

Proceso de instalación de Apache HiveSQL, Apache Hadoop, MySQL, y creación de un sistema de computer visión para detectar 3 incidentes en una escalera

- **OpenJDK:** OpenJDK 21.
- **Hadoop:** Apache Hadoop 3.3.4.
- **Apache Hive:** Apache Hive 4.0.0.

1. Instalar Java (JDK)

```
sudo apt-get install openjdk-8-jdk
java --version
```

```
arojaspa@UBUNTUTESIS:~$ java --version
openjdk 21.0.4 2024-07-16
OpenJDK Runtime Environment (build 21.0.4+7-Ubuntu-1ubuntu222.04)
OpenJDK 64-Bit Server VM (build 21.0.4+7-Ubuntu-1ubuntu222.04, mixed mode, sharing)
arojaspa@UBUNTUTESIS:~$
```

2. Apache hadoop versión 3.4.0

```
wget https://downloads.apache.org/hadoop/common/hadoop-3.4.0/hadoop-3.4.0.tar.gz
tar -xzf hadoop-3.4.0.tar.gz
sudo mv hadoop-3.4.0 /usr/local/Hadoop
nano ~/.bashrc
echo "export HADOOP_HOME=/usr/local/Hadoop" >> ~/.bashrc
echo "export PATH=\$PATH:\$HADOOP_HOME/bin" >> ~/.bashrc
source ~/.bashrc
cd /usr/local/Hadoop/etc/Hadoop
```

```
nano hadoop-env.sh
```

Busca la línea que dice export JAVA_HOME= y agrégle la ruta de Java

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

formatear los namenodes

```
hdfs namenode -format
```

```
hadoop version
```

Configuración para trabajar standalone

Verificar permisos de escritura en los directorios de hadoop

```
sudo chown -R arojaspa:hadoop /usr/local/hadoop
sudo chown -R arojaspa:hadoop/logs
```

detener todos los servicios

```
$HADOOP_HOME/sbin/stop-dfs.sh
$HADOOP_HOME/sbin/stop-yarn.sh
```

Validar servicios en ejecución

```
jps
```

```
nano $HADOOP_HOME/etc/hadoop/hadoop-env.sh
export HADOOP_NICENESS=0
```

```
nano $HADOOP_HOME/etc/hadoop/core-site.xml
```

Validar que el archive este configurado así:

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

Usar hadoop en pseudo-distribuido (standalone)

```
nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml
```

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

Verificar si existe el archivo mapred-site.xml, sino, se crea uno así:

```
cp $HADOOP_HOME/etc/hadoop/mapred-site.xml.template
$HADOOP_HOME/etc/hadoop/mapred-site.xml
```

```
nano $HADOOP_HOME/etc/hadoop/mapred-site.xml
```

```
<configuration>
  <property>
```

```
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

`$HADOOP_HOME/sbin/start-dfs.sh`

- **NameNode UI:** <http://localhost:9870> para monitorear HDFS.
- **ResourceManager UI:** <http://localhost:8088> para gestionar aplicaciones y recursos.
- **SecondaryNameNode UI:** <http://localhost:9868> para verificar el estado del respaldo del NameNode.
- **YARN Application History:** <http://localhost:8188> para el historial de aplicaciones.

Para solucionar el error `InconsistentFSStateException: Directory /tmp/hadoop-arojaspa/dfs/name is in an inconsistent state: storage directory does not exist or is not accessible`.

Verificar si el Directorio existe:

```
ls -ld /tmp/hadoop-arojaspa/dfs/name
```

```
crear el Directorio sino existe
sudo mkdir -p /tmp/hadoop-arojaspa/dfs/name
sudo chown -R arojaspa:hadoop /tmp/hadoop-arojaspa/dfs/name
sudo chmod 755 /tmp/hadoop-arojaspa/dfs/name
```

```
revisar hdfs-site.xml
nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml
```

```
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:///tmp/hadoop-arojaspa/dfs/name</value>
</property>
```

Formatear el namenode

```
hdfs namenode -format
```

```
$HADOOP_HOME/sbin/stop-dfs.sh
$HADOOP_HOME/sbin/start-dfs.sh
```

3. Apache HiveQL

```
wget https://downloads.apache.org/hive/hive-4.0.0/apache-hive-4.0.0-bin.tar.gz
```

```
tar -xzf apache-hive-4.0.0-bin.tar.gz
```

```
sudo mv apache-hive-4.0.0-bin /usr/local/hive
```

```
nano ~/.bashrc
```

```
export HIVE_HOME=/usr/local/hive
```

```
export PATH=$PATH:$HIVE_HOME/bin
```

```
source ~/.bashrc
```

para crear estos archivos es necesario que hadoop este arriba

```
hdfs dfs -mkdir -p /user/hive/warehouse
```

```
hdfs dfs -chmod g+w /user/hive/warehouse
```

```
hdfs dfs -mkdir -p /user/hive/warehouse/scratchdir
```

```
hdfs dfs -chmod g+w /user/hive/warehouse/scratchdir
```

```
hdfs dfs -mkdir -p /user/hive/warehouse
```

```
hdfs dfs -chmod 777 /user/hive/warehouse
```

```
hdfs dfs -mkdir /tmp
```

```
hdfs dfs -chmod 777 /tmp
```

```
hdfs dfs -chmod g+w /user
```

```
hdfs dfs -chmod g+wx /user
```

```
hdfs dfs -chmod g+w /tmp
```

```
hdfs dfs -chmod g+wx /tmp
```

```
hdfs dfs -mkdir -p /cursobsg/pruebaseventos
```

```
hdfs dfs -chmod g+w /cursobsg
```

```
hdfs dfs -chmod g+wx /cursobsg
```

Eliminar la base de datos metastore_db para limpiar cualquier instalación previa y comenzar desde cero. Si no tienes datos importantes, puedes eliminar la base de datos para realizar una nueva instalación

