Департамент образования города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ОТЧЕТ

по дисциплине «Интеграция и развертывание программного обеспечения с помощью контейнеров»

Направление подготовки 38.03.05 – бизнес-информатика

Профиль подготовки «Аналитика данных и эффективное управление»

(очная форма обучения)

Лабораторная работа 3.1 Интеграция данных из нескольких источников. Обработка и согласование данных из разных источников

Выполнил: st\_89

Руководитель:

Москва  
2025

**Цель:** Получить практические навыки интеграции, обработки и согласования данных из различных источников с использованием Python и его библиотек. (Вариант 2)

Задачи:

1. Верхнеуровневая диаграмма
2. Вариант 2.
   1. Интеграция данных о сотрудниках: -
      1. PostgreSQL база с личными данными.
      2. Excel файл с данными об окладах. - CSV файл с данными о премиях.
      3. Задача: создать единый отчет по заработной плате

**Ход работы:**

1. Верхнеуровневая архитектура

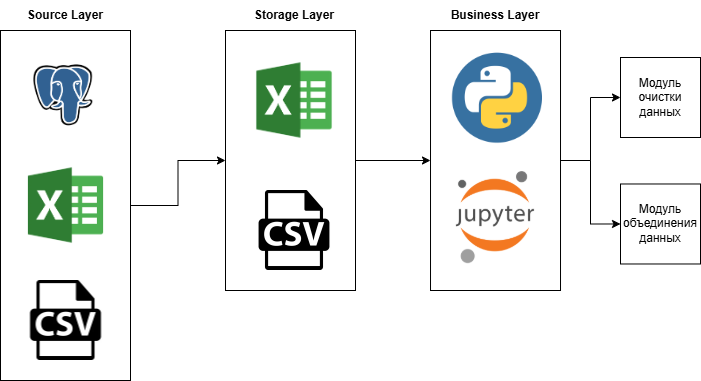


Рис. 1 - Верхнеуровневая архитектура

1. Загружаем все необходимые файлы, включая с подключением к PostgreSQL

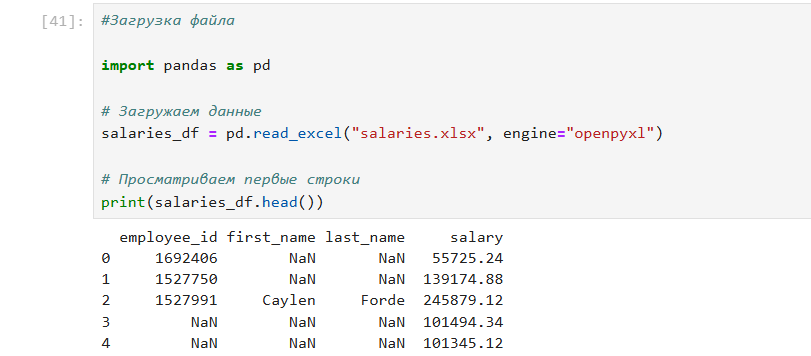


Рис. 2 – Загрузка данных формата из xlsx



Рис. 3 – Подключение к базе и вывод данных

Получим следующий набор данных из PostgreSQL

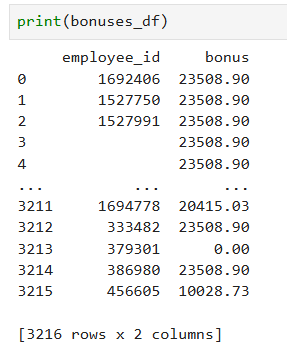


Рис. 4 – Данные премий сотрудников

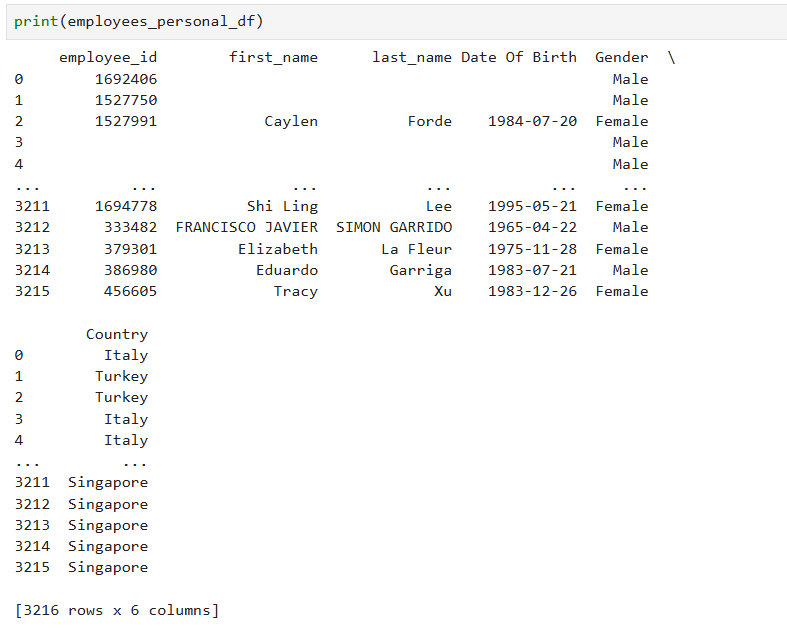


Рис. 5 – Персональные данные сотрудников

1. Предобработка данных

На этом этапе сделаем очистку данных

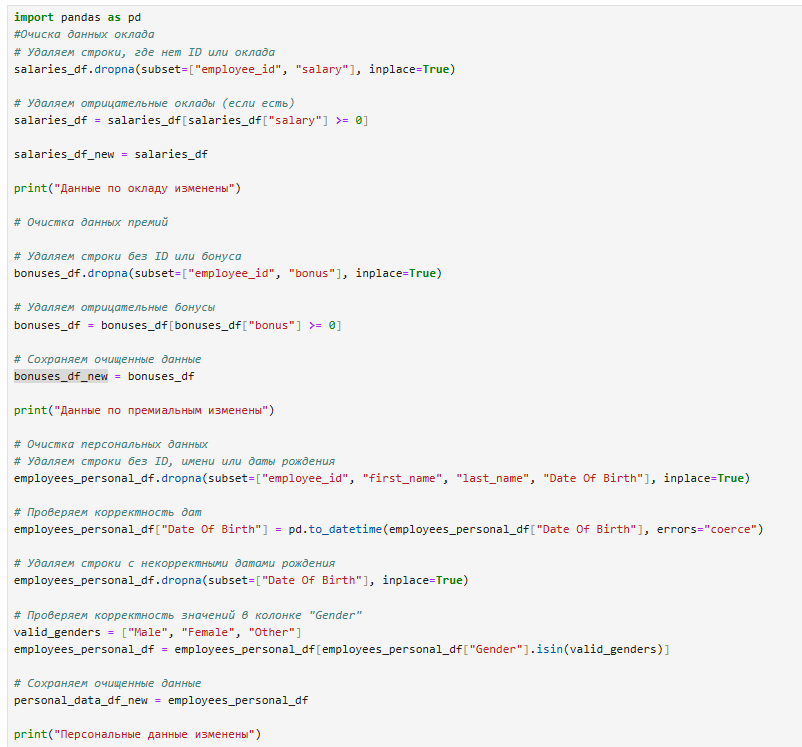


Рис. 6 – Очистка данных

1. Создание отчета

С помощью merge объединим данные и создадим отчет.  
Также добавим модуль логирования.



