DUMPSARENA

Cisco AppDynamics Associate Performance Analyst(CAAPA)

Cisco 500-420

Version Demo

Total Demo Questions: 14

Total Premium Questions: 141

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, AppDynamics Fundamentals	16
Topic 2, Application Monitoring	58
Topic 3, Application Performance Troubleshooting	24
Topic 4, Server Monitoring	5
Topic 5, Database Monitoring	5
Topic 6, End User Monitoring	5
Topic 7, Analytics	6
Topic 8, Alerting and Reporting	22
Total	141

QUESTION NO: 1

What does the Live Preview option in a Business Transaction Discovery Session allow a user to do?

A. See if new Transaction Discovery Rules work

B. Provide streaming logs from the application agent

C. Search classes and methods currently running on the system

D. Stream entry points in your application

ANSWER: A

Explanation:

See if new Transaction Discovery Rules work is correct because Live Preview provides an immediate way to validate transaction discovery-rule configuration before relying on it for ongoing business-transaction monitoring. In a Business Transaction Discovery Session, AppDynamics observes transaction entry points detected by the application agent and presents preview results for the configured rule criteria. This lets an administrator confirm that the rule identifies the intended transactions and produces the expected business-transaction naming and categorization behavior.

This feedback is particularly useful when refining rule scope, matching conditions, naming conventions, or priority. Rather than waiting for a configuration change to affect the monitored application and then checking results later, the user can use the discovery session to assess the rule against current application activity. The preview supports iterative tuning so that the resulting business transactions are meaningful, appropriately grouped, and suitable for monitoring performance and troubleshooting. AppDynamics documents Business Transaction Discovery Sessions as a workflow for discovering transaction entry points and creating transaction detection rules; Live Preview is the validation capability within that workflow. See the AppDynamics documentation on Business Transaction Discovery Sessions and the transaction detection rules documentation.

QUESTION NO: 2

Business users of an e-commerce site want to see the running aggregate checkout value stored in the Java class Checkout. How would a Performance Analyst capture this data?

A. Create a dashboard widget to total Business Transactions

B. Create a method invocation data collector

C. Create an HTTP data collector

D. Export the data hourly with a REST call

ANSWER: B

Explanation:

Create a method invocation data collector. This is the appropriate AppDynamics mechanism when the desired business value resides in a Java application object, such as an instance of the `Checkout` class, rather than being directly available in an HTTP request. The analyst configures the collector for the relevant class and method that executes during the checkout flow, then specifies how AppDynamics should obtain the value. Depending on the application design, the collector can capture a method parameter, return value, object field, or a value reached through a getter chain. When that instrumented method runs as part of a monitored Business Transaction, AppDynamics records the checkout amount as business data associated with the transaction. That captured business data is then available for analysis and can be aggregated over time to provide the running or total checkout value requested by business users. Method invocation data collectors are specifically intended to extract application-level data from Java method execution, making them suitable for values held within application classes. See the AppDynamics documentation on collecting business data and data collectors.

QUESTION NO: 3

What are two default Business Transaction limits at the Application and Node levels? (Choose two.)

A. Application 100

B. Application 200

C. Application 300

D. Node 50

E. Node 75

F. Node 200

ANSWER: B D

Explanation:

The default Business Transaction registration limits are **Application 200** and **Node 50**. AppDynamics uses the application-level ceiling to limit the total number of automatically discovered and registered Business Transactions for an application. This protects the Controller and monitored application environment from uncontrolled growth in Business Transaction definitions and the associated metrics. The node-level limit restricts the number of Business Transactions that an individual application agent node can register through automatic discovery, helping prevent a single node with highly variable request patterns from consuming an excessive portion of the available registration capacity.

These defaults are important when analyzing why expected request patterns may not appear as separately named Business Transactions. After registration capacity is reached, additional discovered traffic is generally handled as unregistered or aggregated traffic unless administrators refine discovery behavior. Teams can preserve visibility for important flows by defining custom match rules, excluding unnecessary traffic, improving naming rules, or using Business Transaction lock down after the desired transactions are registered. Cisco AppDynamics documents Business Transaction discovery, registration, and limits in its Business Transaction Registration documentation and provides configuration guidance in the Business Transactions documentation.

QUESTION NO: 4

Business Transaction Discovery Sessions work with which two code languages? (Choose two.)

