

# Project

In this project we must transform the data files as we must get the data in the required format. For this I will use Azure Data factory as a transformation service by using Data Flows for serving this purpose.

## First, we have to transform Cases\_deaths.csv file

This file contains the following columns:

country	country_code	continent	population	indicator	daily_count	date	rate_14_day	source
---------	--------------	-----------	------------	-----------	-------------	------	-------------	--------

## Know The Data

**Country column:** We want this column as it is.

**Country\_Code:** This country code is available in 3 letter format but we want to have both three letter and two letter format country codes in our transformed file so that in future we can join this file with any other file containing any one of these two formats.

To do this we have another file containing country and their codes in both of these formats. We will use that file to achieve our purpose.

**Continent:** This column contains data for five continents: Africa, Europe, Asia, America and Oceania. We will filter our data for Europe only and will not include continent as a column in our transformed file.

**Population:** This column will remain as it is.

**Indicator and daily\_count:** The indicator column contains two unique values: confirmed cases and deaths. Whereas the daily count column contains the number of confirmed cases and deaths on a particular date.

This means that for every date there will be two replica of rows for the two unique indicators having all the content same except the indicator and daily\_count.

To eradicate this, we will pivot the indicator and daily\_count so that we will have confirmed cases count and deaths count as separate column for each date.

This will now make date and other fields as unique values.

**Date:** This data is from 2 Jan, 2020 to 25 Oct, 2020

**Rate\_14\_day:** This indicates the rate of cases for the last 14 days. But we do not require this.

**Source:** This indicates the source for the data.

## Additional Resource

**For Lookup:** As discussed above we have to include the country code both in 3 letter and 2 letter formats. So, for that we need one lookup file.

Name: country\_lookup

**For Testing:** In order to test whether our filter continent = Europe is successful or not. We will first make a separate file containing the data for one or two countries that are not in Europe. After Transformation we will test the data using this file.

## Process

**Note:** During the development of data flow, if we want to preview the data after every step. That means any transformations that we have applied on the source so far will have to be executed and then ADF should return the data.

For that ADF will convert this data flow into apache spark code and will run that on a spark cluster.

For this we need to enable the DEBUG mode first. Enabling the debug mode will create a spark cluster with the default configuration. This will be a general compute cluster with 4 cores and some memory and will be there active for 60 minutes.

If you want to change the configuration, you have to create a new Azure integration runtime with your specified spark cluster configuration which you can give during creation.

- Generate a dataflow named df\_Cases\_And\_Deaths. We need atleast one source and one sink but we can have any number of source and sinks.
- Generate a source and give that name as CasesAndDeathsSource. Now for dataset we can either use the datasets defined in ADF or we can create a dataset local to this dataflow which is called inline datasets. But this supports only few file formats like excel, delta etc. As our file is a delimited file so will have the dataset from ADF.
  - We will specify the dataset which is CasesAndDeathsSource in our case which points to a Data Lake storage.
- Then we will use a filter activity to filter out our data on the basis of

The screenshot shows the 'Source settings' tab in the Azure Data Factory portal. The 'Output stream name' is 'CasesAndDeathsSource'. The 'Description' is 'Import data from CasesAndDeathsSource'. The 'Source type' is set to 'Dataset' (with 'Inline' also available). The 'Dataset' dropdown is set to 'CasesAndDeathsSource'. Below this are links for 'Test connection', 'Open', and 'New'. The 'Options' section includes 'Allow schema drift' (checked), 'Infer drifted column types' (unchecked), and 'Validate schema' (unchecked). The 'Skip line count' field is empty. The 'Sampling' option is set to 'Disable'.

