**INVISTA – Supply Chain Analytics**

**Technical Documentation for Data Automation**

A picture containing clipart

Description generated with very high confidence

**Prepared by: KBS Analytics Solutions**

***Table of contents***

|  |  |
| --- | --- |
| **Section** | **Page** |
| Project Overview | 3 |
| High Level Architecture | 4 |
| High Level Data Flow | 5 |
| Data Flow Details - Diagram | 6 |
| Data Flow Details - Description | 7 |
| AWS IAM Access and Security | 10 |
| Source Systems | 10 |
| Redshift Cluster | 11 |
| Matillion Instance  Matillion Job Details | 11  12 |
| AWS Services | 12 |
| AWS S3 Folder Structure | 14 |
| Source Code (Version Control) | 15 |
| Maintenance | 17 |

Document Revision History 18

# Project Overview

INVISTA has requested support from KBS Analytics Solutions to improve analytics-driven supply chain decisions. The objective of the project is to improve the accuracy and confidence of forecasts used by supply managers responsible for nylon intermediates at the Orange plant.  The KBS Analytics Solutions will leverage deep machine learning to create forecasts to have more data driven discussions for better supply and demand balancing for the INVISTA supply chain.

The project will contain multiple phases.  The current initiative focus is phase 1, which includes:

* Introduce AWS Machine Learning design patterns to the INVISTA IT/Data Science Capability
* Provide a meaningful, credible set of models, incorporated with existing processes to improve forecast accuracy for production of ADN at the Orange plant

