

  **poplot321** 1 year, 5 months ago

I know C & E makes the most sense, but the Dockerfile is has a bug. If you don't copy requirements.txt from source, you have nothing to pip install. The line after FROM should be:

```
COPY requirements.txt requirements.txt
```

upvoted 4 times

  **Badri9898** 1 year, 8 months ago

The two actions that should be taken to optimize the Dockerfile for faster deployment times without adversely affecting the app's functionality are:

B. Remove dependencies from requirements.txt: The requirements.txt file should only contain necessary dependencies to reduce the number packages to be installed.

C. Use a slimmed-down base image like Alpine Linux: The Alpine Linux image is smaller than Ubuntu and has a smaller attack surface, which reduces the container's build time and image size.



Therefore, options A, D, and E are not correct as they do not directly address the issue of slow deployment times caused by a bloated Dockerfile.

upvoted 1 times

  **jrisl1991** 1 year, 2 months ago

But how do you know that requirements.txt has other dependencies? We don't have the file to confirm if there are unnecessary dependencies. Assuming that we only have the necessary ones, removing dependencies would break the deployment.

upvoted 1 times

  **alekonko** 1 year, 9 months ago

**Selected Answer: CE**

C: use smaller image decrease pull time

E: optimize build time using previous cache layer image. generate new layer only for a different app code and requirements

A: can't remove python

B: the developer choose right deps

D: changing instance type don't directly reduce deploy time

upvoted 2 times

  **omermahgoub** 1 year, 12 months ago

C & E

Using a slimmed-down base image like Alpine Linux can help reduce the size of your Docker image, which can lead to faster deployment times. Alpine Linux is a lightweight Linux distribution that is often used as a base image for Docker images because of its small size.


Additionally, copying the source code after installing the package dependencies can help reduce the image build time because the dependencies will only need to be installed once, rather than every time the source code is changed. This can lead to faster deployment times because the image build process will be faster.

upvoted 6 times

  **omermahgoub** 1 year, 12 months ago

It is not recommended to remove dependencies from the requirements.txt file or remove Python after running pip, as this could adversely affect the functionality of the application. Similarly, using larger machine types for your Google Container Engine node pools may not directly affect the deployment times of your application, as the deployment times are primarily dependent on the size and complexity of the Docker image being deployed.

upvoted 1 times

  **sfsdeniso** 2 years, 2 months ago

B & E

base image is already cached - so no improvement in build time

B is about removing unnecessary dependencies and not about all of them

i remember saw this question on google's site - BE are correct

upvoted 2 times

  **AzureDP900** 2 years, 2 months ago

CE and is perfect

upvoted 1 times

  **minmin2020** 2 years, 2 months ago

**Selected Answer: CE**

C. Use a slimmed-down base image like Alpine Linux

E. Copy the source after the package dependencies (Python and pip) are installed

upvoted 1 times

  **backhand** 2 years, 4 months ago

vote C, E

<https://cloud.google.com/architecture/best-practices-for-building-containers>

upvoted 1 times

  **MathMedrado** 2 years, 5 months ago

As far as I know, it is necessary to copy the requirements.txt first in order to run pip, it is not possible to run pip without first copying the requirements.txt. if they copy requirements.txt first then E would work.

upvoted 2 times

  **amxexam** 2 years, 7 months ago

**Selected Answer: CE**

By means of elimination A B, don't make sense. D is optimizing the package not script. Hence we are left with C & E

upvoted 3 times

  **potorange** 2 years, 7 months ago

**Selected Answer: CE**

C. Alpine is a lightweight Linux distro, with smaller image E. Pushing often changing files down the Dockerfile helps reducing image layers variations. Both help faster image pull operations

upvoted 1 times

#### Question #23

Topic 1

Your solution is producing performance bugs in production that you did not see in staging and test environments. You want to adjust your test and deployment procedures to avoid this problem in the future.

What should you do?

- A. Deploy fewer changes to production
- B. Deploy smaller changes to production
- C. Increase the load on your test and staging environments
- D. Deploy changes to a small subset of users before rolling out to production

  **ghitesh**  4 years, 11 months ago

Question Statement: You want to adjust your test and deployment procedures to avoid this problem in the future

So based on this, I think the option "C" is correct, since it is the only one talking about doing changes in the test environment.

upvoted 82 times

🗨️ 👤 **Septhethus** 6 months, 1 week ago

There is no indication given anywhere that the load is the problem or that the bugs are a result of load and not some other issue encountered when using a specific feature.

upvoted 2 times

🗨️ 👤 **alihabib** 2 weeks, 3 days ago

"Performance" bug is a result of Load

upvoted 1 times

🗨️ 👤 **VedaSW** 4 years, 2 months ago

C. Increase the load on your test and staging environments.

As you have pointed out in "Question Statement", I do not see C covering "deployment procedures". Test and Staging environment is more for testing, but not about deployment procedure to production.

So, the only option that covers test and deployment is D. (Yes, kind of unacceptable to have the users to do "testing", but we make it "ok" by calling it "canary deployment")

upvoted 20 times

🗨️ 👤 **francescogugliottagm** 1 year, 2 months ago

With canary deployment we expose the new version to a small portion of users. With this approach maybe we don't see performance bugs in the canary release, since we don't have the 100% of traffic on the canary. But when we migrate the 100% of traffic to the new release (previous canary) we can see performance bugs.

upvoted 9 times

🗨️ 👤 **Urban\_Life** 3 years ago

The answer is D

upvoted 9 times

🗨️ 👤 **RegisFTM** 2 years, 11 months ago

"Your solution is producing performance bugs in production..." - I don't see how "D" would help to detect performance bugs.

