Query Rewriting In Spatial Database For Access Control

Chen Zhen, Chen Rongguo, Xie Jiong, Zhang Mingbo, Cheng Changxiu, Chen Yingdong
Institute of Geographic Sciences and Natural Resources Research
State Key Laboratory of Resources and Environment Information System
11A, Datun Road, Chaoyang District, Beijing, 100101, China
{Chenzhen,Chenrg,Xiej,Zhangmb,Chengcx,Chenyd}@lreis.ac.cn

ABSTRACT
Current database with numerous users requires access control. It is also needed in spatial database systems that keep both spatial attribute and non-spatial attribute safe. The query rewriting technology in spatial database is discussed in this paper. First the operations for merging authorization policies are discussed. By classifying the subjects and permit flag of the authorization policies, two operations, both intersection and union, are discussed with different purpose. And an algorithm for merging user's authorization policies is prompted. Then a view based query rewriting method is set up by replacing all the relations in the query with the corresponding view. That makes sure the fine-grained access control for the spatial database and spatial objects. Finally, a spatial database as an application is given to validate the effectiveness of query rewriting technology in spatial database.

Categories and Subject Descriptors

General Terms
Algorithms, Security

Keywords
Fine-grained, access control, spatial database, query rewriting.

1. INTRODUCTION
Database access control is the foundation of database security in modern database systems. It identifies the users of database sessions to decide what could be retrieved and what should not when users access data. It guarantees the data in database not be modified or deleted by illegal access and make sure the integrity of the database data. Actually, the issue of access control is to determine if user could access the data, what kind of operation user could use and what data could be accessed.

Query rewriting is a common technology in database for SQL optimizing and fine-grained access control. Stonebrake prompted a method for access control by query rewriting and implemented in Ingres database[1].The idea was changing the SQL statement according to the user’s permits and made the result qualified the definition of user permits, which guaranteed the security of the data in database. This technology has been adopted in some commercial databases for fine-grained access control, for example, Oracle VPD(Virtual Private Database)[2] and Hippocratic[3] database.

Spatial database is a kind of special database that could store spatial data (vector and raster) in a relational database. Unlike traditional relation database, spatial object stored in spatial database is massive, spatial computing is time-consuming and cost a lot of time. The most important is the spatial object is topological sensitive and multi-dimension which result in the traditional database access control and query rewriting technology are not suitable for spatial database. Spatial database authorization policies and query rewriting technology should be studied further.

This paper will discuss the query rewriting technology in a spatial database, including the authorization policy merging operations, view-based query rewriting and finally, a framework will be prompted.

2. AUTHORIZATION POLICY MERGE
2.1 Terms Definition
RBAC[4](role based access control) is a wide-used access control model in database. In this paper we use RBAC model as our reference access control model. In traditional RBAC model, an authorization policy is described as \( s,op,o,pt \), where \( s \) is the role in RBAC; \( op \) is the operation type in database and \( o \) is the object could be manipulated. From a more secure view, we extend the traditional authorization policy by adding a permit flag. A permit flag is the status of either grant or deny. Deny status will make sure user never access the data security administer didn’t want to user to access and make more flexibility.

Definition 1: (Spatial authorization policy): \( a=<s,op,o,pt> \) in which \( s \in U \) means subject, in RBAC model presents a role with \( r \); \( op \) means operations in database, for example, insert, update or delete; \( o \) means the objects of the authorization which is a result of a spatial query, for example “the rivers cross Beijing” or “the roads within Zhejiang province”; \( pt \) means permission flag which is either positive or negative.

Example 1: grant user u could select all roads within Beijing: \( <u,select,<roads within Beijing >,+,> \). It is visualized in Figure 1(a), in which the red features are granted roads.

Example 2: deny user u update the rivers cross Hebei Province: \( <u,update,<rivers cross Hebei >,-> \). It is visualized in Figure 1(b), in which the blue features are the rivers forbidden to access.
In this section, the detail algorithms:

3. \( s_2 = r_j \land s_2 = r_j \land \text{UR}(u, r_j) = \text{true} \land \text{UR}(u, r_j) = \text{true} \) in which \( s_j \) and \( s_2 \) are different role and the user own both two roles.