```
rm -rf metastore_db
```

Reiniciar el proceso de inicialización del esquema:

```
schematool -dbType derby -initSchema
```

```
inicializar hiveserver  
$HIVE_HOME/bin/hiveserver2 &
```

4. Usar MySQL como metastore y base de datos de hive

```
sudo apt install mysql-server  
sudo systemctl status mysql  
mysql --version
```

```
sudo mysql -u root -p  
CREATE DATABASE metastore_db;  
CREATE USER 'mysqlremote'@'localhost' IDENTIFIED BY '123.Abc*.*';  
GRANT ALL PRIVILEGES ON metastore_db.* TO 'mysqlremote'@'localhost';  
FLUSH PRIVILEGES;
```

[How to install MySQL on Ubuntu 22.04 OS | Data Engineering | RDBMS | Part 8 | DM | DataMaking \(youtube.com\)](#)

[How to Install Hadoop on Ubuntu in VirtualBox | Ubuntu tutorials | Data Engineering | IvyProSchool \(youtube.com\)](#)

[Configure Hive on Hadoop in Ubuntu | Data Engineering Tutorial for Beginners | Ivy Pro School \(youtube.com\)](#)

```
nohup hive --service metastore &  
netstat -an | grep 9083  
Puerto e escucha de hive con mysql
```

```
$HIVE_HOME/bin/hive --service metastore &  
$HIVE_HOME/bin/hive --service hiveserver2 &
```

```
$HIVE_HOME/bin/beeline -u jdbc:hive2://localhost:10000
```

```
$HIVE_HOME/bin/beeline -u "jdbc:hive2://localhost:10000/valkyriai" -n arojaspa -p  
123.Abc*.*
```

```
beeline -u jdbc:hive2://
```

```
netstat -tuln | grep 10000  
sudo lsof -i :10000
```

```
<name>hive.server2.authentication</name>  
<value>NONE</value>
```

➔ NOSASL

hdfs dfsadmin -report

5. Validar si en el ambiente virtual se encuentras las librerías, sino se deben instalar en Ubuntu

pip install pandas sqlalchemy pyhive schedule jaydebeapi

6. Resultados Pruebas

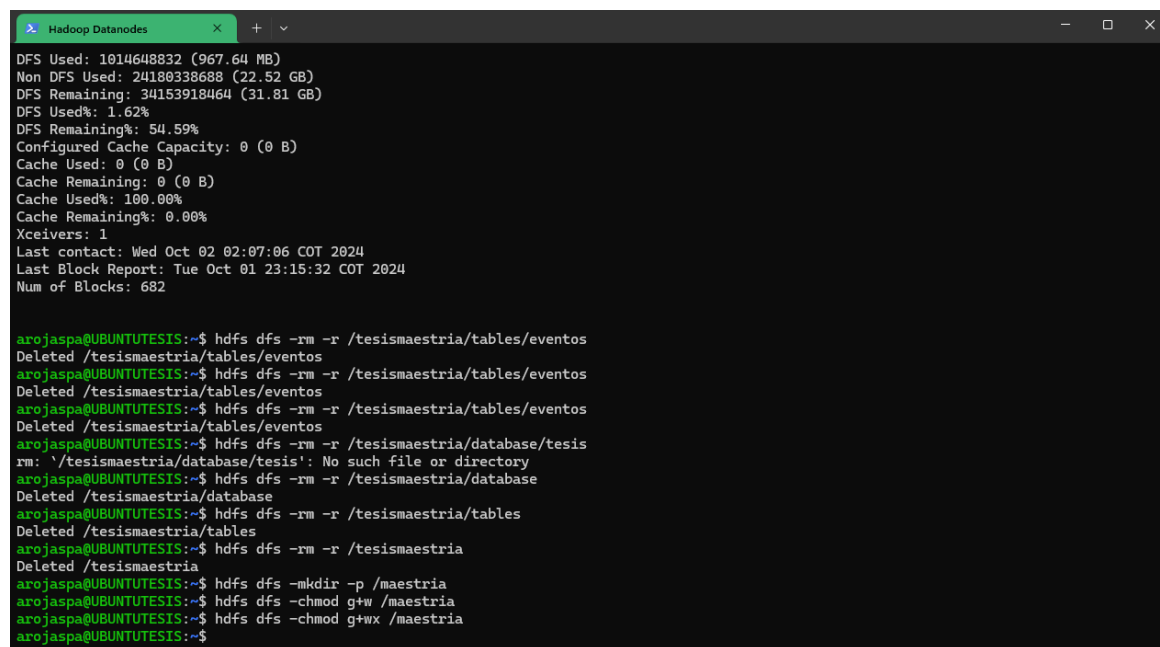
Creacion de directorios para almacenar la información en Hadoop:

Directorio de trabajo

hdfs dfs -mkdir -p /cursobsg

hdfs dfs -chmod g+w /cursobsg

hdfs dfs -chmod g+wx /cursobsg



