

Project Name: PDF Parser Automation

Description: Automated pipeline for downloading, parsing, and staging PDF files with configurable workflows.

Author: Eva Iablocova

Repository: https://github.com/Evalablocova/PDF_Parser

Date: 21.07.2025

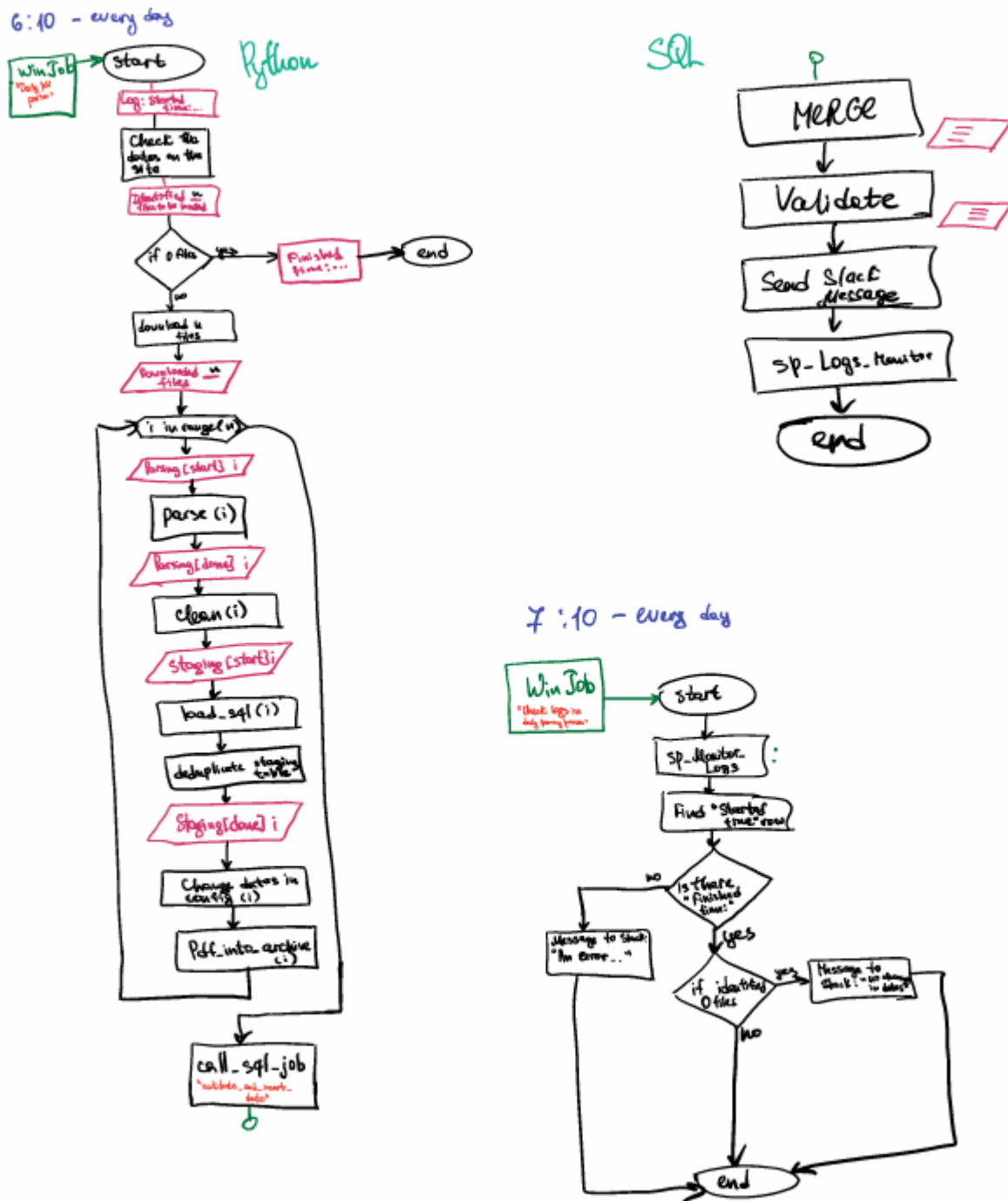
Contents

Project Overview	3
Architecture	4
Installation	6
Usage	7
Project location	7
PDF_parser_sqls – folder with SQL scripts	7
How to run the project	8
Configuration	9
There are 2 configuration files:	9
config.json	9
config_last_dates_in_db.json	10
Scenarios for Using Configuration Files	11
Slack	13
Slack Notification Triggers: Three Scenarios for Sending Messages	13
Adding new Slack recipient	13

