Journal of Information Engineering and Applications
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol.4, No.7, 2014

Modified Query-Roles Based Access Control Model (Q-RBAC) for Interactive Access of Ontology Data

Khamis Abdul-latif Khamis[1*], Zhong Luo[2], H.Z.Song[3]
School of Computer Science and Technology, Wuhan University of Technology, P.R.China
[1]ktifk@hotmail.com, [2]zhongluo@netease.com, [3]juliprn@sina.com

Abstract

The data access model plays an important role during accessing and querying the stored data from the database. It provides an access right and authorization of accessing data into a database. It can distinguish the access boundaries between the administrators and the users where the database administrators can create certain policies either from the client application side or directly from the database side, depending upon the nature of running application. However, the emerging technology on the ontology repository has forced some database developers to adapt most of the access policies from the traditional database system and many of the policies were inherited from the relational database. This method of adopting or borrowing access policies from other storage system has created an unnecessary layer between the ontology repository and database. Most of the emerging ontology repositories lack an independent access model that provides or distinguishes access right between the administrators and users or between the ontology data. This paper proposed the improved access layer from the ontology repository with an additional users' policy creation layer that will lead to increase data security and also increase the performance of querying data. Our effort relies on re-modifying the role based access control model from the traditional one to the new proposed model that organized by the rich users' policies and perfect query rewriting layer. Although it is associated with query module, the proposed model has an additional security layer to restrict unauthorized users from accessing stored data in order to improve querying and data access performance

Keywords: Access methods, Access control, Rule based access control model. Oracle NoSQL database, Virtual data layer, Ontology Query.

1. Introduction

Data access refers to a user's ability to access or retrieve stored data from the database. The user, who gains the access to the database system and the store data, can have the ability to retrieve, move or manipulate data, stored locally or remotely on a wide range of hard drives and external devices (Viljanen, 2012). The ontology-based data access (OBDA) is regarded as key factors of the information system, particularly for the semantic web application and that involves large amount of shared and conceptualized data. It is mainly concerned with querying and data access within and from the data sources in the presence of domain-specific knowledge provided by ontology (Mariano R, 2013). There are many important uses for the OBDA. A vital one is to enrich an incomplete data source with background knowledge, in order to obtain a more complete set of answers to a query. In OBDA, the aim is to use ontology to mediate the access to the data sources which vital point during query answering in the semantic web. OBDA can also be used to enrich the data schema (that is, the relation symbols used in the presentation of the data) with additional symbols to be used in a query (Meghyn B., et al, 2013). Normally, one needs to have an access right from the access layer in order to acquire data access which involves editing, updating and retrieving of stored data. There are several techniques of accessing ontology data, many of them are based on the role based and security control. In this study, we observe two types of ontology based access approaches: (1) Ontology based data access Methods (OBDA) and (2) The rules based access control model (RBAC).

The two proposed access model (OBDA and RBAC) together can be implemented into the key value Oracle NoSQL database (KV Store) and access data are performed in two different approaches. The first approach is applied when the routine access to the stored data performed using Apache JENA APIs under the Java Platform that the application developer uses to allow the application to interact with the NoSQL Database Driver, which communicates with the NoSQL database's Storage Nodes in order to perform whatever application developer requires to access stored data. In addition, administrative access to the store is performed using a command line interface or a browser-based graphical user interface. The system administrators use these interfaces to perform the few administrative actions that are required by Oracle NoSQL Database. In the next section we will discuss the formalization techniques of ontology based data access and then we introduce the modifying process of

Query- roles based data access model. And finally we will outline the proposed rules and procedure for the Query-rules based data access model for the implementation

2. Ontology-based Data Access

OBDA has drawn a great attention from scholars in the fields of ontology engineering, semantic web and knowledge engineering in particular. Ontology constitute the foundation of OBDA approach, which allows users to access data that is spread out over different sources without having to possess specific knowledge on how this data is organized, but instead the user can directly interact with and send queries to retrieve ontology data, that is linked to the real data through appropriate mappings. The OBDA paradigm was formulated a few years ago to tackle the problem of data integration, and more generally that of accessing data sources with a complex structure. The OBDA approach is based on three components:

- The data layer: this is an external and independent data layer, which might consist of a single, possibly federated, database, or by a collection of possibly distributed and heterogeneous data sources (this case is also known as ontology-based data integration). The conceptual model is represented by an ontology, typically formalized in an appropriate description logic, and user requests are expressed as queries over the ontology
- Ontology layer or the conceptual layer: The conceptual model of the application that is used for expressing user requests, The conceptual model is represented by an ontology, typically formalized in an appropriate description logic, and user requests are expressed as queries over the ontology
- The mapping layer: This is a mapping between the two layers, data layer and conceptual layer. The mapping between the conceptual model and the data sources is formalized by mapping assertions, which are based on an appropriate logical language, but which may also incorporate extra-logical features for data manipulation.

