

🔗 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 6 DISCUSSION

Actual exam question from Google's Professional Data Engineer

Question #: 6

Topic #: 1

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

Your weather app queries a database every 15 minutes to get the current temperature. The frontend is powered by Google App Engine and server millions of users. How should you design the frontend to respond to a database failure?

- A. Issue a command to restart the database servers.
- B. Retry the query with exponential backoff, up to a cap of 15 minutes.
- C. Retry the query every second until it comes back online to minimize staleness of data.
- D. Reduce the query frequency to once every hour until the database comes back online.

[Show Suggested Answer](#)

by [deleted] at *March 15, 2020, 8:43 a.m.*

Comments

Type your comment...

[Submit](#)

👤 **Radhika7983** Highly Voted 4 years, 6 months ago

Correct answer is B. App engine create applications that use Cloud SQL database connections effectively. Below is what is written in google cloud documnetation.

If your application attempts to connect to the database and does not succeed, the database could be temporarily unavailable. In this case, sending too many simultaneous connection requests might waste additional database resources and increase the time needed to recover. Using exponential backoff prevents your application from sending an unresponsive number of connection requests when it can't connect to the database.

This retry only makes sense when first connecting, or when first grabbing a connection from the pool. If errors happen in the middle of a transaction, the application must do the retrying, and it must retry from the beginning of a transaction. So even if your pool is configured properly, the application might still see errors if connections are lost.




reference link is <https://cloud.google.com/sql/docs/mysql/manage-connections>

   upvoted 54 times

  **llamaste** Highly Voted  4 years, 9 months ago

<https://cloud.google.com/sql/docs/mysql/manage-connections#backoff>



   upvoted 12 times

  **willyunger** Most Recent  1 month, 2 weeks ago

Selected Answer: B

Exponential backoff avoids swamping the server. Higher rates may only make problem worse. Front-end should not have option to restart DB.



   upvoted 1 times

  **cqrm3n** 3 months, 3 weeks ago

Selected Answer: B

We should use exponential backoff because it reduces load on the failing database, optimizes retry timing and is the industry best practice. Exponential backoff is a retry strategy where the wait time between retry increase exponentially. By gradually increasing the retry interval, the system avoids wasting resources on immediate retries when the database is likely still down.

   upvoted 1 times

  **rtcpst** 7 months, 1 week ago

Selected Answer: B

Exponential backoff is a commonly used technique to handle temporary failures, such as a database server becoming temporarily unavailable. This approach retries the query, initially with a short delay and then with increasingly longer intervals between retries. Setting a cap of 15 minutes ensures that you don't excessively burden your system with constant retries.

Option C (retrying the query every second) can be too aggressive and may lead to excessive load on the server when it comes back online.

Option D (reducing the query frequency to once every hour) would result in significantly stale data and a poor user experience, which is generally not desirable for a weather app.

Option A (issuing a command to restart the database servers) is not a suitable action for a frontend component and might not address the issue effectively. Database server restarts should be managed as a part of the infrastructure and not initiated by the frontend.

   upvoted 3 times

  **samdhimal** 7 months, 1 week ago

correct answer -> Retry the query with exponential backoff, up to a cap of 15 minutes.

If your application attempts to connect to the database and does not succeed, the database could be temporarily unavailable. In this case, sending too many simultaneous connection requests might waste additional database resources and increase the time needed to recover. Using exponential backoff prevents your application from sending an unresponsive number of connection requests when it can't connect to the database.

   upvoted 2 times

  **samdhimal** 2 years, 3 months ago

Exponential backoff with a cap is a common technique used to handle temporary failures, such as database outages. In this approach, the frontend will retry the query with increasing intervals (e.g., 1s, 2s, 4s, 8s, etc.) up to a maximum interval (in this case, 15 minutes), this will help to avoid overwhelming the database servers with too many requests at once, and minimize the impact of the failure on the users.

Option A, is not recommended because it's not guaranteed that restarting the database servers will fix the problem, it could be a network or a configuration problem and it could cause more downtime.

Option C is not recommended because it could cause too many requests to be sent to the server, overwhelming the database and causing more downtime.

Option D is not recommended because reducing the query frequency too much would result in stale data, and users will not receive the most up-to-date information.

   upvoted 2 times

RT_G 1 year, 5 months ago

Selected Answer: B

Retries with exponential backoff seems like the most efficient option in this scenario

👍 ↩ 🚩 upvoted 1 times

rocky48 1 year, 6 months ago

Selected Answer: B

Correct answer is B

👍 ↩ 🚩 upvoted 1 times

gudguy1a 1 year, 8 months ago

Selected Answer: B

good answer, good answer @radhika7983.

👍 ↩ 🚩 upvoted 1 times

Datardp 1 year, 11 months ago

B is anser

👍 ↩ 🚩 upvoted 1 times

vaga1 1 year, 11 months ago

Selected Answer: B

I agree with the exponential backoff technique, even thoght I do not see why 15 minutes should be a desired choice.

👍 ↩ 🚩 upvoted 1 times

vaga1 1 year, 11 months ago

I guess that when u have failed after 15 minutes, your app must go through a serious review before being used again, since it is not able to provide the updated results as quickly as desired.

👍 ↩ 🚩 upvoted 1 times

yafsong 2 years, 4 months ago

Truncated exponential backoff is a standard error-handling strategy for network applications. In this approach, a client periodically retries a failed request with increasing delays between requests

👍 ↩ 🚩 upvoted 4 times

hiromi 2 years, 5 months ago

Selected Answer: B

B is right

👍 ↩ 🚩 upvoted 1 times

shiv14 3 years, 2 months ago

Selected Answer: B

According to the documentation

👍 ↩ 🚩 upvoted 1 times

deep_ROOT 3 years, 3 months ago

B is Correct; this question appeared in Cloud Architect exam also

👍 ↩ 🚩 upvoted 1 times

MaxNRG 3 years, 5 months ago

B,

backoff is a standard error handling strategy for network applications in which a client periodically retries a failed request with increasing delays between requests. Clients should use truncated exponential backoff for all requests to Cloud Storage that return HTTP 5xx and 429 response codes, including uploads and downloads of data or metadata.

👍 ↩ 🚩 upvoted 2 times

anji007 3 years, 6 months ago

Ans: B

👍 ↩ 🚩 upvoted 1 times

[Load full discussion...](#)

Platform

> Home

> All Exams

> Examtopics PRO

> Training Courses



© 2024 ExamTopics