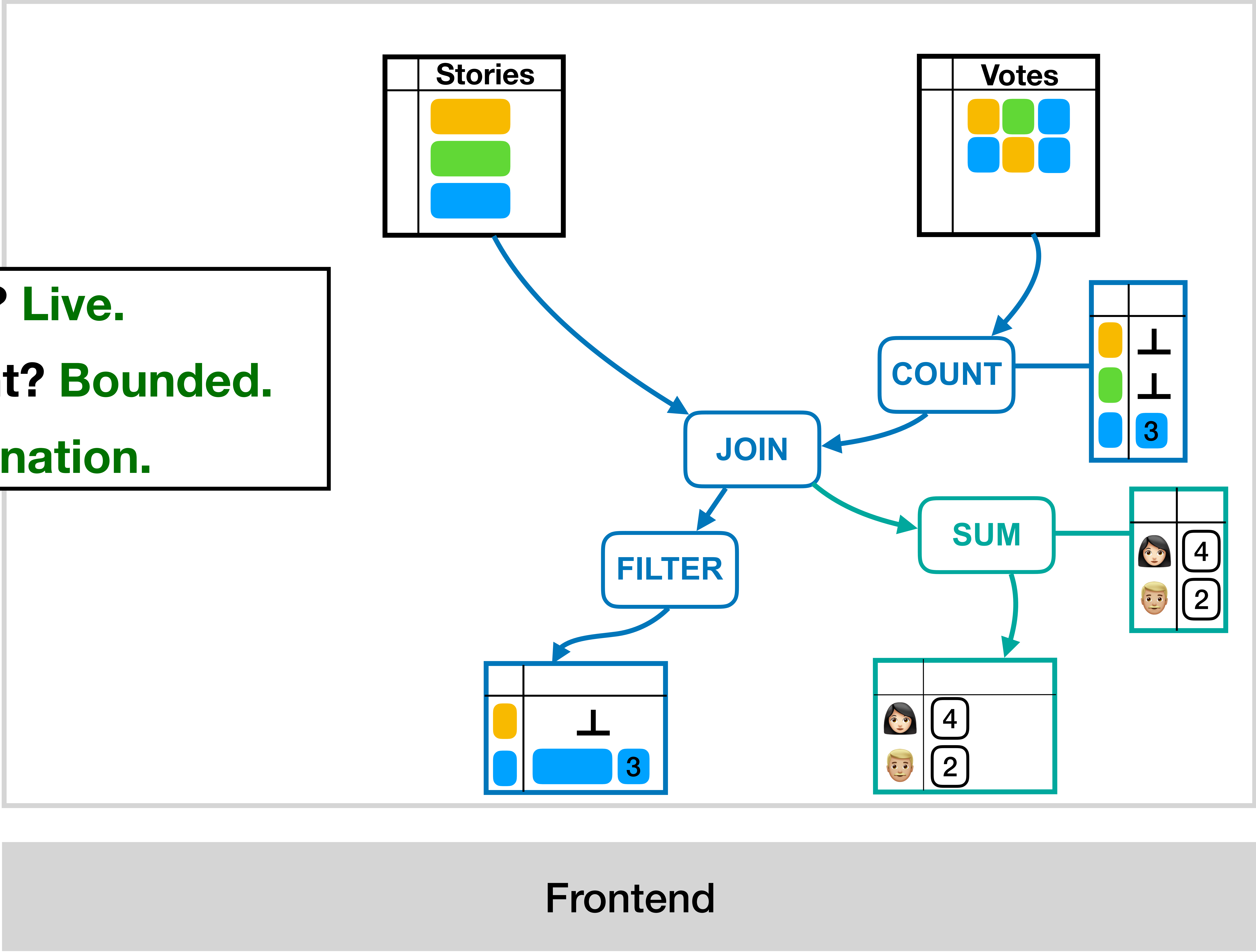
Noria

- Change queries? Live.
- Memory footprint? Bounded.



Noria

- Change queries? **Live.**
- Memory footprint? **Bounded.**
- **No global coordination.**

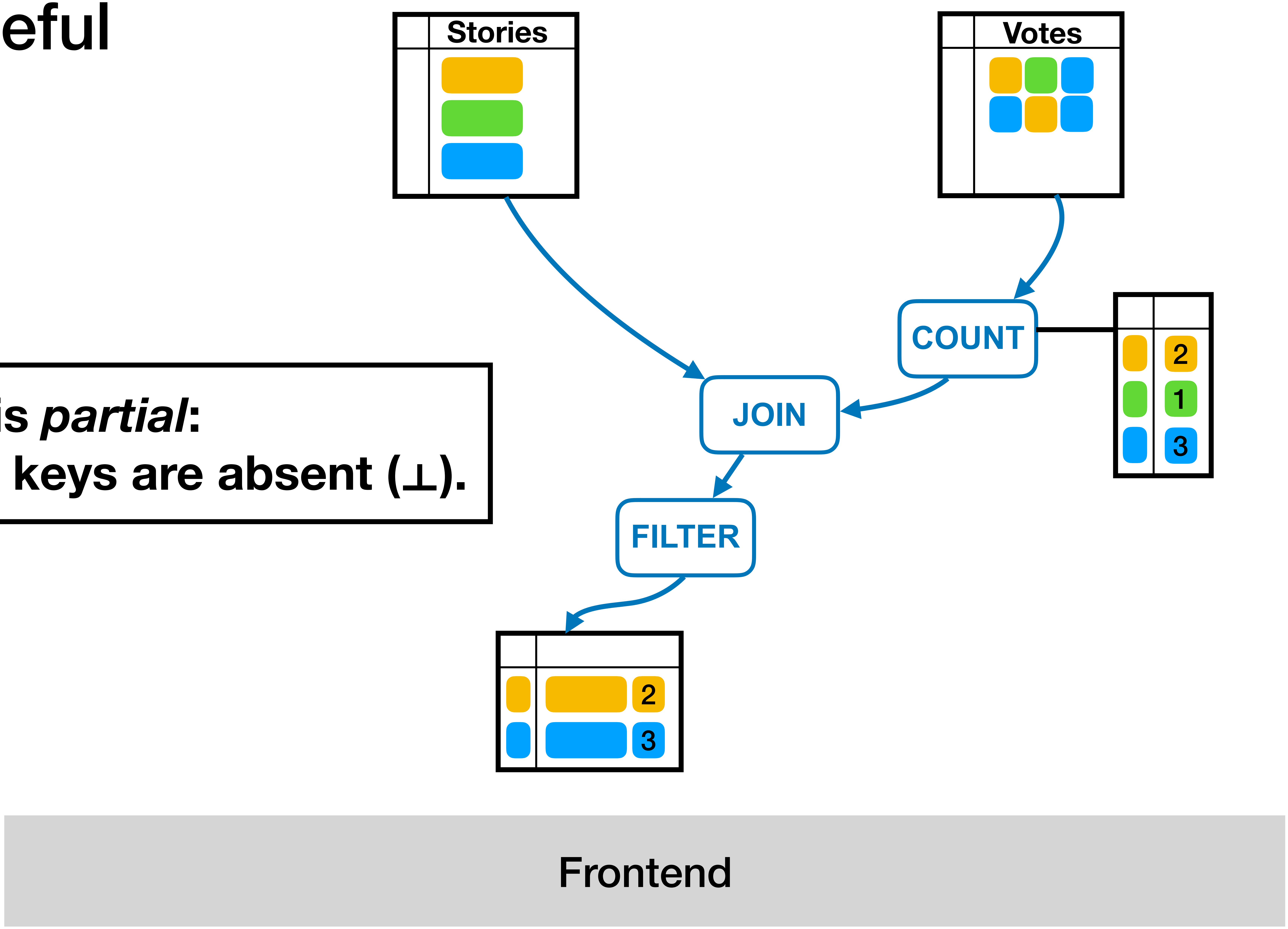


New model:

Partially-stateful data-flow

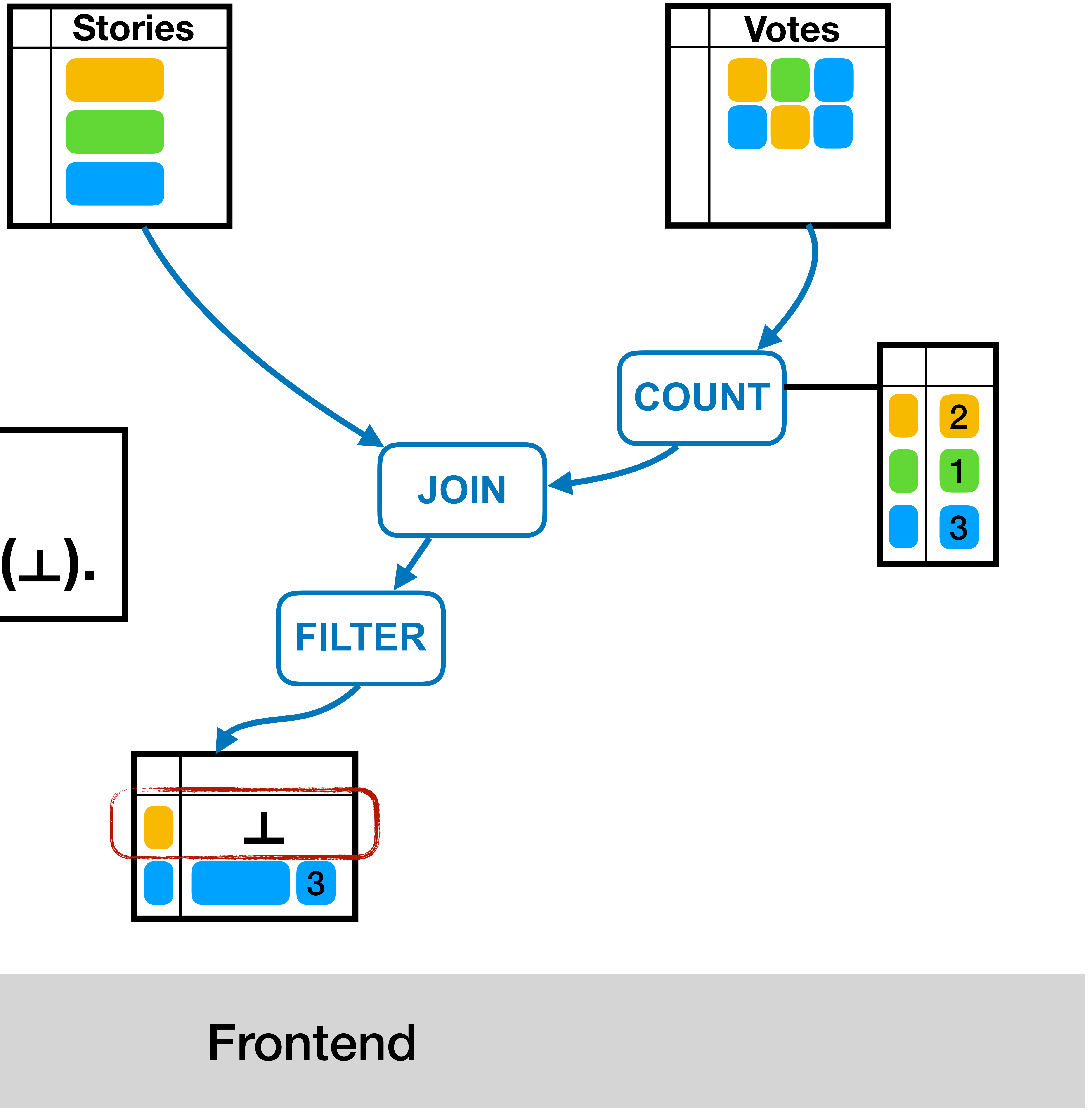
Partially-stateful data-flow

**Data-flow state is *partial*:
entries for some keys are absent (⊥).**



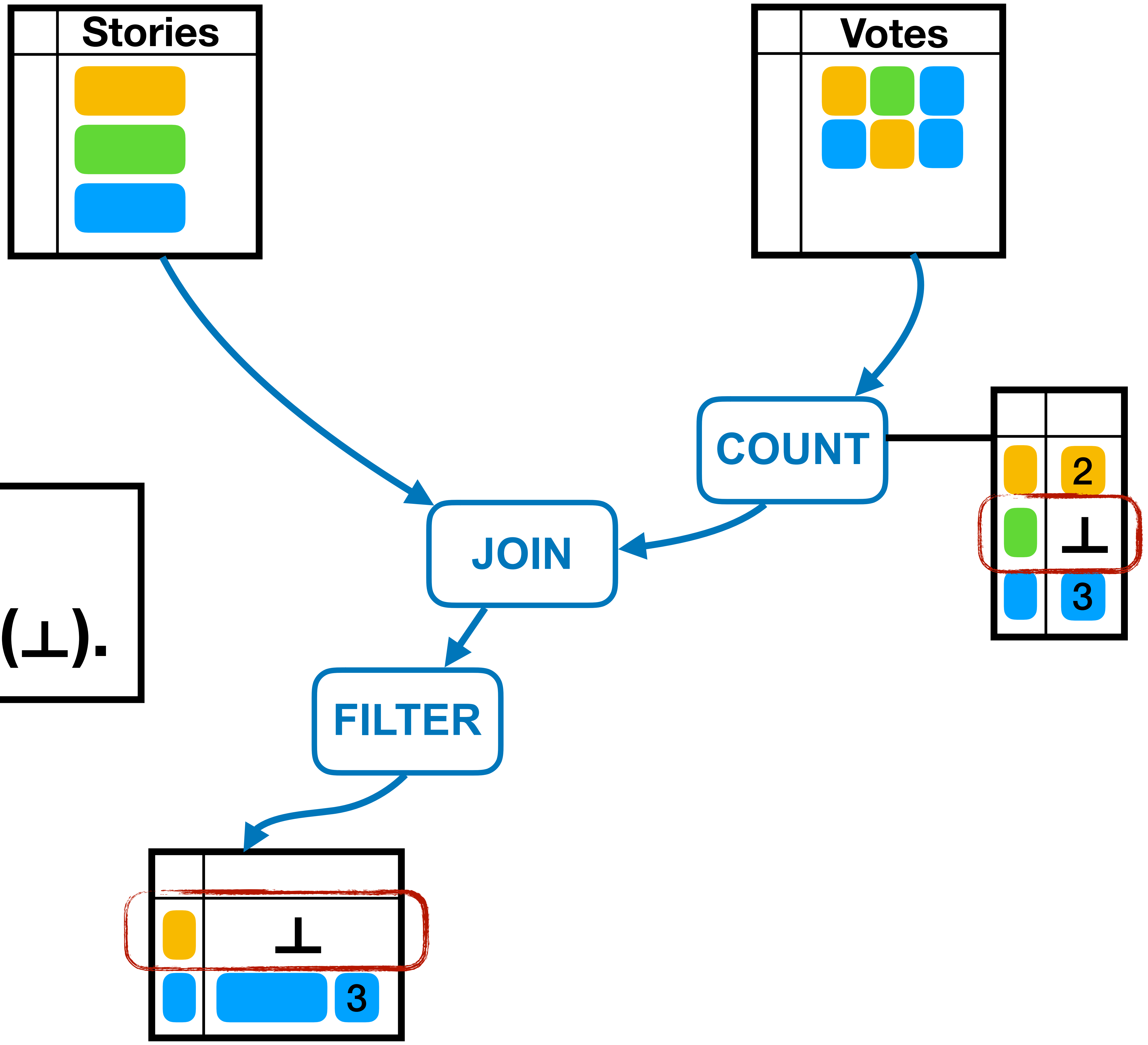
Partially-stateful data-flow

Data-flow state is *partial*:
entries for some keys are absent (⊥).



Partially-stateful data-flow

Data-flow state is *partial*:
entries for some keys are absent (⊥).

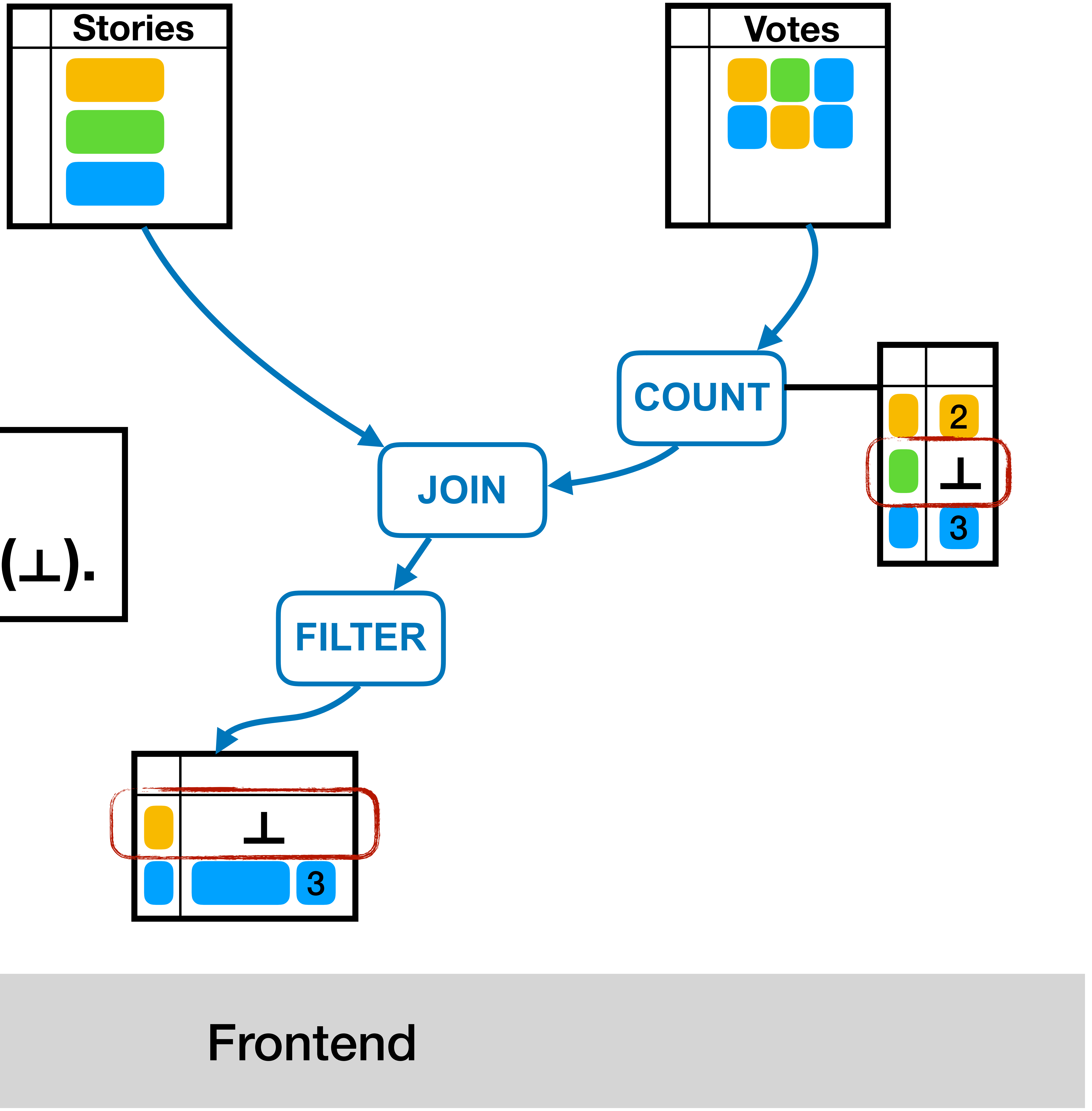


Frontend

Partially-stateful data-flow

Data-flow state is *partial*:
entries for some keys are absent (⊥).

Lower memory footprint.

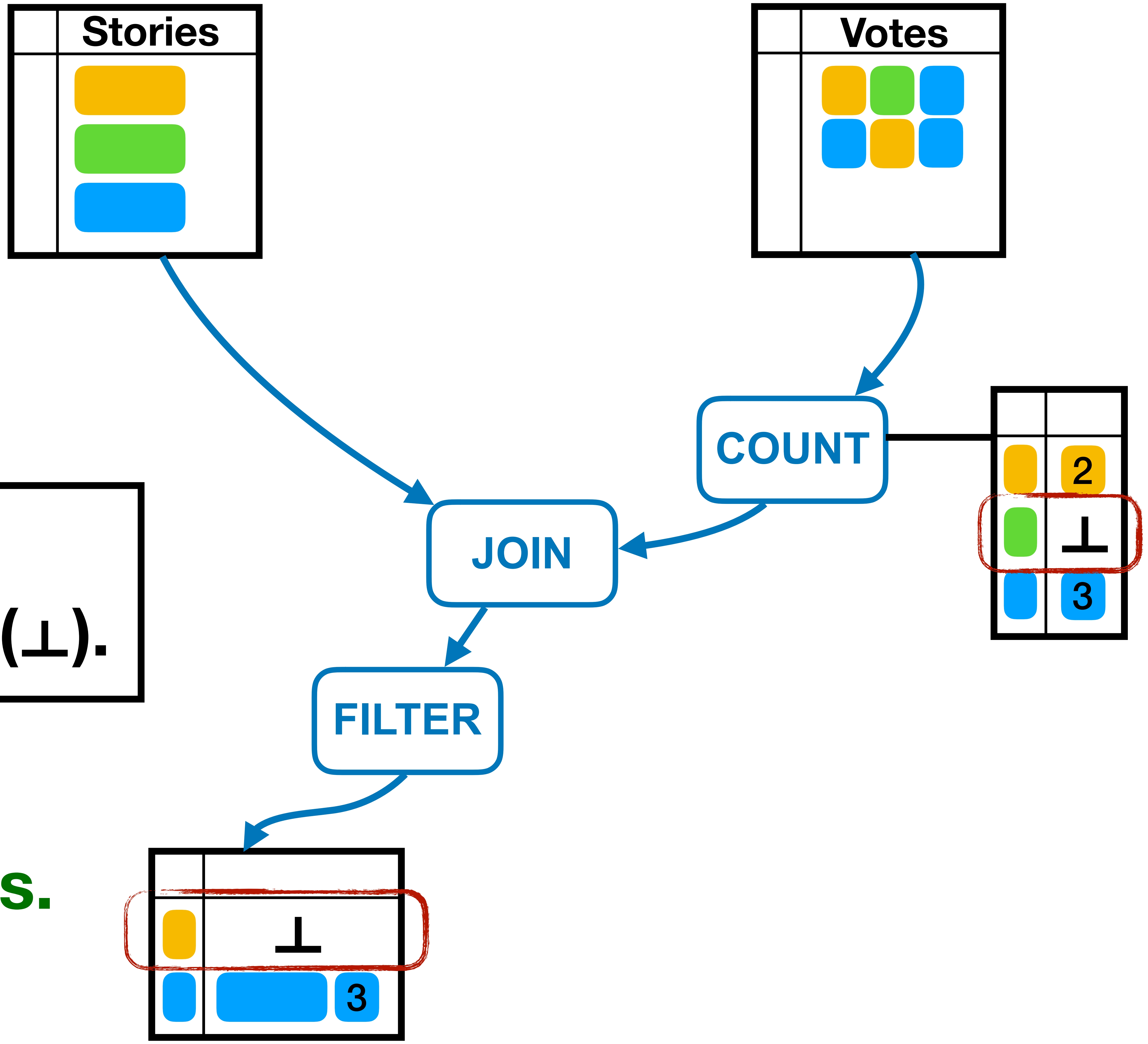


Partially-stateful data-flow

Data-flow state is *partial*:
entries for some keys are absent (⊥).

Lower memory footprint.

No need to update absent entries.



Frontend

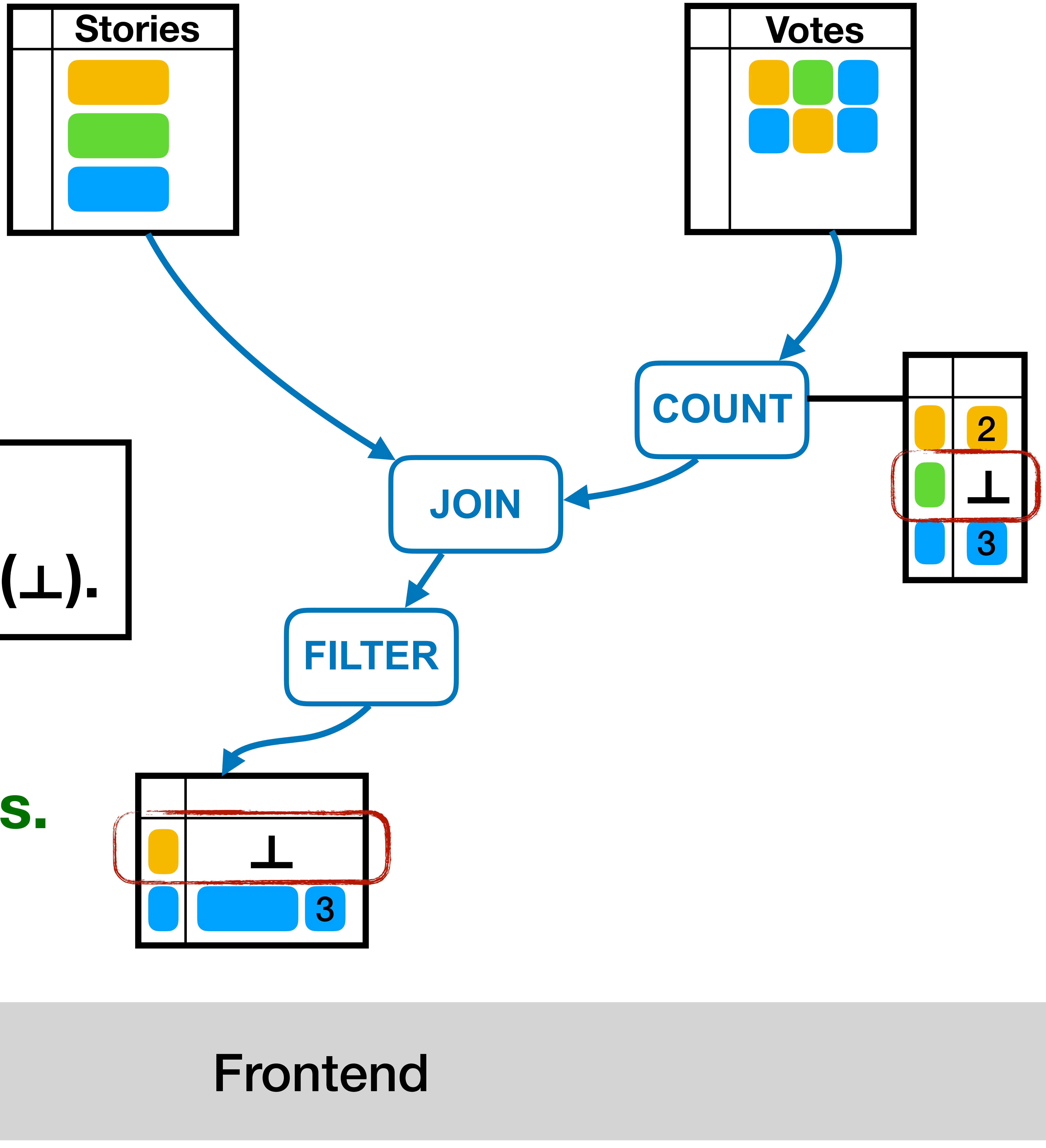
Partially-stateful data-flow

Data-flow state is *partial*:
entries for some keys are absent (⊥).

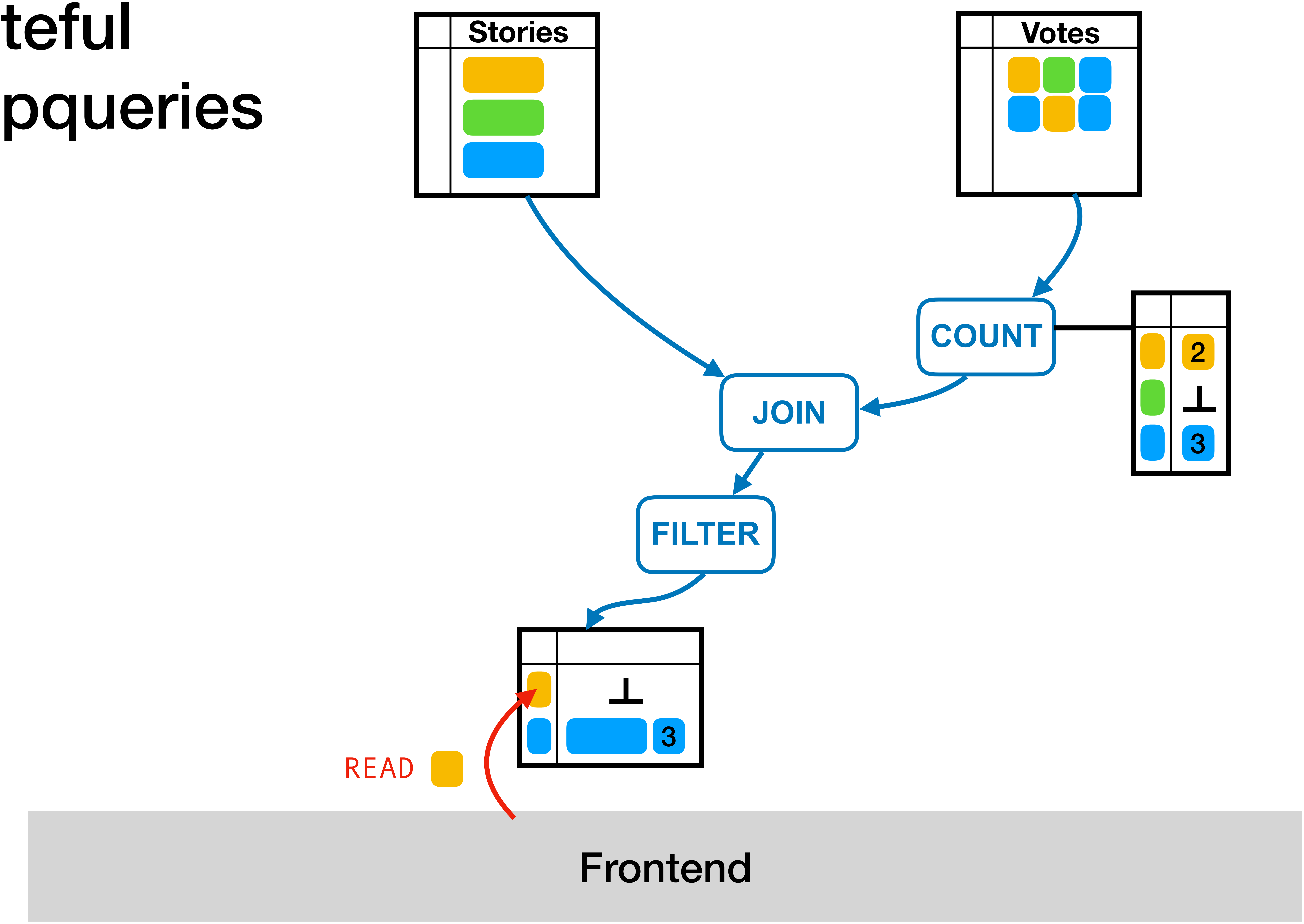
Lower memory footprint.

No need to update absent entries.

