# ACM SIGMOD 2015 Programming Contest

Ismail Oukid (TU Dresden), Ingo Müller (KIT), Iraklis Psaroudakis (EPFL)

# Team *SimpleMind*

http://db.in.tum.de/sigmod15contest/

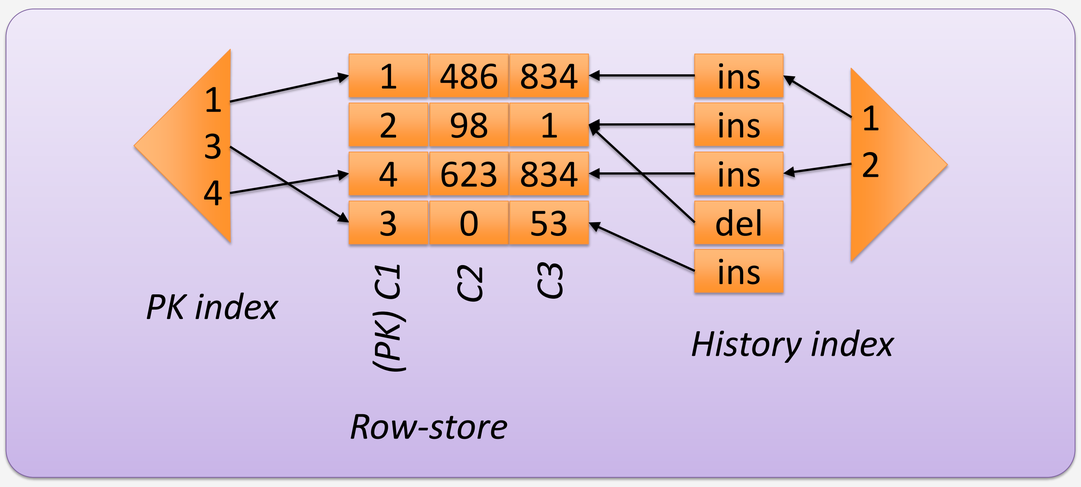
## 1. Task definition

A test driver issues a sequence of **transactions** (insert and delete statements), **validation queries** (select statements on data modified by a range of transactions), and **flush requests**. Upon a flush request, the system needs to output whether each validation **conflicts** (non-empty result set) or not.

## 2. Transaction processing

Each relation consists of:

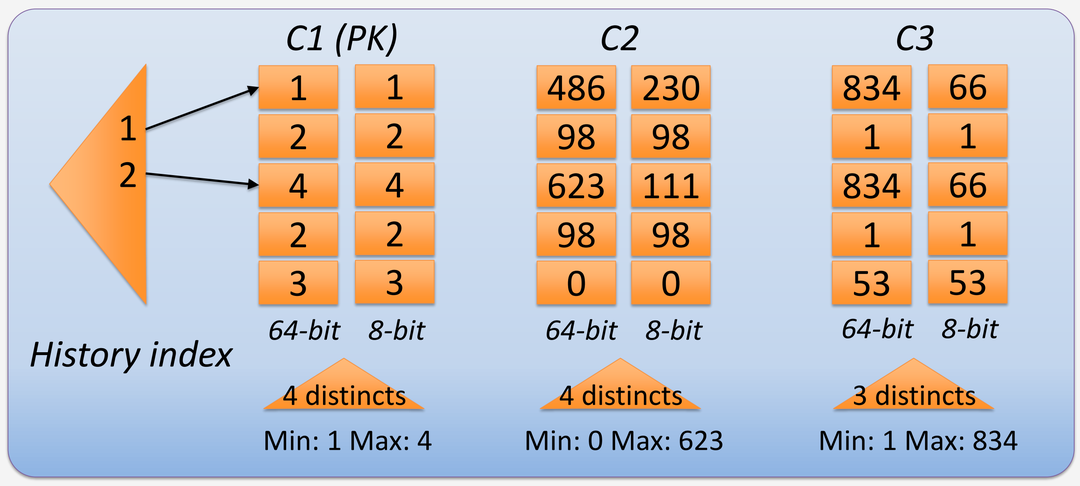
- A **row-store** of valid and deleted rows
- A **primary key** (PK) **index** (PK → valid rows) for fast updates
- A two-level "**history index**" for fast validation of single rows:
  Transaction ID (TX ID) → list of ptrs to modified rows → row




## 3. Data structures for validations

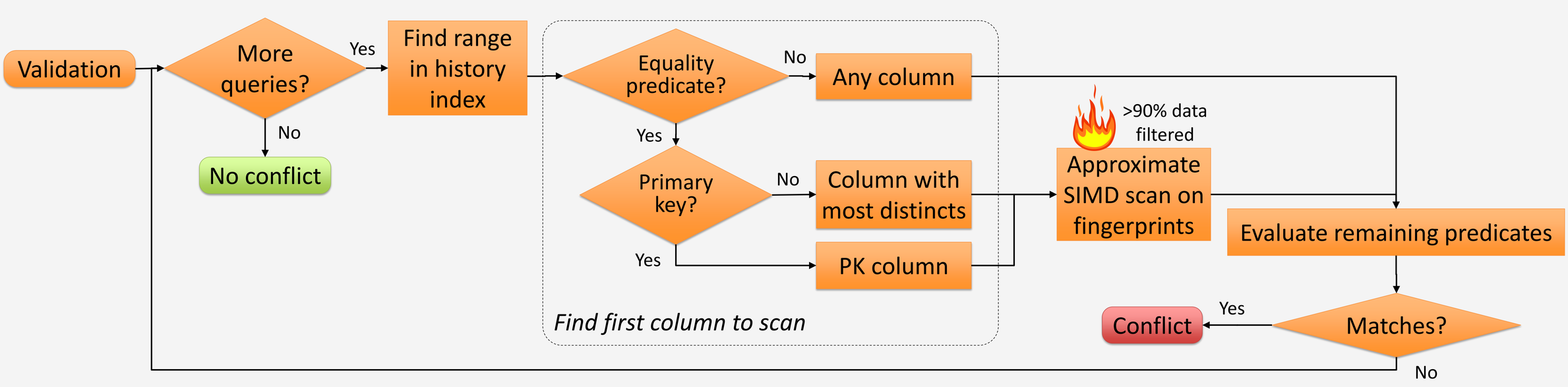
The modified rows are converted periodically to **column-wise** format (see 5). Additional metadata include:

- A single level "**history index**" (TX ID → offset of first modified row)
- 8-bit **fingerprint columns** (for superfast approximate scans)
- A sample of distinct values per column (to estimate selectivity)



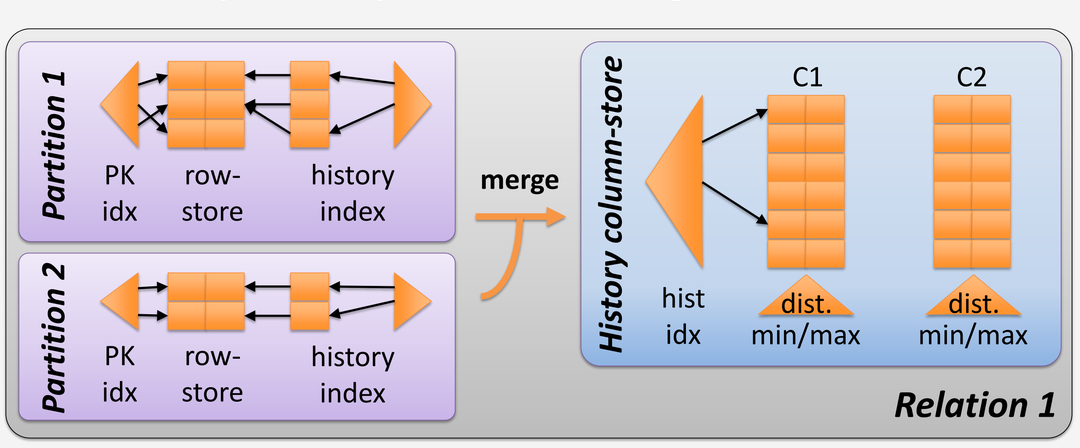

## 4. Validation processing

We scan the parts of column-store and row-store relevant for the transaction range of the validation query to find matching rows (conflicts). Based on heuristics, **fast and selective scans are executed first**. Furthermore, min/max values are used to skip a few predicates.




## 5. Parallelization: bulk-synchronous

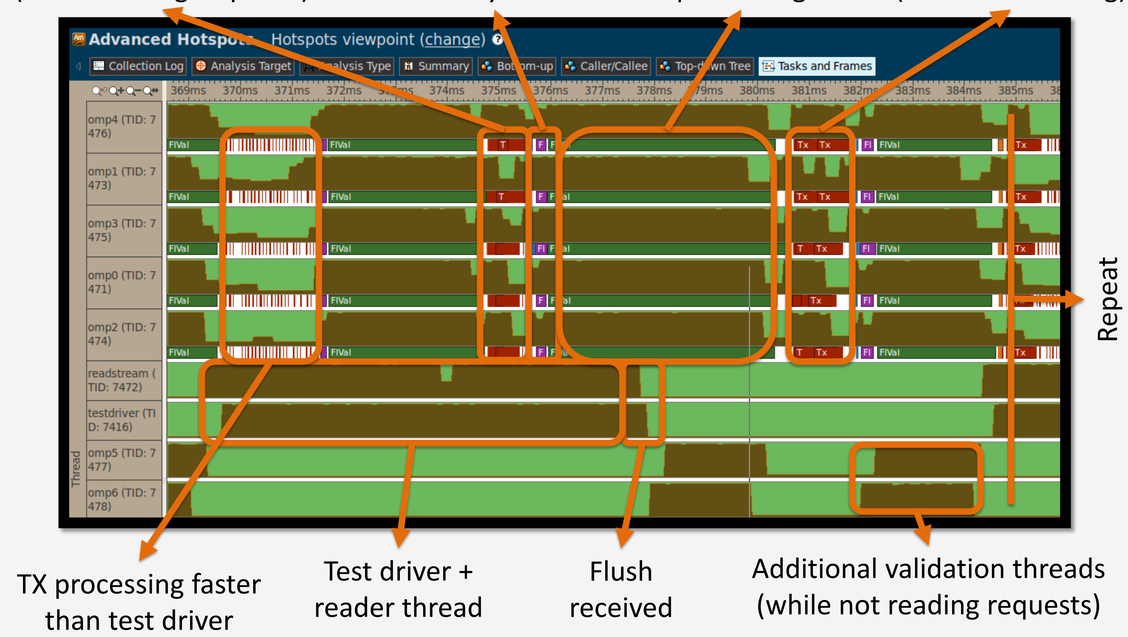
- The row-store is **hash-partitioned**. Each thread only executes **transactions** of its partition. Validations are queued.
- On flush request, the **partitions are merged into the column-store**.




- Afterwards, threads process **validations from the queue**, now accessing **all data structures** in a **read-only** fashion.
- Additional flushes to overcome slow test driver.

## 6. Runtime break-down

This is a screenshot of the execution flow from Intel VTune Amplifier.




SAP HANA Student Campus