The following document describes the test escenarios about possible failures on dataflow job.

Te pipeline constructed has the purpose to test the following escenarios:

1) An logical error during the execution of some element on the pcollection: Maybe there is an possibility that the constructed pipeline cant process the element received, maybe due the following reasons:
    a) some kafka message that can not be processed and raise an error
    b) Maybe there is an external component who change its behavior and causes an unexpected error when some elements are processed, like a call to some external api who responses something not expected, o takes too much time to give the response
2) An infrastructure error related to dataflow service, which may cause unexpected behaviors on the pipeline:
    a) Issues with GCP services related with dataflow: compute engine, cloud storage
    b) Accidental cancellation of the dataflow job

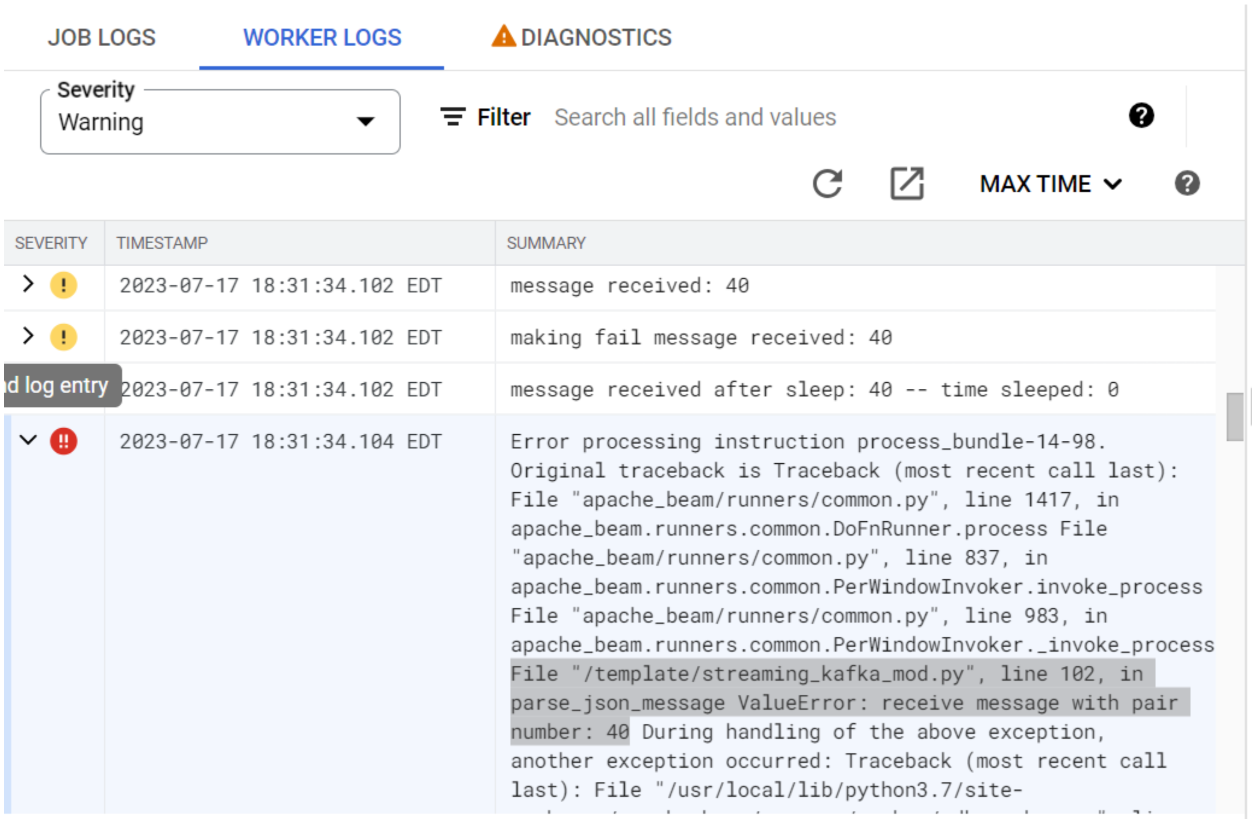
Acording to the previous defined escenarios some tests where maded on dataflow ussing the following pipeline:

Read_from_kafka —> test_transformation —> write_to_kafka

For the first escenario we have this test:

1) Run the test pipeline with:
```
gcloud dataflow flex-template run "test-kafka-resume-job" \
   --template-file-gcs-location "$TEMPLATE_PATH" \
   --max-workers 2 \
   --region "$REGION" \
   --parameters input_topic="$TOPIC" \
   --parameters output_topic="test-kafka-output-dataflow" \
   --parameters group_id="test-consumer-group" \
   --parameters bootstrap_servers="35.193.114.205:9092" \
   --parameters commit_offset_in_finalize=1 \
   --parameters with_metadata=0 \
   --parameters allow_fail=1 \
   --parameters start_read_time=0 \
   --parameters delay_time=60 \
   --parameters messages_per_delay=10000000 \
   --parameters messages_per_fail=10
```

