To create the MongoDB tables for your trip application, you would typically design a database schema that consists of collections representing different entities and their corresponding fields. Based on the provided case study, here's an example schema for your MongoDB tables:1. Collection: users  
 - Fields:  
 - \_id: The unique identifier for each user (automatically generated by MongoDB).  
 - username: The username of the user.  
 - password: The hashed password of the user.  
 - email: The email address of the user.  
 - firstName: The first name of the user.  
 - lastName: The last name of the user.  
 - createdAt: The timestamp indicating when the user account was created.2. Collection: trips  
 - Fields:  
 - \_id: The unique identifier for each trip (automatically generated by MongoDB).  
 - title: The title or name of the trip.  
 - description: A brief description of the trip.  
 - startDate: The start date of the trip.  
 - endDate: The end date of the trip.  
 - location: The location or destination of the trip.  
 - maxParticipants: The maximum number of participants allowed on the trip.  
 - enrolledParticipants: An array of user IDs representing the participants enrolled in the trip.  
 - createdAt: The timestamp indicating when the trip was created.3. Collection: tripInformation  
 - Fields:  
 - \_id: The unique identifier for each trip information (automatically generated by MongoDB).  
 - tripId: The ID of the associated trip.  
 - information: Additional information about the trip.  
 - createdAt: The timestamp indicating when the trip information was created.

1)What is the total number of registered users in the system?

db.users.find().count();

  
2)How many trips are available in the database?

db.trips.find().count();

  
3)What are the usernames of the users who have enrolled in a specific trip?

4)Which trip has the highest number of enrolled participants?

db.trips.aggregate([  
 { $project: { \_id: 1, enrolledParticipants: 1, title: 1 } },  
 { $sort: { enrolledParticipants: -1 } },  
 { $limit: 1 }  
])

  
5)List all the trips that are scheduled to start in the next week.

6)Provide a count of trips grouped by their locations.

db.trips.aggregate([{$group:{\_id:"$location",count:{$sum:1}}}]);



7)What is the average number of enrolled participants per trip?

db.trips.aggregate([{$group:{\_id:"$enrolledParticipants",count:{$avg:"$enrolledParticipants"}}}]);



8)Which user has enrolled in the most number of trips?

db.users.aggregate([

{ $lookup: {

from: "trips",

localField: "\_id",

foreignField: "enrolledParticipants",

as: "enrolledTrips"

}},

{ $project: {

\_id: 1,

username: 1,

numTrips: { $size: "$enrolledTrips" }

}},

{ $sort: { numTrips: -1 } },

{ $limit: 1 }

]);

  
9)Retrieve the trip information for a particular trip.

db.tripinformation.findOne({trip\_id:"01H4V095Z7ER44XECA72DBPDSF"});

  
10)How many trips has each user enrolled in?

db.users.aggregate([

{ $lookup: {

from: "trips",

localField: "\_id",

foreignField: "enrolledParticipants",

as: "enrolledTrips"

}},

{ $project: {

\_id: 1,

username: 1,

numTrips: { $size: "$enrolledTrips" }

}}

]);