- "C" looks more adequate.

upvoted 19 times

🗨️ 👤 **Eroc** Highly Voted 🏆 5 years, 1 month ago

A wouldn't prevent the bugs, it would just avoid them. B would help with root-cause analysis because it'd be a smaller change to review. C would test the performance of the system at its peak processing rates, so this assumes the bugs in production only occur because of usage. D would allow you to test the new code against smaller user sets to see if it occurs then, and if it still does you know it is not because of more user responses. So it's a tossup between C and D, D would be the cheaper/quicker answer so I'd choose D first then C if it's because of usage.

upvoted 38 times

🗨️ 👤 **nitinz** 3 years, 9 months ago

D, canary rollout

upvoted 7 times

🗨️ 👤 **michael\_m** 2 years, 4 months ago

It has nothing to do with the "performance bugs"

upvoted 1 times

🗨️ 👤 **michael\_m** 2 years, 4 months ago

According to the question, [Your solution is producing "performance" bugs in production], so I think it is about the load. Plus canary test would not reproduce the bugs related to high load, I vote for C

upvoted 2 times

🗨️ 👤 **Sreekey** 4 years, 4 months ago

The question is about the performance of the existing Code that they did not detect in Test environments. This is not about new API releases. In order to test the performance they should increase the load in test environment and hence answer C.

upvoted 18 times

🗨️ 👤 **AzureDP900** 2 years, 2 months ago

C is the best

upvoted 2 times

🗨️ 👤 **deep316** Most Recent 1 week ago

**Selected Answer: C**

Bugs are related to performance. So you would need to perform performance testing thoroughly.  
upvoted 1 times

🗨️ 👤 **Daniaw** 1 week, 1 day ago

**Selected Answer: D**

C could be correct but this can help identify performance bottlenecks, but it might not fully replicate the complexity and unpredictability of real world production traffic.  
So D is the most correct one.  
upvoted 1 times

🗨️ 👤 **alihabib** 2 weeks, 3 days ago

**Selected Answer: C**

"Performance" bug, not a "Development" Bug, so it more aligns with C, to increase the load to determine performance related bugs  
upvoted 1 times

🗨️ 👤 **drinkwater** 3 weeks, 4 days ago

I think the right answer here is D. It applies a best practice of the release management like canary deployments.  
Why is not C; adding more load in staging and testing environments can help identify some performance issues, it is often impossible to replicate the exact conditions of a production environment.  
upvoted 1 times

🗨️ 👤 **Ekramy\_Elnaggar** 1 month, 1 week ago

**Selected Answer: C**

Guys, you need to focus on the KEYWORDS in any question, it will help you to determine the best answer. The keyword for this question is "Performance", it is very clear that the load test on stage was not planned correctly (i.e/ lower than it should), so the performance bugs didn't appear, but when it comes to production with much bigger load the issues appear.  
upvoted 1 times

🗨️ 👤 **nareshthumma** 1 month, 3 weeks ago

C & D both works but D make sense  
upvoted 1 times

🗨️ 👤 **dfizban** 2 months ago

**Selected Answer: D**

It's D  
upvoted 1 times

🗨️ 👤 **maxdanny** 3 months, 2 weeks ago

**Selected Answer: C**

C because The performance issues in production might not have been seen in staging or test environments because the load (number of user transactions, data volume, etc.) in those environments is not representative of the load in production. By increasing the load on your test and staging environments to match or exceed production levels, you can better simulate real-world conditions and catch performance issues before deployment  
upvoted 2 times

🗨️ 👤 **Hungdv** 4 months, 1 week ago

I will choose D.  
C: The question does not say the error caused by the load.  
upvoted 1 times

🗨️ 👤 **Haigk** 5 months, 3 weeks ago



**Selected Answer: D**

Without overthinking the wording, canary (and similar) deployment methodologies are often recommended in Google documentation, whereas increasing load in dev environments aren't. (My \$0.02...)  
upvoted 3 times

🗨️ 👤 **a2le** 6 months, 2 weeks ago

**Selected Answer: C**

Me too! I can't see how a "performance bug" might be mitigated via a canary deployment.  
However, I see that C doesn't cover the "deployment" part of the question, then I deduce that the question is ambiguously formulated.  
upvoted 1 times

  **Robert0** 6 months, 3 weeks ago

**Selected Answer: C**

Although all answers can be good practices, I think only option C address the problem described.  
upvoted 1 times

  **santoshchauhan** 9 months, 1 week ago



**Selected Answer: D**

D. Deploy changes to a small subset of users before rolling out to production.

This approach, known as canary releasing or canary deployment, involves rolling out changes to a small group of users before deploying then the entire user base. It is a very effective way to catch performance issues that might not have been apparent during testing.

C. Increase the load on your test and staging environments: This is definitely a good practice, as it can help simulate production-like conditions more closely. However, it may still not capture all real-world scenarios and user behaviors that can lead to performance issues.

upvoted 1 times

  **Patrick2708** 1 year ago

C looks good to me

upvoted 2 times

  **MahAli** 1 year ago

**Selected Answer: C**

Canary deployment is perfect to test new feature but to do stress testing, I do development for 25 years, when we want to resolve performance and scalability issues we do stress and load testing in pre prod environment, something you can't do by exposing the new feature to subset of users.

upvoted 6 times

Question #24

Topic 1

A small number of API requests to your microservices-based application take a very long time. You know that each request to the API can traverse many services.

You want to know which service takes the longest in those cases.

What should you do?



