
metatron-doc-user Documentation

출시 버전 0.4.3

metatron team

2020년 05월 11일

I Metatron Discovery	1
1 디스커버리 퀵가이드	3
2 Metatron Discovery 소개	29
3 데이터 관리	67
4 워크스페이스	105
5 워크북	119
6 노트북	197
7 워크벤치	209
8 데이터 프리퍼레이션	223
9 계정 관리	331
10 데이터 탐색	343
11 엔진 모니터링	355
II EX-pack for Workflow Integrator	365
12 Integrator 확장팩 소개	367
13 워크플로우 리스트	369

14 워크플로우 에디터	371
15 모니터링	381
16 Use Case	383
III EX-pack for Anomaly Detection	385
17 Metatron Anomaly 소개	387
18 통계	391
19 알람 내역 열람하기	393
20 알람 룰	401
21 알고리즘	425
22 대시보드	429
23 검색	439

Part I

Metatron Discovery

디스커버리 퀵가이드

Metatron Discovery는 대용량 데이터를 동일한 환경에서 적재, 전처리, 분석까지 빠르게 처리할 수 있는 툴입니다. 데이터 분석을 위한 기술적인 지식이 없는 비즈니스 현업 사용자도 Metatron Discovery를 이용하면 직접 데이터를 다뤄보고 빠르게 시각화해 보면서 인사이트를 얻을 수 있습니다.

Metatron Discovery로 즉시 데이터 분석을 시작할 수 있는 방법은 두 가지가 있습니다.

- **방법 1:** Metatron Discovery 데모 사이트를 실행하세요. ID와 password는 <metatron'이라고 입력하면 됩니다.
- **방법 2:** 싱글 모드용 Metatron Discovery를 로컬 PC에 다운로드 받으세요. 다운로드에는 세 가지 방법으로 제공됩니다.
 - **Custom install:** Github 레파지토리로부터 소스코드를 다운받거나 빌드 파일을 직접 실행하세요.
 - **Virtual machine:** 가상 머신 이미지로 바로 실행하세요. Windows OS에서도 사용할 수 있습니다.
 - **Docker:** Docker 이미지로 빠르게 설치하고 즉시 실행할 수 있습니다.

아래와 같은 화면을 보고 계신가요? 축하합니다! 이제부터 메타트론 디스커버리와 함께 쉽고 빠르게 데이터 분석을 시작해보세요.



빠른 시작을 위해서 아래의 세 단계의 스텝으로 튜토리얼을 진행해보세요.

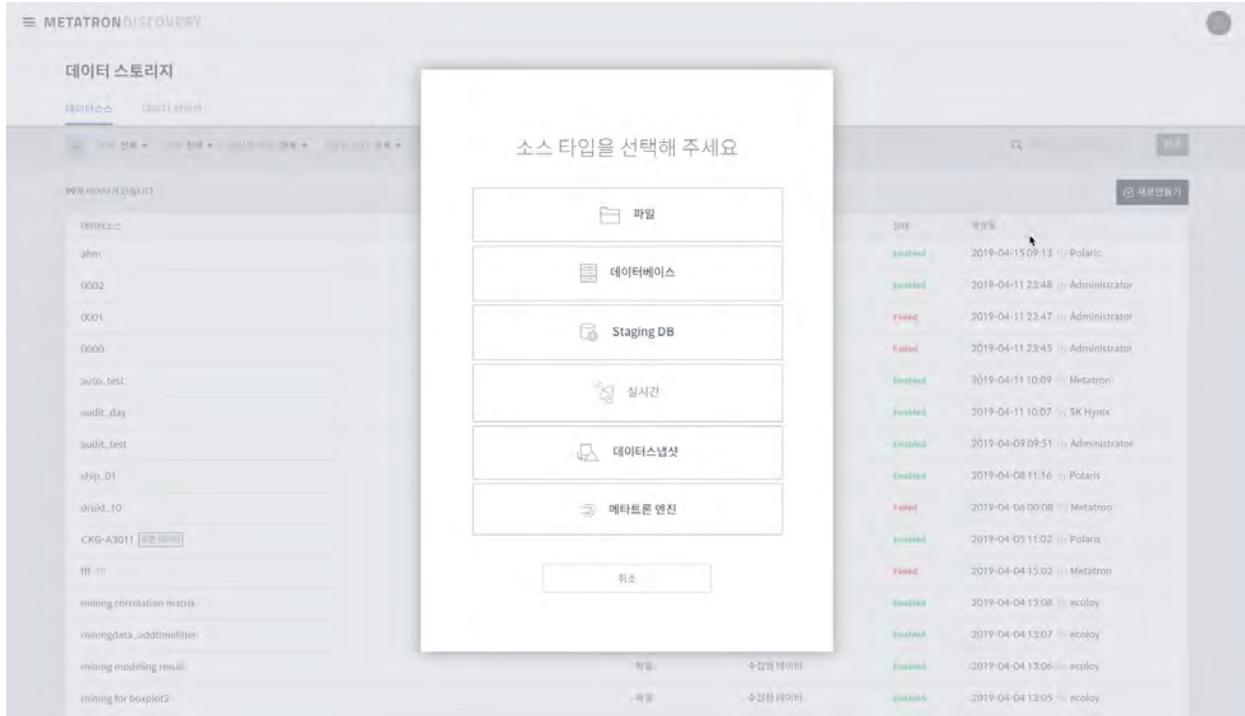
1.1 Step1. 데이터 소스 만들기

데이터 분석을 하기 위해 가장 먼저 해야할 일은 데이터를 시스템에 적재하는 것 입니다. Metatron Discovery는 다양한 원천 데이터를 쉽게 적재할 수 있도록 지원합니다.

본 튜토리얼 예제에서는 로컬 파일에 있는 데이터를 적재하는 방법을 소개합니다. 먼저 데이터를 준비하세요. 흔히 쓰이는 엑셀 파일 (.xls, .xlsx) 또는 .csv 형식의 파일이면 충분합니다. 여기서는 판매 현황 데이터를 활용합니다. 아래 링크에서 다운로드 받으세요.

sample data (.csv)

데이터 소스는 **Management > 데이터 스토리지 > 데이터 소스**에서 조회하고 적재할 수 있습니다. 새로운 데이터 소스를 만들기 위해서 데이터 소스 리스트 우측 상단의 **new** 버튼을 클릭합니다.

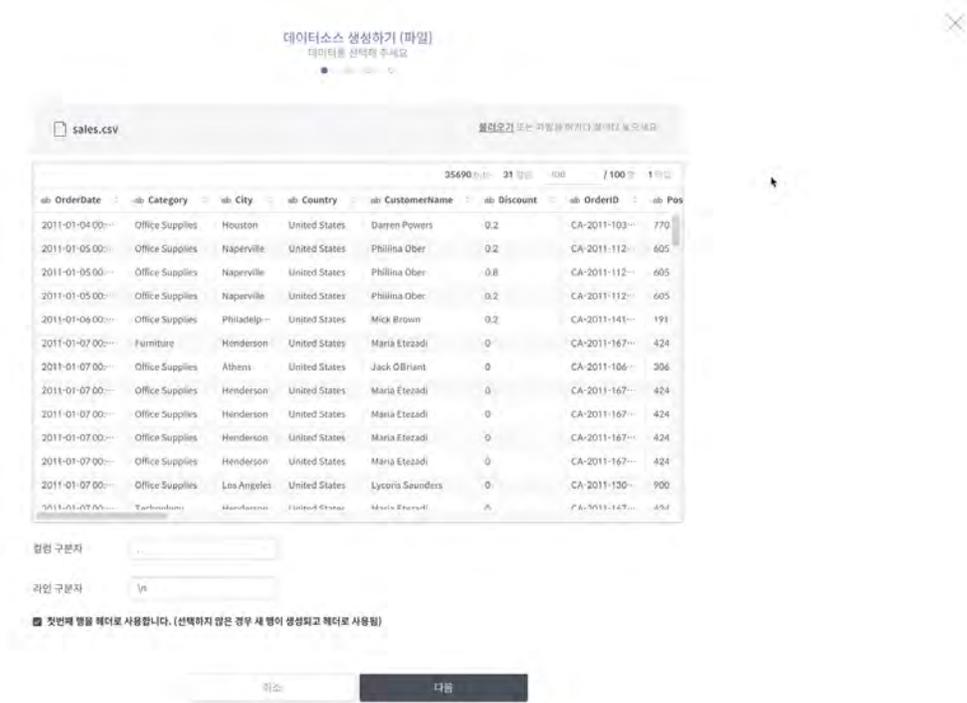


일단 튜토리얼에서는 **파일**을 눌러 로컬 폴더에서 데이터를 가져옵니다. 다른 원천에서 데이터 소스를 만드는 방법은 [데이터 소스 만들기](#) 문서를 참조하세요.

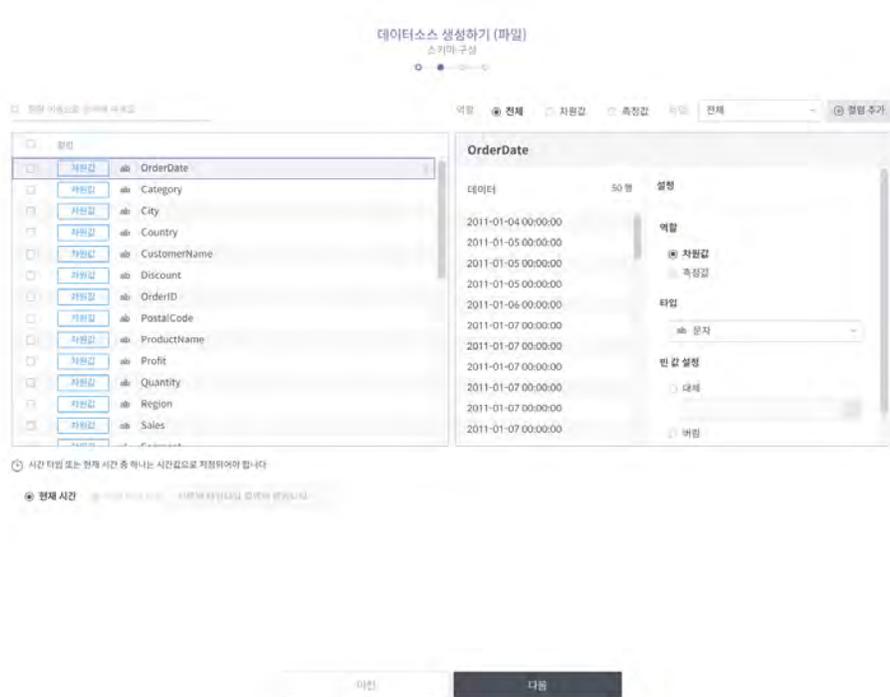
분석하고 싶은 데이터를 drag & drop하거나 디렉토리로부터 불러올 수 있습니다.



판매 현황 데이터를 드래그하면 컬럼과 라인 구분자에 따라 나타나는 데이터 샘플을 최대 100행까지 조회할 수 있습니다. 이 데이터는 기본적으로 설정된 구분자로 충분히 데이터를 잘 나타내는 것 같네요! 다음을 누릅니다.



이제 실제로 데이터를 보면서 컬럼의 타입을 알맞게 조정해야 합니다. 이 작업을 **데이터 스키마 구성**이라고 부릅니다.

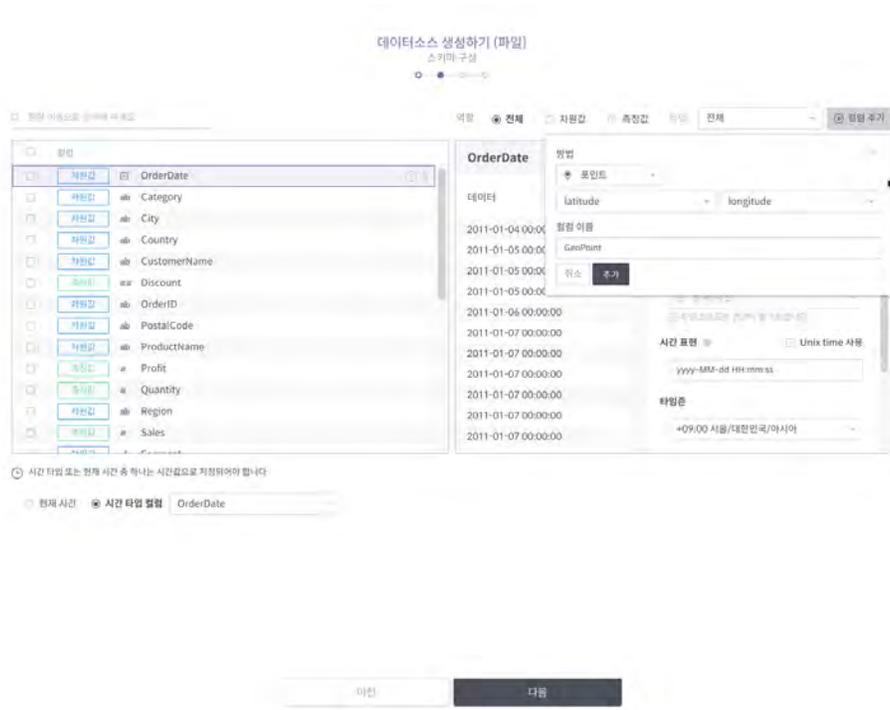


각 컬럼은 <차원값> (dimension) 또는 <측정값> (measure) 이라는 역할로 나뉩니다. 자세한 내용은 <차원값> 과 <측정값> 의 개념 문서를 참조하세요. 이 데이터에서는 Discount, Profit, Quantity, Sales, DaystoShipActual, SalesForecast, DaystoShipScheduled, SalesperCustomer, ProfitRatio 컬럼들을 측정값으로 변경해야 합니다.

그 다음은 컬럼의 데이터 타입을 적절하게 변경해 주어야 합니다. 기본적으로 차원값은 문자로, 측정값은 정수로 설정되어 있습니다. 데이터 샘플을 보면서 가장 알맞는 형식으로 변경해주세요. 아래에 이 데이터에서 변경할 사항들을 나열했습니다.

- Orderdate : 날짜/시간
- Discount : 소수
- ShipDate : 날짜/시간 (시간 표현을 yyyy. MM. dd. 로 변경한 후 체크박스 클릭하여 유효성 확인)
- SalesperCustomer : 소수
- ProfitRatio : 소수
- latitude : 위도
- longitude : 경도

마지막으로 새로운 컬럼을 만들어 줄 차례입니다. 우리는 위도와 경도 컬럼을 갖고 있으므로 Point 타입의 컬럼을 새로 만들 수 있습니다. 우측 상단의 **컬럼 추가** 버튼을 누르세요. **위도** 컬럼에 latitude 컬럼을 선택하고, **경도** 컬럼에 longitude 컬럼을 선택합니다. 컬럼 이름을 적절하게 입력한 후 **추가**를 누르세요. 새로운 Point 타입 컬럼이 생성되었습니다!



스키마 구성 작업이 마무리되었으면 **다음**을 누릅니다. 필요한 경우 Druid의 수집 설정을 변경하는 작업입니다. 지금은 기본 설정으로도 충분합니다.

데이터소스 생성하기 (파일)
수집 설정을 완료해 주세요

타임스탬프 설정

위도 단위

세그먼트 단위

데이터 범위

시작: 2010-12-30 20 종료: 2011-01-25 04 609 세그먼트 범위 수

종류

설정형 설정안함

이전 다음

마지막으로 데이터 소스의 이름과 설명을 입력합니다. 마침을 누르면 즉시 데이터 소스 상세로 넘어갑니다.

데이터소스 생성하기 (파일)
데이터소스 생성을 완료해 주세요

이름: 판매현황

파일: sales.csv

설명: 2010-2011 판매현황 데이터입니다.

이름

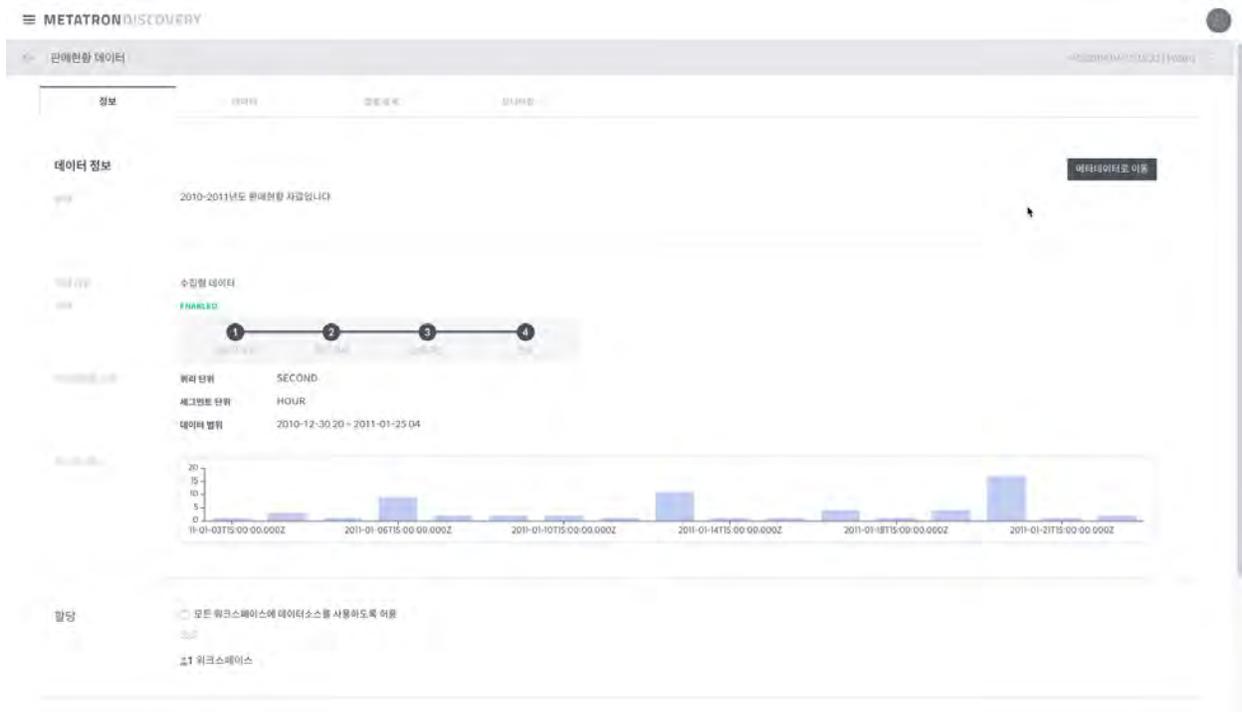
판매현황

설명

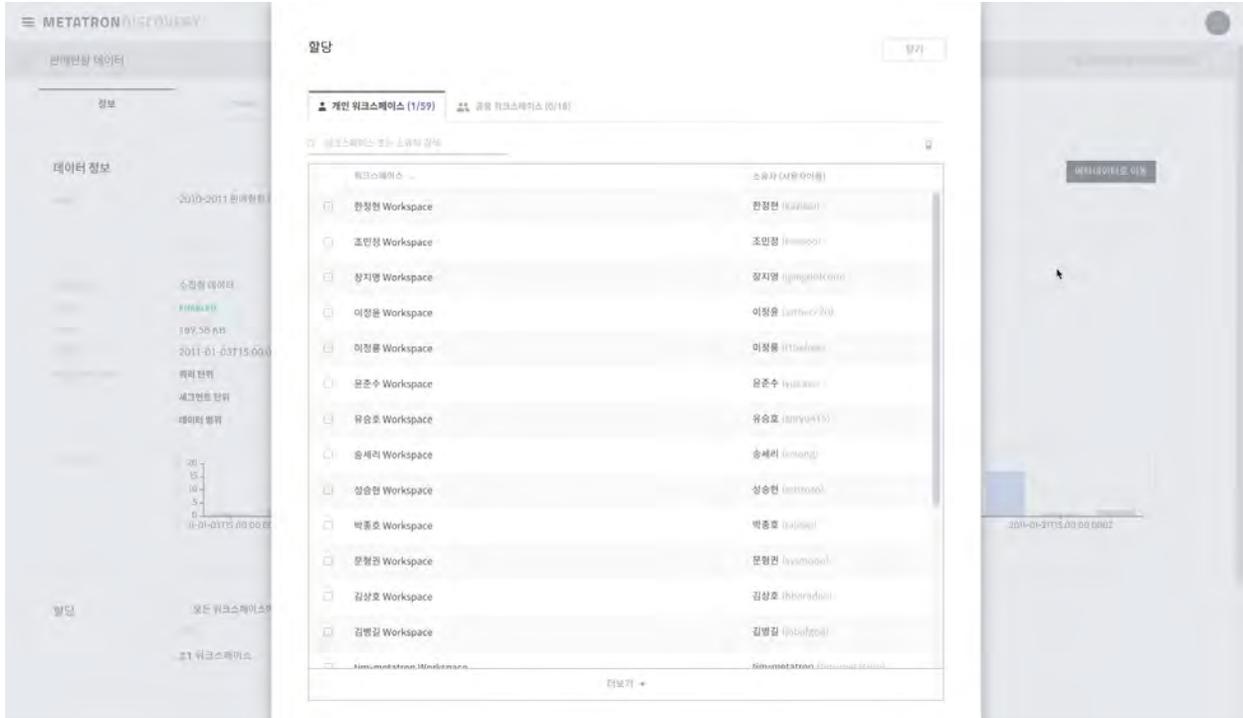
2010-2011 판매현황 데이터입니다.

이전 마침

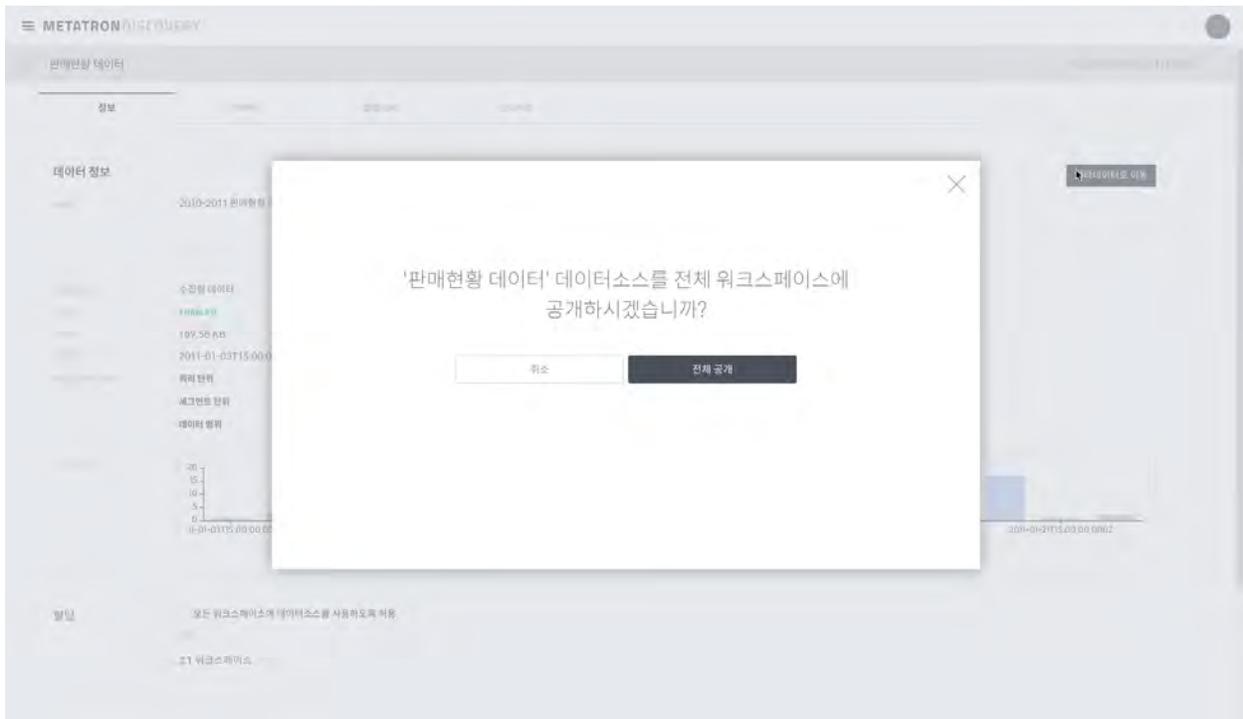
데이터 소스 상세에서는 적재 현황을 실시간으로 볼 수 있습니다. 몇 분 기다리면 아래와 같이 적재가 성공했음을 알리며 히스토그램이 나타납니다. 혹시 다른 데이터 소스를 적재하다 에러가 났을 경우 상세 링크를 클릭하여 Druid 적재 로그를 조회할 수 있습니다. 컬럼명 중복, 컬럼 타입과 불일치하는 데이터 등으로 인해 적재가 실패할 수 있습니다. 이 경우 원인을 찾아 다시 적재를 시도해보아야 합니다.



이 데이터 소스를 다른 사용자들에게 오픈하려면 할당에서 모든 워크스페이스에 데이터소스를 사용하도록 허용 체크박스에 체크합니다. 모든 사용자가 아니라 특정 사용자들에게만 오픈하고 싶으면 수정을 클릭하여 할당하고자 하는 개인 사용자들 또는 팀 워크스페이스를 선택합니다.



이 예제에서는 모든 사용자가 사용할 수 있도록 Open Data로 설정하겠습니다.



적재된 데이터는 데이터 탭에서 조회할 수 있습니다.

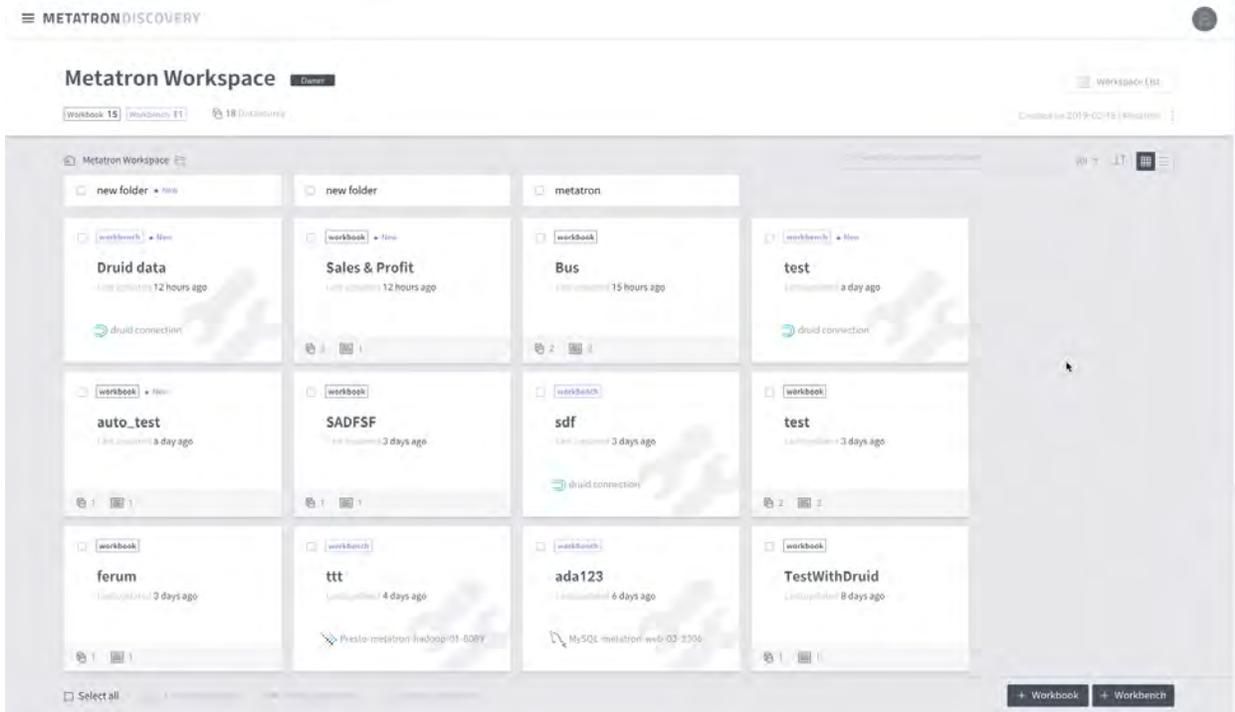
The screenshot shows the METATRON DISCOVERY interface with a data table. The table has the following columns: GeoPoint, OrderDate (UTC+9), Category, City, Country, CustomerName, Discount, OrderID, PostalCode, ProductName, Profit, Quantity, Region, Sales, and Seq. The data rows include information such as OrderDate (e.g., 2011-01-04), Category (Office Supp), City (Houston), Country (United States), CustomerName (Darren Powers), Discount (0.2), OrderID (CA-2011-1-77095), PostalCode (60540), ProductName (Message Book), Profit (6), Quantity (2), Region (Central), Sales (16), and Seq (16).

축하합니다! 이제 데이터 소스를 사용해 볼 차례네요. 다음 스텝으로 넘어가볼까요?

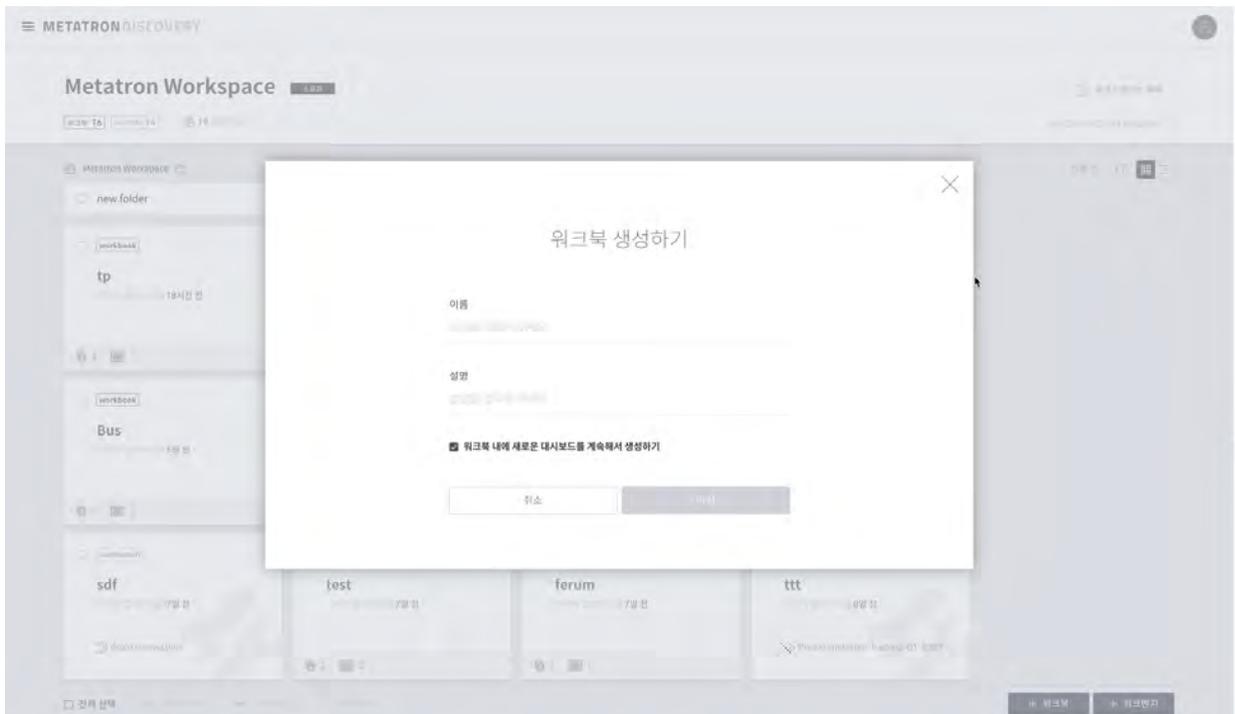
- Step2. 워크북 만들기

1.2 Step2. 워크북 만들기

분석을 위한 데이터가 준비되었나요? 그럼 이제 워크북을 만들 차례입니다. 워크북은 데이터 시각화 기능을 포함하는 모듈입니다. 좌측 상단의 Metatron Discovery 로고를 클릭하면 메인 화면인 내 개인 워크스페이스로 이동합니다.



