© 2020 by the authors; licensee RonPub, Lübeck, Germany. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/4.0/).

Research online Publishing www.ronpub.com

Open Access

Open Journal of Internet of Things (OJIOT)
Volume 6, Issue 1, 2020
http://www.ronpub.com/ojiot
ISSN 2364-7108

The Internet of Things as a Privacy-Aware Database Machine

Andreas Heuer , Hannes Grunert

Database Research Group, University of Rostock, Konrad-Zuse-Haus (KZH), Albert-Einstein-Straße 22, 18059 Rostock, Germany, {andreas.heuer,hannes.grunert}@uni-rostock.de

Abstract

Instead of using a computer cluster with homogeneous nodes and very fast high bandwidth connections, we want to present the vision to use the Internet of Things (IoT) as a database machine. This is among others a key factor for smart (assistive) systems in apartments (AAL, ambient assisted living), offices (AAW, ambient assisted working), Smart Cities as well as factories (IIoT, Industry 4.0). It is important to massively distribute the calculation of analysis results on sensor nodes and other low-resource appliances in the environment, not only for reasons of performance, but also for reasons of privacy and protection of corporate knowledge. Thus, functions crucial for assistive systems, such as situation, activity, and intention recognition, are to be automatically transformed not only in database queries, but also in local nodes of lower performance. From a database-specific perspective, analysis operations on large quantities of distributed sensor data, currently based on classical big-data techniques and executed on large, homogeneously equipped parallel computers have to be automatically transformed to billions of processors with energy and capacity restrictions. In this visionary paper, we will focus on the database-specific perspective and the fundamental research questions in the underlying database theory.

Type of Paper and Keywords

Visionary paper: *IoT, Assistive Systems, Activity Recognition, Big Data Analytics, Privacy, Decentralized Analytics, Query Containment, Answering Queries using Views, Capability-Based Query Rewriting, The CHASE*

1 Introduction

In this section, we will shortly present our aims, our motivation to do this research, and give an overview of the article.

1.1 What We Are Aiming at

The main idea of our project is to use the Internet of Things (IoT) as a database machine instead of using a computer cluster with homogeneous nodes and very fast high bandwidth connections. It is important to massively distribute the calculation of complex database queries on available sensor nodes and other low-resource appliances in the environment, not only for reasons of performance, but also for reasons of privacy and protection of corporate knowledge.

This paper is accepted at the *International Workshop on Very Large Internet of Things (VLIoT 2020)* in conjunction with the VLDB 2020 conference in Tokyo, Japan. The proceedings of VLIoT@VLDB 2020 are published in the Open Journal of Internet of Things (OJIOT) as special issue.

1.2 Original Motivation

Already a few months before the name Edward Snowden became known worldwide, we had the idea for the PArADISE project, which should support a privacy-aware realization of situation, activity and intention recognition (SAIR) in assistance systems. Several

research projects of the DFG coordinated research group MuSAMA and the larger coordinated research cluster *Mobile Assistance Systems* of the state of Mecklenburg-Western Pomerania have been criticized in the press, radio and television because a Big Brother was suspected behind these assistance systems. MuSAMA was to determine the basics of spontaneously coupled appliance ensembles, where the activities and intentions of the users were to be determined from the sensors of the appliances in order to support them with the assistive environment. Even before Edward Snowden and long before the EU-GDPR came into force, we wanted to allow the analysis of certain information for specific purposes, but automatically exclude side effects that could endanger privacy. For this purpose, the diverse sensor data of the users were to be evaluated as close as possible to the sensors, for example directly on the smart appliances.

The horizontally distributed architecture for the development phase of the assistance system, including the training phase of the underlying Machine Learning procedures (PaMeLA), and the vertically distributed architecture for processing the data close to the sensors during operation of the assistance system (PArADISE) will be presented in more detail in Section 3.

1.3 Current Motivation: Corona Proximity Tracing App

The topic of privacy-aware assistance technologies has gained importance in 2020 due to a far-reaching pandemic. How to efficiently determine chains of infection through an App that a very large percentage of the population should use, but without endangering the privacy of the App's users? For example, it should be prevented that data is collected centrally, which would enable other analysis results about the users, such as exact movement profiles, in addition to chains of infection. The best way of preventing such analysis options is, for example, to process the data in a completely decentralized manner without any central storage. The analysis engine for the chains of infection is simply the ensemble of the users' smartphones. In the spring of 2020, there was a dispute in Germany whether the PEPP-PT [33] (Pan-European Privacy-Preserving Proximity Tracing) project should after all store and evaluate the collected data centrally. Many of the researchers originally involved in PEPP-PT denied this and separated the DP-3T [40] (Decentralized Privacy-Preserving Proximity Tracing) project from the PEPP-PT development. In the meantime (at the time of writing this article), the German Ministry of Health has promised that the decentralized version will be ready for use by end of June 2020.

The approach chosen for contact tracing of infection chains is suitable for the completely decentralized processing of user-related data. The user's smartphones only send the results of the analyses, in the case of DP-3T only anonymously to other smartphones affected by infection chains, without revealing the origin of the report.

1.4 What We Are Aiming at, Revisited: Specific Research Questions

The general research question is now: Can we automatically transform algorithms that are supposed to work on centrally collected data gathered from decentralized smartphones (or general IoT devices) in such a way that the capabilities of parallel, decentralized IoT devices are fully utilized? Is it possible to automatically transform the centralized approach into a decentralized one? For what types of analyses is this possible? In other words, for which analyses can a large collection of IoT devices be used as a massively distributed database machine? In our research we are still at the beginning of the way to answer these questions. In this article, we present both the State of the Art in relevant sub-areas of database research, our own previously completed partial works and their results, and especially the vision for a far-reaching solution of the above questions.

1.5 Structure of the Article

In the remainder of the article, we will give a short overview of the State of the Art in relevant database research fields in Section 2. The research results that we have achieved so far are summarized in Section 3. Section 4, on the other hand, introduces open research problems that we want to tackle in the future — and we hope that other database research groups will follow. One of the aspects of the open research problems is a necessary technique to prove equivalence of (1) centralized complex data analysis algorithms such as complex database queries, and (2) a large number of small queries on IoT devices that are also the sources of the data to be analyzed. For the purpose of this research problem, we introduce some techniques of database theory such as query containment, the Answering Queries using Views (AQuV) problem, and the CHASE algorithm as a universal tool in theoretical database research in Section 5. We conclude by presenting *The Hammer and the Dance (with the Tiger)* principle and applying it to the 2020 pandemic as well as to the necessary fundamental database research (Section 6).

