leaplogic
an IMPETUS product

Solution Brief

Automated transformation of DataStage ETL to cloud-native equivalent or PySpark

Modernizing legacy ETL scripts to modern cloud-native platforms or open collaboration-based languages has become a strategic imperative for enterprises struggling with petabytes of unstructured and fast data from multiple sources. Enterprises are looking to move from DataStage because of the high cost of ownership, limited documentation and cloud integration capabilities, and complex set up process.

However, migrating DataStage ETL to modern data architecture is complex as it is difficult to edit columns and mapping between the stages. The unavailability of source code versioning also makes it difficult to roll back to previous versions in case of any errors, where logs are not handy and difficult to decipher and debug. Businesses, therefore, have several concerns:

- Will there be any business downtime?
- How do I transform years of complex business logic and code?
- Will my workloads be optimized to address the nuances of the new environment?
- How do I prioritize the workloads for transformation?
- How do I ensure seamless operationalization of ETL on the target environment?
- How do I validate the transformed code and migrated data?

Key Benefits

Compared to traditional approaches, LeapLogic enables:

- **4x** faster transformation
- **2x** lower cost
- **4x** developer productivity
- **100%** cloud-ready
- **100%** automation across the migration lifecycle

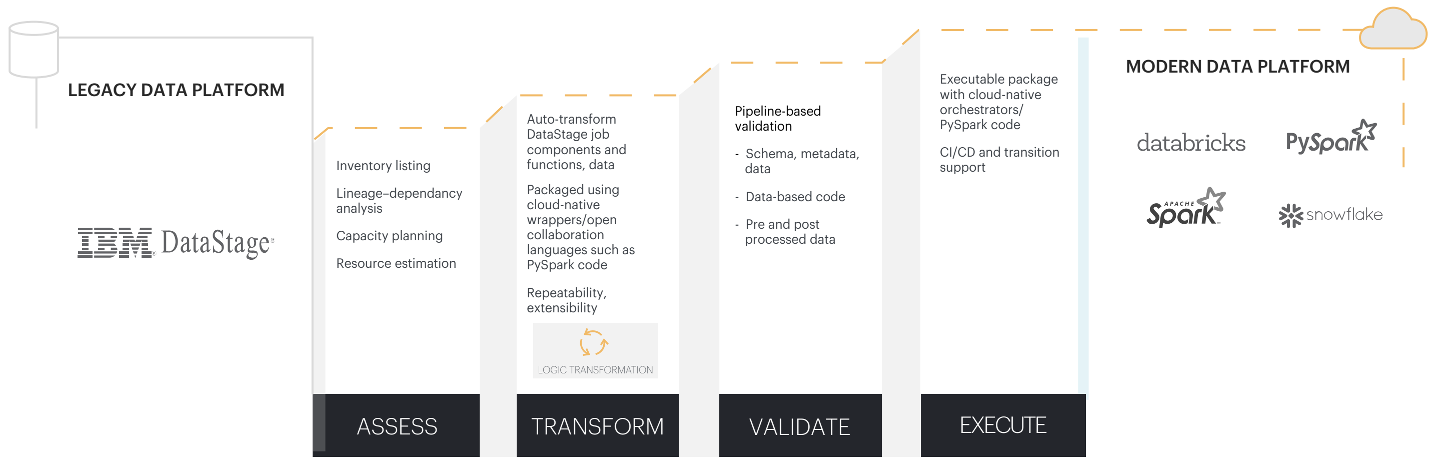
LeapLogic, an Impetus product for automated workload transformation addresses all these concerns. Its intelligent grammar engine identifies optimization opportunities at schema, code, and execution level and automatically converts all types of workloads, logic, and workflows to Snowflake-native equivalent, Databricks-native equivalent, or PySpark.

