**Batch Job:**

**About Batch Job:**

->Mule batch processing components are designed for reliable, asynchronous processing

of larger-than-memory data sets

-> Batch Job is Extracting transforming and loading (ETL) information into a target system

->It helps to process large volumes of data in chunks and in parallel

->The components are the Batch Job, Batch Step, and Batch Aggregator

**Batch Step component:**

* Process
* Load and dispatch
* On complete

**General Configuration of Batch Job**

**Name**: Name of the Batch JOB Activity in Mule Flow

**Max Failed Records**: **Finish processing**, **Continue processing** and **Continue processing** until the batch job accumulates a maximum number of failed records. This behaviour can be controlled by **Max Failed Records**.

* + - The default value is Zero which corresponds to Finish processing.
    - The value -1, corresponds to Continue processing.
    - The value +ve integer, corresponds to Continue processing until the batch job accumulates a maximum number of failed records.

**Scheduling Strategy:**

* + - **ORDERED\_SEQUENTIAL (Default)**

🡪Mule does not execute the second instance until the first one leaves.

the executable state.

* + - **ROUND\_ROBIN**

🡪 Multiple instancesruns at same time .

**Job Instance ID**:

* By default, the Batch Job Instance ID is created as UUID automatically by MuleSoft however this field can be used to overwrite default behaviour.

**Batch Block Size**:

* Number of records treated as chunks and processed by one thread. Default value is 100.

Example 1:-

->Block size value was 100

->Input records was 10

One Block was created.

Example 2:-

->Block size value was 3

->Input records was 10

In this case totally 4 blocks are created ,

\*In the first run it take the records of --> 1,4,7

\*In the second run it takes records of -> 2,5,8

\* In the third run it takes records of ---->3,6,9

\* In the fourth run it takes records of --> 10

**Max Concurrency**:

* + - Max number of Thread to start to process different blocks in parallel. Default and Max Value is 18

**HistoryConfiguration of Batch Job:**

**Max Age and Time Unit**:

* + - Batch process retains the history of batch instances in the temporary directory of Mule Runtime. By default, the retention policy is set to 7 days. A monitoring process will remove the temporary data that has met the expiration criteria. Using Max Age and Time Unit we can change this default behaviour.

**Batch Step configuration:**

A batch step uses two attributes to filter records:

**Accept Expression**:

* + - To process only records that evaluate to true; if the record evaluates to false, the batch step skips the record and sends it to the next one. In other words, the records with an accepted expression that resolves to false are the ones that Mule filters out.

**Accept Policy**:

* + - Batch step to process only the records which, relative to the value of the accept policy attribute, evaluate to true. Refer to the table below for a list of the available values for the accept policy.

**A screenshot of a step process

Description automatically generated**

Batch Aggregator:

Based on the defined Batch Aggregator size value , record wait for the defined size then processes the record

A batch Aggregator uses two attributes :

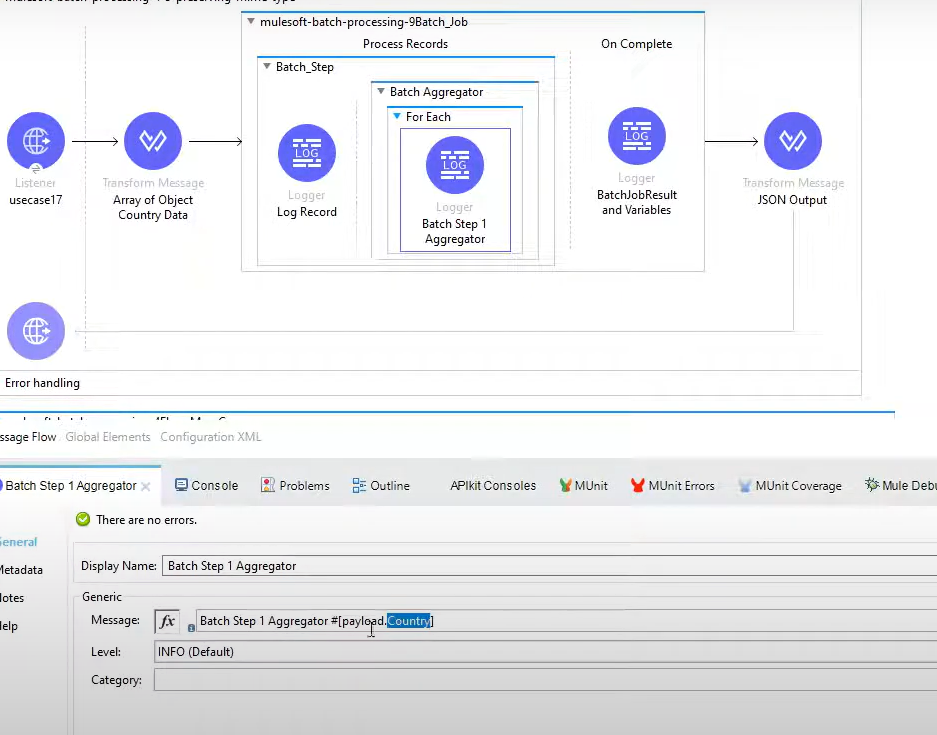
**Streaming:**

**Preserve mime Type:-**

* Batch Aggregator will not all mime Type in the record , so if you want to preserve mime type. Aggregator will not Forword mime type
* If every record should have mime type,then we must enable a preserve mime type.

Example :- in side Batch Aggregator we are using for each scope

In the for each Scope, we are using select operation



* In this case we need to use **Preserve mime Type**

Variables behaviour :

* Before the batch variable can be access on the inside of all the batch steps
* Over right can we possible inside the batch steps

Data Weave Functions for Error Handling:

#[Batch::isSuccessfulRecord()]

A boolean function that returns true if the current record has not thrown

exceptions in any prior step.

#[Batch::isFailedRecord()]

A boolean function that returns true if the current record has thrown

exceptions in any prior step.

#[Batch::failureExceptionForStep(String)]

Receives the name of a step as a String argument.If the current record threw exception on that step,

then it returns the actual Exception object. Otherwise it returns null

#[Batch::getStepExceptions()]

Returns a java Map<String, Exception> in which the keys are

the name of a batch step in which the current record has failed,

and the value is the exception itself.If the record hasn’t failed in

any step, this Map will be empty but will never be null.

Also, the Map contains no entries for steps in which the record

hasn’t failed.

#[Batch::getFirstException()]

Returns the Exception for the very first step in which the current

record has failed. If the record hasn’t failed in any step,

then it returns null.

#[Batch::getLastException()]

Returns the Exception for the last step in which the current record.

has failed. If the record hasn’t failed in any step, then it returns null.

Local Que path for the Batch Job :-

C:\Mule\AnypointStudio\plugins\org.mule.tooling.server.4.4.0.ee\_7.11.0.202206241916\mule\.mule\mule-poc's\queuestore