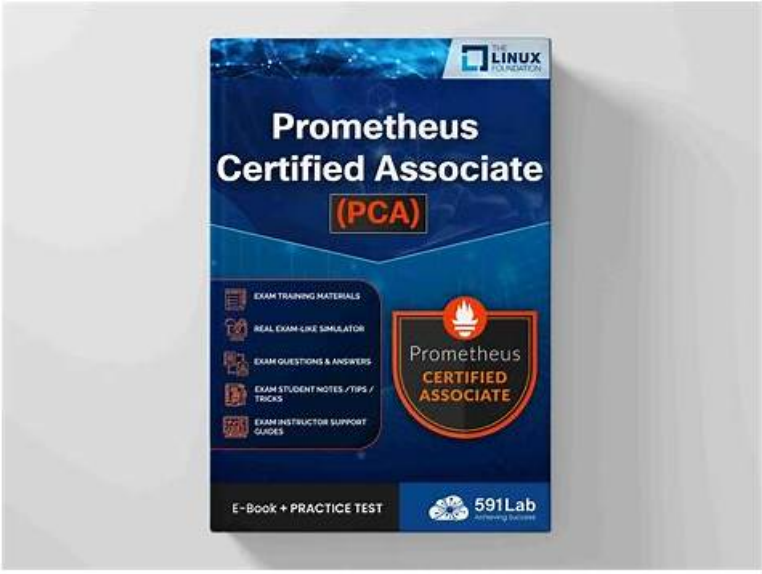
Pass Guaranteed Quiz High-quality PCA - Updated Prometheus Certified Associate Exam Dumps



2026 Latest TestsDumps PCA PDF Dumps and PCA Exam Engine Free Share: https://drive.google.com/open?id=1FGRwCozrXe1byFhDN_IjhTN-zbuRFnQk

The Linux Foundation PCA dumps PDF format of TestsDumps is portable and printable. It means you can print Linux Foundation PCA real questions for off-screen preparation. You can also access Linux Foundation PCA dumps PDF from smartphones, laptops, and tablets anywhere anytime to prepare for the PCA Exam. This version of our PCA questions PDF is beneficial for busy applicants because they can easily use PCA dumps PDF and prepare for the Linux Foundation PCA test in their homes, offices, libraries, and even while traveling.

Linux Foundation PCA Exam Syllabus Topics:

Topic	Details
Topic 1	• 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 2	• 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 3	• 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.
Topic 4	• 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 5	• 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.

Linux Foundation PCA Valid Test Notes - PCA Certification Training

You will receive an email attached with PCA exam study guide within 5-10 min after you pay. It means that you do not need to wait too long to get the dumps you want. Besides, you will have free access to the updated Linux Foundation PCA study material for one year. If there is any update, our system will send the update PCA Test Torrent to your payment email automatically. Please pay attention to your payment email for the latest Linux Foundation PCA exam dumps. If there is no any email about the update, please check your spam.

Linux Foundation Prometheus Certified Associate Exam Sample Questions (Q50-Q55):

NEW QUESTION # 50
Given the metric prometheus_tsdb_lowest_timestamp_seconds, how do you know in which month the lowest timestamp of your Prometheus TSDB belongs?

- A. prometheus_tsdb_lowest_timestamp_seconds % month
- B. month(prometheus_tsdb_lowest_timestamp_seconds)
- C. format_date(prometheus_tsdb_lowest_timestamp_seconds,"%M")
- D. (time() - prometheus_tsdb_lowest_timestamp_seconds) / 86400

Answer: D

Explanation:
The metric prometheus_tsdb_lowest_timestamp_seconds provides the oldest stored sample timestamp in Prometheus's local TSDB (in Unix epoch seconds). To determine the age or approximate date of this timestamp, you compare it with the current time (using time() in PromQL).
The expression:
(time() - prometheus_tsdb_lowest_timestamp_seconds) / 86400
converts the difference between the current time and the oldest timestamp from seconds into days (1 day = 86,400 seconds). This gives the number of days since the earliest sample was stored, allowing you to infer the time range and approximate month manually.
The other options are invalid because PromQL does not support direct date formatting (format_date) or month() extraction functions.
Reference:
Extracted and verified from Prometheus documentation - TSDB Internal Metrics, Time Functions in PromQL, and Using time() for Relative Calculations.

