JKAU: Comp. IT. Sci., **Vol. 8** No. 1, pp: 1 – 9 (1440 A.H. / 2019 A.D.)

Doi: 10.4197/Comp. 8-1.1

Access Control Models in NoSQL Databases: An Overview

Ashwaq A. Alotaibi, Reem M. Alotaibi and **Nermin Hamza**

Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

aalotaibi0553@stu.kau.edu.sa

Abstract. Recently non-relational databases known as NoSQL have become most popular for handling a huge amount of data. Many organizations move from relational databases towards NoSQL databases due to the growing popularity of cloud computing and big data. NoSQL database is designed to handle unstructured data like documents, e-mails, and social media efficiently. It uses distributed and cooperating devices to store and retrieve data. As a large number of people storing sensitive data in NoSQL databases, security issues become critical concerns. NoSQL has many advantages like scalability and availability, but it suffers from some security issues like weak authorization mechanisms. This paper reviews the different models of NoSQL databases and the security issues concerning these databases. In addition, we present the existing access control models in different NoSQL databases.

Keywords: Access control; Big data; Databases; NoSQL database; NoSQL models; Security issues.

1. Introduction

Over the past two decades, data has increased widely in different fields. Depending on a report from International Data Corporation (IDC) in 2011, the volume of data created and replicated in the world was 1.8 trillion gigabytes, which grew by around nine times during five years [1]. This number will be double two in the future. With the enormous increase in global data, the term big data is used to describe huge datasets [2]. The term big data refers to data with high volume, velocity, and variety [3].

Dealing with big data using relational database becomes more and more complicated [4]. Google, Amazon, and Facebook are among the companies that detect the limitations of the relational databases in processing big data and satisfying its users' requirements. The relational database is inappropriate for several requirements such as scalability, control over performance, high availability, low latency, workload distribution and handles big data applications [5].

To overcome these limitations, a new database with a new data management model called NoSQL is designed. The NoSQL stands for "Not only SQL". It offers dynamic schemas, flexibility in a data model, scalability and effciently processing big data. It is designed to handle unstructured data like documents, e-mails, and social media in an effcient way. NoSQL database uses distributed and collaborative devices for storing and retrieving data.

Increasing the use of big data and cloud computing led organizations to move from relational databases towards NoSQL databases [10]. These NoSQL databases are speedily developing and spreading in many companies, and compared to relational databases, NoSQL is considered a suitable choice. Applications of web services, e-commerce, mobile computing, and social media require NoSQL databases to store and process huge amounts of data [6].

Many related organization and websites have ranked the NoSQL databases by its popularity [7]. The top five open source NoSQL databases are MongoDB, Cassandra, CouchDB, Hypertable, and Redis. These systems have critical issues in security. Even though the advantages of NoSQL databases are making them very popular, these systems have critical issues in security [8]. They suffered from many security issues like lack of proper authentication, encryption and fine-grained authorization [9].

One of the most important issue is relevant to the poor data protection mechanisms they currently provide [10]. Access control is the basic unit in data protection of any database management system [8]. Granularity level in access control refers to the size of data which authorized to users [11]. Most of NoSQL databases adopt basic access control mechanisms operating at coarse-grained level [8]. For example, some document-based database grants access control to whole database or none at all. It is still not adequate to provide customized data protection levels, which could increase the usability and expansion of these systems.

Since big data platforms often process user data with personal characteristic, it is significant that data access control being at the finest granularity levels [12]. Fine-grained authorization enables object-level security like field level or row level [13]. The integration of Fine-Grained Access

Control (FGAC) features into data management systems that process sensitive data could benefit it greatly [8].

The FGAC is a fundamental requirement in many applications for an effcient protection of critical data. Most of NoSQL databases adopt basic access control mechanisms operating at coarse-grained level. For example, some document-based database enforced access control at the level of the database. It is still not adequate to provide customized data protection levels, which could increase the usability and expansion of these systems. Only a few NoSQL databases offers a native support for FGAC, such as Accumulo database (key-value database), which enforces access control at the cell level. However, the vast majority of the existing NoSQL databases do not enforce FGAC.

In this work, we review different access control models in NoSQL databases. The rest of the paper is organized as follows: Section 2 presents the different NoSQL database models, section 3 discusses the security issues in NoSQL databases. Section 4 presents the existing access control models in NoSQL databases. Finally, Section 5 concludes the paper.

