Clean4TSDB: A Data Cleaning Tool for Time Series Databases

Xiaoou Ding
Harbin Institute of Technology
dingxiaoou@hit.edu.cn

Yichen Song
Harbin Institute of Technology
22S003013@stu.hit.edu.cn

Hongzhi Wang*
Harbin Institute of Technology
wangzh@hit.edu.cn

Donghua Yang
Harbin Institute of Technology
yang.dh@hit.edu.cn

Chen Wang
Tsinghua University
wang_chen@tsinghua.edu.cn

Jianmin Wang
Tsinghua University
jimwang@tsinghua.edu.cn

ABSTRACT

Billions of data points are generated by devices equipped with thousands of sensors, leading to significant data quality issues in time series data. These errors not only complicate time series data management but also compromise the accuracy and reliability of analysis based on such data. Given the noteworthy characteristics of time series data, existing cleaning methods struggle to provide adequate repairs, and tools supporting expressive constraints for time series remain scarce. To address this, we develop Clean4TSDB, a specialized data cleaning system for time series databases. This system integrates three key modules: expressive data quality constraint discovery, violation detection, and multivariate time series repairing, forming a comprehensive "profiling-detection-repair" workflow. Technically, we introduce TSDD, a data quality constraint that effectively captures contextual relationships within multivariate time series, and implement an efficient algorithm for its automated mining. Leveraging both row- and column-based constraints, we propose an effective time series cleaning algorithm. From a system standpoint, Clean4TSDB is pre-configured for seamless integration with time series databases like `Apache IoTDB`. Using user-provided and algorithmically-mined constraints, it effectively identifies various error patterns and offers reliable cleaning solutions. Furthermore, we establish a comprehensive library of state-of-the-art time series repair algorithms to meet the diverse needs of different management scenarios.

PVLDB Reference Format:
Xiaoou Ding, Yichen Song, Hongzhi Wang, Donghua Yang, Chen Wang, and Jianmin Wang. Clean4TSDB: A Data Cleaning Tool for Time Series Databases. PVLDB, 17(12): 4377 - 4380, 2024. doi:10.14778/3685800.3685879

1 INTRODUCTION

With the continuous progress of data acquisition techniques and significant enhancement of data storage capabilities, time series data, defined as sequences of continuously measured and recorded values over time [6, 12], is accumulating at an unprecedented rate. The deep mining and analysis of massive time series data have brought valuable insights and profound understandings to multiple fields. However, uncertainties inherent in data collection and maintenance often compromise the quality of time series data within information systems. Studies indicate that industrial time series data frequently contains over 20% erroneous records [10]. Issues such as equipment failures during sensor data transfer to databases can lead to data loss or inaccuracies. Furthermore, irregular data maintenance practices might result in a substantial amount of seemingly normal data that actually deviates severely from business rules. Utilizing such low-quality data in data analysis tasks like classification and regression without proper preprocessing can lead to misleading outcomes, potentially causing significant business disruptions.

Comprehensive data cleaning and preprocessing after data is written into the database and before data analysis are crucial to enhancing data quality. This step not only eliminates erroneous data but also improves data consistency and reliability, laying a stronger foundation for data analysis and decision-making. The issue of time series data quality has garnered attention from both academia and industry. However, compared to relational data, time series data possesses noteworthy characteristics, including temporal orders, strong correlation among attributes, and uncertainty in data acquisition process, as highlighted in our current research [2, 3]. These features pose significant challenges for cleaning time series data in real-world scenarios. The main difficulties include:

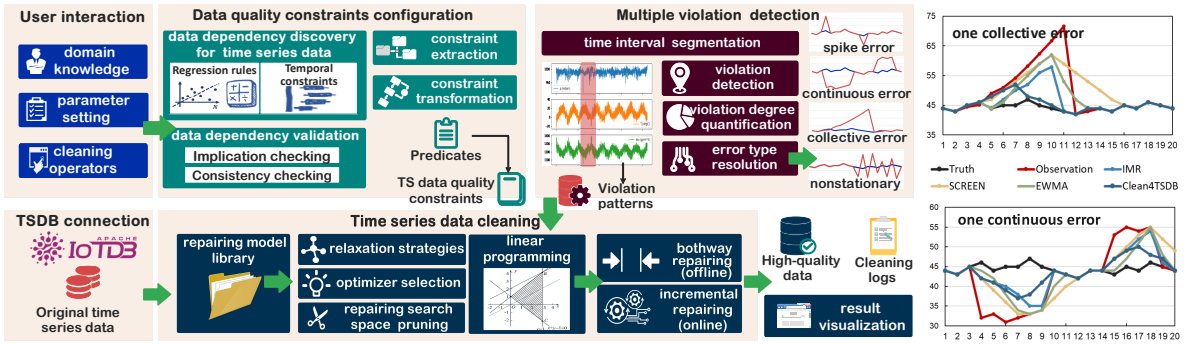
(1) **To represent semantically strong correlations in time series data and efficiently extract these constraints from data**. Time series data show both strong sequence and contextual correlations, requiring enhanced constraint languages. This enriched expressiveness increases computational cost of data quality (DQ) constraint discovery.

(2) **To repair multiple error patterns in multi-dimensional time series data instead of addressing each sequence individually**. Errors in time series tend to accumulate rather than occur discretely, highlighting the need to avoid undetected or falsely detected errors. Repairing these issues can be complex due to the extensive solution space, risking changes to accurate data absent a robust repair mechanism.

Current data cleaning techniques, including rule-driven (*e.g.,* Horizon, and Holoclean), data-driven (*e.g.,* Baran), and model-driven (*e.g.,* Boostclean) strategies, offer reliable solutions for relational data cleaning [7]. However, there's a lack of specialized cleaning tools for time series data. This deficiency mainly manifests in two ways: (1) inadequate support for complex data quality constraint semantics in time series, leading to a shortage of qualitative error detection for such constraints (despite numerous quantitative error detection techniques for time series data [8]), and (2) absence

*Corresponding author.

Proceedings of the VLDB Endowment, Vol. 17, No. 12 ISSN 2150-8097.
doi:10.14778/3685800.3685879



(a) Framework overview (b) Repairing results comparison

Figure 1: Clean4TSDB system overview

of efficient cleaning algorithms that simultaneously consider temporal context correlation and inter-attribute dependencies.

In our previous work, *Cleanits* [4, 5], a cleaning system for industrial time series data, we implemented simple algorithms for filling missing values and matching inconsistent attributes. Continuing our exploration in time series data cleaning, we develop a specialized cleaning system for time series databases, Clean4TSDB in this paper, which incorporates our advanced cleaning techniques [2, 3]. Clean4TSDB offers the following notable features:

(1) **Holistic cleaning workflow**. Clean4TSDB supports a comprehensive "profiling-detection-repair" workflow, with three key steps for time series data cleaning.

(2) **Effective discovery of expressive data quality constraints for time series**. Clean4TSDB provides data quality constraint discovery algorithms specifically for time series, which achieves high-level representation about temporal orders, arithmetic operations, and approximate equality in data quality validation.

(3) **From single-sequence to multivariate-sequence repairing**. Clean4TSDB provides an effective cleaning approach that integrate expressive constraints derived from both attribute dimensions and temporal orders. Besides, it supports a comprehensive library of state-of-the-art time series repair algorithms to meet the diverse needs of different management scenarios.

Clean4TSDB is pre-configured to interface with time series databases, such as `Apache IoTDB` [12]. All the proposed algorithms have been tuned and verified in extensive experiments, which shows that Clean4TSDB provides high-level repairing results in real time series management applications.

2 SYSTEM OVERVIEW

Figure 1(a) outlines the architecture of Clean4TSDB, including four main modules, *i.e.*, data quality constraints configuration, violation detection, error repairing, and TSDB connection.

Data Quality Constraints Configuration. Clean4TSDB facilitates two operations for constraint establishment: business-oriented constraints extraction and instance-driven data dependency discovery. For the former, we extract (implicit) business insights from domain experts to formalize data quality constraints. For the latter, we enrich data dependencies for time series data, thereby deriving expressive DQ constraints. These encompass: *i*) temporal relationship constraints between data values, *ii*) constraints involving arithmetic operations beyond basic relational operators (=, >, <), and *iii*) an approximation of the equality relationship, ensuring both the discovery algorithm's efficiency and the discovered dependencies' practical applicability. To this end, we have devised TSDDiscover algorithm to uncover the intrinsic dependencies within multivariate time series data [3].

