School of Computing and Information Technology
University of Wollongong

Session: **Spring 2019**
Lecturers: **Janusz R. Getta**
Tianbing Xia

CSIT115/CSIT815 Data Management and Security
Assignment 3

Published on 17 October 2019

Scope

This assignment is related to verification of a complex consistency constraint, implementation of a simple auditing system, and database backup and recovery techniques.

Please read very carefully information listed below.

This assignment contributes to 8% of the total evaluation in a subject CSIT115 and it contributes to 8% of the total evaluation in a subject CSIT815.

The outcomes of the assignment work are due by **Saturday 2 November 2019, 7.00 pm (sharp).**

A submission procedure is explained at the end of specification.

This assignment consists of 3 tasks and specification of each task starts from a new page.

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

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

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

A submission of compressed files (zipped, gzipped, rared, tared, 7-zipped, lhzed, … etc) is not allowed. The compressed files will not be evaluated.

All files left on Moodle in a state "`Draft(not submitted)`" will not be evaluated.

An implementation that does not compile due to one or more syntactical errors scores no marks and implementation that has the processing errors scores no marks.

It is expected that all tasks included within **Assignment 3** 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 the assessment task.

Prologue
Connect to Moodle and download the files `dbcreate.sql`, `dbdrop.sql`, `dbload.sql`, `dbcount.sql`, and `dbschema.pdf` from Sample database section on Moodle.

SQL script `dbcreate.sql` can be used to create the relational tables of a sample database. SQL script `dbdrop.sql` can be used to drop the tables of a sample database. SQL script `dbload.sql` can be used to load data into a sample database. SQL script `dbcount.sql` can be used to display the total number of rows in each table included in a sample database. Finally, a file `dbschema.pdf` contains a conceptual schema of a sample database.

Connect to MySQL database server either through command line interface mysql or graphical user interface MySQL Workbench.

When connected, select a database `csit115` with a command `use csit115`.

To create the relational tables of a sample database, process SQL script `dbcreate.sql`.

To load data into the relational tables created in the previous step process SQL script `dbload.sql`.

To list the names of relational tables created, use a command `show tables`.

To list a structure of a relational table `<table-name>` use a command `describe <table-name>`.

To list the total number of rows in each relational table process a script `dbcount.sql`.

Use a pdf viewer to open a file `dbschema.pdf` with a conceptual schema of the sample database.

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

Tasks

Task 1 (2 marks)

An objective of this task is to implement SQL script that verifies the following logical consistency constraint imposed on the contents of a sample database.

"A driver is not allowed to perform more than 1 trip per day"

Download a file `solution1.sql` and insert into the file the implementations of the following actions.

(1) First, the script inserts into a sample database information about a new trip that consist of two legs. A trip must be performed by a driver who has already performed a trip on the same day. You are allowed to examine the contents of a sample database to find out which driver performed at least one trip and later on apply `INSERT` statements to insert information about the same driver who performed another trip on the same day. Next, insert information about two legs that belong to the already inserted trip. All other information about a new trip and its legs is up to you.

(2) Next, the script creates a single column relational table `MESSAGE` to store variable size strings no longer than 500 characters.

(3) Next, the script inserts into a relational table `MESSAGE` information about the contents of a sample database that violate the consistency constraint.

"A driver is not allowed to perform more than 1 trip per day"

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

`A driver` <insert `driving licence number` here> `performed` more than one trip on <insert `trip date` here>

For example, if a `driving licence number` of a driver who performed more than one trip on `1 May 2019` is equal to `7` then verification of the consistency constraint must return the following message.

```
A driver 7 performed more than one trip on 1 May 2019
```

Use a function `CONCAT` to create the messages like the one listed above.

(4) Finally, the script makes the contents of a relational table `MESSAGE` permanent and lists the contents of the table.

When ready connect as `csit115` user, process a script file `solution1.sql`, and save a report from the processing in a file `solution1.rpt`.

To create a report from processing of a script file `solution1.sql` open a Terminal window and start the command line interface `mysql` in the following way:

```
mysql -u csit115 -p -v -c
```

Next, process SQL script `solution1.sql` and save a report in a file `solution1.rpt`. Note, that when started with the options **`-v`** and **`-c`** the command line interface includes both listing of `SELECT` statements processed and the comments included in the original version of a file `solution1.sql`.

Deliverables

A file `solution1.rpt` with a report from processing of SQL script `solution1.sql`. The report must be created with the command line interface mysql, the report MUST NOT include any errors, and the report must list all SQL statements processed and all comments included in the original (downloaded) version of `solution1.sql`. Marks will be deducted for the missing comments. Submission of a file with a different name and/or different extension and/or different type scores no marks.

Task 2 (3 marks)
An objective if this task is to use a backup file and recovery feature of a database system to find historical information from a sample database. On 31 May 2015 a database administrator created a backup of a relational table `DRIVER` and saved a backup in a file `old_driver.bak`. Then to make your task easier, a database administrator updated the file such that it can be restored into a relational table `OLD_DRIVER`.

