Operational hazards of managing PostgreSQL DBs over 100TB

adyen

engineered
for ambition

Why?

Storage Limits

AWS RDS	64TB
Google CloudSQL	64TB
Azure Database Flexible Server	64TB

Storage Limits

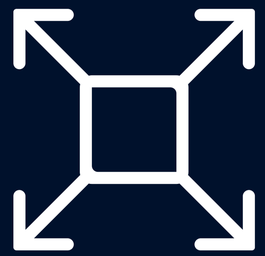
AWS Aurora	256TB
Google AlloyDB	128TB

Teresa Lopes

Database Engineer

Adyen

Agenda



Big, but how big?

Bring balance to the Database

Vacuum, it is always Vacuum

To backup or not To backup

Data, Data everywhere

Big, but how big?

Small DB + Low TPS

Large DB + Low TPS

Small DB + High TPS

Small DB + Strict Latency

Source: https://www.redbull.com/

Perfect Storm

Perfect Storm

Massive DB

Perfect Storm

Massive DB
+ Very High TPS

Perfect Storm

Massive DB

+ Very High TPS

+ Strict Latency

Perfect Storm

Massive DB

+ Very High TPS

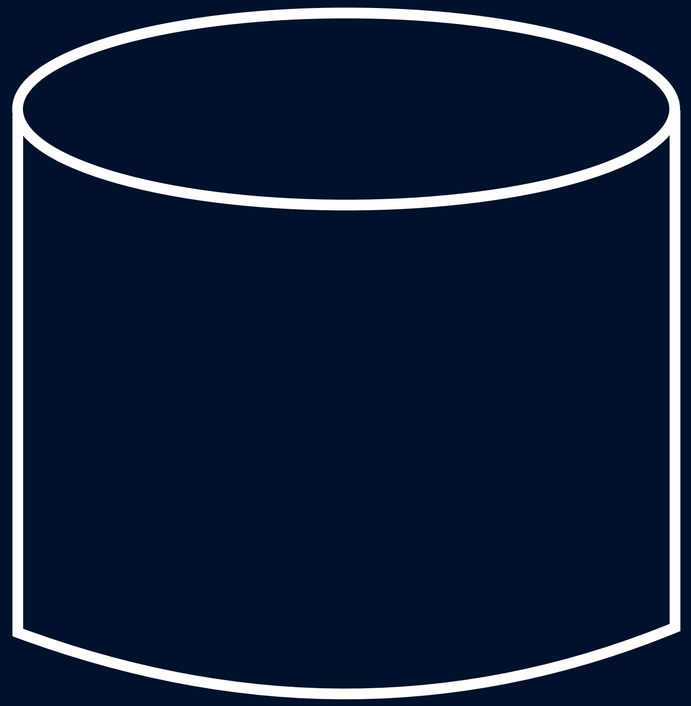
+ Strict Latency

