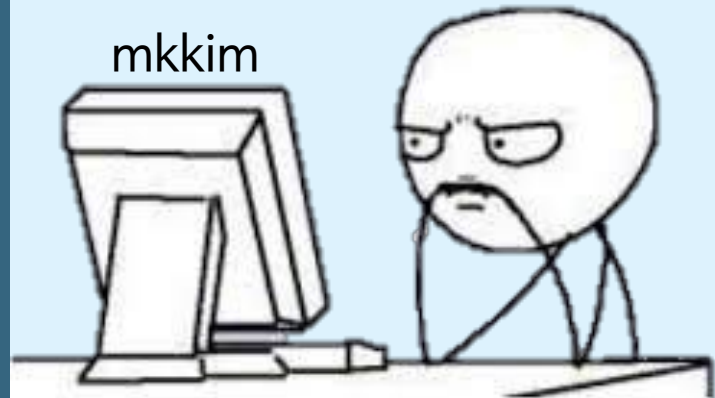


# 데이터 마트 전처리 작업 N 일에서 2시간으로 단축시킨 데이터 파이프라인 자동화의 비밀

feat. KISTEP 다차원 네트워크 시스템 PROJECT

**EUCLIDSOFT**  
유클리드소프트



**김민교** 주임

사업지원본부 솔루션 개발팀

## 발표자 소개



**김민교** 주임

사업지원본부 솔루션 개발팀

- KEARI : 방사성 폐기물 인증 지원 시스템 프로토 타입 개발

\* 프론트엔드, 백엔드 개발

- TIPA : 중소기업기술정보진흥원 다차원 네트워크 시스템

\* 데이터 엔지니어링

- STAR NET : 도서 데이터 다차원 네트워크 시스템

\* 데이터 엔지니어링, 분석 플랫폼 개발 및 배포

- Lime-HR : 사내 인사평가 시스템

\* 데이터 설계 및 프론트엔드, 백엔드 개발

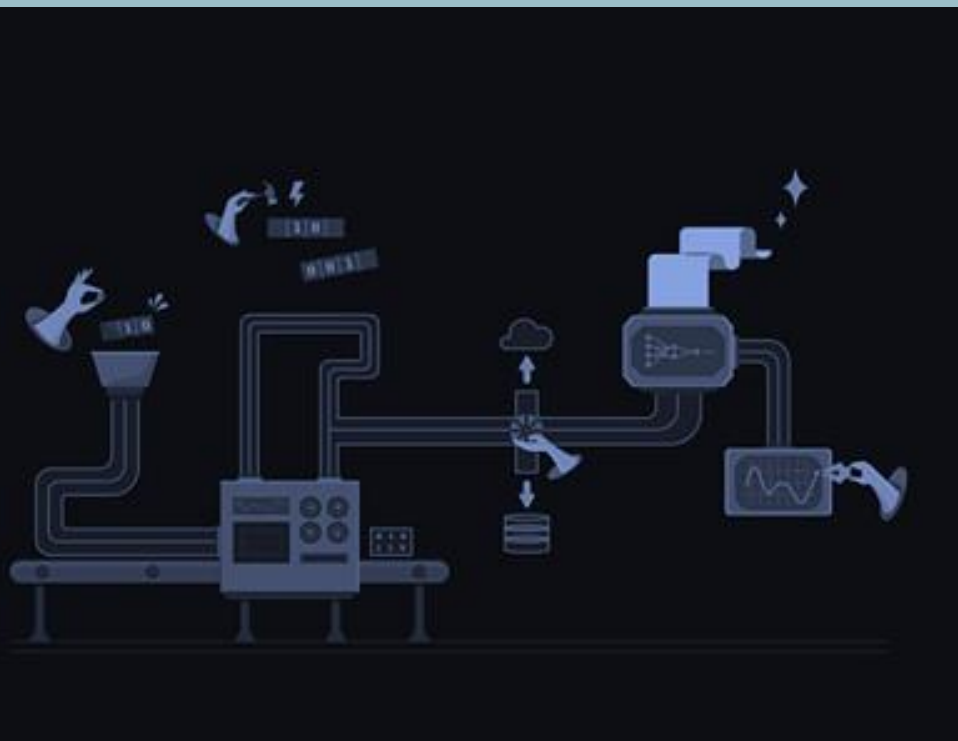
- KISTEP : 한국과학기술기획평가원 다차원 네트워크 시스템

\* 데이터 엔지니어링, 데이터 파이프라인 자동화, 데이터 마트 설계, 분석 플랫폼 개발

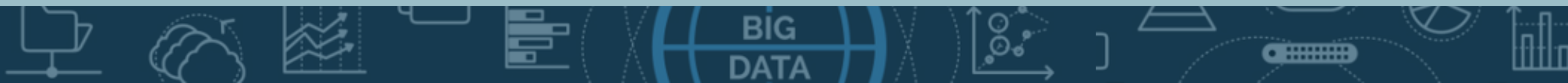
# CONTENTS

1. 데이터 파이프라인 요구사항
2. 요구사항 해결
3. 최종 파이프라인 소개

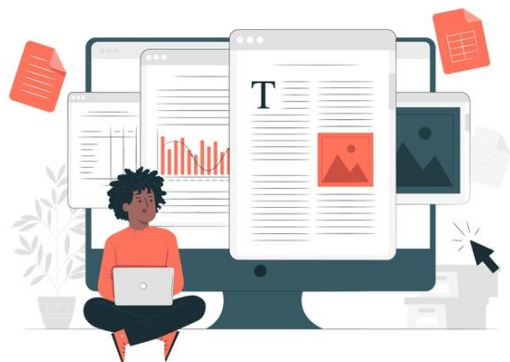
## 데이터 파이프라인 요구사항



- › 분석 데이터 제공을 위한 과제, 연구자 데이터 마트 설계
- › 증분, 수정, 삭제 되는 데이터를 매일 데이터 마트에 반영
- › 비정형 자연어 데이터 전 처리
- › 과학기술표준 자연어 분류모델을 통한 카테고리 예측
- › 데이터 마트 이력 관리
- › 안정성 있는 데이터 마트 제공 보장
- › 파이프라인 자동화 스케줄링 및 작업 워커와 로그 관리
- › 데이터 마트 무중단 서비스



## ➤ 분석 데이터 제공을 위한 과제, 연구자 데이터 마트 설계



## BM25 알고리즘

주어진 쿼리에 대해 문서와의 연관성을 평가하는 랭킹 함수로 사용되는 알고리즘

$$\text{score}(D, Q) = \sum_{i=1}^n \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}$$

IDF: 문서에 자주 등장하는 단어일수록 낮은 가중치를 준다.

TF: 문서에 자주 등장하는 단어일수록 높은 가중치를 준다.

$$\text{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

문서 검색  
단어, 문장, 조건 등등

문서 유사도 score 값 도출  
(BM25)

...

## ➤ 분석 데이터 제공을 위한 과제, 연구자 데이터 마트 설계



...



```
7 POST kistep_sbjt/_search
8 {
9   "_source": ["*"],
10  "sort": [{"_score": {"order": "desc"}}, {"stan_yr": {"order": "desc"}}],
11  "query": {
12    "bool": {
13      "must": [
14        {
15          "query_string": {
16            "fields": ["analysis_target_text"],
17            "query": "(*AI자율주행* OR *센서 기술* OR *컴퓨터
기술'^4 OR ((인공지능'^4))) OR ('실시간 결정 및 제어'
비전'^4 OR ('강화 학습'^4))) OR (('센서 기술'^4 OR
OR ('영상 처리'^4 OR ('영상 인식'^4 OR ('영상 분류
OR (CV2X'^4 OR (ADAS'^4 OR (GNSS'^4 OR (ITS'^4))) OR
처리'^4 OR ('컴퓨터 비전'^4 OR ('강화 학습'^4))) (
) OR ('영상 인식'^4 OR ('영상 분류'^4))) OR (('실
처리'^4 OR ('컴퓨터 비전'^4 OR ('강화 학습'^4))) (
))) OR (('컴퓨터 비전'^4 OR (('객체 인식'^4 OR ('
('V2X 통신'^4 OR ((DSRC'^4 OR (CV2X'^4 OR (ADAS'^4
```