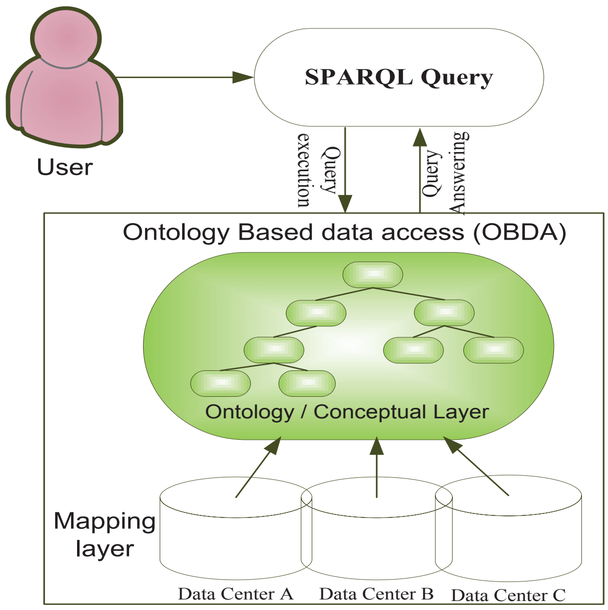

Figure.1 The ontology-based data Access (OBDA) model

The aim of OBDA system is to answer user queries by transforming them into appropriate queries to the data layer, using the ontology and the mapping which provides the users with a conceptual representation of the data contained in them, the possibility of reasoning, and access to the data in terms of this representation. The Figure.1 represents the division of the OBDA layers in the basic architectural model

OBDA-Enabled system which acts as the choral component over query modules, here, the OBDA-Enabled system performs reasoning and query tasks, and the data management work should be done by the DBMS instead of the OBDA - Enabled system itself

OBDA application takes ontology as a unified access entry, and introduces ontology reasoning technology to user query answering in order to deal with more complicated queries. Furthermore, it makes sure that the data used to answer queries are semantically correct. Therefore the current situations over OBDA refer to two fields of information technology, namely data integration and data access.

These fields are mainly concerned with: (1) the services (such as query services and services over consistency checking of ontology data) (2) the type and quantity of data source. From this point of view, let us now overlook the formal approach to represent OBDA based on the user policy creation and security setting layer.

2.1. General definition for OBDA model Framework

It seems reasonable consideration to pay attention to the use of logical implementations, and the development of which resulted in the creation of description logics, in database technologies. Logics of this class became a ground of the key standards of the Semantic Web, as well as of a number of other information technologies, in particular, Therefore ontology based data access OBDA is typically defined as triples

$$\mathbf{\Phi_{RDF-t} = (T, S, M^p)} \tag{1}$$

Where T is representing the intentional level of ontology, and hence our logical model is based on DL, therefore T might also be considered as a family member of DL –T box. S- Is defined as a federated source of database and M^p is a family set of mapping assertion, where each one of the form of:

$$\boldsymbol{\varphi Q(\vec{x}^1) \rightsquigarrow \beta Q(\vec{x}^2)} \tag{2}$$

Where by $\varphi Q(\vec{x}^1)$ is denoted as is a FOL query over the sources of database (S), returning ($\boldsymbol{\Phi_{RDF-t}}$) tuples of values for $\vec{x}^1$ it is also said to be associated query over $\sum r$ which by other modules is denoted as the views associated with mapping M^p.

$\beta Q(\vec{x}^2)$ is a FOL query over conceptual model of T whose free variables are from $\vec{x}^2$and normally denoted as conjunctive query CQ over an alphabet $\sum o$

The mapping assertion map is simply a mapping assertion in which every variable x occurring at $\vec{x}^2$also occurs in $\vec{x}^1$, The intuitive meaning of a mapping assertion M^p of the above form is that all the tuples ($\boldsymbol{\Phi_{RDF-t}}$) satisfying view query (φQ) also satisfies the ontology of conjunctive query CQ which is (βQ). This implies that we are assuming that the source database directly stores the constants denoting the instances of the concepts and the roles in the ontology. More sophisticated approaches to OBDA do not make such a simplified assumption. Rather, they are based on the idea that the objects denoting instances of concepts are not stored in the database, but are constructed through the mappings starting from the values of the data sources. From the above clarification we can now define the OBDA based on its syntax and specification as mention above. If we intend to define OBDA based on the specification let called it ΩB, is simply defined as ΩB is quadruple such that

$$\boldsymbol{\Omega B = (T, S, M^p \text{ and } C_v)} \tag{3}$$

Whereby T is a TBox, deducted as ontology of ΩB, and S is a database schema, called the data source of ΩB. Mp is a set of mapping assertions between S and T , called the mapping of ΩB, and finally, Cv is a set of view inclusion dependencies, called the view inclusions of ΩB, where each view inclusion dependency (or simply view inclusion) is an expression of the form.