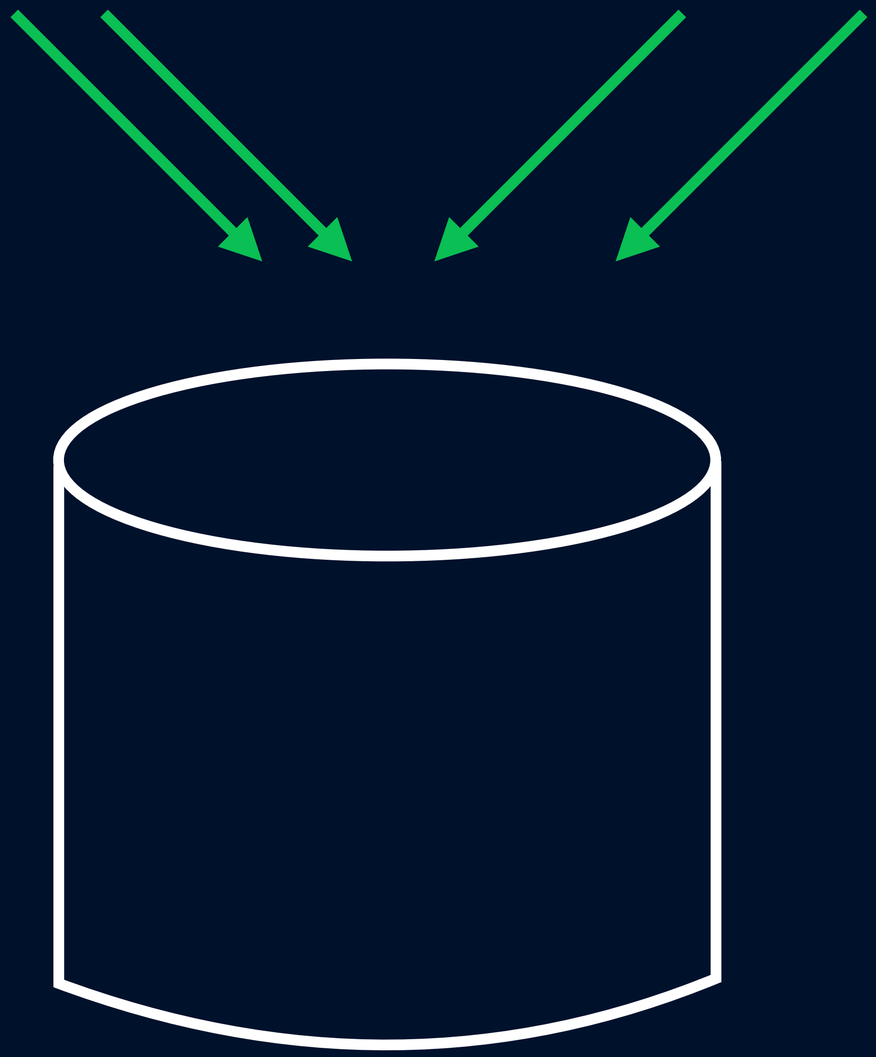
+ Hot Data

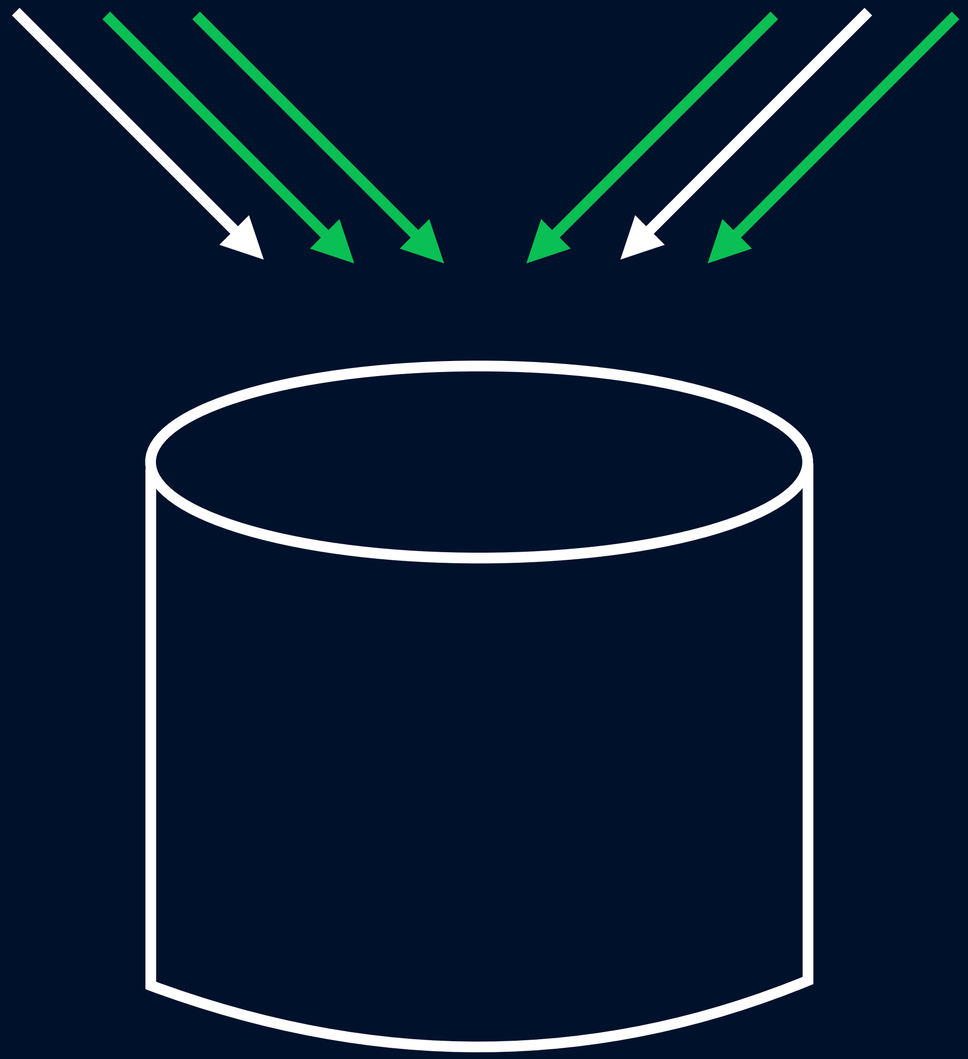
Perfect Storm

Bring balance to the Database

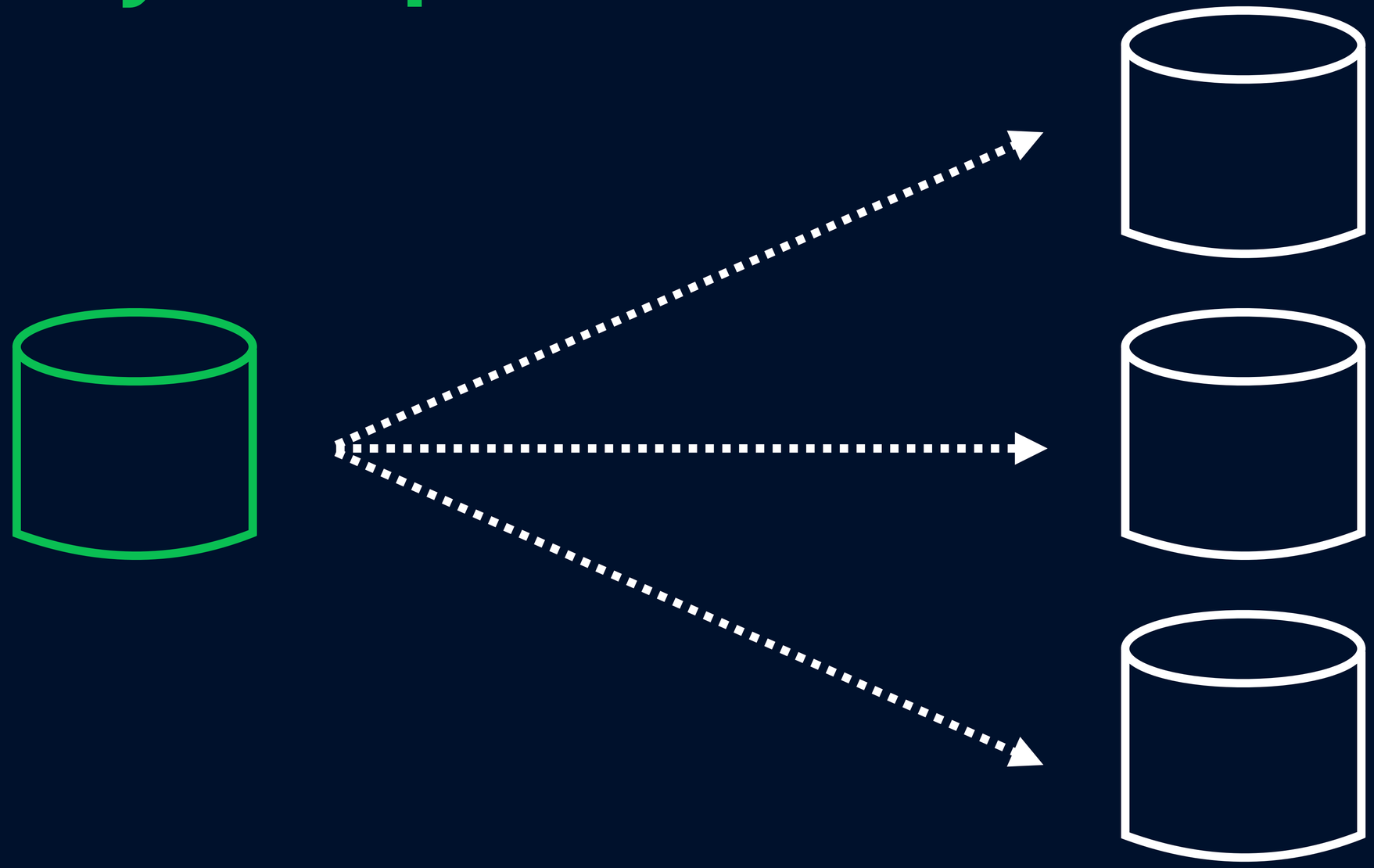
Single Instance



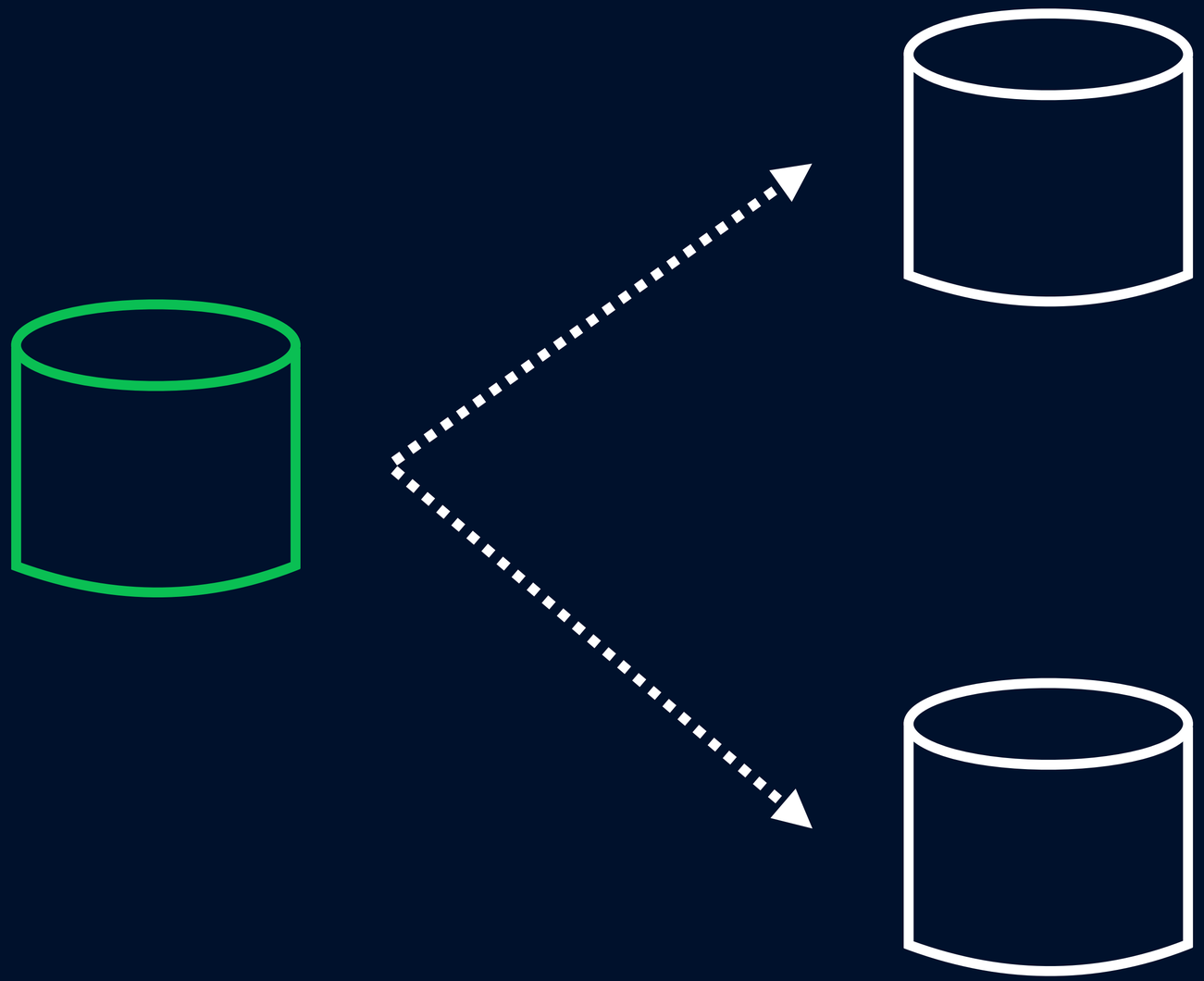




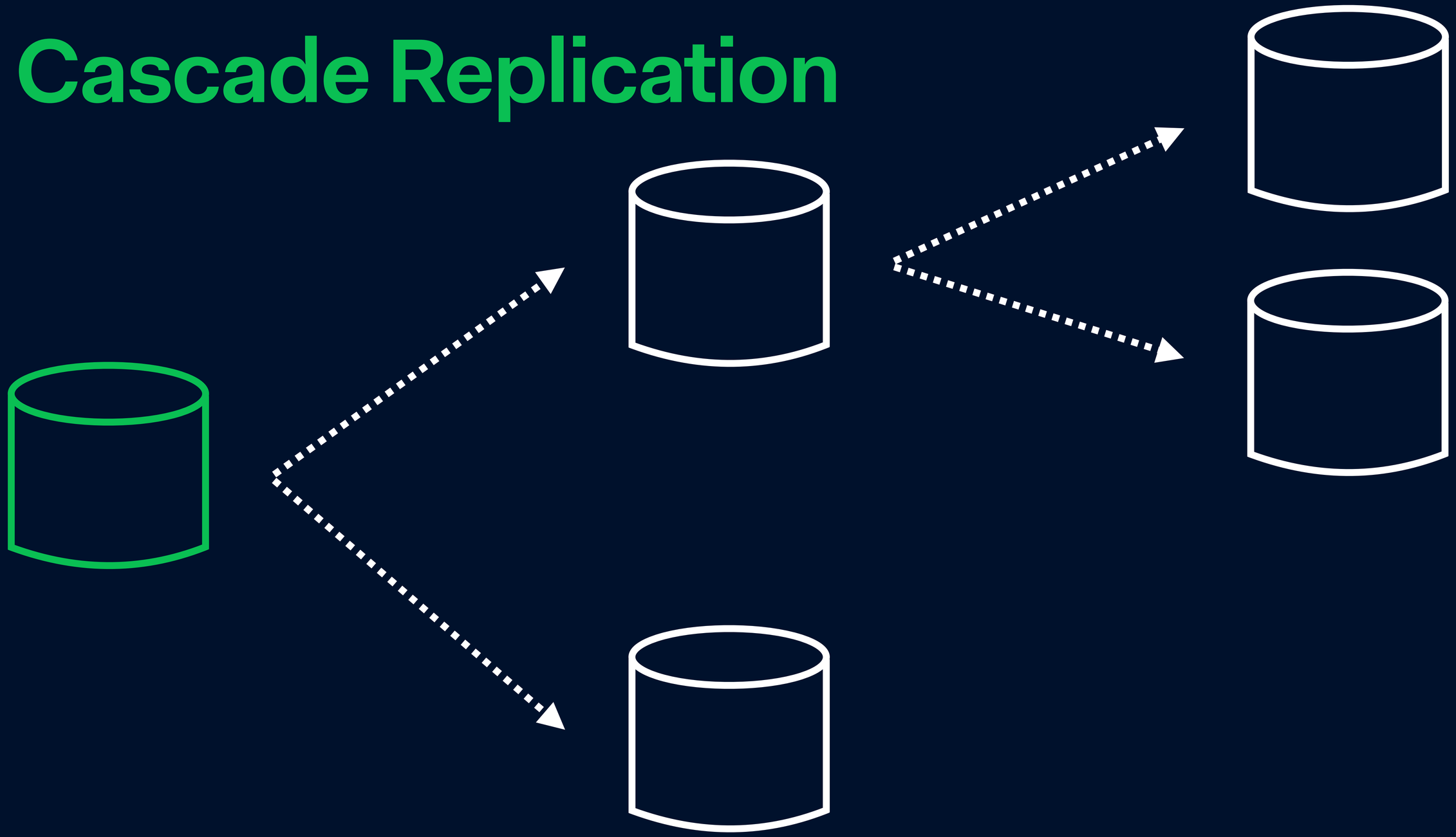
Primary + Replicas



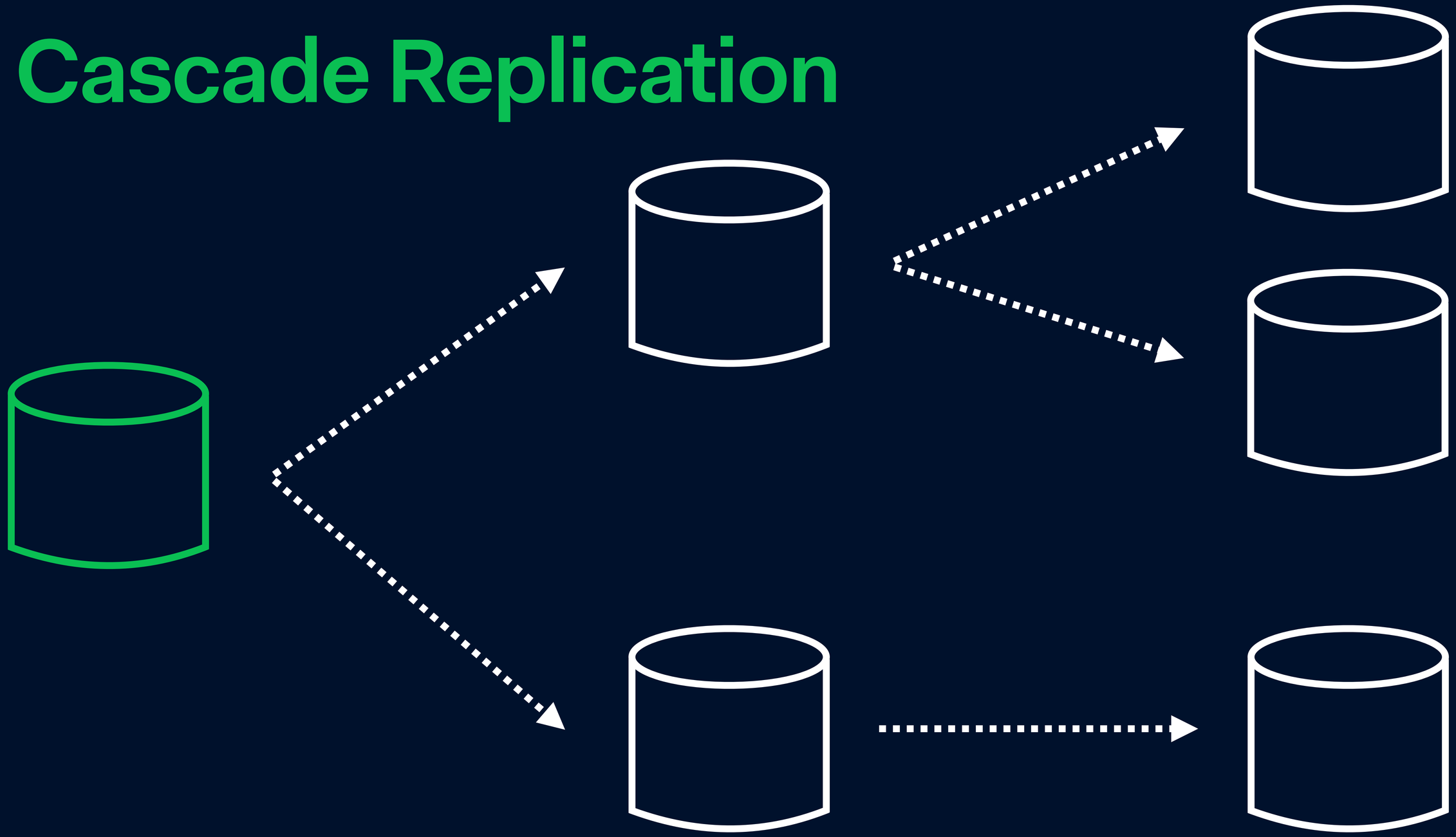
Cascade Replication



Cascade Replication



Cascade Replication



Replication

Replication

PGConf.DE 2025

Myths and Truths about Synchronous Replication in PostgreSQL
Fri. 11:10–11:55 — Berlin 1

Alexander Kukushkin

PGConf.EU 2025

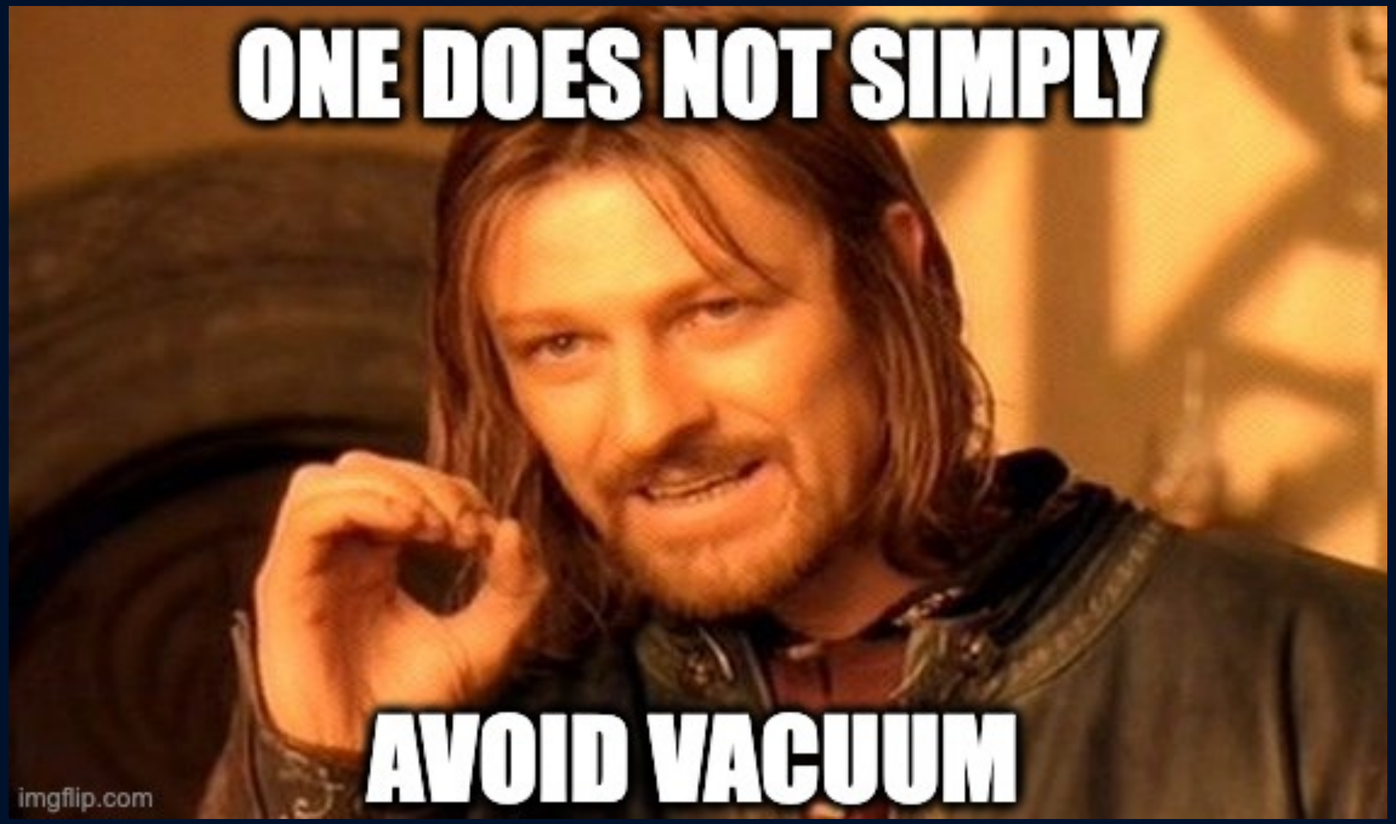
The SyncRep Detective Story: Chasing Ghosts in PostgreSQL, Finding Demons in Storage

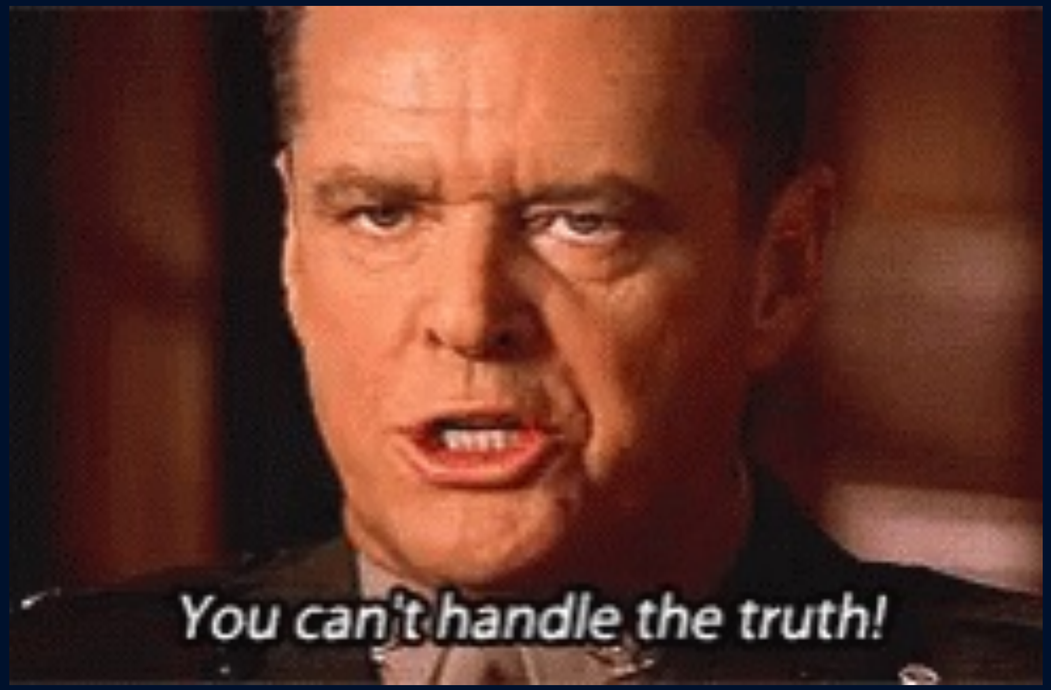
Thursday, October 23 at 11:25–12:15

Dmitry Fomin

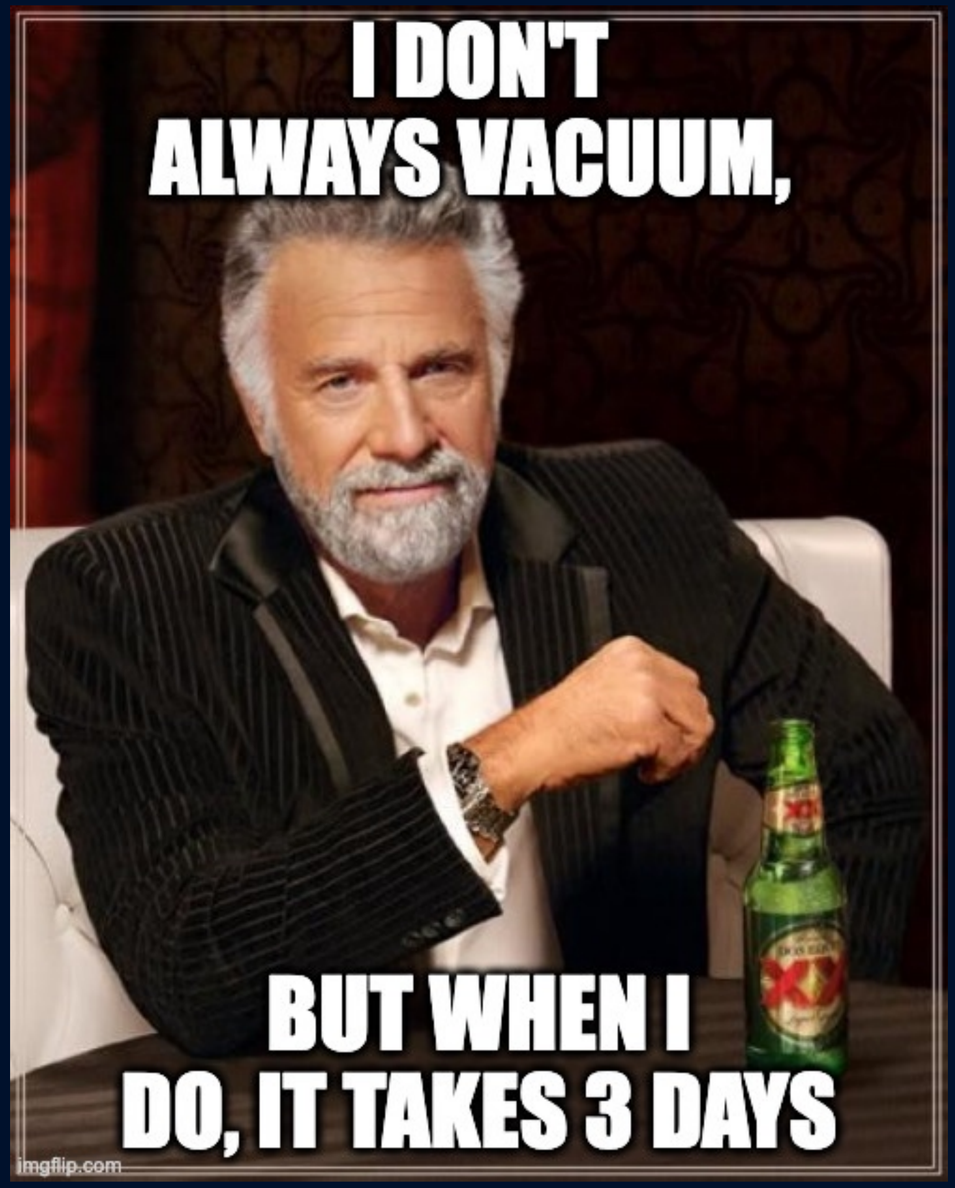
Adyen

Vacuum, it is always Vacuum...


