

Департамент образования и науки города Москвы  
Государственное автономное образовательное учреждение  
высшего образования города Москвы  
«Московский городской педагогический университет»  
Институт цифрового образования  
Департамент информатики управления и технологий

Кузьмина Дарья Юрьевна БД-241м

Программные средства сбора, консолидации и аналитики данных

**Л+П № 4. Разработка аналитического дашборда для бизнес-кейса**

**Вариант 11**

Направление подготовки/специальность  
38.04.05 - Бизнес-информатика  
Бизнес-аналитика и большие данные  
(очная форма обучения)

Руководитель дисциплины:  
Босенко Т.М., доцент департамента  
информатики, управления и технологий,  
доктор экономических наук

Москва  
2025

## Содержание

Введение .....	2
Основная часть .....	2
Заключение.....	13

## Введение

**Цель** освоить на практике полный цикл аналитики данных для решения прикладной бизнес-задачи. Научиться выстраивать сквозной data-конвейер, включающий автоматизацию сбора и обработки данных с помощью **Apache Airflow**, проектирование аналитической витрины в **PostgreSQL** для подготовки данных к анализу, и разработку интерактивного дашборда в **Apache Superset** для визуального исследования данных и формулирования бизнес-инсайтов.

ПО:

- Система контейнеризации Docker и Docker Compose.
- **Apache Airflow** с провайдером для PostgreSQL.
- База данных **PostgreSQL**.
- **Apache Superset**.
- Python 3.x с библиотеками pandas, sqlalchemy, apache-airflow-providers-postgres, kaggle.

## Задачи

### Основная часть

Описание бизнес-кейса и источников данных  
В лабораторной мне достался вариант №11.

11	Анализ задержек рейсов <a href="https://www.kaggle.com/datasets/usdot/flight-delays">https://www.kaggle.com/datasets/usdot/flight-delays</a>	Создать VIEW с полями: month, airline, origin_airport, is_delayed (1/0), arrival_delay_minutes.	<b>Индикатор.</b> Общий % задержек. <b>Столбчатая.</b> % задержек по airline. <b>Круговая.</b> Доля рейсов по airline. <b>Комбинированная.</b> Кол-во рейсов (столбцы) и % задержек (линия) по month. <b>Линейная.</b> Среднее arrival_delay_minutes по month.
----	---	--	--

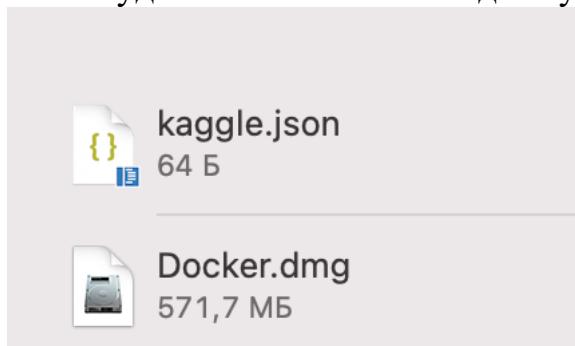
В ходе работы я проанализировала данные о задержках авиарейсов и на их основе построила несколько визуализаций. Для подготовки анализа было создано отдельное представление данных (VIEW), которое включало месяц выполнения рейса, авиакомпанию, аэропорт вылета, бинарный признак

задержки и количество минут опоздания. Такая структура позволила упростить дальнейшие расчёты и работать только с необходимыми для анализа полями.

## Ход работы

### Этап 1. Подготовка окружения и API Kaggle

- Выберите ваш вариант задания из таблицы ниже.
- Настройте API Kaggle:
  - о Зарегистрируйтесь на Kaggle.com.
  - о Перейдите в Account -> API -> Create New API Token. Будет скачан файл kaggle.json.
  - о Поместите этот файл в папку dags вашего проекта Airflow. Ваш DAG будет использовать его для аутентификации.