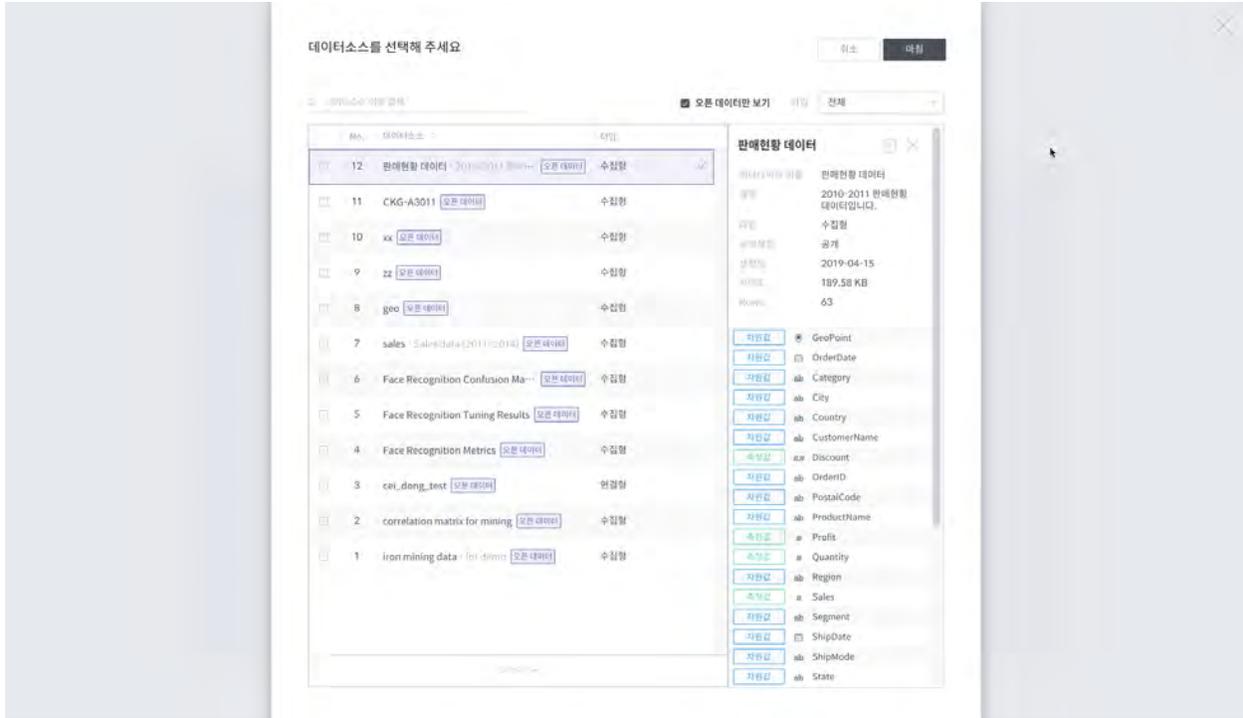
우측 하단의 + 워크북 버튼을 눌러 워크북을 만들어 볼까요? 만들 워크북의 이름과 설명을 입력합니다. 워크북을 만드는 즉시 이어서 대시보드를 만들도록 체크박스에 표시가 되어 있습니다. 여기서 각 워크북은 여러 개의 대시보드를 포함하고, 각 대시보드는 또 여러 개의 차트를 포함하는 구조입니다.



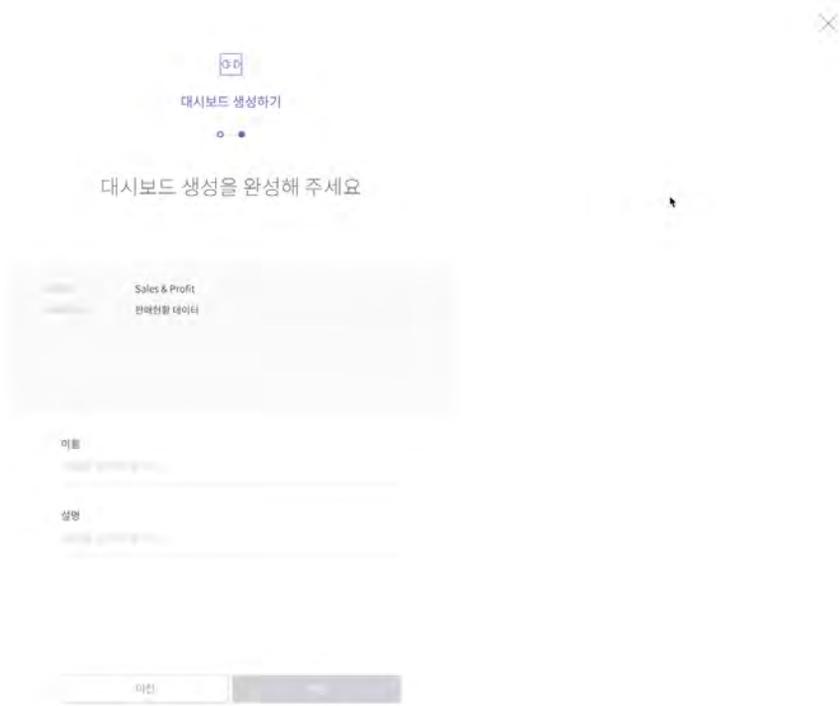
이어서 대시보드를 만들어야 합니다. 대시보드에는 시각화할 데이터 소스가 있어야 합니다. 이 데이터 소스는 단일 소스이거나, join으로 연결된 데이터 소스일 수도 있습니다. 더 자세한 내용은 [대시보드 만들기](#) 문서를 참고하세요. 본 튜토리얼에서는 Step1에서 적재한 판매현황 데이터 하나만 사용합니다.



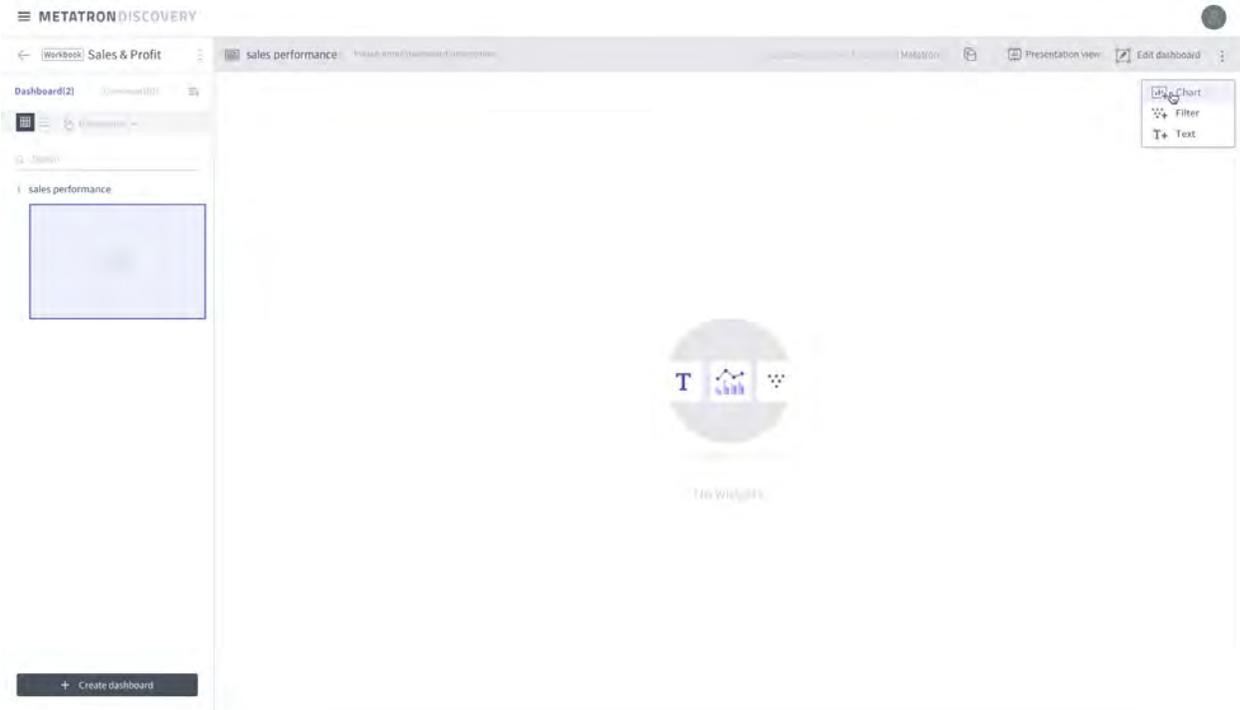
+데이터 소스 추가 버튼을 누르면 데이터 소스 선택 팝업이 나타납니다. 판매현황 데이터를 검색하거나 **오픈 데이터만 보기에** 체크하여 공개된 데이터 중에서 찾아냅니다.



마지막으로 대시보드의 이름과 설명을 입력합니다.



워크북 내에 대시보드가 생성되었습니다! 이제 이 위젯들을 추가해서 대시보드를 구성하는 일만 남았네요.

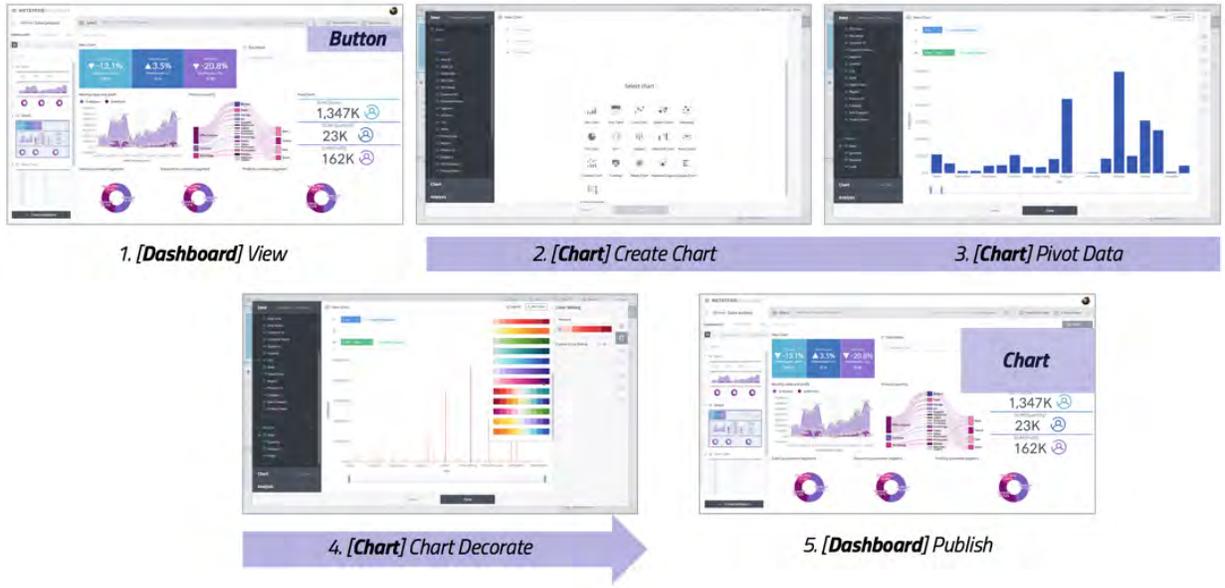


다음 단계로 넘어가 볼까요?

- Step3. 대시보드 구성하기

1.3 Step3. 대시보드 구성하기

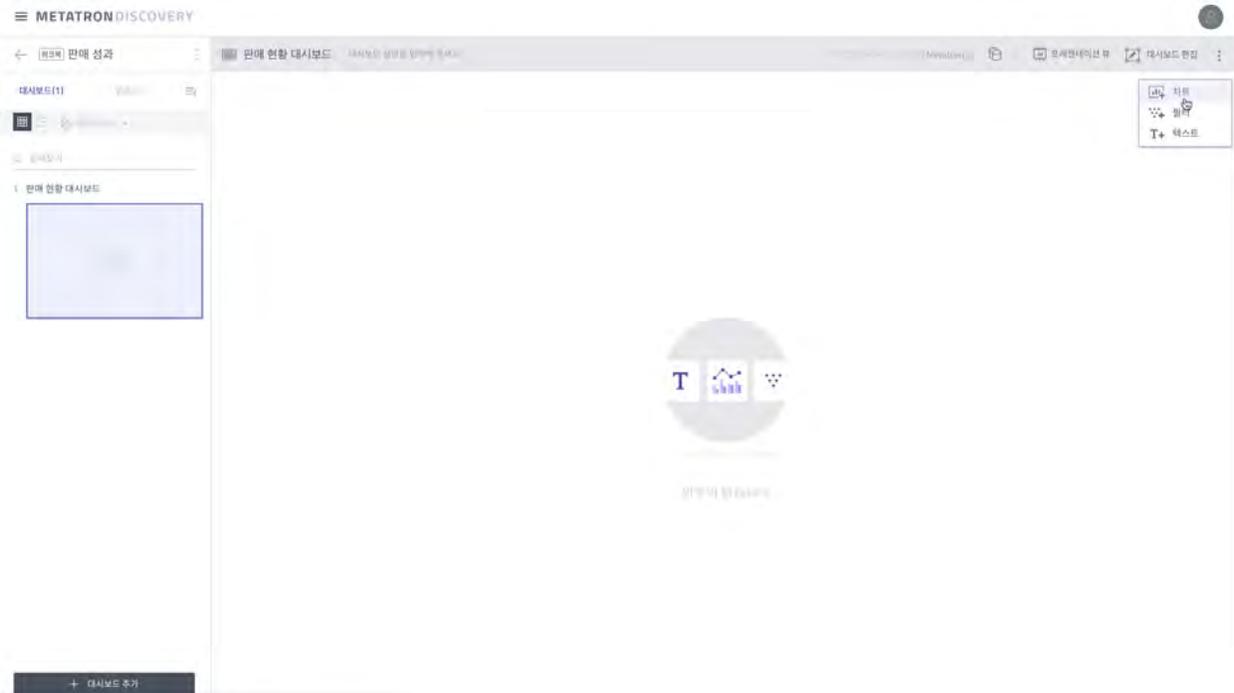
마지막 단계는 빈 대시보드에 차트 위젯, 텍스트 위젯, 필터 위젯을 생성하여 대시보드를 구성하는 작업입니다. 대시보드 편집은 아래와 같은 순서로 진행됩니다.



이제 앞서 만든 판매 현황 데이터를 이용해 아래와 같이 핵심지표 차트와 선형 차트를 그려 대시보드를 만들어보겠습니다.

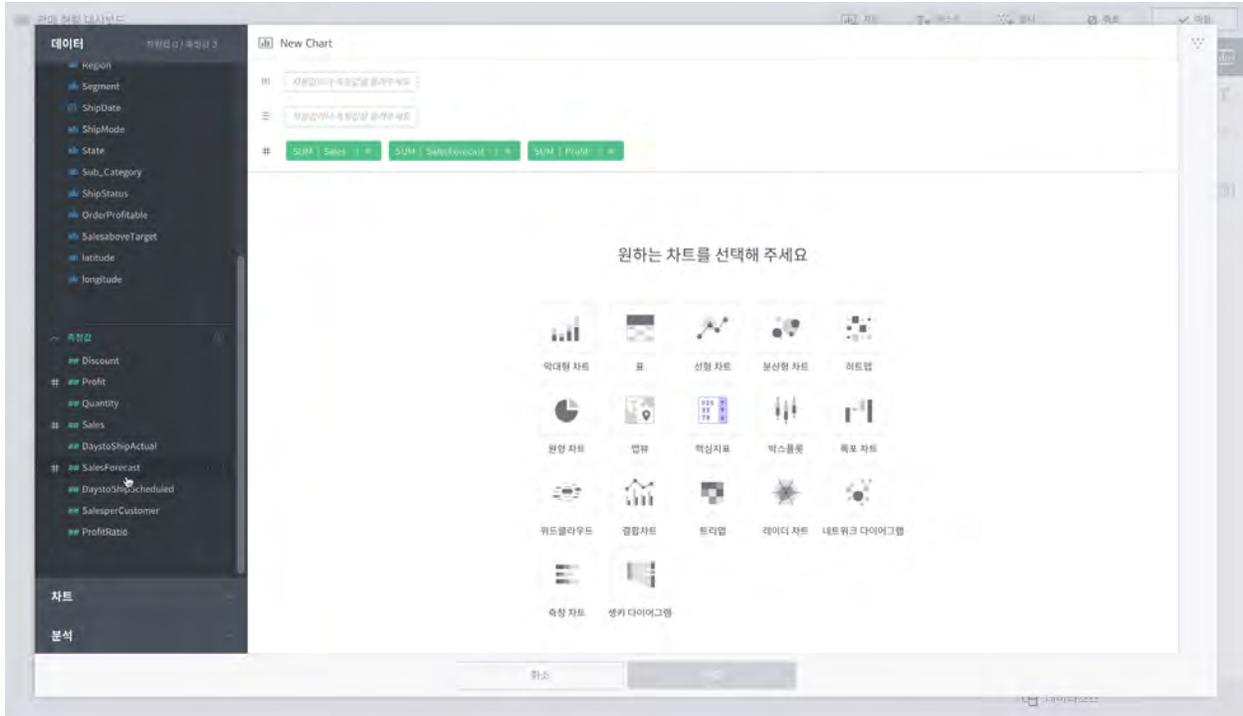


빈 대시보드에서 차트 버튼을 눌러서 차트를 만들어봅시다.

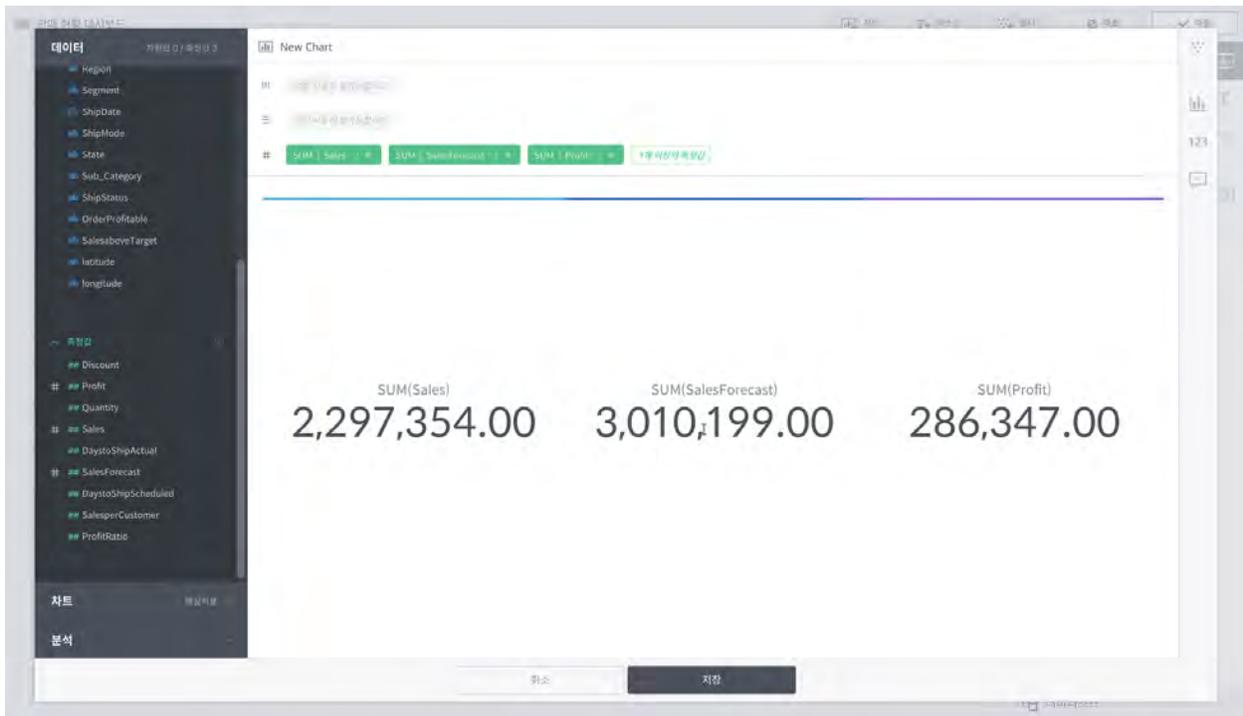


1.3.1 핵심지표 차트 만들기

처음 만들어 볼 차트는 핵심지표 (KPI) 차트입니다. 핵심지표 차트는 조직의 목표를 가장 직관적으로 보여주는 가장 단순하면서도 강력한 차트입니다. 우리가 만들 대시보드는 판매 현황을 잘 나타내는 것이 목표입니다. 따라서 총 판매액, 예상 총 판매액, 총 이익을 핵심지표 차트로 보여주려고 합니다. 어떻게 해야 할까요? 단순히 Sales, SalesForecast, Profit 세 개의 측정값 컬럼을 데이터 메뉴에서 클릭하세요. 이 작업을 피벗팅 (pivoting) 이라고 합니다. 피벗된 컬럼들은 자동으로 집계되어 선반에 올라갑니다. 컬럼을 선반에 올리면 즉시 알맞는 차트 종류가 추천됩니다. 추천된 **핵심지표 차트**를 클릭해 볼까요?

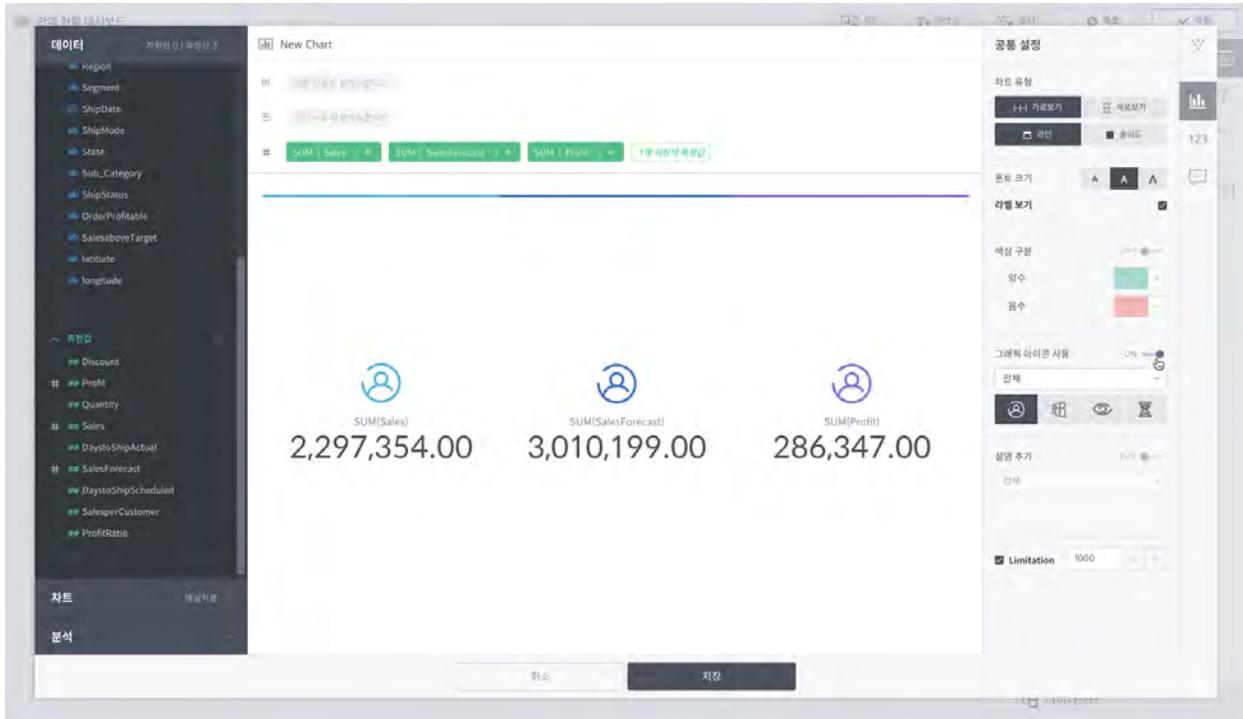


아래와 같이 핵심지표 차트가 만들어졌습니다. 조금 더 보기 좋게 만들기 위해서 우측의 차트 속성 메뉴를 이용해봅시다.

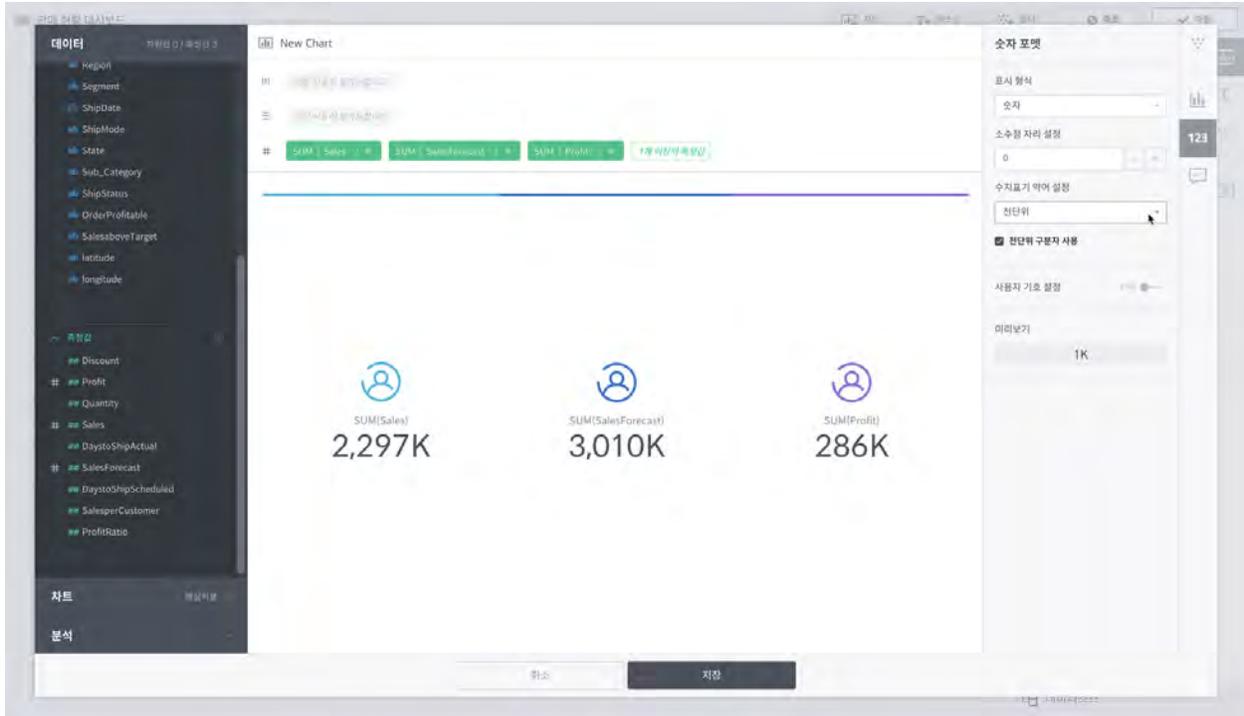




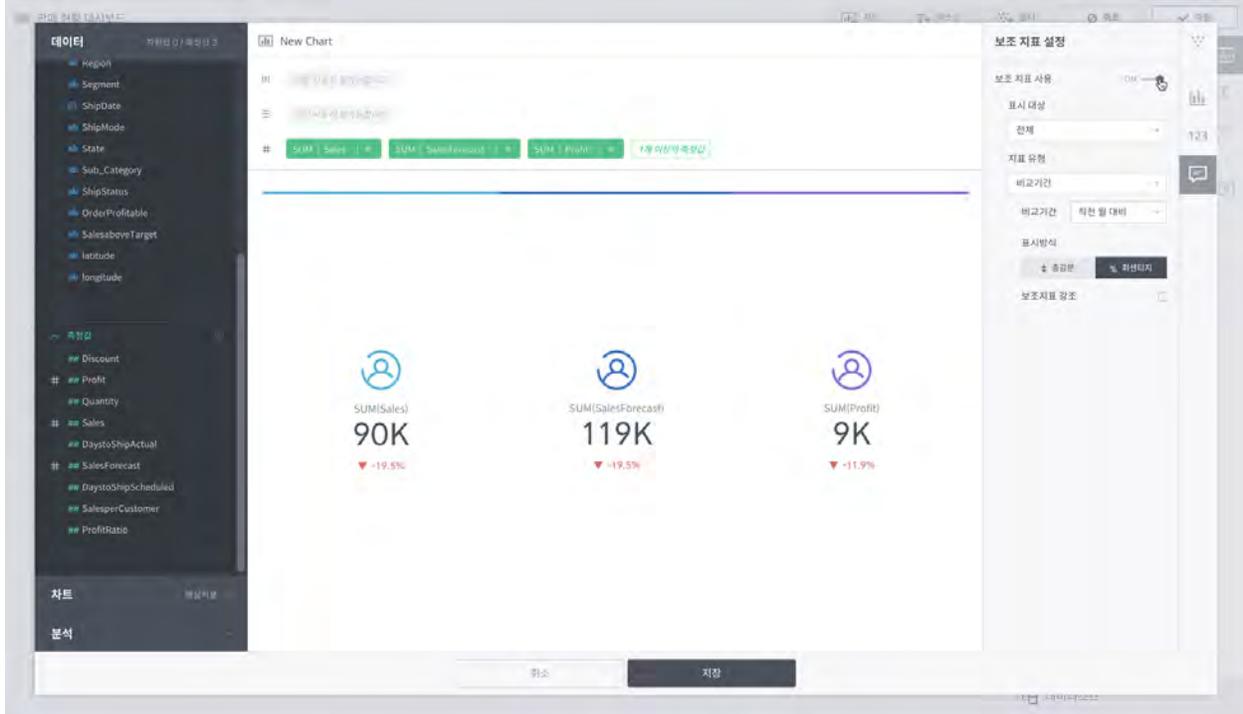
을 눌러 **공통 설정** 패널에서 각 측정값 컬럼에 아이콘을 추가하고



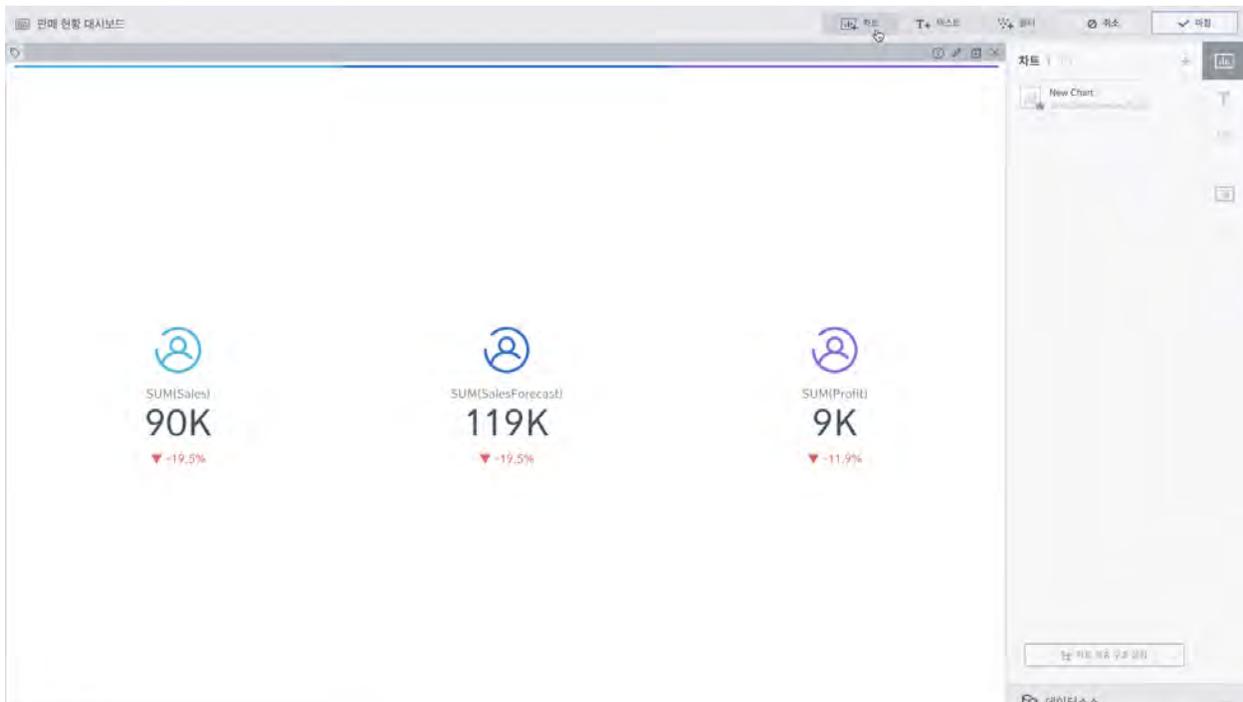
123 을 눌러 **숫자 포맷** 패널에서 소수점 표기 방식과 약어 표기 방식을 변경해주었습니다.



