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. Where as 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.

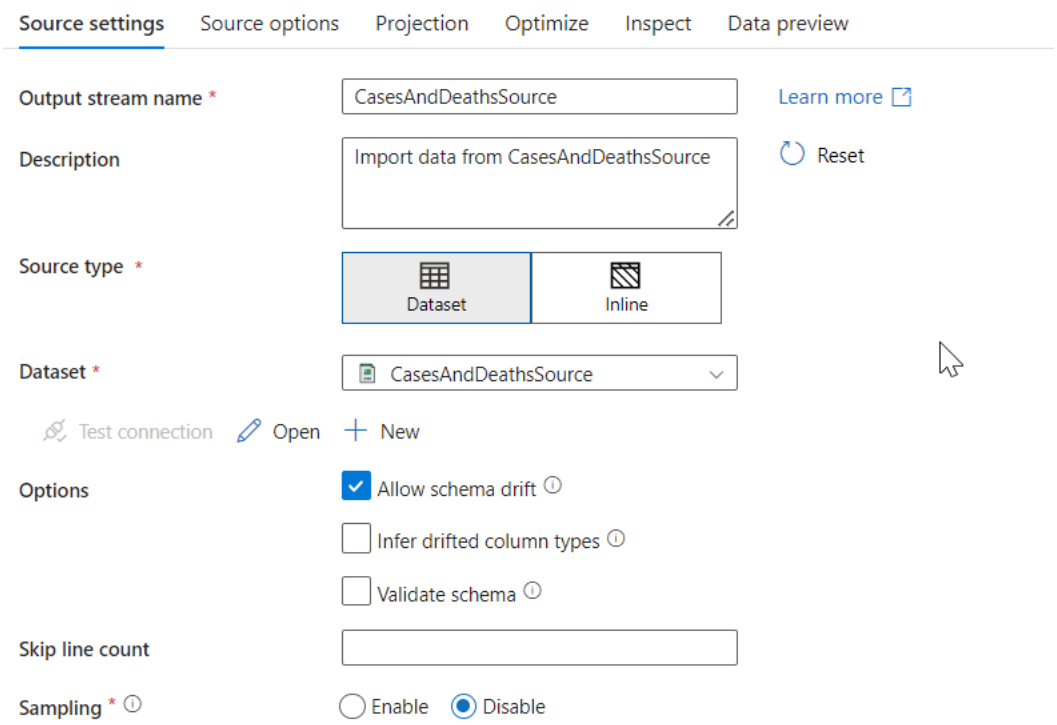
For that ADF will convert this data flow into apache spark code and will run that on a spark cluster.

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.

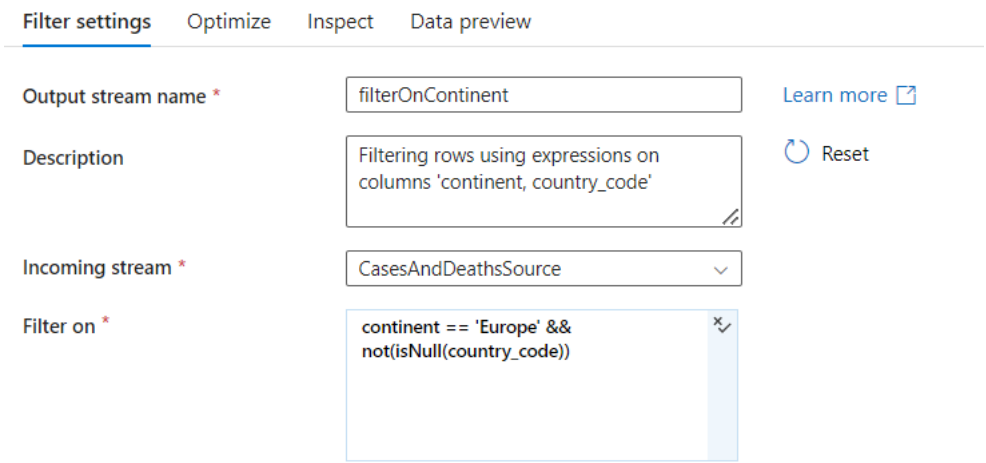
If you want to change the configuration, you have to create a new Azure integration runtime with your specified spark cluster configuration which you can give during creation.

- 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 will have the dataset from ADF.
    - We will specify the dataset which is CasesAndDeathsSource in our case which points to a Data Lake storage.

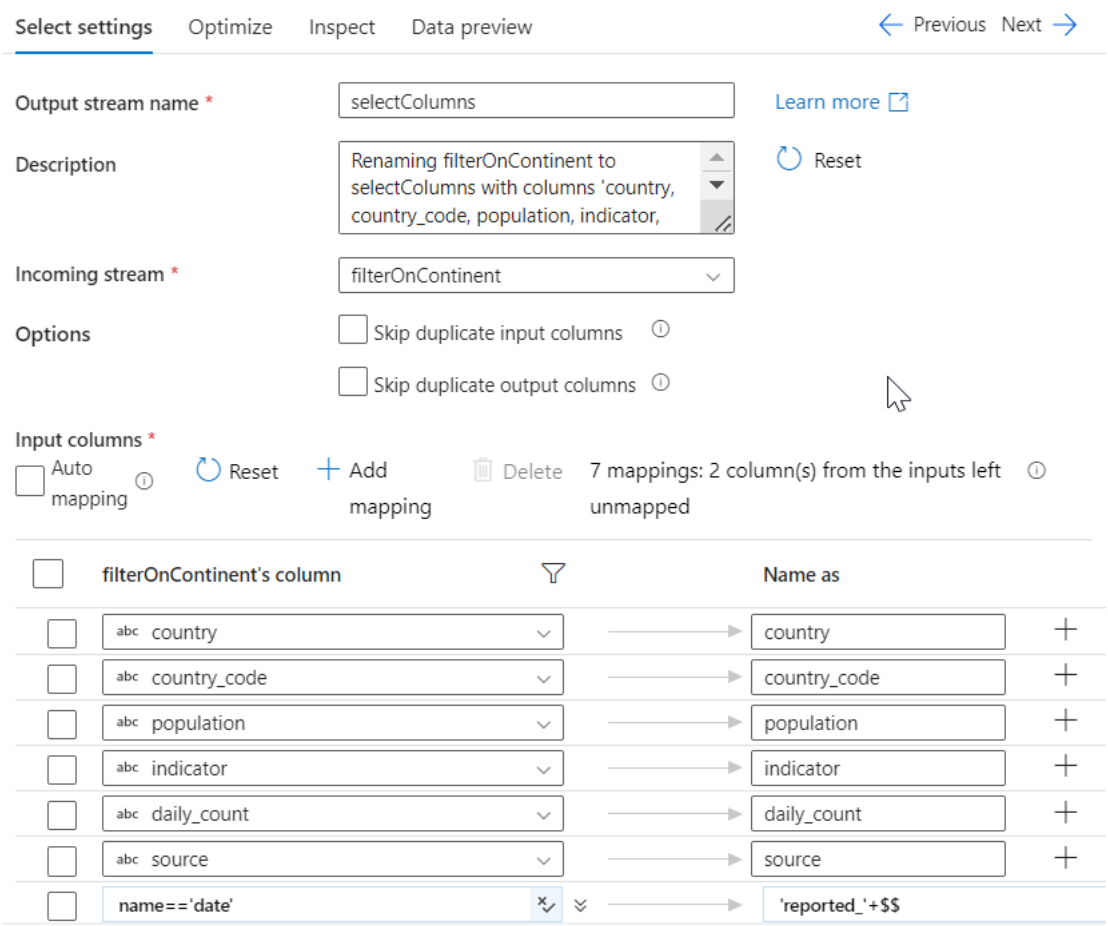
- Then we will use a filter activity to filter out our data on the basis of



continent == ‘Europe’ && not(isNull(Country_code))