$$\mathbf{v_1(i_1 \dots i_k) \subseteq v_2(j_1 \dots j_k)} \tag{4}$$

From Equation 4, the v_1 and v_2 represent a view names, $i_1 \dots i_k$ represent a sequence of pair wise distinct integers ranging from 1 to the arity of v_1,and $j_1 \dots j_k$ sequence of pairwise distinct integers ranging from 1 to the arity of v_2

2.2. OBDA Mapping Layer

In the above section we have already defined the Ontology based on the OBDA system $\boldsymbol{\Phi_{RDF-t} = (T, S, M^p)}$ and therefore the mapping M^p is a crucial component incorporated within the OBDA layers. In general the mapping M^p component of ontology dataset ($\boldsymbol{\Phi_{RDF-DS}}$) encodes how the data in the external source(s) S should be used to populate the elements of T.

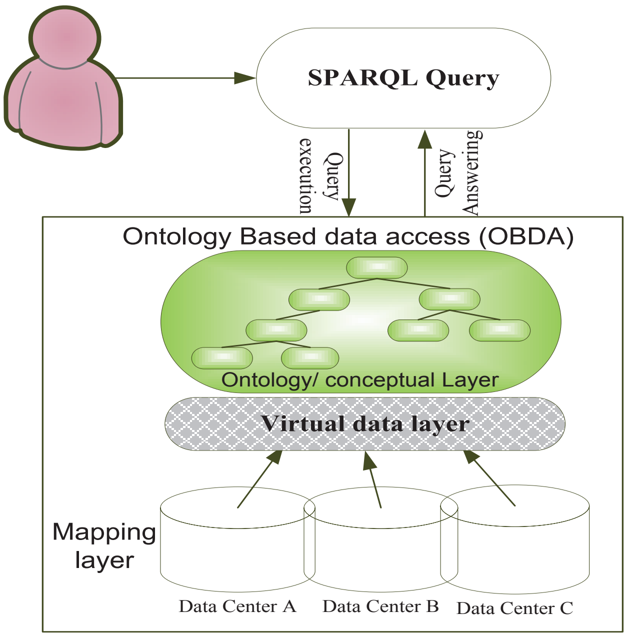

Figure.2 The modified OBDA model with the additional Virtual data layer

It is well understandable that, if we intend to specify OBDA at mapping layer, then it is better to specify only when we are in the presence of a system that includes external sources, mappings and the inclusion of virtual data layer within the OBDA system in such a way that we can increase the performance of data access within the system.

The virtual data layer denoted as v_d is composed with the federal sources S and the mapping data layer $\boldsymbol{M}^p$, and therefore a virtual data layer of $v_d = \boldsymbol{M}^p(S)$: this implies that from this case we can confirm that all the queries are answered based on the (w,r,t) approaches of T and v_d. Note that, we do not really materialize the data of v_d (that's why it is called virtual). Instead, the intentional information on the T and $\boldsymbol{M}^p$ is used to translate queries over T into queries formulated over S.

2.3. OBDA Query Layer

Conjunctive query is a query language used to express queries over DL over a Relational database data wrapping ontologies. We will call them conceptual queries because the variables will range over components (set of tuples) instead of single attributes of the relations in R. The reason for this is that the "ontological" structure is based on components; and for the problems studied in this thesis we don't need a finer query language enabling the retrieval of an attribute not belonging to any component.

The shadow problem that mainly evolves in many systems is that, there is somehow incomplete information for the query answering. The incompleteness of this information during the query execution and query answering are caused by two main factors, these are:

i. The data sources where by naturally considered as incomplete and
ii. The domain constrains encoded in the ontology

In generally Query answering *QA* is set to answer and find out the certain answers cert(φQ, $\boldsymbol{\Phi}_{\boldsymbol{RDF-t}}$) to a query $\varphi Q(\vec{x})$. For example, for those answers that hold in all models of the OBDA system within the Ontology of triple data ($\boldsymbol{\Phi}_{\boldsymbol{RDF-t}}$). From that case we can mention the two concerned condition for the query language to use for querying ontologies

i. Either we should use the ontology language as query language. Where an Ontology language is tailored for capturing in tensional relationships. But the only problems rely on the fact that they are quite poor as query languages.
ii. We should use Full SQL (or equivalently, first-order logic). Because of the stability of the relational database query performance, it is still regarded to be the most perfect query languages for the relational table, but the downfall of it is that in the presence of incomplete information, query answering becomes undecidable and therefore most of the time need logical validity (FOL validity).
iii. We should also use the SPARQL as the most standard query for the triple pattern and Graph data, SPARQL has recently gained the popularity due to its ability to query the unstructured data. But most

of the problems arise when the mindset of the ontology engineering decided to use the persistent storage where it need an additional connectivity, API and Schema and mapping of data, which increase the complexity of retrieving the complete information.

