What Is Dead May Roll Back: Undo Logging in OrioleDB

Artur Zakirov @ Supabase

June 2025, Berlin

supabase

Oriole

About me

- PostgreSQL Core Developer at Supabase since September 2024
- PostgreSQL Developer since 2015
- Big fan of cycling and bouldering

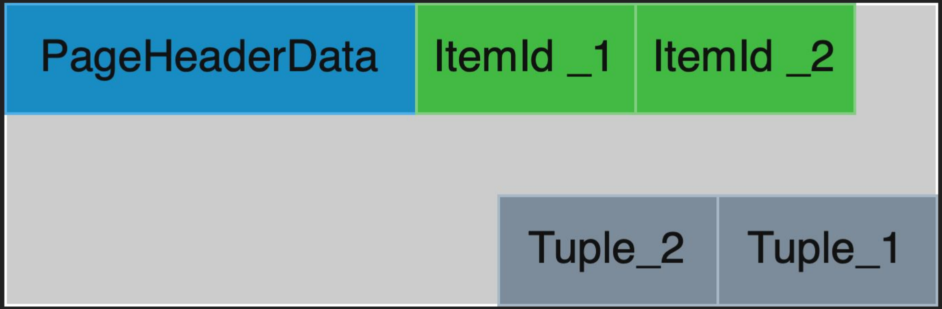
About Supabase

- Backend-as-a-Service (BaaS)
- PostgreSQL database, Authentication, REST and GraphQL APIs, Edge Functions, Realtime subscriptions, Storage, and Vector embeddings
- github.com/supabase/supabase is in Top 100 Stars projects:
  - github.com/EvanLi/Github-Ranking/blob/master/Top100/Top-100-stars.md
- github.com/orioledb/orioledb - OrioleDB repo

MVCC in PostgreSQL

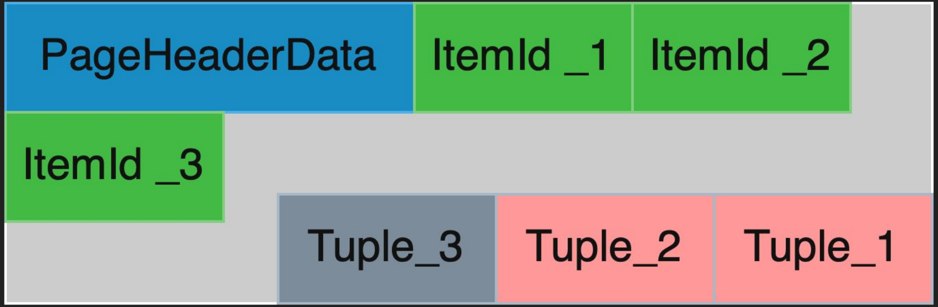
- Multi-version Concurrency Control (MVCC)
  - New version of data on writes
  - Consistent snapshot to read and write data
- “Readers don’t block writers, and writers don’t block readers”

MVCC in PostgreSQL: Heap



	t_xmin	t_xmax	t_ctid	Data
Tuple_1	1	0	(0,1)	...
Tuple_2	2	0	(0,2)	...

MVCC in PostgreSQL: Heap



1. DELETE Tuple_1
2. UPDATE Tuple_2

	t_xmin	t_xmax	t_ctid	Data
Tuple_1	1	3	(0,1)	...
Tuple_2	2	4	(0,3)	...
Tuple_3	4	0	(0,3)	...