```
Hadoop Datanodes
DFS Used: 1014648832 (967.64 MB)
Non DFS Used: 24180338688 (22.52 GB)
DFS Remaining: 34153918464 (31.81 GB)
DFS Used%: 1.62%
DFS Remaining%: 54.59%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 1
Last contact: Wed Oct 02 02:07:06 COT 2024
Last Block Report: Tue Oct 01 23:15:32 COT 2024
Num of Blocks: 682

arojaspa@UBUNTUTESIS:~$ hdfs dfs -rm -r /tesismaestria/tables/eventos
Deleted /tesismaestria/tables/eventos
arojaspa@UBUNTUTESIS:~$ hdfs dfs -rm -r /tesismaestria/tables/eventos
Deleted /tesismaestria/tables/eventos
arojaspa@UBUNTUTESIS:~$ hdfs dfs -rm -r /tesismaestria/tables/eventos
Deleted /tesismaestria/tables/eventos
arojaspa@UBUNTUTESIS:~$ hdfs dfs -rm -r /tesismaestria/database/tesis
rm: '/tesismaestria/database/tesis': No such file or directory
arojaspa@UBUNTUTESIS:~$ hdfs dfs -rm -r /tesismaestria/database
Deleted /tesismaestria/database
arojaspa@UBUNTUTESIS:~$ hdfs dfs -rm -r /tesismaestria/tables
Deleted /tesismaestria/tables
arojaspa@UBUNTUTESIS:~$ hdfs dfs -rm -r /tesismaestria
Deleted /tesismaestria
arojaspa@UBUNTUTESIS:~$ hdfs dfs -mkdir -p /maestria
arojaspa@UBUNTUTESIS:~$ hdfs dfs -chmod g+w /maestria
arojaspa@UBUNTUTESIS:~$ hdfs dfs -chmod g+wx /maestria
arojaspa@UBUNTUTESIS:~$
```

Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

Browse Directory

Show
25
entries
Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	drwxrwxr-x	arojaspa	supergroup	0 B	Oct 02 02:33	0	0 B	maestria	
<input type="checkbox"/>	drwx-wx-wx	arojaspa	supergroup	0 B	Oct 01 00:40	0	0 B	tmp	
<input type="checkbox"/>	drwxrwxr-x	arojaspa	supergroup	0 B	Oct 01 00:08	0	0 B	user	

Showing 1 to 3 of 3 entries

Previous
1
Next

Hadoop, 2022.

Directorio para la base de datos

hdfs dfs -mkdir -p /cursobsg /database

Asignado permisos de acceso a los directorios de hadoop

hdfs dfs -chmod g+w /cursobsg /database

hdfs dfs -chmod g+wx /cursobsg /database

```

Hadoop Datanodes
arojaspa@UBUNTUTESIS:~$ hdfs dfs -mkdir -p /maestria/database
arojaspa@UBUNTUTESIS:~$ hdfs dfs -chmod g+w /maestria/database
arojaspa@UBUNTUTESIS:~$ hdfs dfs -chmod g+wx /maestria/database
arojaspa@UBUNTUTESIS:~$

```

Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

Browse Directory

Show
25
entries
Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	drwxrwxr-x	arojaspa	supergroup	0 B	Oct 02 02:33	0	0 B	database	

Showing 1 to 1 of 1 entries

Previous
1
Next

Hadoop, 2022.

Directorio para las tablas de la base de datos

```
hdfs dfs -mkdir -p /cursobsg /tables
```




Asignado permisos de acceso a los directorios de hadoop

```
hdfs dfs -chmod g+w /cursobsg /tables
```



```
hdfs dfs -chmod g+wx /cursobsg /tables
```

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities ▾

Browse Directory

/maestria   

Show entries Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	drwxrwxr-x	arojaspa	supergroup	0 B	Oct 02 02:33	0	0 B	database	
<input type="checkbox"/>	drwxr-xr-x	arojaspa	supergroup	0 B	Oct 02 02:36	0	0 B	tables	

Showing 1 to 2 of 2 entries Previous **1** Next

Hadoop, 2022.

Creación de la base de datos (cambiar maestría por cursobsg)

```
1 -- Base de datos
2 CREATE DATABASE IF NOT EXISTS tesis
3 COMMENT 'This is an external database'
4 LOCATION '/maestria/database/tesis';
```


Usando la consola de HiveQL:

```
HiveQL
0: jdbc:hive2://localhost:10000/default> -- Base de datos
0: jdbc:hive2://localhost:10000/default> CREATE DATABASE IF NOT EXISTS tesis
. . . . .> COMMENT 'This is an external database'
. . . . .> LOCATION '/maestria/database/tesis';
INFO : Compiling command(queryId=arojaspa_20241002034928_bf4d9404-da78-44b6-bc8d-127507ecd453): CREATE DATABASE
IF NOT EXISTS tesis
COMMENT 'This is an external database'
LOCATION '/maestria/database/tesis'
INFO : Concurrency mode is disabled, not creating a lock manager
INFO : Semantic Analysis Completed (retrial = false)
INFO : Returning Hive schema: Schema(fieldSchemas:null, properties:null)
INFO : Completed compiling command(queryId=arojaspa_20241002034928_bf4d9404-da78-44b6-bc8d-127507ecd453); Time
taken: 0.009 seconds
INFO : Concurrency mode is disabled, not creating a lock manager
INFO : Executing command(queryId=arojaspa_20241002034928_bf4d9404-da78-44b6-bc8d-127507ecd453): CREATE DATABASE
IF NOT EXISTS tesis
COMMENT 'This is an external database'
LOCATION '/maestria/database/tesis'
INFO : Starting task [Stage-0:DDL] in serial mode
INFO : Completed executing command(queryId=arojaspa_20241002034928_bf4d9404-da78-44b6-bc8d-127507ecd453); Time
taken: 0.03 seconds
INFO : OK
INFO : Concurrency mode is disabled, not creating a lock manager
No rows affected (0.045 seconds)
0: jdbc:hive2://localhost:10000/default> |