2 State of the Art

We review the state of the art in corresponding areas like cloud computing and big data, fog computing, and embedded database management systems.

Cloud, Big Data, and Fog Computing: In the era of Big Data, more and more information is stored and processed in Cloud environments. Such systems offer a variety of services and possibilities for data storage, including services for the Internet of Things (e.g. APIs for REST and MQTT) and for assistive systems (such as smart living environments). Unfortunately, privacy is often ignored or, at least, it is not guaranteed by cloud services.

Instead of using hundreds of thousands of computers in a data center (e.g. Google) in the cloud, we can also use hundreds of billions of sensors or devices in the IoT to perform the necessary computations for the activity and intention recognition of assistive systems. This results in fog or edge computing [39, 6] and even in local data processing on sensors. What is missing in fog computing, is a database-centered approach to computation, i.e., given a query representing a necessary computation on IoT sensor data, the question is, how to automatically prevent the system from simply transferring the complete data sets to cloud servers.

Embedded DB Systems: Besides standard database systems, there exist several specialized databases like Berkeley DB [38] and TinyDB [23]. These systems are designed to run on resource limited devices like Raspberry Pis or even as embedded databases directly on the sensor. In [7], several approaches to a distributed database management on sensor networks are compared, TinyDB among them, here especially aiming at energy efficiency. Acquisitional query processing [7, 23] can push queries to sensors and select relevant sensors in a WSN to reduce the amount of sensors needed for a computation (sensor reduction). In the existing approaches mentioned in [7], this sensor reduction is completely done manually (by the programmer). Our aim is to automate this process by dimension reduction operations, that are generated by the query transformation process.

Nori [32] gives an overview on small embedded database systems and lists challenges like mobility, streaming, disconnected state, and embedding. TinyDB [23], Berkeley DB [38], and eXtremeDB [29] are systems which have a small footprint ranging from 150 kByte to 200 kBytes and are configurable, allowing to reduce size further. TinyDB relies on a special OS (TinyOS) and provides a variant of SQL as extremeDB does. Both allow for in-memory databases and use a tuple store. Only TinyDB is suitable for the requirements of (highly) resource-constrained devices, however not all necessary database query operations are included.

Stream Data Processing: Chaudhry et al. give a survey on data stream algorithms, languages and systems [5]. For data stream operators and their semantics in general refer to [24]. Joining data from different sources if operating on sliding windows or streaming sensor data needs special algorithms.

To fight against weak bandwidth and strong latency, it becomes convenient to render the query evaluation into an autonomous and self-adaptable execution on large-scale networks. [31] proposes a mobile relational algebra to decentralize the control of the dynamic query optimization processes to increase this autonomy. Unfortunately, the approach operates just on standard algebra.

Some basic query capabilities we have to consider: To be able to perform complex data analyses, we have to extend the standard relational algebra operators SPJ (Selection, Projection, Join) mostly used in approaches from database theory. To express Machine Learning (ML) algorithms have to add recursion and the aggregates minimum, maximum, sum, count and average in conjunction with linear arithmetic constraints. What is also missing, is an extension of these techniques to more general aggregate and grouping operators as well as window-based (data stream) operations and a new type of join connecting tuples with similar (but not equal) timestamps. In Section 4, we will introduce the SAIR (Situation, Activity, and Intention Recognition) and the ML-QL (Machine Learning Query Language) base operations.

3 Our Contributions

In this section, we will summarize those of our own research results of the last two decades relevant to the problem of transforming a centralized database query to an ensemble of IoT devices with restricted capabilities automatically. What have we done so far?

3.1 The General Framework

As two of our current research projects, we are developing the PaMeLA[1] and PArADISE[2] frameworks. These frameworks aim at supporting developers of assistive systems in three development phases. In Figure 1, these phases are shown as Development (left-hand side), Data and Dimension Reduction (depicted by the arrow in the middle), and Usage (right-hand side):

- **Development:** ML Developers and Data Scientist are trying to detect and predict user activities, using

[1] Parallelization of Machine Learning Algorithms
[2] Privacy-aware assistive distributed information system environment

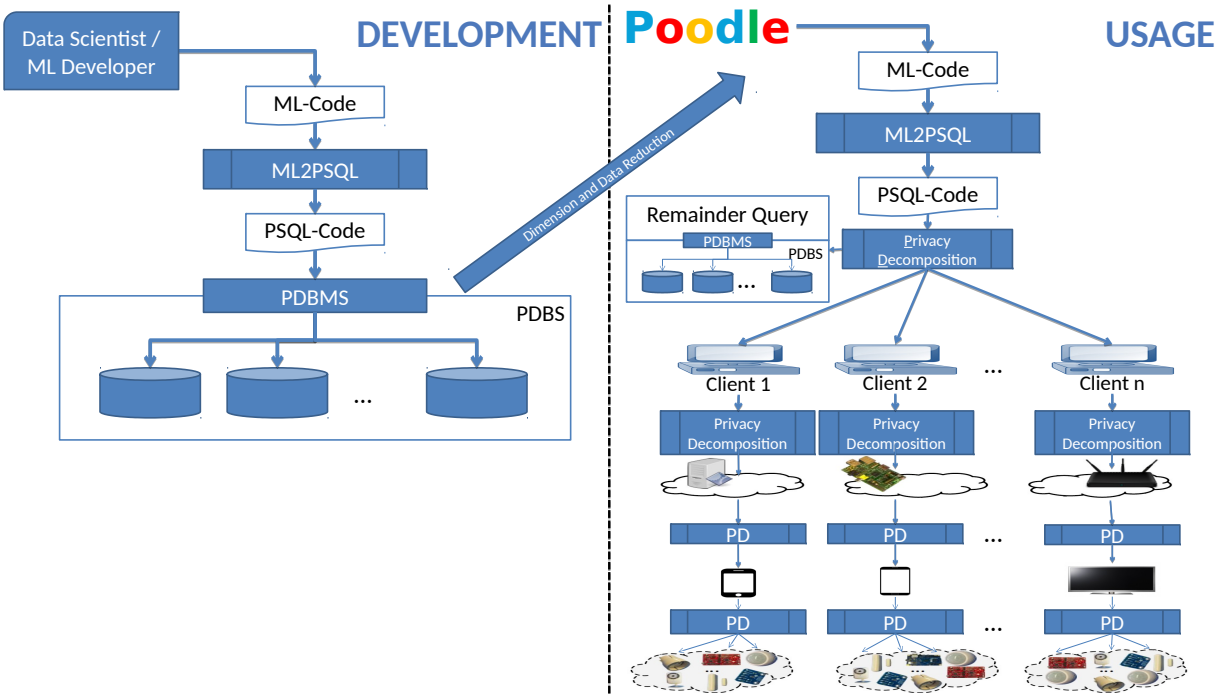


Figure 1: The general Framework: PaMeLA (to the left) and PArADISE (to the right) [27, 28]

