

Содержание

Введение	3
1 Постановка задачи.....	4
2 Реализация практических задач.....	5
2.1 Импорт базы данных в PostgreSQL	5
2.2 Написание CRUD-запросов	7
2.3 Проектирование собственной базы данных.....	8
2.4 Перенос спроектированной базы данных в PostgreSQL	9
2.5 Реализация ETL-процесса.....	9
Заключение	14
Список использованных источников.....	15

Введение

SQL/ETL разработка является очень важным и востребованным видом деятельности. Крупным компаниям всегда приходится иметь дело с большим объемом данных, поэтому проблемы их организации, чтения и модификации являются очень актуальными и хорошо изученными вопросами.

В ходе производственной практики необходимо закрепить полученные из теоретического курса знания о Big Data и реализовать основные методы работы с базами данных.

Место прохождения практики – АО «Неофлекс Консалтинг» – компания, фокусирующаяся на заказной разработке программного обеспечения и внедрении сложных информационных систем. В компании используются передовые технологии и подходы, с некоторыми из которых студентов знакомят за время практики.

1 Постановка задачи

Главной целью работы является применение на практике с целью лучшего понимания и изучения освоенных в теоретическом курсе материалов, таких как: основы языка PL/SQL, введение в ETL (Extract, Transform, Load) процессы, работа с системой управления базами данных (СУБД) PostgreSQL. При этом должны быть реализованы следующие практические задачи:

- 1) Установка и настройка PostgreSQL;
- 2) Импорт базы данных в PostgreSQL;
- 3) Написание CRUD-запросов к базе данных;
- 4) Проектирование собственной базы данных;
- 5) Создание новой базы данных;
- 6) Реализация простейшего ETL-процесса с помощью Python (чтение, преобразование/очистка данных, экспорт);
- 7) Импорт полученных данных в таблицу в PostgreSQL.

2 Реализация практических задач

2.1 Импорт базы данных в PostgreSQL

На персональный компьютер была загружена и установлена система управления базами данных PostgreSQL. В дальнейшем работа с данной СУБД будет осуществляться посредством pgAdmin 4 – платформы для администрирования и разработки PostgreSQL:

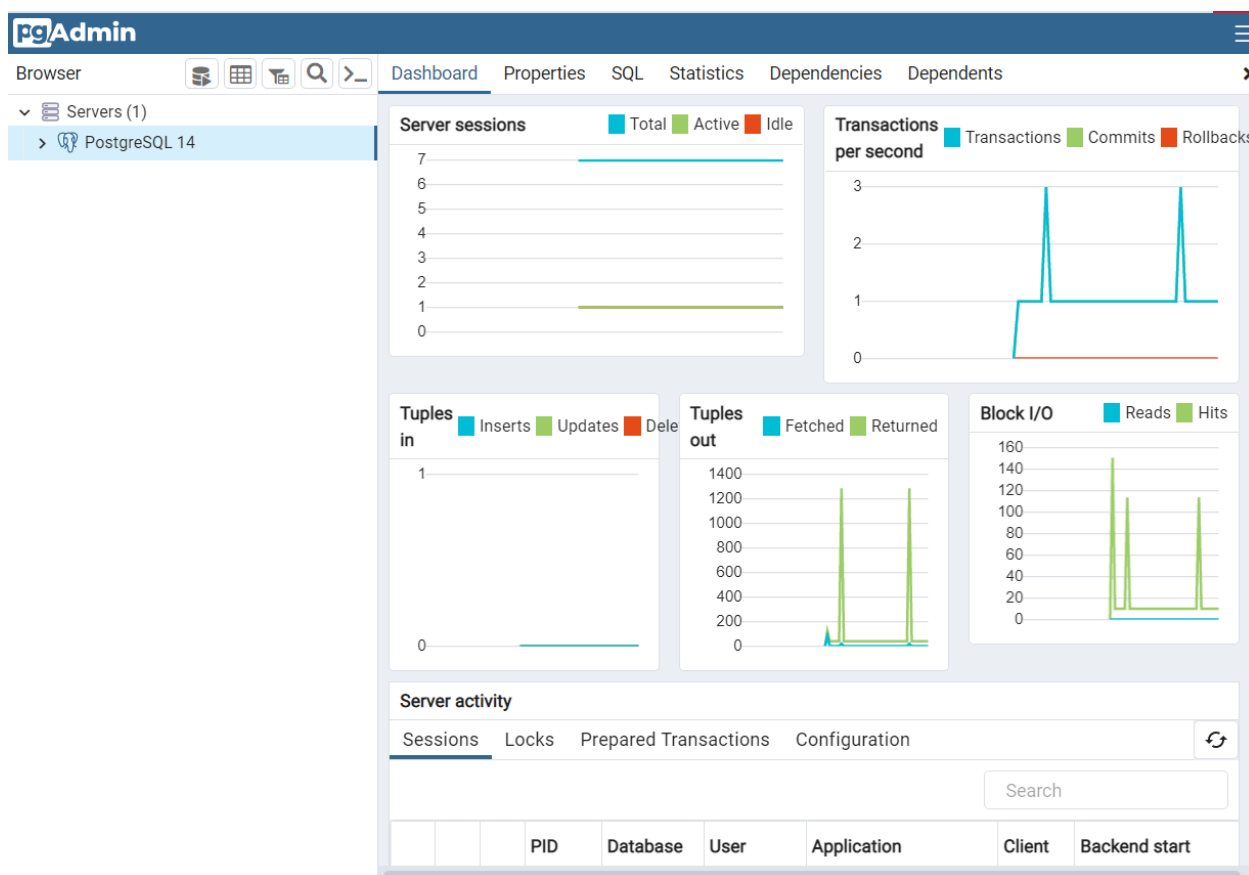


Рис. 2.1.1. Рабочий интерфейс pgAdmin

Также был загружен отправленный куратором SQL-скрипт, который создает новую базу данных, добавляет в неё таблицы, связи, и заполняет данными. С помощью командной строки данные были импортированы в PostgreSQL:

```

C:\Users\ankos>cd C:\Users\ankos\Desktop

C:\Users\ankos\Desktop>"D:\Programming\PostgreSQL\bin\psql" -U postgres -f Cars.sql
Пароль пользователя postgres:
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
INSERT 0 3
INSERT 0 4
INSERT 0 3
INSERT 0 4

C:\Users\ankos\Desktop>

```

Рис. 2.1.2. Работа с PostgreSQL с помощью командной строки

При импорте данных не возникло никаких ошибок. База данных была успешно перенесена:

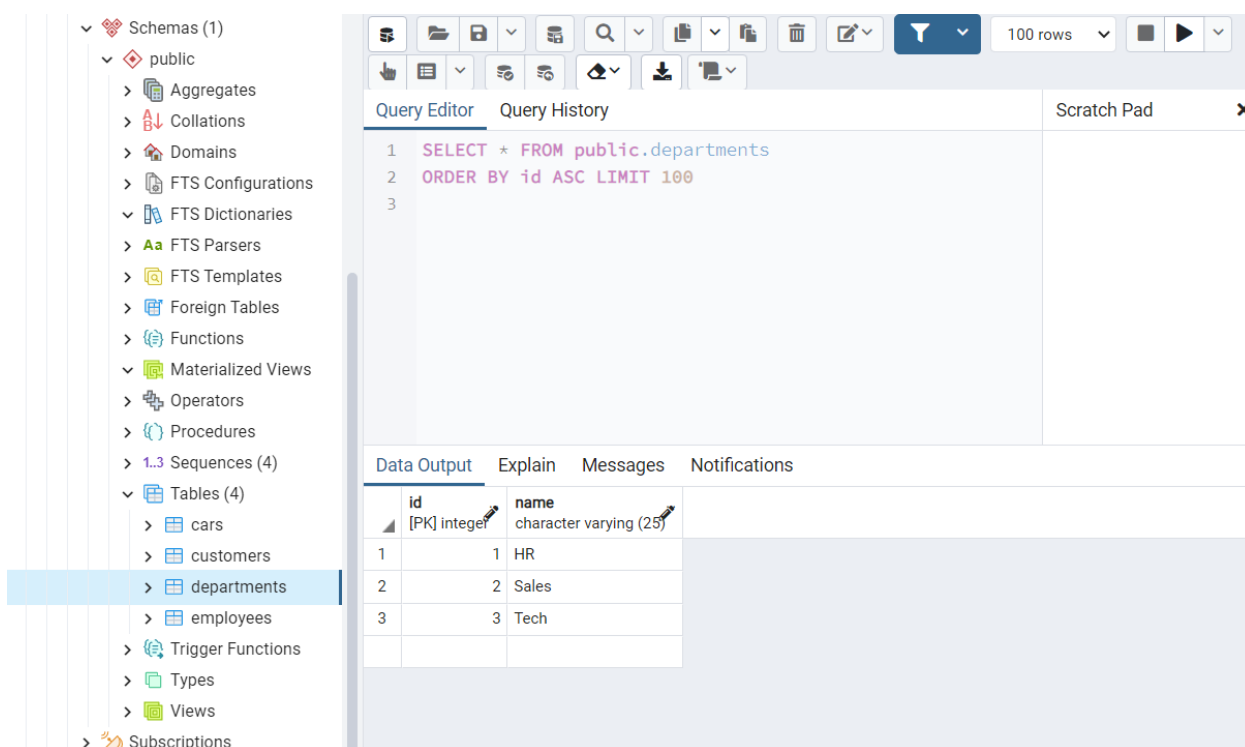


Рис. 2.1.3. База данных в PostgreSQL

Сама база данных содержит в себе информацию об отделе по продажам машин. В ней представлены данные о рабочих, офисах предприятия, покупателях, а также о проданных машинах.

2.2 Написание CRUD-запросов

В целях закрепления полученных знаний о языке PL/SQL было написано несколько SELECT-запросов к базе данных. При написании запросов была продемонстрирована работа с встроенными функциями и подзапросами. Каждый из запросов выдал корректный результат и был проверен куратором:

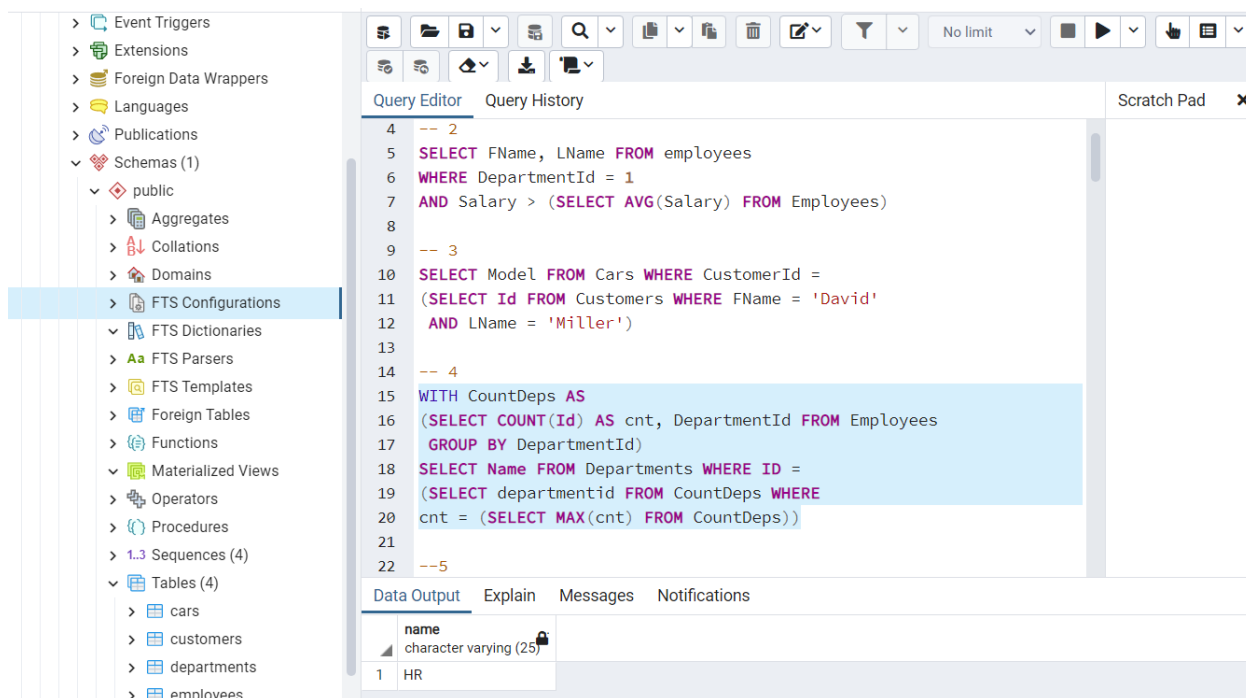


Рис. 2.2.1 SELECT-запросы к базе данных в PostgreSQL

Также были написаны запросы, направленные на изменение, удаление или обновление данных:

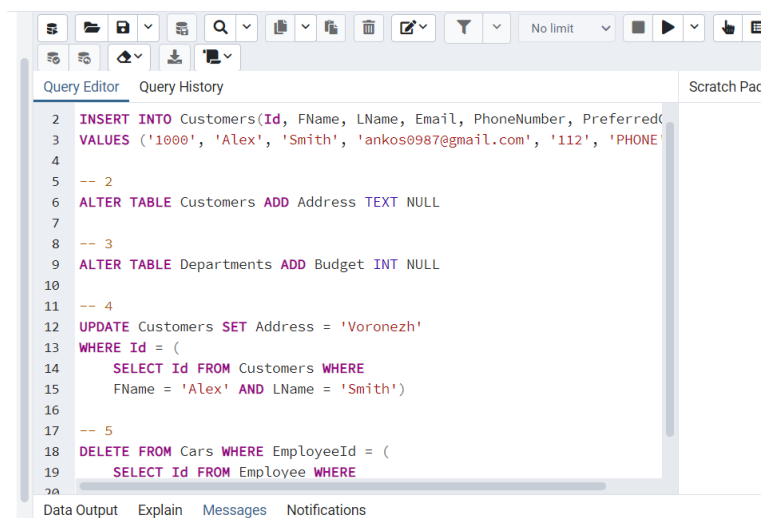


Рис. 2.2.2. Запросы, направленные на манипулирование данными

2.3 Проектирование собственной базы данных

Необходимо спроектировать базу данных, содержащую информацию о проектах, над которыми работают сотрудники компании. При этом должна быть представлена информация об отделениях компании и о её сотрудниках. На основе полученных ранее знаний в программе ErWin Data Modelier была спроектирована следующая схема:

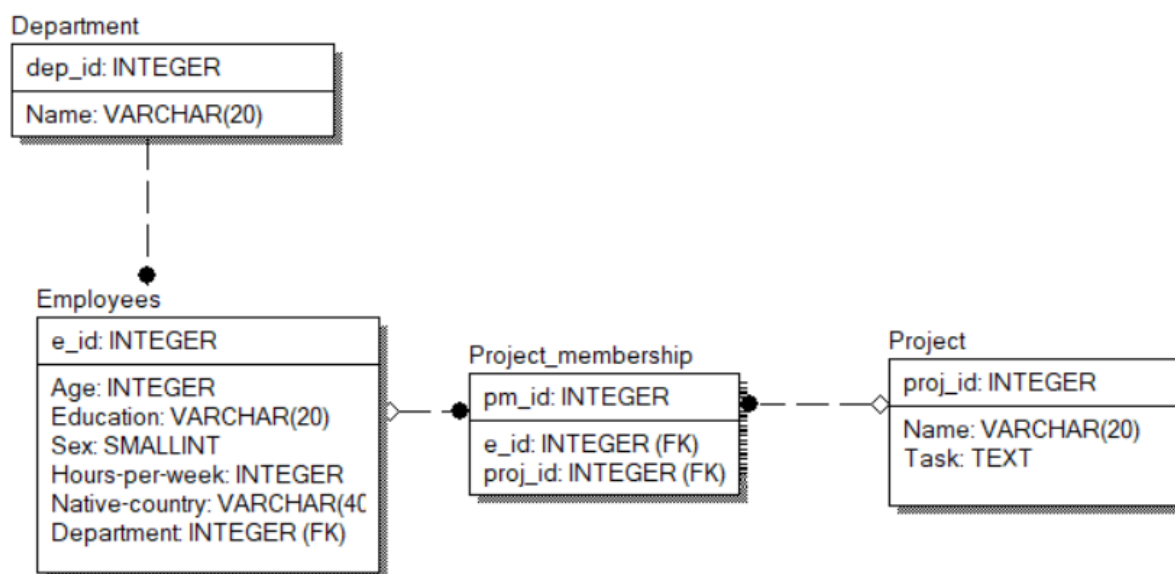


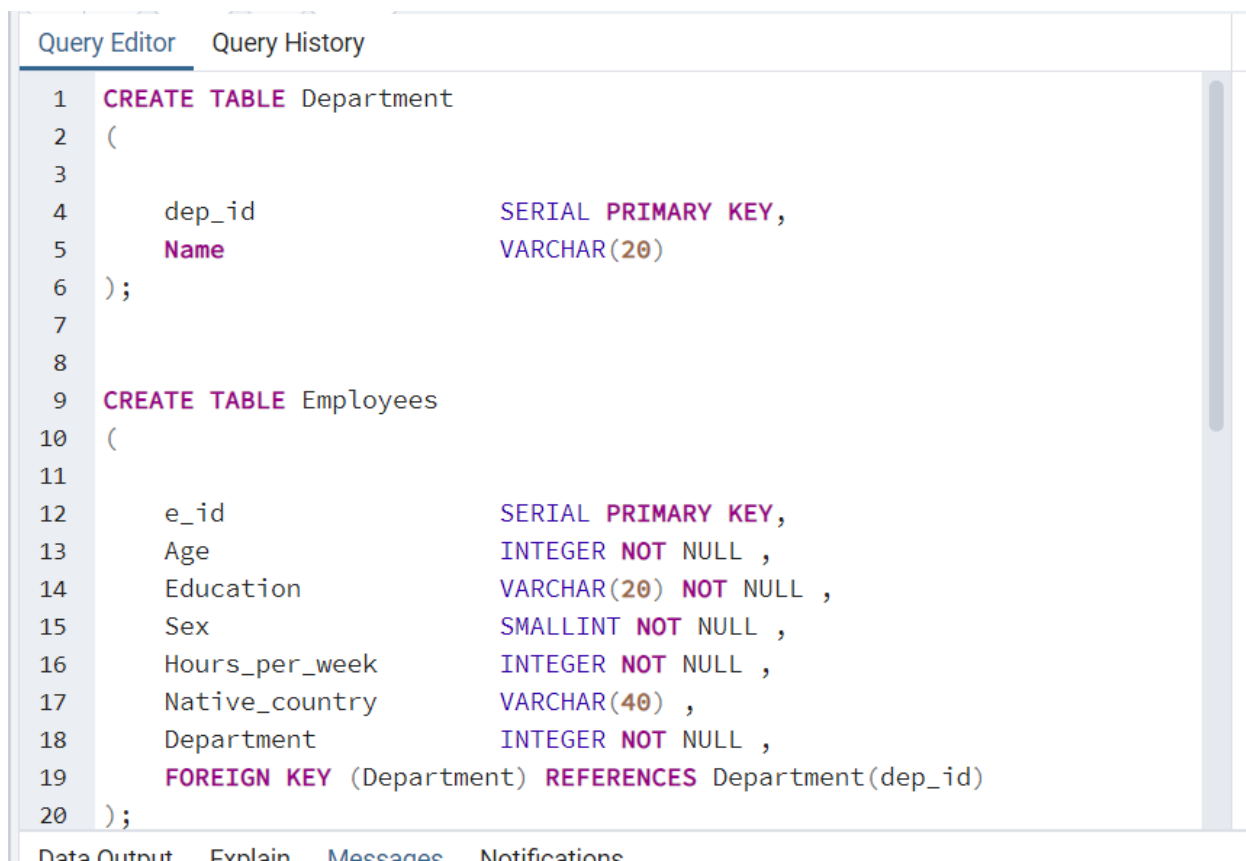
Рис. 2.3.1. Спроектированная схема базы данных

Здесь:

- 1) **Department** – таблица, содержащая данные об отделении предприятия (порядковый номер и название);
- 2) **Employees** – информация о работниках предприятия. Для каждого работника указан его возраст, образование, пол, количество рабочих часов в неделю, гражданство, и отделение, в котором он работает.
- 3) **Project** – информация о проекте (название и его описание);
- 4) **Project_membership** – таблица, реализующая связь «много-ко-многим». Представляет информацию о том, какие сотрудники работают над тем или иным проектом.

