[Home](https://cognizant.udemy.com/)

[Google Certified Associate Cloud Engineer Practice Tests…](https://cognizant.udemy.com/course/gcp-ace-practice-tests-latest/)

[Add to learning path](https://cognizant.udemy.com/course/gcp-ace-practice-tests-latest/learn/)

[Favorite this course](javascript:void(0))

[Archive this course](javascript:void(0))

[Unenroll from course](javascript:void(0))

Premium Practice Test 1 (Updated October 2020) - Results

Return to review

Created with Highcharts 6.1.0

Attempt 1

All questions

[All questions](javascript:void(0))

[Correct](javascript:void(0))

[Incorrect](javascript:void(0))

[Skipped](javascript:void(0))

[Marked for review](javascript:void(0))

Top of Form

Question 1: **Correct**

**You want to migrate a mission-critical application from the on-premises data centre to Google Cloud Platform. Due to the mission-critical nature of the application, you want the application to scale up and scale down efficiently based on demand, but you do not wish to scale down any lower than 3 instances to ensure the application always has enough resources to handle sudden bursts in traffic. How should you configure the scaling to meet this requirement?**

​

Automatic Scaling with min\_idle\_instances set to 3.

(Correct)

​

Manual Scaling with 3 instances.

​

Basic Scaling with min\_instances set to 3.

​

Basic Scaling with max\_instances set to 3.

**Explanation**

Manual Scaling with 3 instances. **is not right.**

Manual scaling uses resident instances that continuously run the specified number of instances regardless of the load level. This allows tasks such as complex initializations and applications that rely on the state of the memory over time. This does not autoscale based on the request rate so doesn't fit our requirements.

Ref: https://cloud.google.com/appengine/docs/standard/python/how-instances-are-managed

Basic Scaling with min\_instances set to 3. **is not right.**

Basic scaling creates dynamic instances when your application receives requests. Each instance will be shut down when the app becomes idle. Basic scaling is ideal for work that is intermittent or driven by user activity. In absence of any load, the App engine may shut down all instances so it is not suitable for our requirement of "at least 3 instances at all times".

Ref: https://cloud.google.com/appengine/docs/standard/python/how-instances-are-managed

Basic Scaling with max\_instances set to 3. **is not right.**

Basic scaling creates dynamic instances when your application receives requests. Each instance will be shut down when the app becomes idle. Basic scaling is ideal for work that is intermittent or driven by user activity. In absence of any load, the App engine may shut down all instances so it is not suitable for our requirement of "at least 3 instances at all times".

Ref: https://cloud.google.com/appengine/docs/standard/python/how-instances-are-managed

Automatic Scaling with min\_idle\_instances set to 3. **is the right answer.**

Automatic scaling creates dynamic instances based on request rate, response latencies, and other application metrics. However, if you specify the number of minimum idle instances, that specified number of instances run as resident instances while any additional instances are dynamic.

Ref: https://cloud.google.com/appengine/docs/standard/python/how-instances-are-managed

Bottom of Form

Top of Form

Question 2: **Incorrect**

**You developed a web application that lets users upload and share images. You deployed this application in Google Compute Engine and you have configured Stackdriver Logging. Your application sometimes times out while uploading large images, and your application generates relevant error log entries that are ingested to Stackdriver Logging. You would now like to create alerts based on these metrics. You intend to add more compute resources manually when the number of failures exceeds a threshold. What should you do in order to alert based on these metrics with minimal effort?**

​

Add the Stackdriver monitoring and logging agent to the instances running the code.

(Incorrect)

​

Create a custom monitoring metric in code, edit the application code to set the metric value when needed, create an alert in Stackdriver based on the new metric.

​

In Stackdriver Logging, create a custom monitoring metric from log data and create an alert in Stackdriver based on the new metric.

(Correct)

​

In Stackdriver logging, create a new logging metric with the required filters, edit the application code to set the metric value when needed, and create an alert in Stackdriver based on the new metric.

**Explanation**

In Stackdriver logging, create a new logging metric with the required filters, edit the application code to set the metric value when needed, and create an alert in Stackdriver based on the new metric. **is not right.**

You don't need to edit the application code to send the metric values. The application already pushes error logs whenever the application times out. Since you already have the required entries in the Stackdriver logs, you don't need to edit the application code to send the metric values. You just need to create metrics from log data.

Ref: https://cloud.google.com/logging

Create a custom monitoring metric in code, edit the application code to set the metric value when needed, create an alert in Stackdriver based on the new metric. **is not right.**

You don't create a custom monitoring metric in code. Stackdriver Logging allows you to easily create metrics from log data. Since the application already pushes error logs to Stackdriver Logging, we just need to create metrics from log data in Stackdriver Logging.

Ref: https://cloud.google.com/logging

Add the Stackdriver monitoring and logging agent to the instances running the code. **is not right.**

The Stackdriver Monitoring agent gathers system and application metrics from your VM instances and sends them to Monitoring. In order to make use of this approach, you need application metrics but our application doesn't generate metrics. It just logs errors whenever the upload times out and these are then ingested to Stackdriver logging. We can update our application to enable custom metrics for these scenarios, but that is a lot more work than creating metrics from log data in Stackdriver Logging

Ref: https://cloud.google.com/logging

In Stackdriver Logging, create a custom monitoring metric from log data and create an alert in Stackdriver based on the new metric. **is the right answer.**

Our application adds entries to error logs whenever the application times out during image upload and these logs are ingested to Stackdriver Logging. Since we already have the required data in logs, we just need to create metrics from this log data in Stackdriver Logging. And we can then set up an alert based on this metric. We can trigger an alert if the number of occurrences of the relevant error message is greater than a predefined value. Based on the alert, you can manually add more compute resources.

Ref: https://cloud.google.com/logging

Bottom of Form

Top of Form

Question 3: **Incorrect**

**You deployed a Flask python application to GCP App Engine Standard service in US-Central region. Most of your customers are based in Japan and are experiencing slowness due to the latency. How can you transfer the application from US-Central region to Asia-Northeast1 region to minimize latency?**

​

Change the default region property setting in the existing GCP project to asia-northeast1.

​

Change the region property setting in the existing App Engine application from us-central to asia-northeast1.

(Incorrect)

​

Create a second App Engine application in the existing GCP project and specify asia-northeast1 as the region to serve your application.

​

Create a new GCP project and create an App Engine application inside this new project. Specify asia-northeast1 as the region to serve your application.

(Correct)

**Explanation**

Change the default region property setting in the existing GCP project to asia-northeast1. **is not right.**

App Engine is regional, which means the infrastructure that runs your apps is located in a specific region and is managed by Google to be redundantly available across all the zones within that region. You cannot change an app's region after you set it.

Ref: https://cloud.google.com/appengine/docs/locations

Change the region property setting in the existing App Engine application from us-central to asia-northeast1. **is not right.**

App Engine is regional, which means the infrastructure that runs your apps is located in a specific region and is managed by Google to be redundantly available across all the zones within that region. You cannot change an app's region after you set it.

Ref: https://cloud.google.com/appengine/docs/locations

Create a second App Engine application in the existing GCP project and specify asia-northeast1 as the region to serve your application. **is not right.**

App Engine is regional and you cannot change an app's region after you set it. You can deploy additional services in the App Engine but they will all be targeted to the same region.

Ref: https://cloud.google.com/appengine/docs/locations

Create a new GCP project and create an App Engine application inside this new project. Specify asia-northeast1 as the region to serve your application. **is the right answer.**

App Engine is regional and you cannot change an app's region after you set it. Therefore, the only way to have an app run in another region is by creating a new project and targeting the app engine to run in the required region (asia-northeast1 in our case).

Ref: https://cloud.google.com/appengine/docs/locations

Bottom of Form

Top of Form

Question 4: **Correct**

**Your company has deployed a wide range of application across several Google Cloud projects in the organization. You are a security engineer within the Cloud Security team, and an apprentice has recently joined your team. To gain a better understanding of your company’s Google cloud estate, the apprentice has asked you to provide them access which lets them have detailed visibility of all projects in the organization. Your manager has approved the request but has asked you to ensure the access does not let them edit/write access to any resources. Which IAM roles should you assign to the apprentice?**

​

Grant roles/resourcemanager.organizationViewer and roles/viewer.

(Correct)

​

Grant roles/resourcemanager.organizationAdmin and roles/browser.

​

Grant roles/resourcemanager.organizationViewer and roles/owner.

​

Grant roles/owner, roles/networkmanagement.admin.

**Explanation**

The security team needs **detailed visibility** of all GCP projects in the organization so they should be able to **view all the projects in the organization** as well as view **all resources within these projects**.

Grant roles/resourcemanager.organizationViewer and roles/owner. **is not right.**

roles/resourcemanager.organizationViewer role provides permissions to see the organization in the Cloud Console without having access to view all resources in the organization.

roles/owner provides permissions to manage roles and permissions for a project and all resources within the project and set up billing for a project.

Neither of the roles give the security team visibility of the projects in the organization.

Ref: https://cloud.google.com/resource-manager/docs/access-control-org

Ref: https://cloud.google.com/iam/docs/understanding-roles#organization-policy-roles

Grant roles/resourcemanager.organizationAdmin and roles/browser. **is not right.**

roles/resourcemanager.organizationAdmin provides access to administer all resources belonging to the organization. This doesn't follow the least privilege principle. Our security team needs detailed visibility i.e. read-only access but should not be able to administer resources..

Ref: https://cloud.google.com/iam/docs/understanding-roles#organization-policy-roles

Grant roles/owner, roles/networkmanagement.admin. **is not right.**

roles/owner provides permissions to manage roles and permissions for a project and all resources within the project and set up billing for a project.

roles/networkmanagement.admin provides full access to Cloud Network Management resources.

Neither of the roles give the security team visibility of the projects in the organization.

Ref: https://cloud.google.com/resource-manager/docs/access-control-org

Ref: https://cloud.google.com/iam/docs/understanding-roles#organization-policy-roles

Grant roles/resourcemanager.organizationViewer and roles/viewer. **is the right answer.**

roles/viewer provides permissions to view existing resources or data.

roles/resourcemanager.organizationViewer provides access to view an organization.

With the two roles, the security team can view the organization including all the projects and folders; as well as view all the resources within the projects.

Ref: https://cloud.google.com/resource-manager/docs/access-control-org

Ref: https://cloud.google.com/iam/docs/understanding-roles#organization-policy-roles

Bottom of Form

Top of Form

Question 5: **Correct**

**You have two compute instances in the same VPC but in different regions. You can SSH from one instance to another instance using their internal IP address but not their external IP address. What could be the reason for SSH failing on external IP address?**

​

The compute instances have a static IP for their external IP.

​

The compute instances are not using the right cross-region SSH IAM permissions

​

The combination of compute instance network tags and VPC firewall rules only allow SSH from the subnets IP range.

(Correct)

​

The external IP address is disabled.

**Explanation**

The compute instances have a static IP for their external IP. **is not right.**

Not having a static IP is not a reason for failed SSH connections. When the firewall rules are set up correctly, SSH works fine on compute instances having ephemeral IP Address.

The external IP address is disabled. **is not right.**

Our question states SSH doesn't work on external IP addresses so it is safe to assume they already have an external IP. Therefore, this option is not correct.

The compute instances are not using the right cross-region SSH IAM permissions. **is not right.**

There is no such thing as cross region SSH IAM permissions.

The combination of compute instance network tags and VPC firewall rules only allow SSH from the subnets IP range. **is the right answer.**

The combination of compute instance network tags and VPC firewall rules can certainly result in SSH traffic being allowed from only subnets IP range. The firewall rule can be configured to allow SSH traffic from just the VPC range e.g. 10.0.0.0/8. In this scenario, all SSH traffic from within the VPC is accepted but external SSH traffic is blocked.

Ref: https://cloud.google.com/vpc/docs/using-firewalls

Bottom of Form

Top of Form

Question 6: **Correct**

**Your company stores sensitive user information (PII) in three multi-regional buckets in US, Europe and Asia. All three buckets have access logging enabled on them. The compliance team has received reports of fraudulent activity and has begun investigating a customer care representative. It believes the specific individual may have accessed some objects they are not authorized to and may have added labels to some files in the buckets to enable favourable discounts for their friends. The compliance team has asked you to provide them with a report of activities for this customer service representative on all three buckets. How can you do this efficiently?**

​

Using the GCP Console, filter the Stackdriver log to view the information.

(Correct)

​

Create a trace in Stackdriver to view the information.

​

View the bucket in the Storage section of the GCP Console.

​

Using the GCP Console, filter the Activity log to view the information.

**Explanation**

Our requirements are - **sensitive** data, **verify** access, **fewest** possible steps.

Using the GCP Console, filter the Activity log to view the information. **is not right.**

Since data access logging is enabled, you can see relevant log entries in both activity Logs as well as stack driver logs. However, verifying what has been viewed/updated is not straightforward in activity logs. Activity logs display a list of all actions and you can restrict this down to a user and further filter by specifying Data access as the Activity types and GCS Bucket as the Resource type. But that is the extent of the filter functionality in Activity logs. It is not possible to restrict the activity logs to just the three buckets that we are interested in. Secondly, it is not possible to restrict the activity logs to just the gets and updates. So we'd have to go through the full list to identify activities of interest before verifying them which is a manual process and can be time taking depending on the number of activities in the list.

Ref: https://cloud.google.com/storage/docs/audit-logs

View the bucket in the Storage section of the GCP Console. **is not right.**

The bucket page in the GCP console does not show the logs.

Create a trace in Stackdriver to view the information. **is not right.**

Stackdriver trace is not supported on google cloud. Stackdriver Trace runs on Linux in the following environments: Compute Engine, Google Kubernetes Engine (GKE), App Engine flexible environment, App Engine standard environment.

Ref: https://cloud.google.com/trace/docs/overview

Using the GCP Console, filter the Stackdriver log to view the information. **is the right answer.**

Data access logs is already enabled, so we already record all API calls that read the configuration or metadata of resources, as well as user-driven API calls that create, modify, or read user-provided resource data. Data Access audit logs do not record the data-access operations on resources that are publicly shared (available to All Users or All Authenticated Users) or that can be accessed without logging into Google Cloud.

Since we are dealing with **sensitive data**, it is safe to assume that these buckets are not publicly shared and therefore enabling Data access logging logs all data-access operations on resources. These logs are sent to Stackdriver where they can be viewed by applying a suitable filter.

Unlike activity logs, retrieving the required information to verify is easier and quicker through Stackdriver as you can apply filters such as

resource.type="gcs\_bucket"

(resource.labels.bucket\_name="gcp-ace-lab-255520" OR resource.labels.bucket\_name="gcp-ace-lab-255521" OR resource.labels.bucket\_name="gcp-ace-lab-255522")

(protoPayload.methodName="storage.objects.get" OR protoPayload.methodName="storage.objects.update")

protoPayload.authenticationInfo.principalEmail="test.gcp.labs.user@gmail.com"

and query just the gets and updates, for specific buckets for a specific user. This involves fewer steps and is more efficient.

Data access logging is not enabled by default and needs to be enabled explicitly. The screenshot below shows a screenshot for enabling the data access logging for Google Cloud Storage.

Bottom of Form

Top of Form

Question 7: **Correct**

**A mission-critical application running on a Managed Instance Group (MIG) in Google Cloud has been having scaling issues. Although the scaling works, it is not quick enough, and users experience slow response times. The solution architect has recommended moving to GKE to achieve faster scaling and optimize machine resource utilization. Your colleague containerized the application and provided you with a Dockerfile. You now need to deploy this in a GKE cluster. How should you do it?**

​

Create a docker image from the Dockerfile and upload it to Cloud Storage. Create a Deployment YAML file to point to that image. Use kubectl to create the deployment with that file.

​

Use *gcloud app deploy <dockerfilename>*.

​

Create a docker image from the Dockerfile and upload it to Container Registry. Create a Deployment YAML file to point to that image. Use kubectl to create the deployment with that file.

(Correct)

​

Use *kubectl app deploy <dockerfilename>*.

**Explanation**

Use *kubectl app deploy <dockerfilename>*. **is not right.**

kubectl does not accept *app* as a verb. Kubectl can deploy a configuration file using kubectl deploy.

Ref: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#apply

Use *gcloud app deploy <dockerfilename>.* **is not right.**

*gcloud app deploy* - Deploys the local code and/or configuration of your app to App Engine. gcloud app deploy accepts a flag --image-url which is the docker image but it can't directly use a docker file.