data from a small amount of test persons, collecting sensor data for a short time period (maybe some weeks), annotating these sensor data with activity information, and then trying to learn the activity models by means of ML algorithms. This is our PaMeLA framework.

- **Data and Dimension Reduction:** In the development phase, there is a small amount of probands, but a large amount of sensors and also high-frequency sensor data. After having derived the activity and intention models, one has to reduce the dimensions of the data (e.g., the number of sensors being evaluated) and the size of the data itself (e.g., measuring and transmitting sensor data every minute instead of every milisecond).

- **Usage:** When using the assistive system afterwards for a huge number of clients (millions of clients having billions of sensors) with the reduced set of sensor data, one has to decompose the SQL queries detecting the activities and intentions of the users. This query decomposition aims at better performance because the query will be vertically pushed down to the sources of the data (the sensors) as close as possible. Even more importantly, the decomposition of the query results in better privacy for the user of the assistive systems, since most of the original sensor data has not to leave his personal equipment, his apartment, or his car. Only a *remainder query*, the part of the query that cannot be pushed down to the clients and sensors, has to be evaluated on the large cluster computers of the provider of the assistive system. This is our PArADISE framework.

To be able to *automatically* decide about the privacy-oriented decomposition of the queries, we have to use SQL queries as a basis for query containment and Anwering-Queries-using-Views techniques. Hence, it is crucial for this approach to be able to express ML code by a sequence of SQL queries in the development phase of the system. Only then, one can use the privacy-by-design principle when constructing the evaluation algorithms in the usage phase.

3.2 PaMeLA

We call the provider of the globally distributed system *Poodle* [12]. Poodle uses ML development tools such as R, Python or higher-level languages to derive the activities and intentions of the user. This ML code will then be transformed to a sequence of SQL statements. These SQL statements will then be evaluated in parallel on a large computer cluster, the parallelization will be introduced by the PaMeLA system. This phase is called ML2PSQL in Figure 1.

Database Support for Assistive Systems: In the

DFG-funded Joint Research Group MuSAMA, we developed basic database support techniques for assistive systems. Here, we transformed Machine Learning (ML) algorithms into complex SQL queries, aiming at parallelizing relational row or column store database query operations [26]. During MuSAMA, several situation, activity and intention recognition algorithms were developed based on Machine Learning techniques on masses of sensor data combining smart (home) appliances [44, 36]. These use cases give us a large amount of sensor data, situation models and activity models for our future research. In [37], we applied stream data management operations for a simulation environment. The supported operations are simple selection and projection operations or application supplied methods.

In [27], we showed how a specific ML algorithm, the Hidden Markov Model (HMM), can be transformed into an optimized sequence of SQL statements. In the coordinated research projects we participated in, the Hidden Markov Model has proven suitable for different SAIR problems that had to be solved to build the assistance systems.

In [28], we further optimized the SQL sequences realizing the ML algorithm by parallelizing the queries on a computer cluster horizontally. The ML algorithm implementation in SQL heavily used matrix operations. It turned out that the HMM problems in our projects resulted in matrix-matrix or matrix-vector products of very sparse matrices. For operations on sparse matrices, parallel database management systems proved superior to other Big Data Frameworks such as Spark or others.

In the remainder of this article, the PaMeLA framework and the development phase of ML algorithms will not be considered further.

3.3 PArADISE

Privacy in Database Systems: In our previous work, we focused on privacy aware query rewriting on "traditional" database systems. We developed efficient algorithms to identify information in high dimensional data which compromise privacy [9], integrated privacy concepts for anonymization by slicing and k-anonymity into relational structures [11, 20] and developed concepts for data minimization through query rewriting [13]. Additionally, we formalized privacy constraints for assistive environments [10] which are included in the query transformation process.

An overview on our previous work concerning privacy in databases can be found in [12]. In our approach, performance aspects of Big Data Analytics and privacy protection are combined. Our privacy aware query processor generates anonymized result sets, which still maintain a high degree of quality for the assistive system while additional knowledge can hardly be derived. The preprocessor allows the analysis and the privacy aware rewriting of database queries and decides whether the query is answered directly on the current layer or if is sent to a lower layer.

To reduce privacy violations, it is necessary (1) to decrease the amount of collected personal information, i.e. to apply the principle of data avoidance (except where data are required), (2) to process data with personal references as less as possible or — at least — as close to the local data sources (sensors) as possible and (3) to anonymize, pseudonymize and delete personal data, unless it is used for further processing and necessary to realize the aim of the assistive environment.

Data minimization and data avoidance are indispensable requirements for the design of smart systems. This requirement can be achieved in databases by transforming both queries and query results, as well as using views, a technique we developed in the MOVI project [18]. The methods developed are restricted to standard SQL language.

Query Containment / Transformation: In [13, 12] we proposed a layered architecture to achieve privacy through query rewriting in the Internet of Things. The architecture consists of several layers (see Figures 2 and 3) with different power regarding expressiveness of queries, memory and energy consumption.

