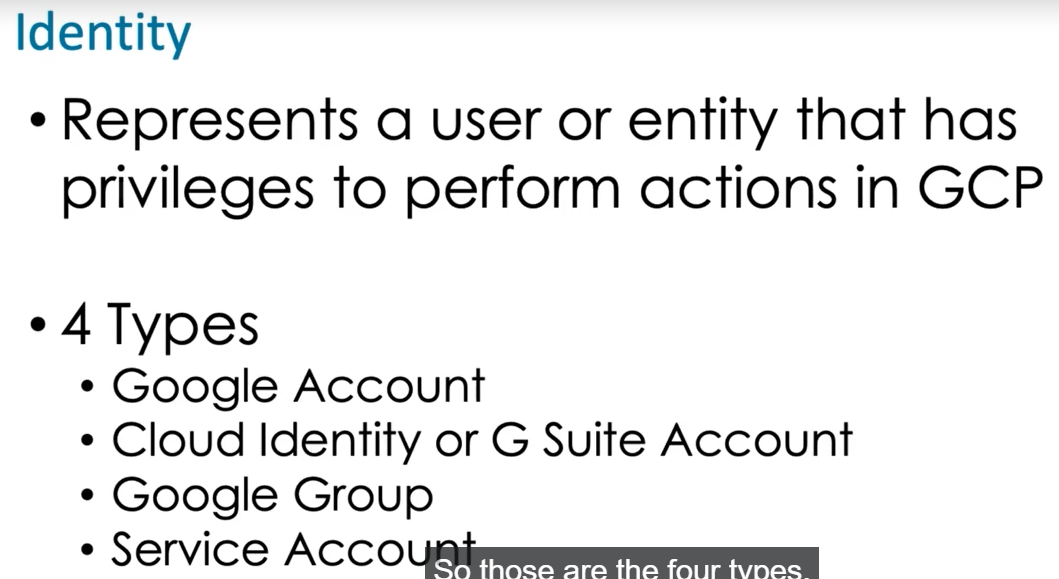
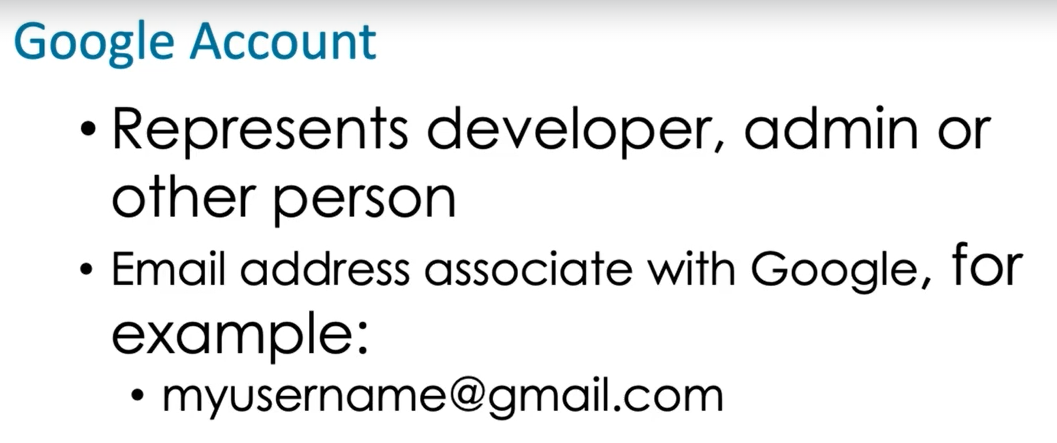
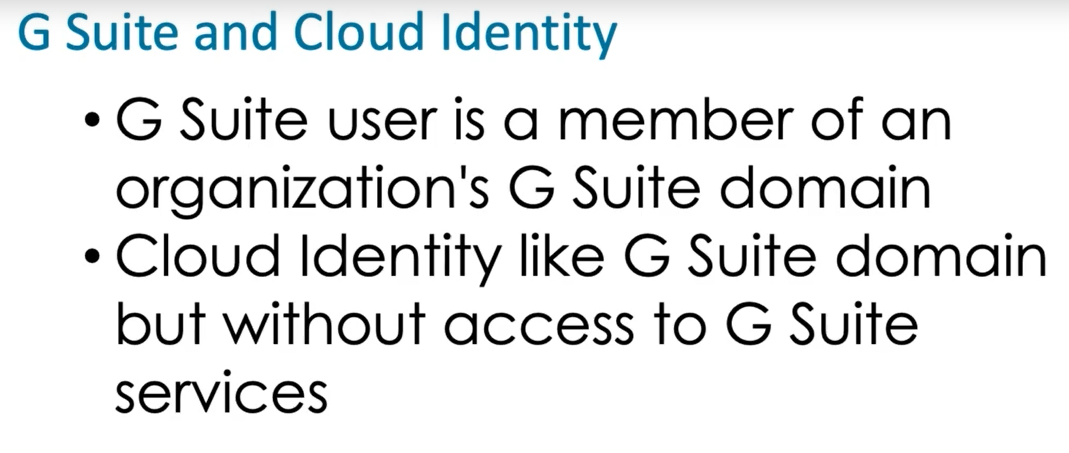
