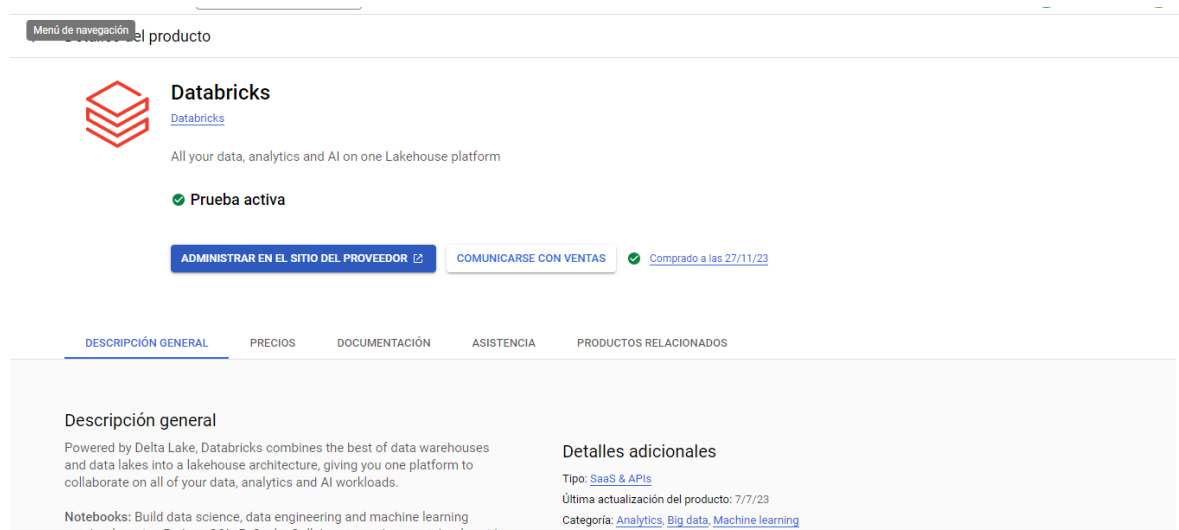


### Lab 3.4: Laboratorio Spark con databricks

Objetivo: Ejecutar cargas de trabajo para analisis de datos usando un cluster de Spark y Notebooks con databricks

Para este laboratorio se creará un cluster de spark con databricks, para ello necesitas una cuenta de databricks puedes solicitar una con GCP ve a la consola de GCP y en el buscador pon databricks, allí se debe realizar la solicitud y te enviará a la página de registro. Una vez aprobada en el resultado de búsqueda de gcp te debe aparecer algo como esto:



Menú de navegación el producto

**Databricks**  
[Databricks](#)

All your data, analytics and AI on one Lakehouse platform

✓ Prueba activa

[ADMINISTRAR EN EL SITIO DEL PROVEEDOR](#) [COMUNICARSE CON VENTAS](#) [Comprado a las 27/11/23](#)

[DESCRIPCIÓN GENERAL](#) [PRECIOS](#) [DOCUMENTACIÓN](#) [ASISTENCIA](#) [PRODUCTOS RELACIONADOS](#)

**Descripción general**

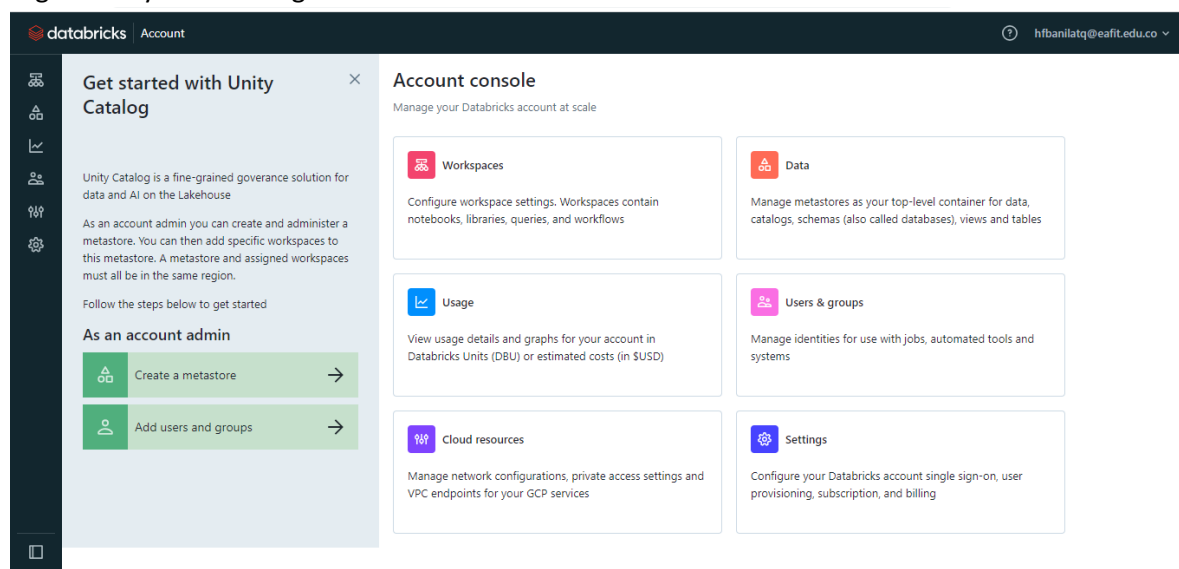
Powered by Delta Lake, Databricks combines the best of data warehouses and data lakes into a lakehouse architecture, giving you one platform to collaborate on all of your data, analytics and AI workloads.

**Detalles adicionales**

Tipo: [SaaS & APIs](#)  
Última actualización del producto: 7/7/23  
Categoría: [Analytics](#), [Big data](#), [Machine learning](#)

**Notebooks:** Build data science, data engineering and machine learning notebooks using Python, SQL, R, Scala. Collaborate on these notebooks with

Al dar clic en “ADMINISTRAR EN EL SITIO DEL PROVEEDOR” te saldrá una alerta de privacidad y al darle aceptar te abrirá una nueva ventana con databricks, inicia sesión con el correo que te registraste y te saldrá algo así :



**databricks** Account hfbani1atq@eafit.edu.co

**Get started with Unity Catalog**

Unity Catalog is a fine-grained governance solution for data and AI on the Lakehouse

As an account admin you can create and administer a metastore. You can then add specific workspaces to this metastore. A metastore and assigned workspaces must all be in the same region.