The cloud layer sends a query Q to the database instance d integrating the entire sensor data recorded in an IoT environment. As data sources, sensors are located in appliances in apartments and buildings which sum up to the mentioned (distributed) database. Instead of shipping d to the cloud server, maximal parts of Q will be evaluated as close to the sensor as possible. The whole query processing procedure is given by:

$$\begin{aligned} Q(D) &:= d_j \\ &:= Q_j(... d_i' = A(d_i = Q_i(... (d_1 = Q_1(d)) ...))) \end{aligned}$$

The query Q is fragmented in queries Q_j (that will be executed on the lower layer) and a remainder query Q_δ that can only be performed at the more powerful layer. While Q could be an iterative machine learning algorithm implemented in R and SQL, and Q_j a complex SQL query with recursion, the lowest layer in the processing chain can only compute some simple selections and aggregations over the last values generated through a window function. Each of the nodes will ship the query result d_j to the node sending the query. After a final anonymization step A, the data d_0 will only be a small subset of the original data d. A detailed description of this concept can be found in [13].

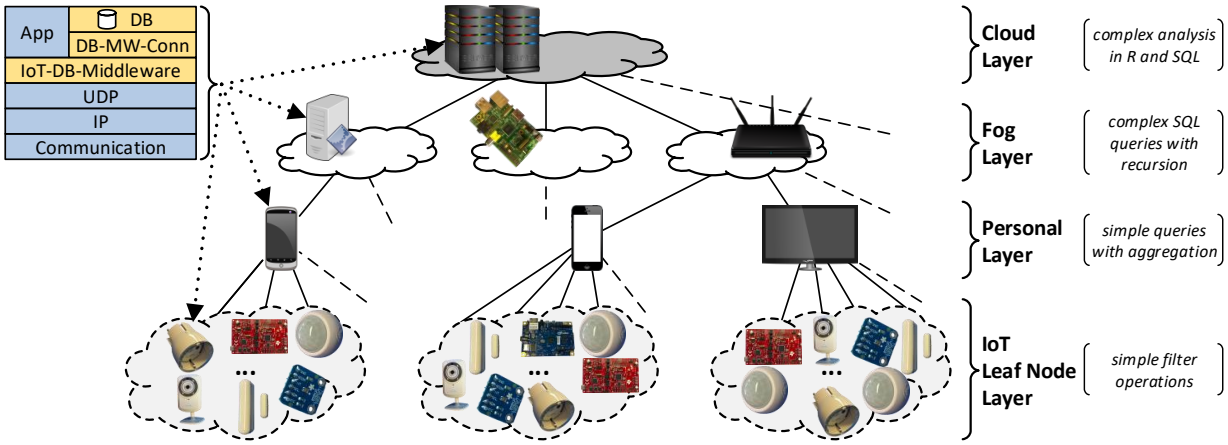


Figure 2: Layered System Approach (MW: Middleware, App: Application; in accordance with [16])

Up to now, the technique mentioned above, only works for simple SQL statements. Within our first systematic studies [14, 15] on applying Query Rewriting on more complex queries, we were able to integrate aggregate as well as arithmetic constraints in the rewriting process. In the future, we want to extend the techniques to SAIR-operations based on ML-QL operators as their basis (all of them being introduced in Section 4).

4 Our Vision: Decentralized Analytics on IoT Appliances

In this section, we will introduce open research problems that, on the one hand, we want to tackle in the future, but, on the other hand, that should be also interesting for other database research groups joining us on the way to deriving decentralized database queries on IoT devices automatically from existing centralized solutions. We will now broaden our PArADISE approach that was restricted to specific smart environments using trained HMM applications for activity and intention recognition.

4.1 The General Aim

The handling of data in IoT environments will be rethought fundamentally in the future, not only for performanced reasons like a reduced energy consumption [39], but also to respect the user's privacy. Currently, data is just pushed to the cloud where it will be combined and analyzed in different ways. A layered approach enables new methods to store, process, and query data on the lower layers. To achieve this, IoT and database middleware have to be collated [16].

We consider the IoT — processors and storage in billions of sensors and smart appliances — as a new hardware platform for database management systems. Our applications can be smart, assistive systems, as in PArADISE, but also other applications making use of data from IoT devices.

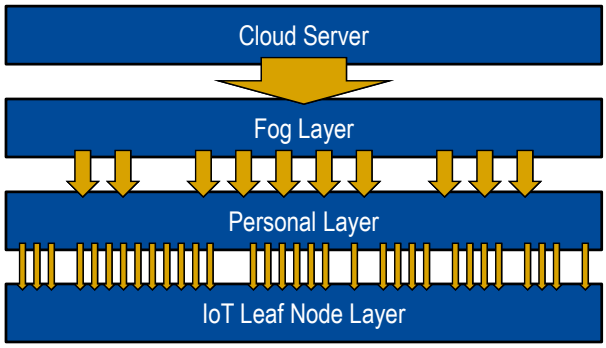
Instead of using a computer cluster, we want to examine how the Internet of Things (IoT) can be used as a database machine. We call our approch **BiSAM** (Billions of Smart Appliances as a Database Machine). Functions crucial for assistive systems, such as situation, activity and intention recognition (SAIR), are to be automatically transformed not only in database queries, but also in local nodes of lower performance.

4.2 The BiSAM Approach

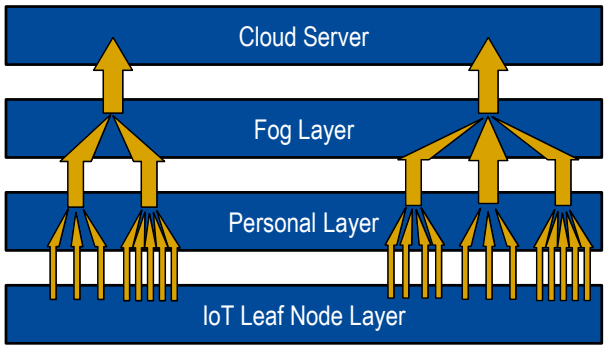
From a database-specific perspective, analysis operations on large quantities of distributed sensor data, currently based on classical big-data techniques and executed on large, homogeneously equipped parallel computers have to be automatically transformed to billions of processors with energy and capacity restrictions.

For this purpose, query-containment techniques on relevant analysis operators, originating from the database theory, are to be developed further, which also take into account capability constraints (Capability-Based Query Rewriting).

From an IoT-specific perspective, database operators are to be integrated into IoT nodes using a lightweight middleware (see Figure 2). These operators should be elementary but also suitable for assistive systems. In particular, scalability, flexibility, energy and quality requirements have to be considered.