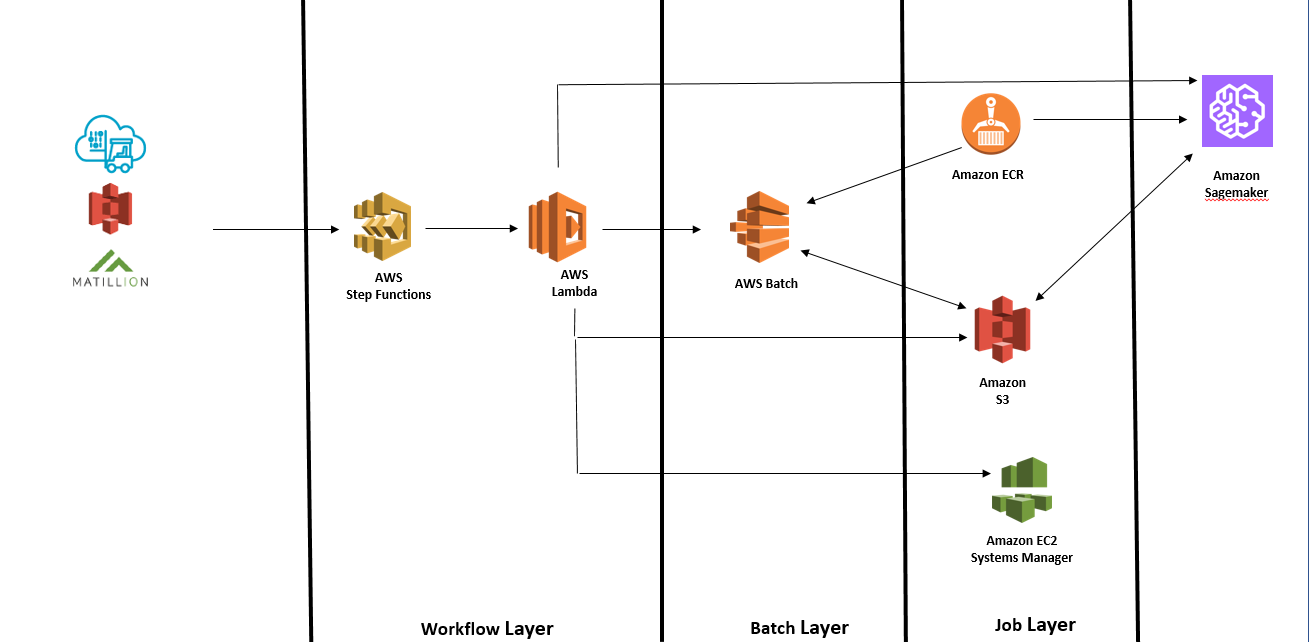
# High Level Architecture

# 

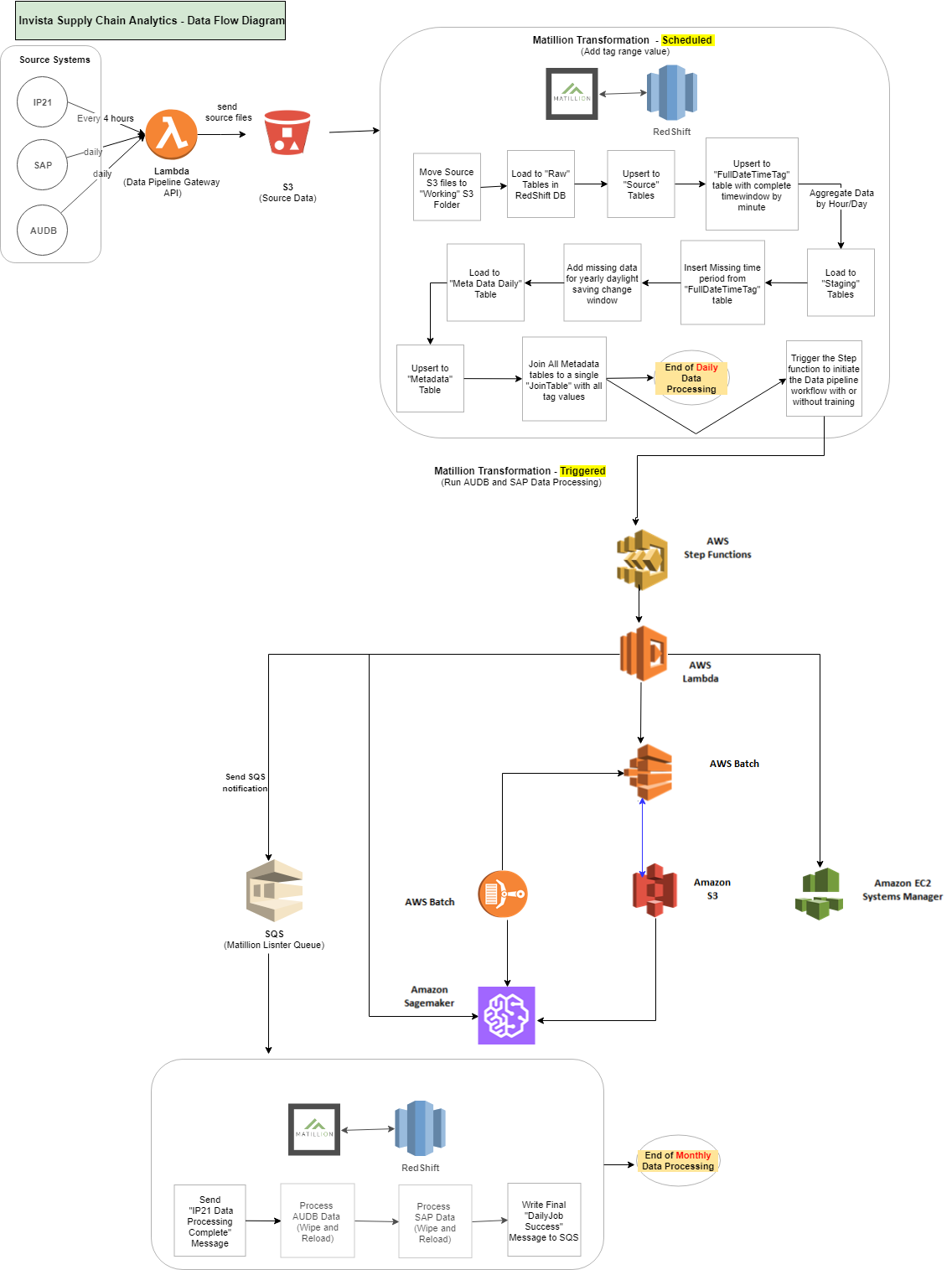
# High Level Data Flow

# 

# Invista Supply Chain Data Pipeline Workflow (High Level)



# Invista Supply Chain Data Pipeline Workflow (Detailed)



# Data Flow Details – Description

The Invista data pipeline workflow has different function components. The visual workflow coordinates these components. Each individual component performs a discrete function that will allow Invista to scale and modify the pipeline quickly.