Follow the steps below to get started

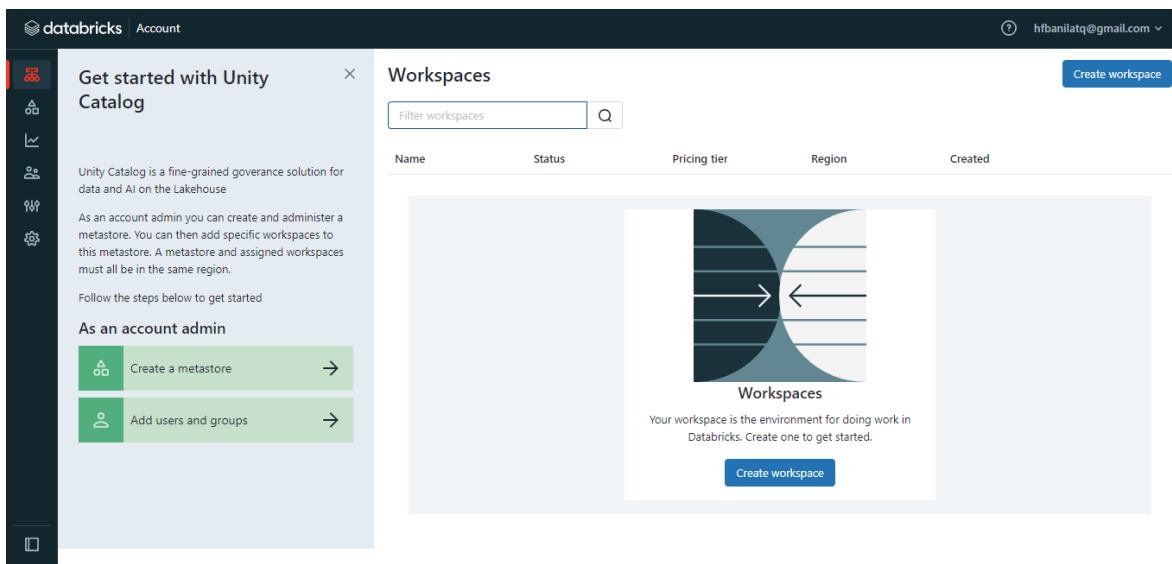
**As an account admin**

- Create a metastore →
- Add users and groups →

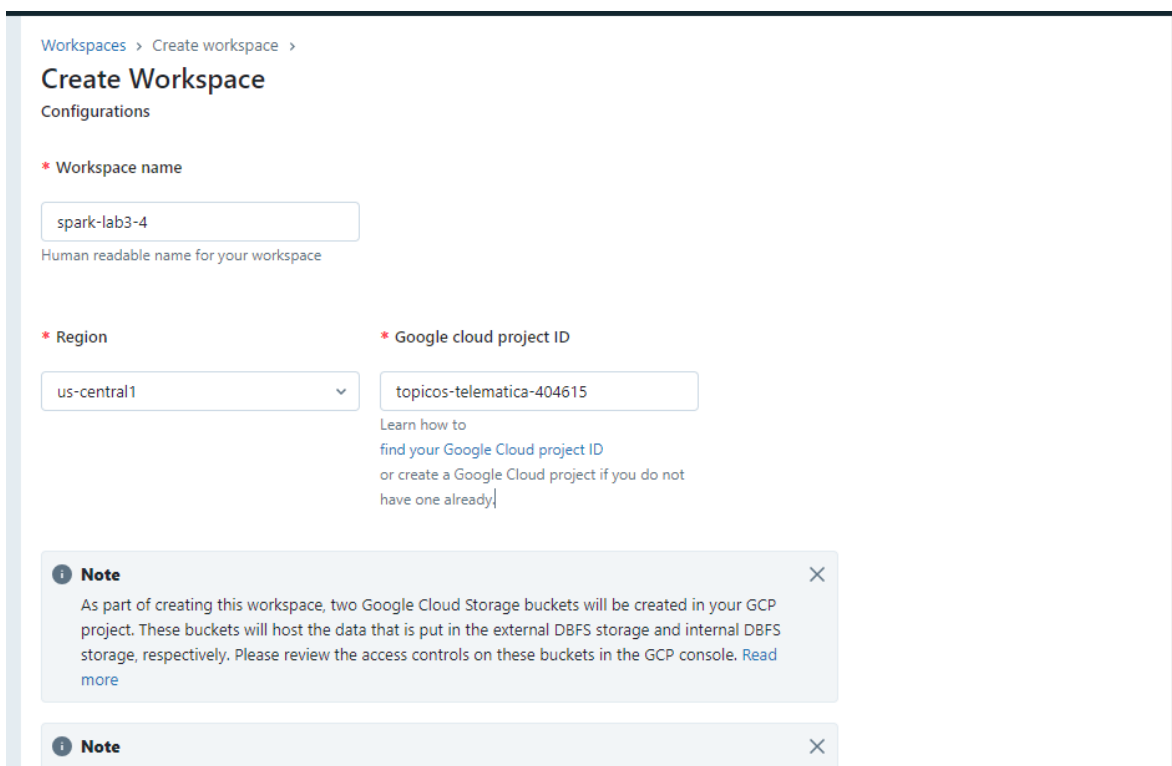
**Account console**  
Manage your Databricks account at scale

- Workspaces**  
Configure workspace settings. Workspaces contain notebooks, libraries, queries, and workflows
- Data**  
Manage metastores as your top-level container for data, catalogs, schemas (also called databases), views and tables
- Usage**  
View usage details and graphs for your account in Databricks Units (DBU) or estimated costs (in \$USD)
- Users & groups**  
Manage identities for use with jobs, automated tools and systems
- Cloud resources**  
Manage network configurations, private access settings and VPC endpoints for your GCP services
- Settings**  
Configure your Databricks account single sign-on, user provisioning, subscription, and billing

Una vez acá necesitamos crear un workspace donde vamos a trabajar en el lab para ello seleccionamos workspaces y damos a crear workspace



Asignamos un nombre al workspace, seleccionamos una zona de disponibilidad y ponemos el id del proyecto con el que vamos a trabajar



Para saber el ID del proyecto ve a la consola de GCP y en donde se selecciona el proyecto dale en administrar y te saldrán los proyectos, allí verás el ID:

Selecciona un proyecto

 PROYECTO NUEVO

Buscar en proyectos y carpetas



RECIENTES

DESTACADOS

TODOS

Nombre		ID
✓ ☆  <a href="#">topicos-telematica</a> ?		topicos-telematica-404615
☆  <a href="#">proyecto3</a> ?		proyecto3-406314

CANCELAR

Creamos el workspace, si no aparece de inmediato cierra sesión e inicia de nuevo. Ahora seleccionamos el workspace

