CSIT115 Database Management and Security
Laboratory 4

Scope

This laboratory includes the following:

- tasks related to the applications of `CREATE DATABASE`, `CREATE USER`, and `GRANT` statements of SQL
- tasks related to granting access right to the subsets of relational tables and verifying consistency constraints in the relational tables

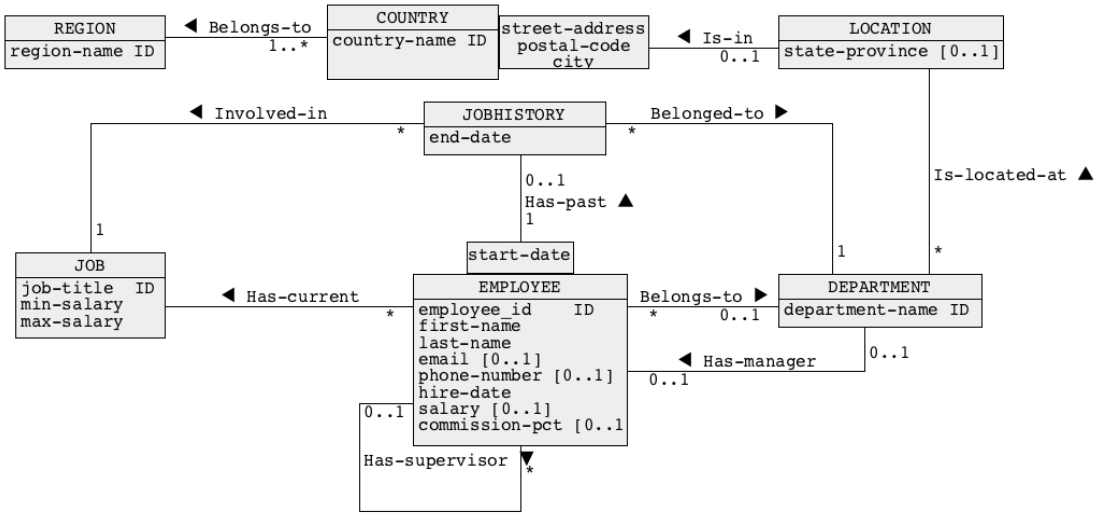
Specification of each task starts from a new page.

It is recommended to solve the problems before attending an enrolled laboratory class in order to efficiently use supervised laboratory time.

Prologue

Download the files `dbcreate.sql`, `dbdrop.sql`, and `dbload.sql`. Copy the files to your USB drive such that you can access the files either through command line interface mysql or graphical user interface MySQL Workbench. You can also email the files to yourself such that you can access it on different systems. Finally, the simplest solution is to download the files directly to Ubuntu Linux from `moodle`.

Connect to MySQL either through command line interface mysql or graphical user interface MySQL Workbench and execute script files `dbcreate.sql` and `dbload.sql`. The script files create and load data into a database that contain 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.



Task 1

Connect to a database server as a user `root` with a password `csit115`.
Implement SQL script `solution1.sql` that performs the following actions.

(1) Create a database with the same name as a *prefix of your University email account*. For example, if your University email account is `xyz007@uow.edu.au` then a name of a database must be `xyz007`.

(2) Create three users with the following user names: *prefix of your University email account*`_1`, *prefix of your University email account*`_2`, and *prefix of your University email account*`_3`. For example, if a prefix of your University email account is `xyz007` then the names of users are `xyz007_1`, `xyz007_2`, and `xyz007_3`. All passwords are up to you.

(3) While connected as a user `root`, execute the scripts `dbcreate.sql` and `dbload.sql` to create and to load data into the relational tables later on used in this laboratory class. All relational tables must be located in a database created in step (1). SQL statements processed by the scripts must NOT be included in a report `solution1.rpt`. It means that before processing of the scripts you must execute `notee` statement to turn off spooling and after processing of the scripts `tee solution1.rpt` to turn on spooling into a report file.

(4) The script grants a read privilege on entire database *prefix of your University email accoun*t to a user *prefix of your University email account*`_1`. The privilege must be granted such that a user *prefix of your University email account*`_1` is not allowed to grant the same privilege to another user.

(5) Next, the script grants write privileges on a relational table `EMPLOYEE` located in a database *prefix of your University email account* to a user *prefix of your University email account*`_2`. The privileges must be granted such that a user *prefix of your University email account*`_2` is able to grant the same privileges to the other users.

(6) Next, the script grants a privilege to create relational tables located in a database *prefix of your University email account* to a user *prefix of your University email account*`_3`. The privilege must be granted such that a user *prefix of your University email account*`_3` is not allowed to grant the same privilege to another user.

(7) Next, the script grants a privilege to read the columns (`department_name, street_address, city, country_name`) in a relational table `DEPARTMENT` located in a database *prefix of your University email account* to a user *prefix of your University email account*`_3`. The privilege must be granted such that a user *prefix of your University email account*`_3` is not allowed to grant the same privilege to another user.

(8) Finally, the script lists all privileges granted to the users *prefix of your University email account*`_1`, *prefix of your University email account*`_2`, and *prefix of your University email account*`_3`. The script must use data dictionary views included in `mysql` database to list the privileges.

