Google Cloud Tutorial

1 Major Components of the Google Cloud

1. **Cloud Console** This is your interactive, **web-based interface** to the Google Cloud. The base address for this console.cloud.google.com. From here you can access interactive interfaces to most or all portions of the Google Cloud Platform (GCP), including doing things like spinning up VMs, ssh'ing to those VMs, *etc*.
    a. General guide to the console (Google documentation)
    b. "Quick tour of the Google Cloud Console" (ISB-CGC created pdf)

2. **cloud SDK** This is your **command-line interface** to the Google Cloud. You can install this toolkit on your local machine (available Mac, Windows, or Linux/Debian/Ubuntu operating systems). It contains several useful tools for interacting with the GCP from the command-line:
    a. **gsutil** provides access to Cloud Storage, and allows you to create/delete buckets, upload/download/delete objects, list buckets/objects, get/set bucket/object ACLs, *etc*
    b. **bq** provides access to BigQuery, and allows you to run queries, load/export data, list datasets, view schemas, *etc*
    c. **gcloud** provides primary command-line interfaces to GCP (pretty much everything that does not specifically involve Cloud Storage or BigQuery)

3. **Cloud Storage** is a cloud-based object store. The top-level storage container is a "bucket", a bucket contains "objects". This not your typical filesystem: you cannot edit or otherwise modify an object -- you can only upload/download/overwrite/delete objects.
    a. bucket names must be globally unique(!)
    b. there is no "true" hierarchy of folders within a bucket, but object names can contain "/" which can be used to "pretend" that there is a hierarchy

4. **Cloud Shell** is a linux environment accessible directly from the Cloud Console which you can use to interact with the GCP if you don't want to use the cloud SDK from your local machine. The Cloud Shell is a temporary VM instance with 5 GB of persistent disk storage and comes pre-installed with the cloud SDK and other tools (*eg* emacs, vim, git, pip, Docker) and language support (*eg* Python)

5. **Compute Engine** provides scalable, high-performance VMs (virtual machines), with attached persistent disks.

6. **BigQuery** is a fast, economical and fully-managed enterprise data warehouse for large-scale data analytics. ISB-CGC uses BigQuery for storing and querying TCGA clinical and molecular data as well as reference data sources such as GENCODE, Ensembl, *etc*.

7. **Genomics** is primarily a database-backed engine optimized specifically for reads and variants, which can be accessed/queried using the GA4GH API. **NB**: this is *different* from BigQuery.

    a. "**Pipelines API**" -- although this should really be under Compute Engine rather than under Genomics, it was the Google Genomics team that first had a need for an easy-to-use Docker-based "task-runner", and so this functionality is considered part of "Google Genomics". (It should also be called a "Task API" rather than a "Pipelines API"). The GA4GH containers and workflows task team is currently trying to define a more general "Task Execution API" to a more generic "Task Execution Service" (aka TES).

2 Germline AWG Specifics

2.1 GCP Project

A GCP "project" is a space where people can work together. ***Members*** of a project can have different "roles" and different access-permissions within a project. A project is identified by a name, an ID, and a number. The information for the ISB-CGC provided GCP for the germline AWG is:

- **name**: PCA germline AWG
- **ID**: isb-cgc-06-0004
- **number**: 843138098792

In the vast majority of cases, you will need to use the ***project ID*** to reference this project, but there are a few places (*eg* the Console and/or the BigQuery web UI) where you will see the "friendlier" ***project name***.

2.2 Cloud Console

Direct link to the Cloud Console for this project:
https://console.cloud.google.com/home/dashboard?project=isb-cgc-06-0004

2.3 Cloud Storage

Direct link to the Cloud Storage Browser for this project:
https://console.cloud.google.com/storage/browser?project=isb-cgc-06-0004

The outputs of the variant-calling effort can be found in "folders" (by tumor type and CGHub "analysis id") under
`gs://dinglab/isb-cgc/tcga/germline/production`
for example, the VCF file based on one specific SKCM bam file is at:
`gs://dinglab/isb-cgc/tcga/germline/production/SKCM/5f56af72-d206-4780-accb-e85e65fa0705/combine/prefilter.snp_indel.vcf.gz`

See Appendix for additional details about data types.