Source settings   Source options   Projection   Optimize   Inspect   Data preview

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Source type \* Dataset Inline

Dataset \*

[Test connection](#) [Open](#) [New](#)

Options ☒ Allow schema drift [?](#)  
☐ Infer drifted column types [?](#)  
☐ Validate schema [?](#)

Skip line count

Sampling \* [?](#) ☐ Enable ☒ Disable

```
continent == 'Europe' && not(isNull(Country_code))
```

[Filter settings](#) [Optimize](#) [Inspect](#) [Data preview](#)

Output stream name \*

filterOnContinent

[Learn more](#)

Description

Filtering rows using expressions on columns 'continent, country\_code'

Reset

Incoming stream \*

CasesAndDeathsSource

Filter on \*

continent == 'Europe' &&  
not(isNull(country\_code))

- Then we need to select only required columns. The select activity is basically used to modify the fields name, mapping and to select specific columns only. Here in this step we will remove the rate\_14\_day and continent columns.
  - Also, as we can add and modify the mappings with the two available options. So we will keep some fixed mappings as it is except for date and will add one rule based mapping for date to rename this to reported\_date.
  - Here we need to pass one matching condition and expression
  - The matching condition will check for all the columns and those columns with which this condition applies it will apply the output expression to that column.
  - The matching condition if we provide as true() it will take all the columns.
  - Here in our case the matching condition will be name=='date'. This will check for all the columns with name as date and apply output expression to those.
  - In the output expression we will give:  
'reported\_' + \$\$
  - This \$\$ means the columns returned by matching condition.

Select settings
Optimize
Inspect
Data preview
Previous
Next

Output stream name \*
[Learn more](#)

Description

Renaming filterOnContinent to selectColumns with columns 'country, country\_code, population, indicator,

Reset

Incoming stream \*

Options
☐ Skip duplicate input columns ⓘ
☐ Skip duplicate output columns ⓘ

Input columns \*
☐ Auto mapping ⓘ
Reset
+ Add mapping
Delete
7 mappings: 2 column(s) from the inputs left unmapped ⓘ

<input type="checkbox"/>	filterOnContinent's column		Name as	
<input type="checkbox"/>	abc country	→	country	+
<input type="checkbox"/>	abc country_code	→	country_code	+
<input type="checkbox"/>	abc population	→	population	+
<input type="checkbox"/>	abc indicator	→	indicator	+
<input type="checkbox"/>	abc daily_count	→	daily_count	+
<input type="checkbox"/>	abc source	→	source	+
<input type="checkbox"/>	name == 'date'	→	'reported_' + \$\$	

- After this as we need to make separate columns for the indicator column unique values and want sum of daily count for them in order to eradicate the duplicity in rows, we will use pivot activity.
  - This pivot activity will take three values:
    - Group by columns
    - Pivot key
    - Pivoted columns
  - In the group by columns we need to specify all column except indicator and daily\_count as we need all of them.
  - In pivot key we need to give indicator and under pivot key we need to specify all the unique values in indicator.
  - In pivoted columns we need to specify the aggregation to be imposed on daily\_count values. Here in our case we need sum. But as the column is in string we will first convert the column to be in ineteger and then apply the sum on that. The expression will look like:

○ `sum(toInteger(daily_count))`

Pivot settings   Optimize   Inspect   Data preview

Output stream name \*  ? Help [Learn more](#)

Description 

Pivots row values into columns, groups columns and aggregates data

 Reset

Incoming stream \*

1. Group by   2. Pivot key   3. Pivoted columns

Columns	Name as		
<input type="text" value="abc country"/>	country		
<input type="text" value="abc country_code"/>	country_code		
<input type="text" value="abc population"/>	population		
<input type="text" value="abc source"/>	source		
<input type="text" value="abc reported_date"/>	reported_date		

Pivot settings   Optimize   Inspect   Data preview

Output stream name \*  ? Help [Learn more](#)

Description 

Pivots row values into columns, groups columns and aggregates data

 Reset

Incoming stream \*

1. Group by   2. Pivot key   3. Pivoted columns

Pivot key \*

Value

<input type="text" value="confirmed cases"/>		
<input type="text" value="deaths"/>		

☐ Null value

Pivot settings   Optimize   Inspect   Data preview

---

Output stream name \*  ? Help [Learn more](#)

Description 

Pivots row values into columns, groups columns and aggregates data

[Reset](#)

Incoming stream \*

1. Group by   2. Pivot key   **3. Pivoted columns**

---

Column name pattern \*

Column arrangement \* 

☐ Normal
☒ Lateral

12L  +

- The next activity we need is to use a lookup activity that will lookup in the Country lookup file and will provide the columns there based on our specified condition.
  - In primary stream we need to specify the previous pivot activity and lookup field we need to specify the lookup source activity.
  - In condition mention the country column from primary and lookup stream that will be evaluated by ==.

