Updated PCA Study Demo–100% High Hit Rate Reliable Prometheus Certified Associate Exam Exam Blueprint

2026 Latest ExamsReviews PCA PDF Dumps and PCA Exam Engine Free Share: https://drive.google.com/open?id=1Jd70wRz1_CHh99j4rWZzpGL8Smpknext

We have authoritative production team made up by thousands of experts helping you get hang of our PCA study question and enjoy the high quality study experience. We will update the content of PCA test guide from time to time according to recent changes of examination outline and current policies. Besides, our PCA Exam Questions can help you optimize your learning method by simplifying obscure concepts so that you can master better. One more to mention, with our PCA test guide, there is no doubt that you can cut down your preparing time in 20-30 hours of practice before you take the exam.

Linux Foundation PCA Exam Syllabus Topics:

Topic	Details
Topic 1	• Alerting and Dashboarding: This section of the exam assesses the competencies of Cloud Operations Engineers and focuses on monitoring visualization and alert management. It covers dashboarding basics, alerting rules configuration, and the use of Alertmanager to handle notifications. Candidates also learn the core principles of when, what, and why to trigger alerts, ensuring they can create reliable monitoring dashboards and proactive alerting systems to maintain system stability.
Topic 2	• PromQL: This section of the exam measures the skills of Monitoring Specialists and focuses on Prometheus Query Language (PromQL) concepts. It covers data selection, calculating rates and derivatives, and performing aggregations across time and dimensions. Candidates also study the use of binary operators, histograms, and timestamp metrics to analyze monitoring data effectively, ensuring accurate interpretation of system performance and trends.
Topic 3	• Observability Concepts: This section of the exam measures the skills of Site Reliability Engineers and covers the essential principles of observability used in modern systems. It focuses on understanding metrics, logs, and tracing mechanisms such as spans, as well as the difference between push and pull data collection methods. Candidates also learn about service discovery processes and the fundamentals of defining and maintaining SLOs, SLAs, and SLIs to monitor performance and reliability.
Topic 4	• Instrumentation and Exporters: This domain evaluates the abilities of Software Engineers and addresses the methods for integrating Prometheus into applications. It includes the use of client libraries, the process of instrumenting code, and the proper structuring and naming of metrics. The section also introduces exporters that allow Prometheus to collect metrics from various systems, ensuring efficient and standardized monitoring implementation.

Topic 5	• Prometheus Fundamentals: This domain evaluates the knowledge of DevOps Engineers and emphasizes the core architecture and components of Prometheus. It includes topics such as configuration and scraping techniques, limitations of the Prometheus system, data models and labels, and the exposition format used for data collection. The section ensures a solid grasp of how Prometheus functions as a monitoring and alerting toolkit within distributed environments.

Free PDF Quiz 2026 Professional Linux Foundation PCA: Prometheus Certified Associate Exam Study Demo

It is our responsibility to relieve your pressure from preparation of PCA exam. To help you pass the PCA exam is our goal. The close to 100% passing rate of our dumps allow you to be rest assured in our products. Not all vendors dare to promise that if you fail the exam, we will give you a full refund. But our IT elite of ExamsReviews and our customers who are satisfied with our PCA Exam software give us the confidence to make such promise.

Linux Foundation Prometheus Certified Associate Exam Sample Questions (Q31-Q36):

NEW QUESTION # 31
What Prometheus component would you use if targets are running behind a Firewall/NAT?

- A. Pull Proxy
- B. PushProx
- C. Pull Gateway
- D. HA Proxy

Answer: B

Explanation:
When Prometheus targets are behind firewalls or NAT and cannot be reached directly by the Prometheus server's pull mechanism, the recommended component to use is PushProx.
PushProx works by reversing the usual pull model. It consists of a PushProx Proxy (accessible by Prometheus) and PushProx Clients (running alongside the targets). The clients establish outbound connections to the proxy, which allows Prometheus to "pull" metrics indirectly. This approach bypasses network restrictions without compromising the Prometheus data model.
Unlike the Pushgateway (which is used for short-lived batch jobs, not network-isolated targets), PushProx maintains the Prometheus "pull" semantics while accommodating environments where direct scraping is impossible.
Reference:
Verified from Prometheus documentation and official PushProx design notes - Monitoring Behind NAT/Firewall, PushProx Overview, and Architecture and Usage Scenarios sections.

