Lab_10 - - > PART-C :

1. Create a capped collection named" logs" with a maximum size of 100 KB and a maximum of 10 documents.

→ db.createCollection("logs" , { capped : true , size : 102400 , max : 10 })

2. Insert below 12 log entries into the "logs" collection. Each entry should contain a message, level (e.g., "info", "warning", "error"), and a timestamp field. Use the insertMany() method.

{ message: "System started", level: "info", timestamp: new Date() }

{ message: "Disk space low", level: "warning", timestamp: new Date() }

{ message: "User login", level: "info", timestamp: new Date() }

{ message: "System reboot", level: "info", timestamp: new Date() }

{ message: "Error in module", level: "error", timestamp: new Date() }

{ message: "Memory usage high", level: "warning", timestamp: new Date() }

{ message: "User logout", level: "info", timestamp: new Date() }

{ message: "File uploaded", level: "info", timestamp: new Date() }

{ message: "Network error", level: "error", timestamp: new Date() }

{ message: "Backup completed", level: "info", timestamp: new Date() }

{ message: "Database error", level: "error", timestamp: new Date() }

{ message: "Service started", level: "info", timestamp: new Date() }

→ db.logs.insertMany([

{ message: "System started", level: "info", timestamp: new Date() },

{ message: "Disk space low", level: "warning", timestamp: new Date() },

{ message: "User login", level: "info", timestamp: new Date() },

{ message: "System reboot", level: "info", timestamp: new Date() },

{ message: "Error in module", level: "error", timestamp: new Date() },

{ message: "Memory usage high", level: "warning", timestamp: new Date() },

{ message: "User logout", level: "info", timestamp: new Date() },

{ message: "File uploaded", level: "info", timestamp: new Date() },

{ message: "Network error", level: "error", timestamp: new Date() },

{ message: "Backup completed", level: "info", timestamp: new Date() },

{ message: "Database error", level: "error", timestamp: new Date() },

{ message: "Service started", level: "info", timestamp: new Date() }])

3. Perform find method on “logs” collection to ensure only the last 10 documents are retained (even though you inserted 12).

→ db.logs.find()

→ db.logs.find().pretty()

4. Insert below 5 more documents and check if the oldest ones are automatically removed.

{ message: "New log entry 1", level: "info", timestamp: new Date() }

{ message: "New log entry 2", level: "info", timestamp: new Date() }

{ message: "New log entry 3", level: "info", timestamp: new Date() }

{ message: "New log entry 4", level: "warning", timestamp: new Date() }

{ message: "New log entry 5", level: "error", timestamp: new Date() }

→ db.logs.insertMany([

{ message: "New log entry 1", level: "info", timestamp: new Date() },

{ message: "New log entry 2", level: "info", timestamp: new Date() },

{ message: "New log entry 3", level: "info", timestamp: new Date() },

{ message: "New log entry 4", level: "warning", timestamp: new Date() },

{ message: "New log entry 5", level: "error", timestamp: new Date() }])