Big Sequence Management

Karima Echihabi
IRDA, Rabat IT Center,
ENSIAS, Mohammed V Univ.
karima.echihabi@gmail.com

Kostas Zoumpatianos
Harvard University
kostas@seas.harvard.edu

Themis Palpanas
LIPADE, Université de Paris &
French University Institute (IUF)
themis@mi.parisdescartes.fr

***Abstract*—Data series are a prevalent data type that has attracted lots of interest in recent years. Specifically, there has been an explosive interest towards the analysis of large volumes of data series in many different domains. This is both in businesses (e.g., in mobile applications) and in sciences (e.g., in biology). In this tutorial, we focus on applications that produce massive collections of data series, and we provide the necessary background on data series storage, retrieval and analytics. We look at systems historically used to handle and mine data in the form of data series, as well as at the state of the art data series management systems that were recently proposed. Moreover, we discuss the need for fast similarity search for supporting data mining applications, and describe efficient similarity search techniques, indexes and query processing algorithms. Finally, we look at the gap of modern data series management systems in regards to support for efficient complex analytics, and we argue in favor of the integration of summarizations and indexes in modern data series management systems. We conclude with the challenges and open research problems in this domain.**

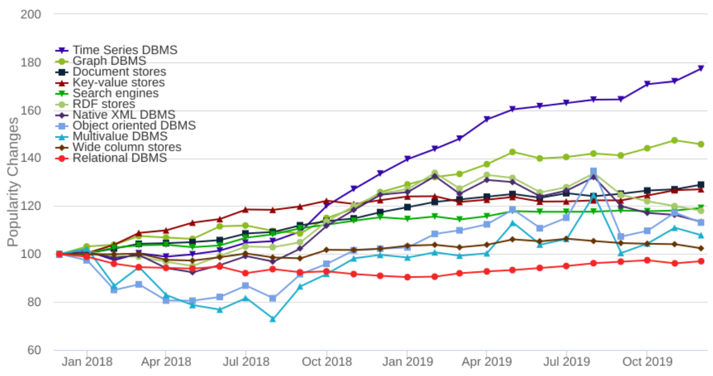

Fig. 1. DBMS category popularity change trend [2]

I. Introduction

In various scientific and industrial domains analysts are required to measure quantities as they fluctuate over a dimension; these values are commonly called *data series* or *sequences*. The dimension over which data series are ordered depends on the application domain and can have various diverse physical meanings. By far, the most common dimension over which data are ordered is time. In this case, we specifically talk about *time series*. Other applications though, produce series ordered over position (DNA sequences), mass (mass spectrometry) or angle (shapes). In all cases, data have to be captured, stored and analyzed as series rather than individual values.

Applications range from forecasting methods to correlation analysis, summarization, representation methods, sampling, outlier detection and more [6]–[8], [32], [40]. Moreover, it is not unusual for applications to involve numbers of sequences in the order of hundreds of millions to billions [1], [3]. As a result, analysts are more frequently than ever deluged by the vast amounts of data series that they have to filter, process and understand. Consider for instance, that for several of their analysis tasks, neuroscientists are currently reducing each of their 3,000 point long sequences to a single number (the global average) in order to be able to analyze their huge datasets [1]. In astronomy, there are currently available more than 70TB of spectroscopic sequence data from 200 million sky objects, collected by the Sloan Digital Sky Survey [3], allowing scientists to study the universe. These data have to be processed and analyzed, in order to identify patterns, gain insights, and detect abnormalities.

Recent advances in domains such as cloud computing and data centers, IoT and smart cities, self-driving cars and communications, generated a tremendous interest in developing specialized systems able to manage and mine data series. This is evident both by industrial [17], [41] and academic interest [5], [25], as well as through popularity studies [2], where time series management systems gather the most intense interest change over the last two years, as shown in Figure 1.

Our goal through this tutorial is to describe the current state in data series management, including applications, query types and data types, complex analytic algorithms, their components and their implementation in modern data systems. Further on we will explore how modern techniques can be leveraged to speed up complex analytical pipelines, and take a glimpse on how these techniques can be improved by applying machine learning.

