Experiment 1:

Using MongoDB, create a collection called transactions in database usermanaged (drop if it already exists) and bulk load the data from a json file, transactions.json. Upsert the record from the new file called transactions_upsert.json in Mongodb.

Initial Settings:

- **In command prompt , change the path to where your bin folder is located , there you type mongod.. This runs the daemon process**

  ```
  C:\Users\DELL>cd C:\ProgramFiles\MongoDB\Server\8.0\bin

  C:\Program Files\MongoDB\Server\8.0\bin>mongod
  ```

- **Open another command prompt (to run the client/shell)**
- **C:\Users\DELL>cd C:\ProgramFiles\MongoDB\Server\8.0\bin**
- **C:\Program Files\MongoDB\Server\8.0\bin>mongsh**

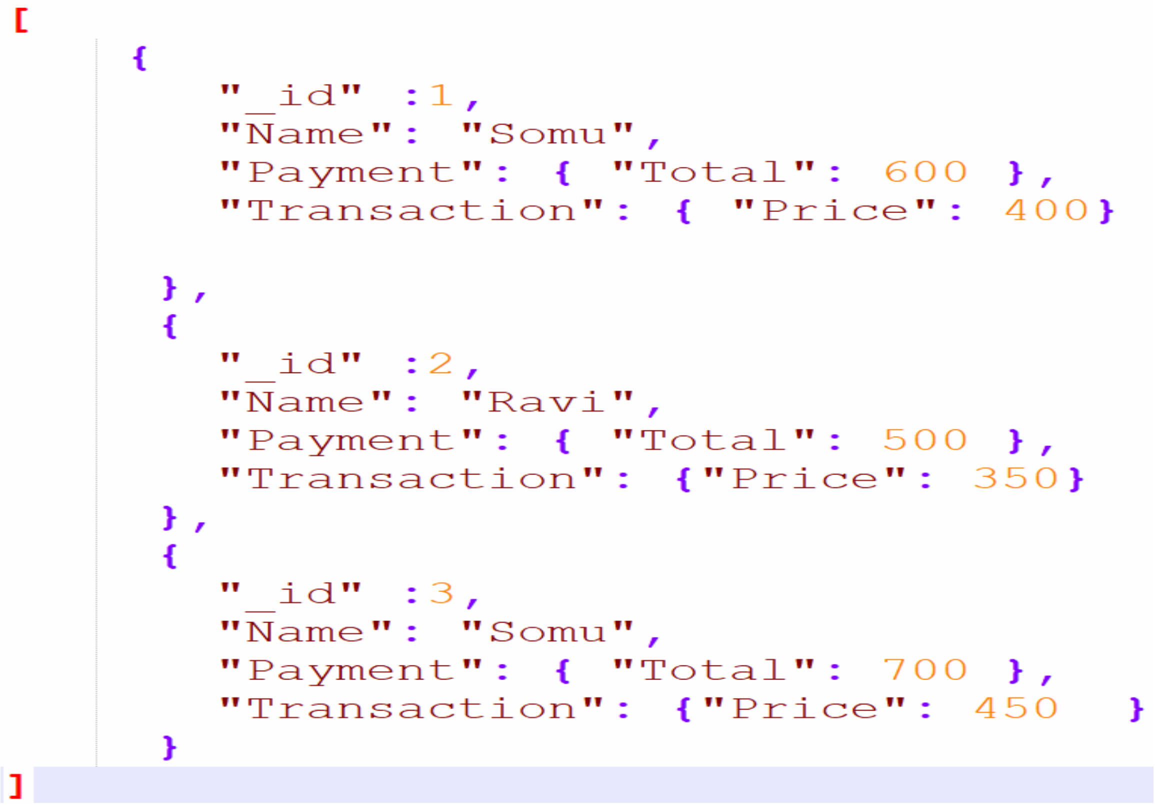
Execution steps:

1: create two json file transactions.json and transactions_upsert.json containing set of documents.