ONE DOES NOT SIMPLY
AVOID VACUUM
imgflip.com


You can't handle the truth!

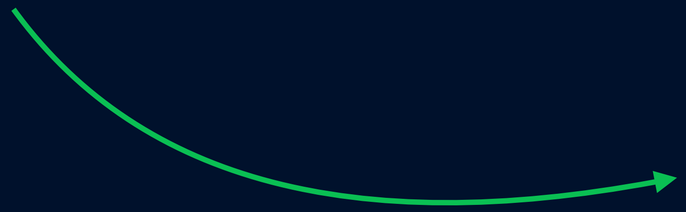
Why?


I DON'T
ALWAYS VACUUM,
BUT WHEN I
DO, IT TAKES 3 DAYS
imgflip.com

How?

Autovacuum 101

Autovacuum 101



```
autovacuum_vacuum_threshold

autovacuum_vacuum_scale_factor

autovacuum_work_mem (or maintenance_work_mem)

autovacuum_analyze_threshold

autovacuum_analyze_scale_factor

autovacuum_max_workers

log_autovacuum_min_duration
```

Autovacuum

Difficulty Level: HARD

Autovacuum
Difficulty Level: HARD

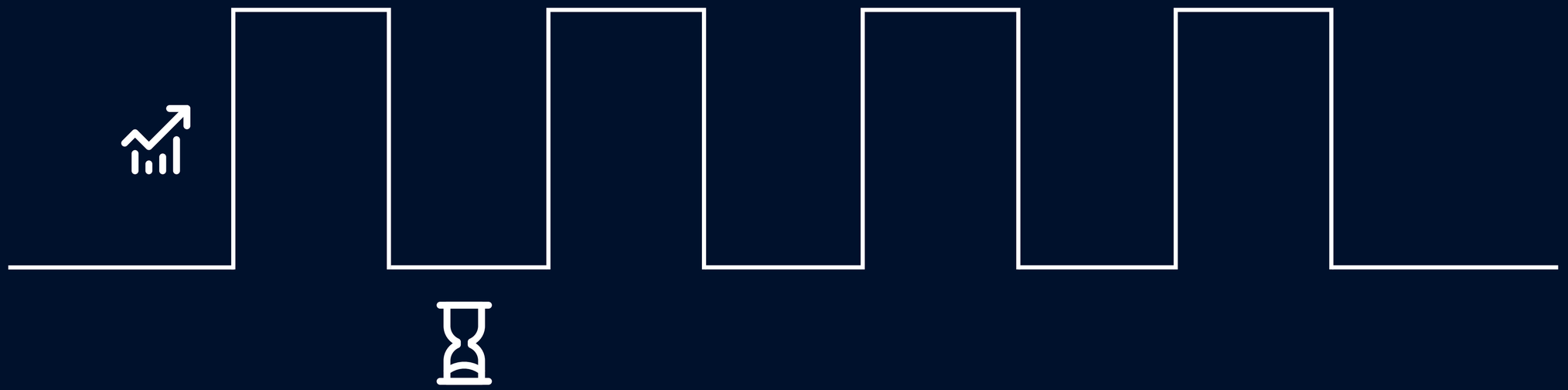
cost

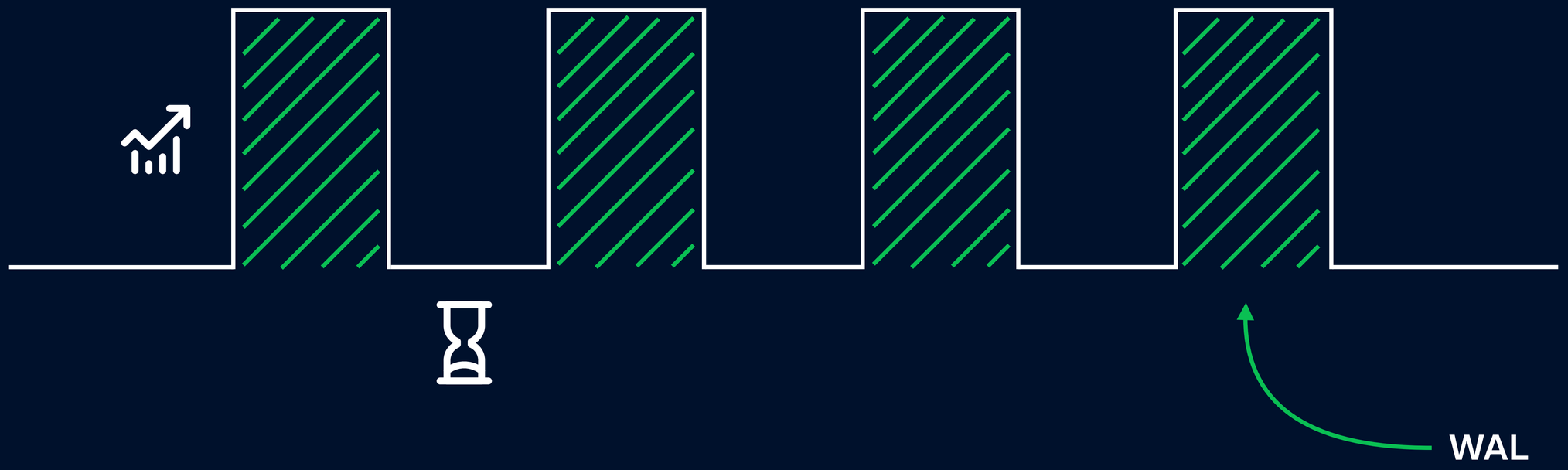
Autovacuum
Difficulty Level: HARD

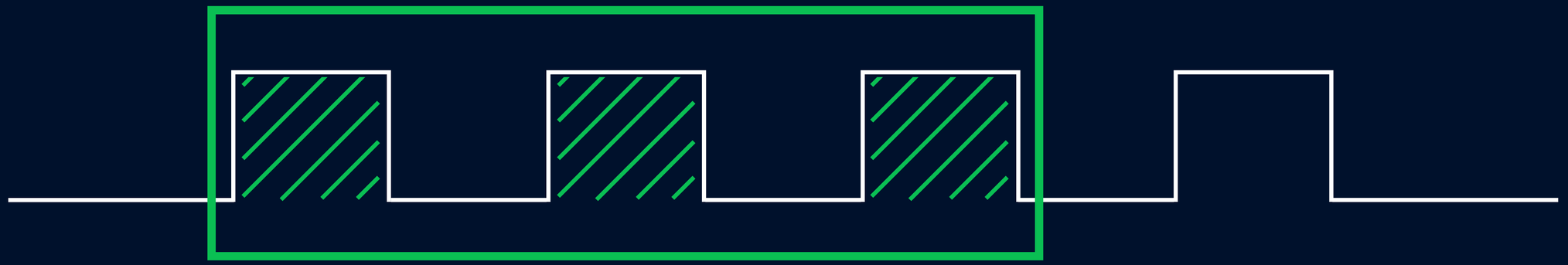
cost

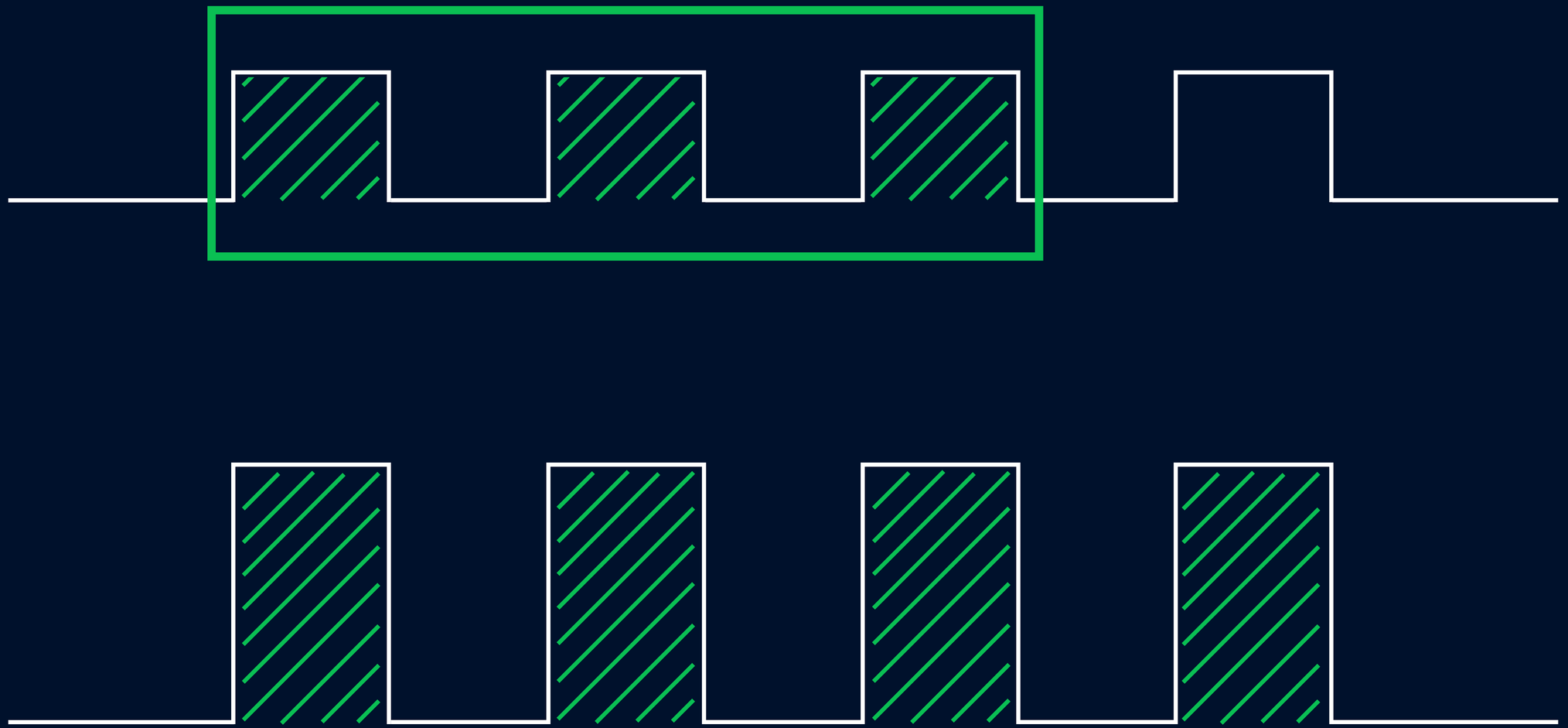
vs

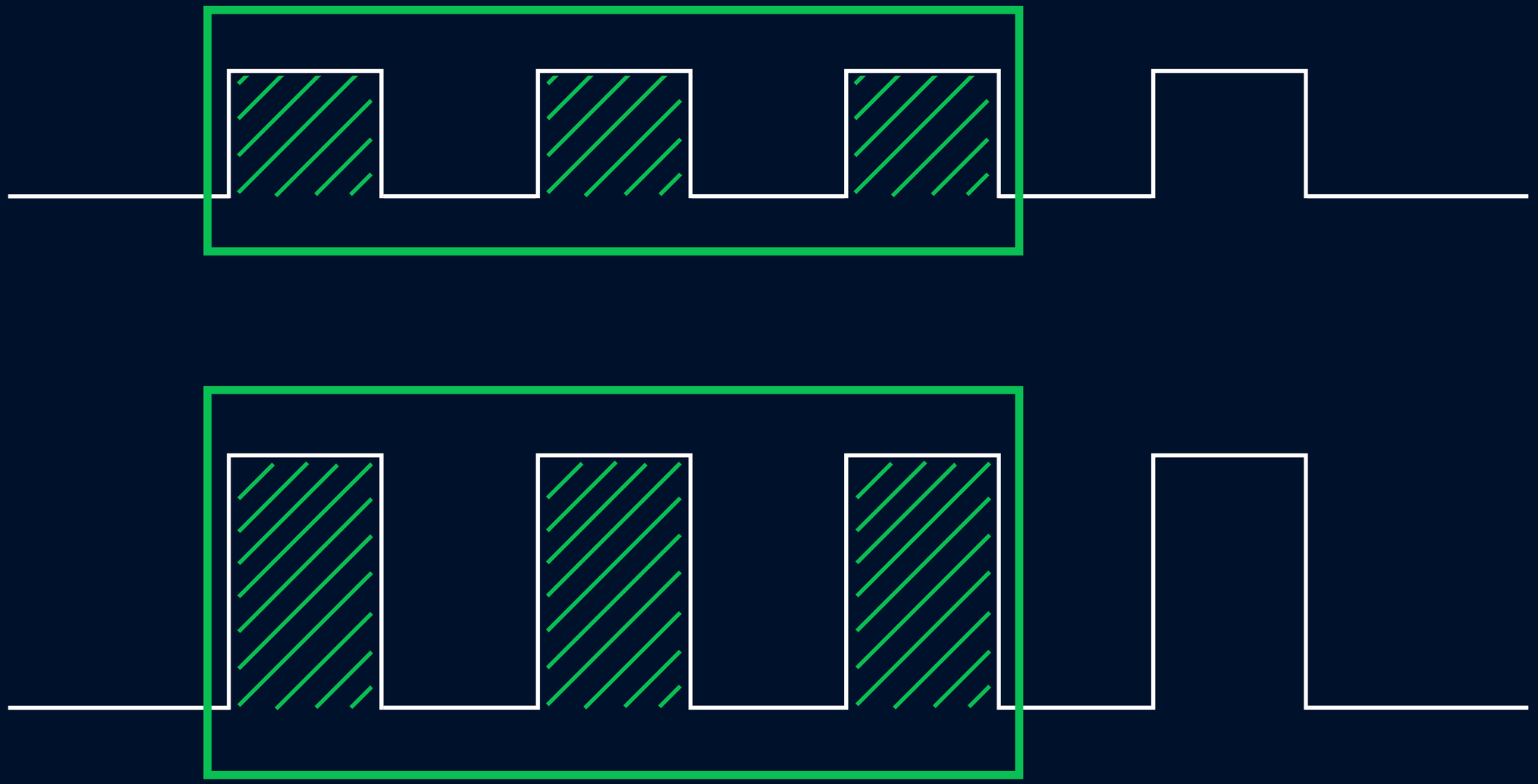
delay

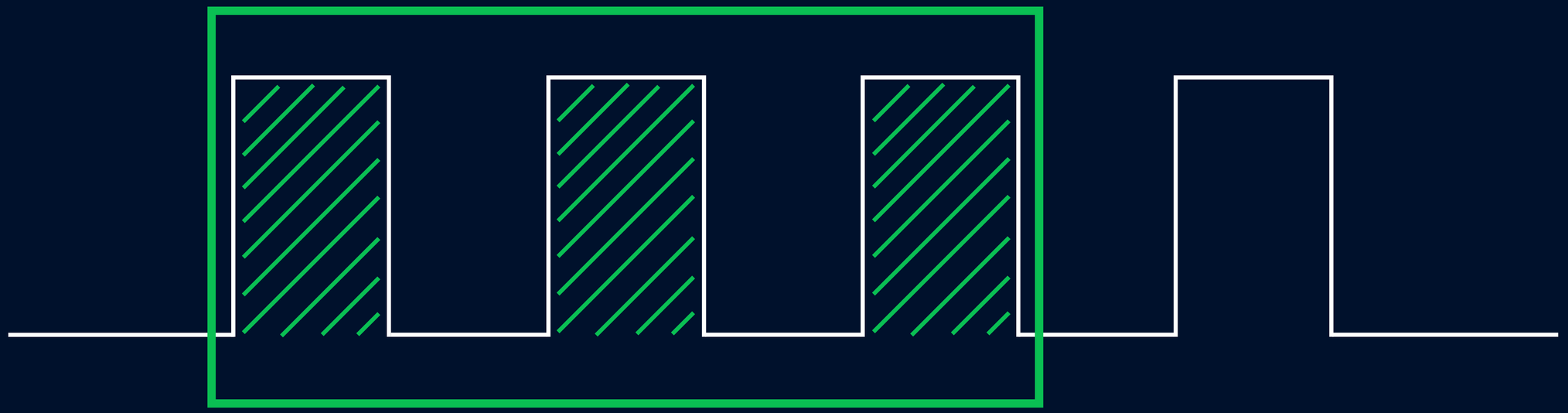


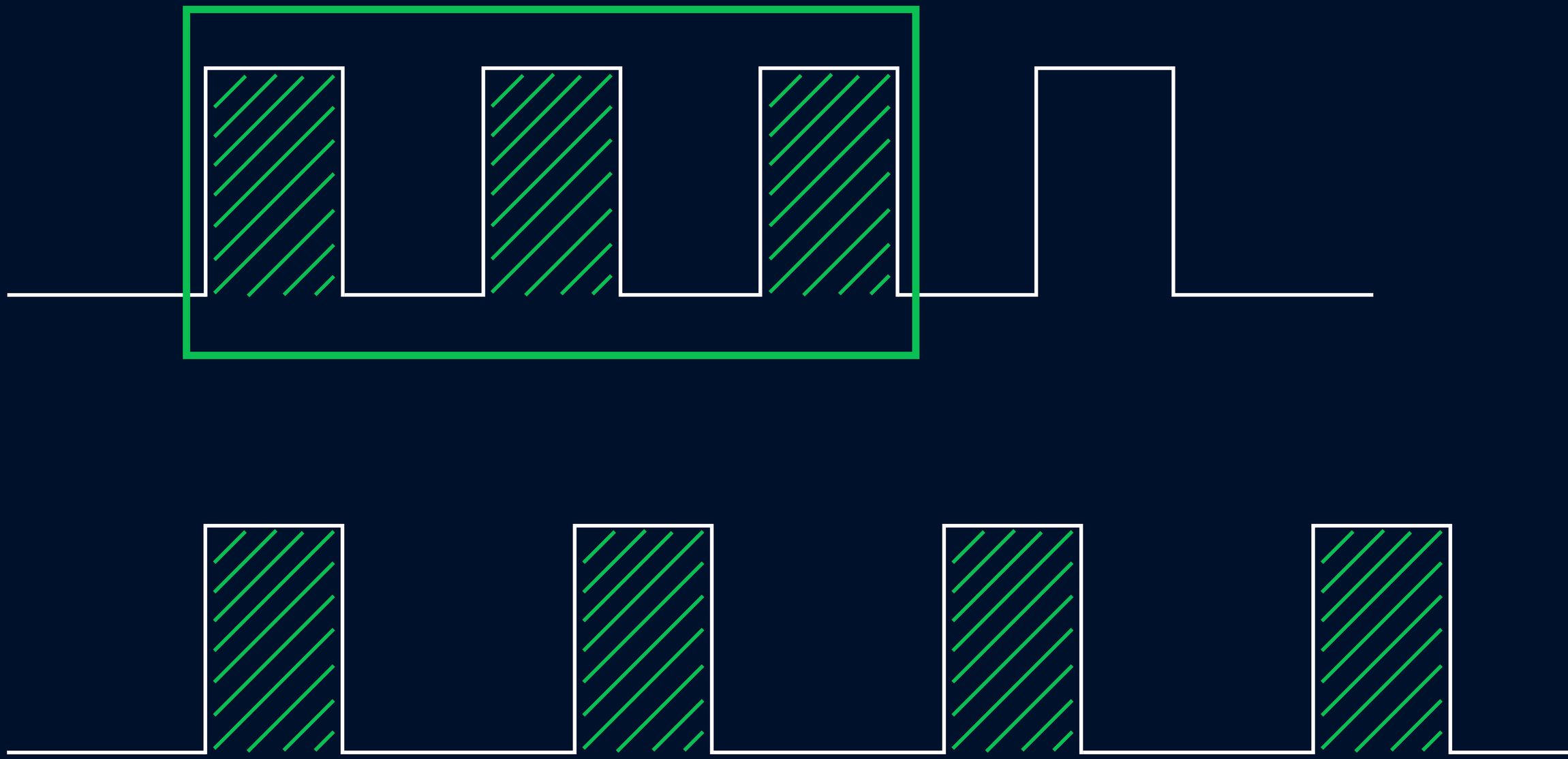

WAL

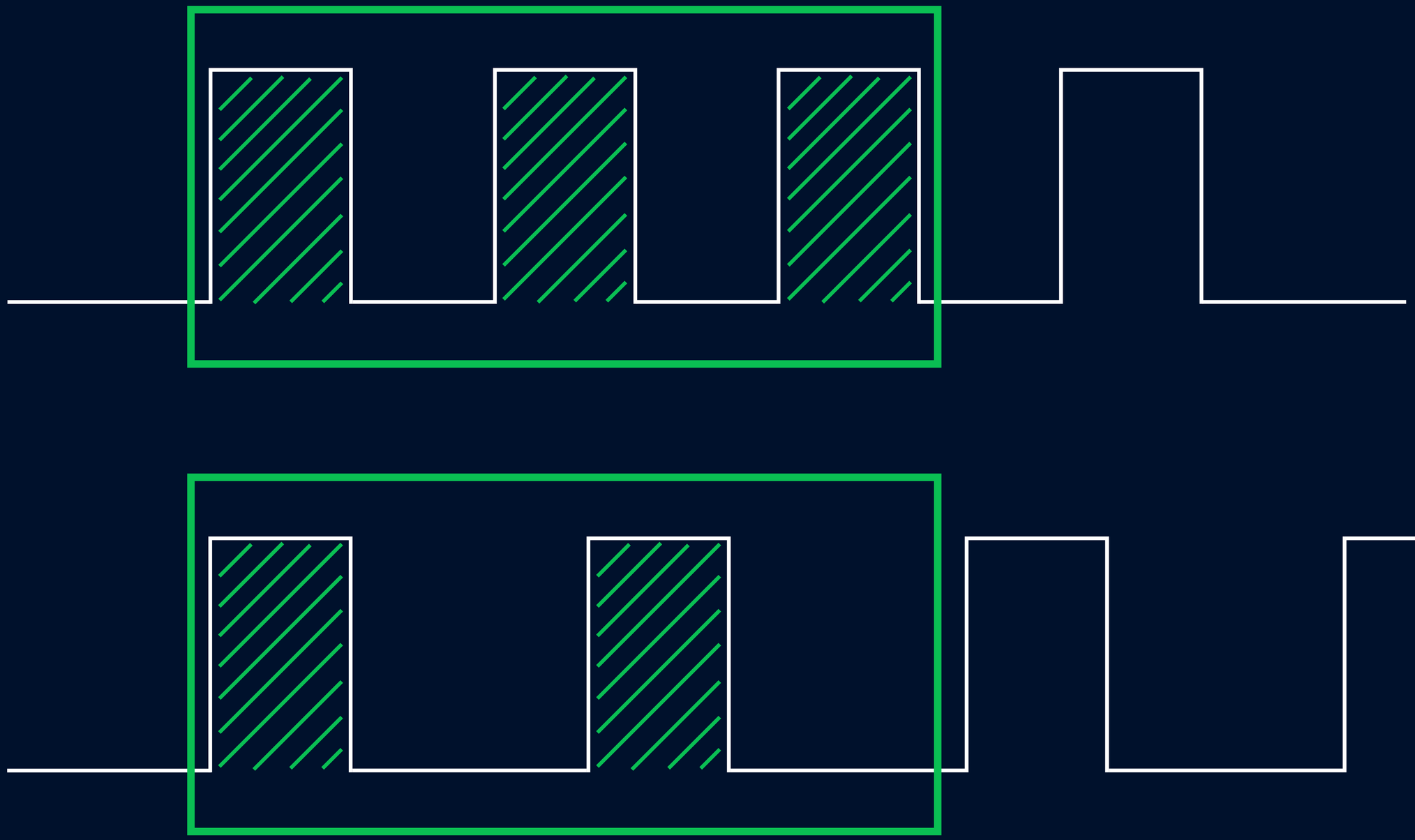










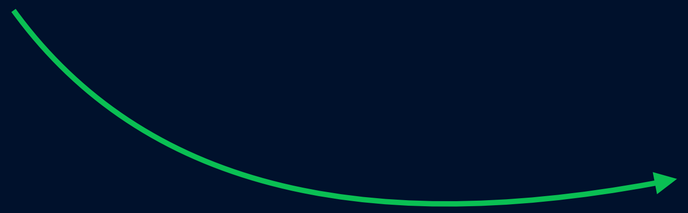


Manual Vacuum

Ruthless!

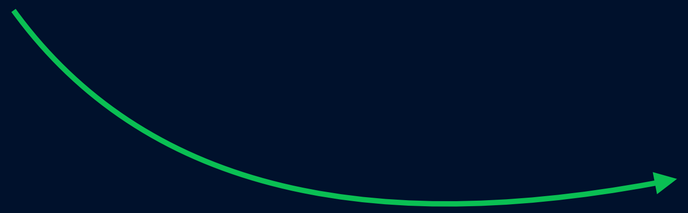
Manual Vacuum

Manual Vacuum



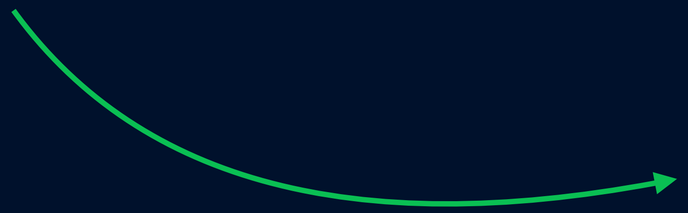
- Top bloated tables

Manual Vacuum



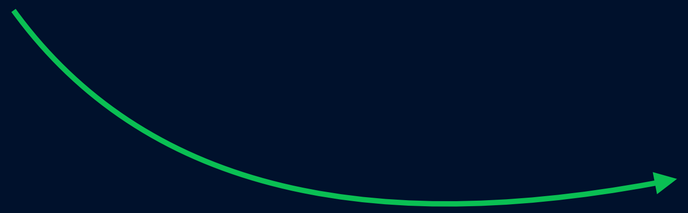