핵심지표 차트에서 특히 중요한 건 이전 대비 얼마나 성과가 있는지를 파악하는 것 입니다.  을 눌러 보조 지표 설정 패널에서 보조 지표를 설정하고 전월 대비 몇 %나 잘하고 있는지 확인해보겠습니다. 원한다면 원 지표 대신 보조 지표를 강조할 수도 있습니다.

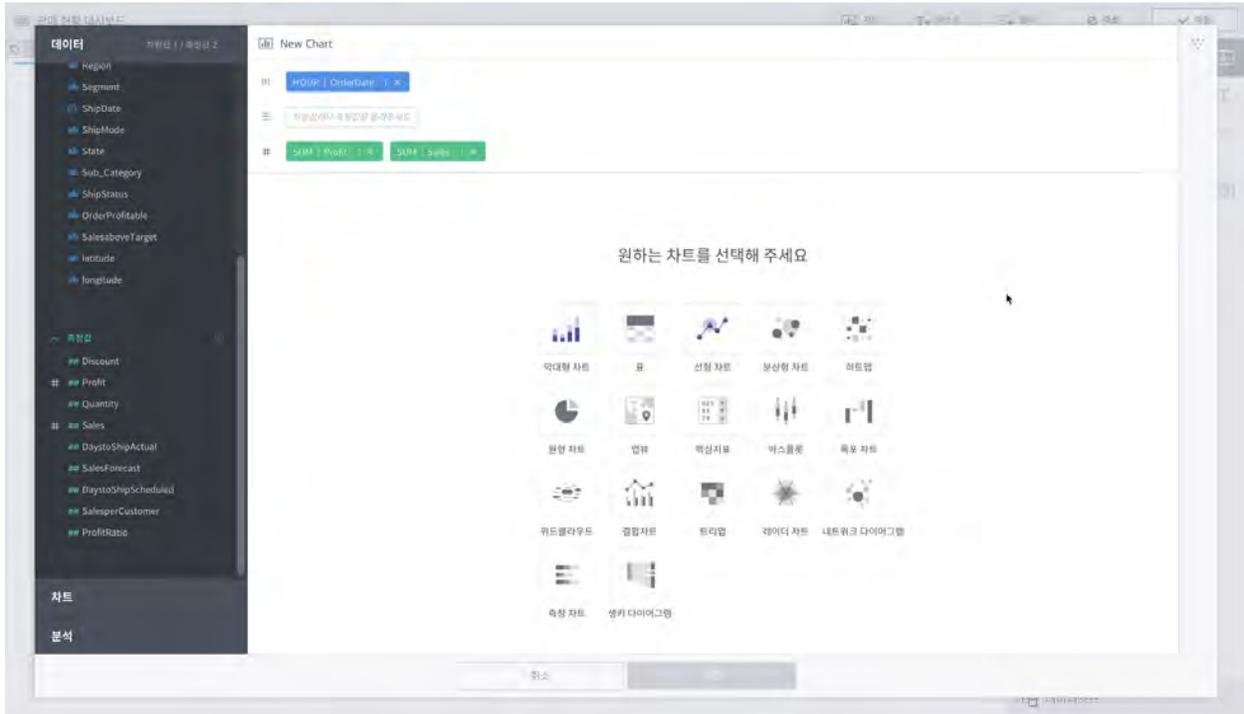


저장을 누르면 차트가 대시보드에 나타납니다.

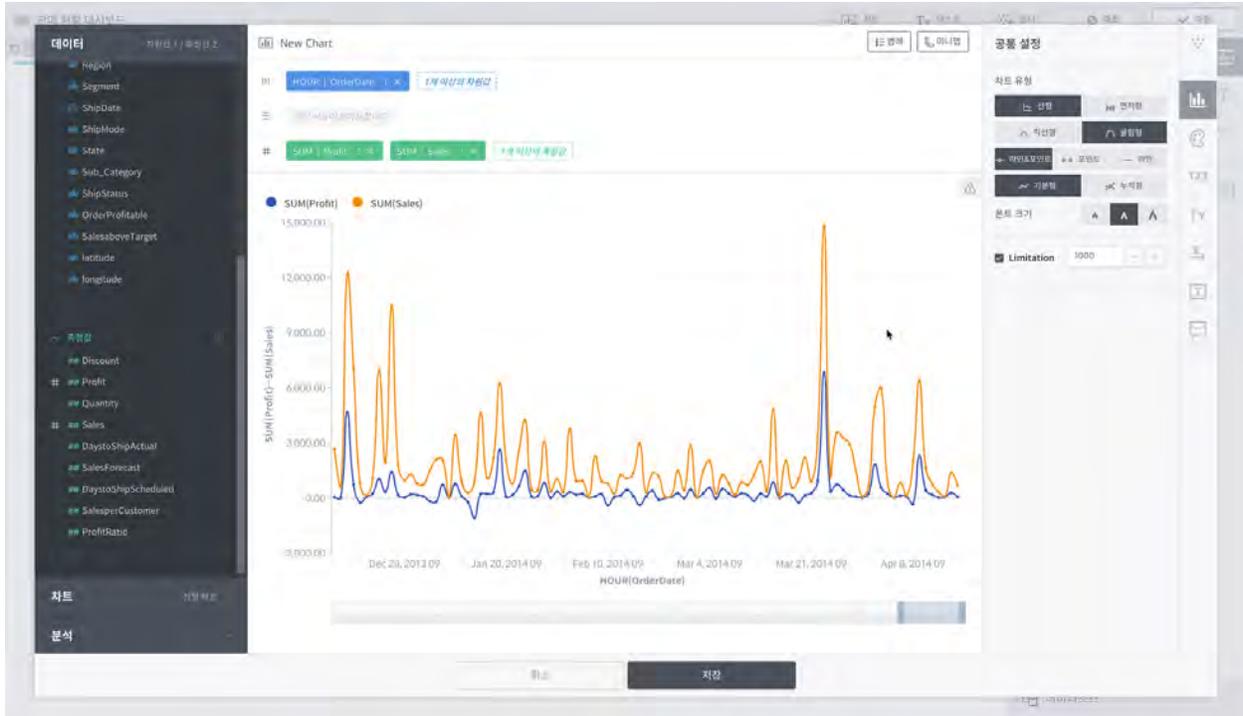


1.3.2 선형 차트 만들기

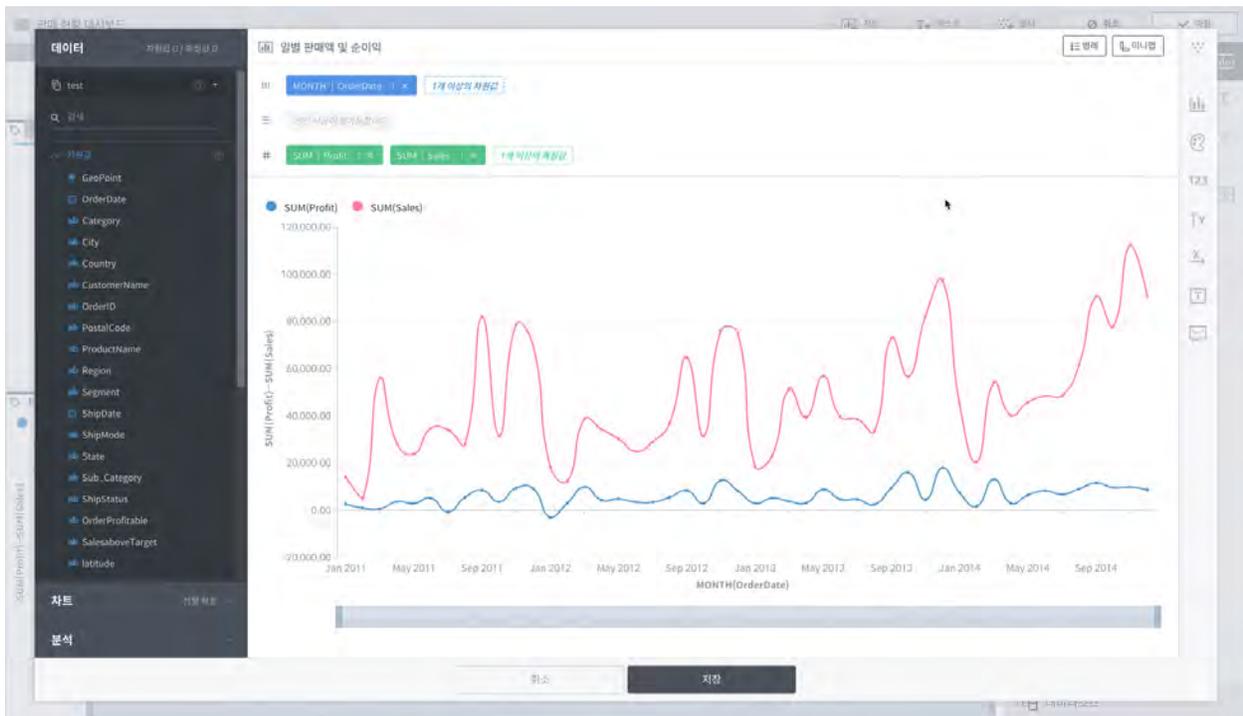
그 다음으로 가장 기본적인 차트인 선형 차트를 그려보겠습니다. 시간에 따라 매출과 수익이 어떻게 변하는지 한 눈에 볼 수 있도록 해 볼까요? 다시 **차트** 버튼을 눌러 새 차트 그리기로 들어갑니다. OrderDate, Profit, Sales 컬럼을 눌러 시간 차원에 따라 값들이 변하는 모습을 보고자 합니다. 추천된 **선형 차트**를 클릭해보세요.



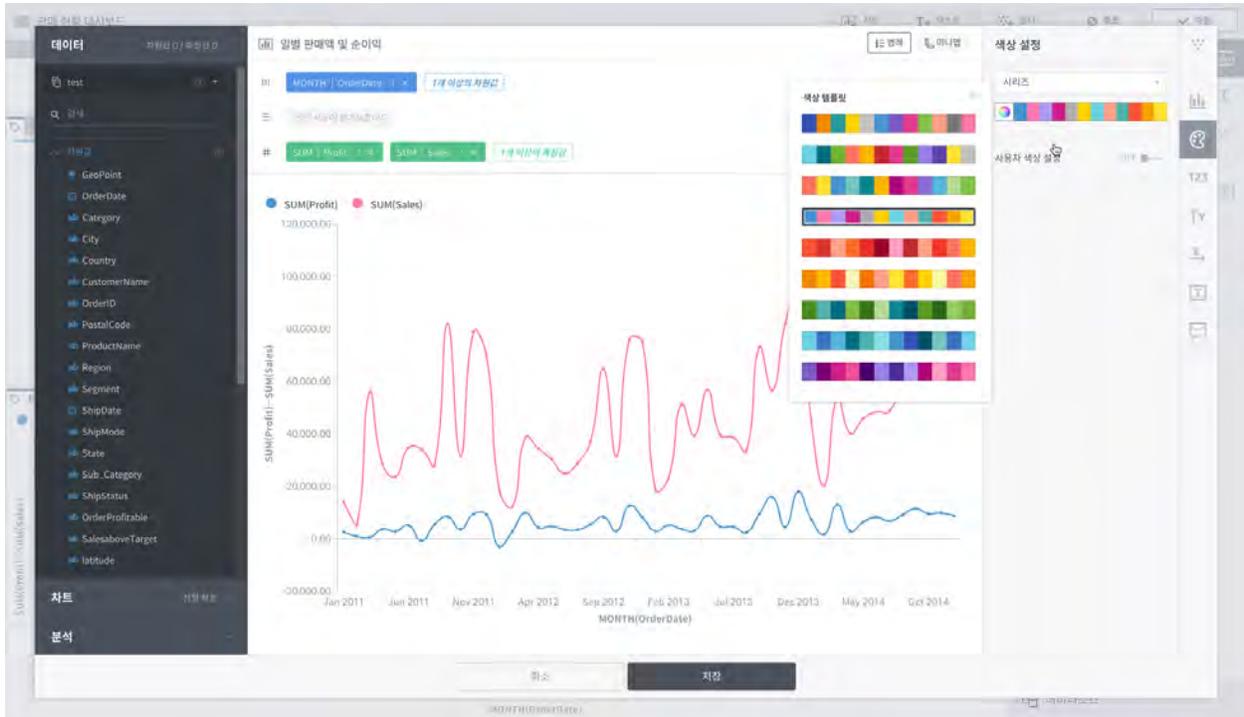
선형 차트가 그려졌습니다. 차트 속성창을 열어서 라인의 모양을 둥글게 변경해주었습니다.



지금은 OrderDate의 집계 단위가 시간이라 너무 데이터가 많습니다. 월 단위로 보기 위해서 선반에 올라간 OrderDate 컬럼의 메뉴에서 Granularity를 월 (Month)로 선택해줍니다. 이제 모든 데이터가 보이네요! 우측 상단의 미니맵을 클릭해서 차트에서 미니맵을 제거해줍니다.



우측 메뉴에서  을 눌러 색상 설정 패널에서 색상 설정도 변경해줍니다.



저장을 누르고 차트를 drag & drop하여 적절하게 위치를 바꿔주세요. 텍스트 위젯도 추가하여 대시보드에 적절한 정보를 추가합니다. 마침을 누르면 대시보드 편집이 완료됩니다.



본 튜토리얼을 통해 간단한 차트 두 가지를 완성해보았습니다. 대시보드는 인터랙티브하게 동작하여 차트를 선택하거나 필터를 추가하면 원하는 방식으로 프레젠테이션할 수 있습니다. 또 필요할 때마다 언제든지 차트를 수정하거나 추가·삭제할 수 있습니다.



이제 Metatron Discovery에 대해 더 알아보시겠어요?

- [Metatron Discovery 개요](#)
- [Metatron Discovery 구성](#)
- [Metatron 기본 엔진: Druid](#)

Metatron Discovery 소개

Metatron Discovery는 Metatron 운용 서버 클러스터에 적재된 데이터를 단순하면서도 고도화된 방식으로 분석한 후 그 결과를 다양한 형식의 차트와 보고서로 사용자 PC에 출력해주는 솔루션입니다. 웹 템플릿 형식으로 구성되어 있으며 계정이 발급되면 아무 PC에서나 원격으로 접속할 수 있기 때문에 매우 편리한 접근성을 보장합니다.

본 장에서는 Metatron Discovery의 기술 기반과 구조, 그리고 Metatron 기본 엔진인 Druid에 대해 소개합니다.

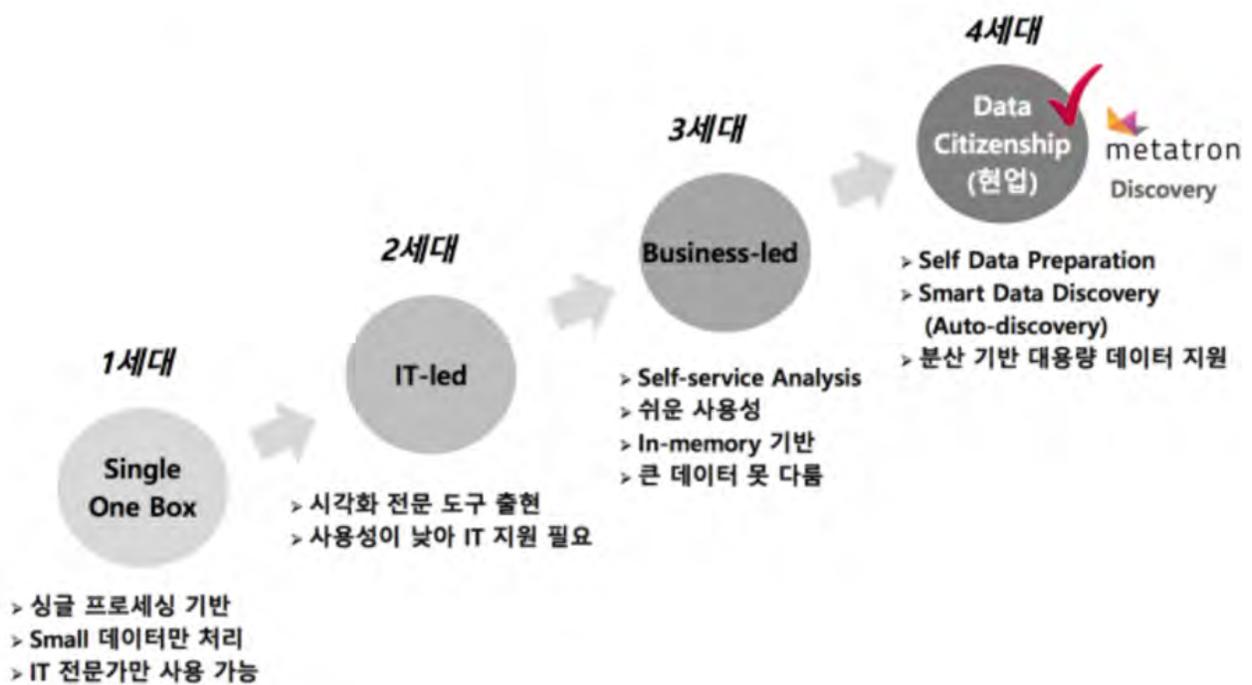
2.1 Metatron Discovery 개요

Metatron Discovery는 OLAP, 시각화, 머신러닝 기술이 융합하여 비전문가도 데이터로부터 상위 레벨의 가치를 빠르고 손쉽게 얻을 수 있는 4세대 OLAP 기반 Business Intelligence (BI) 솔루션입니다.



2.1.1 4세대 BI 솔루션

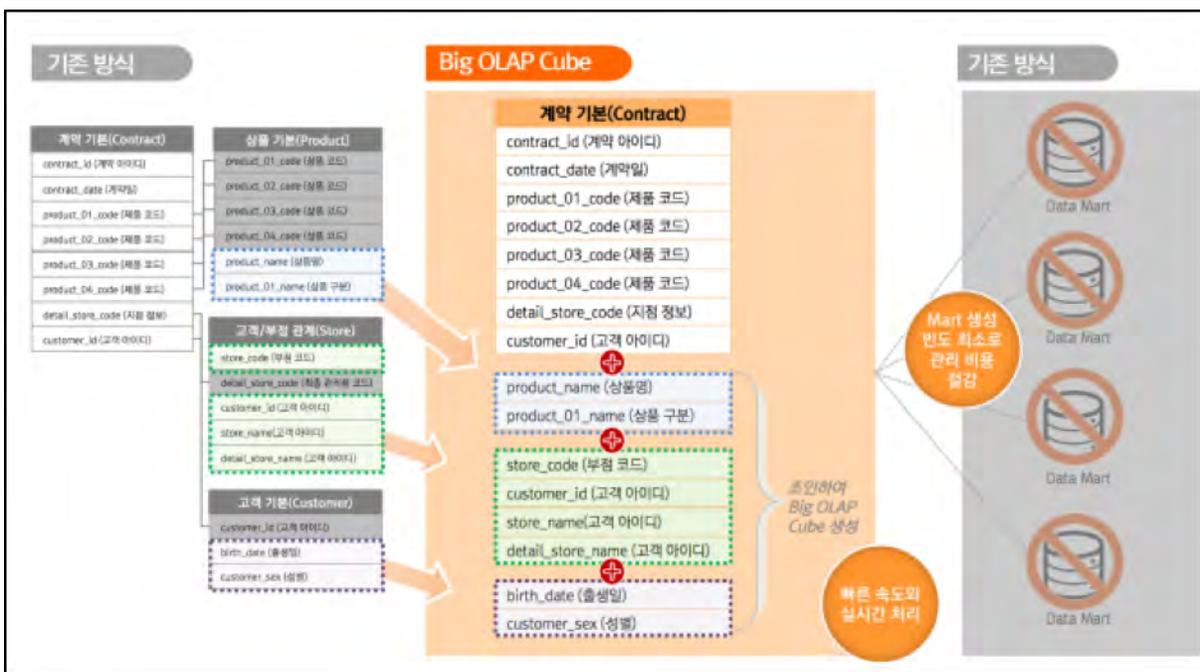
아래 그림은 1세대부터 4세대까지의 BI의 흐름을 나타낸 그림입니다.



현재 BI 시장은 2세대, 3세대 제품이 주류를 이루고 있으며, 4세대 제품이 주목받고 있습니다. Metatron Discovery는 4세대 BI 솔루션으로서, Self & Ad-hoc Discovery를 지향하며 빅데이터에 대해서도 빠른 응답 속도를 보장합니다.

2.1.2 Big OLAP 기반

Metatron Discovery는 대용량 Fact 데이터를 기준으로 다양한 차원 (Dimension) 데이터를 결합하여 하나의 Big OLAP Cube (Mart) 를 생성할 수 있습니다.

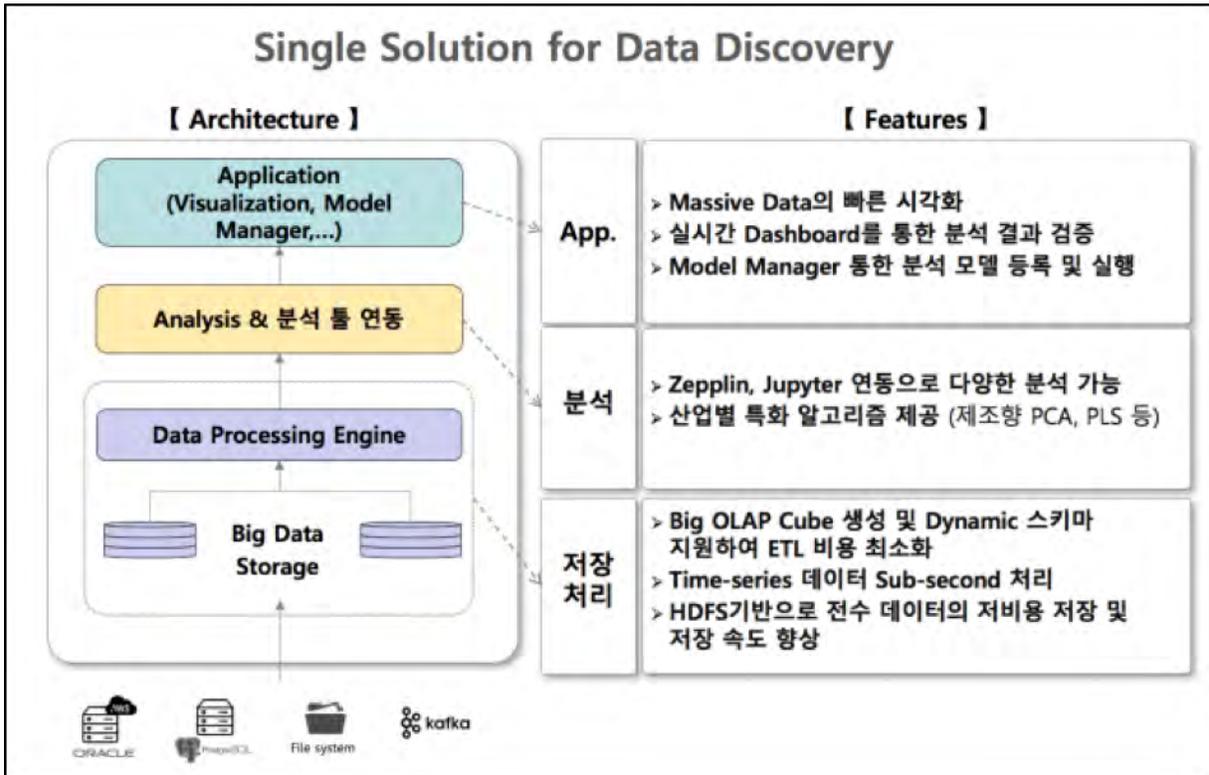


Big OLAP Cube로 사용 시, 다음과 같은 장점이 있습니다.

- 데이터 마트 (data mart) 의 개수를 최소화
 - 데이터 마트 생성을 위한 ETL 비용 감소
 - 구조 변경에 따른 영향도를 최소화 할 수 있음
 - Fact 데이터를 모두 저장하므로, 다양한 요구사항에 대응 가능
- 분산 아키텍처 기반으로 큰 규모의 데이터도 저장 가능하며, 빠른 속도로 결과 출력
- Dynamic 스키마 채용으로 스키마 변경 시에도 별도의 스키마 재정의가 필요치 않음
- 실시간 처리 테이블을 원본 그대로 저장하므로 레코드 단위의 실시간 처리 가능

2.1.3 Metatron Discovery 아키텍처

Metatron Discovery는 대용량 데이터에 대한 프리퍼레이션 (preparation) 부터 시각화 기반 데이터 탐색, 고급 분석 까지 Data Discovery의 전 과정을 지원하는 End-to-End 솔루션입니다. 아래 그림은 Metatron의 아키텍처와 주요 특징을 정리한 것입니다.



