ByteHTAP: HTAP System with High Data Freshness and Strong Data Consistency

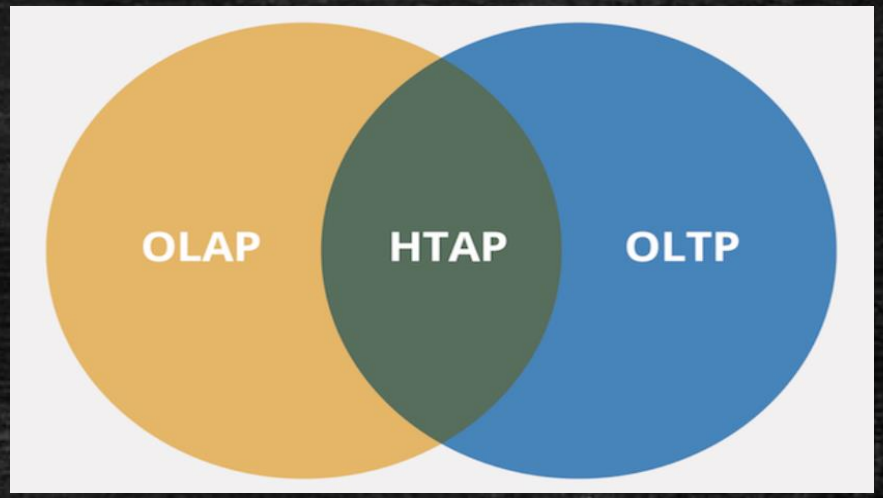
Jianjun Chen, Yonghua Ding, Ye Liu, Fangshi Li, Li Zhang, Mingyi Zhang, Kui Wei, Lixun Cao, Dan Zou*, Yang Liu*, Lei Zhang*, Rui Shi*, Wei Ding, Kai Wu, Shangyu Luo, Jason Sun, Yuming Liang* ByteDance US Infrastructure System Lab, *ByteDance, Inc*

Layout

- Introduction
- System requirements
- Architecture overview
- Consistency
- Freshness & Optimization
- Results

HTAP

- HTAP combines transactional and analytical processing in a single database system.
- Simplified architecture eliminates need for ETL.
- Instantly run analytical queries on fresh transactional data.
- Types
  - **Single Engine vs Muti Engine**
  - **Single Store vs Multi Store**



System Requirements

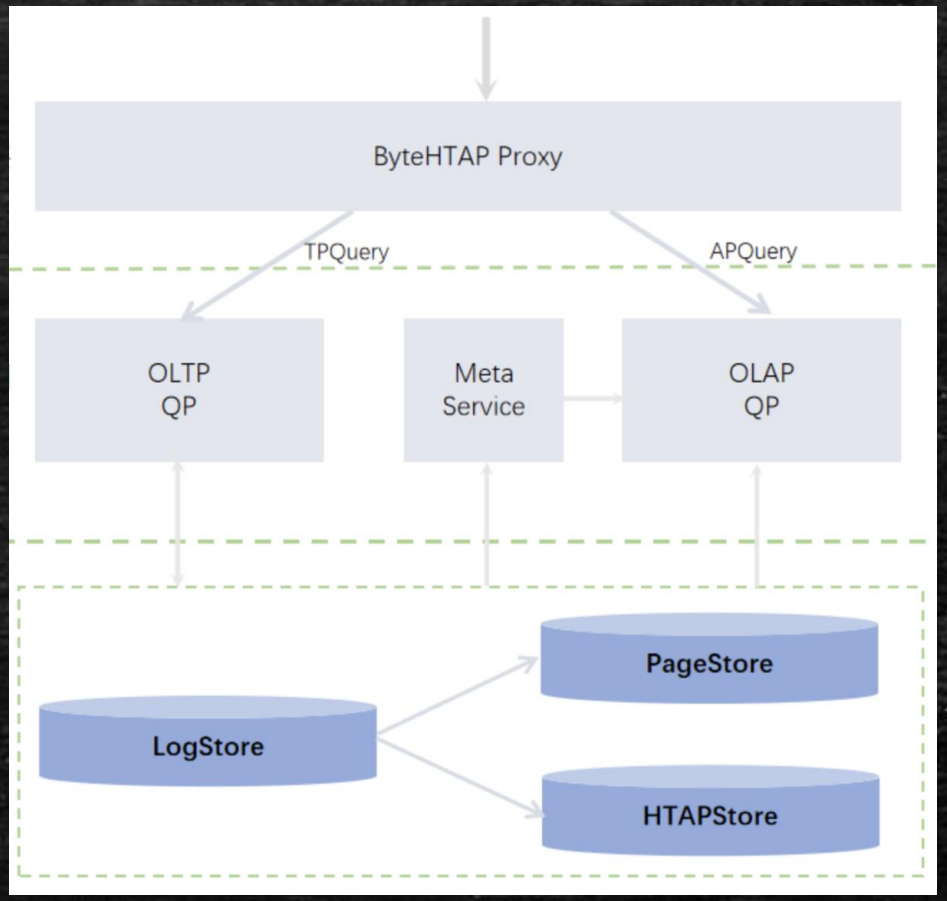
- Large scale – support scale up to petabytes.
- Real time – Comparable performance to individual OLTP / OLAP systems.
- Highly fresh data changes – Support OLAP querying on data as recent as 1 second delay.
- **Strong data consistency** - Native support for strong data consistency.