HiveQL
0: jdbc:hive2://localhost:10000/default> show databases;
INFO : Compiling command(queryId=arojaspa_20241002035044_216cddb-8956-4ae9-bf7e-7c0d2bbab6f2): show databases
INFO : Concurrency mode is disabled, not creating a lock manager
INFO : Semantic Analysis Completed (retrial = false)
INFO : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:database_name, type:string, comment:from de
serializer)], properties:null)
INFO : Completed compiling command(queryId=arojaspa_20241002035044_216cddb-8956-4ae9-bf7e-7c0d2bbab6f2); Time
taken: 0.017 seconds
INFO : Concurrency mode is disabled, not creating a lock manager
INFO : Executing command(queryId=arojaspa_20241002035044_216cddb-8956-4ae9-bf7e-7c0d2bbab6f2): show databases
INFO : Starting task [Stage-0:DDL] in serial mode
INFO : Completed executing command(queryId=arojaspa_20241002035044_216cddb-8956-4ae9-bf7e-7c0d2bbab6f2); Time
taken: 0.013 seconds
INFO : OK
INFO : Concurrency mode is disabled, not creating a lock manager
+-----+
| database_name |
+-----+
| default      |
| tesis       |
+-----+
2 rows selected (0.039 seconds)
0: jdbc:hive2://localhost:10000/default> |
```

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities ▾

Browse Directory

Show entries Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	arojaspa	supergroup	0 B	Oct 02 03:49	0	0 B	tesis	<input type="checkbox"/>

Showing 1 to 1 of 1 entries Previous **1** Next

Creación de la tabla de control de eventos identificados (cambiar maestría por cursobsg)

```
1 -- Tablas
2 CREATE EXTERNAL TABLE eventos (
3     dispositivo STRING,
4     tipoinfraccion STRING,
5     imagen STRING,
6     ubicacion STRING,
7     zonainterres STRING,
8     fechahora STRING
9 )
10 ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
11 STORED AS TEXTFILE
12 LOCATION '/maestria/tables/eventos';
```

```
HiveQL
0: jdbc:hive2://localhost:10000/default> -- Tablas
0: jdbc:hive2://localhost:10000/default> CREATE EXTERNAL TABLE eventos (
    dispositivo STRING,
    tipoinfraccion STRING,
    imagen STRING,
    ubicacion STRING,
    zonainterres STRING,
    fechahora STRING
)
    ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    STORED AS TEXTFILE
    LOCATION '/maestria/tables/eventos';|
```

```
HiveQL
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/maestria/tables/eventos'
INFO : Concurrency mode is disabled, not creating a lock manager
INFO : Semantic Analysis Completed (retrial = false)
INFO : Returning Hive schema: Schema(fieldSchemas:null, properties:null)
INFO : Completed compiling command(queryId=arojaspa_20241002035447_f449ab1f-fc60-4f9c-98b3-9702ba47031e); Time
taken: 0.012 seconds
INFO : Concurrency mode is disabled, not creating a lock manager
INFO : Executing command(queryId=arojaspa_20241002035447_f449ab1f-fc60-4f9c-98b3-9702ba47031e): CREATE EXTERNAL
TABLE eventos (
dispositivo STRING,
tipoinfraccion STRING,
imagen STRING,
ubicacion STRING,
zonaintereres STRING,
fechahora STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/maestria/tables/eventos'
INFO : Starting task [Stage-0:DDL] in serial mode
INFO : Completed executing command(queryId=arojaspa_20241002035447_f449ab1f-fc60-4f9c-98b3-9702ba47031e); Time
taken: 0.028 seconds
INFO : OK
INFO : Concurrency mode is disabled, not creating a lock manager
No rows affected (0.047 seconds)
0: jdbc:hive2://localhost:10000/default> |
```

Script de ejecución del job que procesa la instrucción INSERT en la base de datos

```
1 import pandas as pd
2 import time
3 import schedule
4 from pyhive import hive
5 import os
6 import argparse
7
8 '''
9 Ingeniero: Andrés Felipe Rojas Parra
10 Maestria en Big Data y Data Science
11
12 Descripcion del script:
13 Script para automatizar el envio de informacion al sistema de big data (HiveQL y Hadoop)
14
15 Keypoint:
16 table_name: Nombre de la tabla donde se almacenará la informacion de eventos
17 csvalertpath: Ruta donde esta el archivo generado por la NVIDIA Jetson Nano de los eventos capturados por el modelo.
18 hiveqlhost: Direccion IP o url de conexion al servidor HiveQL
19 hiveiport: Puerto de conexion al servidor de HiveQL
20
21 '''
22
23
24 def validate_csv_sent_file(file_path):
25     '''
26     Funcion por validar que exista archivo
27
28     Parametros:
29     file_path: Ruta del archivo csv a validar.
30
31     Retorna un dataframe
32     '''
33     bReturn = False
34
35     if os.path.exists(file_path):
36         bReturn = True
37     else:
38         bReturn = False
39     return bReturn
40
41 def delete_csv_sent_file(file_path):
42
43     bReturn = False
44     if os.path.exists(file_path):
45         os.remove(file_path)
46         print(f"El archivo {file_path} ha sido borrado.")
47         bReturn = True
48     else:
49         print(f"El archivo {file_path} no existe.")
50         bReturn = False
51
52     return bReturn
53
54 def read_csv_file(file_path):
55     '''
56     Funcion por leer archivos
57
58     Parametros:
59     file_path: Ruta del archivo csv con las alertas.
60
61     Retorna un dataframe
62     '''
63
64     return pd.read_csv(file_path)
```