2. NoSQL Database Models

NoSQL databases categorized into different models. It generally can be classified into the following four groups depending on the data storage model [14]. Figure 1 shows NoSQL categories with examples for each model.

The key-value model is the simplest one among all NoSQL models. It stores data values into a system that can be recalled later using a key (hash) [4]. The key is used as an index similar to the hash table. It is a simple, effcient and powerful model which storing data in a schema-less form.

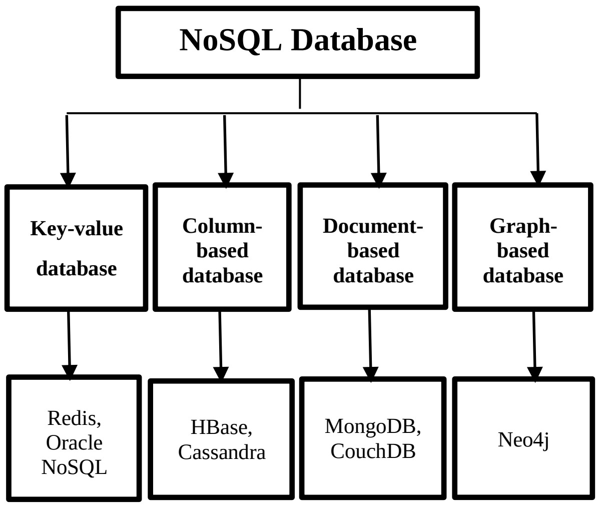

Fig. 1 NoSQL database models with examples.

A. Key-value Database

The key-value database has a very simple application programming interface (API). It is helpful for quickly getting data from the database, process a large amount of data and it uses less storage capacity to store data [15]. It is used in forums and websites for online shopping [5]. Redis, Oracle, and Accumulo are some examples of this model.

B. Column-based Database

The column-based database model stores data in a similar way of the key-value database, but the key is an integration of row, column, and/or time-stamp, which refers to one or many columns (Column Family) [4]. The column family is equivalent to a table in the relational databases.

This model operates well with both complex datasets and a large amount of data in distributed systems [2]. It has a faster query than relational databases. It is easy to add new columns by creating a new file while there is a need to rebuild table in case of relational databases. The model is suitable for an analytic application, data mining and web applications [15]. HBase and Cassandra are examples of the column-based database.

C. Document-based Database

The document-based database model consists of two main elements which are key and document [5]. This database stores data as documents that contain one or more fields. Each document has a unique key that is used to insert, delete and update data in the document. The document-based database supports structured data, semi-structured data (XML files) or unstructured data (text). It also offers high performance and horizontal scalability [4].

The document inside a database is comparable to the record in the relational database and has a dynamic structure that allows modifying, adding or deleting fields. The indexing feature on specified fields allows fast data retrieval. The document-based database is more flexible than the relational database since it is schema-less. It is used for blog software and content management systems [15]. MongoDB and CouchDB are the most popular document-based databases.

D. Graph-based Database

The graph-based model stores data in a graph form that composed of edges and nodes. The node represents an object while the edge is a relationship between objects [15]. The database has schema-less and effciently store data.

The graph-based database is used in many applications like recommendation software, social networking applications, and content management. It is scalable but has complexity [11]. The graph-based database uses shortest path algorithms in order to improve the efficiency of data queries. Neo4j is an example of the graph-based database.

3. NoSQL Database Security Issues

There are many studies discuss and analyze security issues in NoSQL databases. In this section, we focus on some of them.

Zaki [16] discussed the security threats in NoSQL databases. The main threats are about integrity, authentication, injection attacks, consistency, and insider attacks. He showed that these databases do not offer any feature of security in the database itself. These databases provide a very weak security layer. To overcome the security problems of NoSQL databases, Zaki suggested that developers must enforce the security mechanism in the middleware without effect on the scalability features.

Okman *et al.* [10] analyzed the authentication and authorization features of MongoDB and Cassandra. Also, they proposed possible strategies to enhance them. In Cassandra, authorization mechanism is enforced at column family level. In MongoDB, both read-only and read-write permissions set to users in unshared mode. However, there is no support for authorization in shared mode. Both databases provide simple authorization mechanisms. So, improving authorization techniques in these databases is needed.