- Top bloated tables
- **Freeze tables**

Manual Vacuum



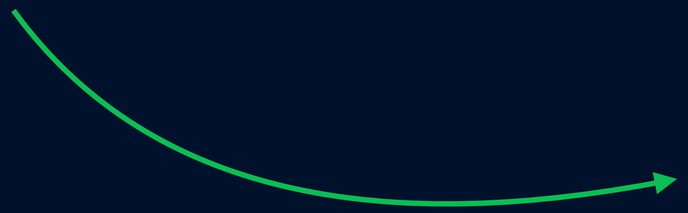
- Top bloated tables
- Freeze tables
- **Wraparound**

Manual Vacuum



- Top bloated tables
- Freeze tables
- Wraparound
- **Top N big tables**

Manual Vacuum

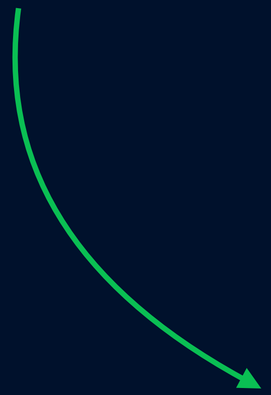


- Top bloated tables
- Freeze tables
- Wraparound
- Top N big tables
- **index_cleanup off**

“So, unless you are sure you know what you are doing (that is, you are in a wraparound-point emergency), please pretend this option [index_cleanup off] doesn’t exist. Really.”

Christophe Pettus

PostgreSQL v17



“**New memory management system for VACUUM…**

- **Allow vacuum to more efficiently remove and freeze tuples**

(Melanie Plageman, Heikki Linnakangas)

- **Allow vacuum to more efficiently store tuple references**

(Masahiko Sawada, John Naylor)”

Manual Vacuum

Manual Vacuum
Do not forget the Analyze!

To backup or not To backup


RUNNING PG_BASEBACKUP ON YOUR +100TB DATABASE
imgflip.com

Some “Math”:

Some “Math”:

Backup

=

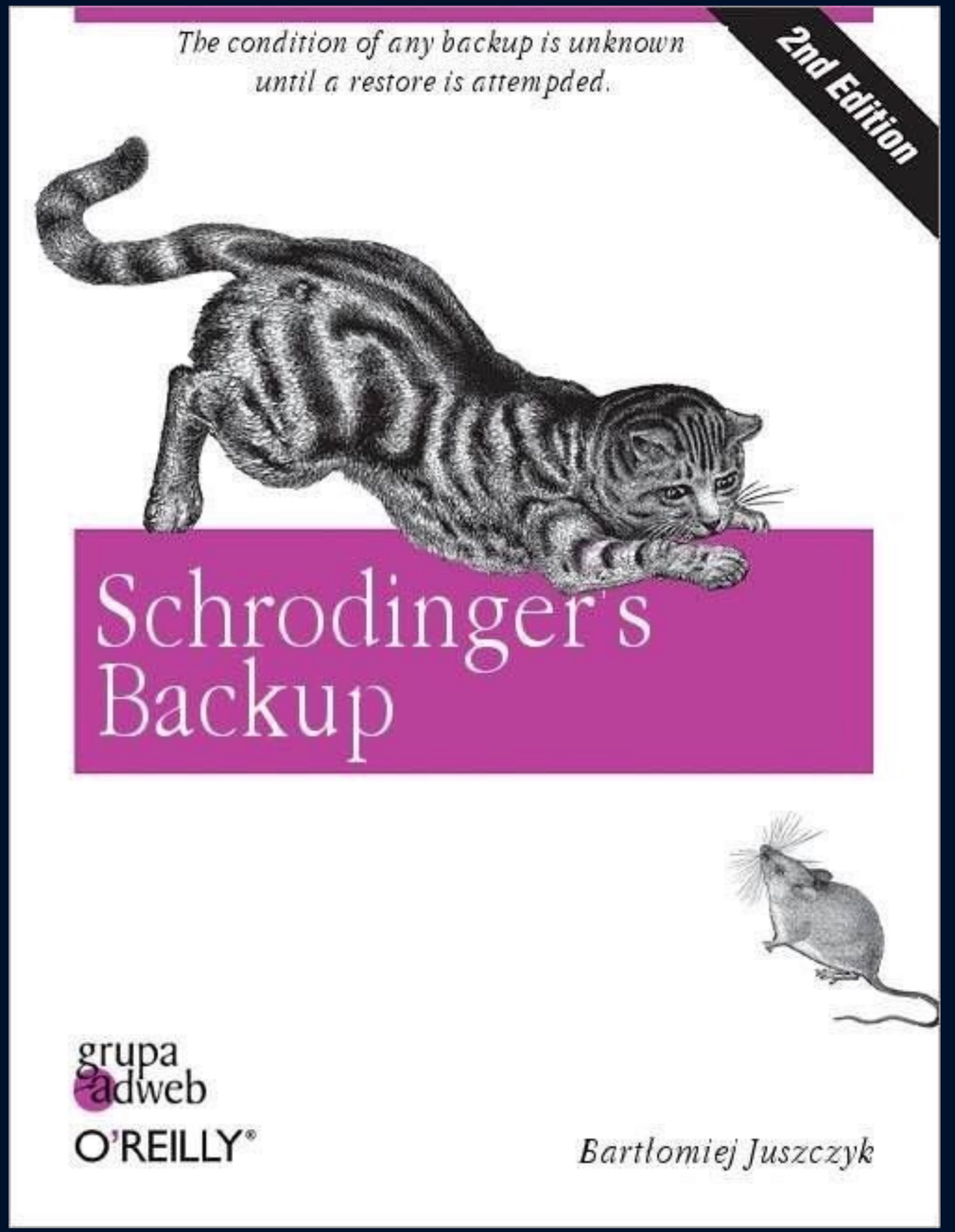
Backup + Restore

Some “Math”:

Backup

=

Backup + Restore



Source: https://orlybooks.com/

How we do it then?

