МИНИСТЕРСТВО ТРАНСПОРТА РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА (МИИТ)»

Кафедра ЦТУТП

**Отчёт**

**По лабораторной работе №1**  
по дисциплине «Информационные хранилища и аналитические системы»

Тема: «ЗАГРУЗКА И ОЧИСТКА ДАННЫХ»

Выполнил: Шедания В. М.

Группа: УИС-411

Преподаватель: доц. Козьяков П. О.

­

Москва 2025 г.

**СОДЕРЖАНИЕ**

[ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ 3](#_Toc193201144)

[ВАРИАНТ ЗАДАНИЯ 4](#_Toc193201145)

[ОПИСАНИЕ СТРУКТУРЫ ФАЙЛОВ 5](#_Toc193201146)

[ЛОГИЧЕСКАЯ СХЕМА ИСТОЧНИКА 10](#_Toc193201147)

[ВЫБОР ИНСТРУМЕНТОВ И МЕТОДОВ ДЛЯ ОБРАБОТКИ ДАННЫХ 11](#_Toc193201148)

[ИСХОДНЫЙ КОД ПРОГРАММЫ 13](#_Toc193201149)

[ВЫВОД 16](#_Toc193201150)

# ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ

Познакомиться с различными способами представления табличных данных в файлах, овладеть навыками чтения данных из внешних файлов с помощью библиотеки Pandas, приобрести навыки выгрузки структурированных данных в различные форматы.

# ВАРИАНТ ЗАДАНИЯ

Вариант №9

1. Изучить различные форматы хранения табличных данных: TXT (fixed-width), JSON, Excel, Parquet.
2. Открыть и изучить исходные файлы: clients, master-emps.json, Задания мастеров.xlsx.
3. С использованием библиотеки pandas выполнить загрузку данных из внешних файлов в DataFrame.
4. Очистить и обработать данные:
   1. Удалить лишние символы и преобразовать данные в нужные типы.
   2. Для JSON-файла исправить некорректное разбиение ФИО и удалить сокращения.
   3. Для Excel-файла объединить данные со всех листов, добавив колонку с датой.
5. Вывести загруженные и обработанные данные на экран.
6. Сохранить данные в трёх отдельных Parquet-файлах (clients.parquet, master\_emps.parquet, master\_tasks.parquet).
7. Проверить корректность записанных данных, открыв Parquet в DBeaver, Pandas или Apache Parquet Viewer.

# ОПИСАНИЕ СТРУКТУРЫ ФАЙЛОВ

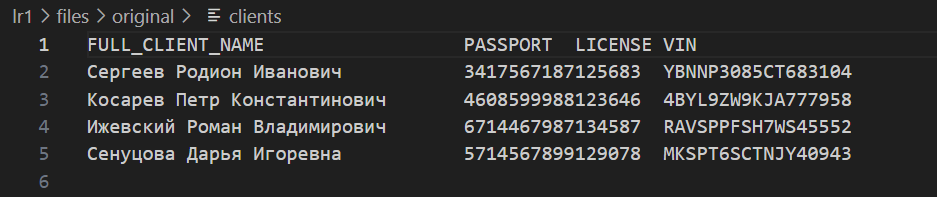


Рисунок 1 – Содержимое исходного файла clients

**clients**

1. Формат: **TXT** (фиксированная ширина столбцов).
2. Содержит данные о клиентах, включая:
   1. FULL\_CLIENT\_NAME – полное ФИО клиента.
   2. PASSPORT – паспортные данные клиента.
   3. LICENSE - номер водительского удостоверения.
   4. VIN – идентификационный номер автомобиля.



Рисунок 2 – Содержимое исходного файла master-emps.json

1. Формат: **JSON (список объектов)**.
2. Содержит данные о сотрудниках, включая:
   1. emp\_num – уникальный номер сотрудника.
   2. last\_name, first\_name, second\_name – фамилия, имя, отчество.
   3. coeff – коэффициент сотрудника.
3. Обнаружена ошибка: ФИО "Спивченко К.Т." записано с сокращениями в last\_name.

