ASTES

Advances in Science, Technology and Engineering Systems Journal Vol. 5, No. 5, 1282-1287 (2020)

www.astesj.com

Special Issue on Multidisciplinary Innovation in Engineering Science & Technology

ASTESJ

ISSN: 2415-6698

An Overview on CryptDb and Word2vec Approaches

Hana Yousuf[1], Asma Qassem Al-Hamad[2], Said Salloum[1,3,*]

[1]*Faculty of Engineering & IT, The British University in Dubai, 345015, UAE*

[2]*Librarian and information department, Imam Abdulrahman Bin Faisal University, 1982, KSA*

[3]*Research Institute of Sciences & Engineering, University of Sharjah, 27272, UAE*

ARTICLE INFO

Article history:
Received: 24 June, 2020
Accepted: 07 October, 2020
Online: 24 October, 2020

Keywords:
Big data
CryptDB
SQL databases
Word2Vec
Secure vectorization

ABSTRACT

Big data is a vast data set that was used in many areas. Online applications are subject to theft of confidential information because opponents can exploit software errors to access private data, and because curious or malicious officials can capture and lose data. CryptDB is a functional system that provides security and confidentiality through a set of operations. The obvious confidentiality of these attacks is for applications supported by SQL databases. It works by executing SQL queries on encrypted data using robust coding systems that support SQL. Word2Vec outputs word vectors that can be displayed as a large piece of text, or even we first train data. Word2Vec forms and word similarity assessment. Without a doubt, this article calls for proper research that sheds light on the security features using CryptDB to prevent data theft and privacy breaches in the server. The motivation of this research is to have an overview of CryptDB and Word2Vec implementation on the existing research approaches.

1. Introduction

The theft of private data is a big issue [1–3], especially for online applications [4–11]. The SQL databases may be attacked and vulnerable to sensitive information or theft since they can exploit the bugs to gain access to private information. Also, the attackers can capture and leak the data to those who require them. Hence, the dataset must be safeguarded. CryptDB provides confidentiality against the attackers through SQL databases [10]. The processes are encrypted in SQL by collecting effective SQL aware encryption schemes. The process can also be used to encrypt the credential keys so that an item can be decrypted with simply a password. Since the processes and data are encrypted, the database administrator will not view the data. Even if there is an attack on the server, the attacker will not access the decrypted data. CryptDB can handle multiple queries simultaneously and has less overhead.

Word2vec is one type of word embedding technique used to represents a string in a set of real numbers. This is a technique used in Natural Language Processing (NLP) through deep learning to extract data from the requested document. Word2Vec represents the input text as a statistical or vector form using a two-layer neural net that process text using. Word2vec's applications extend on the far side than analyzing sentences; it can be used in an application that has a well-defined pattern

The motivation to carry out this research is to have an overview of Word2Vec and CryptDB approaches and identify the challenges in the current process in both techniques.

1.1. CryptDb

One of the most widely known machines is employed in various shifts in classification tasks. Two threats are tackled by CryptDB [12]. The first of these is a curious database administrator (DBA) that attempts to learn private data (such as financial statements, health records, personal information, etc.) by sneaking into the DBMS server; however, CryptDB does not allow the DBA to access the private data. The other threat is an adversary that acquires full control of the application and DBMS services. Here, no guarantees can be offered by CryptDB for users logged in the application when an attack occurs; however, it can make sure that the data of the users who are not logged in is protected from threats. A simple solution is to create a different database encryption key

*Corresponding Author: Said Salloum, University of Sharjah, UAE. Tel: +971507679647 Email: ssalloum@sharjah.ac.ae

for every user not applicable for their data in applications, including shared data, such as conference review sites and bulletin boards [13].