Ref: https://cloud.google.com/sdk/gcloud/reference/app/deploy

Create a docker image from the Dockerfile and upload it to Cloud Storage. Create a Deployment YAML file to point to that image. Use kubectl to create the deployment with that file. **is not right.**

You can not upload a docker image to cloud storage. They can only be pushed to a Container Registry (e.g. GCR, Dockerhub etc.)

Ref: https://cloud.google.com/container-registry/docs/pushing-and-pulling

Create a docker image from the Dockerfile and upload it to Container Registry. Create a Deployment YAML file to point to that image. Use kubectl to create the deployment with that file. **is the right answer.**

Once you have a docker image, you can push it to the container register. You can then create a deployment YAML file pointing to this image and use *kubectl apply -f <deployment YAML filename>* to deploy this to the Kubernetes cluster. This assumes you already have a Kubernetes cluster and you gcloud environment is set up to talk to this container by executing *gcloud container clusters get-credentials <cluster name> --zone=<container\_zone>*

Ref: https://cloud.google.com/container-registry/docs/pushing-and-pulling

Ref: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#apply

Ref: https://cloud.google.com/sdk/gcloud/reference/container/clusters/get-credentials

Bottom of Form

Top of Form

Question 8: **Correct**

**31. You created an update for your application on App Engine. You want to deploy the update without impacting your users. You want to be able to roll back as quickly as possible if it fails. What should you do?**

​

Deploy the update as the same version that is currently running. You are confident the update works so you don't plan for a rollback strategy.

​

Deploy the update as the same version that is currently running. If the update fails, redeploy your older version using the same version identifier.

​

Notify your users of an upcoming maintenance window and ask them not to use your application during this window. Deploy the update in that maintenance window.

​

Deploy the update as a new version. Migrate traffic from the current version to the new version. If it fails, migrate the traffic back to your older version.

(Correct)

**Explanation**

Deploy the update as the same version that is currently running. You are confident the update works so you don't plan for a rollback strategy. **is not right.**

Irrespective of the level of confidence, you should always prepare a rollback strategy as things can go wrong for reasons out of our control.

Deploy the update as the same version that is currently running. If the update fails, redeploy your older version using the same version identifier. **is not right.**

While this can be done, the rollback process is not quick. Your application is unresponsive until you have redeployed the older version which can take quite a bit of time depending on how it is set up.

Notify your users of an upcoming maintenance window and ask them not to use your application during this window. Deploy the update in that maintenance window. **is not right.**

Our requirement is to deploy the update without impacting our users but by asking them to not use the application during the maintenance window, you are impacting all users.

Deploy the update as a new version. Migrate traffic from the current version to the new version. If it fails, migrate the traffic back to your older version. **is the right answer.**

This option enables you to deploy a new version and send all traffic to the new version. If you realize your updated application is not working, the rollback is as simple as marking your older version as default. This can all be done in the GCP console with a few clicks.

Ref: https://cloud.google.com/appengine/docs/admin-api/deploying-apps

Bottom of Form

Top of Form

Question 9: **Correct**

**You deployed a number of services to Google App Engine Standard. The services are designed as microservices with several interdependencies between them. Most services have few version upgrades but some key services have over 20 version upgrades. You identified an issue with the service pt-createOrder and deployed a new version *v3* for this service. You are confident this works and want this new version to receive all traffic for the service. You want to minimize effort and ensure the availability of service. What should you do?**

​

Execute *gcloud app versions stop v2 --service="pt-createOrder" and gcloud app versions start v3 --service="pt-createOrder"*

​

Execute *gcloud app versions migrate v3*

​

Execute *gcloud app versions migrate v3 --service="pt-createOrder"*

(Correct)

​

Execute *gcloud app versions stop v2 and gcloud app versions start v3*

**Explanation**

Execute *gcloud app versions migrate v3*. **is not right.**

gcloud app versions migrate v3 migrates all services to version v3. In our scenario, we have multiple services with each service potentially being on a different version. We don't want to migrate all services to v3, instead, we only want to migrate the pt-createOrder service to v3. Ref: https://cloud.google.com/sdk/gcloud/reference/app/versions/migrate

Execute *gcloud app versions stop v2 --service="pt-createOrder" and gcloud app versions start v3 --service="pt-createOrder"*. **is not right.**

Stopping version v2 and starting version v3 for pt-createOrder service would result in v3 receiving all traffic for pt-createOrder. While this is the intended outcome, stopping version v2 before starting version v3 results in service being unavailable until v3 is ready to receive traffic. As we want to "ensure availability", this option is not suitable.

Ref: https://cloud.google.com/sdk/gcloud/reference/app/versions/migrate

Execute *gcloud app versions stop v2 and gcloud app versions start v3*. **is not right.**

Stopping version v2 and starting version v3 would result in migrating all services to version v3 which is undesirable. We don't want to migrate all services to v3, instead, we only want to migrate the pt-createOrder service to v3. Moreover, stopping version v2 before starting version v3 results in service being unavailable until v3 is ready to receive traffic. As we want to "ensure availability", this option is not suitable.

Ref: https://cloud.google.com/sdk/gcloud/reference/app/versions/migrate

Execute *gcloud app versions migrate v3 --service="pt-createOrder"*. **is the right answer.**

This command correctly migrates the service pt-createOrder to use version 3 and produces the intended outcome while minimizing effort and ensuring the availability of service.

Ref: https://cloud.google.com/sdk/gcloud/reference/app/versions/migrate

Bottom of Form

Top of Form

Question 10: **Correct**

**You want to deploy a cost-sensitive application to Google Cloud Compute Engine. You want the application to be up at all times, but because of the cost-sensitive nature of the application, you only want to run the application in a single VM instance. How should you configure the managed instance group?**

​

Set autoscaling to On, set the minimum number of instances to 1, and then set the maximum number of instances to 1.

(Correct)

​

Set autoscaling to On, set the minimum number of instances to 1, and then set the maximum number of instances to 2.

​

Set autoscaling to Off, set the minimum number of instances to 1, and then set the maximum number of instances to 2.

​

Set autoscaling to Off, set the minimum number of instances to 1, and then set the maximum number of instances to 1.

**Explanation**

Requirements

1. Since we need the application running at all times, we need a **minimum 1 instance**.

2. Only a single instance of the VM should run, we need a **maximum 1 instance**.

3. We want the application running at all times. If the VM crashes due to any underlying hardware failure, we want another instance to be added to MIG so that application can continue to serve requests. We can achieve this by enabling **autoscaling**.

The only option that satisfies these three is Set autoscaling to On, set the minimum number of instances to 1, and then set the maximum number of instances to 1.

Ref: https://cloud.google.com/compute/docs/autoscaler

Bottom of Form

Top of Form

Question 11: **Correct**

**Your colleague updated a deployment manager template of a production application serving live traffic. You want to deploy the update to the live environment later during the night when user traffic is at its lowest. The git diff on the pull request shows the changes are substantial and you would like to review the intended changes without applying the changes in the live environment. You want to do this in the most efficient way possible. What should you do?**

​

Use granular logging statements within the Deployment Manager template authored in Python.

​

Execute the Deployment Manager template using the *--preview* option in the same project, and observe the status of interdependent resources

(Correct)

​

Monitor activity of the Deployment Manager executing on Stackdriver logging page of the GCP console.

​

Execute the Deployment manager template against a separate project with the same configuration, and monitor for failures

**Explanation**

Requirements - **confirm dependencies**, **rapid feedback**.

Use granular logging statements within the Deployment Manager template authored in Python. **is not right.**

Deployment Manager doesn't provide the ability to set granular logging statements. Moreover, if that was possible the logging statements wouldn't be written to a log file until the template is applied and it is already too late as the template is applied and we haven't had a chance to confirm that the dependencies of all defined resources are properly met

Monitor activity of the Deployment Manager executing on Stackdriver logging page of the GCP console. **is not right.**

This doesn't give us a chance to confirm that the dependencies of all defined resources are properly met before executing it.

Execute the Deployment manager template against a separate project with the same configuration, and monitor for failures. **is not right.**

While we can identify whether dependencies are met by monitoring the failures, it is not rapid. We need rapid feedback on changes and we want that before changes are committed (i.e. applied) to the project

Execute the Deployment Manager template using the --preview option in the same project, and observe the status of interdependent resources. **is the right answer.**

After we have written a configuration file, we can preview the configuration before you create a deployment. Previewing a configuration lets you see the resources that Deployment Manager would create but does not actually instantiate any actual resources. In gcloud command-line, you use the create sub-command with the **--preview** flag to preview configuration changes.

Ref: https://cloud.google.com/deployment-manager

Bottom of Form

Top of Form

Question 12: **Correct**

**You’ve deployed a microservice that uses sha1 algorithm with a salt value to has usernames. You deployed this to GKE cluster using deployment file:**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: sha1\_hash\_app-deployment**

**spec:**

**selector:**

**matchLabels:**

**app: sha1\_hash\_app**

**replicas: 3**

**template:**

**metadata:**

**labels:**

**app: sha1\_hash\_app**

**spec:**

**containers:**

**- name: hash-me**

**image: gcr.io/hash-repo/sha1\_hash\_app:2.17**

**env:**

**- name: SALT\_VALUE**

**value: "z0rtkty12$!"**

**ports:**

**- containerPort: 8080**

**You need to refactor this configuration so that the database password is not stored in plain text. You want to follow Google-recommended practices. What should you do?**

​

Store the database password inside a Secret object. Modify the YAML file to populate the SALT\_VALUE environment variable from the Secret.

(Correct)

​

Store the database password in a file inside a Kubernetes persistent volume, and use a persistent volume claim to mount the volume to the container.

​

Store the database password inside the Docker image of the container, not in the YAML file.

​

Store the database password inside a ConfigMap object. Modify the YAML file to populate the SALT\_VALUE environment variable from the ConfigMap.

**Explanation**

Store the database password inside the Docker image of the container, not in the YAML file. **is not right.**

Baking passwords into Docker images is a very bad idea. Anyone who spins up a container from this image has access to the password.

Store the database password inside a ConfigMap object. Modify the YAML file to populate the SALT\_VALUE environment variable from the ConfigMap. **is not right.**

ConfigMaps are useful for storing and sharing non-sensitive, unencrypted configuration information. To use sensitive information in your clusters, you must use Secrets.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/configmap

Store the database password in a file inside a Kubernetes persistent volume, and use a persistent volume claim to mount the volume to the container. **is not right.**

Persistent volumes should not be used for storing sensitive information. PersistentVolume resources are used to manage durable storage in a cluster and PersistentVolumeClaim is a request for and claim to a PersistentVolume resource.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/persistent-volumes

Store the database password inside a Secret object. Modify the YAML file to populate the SALT\_VALUE environment variable from the Secret. **is the right answer.**

In GKE, you can create a secret to hold the password; and then use the secret as an environment variable in the YAML file.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/secret

You can create a secret using *kubectl create secret generic passwords --from-literal* sha1\_hash\_app*\_SALT\_VALUE=*z0rtkty12$!

And you can then modify the YAML file to reference this secret as shown below.

apiVersion: apps/v1

kind: Deployment

metadata:

name: sha1\_hash\_app-deployment

spec:

selector:

matchLabels:

app: sha1\_hash\_app

replicas: 3

template:

metadata:

labels:

app: sha1\_hash\_app

spec:

containers:

- name: hash-me

image: gcr.io/hash-repo/sha1\_hash\_app:2.17

env:

- name: SALT\_VALUE

valueFrom:

secretKeyRef:

name: passwords

key: sha1\_hash\_app\_SALT\_VALUE

ports:

- containerPort: 8080

Bottom of Form

Top of Form

Question 13: **Incorrect**

**You have a collection of audio/video files over 80GB each that you need to migrate to Google Cloud Storage. The files are in your on-premises data center. What migration method can you use to help speed up the transfer process?**

​

Use the Cloud Transfer Service to transfer.

(Incorrect)

​

Start a recursive upload.

​

Use multithreaded uploads using the -m option.

​

Use parallel uploads to break the file into smaller chunks then transfer it simultaneously.

(Correct)

**Explanation**

Use parallel uploads to break the file into smaller chunks then transfer it simultaneously. **is the right answer.**

With cloud storage, Object composition can be used for uploading an object in parallel: you can divide your data into multiple chunks, upload each chunk to a distinct object in parallel, compose your final object, and delete any temporary source objects. This helps maximize your bandwidth usage and ensures the file is uploaded as fast as possible.

Ref: https://cloud.google.com/storage/docs/composite-objects#uploads

Use multithreaded uploads using the -m option. **is not right.**

Using the -m option lets you upload multiple files at the same time, but in our case, the individual files are over 80GB each. The best upload speed can be achieved by breaking the file into smaller chunks and transferring it simultaneously.

Use the Cloud Transfer Service to transfer. **is not right.**

Cloud Transfer Service is used for transferring massive amounts (in the range of petabytes of data) of data to the cloud. While nothing stops us from using Cloud Transfer Service to upload our files, it would be an overkill and very expensive.

Ref: https://cloud.google.com/products/data-transfer

Start a recursive upload. **is not right.**

In Google Cloud Storage, there is no such thing as a recursive upload.

Bottom of Form

Top of Form

Question 14: **Incorrect**

**In Cloud Shell, your active gcloud configuration is as shown below.**

**$ gcloud config list**

**[component\_manager]**

**disable\_update\_check = True**

**[compute]**

**gce\_metadata\_read\_timeout\_sec = 5**

**zone = europe-west2-a**

**[core]**

**account = gcp-ace-lab-user@gmail.com**

**disable\_usage\_reporting = False**

**project = gcp-ace-lab-266520**

**[metrics]**

**environment = devshell**

**You want to create two compute instances - one in europe-west2-a and another in europe-west2-b. What should you do? (Select 2)**

​

*gcloud compute instances create instance1*

*gcloud config set compute/zone europe-west2-b*

*gcloud compute instances create instance2*

(Correct)

​

*gcloud compute instances create instance1*

*gcloud compute instances create instance2 --zone=europe-west2-b*

(Correct)

​

*gcloud compute instances create instance1*

*gcloud config set zone europe-west2-b*

*gcloud compute instances create instance2*

​

*gcloud compute instances create instance1*

*gcloud configuration set compute/zone europe-west2-b*

*gcloud compute instances create instance2*

​

*gcloud compute instances create instance1*

*gcloud compute instances create instance2*

**Explanation**

gcloud compute instances create instance1

gcloud compute instances create instance2. **is not right.**

The default compute/zone property is set to europe-west2-a in the current gcloud configuration. Executing the two commands above would create two compute instances in the default zone i.e. europe-west2-a which doesn't satisfy our requirement.

Ref: https://cloud.google.com/sdk/gcloud/reference/compute/instances/create

gcloud compute instances create instance1

gcloud config set zone europe-west2-b

gcloud compute instances create instance2. **is not right.**

The approach is right but the syntax is wrong. gcloud config does not have a core/zone property. The syntax for this command is *gcloud config set SECTION/PROPERTY VALUE*. If *SECTION* is missing, *SECTION* is defaulted to *core*. We are effectively trying to run *gcloud config set core/zone europe-west2-b* but the core section doesn't have a property called *zone*, so this command fails.

Ref: https://cloud.google.com/sdk/gcloud/reference/config/set

gcloud compute instances create instance1

gcloud configuration set compute/zone europe-west2-b

gcloud compute instances create instance2. **is not right.**

Like above, the approach is right but the syntax is wrong. You want to set the default compute/zone property in gcloud configuration to europe-west2-b but it needs to be done via the command gcloud config set and not gcloud configuration set.

Ref: https://cloud.google.com/sdk/gcloud/reference/config/set

gcloud compute instances create instance1

gcloud config set compute/zone europe-west2-b

gcloud compute instances create instance2. **is the right answer.**

The default compute/zone property is europe-west2-a in the current gcloud configuration so executing the first *gcloud compute instances create* command creates the instance in europe-west2-a zone. Next, executing the *gcloud config set compute/zone europe-west2-b* changes the default compute/zone property in default configuration to europe-west2-b. Executing the second *gcloud compute instances create* command creates a compute instance in europe-west2-b which is what we want.

