Airflow: Data Pipelines Tool

20192208 김형훈 20201349 이선표 20210365 이예찬

Table of Contents

Section 1

Airflow 등장 배경 및 개념 설명

20210365 이예찬

Background: Airflow 등장 배경

Cron 작업과 스크립트 의존

워크플로우 관리를 위해 Cron 작업이나 파이썬 스크립트를 사용했지만, 복잡한 의존성 처리가 어려웠음

데이터 규모 증가에 따른 확장성 부족

데이터 규모가 커지면서 기존 도구로는 관리하기 어려워졌음

작업 실패 시 모니터링 어려움

작업 실패 원인 파악과 디버깅이 쉽지 않았음

팀 간 비효율적 협업

데이터 엔지니어, 분석가, 개발자가 각자 다른 도구를 사용하면서 팀 간 단절이 발생

이러한 문제를 해결하기 위해 Airflow가 등장했고, DAG 기반 워크플로우, 자동화된 스케줄링, 모니터링 대시보드 등의 혁신적 기능을 제공했다.

Background: Airflow History

Airflow 특징

- **DAG 기반 Workflow Automation and Orchestration**
  DAG(Directed Acyclic Graph)를 통해 작업 간 의존성을 직관적으로 관리 가능
- **Scalibility**
  작업량이 증가하더라도 시스템이 안정적으로 수평 확장 가능
- **Monitoring and Dashboard**
  Web UI를 통해 작업들의 현재 진행 상태를 모니터링 할 수 있음
- **다양한 Community Plugin 제공**
  원하는 기능이 있는 사용자 정의 plugin들을 자유롭게 사용 가능

Airflow Concepts: 구성 요소

Data Engineer는 Workflow와 Task 간 의존성을 정의하는 설정 파일인 **DAG 파일을 생성합니다.**

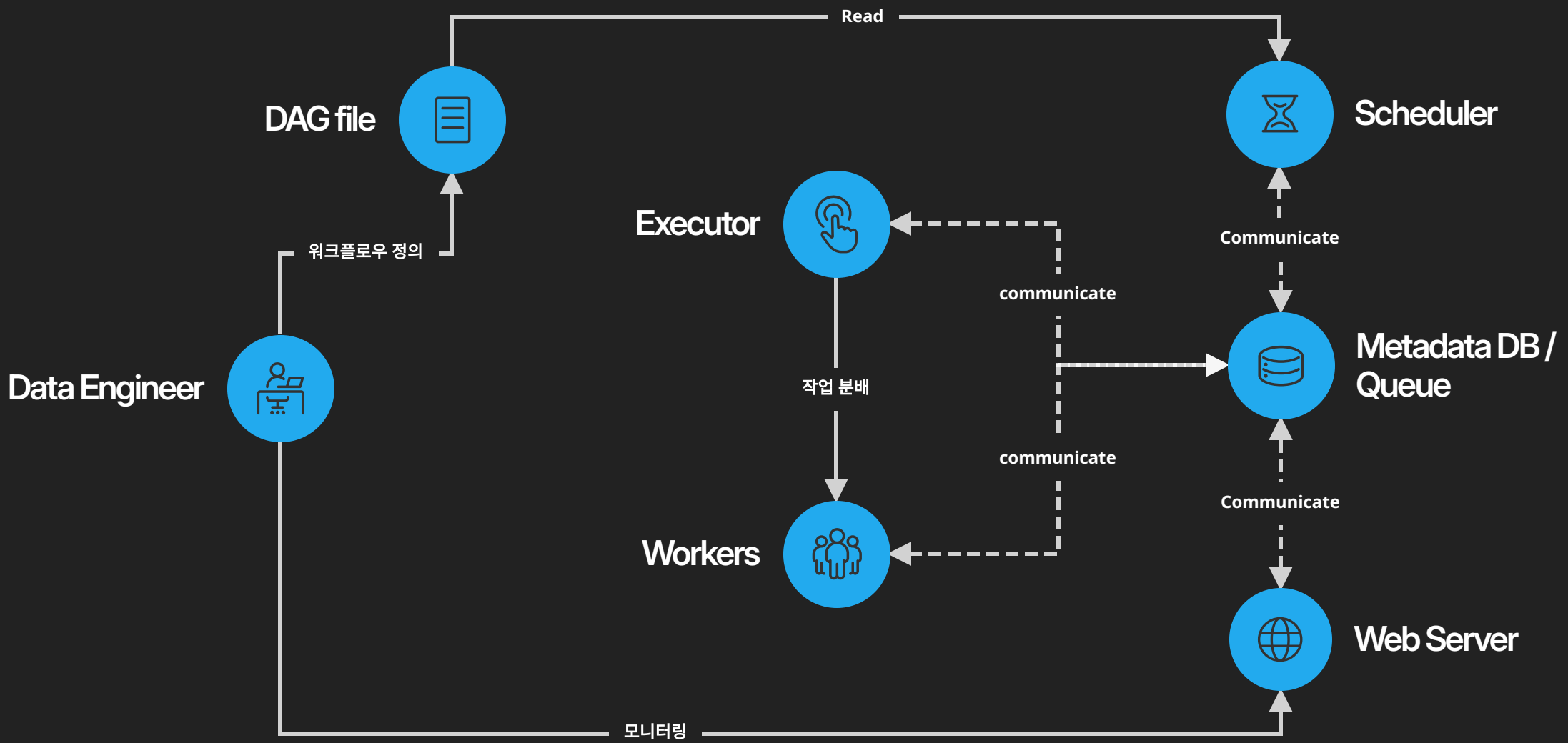
Scheduler는 **DAG 파일을 지속적으로 모니터링 하고, 파일에** 정의된 의존성을 바탕으로 Task 실행 시점과 순서를 결정합니다.

실행할 준비가 된 Task들을 Executor가 적절하게 Worker들에게 분배합니다. 대규모 환경에서도 안정적으로 작업을 수행할 수 있도록 분배가 이루어집니다.

Worker는 할당받은 작업을 처리합니다.

