



# [IBM StreamSets] DB to DB Pipeline 생성 가이드

## References

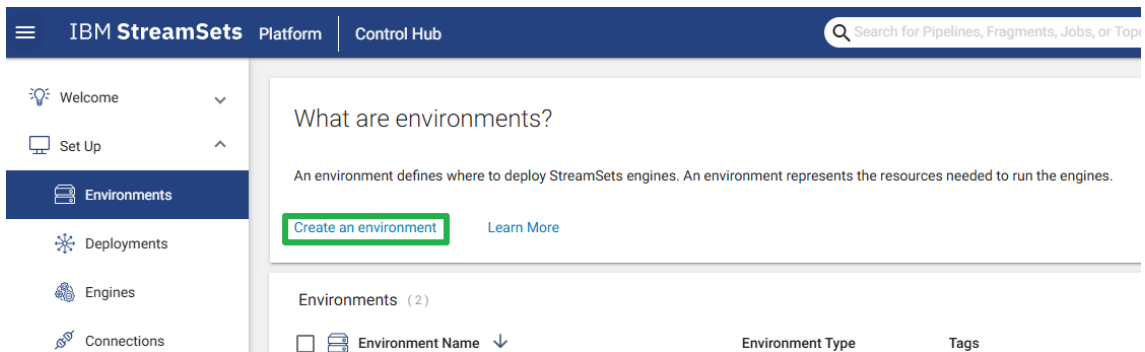
- [IBM StreamSets - Data Collector Engine Guide](#)
- [IBM StreamSets - Transformer Engine Guide](#)

## ※ 사전 세팅

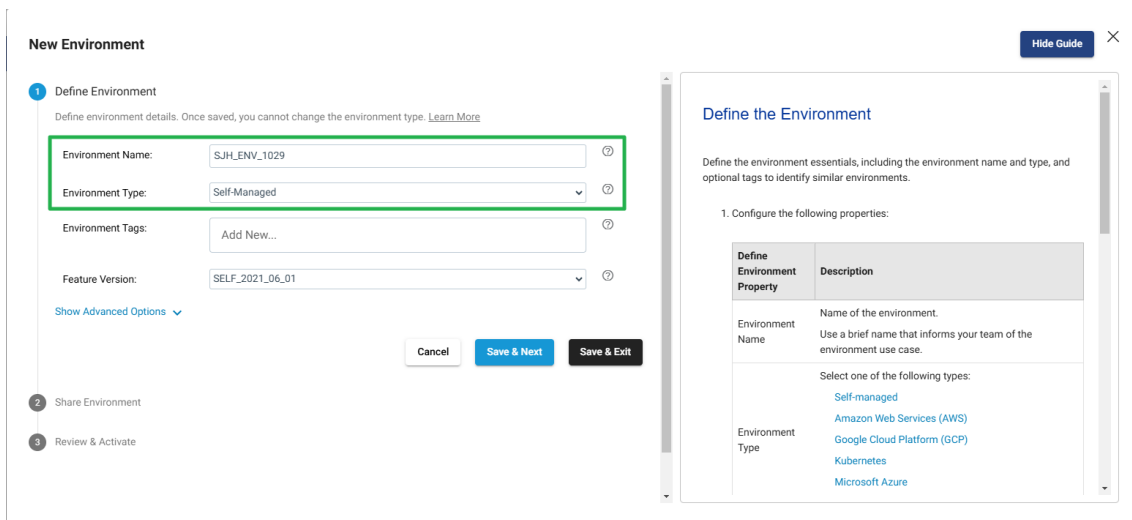
- DB 구성 - 가이드 에서는 PostgreSQL 사용
- Self Managed VM 최소 사양 - [2 Core, 1 GB RAM, 6 GB Disk space, 32768 File descriptors](#)
- 엔진을 배포 할 각 VM (Datacollector, Transformer) 에 Docker 설치

## STEP 1. Environments 생성

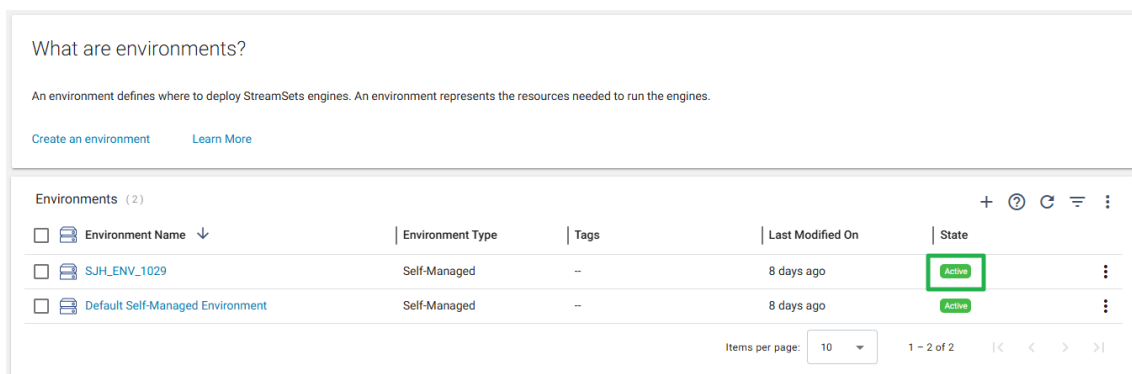
- Environments 는 StreamSets 엔진 (Data Collector, Transformer) 의 배포 위치를 정의
- Environment Type 은 Self-managed, AWS, GCP, Azure, Kubernetes 로 선택 가능



IBM StreamSets 플랫폼 메인 화면에서 Environments 탭 클릭 > Create an environment 클릭



자체 VM에 엔진을 배포할 경우 Self-Managed 를 클릭하고 Save & Exit



Activate 된 Environment State 확인

## STEP 2. Data Collector 생성 및 배포

- Data Collector 인스턴스 그룹 생성

[참고] Docker 설치 스크립트 - 각 VM 에 사전 설치

```
$ yum remove -y docker \
docker-client \
docker-client-latest \
docker-common \
docker-latest \
docker-latest-logrotate \
docker-logrotate \
docker-engine

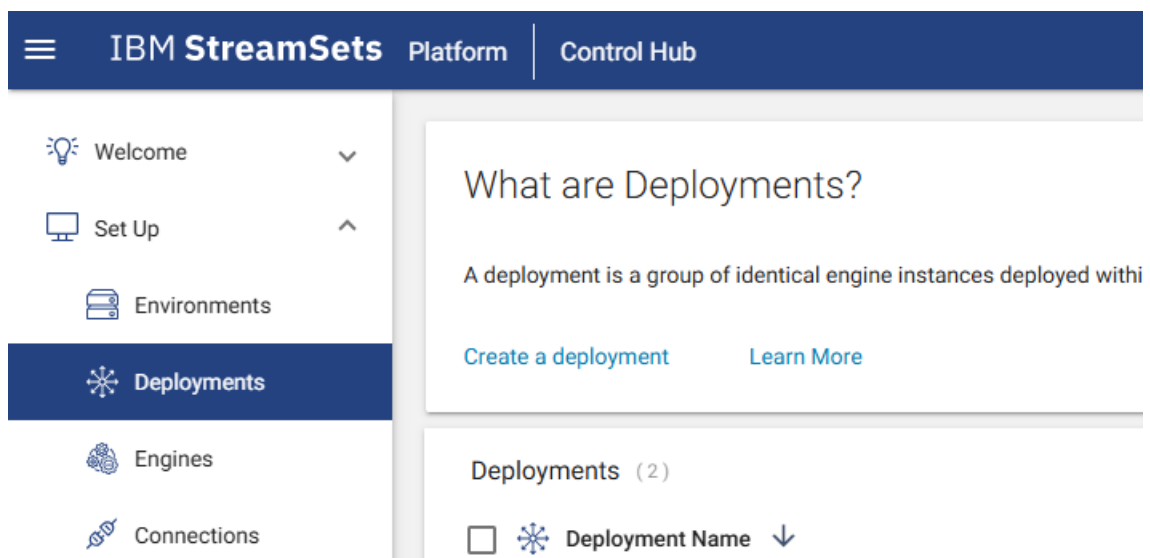
$ yum install -y yum-utils

$ yum-config-manager \
--add-repo \
https://download.docker.com/linux/centos/docker-ce.repo

$ yum update -y && yum install -y docker-ce docker-ce-cli containerd.io

$ systemctl start docker && systemctl enable --now docker

## Docker version 체크
$ docker version
```



Deployments 탭 > Create a deployment 클릭

## New Deployment

1 Define Deployment

Define deployment details. Once saved, you cannot change the deployment type, the engine version, or the environment that the deployment belongs to. [Learn more](#)

Deployment:

Name:

Deployment Type:

Environment:

Engine Type:   
☒ **Data Collector** - Runs data ingestion pipelines that perform record-based data transformations in streaming, CDC, or batch modes  
☐ **Transformer** - Runs data processing pipelines on Spark that perform set-based transformations such as joins, aggregates, and sorts on the entire data set

Engine Version:

Deployment Tags:

### Define the Deployment

Define the deployment essentials, including the deployment name and type, the environment that the deployment belongs to, and the engine type and version to deploy.