The CryptDB's function is to perform queries on encrypted information. The main reason for this practice is that SQL uses a specific set of factors; each can effectively support encrypted data [13]. CryptDB combats with these challenges using three core ideas:

- The first one is to run SQL questions on encrypted information. CryptDB applies this concept by employing an encryption methodology that supports SQL and takes advantage of the truth that all SQL inquiries comprise a well-defined set of primitive factors, like Equality Confirmation, framework comparisons, total (amounts), and links. By adjusting known encryption systems (for correspondence, increments, and work confirmation) and employing a new encryption strategy to ensure the protection of joins, CryptDB encrypts each data component so that DBMS can execute the converted information [14]. CryptDB is practical security since it uses essentially symmetric-key encryption in order to prevent completely identical encryption, and runs an unmodified DBMS program (utilizing custom capacities).
- The second strategy is flexible encryption based on the query. A few encryption systems pass more data to the DBMS server than others but should undergo some queries. To avoid leaking all information encryption to the database management system, CryptDB carefully alters the SQL encryption framework for any specific data element, based on queries. To execute these modifications effectively, CryptDB utilizes 'onions of encryption'. Onions are a new approach to store numerous encryption texts within the database and maintain a distance from exorbitant re-encryption [13].
- The third concept is to link encryption keys to users' passwords so that every information within the database can be decrypted by employing a keychain established within the client's password who can approach that data. If the client does not log on to the app and the rival does not know the password, the adversary will not decode the user's information even if the DBMS and the application server are completely under threat [13]. To form a critical chain that captures the app's protection and sharing arrangement, CryptDB permits the developers to give policy comments through the SQL app chart and identifying clients who have access to each part of the information.

1.2. Word2Vec

Google developed Word2vec in 2013 [12], which is a neural network through which text data is processed. Word2vec is not a single algorithm, it is made up of two learning models, which are the Common Bag of Word bag (CBOW) and Skip grams. The word is predicted by CBOW based on its context, while in Skip-Gramm, context determines a word. Word2Vec ultimately developed in a model word vector by feeding the text. Word2Vec first generates a vocabulary from group training text and learning vector representations for every word [12]. The architecture of the word2vec algorithm is given in Figure 1.

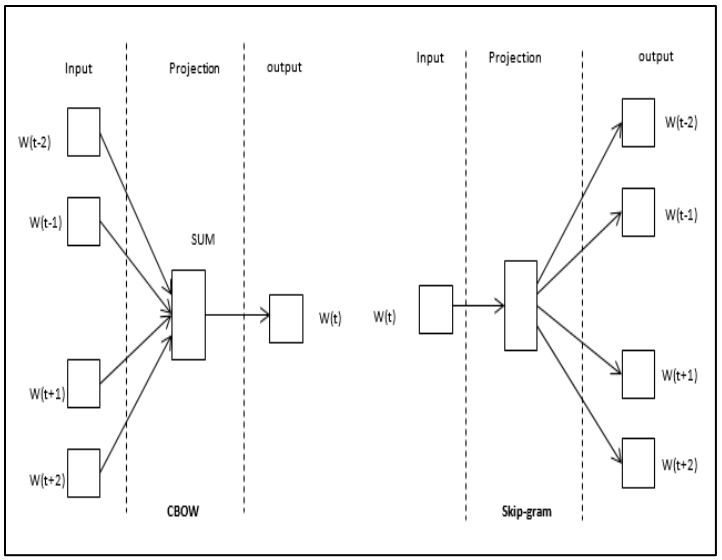

Figure 1: Word2Vec Architecture

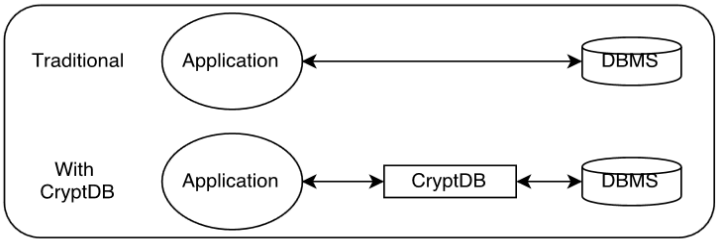

Figure 2: CryptDB along with traditional DB [12].

The purpose of using Word2vec is to cluster the vectors of similar words along in vector-space, then detect the similarity by measuring cosine similarity; it creates vectors that are distributed numerical representations of the word. Word2vec will build extremely accurate guesses of a word's definition based on previous presences. The guesses are used to construct a word that associate with the other word in the sentence.

2. Literature Review

A study by [15] has used effective Galois Field mathematical environments for encrypting algorithms effectively. The increase in network sensor systems and network databases has led to more interest in using cryptology in sensors algorithms and databases. Vectorized Advanced Encrypted Standard (AES) has been implemented for the database systems. Due to vectorization, the implementation is very small and requires 100 times smaller code than ordinary AES implementation. It is also fast and has an effective design. The implementation achieves higher speed which is comparable to real-time prototyping with OpenSSL with good database analytics and processing.