2. Execute following mongodb commands in mongo shell.

```
// Step 1: Connect to MongoDB and switch to the usermanaged database

use usermanaged

// Step 2: Drop the transactions collection if it already exists

db.transactions.drop()

// Step 3: Create a new transactions collection

db.createCollection("transactions")

// Step 4: Bulk load the data from transactions.json into the transactions collection

mongoimport --db usermanaged --collection transactions --file transactions.json --jsonArray

// Step 5: Upsert the records from transactions_upsert.json into the transactions collection

mongoimport --db usermanaged --collection transactions --file transactions_upsert.json --jsonArray --mode upsert
```

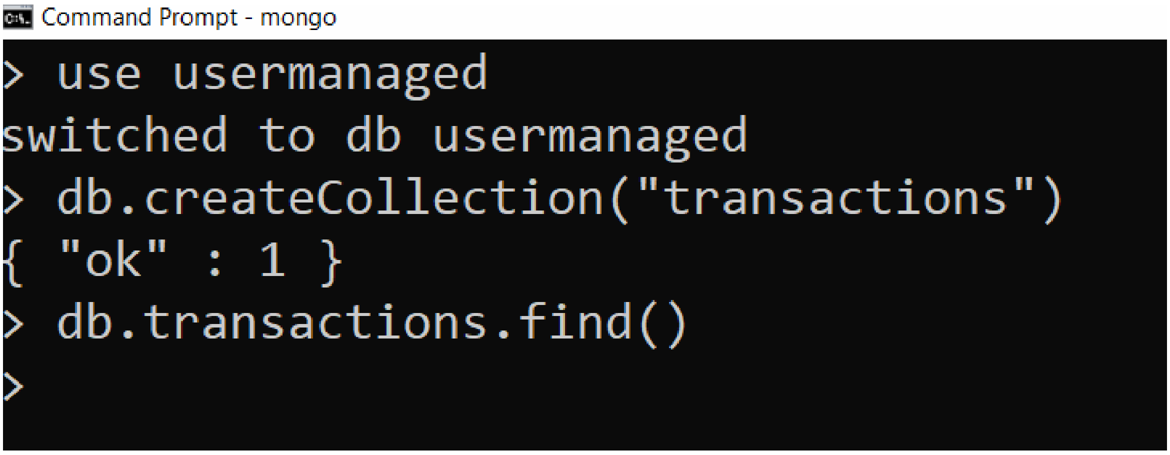
1. Create a transactions.json



```
[
     {
        "_id" :1,
        "Name":  "Somu",
        "Payment": { "Total": 600 },
        "Transaction": { "Price": 400}

      },
      {
        "_id" :2,
        "Name":  "Ravi",
        "Payment": { "Total": 500 },
        "Transaction": {"Price": 350}
      },
      {
        "_id" :3,
        "Name":  "Somu",
        "Payment": { "Total": 700 },
        "Transaction": {"Price": 450   }
      }
]
```

2. Open mongo shell and enter the following

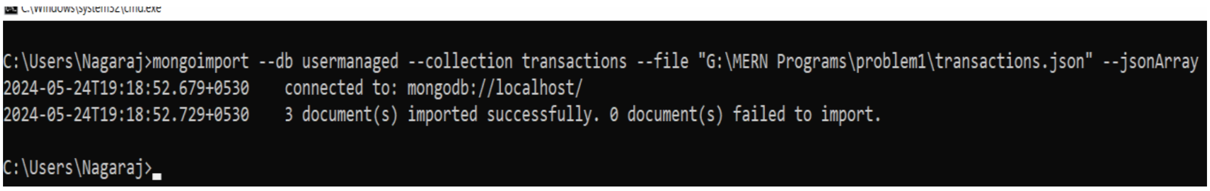
Before loading data into the collection transactions, mongo shell prompts as there are no documents in it



```
> use usermanaged
switched to db usermanaged
> db.createCollection("transactions")
{ "ok" : 1 }
> db.transactions.find()
>
```

3. Load bulk data from transactions.json to database using mongoimport tool, open the command prompt enter the following: (*Remember mongoimport tool works in windows powershell , so you should give the below commend in powershell*)