ByteDance's HTAP choices

- **Separate Engine**
- **Shared storage**

Architecture Overview

- Unified SQL Interface API
- **OLTP** – SQL Engine (ByteNDB) + Log Store + Page Store
- **OLAP** – Custom implementation of Flink
- **OLAP Store** – Delta Store + Base Store
- Metadata Service – Zookeeper
- Replication Framework



OLTP - ByteNDB

- Similar to **Amazon Aurora**
- **WAL** – Every DB action is loaded as Log. WAL is replicated across AZs
- **Log Store** persists redo logs and **Page Store** that stores versions of data pages.
- Each Log has an LSN, retained in sorted order. Proxy directs read requests via LSN.
- Quorum protocol for consistency
- Gossip protocol for replicas to fill the gaps in log sequence.

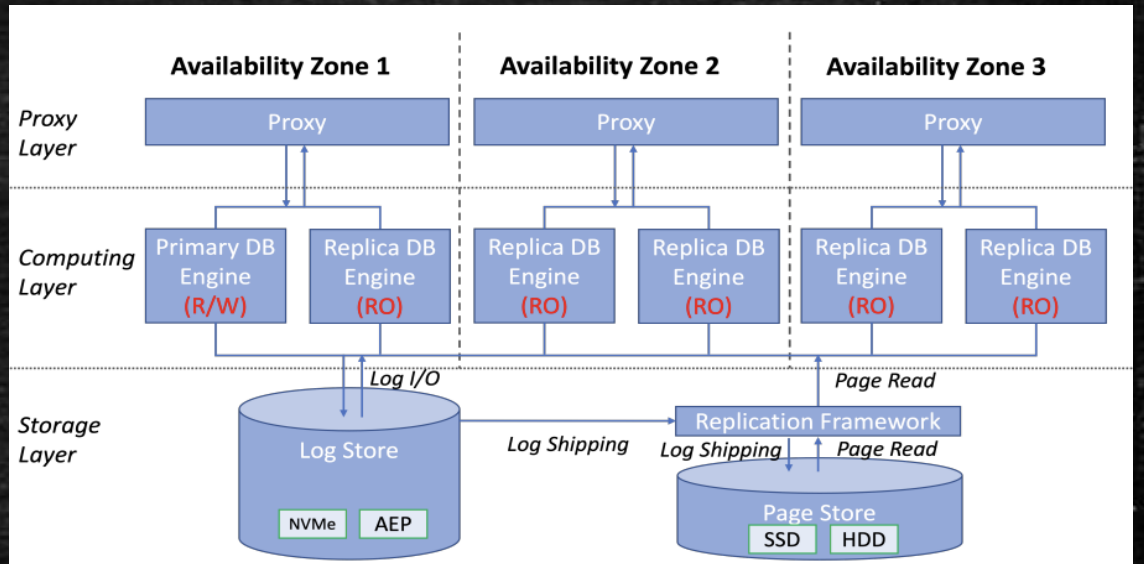
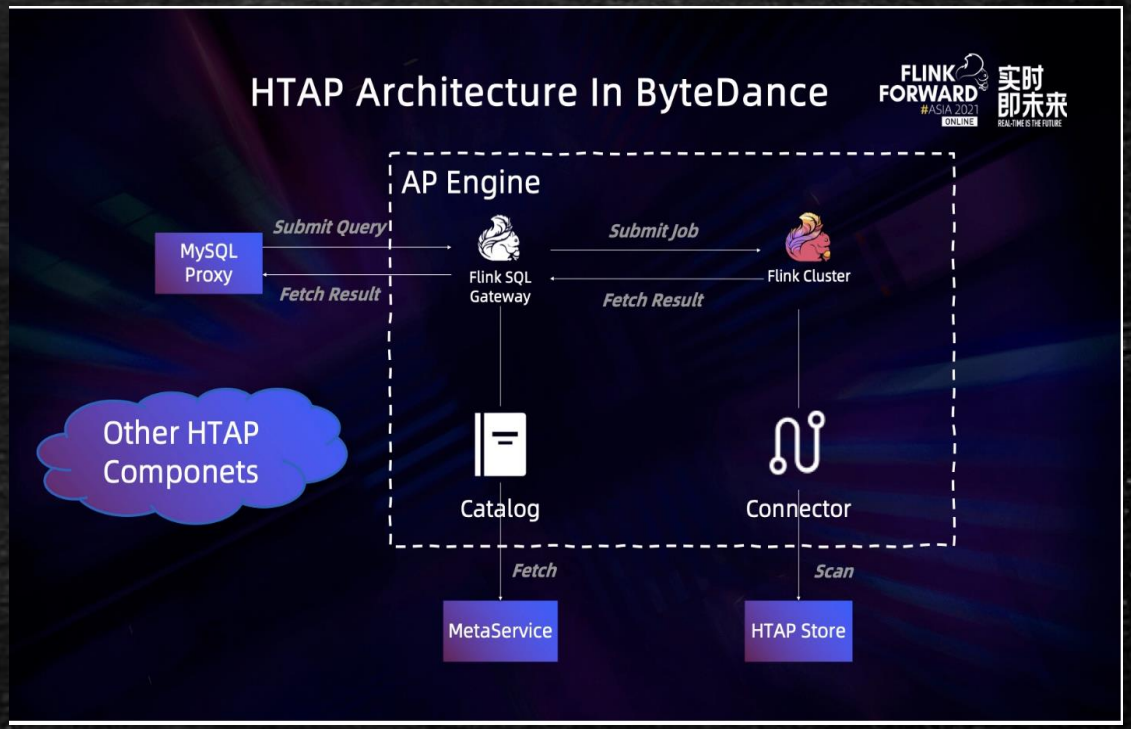


Figure 1: An illustration of ByteNDB architecture.

OLAP Engine - FLINK

- Flink is a stream / batch processing framework.
- Massive parallel processing within Flink cluster is adapted to support HTAP.
- Engineering team's Familiarity is one of the reason for adaption.
- Optimization – Support for Pushdown computation, Async Reads, Optimal parallelism module.



