Google Discussions

Exam Professional Data Engineer All Questions

View all questions & answers for the Professional Data Engineer exam

Go to Exam

EXAM PROFESSIONAL DATA ENGINEER TOPIC 1 QUESTION 57 DISCUSSION

Actual exam question from Google's Professional Data Engineer

Question #: 57

Topic #: 1

[All Professional Data Engineer Questions]

Your company is currently setting up data pipelines for their campaign. For all the Google Cloud Pub/Sub streaming data, one of the important business requirements is to be able to periodically identify the inputs and their timings during their campaign. Engineers have decided to use windowing and transformation in Google Cloud Dataflow for this purpose. However, when testing this feature, they find that the Cloud Dataflow job fails for the all streaming insert. What is the most likely cause of this problem?

A. They have not assigned the timestamp, which causes the job to fail

B. They have not set the triggers to accommodate the data coming in late, which causes the job to fail

C. They have not applied a global windowing function, which causes the job to fail when the pipeline is created

D. They have not applied a non-global windowing function, which causes the job to fail when the pipeline is created

Show Suggested Answer

by jvg637 at *March 15, 2020, 5:05 p.m.*

Comments

Type your comment...

Submit

⊟ [Removed] Highly Voted 👍 5 years, 1 month ago

Answer: D
Description: Caution: Beam's default windowing behavior is to assign all elements of a PCollection to a single, global window and discard late data, even for unbounded PCollections. Before you use a grouping transform such as GroupByKey on an unbounded PCollection, you must do at least one of the following:
—->>>>>>Set a non-global windowing function. See Setting your PCollection's windowing function.
Set a non-default trigger. This allows the global window to emit results under other conditions, since the default windowing behavior (waiting for all data to arrive) will never occur.
—->>>>If you don't set a non-global windowing function or a non-default trigger for your unbounded PCollection and subsequently use a grouping transform such as GroupByKey or Combine, your pipeline will generate an error upon construction and your job will fail.
So it looks like D

👍 ↩ ⚑ upvoted 67 times

samdhimal 2 years, 3 months ago

Why not C?
Because I think that the most likely cause of the problem is C. They have not applied a global windowing function, which causes the job to fail when the pipeline is created.

In Dataflow, windowing is used to divide the input data into smaller time intervals, called windows. Without a windowing function, all the data may be treated as part of the same window and the pipeline may not be able to process the data correctly. In this specific scenario, the engineers are trying to use windowing and transformation in Google Cloud Dataflow to periodically identify the inputs and their timings during the campaign, so it's likely that they need to use a windowing function to divide the data into smaller time intervals in order to process it correctly. Not applying a windowing function, or applying the wrong one can cause the job to fail.

Someone Clarify? Am I missing an important point?

👍 ↩ ⚑ upvoted 1 times

Mathew106 1 year, 9 months ago

You are missing that the global window is the default window that we typically use for batch processing. The global window by default waits until all data is available before processing it so if you want to use it with streaming you need to set some custom trigger so that we don't wait indefinitely to wait until we aggregate. All in all it doesn't sound right.

https://www.youtube.com/watch?v=oJ-LueBvOcM
https://www.youtube.com/watch?v=MuFA6CSti6M

👍 ↩ ⚑ upvoted 4 times

jvg637 Highly Voted 👍 5 years, 1 month ago

Global windowing is the default behavior, so I don't think C is right.
An error can occur if a non-global window or a non-default trigger is not set.
I would say D.
(https://beam.apache.org/documentation/programming-guide/#windowing)

👍 ↩ ⚑ upvoted 15 times

Parandhaman_Margan Most Recent ⊙ 1 month, 3 weeks ago

Selected Answer: A

Google Cloud Dataflow requires event timestamps when using windowing in streaming mode.
By default, Pub/Sub messages do not have timestamps; they need to be assigned manually using withTimestampFn()

👍 ↩ ⚑ upvoted 1 times

Yad_datatonic 3 months, 1 week ago

Selected Answer: B

The job fails because triggers are not set to handle late-arriving data, causing the pipeline to mishandle or drop delayed records.

👍 ↩ ⚑ upvoted 1 times

Rav761 4 months, 1 week ago

Selected Answer: A

A. They have not assigned the timestamp, which causes the job to fail

Analysis: Cloud Dataflow relies on timestamps to perform windowing operations. Without proper event-time timestamps, windowing cannot be applied correctly, and the job may fail or behave unpredictably. This is a common issue when processing streaming data from Google Cloud Pub/Sub, as timestamps must be explicitly assigned if not already embedded in the data.
This is the most likely cause.

👍 ↩ ⚑ upvoted 1 times

Erg_de 6 months ago

Selected Answer: A

This option is very likely, as without timestamps assigned to streaming data, the system cannot properly process time windows. Timestamps are crucial for the correct time handling in Dataflow pipelines

upvoted 2 times

39405bb 11 months, 2 weeks ago

The most likely cause of this problem is A. They have not assigned the timestamp, which causes the job to fail.

Here's why:

Importance of Timestamps in Windowing: Windowing in Dataflow relies on timestamps to group elements into windows. If timestamps are not explicitly assigned or extracted from the data, Dataflow cannot determine which elements belong to which windows, leading to failures in the job.
Let's look at the other options:

B. They have not set the triggers to accommodate the data coming in late: While triggers are important for managing late data, not setting them would not cause the job to fail for all streaming inserts. It might affect the accuracy of the results, but the job would still run.
C & D. Global vs. Non-global Windowing: The choice between global and non-global windowing depends on the specific requirements of the analysis. While incorrect windowing choices can lead to unexpected results, they would not typically cause the job to fail completely.

upvoted 6 times

philli1011 1 year, 3 months ago

D
You have to apply a non-global windowing function because the global windowing function is a default windowing function for every pub/sub stream or batch data.

upvoted 1 times

MikkelRev 1 year, 7 months ago

option B: They have not set the triggers to accommodate the data coming in late, which causes the job to fail.