Oldest-to-newest version chain

MVCC in PostgreSQL: Pitfalls

- Table bloat
  - Can be dealt by VACUUM FULL (requires ACCESS EXCLUSIVE lock), pg_repack
- Write amplification
  - Heap-only tuples (HOT) updates can reduce write amplification
- Autovacuum process
  - Tuning is tricky
  - Additional IO
  - Struggle with big tables

Table access method

```
=# SELECT amname FROM pg_am WHERE amtype = 't';
 heap
 orioledb
=# CREATE TABLE tab (id int) USING orioledb;
```

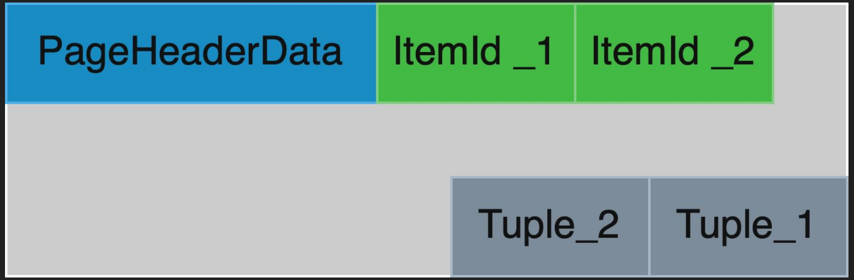
Notable table access methods:

- zheap (discontinued): github.com/EnterpriseDB/zheap
- columnar: github.com/citusdata/citus/tree/main/src/backend/columnar

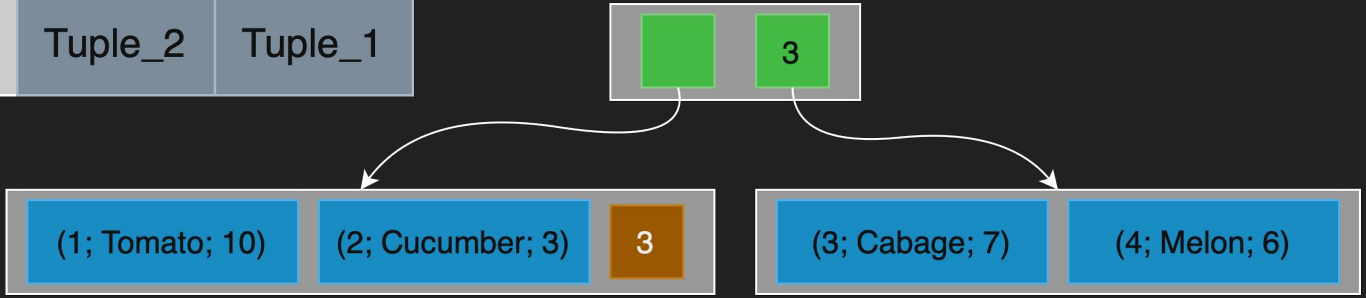
www.postgresql.org/docs/current/tableam.html