Once saved, you cannot change the deployment type, the engine version, or the environment.

1. Configure the following properties:

Define Deployment Property	Description
Deployment Name	Name of the deployment. Use a brief name that informs your team of the deployment use case.
Deployment Type	Select one of the following types: <a href="#">Self-managed</a> <a href="#">Amazon EC2</a> <a href="#">Presto/Presto Engine (PEP)</a>

Deployment 명 입력, Data Collector 선택 후 Save & Next

## New Deployment

1 Define Deployment

2 Configure Engine

Define the configuration of the engine to deploy. You can use the defaults to get started. [Learn more](#)

Stage Libraries:

Advanced Configuration: [Click here to configure](#)

External Resource Source:

Engine Labels:

Max CPU Load (%):

Max Memory (%):

Max Running Pipeline Count:

### Configure the Engine

Define the configuration of the engine to deploy. You can use the defaults to get started.

1. Configure the following properties:

Engine Property	Description
Stage Libraries	Stage libraries to install on the engine. The available stage libraries depend on the selected engine type and version. Not applicable for a Transformer for Snowflake deployment.
	Access to advanced configuration properties to further customize the engine. As you get started with StreamSets, the default values should work in most cases. The available properties depend on the selected engine type.

엔진에 설치할 라이브러리를 선택하기 위해 Stage Libraries 를 클릭

### Select Stage Libraries

Select the stage and external resource to add or remove from the deployment. Stage libraries not showing up in your search? Check the job version.

Available Stages: 0

Selected Stages: 1

### Select Stage Libraries

Select the stage and external resource to add or remove from the deployment. Stage libraries not showing up in your search? Check the job version.

Available Stages: 0

Selected Stages: 4

각 Stage 에서 사용할 라이브러리를 설치

편의상 JDBC, Snowflake 관련 라이브러리를 모두 선택



```
root@localhost:/
[root@localhost /]#
[root@localhost /]#
[root@localhost /]# docker run -p 19630:19630 -d -e STREAMSETS_DEPLOYMENT_SCH_URL=https://eu01.hub.st
reamsets.com -e STREAMSETS_DEPLOYMENT_ID=31dc4316-blee-4d89-8ece-d27aac628c41:b28b3dc6-9595-11ef-8679
-1574607af862 -e STREAMSETS_DEPLOYMENT_TOKEN=eyJ0eXAiOiJKV1QiLCJhbGciOiJub251In0.eyJzIjoiMmQzODBmMDIx
ODM2N2Q0NjcwZjA2NGM4MjFkNzFjM2U3MTZkYWZhYzBmMDYzN2UzMjE2YzdhY2JlNzU0NGVhbnV5ODY4Y2MyYjYkODMzQ1ZDRjOTkxY
WRhY2E3MjQzZDgyZTVlZGY5MDQ0ZWVlMTcxNWRkMmVhOTY3NGY2N2N1NWYiLCJ2IjoxLCJpc3MiOiJldTAXIiwianRpIjoiZDM2Nj
U4ODYtN2ZlMy00MDgxLTliODgtMzY2NDBlZTY2ZjZkdIiwib3VlIjoiImIyOGIzZGM2LTk1OTU0MTF1Zi04Njc5LTB1NzQ2MDdhZjg2MiJ
9. streamsets/transformer:scala-2.12_5.8.0
```

```
root@localhost:/
[root@localhost /]#
[root@localhost /]#
[root@localhost /]# docker ps -al
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS        NAMES
ae545e6a40b1   streamsets/datacollector:JDK17_5.12.0 "/docker-entrypoint..." 8 days ago    Up 8 days    18630/tcp    vibrant_moore
```

실행 후 docker ps -al 명령으로 정상적으로 컨테이너가 실행 중인지 확인

The screenshot shows the IBM StreamSets Platform Control Hub interface. The left sidebar contains navigation links: Welcome, Set Up, Environments, Deployments (selected), Engines, Connections, Build, Run, Monitor, and Manage. The main content area displays 'Deployments > SJH\_DC\_1029\_2'. The 'Deployment Details' section shows the following information:

- Name:** SJH\_DC\_1029\_2
- Engine for Deployment:** Data Collector 5.12.0
- State:** Active
- Last Modified By:** suh1027@gmail.com
- Deployment Type:** Self-Managed
- Environment:** SJH\_ENV\_1029
- Install Type:** Tarball
- Engine Labels:** sjh\_dc\_1029\_2
- Deployment ID:** 7c4d8501-a00a-4963-862a-0f9ef78eb4fc:b28b3dc6-9595-11ef-8679-1574607af862

Below the deployment details, the 'Engines in Deployment' section shows a table with the following data:

Hostname	Running Pipelines	CPU Load	Memory Used	Last Reported Time
ae545e6a40b1:18630	0	0.29 %	24.66 %	a few seconds ago

설치 및 실행이 완료 되면, Engines in Deloyment 에서 실행 중인 컨테이너가 확인 가능하다.

Container ID : 포트 로 구분 가능

### STEP 3. Transformer 생성 및 배포

- Transformer 인스턴스 그룹 생성

**Edit Deployment**

Define deployment details. Once saved, you cannot change the deployment type, the engine version, or the environment that the deployment belongs to.

**Define Deployment**

Deployment Name:

Deployment Type:

Environment:

Engine Type: ☒ Data Collector - Runs data ingestion pipelines that perform record-based data transformations in streaming, CDC, or batch modes. ☐ Transformer - Runs data processing pipelines on Spark that perform set-based transformations such as joins, aggregates, and sorts on the entire data set.

Engine Version:

Deployment Tags:

**Define the Deployment**

Define the deployment essentials, including the deployment name and type, the environment that the deployment belongs to, and the engine type and version to deploy.

Once saved, you cannot change the deployment type, the engine version, or the environment.

1. Configure the following properties:

Define Deployment Property	Description
Deployment Name	Name of the deployment. Use a brief name that informs your team of the deployment use case.
	Select one of the following types: <a href="#">Self-managed</a>

[Save](#)

Data Collector 와 설치 과정은 동일하며, Deployment 명, Environment, Engine Type 선택 후 Next

**Edit Deployment**

Define the configuration of the engine to deploy. You can use the defaults to get started.