NEW QUESTION # 51
How do you calculate the average request duration during the last 5 minutes from a histogram or summary called http_request_duration_seconds?

- A. rate(http_request_duration_seconds_total[5m]) / rate(http_request_duration_seconds_average[5m])
- B. rate(http_request_duration_seconds_sum[5m]) / rate(http_request_duration_seconds_average[5m])
- C. rate(http_request_duration_seconds_total[5m]) / rate(http_request_duration_second$_count[5m])
- D. rate(http_request_duration_seconds_sum[5m]) / rate(http_request_duration_seconds_count[5m])

Answer: D

Explanation:

In Prometheus, histograms and summaries expose metrics with _sum and _count suffixes to represent total accumulated values and sample counts, respectively. To compute the average request duration over a given time window (for example, 5 minutes), you divide the rate of increase of _sum by the rate of increase of _count:
\text{Average duration} = \frac{\text{rate(http_request_duration_seconds_sum[5m])}}
{\text{rate(http_request_duration_seconds_count[5m])}} Here,
http_request_duration_seconds_sum represents the total accumulated request time, and
http_request_duration_seconds_count represents the number of requests observed.
By dividing these rates, you obtain the average request duration per request over the specified time range.
Reference:
Extracted and verified from Prometheus documentation - Querying Histograms and Summaries, PromQL Rate Function, and Metric Naming Conventions sections.

NEW QUESTION # 52
What is a rule group?

- A. It is a set of rules, split into groups by type.
- B. It is a set of rules that are grouped by labels.
- C. It is a set of rules that are executed sequentially.
- D. It is the set (the group) of all the rules in a file.

Answer: C

Explanation:
In Prometheus, a rule group is a logical collection of recording and alerting rules that are evaluated sequentially at a specified interval. Rule groups are defined in YAML files under the groups: key, with each group containing a name, an interval, and a list of rules.
For example:
groups:
- name: example
interval: 1m
rules:
- record: job:http_inprogress_requests:sum
expr: sum(http_inprogress_requests) by (job)
All rules in a group share the same evaluation schedule and are executed one after another. This ensures deterministic order, especially when one rule depends on another's result.
Reference:
Verified from Prometheus documentation - Rule Configuration, Rule Groups and Evaluation Order, and Recording & Alerting Rules Guide.

NEW QUESTION # 53
Where does Prometheus store its time series data by default?

- A. In an embedded TSDB on local disk.
- B. In-memory only.
- C. In etcd.
- D. In an external database such as InfluxDB.

Answer: A

Explanation:
By default, Prometheus stores its time series data in a local, embedded Time Series Database (TSDB) on disk. The data is organized in block files under the data/ directory inside Prometheus's storage path.
Each block typically covers two hours of data, containing chunks, index, and metadata files. Older blocks are compacted and deleted based on retention settings.

NEW QUESTION # 54
Which of the following is an invalid @ modifier expression?

- A. go_goroutines @ start()
- B. sum(http_requests_total{method="GET"}) @ 1609746000
- C. go_goroutines @ end()
- D. sum(http_requests_total{method="GET"} @ 1609746000)

Answer: D

Explanation:
The @ modifier in PromQL allows querying data as it existed at a specific point in time rather than the evaluation time. It can be applied after a selector or an entire expression, but the syntax rules are strict.
□ go_goroutines @ start() → Valid; queries value at the start of the evaluation range.
□ sum(http_requests_total{method="GET"}) @ 1609746000 → Valid; applies the modifier after the full expression.
□ go_goroutines @ end() → Valid; queries value at the end of the evaluation range.
□ sum(http_requests_total{method="GET"} @ 1609746000) → Invalid, because the @ modifier cannot appear inside the selector braces; it must appear after the selector or aggregation expression.
This invalid placement violates PromQL's syntax grammar for subquery and modifier ordering.
Reference:
Verified from Prometheus documentation - PromQL @ Modifier Syntax, Evaluation Modifiers, and PromQL Expression Grammar sections.

NEW QUESTION # 55
.....

Our TestsDumps will provide you with the most satisfying after sales service. We provide one-year free update service to you one year after you have purchased PCA exam software., which can make you have a full understanding of the latest and complete PCA Questions so that you can be confident to pass the exam. If you are unlucky to fail PCA exam for the first time, we will give you a full refund of the cost you purchased our dump to make up your loss.