(a) Query Push Down



(b) Result Pull Up

Figure 3: Query Push Down and Result Pull Up

Technical Aims: Privacy, Protection of Corporate Knowledge, Performance

Privacy, Protection of Corporate Knowledge: While SAIR queries supporting the main aims of assistive systems should be allowed to operate, other possible Big Data Analytics applications to derive user or motion profiles should be automatically rejected. We can do this by reducing the amount of detailed sensor data to be collected on cloud servers, as well as by checking privacy constraints against the queries performed, resulting in query containment techniques.

Performance: While collecting and analyzing all IoT sensor data of billions of smart appliances on centralized servers may be too expensive, we aim at pushing maximal parts of the analysis as close to the data sources (IoT leaf nodes such as sensors) as possible. We then can analyze masses of data extremely in parallel.

In order to achieve these goals, we will turn the architecture of assistive systems upside down: instead of transferring the data to the manufacturer via the cloud, we will evaluate the SAIR operations as early as possible at the sensor. To achieve this, we introduce a layered approach in which a local fog layer, a personal layer and finally an IoT leaf node layer are introduced below the Cloud Layer.

Our Approach: Automatic Pushdown of ML-QL Operations from Cloud to IoT Nodes

In smart, assistive systems, the situation, activity, and intention recognition (SAIR operations) can be realized by ML (Machine Learning) algorithms. We will formulate ML algorithms by an ML query language (ML-QL). The ML-QL base operations are simple selections (filters), projections (without duplicate elimination, so stream-based projections are possible), uncertain timestamp join (on most recent time windows, see also [30]), aggregations over most recent values, grouping over time dimension, as well as several matrix operations resulting in complex matrix equational systems. Selection and aggregation operations are important for data reduction, projection is important for dimension reduction. Data and dimension reduction are important for privacy and performance reasons.

We will investigate the concrete set of database querying operations that are necessary to do the SAIR computations. This will result in base operations of the ML-QL query language. Our focus is on the following sets of operations: from simple aggregates, like maxima and sums, we want to extend the approach to complex functions like correlation or regression. Further, we want to identify special cases for complex queries where maximal query transformations and query containment are undecidable in general but can be solved under certain conditions.

Figure 3 describes the process of rewriting a complex query to match the capabilities of the lower layers of our architecture. The query push down mechanism calculates queries resulting in a minimal superset of the final result. We will see in Section 5, that classical Answering-Queries-using-Views techniques calculate a maximal subset of the original result instead.

We will then push down these operators to fog, personal and IoT leaf node layer, applying the ML-QL selection, projection, join and aggregation operations as close to the data (sensors; appliances; fog) as possible. The ML-QL base operations have to consider (1) time constraints, (2) space constraints (volatile vs non-volatile memory on IoT leaf nodes or appliances), (3) quality constraints (when joining information over similar but not equal timestamps), (4) energy constraints, and (5) the privacy constraints, the user has defined for his IoT or personal layer.

Complex algorithms like Hidden Markov Models and auto-correlations are based on advanced aggregates, e.g., linear regression and standard deviation. These aggregates can be splitted into basic aggregates like sums, averages, and so on. In our approach, these transformations are done automatically, so that the simple aggregates are executed on resource limited

devices while a remainder query Q_δ will combine the pre-aggregated data in order to compute the final result.

By the transformation of the queries, we aim to enhance not only the performance of the system. Due to the transformation of queries, less data are transmitted to service providers. In general, this will lead to more privacy by data minimization and data avoidance, but it is not known, if the anonymized result still contains personal information.

We will then apply query containment techniques to compare the queries performed on the fog layer with the privacy constraints that are expressed by views in ML-QL. We therefore have to extend the query containment techniques to the ML-QL base operations used in our approach.

Our work will extend our preliminary work on privacy-aware query processing [13] where we used simple SPJ queries up to now, and the work in [14, 15] where we already added aggregate functions and arithmetic constraints to the simple SPJ queries.

This results in the research challenges of query transformation (Capability-Based Query Rewriting), checking query containment and privacy, and incorporating the quality, resource and privacy constraints given.

4.3 Our Vision: Optimal Pushdown, Proving Equivalence of Solutions

The capabilities / constraints detected and defined in earlier phases, are now used to influence the transformation of the ML-QL queries. We will integrate the time and space constraints of appliances (Personal Layer) and IoT leaf nodes as well as the quality, energy and privacy constraints into the query transformation, resulting in a multi-level capability-based query rewriting.

One of the aspects of the open research problems is a necessary technique to prove equivalence of (1) centralized complex data analysis algorithms such as complex database queries, and (2) a large number of small queries on IoT devices that are also the sources of the data to be analyzed. For the purpose of this research problem, we introduce some techniques of database theory such as query containment, the Answering Queries using Views (AQuV) problem, and the CHASE algorithm as a universal tool in theoretical database research in the next section.

5 The Database Theory behind BiSAM

Our approach to an optimal, automated distribution of complex queries from the cloud to the Internet of Things, is a technique based on three different but related concepts of (relational) database theory. The optimal distribution of complex queries to capacity-restricted subsystems can be calculated by combining the Query Containment Problem (QCP), Answering Queries using Views (AQuV), and the CHASE.

5.1 Query Containment

The formal principles of query containment and query equivalence and their complexity proof of NP-completeness for conjunctive queries were introduced by Chandra and Merlin in [4]. The article also proves that the QCP is undecidable for general queries. The query containment problem is defined as follows:

A query Q_2 is included in the query Q_1 ($Q_2 \sqsubseteq Q_1$), exactly when for each database instance $d \in D$ it is true that query results of Q_2 are a subset of query results of Q_1, i.e. $\forall d \in D : Q_2(d) \subseteq Q_1(d)$. Two queries Q_1 and Q_2 are equivalent ($Q_1 \equiv Q_2$), exactly when each of the two queries contains the other one: $Q_2 \sqsubseteq Q_1 \wedge Q_1 \sqsubseteq Q_2$, i.e. the queries have equivalent results.

For simple conjunctive queries (consisting of simple selections for equality, projections and connected), Chandra and Merlin [4] show that the problem is solvable. Based on this work, many other approaches have been developed for further classes of queries: In [43] and [1] additional negated query predicates are allowed in the query. Klug [22] extends the possible comparison predicates by further arithmetic comparison operators ($<, \leq, \geq, >$).