Data Engineer는 Airflow의 Web Server를 사용해 워크플로우 진행 상황을 모니터링합니다

Airflow Concepts: 구성 요소

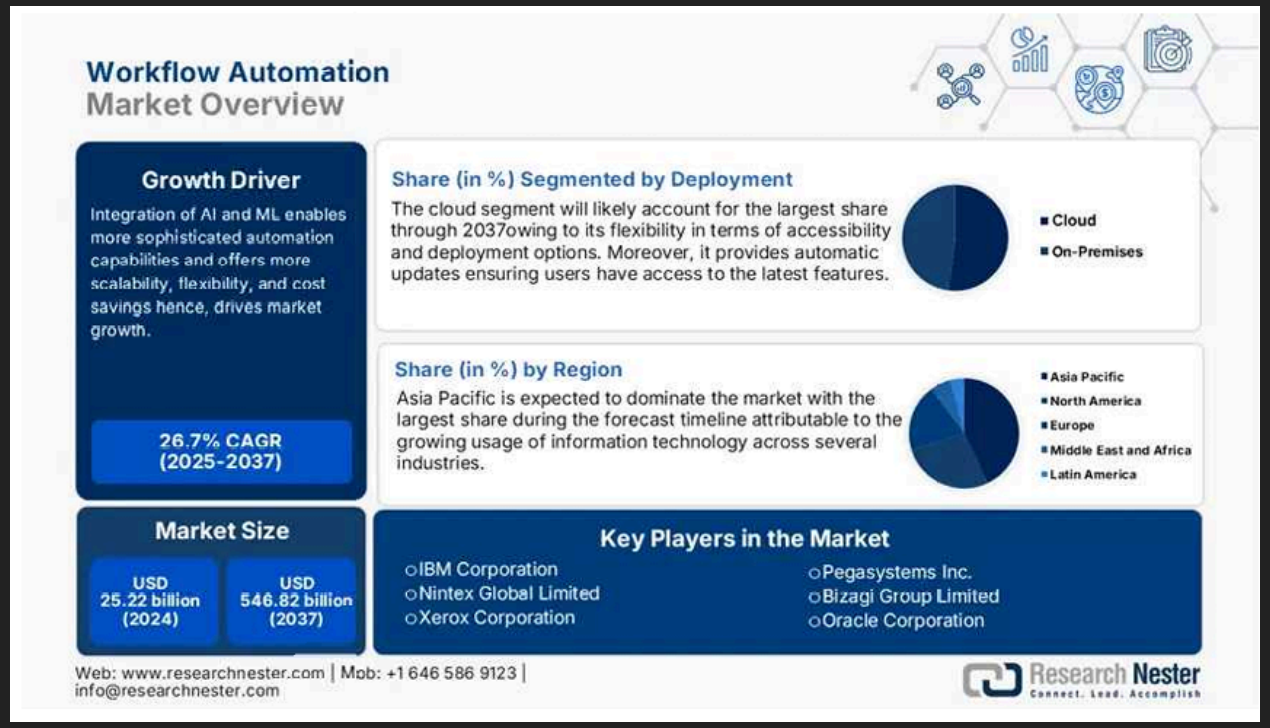


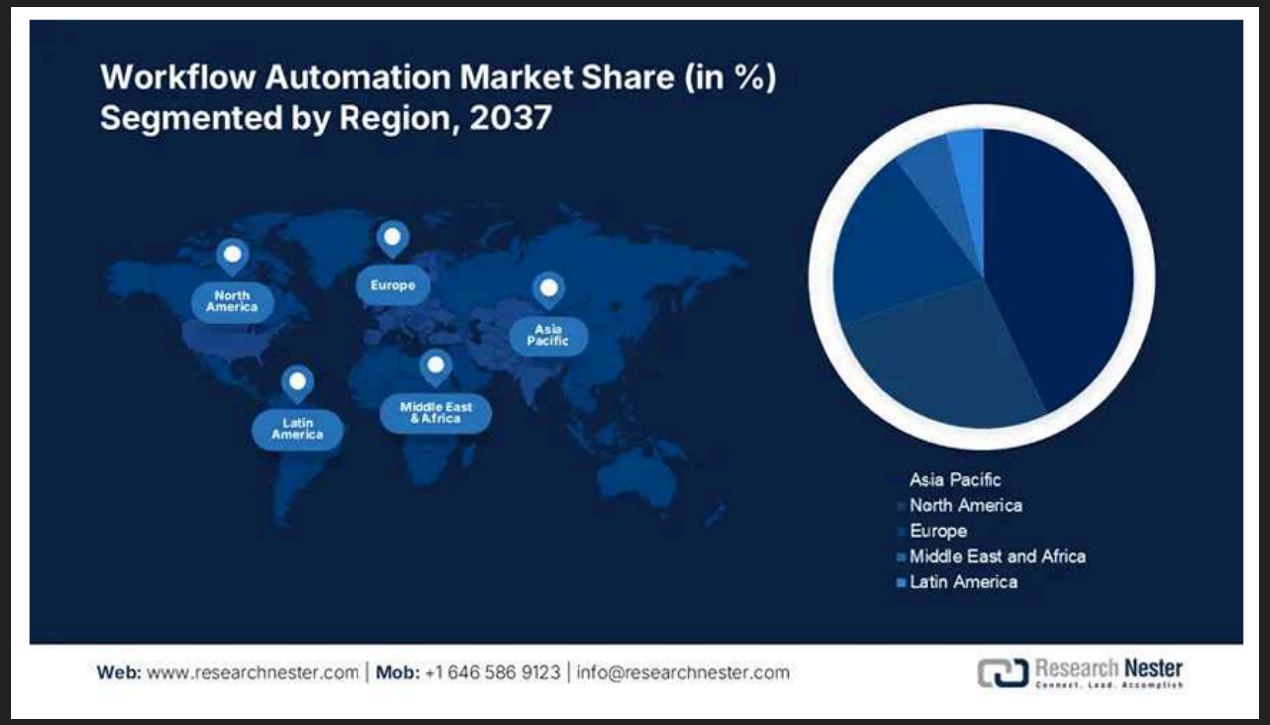
Section 2

Airflow 사용 현황

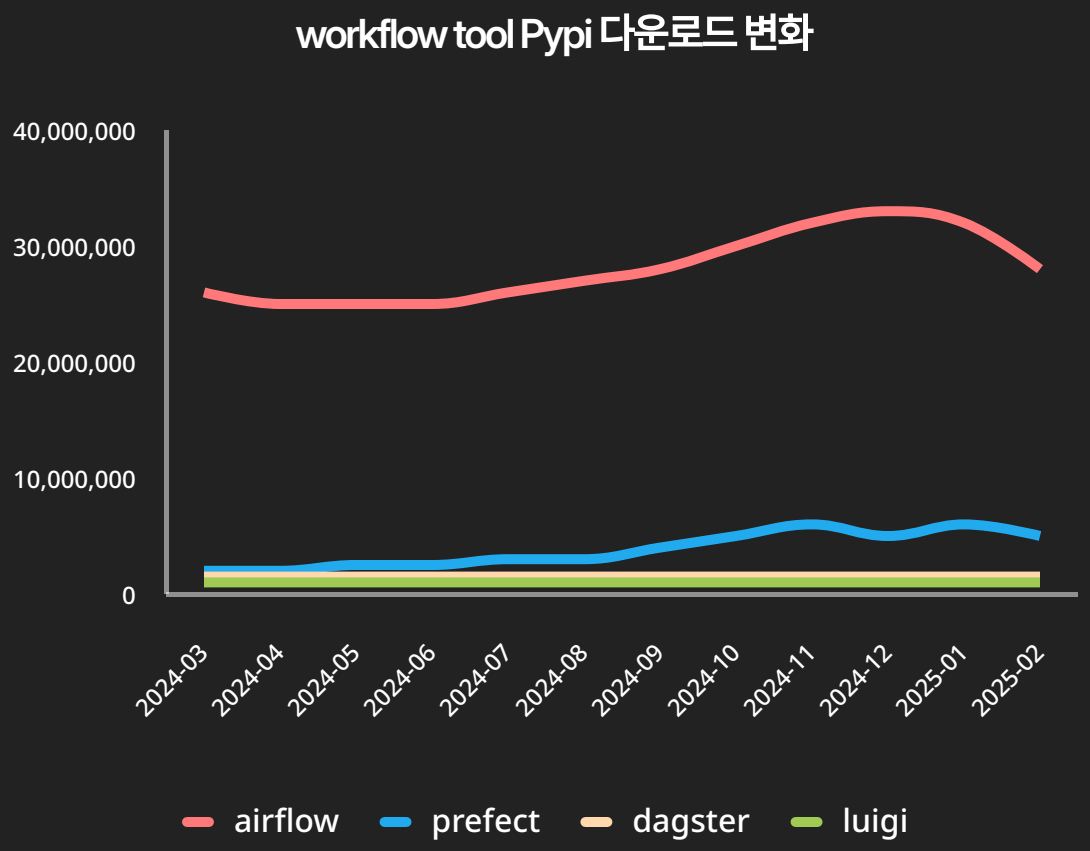
20201349 이선표

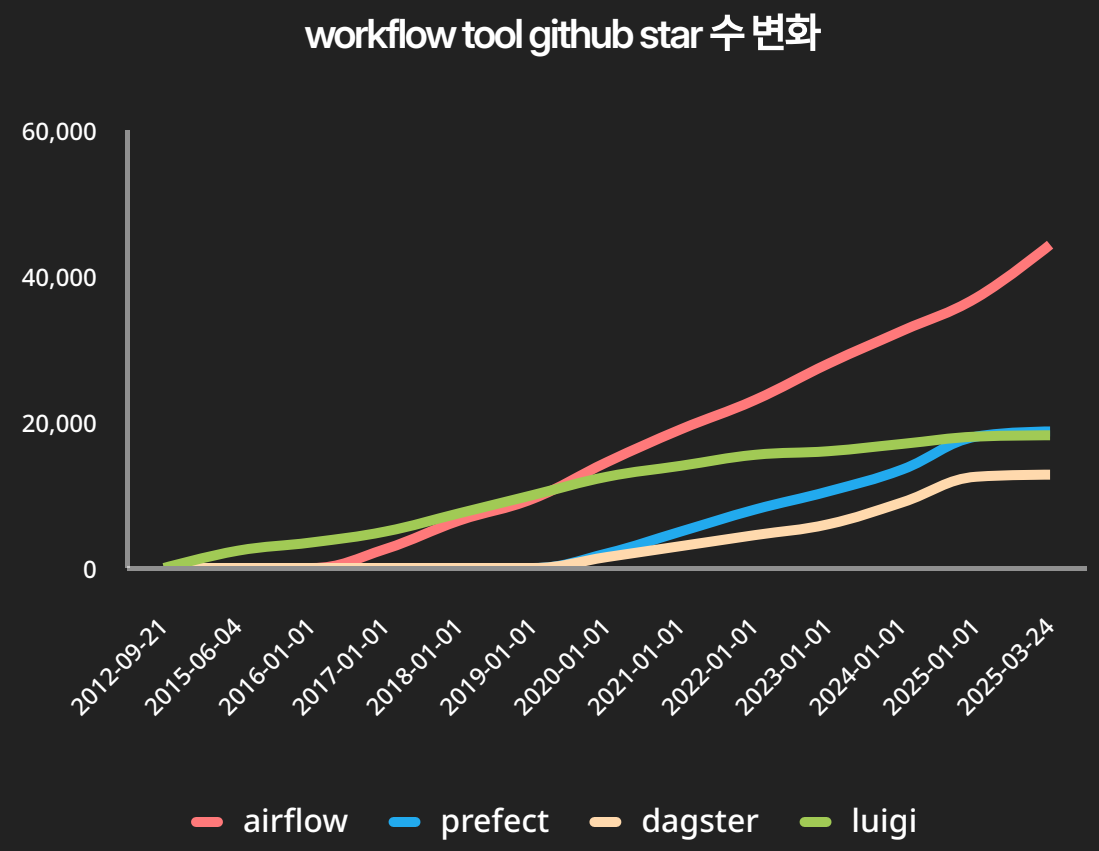
Airflow Trend: 데이터 분석 분야에서의 Workflow Tool



