**Инструкция по использованию парсера файлов «Fusion Power Predator Parser v.1.0»**

2022

Содержание

[**1.** **Подготовка** 3](#_Toc94109321)

[**3.** **Запуск** 4](#_Toc94109322)

[**4.** **Начало парсинга** 5](#_Toc94109323)

[**5.** **Парсинг CSV и TXT файлов** 6](#_Toc94109324)

[**6.** **Парсинг JSON файлов** 8](#_Toc94109325)

[**7.** **Скрипт проверки** 9](#_Toc94109326)

[**8.** **Загрузка данных на сервер** 10](#_Toc94109327)

# **Подготовка**

* 1. Для использования парсера необходимо установить на ОС:
* Python 3.9 и выше -> https://www.python.org/downloads/
* Emeditor -> <https://www.emeditor.com/download/>
* Notepad++ -> <https://notepad-plus-plus.org/downloads/>
* Исходный код парсера(запросить у Тимлида)
  1. После установки требуемого ПО, необходимо в терминале перейти в папку с исходным кодом парсера.

$ **cd work\parser\_dir** (путь до парсера)

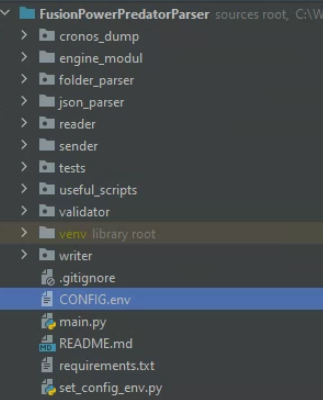
* 1. Создать папку виртуального окружения.

$ **py -m venv venv**

* 1. Установить зависимости парсера

$ **pip install -r requirements.txt**

1. **Создание файла персональных параметров для парсера**
   1. Для работы парсера необходимо заполнить файл с переменными окружения *CONFIG.env*.



В нём должны быть следующие переменные:

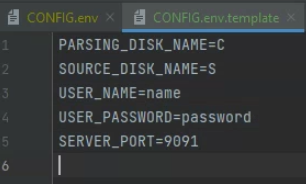
***PARSING\_DISK\_NAME=C*** – буква диска на котором будет выполняться обработка файлов

***SOURCE\_DISK\_NAME=S*** - буква диска на котором находятся исходные данные

***USER\_NAME=name*** – имя пользователя в Сlickhouse

***USER\_PASSWORD=password*** – пароль пользователя в Сlickhouse

***SERVER\_PORT=9091*** – порт подключения Сэндера



# **Запуск**

* 1. Перед запуском парсера необходимо активировать виртуальное окружение python. Для этого в терминале перейдите в каталог парсера и выполните команду.

$ **venv\Scripts\activate**

* 1. Запуск парсера ручном режиме выполняется командой.

(venv)$ **py main.py**

Для запуска парсера в полуавтоматическом режиме необходимо указать аргумент --auto-parse. В этом режиме парсер будет автоматически обрабатывать файлы подходящие под шаблоны автопарсинга.

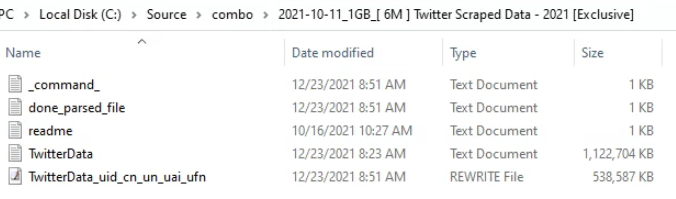
(venv)$ **py main.py --auto-parse**

Для запуска в полностью автоматического режима, необходимо добавить аргумент --full-auto. В этом режиме парсер обрабатывает только файлы подходящие под шаблоны автопарсинга. Все файлы не подходящие под шаблоны, парсер пропускает.