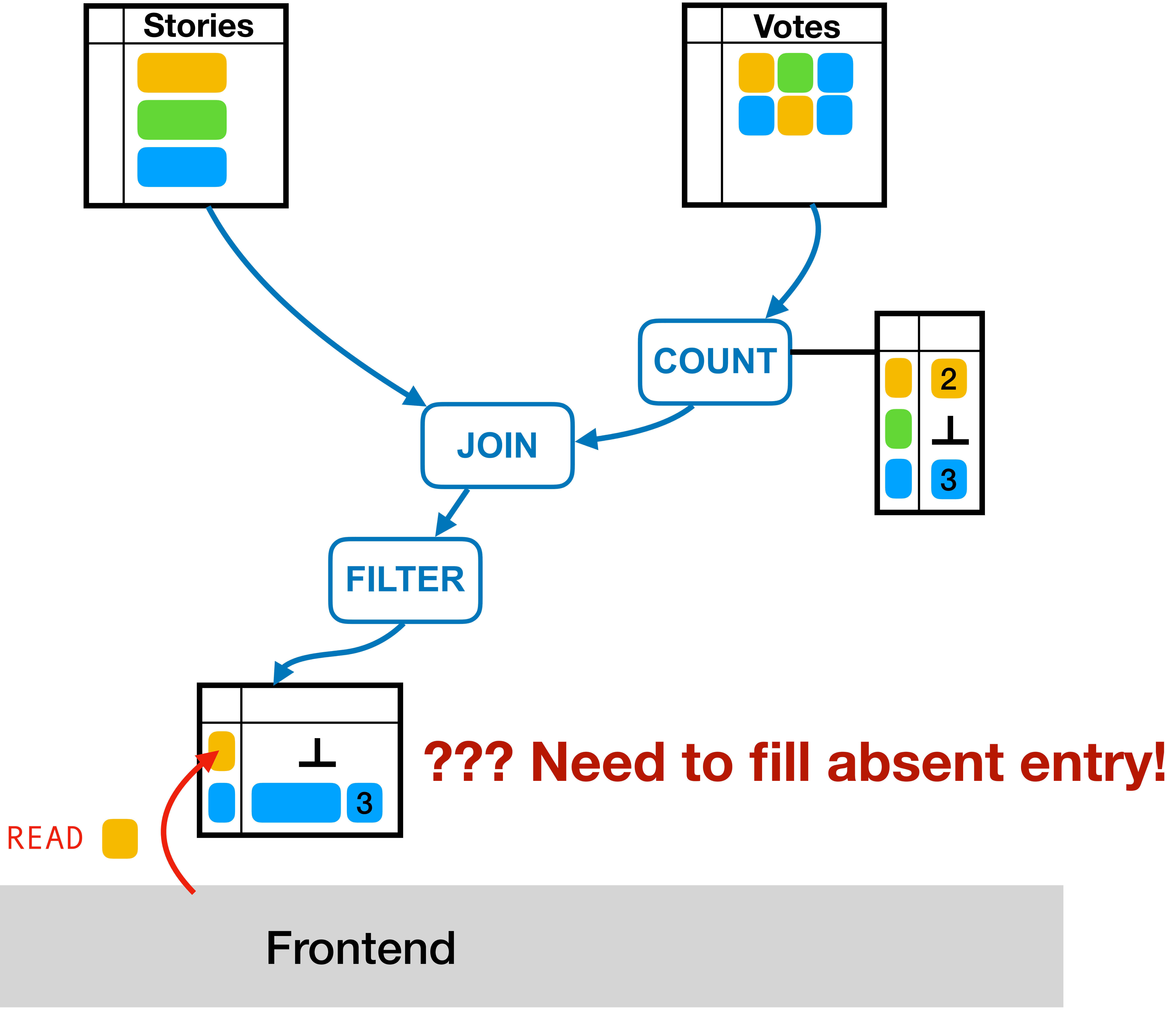
Enables live data-flow changes.



Partially-stateful data-flow: upqueries

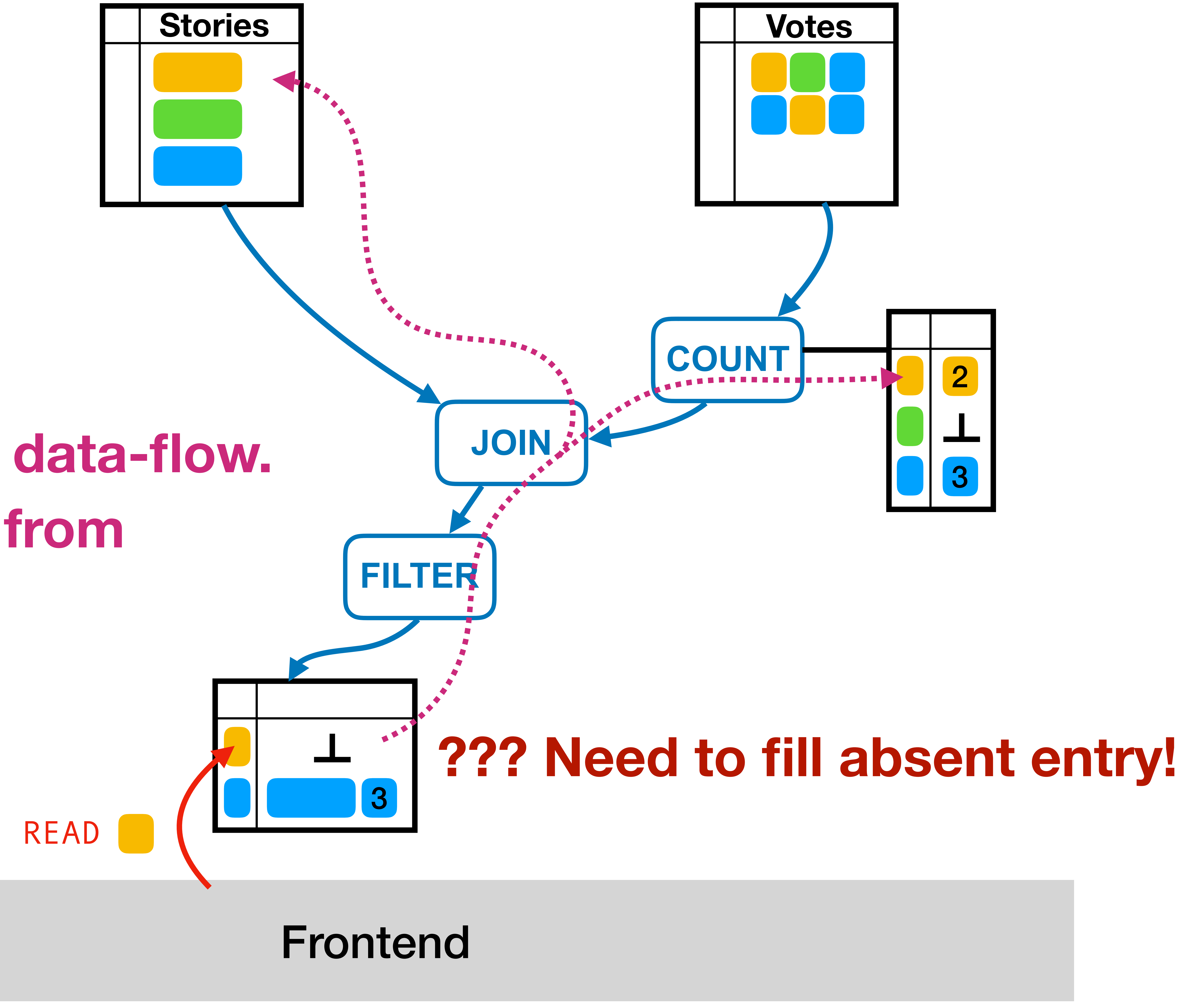


Partially-stateful data-flow: upqueries



??? Need to fill absent entry!

Partially-stateful data-flow: upqueries



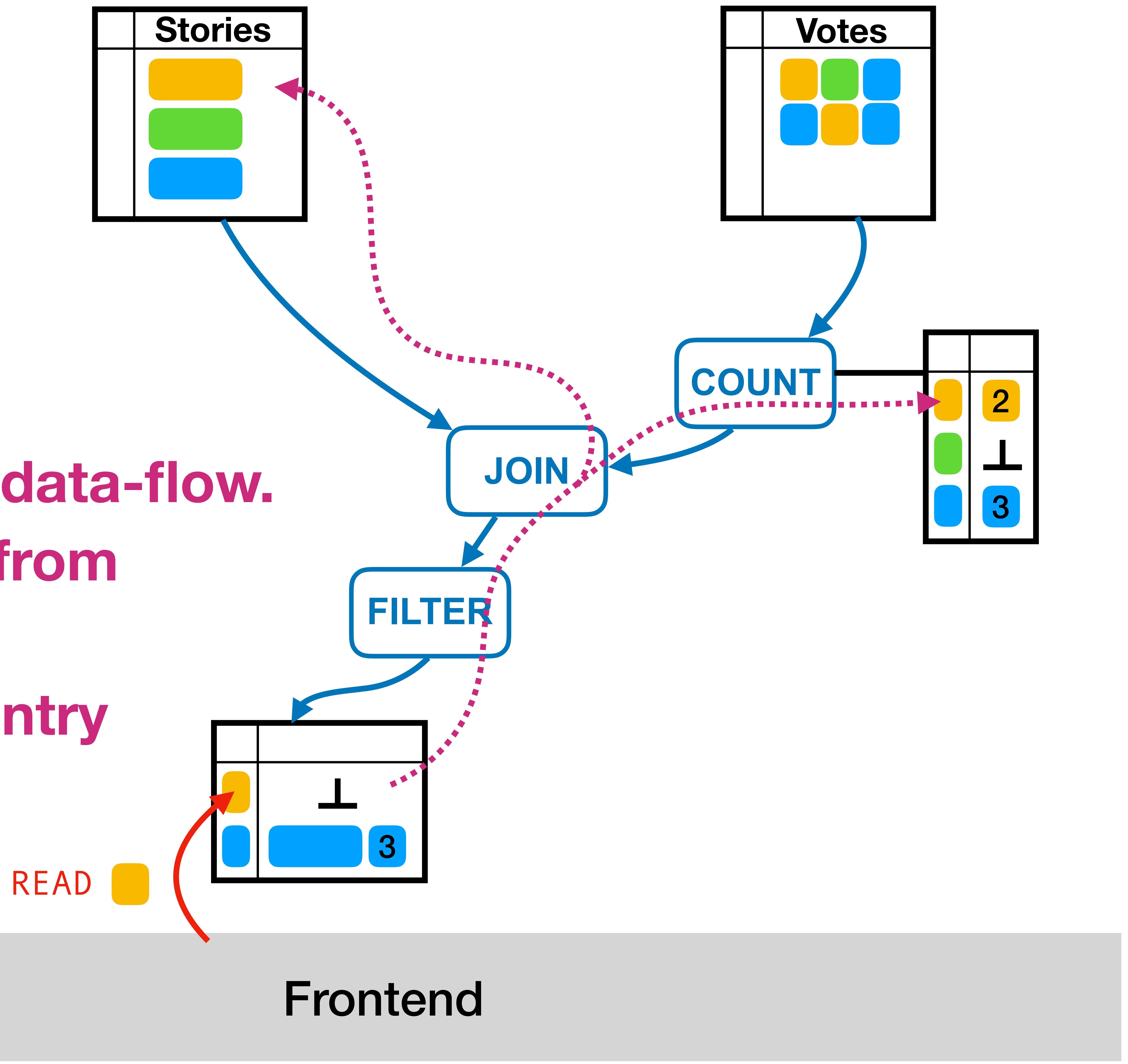
Solution: *upquery* through data-flow.

- **Compute missing entry from upstream state**

Partially-stateful data-flow: upqueries

Solution: *upquery* through data-flow.

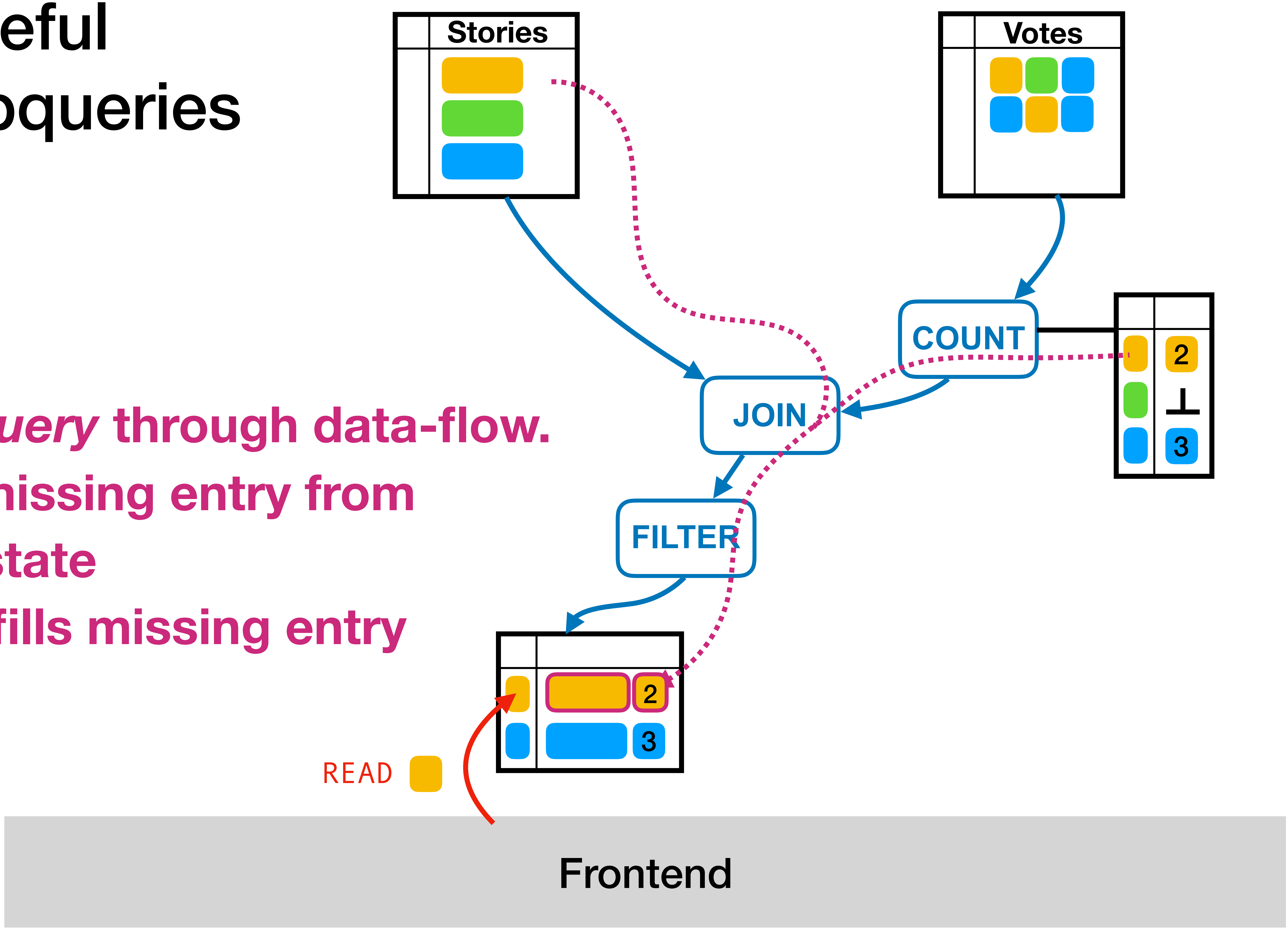
- Compute missing entry from upstream state
- Response fills missing entry



Partially-stateful data-flow: upqueries

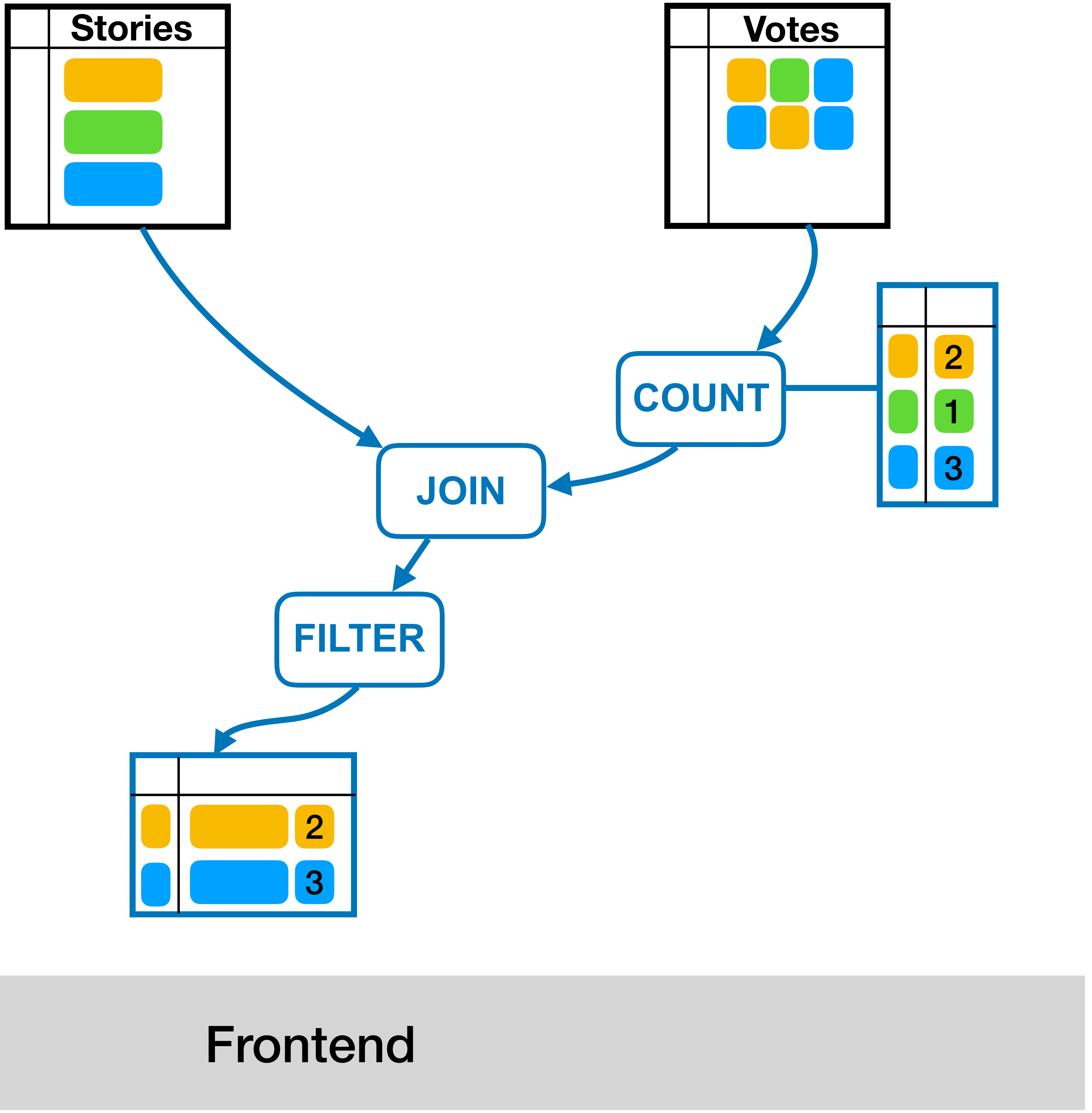
Solution: *upquery* through data-flow.

- Compute missing entry from upstream state
- Response fills missing entry



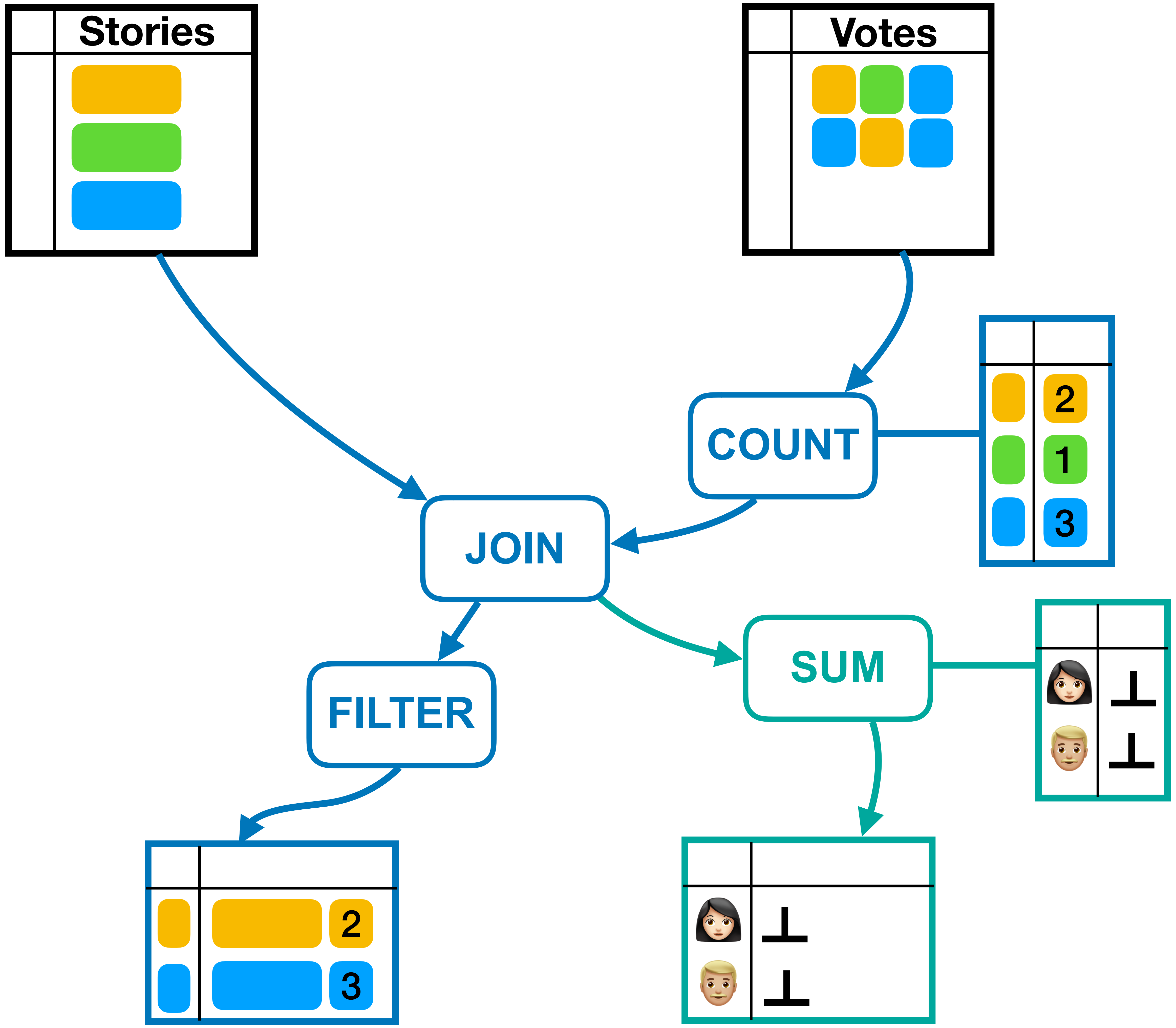
Partial state enables live data-flow changes

Start new views and operator state **empty**, **fill via** upqueries.



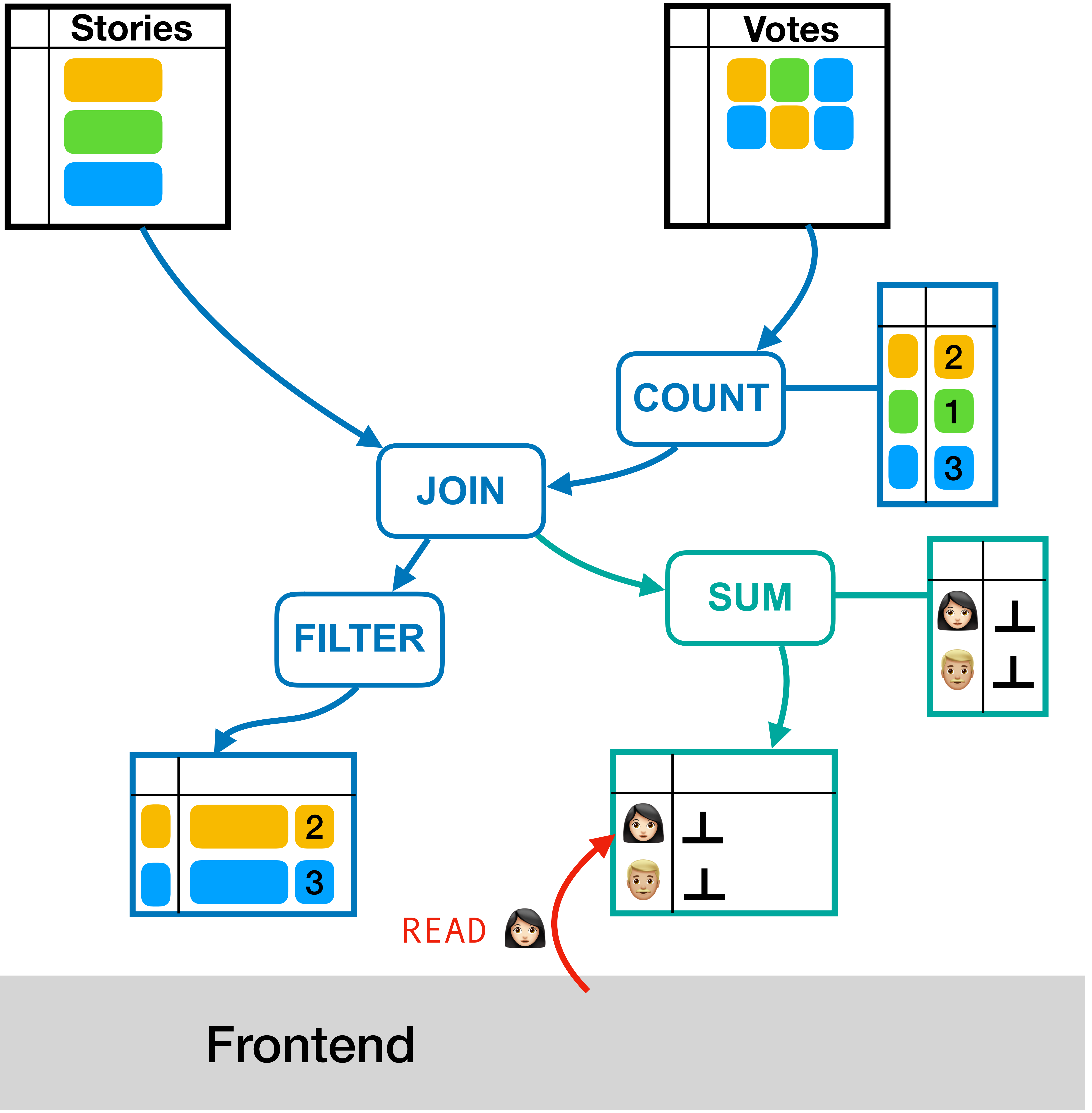
Partial state enables live data-flow changes

Start new views and operator state **empty**, **fill via** upqueries.



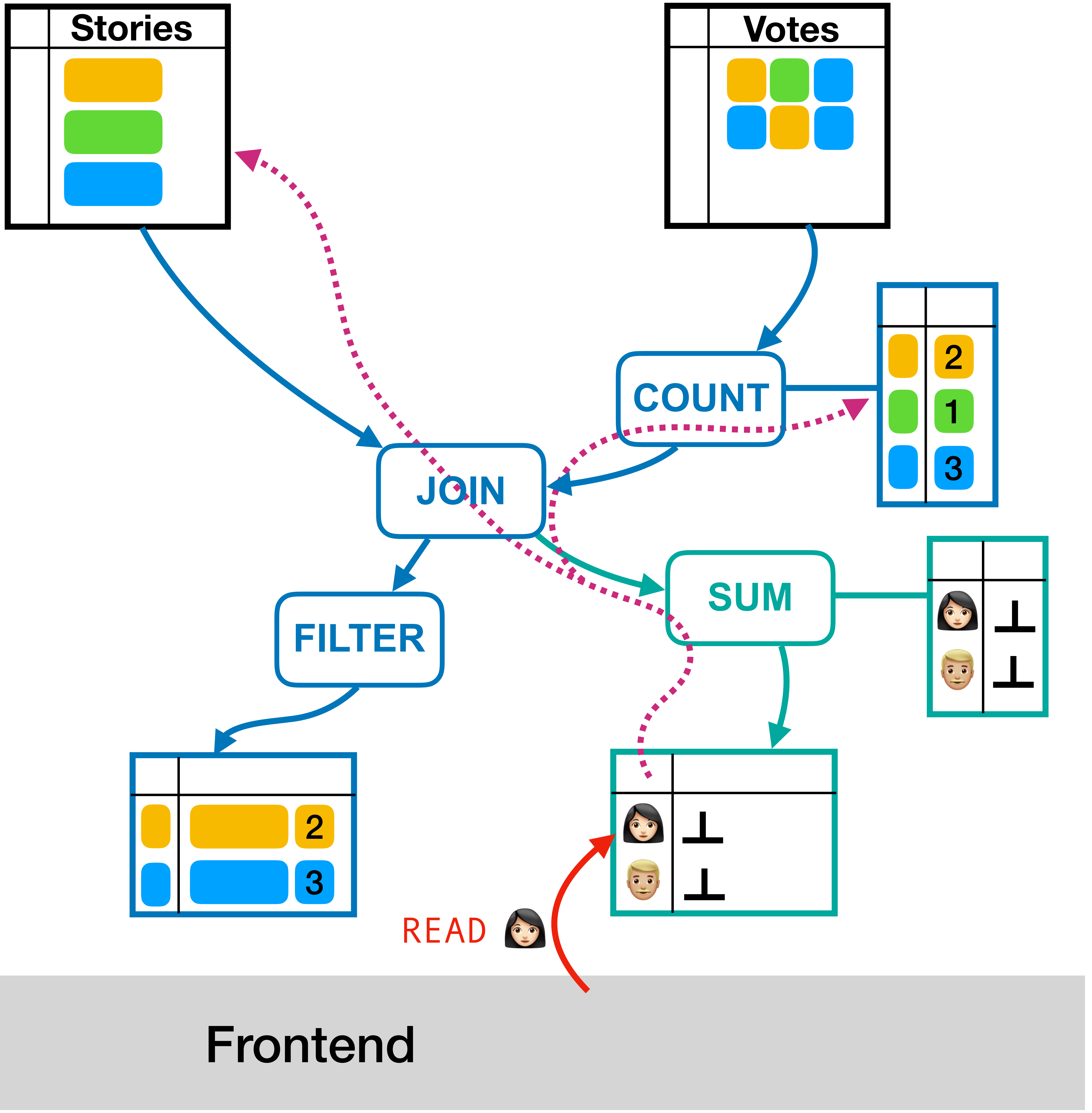
Partial state enables live data-flow changes

Start new views and operator state **empty**, **fill via** upqueries.