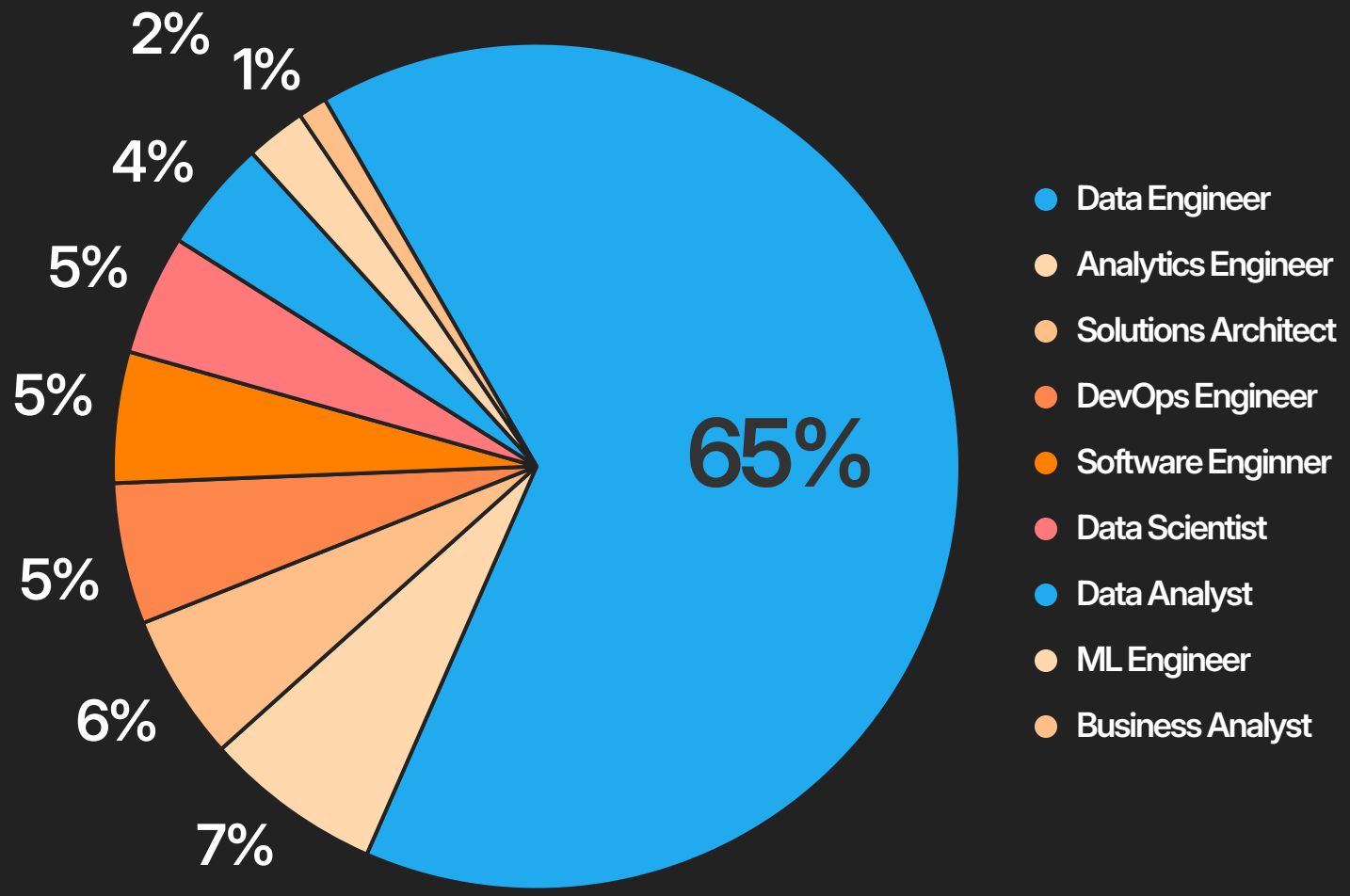


Airflow Trend: 데이터 분석 분야에서의 Workflow Tool





Airflow Trend: 직종 비율



Airflow Trend: 기업 사용 사례

Airbnb

Spotify

Mayo Clinic

General Electric

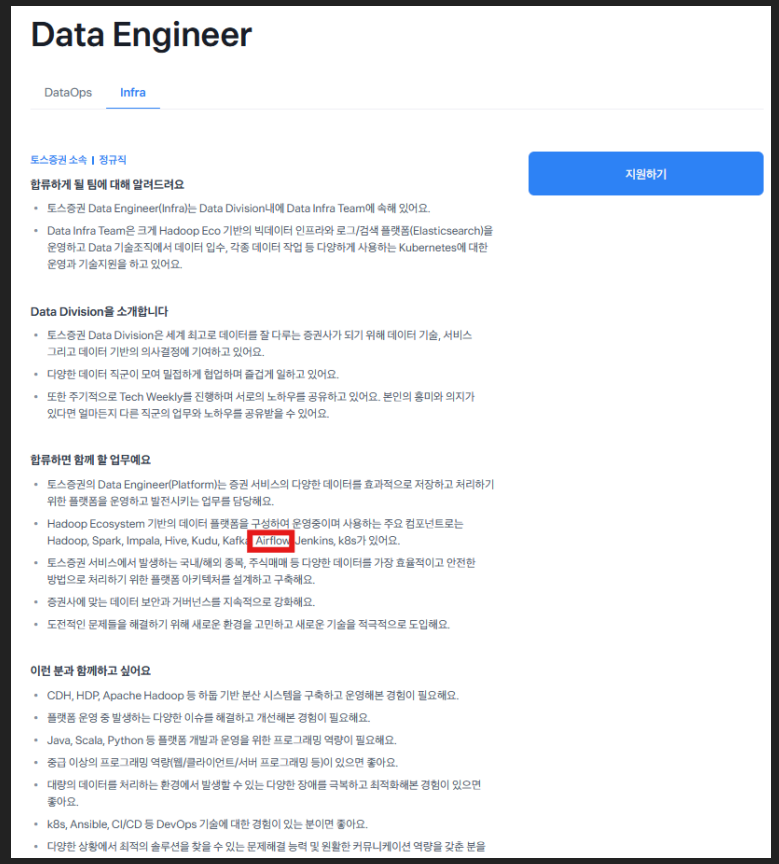
오늘의 집

쏘카

우아한 형제들

무신사

Airflow Trend: 채용 시장에서의 수요



Data Engineer

DataOps | Infra

토스증권 소속 | 정규직

지원하기

함류하게 될 팀에 대해 알려드려요

- 토스증권 Data Engineer(Infra)는 Data Division내에 Data Infra Team에 속해 있어요.
- Data Infra Team은 크게 Hadoop Eco 기반의 빅데이터 인프라와 로그/검색 플랫폼(Elasticsearch)을 운영하고 Data 기술조직에서 데이터 입수, 각종 데이터 작업 등 다양하게 사용하는 Kubernetes에 대한 운영과 기술지원을 하고 있어요.

Data Division을 소개합니다

- 토스증권 Data Division은 세계 최고로 데이터를 잘 다루는 증권사가 되기 위해 데이터 기술, 서비스 그리고 데이터 기반의 의사결정에 기여하고 있어요.
- 다양한 데이터 직군이 모여 밀접하게 협업하며 즐겁게 일하고 있어요.
- 또한 주기적으로 Tech Weekly를 진행하며 서로의 노하우를 공유하고 있어요. 본인의 흥미와 의지가 있다면 얼마든지 다른 직군의 업무와 노하우를 공유받을 수 있어요.

함류하면 함께 할 업무에요

- 토스증권의 Data Engineer(Platform)는 증권 서비스의 다양한 데이터를 효과적으로 저장하고 처리하기 위한 플랫폼을 운영하고 발전시키는 업무를 담당해요.
- Hadoop Ecosystem 기반의 데이터 플랫폼을 구성하여 운영중이며 사용하는 주요 컴포넌트로는 Hadoop, Spark, Impala, Hive, Kudu, Kafka, Airflow, Jenkins, k8s가 있어요.
- 토스증권 서비스에서 발생하는 국내/해외 종목, 주식매매 등 다양한 데이터를 가장 효율적이고 안전한 방법으로 처리하기 위한 플랫폼 아키텍처를 설계하고 구축해요.
- 증권사에 맞는 데이터 보안과 거버넌스를 지속적으로 강화해요.
- 도전적인 문제들을 해결하기 위해 새로운 환경을 고민하고 새로운 기술을 적극적으로 도입해요.