- A. Set timeouts on your application so that you can fail requests faster
- B. Send custom metrics for each of your requests to Stackdriver Monitoring
- C. Use Stackdriver Monitoring to look for insights that show when your API latencies are high

D. Instrument your application with Stackdriver Trace in order to break down the request latencies at each microservice

  **euclid** Highly Voted 4 years, 11 months ago



D is correct !

upvoted 23 times

  **tartar** 4 years, 4 months ago

D is ok

upvoted 9 times

  **nitinz** 3 years, 9 months ago

D, trace is just for latency testing.

upvoted 4 times

  **omermahgoub** Highly Voted 1 year, 12 months ago

D. Instrument your application with Stackdriver Trace in order to break down the request latencies at each microservice

Stackdriver Trace is a distributed tracing system that allows you to understand the relationships between requests and the various microservices that they touch as they pass through your application. By instrumenting your application with Stackdriver Trace, you can get a detailed breakdown of the latencies at each microservice, which can help you identify which service is taking the longest in those cases where a small number of API requests take a very long time.

Setting timeouts on your application or sending custom metrics to Stackdriver Monitoring may not provide the level of detail that you need to identify the specific service that is causing the latency issues. Looking for insights in Stackdriver Monitoring may also not provide the necessary level of detail, as it may not show the individual latencies at each microservice.

upvoted 8 times

  **Ekramy\_Elnaggar** Most Recent 1 month, 1 week ago

**Selected Answer: D**



1. Distributed Tracing: Stackdriver Trace (now called Cloud Trace) is specifically designed to analyze the performance of requests as they travel through multiple services in a microservices architecture. It helps you pinpoint exactly where latency is occurring.
2. Detailed Breakdown: Cloud Trace will provide a detailed breakdown of how long each microservice takes to process the request, allowing you to identify the bottleneck quickly.
3. Visualization: Cloud Trace provides visualizations like flame graphs and waterfall diagrams that make it easy to see the flow of the request and identify slow services.

upvoted 1 times

  **Hungdv** 4 months, 1 week ago

Choose D



upvoted 1 times

  **Robert0** 6 months, 3 weeks ago

**Selected Answer: D**

This question is not updated. It will be referred as Cloud Trace as part of Google Cloud Operation suite

upvoted 4 times

  **Robert0** 6 months, 3 weeks ago

This question is not updated. Stackdriver is now called "Google Cloud operations"

upvoted 2 times

  **eka\_nostra** 1 year, 4 months ago

**Selected Answer: D**

Trace is a signal that can be used to follow the program's data and flow, including the duration of the program's components.

upvoted 2 times

  **LaxmanTiwari** 1 year, 6 months ago

D should be correct, the headline in GC trace documentation says it all: "Cloud Trace is a distributed tracing system for Google Cloud that collects latency data from applications and displays it in near real-time in the Google Cloud Console."

upvoted 2 times

  **LaxmanTiwari** 1 year, 6 months ago

even got confused with the C ... after reading the conversation and reference doc D make sense to me.

upvoted 1 times

- 🗨️ 👤 **alekonko** 1 year, 9 months ago  
Selected Answer: D  
D is the correct answer  
upvoted 1 times
- 🗨️ 👤 **MestreCholas** 1 year, 9 months ago  
Why not C?  
upvoted 1 times
- 🗨️ 👤 **surajkrishnamurthy** 2 years ago  
Selected Answer: D  
D is the correct answer  
upvoted 1 times
- 🗨️ 👤 **AniketD** 2 years, 1 month ago  
Selected Answer: D  
Stackdriver Trace would trace the APIs and helps to identify the bottleneck  
upvoted 1 times
- 🗨️ 👤 **kchandank** 2 years, 1 month ago  
Selected Answer: D  
D is correct trace would report to the latency  
upvoted 1 times
- 🗨️ 👤 **Mahmoud\_E** 2 years, 1 month ago  
D is correct trace would report to the latency  
upvoted 1 times
- 🗨️ 👤 **minmin2020** 2 years, 2 months ago  
Selected Answer: D  
D. Instrument your application with Stackdriver Trace in order to break down the request latencies at each microservice  
upvoted 2 times
- 🗨️ 👤 **AzureDP900** 2 years, 2 months ago  
this is the best option to find more details  
upvoted 1 times
- 🗨️ 👤 **abirroy** 2 years, 3 months ago  
Selected Answer: D  
Instrument your application with Stackdriver Trace in order to break down the request latencies at each microservice  
upvoted 2 times
- 🗨️ 👤 **[Removed]** 2 years, 4 months ago  
Selected Answer: B  
D is correct !  
upvoted 1 times

During a high traffic portion of the day, one of your relational databases crashes, but the replica is never promoted to a master. You want to avoid this in the future.

What should you do?

- A. Use a different database
- B. Choose larger instances for your database
- C. Create snapshots of your database more regularly
- D. Implement routinely scheduled failovers of your databases

  **Narigdo** Highly Voted 5 years ago

Answer is D

upvoted 97 times

  **Jos** 5 years ago

Yep, +1 for D

upvoted 20 times

  **Eroc** Highly Voted 5 years, 1 month ago

@chiar, I agree the question is not clear. In GCP larger instances have larger number of CPUs, Memory and come with their own private network. So increasing the instance size would help prevent the need for failover during high traffic times. However, routinely scheduled failovers would allow the team to test the failover when it is not required. This would make sure it is working when it is required.

upvoted 45 times

  **Shariq** 5 years ago

exactly how do you know the optimal size. it will be a guess. answer should be D

upvoted 9 times

  **alihabib** Most Recent 2 weeks, 3 days ago

**Selected Answer: D**

Only a routine check, would indicate a best practice. Larger Instance doesn't guarantee a failover would be prevented

upvoted 1 times

  **Ekramy\_Elnaggar** 1 month, 1 week ago

**Selected Answer: D**

1. Proactive Testing: Regularly scheduled failovers proactively test your database's ability to recover and ensure the replica can successfully transition as the master. This helps identify and address any issues in the failover process before a real crisis occurs.