NoSQL did not support proper authentication and role management when it starts [17]. But now it is possible to manage proper authentication and authorization on the most popular NoSQL databases [9]. Gayatri and Rustom [18] performed a comparison between Role-Based Access Control (RBAC) [19] of most popular cloud and NoSQL databases. They selected MongoDB and Cassandra as NoSQL databases. The MongoDB supports RBAC which performs access to document collections level based on the privileges granted to roles. In Cassandra, the authorization is done at column family level.

Milic *et al.* [20] analyzed the security features like authentication, access control, auditing, data encryption, data access encryption, replication, and integrity in some NoSQL databases which used in web applications. They focused on MongoDB and CouchDB which are document-based databases. These databases fulfill minimum criteria in the security features. In access control, the MongoDB supports RBAC in unsharded mode while no support for RBAC in sharded mode. Also, it supports Mandatory Access Control (MAC) [21], and Task-Based Access Control (TBAC) [22]. The CouchDB only support RBAC.

The main security problem of some NoSQL databases is the lack of access control. Dadapeer *et al.* [4] compared and analyzed security features which include authentication, authorization, data encryption, data access encryption, and auditing in some popular NoSQL databases like Cassandra, MongoDB, CouchDB, Redis, and HBase. They found that authorization techniques vary from one NoSQL database to another. NoSQL databases have ineffcient authorization mechanisms. Most of them implement authorization at a higher level instead of performing authorization at a lower level. More precisely, authorization mechanism is implemented at a database level instead of at the collection level. NoSQL databases suffer from many security issues such as the lack of fine-grained access control, proper authentication, and encryption [9]. Next, we will present some research studies that proposed access control models for different NoSQL databases.

4. Access Control Models in NoSQL Databases

Most NoSQL databases implement access control mechanisms at the coarse-grained level [8]. However, fine-grained access control (FGAC) is a fundamental requirement in many applications. There are research studies that focus on the integration of FGAC into NoSQL databases.

In column-based databases, Kulkarni [23] proposed an access control model that operates at the fine-grained level. The model implemented the access control policies at various levels like a column family, column or row. It is designed to work with Cassandra database and then expanded to operate with HBase. However, the proposed model has dedicated implementation. So, it cannot be adapted easily to other databases.

For document-based databases, many research studies focused on MongoDB which is one of the most popular NoSQL databases. MongoDB supports the RBAC model which implements access control at the collection level. In order to enforce the access control at the document level, the authors [24] proposed the incorporation of a purpose-based model working at the document level into MongoDB. They developed the MongoDB RBAC to support purpose-based policy specification. They refined the granularity level of the MongoDB RBAC model to works at the document level by integrating some purpose related concepts.

In the proposed model, the authors developed an enforcement monitor called Mem (MongoDB enforcement monitor). It was designed to work with any MongoDB deployment. The Mem works as a proxy between MongoDB clients and the server. It observes and may change the flow of messages that are exchanged by the clients. However, the approach is specified to MongoDB and generalizing it to operate with multiple NoSQL databases is required. Also, refining the access control to operate at the filed level is needed.

Many recent applications use the context-related information to support highly specified services. Enhancing NoSQL databases with fine-grained context-aware access control to work at the filed level is required. The authors [25] proposed an access control model work at the field level for the MongoDB database. The proposed model supports both content-based and context-based access control policies.

The authors developed the RBAC model of MongoDB with fine-grained context and content-based policies. An enforcement monitor called ConfinedMem (context aware fine-grained MongoDB enforcement monitor) was designed to implement the model. The monitor designed to integrate into any MongoDB deployment. It is defined as a MongoDB Wire protocol interpreter and acts as a proxy that analyzes and possibly alters messages that exchanged between MongoDB clients and server. However, the enforcement mechanism cannot be implemented in the same efficient way for all query types due to technological restrictions of MongoDB.

To generalize the proposed solutions [24] [25], Colombo and Ferrari [8] discussed issues of integration FGAC into NoSQL databases. They used MongoDB to recognize some methods to define and integrate FGAC into NoSQL databases. The improvement of NoSQL databases to support FGAC needs identifying suitable engineering solutions for encoding of policies, defining an enforcement monitor and integration it into a target NoSQL database.