The manifest used for the runs is available at gs://isb-cgc-open/tcga/GCS_listing.02jun2016.tsv -- if you download that file (or copy it to a VM), you can use it to get information based on the analysis id, eg:

```
$ grep 5f56af72-d206-4780-accb-e85e65fa0705 GCS_listing.02jun2016.tsv
5f56af72-d206-4780-accb-e85e65fa0705    TCGA-EB-A5SF-10A-01D-A30X-08    408d6bbf-3301-416b-9499-ce6ef2c0c8ce
gs://5aa919de-0aa0-43ec-9ec3-288481102b6d/tcga/SKCM/DNA/WXS/BI/ILLUMINA/C828.TCGA-EB-A5SF-10A-01D-A30X-08.1.bam
8794290263
```

Copies of this manifest separated into tumor types are found as
gs://dinglab/manifests/GCS_listing.02jun2016.tcga.*.DNA.WXS.normals.tsv.gz

```
$ grep "5f56af72-d206-4780-accb-e85e65fa0705" CGHub_Manifest_24jun2016.csv
```

```
TCGA,TCGA-EB-A5SF-10A-01D-A30X-08,SKCM,SKIN CUTANEOUS MELANOMA,NB,Blood Derived
Normal,DNA,WXS,BI,ILLUMINA,Illumina,HG19_Broad_variant,C828.TCGA-EB-A5SF-10A-01D-A30X-08.1.bam,8794290263,652f75
054f5e8cf1a84e7b0cce97b2a1,5f56af72-d206-4780-accb-e85e65fa0705,408d6bbf-3301-416b-9499-ce6ef2c0c8ce,32e40e0b-e7
4b-428c-acd3-84ec6a80c024,3681e14a-ae9a-4740-8597-e4c9caaecf1c,EB,,2013-09-17,2013-09-17,2016-05-16,Live,10,D,Il
lumina HiSeq 2000,bam,,Agilent,"Custom V2 Exome Bait, 48 RXN X 16
tubes",931070,,https://bitbucket.org/cghub/cghub-capture-kit-info/raw/d8b126dd4f33eb7164535e00f0ec9a5985056f34/B
I/vendor/Agilent/whole_exome_agilent_1.1_refseq_plus_3_boosters.targetIntervals.bed,https://bitbucket.org/cghub/
cghub-capture-kit-info/raw/d8b126dd4f33eb7164535e00f0ec9a5985056f34/BI/vendor/Agilent/whole_exome_agilent_1.1_re
fseq_plus_3_boosters.baitIntervals.bed
```

And this information is also available in an BigQuery table.

For convenience, unzipped VCF files have been saved as similarly-named objects -- just replace "production" with "unzip" and remove the ".gz" ending:

```
gs://dinglab/isb-cgc/tcga/germline/unzip/SKCM/5f56af72-d206-4780-accb-e85e65fa0705/combine/prefilter.snp_indel.v
cf
```

Storage costs in GCS (Google Cloud Storage) depend on the type of storage, but storage costs for a typical multi-regional bucket is $0.026 per GB per month. The total size of the gs://dinglab bucket is currently 20.36 TiB.

2.4 Compute Engine

Direct link to the Compute Engine > VM instances (Cloud Console page) for this project:
https://console.cloud.google.com/compute/instances?project=isb-cgc-06-0004

2.5 BigQuery

Direct Link to BigQuery web UI for this project:
https://bigquery.cloud.google.com/queries/isb-cgc-06-0004

2.6 Genomics / GA4GH API

Direct Link to the Cloud Console Genomics page for this project:
https://console.cloud.google.com/genomics/datasets?project=isb-cgc-06-0004

