Министерство науки и высшего образования
Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»

Институт информатики, математики и электроники

Факультет информатики
Кафедра технической кибернетики

ETL ПАЙПЛАЙН ДЛЯ ПОГОДНЫХ ДАННЫХ

Лабораторная работа № 1

по курсу «Инженерия данных»

Выполнил: Кречко В.Н.

Группа: 6233-010402D

САМАРА
2025

1. Архитектура системы

Реализованный ETL пайплайн представляет собой автоматизированную систему сбора,
обработки и хранения метеорологических данных. Архитектура построена на микросервисном
подходе с использованием контейнеризации Docker.

Ключевые компоненты:

• **Prefect 3.0** - оркестратор workflow, выбран за современный Python-friendly API,
встроенную систему retry и удобный UI для мониторинга
• **ClickHouse** - колоночная СУБД для аналитических запросов, оптимальна для временных рядов благодаря партиционированию по датам
• **MinIO** - S3-совместимое объектное хранилище для сырых JSON данных, обеспечивает
аудит и возможность повторной обработки
• **Open-Meteo API** - бесплатный источник погодных данных без необходимости
регистрации, высокая точность прогнозов

Пайплайн запускается автоматически по расписанию (cron), извлекает прогноз на завтра для Москвы и Самары, сохраняет сырые данные, трансформирует их в две структуры (почасовые и дневные агрегаты), загружает в ClickHouse и опционально отправляет уведомления в Telegram.

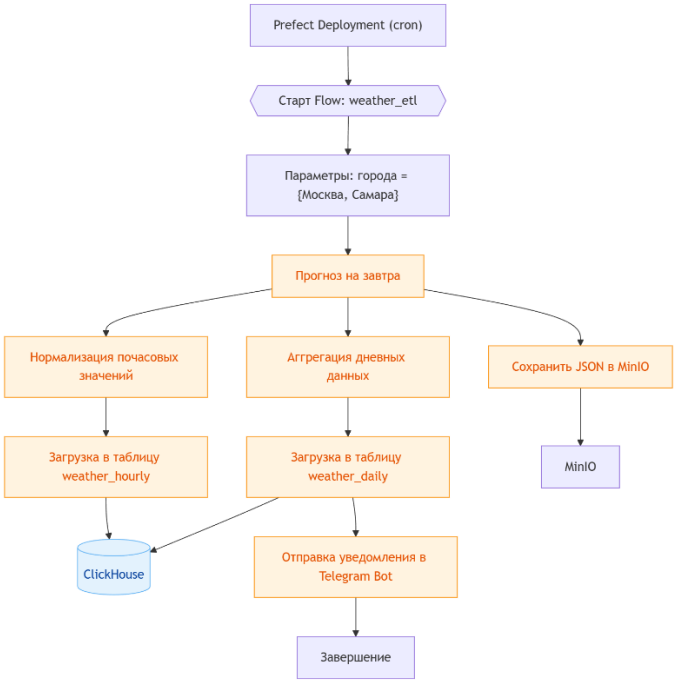

Рисунок 1 - Диаграмма архитектуры

Рисунок 2 - Docker Compose структура

2. Источник данных

Источником данных выступает Open-Meteo API (https://api.open-meteo.com/v1/forecast),
предоставляющий бесплатный доступ к метеорологическим прогнозам.

Параметры запроса:

• **latitude, longitude** - координаты городов (Москва: 55.7558, 37.6173; Самара: 53.1959, 50.1002)
• **hourly** - список переменных: temperature_2m, precipitation, wind_speed_10m, wind_direction_10m
• **start_date, end_date** - дата завтрашнего дня для получения прогноза
• **timezone** - автоматическое определение часового пояса

API возвращает JSON с почасовыми значениями всех запрошенных параметров.
Преимущества: отсутствие лимитов на запросы, актуальные данные, простой REST API.