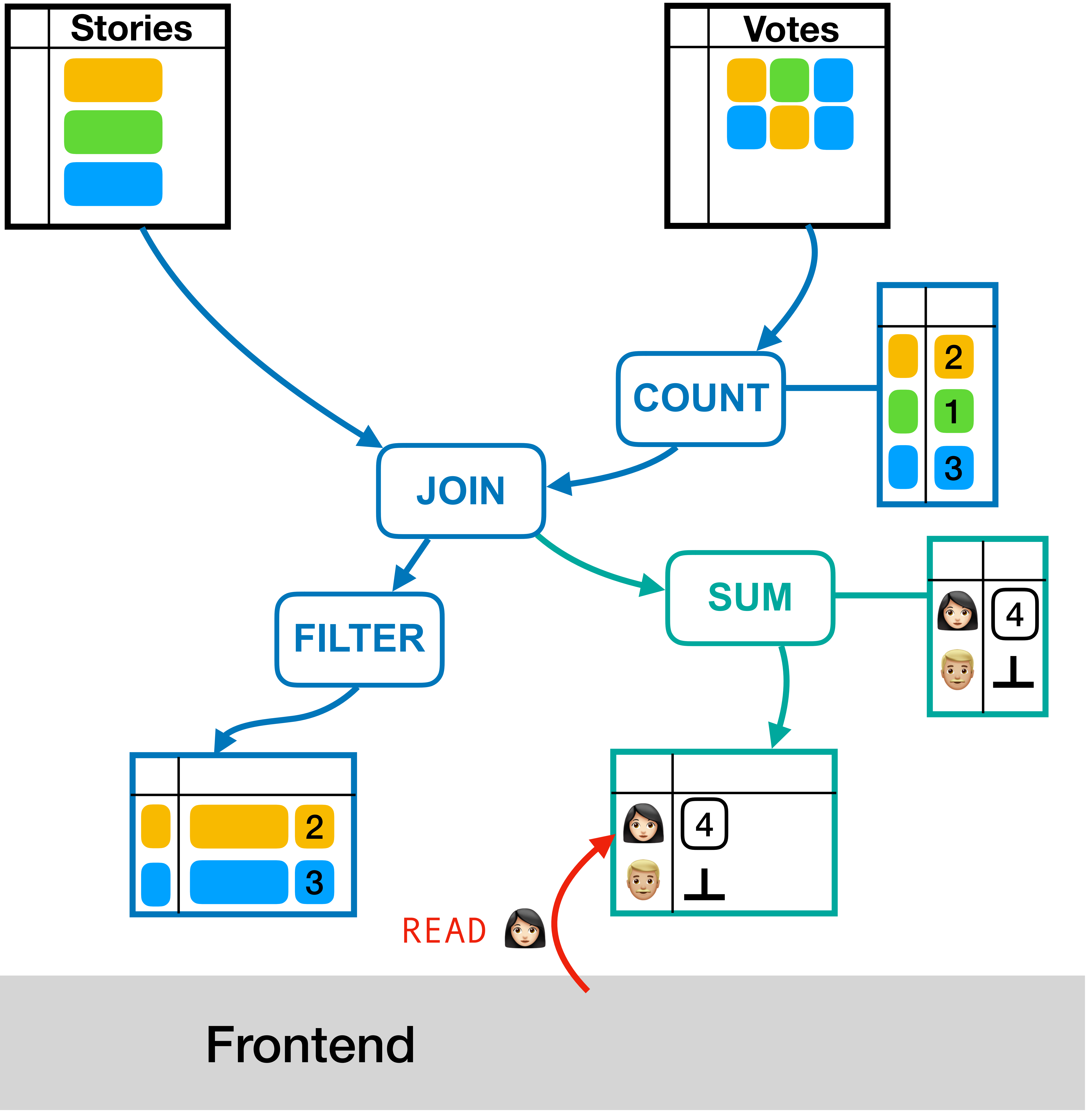
Partial state enables live data-flow changes

Start new views and operator state **empty**, **fill via upqueries**.

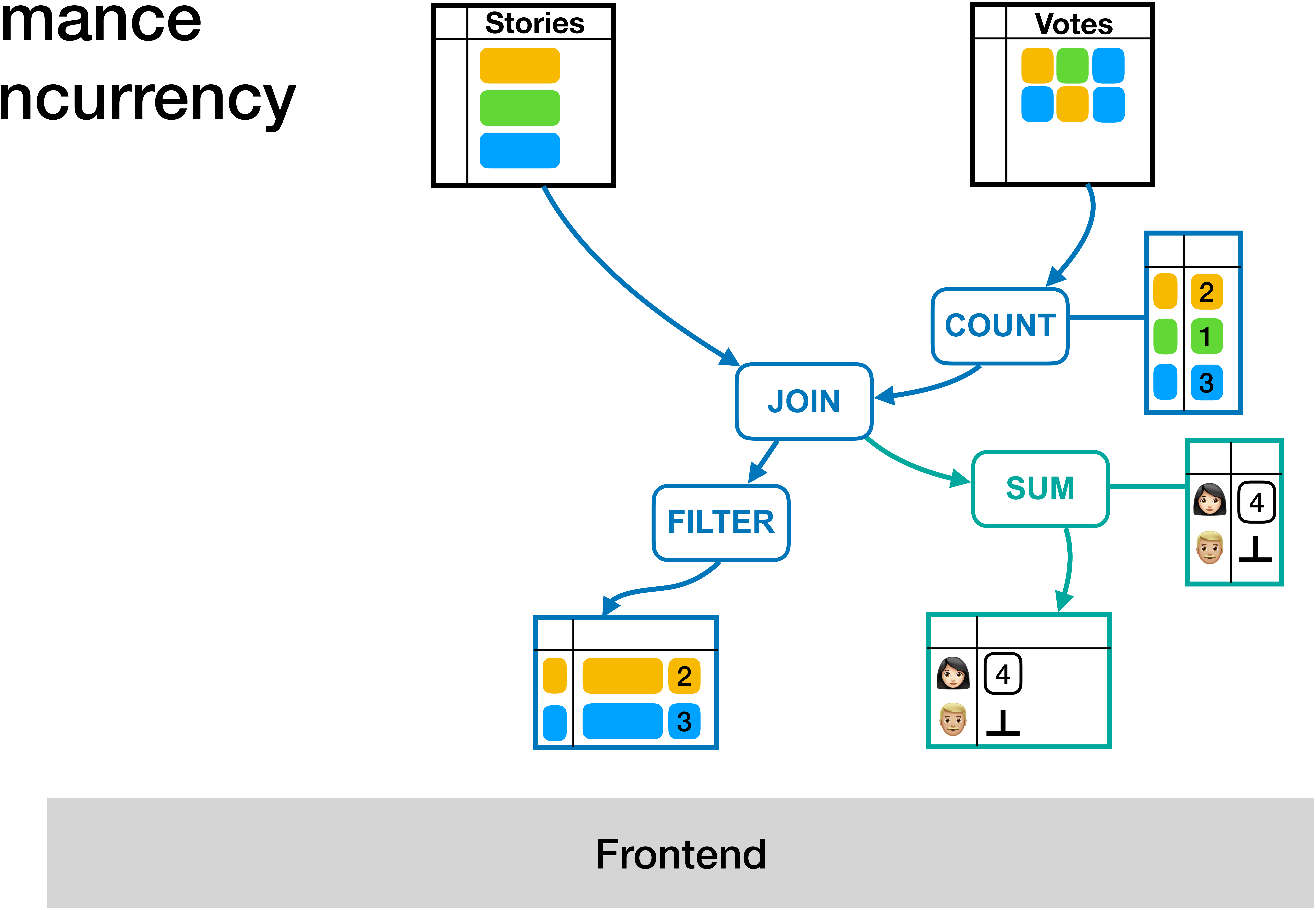


Partial state enables live data-flow changes

Start new views and operator state **empty**, **fill via upqueries**.



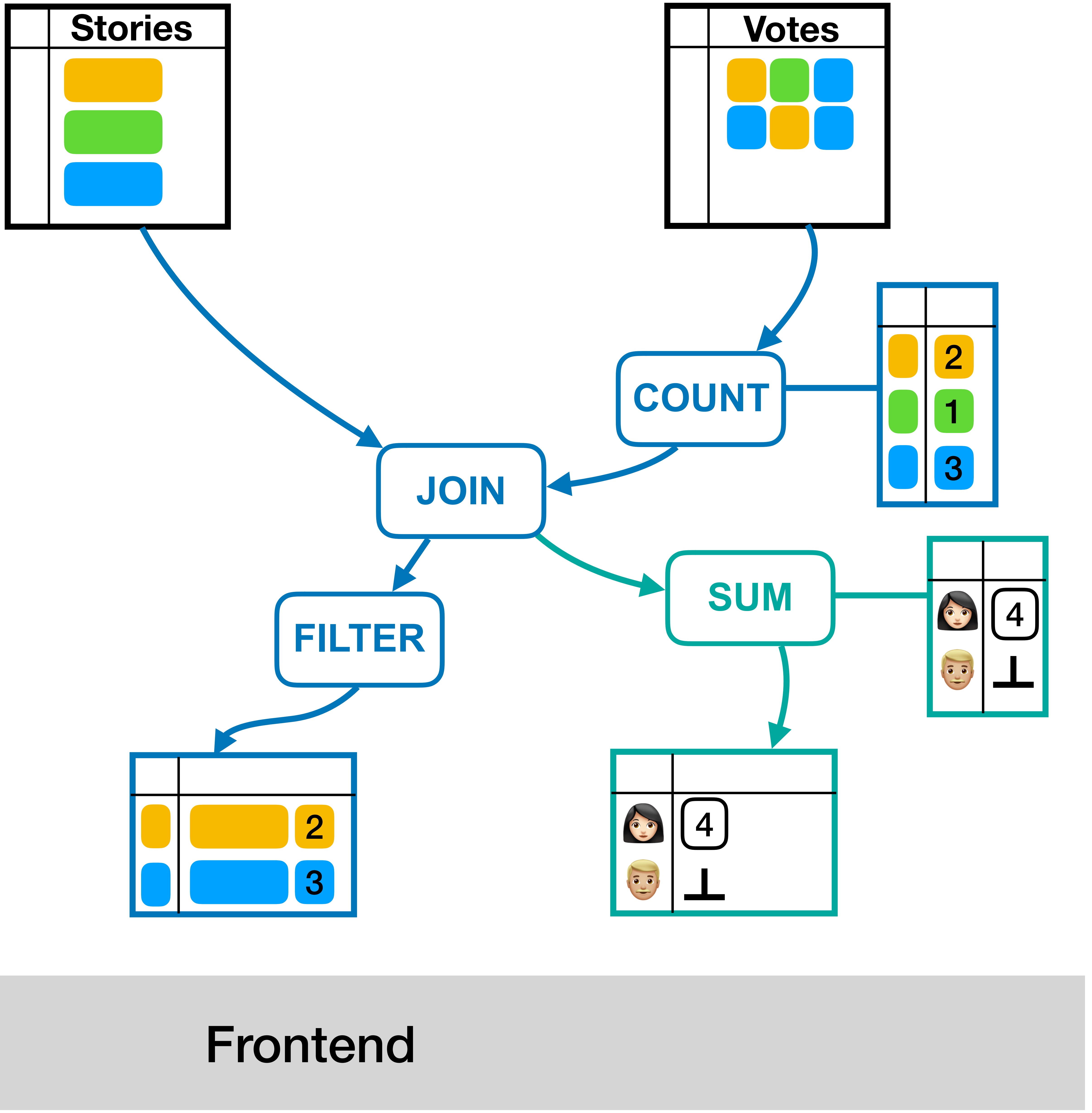
High performance requires concurrency



High performance requires concurrency

Process operators concurrently.
Read from views concurrently.
Process shards concurrently.

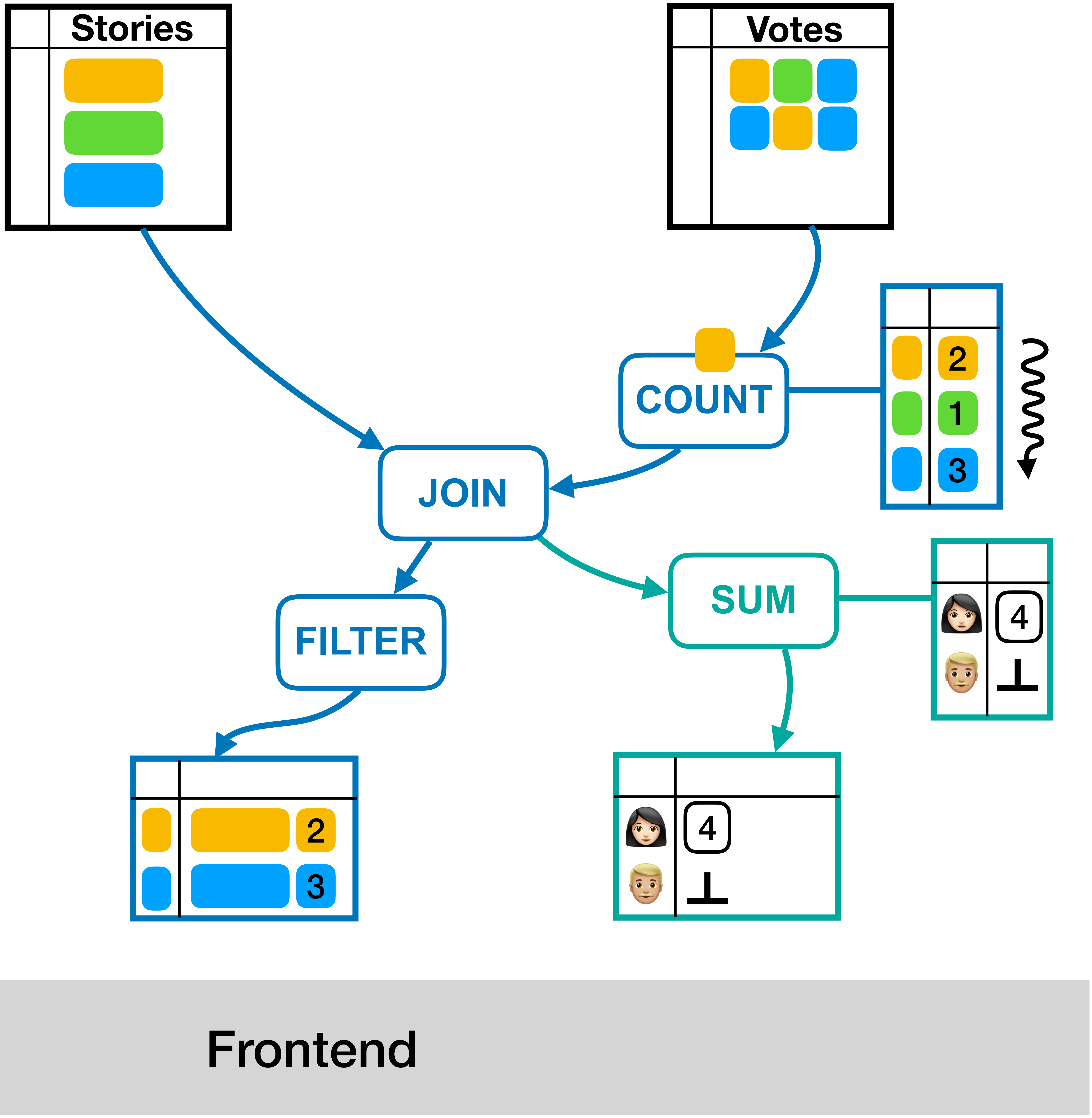
Without global coordination!



High performance requires concurrency

Process operators concurrently.
Read from views concurrently.
Process shards concurrently.

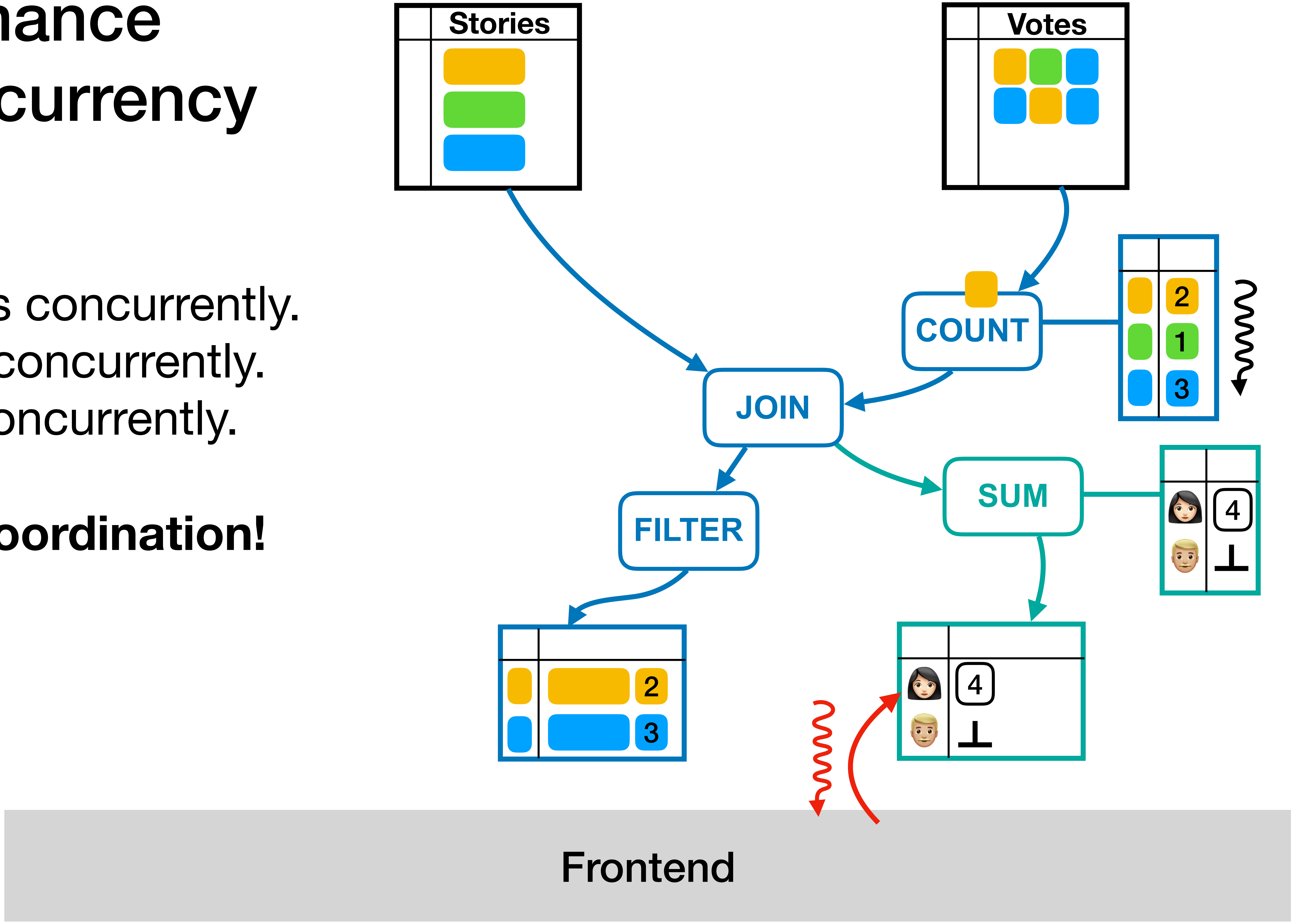
Without global coordination!



High performance requires concurrency

Process operators concurrently.
Read from views concurrently.
Process shards concurrently.

Without global coordination!



Challenges implementing partially-stateful data-flow

Challenges implementing partially-stateful data-flow

1. Concurrent upqueries and forward processing — races!

Must maintain **correctness** under concurrency!

Challenges implementing partially-stateful data-flow

1. Concurrent upqueries and forward processing — races!

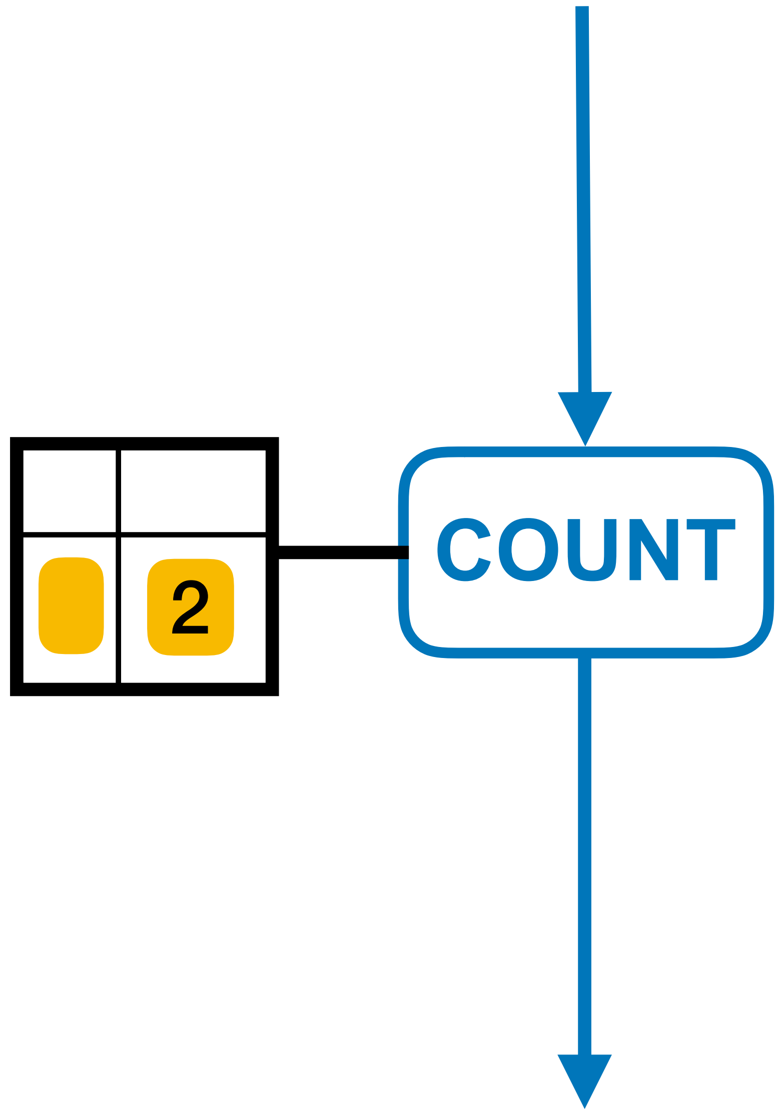
Must maintain **correctness** under concurrency!

Correctness under concurrency

Goal: upquery restores state as if present all along.

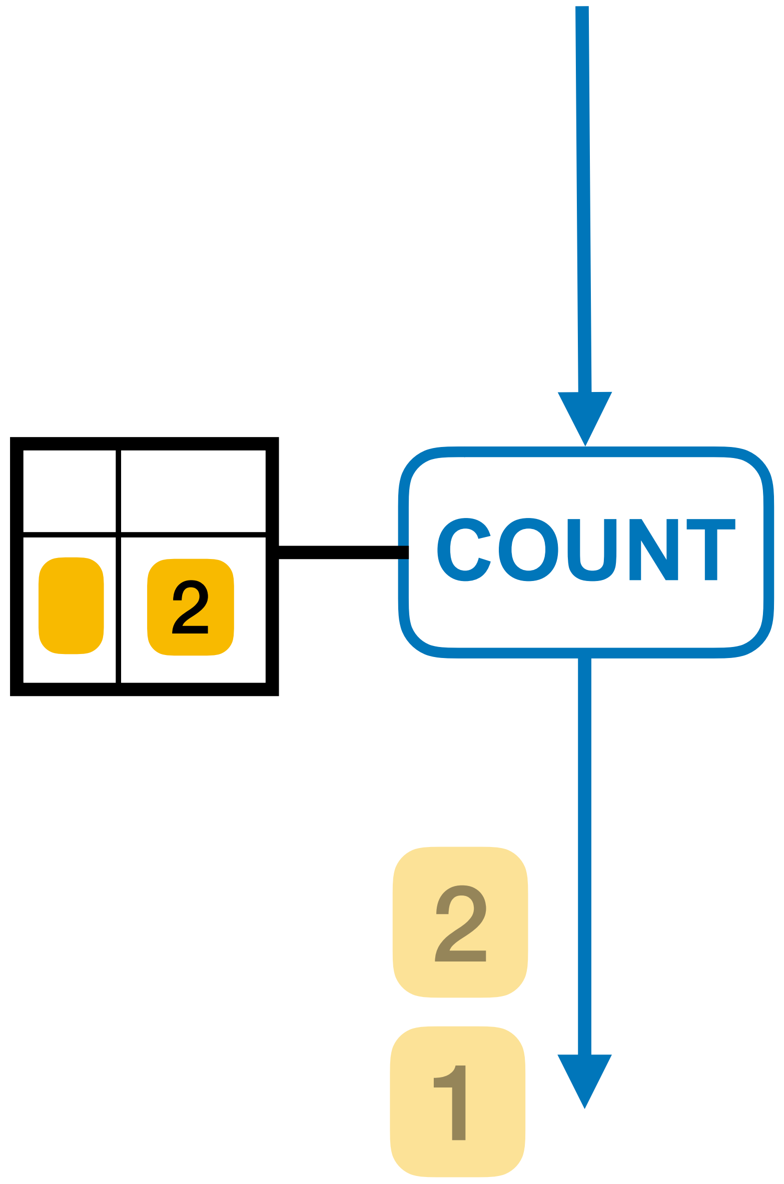
Correctness under concurrency

Goal: upquery restores state as if present all along.



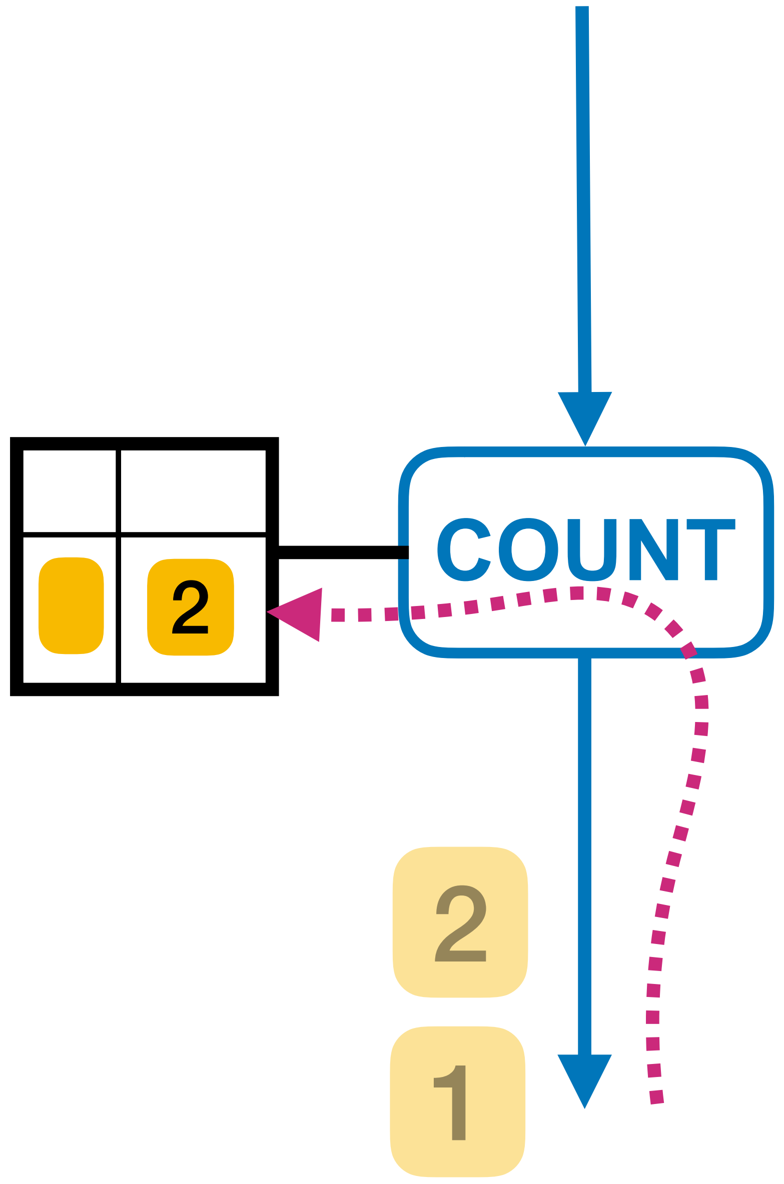
Correctness under concurrency

Goal: upquery restores state as if present all along.



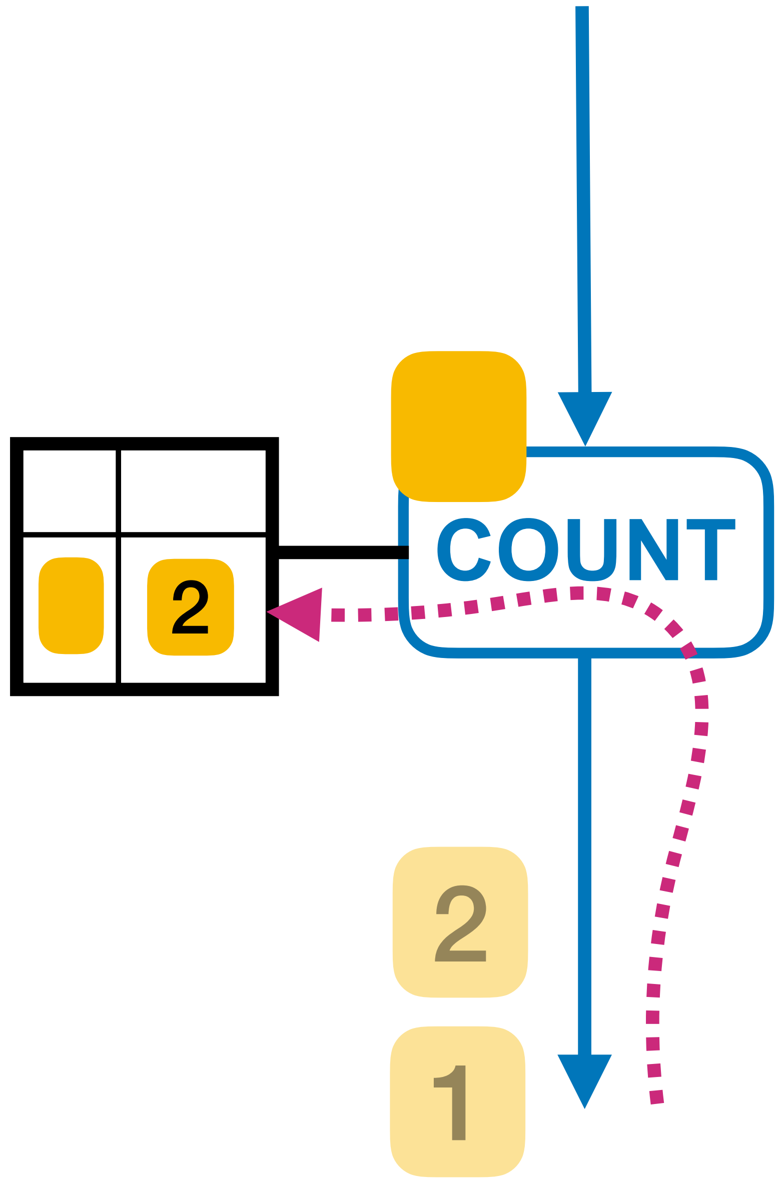
Correctness under concurrency

Goal: upquery restores state as if present all along.



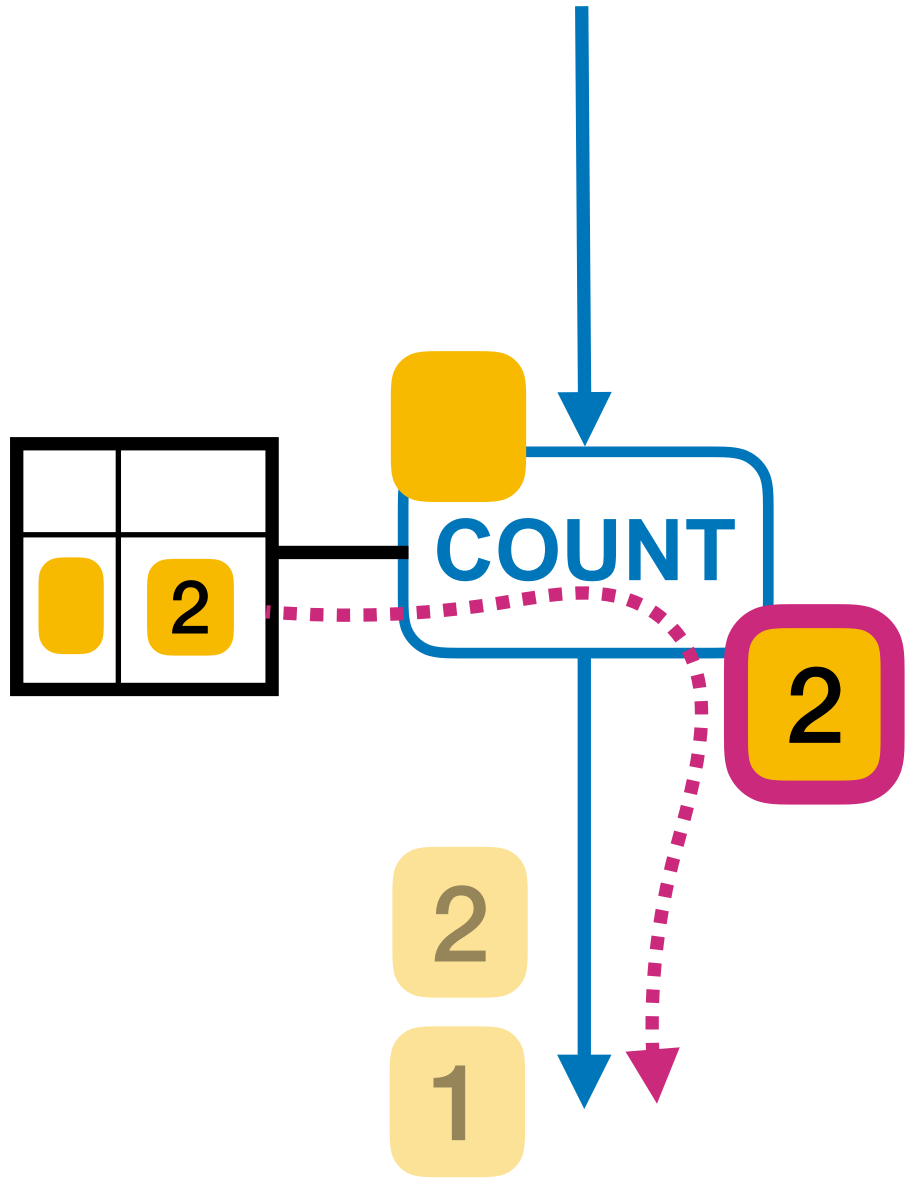
Correctness under concurrency

Goal: upquery restores state as if present all along.



