regoUniversity

NASHVILLE • 2022

Integrations with Rego Data Processor and Data Extractor

Your Guides:

Dave Matzdorf and Luis Palacios

Agenda

- Introduction
- Understanding Clarity and its Architecture
- Why use Integrations
  - Wheel and Spoke
  - Outbound Integrations
  - Inbound Integrations
- Rego’s Approach
  - Rego’s Data Processor
  - Rego’s Data Extractor
  - Recommendations

Introduction

regoUniversity
NASHVILLE • 2022

Introductions

- Take 5 Minutes
- Turn to a Person Near You
- Introduce Yourself
- Business Cards

Understanding Clarity and its Architecture

Understanding Clarity

- Clarity’s Services:
  - Application Service:
    - Web application, acts as Clarity’s end user facing service to access the UI
    - Exposes Clarity’s XOG Webservice (SOAP based Services)
    - Exposes Clarity’s REST API
  - Process Engine and Job Scheduler:
    - Acts as Clarity’s automation engine
    - Can run custom script (based on GEL/Apache Jelly) to ingest files and connect to other applications
- Connectivity and Capabilities
  - XOG (SOAP Webservice)
  - REST API
  - SFTP (SaaS AWS and SaaS GCP)
  - ODATA

Clarity Web Services: SOAP

- Legacy API
- Simple Object Access Protocol (SOAP)
- Uses only XML for messaging
- Interface is defined in a WSDL (Web Service Description Language) file
- Can be complex to build and parse requests
- Object based

Pros	Cons
Been in the product since the beginning	No new UX support
Easy to work with	XML will have a larger payload
All new custom objects get their own WSDL	Can occasionally differ from core functionality

Clarity Web Services: REST

- Actively being invested on, on latest releases
- Actively used throughout the application and OOTB functionality:
  - New UX Operations
  - Rally Integration
- Provides functionality:
  - Read/write well known objects related to Projects and New UX functionality
  - Multiple authentication mechanisms (basic, token and API Key)
  - Available documentation via “describe” URLs, following industry standards

Pros	Cons
Industry standard	No dot walking support
Easy to work with	Not all tables have public REST API’s
Can call external product REST API’s (have always been able to do this)	The new UX is built on top of the REST API’s ensuring continuity

Clarity Web Services: ODATA

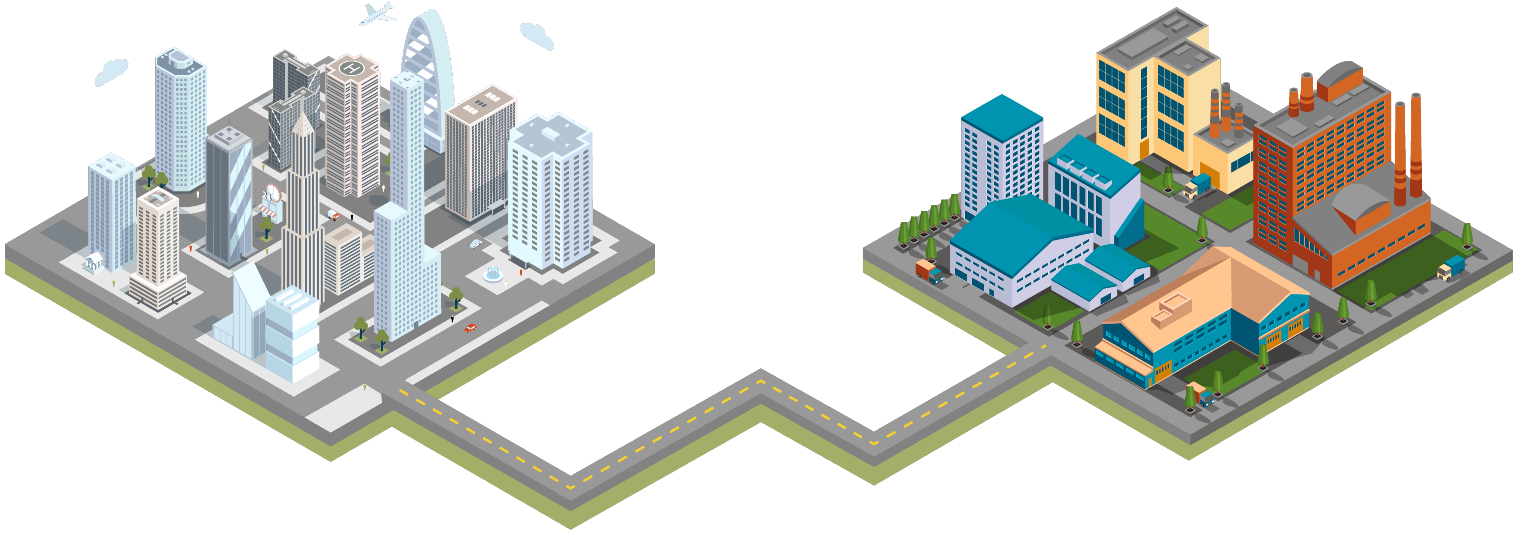
- REST protocol; open standard
- Primarily used in the industry for BI tools
- Used within Clarity for access to the Data Warehouse

Pros	Cons
Provides industry standard access	Only supports the Data Warehouse schema
Most BI tools support ODATA	Not supported for Clarity On Premise
	Hosted by third party software

Why use Integrations?