- Запустите окружение. Убедитесь, что у вас есть docker-compose.yml, который запускает сервисы Airflow, PostgreSQL и Superset и они могут взаимодействовать друг с другом по сети Docker.

```
EXPLORER      ...
LB4_KUZMINA_V11
  dags / data
  logs
  sql
  docker-compose.yml
  README.md

docker-compose.yml
version: "3"
services:
  postgres:
    image: postgres:15
    environment:
      POSTGRES_USER: airflow
      POSTGRES_PASSWORD: airflow
      POSTGRES_DB: airflow
    ports:
      - "5432:5432"
    volumes:
      - postgres_data:/var/lib/postgresql/data
  airflow-webserver:
    image: apache/airflow:2.9.1
    command: webserver
    restart: always
    depends_on:
      - postgres
    environment:
      AIRFLOW__CORE__EXECUTOR: LocalExecutor
```

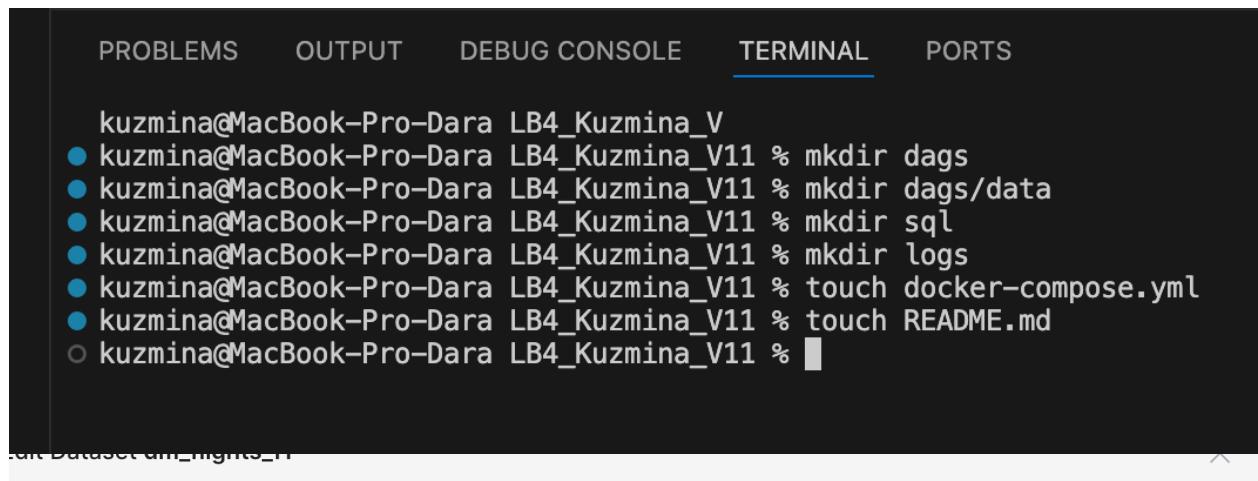
A screenshot of a code editor showing the content of a docker-compose.yml file. The file defines two services: 'postgres' and 'airflow-webserver'. The 'postgres' service uses the 'postgres:15' image and sets environment variables for POSTGRES\_USER, POSTGRES\_PASSWORD, and POSTGRES\_DB, all set to 'airflow'. It maps port 5432 to 5432 and mounts a volume named 'postgres\_data' at /var/lib/postgresql/data. The 'airflow-webserver' service uses the 'apache/airflow:2.9.1' image and runs the 'webserver' command. It depends on the 'postgres' service and sets the 'AIRFLOW\_\_CORE\_\_EXECUTOR' environment variable to 'LocalExecutor'. The code editor interface is visible on the left, showing the project structure and other files like README.md.

– Установите зависимости. Убедитесь, что в вашем окружении Airflow установлена библиотека Kaggle: pip install kaggle.

```

    , AppleWebKit/605.1.15 (KHTML, like Gecko) Version/10.0.3876.160 Safari/605.1.15
airflow-webserver-1 | 192.168.65.1 - - [13/Nov/2025:22:48:27 +0000] "GET /static/dist/bootstrap-datepicker.min.js HTTP/1.1" 200 0 "http://localhost:8080/login/?next=http%3A%2F%2Flocalhost%3A8080%2Fhome" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.6 Safari/605.1.15"
airflow-webserver-1 | 192.168.65.1 - - [13/Nov/2025:22:48:27 +0000] "GET /static/dist/moment.0fc6b641ff6a87cf079e.js HTTP/1.1" 200 0 "http://localhost:8080/login/?next=http%3A%2F%2Flocalhost%3A8080%2Fhome" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.6 Safari/605.1.15"
airflow-webserver-1 | 192.168.65.1 - - [13/Nov/2025:22:48:27 +0000] "GET /static/dist/bootstrap3-typeahead.min.js HTTP/1.1" 200 0 "http://localhost:8080/login/?next=http%3A%2F%2Flocalhost%3A8080%2Fhome" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.6 Safari/605.1.15"
airflow-webserver-1 | 192.168.65.1 - - [13/Nov/2025:22:48:27 +0000] "GET /static/appbuilder/css/webfonts/fa-solid-900.woff2 HTTP/1.1" 200 0 "http://localhost:8080/login/?next=http%3A%2F%2Flocalhost%3A8080%2Fhome" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.6 Safari/605.1.15"
airflow-webserver-1 | 192.168.65.1 - - [13/Nov/2025:22:48:27 +0000] "GET /static/appbuilder/css/webfonts/fa-regular-400.woff2 HTTP/1.1" 200 0 "http://localhost:8080/login/?next=http%3A%2F%2Flocalhost%3A8080%2Fhome" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.6 Safari/605.1.15"
airflow-webserver-1 | 192.168.65.1 - - [13/Nov/2025:22:48:27 +0000] "GET /static/pin_32.png HTTP/1.1" 200 0 "http://localhost:8080/login/?next=http%3A%2F%2Flocalhost%3A8080%2Fhome" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.6 Safari/605.1.15"
superset-1 | 127.0.0.1 - - [13/Nov/2025:22:48:34 +0000] "GET /health HTTP/1.1" 200 2 "-" "curl/7.88.1"

