This submission template is a convenient document for you to provide the screenshots and explanations for Assignment 5.1. This submission template is intended to be used in conjunction with the Assignment 5.1 Instructions document. The instructions document illustrates how to correctly execute each SQL construct, explains important theoretical and practical details, and contains the complete set of instructions on how to complete this lab.

**Name**: Mi Gao

**Date: 08/15/2020**

**Section 1**

**Question 1a):** What would you implement as a 1:Many documents through linking? List two document names and what is the link between them. (short answer).

I will use Linking Representation for sure because their have too many **movie persons** who show in many different **dvd**.

**Question 1b):** What would you implement as a 1:Few document through embedding List the document name and what would be embedded within the document. (short answer).

I thought it should be **payment** document. I do not really believe there are going to have too many different ways of payment methods under each customer’s account.

However, in the files you provided, we do not have it yet. So that I think it should be **genres** in **DVD because genres can be at least one but not too many for a movie.**

**Question 1c):** What would you implement as a 1:1 relationship through embedding within the movie rental database? (short answer)

Based on what I did in past assignments, I would like to set 1:1 relationship between **rental** table and **dvd\_copy** table which they will become to collection in mongoDB. The reason is only one dvd copy has been tracked in single rental record.

However, because of the same reason above, I think it should be **rating** in **DVD**. No more than one but at least one rating level will have for a movie.

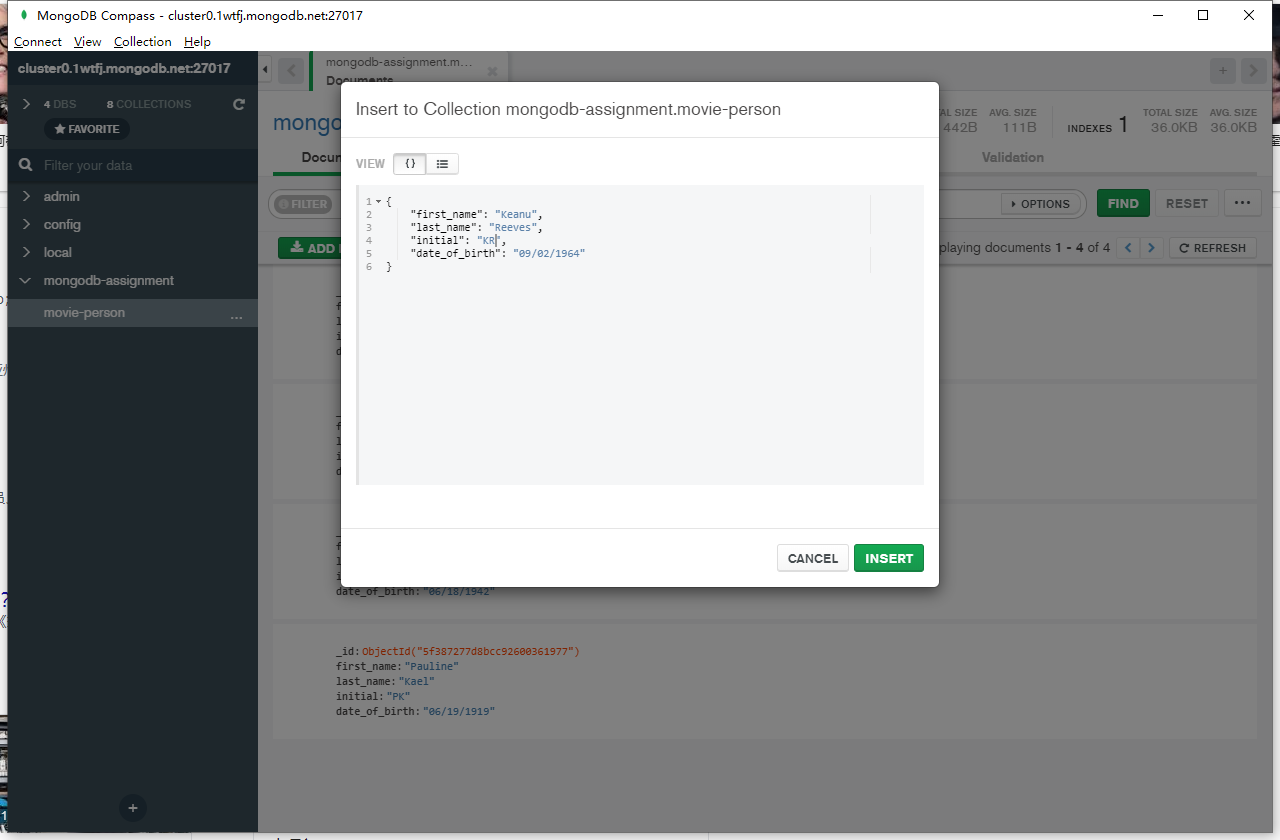
**Section 3**

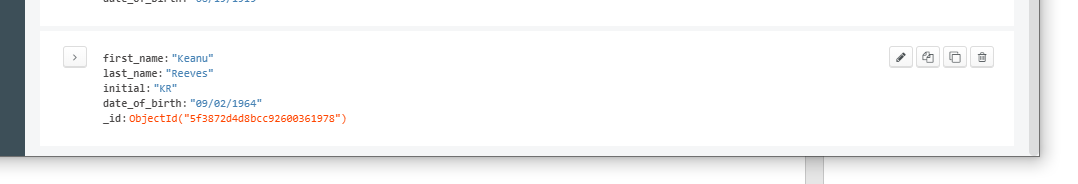
**Question 3a):**

Import the rest of the movie people by using either the copy paste method or importing the JSON document.

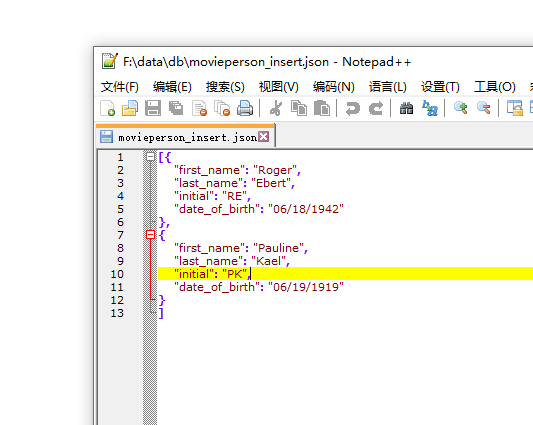
Paste screenshot showing both copy and paste or importing the JSON document and the inserted data result. The screenshot of the result should show how many total documents are in the movie-person collection.