Ref: https://cloud.google.com/sdk/gcloud/reference/config/set

Ref: https://cloud.google.com/sdk/gcloud/reference/compute/instances/create

*gcloud compute instances create instance1*

*gcloud compute instances create instance2 --zone=europe-west2-b.* **is the right answer.**

The default compute/zone property is europe-west2-a in the current gcloud configuration so executing the first *gcloud compute instances create* command creates the instance in europe-west2-a zone. Next, executing the second *gcloud compute instances create* command with --zone property creates a compute instance in provided zone i.e. europe-west2-b instead of using the default zone from the current active configuration.

Ref: https://cloud.google.com/sdk/gcloud/reference/compute/instances/create

Bottom of Form

Top of Form

Question 15: **Incorrect**

**Your company wants to move 200 TB of your website clickstream logs from your on-premise data center to Google Cloud Platform. These logs need to be retained in GCP for compliance requirements. Your business analysts also want to run analytics on these logs to understand user click behavior on your website. Which of the below would enable you to meet these requirements? (Select Two)**

​

Load logs into Google BigQuery.

(Correct)

​

Import logs into Google Stackdriver.

​

Insert logs into Google Cloud Bigtable.

(Incorrect)

​

Upload log files into Google Cloud Storage.

(Correct)

​

Load logs into Google Cloud SQL.

**Explanation**

Load logs into Google Cloud SQL. **is not right.**

Cloud SQL is a fully-managed relational database service. Storing logs in Google Cloud SQL is very expensive. Cloud SQL doesn't help us with analytics. Moreover, Google Cloud Platform offers several storage classes in Google Cloud Storage that are more apt for storing logs at a much cheaper cost.

Ref: https://cloud.google.com/sql/docs

Ref: https://cloud.google.com/sql/pricing#sql-storage-networking-prices

Ref: https://cloud.google.com/storage/pricing

Import logs into Google Stackdriver. **is not right.**

You can push custom logs to Stackdriver and set custom retention periods to store the logs for longer durations. However, Stackdriver doesn't help us with analytics. You could create a sink and export data into Cloud BigQuery for analytics but that is more work. Moreover, Google Cloud Platform offers several storage classes in Google Cloud Storage that are more apt for storing logs at a much cheaper cost.

Ref: https://cloud.google.com/logging

Ref: https://cloud.google.com/storage/pricing

Insert logs into Google Cloud Bigtable. **is not right.**

Cloud Bigtable is a petabyte-scale, fully managed NoSQL database service for large analytical and operational workloads. Storing data in Bigtable (approx $0.17/GB + $0.65/hr per node) is very expensive compared to storing data in Cloud Storage (approx $0.02/GB in standard storage class) - which can go down further if you transition to Nearline/Coldline after running analytics.

Ref: https://cloud.google.com/bigtable/

Upload log files into Google Cloud Storage. **is the right answer.**

Google Cloud Platform offers several storage classes in Google Cloud Storage that are suitable for storing/archiving logs at a reasonable cost.

GCP recommends you use

1. Standard storage class if you need to access objects frequently

2. Nearline storage class if you access infrequently i.e. once a month

3. Coldline storage class if you access even less frequently e.g. once a quarter

4. Archive storage for logs archival.

Ref: https://cloud.google.com/storage/docs/storage-classes

Load logs into Google BigQuery. **is the right answer.**

By loading logs into Google BigQuery, you can securely run and share analytical insights in your organization with a few clicks. BigQuery’s high-speed streaming insertion API provides a powerful foundation for real-time analytics, making your latest business data immediately available for analysis.

Ref: https://cloud.google.com/bigquery#marketing-analytics

Bottom of Form

Top of Form

Question 16: **Correct**

**Your company retains all its audit logs in BigQuery for 10 years. At the annual audit every year, you need to provide the auditors' access to the audit logs. How should you do this efficiently?**

​

Add the auditor user accounts to two new custom IAM roles.

​

Add the auditors group to two new custom IAM roles.

​

Add the auditor user accounts to the 'logging.viewer' and 'bigQuery.dataViewer' predefined IAM roles.

​

Add the auditors group to the 'logging.viewer' and 'bigQuery.dataViewer' predefined IAM roles.

(Correct)

**Explanation**

Add the auditor user accounts to the 'logging.viewer' and 'bigQuery.dataViewer' predefined IAM roles. **is not right.**

Since auditing happens several times a year, we don't want to repeat the process of granting multiple roles to multiple users every time. Instead, we want to define a group with the required grants (a one time task) and assign this group to the auditor users during the time of the audit.

Add the auditor user accounts to two new custom IAM roles. **is not right.**

Google already provides roles that fit the external auditing requirements so we don't need to create custom roles. Nothing stops us from creating custom IAM roles to achieve the same purpose, but this doesn't follow "Google-recommended practices"

Add the auditors group to two new custom IAM roles. **is not right.**

Google already provides roles that fit the external auditing requirements so we don't need to create custom roles. Nothing stops us from creating custom IAM roles to achieve the same purpose, but this doesn't follow "Google-recommended practices"

Add the auditors group to the 'logging.viewer' and 'bigQuery.dataViewer' predefined IAM roles. **is the right answer.**

For external auditors, Google recommends we grant logging.viewer and bigquery.dataViewer roles. Since auditing happens several times a year to review the organization's audit logs, it is recommended we create a group with these grants and assign the group to auditor user accounts during the time of the audit.

Bottom of Form

Top of Form

Question 17: **Incorrect**

**You ran the following commands to create two compute instances.**

**gcloud compute instances create instance1**

**gcloud compute instances create instance2**

**Both compute instances were created in europe-west2-a zone but you want to create them in other zones. Your active gcloud configuration is as shown below.**

**$ gcloud config list**

**[component\_manager]**

**disable\_update\_check = True**

**[compute]**

**gce\_metadata\_read\_timeout\_sec = 5**

**zone = europe-west2-a**

**[core]**

**account = gcp-ace-lab-user@gmail.com**

**disable\_usage\_reporting = False**

**project = gcp-ace-lab-266520**

**[metrics]**

**environment = devshell**

**You want to modify the gcloud configuration such that you are prompted for a zone when you execute the create instance commands above. What should you do?**

​

*gcloud config unset zone*

​

*gcloud config unset compute/zone*

(Correct)

​

*gcloud config set compute/zone ""*

(Incorrect)

​

*gcloud config set zone ""*

**Explanation**

*gcloud config unset zone.* **is not right.**

*gcloud config* does not have a core/zone property. The syntax for this command is *gcloud config unset SECTION/PROPERTY. If SECTION* is missing from the command, *SECTION* is defaulted to *core*. We are effectively trying to run the command *gcloud config unset core/zone* but the core section doesn't have a property called *zone*, so this command fails.

$ gcloud config unset zone

ERROR: (gcloud.config.unset) Section [core] has no property [zone].

Ref: https://cloud.google.com/sdk/gcloud/reference/config/unset

*gcloud config set zone "".* **is not right.**

gcloud config does not have a core/zone property. The syntax for this command is *gcloud config set SECTION/PROPERTY VALUE*. If *SECTION* is missing, *SECTION* is defaulted to *core*. We are effectively trying to run *gcloud config set core/zone ""* but the core section doesn't have a property called *zone*, so this command fails.

$ gcloud config set zone ""

ERROR: (gcloud.config.unset) Section [core] has no property [zone].

Ref: https://cloud.google.com/sdk/gcloud/reference/config/set

*gcloud config set compute/zone "".* **is not right.**

This command uses the correct syntax but it doesn't unset the *compute/zone* property correctly. Instead it sets it to "" in gcloud configuration. When the *gcloud compute instances create* command runs, it picks the zone value from this configuration property which is "" and attempts to create an instance in "" zone and fails because zone "" doesn't exist. gcloud doesn't treat "" zone as an unset value. The zone must be explicitly unset if it is to be removed from the configuration.

$ gcloud config set compute/zone ""

$ gcloud compute instances create instance1

Zone: Expected type (<type 'int'>, <type 'long'>) for field id, found projects/compute-challenge-lab-266520/zones/ (type <type 'unicode'>)

Ref: https://cloud.google.com/sdk/gcloud/reference/config/set

*gcloud config unset compute/zone.* **is the right answer.**

This command uses the correct syntax and correctly unsets the zone in gcloud configuration. The next time *gcloud compute instances create* command runs, it knows there is no default zone defined in gcloud configuration and therefore prompts for a zone before the instance can be created.

Ref: https://cloud.google.com/sdk/gcloud/reference/config/unset

Bottom of Form

Top of Form

Question 18: **Correct**

**Your finance department wants you to create a new billing account and link all development and test Google Cloud Projects to the new billing account. What should you do?**

​

Verify that you are Billing Administrator for the billing account. Create a new project and link the new project to the existing billing account.

​

Verify that you you have Billing Account Creator role and are Project Billing Manager for the GCP project. Create a new billing account and link the new billing account to the existing project.

(Correct)

​

Verify that you you have Billing Account Creator role and are Project Billing Manager for the GCP project. Update the existing project to link it to the existing billing account.

​

Verify that you are Billing Administrator for the billing account. Update the existing project to link it to the existing billing account.

**Explanation**

Our requirement is to **link** an **existing google cloud project** with a **new billing account.**

Verify that you are Billing Administrator for the billing account. Create a new project and link the new project to the existing billing account. **is not right.**

We do not need to create a new project.

Verify that you you have Billing Account Creator role and are Project Billing Manager for the GCP project. Update the existing project to link it to the existing billing account. **is not right.**

We want to link the project with a new billing account so is not right.

Verify that you are Billing Administrator for the billing account. Update the existing project to link it to the existing billing account. **is not right.**

We want to link the project with a new billing account so is not right.

Verify that you you have Billing Account Creator role and are Project Billing Manager for the GCP project. Create a new billing account and link the new billing account to the existing project. **is the right answer.**

The purpose of Project Billing Manager is to Link/unlink the project to/from a billing account. It is granted at the organization or project level. Project Billing Manager role allows a user to attach the project to the billing account, but does not grant any rights over resources. Project Owners can use this role to allow someone else to manage the billing for the project without granting them resource access.

Billing Account Creator - Use this role for initial billing setup or to allow creation of additional billing accounts.

Ref: https://cloud.google.com/billing/docs/how-to/billing-access

Bottom of Form

Top of Form

Question 19: **Incorrect**

**You have been asked to create a new Kubernetes Cluster on Google Kubernetes Engine that can autoscale the number of worker nodes as well as pods. What should you do? (Select 2)**

​

Configure a Compute Engine instance as a worker and add it to an unmanaged instance group. Add a load balancer to the instance group and rely on the load balancer to create additional Compute Engine instances when needed.

​

Create Compute Engine instances for the workers and the master and install Kubernetes. Rely on Kubernetes to create additional Compute Engine instances when needed.

​

Enable Horizontal Pod Autoscaling for the Kubernetes deployment.

(Correct)

​

Create a GKE cluster and enable autoscaling on the instance group of the cluster.

(Incorrect)

​

Create a GKE cluster and enable autoscaling on Kubernetes Engine.

(Correct)

**Explanation**

Create a GKE cluster and enable autoscaling on the instance group of the cluster. **is not right.**

GKE's cluster auto-scaler automatically resizes the number of nodes in a given node pool, based on the demands of your workloads. However, we should not enable Compute Engine autoscaling for managed instance groups for the cluster nodes. GKE's cluster auto-scaler is separate from Compute Engine autoscaling.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/cluster-autoscaler

Configure a Compute Engine instance as a worker and add it to an unmanaged instance group. Add a load balancer to the instance group and rely on the load balancer to create additional Compute Engine instances when needed. **is not right.**

When using GKE to manage your Kubernetes clusters, you can not add manually created compute instances to the worker node pool. A node pool is a group of nodes within a cluster that all have the same configuration. Node pools use a NodeConfig specification.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/node-pools

Moreover, Unmanaged instance groups do not autoscale. An unmanaged instance group is simply a collection of virtual machines (VMs) that reside in a single zone, VPC network, and subnet. An unmanaged instance group is useful for grouping together VMs that require individual configuration settings or tuning.

Ref: https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-unmanaged-instances

Create Compute Engine instances for the workers and the master and install Kubernetes. Rely on Kubernetes to create additional Compute Engine instances when needed. **is not right.**

When using Google Kubernetes Engine, you can not install master node separately. The cluster master runs the Kubernetes control plane processes, including the Kubernetes API server, scheduler, and core resource controllers. The master's lifecycle is managed by GKE when you create or delete a cluster.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/cluster-architecture

Also, you can not add manually created compute instances to the worker node pool. A node pool is a group of nodes within a cluster that all have the same configuration. Node pools use a NodeConfig specification.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/node-pools

Create a GKE cluster and enable autoscaling on Kubernetes Engine. **is the right answer.**

GKE's cluster autoscaler automatically resizes the number of nodes in a given node pool, based on the demands of your workloads. You don't need to manually add or remove nodes or over-provision your node pools. Instead, you specify a minimum and maximum size for the node pool, and the rest is automatic. When demand is high, cluster autoscaler adds nodes to the node pool. When demand is low, cluster autoscaler scales back down to a minimum size that you designate. This can increase the availability of your workloads when you need it while controlling costs.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/cluster-autoscaler

Enable Horizontal Pod Autoscaling for the kubernetes deployment. **is the right answer.**

Horizontal Pod Autoscaler scales up and scales down your Kubernetes workload by automatically increasing or decreasing the number of Pods in response to the workload's CPU or memory consumption, or in response to custom metrics reported from within Kubernetes or external metrics from sources outside of your cluster. Horizontal Pod Autoscaling cannot be used for workloads that cannot be scaled, such as DaemonSets.

Ref: https://cloud.google.com/kubernetes-engine/docs/concepts/horizontalpodautoscaler

Bottom of Form

Top of Form

Question 20: **Correct**

**You have asked your supplier to send you a purchase order and you want to enable them to upload the file to a cloud storage bucket within the next 4 hours. Your supplier does not have a Google account. You want to follow Google recommended practices. What should you do?**

​

Create a service account with just the permissions to upload files to the bucket. Create a JSON key for the service account. Execute the command *gsutil signurl -httpMethod PUT -d 4h <JSON Key File> gs://<bucket>/\*\**.

​

Create a JSON key for the Default Compute Engine Service Account. Execute the command *gsutil signurl -m PUT -d 4h <JSON Key File> gs://<bucket>/\*\**.

​

Create a service account with just the permissions to upload files to the bucket. Create a JSON key for the service account. Execute the command *gsutil signurl -d 4h <JSON Key File> gs://<bucket>/*.

​

Create a service account with just the permissions to upload files to the bucket. Create a JSON key for the service account. Execute the command *gsutil signurl -m PUT -d 4h <JSON Key File> gs://<bucket>/po.pdf*.

(Correct)

**Explanation**

Create a service account with just the permissions to upload files to the bucket. Create a JSON key for the service account. Execute the command *gsutil signurl -d 4h <JSON Key File> gs://<bucket>/*. **is not right.**

This command creates signed URLs for retrieving existing objects. This command does not specify a HTTP method and in the absence of one, the default HTTP method is GET.

Ref: https://cloud.google.com/storage/docs/gsutil/commands/signurl

Create a service account with just the permissions to upload files to the bucket. Create a JSON key for the service account. Execute the command *gsutil signurl -httpMethod PUT -d 4h <JSON Key File> gs://<bucket>/\*\**. **is not right.**

gsutil signurl does not accept -httpMethod parameter.

$ gsutil signurl -d 4h -httpMethod PUT keys.json gs://gcp-ace-lab-255520/\*

CommandException: Incorrect option(s) specified. Usage:

The HTTP method can be provided through -m flag.

Ref: https://cloud.google.com/storage/docs/gsutil/commands/signurl

Create a JSON key for the Default Compute Engine Service Account. Execute the command *gsutil signurl -m PUT -d 4h <JSON Key File> gs://<bucket>/\*\**. **is not right.**

Using the default compute engine service account violates the principle of least privilege. The recommended approach is to create a service account with just the right permissions needed and create JSON keys for this service account to use with *gsutil signurl* command.

Create a service account with just the permissions to upload files to the bucket. Create a JSON key for the service account. Execute the command *gsutil signurl -m PUT -d 4h <JSON Key File> gs://<bucket>/po.pdf*. **is the right answer.**

This command correctly creates a signed url that is valid for 4 hours and allows PUT (through the -m flag) operations on the file po.pdf in the bucket. The supplier can then use the signed URL to upload a file to this bucket within 4 hours.

