Mongo Db Quries

Model the following Property system as a document database. Consider a set of Property, Owner. One owner can buy many properties.

1. Assume appropriate attributes and collections as per the query requirements.

```
use PraticalExam
db.createCollection("Properties")
db.createCollection("Owner")
```

Insert at least 05 documents in each collection.

```
db.Properties.insertMany([{"property_id": 1, "area": "Nashik","rate": 95000, "owner_id": 101},
{"property_id": 2,"area": "Mumbai","rate": 120000,"owner_id": 102},
{"property_id": 3,"area": "Pune","rate": 80000, "owner_id": 103},
{"property_id": 4, "area": "Nagpur","rate": 105000, "owner_id": 101},
{"property_id": 5, "area": "Nashik","rate": 85000, "owner_id": 104}])

db.Owner.insertMany([{"owner_id": 101,"name": "Mr. Patil"},
{"owner_id": 102,"name": "Mr. Singh"},
{"owner_id": 103,"name": "Mrs. Gupta"},
{"owner_id": 104,"name": "Ms. Deshmukh"},
{"owner_id": 105,"name": "Mr. Sharma"}])
```

Display area wise property details

```
db.Properties.aggregate([
  { $group: { _id: "$area", properties: { $push: "$$ROOT" } } }
])
```

Display property owned by 'Mr.Patil' having minimum rate:

```
db.Properties.aggregate([ { $match: { owner_id: 101 } },{ $sort: { rate: 1 } },{ $limit: 1 }])
```

Display area of property whose rate is less than 100000:

```
db.Properties.find({ rate: { $lt: 100000 } }, { area: 1, _id: 0 })
```

Give the details of owner whose property is at "Nashik"

```
db.Owners.aggregate([
  {
    $lookup: {
      from: "Properties",
      localField: "owner_id",
      foreignField: "owner_id",
      as: "properties"
    }
  },
  { $unwind: "$properties" },
  { $match: { "properties.area": "Nashik" } }
])
```

Mongo Db Quries

Model the following system as a document database. Consider a database of newspaper, publisher, and city. Different publisher publishes various newspapers in different cities

Assume appropriate attributes and collections as per the query requirements

```
db.createCollection("Newspapers")

db.createCollection("Publishers")

db.createCollection("Cities")
```

Insert at least 5 documents in each collection

```
db.Newspapers.insertMany([{"newspaper_id": 1,"name": "Times of India","language":
"English","publisher_id": 101,"city_id": 201},
{"newspaper_id": 2,"name": "Loksatta","language": "Marathi","publisher_id": 102,"city_id": 202},
{"newspaper_id": 3,"name": "Economic Times","language": "English","publisher_id": 103,"city_id": 203},
{"newspaper_id": 4,"name": "aamchi Mumbai","language": "Marathi","publisher_id": 104,"city_id": 204},
```

```
db.Publishers.insertMany([{"publisher_id": 101,"name": "Bennett Coleman & Co. Ltd","state":
"Maharashtra"},
{"publisher_id": 102,"name": "Indian Express Group","state": "Gujarat"},
{"publisher_id": 103,"name": "Sakal Media Group","state": "Maharashtra"},
{"publisher_id": 104,"name": "Dainik Bhaskar Group","state": "Rajasthan"},
{"publisher_id": 105,"name": "The Hindu Group","state": "Tamil Nadu"}])
```

```
db.Cities.insertMany([{"city_id": 201,"name": "Mumbai"},
{"city_id": 202,"name": "Pune"},
{"city_id": 203,"name": "Nashik"},
{"city_id": 204,"name": "Delhi"},
{"city_id": 205,"name": "Bangalore"}])
```

List all newspapers available in "NASHIK" city:

```
db.Newspapers.aggregate([
 { $lookup: { from: "Cities", localField: "city_id", foreignField: "city_id", as: "city" } },
 { $match: { "city.name": "Nashik" } }
```

List all the newspapers of "Marathi" language:

```
db.Newspapers.find({ language: "Marathi" })
```

Count the number of publishers in "Gujarat" state

```
db.Publishers.countDocuments({ state: "Gujarat" })
```

Mongo Db Quries

Write a cursor to show newspapers with the highest sale in Maharashtra state

```
var cursor = db.Newspapers.aggregate([
  { $lookup: { from: "Publishers", localField: "publisher_id", foreignField: "publisher_id", as: "publisher" } },
  { $match: { "publisher.state": "Maharashtra" } },
  { $group: { _id: "$name", total_sales: { $sum: 1 } } },
  { $sort: { total_sales: -1 } }
])
```