Below is the overview of the AWS Services

Below is walk through of the Data pipeline implemented,

1. Matillion triggers step functions through lambda to run the workflow with or without training.
2. State machine checks to make sure the added file in the S3 bucket has new data since the last prediction run, pulls data file from S3 to be used for the ip21add-stats job, adds the file to manifest, and submit the ip21addstats batch job.
3. State machine periodically checks the status of the ip21addstats job until it is successfully completed.
4. If the ip21addstats job completed, sate machine goes through the same process as in step 2 and 3 to run the final data preparation batch job.
5. If state machine was triggered to run with model training, sate machine does similar process in step 2 and 3 to run the Hyperparameter tuning job.
6. State machine exports the best training job.
7. If state machine was triggered to run without model training, it skips step 6. State machine goes through similar process as in step 2 and 3 to run the batch job for prediction and then generate prediction from the latest best training job.
8. State machine sends out email notifications through SQS providing status update for the batch and tuning jobs at each step
9. **Daily Data Processing**
10. Invista sends source files to S3.
    1. Source file location:

IP21: S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/LiveData/

AUDB: S3://inv-nonprod-aa-scm/audbdata/fullload/csv/SourceData/

SAP: S3://inv-nonprod-aa-scm/sapdata/fullload/csv/SourceData/

* 1. Source file reception interval:

IP21: every 4 hours (00, 04, 08, 12, 16, 20th hour daily)

AUDB: not automated (not used by predictive model yet)

SAP: not automated (not used by predictive model yet)

1. Matillion daily scheduled job processes the data and stores it in RedShift DB.

2-1. Matillion job name:

MasterJob\_InvSupplyChainAnalytics\_DailyScheduledLoad

2-2. Matillion job schedule:

00:30 daily except for 2nd day of month

2-3. Transformation Steps:

2-3.1 Clear working folder by moving files in working folder (S3://inv-nonprod-aa- scm/ip21data/csv/alltags/sapc11/ProcessingData/) to “unprocessed” folder (S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/UnprocessedData/).

2-3.2 Move IP21 source files to working folder (S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/ProcessingData/).

2-3.3 Loop 159 IP21 tags defined in “distincttagnames” table in RedShift DB and do operations for each tag:

2-3.3.1. Truncate “raw” table in RedShift DB.

2-3.3.2. Load IP21 source data from S3 to RedShift “raw” table.

2-3.3.3. Upsert data from “raw” table to “source” table.

2-3.3.4. Write to “fulldatetimetag\_daily” table with tag and distinct datetime from the raw data.

2-3.3.5. Load data to “staging” table aggregated by hour.

2-3.3.6. Insert missing time period in “staging” table from “fulldatetimetag\_daily” table.

2-3.3.7. Insert missing time period caused by day light saving time change

2-3.3.8. Get below/above/within/undefined indicator for the tag value and write to “meta daily” table.

* Tag’s min/max value range is defined in “distincttags\_metadata” table.
  + - 1. Upsert from “metadata daily” table to “metadata” daily table.

2-3.4 Create “jointable” with all tags data.

2-3.4.1 Create “jointablesbyhour” table with all tags with all history from 2000 (historical data).

2-3.4.2 Create “jointablesbyhour\_daily” table with all tags with data in the “raw” table (daily data).

1. **Monthly Data Processing**
2. Matillion monthly scheduled job processes the data and stores it in RedShift DB.

1-1 Matillion job name:

MasterJob\_InvSupplyChainAnalytics\_MonthlyScheduledLoad

1-2 Matillion job schedule:

00:30 2nd day of month

1-3 Transformation Steps:

1-3.1 Clear working folder by moving files in working folder (S3://inv-nonprod-aa scm/ip21data/csv/alltags/sapc11/ProcessingData/) to “unprocessed” folder (S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/UnprocessedData/).

1-3.2 Move IP21 source files to working folder (S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/ProcessingData/).

1-3.3 Loop 159 IP21 tags defined in “distincttagnames” table in RedShift DB and do operations for each tag:

1-3.3.1 Truncate “raw” table in RedShift DB.

1-3.3.2 Load IP21 source data from S3 to RedShift “raw” table.

1-3.3.3 Upsert data from “raw” table to “source” table.

1-3.3.4 Write to “fulldatetimetag\_daily” table with tag and distinct datetime from the raw data.

1-3.3.5 Load data to “staging” table aggregated by hour.

1-3.3.6 Insert missing time period in “staging” table from “fulldatetimetag\_daily” table.

1-3.3.7 Insert missing time period caused by day light saving time change

1-3.3.8 Get below/above/within/undefined indicator for the tag value and write to “meta daily” table.

* Tag’s min/max value range is defined in “distincttags\_metadata” table.

1-3.3.9 Upsert from “metadata daily” table to “metadata” daily table.

1-3.3 Create “jointable” with all tags data.