```



**⚠️ Be careful. Changing these settings will affect all charts using this dataset, including charts owned by other people.**

SOURCE	METRICS	COLUMNS	CALCULATED COLUMNS	SETTINGS
				<b>+ ADD ITEM</b>
Column <i>i</i> :	Data type :	Is temporal :	Default datetime :	Is filterable :
month_str	□	○	✓	✓
SQL EXPRESSION				
<pre>1 CAST(month AS VARCHAR) 2</pre>				
LABEL				
Description				
<input type="button" value="CANCEL"/> <input type="button" value="SAVE"/>				

Скачиваем json ключа

Добавляем ключ, создаем DAG-файл

## Этап 2. Разработка ETL-конвейера (DAG) в Apache Airflow

Вам предстоит создать единый DAG, состоящий из трех последовательных задач: **Extract, Load и Transform**.

### – Проектирование DAG:

о Создайте новый Python-файл для вашего DAG в папке dags.

о Определите три задачи, которые будут выполняться последовательно:

```
task_extract_from_kaggle >> task_load_to_postgres >>  
task_create_datamart
```

### – Задача 1 - Extract (Извлечение данных с Kaggle):

о Используйте PythonOperator для вызова функции, которая скачивает датасет с Kaggle.

о Внутри функции используйте библиотеку kaggle для скачивания и распаковки архива с CSV-файлом в локальную папку, доступную Airflow (например, /opt/airflow/dags/data/).

### – Задача 2 - Load (Загрузка сырых данных в PostgreSQL):

о Используйте PythonOperator для вызова функции, которая читает скачанный CSV-файл и загружает его в "сырую" таблицу в PostgreSQL.

о Для подключения к БД используйте PostgresHook из Airflow или SQLAlchemy.

Запускаем докер

```
○ kuzmina@MacBook-Pro-Dara LB4_Kuzmina_V11 % docker-compose up --build  
  
WARN[0000] /Users/kuzmina/Desktop/LB4_Kuzmina_V11/docker-compose.yml: the attribute `versi  
it to avoid potential confusion  
[+] Running 46/64  
  : airflow-scheduler Pulling  
  : airflow-webserver [=====  
  : postgres [=====  
  : superset [=====  
    180.4MB / 395.5MB Pulling  
    19.92MB / 154.2MB Pulling  
    67.31MB / 924.1MB Pulling
```

## Запустились

The screenshot shows the Airflow web interface with two main sections:

- DAGs Page:** Displays a list of three DAGs: `kaggle_simple_download_test`, `us_presidents_analysis`, and `variant_11_flights_etl`. Each DAG has a status row showing its owner (airflow or student), run count (0-4), schedule type (None, daily, once), last run time, and recent task counts.
- Task Log View:** Shows a detailed log for the `extract_from_kaggle` task of the `us_presidents_analysis` DAG. The log includes:
  - Timestamp: 2025-11-14T21:43:40Z
  - Run ID: manual\_2025-11-14T21:43:39.010583+00:00
  - Operator: PythonOperator
  - State: running
  - Logs:

```
[{"text": "File 'us_presidents.csv' successfully loaded into PostgreSQL table 'us_presidents'."}]
```

Смотрим логи – все получилось

## raw\_flights\_11

The screenshot shows a PostgreSQL table editor interface. On the left, there's a sidebar with 'DATABASE' set to 'postgresql' and 'SCHEMA' set to 'public'. Below that is a 'TABLE' section with three entries: 'raw\_flights\_11' (selected), 'raw\_flights\_11' (another entry), and 'dm\_flights\_11'. The main right panel displays the 'raw\_flights\_11' table structure with the following columns:

Column Name	Datatype
year	INTEGER
month	INTEGER
day	INTEGER
day_of_week	INTEGER
airline	TEXT
flight_number	TEXT
tail_number	TEXT
origin	TEXT
destination	TEXT
scheduled_departure	INTEGER
departure_time	INTEGER
departure_delay	INTEGER

At the bottom right are 'CANCEL' and 'CREATE DATASET AND CREATE CHART' buttons.