Violation Detection. We implement three main functions in violation detection phases: violation identification, error type resolution, and violation degree quantification. Clean4TSDB segments data by time intervals, and screens out "suspected" errors during each interval. Our system profiles error patterns within *single* sequences, concentrating on four prevalent error types in time series: spike, continuous, collective, and nonstationary errors. The detected errors are subsequently mapped for violation classification. We have defined four violation forms specific to time series data, derived from the cartesian product of violation pattern *length* and *breadth*. Here, *length* refers to patterns on attributes (TSDB table columns), while *breadth* gauges patterns on instances (rows). This profiling approach minimizes false modifications to normal data, ensuring more precise repair outcomes.

Furthermore, we extend beyond qualitative error assessment, introducing a quantification of the degree to which data breaches a given constraint. This enhancement allows Clean4TSDB to respond more effectively to error repair needs.

Data Repairing. Considering the generality of the repairing phase, we implement a library of repair models to fulfill unique needs arising from various applications. Notably, we have developed novel approaches for time series repairing with the combination of expressive constraints from both attributes and temporal orders [2]. Clean4TSDB efficiently narrows down the potential repair space by incorporating time-related constraints, such as speed constraints [9], and refines the "minimum repair principle" for time series by minimizing the degree of the violations.

In addition, Clean4TSDB is designed to allow for the relaxation of data quality constraints, enabling users to swiftly identify practical solutions while sidestepping over-fitting issues. It meticulously documents cleaning logs for each action and ultimately delivers high-quality data back to the database.

TSDB Connection & User Interaction. Clean4TSDB is user-friendly to provide a variety of cleaning modes for users including overall mode, high-efficiency mode, auto-clean mode, etc. It is pre-configured to interface with time series databases, particularly to adapt the data storage and data query in TSDB. We achieve to connect to `Apache IoTDB` [12], a IoT-oriented time series database

built upon columnar time series file format, `TsFile` [12]. We highlight that Clean4TSDB supports Temporal SQL (`TSQL`) interface by auto-translating user's UI operations. For example, when users wish to view both original and cleaned data for a specified time range, Clean4TSDB autonomously generates the `TSQL` command:

```
select * from root.exampledb where t1 < time < t2
```

It is a notable feature of Clean4TSDB to support for such queries as *Range Query* with time predicates (t_1, t_2), and *Alignment Query* from multivariate time series. This feature simplifies the comparison of data states, thereby improving user interaction with the database and enabling efficient data processing.

3 IMPLEMENTATION DETAILS

3.1 Data quality constraints discovery

Preliminaries. $\mathcal{S} = \{S_1, ..., S_M\} \in \mathbb{R}^{N \times M}$ denotes M-dimensional time series data, where M is the total number of attributes. $S = \langle s_1, ..., s_N \rangle$ represents one sequence in $\mathcal{S}$, and $s_n = \langle x_n, t_n \rangle, (n \in [1, N])$, where x_n is a real-valued number with timestamp t_n. $\Phi = \{\phi_1, ..., \phi_n\}$ is the set of constraints defined on data $\mathcal{S}$, where $\phi \in \Phi$ is a formulated or learnt constraint the data are expected to satisfy.

Considering the characteristics of time series, we expand the expressiveness of data dependencies [3] for time series data from three aspects: *i*) to support predicate verification in tuples within time context, *ii*) to support linear functions to describe complex dependencies between attributes, and *iii*) to support relaxation in the satisfaction of constraints rather than extremely checking "equal" relationship in values. Such DQ constraints syntax is much more appropriate for numerical data, especially for IoT scenarios.

Specifically, given time series data schema $\mathcal{T}$ and a time window w, $C(r_i, w)$ denotes the w-length context consisting tuples from r_{i-w+1} to r_i. One DQ constraint defined over $\mathcal{T}$ is ϕ: $\forall r_i \in C(r_i, w), \neg(p_1 \wedge p_2 \wedge \cdots \wedge p_x)$. $p_{i \in [1,x]}$ is a predicate applied for values or patterns between attributes. Clean4TSDB supports linear function $f(X, Y)$: $\mathbb{R}^{(X)} \rightarrow \mathbb{R}$ among attributes. We have proved that such inference system for the DQ constraints extended in time series data is correct and sound in our recent work [3].