Correctness under concurrency

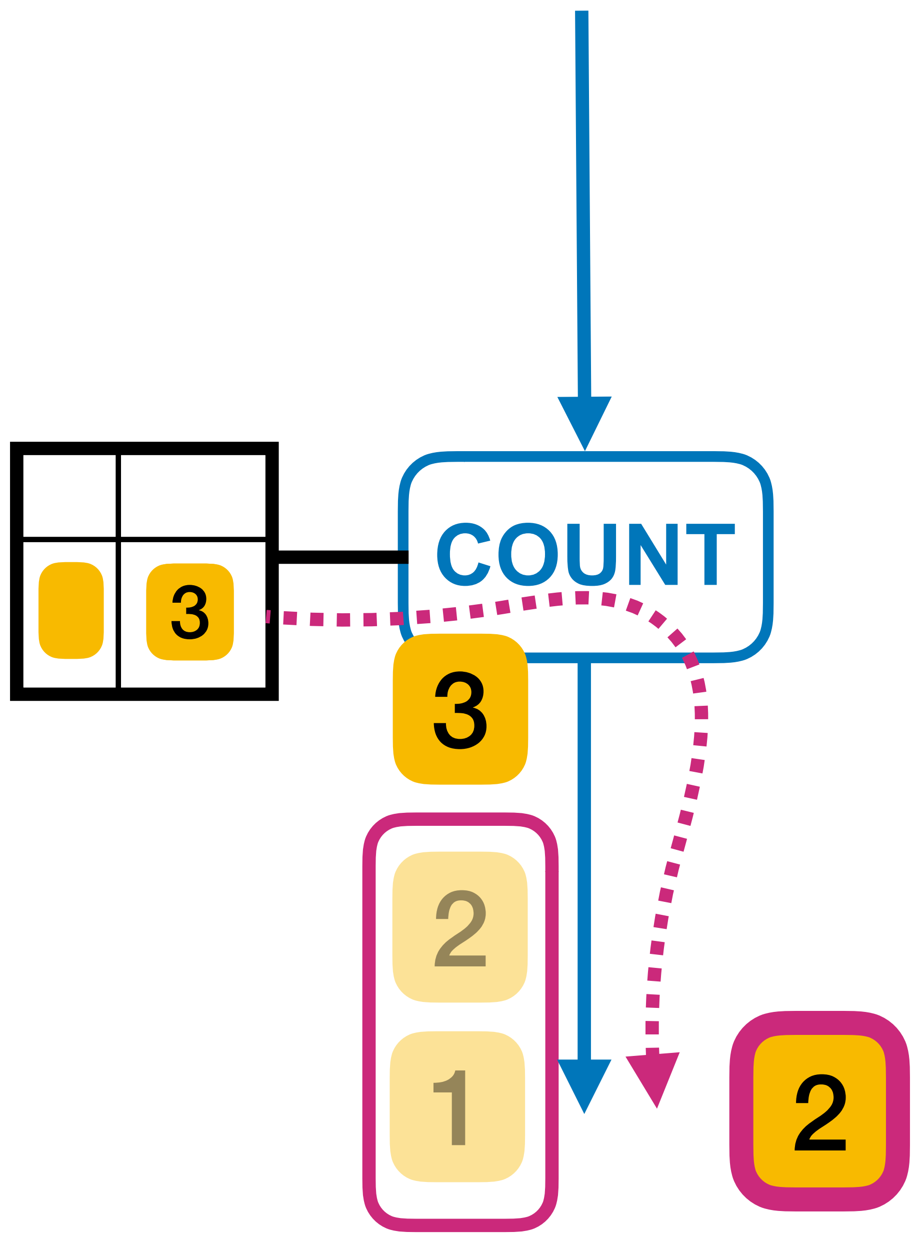
Goal: upquery restores state as if present all along.



Upquery response is a **snapshot** of state

Correctness under concurrency

Goal: upquery restores state as if present all along.



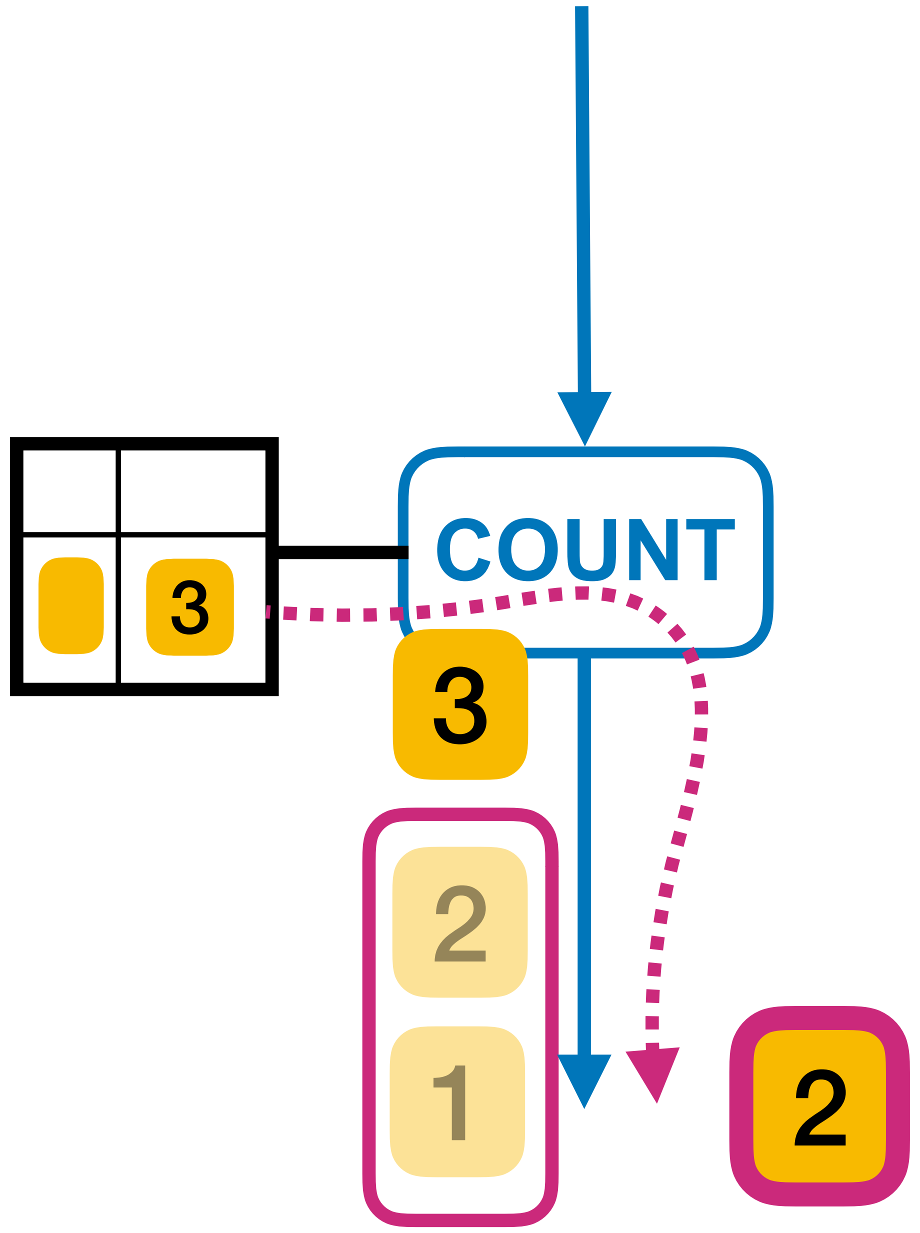
Upquery response is a **snapshot** of state

includes 2 1

does **not** include 🟨

Correctness under concurrency

Goal: upquery restores state as if present all along.



Upquery response is a **snapshot** of state

includes 2 1

does **not** include 🟨

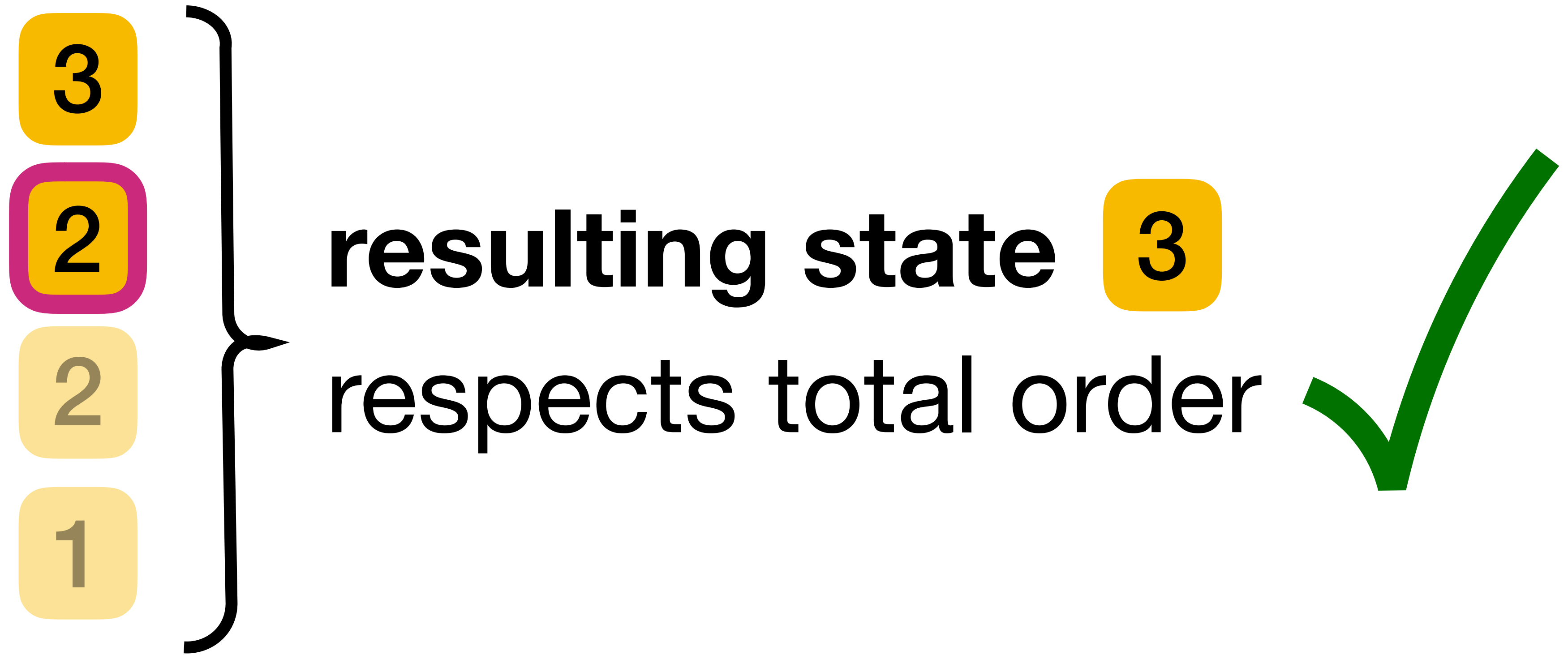
Solution: Maintain **order** of upquery response and surrounding updates, despite lack of global coordination.

Upquery responses in total order with updates

Goal: upquery restores state as if present all along.

Upquery responses in total order with updates

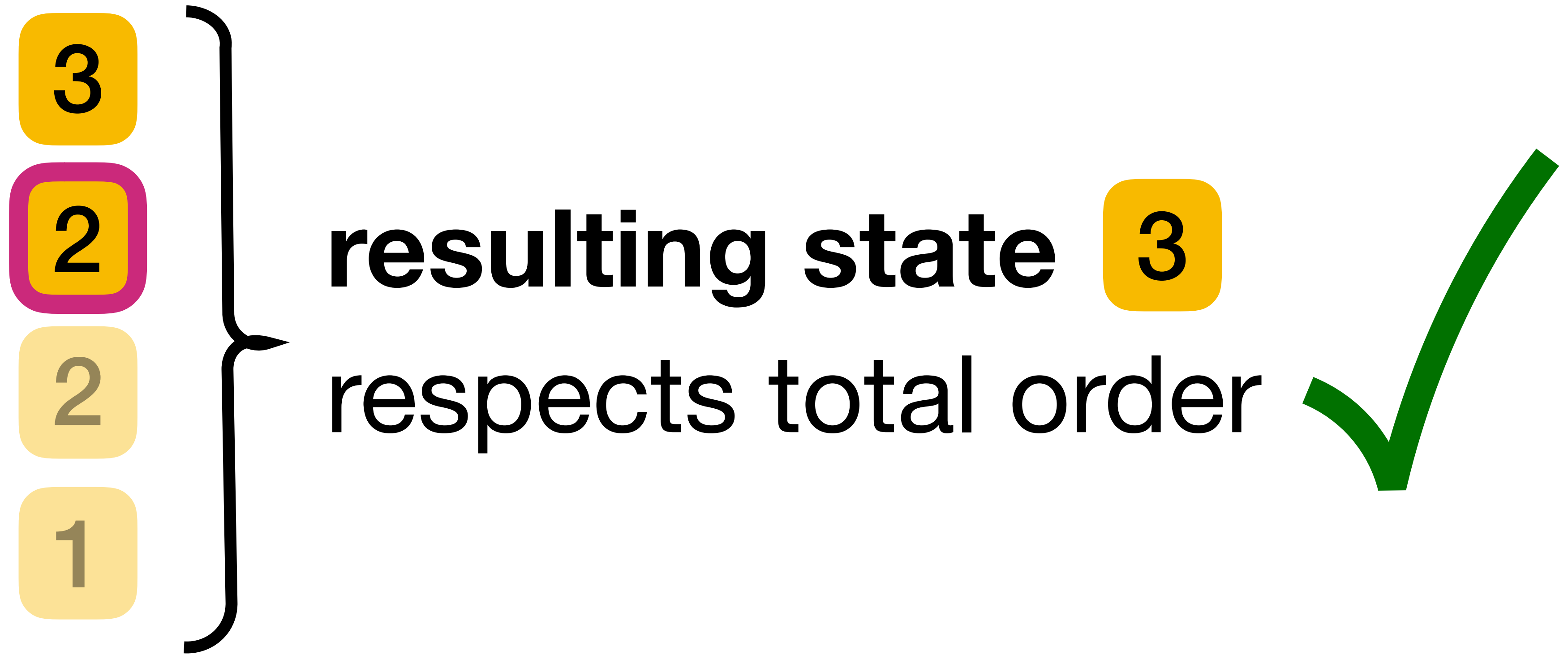
Goal: upquery restores state as if present all along.



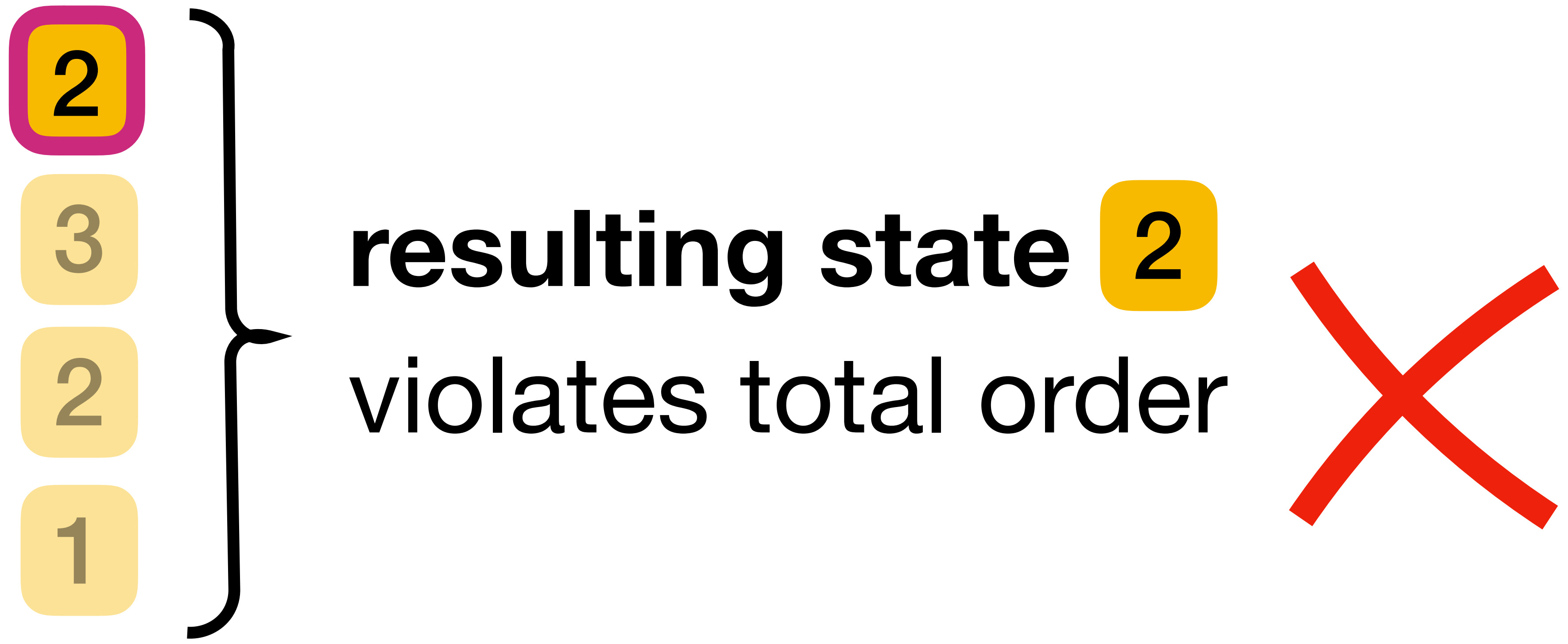
resulting state 3
respects total order ✓

Upquery responses in total order with updates

Goal: upquery restores state as if present all along.



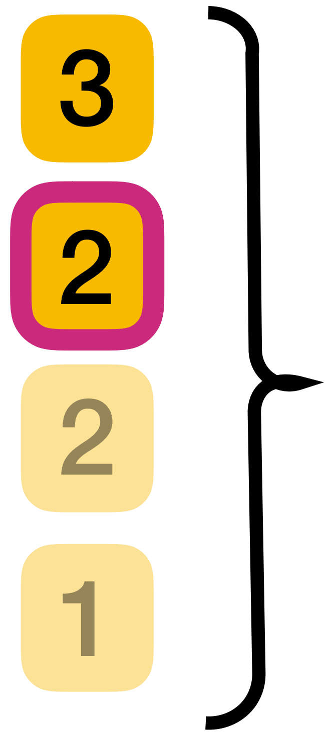
resulting state 3
respects total order ✓



resulting state 2
violates total order ✗

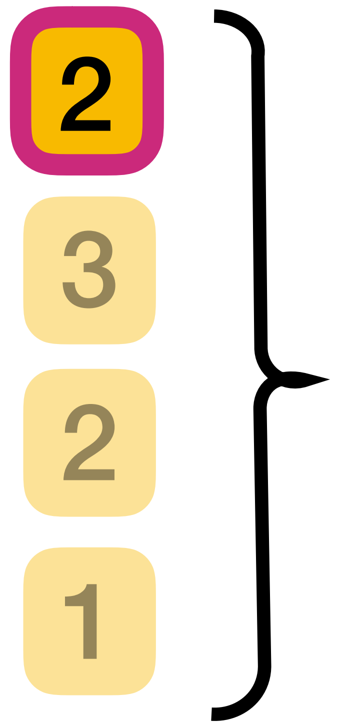
Upquery responses in total order with updates

Goal: upquery restores state as if present all along.



resulting state 3

respects total order ✓



resulting state 2

violates total order ✗

More complex cases: merged upquery responses, evictions (**Paper**).

Challenges implementing partially-stateful data-flow

1. Concurrent upqueries and forward processing — races!

Must maintain **correctness** under concurrency!

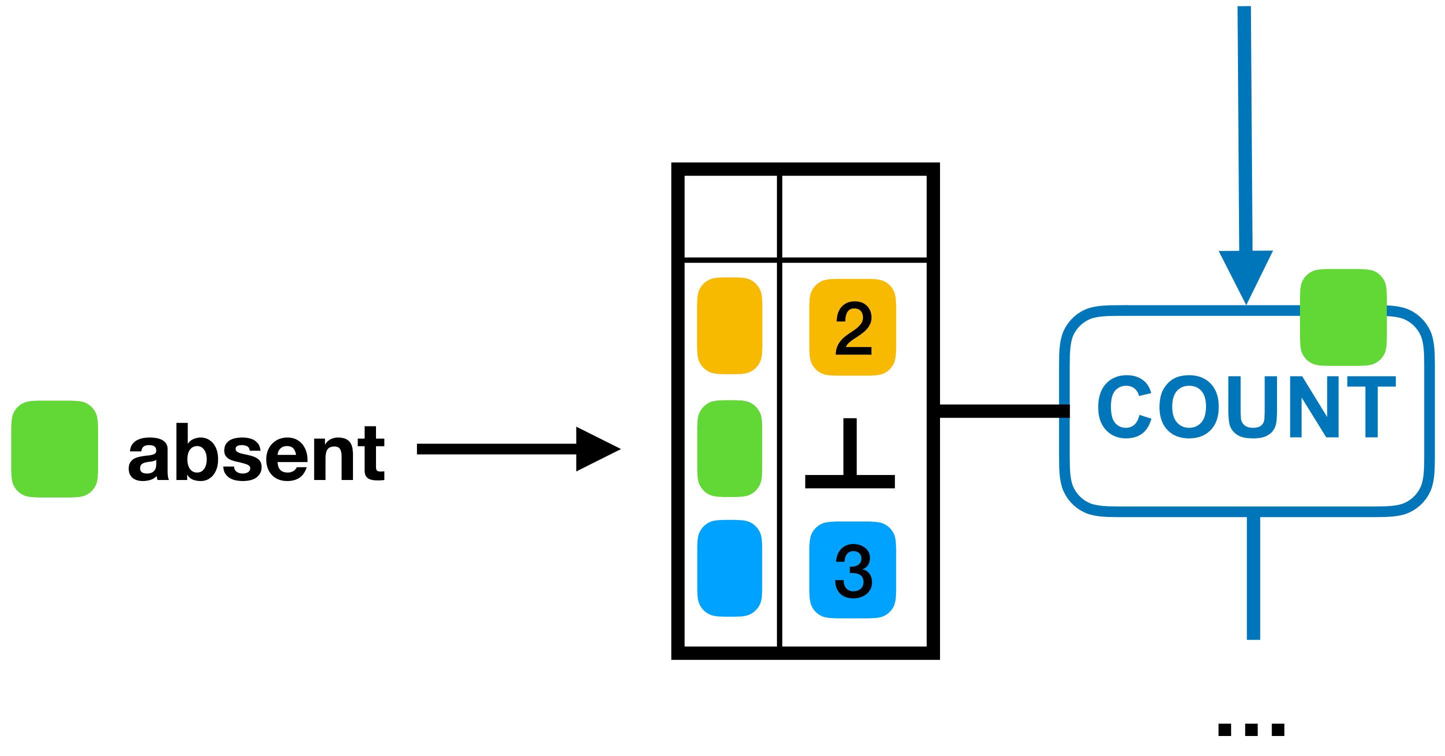
2. Update processing may require absent state

Challenges implementing partially-stateful data-flow

1. Concurrent upqueries and forward processing — races!

Must maintain **correctness** under concurrency!

2. Update processing may require absent state

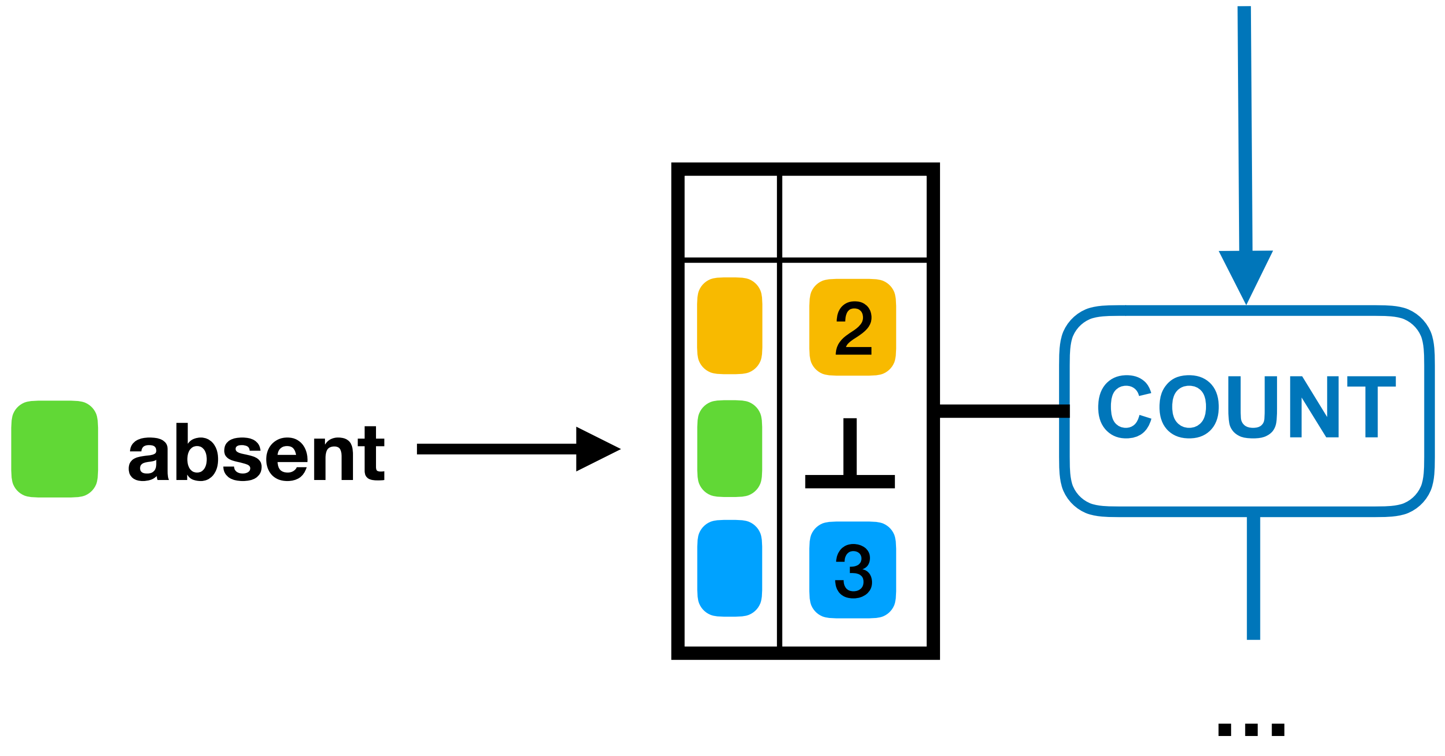


Challenges implementing partially-stateful data-flow

1. Concurrent upqueries and forward processing — races!

Must maintain **correctness** under concurrency!

2. Update processing may require absent state



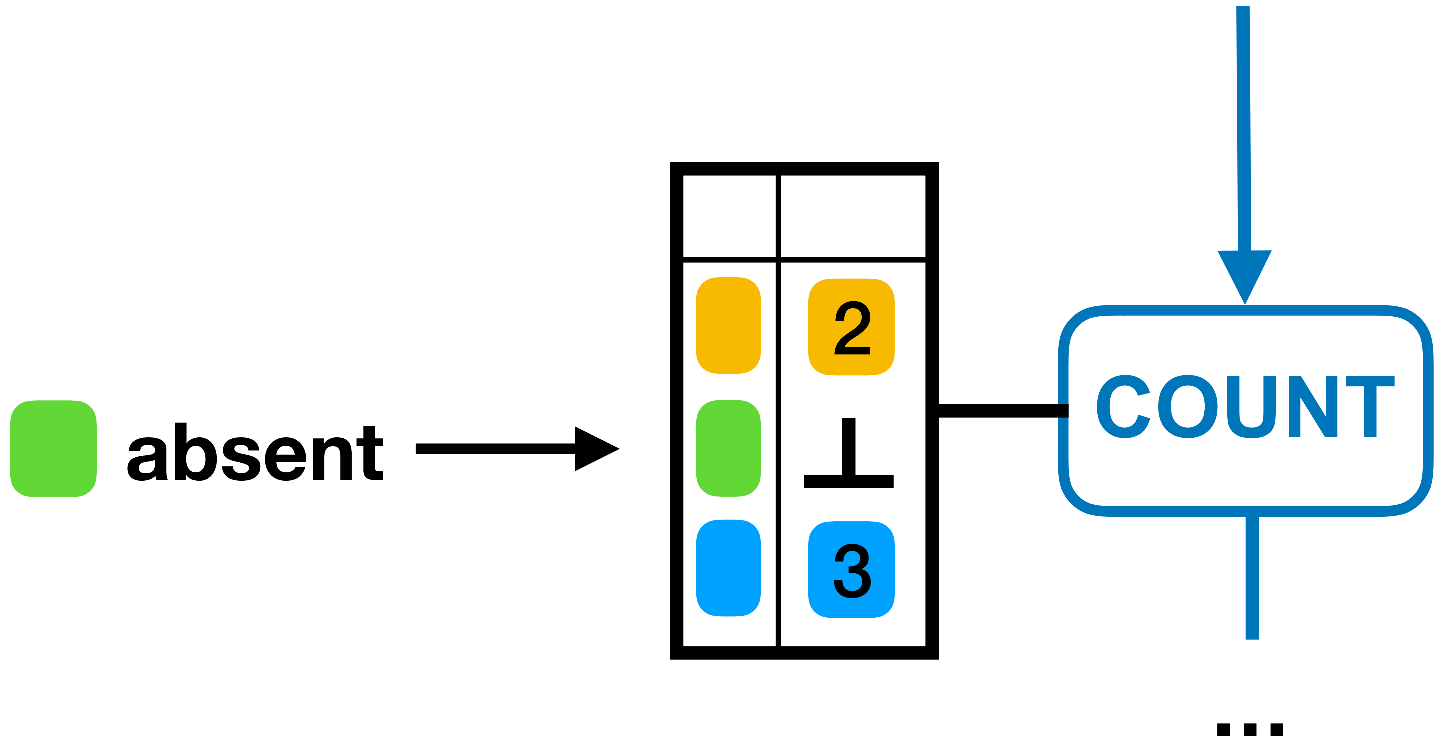
Drop updates that touch absent state, future upquery repeats them.

Challenges implementing partially-stateful data-flow

1. Concurrent upqueries and forward processing — races!

Must maintain **correctness** under concurrency!