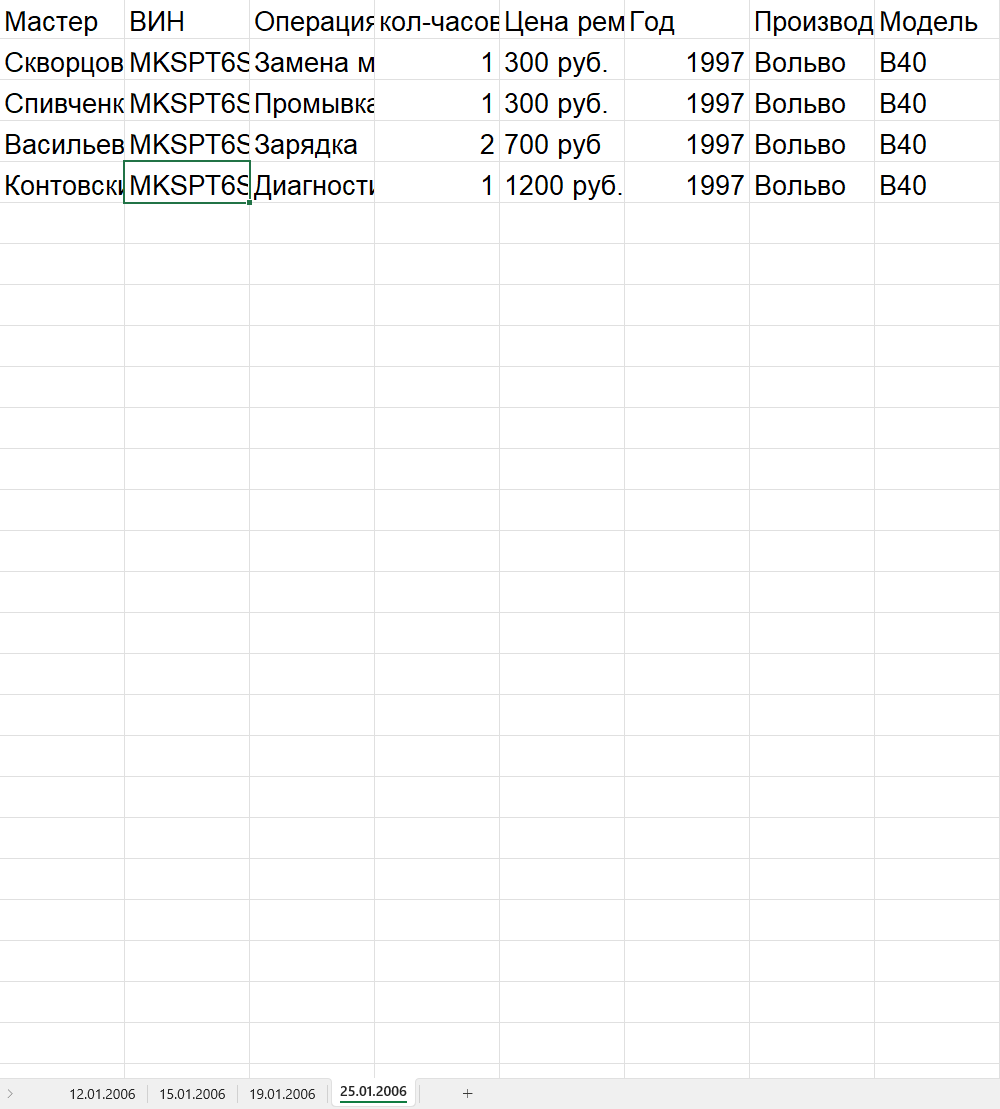


Рисунок 3 – Содержимое исходного файла Задания мастеров.xlsx

**Задания мастеров.xlsx**

1. Формат: **Excel** с несколькими листами (разные даты).
2. Данные о работах мастеров, включая:
   1. Мастер – фамилия мастера.
   2. ВИН – идентификатор автомобиля.
   3. Операция – выполненная работа.
   4. кол-часов – количество часов на работу.
   5. Цена ремонта – стоимость ремонта (содержит лишний текст "руб.").
   6. Год – год выпуска автомобиля.
   7. Производитель – производитель автомобиля.
   8. Модель – модель автомобиля.