**Stage Libraries:**

Advanced Configuration: [Click here to configure](#)

External Resource Source:

Engine Labels:  [Add New...](#)

Max CPU Load (%):

Max Memory (%):

Max Running Pipeline Count:

**Configure the Engine**

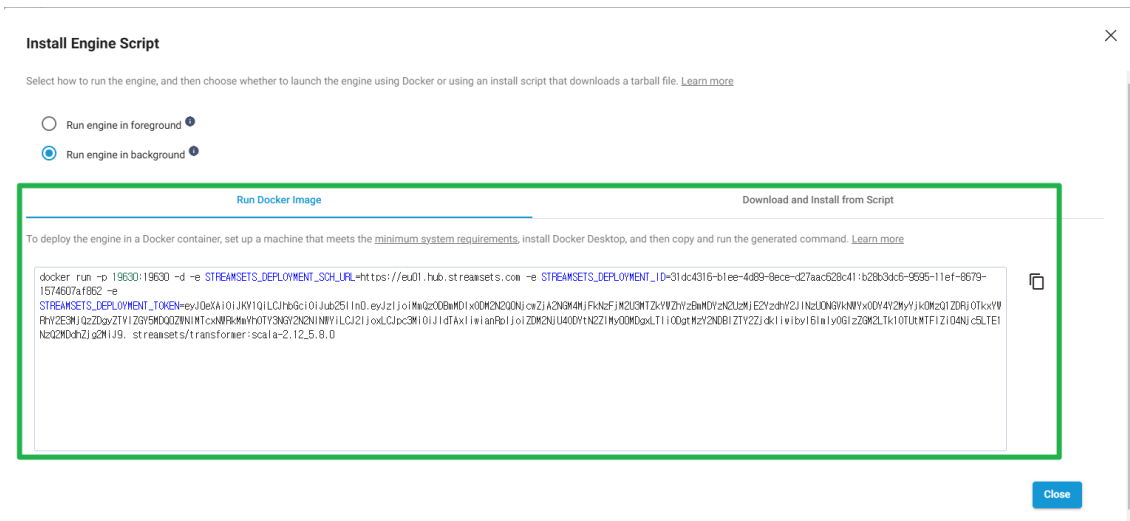
Define the configuration of the engine to deploy. You can use the defaults to get started.

1. Configure the following properties:

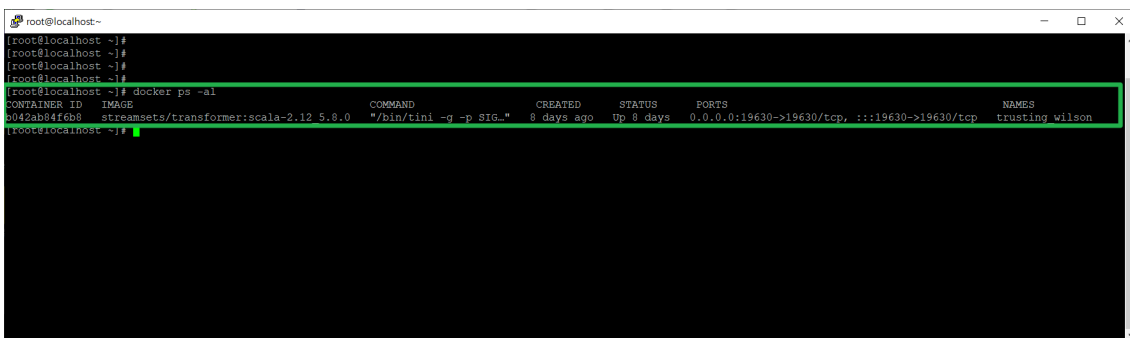
Engine Property	Description
<a href="#">Stage Libraries</a>	Stage libraries to install on the engine. The available stage libraries depend on the selected engine type and version. Not applicable for a <a href="#">Transformer for Snowflake deployment</a> .
	Access to advanced configuration properties to further customize the engine. As you get started with StreamSets, the default values should work in most cases. The available properties depend on the selected engine type.

[Save](#)

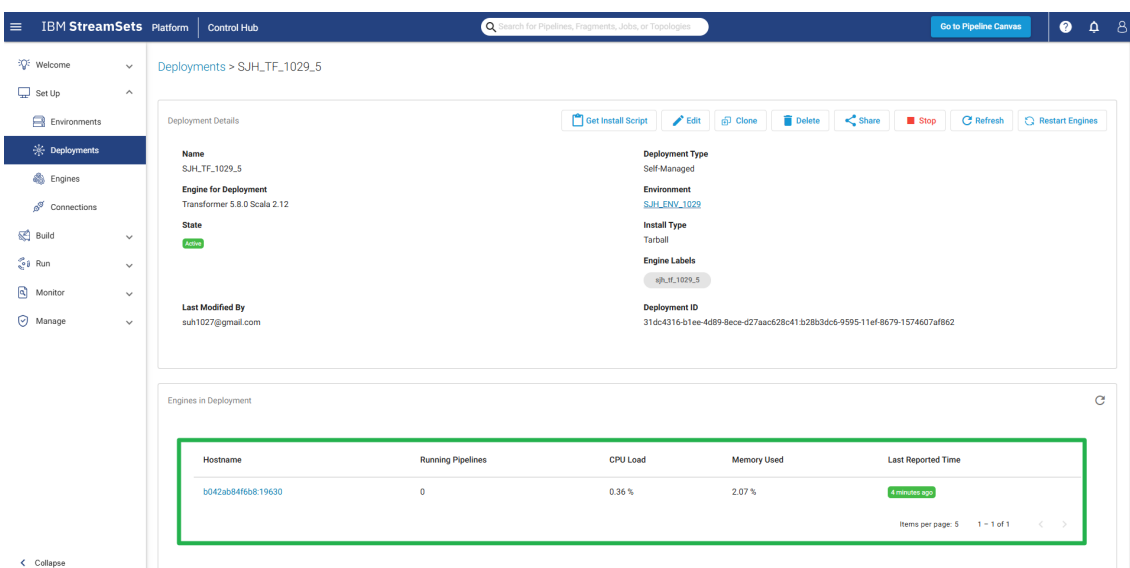
Stage Libraries 클릭 후 JDBC, Snowflake 관련 라이브러리 모두 선택