Storage
Snapshots

+

WAL
Backups

But, and the restores?!

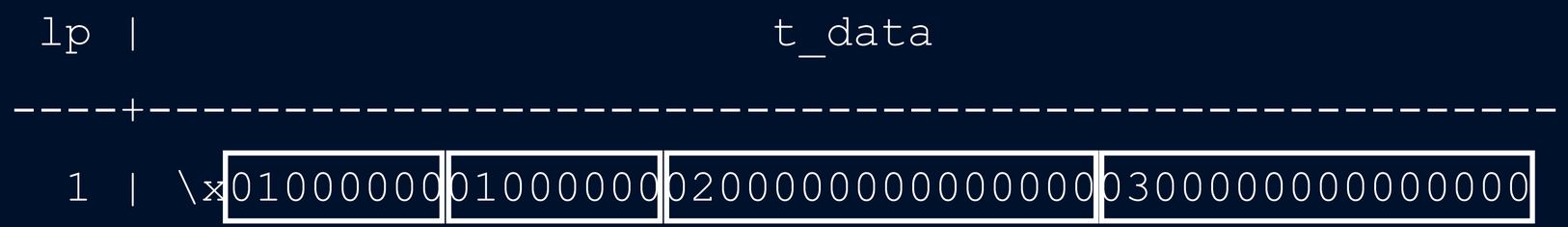
Data, Data everywhere

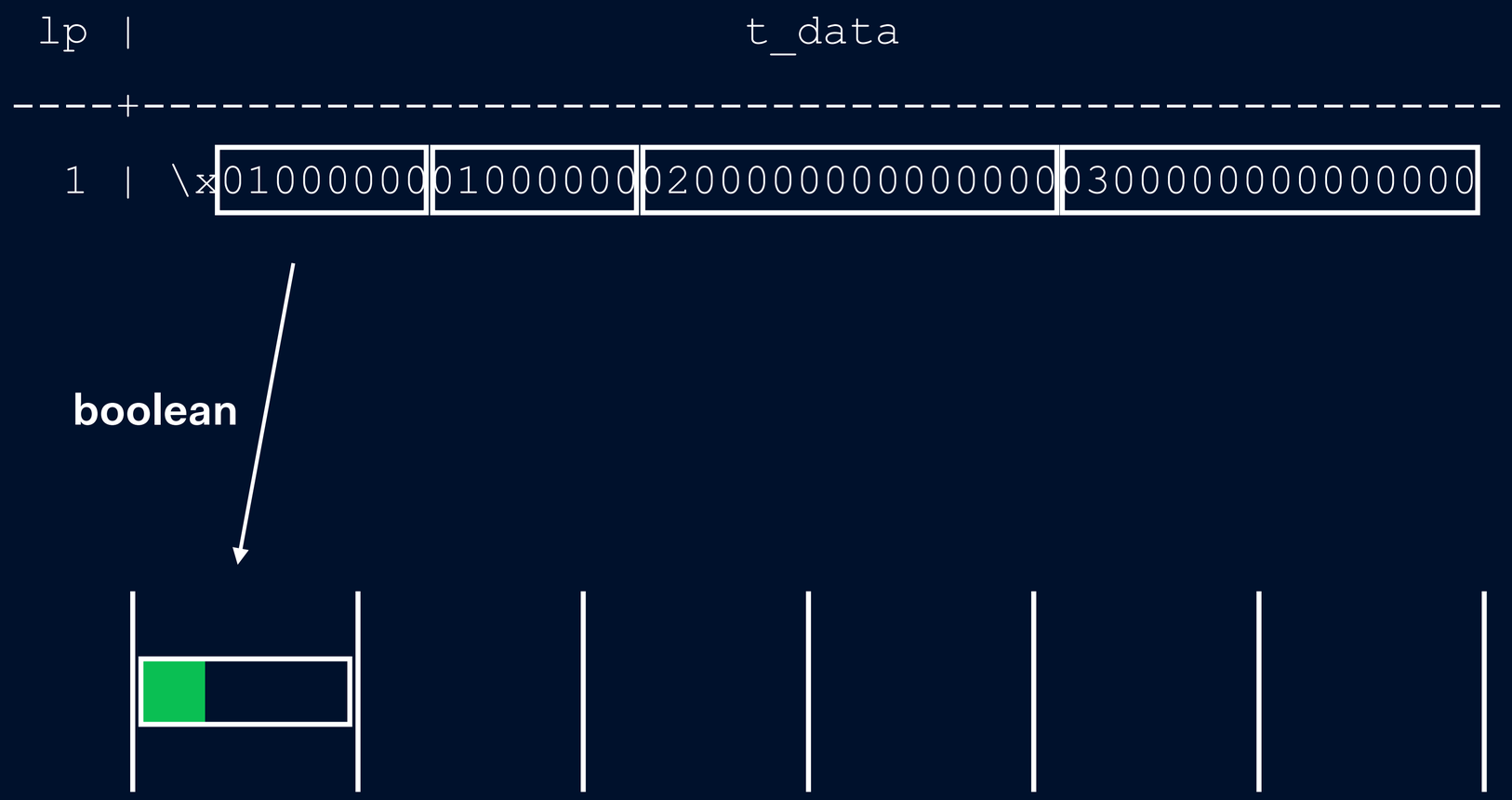

ALL OF THIS DATA...
WHY SHOULDN'T I KEEP IT?
imgflip.com

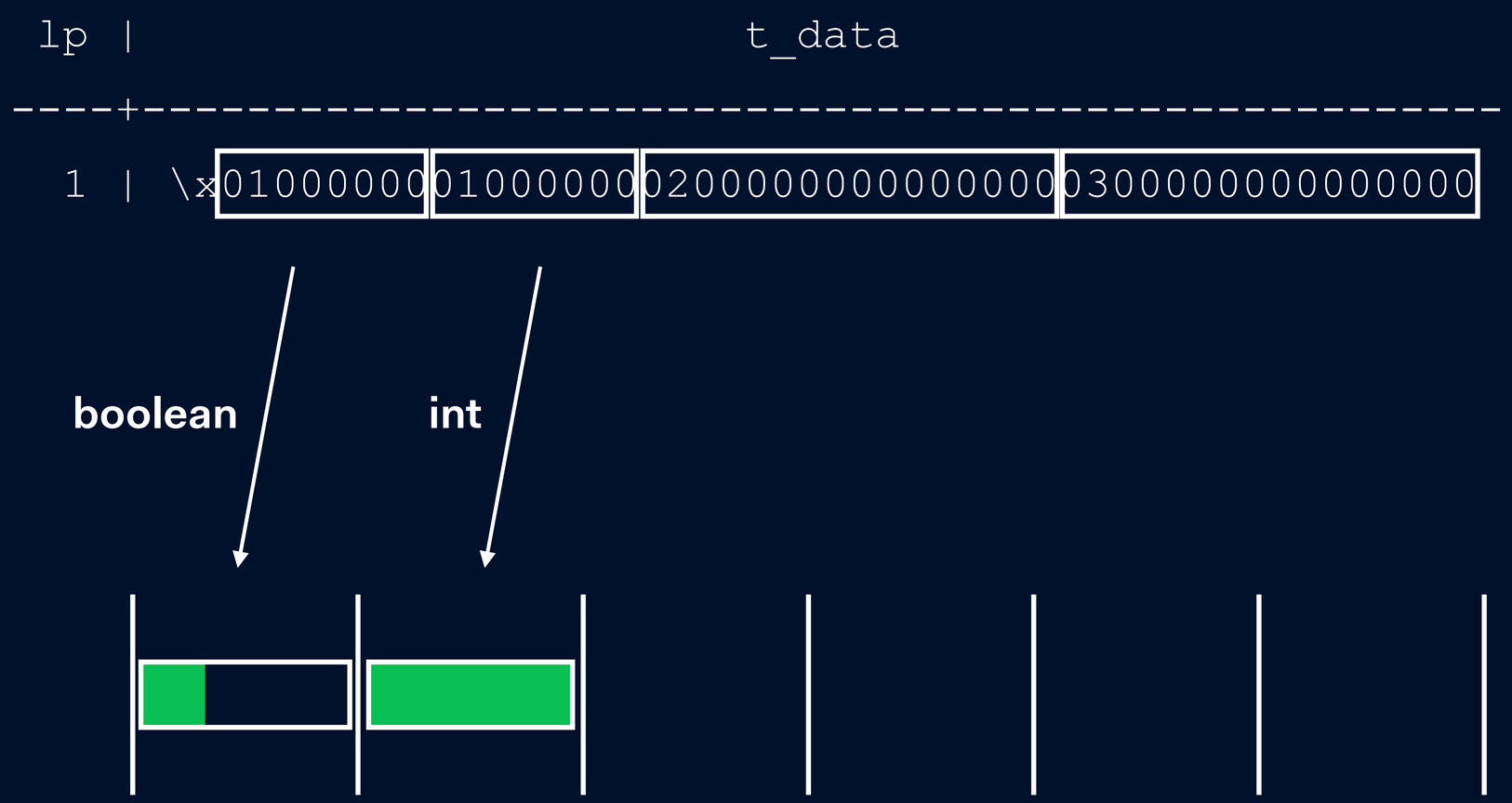
Get your columns in order

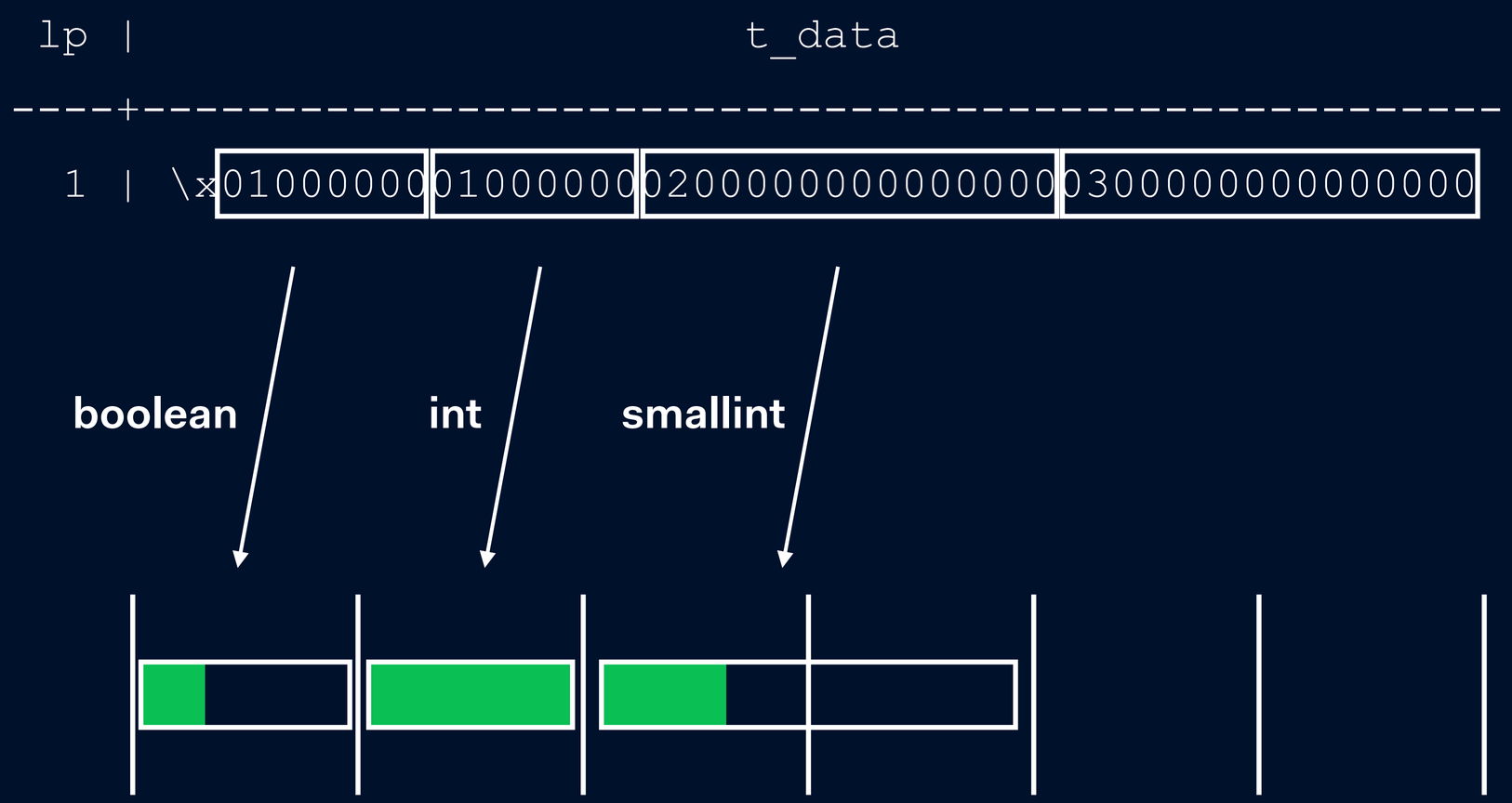
```
teresal=# \d test_order
             Table "public.test_order"
 Column |   Type   | Collation | Nullable | Default
--------+----------+-----------+----------+---------
 c1     | boolean  |           |          |
 c2     | integer  |           |          |
 c3     | smallint |           |          |
 c4     | bigint   |           |          |
```

