## Problem Statement:

In any E-commerce website when a new product has to be added for sale, it has to adhere to many business Rules. (For multiple countries i.e., product attributes change w.r.t country Ex: Language changes as per country)

Consider a case where we have to upload a large number of new products for each country to E-commerce website. Below are the problems:

1. If a product did not adhere to any one business rule it is rejected, but product owner is not able to figure out for which business rule it is failing to correct the data
2. If product owner is correcting one error by looking at the error report and again uploads data, it may fail for next business rule i.e. all possible errors are not captured in one go.
3. Management is not able to have a big picture of statistics i.e. for each country how many products passed and how many failed. So that they can concentrate on the country which has less data quality for product data.

## Problem Statement Details and Steps:

1. Input Files will be provided country wise i.e. each country will have separate input file.
   1. Place the input files in a folder and read each of the file iteratively (Implement the knowledge of iteration links and tFileList Component)
   2. Input file Structure will be as below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| File Name :  Products\_For\_India.csv |  | | | |
| Product ID | Model | Country | Available  QTY | Global  Delivery |
| 111 | 1988 | IN | 45 | NO |

* 1. After reading the input file, lookup for Price details in Price Lookup file (Implement the knowledge of join/lookup components)

|  |  |  |  |
| --- | --- | --- | --- |
| File Name :  Prices\_For\_India.csv |  | | |
| Product ID | Country | Price Per  QTY | Currency |
| 111 | IN | 45 | INR |

* 1. Load the looked up data in MYSQL database in below structure. (Implement the knowledge of MySql components)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table :  Products\_Staging |  | | | | | |
| Product ID | Model | Country | Available  QTY | Price Per  QTY | Currency | Global  Delivery |
| 111 | 1988 | IN | 45 | 2000 | INR | NO |

1. Once staging the data is ready in the MySQL database, apply the below business Rules.
   1. Product ID should be unique for each country. (i.e. same country cannot have two products with same product id, however same product can exist for two

Countries). Reject the product if it is duplicate for same country and log error Message.

(Implement knowledge of unique/ filter components)

* 1. Reject the product if either of the below fields are NULL or empty on invalid by logging separate error message for each field.
     1. Product ID
     2. Model.
     3. Country
     4. Price
     5. Currency.

(Implement the knowledge of tMap to divide the input rows based on conditions above)

1. Error Report Format should be as below and should be loaded to Hive:

(Implement the knowledge of HDFS components to move the generated file to HDFS and then load to Hive)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Product  ID | Country | Business Rule | Attribute | Reason |
| 1 | US | A | Product  ID | Duplicate Product  ID |
| 1 | IN | B | Currency | Currency is null |

1. On the data which has successfully passed the business rules, perform the following actions:
   1. Introduce a new column OUT\_OF\_STOCK and populate the value for that field

as TRUE if ‘Available QTY’ from input file is 0 else populate the field as FALSE.

* 1. Divide the products country wise.
  2. The final output which has be sent to Kafka will be country wise, as shown below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| KAFKA TOPIC :  INDIA\_PRODUCTS |  | | | | | | |
| Product ID | Model | Country | Available  QTY | Price Per  QTY | Currency | Global  Delivery | OUT\_OF\_STOCK |
| 111 | 1988 | IN | 45 | 2000 | INR | NO | FALSE |
| 222 | 2000 | IN | 0 | 1400 | INR | YES | TRUE |
| KAFKA TOPIC :  US\_PRODUCTS |  | | | | | | |
| Product ID | Model | Country | Available  QTY | Price Per  QTY | Currency | Global  Delivery | OUT\_OF\_STOCK |
| 111 | 1988 | US | 0 | 2000 | USD | YES | TRUE |

1. As the last step a reconciliation report is to be generated and loaded to PIG by getting details from input file and aggregating the Error Report data i.e.
   1. Have count of products from input file country wise (Available products).
   2. Have count of Products passed all business Rules (Processed Products).
   3. Aggregate the error report (because error report will have more than one error per product).

(Implement the knowledge of HDFS components to move the generated file to HDFS and then load it to PIG and later use PIG Aggregator for results).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Country | Total  Errors | Available  Products | Processed  Products | Rejected  Products |
| US | 3 | 2 | 1 | 1 |
| IN | 2 | 2 | 1 | 1 |

1. Once done with all the above steps, read the country wise processed data available in Kafka using Kafka input component and store them into HDFS as last step.

Ans.

1. right click on jobs and click on create job.in order to create a job and name it as project\_talend.
2. create a sub job to establish different connections.
3. drag tmysqlconnection to the canvas and mention properties like username,password,host and database name.
4. drag thdfsconnection to the canavs and mention the details required to connect to hadoop frame work.
5. drag thiveconnection to the canavs and mention the details required for connection.

note that “hive –service hiveserver2” command is entered in CLI before trying to connect to hive server.

1. drag tkafkaconnection to the canavs and mention the details required for connection.

note that set of commands should entered in CLI before trying to connect to kafka server.