Implement a script file `solution2.sql` that performs the following actions.

(1) Create a relational table `OLD_DRIVER` that has the same columns as a relational table `DRIVER`. Enforce appropriate consistency constraints for a relational table `OLD_DRIVER`.

(2) Use a backup file `old_driver.bak` to load the old pre 31 May 2015 contents of a relational table `DRIVER` into a relational table `OLD_DRIVER`.

(3) Use `SELECT` statements to list the employee numbers and driving license numbers of drivers and who left the transportation company after 31 May 2015.

(4) Use `SELECT` statements to list the employee numbers, driving license numbers, first name and last name of drivers and joined the transportation company after 31 May 2015.

Before processing of a script file `solution2.sql` it is strongly recommended to connect to MySQL either through command line interface mysql or graphical user interface MySQL Workbench and process a script file `dbdrop.sql` and immediately after that the scripts `dbcreate.sql` and `dbload.sql` to refresh a sample database.

When ready connect as `csit115` user, process a script file `solution2.sql`, and save a report from processing in a file `solution2.rpt`.

To create a report from processing of a file `solution2.sql` open a Terminal window and start the command line interface `mysql` in the following way:

```
mysql -u csit115 -p -v -c
```

Next, process SQL script `solution2.sql` and save a report in a file `solution2.rpt`. Note, that when started with the options **`-v`** and **`-c`** the command line interface includes both listing of SQL statements processed and the comments included in the original version of a file `solution2.sql`.

Deliverables

A file `solution2.rpt` with a report from processing of SQL script `solution2.sql`. The report must be created with the command line interface mysql, the report MUST NOT include any errors, and the report must list all SQL statements processed and all comments included in the original (downloaded) version of `solution2.sql`. Marks will be deducted for the missing comments. Submission of a file with a different name and/or different extension and/or different type scores no marks.

Task 3 (3 marks)
An objective of this task is to implement your own simple method of auditing the database activities.

It is strongly recommended to connect to MySQL either through command line interface mysql or graphical user interface MySQL Workbench and process a script file `dbdrop.sql` and immediately after that the scripts `dbcreate.sql` and `dbload.sql` to refresh a sample database.

Download a file `solution3.sql` and insert into the file the implementations of the following actions.

(1) First, the script makes a relational table that contains a general log empty.

(2) Next, the script sets the appropriate values of the variables to save a general log in a relational table and to start recording a general log from now.

(3) Next, the script makes a database `csit115` a default database, it stops recording a report, it executes a script file `workload.sql`, and it resumes recording a report into a file `solution3.rpt`.

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

(5) Finally, the script finds and lists how many times each one of the relational tables included in a sample database have been used by the successfully processed SQL statements included in SQL script `workload.sql`. You have to consider the relational tables with the following names `EMPLOYEE`, `DRIVER`, `ADMIN`, `TRUCK`, `TRIP`, and `TRIPLEG` and no other relational tables. The script must list the names of relational tables together with the total number of times each table has been used. Please, find a fragment of a sample output listed below.

```
+------------+-------+
| TABLE_NAME | TOTAL |
+------------+-------+
| EMPLOYEE   |     5 |
| TRIP       |     3 |
  ...             ...
+------------+-------+
6 rows in set (0.01 sec)
```

To simplify this task, assume that a relational table is used no more than one time in SQL statement.

The results must be listed in the descending order of the total number of times each one of the relational tables has been used by the successfully processed SQL statements included in a script `workload.sql`. Note, that some of SQL statements included in a script `workload.sql` cannot be successfully processed and because of that counting the total number of names of relational tables in the script does not provide the correct results. To find the correct results you must access an earlier recorded general log.

When ready connect as `root` user, process a script file `solution3.sql`, and save a report from processing in a file `solution3.rpt`.

To create a report from processing of a file `solution3.sql` open a Terminal window and start the command line interface `mysql` in the following way:

```
mysql -u root -p -v -c
```

Next, process SQL script `solution3.sql` and save a report in a file `solution3.rpt`. Note, that when started with the options **`-v`** and **`-c`** the command line interface includes both listing of `SELECT` statements processed and the comments included in the original version of a file `solution3.sql`.

Deliverables
A file `solution3.rpt` with a report from processing of SQL script `solution3.sql`. The report must be created with the command line interface mysql, the report MUST NOT include any errors, and the report must list all SQL statements processed and all comments included in the original (downloaded) version of `solution3.sql`. Marks will be deducted for the missing comments. 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 and correct types. No other submission is possible !

Submit the files **`solution1.rpt`**, **`solution2.rpt`**, and **`solution3.rpt`** 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 **`CSIT115/CSIT815 (S219)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 3`**
(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`** and **`solution3.rpt`**.
(9) Click at a button **`Save changes`**
(10) Click at a button **`Submit assignment`**
(11) 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 the authorship of your submission
(12) Click at a button **`Continue`**

End of specification