2. Confidence in Failover: By routinely testing failover, you gain confidence that your system can recover automatically in case of a primary database failure, minimizing downtime and data loss.

3. Improved Recovery Time: Regular failovers help optimize the recovery process, reducing the time it takes to switch to the replica.

upvoted 1 times

  **beagle\_Masato** 1 month, 2 weeks ago

**Selected Answer: D**

Answer is D

upvoted 1 times

  **james2033** 6 months, 3 weeks ago

**Selected Answer: D**

- \*\*A. Use a different database\*\*: Simply switching to a different database does not inherently solve the problem of failover and promotion mechanisms. The issue is more about the setup and management of failover strategies rather than the specific database technology used.

- \*\*B. Choose larger instances for your database\*\*: While using larger instances might improve performance and potentially reduce the risk of crash due to resource constraints, it does not address the failover mechanism. The key problem is the replica not being promoted, which larger instances alone won't fix.

- \*\*C. Create snapshots of your database more regularly\*\*: Regular snapshots are useful for backups and recovery but do not help with automatic failover and high availability. Snapshots do not ensure that a replica will be promoted to a master if the primary fails.

upvoted 2 times



🗨️ 👤 **huuthanhdlv** 7 months ago

I think the answer is B, as the most important thing is customer experience. We can NOT expect database fails as a normal event which have direct customer and business impacts (if current data base fail because of load then replica database may fail as well).

Of course we need to setup the failover process to work, but the more important task will be to increase database load first, thus I choose B.  
upvoted 1 times

🗨️ 👤 **Jen3** 9 months, 2 weeks ago

I think the answer is D because we can not assume the crash was due to load.  
upvoted 1 times

🗨️ 👤 **Jen3** 9 months, 2 weeks ago

Further the issue isn't that a crash occurred, the issue is the contingency that was in place didn't kick in. Hence D as my choice.  
upvoted 2 times

🗨️ 👤 **ashishdwi007** 11 months ago

**Selected Answer: D**

D is correct with given choice, because what if larger instance size is also failed. How ever question ignores logging and networking complete the best way is to use logging why the DB is crashed, failover is just a remedy to solve the issues at current, not solving the problem itself.  
upvoted 2 times

🗨️ 👤 **discuss24** 11 months, 2 weeks ago

It is important to identify key words, the issue is replica not being promoted when primary instance fails. Regular testing would identify issue  
upvoted 2 times

🗨️ 👤 **AWS\_Sam** 11 months, 2 weeks ago

The correct answer is D

The question is asking how to avoid the situation of not failing over. The answer is to test the failover procedure. The question does \*NOT\*ask about what happens in the future and whether the secondary node is large enough.  
upvoted 1 times

🗨️ 👤 **pakilodi** 1 year ago

**Selected Answer: D**

I would say D. Beacuse also B is correct, but you have a replica here, that is never promoted. So we need failover strategy.  
upvoted 1 times

🗨️ 👤 **owenshinobi** 1 year, 1 month ago

**Selected Answer: B**

My Answer is B

Why not D ?

I think

- Each day we cannot know. What times of the day are peak usage times? Implementing routinely scheduled failovers won't solve the problem  
- if Implement routinely scheduled failovers of your databases but replica database server is a same spec with main database server , replica database server will crashes by high traffic portion same a main database server

upvoted 2 times

🗨️ 👤 **Nora9** 1 year, 1 month ago

**Selected Answer: D**

Answer is D. Implement routinely scheduled failovers of your databases

This option is most aligned with addressing the issue. Routine failovers can help ensure that the failover process is working correctly and that system is resilient to crashes. It can be part of a disaster recovery plan, where you routinely test the failover to the replica to ensure that it can handle being promoted to a master if needed. For the above reasons, i believe D is correct.

upvoted 1 times

🗨️ 👤 **piiizu** 1 year, 2 months ago

Why not B?

Using a large instance during low traffic hours means incurring more cost than benefit except the instance is elastic. Therefore using a larger instance is not cost effective and the answer is D.

upvoted 1 times

🗨️ 👤 **Palan** 1 year, 4 months ago

Answer must be D i.e. Implement routinely scheduled failovers of your database because the main issue was the replica couldnt get created. its wise to ensure that failovers are checked on routine basis.

upvoted 1 times

🗨️ 👤 **Samley** 1 year, 4 months ago

Answer is D

upvoted 1 times

#### Question #26

Topic 1

Your organization requires that metrics from all applications be retained for 5 years for future analysis in possible legal proceedings. Which approach should you use?

- A. Grant the security team access to the logs in each Project
- B. Configure Stackdriver Monitoring for all Projects, and export to BigQuery
- C. Configure Stackdriver Monitoring for all Projects with the default retention policies
- D. Configure Stackdriver Monitoring for all Projects, and export to Google Cloud Storage

🗨️ 👤 **JoeShmoe** Highly Voted 🏆 5 years, 1 month ago

D is correct and best practice for long term log storage

upvoted 149 times

🗨️ 👤 **AndreaMa** 5 months, 3 weeks ago

same for me. The best approach for the long time log is to export it from monitoring to Cloud Storage Archival type

upvoted 1 times

🗨️ 👤 **AndreUanKenobi** 3 years, 9 months ago

+1. For archival purposes, Customer should use Cloud Storage. BigQuery is a datawarehouse, and could eventually import data from Cloud Storage if necessary.

upvoted 22 times

🗨️ 👤 **anjuagrawal** 2 years, 11 months ago

+1 Due to long term storage, cloud storage is better answer than BigQuery

upvoted 14 times

MeasService Highly Voted 4 years, 11 months ago

A and C can be quickly ruled out because none of them is solution for the requirements "retained for 5 years"

Between B and D, the different is where to store, BigQuery or Cloud Storage. Since the main concern is extended storing period, D (Correct Answer) is better choice, and the "retained for 5 years for future analysis" further qualifies it, for example, using Coldline storage class.