Project Overview

This project automates the process of downloading, parsing, and staging PDF files based on date changes and configuration settings. It identifies which files need to be processed, downloads them, parses their contents, loads the results into a database, and manages file archiving. The workflow is controlled by command-line arguments and configuration files, with logging at each step for traceability. The main technologies used are Python and SQL.

Architecture



Pic. 1 - Data Flow

The project is structured as a modular pipeline for processing PDF files:

Dates are loaded and compared.

Changed PDFs are downloaded.

Files are parsed, cleaned, and loaded into the database.

Config is updated and files are archived.

Staging and post-processing are performed (SQL job – merge and validate data, send Slack message)

Installation

1. Clone the repository to your local machine:

```
git clone <repository_url>
```

2. Install Python 3.8 or newer.
3. Install required dependencies:

```
pip install -r requirements
```

4. Configure the project:

Edit config.json to set paths for download_dir, parsed_data_dir, and other settings.

Ensure all required scripts (1_load_dates_from_site.py, 3_1_parser.py, etc.) are present in the project directory.

5. Run the entry point script:

```
python 0_1_entry_point.py [download| parse | stage]
```

Use **download** to start from downloading files.

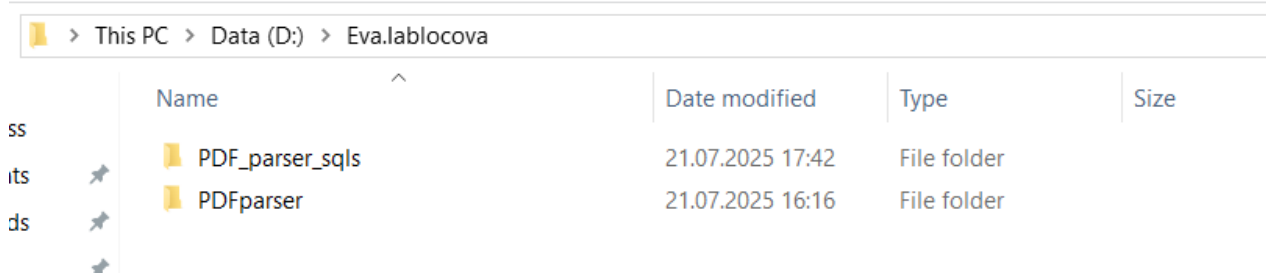
Use **parse** to start from parsing downloaded files.

Use **stage** to stage parsed data.

6. Check logs for progress and errors.

Usage













Project location



This PC > Data (D:) > Eva.lablocova				
	Name	Date modified	Type	Size
ss				
its	PDF_parser_sqls	21.07.2025 17:42	File folder	
ds	PDFparser	21.07.2025 16:16	File folder	

Pic.2 - Project location

PDF_parser_sqls – folder with SQL scripts

	_development - generage DEDUPE script	21.07.2025 11:13	Microsoft SQL Ser...	1 KB
	0_Deployment_sript	21.07.2025 13:55	Microsoft SQL Ser...	26 KB
	1_Merge	17.07.2025 15:38	Microsoft SQL Ser...	5 KB
	2_Validation	17.07.2025 15:38	Microsoft SQL Ser...	34 KB
	3_agregate_mesage	21.07.2025 11:46	Microsoft SQL Ser...	9 KB
	4_call_sp_to_check_logs	18.07.2025 9:32	Microsoft SQL Ser...	1 KB
	add_present_values	21.07.2025 17:42	Microsoft SQL Ser...	1 KB
	create_slack_config	17.07.2025 15:40	Microsoft SQL Ser...	2 KB
	create_sp_slack	17.07.2025 15:41	Microsoft SQL Ser...	3 KB
	DEDUPE stored procedure	21.07.2025 11:13	Microsoft SQL Ser...	5 KB
	run_4_sql_monitor_logs	18.07.2025 12:35	Windows Batch File	1 KB
	sp_Monitor_Logs	18.07.2025 9:25	Microsoft SQL Ser...	2 KB

