

Google Cloud Platform

Access Control

BigQuery for Data Analysts

V1.2

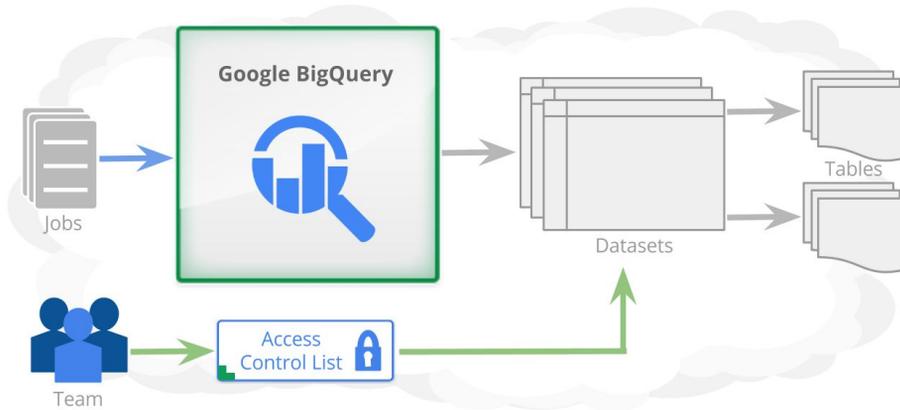
Approximate timing: 45 minutes

Agenda

- 1 Overview of Access Control Lists
- 2 Project and Dataset Roles
- 3 Applying Views for Row-Level Security
- 4 Identity and Access Management
- 5 Lab

Access Control Lists (1 of 2)

- ACLs define permissions given to a role (or grantee) for a target (project/dataset)



Access Control Lists (2 of 2)

- ACLs consist of one or more **entries** that grant **permission** to a **role** (or grantee)
- **Permissions** define the actions that can be performed against a project or dataset
 - **Scope** defines to *whom* the permission applies
- **Roles:**
 - *Project roles* – Users can run jobs or manage the project
 - *Dataset roles* – Define user access to datasets in a project

Agenda

- 1 Overview of Access Control Lists
- 2 Project and Dataset Roles
- 3 Applying Views for Row-Level Security
- 4 Identity and Access Management
- 5 Lab

Project Roles

- Granted/revoked using Cloud Platform Console
- Roles are assigned by email address for:
 - Individual users
 - Groups
 - Service accounts
- Project owners can modify project roles
 - Automatically granted to project creator

Permissions for Project Roles

Role	Permissions
Viewer	<ul style="list-style-type: none">● Can start a job - Dataset roles also required depending on job type● List and get all jobs <i>they started</i>● Granted READER dataset role by default for new datasets in project
Editor	<ul style="list-style-type: none">● Same as Viewer, plus:<ul style="list-style-type: none">○ Can create new dataset in project○ Is granted WRITER role by default for new datasets in project
Owner	<ul style="list-style-type: none">● Same as Editor, plus:<ul style="list-style-type: none">○ Can list all datasets in the project○ Can delete any dataset in the project○ Can list and get <i>all</i> jobs run

Dataset Roles (1 of 3)

- The project ACL becomes default ACL for datasets in the project
 - Default access can be overridden on a per-dataset basis
 - Tables inherit ACLs from dataset
 - ACLs cannot be configured on tables
- Dataset ACLs allow resource separation
 - No need for additional clusters and data duplication
 - Saves money, simplifies operations

Dataset Roles (2 of 3)

- Dataset roles are granted or revoked using:
 - The BigQuery web UI
 - Using the 'Share dataset' option
 - The BigQuery API
 - Using `Datasets:update`

Dataset Roles (3 of 3)

- Roles are assigned by email address to:
 - Single users
 - [Google Groups](#)
 - Predefined group of users, such as all users, or a group of users with same [project role](#)

Share Dataset

Who has access

Project Owners	Is owner ↕	×
Project Editors	Can edit ↕	×
Project Viewers	Can view ↕	×

Add people:

- User by e-mail
- Group by e-mail
- Domain
- All Authenticated Users
- Project Owners
- Project Viewers
- Project Editors
- Authorized View

Can view ↕ Add

missions.