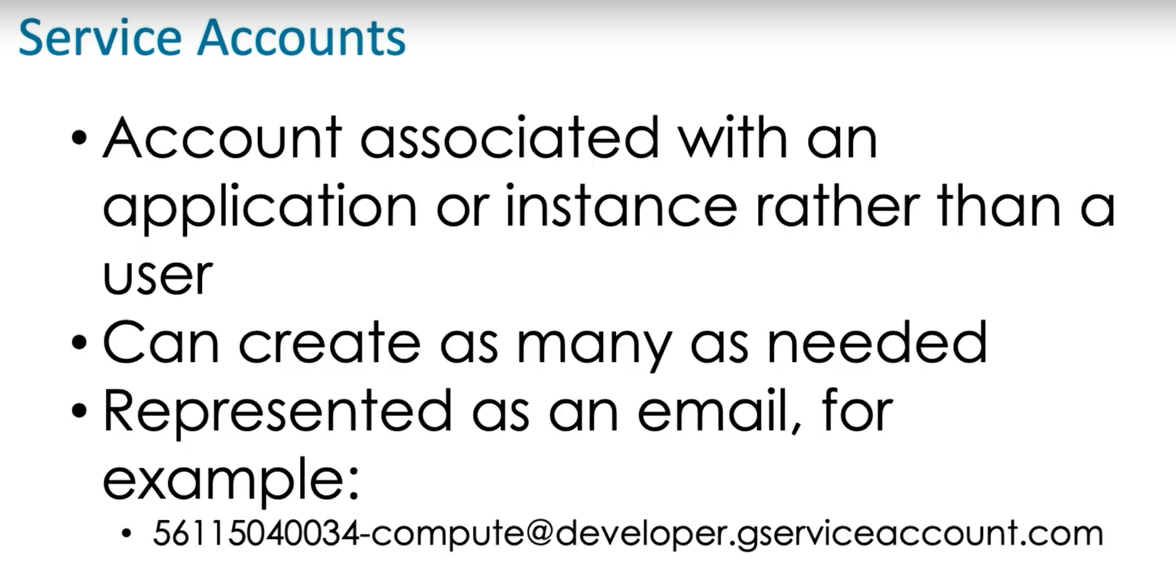
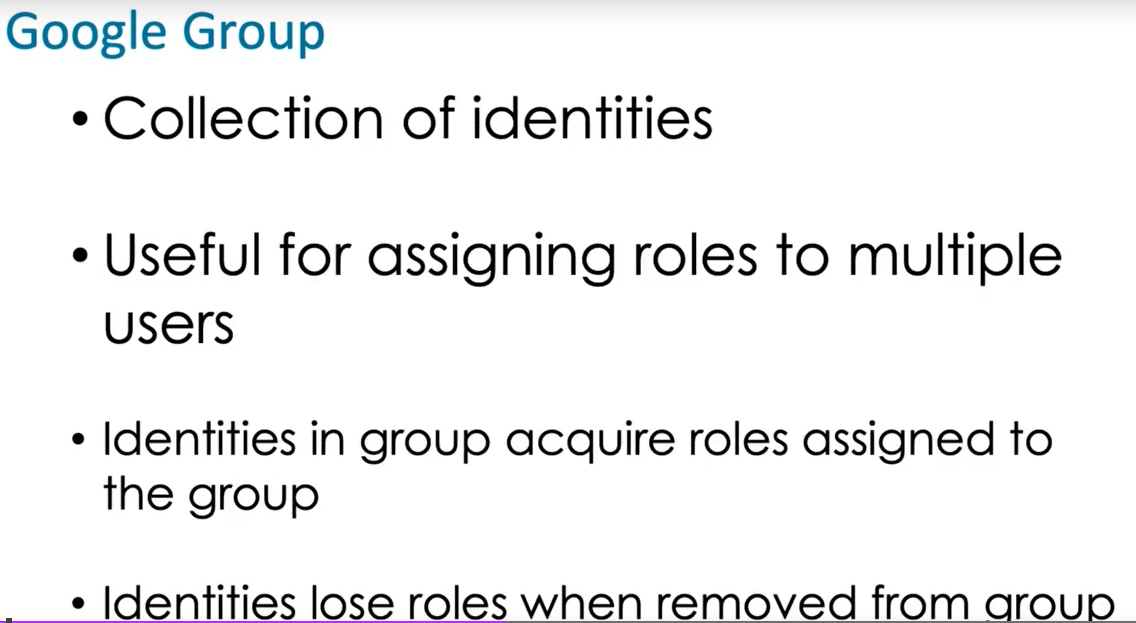
Who ?