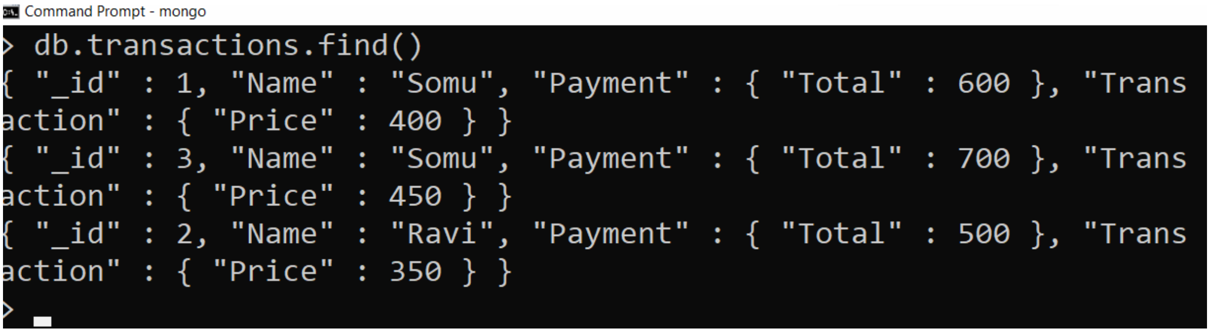
mongoimport - -db usermanaged - -collection transactions - -file “*copy paste the path of json file*” –jsonArray



```
C:\Users\Nagaraj>mongoimport --db usermanaged --collection transactions --file "G:\MERN Programs\problem1\transactions.json" --jsonArray
2024-05-24T19:18:52.679+0530    connected to: mongodb://localhost/
2024-05-24T19:18:52.729+0530    3 document(s) imported successfully. 0 document(s) failed to import.

C:\Users\Nagaraj>_
```

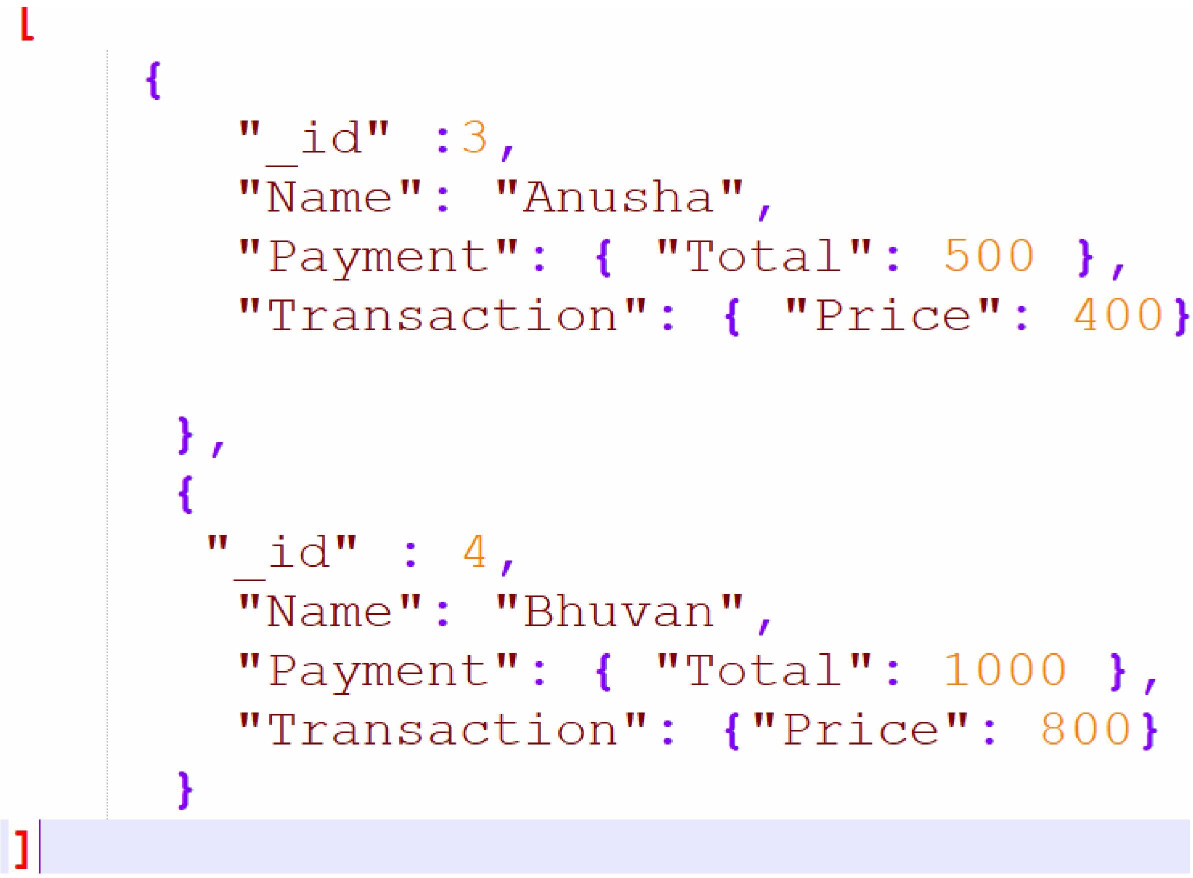
4. Documents in a collection after loading data.