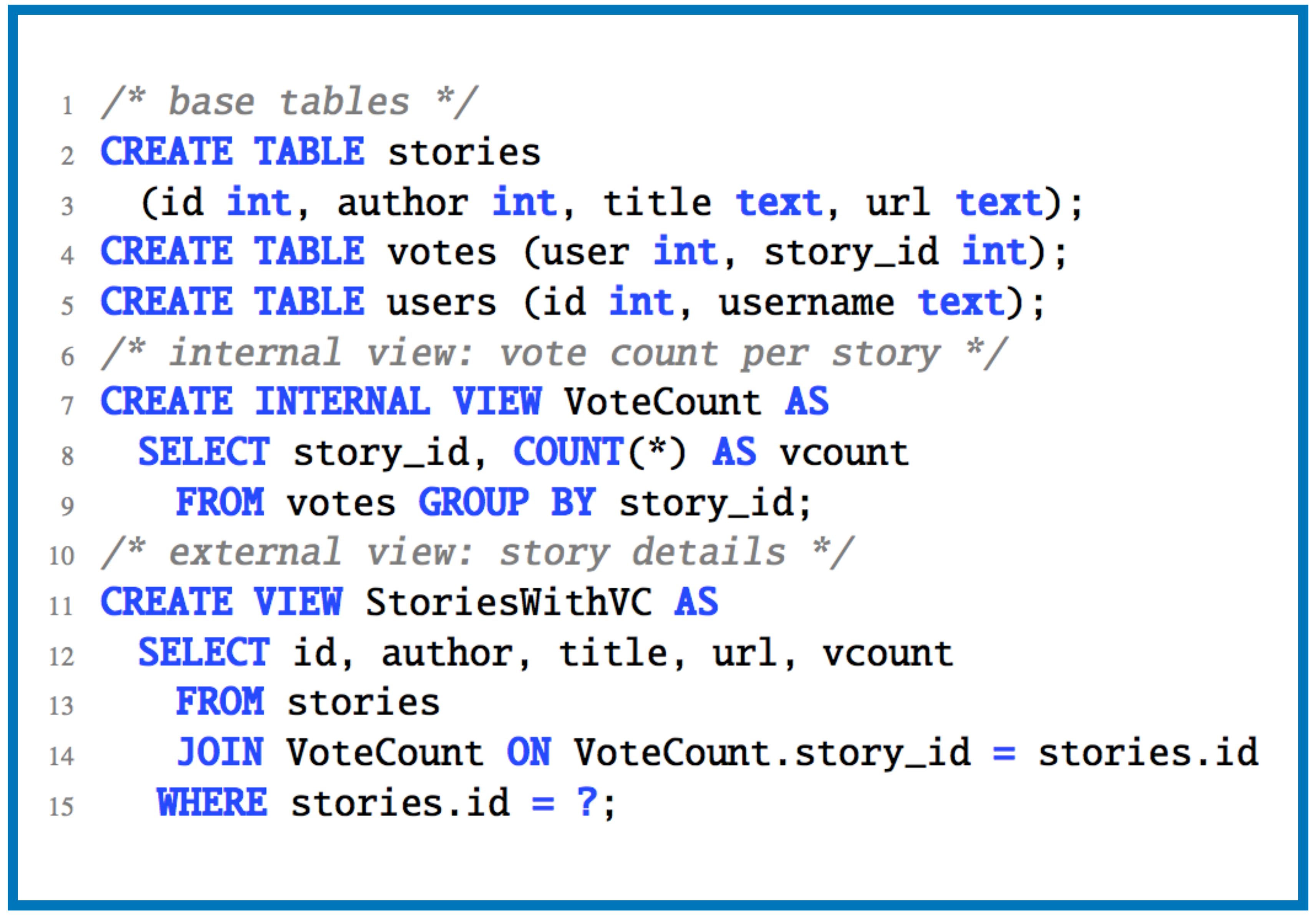
2. Update processing may require absent state **(see Paper)**



Drop updates that touch absent state, future upquery repeats them.

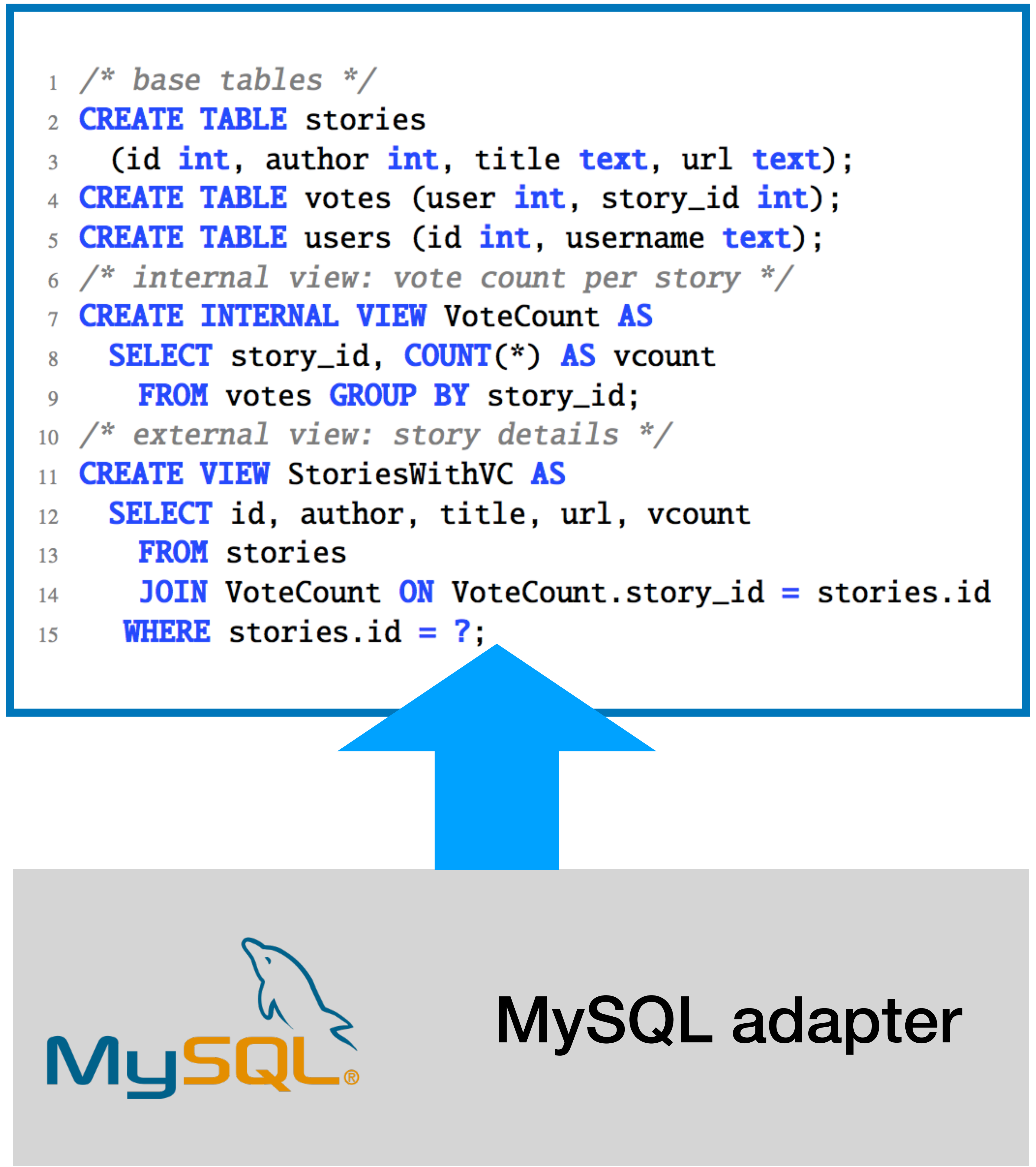
Noria implementation

Noria implementation



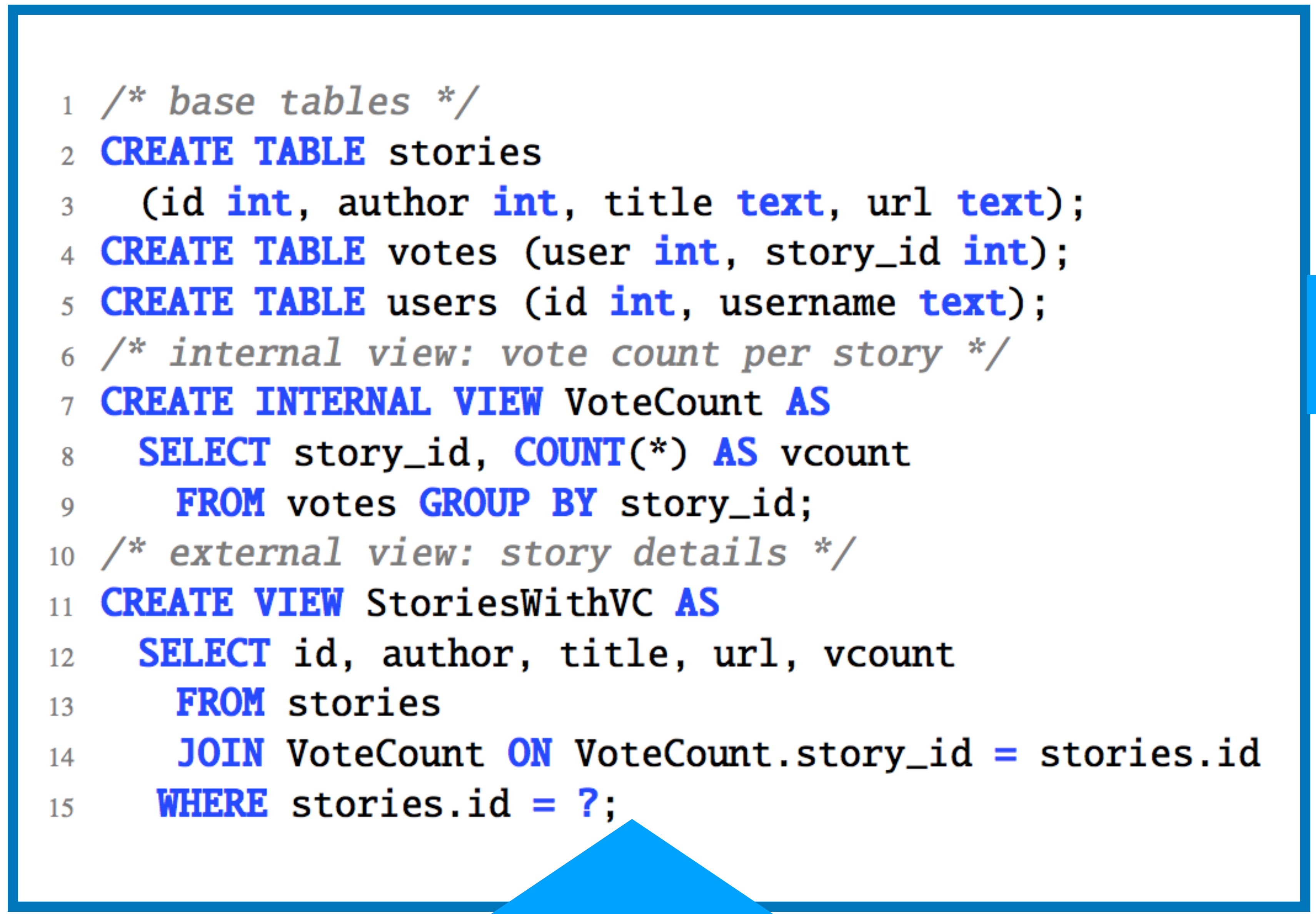
```
/* base tables */
CREATE TABLE stories
  (id int, author int, title text, url text);
CREATE TABLE votes (user int, story_id int);
CREATE TABLE users (id int, username text);
/* internal view: vote count per story */
CREATE INTERNAL VIEW VoteCount AS
  SELECT story_id, COUNT(*) AS vcount
    FROM votes GROUP BY story_id;
/* external view: story details */
CREATE VIEW StoriesWithVC AS
  SELECT id, author, title, url, vcount
    FROM stories
    JOIN VoteCount ON VoteCount.story_id = stories.id
   WHERE stories.id = ?;
```

Noria implementation



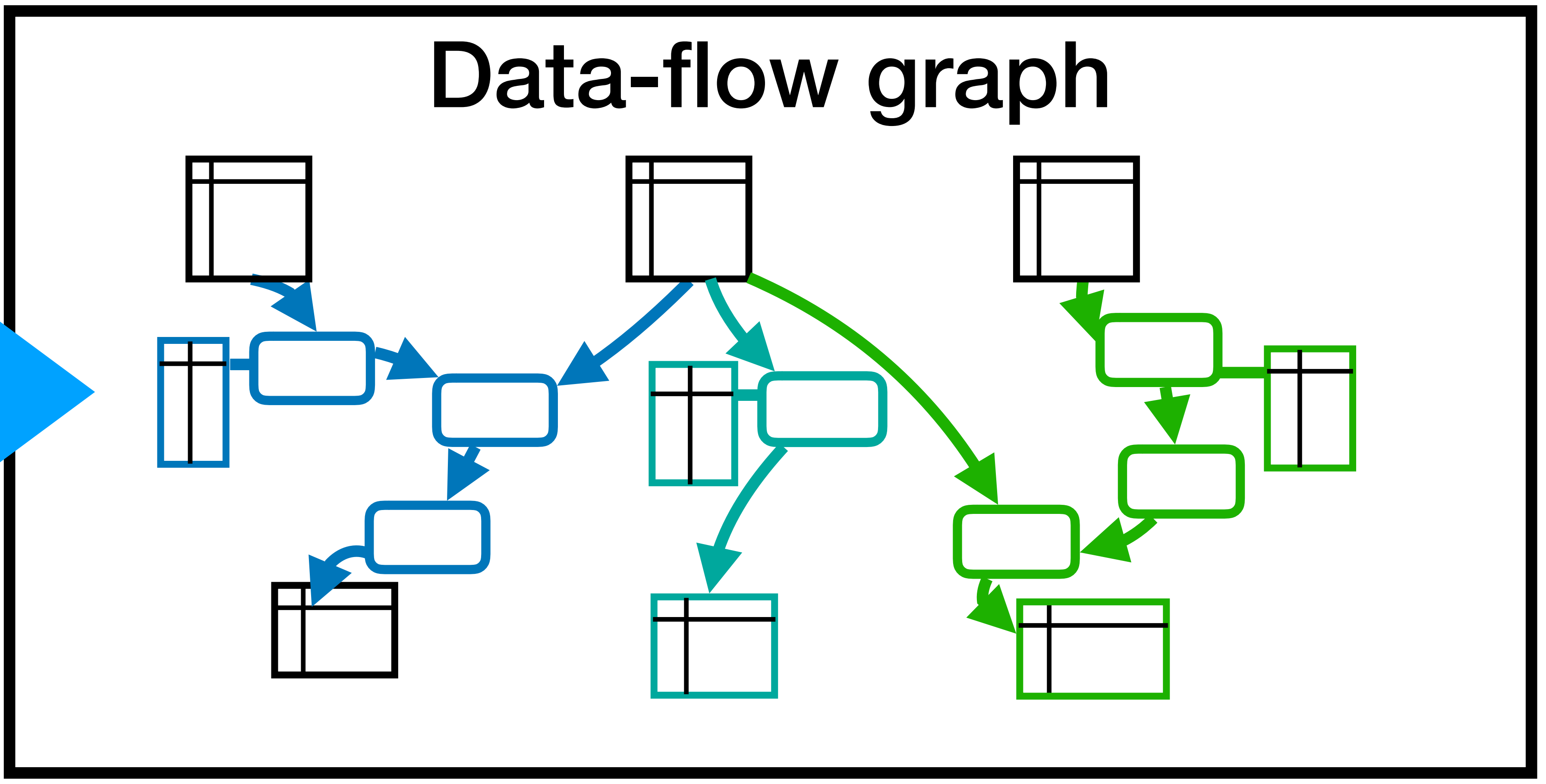
```
/* base tables */
CREATE TABLE stories
  (id int, author int, title text, url text);
CREATE TABLE votes (user int, story_id int);
CREATE TABLE users (id int, username text);
/* internal view: vote count per story */
CREATE INTERNAL VIEW VoteCount AS
  SELECT story_id, COUNT(*) AS vcount
    FROM votes GROUP BY story_id;
/* external view: story details */
CREATE VIEW StoriesWithVC AS
  SELECT id, author, title, url, vcount
    FROM stories
    JOIN VoteCount ON VoteCount.story_id = stories.id
   WHERE stories.id = ?;
```

Noria implementation



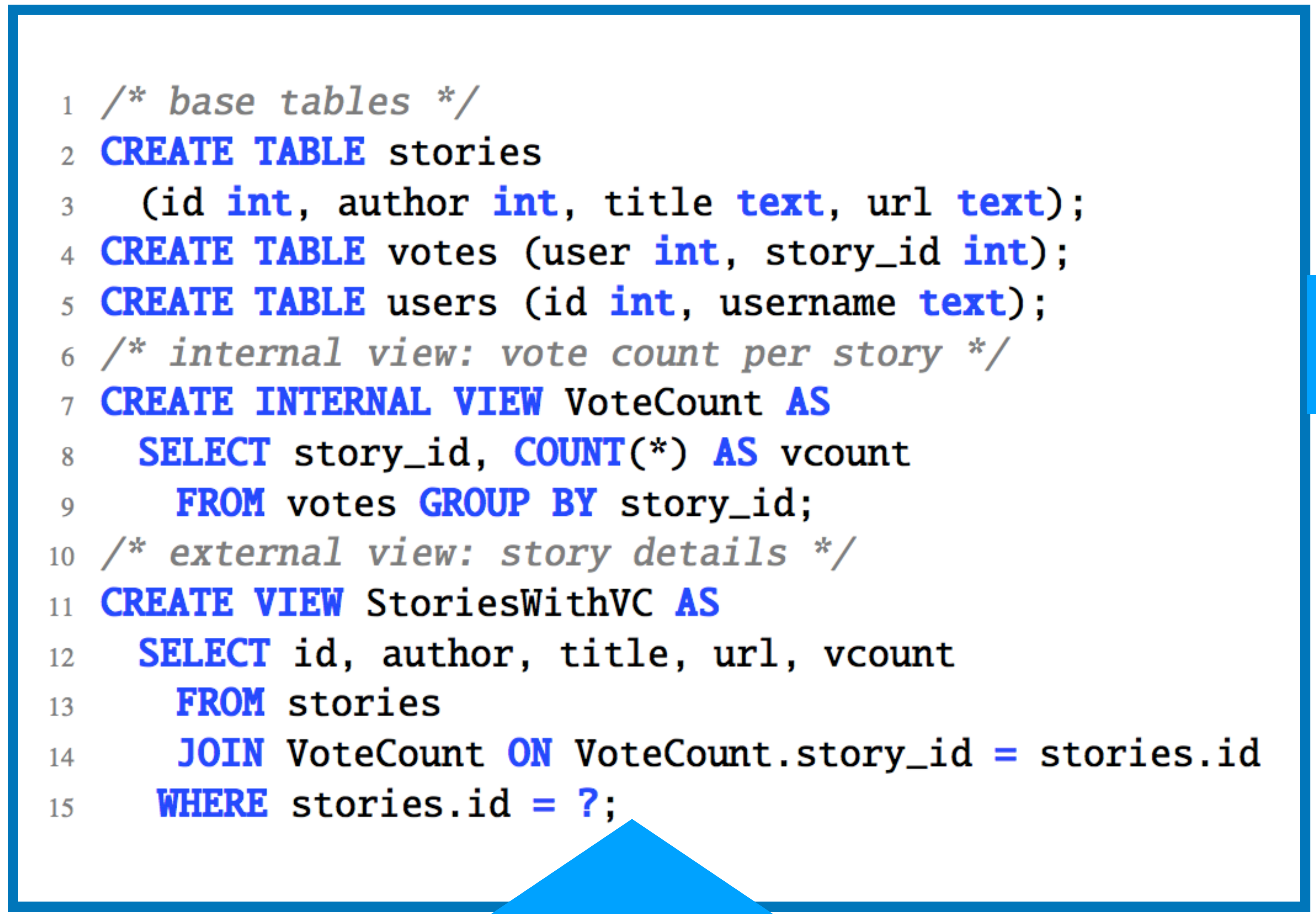
```
/* base tables */
CREATE TABLE stories
  (id int, author int, title text, url text);
CREATE TABLE votes (user int, story_id int);
CREATE TABLE users (id int, username text);
/* internal view: vote count per story */
CREATE INTERNAL VIEW VoteCount AS
  SELECT story_id, COUNT(*) AS vcount
    FROM votes GROUP BY story_id;
/* external view: story details */
CREATE VIEW StoriesWithVC AS
  SELECT id, author, title, url, vcount
    FROM stories
    JOIN VoteCount ON VoteCount.story_id = stories.id
   WHERE stories.id = ?;
```

Transform

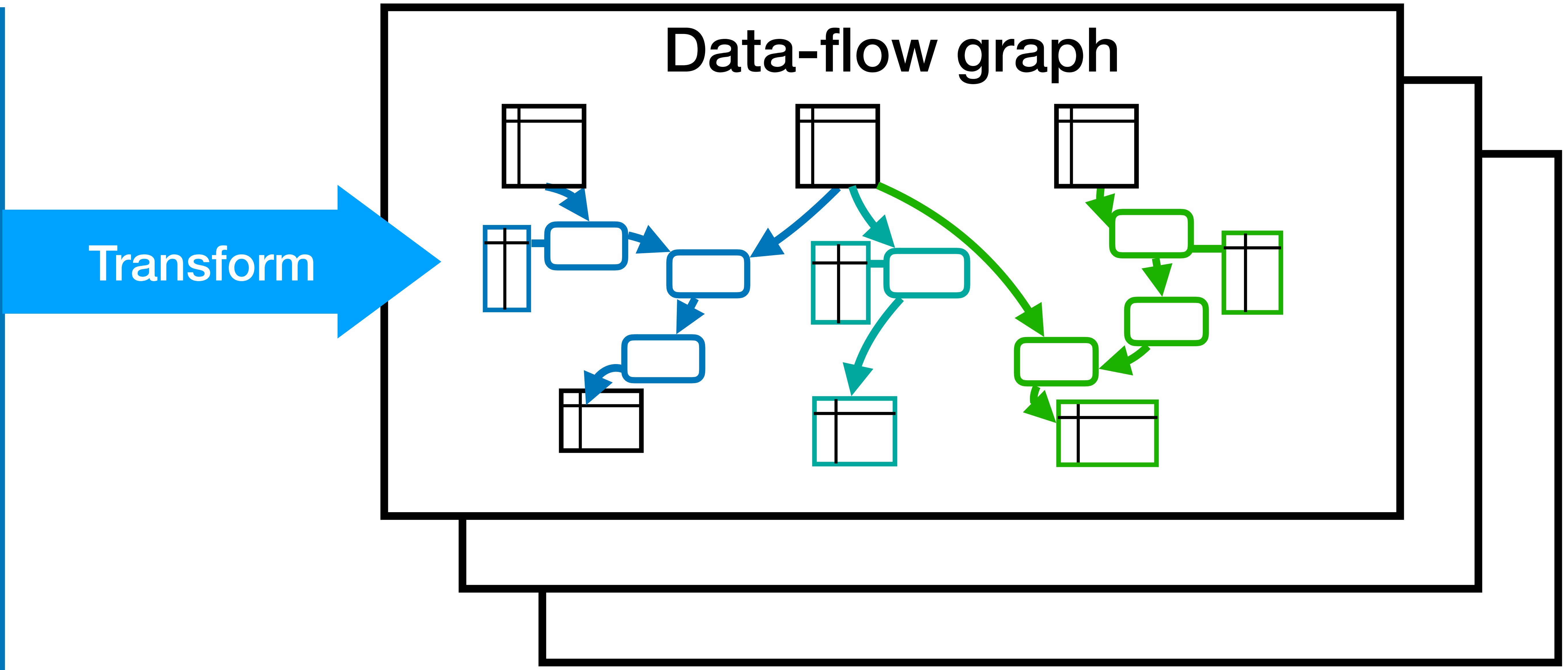


MySQL adapter

Noria implementation

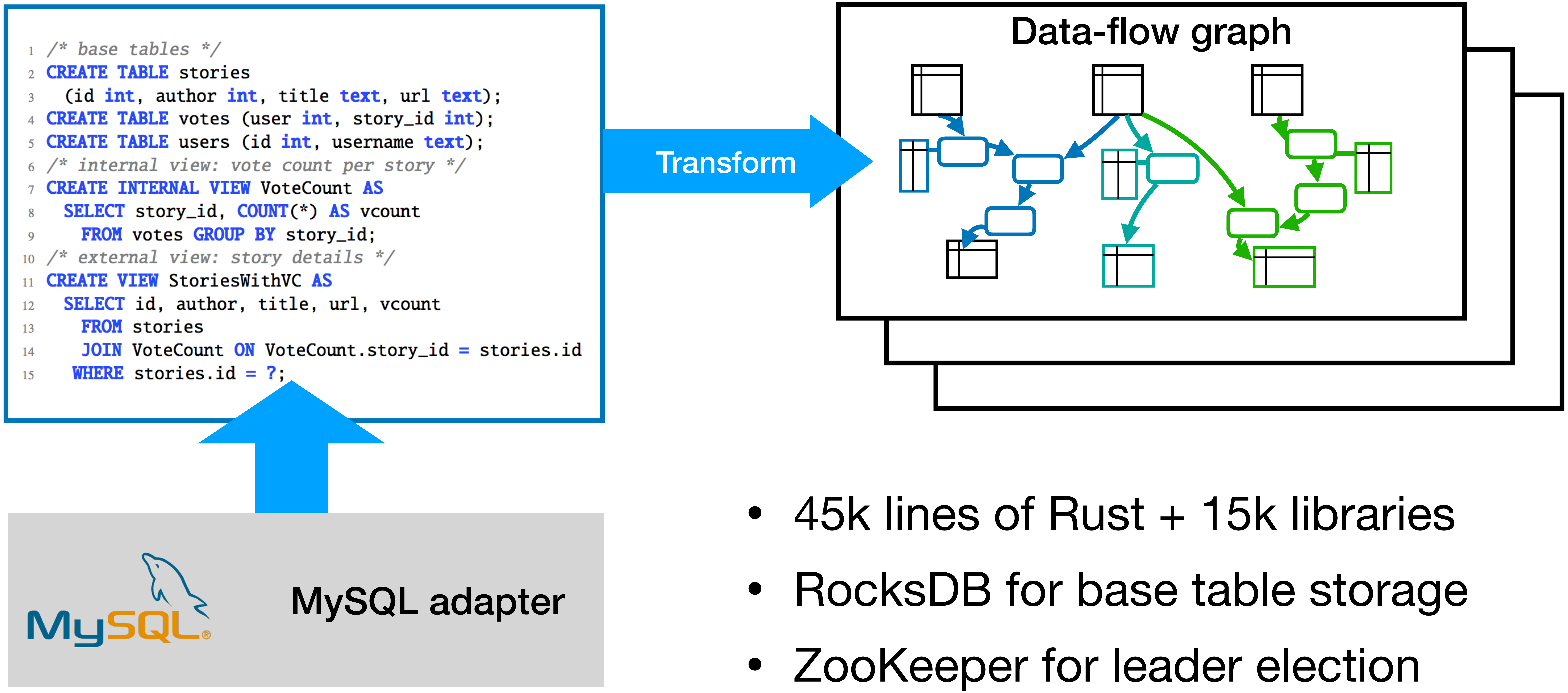


```
/* base tables */
CREATE TABLE stories
  (id int, author int, title text, url text);
CREATE TABLE votes (user int, story_id int);
CREATE TABLE users (id int, username text);
/* internal view: vote count per story */
CREATE INTERNAL VIEW VoteCount AS
  SELECT story_id, COUNT(*) AS vcount
    FROM votes GROUP BY story_id;
/* external view: story details */
CREATE VIEW StoriesWithVC AS
  SELECT id, author, title, url, vcount
    FROM stories
    JOIN VoteCount ON VoteCount.story_id = stories.id
   WHERE stories.id = ?;
```



MySQL adapter

Noria implementation



```sql
/* base tables */
CREATE TABLE stories
  (id int, author int, title text, url text);
CREATE TABLE votes (user int, story_id int);
CREATE TABLE users (id int, username text);
/* internal view: vote count per story */
CREATE INTERNAL VIEW VoteCount AS
  SELECT story_id, COUNT(*) AS vcount
    FROM votes GROUP BY story_id;
/* external view: story details */
CREATE VIEW StoriesWithVC AS
  SELECT id, author, title, url, vcount
    FROM stories
    JOIN VoteCount ON VoteCount.story_id = stories.id
   WHERE stories.id = ?;
```

Transform

Data-flow graph

MySQL adapter

- 45k lines of Rust + 15k libraries
- RocksDB for base table storage
- ZooKeeper for leader election

Evaluation

1. Can Noria improve a real web application’s performance?

2. How does Noria compare to alternative approaches?

3. Can Noria change queries without downtime?

Evaluation

1. Can Noria improve a real web application's performance?
2. How does Noria compare to alternative approaches?
3. Can Noria change queries without downtime?

Setup Amazon EC2 c5.4xlarge instance (16 vCPUs)

Open-loop clients, measuring latency & throughput

Evaluation

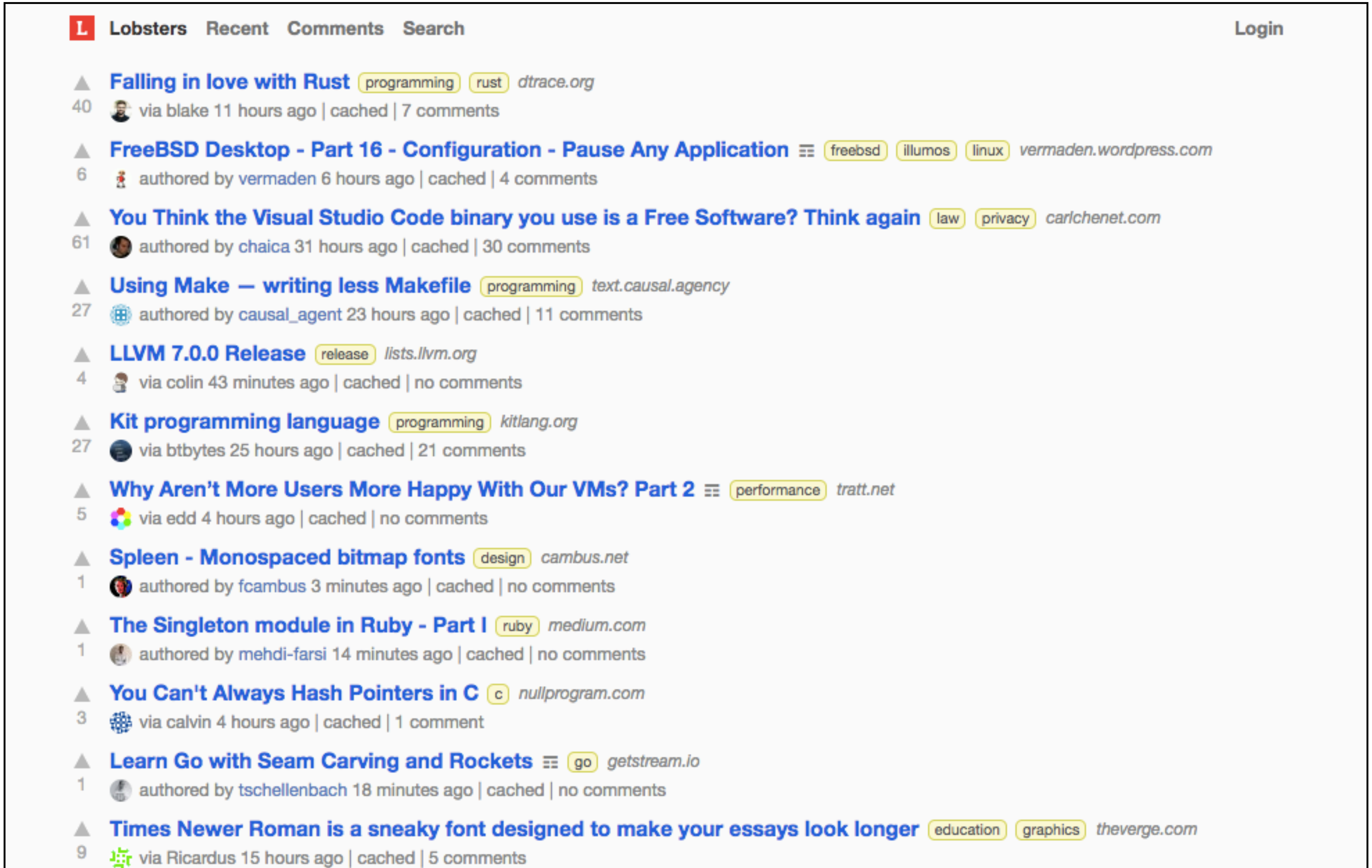
1. Can Noria improve a real web application's performance?
2. How does Noria compare to alternative approaches?
3. Can Noria change queries without downtime?

