School of Computing and Information Technology
University of Wollongong

Session: **Summer 2020**
Lecturer: **Janusz R. Getta**

CSCI835 Database Systems
Assignment 5
19 July 2020

Scope

This assignment includes the tasks related to data manipulations on MongoDB database, and implementation of queries and data transformations with aggregation framework and implementation of data validation with JSON schema.

The outcomes of the assignment are due by **Saturday, 25 July, 2020, 9.00 pm (sharp).**

Please read very carefully information listed below.

This Assignment contributes to 13% of the total evaluation in a subject CSCI835 Database Systems.

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.

It is expected that all tasks included within **Assignment 5** 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 the laboratory classes. Plagiarism will result in a **FAIL** grade being recorded for the assessment task.

Prologue

If you do not have access to MongoDB database server the perform the actions listed below. Otherwise, progress to a specification of **Task 1**.

If VirtualBox is not installed yet then install it on your system first. It is explained at

https://documents.uow.edu.au/~jrg/115/cookbook/e1-1-frame.html

how to install and how to use VirtualBox.

Next, use a link provided on Moodle and located just below a link to a lecture class to download `Ubuntu18.04-64bits-MongoDB-4.2.2-08-JAN-2020.ova` file with an image of virtual machine running `MongoDB 4.2.2` on `Ubuntu 18.04`. When downloading, you must use Firefox browser. You can also use a link given below.

https://cloudstor.aarnet.edu.au/plus/s/qg8J3vE4SoxRzKo

Next, start VirtualBox and import a virtual machine `Ubuntu18.04-64bits-MongoDB-4.2.2-08-JAN-2020.ova`.

Start VirtualBox and import a virtual machine `Ubuntu18.04-64bits-MongoDB-4.2.2-08-JAN-2020.ova`.

Next, start the virtual machine and login as an Ubuntu Linux user `CSCI235` with a password `csci235`.

Next, start `Terminal` program and within a `Terminal` window start `MongoDB` server in the following way.

```
mongod –dbpath DATA –port 4000
```

A server displays a lot of messages. A successful start of `MongoDB` server is confirmed with a message like

```
…
… I NETWORK  [initandlisten] waiting for connections on port 4000…
…
```

pretty well hidden somewhere at the end of a long list of other messages issued by the starting server.

Note, that the `Terminal` window you use just now becomes a console of the running `MongoDB` server and you cannot use it any more. Do not close the `Terminal` window ! Just minimize it.

Open a new `Terminal` window. To create a `BSON` collection `orders` use a command

```
cd CSCI235
```

to move to `CSCI235` folder with the scripts.

Next, to start a command line client `mongo` process the following command.

```
mongo –port 4000
```

To make sure that client is connected to a database server process the commands `help` and `show dbs to` display available databases.

Tasks
Task 1 (5 marks)
Data manipulations

Implement the following 5 data manipulation operations on a collection `orders`.

(1) Append a new product `Changde Noodles` that belongs to a category `Noodles` to a list of products supplies by a supplier located in a city `Zaandam`. All other information is unknown at the moment. Display the names of products supplied by a supplier located in a city `Zaandam`.

(2) Remove information about a product `Longlife Tofu` supplied by a supplier `Tokyo Traders`. Display the names of products supplied by a supplier `Tokyo Traders`.

(3) Increase a `unit price` of a product `Flotemysost` by 100%. Display the `product name` and the changed `unit price` in a pretty format.

(4) Rename a key `submits` to a key `sends` in the orders submitted by a customer `FAMIA`. Display all information about a customer `FAMIA`.

(5) An order with `order_id` equal to `310` is now handled by an employee with `employee id` equal to `7`. Update the database. After update display `order_id` and `employee id` in a pretty format.

Implement the data manipulations listed above in a data manipulation language of MongoDB.

When ready create a report from processing of data manipulations and save it in file `solution1.lst`. To do so, use `gedit` editor and open a new file `solution1.lst`. Next, select the entire contents of the Terminal window and Copy&Paste it into a file `solution1.lst`. Save a file `solution1.lst`.

Deliverables
A file `solution1.lst` with a report from processing of data manipulations.

Task 2 (5 marks)
Query processing and data transformation with aggregation framework

Use aggregation framework to implement the following 5 queries and data transformations.