## Workspaces

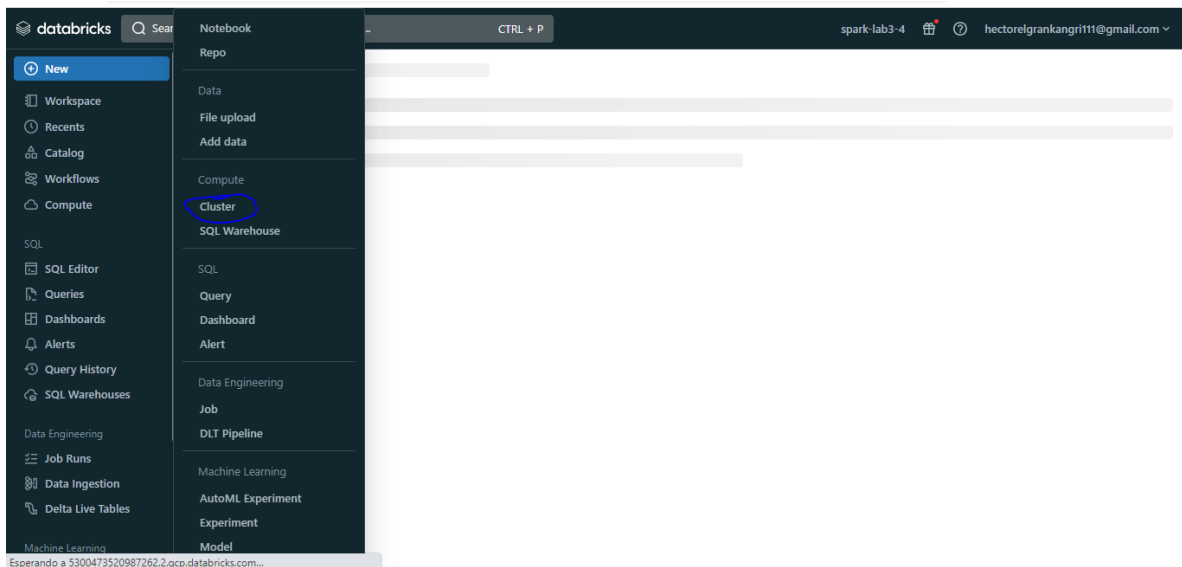
Create workspace

Filter workspaces

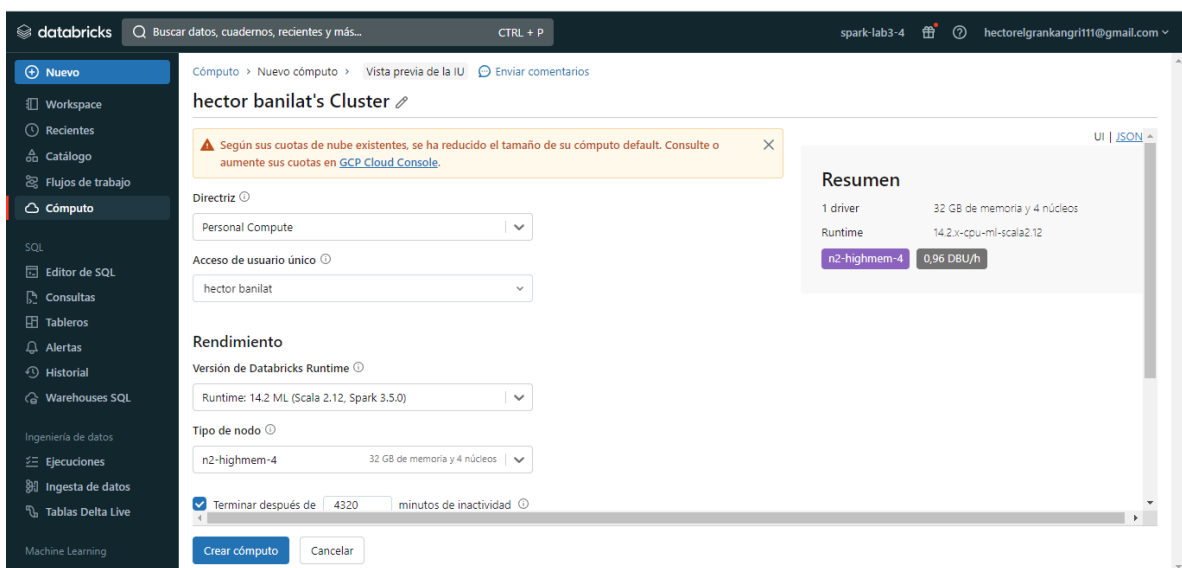


Name	Status	Pricing tier	Region	Created
spark-lab3-4	 Provisioning	Premium	us-west1	today at 7:12 PM

Una vez termine de aprovisionar damos clic en el nombre del workspace y damos clic en Open Workspace, esto nos abrirá una nueva ventana con la que podremos crear Notebooks y clusters para trabajar, en nuestro caso vamos a crear un cluster personal para trabajar con Spark. Para ello damos clic en nuw y seleccionamos cluster



Seleccionamos personal Compute, y nuestro usuario y damos clic en crear, también se puede cambiar la versión de spark y scala, en mi caso lo dejé por defecto.



Esto tardará un tiempo, una vez creado podremos crear notebooks y ejecutarlos usando el cluster que creamos. Mientras eso pasa, vamos a crear un bucket con cloud storage donde vamos a subir los archivos que vamos a emplear en el análisis

Para ello vamos a la consola de GCP y buscamos cloud storage, seleccionamos la opción , allí damos clic en crear

Cloud Storage Buckets **CREAR** ACTUALIZAR APRENDIZAJE

**Buckets**

- Supervisión
- Configuración

**Revisa tus buckets de Autoclass**

Autoclass tendrá nuevos cambios. Revisa los buckets de Autoclass para asegurarte de estar preparado.

MÁS INFORMACIÓN

**Potencia la replicación y las estadísticas casi en tiempo real con transferencias controladas por eventos**

Ahora puedes capturar cambios más rápido en tus fuentes de Google Cloud Storage y Amazon S3 a través de transferencias controladas por eventos, lo que te permite tomar medidas basadas en tus datos casi en tiempo real. Para comenzar, crea un trabajo de transferencia con una transmisión de eventos basada en Pub/Sub o AWS SQS configurada para enviar notificaciones de eventos cuando se crean o actualizan objetos.

CREAR TRABAJO DE TRANSFERENCIA

MÁS INFORMACIÓN

Filtro Filtrar depósitos

Nombre ↑	Fecha de creación	Tipo de ubicación	Ubicación	Clase de almacenamiento predeterminada	Últ
lab-sparkhfbanilatq	27 nov 2023 19:14:31	Region	us-west1	Standard	27

Asignamos un nombre y damos clic en continuar

← Crear un bucket

✓ **Asigna un nombre a tu bucket**

Selecciona un nombre permanente globalmente único. [Lineamientos para asignar nombre](#)

lab-sparkhfbanilatq


Sugerencia: No incluyas información sensible

✓ ETIQUETAS (OPCIONAL)

CONTINUAR

El siguiente paso seleccionamos regional y seleccionamos una región

## ✓ Elige dónde almacenar tus datos

Esta opción define la ubicación geográfica de tus datos y afecta el costo, el rendimiento y la disponibilidad. No se puede cambiar más adelante. [Más información](#) 

### Tipo de ubicación

- ☐ Multi-region  
Máxima disponibilidad en el área más amplia
- ☐ Dual-region  
Alta disponibilidad y baja latencia en 2 regiones
- ☒ Region  
Latencia mínima dentro de una sola región

us-east1 (Carolina del Sur) ▼

CONTINUAR

Deseleccionamos el “Aplicar prevención de acceso público al bucket” en la pestaña de “Elegir como controlar el acceso a los objetos”