Ref: https://cloud.google.com/storage/docs/gsutil/commands/signurl

Bottom of Form

Top of Form

Question 21: **Correct**

**You want to list all the compute instances in zones us-central1-b and europe-west1-d. Which of the commands below should you run to retrieve this information?**

​

*gcloud compute instances get --filter="zone:( us-central1-b )"* and *gcloud compute instances list --filter="zone:( europe-west1-d )"* and combine the results.

​

*gcloud compute instances get --filter="zone:( us-central1-b europe-west1-d )"*

​

*gcloud compute instances list --filter="zone:( us-central1-b )"* and *gcloud compute instances list --filter="zone:( europe-west1-d )"* and combine the results.

​

*gcloud compute instances list --filter="zone:( us-central1-b europe-west1-d )"*

(Correct)

**Explanation**

*gcloud compute instances get --filter="zone:( us-central1-b europe-west1-d )"*. **is not right.**

*gcloud compute instances* command does not support *get* action.

Ref: https://cloud.google.com/sdk/gcloud/reference/compute/instances

*gcloud compute instances get --filter="zone:( us-central1-b )"* and *gcloud compute instances list --filter="zone:( europe-west1-d )"* and combine the results. **is not right.**

*gcloud compute instances* command does not support *get* action.

Ref: https://cloud.google.com/sdk/gcloud/reference/compute/instances

*gcloud compute instances list --filter="zone:( us-central1-b )"* and *gcloud compute instances list --filter="zone:( europe-west1-d )"* and combine the results. **is not right.**

The first command retrieves compute instances from us-central1-b and the second command retrieves compute instances from europe-west1-d. The output from the two statements can be combined to create a full list of instances from us-central1-b and europe-west1-d, however, this is not efficient as it is a manual activity. Moreover, gcloud already provides the ability to list and filter on multiple zones in a single command.

Ref: https://cloud.google.com/sdk/gcloud/reference/compute/instances/list

*gcloud compute instances list --filter="zone:( us-central1-b europe-west1-d )"*. **is the right answer.**

gcloud compute instances list - lists Google Compute Engine instances. The output includes internal as well as external IP addresses. The filter expression --filter="zone:( us-central1-b europe-west1-d )" is used to filter instances from zones us-central1-b and europe-west1-d.

Ref: https://cloud.google.com/sdk/gcloud/reference/compute/instances/list

Here's a sample output of the command.

$gcloud compute instances list

NAME ZONE MACHINE\_TYPE PREEMPTIBLE INTERNAL\_IP EXTERNAL\_IP STATUS

gke-cluster-1-default-pool-8c599c87-16g9 us-central1-a n1-standard-1 10.128.0.8 35.184.212.227 RUNNING

gke-cluster-1-default-pool-8c599c87-36xh us-central1-b n1-standard-1 10.129.0.2 34.68.254.220 RUNNING

gke-cluster-1-default-pool-8c599c87-lprq us-central1-c n1-standard-1 10.130.0.13 35.224.96.151 RUNNING

$gcloud compute instances list --filter="zone:( us-central1-b europe-west1-d )"

NAME ZONE MACHINE\_TYPE PREEMPTIBLE INTERNAL\_IP EXTERNAL\_IP STATUS

gke-cluster-1-default-pool-8c599c87-36xhus-central1-bn1-standard-110.129.0.234.68.254.220RUNNING

Bottom of Form

Top of Form

Question 22: **Incorrect**

**Users of your application are complaining of slowness when loading the application. You realize the slowness is because the App Engine deployment serving the application is deployed in us-central whereas all users of this application are closest to europe-west3. You want to change the region of the App Engine application to europe-west3 to minimize latency. What's the best way to change the App Engine region?**

​

Use the gcloud app region set command and supply the name of the new region.

​

Contact Google Cloud Support and request the change.

​

Create a new project and create an App Engine instance in europe-west3

(Correct)

​

From the console, under the App Engine page, click edit, and change the region drop-down.

(Incorrect)

**Explanation**

Use the gcloud app region set command and supply the name of the new region. **is not right.**

*gcloud app region* command does not provide a set action. The only action *gcloud app region* command currently supports is *list* which lists the availability of flex and standard environments for each region.

Ref: https://cloud.google.com/sdk/gcloud/reference/app/regions/list

Contact Google Cloud Support and request the change. **is not right.**

Unfortunately, Google Cloud Support isn't of much use here as they would not be able to change the region of an App Engine Deployment. App engine is a regional service, which means the infrastructure that runs your app(s) is located in a specific region and is managed by Google to be redundantly available across all the zones within that region. Once an app engine deployment is created in a region, it can't be changed.

Ref: https://cloud.google.com/appengine/docs/locations

From the console, Click edit in App Engine dashboard page and change the region drop-down. **is not right.**

The settings mentioned in this option aren't available in the App Engine dashboard. App engine is a regional service. Once an app engine deployment is created in a region, it can't be changed. As shown in the screenshot below, Region is greyed out.

Create a new project and create an App Engine instance in europe-west3. **is the right answer.**

App engine is a regional service, which means the infrastructure that runs your app(s) is located in a specific region and is managed by Google to be redundantly available across all the zones within that region. Once an app engine deployment is created in a region, it can't be changed. The only way is to create a new project and create an App Engine instance in europe-west3, send all user traffic to this instance and delete the app engine instance in us-central.

Ref: https://cloud.google.com/appengine/docs/locations

Bottom of Form

Top of Form

Question 23: **Incorrect**

**Your company runs all its applications in us-central1 region in a single GCP project. The company has recently expanded its operations to Europe, but customers in the EU are complaining about slowness accessing the application. Your manager has requested you to deploy a new instance in the same project in europe-west1 region to reduce latency to the EU customers. The newly deployed VM needs to reach a central Citrix Licensing Server in us-central-1. How should you design the network and firewall rules?**

​

1. Create a subnet in the same VPC, in europe-west1.

2. Create the new instance in the new subnet and use the Citrix Licensing Server's private address as the endpoint.

(Correct)

​

1. Create a VPC and a subnet in europe-west1.

2. Peer the 2 VPCs.

3. Create the new instance in the new subnet and use the Citrix Licensing Server's private address as the endpoint.

(Incorrect)

​

1. Create a subnet in the same VPC, in europe-west1.

2. Use Cloud VPN to connect the two subnets.

3. Create the new instance in the new subnet and use the Citrix Licensing Server's private address as the endpoint.

​

1. Create a VPC and a subnet in europe-west1.

2. Expose the application with an internal load balancer.

3. Create the new instance in the new subnet and use the load balancer's address as the endpoint.

**Explanation**

Our requirements are to **connect** the instance in europe-west1 region with the application running in us-central1 region following **Google-recommended practices**. The two instances are in the **same project**.

1. Create a VPC and a subnet in europe-west1.

2. Expose the application with an internal load balancer.

3. Create the new instance in the new subnet and use the load balancer's address as the endpoint. **is not right.**

We have two different VPCs. There is no mention of the CIDR range so let's assume the two subnets in two VPCs use different CIDR ranges. However, there is no communication route between the two VPCs. If we create an internal load balancer, that load balancer is not visible outside the VPC. So the new instance cannot connect to the load balancer's internal address.

Ref: https://cloud.google.com/load-balancing/docs/internal

1. Create a subnet in the same VPC, in europe-west1.

2. Use Cloud VPN to connect the two subnets.

3. Create the new instance in the new subnet and use the Citrix Licensing Server's private address as the endpoint. **is not right.**

Cloud VPN securely connects your on-premises network to your Google Cloud (GCP) Virtual Private Cloud (VPC) network through an IPsec VPN connection. It is not meant to connect two subnets within the same VPC. Moreover, subnets within the same VPC can communicate with each other by setting up relevant firewall rules.

1. Create a VPC and a subnet in europe-west1.

2. Peer the 2 VPCs.

3. Create the new instance in the new subnet and use the Citrix Licensing Server's private address as the endpoint. **is not right.**

Given that the new instance wants to access the application on the existing compute engine instance, these applications seem to be related so they should be within the same VPC. This option does not mention how the VPC networks are created and what the subnet range is.

1. You can't connect two auto mode VPC networks together using VPC Network Peering because their subnets use identical primary IP ranges. We don't know how the VPCs were created.

2. There are a number of restrictions based on the subnet ranges. <https://cloud.google.com/vpc/docs/vpc-peering#restrictions>

Even if we assume the above restrictions don’t apply and enable peering is possible, this is still a lot of additional work and we can simplify this by choosing the option below (which is the answer)

1. Create a subnet in the same VPC, in europe-west1.

2. Create the new instance in the new subnet and use the Citrix Licensing Server's private address as the endpoint. **is the right answer.**

We can create another subnet in the same VPC and this subnet is located in europe-west1. We can then spin up a new instance in this subnet. We also have to set up a firewall rule to allow communication between the two subnets. All instances in the two subnets with the same VPC can communicate through the internal IP Address

Ref: https://cloud.google.com/vpc

Bottom of Form

Top of Form

Question 24: **Incorrect**

**You are migrating a mission-critical application from your on-premises data centre to Google Cloud, and you need to ensure unhealthy compute instances within the autoscaled managed instances group are recreated automatically. What should you do?**

​

In the Instance Template, add the label *health-check*.

(Incorrect)

​

Select Multi-Zone instead of Single-Zone when creating the Managed Instance Group.

​

In the Instance Template, add a startup script that sends a heartbeat to the metadata server.

​

Create a health check on port 443 and use that when creating the Managed Instance Group.

(Correct)

**Explanation**

Our requirement is to ensure **unhealthy VMs are recreated**.

Select Multi-Zone instead of Single-Zone when creating the Managed Instance Group. **is not right.**

You can create two types of MIGs: A zonal MIG, which deploys instances to a single zone and a regional MIG, which deploys instances to multiple zones across the same region. However, this doesn't help with recreating unhealthy VMs.

Ref: https://cloud.google.com/compute/docs/instance-groups

In the Instance Template, add the label *health-check*. **is not right.**

Metadata entries are key-value pairs and do not influence any other behavior.

Ref: https://cloud.google.com/compute/docs/storing-retrieving-metadata

In the Instance Template, add a startup script that sends a heartbeat to the metadata server. **is not right.**

The startup script is executed only at the time of startup. Whereas we need something like a liveness check that monitors the status of the server periodically to identify if the VM is unhealthy. So this is not going to work.

Ref: https://cloud.google.com/compute/docs/startupscript

Create a health check on port 443 and use that when creating the Managed Instance Group. **is the right answer.**

To improve the availability of your application and to verify that your application is responding, you can configure an auto-healing policy for your managed instance group (MIG). An auto-healing policy relies on an application-based health check to verify that an application is responding as expected. If the auto healer determines that an application isn't responding, the managed instance group automatically recreates that instance. Since our application is a HTTPS web application, we need to set up our health check on port 443 which is the standard port for HTTPS.

Ref: https://cloud.google.com/compute/docs/instance-groups/autohealing-instances-in-migs

Bottom of Form

Top of Form

Question 25: Skipped

**Your team created two networks (VPC) in Google Cloud in the same region. The first VPC hosts an encryption service on Cluster Autoscaler enabled GKE cluster. The encryption service provides TCP endpoints to encrypt and decrypt data. The second VPC *pt-network* hosts a user management system on a single Google Cloud Compute Engine VM. The user management system deals with PII data and needs to invoke the encryption endpoints running on the GKE cluster to encrypt data before persisting to disk. The CIDR ranges of the VPCs do not overlap. What should you do to enable the compute engine VM invoke the TCP encryption endpoints while minimizing effort?**

​

1. In GKE, create a *service* of type *LoadBalancer* that uses the application’s pods as backend.

2. Set the service’s *externalTrafficPolicy* to *Cluster.*

3. Configure the Compute Engine instance to use the address of the load balancer that has been assigned.

​

1. In GKE, create a *Service* of type *LoadBalancer* that uses the application’s Pods as backend.

2. Add an annotation to this service *cloud.google.com/load-balancer-type: Internal*

3. Peer the two VPCs together

4. Configure the Compute Engine instance to use the address of the load balancer that has been created.

(Correct)

​

1. In GKE, create a *service* of type *NodePort* that uses the application’s pods as backend.

2. Create a Compute Engine instance called *proxy* with 2 network interfaces one in VPC.

3. Use iptables on the instance to forward traffic from *pt-network* to the GKE nodes.

4. Configure the Compute Engine instance to use the address of *proxy* in *pt-network* as endpoint.

​

1. In GKE, create a *Service* of type *LoadBalancer* that uses the application’s Pods as backend.

2. Add a Cloud Armor Security Policy to the load balancer that whitelists the internal IPs of the MIG’s instances.

3. Configure the Compute Engine instance to use the address of the load balancer that has been created.

**Explanation**

While it may be possible to set up the networking to let the compute engine instance in pt-network communicate with pods in the GKE cluster in multiple ways, we need to look for an option that **minimizes effort**. Generally speaking, this means using Google Cloud Platform services directly and configuring them to achieve the intended outcome; over setting up a service ourselves, installing/managing/upgrading it ourselves which is manual, error-prone, time-consuming and add to operational overhead.

1. In GKE, create a service of type LoadBalancer that uses the application’s pods as backend.

2. Set the service’s externalTrafficPolicy to Cluster.

3. Configure the Compute Engine instance to use the address of the load balancer that has been assigned. **is not right.**

In GKE, services are used to expose pods to the outside world. There are multiple types of services. The three common types are - NodePort, ClusterIP, and LoadBalancer (there are two more service types - ExternalName and Headless which are not relevant in this context). We do not want to create a Cluster IP as this is not accessible outside the cluster. And we do not want to create NodePort as this results in exposing a port on each node in the cluster; and as we have multiple replicas, this will result in them trying to open the same port on the nodes which fail. The compute engine instance in pt-network needs a single point of communication to reach GKE. This is achieved by creating a service of type LoadBalancer. This gives the service a public IP that is externally accessible.

Ref: https[://cloud.google.com/kubernetes-engine/docs/how-to/exposing-apps](https://cloud.google.com/kubernetes-engine/docs/how-to/exposing-apps)

externalTrafficPolicy denotes how the service should route external traffic - including public access. Rather than trying to explain, I’ll point you to a very good blog that does a great job of answering how this works. <https://www.asykim.com/blog/deep-dive-into-kubernetes-external-traffic-policies>

Since we have cluster autoscaling enabled, we can have more than 1 node and possibly multiple replicas running on each node. So externalTrafficPolicy set to Cluster plays well with our requirement.

Finally, we configure the compute engine to use the (externally accessible) address of the load balancer.

So this certainly looks like an option, but is it the best option that **minimizes effort**? One of the disadvantages of this option is that it exposes the pods **publicly** by using a service of type LoadBalancer. We want our compute engine to talk to the pods, but do we really want to expose our pods to the whole world? Maybe not!! Let’s look at the other options to find out if there is something more relevant and secure.

1. In GKE, create a service of type NodePort that uses the application’s pods as backend.

2. Create a Compute Engine instance called proxy with 2 network interfaces one in VPC.

3. Use iptables on the instance to forward traffic from pt-network to the GKE nodes.

4. Configure the Compute Engine instance to use the address of proxy in pt-network as endpoint. **is not right.**

For reasons explained in the above option, we don’t want to create a service of type NodePort. This opens up a port on each node for each replica (pod). If we choose to do this, the compute engine doesn’t have a single point to contact. Instead, it would need to contact the GKE cluster nodes individually - and that is bound to have issues because we have autoscaling enabled and the nodes may scale up and scale down as per the scaling requirements. New nodes may have different IP addresses to the previous nodes, so unless the Compute engine is continuously supplied with the IP addresses of the nodes, it can’t reach them. Moreover, we have multiple replicas and it is possible we might have multiple replicas of the pod on the same node in which case they all can’t open the same node port - once a node port is opened by one replica (pod), it can’t be used by other replicas on the same node. So this option can be ruled out without going into the rest of the answer.

1. In GKE, create a Service of type LoadBalancer that uses the application’s Pods as backend.

2. Add a Cloud Armor Security Policy to the load balancer that whitelists the internal IPs of the MIG’s instances.

3. Configure the Compute Engine instance to use the address of the load balancer that has been created. **is not right.**

Creating a service of type LoadBalancer and getting a Compute Engine instance to use the address of the load balancer is fine, but Cloud Armor is not required. You could certainly use Cloud Armor to set up a whitelist policy to only let traffic through from the compute engine instance, but hang on - this option says “MIG instances”. We don’t have a managed instance group. The question mentions a single instance but not MIG. If we were to assume the single instance is part of a MIG, i.e. a MIG with a single instance, this option works too. It is more secure than the first option discussed in the explanation but at the same time more expensive. Let’s look at the other option to see if it provides a secure yet cost-effective way of achieving the same.