Permissions for Dataset Roles

Role	Permissions
<i>Reader</i>	<ul style="list-style-type: none">• Can read, query, copy or export tables in the dataset<ul style="list-style-type: none">○ Can call <code>get</code> on the dataset and tables in dataset○ Can call <code>list</code> on table data for tables in the dataset
<i>Writer</i>	<ul style="list-style-type: none">• Same as READER, plus:<ul style="list-style-type: none">○ Can edit or append data in the dataset<ul style="list-style-type: none">■ Can call <code>insert</code>, <code>insertAll</code>, <code>update</code> or <code>delete</code>■ Can use tables in the dataset as destinations for load, copy or query jobs
<i>Owner</i>	<ul style="list-style-type: none">• Same as WRITER, plus:<ul style="list-style-type: none">○ Can call <code>update</code> on the dataset○ Can call <code>delete</code> on the dataset <p>Note: A dataset must have at least one entity with the OWNER role.</p>

Agenda

- 1 Overview of Access Control Lists
- 2 Project and Dataset Roles
- 3 Applying Views for Row-Level Security
- 4 Identity and Access Management
- 5 Lab

Row-Level Security (1 of 2)

- Not natively supported
 - Define a view to give access to a specific view of the data
- View – a BigQuery SQL query that limits the rows and columns (a virtual table) that a user can see
- BigQuery views are re-executed every time the view is queried
- Create view in dataset separate from underlying table's dataset and assign ACLs to both datasets

Notes:

Remind them that users who have access to a dataset have access to the entire dataset including all tables and views. BigQuery doesn't natively support application of row-level security. So to prevent the revealing of confidential or proprietary data, you have to apply a view and place it in its own dataset that your users may access. You set the ACL to allow users to access that dataset that contains the view while restricting access to the source table(s) dataset.

A view is a virtual table defined by a SQL query. You can query views in the browser tool, or by using a query job.

BigQuery's views are logical views, not materialized views, which means that the query that defines the view is re-executed every time the view is queried. Queries are billed according to the total amount of data in all table fields referenced directly or indirectly by the top-level query.

BigQuery supports up to eight levels of nested views; if there are more than eight levels, an `INVALID_INPUT` error returns. Also, views can only reference other tables and views with the same Dataset location. Querying a view requires the `READER` role for all datasets that contain the tables in the view chain. For more information about roles, see [access control](#).

Row-Level Security (2 of 2)

Google BigQuery

COMPOSE QUERY

Query History
Job History

Big Query Training Project

- ▶ flight_data
- ▶ reference
- ▼ Report_Views
 - Exec_Staff
- ▶ bigquery-e2e:backups
- ▶ bigquery-e2e:ch10
- ▶ bigquery-e2e:ch11
- ▶ bigquery-e2e:ch14

New Query ?

```
1 SELECT *  
2 FROM [Report_Views.Exec_Staff] LIMIT 1000
```

New dataset separate
from restricted table

RUN QUERY

itions Query complete

Query Results Oct 19, 2015, 12:00:54 PM

Table	JSON			
Row	Name	Position	Location	
1	Dan Hesse	Chief Executive Officer	San Jose	
2	Suzanne T. Raley	Chief Financial Officer	San Jose	

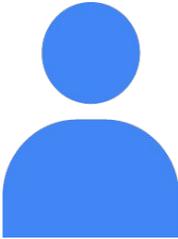
Notes:

The project owner would still need to set access permissions on the dataset.

Agenda

- 1 Overview of Access Control Lists
- 2 Project and Dataset Roles
- 3 Applying Views for Row-Level Security
- 4 Identity and Access Management
- 5 Lab

Identity and Access Management



Who



can do what



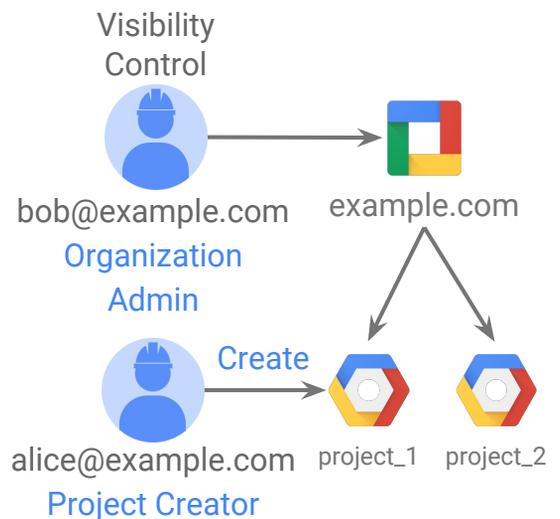
on which resource

Benefits of IAM for BigQuery

- Admins can isolate permissions to BigQuery
 - For example, BigQuery roles have no authority to manage Compute Engine virtual machines
- Consistent IAM controls across all GCP products
- Narrow permissions allow more fine-grained control
- Backwards compatible
 - Legacy project permissions are preserved, and the familiar UI, API, and CLI will continue to work as before with minimal changes