2.4 Перенос спроектированной базы данных в PostgreSQL

Был сгенерирован SQL-скрипт для создания соответствующих таблиц, после чего полученный скрипт был выполнен в PostgreSQL. В результате схема была полностью перенесена:



```

1  CREATE TABLE Department
2  (
3
4      dep_id          SERIAL PRIMARY KEY,
5      Name            VARCHAR(20)
6  );
7
8
9  CREATE TABLE Employees
10 (
11
12     e_id             SERIAL PRIMARY KEY,
13     Age              INTEGER NOT NULL ,
14     Education         VARCHAR(20) NOT NULL ,
15     Sex              SMALLINT NOT NULL ,
16     Hours_per_week   INTEGER NOT NULL ,
17     Native_country   VARCHAR(40) ,
18     Department        INTEGER NOT NULL ,
19     FOREIGN KEY (Department) REFERENCES Department(dep_id)
20 );
  
```

Рис. 2.4.1. Скрипт для создания таблиц и связей

В полученную таблицу также были добавлены некоторые данные.

2.5 Реализация ETL-процесса

Далее необходимо реализовать простейший ETL-процесс. Требуется заполнить таблицу с сотрудниками предприятия данными из csv-файла. Файл содержит в себе всю необходимую для базы данных информацию, но в ней также много ненужных данных, а часть данных не приведена к нужному для базы данных формату. Изначально файл имеет следующий вид:

39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K
37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	0	0	40	United-States	<=50K
49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0	0	16	Jamaica	<=50K
52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	45	United-States	>50K
31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female	14084	0	50	United-States	>50K
42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	5178	0	40	United-States	>50K
37	Private	280464	Some-college	10	Married-civ-spouse	Exec-managerial	Husband	Black	Male	0	0	80	United-States	>50K
30	State-gov	141297	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	Asian-Pac-Islander	Male	0	0	40	India	>50K
23	Private	122272	Bachelors	13	Never-married	Adm-clerical	Own-child	White	Female	0	0	30	United-States	<=50K
32	Private	205019	Assoc-acdm	12	Never-married	Sales	Not-in-family	Black	Male	0	0	50	United-States	<=50K
40	Private	121772	Assoc-voc	11	Married-civ-spouse	Craft-repair	Husband	Asian-Pac-Islander	Male	0	0	40	?	>50K
34	Private	245487	7th-8th	4	Married-civ-spouse	Transport-moving	Husband	Amer-Indian-Eskimo	Male	0	0	45	Mexico	<=50K
25	Self-emp-not-inc	176756	HS-grad	9	Never-married	Farming-fishing	Own-child	White	Male	0	0	35	United-States	<=50K
32	Private	186824	HS-grad	9	Never-married	Machine-op-inspct	Unmarried	White	Male	0	0	40	United-States	<=50K
38	Private	28887	11th	7	Married-civ-spouse	Sales	Husband	White	Male	0	0	50	United-States	<=50K

Рис. 2.5.1. Данные в файле .csv

Данные будут открыты, преобразованы и экспортированы с помощью Python. Для работы с файлами будет использоваться библиотека `pandas`, предназначенная для обработки и анализа данных.

Файл был открыт с помощью `pandas`. Поскольку в исходном csv-файле не заданы имена столбцов, они были добавлены с помощью атрибута `names` при чтении:

```
[47] import pandas as pd
db = pd.read_csv('adult.data.csv', names = ["age", "workclass", "fnlwgt", "education",
"education-num", "marital-status", "occupation",
"relationship", "race", "sex", "capital-gain",
"capital-loss", "hours_per_week", "native_country", "salary"])
```

```
[48] db.head()
```

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours_per_week	native_country
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	United-States

Рис. 2.5.2. Чтение файла и полученные данные

Файл содержит в себе много лишней информации. Она была удалена с помощью метода «drop»:

```
db = db.drop(['workclass', 'fnlwgt', 'education-num', 'marital-status', 'occupation',
            'relationship', 'race', 'capital-gain', 'capital-loss', 'salary'], axis = 1)
db.head()
```

	age	education	sex	hours_per_week	native_country
0	39	Bachelors	Male	40	United-States
1	50	Bachelors	Male	13	United-States
2	38	HS-grad	Male	40	United-States
3	53	11th	Male	40	United-States
4	28	Bachelors	Female	40	Cuba