With regards of BigQuery, while it is also a low-cost storage, but the main purpose is for analysis. Also, logs stored in Cloud Storage is easy to transport to BigQuery or do query directly against the files saved in Cloud Storage if and whenever needed.

upvoted 70 times

Shyeom 4 years, 11 months ago

point : organization requires that metrics from all applications be retained for 5 years

upvoted 2 times

Shyeom 4 years, 11 months ago

I mean answer : D

upvoted 4 times

Cloudy\_Apple\_Juice 4 years, 2 months ago

If you have 2 viable solutions (B&D), then always chose the one that is cost optimised - I chose D

upvoted 5 times

jvale 2 years, 9 months ago

Bigquery long term storage cost: \$0.020 per GB

Cloud Storage archive cost: \$0,0012 per GB

Only if metrics need less than 10 GB (free service part on Bigquery) then the correct solution will be B... But all metrics for all application during more than 5 years... I think never will be the case :D

upvoted 11 times

Vika 3 years, 9 months ago

second that! like the way u explained..

upvoted 1 times

trainor 4 years ago

The question is about metrics, not logs. I'd go for B.

See <https://cloud.google.com/solutions/stackdriver-monitoring-metric-export>

upvoted 16 times

bnlcnd 3 years, 10 months ago

This is a good example. thanks.

But, we can easily change that implementation to dump the metrics to buckets to save lots of money. And, when talking about legal purpose, 1 hour interval may not be enough. You may have to keep more frequent metrics. So, only cold line or archive work for that purpose.

upvoted 5 times

alihabib Most Recent 2 weeks, 3 days ago

**Selected Answer: D**

Since the data is not actively queried, and cited as "possible" use qualifies it to be archived. Which means Cloud Storage

upvoted 1 times

drinkwater 3 weeks, 4 days ago

B is the right answer

why is not D, because while Google Cloud Storage can handle long-term storage, it is less efficient than BigQuery for analysis. Retrieving and querying metrics from Cloud Storage would require additional tools or steps, making it less suitable for the described use case.

upvoted 1 times

alpay 1 month ago

B in my opinion.

Take a look: [https://cloud.google.com/architecture/monitoring-metric-export#store\\_metrics](https://cloud.google.com/architecture/monitoring-metric-export#store_metrics)

BQ is also long-term storage with option to reduce cost of older data.

<https://cloud.google.com/bigquery/docs/best-practices-storage>

upvoted 1 times



  **Ekramy\_Elnaggar** 1 month, 1 week ago

**Selected Answer: B**

1. Long-term Retention: BigQuery is a data warehouse designed for long-term storage and analysis of large datasets. It's the ideal place to store metrics for 5 years to meet your organization's legal requirements.
2. Cost-Effective: BigQuery's storage pricing is very competitive, especially for long-term data retention.
3. Analysis and Reporting: BigQuery provides powerful tools for analyzing and querying data, making it easy to extract insights and generate reports from the stored metrics.
4. Integration: Stackdriver Monitoring (now Cloud Monitoring) can be easily configured to export metrics to BigQuery.

D is not correct as while Cloud Storage can store data for long periods, it's not optimized for querying and analyzing data like BigQuery.

upvoted 1 times

  **VedaSW** 3 months, 2 weeks ago

I go for B, as the question is about 5 years worth of data "for future analysis in possible legal proceedings", and the "future" can be next day, based on when the legal proceeding happen.



It is not about long term log storage.

Even the argument of "future" means 100 years later, the Cold Storage Archival still does not fulfill the "analysis" portion of the requirements.

You will need to move the data from Cold Storage to BigQuery for the analysis.


So the ideal answer should be combination of D and B, but we do not have such option, hence the answer can meet all requirements is B.

upvoted 1 times

  **Hungdv** 4 months, 1 week ago

D is answer. Monitoring has only 24 months retention.



upvoted 1 times

  **joecloud12** 4 months, 2 weeks ago

**Selected Answer: D**

for storing the logs you need cloud storage.



upvoted 1 times

  **monus** 4 months, 3 weeks ago

**Selected Answer: B**

B should be correct. How can Cloud Storage analyze the data?



upvoted 1 times

  **desertlotus1211** 4 months, 3 weeks ago

The statement is not about the actual analysis of the data, but 'where' to store the data for future analysis. Who know when that will be???

GCS is the best answer. When need be, it can be import into BQ

upvoted 1 times

  **desertlotus1211** 4 months, 3 weeks ago

Also it said retained for 5 years for future analysis... you don't store in BQ

upvoted 1 times

  **a2le** 6 months, 2 weeks ago

**Selected Answer: D**

I mean, "for possible future legal proceedings", I think that immutable storage that grants data integrity is the best option here, what's more, it also the cheapest one...



upvoted 1 times

  **james2033** 6 months, 3 weeks ago

**Selected Answer: D**

Google Cloud Storage for 5 years storing legally.

upvoted 1 times

  **arrase** 8 months, 1 week ago

**Selected Answer: D**

best practice for long term log storage

upvoted 1 times

  **tosinogunfile** 10 months, 2 weeks ago

**Selected Answer: B**

I think B is the right answer because transferring the data could be a basis for discrediting the data for legal use. Since Big Query can store the data and retain it with all the metadata intact, I will go for it.

upvoted 1 times

  **arielrahamim** 10 months, 3 weeks ago

**Selected Answer: D**

"organization requires that metrics from all applications be retained for 5 years" means cloud storage

upvoted 1 times

  **Teckexam** 11 months ago

**Selected Answer: D**

Cloud storage is appropriate for long term storage

upvoted 1 times

  **yas\_cloud** 11 months ago

This should be D. The tool lists option B as correct option which doesn't sound right. Bigquery export is fine for analysis but not for long term storage for 5 yrs.

upvoted 1 times

Question #27

Topic 1

Your company has decided to build a backup replica of their on-premises user authentication PostgreSQL database on Google Cloud Platform.