Model the following system as a document database. Consider employee and department's information. Assume appropriate attributes and collections as per the query requirements.

```
db.createCollection("employees")

db.createCollection("departments")
```

Insert at least 5 documents in each collection

```
db.employees.insertMany([
  { name: "Amaan", department: "Sales", salary: 60000 },
  { name: "Vanshay", department: "Sales", salary: 55000 },
  { name: "Rushikesh", department: "HR", salary: 48000 },
  { name: "Aditya", department: "Marketing", salary: 52000 },
  { name: "Sneha", department: "IT", salary: 65000 }
])
```

```
db.departments.insertMany([
  { name: "Sales", employees: 2 },
  { name: "HR", employees: 1 },
  { name: "Marketing", employees: 1 },
  { name: "IT", employees: 1 }
])
```

Display name of employee who has the highest salary.

```
db.employees.find().sort({ salary: -1 }).limit(1)
```

Display the biggest department with the maximum number of employees.

```
db.departments.find().sort({ employees: -1 }).limit(1)
```

List all the employees who work in Sales dept and salary > 50000

Mongo Db Quries

```
db.employees.find({ department: "Sales", salary: { $gt: 50000 } })
```

Write a cursor which shows department wise employee information

```
var cursor = db.departments.find();
while (cursor.hasNext()) {
  var department = cursor.next();
  print("Department:", department.name);
  var employeesCursor = db.employees.find({ department: department.name });
  while (employeesCursor.hasNext()) {
    var employee = employeesCursor.next();
    print("\tEmployee:", employee.name, "\tSalary:", employee.salary);
  }
}
```

Model the following information system as a document database. Consider hospitals around Nashik. Each hospital may have one or more specializations like Pediatric, Gynaec, Orthopedic, etc. A person can recommend/provide review for a hospital. A doctor can give service to one or more hospitals.

```
db.createCollection("hospitals")

db.createCollection("doctors")
```

```
db.hospitals.insertMany([{name:"Nashik Medical
Center",city:"Nashik",rating:4,specializations:["Pediatric","Orthopedic"]},
{name:"Sunrise Hospital",city:"Nashik",rating:3,specializations:["Pediatric","Gynaec,"]},
{name:"Lotus Hospital",city:"Nashik",rating:3,specializations:["Gynaec,","Orthopedic"]},
{name:"Surya Hospital",city:"Nashik",rating:2,specializations:["Oncology","Nephrology"]},
{name:"Cure Hospital",city:"Nashik",rating:5,specializations:["Dentistry","ENT"]},
{name:"RainbowHospital",city:"Nashik",rating:4.5,specializations:["Psychiatry","Orthopedic"]},
{name:"Greenfield Hospital",city:"Nashik",rating:4,specializations:["Internal Medicine","Endocrinology"]},
{name:"Cosmo Hospital",city:"Nashik",rating:5,specializations:["Pediatric","Urology"]},
{name:"Hope Hospital",city:"Nashik",rating:4,specializations:["Pediatric","Rheumatology"]},
{name:"Apollo Hospital",city:"Nashik",rating:2,specializations:["Radiology","Gynaec,"]}])
```

```
db.doctors.insertMany([{name:"Dr. Deshmukh",hospitals_visited:["Nashik Medical Center","Sunrise
Hospital","Greenfield Hospital"]},
{name:"Dr. Gupta",hospitals_visited:["Lotus Hospital","Surya Hospital","Hope Hospital"]},
{name:"Dr. Khan",hospitals_visited:["Cure Hospital","Rainbow Hospital","Apollo Hospital"]},
{name:"Dr. Joshi",hospitals_visited:["Cosmo Hospital"]},
{name:"Dr. Singh",hospitals_visited:["Greenfield Hospital","Apollo Hospital"]},
{name:"Dr. Patel",hospitals_visited:["Hope Hospital","Rainbow Hospital","Cure Hospital"]},
{name:"Dr. Kumar",hospitals_visited:["Surya Hospital","Lotus Hospital"]},
{name:"Dr. Khan",hospitals_visited:["Nashik Medical Center","Sunrise Hospital","Greenfield Hospital"]},
```