A. .NET

B. C++

C. PHP

D. Java

E. Node.js

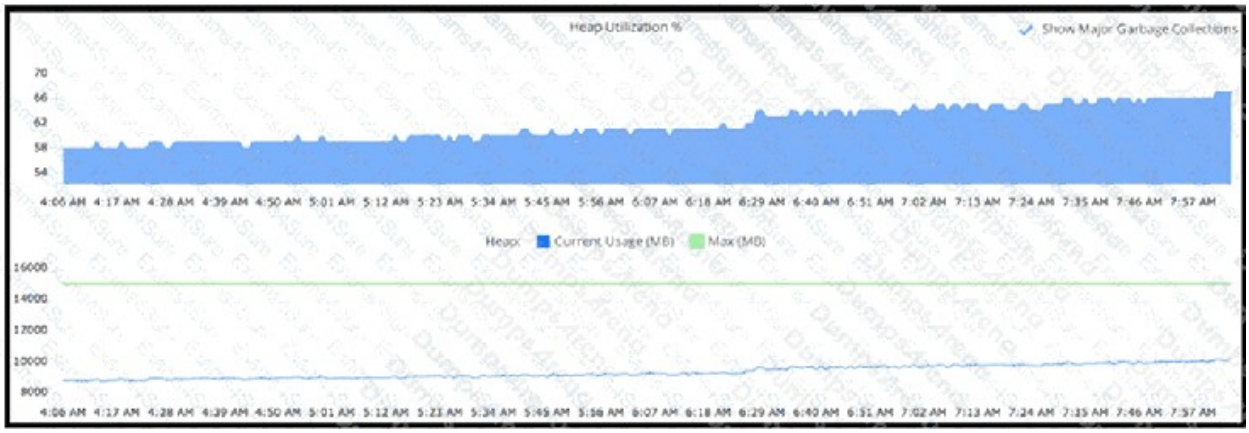
ANSWER: A D

Explanation:

Business Transaction Discovery Sessions are available for applications monitored by the Java Agent and the .NET Agent, so **.NET** and **Java** are the two correct selections. This feature supports the identification of business-transaction entry points when the standard automatic detection mechanisms do not identify the application's relevant execution path. An analyst starts a discovery session on a tier, exercises the relevant workload, and reviews the classes and methods observed by the agent. That runtime information can be used to define a custom match rule and establish an appropriately named Business Transaction. Because this workflow requires agent-level visibility into executed classes and methods, it is specifically implemented for the Java and .NET application-agent environments. Cisco AppDynamics documentation describes discovery sessions as a method for discovering transaction entry points and creating custom detection rules, while the Java and .NET agent documentation identifies those agents as the instrumentation components used to monitor their respective application runtimes. See Business Transaction Discovery Sessions and App Server Agents.

QUESTION NO: 5

Refer to the exhibit.



Using this heap-utilization graph, which method is used to confirm whether a memory leak is occurring during a specific timeframe?

A. In Metric Browser, navigate through Application Infrastructure > Hardware Resources and select Memory Total (MB) and Used (MB)

B. In Tiers and Nodes, open the JMX tab and select JVM > Memory > Heap > Max Available (MB) and Current Usage (MB)

C. In Metric Browser, navigate through Application Infrastructure > Hardware Resources and select Memory Total (MB) and Swap Used (MB)

D. In Tiers and Nodes, open the Memory tab and visualize Heap Utilization (%) and Heap Current Usage (MB) versus Max (MB)

ANSWER: D

Explanation:

In Tiers and Nodes, open the Memory tab and visualize Heap Utilization (%) and Heap Current Usage (MB) versus Max (MB) is the appropriate method because it provides the JVM heap-specific measurements needed to investigate the selected period directly. Heap utilization expresses current heap consumption as a proportion of the configured maximum heap, while Heap Current Usage and Max show the absolute amount of memory consumed and the JVM's available heap ceiling. Viewing these metrics together over the timeframe represented in the graph lets an analyst determine whether heap usage continually trends upward and fails to return to a stable baseline after garbage-collection activity. That sustained growth pattern, particularly when the maximum remains unchanged, is the key evidence used to confirm a likely memory leak. The node-level Memory tab is designed for this JVM-focused operational analysis and supports examining the metric behavior in the relevant time window rather than relying on host-wide memory values. AppDynamics describes its application monitoring capabilities, including JVM and infrastructure visibility, in its documentation portal. Additional product information about application performance monitoring and diagnostic visibility is available from Cisco AppDynamics.

QUESTION NO: 6

Which two Key Performance Indicators (KPIs) accurately provide insight into server-level resource consumption? (Choose two.)

