Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики управления и технологий

Ли Александр Андреевич БД-241м

**Лабораторная работа 4.3 Интеграция данных из нескольких источников.**

Направление подготовки/специальность

38.04.05 - Бизнес-информатика

Бизнес-аналитика и большие данные

(очная форма обучения)

Вариант 12

Москва

2024

**Цель работы**: получить практические навыки интеграции, обработки и согласования данных из различных источников.

**Задачи**:

Изучить методы чтения данных из разных источников.

Освоить техники обработки и очистки данных.

Научиться согласовывать данные из разных источников.

Реализовать сохранение обработанных данных.

**Условие индивидуального задания**

|  |  |  |
| --- | --- | --- |
| Оборудование | • PostgreSQL база данных инвентаризации • Excel файл с графиком обслуживания • CSV файл с историей ремонтов | Создать систему учета технического обслуживания |

Индивидуальное задание 3

Написали код для создания синтетических данных

!pip install Faker

import pandas as pd

import numpy as np

from datetime import datetime, timedelta

import random

from faker import Faker

import os

from google.colab import files

np.random.seed(42)

fake = Faker('ru\_RU')

products = [

'cisco\_switch', 'cisco\_router', 'pc',

'nas', 'printer', 'scaner','ip\_phone'

]

service = [

'software update', 'hardware diagnostics', 'dust cleaning',

'port testing', 'thermalpaste replace', 'error log analysis'

]

def generate\_inventory\_and\_maintenance\_data(num\_records):

    data = []

    current\_date = datetime(2025, 3, 1)  # начальная дата

    for \_ in range(num\_records):

        equipment = random.choice(products)  # выбор оборудования

        maintenance\_status = random.choice(['Запланировано', 'Выполнено', 'Отложено', 'Не выполнено'])  # Статус обслуживания

        maintenance\_type = random.choice(service)  # Тип обслуживания

        record = {

            'maintenance\_id': f'INV{\_+1:05d}',  # Уникальный идентификатор инвентаря

            'equipment\_name': equipment,  # Название оборудования

            'installation\_date': (current\_date - timedelta(days=random.randint(1, 365))).strftime('%Y-%m-%d'),  # Дата установки

            'maintenance\_interval\_days': random.choice([30, 60, 90]),  # Интервал обслуживания в днях

            'operating\_condition': random.choice(['Хорошее', 'Удовлетворительное', 'Плохое']),  # Эксплуатационные условия

            'maintenance\_status': maintenance\_status,  # Статус обслуживания

            'maintenance\_type': maintenance\_type,  # Тип обслуживания

            'quality\_score': random.randint(75, 100),  # Оценка качества оборудования

            'inventory\_status': random.choice(['На складе', 'В эксплуатации', 'В ремонте'])  # Статус инвентаря

        }

        data.append(record)

        current\_date += timedelta(days=random.randint(1, 5))  # Обновление даты для следующей записи

    return pd.DataFrame(data)

num\_records = 1000  # Количество записей

inventory\_maintenance\_df = generate\_inventory\_and\_maintenance\_data(num\_records)

print(inventory\_maintenance\_df)

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.1 Полученные данные для postgress

inventory\_maintenance\_df.to\_csv("quality\_control\_data.csv", index=False, encoding="utf-8-sig", header=False)

Сохраняем файл в csv формате

создали таблицу в Postgress

CREATE TABLE inventory\_and\_maintenance (

maintenance\_id VARCHAR(10) PRIMARY KEY,

equipment\_name VARCHAR(13),

installation\_date DATE,

maintenance\_interval\_days INT,

operating\_condition VARCHAR(19),

maintenance\_status VARCHAR(13),

maintenance\_type VARCHAR(21),

quality\_score INT,

inventory\_status VARCHAR(15));

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описание

Рис.2 создание таблицы в postgresql

Загружаем csv файл в pgadmin

Нажимаем import/export data и указываем путь к файлу

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рис.3 Загружаем файл

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рис.4 Загружаем файл

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рис.4 Проверка загруженных файлов

Для создания exel файла

def generate\_maintenance\_records(num\_records):

    data = []

    current\_date = datetime(2025, 3, 1)  # Стартовая дата

    for i in range(num\_records):

equipment = random.choice(products)  # Случайный выбор оборудования

record = {

'maintenance\_id': f'MTN{i+1:05d}',  # Уникальный идентификатор обслуживания

'equipment\_name': equipment,  # Название оборудования

'next\_maintenance\_date': (current\_date + timedelta(days=random.randint(30, 90))).strftime('%Y-%m-%d'),  # Следующее обслуживание

'last\_maintenance\_date': (current\_date - timedelta(days=random.randint(30, 90))).strftime('%Y-%m-%d'),  # Последнее обслуживание

'maintenance\_interval': random.choice([30, 60, 90]),  # Интервал обслуживания в днях

'operating\_condition': random.choice(['Хорошее', 'Удовлетворительное', 'Плохое']),  # Эксплуатационные условия

'last\_maintenance\_score': random.randint(75, 98),  # Оценка качества последнего обслуживания

'failure\_rate': round(random.uniform(0.01, 0.05), 3),  # Максимальная вероятность отказа