3 Tutorial Examples

3.1 Cloud Shell

3.1.1 Tutorial #1 -- A few simple examples

1. start from the Google Cloud Platform console
2. click on the Cloud Shell icon in the top blue bar -- it looks like this: **[>_]** and when you hover over it it will say "Activate Google Cloud Shell"
    a. after you click on it, a new "terminal" window will open in the bottom of your browser view and it will say something like `"Provisioning your Google Cloud Shell machine... Since you haven't used your Cloud Shell for a while, it may take some time to unarchive your disk."`
    b. when you see a prompt (like **<user-name>@isb-cgc-06-0004:~$**) your Cloud Shell is ready
    c. you can open this terminal in a new/larger window by clicking on the "pop-out" icon in the upper-right corner of your terminal, between the hide ("_") and close ("**X**") icons
    d. some commands to try (omit the '**$**', this is simply to indicate that these are to be typed at the command-line prompt -- also if they span multiple lines below, be sure to copy-paste the entire command):

```
$ gcloud config list
$ gcloud --help
$ gcloud compute --help
$ gcloud compute instances list

$ gsutil ls -l gs://dinglab
$ gsutil ls -l
gs://dinglab/isb-cgc/tcga/germline/unzip/BRCA/022710d0-4057-4aa2-9d67-315088dc14da/combine/prefilter.snp_indel.v
cf
$ gsutil ls -l gs://dinglab/isb-cgc/tcga/germline/*/BRCA/022710d0-4057-4aa2-9d67-315088dc14da/combine/*

$ pwd
$ mkdir scratch
$ cd scratch
$ gsutil cp
gs://dinglab/isb-cgc/tcga/germline/production/BRCA/022710d0-4057-4aa2-9d67-315088dc14da/combine/prefilter.snp_in
del.vcf.gz .
$ gunzip prefilter.snp_indel.vcf.gz
$ ls -l prefilter.*
$ grep "^#CHROM" prefilter.snp_indel.vcf | cut -f10


$ bq ls
$ bq ls testing
```

`$ exit` (this will close your Cloud Shell, but anything you have saved "locally" will persist and will still be there the next time you invoke the Cloud Shell)

3.2 BigQuery

3.2.1 Tutorial #1 -- Finding open-access datasets from other projects

Start at the BigQuery web UI at bigquery.cloud.google.com

1. The left-side "navigation panel" shows you a list of (some of) the projects > datasets > tables that you may access.
2. If you do not see ISB-CGC (isb-cgc) in this list, you can add it to your "BQ view" by clicking on the small blue triangle near the top of the left panel, selecting "Switch to project" and then "selecting Display project…" Type "**`isb-cgc`**" (w/o quotes) in the Project ID box, the "Display project in navigation panel" option should be selected, and click OK.
3. Repeat the above for a project called "**`silver-wall-555`**"

3.2.2 Tutorial #2 -- A simple query on a large VCF table

Start at the BigQuery web UI at bigquery.cloud.google.com

1. In BigQuery, tables are organized in "datasets", and "datasets" are owned by "projects".
2. At the top of your left-side navigation panel, you should see that **PCA germline AWG** project, with two datasets underneath: **ExAC** and **testing**.
3. Click on the "**testing**" dataset name. The list of tables in that dataset will appear in the navigation panel, and the "Dataset Details" will open to the right.
4. Click on the "germline_vcf_14403" table name.
5. Notice the three buttons in the main panel, below the heading **Table Details: germline_vcf_14403**: **Schema**, **Details**, and **Preview.** Click on each one.
    a. The "Schema" page shows you this table's schema -- not that this schema is relatively complicated: it contains ***repeated fields*** as well as a ***repeated record*** (the "**call**")
    b. The "Details" page tells you that this table has over 420 million rows and takes up 887 GB. (Storing this table will cost $17.74 per month initially. If it remains unchanged for 3 months, the costs will drop to $8.87 per month.) A query that scans this ***entire*** table (all fields) will cost (at least) **$4.435** -- so please try to be efficient with your queries!
    c. The "Preview" page lets you see the first few rows in this table. (It's like a ***free*** SELECT * query). Use the scroll bar at the bottom to see all of the fields. Notice how repeated and nested fields are shown.
6. Notice the four buttons near the upper-right corner: Query Table, Copy Table, Export Table, and Delete Table. Click on **Query Table**.
    a. A prototype, partial SELECT statement will appear in a "New Query" text box.
    b. Note that BigQuery originally supported a non-standard version of SQL now known as "Legacy SQL". This is still the default in the web UI, but you probably want to start using Standard SQL from the get-go. To enable Standard SQL, click on the "**Show Options**" button, uncheck the box that says "Use Legacy SQL", and then click on "**Hide Options**".
    c. Now click on **Query Table** again, and notice that the template query changes from:
        ```
        SELECT  FROM [isb-cgc-06-0004:testing.germline_vcf_14403] LIMIT 1000
        ```
        to
        ```
        SELECT  FROM `isb-cgc-06-0004.testing.germline_vcf_14403` LIMIT 1000
        ```
        How you specify the table name is one key difference between Legacy SQL and Standard SQL. A "complete" table name is of the form `[<project-id>:<dataset-name>.<table-name>]` in legacy SQL and `` `<project-id>.<dataset-name>.<table-name>` `` in standard SQL.
    d. Paste the following query into the New Query box:
        ```
        SELECT reference_name, COUNT(*) AS n FROM `isb-cgc-06-0004.testing.germline_vcf_14403`
        GROUP BY reference_name ORDER BY n DESC
        ```
    e. Click on the **Format Query** button (this is not necessary, but it's a nice feature)
    f. Notice the green check-mark to the right -- this indicates that the query appears to be valid. If you instead see a red exclamation mark, click on that exclamation mark to see where the error