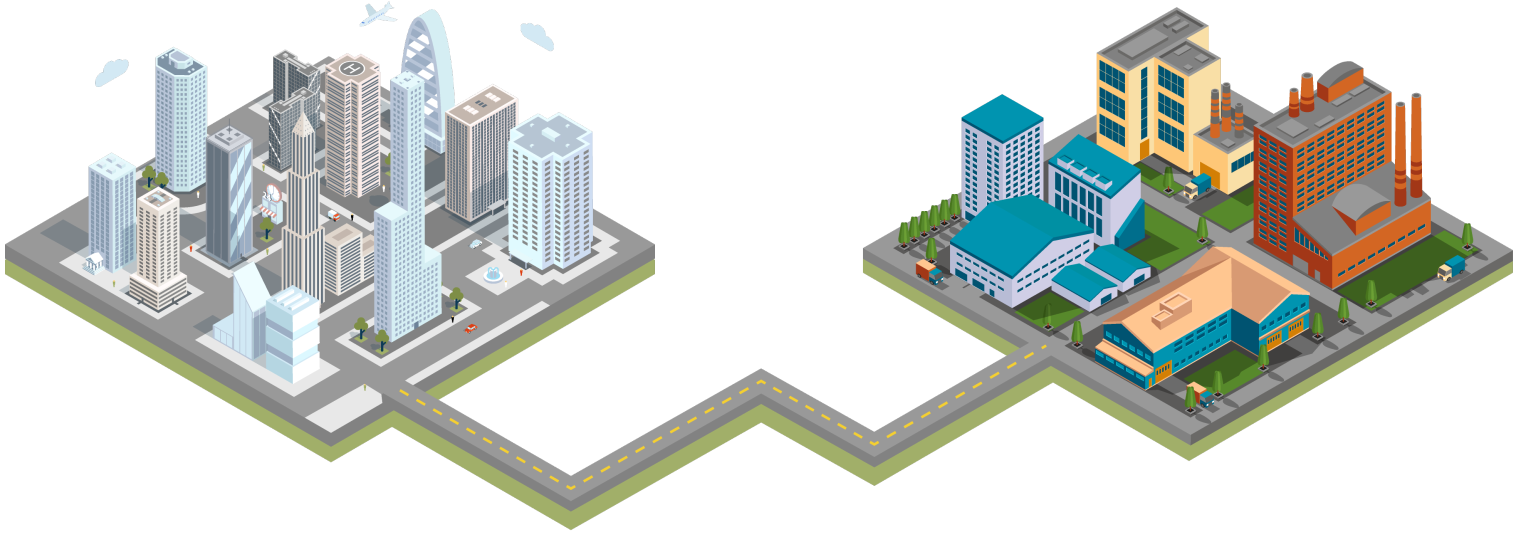
Understanding Integrations

- Organizations have a varied ecosystem, relying on multiple applications with specialized enterprise functionality.
- Organizations are becoming more flexible, allowing teams to work and use their own applications. Increasing the complexity of the ecosystem.
- Data is becoming a valuable asset and deriving insights is key for Organizations to meet their goals.

