**Gen AI Based Data Profiling**

**Title**: Automating FR Y-14Q Data Profiling & Validation using GenAI(Schedule H—Wholesale Risk - H.1 ‐ Corporate Loan Data Schedule)

**Subtitle**: Extracting, Validating & Enhancing Regulatory Data

**Problem Statement:**

* Use LLMs to extract,interpret and refine reporting instructions to identify validation requirements.
* Manual validation is time-consuming and error-prone.
* Objective: Build an AI-powered Data Profiling Solution to streamline data validation and reporting processes by automating the extraction, interpretation, and execution of business rules

**Solution Approach:**

Step 1: Extract FR Y-14Q Text

* Load PDF & extract text using **pdfplumber.**

Step 2: Generate Profiling Rules

* Use **OpenRouter** API + **DeepSeek** Model to generate JSON-based validation rules.

Step 3: Validate Input Data

* Use **pandas** to check constraints (nulls, value ranges, etc.).

Step 4: Highlight Errors & Suggest Fixes

* Save results to **Excel** with invalid data.

Processing Flow:

* Input: FR Y-14Q PDF(Schedule H-Wholesale Risk) + Sample Data CSV
* Processing: Extract Rules → Validate Data → Highlight Errors
* Output: validation\_results.xlsx (Errors highlighted + Fixes)

**Tech Stack:**

Programming Language:

* Python – Used for data processing, validation, and reporting.

Data Handling & Processing:

* pandas – Validate Data Against Extracted Rules.
* LLM – Generate Profiling Rules.
* JSON - Structured profiling rules

Libraries Used:

* pdfplumber – PDF text extraction
* Used LLMs (DeepSeek, OpenRouter) – Extracting profiling rules
* pandas – Data manipulation, analysis & validation
* json – Handling structured output
* requests– HTTP requests
* Sklearn.ensemble - Create multiple models and then combine them to produce improved results.
* re - Regular expression support
* openpyxl - Read ,Write and modify excel files.
* csv - Handling the comma separated files

Environment:

* Google Colab

**Key Validation Rules**

* Customer ID – No carriage return, line feed, comma, or unprintable characters. Unique identifier.
* Internal ID – must be unique.
* Original Internal ID – Must match Internal ID if first submission; no special characters.
* City – Free text, valid city name.
* Country – Must be a 2-letter ISO country code.
* Zip Code – US: 5-digit ZIP (retain leading zeroes), International: valid postal code.
* Industry Code – 4 to 6-digit numeric code (NAICS, SIC, or GICS).
* Industry Code Type – 1 (NAICS), 2 (SIC), or 3 (GICS).
* Obligor Internal Risk Rating – Free text, must align with Schedule H.4.

**Challenges Faced**

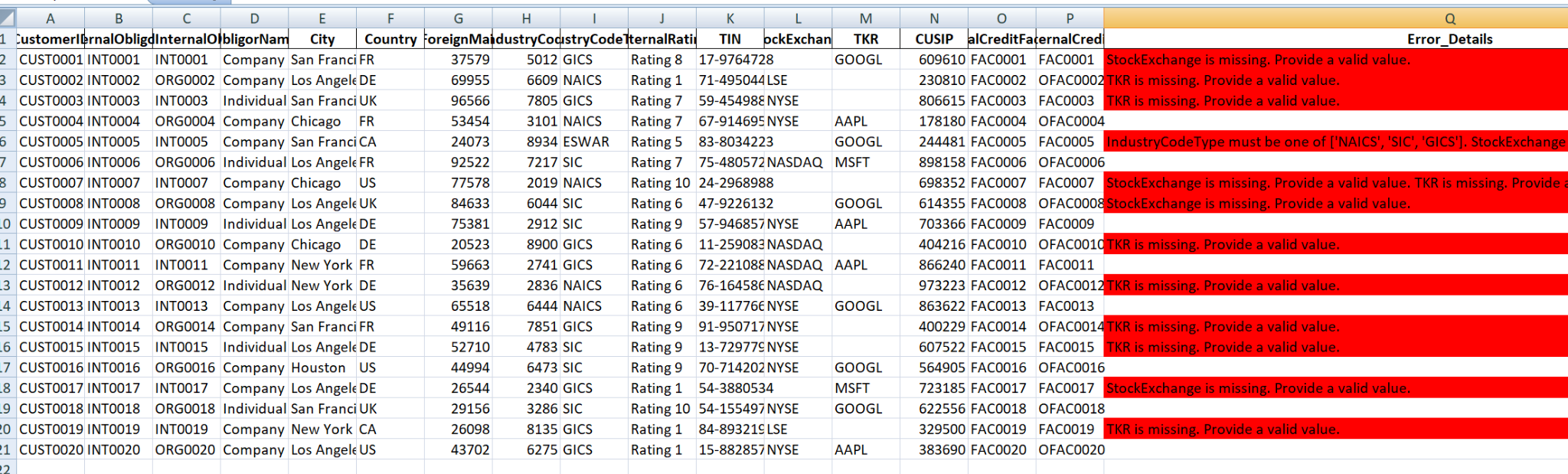
* Lack of Standardized Datasets - Had to create input data and datasets from scratch for testing.
* Performance Optimization – Processing large datasets efficiently.
* Rule Customization – Making validation rules configurable.

**Results & Output**

Generated Reports:

* validation\_results.xlsx – red-highlighted errors & suggested remediation.
* Extracted rules – JSON format structured rules in Output.

**Example Output:**



**Key Benefits:**

**✔** Automates Data Profiling – No manual rule definitions.

✔ Scalable & Explainable – Works across datasets, rule-based.

✔ Regulatory Compliance – Aligns with FR Y-14Q guidelines.

✔ Actionable Insights – Identifies & suggests fixes for errors

**Pseudo Code Explanation:**

Step 1: Extract Data from PDF:

LOAD PDF using pdfplumber

EXTRACT text content from all pages

RETURN extracted text

Step 2: Generate Profiling Rules using LLM

SEND extracted text to OpenRouter AI (DeepSeek Model)

REQUEST structured profiling rules in JSON format

RECEIVE JSON response containing validation rules

ENSURE unique column names in extracted rules

RETURN profiling rules

Step 3: Validate Data Against Extracted Rules

LOAD input dataset (CSV) using pandas

FOR each rule in extracted rules:

IF column exists in dataset:

APPLY validation checks (non-null, allowed values, numeric range)

FLAG errors and append error messages

ELSE:

ADD missing column error message

SAVE results with errors and remediation suggestions

Step 4: Highlight Errors in Excel Output

SAVE validated data to Excel

LOAD Excel workbook using openpyxl

APPLY red highlight to error cells

ADD remediation suggestions for flagged records

SAVE final Excel output with formatting