1. add oncomponent trigger between the above components.
2. create a sub job to save the input files in mysql .
3. drag tfilelist\_1 to the canvas and provide the directory to read the list of files (related to product\_details ) from the directory.
4. drag tfileinputexcel\_1 to the canvas and take the filename with path from the outline tab and mention that piece of code in file location value in which it is present in the component view of tfileinputexcel\_1, mention the sheet name as “sheet1”.
5. now add iterate link from tfilelist\_1 to the tfileinputexcel\_1.
6. drag another tfilelist\_2 to the canvas and provide the directory to read the list of files (related to product\_price ) from the directory.
7. drag another tfileinputexcel\_2 to the canvas and take the filename from the outline tab and mention that piece of code in file location value in which it is present in the component view of tfileinputexcel\_2 , mention the sheet name as “sheet1”.
8. now add iterate link from tfilelist\_2 to the tfileinputexcel\_2.
9. drag tjoin to the canvas and mention the rows called product\_id and country as a key,provide the schema as per requirement and include the lookup columns also. (note that include look columns check box is checked).
10. drag tmysqloutput to the canvas and mention the property “use existing connection” and select tmysqlconnection under components list.now give table name as “product” and mention operation as create table if doesnt exist and action as insert or update.
11. drag tfileoutputdelimited\_1 to the canvas and mention the file properties required to store the rejected result.
12. now right click on tjoin ,select row>main and link it to the tmysqloutput.
13. once again right click on tjoin, select row>reject and link it to the tfileoutputdelimited\_1.
14. create a sub job to apply business rules and process the data according to that rules.
15. drag tmysqlinput to the canvas and mention the details for retrieving the table data.
16. now right click on tmysqlinput,select row>main and link it to the tuniqrow.
17. drag tuniqrow to the canvas,mention the schema and mention priduct\_id and country keys in which they should be unique .
18. drag tmap\_1 to the canvas and catch the duplicate values.mention the business rule “A” and reason “Duplicate value” in the tmap\_1 . create another out for accepted data without any error or rejects which named as “accepted”.
19. drag tmap\_2 to the canvas.
20. right click on tuniqrow,select row>uniqe and link it to the tmap\_2.
21. create a out main table in the tmap to filter the null values.mention the business rule “B” and keep particular column is null.
22. drag tunite to the canavs and link both tmap\_1 and tmap\_2 output to the tunite for combining the result.
23. drag thiveoutput and mention the details required to create hive table.
24. now link tunite to thiveoutput to store the result in the hive database.
25. you can check the table in hive.
26. for the data which is according to business rule.
27. drag tmap\_3 to the canvas and create two out main tables in which one for india\_product\_details and another for us\_product\_details.
28. right click on tmap\_2,select row>accepted and link it to the tmap\_3.
29. in tmap create OUT\_OF\_STOCK column and populate the value for that field as TRUE if ‘Available QTY’ from input file is 0 else populate the field as FALSE by using condinal operator.
30. now create out\_india table and out\_us. for that you should use equals() method for checking country either india or US.
31. drag tkafkacreatetopic\_1 and fill fields which are required
32. drag tkafkaoutput\_1 to the canvas and fill the required fields to store the india product details.
33. now add oncomponentOk trigger from tkafkacreatetopic\_1 to tkafkaoutput\_1
34. drag tkafkacreatetopic\_2 and fill fields which are required.
35. drag tkafkaoutput\_2 to the canvas and fill the required fields to store the india product details.
36. now add row oncomponentOk trigger from tkafkacreatetopic\_2 to tkafkaoutput\_2
37. now link tmaps out\_india row to tkafkaoutpu\_1 and out\_US row to tkafkaoutpu\_2.
38. after execution of the job you can check the result in the server.
39. now create another subjob for reconciliation report.
40. drag thiveinput to the canvas, in which we stored the error reports in the hive. give the required details .
41. drag tfilterrow\_1. link thiveinput to the tfilterrow\_1 and filter the india and US columns.
42. drag tmap to the canvas and create out\_error table.
43. now add oncomponentok trigger from tfilterrow\_1 to tmap.
44. create following columns in tmap

country, Total Error, Available Products, Processed Products and Rejected Products

1. take nb\_line which is under tfilterrow\_1 from outline tab and mention it beside the Total Error expression value for respective country.
2. tkafkainput\_1 and tkafkainput\_2 and link it to the taggregaterow\_1 and taggregaterow\_2 to sum up the available products.
3. now link taggregaterow to tmap.
4. store the result in available products of tmap of their respective country.
5. again take nb\_line from outline tab under tkafkainput\_1 and tkafkainput\_2.mention those pieces of code under expression value of processed data in their respective country.
6. drag tfileoutputdelimited to the canavs and tfilterrow\_2 also.
7. link the tfileinputdelimited component with the row-->main to the tfilterrow\_2.
8. now filter the rejected output with respective country either india or US using tfilterrow\_2.
9. now add a triger oncomponentok from tfileinputdelimited to tmap.
10. once again take nb\_line from outline tab under tfilterrow\_2 and mention those piece of code under expression value of country column.
11. drag thdfsput to canvas.
12. link tmap to thdfsput and store the result in hdfs.
13. drag tpigload and give the required details.
14. now add a onsubjobok trigger from thdfsput to tpigload.
15. drag tpigstoreresult to the canvas and mention the required fields.
16. link tpigload to tpigstoreresult and fianlly save the result.
17. you can also check the result in server.
18. create a subjob to store the data into hdfs.
19. drag tkafkainput\_1 to the canvas and mention the required details to retrieve the data from kafka server.
20. drag thdfsoutput\_1 to the canvas and mention the required details in order to store the data in hdfs server.
21. now link tkafkainput\_1 to thdfsoutput\_1.
22. drag tkafkainput\_2 to the canvas and mention the required details to retrieve the data from kafka server.
23. drag thdfsoutput\_2 to the canvas and mention the required details in order to store the data in hdfs server.
24. now link tkafkainput\_2 to thdfsoutput\_2.
25. go and check in the hdfs server to see the final result of country wise processed data.
26. now connect all sub jobs with onsubjobok trigger so that subjobs executes after another.
27. finally save the job .
28. click on run and check the result.