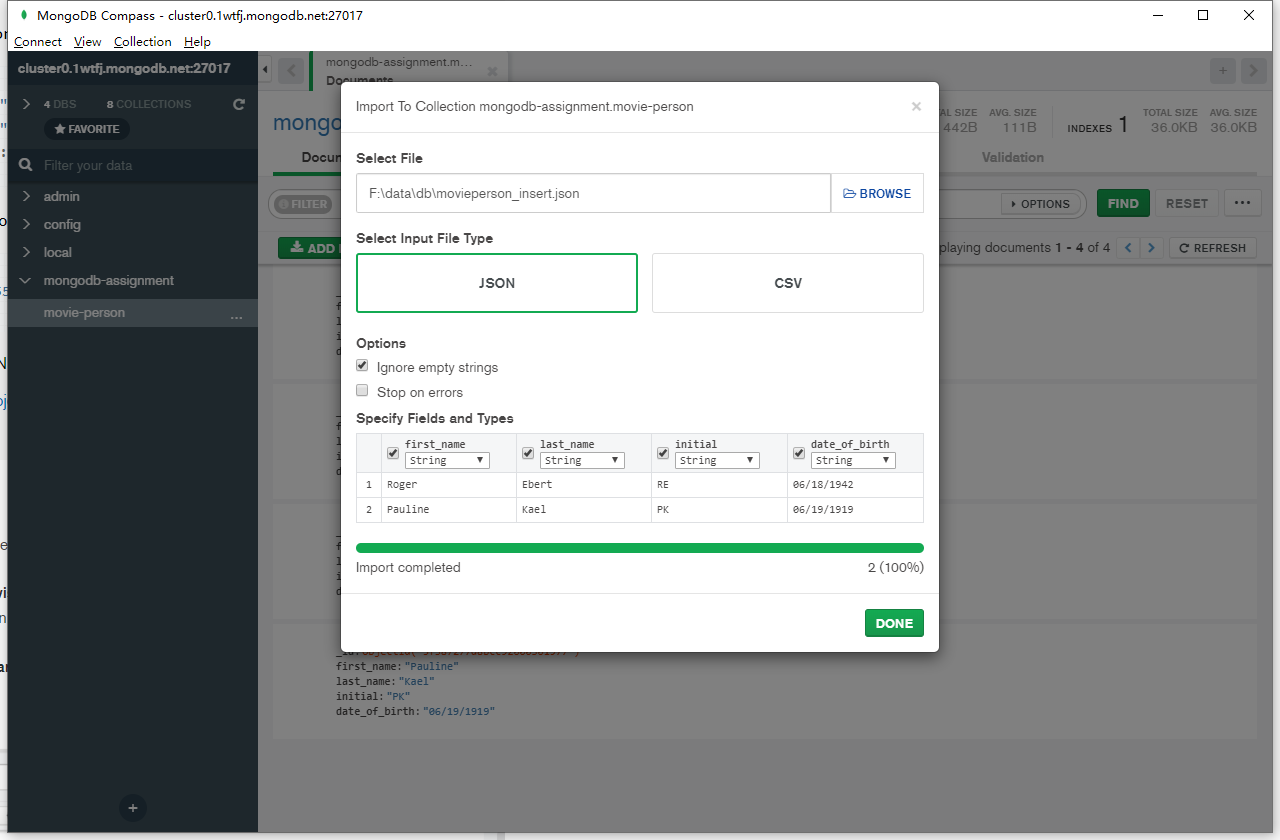
Copy and paste:



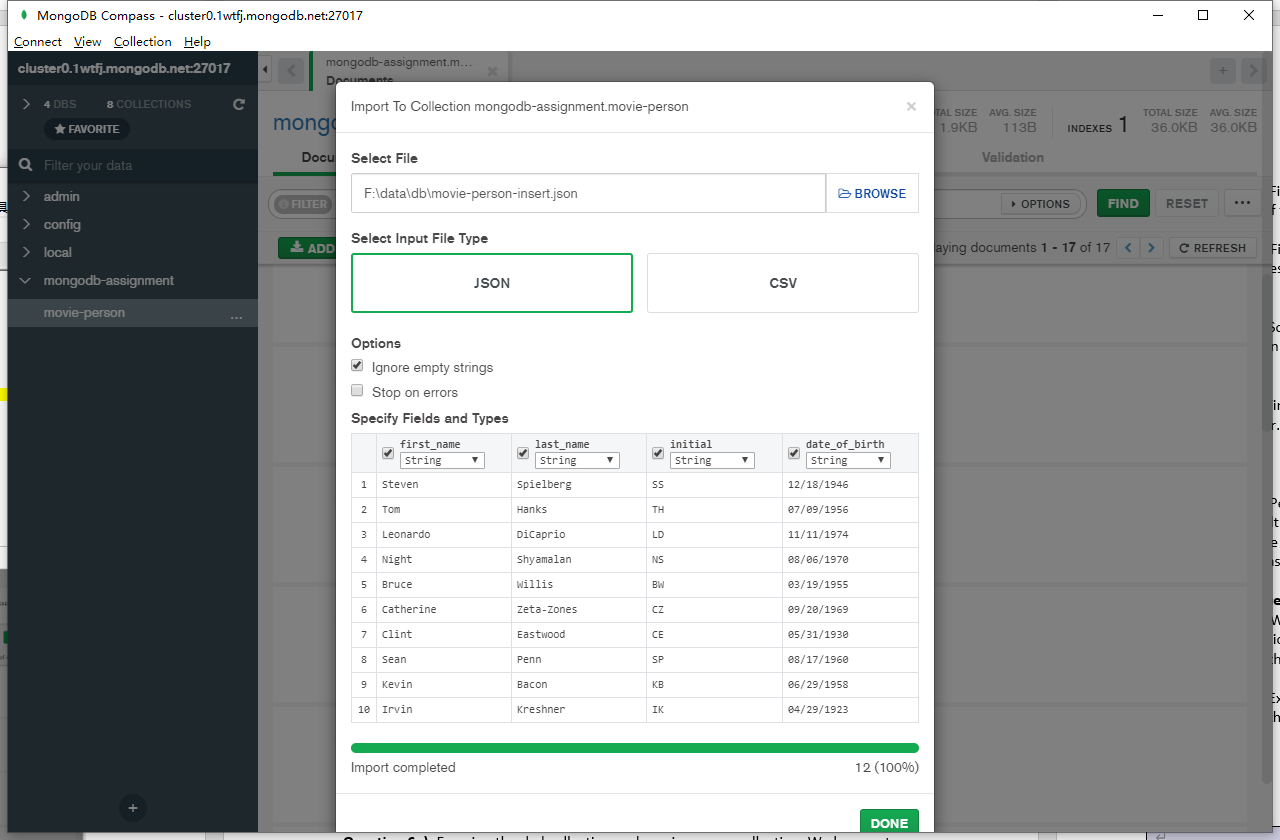


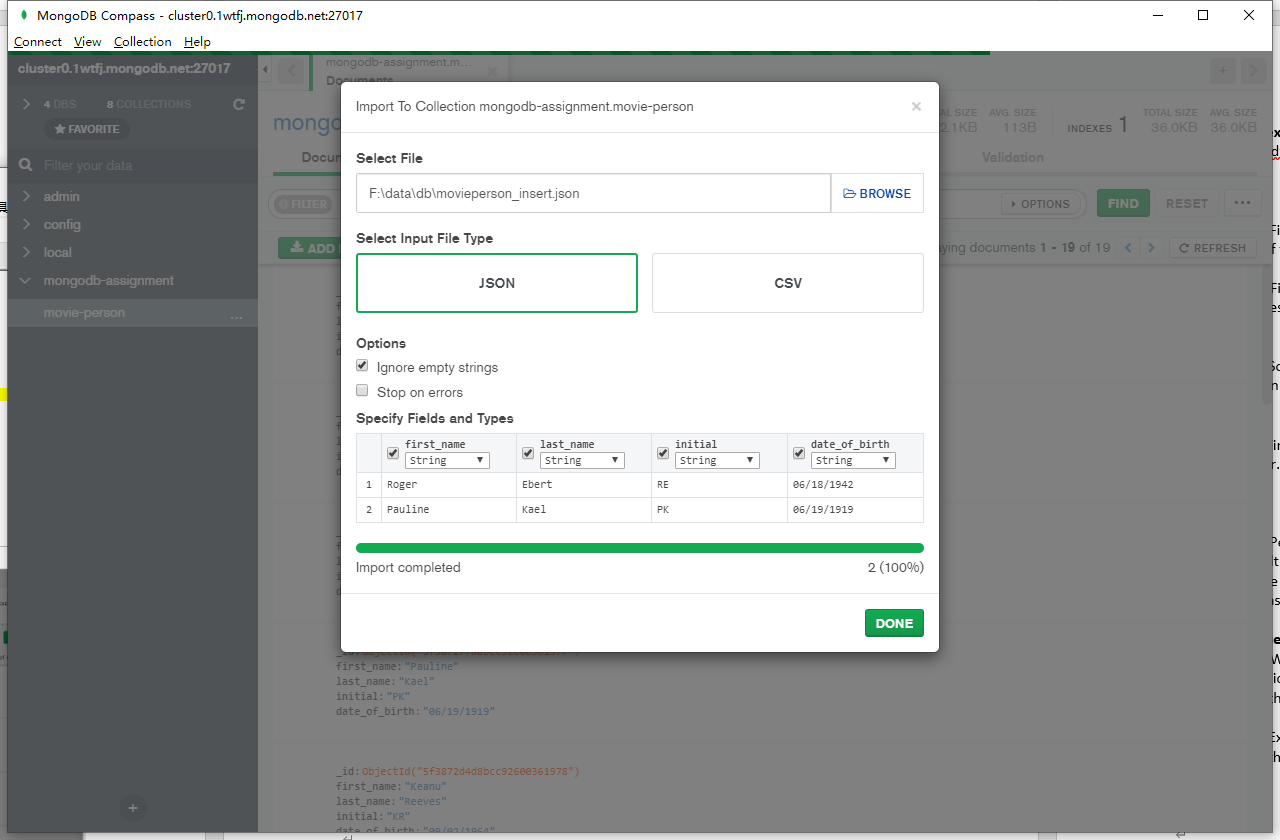
The JSON file create by myself trying:

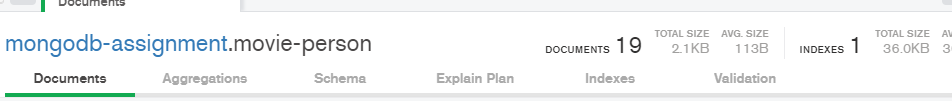




Files you provided:

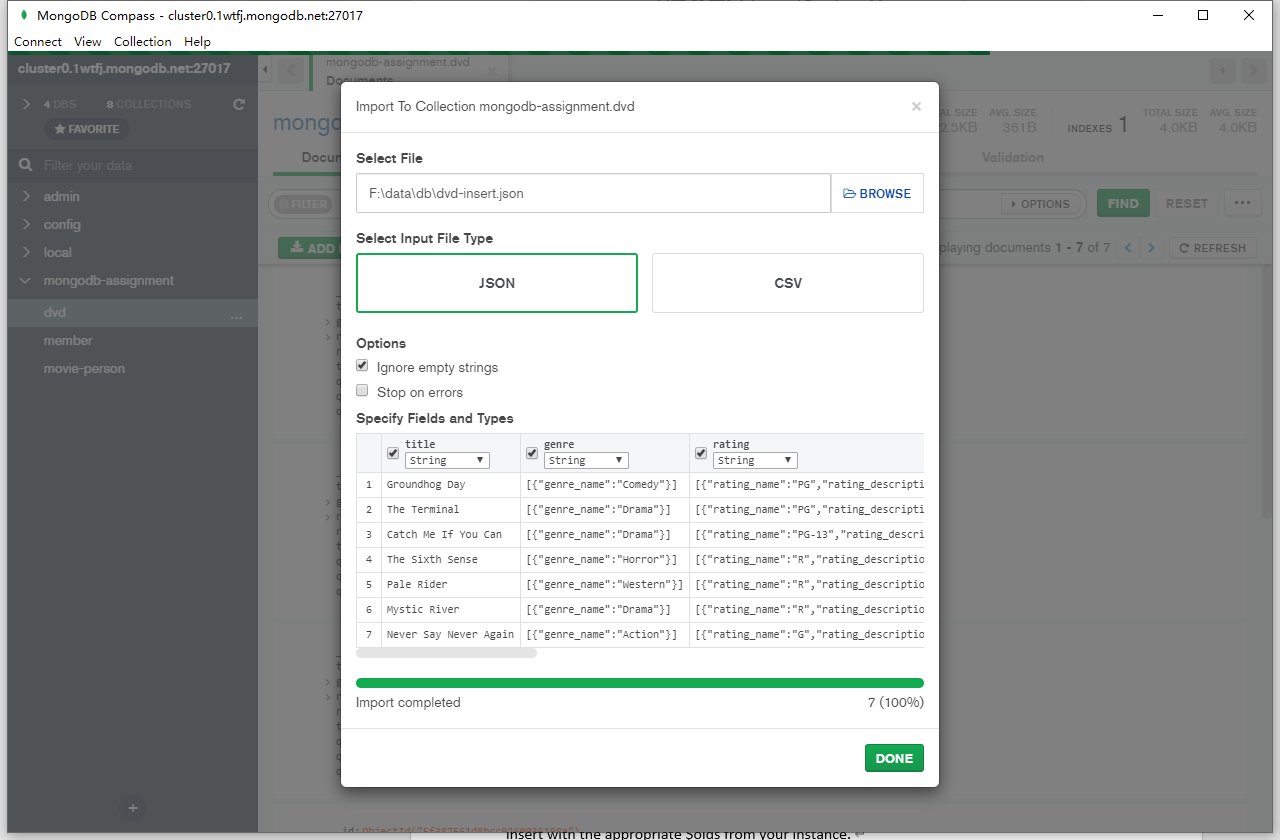


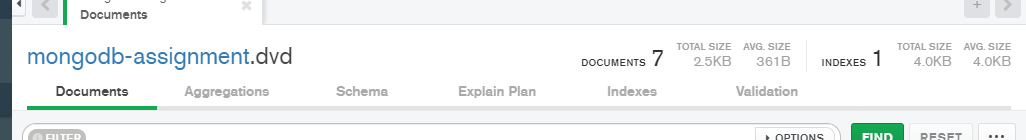


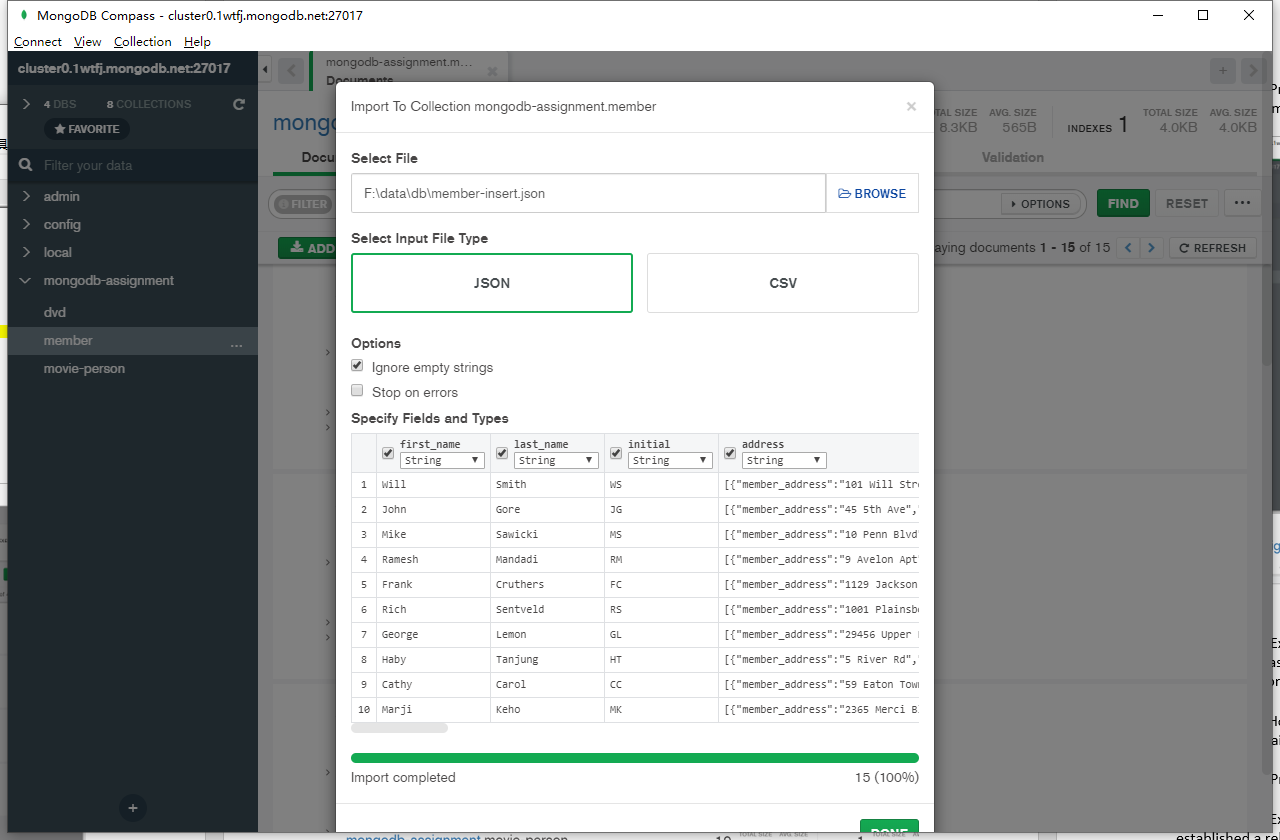


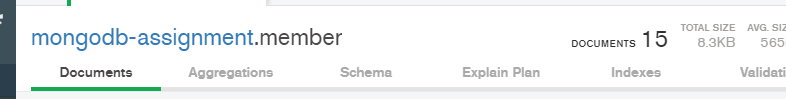
**Section 6**

**Question 6a):** Provide screen-shots of both dvd and member collections loaded with the documents from the provided JSON files, make sure to show the record counts.









**Question 6b):** Examine the DVD document and specify how the Genre and Rating relationship was implemented, was it through embed or linking? What is the cardinality of these relationships and does it correctly support the design choice? (short answer)

Both are embed. The cardinality is all the one because I see only one kind of genre and only one rating level for single movie. The setting of rating is reasonable I thought because only one kind of level can be for single movie. The setting of genre I thought it can be more than one so that to be linking because I knew the movie can have more than one genre. However, it’s okay so far. I hope this is what you want me to say about my understanding about the cardinality in here.