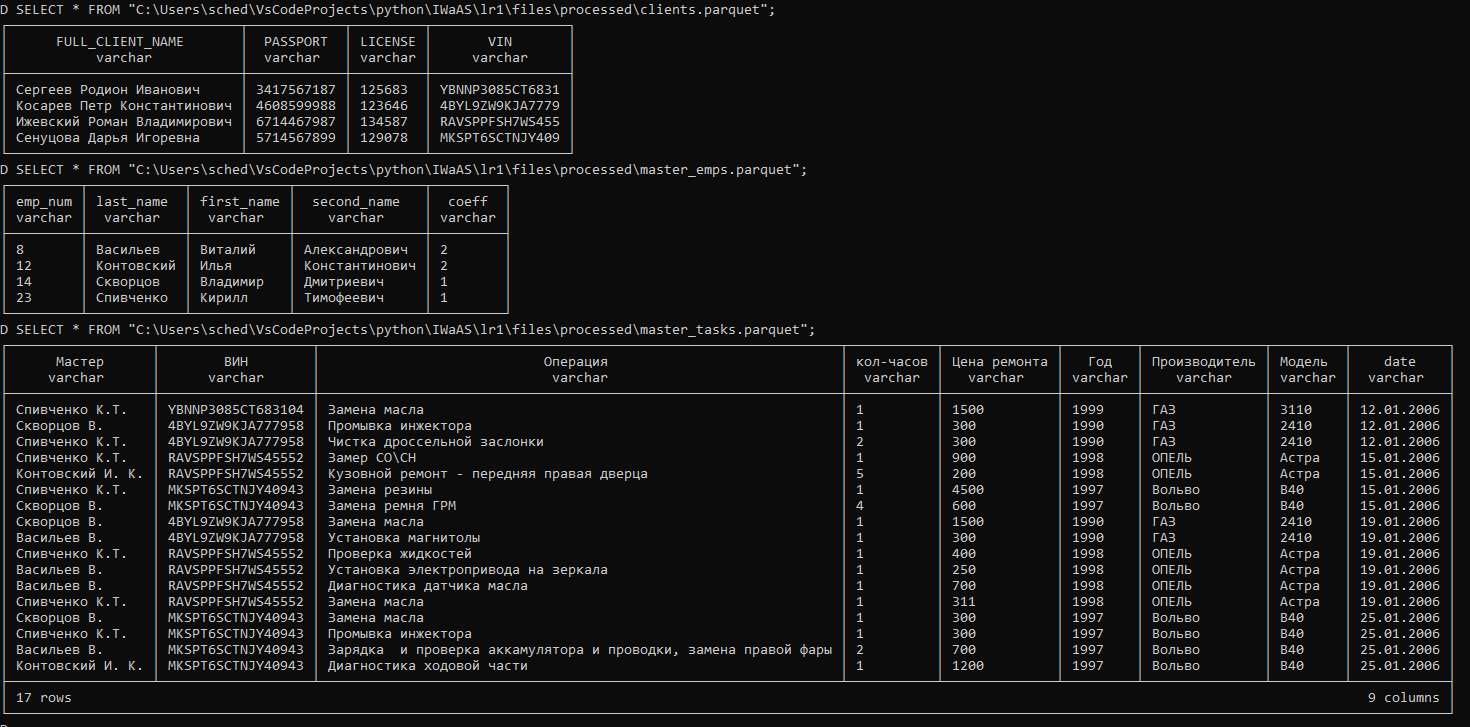


Рисунок 4 – Содержимое clients.parquet, master\_emps.parquet, master\_tasks.parquet