## ✓ Elige cómo controlar el acceso a los objetos

### Impide el acceso público

Restringe el acceso público a los datos a través de Internet. Esto evitará que el bucket se use para el hosting web. [Más información](#)

☐ Aplicar la prevención de acceso público a este bucket

### Control de acceso

#### ☒ Uniforme

Garantiza el acceso uniforme a todos los objetos del bucket mediante el uso exclusivo de permisos a nivel de bucket (IAM). Esta opción se aplicará de manera permanente después de 90 días. [Más información](#)

#### ☐ Preciso

Especifica el acceso a objetos individuales mediante el uso de permisos a nivel de objeto (LCA) además de los permisos a nivel de bucket (IAM). [Más información](#)

CONTINUAR

Y listo, lo demás se queda tal cual está y damos clic en crear. Nos quedará algo así:

The screenshot shows the Google Cloud console interface. On the left is a navigation menu with 'Buckets' selected. The main area displays the 'Detalles del bucket' for 'lab-sparkhfbaniatq'. It shows the location as 'us-east1 (Carolina del Sur)', storage class as 'Standard', public access as 'No público', and protection as 'Ninguno'. Below this are tabs for 'OBJETOS', 'CONFIGURACIÓN', 'PERMISOS', 'PROTECCIÓN', 'CICLO DE VIDA', 'OBSERVABILIDAD', and 'INFORMES DE INVENTARIO'. The 'OBJETOS' tab is active, showing a list of objects with columns for Nombre, Tamaño, Tipo, Fecha de creación, Clase de almacenamiento, Última modificación, Acceso público, Historial de versiones, and Encriptación. A message at the bottom states 'Se creó el bucket lab-sparkhfbaniatq'.

Ahora vamos a subir los datos al bucket para ello necesitas descargar el dataset en el repo, en la carpeta reto3.1 se encuentra.

Creamos la carpeta covid-19 y subimos los archivos a esta carpeta, que se encuentran en la carpeta covid dentro de datasets. Subimos ambos archivos

**lab-sparkhfbanilatq**

Ubicación: us-east1 (Carolina del Sur) | Clase de almacenamiento: Standard | Acceso público: No público | Protección: Ninguno

OBJETOS | CONFIGURACIÓN | PERMISOS | PROTECCIÓN | CICLO DE VIDA | OBSERVABILIDAD | INFORMES DE INVENTARIO

Depósitos > lab-sparkhfbanilatq > covid-19

SUBIR ARCHIVOS | SUBIR CARPETA | CREAR CARPETA | TRANSFERIR LOS DATOS | ADMINISTRAR CONSERVACIONES | DESCARGAR | BORRAR

Filtrar solo por prefijo de nombre | Filtro: Filtrar objetos y carpetas | Mostrar datos borrados

Nombre	Tamaño	Tipo	Fecha de creación	Clase de almacenamiento	Última modificación	Acceso público
Casos_positivos_de_COVID-19_en_...	16.4 MB	text/csv	27 nov 2023 19:41:45	Standard	27 nov 2023 19:41:45	No público
Casos_positivos_de_COVID-19_en_...	171.8 KB	text/csv	27 nov 2023 19:41:39	Standard	27 nov 2023 19:41:39	No público

2 archivos con carga exitosa.

Una vez hecho esto, ya podemos proceder a crear el Notebook para analizar los datos, databricks tiene las variables sc y pyspark por defecto, por lo que no será necesario crearlas. Vamos a crear el Notebooks, para ello nos dirigimos al workspace de Databricks y damos en new -> Notebook (Cuaderno si está en español como el mío)

**databricks** | Workspace | Nuevo | Workspace | Recientes | Catálogo | Flujos de trabajo | Cómputo | SQL | Editor de SQL | Consultas | Tableros | Alertas | Historial | Warehouses SQL | Ingeniería de datos | Ejecuciones | Ingesta de datos | Tablas Delta Live | Machine Learning

Repositorio: Datos | Cargar archivo | Añadir datos | Cómputo | Clúster | Warehouse SQL | SQL | Consulta | Tablero | Alerta | Ingeniería de datos | Trabajo | Canal DLT | Machine Learning | Experimento de AutoML | Experimento | Modelo

Workspace > Users > hectorrelgrankangri111@gmail.com

Nombre | Type | Owner | Creado

Esta carpeta está vacía

7:46 p.m. 27/11/2023

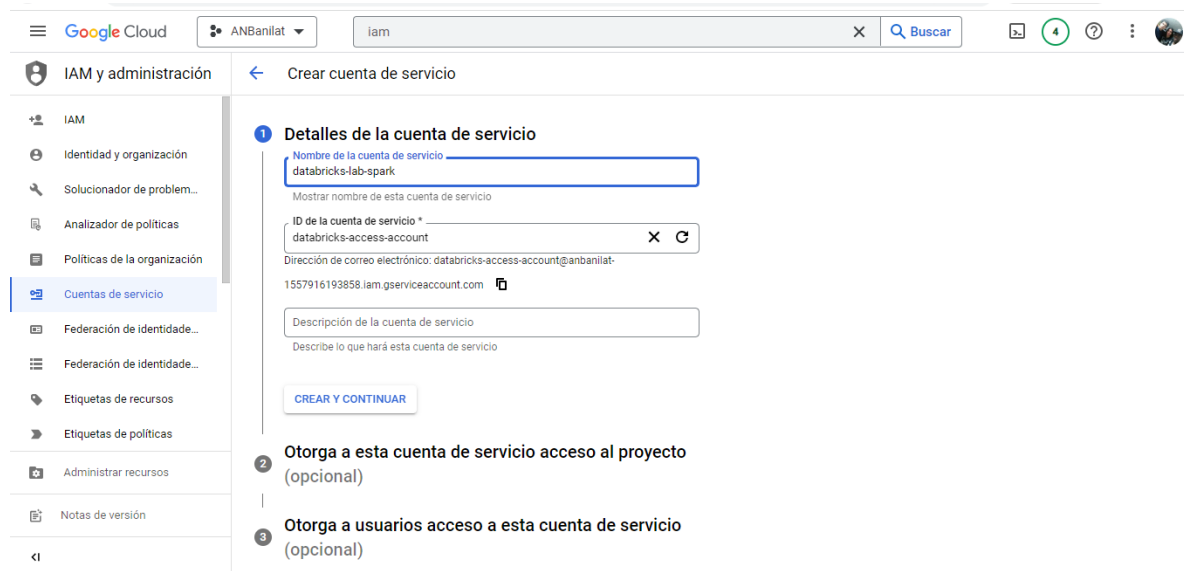


Esto nos abrirá un cuaderno en blanco, podemos programar en Scala, Python, Sql o R, para propósitos del laboratorio vamos a trabajar en Python, ya que en el repo del profesor <https://github.com/st0263eafit/st0263-232/tree/main/bigdata/02-spark> se encuentra una guía sobre cómo trabajar con los datos. Si gustas puedes primero intentar trabajar con los códigos que indica el profe para que entiendas un mejor este trabajo.

