W

Windows

Carlo Zaniolo
Computer Science Department, University of California, Los Angeles, CA, USA

Synonyms

Logical window = Time-based window; Physical window = Tuple-based windows

Definition

Windows were introduced as part of SQL:1999 OLAP Functions. For instance, given a sequence of bids we can use the following SQL:2003 statement to find the last 40 offers (the current offer and the previous 39) for item 0021:

```
SELECT itemID, avg(Offer)
OVER (ROWS 39 PRECEDING ORDER BY TIME)
FROM BIDS
WHERE ItemID=0021
```

When BIDS is instead a data stream, the "ORDER BY TIME" clause becomes redundant, and clauses such as "FOLLOWING" are often not supported in continuous query languages. However, these languages still provide a "PARTITION BY" clause (or the more traditional "GROUP BY" clause), whereby a user can specify that the average of the last 40 offers must be computed for all items, not just item 0021. In addition to entailing powerful and flexible analytical queries on ordered sequences and time-series, as in databases, windows on data streams play the key role of *synopses*, and are widely employed in this capacity. In particular, since it is not feasible to memorize unbounded data streams, window joins are used instead. In window joins, the newly arriving tuples in one data stream are joined with the recent tuples in the window of the other data streams (and symmetrically). The design and implementation of window-based extensions for aggregates and joins operators for data streams has generated significant research work [1–3].

Main Text

Traditional SQL aggregates are blocking (and thus not suitable for continuous queries on data streams) but their window versions are not. Therefore, the window concept is the cornerstone of many continuous query languages, where its functionality has also been extended with new constructs, such as *slides*, *tumbles*, and *landmark* windows, that are not in SQL:2003. In a sliding window of w tuples (seconds) the aggregate computed over w is returned for each new incoming tuple: when a slide of size s is also specified, then results are only returned every s tuples (seconds). With $s = w$ we have a tumbling window in which results are only returned at the end of each window. When $w/s = k$, then the window, and the computation of the aggregate, is partition into k panes [1]. A landmark window is one where an occurrence of some event of semantic significance, e.g. a punctuation mark [3], defines one or both endpoints. Efficient implementation requires delta computations that exploit the algebraic properties of aggregates – e.g., by increasing (decreasing) the current sum with the value from the tuple entering (leaving) the window – and architectures that consolidate the vast assortment of windows and aggregates at hand [1,2]. Windows provide the basic synopsis needed to support joins with limited memory. Special techniques are used to optimize multi-way joins, and response time for all joins. Further optimization issues occur when load-shedding is performed by either (i) using secondary store to manage overflowing window buffers, or (ii) dropping tuples from the windows in such a way that either a max-subset or a random sample of the original window join is produced. Aged-based policies, where older

tuples are dropped first, is preferable in certain applications. Sketching techniques are often used to estimate the productivity of tuples, whereby the least productive tuples are dropped first [3]. The same techniques can also be used to estimate duplicates in `DISTINCT` aggregates; reservoir-sampling inspired techniques have instead been proposed for aggregates where duplicates are not ignored.

Cross-references

- ► Data Sketch/Synopsis
- ► Join
- ► One-Pass Query Processing
- ► Punctuations
- ► SQL
- ► Stream Processing

References

1. Li J. et al. Semantics and evaluation techniques for window aggregates in data streams. In SIGMOD Conference. 2005, pp. 311–322.
2. Bai Y. et al. A data stream language and system designed for power and extensibility. In CIKM. 2006, pp. 337–346.
3. Maier D., Tucker P.A., and Garofalakis M. Filtering, punctuation, windows and synopses. In Stream Data Management, Vol. 30, N. Chaudhry, K. Shaw, M. Abdelguerfi (eds.). Kluwer, Dordecht, 2005, pp. 35–56.