1-3.3.1 Create “jointablesbyhour” table with all tags with all history from 2000 (historical data).

1-3.3.2 Create “jointablesbyhour\_daily” table with all tags with data in the “raw” table (daily data).

1-3.4 Unload “jointablesbyhour” table data from RedShift to S3.

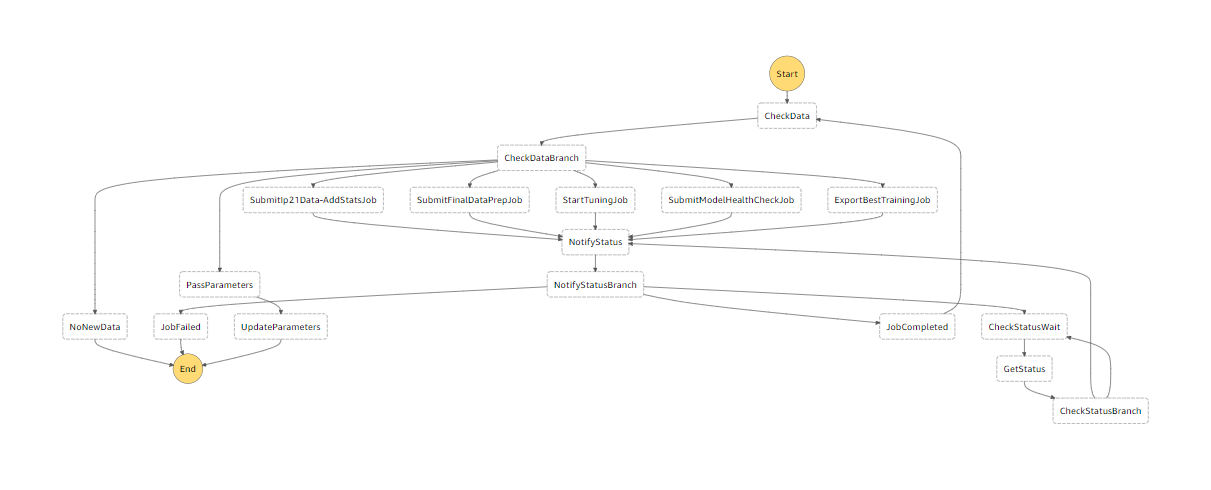
1-3.4.1 Unload S3 location: S3://inv-nonprod-aa-scm/ip21data/UnloadfromRedshift/

1. Matillion job to trigger the data processing, training and prediction pipeline with or without training the Model

To Explain the entire workflow, we can refer the below diagram. The workflow pipeline has state machine steps that passes input to the specified Lambda function.

Below is walk through of the entire workflow end to end,

* 1. Matillion triggers step functions through lambda to run the workflow with or without training.
  2. State machine checks to make sure the added file in the S3 bucket has new data since the last prediction run, pulls data file from S3 to be used for the ip21add-stats job, adds the file to manifest, and submit the ip21addstats batch job.
  3. State machine periodically checks the status of the ip21addstats job until it is successfully completed.
  4. If the ip21addstats job completed, sate machine goes through the same process as in step 2 and 3 to run the final data preparation batch job.
  5. If state machine was triggered to run with model training, sate machine does similar process in step 2 and 3 to run the Hyperparameter tuning job.
  6. State machine exports the best training job.
  7. If state machine was triggered to run without model training, it skips step 6. State machine goes through similar process as in step 2 and 3 to run the batch job for prediction and then generate prediction from the latest best training job.
  8. State machine sends out email notifications through SQS providing status update for the batch and tuning jobs at each step



1. Lambda Functions used to run the entire workflow is explained here,
   1. Inv-supplychain-workflow-input.py:

Receives input from matillion whether to run the workflow with or without model training

* 1. Inv-supplychain-workflow-input.py:

Receives input from matillion whether to run the workflow with or without model training

* 1. Inv-supplychain-checkdata.py:

Has all the configuration for the model tuning and checks existence of file before executing the next step

* 1. Inv-SupplyChain-ip21-AddStats-JobStatusPol-SubmitJob.py:

Submits a batch job to add statistical calculations to the data

* 1. Inv-SupplyChain-FinalDataPrep-JobStatusPol-SubmitJob.py:

Submits a batch job to do the final data preparation

* 1. Inv-supply-chain-start-tuning-job.py:

Creates hyperparameter tuning job

* 1. Inv-SupplyChain-ModelHealthCheck-JobStatusPol-SubmitJob.py:

Submits a batch job to run predictions

* 1. Inv-supplychain-output-best-training-job.py:

Exports the best training job

* 1. Invista-supplychain-get-status.py:

Get the status of AWS batch and Sagemaker during the execution

* 1. Inv-supplychain-notify-status.py:

notify status of the execution.