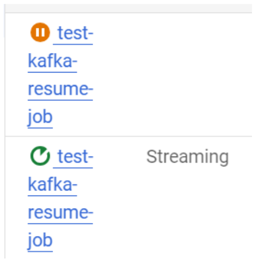
   We can get the following errors on the pipeline:



SEVERITY	TIMESTAMP	SUMMARY
!	2023-07-17 18:31:34.102 EDT	message received: 40
!	2023-07-17 18:31:34.102 EDT	making fail message received: 40
	2023-07-17 18:31:34.102 EDT	message received after sleep: 40 -- time sleeped: 0
!!	2023-07-17 18:31:34.104 EDT	Error processing instruction process_bundle-14-98. Original traceback is Traceback (most recent call last): File "apache_beam/runners/common.py", line 1417, in apache_beam.runners.common.DoFnRunner.process File "apache_beam/runners/common.py", line 837, in apache_beam.runners.common.PerWindowInvoker.invoke_process File "apache_beam/runners/common.py", line 983, in apache_beam.runners.common.PerWindowInvoker._invoke_process File "/template/streaming_kafka_mod.py", line 102, in parse_json_message ValueError: receive message with pair number: 40 During handling of the above exception, another exception occurred: Traceback (most recent call last): File "/usr/local/lib/python3.7/site-

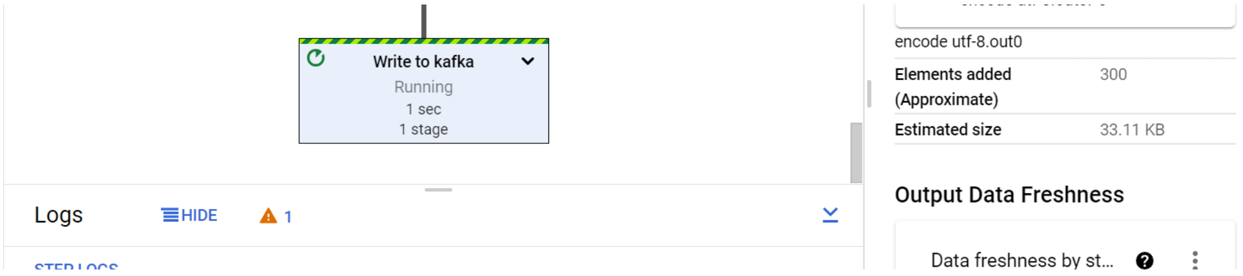
This error is caused intentionally for this test.

To solve this problem without any loose of data, we can update the dataflow job with allow_fail=0:



test-kafka-resume-job	
test-kafka-resume-job	Streaming

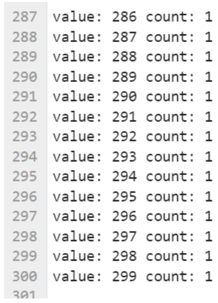
This disable the above error, and we can see how all elements are processed without any error:



encode utf-8.out0	
Elements added (Approximate)	300
Estimated size	33.11 KB

Conclusion: No matter the logic code issues or application issues that we can have on the future, if we are able to update the existing running pipeline, no data will be losed or duplicated

Consuming the output topic we can see that all messages has been sent only one time:

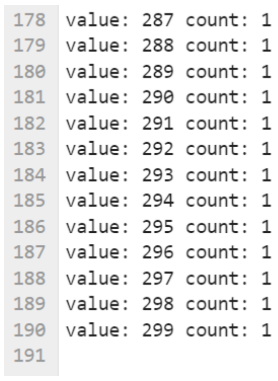


```
value: 286 count: 1
value: 287 count: 1
value: 288 count: 1
value: 289 count: 1
value: 290 count: 1
value: 291 count: 1
value: 292 count: 1
value: 293 count: 1
value: 294 count: 1
value: 295 count: 1
value: 296 count: 1
value: 297 count: 1
value: 298 count: 1
value: 299 count: 1
```

For the second scenario we can do:

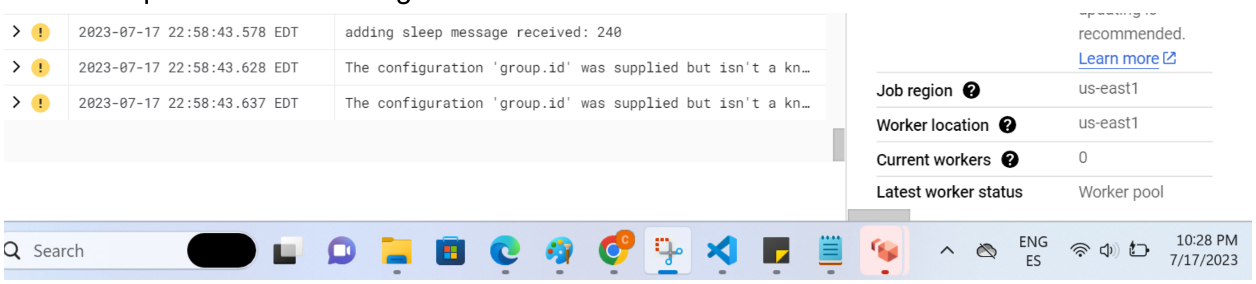
2) Run the following command

Here we can see that I delete the cluster and the tmp files when not all data was processed an store on the output topic



```
value: 287 count: 1
value: 288 count: 1
value: 289 count: 1
value: 290 count: 1
value: 291 count: 1
value: 292 count: 1
value: 293 count: 1
value: 294 count: 1
value: 295 count: 1
value: 296 count: 1
value: 297 count: 1
value: 298 count: 1
value: 299 count: 1
```

I leave the process stuck during half hour



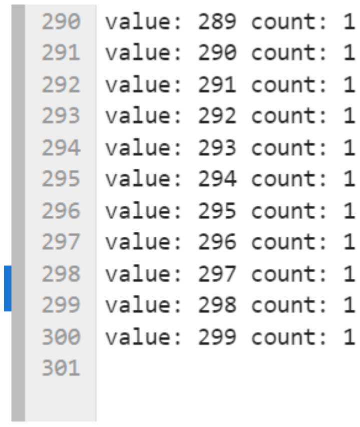
```
2023-07-17 22:58:43.578 EDT   adding sleep message received: 240
2023-07-17 22:58:43.628 EDT   The configuration 'group.id' was supplied but isn't a kn…
2023-07-17 22:58:43.637 EDT   The configuration 'group.id' was supplied but isn't a kn…
```

recommended.
Learn more

Job region	us-east1
Worker location	us-east1
Current workers	0
Latest worker status	Worker pool

And now I update exactly the same job, no changes

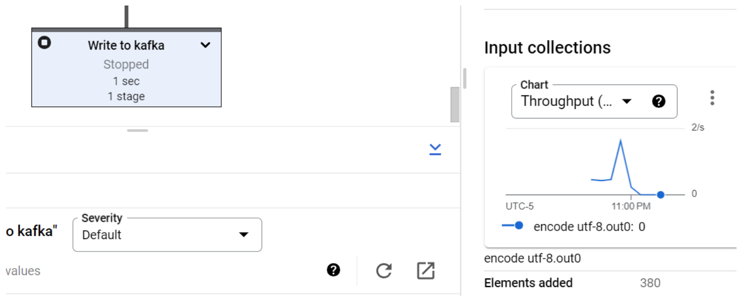
The updated job continue processing the elements that wasnt procesed before when I delete the instance group and the gcs staging, It only process the elements that wasnt processed before so no data was lossed or duplicated. (300 messages received, counted one time)



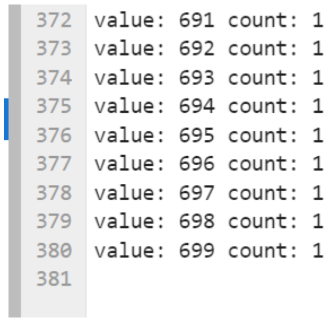
```
value: 289 count: 1
value: 290 count: 1
value: 291 count: 1
value: 292 count: 1
value: 293 count: 1
value: 294 count: 1
value: 295 count: 1
value: 296 count: 1
value: 297 count: 1
value: 298 count: 1
value: 299 count: 1
```

Accidental cancelation:

I send another 400 messages and cancel the process just when I receive the first batch of writes on kafka output topic, 80 messages were received:

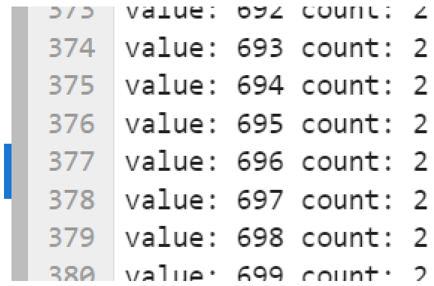


encode utf-8.out0
Elements added 380



```
value: 691 count: 1
value: 692 count: 1
value: 693 count: 1
value: 694 count: 1
value: 695 count: 1
value: 696 count: 1
value: 697 count: 1
value: 698 count: 1
value: 699 count: 1
```

When I create another job with the same name as the cancelled, it reprocess all the messages, so no checkpoint was shared:



```
value: 692 count: 2
value: 693 count: 2
value: 694 count: 2
value: 695 count: 2
value: 696 count: 2
value: 697 count: 2
value: 698 count: 2
value: 699 count: 2
```