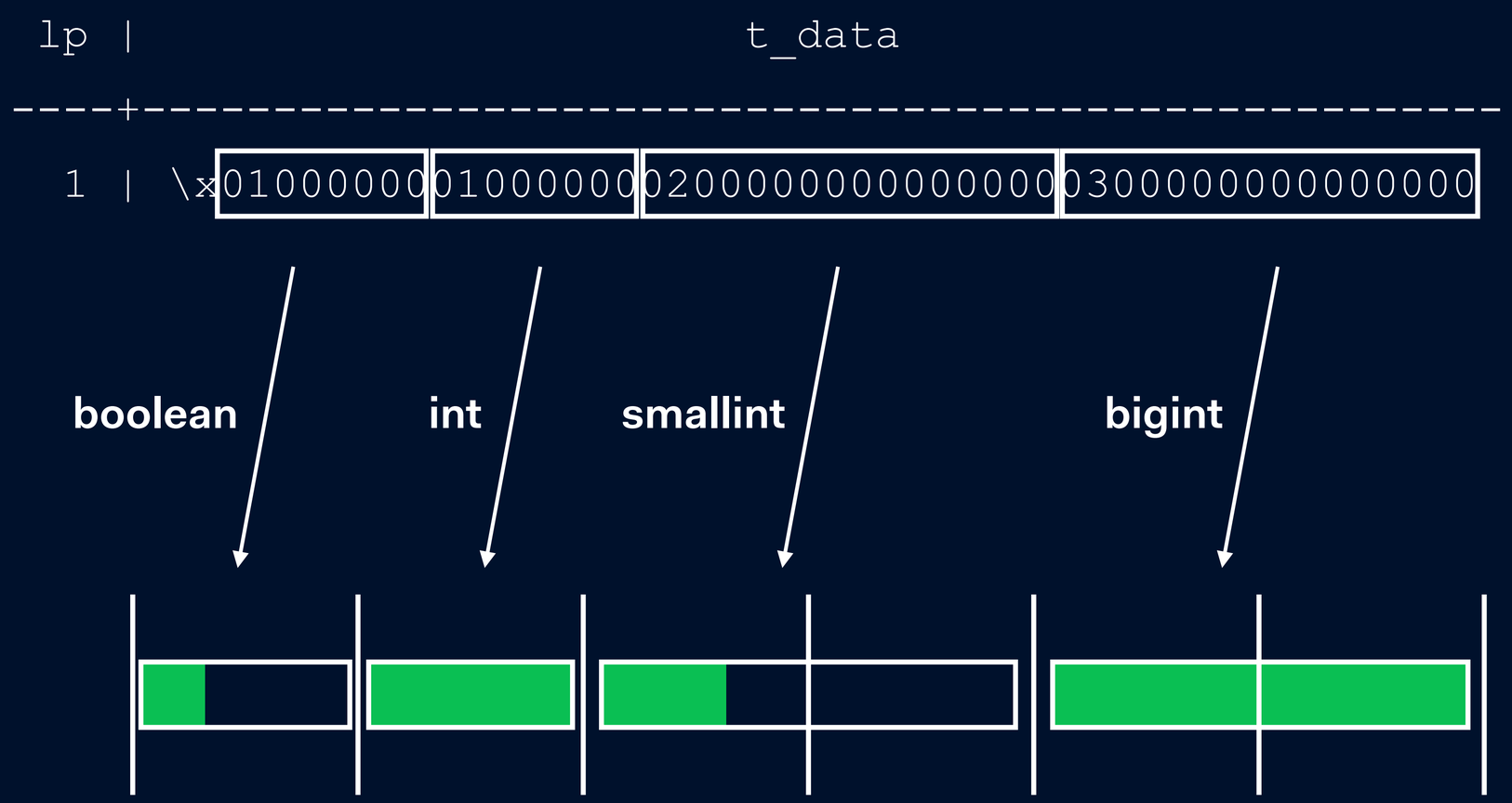
```
teresal=# \d test_order
             Table "public.test_order"
 Column |   Type   | Collation | Nullable | Default
--------+----------+-----------+----------+---------
 c1     | boolean  |           |          |
 c2     | integer  |           |          |
 c3     | smallint |           |          |
 c4     | bigint   |           |          |

teresal=# select * from test_order;
 c1 | c2 | c3 | c4
----+----+----+----
 t  |  1 |  2 |  3
```

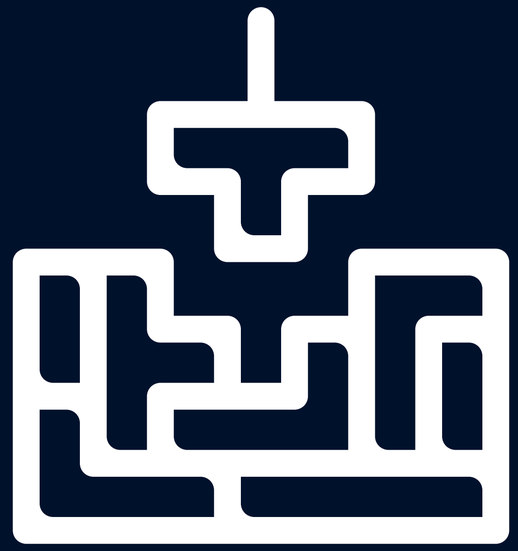

lp |
t_data
1 | \x
01000000
01000000
02000000000000000
03000000000000000


lp | t_data
1 | \x01000000010000000200000000000000300000000000000
boolean

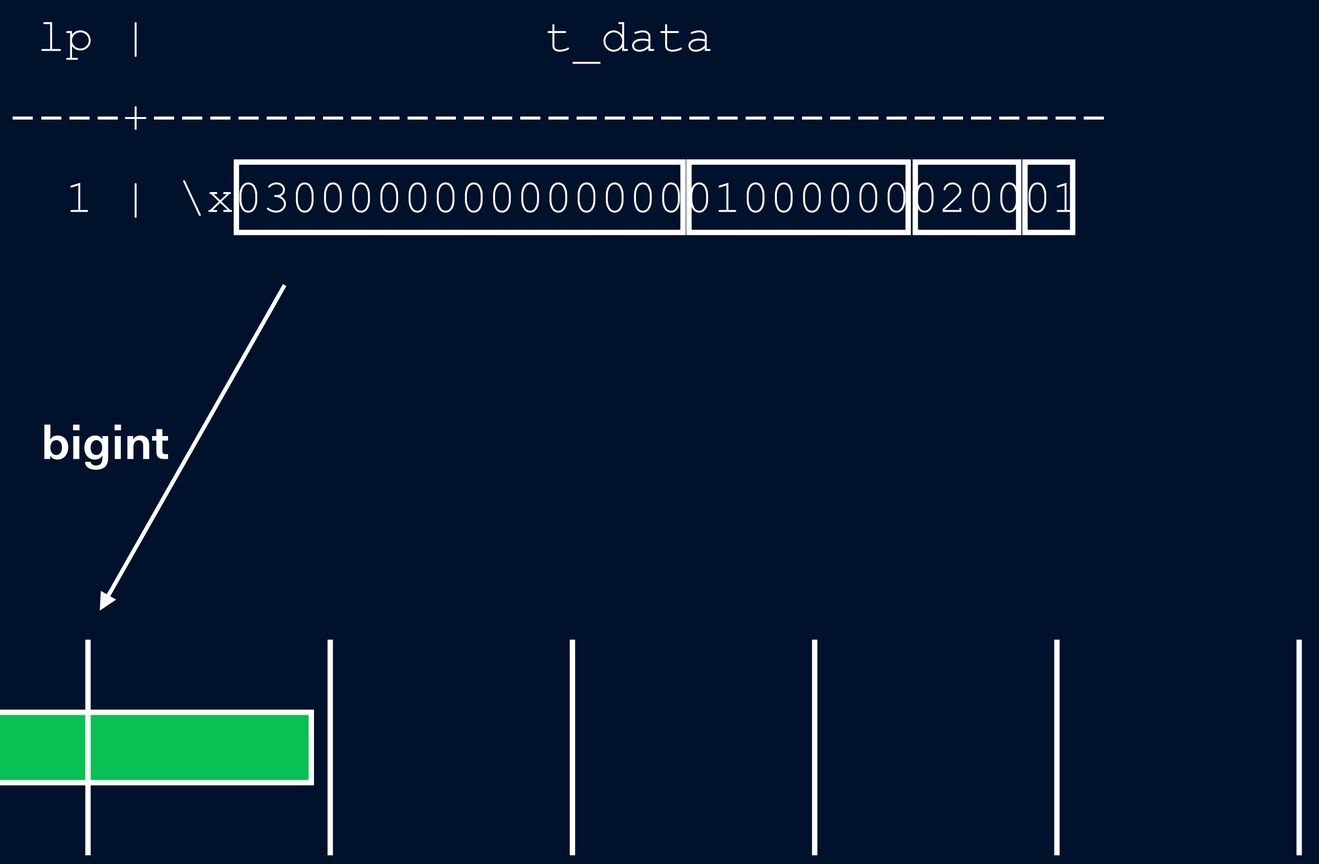

lp |
t_data
1 | \x
01000000
01000000
02000000000000000
03000000000000000
boolean
int

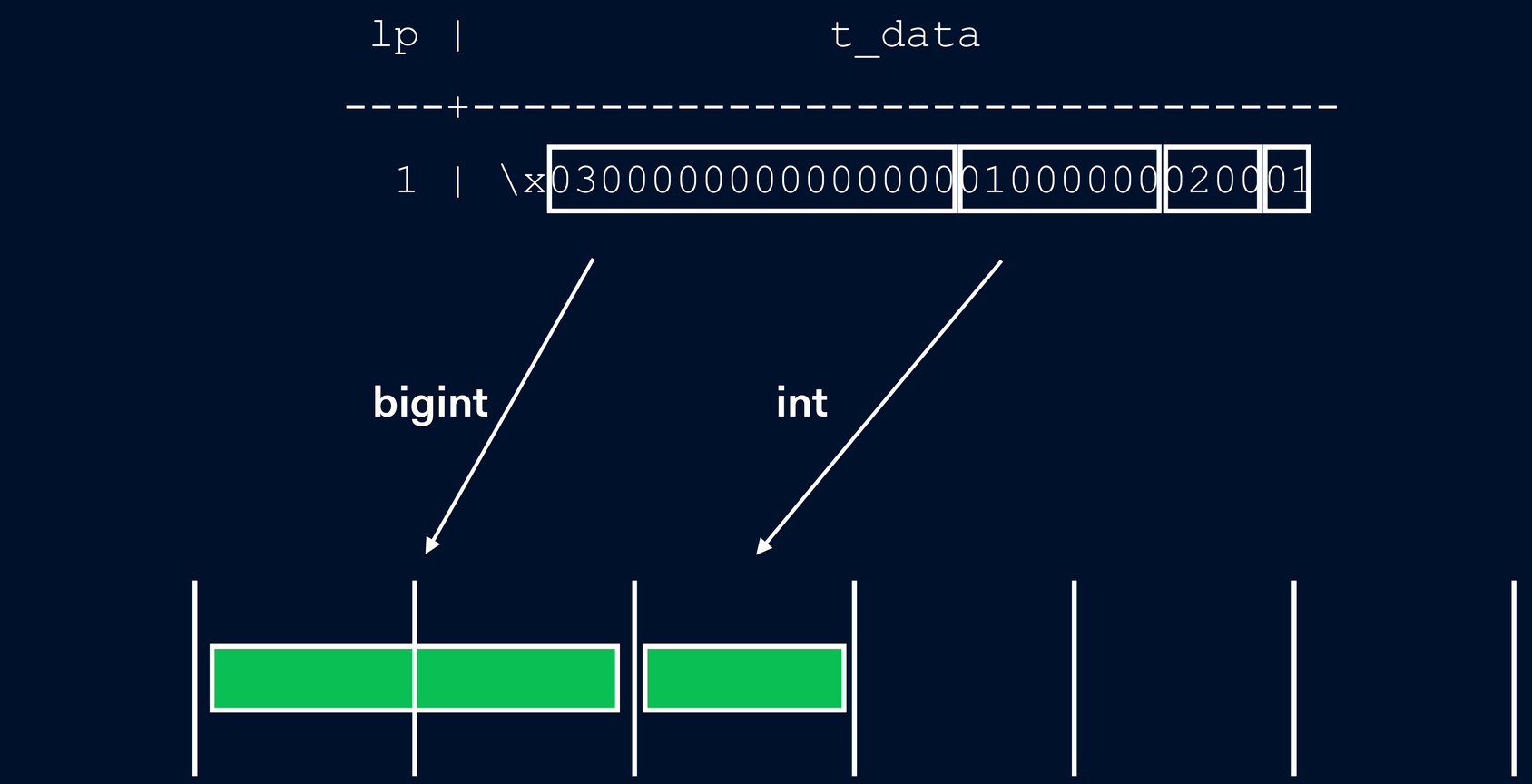

lp |
t_data
1 | \x
01000000
01000000
02000000000000000
03000000000000000
boolean
int
smallint

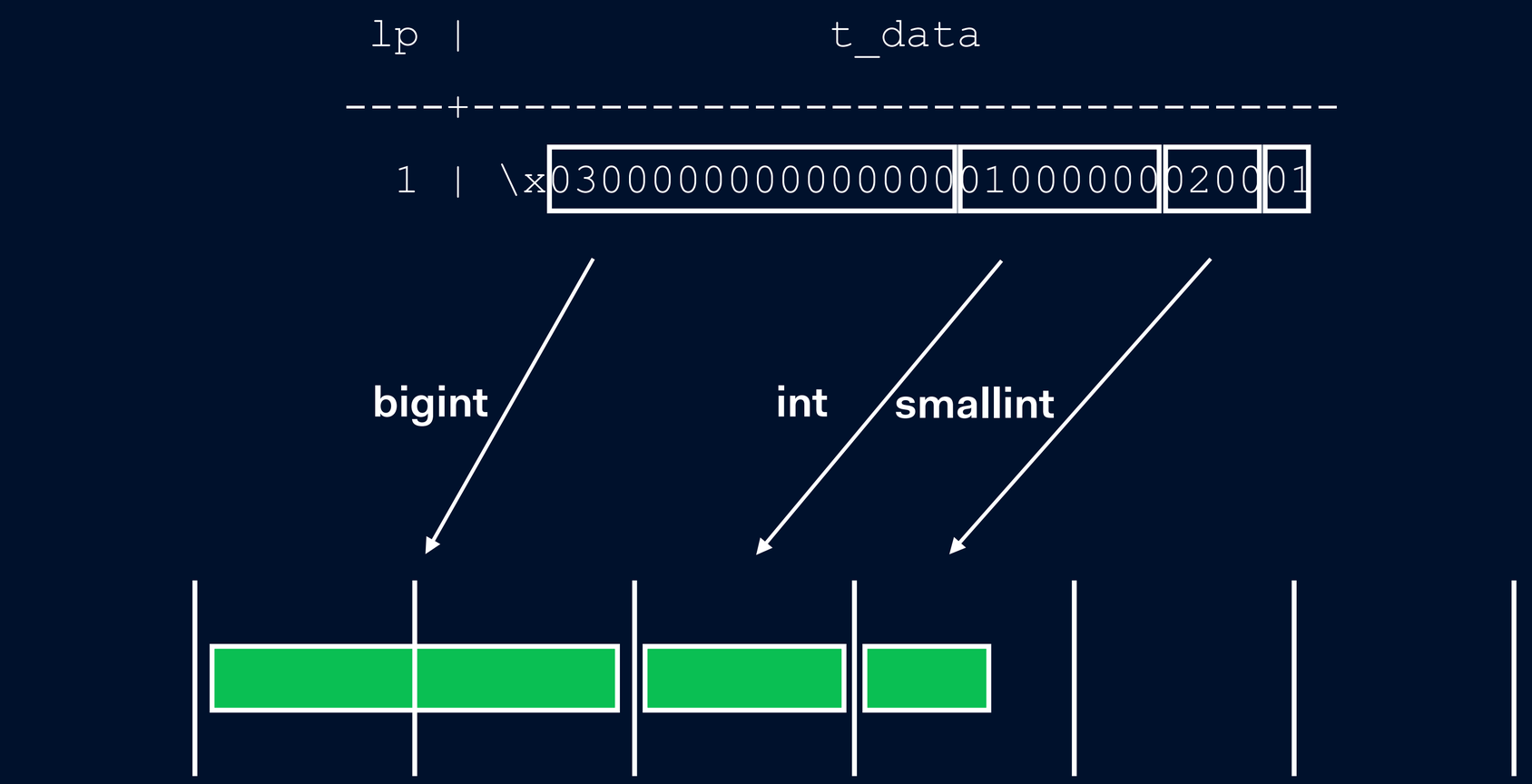

lp |
t_data
1 | \x
01000000
01000000
02000000000000000
03000000000000000
boolean
int
smallint
bigint