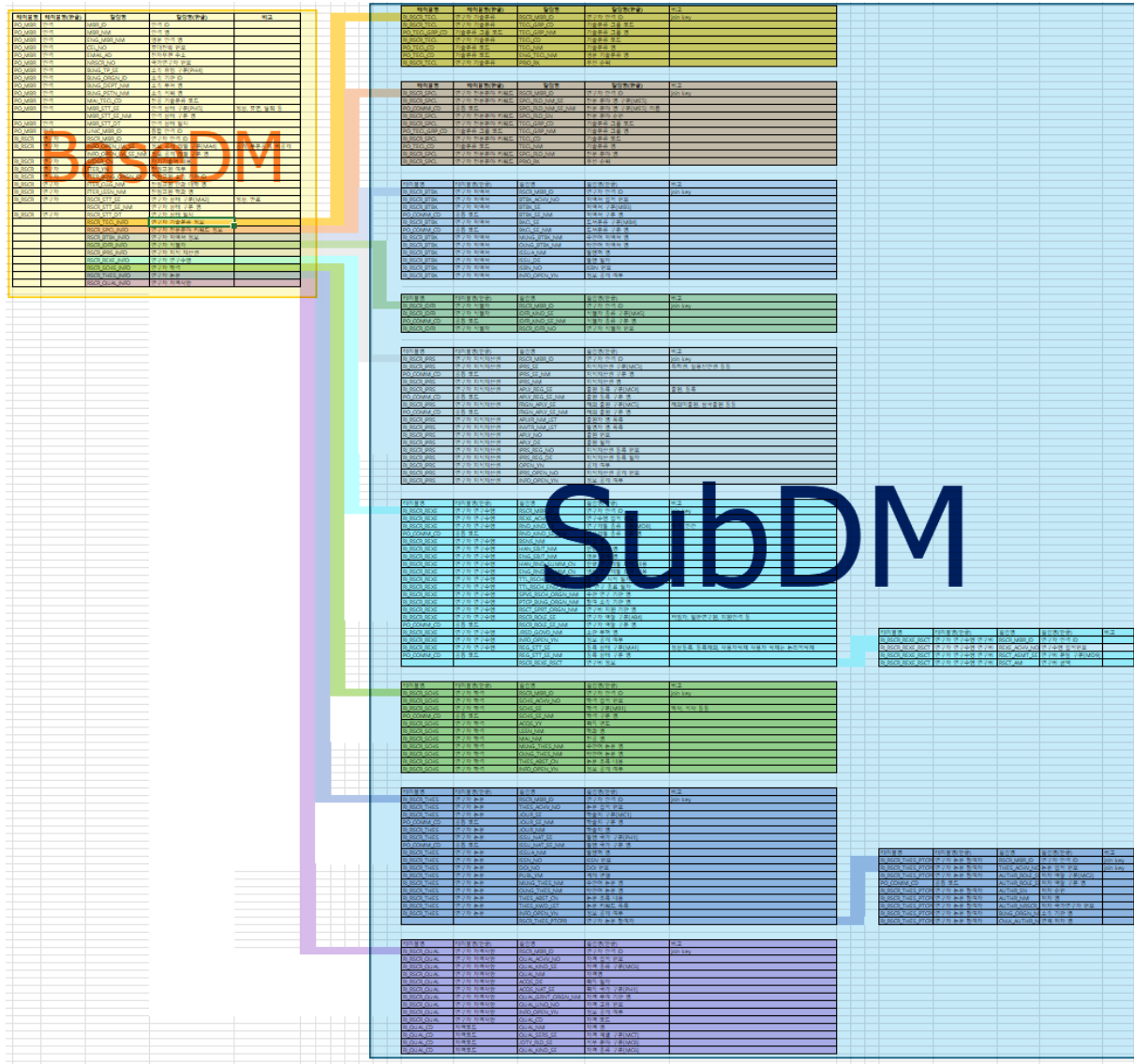


```
16   "max_score" : null,
17   "hits" : [
18     {
19       "_index" : "kistep_sbjt_20231126180312",
20       "_type" : "_doc",
21       "_id" : "1711071199",
22       "_score" : 899.3123,
23       "_routing" : "OB",
24       "_source" : { },
25       "sort" : [ ]
26     },
27     {
28       "_index" : "kistep_sbjt_20231126180312",
29       "_type" : "_doc",
30       "_id" : "1415145619",
31       "_score" : 711.9099,
32       "_routing" : "EF",
33       "_source" : { },
34       "sort" : [ ]
35     },
36     {
37       "_index" : "kistep_sbjt_20231126180312",
38       "_type" : "_doc",
39       "_id" : "1415157181",
40       "_score" : 700.591,
41       "_routing" : "EH",
42       "_source" : { },
43       "sort" : [ ]
44     },
45     {
46       "_index" : "kistep_sbjt_20231126180312",
47       "_type" : "_doc",
48       "_id" : "1425154909",
49       "_score" : 678.5991,
50       "_routing" : "LB",
51       "_source" : { },
52       "sort" : [ ]
53     },
54     {
55       "_index" : "kistep_sbjt_20231126180312",
56       "_type" : "_doc",
57       "_id" : "1345275780",
58       "_score" : 671.28796,
59       "_routing" : "EF",
60       "_source" : {
61         "doc_id" : "1345275780"
62       }
63     }
64   ]
65 }
```

문서 인덱싱

문서 검색

## ➤ 분석 데이터 제공을 위한 과제, 연구자 데이터 마트 설계



# MainDM



<https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.astera.com%2Ftype%2Fblog%2Fdata-mart-architecture%2F&psig=AOvWawZY5KzpVNYdOVg4V3JZns0&ust=1701154314521000&source=images&cd=ef&opt=89978449&ved=0CBMCqjxqfwoTCpJNpuIL44IDFQAAAAaDAABAF>

## › 증분, 수정, 삭제 되는 데이터를 매일 데이터 마트에 반영

### 1단계

- 원본 데이터 수집
- BaseDM, SubDM에 들어가야 하는 데이터 추출
- 컬럼별 비정형 문자열 데이터 전처리
- 컬럼별 default 값 셋팅
- 컬럼별 default type 셋팅
- null 처리
- BaseDM, SubDM 데이터 INSERT



### 2단계

- BaseDM, SubDM 추출
- BaseDM, SubDM 형변환
- MainDM으로 합치기
- analysis\_target\_text 생성
- predict\_target\_text 생성
- 과학기술표준분류 모델 예측
- MainDM INSERT *완료*