Key Features

- Intelligent assessment and recommendation for the target architecture and tech stack
- Business logic conversion
- End-to-end transformation to PySpark or a cloud-native equivalent
- End-to-end packaging, orchestration, and execution for the target
- Code optimization and query validation to avoid business disruption
- Automated legacy code translation to PySpark with multiple query engine support
- Cost-performance ratio optimization
- Data governance and security compliance

How it works

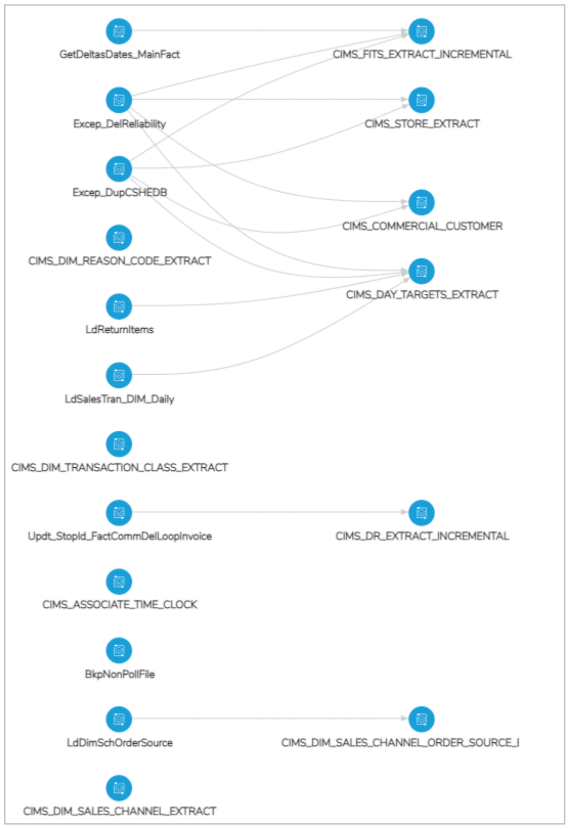
LeapLogic enables end-to-end transformation, operationalization, and transitioning of workloads in four steps:


LEGACY DATA PLATFORM
IBM DataStage
Inventory listing
Lineage-dependancy analysis
Capacity planning
Resource estimation
ASSESS
Auto-transform DataStage job components and functions, data
Packaged using cloud-native wrappers/open collaboration languages such as PySpark code
Repeatability, extensibility
LOGIC TRANSFORMATION
TRANSFORM
Pipeline-based validation
- Schema, metadata, data
- Data-based code
- Pre and post processed data
VALIDATE
Executable package with cloud-native orchestrators/ PySpark code
CI/CD and transition support
EXECUTE
MODERN DATA PLATFORM
databricks
PySpark
Apache Spark
snowflake