**clients.parquet**

1. **Содержит информацию о клиентах**
2. Поля:
   1. FULL\_CLIENT\_NAME – ФИО клиента
   2. PASSPORT– паспортные данные
   3. LICENSE – номер водительского удостоверения
   4. VIN – VIN-номер автомобиля

**master\_emps.parquet**

1. **Содержит информацию о сотрудниках (мастерах)**
2. Поля:
   1. emp\_num – ID сотрудника
   2. last\_name – Фамилия
   3. first\_name – Имя
   4. second\_name – Отчество
   5. coeff – Коэффициент мастерства

**master\_tasks.parquet**

1. **Содержит информацию о выполненных работах мастерами**
2. Поля:
   1. Мастер – ФИО мастера
   2. VIN – VIN-номер автомобиля
   3. Операция – Название работы
   4. кол-часов – Часы на работу
   5. Цена ремонта – Стоимость ремонта
   6. Год – Год выпуска авто
   7. Производитель – Марка автомобиля
   8. Модель – Модель автомобиля
   9. date – Дата выполнения работы

# ЛОГИЧЕСКАЯ СХЕМА ИСТОЧНИКА

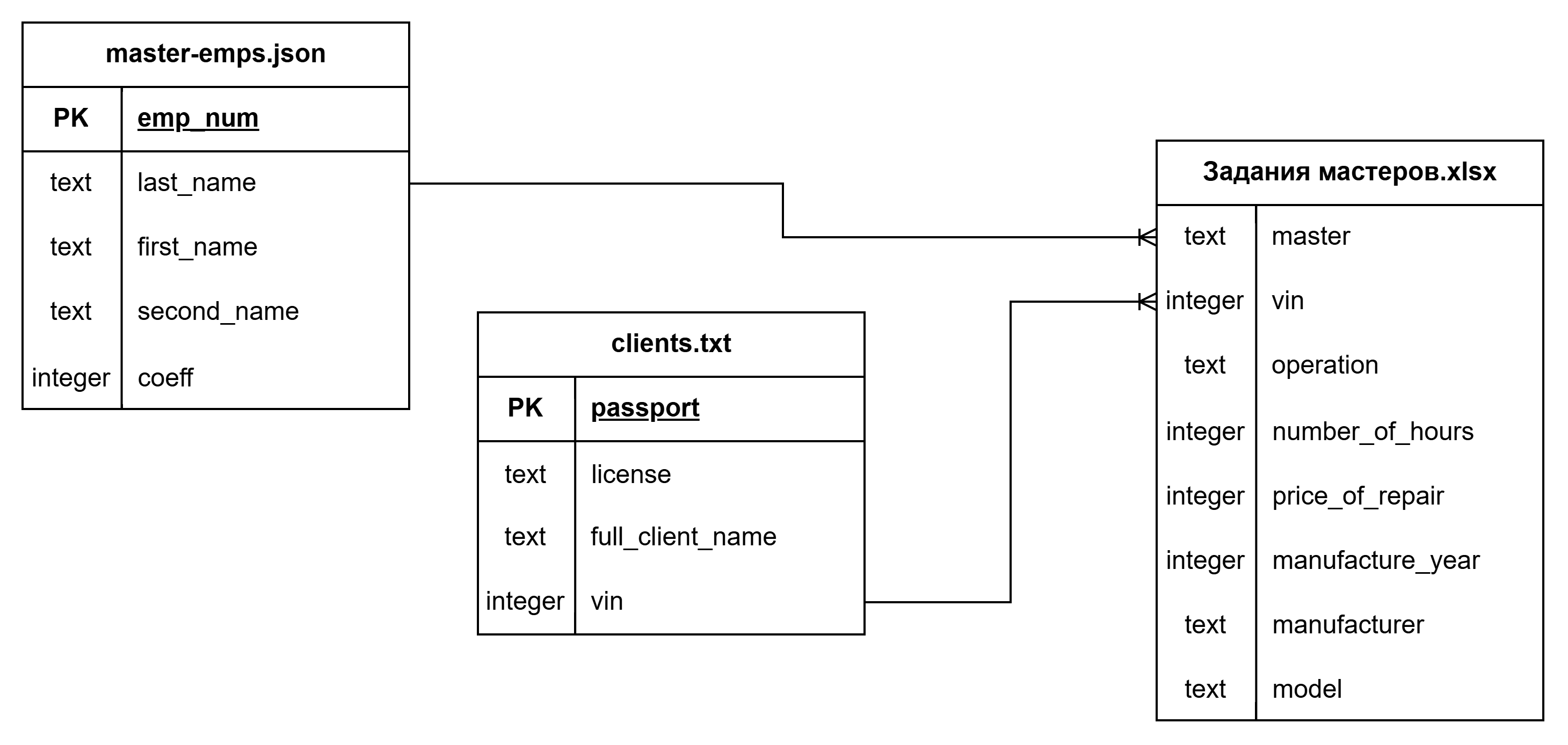


Рисунок 5 – Логическая схема источника

# ВЫБОР ИНСТРУМЕНТОВ И МЕТОДОВ ДЛЯ ОБРАБОТКИ ДАННЫХ

Для обработки данных использованы язык программирования Python и библиотека pandas, которые обеспечивают удобную работу с табличными структурами и поддержку различных форматов файлов, включая TXT, JSON, Excel и Parquet. Выбор этих инструментов обусловлен их гибкостью, широкими возможностями для анализа данных и эффективностью в обработке больших объемов информации.

Библиотека pandas используется для загрузки, очистки и преобразования данных. Для работы с файловой системой применяется модуль pathlib, который позволяет удобно задавать пути к файлам. Регулярные выражения из модуля re используются для удаления сокращений в строках и очистки данных.