이런 분과 함께하고 싶어요

- CDH, HDP, Apache Hadoop 등 하둡 기반 분산 시스템을 구축하고 운영해본 경험이 필요해요.
- 플랫폼 운영 중 발생하는 다양한 이슈를 해결하고 개선해본 경험이 필요해요.
- Java, Scala, Python 등 플랫폼 개발과 운영을 위한 프로그래밍 역량이 필요해요.
- 중급 이상의 프로그래밍 역량(웹/클라이언트/서버 프로그래밍 등)이 있으면 좋아요.
- 대량의 데이터를 처리하는 환경에서 발생할 수 있는 다양한 장애를 극복하고 최적화해본 경험이 있으면 좋아요.
- k8s, Ansible, CI/CD 등 DevOps 기술에 대한 경험이 있는 분이면 좋아요.
- 다양한 상황에서 최적의 솔루션을 찾을 수 있는 문제해결 능력 및 원활한 커뮤니케이션 역량을 갖춘 분을

토스 증권 Data Engineer

MLOps Engineer Internship

Platform & Infra Magok, Seoul

팀 소개

Platform&Infra팀은 AI 모델의 개발부터 서비스 운영을 위한 배포에 이르기까지 AI 모델의 수명 주기를
인을 구축합니다. 또한 AI 서비스의 안정적인 운영 지원을 위한 보안성 강화, 인프라 관리 및 자원 최적화

수행 업무

- AI 모델의 학습/추론 플랫폼을 설계하고 구축하며 운영합니다.
- AI/ML 플랫폼 운영 및 생산성 향상을 위한 다양한 서비스와 도구를 개발합니다.

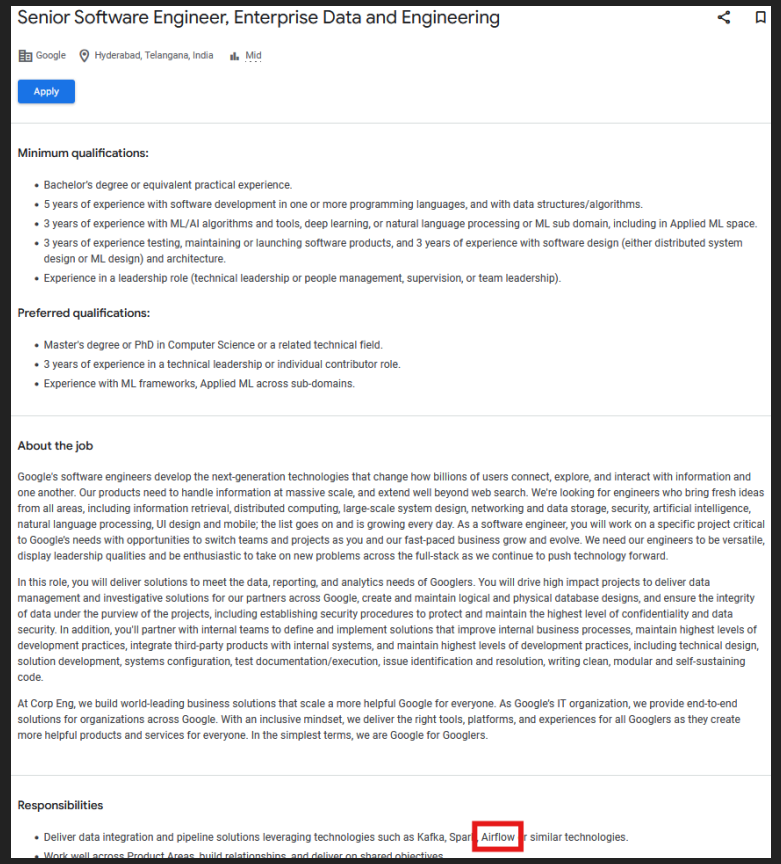
지원자격

- Linux 및 CLI 환경을 다뤄본 경험이 있으신 분
- Python, Go 등 프로그래밍 언어를 활용한 웹 애플리케이션 개발을 해봤으면 좋아요.
- Docker 및 Kubernetes 같은 컨테이너 기술의 기본 개념을 이해하고 있으면 좋아요.

우대사항

- GCP, AWS, Azure 같은 Public Cloud 환경에서 개발을 해봤으면 좋아요.
- CI/CD 도구(Helm, Kustomize, ArgoCD 등)를 활용한 개발 및 운영을 해봤으면 좋아요.
- Triton, TensorRT 같은 AI 서빙 프레임워크를 사용해봤으면 좋아요.
- Airflow, Kubeflow 같은 Workflow 툴을 사용해봤으면 좋아요.
- 기술적인 내용을 문서화하고 팀원들과 공유해봤으면 좋아요.

LG MLOps Engineer Internship



Senior Software Engineer, Enterprise Data and Engineering

Google Hyderabad, Telangana, India Mid

Apply

Minimum qualifications:

- Bachelor's degree or equivalent practical experience.
- 5 years of experience with software development in one or more programming languages, and with data structures/algorithms.
- 3 years of experience with ML/AI algorithms and tools, deep learning, or natural language processing or ML sub domain, including in Applied ML space.
- 3 years of experience testing, maintaining or launching software products, and 3 years of experience with software design (either distributed system design or ML design) and architecture.
- Experience in a leadership role (technical leadership or people management, supervision, or team leadership).

Preferred qualifications:

- Master's degree or PhD in Computer Science or a related technical field.
- 3 years of experience in a technical leadership or individual contributor role.
- Experience with ML frameworks, Applied ML across sub-domains.

About the job

Google's software engineers develop the next-generation technologies that change how billions of users connect, explore, and interact with information and one another. Our products need to handle information at massive scale, and extend well beyond web search. We're looking for engineers who bring fresh ideas from all areas, including information retrieval, distributed computing, large-scale system design, networking and data storage, security, artificial intelligence, natural language processing, UI design and mobile; the list goes on and is growing every day. As a software engineer, you will work on a specific project critical to Google's needs with opportunities to switch teams and projects as you and our fast-paced business grow and evolve. We need our engineers to be versatile, display leadership qualities and be enthusiastic to take on new problems across the full-stack as we continue to push technology forward.

In this role, you will deliver solutions to meet the data, reporting, and analytics needs of Googlers. You will drive high impact projects to deliver data management and investigative solutions for our partners across Google, create and maintain logical and physical database designs, and ensure the integrity of data under the purview of the projects, including establishing security procedures to protect and maintain the highest level of confidentiality and data security. In addition, you'll partner with internal teams to define and implement solutions that improve internal business processes, maintain highest levels of development practices, integrate third-party products with internal systems, and maintain highest levels of development practices, including technical design, solution development, systems configuration, test documentation/execution, issue identification and resolution, writing clean, modular and self-sustaining code.

At Corp Eng, we build world-leading business solutions that scale a more helpful Google for everyone. As Google's IT organization, we provide end-to-end solutions for organizations across Google. With an inclusive mindset, we deliver the right tools, platforms, and experiences for all Googlers as they create more helpful products and services for everyone. In the simplest terms, we are Google for Googlers.

Responsibilities

- Deliver data integration and pipeline solutions leveraging technologies such as Kafka, Spark, Airflow, or similar technologies.
- Work well across Product Areas, build relationships, and deliver on shared objectives.

Google Senior Software Engineer

Section 3

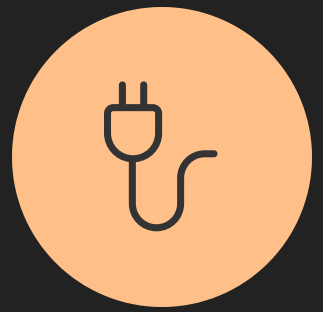
Airflow Demo 실습

20192208 김형훈

Airflow Concepts: Key Object in DAG

Operators

DAG 파일에서 사용 가능한 모듈화된 독립적 Task. 기본 내장 Operator 뿐만 아니라 다양한 커뮤니티 정의 Operator도 사용 가능



Connections

mysql이나 cloud service와 같은 외부 시스템과 소통하기 위한 credentials과 configuration이 정의된 객체

Hooks

Connection에 연결해 다양한 함수를 사용할 수 있는 기능을 제공하는 객체

Airflow Demo: Automating Data Pipelines

서로 다른 리소스에서 데이터를 추출

추출된 데이터 전처리 및 통합

통합된 데이터를 사용하여 통계적 분석을 진행하고, 보고서 작성

작성된 보고서를 이메일로 전송

Airflow Demo: Dummy Data

```sql
CREATE TABLE shopping_data (
    amount FLOAT,
    date DATE
);

INSERT INTO shopping_data (amount, date) VALUES
    (300.00, '2025-04-01'),
    (280.00, '2025-04-02'),
    (290.00, '2025-04-03'),
    (200.00, '2025-04-04'),
    (null, '2025-04-05'),
    (130.00, '2025-04-06'),
    (140.00, '2025-04-07'),
    (160.00, '2025-04-08'),
    (270.00, '2025-04-09'),
    (180.00, '2025-04-10'),
    (120.00, '2025-04-11'),
    (70.00,  '2025-04-12'),
    (60.00,  '2025-04-13'),
    (65.00,  '2025-04-14'),
    (null, '2025-04-15'),
    (110.00, '2025-04-16'),
    (250.00, '2025-04-17'),
    (260.00, '2025-04-18'),
    (255.00, '2025-04-19'),
```