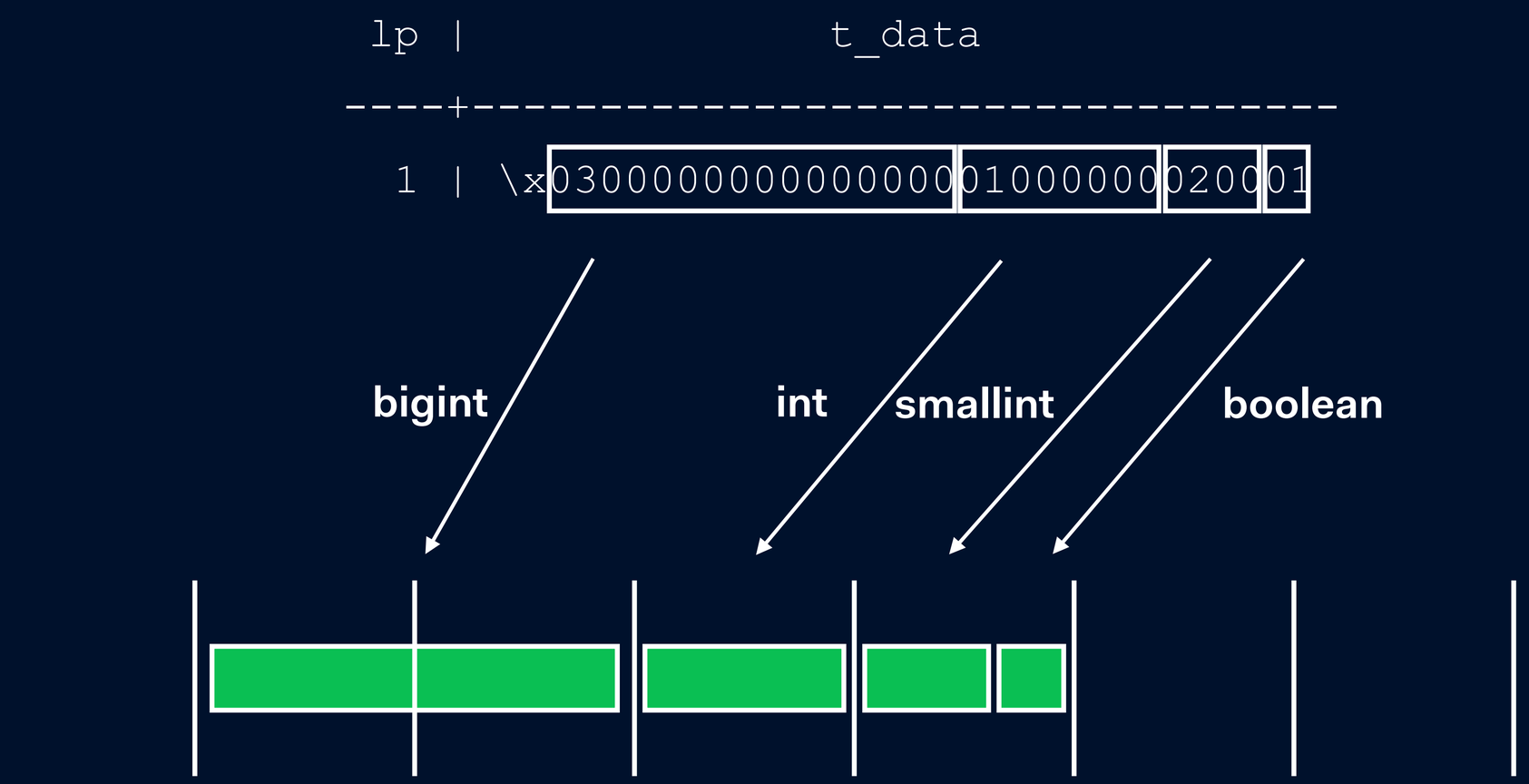


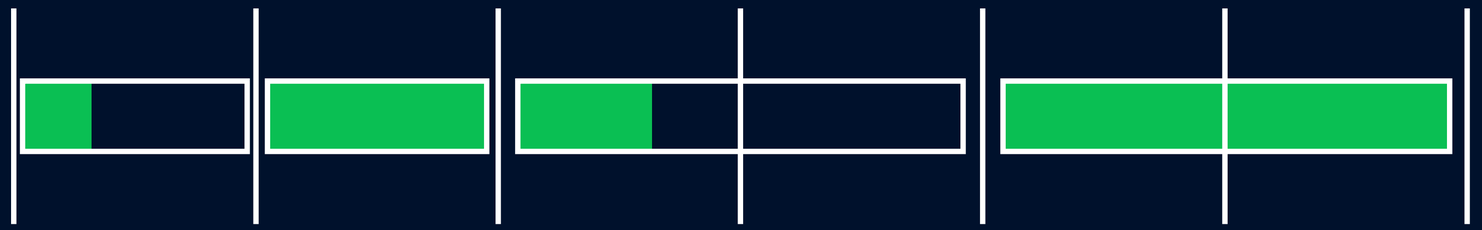

lp | t_data
1 | \x03000000000000000100000002000 1

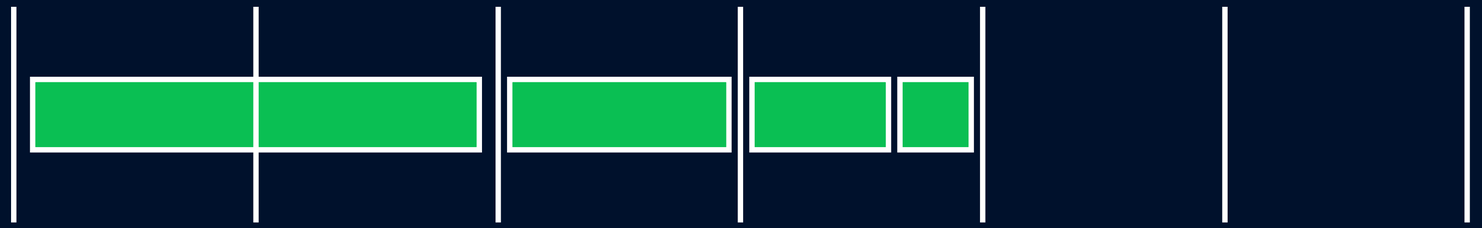

lp |
t_data
1 | \x03000000000000000100000002000 1
bigint


lp | t_data
1 | \x03000000000000000100000002000 1
bigint
int


lp |
t_data
1 | \x03000000000000000100000002 0001
bigint
int
smallint


lp |
t_data
1 | \x0300000000000000010000000200 01
bigint
int
smallint
boolean





More “Math”:

10% of +100TB

=

+10TB

More “Math”:

10% of +100TB

=

+10TB

More “Math”:

10% of +100TB

=

+10TB

DECEMBER 6, 2024 • 41 MINUTES

Column Tetris

Nikolay and Michael discuss "Column Tetris" — what it is, why it matters, how to order columns for new tables, and how to re-organise existing ones. Here are some link...

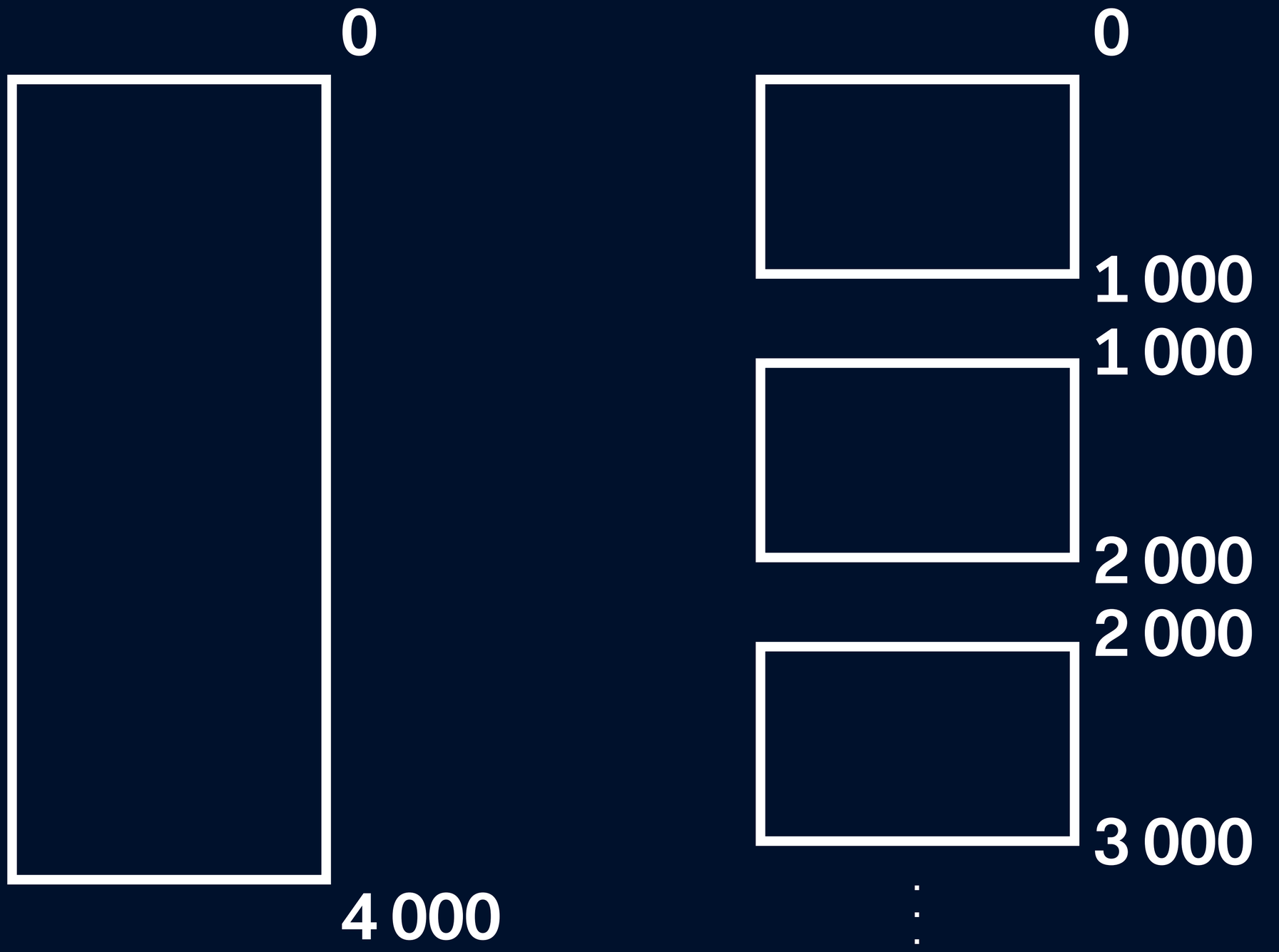
FOSDEM 2024

Reducing Costs and Improving Performance With Data Modeling in Postgres	Charly Batista

Tame the chaos,

Partition!


0
4 000


0
4 000
0
1 000
1 000
2 000
2 000
3 000

1 table ✓

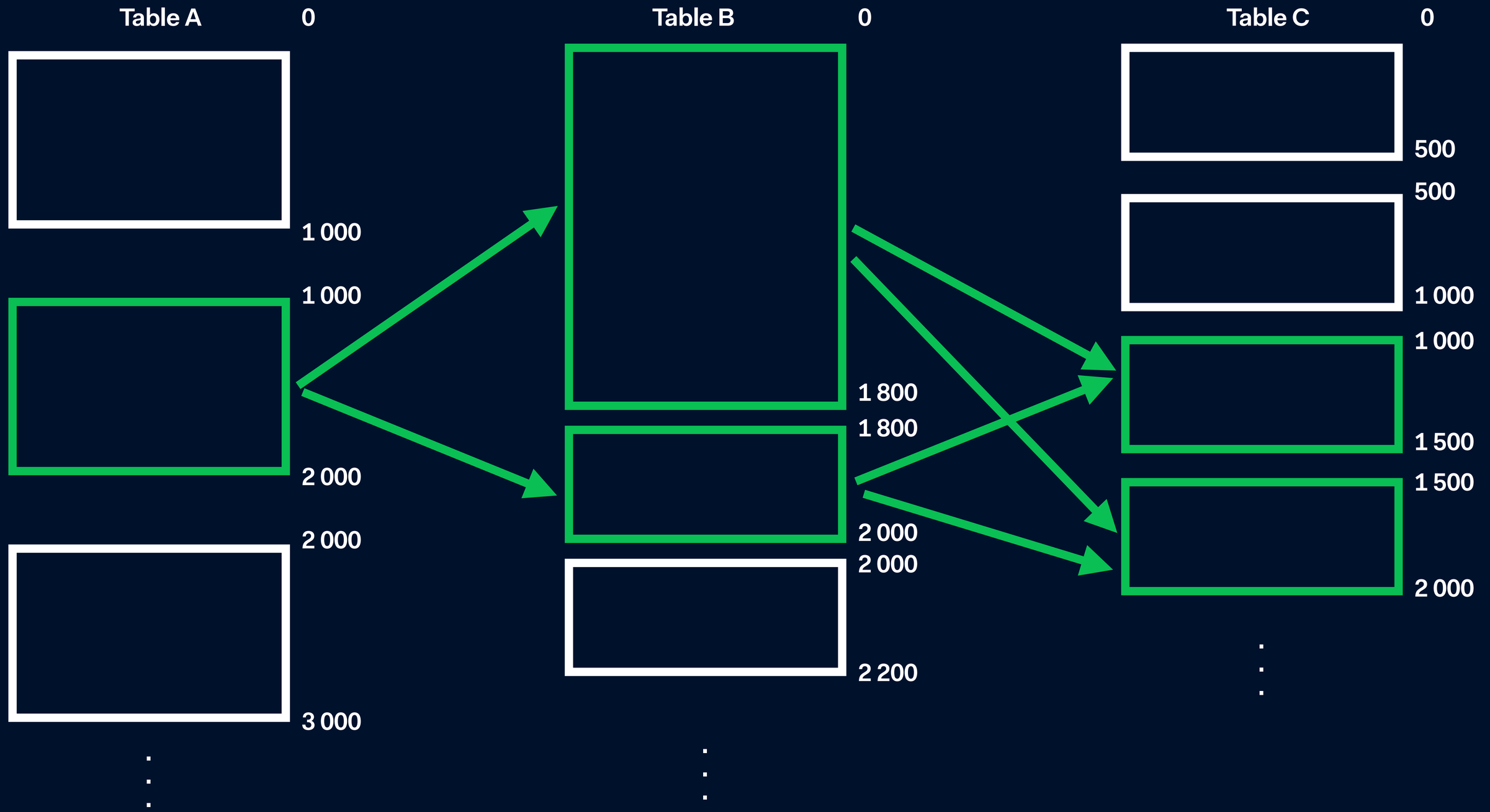
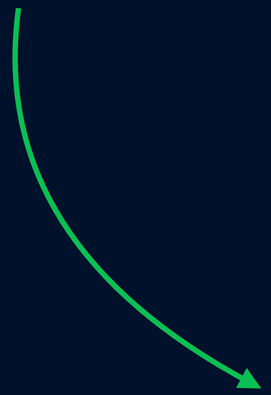