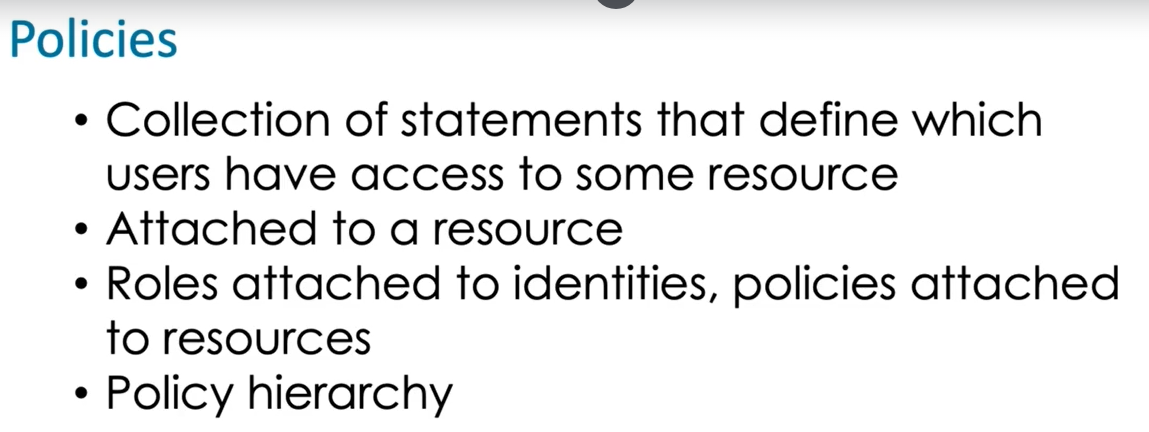












**Creating custom roles :**

Roles > Create roles > Title , Description , ID , role launch stage , add Permissions .

**Role launch stage -**

alpha , beta , (may not be finalized )

general availability , ("General Availability" means the roles is available for use, and the configuration of that role is not likely to change. )

disabled .(no longer used )

it's recommended that you use predefined roles. Now the predefined roles are roles that Google has created and they're based on common tasks that people perform at various IT environments. And one of the nice things about predefined roles is that they typically contain the minimal set of privileges that are needed to perform a particular task. So this helps us implement the principle of least privilege.

Another best practice is defining a resource hierarchy. And that means we specify an organization and a folder in project structure, and we then attach policies at the appropriate level so that we take advantage of inheritance.

