考试题目：

1. Which of the following describe situations where MongoDB has a clear advantage over a typical relational database? Check all that apply.

A.The data model changes frequently

B.Developers need to get an application into production quickly

C.Cascade delete of referenced documents across collections

D.Perform range based queries on numerical data

2.Which write operations are atomic in MongoDB?

Check all answers that apply:

A.Deletion of a single document on a single mongod

B.Bulk update of several documents in a MongoDB sharded cluster

C.Update of a set of fields on a single document in a sharded cluster

D.Update of a field in multiple documents in a single mongod

3.Consider the following query:

db.stuff.updateMany( { b : { $gte : 10 } }, { $set : { b : 15 } }, { upsert : true } )

Assume that all documents in the answer set of part 1 are present in the collection for subsequent parts.

Part 1

Which documents will be matched by the query?

Check all answers that apply:

A.{ "\_id" : 21, "a" : 15, "b" : 1 }

B.{ "\_id" : 12, "a" : 22, "b" : 3 }

C.{ "\_id" : 3, "a" : -3, "b" : 76 }

D.{ "\_id" : 41, "a" : 4, "b" : 10 }

E.{ "\_id" : 25, "a" : 5, "b" : 15 }

Part 2

How many documents will be inserted by the query?

Choose the best answer:

A.0

B.1

C.2

D.3

E.5

Part 3

Which documents will be updated by the query?

Check all answers that apply:

A.{ "\_id" : 21, "a" : 15, "b" : 1 }

B.{ "\_id" : 12, "a" : 22, "b" : 3 }

C.{ "\_id" : 3, "a" : -3, "b" : 76 }

D.{ "\_id" : 41, "a" : 4, "b" : 10 }

E.{ "\_id" : 25, "a" : 5, "b" : 15 }

4.Consider the following example document from the "sample" collection. All documents in this collection have the same schema.

{

"\_id" : 3,

"a" : 7,

"c" : [ 3 , 4 , 7, -1, -1 ]

}

Which update below would remove the "c" field for the document with "\_id" : 12?

Choose the best answer:

A.db.sample.updateOne( { "\_id" : 12 } , { "c" : { "$delete" : true } } )

B.db.sample.updateOne( { "\_id" : 12 } , { "$unset" : { "c" : "" } } )

C.db.sample.updateOne( { "\_id" : 12 } , { "$remove" : { "c" : "" } } )

D.db.sample.updateOne( { "\_id" : 12 } , { "c" : null } )

E.db.sample.deleteOne( { "\_id" : 12 } , { "c" : 1} )

5.Which of the documents below in the people collection will be affected by the following query?

> db.people.deleteMany( { "lastName" : "Aaronson", "firstName" : "Arthur" } )

Check all that apply.

Check all answers that apply:

A.{ "\_id" : ObjectId("57fd413814c25e5820455eec"), "lastName" : "Barnes", "firstName" : "Arthur" }

B.{ "\_id" : ObjectId("57fd414c14c25e5826a65918"), "lastName" : "Chan", "firstName" : "Dorothy" }

C.{ "\_id" : ObjectId("57fd415614c25e582c83aac1"), "lastName" : "Aaronson", "firstName" : "Beth" }

D.{ "\_id" : ObjectId("57fd415d14c25e583246212a"), "lastName" : "Aaronson", "firstName" : "Arthur", "middleName" : "Alan" }

E.{ "\_id" : ObjectId("57fd416714c25e5838ab079c"), "lastName" : "Aaronson", "firstName" : "Arthur" }

CRUD: Question 6

Consider the following collection:

{ \_id : 3, a : [3, 7] }

{ \_id : 4, a : [9, 7] }

{ \_id : 6, a : [3, 4] }

Which of the documents will be returned by the query?

db.collection.find( { a : { $lt : 5, $gt : 5 } } )

Check all that apply.

Check all answers that apply:

A.{ \_id : 3, a : [3, 7] }

B.{ \_id : 4, a : [9, 7] }

C.{ \_id : 6, a : [3, 4] }

7.Given a collection "col" composed of only these documents:

{ "\_id" : 1 , "a" : 1 , "b" : [ 3 ] }

{ "\_id" : 11 , "a" : "5" , "b" : [ 5 , 3 , 7 ] }

{ "\_id" : 111 , "a" : 1 , "b" : [ 3 , 5 ] }

{ "\_id" : 12 , "a" : "5" , "b" : ["alpha", 5] }

which one of the queries below will return only the following documents from this collection?

{ "\_id" : 11 , "a" : "5" , "b" : [ 5 , 3 , 7 ] }

{ "\_id" : 111 , "a" : 1 , "b" : [ 3 , 5 ] }

Choose the best answer:

A.db.col.find( { "b" : { "$and" : [ 3 , 5 ] } } )

B.db.col.find( { "b" : 3, "b" : 5} )

C.db.col.find( { "b" : { "$all" : [ 3 , 5 ] } } )

D.db.col.find( { "b" : { "$in" : [ 3 , 5 ] } } )

E.db.col.find( { "b" : [ 3 , 5 ] } )

8.The buildings collection has documents of the following form:

{

"\_id" : { "address" : "123 Example Street" },

"type" : "residential",

"residents" : [ { "name" : "Alice", "age" : 35 },

{ "name" : "Bob", "age" : 22 } ]

}

You would like to find all documents in the buildings collection where at least one of the residents subdocuments has age : 35.

Which of the following queries will accomplish this? Check all that apply.

Check all answers that apply:

A.db.buildings.find( { “residents.age” : 35 } )

B.db.buildings.find( { residents : 35 } )

C.db.buildings.find( { residents : { age : 35 } } )

9.Given a collection "col" composed of only these documents:

{ "\_id" : 1 , "a" : 1 , "b" : [ 3 ] }

{ "\_id" : 11 , "a" : "5" , "b" : [ 5 , 3 , 7 ] }

{ "\_id" : 111 , "a" : 1 , "b" : [ 3 , 5 ] }

{ "\_id" : 12 , "a" : "5" , "b" : ["alpha", 5] }

which one of the queries below will return only the following documents from this collection?

{ "\_id" : 11 , "a" : "5" , "b" : [ 5 , 3 , 7 ] }

{ "\_id" : 111 , "a" : 1 , "b" : [ 3 , 5 ] }

Choose the best answer:

A.db.col.find( { "b" : { "$and" : [ 3 , 5 ] } } )

B.db.col.find( { "b" : 3, "b" : 5} )

C.db.col.find( { "b" : { "$all" : [ 3 , 5 ] } } )

D.db.col.find( { "b" : { "$in" : [ 3 , 5 ] } } )

E.db.col.find( { "b" : [ 3 , 5 ] } )

CRUD: Question 10

Which of the commands below will append the value 5 to an array in the "a" field if and only if the field "a" has the following properties?

Field "a" exists.

Field "a" is an array field.

The \_id is 12.

Choose the best answer:

A.db.sample.updateOne( { \_id : 13 , a : { $exists : true } } , { $push : { a : 5 } } )

B.db.sample.updateOne( { \_id : 13 } , { $addToSet : { a : 5 } } )

C.db.sample.updateOne( { \_ id : 13 } , { $push : { a : 5 } } )

D.db.sample.updateOne( { $push : { a : 5 } } , { \_id : 12 } )

E.db.sample.updateOne( { \_id : 13 }, { $pop : { a : 5 } } )

CRUD: Question 11

Consider the query:

db.sample.find( { "b" : { "$gte" : 4, "$lte" : 6 } } )

Which of the following documents would be retrieved by this query? Assume all documents are in the sample collection. Check all that apply.

Check all answers that apply:

A.{ "\_id" : 1 , "a" : 1 , "b" : 10 }

B.{ "\_id" : 13 , "a" : 2 , "b" : 6 }

C.{ "\_id" : 24 , "a" : 3 , "b" : 3 }

D.{ "\_id" : 25 , "a" : 3 , "b" : 7 }

E.{ "\_id" : 26 , "a" : 3 , "b" : 5 }

12.

How would the following query change if we set { upsert : true } as the third parameter? Check all that apply.

db.sample.updateMany( { a : 4 }, { $set : { b : 2, d : -4 } ,{upsert:true}} )

Check all answers that apply:

A.The upsert will insert a document if no documents exist where "a" is equal to 4.

B.An upsert will remove a document, if found, before inserting a new document.

C.The upsert will both insert and update a document.

13.Which documents will be returned by the following query?

db.cars.find( { "miles" : { "$lt" : 300 } , "model.reviews.stars" : { "$gte" : 4 } } )

Check all answers that apply:

A.{ "\_id" : 21, "year" : 2017, "miles" : 200, "model" : { "reviews": [ { "user" : "fred", "comment" : "Great!" , "stars" : 5 }, { "user" : "bill" , "comment" : "could be better" , "stars" : 3 } ] } }

B.{ "\_id" : 3, "year" : 2001, "miles" : 278, "model" : { "name": "627", "reviews": { "user": "brad", "comment": "very slick", "stars": 5} } }

C.{ "\_id" : 971, "year" : 2000, "miles" : 300, "model" : { "name": "Corolla", "reviews": [ { "user" : "fred", "comment" : "Great!" , "stars" : 5 }, { "user" : "bill" , "comment" : "could be better" , "stars" : 3 } ] } }

D.{ "\_id" : 3441, "year" : 2007, "miles" : 256, "model" : { "name": "Continental", "reviews": [ { "user" : "bridgett", "comment" : "average" , "stars" : 3 }, { "user" : "andreas" , "comment" : "Really liked the color of the leather" , "stars" : 3 } ,{ "user" : "manafort" , "comment" : "not russian enough" , "stars" : 2 } ] } }

E.{ "\_id" : 671, "year" : 1983, "miles" : 2000, "model" : { "name": "Fiesta", "reviews": [ { "user" : "marjory" , "comment" : "loved it" , "stars" : 10 } ] } }

概念：

1.Suppose you would like to store an account balance of $10.45 (ten dollars and forty-five cents) in an interest-bearing account, with the balance stored in MongoDB, accurate to 1/1000 of a dollar.