Researchers of [16] have introduced a ciphertext policy with constant sizes known as Cipher-text Policy Hidden Vector Encryption (CP-HVE). An HVE is a unique type of anonymous IBE, which uses identity as the main parameters where the attributes linked with the ciphertext or secret keys contain wild-cards. Two different schemes have been used, one with a composite ordered bi-linear group and a prime ordered bilinear group. Both of these schemes have high security by differentiating

the ciphertext from plaintext. This vectorized encryption has the ability to have a regular size than the other HVE methods.

HVE was also implemented by [17], a type of predictive encryption technique that supports conjunctive equality and ranges on the encrypted data. A novel HVE scheme has been built that is fully secure under standard assumptions. The proposed scheme is based on bilinear maps that are more advantageous for the private keys and pairing computations for decryptions. Tag-based dual system encryption has been developed to hide the vector components and compress the tag's values into one. It is challenging to extend the HVE along with IBE for supporting a higher hierarchical mechanism. Hence, it is still challenging to construct HVE for providing encryption-based security. The number of pairings in the computations is high, which must be reduced in the future to increase the speed of the processes.

In [18], author has described the types of CryptDB. Random (RND) is a type of CryptDB that relatively provides the most security that is selected under a probabilistic adaptive plain text attack. Here, two equal halves of data are mapped into different ciphertexts. RND ensures that computations are performed on ciphertexts directly. RND uses encryptions like AES and Blowfish simultaneously with a random initialization vector. AES is used more than Blowfish for 64-bit block sizes since AES has 128-bit blocks that cause the ciphertexts to be very long. For high security, it is assumed that the server does not change results.

3. CryptDB is a solution for facing the threats

CryptDB works by intercepting all SQL queries in that database proxy and rewrites queries to perform for encrypted data. All inquiries pass through the agent. The agent encrypts and decrypts all data, and some parameters change while maintaining the query's semantics. The DBMS server never receives decryption. A clear text key so that confidential data is not displayed to ensure that strange DBA cannot access private information (Threat 1).

To protect themselves from compromises in applications, proxy servers, and DBMS servers (Threat 2), developers comment on the specific SQL schema different principles; its keys to decode different parts database. They are also making a slight change to their apps, providing encryption keys to the proxy. The agent specifies which parts of the database are to encrypt under any key. The result is that CryptDB guarantees the confidentiality of data owned by users who are not logged in within one. The compromise that just logs in the admin recognizes and corrects the settlement. CryptDB protects data confidentiality, but it does not guarantee its safety, freshness, or completeness of the results returned to implementation. The opponent who threatens the application, the agent, or a malicious DBMS or DBA server that can delete one or all data stored in the database. Attacks on users' computers, Cross-site scripting outside of CryptDB. The security guarantees are provided for in these threat models [19].

3.1. Threat 1: DBMS Server compromise

CryptDB provides security against a curious DBA or another external attacker in this threat while offering complete access to the DBMS server's data. It focuses on confidentiality (data privacy) rather than availability or integrity. It is presumed that the attacker is passive, wanting to get access to confidential data without modifying queries presented by the application, query results, or the data within DBMS. This threat comprises DBMS software being compromised, gaining root access to DBMS, and access to the RAM of physical machines. There is now a greater degree of database consolidation in enterprise data centers, databases are being outsourced to public cloud computing infrastructures, and third party-DBAs are being used, which is why it is becoming very important to deal with this threat [19–21].

The goal of CryptDB is to ensure that the data remains secure from this threat by carrying out SQL queries over the encrypted data DBMS server. The agent uses secret keys for encrypting the entire data; it includes or incorporates them in the outgoing DBMS queries [22]. This tries to make it possible to use the DBMS server to process queries for encrypted data, similar to an unencrypted database. This is done by activating the particular functions for data elements required for encrypted data. The DBMS should have the ability to identify the factors it consists of. There is the same column; however, the actual content items are different. Hence, the DBMS server should be activated by the proxy to identify the relationships among the data required for query processing. SQL-enabled encryption should be used as it is capable of adapting vigorously. CryptDB, when issuing requests, manages the relationships it reveals between lines to the server. The order of the items in the column is not known, which is not even required to learn more about the rest of the columns. When DBMS is needed, CryptDB will identify it by carrying out an ORDER command or determine MAX or MIN items within this column, and not by any other method [22].