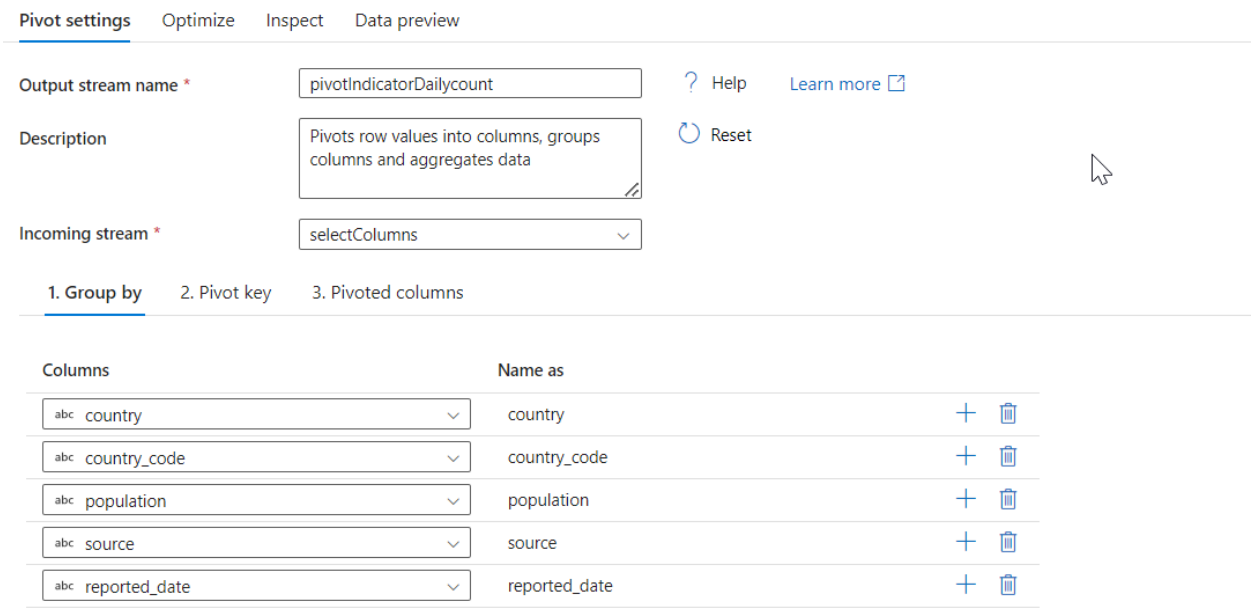


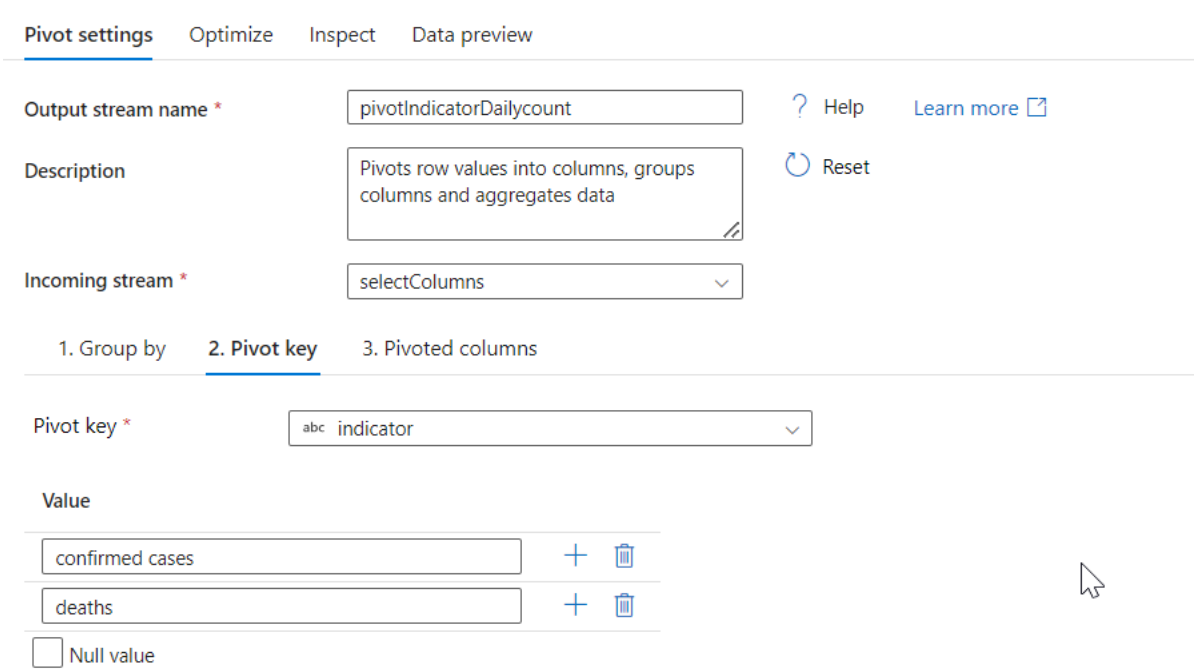
- Then we need to select only required columns. The select activity is basically uses to modify the fields name, mapping and to select specific columns only. Here in this step we will remove the rate_14_day and continent columns.
    - Also, as we can add and modify the mappings with the two available 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.

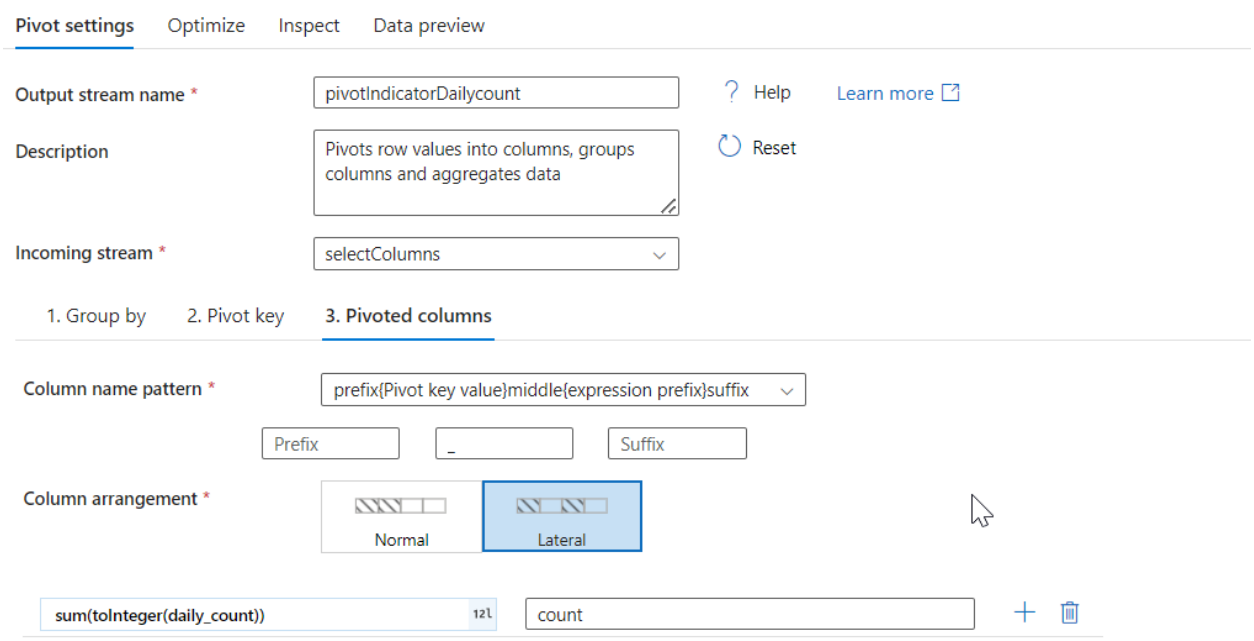


- After this as we need to make separate columns for the indicator column unique values and want sum of daily count for them in order to eradicate the duplicity in rows, we will use pivot activity.
    - This pivot activity will take three values:
        - Group by columns
        - Pivot key
        - 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:

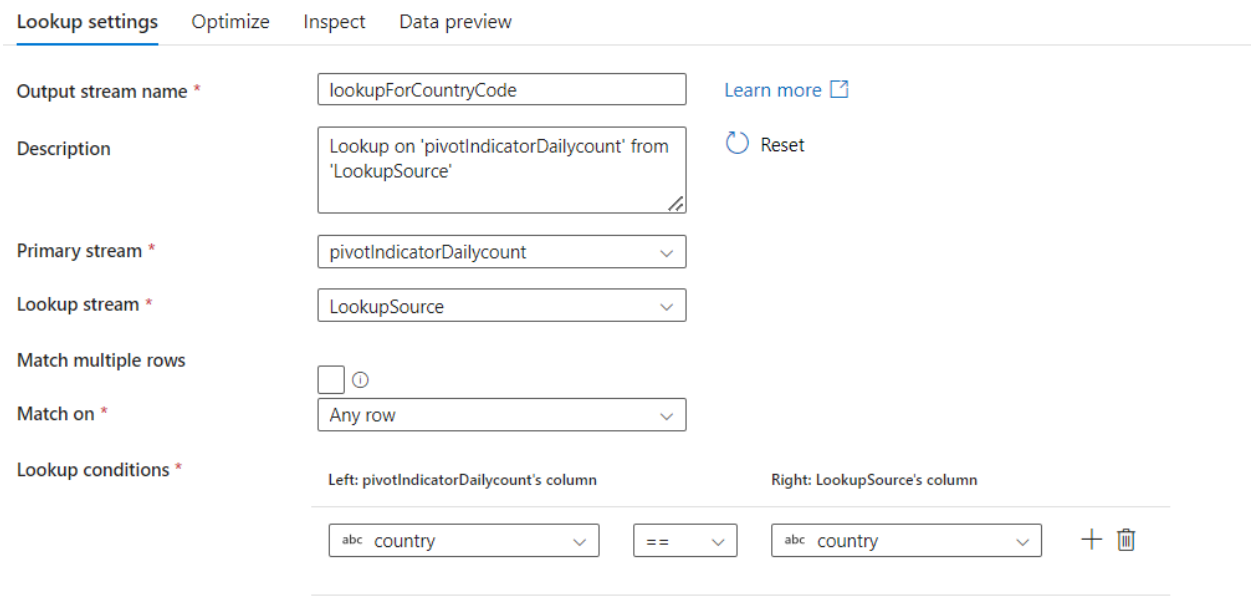
- sum(toInteger(daily_count))



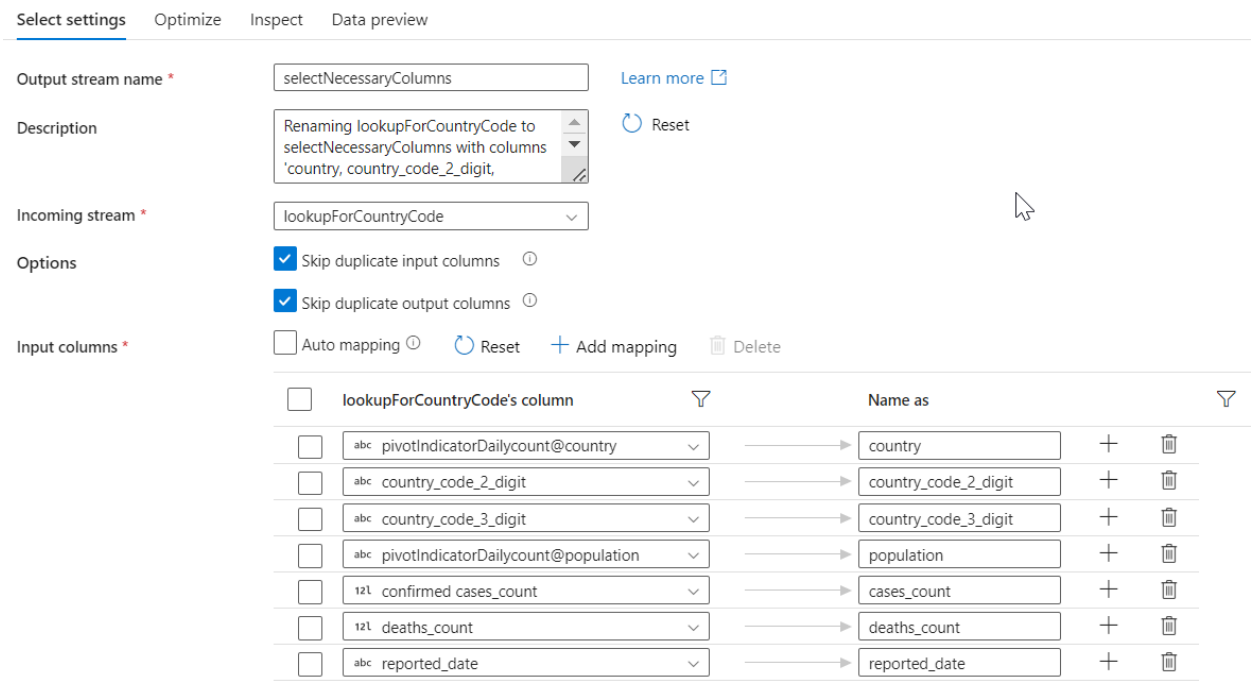




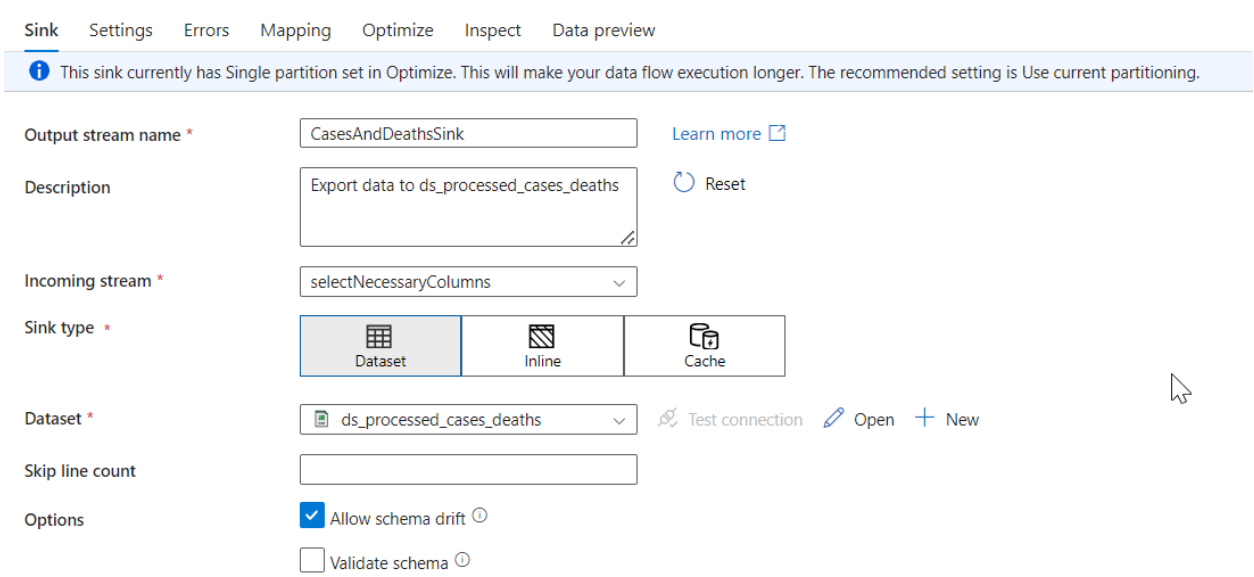
- The next activity we need is to use a lookup activity that will lookup in the Country lookup file and will provide the columns there based on our specified condition.
    - In primary stream we need to specify the previous pivot activity and lookup field we need to specify the lookup source activity.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