A. Calls per Minute

B. Availability

C. Memory Used%

D. Average Response Time

E. Application Restarts

F. CPU %Busy

ANSWER: C F

Explanation:

Memory Used% and **CPU %Busy** are the appropriate KPIs because they directly quantify consumption of fundamental host resources on the server monitored by AppDynamics. Memory Used% indicates the proportion of installed physical memory currently in use. Tracking it over time helps an analyst recognize sustained memory demand, assess whether host capacity is adequate, and correlate memory pressure with application slowdowns or instability. CPU %Busy measures the share of processor time spent executing work rather than remaining idle. It is therefore a direct measure of compute utilization and is especially useful when correlating high processing demand with slower transaction execution, queueing, or limited server capacity.

AppDynamics Infrastructure Visibility gathers machine-level metrics through the Machine Agent and presents them in the metric hierarchy so analysts can relate infrastructure health to application behavior. CPU and memory metrics are core machine-resource indicators: they describe the utilization of the server itself, rather than merely reporting application workload or a service outcome. Reviewing both KPIs together provides a practical view of whether a host is constrained by compute, memory, or both, and supports capacity planning and root-cause investigation. See the AppDynamics Machine Agent metrics documentation and Infrastructure Visibility documentation.

QUESTION NO: 7

Which two methods are used to plot Host CPU and GC Time Spent in a single view? (Choose two.)

A. Server tab under " Tiers and Nodes "

B. JMX tab under " Tiers and Nodes "

C. Memory tab under Tier and Nodes "

D. Metrics Browser

ANSWER: B D

Explanation:

The **JMX tab under "Tiers and Nodes"** can be used to graph both measurements because the Java agent exposes JVM-management data through JMX, including garbage-collection collection-time values and operating-system CPU utilization values available from the JVM's platform MBeans. Selecting the applicable MBean attributes and plotting them together makes it possible to correlate host CPU activity with time spent in garbage collection for the selected node. Cisco's JMX metrics documentation describes how AppDynamics discovers, displays, and collects metrics exposed through JMX: AppDynamics JMX Metrics.

The **Metrics Browser** also supports this use case because it provides access to the application, JVM, and machine metric hierarchy and allows selected metrics to be plotted on the same graph. An analyst can choose the Host CPU metric and the JVM garbage-collection time-spent metric for the relevant node, then view their trends together over the same time range. This is particularly useful when investigating whether elevated CPU utilization coincides with increased garbage-collection overhead. Cisco documents metric selection and graphing workflows in the Metric Browser documentation.

QUESTION NO: 8

Which tool should be used to analyze all the External Call details, including database and remote service calls, for a specific Business Transaction?

A. Business Transaction List

B. Application Dashboard

C. Transaction Snapshot

D. Metric Browser

ANSWER: C

Explanation:

Transaction Snapshot is the appropriate tool because it records detailed diagnostic data for an individual execution of a Business Transaction. This execution-level context lets a performance analyst trace the transaction's call flow and assess the time spent in downstream dependencies, rather than relying only on aggregated metrics. A snapshot includes a call graph and identifies exit calls made during that transaction instance, providing the context needed to investigate database activity and remote-service activity associated with the request.

The snapshot's DB and Remote Services Calls information is specifically intended to help identify external dependencies involved in the transaction. It provides details such as the backend or remote endpoint, call counts, execution time, and the contribution of calls to the overall transaction time. Where the agent captures database details, the analyst can drill into database-call information to correlate SQL activity and database time with the observed transaction behavior. This makes Transaction Snapshot particularly useful when diagnosing latency or failures in one Business Transaction execution and determining which external dependency contributed to it.

For Cisco AppDynamics documentation on snapshot diagnostics and call graphs, see Transaction Snapshots and Transaction Snapshot Details.

QUESTION NO: 9

In which two features of AppDynamics can Information Points metric data be used? (Choose two.)

A. Alerting

B. Analytics

C. Flow Maps

D. Custom Dashboards

ANSWER: A D

Explanation:

Information Points create custom application metrics by collecting data at configured points in an application, such as a method invocation, a method argument, or a return value. Once collected, this data becomes available as metric data in the Controller, allowing it to participate in the platform's standard monitoring and visualization workflows. **Alerting** is a valid use because Information Point metrics can be evaluated through health-rule conditions. Health-rule violations can then trigger policies and notification actions, enabling teams to receive alerts when an Information Point metric crosses an operational threshold. **Custom Dashboards** is also a valid use because dashboard metric widgets can display Information Point metric values and trends alongside application, business-transaction, and infrastructure metrics. This lets an analyst build focused operational views for custom technical or business measurements. Cisco AppDynamics documents Information Points as a mechanism for collecting custom metric data, while its health-rule and dashboard capabilities consume Controller metric data for monitoring, alerting, and visualization. See the AppDynamics Information Points documentation and the AppDynamics Health Rules documentation.

