Modeling Access Rights Using the CRUD Security Cube: A Database Proof-of-Concept Example

Michael R. Collins, High Point University mcollins@highpoint.edu

Dale L. Lunsford, University of Southern Mississippi dlunsford@cableone.net

Modeling Access Rights Using the CRUD Security Cube: A Database Proof-of-Concept Example

INTRODUCTION

Defining access rights is a challenge in many settings. Since a database often serves as the foundation for information systems, proper specifications at the database level can ensure that proper access rights exist within the system. How do organization set and maintain user and group access rights to information systems in general and within databases specifically? Turnover, promotions, job and task shifts are just a few of the situations that arise in maintaining an up-to-date set of security and access rights for users and groups within organizations today. This paper describes a database implementation of access rights using the CRUD Security Cube (Lunsford & Collins, 2008).

Access Rights

Although the nature of an access right varies from system to system, most contemporary systems provide some mechanism for managing access to resources. Access rights, also known as permissions or privileges, define the types of access that a user or group has to a securable object. In many systems, access rights apply to either users or groups. In Unix systems, access rights apply to an object's owner, a group, and the world (December, 2008). In Windows systems using the NT File System (NTFS), access rights apply to users and groups (Melber, 2006). The targets resources for access rights include directories and files, devices, executables, as well as other objects (Changing Access Security on Securable Objects, 2008). Common access types include full

control, modify, read & execute, read, and write under NTFS (Melber, 2006; Eckel, 2007) and read, write, and execute under Unix (December, 2008). NTFS offers advanced mechanisms for access rights, including inheritance and the ability to deny access (Melber, 2006; Mullins, 2006; Eckel, 2007). Additionally, under NTFS the specification of access rights is either explicit or inherited. Finally, NTFS provides the ability to deny a user or group any particular access type.

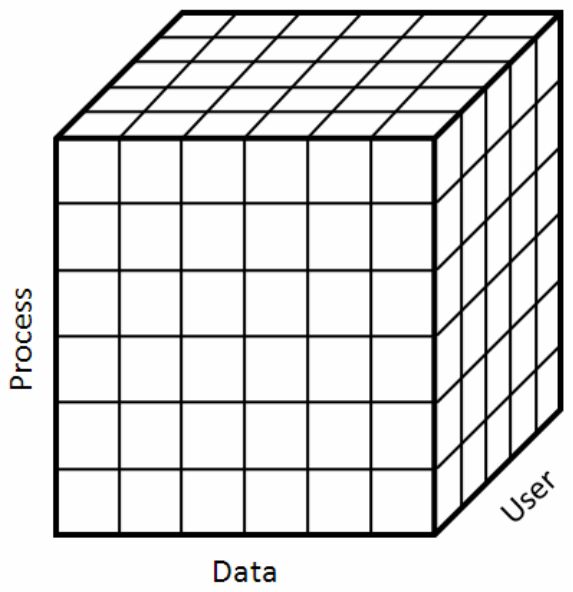
THE CRUD SECURITY CUBE

The traditional CRUD matrix provides a method for identifying the types of access system processes have to data objects. The CRUD Security Cube adds a user/group dimension to the CRUD matrix (Lunsford & Collins, 2008). This dimension documents the access rights for users or groups to processes and data. Analysts may use the CRUD Security Cube to specify security for information systems, including any setting where the user employs specific programs to access data objects.

A Database Example

The CRUD matrix assists database administrators in mapping out usage access for databases within an organization. Working from CRUD Security Cube extension, this paper develops a database proof-of-case by working through a simplified database example. First we examine the data model used for the case as well as the simplified MS Access database used in the paper. Next, we examine the use of the CRUD Security Cube in our specific example. Lastly, we detail how one can use information from the database to update a database's access rights and privileges automatically using code with Structured Query Language (SQL) calls embedded within the code.