```

1 def remove_duplicates(dataframe):
2     '''
3     Funcion para eliminar los registros duplicados
4
5     Parametros:
6     dataframe: Dataframe del archivo leido
7
8     Retorna un dataframe sin duplicados
9     '''
10
11     clean_data = dataframe.drop_duplicates(subset=['fechahora'])
12     return clean_data
13
14 def insert_into_hive(clean_data):
15     '''
16     Funcion para enviar la informacion a la base de datos
17
18     Parametro:
19     clean_data: Dataframe sin duplicados
20     '''
21
22     hiveqlhost = '10.0.0.103'
23     hiveipport= 10000
24     hiveuser='arojaspa'
25     hiveauth='NONE'
26     hivedatabase='tesis'
27     table_name = 'eventos'
28
29     conn = hive.Connection(host=hiveqlhost, port=hiveipport, username=hiveuser, database=hivedatabase, auth=hiveauth)
30     cursor = conn.cursor()
31
32     print("Enviando Datos..")
33
34     for index, row in clean_data.iterrows():
35         query = f"""INSERT INTO {table_name} (dispositivo,tipoinfraccion,imagen,ubicacion,zonainteres,fechahora) VALUES
36         ('{row['dispositivo']}', '{row['tipoinfraccion']}', '{row['imagen'].replace('\\',
37         '/')}', '{row['ubicacion']}', '{row['zonainteres']}', '{row['fechahora']}')"""
38         print(query) # Para validar el registro enviado
39         cursor.execute(query)
40
41     # Commit los cambios
42     conn.commit()
43
44     # cerrando el cursor y la conexion
45     cursor.close()
46     conn.close()

```

```

1 def process_csv_and_insert_into_hive(debug):
2     """
3     Funcion que se utiliza para activar el job. Se ejecuta segun la configuracion que se haya realizado el job
4     """
5     try:
6         csvalertpath = 'eventosdetectadosnvidia.csv'
7         csvsentpath = 'eventosdetectadosnvidia_inHiveQL.csv'
8         bSentFile = False
9         bNvidiaFile = False
10        bError = False
11
12        if validate_csv_sent_file(csvsentpath):
13            df_sent = read_csv_file(csvsentpath)
14            bSentFile = True
15            if debug:
16                print("Se validó el archivo de eventos a enviados")
17        else:
18            if debug:
19                print("No se encontró el archivo de elementos enviados")
20
21        if validate_csv_sent_file(csvalertpath):
22            nvidia = read_csv_file(csvalertpath)
23            bNvidiaFile = True
24            if debug:
25                print("Se validó el archivo de eventos a enviar")
26        else:
27            if debug:
28                print("No se encontró el archivo de elementos a enviar")
29
30        if not bSentFile and not bNvidiaFile:
31            print("No hay informacion para ser procesada. Solicitando la cancelacion del Job.")
32            bError = True
33            raise RuntimeError()
34
35        elif not bSentFile and bNvidiaFile:
36            # Elimina duplicados de los eventos
37            if debug:
38                print("Removiendo duplicados de eventos a enviar")
39            clean_data = remove_duplicates(nvidia)
40
41        elif bSentFile and bNvidiaFile:
42            if debug:
43                print("Removiendo duplicados de eventos a enviar")
44
45            # Elimina duplicados de los eventos
46            nvidia_without_duplicates = remove_duplicates(nvidia)
47
48            if debug:
49                print("Comparando datos de eventos enviados vs los que se van a enviar")
50
51            # despues de limpiar duplicados, compara con el df enviados para solo enviar los nuevos eventos.
52            clean_data = nvidia_without_duplicates[~nvidia_without_duplicates.isin(df_sent).all(axis=1)]
53
54    except RuntimeError:
55        print('Error: El job fue cancelado por falta de informacion', )
56        schedule.cancel_job
57
58    finally:
59
60        if not bError:
61            print("Procesando informacion...")
62            rows, columns = clean_data.shape
63
64            if rows > 0:
65                insert_into_hive(clean_data)
66            else:
67                print("No hay datos para insertar en la base de datos...")
68
69            # Si el archivo existe, lo borra para crear uno nuevo. Si no existe, lo crea.
70            delete_csv_sent_file(csvsentpath)
71
72            # copia la informacion enviada a HiveQL en el dataframe df_sent
73            if not bSentFile:
74                df_sent = pd.DataFrame()
75
76            df_sent = pd.concat([df_sent,clean_data], ignore_index=True)
77
78            # Guarda la informacion en el archivo csv
79            df_sent.to_csv("eventosdetectadosnvidia_inHiveQL.csv", index=False)
80
81        else:
82            print('Inicializando otra instancia...', )

```