All previous studies focused on enhancing RBAC in NoSQL databases. Actually, there is no commercial NoSQL database integrates Attribute Based Access Control (ABAC) [26] which supports attributes to define access control rules. The authors [27] proposed an ABAC model for Big Data applications and traditional data management. They claim that the proposed model can be used in a relational database, NoSQL database, and Hadoop. The model based on a query modification method which combines the ABAC mechanism into the source code of user transactions.

On the other hand, the enforcement mechanism specified for SQL queries. The SQL cannot address data variability of non-relational models. There is a need for general ABAC framework to operate with NoSQL databases. To satisfy this requirement, the authors [28] proposed a general approach to enforce fine-grained ABAC into NoSQL databases.

The proposed approach implemented different ABAC policies at field level for documents with a different format without any previous knowledge of the document structures. It based on defining SQL++ query rewriting approach and targets any document database that supports SQL++.

In addition, the ABAC model elastic enough to support the implementation of content-based, context-based and purpose-based policies [28]. Also, it is flexible to support any access control rule that consists of subject attributes, object attributes, and environment attributes at document or field level. However, using a tool for specifying policies and binding is needed to simplify this process. Also, monitors should be implemented to facilitate the integration of the proposed approach with different databases that support SQL++.

In key-value databases, Redis which is one of the most popular NoSQL databases does not provide enough security [16]. Zaki and Indiramma [29] proposed a Redis Client which supports many security services like authentication, authorization, and encryption for different types of data.

The main idea is that a separate Key is created and stored in the database. The value of the key is a data that encrypted using symmetric key by AES algorithm. This data included all other key values being concatenated and encrypted. When a query has created, the system first elicits data entities and then uses the symmetric key to decrypt the data.

The authors also developed a user interface for the system to determine the effciency of the system with the proposed security implementation. On the other hand, using AES improves security but increases the data length. So, extra bandwidth for Redis environment is required. Also, the security extensions in the proposed system will add extra computational overhead, so it should reduce it.

To improve security in graph-based databases, the work [30] proposed a security model that performs access control for NoSQL graph-oriented database. The proposed model uses metadata and provides Data Definition Language (DDL) and Data Manipulation Language (DML) operations. The goal of the model is to allow different applications to implement their own access control when using the graph-oriented database.

The model provides a structure with authorization principles to implement access control. The model was implemented for the Neo4j database and success in preventing unauthorized access. However, the model should be implemented in the core of Neo4j and assessed the performance to evaluate its feasibility. Also, extension the access control to a finer granularity level is required.

In order to examine the feasibility of a granular security on the graph database, the authors [31] used Neo4j which is the most popular graph-based database. They used graph concepts to find a technique that permits access to data while maintaining the security. The method used mathematical formulas to determine two-hop connections that exit from and return to a security layer of the network. These connections can be disclosed to a user without breach the security that specified by security layer.

Table 1 summarizes the previously mentioned access control models for different NoSQL databases. As shown in the table, it is obvious that some popular NoSQL databases do not have fine-grained access control.

5. Conclusion

NoSQL database has many advantages, but it suffers from some security issues like weak authorization mechanisms. The access control feature is the basic unit in data protection of any database management system. In this work, we presented a survey on access control models in NoSQL databases. We gave an overview of NoSQL databases and discussed security issues in this area.

Most of NoSQL databases adopt basic access control mechanisms operating at the coarse-grained level. But this is still not adequate to provide customized data protection levels, which could increase the usability and expansion of these systems. Some research studies presented a new access control model for NoSQL databases which was introduced in this study.

Furthermore, we described access control models in different models of NoSQL databases. As it was mentioned some of the popular NoSQL databases do not have the fine-grained access control. A new access control model is needed which implements an authorization at the fine-grained level and enhances the security of these NoSQL databases.

Table 1. Summary of access control Models in NoSQL Databases.

Proposed Access control	Target Database	Maximum granularity	Access control model
[23]	Column-based database	Filed	-
[24]	Document-based databases (MongoDB)	Document	RBAC
[25]	Document-based databases (MongoDB)	Filed	RBAC
[28]	All databases that support SQL++	Filed	ABAC
[27]	All types	-	ABAC
[30]	Graph-based databases	-	-
[31]	Graph-based databases (Neo4j)	-	-

References

[1] **Gantz, J.** and **Reinsel, D.,** "*Extracting value from chaos*," 2011, IDC IVEW, Available at: http://www.itu.dk/ people/rkva/2011-FallSMA/readings/ ExtractingValuefromChaos.pdf.