### 3단계

- MainDM 추출
- ES index 생성
- data indexing

정제 대상 컬럼 개수

과제 : 88 개

연구자: 164 개

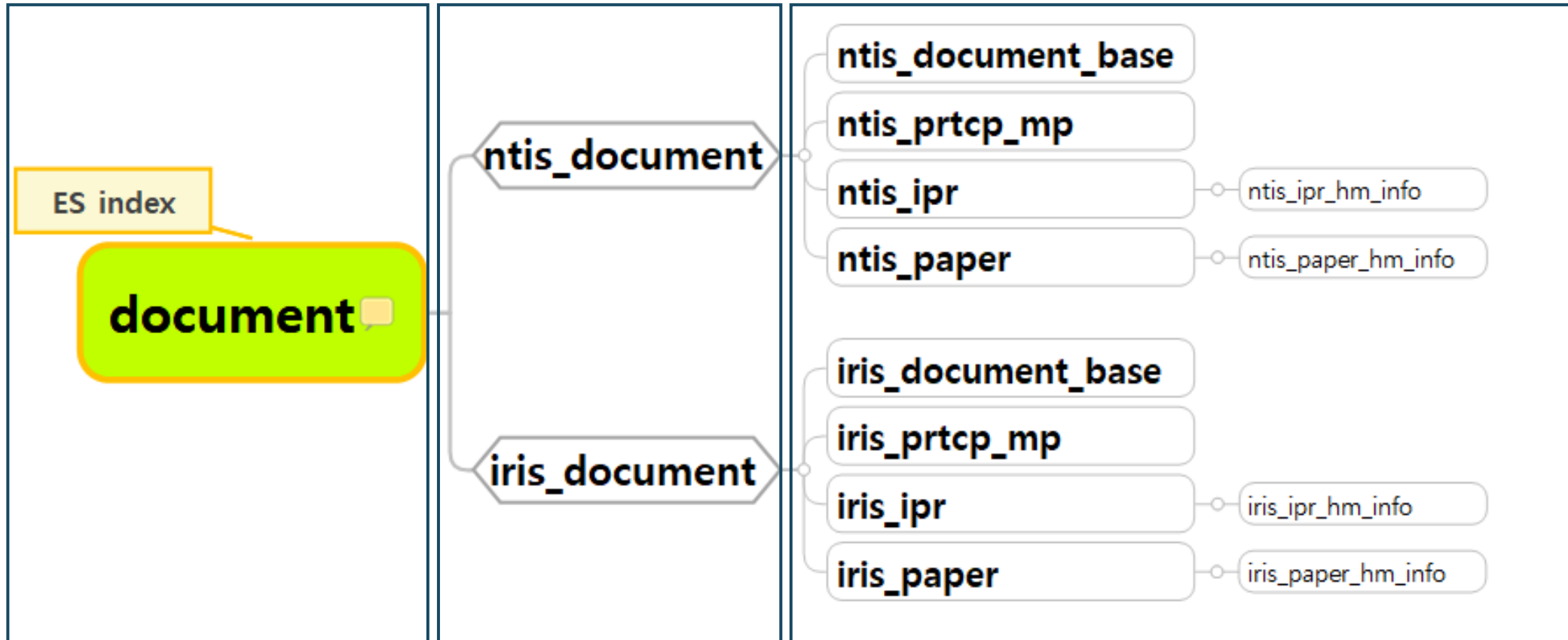


➤ 증분, 수정, 삭제 되는 데이터를 매일 데이터 마트에 반영

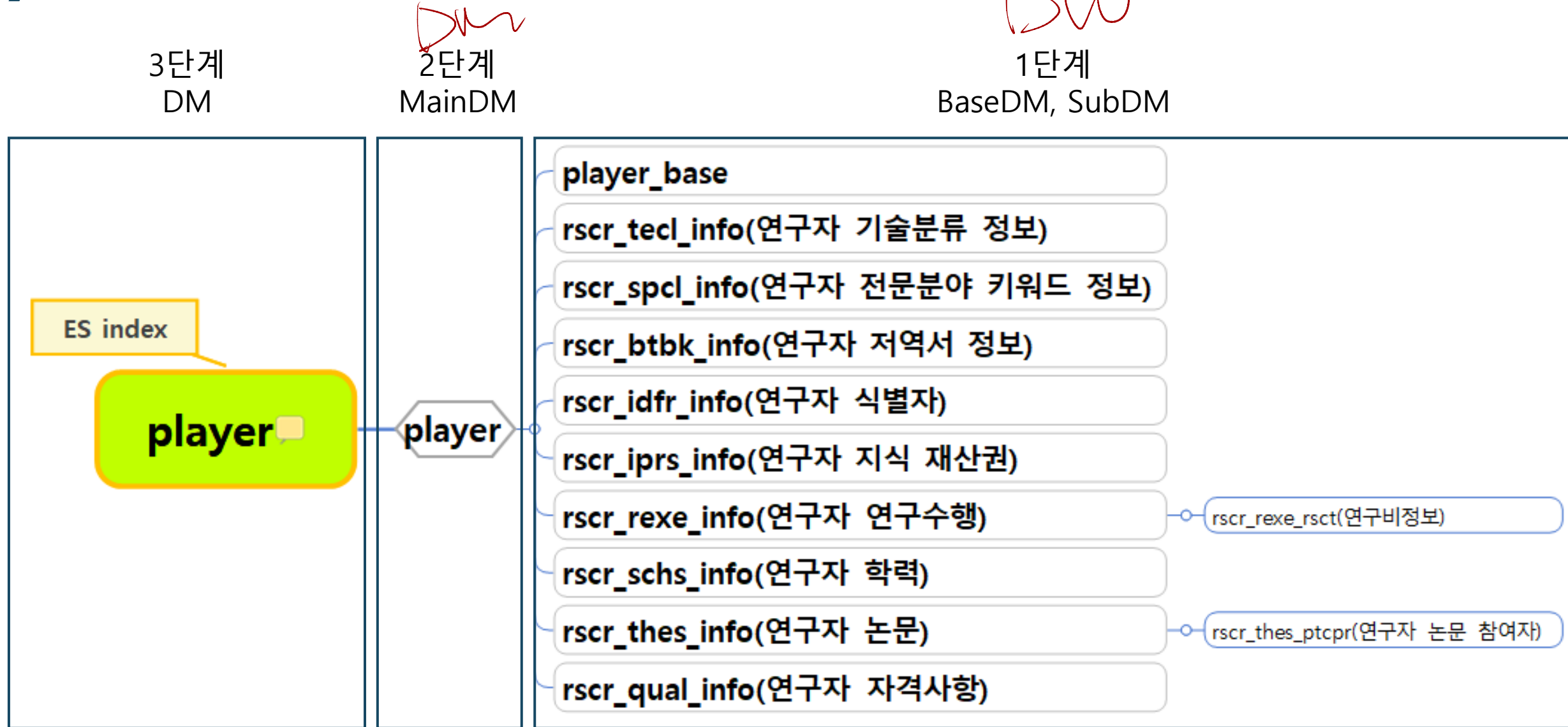
3단계  
DM

*DM*  
2단계  
MainDM

*DW*  
1단계  
BaseDM, SubDM



› 증분, 수정, 삭제 되는 데이터를 매일 데이터 마트에 반영



## ➤ 증분, 수정, 삭제 되는 데이터를 매일 데이터 마트에 반영

### - 개발 상황 가정

1. 데이터 품질을 위해 전처리 코드가 언제든지 수정될 수 있다.
2. 연구소에서 필요한 컬럼이 앞으로 더 추가되거나, 불필요한 컬럼이 데이터 마트에서 삭제될 수 있다.
3. 데이터 마트 대상 컬럼이 추가되고 삭제되는 과정에서 데이터 마트가 여러 번 초기화 될 수 있다.
4. 과학기술표준분류 대상이 되는 데이터가 추가되거나 삭제될 수 있다.

### - 개발 요구사항 도출

1. 데이터 마트에서 전처리 코드의 유지보수가 쉬워야 한다.
2. 데이터 마트 대상 컬럼을 추가하고 삭제하는 것이 쉬워야 한다.
3. 데이터 마트 초기화는 가능한 한 빠르게 이루어져야 한다.
4. 과학기술표준분류 대상 데이터가 수정되더라도 카테고리를 새로 넣는데 시간적 부담이 적어야 한다.

수백만 건의 데이터를 join 쿼리와 함께 문자열을 정제하기 위해 SUBSTRING, CONCAT, IF 등의 메서드들을 사용했고, 한 ROW를 만들기에는 DB엔진의 처리 효율이 좋지 않았다.

## ▶ 비정형 자연어 데이터 전 처리

### - 2차 구상 (실패)

python과 pandas 라이브러리를 사용  
공통 코드는 모듈화 하고  
처리 과정은 모놀리식으로 구성



BaseDM을 생성하는 데 30분 미만으로 단축



SubDM들 까지 처리하는데 과학기술표준분류 모  
델 실행 외에 1시간 ~ 1시간 30분으로 단축

### - 문제

python과 pandas 라이브러리를 사용하여 처리 속도를 높였지만

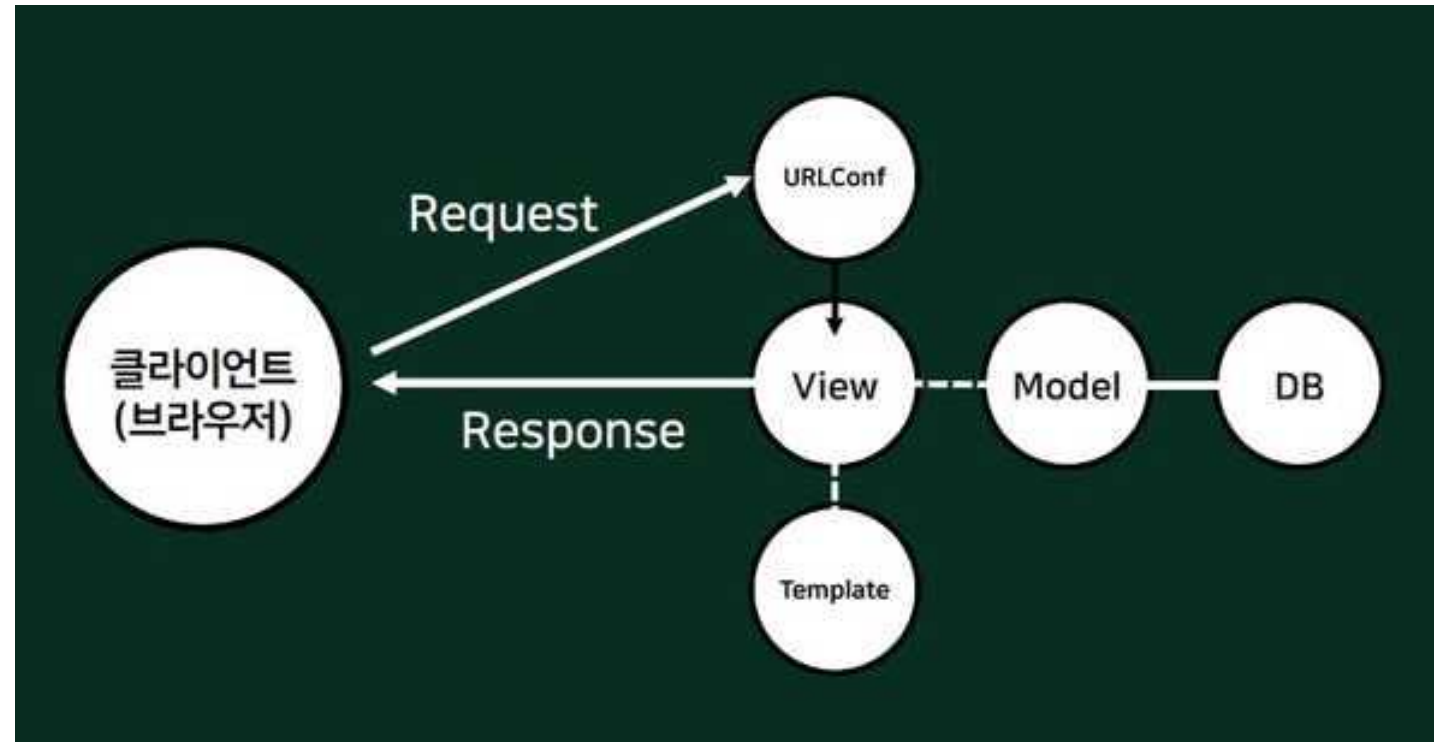
- 중복 되는 코드가 많음
- 처리 과정을 한눈에 볼 수 없음
- 수정을 위해 봐야 할 앞 뒤 맥락이 많음
- 유지보수를 위해 흐름을 읽기가 어려움