FIGURE 1: CRUD SECURITY CUBE



Implementation

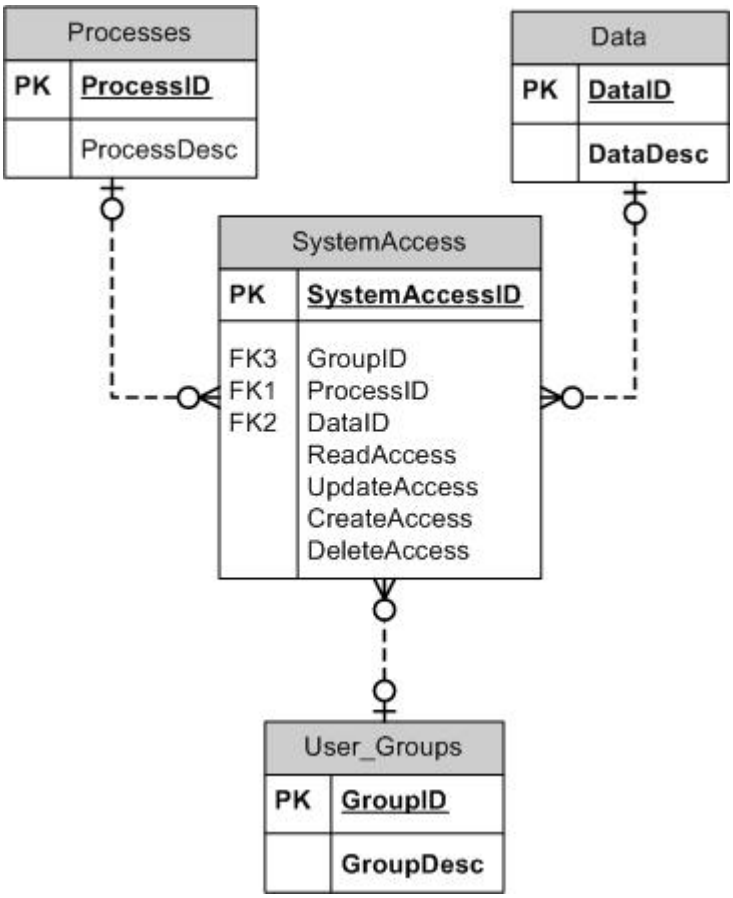
Figure 2 depicts the process employed to implement a system to establish access rights automatically based on a CRUD Security Cube.

FIGURE 2: PROCESS OVERVIEW

The Case Design

Essentially a database model storing data, processes, and group information is developed. Figure 3 illustrates the entity relationship model for our simplified example.

FIGURE 3: CRUD SECURITY CUBE DATA MODEL



As you can see from the ER Model, the System Access Interaction table would contain the information with respect to creating, reading, updating, and deleting of