The database is 4



TB, and large updates are frequent. Replication requires private address space communication.

Which networking approach should you use?



- A. Google Cloud Dedicated Interconnect
- B. Google Cloud VPN connected to the data center network
- C. A NAT and TLS translation gateway installed on-premises
- D. A Google Compute Engine instance with a VPN server installed connected to the data center network

  **AWS56** Highly Voted 4 years, 11 months ago

A is the one  
upvoted 28 times

  **tartar** 4 years, 4 months ago

A is ok  
upvoted 8 times

  **nitinz** 3 years, 9 months ago

A, direct connect is private. VPN not enough for 4 TB with huge frequent changes.  
upvoted 3 times

  **amxexam** Highly Voted 3 years, 3 months ago

Let's go with option elimination  
A. Google Cloud Dedicated Interconnect  
>> Secured, fast connection, hence the choice. This will allow private connection from GCP to the data centre with a fast connection. Cost is mentioned in the requirement to eliminate this option.  
B. Google Cloud VPN connected to the data centre network  
>> We have to think about data flowing on the internet and the requirement talks about private connect. Also not sure how well you connect V with Data Center until you use the hybrid option. <https://cloud.google.com/network-connectivity/docs/vpn/concepts/overview> hence eliminate  
C. A NAT and TLS translation gateway installed on-premises  
>> This is a VM option to reach outside won't for this requirement hence eliminate  
D. A Google Compute Engine instance with a VPN server installed connected to the data centre network  
>> This is a slow option hence eliminate



Hence A  
upvoted 17 times

  **Ekramy\_Elnaggar** Most Recent 1 month, 1 week ago

**Selected Answer: A**

1. High Bandwidth and Reliability: Dedicated Interconnect provides a direct physical connection between your on-premises network and Google Cloud, offering high bandwidth and low latency. This is essential for replicating a 4TB database with frequent large updates.
2. Private Address Space: Dedicated Interconnect allows you to extend your private IP address space to Google Cloud, ensuring secure and private communication for database replication.
3. Security: Dedicated Interconnect offers a more secure connection compared to VPN, as traffic doesn't traverse the public internet.

Note: Question didn't mention anything about costs, so guys please stop overthinking and focus on the question key words.  
upvoted 2 times

  **Robert0** 6 months, 3 weeks ago

**Selected Answer: B**

Option B  
Interconnect is incredibly expensive and the usecase do not justify it.  
Properly configured, a VPN provides similar features. If the question included an unusually high SLA, I will go with Interconnect. If not, VPN is a great option.  
upvoted 4 times

  **tocsa** 6 months, 2 weeks ago

A simple VPN may not provide enough bandwidth for replication if the DB is busy. We know that the auth DB is 4TB, I'd say this must be a quite big company, possibly they can offer an interconnect? But it surely is expensive  
upvoted 1 times

🗨️ 👤 **sidiosidi** 7 months, 3 weeks ago

**Selected Answer: B**

I'll go for VPN.

First, the database is only for authentication and updates will be on this part, small portion of data needs to be replicated between on-premise and cloud. so no need for high bandwidth. the first migration will needs bandwidth but not toom much (5T) can be migrated using VPN.

VPN permits to use private networking and it's secure.

VPN not expensive as direct connect.

As architect you should also evaluate the cost over the requirement, at the end you need convince business with solution. Paying 5K will kick y out the project for such small requirement.

upvoted 3 times

🗨️ 👤 **Robert0** 6 months, 3 weeks ago

I think this fella hit the righ nail. Interconnect is incredibly expensive and the usecase do not justify it.

Properly configured, a VPN provides similar features. If the question included an inusally high SLA, I will go with Interconnect. If not, VPN is great option.

upvoted 1 times

🗨️ 👤 **Jen3** 9 months, 2 weeks ago

If you tried to sell me on Interconnect when all I needed was a VPN (meets bandwidth req, private address space, encryption of traffic possible) would reach out to AWS for a quote...

upvoted 1 times

🗨️ 👤 **lisabisa** 1 year, 2 months ago

GoogleVPN throughput is 3Gbps. It supports private IP connection and cheaper than Direct Connection.

Direct connect supports 8 \* 10Gbps or 2\*100Gbps. But too expensive for this

upvoted 3 times

🗨️ 👤 **eka\_nostra** 1 year, 4 months ago

**Selected Answer: A**

Connect to private space with high-speed bandwidth will go to A.

upvoted 2 times

🗨️ 👤 **mrhege** 1 year, 7 months ago

B: Dedicated Interconnect would be a major overkill here and a quite expensive one as well. Requirements mention private \_address space\_, private connection. Data over VPN is just as secure. Also there is no mention that a Google PoP would be available.

