

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики
Кафедра технической кибернетики

Отчет по лабораторной работе №1

Дисциплина: «Инженерия данных»

Выполнил: Осепян С.М.

Группа: 6233-010402D

Самара 2025

Цели работы

- На практике освоить полный цикл ETL: извлечение из публичного API, трансформация и загрузка в ClickHouse.
- Научиться собирать и отлаживать пайплайны в **Prefect**.
- Научиться настраивать окружение для работы с данными.

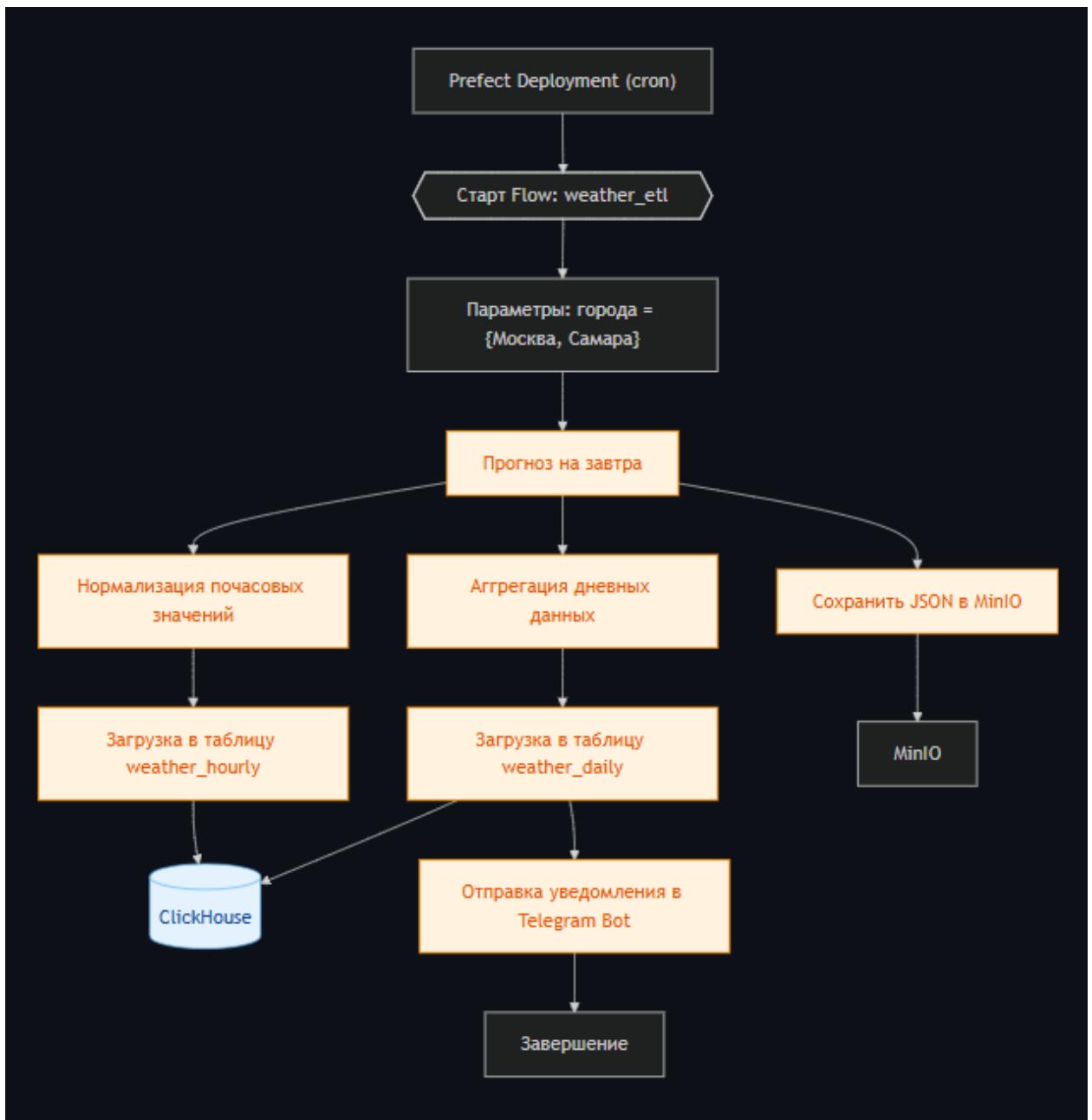


Рисунок 1 – схема работы пайплайна.