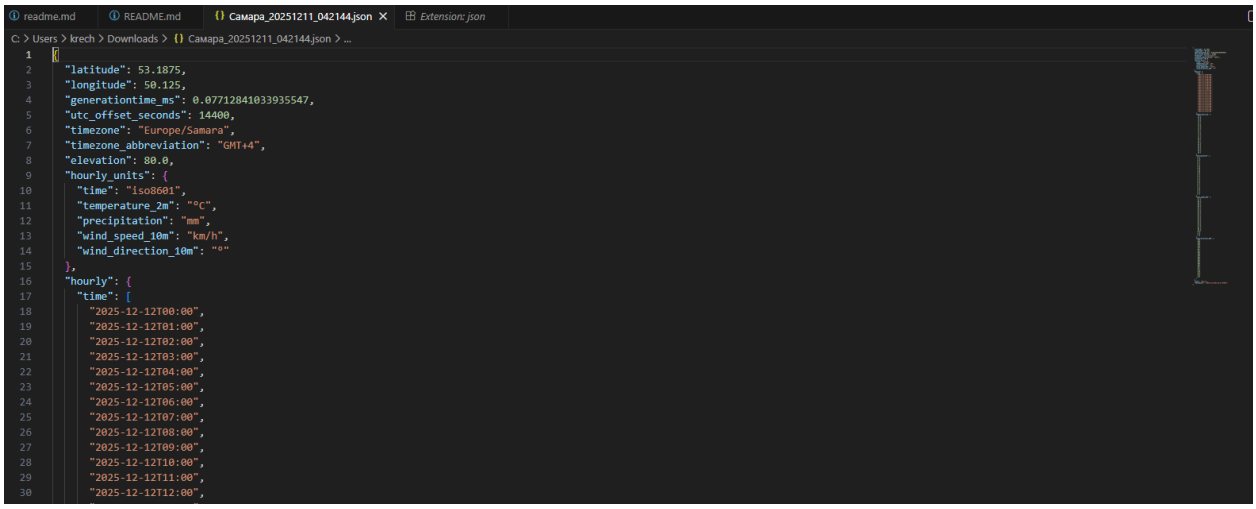

Рисунок 3 - Пример JSON ответа от API

3. Extract этап

Этап Extract реализован в модуле tasks/extract.py с использованием декоратора @task из Prefect. Функция fetch_weather_forecast() выполняет HTTP GET запрос к API Open-Meteo с параметрами для каждого города. Реализован механизм retry с тремя попытками и задержкой 10 секунд при сетевых сбоях. Сырые JSON-ответы сохраняются в MinIO через функцию save_to_minio() для обеспечения audit trail и возможности повторной обработки. Имена файлов содержат город и timestamp для уникальности.

4. Transform этап

Transform этап включает две функции в tasks/transform.py:

1. **transform_to_hourly()** - нормализует вложенную JSON структуру в плоский DataFrame с полями: city, timestamp, temperature, precipitation, wind_speed, wind_direction. Обрабатывает 24 записи (по одной на каждый час завтрашнего дня) для каждого города.

2. **transform_to_daily()** - агрегирует почасовые данные в дневную статистику: вычисляет min, max, avg температуры и сумму осадков. Результат - одна запись на город с полями: city, date, temp_min, temp_max, temp_avg, total_precipitation.

Обе функции используют pandas для эффективной обработки данных и включают логирование через get_run_logger() для отладки.

5. Load этап

Load этап реализован в tasks/load.py через clickhouse-connect библиотеку. Функции load_to_clickhouse_hourly() и load_to_clickhouse_daily() используют метод insert_df() для массовой загрузки DataFrame в ClickHouse с автоматической компрессией. Таблицы партиционированы по месяцам (PARTITION BY toYYYYMM) для оптимизации запросов и управления данными. Движок MergeTree обеспечивает эффективную работу с временными рядами. Реализован retry механизм с тремя попытками для устойчивости к временным проблемам с БД.

6. Качество данных и обработка ошибок

Реализованные проверки качества данных (tasks/validate.py):

1. Проверка на пустоту DataFrame - предотвращает загрузку пустых наборов данных
2. Выявление пропущенных значений - логирует количество NULL в каждом столбце

3. Валидация диапазонов температуры (-90°C до +60°C) - отсекает аномальные значения
4. Обнаружение дубликатов по ключам (city, timestamp/date) - предотвращает дублирование данных в БД

Механизмы обработки ошибок:
• Retry стратегии: 3 попытки для API запросов, 2 попытки для БД операций
• Экспоненциальная задержка между retry для предотвращения перегрузки
• Детальное логирование через Prefect logger для анализа сбоев
• Graceful degradation: при отказе Telegram уведомлений пайплайн продолжает работу

Возможные точки сбоя:
1. Недоступность Open-Meteo API (сетевые проблемы, maintenance)
2. Переполнение хранилища MinIO или ClickHouse
3. Некорректные данные в API ответе (неожиданная структура JSON)
4. Проблемы подключения к Docker контейнерам
5. Недостаточные права доступа к БД или хранилищу