Setup Amazon EC2 c5.4xlarge instance (16 vCPUs)

Open-loop clients, measuring latency & throughput

multi-machine experiments
comparison with differential dataflow } **see Paper**

Case study: Lobsters (http://lobste.rs)



▲ 40 Falling in love with Rust programming rust dtrace.org
via blake 11 hours ago | cached | 7 comments

▲ 6 FreeBSD Desktop - Part 16 - Configuration - Pause Any Application freebsd illumos linux vermaden.wordpress.com
authored by vermaden 6 hours ago | cached | 4 comments

▲ 61 You Think the Visual Studio Code binary you use is a Free Software? Think again law privacy carlchenet.com
authored by chaica 31 hours ago | cached | 30 comments

▲ 27 Using Make — writing less Makefile programming text.causal.agency
authored by causal_agent 23 hours ago | cached | 11 comments

▲ 4 LLVM 7.0.0 Release release lists.llvm.org
via colin 43 minutes ago | cached | no comments

▲ 27 Kit programming language programming kitlang.org
via btbytes 25 hours ago | cached | 21 comments

▲ 5 Why Aren't More Users More Happy With Our VMs? Part 2 performance tratt.net
via edd 4 hours ago | cached | no comments

▲ 1 Spleen - Monospaced bitmap fonts design cambus.net
authored by fcambus 3 minutes ago | cached | no comments

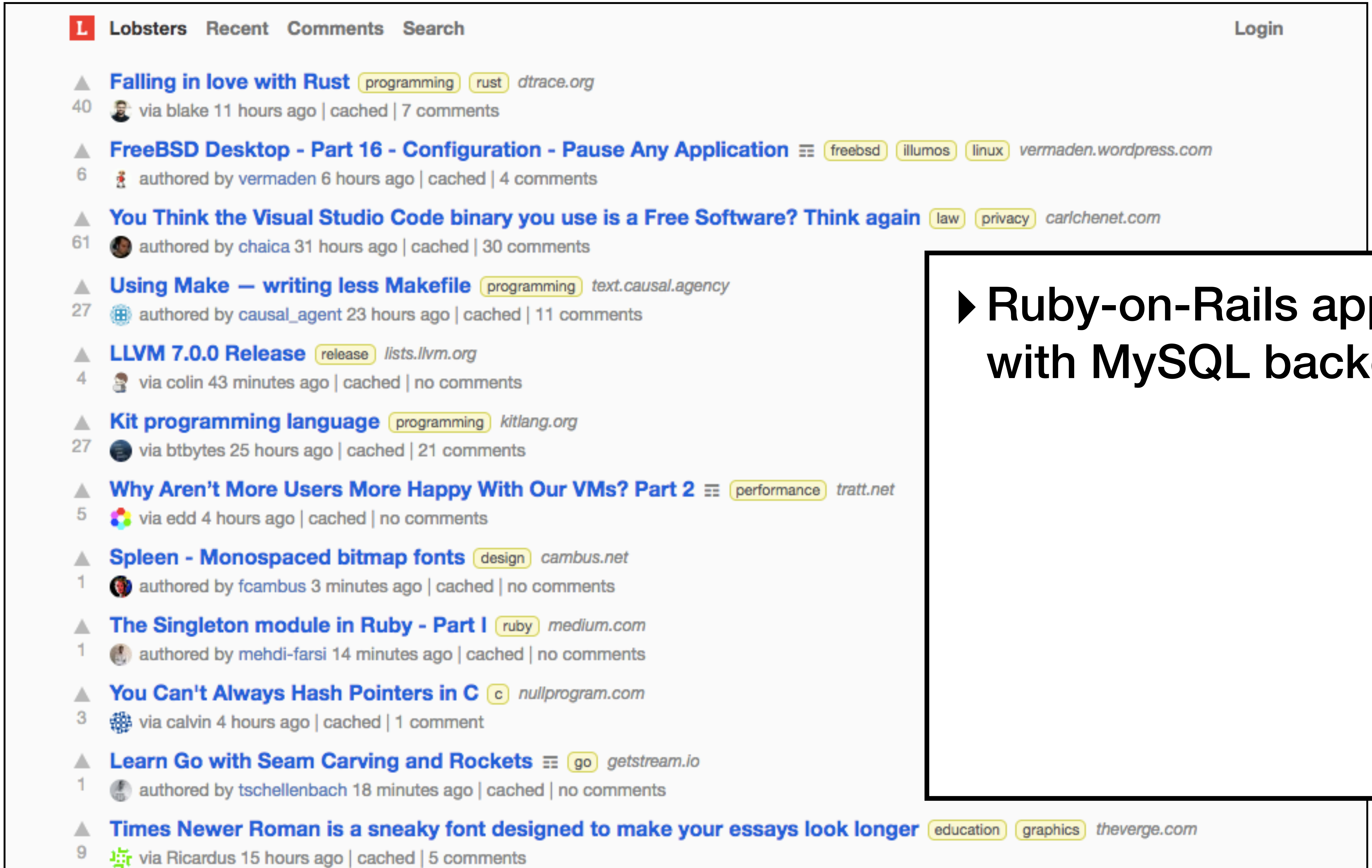
▲ 1 The Singleton module in Ruby - Part I ruby medium.com
authored by mehdi-farsi 14 minutes ago | cached | no comments

▲ 3 You Can't Always Hash Pointers in C c nullprogram.com
via calvin 4 hours ago | cached | 1 comment

▲ 1 Learn Go with Seam Carving and Rockets go getstream.io
authored by tschellenbach 18 minutes ago | cached | no comments

▲ 9 Times Newer Roman is a sneaky font designed to make your essays look longer education graphics theverge.com
via Ricardus 15 hours ago | cached | 5 comments

Case study: Lobsters (http://lobste.rs)



L Lobsters Recent Comments Search Login

▲ 40 Falling in love with Rust (programming) (rust) *dtrace.org*
via blake 11 hours ago | cached | 7 comments

▲ 6 FreeBSD Desktop - Part 16 - Configuration - Pause Any Application (freebsd) (illumos) (linux) *vermaden.wordpress.com*
authored by vermaden 6 hours ago | cached | 4 comments

▲ 61 You Think the Visual Studio Code binary you use is a Free Software? Think again (law) (privacy) *carlchenet.com*
authored by chaica 31 hours ago | cached | 30 comments

▲ 27 Using Make — writing less Makefile (programming) *text.causal.agency*
authored by causal_agent 23 hours ago | cached | 11 comments

▲ 4 LLVM 7.0.0 Release (release) *lists.llvm.org*
via colin 43 minutes ago | cached | no comments

▲ 27 Kit programming language (programming) *kitlang.org*
via btbytes 25 hours ago | cached | 21 comments

▲ 5 Why Aren't More Users More Happy With Our VMs? Part 2 (performance) *tratt.net*
via edd 4 hours ago | cached | no comments

▲ 1 Spleen - Monospaced bitmap fonts (design) *cambus.net*
authored by fcambus 3 minutes ago | cached | no comments

▲ 1 The Singleton module in Ruby - Part I (ruby) *medium.com*
authored by mehdi-farsi 14 minutes ago | cached | no comments

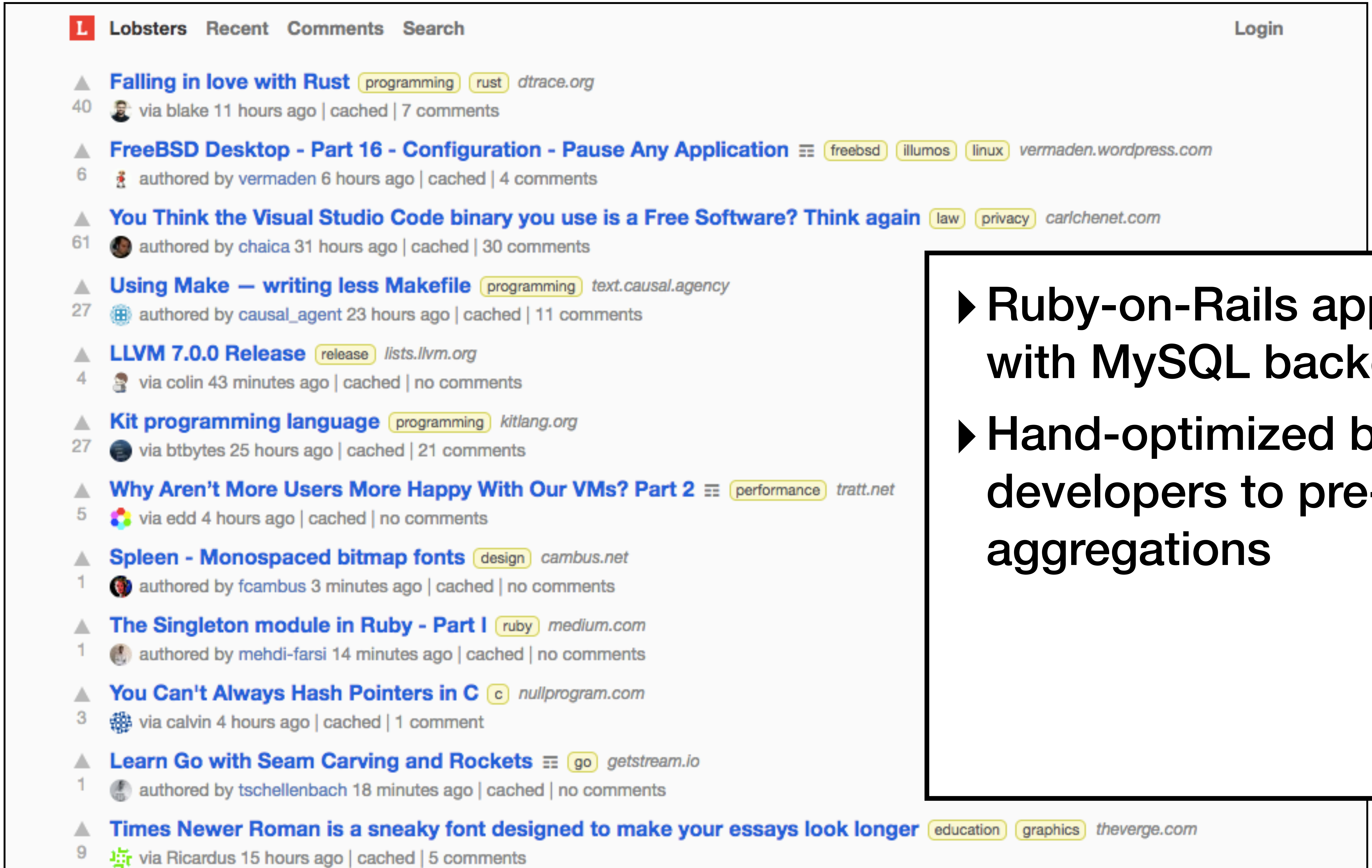
▲ 3 You Can't Always Hash Pointers in C (c) *nullprogram.com*
via calvin 4 hours ago | cached | 1 comment

▲ 1 Learn Go with Seam Carving and Rockets (go) *getstream.io*
authored by tschellenbach 18 minutes ago | cached | no comments

▲ 9 Times Newer Roman is a sneaky font designed to make your essays look longer (education) (graphics) *theverge.com*
via Ricardus 15 hours ago | cached | 5 comments

- Ruby-on-Rails application with MySQL backend

Case study: Lobsters (http://lobste.rs)



L Lobsters Recent Comments Search Login

▲ 40 Falling in love with Rust programming rust dtrace.org
via blake 11 hours ago | cached | 7 comments

▲ 6 FreeBSD Desktop - Part 16 - Configuration - Pause Any Application freebsd illumos linux vermaden.wordpress.com
authored by vermaden 6 hours ago | cached | 4 comments

▲ 61 You Think the Visual Studio Code binary you use is a Free Software? Think again law privacy carlchenet.com
authored by chaica 31 hours ago | cached | 30 comments

▲ 27 Using Make — writing less Makefile programming text.causal.agency
authored by causal_agent 23 hours ago | cached | 11 comments

▲ 4 LLVM 7.0.0 Release release lists.llvm.org
via colin 43 minutes ago | cached | no comments

▲ 27 Kit programming language programming kitlang.org
via btbytes 25 hours ago | cached | 21 comments

▲ 5 Why Aren't More Users More Happy With Our VMs? Part 2 performance tratt.net
via edd 4 hours ago | cached | no comments

▲ 1 Spleen - Monospaced bitmap fonts design cambus.net
authored by fcambus 3 minutes ago | cached | no comments

▲ 1 The Singleton module in Ruby - Part I ruby medium.com
authored by mehdi-farsi 14 minutes ago | cached | no comments

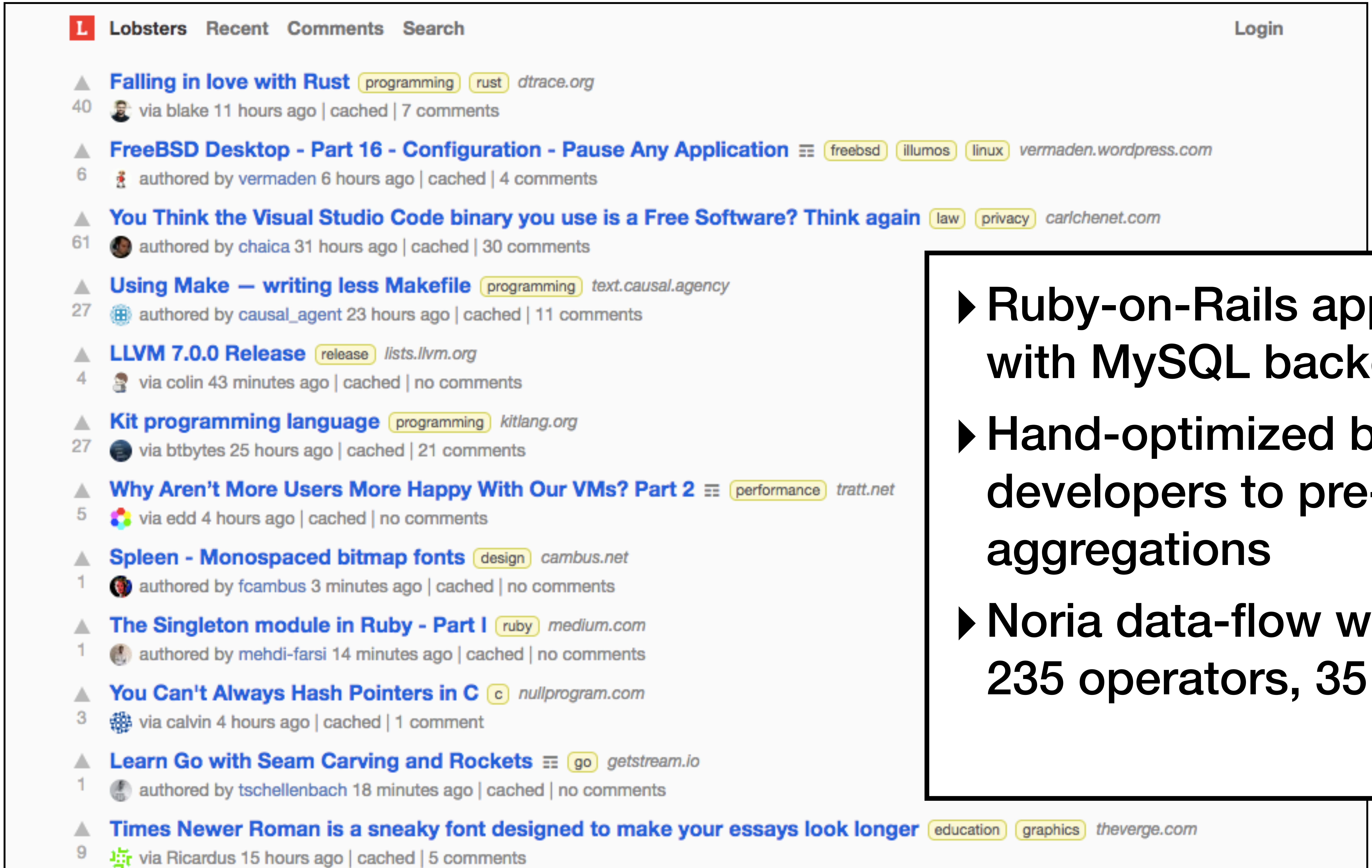
▲ 3 You Can't Always Hash Pointers in C c nullprogram.com
via calvin 4 hours ago | cached | 1 comment

▲ 1 Learn Go with Seam Carving and Rockets go getstream.io
authored by tschellenbach 18 minutes ago | cached | no comments

▲ 9 Times Newer Roman is a sneaky font designed to make your essays look longer education graphics theverge.com
via Ricardus 15 hours ago | cached | 5 comments

- Ruby-on-Rails application with MySQL backend
- Hand-optimized by developers to pre-compute aggregations

Case study: Lobsters (http://lobste.rs)



L Lobsters Recent Comments Search Login

▲ 40 Falling in love with Rust (programming) (rust) dtrace.org
via blake 11 hours ago | cached | 7 comments

▲ 6 FreeBSD Desktop - Part 16 - Configuration - Pause Any Application (freebsd) (illumos) (linux) vermaden.wordpress.com
authored by vermaden 6 hours ago | cached | 4 comments

▲ 61 You Think the Visual Studio Code binary you use is a Free Software? Think again (law) (privacy) carlchenet.com
authored by chaica 31 hours ago | cached | 30 comments

▲ 27 Using Make — writing less Makefile (programming) text.causal.agency
authored by causal_agent 23 hours ago | cached | 11 comments

▲ 4 LLVM 7.0.0 Release (release) lists.llvm.org
via colin 43 minutes ago | cached | no comments

▲ 27 Kit programming language (programming) kitlang.org
via btbytes 25 hours ago | cached | 21 comments

▲ 5 Why Aren't More Users More Happy With Our VMs? Part 2 (performance) tratt.net
via edd 4 hours ago | cached | no comments

▲ 1 Spleen - Monospaced bitmap fonts (design) cambus.net
authored by fcambus 3 minutes ago | cached | no comments

▲ 1 The Singleton module in Ruby - Part I (ruby) medium.com
authored by mehdi-farsi 14 minutes ago | cached | no comments

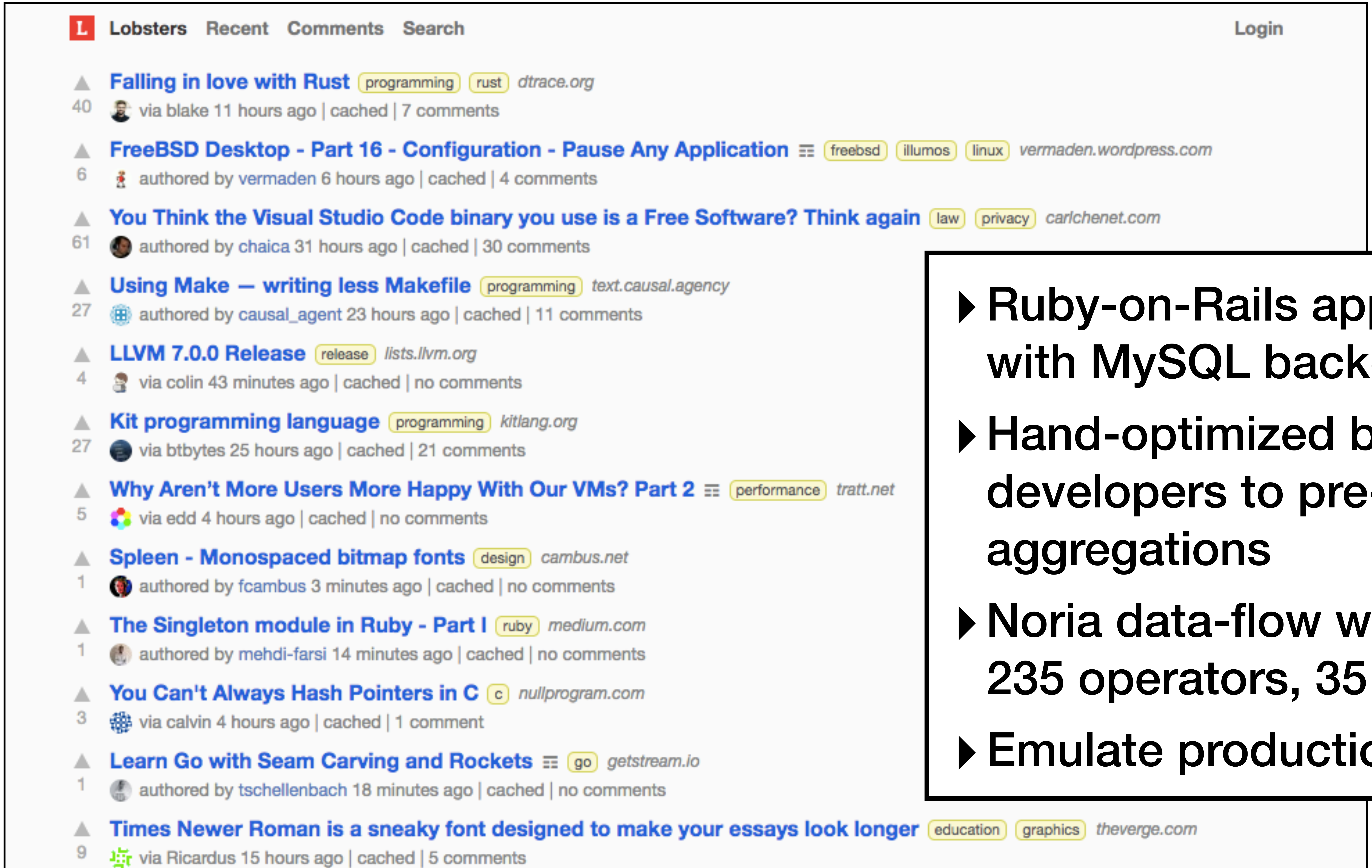
▲ 3 You Can't Always Hash Pointers in C (c) nullprogram.com
via calvin 4 hours ago | cached | 1 comment

▲ 1 Learn Go with Seam Carving and Rockets (go) getstream.io
authored by tschellenbach 18 minutes ago | cached | no comments

▲ 9 Times Newer Roman is a sneaky font designed to make your essays look longer (education) (graphics) theverge.com
via Ricardus 15 hours ago | cached | 5 comments

- Ruby-on-Rails application with MySQL backend
- Hand-optimized by developers to pre-compute aggregations
- Noria data-flow with 235 operators, 35 views

Case study: Lobsters (http://lobste.rs)



Lobsters Recent Comments Search Login

40 Falling in love with Rust programming rust dtrace.org
via blake 11 hours ago | cached | 7 comments

6 FreeBSD Desktop - Part 16 - Configuration - Pause Any Application freebsd illumos linux vermaden.wordpress.com
authored by vermaden 6 hours ago | cached | 4 comments

61 You Think the Visual Studio Code binary you use is a Free Software? Think again law privacy carlchenet.com
authored by chaica 31 hours ago | cached | 30 comments

27 Using Make — writing less Makefile programming text.causal.agency
authored by causal_agent 23 hours ago | cached | 11 comments

4 LLVM 7.0.0 Release release lists.llvm.org
via colin 43 minutes ago | cached | no comments

27 Kit programming language programming kitlang.org
via btbytes 25 hours ago | cached | 21 comments

5 Why Aren't More Users More Happy With Our VMs? Part 2 performance tratt.net
via edd 4 hours ago | cached | no comments

1 Spleen - Monospaced bitmap fonts design cambus.net
authored by fcambus 3 minutes ago | cached | no comments

1 The Singleton module in Ruby - Part I ruby medium.com
authored by mehdi-farsi 14 minutes ago | cached | no comments

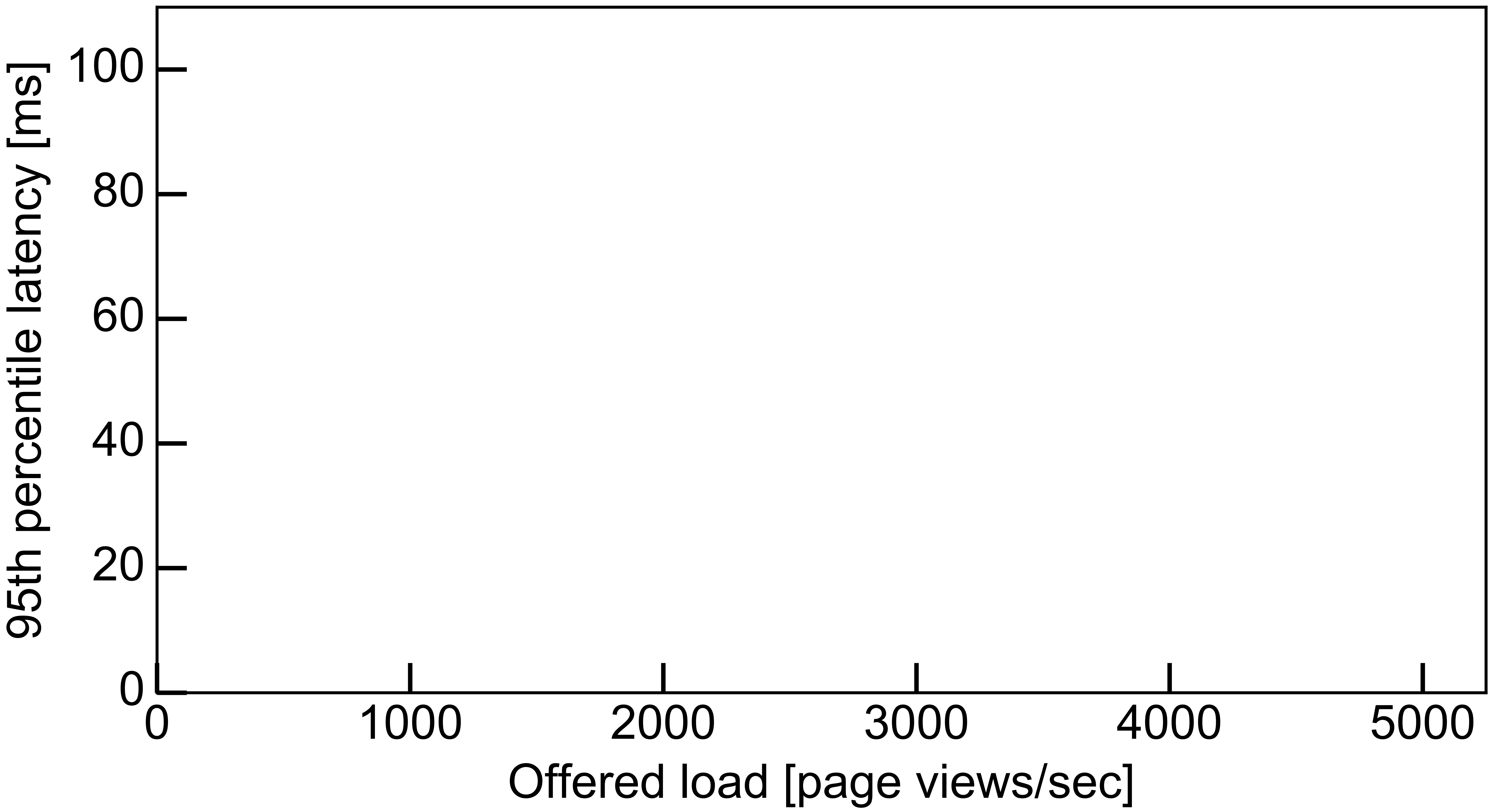
3 You Can't Always Hash Pointers in C c nullprogram.com
via calvin 4 hours ago | cached | 1 comment

1 Learn Go with Seam Carving and Rockets go getstream.io
authored by tschellenbach 18 minutes ago | cached | no comments

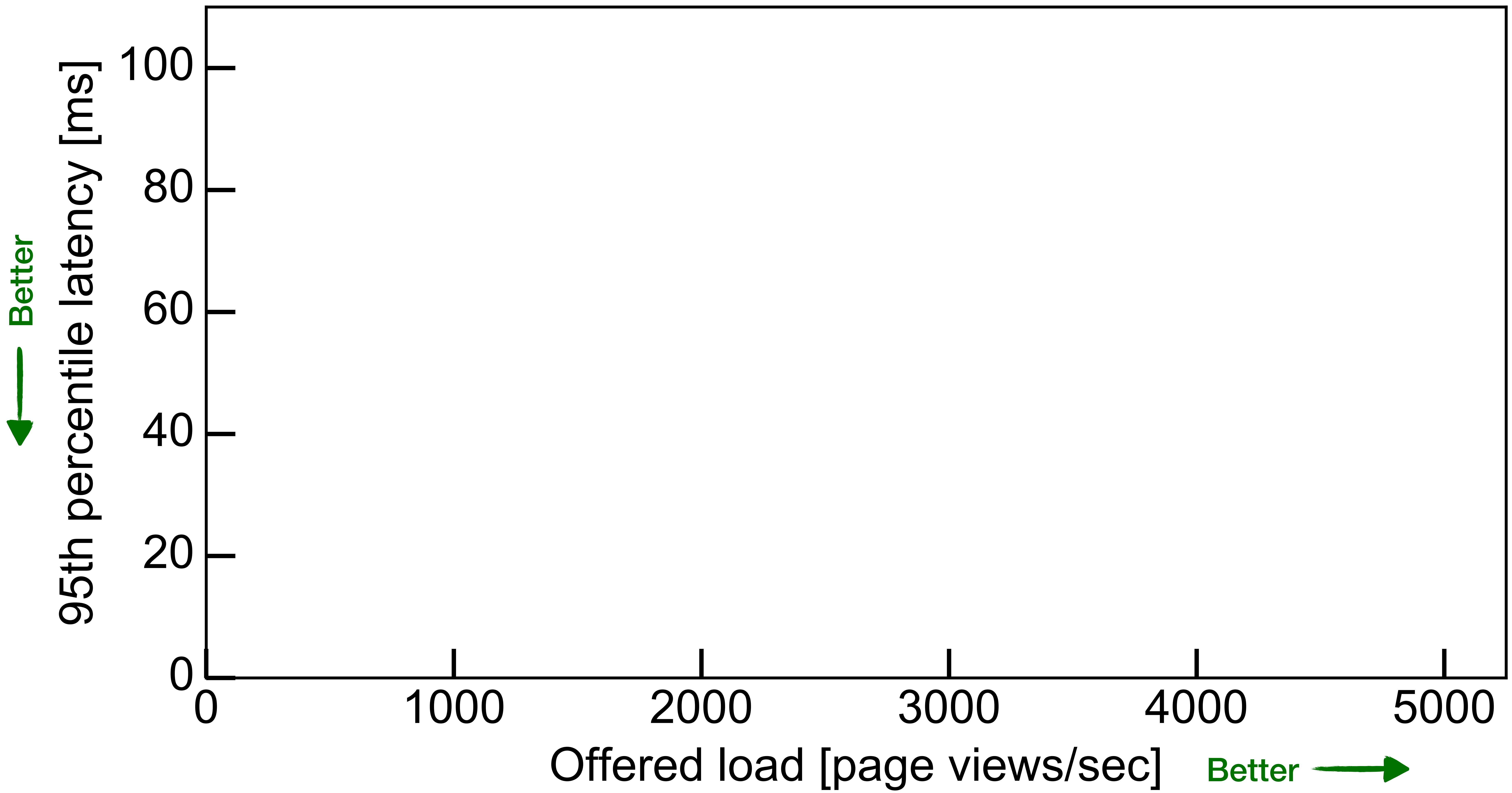
9 Times Newer Roman is a sneaky font designed to make your essays look longer education graphics theverge.com
via Ricardus 15 hours ago | cached | 5 comments

- Ruby-on-Rails application with MySQL backend
- Hand-optimized by developers to pre-compute aggregations
- Noria data-flow with 235 operators, 35 views
- Emulate production load

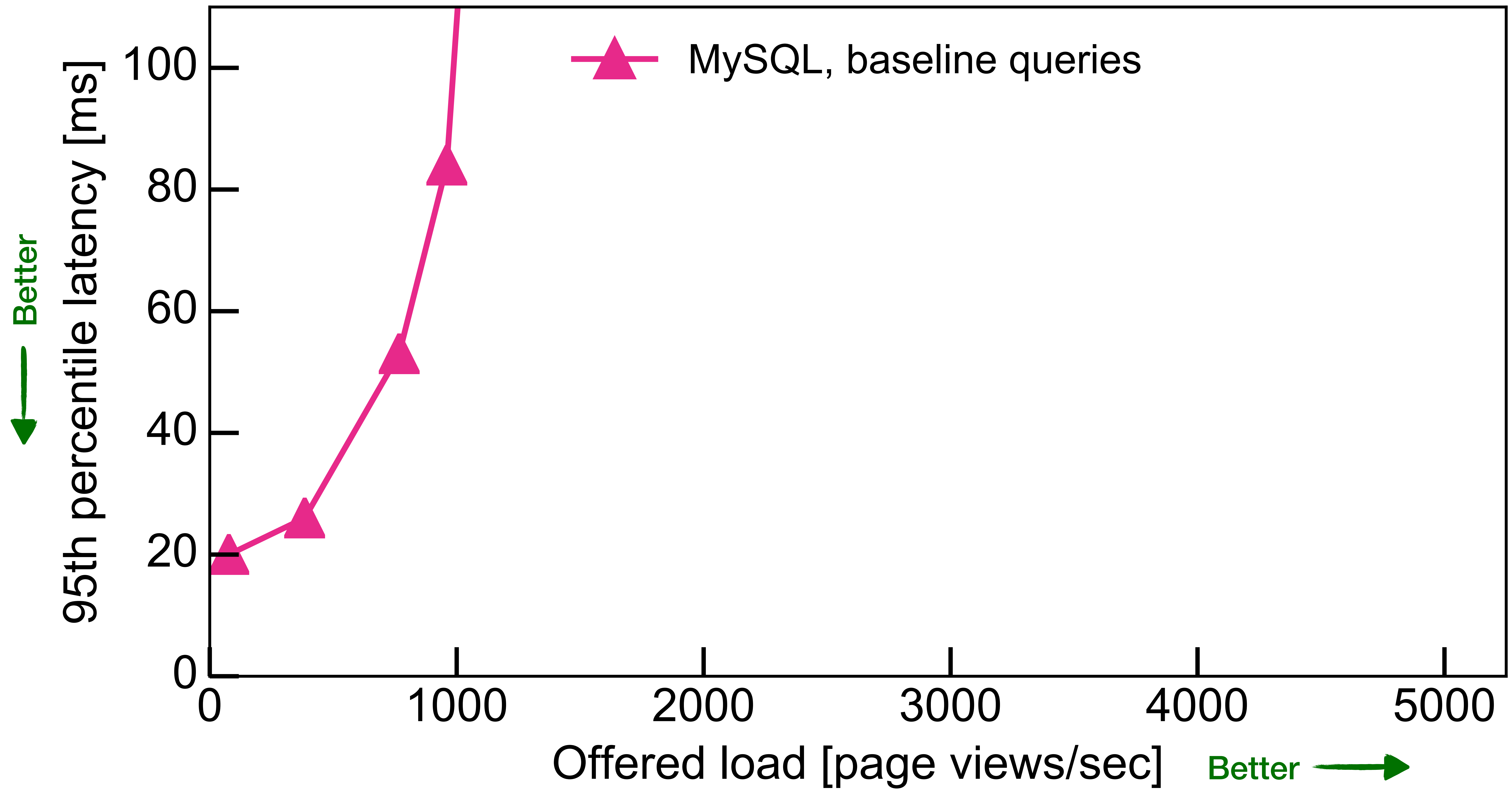
Can Noria improve Lobsters' performance?