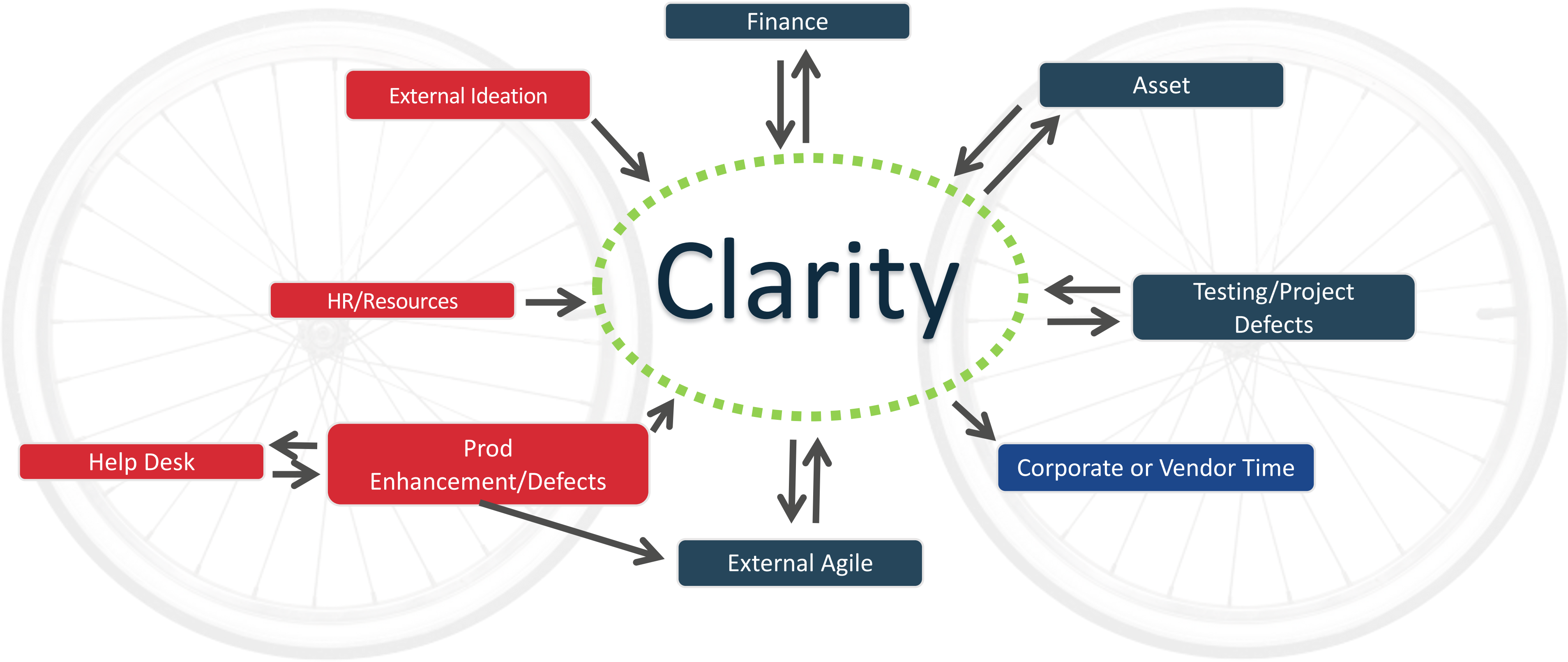


Understanding Integrations

- Integrations are intended to close the gap between different ways of working and applications.
- They provide immediate benefits like:
  - Data quality
  - Avoiding double entry
  - Benefits in performance, cost and time
  - License costs savings
  - Disseminating data across applications



Wheel and Spoke



Common Use Cases

- HR (Workday, Peoplesoft, SAP, Oracle, Active Directory)
  - User provisioning
  - Employee/Resource reporting data
- Financial (SAP, Oracle)
  - Timesheet data
  - Financial transactions
  - Cost plans
- Organizational Data
  - Organizational Structure (OBS)
  - Master data (Lookups, Mapping tables, etc.)
- Project Management
  - Time reporting
  - Project structure (Projects, Teams, Tasks, etc)

Clarity Common Interfaces

Inbound:

- Resource
- Financial Transactions
- Department
- Timesheet
- Project/Idea/Application
- OBS
- Cost Plan
- Rate Matrix
- Resource Calendar

Rego's Approach

regoUniversity
NASHVILLE • 2022

Interface Architecture

- Connect to the Data Source and store the data into a location where it can be manipulated. Common data sources for Clarity integrations:
    - Flat Files
    - Web services
    - Clarity
- Derive and Validate Data
    - Lookup Values/Keys from the target system
    - Apply transformation and validation logic
    - Use mappings to match the target system
- Error Handling and Validation
    - Provide descriptive messages
    - Monitor and log interactions with Webservices and/or target systems
- Load Data
    - Be efficient and avoid unneeded data updates
    - Provide granular results and details

Best Practices and Recommendations

- Avoid over integrating
- Avoid falling into the development shop
- Use reusable and resilient assets and development
- Understand Clarity's capabilities and limitations
- Use Flat Files for large volume batch interfaces
    - When to use
        - Tried and true data exchange method
        - Available for On-Premise and SaaS
        - Batch interfaces
        - Large data volume
        - Secure (During transit and processing)
        - Avoids having to expose your network or applications
    - When not to use
        - Real time interfaces
        - Small, surgical volume