7. Результаты работы

Пайплайн был успешно протестирован с получением реальных погодных данных для Москвы и Самары. Ниже представлены скриншоты ключевых результатов работы системы, демонстрирующие корректность всех этапов ETL процесса.

```
PS C:\Users\krech\PycharmProjects\data-engineering-2025\lab_1> python -m flows.weather_etl
04:59:07.628 | INFO    | prefect - Starting temporary server on http://127.0.0.1:8137
See https://docs.prefect.io/v3/concepts/server#how-to-guides for more information on running a dedicated Prefect server.
04:59:13.070 | INFO    | Flow run 'handsome-gazelle' - Beginning flow run 'handsome-gazelle' for flow 'Weather ETL Pipeline'
04:59:13.073 | INFO    | Flow run 'handsome-gazelle' - Запуск ETL пайплайна для погодных данных...
04:59:13.075 | INFO    | Flow run 'handsome-gazelle' -
Обработка данных для города: Москва
04:59:13.444 | INFO    | Task run 'fetch_weather_forecast_with_backoff-246' - Finished in state Completed()
04:59:13.514 | INFO    | Task run 'save_to_minio-e1e' - Finished in state Completed()
04:59:13.515 | INFO    | Flow run 'handsome-gazelle' - Сырые данные сохранены: raw/Москва_20251211_045913.json
04:59:13.524 | INFO    | Task run 'transform_to_hourly_with_logging-238' - Начало трансформации данных для города Москва
04:59:13.527 | INFO    | Task run 'transform_to_hourly_with_logging-238' - Успешно обработано 24 записей
04:59:13.530 | INFO    | Task run 'transform_to_hourly_with_logging-238' - Finished in state Completed()
04:59:13.540 | INFO    | Task run 'transform_to_daily_with_logging-a7e' - Создание дневной статистики для Москва
04:59:13.541 | INFO    | Task run 'transform_to_daily_with_logging-a7e' - Статистика: min=-4.6°C, max=3.7°C
04:59:13.544 | INFO    | Task run 'transform_to_daily_with_logging-a7e' - Finished in state Completed()
04:59:13.557 | INFO    | Task run 'validate_data_quality-c38' - Валидация данных для hourly успешно пройдена
04:59:13.560 | INFO    | Task run 'validate_data_quality-c38' - Finished in state Completed()
04:59:13.568 | INFO    | Task run 'validate_data_quality-ef9' - Валидация данных для daily успешно пройдена
04:59:13.571 | INFO    | Task run 'validate_data_quality-ef9' - Finished in state Completed()
04:59:13.802 | INFO    | Task run 'load_to_clickhouse_hourly-eb5' - Finished in state Completed()
04:59:13.803 | INFO    | Flow run 'handsome-gazelle' - Загружено 24 почасовых записей
04:59:14.055 | INFO    | Task run 'load_to_clickhouse_daily-55f' - Finished in state Completed()
04:59:14.055 | INFO    | Flow run 'handsome-gazelle' - Загружено 1 дневных записей
04:59:14.409 | INFO    | Task run 'send_telegram_notification-933' - Finished in state Completed()
04:59:14.411 | INFO    | Flow run 'handsome-gazelle' - Уведомление отправлено в Telegram
04:59:14.412 | INFO    | Flow run 'handsome-gazelle' -
Обработка данных для города: Самара
04:59:14.699 | INFO    | Task run 'fetch_weather_forecast_with_backoff-6a2' - Finished in state Completed()
04:59:14.780 | INFO    | Task run 'save_to_minio-c8f' - Finished in state Completed()
04:59:14.780 | INFO    | Flow run 'handsome-gazelle' - Сырые данные сохранены: raw/Самара_20251211_045914.json
04:59:14.790 | INFO    | Task run 'transform_to_hourly_with_logging-115' - Начало трансформации данных для города Самара
04:59:14.792 | INFO    | Task run 'transform_to_hourly_with_logging-115' - Успешно обработано 24 записей
04:59:14.796 | INFO    | Task run 'transform_to_hourly_with_logging-115' - Finished in state Completed()
04:59:14.806 | INFO    | Task run 'transform_to_daily_with_logging-6b8' - Создание дневной статистики для Самара
04:59:14.808 | INFO    | Task run 'transform_to_daily_with_logging-6b8' - Статистика: min=-4.0°C, max=-0.5°C
04:59:14.811 | INFO    | Task run 'transform_to_daily_with_logging-6b8' - Finished in state Completed()
04:59:14.825 | INFO    | Task run 'validate_data_quality-5c5' - Валидация данных для hourly успешно пройдена
04:59:14.829 | INFO    | Task run 'validate_data_quality-5c5' - Finished in state Completed()
04:59:14.838 | INFO    | Task run 'validate_data_quality-237' - Валидация данных для daily успешно пройдена
04:59:14.842 | INFO    | Task run 'validate_data_quality-237' - Finished in state Completed()
04:59:15.098 | INFO    | Task run 'load_to_clickhouse_hourly-75c' - Finished in state Completed()
04:59:15.099 | INFO    | Flow run 'handsome-gazelle' - Загружено 24 почасовых записей
04:59:15.347 | INFO    | Task run 'load_to_clickhouse_daily-ce0' - Finished in state Completed()
04:59:15.348 | INFO    | Flow run 'handsome-gazelle' - Загружено 1 дневных записей
04:59:15.807 | INFO    | Task run 'send_telegram_notification-dde' - Finished in state Completed()
04:59:15.808 | INFO    | Flow run 'handsome-gazelle' - Уведомление отправлено в Telegram
04:59:15.810 | INFO    | Flow run 'handsome-gazelle' -
ETL пайплайн успешно завершен!
04:59:15.840 | INFO    | Flow run 'handsome-gazelle' - Finished in state Completed()
04:59:15.853 | INFO    | prefect - Stopping temporary server on http://127.0.0.1:8137
PS C:\Users\krech\PycharmProjects\data-engineering-2025\lab_1>
```

