**Model Coding Challenge (9-6-25)**

**MongoDB**

Creating Database Students,

A screenshot of a computer

AI-generated content may be incorrect.

The `students` collection includes:

{

"name": "Amit Kumar",

"age": 17,

"gender": "Male",

"grade": "11th",

"subjects": ["Math", "Physics", "English"],

"marks": { "Math": 82, "Physics": 91, "English": 75 },

"address": {

"city": "Delhi",

"pincode": "110001"

}

}

Q1. Insert 5 student records

db.students.insertMany([

{

name: "Amit Kumar",

age: 17,

gender: "Male",

grade: "11th",

subjects: ["Math", "Physics", "English"],

marks: { Math: 82, Physics: 91, English: 75 },

address: { city: "Delhi", pincode: "110001" }

},

{

name: "Sneha Verma",

age: 16,

gender: "Female",

grade: "10th",

subjects: ["Biology", "Chemistry", "English"],

marks: { Biology: 78, Chemistry: 88, English: 92 },

address: { city: "Mumbai", pincode: "400001" }

},

{

name: "Rahul Singh",

age: 18,

gender: "Male",

grade: "12th",

subjects: ["Math", "Computer", "Physics"],

marks: { Math: 95, Computer: 89, Physics: 93 },

address: { city: "Delhi", pincode: "110002" }

},

{

name: "Anjali Mehra",

age: 17,

gender: "Female",

grade: "11th",

subjects: ["Math", "English", "History"],

marks: { Math: 39, English: 42, History: 78 },

address: { city: "Bangalore", pincode: "560001" }

},

{

name: "Karan Patel",

age: 15,

gender: "Male",

grade: "9th",

subjects: ["Science", "English", "Math"],

marks: { Science: 65, English: 38, Math: 41 },

address: { city: "Ahmedabad", pincode: "380001" }

}

])

A screenshot of a computer

AI-generated content may be incorrect.

Q2. Find all students who take “Math” as a subject

db.students.find(

{ subjects: "Math" },

{ \_id: 0, name: 1, grade: 1, subjects: 1 }

)

A black screen with a black background

AI-generated content may be incorrect.

Q3. Find students from city "Delhi" scoring more than 85 in Physics

db.students.find(

{

"address.city": "Delhi",

"marks.Physics": { $gt: 85 }

},

{

\_id: 0,

name: 1,

marks: 1,

"address.city": 1

}

)

A computer screen shot of a black screen

AI-generated content may be incorrect.

Q4. Sort all students by English marks (descending) and show top 3

db.students.find(

{ "marks.English": { $exists: true } },

{ \_id: 0, name: 1, subjects: 1, marks: 1 }

).sort({ "marks.English": -1 }).limit(3)

A screenshot of a computer program

AI-generated content may be incorrect.

Q5. Group students by city and count how many are from each

db.students.aggregate([

{

$group: {

\_id: "$address.city",

total\_students: { $sum: 1 }

}

}

])

A screen shot of a computer program

AI-generated content may be incorrect.

Q6. Update all students who scored less than 40 in any subject

db.students.updateMany(

{

$or: [

{ "marks.Math": { $lt: 40 } },

{ "marks.English": { $lt: 40 } },

{ "marks.Physics": { $lt: 40 } },

{ "marks.Computer": { $lt: 40 } },

{ "marks.Chemistry": { $lt: 40 } },

{ "marks.Biology": { $lt: 40 } },

{ "marks.Science": { $lt: 40 } },

{ "marks.History": { $lt: 40 } }

]

},

{ $set: { status: "Needs Improvement" } }

)

A screen shot of a computer program

AI-generated content may be incorrect.

Q7. Calculate average marks of all students in each subject

db.students.aggregate([

{

$group: {

\_id: null,

avgMath: { $avg: "$marks.Math" },

avgPhysics: { $avg: "$marks.Physics" },

avgEnglish: { $avg: "$marks.English" },

avgComputer: { $avg: "$marks.Computer" },

avgBiology: { $avg: "$marks.Biology" },

avgChemistry: { $avg: "$marks.Chemistry" },

avgScience: { $avg: "$marks.Science" },

avgHistory: { $avg: "$marks.History" }

}

}

])

A screenshot of a computer screen

AI-generated content may be incorrect.