Lookup settings   Optimize   Inspect   Data preview

---

Output stream name \*  [Learn more](#)

Description 

Lookup on 'pivotIndicatorDailycount' from 'LookupSource'

[Reset](#)

Primary stream \*

Lookup stream \*

Match multiple rows ☐ ⓘ

Match on \*

Lookup conditions \*
 

Left: pivotIndicatorDailycount's column

Right: LookupSource's column

+

- Then we select the necessary columns, we will take country\_2\_digit\_code and country\_3\_digit\_code and drop country\_code and also rename the pivot key column to cases\_count and deaths\_count.

Select settings   Optimize   Inspect   Data preview

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Incoming stream \*

Options ☒ Skip duplicate input columns [?](#) ☒ Skip duplicate output columns [?](#)

Input columns \* ☐ Auto mapping [?](#) [Reset](#) [+ Add mapping](#) [Delete](#)

<input type="checkbox"/>	lookupForCountryCode's column		Name as		
<input type="checkbox"/>	abc pivotIndicatorDailycount@country	→	country	+	
<input type="checkbox"/>	abc country_code_2_digit	→	country_code_2_digit	+	
<input type="checkbox"/>	abc country_code_3_digit	→	country_code_3_digit	+	
<input type="checkbox"/>	abc pivotIndicatorDailycount@population	→	population	+	
<input type="checkbox"/>	12L confirmed cases_count	→	cases_count	+	
<input type="checkbox"/>	12L deaths_count	→	deaths_count	+	
<input type="checkbox"/>	abc reported_date	→	reported_date	+	

- At last we will put this data into a sink where in settings we will tick the clear the folder so that every time first the folder will be clear and then we put the data. And in file name option we will specify output to a single file

Sink   Settings   Errors   Mapping   Optimize   Inspect   Data preview

**i** This sink currently has Single partition set in Optimize. This will make your data flow execution longer. The recommended setting is Use current partitioning.

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Incoming stream \*

Sink type \* ☒ Dataset ☐ Inline ☐ Cache

Dataset \*  [Test connection](#) [Open](#) [+ New](#)

Skip line count

Options ☒ Allow schema drift [?](#) ☐ Validate schema [?](#)



and also specify the output name with extension(it is necessary as if we do not specify this then we will get error in data lake while preview).

SinkSettingsErrorsMappingOptimizeInspectData preview

This sink currently has Single partition set in Optimize. This will make your data flow execution longer. The recommended setting is Use current partitioning.

Clear the folder

☒

File name option \*

Output to single file

Output to single file \* ⓘ

CasesAndDeaths.csv

Quote All ⓘ

☐

Headers ⓘ

Enter expression...

ANY

Umask ⓘ

Owner☐ R☐ W☐ X

Group☐ R☒ W☐ X

Others☐ R☒ W☐ X

Octal

022

Pre/post commands ⓘ

File pre command

mkdir, mv, cp, rm

+🗑

File post command

mkdir, mv, cp, rm

+🗑

CasesAndDeath...

Import data from CasesAndDeathsSource

+

filterOnContinent

Filtering rows using expressions on columns 'continent'

+

selectColumns

Renaming filterOnContinent to selectColumns with

+

pivotIndicatorDail...

Pivots row values into columns, groups columns and

+

lookupForCountry...

Lookup on 'pivotIndicatorDailycount' from 'LookupSource'

+

selectNecessaryC...

Renaming lookupForCountryCode to

+

CasesAndDeath...

Export data to ds\_processed\_cases\_deaths

LookupSource

Import data from DL\_LookupFile

+

Data flow

Data flow Cases...

✓✗➔

Activity runs

Pipeline run ID 655bf0dc-989a-43ca-84b7-06460efff8ac

All status ▾

Showing 1 - 1 of 1 items

Activity name	Status	Activity type	Run start	Duration
Data flow Cases and Deaths	✔ Succeeded	Data flow	7/19/2023, 2:31:52 PM	1m 50s

## Second, we have to transform hospital\_admissions.csv file

This file has the following columns.

country	indicator	date	year_week	value	source	url
---------	-----------	------	-----------	-------	--------	-----

The granularity of this file is on a daily and weekly basis.