Рис. 2.5.3. Файл после удаления лишних данных

Столбец с информацией о поле работника всё ещё не удовлетворяет установленным требованиям. В базе данных этот параметр представлен bool-значением: «0», если сотрудник – мужчина, или «1», если сотрудник – женщина. Необходимо «закодировать» данный признак. В библиотеке pandas присутствует метод «factorize», после применения которого в файле данные о поле будут представлены числовыми значениями:

```
✓ [50] db['sex'] = pd.factorize(db['sex'])[0]
0 сек. db.head()
```

	age	education	sex	hours_per_week	native_country
0	39	Bachelors	0	40	United-States
1	50	Bachelors	0	13	United-States
2	38	HS-grad	0	40	United-States
3	53	11th	0	40	United-States
4	28	Bachelors	1	40	Cuba

Рис. 2.5.4. Кодирование признака «Пол» и полученные результаты

Видно, что в столбцах с информацией об образовании и гражданстве вместо «пробелов» стоят тире. Обратная замена символов была произведена с помощью метода «str.replace». После этого вызван метод для получения информации о типах данных файла. Видно, что все записи были приведены к требуемому формату:

```
] db['education'] = db['education'].str.replace('-', ' ')
db['native_country'] = db['native_country'].str.replace('-', ' ')
db.head()
```

	age	education	sex	hours_per_week	native_country
0	39	Bachelors	0	40	United States
1	50	Bachelors	0	13	United States
2	38	HS grad	0	40	United States
3	53	11th	0	40	United States
4	28	Bachelors	1	40	Cuba



```
] db.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 5 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   age                   32561 non-null  int64
1   education              32561 non-null  object
2   sex                   32561 non-null  int64
3   hours_per_week        32561 non-null  int64
4   native_country        32561 non-null  object
dtypes: int64(3), object(2)
memory usage: 1.2+ MB
```

Рис. 2.5.5. Замена символов и данные после приведения

Поскольку в базе данных присутствует атрибут «Department», который обозначает отделение, в котором работает сотрудник, а в рассматриваемом csv-файле данная информация отсутствует, к данным был добавлен столбец случайно сгенерированных чисел от «1» до «2». Данные были экспортированы в новый csv-файл:

```
import numpy as np
db['department'] = np.random.randint(1, 3, size = (32561,1))

[79] db.to_csv('employees.csv', sep='\t', encoding='utf-8')
```

Рис. 2.5.6. Добавление столбца и экспорт данных

Полученный файл был импортирован в PostgreSQL с помощью pgAdmin:



Рис. 2.5.7. Импорт файла в PostgreSQL

Файл был успешно импортирован:

Query Editor	Query History	Scratch Pad
<pre>1 SELECT * FROM public.employees 2 ORDER BY e_id ASC LIMIT 100 3</pre>		
Data Output	Explain	Messages
Notifications		
e_id	age	education
[PK] integer	integer	character varying (20)
1	0	39 [...] Bachelors
2	1	50 [...] Bachelors
3	2	38 [...] HS grad
4	3	53 [...] 11th
5	4	28 [...] Bachelors
6	5	37 [...] Masters
7	6	49 [...] 9th
8	7	52 [...] HS grad
9	8	31 [...] Masters
10	9	42 [...] Bachelors
11	10	37 [...] Some college
12	11	30 [...] Bachelors
13	12	23 [...] Bachelors

Рис. 2.5.8. Импортированный файл в PostgreSQL

Заключение

В результате прохождения производственной практики были реализованы основные методы работы с базами данных. В частности, были разработаны SQL-запросы для модификации данных и управления ими, а также был реализован простейший ETL процесс.

Все поставленные задачи были выполнены в полном объеме.

Список использованных источников

- 1) PostgreSQL : Документация. – URL: <https://postgrespro.ru/docs/postgresql> (дата обращения: 21.05.2022)
- 2) pandas - Python Data Analysis Library. – URL: <https://pandas.pydata.org/> (дата обращения: 21.05.2022)