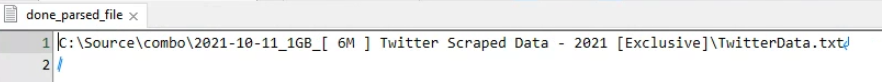
(venv)$ **py main.py --auto-parse --full-auto**

# **Начало парсинга**

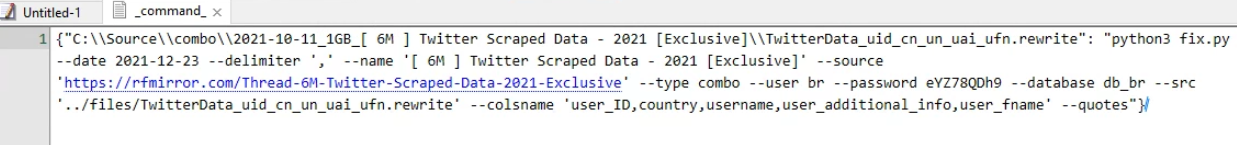
* 1. При парсинге файлов происходит их обработка, валидация и сохранение в формате для csv с расширением *.rewrite* в том же каталоге, что и исходный файл.



Парсер сохраняет список обработанных файлов в файле ***done\_parsed\_file.txt***, который создается внутри каждой обрабатываемой базы.



В файл \_command\_.txt сохраняются JSON-ы, с указанием пути до конвертированного файла и команды для загрузки в Clickhouse.



* 1. В корне папки db/combo создаются файлы для записи списка обработанных баз. В файле ***\_dirs\_complete\_.txt*** сохраняются базы, обработка которых прошла успешно. В ***\_dirs\_passed\_.txt*** сохраняются базы обработка, которых была пропущена.
  2. После запуска, парсер предлагает выбрать тип баз для обработки.



* 1. Парсер сохраняет список отработанных файлов, если необходимо заново распарсить обработанные файлы, то это надо указать в запросе парсера.



# **Парсинг CSV и TXT файлов**

* 1. После запуска и обнаружения парсером файлов с расширениями .txt, .csv на экран выводится:

- Количество фалов в базе

- Сколько всего баз осталось обработать

- Сколько файлов осталось обработать в данной базе

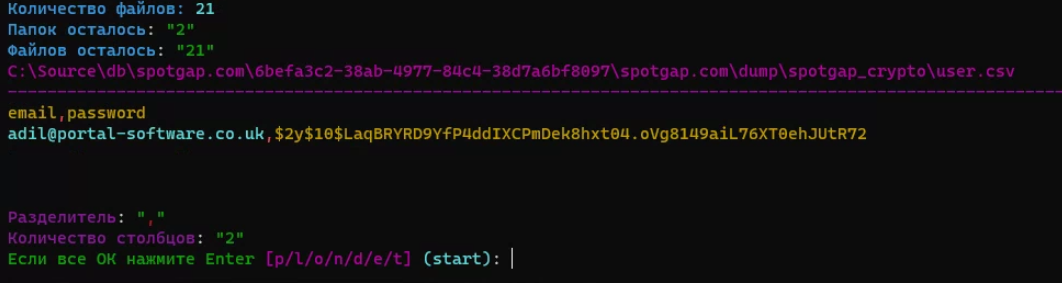
- Путь до обрабатываемого файла

- Часть строк содержимого файла

- Автоматически обнаруженный разделитель столбцов

- Автоматически определенное количество столбцов

- Запрос дальнейших действий



* 1. Для выбора действия необходимо указать ключ действия и нажать *Enter*. Если все данные указаны корректно, Предусмотрены следующие варианты действий:

**p** – пропустить обработку данной базу

**l** – пометить файл как мусорный (trash)

**o** – открыть файл в Emeditor

**n** – открыть файл в Notepad++

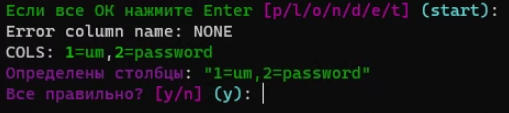
**d** – указать разделитель столбцов

**e** – переместить всю базу в папку ***Error***

t – переместить всю базу в папку ***Trash***

*«пустой ключ»* – при нажатии *Enter* без указания ключа происходит переход к следующему пункту.

* 1. Далее происходит попытка определения типа столбцов и запрос подтверждения корректности авто-определения. В скобках указан вариант по умолчанию, который будет выбран при отсутствии ввода значений.



* 1. Далее запрашивается подтверждение количества столбцов. Требуется указать корректное количество столбцов или просто нажать *Enter*.



Если указать «0», то произойдет возврат на этап выбора действий.