Which of the following is/are in a format that would ensure no loss of precision? You may assume that you

r application will correctly convert from the stored value to dollars using the scaleFactor value shown. Check all that apply.

Check all answers that apply:

A.{ balance : NumberInt(10450), currency : "dollars", scaleFactor : NumberInt(1000) }

B.{ balance : NumberDecimal("10.45"), currency : "dollars", scaleFactor : NumberInt(1) }

C.{ balance : 10.45, currency : "dollars", scaleFactor : NumberInt(1) }

2.Which of the following are advantages of the NumberDecimal type over a NumberLong type?

Check all answers that apply:

A.NumberDecimal can perform integer arithmetic without loss of precision, while NumberLong cannot.

B.NumberDecimal will perform floating point arithmetic without loss of precision, while NumberLong may not.

C.NumberDecimal offers higher precision than NumberLong and therefore is more suited to currency-related uses.

3.Which one of the following best describes when it is recommended to represent one-to-many relationships using multiple collections and references?

Choose the best answer:

A.When the "many" is large

B.When the "many" is small (1-9, for example)

C.This is always the recommended option

D.This is never recommended

E.When disk space is limited

4.Of the following, which is the most important MongoDB schema design consideration?

Choose the best answer:

A.Normalizing your data

B.Limiting the amount of disk space used

C.Making your schema human readable

D.Making your design extensible

E.Matching the data access patterns of your application

5.DATA MODELING: Question 5

Suppose you are working with a library catalog system containing collections for patrons, publishers, and books. Book documents maintain a field "available" that identifies how many copies are currently available for checkout.

There is also a field "checkout" that holds a record of all patrons that are currently borrowing a copy of the book. For example, the document below indicates that the library owns four copies of "Good Book". Three are currently available for checkout. One has been checked out by patron "33457".

Which of the following is the primary advantage to this design?

Choose the best answer:

A.We get the ability to retrieve a complete checkout history for books as a side effect.

B.We maintain the ability to update patrons, publishers, and books independently and safely rely on MongoDB's foreign key constraints.

C.Books will not grow beyond the MongoDB document size limit.

D.Can retrieve all data about a book, and the name and address of the patrons who have the book checked out with a single find() query.

E.Can make atomic updates as books are checked out or turned in.

6.DATA MODELING: Question 6

You have two types of documents -- A documents and B documents. There is a one-to-many relationship from A to B. Which of the following are conditions that favor embedding B documents within A documents rather than referencing?

Check all answers that apply:

A.All of the Bs associated with a given A are required every time

B.The set of Bs associated with a given A changes often

C.The number of Bs is bounded and relatively small

D.Report a Problem

1.You run a medical records database. Each visit, a patient fills out a set of forms (2-10 of them, depending on the visit).

You already plan to record, in a document, all of the important data for later use (and for analytics), but are also required to keep scans of the forms in your records for regulatory compliance reasons, even though your system won't use the scanned forms in routine business.

What would be the best way to store the scanned forms?

Choose the best answer:

A.Put the scanned forms in a separate collection, and reference them by their \_id in an array in the document that represents the patient's visit

B.Embed the scanned forms in the document representing the patient's visit, but do not index the fields that contain the forms

C.Embed the scanned forms in the document representing the patient's visit, and also Index the fields that use the scanned documents as values

D.Delete the scanned forms once all of their relevant information has been imported digitally

E.Keep only the scanned forms because they contain all required information, but index the binary data representing the forms.

2.

If you wanted to use MongoDB to track cash balances for each of an individual's financial accounts, which one of the following schema designs could you use for a person to transfer cash balances atomically between his/her own accounts atomically (ie, one update per transfer)?

Choose the best answer:

A.A document for each account

B.A document for each account and another linked document for the balances

C.A document for each customer with an array of account documents

D.A document for each customer and a document in another collection for each account with its balance

E.None of these

INDEXES AND PERFORMANCE: Question 1

Which of the following are advantages of WiredTiger over MMAPv1? Check all that apply.

Check all answers that apply:

A.More granular concurrency

B.Better utilization of multi-core servers

D.Compression

2.Which of the following are officially supported storage engines in MongoDB 3.4?

Check all answers that apply:

A.WiredTiger

B.MMAPv1

C.RocksDB

3.Which of the following MongoDB supported storage engines support document level concurrency control?

Check all answers that apply:

A.Encrypted Storage Engine

B.WiredTiger

C.MMAPv1

4.INDEXES AND PERFORMANCE: Question 4

You would like to store the data from a collection as BSON. Which of the following utilities will help to do this?

Choose the best answer:

A.mongoexport

B.mongodump

C.mongofiles

E.bsondump

F.mongorestore

5.INDEXES AND PERFORMANCE: Question 5

Within the travel collection, all documents have the same set of fields as this document:

{

"\_id" : ObjectId("5ab0615f5b3e785dd5fa3705"),

"name" : "Randy Salazar",

"age" : 60,

"visited" : [

"Israel",

"São Tomé and Príncipe",

"Malta",

"Ethiopia"

]

}

5.Consider the following index:

db.travel.createIndex({ "name": 1, "age": 1 })

Which of the following queries will result in a covered query? Check all that apply.

查询中的所有字段都是索引的一部分，并且

结果中返回的所有字段都在同一个索引中。

Check all answers that apply:

A.db.travel.find( {"name": "Randy Salazar"}, { "age":1, "\_id": 0 } )

B.db.travel.find( {"age": {"$gt": 18}}, { "visited": 0, "\_id": 0 } )

C.db.travel.find( {"age": {"$ne": 12} }, { "name": 1, "age":1, "\_id": 0 } )

D.db.travel.find( {"name": /^R/ }, { "name": 1, "age":1} )

E.db.travel.find( {"name": "Randy Salazar", "age": 60 }, {"name": 1, "\_id": 0 } )

下面这个不会使用索引

 db.travel.find( {"age": 60 }, {"name": 1, "\_id": 0 } )

E:



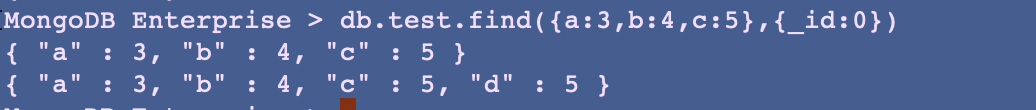
下面都是覆盖查询：

db.test.find({a:31,b:1},{\_id:0,a:1}).explain()

db.test.find({a:31,b:1,c:4},{\_id:0,a:1,b:1,c:1}).explain()

下面都不是覆盖查询：

db.test.find({a:31,b:1,c:4},{\_id:0}).explain()



db.test.find({a:31,b:1,c:4}).explain()

6.INDEXES AND PERFORMANCE: Question 6

You have created the following index on the travel collection:

db.travel.createIndex( { "name" : 1, "age" : -1, "visited" : 1 } )

Which of the following queries will be able to fulfill the query without an in-memory sort (i.e., it's able to use the index to sort)? Check all that apply.

Check all answers that apply:

A.db.travel.find( { "age" : {"$gt": 22} } ).sort({ "visited": -1 })

B.db.travel.find( {"name": "Randy Salazar"} ).sort({"age": -1} )

C.db.travel.find( {"name": "Lillian Hernandez", "visited" : /^I/ } ).sort( {"age": 1} )

E:

{ "\_id" : ObjectId("5e0d8eb22a57eae4d7e3b7d9"), "a" : 1, "b" : 2, "c" : 3 }

{ "\_id" : ObjectId("5e0d8ebf2a57eae4d7e3b7da"), "a" : 31, "b" : 1, "c" : 4 }

{ "\_id" : ObjectId("5e0d8ec82a57eae4d7e3b7db"), "a" : 2, "b" : 2, "c" : 2 }

下面两个都用了索引：

db.test.find({"a":31,"c":4}).sort({"b":1}).explain()

db.test.find({"a":31,"b":2}).sort({"c":1}).explain()



7.When building indexes in MongoDB, what are features of a compound index? Check all that apply.

Check all answers that apply:

A.Compound indexes can be more selective, examining fewer documents in order to fulfill the search criteria.

B.A compound index can be used just like a single-field index on its first field.

C.A compound index allows for covered queries in more cases than a single-field index.

Report a Problem

8.Consider the collection travel which contains the following 2 documents:

{

"\_id" : ObjectId("5ab0615f5b3e785dd5fa3705"),

"name" : "Randy Salazar",

"age" : 60,

"visited" : [

"Israel",

"São Tomé and Príncipe",

"Malta",

"Ethiopia"

]

}

{

"\_id" : ObjectId("5ab0615f5b3e785dd5fa3706"),

"name" : "Lillian Hernandez",

"age" : 38,

"visited" : [

"Canary Islands"

]

}

And the following command is run:

db.travel.createIndex({visited: 1, age:1})

How many index entries will be created as a result of this command?

Choose the best answer:

A.0

B.2

C.3

D.4

E.5 or more

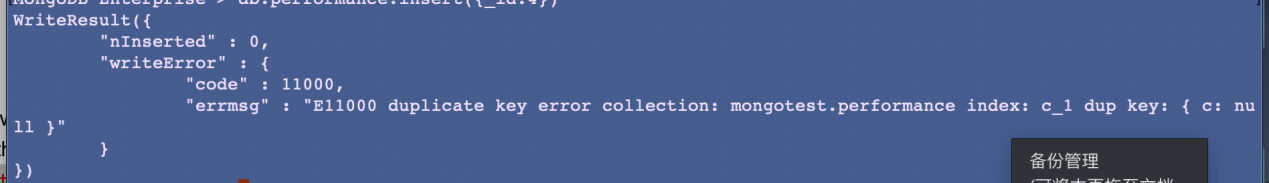
9.Which of the following is/are true of unique indexes? Check all that apply.