1. In GKE, create a Service of type LoadBalancer that uses the application’s Pods as backend.

2. Add an annotation to this service cloud.google.com/load-balancer-type: Internal

3. Peer the two VPCs together

4. Configure the Compute Engine instance to use the address of the load balancer that has been created. **is the right answer.**

Creating a service of type LoadBalancer and getting a Compute Engine instance to use the address of the load balancer is fine. We covered this previously in the first option in the explanations section.

Adding the annotation *cloud.google.com/load-balancer-type: Internal* makes your cluster's services accessible to applications outside of your cluster that use the same VPC network and are located in the same Google Cloud region. So this improves security by not allowing public access, however, the compute engine is located in a different VPC so it can’t access.

Ref: https[*://cloud.google.com/kubernetes-engine/docs/how-to/internal-load-balancing*](https://cloud.google.com/kubernetes-engine/docs/how-to/internal-load-balancing)

But peering the VPCs together enables the compute engine to access the load balancer IP. And peering is possible because they do not use overlapping IP ranges. Peering essentially links up the two VPCs and resources inside the VPCs can communicate with each other as if they were all in a single VPC. More info about VPC peering: <https://cloud.google.com/vpc/docs/vpc-peering>

**So this option is the right answer**. It provides a **secure** and **cost-effective** way of achieving our requirements. There are several valid answers but this option is more correct than the others.

Bottom of Form

Top of Form

Question 26: **Incorrect**

**You deployed a Java application in a Google Compute Engine VM that has 3.75 GB Memory and 1 vCPU. At peak usage, the application experiences *java.lang.OutOfMemory* errors that take down the application entirely and requires a restart. The CPU usage at all times is minimal. Your operations team have asked you to increase the memory on the VM instance to 8 GB. You want to do this while minimizing the cost. What should you do?**

​

Stop the VM, increase the memory to *8 GB* and start the VM

(Correct)

​

Use gcloud to add metadata to the VM. Set the key to *required-memory-size* and the value to *8 GB*

(Incorrect)

​

Rely on live migration to move the workload to a machine with more memory.

​

Stop the VM, change the machine type to *n1-standard-2* and start the VM

**Explanation**

Rely on live migration to move the workload to a machine with more memory. **is not right.**

Live migration migrates your running instances to another host in the same zone so that Google can perform maintenance such as a software or hardware update. It can not be used for changing machine type.

Ref: https://cloud.google.com/compute/docs/instances/live-migration

Use gcloud to add metadata to the VM. Set the key to *required-memory-size* and the value to 8 GB. **is not right.**

There is no such setting as required-memory-size.

Stop the VM, change the machine type to *n1-standard-2* and start the VM. **is not right.**

n1-standard-2 instance offers less than 8 GB (7.5 GB to be precise) so this falls short of the required memory.

Ref: https://cloud.google.com/compute/docs/machine-types

Stop the VM, increase the memory to 8 GB and start the VM. **is the right answer**.

In Google compute engine, if predefined machine types don't meet your needs, you can create an instance with custom virtualized hardware settings. Specifically, you can create an instance with a custom number of vCPUs and custom memory, effectively using a custom machine type. Custom machine types are ideal for the following scenarios:

1. Workloads that aren't a good fit for the predefined machine types that are available to you.

2. Workloads that require more processing power or more memory but don't need all of the upgrades that are provided by the next machine type level.

In our scenario, we only need a memory upgrade. Moving to a bigger instance would also bump up the CPU which we don't need so we have to use a custom machine type. It is not possible to change memory while the instance is running so you need to first stop the instance, change the memory and then start it again. See below a screenshot that shows how CPU/Memory can be customized for an instance that has been stopped.

Ref: https://cloud.google.com/compute/docs/instances/creating-instance-with-custom-machine-type

Bottom of Form

Top of Form

Question 27: **Correct**

**You want to create a Google Cloud Storage regional bucket logs-archive in the Los Angeles region (us-west2). You want to use Coldline storage class to minimize costs and you want to retain files for 10 years. Which of the following commands should you run to create this bucket?**

​

*gsutil mb -l us-west2 -s nearline --retention 10y gs://logs-archive*

​

*gsutil mb -l los-angeles -s coldline --retention 10m gs://logs-archive*

​

*gsutil mb -l us-west2 -s coldline --retention 10y gs://logs-archive*

(Correct)

​

*gsutil mb -l us-west2 -s coldline --retention 10m gs://logs-archive*

**Explanation**

*gsutil mb -l us-west2 -s nearline --retention 10y gs://logs-archive.* **is not right.**

This command creates a bucket that uses nearline storage class whereas we want to use Coldline storage class.

Ref: https://cloud.google.com/storage/docs/gsutil/commands/mb

*gsutil mb -l los-angeles -s coldline --retention 10m gs://logs-archive.* **is not right.**

This command uses los-angeles as the location but los-angeles is not a supported region name. The region name for Los Angeles is us-west-2.

Ref: https://cloud.google.com/storage/docs/locations

*gsutil mb -l us-west2 -s coldline --retention 10m gs://logs-archive.* **is not right.**

This command creates a bucket with retention set to 10 months whereas we want to retain the objects for 10 years.

Ref: https://cloud.google.com/storage/docs/gsutil/commands/mb

*gsutil mb -l us-west2 -s coldline --retention 10y gs://logs-archive.* **is the right answer.**

This command correctly creates a bucket in Los Angeles, uses Coldline storage class and retains objects for 10 years.

Ref: https://cloud.google.com/storage/docs/gsutil/commands/mb

Bottom of Form

Top of Form

Question 28: **Incorrect**

**Your company stores customer PII data in Cloud Storage buckets. A subset of this data is regularly imported into a BigQuery dataset to carry out analytics. You want to make sure the access to this bucket is strictly controlled. Your analytics team needs read access on the bucket so that they can import data in BigQuery. Your operations team needs read/write access to both the bucket and BigQuery dataset to add Customer PII data of new customers on an ongoing basis. Your Data Vigilance officers need Administrator access to the Storage bucket and BigQuery dataset. You want to follow Google recommended practices. What should you do?**

​

At the Organization level, add your Data Vigilance officers user accounts to the Owner role, add your operations team user accounts to the Editor role, and add your analytics team user accounts to the Viewer role.

​

Use the appropriate predefined IAM roles for each of the access levels needed for Cloud Storage and BigQuery. Add your users to those roles for each of the services.

(Correct)

​

Create 3 custom IAM roles with appropriate permissions for the access levels needed for Cloud Storage and BigQuery. Add your users to the appropriate roles.

​

At the Project level, add your Data Vigilance officers user accounts to the Owner role, add your operations team user accounts to the Editor role, and add your analytics team user accounts to the Viewer role.

(Incorrect)

**Explanation**

At the Organization level, add your Data Vigilance officers user accounts to the Owner role, add your operations team user accounts to the Editor role, and add your analytics team user accounts to the Viewer role. **is not right.**

Google recommends we apply the security principle of least privilege, where we grant only necessary permissions to access specific resources.

Ref: https://cloud.google.com/iam/docs/overview

Providing these primitive roles at the organization levels grants them permissions on all resources in all projects under the organization which violates the security principle of least privilege.

Ref: https://cloud.google.com/iam/docs/understanding-roles

At the Project level, add your Data Vigilance officers user accounts to the Owner role, add your operations team user accounts to the Editor role, and add your analytics team user accounts to the Viewer role. **is not right.**

Google recommends we apply the security principle of least privilege, where we grant only necessary permissions to access specific resources.

Ref: https://cloud.google.com/iam/docs/overview

Providing these primitive roles at the project level grants them permissions on all resources in the project which violates the security principle of least privilege.

Ref: https://cloud.google.com/iam/docs/understanding-roles

Create 3 custom IAM roles with appropriate permissions for the access levels needed for Cloud Storage and BigQuery. Add your users to the appropriate roles. **is not right.**

While this has the intended outcome, it is not very efficient particularly when there are predefined roles that can be used. Secondly, if Google adds/modifies permissions for these services in the future, we would have to update our roles to reflect the modifications. This results in operational overhead and increases costs.

Ref: https://cloud.google.com/storage/docs/access-control/iam-roles#primitive-roles-intrinsic

Ref: https://cloud.google.com/bigquery/docs/access-control

Use the appropriate predefined IAM roles for each of the access levels needed for Cloud Storage and BigQuery. Add your users to those roles for each of the services. **is the right answer.**

For Google Cloud Storage service, Google provides predefined roles roles/owner, roles/editor, roles/viewer that match the access levels we need.

Similarly, Google provides the roles roles/bigquery.dataViewer, roles/bigquery.dataOwner, roles/bigquery.admin that match the access levels we need.

We can assign these predefined IAM roles to the respective users. Should Google add/modify permissions for these services in the future, we don't need to modify the roles above as Google does this for us; and this helps future proof our solution.

Ref: https://cloud.google.com/storage/docs/access-control/iam-roles#primitive-roles-intrinsic

Ref: https://cloud.google.com/bigquery/docs/access-control

Bottom of Form

Top of Form

Question 29: **Correct**

**Your company wants to move all documents from a secure internal NAS drive to a Google Cloud Storage (GCS) bucket. The data contains personally identifiable information (PII) and sensitive customer information. Your company tax auditors need access to some of these documents. What security strategy would you recommend on GCS?**

​

Grant no Google Cloud Identity and Access Management (Cloud IAM) roles to users, and use granular ACLs on the bucket.

(Correct)

​

Create randomized bucket and object names. Enable public access, but only provide specific file URLs to people who do not have Google accounts and need access.

​

Use signed URLs to generate time-bound access to objects.

​

Grant IAM read-only access to users, and use default ACLs on the bucket.

**Explanation**

Use signed URLs to generate time-bound access to objects. **is not right.**

When dealing with sensitive customer information such as PII, using signed URLs is not a great idea as anyone with access to the URL has access to PII data. Signed URLs provide time-limited resource access to anyone in possession of the URL, regardless of whether they have a Google account. With PII Data, we want to be sure who has access and signed URLs don't guarantee that.

Ref: https://cloud.google.com/storage/docs/access-control/signed-urls

Grant IAM read-only access to users, and use default ACLs on the bucket. **is not right.**

We do not need to grant all IAM read-only access to this sensitive data. Just the users who need access to sensitive/PII data should be provided access to this data.

Create randomized bucket and object names. Enable public access, but only provide specific file URLs to people who do not have Google accounts and need access. **is not right.**

Enabling public access to the buckets and objects makes them visible to everyone. There are a number of scanning tools out in the market with the sole purpose of identifying buckets/objects that can be reached publicly. Should one of these tools be used by a bad actor to find out our public bucket/objects, it would result in a security breach.

Grant no Google Cloud Identity and Access Management (Cloud IAM) roles to users, and use granular ACLs on the bucket. **is the right answer.**

We start with no explicit access to any of the IAM users, and the bucket ACLs can then control which users can access what objects. This is the most secure way of ensuring just the people who require access to the bucket are provided with access. We block everyone from accessing the bucket and explicitly provided access to specific users through ACLs.

Bottom of Form

Top of Form

Question 30: **Correct**

**You want to reduce storage costs for infrequently accessed data. The data will still be accessed approximately once a month and data older than 2 years is no longer needed. What should you do to reduce storage costs? (Select 2)**

​

Store infrequently accessed data in a Multi-Regional bucket.

​

Set an Object Lifecycle Management policy to change the storage class to Coldline for data older than 2 years.

​

Store infrequently accessed data in a Nearline bucket.

(Correct)

​

Set an Object Lifecycle Management policy to change the storage class to Archive for data older than 2 years.

​

Set an Object Lifecycle Management policy to delete data older than 2 years.

(Correct)

**Explanation**

Set an Object Lifecycle Management policy to change the storage class to Coldline for data older than 2 years. **is not right.**

Data older than 2 years is not needed so there is no point in transitioning the data to Coldline. The data needs to be deleted.

Set an Object Lifecycle Management policy to change the storage class to Archive for data older than 2 years. **is not right.**

Data older than 2 years is not needed so there is no point in transitioning the data to Archive. The data needs to be deleted.

Store infrequently accessed data in a Multi-Regional bucket. **is not right.**

While infrequently accessed data can be stored in Multi-Regional bucket, there are several other storage classes offered by Google Cloud Storage that are primarily aimed at storing infrequently accessed data and cost less. Multi-Region buckets are primarily used for achieving geo-redundancy.

Ref: https://cloud.google.com/storage/docs/locations

Set an Object Lifecycle Management policy to delete data older than 2 years. **is the right answer.**

Since you don't need data older than 2 years, deleting such data is the right approach. You can set a lifecycle policy to automatically delete objects older than 2 years. The policy is valid on current as well as future objects and doesn't need any human intervention.

Ref: https://cloud.google.com/storage/docs/lifecycle

Store infrequently accessed data in a Nearline bucket. **is the right answer.**

Nearline Storage is a low-cost, highly durable storage service for storing infrequently accessed data. Nearline Storage is ideal for data you plan to read or modify on average once per month or less.

Ref: https://cloud.google.com/storage/docs/storage-classes#nearline

Bottom of Form

Top of Form

Question 31: **Incorrect**

**You want to migrate a public NodeJS application, which serves requests over HTTPS, from your on-premises data centre to Google Cloud Platform. You plan to host it on a fleet of instances behind Managed Instances Group (MIG) in Google Compute Engine. You need to configure a GCP load balancer to terminate SSL session and pass HTTP traffic to the VMs. Which GCP Load balancer should you use?**

​

Configure an internal TCP load balancer.

​

Configure an external SSL proxy load balancer.

(Incorrect)

​

Configure an external TCP proxy load balancer.

​

Configure an HTTP(S) load balancer.

(Correct)

**Explanation**

Configure an internal TCP load balancer. **is not right.**

Internal TCP Load Balancing is a regional load balancer that enables you to run and scale your services behind an internal load balancing IP address that is accessible only to your internal virtual machine (VM) instances. Since we need to serve public traffic, this load balancer is not suitable for us.

Ref: https://cloud.google.com/load-balancing/docs/internal

Configure an external SSL proxy load balancer. **is not right.**

Google says "SSL Proxy Load Balancing is intended for non-HTTP(S) traffic. For HTTP(S) traffic, we recommend that you use HTTP(S) Load Balancing."

So this option can be ruled out.

Ref: https://cloud.google.com/load-balancing/docs/ssl

Configure an external TCP proxy load balancer. **is not right.**

Google says "TCP Proxy Load Balancing is intended for non-HTTP traffic. For HTTP traffic, use HTTP Load Balancing instead. For proxied SSL traffic, use SSL Proxy Load Balancing." So this option can be ruled out.

Ref: https://cloud.google.com/load-balancing/docs/tcp

Configure an HTTP(S) load balancer. **is the right answer.**

This is the only option that fits all requirements. It can serve public traffic, can terminate SSL at the load balancer and follows google recommended practices.

● "The backends of a backend service can be either instance groups or network endpoint groups (NEGs), but not a combination of both."

● "An external HTTP(S) load balancer distributes traffic from the internet"

● "The client SSL session terminates at the load balancer."

● "For HTTP traffic, use HTTP Load Balancing instead."

Ref: https://cloud.google.com/load-balancing/docs/https

Bottom of Form

Top of Form

Question 32: **Incorrect**

**Your company owns a mobile game that is popular with users all over the world. The mobile game backend uses Cloud Spanner to store user state. An overnight job exports user state to a Cloud Storage bucket. The app pushes all time-series events during the game to a streaming Dataflow service that saves them to Cloud Bigtable. You are debugging an in-game issue raised by a gaming customer, and you want to join the user state information with data stored in Bigtable to debug. How can you do this one-off join efficiently?**

​

Create two separate BigQuery external tables on Cloud Storage and Cloud Bigtable. Use the BigQuery console to join these tables through user fields, and apply appropriate filters.

(Correct)

​

Create a Cloud Dataproc cluster that runs a Spark job to extract data from Cloud Bigtable and Cloud Storage for specific users.

(Incorrect)

​

Create a dataflow job that copies data from Cloud Bigtable and Cloud Spanner for specific users.

​

Create a dataflow job that copies data from Cloud Bigtable and Cloud Storage for specific users.

**Explanation**

Our requirements are to **join** user sessions with user events **efficiently**. We need to look for an option that is primarily a Google service and provides this feature out of the box or with minimal configuration.

