School of Computing and Information Technology
University of Wollongong

Session: **Autumn 2017**
Lecturer: **Janusz R. Getta**
Tianbing Xia

CSIT115/CSIT815 Data Management and Security
Assignment 4

24 May 2017

Scope

This assignment consists of the tasks related to implementation of discretionary access control, granting system resources, verification of complex consistency constraints, using backup and restore features of DBMS to find the differences between two states of a relational table, and implementation of a simple auditing system.

The outcomes of the assignment are due by **Saturday, 3 June, 2017, 7.00 pm sharp.**

This assignment contributes to 6% (5% for CSIT815) of the total evaluation in the subject.

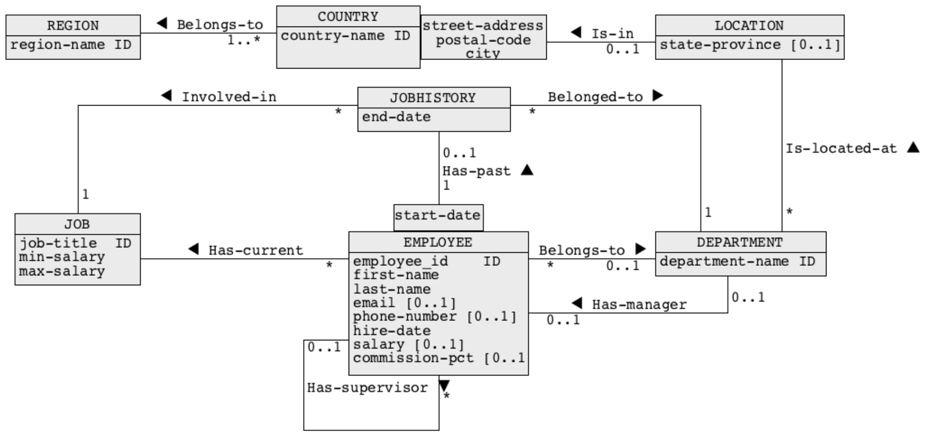
A submission procedure is explained at the end of assignment specification.

This assignment consists of 4 tasks and specification of each task starts from a new page.

Prologue

Download and unzip a file `assignment4-all-files.zip`. You should get the files `Assignment4.pdf`, `dbcreate.sql`, `dbload.sql`, `dbdrop.sql`, `dbload.sql`, `dbchange2.sql` and `dbchange4.sql`. Copy the files to your USB drive such that you can access both files either through command line interface mysql or graphical user interface MySQL Workbench.

The script files create and load data into a database that contains information about a company and its employees. The company consists of several departments located in the cities all over the world. The database also contains information about the present and past jobs of its employees and about the present managerial structure. A conceptual schema of the database is given below.



<u>Tasks</u>

Task 1 (1 mark)

(1) Connect to MySQL as a user `root` either through command line interface mysql or graphical user interface MySQL Workbench and create a new database with a name the same as a *prefix of your University email account*.

(2) While connected as a user `root` use SQL script `dbcreate.sql` to create the relational tables in a database created in the previous step. A script `dbcreate.sql` creates the relational tables that can be used to store information about the region, country, location, department, job, employee, and job history. Execute a script `dbload.sql` to load data into the relational tables that created by `dbcreate.sql`. You can use a script `dbdrop.sql` to drop the relational tables. Do not drop the relational tables now!

DO NOT submit any report from processing of the actions listed above.

Implement SQL script `solution1.sql` that performs the following actions as a user `root`.

(1) First, the script creates a new user with a name the same as *a prefix of your University email account*. A password of the new user is up to you.

(2) Next, the script grants to the new user to read data from the relational tables `EMPLOYEE` and `JOBHISTORY`, and to create the views located in a database with the same name as *a prefix of your University email account*. The privileges must be granted such that the new user is able to grant all privileges listed above to the other users.

(3) Next, the script sets the following values of resource limits to a name of account owner is the same as *a prefix of your University email account*: total number of queries an account owner can issue per hour must be set to `100,` and total number of simultaneous connections to the server by an account owner must be set to `3`, total number of times an account can connect to the server per hour must be `10`.

(4) Next, the script locks an account of the new user with the same name as *the prefix of your University email account*.

