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

ОТЧЁТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
«Введение в ETL пайплайны. Знакомство с Prefect и ClickHouse»

по дисциплине
«Инженерия данных»

Студент _____ А.Е. Федякин

Преподаватель _____ Р. А. Паринге

Самара 2025

В данной работе был реализован автоматизированный ETL-пайплайн для получения, обработки и хранения прогноза погоды на завтра по городам Москва и Самара.

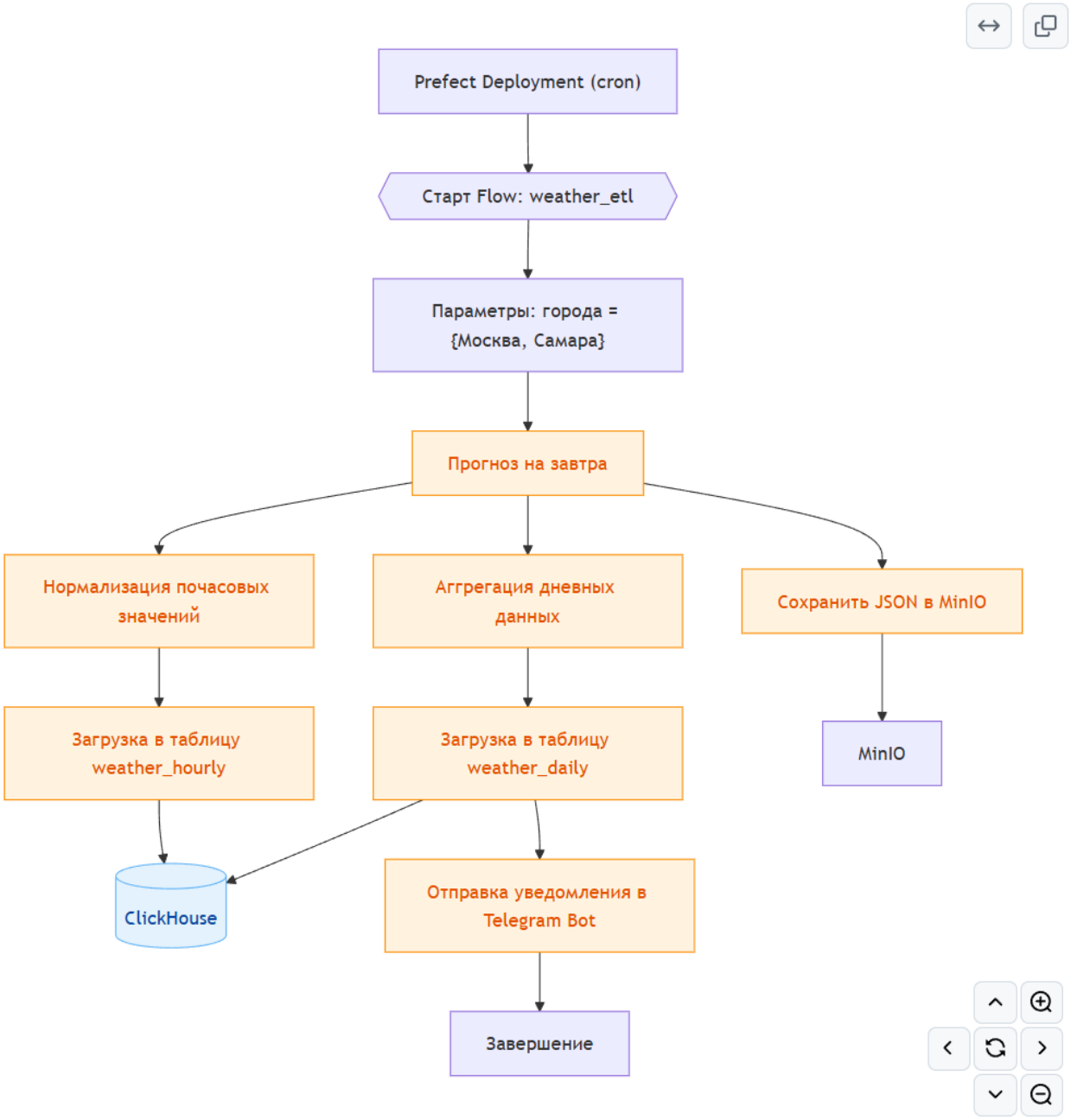

Рисунок 1 – Схема пайплайна

Архитектура включает следующие компоненты:

Prefect — оркестратор задач, обеспечивающий выполнение по расписанию, повторные попытки при ошибках и визуализацию пайплайна;

Open-Meteo API — бесплатный источник погодных данных;

MinIO — объектное хранилище для архивации сырых JSON-ответов API (обеспечивает воспроизводимость и отладку);

ClickHouse — колоночная СУБД для эффективного хранения и анализа почасовых и дневных агрегатов;

Telegram Bot — канал уведомлений о завершении загрузки и аномальных погодных условиях.

Выбор инструментов обусловлен их популярностью в современных data-инженерных стеках и лёгкостью развёртывания в Docker.

Для источника данных, как было указано раннее использовалось Open-Meteo API с эндпоинтом *https://api.open-meteo.com/v1/forecast*.

В запросе применялись следующие параметры:

- *latitude, longitude* — координаты городов (Москва: 55.7522, 37.6156; Самара: 53.1959, 50.1001);
- *hourly=temperature_2m,precipitation,windspeed_10m,winddirection_10m*;
- *timezone=Europe/Moscow*;
- *start_date* и *end_date* — дата завтрашнего дня.

В лабораторной работе были реализованы 3 блока: Extract, Transform, Load.

С помощью Prefect-задачи выполняется HTTP-запрос к Open-Meteo API для каждого города. Сырые JSON-ответы сохраняются в MinIO с именем weather_{город}_{дата}.

Реализованы две ветки трансформации:

1) Почасовые данные нормализуются в плоские записи с полями city, timestamp (как DateTime), temperature, precipitation, windspeed, winddirection.

2) Дневные агрегаты рассчитываются как min/max/avg температуры, суммарные осадки и максимальная скорость ветра.

После этого данные загружаются в две таблицы ClickHouse:

1) weather_hourly — почасовые замеры;

2) weather_daily — дневные агрегаты.

Загрузка выполняется через clickhouse-connect с предварительным созданием базы и таблиц при первом запуске.

Также в работе были реализованы следующие проверки:

- Валидация структуры ответа API (проверка наличия ключей hourly.time и др.);
- Замена None на дефолтные значения (0.0) для числовых полей;
- Преобразование строковых дат в объекты datetime/date для корректной сериализации.\

Также с помощью команды:

```
prefect deploy main.py:weather_etl --name "daily-weather" --cron "0 6 * * *" --
pool "lab-pool"
```

можно запустить деплой пайплайна, с помощью которого он будет выполняться каждый день в определенное время.