3. Proposed solution

In this section we aim to provide our proposed model based on the above OBDA model and its modified OBDA framework, our proposed model use constructors to create objects of the ontology from RDF tuples in the DB. The constructors are modeled through Skolem functions between query model and mapping data layer:

$$\Phi Mp\colon \varphi Q(\vec{x}^{1}) \rightsquigarrow \beta Q(\overrightarrow{\delta f}.\vec{x}^{2}) \tag{5}$$

Based on the Equation .1, Techniques for partial evaluation of logic programs are adapted for unfolding queries over T, by using ΦM^{p}, into queries over S.

3.1. Conjunctive query (CQs)

From the proposed model of OBDA framework and from the modified architecture, we do propose to use the conjunctive query CQs and the union of conjunctive query CQs (UCQs) as the best solution to handle the incompleteness of information to Access triple data during querying processing and Query answering corresponding to the SPARQL with logical relation algebra

A conjunctive query (CQs): A conjunctive query (CQ) is a first-order query of the form

$$\varphi Q(\vec{x}) \leftarrow \exists \vec{y} R_1(\vec{x},\vec{y}) \wedge \ldots\ldots \wedge R_k(\vec{x},\vec{y}) \tag{6}$$

Where, each $R_i(\vec{x},\vec{y})$ is an atom using (some of) the free variables$(\vec{x})$, the existentially quantified variables $(\vec{y})$, and possibly constants. The modules support a conjunctive queries over an Ontology model, a conjunctive query (CQ) φQ over an of triple set $\Phi_{RDF-t})$ is an expression of the form of

$$\varphi Q(\Phi_{RDF-tx}) \leftarrow \beta(\Phi_{RDF-tx}, \Phi_{RDF-ty}) \tag{7}$$

where Φ_{RDF-tx} is a tuple of distinct variables, called distinguished, Φ_{RDF-ty} is a tuple of distinct variables not occurring in Φ_{RDF-tx}, called non-distinguished, and (Φ_{RDF-tx}; Φ_{RDF-ty}) is a conjunction of atoms with variables in Φ_{RDF-tx} and Φ_{RDF-ty}, whose predicates are atomic concepts and roles of $\Phi_{RDF-t})$. We call φQ (Φ_{RDF-tx}) the head of the query and (Φ_{RDF-tx}; Φ_{RDF-ty}) its body. A union of CQs (UCQs) is a set of CQs (called disjuncts) with the same head. A union of conjunctive queries (UCQs) is an expression of the form

$$\varphi Q\ (\Phi_{RDF-tx}) \leftarrow CQ1\ (\Phi_{RDF-tx}, \Phi_{RDF-ty}1) \cup \ldots \cup CQn\ (\Phi_{RDF-tx}, \Phi_{RDF-ty}n) \tag{8}$$

Where each **CQ**$_1$ (Φ_{RDF-tx}, $\Phi_{RDF-ty}i$) is a conjunction of atoms. Similarly, for a given interpretation *I*, φQ^{I} is the set of RDF tuples Φ_{RDF-t} of domain elements that assigned to Φ_{RDF-tx} make the formula:

$$\exists \Phi_{RDF-ty}1.\ CQ1\ (\Phi_{RDF-tx}, \Phi_{RDF-ty}1) \vee \ldots \vee \exists \Phi_{RDF-ty}n.CQn\ (\Phi_{RDF-tx}, \Phi_{RDF-ty}n) \tag{9}$$

Equation.9 is true in a given interpretation - I.

Note that:

- Conjunctive query CQs contains no disjunction, no negation, no universal quantification.
- Correspond to SPARQL/relational algebra select-project-join (SPJ) queries {the most frequently asked queries.
- They also form the core of SPARQL query modules

3.2. Challenges facing query answering in the OBDA

Various parameters affect the complexity of query answering over an ontology modules ad ontology repository in generally. Depending on which parameters we consider, we get different complexity measures: most of these are caused by the three corresponding issues regarding the Ontology data, storage schema and the combination of both schemata and data complexity

- Data complexity: for this case only the size of the ABox of the triple pattern (i.e., the triple data) whereby the main scenario is that; the TBox under triple data and query are considered as fixed.
- Schema complexity: for the schemata, there is an issue of the size of the TBox (i.e., the schema) whereby the main focus is that the ABox under Ontology model and query are also considered to be fixed.
- Combined complexity: when we combine the two facts together data complexity and schema complexity, then we get no parameter is considered fixed.