* 1. Inv-supplychain-model-input-update-parameters.py:

This updates the date when the final prediction

* 1. Inv-supplychain-workflow.yaml:

This is AWS step functions code that coordinates and run the lambda functions. State machine inv-supply chain-workflow coordinates the workflow.

* 1. Matillion-run-workflow-with-training.py:

Matillion invokes lambda (inv-supplychain-workflow-input.py) through this code. Lambda intern invokes the step function to run the work flow with training

* 1. Matillion-run-workflow-with-training.py:

Does the same thing as above except it sends data to run the work flow without model training?

1. First AWS Batch job is triggered by Lambda function to run a Python script to add additional statistical calculations to the Matillion-produced data set.

4-1 Lambda function:

4-1.1 Name: Inv-SupplyChain-ip21-AddStats-JobStatusPol-SubmitJob.py

4-1.2 Calls AWS Batch job ‘[inv-supplychain-ip21data-addstats’.](https://console.aws.amazon.com/batch/home?region=us-east-1)

4-1.3 AWS Batch job executes Python script in Docker image that adds 2000 + statistical attributes to input data set.

4-1.4 ECR Repository name containing Docker image: inv-supplychain-ip21data-addstats

4-1.5 Creates output file to S3://inv-nonprod-aa-scm/ModelInputFile/InvistaSupplyChain\_ModelInput.txt.

1. Second AWS Batch job is triggered by Lambda function to run a Python script to do additional data processing.

5-1 Lambda function:

5-1.1 Name: Inv-SupplyChain-FinalDataPrep-JobStatusPol-SubmitJob.py 3-2.2 Calls AWS Batch job ‘[inv-supplychain-final-data-prep](https://console.aws.amazon.com/batch/home?region=us-east-1) ’.

5-2.2 AWS Batch job executes Python script in Docker image that applies additional data processing (rescaling/imputation/cleanup).

5-2.3 ECR Repository name containing Docker image: inv-supplychain-final-data-prep

5-2.4 Creates output file to S3://inv-nonprod-aa-scm/ModelInputFile/rescaled/data.csv.

1. Third AWS Batch job is triggered by Lambda function to run a Python script to do Model health and prediction.

6-1 Lambda function:

6-2.1 Name: Inv-SupplyChain-FinalDataPrep-JobStatusPol-SubmitJob.py 3-2.2 Calls AWS Batch job ‘[inv-supplychain-final-data-prep](https://console.aws.amazon.com/batch/home?region=us-east-1) ’.

6-2.2 AWS Batch job executes Python script in Docker image that applies additional data processing (rescaling/imputation/cleanup).

6-2.3 ECR Repository name containing Docker image: inv-supplychain-final-data-prep

6-2.4 Creates output file to S3://inv-nonprod-aa-scm/ModelInputFile/rescaled/data.csv.

1. Lambda function is triggered to execute Matillion job that processes AUDB and SAP data.

7-1 Lambda Trigger: file creation in S3 (S3://inv-nonprod-aa-scm/ModelInputFile/Rescaled/).

7-2 Lambda function:

7-2.1 Name: Inv-supplychain-modelinput-notification

7-2.2 Executes Matillion job ‘Masterjob-MonthlyLoad2-Triggered‘by writing to SQS queue and sends job success notification.

* + - Matillion Listen Queue: inv-nonprod-aa-scm-orange-ip21
    - AUDB and SAP data processing steps are currently disabled as the data is not used by the predictive model.

# AWS IAM Access and Security

These policies are used on KBS-Analytics IAM and KBS-Analytics-Lambda roles for development purpose in Dev environment. INVISTA will need to setup equivalent role with similar IAM policies.