백그라운드 실행 및 Docker Image 로 선택하여 스크립트 복사 후 VM 에서 실행



docker ps -al 명령으로 컨테이너 실행 확인

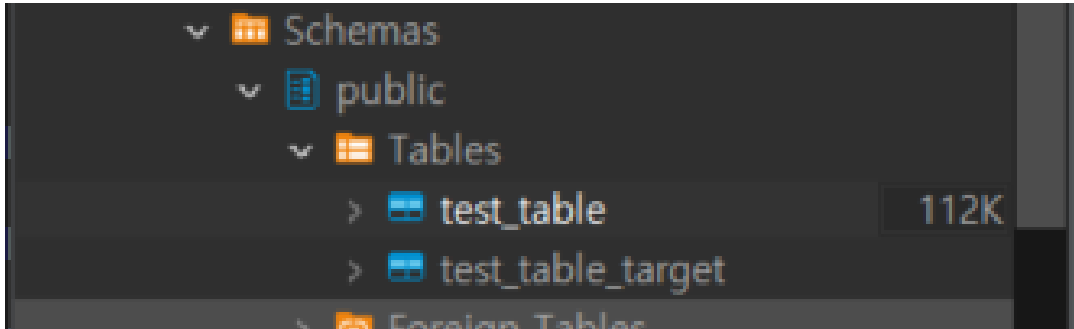


엔진 실행 확인

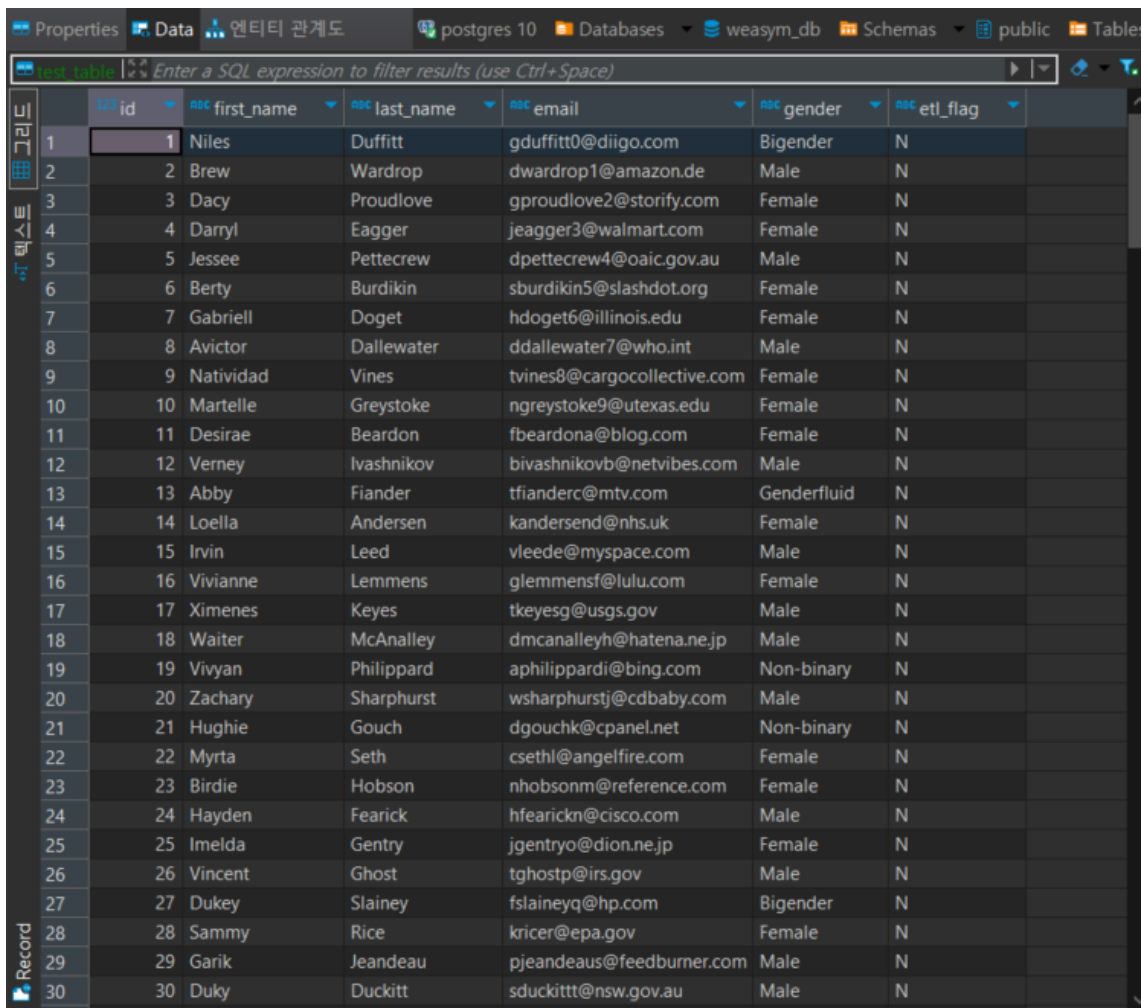


#### STEP 4. DB to DB Pipeline 생성

- 더미 데이터를 사용한 Pipeline 생성 Tutorial
- [파일 다운로드 링크](#) - MOCK\_DATA.csv 다운로드

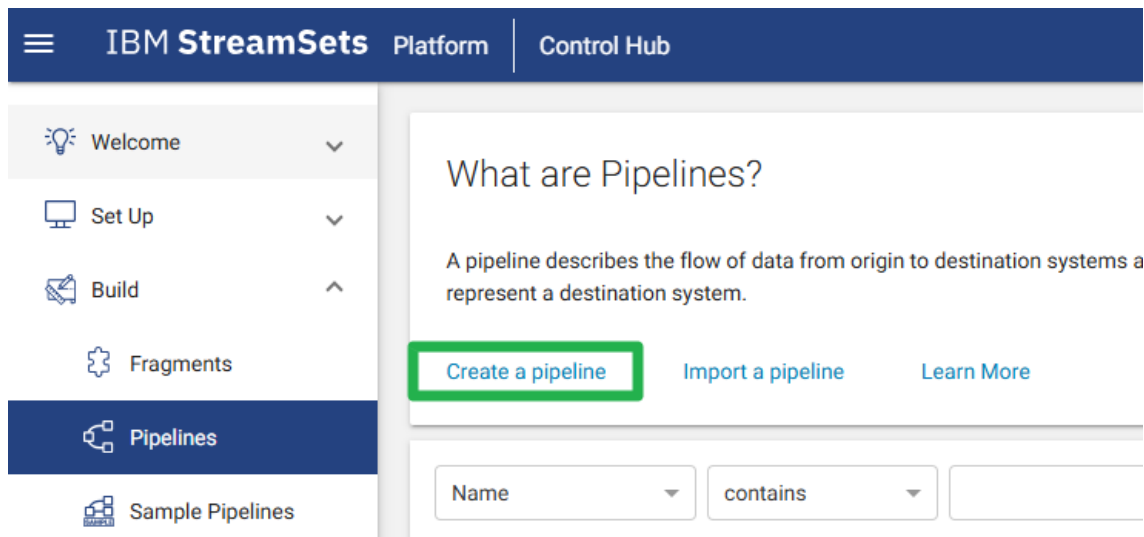


미리 생성해둔 DB에 TEST\_TABLE, TEST\_TABLE\_TARGET 의 명칭으로 두개의 테이블을 생성.

