Google Discussions

EXAM PROFESSIONAL DATA ENGINEER TOPIC 1 QUESTION 266 DISCUSSION

Actual exam question from Google's Professional Data Engineer
Question #: 266
Topic #: 1
[All Professional Data Engineer Questions]

You are building a streaming Dataflow pipeline that ingests noise level data from hundreds of sensors placed near construction sites across a city. The sensors measure noise level every ten seconds, and send that data to the pipeline when levels reach above 70 dBA. You need to detect the average noise level from a sensor when data is received for a duration of more than 30 minutes, but the window ends when no data has been received for 15 minutes. What should you do?

A. Use session windows with a 15-minute gap duration.

B. Use session windows with a 30-minute gap duration.

C. Use hopping windows with a 15-minute window, and a thirty-minute period.

D. Use tumbling windows with a 15-minute window and a fifteen-minute .withAllowedLateness operator.

Show Suggested Answer

by scaenruy at *Jan. 3, 2024, 6:23 p.m.*

Comments

Type your comment...

Submit

datapassionate Highly Voted 1 year, 3 months ago

Selected Answer: A

to detect average noise levels from sensors, the best approach is to use session windows with a 15-minute gap duration (Option A). Session windows are ideal for cases like this where the events (sensor data) are sporadic. They group events that occur within a certain time interval (15 minutes in your case) and a new window is started if no data is received for the duration of the gap. This matches your requirement to end the window when no data is received for 15 minutes, ensuring that the average noise level is calculated over periods of continuous data

upvoted 11 times

ashdam 1 year, 2 months ago

But you are not fulfilling this requirement "You need to detect the average noise level from a sensor when data is received for a duration of more than 30 minutes". I would say C

upvoted 2 times

saschak94 Highly Voted 1 year, 2 months ago

Selected Answer: A

You need a window that start when data for a sensor arrives and end when there's a gap in data. That would rule out hopping and tumbling windows.

- > Windows need to stay open as long as there's data arriving - 30+ mins
-> Window Should close when no data has been received for 15 mins -> Gap 15 mins

upvoted 10 times

Pime13 Most Recent 3 months, 3 weeks ago

Selected Answer: D

D for me, clearly stated on example: https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#hopping-windows

upvoted 1 times

shanks_t 8 months, 2 weeks ago

Selected Answer: A

The problem requires detecting average noise levels when data is received for more than 30 minutes, but the window should end when no data has been received for 15 minutes.
Session windows are ideal for this scenario because:
They are designed to capture bursts of activity followed by periods of inactivity.
They dynamically size based on the data received, which fits well with the variable duration of noise events.
The gap duration can be set to define when a session ends.
The 15-minute gap duration aligns perfectly with the requirement to end the window when no data has been received for 15 minutes.
Session windows will naturally extend beyond 30 minutes if data keeps coming in, satisfying the requirement to detect levels for durations of more than 30 minutes.

upvoted 3 times

viciousjpjp 8 months, 3 weeks ago

Selected Answer: D

D. Using a 15-minute window with a 15-minute tumbling window withAllowedLateness is the most suitable option for the following reasons:

Flexibility: By allowing a 15-minute delay, it can accommodate various situations such as network latency or sensor failures.
Processing efficiency: Using a fixed window improves processing efficiency.
Compliance with conditions: The window ends if no data is received for 15 minutes, meeting the specified condition.
Implementation points:

.withAllowedLateness: This operator allows delayed events to be included in the current window.
Trigger: When 30 minutes of data is collected, a trigger event is generated, and the average value is calculated based on this event.
Watermark: By setting a watermark, processing of old data can be terminated.

upvoted 1 times

JamesKarianis 9 months ago

Selected Answer: A

Without a doubt A: https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#session-windows

upvoted 3 times

iooj 9 months ago

Selected Answer: A

The requirements are
- recieve data for a duration of MORE than 30 minutes
- end the window based on inactivity

upvoted 1 times

josech 11 months, 3 weeks ago

josech 11 months, 3 weeks ago

Correct answer: A.
Use a session Window to cature data and create an aggregation when the Session is larger than 30 minutes.
https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#session-windows
https://beam.apache.org/releases/javadoc/2.6.0/org/apache/beam/sdk/transforms/windowing/Sessions.html

upvoted 3 times

f74ca0c 11 months, 3 weeks ago

Selected Answer: C

C- Running average: https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#hopping-windows

upvoted 1 times

joao_01 1 year ago

Guys, I think its B. I was considering C, but C is calculating every 30 min, the 15min window gap data.Thats not what the questions wants. The questions wants a solution to get the average data of a 30 min window. So Its B.

Look at this relating to C:
"Use hopping windows with a 15-minute window, and a thirty-minute period" --> Wrong
(IS DIFFERENT THEN)
"Use hopping windows with a 30-minute window, and a 15-minute period" --> Right.

Thats why I think the B is the right answer.

upvoted 2 times

joao_01 1 year ago

Actually I think with B, the window will never be closed because the probability of having a period of inactivity of 30 minutes is very low. In that case I think the option A is the more correct one.

upvoted 1 times

342f1c6 1 year, 1 month ago

Selected Answer: C

To take running averages of data, use hopping windows. You can use one-minute hopping windows with a thirty-second period to compute a one-minute running average every thirty seconds.

upvoted 1 times

kck6ra4214wm 1 year, 2 months ago

Selected Answer: A

https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#session-windows
A session window contains elements within a gap duration of another element. The gap duration is an interval between new data in a data stream. If data arrives after the gap duration, the data is assigned to a new window.
So, we need

upvoted 3 times

imiu 1 year, 3 months ago

Selected Answer: C

https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#hopping-windows To take running averages of data, use hopping windows. You can use one-minute hopping windows with a thirty-second period to compute a one-minute running average every thirty seconds.

upvoted 1 times

Matt_108 1 year, 3 months ago

Selected Answer: D

Option D to me, It aligns with the specified criteria for detecting the average noise level within a 30-minute duration and handling the end of the window when no data is received for 15 minutes.

upvoted 2 times

AllenChen123 1 year, 3 months ago

Agree D.
Data comes -> 30 mts duration.
Data didn't come in 15 mts -> 15 mts duration

upvoted 1 times

BIGQUERY_ALT_ALT 1 year, 3 months ago

Selected Answer: D

OPTION D is correct for the specific scenario where we want to detect the average noise level for a duration of more than 30 minutes but end the window when no data has been received for 15 minutes.

Explanation:

- Tumbling windows are non-overlapping windows, and in this case, you want to capture data continuously for 30-minute

- Tumbling windows are non-overlapping windows, and in this case, you want to capture data continuously for 30-minute intervals.

- Using a tumbling window with a 15-minute window size aligns with your requirement to detect the average noise level for a duration of more than 30 minutes.

- Adding a .withAllowedLateness operator with a duration of fifteen minutes ensures that the window will still consider late-arriving data within that time frame. After fifteen minutes of no data, the window will be closed, and any late-arriving data will not be considered.

Option A and B invalid as they capture fixed logic with 15 or 30 mins. Option C captures only 15 min average with 30 min trigger hence not suitable.

upvoted 2 times

Sofiia98 1 year, 3 months ago

Selected Answer: C

Hopping windows (called sliding windows in Apache Beam).
To take running averages of data, use hopping windows.

upvoted 2 times

scaenruy 1 year, 4 months ago

Selected Answer: C

C. Use hopping windows with a 15-minute window, and a thirty-minute period.

upvoted 3 times