Pic.3 - PDF_parser_sqls

0_Deployment_script.sql – one run time script for creating the structure and constraints

1_Merge.sql – the first step in SQL job

2_Validation.sql – the second step in SQL job

3_agregate_message.sql – the third step in SQL job

4_call_sp_to_check_logs.sql - the fourth step in SQL job

How to run the project

1 variant:

1. Choose the operation mode: download, parse, or stage (by default – download)
2. Run the entry point script with the desired mode:

```
python 0_1_entry_point.py [download|parse|stage]
```

```
D:\Eva.Iablokova\PDFparser>python 0_1_entry_point.py
Starting from step: download
Dates have not changed.
Files to process: []
No files to process.
```

download: Downloads new or changed PDF files based on date comparison.

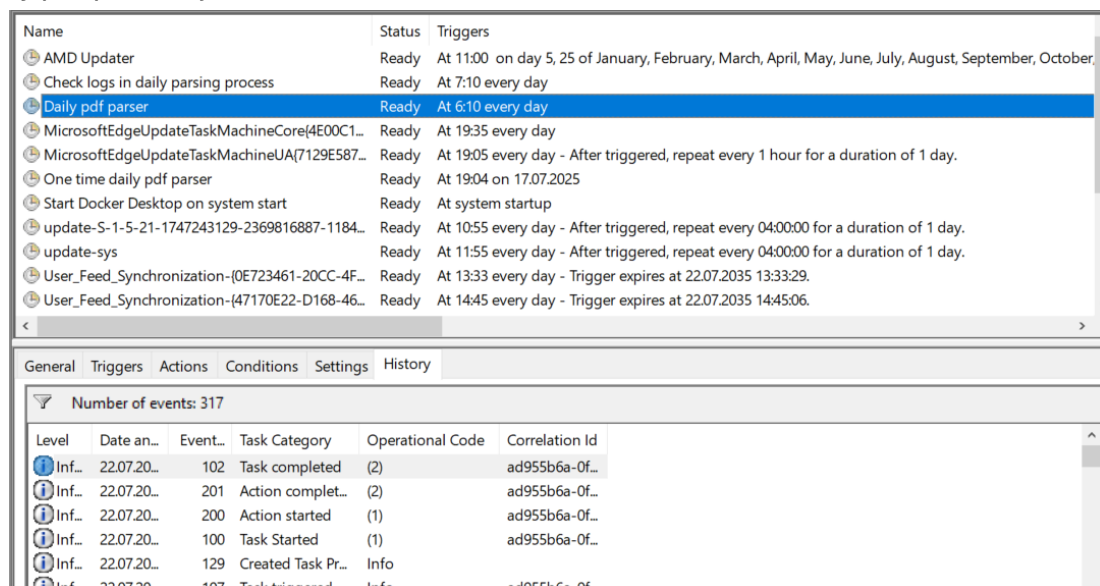
parse: Parses downloaded files (all files in “files_to_parse” folder), cleans data, loads results into the database, and archives processed files.

stage: Stages parsed data (all files in “parsed_files” folder) for further processing and updates configuration.

3. Monitor progress and errors in the log messages generated during each step.
4. Check the output directories (download_dir, parsed_data_dir) for processed files.