II. Projected Audience and Expected Background

Modern day research has entered an age where the amount of recorded data is increasing exponentially. The analysis of the time-series data associated with multiple fields is now beginning to push both computational power and resources to their limit. This tutorial is for both data analysts and researchers, and will focus on recent advances in academia and industry in the area of time-series analysis. The tutorial aims at fostering collaborations between the data management community and data science practitioners in various domains, including networking and communications, as well as on gathering awareness and interest on the domain of data series analytics.

In the material that we will present, we will include the background necessary in order to follow the entire presentation, as well as technical details of existing solutions, and discussions of drawbacks, open problems and challenges. Therefore, both experts and newcomers in the area will be able to follow the material and benefit from the tutorial.

III. Tutorial Scope

In this 90 minute tutorial, we take a holistic look at the problem of managing and analyzing very large collections of data series, discuss the state-of-the-art and pinpoint the opportunities for optimizing complex query execution.

[Introduction and Foundations] We will start by looking at some foundational aspects of data series management. Those include the *data characteristics*, the query workloads, and the *specialized data structures* used to index sequential data. Data series can be categorized under many dimensions: i.e., the way that data arrive: streaming vs static, the lengths of data series: fixed vs variable length per series, the way that points are sampled: fixed intervals vs variable sampling intervals, and the presence of uncertainty in their values.

In terms of workloads, we will then look at various applications and query patterns that recur in each one of those. Specifically, we will discussing both simple Selection-Projection-Transformation (SPT) queries, where analysts filter based on data properties (e.g., thresholds) or meta-data values, as well as complex data mining (DM) analytics, like clustering, outlier detection and more [38]. We will look at the core component of advanced analytics, which is similarity search, and look at the different flavors of this problem. Those include whole matching vs sub-sequence matching, exact vs approximate similarity search, as well as various distance measures that are commonly used in practice. Finally, we will briefly talk about the different data structure categories that exist, and how they are used to organize and retrieve data in each one of the aforementioned query patterns.

[Complex Analytics] We will dive in analytics like outlier detection [12], [16], frequent pattern mining [48], clustering [26], [49], [51], [61], and classification [13]. Such analytics involve a series of operations that are performed in a pre-processing step (e.g., sliding windows, normalization, interpolation, etc.), as well as operations that are repeated in the context of an iterative algorithm (e.g., similarity search). We will discuss these operations, and pinpoint the ones that can be optimized at the database kernel level. Such operations include sliding windows, normalization, interpolation, and various transformations such as dft that are specific to each algorithm. During the iterative part of these analytics, multiple similarity search operations need to be performed. This is useful for finding series within a given radius from a centroid in clustering, or for identifying distances from a given model in anomaly detection and classification, but also for retrieving patterns in frequent pattern mining. All of these operations can be implemented externally, in the application side. However, since some of them are data-intensive, pruning or incremental computation can significantly improve their performance. For this reason, performing them at the database level can provide large improvements in terms of execution time. We will focus on similarity search as such an example, being a crucial and expensive component of most mining algorithms, and motivate a deep-dive at its characteristics and scalable implementations.

[Systems for Data Series Management] We will then look at current state-of-the-art systems, describing their storage layers and data structures, as well as how they implement the aforementioned data manipulation operations. In particular, we will both look at systems that have been specifically designed to support sequential data, as well as systems that have been adapted to support them.

Specialized systems either utilize custom storage layers, or existing solutions. Common off-the-shelf storage systems are log-structure merge tree (LSM) based engines like RocksDB and LevelDB, and distributed systems such as HBase. Custom engines utilize domain-specific compression, indexing and data partitioning to increase efficiency. They support both simple and complex analytical queries and some of the systems offer encryption and distributed query processing.

Beringei [41] is developed by Facebook, it has a custom in-memory storage engine. It compresses and organizes data in a series per series scheme. CrateDB [15] partitions data in chunks, stores them in a distributed file system, and indexes them using Apache Lucene. InfluxDB [24] uses Time-Structured Merge Trees (LSM tree variant), logging data on disk as they arrive, and periodically merge-sorting overlapping time-stamps. Prometheus [45] is based on the Beringei ideas. QuasarDB [46] utilizes either RocksDB or Hellium [23]. Riak TS [50] supports both LevelDB or Bitcask, which is a custom log structured hash table. Timescale [57] is a Postgres extension. It partitions time series both in groups of series as well as in distinct time segments. It then provides an abstraction of a single table. Finally, various systems such as OpenTSDB [37], Timely [56] (concentrated on security) and Warp10 [60] are developed on top of HBase.