(5) Finally, the script lists the privileges granted to the new user, the values of resource limits set in a step (3) and a status of the account *a prefix of your University email account* set in a step (4). To do so your script must access appropriate relational tables in a database `mysql`. Do not list information not related to the actions performed above !

Deliverables
Submit a file `solution1.rpt` with a report from processing of SQL script `solution1.sql`. The report MUST have no errors and the report MUST list all SQL statements processed.

A name of the database and a names of new users created in Task 1 must be the same as *a prefix of your University account.* The names different from *above names* means that your work has been done by another student with all consequences implied by such fact.

A report that contains no listing of executed SQL statements scores no marks !

A report that contains any processing errors scores no marks !

Processing of the script that contains SQL statements that do not implement the subtasks listed above scores no marks !

Submission of a file with a different name and/or different extension and/or different type scores no marks !

Task 2 (1.5 marks)

(1) Connect as a user `csit115` to MySQL either through command line interface `mysql` or graphical user interface MySQL Workbench and execute a script file `dbdrop.sql`, then immediately after that execute script `dbcreate.sql` and `dbload.sql` to refresh a sample database. Exit command line interface `mysql` or graphical user interface MySQL Workbench

(2) Create a logical backup of a relational table `EMPLOYEE` and save it in a file with the same name as *a prefix of your University email account.*`bak`.

(3) Connect as a user `csit115` to MySQL either through command line interface `mysql` or graphical user interface MySQL Workbench and execute a script file `dbchange2.sql`.

(4) Create an empty relational table with the same name as *a prefix of your University email account* concatenated with `_EMP` with the same structure as a relational table `EMPLOYEE`. Remember, to create appropriate consistency constraints for a new table.

(5) Use a text editor and modify a backup file obtained in a step (2) such that a backup of a relational table `EMPLOYEE` can be restored into a relational table with the same name as *a prefix of your University email account_EMP*.

(6) Use an updated backup in file *a prefix of your University email account*.`bak` to load the contents of the backup into a relational table *prefix of your University email account_EMP*. DO NOT delete the backup file!

No report is expected from the implementation of the steps listed above.

Implement SQL script `solution2.sql` that finds the differences between the contents of a relational table `EMPLOYEE` and a relational table with the same name as *a prefix of your University email account_EMP,* The script must first list the rows added to the relational table `EMPLOYEE` after the backup file was created, then the rows deleted from a relational tables `EMPLOYEE` after the backup file was created, and finally list the rows changed in relational table `EMPLOYEE` after the backup file was created. In brief, the script must first list all added rows, then all deleted rows, and finally all changed rows in a relational table `EMPLOYEE`. It is allowed to use more than one `SELECT` statement to implement this task.

Deliverables

Submit a file `solution2.rpt` with a report from processing of SQL script `solution2.sql` and the updated backup file in a step (5). A report included in a file `solution2.rpt` MUST have no errors and it MUST list all SQL statements processed.

A submission updated backup file that does not contain a logical backup of relational table *a prefix of your University email account*`_EMP` scores no marks.

A prefix of a name of backup file created in Task 2 must be the same as *a prefix of your University account*. The name different from *a prefix of your University account* means your work has been done by another student with all consequences implied by such fact.

A report that contains no listing of executed SQL statements scores no marks !

A report that contains processing errors scores no marks !

Processing of the script on an empty database scores no marks !

Processing of the script that contains statements other from `SELECT`, `source`, and `notee` scores no marks !

Submission of a file with a different name and/or different extension and/or different type scores no marks !

Task 3 (2 marks)
Connect to MySQL either through command line interface mysql or graphical user interface MySQL Workbench as `csit115` user and execute a script file `dbdrop.sql` and immediately after that the scripts `dbcreate.sql` and `dbload.sql` to refresh the contents of a database `csit115`. No report from refreshing of a database `csit115` is expected.

Implement SQL script `solution3.sql` that performs the following actions.

(1) First, the script modifies a description of one department in a relational table `DEPARTMENT` such that a new manager of the department is a member of a department different from the managed department.

(2) Next, the script finds all cases that violate in a database `csit115` the following consistency constraint.

"An employee who is a manager of a department must be a member of the same department"

The script must list the outcomes of verification of the consistency constraint as a single column table with the following messages as the following rows.

`An employee` <Insert employee id here> `who manages a department` <insert name of department here> `is a member of department` <Insert name of department here>

Use a function `CONCAT` to create the messages above.