## ➤ 비정형 자연어 데이터 전 처리

- 3차 구상 전처리 코드의 모듈화 & 패턴화

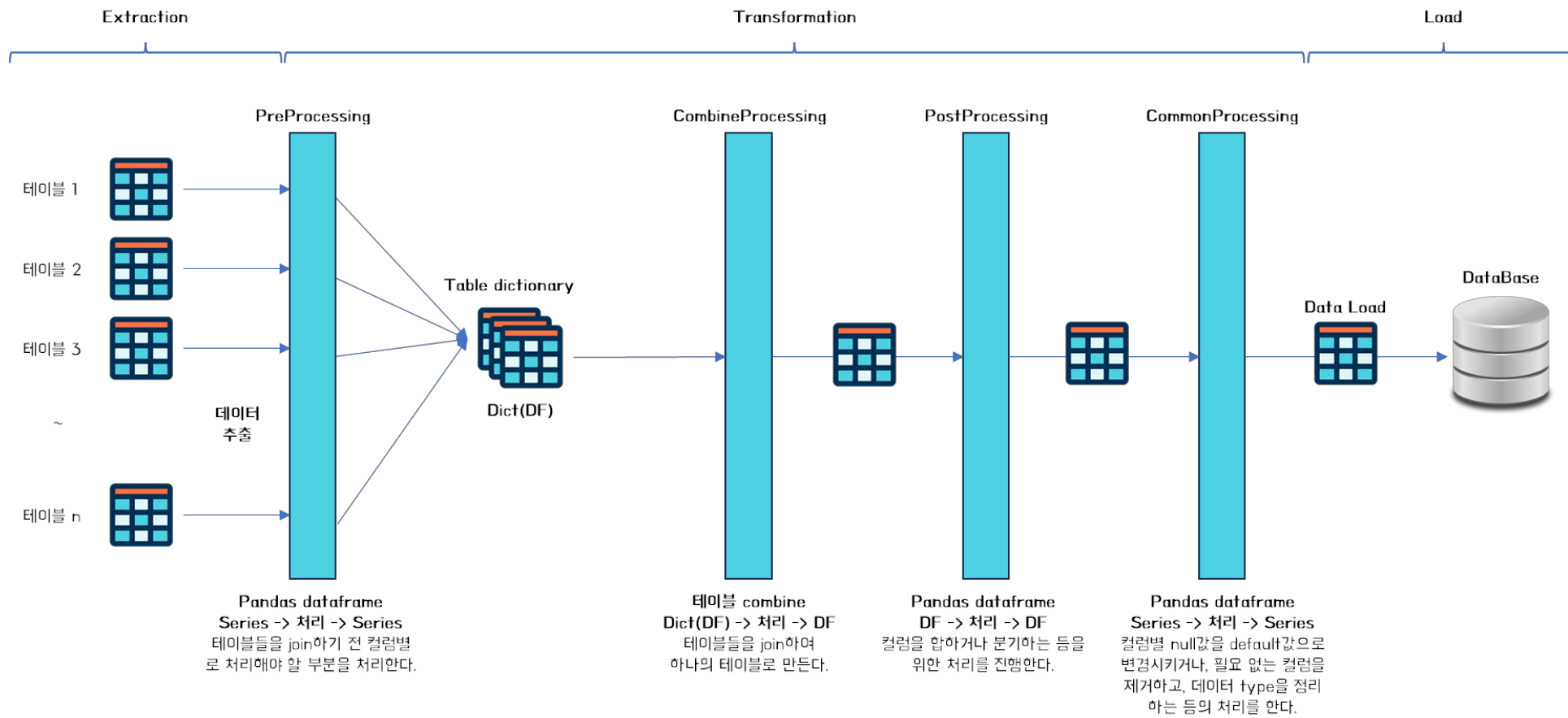
### django의 MTV 패턴



- URLConf : api와 처리 로직 맵핑
- View : back-end 비즈니스 로직 처리
- Template : 화면 구성
- Model : DB 연결

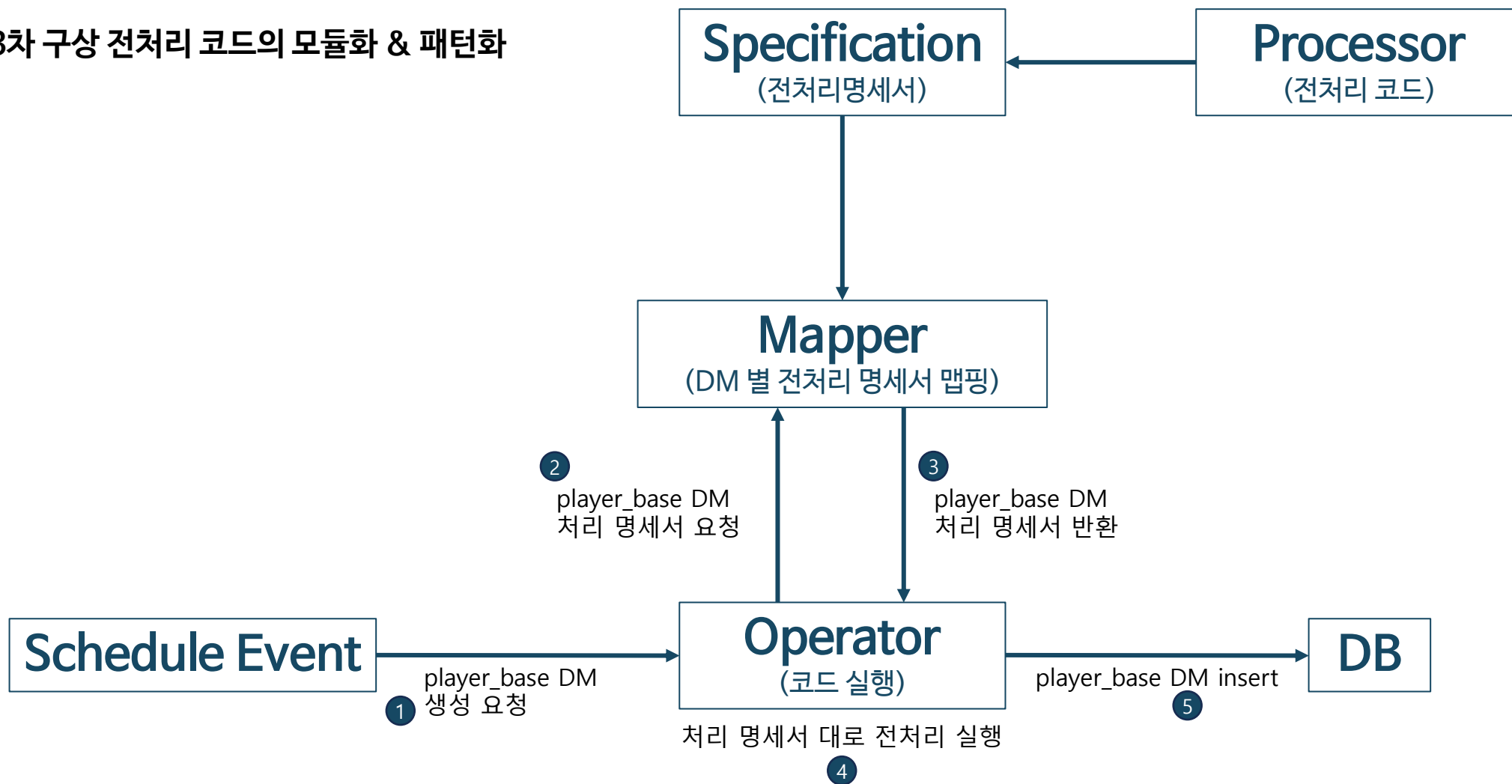
## 비정형 자연어 데이터 전 처리

### - 3차 구상 전처리 코드의 모듈화 & 패턴화



## ➤ 비정형 자연어 데이터 전 처리

- 3차 구상 전처리 코드의 모듈화 & 패턴화





## 비정형 자연어 데이터 전 처리 - 3차 구상 전처리 코드의 모듈화 & 패턴화

```
from .Processors.PreProcessor import PreProcessor
from .Processors.PostProcessor import PostProcessor
from .Processors.CommonProcessor import CommProcessor
```

```
class DocumentNtisSpecification:
    def __init__(self, pj_hist_mapper={}, cd_dtl_mapper={}, offc_cd_mapper={}) -> None:
        self.pre = PreProcessor(pj_hist_mapper=pj_hist_mapper, cd_dtl_mapper=cd_dtl_mapper)
        self.post = PostProcessor(offc_cd_mapper=offc_cd_mapper)
        self.comm = CommProcessor()
        self.DOCUMENT_SPECIFICATION = {
            "mappings" : {
                ~
                "pjt_prfrm_org_nm": {# <- DataMart 컬럼명
                    "comment" : "과제수행기관명", # <- DataMart 컬럼 코멘트
                    "source" : [
                        {
                            "tbl1" : "TIA_PJ_IRIS", # <- 원본 소스 테이블 명
                            "col" : "MAIN_RSCH_ORG_NM" # <- 원본 소스 컬럼명
                        }
                    ],
                    "type": "str", # <- DataMart 컬럼의 기본 타입
                    "default_value": "", # <- DataMart 컬럼의 기본 값
                    "pre_processing" : [self.pre.make_pre_pjt_prfrm_org_nm],
                    "post_processing" : [self.post.make_prog_mstr_cd, self.post.make_prog_mstr_nm],
                    "common_processing": [self.comm.l_r_strip, self.comm.trash_space_to_one_space]
                },
                ~
            }
            "combine_settings" : [
                {
                    "mode": "merge",
                    "how": "left",
                    "left": "TIA_PJ_IRIS",
                    "right": "TIA_INV_ANA_EVAL_TRGT_BZ_IRIS",
                    "on": ["org_bdgt_prog_cd", "stan_yr"]
                },
                ~
            ]
            "drop_target_cols" : ["prog_mstr_cd_NEW", "prog_mstr_cd_OLD"]
        }
```

Pre, Post, Common 프로세서의 객체를 생성한다.