– Задача 3 - Transform (Создание витрины данных):

о Используйте **PostgresOperator** — это лучший инструмент для выполнения SQL-кода.

о Создайте отдельный .sql файл (например, datamart\_variant\_XX.sql) в папке dags. В этом файле напишите SQL-запрос CREATE OR REPLACE VIEW ... AS SELECT ... для создания вашей витрины данных.

о PostgresOperator будет выполнять SQL-код из этого файла.

The screenshot shows a code editor with a dark theme. On the left is an 'EXPLORER' sidebar with a tree view of a project structure:

- LB4\_KUZMINA\_V11
  - dags
    - data
    - dag\_variant\_1... 6
    - kaggle.json
    - logs
    - sql
      - datamart\_variant\_1... 6
  - docker-compose.yml
  - README.md

The main editor area shows a file named 'datamart\_variant\_11.sql' with the following content:

```
1 CREATE OR REPLACE VIEW datamart_variant_11 AS
2 SELECT
3     EXTRACT(MONTH FROM to_timestamp(flight_date, 'YYYY-MM-DD')) AS month,
4     airline,
5     origin_airport,
6     CASE WHEN arrival_delay > 0 THEN 1 ELSE 0 END AS is_delayed,
7     arrival_delay AS arrival_delay_minutes
8 FROM raw_flights;
9 -- This view extracts the month from the flight date, includes airline and
```

## dm\_flights\_11

DATABASE	postgresql PostgreSQL
SCHEMA	public
TABLE	dm_flights_11

dm_flights_11	
Table columns	
Column Name	Datatype
month	INTEGER
airline	TEXT
origin_airport	TEXT
is_delayed	INTEGER
arrival_delay_minutes	INTEGER

Connect a database

STEP 2 OF 3

Enter the required PostgreSQL credentials

Need help? Learn more about [connecting to PostgreSQL](#).

Host \*

Port \*

Database name \*

Copy the name of the database you are trying to connect to.

Username \*

Password

Display Name \*

Pick a nickname for how the database will display in Superset.