**Question 6c**): How are Member, DVD and rental document related? (through embdding or linking, explain your answer brifly)

**Many:Many** relationship use **linking** between those three because you can see the structure of rental:

{

"member\_id": {

"$oid": ""

},

"dvd\_id": {

"$oid": ""

},

"rental\_request\_date": "",

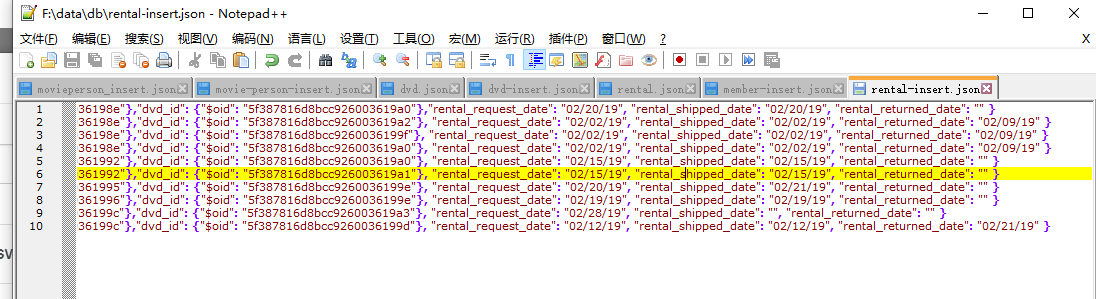
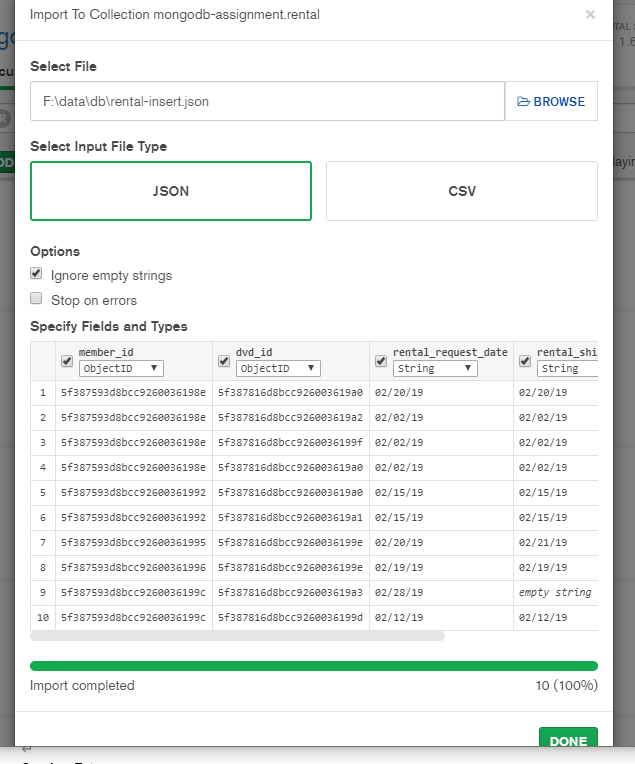
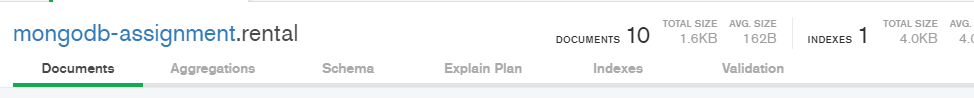
"rental\_shipped\_date": "",

"rental\_returned\_date": ""

}

The rental is the bridge of member and DVD.

**Question 6d):** Provide a screenshot of the populated rental collection

**Question 6e):** Examine the dvd collection and movie-person collection. We have not established a relationship between them. Briefly describe how would you establish the relationship i.e. via linking or embedding and what kind of a relationship is it.

DVD and movie-person should have linking relationship which would be **1:Many**. The structure should be like this in DVD:

{

"\_id": "",

"title": "",

"genre": ["genre\_1", "genre\_2"],

"rating": [{

"rating\_name": "",

"rating\_description": ""

},{

"rating\_name": "",

"rating\_description": ""

}],

"release\_date": "",

"theatre\_release\_date": "",

"quantity\_on\_hand": "",

"quantity\_on\_rent": "",

"quantity\_lost": "",

"movie\_person": [{

"movie\_person\_id": "",

"role\_name": []

}, {

"movie\_person\_id": "",

"role\_name": []

}]

}

**You will see it clearly; the id of movie-person will be linked within DVD, and their role also.**

**Question 6f - extra credit):** Create a linked collection for DVD and Movie-Person called movie-person-dvd and populate it.

{

"dvd\_id": {

"$oid": "5f387816d8bcc9260036199d"

},

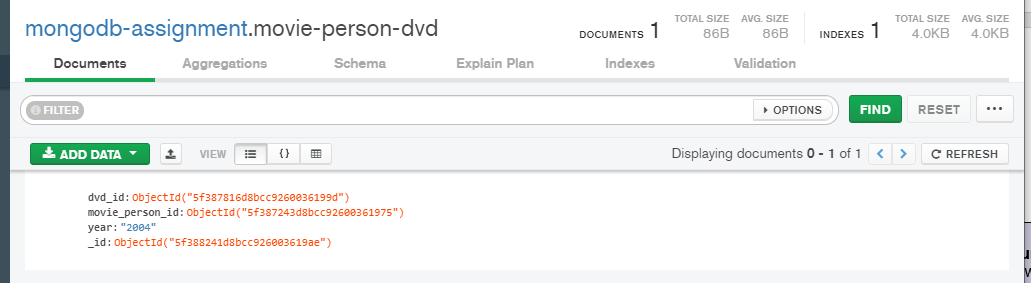
"movie\_person\_id": {

"$oid": "5f387243d8bcc92600361975"

},

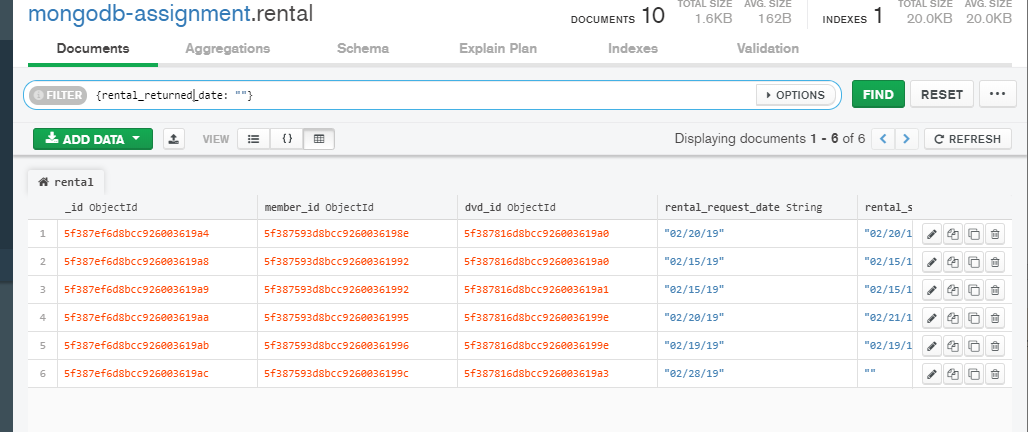
"year": "2004"