Mongo Db Quries

```
{name:"Dr. Reddy",hospitals_visited:["Cure Hospital","Cosmo Hospital","Rainbow Hospital"]},
{name:"Dr. Verma",hospitals_visited:["Apollo Hospital","Surya Hospital"]}]
```

List the names of hospitals with Gynaec specialization

```
db.hospitals.find({ specializations: "Gynaec," })
```

List the Names of all hospitals located in Nashik city

```
db.hospitals.find({ city: "Nashik" }, { _id: 0, name: 1 })
```

List the names of hospitals where Dr. Deshmukh visit

```
db.doctors.findOne({ name: "Dr.Deshmukh" }).hospitals_visited
```

List the names of hospitals whose rating >= 4

```
db.hospitals.find({ rating: { $gte: 4 } }, { _id: 0, name: 1 })
```

Model the following Mongodb database in mongosh. Many employees working on one project. A company has various ongoing projects. 2. Assume appropriate attributes and collections as per the query requirements.

```
db.createCollection("employees")

db.createCollection("projects")
```

```
db.employees.insertMany([
  { name: "Mr.Patil", project: "Project A" },
  { name: "Mr.Shaikh", project: "Project B" },
  { name: "Mr.Johnson", project: "Project A" },
  { name: "Mr.Kulkarni", project: "Project C" },
  { name: "Mr.Khan", project: "Project B" }
])
```

```
db.projects.insertMany([
  { name: "Project A", type: "Type1", duration_months: 4 },
  { name: "Project B", type: "Type2", duration_months: 2 },
  { name: "Project C", type: "Type1", duration_months: 5 },
  { name: "Project D", type: "Type3", duration_months: 6 },
  { name: "Project E", type: "Type2", duration_months: 3 }
])
```

Mongo Db Quries

List all names of projects where Project_type = "Type1"

```
db.projects.find({ type: "Type1" }, { _id: 0, name: 1 })
```

List all the projects with duration greater than 3 months

```
db.projects.find({ duration_months: { $gt: 3 } })
```

Count no. of employees working on a particular project (e.g., "Project A")

```
db.employees.countDocuments({ project: "Project A" })
```

List the names of projects on which Mr. Patil is working

```
db.employees.find({ name: "Mr.Patil" }, { _id: 0, project: 1 })
```

Model the following information as a Mongodb database in mongoshell. A customer can take different policies and get the benefit. There are different types of policies provided by various companies. Assume appropriate attributes and collections as per the query requirements. Insert at least 5 documents in each collection.

```
db.createCollection("customers")
```

```
db.customers.insertMany([
   { name: "Amaan", policy: "Komal Jeevan", premium_amount: 200,type: "Yearly",company:"Life Insurers Ltd" },
   { name: "Vanshay", policy: "Health Guard", premium_amount: 250,type: "Monthly",company:"Health Insurers Inc"},
   { name: "Adityya", policy: "Retirement Plus", premium_amount: 300,type: "Half Yearly",company:"Retirement Solutions" },
   { name: "Rushikesh", policy: "Komal Jeevan", premium_amount: 220,type: "Yearly",company:"Life Insurers Ltd"},
   { name: "Sneha", policy: "Accident Protector", premium_amount: 270,type: "Monthly",company:"Health Insurers Inc"}
])
```

List the details of customers who have taken "Komal Jeevan" Policy

```
db.customers.find({ policy: "Komal Jeevan" })
```

Display average premium amount

```
db.customers.aggregate([
```

Mongo Db Quries

```
  { $group: { _id: null, avg_premium: { $avg: "$premium_amount" } } }])
```

Increase the premium amount by 5% for policy type="Monthly"

```
db.policies.updateMany({ type: "Monthly" }, { $mul: { premium_amount: 1.05 } })
```

Count the number of customers who have taken policy type "half yearly

```
db.customers.find({ "type": "Half Yearly" }).count()
```

1. Model the following information as a Mongodb database in mongo shell. A customer operates his bank account, does various transactions and get the banking services Assume appropriate attributes and collections as per the query requirements. Insert at least 5 documents in each collection.

```
db.createCollection("customers")

db.createCollection("customers")


db.customers.insertMany([
   { firstName: "Amaan", acctype: "Saving", branch: "A" },
   { firstName: "Sara", acctype: "Loan", branch: "B" },
   { firstName: "Sam", acctype: "Saving", branch: "A" },
   { firstName: "Denzil", acctype: "Loan", branch: "C" },
   { firstName: "Merlin", acctype: "Saving", branch: "B" }
])

db.transactions.insertMany([
   { customerId: 1, date: "2020-01-01" },
   { customerId: 2, date: "2020-01-01" },
   { customerId: 3, date: "2021-03-15" },
   { customerId: 4, date: "2020-01-01" },
   { customerId: 5, date: "2020-01-01" }
])
```

