MongoDB Exercise Set #2: Project & Task Tracker

Database: taskmanager Collections: projects , tasks , users

Section 1: Basic Document Modeling & Insertion

1 Create a database called taskmanager

Use taskmanager

2 Insert 3 users into a users collection. Each should have:
name (string)
email (string)
role (either "admin" , "manager" , or "developer")
active (boolean)

```
db.users.insertMany([
 {
  name: "Admin User",
  email: "admin@company.com",
  role: "admin",
  active: true
 },
 {
  name: "Project Manager",
  email: "manager@company.com",
  role: "manager",
  active: true
 },
 {
  name: "Developer One",
  email: "dev1@company.com",
  role: "developer",
  active: true
 }
])
```

3 Insert 2 projects into a projects collection:

title , description , startDate , status (e.g. "active" , "completed") Embed a createdBy sub-document containing the user's _id , name

```
const adminId = db.users.findOne({name: "Admin User"})._id
const managerId = db.users.findOne({name: "Project Manager"})._id
db.projects.insertMany([
 {
  title: "Website Redesign",
  description: "Complete redesign of company website",
  startDate: new Date("2025-05-01"),
  status: "active",
  createdBy: {
   _id: adminId,
   name: "Admin User"
  }
 },
 {
  title: "Mobile App Development",
  description: "Build new iOS and Android app",
  startDate: new Date("2025-04-15"),
  status: "active",
  createdBy: {
   _id: managerId,
   name: "Project Manager"
  }
 }
])
```

4 Insert 5 tasks into a tasks collection:

Fields: title , assignedTo (user _id), projectId , priority , dueDate , status

```
const websiteProjectId = db.projects.findOne({title: "Website Redesign"})._id
const appProjectId = db.projects.findOne({title: "Mobile App Development"})._id
const devId = db.users.findOne({name: "Developer One"})._id

db.tasks.insertMany([
 {
  title: "Design homepage",
  assignedTo: devId,
```

```
  projectId: websiteProjectId,
  priority: "high",
  dueDate: new Date("2025-07-30"),
  status: "in progress"
 },
 {
  title: "Implement login API",
  assignedTo: devId,
  projectId: appProjectId,
  priority: "high",
  dueDate: new Date("2025-07-15"),
  status: "not started"
 },
 {
  title: "Write documentation",
  assignedTo: devId,
  projectId: websiteProjectId,
  priority: "medium",
  dueDate: new Date("2025-07-01"),
  status: "not started"
 },
 {
  title: "Test mobile UI",
  assignedTo: devId,
  projectId: appProjectId,
  priority: "medium",
  dueDate: new Date("2025-07-10"),
  status: "in progress"
 },
 {
  title: "Deploy staging environment",
  assignedTo: devId,
  projectId: websiteProjectId,
  priority: "low",
  dueDate: new Date("2025-07-05"),
  status: "completed"
 }
])
```

Section 2: Filtering & Querying

5 Find all tasks with priority "high" that are not completed

```
db.tasks.find({
  priority: "high",
  status: { $ne: "completed" }
})
```

6 Query all active users with role "developer"

```
db.users.find({
  role: "developer",
  active: true
})
```

7 Find all tasks assigned to a specific user (by ObjectId)

```
db.tasks.find({
  assignedTo: devId
})
```

8 Find all projects started in the last 30 days

```
db.projects.find({
  startDate: { $gt: new Date(Date.now() - 30 * 24 * 60 * 60 * 1000) }})
```

Section 3: Update Operations

9 Change the status of one task to "completed"

```
db.tasks.updateOne(
  { title: "Design homepage" },
  { $set: { status: "completed" } })
```

10 Add a new role field called "teamLead" to one of the users

```
db.users.updateOne(
  { name: "Developer One" },
  { $set: { role: "teamLead" } }
```

```
)
```

11 Add a new tag array to a task: ["urgent", "frontend"]

```
db.tasks.updateOne(
  { title: "Implement login API" },
  { $set: { tags: ["urgent", "frontend"] } }
)
```

Section 4: Array and Subdocument Operations

12 Add a new tag "UI" to the task's tags array using $addToSet

```
db.tasks.updateOne(
  { title: "Implement login API" },
  { $addToSet: { tags: "UI" } })
```

13 Remove "frontend" from a task's tag list

```
db.tasks.updateOne(
  { title: "Implement login API" },
  { $pull: { tags: "frontend" } })
```

14 Use $inc to increment a project 's progress field by 10

```
db.projects.updateOne(
  { title: "Website Redesign" },
  { $inc: { progress: 10 } }
)
```

Section 5: Aggregation & Lookup

15 Use $lookup to join tasks with users and show task title + assignee name

```
db.tasks.aggregate([
  { $lookup: {
     from: "users",
     localField: "assignedTo",
     foreignField: "_id",
     as: "assignee"}},
```

```
 {$project: {
    title: 1,
    "assignee.name": 1}}])
```

16 Use $lookup to join tasks with projects , and filter tasks where project status = active

```
db.tasks.aggregate([
 {$lookup: {
    from: "projects",
    localField: "projectId",
    foreignField: "_id",
    as: "project"}},
 {$match: {
    "project.status": "active"}}])
```

17 Use $group to get count of tasks per status

```
db.tasks.aggregate([
 {
   $group: {
    _id: "$status",
    count: { $sum: 1 }}}])
```

18 Use $match , $sort , and $limit to get top 3 soonest due tasks

```
db.tasks.aggregate([
 {$match: {
    status: { $ne: "completed" }}},
 {$sort: {
    dueDate: 1 }},
 {$limit: 3}])
```