appears to be. If you click on the green check-mark you should see the following: **Valid**: This query will process 1.33 GB when run. (Queries cost $5 per TB scanned, so this should cost less than a penny.)

g. Click on the **RUN QUERY** button -- after a few seconds you should see the query results in a table below the query. You can download the results, save them as a new table, or save them to Google Sheets (this last option works only if the results contain 16000 or fewer rows).

h. Notice the three buttons side-by-side above the results: **Results | Explanation | Job Information**

   i. **Results** shows you the query results in a Table (or as JSON -- see the buttons at the bottom of the table, you can also scroll through the results using the First / Prev / Next / Last buttons)

   ii. **Explanation** shows information about the query execution plan, the timing of different stages, etc -- can be useful for optimizing or debugging queries.

   iii. **Job Information** shows you top-level information about the job. Notice that there is a "Destination Table" for all jobs -- if you rerun this exact query in the near future, you will not be billed for it again (and it will run much faster) because the results will have been cached.

3.2.3 Tutorial #3 -- A simple JOIN example

Start at the BigQuery web UI at bigquery.cloud.google.com

1. This time, we're going to JOIN the VCF table from above to one of the ISB-CGC tables.
2. Click on the **COMPOSE QUERY** button in the upper left corner.
3. Enable Standard SQL as before (uncheck the Use Legacy SQL option).
4. Paste this query into the New Query text box:

```
WITH
  varCounts AS (
  SELECT
    SUBSTR(c.call_set_name,1,12) AS case_barcode,
    COUNT(*) AS nVar
  FROM
    `isb-cgc-06-0004.testing.germline_vcf_14403`,
    UNNEST(call) AS c
  WHERE
    STARTS_WITH(c.call_set_name, "TCGA-")
  GROUP BY
    c.call_set_name ),
  clinInfo AS (
  SELECT
    ParticipantBarcode,
    Study,
    gender,
    vital_status,
    days_to_last_known_alive,
    tumor_tissue_site,
    age_at_initial_pathologic_diagnosis,
    year_of_initial_pathologic_diagnosis
  FROM
    `isb-cgc.tcga_201607_beta.Clinical_data` )
SELECT
  *
FROM
  varCounts
JOIN
  clinInfo
ON
  case_barcode=ParticipantBarcode
ORDER BY
  nVar DESC
```

5. Click Run Query!

3.3 Compute Engine

3.3.1 Tutorial #1 -- Creating and using a VM

3.3.1.1 Creating a VM with a Data Disk (in addition to Boot Disk)

Start at the Cloud Console Compute Engine VM instances page here.

1. Click on **[+] CREATE INSTANCE**
    a. set Name (*eg* <my-initials>-test-1) -- do not use the default "instance-1"
    b. set Zone (*eg* us-central1-c)
    c. set Machine type (*eg* 4 vCPUs with 15 GB memory)
    d. ***optionally***, change the boot disk to (for example) Ubuntu 14.04 LTS with 10 GB standard persistent disk -- note that the **boot disk** will be named with the same name as the VM (*eg* smr-test-1)
    e. leave the Identity and API access box as is (with "Compute Engine default service account" and "Allow default access" selected)
    f. expand the **Management, disk, networking, SSH keys** section
        i. select the **Disks** tab
        ii. click on + Add item
            1. in the **Name** pull-down, select "Create disk" -- a "Create a disk" panel will open
                a. set Name (*eg* smr-data-disk-1) -- do not use the same name as the VM!
                b. set Source type to "None (blank disk)"
                c. set Size (*eg* 500 GB)
                d. leave default Encryption (which is "Automatic (recommended)")
                e. click on the blue **Create** button -- this will only create the ***disk*** for now
    g. ***before*** clicking on the Create button (for the VM), click on the *bottom* line where it says Equivalent **REST** or **command line** -- you can re-use this later and create this same VM from the command-line rather than having to go through these interactive steps -- they are also a nice record of exactly how you created the VM
    h. **NB**: that this VM will cost an estimated **$122.60 per month**. If you "stop" the VM, you will no longer pay for the VM itself, but you ***will*** continue to pay for the disk space, every hour of every day, until you "delete" this VM.
    i. click on the blue **Create** button -- you will now see your VM spinning up on the VM instances page of the Console
    j. if you go to the "Disks" page, you will see two disks "in use by" this VM: the 10 GB boot disk and the 500 GB data-disk

