CSE 5330 **Fall 2020**

Project #2 Description

In this project, you will design and implement a database for keeping track of information about a DBMS subsystem for managing discretionary access control. You will first design an EER schema diagram for this database application. Then, you will map the EER schema into a relational database schema and implement it on ORACLE or on MySQL (you can also use some other DBMS if you have your own version, as long as you get *prior approval*). Finally, you will load some data into your database, and create some queries and update transactions.

Assume that the following requirements were collected for this application:

1. The database keeps track of **USER_ACCOUNTs**. Each USER_ACCOUNT has a unique IdNo (assume this is a unique integer generated by the system for each new USER_ACCOUNT, such as 1, 2, 3, …), a Name (assume this is string consisting of an single initial and last name only for simplicity, such as "J.Smith" or "R.Wong"), and a Phone (a string of 12 characters such as "817-272-3000").
2. The database keeps track of **PRIVILEGEs** (SELECT, INSERT, DELETE, UPDATE, CREATETAB, etc.). The system should allow addition of new privileges as needed. There are two types of privileges: **ACCOUNT_PRIVILEGEs** and **RELATION_PRIVILEGEs**: each privilege belongs to only one of the two types.
3. In addition to **USER_ACCOUNTS**, the system will keep track of **USER_ROLEs**. The roles are different for each database application, and new roles can be added at any time. Each USER_ROLE has a unique **RoleName**, plus possibly other attributes, such as Description.
4. The database keeps track of **TABLEs** (these are the tables for a particular database schema). Each table has a unique TableName, and is related to a single USER_ACCOUNT who is the owner of the TABLE.
5. The database will keep track of the current (active) RELATIONSHIPS as follows:
   - The binary relationship between USER_ACCOUNTs and USER_ROLEs (assume that each USER_ACCOUNT is related to exactly one role).
   - The binary relationship between USER_ROLEs and ACCOUNT_PRIVILEGES.
   - The ternary relationship between USER_ROLEs, RELATION_PRIVILEGEs, and TABLEs.
6. It is assumed that if a role is related to a particular privilege, then all user accounts related to that role will have that privilege

PART 1: You will first design an EER schema diagram based upon the **SECURITY** database requirements specified above and create an EER schema diagram and documentation report describing your design choices. As part of this assignment, you should identify any missing or incomplete requirements, and explicitly state them in your documentation. For such missing or incomplete requirements, make some reasonable assumption and state it clearly in your documentation. You should also explicitly state any additional assumptions you made that were not part of the requirements listed above.

PART 2: The second part of the assignment will be to map the EER schema design to a relational database schema, and create the tables corresponding to the relational schema using the ORACLE DBMS (or MySQL). You will add to your report a listing of the CREATE TABLE statements for the SECURITY database. Specify as many constraints (key, referential integrity) as you can in the relational schema. You should state the choices you made during the EER-to-relational mapping, and the reasons for your choices. (Important: You are allowed to change the EER design during this phase if needed, but you must clearly document all the changes, and the reason for each change).

PART 3: The third part of the project is to load some data into the database, and apply certain update transactions and retrieval queries. You will create your own data. Include at **least 3 roles,**

10 user accounts, and 10 tables (you should choose table names that fit a particular database application, or you can use generic table names such as TAB1, TAB2, …). Assign each user account to one role. Use (as a minimum) the privileges that are described in the book/class lectures. Each table should have one owner.

The following are the tasks for the third part of the project:

1. Load some initial data (as discussed above) into the database tables that you created in Part 2 of the assignment. You can either write a loading program, or use SQL/PLUS (insert command), or use SQL/FORMS. Your data should be kept in files so that it can easily be reloaded during debugging. The data format should be designed by you. (Note: You can also use the transactions created by you in item 3 below to load some of the data).
2. Write queries to retrieve and print all the data you entered. Try to print the data so that it is easy to understand (for example, print appropriate headings, such as: UserAccounts, RelationPrivileges, Tables, Roles, etc).
3. Write the following database update transactions using either JAVA/JDBC or PHP or some other programming language or scripting language.
   - 3.1 The first transaction is to add all the information about a new `USER_ACCOUNT`.
   - 3.2 The second transaction is to add all the information about a new `ROLE`.
   - 3.3 The third transaction is to add all the information about a new `TABLE`. This should include specifying the owner user account of the table.
   - 3.4 The fourth transaction is to insert a new PRIVILEGE, including the privilege type.
   - 3.5 The fifth transaction is to relate a USER_ACCOUNT to a ROLE.
   - 3.6 The sixth transaction is to relate an ACCOUNT_PRIVILEGE to a ROLE.
   - 3.7 The seventh transaction is to relate a RELATION_PRIVILEGE, ROLE, and TABLE.
   - 3.8 Write queries to retrieve all privileges associate with a particular ROLE, and all privileges associated with a particular USER_ACCOUNT. Also, write queries that check whether a particular privilege is currently available (granted) to a particular user account.
4. Each transaction should have a user friendly interface to enter the information needed by the transaction. This can either be a Web-based interface, a command line interface, or a forms interface.
5. Test your transactions by adding a few new entities and relationship instances.

Due Dates:

1. (Parts 1and 2 Due Date: **Friday Nov 13)**. For **Part 1**, this should include the EER diagrams for your designs as well as documentation describing any assumptions you made, and the reasons for your design choices. Draw the EER diagrams using the notation in the textbook. You can also use UML class diagrams notation. You can use any drawing tool for drawing your diagrams. Describe clearly all the assumptions that you made that were not part of the given requirements. For **Part 2**, what you turn in should include your relational schema diagram design and your CREATE table statements, and documentation describing your EER-to-relational mapping choices. If the initial EER design from Part 1 was changed, you should describe the changes and the reasons that led to these changes.
2. (Part 3 Due Date: **Sunday Nov 22)**. This will include a demo demonstrating that your implementation works, as well as demonstrating your transactions to the grader. Source code of all your transactions should be submitted, as well as the data files. A demo schedule will be determined before the due date.

Late penalty: -5% per day late.

Project teams: Each project can be done in a team of 2 persons, or individually.