Boost up Your Certification Score

MongoDB
C100DBA
MongoDB Certified DBA Associate Exam (Based on MongoDB 4.4)

Exams Boost
Boost Up Your Career

For More Information – Visit link below:

https://www.examsboost.com/

Product Version

✓ Up to Date products, reliable and verified.
✓ Questions and Answers in PDF Format.

Visit us at: https://www.examsboost.com/test/c100dba

Latest Version: 9.0

Question: 1

Given the following MongoDB documents in the cars collection:

```
{
"carid": 1,
"make": "Toyota",
"model": "Celica",
"mileage": 143121,
"color": "Red",
"year": 2003,
"features": ["Leather Interior", "2WD", "CD Player"]
},
{
"carid": 2,
"make": "Honda",
"model": "Civic",
"mileage": 12000,
"color": "White",
"year": 2020,
"features": ["Cloth Interior", "Sun Roof", "XM Radio"]
},
{
"carid": 3,
"make": "Tesla",
"model": "Model S",
"mileage": 5,
"color": "Silver",
"year": 2022,
"features": ["Leather Interior", "Sun Roof", "Auto Pilot"]
}
```

What documents are returned from running the following $in query?

```
db.cars.find({"features": { $in: ["Leather Interior","2WD","Auto Pilot"]}})
```

A. [{ "carid": 1, "make": "Toyota", "model": "Celica", "mileage": 143121,"color": "Red", "year": 2003,"features": ["Leather Interior","2WD","CD Player"] },{ "carid": 3, "make": "Tesla", "model": "Model S", "mileage": 5,"color": "Silver", "year": 2022,"features": ["Leather Interior","Sun Roof","Auto Pilot"] }]

B. Documents with carid 1 and 2.

C. Documents with carid 1, 2, and 3.

D. Only the document with carid 3.

Answer: A

Explanation:
The $in query predicate matches a document when the queried field equals any value listed in the $in array. When the target field is itself an array, MongoDB compares the array elements against the values supplied to $in; one matching element is sufficient. Car 1 qualifies because its features array contains both "Leather Interior" and "2WD". Car 3 also qualifies because its array contains "Leather Interior" and "Auto Pilot". Car 2 contains "Cloth Interior", "Sun Roof", and "XM Radio", none of which are in the predicate array, so it is excluded. A common mistake is to read $in as requiring all values to be present. It does not: the semantics are an OR-style membership test. If the requirement were to ensure that several specified values were all present in an array, a different construct such as $all would be appropriate. Therefore, the result consists of the Toyota Celica and Tesla Model S documents only, making option A correct.
Study Guide reference/topic:MongoDB Architecture and Core Operations - querying documents, arrays, and the $in predicate.
==========

Question: 2

Which TWO commands can be used to check for all active operations on the current node?

A. db.currentOp({ op : "command" })
B. db.adminCommand({ "all" : currentOp})
C. db.currentOp(true)
D. db.adminCommand({ currentOp : true })
E. db.currentOp({ "all" : true })

Answer: C, D

Explanation:
db.currentOp() is the mongosh helper used to inspect operations currently executing on a MongoDB server. Passing true requests a broad view that includes all in-progress operations as well as additional idle/system entries, so it can be used when the administrator wants to see the complete current-operation picture. The database-command form, db.adminCommand({ currentOp: true }), invokes the corresponding administrative command and is also valid for obtaining in-progress operation information. Option A adds a filter restricting the result to operations whose op field is "command"; it therefore does not represent all operation types. Option B is not valid currentOp command syntax, and option E uses an all field that is not the documented special form for this helper. In modern MongoDB administration, the $currentOp aggregation stage is generally preferred for richer filtering and aggregation, but the question is testing the traditional helper and command interfaces. Therefore, C and D are the two valid choices among those presented.
Study Guide reference/topic:Performance Monitoring and Tuning - current operations, currentOp, and administrative diagnostics.

Visit us at: https://www.examsboost.com/test/c100dba

==========

Question: 3

Which TWO indicators should an administrator keep an eye on to determine if the storage capacity is nearing its limit?

A. The execution time of the same query increases over time.
B. Disk space utilization on the server is consistently above 90%.
C. The number of documents in the database is increasing.
D. Message errno:28 No space left on device is present in the log file.
E. The number of available read or write tickets is consistently low.

