

In this tutorial, we'll focus on the “**Extract**”, “**Transform**”, and “**Load**” steps in the ETL process using Mage.ai and Python for streaming data from Kafka. Don't worry if you're not familiar with them—this guide is designed to be easy to follow! 😊

Before we dive into Mage.ai, let's discuss about the data that we will be dealing with. The data has a similar format as the Singapore dataset used in the batch data pipeline guide – a healthcare data for Malaysia that provides the total number of cases for various infectious diseases every week from years 2012 to 2022. A snippet of the data can be seen here:

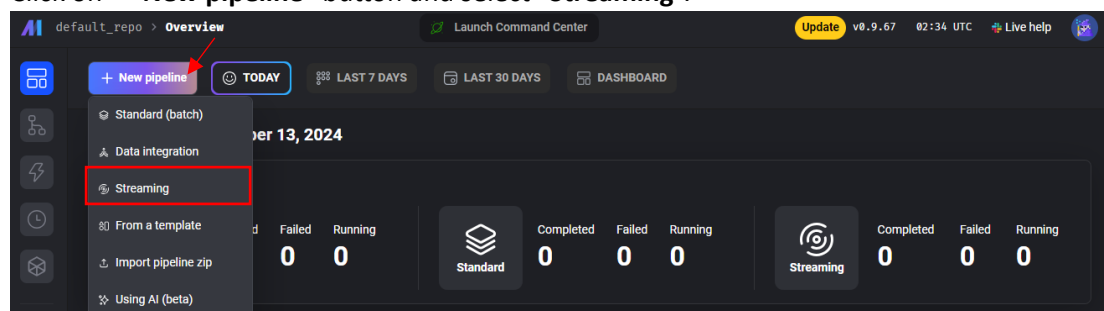
	A	B	C
1	epi_week	disease	no._of_cases
2	2012-W01	Acute Viral hepatitis B	0
3	2012-W01	Acute Viral hepatitis C	0
4	2012-W01	Avian Influenza	0
5	2012-W01	Campylobacterenterosis	6
6	2012-W01	Chikungunya Fever	0
7	2012-W01	Cholera	0
8	2012-W01	Dengue Fever	74
9	2012-W01	Dengue Haemorrhagic Fever	0
10	2012-W01	Diphtheria	0
11	2012-W01	Encephalitis	0
12	2012-W01	Haemophilus influenzae type b	0
13	2012-W01	Hand, Foot Mouth Disease	326

In this course, we will aim to find out the top infectious diseases in Malaysia while keeping the year column. Thus, we need to:

- Get the year of each column [which means we will have many rows with the same year and disease]
- Get the total number of cases based on the same year and disease [which will be stored in the existing column]
- Drop duplicate rows
- Add country column to indicate this is a column for Malaysia dataset.

With this information, let's get started on Mage.ai!

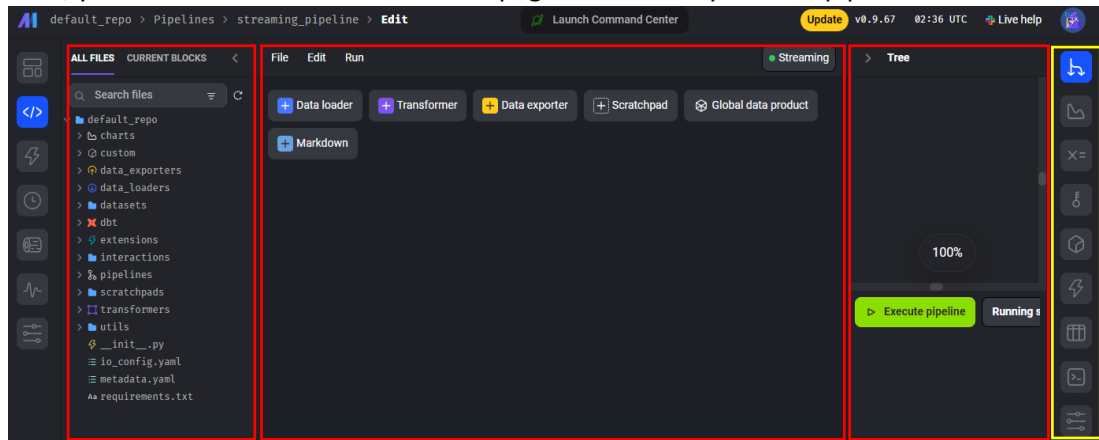
- Click on “+ New pipeline” button and select “**Streaming**”.