NEW QUESTION # 32
Which metric type uses the delta() function?

- A. Info
- B. Gauge
- C. Counter
- D. Histogram

Answer: B

Explanation:
The delta() function in PromQL calculates the difference between the first and last samples in a range vector over a specified time window. This function is primarily used with gauge metrics, as they can move both up and down, and delta() captures that net change directly.
For example, if a gauge metric like node_memory_Active_bytes changes from 1000 to 1200 within a 5-minute window,

delta(node_memory_Active_bytes[5m]) returns 200.
Unlike rate() or increase(), which are designed for monotonically increasing counters, delta() is ideal for metrics representing resource levels, capacities, or instantaneous measurements that fluctuate over time.
Reference:
Verified from Prometheus documentation - PromQL Range Functions - delta(), Gauge Semantics and Usage, and Comparing delta() and rate() sections.

NEW QUESTION # 33
How can you send metrics from your Prometheus setup to a remote system, e.g., for long-term storage?

- A. With "scraping"
- B. With S3 Buckets
- C. With "remote write"
- D. With "federation"

Answer: C

Explanation:
Prometheus provides a feature called Remote Write to transmit scraped and processed metrics to an external system for long-term storage, aggregation, or advanced analytics. When configured, Prometheus continuously pushes time series data to the remote endpoint defined in the remote_write section of the configuration file.
This mechanism is often used to integrate with long-term data storage backends such as Cortex, Thanos, Mimir, or InfluxDB, enabling durable retention and global query capabilities beyond Prometheus's local time series database limits.
In contrast, "scraping" refers to data collection from targets, while "federation" allows hierarchical Prometheus setups (pulling metrics from other Prometheus instances) but does not serve as long-term storage. Using "S3 Buckets" directly is also unsupported in native Prometheus configurations.
Reference:
Extracted and verified from Prometheus documentation - Remote Write/Read APIs and Long-Term Storage Integrations sections.

NEW QUESTION # 34
How can you select all the up metrics whose instance label matches the regex fe-.*?

- A. up{instance~"fe-.*"}
- B. up{instance="fe-.*"}
- C. up{instance=~"fe-.*"}
- D. up{instance=regexp(fe-.*)}

Answer: C

Explanation:
PromQL supports regular expression matching for label values using the =~ operator. To select all time series whose label values match a given regex pattern, you use the syntax {label_name=~"regex"}.
In this case, to select all up metrics where the instance label begins with fe-, the correct query is:
up{instance=~"fe-.*"}
Explanation of operators:
= → exact match.
!= → not equal.
=~ → regex match.
!~ → regex not match.
Option D uses the correct =~ syntax. Options A and B use invalid PromQL syntax, and option C is almost correct but includes a misplaced extra quote style (~"), which would cause a parsing error.
Reference:
Verified from Prometheus documentation - Expression Language Data Selectors, Label Matchers, and Regular Expression Matching Rules.

NEW QUESTION # 35
When can you use the Grafana Heatmap panel?

- A. You can use it to graph a gauge metric.
- B. You can use it to graph a histogram metric.
- C. You can use it to graph a counter metric.
- D. You can use it to graph an info metric.

Answer: B

Explanation:
The Grafana Heatmap panel is best suited for visualizing histogram metrics collected from Prometheus. Histograms provide bucketed data distributions (e.g., request durations, response sizes), and the heatmap effectively displays these as a two-dimensional density chart over time.
In Prometheus, histogram metrics are exposed as multiple time series with the _bucket suffix and the label le (less than or equal). Grafana interprets these buckets to create visual bands showing how frequently different value ranges occurred.
Counters, gauges, and info metrics do not have bucketed distributions, so a heatmap would not produce meaningful output for them.
Reference:
Verified from Grafana documentation - Heatmap Panel Overview, Visualizing Prometheus Histograms, and Prometheus documentation - Understanding Histogram Buckets.

NEW QUESTION # 36
.....