

T-Store: Tunable Storage for Large Sequential Data

Kostas Zoumpatianos^{1,2}, Stratos Idreos¹ Themis Palpanas²

¹ Harvard University ² Paris Descartes University

Data Series

Sequences of points ordered over a dimension

Finance IoT Astronomy
Physics Neuroscience
Time Angle Position
Biology (i.e., DNA sequencing)
Paleontology

Data Layout

Position	100	...	210	250	...	800	850	...	995
SequenceID	20	...	16	35	...	90	38	...	74
Value1	900	78	54	31	...	33	42	68	55
Value2	2.4	4.6	7.6	9.8	...	2.1	3.1	7.2	6.6

Solution 1: Ordered by position

Solution 2: Ordered by sequence id

Q1: positions 105-210 for all sequences
Q2: all positions for sequence 90

DBMS Performance

Solution 1: Clustered index on position
Solution 2: Clustered index on seq. id

25% of positions 35% of positions 45% of positions
55% of positions 65% of positions 75% of positions

There is no perfect storage scheme

Design Continuum: From Fixed to Hybrid Layouts

Sequence first | Groups of series | Position first

Current Systems: Choose a fixed layout without considering the workload. Moreover, all series are stored in the same way since layouts are applied globally to all series.

How to achieve the best layout for a given workload and a given set of data series?

Our Solution: Hybrid Layouts with T-Store

Hybrid store: supports three different data layouts simultaneously

- Position-first
- Sequence-first
- Groups-of-series (sorted position-first)

Storage advisor: chooses the appropriate layout for each series

- Constructs a set of features that describe a user workload
- Partitions by solving an optimization problem using modeling

Smart grouping: Constructs groups-of-series for various applications

- Data mining (i.e., similarity search) by grouping similar shapes
- Data ingestion by grouping series that are updated together

Optimization

Loneliness Factor: Used to identify series usually queried in isolation. For a query Q that touches a set of series that also includes series X:

$$LF(Q, X) = \frac{\text{\# series in database}}{\text{\# series accessed by Q}}$$

Series Selectivity Factor: Captures how much of the total positions of a series are accessed by a query X that touches it. We compute this as follows:

$$SF(Q, X) = \frac{\text{\# points of series X selected by Q}}{\text{\# points of series X}}$$

Expected Variance

Cost Modelling

Position-first data layout:

$$c_{pfirst}(q_i, D) = \left(\log_f \sum_{s_i \in D} |s_i| \right) * c_{lat} * \alpha + \left[sel_{pts}^D(q_i) * \frac{r_{size}}{p_{size}} \right] * c_{thr} * \beta \quad (1)$$

Sequence-first data layout:

$$c_{sfirst}(q_i, D) = sel_{seq}(q_i) * (\log_f |D|) * c_{lat} * \alpha + \sum_{s_i \in q_i} \log_2(|s_i|) * c_{lat} + \left[sel_{pts}(q_i) * \frac{r_{size}}{p_{size}} \right] * c_{thr}$$

Optimization Problem

T-Store Architecture

Sequence Identifier Range: 1 - 15 | 15 - 35 | 35 - 50

Position-first layout: Index on positions. Big Group.

Group-of-series layout: Index on sequence identifiers. Group 1, Group 2, Group 3.

Sequence-first layout: Index on sequence identifiers. Series sorted to minimize overlap in positions across files.

Results

T-Store outperforms static layouts by intelligently partitioning data series and storing them using the appropriate data layout

FSMP: Part of series for which the workload that touches few series and many positions
MSFP: Part of the series for which workload that touches many series but few positions
MIX: Part of the series for which both access patterns are exhibited

T-Store identifies the optimal data layout for each query