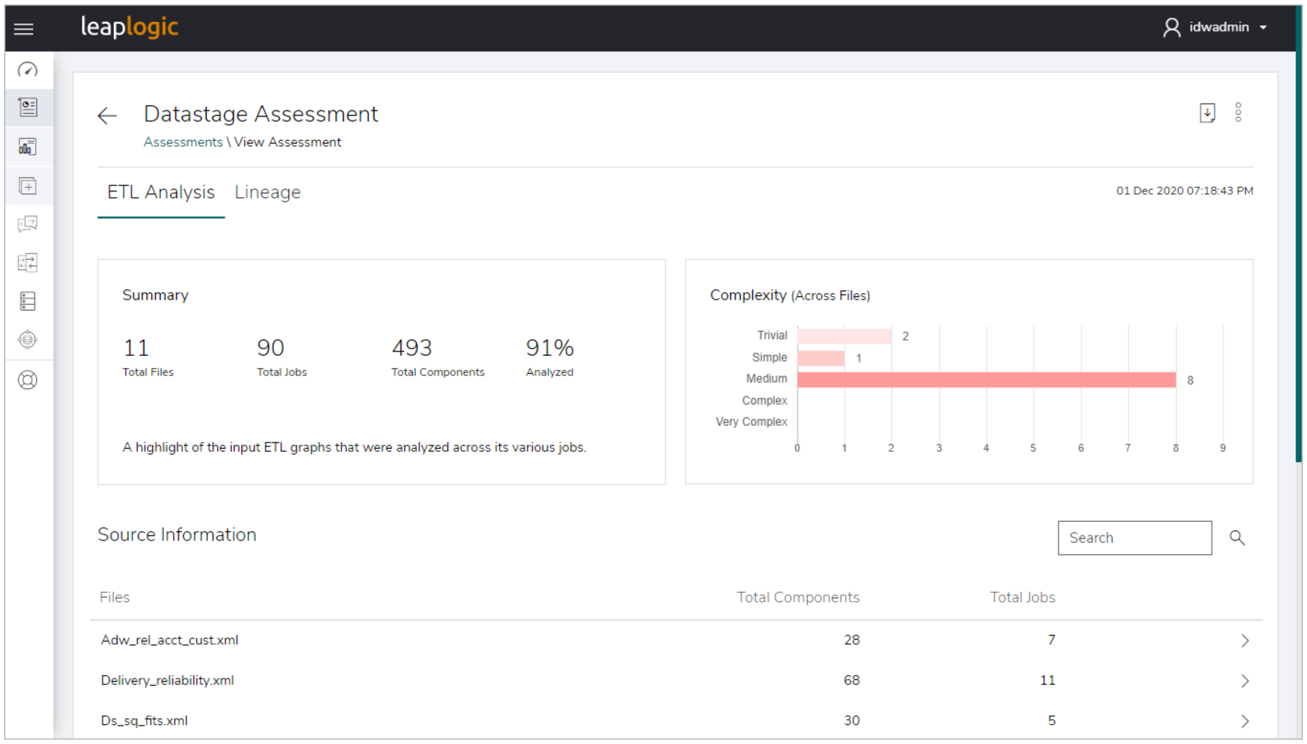
Automated workload transformation

Step 1: Assessment and prescription

- Lists entire inventory for diverse workloads
  - Assesses ETL scripts, DML and DDL scripts, procedures, scheduler/orchestrator scripts (jobs), etc.
  - Identifies total jobs and components per ETL script
  - Identifies job activities, parallel jobs, sequential jobs, aggregator functions, joins, filters, execution command activities, exception handler, etc.
  - Provides actionable insights and prescriptive recommendations with detailed downloadable reports
- Identifies complex interdependencies to group workloads for offload
  - Identifies job-level complexity and its aggregate across files
  - Plots comprehensive, graphical lineage of DataStage XMLs across jobs highlighting the participating input and output tables
  - Provides advanced filters according to workload type and an interactive graphical interface to deep dive into certain flows


GetDeltasDates_MainFact
CIMS_FITS_EXTRACT_INCREMENTAL
Excep_DelReliability
CIMS_STORE_EXTRACT
Excep_DupCSHEDB
CIMS_COMMERCIAL_CUSTOMER
CIMS_DIM_REASON_CODE_EXTRACT
CIMS_DAY_TARGETS_EXTRACT
LdReturnItems
LdSalesTran_DIM_Daily
CIMS_DIM_TRANSACTION_CLASS_EXTRACT
Updt_StopId_FactCommDelLoopInvoice
CIMS_DR_EXTRACT_INCREMENTAL
CIMS_ASSOCIATE_TIME_CLOCK
BkpNonPollFile
LdDimSchOrderSource
CIMS_DIM_SALES_CHANNEL_ORDER_SOURCE_I
CIMS_DIM_SALES_CHANNEL_EXTRACT

DataStage Jobs Assessment



Datastage Assessment

Assessments \ View Assessment

01 Dec 2020 07:18:43 PM

Summary

11	90	493	91%
Total Files	Total Jobs	Total Components	Analyzed

A highlight of the input ETL graphs that were analyzed across its various jobs.

Complexity (Across Files)

Complexity	Count
Trivial	2
Simple	1
Medium	8
Complex	
Very Complex	

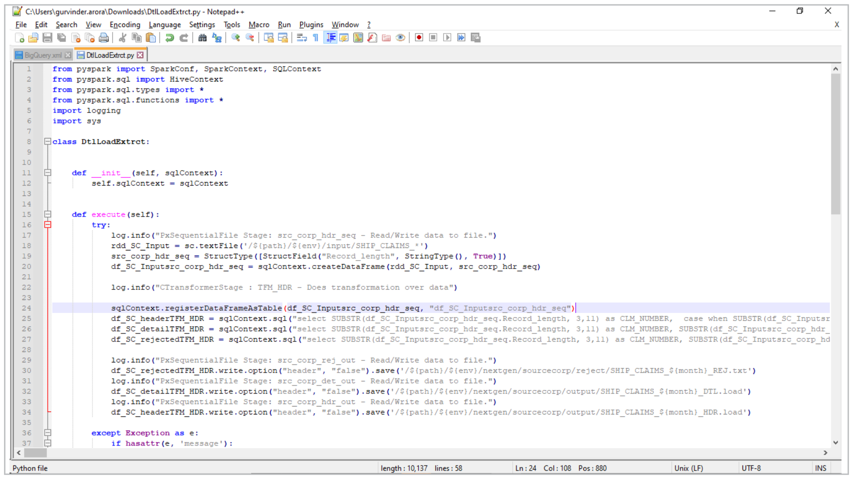
Source Information

Files	Total Components	Total Jobs
Adw_rel_acct_cust.xml	28	7
Delivery_reliability.xml	68	11
Ds_sq_fits.xml	30	5