}

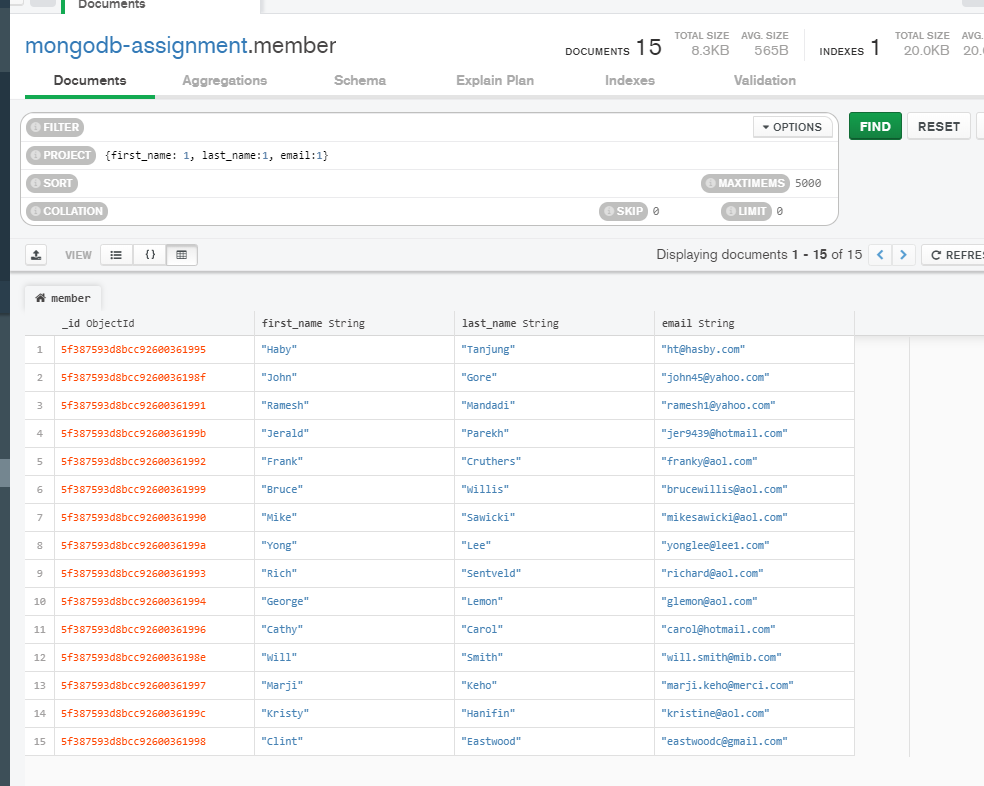


**Section 7**

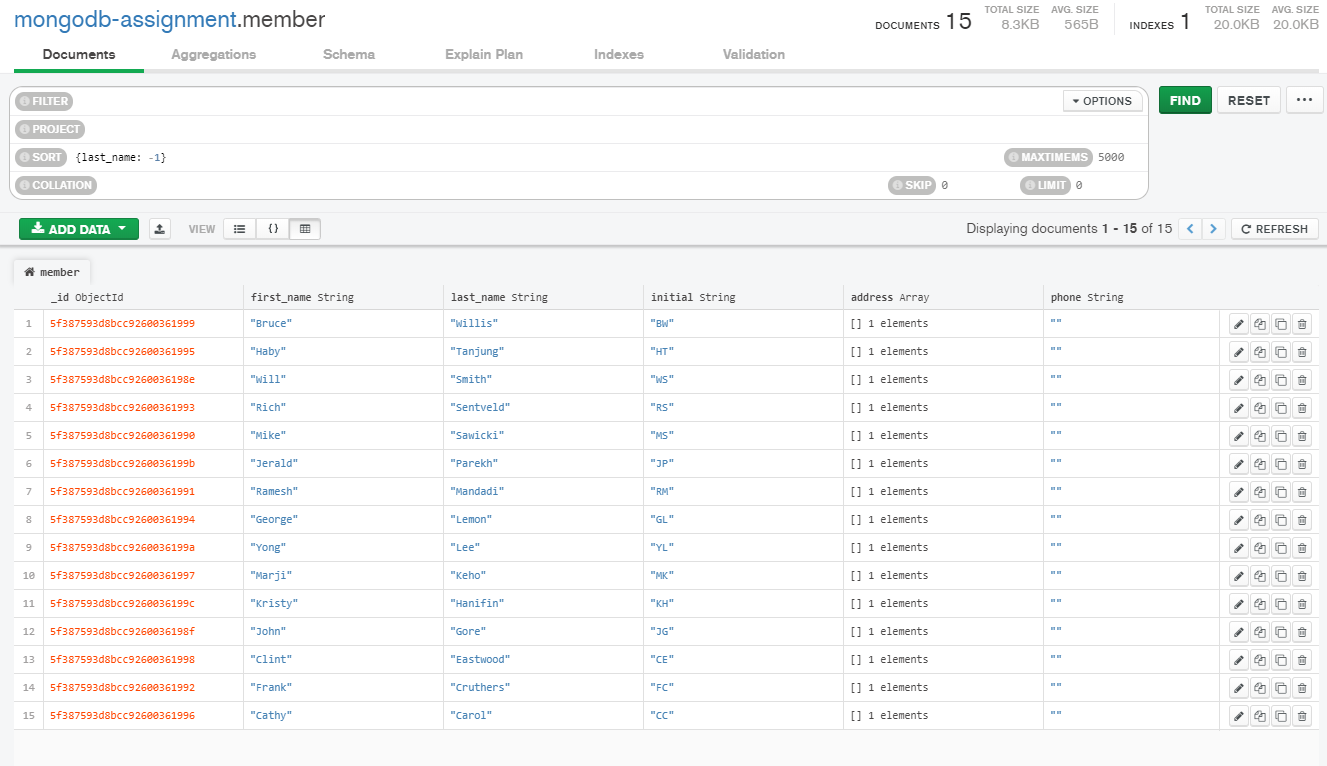
**Question 7a):** Find all the rental documents for DVDs that have not been returned. Take a screenshot of the result.