```
1 def main(debug):
2     '''
3     Funcion de inicio del script
4     '''
5
6     last = None
7
8     # Configuración del job
9     schedule.every(1).hours.do(lambda: process_csv_and_insert_into_hive(debug))
10
11     while True:
12
13         schedule.run_pending()
14
15         # Muestra la informacion de la siguiente ejecucion
16         next_run = schedule.next_run()
17         if next_run != last:
18             print(f"Next run: {next_run}")
19             last = next_run
20
21         time.sleep(1)
22
23
24 if __name__ == '__main__':
25
26     # Initialize the argument parser
27     parser = argparse.ArgumentParser(description="Arguments to setup the Camera Object")
28
29     parser.add_argument('--debug', dest='debug', action='store_true', help="To do debug")
30     parser.add_argument('--no-debug', dest='debug', action='store_false', help="To do debug")
31
32     # Parse the arguments
33     args = parser.parse_args()
34
35     if args.debug:
36         print("Debug in __Main__: ", args.debug)
37
38     main(args.debug)
```

Script de python que analiza 3 posibles tipos de accidentes (subir y bajar hablando por celular, subir y bajar con una laptop abierta, subir y bajar con una taza en la mano) que se pueden presentar en una escalera entre dos pisos de una oficina:

```
1 import cv2
2 import numpy as np
3 import tensorflow as tf
4 import tensorflow_hub as hub
5 from ultralytics import YOLO
6 from datetime import datetime
7 import os
8 import csv
9
10 class DeteccionInfracciones:
11     def __init__(self, modelo_yolo_path, carpeta_infracciones, archivo_csv, fuente=0, ancho_deseado=640, alto_deseado=480,
12 id_camara=1, zona="Zona 1"):
13         # Inicializar captura de video
14         self.cap = cv2.VideoCapture(fuente)
15
16         # Cargar el modelo YOLO
17         self.modelo_yolo = YOLO(modelo_yolo_path)
18
19         # Cargar el modelo MoveNet para detección de poses
20         self.modelo_movenet =
21         hub.load('https://tfhub.dev/google/movenet/multipose/lightning/1').signatures['serving_default']
22
23         # Cargar el clasificador de rostros
24         self.clasificador_rostros = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
25
26         # Configuración del sustractor de fondo
27         self.fgbg = cv2.createBackgroundSubtractorMOG2()
28         self.kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (3, 3))
29
30         # Dimensiones deseadas para el video redimensionado
31         self.ancho_deseado = ancho_deseado
32         self.alto_deseado = alto_deseado
33
34         # Indices de las clases de interés
35         self.clases_interes = {
36             0: 'persona',
37             41: 'objetomano',
38             63: 'laptop',
39             67: 'celular'
40         }
41
42         # Colores para cada clase
43         self.colores = {
44             'persona': (0, 255, 0), # Verde
45             'objetomano': (0, 128, 255), # Naranja
46             'celular': (255, 0, 0), # Azul
47             'laptop': (0, 0, 255) # Rojo
48         }
49
50         # Puntos del área de interés (ROI)
51         self.area_pts = np.array([[230, 70], [380, 70], [1060, self.alto_deseado], [-280, self.alto_deseado]])
52
53         # Carpeta donde guardar las infracciones
54         self.carpeta_infracciones = carpeta_infracciones
55         if not os.path.exists(self.carpeta_infracciones):
56             os.makedirs(self.carpeta_infracciones)
57
58         # Archivo CSV para guardar las infracciones
59         self.archivo_csv = archivo_csv
60         if not os.path.exists(self.archivo_csv):
61             self.crear_csv()
62
63         # Variables para el seguimiento y rastreo de infracciones
64         self.infracciones_guardadas = {} # Diccionario para llevar el registro de infracciones por persona
65
66         # Identificación de cámara y zona de infracción
67         self.id_camara = id_camara
68         self.zona = zona
69
70         # Bordes para conexiones de puntos clave (keypoints) en pose estimation
71         self.EDGES = {
72             (0, 1): 'm', (0, 2): 'c', (1, 3): 'm', (2, 4): 'c', (0, 5): 'm',
73             (0, 6): 'c', (5, 7): 'm', (7, 9): 'm', (6, 8): 'c', (8, 10): 'c',
74             (5, 6): 'y', (5, 11): 'm', (6, 12): 'c', (11, 12): 'y', (11, 13): 'm',
75             (13, 15): 'm', (12, 14): 'c', (14, 16): 'c'
76         }
```