<https://cloud.google.com/network-connectivity/docs/how-to/choose-product>

upvoted 1 times

🗨️ 👤 **mohideenks** 2 years ago

**Selected Answer: A**

A is the correct answer

upvoted 1 times

🗨️ 👤 **Mahmoud\_E** 2 years, 1 month ago

**Selected Answer: A**

A is great but expensive for just a database DR but what can we do about that

upvoted 1 times

🗨️ 👤 **zr79** 2 years, 2 months ago

VPN is not private, it is public but encrypted. Also, VPN is not suitable for large updates that happen frequently

upvoted 1 times

🗨️ 👤 **AzureDP900** 2 years, 2 months ago

without any second thought A is right

upvoted 1 times

🗨️ 👤 **minmin2020** 2 years, 2 months ago

**Selected Answer: A**

A. Google Cloud Dedicated Interconnect - large updates and better security, however may not be the most cost effective choice

upvoted 1 times

🗑️ 👤 **[Removed]** 2 years, 7 months ago

**Selected Answer: A**

A is the one  
upvoted 1 times

🗑️ 👤 **Nirca** 2 years, 8 months ago

**Selected Answer: A**

Direct connect.  
upvoted 1 times

🗑️ 👤 **hibi6x** 3 years ago

Challenge me but this is answer B. I have 4TB DB, frequent update would be what ? 50% daily change means 2TB daily means ~25Mbps. Wi VPN I can easily achieved that. It is typical ingress to cloud free ....It would be madness to pay 5k montly only for Directo Connect...  
upvoted 5 times

🗑️ 👤 **AmitAr** 2 years, 7 months ago

Key points of quesiton -

- 1) Huge data and
  - 2) on-premises user authentication PostgreSQL - which means security - vpn uses public internet .. so B is not option.
- A - should be correct answer

upvoted 2 times

## Question #28

## Topic 1

Auditors visit your teams every 12 months and ask to review all the Google Cloud Identity and Access Management (Cloud IAM) policy changes in the previous 12 months. You want to streamline and expedite the analysis and audit process.

What should you do?

- A. Create custom Google Stackdriver alerts and send them to the auditor
- B. Enable Logging export to Google BigQuery and use ACLs and views to scope the data shared with the auditor
- C. Use cloud functions to transfer log entries to Google Cloud SQL and use ACLs and views to limit an auditor's view
- D. Enable Google Cloud Storage (GCS) log export to audit logs into a GCS bucket and delegate access to the bucket

🗑️ 👤 **ghitesh** **Highly Voted** 🏆 4 years, 11 months ago

B. [https://cloud.google.com/iam/docs/roles-audit-logging#scenario\\_external\\_auditors](https://cloud.google.com/iam/docs/roles-audit-logging#scenario_external_auditors)  
upvoted 98 times

🗑️ 👤 **rockstar9622** 4 years, 11 months ago

b) seems correct  
upvoted 3 times

🗑️ 👤 **anton\_royce** 4 years, 8 months ago

I agree. Answer B  
upvoted 5 times

🗑️ 👤 **MikeB19** 3 years, 3 months ago

The article references either gcs or bq. I think this q is referring to gcs  
upvoted 1 times

🗑️ 👤 **TheCloudBoy77** 3 years ago

B makes more sense after reading it. thx  
upvoted 4 times





  **jcmoranp** Highly Voted 5 years, 1 month ago



Think B is better. Export to Bigquery and restrict access to queries with ACLs to auditors  
upvoted 37 times

  **trainor** 4 years ago



I think D is better. B implies too much data manipulation to make it suitable for an audit.  
upvoted 4 times

  **tartar** 4 years, 4 months ago

D is ok.  
upvoted 7 times

  **tartar** 4 years, 4 months ago



Sorry, changed my view. B is the recommended practice  
upvoted 14 times

  **alii** 3 years, 11 months ago

don't change your view, D was right :)  
upvoted 4 times

  **RKS\_2021** 3 years, 5 months ago

B is correct  
upvoted 1 times

  **nitinz** 3 years, 9 months ago

D, rest all options are no good.  
upvoted 3 times

  **AmitAr** 2 years, 7 months ago

Please check the keywords in question -- "streamline and expedite" -- Bigquery is suitable not storage bucket. so it should be (B)  
upvoted 3 times

  **passnow** 5 years ago

I thought same as well. I would go with B  
upvoted 5 times

  **Ekramy\_Elnaggar** Most Recent 1 month, 1 week ago

**Selected Answer: B**

1. Comprehensive Audit Trail: Cloud Logging automatically captures audit logs for all Cloud IAM activity. Exporting these logs to BigQuery provides a centralized and comprehensive audit trail for analysis.

2. Powerful Analysis: BigQuery's analytical capabilities allow auditors to efficiently query and analyze IAM policy changes over the 12-month period. They can filter, aggregate, and generate reports to identify any anomalies or security concerns.



3. Granular Access Control: BigQuery's Access Control Lists (ACLs) and views enable you to precisely control which data the auditors can access. This ensures that they only see the information relevant to their audit without exposing sensitive data.

Note: While exporting logs to Cloud Storage is possible, it's less efficient for analysis compared to BigQuery.

upvoted 1 times

  **nareshthumma** 1 month, 3 weeks ago

Answer B  
upvoted 1 times

  **maxdanny** 3 months, 2 weeks ago

**Selected Answer: B**

Option B is the best approach. Enable Logging export to Google BigQuery and use ACLs and views to scope the data shared with the auditor. This method provides robust querying capabilities, ensures that historical IAM policy changes can be analyzed effectively, and allows you to control access securely.

upvoted 1 times

🗨️ 👤 **joecloud12** 4 months, 2 weeks ago

**Selected Answer: B**

b is correct because it is easier to implement compared to D  
upvoted 1 times

🗨️ 👤 **H\_S** 5 months, 1 week ago

**Selected Answer: D**

READ THIS, ACL is not available in BIG QUERY , therefore D. Enable Google Cloud Storage (GCS) log export to audit logs into a GCS bucket . delegate access to the bucket  
upvoted 4 times

🗨️ 👤 **Jen3** 9 months, 2 weeks ago

ACLs would provide year-round access to the data which is more privileges than necessary. Logs will need to be retained for a full year because hypothetically, January logs could be looked at in December. Cloud Storage offers signed URLs, and less expensive storage options.  
upvoted 1 times

🗨️ 👤 **lisabisa** 10 months ago

Both B and D are ok.