3.2. Threat 2: Arbitrary threats

It is possible that the proxy and DBMS server infrastructure experience arbitrary security breaches. Due to the opponent, the method used in Threat 1 is not adequate. It is now possible to access the keys used to encrypt the complete database. To deal with this, different data elements (for example, data for various users) should be encoded using different keys. The application database schema should be suspended to present further privacy guidelines to choose the key developers who would be using each data item. It is still impossible for strange DBA to obtain private data on a DBMS server (Threat 1). The application server or proxy can become decrypted—data from only the presently registered (stored in the proxy). Data from the users who are not active will not be encrypted with the keys available to the attacker and will stay confidential. CryptDB provides substantial guarantees in this configuration in dealing with the arbitrary server-side breaches, including the ones that obtain root access to the proxy or application. CryptDB leaks the majority of the data users that have been active for some time. Concerning an SQL attack, the compromise duration comprises of the SQL queries of the attacker. The system is considered vulnerable until the attacker's email address stays within the database [22], [23].

4. CryptDB Implementation

SQL queries are carried out by CryptDB using encrypted data. There is a lack of trust in the DBMS machines and administrators; however, the application and the proxy can be trusted. CryptDB allows the DBMS server to perform SQL queries on encrypted data, just like how it would be carrying out the same queries on

plaintext data. No modifications in the existing applications are required. Usually, the DBMS query plan for an encrypted query is similar to the actual query, except that the operators forming the query, like projections, selections, aggregates, joins, and orderings, are done on cipher tests. In a few scenarios, modified operators are used. A secret master key MK is stored by CryptDB's proxy, in addition to the database scheme and the existing encryption layers of each column [24]. An anonymized schema is observed by the DBMS server (where the names of tables and columns are substituted with opaque identifiers), encrypted user data, and certain auxiliary tables utilized CryptDB. CryptDB-specific user-defined functions (UDFs) are also provided by CryptDB to the server, using which the server can use ciphertexts for specific functions [24]. Four steps are followed when processing a query in CryptDB.

- A query is presented by an application, which is intercepted and rewritten by the proxy: each table and column name is anonymized. With the master key MK's help, each constant is encrypted in the query using an encryption scheme that is most appropriate for the required operation.
- It is determined by the proxy whether the DBMS server should be provided keys to modify the encryption layers before issuing the query. If this is the case, then an UPDATE query is issued at the DBMS server that brings about a UDF to modify the relevant columns' encryption layer.
- The encrypted query is forwarded by the proxy to the DBMS server, which performs it using standard SQL (often bringing about UDFs for performing aggregation or keyword search).
- The encrypted query result s returned by the DBMS server that is decrypted by the proxy and sent back to the application.

5. Word2Vec Implementation

We prepare our data to be trained using the Word2Vec model. We are taking the below sentence as an example:

We must love life

Step-1: Consider the word 'we' as input and 'must listen' is the output word as follows:

Input	Output
We	must
We	love

Step-2: Now consider 'must' as input word; the output will be the close words as follows

Input	Output
We	must
We	love
Must	we
Must	love
Must	life

Step-3: The same will be done for the rest of the sentence as follows:

Input	Output
We	must
We	love
Must	we
Must	love
Must	life
Love	we
Love	must
Love	life
Love	must
Love	live

At the end of creating a training sample for the sentence, we gotten samples. After that, we will obtain Word2Vec embedding by having a data set consists of 4000 unique words and word vectors size 80 each.

- Vocabulary Size = 4000
- Word vector size =80

The input will be our input vector, and the output will be the probability of nearby vectors. The learned weight matrix can be extracted after the model finishes the training process, and the word vector can be extracted, as shown in Figure 2 and Figure 3.