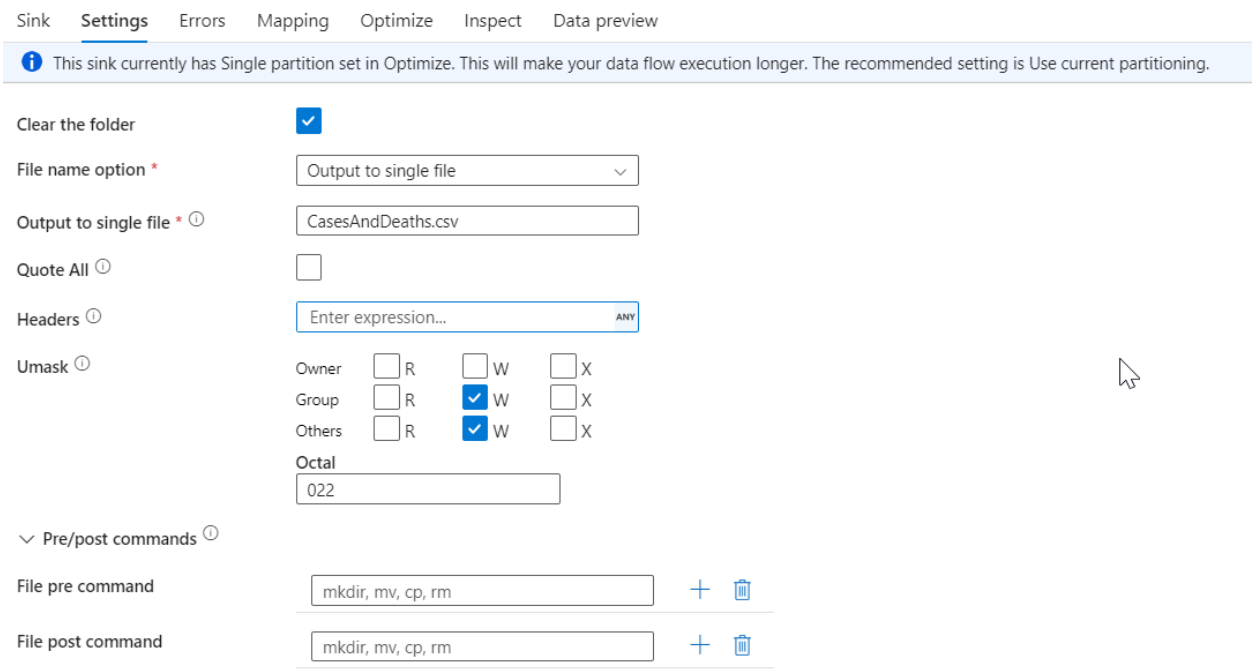
- Then we select the necessary columns, we will take country_2_digit_code and country_3_digit_code and drop country_code and also rename the pivot key column to cases_count and deaths_count.

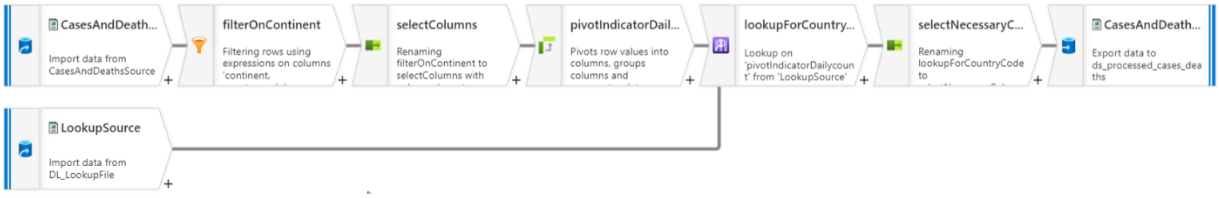


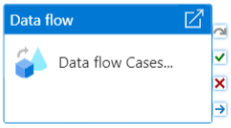
- At last we will put this data into a sink where in settings we will tick the clear the folder so that every time first the folder will be clear and then we put the data. And in file name option we will specify output to a single file

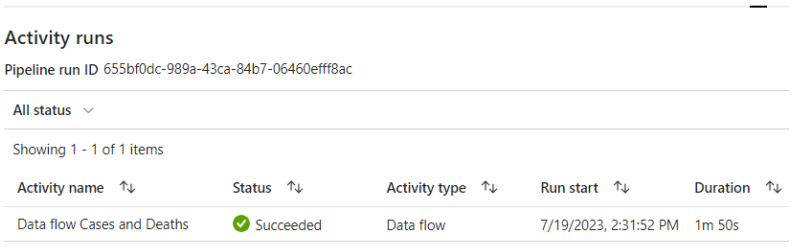


and also specify the output name with extension(it is necessary as if we do not specify this then we will get error in data lake while preview).









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.

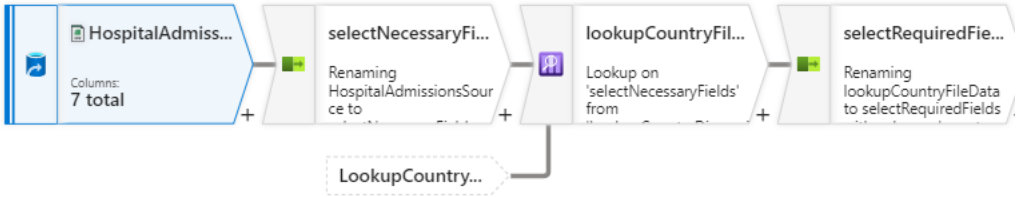
Resources Needed

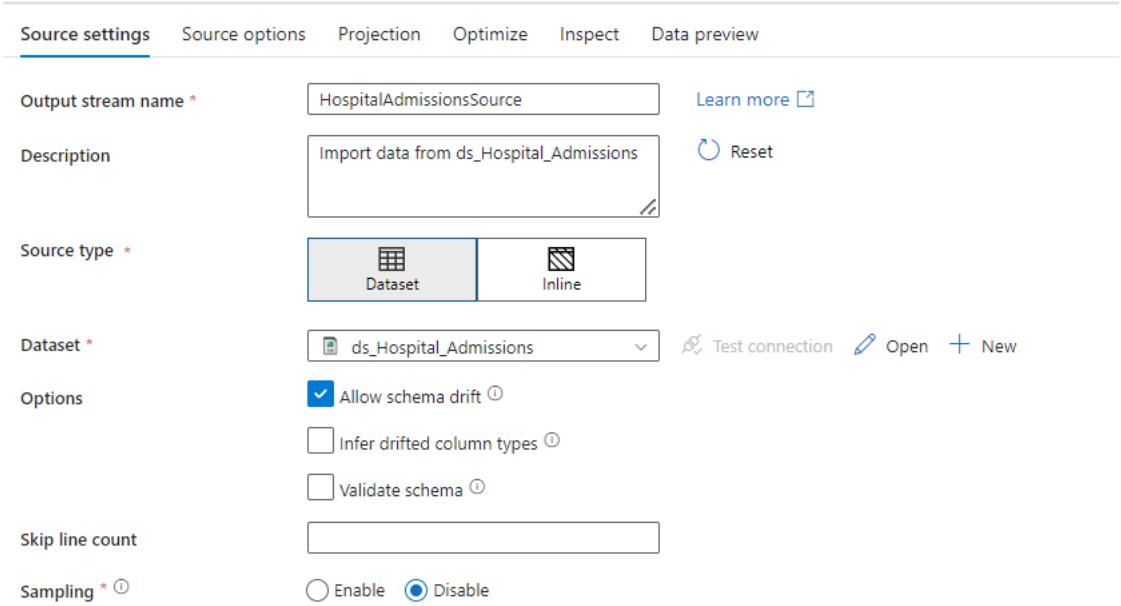
We need a lookup file to gather the 2 letter and 3 letter country code. For this also we will use same file country_lookup.csv