[2] **Bhogal, J.** and **Choksi, I.,** "Handling big data using nosql," In: *Advanced Information Networking and Applications Workshops (WAINA), IEEE 29th International Conference*, pp: 393–398, 2015.

[3] **Sahafizadeh, E.** and **Nematbakhsh, M.A.,** A Survey on Security Issues in Big Data and NoSQL[J]. *Advances in Computer Science: An International Journal (ACSIJ)*, 4 (16,): 68-72, Jul. 2015.

[4] **Dadapeer, N.** and **Indravasan, G, A.,** "A survey on security of nosql databases," *International Journal of Innovative Research in Computer and Communication Engineering*, 4(4): 5250–5254, Aug. 2016.

[5] **Priyanka, S.** and **AmitPal,** "A Review of NoSQL Databases, Types and Comparison with Relational Database," *International Journal of Engineering Science and Computing*, 6(5): 4963-4966, 2016.

[6] **Haseeb, A.** and **Pattun, G.,** "A review on nosql: Applications and challenges," *International Journal of Advanced Research in Computer Science*, 8(1), December 2017.

[7] **Noiumkar, P.** and **Chomsiri, T.,** "A comparison the level of security on top 5 open source nosql databases," In: *The 9th International Conference on Information Technology and Applications (ICITA2014)*, 2014.

[8] **Colombo, P.** and **Ferrari, E.,** "Fine-grained access control within nosql document-oriented datastores," *Data Science and Engineering*, 1(3): 127–138, 2016.

[9] **Ron, A., Shulman-Peleg, A.** and **Bronshtein, E.,** "No sql, no injection? examining nosql security," In: *Proceedings of the 9th Workshop on Web 2.0 Security and Privacy (W2SP)*, vol. 1, 2015.

[10] **Okman, L., Gal-Oz, N., Gonen, Y., Gudes, E.** and **Abramov, J.,** "Security issues in nosql databases," In: *IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp: 541–547, 2011.

[11] **Rizvi, S., Mendelzon, A., Sudarshan, S.** and **Roy, P.,** "Extending query rewriting techniques for fine-grained access control," In: *Proceedings of the ACM SIGMOD international conference on Management of data*, pp:

551–562, 2004.

[12] **Colombo, P.** and **Ferrari, E.,** "Access control in the era of big data: State of the art and research directions," In: *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, pp: 185–192, 2018.

[13] **Gupta, N.** and **Agrawal, R.,** "Nosql security," In: *Advances in Computers*, Elsevier, 109: 101–132, 2018.

[14] **Fidels Cybersecurity**, "*Current Data Security Issues of NoSQL Databases*", Jan., 2014.

[15] **Nayak, A., Poriya, A.** and **Poojary, D.,** "Type of nosql databases and its comparison with relational databases," *International Journal of Applied Information Syst*ems, 5 (4): 16–19, 2013 .

[16] **Zaki, K.,** " NoSQL DATABASES: New Millennium Database for Big Data, Big Users, Cloud Computing and Its Security Challenges", *International Journal of Research in Engineering and Technology (IJRET),* 3(3): May 2014.

[17] **Factor, M., Hadas, D., Harnama, A., Har'El, N., Kolodner, E.K., Kurmus, A., Shulman-Peleg, A.** and **Sorniotti, A.,** "Secure logical isolation for multi-tenancy in cloud storage," In: *IEEE 29th Symposium on Mass Storage Systems and Technologies (MSST),* pp: 1–5, Oct. 2013.

[18] **Kapadia, G.S., "Comparative study of role-based access control in cl**oud databases and nosql databases," *International Journal of Advanced Research in Computer Science,* 8(5), 2017.

[19] **Ferraiolo, D.F., Sandhu, R., S., Kuhn, D.R.** and **Chandramouli, R.,** "Proposed Gavrila, NIST standard for role-based access control," *ACM Transactions on Information and System Security (TISSEC),* 4(3): 224–274, November 2001.

[20] **Milic, P., Kuk, K., Trajkovi, S., Ranelovi, D.** and **Popovi, B.,** "*Security analysis of open source databases in web application development*," pp: 310–315, 2016.