2 variant:

1. Go to the scheduler
2. Find “Daily pdf parser” job and run it



Pic. 4 – Windows job

Configuration

There are 2 configuration files:

his PC > Data (D:) > Eva.lablocova > PDFparser

Name	Date modified	Type	Size
.idea	20.07.2025 9:07	File folder	
__pycache__	21.07.2025 11:29	File folder	
archive	21.07.2025 18:31	File folder	
files_to_parse	21.07.2025 18:31	File folder	
parsed_files	21.07.2025 18:31	File folder	
0_1_entry_point	21.07.2025 11:22	Python File	10 KB
0_2_date	21.07.2025 10:27	Python File	3 KB
0_3_write_to_log	18.07.2025 20:17	Python File	2 KB
1_load_dates_from_site	18.07.2025 20:17	Python File	2 KB
2_download_changed_pdf	19.07.2025 15:26	Python File	3 KB
3_1_parser	19.07.2025 17:28	Python File	5 KB
3_2_no_pattern_parser	18.07.2025 20:17	Python File	7 KB
4_clean	18.07.2025 20:17	Python File	5 KB
5_load_sql	21.07.2025 14:23	Python File	4 KB
6_change_dates_in_config	19.07.2025 17:52	Python File	1 KB
7_pdf_into_archive	19.07.2025 18:00	Python File	1 KB
8_call_job	18.07.2025 20:17	Python File	2 KB
cleaned_combined_tables.csv	21.07.2025 18:31	CSV File	26 KB
config	18.07.2025 20:17	JSON File	8 KB
config_last_dates_in_db	21.07.2025 18:31	JSON File	3 KB
debug_no_pattern.csv	21.07.2025 18:31	CSV File	26 KB
processed_combined_tables.csv	21.07.2025 18:31	CSV File	25 KB
today_dates	22.07.2025 6:10	JSON File	3 KB

Pic.5 – Configuration files

config.json

config.json – contains all settings parameters for parsing files

Edit the config.json file to set the following parameters:

download_dir: Directory for downloaded PDF files.

parsed_data_dir: Directory for parsed data files.

config_last_dates_in_db: Stores the last processed dates for each file.

today_file: Path to the file containing today's dates.

file_configs: List of dictionaries specifying keywords and parsing settings for each file type.

```
"file_configs": [  
  {  
    "keyword": "Denumirea",  
    "count_columns": 7,  
    "sizes": [  
      0,  
      55,  
      100,  
      170,  
      313,  
      470,  
      640,  
      820  
    ],  
    "search_pattern": "MD-\\d{4}",  
    "address_column_number": [  
      4  
    ],  
    "equal_columns_numbers": [  
      3,  
      6  
    ],  
    "idno_column_number": 2,  
    "date_column_number": [  
      1  
    ],  
    "estimated_rows_count": 238,  
    "need_cleaning_columns": [  
      3,  
      6  
    ],  
    "headers": [  

```

Pic. 6 – part of config.json file

config_last_dates_in_db.json

config_last_dates_in_db.json – contains last updated dates for each file.

```

{
  "FileName": "Sediul_2008_2024.pdf",
  "start_date": "2008-06-01",
  "end_date": "2024-12-31"
},
{
  "FileName": "Inactive.pdf",
  "start_date": "2008-06-01",
  "end_date": "2025-07-18"
},
{
  "FileName": "Lichidarea.pdf",
  "start_date": "2025-01-01",
  "end_date": "2025-07-18"
},
{
  "FileName": "Lichidarea_2008_2024.pdf",
  "start_date": "2008-06-01",
  "end_date": "2024-12-31"
},
{
  "FileName": "Lichidarea_term_exp.pdf",
  "start_date": "2025-01-01",
  "end_date": "2025-07-18"
},
{
  "FileName": "Lichidarea_term_exp_2018_2024.pdf",
  "start_date": "2018-01-01",
  "end_date": "2024-12-31"
},
},