Check all answers that apply:

A.No two documents in the collection may be missing all of the fields in the index

B.An index may not be both unique and hashed

C.A unique index must be created on a collection before any documents are inserted



10.INDEXES AND PERFORMANCE: Question 10

Indexes may sometimes reduce the throughput of which of the following types of operations? Check all that apply.

You may assume for this problem that your entire data set, and your indexes, are already in RAM when you perform the query, in order to avoid considerations of disk access time.

Check all answers that apply:

A.Updates

B.Deletes

C.Finds

11.Indexes are created to optimize the execution of certain operations.

However, indexes may reduce the throughput of other operations.

From the following, select all operations that might experience reduced throughput by adding indexes to your collections

You may assume for this problem that your entire data set, and your indexes, are already in RAM when you perform the query, in order to avoid considerations of disk access time.

Check all answers that apply:

A.Bulk Delete

B.Inserts

C.Finds with Sort

Aggregation:

1:

AGGREGATION: Question 1

The agencies collection contains the following 3 documents:

{ "\_id" : ObjectId("565385949a9c0d40e8e6f83f"),

"type" : "A",

"agents" : [ "red", "blue", "green" ],

"files" : [ "a", "a", "b", "b" ] }

{ "\_id" : ObjectId("565385c39a9c0d40e8e6f840"),

"type" : "A",

"agents" : [ "yellow", "blue" ],

"files" : [ "b" ] }

{ "\_id" : ObjectId("5653862c9a9c0d40e8e6f841"),

"type" : "A",

"agents" : [ "red", "green" ],

"files" : [ "a" ] }

The following aggregation query is run against this collection:

db.agencies.aggregate( [

{$unwind: "$agents"},

{$unwind: "$files"},

{$match: {files: "b"}},

{$group: {\_id: "$type", count: {$sum: 1}}}])

What is the result?

{ "\_id" : "A", "count" : 8 }

2.You have the following collection:

> db.incidents.findOne()

{

"\_id" : ObjectId("5693e0955b43583dae8b61fa"),

"date" : ISODate("2007-01-02T02:04:00Z"),

"type" : "Vandalism",

"location\_string" : "58XX E NORTHWEST HWY, DALLAS, TX",

"location" : {

"type" : "Point",

"coordinates" : [

-100.884627,

32.705833

]

},

"link" : "http://spotcrime.com/crime/363082-285b7604c923ca660aeb301e36843bbd",

"description" : "CRIMINAL MISCHIEF"

}

You would like to capture just the "location" subdocument, and pass that to the next stage of the aggregation pipeline. Which aggregation stage can do this most easily (i.e., with the shortest syntax)?

Choose the best answer:

A.$limit

B.$facet

C.$sample

D.$out

E.$replaceRoot

AGGREGATION: Question 3

Which of the following will improve the performance of the aggregation pipeline? You may assume that manually performing an optimization that the database would perform on its own will not affect performance.

Check all answers that apply:

A.Placing a $match immediately before a $sort stage, rather than after

B.Putting $match, $limit, and $skip as early as possible

C.Combining consecutive $match stages.

4.AGGREGATION: Question 4

You have the following two collections in the database:

> db.people.find()

{ "\_id" : 1, "lastName" : "Ansari", "firstName" : "Alice", "city" : "Little Rock", "state" : "Arkansas" }

{ "\_id" : 2, "lastName" : "Bai", "firstName" : "Brian", "city" : "Patna", "state" : "Bihar" }

{ "\_id" : 3, "lastName" : "Chang", "firstName" : "Cindy", "city" : "Palo Alto", "state" : "California" }

{ "\_id" : 4, "lastName" : "Díaz", "firstName" : "Diego" }

> db.states.find()

{ "\_id" : "Alabama", "motto" : "Audemus jura nostra defendere" }

{ "\_id" : "Alaska", "motto" : "North to the Future" }

{ "\_id" : "Arizona", "motto" : "Ditat Deus" }

{ "\_id" : "Arkansas", "motto" : "Regnat Populus" }

{ "\_id" : "California", "motto" : "Eureka" }

You perform the following aggregation:

db.people.aggregate( [ { $lookup : { from : "states", localField : "state", foreignField : "\_id", as : "stateInfo" } } ] )

Which of the following will be the result?

Choose the best answer:

A.{ "\_id" : 1, "lastName" : "Ansari", "firstName" : "Alice", "city" : "Little Rock", "state" : "Arkansas", "stateInfo" : [ { "\_id" : "Arkansas", "motto" : "Regnat Populus" } ] }

{ "\_id" : 2, "lastName" : "Bai", "firstName" : "Brian", "city" : "Patna", "state" : "Bihar", "stateInfo" : [ ] }

{ "\_id" : 3, "lastName" : "Chang", "firstName" : "Cindy", "city" : "Palo Alto", "state" : "California", "stateInfo" : [ { "\_id" : "California", "motto" : "Eureka" } ] }

{ "\_id" : 4, "lastName" : "Díaz", "firstName" : "Diego", "stateInfo" : [ ] }

B.{ "\_id" : 1, "lastName" : "Ansari", "firstName" : "Alice", "city" : "Little Rock", "state" : "Arkansas", "stateInfo" : [ { "\_id" : "Arkansas", "motto" : "Regnat Populus" } ] }

{ "\_id" : 3, "lastName" : "Chang", "firstName" : "Cindy", "city" : "Palo Alto", "state" : "California", "stateInfo" : [ { "\_id" : "California", "motto" : "Eureka" } ] }

C.{ "\_id" : 1, "lastName" : "Ansari", "firstName" : "Alice", "city" : "Little Rock", "state" : "Arkansas", "stateInfo" : [ { "\_id" : "Arkansas", "motto" : "Regnat Populus" } ] }

{ "\_id" : 2, "lastName" : "Bai", "firstName" : "Brian", "city" : "Patna", "state" : "Bihar", "stateInfo" : [ ] }

{ "\_id" : 3, "lastName" : "Chang", "firstName" : "Cindy", "city" : "Palo Alto", "state" : "California", "stateInfo" : [ { "\_id" : "California", "motto" : "Eureka" } ] }

D.{ "\_id" : 1, "lastName" : "Ansari", "firstName" : "Alice", "city" : "Little Rock", "state" : "Arkansas", "stateInfo" : [ { "\_id" : "Arkansas", "motto" : "Regnat Populus" } ] }

{ "\_id" : 3, "lastName" : "Chang", "firstName" : "Cindy", "city" : "Palo Alto", "state" : "California", "stateInfo" : [ { "\_id" : "California", "motto" : "Eureka" } ] }

{ "\_id" : 4, "lastName" : "Díaz", "firstName" : "Diego", "stateInfo" : [ ] }

E.{ "\_id" : 1, "lastName" : "Ansari", "firstName" : "Alice", "city" : "Little Rock", "state" : "Arkansas", "stateInfo" : [ { "\_id" : "Arkansas", "motto" : "Regnat Populus" } ] }

Report a Problem

6.AGGREGATION: Question 6

Consider the following schema:

> db.grades.findOne()

{

"\_id" : ObjectId("590b3e2e5e8cdc2dbd114aa4"),

"student\_name" : "Priya",

"homework\_scores" : [

97,

100,

99,

100,

100,

93,

100

],

"quiz\_scores" : [

96,

98,

92

],

"test\_scores" : [

100,

99

]

}

...

You would like to perform an aggregation query. Your goal is to keep all existing fields, and also have fields for the average score in each of the three arrays. Which of the following aggregation stages can accomplish this? Check all that apply.

Check all answers that apply:

A.$addFields

B.$project

C.$count

7.AGGREGATION: Question 7

You have a collection with the following schema:

db.people.findOne( { \_id : "naomi" } )

{ "\_id" : "naomi", "parents" : [ "leslie", "will" ] }

You would like to find each person's parents, and grandparents (their parents' parents). You perform the following query, with only the stage operator missing:

db.people.aggregate( [

{

STAGE :

{

from : "people",

startWith : "$parents",

connectFromField : "parents",

connectToField : "\_id",

as : "ancestors",

maxDepth : 1

}

}

] )

What is the top-level operator that is missing (and replaced by the word STAGE, above)?

Choose the best answer:

A.$lookup

B.$graphLookup

C.$replaceRoot

D.$count

E.$bucket

8.The people collection contains the following two documents:

> db.people.find()

{ "\_id" : ObjectId("5900efbf0986fdb6574b86f6"), "lastName" : "Cross", "firstName" : "William" }

{ "\_id" : ObjectId("5900efbf0986fdb6574b86f7"), "lastName" : "Leniz", "firstName" : "Nathan" }

You perform the following aggregation:

> db.people.aggregate( [ { $project : { lastName : 1 } } ] )

Which fields are displayed in the result set? Check all that apply.

Check all answers that apply:

9.AGGREGATION: Question 9

You have a collection with the following schema:

> db.users.find()

{ "\_id" : ObjectId("590a3de35e8cdc2dbd114aa0"), "username" : "alice", "numFriends" : 22 }

{ "\_id" : ObjectId("590a3df45e8cdc2dbd114aa1"), "username" : "brian", "numFriends" : 3 }

...

The collection has approximately 200,000 documents, each representing a single user.

You would like to segment (or "bin") your data into distinct group ranges based on the "numFriends" field, counting the number of documents in each segment.

Which of the following aggregation stages could accomplish this in a single-stage pipeline? Check all that apply.

Check all answers that apply:

A.$bucket

B.$bucketAuto

C.$count

10.AGGREGATION: Question 10

Consider the stuff collection:

> db.stuff.find()

{ "\_id" : 1, "x" : 1, "y" : [ 1, 2, 3, 4 ] }

{ "\_id" : 2, "x" : 2, "y" : [ 1, 2, 3, 4 ] }

{ "\_id" : 3, "x" : 3, "y" : [ 1, 2, 3, 4 ] }

{ "\_id" : 4, "x" : 4, "y" : [ 1, 2, 3, 4 ] }

{ "\_id" : 5, "x" : 5, "y" : [ 1, 2, 3, 4 ] }

If you perform the following aggregation query, how many documents will be in the

otherStuff

collection when you are done? You can assume that the

otherStuff

collection does not have any indexes except for its standard index on

\_id

.