It's recommended that you use Google groups and service accounts. Now remember, a group is a collection of users and it lets us assign roles to the group and then as we add and remove users to or from the group those users acquire the permissions that are associated with the roles that have been assigned to the group.

Also, it's a good practice to use different service accounts for separate functions.

So if you have a set of instances that are all running the same application then it's appropriate to use a single service account. However, if you have a backend service that runs a database and another one that runs some business logic and another server that's running front ends those are three distinct areas, those should have three distinct service accounts associated with them.

Another good practice is to use custom roles sparingly. Ideally, we use predefined roles,

and we only use custom roles when a predefined role doesn't exist that meets our needs.

Just a tip, this isn't really a best practice but it's just something to keep in mind is that you cannot assign permissions directly to an identity. Permissions are assigned to roles and roles are assigned to identities that's how an identity gets a particular permission.

Administrators, developers, and auditors should each be assigned to a group and roles should be assigned to those groups. Identities will acquire the permissions assigned to the roles when they are added to the group.

An identity represents an entity such as a person or service account that can be granted privileges by assigning roles to the identity. Identity types include Google accounts, Cloud Identity or G Suite accounts, Google groups, and service accounts.

A service account is designed to grant VMs and application roles that enable them to execute actions within the GCP. Google accounts and Google groups are valid identity types but they are designed for persons, not VMs or applications. There is no such thing as a batch account.

It is a best practice to use predefined roles. Predefined roles are designed to provide permissions needed to perform common tasks, such as administering databases. Custom roles should only be used if predefined roles do not meet your requirements. Primitive roles may be used in small teams where each member has broad responsibilities for administering GCP but otherwise using primitive roles is not recommended.

Folders are used to group resources that share common IAM policies. Service accounts are specific to a set of operating requirements within a project. Permissions are associated with roles but not directly with folders. IAM roles are granted to identities, not folders.

To load data, an identity must have bigquery.tables.create, bigquery.tables.updateData, and bigquery.jobs.create.

The role/bigtable.reader gives the data scientist the ability to read data but not write data or modify the database. This follows the Principle of Least Privilege as recommended by Google.

Service accounts are used to provide applications and instances with an identity that can have roles that give the identity sufficient permission to execute operations it needs to perform.

Cloud Identity provides domain verification records, which are added to DNS settings for the domain.

billing.accounts.create is the permission needed to create a billing account. Roles are sets of permissions but they are not permissions themselves.

Google Cloud's Identity and Access Management (IAM) service lets you create and manage permissions for Google Cloud resources. Cloud IAM unifies access control for Google Cloud services into a single system and provides a consistent set of operations.

There are four roles:

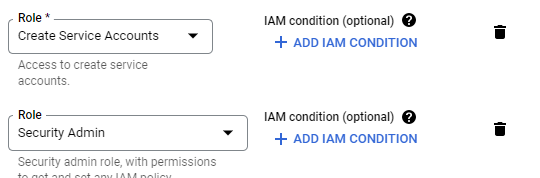
* Browser
* Editor
* Owner
* Viewer

These are *primitive roles* in Google Cloud. Primitive roles set project-level permissions and unless otherwise specified, they control access and management to all Google Cloud services.

The following table pulls definitions from the Google Cloud IAM article, [Basic roles](https://cloud.google.com/iam/docs/understanding-roles#primitive_roles), which gives a brief overview of browser, viewer, editor, and owner role permissions:

| **Role Name** | **Permissions** |
| --- | --- |
| roles/viewer | Permissions for read-only actions that do not affect state, such as viewing (but not modifying) existing resources or data. |
| roles/editor | All viewer permissions, plus permissions for actions that modify state, such as changing existing resources. |
| roles/owner | All editor permissions and permissions for the following actions:   * Manage roles and permissions for a project and all resources within the project. * Set up billing for a project. |
| roles/browser | Read access to browse the hierarchy for a project, including the folder, organization, and Cloud IAM policy. This role doesn't include permission to view resources in the project. |

To create a service account an identity must have below permissions .



IAM > Service account > Create

We can assign roles to service accounts and assign users who can manage service accounts .