db.transactions.find()



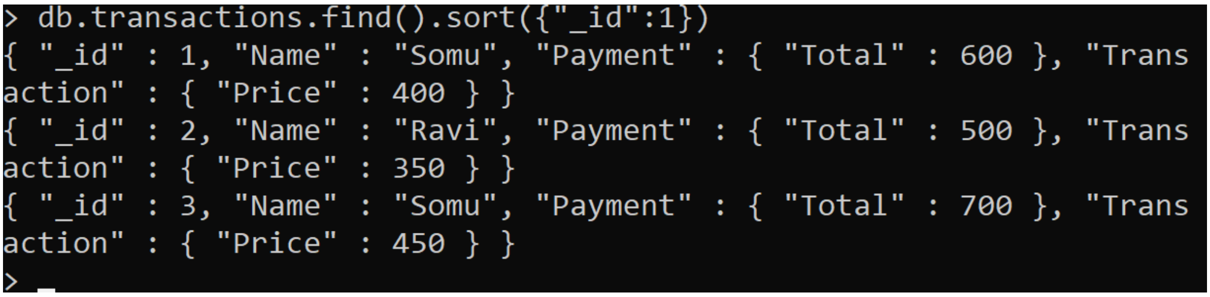
```
> db.transactions.find()
{ "_id" : 1, "Name" : "Somu", "Payment" : { "Total" : 600 }, "Trans
action" : { "Price" : 400 } }
{ "_id" : 3, "Name" : "Somu", "Payment" : { "Total" : 700 }, "Trans
action" : { "Price" : 450 } }
{ "_id" : 2, "Name" : "Ravi", "Payment" : { "Total" : 500 }, "Trans
action" : { "Price" : 350 } }
> _
```

5. create transactions_upsert.json file



```
[
    {
        "_id" :3,
        "Name": "Anusha",
        "Payment": { "Total": 500 },
        "Transaction": { "Price": 400}


     },
     {
       "_id" : 4,
        "Name": "Bhuvan",
        "Payment": { "Total": 1000 },
        "Transaction": {"Price": 800}
     }
]
```