QUESTION NO: 10

A Performance Analyst must send a critical health-rule violation to an external incident-management platform through its webhook API. Which two AppDynamics configuration components are required? (Choose two.)

A. A policy that matches the critical health-rule violation event

B. An HTTP Request action configured for the webhook endpoint

C. A custom dashboard containing the affected metrics

D. A Business Transaction naming rule

E. A method invocation data collector

ANSWER: A B

Explanation:

A policy and an HTTP Request action are required. The policy determines which events, such as critical health-rule violations, trigger an action. The HTTP Request action sends an outbound request to the webhook endpoint and can include event information in the request payload.

A custom dashboard displays metrics but does not initiate external notifications. A Business Transaction naming rule changes transaction identification behavior and is unrelated to alert routing. A data collector enriches diagnostic data but does not send policy events to an external system.

QUESTION NO: 11

Which two match conditions can be added when you configure an Information Point? (Choose two.)

A. Match based on a regex applied to the method

B. Match based on the invoked object

C. Match based on the Business Transaction

D. Match based on the return value

ANSWER: A B

Explanation:

“Match based on a regex applied to the method” and “Match based on the invoked object” are the available match-condition types for an AppDynamics Information Point. An Information Point is used to monitor selected method calls that may not otherwise be represented as business transactions, allowing the agent to collect metrics for calls of specific operational interest. A regular-expression method condition provides flexible selection of method calls by matching method names or signatures against a defined pattern. This is useful when the same monitoring intent applies to a group of similarly named methods rather than one fixed method.

An invoked-object condition narrows monitoring according to the object on which the method is called. This lets an analyst focus collection on calls associated with the relevant object type or instance context. These two conditions can also be combined so that the Information Point captures only calls satisfying both the method-pattern and object criteria. This targeted configuration helps retain meaningful visibility while avoiding unnecessary monitoring of unrelated calls. Cisco’s AppDynamics documentation describes Information Points and their matching configuration in the Information Points documentation and provides related monitoring configuration guidance in the Business Transactions documentation.

QUESTION NO: 12

A Performance Analyst has enabled Development Level Monitoring for an application. For a default configuration, in which scenario will Development Level Monitoring get automatically disabled?

A. A maximum of 500 calls per minute limit is exceeded, and Maximum heap utilization percentage goes above 90%

B. A maximum of 1500 calls per minute limit is exceeded, and Maximum heap utilization percentage goes above 90%

C. A maximum of 1000 calls per minute limit is exceeded, and Maximum heap utilization percentage goes above 95%

D. A maximum of 2000 calls per minute limit is exceeded, and Maximum heap utilization percentage goes above 95%

ANSWER: A

Explanation:

Development Level Monitoring is intended to provide highly detailed diagnostic visibility while an application is being developed or tested. Because this level of instrumentation can add overhead, AppDynamics includes an automatic safeguard in its default configuration. Development Level Monitoring is disabled when both protection thresholds are crossed: the application exceeds 500 calls per minute and the maximum JVM heap-utilization percentage exceeds 90%. This combination indicates that the monitored application is under sufficient load and memory pressure that retaining development-oriented instrumentation could create an undesirable performance impact. The setting is therefore designed to protect application responsiveness and agent stability while still allowing detailed monitoring in low-volume development environments. A Performance Analyst can review or adjust Development Level Monitoring settings where operational requirements justify different thresholds, but the question asks specifically for the default behavior. Cisco AppDynamics documentation describes Development Level Monitoring and its safeguards in the Java Agent configuration documentation: Java Agent Configuration Properties. For broader context on configuring Java-agent monitoring behavior, see Java Agent.

QUESTION NO: 13

The performance impact on the_____would lead a Performance Analyst to limit the duration and frequency of automatic diagnostic sessions.