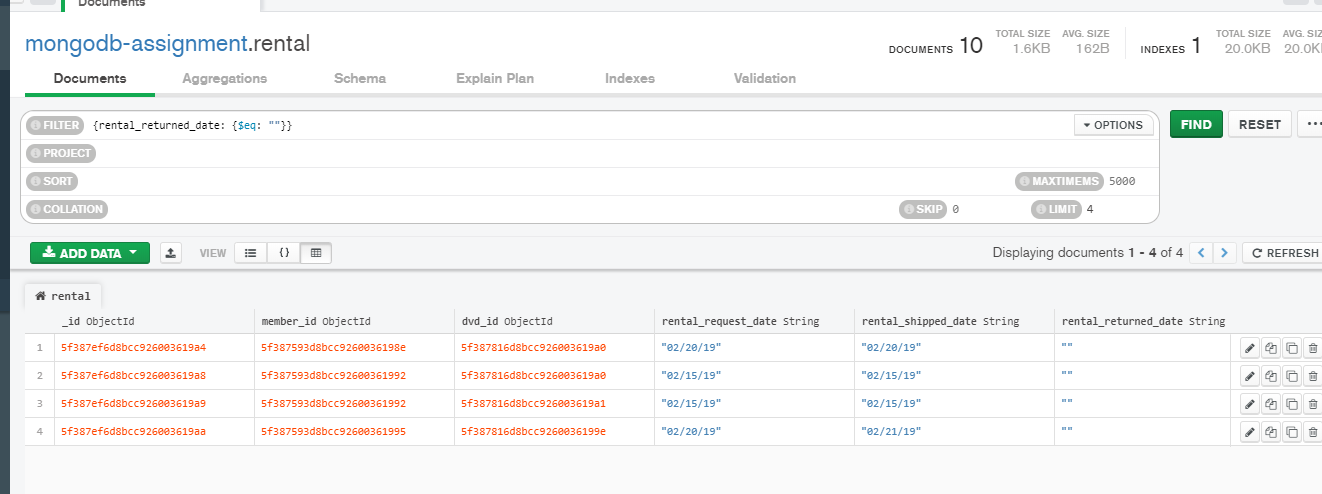
**Question 7b):** Find the first name, last name and email of all the members. Please make sure that the result does not have the \_id field. Take a screenshot of project command the result.



**Question 7c):** Sort the documents in the member collection based on the last name of the members in descending order. Take a screenshot of the sort command and the result.

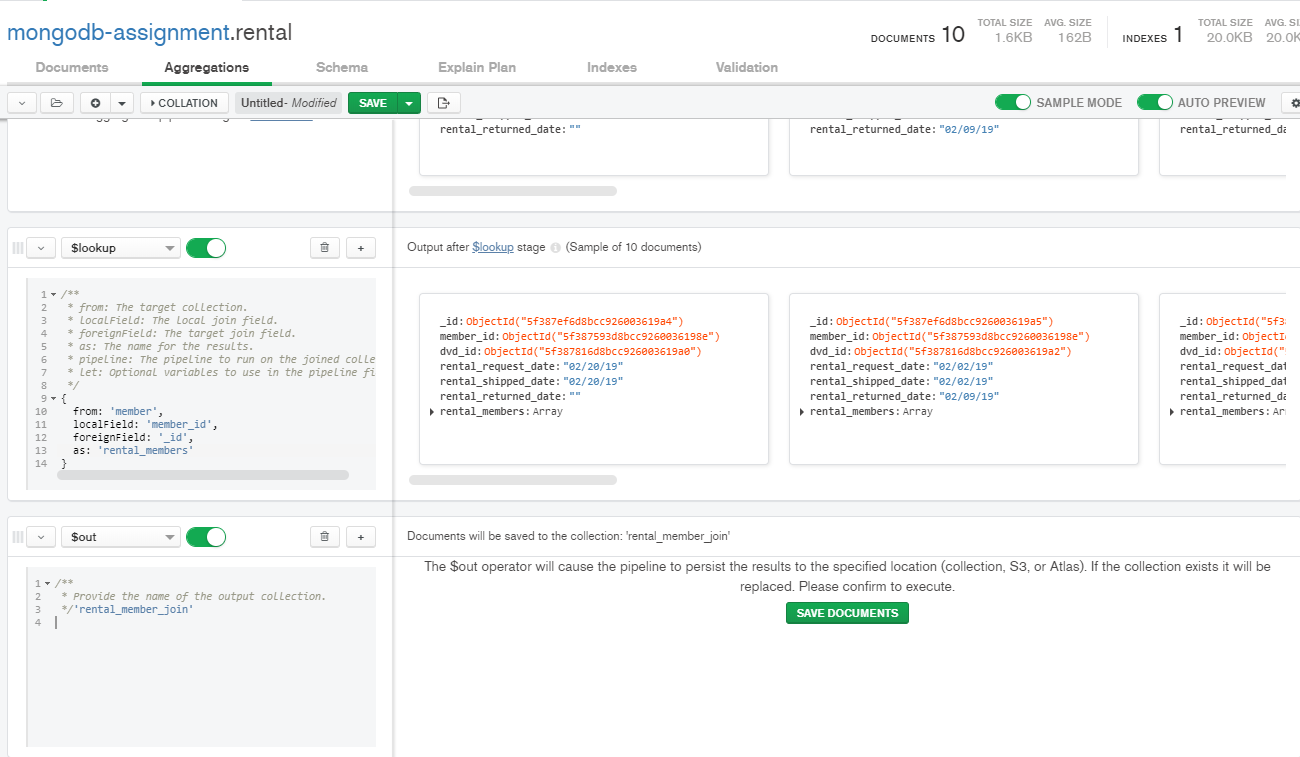


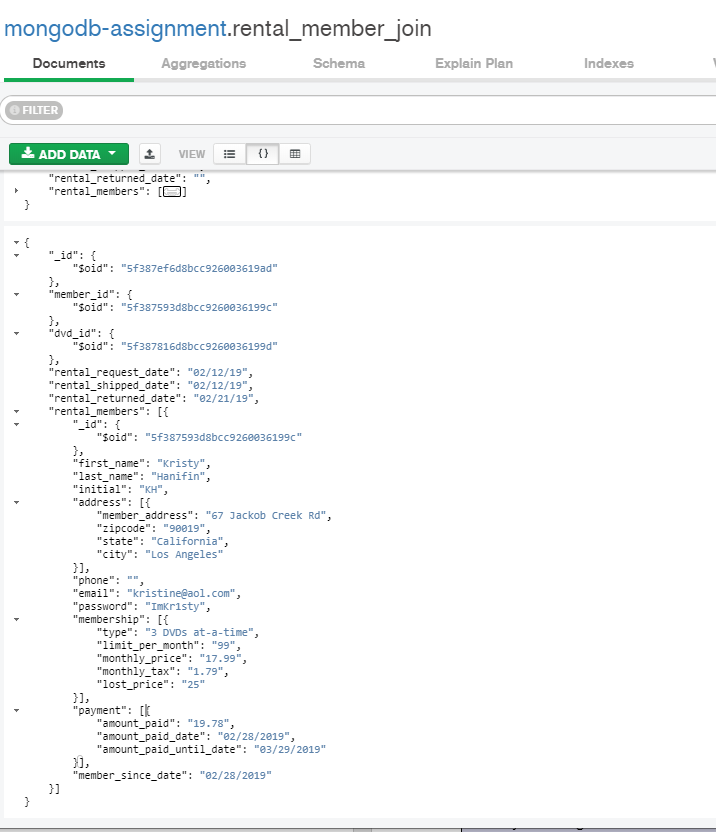
**Question 7d)** Find 4 documents of the rental collection where the DVD is not returned by the member. Take a screenshot of the result and attach it to the final submission.



