PASS QUESTION

PRACTICE QUESTIONS

EXAM PREPARATION GUIDE

Learn Smarter

Practice Effectively

Achieve Your Goals

https://www.passquestion.com

One Year of Content Updates Included

Exam : **PCA**

Title : Prometheus Certified Associate Exam

Version : DEMO

1.How many metric types does Prometheus text format support?
A. 40
B. 4
C. 8
D. 10
Answer: B
Explanation:
Prometheus defines four core metric types in its official exposition format, which are: Counter, Gauge, Histogram, and Summary. These types represent the fundamental building blocks for expressing quantitative measurements of system performance, behavior, and state.
A Counter is a cumulative metric that only increases (e.g., number of requests served).
A Gauge represents a value that can go up and down, such as memory usage or temperature.
A Histogram samples observations (e.g., request durations) and counts them in configurable buckets, providing both counts and sum of observed values.
A Summary is similar to a histogram but provides quantile estimation over a sliding time window along with count and sum metrics.
These four types are the only officially supported metric types in the Prometheus text exposition format as defined by the Prometheus data model. Any additional metrics or custom naming conventions are built on top of these core types but do not constitute new types.
Reference: Extracted and verified from Prometheus official documentation sections on Metric Types and Exposition Formats in the Prometheus study materials.

2.What does scrape_interval configure in Prometheus?
A. It defines how frequently to scrape targets.
B. It defines how frequently to evaluate rules.
C. It defines how often to send alerts.
D. It defines how often to refresh metrics.
Answer: A
Explanation:
In Prometheus, the scrape_interval parameter specifies how frequently the Prometheus server should scrape metrics from its configured targets. Each target exposes an HTTP endpoint (usually /metrics) that Prometheus collects data from at a fixed cadence. By default, the scrape_interval is set to 1 minute, but it can be overridden globally or per job configuration in the Prometheus YAML configuration file.
This setting directly affects the resolution of collected time series data—a shorter interval increases data granularity but also adds network and storage overhead, while a longer interval reduces load but might miss short-lived metric variations.
It is important to distinguish scrape_interval from evaluation_interval, which defines how often Prometheus evaluates recording and alerting rules. Thus, scrape_interval pertains only to data collection frequency, not to alerting or rule evaluation.
Reference: Extracted and verified from Prometheus documentation on Configuration File – scrape_interval and Scraping Fundamentals sections.

3.What is a difference between a counter and a gauge?
A. Counters change value on each scrape and gauges remain static.

B. Counters and gauges are different names for the same thing.
C. Counters have no labels while gauges can have many labels.
D. Counters are only incremented, while gauges can go up and down.
Answer: D
Explanation:
The key difference between a counter and a gauge in Prometheus lies in how their values change over time. A counter is a cumulative metric that only increases—it resets to zero only when the process restarts. Counters are typically used for metrics like total requests served, bytes processed, or errors encountered. You can derive rates of change from counters using functions like rate() or increase() in PromQL.
A gauge, on the other hand, represents a metric that can go up and down. It measures values that fluctuate, such as CPU usage, memory consumption, temperature, or active session counts. Gauges provide a snapshot of current state rather than a cumulative total.
This distinction ensures proper interpretation of time-series trends and prevents misrepresentation of one-time or fluctuating values as cumulative metrics.
Reference: Extracted and verified from Prometheus official documentation – Metric Types section explaining Counters and Gauges definitions and usage examples.

4.What's "wrong" with the myapp_filG_uploads_total{userid=,,5123",status="failed"} metric?
A. The userid should not be exposed as a label.
B. The status should not be exposed as a label.
C. The _total suffix should be omitted.
D. The metric name should consist of dashes instead of underscores.
Answer: A
Explanation:
In Prometheus best practices, high-cardinality labels—especially those containing unique or user-specific identifiers—should be avoided. The metric myapp_filG_uploads_total{userid="5123",status="failed"} exposes the userid as a label, which is problematic. Each distinct value of a label generates a new time series in Prometheus. If there are thousands or millions of unique users, this would exponentially increase the number of time series, leading to cardinality explosion, degraded performance, and high memory usage.
The _total suffix is actually correct and required for counters, as per the Prometheus naming convention. The use of underscores in metric names is also correct, as Prometheus does not support dashes in metric identifiers. The status label, however, is perfectly valid because it typically has a low number of possible values (e.g., “success”, “failed”).
Reference: Verified from Prometheus official documentation sections Instrumentation – Metric and Label Naming Best Practices and Writing Exporters.

5.Which exporter would be best suited for basic HTTP probing?
A. JMX exporter
B. Blackbox exporter
C. Apache exporter
D. SNMP exporter
Answer: B

Explanation:
The Blackbox Exporter is the Prometheus component designed specifically for probing endpoints over various network protocols, including HTTP, HTTPS, TCP, ICMP, and DNS. It acts as a generic probe service, allowing Prometheus to test endpoints' availability, latency, and correctness without requiring instrumentation in the target application itself.
For basic HTTP probing, the Blackbox Exporter performs HTTP GET or POST requests to defined URLs and exposes metrics like probe success, latency, response code, and SSL certificate validity. This makes it ideal for uptime and availability monitoring.
By contrast, the JMX exporter is used for collecting metrics from Java applications, the Apache exporter for Apache HTTP Server metrics, and the SNMP exporter for network devices. Thus, only the Blackbox Exporter serves the purpose of HTTP probing.
Reference: Verified from Prometheus documentation – Blackbox Exporter Overview and Exporter Usage Guidelines.