1. В качестве источника данных предлагается использовать [Free Weather API](#)
2. (Extract) Получить прогноз погоды **на завтра** по переменным: *температура, осадки, скорость и направление ветра*, для городов **Самара и Москва**.
Сырые ответы API сохранить в объектном хранилище
3. (Transform)

- Извлечь почасовые значения и нормализовать для таблицы weather_hourly
 - Посчитать дневную статистику (min, max, avg температуру и количество осадков) и подготовить для сохранения в таблице weather_daily
4. (Load) Загрузить преобразованные данные в соответствующие таблицы ClickHouse
 5. Автоматически отправить уведомления в Telegram с кратким прогнозом на завтра и предупреждать о сильном ветре/осадках
 6. **(Опционально)** Реализовать обработку различных ошибок.

Создадим бота в телеграмме с помощью @BotFather

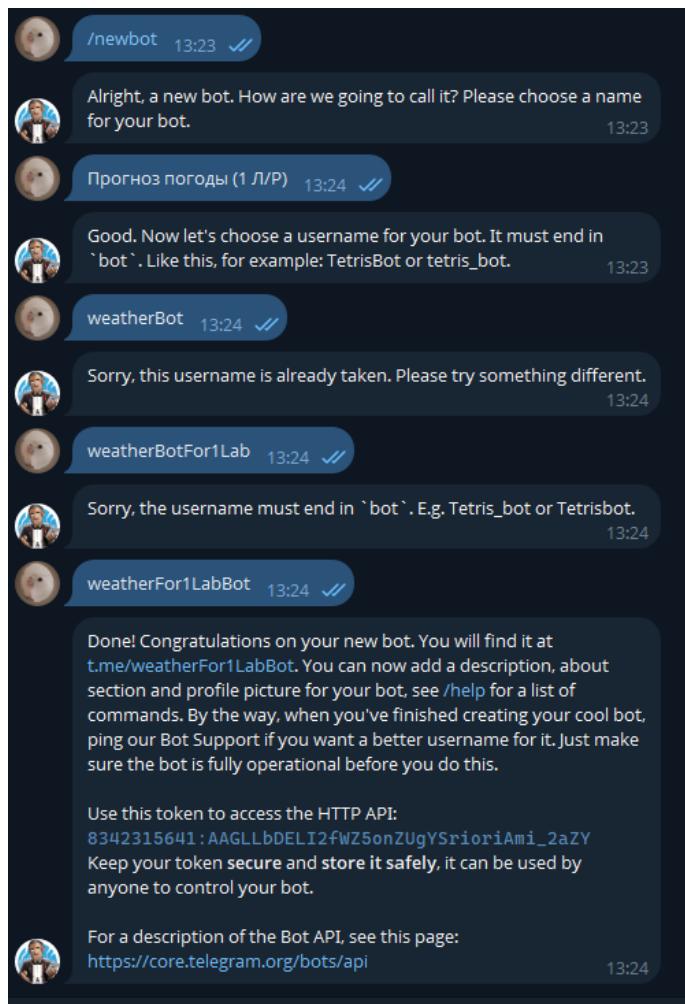


Рисунок 2 – Создание бота в телеграмме.

В результате чего BotFather дал нам токен для нашего бота, который мы будем использовать в будущем, для того, чтобы созданный нами бот мог отправлять сообщения, создадим чат в телеграмме, добавив туда нашего бота, указав его тэг при добавлении участников



Рисунок 3 – Добавление бота в чат телеграмма.