> db.otherStuff.drop()

> db.stuff.aggregate( [ { $unwind : "$y" }, { $project : { x : 1, y : 1, \_id : 0 } }, { $out : "otherStuff" } ] )

Choose the best answer:

The answer depends on the state of the otherStuff collection initially.

A.1

B.4

C.5

D.20

1.REPLICATION: Question 1

Your people collection contains the following documents:

{ "\_id" : ObjectId("57fd59a2d630a0fd9685a148"), "firstName" : "Arthur", "lastName" : "Aaronson", "state" : "WA", "city" : "Seattle", "likes" : [ "dogs", "cats" ] }

{ "\_id" : ObjectId("57fd59a2d630a0fd9685a149"), "firstName" : "Beth", "lastName" : "Barnes", "state" : "WA", "city" : "Richland", "likes" : [ "forest", "desert" ] }

{ "\_id" : ObjectId("57fd59a2d630a0fd9685a14a"), "firstName" : "Charlie", "lastName" : "Carlson", "state" : "CA", "city" : "San Diego", "likes" : [ "desert", "beach" ] }

{ "\_id" : ObjectId("57fd59a2d630a0fd9685a14b"), "firstName" : "Dawn", "lastName" : "Davis", "state" : "WA", "city" : "Seattle", "likes" : [ "forest", "mountains" ] }

You perform the following query:

db.people.updateMany( { state : "NY" }, {"$push": { "likes": "hot dogs" }} )

How many oplog entries are created as a result of this query?

Choose the best answer:

A.0

B.1

C.2

D.3

E.4

REPLICATION: Question 2

Under which circumstances is it wise to use delayed replica set members?

Choose the best answer:

A.To allow secondary reads with max staleness.

B.To enable easy-to-query secondary nodes.

C.To allow copies of data that are frozen in time and never change.

D.To create indexes on secondary nodes.

E.To provide resilience against application level corruption.

REPLICATION: Question 3

Which of the following describe the primary reasons MongoDB supports replication?Check all that apply

Check all answers that apply:

A.Ensure database high availability

B.Allow for data partitioning

C.Ensure redundant copies of data

1. How many times has replica set rs0 been reconfigured?

rs.config()

{

"\_id" : "rs0",

"version" : 332,

"protocolVersion" : NumberLong(1),

"members" : [

{

"\_id" : 0,

"host" : "192.168.103.100:26000",

"arbiterOnly" : false,

"buildIndexes" : true,

"hidden" : false,

"priority" : 1,

"tags" : {

},

"slaveDelay" : NumberLong(0),

"votes" : 1

},

{

"\_id" : 1,

"host" : "192.168.103.100:26001",

"arbiterOnly" : false,

"buildIndexes" : true,

"hidden" : false,

"priority" : 1,

"tags" : {

},

"slaveDelay" : NumberLong(0),

"votes" : 1

},

{

"\_id" : 2,

"host" : "192.168.103.100:26002",

"arbiterOnly" : false,

"buildIndexes" : true,

"hidden" : false,

"priority" : 1,

"tags" : {

},

"slaveDelay" : NumberLong(0),

"votes" : 1

}

],

"settings" : {

"chainingAllowed" : true,

"heartbeatIntervalMillis" : 2000,

"heartbeatTimeoutSecs" : 10,

"electionTimeoutMillis" : 10000,

"catchUpTimeoutMillis" : -1,

"catchUpTakeoverDelayMillis" : 30000,

"getLastErrorModes" : {

},

"getLastErrorDefaults" : {

"w" : 1,

"wtimeout" : 0

},

"replicaSetId" : ObjectId("5ab0faf319ffd5021c4270db")

}

}

How many times has replica set rs0 been reconfigured?

Choose the best answer:

A.331

B.0

C.330

D.1

E.3

5.REPLICATION: Question 5

Your deployment is a replica set, and your people collection contains the following documents:

{ firstName : "Arthur", lastName: "Aaronson", state: "WA", city: "Seattle", likes: [ "dogs", "cats", "hiking" ] }

{ firstName : "Beth", lastName: "Barnes", state: "CA", city: "Richland", likes: [ "forest", "desert" ] }

{ firstName : "Charlie", lastName: "Carlson", state: "CA", city: "San Diego", likes: [ "desert", "beach" ] }

{ firstName : "Dawn", lastName: "Davis", state: "WA", city: "Seattle", likes: [ "forest", "mountains", "hiking" ] }

You perform the following update:

db.people.updateMany( { state : "WA" }, { $pull : { likes : "hiking" } } )

How many oplog entries are created as a result of this updateMany operation?

Choose the best answer:

A.0

B.1

C.2

D.3

E.4

Sharding：

1.In a sharded cluster, which of the following require unique values? Check all that apply.

Check all answers that apply:

A.Single key index

B.A compound-field shard key index

C.The \_id index

2.Suppose you have a sharded cluster. Given a compound shard key of { "a" : 1 , "b" : 1 }, which of the following queries will enable a mongos to target queries at specific shards rather than doing a scatter gather query? Check all that apply.

Check all answers that apply:

A.db.sample.find( { "b" : { "$gt" : 5 }} )

B.db.sample.find( { "a" : { "$lte" : 7 } } )

C.db.sample.find( { "a" : 12, "b" : { "$gt" : "jello" } } )

3.Consider the following chunks:

{ "productId" : { "$minKey" : 1 } } -->> { "productId" : 9294 } on : shard0000 { "t" : 2, "i" : 0 }

{ "productId" : 9294 } -->> { "productId" : 18684 } on : shard0002 { "t" : 3, "i" : 0 }

{ "productId" : 18684 } -->> { "productId" : 27851 } on : shard0003 { "t" : 4, "i" : 0 }

{ "productId" : 27851 } -->> { "productId" : 36852 } on : shard0004 { "t" : 5, "i" : 0 }

At a certain point, chunk defined by { "productId" : 18684 } -->> { "productId" : 27851 } on : shard0003 { "t" : 4, "i" : 0 } gets migrated to shard0000

While the migration operation is taking place (before completing), which of the following is/are true?Check all that apply.

Check all answers that apply:

A.All queries on documents with productId value ranging between 18684 and 27851 will be routed to shard0000

B.All queries on documents with productId value ranging between 18684 and 27850 will be routed to shard0003

C.Documents matching query { "productId": 27851 } will be found on shard0004

SHARDING: Question 4

For the field position we can have only one of the following set of values (the only possible values for this field):

position = one\_of( "operator", "supervisor", "manager")

Given this field value cardinality (number of different values), if we decide to use position as our shard key, single field shard key, which type of problems can occur to the shard cluster?

Check all answers that apply:

A.Creation of indivisible jumbo chunks

B.Reduced number of targeted queries

C.Limits the size of our shard key index

5.Suppose you have a sharded cluster with a status as follows:

mongos> sh.status()

--- Sharding Status ---

sharding version: {

"\_id" : 1,

"version" : 3,

"minCompatibleVersion" : 3,

"currentVersion" : 4,

"clusterId" : ObjectId("51de8630162b88d59cd7b006")

}

shards:

{ "\_id" : "shard0000", "host" : "localhost:30000" }

{ "\_id" : "shard0001", "host" : "localhost:30001" }

{ "\_id" : "shard0002", "host" : "localhost:30002" }

{ "\_id" : "shard0003", "host" : "localhost:30003" }

{ "\_id" : "shard0004", "host" : "localhost:30004" }

databases:

{ "\_id" : "admin", "partitioned" : false, "primary" : "config" }

{ "\_id" : "test", "partitioned" : true, "primary" : "shard0001" }

test.products

shard key: { "productId" : 1 }

chunks:

shard0000 2

shard0002 3

shard0003 2

shard0004 2

shard0001 2

{ "productId" : { "$minKey" : 1 } } -->> { "productId" : 9294 } on : shard0000 { "t" : 2, "i" : 0 }

{ "productId" : 9294 } -->> { "productId" : 18684 } on : shard0002 { "t" : 3, "i" : 0 }

{ "productId" : 18684 } -->> { "productId" : 27851 } on : shard0003 { "t" : 4, "i" : 0 }

{ "productId" : 27851 } -->> { "productId" : 36852 } on : shard0004 { "t" : 5, "i" : 0 }

{ "productId" : 36852 } -->> { "productId" : 46047 } on : shard0000 { "t" : 6, "i" : 0 }

{ "productId" : 46047 } -->> { "productId" : 55450 } on : shard0002 { "t" : 7, "i" : 0 }

{ "productId" : 55450 } -->> { "productId" : 64644 } on : shard0004 { "t" : 8, "i" : 0 }

{ "productId" : 64644 } -->> { "productId" : 73769 } on : shard0003 { "t" : 9, "i" : 0 }

{ "productId" : 73769 } -->> { "productId" : 82950 } on : shard0002 { "t" : 10, "i" : 0 }

{ "productId" : 82950 } -->> { "productId" : 91983 } on : shard0001 { "t" : 10, "i" : 1 }

{ "productId" : 91983 } -->> { "productId" : { "$maxKey" : 1 } } on : shard0001 { "t" : 1, "i" : 10 }

Which shards would be involved in answering the following query:

use test;

db.products.find({ "$or": [ {"productId" : {"$gte" : 91983} , {"productId" : { "$lt" : 18684} } ] } );

Check all that apply.

Check all answers that apply:

A.shard0001

B.shard0003

C.shard0004

D.shard0000

E.shard0002

6.Consider a collection of users with the following fields and possible values:

* phone\_number -- a 10-digit telephone number (string)
* eye\_color -- "brown", "hazel", "blue", "green", or "other" (string)
* weight -- an integer in pounds; known for about half the users
* started\_driving\_at -- the age at which the user got their driver's license in years. For most users this is 15, 16, 17, or 18. (integer)
* \_id -- automatically created on insert (ObjectId)

Assuming the data-access patterns also support your choice, which of these fields would make the best shard key?