DataStage Jobs Assessment

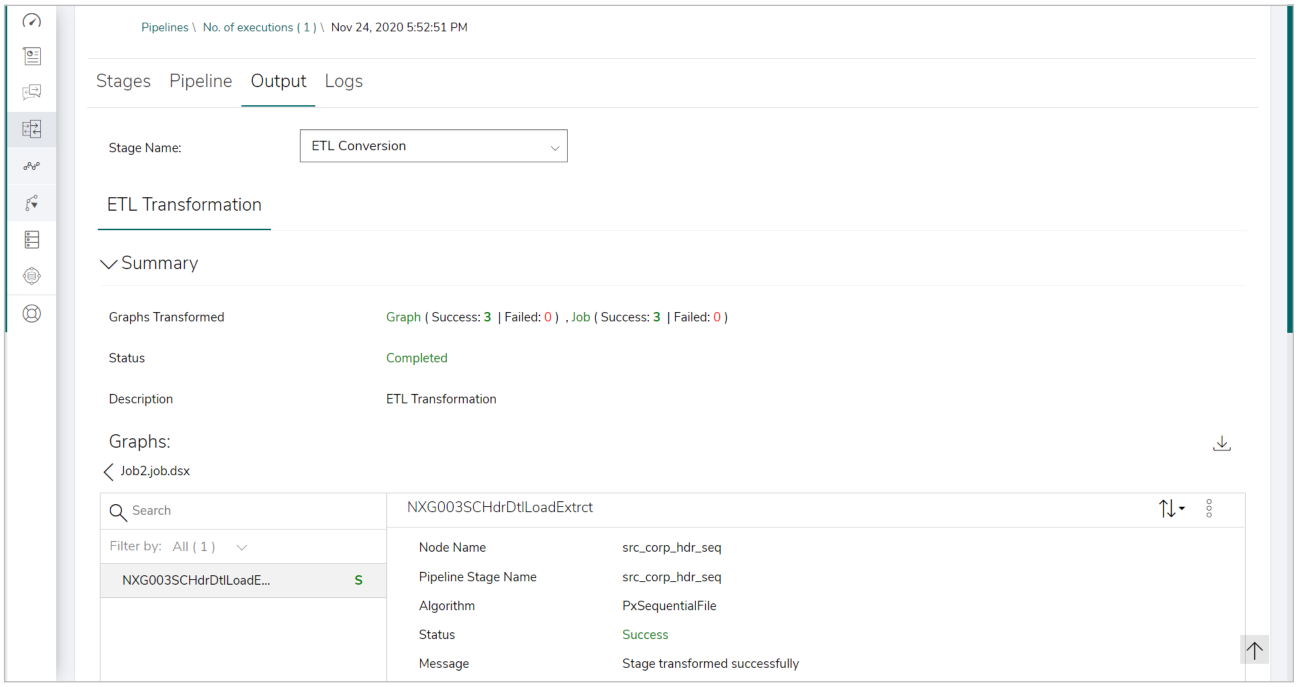
Step 2: Transformation

- Transforms DataStage ETL scripts and migrates schema and dataset to the target store of choice
- Native conversion, packaging, and orchestration
  - Transforms core business logic to cloud-native wrappers or orchestrators
  - Re-packages business logic with scripts to production-ready jobs
  - Generates end-to-end executable package for on staging and production environments after a thorough system integration testing
- Extensible tool and methodology
  - Extensible, repeatable, and verifiable methodology
  - Converts code to a variety of target stores and formats, enabling enterprise-wide usage