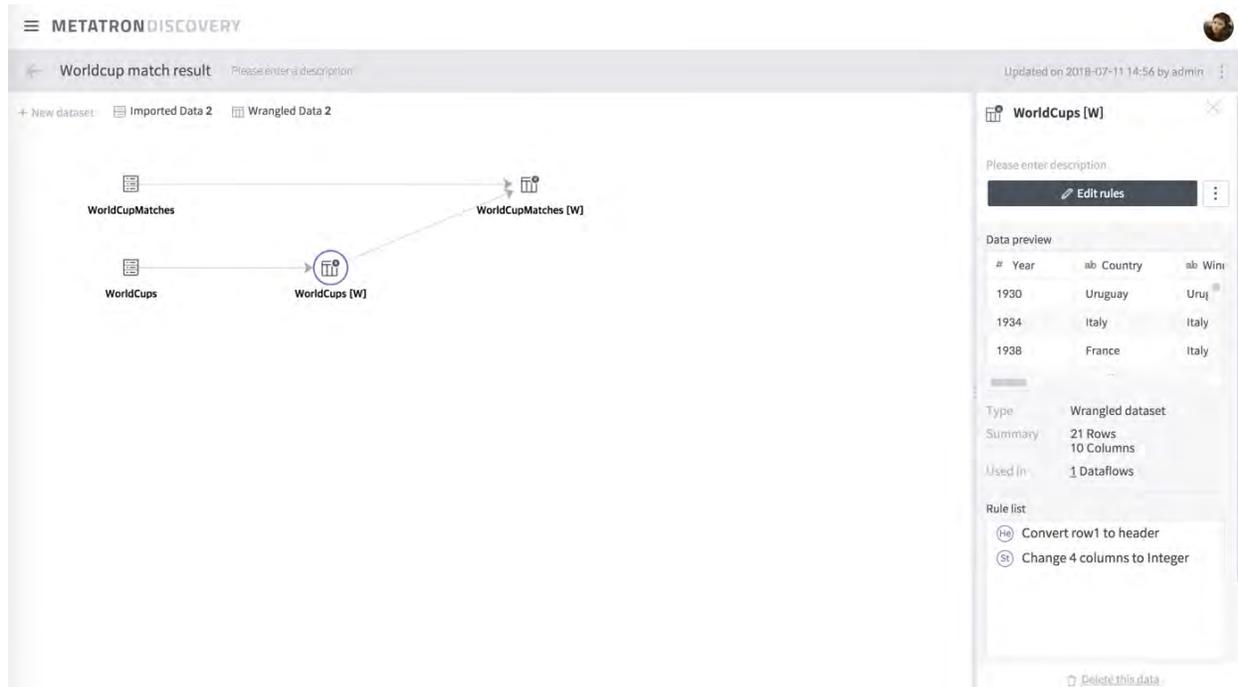
2.2 Metatron Discovery 구성

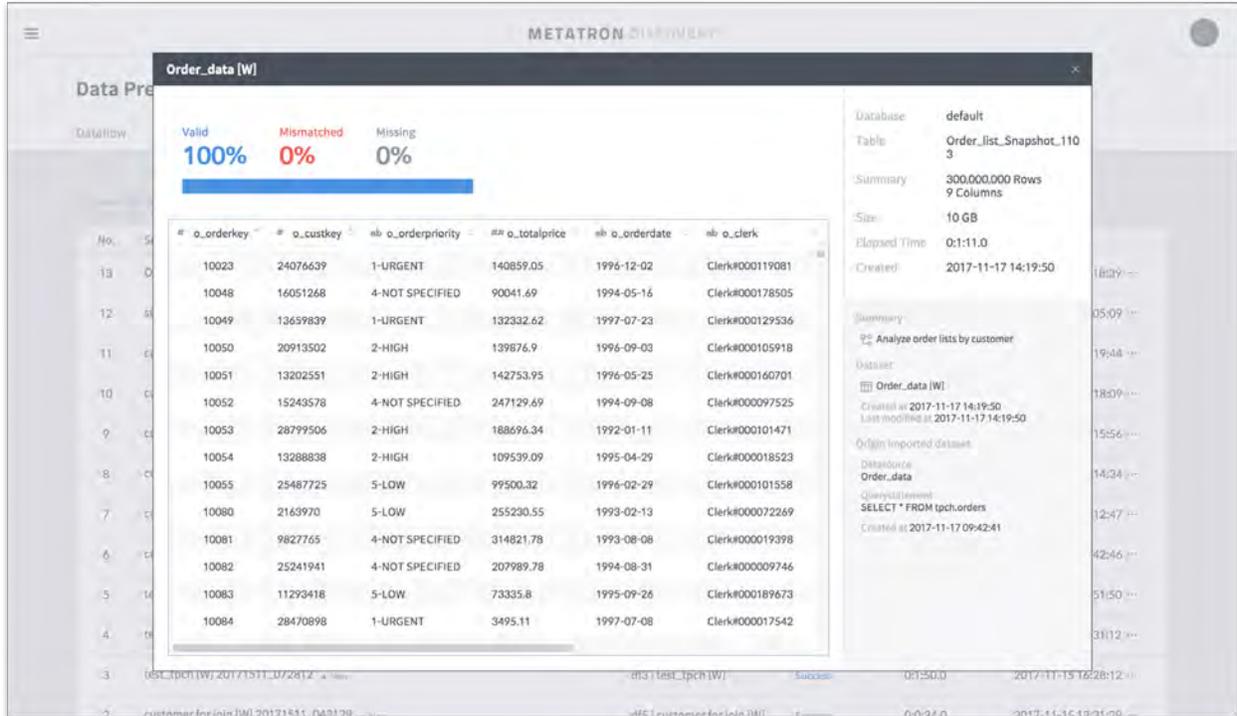
Metatron Discovery는 Metatron 운용 서버에 적재된 데이터 소스나 그 밖의 외부 데이터 소스로부터 원하는 데이터를 불러와서 각종 고급 분석 기능을 통해 분석한 후, 그 결과를 다양한 형식의 차트와 보고서로 출력하는 기능을 합니다. 본 모듈을 이용하려면 다음과 같은 전반적인 구조를 이해해야 합니다.



2.2.1 데이터 프리퍼레이션

데이터 프리퍼레이션은 원천 데이터에서부터 데이터를 정제 및 가공하여 Metatron으로 적재하는 기능을 제공합니다. 데이터 프리퍼레이션에 대한 자세한 설명은 데이터 프리퍼레이션 항목을 참조하십시오.





2.2.2 데이터 스토리지

데이터 스토리지는 분석·시각화를 위해 Metatron 엔진에 적재된 데이터를 관리합니다. 데이터 관리 기능에 대한 자세한 설명은 데이터 관리 항목을 참조하십시오.

≡ METATRON DISCOVERY


← Financial_data SBC Financial ERP Data
Updated on 2018-08-16 18:15 by Administrator

Information

Data

Column details

Monitoring

Data Information

Source type: FILE

Status: Enabled (Datasource available via engine rules)

Size: 115.09 KB

Duration: 2013-09-01T00:00:00.000Z ~ 2013-09-01T00:00:00.000Z

Timestamp settings: Segment Granularity: MONTH
Query Granularity: NONE

Histogram Detail



Timestamp	Frequency
2013-09-01T00:00:00.000Z	30
2013-10-01T00:00:00.000Z	60
2013-11-01T00:00:00.000Z	30
2013-12-01T00:00:00.000Z	30
2014-01-01T00:00:00.000Z	30
2014-02-01T00:00:00.000Z	30
2014-03-01T00:00:00.000Z	30
2014-04-01T00:00:00.000Z	30
2014-05-01T00:00:00.000Z	30
2014-06-01T00:00:00.000Z	30
2014-07-01T00:00:00.000Z	60
2014-08-01T00:00:00.000Z	30
2014-09-01T00:00:00.000Z	30
2014-10-01T00:00:00.000Z	30
2014-11-01T00:00:00.000Z	60
2014-12-01T00:00:00.000Z	30
2015-01-01T00:00:00.000Z	60

Permission

Allow all workspaces to use this datasource

All workspaces

Ingestion information

Master data: Type excel



Create data connection

Please set required items and complete data connection creation

DB type

Oracle ✓
 MySQL
 PostgreSQL
 Hive
 presto
 APACHE PHOENIX
 Tiberio

Server

Host	Port	SID
http://192.10.20.85	3306	

URL only

User ID for test: polaris

Password for test: *****

Security

Always connect
 Connect by user's account
 Connect with ID and password

Validation Check: Invalid Connection. Please check server and account information

Permission

1 Workspace [Edit](#)

Allow all workspaces to use this datasource

Advanced setting

Socket timeout: 60 Sec

Connection name

Previous Next

2.2.3 데이터 분석 및 시각화

아래 각 모듈은 데이터 스토리지에 저장한 데이터를 사용자가 시각화 기반 탐색, 분석하는 기능을 제공합니다.

워크스페이스

워크스페이스는 조직 내에서 사용할 워크북, 워크벤치, 노트북을 권한에 따라 관리할 수 있습니다. 워크스페이스 기능에 대한 자세한 설명은 [워크스페이스](#) 항목을 참조하십시오.

≡ METATRON DISCOVERY 🌐

Admin workspace Owner Workspace List

Workbook 20 | Notebook 0 | Workbench 7 | **Datasource 23** Created on 2018-06-11 by Administrator

Datasource (23)

🔍 Search by datasource name Show open data only Type: All

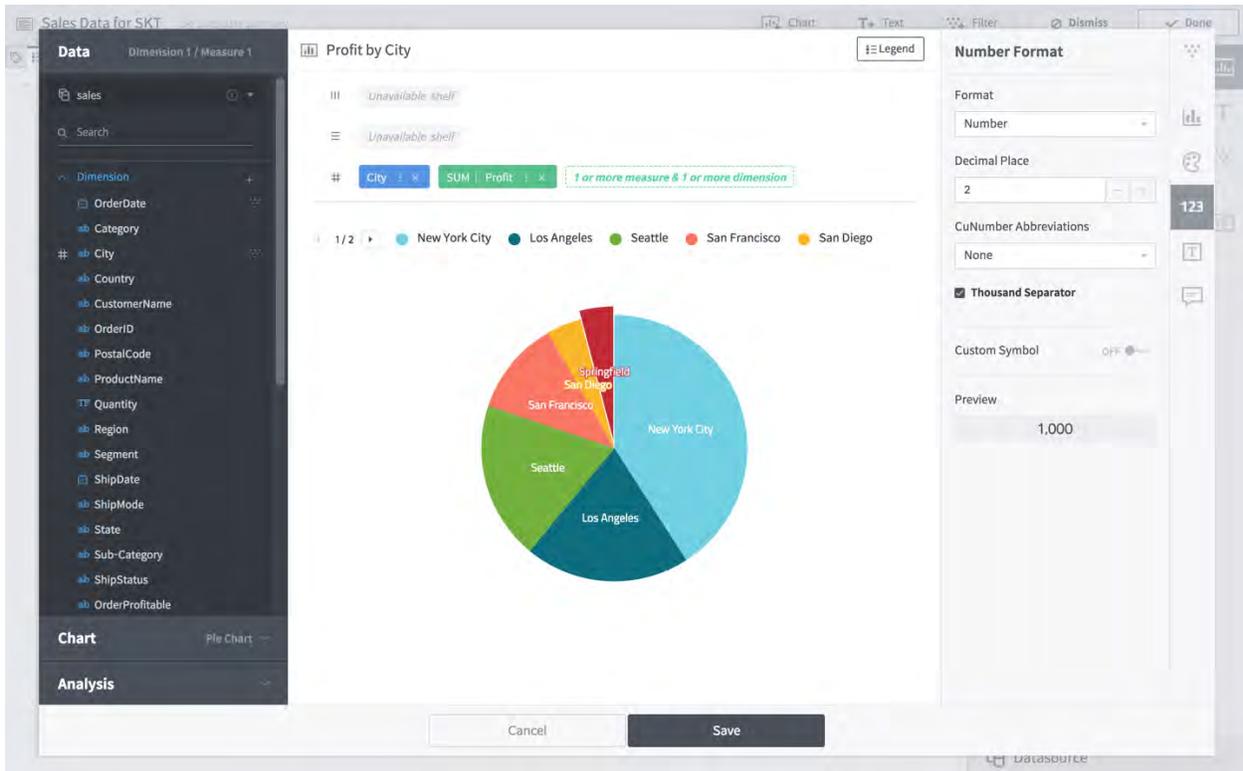
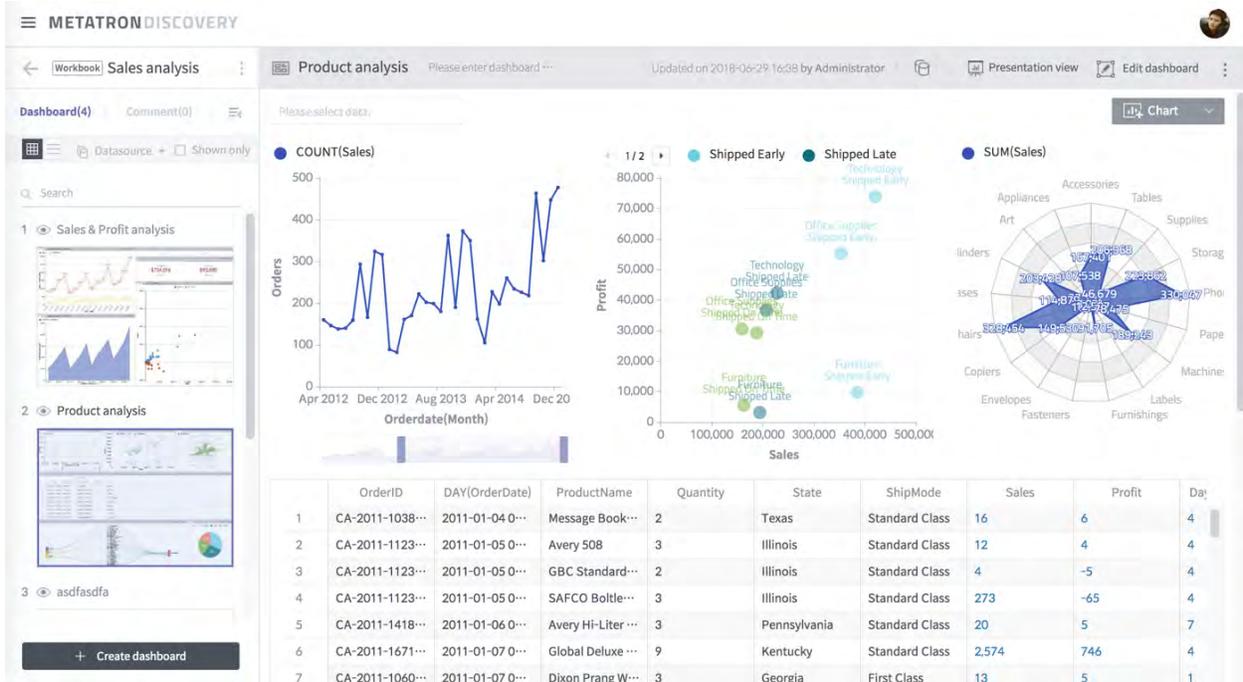
No.	Datasource	Type	Used in	Full size	Updated
16	The_2014_Inc_5000 Open data	Ingested type	Open data	1.19 MB	2018-07-10
17	EMSI_JobChange_UK Open data	Ingested type	Open data	46.73 KB	2018-07-10
18	OECD_TAX_ALL_02 Open data	Ingested type	Open data	926.70 KB	2018-07-09
19	WorldCup_Matches Open data	Ingested type	Open data	69.31 KB	2018-07-06
20	oecd_test Open data	Ingested type	Open data	30.61 KB	2018-07-06
21	tour de france Open data	Ingested type	Open data	27.94 KB	2018-07-06
22	cell_1h	Ingested type	2 Workspaces	90.79 MB	2018-07-06
23	FIFA_18_Player_Ratings Open data	Ingested type	Open data	3.41 MB	2018-07-06

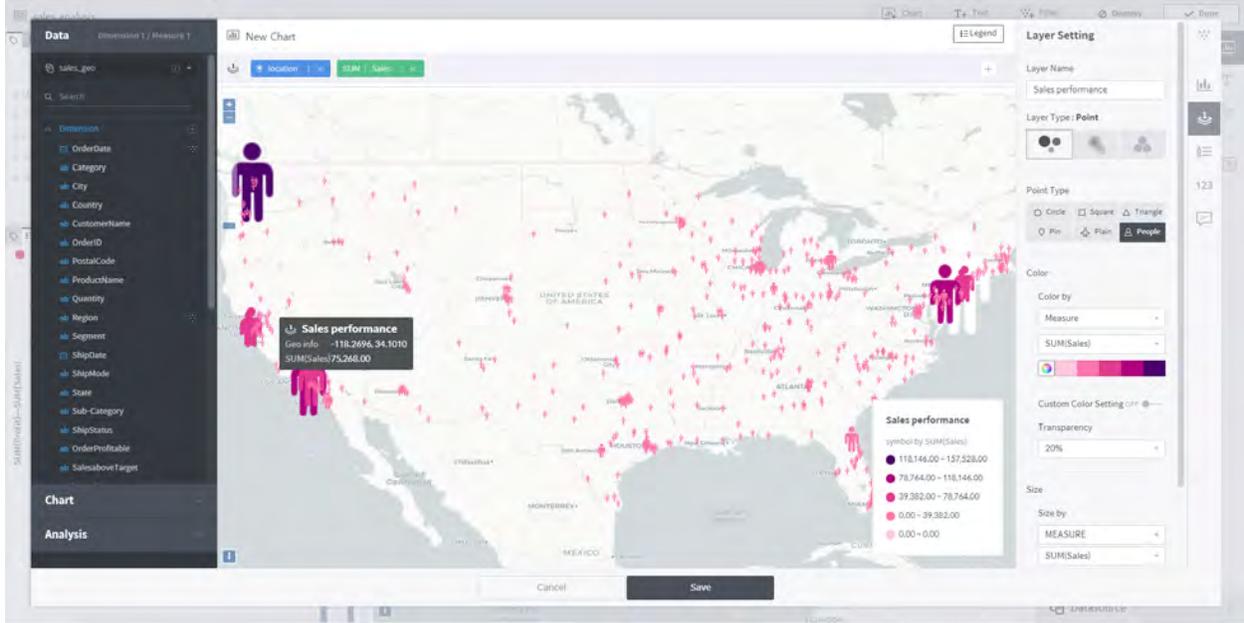
More +

Close

워크북, 대시보드, 차트

워크북은 그 안에 담긴 여러 대시보드와 차트를 PPT처럼 작업하고 이를 공유·프리젠테이션할 수 있게 해주는 모듈입니다. 워크북 기능에 대한 자세한 설명은 [워크북](#) 항목을 참조하십시오.





노트북

노트북에서는 Machine Learning 기반 고급 분석을 수행할 수 있습니다. 노트북 기능에 대한 자세한 설명은 [노트북 항목](#)을 참조하십시오.

```
// 1. load dataset
import app.metatron.discovery.connector._;
val conf = new MetisClientSetting();
conf.setting("host", "metatron-web-01").setting("port", "8080");
val client = new MetisClient(conf);
val dataset = client.loadData(spark, "datasources", "ds-gis-37", "1000")

// 2. analyze
dataset.show()
```

워크벤치

SQL 기반 분석을 수행할 수 있습니다. 워크벤치 기능에 대한 자세한 설명은 [워크벤치 항목](#)을 참조하십시오.

The screenshot displays the Metatron Discovery web interface. On the left, a sidebar shows a 'Table list' with various tables like 'contract', 'employee', and 'test'. The main area is split into two sections. The top section shows a Hive SQL query for '쿼리 01' with line numbers 1 through 18. The query includes a SELECT statement with columns A.C.ONE, A.C.TWO, and SUM(A.C.TEN) from TB_NUM AS A, filtered by A.C.ONE = 5 and grouped by A.C.ONE, A.C.TWO. It also includes a comment on the table 'USER_INFO_EX' and a SELECT statement from 'USER_TAB_COMMENTS' where TABLE_NAME = 'USER_INFO_EX'. The bottom section shows the query results for '쿼리 01 - 결과 1' in a table format with columns SEQ, L_orderkey, L_partkey, L_suppkey, L_linenumber, L_quantity, L_extendedprice, and L_discour. The results show four rows of data.

```

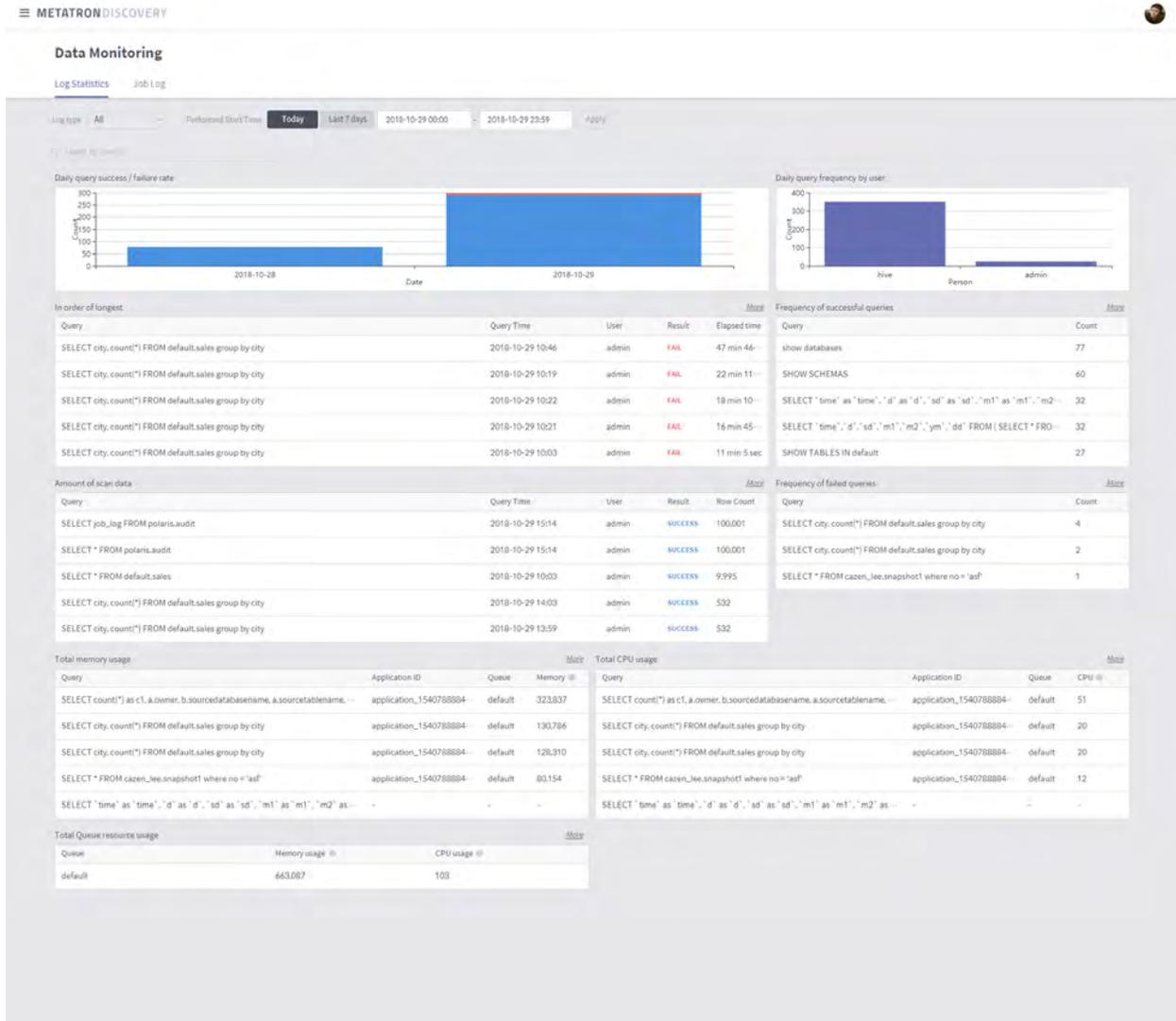
1 SELECT A.C.ONE,
2         A.C.TWO,
3         SUM(A.C.TEN)
4 FROM TB_NUM AS A
5 WHERE A.C.ONE = 5
6 GROUP BY A.C.ONE, A.C.TWO;
7
8 USING 'GROUP BY' QUERY EXAMPLE
9 COMMENT ON TABLE USER_INFO_EX
10      IS '고객 정보 확인'; -- USER_INFO_EX 테이블에 주석 추가
11
12
13 SELECT *
14 FROM USER_TAB_COMMENTS
15 WHERE TABLE_NAME = 'USER_INFO_EX'; -- USER_INFO_EX 테이블의 주석 확인
16
17 COMMENT ON COLUMN USER_INFO_EX.RNAME
18      IS '고객 실제 이름'; -- USER_INFO_EX 의 RNAME 컬럼에 주석 추가

```

SEQ	L_orderkey	L_partkey	L_suppkey	L_linenumber	L_quantity	L_extendedprice	L_discour
1	1	31037869	1537885	1	17.0	30690.27	0.04
2	1	13461816	1461817	2	36.0	63977.04	0.09
3	1	12739956	739957	3	8.0	15962.56	0.1
4	1	426299	926300	4	28.0	34307.56	0.09

2.2.4 데이터 모니터링

데이터 쿼리 통계, 쿼리 로그 감사를 통해 데이터 사용에 대한 모니터링 기능을 제공합니다. 데이터 모니터링 기능에 대한 자세한 설명은 [데이터 모니터링 항목](#)을 참조하십시오.



2.2.5 사용자 권한 및 계정

사용자를 추가, 삭제하거나 사용자의 권한을 관리합니다.

2.2.6 로그인/로그아웃

계정을 부여받은 사용자는 Metatron Discovery에 로그인하여 사용 권한에 맞게 자유롭게 이용할 수 있습니다. 로그인 중인 계정은 외부 시스템에서도 로그아웃이 가능합니다.

2.3 Metatron 기본 엔진: Druid

근래에 들어 정보통신 기술의 발전과 함께 데이터 발생량이 비약적으로 증가하였고, 이를 효율적으로 수집·관리·활용하는 것의 중요성이 대두되고 있습니다. 하지만 RDBMS 위주의 레거시 시스템들로는 대용량의 다차원 데이터를 온전히 처리할 수가 없기 때문에 차세대 빅데이터 수요를 충족시키기 위한 새로운 방법론과 솔루션들이 대거 등장하기 시작하였습니다.

미국 실리콘 벨리에 소재한 스타트업 Metamarkets사에서는 2011년 Druid라는 컬럼 기반 분산형 데이터 스토어를 출시하였고 2012년 10월 이를 오픈소스로 전환하였는데, 빠르고 효율적인 데이터 처리를 비롯한 여러 장점 때문에 많은 기업에서 Druid를 backend 기술로 활용하고 있습니다.

이러한 흐름에서 SK텔레콤도 B2C 이동통신 서비스 제공업체로서 매순간 이용자들로부터 발생하는 엄청난 양의 네트워크 데이터를 효과적으로 관리·분석할 필요가 있었으며, 2016년 Druid를 기본 엔진으로 활용한 end-to-end 비즈니스 인텔리전스 솔루션 Metatron을 개발·출시하였습니다.



본 단원에서는 시계열 데이터 처리에 적합한 Druid의 특징에 대해 알아본 후 SK텔레콤의 Metatron에서 이를 어떤 식으로 적용하고 추가 기능들을 개발하였는지 소개합니다.

2.3.1 Druid 개발 배경

Druid는 대량의 트랜잭션 이벤트 (로그 데이터) 를 ingestion하고 탐색할 수 있도록 지원하는 엔진으로서 다음과 같은 니즈를 충족하기 위해 개발되었습니다.

- 개발자들은 어떠한 차원들의 조합에 대한 쿼리라도 즉각적으로 결과가 반환되며, 데이터를 신속히 임의로 slice 및 dice하고 이를 아무 제약 없이 효과적으로 drill down하는 기능을 구현하고자 했습니다. 이러한 기능들은 사용자가 데이터 대시보드에서 이벤트 스트림들을 interactive한 방식으로 자유롭게 탐색하고 시각화하는 데 필요하였습니다.
- 개발자들은 이벤트들이 발생하는 즉시 해당 데이터를 ingestion하여 쿼리 가능하도록 인덱싱하는 기능을 원했습니다. 이러한 기능은 사용자가 데이터를 실시간으로 수집·분석함으로써 시의적절하게 상황을 판단·예측하고 비즈니스 의사 결정을 내릴 수 있도록 하는 데 반드시 필요했습니다. 당시에 Hadoop 등의 유명한 오픈소스 데이터 웨어하우스 시스템들은 sub-second 데이터 ingestion이 불가능했습니다.
- 개발자들은 multitenancy와 high availability를 보장하고자 했습니다. 이들이 제공하는 서비스의 기반 시스템은 많은 사용자들의 동시 접속을 보장하고, 항상 가동 상태를 유지하며 downtime 없이 모든 고장에 견딜 수 있어야 했습니다. downtime의 존재는 비용을 발생시키며, 소프트웨어 업그레이드나 네트워크 장애 발생 시 사용할 수 없는 시스템으로는 많은 기업의 비즈니스가 불가능합니다.

2.3.2 Druid 특징

데이터 테이블 형태

Druid의 데이터 테이블 (Druid에서는 <데이터 소스>라고 함) 은 OLAP 쿼리용으로 설계된 시계열 이벤트들로 구성됩니다. 데이터 소스는 세 종류의 컬럼으로 구성됩니다 (여기서는 온라인 광고 데이터를 예시로 사용).

Timestamp column	Dimension columns				Metric columns	
timestamp	publisher	advertiser	gender	country	click	price
2011-01-01T01:01:35Z	bieberfever.com	google.com	Male	USA	0	0.65
2011-01-01T01:03:63Z	bieberfever.com	google.com	Male	USA	0	0.62
2011-01-01T01:04:51Z	bieberfever.com	google.com	Male	USA	1	0.45
2011-01-01T01:00:00Z	ultratriumfast.com	google.com	Female	UK	0	0.87
2011-01-01T02:00:00Z	ultratriumfast.com	google.com	Female	UK	0	0.99
2011-01-01T02:00:00Z	ultratriumfast.com	google.com	Female	UK	1	1.53

그림 1: Source: <http://druid.io>

- **타임스탬프 컬럼 (Timestamp column):** Druid는 데이터 소스에서 타임스탬프 컬럼을 별도로 구성함으로써 모든 쿼리가 시간 축을 중심으로 이루어지게 합니다 (시계열 속성이 없는 데이터를 일괄적으로 ingestion할 경우에는 현재 시간을 기준으로 타임스탬프가 부여되어 Druid에서 활용할 수 있는 형태가 됩니다).
- **차원 컬럼 (Dimension columns):** 차원 컬럼은 각 이벤트의 문자열 속성들을 담고 있으며, 데이터 필터링 시 가장 흔히 사용됩니다. 위 데이터셋 예시에서는 publisher, advertiser, gender, country가 차원 컬럼입니다. 데이터 탐색 시에는 이러한 차원 컬럼들을 축으로 하여 데이터를 slice하게 됩니다.
- **측정값 컬럼 (Metric columns):** 측정값 컬럼들은 집계 및 연산에 사용됩니다. 위 예에서는 clicks 및 price가 측정값 컬럼입니다. 측정값 컬럼의 자료형은 대체로 수치 값이며, 이들은 계수, 합산, 평균 등의 방식으로 집계할 수 있습니다 (Metatron에서는 지원되는 자료형을 증대하였습니다).

데이터 ingestion

Druid는 실시간 및 일괄 (batch) ingestion을 지원합니다.

이 중에서 실시간 ingestion은 Druid의 주요 특징 중 하나인데, 이를 전담하는 real-time 노드군이 있기 때문에 가능한 것입니다 (자세한 설명은 [Real-time 노드](#) 참조). 실시간으로 ingestion되는 데이터 스트림 내 이벤트들은 발생 후 수 초 이내에 Druid 클러스터에서 쿼리가 가능한 포맷으로 인덱싱됩니다.

데이터 roll-up

무수히 많은 개별 이벤트를 단순히 열거하기만 해서는 중요한 의미를 찾을 수 없습니다. 하지만 이러한 데이터를 적절한 시간대를 기준으로 취합하면 유용한 인사이트를 얻을 수 있습니다. Druid는 roll-up이라는 옵션을 통해 ingestion되는 원천 데이터를 취합할 수 있습니다. 아래는 roll-up의 예시를 나타낸 것입니다.

timestamp	domain	gender	clicked
2011-01-01T00:01:35Z	bieber.com	Female	1
2011-01-01T00:03:03Z	bieber.com	Female	0
2011-01-01T00:04:51Z	ultra.com	Male	1
2011-01-01T00:05:33Z	ultra.com	Male	1
2011-01-01T00:05:53Z	ultra.com	Female	0
2011-01-01T00:06:17Z	ultra.com	Female	1
2011-01-01T00:23:15Z	bieber.com	Female	0
2011-01-01T00:38:51Z	ultra.com	Male	1
2011-01-01T00:49:33Z	bieber.com	Female	1
2011-01-01T00:49:53Z	ultra.com	Female	0

→

timestamp	domain	gender	clicked
2011-01-01T00:00:00Z	bieber.com	Female	1
2011-01-01T00:00:00Z	ultra.com	Female	2
2011-01-01T00:00:00Z	ultra.com	Male	3

그림 2: Source: Interactive Exploratory Analytics with Druid | DataEngConf SF <17

왼쪽의 원본 이벤트 목록은 2011년 1월 1일 00:00:00~01:00:00 사이에 발생한 도메인 클릭 이벤트를 열거한 것입니다. 하지만 분석가 입장에서는 분 이하 단위의 개별 이벤트가 별다른 의미를 갖지 못하기 때문에 1시간의 granularity를 기준으로 데이터를 취합했습니다. 그 결과 오른쪽 테이블에 나타난 것처럼 2011년 1월 1일 00~01시 시간대에 각 도메인을 남성과 여성이 각각 클릭한 횟수를 보여주는 보다 의미 있는 결과물이 도출되었습니다.