Ahora procedemos a crear el código de Python, primero obtendremos los datos de nuestro bucket. Para ello necesitamos configurar databricks para que pueda acceder al bucket, debemos seguir este tutorial para darle acceso al bucket a databricks

<https://docs.databricks.com/en/storage/gcs.html>

Acá te doy un resumen, debemos ir a la consola de GCP, y buscar IAM and Admin, Ahí vamos a Cuentas de servicio y creamos una nueva para darle solo acceso al bucket



The screenshot shows the Google Cloud IAM and Admin console. The left sidebar lists navigation options: IAM, Identidad y organización, Solucionador de problem..., Analizador de políticas, Políticas de la organización, Cuentas de servicio (highlighted), Federación de identidad..., Federación de identidad..., Etiquetas de recursos, Etiquetas de políticas, Administrar recursos, and Notas de versión. The main content area is titled 'Crear cuenta de servicio' and contains three steps:

- 1 Detalles de la cuenta de servicio**
  - Nombre de la cuenta de servicio**: `databricks-lab-spark`
  - ID de la cuenta de servicio \***: `databricks-access-account`
  - Dirección de correo electrónico**: `databricks-access-account@anbanilat-`
  - 1557916193858.iam.gserviceaccount.com**
  - Descripción de la cuenta de servicio**: (empty field)
  - Describe lo que hará esta cuenta de servicio**: (empty field)
  - CREAR Y CONTINUAR** button
- 2 Otorga a esta cuenta de servicio acceso al proyecto (opcional)**
- 3 Otorga a usuarios acceso a esta cuenta de servicio (opcional)**

Damos el nombre la descripción y creamos. D

Una vez creada la debemos seleccionar la cuenta recién creada (DEBEMOS COPIAR EL CORREO QUE APARECE LO USAREMOS MAS ADELANTE), y damos clic en CLAVES, y damos crear nueva clave, seleccionamos json y listo

← databricks-lab-spark

DETALLES

PERMISOS

CLAVES

MÉTRICAS

REGISTROS

Claves

⚠

Las claves de cuenta de servicio podrían poner en riesgo la seguridad si se ven comprometidas. Te recomendamos que no descargues claves de cuenta de servicio y que, en su lugar, uses la [Federación de identidades para cargas de trabajo](#). Puedes obtener más información sobre cuál es la mejor manera de autenticar las cuentas de servicio en Google Cloud [aquí](#).

Agrega un nuevo par de claves o sube un certificado de clave pública de un par de claves existente.

Impide la creación de claves de cuentas de servicio con las [políticas de la organización](#).

Más información para configurar políticas de la organización en cuentas de servicio

AGREGAR CLAVE

Tipo	Estado	Clave	Fecha de creación de la clave	Fecha de vencimiento de la clave	
	Activa	fe4e0d345b09f465f14cbec9ec5d3815b2a18ab5	27 nov 2023	31 dic 9999	

Se creó la cuenta de servicio

Ahora vamos al bucket, lo seleccionamos y damos clic en permisos. Y allí OTORGAR ACCESO. En esta parte copiamos el correo que se generó en el paso de creación de la cuenta de servicio, luego adicionar Role -> Cloud Storage -> Administrador de almacenamiento

## Edita el acceso a "lab-sparkhfbanilatq"

### Principal ?

databricks-access-account@anbanilat-1557916193858.iam.gserviceaccount.com

### Recurso

lab-sparkhfbanilatq

## Asignar funciones

Los roles se componen de conjuntos de permisos y determinan lo que la principal puede hacer con este recurso. [Más información](#)

Rol

Administrador de almacenamien...

Condición de IAM (opcional) ?

+ AGREGAR CONDICIÓN DE IAM



Otorga control total sobre los buckets y los objetos.

+ AGREGAR OTRO ROL

GUARDAR

PROBAR CAMBIOS



CANCELAR

Ahora vamos a configurar el cluster con este correo de cuenta de acceso, para ello vamos al menú compute ahí seleccionamos nuestro cluster y luego damos en editar

Cómputo > Vista previa de la IU [Enviar comentarios](#)

### hector banilat's Cluster

Ver más ...

Terminar

Editar

Configuración Cuadernos (0) Bibliotecas Log de eventos IU de Spark Logs del driver Métricas Aplicaciones IU de cómputo Spark - Máster

Directriz

Personal Compute

Modo de acceso

Acceso de usuario único

Usuario único

hector banilat

### Rendimiento

Versión de Databricks Runtime

14.2 ML (includes Apache Spark 3.5.0, Scala 2.12)

☐ Utilizar aceleración Photon

Tipo de nodo

n2-highmem-4

32 GB de memoria y 4 núcleos

☒ Terminar después de 4320 minutos de inactividad

### Etiquetas

Sin etiquetas personalizadas

UI | JSON

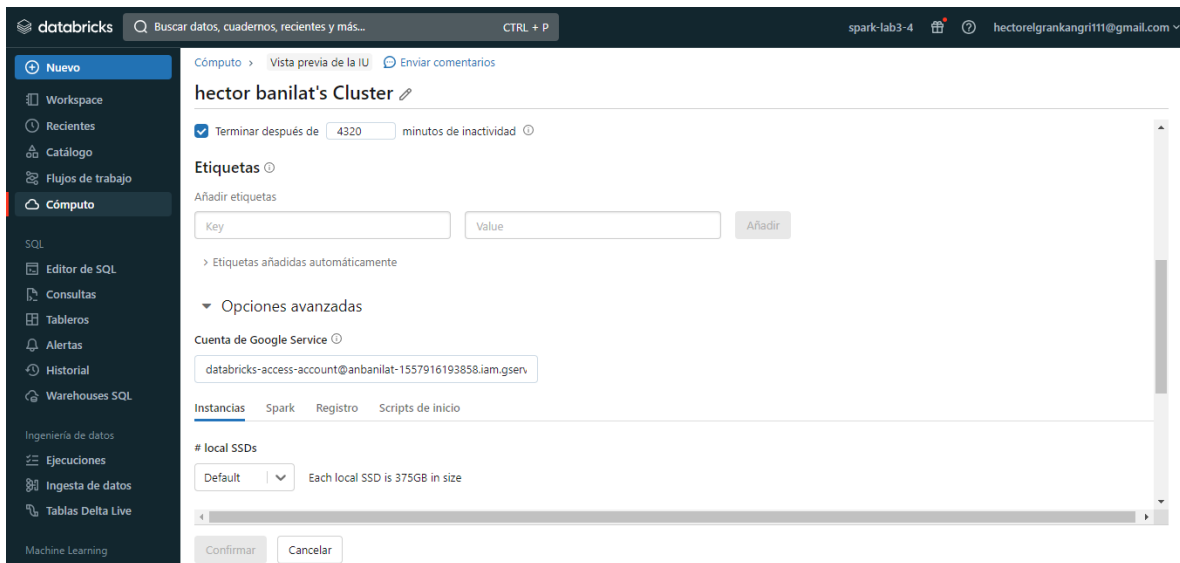
### Resumen

1 driver 32 GB de memoria y 4 núcleos

Runtime 14.2.x-cpu-ml-scala2.12

n2-highmem-4

0,96 DBU/h



En opciones avanzadas pegamos la cuenta de servicio que creamos con el acceso Y ya deberíamos tener acceso al bucket.