All the aforementioned systems support range scans in the positions, aggregation functions and filtering. Beringei additionally supports correlation queries through a brute force implementation. Crate supports geospatial queries. InfluxDB supports queries like moving averages, prediction, transformations, etc, and Timescale supports gap filling.

[Advanced Techniques for Optimizing Analytics] We will present techniques for speeding up similarity search, which plays a central role in several algorithms related to complex data series analytics, and discuss opportunities for integrating such techniques in modern data series management systems. Previous work on similarity search has proposed the use of spatial indexes such as R-Trees with DFT [4], [47] and DHWT [11]. Specialized indexes are based on domain specific summarizations. Examples include DS-Tree [59], iSAX [39], [54], iSAX 2.0 [9], iSAX2+ [10], ADS+ [64], [64], SFA [53], Coconut [28], [29], ULISSE [30], [31], DPiSAX [62], [63], ParIS+ [42], [44], MESSI [43]. Apart from exact indexes, there are also various approximate index structures proposed in the literature. Those include methods based on hashing [27], [55], sketches and grid indexes [14], and kNN-Graphs [33], [34]. Recent studies [19], [20] have compared several data series and high-dimensional similarity search methods under a common framework, revealing multiple promising future research directions, which we will analyze.

[Challenges and Conclusions] Massive data series collections are becoming a reality for virtually every scientific and social domain. This leads to the need of designing and developing general-purpose Data Series Management Systems, able to cope with big data series, that is, very large and fast-changing collections of data series, which can be heterogeneous (i.e., originate from disparate domains and thus exhibit very different characteristics), and which can have uncertainty in their values (e.g., due to inherent errors in the measurements). These systems should have data series indexes and summarizations integrated into their engines, so as to speedup the time-intensive operations of complex analytics pipelines, and support interactive exploration of big data series. To this end, progressive analytics operators would also be very useful [21], [22], [58]. At the same time, the role that deep learning techniques can play should be studied in more detail, especially with regards to similarity search [18] and query optimization. Finally, there is a pressing need for developing data series specific benchmarks [65], [66] able to stress test index structures in a principled way.

A. *Outline*

Next, we report the outline of the tutorial (duration: 3 hours).

1) **Introduction, Motivation, and Foundations (30 min)**
   - Domains that produce massive sequence collections and application examples
   - Definition of data series and their properties (streaming vs static, fixed length vs variable length, fixed vs arbitrary sampling rate, univariate vs multivariate, etc.)
2) **Complex Analytics (45 min)**
   - Different types of data series queries (simple queries vs complex analytics, subsequence vs whole-matching, approximate vs progressive vs exact)
   - Commonly used complex analytics: clustering, classification, anomaly detection, frequent pattern mining, forecasting, etc.
3) **Advanced Techniques for Optimizing Analytics (60 min)**
   - Summarizing and compressing data series: reducing memory and computational overhead in loss-less or lossy ways.
   - Similarity search: Exact and approximate methods (iSAX2+, ADS, DS-Tree, SFA, Coconut, ULISSE, DPiSAX, ParIS, MESSI, kNN-Graphs, LSH, Sketches, etc.) and alternatives (scan acceleration/progressive search)
   - Results and lessons from an extensive experimental comparison of similarity search methods
4) **Systems for Data Series Management (30 min)**
   - Historical overview of data series management (Illustra, SEQ, TS-SQL, Rasdaman, SciDB, etc.)
   - Modern data series management systems (Beringei, CrateDB, InfluxDB, QuasarDB, RiakTS, Timescale, etc.)
     - Supported operators (e.g., sliding windows, similarity search, etc.) and their implementation
     - Storage layer description (data structures, algorithms, etc.) and its design goals (read optimized, write optimized, etc.)
     - Strong and weak points of each system
   - In-depth discussion of current bottlenecks and opportunities for optimization
5) **Challenges and Conclusions (15 min)**
   - Open problems in data series management systems
   - Open problems in complex data series analytics
   - Opportunities and challenges for deep learning techniques

Relation to Previous Tutorials. We note that previous tutorials in the domain of data series have either concentrated on time series mining algorithms [36], [52], or specifically on the characteristics of different time series similarity measures [35], with no reference to index data structures and data series management systems, which are now becoming popular and necessary for handling the increasingly larger data series collections in different applications across many domains.