Create two separate BigQuery external tables on Cloud Storage and Cloud Bigtable. Use the BigQuery console to join these tables through user fields, and apply appropriate filters **is the right answer.**

Big query lets you create tables that reference external data sources such as Bigtable and Cloud Storage. You can then join up these two tables through user fields and apply appropriate filters. You can achieve the end result with minimal configuration using this option.

Ref: https[://cloud.google.com/bigquery/external-data-sources](https://cloud.google.com/bigquery/external-data-sources)

Create a dataflow job that copies data from Cloud Bigtable and Cloud Spanner for specific users. **is not right.**

You can make use of the Cloud Dataflow connector for Cloud Spanner (<https://cloud.google.com/spanner/docs/dataflow-connector>) and Dataflow Connector for Cloud Bigtable (<https://cloud.google.com/bigtable/docs/hbase-dataflow-java>) to retrieve data from these sources but you can’t use Dataflow SQL to restrict the result set to **specific users**. Dataflow SQL only works when reading data from [Pub/Sub topics](https://cloud.google.com/dataflow/docs/guides/sql/data-sources-destinations#reading-pubsub-topic), [Cloud Storage file sets](https://cloud.google.com/dataflow/docs/guides/sql/data-sources-destinations#reading-gcs-fileset), and [BigQuery tables](https://cloud.google.com/dataflow/docs/guides/sql/data-sources-destinations#reading-bigquery-table).

Ref: <https://cloud.google.com/dataflow/docs/guides/sql/data-sources-destinations>

Create a Cloud Dataproc cluster that runs a Spark job to extract data from Cloud Bigtable and Cloud Storage for specific users. **is not right.**

While it is certainly possible to do this using a Spark job, it is complicated as we would have to come up with the code/logic to extract the data and certainly not straightforward.

Create a dataflow job that copies data from Cloud Bigtable and Cloud Storage for specific users. **is not right.**

This is possible but it is **not as efficient** as using Big Query.

Ref: https[://cloud.google.com/dataflow/docs/guides/sql/dataflow-sql-intro](https://cloud.google.com/dataflow/docs/guides/sql/dataflow-sql-intro)

Here is some more documentation around this option, some of the issues are

1. Dataflow SQL expects CSV files in Cloud Storage filesets. CSV files must not contain a header row with column names; the first row in each CSV file is interpreted as a data record. - but our question doesn’t say how the exported data is stored in cloud storage.

2. You can only run jobs in regions that have a Dataflow regional endpoint. Our question doesn’t say which region. Ref: https[://cloud.google.com/dataflow/docs/concepts/regional-endpoints](https://cloud.google.com/dataflow/docs/concepts/regional-endpoints).

3. Creating a Dataflow job can take several minutes - unlike Big Query external tables which can be created very easily.

**Too many unknowns**. Otherwise, this option is a good option.

Here is some more information if you’d like to get a better understanding of how to use Cloud Dataflow to achieve this result.

Cloud Dataflow SQL lets you use SQL queries to develop and run Dataflow jobs from the BigQuery web UI. You can join streams (such as Pub/Sub) and snapshotted datasets (such as BigQuery tables and Cloud Storage filesets); query your streams or static datasets with SQL by associating schemas with objects, such as tables, Cloud Storage filesets and Pub/Sub topics; and write your results into a BigQuery table for analysis and dashboarding.

Cloud Dataflow SQL supports multiple data sources including Cloud Storage and Big Query tables which are of interest for this scenario.

<https://cloud.google.com/dataflow/docs/guides/sql/data-sources-destinations>

Bottom of Form

Top of Form

Question 33: **Correct**

**To facilitate disaster recovery, your company saves database backup tar files in Cloud Storage bucket. You want to minimize the cost. Which GCP Cloud Storage class should you recommend?**

​

Multi-Regional Storage

​

Nearline Storage

​

Coldline Storage

(Correct)

​

Regional Storage

**Explanation**

The ideal answer to this would have been **Archive Storage** but that is not one of the options.

Archive Storage is the lowest-cost, highly durable storage service for data archiving, online backup, and disaster recovery. Your data is available within milliseconds, not hours or days.

https://cloud.google.com/storage/docs/storage-classes#archive

In the absence of **Archive Storage**, the next best option for storing backups is Coldline Storage.

**Coldline Storage** is a very-low-cost, highly durable storage service for storing infrequently accessed data. Coldline Storage is a better choice than Standard Storage or Nearline Storage in scenarios where slightly lower availability, a 90-day minimum storage duration, and higher costs for data access are acceptable trade-offs for lowered at-rest storage costs. Coldline Storage is ideal for data you plan to read or modify at most once a quarter.

Ref: https://cloud.google.com/storage/docs/storage-classes#coldline

Although Nearline, Regional and Multi-Regional can also be used to store the backups, they are expensive in comparison and Google recommends we use Coldline for backups.

More information about Nearline: https://cloud.google.com/storage/docs/storage-classes#nearline

More information about Standard/Regional: https://cloud.google.com/storage/docs/storage-classes#standard

More information about Standard/Multi-Regional: https://cloud.google.com/storage/docs/storage-classes#standard

Bottom of Form

Top of Form

Question 34: **Correct**

**Your company specializes in helping other companies detect if any pages on their website do not align to the specified standards. To do this, your company has deployed a custom C++ application in your on-premises data centre that crawls all the web pages of a customer’s website, compares the headers and template to the expected standard and stores the result before moving on to another customer’s website. This testing takes a lot of time and has resulted in it missing out on the SLA several times recently. The application team is aware of the slow processing time and knows the fix is to run the application on multiple virtual machines to split the load, but there is no free space in the data centre. You have been asked to identify if it is possible to migrate this application to Google cloud, ensuring it can autoscale with little to no changes to the application code. What GCP service should you recommend?**

​

Use Google App Engine and Google Stackdriver for logging.

​

Use Google Compute Engine unmanaged instance groups with a Network Load Balancer.

​

Use Google Cloud Dataproc and run Apache Hadoop jobs to process each test.

​

Use Google Compute Engine managed instance groups and autoscaling.

(Correct)

**Explanation**

Use Google Compute Engine unmanaged instance groups with a Network Load Balancer. **is not right.**

An unmanaged instance group is a collection of virtual machines (VMs) that reside in a single zone, VPC network, and subnet. An unmanaged instance group is useful for grouping together VMs that require individual configuration settings or tuning. Unmanaged instance group does not autoscale, so it does not help reduce the amount of time it takes to fully test a change to the system.

Ref: https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-unmanaged-instances

Use Google App Engine and Google Stackdriver for logging. **is not right.**

App Engine supports many popular languages like Java, PHP, Node.js, Python, C#, .Net, Ruby, and Go. However, C++ isn’t supported by App Engine.

Ref: https://cloud.google.com/appengine

Use Google Cloud Dataproc and run Apache Hadoop jobs to process each test. **is not right.**

Cloud Dataproc is a fast, easy-to-use, fully managed cloud service for running Apache Spark and Apache Hadoop clusters in a simpler, more cost-efficient way. While Dataproc is very efficient at processing ETL and Big Data pipelines, it is not as suitable for running a ruby application that runs tests each day.

Ref: https://cloud.google.com/dataproc

Use Google Compute Engine managed instance groups and autoscaling. **is the right answer.**

A managed instance group (MIG) contains identical virtual machine (VM) instances that are based on an instance template. MIGs support auto-healing, load balancing, autoscaling, and auto-updating. Managed instance groups offer auto-scaling capabilities that let you automatically add or delete instances from a managed instance group based on increases or decreases in load. Autoscaling helps your apps gracefully handle increases in traffic and reduce costs when the need for resources is lower. Autoscaling works by adding more instances to your instance group when there is more load (upscaling), and deleting instances when the need for instances is lowered (downscaling).

Ref: https://cloud.google.com/compute/docs/autoscaler/

Bottom of Form

Top of Form

Question 35: **Correct**

**You have two Kubernetes resource configuration files.**

***deployments.yaml* - creates a deployment**

***service.YAML* - sets up a LoadBalancer service to expose the pods.**

**You don't have a GKE cluster in the development project and you need to provision one. Which of the commands below would you run in Cloud Shell to create a GKE cluster and deploy the YAML configuration files to create a deployment and service?**

​

gcloud container clusters create cluster-1 --zone=us-central1-a

gcloud container clusters get-credentials cluster-1 --zone=us-central1-a

kubectl apply -f deployment.yaml

kubectl apply -f service.yaml

(Correct)

​

gcloud container clusters create cluster-1 --zone=us-central1-a

gcloud container clusters get-credentials cluster-1 --zone=us-central1-a

kubectl deploy -f deployment.yaml

kubectl deploy -f service.yaml

​

gcloud container clusters create cluster-1 --zone=us-central1-a

gcloud container clusters get-credentials cluster-1 --zone=us-central1-a

gcloud gke apply -f deployment.yaml

gcloud gke apply -f service.yaml

​

kubectl container clusters create cluster-1 --zone=us-central1-a

kubectl container clusters get-credentials cluster-1 --zone=us-central1-a

kubectl apply -f deployment.yaml

kubectl apply -f service.yaml

**Explanation**

kubectl container clusters create cluster-1 --zone=us-central1-a

kubectl container clusters get-credentials cluster-1 --zone=us-central1-a

kubectl apply -f deployment.yaml

kubectl apply -f service.yaml.

**is not right.**

kubectl doesn't support *kubectl container clusters create* command. kubectl can not be used to create GKE clusters. To create a GKE cluster, you need to execute *gcloud container clusters create* command.

Ref: https://cloud.google.com/sdk/gcloud/reference/container/clusters/create

gcloud container clusters create cluster-1 --zone=us-central1-a

gcloud container clusters get-credentials cluster-1 --zone=us-central1-a

kubectl deploy -f deployment.yaml

kubectl deploy -f service.yaml.

**is not right.**

kubectl doesn't support *kubectl deploy* command. The YAML file contains the cluster resource configuration. You don't create the configuration, instead, you apply the configuration to the cluster. The configuration can be applied by running *kubectl apply* command

Ref: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#apply

gcloud container clusters create cluster-1 --zone=us-central1-a

gcloud container clusters get-credentials cluster-1 --zone=us-central1-a

gcloud gke apply -f deployment.yaml

gcloud gke apply -f service.yaml.

**is not right.**

gcloud doesn't support *gcloud gke apply* command. The YAML file contains the cluster resource configuration. You don't create the configuration, instead, you apply the configuration to the cluster. The configuration can be applied by running *kubectl apply* command

Ref: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#apply

gcloud container clusters create cluster-1 --zone=us-central1-a

gcloud container clusters get-credentials cluster-1 --zone=us-central1-a

kubectl apply -f deployment.yaml

kubectl apply -f service.yaml.

**is the right answer.**

You create a cluster by running *gcloud container clusters create* command. You then fetch credentials for a running cluster by running *gcloud container clusters get-credentials* command. Finally, you apply the Kubernetes resource configuration by running *kubectl apply -f*

Ref: https://cloud.google.com/sdk/gcloud/reference/container/clusters/create

Ref: https://cloud.google.com/sdk/gcloud/reference/container/clusters/get-credentials

Ref: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#apply

Bottom of Form

Top of Form

Question 36: **Incorrect**

**You are working for a cryptocurrency startup, and you have enabled a link to the company’s Initial Coin Offering (ICO) whitepaper on the company website – which runs off Google Cloud Storage. Your CTO clicked on this link and got prompted to save the file to their desktop. The CTO thinks this is a poor user experience and has asked you to identify if it is possible to render the file directly in the browser for all users. What should you do?**

​

Set Content-Type metadata to application/pdf on the PDF file objects

(Correct)

​

Enable *Share publicly* on the PDF file objects

​

Enable Cloud CDN on the website frontend.

(Incorrect)

​

Add a label to the storage bucket with a key of Content-Type and a value of application/pdf.

**Explanation**

Set Content-Type metadata to application/pdf on the PDF file objects **is the right answer**.

**Content-Type** allows browsers to render the object properly. If the browser prompts users to save files to their machine, it is likely the browser does not see the Content-Type as application/pdf. Setting this would ensure the browser displays PDF files within the browser instead of popping up a download dialog.

Ref: https://cloud.google.com/storage/docs/gsutil/addlhelp/WorkingWithObjectMetadata#content-type\_1

Enable Cloud CDN on the website frontend. **is not right.**

CDN helps with caching content at the edge but doesn't help the browser in displaying pdf files.

Enable Share publicly on the PDF file objects. **is not right.**

The fact that the browser lets users download the file suggests the browser is able to reach out and download the file. Sharing publicly wouldn't make any difference.

Add a label to the storage bucket with a key of Content-Type and a value of application/pdf. **is not right.**

Bucket labels are key: value metadata pairs that allow you to group your buckets along with other Google Cloud resources such as virtual machine instances and persistent disks. They don't determine the file's content type.

Bottom of Form

Top of Form

Question 37: **Correct**

**Your organization is planning the infrastructure for a new large-scale application that will need to store anything between 200 TB to a petabyte of data in NoSQL format for Low-latency read/write and High-throughput analytics. Which storage option should you use?**

​

Cloud Bigtable.

(Correct)

​

Cloud SQL.

​

Cloud Spanner.

​

Cloud Datastore.

**Explanation**

Cloud Spanner. **is not right.**

Cloud Spanner is not a NoSQL database. Cloud SQL is a fully-managed relational database service.

Ref: https://cloud.google.com/sql/docs

Cloud SQL. **is not right.**

Cloud SQL is not a NoSQL database. Cloud Spanner is a highly scalable, enterprise-grade, globally-distributed, and strongly consistent relational database service

Ref: https://cloud.google.com/spanner

Cloud Datastore. **is not right.**

While Cloud Datastore is a highly scalable NoSQL database, it can't handle petabyte-scale data.

https://cloud.google.com/datastore

Cloud Bigtable. **is the right answer.**

Cloud Bigtable is a petabyte-scale, fully managed NoSQL database service for large analytical and operational workloads.

Ref: https://cloud.google.com/bigtable/

Bottom of Form

Top of Form

Question 38: **Correct**

**Your company installs and manages several types of IoT devices all over the world. Events range from 50,000 to 500,000 messages a second. You want to identify the best solution for ingesting, storing and analyzing this data in GCP platform. What GCP services should you use?**

[Larger image](javascript:void(0))

​

Cloud Pub/Sub, Cloud Storage, BigQuery, Cloud Bigtable

​

Cloud Pub/Sub, Cloud Dataflow, Cloud Datastore, BigQuery

​

Cloud Pub/Sub, Cloud Dataflow, Cloud Bigtable, BigQuery

(Correct)

​

Firebase Messages, Cloud Pub/Sub, Cloud Spanner, BigQuery

**Explanation**

For box 1 where you want to ingest time series data, your best bet is **Cloud Pub/Sub**.

For box 2 where you want to process the data in pipelines, your best bet is **Cloud Dataflow**.

That leaves us with two remaining options, both have **BigQuery** as no 4. For (storage) 3, it is a choice between Bigtable and Datastore. Bigtable provides out of the box support for time series data. So using **Bigtable** for Storage is the right answer.

Ref: https[://cloud.google.com/bigtable/docs/schema-design-time-series](https://cloud.google.com/bigtable/docs/schema-design-time-series)

The answer is **Cloud Pub/Sub, Cloud Dataflow, Cloud Bigtable, BigQuery**

Bottom of Form

Top of Form

Question 39: **Incorrect**

**Your company stores sensitive PII data in a cloud storage bucket. The objects are currently encrypted by Google-managed keys. Your compliance department has asked you to ensure all current and future objects in this bucket are encrypted by customer-managed encryption keys. You want to minimize effort. What should you do?**

​

1. In the bucket advanced settings, select the customer-supplied key and then select a Cloud KMS encryption key.

2. Delete all existing objects and upload them again so they use the new customer-supplied key for encryption.

​

1. In the bucket advanced settings, select the Customer-managed key and then select a Cloud KMS encryption key.

2. Rewrite all existing objects using *gsutil* rewrite to encrypt them with the new Customer-managed key.

(Correct)

​

1. In the bucket advanced settings, select the Customer-managed key and then select a Cloud KMS encryption key.

2. Existing objects encrypted by Google-managed keys can still be decrypted by the new Customer-managed key.

​

1. Rewrite all existing objects using *gsutil rewrite* to encrypt them with the new Customer-managed key.

2. In the bucket advanced settings, select the Customer-managed key and then select a Cloud KMS encryption key.

(Incorrect)

**Explanation**

1. In the bucket advanced settings, select the Customer-managed key and then select a Cloud KMS encryption key.

2. Existing objects encrypted by Google-managed keys can still be decrypted by the new Customer-managed key. **is not right.**

While changing the bucket encryption to use the Customer-managed key ensures all new objects use this key, existing objects are still encrypted by the Google-managed key. This doesn't satisfy our compliance requirements. Moreover, the customer managed key can't decrypt objects created by Google-managed keys.