A. Application

B. Controller

C. Network

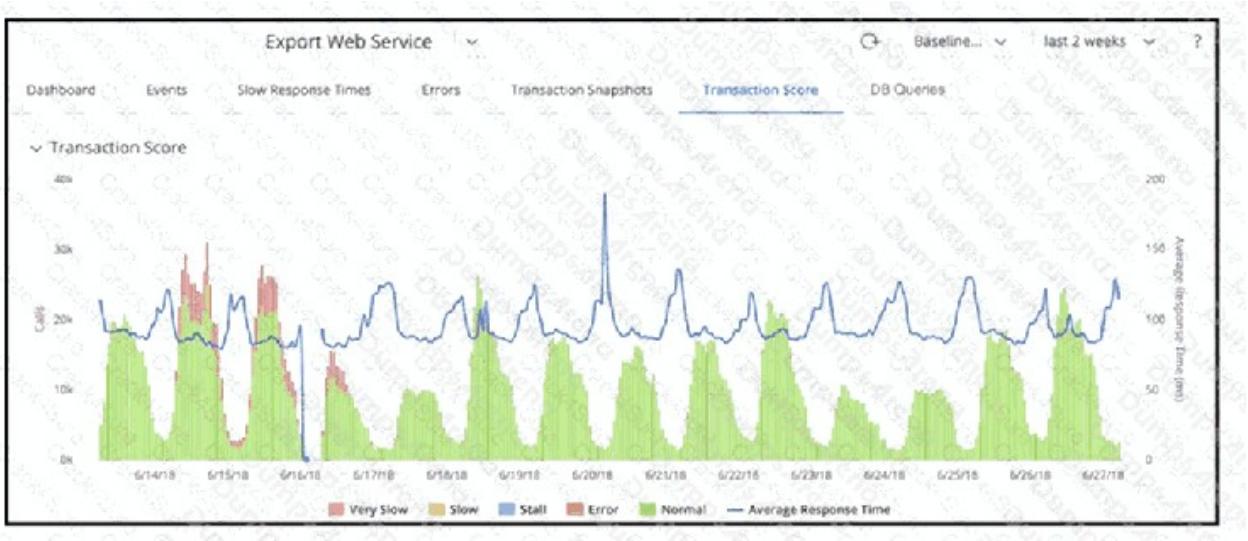
D. Operating System

ANSWER: A

Explanation:

Application is correct because automatic diagnostic sessions collect deeper runtime information from the monitored application environment, such as diagnostic data associated with affected business transactions. This additional collection activity consumes resources in the application process and on the host where the agent runs. Depending on the session configuration, transaction volume, and application workload, extended or frequent sessions can add measurable processing overhead. A Performance Analyst must therefore balance the value of detailed diagnostics against the need to preserve normal application responsiveness and a reliable end-user experience. Limiting a session's duration prevents diagnostic collection from continuing beyond the period needed to investigate an issue, while limiting frequency avoids repeatedly applying that overhead during normal operation. AppDynamics documents diagnostic sessions as a mechanism for collecting detailed diagnostic information from application agents and provides controls for configuring their operation. These controls support targeted troubleshooting while helping maintain the performance of the monitored application. See the AppDynamics Diagnostic Sessions documentation and the AppDynamics Application Monitoring documentation.

QUESTION NO: 14



Refer to the exhibit. When looking at the Transaction Score for a specific transaction, how are errors in the transaction identified?

A. Set the time range and drill down into the snapshots in the Error tab.

B. Set the time range and examine the Slow Response Times tab.

C. Set the time range and examine the dashboard for errors.

D. Set the time range and drill down into the snapshots in the Events tab.

ANSWER: A

Explanation:

Set the time range and drill down into the snapshots in the Error tab. In AppDynamics, the Transaction Scorecard provides a focused view of a selected business transaction's performance over the chosen time window. The Errors area isolates transaction executions that AppDynamics classified as errors during that interval, enabling the analyst to investigate the affected requests rather than relying only on aggregate health or response-time metrics. Opening an error snapshot supplies transaction-level diagnostic context, such as the error details, call graph, execution path, and associated backend activity. This evidence helps a Performance Analyst determine where the failed request occurred and correlate the error with application behavior in the same time period. Choosing an appropriate time range is important because the scorecard and its available snapshots are scoped to that selected interval; it ensures the analyst reviews the errors that correspond to the observed Transaction Score. AppDynamics documentation describes transaction snapshots as diagnostic captures for individual business-transaction executions and explains how error transactions can be investigated through snapshot data. See the AppDynamics Transaction Snapshots documentation and the AppDynamics Business Transaction monitoring documentation.