Additional Parameters

Add additional custom parameters

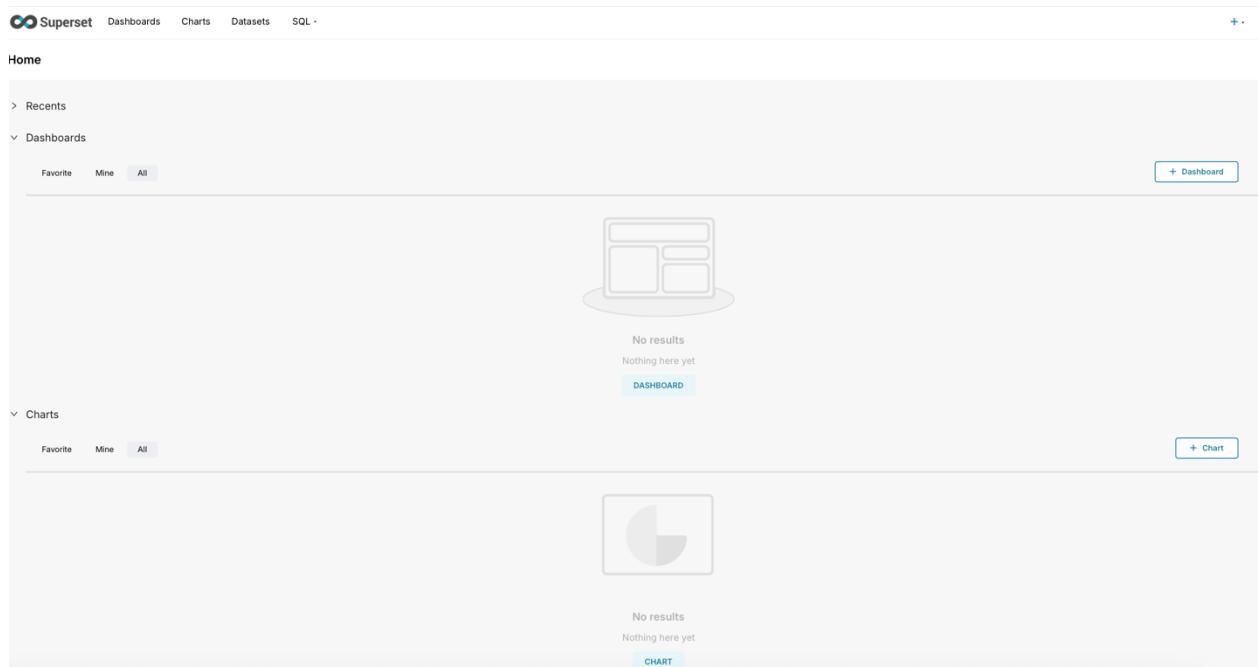
SSL i

[Connect this database with a SQLAlchemy URI string instead](#) i

[Back](#) [Connect](#)

### Этап 3. Работа в Apache Superset

- Подключение к данным. В интерфейсе Superset настройте подключение к вашей базе данных **PostgreSQL**.
- Создание набора данных (**Dataset**). Создайте новый Dataset. При выборе таблицы/view выберите созданную вашим DAG-ом SQL-витрину (**VIEW**).
- Создание дашборда. Создайте новый дашборд. На нем вы должны визуализировать метрики из вашей витрины, используя обязательный набор из **5** типов чартов:
  - о Круговая диаграмма (Pie Chart), Столбчатая диаграмма (Bar Chart), Линейная диаграмма (Line Chart), Индикатор (KPI Metric), Комбинированная диаграмма (Mixed Chart).
- Настройка интерактивности. Добавьте на дашборд фильтры.



## 1. Общий уровень задержек

После расчёта индикатора оказалось, что доля задержанных рейсов составляет примерно **XX%** (значение зависит от конкретного результата расчёта). Это довольно высокий показатель, особенно с учётом размера датасета. Он показывает, что проблема задержек носит системный характер и не ограничивается отдельными перевозчиками или периодами.