OLAP Store - Delta Store

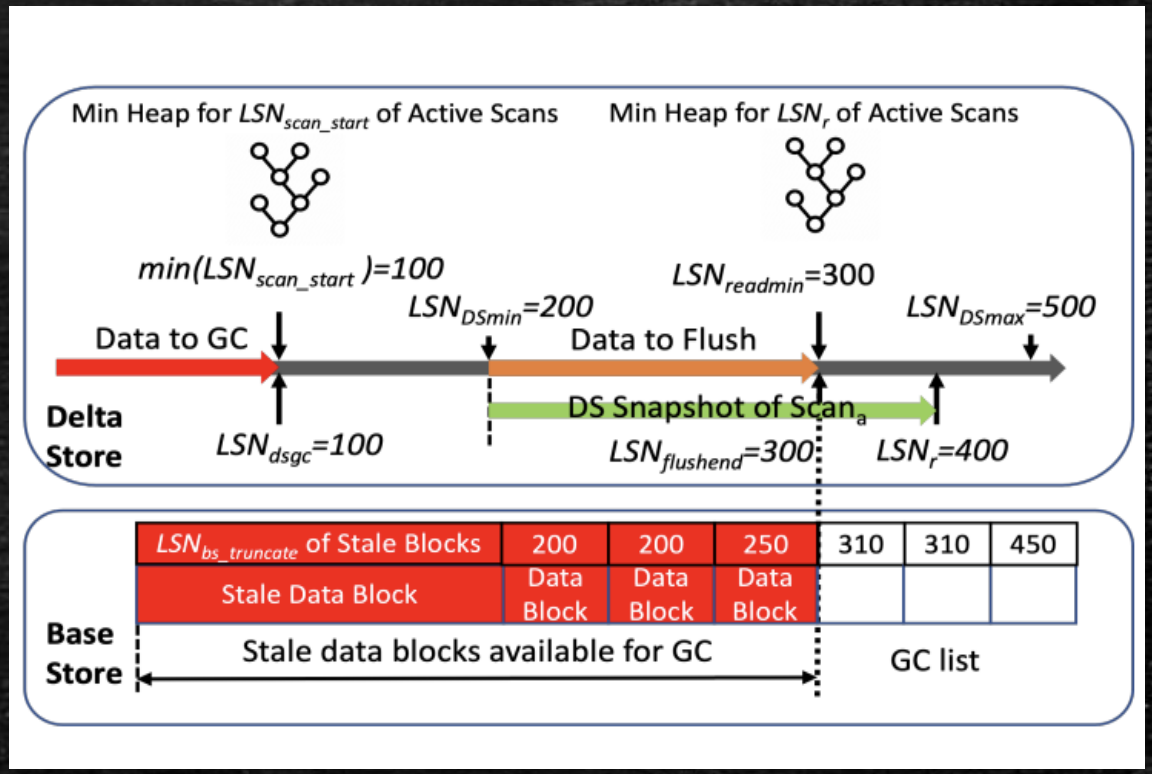
- Delta Store is an in-memory, row-format store.
- The OLAP table is partitioned and each partition has 3 replicas and each partition replica has a corresponding Delta Store.
- Insertion list & deletion list - Responsible for recording inserts and deletions in the order of LSN.
- Quick delete support - delete hash map maintained for efficient delete checks.
- Operations – **LogApply, Scan, Flush, Garbage Collection**.

Base Store

- Durable Columnar storage
- Stored as data blocks in Partitioned Attributes Across(PAX) storage format.
- Each data block includes metadata (min-max, stats - Allows for pushdown aggregation and filtering) and encoded data.
- Soft Deletes – Delete information stored in a bitmap (RocksDB).
- Uses Compaction and Garbage Collection to merge Data blocks and remove delete content via background operations.