## Know The Data

**Country:** It contains country names for which we have admission count. But with country we also need to have 2 letter country code and 3 letter country code. For that

**Indicator:** This column contains four flags 2 for daily and 2 for weekly. Having following values:

- Weekly new ICU admissions per 100k
- Weekly new hospital admissions per 100k
- Daily ICU occupancy
- Daily hospital occupancy

**Date:** It contains date from 2 February, 2020 to 25 October, 2020.

**Year\_week:** It contains year and week number concatenated by hyphen (-). For example: 2020-W41

**Value:** This contains number of admissions done for a day.

**Source:** This contains information of the source where we have gathered this information.

**URL:** It contains URL to the source.

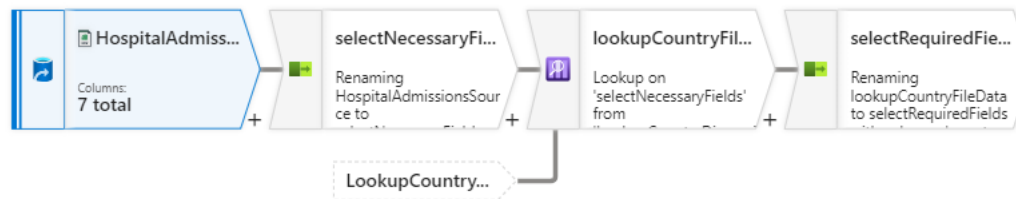
# Resources Needed

We need a lookup file to gather the 2 letter and 3 letter country code. For this also we will use same file country\_lookup.csv

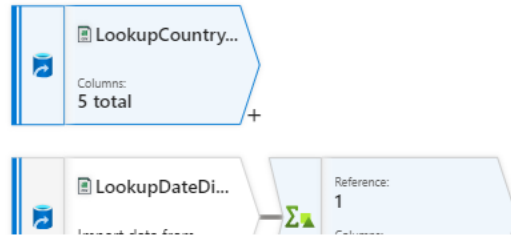
We need a lookup file to get the start and end date of a week based on year\_week. For this we will use Dim\_date.csv file.

## Process

- Generate a pipeline named as df\_hospital\_admissions
- Create a three source activities pointing to hospital\_admissions.csv, lookup\_country.csv and dim\_date.csv files. These files are there in data lake.



Source settings	Source options	Projection	Optimize	Inspect	Data preview
Output stream name *	HospitalAdmissionsSource		<a href="#">Learn more</a>		
Description	Import data from ds_Hospital_Admissions		<a href="#">Reset</a>		
Source type *	<div><div>Dataset</div><div>Inline</div></div>				
Dataset *	ds_Hospital_Admissions		<a href="#">Test connection</a> <a href="#">Open</a> <a href="#">New</a>		
Options	<div><input checked="" type="checkbox"/> Allow schema drift <a href="#">?</a></div> <div><input type="checkbox"/> Infer drifted column types <a href="#">?</a></div> <div><input type="checkbox"/> Validate schema <a href="#">?</a></div>				
Skip line count	<input type="text"/>				
Sampling * <a href="#">?</a>	<div><input type="radio"/> Enable <input checked="" type="radio"/> Disable</div>				



Source settings

Source options

Projection

Optimize

Inspect

Data preview

Output stream name \*

LookupCountryDimension

[Learn more](#)

Description

Import data from DL\_LookupFile

Reset

Source type \*

Dataset

Inline

Dataset \*

DL\_LookupFile

Test connection

Open

New

Options

☒ Allow schema drift

☐ Infer drifted column types

☐ Validate schema

Skip line count

LookupDateDi...

Columns: 12 total

+

aggregateToAddS...

Aggregating data by 'ecdc\_year\_week' producing columns

+

Add Source

▼

Source settings

Source options

Projection

Optimize

Inspect

Data preview

Output stream name \*

LookupDateDimension

[Learn more](#)

Description

Import data from LookupDateFile

Reset

Source type \*

Dataset

Inline

Dataset \*

LookupDateFile

Test connection

Open

New

Options

☒ Allow schema drift

☐ Infer drifted column types

☐ Validate schema

Skip line count

Sampling \*

Enable

Disable

- Then first we will create a select activity on hospital\_admissions.csv source activity and eradicate the URL column and select all other.