1. phone\_number
2. eye\_color

C.weight

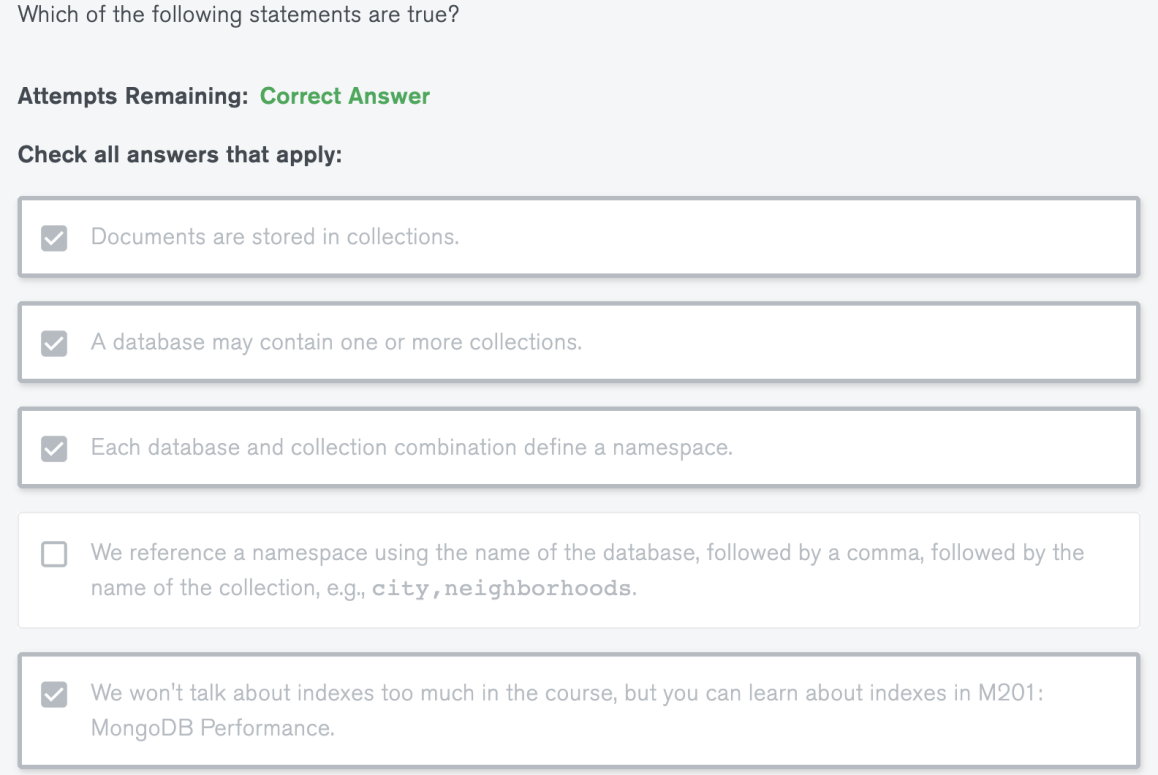
D.started\_driving\_at

E.\_id

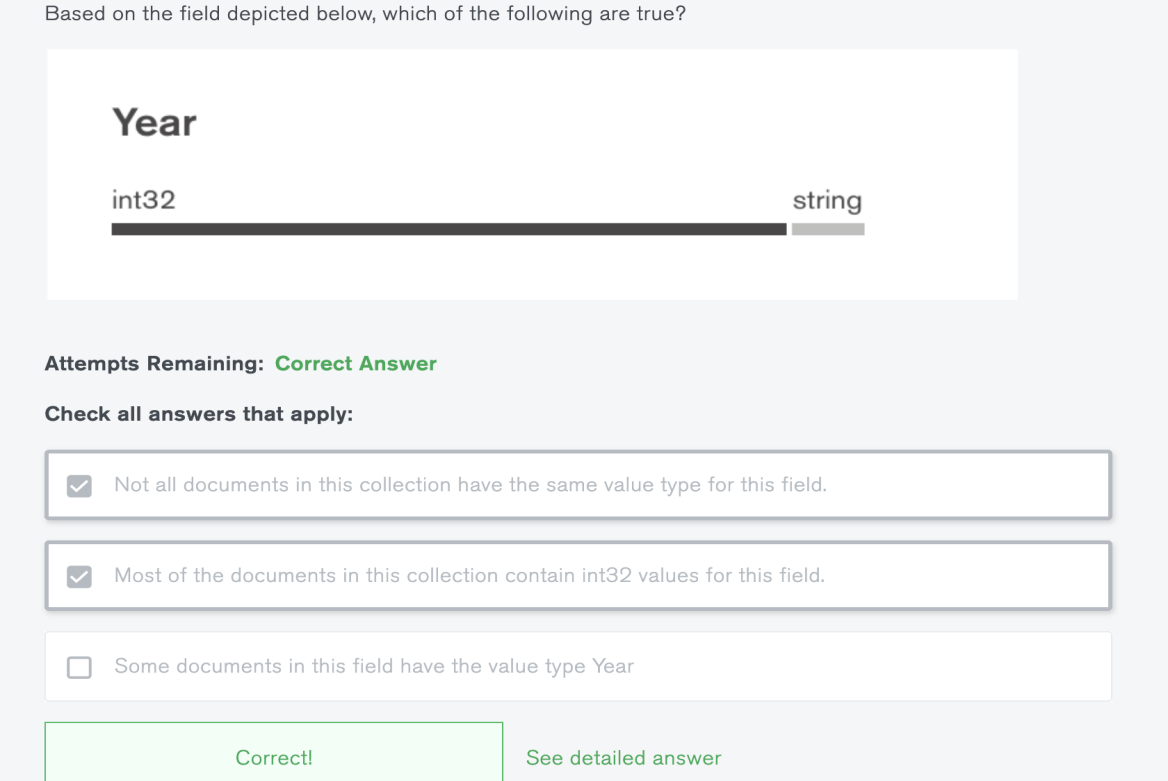
练习题：

Mongodb Basic基础

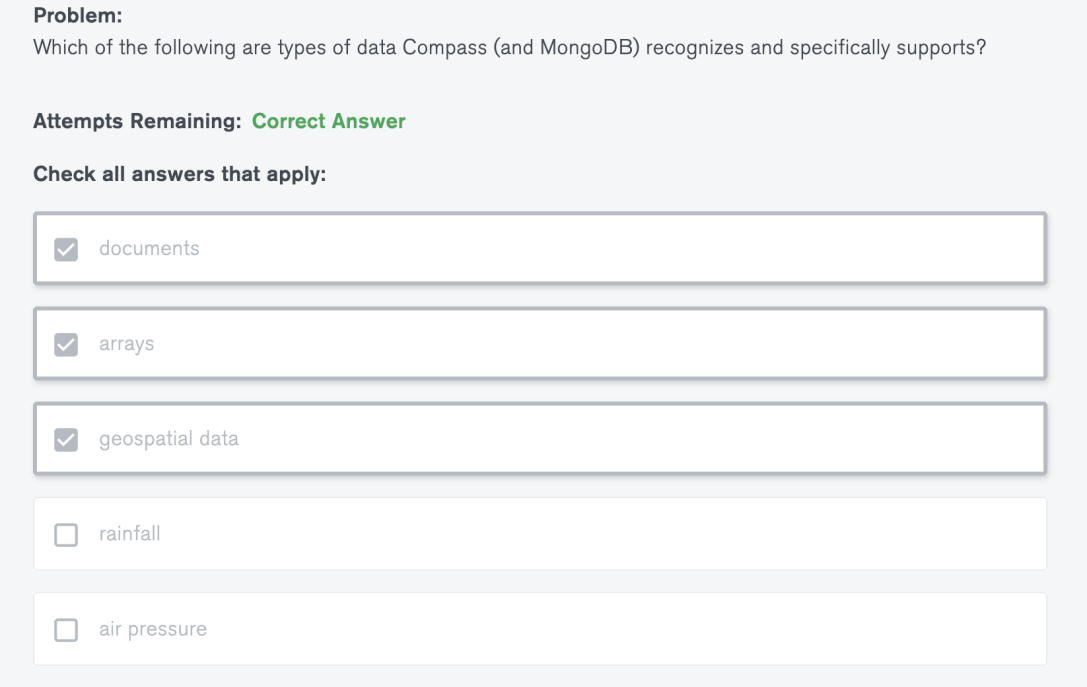
1：



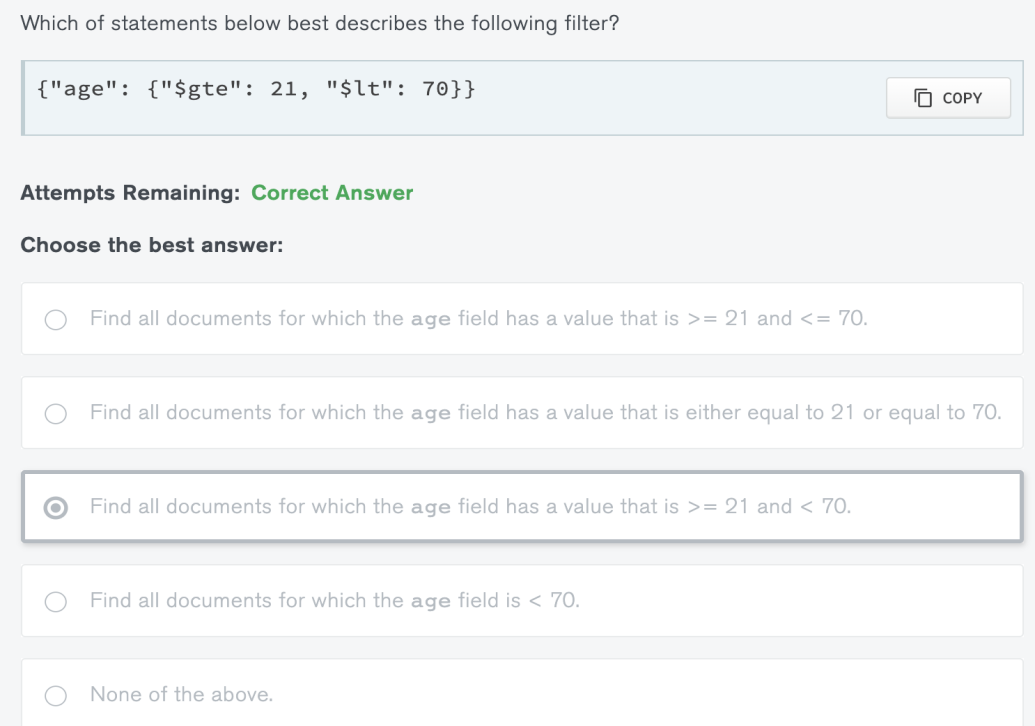
2.