Organization Node **Beta**

- Organization node is root node for Google Cloud resources
- 2 organization roles:
 - *Organization Admin* - Control over all cloud resources
 - *Project Creator* - Controls project creation



Notes:

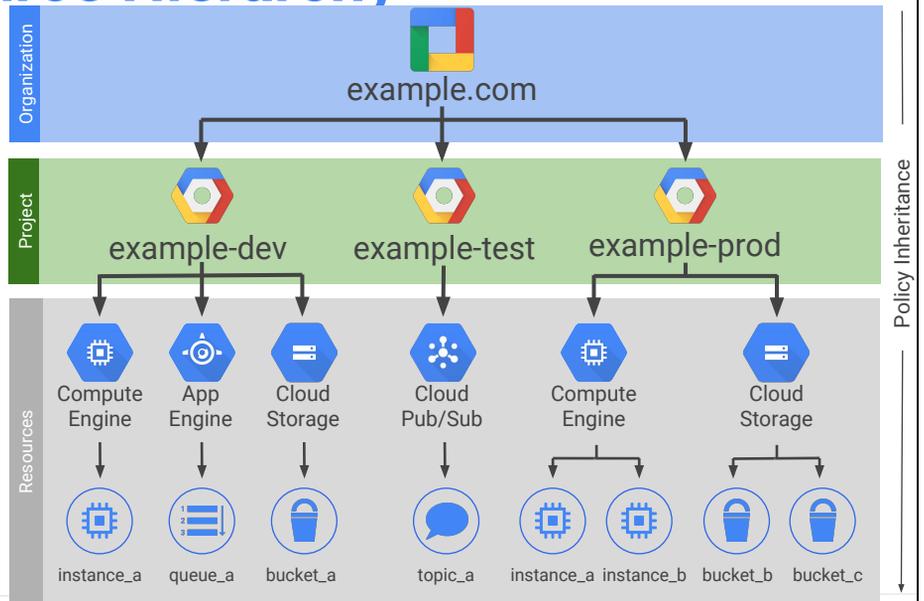
A large number of projects can become unwieldy to manage at scale. This is why IAM includes the concept of an Organization Node. The Organization Node sits above Projects and is your company's root node for Google Cloud resources. If you have a Google for Work account, when you enable the Organization Node, any project created by users in your domain will automatically belong to your Organization Node - no more shadow projects and no more rogue admins.

The Organization Admin role gives your admin visibility and control over all of your company's resources on Google Cloud Platform. Using the Project Creator role, you can restrict who can create projects regardless of whether they have policies on individual projects. The project roles can also be applied at the organization level and can be inherited by all the projects in your company. For example, you can assign your networking team the Network Admin role at the organization level so they have permissions to manage all the networks in all the projects in your company.

As of this writing, the Organization Node is in Alpha. If you have a Google for Work account, you can work with your account manager or support partner to request to create an Organization Node for your company.

IAM Resource Hierarchy

- A policy is set on a resource
 - Each policy contains: Set of roles, role members
- Resources inherit policies from parent
 - Resource policies are a union of parent and resource
- If parent policy less restrictive, overrides more restrictive resource policy



Notes:

Cloud Platform resources are organized hierarchically, where the Organization node is the root node in the hierarchy, the projects are the children of the Organization, and the other resources are the children of projects. Each resource has exactly one parent.

IAM allows you to set policies at the following levels of the resource hierarchy:

- **Organization level.** The Organization resource represents your company. IAM roles granted at this level are inherited by all resources under the organization.
- **Project level.** Projects represent a trust boundary within your company. Services within the same project have a default level of trust. For example, App Engine instances can access Cloud storage buckets within the same project. IAM roles granted at the project level are inherited by resources within that project. When setting policies at the project level, be sure to use audit logs to track project level permission changes.
- **Resource level.** In addition to the existing Cloud Storage and BigQuery ACL systems, additional resources such as Genomics Datasets and Pub/Sub topics support resource-level roles so that you can grant certain users permission to a single resource.