Klug [21] allows the integration of simple aggregates (sum, max, min, count) by extending relational algebra and the relational calculus. More complex query containment considerations, particularly with regard to the combination of aggregates with arithmetic and arithmetic comparisons (including negation) and functional dependencies and inclusion dependencies, are considered by Can Türker in [41].

5.2 Answering Queries using Views (AQuV)

A special case of the QCP is the problem, whether a query against a database can also be expressed by a query only against user views. The reason can be that a user was only granted access to a set of restricted views and — due to privacy reasons — not to the database relations directly. Different AQuV techniques calculate a query, performing the same query only with the defined views (if possible), or calculating the maximal subset of the result by the use of views, at least.

The classical techniques for AQuV are the bucket algorithm [8] and the MiniCon algorithm [34]. MiniCon is one of the most effective approaches to the AQuV problem. By the extensions or approaches in [2, 17] it is

also possible to integrate semantic integrity constraints into the queries.

Unfortunately, the classical techniques are only suitable for simple SPJ queries and assume that the data sources have unlimited main and disk space for computation. What is missing in the classical approaches are extensions of relational queries to include any combination of aggregate functions, groupings, as well as window functions and fuzzy joins, for example on time stamps. The operations are needed in smart or assistive IoT environments and applications. The previously introduced SAIR operators represent complex queries for this purpose, which can only be analyzed inadequately by the means available so far.

5.3 The CHASE as a Universal Tool

More recent approaches, such as the one presented in [19], use a modification of the CHASE/BACKCHASE procedure using provenance information to solve the AQuV problem. The CHASE was formally introduced or specified as a universal tool in [25, 42]. As one of the first use cases, it was shown how tuple-generating-dependencies (tgds) and equality-generating dependencies (egds) can be integrated into query patterns using the CHASE [3].

The CHASE algorithm also applies a set of so-called st-tgds (source-to-target tgds) to a query. An st-tgd can be interpreted as a query against the source yielding the target as the query result. In this way, we can express views as st-tgds and can CHASE these views into the original query. The expanded query is then optimized by reducing the query to operate only on views. This is done after the CHASE phase in a second BACKCHASE phase.

With more complex applications of the CHASE such as the AQuV problem, two problems arise: (1) While the original, simple CHASE terminates, the extended CHASE on st-tgds for the AQuV problem does not terminate in some cases. (2) The st-tgds for the view definitions as well as the CHASEd query should be simple SPJ-queries. So we will have to start the *Dance with the tiger* (see Section 6) in our research: we have to extend the CHASE because of (2) but have to restrict the AQuV-CHASE because of (1).

5.3.1 What Database Theory Can Do as One Step towards Our Vision

The CHASE solving the AQuV problem can not only incorporate views into a query and reduce / optimize the expanded query by the BACKCHASE phase. As we have been able to express views as st-tgds, we can also express other constraints as some kind of dependencies. In our research, we have to define the following constraints to be CHASEable: (1) time constraints, (2) space constraints (volatile vs non-volatile memory on IoT leaf nodes or appliances), (3) quality constraints (when joining information over similar but not equal timestamps), (4) energy constraints, and (most importantly) (5) the privacy constraints, as well as (6) the query operators avaliable at a lower level.

As an example: If the personal or IoT level is not able to perform a join, but a simple equality selection, these operators available at a specific level should be CHASEd into the original query — similar to the views being CHASEd into the query for the AQuV case.

Thus, we use an extension of the AQuV technique to answer queries using only restricted capabilities of query processors (Capabilty-Based Query Rewriting, here performed by the CHASE). The AQuV problem has to be generalized to the AQuO problem (Anwering Queries using Operators), describing restricted query operators available on the devices in the Internet of Things. The idea is to CHASE these operators into the query to be able to BACKCHASE the enhanced query to fragment and remainder queries automatically.

6 Summary and Conclusions

At the end of this article, we would like to summarize our research results and state the goals of our future research: The vision of an Internet of Things, which processes the data coming from IoT devices in a distributed manner locally, and thus represents a privacy-aware database machine.

6.1 Vision: The Internet of Things as a Privacy-Aware Database Machine

Instead of collecting data from IoT devices and then having it analyzed centrally on cluster computers of large service providers, we want to analyze as much data as possible locally and only perform absolutely necessary analysis steps (the remainder queries) centrally.

For the support of assistive systems, we initially have a classical, horizontally distributed architecture on a computer cluster at the time of development, on which methods of situation, activity and intention recognition (SAIR) from sensor data are trained in parallel using Machine Learning techniques. Here we have to evaluate a large number of sensors from comparatively few test subjects. After a dimension reduction, we want to limit the recognition algorithms to fewer sensors, but scale the overall system to a huge number of participants. The evaluation algorithms should then be pushed over different layers (like fog layer and personal layer) to the IoT devices.

For the push of analysis operators we have shifted as many partial queries as possible to deep layers of our vertical architecture in a kind of algebraic optimization of database queries. So far we can already show with this technique that with respect to the algebraic rules known to us, the operators are executed as close as possible to the data sources.

However, in order to determine the optimum of such a vertical distribution, we have to use general procedures of query containment and the determination of the equivalence of queries. These are realized in database theory using the CHASE procedure. We want to extend the CHASE procedures, which could already solve the Answering Queries using Views problem, to an Answering Queries using Capabilities (or Operators) problem and at the same time CHASE the privacy policies as integrity conditions into the central query, in order to automatically derive the optimal, privacy-aware distribution of the query on the vertical architecture.

This technique would allow that the analysis algorithms continue to be developed at a central level on a (virtual) central database and to be automatically transformed into a privacy-aware variant that evaluates as close as possible to the sensors. A prerequisite for our technique is that the necessary analyses can be expressed with the database language SQL.

6.2 Underlying Theory: The Hammer and the Dance

As we mentioned earlier in this paper, an automatic pushdown of fragments of analysis algorithms to IoT devices such as smartphones would have been helpful to transform a centralized Proximity Tracing App to a decentralized one for detecting SARS-CoV-2 infection chains. The decentralized Proximity Tracing App is one of the building bricks of a successful *The Hammer and the Dance* strategy in fighting against the pandemic [35].

