Limpieza de los datos recibidos por el broker MQTT

Para recibir en tiempo real los datos enviados por los sensores, me conecto al broker MQTT a través del cliente Paho.

```python
# Creación del cliente MQTT
def on_connect(client, userdata, flags,rc):
    print('connected(%s)' %client._client_id)
    client.subscribe(topic='cultivo/sector_uno/tomate', qos=0)

def on_message(client, userdata, message):
    print('------------------------------')
    print('topic: %s' % message.topic)
    print('payload: %s' % message.payload)
    print('qos: %d' %message.qos)
    data_recibida.append(message.payload.decode('utf-8'))

def main():
    client = mqtt.Client(client_id='vladimir_mqtt', clean_session=False)
    client.on_connect = on_connect
    client.on_message = on_message
    #client.connect(host='127.0.0.1',port=1883)
    client.connect(host='broker.hivemq.com',port=1883)
    #client.loop_forever()
    client.loop_start()
    time.sleep(600)  # toma datos cada 60 segundos
    client.loop_stop()

if __name__ == '__main__':
    main()
```

```
connected(b'vladimir_mqtt')
------------------------------
topic: cultivo/sector_uno/tomate
payload: b'{"ppm": 264.6335, "Luz": 63.63859, "lluvia": 0.0, "Humedad_aire": 19, "Humedad_su
qos: 0
connected(b'vladimir_mqtt')
```

Esta información se organiza y se prepara en un dataframe para poder filtrar los datos duplicados, los que tienen datos faltantes y se organiza de forma ascendente por la fecha recibida

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 246 entries, 0 to 245
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   fecha          246 non-null    datetime64[ns]
 1   temperatura    246 non-null    float64
 2   ppm            246 non-null    float64
 3   luz            246 non-null    float64
 4   lluvia         246 non-null    float64
 5   humedad_aire   246 non-null    float64
 6   humedad_suelo  246 non-null    float64
dtypes: datetime64[ns](1), float64(6)
memory usage: 13.6 KB
```

```
# Eliminamos los datos duplicados
df.drop_duplicates(inplace=True)
df.duplicated().sum()
```

```python
# Conversión de string recibidas por MQTT a float
ppm_float = [float(x) for x in ppm]
luz_float = [float(x) for x in luz]
lluvia_float = [float(x) for x in lluvia]
humedad_aire_float = [float(x) for x in humedad_aire]
humedad_suelo_float = [float(x) for x in humedad_suelo]
temperatura_float = [float(x) for x in temperatura]
```

```python
# Convertimos la data recibida en un dataframe
data_total = {
    'fecha': fecha,
    'temperatura': temperatura_float,
    'ppm': ppm_float,
    'luz': luz_float,
    'lluvia': lluvia_float,
    'humedad_aire': humedad_aire_float,
    'humedad_suelo' : humedad_suelo_float
}
df = pd.DataFrame(data_total)
df.head()
```

	fecha	temperatura	ppm	luz	lluvia	humedad_aire	humedad_suelo
0	2024-06-03 20:43:43	27.0	264.6335	63.63859	0.0	19.0	25.00611
1	2024-06-03 20:43:46	27.0	352.6436	45.68986	0.0	19.0	0.00000
2	2024-06-03 20:43:48	27.0	230.3117	46.47130	0.0	19.0	0.00000
3	2024-06-03 20:43:50	27.0	507.3637	47.20391	0.0	19.0	0.00000
4	2024-06-03 20:43:53	27.0	342.4921	62.75947	0.0	19.0	0.00000

Luego el dataframe es convertido a formato json para ser almacenados en la base de datos Firebase donde se almacenarán todos los datos de los sensores y del registro del control de apertura y cierre de las válvulas.