

PRACTICE QUESTIONS

Microsoft

AI-500

Multi-Agent AI Solutions Expert

SHARED BY [u/Warm-Walrus-6045](#) on aiexamcerts

EDITION Free Set 2

QUESTIONS 10

SHARED October 18, 2026

EXAM Microsoft AI-500, Multi-Agent AI Solutions Expert

Shared by [u/Warm-Walrus-6045](#) on aiexamcerts

10 practice questions for the Microsoft AI-500 exam, Multi-Agent AI Solutions Expert, each followed by its answer and a short explanation. More free sets and the full practice exam: aiexamcerts.com/microsoft/ai-500. Spotted a mistake? Email support@aiexamcerts.com with the exam code AI-500 and the question number.

1. Which caching approach returns a stored answer for a new question that has the same meaning as an earlier one, even when the wording differs?
 - A Response caching keyed on the exact request bytes
 - B Prompt caching of a static prompt prefix
 - C Browser caching of image files
 - D Semantic caching

Answer: D

Semantic caching compares the meaning of an incoming question with cached questions using embeddings and reuses the answer when they are close enough. Response caching only matches identical requests, and prompt caching reuses the processed prefix of a prompt rather than whole answers. Browser caching of images is unrelated to model calls.

2. Which technique adds only the most relevant retrieved passages to the prompt at request time?
 - A Dynamic context injection
 - B Fine-tuning on the full document set
 - C Hard-coding every fact into the system prompt
 - D Raising the temperature for more creative answers

Answer: A

Dynamic context injection selects relevant content for each request and places it in the prompt, so the model sees current, targeted facts. Fine-tuning changes model weights rather than selecting context per request. Hard-coding every fact bloats the prompt, and temperature affects randomness, not what context is supplied.

3. Fine-tuning a model is usually the most efficient way to give an agent access to current company policy, because fine-tuning updates the model's knowledge with every new document.
 - A True
 - B False

Answer: B (False)

Fine-tuning is costly to repeat and fixes behavior and style more than fresh facts, so policies that change often are better supplied through retrieval. The statement is false.

4. Users ask about a policy document that was revised last week. Which approach gives the agent the most current answer?
- A Lowering the maximum output tokens
 - B Retrieval-augmented generation over an index that is refreshed when the documents change
 - C Freezing the vector index so that it cannot change
 - D Disabling retrieval so the model answers from memory

Answer: B

Retrieval-augmented generation pulls the current text from an index that is refreshed when sources change, so the answer reflects last week's revision. A model that answers from memory relies on training data that predates the revision. Lowering output limits and freezing the index do not bring in new content.

5. Retrieved answers often omit the figure a question needs, because chunks were split in the middle of a table. What is the underlying problem?
- A Token rate limiting on the model endpoint
 - B A slow TLS handshake to the search service
 - C Poor chunk boundaries that separate related content and lose its context
 - D A missing RBAC role on the storage account

Answer: C

Splitting a table or sentence across chunks separates the value from its heading and neighbors, so retrieval returns fragments that cannot answer the question. Rate limits, TLS latency and RBAC roles can cause errors or slowness, but they do not explain fragments that lack the needed content.

6. An MCP client must call a tool implemented as an Azure Function. Which design is correct?
- A Give the client the storage account key of the function app so that it can reach the files the function uses directly
 - B Let the client call the function's internal Kudu console
 - C Compile the function into the client binary
 - D Expose the function through an MCP server endpoint that the client connects to

Answer: D

The function is exposed through an MCP server, so the client discovers and calls the tool through the standard protocol while the server enforces authentication. Sharing the storage key or the Kudu console grants far more access than a tool call needs, and compiling the function into the client couples the two and bypasses the server.

7. A tool returns malformed JSON to the agent. What should the agent do first?
- A Validate the result against the expected schema, then retry or use a fallback
 - B Forward the malformed output to the user as the final answer, since the user can judge whether it is usable
 - C Disable all tools for the rest of the session
 - D Increase the context window so the output fits

Answer: A

Validating the result against the schema catches the malformed output before it influences a decision, and the agent can then retry or switch to a fallback path. Passing it to the user spreads the error, disabling all tools removes needed capability, and a larger context window does not repair a malformed response.

8. Tool result validation should check outputs against an expected schema and value ranges before the agent uses them to make a decision.
- A True
 - B False

Answer: A (True)

Checking the schema and value ranges before use keeps bad data from driving decisions or further tool calls. This is one of the quality checks the solution should build in. The statement is true.

9. Which mechanism lets a multi-agent solution reuse logging, authorization and exception handling across all of its agents?
- A A shared spreadsheet of error messages
 - B Middleware that wraps each agent and tool call
 - C A nightly cron job that copies log files from every agent host into a shared folder for later review
 - D A DNS record for each agent name

Answer: B

Middleware wraps agent and tool invocations in one place, so logging, authorization and error handling are applied consistently without repeating code in each agent. A spreadsheet, a nightly copy job and DNS records do not run around each call.

10. A tool call times out during a workflow. Which fallback design is the most sound?
- A Repeat the same call in a tight loop with no delay
 - B Invent a plausible result and continue the workflow
 - C Retry with backoff, then switch to an alternate tool or return a clearly labeled partial answer
 - D Stop the entire system until someone restarts it manually

Answer: C

Retrying with backoff handles transient timeouts, and a labeled fallback keeps the workflow useful when the tool stays unavailable. A tight retry loop adds load, and inventing results corrupts the answer. Stopping the whole system is an outage, not a fallback design.