**Ingesting FHIR Data with the Healthcare API**

Use the basic functionality of Cloud Healthcare API using Fast Healthcare Interoperability Resources (FHIR) data model, how to export data to BigQuery, and how to access data in BigQuery via SQL.

### **What you learn**

In this lab, you will:

* Gain a general understanding of Cloud Healthcare API and its role in managing healthcare data.
* Learn how to create Cloud Healthcare API datasets and stores.
* Import and export FHIR data using the Cloud Healthcare API.
* Export data from Cloud healthcare API to BigQuery
* Access data in BigQuery via SQL

## Healthcare API introduction

Cloud Healthcare API provides a managed solution for storing and accessing healthcare data in Google Cloud, providing a critical bridge between existing care systems and applications hosted on Google Cloud. Using the API, you can unlock significant new capabilities for data analysis, machine learning and application development, and use these capabilities to build the next generation of healthcare solutions.

The API is comprised of three modality-specific interfaces that implement key industry-wide standards for healthcare data:

* FHIR, an emerging standard for health data interchange
* HL7v2, the most widely adopted method for health systems integration
* DICOM, the dominant standard for radiology and imaging-related disciplines

Each interface is backed by a standards-compliant data store that provides read, write, search, and other operations on the data.

The Cloud Healthcare API provides a number of key features that are critical to bridging current technologies to the next generation of healthcare systems and applications:

* **Standards conformance** - Google supports the use of standards-based interoperability through its participation in a number of healthcare standards bodies. In the Cloud Healthcare API each modality-specific data store and its associated API is substantially conformant with its relevant standard. For example, FHIR stores implement STU3, the current version of the FHIR specification, and DICOM stores implement DICOMweb, a web-based standard for exchanging medical images. In future updates, we expect to support additional versions of these specifications as well as the ability to request a resource in a different version than its canonical representation.
* **Compliance with privacy regulations** - Google Cloud provides detailed guidance regarding how it supports compliance with HIPAA in the US, the PIPEDA in Canada, and other global privacy standards at [cloud.google.com/security/compliance](http://cloud.google.com/security/compliance).
* **Data location control** - The Cloud Healthcare API treats data location as a core component of the API. You have the option to select the storage location for each dataset from a list of currently available locations which correspond to distinct geographic areas aligned with Google Cloud's regional structure. Future Google Cloud regions will allow for the distribution of storage across wider geographic areas.
* **Security** - The Cloud Healthcare API security model is based on Google's proven Identity and Access Management (IAM) system. IAM's fine-grained permissions give you complete control over access to your healthcare data. In addition, we've created open-source proxies for our powerful [Apigee API Management system](https://apigee.com/), which provides comprehensive threat detection and traffic management capabilities that allow you to securely expose sensitive ePHI with patient and provider applications.
* **Bulk import and export** - The Cloud Healthcare API's DICOM and FHIR modalities support bulk import and export of data, making it easier to transfer data via the Cloud Storage system.
* **De-identification** - De-identification support for DICOM is available, making it much easier to redact patient information from studies for research and other purposes. The de-identification process operates on a data store basis.
* **Auditability** - Both administrative and data access requests to the Cloud Healthcare API can be audited. Logs are available through Google Cloud's [Stackdriver](https://cloud.google.com/stackdriver/" \t "_blank) hybrid monitoring system.
* **High availability** - Availability for mission-critical scenarios is made possible through Google Cloud's robust and highly redundant infrastructure.

For many applications, the Cloud Healthcare API can provide a modern alternative to legacy stacks implementing DICOM, HL7v2 or FHIR STU3 standards, simplifying data integration with existing systems and enabling the application developers to focus on their differentiating features such as UX and intelligence.

## Healthcare API concepts

To get the most out of the Cloud Healthcare API, there are a few key concepts you'll want to understand. The information below should give you a good sense of Cloud Healthcare API capabilities, but you can find more details in the [Cloud Healthcare API documentation](https://cloud.google.com/healthcare/docs/).

### **General structure of the Cloud Healthcare API**

The Cloud Healthcare API exposes interfaces that enable you to perform different types of functions:

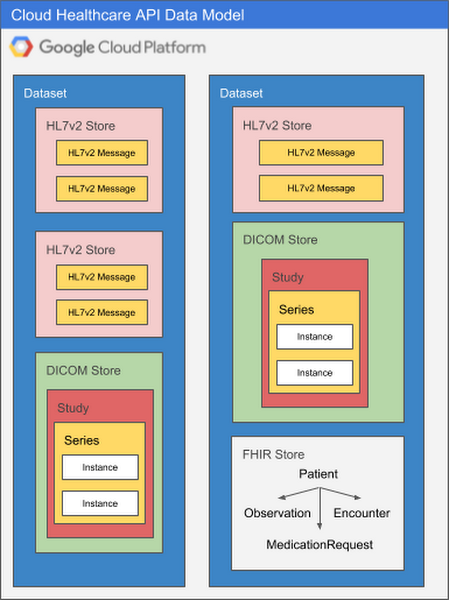
* **Administrative functions**, such as creating or listing datasets and stores that will contain your data.
* **Data access functions** that allow you to create, update, delete and search the data stored in Cloud Healthcare API, or to perform bulk import and export operations.
* **Security functions** that allow you to impose access controls on data stored in Cloud Healthcare API.
* **De-identification functions** that allow you to replace ePHI with anonymized data, or to obfuscate ePHI so that it cannot be used.
* **Metadata functions**, such as retrieval of a [FHIR](http://hl7.org/fhir/) capabilities statement for the FHIR API.