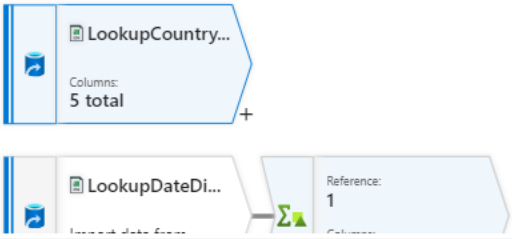
We need a lookup file to get the start and end date of a week based on year_week. For this we will use Dim_date.csv file.

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 * LookupCountryDimension Learn more

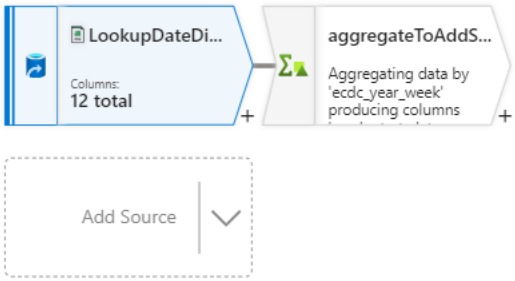
Description Import data from DL_LookupFile Reset

Source type * Dataset | Inline

Dataset * DL_LookupFile Test connection Open New

Options
- [x] Allow schema drift
- [ ] Infer drifted column types
- [ ] Validate schema

Skip line count



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

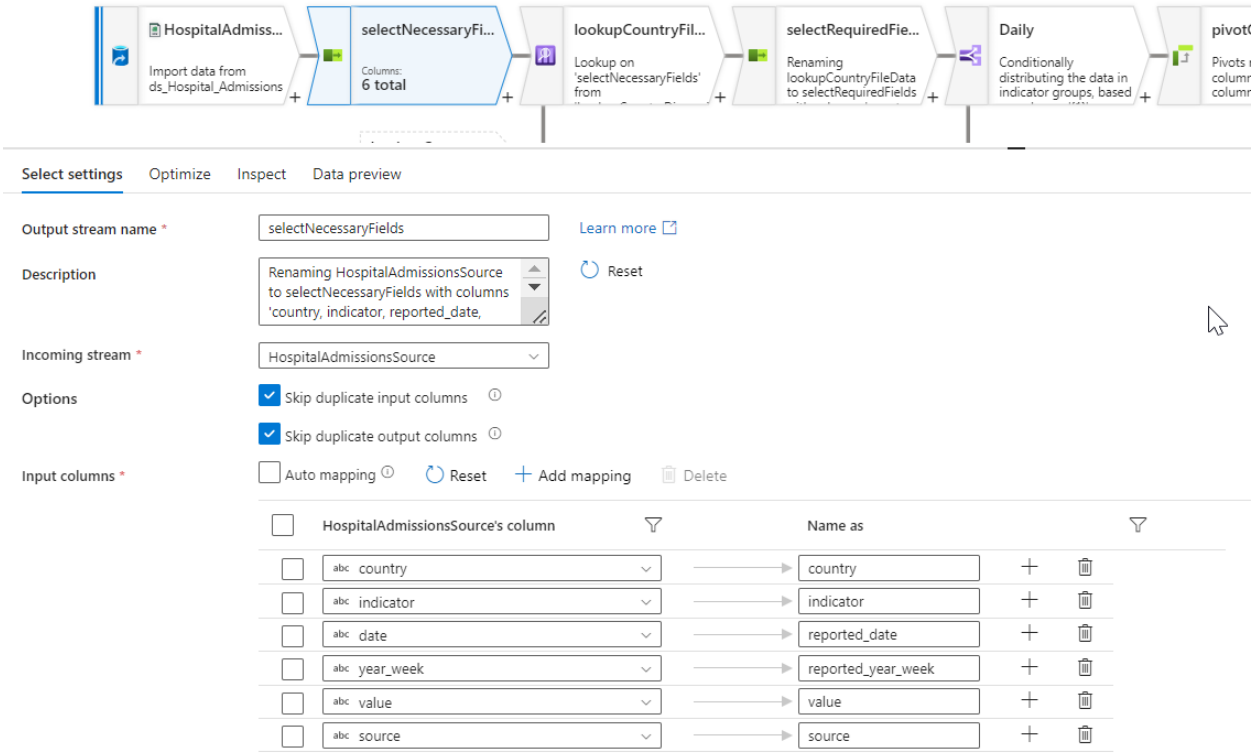
Dataset * LookupDateFile Test connection Open New

Options
- [x] Allow schema drift
- [ ] Infer drifted column types
- [ ] Validate schema