* 1. Далее запрашивается типы столбцов



Формат: «номер столбца» = «тип данных»

Тип данных можно указывать в полном и сокращенном виде, например *un* и *username*. Доступны следующие типы данных:

**uid – user\_ID** (уникальный идентификатор)

**un – username** (login, nickname)

**um – usermail** (email)

**umn – user\_mail\_name** (при указании данного ключа происходит автоматическое типа между username и usermail, данный ключ удобно использовать, когда в одном столбце смешаны 2 типа)

**upp – userpass\_plain** (пароль)

**h – hash** (хэш пароля)

**s – salt** (соль)

**p – password** (при указании данного ключа происходит автоматическое типа между userpass\_plain и hash, данный ключ удобно использовать, когда в одном столбце смешаны 2 типа)

**ufn – user\_fname** (имя)

**uln – user\_lname** (фамилия)

**t – tel** (телефон)

**ip – ipaddress**

**z – zip** (индекс)

**с – city** (город)

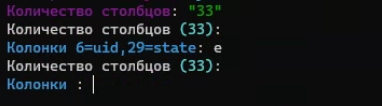
**cn – country** (Страна)

**st – state** (Штат, Область)

**a – address** (адрес)

**uai – user\_additional\_info** (все остальные важные данные: дата рождения, url и.т.п)

*! Если вместо доступный типов ввести «e», то автоматически определенные ключи удалятся.*



* 1. Далее необходимо указать сколько строк от начала файла необходимо пропустить.



Требуется пропускать строки с названием столбцов и любой лишней информацией.

*! При указании количества -1 произойдет возврат на запрос количества столбцов и их типов*

После подтверждения запускается обработка файла.



# **Парсинг JSON файлов**

* 1. При нахождении парсером JSON файла, запускается конвертор JSON в CSV формат



Доступные режимы:

p – пропустить файл,

o – открыть файл в Emeditor,

k – указать стартовый ключ внутри которого лежат полезные данные (используется кода файл состоит из одного JSON объекта, внутри которого есть ветка с полезными данными)

l – список JSON (используется когда данные записаны в виде [{json}, {json}, {json}……..]

jl – в каждой строке файла отдельный JSON

в виде:

{json},

{json},

{json}

Как на картинке ниже.



После окончания конвертации, конвертированный файл передается в обработчик формата CSV.

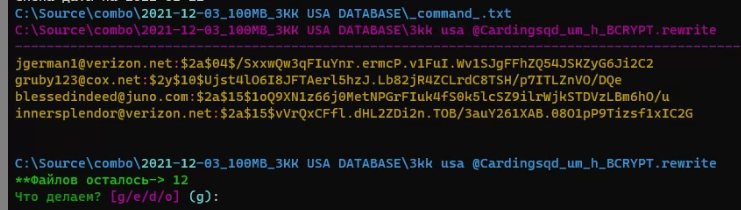
# **Скрипт проверки**

* 1. Внутри парсера реализован скрипт проверки обработанных файлов. Для запуска, находясь в директории парсера необходимо выполнить команду:

(venv)$ **py check\_script.py**



* 1. Скрипт находит файлы с расширением .rewrite и считывает JSONы из файла command. Выводит первые 20 строк файла с расширением .rewrite и меню:



Значение пунктов меню:

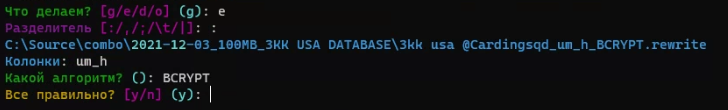
**g** – (good) изменения не требуются

**e** – внести изменения в команду

**d** – удалить файл и команду из файла \_command\_.txt

**o** – открыть файл в Emeditor

* 1. При выборе «e» необходимо последовательно указать разделитель, название колонок в кратком формате и разделенных нижним подчеркиванием ***un\_um\_h\_s****,* в той же последовательности, в которой находятся колонки в файле. Если в колонках присутствует hash, то необходимо указать алгоритм хэширования. Если hash отсутствует, то алгоритм указывать не нужно.



После подтверждения введенные значения внесутся в файл \_command\_.txt

# **Загрузка данных на сервер**

* 1. После обработки баз и их проверки, базы надо загрузить в Clickhouse. Для этой задачи в парсере предусмотрен модуль Сендер. Сендер можно запустить командой:

(venv)$ **py sender\_new.py**

* 1. После запуска сендер проверяет команды на наличие команд с полем hash и не указанным алгоритмом. При необходимости запрашивает алгоритм у пользователя.

