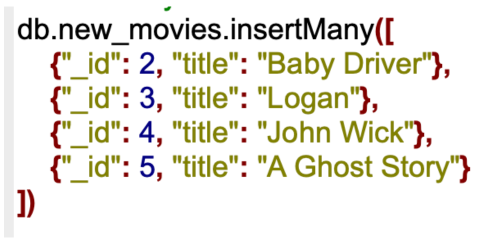
In Class Quiz – Week 6

Section I

1. In MongoDB, updateOne() and updateMany() can be used to add a new data field, different from the UPDATE statement in SQL.
True
False
Answer: True

2. In MongoDB, when using insertOne() to add a new document and an _id field is not specified, the database autogenerates an ID.
True
False
Answer: True

3. In MongoDB, when using the following code to add multiple documents to the new_movies collection, if one document with _id = 4 already exists in the collection, how many documents will be added to this collection?



```
db.new_movies.insertMany([
   {"_id": 2, "title": "Baby Driver"},
   {"_id": 3, "title": "Logan"},
   {"_id": 4, "title": "John Wick"},
   {"_id": 5, "title": "A Ghost Story"}
])
```

A. 1
B. 2
C. 3
D. 4
Answer: B

4. In MongoDB, the _id field can't be changed with updateOne(). We can only use replaceOne() to replace a current document with a new document that contains a different _id.
True
False
Answer: False

5. In MongoDB, which of the following is true regarding findOneAndDelete() and its differences with deleteOne()?
A. Projection can be used to include or exclude fields from the document in response.
B. In the case of multiple document matches, the sort option can be used to influence which document gets deleted.
C. Once deleted, it returns the deleted document as a response.
D. All the above
Answer: D

6. In MongoDB, replacing a document is a more effective approach compared to deleting and re-inserting a new document.
True
False
Answer: True

7. Which of the following SQL statement can be used for the same purpose as $unset does in MongoDB?
A. UPDATE
B. ALTER TABLE
C. DELETE FROM
D. INSERT INTO
Answer: B

8. _________ calculates aggregate values for the data in a collection.
A. db.collection.aggregate
B. db.collection.agg
C. db.collection.pipeline
D. All the above
Answer: A

9. For MongoDB Aggregation, point out the wrong statement.
A. Some pipeline stages may generate new documents or filter out documents
B. Pipeline stages do need to produce one output document for every input document
C. Pipeline stages can appear multiple times in the pipeline
D. None of the above
Answer: B

10. Which of the following pipeline is used for aggregation in MongoDB?
a) data processing
b) information processing
c) knowledge processing
d) none of the mentioned
Answer: A, Explanation: MongoDB can return counts of the number of documents that match a query, or return the number of distinct values for a field.

11. Which of the following options are array operators in MongoDB?
A) $project
B) $group
C) $arrayElemAt
D) $match
Answer: C) $arrayElemAt

12. In which of the following ways can the aggregation pipeline output results?
A) In a collection
B) Returning a cursor to the result set

C) Inline as a document containing the result set
D) All of the above
Answer: D

13. The following aggregation is correct:

```
db.users.aggregate([
   {$match: {
      $or: [{"name": "Catelyn Stark"}, {"name": "Ned Stark"}]
   }},
   {$lookup: {
      from: "comments",
      localField: "name",
      foreignField: "name",
      as: "comments"
   }},
   {$unwind: "comments"},
   {$limit: 3},
])
```

True
False
Answer: False

Section 2

14. Come up with a business scenario for using Upsert

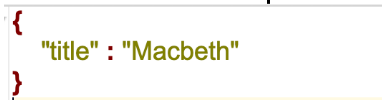
Answer: (open ended)
In real-world scenarios, you will mostly be doing these operations in large numbers. Consider that your system receives daily updates from a user server, where the server sends you all the documents that were modified during the day. These daily updates might include records of the new users signed up with the server as well as changes to the existing users' profiles. On a large-scale system, performing a two-step update or insert operation for each of the records will be very time-consuming and error prone. However, having a dedicated command, you can simply prepare and execute an upsert command for each of the records you receive and let MongoDB do the update or insert.

15. What does this code do? What will be the output?

```
db.movies.insertMany([
   { "_id": 1011, "title": "Macbeth" },
   { "_id": 1513, "title": "Macbeth" },
   { "_id": 1651, "title": "Macbeth" },
   { "_id": 1819, "title": "Macbeth" },
   { "_id": 2117, "title": "Macbeth" }
])
```

```
db.movies.findOneAndReplace(
   {"title": "Macbeth"},
   {"title": "Macbeth", "latest": true},
   {sort: {"_id": -1}, projection : {"_id": 0}}
)
```

Answer: Find the last movie we added. Replace it with a new document. Print out the document before replace.



```
{
    "title" : "Macbeth"
}
```