또한 데이터 roll-up은 원천 데이터의 저장 용량을 최소화함으로써 (많은 것은 100배까지도 축소 가능), 스토리지 리소스를 절약하고 쿼리 속도를 빠르게 합니다.

그러나 데이터를 roll-up하면 개별 이벤트들에 대해 쿼리할 수 없게 됩니다. roll-up granularity는 데이터를 탐색할 수 있는 최소 단위가 되며 이벤트들은 이러한 granularity 단위로 배열됩니다. granularity 단위는 사용자가 원하는 대로 설정할 수 있으며, 원치 않으면 roll-up을 비활성화하고 모든 개별 이벤트를 전부 ingestion할 수도 있습니다.

데이터 sharding

데이터 소스는 시계열 이벤트들의 집합으로서 여러 shard로 분할 저장되는데, Druid에서는 이를 <세그먼트>라고 부르며 각 세그먼트는 대개 500~1,000만 행으로 이루어집니다. Druid는 데이터 소스들을 정의된 시간 간격 (통상적으로 1시간이나 하루)을 기준으로 분할하며, 그 밖의 컬럼에 있는 값들을 기준으로 추가 분할을 실시함으로써 세그먼트 크기를 적절하게 맞출 수 있습니다.

아래는 1시간 단위로 분할된 데이터 테이블을 예시로 나타낸 것입니다.

세그먼트 sampleData_2011-01-01T01:00:00:00Z_2011-01-01T02:00:00:00Z_v1_0:

2011-01-01T01:00:00Z	ultratrimfast.com	google.com	Male	USA	1800	25	15.70
2011-01-01T01:00:00Z	bieberfever.com	google.com	Male	USA	2912	42	29.18

세그먼트 sampleData_2011-01-01T02:00:00:00Z_2011-01-01T03:00:00:00Z_v1_0:

2011-01-01T02:00:00Z	ultratrimfast.com	google.com	Male	UK	1953	17	17.31
2011-01-01T02:00:00Z	bieberfever.com	google.com	Male	UK	3194	170	34.01

이와 같이 시간 단위로 세그먼트를 구분하는 것은 데이터 소스 내의 모든 이벤트에 타임스탬프가 포함되기 때문에 가능합니다.

세그먼트는 Druid 테이블의 기본 저장 단위에 해당하며, 클러스터 내 데이터의 복제 (replication) 및 분산은 세그먼트 단위로 이루어집니다. 세그먼트 내 데이터는 변경할 수 없도록 되어 있습니다. 이렇게 함으로써 읽기와 쓰기 동작 사이에 경합이 발생하지 않게 됩니다. Druid의 세그먼트는 매우 신속하게 읽히기 위한 읽기 전용 데이터셋입니다.

뿐만 아니라, 이러한 데이터 세그먼트 분할은 Druid 분산 환경에서의 병렬 처리를 위한 핵심 역할을 합니다. 각 CPU가 한번에 하나의 세그먼트를 스캔할 수 있기 때문에 데이터를 여러 세그먼트로 분할하면 이를 여러 CPU가 동시에 병렬적으로 스캔할 수 있으므로, 쿼리 결과를 신속하게 반환하고 부하를 안정적으로 분산시킬 수 있게 됩니다.

데이터 저장 포맷 및 인덱싱

Druid의 데이터 구조를 분석 쿼리에 최적화시키는 주요 요소 중 하나는 Druid가 데이터를 저장하는 방식입니다. 본 절에서는 설명을 위해 아래의 Druid 테이블 예시를 사용합니다.

Timestamp	Page	Username	Gender	City	Characters Added	Characters Removed
2011-01-01T01:00:00Z	Justin Bieber	Boxer	Male	San Francisco	1800	25
2011-01-01T01:00:00Z	Justin Bieber	Reach	Male	Waterloo	2912	42
2011-01-01T02:00:00Z	Ke\$ha	Helz	Male	Calgary	1953	17
2011-01-01T02:00:00Z	Ke\$ha	Xeno	Male	Taiyuan	3194	170

그림 3: Source: Druid: A Real-time Analytical Data Store

컬럼 기반 저장 및 인덱싱

Druid는 컬럼들을 각각 따로 저장합니다. Druid가 주로 이벤트 스트림을 집계하는 데 사용된다는 점을 고려할 때, 이러한 컬럼 기반 저장 방식을 취하면 각 쿼리에 관련된 컬럼만을 로드 · 스캔하므로 CPU 리소스를 보다 효율적으로 사용할 수 있습니다. 행 기반 데이터 스토어에서는 집계 시 대상 행과 관련된 모든 컬럼을 컬럼별로 상이한 방식으로 압축할 수 있으며 그에 따라 각기 다른 인덱스를 활용함으로써 컬럼을 메모리나 디스크에 저장하는 데 드는 리소스 비용을 줄일 수 있습니다. 위 예에서 page, user, gender, city 컬럼은 문자열만을 포함합니다. 직접 문자열들을 저장하는 것은 불필요한 비용을 발생시키므로 이들을 고유한 정수 식별자로 매핑할 수 있습니다. 예를 들면,

```
Justin Bieber -> 0
Ke$ha -> 1
```

이 매핑을 사용하면 page 컬럼을 정수 배열로 나타낼 수 있는데, 여기서 배열 인덱스 각각은 원본 데이터셋의 각 행에 해당합니다. page 컬럼의 경우, 각 행의 page 값을 아래와 같이 표시할 수 있습니다.

```
[0, 0, 1, 1]
```

이처럼 문자열들이 고정 길이 정수들로 바뀌어 저장되므로 압축하기가 훨씬 더 수월합니다. Druid는 각 shard(세그먼트) 단위로 데이터를 인덱싱합니다.

데이터 필터링을 위한 인덱싱

Druid는 검색 인덱스를 추가로 만들어서 문자열 컬럼에 대한 필터링을 용이하게 할 수 있습니다. 위 예시 테이블을 다시 보자. 가령 《샌프란시스코에 사는 남성 사용자들이 Wikipedia 편집을 한 횟수는?》과 같은 쿼리가 있을 수 있습니다. 이 쿼리 예시에는 도시(San Francisco)와 성별(Male)이라는 두 가지 차원이 포함됩니다. 각 차원별로 아래와 같은 바이너리 배열이 생성되는데, 여기서 배열 인덱스 각각은 해당 행이 쿼리 필터 조건에 부합하는지 여부를 나타냅니다.

```
San Francisco (City) -> rows [1] -> [1][0][0][0]
Male (Gender) -> rows [1, 2, 3, 4] -> [1][1][1][1]
```

그런 다음 쿼리 필터는 이러한 두 배열에 대해 AND 연산을 실시합니다.

```
[1][0][0][0] AND [1][1][1][1] = [1][0][0][0]
```

그 결과, 행 1만 스캔 대상이 됩니다. 이런 식으로 필터링된 행만 검색함으로써 불필요한 부하를 방지하는 것입니다. 이러한 바이너리 배열은 압축하기도 매우 쉽습니다.

이러한 검색 인덱스는 OR 연산에도 사용할 수 있습니다. 어떤 쿼리가 San Francisco 또는 Calgary를 필터링하는 경우, 배열 인덱스들은 차원값별로 다음과 같을 것입니다.

```
San Francisco (City) -> rows [1] -> [1][0][0][0]
Calgary (City) -> rows [3] -> [0][0][1][0]
```

그런 다음 두 배열에 대해 OR 연산이 수행됩니다.

```
[1][0][0][0] OR [0][0][1][0] = [1][0][1][0]
```

그 결과, 쿼리는 행 1과 3만 스캔합니다.

대형 비트맵 셋에 boolean 연산을 실시하는 이러한 접근방식은 검색 엔진에서 널리 사용됩니다.

쿼리 언어

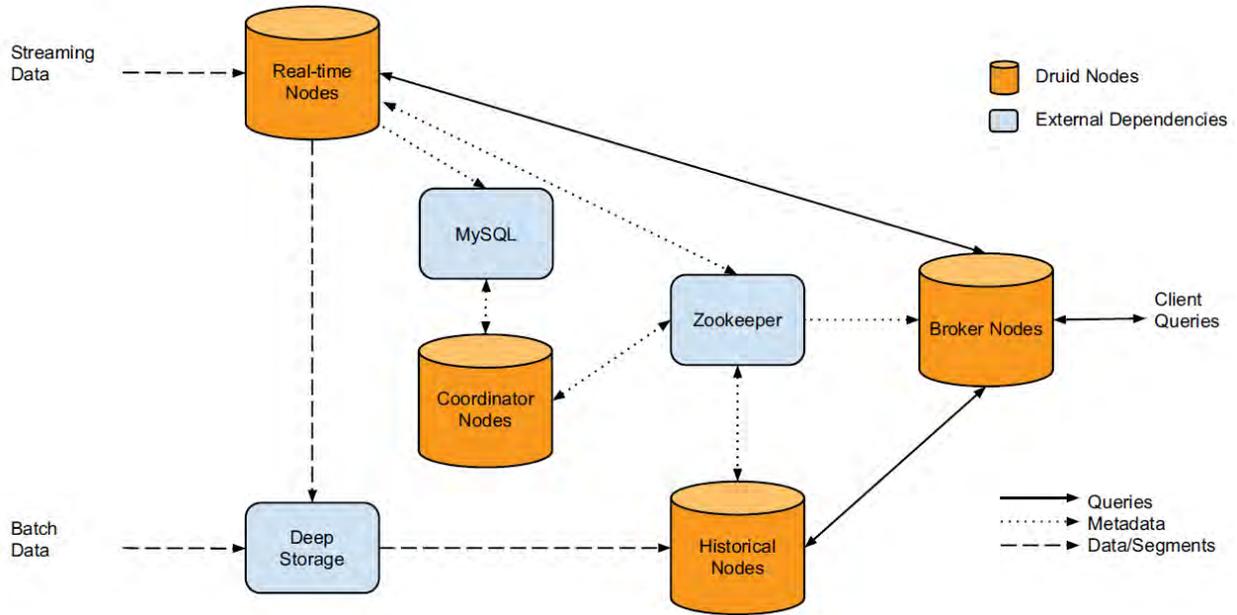
Druid의 네이티브 쿼리 언어는 JSON over HTTP이며, 주요 쿼리는 다음과 같습니다.

- Group By
- 시계열 기반 roll-up
- 임의적 boolean 필터링
- Sum, Min, Max, Avg 등의 집계 연산
- 차원값 검색

하지만 이 외에도 SQL을 비롯한 다양한 언어로 이루어진 쿼리 라이브러리가 생성·공유되고 있습니다.

2.3.3 Druid 기본 클러스터 아키텍처

Druid 클러스터는 여러 유형의 노드군으로 구성되며, 각 유형의 노드군별로 고유의 역할을 수행합니다.



Real-time 노트

real-time 노트군은 이벤트 스트림을 ingestion하고 쿼리하는 기능을 합니다. 이 노트들은 최근 발생한 짧은 시간 범위 내 이벤트들만을 처리하며, 주기적으로 이들을 딥 스토리지로 넘기는데, 그 절차는 다음과 같습니다.

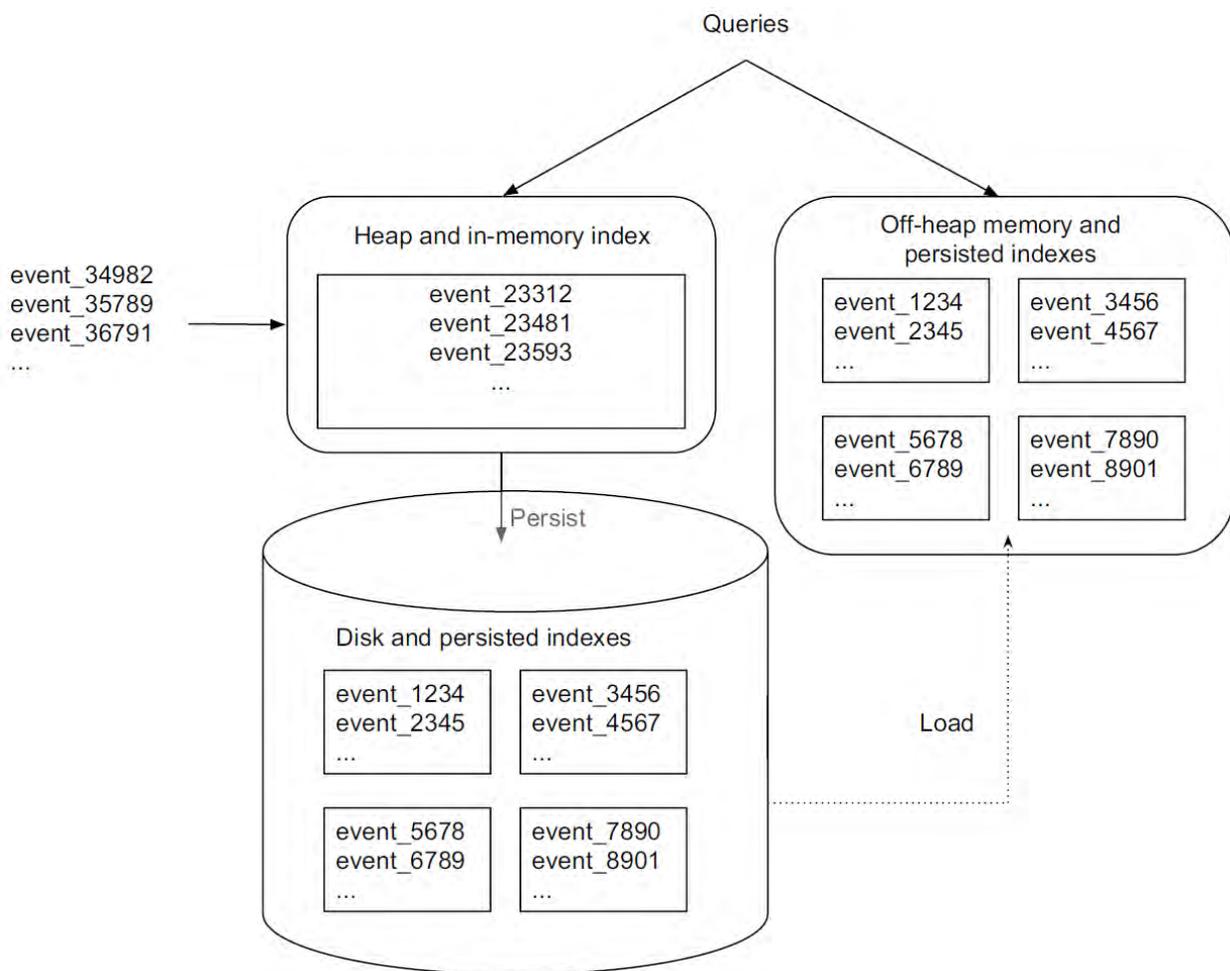


그림 4: Source: Druid: A Real-time Analytical Data Store

1. 유입되는 이벤트들은 메모리에 인덱싱되면서 즉시 쿼리에 사용될 수 있습니다.
2. 메모리 상의 데이터는 정기적으로 디스크에 저장되면서 수정 불가능한 (읽기 전용) 컬럼형 포맷으로 변환됩니다.
3. 저장된 데이터는 off-heap 메모리로 로드되기 때문에 쿼리 가능한 상태가 유지됩니다.
4. 디스크에 저장된 인덱스들을 주기적으로 병합되어 데이터 <세그먼트>를 구성한 후 딥 스토리지로 이관됩니다.

이런 식으로 real-time 노드로 ingestion된 모든 이벤트는 디스크 저장 전후를 막론하고 on-heap 또는 off-heap 메모리 상에 존재하므로 쿼리가 가능한 상태를 유지합니다 (쿼리는 메모리 상의 인덱스와 디스크에 저장된 인덱스 모두에 전달됩니다). 이러한 real-time 노드 기능을 통해 Druid는 실시간 데이터 ingestion을 수행할 수 있습니다. 즉, 이벤트들이 발생하면 곧 이어서 쿼리 대상이 됩니다. 그리고 이러한 과정에서 데이터 손실이 발생하지 않습니다.

real-time 노드는 Druid 클러스터 내 다른 노드들과의 유기적인 동작을 위해 자신의 온라인 상태와 처리 중인 데이터를

Zookeeper(외부 종속 모듈 참조)에 보고합니다.

Historical 노드

historical 노드군은 real-time 노드가 생성한 읽기 전용 데이터 블록(세그먼트)을 로드하고 처리하는 기능을 합니다. 이 노드들은 딥 스토리지에서 읽기 전용 세그먼트를 다운로드하고 이에 대한 쿼리를 처리합니다(예: 데이터 집계/필터링). 이 노드들은 shared-nothing 아키텍처에 기반하며 동작이 단순합니다. 이들 간에는 경합이 발생하지 않으며 단순히 Zookeeper의 지시에 따라 세그먼트를 로드, 드롭, 처리할 뿐입니다.

historical 노드가 쿼리를 처리하는 프로세스는 다음과 같습니다.

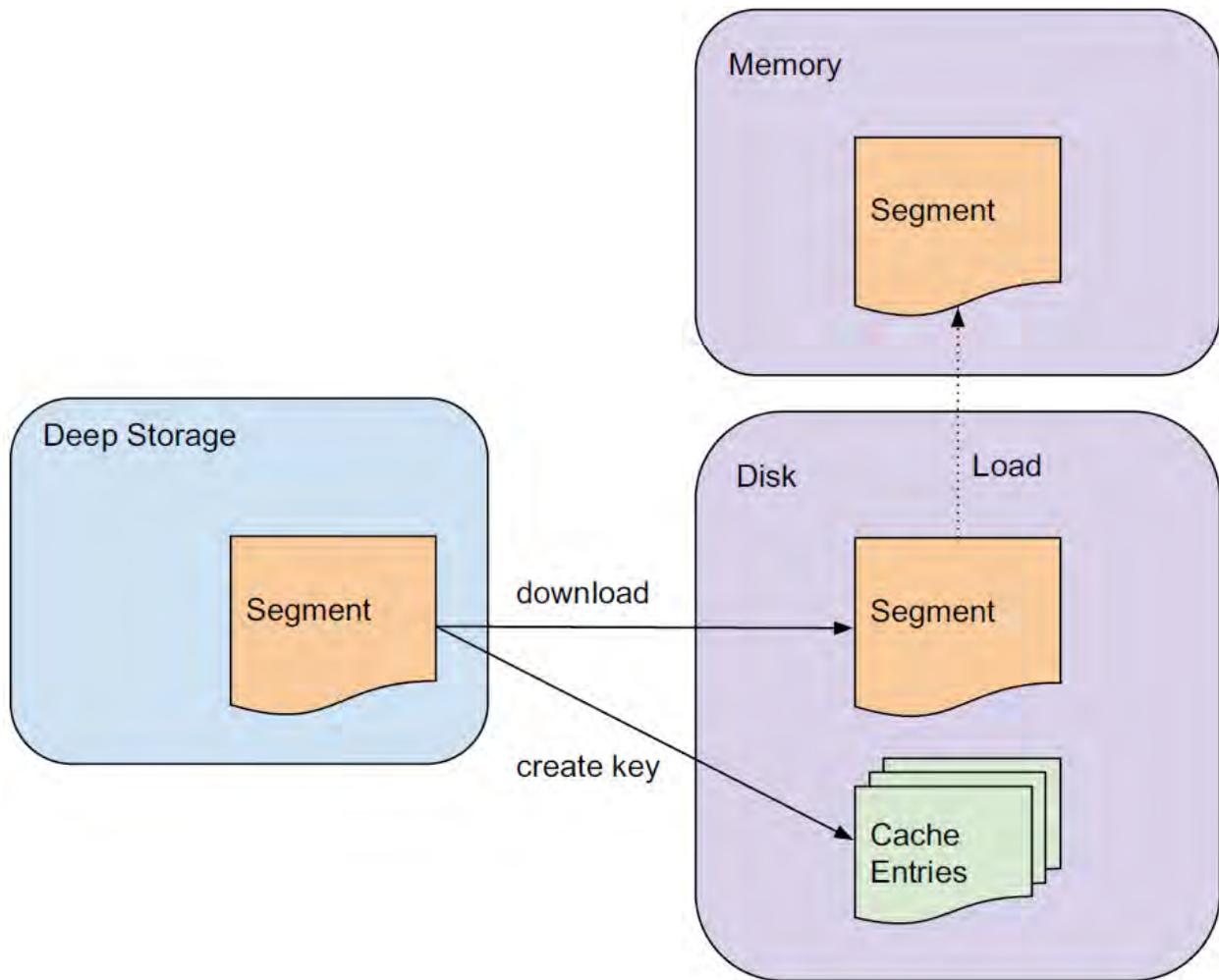


그림 5: Source: Druid: A Real-time Analytical Data Store

쿼리를 받으면 historical 노드는 우선 자신에게 이미 어떤 세그먼트가 존재하는지에 관한 정보를 보관하는 로컬 캐시를

확인합니다. 어떤 세그먼트에 관한 정보가 캐시에 없으면 노드는 딥 스토리지에서 해당 세그먼트를 다운로드합니다. 그런 다음, 해당 세그먼트는 Zookeeper에서 선언되어 쿼리가 가능한 대상이 되며, 노드는 이 세그먼트에 대해 요청된 쿼리를 수행합니다.

historical 노드는 읽기 전용 데이터만을 다루므로 read consistency를 보장할 수 있습니다. 읽기 전용 데이터 블록들은 또한 단순한 병렬 모델을 가능케 합니다. 즉, historical 노드들은 읽기 전용 데이터 블록들을 서로 간섭하지 않고 동시에 스캔·집계할 수 있습니다.

real-time 노드와 마찬가지로 historical 노드들도 자신들의 온라인 상태와 처리 중인 데이터를 Zookeeper에 보고합니다.

Broker 노드

broker 노드군은 Zookeeper에 보고된 메타데이터를 통해 어떤 세그먼트들이 쿼리 가능한지와 이 세그먼트들이 각각 어디에 저장되어 있는지를 파악합니다. broker 노드들은 입력된 쿼리들의 경로를 지정함으로써 각 쿼리가 올바른 historical 또는 real-time 노드에 도달되게끔 합니다. 그런 다음 historical 및 real-time 노드 각각에서 산출된 결과들을 취합하여 최종 쿼리 결과를 호출자에게 반환합니다.

broker 노드는 리소스 효율성을 높이기 위해 다음과 같이 캐시를 사용합니다.

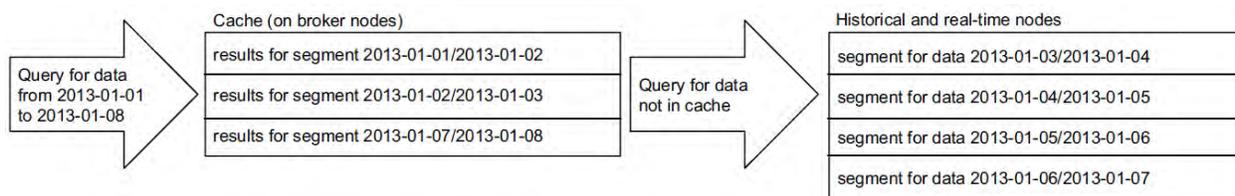


그림 6: Source: Druid: A Real-time Analytical Data Store

어떤 쿼리가 여러 세그먼트를 포괄할 경우 broker 노드는 캐시에 이미 존재하는 세그먼트들을 우선 확인합니다. 그리고 캐시에 없는 세그먼트들에 대해서는 그것이 보관된 historical 및 real-time 노드로 쿼리를 전달합니다. historical 노드들이 결과를 반환하면, broker 노드는 이 결과를 나중에 사용할 수 있도록 세그먼트별로 캐시에 저장합니다. real-time 노드의 데이터는 캐시에 저장되지 않으며, 따라서 real-time 데이터에 대한 요청은 항상 real-time 노드로 전달됩니다. real-time 노드의 데이터는 가변적이기 때문에 그 결과를 캐시에 저장하는 것은 안정적이지 않기 때문입니다.

Coordinator 노드

coordinator 노드군은 주로 historical 노드 데이터의 관리 및 분산을 담당합니다. coordinator 노드는 어떤 historical 노드가 어떤 세그먼트에 대해 쿼리를 수행할지 결정하고 이들에게 새 데이터를 로드하고, 기한이 지난 데이터를 드롭하고,

데이터를 복제하고, 데이터를 이동하여 부하 밸런스를 맞추도록 지시합니다. 이렇게 함으로써 분산형 historical 노드 그룹에서 빠르고 효율적이며 안정으로 데이터를 처리할 수 있습니다.

다른 모든 Druid 노드와 마찬가지로, coordinator 노드들도 Zookeeper 연결을 유지함으로써 클러스터의 현황을 파악합니다. coordinator 노드들은 MySQL 데이터베이스와의 연결도 유지하는데, 이 데이터베이스에서는 클러스터 내 세그먼트의 생성, 소멸, 복제 규칙과 같은 추가적인 연산 매개변수 및 구성 정보를 관리합니다.

Druid 클러스터의 안정성을 위해 coordinator 노드는 이중화되며 일반적으로 하나의 coordinator 노드만 활성 상태를 유지합니다.

외부 종속 모듈

Druid는 클러스터 동작을 위해 몇 가지 외부 종속 모듈을 사용합니다.

- **Zookeeper:** Druid는 Zookeeper를 통해 클러스터 내부 통신을 합니다.
- **메타데이터 스토리지:** Druid는 메타데이터 스토리지를 통해 데이터 세그먼트 및 구성에 관한 메타데이터를 저장합니다. 메타데이터 스토리지로는 주로 MySQL과 PostgreSQL이 사용됩니다.
- **딥 스토리지:** Druid 세그먼트들을 영구적으로 백업 저장하는 공간입니다. Druid에 ingestion되는 데이터는 세그먼트 형태로 딥 스토리지에 업로드되고, historical 노드들이 필요한 세그먼트를 여기서 다운로드합니다. 딥 스토리지로는 주로 S3 및 HDFS가 사용됩니다.

High Availability 특성

Druid는 어느 한 노드가 고장난다고 해서 클러스터의 동작이 중단되지 않도록 설계되었습니다. 또한 서로 다른 유형의 노드군끼리도 상호 간에 상당히 독립적이기 때문에, 클러스터 내부에 통신 장애가 생겨도 데이터 가용성에는 최소한의 영향만을 미칩니다. Druid 클러스터에서 highly availability를 확보하려면, 노드군별로 2개 이상의 노드가 구성되어야 합니다.

아키텍처 확장성

Druid는 위에서 소개한 기본 아키텍처에 다양한 외부 모듈을 추가할 수 있는 모듈 확장형 플랫폼을 지향합니다. 아래는 Druid의 확장성을 활용한 모듈 조합의 예시입니다.

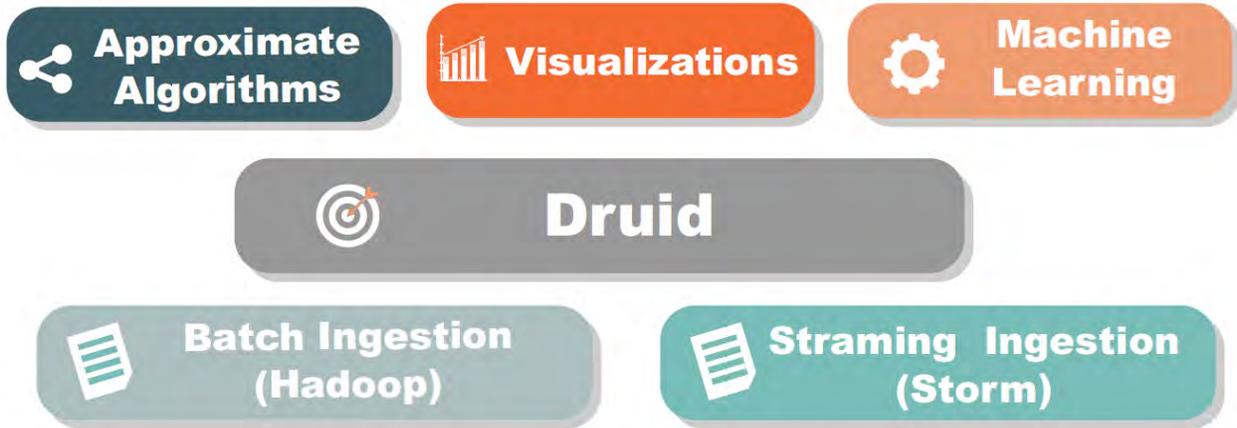


그림 7: Source: MetaMarkets - Introduction to Druid by Fangjin Yang

Metatron Discovery 엔진도 역시 비즈니스 인텔리전스를 위한 end-to-end 솔루션으로서 기능하기 위해 Druid 엔진 전후단에 다양한 모듈을 추가한 것입니다.

2.3.4 Druid 성능 평가

Druid는 <실시간> 탐색이 가능한 데이터 스토어를 지향하는 만큼 수치화된 성능을 평가함에 있어서는 다음의 두 가지 측면에 초점이 맞춰집니다.

- Query latency
- Ingestion latency

쿼리 처리와 ingestion에서 소요되는 시간을 최소화하는 것이 <실시간>을 이루는 핵심이 되기 때문입니다. 지금까지 Druid 개발진을 비롯한 여러 기관 및 개인이 이러한 기준으로 Druid 성능을 평가한 benchmark들을 산출하고 그 밖의 지표를 통해 Druid를 다른 데이터베이스 관리 시스템들과 비교한 결과를 공개하였습니다.

Druid 개발진의 자체 평가

Druid 개발진이 2014년 발표한 백서 <Druid: A Real-time Analytical Data Store>¹의 Chapter 6 Performance 에서는 Druid의 query 및 ingestion latency를 다방면에서 평가한 결과를 상세하게 설명하고 있습니다. 본 절에서는 이 중에서 Druid의 성능을 직관적으로 살펴볼 수 있는 지표 위주로 간단히 소개합니다.

¹

F. Yang, E. Tschetter, X. Laut, N. Ray, G. Merlino, and D. Ganguli. (2014). Druid: a real-time analytical data store. Retrieved from <http://druid.io/docs/0.12.1/design/index.html>.