In the integration setting of OBDA of a modified framework as shown in the Figure.2, the size of the Virtual data largely dominates the size of the concept layer, meaning that it also dominants the query expression and Query answering.

This gives us another direction to conclude that; the data complexity is the relevant to the complexity measure. In that case it is better now to consider the inference layer within the OBDA framework, let us now define the inference of query answering.

To be able to deal with data efficiencies, we need to separate the contribution of A – as ABox from the contribution of conjunction query φQ and T as TBox.

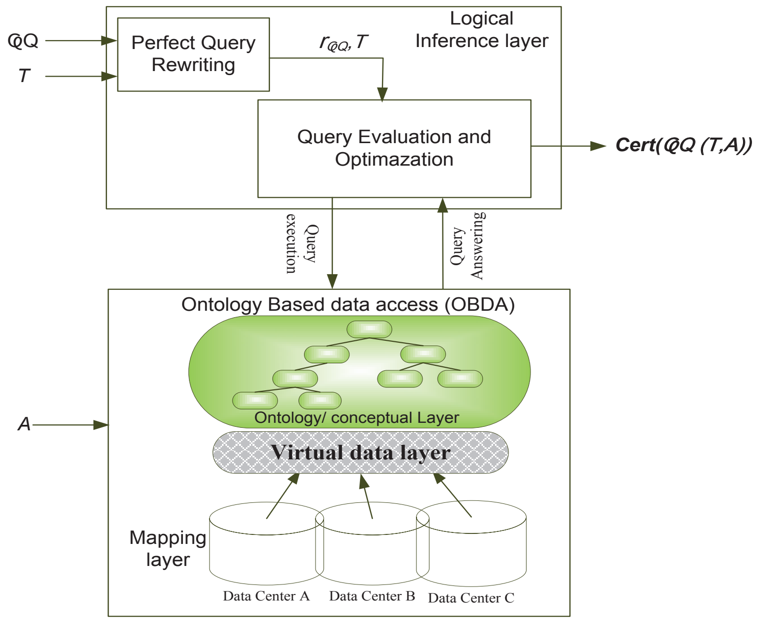

Figure.3 The modified OBDA model with the Virtual data layer and Logical Inference layer for QA

Based on the Figure.3, we have tried to implement the Query answering methods by introducing query rewriting model, From now on the Query answering can now be implemented easily in two phases:

i. The first phase is using the Perfect rewriting approach: this generates a new query $r_{\varphi Q}:T$ from and TBox, Query rewrites approach is a method of rewriting the SPARQL query before query execution in the ontology storage. The query rewrites to allow the information to user by authority. This approach just rewrites queries, so it is easy to adapt to any type of ontology storage which is
ii. able to SPARQL query and
iii. The second phase is to use Query evaluation and optimization approach: this approach evaluate $r_{\varphi Q}:T$ over the ABox A, which is seen as a complete Mapping database whereby at the end of processing and query answering produces $cert(\varphi Q,(T:A))$ as a perfect Query answering without any restriction from the generated new query $(r_{\varphi Q}:T)$

Based on the The new proposed model introduced in Figure.3 above, it seems that OBDA has shown an ability to handle query processing where by the new generated query $(r_{\varphi Q}:T)$ must be perfectly rewritten and also evaluated, likewise to the query answering must also be evaluated and optimized to produce a new generated query answering $[cert\ (\varphi Q,(T\colon A))]$. This will reduce complexity of incomplete information.

4. The Query-Roles Based Access Control Model (Q-RBAC)

Our second approach is to implement the Q-RBAC – the Query-roles based access control within the OBDA model framework. We implement the Q-RBAC policies under the query rewriting and execution, this extension of policies is shadowed under the OBDA framework to increase the scalability under the user side and restrict any fraud confrontation when evolved in order to ensure a rapid access to information to the fact that the performances do not decrease as fast as the amount of data increases. Recalling from the above section we did present the OBDA modification that provides access control and the right, which is in large extents is based on the storage system itself rather than the user side of the system.