6. transactions collection data before upsert and sort function sorts documents

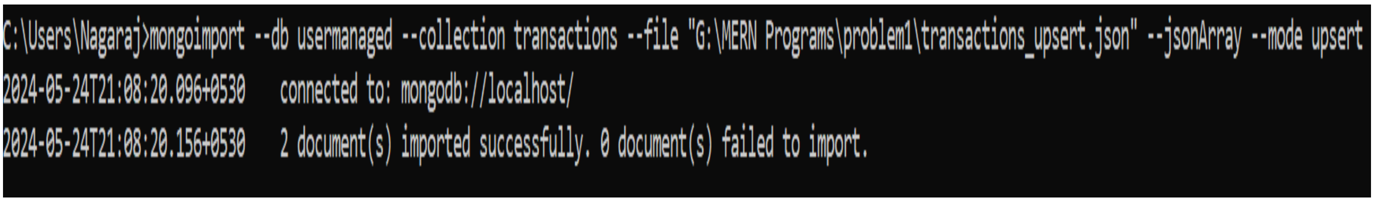


```
> db.transactions.find().sort({"_id":1})
{ "_id" : 1, "Name" : "Somu", "Payment" : { "Total" : 600 }, "Trans
action" : { "Price" : 400 } }
{ "_id" : 2, "Name" : "Ravi", "Payment" : { "Total" : 500 }, "Trans
action" : { "Price" : 350 } }
{ "_id" : 3, "Name" : "Somu", "Payment" : { "Total" : 700 }, "Trans
action" : { "Price" : 450 } }
> _
```

Here we are sorting in ascending order based on id _

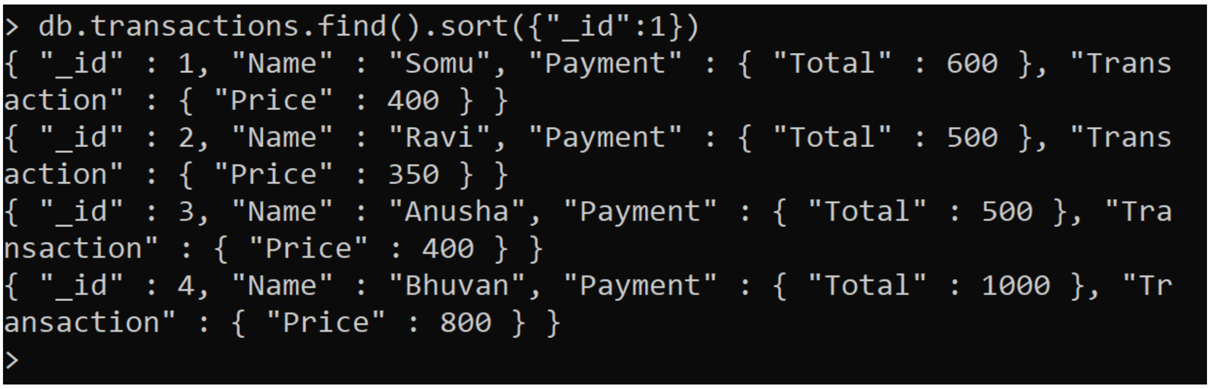
7. upsert the data from transactions_upsert.json

mongoimport - -db usermanaged - -collection transactions - -file "*copy paste the path of transactions_upsert.json file*" –jsonArray - -mode upsert



```
C:\Users\Nagaraj>mongoimport --db usermanaged --collection transactions --file "G:\MERN Programs\problem1\transactions_upsert.json" --jsonArray --mode upsert
2024-05-24T21:08:20.096+0530    connected to: mongodb://localhost/
2024-05-24T21:08:20.156+0530    2 document(s) imported successfully. 0 document(s) failed to import.
```

8. transactions collection data after upsert



```
> db.transactions.find().sort({"_id":1})
{ "_id" : 1, "Name" : "Somu", "Payment" : { "Total" : 600 }, "Trans
action" : { "Price" : 400 } }
{ "_id" : 2, "Name" : "Ravi", "Payment" : { "Total" : 500 }, "Trans
action" : { "Price" : 350 } }
{ "_id" : 3, "Name" : "Anusha", "Payment" : { "Total" : 500 }, "Tra
nsaction" : { "Price" : 400 } }
{ "_id" : 4, "Name" : "Bhuvan", "Payment" : { "Total" : 1000 }, "Tr
ansaction" : { "Price" : 800 } }
>
```

Experiment 2:

Query MongoDB with Conditions: [Create appropriate collection with necessary documents to answer the query]

a. Find any record where Name is Somu

b. Find any record where total payment amount (Payment.Total) is 600.

c. Find any record where price (Transaction.price) is between 300 to 500.

d. Calculate the total transaction amount by adding up Payment.Total in all records.