Task 2

Process a script `solution1.sql` implemented in the previous step and do not drop a database, users, and relational tables created by the script. Do not change the access rights granted by the script.

Use mysql command line interface to perform the following actions.

(1) Start mysql command line interface and connect as a user *prefix of your University email account*`_1`.

(2) Execute a command `tee solution2.rpt`.

(3) Execute two SQL statements that show the validity of two different privileges granted to a user *prefix of your University email account*`_1` in the previous task. Note, that you have to use a database *prefix of your University email account*. Each statement must retrieve precisely two rows.

(4) Execute any SQL statement that shows a lack of privilege to access a database *prefix of your University email account* in write mode by a user *prefix of your University email account*`_1`.

(5) Exit mysql command line interface.

(6) Start mysql command line interface and connect as a user *prefix of your University email account*`_2`.

(7) Execute a command `tee solution2.rpt`.

(8) Execute two SQL statement that shows the validity of two different privileges on a relational table `EMPLOYEE` located in a database *prefix of your University email account* and granted to a user *prefix of your University email account*`_2` in the previous task.

(9) Execute three SQL statement that shows a lack of write privilege on a relational table `DEPARTMENT` located in a database *prefix of your University email account* by a user *prefix of your University email account*`_2` in the previous task.

(10) Exit mysql command line interface.

(11) Start mysql command line interface and connect as a user *prefix of your University email account*`_3`.

(12) Execute a command `tee solution2.rpt`.

(13) Execute any SQL statement that shows the validity of a privilege to create a relational table located in a database *prefix of your University email account* and granted to a user *prefix of your University email account*_3 in the previous task.

(14) Execute any SQL statement that shows a lack of privilege to create a relational table in a database csit115 by a user *prefix of your University email account*_3.

(15) Execute any SQL statement that shows the validity of privilege to read the columns (department_name, street_address, city, country_name) from a relational table DEPARTMENT located in a database *prefix of your University email account* and granted to a user *prefix of your University email account*_3 in the previous task. The statement must retrieve precisely one row.

(16) Execute any SQL statement that shows a lack of privileges to read a column other than (department_name, street_address, city, country_name) from a relational table DEPARTMENT located in a database *prefix of your University email account* and granted to a user *prefix of your University email account*_3 in the previous task.

(17) Execute a command notee.

(18) Exit mysql command line interface.

Task 3

Remove all relational tables from a database `csit115`. No report is expected from this step.

Execute the commands (or scripts) that perform the following actions.

Login as a user `root` through a command line interface mysql and perform the following actions.

(1) Execute a command `tee solution3.rpt`.

(2) Create two database users with the names the same as *a prefix of your University account* concatenated with `_1` in a case of the first user and concatenated with `_2` in a case of the second user.

(3) Execute the command `notee`.

(4) Execute the command `exit` to logout as a user `root`.

(5) Login as a user `csit115` through command line interface mysql and execute command `use csit115`.

(6) Execute SQL scripts `dbcreate.sql` and `dbload.sql` to create and to load data into a sample database.

(7) Execute the command `tee solution3.rpt`.

(8) Create in a database `csit115` a relational view `EMPJOBS` that allows for access to information about employees and total number of finished jobs in the following format.

```
  ENUM  |  NAME          |  EMAIL    | FINISHEDJOBS
-------------------------------------------------------------
  100   | Steven King    | SKING     | 0
  101   | Neena Kochhar  | NKOCHHAR  | 2
 ...    |  ...           |  ...      | ...
```

(9) Grant a read privilege to all information included a view `EMPJOBS` to a user with the same name as *a prefix of your University email account* `_1`.

(10) Grant a read privilege to all information included in a view `EMPJOBS` except the column `FINISHEDJOBS` to a user with the same name as *a prefix of your University email account* `_2`.

(11) Execute the command `notee`.

(12) Execute command `exit` to logout the user `csit115`.

(13) Login the user `root` through command line interface mysql and execute a command `tee solution3.rpt`.

(14) Display the read privileges granted to both users. The information should include user name, database name, table name, table privileges and column privileges. You must use data dictionary views included in `mysql` database to list the privileges.

(15) Execute a command `notee`.

Task 4

Refresh the contents of `csit115` database with SQL scripts `dbdrop.sql`, `dbcreate.sql` and `dbload.sql`. No report is expected from this step.

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

(1) The script uses a database `csit115`.

(2) Next, the script changes a value of a system variable `AUTOCOMMIT` to `'OFF'`.

(3) The script changes the contents of a relational table `EMPLOYEE` by changing a name of a department to `Shipping` for an employee `177`.

(4) The script changes the contents of a relational table `EMPLOYEE` by changing a name of a department to `Executive` for an employee `144`.

(5) Next, the script verifies the following consistency constraint.

A*ll employees that have the same job title must belong to the same department.*

For example: All employees that work as `Stock Manager`s belongs to a department `Shipping`.

If any of the employees work in the other department with the same job title, the script must display the violations of the consistency constraint defined above in the following format.

```
JOB TITLE      |    EMPLOYEE ID    | DEPARTMENT NAME
```

(6) Next, the script reverses the modifications done in the steps (3) and (4) in the simplest possible way.

(7) Finally, the script repeats verification of the same consistency constraint as in a step (5).