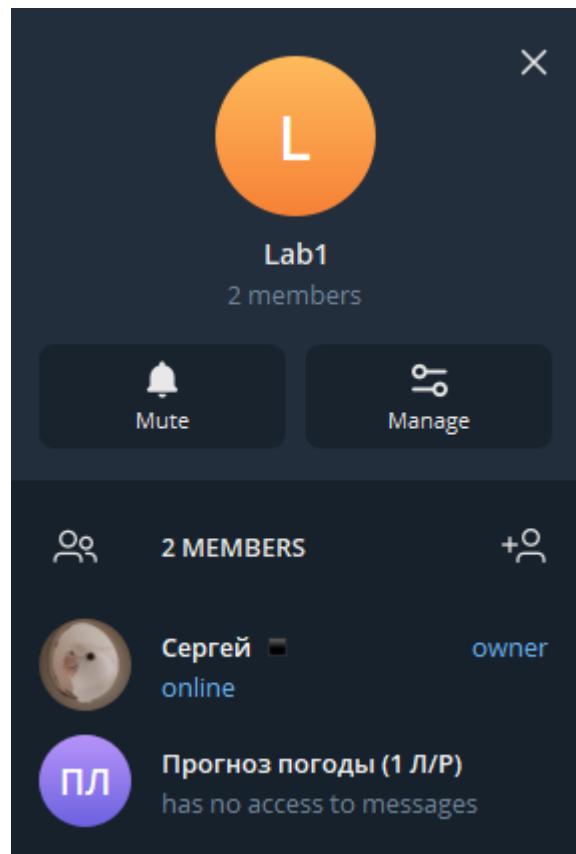


Рисунок 4 – Созданный чат в телеграмме.

Также нам необходимо знать ID чата, куда бот должен отправлять сообщение, для этого зайдем в бота @webhelpiebot, в нем нажимаем на кнопку “чаты”,

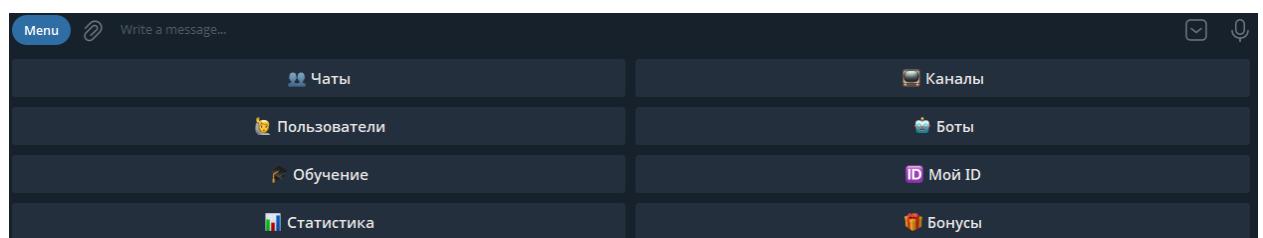


Рисунок 5 – Панель бота @webhelpiebot

И выбираем созданный нами чат, в результате чего бот покажет нам ID чата.

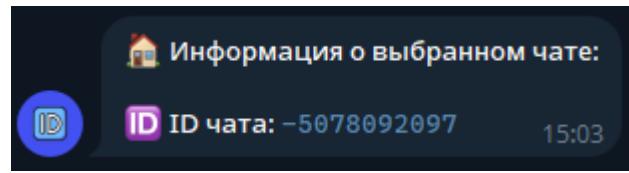


Рисунок 6 – ID нужного чата.

Таким образом мы завершили настройку телеграм бота, который будет отправлять нам уведомления.

Написав код, настроив docker-compose запустим его командой docker-compose up -d build.

```
PS C:\paring\Lab-1-2025\weather-etl-pipeline> docker-compose up -d --build
time="2025-12-05T15:27:26+04:00" level=warning msg="C:\paring\Lab-1-2025\weather-etl-pipeline\docker-compose.yaml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 6/6
  ✓ Network weather-etl-pipeline_etc-network   Created
  ✓ Container jupyter                         Started      0.1s
  ✓ Container minio                           Started      0.9s
  ✓ Container prefect-server                  Started      0.9s
  ✓ Container clickhouse                      Started      0.9s
  ✓ Container prefect-agent                  Started      1.0s
  ✓ Container prefect-agent                  Started      1.0s
PS C:\paring\Lab-1-2025\weather-etl-pipeline>
```