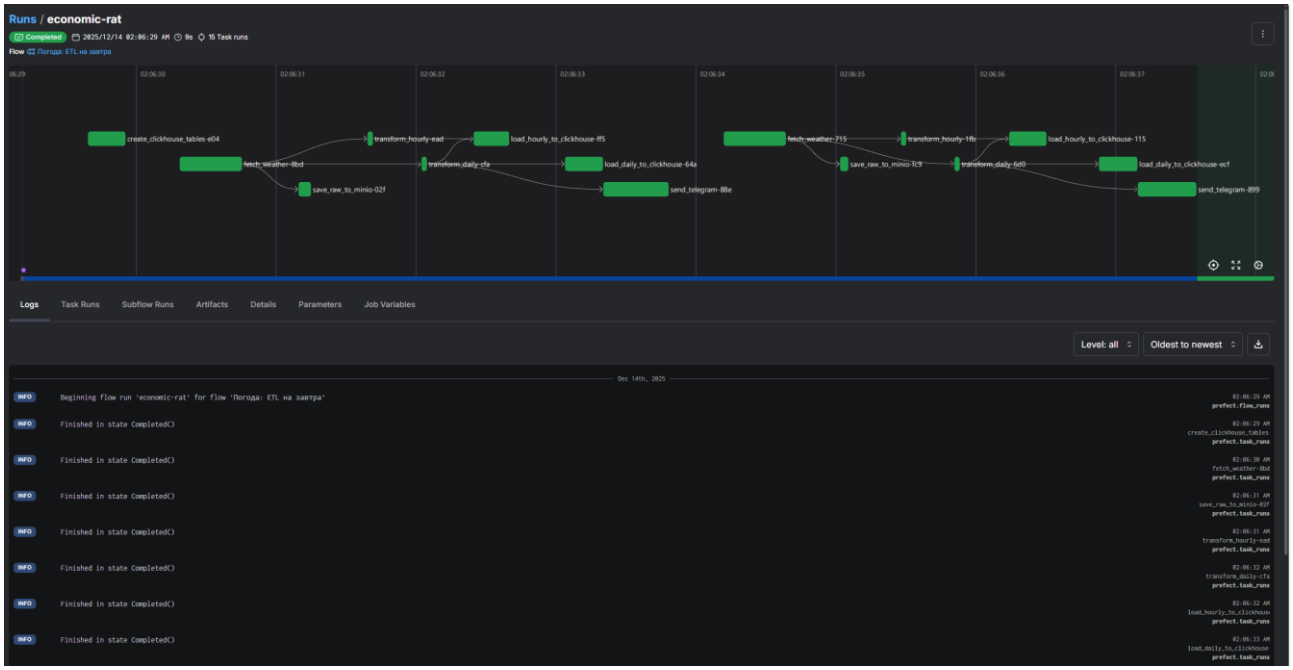

Рисунок 2 – Демонстрация логов Prefect

Пример содержимого бакета в MinIO:

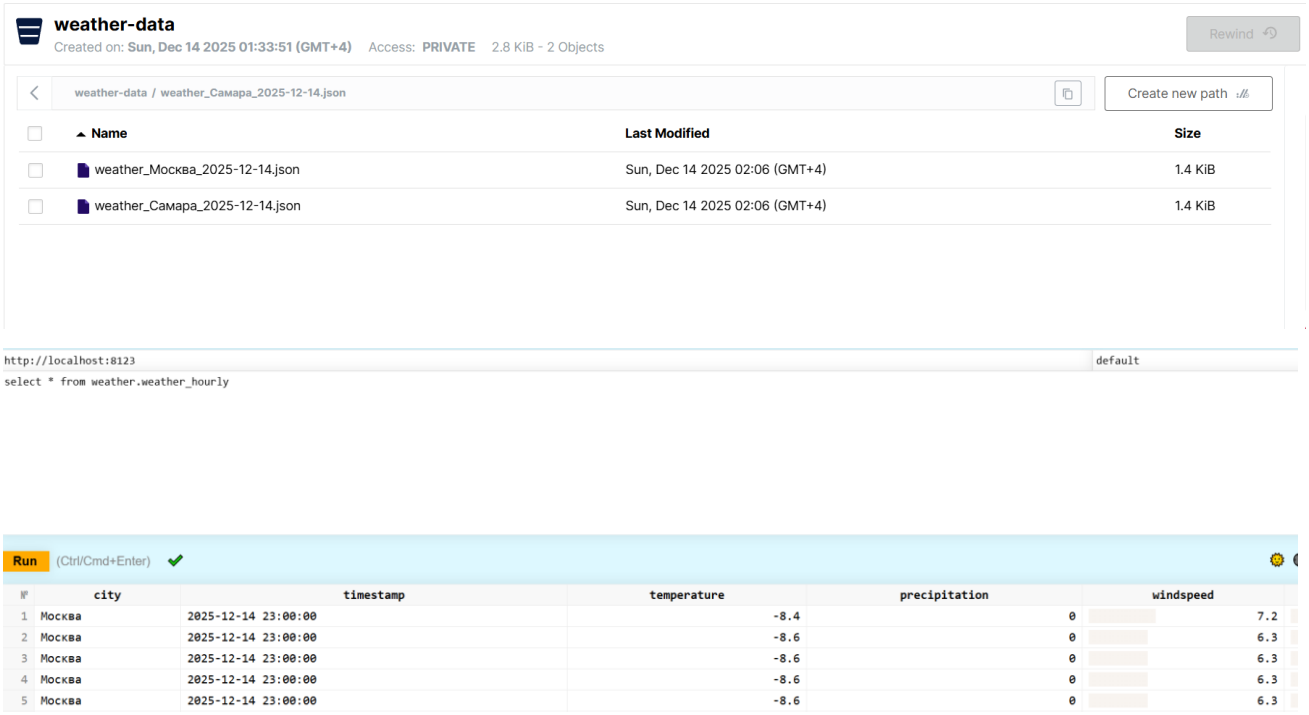

Рисунок 3 – Запрос в ClickHouse для weather_hourly

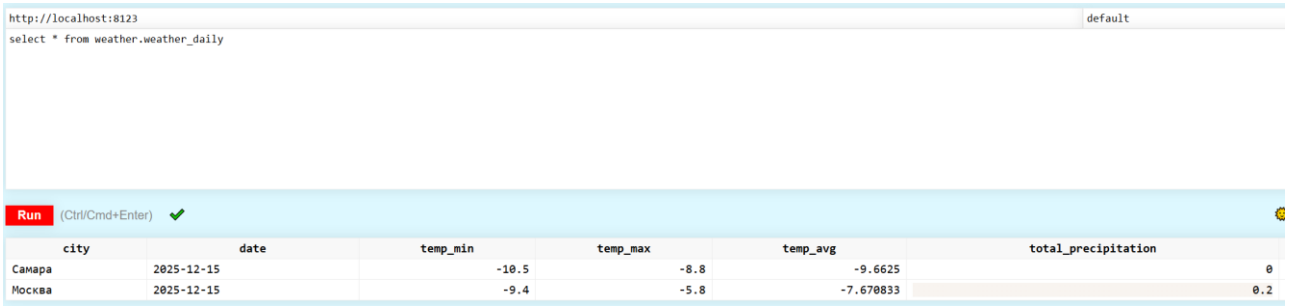

Рисунок 4 – Запрос в ClickHouse для weather_daily

Фрагмент примера json:


```
{
  "latitude": 53.1875,
  "longitude": 50.125,
  "generationtime_ms": 0.04315376281738281,
  "utc_offset_seconds": 10800,
  "timezone": "Europe/Moscow",
  "timezone_abbreviation": "GMT+3",
  "elevation": 80,
  "hourly_units": {
    "time": "iso8601",
    "temperature_2m": "°C",
    "precipitation": "mm",
    "windspeed_10m": "km/h",
    "winddirection_10m": "°"
  },
  "hourly": {
    "time": [
      "2025-12-15T00:00",
      "2025-12-15T01:00",
      …
    ],
    "temperature_2m": [
      -10.5,
      -10.4
      …
    ],
    "precipitation": [
      0,
      0,
      …
    ],
    "windspeed_10m": [
      18.4,
      17.6,
      …
    ],
    "winddirection_10m": [
      298,
      293,
      …
    ]
  }
}
```

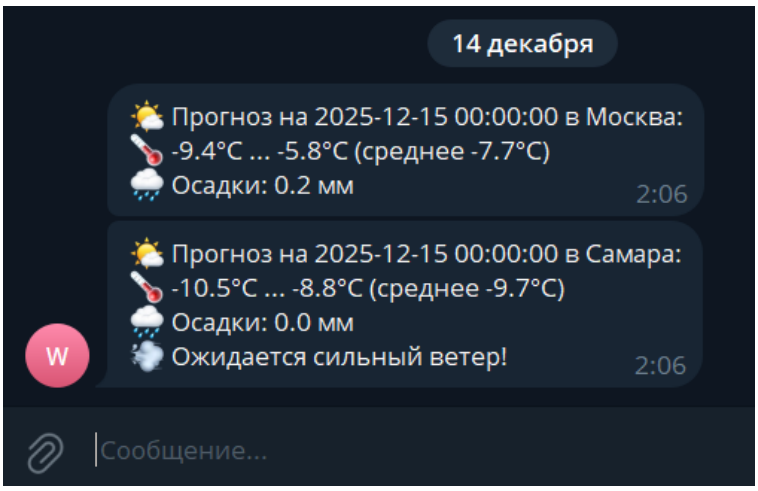

Рисунок 5 – Демонстрация уведомления в Telegram

Из сложностей в данной лабораторной работе было разобраться с конвертацией типов данных после их обработки при сохранении в ClickHouse. Улучшить в данной работе можно обработку ошибок и инициализацию базы данных.