The DQ constraint discovery problem for $\mathcal{T}$ is to find a *valid* and *minimal* set Φ that holds on $\mathcal{T}$'s instance $\mathcal{S}$, where any $\phi \in \Phi$ holds on $\mathcal{S}$. Faced with huge predicate spaces, we involve the principle of business-driven and conciseness into the discovery algorithm, which assists to prune the space of expression structure and constructed predicates. Specifically, we adopt a supervised learning strategy based on *symbolic regression* to realize the search of function expressions, and introduce cutting operations for the length of ϕ, *i.e.,* the number of predicates in ϕ. Further, an efficient search algorithm is applied to mine the evidence set in the predicate space. Note that we carefully evaluate the implication and consistency issues for the discovered constraint set Φ with the proposed inference system, thus, we provide Φ with high-level interestingness and conciseness for users. We note that Clean4TSDB is designed to support the relaxation on the relation "identical" in predicates according to relaxed function dependencies techniques. Such relaxation is more suitable for time series data quality management [6].

3.2 Multi-dimensional time series data cleaning

Problem description. We transform DQ constraints into the boundary conditions of linear programming problems (LP), and formalize the multivariate time series data repairing problem as to find the optimal solution of LP, *i.e.,* $\min \triangle(\mathcal{S}', \mathcal{S}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \left| S'_{i[t_j]} - S_{i[t_j]} \right|$, $s.t.\ \forall \phi \in \Phi, \phi.F(S_i) \in [x_{i.\min}, x_{i.\max}]$, where S' represents the repaired m-dimensional time series, n is the length of S', and Φ is the constraint set. $\phi.F(S_i)$ denotes the feasible value range of S_i *w.r.t* ϕ.

Violation detection and error data profiling. Aiming to uncover the hidden errors from data, we adopt four types of DQ constraints, which classifies constraints for time series by dependence on attributes (columns in database) and instances (rows) [1]. Thus, we identify error patterns with both row- and column- violation features [2]. Further, we quantify the violation degree of error instances to achieve a more accurate analysis of the changing trends of errors. For a given constraint σ and the set of cells involved in σ, *i.e., C*, the degree of violation is represented as $VDdeg(C, \phi) = \min\left\{|f(C, \phi) - f_{min}|, |f(C, \phi) - f_{max}|\right\}$, where $f(C, \phi)$ calculates the value for cell C according to ϕ. Specifically, we represent the violated constraints and the involved cells in the form of a constraint hypergraph. We profile violation *w.r.t* constraint ϕ with a feature set $Vio(\phi) = \big(S[T], Vtype, Vdeg\big)$. $S[T]$ denotes the involved attributes in time interval T, and $Vtype$ represents the violation type.

Key cell determination and data repairing. We sort the constraints in Φ by their degree of violation *i.e., VDdeg*. Further, we quickly identify the data where the actual errors occurred, which we call *key cells*. We introduce the minimum vertex cover (MVC) approximation algorithm to solve the problem, and designed a heuristic algorithm by calculating the degree of constraint violation and the number of vertices corresponding to hyperedges on the graph to find the key cells more accurately. After that, we repair the identified key cells. Note that we construct a LP problem to solve for the repair values of the vertices and the sorted constraint. Specifically, we solve this unconstrained optimization problem with a well-design objective function. This effectively reduces the size of the repairing problem. The process is repeated for each row in $\mathcal{S}$ until all rows have been processed, resulting in the repaired $\mathcal{S}'$.

Theoretical guarantee. By employing expressive constraints based on inter-attribute dependencies and temporal constraints during the cleaning process, we theoretically demonstrate that the candidate repair space is reduced more significantly compared to using either type of constraint alone. Moreover, our repair strategy consistently repair data to a valid range satisfying inter-attribute dependencies. Detailed proofs are reported in [2].

4 EXPERIMENTS & DEMONSTRATION

4.1 Repair method library

We will demonstrate our system on real IoT datasets [2], and compare the proposed repairing algorithm in Clean4TSDB to the benchmark cleaning approaches for time series data, including (1) a SOTA constraint-based time series cleaning method Screen [11], and its extended version with acceleration constraints, *i.e.,* Speed+Acc [9]. (2) HoloClean, a SOTA cleaning tools for general big data. (3)IMR [13], a SOTA time series cleaning method with labels. (3) smoother and filter-based EWMA ,Median Filter, Kalman Filter. Applied **metrics** include *L1-error*, *RRA*, *F1-score*, and time cost, etc.