**Select settings** | Optimize | Inspect | Data preview

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Incoming stream \*

Options

- ☒ Skip duplicate input columns ⓘ
- ☒ Skip duplicate output columns ⓘ

Input columns \*

☐ Auto mapping ⓘ [Reset](#) [+ Add mapping](#) [Delete](#)

<input type="checkbox"/>	HospitalAdmissionsSource's column		Name as		
<input type="checkbox"/>	abc country	→	country	+	🗑️
<input type="checkbox"/>	abc indicator	→	indicator	+	🗑️
<input type="checkbox"/>	abc date	→	reported_date	+	🗑️
<input type="checkbox"/>	abc year_week	→	reported_year_week	+	🗑️
<input type="checkbox"/>	abc value	→	value	+	🗑️
<input type="checkbox"/>	abc source	→	source	+	🗑️

- Then we need 2 letter and 3 letter country code which we will get from lookup\_country.csv file by looking up at country column. So we will then create a lookup activity to do the same.

**Lookup settings** | Optimize | Inspect | Data preview

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Primary stream \*

Lookup stream \*

Match multiple rows ☐ ⓘ

Match on \*

Lookup conditions \*

Left: selectNecessaryFields's column      Right: LookupCountryDimension's column

abc country	==	abc country	<a href="#">+</a> <a href="#">🗑️</a>
-------------	----	-------------	--------------------------------------

- Now again as after lookup we get some extra columns we will use select activity to select the required columns. We will keep 2 letter country code, 3 letter country code and population columns which we get after lookup with the existing columns and drop everything else.

**selectRequiredFields**

Output stream name:  [Learn more](#)

Description:  [Reset](#)

Incoming stream:

Options:

- ☒ Skip duplicate input columns
- ☒ Skip duplicate output columns

Input columns:

☐ Auto mapping [Reset](#) [Add mapping](#) [Delete](#)

<input type="checkbox"/>	lookupCountryFileData's column	Name as	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc selectNecessaryFields@country	country	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc country_code_2_digit	country_code_2_digit	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc country_code_3_digit	country_code_3_digit	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc population	population	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc indicator	indicator	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc reported_date	reported_date	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc reported_year_week	reported_year_week	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc value	value	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	abc source	source	<input type="checkbox"/>	<input type="checkbox"/>

- Now as we want that the grain of the file should be either daily or weekly not both we will split the file and will daily related flag rows in Daily file and weekly flag related rows in weekly file.

**Conditional split**

Output stream name:  [Learn more](#)

Description:  [Reset](#)

Incoming stream:

Split on: ☒ First matching condition ☐ All matching conditions

Split condition:

Stream names	Condition	<input type="checkbox"/>	<input type="checkbox"/>
Daily	indicator == 'Daily hospital occupancy' && indicator == 'Daily ICU ...'	<input type="checkbox"/>	<input type="checkbox"/>
Weekly	Rows that do not meet any condition will use this output stream	<input type="checkbox"/>	<input type="checkbox"/>

- On daily file then we will pivot the indicator and values columns.

Pivot settings   Optimize   Inspect   Data preview

Output stream name \*  ? Help [Learn more](#)

Description  Reset

Incoming stream \*

1. Group by   **2. Pivot key**   3. Pivoted columns

Pivot key \*

Value

+

+

☐ Null value

Pivot settings   Optimize   Inspect   Data preview

Output stream name \*  ? Help [Learn more](#)

Description  Reset

Incoming stream \*

1. Group by   2. Pivot key   **3. Pivoted columns**

Column name pattern \*

Prefix  -  Suffix

Column arrangement \* ☐ Normal ☒ Lateral

+

- On weekly file as we need to have start and end date. But on dim date file there is no column that is of same format as year\_week.
  - So, we will first create an aggregate activity to create year\_week like column with a column for start date and for end date.
  - So we will specify a new column name in group by ecdc\_year\_week with following expression: year+'-W'+lpad(week\_of\_year,2,'0')

- Then in aggregation we will specify two new column week\_start\_date with expression min(date) and week\_end\_date with expression of max(date).