These functions may vary slightly depending on the modality of data (FHIR, [HL7](http://www.hl7.org/) v2 or [DICOM](https://www.dicomstandard.org/)) being operated on. For example, data retrieval operations against an FHIR data store use an API that conforms to the FHIR standard, but data retrieval operations against an HL7 v2 store use operations better suited to operating on HL7v2-structured data.

### **Datasets and stores**

### Because each healthcare data modality has different structural and processing characteristics, datasets are split into modality-specific **stores**. A single dataset can contain one or many stores, and those stores can all service the same modality or different modalities as application needs dictate.

The diagram below illustrates two datasets in a Google Cloud project, each of which contains multiple stores.



There are many ways to structure datasets and stores. As you design systems that use the Cloud Healthcare API, you may want to take the following into consideration:

* **Security and access control:** Rules can be defined at both a dataset and store level, but you may choose to group all data for a particular application into the same dataset, and set access control rules such that only that application can access the dataset.
* **Application requirements:** An application processing different types of data may have all of its data for all modalities in a single dataset.
* **Source systems:** Often, the structure of healthcare data can vary according to the source system and modality. Separating data for different source systems into their own datasets may facilitate processing.
* **Intended use:** Data from different systems can have different intended uses, such as research, analytics, or machine learning predictions. Grouping data by intended use may facilitate ingestion into the target system.
* **Separating ePHI from de-identified data:** Cloud Healthcare API data de-identification functions read from a source dataset and write the output into a new dataset that you specify. If you are preparing data to be used by researchers, for example, this approach to de-identifying data may be a consideration in how you use datasets to segregate data.

### **MLLP Adapter**

The [minimal lower layer protocol (MLLP)](http://www.hl7.org/implement/standards/product_brief.cfm?product_id=55) is the standard used for transmitting HL7v2 messages over TCP/IP connections within a network, such as a hospital.

MLLP does not offer an exact mapping to the Cloud Healthcare API HL7v2 REST API], which uses HTTP. Therefore, an MLLP adapter must be used to convert messages transmitted over MLLP into a format that an HTTP/REST API can accept. To transmit messages over MLLP and then to the Cloud Healthcare API, use the [Google Cloud MLLP adapter](https://github.com/GoogleCloudPlatform/mllp/).

There are many ways to structure datasets and stores. As you design systems that use the Cloud Healthcare API, you may want to take the following into consideration:

* **Security and access control:** Rules can be defined at both a dataset and store level, but you may choose to group all data for a particular application into the same dataset, and set access control rules such that only that application can access the dataset.
* **Application requirements:** An application processing different types of data may have all of its data for all modalities in a single dataset.
* **Source systems:** Often, the structure of healthcare data can vary according to the source system and modality. Separating data for different source systems into their own datasets may facilitate processing.
* **Intended use:** Data from different systems can have different intended uses, such as research, analytics or machine learning predictions. Grouping data by intended use may facilitate ingestion into the target system.
* **Separating ePHI from de-identified data:** Cloud Healthcare API data de-identification functions read from a source dataset and write the output into a new dataset that you specify. If you are preparing data to be used by researchers, for example, this approach to de-identifying data may be a consideration in how you use datasets to segregate data.

### **API structure**

Data in the Cloud Healthcare API datasets and stores can be accessed and managed using a REST API that identifies each store using its project, location, dataset, store type and store name. This API implements modality-specific standards for access that are consistent with industry standards for that modality. For example, the Cloud Healthcare DICOM API natively provides operations for reading DICOM studies and series that are consistent with the DICOMweb standard, and supports the DICOM DIMSE C-STORE protocol via an [open-source adapter](https://github.com/GoogleCloudPlatform/healthcare/tree/master/imaging/dicom_adapter). Similarly, the FHIR API provides operations for accessing or searching FHIR entity types that is based on the FHIR standard, and the HL7v2 API provides operations for reading and searching HL7v2 messages based on HL7v2 message or segment criteria.

Operations that access a modality-specific store use a request path that is comprised of two pieces: a base path, and a modality-specific request path. Administrative operations—which generally operate only on locations, datasets and stores—may only use the base path, but data modality-specific retrieval operations use both the base path (for identifying the store to be accessed) and request path (for identifying the actual data to be retrieved).

To reference a particular store within a Cloud Healthcare API dataset, you would use a base path structured like this:

/projects/<PROJECT>/locations/<LOCATION>/datasets/<DATASET>/<STORE-TYPE>/<STORE-NAME>

A concrete base path example might look like this:

/projects/myProj/locations/REGION/datasets/central-ds1/hl7V2Stores/clinical-store1

which references a Cloud Healthcare HL7 v2 store in the Google Cloud project "myProj", in the "REGION" region, in a dataset called "central-ds1", and with a name of "clinical-store1". This is an HL7 v2 store because of the "hl7V2Stores" type; if you want to access a FHIR store in the same dataset you can use the "fhirStores" type, and if the store contained DICOM data you can used the "dicomStores" type.