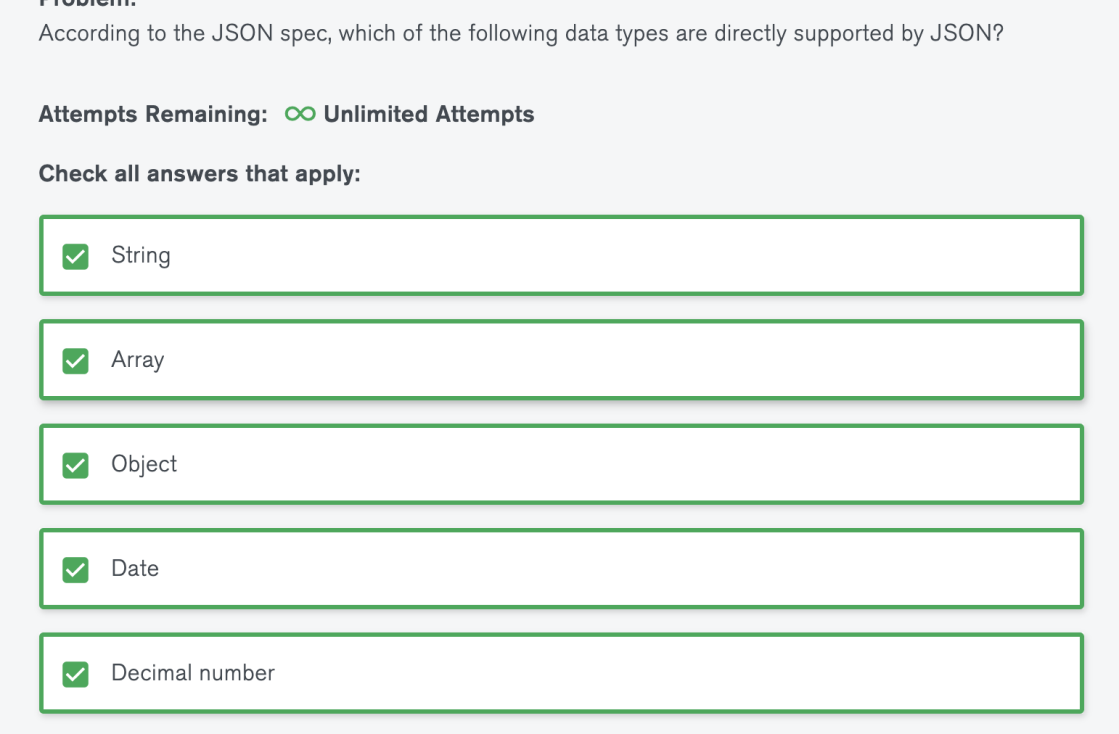
3.



4.



5.



二数据模型：

1Which of the following statements are true about data modeling using MongoDB?

Attempts Remaining:Correct Answer

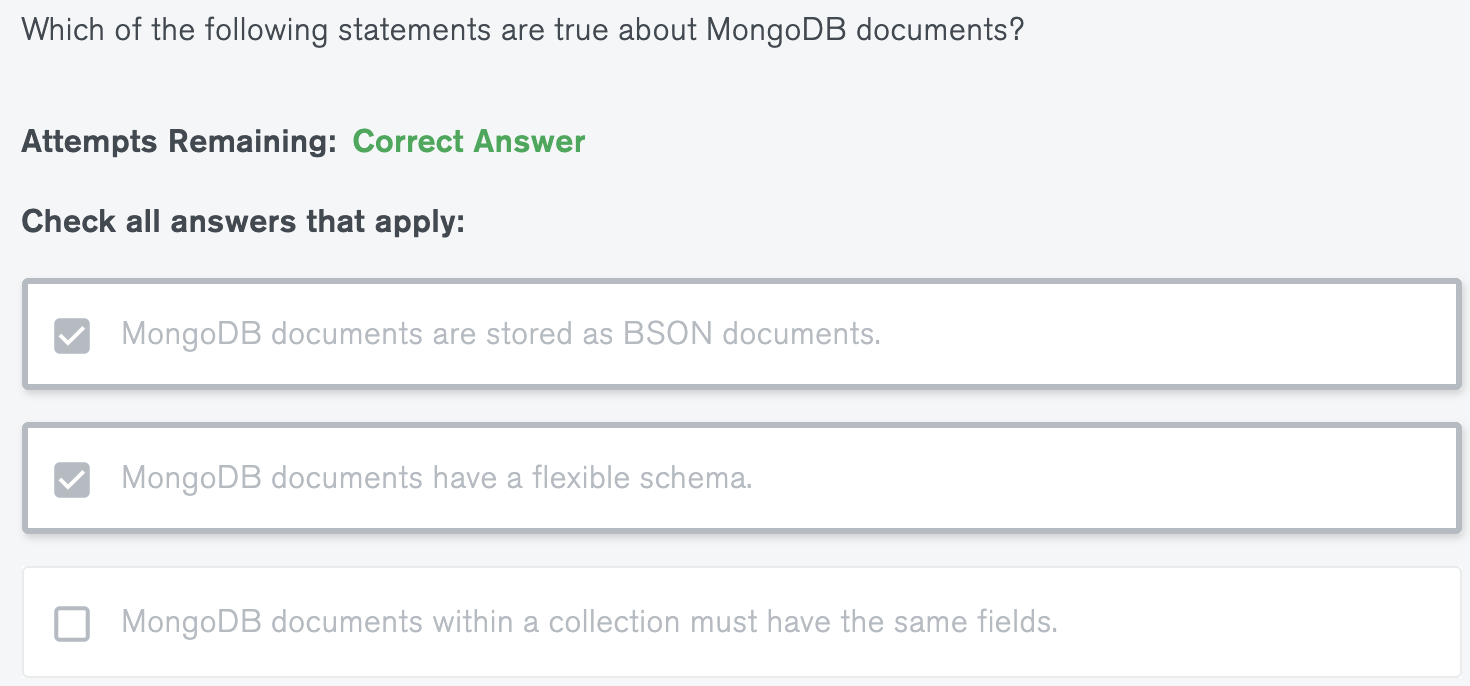
Check all answers that apply:

A>MongoDB will help you iterate on the schema designs of your models throughout your application's lifecycle.

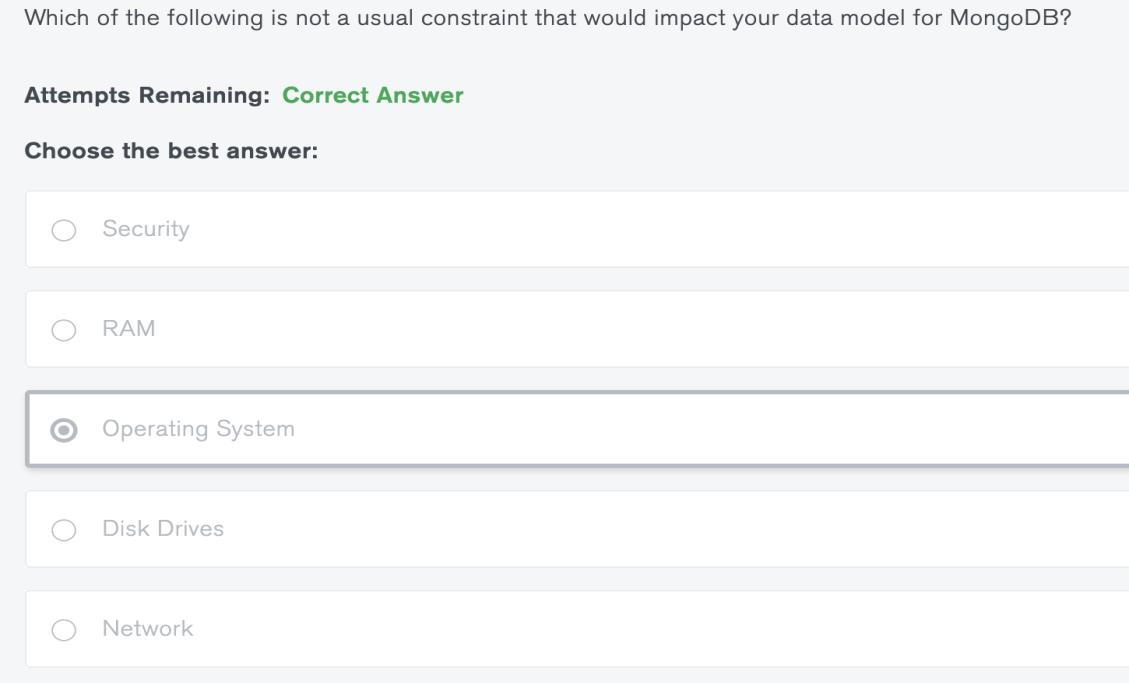
B>MongoDB is schema-less so you should not worry about designing a schema for your models.

C>MongoDB should only be used for unstructured datasets.

2.



3.



4.