The access and security will need to be adjusted and tested with deployment to Production environment.

|  |  |
| --- | --- |
| **Policies** | **Sub-Policies** |
| [AWSLambdaFullAccess](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAWSLambdaFullAccess) | N/A |
| [ReadOnlyAccess](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FReadOnlyAccess) | [kbs-ec2-allow](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3A291428023636%3Apolicy%2Fkbs-ec2-allow) |
| [ReadOnlyAccess](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FReadOnlyAccess) | [kbs-analytics](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3A291428023636%3Apolicy%2Fkbs-analytics) |
| [AmazonAthenaFullAccess](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonAthenaFullAccess) | N/A |
| [AmazonRedshiftFullAccess](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonRedshiftFullAccess) | [kbs-kms](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3A291428023636%3Apolicy%2Fkbs-kms) |
| [AmazonSageMakerFullAccess](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonSageMakerFullAccess) | N/A |
| [AWSGlueConsoleFullAccess](https://console.aws.amazon.com/iam/home?#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAWSGlueConsoleFullAccess) | N/A |
| AmazonSQSFullAccess | N/A |
| AWSLambdaVPCAccessExecutionRole | N/A |
| ECR-Lambda | N/A |
| Kbs-analytics-lambda | N/A |
| SageMaker-Lambda | N/A |
| Sagemaker-role-batch-integration | N/A |

# Source Systems

|  |  |
| --- | --- |
| **Source Name** | **Description** |
| IP21 | Plant historian data which contains: equipment sensor, temperature sensor and lab sample data. During phase one of the project data was collected for 159 sensors ranging from 2000-present at the Orange, TX site. IP21 is the main source of data for the existing predictive model. |
| SAP | The stream of SAP data is narrowed to relevant master data and the PM module only. Master data contains: general SAP equipment and location data. The PM module houses data for: plant maintenance workplans, work plans and scheduled maintenance items. This data is not used in the model at this time but may be leveraged in future iterations. |
| AUDB | Production outage and loss data. AUDB quantifies and categorizes plant losses and provides reasons for each loss. |

# Redshift Cluster

|  |  |  |
| --- | --- | --- |
| **Cluster Name** | **Database Name** | **Username** |
| jdbc: redshift://invista-aa-dev-scm.cytqcbjz8llt.us-east-1.redshift.amazonaws.com:5439/invistaaadevscm | invistaaadevscm | Invistaaadevscm |

# Matillion Instance

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EC2 Server Name** | **Instance Name** | **Admin User Account Name** | **EC2 Server Scheduled Running Window** | **Note** |
| invista-matillion-aa-dev | ec2-*[dynamic]*.  compute-1.amazonaws.com | ec2-user | 12:00am –  3:00am daily | * Once Matillion instance stops and starts, the DNS instance will change (not a static URL). This is to avoid charge by using static IP assigned to the EC2 machine. * Matillion Admin should set up additional user account for each developer. |

# Matillion Job Details

|  |  |  |  |
| --- | --- | --- | --- |
| **Project** | **Daily schedule job name** | **Monthly schedule job name** | **Email Notification** |
| InvistaSupplyChainAnalytics | **MasterJob-DailyLoad-Scheduled**  (schedule: daily at 00:30 am except for 2nd day of month) | **MasterJob-MonthlyLoad1-Scheduled**  (schedule: 2nd of month at 00:30 am) | [InvistaBIDevelopers@invista.com](mailto:InvistaBIDevelopers@invista.com) |