{

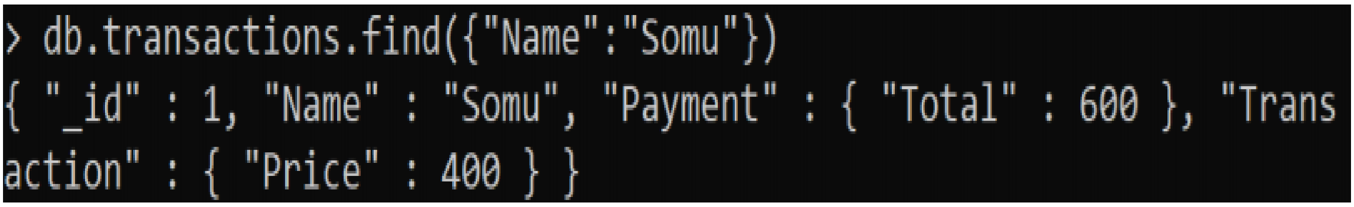
Note:You can use the same collection created in previous experiment i.e transactions.json or you can create new collection but make sure you have data to match with the query

}

Queries:

a. Find any record where Name is Somu

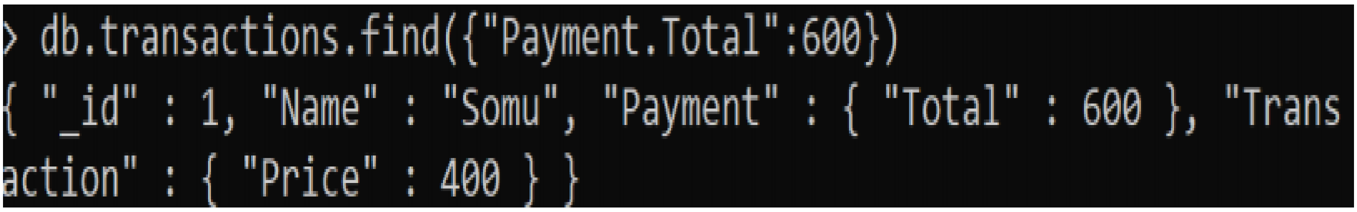
db.transactions.find({ "Name": "Somu" })



```
> db.transactions.find({"Name":"Somu"})
{ "_id" : 1, "Name" : "Somu", "Payment" : { "Total" : 600 }, "Trans
action" : { "Price" : 400 } }
```

b. Find any record where total payment amount (Payment.Total) is 600.

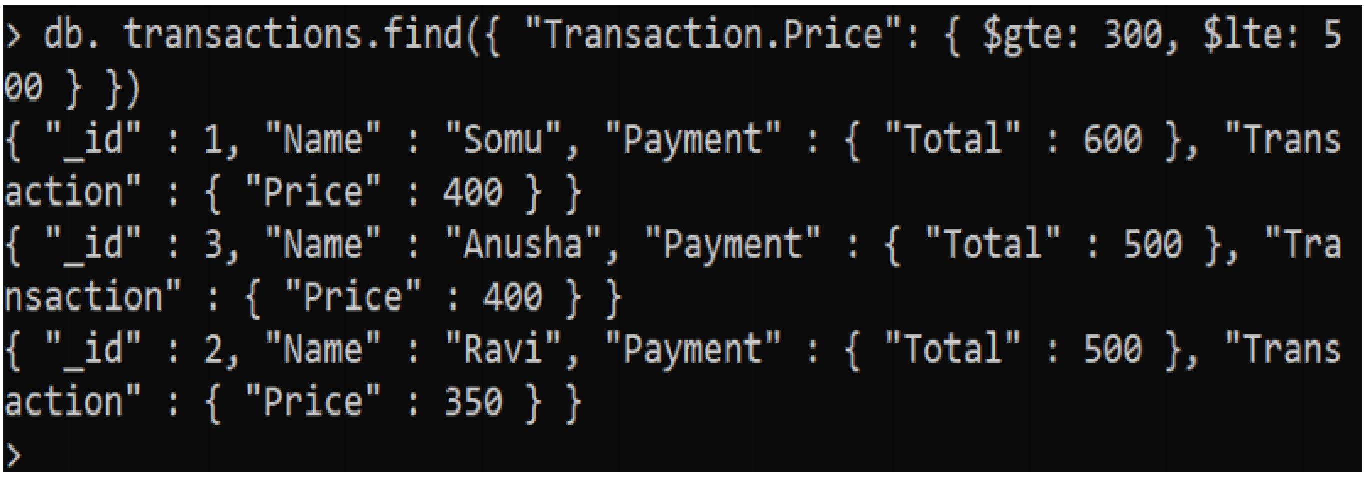
db.transactions.find({ "Payment.Total": 600 })



```
> db.transactions.find({"Payment.Total":600})
{ "_id" : 1, "Name" : "Somu", "Payment" : { "Total" : 600 }, "Trans
action" : { "Price" : 400 } }
```

