**Министерство науки и высшего образования Российской Федерации**

**ФГАОУ ВО «УрФУ имени первого Президента России Б.Н. Ельцина»**

ОТЧЕТ

Лабораторная работа №1: Основы работы с Docker и PostgreSQL

Студент Ярославцев А. А.

**ФИО студента**

Специальность (направление подготовки) 09.04.02 Информационные системы и технологии

Группа РИМ-150950

Екатеринбург 2025

**Цель работы:**

Освоить фундаментальные концепции и базовые операции Docker: создание образов, запуск контейнеров, управление ими, работа с сетями и томами. На практике закрепить навыки, запустив изолированную базу данных PostgreSQL и подключившись к ней извне.

**Задачи:**

1. Установить и проверить работу Docker.

2. Изучить базовые команды Docker.

3. Запустить контейнер с PostgreSQL в изолированном режиме.

4. Запустить контейнер с pgAdmin и подключить его к контейнеру с БД через сеть Docker.

5. Подключиться к БД из pgAdmin, создать схему и выполнить запросы.

6. Обеспечить сохранность данных БД с помощью томов Docker.

**Часть 0: Установка и проверка Docker**

1. **Установил Docker Desktop** (<https://www.docker.com/products/docker-desktop/>).
2. Обновил WSL
3. Открыл терминал и проверил установку:

Изображение выглядит как текст, электроника, снимок экрана, компьютер

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Часть 1: Базовые команды Docker. Работа с образами и контейнерами**

1. **Просмотр информации:**

**Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Содержимое, созданное искусственным интеллектом, может быть неверным.**

1. **Запуск простого контейнера (на примере Nginx):**



**-d** (detach) — запустить контейнер в фоне.

**-p 8080:80** — проброс порта (хост:контейнер).

**--name** — задать понятное имя контейнеру.

**nginx:alpine** — имя образа и его тег (версия).

Изображение выглядит как текст, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

1. **Проверил, что контейнер работает:** Открыл в браузере **http://localhost:8080**. Увидел стартовую страницу Nginx.
2. **Остановил и удалил контейнер:**

Изображение выглядит как текст, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

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

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Часть 2: Запуск PostgreSQL в контейнере**

1. **Запустил контейнер с PostgreSQL:**

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Содержимое, созданное искусственным интеллектом, может быть неверным.

**-e** — установка переменных окружения внутри контейнера.

**-p** – проброс портов для работы с БД внешних систем

Переменные **POSTGRES\_\*** необходимы для первоначальной настройки СУБД.

1. **Проверил, что контейнер запущен и слушает порт:**



1. **Подключился к БД прямо из контейнера (через psql):**



Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.

Оказался в консоли PostgreSQL. Выполнил тестовые запросы:

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

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

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Часть 3: Подключение к БД через pgAdmin из второго контейнера**

1. **Создал сеть Docker:** Контейнеры по умолчанию изолированы. Чтобы они "увидели" друг друга по именам, нужна общая сеть.



1. **Подключил контейнер с PostgreSQL к сети:**



1. **Запустил pgAdmin в той же сети:**

Изображение выглядит как текст, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Ключевой параметр --network my\_network помещает этот контейнер в ту же сеть, что и БД.

4. **Настроил подключение в pgAdmin:**

Открыл **http://localhost:8080** в браузере.

Вошёл под **admin@example.com** / **admin**.

**Добавил новый сервер:**

**General -> Name:** **Docker PostgreSQL**

**Connection -> Host name/address:** **my\_postgres\_db**

**Connection -> Username:** **postgres**

**Connection -> Password:** **postgres**

Сохранил. Подключение прошло успешно.

**Через Query Tool в pgAdmin выполнил запрос:**



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

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Часть 4: Сохранение данных с помощью Томов (Volumes)**

1. **Остановил и удалил текущий контейнер с БД:**

*Все данные, созданные внутри контейнера, будут безвозвратно утеряны.*

1. **Создал том для хранения данных БД:**



1. **Запустил новый контейнер с PostgreSQL, подключив том:**

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Проверил сохранность данных:**

1. Через pgAdmin ([**http://localhost:8080**](http://localhost:8080)) и через PowerShell снова выполнил запрос **SELECT \* FROM users;** добавил таблицу и данные
2. Остановил контейнеры с БД и pgAdmin
3. Запустил контейнеры 
4. Проверил таблицу users

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

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Ответы на вопросы:**

1. Что такое docker?

— это платформа для контейнеризации приложений. Если говорить простыми словами, это такая "виртуализация на уровне операционной системы", которая позволяет упаковать приложение со всеми его зависимостями (библиотеками, настройками, переменными окружения) в изолированный контейнер.

На практике это значит, что я могу разработать приложение на своем ноутбуке, упаковать его в контейнер, и быть уверенным, что оно точно так же заработает на сервере или у другого разработчика, без проблем с версиями ПО или настройками окружения.

1. Для чего нужны тома и сети docker?

Тома — это механизм для сохранения данных. Когда контейнер удаляется, все данные в нем теряются. Тома позволяют сохранить данные (например, базу данных, файлы приложения) вне контейнера.

Сети — обеспечивают взаимодействие между контейнерами. По умолчанию контейнеры изолированы, но через сети они могут "видеть" друг друга по именам.

1. Как подключиться к контейнеру и выполнить в нём команды?

# Для интерактивной работы в контейнере

docker exec -it имя\_контейнера bash

# Для выполнения конкретной команды

docker exec имя\_контейнера команда

# Например, для работы с PostgreSQL

docker exec -it my\_postgres\_db psql -U postgres -d test\_db

1. Для чего нужен pgAdmin?

— это веб-интерфейс для управления PostgreSQL. Вместо того чтобы работать с БД только через командную строку (psql), pgAdmin предоставляет удобный графический интерфейс где можно:

* Просматривать структуру БД (таблицы, индексы, процедуры)
* Выполнять SQL-запросы в удобном редакторе
* Анализировать производительность
* Управлять пользователями и правами
* Создавать резервные копии

**Вывод:**

В ходе работы успешно освоены основы контейнеризации приложений с помощью Docker. Настроена связка PostgreSQL и pgAdmin, обеспечено их взаимодействие через Docker-сеть и сохранение данных с помощью томов. Автоматизировано развертывание всей системы через Docker Compose.