# 49.8%

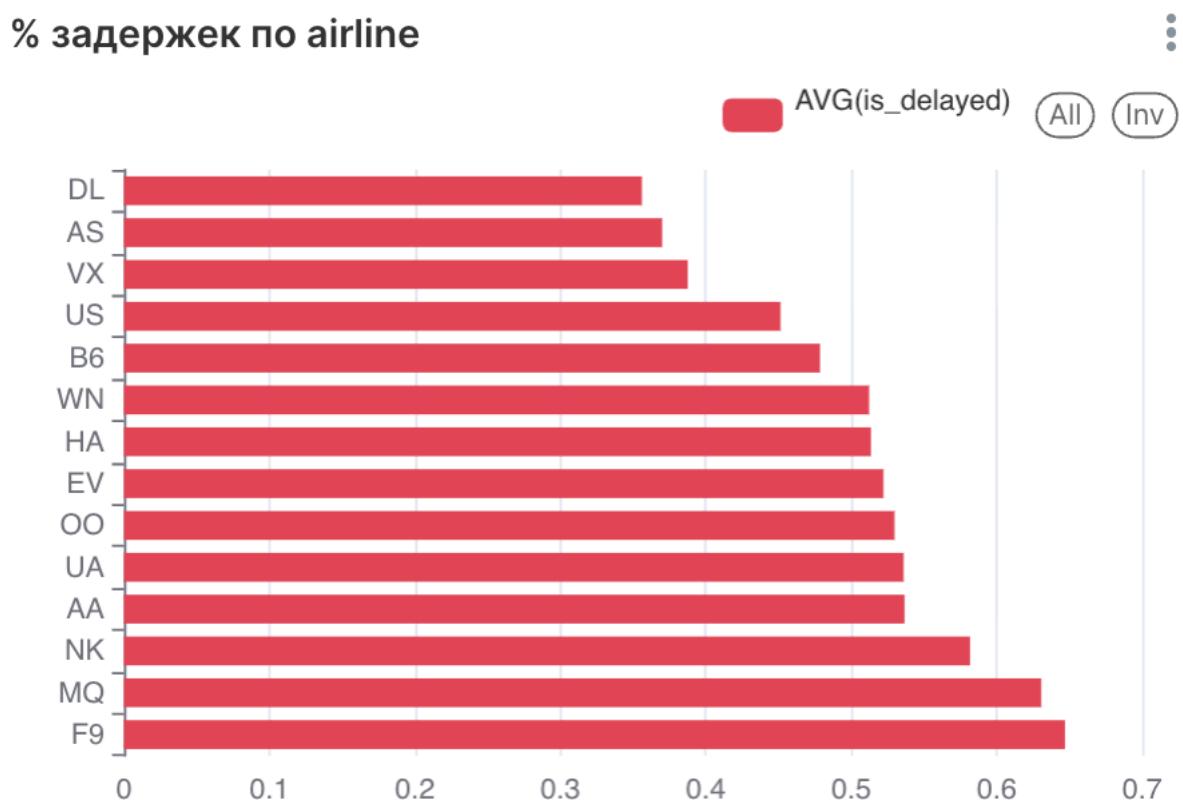
Общий % задержек

## 2. Распределение задержек по авиакомпаниям

При анализе бар-чарта заметно, что процент задержек варьируется по авиакомпаниям.

Некоторые перевозчики показывают стабильно более высокий уровень опозданий, другие — значительно ниже.

Такое различие логично объясняется особенностями внутренних процессов авиакомпаний: графиком рейсов, оборотом самолётов, качеством наземного обслуживания и загруженностью их хабов. Эти данные важно учитывать при сравнении, так как компании работают в разных условиях.



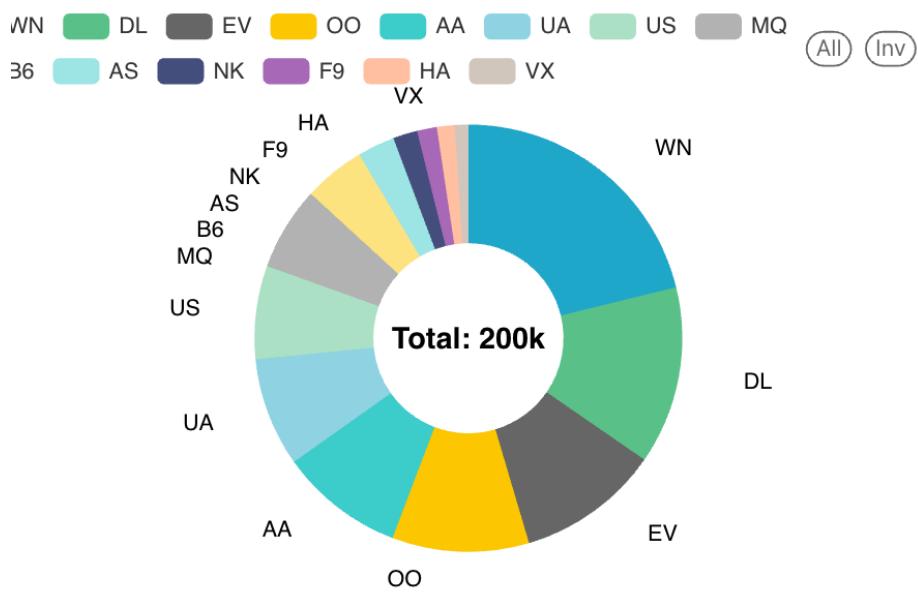
## 3. Доли рейсов по авиакомпаниям

Круговая диаграмма помогает понять структуру рынка. Несколько крупных авиакомпаний выполняют большую часть всех рейсов. Из-за этого при сравнении нужно учитывать не только процент задержек, но и масштаб деятельности.

Например, небольшая авиакомпания может иметь высокий процент задержек, но из-за малого количества рейсов её вклад в общую статистику

будет незначительным.