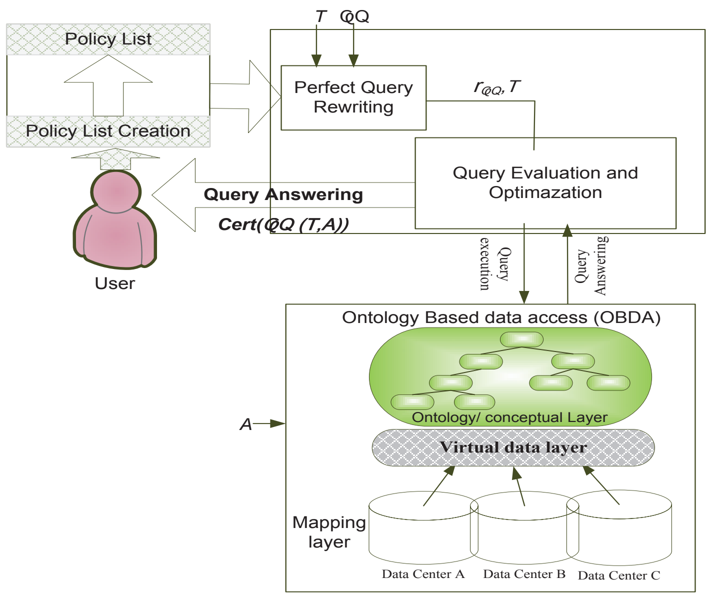

Figure.4 The Modified OBDA Model with the Q-RBAC Users' Policies

Role-based Access Control (RBAC) divides users by roles, match up all objects with operations (access or deny) as permission, and define policies to pair of roles and permissions. Because RBAC identifies users in web ontology and makes available access not only to storage systems, but also information stored, it is a suitable technique to define policy for access control. On the other side of the RBAC the user itself cannot be associated directly with the permission, but can be associated with the roles. In this scenario permissions must be authorized for roles and roles must be associated and authorized to the system user. Figure.4 below depict the inclusion of RBCP for users to get the required access to the Ontology data stored under persistent storage module

4.1. Policy Creation

Based on our study objectives, the modified access model shown in the Figure.4 above provides the access right to the user through querying a triple data from the persistent storage; it has an additional subpart of Access control model based on the User policy setting, which is policy list creation. At a policy list creation has only one main activity, which is to validate policies associated with the roles of the user, whereby In most cases the user has to use the User ID so that can be assigned to the specific policy deposited by the system administrator

The implementation of Q-RBAC into OBDA required high queue of understand the Role of user on Querying or Accessing stored data and also the Policy setting within the storage model. In generally a role is a collection of permissions needed to perform a certain job function while permission, is an access mode that can be exercised on the object; and

- Roles → Permissions: the permission assignment function that assigns to the roles, the permissions needed to complete their jobs;
- Users → Roles: the user assignment function, meaning that assigns users to specific roles;

The goal of our approach is to enforce the privacy requirements by the SPARQL query rewriting. The principle of our approach is to rewrite the initial SPARQL query in order to protect personal data from unauthorized access. That rewriting algorithm is instrumented by a privacy aware model. The decision to use the RBAC or a modified version of Q-RBAC into OBDA is that, we can be assured to prevent global access of information and also to grant the access rights to the information needed for those who need it. This process validates the policies which role of the user is using user ID.

4.2. Roles for Q-RBAC Model

From the proposed storage model point of view, it is necessary to connect storage model approach and Inferencing layer, because at the inference layer we define rules and facts to inference RDF data through concept and relationships. Apart from that we gave an explicit definition for an RDF property by defining and implementing some rules where we use them into reasoning algorithms and in the proposed framework in order to improve comprehensions of query processing. Therefore, through the modified model of Q-RBCA model we also intend to implement two important roles (i) constraint role and (ii) functions, roles that will guide the user through accessing RDF data from the Oracle NoSQL database via SPARQL query.

i. **Constraint role** $< \varphi cr > \varphi cr$ Is constraint rule, applied to the user that has restriction of relationship in maintaining access control of an RDF data from the subjective database; φcr helps to validate the applications and whether the relationship can be established after changes. For example, if a user has established a session and wants to activate a role in the session, we must check whether the user has this role through the rule (I to III). If the user has the role, then the related access control information can be updated in the application.

ii. **Function Role** φfr φfr Is a function rule, describes objects with given conditions based on object context. There are some relationships which cannot be expressed directly in policy creation but they can be found by reasoning with φfr, i.e. when user or system administrator want to browse access control for RDF data information in a database application, then the φfr can help application to provide a convenient view to simplify Query operation.