Resources inherit the policies of the parent resource. If you set a policy at the organization level, it is automatically inherited by all its children projects, and if you set a policy at the project level, it is inherited by all its children resources. The effective policy for a resource is the union of the policy set at that resource and the policy inherited from its parent. This policy inheritance is transitive; in other words, resources inherit policies from the project, which inherit policies from the organization. Therefore, the organization-level policies also apply at the resource level.

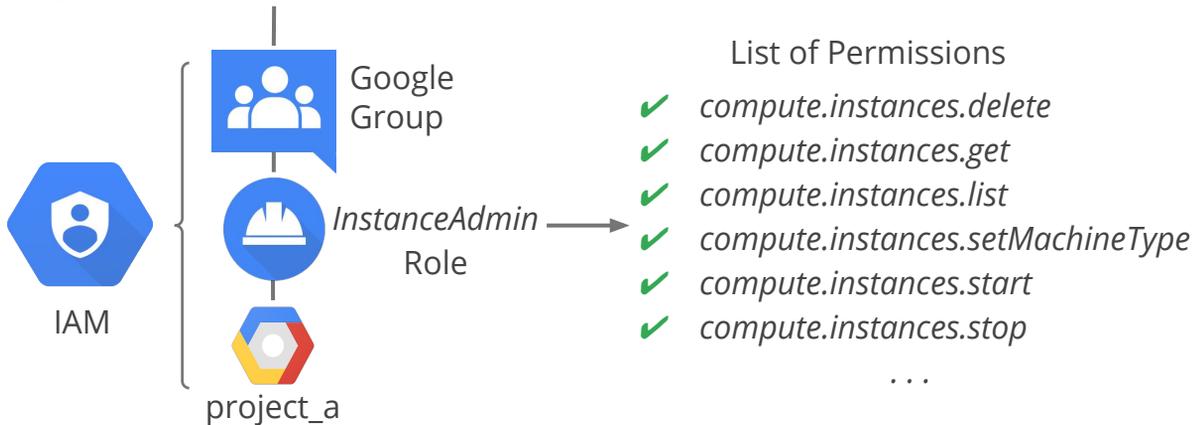
The IAM policy hierarchy follows the same path as the Cloud Platform resource hierarchy. If you change the resource hierarchy, the policy hierarchy changes as well. For example, moving a project into an organization will update the project's IAM policy to inherit from the organization's IAM policy.

Child policies cannot restrict access granted at the parent. For example, if you grant editor role to a user for a project, and grant viewer role to the same user for a child resource, then the user still has editor role for the child resource. When using IAM, a best practice is to follow the principle of least privilege. The principle applies to identities, roles, and resources. Always select the smallest scope that's necessary to reduce your exposure to risk. You wouldn't want to grant everybody the owner role on your entire organization - intentional hacks or accidental mistakes can bring down your apps. You want to be specific and deliberate. Assign a specific security admin group the security admin role to manage SSL and firewall rules on specific projects.

For more information, see: [Identity and Access Management Overview](#)

IAM Roles - Curated Roles

-  Google Account (*test@gmail.com*)
-  Service Account (*test@project_id.iam.gserviceaccount.com*)
-  Google Group (*test@googlegroups.com*)
-  Google Apps Domain (*test@example.com*)



Notes:

The “can do what” part is defined by an IAM role. An IAM role is a collection of permissions. Most of the time to do any meaningful operations you need more than 1 permission. For example to manage instances in a project, you need to create, delete, start, stop and change an instance. So the permissions are grouped together into a role to make it easier to manage.

To give a user the desired permissions, you grant a role to the user on a resource. In this example we are granting a group of users the InstanceAdmin role on project a so the user can manage instances in the project. Whenever possible, it is a best practice to use groups. You should also strictly control the ability to change policies and group memberships which will allow additional users to gain access to resources.