Ya podemos empezar con el desarrollo, para ello vamos al notebook que creamos anteriormente y pegamos cada celda que está en el archivo Analisis\_Covid.ipynb (También podemos importarlo)

Y damos ejecutar todas las celdas y listo, ya tenemos nuestro análisis, acá te voy a dejar una captura de pantalla de cada celda y su resultado:



```

1 # Renombrar todas las columnas: reemplazar espacios con '_' y convertir a minúsculas
2 mapeo_acentos = {
3     'Á': 'A', 'É': 'E', 'Í': 'I', 'Ó': 'O', 'Ú': 'U',
4     'á': 'a', 'é': 'e', 'í': 'i', 'ó': 'o', 'ú': 'u',
5     'Ñ': 'N', 'ñ': 'n', 'ã': 'a', 'ê': 'e', 'ï': 'i', 'ö': 'o', 'ü': 'u',
6     'Ã': 'A', 'Ê': 'E', 'Ï': 'I', 'Ö': 'O', 'Ü': 'U'
7 }
8
9 def remover_acentos(input_str):
10     return ''.join(mapeo_acentos.get(letra, letra) for letra in input_str)
11 for col in df.columns:
12     new_col = remover_acentos(col)
13     df = df.withColumnRenamed(col, new_col.replace(" ", "_").lower())
14
15 df.printSchema() # Imprimir las columnas y el tipo de dato

```

df: pyspark.sql.dataframe.DataFrame = [fecha\_reporte\_web: string, id\_de\_caso: integer ... 21 campos adicionales]

```

root
|-- fecha_reporte_web: string (nullable = true)
|-- id_de_caso: integer (nullable = true)
|-- fecha_de_notificacion: string (nullable = true)
|-- codigo_divipola_departamento: integer (nullable = true)
|-- nombre_departamento: string (nullable = true)
|-- codigo_divipola_municipio: integer (nullable = true)
|-- nombre_municipio: string (nullable = true)
|-- edad: integer (nullable = true)

```

```

1 # Seleccionar los valores unicos en la columna nombre_departamento
2
3 from pyspark.sql.functions import col
4
5 departamentos = df.select("nombre_departamento").distinct().orderBy(col("nombre_departamento")).show(33)

```

(2) trabajos de Spark

```

+-----+
|nombre_departamento|
+-----+
|      AMAZONAS      |
|      ANTIOQUIA      |
|      ARAUCA        |
|      ATLANTICO      |
|      BARRANQUILLA   |
|      BOGOTA         |
|      BOLIVAR        |
|      BOYACA         |
|      CALDAS         |
|      CAQUETA        |
|      CARTAGENA      |
|      CASANARE       |
|      CAUCA          |
|      CESAR          |
|      CHOCO          |
|      CORDOBA        |
|      CUNDINAMARCA   |
|      GUANTANAMO     |

```

```

1 # Filtrar datos (ejemplo: casos en un departamento específico) Se convierte a pandas para una mejor visibilidad
2 df.filter(df["nombre_departamento"] == "ANTIOQUIA").select("fecha_reporte_web", "id_de_caso", "fecha_de_notificacion",
3 "nombre_departamento", "nombre_municipio", "edad", "tipo_de_contagio").show(30)

```

► (1) trabajos de Spark

fecha_reporte_web	id_de_caso	fecha_de_notificacion	nombre_departamento	nombre_municipio	edad	tipo_de_contagio
9/3/2020 0:00:00	3	7/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	50	Importado
11/3/2020 0:00:00	4	9/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	55	Relacionado
11/3/2020 0:00:00	5	9/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	25	Relacionado
11/3/2020 0:00:00	6	10/3/2020 0:00:00	ANTIOQUIA	ITAGUI	27	Relacionado
14/3/2020 0:00:00	20	11/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	26	Relacionado
14/3/2020 0:00:00	21	11/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	28	Relacionado
14/3/2020 0:00:00	22	12/3/2020 0:00:00	ANTIOQUIA	RIONEGRO	36	Importado
15/3/2020 0:00:00	32	11/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	55	Importado
19/3/2020 0:00:00	106	19/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	44	Importado
19/3/2020 0:00:00	107	12/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	56	Importado
19/3/2020 0:00:00	108	17/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	57	Importado
20/3/2020 0:00:00	131	15/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	22	Importado
20/3/2020 0:00:00	133	16/3/2020 0:00:00	ANTIOQUIA	RIONEGRO	51	Relacionado
20/3/2020 0:00:00	134	17/3/2020 0:00:00	ANTIOQUIA	LA ESTRELLA	28	Relacionado
20/3/2020 0:00:00	135	17/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	44	Importado
20/3/2020 0:00:00	136	17/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	37	Relacionado
20/3/2020 0:00:00	137	17/3/2020 0:00:00	ANTIOQUIA	ENVIGADO	54	Importado
20/3/2020 0:00:00	141	17/3/2020 0:00:00	ANTIOQUIA	MEDELLIN	62	Importado

Comando ejecutado en 0,50 segundos -- por hectorgrankangr1111@gmail.com el 27/11/2023, 22:33:34 en «Hector Banilat's Cluster»

```

1
2 # Adicionar la columna categoria_edad para saber si es menor, adulto, o adulto mayor
3 def categorizar_edad(edad):
4     if edad < 18:
5         return "Menor"
6     elif edad <= 60:
7         return "Adulto"
8     else:
9         return "Mayor"
10
11 categoria_udf = udf(categorizar_edad, StringType())
12 df = df.withColumn("categoria_edad", categoria_udf(df["edad"]))