(1) Save all information about the names of products supplied by a supplier `Gai paturage` into a collection `products1`. Display in a pretty format without document identifiers all documents in a collection `products1`.

(2) Save all information about the names of products supplied by a supplier `Gai paturage` into a collection `products2` that consists of the documents like `{"product name": a-name-of-product}`. Display in a pretty format without document identifiers all documents in a collection `products2`.

(3) Find the total number of products in a collection `orders`. Display a result in a format `{"total number of products":integer-value}`.

(4) List in the ascending order the ids of the first 3 employees who handled at least one order. Display the results in a format `{"employee id":a-value-of employee-id}`. List only distinct values.

(5) Find the company names of suppliers together with the total number of supplied products by each company. Display the results in a format `{"total products":integer-value,"company name":a-company-name}`.

Use the methods `aggregate()` and `pretty()` to implement the queries and data transformations and to display the results. Note, that you may need two or more statements to implement a single task.

When ready create a report from processing of simple data manipulations in file `solution2.lst`. To do so, use `gedit` editor and open a new file `solution2.lst`. Next, select the entire contents of the Terminal window with the results from processing of the methods `aggregate()` and `pretty()` and Copy&Paste it into a file `solution2.lst`. Save a file `solution2.lst`.

Deliverables
A file `solution2.lst` with a report from processing of `aggregate()` and `pretty()` methods.

Task 3 (3 marks)
Implementation of validation with JSON schema

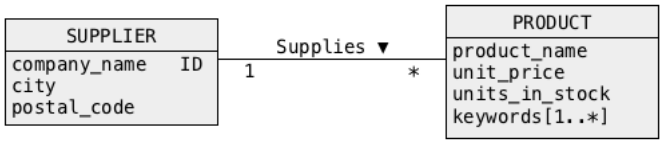
Start a command line interface mongo to MongoDB database server and process the following statements.

```
db.adminCommand( {setFeatureCompatibilityVersion:"4.2"} )
use test;
```

The first statement sets an appropriate compatibility level for application of JSON schema validator and the second statement sets an appropriate default database to be used with JSON Schema validator.

No report is expected from the actions listed above.

Consider the following conceptual schema of a database that contains information about suppliers and products.



An objective of this task is to create a new collection of documents `task3` that contains information represented by a conceptual schema above and such that the collection is validated with JSON schema validator.

Use a method `createCollection()` to create a collection of documents `task3` and use JSON schema validator to enforce the following constraints on the collection.

(1) Information about products must be nested within information about suppliers.
(2) Information about keywords describing products must be nested within information about products.
(3) The values associated with the key names `company name`, `city`, (within `SUPPLIER` class), `product name`, `keywords` (within `PRODUCT` class) must be of type string. All values are mandatory.
(4) The values associated with a key `postal code` (within `SUPPLIER` class must be of type integer in a range `0..9999`.
(5) The values associated with a key `units in stock` (within `PRODUCT` class) must be of type integer and must be positive.
(6) The values associated with a key `unit price` must be of type double and must be positive and less than `100.00`.
(7) A key `company name` is mandatory.

Next, insert into a collection `task3` two sample documents. The first document must pass all validations of the constraints listed above. The second document must fail a validation of only one of the constraints listed above. Provide information why a document fails a validation.

When ready create a report from processing of `createCollection()` method with JSON schema validator and with the validation of two documents in file `solution3.lst`. To do so, use `gedit` editor and open a new file `solution3.lst`. Next, select the entire contents of the Terminal window with the results from processing of `createCollection()` method with JSON schema validator and with the validation of two documents and Copy&Paste it into a file `solution3.lst`. Save a file `solution3.lst`.

Deliverables

A file `solution3.lst` with a report from processing of of `createCollection()` method with JSON schema validator and with the validation of two documents.

Submission

Submit the files **solution1.lst**, **solution2.lst**, and **solution3.lst** 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 **CSCI835 (JIS20) Database Systems**
(4) Scroll down to a section **SUBMISSIONS**
(5) Click at a link **In this place you can submit the outcomes of Assignment 5**
(6) Click at a button **Add Submission**
(7) Move a file **solution1.lst** into an area **You can drag and drop files here to add them**. You can also use a link **Add**...
(8) Repeat a step (7) for the files **solution2.lst**, and **solution3.lst**.
(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