```

Pic. 7 – part of config_last_dates_in_db.json file

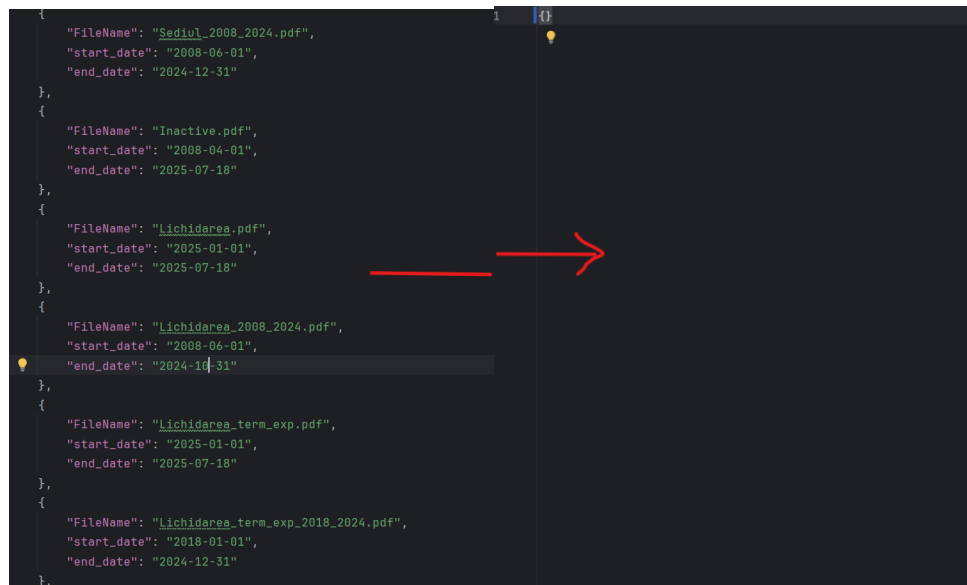
Scenarios for Using Configuration Files

- If we need to download 1 or several files, we have to change either “start_time” or “end_time” for these files and run the project (or Windows job “Daily pdf parser”).

Before	After
<pre> { "FileName": "Sediul_2008_2024.pdf", "start_date": "2008-06-01", "end_date": "2024-12-31" }, { "FileName": "Inactive.pdf", "start_date": "2008-06-01", "end_date": "2025-07-18" }, { "FileName": "Lichidarea.pdf", "start_date": "2025-01-01", "end_date": "2025-07-18" }, { "FileName": "Lichidarea_2008_2024.pdf", "start_date": "2008-06-01", "end_date": "2024-12-31" }, { "FileName": "Lichidarea_term_exp.pdf", "start_date": "2025-01-01", "end_date": "2025-07-18" }, { "FileName": "Lichidarea_term_exp_2018_2024.pdf", "start_date": "2018-01-01", "end_date": "2024-12-31" }, }, </pre>	<pre> { "FileName": "Sediul_2008_2024.pdf", "start_date": "2008-06-01", "end_date": "2024-12-31" }, { "FileName": "Inactive.pdf", "start_date": "2008-06-01", "end_date": "2025-07-18" }, { "FileName": "Lichidarea.pdf", "start_date": "2025-01-01", "end_date": "2025-07-18" }, { "FileName": "Lichidarea_2008_2024.pdf", "start_date": "2008-06-01", "end_date": "2024-10-31" }, { "FileName": "Lichidarea_term_exp.pdf", "start_date": "2025-01-01", "end_date": "2025-07-18" }, { "FileName": "Lichidarea_term_exp_2018_2024.pdf", "start_date": "2018-01-01", "end_date": "2024-12-31" }, }, </pre>

Pic. 8 – Before and after changings for a few files

- If we need to make full run, we just can insert into “config_last_dates_in_db.json” instead of all information “{}”.