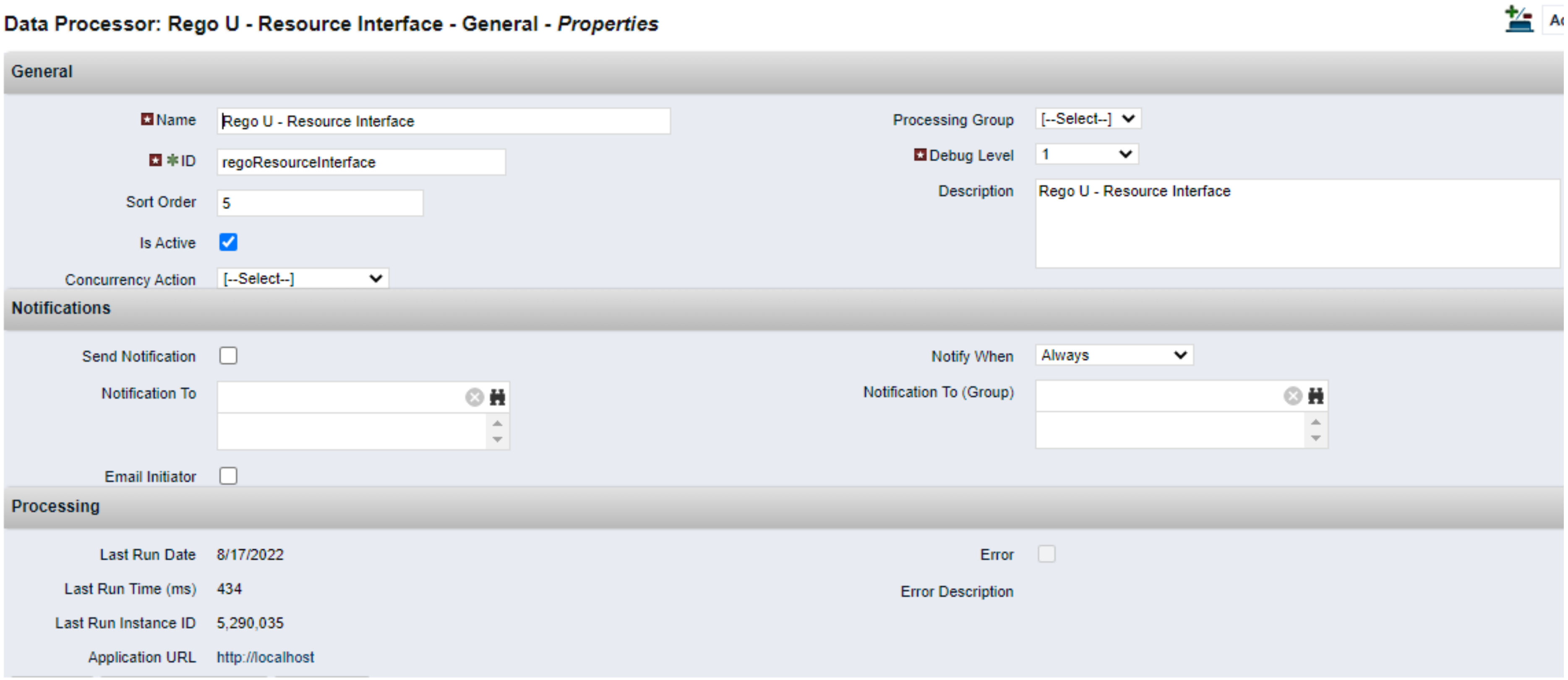
Rego’s Data Processor

Rego’s Data Processor

- Data Processor provides an integration framework using familiar Clarity functionality.
  - Relying on custom objects to store configuration
  - Clarity’s Jobs and Processes to execute
- Supports multiple data sources and targets like:
  - Clarity’s and External Webservices
  - Flat files
  - Clarity to Clarity interfaces
- Ensures a standardized approach to develop interfaces by allowing developers to define a series of actions which:
  - Allow for data to be ingested
  - Data validation and error handling
  - Process and/or load data
  - Provides error handling and granular results
  - Persists data and execution details providing added visibility
- Incorporates industry standards to handle data and access:
  - Data encryption/decryption
  - Retains data within Clarity and its components, avoiding additional exposure
  - Uses a standardized model to store and manage connections and credentials using Rego’s Connection Manager

Data Process Properties

- Data Process Metadata
- Debug Level
- Notifications
- Error Reporting



Data Processor: Rego U - Resource Interface - General - *Properties*

General

Name: Rego U - Resource Interface

ID: regoResourceInterface

Sort Order: 5

Is Active: ☑

Concurrency Action: [--Select--]

Processing Group: [--Select--]

Debug Level: 1

Description: Rego U - Resource Interface

Notifications

Send Notification: □

Notification To:

Email Initiator: □

Notify When: Always

Notification To (Group):

