School of Computing and Information Technology
University of Wollongong

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

CSCI235/CSCI835 Database Systems
Assignment 3
26 October 2020

Scope

This assignment includes the tasks related to a design of logical schema of MongoDB database, data manipulation in MongoDB, application of aggregation framework in MongoDB, and indexing of the collections of documents.

The outcomes of the laboratory work are due by **Saturday 14 November, 2020, 7.00 pm (sharp).**

Please read very carefully information listed below.

This assignment contributes to 20% of the total evaluation in a subject CSCI235 and it contributes to 17% of the total evaluation in a subject CSCI835.

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 and/or run time 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
Install VirtualBox on your system, if it is not installed yet. If you do not remember how you did it in CSIT115 then it is explained in

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

how to do it.

Download from Moodle `ova` image of a virtual machine with Ubuntu and MongoDB. The image is available in a section `OTHER RESOURCES`. You should get a file:

```
Ubuntu18.04-64bits-MongoDB-4.2.2-08-JAN-2020.ova
```

Start VirtualBox and import `ova` image of a virtual machine with Ubuntu and MongoDB. You should get a new virtual machine `Ubuntu18.04-64bits-MongoDB-4.2.2-08-JAN-2020.`

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

A password to login as `CSCI235` user is:

```
csci235
```

When logged in, start Terminal program (3rd icon from bottom in a column of icons on the left-hand size of a screen).

To start MongoDB server, process the following command in Terminal window.

```
mongod --dbpath DATA --port 4000
```

When MongoDB server is ready then among many, many, … the other messages you should get a message:

```
… waiting for connection on port 4000
```

Minimize Terminal window. Do not close the window, from now, it is used as a console window by MongoDB server.

Open another Terminal window and to start MongDB command line interface, process the following command.

```
mongo –port 4000
```

For a good start, process a command `help`.

Download to your virtual machine the files: `bsontpchr.bmp`, `customer.zip`, `part.zip`, and `supplier.zip` from a section `SAMPLE DATABASES` on Moodle.

Unzip the files: `customer.zip`, `part.zip`, and `supplier.zip`.

You should get the files: `customer.js`, `part.js`, and `supplier.js`.

To create a collection `tpchr` and to load the documents into the collection, process the scripts `customer.js`, `part.js`, and `supplier.js`. at > prompt of `mongo` client in the following way.

```
load("customer.js");
load("part.js");
load("supplier.js");
```

A logical schema of a collection `tpchr` is available in a file `bsontpchr.bmp`. It is strongly recommended to make yourself familiar with a logical schema of a sample database.

Next, you can use the methods

`db.orders.find().count()` and
`db.orders.find().pretty()`

to count the total number of the documents in a collection `tpchr` and to list all documents in a pretty format.

Task 1 (7.5 marks)
Data manipulations

Download and unzip a file solution1.zip. You should get a file solution1.js. The file contains the specifications of the following 5 data manipulation operations on a collection tpchr.

(1) Remove information about an address from a description of a customer who submitted an order with an order key 1031. Next list a complete description of a customer who submitted an order with an order key 1031.

(2) Rename a key "shipped" to "shipped by" in the parts that have a retail price less than 902. Next, display all information about parts that have a retail price less than 902.

(3) Increase an account balance of all suppliers from Japan by 50% of the present value. Next, display a supplier name, nation and balance of all suppliers from Japan in a pretty format.

(4) Append to all parts that belong to a brand Brand#54 the following information about a new shipment:
```
"partsupp_id" : "5_1",
"availqty" : 100,
"supplycost" : 17,
"ref supplier" : "1".
```
Next list all information about all parts that belong to a brand Brand#54 in a pretty format.

(5) Remove information about an order that has a key 1031 submitted by a customer whose key is 8. Next, list all information about all orders submitted by a customer whose key is 8.

Implement the data manipulations listed above in a data manipulation language of MongoDB. Write your solutions into the empty slots following a specification of each data manipulation in a file solution1.js. Do not remove the specifications of the data manipulations and semicolons following the specifications.

Implementation of each data manipulation is worth 1.5 mark.

When ready create a report from processing of the data manipulations in the following way.

Use gedit editor to open a file solution1.js with the specifications and implementations of the data manipulations.

Select the entire contents of the file and Copy it into a buffer.

Open a new `Terminal` window and start `mongo` client in the following way.

```
mongo -port 4000
```

Paste the contents of the buffer copied earlier from `gedit` window in front of > prompt of `mongo` client. You may have to press `Enter` key to process the last data manipulation in a case when it is not followed by a newline control character.

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 MongoDB script `solution1.js` with the implementation of the data manipulations listed above.

And again, please remember that:

- a report without the specifications of the data manipulations and listings of the processed data manipulations scores no marks,
- a report that contains any kind of processing errors scores no marks.