날짜 별 판매량 데이터

```sql
CREATE TABLE factor_data (
    date DATE,
    staff_count INT,
    operating_hours FLOAT,
    product_variety INT,
    event_frequency INT,
    store_cleanliness INT,
    training_hours FLOAT
);

INSERT INTO factor_data (date, staff_count, operating_hours, \
    product_variety, event_frequency, store_cleanliness, training_hours) VALUES
    ('2025-04-01', 10, 12.0, 50, 2, 8, 5.0),
    ('2025-04-02', 9,  11.5, 45, 1, 7, 4.5),
    ('2025-04-03', 10, 12.0, 48, 2, 8, 5.0),
    ('2025-04-04', 7,  10.0, 40, 1, 6, 3.0),
    ('2025-04-05', 5,  9.0,  35, 0, 5, 2.0),
    ('2025-04-06', 4,  8.5,  30, 0, 4, 1.5),
    ('2025-04-07', 5,  9.0,  35, 1, 5, 2.0),
    ('2025-04-08', 6,  10.0, 38, 1, 6, 3.0),
    ('2025-04-09', 9,  11.5, 45, 2, 7, 4.0),
    ('2025-04-10', 7,  10.5, 40, 1, 6, 3.5),
    ('2025-04-11', 4,  8.0,  30, 0, 4, 1.0),
    ('2025-04-12', 3,  7.5,  25, 0, 3, 0.5),
    ('2025-04-13', 3,  7.0,  20, 0, 3, 0.5),
    ('2025-04-14', 4,  8.0,  28, 0, 4, 1.0),
```

날짜 별 기타 요소 데이터

Airflow Demo: Download and Setup

- **Download Airflow configuration file**

  curl -LfO 'https://airflow.apache.org/docs/apache-airflow/2.5.1/docker-compose.yaml'

- **Add Email Settings**

  다운받은 config 파일에 메일 전송 과정을 위해 필요한 환경 설정을 추가해줍니다. 메일 전송은 Gmail SMTP server를 이용해서 구현할 예정입니다.

- **Set up MySQL Databases**

  다운받은 config 파일에 앞에서 제시한 데이터베이스를 위한 환경 설정을 추가해줍니다.

- **Execute Airflow**

  설정이 완료되면 'docker compose up -d' 명령어를 입력하여 airflow를 실행합니다.

Airflow Demo: Setup

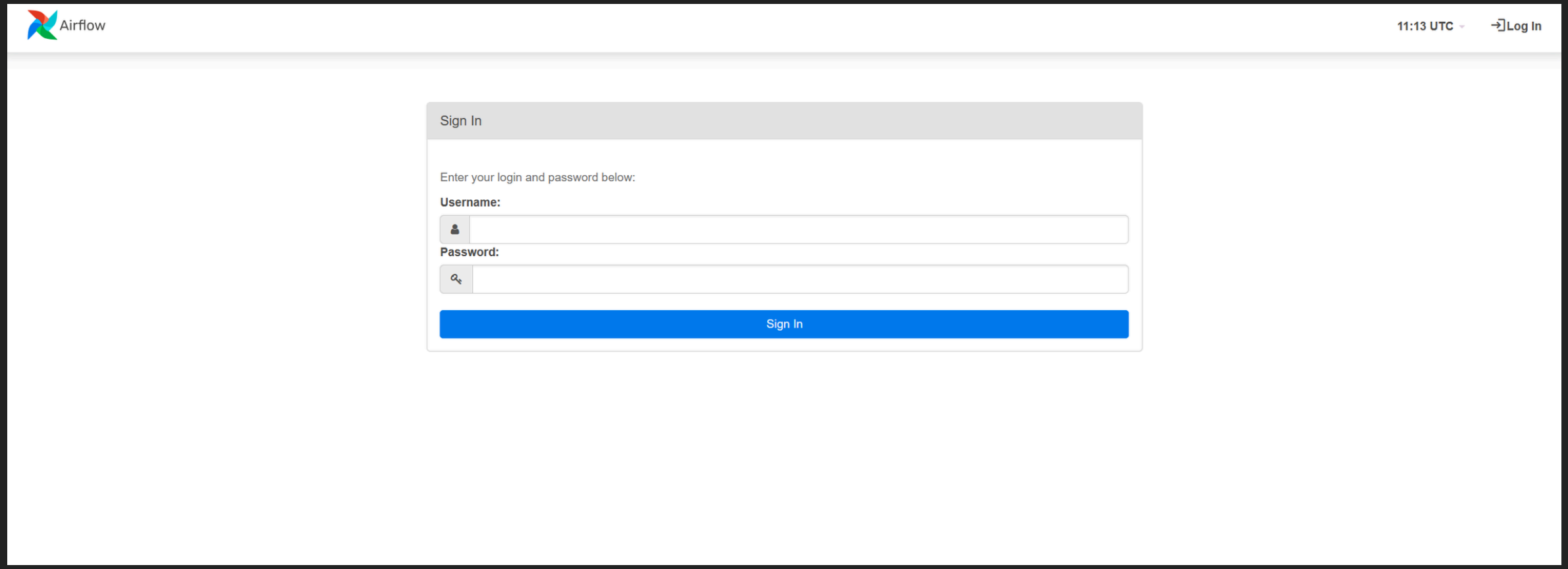
```
AIRFLOW__EMAIL__EMAIL_BACKEND: airflow.utils.email.send_email_smtp
AIRFLOW__SMTP__SMTP_HOST: smtp.gmail.com
AIRFLOW__SMTP__SMTP_PORT: ${SMTP_PORT:-587}
AIRFLOW__SMTP__SMTP_USER: ${SMTP_USER}
AIRFLOW__SMTP__SMTP_PASSWORD: ${SMTP_PASSWORD}
AIRFLOW__SMTP__SMTP_MAIL_FROM: ${SMTP_FROM_MAIL}
```

메일 전송 환경 변수

```
services:
  mysql1:
    image: mysql:8.0
    environment:
      MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD:-root}
      MYSQL_DATABASE: ${SHOPPING_DB_NAME:-shopping_db}
      MYSQL_USER: ${MYSQL_USER:-user}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD:-secret}
    volumes:
      - ./mysql1-data:/var/lib/mysql
      - ./init-mysql1.sql:/docker-entrypoint-initdb.d/init.sql
    healthcheck:
      test: ["CMD", "mysqladmin", "ping", "-h", "localhost", "-u", "root", \
             "-prootpassword"]
      interval: 10s
      timeout: 5s
      retries: 5
      start_period: 10s

  mysql2:
    image: mysql:8.0
    environment:
      MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD:-root}
      MYSQL_DATABASE: ${WEATHER_DB_NAME:-weather}
      MYSQL_USER: ${MYSQL_USER:-user}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD:-secret}
```