Problem:

1>Which of the following phases are included in our data modeling methodology for MongoDB?

Attempts Remaining:∞Unlimited Attempts

Check all answers that apply:

A>Applying schema design patterns.

B>Identifying the relationships between pieces of information.

C>Identifying the workload of the system.

2>Which of the following are use cases in which you should model your data for performance rather than simplicity?

Attempts Remaining:Correct Answer

Check all answers that apply:

A>It is expected that the solution will be designed with only 10 shards.

B>The application is being developed by 100 engineers.

C>There is not an applicable design pattern to the solution.

3>Which of the following is not part of the first phase of the data modeling methodology?

Attempts Remaining:∞Unlimited Attempts

Choose the best answer:

A>Listing the read operations.

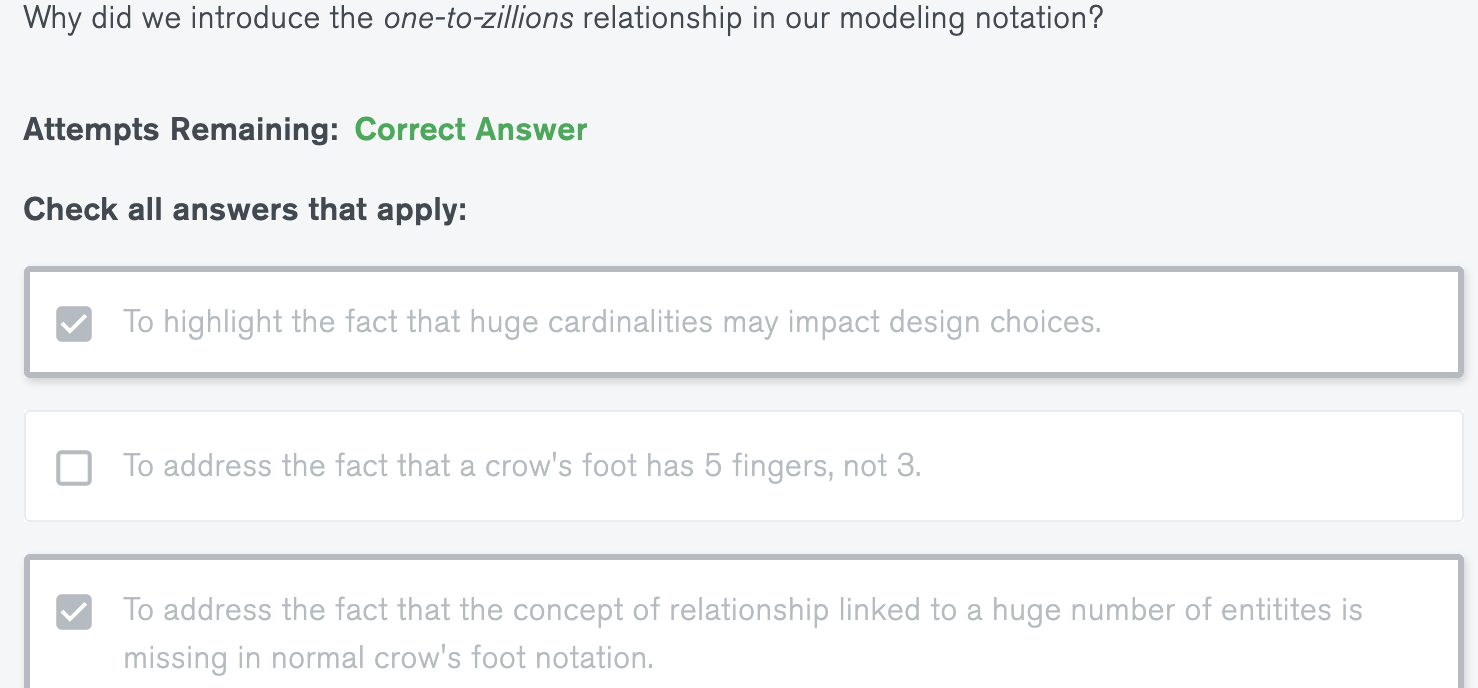
B>Identifying the durability of each write operation.

C>Identifying the relationships between the units of data.

D>Quantifying each of the operations in terms of latency and frequency.

E>Listing the write operations.

4.



5.

Consider a one-to-many relationship observed between a county and the cities in that county.

Which of the following are valid ways to represent this one-to-many relationship with the document model in MongoDB?

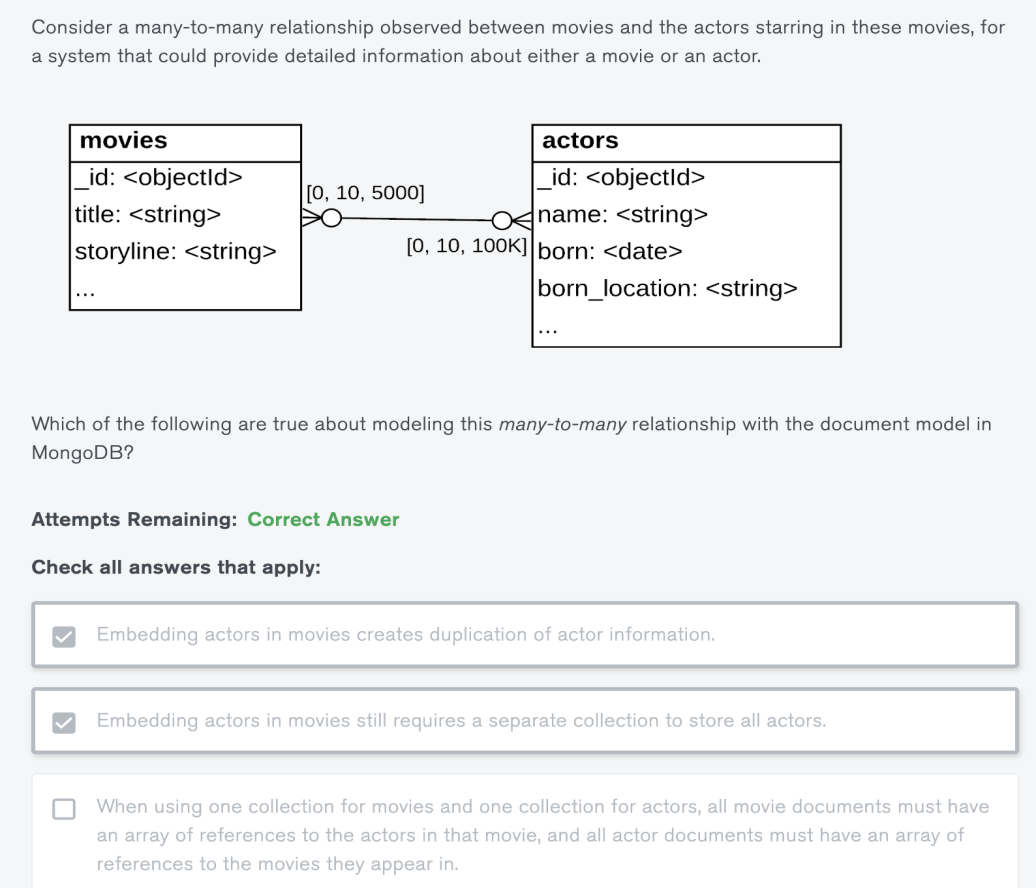
Check all answers that apply:

A>Embed the entities for the cities as an array of sub-documents in the corresponding county document.

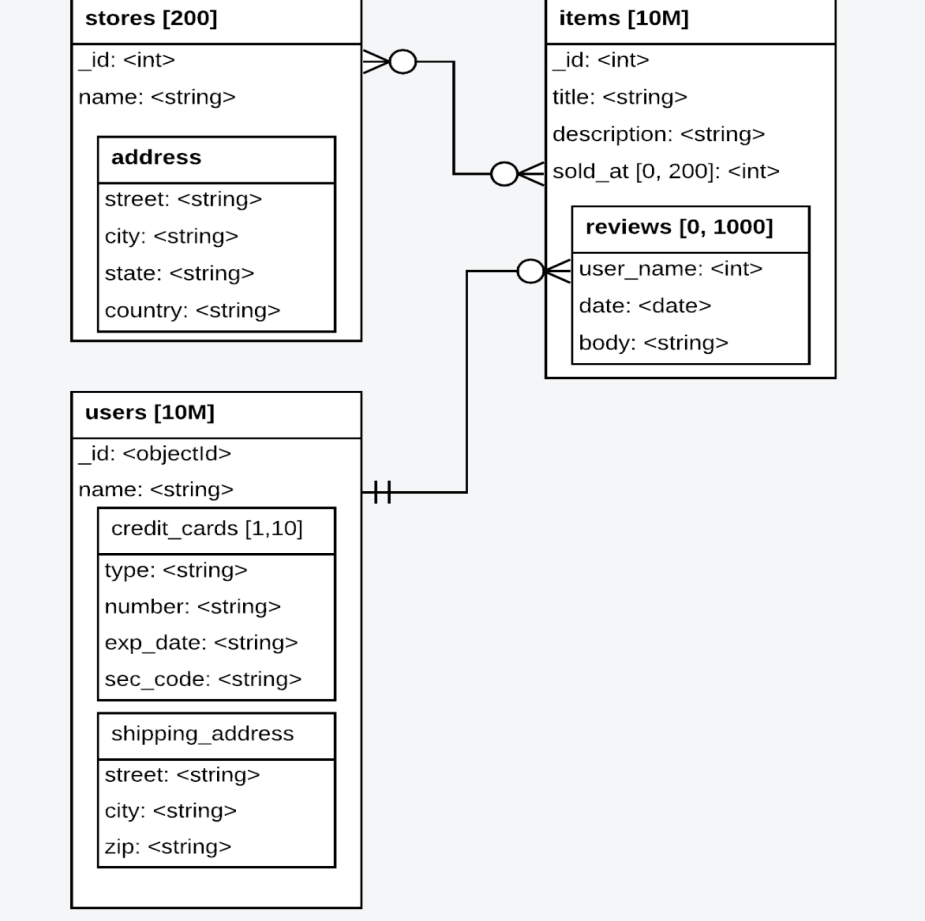
B>Have a collection for the counties and a collection for the cities with each city document having a field to reference the document of its county.