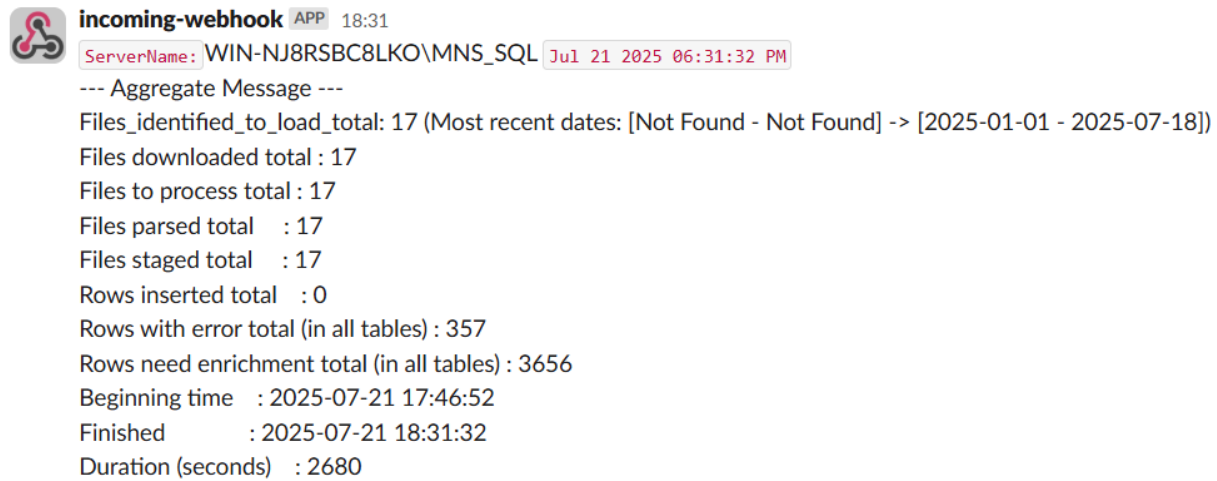
```
{
  "FileName": "Sediul_2008_2024.pdf",
  "start_date": "2008-06-01",
  "end_date": "2024-12-31"
},
{
  "FileName": "Inactive.pdf",
  "start_date": "2008-06-01",
  "end_date": "2025-07-18"
},
{
  "FileName": "Lichidarea.pdf",
  "start_date": "2025-01-01",
  "end_date": "2025-07-18"
},
{
  "FileName": "Lichidarea_2008_2024.pdf",
  "start_date": "2008-06-01",
  "end_date": "2024-12-31"
},
{
  "FileName": "Lichidarea_term_exp.pdf",
  "start_date": "2025-01-01",
  "end_date": "2025-07-18"
},
{
  "FileName": "Lichidarea_term_exp_2018_2024.pdf",
  "start_date": "2018-01-01",
  "end_date": "2024-12-31"
}
}
```

Pic. 9 – Before and after changings for all files

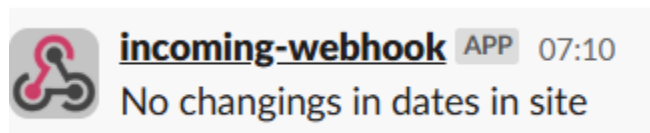
Slack

Slack Notification Triggers: Three Scenarios for Sending Messages

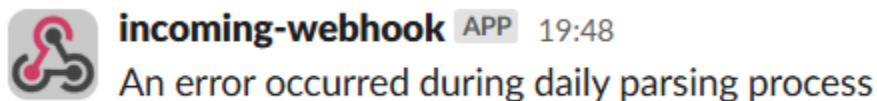
- 1) When task completed successfully and files were downloaded



- 2) When Task completed successfully and files were not downloaded, because dates were not changed on the site



- 3) When Task completed with error



Adding new Slack recipient

We have table for Slack configuration settings

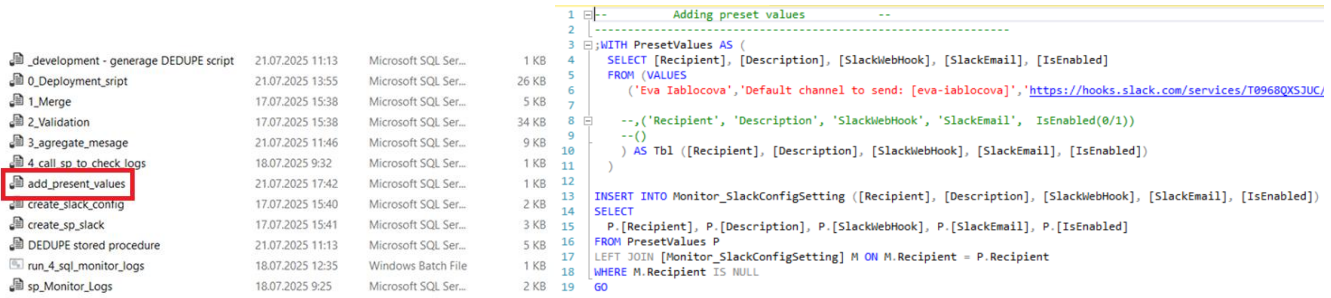
```
2 SELECT TOP (1000) [SlackConfigSetting_SID]
3     , [Recipient]
4     , [Description]
5     , [SlackWebHook]
6     , [SlackEmail]
7     , [DateCreated]
8     , [isEnabled]
9 FROM [PDFparser].[dbo].[Monitor_SlackConfigSetting]
```

