## 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.createCollection("users")
```

```
db.users.insertMany(
[
{name: "Erwin", email: "smith@gmail.com", role: "manager", active: false},
{name: "Levi", email: "ackerman@gmail.com", role: "developer", active: true},
{name: "Floch", email: "eldia@gmail.com", role: "admin", 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

```
db.createCollection("projects")
```

```
db.projects.insertMany(
[
{title: "Health Tracker", description: "Try atleast once", startDate: new Date("2023-09-04"), status:
"completed", createdBy: {_id: ObjectId('6831fc7c7476a9f8a3e1200e'), name: "Levi"}},
{title: "Gym Hub", description: "Hit gym bro!!", startDate: new Date("2025-04-09"), status: "active",
createdBy: {_id: ObjectId('6831fc7c7476a9f8a3e1200f'), name: "Floch"}}
]
)
```

4. Insert 5 tasks into a tasks collection:

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

```
db.tasks.insertMany(
[
{title: "Login field", assignedTo: ObjectId('6831fc7c7476a9f8a3e1200f'), projectId:
ObjectId('6831fe377476a9f8a3e12011'), priority: "Medium", dueDate: new Date("2024-12-03"), status:
"completed"},
{title: "Code rewise", assignedTo: ObjectId('6831fc7c7476a9f8a3e1200e'), projectId:
ObjectId('6831fe377476a9f8a3e12010'), priority: "Low", dueDate: new Date("2025-04-03"), status:
"active"},
{title: "Debug", assignedTo: ObjectId('6831fc7c7476a9f8a3e1200f'), projectId:
ObjectId('6831fe377476a9f8a3e12011'), priority: "High", dueDate: new Date("2025-01-01"), status:
"completed"},
{title: "Crash issue", assignedTo: ObjectId('6831fc7c7476a9f8a3e1200e'), projectId:
ObjectId('6831fe377476a9f8a3e12010'), priority: "High", dueDate: new Date("2024-09-23"), status:
"completed"},
{title: "Q1 report", assignedTo: ObjectId('6831fc7c7476a9f8a3e1200d'), projectId:
ObjectId('6831fe377476a9f8a3e12011'), priority: "Medium", dueDate: new Date("2025-04-28"), status:
"active"}
]
)
```

## Section 2: Filtering & Querying

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

```
db.tasks.find({priority: "High", status: "active"})
```

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: ObjectId('6831fc7c7476a9f8a3e1200e')})
```

8. Find all projects started in the last 30 days

```
db.projects.find({
  startDate: {
    $gte: new Date(new Date().setDate(new Date().getDate() - 30))
  }
})
```

## Section 3: Update Operations

9. Change the status of one task to "completed"

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

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

```
db.users.updateOne({name: "Erwin"}, {$set: {teamLead: true}})
```

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

```
db.tasks.updateMany(
{_id: {$exists: true}},
{$set: {tag: ["urgent", "frontend"]}}
)
```

## Section 4: Array and Subdocument Operations

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

```
db.tasks.updateMany({_id: {$exists: true}}, {$addToSet: {tag: "UI"}})
```

13. Remove "frontend" from a task’s tag list

```
db.tasks.updateMany({_id: {$exists: true}}, {$pull: {tag: {$in: ["frontend"]}}})
```

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

```
db.projects.updateMany({}, {$inc: {progress: 10}})
```

## Section 5: Aggregation & Lookup

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

```
db.tasks.aggregate(
[
{$lookup: {from: "users", localField: "assignedTo", foreignField: "_id", as: "info"}},
{$project: {title: 1, _id: 0, "info.name": 1}}
]
)
```

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

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

17. Use $group to get count of tasks per status

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

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

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