Query latency 성능

Druid의 query latency 성능에 대해 백서에서는 현장에서 실제 사용되는 데이터셋 8종과 TPC-H 데이터셋에 대한 쿼리 결과를 기준으로 평가하였는데, 여기서는 TPC-H 데이터셋에 대한 쿼리 결과를 소개합니다. TPC-H 데이터셋에 대한 query latency는 MySQL과의 비교 평가 방식으로 진행하였고, 이때 사용한 클러스터 사양은 다음과 같았습니다.

- **Druid historical 노드:** Amazon EC2 m3.2xlarge instance types (Intel® Xeon® E5-2680 v2 @ 2.80GHz)
- **Druid broker 노드:** c3.2xlarge instances (Intel® Xeon® E5-2670 v2 @ 2.50GHz)
- **MySQL Amazon RDS instance** (Druid와 동일한 m3.2xlarge instance type)

아래는 단일 노드에서의 1GB 및 100GB TPC-H 데이터셋에 대한 Druid와 MySQL의 query latency를 비교한 결과를 정리한 그래프입니다.

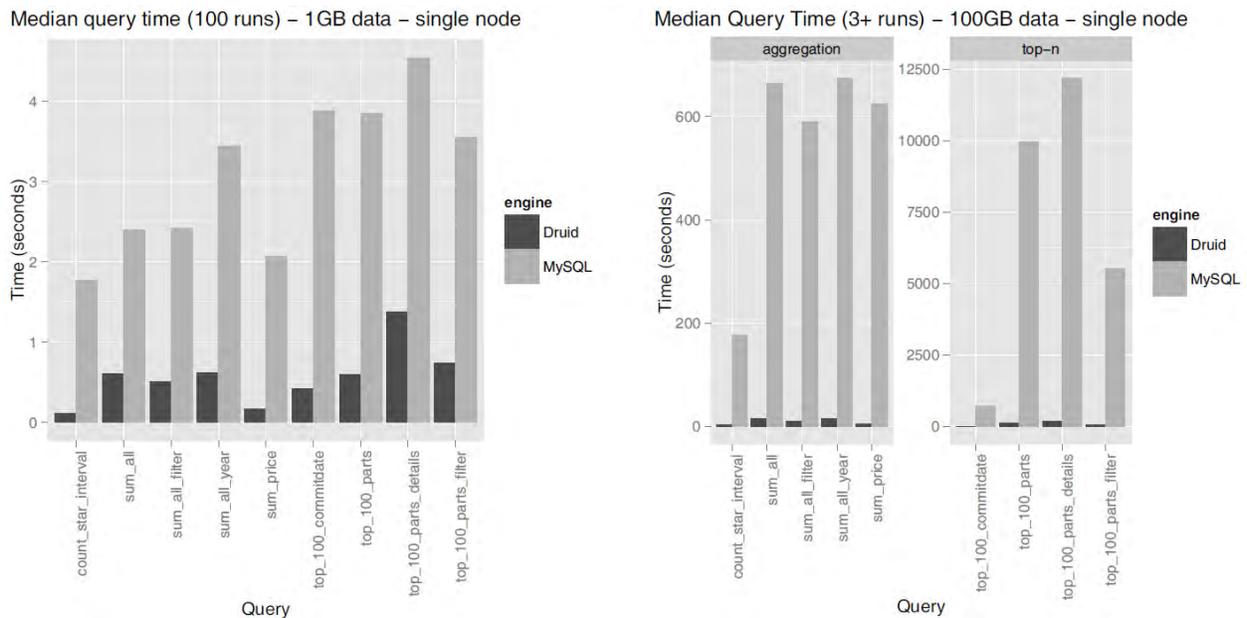


그림 8: Source: Druid: A Real-time Analytical Data Store

이러한 결과는 Druid의 도입으로 기존 관계형 데이터베이스 시스템에 비해 획기적으로 빠른 쿼리 속도를 낼 수 있음을 시사합니다.

또한 여러 노드를 엮어서 클러스터를 구성할 경우 쿼리 처리 속도가 어느 정도 향상되는지도 측정하였습니다. 쿼리 대상 데이터셋으로서 100GB TPC-H를 사용하였으며 단일 노드 (8개 코어)와 6개 노드 클러스터 (48개 코어) 간의 성능 차이는 다음과 같았습니다.

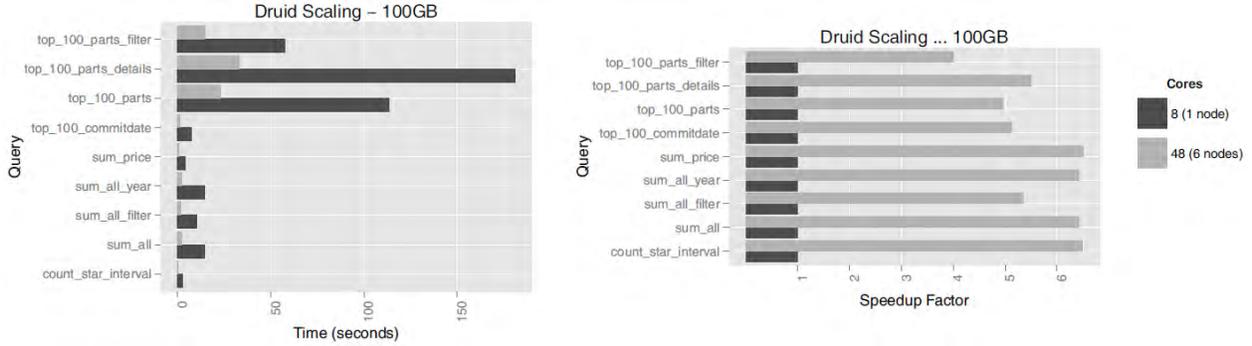


그림 9: Source: Druid: A Real-time Analytical Data Store

모든 쿼리가 linear scalability를 달성하지는 않았으나 상대적으로 단순한 쿼리들의 경우에는 거의 코어 수에 정비례하는 처리 속도 증대를 보여주었습니다 (SK텔레콤 Metatron에서는 더욱 뚜렷한 linear scalability를 달성할 수 있도록 기능을 보강하였습니다).

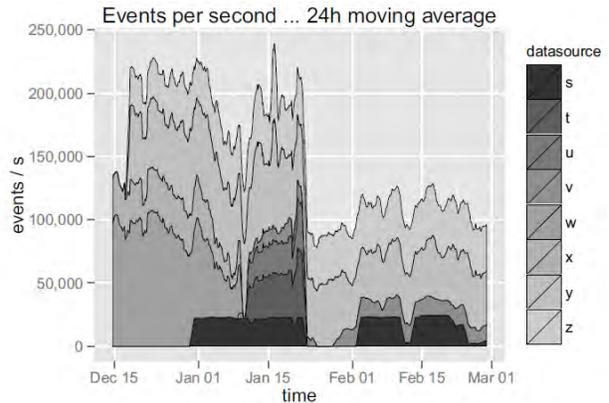
Ingestion latency 성능

Druid의 ingestion 성능에 대해서도 평가하였는데, 이때 사용된 클러스터 환경은 다음과 같았습니다.

- 6개 노드, 총 메모리 360GB 및 96개 코어 (12 x Intel® Xeon® E5-2670)

ingestion 대상으로는 현장에서 실제 사용되는 데이터 소스 8종이었으며 데이터 소스 각각의 특징과 ingestion 결과는 아래와 같았습니다. 참고로 ingestion 측정을 하는 기간 동안 해당 클러스터에서는 그 외 다른 데이터 소스에 대한 ingestion 동작도 병행해서 실시하였습니다.

Data Source	Dimensions	Metrics	Peak events/s
s	7	2	28334.60
t	10	7	68808.70
u	5	1	49933.93
v	30	10	22240.45
w	35	14	135763.17
x	28	6	46525.85
y	33	24	162462.41
z	33	24	95747.74



Ingestion characteristics of various data sources

Combined cluster ingestion rates

그림 10: Source: Druid: A Real-time Analytical Data Store

데이터 ingestion 속도는 데이터의 복잡성 등 여러 가지 변수의 영향을 받지만, 측정 결과를 놓고 볼 때 대체로 <interactivity>라는 Druid의 개발 목표에 부합한다고 할 수 있습니다.

SK텔레콤의 Druid 성능 평가

SK텔레콤에서는 다음과 같이 Druid의 query latency와 ingestion latency를 측정하였습니다.

Query latency 테스트

Query latency를 측정하는 조건은 다음과 같았습니다.

- 데이터: TPC-H 100G dataset (9억 rows)
- Pre-aggregation 기준: day
- 서버: r3.4xlarge nodes, (2.5GHz * 16, 122G, 320G SSD) * 6
- Historical 노드 개수: 6개
- Broker 노드 개수: 1개

그 결과 TPC-H 100G dataset의 5개 쿼리의 반환 속도는 다음과 같았습니다 (Hive의 쿼리 처리 속도도 참조용으로 함께 측정하였습니다).

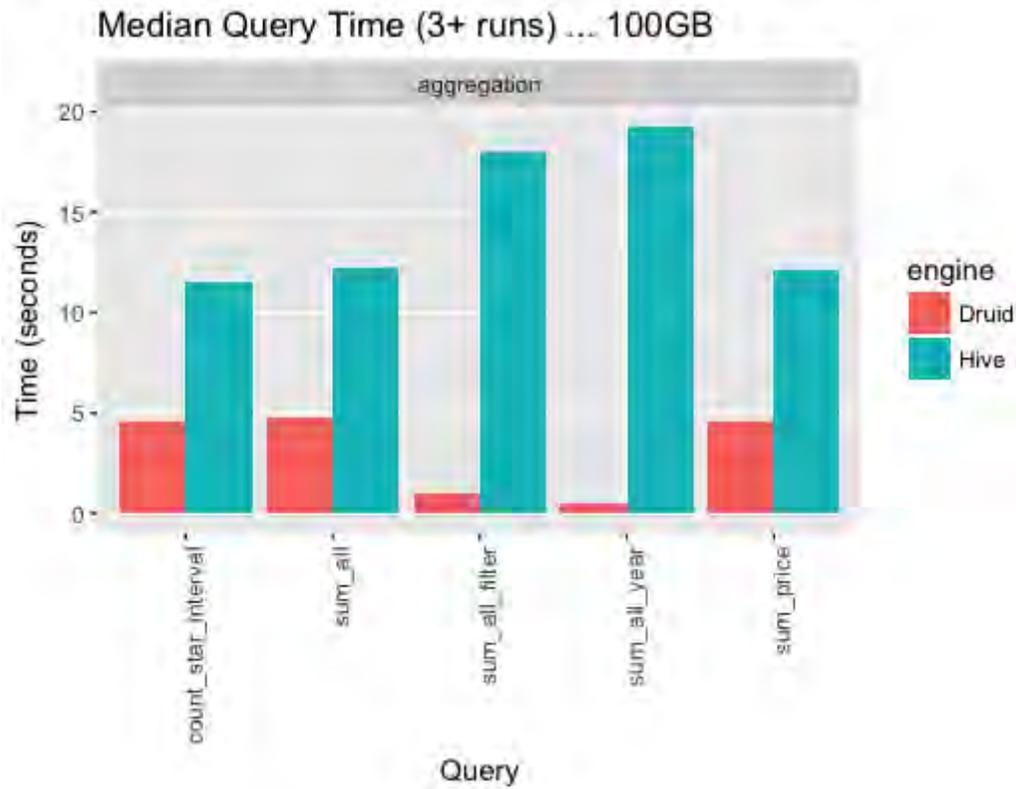


그림 11: Source: SK Telecom T-DE WIKI Metatron Project

참고: Hive의 benchmark가 현저하게 떨어지는 원인 중 일부는 Thrift로 측정된 것과 partition없이 test set이 구성되어 있기 때문입니다.

Ingestion latency 테스트

Ingestion latency를 측정하는 조건은 다음과 같았습니다.

- Ingestion data size: 1일 30억 rows, 10 columns
- 메모리: 512 GB
- CPU: Intel (R) Xeon (R) Gold 5120 CPU @ 2.20 GHz (core 56개)
- Historical 노드 개수: 100개
- Broker 노드 개수: 2개

- 총 10개의 middle manager 노드 중 3개에서 job 수행
- Ingestion 도구: Apache Kafka

이와 같은 조건으로 data ingestion을 100회 수행하였고 평균 ingestion latency는 1.623439초였습니다. 여기서 ingestion latency는 아래 도식화한 것과 같이 Kaka ingestion, Druid ingestion, Druid query 처리에 소요되는 시간을 모두 합산한 것입니다.

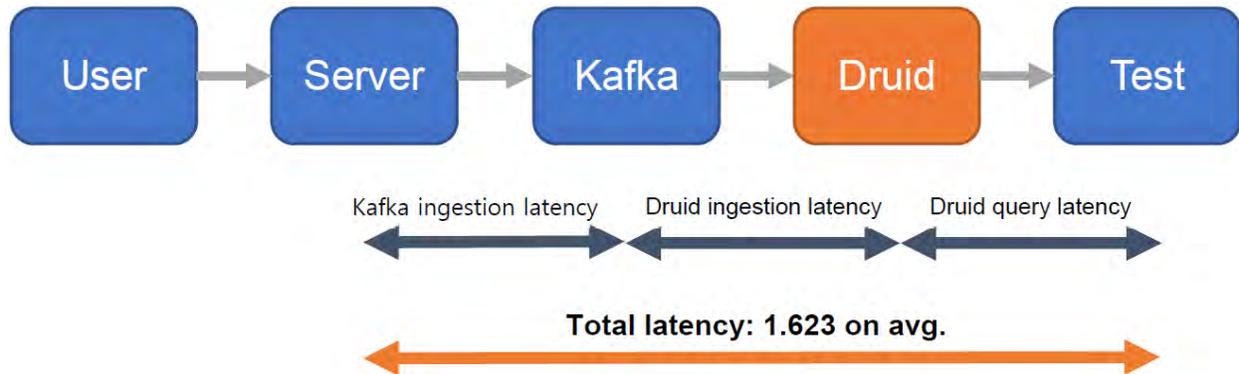


그림 12: Source: SK Telecom T-DE WIKI Metatron Project

Druid에 대한 제3자의 평가

Outlier의 Druid 평가

다음은 Outlier 블로그에 2016년 8월 26일에 게재된 Top 10 Time Series Databases라는 포스트²에서는 20개의 주요 오픈소스 시계열 데이터베이스 시스템을 평가하였습니다. 기고자인 Steven Acreman이 개인적으로 매긴 성능 랭킹에서 Druid는 20개 중 9위를 차지하였는데, 여기서 밝힌 Druid의 주요 성능은 다음과 같습니다.

² Steven Acreman. (2016, Aug 26). Top 10 Time Series Databases. Retrieved from <https://blog.outlyer.com/top10-open-source-time-series-databases>.

표 1: Outlier의 주요 Druid 평가 내용

평가 기준	Druid의 성능
쓰기 성능 - 단일 노드	25k metrics/sec 출처 : https://groups.google.com/forum/#!searchin/druid-user/benchmark%7Csort:relevance/druid-user/90BMCxz22Ko/73D8HidLCgAJ
쓰기 성능 - 5개 노드 클러스터	100k metrics/sec (추산 결과)
쿼리 성능	양호
개발 수준	안정적인 제품을 제공하는 단계에 이름
장점	괜찮은 데이터 모델이면서 좋은 분석 도구 기능들을 갖추고 있음. 주로 batch 로드된 대량 데이터셋에 대해 신속하게 쿼리하는 데 사용되도록 설계되었으며, 이 점에서 탁월한 성능을 보임.
단점	시스템 운용이 힘들. 쓰기 처리 속도가 아주 빠르지는 않음. 실시간 ingestion 셋업이 까다로움

DB-Engines의 Druid 평가

온라인 웹사이트 DB Engines³에서는 다양한 데이터베이스 관리 시스템 (DBMS) 의 시장 인기도를 매달 평가하며, 이때 다음과 같은 지표를 사용합니다.

- 인터넷에서 언급되는 횟수: Google, Bing, Yandex에서의 검색 결과로 측정
- 일반적인 관심: Google Trends에서의 검색 빈도를 기준으로 측정
- 기술 토론 빈도: 유명 IT 관련 Q&A 사이트인 Stack Overflow 및 DBA Stack Exchange 포스팅 현황을 기준으로 측정
- 구인 게시글 수: Indeed 및 Simply Hired의 게시글을 기준으로 측정
- 해당 커리어를 지닌 인재의 수: LinkedIn 및 Upwork에 게시된 프로필을 기준으로 측정
- SNS에서의 언급 수: Twitter의 트윗수를 기준으로 측정

그 결과 Druid는 2018년 7월 기준으로 총 343개 시스템 중에서 118위를 차지하였고, 그 중 시계열 데이터베이스 시스템만을 두고 집계했을 때 총 25개 시스템 중 7위를 차지하였습니다.

³ DB-Engines website. <https://db-engines.com>, July 2018.

Apache Spark와의 비교

Druid를 Apache Spark와 비교하는 것은 상당히 의미 있는 작업입니다. 둘 다 차세대 대용량 데이터 분석 솔루션으로 각광 받고 있으며, 서로 다른 장점을 가지고 있어 매우 상호보완적으로 조합이 가능하기 때문입니다. Metatron에서도 Druid를 데이터 저장/처리용 엔진으로 사용하고 Spark를 고급 분석용 모듈로 사용함으로써 이들 간의 시너지를 잘 활용하고 있습니다.

여기서는 Sparkline Data Inc.의 창업자 Harish Butani가 공개한 Druid vs Spark 성능 비교 보고서⁴⁵의 내용을 간단히 소개합니다. 보고서에서는 애초에 두 솔루션이 경쟁 관계에 있다기 보다는 상보적인 역할을 한다고 상정을 하고 성능 비교를 시작합니다.

Apache Spark의 특징

Apache Spark는 오픈소스 클러스터 컴퓨팅 프레임워크로서 Java, Scala, Python, R 언어로 이루어진 다양한 API를 제공합니다. Spark의 프로그래밍 모델은 SQL, 머신러닝, 그래프 프로세싱을 결합한 분석 솔루션을 구축하는 것입니다. Spark는 규모가 크거나 복잡한 데이터를 가공할 수 있도록 강력한 기능들을 지원하지만, Druid와 같은 interactive한 쿼리 처리에 최적화되지는 않았습니다.

데이터셋, 쿼리, 성능 비교 결과

본 성능 비교를 위한 데이터셋으로 TPCH 10G benchmark data set을 이용했습니다. 본래 이 데이터셋은 관계형 데이터베이스에 적합한 스타 스키마 구조를 갖기 때문에 이를 역정규화시킨 후 Druid와 Spark에서 처리할 수 있도록 재구성하였습니다. 이러한 처리를 거친 데이터셋의 크기는 각각 다음과 같았습니다.

- TPCH Flat TSV: 46.80GB
- Druid Index in HDFS: 17.04GB
- TPCH Flat Parquet: 11.38GB
- TPCH Flat Parquet Partition by Month: 11.56GB

그런 다음 두 솔루션의 쿼리 처리 속도를 다각도에서 분석할 수 있는 여러 쿼리를 아래와 같이 구성하였습니다.

⁴ Harish Butani. (2018, Sep 18). Combining Druid and Spark: Interactive and Flexible Analytics at Scale. Retrieved from <https://www.linkedin.com/pulse/combining-druid-spark-interactiveflexible-analytics-scale-butani>.

⁵ Harish Butani. (2015, Aug 28). TPCB Benchmark. Retrieved from <https://github.com/SparklineData/spark-druid-olap/blob/master/docs/benchmark/BenchMarkDetails.pdf>.

표 2: Druid와 Apache Spark의 query latency 비교 평가에 사용된 쿼리
내역

Query	Interval	Filters	Group By	Aggregations
Basic Aggregation.	None	None	ReturnFlag LineStatus	Count(*) Sum(exdPrice) Avg(avlQty)
Ship Date Range	1995-12/1997-09	None	ReturnFlag LineStatus	Count(*)
SubQry Nation, pType ShpDt Range	1995-12/1997-09	P_Type S_Nation + C_Nation	S_Nation	Count(*) Sum(exdPrice) Max(sCost) Avg(avlQty) Count(Distinct oKey)
TPCH Q1	None	None	ReturnFlag LineStatus	Count(*) Sum(exdPrice) Max(sCost) Avg(avlQty) Count(Distinct oKey)
TPCH Q3	1995-03-15-	O_Date MktSegment	Okey Odate ShipPri	Sum(exdPrice)
TPCH Q5	None	O_Date Region	S_Nation	Sum(exdPrice)
TPCH Q7	None	S_Nation + C_Nation	S_Nation C_Nation ShipDate.Year	Sum(exdPrice)
TPCH Q8	None	Region Type O_Date	ODate.Year	Sum(exdPrice)

테스트 결과는 다음과 같았습니다.

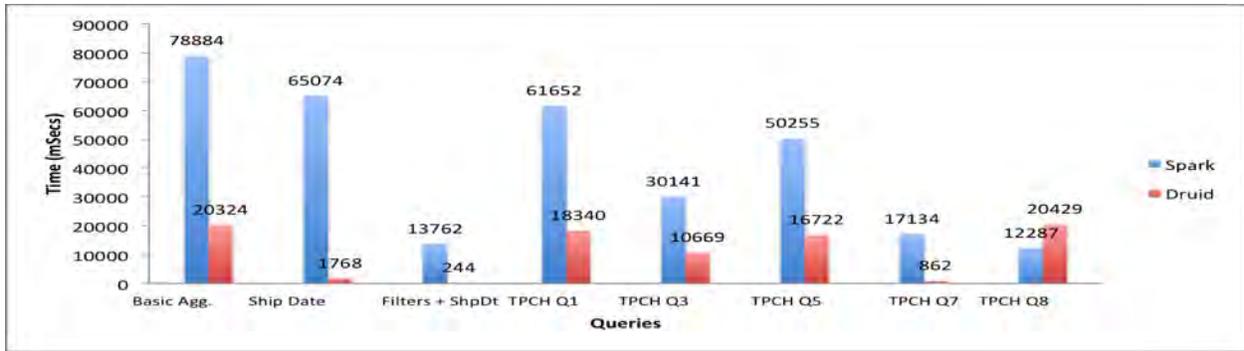


그림 13: Source: Combining Druid and Spark: Interactive and Flexible Analytics at Scale

- Filters + Ship Date 쿼리는 Druid에 특화된 slice-and-dice 성능을 테스트하는 것이었고, 예상대로 무려 50배 이상 속도 상에 우위를 보였습니다. 마찬가지로 TPCH Q7 쿼리를 처리하는 데도 Druid에서 수 밀리초가 소요된 반면, Spark에서는 수초가 소요되었습니다.
- TPCH Q3, Q5, Q8 쿼리의 경우에는 Druid가 위 경우와 같은 극대화된 효율성을 보여주지 못했습니다. OrderDate 술어는 Druid에서 JavaScript 필터로 번역이 되는데, 이는 네이티브 Java 필터에 비해 현저히 느리기 때문입니다.
- Basic Aggregation 및 TPCH Q1 쿼리의 경우에도 Druid에서 훨씬 빠른 처리 속도를 보여주었습니다. Druid에서는 Count-Distinct 동작이 cardinality aggregator로 번역이 되는데, 이는 approximate count에 해당합니다. 이러한 장점 덕에 Druid는 cardinality가 큰 차원들을 탐색할 때 유리합니다.

여러 가지 조건에 따라 결과는 달라질 수 있지만, 한 가지 분명한 것은 시간 파티셔닝 (time partitioning) 또는 차원 술어 (dimensional predicates) 를 포함하는 쿼리는 Druid에서 현저히 빠르게 처리한다는 것입니다.

시사점

이러한 테스트 결과는 Druid의 초고속 쿼리 처리 능력과 Spark의 고급 분석 기능을 결합하면 아주 훌륭한 시너지 효과를 기대할 수 있음을 시사합니다. Druid를 통해 신속하고 효율적으로 분석에 필요한 데이터만 추려낸 후 Spark의 풍부한 프로그래밍 API들을 활용하여 심층적인 분석을 실시하는 것입니다. 이렇게 함으로써 강력하고 유연하며 쿼리 latency가 매우 낮은 분석 솔루션을 구축할 수 있습니다.

참고자료

2.3.5 Metatron 엔진으로서 Druid

앞서 설명한 바와 같이 Metatron은 Druid를 기본 엔진으로서 도입하였고, Metatron만의 용도에 맞게 자체적으로 기능을 개발·보강하였습니다. 본 절에서는 이에 관한 구체적인 배경과 과정, 그리고 결과에 대해 소개합니다.

Metatron 개발 배경과 Druid 기술의 도입

빅데이터 분석 솔루션으로서 Metatron의 니즈

SK텔레콤은 국내 최대 가입자를 보유하고 있는 이동통신 서비스 제공업체로서, 수많은 이용자들로부터 발생하는 엄청난 양의 네트워크 데이터 로그를 활용하여 안정적인 네트워크 환경을 구축하는데 많은 노력을 기울이고 있습니다.

기존의 IT 인프라로는 이런 대용량 데이터 처리에 한계가 있기 때문에 SK텔레콤은 대규모 빅데이터 시스템 (Apache Hadoop) 과 이를 활용하기 위한 빅데이터 분석 솔루션이 필요했습니다. SK텔레콤은 저비용으로 대용량 데이터의 저장하기 위해 대규모 Hadoop 인프라를 자체 구축하였지만 다음과 같은 한계가 있었습니다.

- 수많은 사용자의 네트워크 데이터를 실시간으로 분석할 수 없었습니다. 빅데이터의 저장/처리는 가능했지만 데이터를 시각화하는 데는 한계가 있어 결국 과거와 같이 일부 필요한 데이터만 샘플링해서 확인해야 했습니다.
- ETL, DW, BI 등 데이터 분석 단계별로 각기 다른 솔루션, 각기 다른 담당자가 지원하는 기존의 방식은 데이터 분석을 하는 데 시간과 비용이 많이 들며 데이터 접근성을 현저하게 떨어뜨렸습니다. 단순하면서도 신속하게 데이터를 분석할 수 있으려면 분석의 전 단계를 한꺼번에 처리할 수 있는 end-to-end 솔루션이 필요했습니다.

Druid를 엔진으로 채택한 이유

Druid는 다음과 같은 특징들 때문에 위와 같은 니즈를 충족해야 하는 Metatron 솔루션의 엔진으로서 적합했습니다.

- Druid는 대용량의 데이터를 실시간으로 수집하여 즉시 쿼리 가능한 형태로 인덱싱하며, 분산 처리 기반을 통해 대용량의 데이터 집계를 아주 빠른 시간 (최대 수초) 이내에 처리해줍니다.
- Druid의 시계열 기반 OLAP Cube 데이터 포맷은 분석가가 원하는 대로 탐색, 필터링, 시각화하기 용이합니다. 실무자들이 쿼리 구성을 고민하지 않고 직관적으로 필요한 데이터를 선별하여 알고자 하는 상관 관계를 바로 출력할 수 있게 하려면 이러한 탐색 자유도와 유연성이 필수적입니다.
- Druid는 확장성이 탁월하여 각종 모듈을 추가하기 용이합니다.

Metatron은 Druid의 이러한 특성을 활용하여 데이터 수집, 저장, 처리, 분석, 시각화 등의 모든 layer를 포괄하는 end-to-end 솔루션을 구축하였습니다.

Druid 응용 방식

Metatron에서 Druid 엔진을 응용하는 방식은 다음과 같습니다.

- 사용자(현업/빅데이터 분석가) 측면에서 Druid를 기본 처리/분석 엔진으로 사용하기 쉽도록 GUI 화면을 구성하여 데이터 전처리, 분석, 시각화 등 데이터 관련 업무를 수행하고 이를 공유할 수 있게 하였습니다.

- IT 운영자는 Druid 내 데이터 소스를 관리/모니터링 할 수 있고, 필요시 데이터 전처리 작업을 통해 고품질의 데이터 소스를 제공할 수 있도록 지원합니다.

Metatron에서 보강한 Druid 기능들

Druid는 강력한 데이터 수집·처리 기능을 지원하지만, Metatron이 end-to-end 솔루션으로 온전히 기능하기 위해서는 기존의 오픈소스 Druid를 개선할 필요가 있었습니다. 본 절에서는 기존 오픈소스 Druid의 한계와 Metatron에서 보강한 기능들을 살펴보겠습니다.

오픈소스 Druid의 한계

기존의 오픈소스 Druid의 한계는 다음과 같습니다.

- Druid는 데이터 테이블 join에 제약이 많습니다. 이런 이유로 Metatron은 데이터 전처리를 위해 다른 SQL 엔진으로 사용합니다.
- Druid는 SQL 쿼리를 부분적으로만 지원합니다.
- 데이터 레이크용으로는 기존의 SQL 엔진을 사용하는 편이 더 낫습니다.
- 이미 인덱싱된 세그먼트를 수정하거나 행을 추가할 수 없습니다. 물론 incremental ingestion과 같은 특수한 상황에는 가능하지만 일반적이지 않습니다.
- null 값이 지원되지 않습니다.
- 측정값에 대한 필터링이 지원되지 않습니다.
- Linear scalability가 보장되지 않습니다. 서버의 수를 늘려도 성능이 크게 개선되지 않습니다.
- Druid에서 지원하는 자료형은 제한적이며 추가하기가 어렵습니다.
- 관리 및 모니터링 도구가 성능이 우수하지 않습니다.

Metatron에서 보강한 Druid 기능들

Metatron은 Druid에서 미비한 기능들을 다음과 같이 보강하였습니다.