Рис. 7 – Создание отчета

1. Анализ данных

Проведем анализ данных по окладу сотрудников

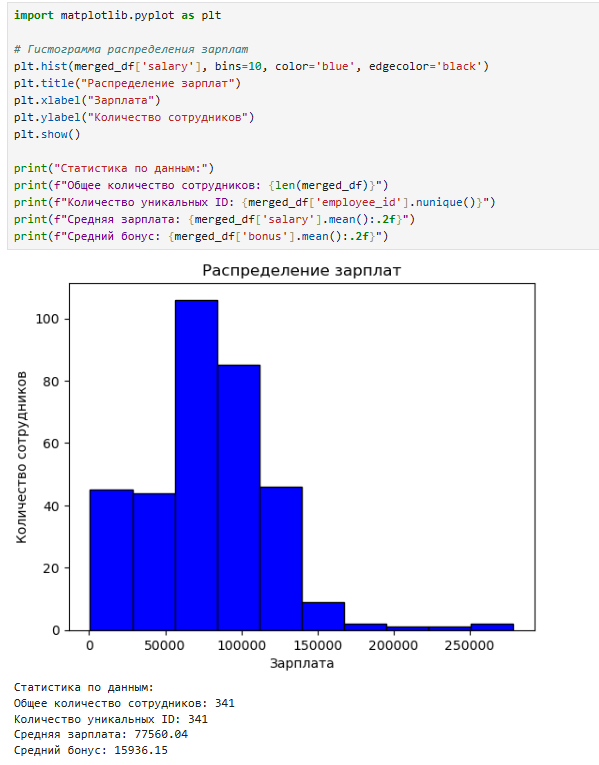


Рис. 8 – Анализ данных

Получим, что средняя зарплата 77560.04 и средние премиальные: 15936.15

**Вывод:** Была проведена загрузка данных из PostgreSQL. Проведена обработка данных и создание отчета по данным сотрудников. Также сделан анализ и выведены средние показатели

**Контрольные вопросы:**

**1. Какие методы чтения данных предоставляет pandas?**

* **pd.read\_csv()**: Чтение данных из CSV-файлов.
* **pd.read\_excel()**: Чтение данных из Excel-файлов.
* **pd.read\_sql()**: Чтение данных из базы данных SQL.
* **pd.read\_json()**: Чтение данных из JSON-файлов.
* **pd.read\_html()**: Чтение данных с веб-страниц (если таблица в формате HTML).
* **pd.read\_parquet()**: Чтение данных из файлов формата Parquet.
* **pd.read\_sql\_table()**: Чтение данных из таблицы SQL.
* **pd.read\_feather()**: Чтение данных из файлов Feather

### 2. ****Как обрабатывать пропущенные значения?****

* **isna() или isnull()**: Определение пропущенных значений.
* **fillna()**: Заполнение пропущенных значений.
* **dropna()**: Удаление строк или столбцов с пропущенными значениями.

### 3. ****Какие типы объединения данных существуют?****

* **merge()**: Слияние данных по ключевым колонкам.
  + how='inner': Включает только те строки, которые существуют в обоих DataFrame (по умолчанию).
  + how='left': Включает все строки из левого DataFrame и соответствующие строки из правого.
  + how='right': Включает все строки из правого DataFrame и соответствующие строки из левого.
  + how='outer': Включает все строки из обоих DataFrame, заполняя пропуски NaN.
* **concat()**: Конкатенация данных по осям (по строкам или столбцам).
* **join()**: Объединение DataFrame с использованием индексов или других колонок.

### 4. ****Как проверить качество объединения данных?****

* **Проверка на пропущенные значения**:

df.isnull().sum()

* **Проверка количества строк**:

print(f"До объединения: {len(df1)}")

print(f"После объединения: {len(merged\_df)}")

* **Проверка на дубликаты**:
  + df.duplicated().sum()

### 5. ****Какие методы дедупликации данных существуют?****

* **drop\_duplicates()**: Удаление дубликатов на основе всех или выбранных колонок.
* **Удаление дубликатов по всем столбцам**:
  + **duplicated()**: Проверка на дубликаты. Этот метод возвращает булевый массив, показывающий, являются ли строки дубликатами.
* **Удаление дубликатов с сохранением первой или последней строки**:

df.drop\_duplicates(keep='first', inplace=True): Сохранение первой строки

df.drop\_duplicates(keep='last', inplace=True): Сохранение последней строки