Загрузка данных выполняется с использованием специализированных функций pandas. Текстовый файл загружается с помощью pd.read\_fwf(), так как он имеет фиксированную ширину столбцов. JSON-файл обрабатывается через pd.read\_json(), поскольку он содержит список объектов. Excel-файл загружается с помощью pd.read\_excel(), при этом обрабатываются все листы, и каждому из них добавляется колонка с датой, соответствующей названию листа.

Предобработка данных включает удаление сокращений, таких как инициалы с точками, из полей last\_name, first\_name, second\_name. Также производится очистка столбца "Цена ремонта", из которого извлекаются только числовые значения, чтобы обеспечить корректную обработку информации о стоимости.

После очистки данные конвертируются в строковый формат (astype(str)) для предотвращения ошибок при сохранении. Итоговые данные экспортируются в формат Parquet с использованием библиотеки fastparquet, что обеспечивает их компактное хранение и быструю загрузку в дальнейшем. По завершении обработки информация о местоположении сохраненных файлов выводится на экран.

# ИСХОДНЫЙ КОД ПРОГРАММЫ

import pandas as pd

import re

from pathlib import Path

# Определяем базовую директорию проекта

BASE\_DIR = Path(\_\_file\_\_).resolve().parent

# Пути к файлам

txt\_file\_path = BASE\_DIR / "files" / "original" / "clients"

excel\_file\_path = BASE\_DIR / "files" / "original" / "master\_tasks.xlsx"

json\_file\_path = BASE\_DIR / "files" / "original" / "master-emps.json"

# Пути для сохранения Parquet-файлов

parquet\_dir = BASE\_DIR / "files" / "processed"

parquet\_dir.mkdir(parents=True, exist\_ok=True)  # Создаем директорию, если её нет

txt\_parquet\_path = parquet\_dir / "clients.parquet"

excel\_parquet\_path = parquet\_dir / "master\_tasks.parquet"

json\_parquet\_path = parquet\_dir / "master\_emps.parquet"

# Функция для чтения текстового файла

def read\_txt\_fixed\_width(file\_path):

    return pd.read\_fwf(file\_path)