Table access method

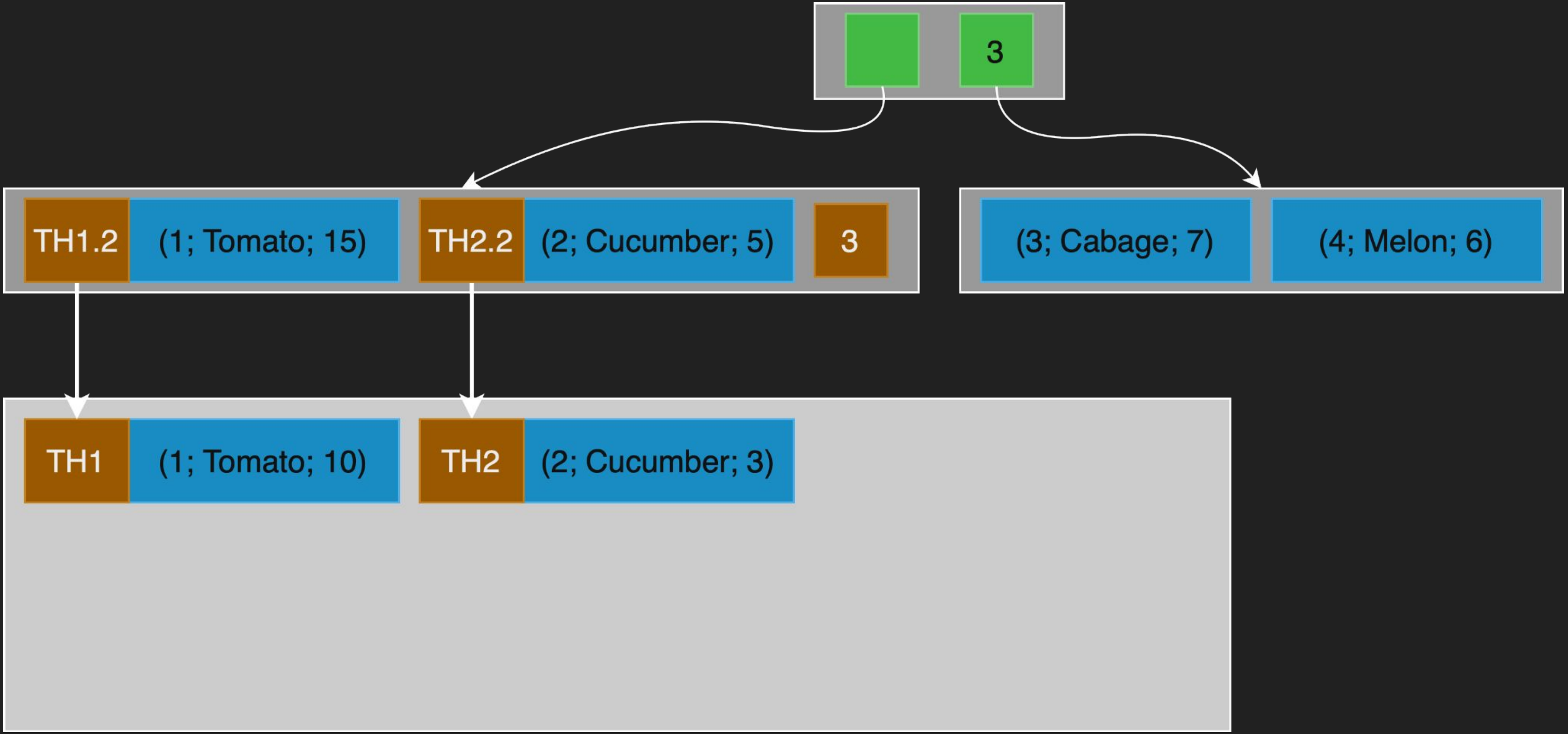
heap



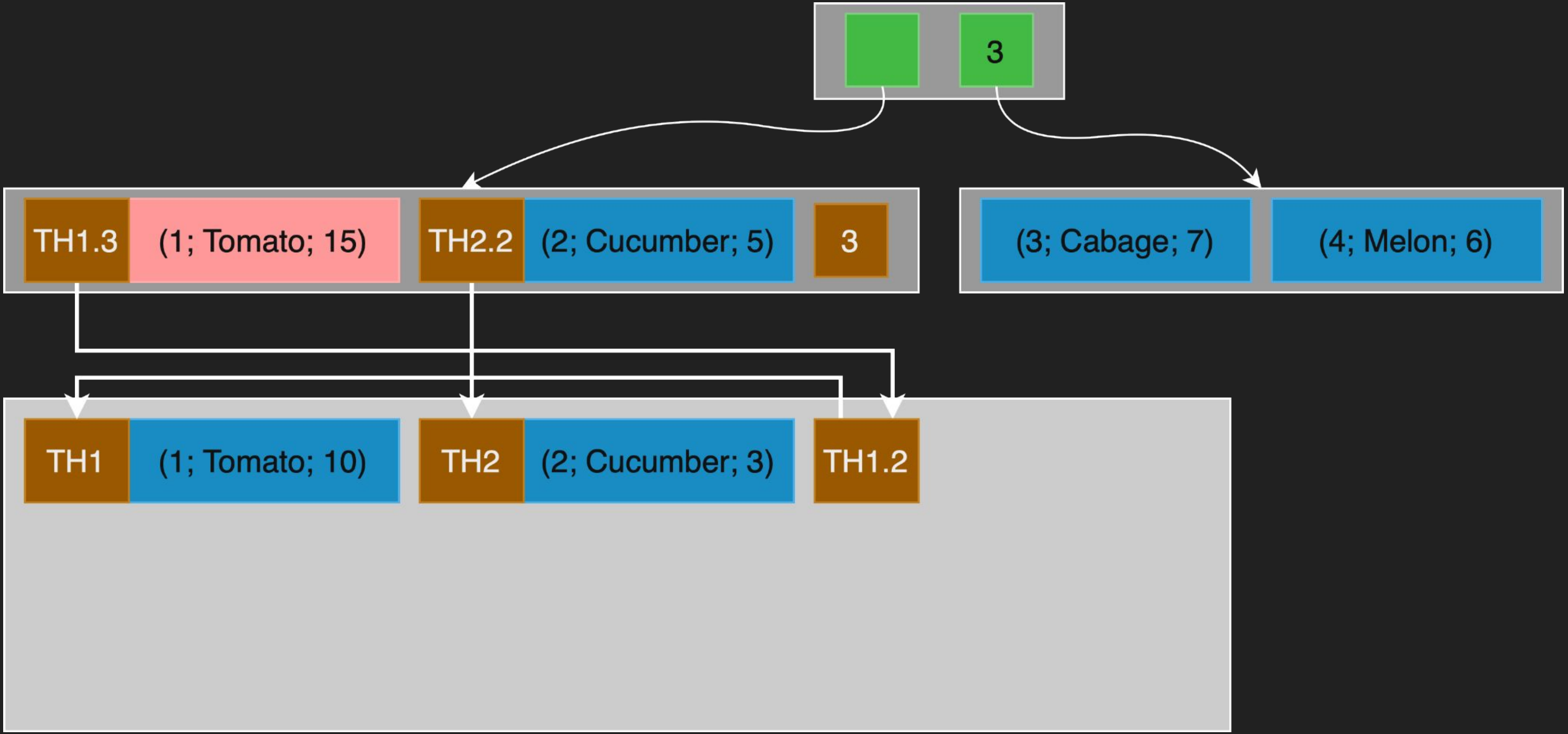
orioledb



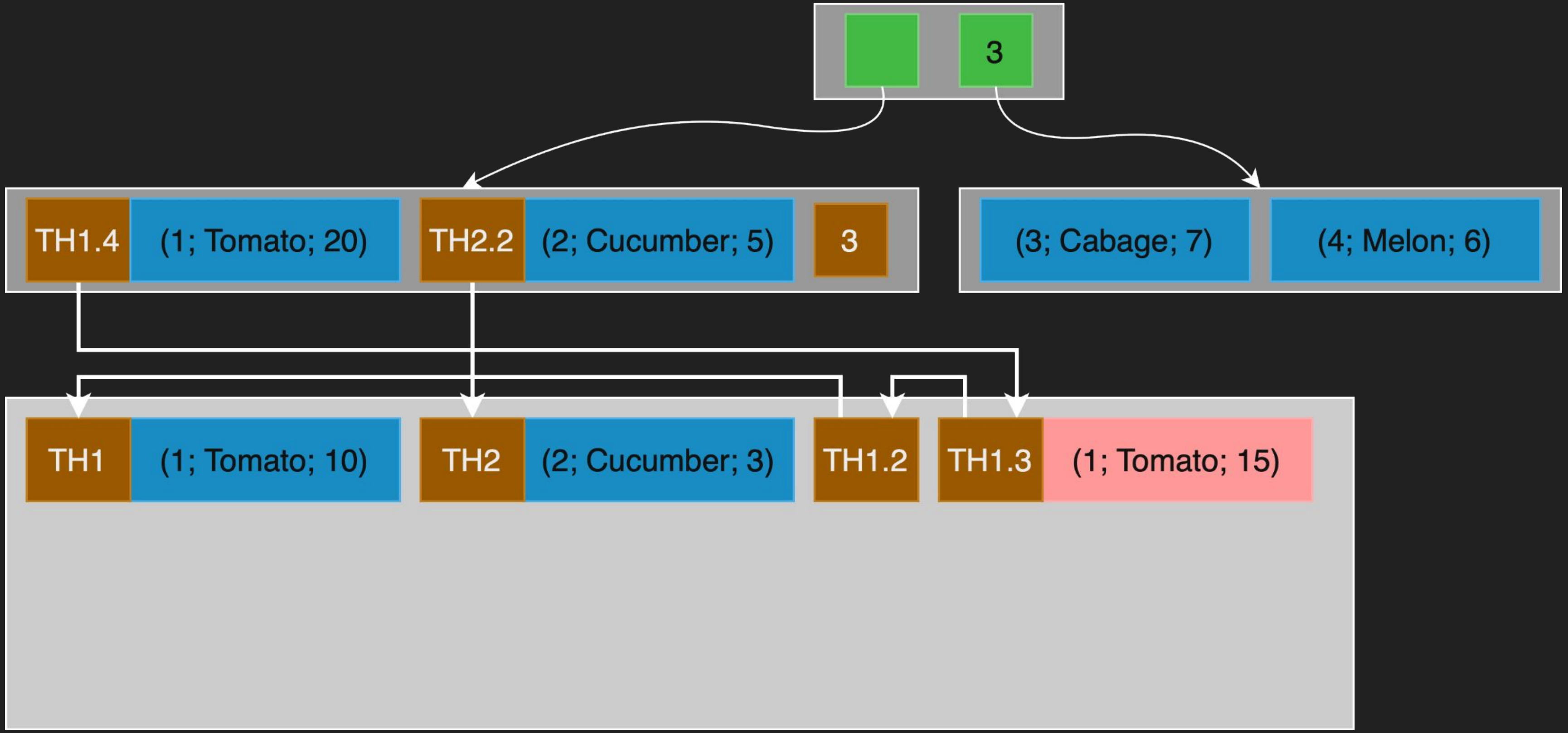
Undo logs in OrioleDB: UPDATE



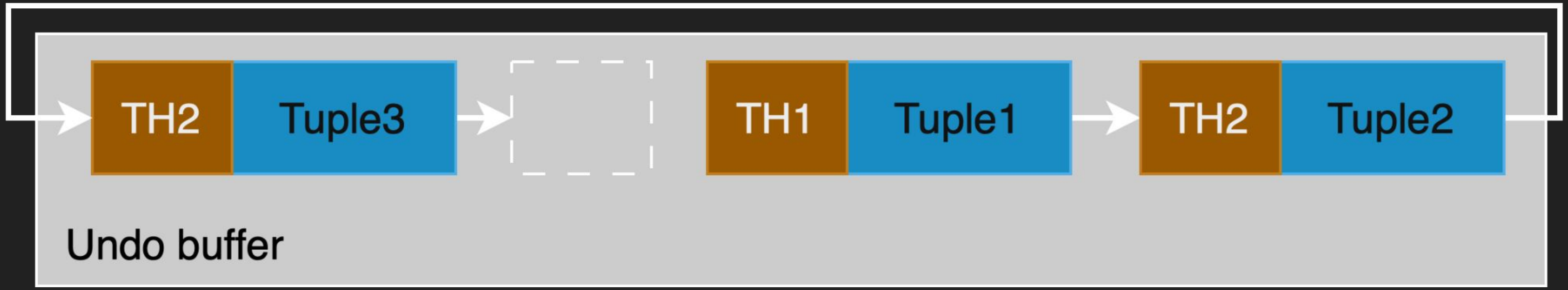
Undo logs in OrioleDB: DELETE



Undo logs in OrioleDB: INSERT