쿼리 성능 개선

- groupBy 쿼리 성능 개선
- 기타 쿼리 성능 소폭 개선

기능 추가

- 가상 컬럼 (map, expression 등)
- 측정값 자료형 확장 (double, string, array 등)
- 계산식 기능 확장
- Druid 쿼리 결과를 HDFS 또는 파일로 내보낼 수 있음
- 데이터 테이블에 대한 메타 정보 및 통계용 쿼리 추가
- 집계 기능 추가 (분산, 상관관계 등)
- (제한적) 관계형 DB의 window 기능들 구현 (lead, lag, running aggregations 등)
- (제한적) join 기능 강화
- (제한적) 서브 쿼리 추가
- 임시 데이터 소스
- 복합 쿼리 추가 (데이터 소스 summarization, 데이터 소스 간 상관관계, k-평균 등)
- 사용자 정의 컬럼 grouping 지원
- GIS(지도-GEO서버 연계 adaptor 제공) 기능 지원
- 컬럼별 histogram 제공
- Bit-slice indexing 지원

인덱스 구조 개선

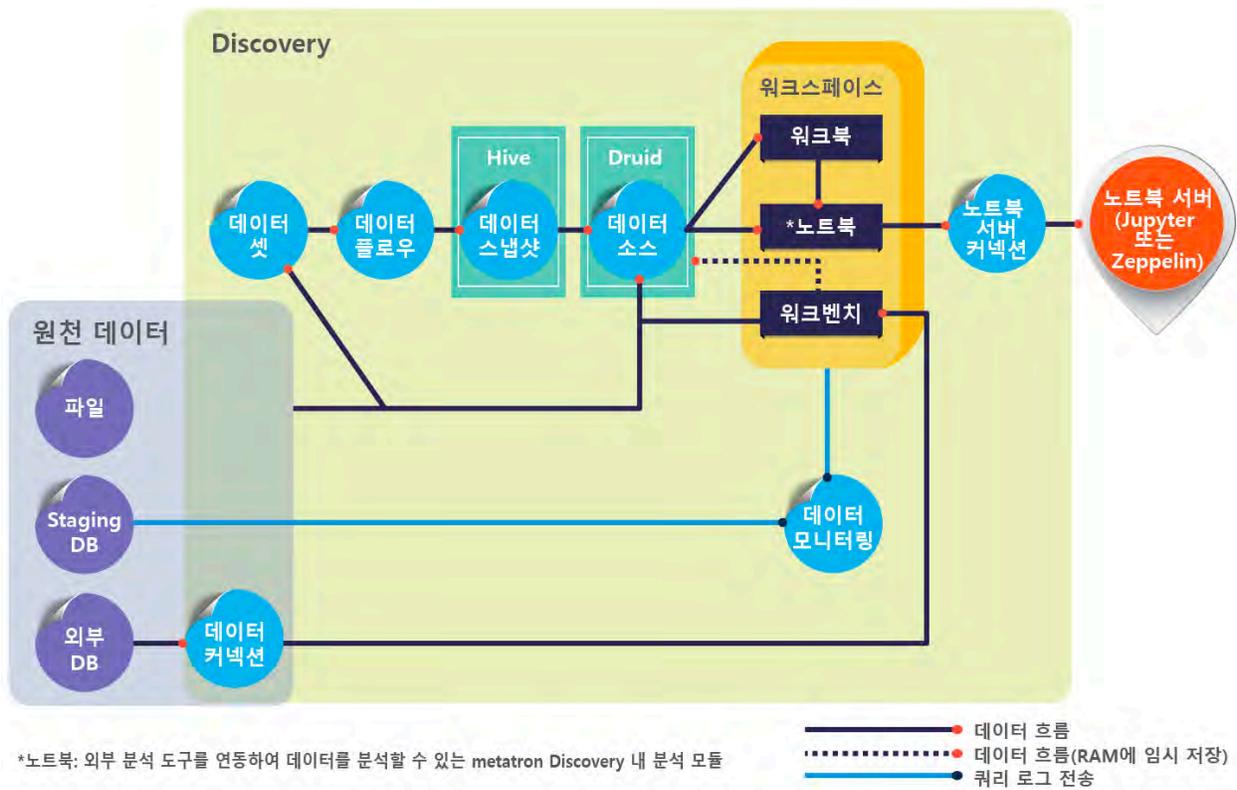
- 측정값 필터링용 히스토그램
- 텍스트 필터링용 lucene 포맷 지원

다른 시스템들과의 연동 지원

- Hive 저장소 핸들러
- Hive 테이블 ingestion(Hive 메타 스토어와 연결)
- ORC 포맷 ingestion
- JDBC를 통한 RDBMS 데이터 ingestion
- (제한적) Backport된 SQL 지원

기타 개선사항

- 버그 수정 (50건 이상), 기능 추가 및 기타 경미한 사항들 개선



위 그림과 같이 Metatron Discovery의 3가지 분석 모듈 (워크북, 노트북, 워크벤치)에서 사용하는 데이터는 다양한 원천 데이터 유형과 여러 엔진 및 저장소를 통해 마련됩니다. 따라서 이러한 데이터 흐름을 정형화 및 관리하고 여러 원천 데이터를 연결시켜주는 작업이 반드시 필요합니다.

데이터 분석 및 시각화에 필요한 원천 데이터는 Metatron 내부 엔진로 가져와 **데이터 소스** 단위로 저장하거나, 아니면 **데이터 커넥션**을 통해 외부 데이터베이스와 직접 연결하여 사용할 수 있습니다. 그리고 이러한 데이터의 사용 현황은 **데이터 모니터링**을 이용하여 감독하고 추적할 수 있습니다.

3.1 데이터 소스

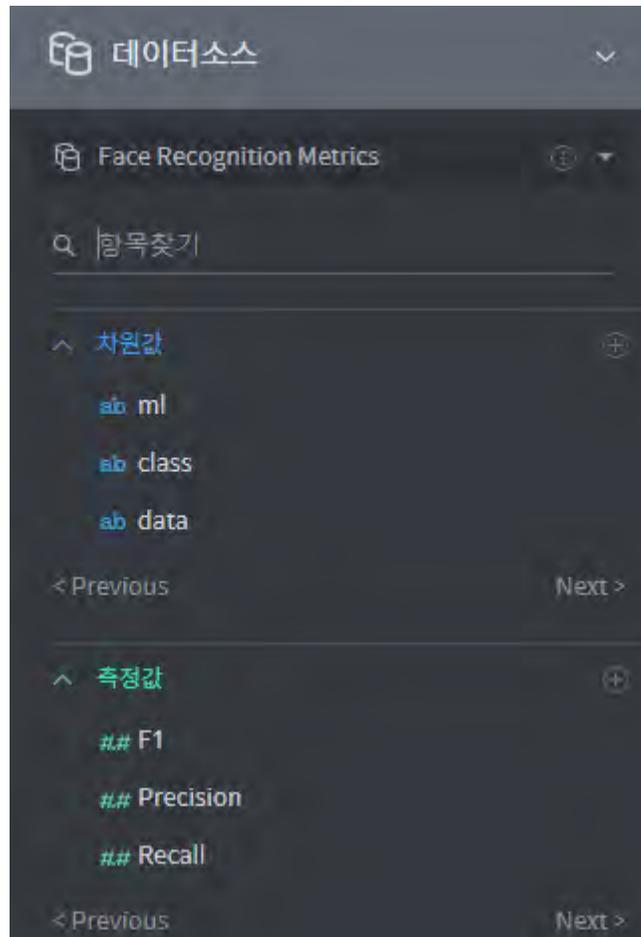
Metatron Discovery에서 <데이터 소스>는 Druid 엔진에 수집되는 데이터 단위를 의미합니다. 각 데이터 소스는 Druid 데이터베이스에서 하나의 테이블로서 저장됩니다. 이러한 데이터 소스들은 <워크북>이나 <노트북>에서 분석·시각화하는데 사용됩니다.

데이터 소스 메뉴는 메인 화면 좌측 패널에서 **MANAGEMENT > 데이터 스토리지 > 데이터 소스**를 통해 진입할 수 있습니다.



3.1.1 <차원값> 과 <측정값> 의 개념

대시보드에 연동된 데이터 소스의 컬럼들은 아래와 같이 차원값 컬럼과 측정값 컬럼으로 구분됩니다. Discovery의 데이터 분석·시각화 기능을 온전히 활용하기 위해서는 차원값과 측정값의 개념을 명확하게 이해해야 합니다.



차원값 (Dimension) 컬럼

범주형 데이터 컬럼을 가리키며, 특징은 아래와 같습니다.

- 집계 (aggregated) 보다는 분류 (categorical) 에 의미가 있는 데이터 필드 (예: Category, Region, Organization 등)
- 측정값을 표시하는 데 기준이 됨.

측정값 (Measure) 컬럼

수량적 데이터 필드를 가리키며, 특징은 아래와 같습니다.

- 집계 (aggregated) 할 수 있거나 양적인 (quantitative) 정보를 포함하는 필드 (예: Sales 등)
- 차원값에 의해 제시된 기준을 토대로 차트에 표현되는 데이터

3.1.2 데이터 소스 관리 홈 화면

본 화면에서는 데이터 소스의 신규 등록 · 편집 · 조회가 가능합니다.

6 데이터 소스	소스 타입	적재 타입	상태	생성일
DATA (CSV)	파일	수집형 데이터	Enabled	2019-01-25 18:26 by Polaris
CSFB (Excel)	파일	수집형 데이터	Failed	2019-01-25 18:17 by Polaris
CSFB (CSV)	파일	수집형 데이터	Enabled	2019-01-25 18:15 by Polaris
CSFB (Excel)	파일	수집형 데이터	Failed	2019-01-25 18:13 by Polaris
loc	파일	수집형 데이터	Enabled	2019-01-25 17:33 by Polaris
CSFB	파일	수집형 데이터	Failed	2019-01-25 17:00 by Polaris
tet-test	데이터베이스	연결형 데이터	Enabled	2019-01-25 15:58 by Administrator
test sale-test sale	데이터베이스	수집형 데이터	Failed	2019-01-25 15:49 by Administrator

1. **상태:** 현재 데이터 스토리지에 저장되어 있는 데이터 소스의 가용 여부로 선별하여 조회합니다.

- **Enable:** 데이터 ingestion을 마쳐 워크북이나 워크벤치에서 사용 가능한 데이터 소스들이 출력됩니다.
- **Preparing:** 아직 생성된 지 얼마 되지 않아서 데이터 ingestion이 진행 중인 데이터 소스들이 출력됩니다.
- **Failed:** 생성이 제대로 되지 않은 데이터 소스들이 출력됩니다.
- **Disabled:** 데이터 ingestion을 마쳤으나 Druid 엔진의 일부 프로세스에서 제대로 진행이 되지 않아 사용이 불가능한 데이터 소스들이 출력됩니다.

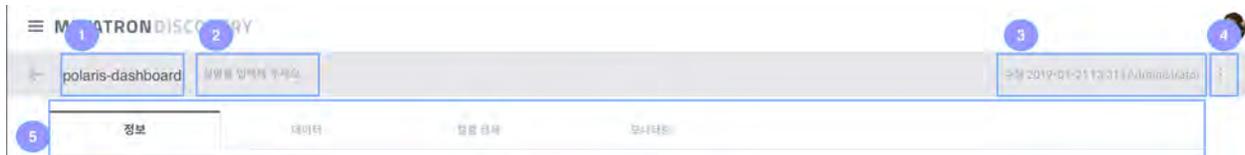
2. **공개:** 데이터 소스의 공개 대상여부로 선별하여 조회합니다.

- **공개 데이터:** 모든 워크스페이스에서 사용이 허용된 데이터 소스들만 선별하여 조회합니다.
- **Admin workspace:** 어드민 워크스페이스에서 사용이 허용된 데이터 소스들만 선별하여 조회합니다.

- **Shared Workspace:** shared workspace에서 사용이 허용된 데이터 소스들만 선별하여 조회합니다.
3. **생성한 사람:** 해당 데이터 소스를 생성한 사용자 또는 그룹을 조회합니다.
 4. **생성한 시간:** 데이터 소스 조회 시 적용되는 시간 기준입니다. 생성일과 수정일 중 원하는 기준으로 선택할 수 있으며 시간 범위는 전체/오늘/지난 7일/특정 날짜 기간 중 선택이 가능합니다.
 5. **데이터 소스 이름으로 검색:** 현재 등록된 데이터 소스를 이름으로 검색합니다
 6. **데이터 소스 목록:** 설정한 선별 조건에 맞는 데이터 소스들을 보여줍니다. 이 중 하나를 클릭하면 상세 내역을 볼 수 있습니다. ([데이터 소스 상세 정보 참조](#))
 7. **삭제:** 해당 데이터 소스에 마우스 오버 시 휴지통 아이콘이 나타납니다. 클릭하면 해당 데이터 소스를 삭제할 수 있습니다.

3.1.3 데이터 소스 상세 정보

데이터 소스 관리 홈에 열거된 데이터 소스 중 하나를 클릭하면 해당 데이터 소스에 관한 다양한 속성을 확인할 수 있습니다. 아래 각 영역을 확인할 때에는 각 데이터 소스가 Metatron의 내부 Druid 데이터베이스에 하나의 테이블로 저장되며, Druid의 시계열 특성 때문에 반드시 타임스탬프 컬럼을 포함하게 됨을 유의하십시오.



상단 공통 영역

1. **이름:** 해당 데이터 소스의 이름입니다. 클릭 시 수정 가능합니다.
2. **설명:** 해당 데이터 소스에 대한 설명입니다. 클릭 시 수정 가능합니다.
3. **마지막 수정정보:** 해당 데이터 소스를 누가 언제 마지막으로 수정했는지 보여줍니다.
4. **삭제:** 이 아이콘을 누르면 해당 데이터 소스를 삭제할 수 있는 메뉴가 나타납니다.
5. **탭 선택 영역:** 각 탭은 해당 데이터 소스에 관한 특정한 속성군을 보여줍니다. 데이터 소스의 종류에 따라 3개 탭이 모두 나오지 않을 수도 있습니다. 각각에 대한 자세한 설명은 아래의 각 절에서 확인하십시오.

데이터 정보 영역

이 영역에서는 해당 데이터 소스의 기본 정보를 보여줍니다.

The screenshot displays the '데이터 정보' (Data Info) section of the METATRON DISCOVERY interface. It lists five key attributes of the data source:

- 데이터 타입 (Data Type):** 수집할 데이터 (Data to be collected)
- 상태 (Status):** ENABLED
- 사이즈 (Size):** 35.43 KB
- 인터벌 (Interval):** 2018-06-01T00:00:00.000Z ~ 2018-10-01T00:00:00.000Z
- 타임스탬프 설정 (Timestamp Settings):** 쿼리 단위 (Query Unit) NONE, 세그먼트 단위 (Segment Unit) NONE

Below the list is a histogram titled '히스토그램' (Histogram) showing data volume in KByte over time. The x-axis represents time intervals from 2018-06-01 to 2018-09-01, and the y-axis represents volume in KByte, ranging from 0 to 80. The bars show varying levels of data volume across these intervals.

- 데이터 타입:** 해당 데이터 소스 생성 시 가져온 원천 데이터의 타입입니다.
- 상태:** 해당 데이터 소스의 현재 사용 가능 여부를 나타냅니다.
- 사이즈:** 해당 데이터 소스의 크기를 나타냅니다.
- 인터벌:** 해당 데이터 소스에 포함된 타임스탬프의 시간 범위를 나타냅니다.
- 타임스탬프 설정:** 해당 데이터 소스 생성 시 설정한 Granularity 주기를 나타냅니다.
 - 쿼리단위:** 분석에서 수행하고자 하는 최소 시간 단위를 결정합니다. 이는 최소 단위까지의 결과를 미리 생성하여 이후에 보다 빠른 응답을 얻을 수 있도록 하기 위함입니다.
 - 세그먼트단위:** 분산 환경에서 동작하는 Druid의 특성을 활용하기 위해 데이터를 분할하여 저장하게 되는데 이때 저장하는 단위를 결정합니다.
 - 히스토그램:** 각 시간대별로 저장된 데이터의 용량을 KByte 단위로 보여주는 그래프입니다. 이러한 히스토그램은 레코드별로 반드시 타임스탬프 기록을 남겨야하는 Druid 엔진의 특성에 따른 것입니다.

데이터 소스에 Publish 권한주기

이 영역에서는 해당 데이터 소스를 어느 워크스페이스에서 사용할 수 있는지를 확인·설정합니다.

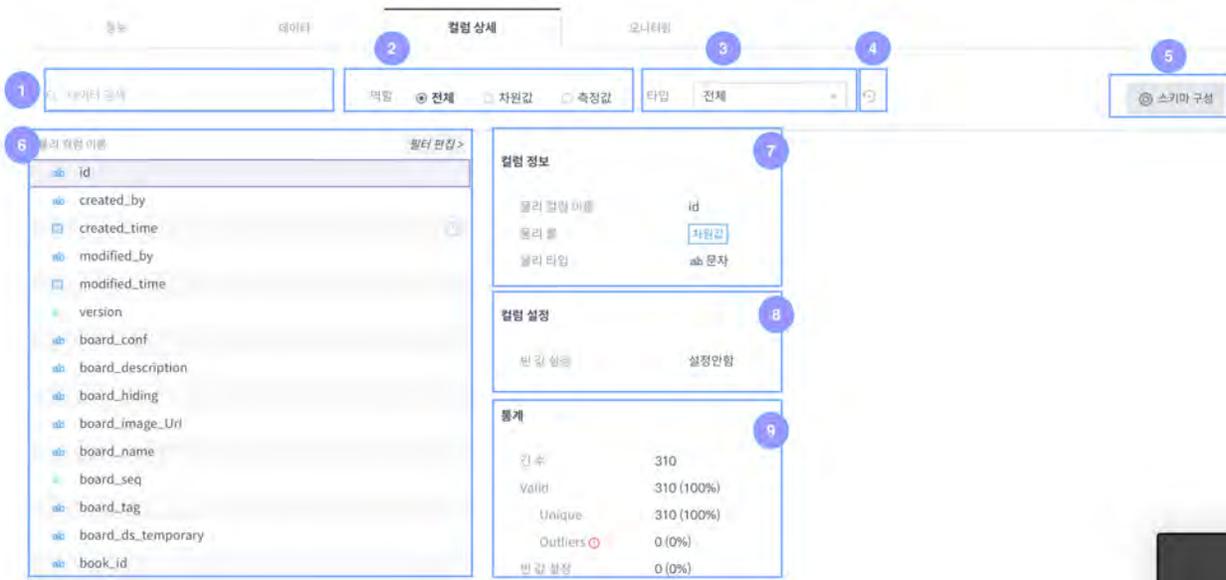


1. **모든 워크스페이스에 데이터 소스를 사용하도록 허용**: 이 확인란에 체크하면 모든 워크스페이스에서 해당 데이터 소스를 사용할 수 있습니다.
2. **수정**: 해당 데이터 소스의 사용을 허용할 특정 워크스페이스를 지정할 때 사용합니다. 해당 데이터 소스를 오픈 데이터로 지정하면 이 버튼이 사라집니다.
3. **공유 워크스페이스 수**: 해당 데이터 소스를 사용할 수 있도록 허용된 워크스페이스의 수를 나타냅니다.

데이터 스키마 변경하기

컬럼 상세 탭의 상단부는 원하는 조건으로 컬럼을 선별하는 사용자 인터페이스를 제공합니다. 조건에 맞게 선별된 컬럼들은 좌측 영역에 조회됩니다. 또한 여기서 컬럼 설정값을 수정할 수도 있습니다.

컬럼 조회/설정 기능



1. **데이터 검색**: 컬럼 이름으로 검색이 가능합니다.
2. **역할**: 데이터 테이블의 컬럼을 전체/차원값/측정값으로 선별하여 조회합니다.
3. **타입**: 데이터 테이블의 컬럼을 필드 타입별로 선별하여 조회합니다.

4. **전체 보기:** 데이터 검색, 역할, 타입 옵션에서 설정한 모든 선별 조건을 취소하고 전체 컬럼 보기로 회귀합니다.
5. **스키마구성:** 클릭하면 현재의 컬럼 설정값을 바꿀 수 있는 창이 출력됩니다.
6. **컬럼목록:** 해당 테이블을 구성하는 컬럼들을 열거합니다.
7. **컬럼정보:** 선택된 컬럼의 속성을 보여줍니다.
8. **컬럼설정:** 선택된 컬럼의 메타 데이터 정보를 보여줍니다.
9. **통계:** 선택된 컬럼에 입력된 값에 대해서 건수 및 통계값을 보여줍니다.

스키마구성

컬럼들의 명칭 및 타입 수정을 하는 사용자 인터페이스를 제공합니다.

스키마 설정

취소 저장

1	2	3	4	5	6
역할	물리 이름	용어 이름	논리 타입	표시 형식	설명
차원값	id	id	ab 문자		
차원값	created_by	created_by	ab 문자		
차원값	created_time	created_time	TF 불린		
차원값	modified_by	modified_by	# 정수		yyyy-MM-dd HH:mm:ss.SSS
차원값	modified_time	modified_time	## 소수		yyyy-MM-dd HH:mm:ss.SSS
측정값	version	version	# 정수		
차원값	board_conf	board_conf	ab 문자		
차원값	board_descripti...	board_descripti...	ab 문자		
차원값	board_hiding	board_hiding	ab 문자		
차원값	board_image_Url	board_image_Ur...	ab 문자		
차원값	board_name	board_name	ab 문자		
측정값	board_seq	board_seq	# 정수		

1. **역할:** 해당 컬럼 차원값/측정값의 여부를 표시합니다.
2. **물리이름:** 해당 컬럼의 실제 명칭을 표시합니다.

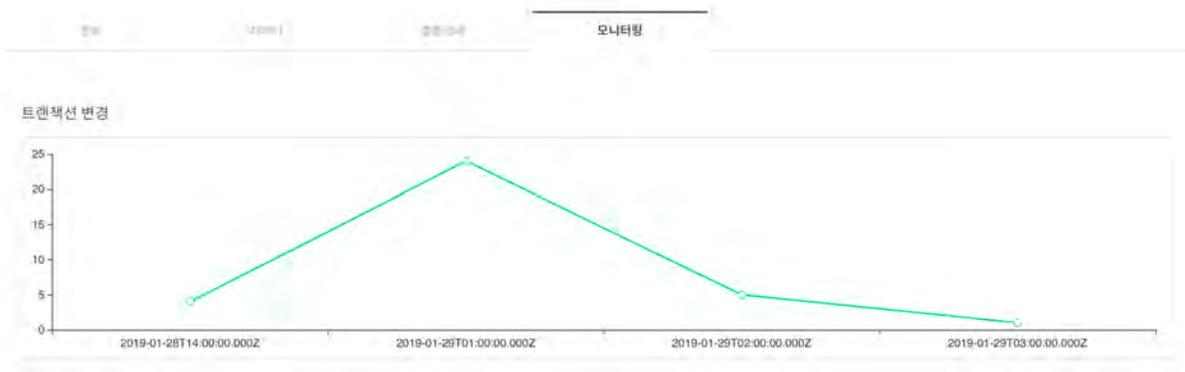
3. **용어이름**: 해당 컬럼이 해당 시스템에서 표시될 용어 이름을 표시하고 수정할수 있습니다.
4. **논리타입**: 해당 컬럼의 논리타입 (문자/숫자/날짜등) 을 표시하고 수정할수 있습니다.
5. **표시형식**: 해당 컬럼이 타임스탬프 등인 경우 표시 포맷을 표시합니다.
6. **설명**: 해당 컬럼의 상세 설명을 표시하고 수정할수 있습니다.

데이터 통계 분석하기

모니터링 탭에서는 데이터 소스가 사용된 로그를 볼 수 있습니다.

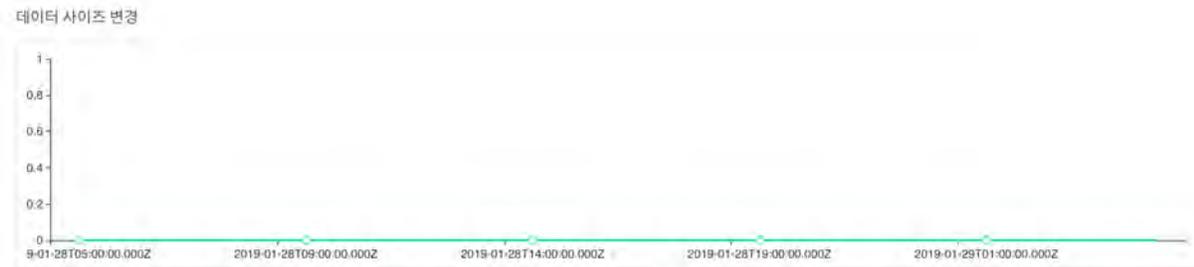
트랙잭션 변경

해당 데이터 소스의 시간에 따른 트랜잭션량 추이를 보여줍니다.



데이터 사이즈 변경

해당 데이터 소스의 시간에 따른 용량 추이를 보여줍니다.



쿼리사용분포 (지난 1주일 동안)

사용자별 쿼리사용 분포 (지난 1주일 동안)



응답 시간별 쿼리 분포 (지난 1주일 동안)



- 사용자별 쿼리사용 분포 (지난 1주일 동안): 지난 한 주간 쿼리를 수행한 사용자별로 분류하여 그래프로 보여줍니다.
- 응답 시간별 쿼리사용 분포 (지난 1주일 동안): 지난 한 주간 쿼리를 수행한 소요시간별로 분류하여 그래프로 보여줍니다.

쿼리 로그

수행된 각 쿼리에 대한 상세 이력을 확인할 수 있습니다.



1. 날짜: 확인하고자 하는 쿼리들의 실행 시간대를 설정합니다.
2. 쿼리 타입: 실행한 쿼리를 타입별로 선별하여 출력합니다.
3. 상태: 쿼리 결과를 성공/실패로 선별하여 출력합니다.
4. 쿼리 목록: 설정한 조건에 부합하는 쿼리들이 나열됩니다.
5. 자세히: 클릭하면 해당 쿼리문을 확인할 수 있습니다.

3.1.4 데이터 소스 만들기

본 절에서는 다양한 형태의 원천 데이터를 Metatron 엔진으로 가져와 데이터 소스로 만드는 과정을 설명합니다.

데이터 소스를 생성하려면 데이터 소스 홈 화면 우측 상단에서 + 새로만들기 버튼을 클릭합니다.

데이터 스토리지



그런 다음 원천 데이터의 타입을 선택합니다.

소스 타입을 선택해 주세요

 파일

 데이터베이스

 Staging DB

 실시간

 데이터스냅샷

 메타트론 엔진

취소

- **파일:** 사용자의 로컬 PC에 저장되어 있는 파일을 가져와서 데이터 소스를 생성합니다 (자세한 절차는 [파일로 데이터 소스 만들기](#) 참조).
- **데이터베이스:** 외부 데이터베이스에서 데이터를 가져와서 데이터 소스를 생성합니다 (자세한 절차는 [DB로 데이터 소스 만들기](#) 참조).
- **Staging DB:** Metatron의 내부 Hive 데이터베이스에서 가져온 데이터를 기반으로 데이터 소스를 생성합니다 (자세한 절차는 [StagingDB로 데이터 소스 만들기](#) 참조).
- **실시간:** 현재 해당 기능은 지원하지 않습니다.
- **데이터스냅샷:** 현재 해당 기능은 지원하지 않습니다.
- **Metatron 엔진:** Metatron 이전 버전에 저장된 데이터 소스를 마이그레이션 합니다 (자세한 절차는 [Metatron 엔진을 통해 데이터 소스 추가하기](#) 참조).

파일로 데이터 소스 만들기

사용자의 로컬 PC에 저장되어 있는 파일을 가져와서 데이터 소스를 생성합니다.

1. 원천 데이터 타입 선택 화면에서 **파일**을 선택합니다.
2. 사용자 로컬 PC에서 데이터 소스로 사용할 파일을 가져옵니다. **Import** 버튼을 클릭하여 파일을 선택할 수도 있고 화면 상으로 파일을 끌어다 놓을 수도 있습니다. 파일을 가져왔으면 다음 버튼을 누릅니다.



- 가져온 파일에서 데이터 소스에 포함시킬 시트를 선택합니다.

참고: 데이터가 있음에도 불구하고 <미리보기 데이터가 없습니다>로 나오는 경우에는, 컬럼 구분자 및 라인 구분자를 맞게 설정했는지 확인해야 합니다. 이 예제에서는 라인 구분자가 MS Windows의 carriage return인 <r>로 입력이 되어야 합니다.



- **파일 이름:** 가져온 파일의 이름입니다. 다른 파일을 다시 가져올 수도 있습니다.
- **파일 시트 목록:** 가져온 파일에 포함된 시트들을 보여줍니다. 여기서 데이터 소스로 만들 시트를 선택합니다.
- **파일 시트 이름:** 현재 선택된 시트 이름입니다.
- **용량:** 가져온 파일의 용량입니다.
- **컬럼:** 가져온 파일의 컬럼 개수입니다.
- **행:** 가져온 파일의 행 개수입니다. 숫자를 입력하면 해당 숫자만큼의 행이 화면에 나타납니다.
- **타입:** 각 컬럼으로부터 인식한 데이터 타입이 몇 종류인지 보여줍니다. 컬럼별 데이터 타입은 이후 화면에서

수정할 수 있습니다.

- **첫째 행을 컬럼명으로 사용 여부 확인란:** 선택하면 파일 내의 첫번째 행의 내용이 컬럼명으로 사용됩니다. 선택하지 않을 경우 컬럼명을 기재할 행이 새로 생성됩니다.

4. 데이터 소스에서 구현하고자 하는 스키마를 설정합니다.

데이터소스 생성하기 (파일)
스키마 구성

🔍 컬럼 이름으로 검색해 주세요

역할: 전체 | 타입: 전체 | + 컬럼 추가

선택	컬럼	타입	역할
<input type="checkbox"/>	차원값 ab OrderDate		
<input type="checkbox"/>	차원값 ab Category		
<input type="checkbox"/>	차원값 ab City		
<input type="checkbox"/>	차원값 ab Country		
<input type="checkbox"/>	차원값 ab CustomerName		
<input type="checkbox"/>	차원값 ab Discount		
<input type="checkbox"/>	차원값 ab OrderID		
<input type="checkbox"/>	차원값 ab PostalCode		
<input checked="" type="checkbox"/>	차원값 ab ProductName		
<input checked="" type="checkbox"/>	차원값 ab Profit		
<input type="checkbox"/>	차원값 ab Quantity		
<input type="checkbox"/>	차원값 ab Region		
<input type="checkbox"/>	차원값 ab Sales		
<input type="checkbox"/>	차원값 ab Segment		
<input checked="" type="checkbox"/>	차원값 ab ShipDate		
<input type="checkbox"/>	차원값 ab ShipMode		
<input type="checkbox"/>	차원값 ab State		
<input type="checkbox"/>	차원값 ab Sub_Category		
<input type="checkbox"/>	차원값 ab DaystoShipActual		
<input type="checkbox"/>	차원값 ab SalesForecast		
<input type="checkbox"/>	차원값 ab ShipStatus		
<input type="checkbox"/>	차원값 ab DaystoShipScheduled		
<input type="checkbox"/>	차원값 ab OrderProfitable		

ShipDate

데이터

2011-01-08T00:00:00.000Z
2011-01-09T00:00:00.000Z
2011-01-09T00:00:00.000Z
2011-01-09T00:00:00.000Z
2011-01-13T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-08T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-09T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-11T00:00:00.000Z
2011-01-13T00:00:00.000Z
2011-01-14T00:00:00.000Z
2011-01-14T00:00:00.000Z
2011-01-16T00:00:00.000Z
2011-01-16T00:00:00.000Z
2011-01-15T00:00:00.000Z
2011-01-17T00:00:00.000Z
2011-01-19T00:00:00.000Z

역할

차원값
 측정값

타입

ab 문자

빈 값 설정

설정안함
 버림
 대체

3 선택 | 타입 변경 | 삭제

🕒 시간 타입 또는 현재 시간 중 하나는 시간값으로 지정되어야 합니다

현재 시간 | 시간 타입 컬럼 | 선택된 타임타입 컬럼이 없습니다

- **컬럼명으로 검색:** 가져온 파일에 들어있는 컬럼을 이름으로 검색합니다.
- **🗑️ 버튼 (우측 상단):** 선택한 컬럼을 삭제합니다.
- **역할:** 가져온 파일에 들어있는 컬럼을 전체/차원값/측정값으로 선별하여 조회합니다.
- **추천 필터:** 최우선 추천 필터가 적용된 컬럼들만 선별하여 조회합니다.
- **타입:** 가져온 파일에 들어있는 컬럼을 필드 타입별로 선별하여 조회합니다.
- **컬럼 목록 영역:** 설정한 선별 조건에 맞는 컬럼들을 보여줍니다. 컬럼들을 선택하면 하단에 패널이 나타나는데, 여기서 원하는 일괄 동작을 선택한 후 **적용**을 클릭하면 선택한 컬럼들에 대한 일괄

동작이 수행됩니다.

