**Gage Thompson**

**Assignment 4**

**Part 1 (Media Database)**

**I.** Createa collection named “**Media**” in the **sample\_mflix** database. Use the commands in “4\_Assignment\_Mongo\_1\_CreateDoc.txt” to insert some documents into this collection.

**II.** Based on the data you just created, write commands to find the following required information and display it.

Query 1: Find all documents with the media type of “CDs”;

1. db["Media"].find({Type: "CD"}, {Title: 1, Type: 1})

Query 2: Find the books with a length between 200 and 300 pages, and return the top 5 longest ones;

1. db["Media"].find({Page: {$gte: 200, $lte: 300}}, {Title: 1, Page: 1}).sort({Page: -1}).limit(5);

Query 3: Find all the books published by either “Apress” or “O'Reilly Media”;

1. db["Media"].find({ Publisher: { $in: ["Apress", "O'Reilly Media"] } }, { Title: 1, Publisher: 1})

Query 4: Find all the books published by “Apress” in 2014. And display their “Title”, “Publisher”, and “Year”;

1. db["Media"].find({$and: [{Publisher: "Apress"}, {Year: 2014}]}, {Title: 1, Publisher: 1, Year: 1, \_id: 0})

Query 5: Find all the CDs with the New Price not greater than $16;

1. db["Media"].find({"Price.New": {$lte: 16}}, {Title: 1, "Price.New": 1, \_id: 0})

Query 6: Find all the books with either "David Hows" or "Peter Membrey" (or both of them) as the authors;

1. db["Media"].find({Author: {$in: ["David Hows", "Peter Membrey"]}}, {Title: 1, Author: 1, \_id: 0})

Query 7: Update data: For each book published by “Apress”, sets its price with a value of $100;

1. db["Media"].updateMany({Publisher: "Apress"}, {$set: {Price: 100}})

**Part II: What to Do**

**I.** Create a collection named “**EnronEmails**” in the **sample\_mflix** database. The file Enron.json (on ELC) contains all the document data. Import all the documents in Enron.json into the collection **EnronEmails** (there should be 5,929 documents);

**Write commands to complete the following tasks:**

**II.** Develop a single Aggregation query to find all senders and the total number of emails that each of them sent;

db["EnronEmails"].aggregate([

{$group: {\_id: "$sender", totalEmailsSent: {$sum: 1}}},

{$project: {Sender: "$\_id", totalEmailsSent: 1,\_id: 0}}

]);

**III.** Develop a single Aggregation query to complete the following tasks. (Note that you need to develop a single query rather than separate queries):

- First, find all emails sent after 2000-10-20;

- Next, among the emails sent after 2000-10-20, find the earliest email sent by each of the unique senders;

- Next, list each of the unique senders and his/her earliest email date. The list should be in a descending order of the earliest email dates.

db["EnronEmails"].aggregate([

{$match: {date: {$gt: "2000-10-20"}}},

{$group: {\_id: "$sender", earliestEmailDate: { $min: "$date" }}},

{$sort: {earliestEmailDate: -1}},

{$project: {Sender: "$\_id", earliestEmailDate: 1, \_id: 0}}]);