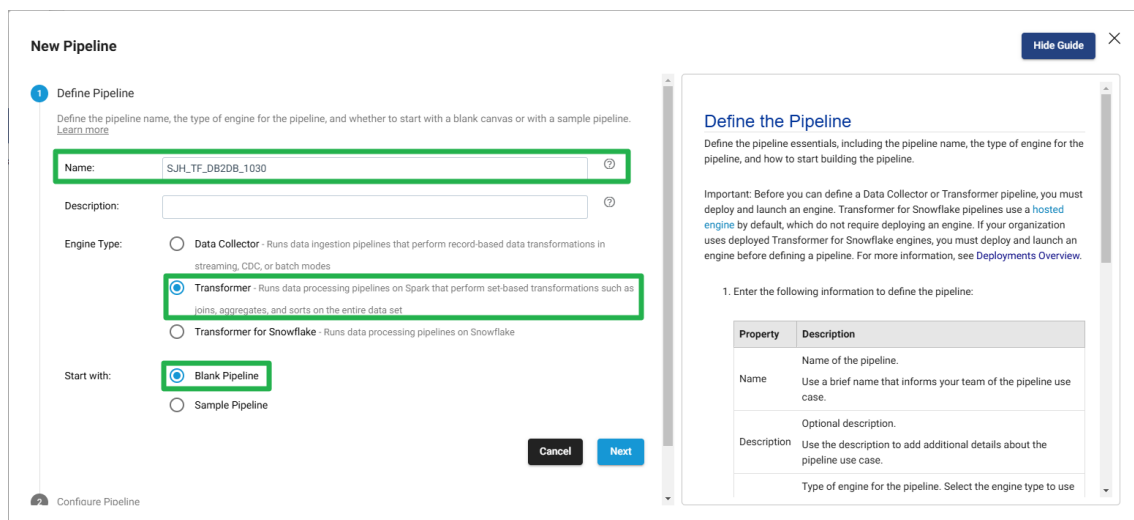


	id	first_name	last_name	email	gender	etl_flag
1	1	Niles	Duffitt	gduffitt0@diigo.com	Bigender	N
2	2	Brew	Wardrop	dwardrop1@amazon.de	Male	N
3	3	Dacy	Proudlove	gproudlove2@storify.com	Female	N
4	4	Darryl	Eagger	jeagger3@walmart.com	Female	N
5	5	Jessee	Pettecrew	dpettecrew4@oaic.gov.au	Male	N
6	6	Berty	Burdikin	sburdikin5@slashdot.org	Female	N
7	7	Gabriell	Doget	hdoget6@illinois.edu	Female	N
8	8	Avictor	Dallewater	ddallewater7@who.int	Male	N
9	9	Natividad	Vines	tvines8@cargocollective.com	Female	N
10	10	Martelle	Greystoke	ngreystoke9@utexas.edu	Female	N
11	11	Desirae	Beardon	fbeardona@blog.com	Female	N
12	12	Verney	Ivashnikov	bivashnikovb@netvibes.com	Male	N
13	13	Abby	Fiander	tfianderc@mtv.com	Genderfluid	N
14	14	Loella	Andersen	kandersend@nhs.uk	Female	N
15	15	Irvin	Leed	vleede@myspace.com	Male	N
16	16	Vivianne	Lemmens	glemmensf@lulu.com	Female	N
17	17	Ximenes	Keyes	tkeyesg@usgs.gov	Male	N
18	18	Waiter	McAnalley	dmcanalleyh@hatena.ne.jp	Male	N
19	19	Vivyan	Philippard	aphilippardi@bing.com	Non-binary	N
20	20	Zachary	Sharphurst	wsharphurstj@cdbaby.com	Male	N
21	21	Hughie	Gouch	dgouchk@cpanel.net	Non-binary	N
22	22	Myrta	Seth	csethl@angelfire.com	Female	N
23	23	Birdie	Hobson	nhobsonm@reference.com	Female	N
24	24	Hayden	Fearick	hfearickn@cisco.com	Male	N
25	25	Imelda	Gentry	jgentryo@dion.ne.jp	Female	N
26	26	Vincent	Ghost	tghostp@irs.gov	Male	N
27	27	Dukey	Slainey	fslaineyq@hp.com	Bigender	N
28	28	Sammy	Rice	kricer@epa.gov	Female	N
29	29	Garik	Jeandeau	pjeandeaus@feedburner.com	Male	N
30	30	Duky	Duckitt	sduckittt@nsw.gov.au	Male	N

TEST\_TABLE 에는 별도의 DB Tool 을 사용, MOCK\_DATA 파일 내 데이터를 가져와 데이터를 입력한다.  
편의상 에러 방지를 위해 컬럼의 크기는 최대로 늘려서 세팅



Build > Pipelines 탭에서 Create a pipeline 을 클릭하여 파이프라인 생성



Transformer 파이프라인에 있는 Stage를 사용하여 테스트를 진행하기 위해 Transformer 엔진 선택  
Pipeline Engine 은 3개의 타입이 있으며 각 타입은 다음과 같은 특징을 갖는다.

Data Collector	스트리밍, CDC 또는 Batch 처리 모드에서 레코드 기반 데이터 파이프라인 수행
Transformer	Apache Spark에서 실행되는 데이터 처리 파이프라인을 실행, Transformer 파이프라인은 클러스터에 배포된 Spark에서 실행되므로 전체 데이터 세트에서 조인, 집계 및 정렬과 같은 세트 기반 변환을 수행
Transformer for Snowflake	파이프라인 구성에 따라 SQL 쿼리를 생성하고 쿼리를 Snowflake에 전달하여 실행. Snowflake 파이프라인은 Snowpark DataFrame 기반 처리를 사용하여 Snowflake 테이블에서 읽고 쓰기를 수행

## 생성한 Pipeline Canvas 화면

The image shows two screenshots of the IBM StreamSets interface. The top screenshot displays the Pipeline Canvas, which is a visual representation of a data pipeline. It consists of three stages: 'PostgreSQL JDBC Table 1', 'Field Replacer 1', and 'JDBC Table 2', connected in a linear sequence. The interface includes a left-hand navigation menu with options like 'Welcome', 'Set Up', 'Build', 'Fragments', 'Pipelines', 'Sample Pipelines', 'Run', 'Monitor', and 'Manage'. The top bar shows the pipeline name 'SJH\_TF\_DB2DB\_1030' and its status 'v1-DRAFT'. The bottom screenshot shows the 'Configuration' tab for the same pipeline. It includes fields for 'Name' (SJH\_TF\_DB2DB\_1030), 'Description', 'Labels', 'Execution Mode' (Batch), and 'Retry Pipeline on Error' (checked). The 'Retry Attempts' are set to 3. There are also checkboxes for 'Enable Ludicrous Mode' and 'Advanced Error Handling'.

## Pipeline 전체 General 설정 예시

Excution Mode 와 Error 시 Retry 여부, 횟수를 설정할 수 있다.

PostgreSQL JDBC Table1 스테이지 General 탭 설정 예시

Connection 탭 예시, + 를 눌러 커넥션을 생성하거나, 생성 된 커넥션을 수정할 수 있다.

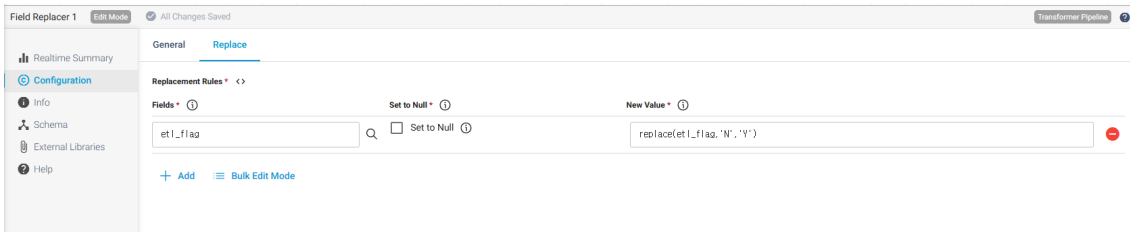
Connection 탭에서 Connection 컴포넌트를 생성할 수 있다.

Connection 명과 Description, Data Collector 엔진을 선택

미리 생성해 둔 DB의 JDBC Connection String 을 입력하고, Credential 을 입력 후  
Connection 테스트 후 Save

Table 탭 설정 예시

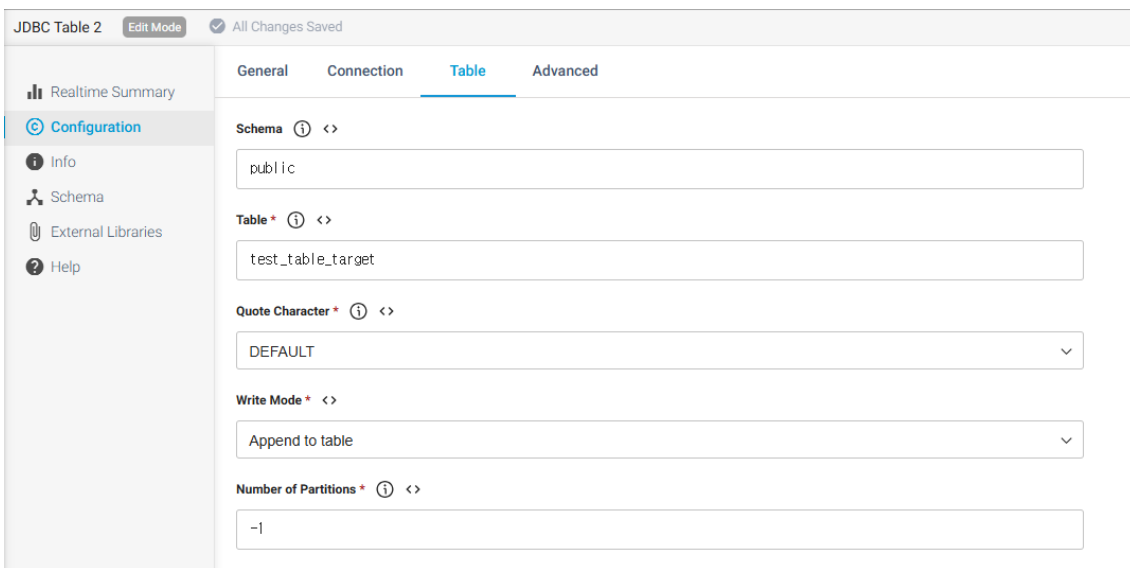
Offset Column 은 파이프라인 실행 시 처리 된 행을 구분하기 위한 컬럼으로, Primary Key 값을 주로 사용한다.



Field Replacer 1 Stage > Replace 탭 설정 예시

ETL\_FLAG 필드의 값을 N → Y 로 Replace 하는 함수를 Value 값에 입력한다.