C:\Users\gurvinder.arora\Downloads\DtlLoadExtrct.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
BigQuery.xml
DtlLoadExtrct.py

```python
from pyspark import SparkConf, SparkContext, SQLContext
from pyspark.sql import HiveContext
from pyspark.sql.types import *
from pyspark.sql.functions import *
import logging
import sys

class DtlLoadExtrct:


    def __init__(self, sqlContext):
        self.sqlContext = sqlContext


    def execute(self):
        try:
            log.info("PxSequentialFile Stage: src_corp_hdr_seq - Read/Write data to file.")
            rdd_SC_Input = sc.textFile('/${path}/${env}/input/SHIP_CLAIMS_*')
            src_corp_hdr_seq = StructType([StructField("Record_length", StringType(), True)])
            df_SC_Inputsrc_corp_hdr_seq = sqlContext.createDataFrame(rdd_SC_Input, src_corp_hdr_seq)

            log.info("CTransformerStage : TFM_HDR - Does transformation over data")

            sqlContext.registerDataFrameAsTable(df_SC_Inputsrc_corp_hdr_seq, "df_SC_Inputsrc_corp_hdr_seq")
            df_SC_headerTFM_HDR = sqlContext.sql("select SUBSTR(df_SC_Inputsrc_corp_hdr_seq.Record_length, 3,11) as CLM_NUMBER,  case when SUBSTR(df_SC_Inputsr
            df_SC_detailTFM_HDR = sqlContext.sql("select SUBSTR(df_SC_Inputsrc_corp_hdr_seq.Record_length, 3,11) as CLM_NUMBER, SUBSTR(df_SC_Inputsrc_corp_hdr_
            df_SC_rejectedTFM_HDR = sqlContext.sql("select SUBSTR(df_SC_Inputsrc_corp_hdr_seq.Record_length, 3,11) as CLM_NUMBER, SUBSTR(df_SC_Inputsrc_corp_hd

            log.info("PxSequentialFile Stage: src_corp_rej_out - Read/Write data to file.")
            df_SC_rejectedTFM_HDR.write.option("header", "false").save('/${path}/${env}/nextgen/sourcecorp/reject/SHIP_CLAIMS_${month}_REJ.txt')
            log.info("PxSequentialFile Stage: src_corp_det_out - Read/Write data to file.")
            df_SC_detailTFM_HDR.write.option("header", "false").save('/${path}/${env}/nextgen/sourcecorp/output/SHIP_CLAIMS_${month}_DTL.load')
            log.info("PxSequentialFile Stage: src_corp_hdr_out - Read/Write data to file.")
            df_SC_headerTFM_HDR.write.option("header", "false").save('/${path}/${env}/nextgen/sourcecorp/output/SHIP_CLAIMS_${month}_HDR.load')

        except Exception as e:
            if hasattr(e, 'message'):
```

Converted PySpark code



Stage Name: ETL Conversion

Graphs Transformed	Graph (Success: 3 | Failed: 0) , Job (Success: 3 | Failed: 0)
Status	Completed
Description	ETL Transformation

Job2.job.dsx

NXG003SCHdrDtlLoadE... S

NXG003SCHdrDtlLoadExtrct

Node Name	src_corp_hdr_seq
Pipeline Stage Name	src_corp_hdr_seq
Algorithm	PxSequentialFile
Status	Success
Message	Stage transformed successfully

Transformation of DataStage components

Step 3: Validation

- Pipeline-based validation of transformed code with little manual intervention
  - Instantly verifies the transformed jobs and components

Step 4: Operationalization

- Delivers targret-specific executable package
  - Cloud-native orchestration and execution on production
- Supports operationalization
  - Supports end-to-end transitioning into production and operationalization
  - Optimizes target capacity
  - Stabilizes environment through a parallel-run period
  - Smooth cut-over planning
  - Ensures implicit data governance and compliance on the cloud
  - Ensures continuous integration and delivery
  - Helps in enabling powerful operational monitoring on target
  - Provides runbook documentation, training, and handholding

To start your automated transformation from DataStage to cloud-native equivalent or PySpark write to us at: **info@leaplogic.io**

BOOK A DEMO

leaplogic

LeapLogic is a product owned by Impetus Technologies Inc. Impetus is focused on enabling a unified, clear, and present view for the intelligent enterprise by enabling data warehouse modernization, unification of data sources, self-service ETL, advanced analytics, and BI consumption. For more than a decade, Impetus has been the 'Partner of Choice' for several Fortune 500 enterprises in transforming their data and analytics lifecycle. The company brings together a unique mix of software products, consulting services, and technology expertise. Our products include industry's only platform for automated transformation of legacy systems to any modern or cloud-native stack and Gathr – an all-in-one data pipeline platform.

To learn more, visit **www.leaplogic.io** or write to **info@leaplogic.io**

© 2021 Impetus Technologies, Inc. All rights reserved. Product and company names mentioned herein may be trademarks of their respective companies. Sept 2021