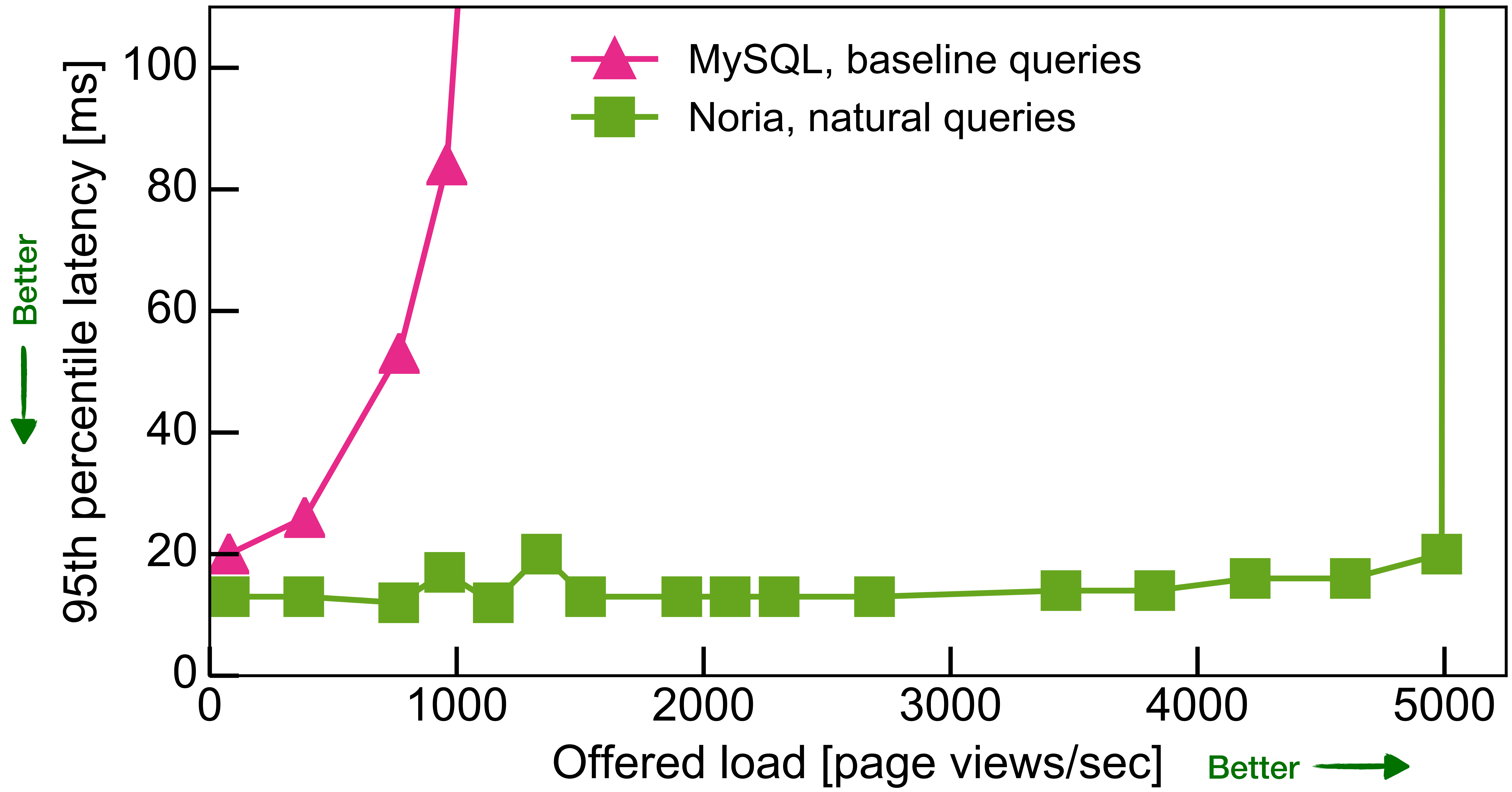
Can Noria improve Lobsters' performance?



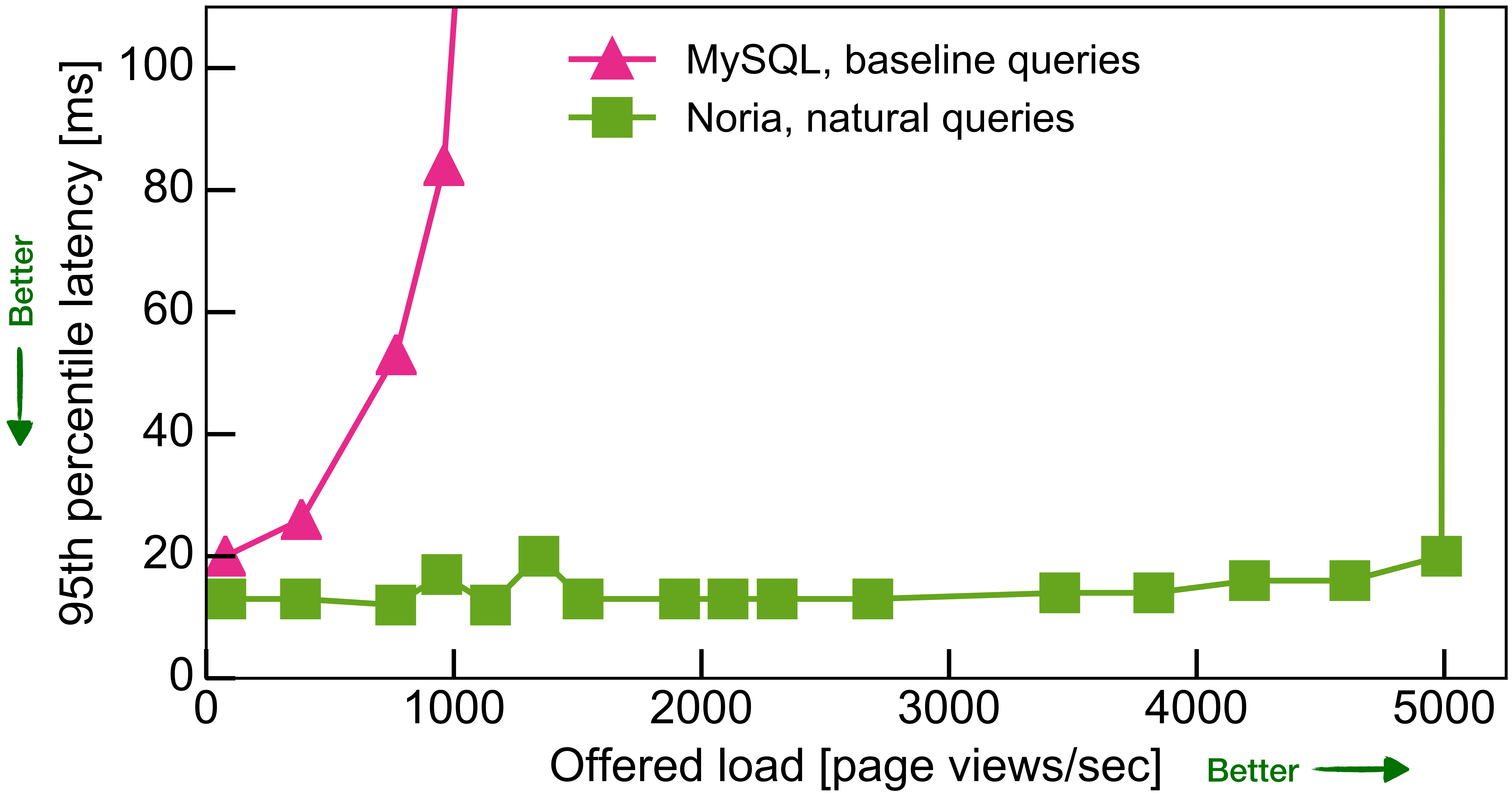
Can Noria improve Lobsters' performance?



Can Noria improve Lobsters' performance?

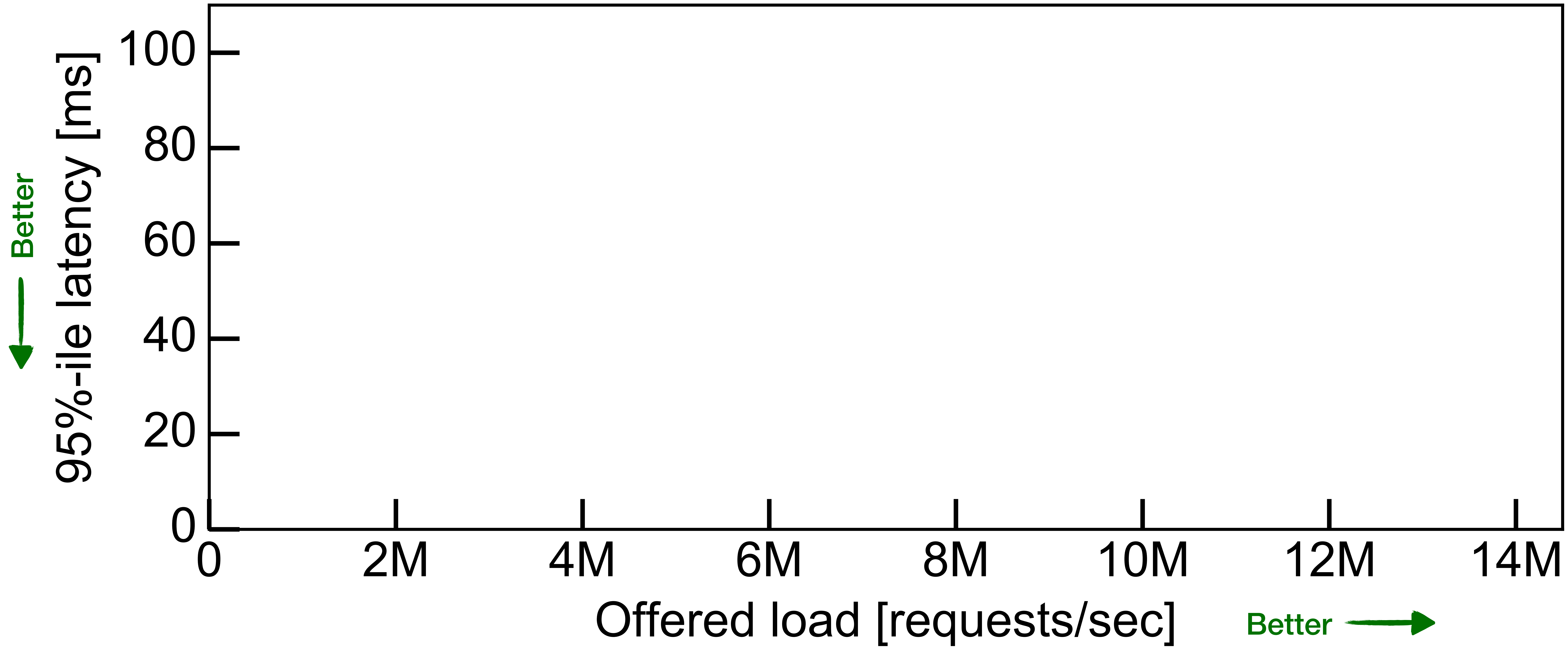


Can Noria improve Lobsters' performance?

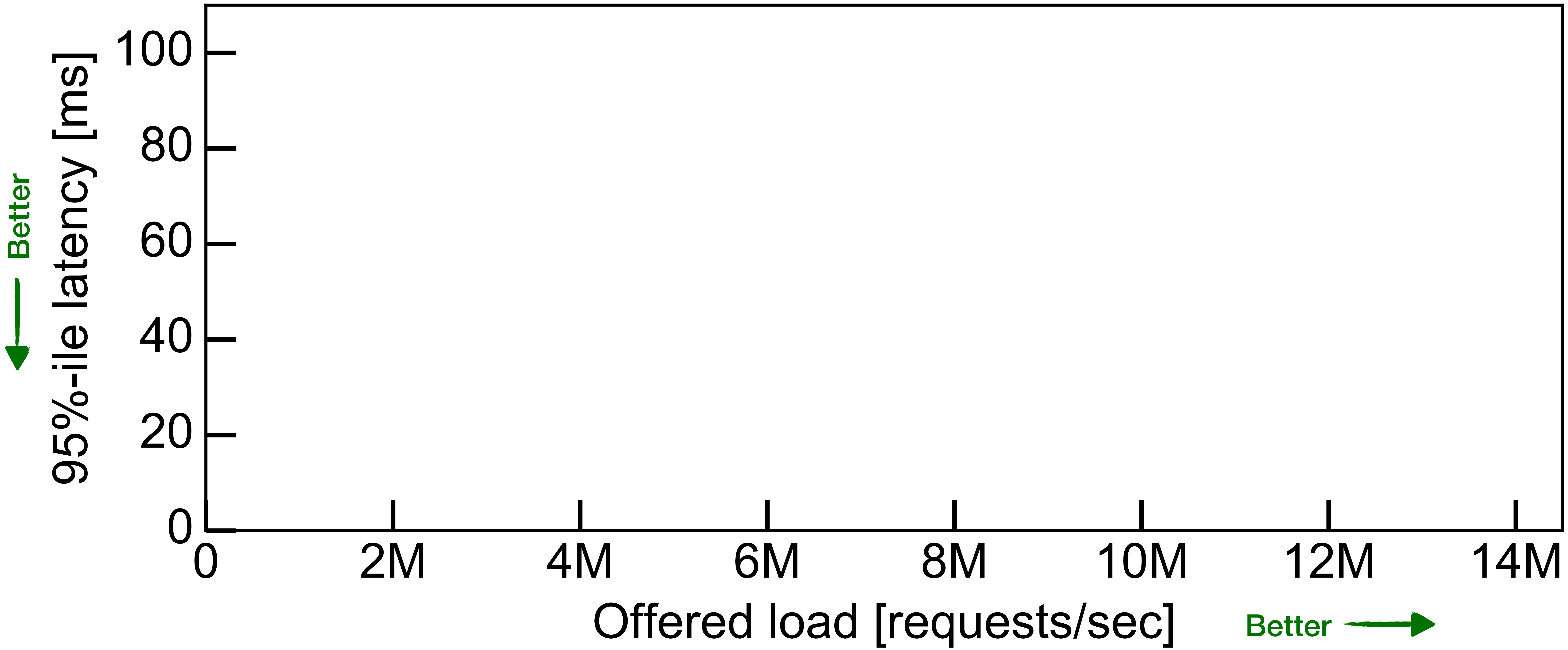


Noria with **natural queries** supports **5x** MySQL's throughput.

How does Noria compare to alternatives?

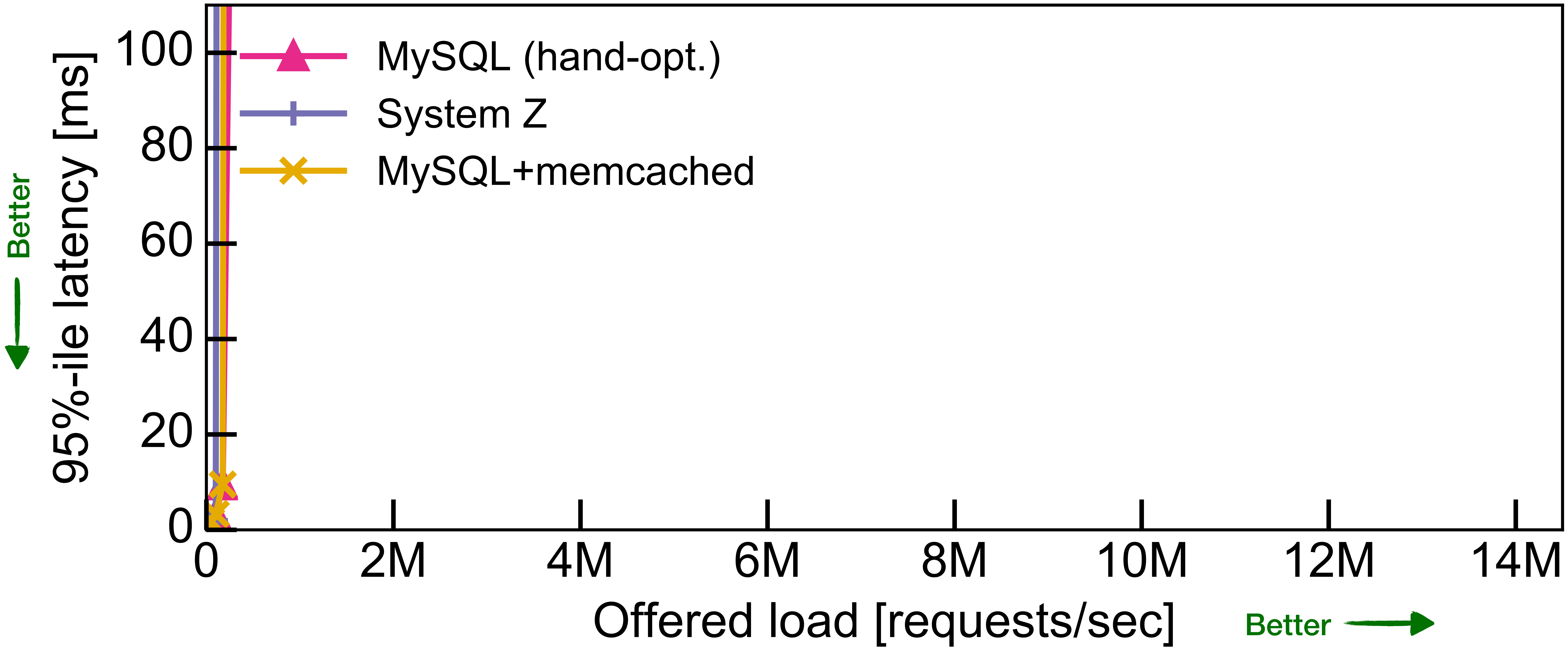


How does Noria compare to alternatives?



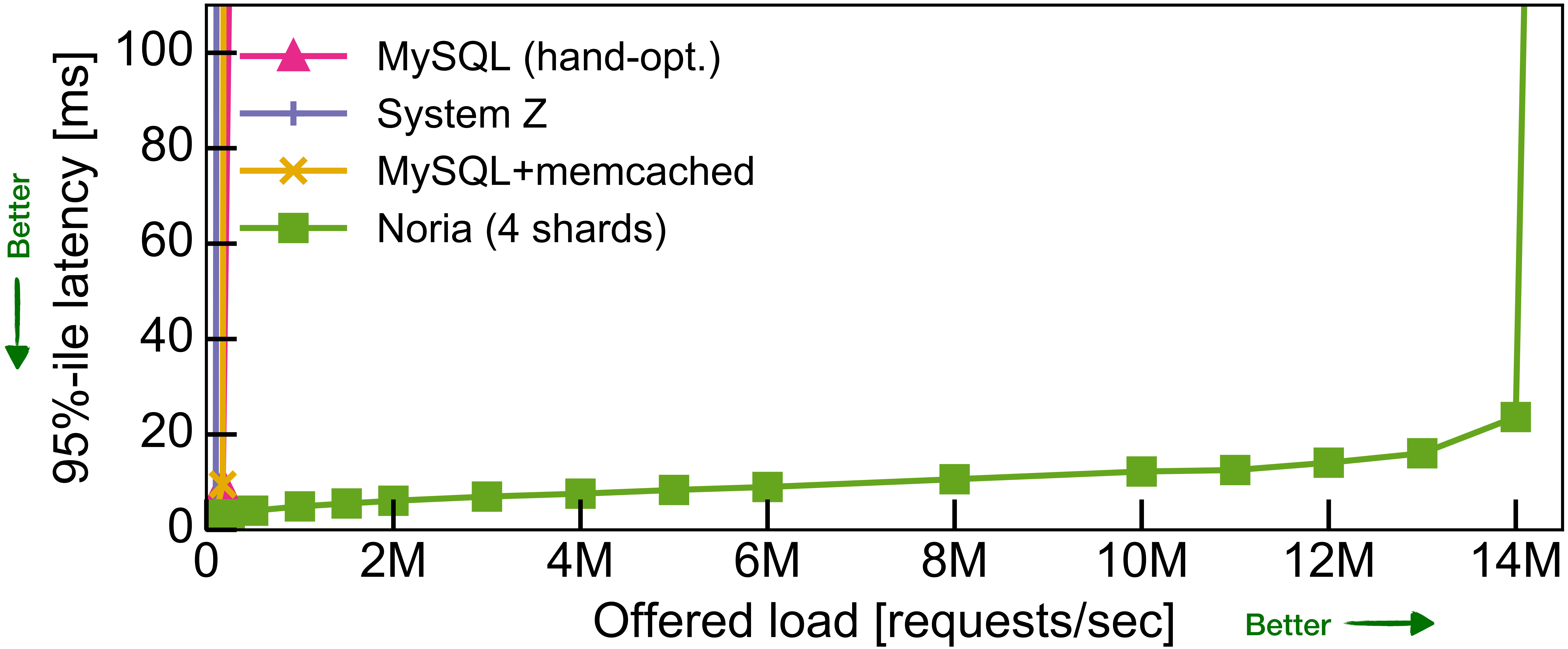
- Zipf-distributed story ID, 95% reads, 5% writes
- No TX, all in-memory

How does Noria compare to alternatives?



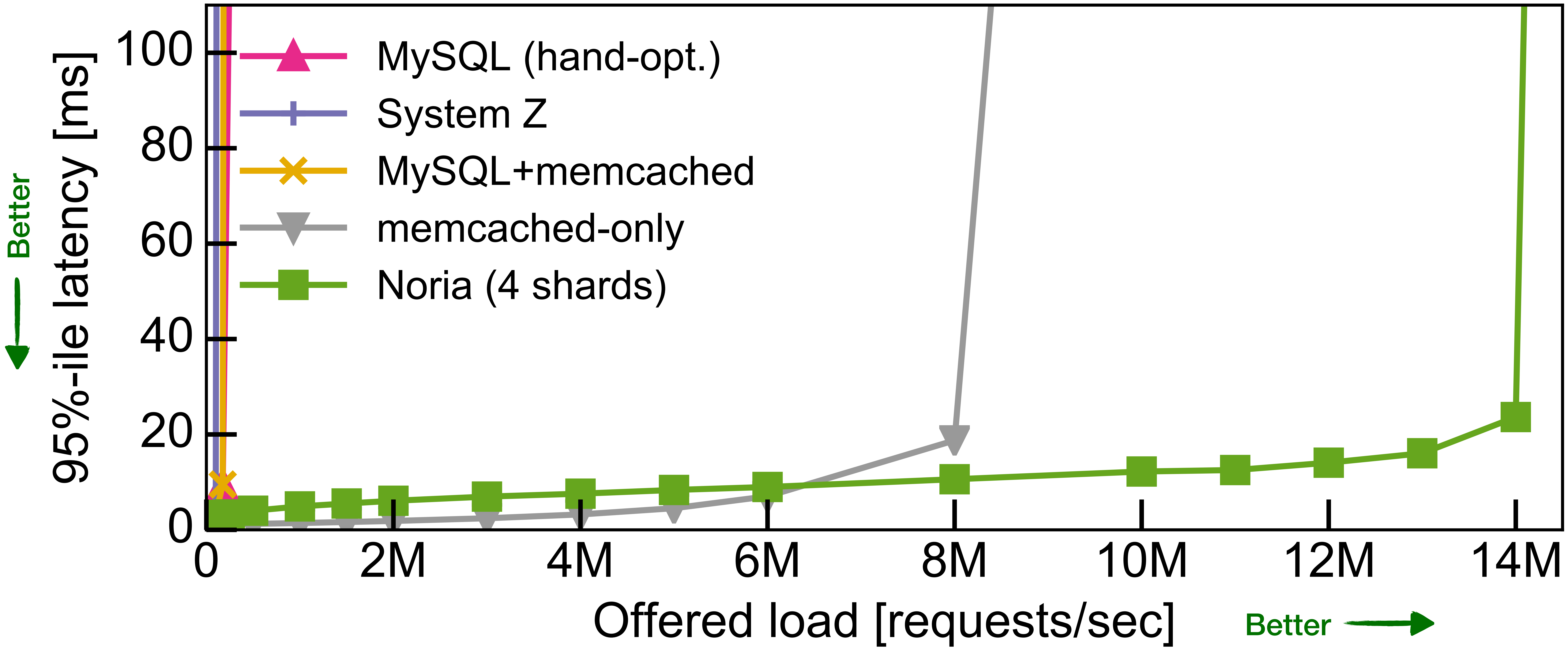
- Zipf-distributed story ID, 95% reads, 5% writes
- No TX, all in-memory

How does Noria compare to alternatives?



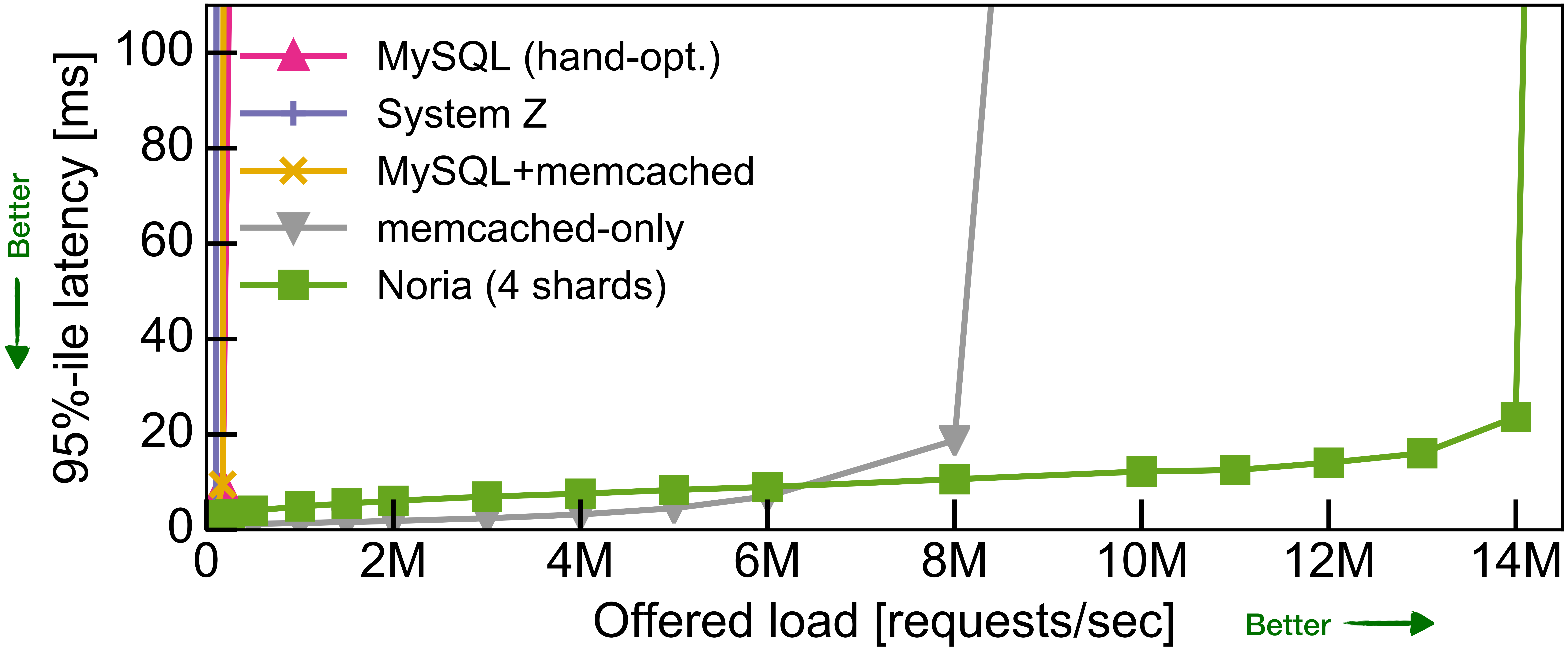
- Zipf-distributed story ID, 95% reads, 5% writes
- No TX, all in-memory

How does Noria compare to alternatives?