In the above case, user has been assigned more than one role and with different permits. So that the user will inherit all permits from roles which make a union operation to return the max result.

4. \( s_1 = u \land s_2 = r \land \text{UR}(u, r) = \text{true} \lor (s_1 = r \land s_2 = u \land \text{UR}(u, r) = \text{true} \) in which, one is role and the other is user, and the user has been assigned the role.

Between user and role, the intersection operation is used to limit the result and keep the data security.

Also, the permit flag should be considered when merging two authorization policies. When the positive policies are merged, the operation should follow the subject, either intersection or union; while negative policies are merged, union operation should be took in order to avoid any leak of information to keep the data secure.

The subject type and permit flag factors in policies merge could be concluded as table 1.

![Figure 1. Spatial authorization policy](image)

### 2.2 Merge Operations

In the last section, definitions for the authorization policy have been defined. We are going to discuss merge operations taken based on different subject relationship and permit flag of the authorization policy.

In general, when merging the authorization policy, these three principals must be followed:

1. User authorization has higher priority. The authorization directly assigned to user has the higher priority than the authorization get from role. That makes sure the security administrator could assign some exception to the specified user.
2. Role principal. If a user has been assigned more than one role, that makes the user with all the permits roles assigned.
3. Security principal. When authorization is satisfied by both user and role, the operation intersection is taken to avoid the leak of information. Compared to union operation, intersection operation returns the less information.
4. Deny priority. When grant and deny authorizations are conflict, deny authorization have the higher priority.

According to the subjects participate the policy merge operation, there will be three categories:

Let \( <s_1, op, o_1, pt_1> \) and \( <s_2, op, o_2, pt_2> \) are the two authorization policies will be merged.

1. \( s_1 = s_2 = u \) in which \( s_1 \) and \( s_2 \) are the same user;
2. \( s_1 = s_2 = r \land \text{UR}(u, r) = \text{true} \), in which \( s_1 \) and \( s_2 \) are the same role. User has been assigned with both \( s_1 \) and \( s_2 \);

In the two cases above, the subjects of the authorizations are the same user or roles with different permits, which is called multi-authorization. The operation to handle these kind of authorization policies is intersection which is of two reasons: (1) accomplish with the previous work[5] and (2) enforce security. Union operation will provide more accessibility with the more authorization policy while the intersection operation is on the contrast.

<table>
<thead>
<tr>
<th>Subject Type</th>
<th>Permit Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u - u )</td>
<td>○</td>
</tr>
<tr>
<td>( u - r )</td>
<td>○</td>
</tr>
<tr>
<td>( r - r )</td>
<td>○</td>
</tr>
</tbody>
</table>

### 2.3 Policy Merge Algorithm

The operations for merge two authorization policies have been discussed in the last section. In this section, the detail algorithms for policy merge will be discussed.

### Definition 4: (authorization policy set \( \rho \))

\( \rho \) means the result of two authorization policy merge, which could be expressed as \( \rho = \langle s, op, o, pt \rangle \), in which \( s \) is the subject for the policy set, \( op \) is the operation and \( o \) means the grant and deny objects.

### Definition 5: (intersection operation)

Intersection operation merges two authorization policies and produces an authorization policy set. It decides how two authorization policies could be merged with intersection operation. Figure 2 is the algorithm of intersection operation.

### Definition 6: (union operation)

Similarly, union operation is defined. (Because the algorithm is similar to intersection operation, it is omitted in this paper).

### Definition 7: (user authorization policy)

If the user is controlled by multi-authorization policies, according to the merging principals and operations, the merge steps will be (1) merge all authorization policies directly assigned to user with intersection operation and get its policy set \( \rho_u \); (2) merge all authorization policies assigned to the roles which belongs to the user with union operation and get its policy set \( \rho_r \); (3) and finally merge \( \rho_u \) and \( \rho_r \) with union operation. This could be expressed as following algorithm (As Figure 3)
relation. For any relation \( R_i \in R \), its corresponding authorization policy set \( \rho_i \) will be calculated after merging the authorization policies of the specified user. The view could be like:

Create View \( V \) as
Select * from \( R \)
Where (tid in \( \rho_i \cdot o \cdot tid \)) and (tid not in \( \rho_i \cdot o \cdot tid \))

In which tid is the unique identify of a spatial object that could identify the spatial object internally in spatial database.