SlackConfigSetting_SID	Recipient	Description	SlackWebHook	SlackEmail	DateCreated	isEnabled
1	Eva Iablova	Default channel to send: [eva-iablova]	https://hooks.slack.com/services/T0968QXSJUC/B09...	eva.iablova@gmail.com	2025-07-21 17:41:31.953	1

Pic. 10 – SlackConfigSetting_SID table

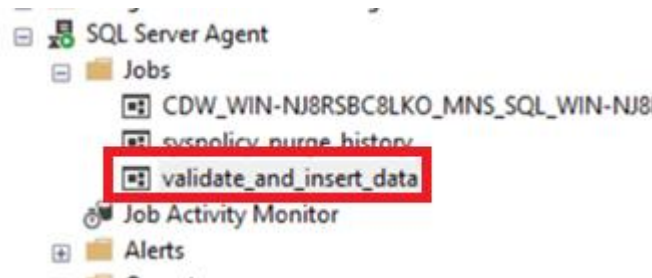
If we need to add new Slack recipient:

- 1) We need to execute this procedure with valid parameters

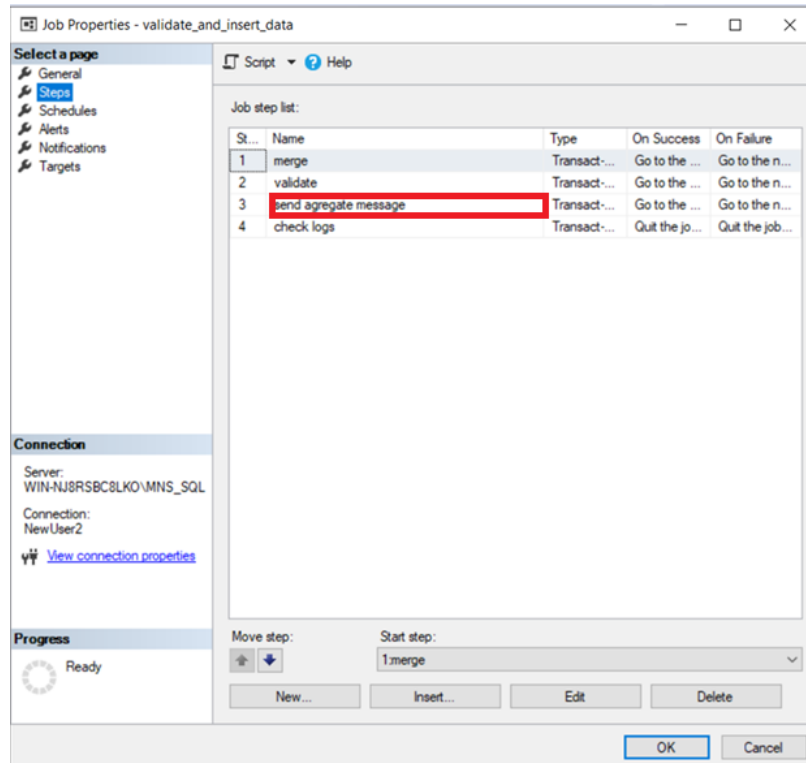


Pic. 11 – add_present_values for Slack messages recipient

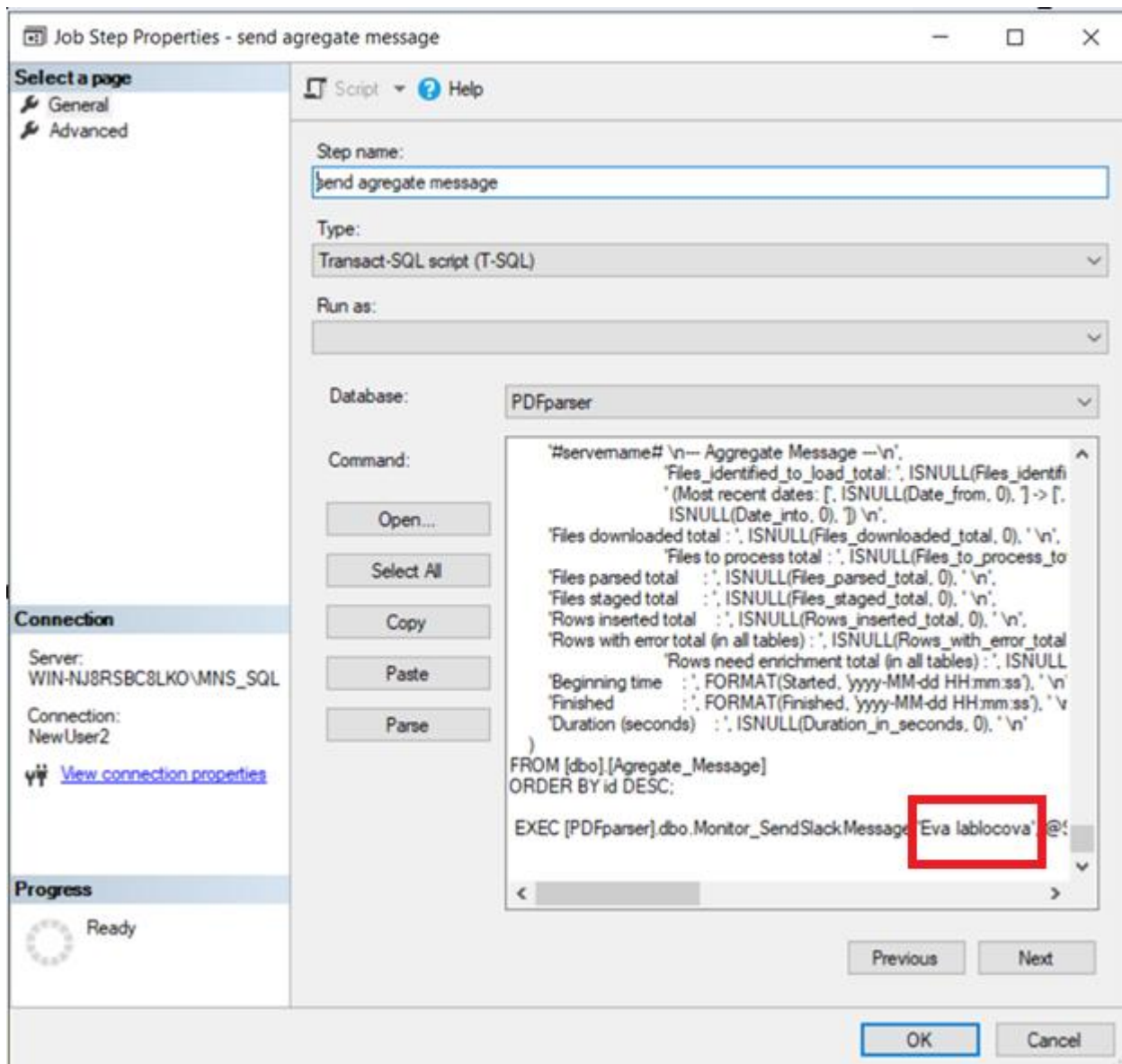
- 2) And change the “Recipient” in SQL job “validate_and_insert_data”



Pic. 12 – Open “validate_and_insert_data” SQL job



Pic. 13 – Open “send_aggregate_message” step



Pic. 14 – Change the “Recipient” at the end of script