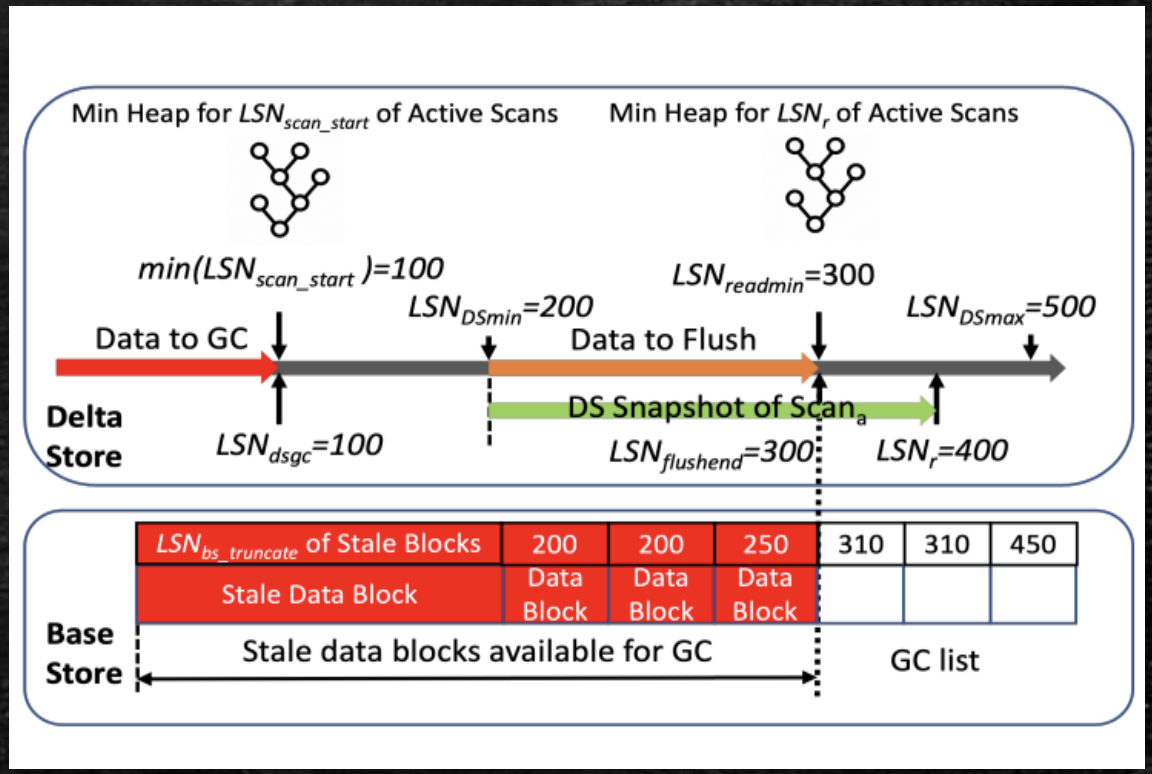
Consistency

- Strong consistency is guaranteed by ordering of the LSN during operations performed on Delta and base store.
- Scan – For a successful *scan*, the LSN of *scan* need's be in between the upper & lower bound of Delta store LSN.
- LogApply – Appends latest LSN logs onto to the delta store – No effect on any scan operation.



Consistency

- Flush - Appends new data on to base store ranging between LSN(min) to LSN(max). *Scan* and *flush* access the shared region by locks.
- Base store, Delta store GC and compaction operation run on blocks which aren't actively scanned.



Optimizations

- Efficient Log Replication
- Computation Pushdown – Predicate and Aggregate Pushdown
- Base store Delete Optimization – Delete bitmaps
- Delta scan Optimization – Delete HashMap
  - Lazy – All rows from base table scanned and delete applied at the end.
  - Eager – Create a selection criteria from the Map and they apply the query.
  - Cost Based – Choice made between the above based on *scan* cost.
- Flink Optimizations