In the coming session we can deduce the two roles mention from the above section and divide them into five categories, these rules are then implemented in first order logic (FOL) in order to provide a better semantic expression of each rule, the rules are better expressed using logical IF-THEN statement as described below:

Rule I: *<Session Constraint Rule(* δS_{cr}*)>* we use Session constraint rules (δS_{cr}) when a user establishes a session and the session has an activated subset of the set of roles the user is assigned. Such that:

IF $\exists u.s\ \text{establis}(u,s) \land \exists u.r\ \text{hasRole}(u,r)$

THEN

$$\delta S_{cr} \rightarrow \exists s.r\ \text{hasActiveRole}(s,r)$$

Rule II: *<Mutually Exclusive Roles Constraint Rule* δME_{cr}*)>*

IF $\exists u.r.\text{hasRole}(u,r) \land \exists r.\Delta r.\text{hasConflict}(r,\Delta r)$

THEN

$$\delta ME_{cr} \rightarrow \exists u.\Delta r\ \text{hasRole}(u,\Delta r)$$

Rules III*< Prerequisite Roles Constraint Rule* (δPR_{cr})*>*

IF $\exists u.r.\text{hasRole}(u,r) \land \exists r.\Delta r.\text{prerequsite}(r,\Delta r)$

THEN

$$\delta PR_{cr} \rightarrow \exists u.\Delta r\ \text{hasRole}(u,\Delta r)$$

Rule IV *< Indirect Relationship Function Rule* (δIR_{fr})*>*

IF $\exists u.s\ \text{establis}(u,s) \land \exists s.r\ \text{hasActiveRole}(s,r) \land \exists r.p.\text{hasPermission}(r,p)$

THEN

$$\delta IR_{fr} \rightarrow \exists u.p\ \text{hasActivePermision}(u,p)$$

Rule V *< Hierarchies Relationship Function Rule* δHR_{fr}*)>*

IF $\exists r.\Delta r\ \text{inherit}(r,\Delta r) \land \exists r.p.\text{hasPermission}(r,p)$

THEN

$$\delta HR_{fr} \rightarrow \exists \Delta r.p.\text{hasPermission}(\Delta r,p)$$

Note that the variables, r and p represent the session, u – represents a user, r- represents roles and p represent a permission. From rule number I to rule number III give the explicit definitions of constraint rules, and rule number IV to rule number V, are based on the function rules implication.

4.3. Perfect Query rewriting

The process of obtaining a perfect query, rewrites any given SPARQL query to adapt the policy creation list, made by previous process. The perfect query rewriting (PQR) is the first access control process, where the inaccessible information from users is filtered at the query level. The PQR uses all necessary policies' resources to obtain all useful data form ontology repository and any useless ontology data must be filtered before model creation. Therefore the perfect section of Query rewriting (PQR) for policy creation and implementation depend upon consistence of Q-RBAC framework which relies on a series of fundamental steps to ensure that only authorized users are capable of accessing the data they are authorized to access within the database.

In the Figure.5 outlines the algorithms process and workflow for the Q- RBCA model where the Q-RBAC defined a user as anyone who has been authorized to access the database including the database developer and

Administrators (DBD & DBAs). Based on this algorithm, we assume that a user has already been granted access right and therefore has been successfully logged into the NoSQL database, and is currently operating under a user session. Therefore, the Q-RBCA system should have knowledge of which user is attempting to access the requested data and can use that information for decision making.

Based on the given algorithms at the Figure.5, the first step of the process workflow begins once a user submits the SPARQL query statement to the database through query client, the Q-RBCA understand all the rules that the user has been assigned to and can also retrieve all the rules and policies that has been assigned to the Q-RBCA model. The Q-RBAC framework emerges on the use of Q-RBCA's policies to grant users access to the underlying data. When a role has not been assigned to any Q-RBCA's policies, that role has no access rights within the database.

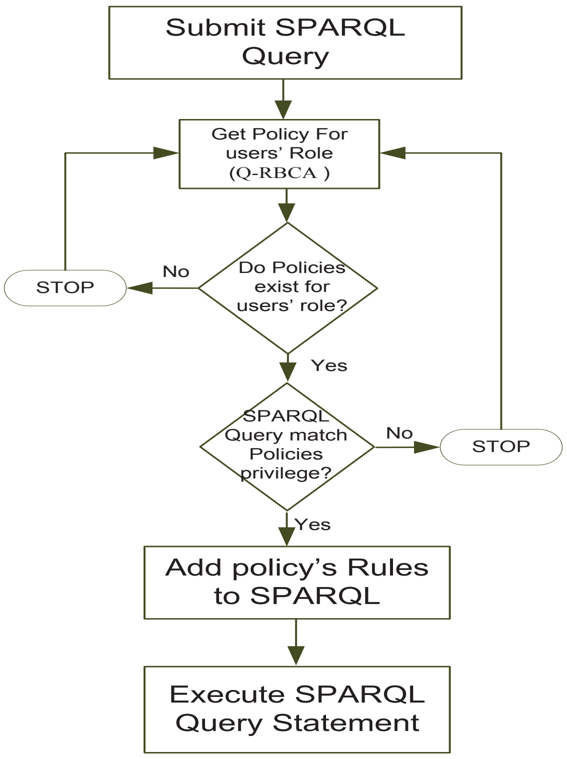
Submit SPARQL Query
Get Policy For users' Role (Q-RBCA)
Do Policies exist for users' role?
No
STOP
Yes
SPARQL Query match Policies privilege?
No
STOP
Yes
Add policy's Rules to SPARQL
Execute SPARQL Query Statement