'maintenance\_status': random.choice(['Запланировано', 'Выполнено', 'Отложено', 'Не выполнено']),  # Статус обслуживания

'repair\_cost\_estimate': random.choice([0, 1000, 2000, 5000, 10000])  # Оценочная стоимость ремонта

}

        data.append(record)

        current\_date += timedelta(days=random.randint(1, 5))

    return pd.DataFrame(data)

maintenance\_df = generate\_maintenance\_records(1000)

Изображение выглядит как текст, снимок экрана, число, дисплей

Автоматически созданное описание

Рис.4 Проверка полученных синтетических данных

Изображение выглядит как текст, снимок экрана, меню, число

Автоматически созданное описание

Рис.5 Проверка полученных синтетических данных

Для создания csv файла

def generate\_maintenance\_data(start\_date, num\_records):

    data = []

    current\_date = start\_date

    for \_ in range(num\_records):

        equip = random.choice(products)

        issue = random.choice(service)

        record = {

            'maintenance\_request\_id': f'M{\_+1:05d}',  # Идентификатор запроса на ремонт

            'repair\_request\_date': current\_date.strftime('%Y-%m-%d'),  # Дата запроса на ремонт

            'equipment\_name': equip,  # Название оборудования

            'equipment\_id': f'EQUIP{random.randint(10000, 99999)}',  # Уникальный ID оборудования

            'install\_date': (current\_date - timedelta(days=random.randint(1, 90))).strftime('%Y-%m-%d'),  # Дата установки

            'fault\_type': issue,  # Тип неисправности

            'fault\_severity': random.choice(['Низкая', 'Средняя', 'Высокая']),  # Степень серьезности неисправности

            'fault\_description': f'Обнаружена {issue.lower()} после {random.randint(1, 30)} дней эксплуатации',  # Описание неисправности

            'repair\_status': random.choice(['Новая', 'В обработке', 'Решена', 'Отклонена']),  # Статус ремонта

            'repair\_cost': random.choice([0, 1000, 2000, 5000, 10000])  # Оценочная стоимость

        }

        data.append(record)

        current\_date += timedelta(days=random.randint(0, 2))

    return pd.DataFrame(data)

start\_date = datetime(2025, 3, 1)

num\_records = 1000

maintenance\_df = generate\_maintenance\_data(start\_date, num\_records)

 maintenance\_df.to\_csv('customer\_complaints.csv', index=False, encoding='utf-8-sig')

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рис.6 Проверка полученных синтетических данных

Cоздаем трансформацию, в которой будут объединены все данные со всех баз данных

Изображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Автоматически созданное описание

Рис.7 Структура трансформации

Испортируем данные с postgres

SELECT maintenance\_id, equipment\_name, installation\_date, maintenance\_interval\_days, operating\_condition, maintenance\_status, maintenance\_type, quality\_score, inventory\_status

FROM public.inventory\_and\_maintenance;

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рис.7 Импортируем данные с postgres

Смотрим на поля полученные из postgres

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рис.8 проверка полученных данных

Импортируем xl файл

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.9 импортируем данные из xl

Проверяем полученные столбцы в select values

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рис.9 проверка полученных данные из xl

Делаем сортировку строк, для того чтобы merge join работал корректно

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Автоматически созданное описание

Рис.10 сортировка данных

Импортируем csv файл

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.11 импортируем данные из csv файла

Проверяем полученные данные в select values

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание Рис.12 проверка данные из csv файла

Сортируем csv данные

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Автоматически созданное описание

Рис.13 сортировка данных

Объединяем данные из xl и csv файлов

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание Рис.14 обьединение xl и csv

Выбираем нужные значения в select values после обьединения xl и csv файла

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рис.15 проверка полученных данных после объединение xl и csv

Далее соединяем данные с postgress и xl,csv

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рис.16 объединение postgresql и xl,csv

Проверяем полученные значения после всех объединений

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рис.17 проверяем полученные значения

Делаем соединение с целевой базой данных и выбираем нужные значения для экспорта

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рис.18 экспорт данных в phpmyadmin

Создаем job

Изображение выглядит как диаграмма, текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис.19 структура job

используем ранее описанную трансформацию

добавляем write to log, чтобы ошибки записывались в логи

Cоздаем таблицу в phpMyAdmin

Создание системы учета технического обслуживания

[CREATE](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) zadanie4\_3

( maintenance\_id [VARCHAR](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-types.html)(6) PRIMARY KEY,

 equipment\_name [VARCHAR](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-types.html)(13),

installation\_date [DATE](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/date-and-time-types.html),

maintenance\_interval\_days [INT](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/numeric-types.html),

operating\_condition [VARCHAR](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-types.html)(20),

 maintenance\_status [VARCHAR](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-types.html)(50),

 maintenance\_type [VARCHAR](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-types.html)(20),

 quality\_score [INT](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/numeric-types.html),

inventory\_status [VARCHAR](http://95.131.149.21:8080/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-types.html)(15)

);

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рис.20 загружаем данные в целевую базу данных

Заключение

В ходе проделанной лабораторной работы, были получены практические навыки генерации синтетических данных, интеграции, обработки и согласования данных из различных источников.