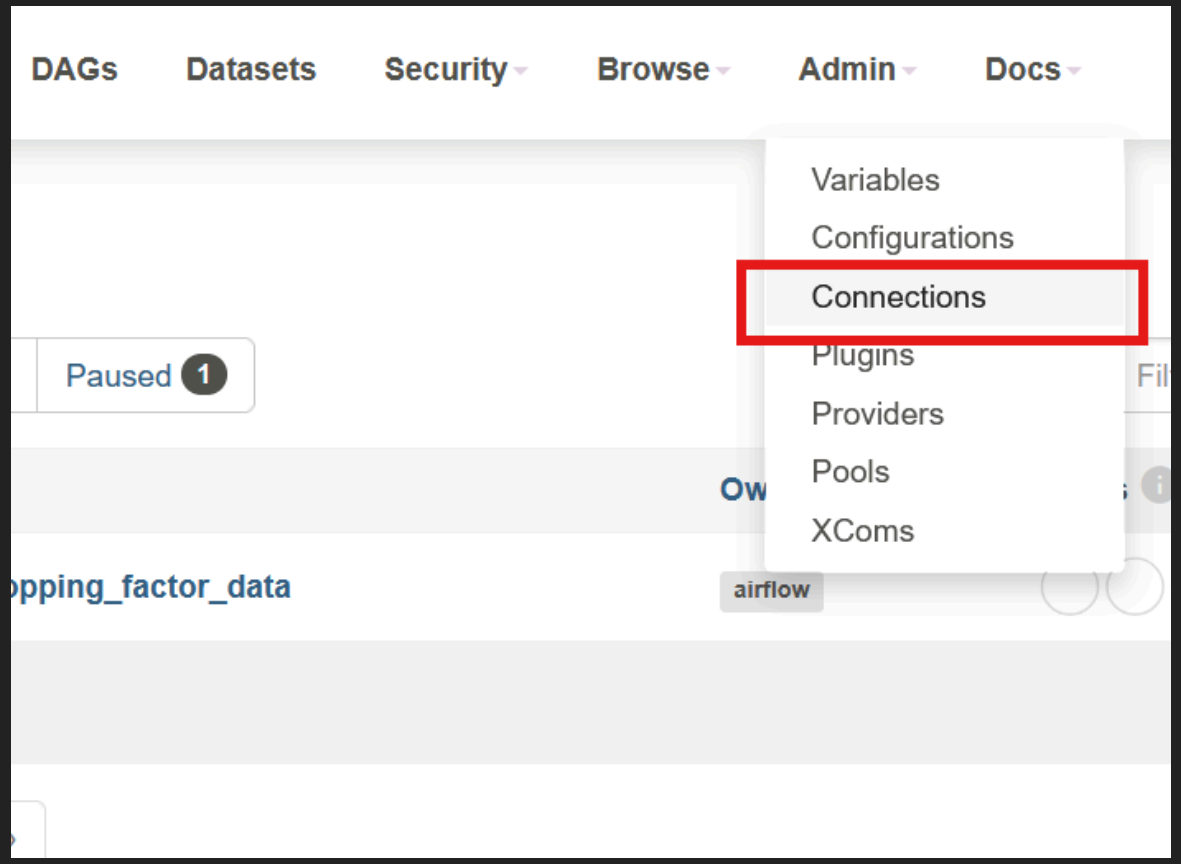
데이터베이스 설정

Airflow Demo: Setup

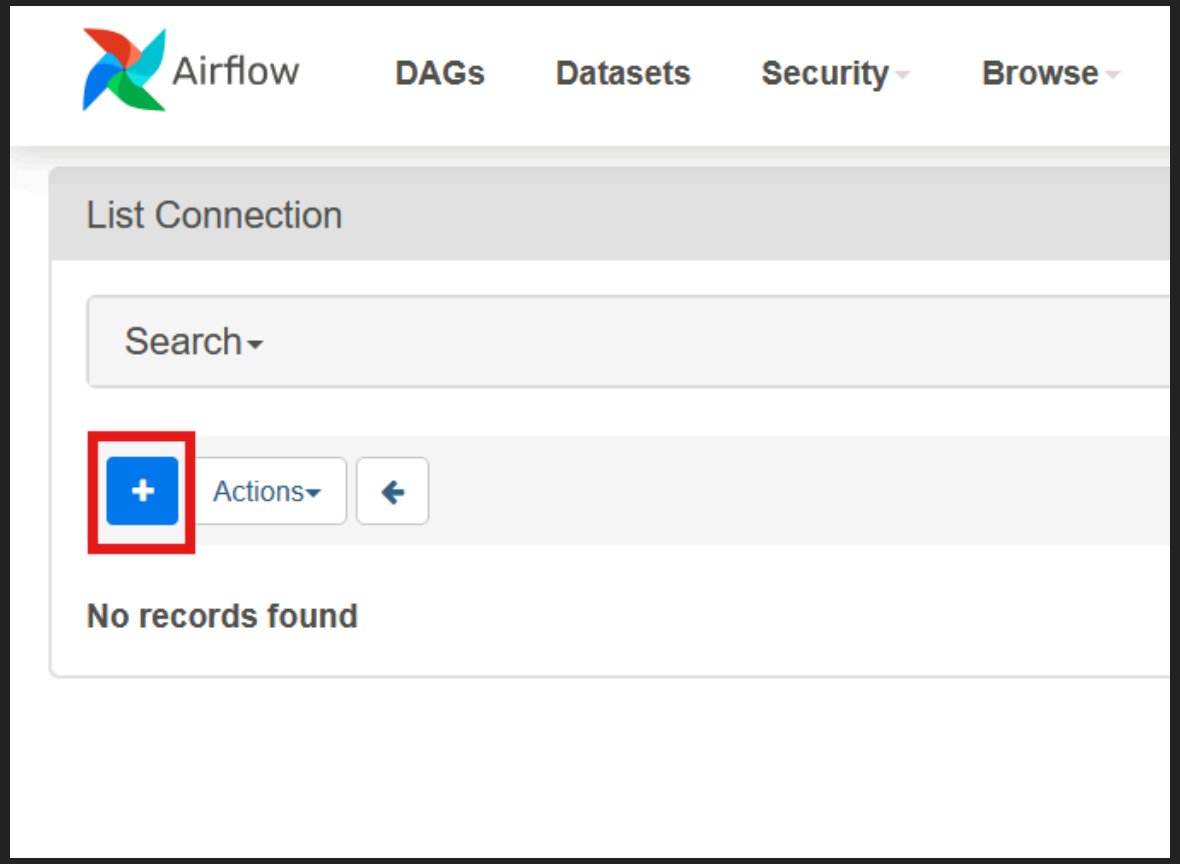


Airflow 실행 후 'localhost:8080' 접속 화면

Airflow Demo: Connection Setup

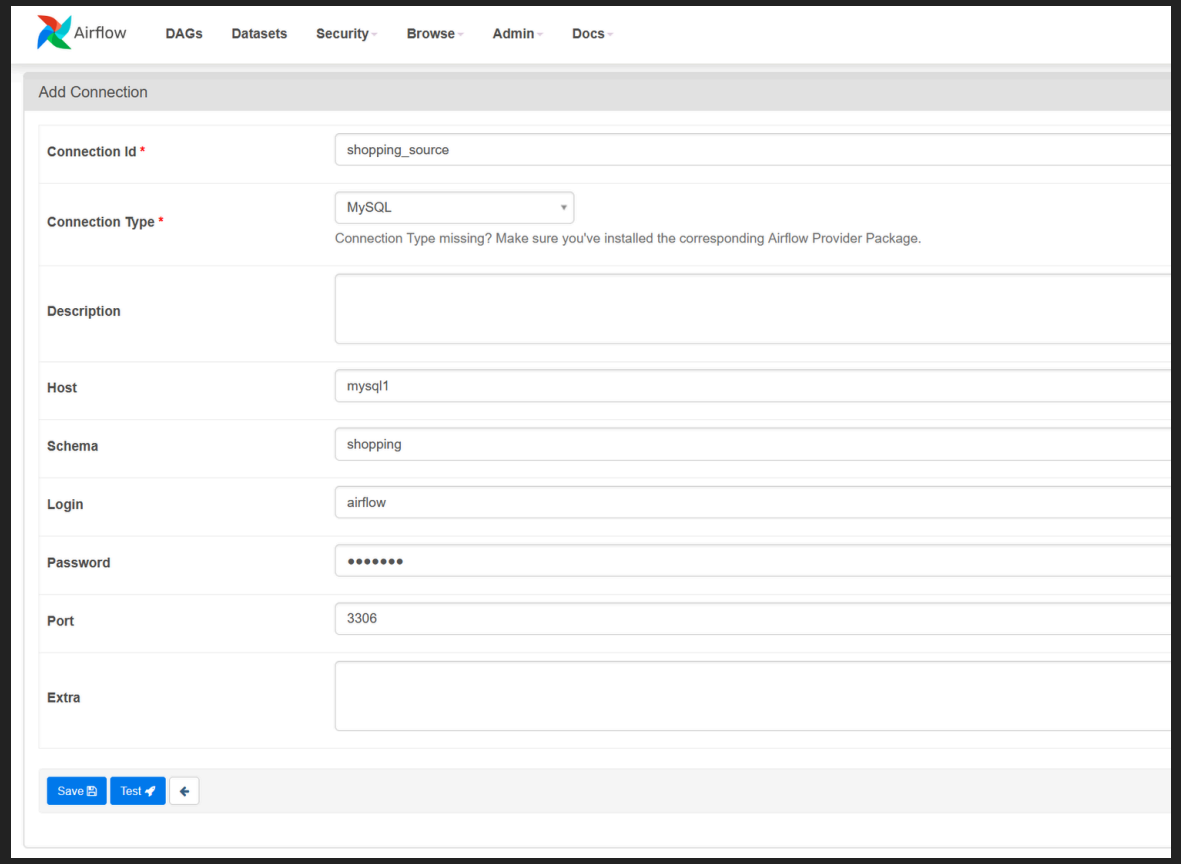


상단 메뉴에서 Connections 클릭

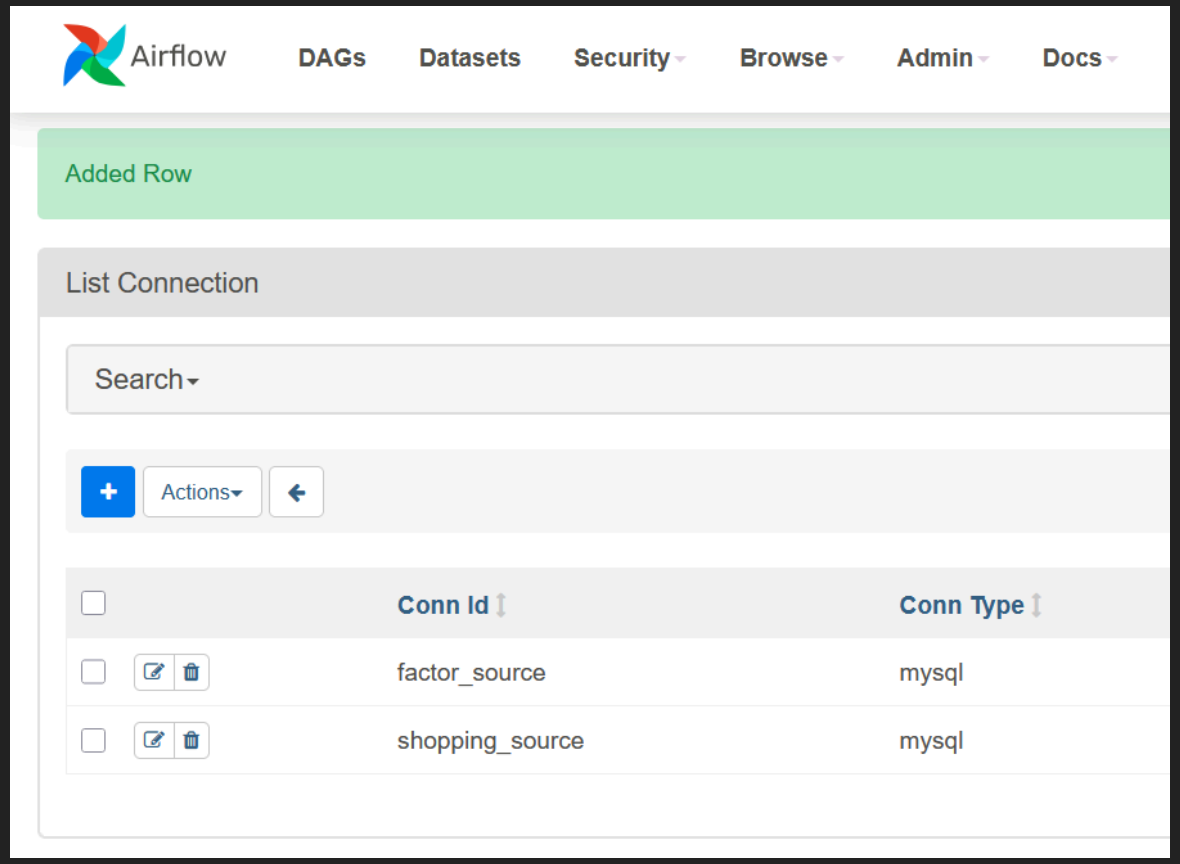


Connections 화면에서 + 클릭

Airflow Demo: Connection Setup



설정에 맞게 Form 작성 후 Save



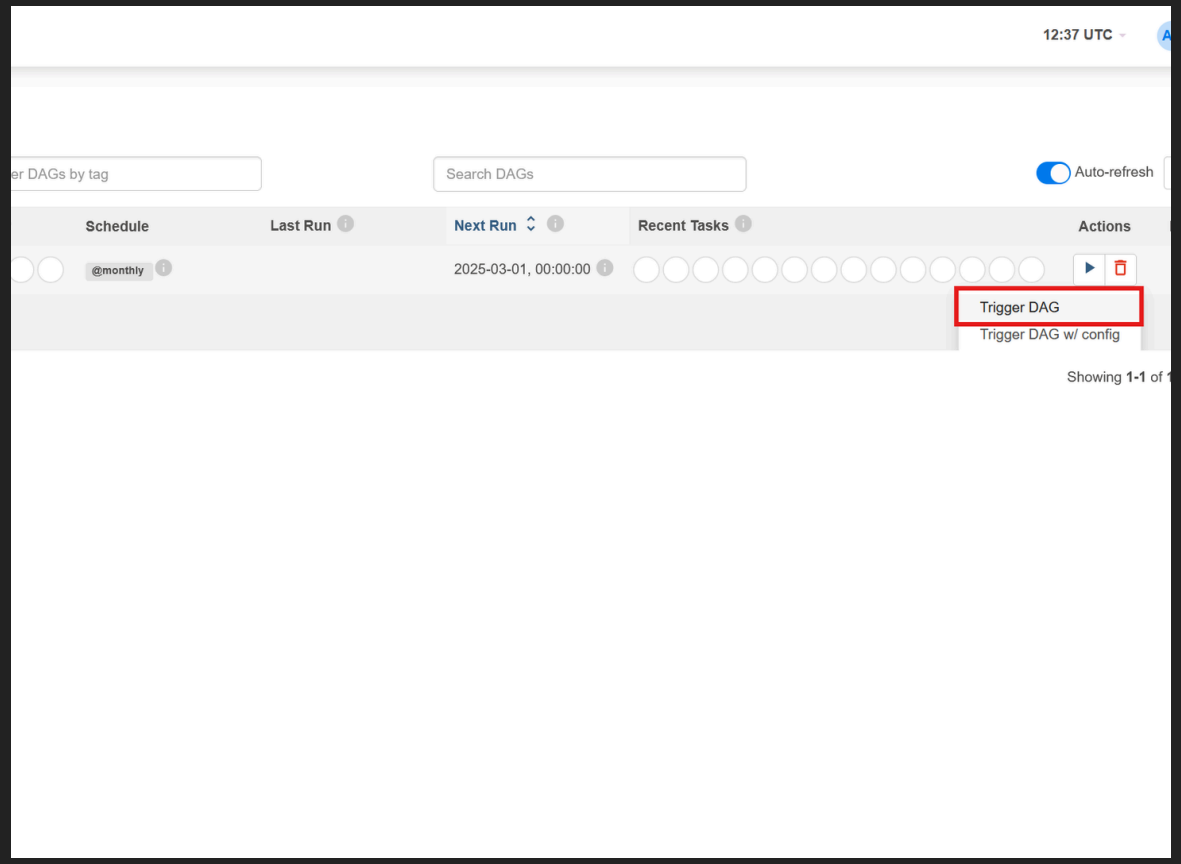
나머지 Database도 알맞게 설정

Airflow Demo: DAG 코드

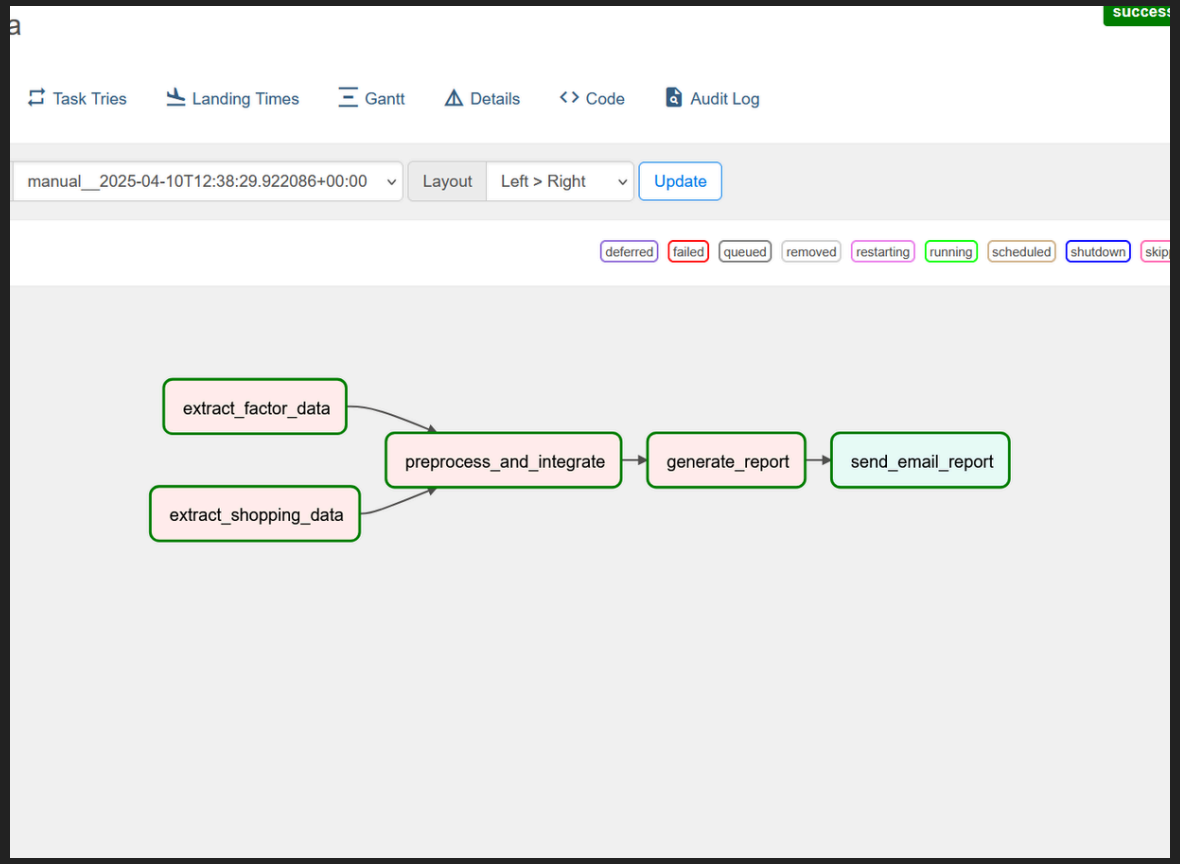
```python
from airflow.operators.email import EmailOperator

with DAG(
    dag_id='integrate_shopping_factor_data',
    schedule='@monthly',
    start_date=datetime(2024, 4, 4),
    catchup=False,
) as dag:

    def extract_shopping_data():
        hook = MySqlHook(mysql_conn_id='shopping_source')
        conn = hook.get_sqlalchemy_engine().connect()
        df = pd.read_sql("SELECT * FROM shopping_data", conn)
        df.to_csv('/tmp/shopping_data.csv', index=False)

    def extract_factor_data():
        hook = MySqlHook(mysql_conn_id='factor_source')
        conn = hook.get_sqlalchemy_engine().connect()
        df = pd.read_sql("SELECT * FROM factor_data", conn)
        df.to_csv('/tmp/factor_data.csv', index=False)

    def preprocess_and_integrate():
        shopping_df = pd.read_csv('/tmp/shopping_data.csv')