In this strategy, the hammer is an extensive lockdown with exit restrictions, restricted social contacts (social distancing), school and business closures to severely limit the spread of the virus. If this hammer was effective, one can move on to the dance phase and relax the regulations as long as the infection process remains manageable, infection chains can be traced (the App helps) and infected persons can be isolated quickly. Should these criteria deteriorate, loosening the regulations must be reduced a little. Loosening and tightening a leash would be the two necessary actions to be able to dance with a dangerous tiger. In this case the dangerous tiger is called SARS-CoV-2.

Analogously, in relational database theory there is a universal tool, a hammer, that can already solve basic problems of the integration of integrity constraints into a query and the integration of given user views into a query. In some cases, however, conditions have to be tightened in order to make the solution of the problem more efficient. Sometimes conditions and operators have to be generalized to be able to express SAIR problems in SQL using Machine Learning methods. This loosening of those restrictions valid for the CHASE up to now, however, must be done pragmatically and nevertheless in a controlled manner so that the dance with the dangerous tiger called undecidability or inefficiency still remains successful.

ACKNOWLEDGEMENTS

Our BiSAM vision is supported by Martin Kasparick and Dirk Timmermann: they added an IoT-specific perspective by integrating DB operators into IoT nodes, and considered scalability, flexibility, energy and quality requirements. Additionally, two team members of our Database Research Group have been supporting this work over the last six years: very special thanks to Holger Meyer and Dennis Marten! We acknowledge financial support by Deutsche Forschungsgemeinschaft and the University of Rostock within the funding programme Open Access Publishing.

REFERENCES

[1] F. N. Afrati, C. Li, and P. Mitra, “On containment of conjunctive queries with arithmetic comparisons,” in *Advances in Database Technology - EDBT 2004, 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004, Proceedings*, ser. Lecture Notes in Computer Science, vol. 2992. Springer, 2004, pp. 459–476.

[2] Q. Bai, J. Hong, and M. F. McTear, “Query rewriting using views in the presence of inclusion dependencies,” in *Fifth ACM CIKM International Workshop on Web Information and Data Management (WIDM 2003), New Orleans, Louisiana, USA, November 7-8, 2003*. ACM, 2003, pp. 134–138.

[3] C. Beeri and M. Y. Vardi, “A proof procedure for data dependencies,” *J. ACM*, vol. 31, no. 4, pp. 718–741, 1984. [Online]. Available: http://doi.acm.org/10.1145/1634.1636

[4] A. K. Chandra and P. M. Merlin, “Optimal implementation of conjunctive queries in relational data bases,” in *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*. ACM, 1977, pp. 77–90.

[5] N. A. Chaudhry, K. Shaw, and M. Abdelguerfi, Eds., *Stream Data Management*, ser. Advances in Database Systems. Springer, 2005, vol. 30.

[6] A. V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, and R. Buyya, "Fog computing: Principles, architectures, and applications," *CoRR*, vol. abs/1601.02752, 2016.

[7] O. Diallo, J. J. P. C. Rodrigues, M. Sene, and J. L. Mauri, "Distributed database management techniques for wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 2, pp. 604–620, 2015.

[8] O. M. Duschka and M. R. Genesereth, "Answering recursive queries using views," in *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 12-14, 1997, Tucson, Arizona, USA*. ACM Press, 1997, pp. 109–116.

[9] H. Grunert and A. Heuer, "Big Data und der Fluch der Dimensionalität: Die effiziente Suche nach Quasi-Identifikatoren in hochdimensionalen Daten," in *Proceedings of the 26th GI-Workshop Grundlagen von Datenbanken, Bozen-Bolzano, Italy, October 21st to 24th, 2014.*, ser. CEUR Workshop Proceedings, vol. 1313. CEUR-WS.org, 2014, pp. 29–34.

[10] H. Grunert and A. Heuer, "Generating privacy constraints for assistive environments," in *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments, PETRA 2015, Corfu, Greece, July 1-3, 2015*, F. Makedon, Ed. ACM, 2015, pp. 271–274.

[11] H. Grunert and A. Heuer, "Slicing in Assistenzsystemen - Wie trotz Anonymisierung von Daten wertvolle Analyseergebnisse gewonnen werden können," in *Proceedings of the 27th GI-Workshop Grundlagen von Datenbanken, Gommern, Germany, May 26-29, 2015.*, ser. CEUR Workshop Proceedings, vol. 1366. CEUR-WS.org, 2015, pp. 24–29.

[12] H. Grunert and A. Heuer, "Datenschutz im PArADISE," *Datenbank-Spektrum*, vol. 16, no. 2, pp. 107–117, 2016.

[13] H. Grunert and A. Heuer, "Privacy protection through query rewriting in smart environments," in *Proceedings of the 19th International Conference on Extending Database Technology, EDBT 2016, Bordeaux, France, March 15-16, 2016, Bordeaux, France, March 15-16, 2016*. OpenProceedings.org, 2016, pp. 708–709.

[14] H. Grunert and A. Heuer, "Rewriting complex queries from cloud to fog under capability constraints to protect the users' privacy," *Open Journal of Internet Of Things (OJIOT)*, vol. 3, no. 1, pp. 31–45, 2017, special Issue: Proceedings of the International Workshop on Very Large Internet of Things (VLIoT 2017) in conjunction with the VLDB 2017 Conference in Munich, Germany. [Online]. Available: http://nbn-resolving.de/urn:nbn:de:101:1-2017080613421

[15] H. Grunert and A. Heuer, "Query rewriting by contract under privacy constraints," *Open Journal of Internet Of Things (OJIOT)*, vol. 4, no. 1, pp. 54–69, 2018, special Issue: Proceedings of the International Workshop on Very Large Internet of Things (VLIoT 2018) in conjunction with the VLDB 2018 Conference in Rio de Janeiro, Brazil. [Online]. Available: http://nbn-resolving.de/urn:nbn:de:101:1-201808051930310938685 8

[16] H. Grunert, M. Kasparick, B. Butzin, A. Heuer, and D. Timmermann, "From cloud to fog and sunny sensors," in *In Proceedings of the Lernen, Wissen, Daten, Analysen (LWDA)*. LWDA, 9 2016, pp. 83–88.

[17] J. Gryz, "Query rewriting using views in the presence of functional and inclusion dependencies," *Inf. Syst.*, vol. 24, no. 7, pp. 597–612, 1999.