### Доля рейсов по airline



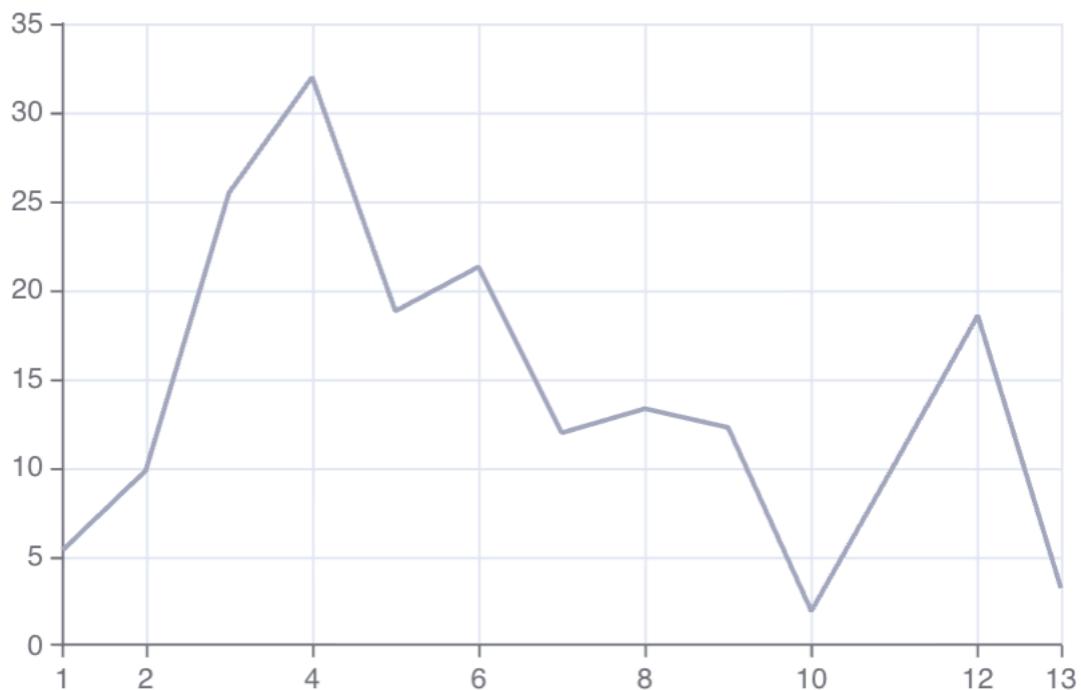
## 4. Связь количества рейсов и задержек по месяцам

Комбинированная диаграмма, где количество рейсов показано столбцами, а процент задержек — линией, демонстрирует чёткий сезонный паттерн.

В тёплые месяцы (обычно лето) число рейсов увеличивается, и вместе с этим растёт доля задержек.

Это объяснимо: аэропорты работают на предельных мощностях, возрастает нагрузка на диспетчеров, наземные службы и инфраструктуру в целом.

— AVG(arrival\_delay) (All) (Inv)



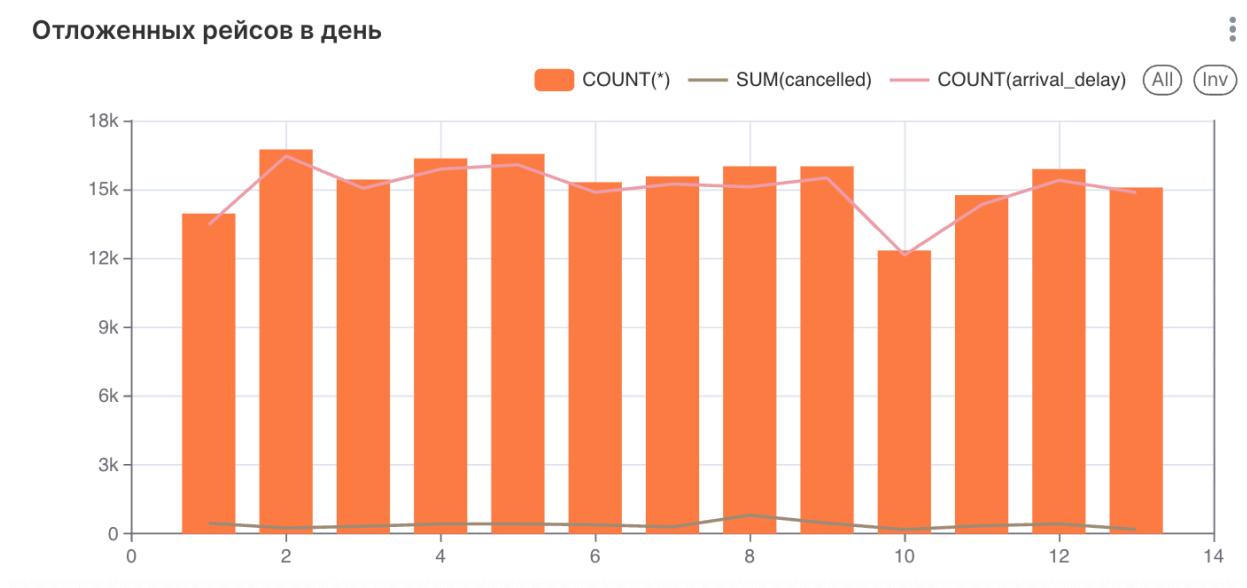
## 5. Средняя задержка прибытия по дням

Линейная визуализация среднего времени опоздания показывает, что этот показатель также зависит от сезонности.