Рисунок 4 - Логи Prefect

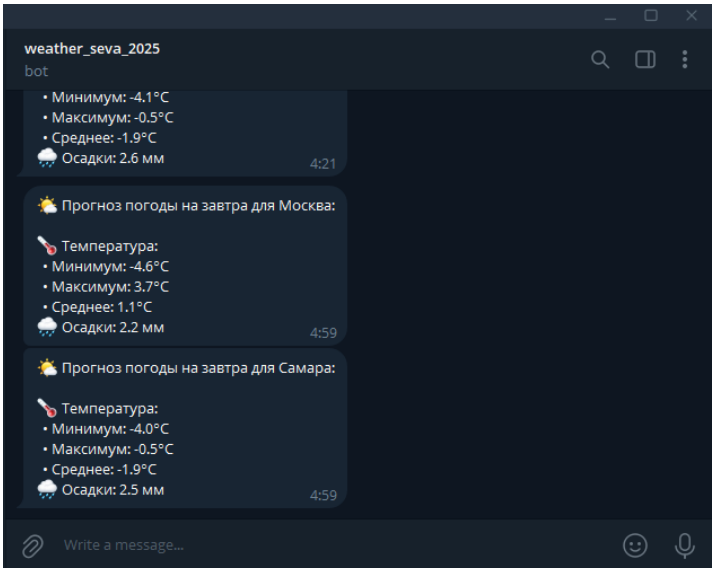

Рисунок 5 – Сообщение бота

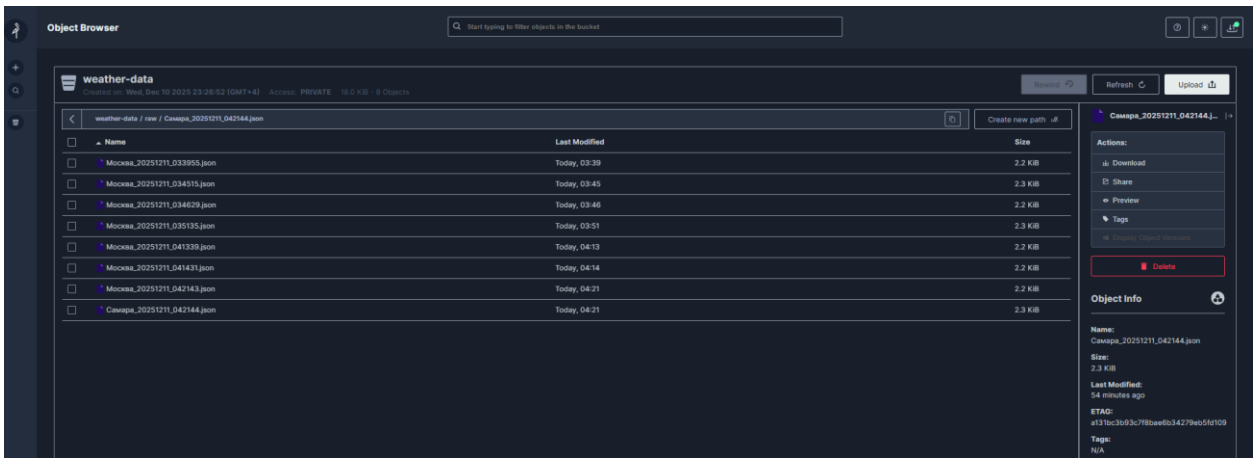

Рисунок 6 – Содержимое MinIO bucket

```
SELECT
    city,
    date,
    temp_min,
    temp_max,
    temp_avg,
    total_precipitation,
    created_at
FROM weather_daily
ORDER BY
    date DESC,
    city ASC

Query id: 272c7a56-ca17-4150-b39f-32cc72422e80

    ┌─city───┬───────date─┬─temp_min─┬─temp_max─┬──temp_avg─┬─total_precipitation─┬──────────created_at─┐
 1. │ Москва │ 2025-12-12 │     -4.6 │      3.7 │ 1.0833334 │                 2.2 │ 2025-12-11 00:59:14 │
 2. │ Москва │ 2025-12-12 │     -4.6 │      3.6 │ 1.0333333 │                 2.4 │ 2025-12-11 00:13:39 │
 3. │ Москва │ 2025-12-12 │     -4.6 │      3.6 │ 1.0333333 │                 2.4 │ 2025-12-11 00:14:31 │
 4. │ Москва │ 2025-12-12 │     -4.6 │      3.6 │ 1.0333333 │                 2.4 │ 2025-12-11 00:21:43 │
 5. │ Самара │ 2025-12-12 │       -4 │     -0.5 │   -1.8875 │                 2.5 │ 2025-12-11 00:59:15 │
 6. │ Самара │ 2025-12-12 │     -4.1 │     -0.5 │ -1.8916667 │                 2.6 │ 2025-12-11 00:21:45 │
    └────────┴────────────┴──────────┴──────────┴───────────┴─────────────────────┴─────────────────────┘

6 rows in set. Elapsed: 0.003 sec.
```

Рисунок 7 – Дневная статистика

```
SELECT
    city,
    COUNT(*) AS total_records,
    MIN(timestamp) AS first_record,
    MAX(timestamp) AS last_record
FROM weather_hourly
GROUP BY city

Query id: 94501490-11f1-4760-ba82-31d5b1af4565

    ┌─city───┬─total_records─┬────────first_record─┬─────────last_record─┐