        factor_df = pd.read_csv('/tmp/factor_data.csv')
        shopping_df['amount'] = shopping_df['amount'].fillna( \
                                    shopping_df['amount'].mean())
        integrated_df = pd.merge(shopping_df, factor_df, on='date', how='inner')
        integrated_df.to_csv('/tmp/integrated_data.csv', index=False)
        return "Data preprocessed and integrated"

    def generate_report(**kwargs):
        df = pd.read_csv('/tmp/integrated_data.csv')
```

```python
extract_shopping_data_task = PythonOperator(
    task_id='extract_shopping_data',
    python_callable=extract_shopping_data,
)

extract_factor_data_task = PythonOperator(
    task_id='extract_factor_data',
    python_callable=extract_factor_data,
)

process_task = PythonOperator(
    task_id='preprocess_and_integrate',
    python_callable=preprocess_and_integrate,
)

report_task = PythonOperator(
    task_id='generate_report',
    python_callable=generate_report,
    provide_context=True,
)

email_task = EmailOperator(
    task_id='send_email_report',
    to='cryscham123@naver.com',
    subject='Monthly Sales Insights Report',
    html_content="{{ ti.xcom_pull(task_ids='generate_report', \
                                  key='html_report') }}",
)

[extract_shopping_data_task, extract_factor_data_task] >> process_task \
  >> report_task >> email_task
```