"mappings" : DataMart의 컬럼별 기본 정보, 처리 내역 맵핑 dictionary  
"pre\_processing", "post\_processing", "common\_processing"에는 각각 위에서 선언한 프로세서 객체의 매서드를 리스트로 나열한다. 먼저 오는 매서드 부터 처리된다.

여러 테이블을 합쳐서 하나의 datamart로 만들 때 필요한 내용을 정의한다. Argument들은 pandas의 merge 메서드, concat 메서드와 동일하다.

전처리 과정에서는 필요했지만 최종적으로 제거해야 할 컬럼목록을 리스트로 나열한다.

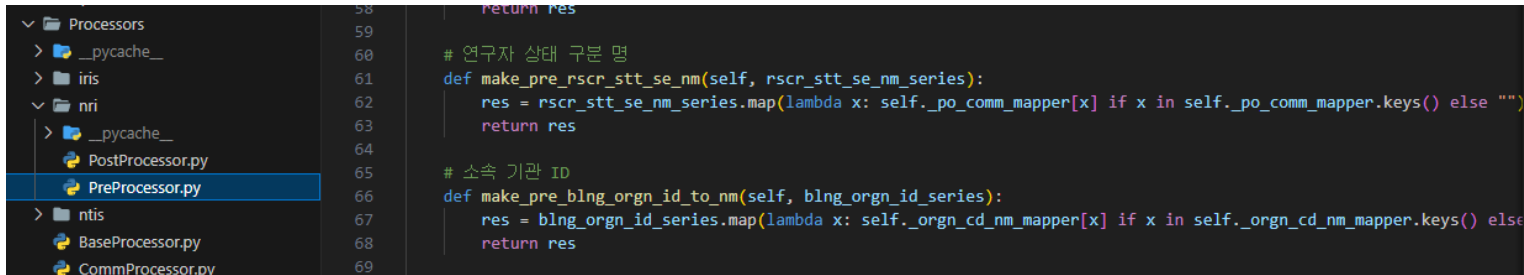
```
def get_document_specification(self):
    return self.DOCUMENT_SPECIFICATION
```

## ▶ 비정형 자연어 데이터 전 처리

### - 3차 구상 전처리 코드의 모듈화 & 패턴화 후 코드 작성의 변화


예시 ) 데이터 품질을 위해 player\_base 테이블의 mbr\_nm(연구자 이름)의 추가적인 전처리가 필요한 상황

#### 1. PreProcessor.py 에 mbr\_nm을 처리하기 위한 코드 추가



```
58     return res
59
60     # 연구자 상태 구분 명
61     def make_pre_rscr_stt_se_nm(self, rscr_stt_se_nm_series):
62         res = rscr_stt_se_nm_series.map(lambda x: self._po_comm_mapper[x] if x in self._po_comm_mapper.keys() else "")
63         return res
64
65     # 소속 기관 ID
66     def make_pre_blng_orgn_id_to_nm(self, blng_orgn_id_series):
67         res = blng_orgn_id_series.map(lambda x: self._orgn_cd_nm_mapper[x] if x in self._orgn_cd_nm_mapper.keys() else "")
68         return res
69
```

#### 2. 해당 DM의 Specification에 해당하는 단계에 해당 함수 명을 추가



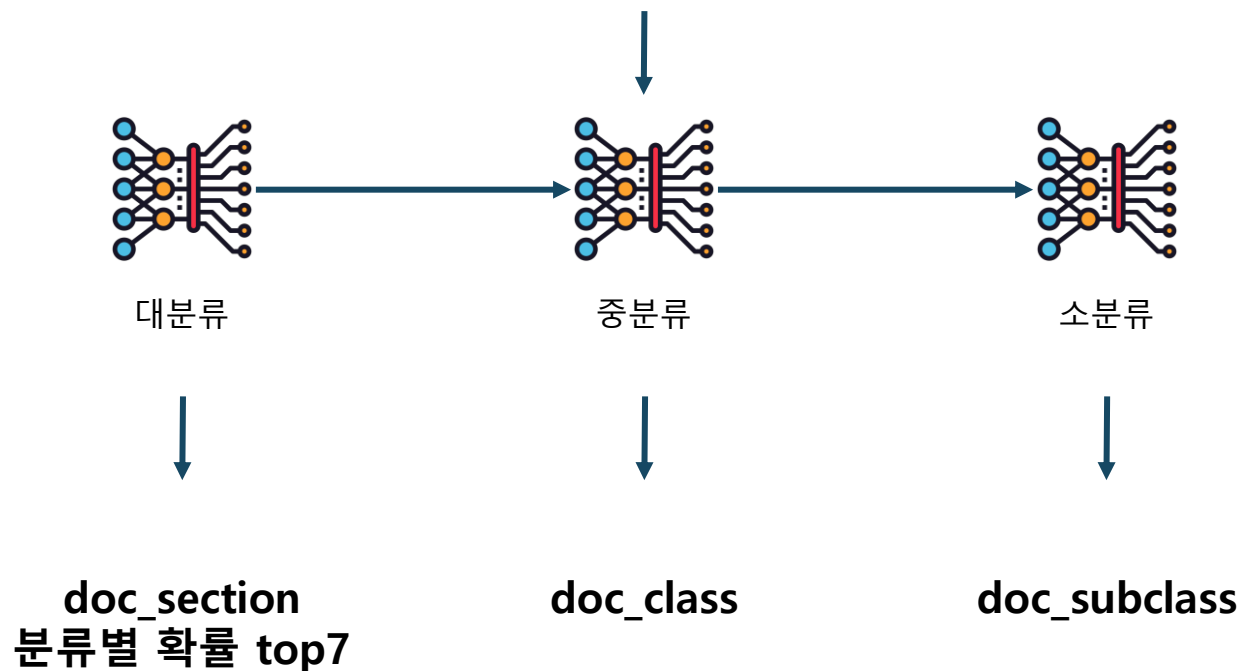
```
55
56
57
58
59
60
61     "mbr_nm": {
62         "comment" : "인력명",
63         "source" : [
64             {
65                 "tbl": "PO_MBR",
66                 "col": "MBR_NM"
67             }
68         ],
69         "type": "str",
70         "default_value": "",
71         "pre_processing": [],
72         "post_processing": [],
73         "common_processing": [self.comm.l_r_strip]
74     },
75
```

전처리 코드 함수 객체 추가

## ➤ 과학기술표준 자연어 분류모델을 통한 카테고리 예측

```
PLAYER_PREDICT_TARGET_COL = {  
    "base": ["maj_tec1_nm"],  
    "rscr_tec1_info" : ([ "tec1_nm", []],  
    "rscr_iprs_info" : ([ "iprs_nm", "iprs_summ_cn", ["aply_de"]],  
    "rscr_rexe_info" : ([ "han_sbjt_nm", ["ttl_rsch_str_de"]],  
    "rscr_thes_info" : ([ "m1ng_thes_nm", "thes_kwd_lst", ["publ_ym"]])  
}
```

predict\_target\_text



과제 데이터 약 100만개

CPU 사용 : 약 1주일 이상

GPU 사용 : 약 1일

## 과학기술표준 자연어 분류모델을 통한 카테고리 예측

## 초기화용 데이터 마트 구성

[illegible]

## ▶ 데이터 마트 이력 관리

1. 매일 구성되는 BaseDM, SubDM마다 dm\_sn(데이터 마트 시리얼 넘버)을 부여

```
kistep_mbr_20231126183358
kistep_sbjt_20231126180312
player_init_cate_202311261834
player_202311261823
rscr_thes_info_202311261800
player_base_202311261800
rscr_iprs_info_202311261800
iris_init_cate_202311261803
rscr_spcl_info_202311261800
ntis_init_cate_202311261801
rscr_tecl_info_202311261800
rscr_rexe_info_202311261800
iris_prtcp_mp_202311261800
iris_document_base_202311261800
```