Processing

Last Run Date: 8/17/2022

Last Run Time (ms): 434

Last Run Instance ID: 5,290,035

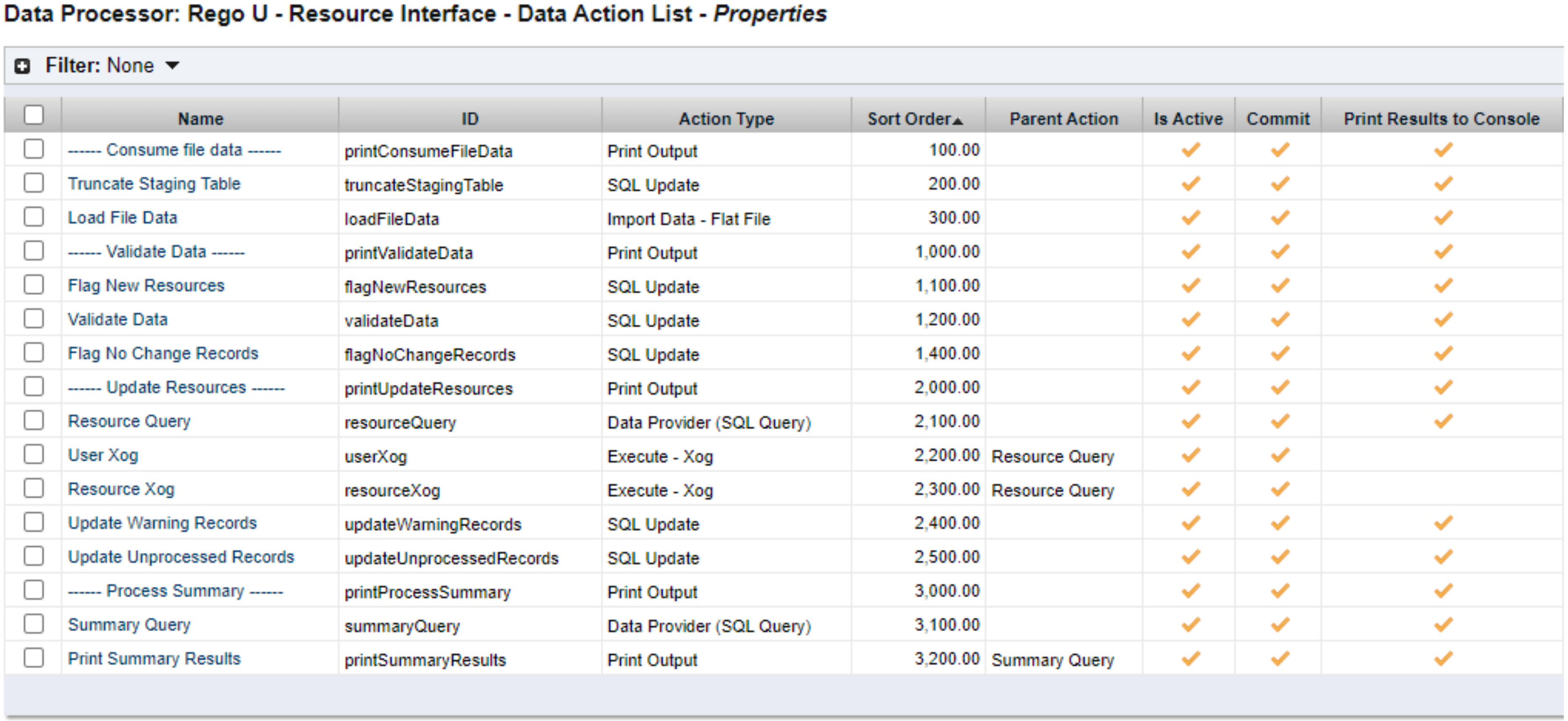
Application URL: http://localhost

Error: □

Error Description:

Data Process Actions

- Action Types
- Sort Order
- Nested Actions



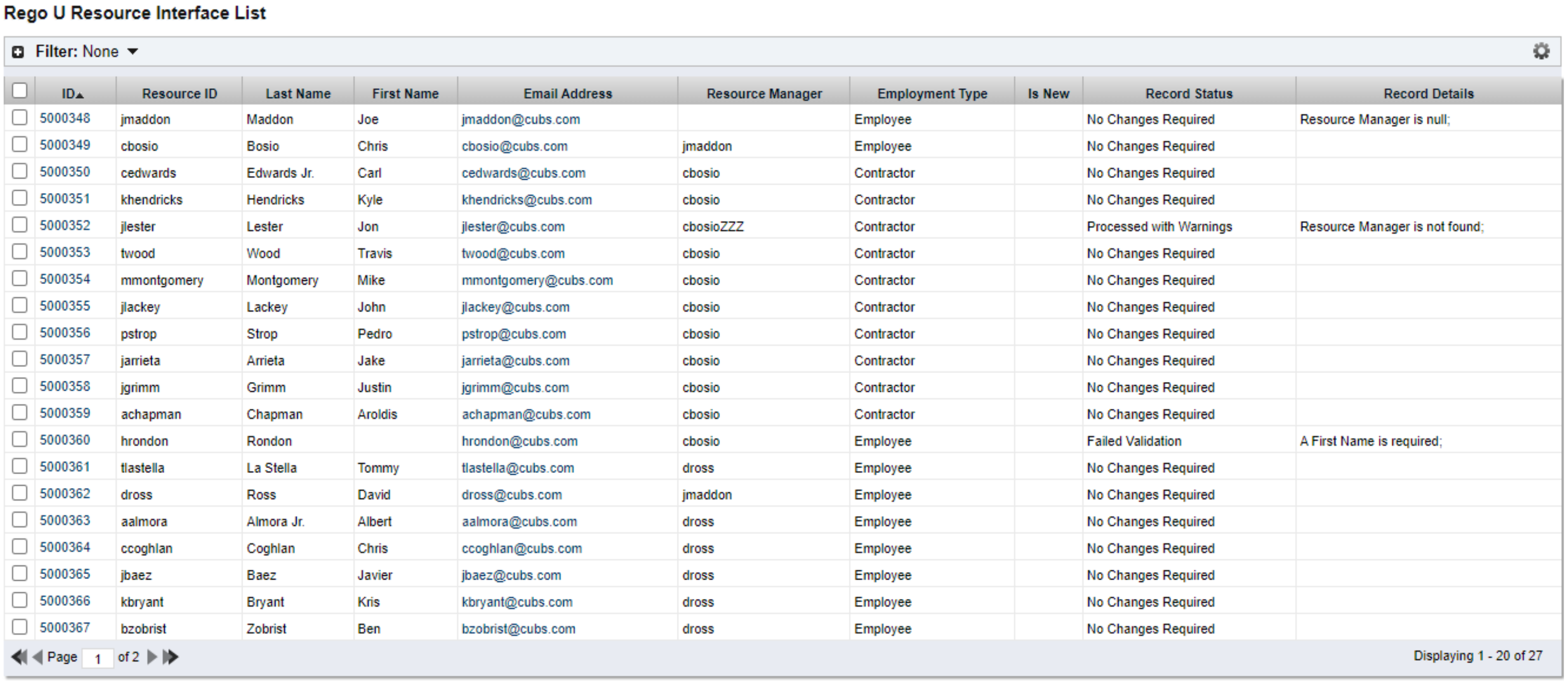
Data Processor: Rego U - Resource Interface - Data Action List - *Properties*