Note, that it is NOT your task to eliminate the violations of consistency constraint listed above.

Deliverables
Submit a file `solution3.rpt` with a report from processing of SQL script `solution3.sql`. The report MUST have no errors and the report MUST list all SQL statements processed.

A report that contains no listing of executed SQL statements scores no marks !

A report that contains processing errors scores no marks !

Processing of the script on an empty database scores no marks !

Processing of the script that contains statements other from `UPDATE`, `SELECT`, `source`, and `notee` scores no marks!

Submission of a file with a different name and/or different extension and/or different type scores no marks!

Task 4 (1.5 marks)
Some of simpler Database Management Systems, like for example MySQL 5.7 Community Edition, do not have the features that allow for automated auditing the database activities. In this task you will implement your own simple method of auditing the database activities.

Connect to MySQL either through command line interface mysql or graphical user interface MySQL Workbench and execute a script file `dbdrop.sql` and immediately after that execute script `dbcreate.sql` and `dbload.sql` to refresh a sample database.

Implement SQL script `solution4.sql` that performs the following actions.

(1) First, the script sets the appropriate values of the variables that allow create a general log, to save a general log in a relational table, and to start recording a general log from now.

(2) Next, the script makes a relational table that contains a general log empty.

(3) Next, the script executes a script file `dbchange4.sql`. Do not put results of execution of script `dbchange4.sql` into a report.

(4) Next, the script sets the appropriate values of all variables that stop recording a general log from now.

(5) Next, the script lists all DDL statements (`CREATE`, `ALTER`, `DROP`) processed in a period of time when a general log was recorded.

(6) Next, the script lists total number of times each one of DML statements like `SELECT`, `INSERT`, `UPDATE`, `DELETE` has been processed in a period of time when a general log was recorded. It is acceptable to ignore DML statement if it has not been processed at all. Sort the results in an ascending order of the total number times a DML statement has been processed.

Deliverables
Submit a file `solution4.rpt` with a report from processing of SQL script `solution4.sql`. The report MUST have no errors and the report MUST list all SQL statements processed.

A report that contains no listing of executed SQL statements scores no marks !

A report that contains processing errors scores no marks !

Processing of the script on an empty database scores no marks !

Processing of the script that contains statements other than implementing the actions from (1), (2), (4), (5), and (6) scores no marks !

Submission of a file with a different name and/or different extension and/or different type scores no marks !

Submission

Note, that you have only one submission. So, make it absolutely sure that you submit the correct files with the correct contents. No other submission is possible !

Submit the files `solution1.rpt`, `solution2.rpt`, `solution3.rpt`, `solution4.rpt`, and a backup file ***a prefix of your University email account*.bak** through Moodle in the following way:

(1) Access Moodle at `http://moodle.uowplatform.edu.au/`

(2) To login use a **Login** link located in the right upper corner the Web page or in the middle of the bottom of the Web page

(3) When logged select a site **CSIT815/CSIT115 (S117) Data Management and Security**

(4) Scroll down to a section **Submissions**

(5) Click at a link **In this place you can submit the outcomes of Assignment 4**

(6) Click at a button **Add Submission**

(7) Move a file **solution1.rpt** into an area **You can drag and drop files here to add them**. You can also use a link **Add**...

(8) Repeat step (7) for the files **solution2.rpt**, **solution3.rpt**, **solution4.rpt**, and backup file ***a prefix of your University email account*.bak**

(8) Click at a button **Save changes**

(9) Click at a button **Submit assignment**

(10) Click at the checkbox with a text attached: **By checking this box, I confirm that this submission is my own work,** ... in order to confirm authorship of your submission.

(11) Click at a button **Continue**

A policy regarding late submissions is included in the subject outline.

Only one submission of the fourth assignment is allowed and only one submission per student is accepted.

A submission marked by Moodle as "late" is always treated as a late submission no matter how many seconds it is late.

A submission that contains an incorrect file attached is treated as a correct submission with all consequences coming from the evaluation of the file attached.

It is expected that all its tasks included in **Assignment 4** will be solved **individually without any cooperation** with the other students. If you have any doubts, questions, etc. please consult your lecturer or tutor during lab classes or office hours. Plagiarism will result in a **FAIL** grade being recorded for that assessment task.

The evaluated outcomes of will be electronically returned to the students before 11.55pm on Monday, 12 June, 2017.

End of specification