 1. │ Самара │            48 │ 2025-12-12 00:00:00 │ 2025-12-12 23:00:00 │
 2. │ Москва │            96 │ 2025-12-12 00:00:00 │ 2025-12-12 23:00:00 │
    └────────┴───────────────┴─────────────────────┴─────────────────────┘

2 rows in set. Elapsed: 0.006 sec.
```

Рисунок 8 – Статистика по городам

8. Выводы

В ходе выполнения лабораторной работы был разработан и реализован полноценный ETL пайплайн для обработки метеорологических данных с использованием современного стека технологий.

Основные достижения:

1. Освоен Prefect 3.0 - современный инструмент оркестрации с декларативным подходом к определению задач через декораторы @task и @flow
2. Настроено Docker окружение с четырьмя сервисами, взаимодействующими через общую сеть
3. Реализован полный цикл ETL с обработкой ошибок и retry механизмами
4. Получен практический опыт работы с колоночной СУБД ClickHouse,

Возможные улучшения:

1. Добавление дашборда в Grafana для визуализации временных рядов температуры и осадков
2. Реализация инкрементальной загрузки с проверкой дубликатов по составному ключу
3. Расширение списка городов через конфигурационный файл
4. Добавление алертов при аномальных значениях (резкие изменения температуры)
5. Интеграция с Prefect Cloud для централизованного мониторинга
6. Добавление unit-тестов для функций трансформации данных
7. Реализация архивации старых данных из ClickHouse в холодное хранилище.

Лабораторная работа продемонстрировала важность правильной архитектуры пайплайна с учетом отказоустойчивости, логирования и мониторинга. Полученные навыки применимы для построения production-ready систем обработки данных.