db.students.insertMany([

{

student\_id: 1,

student\_name: "Samir Shah",

address: "Baroda",

birthdate: new Date("2002-01-20"),

CGPA: 8.5,

fee: 50000,

current\_year: "TE",

join\_date: new Date("2021-06-15"),

department: "CSE",

skills: ["Java", "Python"]

},

{

student\_id: 2,

student\_name: "Meera Patel",

address: "Ahmedabad",

birthdate: new Date("2001-12-15"),

CGPA: 9.2,

fee: 55000,

current\_year: "BE",

join\_date: new Date("2020-07-12"),

department: "CSE",

skills: ["C++", "Typing"]

},

{

student\_id: 3,

student\_name: "Ankit Mehta",

address: "Pune",

birthdate: new Date("2003-03-10"),

CGPA: 7.8,

fee: 48000,

current\_year: "SE",

join\_date: new Date("2022-08-10"),

department: "IT",

skills: ["HTML", "CSS"]

},

{

student\_id: 4,

student\_name: "Sonal Gupta",

address: "Pune",

birthdate: new Date("2004-11-20"),

CGPA: 8.1,

fee: 53000,

current\_year: "FE",

join\_date: new Date("2023-01-15"),

department: "ETC",

skills: ["Typing", "Electronics"]

},

{

student\_id: 5,

student\_name: "Mihir Desai",

address: "Baroda",

birthdate: new Date("2000-04-20"),

CGPA: 8.9,

fee: 52000,

current\_year: "BE",

join\_date: new Date("2019-09-20"),

department: "CSE",

skills: ["AI", "ML", "Typing"]

}

]);

1. Display the count of total no students from the institute:

db.students.countDocuments();

1. Display all the Students in seniority level (based on CGPA):

db.students.find().sort({ CGPA: -1 });

1. List the student details that are from Baroda or Ahmedabad and in CSE department:

db.students.find({

address: { $in: ["Baroda", "Ahmedabad"] },

department: "CSE"

});

1. List of the student\_id, student\_name, department, and skills of students whose birth date is the 20th of any month:

db.students.find({ birthdate: { $regex: "-20$" } }, { student\_id: 1, student\_name: 1, department: 1, skills: 1 });

1. Calculate the age of each student. Consider today's date:

db.students.aggregate([

{

$addFields: {

age: {

$floor: {

$divide: [

{ $subtract: [new Date(), "$birthdate"] },

1000 \* 60 \* 60 \* 24 \* 365

]

}

}

}

}

]);

1. List the name of students whose name starts with ‘s’ or ‘m’, are in the computer department, and have typing skills:

db.students.find({

student\_name: { $regex: "^[SsMm]" },

department: "CSE",

skills: "Typing"

});

1. Count the number of students in the IT department from Pune:

db.students.countDocuments({

department: "IT",

address: "Pune"

});

1. Arrange student names in alphabetical order whose age is between 18 and 20 and are in the ETC department:

db.students.aggregate([

{

$addFields: {

age: {

$floor: {

$divide: [

{ $subtract: [new Date(), "$birthdate"] },

1000 \* 60 \* 60 \* 24 \* 365

]

}

}

}

},

{

$match: {

age: { $gte: 18, $lte: 20 },

department: "ETC"

}

},

{ $sort: { student\_name: 1 } }

]);

1. Perform Create Index, get Index, and drop index operations on collection:

db.students.createIndex({ student\_name: 1 });

db.students.getIndexes();

db.students.dropIndex("student\_name\_1");

1. MapReduce or Aggregation function to display total number of students from each department:

db.students.aggregate([

{

$group: {

\_id: "$department",

total\_students: { $sum: 1 }

}

}

]);