**Section 8**

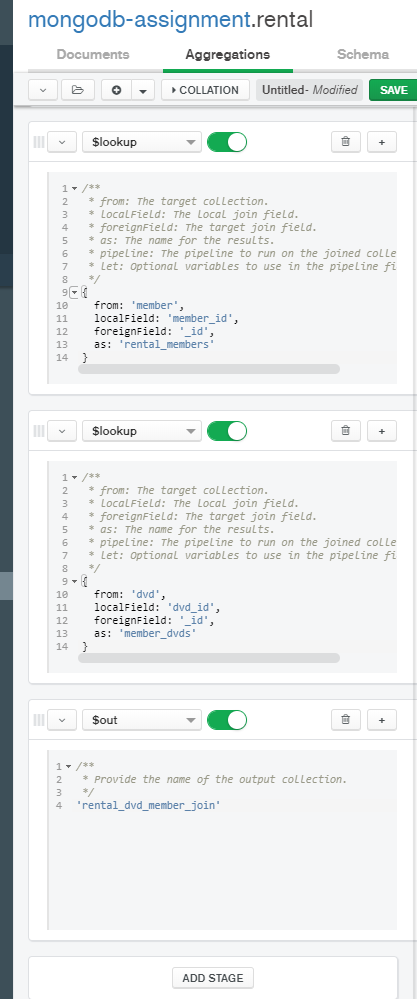
**Question 8a):** Perform a left outer join between the rental and member collection and name the resulting collection as **rental\_member\_join**. Take a screenshot of the JSON structure of the last document (i.e. 11th document) from the rental\_member\_join collection. Please expand all the fields so that the values are visible clearly.

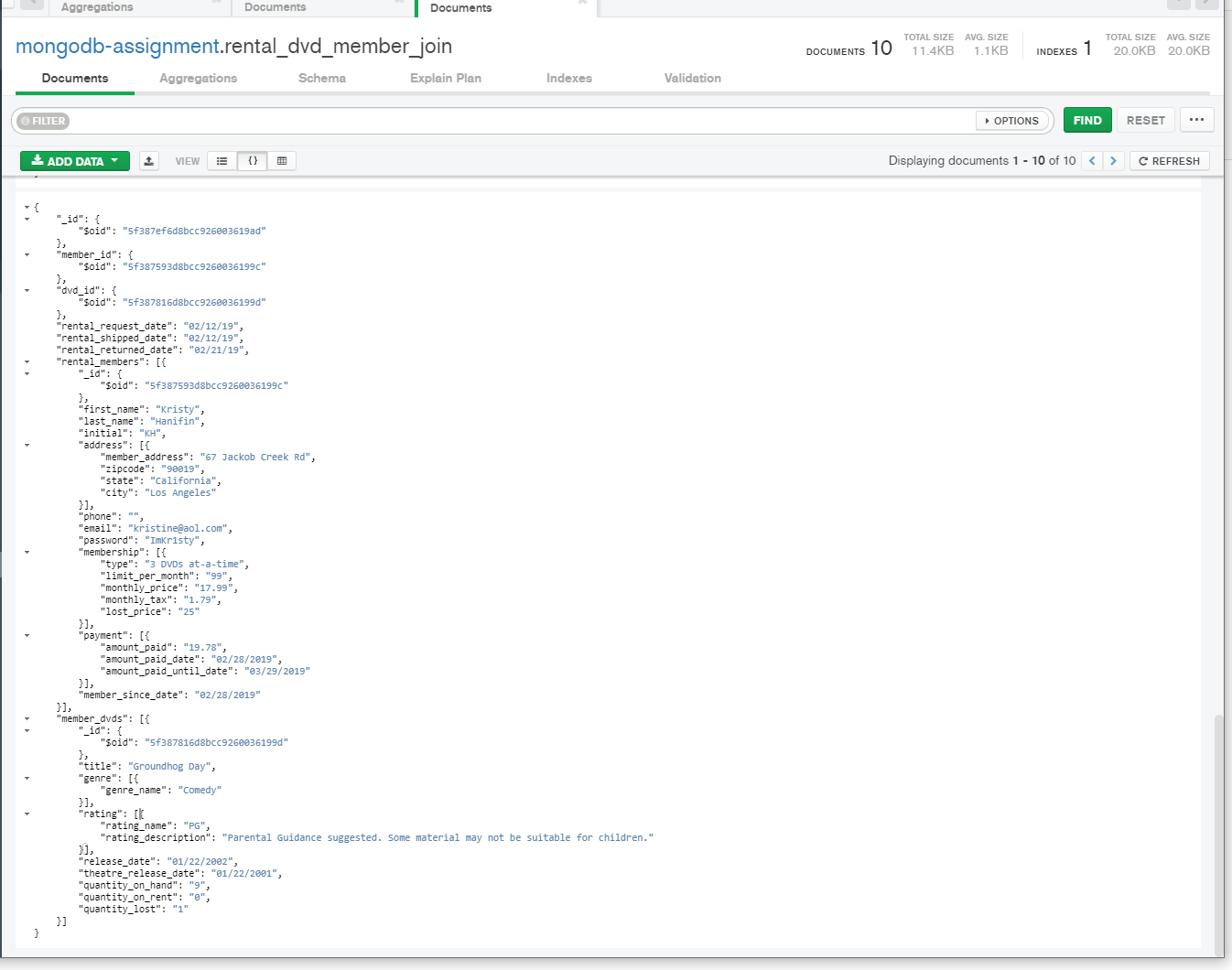




**Extra credit questions:**

**Question 8b):** We want to access the dvd and member data together for each rental document. Which collections would you join to achieve this? Perform the join and take a screenshot of the resulting collection.





**Question 8c):** Examine the resulting collection from Question 9. Briefly describe the problem with this collection.

I am not sure what is the question 9. I assume you was saying about question 8b. I thought the results from the document maybe redundancy or not.

**Conclusion**:

This is amazing experience of using mongoDB. I never tried to use GUI tool when I was using mongoDB. It is more humanity User-friendly and intuitive. The point of how the aggregations work is especially surprised me.

Use the **Ask your Facilitator Discussion Board** if you have any questions regarding the how to approach this assignment.

Save your assignment as ***lastnameFirstname\_lassignment5.1.doc*** and submit it in the *Assignments* section of the course.

For help uploading files please refer to the *Technical Support* page in the syllabus.