Task 2 (7.5 marks)
Query processing and data transformation with aggregation framework

Drop a collection tpchr in the following way.

```
db.tpchr.drop();
```

To re-create a collection tpchr and to load the documents into the collection, process the scripts customer.js, part.js, and supplier.js. at > prompt of mongo client in the following way.

```
load("customer.js");
load("part.js");
load("supplier.js");
```

Download and unzip a file solution2.zip. You should get a file solution2.js. The file contains the comments with the specifications of the following 5 queries and data transformations.

(1) Save information about a customer key, name, and nation of all customers from SUDAN or ROMANIA or CANADA into a collection SUROCAN. Display in a pretty format without document identifiers the contents of a collection SUROCAN.

(2) Save all information about the supply costs (supplycost)of a part with a name floral moccasin royal powder burnished into a collection supplycosts that consists of the documents like {"supply cost": a-value-of-supply-cost}. Display in a pretty format without document identifiers all documents in a collection supplycosts.

(3) Find the total number of part shipments of the parts of type LARGE BURNISHED STEEL or SMALL BURNISHED STEEL. Display a result in a format
{"total number of shipments":integer-value}.

(4) Find the total number of shipments per each part. List the results in a format
{"total number of shipments":integer-value,
 "part key":integer-value"}.

(5) Find 5 largest extended prices (extended price) from all orders. List the results in a format
{"customer key": integer-value,
 "order key":integer-value,
 "line number":integer-value,
 "price":floating-point-value}}.

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

Implementation of each query/data transformation is worth 1.5 mark.

When ready create MongoDB script file solution2.js with the implementations of your queries and create a report from processing of the data manipulations in the following way.

Use gedit editor to open a file solution2.js with the specifications and implementations of the data manipulations.

Select the entire contents of the file and Copy it into a buffer.

Open a new Terminal window and start mongo client in the following way.

```
mongo –port 4000
```

Paste the contents of the buffer copied earlier from gedit window in front of > prompt of mongo client. You may have to press Enter key to process the last data manipulation in a case when it is not followed by a newline control character.

Select the entire contents of the Terminal window 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 MongoDB script solution2.js with the implementation of the data manipulations listed above.

Please remember that:

- a report without the specifications of the queries and data manipulations and listings of the processed queries and data manipulations scores no marks,
- a report that contains any kind of processing errors scores no marks.

Task 3 (5 marks)
Implementation of indexing

Download and unzip a file `solution2.zip`. You should get a file `solution2.js`.

Consider the documents included in a collection `tpchr` and the queries consistent with the following query templates.

(1) Find all parts that belongs to a given brand.

(2) Find all parts that has a retail price greater than a given value.

(3) Find the names of all suppliers.

(4) Find the brands and types of all parts.

(5) Find the names of customers who submitted at least one order.

Repeat the implementations of the following four steps for each one of the query patterns listed above.

Step 1 Create an index that speeds up processing of a query consistent with a pattern.

Step 2 Apply a method `getIndexes()` to list all existing indexes, for example `db.collection.getIndexes()`.

Step 3 Apply a method `explain()` to verify whether the system plans to use the indexes created for processing of a query consistent with a pattern, for example
`db.tpchr.find({"CUSTOMER.nation":"KENYA"}).explain();`
The constants used in a query are up to you.

Step 4 Drop an index created in Step 1 with a method `dropIndex()`, e.g.
`db.collection.dropIndex("index_name").`
You can find a name given to an index by the system from the results of the Step 2.

Write your solutions into a file `solution3.js` in the empty slots following a specification of each problem. Do not remove the comments with the specifications of queries and semicolons following the comments !

Implementation of each index, displaying query processing plans and dropping an index is worth 1 mark.

When ready create a report from processing of the queries in the following way. Use `gedit` editor to open a file `solution3.js` with the specifications of the queries and implementations of the queries.

Select the entire contents of the file and Copy it into a buffer.
Open a new `Terminal` window and start `mongo` client in the following way.

```
mongo -port 4000
```

Paste the contents of the buffer copied earlier from `gedit` window in front of > prompt of `mongo` client. You may have to press Enter key to process the last query in a case when it is not followed by a newline control character.

Select the entire contents of the `Terminal` window and Copy&Paste it into a file `solution3.lst`. Save a file `solution3.lst`. Examine the contents of a file `solution3.lst` for possible errors.

Deliverables
A file `solution3.lst` with a report from processing of MongoDB script `solution3.js` with the implementation of indexing, listing the indexes and query processing plans, and dropping the indexes.

And again, please remember that:
- a report without the listings of applied methods and feedback messages issued by MongoDB scores no marks,
- a report that contains any kind of processing errors scores no marks.

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/CSCI235 (S220) Database Systems**

(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.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