Рисунок 7 – Запуск контейнера

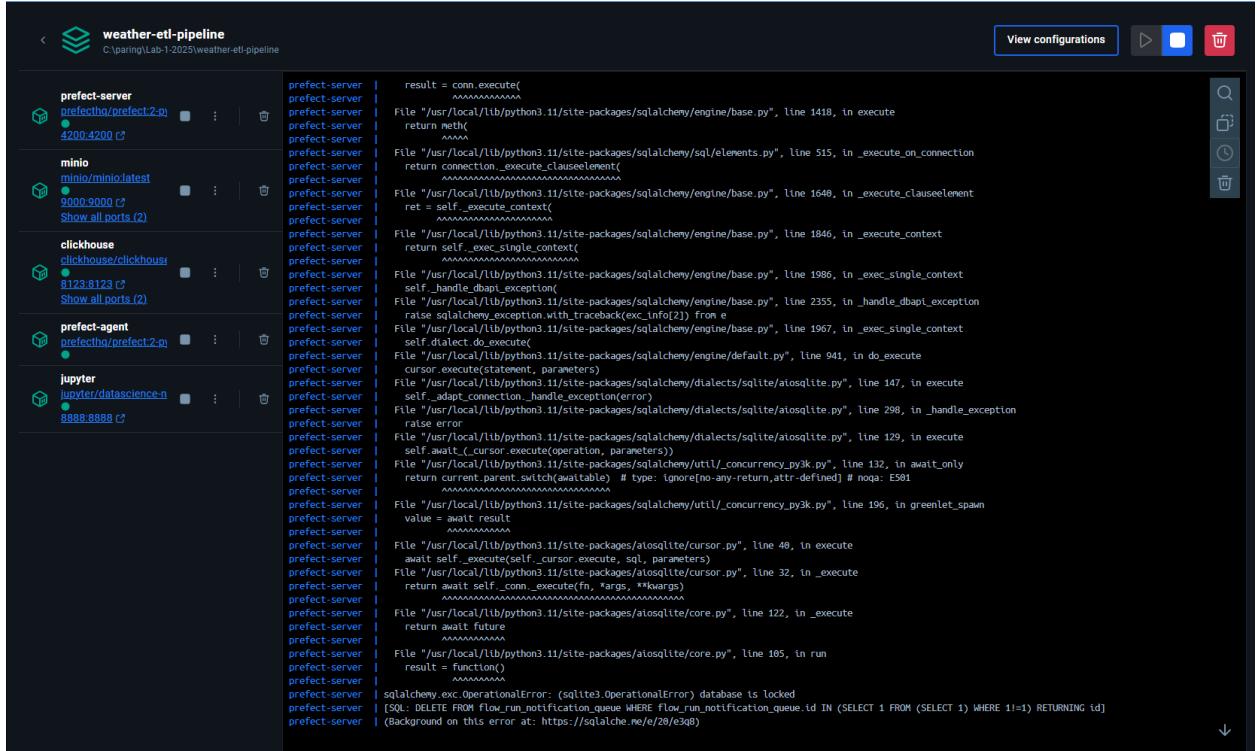


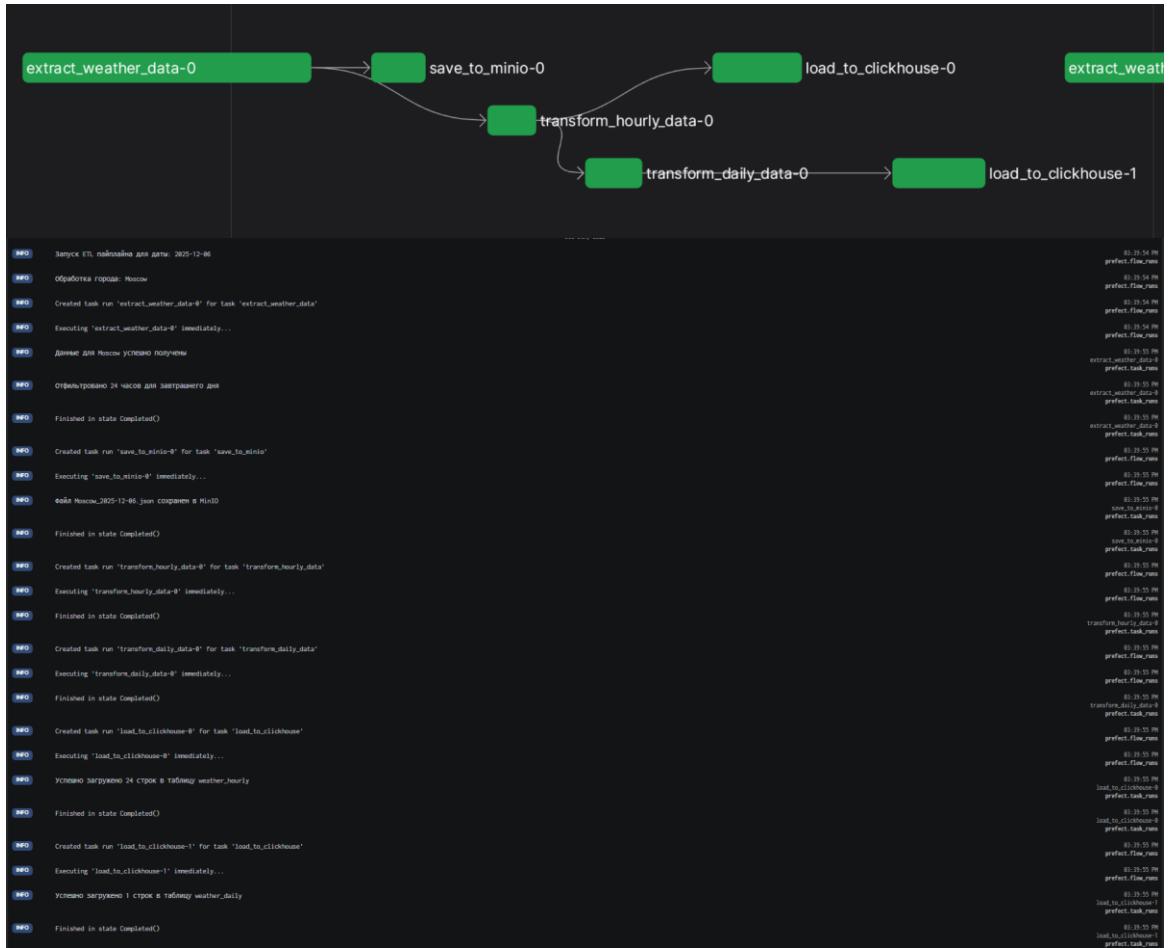
Рисунок 8 – Запущенные сервера

Программа завершилась успешно, в телеграм пришло уведомление



Рисунок 10 – Уведомление от бота

Зайдем во flow-runs в Prefect, там появилась запись `weather_etl`, в ней можно увидеть следующие логи:



```

[INFO] Город Москва успешно обработан
[INFO] Обработка города: Самара
[INFO] Created task run 'extract_weather_data-1' for task 'extract_weather_data'
[INFO] Executing 'extract_weather_data-1' immediately...
[INFO] Данные для Самары успешно получены
[INFO] Отфильтровано 24 часа для заглавленного дня
[INFO] Finished in state Completed()

[INFO] Created task run 'save_to_minio-1' for task 'save_to_minio'
[INFO] Executing 'save_to_minio-1' immediately...
[INFO] файл Samara_2025-12-06.json сохранен в MinIO
[INFO] Finished in state Completed()

[INFO] Created task run 'transform_hourly_data-1' for task 'transform_hourly_data'
[INFO] Executing 'transform_hourly_data-1' immediately...
[INFO] Finished in state Completed()

[INFO] Created task run 'transform_daily_data-1' for task 'transform_daily_data'
[INFO] Executing 'transform_daily_data-1' immediately...
[INFO] Finished in state Completed()

[INFO] Created task run 'load_to_clickhouse-2' for task 'load_to_clickhouse'
[INFO] Executing 'load_to_clickhouse-2' immediately...
[INFO] Успешно загружено 24 строк в таблицу weather_hourly
[INFO] Finished in state Completed()

[INFO] Created task run 'load_to_clickhouse-3' for task 'load_to_clickhouse'
[INFO] Executing 'load_to_clickhouse-3' immediately...
[INFO] Успешно загружено 1 строк в таблицу weather_daily
[INFO] Finished in state Completed()

[INFO] Город Самара успешно обработан
[INFO] Уведомление отправлено в Telegram
[INFO] ETL майнайн завершен
[INFO] Finished in state Completed('All states completed.')