[18] A. Heuer and A. Lubinski, "Data reduction - an adaptation technique for mobile environments," in *Interactive Applications of Mobile Computing (IMC'98)*, 1998.

[19] I. Ileana, B. Cautis, A. Deutsch, and Y. Katsis, "Complete yet practical search for minimal query reformulations under constraints," in *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*. ACM, 2014, pp. 1015–1026.

[20] B. John, "Vergleichende Analyse von Datenschutzalgorithmen und -konzepten," Bachelor's Thesis, University of Rostock, 2016.

[21] A. C. Klug, "Equivalence of relational algebra and relational calculus query languages having aggregate functions," *J. ACM*, vol. 29, no. 3, pp. 699–717, 1982.

[22] A. C. Klug, "On conjunctive queries containing inequalities," *J. ACM*, vol. 35, no. 1, pp. 146–160, 1988.

[23] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query

processing system for sensor networks," *ACM Transactions on Database Systems (TODS)*, vol. 30, no. 1, pp. 122–173, 2005.

[24] D. Maier, J. Li, P. A. Tucker, K. Tufte, and V. Papadimos, "Semantics of data streams and operators," in *Database Theory — ICDT 2005, 10th International Conference, Edinburgh, UK, January 5–7, 2005, Proceedings*, 2005, pp. 37–52.

[25] D. Maier, Y. Sagiv, and M. Yannakakis, "On the complexity of testing implications of functional and join dependencies," *J. ACM*, vol. 28, no. 4, pp. 680–695, 1981.

[26] D. Marten and A. Heuer, "A framework for self-managing database support and parallel computing for assistive systems," in *Proceedings of the 8th ACM International Conference on Pervasive Technologies Related to Assistive Environments, PETRA 2015, Corfu, Greece, July 1-3, 2015*, F. Makedon, Ed. ACM, 2015, pp. 251–254.

[27] D. Marten and A. Heuer, "Machine learning on large databases: Transforming Hidden Markov Models to SQL statements," *Open Journal of Databases (OJDB)*, vol. 4, no. 1, pp. 22–42, 2017. [Online]. Available: http://nbn-resolving.de/urn:nbn:de:101:1-2017100112181

[28] D. Marten, H. Meyer, D. Dietrich, and A. Heuer, "Sparse and dense linear algebra for machine learning on Parallel-RDBMS using SQL," *Open Journal of Big Data (OJBD)*, vol. 5, no. 1, pp. 1–34, 2019. [Online]. Available: http://nbn-resolving.de/urn:nbn:de:101:1-201812231834106917295 7

[29] McObject Inc., "McObject updates eXtremeDB real-time database system," Nov. 2009.

[30] M. L. Möller, "Stromdatenverarbeitung für Data-Science-Anwendungen basierend auf Sensordaten," Master's Thesis, University of Rostock, 2017.

[31] F. Morvan and A. Hameurlain, "A mobile relational algebra," *Mobile Information Systems*, vol. 7, no. 1, pp. 1–20, 2011.

[32] A. Nori, "Mobile and embedded databases," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '07. ACM, 2007, pp. 1175–1177.

[33] PEPP-PT Team, "Pan-european privacy-preserving proximity tracing – high-level overview," PePP-PT e.V. i.Gr, Tech. Rep., 2020.

[34] R. Pottinger and A. Y. Halevy, "MiniCon: A scalable algorithm for answering queries using views," *VLDB J.*, vol. 10, no. 2-3, pp. 182–198, 2001.

[35] T. Pueyo, "Coronavirus: The hammer and the dance — what the next 18 months can look like, if leaders buy us time," https://medium.com/@tomaspueyo, last access: 04.05.2020.

[36] M. Schröder, S. Bader, F. Krüger, and T. Kirste, "Reconstruction of everyday life behaviour based on noisy sensor data," in *Proceedings of the 8th International Conference on Agents and Artificial Intelligence*, Rome, Italy, Feb. 2016, pp. 430–437.

[37] J. Schützel, H. Meyer, and A. M. Uhrmacher, "A stream-based architecture for the management and on-line analysis of unbounded amounts of simulation data," in *SIGSIM Principles of Advanced Discrete Simulation, SIGSIM-PADS '14, Denver, CO, USA, May 18-21, 2014*, 2014, pp. 83–94.

[38] M. I. Seltzer, "Berkeley DB: A retrospective," *IEEE Data Eng. Bull.*, vol. 30, no. 3, pp. 21–28, 2007.

[39] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, May 2016.

[40] C. Troncoso, K. Paterson, B. Preneel, S. Guerses, M. Veale, C. Cremers, R. Binns, and C. C. et al., "Decentralized privacy-preserving proximity tracing," EPFL, ETHZ, KU Leuven, TU Delft, University College London, CISPA, University of Oxford, University of Torino / ISI Foundation, Tech. Rep., 2020.

[41] C. Türker, *Semantic Integrity Constraints in Federated Database Schemata*, ser. DISDBIS. Infix Verlag, St. Augustin, Germany, 1999, vol. 63.

[42] M. Y. Vardi, "Inferring Multivalued Dependencies From Functional and Join Dependencies," *Acta Inf.*, vol. 19, pp. 305–324, 1983.

[43] F. Wei and G. Lausen, "Containment of conjunctive queries with safe negation," in *Database Theory - ICDT 2003, 9th International Conference, Siena, Italy, January 8-10, 2003, Proceedings*, ser. Lecture Notes in Computer Science, vol. 2572. Springer, 2003, pp. 343–357.

[44] K. Yordanova and T. Kirste, "A process for systematic development of symbolic models for activity recognition," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, Dec. 2015.

AUTHOR BIOGRAPHIES

Andreas Heuer got his PhD and Habilitation at the TU Clausthal in 1988 and 1993, resp. Since 1994, he is full professor for Database and Information Systems at the University of Rostock. He is interested in fundamentals of database models and languages, and in big data analytics, here especially performance, privacy and provenance.

Hannes Grunert was born in Ribnitz-Damgarten (Germany). He received his B.Sc. and his M.Sc. degree in Computer Science from the University of Rostock, Germany, in 2011 and 2013, respectively. He is currently a PhD student at the University of Rostock. His work is focused on privacy aware query processing.