Using cloud storage requires additional setup for auditors, pulling data to BQ.  
Using BQ would satisfy "streamline and expedite the analysis and audit process"  
upvoted 1 times

🗨️ 👤 **Teckexam** 11 months ago

**Selected Answer: B**

Based on google documentation B is the correct answer.  
[https://cloud.google.com/iam/docs/job-functions/auditing#scenario\\_external\\_auditors](https://cloud.google.com/iam/docs/job-functions/auditing#scenario_external_auditors)  
Dashboard is available in BigQuery to review historic logs and in case anomaly is found elevated access is provided. Access is revoked after audit activities are done.  
upvoted 4 times

🗨️ 👤 **kip21** 11 months, 1 week ago

D - Correct  
B - his option requires additional work to set up the ACLs and views to limit an auditor's view of the data. This could be time-consuming and complex to implement. Furthermore, BigQuery may not be the ideal tool for auditors who are only interested in reviewing Cloud IAM policy changes.  
upvoted 2 times

🗨️ 👤 **CloudDom** 1 year, 1 month ago

**Selected Answer: B**

That's the only logical one also Bard is confirming this one  
upvoted 1 times

🗨️ 👤 **thewalker** 1 year, 1 month ago

D  
I will not go with B, as the requirement is once for 12 months. Push the data in Coldline for 12 months and retrieve it during audit is enough. S costs.  
upvoted 3 times

🗨️ 👤 **thewalker** 1 year, 1 month ago

Coldline / Archive  
upvoted 2 times

🗨️ 👤 **hogtrough** 11 months, 2 weeks ago

Streamline and expedite analysis is the goal. Costs are never brought up.  
upvoted 2 times

🗨️ 👤 **krisek** 1 year, 1 month ago

**Selected Answer: B**

Reading from Cloud Storage raw audit logs (without filtering applied) is everything but streamlined. Imagine the auditor fetching all audit logs, then write some script to analyze them...  
upvoted 2 times

🗨️ 👤 **Prakzz** 1 year, 2 months ago

**Selected Answer: D**

B talks about ACL in BigQuery and ACL is not associated with BigQuery but with GCS.  
upvoted 4 times

🗨️ 👤 **AdityaGupta** 1 year, 2 months ago

**Selected Answer: B**

You want to streamline and expedite the analysis and audit process.

Big Query, as the data retention is mentioned, and data is related to Cloud IAM policy changes, it is safe to assume long term retention with annual audit.

upvoted 1 times

🗨️ 👤 **TopTalk** 1 year, 2 months ago

**Selected Answer: B**

""To comply with this requirement, a dashboard is available that provides access to the historic logs stored in BigQuery, and on request, to the Cloud Logging Admin Activity logs.

The organization creates a Google group for these external auditors and adds the current auditor to the group. This group is monitored and is typically granted access to the dashboard application.

During normal access, the auditors' Google group is only granted access to view the historic logs stored in BigQuery. If any anomalies are discovered, the group is granted permission to view the actual Cloud Logging Admin Activity logs via the dashboard's elevated access mode. the end of each audit period, the group's access is then revoked."

[https://cloud.google.com/iam/docs/job-functions/auditing#scenario\\_external\\_auditors](https://cloud.google.com/iam/docs/job-functions/auditing#scenario_external_auditors)

upvoted 1 times

#### Question #29

Topic 1

You are designing a large distributed application with 30 microservices. Each of your distributed microservices needs to connect to a database back-end. You want to store the credentials securely.  
Where should you store the credentials?



- A. In the source code
- B. In an environment variable
- C. In a secret management system

D. In a config file that has restricted access through ACLs

  **Eroc** Highly Voted 5 years, 1 month ago



Google Secret Management was designed explicitly for this purpose.

upvoted 33 times

  **tartar** 4 years, 4 months ago

C is ok

upvoted 7 times

  **nitinz** 3 years, 9 months ago

C, microservices = GKE = Kubernetes = secrets.

upvoted 6 times

  **shandy** Highly Voted 5 years ago

C is the answer, since key management systems generate, use, rotate, encrypt, and destroy cryptographic keys and manage permissions to those keys.

A is incorrect because storing credentials in source code and source control is discoverable, in plain text, by anyone with access to the source code. This also introduces the requirement to update code and do a deployment each time the credentials are rotated. B is not correct because consistently populating environment variables would require the credentials to be available, in plain text, when the session is started. D is incorrect because instead of managing access to the config file and updating manually as keys are rotated, it would be better to leverage a key management system. Additionally, there is increased risk if the config file contains the credentials in plain text.

upvoted 13 times

  **Ekramy\_Elnaggar** Most Recent 1 month, 1 week ago

**Selected Answer: C**

1. Centralized and Secure Storage: Secret management systems like HashiCorp Vault, AWS Secrets Manager, or Google Cloud Secret Manager provide a centralized and secure location to store sensitive credentials. This ensures that database credentials are not scattered across multiple microservices or configuration files.

2. Access Control: Secret management systems offer fine-grained access control, allowing you to restrict access to secrets based on roles and permissions. This ensures that only authorized microservices and users can access the database credentials.

3. Rotation and Auditing: These systems often provide features for automatic secret rotation and auditing, which helps improve security and compliance.

4. Integration: Secret management systems can integrate with your deployment pipelines and orchestration tools, making it easier to manage secrets throughout the application lifecycle.

upvoted 1 times

  **ashishdwi007** 11 months ago

**Selected Answer: C**

Other options are not best practices.



upvoted 1 times

  **Teckexam** 11 months ago

**Selected Answer: C**

Need to use key management system for this usecase since other options are not secure.

upvoted 1 times

  **hiromi** 1 year, 8 months ago

**Selected Answer: C**

C is correct

upvoted 1 times