Ref: https://cloud.google.com/storage/docs/encryption/using-customer-managed-keys#add-default-key

1. In the bucket advanced settings, select the customer-supplied key and then select a Cloud KMS encryption key.

2. Delete all existing objects and upload them again so they use the new customer-supplied key for encryption. **is not right.**

The customer-supplied key is not an option when selecting the encryption method in the console. Moreover, we want to use customer-managed encryption keys and not customer-supplied encryption keys. This does not fit our requirements.

1. Rewrite all existing objects using *gsutil* rewrite to encrypt them with the new Customer-managed key.

2. In the bucket advanced settings, select the Customer-managed key and then select a Cloud KMS encryption key. **is not right.**

While changing the bucket encryption to use the Customer-managed key ensures all new objects use this key, rewriting existing objects before changing the bucket encryption would result in the objects being encrypted by the encryption method in use at that point - which is still Google-managed.

1. In the bucket advanced settings, select the Customer-managed key and then select a Cloud KMS encryption key.

2. Rewrite all existing objects using *gsutil* rewrite to encrypt them with the new Customer-managed key. **is the right answer.**

Changing the bucket encryption to use the Customer-managed key ensures all new objects use this key. Now that bucket encryption is changed to use the Customer-managed key, rewrite all existing objects using *gsutil* rewrite results in objects being encrypted by the new Customer-managed key. This is the only option that satisfies our requirements.

Ref: https://cloud.google.com/storage/docs/encryption/using-customer-managed-keys#add-default-key

Bottom of Form

Top of Form

Question 40: **Incorrect**

**You are developing a mobile game that uses Cloud Datastore for gaming leaderboards and player profiles. You want to test an aspect of this solution locally on your workstation which already has Cloud SDK installed. What should you do?**

​

Create a Cloud Datastore index using *gcloud datastore indexes create*

(Incorrect)

​

Export Cloud Datastore data using *gcloud datastore export*

​

Install the *google-cloud-sdk-datastore-emulator* component using the *apt get install* command.

​

Install the *cloud-datastore-emulator* component using the *gcloud components install* command.

(Correct)

**Explanation**

Export Cloud Datastore data using *gcloud datastore* export **is not right.**

By all means, you can export a copy of all or a subset of entities from Google Cloud Datastore to another storage system such as Google Cloud Storage but your application is configured to connect to a Cloud Datastore instance, not another system that stores a raw dump of exported data. So this option is not right.

Create a Cloud Datastore index using *gcloud datastore indexes create*. **is not right.**

You could create an index but this doesn't help your application emulate connections to Cloud Datastore on your laptop. So this option is not right.

Install the *google-cloud-sdk-datastore-emulator* component using the *apt get install* command. **is not right.**

There is no such thing as google-cloud-sdk-datastore-emulator; and you don't install gcloud components using apt get. So this option is not right.

Install the *cloud-datastore-emulator* component using the *gcloud components install* command. **is the right answer.**

The Datastore emulator provides local emulation of the production Datastore environment. You can use the emulator to develop and test your application locally

Ref: https://cloud.google.com/datastore/docs/tools/datastore-emulator

Bottom of Form

Top of Form

Question 41: **Incorrect**

**Your company has chosen to go serverless to enable developers to focus on writing code without worrying about infrastructure. You have been asked to identify a GCP Serverless service that does not limit your developers to specific runtimes. In addition, some of the applications need WebSockets support. What should you suggest?**

​

Cloud Run

​

App Engine Standard

(Incorrect)

​

Cloud Functions

​

Cloud Run for Anthos

(Correct)

**Explanation**

App Engine Standard. **is not right.**

Google App Engine Standard offers a limited number of runtimes - Java, Node.js, Python, Go, PHP and Ruby; and at the same time doesn’t offer support for Websockets.

Ref: https://cloud.google.com/appengine/docs/standard

Cloud Functions. **is not right.**

Like Google App Engine Standard, Cloud functions offer a limited number of runtimes - Node.js, Python, Go and Java; and doesn’t offer support for Websockets.

Ref: https://cloud.google.com/blog/products/application-development/your-favorite-runtimes-now-generally-available-on-cloud-functions

Cloud Run. **is not right.**

Cloud Run lets you run stateless containers in a fully managed environment. As this is container-based, we are not limited to specific runtimes. Developers can write code using their favorite languages (Go, Python, Java, C#, PHP, Ruby, Node.js, Shell, and others). However, Cloud Run does not support Websockets.

Ref: https://cloud.google.com/run

Cloud Run for Anthos. **is the right answer.**

Cloud Run for Anthos leverage Kubernetes and serverless together using Cloud Run integrated with Anthos. As this is container-based, we are not limited to specific runtimes. Developers can write code using their favorite languages (Go, Python, Java, C#, PHP, Ruby, Node.js, Shell, and others). Cloud Run for Anthos is the only serverless GCP offering that supports WebSockets.

https://cloud.google.com/serverless-options

Bottom of Form

Top of Form

Question 42: **Incorrect**

**You have annual audits every year and you need to provide external auditors access to the last 10 years of audit logs. You want to minimize the cost and operational overhead while following Google recommended practices. What should you do? (Select Three)**

​

Grant external auditors Storage Object Viewer role on the logs storage bucket.

(Correct)

​

Configure a lifecycle management policy on the logs bucket to delete objects older than 10 years

(Correct)

​

Export audit logs to BigQuery via an audit log export sink.

(Incorrect)

​

Set a custom retention of 10 years in Stackdriver logging and provide external auditors view access to Stackdriver Logs.

(Incorrect)

​

Export audit logs to Cloud Storage via an audit log export sink.

(Correct)

​

Export audit logs to Cloud Filestore via a Pub/Sub export sink.

**Explanation**

Export audit logs to Cloud Filestore via a Pub/Sub export sink. **is not right.**

Storing logs in Cloud Filestore is expensive. In Cloud Filestore, Standard Tier pricing costs $0.2 per GB per month and Premium Tier pricing costs $0.3 per GB per month. In comparison, Google Cloud Storage offers several storage classes that are significantly cheaper.

Ref: https://cloud.google.com/bigquery/pricing

Ref: https://cloud.google.com/storage/pricing

Set a custom retention of 10 years in Stackdriver logging and provide external auditors view access to Stackdriver Logs. **is not right.**

While it is possible to configure a custom retention period of 10 years in Stackdriver logging, storing logs in Stackdriver is expensive compared to Cloud Storage. Stackdriver charges $0.01 per GB per month, whereas something like Cloud Storage Coldline Storage costs $0.007 per GB per month (30% cheaper) and Cloud Storage Archive Storage costs 0.004 per GB per month (60% cheaper than Stackdriver)

Ref: https://cloud.google.com/logging/docs/storage#pricing

Ref: https://cloud.google.com/storage/pricing

Export audit logs to BigQuery via an audit log export sink. **is not right.**

Storing logs in BigQuery is expensive. In BigQuery, Active storage costs $0.02 per GB per month and Long-term storage costs $0.01 per GB per month. In comparison, Google Cloud Storage offers several storage classes that are significantly cheaper.

Ref: https://cloud.google.com/bigquery/pricing

Ref: https://cloud.google.com/storage/pricing

Export audit logs to Cloud Storage via an audit log export sink. **is the right answer.**

Among all the storage solutions offered by Google Cloud Platform, Cloud storage offers the best pricing for long term storage of logs. Google Cloud Storage offers several storage classes such as Nearline Storage ($0.01 per GB per Month) Coldline Storage ($0.007 per GB per Month) and Archive Storage ($0.004 per GB per month) which are significantly cheaper than the storage options covered by the above options above.

Ref: https://cloud.google.com/storage/pricing

Grant external auditors Storage Object Viewer role on the logs storage bucket. **is the right answer.**

You can provide external auditors access to the logs in the bucket by granting the Storage Object Viewer role which allows them to read any object stored in any bucket.

Ref: https://cloud.google.com/storage/docs/access-control/iam

Configure a lifecycle management policy on the logs bucket to delete objects older than 10 years. **is the right answer.**

You need to archive log files for 10 years but you don't need log files older than 10 years. And since you also want to minimize costs, it is a good idea to set up a lifecycle management policy on the bucket to delete objects that are older than 10 years. Livecycle management configuration is a set of rules which apply to current and future objects in the bucket. When an object meets the criteria of one of the rules, Cloud Storage automatically performs a specified action (delete in this case) on the object.

Ref: https://cloud.google.com/storage/docs/lifecycle

Bottom of Form

Top of Form

Question 43: **Correct**

**Your company, which specializes in the processing of genetically modified (GM) seeds, has most of its customers in South America. All compute workloads are in GCP southamerica-east1 region – which is the default region of both your production GCP projects – saproj1 and saproj2. You want to spin up two VM instances:**

**· a VM instance in southamerica-east1 region in saproj1 GCP project and,**

**· a VM instance in northamerica-northeast1 region in saproj2 GCP project.**

**You want to do this efficiently using gcloud CLI. What should you do?**

​

Create two configurations using *gcloud config configurations create [NAME]*. Run *gcloud config configurations activate [NAME]* to switch between accounts when running the commands to start the Compute Engine instances.

(Correct)

​

Create two configurations using *gcloud config configurations create [NAME]*. Run *gcloud configurations list* to start the Compute Engine instances.

​

Activate two configurations using *gcloud configurations activate [NAME]*. Run *gcloud configurations list* to start the Compute Engine instances.

​

Activate two configurations using *gcloud configurations activate [NAME]*. Run *gcloud config list* to start the Compute Engine instances.

**Explanation**

Create two configurations using *gcloud config configurations create [NAME]*. Run *gcloud configurations list* to start the Compute Engine instances. **is not right.**

*gcloud configurations list* is an invalid command. To list the existing named configurations, you need to execute *gcloud config configurations list* but this does not start the compute engine instances.

Ref: https://cloud.google.com/sdk/gcloud/reference/config/configurations/list

Activate two configurations using *gcloud configurations activate [NAME]*. Run *gcloud configurations list* to start the Compute Engine instances. **is not right.**

*gcloud configurations list* is an invalid command. To list the existing named configurations, you need to execute *gcloud config configurations list* but this does not start the compute engine instances.

Ref: https://cloud.google.com/sdk/gcloud/reference/config/configurations/list

Activate two configurations using *gcloud configurations activate [NAME]*. Run *gcloud config list* to start the Compute Engine instances. **is not right.**

*gcloud configurations activate [NAME]* activates an existing named configuration. It can't be used to activate two configurations at the same time. Moreover, *gcloud config list* lists Cloud SDK properties for the currently active configuration. It does not start the Compute Engine instances.

Ref: https://cloud.google.com/sdk/gcloud/reference/config/configurations/activate

Ref: https://cloud.google.com/sdk/gcloud/reference/config/list

Create two configurations using *gcloud config configurations create [NAME]*. Run *gcloud config configurations activate [NAME]* to switch between accounts when running the commands to start the Compute Engine instances. **is the right answer.**

Each gcloud configuration has a 1 to 1 relationship with the region (if a region is defined). Since we have two different regions, we would need to create two separate configurations using *gcloud config configurations create*

Ref: https://cloud.google.com/sdk/gcloud/reference/config/configurations/create

Secondly, you can activate each configuration independently by running *gcloud config configurations activate [NAME]*

Ref: https://cloud.google.com/sdk/gcloud/reference/config/configurations/activate

Finally, while each configuration is active, you can run the *gcloud compute instances start [NAME]* command to start the instance in the configuration's region.

https://cloud.google.com/sdk/gcloud/reference/compute/instances/start

Bottom of Form

Top of Form

Question 44: **Incorrect**

**You have a web application deployed as a managed instance group based on an instance template. You modified the startup script used in the instance template and would like the existing instances to pick up changes from the new startup scripts. Your web application is currently serving live web traffic. You want to propagate the startup script changes to all instances in the managed instances group while minimizing effort, minimizing cost and ensuring that the available capacity does not decrease. What would you do?**

​

Perform a *rolling-action replace* with *max-unavailable* set to 0 and *max-surge* set to 1

(Correct)

​

Create a new managed instance group (MIG) based on a new template. Add the group to the backend service for the load balancer. When all instances in the new managed instance group are healthy, delete the old managed instance group

(Incorrect)

​

Perform a *rolling-action start-update* with *max-unavailable* set to 1 and *max-surge* set to 0

​

Delete instances in the managed instance group (MIG) one at a time and rely on auto-healing to provision an additional instance.

**Explanation**

Perform a *rolling-action start-update* with *max-unavailable* set to 1 and *max-surge* set to 0. **is not right.**

You can carry out a rolling action start update to fully replace the template by executing a command like

*gcloud compute instance-groups managed rolling-action start-update instance-group-1 --zone=us-central1-a --version template=instance-template-1 --canary-version template=instance-template-2,target-size=100%*

which updates the instance-group-1 to use instance-template-2 instead of instance-template-1 and have instances created out of instance-template-2 serve 100% of traffic.

However, the values specified for *maxSurge* and *maxUnavailable* mean that we will lose capacity which is against our requirements.

*maxSurge* specifies the maximum number of instances that can be created over the desired number of instances. If *maxSurge* is set to 0, the rolling update can not create additional instances and is forced to update existing instances. This results in a reduction in capacity and therefore does not satisfy our requirement to ensure that the available capacity does not decrease during the deployment.

*maxUnavailable* - specifies the maximum number of instances that can be unavailable during the update process. When *maxUnavailable* is set to 1, the rolling update updates 1 instance at a time. i.e. it takes 1 instance out of service, updates it, and puts it back into service. This results in a reduction in capacity while the instance is out of service. Example - if we have 10 instances in service, this combination of setting results in 1 instance at a time taken out of service for replacement while the remaining 9 continue to serve live traffic. That’s a reduction of 10% in available capacity.

Ref: https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#max-unavailable

Ref: https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#max-surge

Create a new managed instance group (MIG) based on a new template. Add the group to the backend service for the load balancer. When all instances in the new managed instance group are healthy, delete the old managed instance group. **is not right.**

While the end result is the same, we have a period of time where the traffic is served by instances from both the old managed instances group (MIG) which doubles our cost and increases effort and complexity.

Delete instances in the managed instance group (MIG) one at a time and rely on auto-healing to provision an additional instance. **is not right.**

While this would result in the same eventual outcome, there are two issues with this approach. First, deleting an instance one at a time would result in a reduction in capacity which is against our requirements. Secondly, deleting instances manually one at a time is error-prone and time-consuming. One of our requirements is to "minimize the effort" but deleting instances manually and relying on auto-healing health checks to provision them back is time-consuming and could take a lot of time depending on the number of instances in the MIG and the startup scripts executed during bootstrap.

Perform a *rolling-action replace* with *max-unavailable* set to 0 and *max-surge* set to 1. **is the right answer.**

This option achieves the outcome in the most optimal manner. The replace action is used to replace instances in a managed instance group. When *maxUnavailable* is set to 0, the rolling update can not take existing instances out of service. And when *maxSurge* is set to 1, we let the rolling update spin a single additional instance. The rolling update then puts the additional instance into service and takes one of the existing instances out of service for replacement. There is no reduction in capacity at any point in time.

Ref: https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#max-unavailable

Ref: <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#max-surge>

Ref: <https://cloud.google.com/sdk/gcloud/reference/alpha/compute/instance-groups/managed/rolling-action/replace>

Bottom of Form

Top of Form

Question 45: **Incorrect**

**A production application serving live traffic needs an important update deployed gradually. The application is in US-Central region in a managed instance group. The application receives millions of requests each minute, and you want to patch the application while ensuring the number of instances (capacity) in the Managed Instance Group (MIG) does not decrease. What should you do?**

​

Perform a *rolling-action start-update* with *maxSurge* set to 1 and *maxUnavailable* set to 0

(Correct)

​

Create a new instance template with the new application version. Update the existing managed instance group with the new instance template. Delete the instances in the managed instance group to allow the managed instance group to recreate the instance using the new instance template.