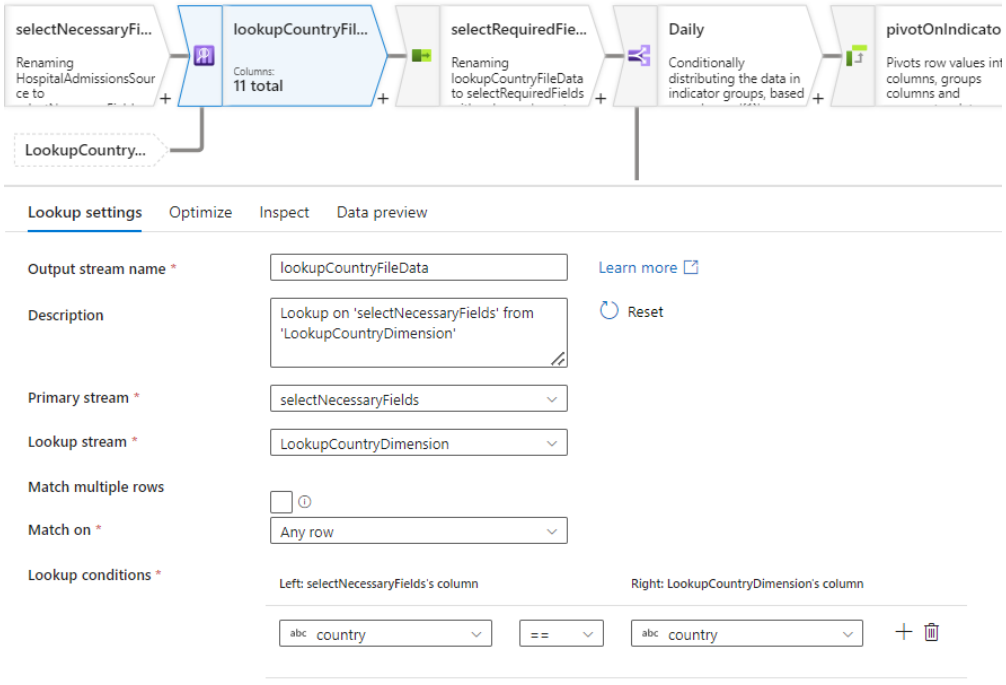
Skip line count

Sampling * Enable Disable

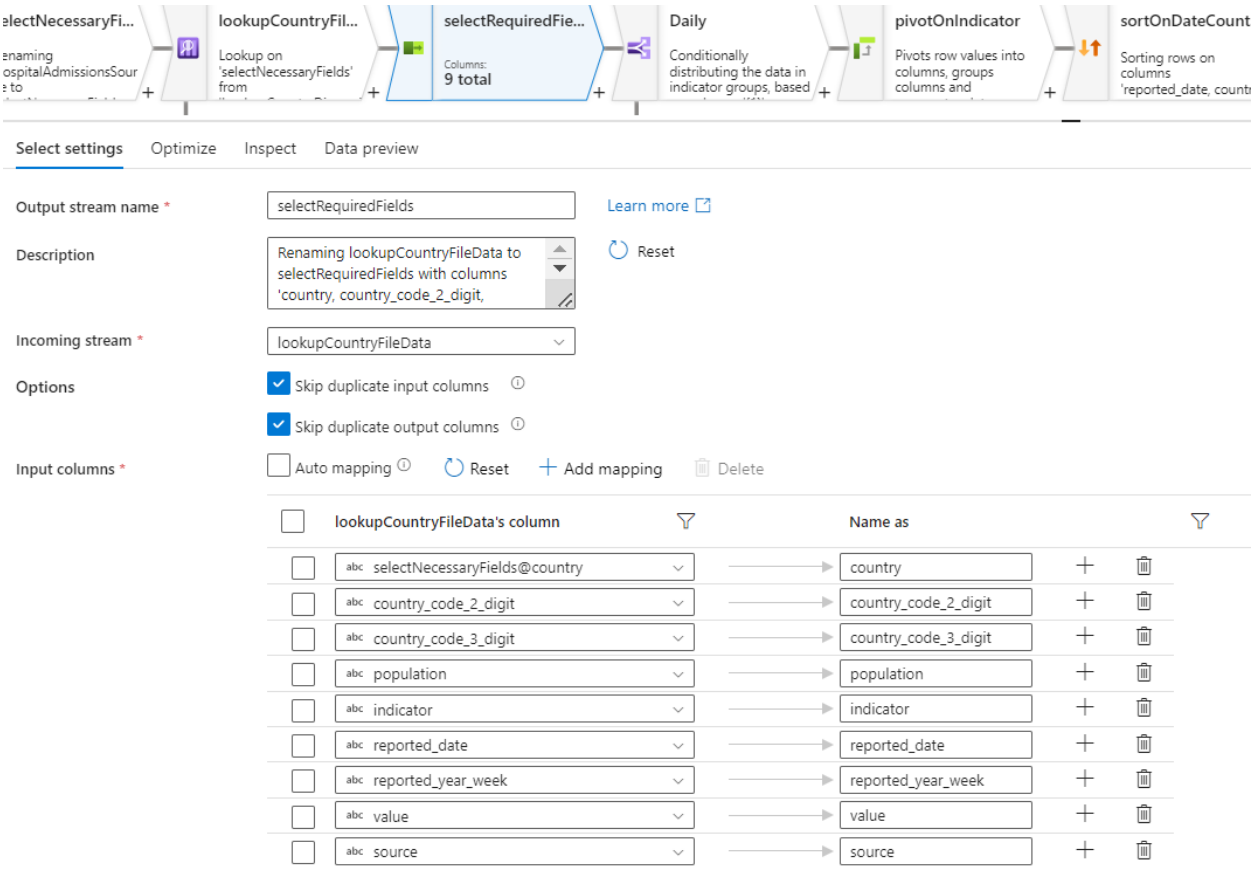
- Then first we will create a select activity on hospital_admissions.csv source activity and eradicate the URL column and select all other.



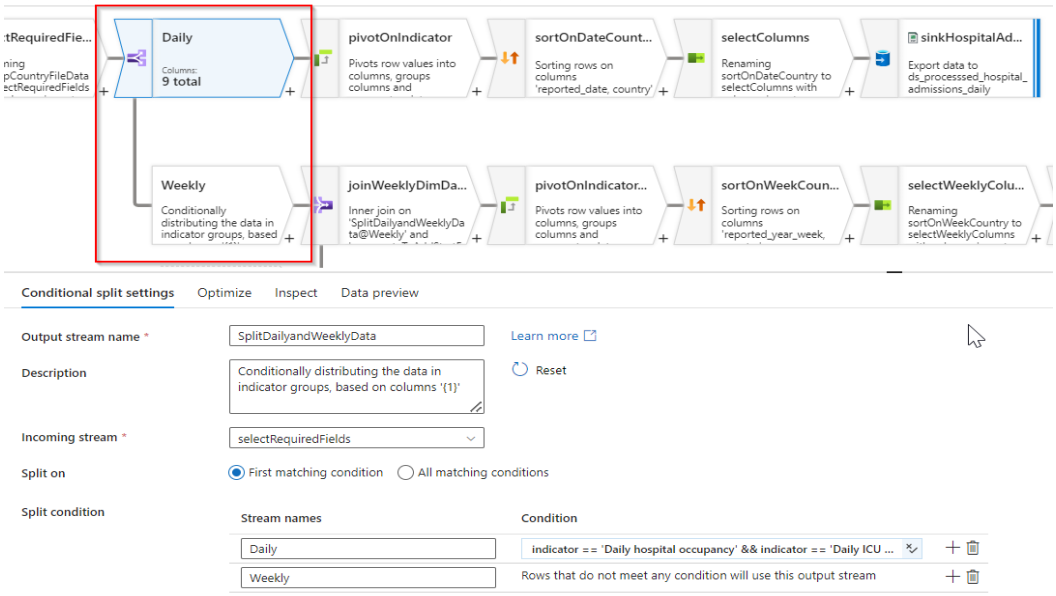
- Then we need 2 letter and 3 letter country code which we will get from lookup_country.csv file by looking up at country column. So we will then create a lookup activity to do the same.



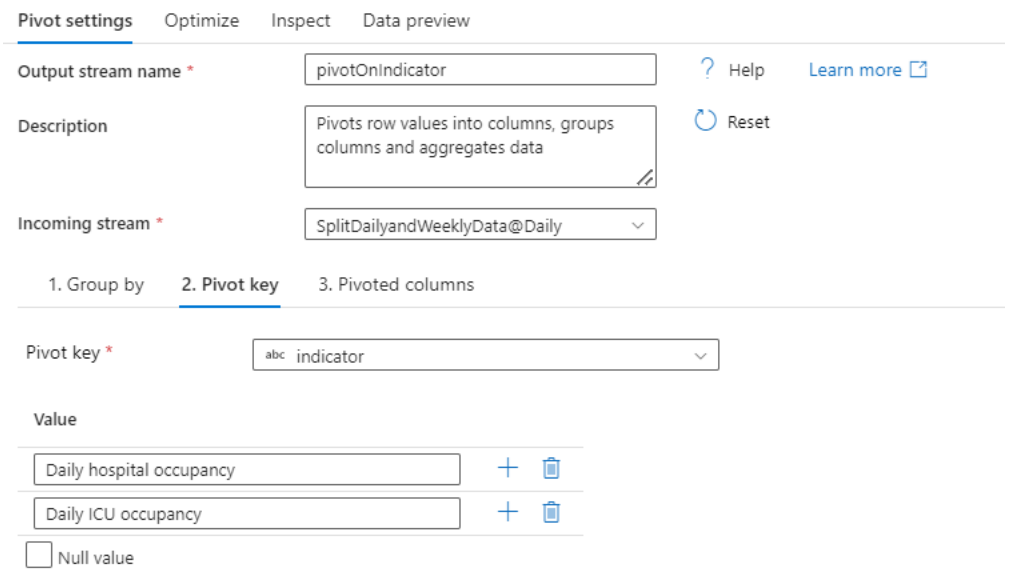
- Now again as after lookup we get some extra columns we will use select activity to select the required columns. We will keep 2 letter country code, 3 letter country code and population columns which we get after lookup with the existing columns and drop everything else.

