

[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 170 DISCUSSION

Actual exam question from Google's Professional Data Engineer

Question #: 170

Topic #: 1

[\[All Professional Data Engineer Questions\]](#)

You are updating the code for a subscriber to a Pub/Sub feed. You are concerned that upon deployment the subscriber may erroneously acknowledge messages, leading to message loss. Your subscriber is not set up to retain acknowledged messages. What should you do to ensure that you can recover from errors after deployment?



- A. Set up the Pub/Sub emulator on your local machine. Validate the behavior of your new subscriber logic before deploying it to production.
- B. Create a Pub/Sub snapshot before deploying new subscriber code. Use a Seek operation to re-deliver messages that became available after the snapshot was created.
- C. Use Cloud Build for your deployment. If an error occurs after deployment, use a Seek operation to locate a timestamp logged by Cloud Build at the start of the deployment.
- D. Enable dead-lettering on the Pub/Sub topic to capture messages that aren't successfully acknowledged. If an error occurs after deployment, re-deliver any messages captured by the dead-letter queue.

[Show Suggested Answer](#)

by [AWSandeep](#) at *Sept. 2, 2022, 7:38 p.m.*

Comments

Type your comment...

  **AWSandeep** Highly Voted  2 years, 2 months ago

Selected Answer: B

B. Create a Pub/Sub snapshot before deploying new subscriber code. Use a Seek operation to re-deliver messages that became available after the snapshot was created.

According to the second reference in the list below, a concern with deploying new subscriber code is that the new executable may erroneously acknowledge messages, leading to message loss. Incorporating snapshots into your deployment process gives you a way to recover from bugs in new subscriber code.



Answer cannot be C because To seek to a timestamp, you must first configure the subscription to retain acknowledged messages using retain-acked-messages. If retain-acked-messages is set, Pub/Sub retains acknowledged messages for 7 days.

References:

<https://cloud.google.com/pubsub/docs/replay-message>



https://cloud.google.com/pubsub/docs/replay-overview#seek_use_cases

   upvoted 13 times

  **jkhong** 1 year, 11 months ago

Don't think we need to configure subscription to retain ack messages. It is defaulted to retain for 7 days

   upvoted 1 times


  **Pime13** Most Recent  3 months, 3 weeks ago

Selected Answer: B

Creating a snapshot allows you to capture the state of your subscription at a specific point in time. If an error occurs, you can use the Seek operation to reset the acknowledgment state of messages to the snapshot, ensuring that no messages are lost

<https://cloud.google.com/pubsub/docs/reference/rest/v1/Snapshot>

   upvoted 1 times

  **Pime13** 3 months, 3 weeks ago

not D because: Dead-lettering is useful for handling messages that can't be processed successfully, but it doesn't help with messages that have been erroneously acknowledged. Once a message is acknowledged, it is considered processed and won't be sent to the dead-letter queue.

   upvoted 1 times

  **f74ca0c** 4 months ago

Selected Answer: C

C. Use a BigQuery view to define your preprocessing logic. When creating your model, use the view as your model training data. At prediction time, use BigQuery's ML.EVALUATE clause without specifying any transformations on the raw input data.

Explanation:

Preventing Data Skew:

Training-serving skew occurs when the transformations applied to training data are not identically applied to prediction data. Using a BigQuery view ensures consistent preprocessing for both training and prediction.

Advantages of BigQuery Views:

Views encapsulate preprocessing logic, ensuring that the same transformations are applied whenever the view is queried.

By referencing the view during both training and prediction, you eliminate the need for manual transformations and the risk of discrepancies.

   upvoted 1 times

  **MaxNRG** 10 months, 2 weeks ago

Selected Answer: B

Taking a snapshot allows redelivering messages that were published while any faulty subscriber logic was running.

The seek timestamp would come after deployment so even erroneously acknowledged messages could be recovered.

https://cloud.google.com/pubsub/docs/replay-overview#seek_use_cases

By creating a snapshot of the subscription before deploying new code, you can preserve the state of unacknowledged messages. If after deployment you find that the new subscriber code is erroneously acknowledging messages, you can use the Seek operation with the snapshot to reset the subscription's acknowledgment state to the time the snapshot was created. This would effectively re-deliver messages available since the snapshot, ensuring you can recover from errors. This approach does not require setting up a local emulator and directly addresses the concern of message loss due to erroneous acknowledgments.

   upvoted 2 times

  **[Removed]** 1 year, 1 month ago

Selected Answer: B

B.

from the documentation:

<https://cloud.google.com/pubsub/docs/replay-message>



Pub/Sub cannot retrieve the messages after you have acknowledged them. However, sometimes you might find it necessary to replay the acknowledged messages, for example, if you performed an erroneous acknowledgment. Then you can use the Seek feature to mark previously acknowledged messages as unacknowledged, and force Pub/Sub to redeliver those messages. You can also use seek to delete the unacknowledged messages by changing their state to acknowledged.

   upvoted 3 times

  **vamgcp** 1 year, 3 months ago

pls correct me if I am wrong , option B Option B only allows you to re-deliver messages that were available before the snapshot was created. If an error occurs after the snapshot was created, you will not be able to re-deliver those messages.

   upvoted 2 times

  **cetanx** 1 year, 4 months ago

Selected Answer: A

Q: You are concerned that upon deployment the subscriber may erroneously acknowledge messages, leading to message loss.

-> So the message is mistakenly acked and removed from topic/subscription. This means even if you have a snapshot of pre-deployment but you don't have a backup or copy of post-deployment messages.

Q: Your subscriber is not set up to retain acknowledged messages.

-> To seek to a time in the past and replay previously-acknowledged messages, "you must first configure message retention on the topic" or "configure the subscription to retain acknowledged messages" (ref:

https://cloud.google.com/pubsub/docs/replay-overview#configuring_message_retention)

So B, C, D do not solve the problem of erroneously acked messages as long as you don't have message retention configured on topic/subscription.

   upvoted 3 times

  **lucaluca1982** 1 year, 7 months ago

Selected Answer: D

You are updating the code for a subscriber to a Pub/Sub feed. You are concerned that upon deployment the subscriber may erroneously acknowledge messages, leading to message loss. Your subscriber is not set up to retain acknowledged messages. What should you do to ensure that you can recover from errors after deployment?

A. Set up the Pub/Sub emulator on your local machine. Validate the behavior of your new subscriber logic before deploying it to production.

B. Create a Pub/Sub snapshot before deploying new subscriber code. Use a Seek operation to re-deliver messages that became available after the snapshot was created.

C. Use Cloud Build for your deployment. If an error occurs after deployment, use a Seek operation to locate a timestamp logged by Cloud Build at the start of the deployment.

D. Enable dead-lettering on the Pub/Sub topic to capture messages that aren't successfully acknowledged. If an error occurs after deployment, re-deliver any messages captured by the dead-letter queue.

   upvoted 1 times

  **musumusu** 1 year, 8 months ago

Option D:

Dead letter option allow you to recover message from errors after deployment by re-delivering any messages captured by the dead-letter queue.

https://cloud.google.com/pubsub/docs/handling-failures#dead_letter_topic

why not B,

because snapshot is time taking process and if messages were erroneously acknowledged, it will not bring them back. It is useful when you want to secure the current data and want to make changes

   upvoted 2 times

  **wjtb** 1 year, 7 months ago

Dead letter queue would help if the messages would not get acknowledged, however here they are talking about messages being erroneously acknowledged. Pub/Sub would interpret the message as being successfully processed -> they would not end up in the dead-letter queue -> D is wrong

   upvoted 7 times

  **zelck** 1 year, 11 months ago

Selected Answer: B

B is the answer.

<https://cloud.google.com/pubsub/docs/replay-overview>

The Seek feature extends subscriber functionality by allowing you to alter the acknowledgement state of messages in bulk. For example, you can replay previously acknowledged messages or purge messages in bulk. In addition, you can copy the state of one subscription to another by using a combination of a Snapshot

state or one subscription to another by using seek in combination with a Snapshot.

👍 ↩ 🚩 upvoted 3 times

📄 👤 **Atnafu** 1 year, 11 months ago

B

The Seek feature extends subscriber functionality by allowing you to alter the acknowledgement state of messages in bulk. For example, you can replay previously acknowledged messages or purge messages in bulk. In addition, you can copy the state of one subscription to another by using seek in combination with a Snapshot.

<https://cloud.google.com/pubsub/docs/replay-overview>

👍 ↩ 🚩 upvoted 1 times

📄 👤 **TNT87** 2 years, 1 month ago

Selected Answer: B

Answer B

👍 ↩ 🚩 upvoted 1 times

📄 👤 **PhuocT** 2 years, 2 months ago

Selected Answer: B

should be B.

👍 ↩ 🚩 upvoted 1 times



Platform

> Home

> All Exams

> Examtopics PRO

> Training Courses



© 2024 ExamTopics