[21] **Hu, V.C., Kuhn, D.R., Xie, T.** and **Hwang, J.,** "Model checking for verification of mandatory access control models and properties," *International Journal of Software Engineering and Knowledge Engineering,* 21 (01): 103–127, 2011.

[22] **Deng, J.B.** and **Hong, F.,** "Task-based access control model," *Journal of Software,* 14(1): 76–82, Sep. 2003.

[23] **Kulkarni, D.,** "A fine-grained access control model for key-value systems," In: *Proceedings of the third ACM conference on Data and application security and privacy*, pp: 161–164, 2013.

[24] **Colombo, P.** and **Ferrari, E.,** "Enhancing mongodb with purpose-based access control," *IEEE Transactions on Dependable and Secure Computing*, 14(6): 591–604, May. 2017.

[25] **Colombo, P.** and **Ferrari, E.,** "Towards virtual private nosql datastores," In: *IEEE 32nd International Conference on Data Engineering (ICDE)*, pp: 193–204, 2016.

[26] **Hu, V.C., Kuhn, D.R., Ferraiolo, D.F.** and **Voas, J.,** "Attribute based access control," *Computer*, 48(2): 85–88, 2015.

[27] **Longstaff, J.** and **Noble, J.,** "Attribute based access control for big data applications by query modification," In: *IEEE Second International Conference on Big Data Computing Service and Applications (Big Data Service)*, pp: 58–65, 2016.

[28] **Colombo, P.** and **Ferrari, E.,** "Towards a unifying attribute-based access control approach for nosql datastores," In: *IEEE 33rd International Conference on Data Engineering (ICDE),* pp: 709–720, Jun. 2017.

[29] **Zaki, K.** and **Indiramma, M.**, "A novel redis security extension for nosql database using authentication and encryption," In: *IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT),* pp: 1–6, 2015.

[30] **Morgado, C., Baioco, G.B., Basso, T.** and **Moraes, R.** "A security model for access control in graph-oriented databases," In: *IEEE International Conference on Software Quality, Reliability and Security (QRS),* pp: 135–142, 2018.

[31] **Crawford, B.,** "*Granular security in a graph database*," Tech. rep., Naval Postgraduate School Monterey United States, Jul. 2017.

نموذج التحكم في الوصول لقواعد بيانات NoSQL: لمحة عامة

أشواق العتيبي، و **ريم العتيبي**، و **نيرمين عويس**

كلية الحاسبات وتقنية المعلومات، جامعة الملك عبدالعزيز، جدة، المملكة العربية السعودية

arushdi@kau.edu.sa

المستخلص. أصبحت قواعد البيانات غير الارتباطية الحديثة المعروفة باسم NoSQL أكثر شيوعًا في التعامل مع كمية البيانات الهائلة. انتقلت العديد من المؤسسات من قواعد البيانات الارتباطيه نحو قواعد بيانات NoSQL بسبب تزايد شعبية الحوسبة السحابية والبيانات الضخمة. تم تصميم قاعدة بيانات NoSQL للتعامل مع البيانات غير الهيكلية، مثل المستندات والبريد الإلكتروني ووسائل الإعلام الاجتماعية على نحو فعال. وتستخدم أجهزة موزعة، وتعمل بشكل تعاوني لتخزين واسترجاع البيانات. ونظرًا لأن عددًا كبيرًا من الأشخاص يقومون بتخزين البيانات الحساسة في قواعد بيانات NoSQL، فإن المشاكل الأمنية أصبحت قضايا مهمة. لدى NoSQL العديد من المزايا مثل قابلية التوسع والتوافر، ولكنها تعاني من بعض المشاكل الأمنية مثل ضعف آليات صلاحيات الدخول. ستعرض هذه الورقة النماذج المختلفة لقواعد بيانات NoSQL ومشاكل الأمان المتعلقة بقواعد البيانات هذه. بالإضافة إلى ذلك، نقدم نماذج التحكم في الوصول للبيانات الموجودة في قواعد بيانات NoSQL المختلفة.

الكلمات المفتاحية: التحكم في الوصول, البيانات الكبيرة، قواعد بيانات، قاعدة بياناتNoSQL، نماذج NoSQL، المشاكل الأمنية.