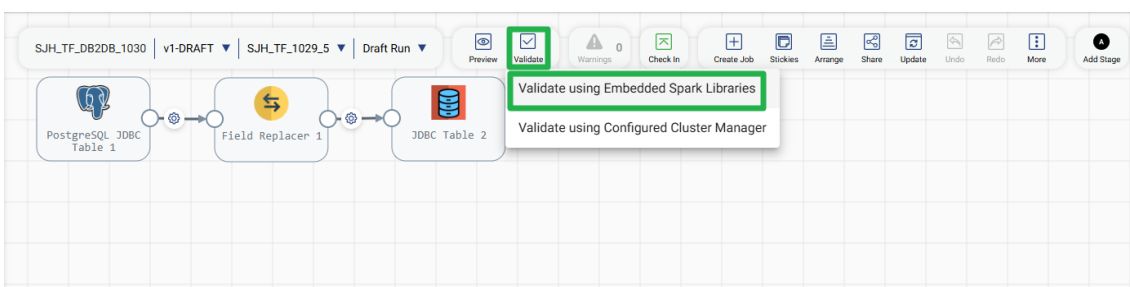
`replace(etl_flag, 'N', 'Y')`



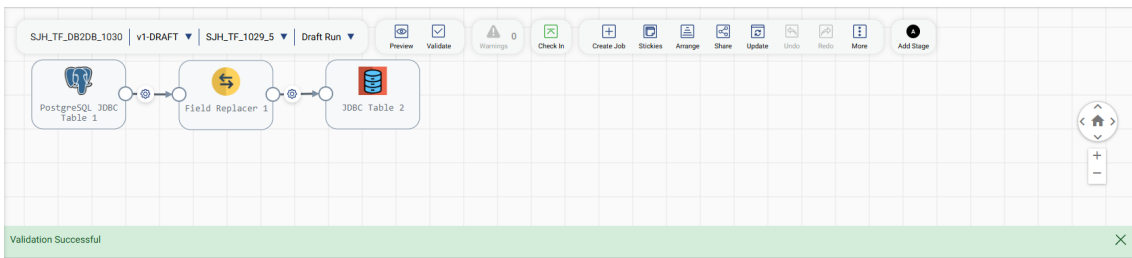
JDBC Table 2 스테이지의 커넥션의 생성 방법은 PostgreSQL JDBC Table1 스테이지와 동일하다.

Table 탭의 설정은 Target DB의 스키마와 테이블 지정.

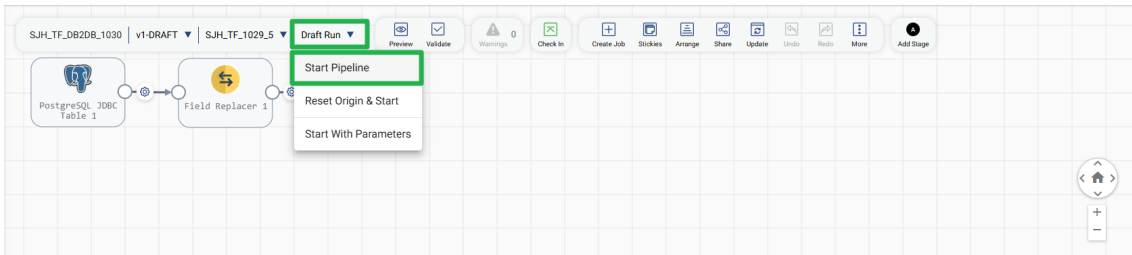
## STEP 5. 생성된 파이프라인 검증 및 실행



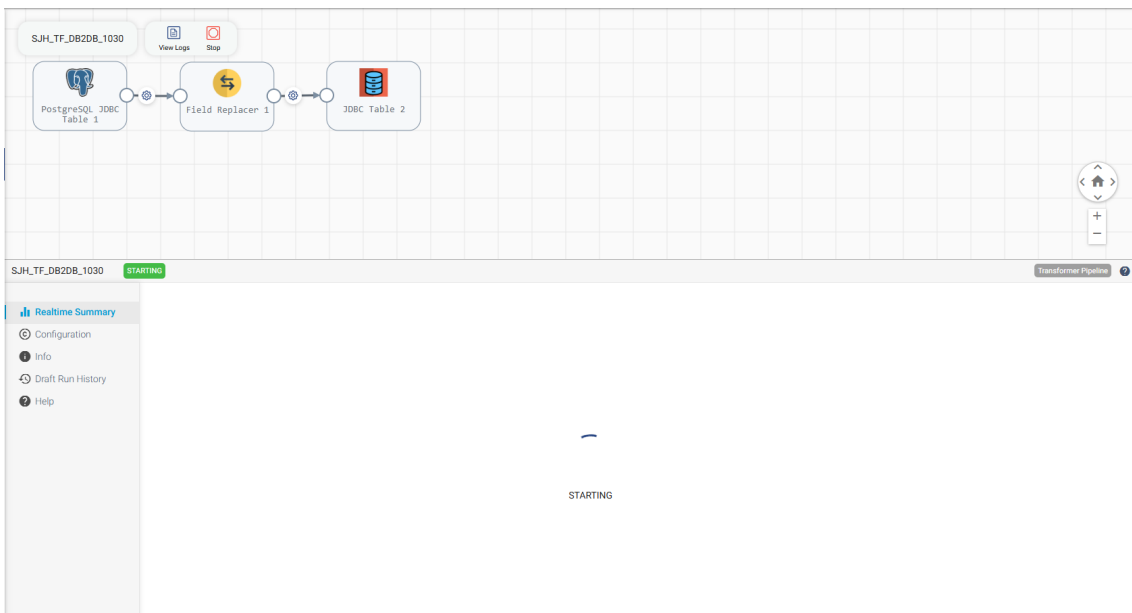
Transformer에 내장된 Spark Libraries를 사용하여 생성한 Pipeline을 Validation한다.

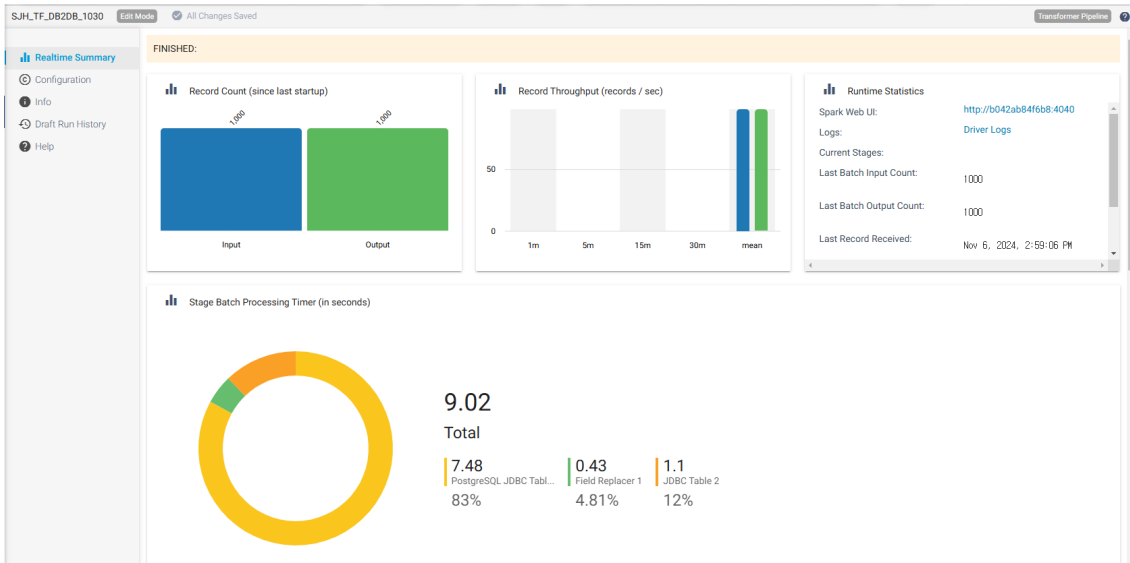


## Validation 성공 예시



Draft Run 을 클릭하여 Start Pipeline, 또는 Origin Reset 후 Start 할 수 있다.