List names of all customers whose first name starts with a “S”

```
db.customers.find({ firstName: /^S/ }, { _id: 0, firstName: 1 })
```

List the names customers where acctype=”Saving”

```
db.customers.find({ acctype: "Saving" }, { _id: 0, firstName: 1 })
```

Mongo Db Quries

Count the total number of loan account holders of a specific branch

```
db.customers.count({ acctype: "Loan", branch: "A" })

db.customers.count({ acctype: "Loan", branch: "C" })
```

Model the following inventory information as a Mongodb database in mongoshell. The inventory keeps track of various items. The items are tagged in various categories. Items may be kept in various warehouses and each warehouse keeps track of the quantity of the item. Assume appropriate attributes and collections as per the query requirements.

```
db.createCollection("items")

db.createCollection("inventories"

db.items.insertMany([
   { name: "Laptop", tags: ["electronics", "laptop","Gadgets","Tech","Office","Gaming"], status: "A",
height:10 },
   { name: "T-shirt", tags: ["clothing", "t-shirt","Fashion","Fabric"], status: "B", height: 5 },
   { name: "Notebook", tags: ["office supplies", "notebook","Stationary","School","Official"], status: "C",
height: 8 },
   { name: "Chair", tags: ["furniture", "chair","Home","Decor"], status: "A", height: 12 },
   { name: "Basketball", tags: ["sports equipment", "basketball"], status: "B", height: 10 }
]);



db.inventories.insertMany([
   { item: "Laptop", warehouse: "Warehouse A", quantity: 500 },
   { item: "T-shirt", warehouse: "Warehouse B", quantity: 100 },
   { item: "Notebook", warehouse: "Warehouse C", quantity: 200 },
   { item: "Chair", warehouse: "Warehouse A", quantity: 50 },
   { item: "Basketball", warehouse: "Warehouse B", quantity: 400 }
]);

db.inventories.insertOne({ item: "Planner", warehouse: "Warehouse B", quantity: 10 })

db.items.insertOne({ name: "Planner", tags: ["office supplies", "notebook","Stationary","School","Official"],
status: "C", height: 8 })
```

List all the items where quantity is greater than 300

```
db.inventories.find({ quantity: { $gt: 300 } }, { _id: 0, item: 1, quantity: 1 })
```

Mongo Db Quries

List all items which have less than 5 tags

```
db.items.find({ $expr: { $lt: [{ $size: "$tags" }, 5] } })
```

List all items having status equal to "B" or having quantity less than 50 and height of the product greater than 8

```
db.items.find({
   $or: [
      { status: "B" },{ $and: [{ quantity: { $lt: 50 } }, { height: { $gt: 8 } }] } ]});
```

Find all warehouses that keep the item "Planner" and have in-stock quantity less than 20

```
db.inventories.find({ item: "Planner", quantity: { $lt: 20 } }, { _id: 0, warehouse: 1 })
```

Model the following Customer Loan information as a document database. Consider Customer Loan information system where the customer can take many types of loans. Assume appropriate attributes and collections as per the query requirements

```
db.createCollection("customers");

db.createCollection("loans")

db.customers.insertMany([
   { name: "David", address: "Vallab Nagar", city: "Pimpri" },
   { name: "Amaan", address: "st Andrews Bandra", city: "Mumbai" },
   { name: "Vanshay", address: "Chinchwad", city: "Pune" },
   { name: "Dharam", address: "Rawat", city: "Pimpri" },
   { name: "Mr.Patil", address: "Andheri", city: "Mumbai" },
   { name: "Derek", address: "Viman Nagar", city: "Pune" },
   { name: "Dinesh", address: "kasarwadi", city: "Pimpri" },
   { name: "Rushikesh", address: "Churchgate", city: "Mumbai" },
   { name: "Aditya", address: "Dadar", city: "Pune" },
   { name: "Duncan", address: "777 Elm St", city: "Pimpri" }
])
```