For a complete list of roles by product, see:

https://cloud.google.com/iam/docs/#supported_cloud_platform_services

BigQuery IAM Roles **Beta**

User	Admin	Data viewer	Data editor
<ul style="list-style-type: none">• Runs jobs such as queries• Can browse the project to see what data is available, but does not have access to it by default• Can be assigned at project level or higher	<ul style="list-style-type: none">• All BigQuery related permissions - Access to read, write, delete all data, view jobs and/or cancel them• Can be assigned at project level or higher	<ul style="list-style-type: none">• Can view all datasets and all data within those datasets within the scope of the role• Can be assigned at dataset level or higher	<ul style="list-style-type: none">• Can edit all datasets and all data within those datasets within the scope of the role• Can be assigned at dataset level or higher

Notes:

Because IAM for BigQuery is in beta, not all customers will see the IAM roles in the Cloud Platform Console.

For more information on IAM in BigQuery, including sample scenarios, see: <https://cloud.google.com/bigquery/docs/iam>.

Agenda

- 1 Overview of Access Control Lists
- 2 Project and Dataset Roles
- 3 Applying Views for Row-Level Security
- 4 Identity and Access Management
- 5 Lab

Lab

Create access controls in BigQuery



Image by
Cennie Zhou

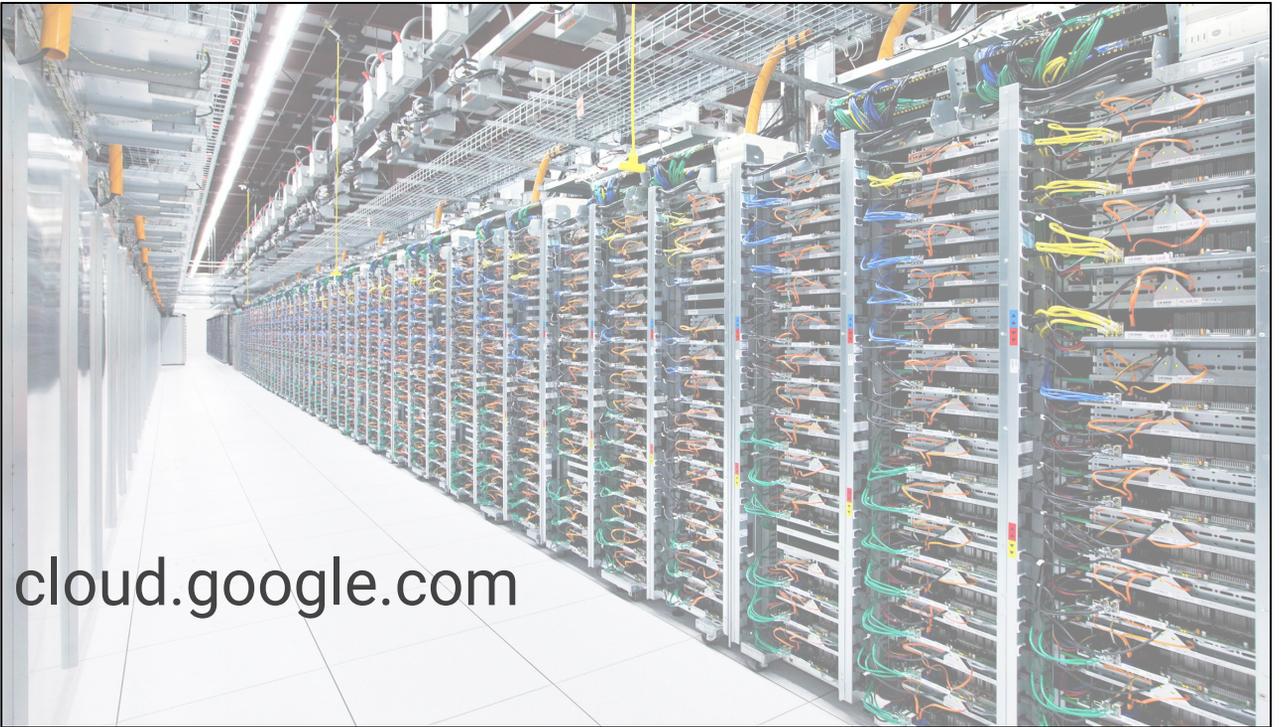
Resources

- Access Control

<https://cloud.google.com/bigquery/access-control>

- Views

<https://cloud.google.com/bigquery/querying-data#views>



cloud.google.com