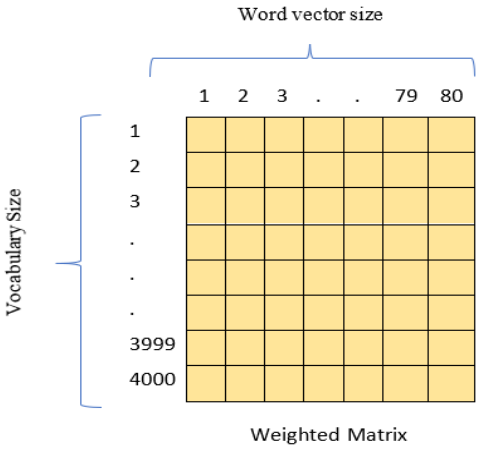

Figure 2: The weight matrix has a size of 4000 x 80.

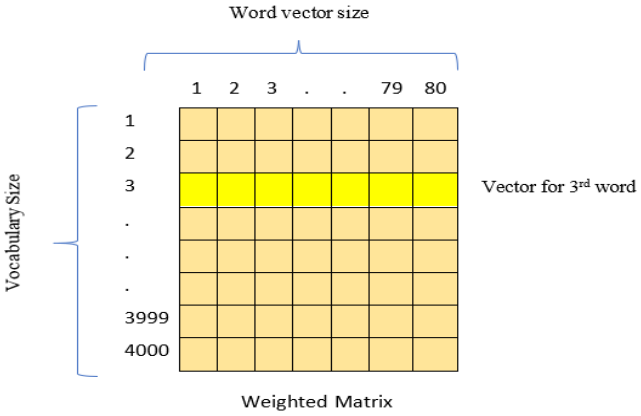

Figure 3: Vector representation for the 3rd word

The above shows how to have a word embedding using Word2Vec. Similar words in the vocabulary set will have similar vectors pointing toward the same direction for example, life and living will have the same direction as shown in Figure 4.

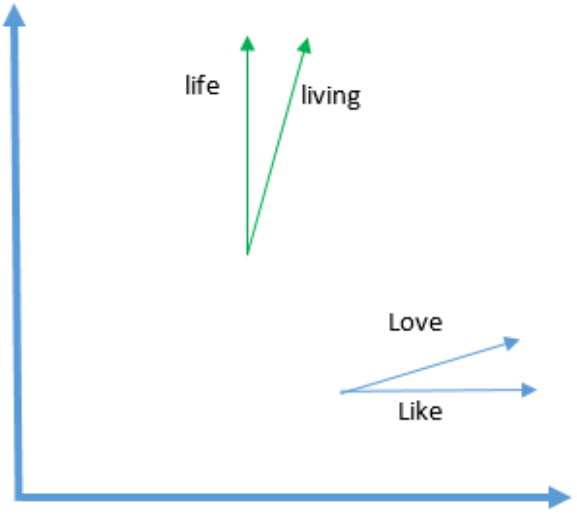

Figure 4: The vector direction for similar words

6. Conclusions and Future Work

CryptDB is a system that offers a logical and powerful level of confidentiality against two considerable threats that attack database-backed applications, i.e., curious DBAs and arbitrary compromises of the application server and the DBMS. Three steps are followed by CryptDB to achieve its objectives: efficiently performing queries over encrypted data through a novel SQL-aware encryption strategy, dynamically modifying the encryption level using onions of encryption to decrease the information given to the untrusted DBMS server, and linking encryption keys to under passwords such that only authorized users can get access to encrypted data. Word2Vec method is an effective data processing technique. It takes into account the significant data dimension issue when handling large-scale training data to offer a means of grouping similar data. This method may be employed to decrease the data dimension.

The CryptDB and Word2Vec approaches will be used together to provide a secure word embedding.

Acknowledgment

This is a part of project done in British University in Dubai.