Figure.5 The PQR process flow for the Q-RBAC model

The implementation of modified Q-RBAC model at the database's level rather than the application level uses a dynamic query rewriting, this framework takes a submitted SPARQL statement and dynamically rewrites it according to security policies and provides new rewritable query ($r_{\varphi Q}:T$). The query rewriter adds rules in the form of WHERE clauses to create a new SPARQL statement, that is then submitted to the NoSQL Database. With this method, security policies are enforced regardless as to whether the database is accessed through an application or directly using a tool.

Therefore, if there are no policies that have been assigned to any of the user's roles, then the Q-RBCA will terminate the process and return the error informing the end user that "No policies exist for this user's role(s)."

4.4. Query Execution

This process creates ontology model by execution rewritten query where the ontology model is created in memory. It is a final step in the Q-RBAC process whereby the transformed SPARQL statement is executed. The result of the transformed SPARQL query is authorized to access only to data that has been explicitly allowed via a policy that the user's role has been assigned. During the SPARQL query execution each policy is interpreted at query-time, which implies that any changes made to the rules of a policy are automatically implemented the next time the policy is used.

Summary

In our proposed solution we have shown the inclusive virtual data layer as a major source of storage performance because it increases the performance of data access, and can provide a unified single repository of federated data from a set of distributed or heterogeneous data sources offering the full power of the relational algebra (conveniently extended to deal also with non-relational data sources) to create, combine, transform and federate data views and deliver the required virtualized data services layers. Meanwhile the proposed modified model uses a connection layer between storage data model and Inferencing layer, where the inference layer provides rules definitions and facts to inference RDF data through concept and relationships. Apart from that we gave the explicit definition for the RDF data stored structure, rules and policies, which are implemented into the reasoning algorithms from the proposed framework in order to improve query performance during data access.

Acknowledgements

This work is supported by "the Fundamental Research Funds for the Central Universities (2014-IV-148)"

References

Wang, Zhen-wu (2010) A generic access control model based on ontology, Wireless Communications, Networking and Information Security (WCNIS), IEEE International Conference, p: 335-339

Mariano Rodrguez-Muro, Roman Kontchakov and Michael Zakharyaschev (2013) Ontology-Based Data Access: Ontop of Databases, Journal The Semantic Web – ISWC 2013, Lecture Notes in Computer Science Volume 8218, p: 558-573

Katal, Avita ; Gupta, Pranjal et al (2013) Authentication and authorization: Domain specific Role Based Access Control using Ontology, 7th International Conference on Intelligent Systems and Control (ISCO), p: 439-444

Meghyn Bienvenu, Balder ten Cate et al (2013) Ontology-based Data Access:A Study through Disjunctive Datalog, CSP, and MMSNP, PODS'13, June 22–27, New York, New York, USA

Viljanen, K.; Tuominen, J. (2012) Normalized Access to Ontology Repositories[C], IEEE Sixth International Conference on Semantic Computing (ICSC), p: 109-116

Cure, O. ; Kerdjoudj, F. ; Chan Le Duc (2012) On the Potential Integration of an Ontology-Based Data Access Approach in NoSQL Stores[C], 3rd International Conference on Emerging Intelligent Data and Web Technologies (EIDWT), p: 166-173

Wei-Tek Tsai ; Qihong Shao (2011) Role-Based Access-Control Using Reference Ontology in Clouds, (ISADS), 10th International Symposium on Autonomous Decentralized Systems Page(s): 121-128

Jain Amit and Farkas Csilla. (2006) Secure resource description framework: an access control model. SACMAT '06: Proc. of the 11th ACM symposium on Access control models and technologies. USA

Yuan, E., and Tong, T. (2005). Attribute Based Access Control (ABAC) for Web Services. Proc. 3rd International Conference on Web Services (ICWS 2005), p. 561-569

Mossgraber, J. ; Rospocher, M. (2012) Ontology Management in a Service-Oriented Architecture: Architecture of a Knowledge Base Access Service, 23rd International Workshop on Database and Expert Systems Applications (DEXA), p: 289-293

Giunchiglia, F. Crispo, B. Rui Zhang (2010), Access control via lightweight ontologies , Fifth IEEE International Conference on Semantic Computing (ICSC)

Funian Tang ; Rongnian Tang(2010) Minimizing influence of ontology evolution in ontology-based data access system, IEEE International Conference on Progress in Informatics and Computing (PIC), Volume:1 p: 10-14