2. Rewriting the SQL parser tree. Replacing the relation with the temporal authorization view.

Figure 4 shows the algorithm of query rewriting.

```
QueryModify(u,op,A,Q)
Input: u // user
      op // operation
      A // database authorization policies
      Q // original query
Output: Q' // modified query
R = GetRelations(Q); // get all related relations
Q' = Q
foreach R_i in R {
    if (a in R_i) {
        AddPolicy_Intersection(a,\rho_i);
    }
    else {
        AddPolicy_Union(a,\rho_i);
    }
} // merge all authorization policies
```

Figure 4. Algorithm of view based query rewriting.

When generating the temporal view, any modifications to the relations should be avoid and keep the consistency of the spatial data. Via query rewriting, relations are replaced with authorization views that add extra where clause to filter the features granted or denied. The process of query rewriting is transparent to the user and is triggered automatically by the system that the user has been granted and abandons the un-granted.

3.2 Reference Framework

The modification of the query is made before the execution of the query. Figure 5 shows a reference framework for query rewriting in spatial database.

As soon as the query is committed, the query parser will parse the query into a form of binary tree which is called query tree. All attributes and relations in the query will be represented as a node of tree. Each relation is passed to authorization view builder that builds temporal authorization view according to the authorization policies of the request user. Then the relation is replaced with the view so that the query tree is modified which guarantees the
The query rewriting for spatial access control in spatial database is discussed in this paper, which limits the spatial objects returned to request user. The authorization policies are merged before query rewriting to make it simple for making query rewriting. By replacing the relations in query with temporal views build with authorization policies, the access control is implemented. That makes the query return the data the user assigned to access and skips the forbidden data. The application improves the query rewriting technology is useful in spatial database and make sense to the similar work.

4. APPLICATION

BeyonDB is a spatial database that could store, retrieve and manipulate spatial data, include vector and raster data type, with standard SQL language. It is compatible with SQL/MM[8] and OGC Simple Feature[9]. Also the spatial authorization for access control is implemented in this spatial database. Security administrator could grant or deny user to access the spatial data of certain attribute or certain spatial characteristics with a visual authorization tool in an integrated management toolset named VDBA.

Figure 6 shows the database security administrator forbidden any user to access the data in the restrict area which is red in the figure 6. Any spatial objects within this area will be invisible to any user. This is a typical spatial authorization.

Figure 6. Spatial authorization in BeyonDB

Once a user logs into database, the roles of the user are determined. As a result, the operations and permits of the user are determined. When the user tries to operate the spatial data in the database by posting SQL query, the following steps will be taken place: first, the permit to the target relation and specified operation is checked to determine if the user could access the data. If not granted, an error returns; then all authorization policies of the user will be merged as an policy set according to the algorithm discussed in section 2.3.; A query rewriting process discussed in section 3 will be triggered automatically by building the temporal views and replacing the relations with temporal views which produced the modified query; finally, the modified query is executed instead the original one that will return less information if access control policies are applied.

Figure 7 shows the result of query rewriting. After rewriting the query, the denied area will not be visited by the specified user which leaves blank in Figure 7.

5. CONCLUSION

The query rewriting for spatial access control in spatial database is discussed in this paper, which limits the spatial objects returned to request user. The authorization policies are merged before query rewriting to make it simple for making query rewriting. By replacing the relations in query with temporal views build with

6. ACKNOWLEDGMENTS

The work is supported by the National High-Tech. R&D Program of the Ministry of Science and Technology of the People’s Republic of China (2007AA120400), by the National Key Technologies R&D Program of the Ministry of Science and Technology of the People’s Republic of China (2007BAH16B03), by National Natural Science Foundation of China under Grant No. 40801162.

7. REFERENCES