Results

OLAP Latency vs #OLTP clients

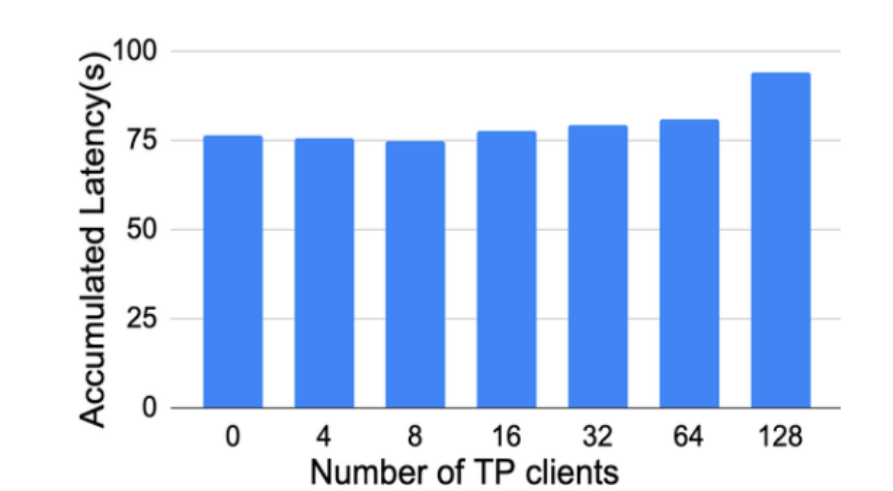


Figure 5: CH-benCHmark queries latency

Scan performance vs % of Flushed Data

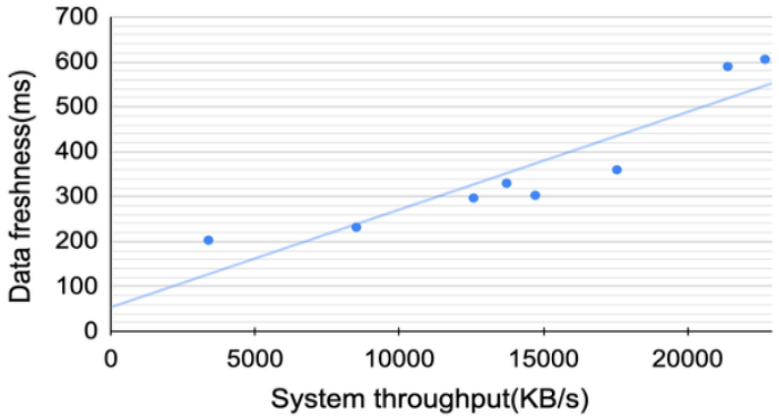
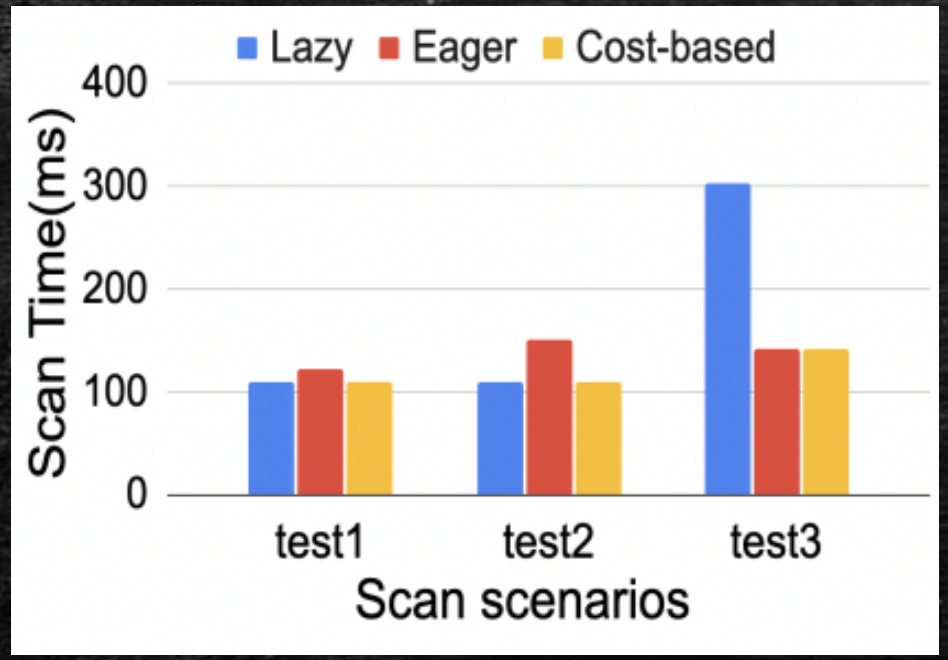