In a streaming data processing pipeline, it's common to encounter data that arrives late, meaning it arrives after the event time has passed for the associated window. If you don't handle late data appropriately by setting triggers, it can cause issues in your pipeline, including job failures.

upvoted 2 times

Oleksandr0501 2 years ago

Selected Answer: A

gpt: The most likely cause of the problem is A, that they have not assigned the timestamp.
In streaming data processing, timestamps are essential for proper windowing and triggering of data. Without timestamps, the system cannot correctly determine which window a particular piece of data belongs to, or when it is safe to trigger processing of a window. If the engineers did not assign timestamps to the data, the Cloud Dataflow job would not be able to process the data correctly, and it would fail.

Option B, not setting triggers to accommodate late data, is also an important consideration for streaming data processing. However, it is less likely to cause the job to fail outright than missing timestamps.

Option C, not applying a global windowing function, and Option D, not applying a non-global windowing function, are also important considerations for windowing in Cloud Dataflow. However, neither of these would cause the job to fail when the pipeline is created. Instead, they would affect the performance and correctness of the data processing.

upvoted 2 times

> **muhusman** 2 years ago
>
> without a correct timestamp, the pipeline still run fine with the default timestamp. The result maybe incorrect but the job will not fail.
>
> upvoted 4 times
>
> > **Oleksandr0501** 2 years ago
> >
> > okay, so D maybe
> >
> > upvoted 2 times
> >
> > **Oleksandr0501** 2 years ago
> >
> > gpt pt2: For example, if your use case requires you to calculate a running average of values over a fixed time interval, you would likely use a non-global windowing function with a fixed time interval. On the other hand, if you need to perform a computation on the entire stream of data at once, a global windowing function might be more appropriate.
> >
> > So, the choice of windowing function should be based on the specific requirements of the data processing task at hand, and it may or may not be important to apply a non-global windowing function when the pipeline is created.
> > ------
> > question says that we need to identify streaming input, time, so non-global needed, let it be d...
> >
> > upvoted 1 times
> >
> > **Oleksandr0501** 2 years ago

- is it impotant to apply a non-global windowing function when the pipeline is created?
GPT: It is important to choose the appropriate windowing function for your data processing needs, but whether it should be a global or non-global windowing function depends on the requirements of your specific use case.
A global windowing function considers all data elements within a bounded time interval as a single window, whereas a non-global windowing function divides the data stream into smaller windows based on specified criteria (such as a fixed time interval or a number of elements).

upvoted 1 times

emmylou 1 year, 5 months ago

Which is the moment I decided that AI was nothing to fear

upvoted 1 times

lucaluca1982 2 years, 1 month ago

what about A? This can cause the job to fail

upvoted 2 times

midgoo 2 years, 2 months ago

Selected Answer: D

A: note that without a correct timestamp, the pipeline still run fine with the default timestamp. The result maybe incorrect but the job will not fail.
D: For unbound collection, this will fail if any aggregation function is done.

upvoted 1 times

musumusu 2 years, 2 months ago

Answer: A
All Streaming Insert failed, because there is no TimeStamp added, otherwise there is already a DEFAULT global windowing function and can execute without assigning any windowing function.
I mean first there should be Timestamp in the data, then according to our aggregation outcome either its full time (global) or batch/chunks time aggregation(non global) will be performed.

upvoted 2 times

DipT 2 years, 4 months ago

Selected Answer: D

https://beam.apache.org/documentation/programming-guide/#windowing
Beam's default windowing behavior is to assign all elements of a PCollection to a single, global window and discard late data, even for unbounded PCollections. Before you use a grouping transform such as GroupByKey on an unbounded PCollection, you must do at least one of the following:

Set a non-global windowing function. See Setting your PCollection's windowing function.
Set a non-default trigger. This allows the global window to emit results under other conditions, since the default windowing behavior (waiting for all data to arrive) will never occur.

upvoted 1 times

Ray0506 2 years, 7 months ago

Selected Answer: D

Answer is D

upvoted 1 times

TOXICcharlie 2 years, 7 months ago

Selected Answer: D

Correct answer is D. C does not make sense because for unbounded source like Pub/Sub, the global functions are applied by default. The reason for failure would be they are using specific aggregations that require non-global window functions, e.g. tumbling or hopping windows.

upvoted 2 times

FrankT2L 2 years, 11 months ago

Selected Answer: C

C is the answer.
https://beam.apache.org/documentation/programming-guide/#windowing-bounded-collections

8.2.4. The single global window
By default, all data in a PCollection is assigned to the single global window, and late data is discarded. If your data set is of a fixed size, you can use the global window default for your PCollection (not our case because we are streaming).
You can use the single global window if you are working with an unbounded data set (e.g. from a streaming data source) but use caution when applying aggregating transforms such as GroupByKey and Combine. The single global window with a default trigger generally requires the entire data set to be available before processing, which is not possible with continuously updating data. To perform aggregations on an unbounded PCollection that uses global windowing, you should specify a non-default trigger for that PCollection.

upvoted 1 times

Load full discussion...

EXAMTOPICS

Platform

- Home
- All Exams
- Examtopics PRO
- Training Courses