```

1  def crear_csv(self):
2      with open(self.archivo_csv, mode='w', newline='') as file:
3          writer = csv.writer(file)
4          #writer.writerow(["dispositivo", "tipoinfraccion", "nombreimagedcapturada", "ubicacion", "zonainteres", "Fecha",
"Hora", "ID Camara", "Zona"])
5          writer.writerow(["dispositivo", "tipoinfraccion", "imagen", "ubicacion", "zonainteres", "fechahora"])
6
7  def guardar_infraccion_csv(self, etiqueta, ruta_archivo):
8      timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
9      #fecha, hora = timestamp.split('_')
10     with open(self.archivo_csv, mode='a', newline='') as file:
11         writer = csv.writer(file)
12         #writer.writerow([etiqueta, ruta_archivo, fecha, hora, self.id_camara, self.zona])
13         writer.writerow(["NVIDIATSS01", etiqueta, ruta_archivo, "Escaleras Universidad Piso 1 Ed Sistemas", self.zona,
timestamp])
14
15 def procesar_frame(self):
16     ret, frame = self.cap.read()
17     if not ret:
18         return None
19
20     # Redimensiona el frame
21     frame_redimensionado = cv2.resize(frame, (self.anchos_deseado, self.alto_deseado))
22     return frame_redimensionado
23
24 def aplicar_zona_interes(self, frame_redimensionado):
25     # Crear una máscara para la zona de interés
26     imAux = np.zeros(shape=(self.alto_deseado, self.anchos_deseado), dtype=np.uint8)
27     imAux = cv2.drawContours(imAux, [self.area_pts], -1, (255), -1)
28     zona_interes = cv2.bitwise_and(frame_redimensionado, frame_redimensionado, mask=imAux)
29
30     # Aplicar el sustractor de fondo a la zona de interés
31     fgmask = self.fgbg.apply(zona_interes)
32     fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_OPEN, self.kernel)
33     fgmask = cv2.dilate(fgmask, None, iterations=2)
34
35     return zona_interes
36
37 def detectar_infraccion(self, frame_redimensionado, zona_interes):
38     # Realizar detección en la zona de interés con YOLO
39     resultados = self.modelo_yolo(zona_interes)
40     for resultado in resultados:
41         detecciones_filtradas = [det for det in resultado.boxes if int(det.cls) in self.clases_interes]
42         for deteccion in detecciones_filtradas:
43             coordenadas = deteccion.xyxy[0].tolist()
44             x1, y1, x2, y2 = map(int, coordenadas)
45             clase = int(deteccion.cls)
46             etiqueta = self.clases_interes.get(clase, 'desconocido')
47             color = self.colores.get(etiqueta, (255, 255, 255))
48
49             # Verificar si la detección está dentro de la zona de interés
50             if cv2.pointPolygonTest(self.area_pts, ((x1 + x2) // 2, (y1 + y2) // 2), False) >= 0:
51                 cv2.rectangle(frame_redimensionado, (x1, y1), (x2, y2), color, 1)
52                 cv2.putText(frame_redimensionado, etiqueta, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 1)
53
54             # Guardar infracción solo si no ha sido registrada previamente
55             if etiqueta in ['celular', 'laptop', 'objetomano'] and (etiqueta not in self.infracciones_guardadas):
56                 # Guardar solo una captura por infracción por persona
57                 timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
58                 nombre_archivo = f'{etiqueta}_infraccion_{timestamp}.jpg'
59                 #ruta_archivo = os.path.join(self.carpeta_infracciones, nombre_archivo)
60                 ruta_archivo = nombre_archivo
61                 cv2.imwrite(ruta_archivo, frame_redimensionado)
62
63                 # Guardar infracción en CSV con los detalles adicionales
64                 self.guardar_infraccion_csv(etiqueta, ruta_archivo)
65
66                 # Marcar la infracción como guardada
67
68                 #Validar esto
69                 self.infracciones_guardadas[etiqueta] = timestamp
70
71     cv2.drawContours(frame_redimensionado, [self.area_pts], -1, (255, 0, 255), 1)
72     return frame_redimensionado

```