This is the first tutorial that looks at the existing full stack of a data series management system, and how state of the art techniques can be combined to enable complex analytics on large data series collections.

IV. Presenters

Karima Echihabi is a PhD student with University of Paris (France) and Mohammed V University (Morocco). Her research interests lie in data analytics and time series, and has performed an extensive analysis of data series indexes. She holds a Masters Degree in Computer Science from the University of Toronto and has worked as a software engineer in the Windows team at Microsoft, Redmond (USA), and the Query Optimizer team at the IBM Toronto Lab (Canada).

Kostas Zoumpatianos is a Marie Curie Individual Fellow affiliated with Harvard University (USA) and University of Paris (France). He holds a PhD in Computer Science from the University of Trento (Italy). He has visited Cornell University (USA) and worked as a Software Engineer. His research involves the domains of data series management, and self-designing and adaptive data systems. His work has been published in top international database conferences and journals.

Themis Palpanas is a Senior Member of the French University Insitute (IUF), and a professor of computer science at the University of Paris (France). He has worked at the IBM T.J. Watson Research Center, the University of California at Riverside, Microsoft Research and IBM Almaden Research Center. He is the author of nine US patents, has received three best paper awards and the IBM SUR award, has been Associate Editor for PVLDB 2019 and 2017, and General Chair for VLDB 2013, and is currently serving as Associate Editor for TKDE, and Editor in Chief for BDR. He has been working in the fields of Data Series Management and Analytics for more than 15 years, and has developed several of the state of the art techniques. He has delivered 12 tutorials in top international conferences, including data management (VLDB/SIGMOD), information retrieval (SIGIR), and the Web (WWW).

Acknowledgments. Work supported by program Investir l'Avenir and University of Paris IDEX Emergence en Recherche ANR-18-IDEX-0001, EU project NESTOR (MSCA #748945), FMJH Program PGMO in conjunction with EDF and THALES, and Institut Universitaire de France (IUT).

REFERENCES

[1] Adhd-200. http://fcon_1000.projects.nitrc.org/indi/adhd200/.

[2] Db-engines. https://db-engines.com/en/ranking_categories.

[3] Sloan digital sky survey. https://www.sdss3.org/dr10/data_access/rvolume.php.

[4] R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient similarity search in sequence databases. In *FODO*, 1993.

[5] A. Bader, O. Kopp, and M. Falkenthal. Survey and comparison of open source time series databases. In *BTW*, 2017.

[6] A. J. Bagnall, R. L. Cole, T. Palpanas, and K. Zoumpatianos. Data series management (dagstuhl seminar 19282). *Dagstuhl Reports*, 9(7), 2019.

[7] P. Boniol, M. Linardi, F. Roncallo, and T. Palpanas. Automated anomaly detection in large sequences. In *ICDE*, 2020.

[8] P. Boniol and T. Palpanas. Series2Graph: Graph-based Subsequence Anomaly Detection for Time Series. *PVLDB*, 2020.

[9] A. Camerra, T. Palpanas, J. Shieh, and E. J. Keogh. isax 2.0: Indexing and mining one billion time series. In *ICDM*, 2010.

[10] A. Camerra, J. Shieh, T. Palpanas, T. Rakthanmanon, and E. J. Keogh. Beyond one billion time series: indexing and mining very large time series collections with isax2+. *KAIS*, 39(1):123–151, 2014.

[11] K. Chan and A. W. Fu. Efficient time series matching by wavelets. In *ICDE*, 1999.

[12] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15, 2009.

[13] Y. Chen, E. K. Garcia, M. R. Gupta, A. Rahimi, and L. Cazzanti. Similarity-based classification: Concepts and algorithms. *J. Mach. Learn. Res.*, 10:747–776, June 2009.

[14] R. Cole, D. E. Shasha, and X. Zhao. Fast window correlations over uncooperative time series. In *KDD*.