```

▼ df: pyspark.sql.dataframe.DataFrame

```

fecha_reporte_web: string
id_de_caso: integer
fecha_de_notificacion: string
codigo_divipola_departamento: integer
nombre_departamento: string
codigo_divipola_municipio: integer
nombre_municipio: string
edad: integer
unidad_de_medida_de_edad: integer
sexo: string
tipo_de_contagio: string
ubicacion_del_caso: string
estado: string
codigo_iso_del_pais: integer
nombre_del_pais: string
recuperado: string

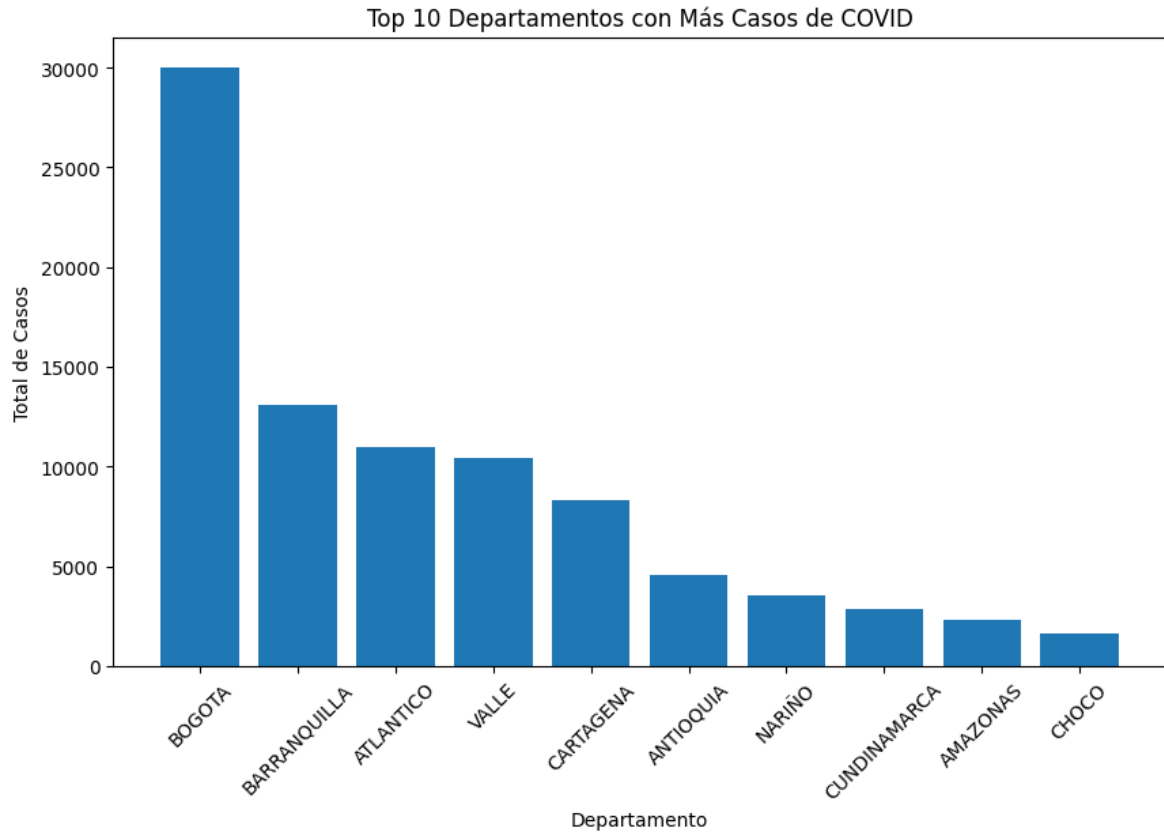
```

Cmd 7

```
1 # Registrar DataFrame como vista temporal para SparkSQL
2 import matplotlib.pyplot as plt
3
4 df.createOrReplaceTempView("covid")
5 # 3.1 Los 10 departamentos con más casos
6 top10_departamentos_by_sql = spark.sql("SELECT nombre_departamento, COUNT(*) as total FROM covid GROUP BY nombre_departamento ORDER BY total DESC LIMIT 10")
7 top10_departamentos_by_sql_pandas = top10_departamentos_by_sql.toPandas()
8
9 # Crear un gráfico de barras
10 plt.figure(figsize=(10, 6))
11 plt.bar(top10_departamentos_by_sql_pandas['nombre_departamento'], top10_departamentos_by_sql_pandas['total'])
12 plt.xlabel('Departamento')
13 plt.ylabel('Total de Casos')
14 plt.title('Top 10 Departamentos con Más Casos de COVID')
15 plt.xticks(rotation=45)
16 plt.show()
```