To access a specific piece of data, the base path is used in combination with a request path that is formatted according to the appropriate modality standard. For example, a request to read a specific FHIR "Patient" entity using the entity ID might look like this:

<basePath>/resources/Patient/{patient\_id}

with /Patient/{patient\_id} being a path—structured according to the FHIR standard—for the Patient resource whose identifier is specified by {patient\_id}. Similarly, DICOMweb requests to a DICOM store might look like this:

<basePath>/dicomWeb/studies/{study\_id}/series?PatientName={patient\_name}

where {study\_id} identifies a particular DICOM study, and the patient's name is specified by {patient\_name}. In this example, the path specification is consistent with the DICOMweb standard path structure.

gcloud config list project

export PROJECT\_ID=$(gcloud config list --format 'value(core.project)')

export PROJECT\_NUMBER=$(gcloud projects list --filter=projectId:$PROJECT\_ID \

--format="value(projectNumber)")

export LOCATION=europe-west4

export DATASET\_ID=dataset1

export FHIR\_STORE\_ID=fhirstore1

export TOPIC=fhir-topic

export HL7\_STORE\_ID=hl7v2store1

bq --location=europe-west4 mk --dataset --description HCAPI-dataset $PROJECT\_ID:$DATASET\_ID

---------------SETTING UP PERMISSIONS-----------------------------

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member=serviceAccount:service-$PROJECT\_NUMBER@gcp-sa-healthcare.iam.gserviceaccount.com \

--role=roles/bigquery.dataEditor

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member=serviceAccount:service-$PROJECT\_NUMBER@gcp-sa-healthcare.iam.gserviceaccount.com \

--role=roles/bigquery.jobUser

---------------GCP HEALTHCARE PERMISSIONS-----------

gcloud healthcare datasets create $DATASET\_ID --location=$LOCATION

#from gcs to datastore

gcloud healthcare fhir-stores import gcs $FHIR\_STORE\_ID \

--dataset=$DATASET\_ID \

--location=$LOCATION \

--gcs-uri=gs://spls/gsp457/fhir\_devdays\_gcp/fhir1/\* \

--content-structure=BUNDLE\_PRETTY

#BULK EXPORT FRM DATASTORE TO BQ

gcloud healthcare fhir-stores export bq $FHIR\_STORE\_ID \

--dataset=$DATASET\_ID \

--location=$LOCATION \

--bq-dataset=bq://$PROJECT\_ID.$DATASET\_ID \

--schema-type=analytics

gcloud healthcare fhir-stores export bq de\_id \

--dataset=$DATASET\_ID \

--location=$LOCATION \

--bq-dataset=bq://$PROJECT\_ID.de\_id \

--schema-type=analytics

-----------

 create a new FHIR Patient resource in the FHIR store and export the newly created FHIR resource to BigQuery using streaming export.

1. To enable BigQuery streaming, you must update the FHIR store's streamConfigs field. To update the FHIR store, make a PATCH request with the following information:

* The parent dataset
* The FHIR store
* The BigQuery dataset
* The BigQuery project
* An update mask
* An access token

----------

**Lab review**

Cloud Healthcare API provides a comprehensive facility for ingesting, storing, managing, and securely exposing healthcare data in FHIR, DICOM, and HL7 v2 formats. Using Cloud Healthcare API, you can ingest and store data from electronic health records systems (EHRs), radiological information systems (RISs), and custom healthcare applications. You can then immediately make that data available to applications for analysis, machine learning prediction and inference, and consumer access.

Cloud Healthcare API enables application access to healthcare data via widely-accepted, standards-based interfaces such as FHIR STU3 and DICOMweb. These APIs allow data ingestion into modality-specific data stores, which support data retrieval, update, search and other functions using familiar standards-based interfaces.

Further, the API integrates with other capabilities in Google Cloud through two primary mechanisms:

* *Cloud Pub/Sub*, which provides near-real-time updates when data is ingested into a Cloud Healthcare API data store, and
* *Import/export APIs*, which allow you to integrate Cloud Healthcare API into both Google Cloud Storage and Google BigQuery.

Using Cloud Pub/Sub with Google Cloud Functions enables you to invoke machine learning models on healthcare data, storing the resulting predictions back in Cloud Healthcare API data store. A similar integration with Cloud Dataflow supports transformation and cleansing of healthcare data prior to use by applications.

To support healthcare research, Cloud Healthcare API offers de-identification capabilities for FHIR and DICOM. This feature allows customers to share data with researchers working on new cutting-edge diagnostics and medicines.

**Congratulations!**

In this lab you:

* Gained a general understanding of Cloud Healthcare API and its role in managing healthcare data.
* Learned how to create datasets and stores for FHIR data.
* Imported FHIR data from Cloud Storage
* Exported FHIR data to BigQuery in both for bulk data export and streaming
* Reviewed a number of queries against FHIR data in BigQuery