Mongo Db Quries

```
db.loans.insertMany([
   { customer_id: ObjectId("65fc8c748b4816bd1a1d4983"), loan_type: "Personal", loan_amount: 10000 },
   { customer_id: ObjectId("65fc8c748b4816bd1a1d4984"), loan_type: "Home", loan_amount: 200000 },
   { customer_id: ObjectId("65fc8c748b4816bd1a1d4985"), loan_type: "Car", loan_amount: 50000 },
   { customer_id: ObjectId("65fc8c748b4816bd1a1d4986"), loan_type: "Education", loan_amount: 30000},
   { customer_id: ObjectId("65fc8c748b4816bd1a1d4987"), loan_type: "Personal", loan_amount: 150000},
   { customer_id: ObjectId("65fc8c748b4816bd1a1d4988"), loan_type: "Home", loan_amount: 250000 },
   { customer_id: ObjectId("65fc8c748b4816bd1a1d4989"), loan_type: "Car", loan_amount: 60000 },
   { customer_id: ObjectId("65fc8c748b4816bd1a1d498a"), loan_type: "Education", loan_amount: 40000},
   { customer_id: ObjectId("65fc8c748b4816bd1a1d498b"), loan_type: "Personal", loan_amount: 200000},
   { customer_id: ObjectId("65fc8c748b4816bd1a1d498c"), loan_type: "Home", loan_amount: 300000 }])
```

List all customers whose name starts with 'D' character.

```
db.customers.find({ name: /^D/ })
```

List the names of customers in descending order who have taken a loan from Pimpri city.

```
db.loans.find( { loan_amount: { $gt: 100000 } })
```

Display customer details having the maximum loan amount.

```
db.loans.aggregate([
   { $group: { _id: "$customer_id", maxLoanAmount: { $max: "$loan_amount" } } },
   { $lookup: { from: "customers", localField: "_id", foreignField: "_id", as: "customerDetails" } },
   { $unwind: "$customerDetails" },
   { $sort: { maxLoanAmount: -1 } },
   { $limit: 1 },
   { $project: { _id: 0, "Customer Name": "$customerDetails.name", "Address": "$customerDetails.address",
"City": "$customerDetails.city", "Max Loan Amount": "$maxLoanAmount" } }
])
```

Update the address of the customer whose name is "Mr. Patil" and loan amount is greater than 100000

```
db.customers.updateOne(
   { name: "Mr.Patil", },
   { $set: { address: "Dahanu" } }
)
```

Mongo Db Quries

Model the following Online shopping information as a document database. Consider online shopping where the customer can get different products from different brands. Customers can rate the brands and products Assume appropriate attributes and collections as per the query requirements

```
db.createCollection("customers")

db.createCollection("products")

db.createCollection("brands")


db.customers.insertMany([
   { name: "Alice", city: "New York", purchase_date: "15/08/2023", bill_amount: 60000 },
   { name: "Bob", city: "Los Angeles", purchase_date: "15/08/2023", bill_amount: 70000 },
   { name: "Charlie", city: "Chicago", purchase_date: "16/08/2023", bill_amount: 45000 },
   { name: "David", city: "New York", purchase_date: "15/08/2023", bill_amount: 80000 },
   { name: "Eve", city: "San Francisco", purchase_date: "16/08/2023", bill_amount: 55000 }
])


db.products.insertMany([
   { name: "Laptop", brand: "Dell", warranty_period: "1 year" },
   { name: "Smartphone", brand: "Apple", warranty_period: "2 years" },
   { name: "TV", brand: "Samsung", warranty_period: "1 year" },
   { name: "Watch", brand: "Fossil", warranty_period: "1 year" },
   { name: "Headphones", brand: "Sony", warranty_period: "2 years" }
])


db.brands.insertMany([
   { name: "Dell", rating: 4.5 },
   { name: "Apple", rating: 4.8 },
   { name: "Samsung", rating: 4.3 },
   { name: "Fossil", rating: 4.2 },
   { name: "Sony", rating: 4.4 }
])
```

List the names of product whose warranty period is one year

```
db.products.find({ warranty_period: "1 year" }, { _id: 0, name: 1 })
```

List the customers has done purchase on “15/08/2023”

```
db.customers.find({ purchase_date: "15/08/2023" })
```

Display the names of products with brand which have highest rating

Mongo Db Quries

```
db.brands.find().sort({ rating: -1 }).limit(1)

db.products.find({ brand: "Apple" }, { _id: 0, name: 1 })
```

Display customers who stay in …… city and billamt >50000

```
db.customers.find({ city: "New York", bill_amount: { $gt: 50000 } })
```

Mongo Db Quries