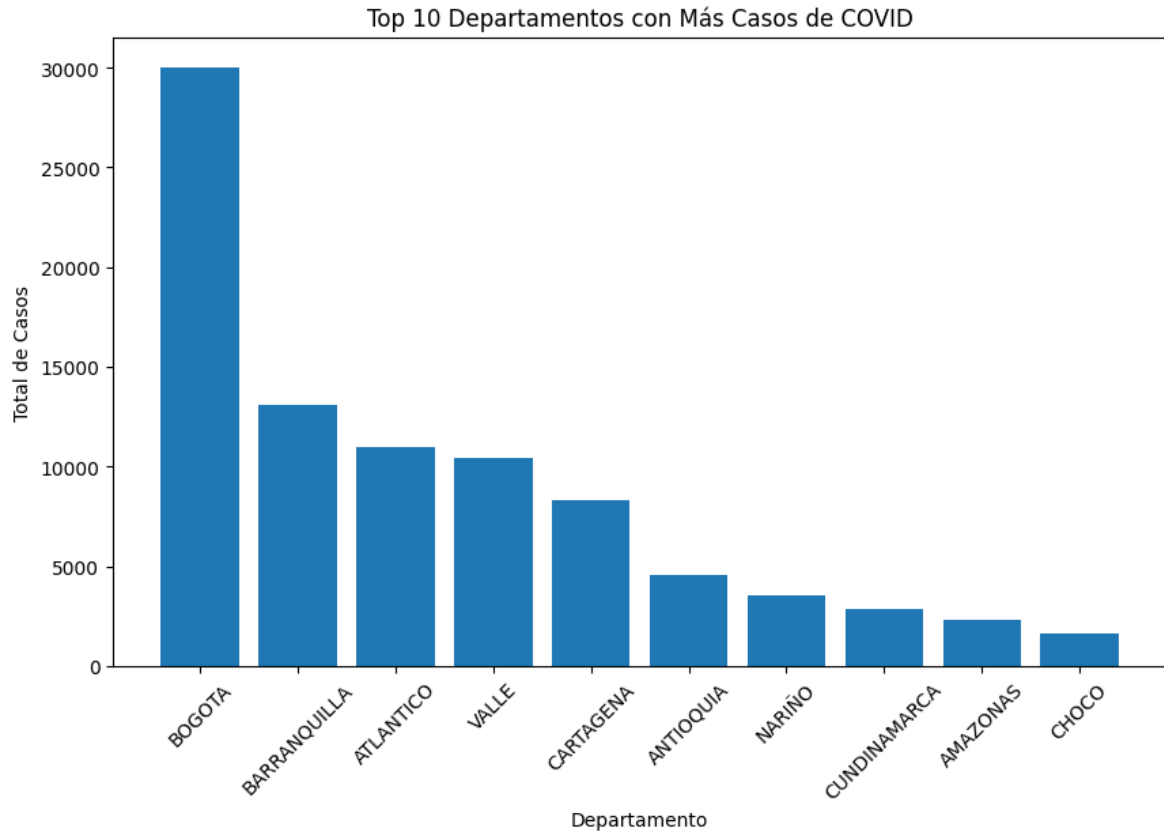
► (2) trabajos de Spark

► top10\_departamentos\_by\_sql: pyspark.sql.dataframe.DataFrame = [nombre\_departamento: string, total: long]



```
1 # 3.2 Las 10 ciudades con más casos
2 top_10_ciudades_by_sql = spark.sql("SELECT nombre_municipio, COUNT(*) as total FROM covid GROUP BY nombre_municipio ORDER BY total
3   DESC LIMIT 10")
4 top_10_ciudades_panda = top_10_ciudades_by_sql.toPandas()
5
6 plt.figure(figsize=(10, 6))
7 plt.bar(top10_departamentos_by_sql_pandas['nombre_departamento'], top10_departamentos_by_sql_pandas['total'])
8 plt.xlabel('Departamento')
9 plt.ylabel('Total de Casos')
10 plt.title('Top 10 Departamentos con Más Casos de COVID')
11 plt.xticks(rotation=45)
12 plt.show()
13
```





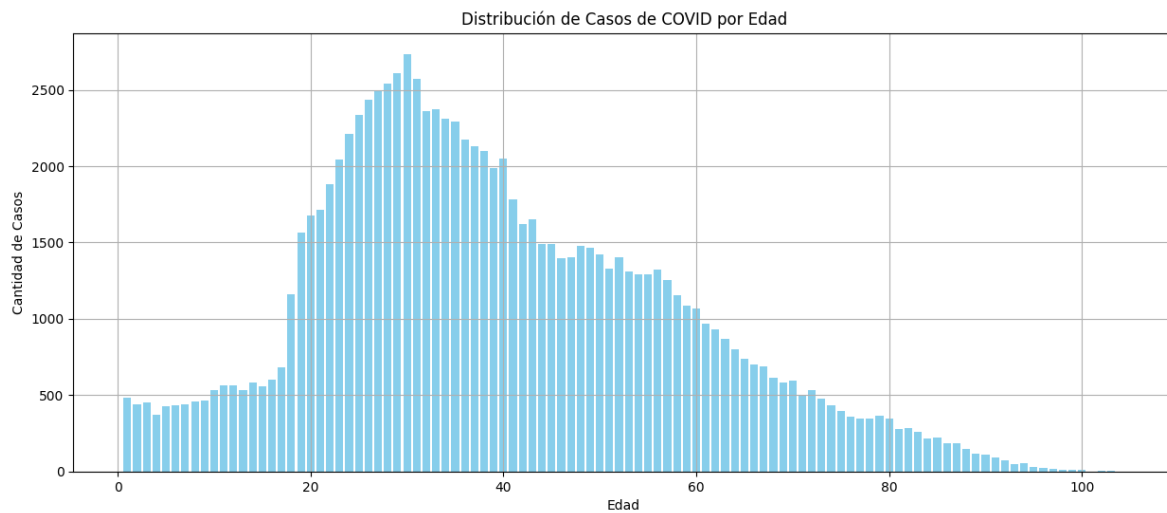
```
1 # 3.3 Dias con mas contagios
2 top_10_dias_contagio = spark.sql("SELECT fecha_de_diagnostico, COUNT(*) as total FROM covid GROUP BY fecha_de_diagnostico ORDER BY
total DESC LIMIT 10")
3 top_10_dias_contagio.show()
4
```

▶ (2) trabajos de Spark

▶ top\_10\_dias\_contagio: pyspark.sql.dataframe.DataFrame = [fecha\_de\_diagnostico: string, total: long]

```
+-----+-----+
|fecha_de_diagnostico|total|
+-----+-----+
| 26/6/2020 0:00:00| 4390|
| 27/6/2020 0:00:00| 4019|
| 28/6/2020 0:00:00| 3580|
| 25/6/2020 0:00:00| 3381|
| 19/6/2020 0:00:00| 3053|
| 18/6/2020 0:00:00| 3040|
| 23/6/2020 0:00:00| 3031|
| 22/6/2020 0:00:00| 2938|
| 21/6/2020 0:00:00| 2781|
| 24/6/2020 0:00:00| 2564|
+-----+-----+
```

Python    



El Siguiente comando guarda los datos en el bucket de GCP

Python

- ▶ (12) trabajos de Spark

Comando ejecutado en 39,90 segundos -- por hectorelgrankangri111@gmail.com el 27/11/2023, 22:34:58 en «hector banilat's Cluster»

lab-sparkhfbanilatq

Ubicación

Clase de almacenamiento

Acceso público

Protección

us-east1 (Carolina del Sur)

Standard

Sujeto a LCA de objeto

Ninguno

OBJETOS

CONFIGURACIÓN

PERMISOS

PROTECCIÓN

CICLO DE VIDA

OBSERVABILIDAD

INFORMES DE INVENTARIO

Depósitos > lab-sparkhfbanilatq > covid-19 > resultados

SUBIR ARCHIVOS

SUBIR CARPETA

CREAR CARPETA

TRANSFERIR LOS DATOS

ADMINISTRAR CONSERVACIONES

DESCARGAR

BORRAR

Filtrar solo por prefijo de nombre

Filtro

Filtrar objetos y carpetas

Mostrar datos borrados

<input type="checkbox"/>	Nombre	Tamaño	Tipo	Fecha de creación	Clase de almacenamiento	Última modificación	Acceso público	H
<input type="checkbox"/>	<a href="#">top10_departamentos_by_sql.csv/</a>	—	Carpeta	—	—	—	—	-
<input type="checkbox"/>	<a href="#">top_10_ciudades_by_sql.csv/</a>	—	Carpeta	—	—	—	—	-
<input type="checkbox"/>	<a href="#">top_10_dias_contagio.csv/</a>	—	Carpeta	—	—	—	—	-
<input type="checkbox"/>	<a href="#">total_casos_por_edad.csv/</a>	—	Carpeta	—	—	—	—	-
<input type="checkbox"/>	<a href="#">total_casos_porsexo.csv/</a>	—	Carpeta	—	—	—	—	-
<input type="checkbox"/>	<a href="#">total_casos_por_tipo_contagio.csv/</a>	—	Carpeta	—	—	—	—	-

Y un extra:

Cmd 15

Python

```
1 # por ultimo un extra, cuantas personas murieron
2 from pyspark.sql.functions import col
3
4 total_muertes_df = df.filter(col("fecha_de_muerte").isNotNull() & (col("fecha_de_muerte") != 'NULL') & (col("fecha_de_muerte") !=
5 print("Total de fallecidos:", total_muertes_df)
```

▶ (2) trabajos de Spark

Total de fallecidos: 5633

Comando ejecutado en 1,27 segundos -- por hector@grankangri111@gmail.com el 27/11/2023, 22:51:38 en «hector banilat's Cluster»

Y listo, con esto finalizamos el Laboratorio.