Answer: B, D

Explanation:
The most direct evidence that storage capacity is approaching exhaustion is high filesystem utilization and explicit operating-system errors showing that no space remains. Sustained disk utilization above 90% is a practical warning threshold because the deployment has little headroom for database growth, index builds, temporary files, journal activity, or operational spikes. The errno:28 No space left on device message is even stronger evidence: the operating system rejected a write because the relevant filesystem could not allocate additional space. The other choices can correlate with workload growth but do not specifically prove a capacity problem. Query execution time can increase because of poor indexes, cache misses, lock or ticket contention, or CPU pressure. A rising document count says nothing about remaining free space without knowing document size, index size, and volume capacity. Low read/write ticket availability reflects storage-engine concurrency pressure rather than filesystem capacity. Administrators should therefore monitor disk-used and disk-free metrics, alert well before full utilization, and investigate any ENOSPC-style log events immediately. B and D are the two storage-capacity indicators.
Study Guide reference/topic:Performance Monitoring and Tuning - disk capacity, storage metrics, alerts, and diagnostic logs.
==========

Question: 4

Consider a clothing collection that contains these documents:
{ "_id": 1, "item": "sweater", "sizes": ["XS", "S"] }
{ "_id": 2, "item": "t-shirt", "sizes": ["S", "M", "XL"] }
{ "_id": 3, "item": "vest", "sizes": ["M", "L", "XL", "XXL"] }
What does the following query return?
db.clothing.find({ "sizes": "M" })

A. Only document _id: 3.
B. All three documents.
C. Only document _id: 2.
D. Documents _id: 2 and _id: 3.

Answer: D

Explanation:
MongoDB supports direct equality matching against elements contained in an array. When the predicate is { sizes: "M" } and sizes is an array field, a document matches when at least one element of the array equals the scalar value "M". Document 1 contains only "XS" and "S", so it does not match. Document 2 contains "M" in ['S','M','XL'], and document 3 contains "M" in ['M','L','XL','XXL']; both therefore match. No $elemMatch expression is required for this simple scalar membership test. $elemMatch becomes important when multiple conditions must be satisfied by the same array element, particularly for arrays of embedded documents or compound conditions. This array equality behavior is a core part of MongoDB's document query model and allows concise filtering of array-valued fields. The correct result is consequently the t-shirt and vest documents, which corresponds to option D. This same matching behavior applies consistently whenever a scalar predicate targets values stored inside a BSON array field.
Study Guide reference/topic:MongoDB Architecture and Core Operations - querying array fields and equality predicates.
==========

Question: 5

Given the following query:
db.load.find({
_id: { $lte: 1000 },
rnd: 1234,
date: { $gte: ISODate("2023-01-01") }
}).sort({ date: -1 })
In a compound index that best supports this query, what field should be first field?

A. no field
B. rnd: 1
C. _id: 1
D. date: 1

Answer: B

Explanation:
The rnd field should lead the compound index because it is the query's equality predicate. MongoDB's common compound-index design rule is Equality, Sort, Range (ESR): fields tested for

exact equality are typically placed first, followed by fields used for sorting and then range predicates, subject to workload-specific selectivity considerations. Here, rnd: 1234 is an equality condition. _id: { $lte: 1000 } is a range condition, and date: { $gte: ... } is also a range condition while the query additionally sorts on date. Leading with rnd sharply narrows the index region that must be scanned before MongoDB handles the sort/range dimensions. Starting with _id or date would place a range-oriented key ahead of the exact-match predicate and generally reduce the efficiency of index bounds for this query shape. The complete optimal key pattern may depend on whether avoiding the sort or maximizing range selectivity matters most, but the question asks only for the first field. Under ESR guidance, that first field is rnd, so option B is correct.
Study Guide reference/topic:Indexing and Query Optimization - compound indexes, selectivity, and the Equality-Sort-Range guideline.
==========

Visit us at: https://www.examsboost.com/test/c100dba

Thank You for Trying Our Product

For More Information – Visit link below:

https://www.examsboost.com/

15 USD Discount Coupon Code:

G74JA8UF

FEATURES

- ✓ 90 Days Free Updates
- ✓ Money Back Pass Guarantee
- ✓ Instant Download or Email Attachment
- ✓ 24/7 Live Chat Support
- ✓ PDF file could be used at any Platform
- ✓ 50,000 Happy Customer