```

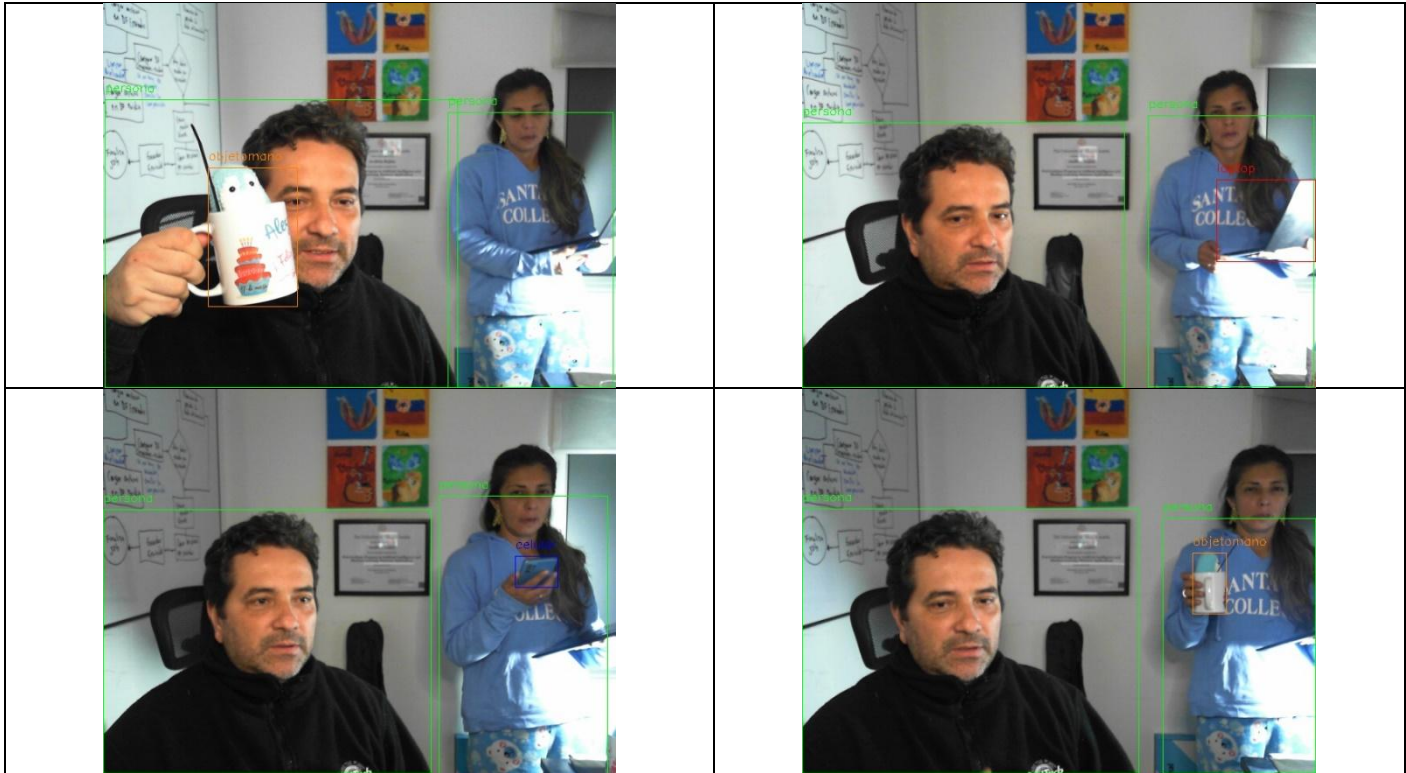
1  def detectar_rostros(self, frame):
2      gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
3      rostros = self.clasificador_rostros.detectMultiScale(gray_frame, scaleFactor=1.1, minNeighbors=5)
4      for (x, y, w, h) in rostros:
5          cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 255), 2)
6
7  def detectar_pose(self, frame):
8      img = tf.image.resize_with_pad(tf.expand_dims(frame, axis=0), 256, 256)
9      img = tf.cast(img, dtype=tf.int32)
10     resultados = self.modelo_movenet(img)
11     puntos = resultados['output_0'].numpy()[0, :, :51].reshape((6, 17, 3))
12
13     for persona in puntos:
14         self.dibujar_pose(frame, persona)
15
16 def dibujar_pose(self, frame, puntos):
17     for (i, j), color in self.EDGES.items():
18         if puntos[i][2] > 0.5 and puntos[j][2] > 0.5: # Solo dibujar si ambos puntos son visibles
19             color_rgb = self.colores.get(color, (255, 255, 255)) # Color por defecto
20             cv2.line(frame, (int(puntos[i][1]), int(puntos[i][0])), (int(puntos[j][1]), int(puntos[j][0])), color_rgb,
21 2)
22
23     for punto in puntos:
24         if punto[2] > 0.5: # Solo dibujar el punto si es visible
25             cv2.circle(frame, (int(punto[1]), int(punto[0])), 5, (0, 255, 255), -1)
26
27 def iniciar(self):
28     while True:
29         frame = self.procesar_frame()
30         if frame is None:
31             break
32
33         zona_interes = self.aplicar_zona_interes(frame)
34         self.detectar_infraccion(frame, zona_interes)
35         self.detectar_rostros(frame)
36         self.detectar_pose(frame)
37
38         cv2.imshow('Modulo escaleras', frame)
39
40         if cv2.waitKey(1) & 0xFF == 27: # Presiona ESC para salir
41             break
42
43     self.cap.release()
44     cv2.destroyAllWindows()
45
46 # Uso de la clase DeteccionInfracciones
47 if __name__ == "__main__":
48     deteccion = DeteccionInfracciones(modelo_yolo_path='yolov8n.pt',
49                                       carpeta_infracciones='imageneseventos',
50                                       archivo_csv='eventosdetectadosnvidia.csv',
51                                       fuente=2,
52                                       ancho_deseado=640,
53                                       alto_deseado=480,
54                                       id_camara=1,
55                                       zona="RIESGO MEDIO")
56     deteccion.iniciar()

```

Los resultados de este script son:

Imágenes capturadas





dispositivo	tipoinfraccion	imagen	ubicacion	zonaintereres	fechahora
NVIDIATSS01	celular	imageneseventos\celular_infraccion_2024-10-02_04-19-28.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:19
NVIDIATSS01	objetomano	imageneseventos\objetomano_infraccion_2024-10-02_04-19-40.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:19
NVIDIATSS01	laptop	imageneseventos\laptop_infraccion_2024-10-02_04-20-01.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:20
NVIDIATSS01	objetomano	imageneseventos\objetomano_infraccion_2024-10-02_04-22-06.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:22
NVIDIATSS01	celular	imageneseventos\celular_infraccion_2024-10-02_04-22-15.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:22
NVIDIATSS01	laptop	imageneseventos\laptop_infraccion_2024-10-02_04-22-31.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:22
NVIDIATSS01	laptop	imageneseventos\laptop_infraccion_2024-10-02_04-25-48.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:25
NVIDIATSS01	celular	imageneseventos\celular_infraccion_2024-10-02_04-25-53.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:25
NVIDIATSS01	objetomano	imageneseventos\objetomano_infraccion_2024-10-02_04-26-06.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:26
NVIDIATSS01	laptop	imageneseventos\laptop_infraccion_2024-10-02_04-27-22.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:27
NVIDIATSS01	celular	imageneseventos\celular_infraccion_2024-10-02_04-27-33.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:27
NVIDIATSS01	objetomano	imageneseventos\objetomano_infraccion_2024-10-02_04-27-52.jpg	Escaleras Universidad Piso 1 Ed Sistemas	RIESGO MEDIO	10/2/2024 4:27