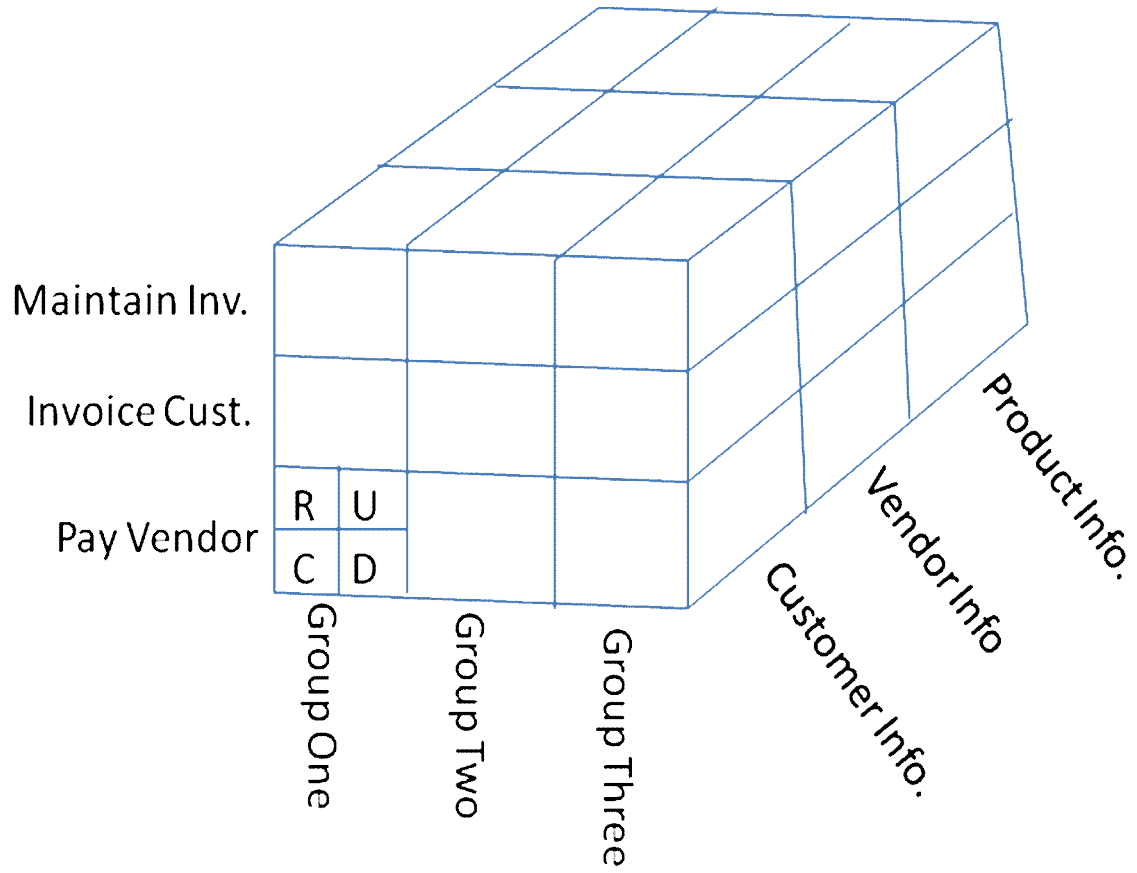
information with respect to specific processes. The information modeled in the ER model is then used to generate an MS Access database filled with information for this example. Figure 4 depicts the groups, processes, and data objects employed in this example.

FIGURE 4: GROUPS, PROCESSES, AND DATA OBJECTS

Groups	Processes	Data
Group One	Maintain Inventory	Customer Information
Group Two	Invoice Customer	Vendor Information
Group Three	Pay Vendor	Product Information

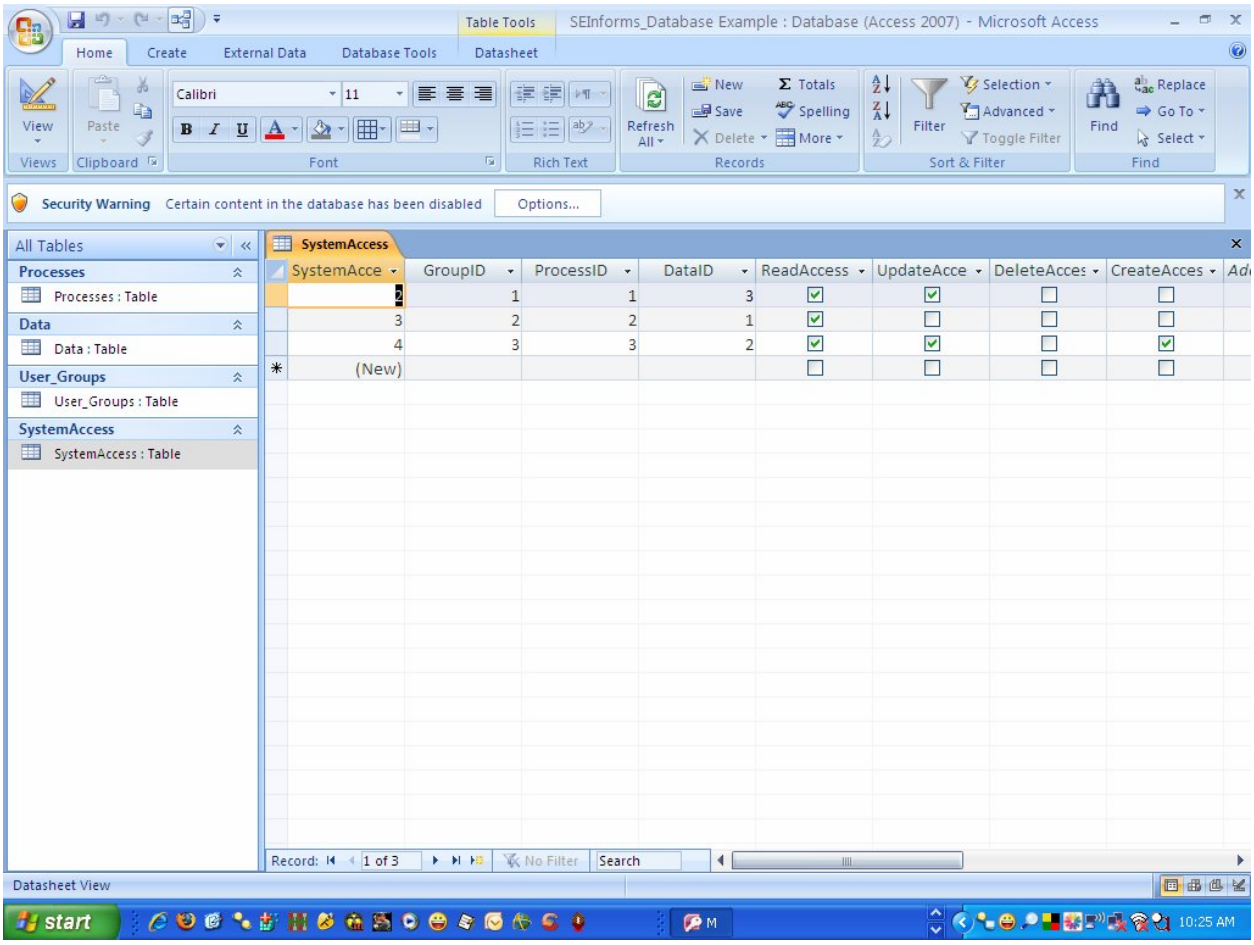
Figure 5 depicts the CRUD Security Cube representation of this data.