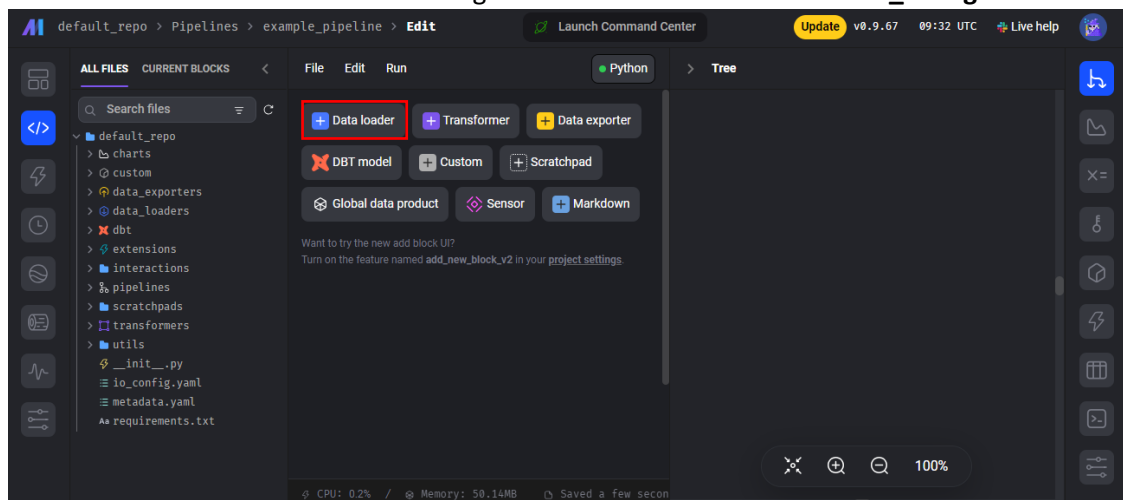
- Mage.ai offers different types of pipelines (TODO: Add usages):
  - Standard (batch),
  - Data Integration,
  - Streaming,
  - From a template,
  - Import pipeline zip, and
  - Using AI.
- You may change/add the pipeline's

For more information regarding Mage.ai's documentation, please refer to <https://docs.mage.ai/>.

- i. Name,
  - ii. Description, and/or
  - iii. Tags to this new pipeline.
2. Next, you will be redirected to the “Edit” page for the newly created pipeline.



- a. Left pane (Directory pane): Directory for “/home/src/default\_repo”.
  - b. Middle pane: Allows you to choose any to add a block to your pipeline and serves as a code editor.
  - c. Right pane: To view the items relevant to the icons on the extreme right that is boxed up in yellow.
3. Firstly, we will load the data from the file to Mage.ai:  
Select “+ Data loader” > “Kafka”. Change the name of this block to “kafka\_config”.



- a. A code editor will be added to the middle pane.
  - b. In the code editor, note that you’re viewing the code for *Data Loader* that is named “kafka\_config”.
  - c. The following block will be created in the middle pane.

```
YAML DATA LOADER kafka_config ← Edit parents
1 connector_type: kafka
2 bootstrap_server: "localhost:9092"
3 topic: topic_name
4 consumer_group: unique_consumer_group
5 include_metadata: false
6 api_version: 0.10.2
7
8 # Uncomment the config below to use SSL config
9 # security_protocol: "SSL"
10 # ssl_config:
11 #   cafile: "CARoot.pem"
12 #   certfile: "certificate.pem"
13 #   keyfile: "key.pem"
14 #   password: password
15 #   check_hostname: true
16
17 # Uncomment the config below to use SASL_SSL config
18 # security_protocol: "SASL_SSL"
19 # sasl_config:
20 #   mechanism: "PLAIN"
21 #   username: username
22 #   password: password
23
24 # Uncomment the config below to use protobuf schema to deserialize message
25 # serde_config:
26 #   serialization_method: PROTOBUF
27 #   schema_classpath: "path.to.schema.SchemaClass"
28
```

d. Replace lines 2 and 3 with the following:

```
bootstrap_server: "kafka:9093"
topic: topic
```

i. This will configure this pipeline to stream messages from hostname “kafka” running on port “9093” that has messages published on topicname “topic”.