[15] Crate. CrateDB: Real-time SQL Database for Machine Data & IoT, 2018.

[16] M. Dallachiesa, T. Palpanas, and I. F. Ilyas. Top-k nearest neighbor search in uncertain data series. *PVLDB*, 8(1), 2014.

[17] L. Dannecker, G. Gaumnitz, B. Ni, and Y. Cheng. Multi-representation storage of time series data, 2015. US Patent 20170161340A1.

[18] K. Echihabi, K. Zoumpatianos, and T. Palpanas. Scalable Machine Learning on High-Dimensional Vectors: From Data Series to Deep Network Embeddings. In *WIMS*, 2020.

[19] K. Echihabi, K. Zoumpatianos, T. Palpanas, and H. Benbrahim. The lernaean hydra of data series similarity search: An experimental evaluation of the state of the art. *PVLDB*, 12(2), 2018.

[20] K. Echihabi, K. Zoumpatianos, T. Palpanas, and H. Benbrahim. Return of the Lernaean Hydra: Experimental Evaluation of Data Series Approximate Similarity Search. *PVLDB*, 2019.

[21] A. Gogolou, T. Tsandilas, K. Echihabi, T. Palpanas, and A. Bezerianos. Data Series Progressive Similarity Search with Probabilistic Quality Guarantees. In *SIGMOD*, 2020.

[22] A. Gogolou, T. Tsandilas, T. Palpanas, and A. Bezerianos. Progressive similarity search on time series data. In *Proceedings of the Workshops of the EDBT/ICDT 2019 Joint Conference, EDBT/ICDT*, 2019.

[23] Hellium. Hellium: Ultra high performance key/value storage, 2018.

[24] InfluxDB. InfluxDB - Open Source Time Series, Metrics, and Analytics Database (http://influxdb.com/), 2018.

[25] S. K. Jensen, T. B. Pedersen, and C. Thomsen. Time series management systems: A survey. *TKDE*, 29(11), 2017.

[26] E. Keogh and M. Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, editors, *KDD*, pages 239–241, New York City, NY, 1998. ACM Press.

[27] Y. B. Kim. *Physiological time series retrieval and prediction with locality-sensitive hashing*. PhD thesis, Massachusetts Institute of Technology, Cambridge, USA, 2017.

[28] H. Kondylakis, N. Dayan, K. Zoumpatianos, and T. Palpanas. Coconut: A scalable bottom-up approach for building data series indexes. *PVLDB*, 11(6):677–690, 2018.

[29] H. Kondylakis, N. Dayan, K. Zoumpatianos, and T. Palpanas. Coconut: sortable summarizations for scalable indexes over static and streaming data series. *VLDBJ*, 28(6), 2019.

[30] M. Linardi and T. Palpanas. Scalable, variable-length similarity search in data series: The ULISSE approach. *PVLDB*, 11(13):2236–2248, 2018.

[31] M. Linardi and T. Palpanas. ULISSE: ULtra compact Index for Variable-Length Similarity SEarch in Data Series. In *ICDE*, 2018.

[32] M. Linardi, Y. Zhu, T. Palpanas, and E. J. Keogh. Matrix Profile Goes MAD: Variable-Length Motif And Discord Discovery in Data Series. 2020.

[33] Y. Malkov, A. Ponomarenko, A. Logvinov, and V. Krylov. Approximate nearest neighbor algorithm based on navigable small world graphs. *Inf. Syst.*, 45:61–68, 2014.

[34] Y. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *CoRR*, abs/1603.09320, 2016.

[35] A. Mueen. Similarity search on time series data: Past, present, and future. In *CIKM*, 2016.

[36] A. Mueen and E. Keogh. Finding repeated structure in time series: Algorithms and applications. In *ICDM*, 2014.

[37] OpenTSDB. OpenTSDB - A Distributed, Scalable Monitoring System (http://opentsdb.net/), 2015.

[38] T. Palpanas. Data series management: The road to big sequence analytics. *SIGMOD Rec.*, 44(2):47–52, 2015.

[39] T. Palpanas. Evolution of a Data Series Index. *CCIS*, 1197, 2020.