c. Find any record where price (Transaction.price) is between 300 to 500.

db. transactions.find({ "Transaction.Price": { $gte: 300, $lte: 500 } })



```
> db. transactions.find({ "Transaction.Price": { $gte: 300, $lte: 5
00 } })
{ "_id" : 1, "Name" : "Somu", "Payment" : { "Total" : 600 }, "Trans
action" : { "Price" : 400 } }
{ "_id" : 3, "Name" : "Anusha", "Payment" : { "Total" : 500 }, "Tra
nsaction" : { "Price" : 400 } }
{ "_id" : 2, "Name" : "Ravi", "Payment" : { "Total" : 500 }, "Trans
action" : { "Price" : 350 } }
>
```

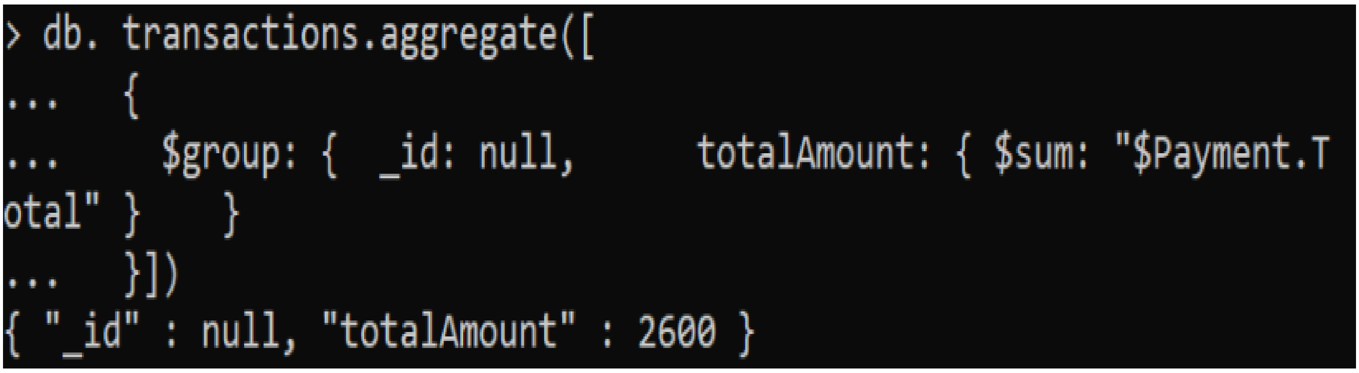
d. Calculate the total transaction amount by adding up Payment.Total in all records.

db. transactions.aggregate([

{

$group: { _id: null, totalAmount: { $sum: "$Payment.Total" } }

}])



```
> db. transactions.aggregate([
...     {
...       $group: {  _id: null,       totalAmount: { $sum: "$Payment.T
otal" }     }
...     }])
{ "_id" : null, "totalAmount" : 2600 }
```