4. Secondly, we will **transform** the data.

a. Get the “year” from “epi\_week”.

i. Scroll to the bottom of the middle pane and add a *Transformer*:

Select “Transformer” block > “Python” > “Generic (no template)”.

ii. The function “transform” will be executed when this block runs. The first parameter that this function takes in (“messages”) is the data being streamed in from Kafka.

iii. To extract “year” from “epi\_week”, replace the code in “transform” function with the following. We will convert “messages” to Pandas’ DataFrame type to transform the data in an orderly manner.

```
df = pd.DataFrame(messages)

# Extracts the first 4 characters of each value and store it under the “year” col
df['year'] = df['epi_week'].str.slice(0, 4)
# Removes “epi_week” column from the dataframe
df.drop(columns=['epi_week'], inplace=True)

# Returns this updated dataframe to pass it to the next block
return df.to_dict('records')
```

b. Get the new number of cases based on year and disease:

- i. Add another *Transformer* block by going to the end of the middle pane and select “**Transformer**” block > “**Python**” > “**Generic (no template)**”.
- ii. Similar to the “**transform**” function mentioned in Section 4.a.ii (TODO: Add link), the first parameter taken in by the function “**transform**” in this newly created *Transformer* is the list of dictionaries returned from the previous *Transformer*.
- iii. To get the new number of cases, group the data based on its disease and year values for that row and sum the no.\_of\_cases.

```
df = pd.DataFrame(messages)

# Groups data by disease and year to get the summed no._of_cases value
df = df.groupby(['disease', 'year']).agg({'no._of_cases': 'sum'}).reset_index()
return df.to_dict('records')
```

- c. Drop duplicate rows
  - i. Based on Section 4.b.iii (TODO: Add link), there may have many duplicated rows in the dataframe (i.e. same year, disease, and number of cases).
  - ii. Add a new *Transformer* from the bottom of the middle pane: select “**Transformer**” block > “**Python**” > “**Generic (no template)**” and replace the code in “**transform**” function with the following:

```
df = pd.DataFrame(messages)

df.drop_duplicates(inplace=True)
return df.to_dict('records')
```

- d. Add “**country**” column
  - i. Now that our data is ready, we want to add a new column called “**country**” and the values in this column will all be “**malaysia**”. This allows us to add other countries’ similar health data in the future.
  - ii. To do so, add a generic *Transformer* block like in Section 4.a.i (TODO: Add link) and replaced the function “**transform**” code with the following:

```
df = pd.DataFrame(messages)
df['country'] = 'malaysia'
return df.to_dict('records')
```

5. Lastly, let’s **load** this data into your database.
  - a. Let’s add a *Data Exporter* block at the end of the middle pane: Select “**Data Exporter**” > “**PostgreSQL**”.

- b. Update the code in this newly added block to the following:

```
# [country, year, disease] is not unique
connector_type: postgres
database: db
host: db
password: password
port: 5432
schema: public
username: postgres
table: healthcare_my
```

6. Now that our pipeline is ready, we can execute the pipeline on the right pane of this page:

The screenshot displays the Mage.ai interface with a pipeline configuration on the left and its execution status on the right. The pipeline configuration includes a transformer block for adding a country and a data exporter block for exporting to a PostgreSQL database. The execution status shows the pipeline running successfully, with messages received from a Kafka source and data exported to the database.

```
def = pd.DataFrame(messages)
df['country'] = 'malaysia'
return df.to_dict('records')
```

```
1 # [country, year, disease] is not unique
2 connector_type: postgres
3 database: db
4 host: db
5 password: password
6 port: 5432
7 schema: public
8 username: postgres
9 table: healthcare_my
```

Execute pipeline Running status

```
[streaming_pipeline] DONE
[streaming_pipeline] [KafkaSource] Received 1 messages from topic-'topic' partition=0 at time=172
6393597.687194
[streaming_pipeline] [KafkaSource] Receive message 0:1830: key=None, value=b'{"epi_week": "2019-W
54", "disease": "Legionellosis", "no._of_cases": 266}' time=1726393597.6872682
[streaming_pipeline] [PostgresSink] Batch ingest 1 records, time=1726393597.6963688. Sample: [{"di
sease": "Legionellosis", "year": "2019", "no._of_cases": 266, "country": "malaysia"}]
[streaming_pipeline]
[streaming_pipeline] | Exporting data to 'public.healthcare_my' ...
[streaming_pipeline] DONE
[streaming_pipeline] [KafkaSource] Received 1 messages from topic-'topic' partition=0 at time=172
6393604.8961719
[streaming_pipeline] [KafkaSource] Receive message 0:1831: key=None, value=b'{"epi_week": "2019-W
```

- a. The output from the pipeline execution will be displayed in the bigger red box. You may stop your pipeline execution at any time.