2. 데이터마트의 상태를 기록할 수 있는 테이블을 구성하여 BaseDM, SubDM, MainDM, index, init\_category 등 여러 데이터 마트 상태를 한곳에서 확인 가능

no	dm_nm	dm_sn	dm_stat	dm_stat_desc	dm_maker	dm_se	created_dt	updated_dt
1,233	bz_map	bz_map_ntis_document_2023112620	activated		airflow	DB	2023-11-27 05:00:56.000	2023-11-27 05:00:56.000
1,232	bz_map	bz_map_iris_document_20231126200	activated		airflow	DB	2023-11-27 05:00:29.000	2023-11-27 05:00:29.000
1,231	kistep_mbr	kistep_mbr_20231126183358	activated	kistep_mbr init	airflow	ES	2023-11-27 03:59:33.000	2023-11-27 03:59:33.000
1,230	kistep_sbjt	kistep_sbjt_20231126180312	activated	kistep_sbjt init	airflow	ES	2023-11-27 03:51:41.000	2023-11-27 03:51:41.000
1,229	player_init_cate	player_init_cate_202311261834	activated	증분, 수정, 삭제 데이터 반영된 초	airflow	DB	2023-11-27 03:34:44.000	2023-11-27 03:34:44.000
1,228	player	player_202311261823	activated	player dm 수정, 증분, 삭제 업데이트	airflow	DB	2023-11-27 03:33:52.000	2023-11-27 03:33:52.000
1,227	rscr_thes_info	rscr_thes_info_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:13:22.000	2023-11-27 03:13:22.000
1,226	player_base	player_base_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:04:34.000	2023-11-27 03:04:34.000
1,225	rscr_iprs_info	rscr_iprs_info_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:03:52.000	2023-11-27 03:03:52.000
1,224	iris_init_cate	iris_init_cate_202311261803	activated	증분, 수정, 삭제 데이터 반영된 초	airflow	DB	2023-11-27 03:03:18.000	2023-11-27 03:03:18.000
1,223	rscr_spcl_info	rscr_spcl_info_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:02:11.000	2023-11-27 03:02:11.000
1,222	ntis_init_cate	ntis_init_cate_202311261801	activated	증분, 수정, 삭제 데이터 반영된 초	airflow	DB	2023-11-27 03:01:45.000	2023-11-27 03:01:45.000
1,221	rscr_tecl_info	rscr_tecl_info_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:00:46.000	2023-11-27 03:00:46.000
1,220	rscr_rexe_info	rscr_rexe_info_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:00:41.000	2023-11-27 03:00:41.000
1,219	iris_prtcp_mp	iris_prtcp_mp_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:00:40.000	2023-11-27 03:00:40.000
1,218	iris_document_base	iris_document_base_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:00:32.000	2023-11-27 03:00:32.000
1,217	iris_paper	iris_paper_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:00:17.000	2023-11-27 03:00:17.000
1,216	rscr_idfr_info	rscr_idfr_info_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:00:15.000	2023-11-27 03:00:15.000
1,215	iris_paper_hm_info	iris_paper_hm_info_202311261800	activated	daily dm operating job	airflow	DB	2023-11-27 03:00:12.000	2023-11-27 03:00:12.000
1,214	bz_map	bz_map_ntis_document_2023112520	deleted		airflow	DB	2023-11-26 05:00:52.000	2023-11-27 05:00:10.000
1,213	bz_map	bz_map_iris_document_20231125200	deleted		airflow	DB	2023-11-26 05:00:25.000	2023-11-27 05:00:10.000

## ➤ 안정성 있는 데이터 마트 제공 보장

1. 전처리가 실패할 경우, 기존 데이터마트의 상태를 activated 하도록 구성
2. 이렇게 필요한 이력의 개수는 constants.py 에서 아래와 같이 상수로 관리하고 있음
3. 에러가 발생한 경우 airflow에서 모니터링하고 로그를 확인할 수 있도록 구성됨(airflow 파트에서 설명)

```
# DB에서 activated 데이터 제외 데이터마트가 가지고 있을 수 있는 데이터 기록의 최대 개수
# ex) 2일 경우 -> activated dm_sn 1개, deactivated dm_sn 2개
NUMBER_OF_DM_RECORD_LIMIT = 2
# ES에서 activated 데이터 제외 인덱스 기록 최대 개수
# ex) 2일 경우 -> activated index 1개, deactivated index 2개
NUMBER_OF_ES_RECORD_LIMIT = 2
```