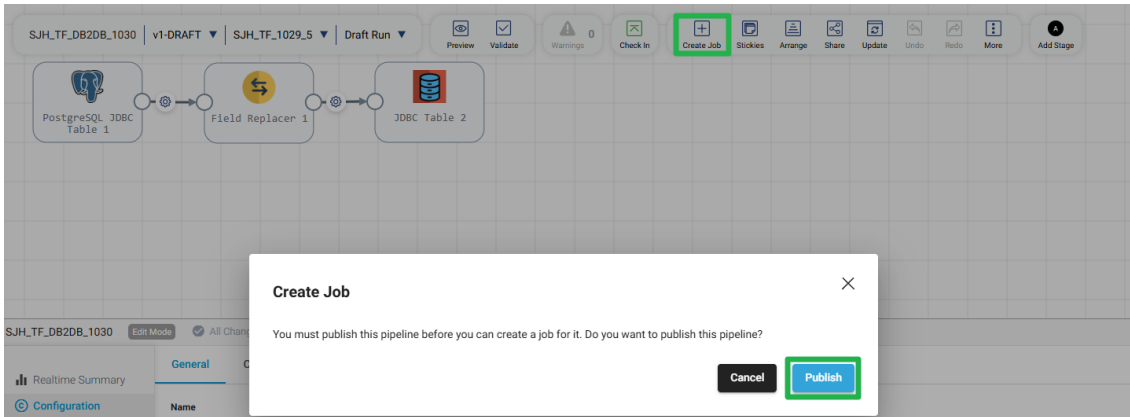


처리 중 각 스테이지를 클릭하여 처리 현황을 대시보드로 실시간 확인할 수 있다.

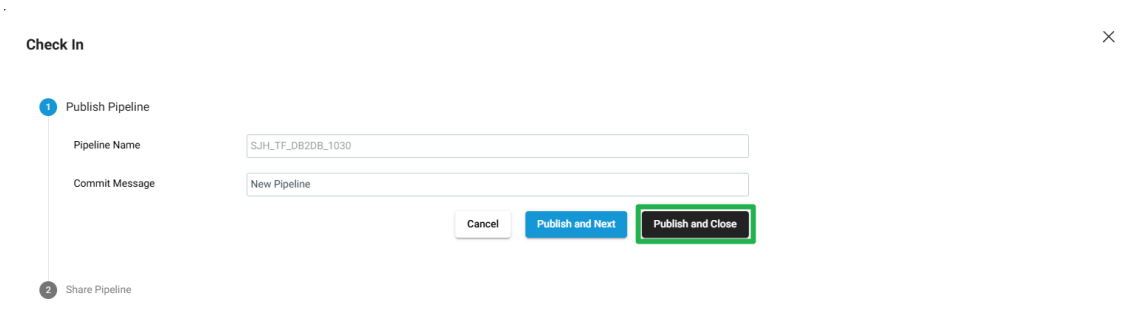
	id	first_name	last_name	email	gender	etl_flag
1	696	Alis	Otto	dottojb@ning.com	Female	Y
2	602	Tyrone	Couzens	rcouzensgp@angelfire.com	Male	Y
3	603	Rudolfo	Zanini	hzaninigq@is.gd	Male	Y
4	604	Augustine	Vigus	nvigusgr@craigslist.org	Male	Y
5	605	Nickola	Wintersgill	iwintersgillgs@histats.com	Male	Y
6	606	Alfi	Le Marquis	glemarquisgt@google.com.hk	Female	Y
7	607	Massimo	Copsey	bcopseygu@blogs.com	Male	Y
8	608	Teri	Hearst	ahearstgv@opensource.org	Female	Y
9	609	Shalom	Le Port	kleportgw@slideshare.net	Agender	Y
10	610	Joelly	Di Napoli	ddinapoligx@ox.ac.uk	Female	Y
11	611	Kacie	Wey	pweygy@vk.com	Female	Y
12	612	Alexandros	Rampage	prampagegz@redcross.org	Male	Y
13	613	Joel	Toon	stoonh0@redcross.org	Male	Y
14	614	Angie	Pull	wpullh1@discovery.com	Female	Y
15	615	Joline	Kyttor	rkyttorh2@booking.com	Female	Y
16	616	Pippa	Gronno	hgronnoh3@pbs.org	Female	Y
17	617	Dalston	Eaglesham	meagleshamh4@businessinsider.com	Male	Y
18	618	Standford	Fyldes	kfyldesh5@creativecommons.org	Male	Y
19	619	Augustine	Queripel	fqueripelh6@narod.ru	Non-binary	Y
20	620	Virgie	Kemmet	rkemmeth7@joomla.org	Male	Y
21	621	Grete	Vanyatin	pvanatinh8@wired.com	Female	Y
22	622	Kirsteni	Melrose	bmelroseh9@prweb.com	Female	Y
23	623	Wynnie	Portigall	uportigallha@un.org	Female	Y
24	624	Alon	Prozescky	gprozesckymb@xinhuanet.com	Male	Y
25	625	Tuesday	Corkan	kcorkanhc@pinterest.com	Female	Y
26	626	Basile	Sandwich	nsandwichhd@taobao.com	Male	Y
27	627	Rozanna	Riply	lriplyhe@dailyemotion.com	Female	Y
28	628	Hope	Parkins	gparkinshf@chronoengine.com	Female	Y
29	629	Flint	Hasson	phassonhg@scribd.com	Male	Y
30	630	Ardelle	Dunton	tduntonhh@trellian.com	Female	Y

처리 완료 된 TEST\_TABLE\_TARGET 데이터 목록





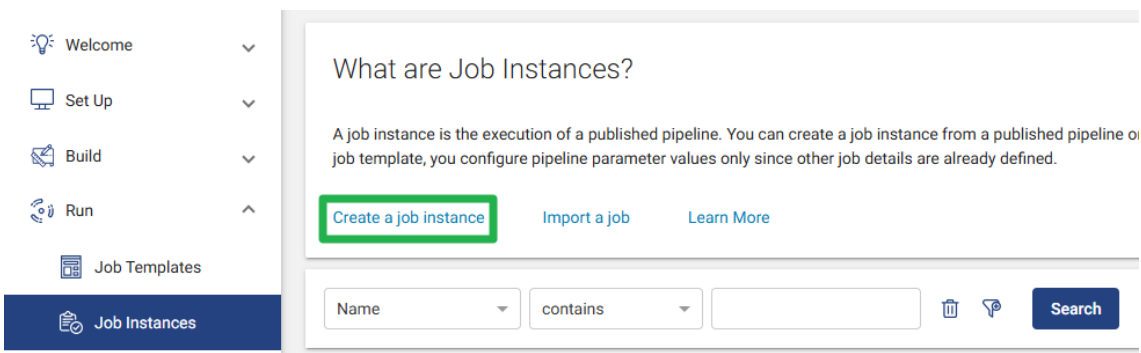
생성 완료 된 Pipeline 는 Publish 해야 Job Instance 생성이 가능하다.



Publish 된 Pipeline 은 Read Only Mode 로 변경.

## STEP 6. Job Instance & Schedule 생성

- Job 을 생성하여 Publish 된 파이프라인 실행 할 수 있다.
- 생성 된 Job 으로 주기적으로 Pipeline 을 실행 시킬 수 있다.



Run > Job Instances 탭 > Create a job instance 를 클릭하여 job을 생성한다.

### Create Job Instances

1 Define Job

Define job details. [Learn more](#)

Name:

SJH\_JOB\_1

?

Description:

?

Job Tags:

Add New...

?

Start with:

☒ Create from pipeline
 ☐ Create from job template

Cancel

Next

2 Select Pipeline

3 Configure Job

4 Review & Start

Hide Guide

×

#### Define the Job

Define the job essentials, including the job name, and optionally a description and tags to identify similar jobs.

Important: Before you can define a job, you must publish a pipeline built in the pipeline canvas. For instructions, see [Publishing a Pipeline](#).