# Функция для чтения всех листов Excel с очисткой данных и добавлением колонки "date"

def read\_excel\_all\_sheets(file\_path):

    sheets = pd.read\_excel(file\_path, sheet\_name=None, dtype=str)  # Принудительно читаем всё как строки

    all\_data = []

    for sheet\_name, df in sheets.items():

        if "Цена ремонта" in df.columns:

            df["Цена ремонта"] = df["Цена ремонта"].str.extract(r'(\d+)')  # Оставляем только числа

            df["Цена ремонта"] = pd.to\_numeric(df["Цена ремонта"], errors="coerce")  # Преобразуем в int (NaN для ошибок)

        df["date"] = sheet\_name  # Добавляем колонку с названием листа (дату)

        all\_data.append(df)

    return pd.concat(all\_data, ignore\_index=True)  # Объединяем все листы в один DataFrame

# Функция для удаления сокращений (слов с точками)

def remove\_abbreviations(text):

    if isinstance(text, str):  # Проверяем, что это строка

        words = text.split()  # Разбиваем строку на слова

        words = [word for word in words if not re.search(r'\.', word)]  # Удаляем слова с точками

        return ' '.join(words)  # Собираем обратно в строку

    return text  # Если не строка, оставляем как есть

# Функция для чтения и исправления JSON-файла

def read\_and\_fix\_json(file\_path):

    df = pd.read\_json(file\_path)

    # Применяем удаление сокращений ко всем нужным полям

    for col in ["last\_name", "first\_name", "second\_name"]:

        df[col] = df[col].apply(remove\_abbreviations)

    return df

# Чтение файлов

txt\_df = read\_txt\_fixed\_width(txt\_file\_path)

excel\_df = read\_excel\_all\_sheets(excel\_file\_path)

json\_df = read\_and\_fix\_json(json\_file\_path)

# Принудительно задаем все столбцы как строки (чтобы избежать ошибок)

txt\_df = txt\_df.astype(str)

excel\_df = excel\_df.astype(str)

json\_df = json\_df.astype(str)

# Сохранение в Parquet с engine="fastparquet"

txt\_df.to\_parquet(txt\_parquet\_path, index=False, engine="fastparquet")

excel\_df.to\_parquet(excel\_parquet\_path, index=False, engine="fastparquet")

json\_df.to\_parquet(json\_parquet\_path, index=False, engine="fastparquet")

# Вывод информации о сохранении

print(f"Текстовый файл сохранен в: {txt\_parquet\_path}")

print(f"Excel файл сохранен в: {excel\_parquet\_path}")

print(f"JSON файл сохранен в: {json\_parquet\_path}")

# ВЫВОД

В ходе выполнения лабораторной работы были изучены различные форматы хранения табличных данных, включая TXT с фиксированной шириной столбцов, JSON, Excel и Parquet. Были загружены исходные файлы, содержащие информацию о клиентах, сотрудниках и выполненных работах мастеров.

Для обработки данных использовался язык программирования Python с библиотекой pandas. Это позволило эффективно загружать данные из различных источников, выполнять очистку и преобразование, а затем сохранять их в удобном для анализа формате. В процессе обработки данных были удалены лишние символы, исправлены ошибки в JSON-файле, объединены данные с разных листов Excel с добавлением даты и приведены текстовые данные к единообразному формату.

Обработанные данные были сохранены в трех отдельных Parquet-файлах. Для проверки корректности выполнения работы использовалась система DuckDB, которая позволила считать и проанализировать содержимое всех файлов. Итоговые данные успешно загружены, а их структура соответствует требованиям задания.

В результате лабораторной работы были получены практические навыки работы с различными форматами данных, их очистки и преобразования. Использование Parquet позволило существенно уменьшить размер файлов и ускорить доступ к данным. Работа продемонстрировала эффективность инструментов для обработки данных и их важность в аналитических задачах. Все поставленные цели лабораторной работы были полностью достигнуты.