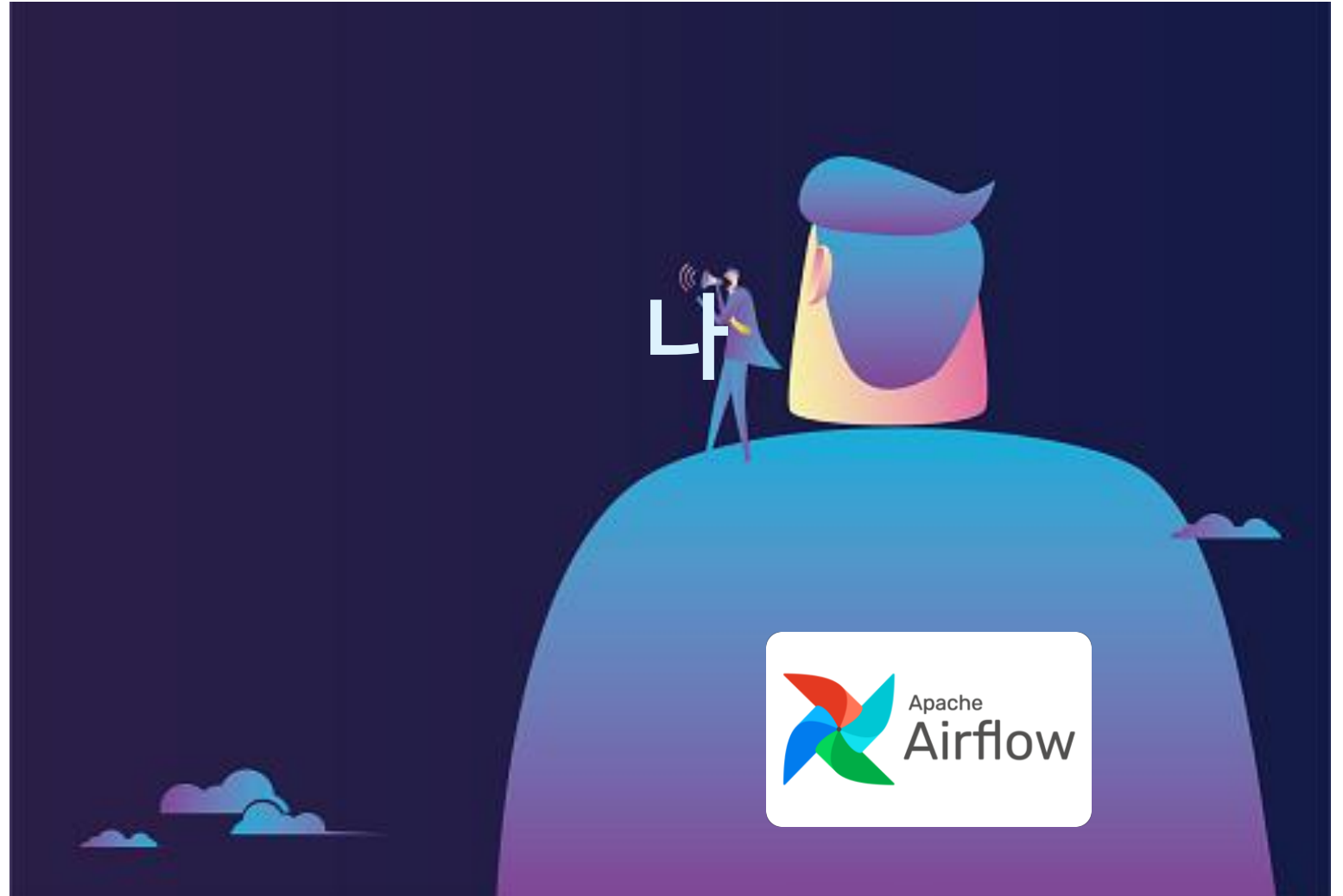
## ➤ 파이프라인 자동화 스케줄링 및 작업 워커와 로그 관리

파이프라인 자동화  
이벤트 스케줄링  
워커 자원 관리  
로그관리

·  
·  
·

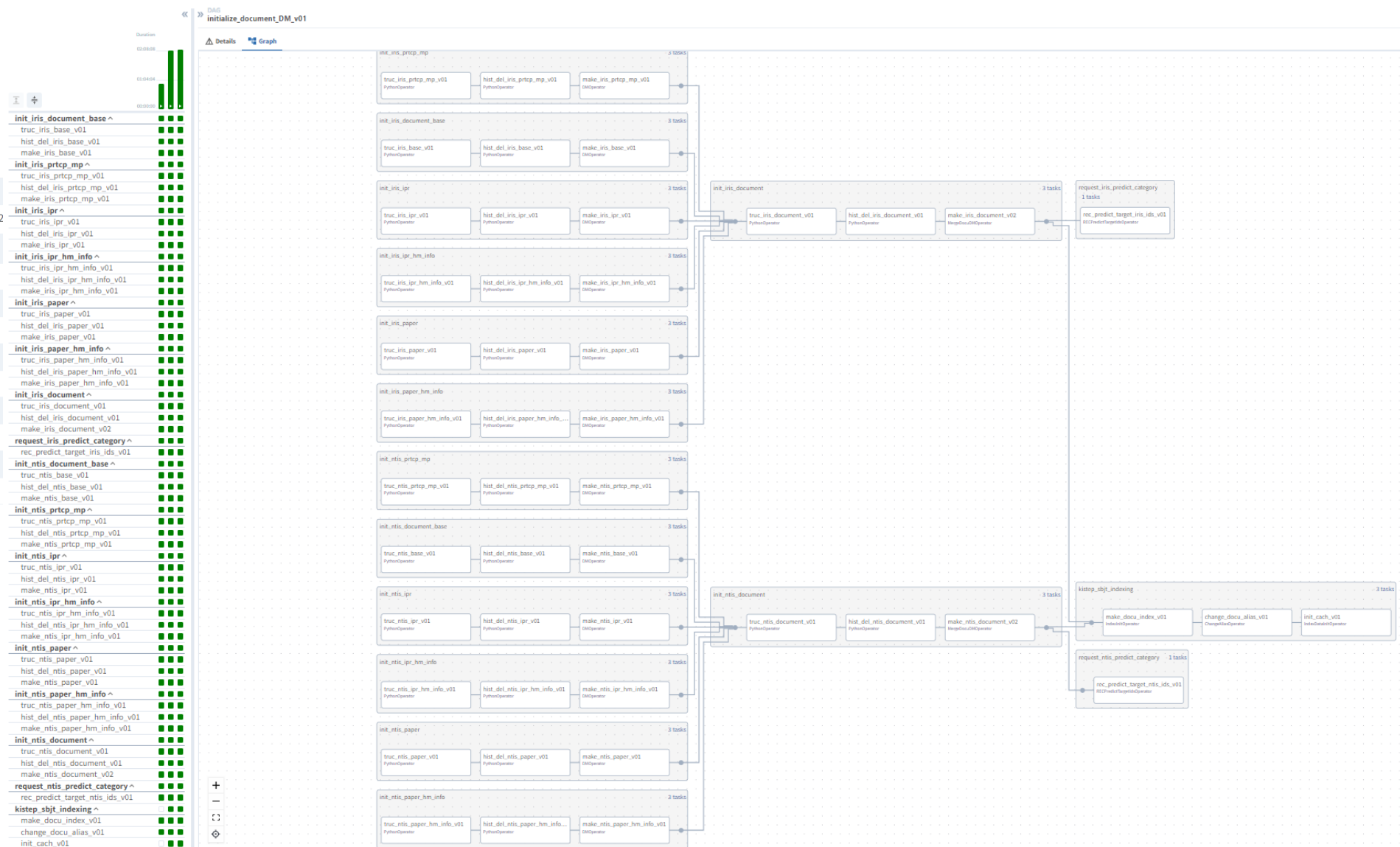
솔직히  
1인 개발자의 능력 및 시간적 한계에 직면

-> Airflow 툴을 도입하여 해결



# 최종 파이프라인 소개

## 1. 과제 데이터마트 초기화 work flow

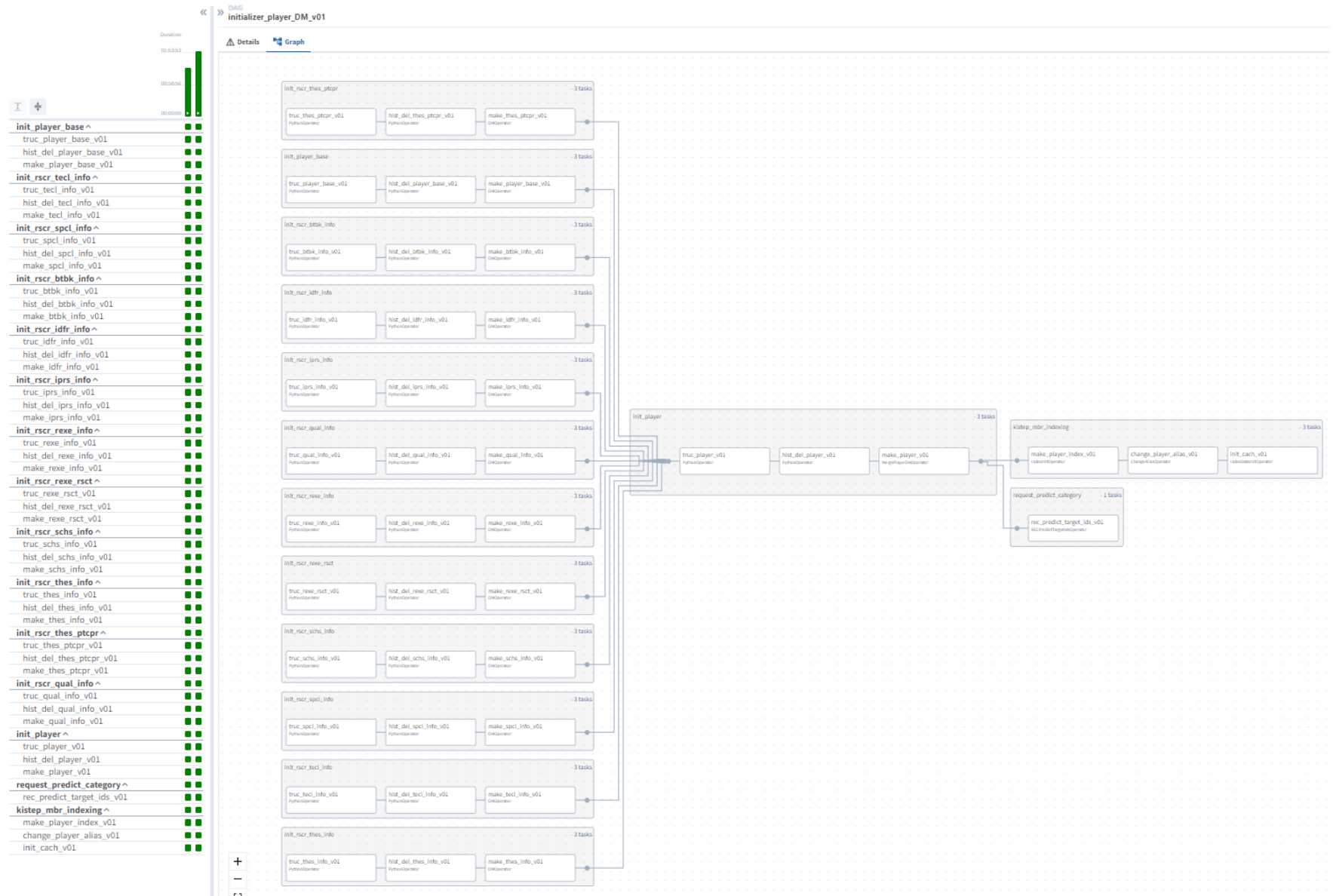




# 최종 파이프라인 소개

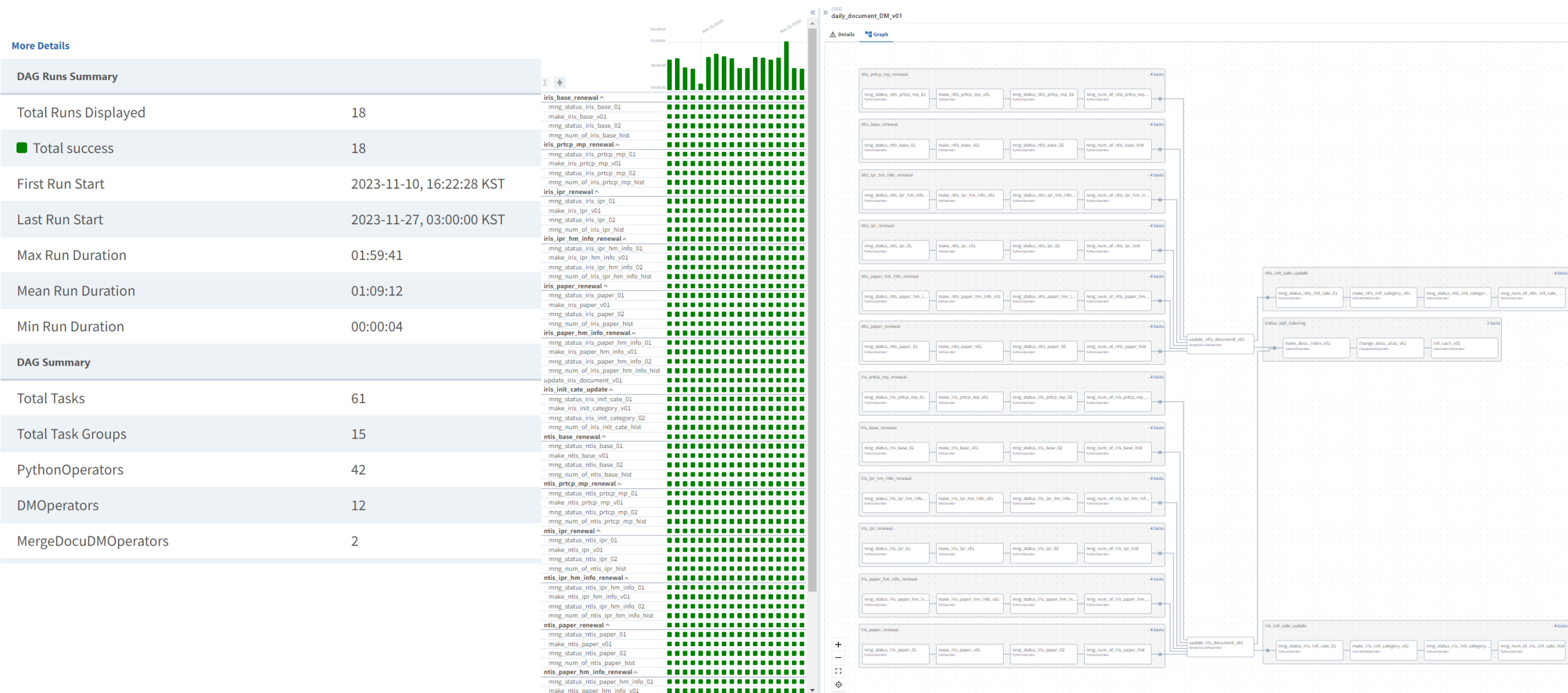
## 1. 연구자 데이터마트 초기화 work flow

Status	success
Run ID	manual__2023-11-20T06:5
Run type	manual
Run duration	01:53:53
Last scheduling decision	2023-11-20, 17:44:54 KST
Queued at	2023-11-20, 15:51:00 KST
Started	2023-11-20, 15:51:00 KST
Ended	2023-11-20, 17:44:54 KST
Data interval start	2023-11-20, 15:50:59 KST
Data interval end	2023-11-20, 15:50:59 KST
Externally triggered	True
Run config	None



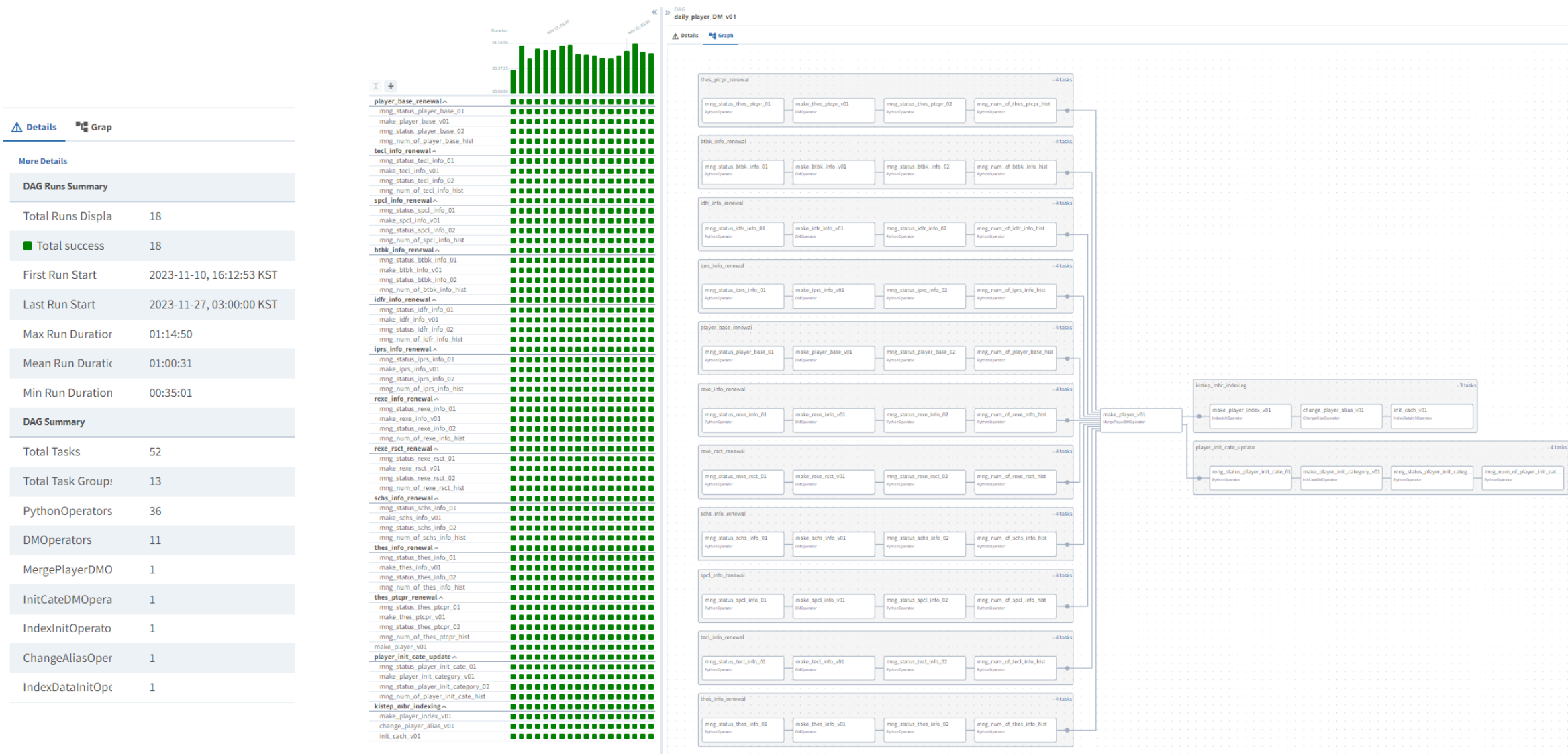
# 최종 파이프라인 소개

## 1. 과제 데이터마트 증분, 수정, 삭제 업데이트 work flow



## 최종 파이프라인 소개

### 1. 연구자 데이터마트 증분, 수정, 삭제 업데이트 work flow



# 최종 파이프라인 소개

kistep data work flow 전체

DAGs

ALL 11ACTIVE 11PAUSED 0

Filter DAGs by tag

Search DAGs

Auto-refresh

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
<div><div>daily_bz_map_DM_v01</div><div>bz_mapprod</div></div>	airflow	<div><div>18</div></div>	05****	2023-11-26, 05:00:00	2023-11-27, 05:00:00	<div><div>6</div></div>	<div><div></div><div></div></div>	
<div><div>daily_dic_maker_to_nori_v01</div><div>produserdict</div></div>	airflow	<div><div>16</div><div>2</div></div>	01****	2023-11-26, 01:00:00	2023-11-27, 01:00:00	<div><div>9</div></div>	<div><div></div><div></div></div>	
<div><div>daily_document_DM_v01</div><div>dailyiris_documentkistep_sbitttis_documentprod</div></div>	airflow	<div><div>18</div></div>	03****	2023-11-26, 03:00:00	2023-11-27, 03:00:00	<div><div>61</div></div>	<div><div></div><div></div></div>	