[40] T. Palpanas and V. Beckmann. Report on the first and second interdisciplinary time series analysis workshop (itisa). *SIGREC*, 48(3), 2019.

[41] T. Pelkonen, S. Franklin, P. Cavallaro, Q. Huang, J. Meza, J. Teller, and K. Veeraraghavan. Gorilla: A fast, scalable, in-memory time series database. *PVLDB*, 8(12):1816–1827, 2015.

[42] B. Peng, P. Fatourou, and T. Palpanas. ParIS: The Next Destination for Fast Data Series Indexing and Query Answering. 2018.

[43] B. Peng, P. Fatourou, and T. Palpanas. Messi: In-memory data series indexing. In *ICDE*, 2020.

[44] B. Peng, P. Fatourou, and T. Palpanas. Paris+: Data series indexing on multi-core architectures. *TKDE*, 2020.

[45] Prometheus. Prometheus – Monitoring system & time series database, 2018.

[46] QuasarDB. QuasarDB: high-performance, distributed, time series database, 2018.

[47] D. Rafiei and A. O. Mendelzon. Similarity-based queries for time series data. In *SIGMOD*, 1997.

[48] T. Rakthanmanon, B. J. L. Campana, A. Mueen, G. Batista, M. B. Westover, Q. Zhu, J. Zakaria, and E. J. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *KDD*, 2012.

[49] T. Rakthanmanon, E. J. Keogh, S. Lonardi, and S. Evans. Time series epenthesis: Clustering time series streams requires ignoring some data. In *ICDM*, 2011.

[50] RiakTS. Riak TS – Basho Technologies, 2018.

[51] P. P. Rodrigues, J. Gama, and J. P. Pedroso. Odac: Hierarchical clustering of time series data streams. In J. Ghosh, D. Lambert, D. B. Skillicorn, and J. Srivastava, editors, *SDM*, pages 499–503. SIAM, 2006.

[52] Y. Sakurai, Y. Matsubara, and C. Faloutsos. Mining big time-series data on the web. In *WWW*, 2016.

[53] P. Schäfer and M. Högqvist. SFA: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In *EDBT*, 2012.

[54] J. Shieh and E. J. Keogh. *i*sax: indexing and mining terabyte sized time series. In *KDD*, 2008.

[55] Y. Sun, W. Wang, J. Qin, Y. Zhang, and X. Lin. SRS: solving c-approximate nearest neighbor queries in high dimensional euclidean space with a tiny index. *PVLDB*, 8(1):1–12, 2014.

[56] Timely. Timely – A secure time series database based on Accumulo and Grafana, 2018.

[57] Timescale. Timescale - an open source time series management system, 2018.

[58] C. Turkay, N. Pezzotti, C. Binnig, H. Strobelt, B. Hammer, D. A. Keim, J. Fekete, T. Palpanas, Y. Wang, and F. Rusu. Progressive data science: Potential and challenges. *CoRR*, abs/1812.08032, 2018.

[59] Y. Wang, P. Wang, J. Pei, W. Wang, and S. Huang. A data-adaptive and dynamic segmentation index for whole matching on time series. *PVLDB*, 6(10):793–804, 2013.

[60] Warp10. Warp 10 – The Most Advanced Time Series Platform., 2018.

[61] T. Warren Liao. Clustering of time series data—a survey. *Pattern Recognition*, 38(11):1857–1874, 2005.

[62] D. E. Yagoubi, R. Akbarinia, F. Masseglia, and T. Palpanas. DPiSAX: Massively Distributed Partitioned iSAX. In *ICDM*, pages 1135–1140, 2017.

[63] D.-E. Yagoubi, R. Akbarinia, F. Masseglia, and T. Palpanas. Massively distributed time series indexing and querying. *TKDE*, 32(1), 2020.

[64] K. Zoumpatianos, S. Idreos, and T. Palpanas. ADS: the adaptive data series index. *VLDB J.*, 25(6):843–866, 2016.

[65] K. Zoumpatianos, Y. Lou, I. Ileana, T. Palpanas, and J. Gehrke. Generating data series query workloads. *VLDB J.*, 27(6), 2018.

[66] K. Zoumpatianos, Y. Lou, T. Palpanas, and J. Gehrke. Query workloads for data series indexes. In *KDD*, 2015.