- Zipf-distributed story ID, 95% reads, 5% writes
- No TX, all in-memory

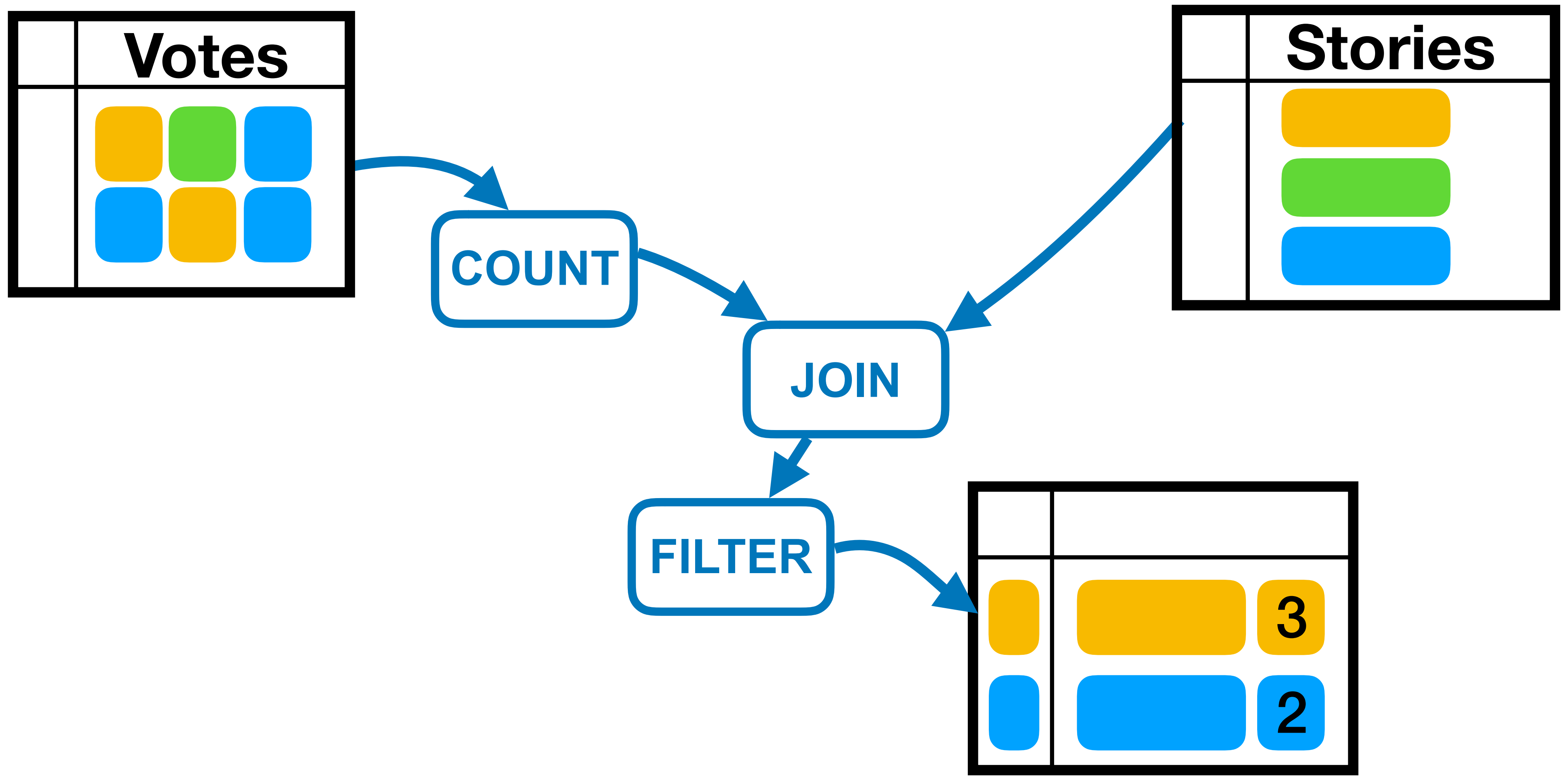
How does Noria compare to alternatives?



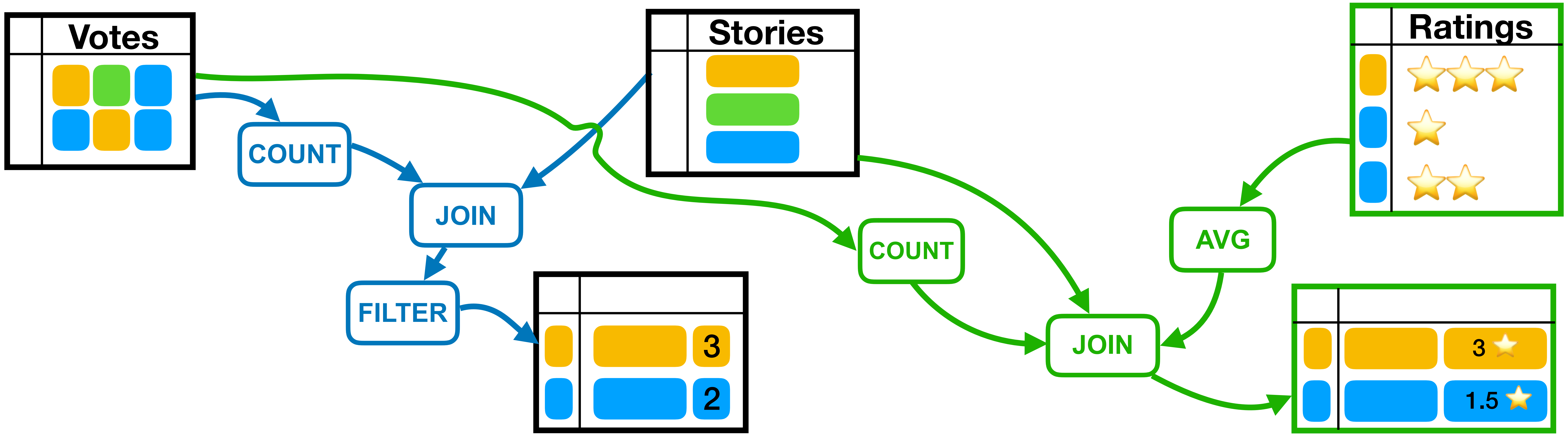
- Zipf-distributed story ID, 95% reads, 5% writes
- No TX, all in-memory

Noria outperforms an **in-memory key-value store** and simplifies its interface.

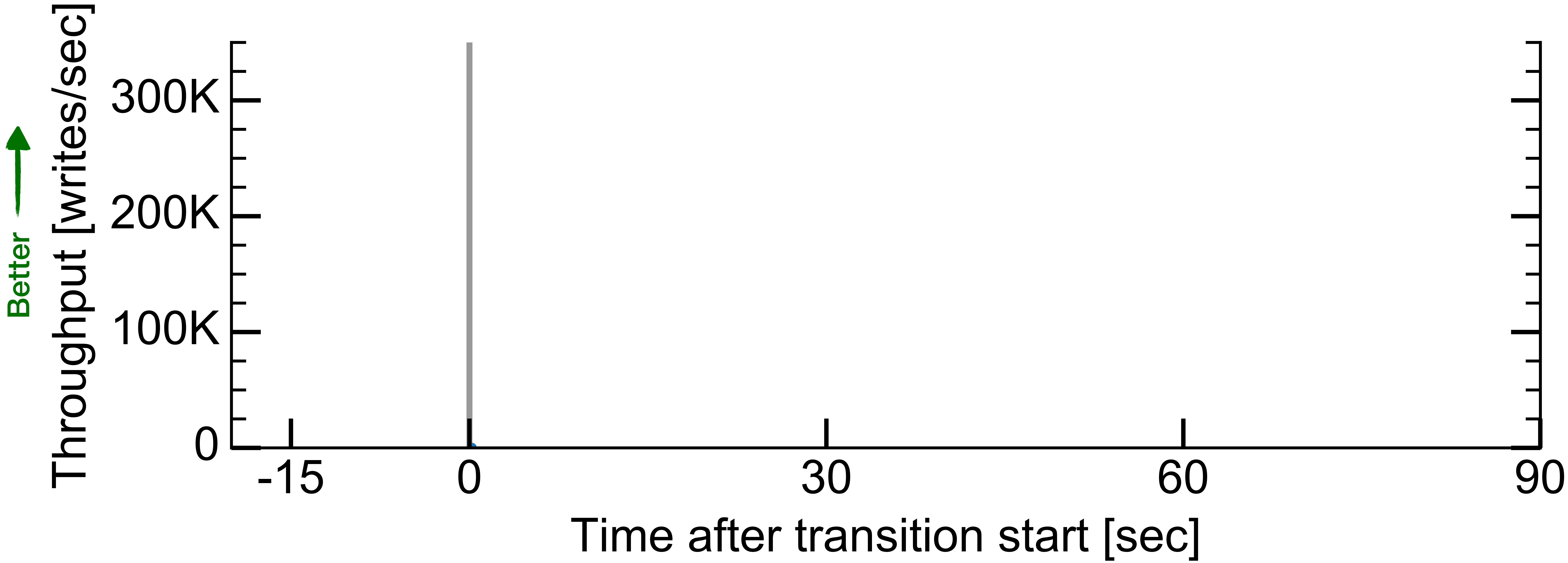
Can Noria change queries without downtime?



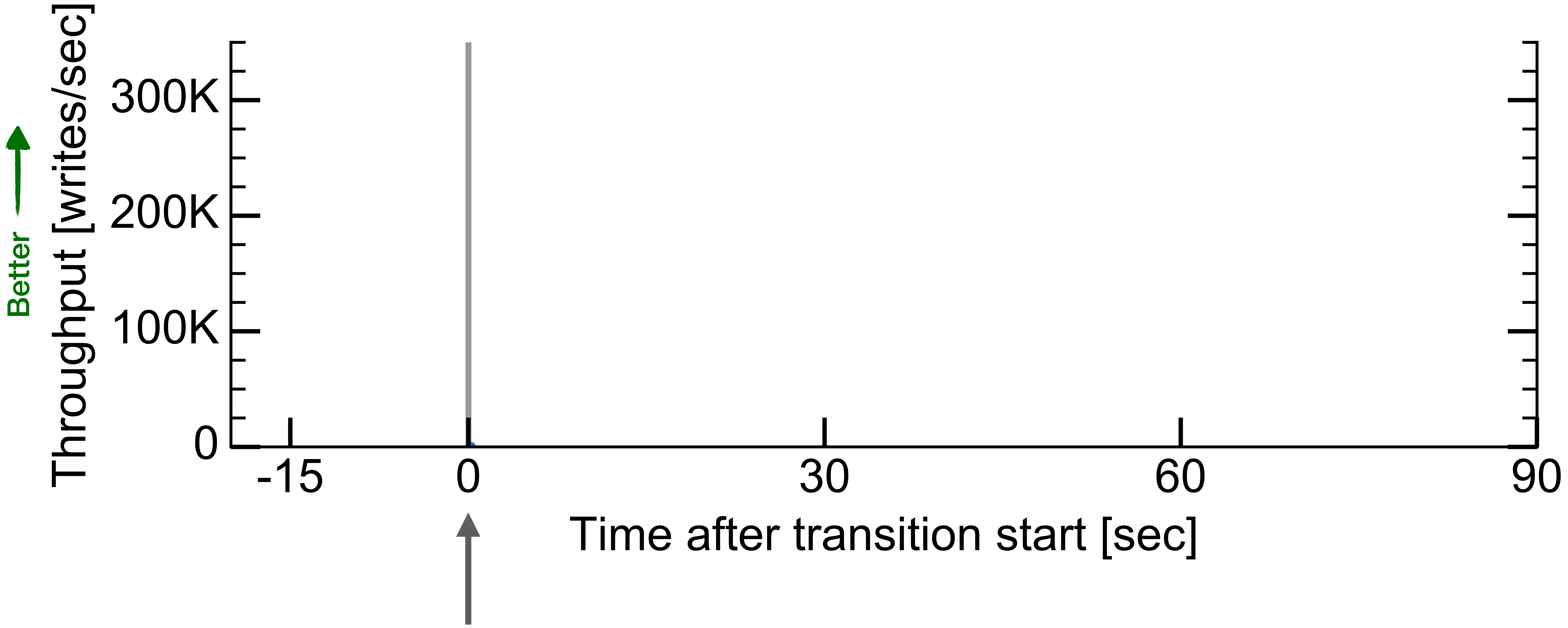
Can Noria change queries without downtime?



Can Noria change queries without downtime?

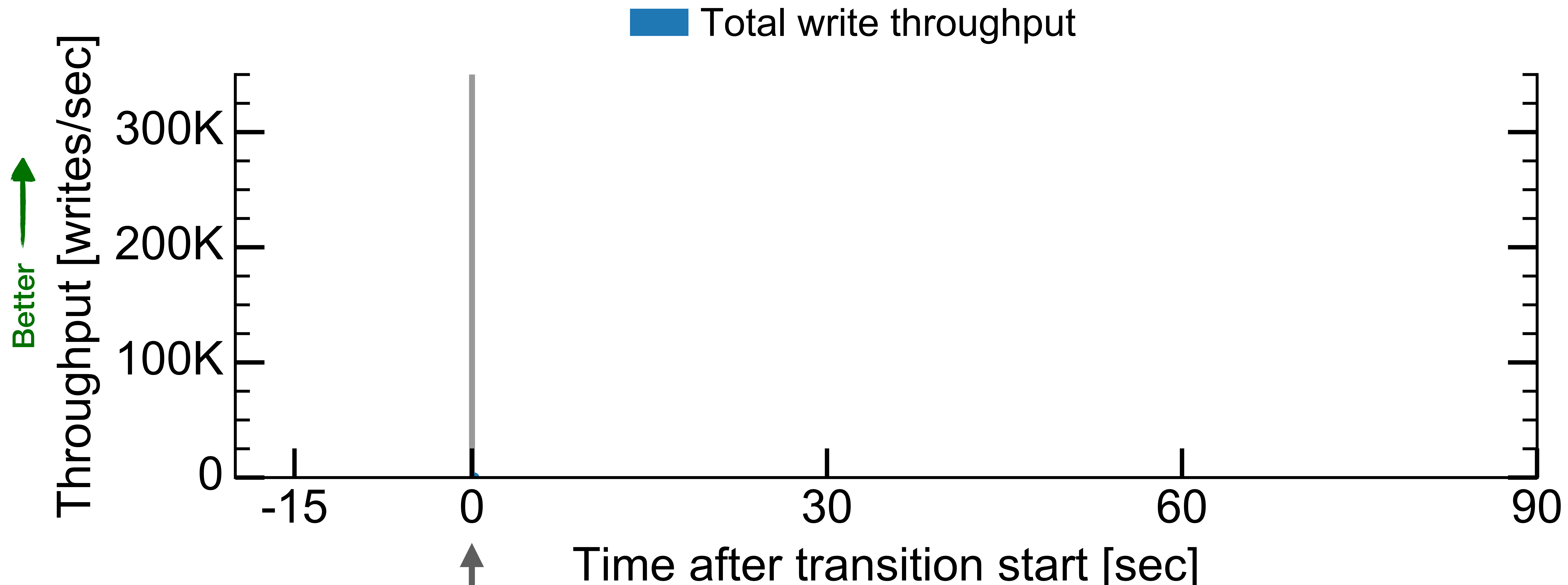


Can Noria change queries without downtime?



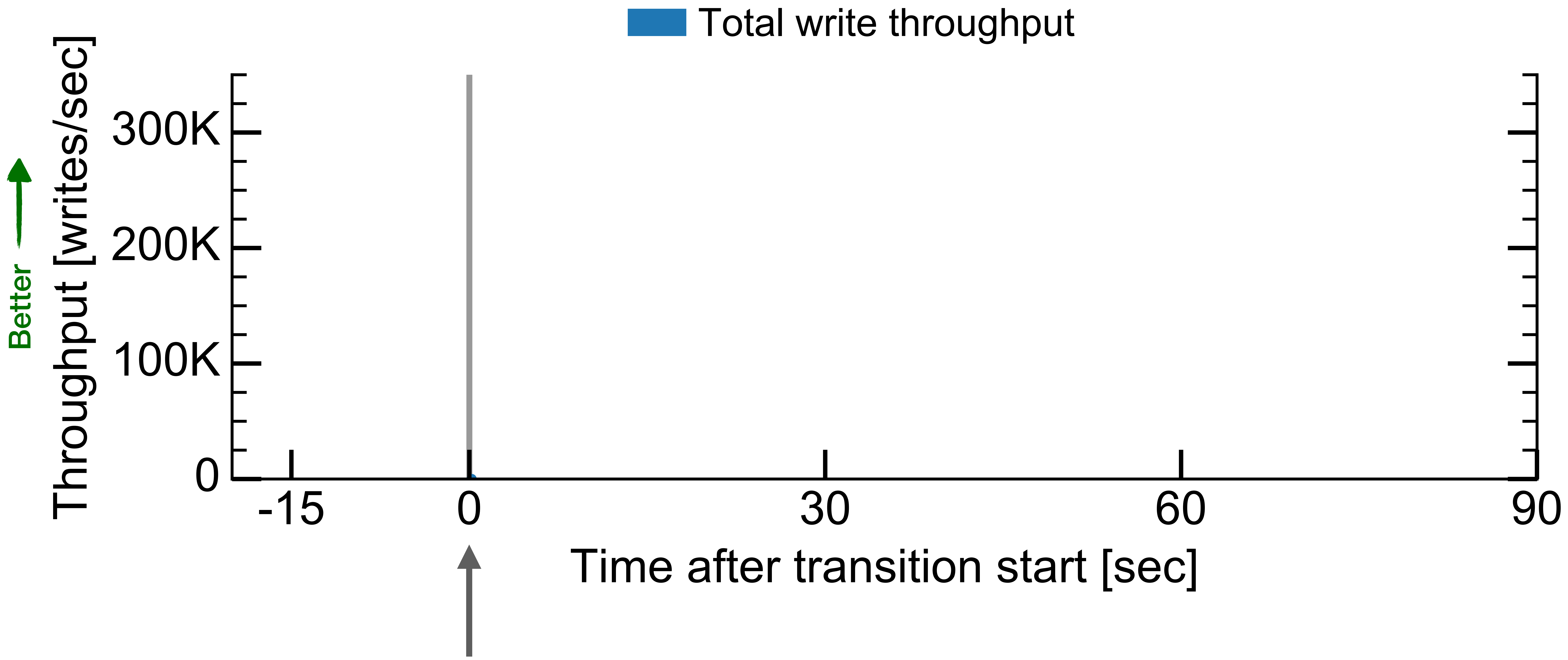
new table & query added

Can Noria change queries without downtime?



new table & query added

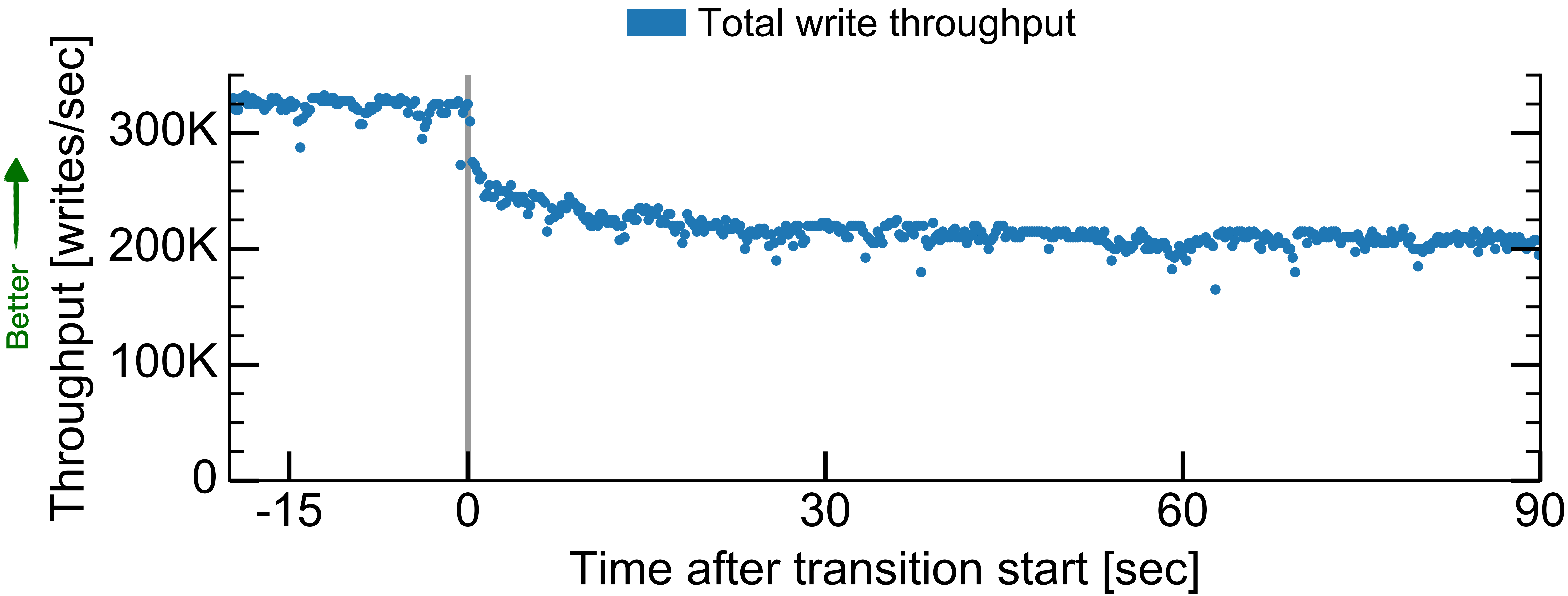
Can Noria change queries without downtime?



new table & query added

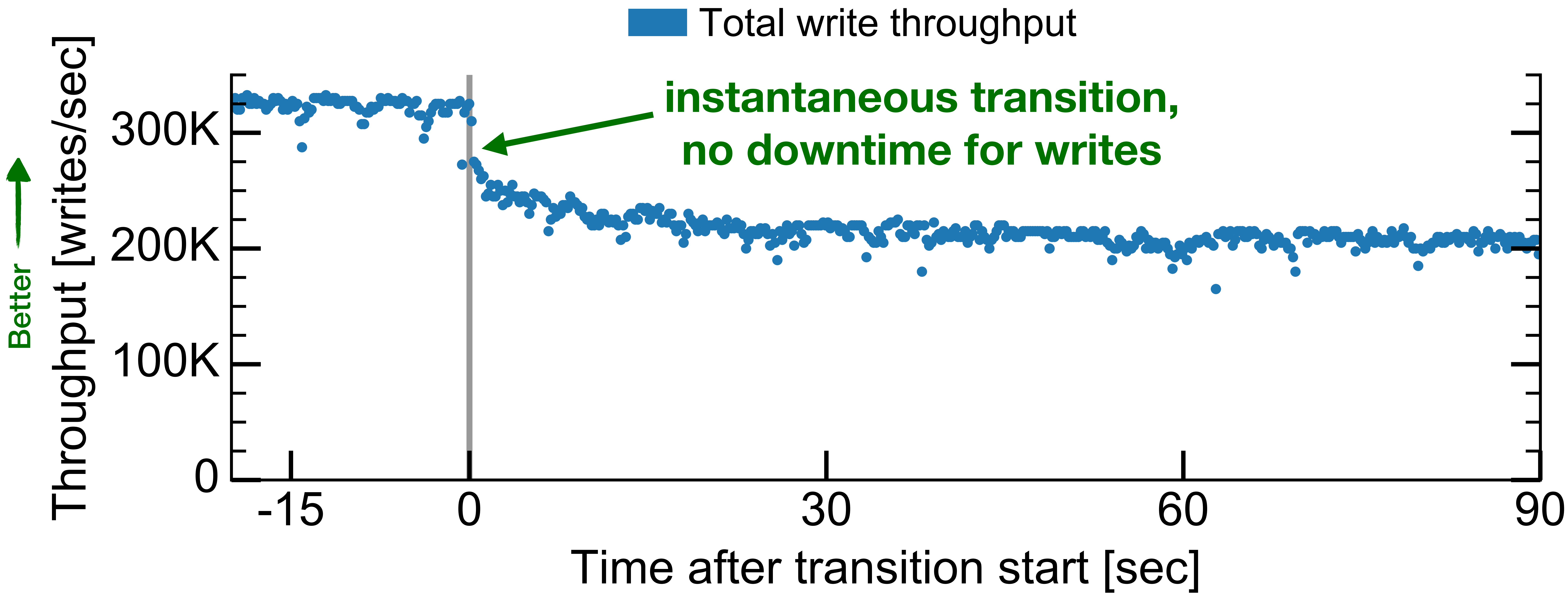
- Zipf-distributed story ID, 95% reads; 2M existing votes at transition

Can Noria change queries without downtime?



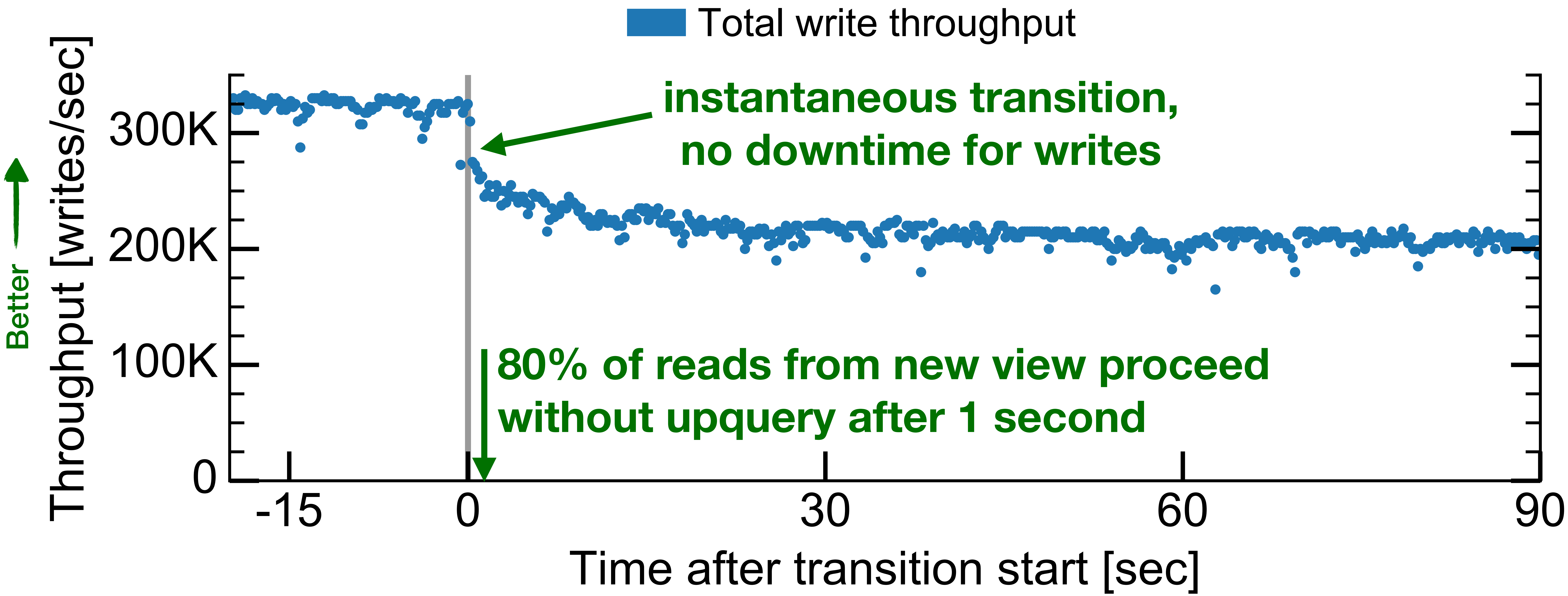
- Zipf-distributed story ID, 95% reads; 2M existing votes at transition
- Old view reads are live throughout

Can Noria change queries without downtime?



- Zipf-distributed story ID, 95% reads; 2M existing votes at transition
- Old view reads are live throughout

Can Noria change queries without downtime?

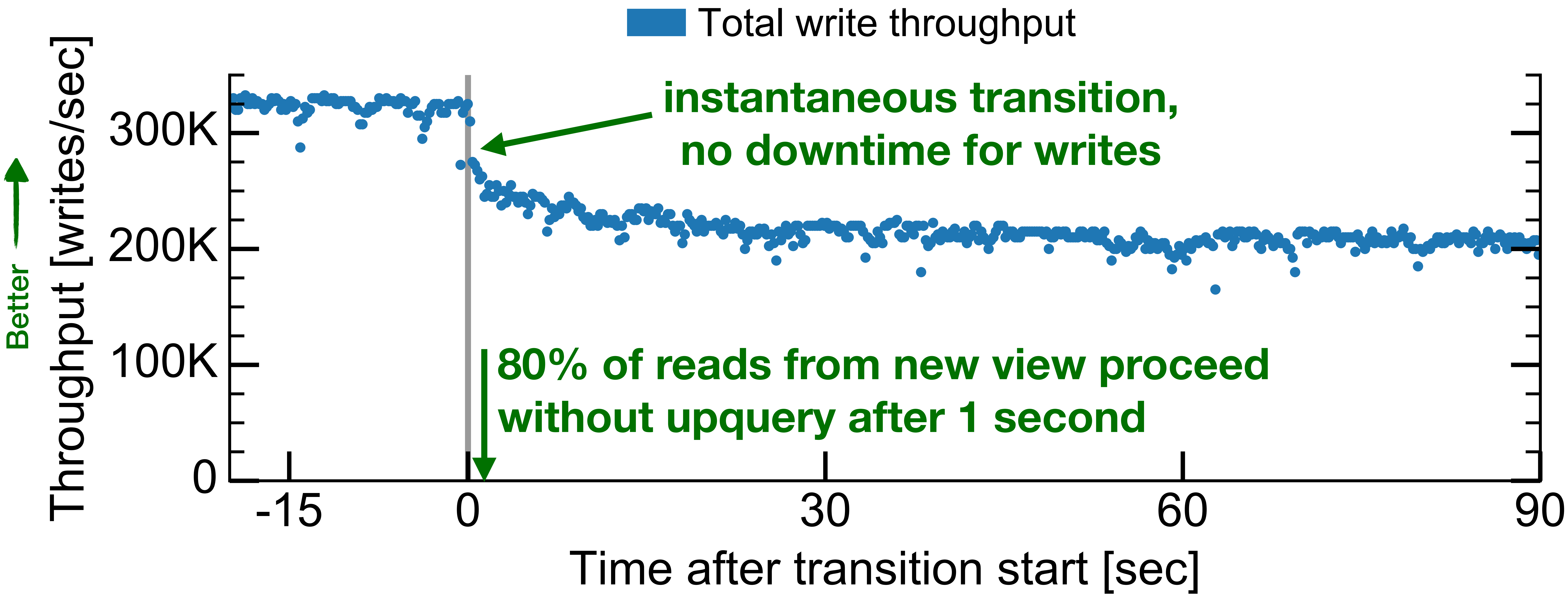


instantaneous transition, no downtime for writes

80% of reads from new view proceed without upquery after 1 second

- Zipf-distributed story ID, 95% reads; 2M existing votes at transition
- Old view reads are live throughout

Can Noria change queries without downtime?



instantaneous transition, no downtime for writes

80% of reads from new view proceed without upquery after 1 second

Noria **achieves downtime-free query change** with partial state.

- Zipf-distributed story ID, 95% reads; 2M existing votes at transition
- Old view reads are live throughout

Noria — Summary

- New partially-stateful data-flow model.
- Noria: new web application backend based on data-flow.
- Partial state saves space and allows live change.
- Supports high throughput on one or more machines.
- Open source, try it out!

Noria — Summary

- New partially-stateful data-flow model.
- Noria: new web application backend based on data-flow.
- Partial state saves space and allows live change.
- Supports high throughput on one or more machines.
- Open source, try it out!

`https://pdos.csail.mit.edu/noria`

(see our demo at poster #37 today!)