# AWS Services

|  |  |  |
| --- | --- | --- |
| **AWS Service Name** | **Object Name** | **Description** |
| EC2 | invista-mtillion-aa-dev | EC2 Server that runs Matillion jobs |
| Batch | [inv-supplychain-ip21data-addstats-batch](https://console.aws.amazon.com/batch/home?region=us-east-1#/queues/arn%3Aaws%3Abatch%3Aus-east-1%3A291428023636%3Ajob-queue~2Finv-supplychain-ip21data-addstats-batch) | ETL job for feature engineering (creating mean, standard deviation, and ratio calculations) |
| inv-supplychain-final-data-prep | ETL job for rescaling the feature engineering (creating mean, standard deviation, and ratio calculations) |
| S3 | [inv-nonprod-aa-scm](https://s3.console.aws.amazon.com/s3/) | File storage for raw data from source systems, cleansed data, model input, and model output |
| VPC | [vpc-a170dada](https://console.aws.amazon.com/vpc/home?region=us-east-1#vpcs:filter=vpc-a170dada) | Virtual private cloud for the INVISTA Supply Chain Analytics project |
| IAM | kbs-analytics (role) | Identity & Access Management for securing the cloud environment |
| Redshift | invista-aa-dev-scm | Database for Matillion transformations |
| SQS | Inv\_nonprod-aa-scm-orange-ip21 | Listener queue for Matillion job initiation |
| Inv\_nonprod-aa-scm-orange-ip21-success | Queue for job success message |
| Inv\_nonprod-aa-scm-orange-ip21-failure | Queue for job failure message |
| SNS | Inv\_SupplyChainAnalytics\_IP21 | Success/Failure notifications (email) for ETL jobs |
| CloudWatch | Kbs-managed-lambda | Scheduled to invoke lambda to run Sagemaker training jobs |
| Lambda | Inv-supplychain-workflow-input.py: | Receives input from matillion whether to run the workflow with or without model training |
| Lambda | Inv-supplychain-workflow-input.py: | Receives input from matillion whether to run the workflow with or without model training |
| Lambda | Inv-supplychain-checkdata.py | Has all the configuration for the model tuning and checks existence of file before executing the next step |
| Lambda | Inv-SupplyChain-ip21-AddStats-JobStatusPol-SubmitJob.py: | Submits a batch job to add statistical calculations to the data |
| Lambda | Inv-SupplyChain-FinalDataPrep-JobStatusPol-SubmitJob.py: | Submits a batch job to do the final data preparation |
| Lambda | Inv-supply-chain-start-tuning-job.py: | Creates hyperparameter tuning job |
| Lambda | Inv-SupplyChain-ModelHealthCheck-JobStatusPol-SubmitJob.py: | Submits a batch job to run predictions |
| Lambda | Inv-supplychain-output-best-training-job.py: | Exports the best training job |
| Lambda | Invista-supplychain-get-status.py: | Get the status of AWS batch and Sagemaker during the execution |
| Lambda | Inv-supplychain-notify-status.py: | notify status of the execution. |
| Lambda | Inv-supplychain-model-input-update-parameters.py: | This updates the date when the final prediction |
| Lambda | Inv-supplychain-workflow.yaml: | This is AWS step functions code that coordinates and run the lambda functions. State machine inv-supply chain-workflow coordinates the workflow. |
| Lambda | Matillion-run-workflow-with-training.py: | Matillion invokes lambda (inv-supplychain-workflow-input.py) through this code. Lambda intern invokes the step function to run |
| Lambda | [inv-supplychain-modelinput-notification](https://console.aws.amazon.com/lambda/home?region=us-east-1#/functions/inv-supplychain-modelinput-notification) | Trigger based events (presence of a new file in S3) to initiate ETL jobs |
| ECR | Inv-supplychain-ip21data-addstats | Registered Docker image that executes Python script that adds statistical calculations to IP21 data set prepared by Matillion job |
| inv-supplychain-final-data-prep | Registered Docker image that executes Python script that rescale the statistical calculations to IP21 data set prepared by ‘ Inv-supplychain-ip21data-addstats’ |
| ahlstm-rescaled-all | Docker container image to train prediction model. inv\_supplychain\_model\_scheduler calling this image through HyperParameterTuningJob and get processed in SageMaker. |
|  | inv-supplychain-model-health-check | Registered Docker image that executes Python script that do the prediction as well as health calculation of the model |
| Sagemaker | [maxim-sagemaker-ml-m4-10xlarge](https://console.aws.amazon.com/sagemaker/home?region=us-east-1#/notebook-instances/maxim-sagemaker-ml-m4-10xlarge) | AWS hosted environment for predictive data modeling |

# AWS S3 Folder Structure

|  |  |  |
| --- | --- | --- |
| **Source System** | **S3 Path** | **Description** |
| **IP21** | S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/LiveData/ | IP21 source data is received from Invista to this folder every 4 hours |
| S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/ProcessingData/ | Working folder for Matillion. Source files in Live Folder data is copied to this folder before Matillion load the data to RedShift |
| S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/ArchiveData/ | Files are moved to this location after successful processing in Matillion. |
| S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/ErrorData/ | Files are moved to this location when Matillion encounters error processing the data. |
| S3://inv-nonprod-aa-scm/ip21data/csv/alltags/sapc11/UnprocessedData/ | Files in Processing Data Folder is moved to this location at the beginning of Matillion process |
| **AUDB** | S3://inv-nonprod-aa-scm/audbdata/fullload/csv/SourceData/ | AUDB source data is received to this location. |
| S3://inv-nonprod-aa-scm/audbdata/fullload/csv/Archivedata/ | AUDB source files are moved to this location upon successful processing in Matillion. |
| **SAP** | S3://inv-nonprod-aa-scm/sapdata/fullload/csv/SourceData/ | SAP source data is received to this location. |
| S3://inv-nonprod-aa-scm/sapdata/fullload/csv/Archivedata/ | SAP source files are moved to this location upon successful processing in Matillion. |
| **N/A** | S3://inv-nonprod-aa-scm/ip21data/UnloadfromRedshift/ | Matillion-processed output file is sent to this location before statistical calculations are added (currently with IP21 data only). |
| S3://inv-nonprod-aa-scm/ModelInputFile/ | AWS Batch job adds statistical calculations to the Matillion produced output file and send it to this location. |
| S3://inv-nonprod-aa-scm/ModelOutputFile/ | SageMaker model saves its outputs to this location. |