Diagram showing data flow from **LookupDateDimension** (Import data from LookupDateFile) to **aggregateToAddStartEndDate** (Columns: 3 total).

---

**Aggregate settings** | Optimize | Inspect | Data preview

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Incoming stream \*

**Group by** [Aggregates](#)

Columns	Name as
<input type="text" value="year+'-W'+lpad(week_of_year,2,'0')"/> <a href="#">abc</a>	<input type="text" value="ecdc_year_week"/> <a href="#">+</a> <a href="#">🗑️</a>

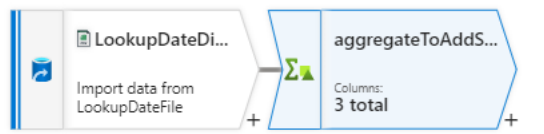


Diagram showing data flow from **LookupDateDimension** (Import data from LookupDateFile) to **aggregateToAddStartEndDate** (Columns: 3 total).

---

**Aggregate settings** | Optimize | Inspect | Data preview

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Incoming stream \*

**Group by** [Aggregates](#)

Grouped by: ecdc\_year\_week

[+](#) Add [📄](#) Clone [🗑️](#) Delete [🔗](#) Open expression builder

<input type="checkbox"/> Column	Expression
<input type="checkbox"/> week_start_date <a href="#">▼</a>	<input type="text" value="min(date)"/> <a href="#">abc</a> <a href="#">+</a> <a href="#">🗑️</a>
<input type="checkbox"/> week_end_date <a href="#">▼</a>	<input type="text" value="max(date)"/> <a href="#">abc</a> <a href="#">+</a> <a href="#">🗑️</a>



- Now we will join this aggregate output with weekly file on `year_week == ecdc_year_week`.

**Join settings** | Optimize | Inspect | Data preview

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Left stream \*

Right stream \*

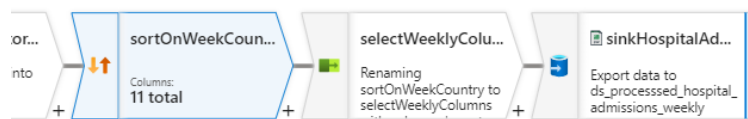
Join type \* Full outer **Inner** Left outer Right outer Custom (cross)

Use fuzzy matching ☐

Join conditions \*

Left: SplitDailyandWeeklyData@Weekly's column	Operator	Right: aggregateToAddStartEndDate's column
reported_year_week	=	ecdc_year_week

- Then on weekly also we perform pivot for indicator and value columns.
- Then we sort the both the data reported date descending and country ascending.
  - Here as we are using spark cluster that do things in distributed fashion and in the end also it will copy part of files in sink. But we want the output in a single file so sorting also should be performed like this.
  - So in optimize we will click on single partition



Sort settings   Optimize   Inspect   Data preview

Output stream name \*  [Learn more](#)

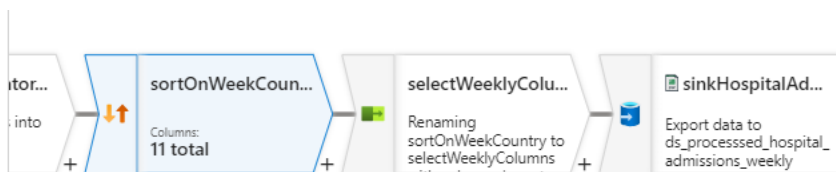
Description  [Reset](#)

Incoming stream \*

Options \* ☐ Case insensitive ☐ Sort only within partition

Sort conditions \*

pivotOnIndicatorWeekly's column	Order	Nulls first
abc reported_year_week	Descending	<input checked="" type="checkbox"/>
abc country	Ascending	<input checked="" type="checkbox"/>



Sort settings   **Optimize**   Inspect   Data preview

Partition option \* ☐ Use current partitioning ☒ Single partition ☐ Set partitioning

- Then we will use two sinks to copy this data as a single file hospital\_admissions\_daily.csv and hospital\_admissions\_weekly.csv in data lake output container.