References

[1] M. Alshurideh, B. Al Kurdi, S.A. Salloum, I. Arpaci, M. Al-Emran, "Predicting the actual use of m-learning systems: a comparative approach using PLS-SEM and machine learning algorithms," Interactive Learning Environments, 1–15, 2020.
[2] M. AlShamsi, S.A. Salloum, M. Alshurideh, S. Abdallah, Artificial Intelligence and Blockchain for Transparency in Governance, Springer: 219–230.
[3] J. Almaazmi, M. Alshurideh, B. Al Kurdi, S.A. Salloum, "The Effect of Digital Transformation on Product Innovation: A Critical Review," in International Conference on Advanced Intelligent Systems and Informatics, Springer: 731–741, 2020.
[4] A.Y. Zainal, H. Yousuf, S.A. Salloum, "Dimensions of Agility Capabilities Organizational Competitiveness in Sustaining," in Joint European-US Workshop on Applications of Invariance in Computer Vision, Springer: 762–772, 2020.
[5] H. Yousuf, S. Salloum, "Survey Analysis: Enhancing the Security of Vectorization by Using word2vec and CryptDB."
[6] S.K. Yousuf H., Lahzi M., Salloum S.A., "Systematic Review on Fully Homomorphic Encryption Scheme and Its Application.," In: Al-Emran M., Shaalan K., Hassanien A. (Eds) Recent Advances in Intelligent Systems and Smart Applications. Studies in Systems, Decision and Control, Vol 295. Springer, Cham, 2021.
[7] S.A. Salloum, R. Khan, K. Shaalan, "A Survey of Semantic Analysis Approaches," in Joint European-US Workshop on Applications of Invariance in Computer Vision, Springer: 61–70, 2020.
[8] S.A. Salloum, M. Alshurideh, A. Elnagar, K. Shaalan, "Machine Learning and Deep Learning Techniques for Cybersecurity: A Review," in Joint European-US Workshop on Applications of Invariance in Computer Vision, Springer: 50–57, 2020.
[9] S.A. Salloum, M. Alshurideh, A. Elnagar, K. Shaalan, "Mining in Educational Data: Review and Future Directions," in Joint European-US Workshop on Applications of Invariance in Computer Vision, Springer: 92–102, 2020.
[10] S.A. Salloum, A.Q. AlHamad, M. Al-Emran, K. Shaalan, A survey of Arabic text mining, 2018, doi:10.1007/978-3-319-67056-0_20.
[11] S.A. Salloum, M. Al-Emran, A.A. Monem, K. Shaalan, Using text mining techniques for extracting information from research articles, 2018, doi:10.1007/978-3-319-67056-0_18.
[12] Y.-C. Chang, M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in International Conference on Applied Cryptography and Network Security, Springer: 442–455, 2005.
[13] K.C. M. Viktor, "Big data: A revolution that will transform how we live, work, and think," Houghton Mifflin Harcourt, 2013.
[14] N. Aburawi, CryptDB mechanism on graph databases, 2020.
[15] J. Kepner, V. Gadepally, B. Hancock, P. Michaleas, E. Michel, M. Varia, "Parallel vectorized algebraic AES in Matlab for rapid prototyping of encrypted sensor processing algorithms and database analytics," in 2015 IEEE High Performance Extreme Computing Conference (HPEC), IEEE: 1–8, 2015.
[16] T.V.X. Phuong, G. Yang, W. Susilo, "Efficient hidden vector encryption with constant-size ciphertext," in European Symposium on Research in Computer Security, Springer: 472–487, 2014.
[17] J.H. Park, K. Lee, W. Susilo, D.H. Lee, "Fully secure hidden vector encryption under standard assumptions," Information Sciences, **232**, 188–207, 2013.
[18] R.A. Popa, C.M.S. Redfield, N. Zeldovich, H. Balakrishnan, "CryptDB: protecting confidentiality with encrypted query processing," in Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, 85–100, 2011.
[19] K. Lang, "Newsgroups Data Set," Available at: Qwone. Com/~ Jason/20Newsgroups/.[Accessed 30-Sep-2015], 20AD.

[20] H.H.O. Nasereddin, A.J. Darwesh, "An Object Oriented Programming on Encrypted Database System (CryptDB)," Journal of Talent Development and Excellence, **12**(1), 5140–5146, 2020.
[21] X. Jiang, X. Kong, Z. Xu, "Research on order-preserving encryption scheme based on CryptDB," in Journal of Physics: Conference Series, IOP Publishing: 32106, 2020.
[22] S.S.M. Chow, J.-H. Lee, L. Subramanian, "Two-Party Computation Model for Privacy-Preserving Queries over Distributed Databases.," in NDSS, Citeseer, 2009.
[23] H. Yousuf, A.Y. Zainal, M. Alshurideh, S.A. Salloum, Artificial Intelligence Models in Power System Analysis, Springer: 231–242.
[24] S. Rizvi, A. Mendelzon, S. Sudarshan, P. Roy, "Extending query rewriting techniques for fine-grained access control," in Proceedings of the 2004 ACM SIGMOD international conference on Management of data, 551–562, 2004.