Filter: None

□	Name	ID	Action Type	Sort Order▲	Parent Action	Is Active	Commit	Print Results to Console
□	------ Consume file data ------	printConsumeFileData	Print Output	100.00		✓	✓	✓
□	Truncate Staging Table	truncateStagingTable	SQL Update	200.00		✓	✓	✓
□	Load File Data	loadFileData	Import Data - Flat File	300.00		✓	✓	✓
□	------ Validate Data ------	printValidateData	Print Output	1,000.00		✓	✓	✓
□	Flag New Resources	flagNewResources	SQL Update	1,100.00		✓	✓	✓
□	Validate Data	validateData	SQL Update	1,200.00		✓	✓	✓
□	Flag No Change Records	flagNoChangeRecords	SQL Update	1,400.00		✓	✓	✓
□	------ Update Resources ------	printUpdateResources	Print Output	2,000.00		✓	✓	✓
□	Resource Query	resourceQuery	Data Provider (SQL Query)	2,100.00		✓	✓	✓
□	User Xog	userXog	Execute - Xog	2,200.00	Resource Query	✓	✓	
□	Resource Xog	resourceXog	Execute - Xog	2,300.00	Resource Query	✓	✓	
□	Update Warning Records	updateWarningRecords	SQL Update	2,400.00		✓	✓	✓
□	Update Unprocessed Records	updateUnprocessedRecords	SQL Update	2,500.00		✓	✓	✓
□	------ Process Summary ------	printProcessSummary	Print Output	3,000.00		✓	✓	✓
□	Summary Query	summaryQuery	Data Provider (SQL Query)	3,100.00		✓	✓	✓
□	Print Summary Results	printSummaryResults	Print Output	3,200.00	Summary Query	✓	✓	✓

Data Process Staging Table

- Stage imported data
- Record Status
- Record Details



Rego U Resource Interface List

Filter: None

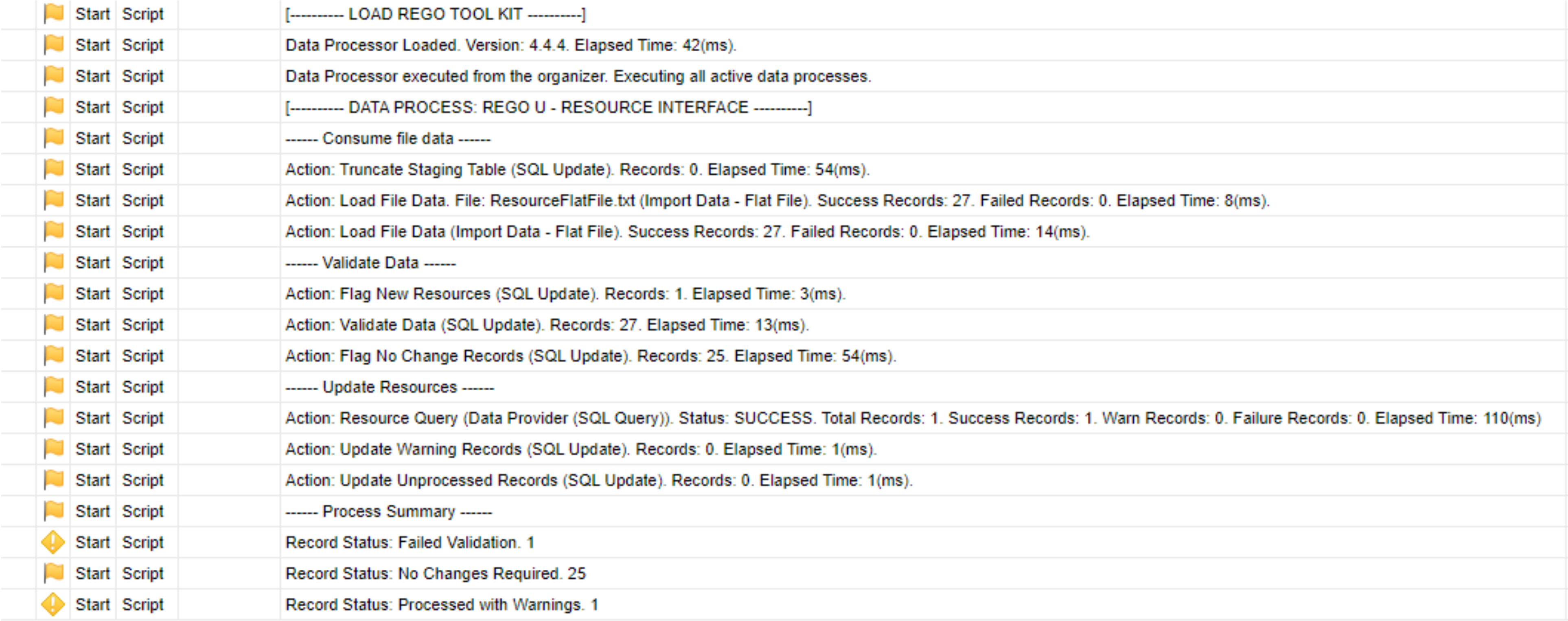
□	ID▲	Resource ID	Last Name	First Name	Email Address	Resource Manager	Employment Type	Is New	Record Status	Record Details
□	5000348	jmaddon	Maddon	Joe	jmaddon@cubs.com		Employee		No Changes Required	Resource Manager is null;
□	5000349	cbosio	Bosio	Chris	cbosio@cubs.com	jmaddon	Employee		No Changes Required	
□	5000350	cedwards	Edwards Jr.	Carl	cedwards@cubs.com	cbosio	Contractor		No Changes Required	
□	5000351	khendricks	Hendricks	Kyle	khendricks@cubs.com	cbosio	Contractor		No Changes Required	
□	5000352	jlester	Lester	Jon	jlester@cubs.com	cbosioZZZ	Contractor		Processed with Warnings	Resource Manager is not found;
□	5000353	twood	Wood	Travis	twood@cubs.com	cbosio	Contractor		No Changes Required	
□	5000354	mmontgomery	Montgomery	Mike	mmontgomery@cubs.com	cbosio	Contractor		No Changes Required	
□	5000355	jlackey	Lackey	John	jlackey@cubs.com	cbosio	Contractor		No Changes Required	
□	5000356	pstrop	Strop	Pedro	pstrop@cubs.com	cbosio	Contractor		No Changes Required	
□	5000357	jarrieta	Arrieta	Jake	jarrieta@cubs.com	cbosio	Contractor		No Changes Required	
□	5000358	jgrimm	Grimm	Justin	jgrimm@cubs.com	cbosio	Contractor		No Changes Required	
□	5000359	achapman	Chapman	Aroldis	achapman@cubs.com	cbosio	Contractor		No Changes Required	
□	5000360	hrondon	Rondon		hrondon@cubs.com	cbosio	Employee		Failed Validation	A First Name is required;
□	5000361	tlastella	La Stella	Tommy	tlastella@cubs.com	dross	Employee		No Changes Required	
□	5000362	dross	Ross	David	dross@cubs.com	jmaddon	Employee		No Changes Required	
□	5000363	aalmora	Almora Jr.	Albert	aalmora@cubs.com	dross	Employee		No Changes Required	
□	5000364	ccoghlan	Coghlan	Chris	ccoghlan@cubs.com	dross	Employee		No Changes Required	
□	5000365	jbaez	Baez	Javier	jbaez@cubs.com	dross	Employee		No Changes Required	
□	5000366	kbryant	Bryant	Kris	kbryant@cubs.com	dross	Employee		No Changes Required	
□	5000367	bzobrist	Zobrist	Ben	bzobrist@cubs.com	dross	Employee		No Changes Required	