# Source Code (Version Control)

**Invista KochSource Respository**

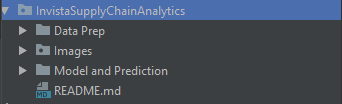
KochSource needs to be set up by Invista to own and manage the source code going forward.

**KBS Koch Source Repository**

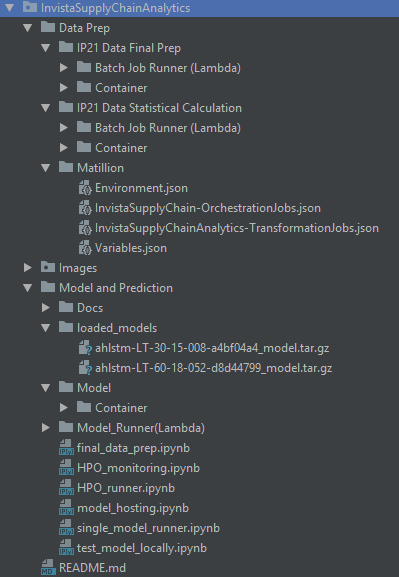
# KBS has hosted the source code under repository named “InvistaSupplyChainAnalytics” and can be found with url mentioned below,

<https://kochsource.io/KBS_Analytics_Solutions/INVISTA/InvistaSupplyChainAnalytics.git>

The high-level structure of the project code repository is as per below snapshot,



The detailed structure of the project code repository is as per below snapshot,



**Notes**

* Matillion Source Code Control:

Matillion has its own built-in source control features called versioning. Use Matillion’s versioning feature to create code snapshots, then use its import/export features to retrieve versioned files from Matillion server as needed. The exported files in json format from Matillion can be stored and version-controlled in KochSource. Then the json files from KochSource can be imported back to Matillion. There is no built-in direct integration in Matillion with source control software such as Git, so Matillion’s export/import feature should be used to interact with Git, and the process can be automated with Matillion REST API.

# Maintenance

**Add new Tags to IP21 data processing**

We need to do some manual steps to have new IP21 tags added to the daily Matilion data processing. Steps are below:

1. Add new tags to public.distincttags with raw/source/staging/metadata\_daily/metadata table name for each tag.
2. Add new tags to public.distincttags\_metadata with min, max range and description.
3. Create raw, source, staging, metadata daily, metadata table for the new tags as defined in step 1.
4. Modify SQL query in ‘Metadata Tables Join Query’ in Matillion ‘JoinTablesByHour’ and ‘JoinTablesByDay’ jobs to include the new tables created for the new tags.
5. Modify SQL query in ‘Unload Transformed Data to S3’ in Matillion ‘IP21-UnloadTransformedDataToS3’ job to include the new tag.

Long term, adding new tags to the daily data processing job can be automated by

1. Create external source of truth repository for holing distinct tag list and its meta data (i.e. S3 file). Grant access to the data owner(s) in Invista to manage the data.
2. Build ETL step to pull the tag control data from user managed source to RedShift control tables.
3. Add step in ETL to create needed raw/source/staging/metadata tables dynamically if they don’t exist.
4. Modify SQL queries in Matillion jobs mentioned in step 4 and 5 to add new tag data for model consumption. These queries can be created dynamically from the distinct tag list using system table info so that the queries don’t have to be manually updated every time new tag is added.

**Remove Tags from IP21 data processing**

1. Remove tags from public.distincttags in RedShift DB.
2. Remove tags from public.distincttags\_metadata table in RedShift DB.
3. Delete raw, source, staging, metadata daily, metadata table for the removed tags.
4. Modify SQL query in ‘Metadata Tables Join Query’ in Matillion ‘JoinTablesByHour’ and ‘JoinTablesByDay’ jobs to remove the tables from the query.

# Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **User** | **Date** | **Change Description** |
| 1.0 - Original | KBS Analytics | 8/7/2018 | Original document shared with Invista for the first knowledge transfer session after phase 1 development |