Equivalent gcloud command line (for ***my*** instance):

```
gcloud compute --project "isb-cgc-06-0004" instances create "smr-test-1" --zone "us-central1-c" --machine-type
"n1-standard-4" --subnet "default" --maintenance-policy "MIGRATE" --scopes
843138098792-compute@developer.gserviceaccount.com="https://www.googleapis.com/auth/devstorage.read_only","https
```

```
://www.googleapis.com/auth/logging.write","https://www.googleapis.com/auth/monitoring.write","https://www.google
apis.com/auth/servicecontrol","https://www.googleapis.com/auth/service.management.readonly","https://www.googlea
pis.com/auth/trace.append" --disk "name=smr-data-disk-1,device-name=smr-data-disk-1,mode=rw,boot=no" --image
"/debian-cloud/debian-8-jessie-v20161215" --boot-disk-size "10" --boot-disk-type "pd-standard"
--boot-disk-device-name "smr-test-1"
```

3.3.1.2 Configure the VM

Now you can ssh to your VM from any command-line where you have the cloud SDK installed. (You can also SSH to it by using the SSH button for your VM on the VM instances page.) If you don't have the cloud SDK installed on your own machine, you can also use the Cloud Shell.

```
$ gcloud compute --project "isb-cgc-06-0004" ssh --zone "us-central1-c" "YOUR-VM-NAME"
```

3.3.1.3 Format and Mount the Disk

You can see the disks that are attached to your VM like this:

```
$ ls /dev/disk/by-id
```

which should respond with something like:

```
google-smr-data-disk-1  google-smr-test-1-part1                        scsi-0Google_PersistentDisk...
google-smr-test-1       scsi-0Google_PersistentDisk_smr-data-disk-1    scsi-0Google_PersistentDisk...
```

The first one above (google-smr-data-disk-1) is the additional disk that was created, while the second one (google-smr-test-1) is the boot disk which has the name name as the VM. The following commands will format and mount the data disk. (Note that you will need to use the proper disk and VM names for ***your*** VM rather than the "smr" examples below.)

```
$ sudo mkfs.ext4 -F -E lazy_itable_init=0,lazy_journal_init=0,discard /dev/disk/by-id/google-smr-data-disk-1
$ sudo mkdir -p /mnt/SCRATCH
$ sudo mount -o discard,defaults /dev/disk/by-id/google-smr-data-disk-1 /mnt/SCRATCH
$ sudo chmod 777 /mnt/SCRATCH
```

You can verify that the disk has been properly mounted using the "df -h" command:

```
$ df -h
```

which should respond with something including a line like:

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb        493G   70M  467G   1% /mnt/SCRATCH
```

where you will see close to 500G of space available mounted as `/mnt/SCRATCH`.

3.3.1.4 Install Packages

Once you have ssh'd into your VM, you can install packages using, for example "apt-get …" or "git clone …". A few important notes:

- You have sudo privileges on this VM
- Unlike the Cloud Shell VM where you are authenticated with your personal credentials, when you ssh to a VM you will be using the "Compute Engine default service account" credentials. (This may matter

depending on what data you are trying to access.) You can verify which credentials you are using by running the "`$ gcloud config list`" command.

You now have a Google Compute Engine VM at your disposal and you should be ready to get to work!

3.3.1.5 Clean Up (also VERY important!!!)

As mentioned above, you will continue to pay for this VM and the attached disks until you stop or delete this VM. You can do this directly from the Console or using the gcloud command-line interface. If you're going to need the VM again later this week but won't be using it for a few days, it's very easy to STOP and RESTART the VM. If you're not going to need this VM for several weeks (or ever), be sure to DELETE it, and also make sure that you haven't forgotten to DELETE any persistent disks you may have created. (Depending on how/when you create persistent disks, they may be deleted when the VM is DELETED, or the may not be.)

4 Appendix

Details to appear here.