Airflow Demo: DAG 실행

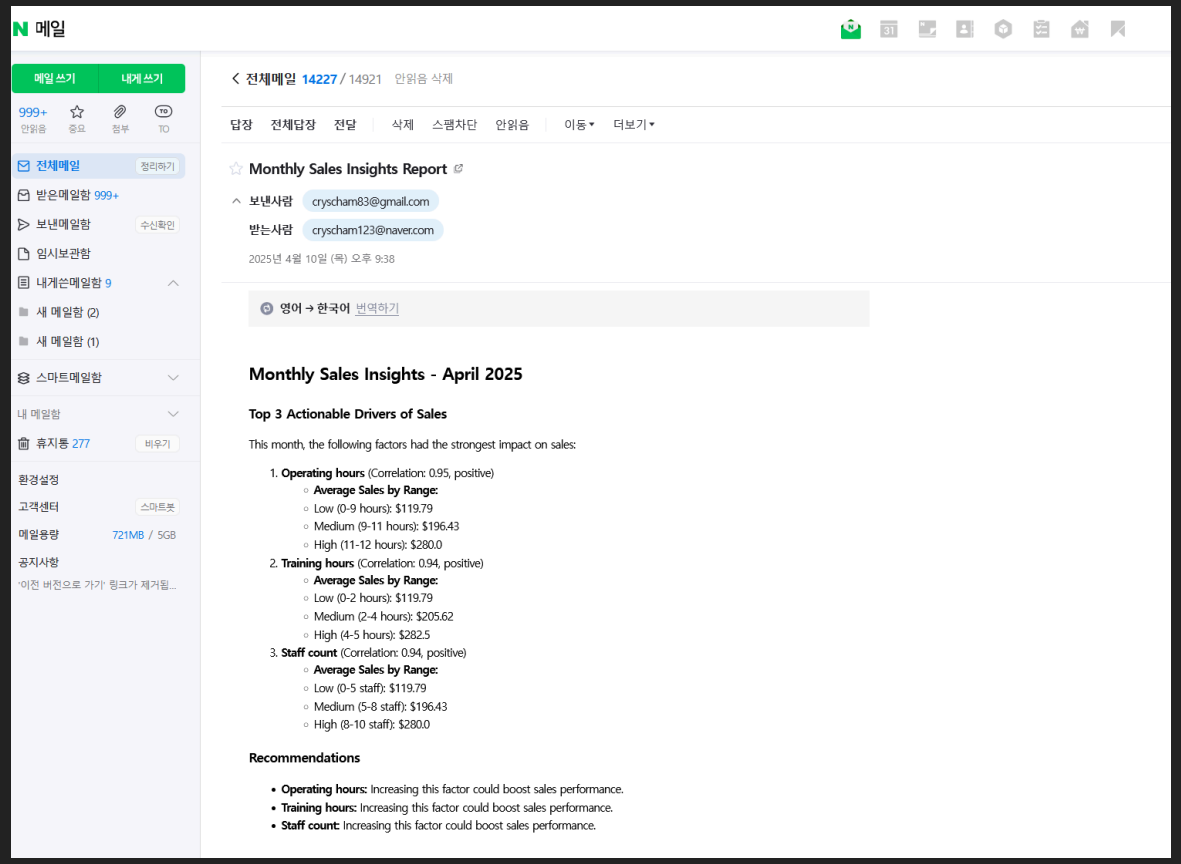


다시 UI로 돌아와 수동으로 Trigger 해줌

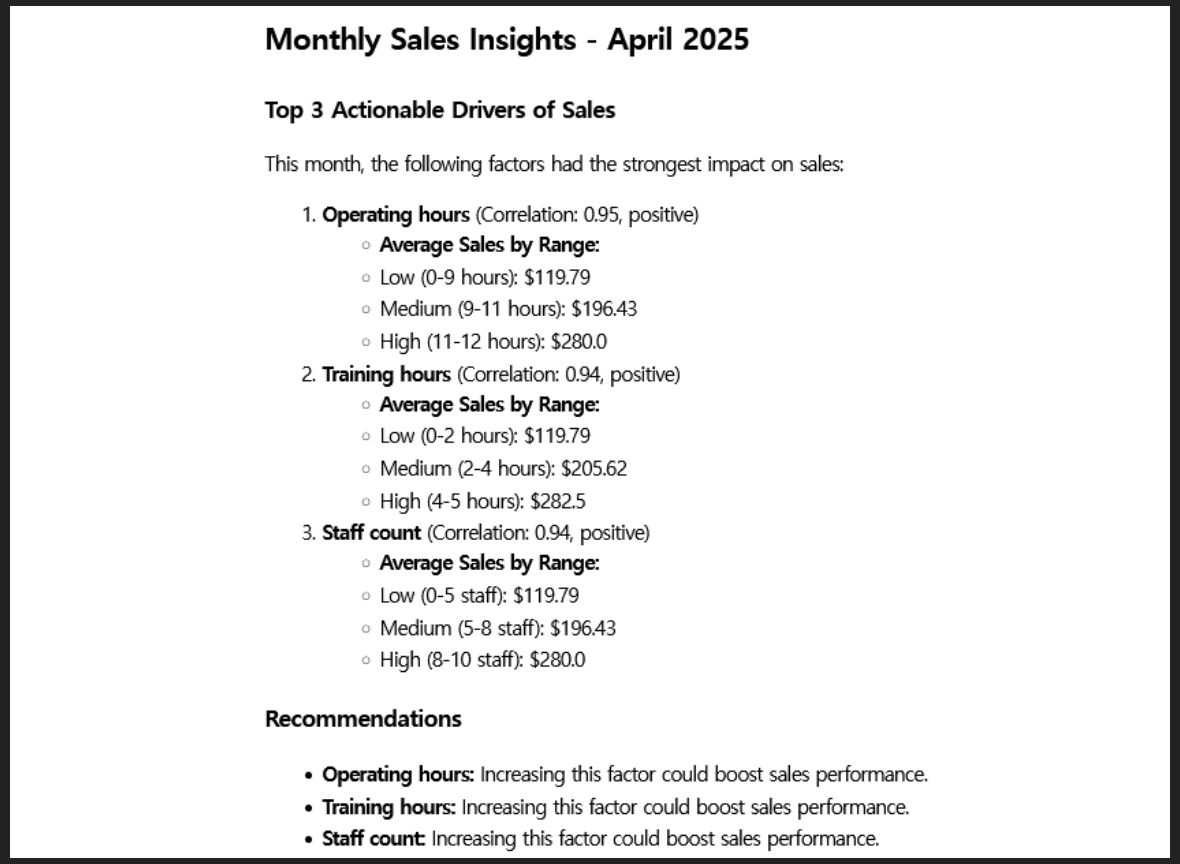


결과화면에서 의존 관계와 실행 결과 확인 가능

Airflow Demo: 최종 결과



메일 도착 화면



메일 내용