1. Configure the following information to define the job:

Property	Description
Name	Name of the job. Use a brief name that informs your team of the job use case.
Description	Optional description. Use the description to add additional details about the job use case.
Job Tags	Tags that identify similar jobs. Use job tags to easily search and filter jobs. Enter nested tags using the following format: <tag1>/<tag2>/<tag3>

Job 컴포넌트 명, Create from pipeline 선택

### Create Job Instances

1 Define Job

2 Select Pipeline

3 Configure Job

4 Review & Start

Hide Guide

×

#### Select the Pipeline

Select the published pipeline that you want to add to the job and then run.

Note: If you launched the job wizard after selecting a published pipeline in the Pipelines view, the pipeline and pipeline version are selected for you. You can change them as needed.

- To select a published pipeline, click **Click here to select** next to the **Pipeline** property.  
In the **Select a Pipeline** window, select the pipeline to use, and then click **Save** to return to the job wizard.  
By default, the latest published version is selected.
- To select an earlier published version, click **Click here to select** next to the **Pipeline Version** property.  
In the **Select a Pipeline Version** window, select an earlier version, and then click **Save** to return to the job wizard.
- Click one of the following buttons:  
**Back** - Returns to the previous step in the wizard.  
**Next** - Saves the job definition and continues.

게시 된 파이프라인 및 버전을 선택 후 Next

### Create Job Instances

1 Define Job

2 Select Pipeline

3 Configure Job

4 Review & Start

Configure job details to determine how engines run the pipeline. The default values for the advanced options should work in most cases. [Learn more](#)

Deployment: SJH\_TF\_1029\_5 (Self-Managed)

Engine Labels: sjh\_tf\_1029\_5 Add New...

Enable Failover: ☒

[Show Advanced Options](#)

Back Save & Next Save & Exit

Hide Guide

### Configure the Job

Configure the job to determine how engines run the pipeline.

**Note:** When your organization uses the Transformer for Snowflake [engine hosted by StreamSets](#), StreamSets automatically handles how the engine runs pipelines. As such, the job template wizard skips this step for Transformer for Snowflake jobs that run on the hosted engine.

1. Configure the following job properties:

Property	Description
Deployment	Deployment associated with the group of engines that run the pipeline.
Engine Labels	Label or labels that determine the group of engines that run the pipeline. Labels are case sensitive.  By default, the labels assigned to the deployment are automatically selected as the engine labels. In most cases, you can use the default labels so that the pipeline runs on a deployment of identical engine instances.
	Enables Control Hub to restart a pipeline on another

## Save & Exit

### What are Job Instances?

A job instance is the execution of a published pipeline. You can create a job instance from a published pipeline or from a job template. When you create a job instance from a pipeline, you configure all of the job details. When you create a job instance from a job template, you configure pipeline parameter values only since other job details are already defined.

[Create a job instance](#) [Import a job](#) [Learn More](#)

Name contains  Search

Job Instances (1)	Name	Pipeline	Version	Last Modified On	Job Status	Pipeline Status
<input type="checkbox"/>	SJH_JOB_1	SJH_TF_DB2DB_1030	v1	a few seconds ago	ACTIVE	STARTING

Items per page: 50 1 - 1 of 1

실행 중인 Pipeline 상태를 Job instances 탭에서 실시간으로 확인 가능하다.

### What are Scheduled Tasks?

A scheduled task starts, stops, or upgrades jobs on a regular basis. For example, a scheduled task can start a job on a weekly or monthly basis.

[Create a scheduled task](#) [Learn more](#)

### Scheduled Tasks

Name	Cron Expression	Time Zone	Status	Next Execution Time
No Scheduler Tasks found				

Scheduled Tasks 탭 > Create a scheduled task 를 클릭하여 Job 의 주기적 실행이 가능하도록 설정 가능하다.

### Create Scheduled Task

Name  
SJH\_JOB\_1

Description

Owner  
b47a7446-1ee6-11ef-a44b-b19fd8618c18@b28b3dc6-9595-11ef-8679-1574607af862

Action  
START

Cron Expression ?

Minutes Hourly **Daily** Weekly Monthly Yearly Advanced

Every 1 day(s) at 0 0

2 / 2

Cancel Back Save

Cron 의 설정과 유사하며, 생성 된 Job 을 선택 하여 설정 가능.

### \* Data Collector, Transformer 엔진 내 (Docker) 실행 이력 로그 확인 방법

```

root@localhost:~#
root@localhost ~#
root@localhost ~# docker ps -al
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
b042ab84f6b8   streamsets/transformer:scala-2.12_5.8.0  "/bin/tini -g -p SIG..."  8 days ago    Up 8 days    0.0.0.0:19630->19630/tcp, :::19630->19630/tcp  trusting_wilson
root@localhost ~#
root@localhost ~# docker exec -it b042ab84f6b8 /bin/bash
transformer@b042ab84f6b8:/$
transformer@b042ab84f6b8:/$

```

## 실행중인 Engine 의 Container ID 확인

\$ docker ps -al

CONTAINER ID	IMAGE	COMMAND
b042ab84f6b8	streamsets/transformer:scala-2.12_5.8.0	"/bin/tini -g -p SIG

## Docker 명령으로 컨테이너 접속

\$ docker exec -it b042ab84f6b8 /bin/bash

```

root@localhost:~
transformer@b042ab84f6b8:/data/transformer$
transformer@b042ab84f6b8:/data/transformer$
transformer@b042ab84f6b8:/data/transformer$
transformer@b042ab84f6b8:/data/transformer$ pwd
/data/transformer
transformer@b042ab84f6b8:/data/transformer$
transformer@b042ab84f6b8:/data/transformer$ ls -al
total 300
drwxr-xr-x. 1 transformer transformer 219 Nov  6 06:43 .
drwxr-xr-x. 1 root          root      25 Jun 25 09:07 ..
drwxr-xr-x. 2 transformer transformer 27 Oct 29 04:19 blobstore
-rw-r--r--. 1 transformer transformer 195 Oct 29 04:20 control-hub-pushed.properties
-rw-----. 1 transformer transformer 1526 Nov  6 06:43 meteringSnapshot.json
-rw-r--r--. 1 transformer transformer  65 Nov  5 21:05 opt-stats.json
drwxr-xr-x. 4 transformer transformer 4096 Nov  6 06:08 pipelines
drwxr-xr-x. 5 transformer transformer 249 Nov  6 06:08 runHistory
drwxr-xr-x. 4 transformer transformer 4096 Nov  6 06:08 runInfo
-rw-r--r--. 1 transformer transformer  36 Oct 29 04:19 sdc.id
-rw-----. 1 transformer transformer 275744 Nov  6 06:43 stats.json
-rw-----. 1 transformer transformer  1528 Nov  6 06:43 statsSnapshot.json
transformer@b042ab84f6b8:/data/transformer$

```

## Transformer 엔진 내부 확인

```

$ cd /data/transformer
$ ls -al
drwxr-xr-x. 1 transformer transformer 219 Nov  6 06:43 .
drwxr-xr-x. 1 root          root      25 Jun 25 09:07 ..
drwxr-xr-x. 2 transformer transformer 27 Oct 29 04:19 blobstore
-rw-r--r--. 1 transformer transformer 195 Oct 29 04:20 control-hub-pushed.
-rw-----. 1 transformer transformer 1526 Nov  6 06:43 meteringSnapshot.js
-rw-r--r--. 1 transformer transformer  65 Nov  5 21:05 opt-stats.json
drwxr-xr-x. 4 transformer transformer 4096 Nov  6 06:08 pipelines
drwxr-xr-x. 5 transformer transformer 249 Nov  6 06:08 runHistory
drwxr-xr-x. 4 transformer transformer 4096 Nov  6 06:08 runInfo
-rw-r--r--. 1 transformer transformer  36 Oct 29 04:19 sdc.id
-rw-----. 1 transformer transformer 275744 Nov  6 06:43 stats.json
-rw-----. 1 transformer transformer  1528 Nov  6 06:43 statsSnapshot.json

```

## runHistory 폴더 내 에서 실행 이력 확인 가능

## 하위 폴더를 Container 안에서 밖으로 Copy 후 파일 확인 (docker cp 명령 사용)

```

$ exit
$ docker cp trusting_wilson:/data/transformer /back
Successfully copied 5.59MB to /back

```