```

Рисунок 11 – Логи Prefect

В minIO также появились бакет weather-raw



Рисунок 12 – Содержимое weather-raw в minIO

И также зайдем в clickHouse, и просмотрим содержимое таблицы

PS C:\Users\osepa> Invoke-RestMethod -Uri "http://localhost:8123/?query=SELECT * FROM weather_db.weather_hourly&user=admin&password=password"									
Samara	2025-12-06	0	0.8	0	5.1	188	2025-12-05	11:39:56	
Samara	2025-12-06	1	0.8	0	4.7	203	2025-12-05	11:39:56	
Samara	2025-12-06	2	0.7	0	4.7	189	2025-12-05	11:39:56	
Samara	2025-12-06	3	0.8	0	5.1	188	2025-12-05	11:39:56	
Samara	2025-12-06	4	0.6	0.1	5.5	212	2025-12-05	11:39:56	
Samara	2025-12-06	5	0.7	0	5.7	235	2025-12-05	11:39:56	
Samara	2025-12-06	6	0.8	0	6.2	260	2025-12-05	11:39:56	
Samara	2025-12-06	7	0.7	0	5.9	259	2025-12-05	11:39:56	
Samara	2025-12-06	8	0.5	0	6.4	243	2025-12-05	11:39:56	
Samara	2025-12-06	9	0.5	0.1	6.6	248	2025-12-05	11:39:56	
Samara	2025-12-06	10	0.8	0	7.8	257	2025-12-05	11:39:56	
Samara	2025-12-06	11	0.8	0	8.3	270	2025-12-05	11:39:56	
Samara	2025-12-06	12	1	0	7.2	264	2025-12-05	11:39:56	
Samara	2025-12-06	13	1	0	7.3	261	2025-12-05	11:39:56	
Samara	2025-12-06	14	0.9	0.1	7.2	264	2025-12-05	11:39:56	
Samara	2025-12-06	15	0.8	0.1	6.6	257	2025-12-05	11:39:56	
Samara	2025-12-06	16	0.5	0.1	5.6	255	2025-12-05	11:39:56	
Samara	2025-12-06	17	0.4	0.1	5.2	258	2025-12-05	11:39:56	
Samara	2025-12-06	18	0.5	0	5.1	266	2025-12-05	11:39:56	
Samara	2025-12-06	19	0.5	0	5	270	2025-12-05	11:39:56	
Samara	2025-12-06	20	0.3	0.1	4.7	266	2025-12-05	11:39:56	
Samara	2025-12-06	21	0.4	0.1	4.3	265	2025-12-05	11:39:56	
Samara	2025-12-06	22	0.6	0	4.7	261	2025-12-05	11:39:56	
Samara	2025-12-06	23	0.6	0	4.7	274	2025-12-05	11:39:56	
Moscow	2025-12-06	0	2.2	0	3.1	234	2025-12-05	11:39:55	
Moscow	2025-12-06	1	2.2	0	2.9	210	2025-12-05	11:39:55	
Moscow	2025-12-06	2	2.2	0	3.9	214	2025-12-05	11:39:55	
Moscow	2025-12-06	3	2.3	0	4	207	2025-12-05	11:39:55	
Moscow	2025-12-06	4	2.2	0	4.4	215	2025-12-05	11:39:55	
Moscow	2025-12-06	5	2.2	0	4.6	225	2025-12-05	11:39:55	
Moscow	2025-12-06	6	2.2	0	4.4	215	2025-12-05	11:39:55	
Moscow	2025-12-06	7	2.2	0	4.4	215	2025-12-05	11:39:55	
Moscow	2025-12-06	8	2.1	0	4.2	211	2025-12-05	11:39:55	
Moscow	2025-12-06	9	2	0	4.5	209	2025-12-05	11:39:55	
Moscow	2025-12-06	10	2.1	0	4.6	219	2025-12-05	11:39:55	
Moscow	2025-12-06	11	2.3	0	4.6	219	2025-12-05	11:39:55	
Moscow	2025-12-06	12	2.6	0	5.2	214	2025-12-05	11:39:55	
Moscow	2025-12-06	13	2.8	0	4.8	222	2025-12-05	11:39:55	
Moscow	2025-12-06	14	2.9	0	5.1	219	2025-12-05	11:39:55	
Moscow	2025-12-06	15	2.9	0.2	4.8	222	2025-12-05	11:39:55	
Moscow	2025-12-06	16	2.7	0.6	4.7	203	2025-12-05	11:39:55	
Moscow	2025-12-06	17	2.6	0	5	210	2025-12-05	11:39:55	

Рисунок 13 – Содержимое таблицы weather_hourly

Выводы

В ходе выполнения лабораторной работы был успешно реализован и протестирован полный ETL-пайплайн для сбора и обработки данных о погоде.

Была построена ETL-архитектура с использованием инструментов:

1. Prefect 2.0 для оркестрации и управления workflow
2. Open-Meteo API в качестве источника данных
3. MinIO для хранения сырых JSON-данных
4. ClickHouse как аналитическая колоночная СУБД
5. Telegram Bot API для уведомлений

Трудности: Изначально возникали ошибки с корутинаами при отправке сообщений в Telegram, связанные с асинхронностью Telegram Bot API

Что можно добавить:

- Возможность выбора городов с помощью телеграм бота
- Создать визуализацию на основе накопленных данных