FIGURE 5: SAMPLE DATA IN A CRUD SECURITY CUBE REPRESENTATION



As you can see from the cube representation above, the cube allows a database administrator to break down individual access rights by group, within a process, for specific data. This information can then be entered into a database and updated as needed. Once the database is updated with the information a program can be written to pull the data and settings from the database and update the security and access rights for groups and users automatically. A snapshot of the system access table would look similar to Figure 5.

FIGURE 5: MICROSOFT ACCESS IMPLEMENTATION



Using this system access table presented in figure 5, the groups or users documented access and security privileges could be extracted and updated in a separate database using Oracle, SQLSever, MySQL, or just about any other SQL-based DBMS on the market today.

Extensions to this research

Extensions to this research could include additional proof-of-case scenarios that show the versatility of this approach to apply to any type of information system access rights' settings. In this paper we have shown a proof-of-concept example of how the CRUD security cube could be implemented within a database management systems environment. The approach proposed in this paper could be used to automate the setting of security and accessibility settings for objects with respect to data within individual processes and with respect to groups or individuals of an organization.

WORKS CITED

[1] *Changing Access Security on Securable Objects.* (2008, February 14). Retrieved February 26, 2008, from MSDN: http://msdn2.microsoft.com/en-us/library/aa384905(VS.85).aspx

[2] December, J. (2008, January 21). *Permissions.* Retrieved February 22, 2008, from December.com: http://www.december.com/unix/tutor/permissions.html

[3] Eckel, E. (2007, January 22). *How do I... Secure Windows XP NTFS files and shares?* Retrieved February 7, 2008, from TechRepublic.com: http://articles.techrepublic.com.com/5100-10877_11-6152061.html

[4] Lunsford, D. L., & Collins, M. R. (2008). The CRUD Security Matrix: A Technique for Documenting Access Rights. *7th Annual Security Conference.* Las Vegas, NV.

[5] Melber, D. (2006, May 3). *Understanding Windows NTFS Permissions.* Retrieved January 25, 2008, from WindowsSecurity.com: http://www.windowsecurity.com/articles/Understanding-Windows-NTFS-Permissions.html

[6] Mullins, M. (2006, June 15). *Windows 101: Know the basics about NTFS permissions.* Retrieved June 19, 2006, from TechRepublic.com: http://techrepublic.com.com/5102-1009-6084446.html