Figure 1(b) shows the repairing effectiveness comparison for different methods, which verifies that Clean4TSDB provides better

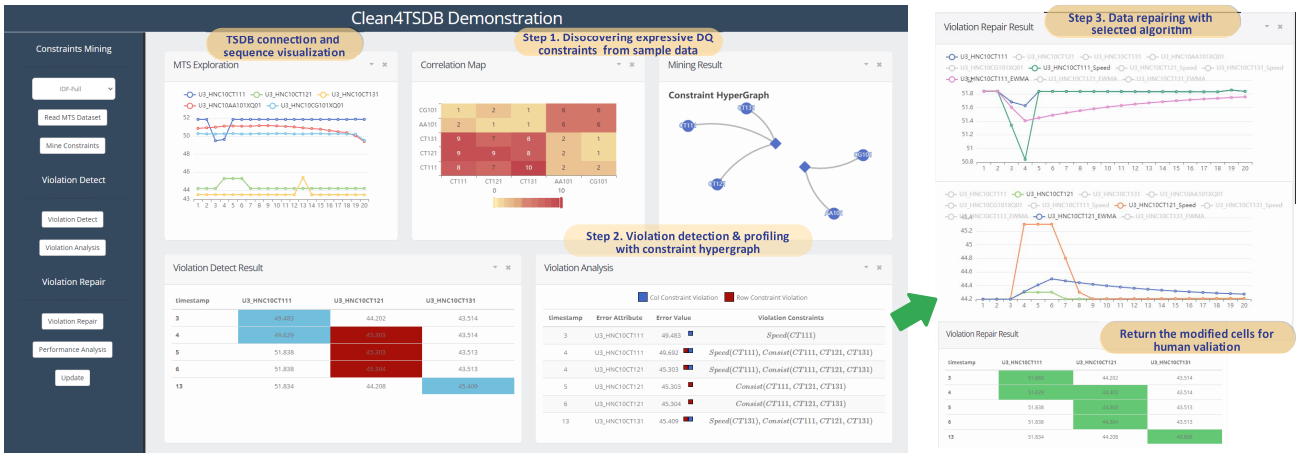


Figure 2: Pages demonstration in Clean4TSDB

Table 1: Overall repairing performance comparison

	IDF				*SWaT*			
	L1error ↓	*RRA* ↑	*F1* ↑	Time(*s*)	*L1error*	*RRA*	*F1*	Time(*s*)
Clean4TSDB	**0.1249**	**0.975**	**0.97**	112.1	**0.4048**	**0.927**	**0.99**	183.3
Screen	0.9671	0.735	0.67	164.8	2.7043	0.180	0.41	274.6
Speed+Acc	0.8963	0.767	0.68	339.9	2.6690	0.199	0.44	609.1
EWMA	1.6130	0.308	0.22	0.01	2.4749	0.301	0.22	0.02
Median	1.9710	0.003	0.22	0.01	2.9935	0.059	0.21	0.02
Kalman	1.8160	0.147	0.22	147.1	2.7699	0.147	0.22	243.9
IMR	0.9579	0.668	0.56	171.2	1.2535	0.736	0.61	221.8
HoloClean	1.6900	0.429	0.83	310.3	2.1408	0.518	0.78	560

repair solutions which are closer to the truth values, especially for collective and continuous errors in data. We report part of comparison results in Table 1. It confirms that Clean4TSDB always better identifies and repairs violations hidden in multi-dimensional time series, and it shows more advantages for dealing with complex error patterns with strong correlation. It highlights again the necessity of extended DQ constraint discovery for time series.

4.2 Demonstration

We intend to demonstrate all functions in Clean4TSDB with connection to `Apache IoTDB` with case studies from real industry IoT data quality requirement. Users first interface to the `config` file in the `.streamlit` folder, and Clean4TSDB automatically reads data from TSDB and convert it into `dataframe` format in memory.

Once data are uploaded, users can configure parameters (*e.g.*, sliding window length) and upload personalized DQ constraints. As shown in Fig. 2, it discovers data dependencies from clean sample data, and it is user-friendly to provide visualization of dependency patterns. Note that Clean4TSDB allows to set upper and lower bounds of predicate in constraints to support relaxation dependencies. Users could begin cleaning by clicking the button, and detection results for all error types will first be displayed on the page. Clean4TSDB provides data profiling results for error instances including violation degree and some necessary feature values.