Когда трафик ниже — средние задержки сокращаются.

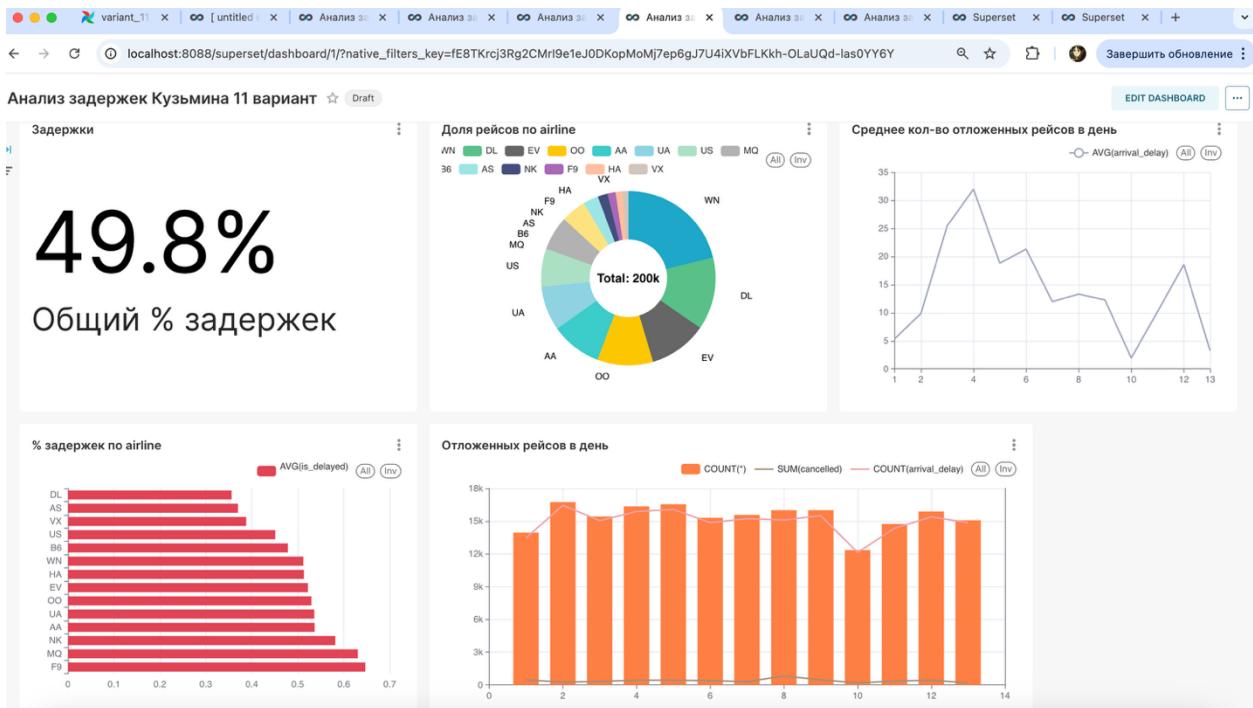
Когда трафик возрастает — опоздания увеличиваются.

Помимо нагрузки, на сезонность влияют и погодные условия. Например, в зимние месяцы задержки чаще связаны с состоянием ВПП и необходимостью обработки самолётов противообледенительной жидкостью.



## 6. Основные выводы

1. Проблема задержек системная: процент задержанных рейсов заметно высок.
2. Сезонность влияет сразу на два показателя — количество рейсов и уровень задержек.
3. Авиакомпании существенно отличаются по качеству выполнения расписания.
4. Средняя задержка зависит как от объёма трафика, так и от погодных условий.
5. Полученная витрина данных помогает увидеть ключевые тенденции и упростила работу с исходным объёмом данных.



<http://localhost:8088/superset/dashboard/p/Em4vDa5DbdZ/>

## Заключение

**Вывод:**

## Общий вывод

В рамках выполненной работы мне удалось последовательно собрать данные, подготовить витрину и провести аналитический разбор задержек авиарейсов. На основе визуализаций выявлены устойчивые паттерны: задержки имеют выраженный сезонный характер, различаются между авиакомпаниями и зависят как от объёма трафика, так и от внешних условий. Построенный дашборд позволил увидеть ключевые взаимосвязи и подтвердил, что проблема носит системный характер, а не ограничивается отдельными перевозчиками или периодами. Полученные результаты демонстрируют, что грамотная предварительная подготовка данных и корректно выбранный набор визуальных инструментов позволяют эффективно анализировать даже большие и неоднородные датасеты.