Table A
0
1 000
1 000
2 000
2 000
3 000
Table B
0
1 800
1 800
2 000
2 000
2 200
Table C
0
500
500
1 000
1 000
1 500
1 500
2 000

fast-path locks

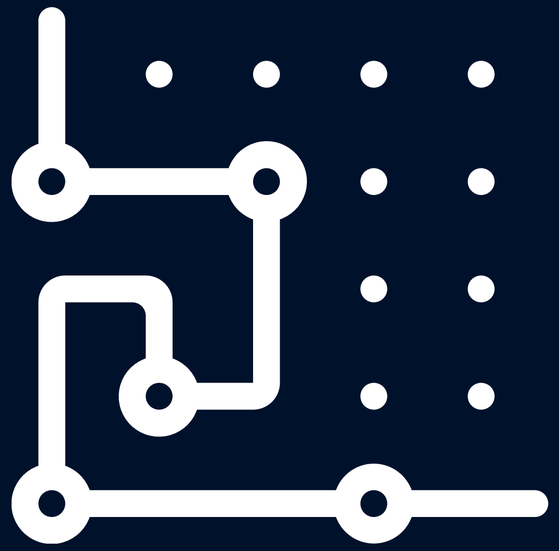
LWLock:lock_manager

PostgreSQL v18



Improve the locking performance of queries that access many relations

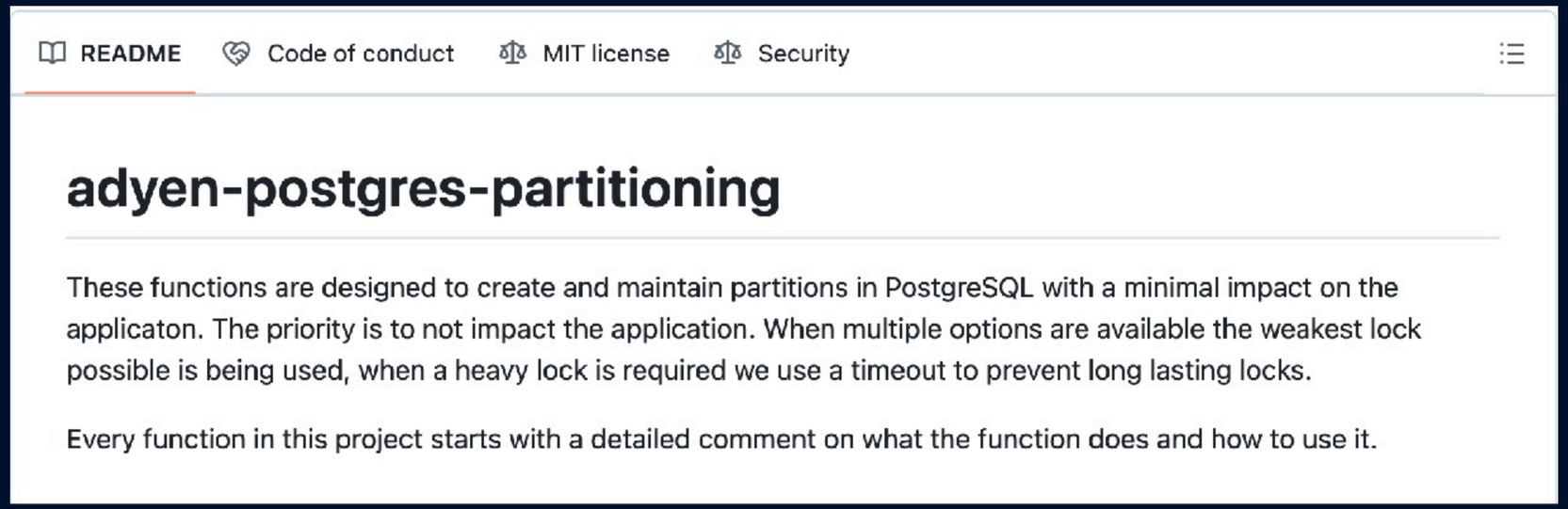
(Tomas Vondra)



Planning Time: 12.389 ms

Execution Time: 0.164 ms

Partitioning at Adyen



Partitioning at Adyen

Partitioning at Adyen


Tech stories

A Deep Dive into Table Partitioning part 1 🐰: Introduction to Table Partitioning

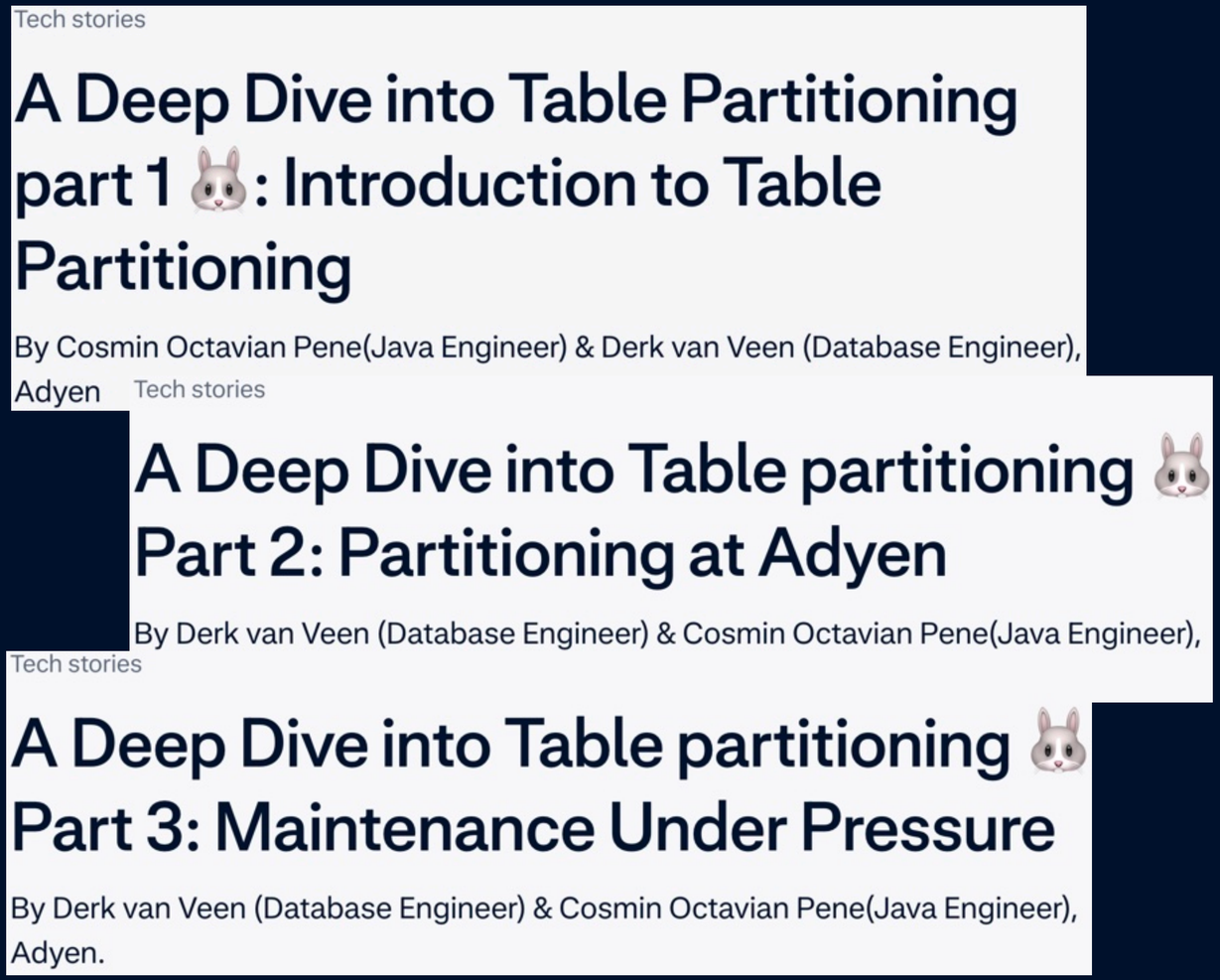
By Cosmin Octavian Pene(Java Engineer) & Derk van Veen (Database Engineer), Adyen


Tech stories

A Deep Dive into Table partitioning 🐰 Part 2: Partitioning at Adyen

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer), Adyen

Partitioning at Adyen



Tech stories

A Deep Dive into Table Partitioning part 1 🐰: Introduction to Table Partitioning

By Cosmin Octavian Pene(Java Engineer) & Derk van Veen (Database Engineer), Adyen

Tech stories

A Deep Dive into Table partitioning 🐰 Part 2: Partitioning at Adyen

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer),

Tech stories

A Deep Dive into Table partitioning 🐰 Part 3: Maintenance Under Pressure

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer), Adyen.

Partitioning at Adyen

Tech stories

A Deep Dive into Table Partitioning part 1 🐇: Introduction to Table Partitioning

By Cosmin Octavian Pene(Java Engineer) & Derk van Veen (Database Engineer), Adyen

Tech stories

A Deep Dive into Table partitioning 🐇 Part 2: Partitioning at Adyen

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer),

Tech stories

A Deep Dive into Table partitioning 🐇 Part 3: Maintenance Under Pressure

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer), Adyen.

Tech stories

A Deep Dive into Table partitioning 🐇 Part 4: How the default partition saved the day

By Derk van Veen, Database Engineer

Partitioning at Adyen

Tech stories

A Deep Dive into Table Partitioning part 1 🐰: Introduction to Table Partitioning

By Cosmin Octavian Pene(Java Engineer) & Derk van Veen (Database Engineer), Adyen

Tech stories

A Deep Dive into Table partitioning 🐰 Part 2: Partitioning at Adyen

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer),

Tech stories

A Deep Dive into Table partitioning 🐰 Part 3: Maintenance Under Pressure

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer), Adyen.

Tech stories

A Deep Dive into Table partitioning 🐰 Part 4: How the default partition saved the day

By Derk

Tech stories

Efficient Data Cleanup in Partitioned PostgreSQL Tables using Common Table Expressions

By Dwarka Rao, Database Engineer & Aakash Agarwal, Java Software Engineer.

Partitioning at Adyen

Tech stories

A Deep Dive into Table Partitioning part 1 🐇: Introduction to Table Partitioning

By Cosmin Octavian Pene(Java Engineer) & Derk van Veen (Database Engineer), Adyen

Tech stories

A Deep Dive into Table partitioning 🐇 Part 2: Partitioning at Adyen

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer),

Tech stories

A Deep Dive into Table partitioning 🐇 Part 3: Maintenance Under Pressure

By Derk van Veen (Database Engineer) & Cosmin Octavian Pene(Java Engineer), Adyen.

Tech stories

A Deep Dive into Table partitioning 🐇 Part 4: How the default partition saved the day

By Derk

Tech stories

Efficient Data Cleanup in Partitioned PostgreSQL Tables using Common Table Expressions

By Dwarka Rao, Database Engineer & Aakash Agarwal, Java Software Engineer.

Efficiently RePartitioning Large Tables in PostgreSQL

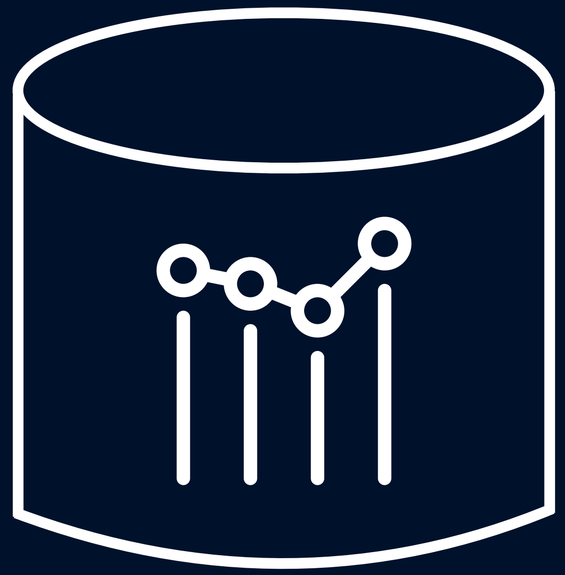
By Cagri Biroglu, Database Engineer

Data Temperature

Query Performance

Prepare your Statements

Prepare your Statements (Carefully)



Plans

Generic

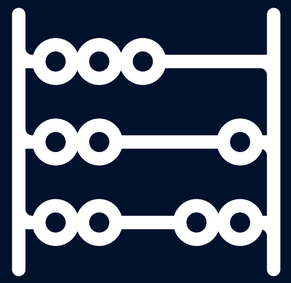
```
plan_cache_mode=force_generic_plan
```

Generic

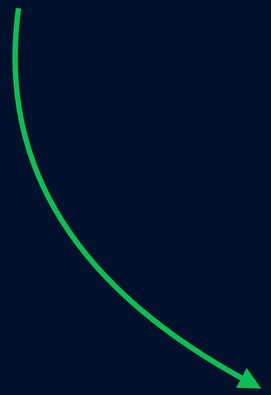
`plan_cache_mode=force_generic_plan`

Custom

`plan_cache_mode=force_custom_plan`



PostgreSQL v16



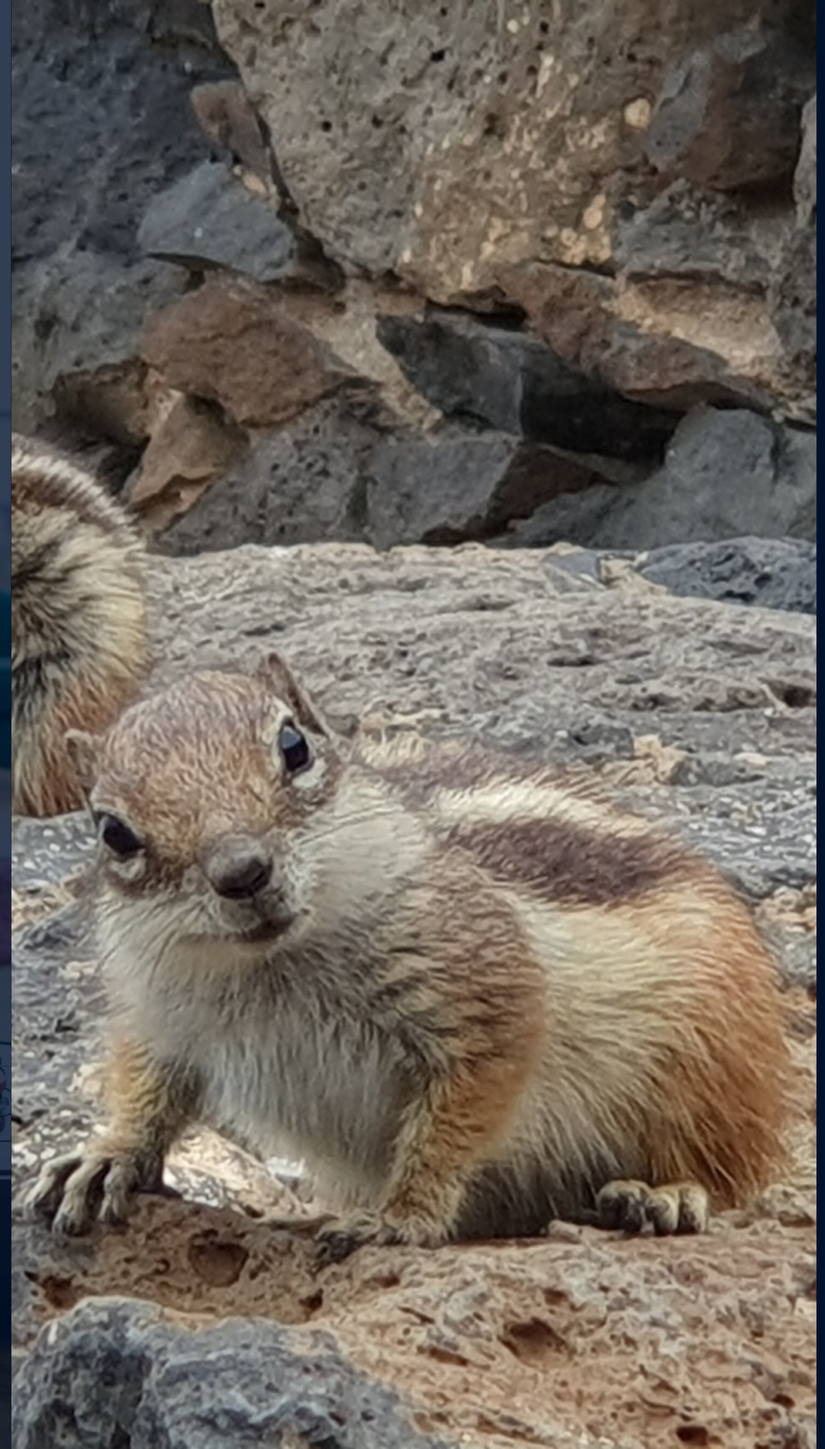
Add EXPLAIN option GENERIC_PLAN to display the generic plan for a parameterized query

(Laurenz Albe)

Wrap Up


OUT
OF
ORDER




OUT
OF
ORDER





OUT
OF
ORDER

What works at 100GB will fail catastrophically at +100TB.

Operational hazards of managing PostgreSQL DBs over 100TB

adyen

engineered
for ambition