<div><div>daily_log_delete_v01</div><div>logprod</div></div>	airflow	<div><div>7</div></div>	@daily	2023-11-26, 00:00:00	2023-11-27, 00:00:00	<div><div>1</div></div>	<div><div></div><div></div></div>	
<div><div>daily_player_DM_v01</div><div>dailykistep_mbrplayerprod</div></div>	airflow	<div><div>18</div></div>	03****	2023-11-26, 03:00:00	2023-11-27, 03:00:00	<div><div>52</div></div>	<div><div></div><div></div></div>	
<div><div>daily_spotfire_data_v01</div><div>prodspotfire</div></div>	airflow	<div><div>12</div></div>	05****	2023-11-26, 05:00:00	2023-11-27, 05:00:00	<div><div>3</div></div>	<div><div></div><div></div></div>	
<div><div>daily_sync_iris_tables_v01</div><div>documentirisiris_documentprodsync</div></div>	airflow	<div><div>19</div></div>	02****	2023-11-26, 02:30:00	2023-11-27, 02:30:00	<div><div>11</div></div>	<div><div></div><div></div></div>	
<div><div>daily_sync_nri_table_v01</div><div>nriplayerprodsync</div></div>	airflow	<div><div>20</div></div>	02****	2023-11-27, 15:33:06	2023-11-27, 02:40:00	<div><div>12</div></div>	<div><div></div><div></div></div>	
<div><div>initialize_document_DM_v01</div><div>documentiris_documentkistep_sbitttis_documentprod</div></div>	airflow	<div><div>3</div></div>	None	2023-11-20, 15:50:52		<div><div>47</div></div>	<div><div></div><div></div></div>	
<div><div>initializer_player_DM_v01</div><div>kistep_mbrplayerprod</div></div>	airflow	<div><div>2</div></div>	None	2023-11-20, 15:50:59		<div><div>43</div></div>	<div><div></div><div></div></div>	
<div><div>player_profiling_batch_v2</div><div>prodtopic_modeling</div></div>	airflow	<div><div>2064</div><div>1</div></div>	*/5****	2023-11-27, 20:30:00	2023-11-27, 20:35:00	<div><div>2</div><div>1</div><div>4</div></div>	<div><div></div><div></div></div>	

1

Showing 1-11 of 11 DAGs

## ➤ airflow 파이프라인 구축기

to be continue...