

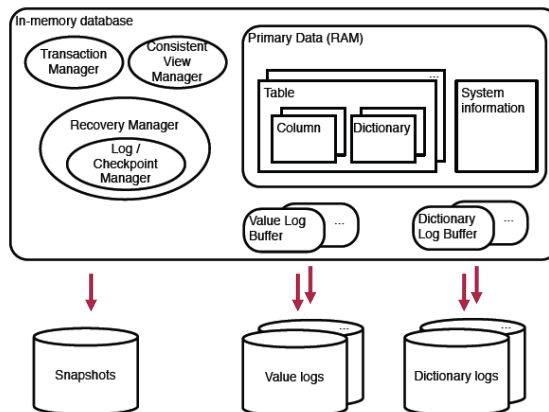
# Logging

**Prof. Hasso Plattner**

## What is Logging and Recovery?

- ❑ In-memory databases need to provide durability guarantees to be used in productive enterprise applications
- ❑ Potential strategies
  - **Logging**: writing log information to backup storage (covered here)
  - **Server replication**: run transactions simultaneously in primary and replicated database
- ❑ Recovery is the process of bringing a system back to a productive mode after a system failure
- ❑ Desirable characteristics of any recovery mechanism:
  - Minimal impact on running system
  - Fast recovery without any data loss

# Logging Infrastructure for an In-Memory Database



- Snapshots of primary data stores are written periodically (e.g. during merge process)
- Log files for updates are written between snapshots to recover the database
- Dictionary-compressed column logging allows parallel log file writing and recovery

3

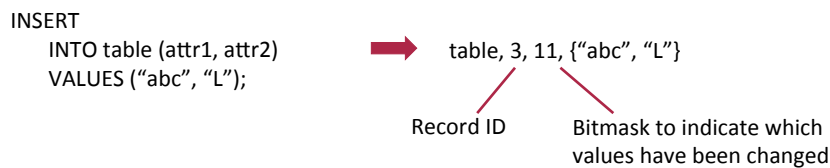
## Snapshots

- A snapshot writes the primary data of the in-memory database to persistent memory
- Purpose of snapshots:
  - Speed-up of recovery process, as only the log files after the snapshot need to be replayed
  - Creation of a system copy
- A snapshot is ideally written after each merge process
  - Snapshot contains new read-optimized store
  - Changes in the delta store are logged in log files

4

## Limited Applicability of Logical Logging

- Logical logging as obvious approach: write update SQL statements and their parameters (record ID and attribute values) to log file



- Main drawbacks for dictionary compressed columns:
  - Large log volume as multiple values are logged multiple times
  - With dictionary compression, parallel logging/recovery not possible as order of logs has to be preserved to recover the dictionary

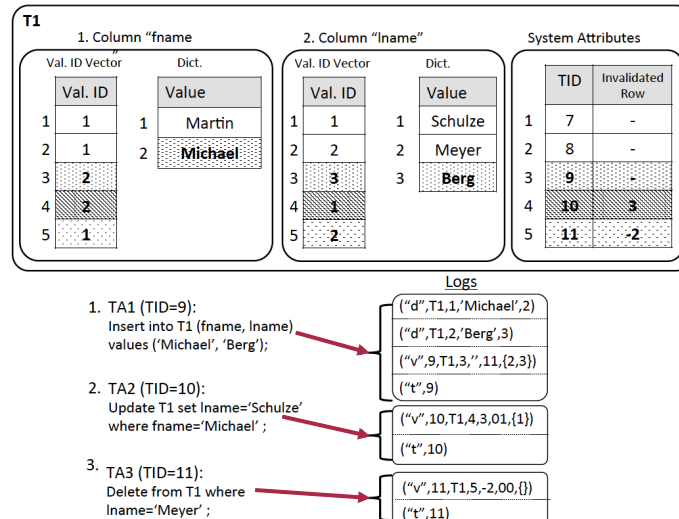
5

## Logging for Dictionary-Encoded Columns

- Key idea:
  - Store data dictionary-encoded and decouple dictionary inserts from the transactional context of a query
  - Allows to recover valueID vectors and dictionaries in parallel
  - Reduces log size for enterprise work loads
- Three different logs; separate log file for dictionary logs
  - Dictionary logs ("d")
    - Used to restore dictionaries per column
    - Required information: table ID, column index, value, value ID
  - Value logs ("v")
    - Used to restore value vector
    - Required information: transaction ID, table ID, row ID, invalidated row ID, bitmask about which attributes have been changed, value ID vector
  - Transaction logs ("t")
    - Are written after all corresponding dictionary and value log buffers are flushed
    - Required information: transaction ID (TID)

6

# Logging for Dictionary-Encoded Columns



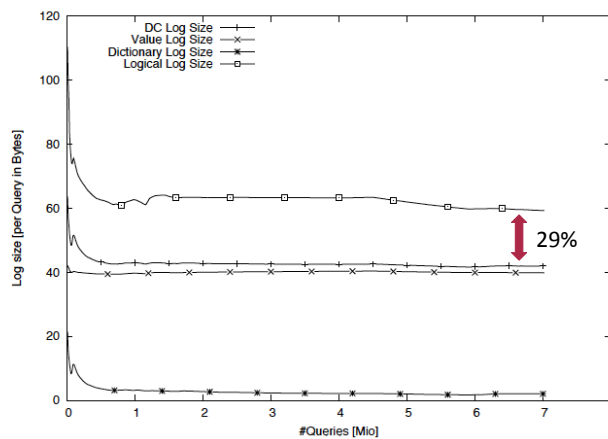
7

## When is Dictionary-Encoded Logging Superior?

- ❑ Large values, such as strings, are inserted multiple times
  - Dictionary compression avoids saving large values multiple times
- ❑ The number of distinct values is low
  - Low number of distinct values leads to less dictionary logs
- ❑ Only some of the distinct values occur frequently, while the majority is rare (Zipf distribution with a high alpha value)
  - Cumulated average log size per query is less dependent on number of logged queries
- ❑ Enterprise data sets typically show these data characteristics

8

# Log Size Comparison of Logical Logging and Dictionary-Encoded Logging



- 7 Mio. write operations on sales item table of productive enterprise system
- Overall reduction by 29%
- Reduction because of recurring values