Linux Foundation PCA최신버전덤프공부 - PCA최신시험예상문제모음

ITDumpsKR PCA 최신 PDF 버전 시험 문제집을 무료로 Google Drive에서 다운로드하세요: https://drive.google.com/open?id=1LfRMt3ZU3CQSIJOZ8LEn5C5MYnOhhPGW

ITDumpsKR의 Linux Foundation PCA덤프로Linux Foundation PCA시험준비를 하면 시험패스는 간단한 일이라는걸 알게 될것입니다. Linux Foundation PCA덤프는 최근Linux Foundation PCA시험의 기출문제모음으로 되어있기에 적중율이 높습니다.시험에서 떨어지면 덤프비용 전액 환불해드리기에 우려없이 덤프를 주문하셔도 됩니다.

Linux Foundation PCA 시험요강:

주제	소개
주제 1	• 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.
주제 2	• 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.
주제 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.

주제 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.
주제 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.

최신 Cloud & Containers PCA 무료샘플문제 (Q16-Q21):

질문 # 16
Which metric type uses the delta() function?

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

정답：C

설명：
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.

질문 # 17
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. format_date(prometheus_tsdb_lowest_timestamp_seconds,"%M")
- C. (time() - prometheus_tsdb_lowest_timestamp_seconds) / 86400
- D. month(prometheus_tsdb_lowest_timestamp_seconds)

정답：C

설명：

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.

질문 # 18
What's "wrong" with the myapp_filG_uploads_total{userid=,,5123",status="failed"} metric?

- A. The status should not be exposed as a label.
- B. The _total suffix should be omitted.
- C. The metric name should consist of dashes instead of underscores.
- D. The userid should not be exposed as a label.

정답：D

설명：
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.

질문 # 19
Which Alertmanager feature prevents duplicate notifications from being sent?

- A. Inhibition
- B. Grouping
- C. Deduplication
- D. Silencing

정답：C

설명：
Deduplication in Alertmanager ensures that identical alerts from multiple Prometheus servers or rule evaluations do not trigger duplicate notifications.
Alertmanager compares alerts based on their labels and fingerprints; if an alert with identical labels already exists, it merges or refreshes the existing one instead of creating a new notification.
This mechanism is essential in high-availability setups where multiple Prometheus instances monitor the same targets.

질문 # 20
What is a difference between a counter and a gauge?

- A. Counters change value on each scrape and gauges remain static.
- B. Counters have no labels while gauges can have many labels.

- C. Counters are only incremented, while gauges can go up and down.
- D. Counters and gauges are different names for the same thing.

정답：C

설명：
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.

질문 # 21
.....

myportal.utt.edu.tt, 24hoursschool.com, www.stes.tyc.edu.tw, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, Disposable vapes