(Incorrect)

​

Perform a *rolling-action start-update* with *maxSurge* set to 0 and *maxUnavailable* set to 1

​

Create a new managed instance group with an updated instance template. Add the group to the backend service for the load balancer. When all instances in the new managed instance group are healthy, delete the old managed instance group.

**Explanation**

Our requirements are

1. Deploy a new version **gradually**

2. Ensure **available capacity does not decrease** during deployment

Create a new managed instance group with an updated instance template. Add the group to the backend service for the load balancer. When all instances in the new managed instance group are healthy, delete the old managed instance group. **is not right**.

First of all instance templates can not be updated. So the phrase *updated instance template* rules out this option.

Ref: https[://cloud.google.com/compute/docs/instance-templates/](https://cloud.google.com/compute/docs/instance-templates/)

Create a new instance template with the new application version. Update the existing managed instance group with the new instance template. Delete the instances in the managed instance group to allow the managed instance group to recreate the instance using the new instance template. **is not right.**

If we follow these steps, we end up with a full fleet of instances belonging to the new managed instances group (i.e. based on the new template) behind the load balancer, but our requirement to **gradually** deploy the new version is not met. In addition, deleting the existing instances of the managed instance group would almost certainly result in an outage to our application which is not desirable when we are **serving live web traffic**.

Perform a *rolling-action start-update* with *maxSurge* set to 0 and *maxUnavailable* set to 1 **is not right.**

*maxSurge* specifies the maximum number of instances that can be created over the desired number of instances. If *maxSurge* is set to 0, the rolling update can not create additional instances and is forced to update existing instances. This results in a reduction in capacity and therefore does not satisfy our requirement to ensure that the **available capacity does not decrease** during the deployment**.**

*maxUnavailable* - specifies the maximum number of instances that can be unavailable during the update process. When *maxUnavailable* is set to 1, the rolling update updates 1 instance at a time. i.e. it takes 1 instance out of service, updates it, and puts it back into service. This results in a **reduction in capacity** while the instance is out of service. Example - if we have 10 instances in service, this combination of setting results in 1 instance at a time taken out of service for an upgrade while the remaining 9 continue to serve live traffic. That’s a reduction of 10% in available capacity and does not satisfy our requirement to ensure that the **available capacity does not** decrease during the deployment**.**

Perform a *rolling-action start-update* with *maxSurge* set to 1 and *maxUnavailable* set to 0 **is the right answer.**

This is the only option that satisfies our two requirements - **deploying gradually** and **ensuring the available capacity does not decrease.** When maxUnavailable is set to 0, the rolling update can not take existing instances out of service. And when maxSurge is set to 1, we let the rolling update spin a single additional instance. The rolling update then puts the additional instance into service and takes one of the existing instances out of service for the upgrade. There is **no reduction in capacity** at any point in time. And the rolling upgrade upgrades 1 instance at a time so we **gradually deploy** the new version. Example - if we have 10 instances in service, this combination of setting results in 1 additional instance put into service (resulting in 11 instances serving traffic), then a older instance taken out of service (resulting in 10 instances serving traffic) and puts the upgraded instance back into service (resulting in 11 instances serving traffic). The rolling upgrade continues updating the remaining 9 instances one at a time. Finally, when all 10 instances have been upgraded, the additional instance that is spun up is deleted. We still have 10 instances serving live traffic but now on the new version of code.

Ref: https[://kubernetes.io/docs/concepts/workloads/controllers/deployment/#max-unavailable](https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#max-unavailable)

Ref: <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#max-surge>

Bottom of Form

Top of Form

Question 46: **Correct**

**You recently deployed a new application in Google App Engine to serve production traffic. After analyzing logs for various user flows, you uncovered several issues in your application code and have developed a fix to address the issues. Parts of your proposed fix could not be validated in the pre-production environment by your testing team as some of the scenarios can only be validated by an end-user with access to specific data in your production environment. In the company's weekly Change Approval Board meeting, concerns were raised that the fix could possibly take down the application. It was unanimously agreed that while the fix is risky, it is a necessary change to the application. You have been asked to suggest a solution that minimizes the impact of the change going wrong. You also want to minimize costs. What should you do?**

​

Deploy the new application version temporarily, capture logs and then roll it back to the previous version.

​

Create a second Google App Engine project with the new application code, and onboard users gradually to the new application.

​

Deploy a new version of the application, and use traffic splitting to send a small percentage of traffic to it.

(Correct)

​

Set up a second Google App Engine service, and then update a subset of clients to hit the new service.

**Explanation**

Deploy the new application version temporarily, capture logs and then roll it back to the previous version. **is not right.**

Deploying a new application version and promoting it would result in your new version serving all production traffic. If the code fix doesn't work as expected, it would result in the application becoming unreachable to all users. This is a risky approach and should be avoided.

Create a second Google App Engine project with the new application code, and onboard users gradually to the new application. **is not right.**

You want to minimize costs. This approach effectively doubles your costs as you have to pay for two identical environments until all users are moved over to the new application. There is an additional overhead of manually onboarding users to the new application which could be expensive as well as time-consuming.

Set up a second Google App Engine service, and then update a subset of clients to hit the new service. **is not right.**

It is not straightforward to update a set of clients to hit the new service. When users access an App Engine service, they use an endpoint like https://SERVICE\_ID-dot-PROJECT\_ID.REGION\_ID.r.appspot.com. Introducing a new service introduces a new URL and getting your users to use the new URL is possible but involves effort and coordination. If you want to mask these differences to the end-user, then you have to make changes in the DNS and use a weighted algorithm to split the traffic between the two services based on the weights assigned.

Ref: https://cloud.google.com/appengine/docs/standard/python/splitting-traffic

Ref: https://cloud.google.com/appengine/docs/standard/python/an-overview-of-app-engine

This approach also has the drawback of doubling your costs until all users are moved over to the new service.

Deploy a new version of the application, and use traffic splitting to send a small percentage of traffic to it. **is the right answer.**

This option minimizes the risk to the application while also minimizing the complexity and cost. When you deploy a new version to App Engine, you can choose not to promote it to serve live traffic. Instead, you could set up traffic splitting to split traffic between the two versions - this can all be done within Google App Engine. Once you send a small portion of traffic to the new version, you can analyze logs to identify if the fix has worked as expected. If the fix hasn't worked, you can update your traffic splitting configuration to send all traffic back to the old version. If you are happy your fix has worked, you can send more traffic to the new version or move all user traffic to the new version and delete the old version.

Ref: https://cloud.google.com/appengine/docs/standard/python/splitting-traffic

Ref: https://cloud.google.com/appengine/docs/standard/python/an-overview-of-app-engine

Bottom of Form

Top of Form

Question 47: **Correct**

**You are designing a mobile game which you hope will be used by numerous users around the world. The game backend requires a Relational DataBase Management System (RDBMS) for persisting game state and player profiles. You want to select a database that can scale to a global audience with minimal configuration updates. Which database should you choose?**

​

Cloud Datastore

​

Cloud Spanner

(Correct)

​

Cloud Firestore

​

Cloud SQL

**Explanation**

Our requirements are **relational data, global users, scaling**

Cloud Firestore **is not right.**

Cloud Firestore is not a relational database. Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform.

Ref: https://firebase.google.com/docs/firestore

Cloud Datastore **is not right.**

Cloud Datastore is not a relational database. Datastore is a NoSQL document database built for automatic scaling, high performance, and ease of application development

Ref: https://cloud.google.com/datastore/docs/concepts/overview

Cloud SQL **is not right.**

While Cloud SQL is a relational database, it does not offer infinite automated scaling with minimum configuration changes. Cloud SQL is a fully-managed database service that makes it easy to set up, maintain, manage, and administer your relational databases on Google Cloud Platform

Ref: https://cloud.google.com/sql/docs

Cloud Spanner **is the right answer.**

Cloud Spanner is a relational database and is highly scalable. Cloud Spanner is a highly scalable, enterprise-grade, globally-distributed, and strongly consistent database service built for the cloud specifically to combine the benefits of relational database structure with a non-relational horizontal scale. This combination delivers high-performance transactions and strong consistency across rows, regions, and continents with an industry-leading 99.999% availability SLA, no planned downtime, and enterprise-grade security

https://cloud.google.com/spanner

Bottom of Form

Top of Form

Question 48: **Correct**

**Your company’s new mobile game has gone live, and you have transitioned the backend application to the operations team. The mobile game uses Cloud Spanner to persist game state, leaderboard and player profile. Your operations team require access to view and edit table data to support runtime issues. What should you do?**

​

Run *gcloud iam roles describe roles/spanner.databaseUser*. Add the users to a new group. Add the group to the role.

(Correct)

​

Run *gcloud iam roles describe roles/spanner.viewer -- project my-gcp-ace-project*. Add the users to a new group. Add the group to the role.

​

Run *gcloud iam roles describe roles/spanner.viewer -- project my-gcp-ace-project*. Add the users to the role.

​

Run *gcloud iam roles describe roles/spanner.databaseUser*. Add the users to the role.

**Explanation**

Our requirements

1. View and **Edit** table data

2. **3 users** (i.e. multiple users)

3 users should give us the idea that we do not want to assign roles/permissions at the user level and instead want to do it based on groups so that we can create one group with all the required permissions and all such users who need this access can be assigned to the group.

Ref: https://cloud.google.com/iam/docs/reference/rest/v1/Policy#Binding

Ref: https://cloud.google.com/iam/docs/granting-changing-revoking-access#granting-gcloud-manual

Run *gcloud iam roles describe roles/spanner.databaseUser*. Add the users to the role. **is not right.**

We are looking for an option that assigns users to a group (in order to maximise reuse and minimize maintenance overhead). This option assigns users to the role so is not the right answer.

Run *gcloud iam roles describe roles/spanner.viewer -- project my-gcp-ace-project*. Add the users to the role.**is not right.**

We are looking for an option that assigns users to a group (in order to maximise reuse and minimize maintenance overhead). This option assigns users to the role so is not the right answer.

Run *gcloud iam roles describe roles/spanner.viewer -- project my-gcp-ace-project*. Add the users to a new group. Add the group to the role. **is not right.**

Adding users to a group and granting the role to the group is the right way forward. But the role used in this option is spanner.viewer which allows viewing all Cloud Spanner instances (but cannot modify instances), and allows viewing all Cloud Spanner databases (but cannot modify databases and cannot read from databases). Since we required edit access as well, this option is not right.

Ref: https://cloud.google.com/spanner/docs/iam

Run *gcloud iam roles describe roles/spanner.databaseUser*. Add the users to a new group. Add the group to the role. **is the right answer.**

Adding users to a group and granting the role to the group is the right way forward. In addition, we assign the role spanner.databaseUser which allows Read from and write to the Cloud Spanner database; execute SQL queries on the database, including DML and Partitioned DML; and View and update schema for the database. This is the only option that grants the right role to a group and assigns users to the group.

Bottom of Form

Top of Form

Question 49: **Correct**

**You want to create a new role and grant it to the SME team. The new role should provide your SME team BigQuery Job User and Cloud Bigtable User roles on all projects in the organization. You want to minimize operational overhead. You want to follow Google recommended practices. How should you create the new role?**

​

In GCP Console under IAM Roles, select both roles and combine them into a new custom role. Grant the role to the SME team group at the organization level.

(Correct)

​

Execute command *gcloud iam combineroles --global* to combine the 2 roles into a new custom role and grant them globally to SME team group.

​

In GCP Console under IAM Roles, select both roles and combine them into a new custom role. Grant the role to the SME team group at project. Use *gcloud iam promote-role* to promote the role to all other projects and grant the role in each project to the SME team group.

​

In GCP Console under IAM Roles, select both roles and combine them into a new custom role. Grant the role to the SME team group at project. Repeat this step for each project.

**Explanation**

We want to create a new role and grant it to a team. Since you want to minimize operational overhead, we need to grant it to a group - so that new users who join the team just need to be added to the group and they inherit all the permissions. Also, this team needs to have the role for all projects in the organization. And since we want to minimize the operational overhead, we need to grant it at the organization level so that all current projects, as well as future projects, have the role granted to them.

In GCP Console under IAM Roles, select both roles and combine them into a new custom role. Grant the role to the SME team group at project. Repeat this step for each project. **is not right.**

Repeating the step for all projects is a manual, error-prone and time-consuming task. Also, if any projects were to be created in the future, we have to repeat the same process again. This increases operational overhead.

In GCP Console under IAM Roles, select both roles and combine them into a new custom role. Grant the role to the SME team group at project. Use *gcloud iam promote-role* to promote the role to all other projects and grant the role in each project to the SME team group. **is not right.**

Repeating the step for all projects is a manual, error-prone and time-consuming task. Also, if any projects were to be created in the future, we have to repeat the same process again. This increases operational overhead.

Execute command *gcloud iam combine-roles --global* to combine the 2 roles into a new custom role and grant them globally to all. **is not right.**

There are several issues with this. *gcloud iam* command doesn't support the action *combine-roles*. Secondly, we don't want to grant the roles globally. We want to grant them to the SME team and no one else.

In GCP Console under IAM Roles, select both roles and combine them into a new custom role. Grant the role to the SME team group at the organization level. **is the right answer.**

This correctly creates the role and assigns the role to the group at the organization. When any new users join the team, the only additional task is to add them to the group. Also, when a new project is created under the organization, no additional human intervention is needed. Since the role is granted at the organization level, it automatically is granted to all the current and future projects belonging to the organization.

Bottom of Form

Top of Form

Question 50: **Correct**

**You host a production application in Google Compute Engine in the us-central1-a zone. Your application needs to be available 24\*7 all through the year. The application suffered an outage recently due to a Compute Engine outage in the zone hosting your application. Your application is also susceptible to slowness during peak usage. You have been asked for a recommendation on how to modify the infrastructure to implement a cost-effective and scalable solution that can withstand zone failures. What would you recommend?**

​

Use Unmanaged instance groups across multiple zones. Enable Autoscaling on the Unmanaged instance group.

​

Use Managed instance groups with preemptible instances across multiple zones. Enable Autoscaling on the Managed instance group.

​

Use Managed instance groups with instances in a single zone. Enable Autoscaling on the Managed instance group.

​

Use Managed instance groups across multiple zones. Enable Autoscaling on the Managed instance group.

(Correct)

**Explanation**

Use Managed instance groups with preemptible instances across multiple zones. Enable Autoscaling on the Managed instance group. **is not right.**

A preemptible VM runs at a much lower price than normal instances and is cost-effective. However, Compute Engine might terminate (preempt) these instances if it requires access to those resources for other tasks. Preemptible instances are not suitable for production applications that need to be available 24\*7.

Ref: https://cloud.google.com/compute/docs/instances/preemptible

Use Unmanaged instance groups across multiple zones. Enable Autoscaling on the Unmanaged instance group. **is not right.**

Unmanaged instance groups do not autoscale. An unmanaged instance group is simply a collection of virtual machines (VMs) that reside in a single zone, VPC network, and subnet. An unmanaged instance group is useful for grouping together VMs that require individual configuration settings or tuning.

Ref: https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-unmanaged-instances

Use Managed instance groups with instances in a single zone. Enable Autoscaling on the Managed instance group. **is not right.**

While enabling auto-scaling is a good idea, autoscaling would spin up instances in the same zone. Should there be a zone failure, all instances of the managed instance group would be unreachable and cause the application to be unreachable. Google recommends you distribute your resources across multiple zones to tolerate outages.

Ref: https://cloud.google.com/compute/docs/regions-zones

Use Managed instance groups across multiple zones. Enable Autoscaling on the Managed instance group. **is the right answer.**

Distribute your resources across multiple zones and regions to tolerate outages. Google designs zones to be independent of each other: a zone usually has power, cooling, networking, and control planes that are isolated from other zones, and most single failure events will affect only a single zone. Thus, if a zone becomes unavailable, you can transfer traffic to another zone in the same region to keep your services running.

Ref: https://cloud.google.com/compute/docs/regions-zones

In addition, a managed instance group (MIG) contains offers auto-scaling capabilities that let you automatically add or delete instances from a managed instance group based on increases or decreases in load. Autoscaling helps your apps gracefully handle increases in traffic and reduce costs when the need for resources is lower. Autoscaling works by adding more instances to your instance group when there is more load (upscaling), and deleting instances when the need for instances is lowered (downscaling).

Ref: https://cloud.google.com/compute/docs/autoscaler/

Bottom of Form