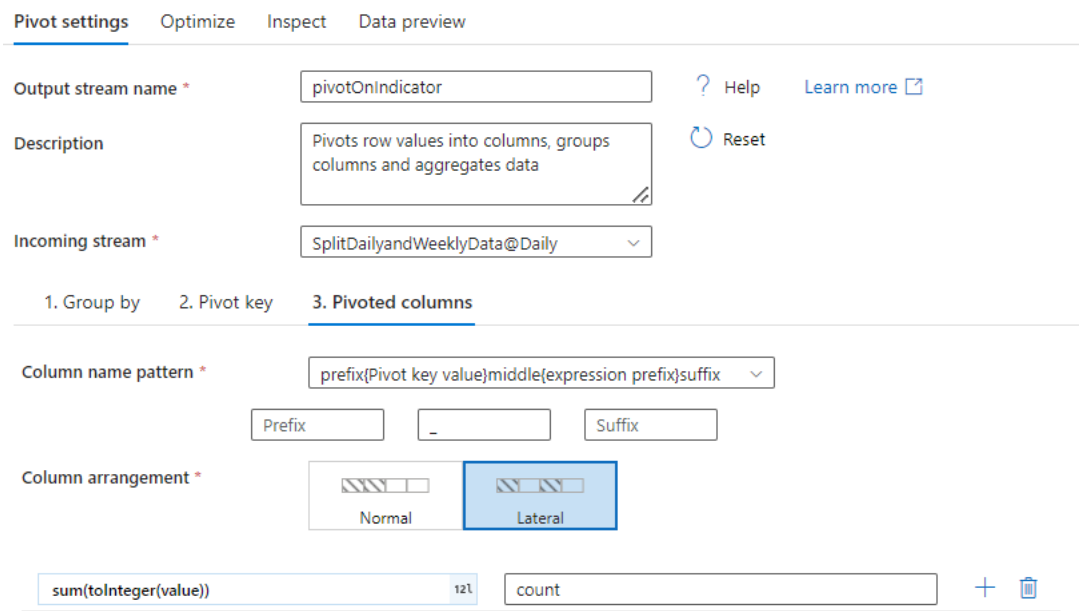


- Now as we want that the grain of the file should be either daily or weekly not both we will split the file and will daily related flag rows in Daily file and weekly flag related rows in weekly file.



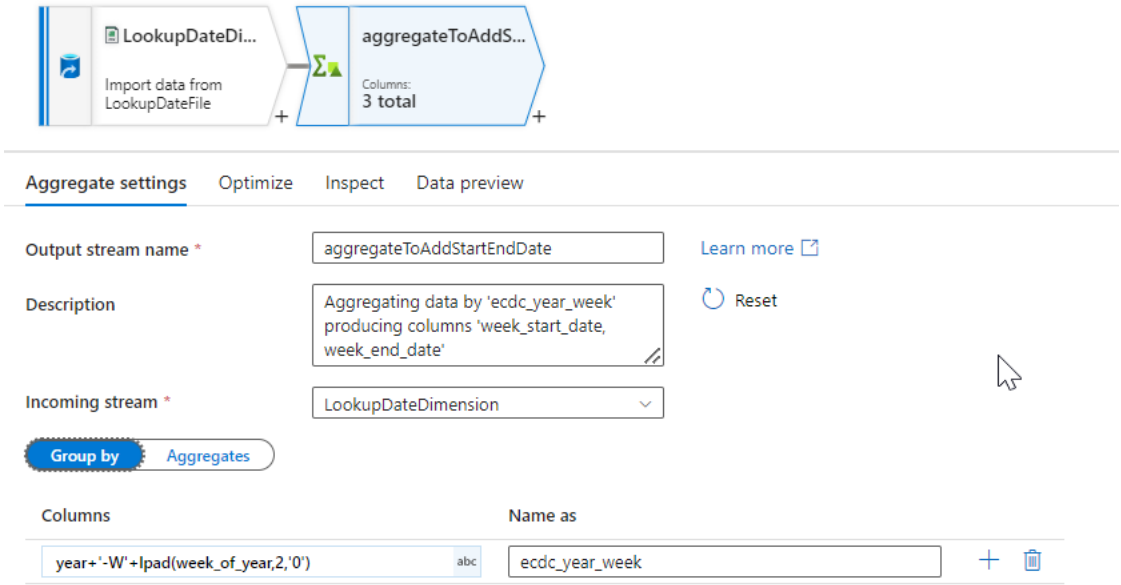
- On daily file then we will pivot the indicator and values columns.