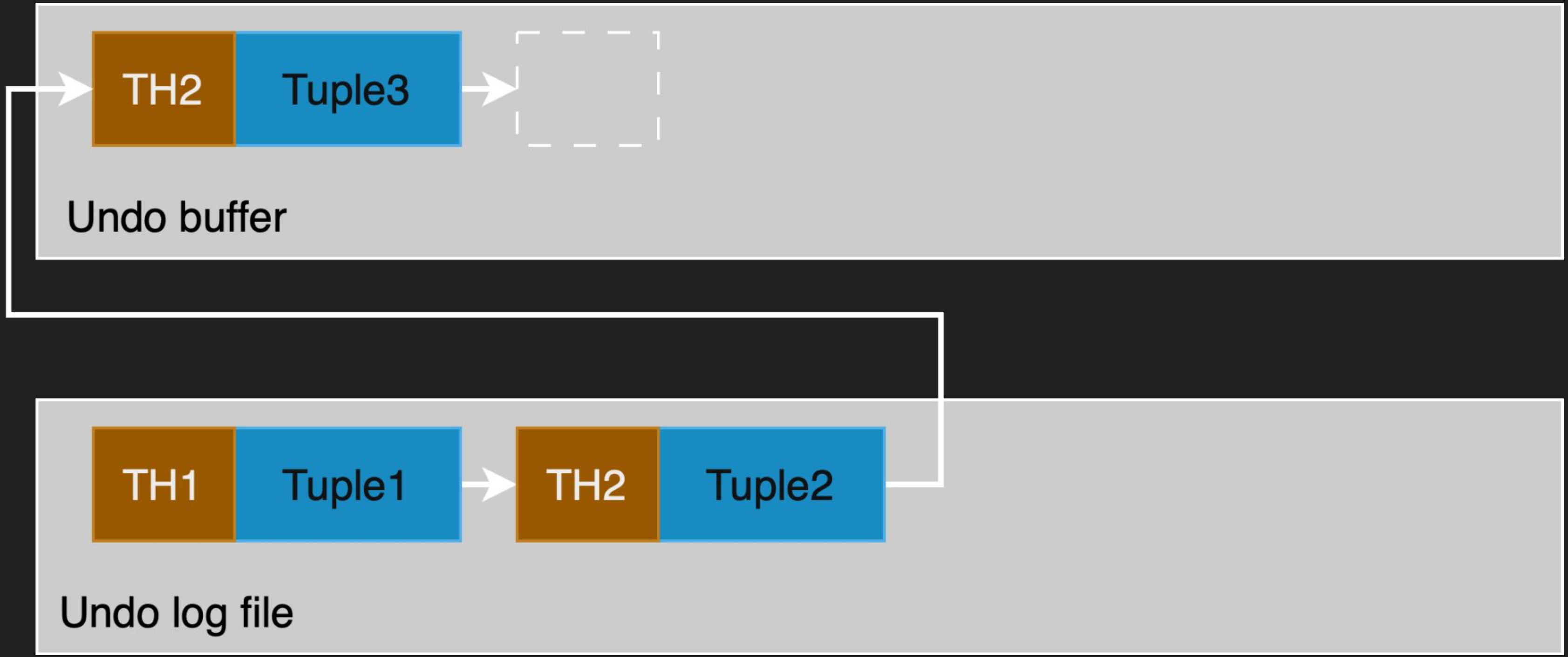


Undo logs in OrioleDB: Circular buffer



- `orioledb.undo_buffers` option to control buffer size
- 3 undo log buffers
  - row level
  - page level - used in split and merge operations
  - system metadata

Undo logs in OrioleDB: Circular buffer



Undo logs in OrioleDB: Summary

- In-place updates
- WAL logged
- Circular buffer for undo records
  - Undo records can be stored in files
- Row and page level undo records
  - Page level undo records are used in split and merge operations
- Transactional DDL, but not always MVCC-safe (same as in PostgreSQL)
- No need to have autovacuum process

Pitfalls:

- Rolling back can be more expensive

Links

Repo: github.com/orioledb/orioledb

Docs: www.orioledb.com/docs

Docs: supabase.com/docs/guides/database/orioledb

Docker: hub.docker.com/r/orioledb/orioledb

https://www.cs.cmu.edu/~pavlo/blog/2023/04/the-part-of-postgresql-we-hate-the-most.html

Thank you!