C>Embed all the fields for a city as a subdocument in the corresponding county document.（可能超出16mb限制）

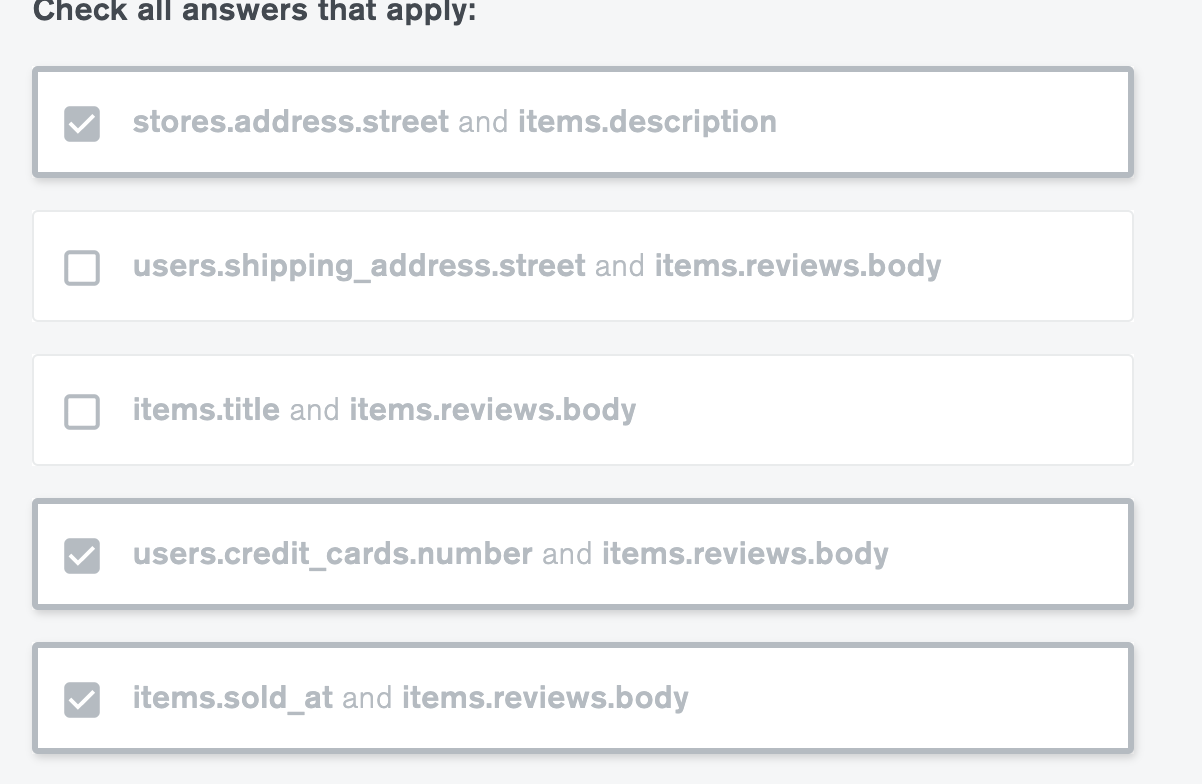
6.



识别多对多关系：



answer



Problem:

Which of the following are valid ways to represent a one-to-one relationship with the document model in MongoDB?

Check all answers that apply:

A>Link to a single document in another collection.

B>Embed the fields in the document.

B>Embed the fields as a sub-document in the document.

Problem:

Which of the following statements are true about one-to-zillions relationships?

Check all answers that apply:

A>The relationship representations that embed documents are not recommended.

B>We must take extra care when writing queries that retrieve data on the zillions side.

C>It is a special case of the one-to-many relationship.

Which of the following are valid concerns regarding duplication, staleness and referential integrity management in a MongoDB database and appropriate resolution techniques?

A>Data integrity issues can be minimized by using multi-document transactions.

B>Data duplication should not exist and can be avoided with multi-document transactions.

C>Data staleness issues can be minimized with frequent batch updates.

2Which one of the following scenarios is best suited for the application of the Attribute Pattern?

Choose the best answer:

A>The documents need strict validation.

B>The system is accessing the disk too frequently.

C>Some fields share a number of characteristics, and we want to search across those fields.

D>The working set does not fit in memory.

E>The documents are large.

Which one of the following scenarios is the best candidate to use the Extended Reference Pattern to avoid doing additional reads through joins/lookups?

A>A product model needs to store references to images of the product that are kept in an external location outside the database.

B>An order model needs to store the product ID, the price sold, and the quantity ordered for each product in an order.

C>A product model needs to store a counter representing the number of times it was purchased.

D>An app needs to retrieve a product and information about its supplier.

E>An app needs to retrieve a product and its ten most recent reviews.

Which one of the following scenarios is the best candidate for use of the Subset Pattern?

Attempts Remaining:Correct Answer

Choose the best answer

The developers of the system have left and no one understands the application.

The working set does not fit in memory and it is difficult to scale the hardware.

The system is running out of RAM.

The system is accessing the disk too frequently

The documents are too big.

Which one of the following scenarios is best suited for the application of the Computed Pattern?

We need to calculate a value that is displayed once per minute and is based on a field which updates 100 times per minute.

We have too much information to store in a single document.

We need to calculate a value that is displayed 100 times a minute and is based on a field which updates 100 times per minute.

We need to group documents and sum on a field.

We need to calculate a value that is displayed 100 times a minute and is based on a field which updates once a minute.

Which one of the following requirements in our system is the best candidate to use the Bucket Pattern?

Choose the best answer:

Our system ingests thousands of log lines each day for each host it monitors.

Our system handles 1 million IOT devices.

Our system performs sums and averages over all elements of certain arrays.

Our system must embed a one-to-many relationship in one of our models, however, some of the result documents would be too big.

Our system ingests 10 million pieces of data per day from 1 million devices, with 20% coming from 10 devices.

Which one of the following scenarios is the best candidate for the use of the Schema Versioning Pattern?

Attempts Remaining:Correct Answer

I want to avoid downtime when upgrading my schema.

I can schedule a window of downtime long enough to migrate the documents to the new version.

I want to keep track of the changes to my documents.

I have billions of documents.

I have many obsolete documents.

Which of the following scenarios would be ideal to use the Tree Pattern?

Attempts Remaining:Correct Answer

Contact lists of users

Company organization charts

Product categories

Which one of the following scenarios is best suited for the use of the Polymorphic Pattern?

Attempts Remaining:Correct Answer

The organization acquired different companies over the years, serving the same markets with the same customers and there is a requirement to merge all systems into one.

There is a requirement to keep many versions of a document, and these versions may have different fields for each version.

The application has a base class with some derived classes.

There are billions of documents.

There is a requirement to store addresses for my customers.

MODEL

Consider a system that collects census data on all countries in the world.

Which of the following implementations are recommended ways to model a One-to-Zillions relationship between a person entity and the country entity in which that person was born?

Check all answers that apply:

A>>Referencing from the One side.

In the document for a country, we have an array of references to the documents of the people who are kept in a separate collection.

B>>Embedding as an array.

In the document for a country, we embed all the people born in that country.

C>>Referencing from the Zillions side.

In each person document, we reference the corresponding country document.

Your team just hired a new Data Architect, and her amazing ideas are gaining traction with the company leadership.

You already updated your application to be able to handle the new data organization in your database. Now you have been tasked with implementing her proposed new data organization approach to your database with minimum downtime for the users of the application.

Which pattern solution is best suited for this situation?

The Bucket Pattern

The Attribute Pattern

The Extended Reference Pattern

The Subset Pattern

The Schema Versioning Pattern

A new decision maker came aboard your online bookstore team. They want to be able to track which genres are most popular daily. To keep this metric up to date without running massive queries for obtaining it, which of the following schema patterns would you choose to implement?

The Bucket Pattern

The Polymorphic Pattern

The Subset Pattern

The Extended Reference Pattern

The Computed Pattern

You work as a developer at a factory. Your factory wants to track the usage statistics of the automatic lighting that was recently installed throughout its facilities. The lights send an update to the database every 10 seconds, but the management is interested in an hourly report instead. Additionally, we are only looking to store this information for at most 5 years, so an easy way to purge old data would be beneficial to our data modeling approach.

Which pattern solution is best suited for this situation?

Attempts Remaining:Incorrect Answer

The Computed Pattern

The Bucket Pattern

The Subset Pattern

The Polymorphic Pattern

The Extended Reference Pattern

With the digitization of every area of our lives, the famous NYC bodegas (convenience stores) are trying to keep up. Bodegas don't just know everything that goes on in the neighborhood, they also supply all types of household goods, hardware supplies, and groceries. The New York City Bodega Association is looking to create an app that will help them keep track of their unique, versatile inventory and help customers look up whether items are in stock before visiting the bodega.

The Polymorphic Pattern

The Subset Pattern

The Extended Reference Pattern

The Computed Pattern

The Bucket Pattern

As a chemical manufacturer, you tend to keep your factory and your data organized, since dealing with chemicals requires a lot of precision, attention, and safety mechanisms. One of the safety mechanisms in the factory is the documentation about produced chemicals. This documentation is recorded in Material Safety Data Sheets which are large pdf documents containing safety details about a given chemical. These Data Sheets are part of the documents in the inventory collection, where other information such as the price, quantity, and warehouse location of the chemical is stored as well.

Keeping track of production, sales, and purchases requires a lot of data manipulation on an hourly basis. You notice that at especially busy times, your inventory tracking application slows down by a lot.

Which pattern solution is best suited for solving this issue?

The Subset Pattern

The Bucket Pattern

The Computed Pattern

The Polymorphic Pattern

The Extended Reference Pattern