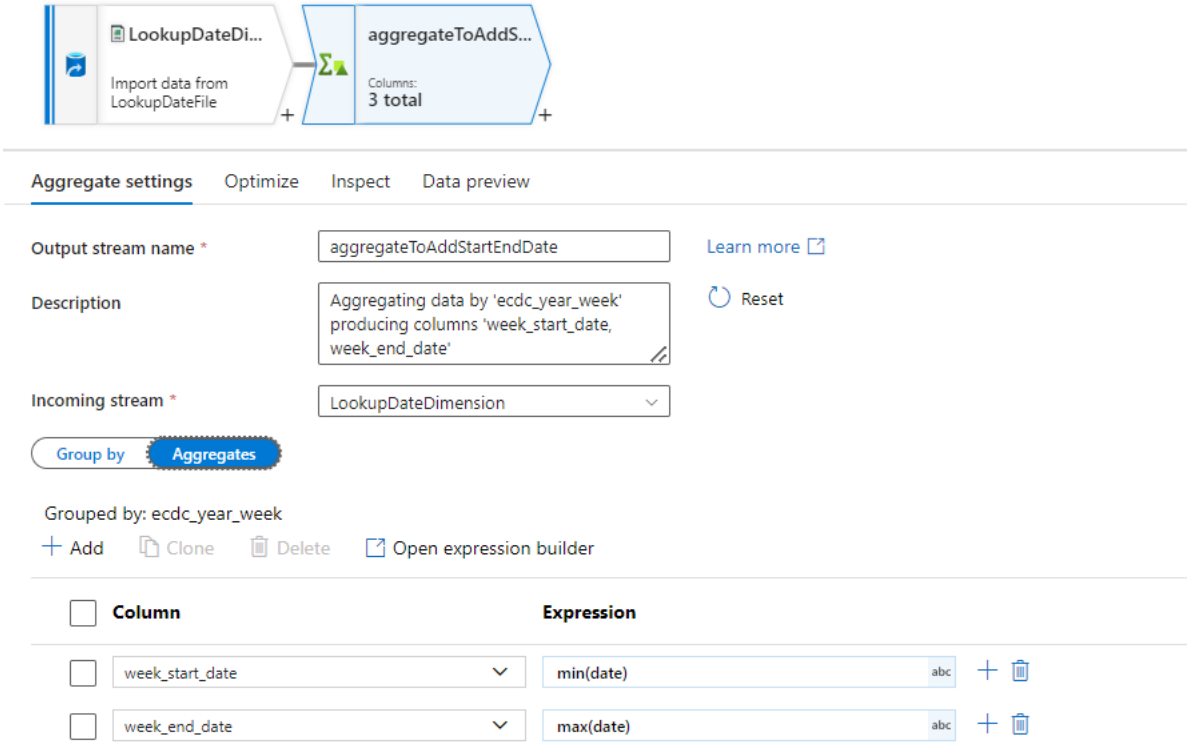




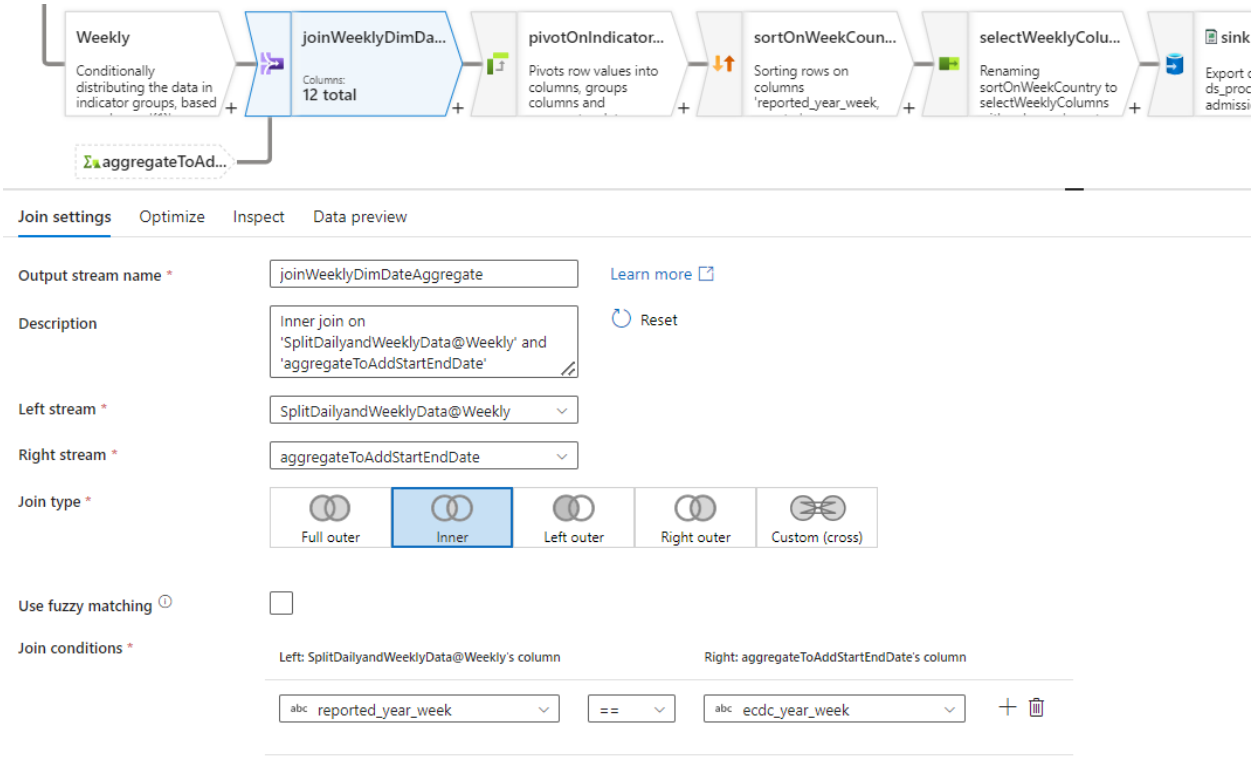
- On weekly file as we need to have start and end date. But on dim date file there is no column that is of same format as year_week.
    - So, we will first create an aggregate activity to create year_week like column with a column for start date and for end date.
    - So we will specify a new column 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).

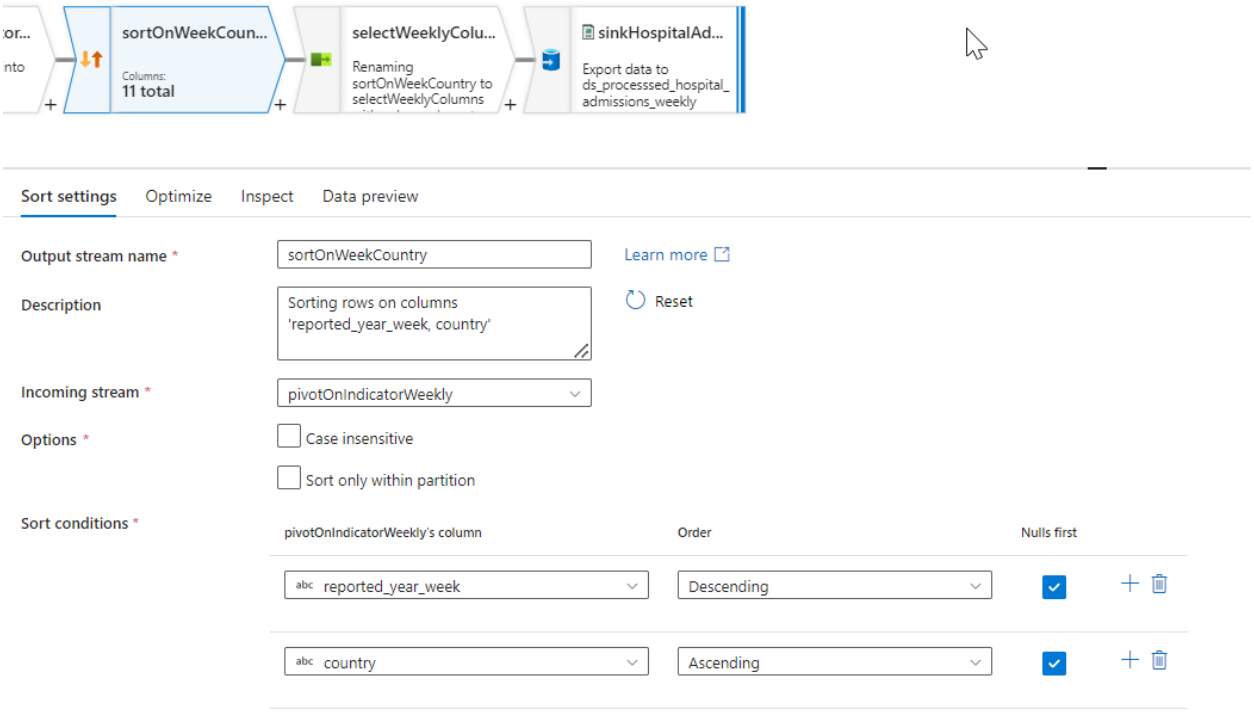


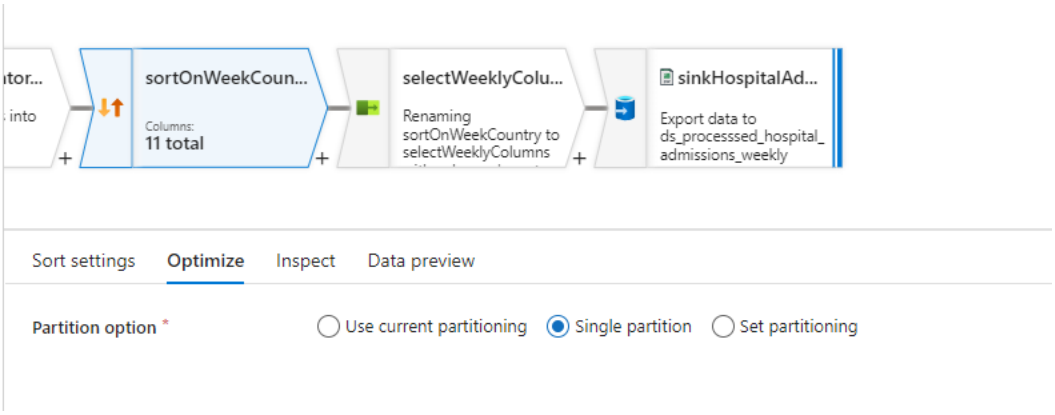


- Now we will join this aggregate output with weekly file on year_week == ecdc_year_week.



- 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





- 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.