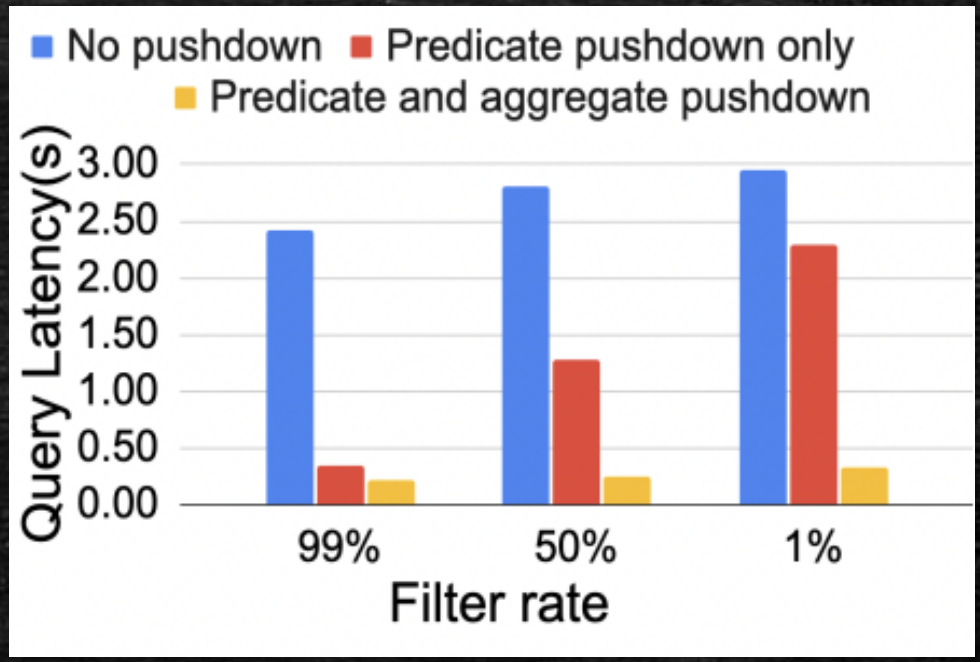
Figure 7: Data freshness with different throughputs

Results

Delete optimizations

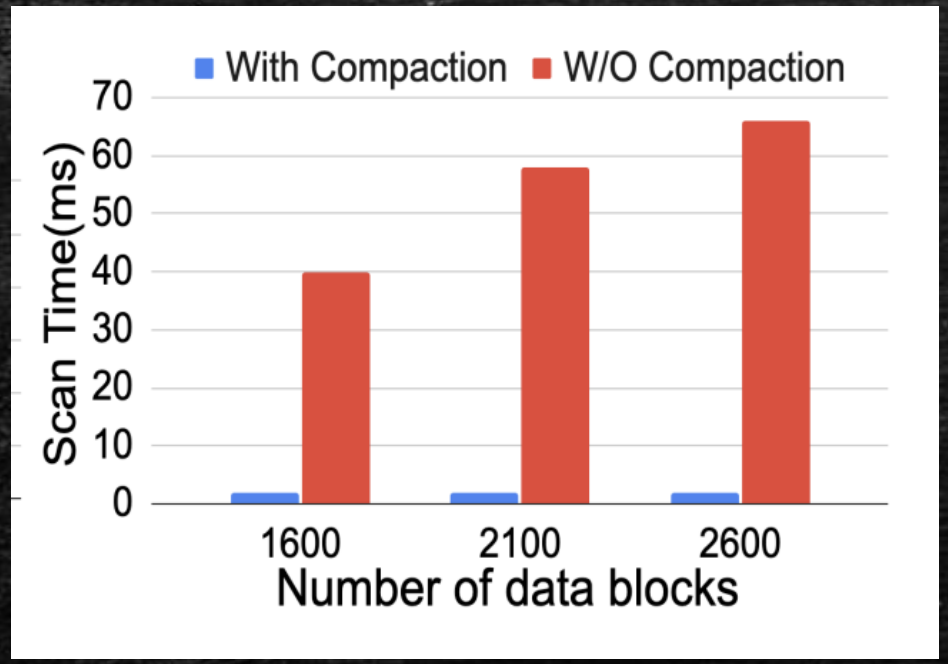


Computational Pushdown



Results

Scan Time & Compaction



Scan performance speedups

Table 2: Scan performance with different flush rates.

Flushed Data (%)	Scan Speedups
100%	2.90
50%	1.78
0%	1.00

Thank You

Questions ?