Page 1 of 2

Displaying 1 - 20 of 27

Data Process Output

- Records Impacted
- Elapsed Time



```
⚑  Start  Script    [---------- LOAD REGO TOOL KIT ----------]
⚑  Start  Script    Data Processor Loaded. Version: 4.4.4. Elapsed Time: 42(ms).
⚑  Start  Script    Data Processor executed from the organizer. Executing all active data processes.
⚑  Start  Script    [---------- DATA PROCESS: REGO U - RESOURCE INTERFACE ----------]
⚑  Start  Script    ------ Consume file data ------
⚑  Start  Script    Action: Truncate Staging Table (SQL Update). Records: 0. Elapsed Time: 54(ms).
⚑  Start  Script    Action: Load File Data. File: ResourceFlatFile.txt (Import Data - Flat File). Success Records: 27. Failed Records: 0. Elapsed Time: 8(ms).
⚑  Start  Script    Action: Load File Data (Import Data - Flat File). Success Records: 27. Failed Records: 0. Elapsed Time: 14(ms).
⚑  Start  Script    ------ Validate Data ------
⚑  Start  Script    Action: Flag New Resources (SQL Update). Records: 1. Elapsed Time: 3(ms).
⚑  Start  Script    Action: Validate Data (SQL Update). Records: 27. Elapsed Time: 13(ms).
⚑  Start  Script    Action: Flag No Change Records (SQL Update). Records: 25. Elapsed Time: 54(ms).
⚑  Start  Script    ------ Update Resources ------
⚑  Start  Script    Action: Resource Query (Data Provider (SQL Query)). Status: SUCCESS. Total Records: 1. Success Records: 1. Warn Records: 0. Failure Records: 0. Elapsed Time: 110(ms)
⚑  Start  Script    Action: Update Warning Records (SQL Update). Records: 0. Elapsed Time: 1(ms).
⚑  Start  Script    Action: Update Unprocessed Records (SQL Update). Records: 0. Elapsed Time: 1(ms).
⚑  Start  Script    ------ Process Summary ------
⚠  Start  Script    Record Status: Failed Validation. 1
⚑  Start  Script    Record Status: No Changes Required. 25
⚠  Start  Script    Record Status: Processed with Warnings. 1
```