In the repairing phase, users can choose algorithms from the repair model library, or apply the recommended method via the parameter-free repairing mode. Repair results are shown on the right side of Fig. 2. We emphasize that users are able to view the repairing results for multi-sequences simultaneously along the timeline. Also, visual result comparisons for all repairing models in the library are available in Clean4TSDB. The observation of results from various strategies contributes to a better understand of the data quality problems in TSDB. Clean4TSDB records both the detection and repairing results in logs and returns them to TSDB together with the cleaned version of data. Accordingly, users could in turn make targeted DQ constraints and develop more reliable solutions for TSDB data quality management with Clean4TSDB.

ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China (2021YFB3300502); National Natural Science Foundation of China (NSFC) (62202126, 62232005, 92267203); China Postdoctoral Science Foundation (2022M720957); Heilongjiang Postdoctoral Financial Assistance (LBH-Z21137).

REFERENCES

[1] Tamraparni Dasu, Rong Duan, and Divesh Srivastava. 2016. Data Quality for Temporal Streams. *IEEE Data Eng. Bull.* 39, 2 (2016), 78–92.
[2] Xiaoou Ding, Genglong Li, Hongzhi Wang, Chen Wang, and Yichen Song. 2024. Time Series Data Cleaning Under Expressive Constraints on Both Rows and Columns. In *ICDE*. IEEE, 3682–3695.
[3] Xiaoou Ding, Yingze Li, Hongzhi Wang, Chen Wang, Yida Liu, and Jianmin Wang. 2024. TSDDISCOVER: Discovering Data Dependency for Time Series Data. In *ICDE*. 3668–3681.
[4] Xiaoou Ding, Yichen Song, Hongzhi Wang, Donghua Yang, and Yida Liu. 2023. Cleanits-MEDetect: Multiple Errors Detection for Time Series Data in Cleanits. In *DASFAA*, Vol. 13946. Springer, 674–678.
[5] Xiaoou Ding, Hongzhi Wang, Jiaxuan Su, Zijue Li, Jianzhong Li, and Hong Gao. 2019. Cleanits: A Data Cleaning System for Industrial Time Series. *PVLDB* 12, 12 (2019), 1786–1789.
[6] Aimad Karkouch, Hajar Mousannif, Hassan Al Moatassime, and Thomas Noël. 2016. Data quality in internet of things: A state-of-the-art survey. *J. Netw. Comput. Appl.* 73 (2016), 57–81.
[7] Wei Ni, Xiaoye Miao, Xiangyu Zhao, Yangyang Wu, and Jianwei Yin. 2023. Automatic Data Repair: Are We Ready to Deploy? *CoRR* abs/2310.00711 (2023).
[8] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. 2022. Anomaly Detection in Time Series: A Comprehensive Evaluation. *Proc. VLDB Endow.* 15, 9 (2022), 1779–1797.
[9] Shaoxu Song, Fei Gao, Aoqian Zhang, Jianmin Wang, and Philip S. Yu. 2021. Stream Data Cleaning under Speed and Acceleration Constraints. *ACM Trans. Database Syst.* 46, 3 (2021), 10:1–10:44.
[10] Shaoxu Song and Aoqian Zhang. 2020. IoT Data Quality. In *CIKM*. ACM, 3517–3518.
[11] Shaoxu Song, Aoqian Zhang, Jianmin Wang, and Philip S. Yu. 2015. SCREEN: Stream Data Cleaning under Speed Constraints. In *SIGMOD*. 827–841.
[12] Chen Wang, Jialin Qiao, Xiaodong Huang, Shaoxu Song, Haonan Hou, Tian Jiang, Lei Rui, Jianmin Wang, and Jiaguang Sun. 2023. Apache IoTDB: A Time Series Database for IoT Applications. In *SIGMOD*. ACM.
[13] Aoqian Zhang, Shaoxu Song, Jianmin Wang, and Philip S. Yu. 2017. Time Series Data Cleaning: From Anomaly Detection to Anomaly Repairing. *PVLDB* 10, 10 (2017), 1046–1057.