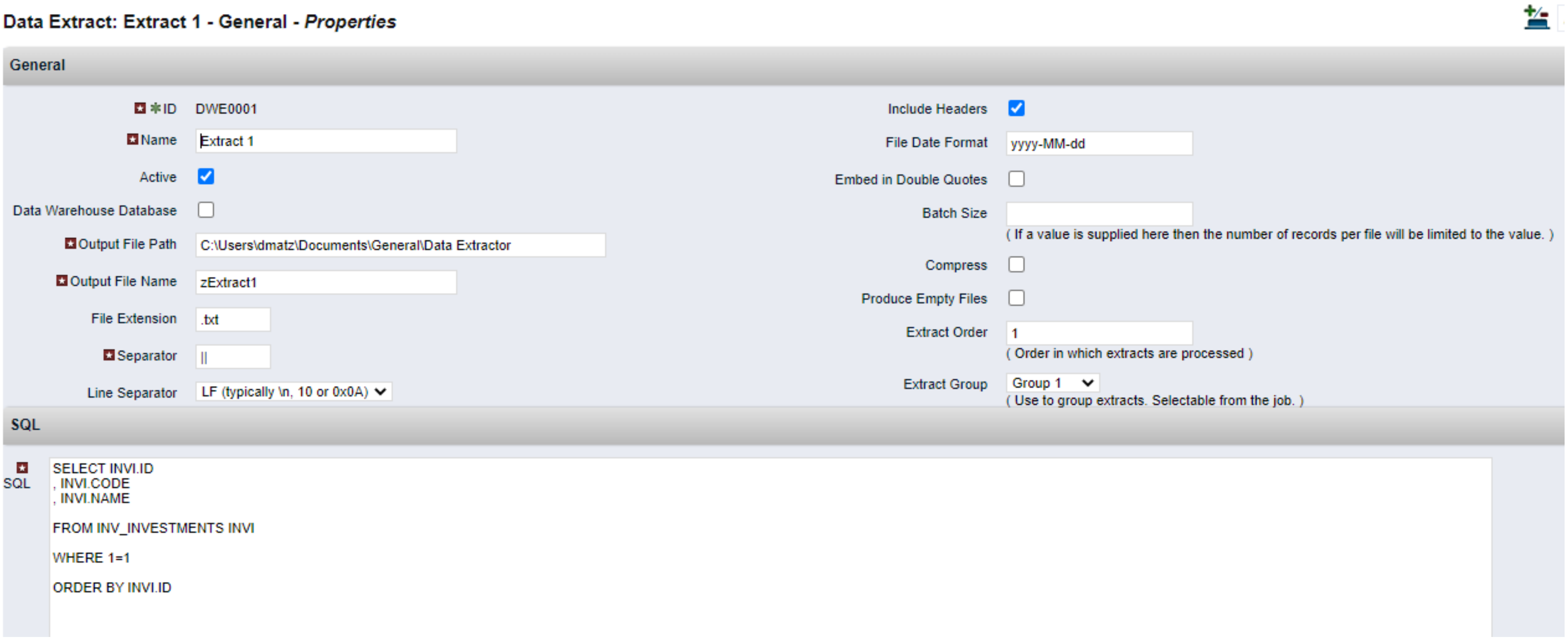
Demo

Rego’s Data Extractor

Rego's Data Extractor

- Data Extractor allows admins to define and run extracts to generate file exports
- Easily configure and schedule through known functionality like:
    - Clarity List Views and Properties to create new configurations
    - Using jobs to easily schedule and generate files
    - Easily access files through Clarity's SFTP, Knowledge Store or email
- The asset is not restricted by known limitations like:
    - Governors, which limit the number of records that can be pulled out
    - Avoids handling the dataset in memory, causing application overhead
    - Stores files within the application, so its capabilities are limited by storage, which normally can be easily incremented
- By avoiding this restrictions, it guarantees performance with little overhead to the application
- Normally used for outbound integrations and/or feed reporting systems, by providing functionality like:
    - Extremely large files can be extracted quickly and efficiently. (Use case: Customer doing a daily DWH dump, over 40 million records across 40 files)
    - Files need to be created synchronously or chained together for the fastest extract time
    - Batching is needed
    - Zipped files are needed
    - Custom data (comma for CSV, tab, etc.) and line separators (line feed, carriage return and CRLF)
    - Dynamic filenames with flexible date formatting
    - Produce Empty Files
    - Produce Files with or without header row
    - Full file encryption capability (PGP)

Data Extract Configuration



Data Extract: Extract 1 - General - *Properties*

General

✱ID	DWE0001	Include Headers	☑
Name	Extract 1	File Date Format	yyyy-MM-dd
Active	☑	Embed in Double Quotes	□
Data Warehouse Database	□	Batch Size	(If a value is supplied here then the number of records per file will be limited to the value.)
Output File Path	C:\Users\dmatz\Documents\General\Data Extractor	Compress	□
Output File Name	zExtract1	Produce Empty Files	□
File Extension	.txt	Extract Order	1 (Order in which extracts are processed)
Separator	||		
Line Separator	LF (typically \n, 10 or 0x0A)	Extract Group	Group 1 (Use to group extracts. Selectable from the job.)

SQL

SQL

```
SELECT INVI.ID
, INVI.CODE
, INVI.NAME

FROM INV_INVESTMENTS INVI

WHERE 1=1

ORDER BY INVI.ID
```

Questions?

Thank You For Attending regoUniversity

Instructions for PMI credits

- Access your account at pmi.org
- Click on **Certifications**
- Click on **Maintain My Certification**
- Click on **Visit CCR's** button under the **Report PDU's**
- Click on **Report PDU's**
- Click on **Course or Training**
- Class Provider = **Rego Consulting**
- Class Name = **regoUniversity**
- Course **Description**
- Date Started = **Today's Date**
- Date Completed = **Today's Date**
- Hours Completed = **1 PDU per hour of class time**
- Training classes = **Technical**
- Click on **I agree** and **Submit**

Let us know how we can improve!
Don't forget to fill out the class survey.

Phone
888.813.0444

Email
info@regoconsulting.com

Website
www.regouniversity.com