- **개별 컬럼 설정 영역:** 컬럼 목록에서 선택한 컬럼의 속성들을 설정할 수 있는 영역입니다. 여기서 빈 값 설정은 컬럼 내 Null 값을 처리하는 방식을 설정하는 항목입니다.
 - **대체:** 여기에 입력된 값으로 Null 값이 대체됩니다.
 - **버림:** Null 값을 버립니다.
 - **설정안함:** Null 값이 그대로 보여집니다. 단 데이터 소스의 타임스탬프의 Null 값은 무조건 버려지게 됩니다.
- **타임스탬프 설정:** 각 행에 타임스탬프를 지정하는 방식을 결정합니다. 기존 데이터가 보유하고 있는 시간 타입 컬럼을 타임스탬프로 지정하거나, 아니면 현재 시간 값으로 이루어진 시간 타입 컬럼을 생성하여 타임스탬프로 지정할 수 있습니다.

참고: Metatron 엔진은 데이터 소스 저장 시 무조건 시간 값을 보유해야 하는 시계열 엔진입니다.

- **컬럼 추가:** 데이터에 위도, 경도 컬럼이 있는 경우 이를 결합하여 Point 타입의 신규 컬럼을 추가할 수 있습니다. 이 컬럼을 지우면 다른 컬럼들과 동일하게 동작합니다.

5. 데이터 소스 수집 설정을 하고 다음 버튼을 누릅니다.

데이터소스 생성하기 (파일)
수집 설정을 완료해 주세요

타임스탬프 설정

세그먼트 단위
월

쿼리 단위
월

롤업
 설정함
 설정안함

고급 설정 ▾

이전 다음

- **세그먼트 단위:** 분산 노드 환경에서 동작하는 Druid의 특성을 활용하기 위해 데이터를 분할하여 저장하게 되는데 이때 저장하는 시간 단위를 결정합니다.
- **쿼리 단위:** 분석에서 수행하고자 하는 최소 시간 단위를 결정합니다. 이는 최소 단위까지의 결과를 미리 생성하여 이후에 보다 빠른 응답을 얻을 수 있도록 하기 위함입니다.
- **롤업:** <데이터 롤업>은 차원값을 기준으로 데이터를 요약하는 작업입니다 (<데이터 롤업>의 개념에 대한 보다 상세한 설명은 [데이터 roll-up](#) 참조). 요약 규칙은 계층 구조를 따라 합계를 계산하거나 profit=sales-expenses와 같은 수식 집합을 적용하는 것일 수 있습니다.
- **고급설정:** 데이터 적재 성능을 설정합니다. 텍스트상자에 JSON 형식의 구문을 입력하십시오. 예,

```
{maxRowsInMemory : 75000,
maxOccupationInMemory : -1,
maxShardLength : -2147483648,
leaveIntermediate : false,
cleanupOnFailure : true,
overwriteFiles : false,
ignoreInvalidRows : false,
assumeTimeSorted : false}
```

6. 가져온 파일에서 설정한 데이터에 관한 정보를 확인한 뒤, 이름과 설명을 입력하고 완료 버튼을 누르면 데이터 소스가 생성됩니다. 이때, 원천 데이터에서부터 Metatron 내장 엔진 (Druid) 으로 적재 (ingestion) 하기 때문에 데이터량에 따라 수 초~분의 시간이 소요될 수 있습니다.

← us 500 이름을 입력해 주세요

정보
데이터
컬럼 상세
모니터링

데이터 정보

적재 타입

상태

이원스탬프 설정

히스토그램

수집형 데이터

ENABLED

1

2

3

4

데이터 준비

엔진 적재

상태 확인

완료

세그먼트 단위 : MONTH

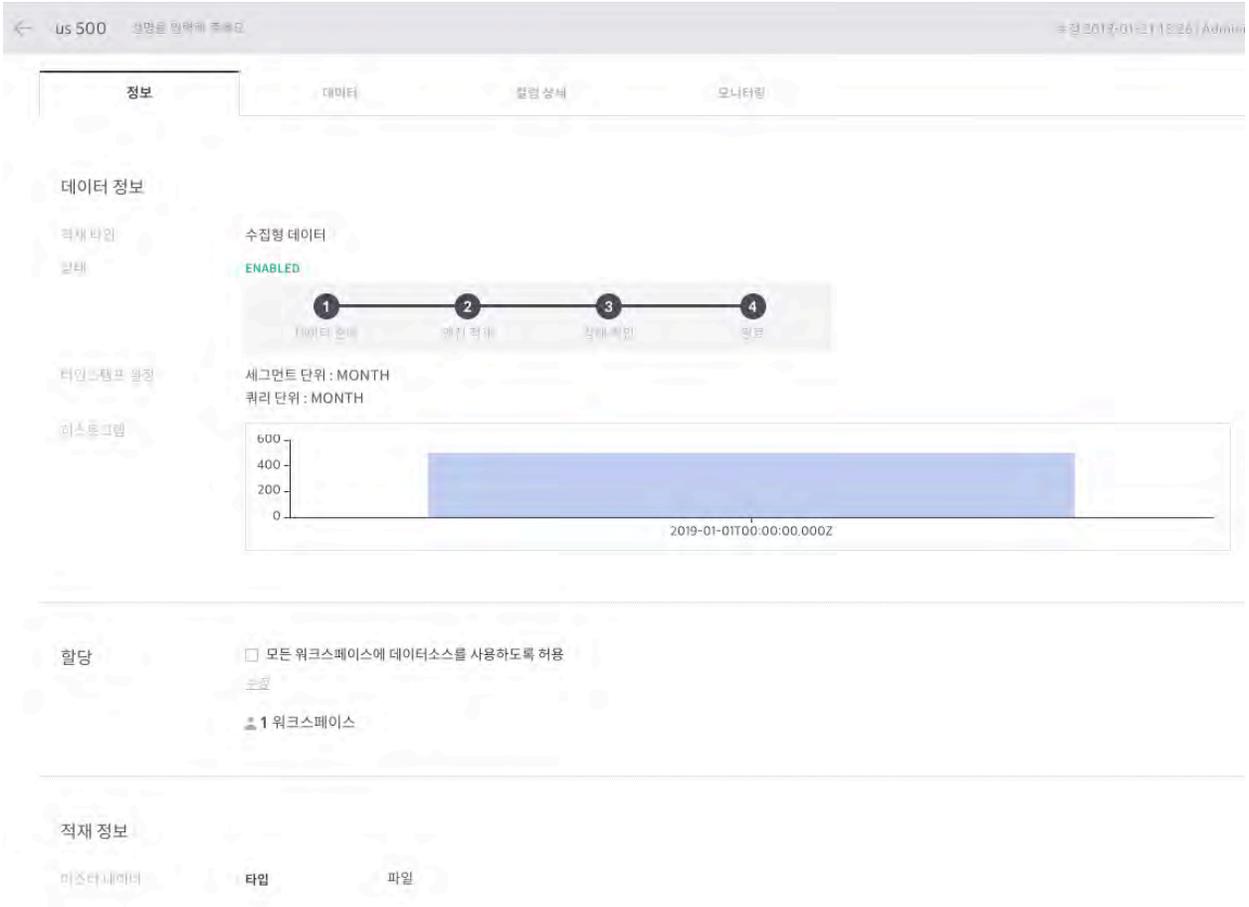
쿼리 단위 : MONTH

할당

모든 워크스페이스에 데이터소스를 사용하도록 허용

추천

7. 데이터 적재가 완료된 후 상태를 확인해볼 수 있습니다. 아래 예시에서는 상태가 **ENABLED**로 되어 있으며 히스토그램이 보입니다.



8. 데이터 탭으로 이동을 하면 적재된 데이터를 테이블 형태로 확인할 수 있습니다.

us 500 생성된 일대일 주해도 수정 2019-01-21 13:26 | Administrator

정보 **데이터** 활용 상세 모니터링

데이터 검색 역할 전체 차원값 측정값 타입 전체 100 행 다운로드

ab first_name	ab last_name	ab company_na...	ab address	ab city	ab county	ab state	ab zip	ab phone1	ab phone2	ab email
Abel	Maclead	Rangoni Of Florence	37275 St R...	Middle Island	Suffolk	NY	11953	631-335-3...	631-677-3...	amacl
Adelina	Nabours	Courtyard By Marr...	80 Pittsfor...	Cleveland	Cuyahoga	OH	44103	216-230-4...	216-937-5...	adelin
Adell	Lipkin	Systems Graph Inc...	65 Mountai...	Whippany	Morris	NJ	07981	973-654-1...	973-662-8...	adell.l
Ahmed	Angalich	Reese Plastics	2 W Beverl...	Harrisburg	Dauphin	PA	17110	717-528-8...	717-632-5...	ahmed
Aja	Gehrett	Stero Company	993 Washin...	Nutley	Essex	NJ	07110	973-544-2...	973-986-4...	aja_gé
Alaine	Bergesen	Hispanic Magazine	7667 S Hul...	Yonkers	Westchester	NY	10701	914-300-9...	914-654-1...	alaine
Albina	Glick	Giampetro, Antho...	4 Ralph Ct	Dunellen	Middlesex	NJ	08812	732-924-7...	732-782-6...	albina
Alease	Buemi	Porto Cayo At Ha...	4 Webbs C...	Boulder	Boulder	CO	80303	303-301-4...	303-521-9...	alease
Alecia	Bubash	Petersen, James E...	6535 Joyce...	Wichita Falls	Wichita	TX	76301	940-276-7...	940-302-3...	alecia
Alesia	Hixenbaugh	Kwikprint	9 Front St	Washington	District of C...	DC	20001	202-646-7...	202-276-6...	alesia
Alex	Loader	Sublett, Scott Esq	37 N Elm St...	Tacoma	Pierce	WA	98409	253-660-7...	253-875-9...	alex@
Alisha	Slusarski	Wtlz Power 107 Fm	3273 State St	Middlesex	Middlesex	NJ	08846	732-658-3...	732-635-3...	alisha
Alishia	Sergi	Milford Enterprise...	2742 Distri...	New York	New York	NY	10025	212-860-1...	212-753-2...	asergi
Aliza	Baltimore	Andrews, J Robert ...	1128 Delaw...	San Jose	Santa Clara	CA	95132	408-504-3...	408-425-1...	aliza@
Allene	Iturbide	Ledecky, David Esq	1 Central Ave	Stevens Point	Portage	WI	54481	715-662-6...	715-530-9...	allene
Alline	Jeanty	W W John Holden ...	55713 Lake...	South Bend	St Joseph	IN	46601	574-656-2...	574-405-1...	ajeant
Alpha	Palaia	Stoffer, James M Jr	43496 Com...	Cherry Hill	Camden	NJ	08003	856-312-2...	856-513-7...	alpha
Alyce	Arias	Fairbanks Scales	3196 S Rid...	Stockton	San Joaquin	CA	95207	209-317-1...	209-242-7...	alyce@
Amber	Monarrez	Branford Wire & M...	14288 Fost...	Jenkintown	Montgomery	PA	19046	215-934-8...	215-329-6...	amber

9. 데이터 소스 관리 홈 화면으로 이동하면 생성된 데이터 소스를 화면에서 확인할 수 있습니다. 데이터 적재가 수행되는 중에는 아래와 같이 상태가 **Disabled**로 표시되게 되고 적재가 완료되면 **Enabled**로 변경됩니다. 이때부터 데이터 소스를 사용할 수 있습니다.

데이터소스 데이터 커넥션

+ 상태: 전체 공개: 전체 생성된 시점: 전체 일련번호 시점: 전체 검색

148개 데이터가 있습니다 새로만들기

데이터소스	소스 타입	적재 타입	상태	생성일
us 500 오픈 데이터	파일	수집형 데이터	Enabled	2019-01-21 13:26 by Administrator

DB로 데이터 소스 만들기

외부 데이터베이스에서 데이터를 가져와서 데이터 소스를 생성합니다.

1. 원천 데이터 타입 선택 화면에서 **데이터베이스**를 선택합니다.
2. 연결할 데이터베이스의 정보를 입력합니다.

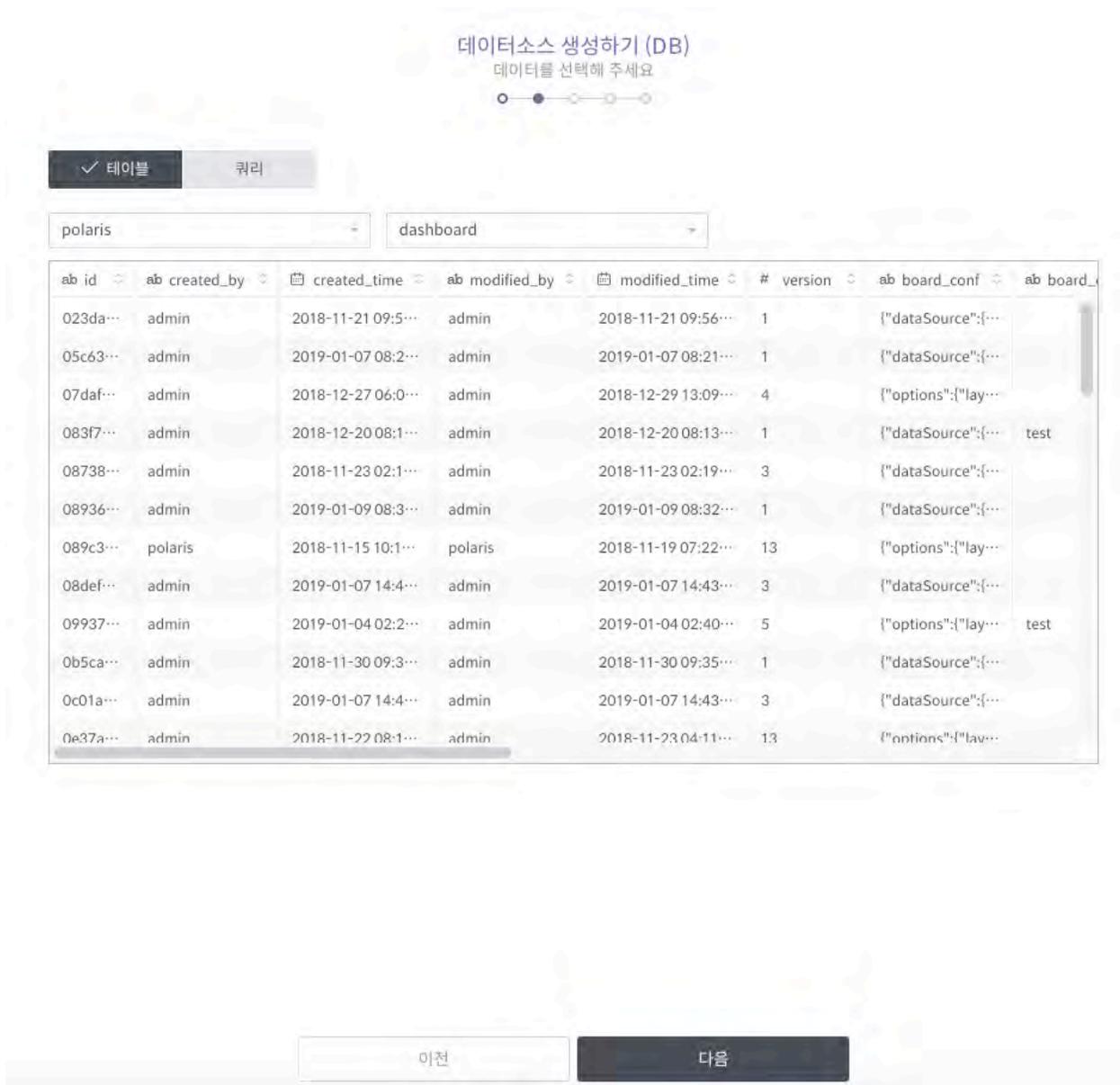
- **수집 타입**: 데이터 소스가 데이터를 수집하는 방식을 선택합니다.
 - **수집형 데이터 (Ingested)**: 데이터를 Metatron 서버에 직접 저장하는 방식으로 수집된 데이

터 소스들이 출력됩니다.

- **연결형 데이터 (Linked):** 연결된 데이터베이스에서 필요한 시점마다 데이터를 가져오는 방식의 데이터 소스들이 출력됩니다.

- **데이터 커넥션 로드:** 기존에 등록된 데이터 커넥션에 연결되어 있는 데이터베이스의 접근 정보를 자동으로 불러올 수 있습니다. 단, 이때도 **유효성 체크** 버튼을 눌러서 연결 검증은 반드시 실시해야 합니다.
- **DB 타입:** 연결할 데이터베이스의 타입을 선택합니다.
- **Host:** 연결할 호스트 값을 입력합니다.
- **Port:** 연결할 포트 번호를 입력합니다.
- **사용자이름:** 해당 데이터베이스의 username을 입력합니다.
- **비밀번호:** 해당 데이터베이스의 비밀번호를 입력합니다.
- **유효성 체크:** 모든 입력 항목을 다 작성하면 테스트 버튼이 활성화 됩니다. 클릭하면 커넥션이 정상적인지 여부가 버튼 하단에 나타납니다. 정상적이라면 **유효한 커넥션**, 비정상적이라면 **잘못된 커넥션**이라는 문구가 나타납니다.

3. 데이터를 선택합니다. 연결된 데이터베이스 계정에서 테이블을 선택할 수도 있고 쿼리문을 직접 작성할 수도 있습니다.



- **테이블**: 데이터베이스와 테이블명을 선택한 후 실제 저장될 데이터가 조회되면, 확인 후 **다음** 버튼을 누릅니다.
 - **쿼리**: 원하는 데이터를 가져올 수 있는 쿼리문을 직접 작성하고 **실행** 버튼을 클릭하면 하단에 데이터가 보여집니다. 데이터를 확인한 후 **다음** 버튼을 누르십시오.
4. 이후 절차는 **파일로 데이터 소스 만들기** 항목과 동일합니다. 단, 데이터베이스로부터 데이터 소스를 생성할 경우 수집 설정 시 아래와 같이 **적재 설정** 항목을 추가로 설정해야 합니다.

데이터소스 생성하기 (DB)
수집 설정을 완료해 주세요

○ ○ ○ ● ○

적재 설정

1회 수집 주기적 수집

수집 데이터 범위

전체 최대 건수 10000 행

타임스탬프 설정

세그먼트 단위

시간

쿼리 단위

초

물업

설정함 설정안함

고급 설정 ▾

이전 다음

- **1회 수집**: 현재 데이터베이스에 있는 데이터를 이번 한번만 적재합니다. **최대 건 수**를 선택할 경우 제 1행부터 몇 번째 행까지 적재할지 지정할 수 있습니다.

적재 설정

1회 수집 주기적 수집

수집 데이터 범위

전체 최대 건수 10000 행

- **주기적 수집**: 기간을 두어 데이터 저장을 주기적으로 실행합니다.

적재 설정

1회 수집 주기적 수집

추첨 데이터 범위

증감분 데이터 전체

배치 주기

매시

최대 행 수

StagingDB로 데이터 소스 만들기

Metatron의 내부 Hive 데이터베이스에서 가져온 데이터를 기반으로 데이터 소스를 생성합니다.

1. 원천 데이터 타입 선택 화면에서 **Staging DB**를 선택합니다.
2. 연결할 데이터베이스와 테이블을 선택하면 데이터가 출력됩니다.

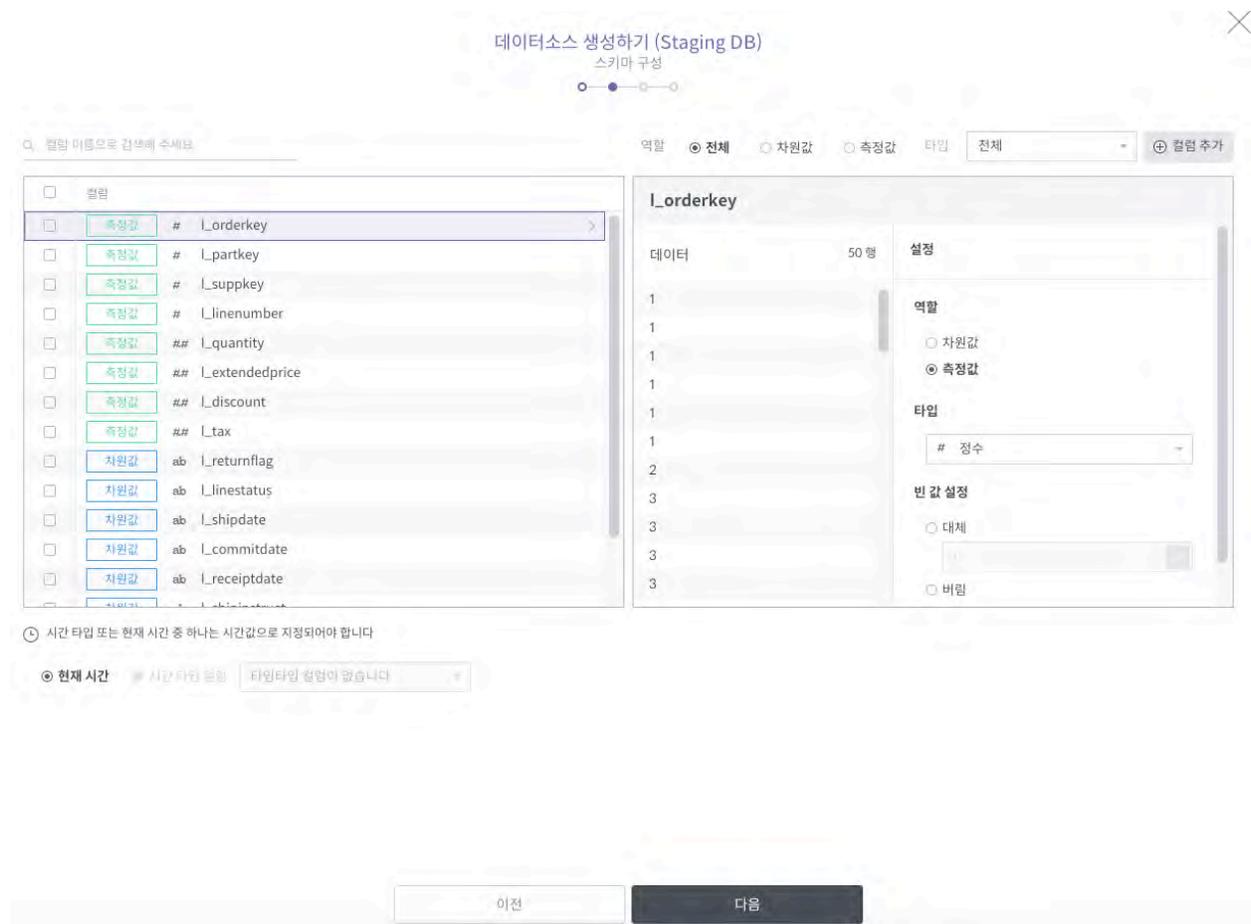
데이터소스 생성하기 (Staging DB)
데이터를 선택해 주세요

● ○ ○ ○ ○

tpch_10 lineitem

#_L_orderkey	#_L_partkey	#_L_suppkey	#_L_linenumber	##_L_quantity	##_L_extendedprice	##_L_discount	##_
1	1551894	76910	1	17	33078.94	0.04	
1	673091	73092	2	36	38306.16	0.09	
1	636998	36999	3	8	15479.68	0.1	
1	21315	46316	4	28	34616.68	0.09	
1	240267	15274	5	24	28974	0.1	
1	156345	6348	6	32	44842.88	0.07	
2	1061698	11719	1	38	63066.32	0	
3	42970	17971	1	45	86083.65	0.06	
3	190355	65359	2	49	70822.15	0.1	
3	1284483	34508	3	27	39620.34	0.06	
3	293797	18800	4	2	3581.56	0.01	
3	1830941	5996	5	28	52411.8	0.04	

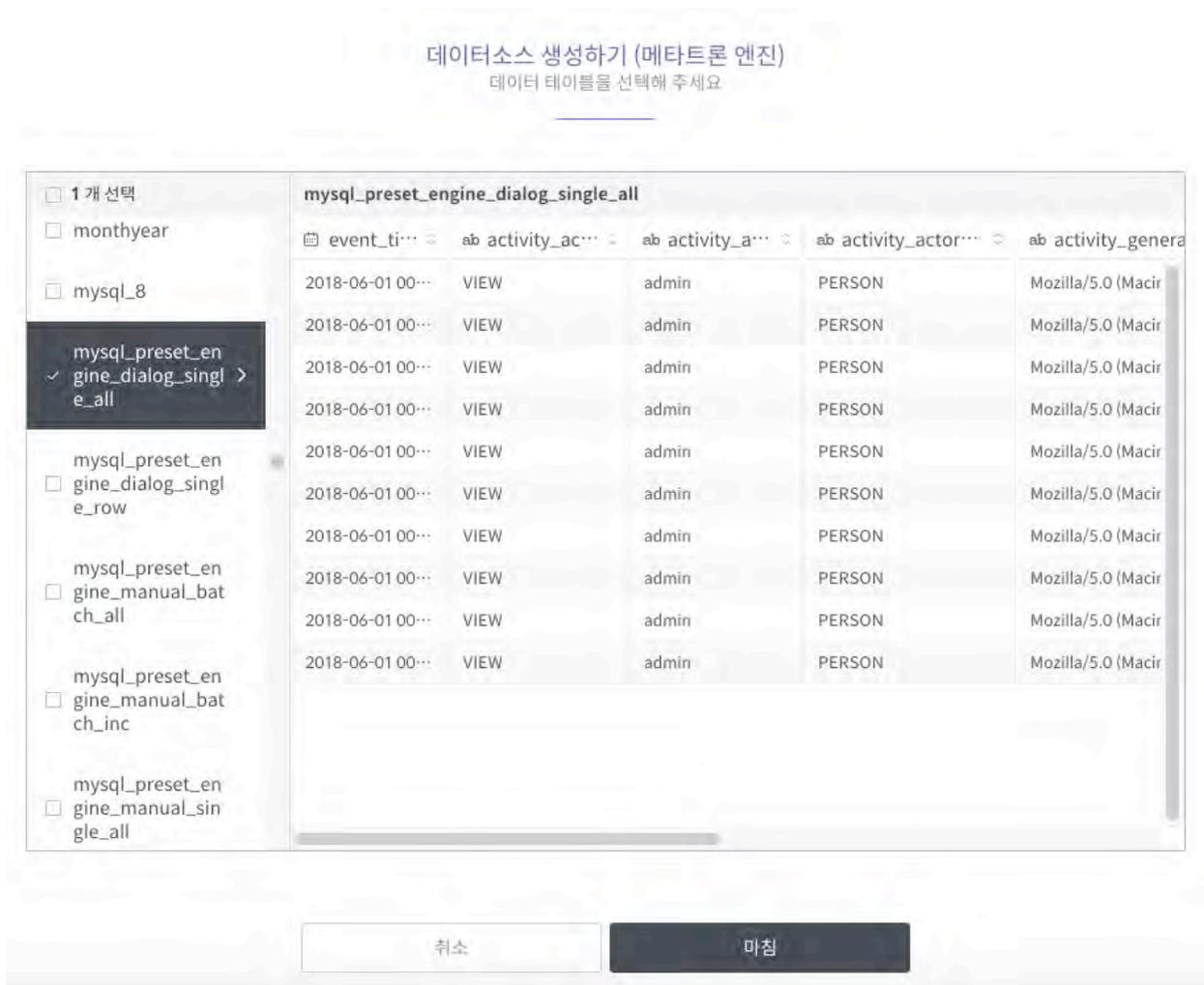
3. 이후 절차는 DB로 데이터 소스 만들기 항목과 동일합니다.



Metatron 엔진을 통해 데이터 소스 추가하기

Metatron 이전 버전에 저장된 데이터 소스를 마이그레이션합니다.

1. 원천 데이터 타입 선택 화면에서 **Metatron 엔진**을 선택합니다.
2. 아래와 같이 이전 버전의 Metatron에서 만든 데이터 소스가 좌측 화면에 나열되면, 그 중에서 현 버전으로 마이그레이션하고자 하는 데이터 소스들의 확인란에 체크합니다.



3. 마침 버튼을 누르면 선택한 데이터 소스들이 마이그레이션됩니다.

mysql_preset_engine_dialog_single_all 메타트론 엔진 수집형 데이터 Enabled 2019-01-21 13:35 by Administrator

3.2 데이터 커넥션

Metatron Discovery는 외부 데이터베이스를 직접 연결하는 기능을 지원합니다. 외부 데이터베이스를 연결하려면 해당 데이터베이스에 대한 접근 정보가 담긴 데이터 커넥션을 생성·관리해야 합니다. 데이터 커넥션을 등록해 두면 새로운 데이터베이스 접속 정보를 다시 입력해야 하는 수고를 덜 수 있습니다.

데이터 커넥션 메뉴는 메인 화면 좌측 패널에서 MANAGEMENT > 데이터 스토리지 > 데이터 커넥션을 통해 진입할 수 있습니다.



3.2.1 데이터 커넥션 관리 홈 화면

데이터 커넥션 화면에서는 데이터베이스 커넥션의 신규 등록·편집·조회가 가능합니다.

The screenshot shows the '데이터 스토리지' (Data Storage) section of the Metatron Discovery interface. It features a navigation bar with '데이터 커넥션' (Data Connections) selected. Below the navigation bar, there are filter tabs for '공개: 전체', '생성한 사람: 전체', 'DB 타입: 전체', '보안: 전체', and '생성한 시간: 전체'. A search bar is also present. The main content area displays a table with 4 data connections. The table has the following data:

데이터 커넥션	DB 타입	Host/Port(URL)	생성일
하이브	Hive	metatron-hadoop-04 / 10000	2018-12-05 12:15 by Administrator
Hive-metatron-hadoop-04-10000	Hive	metatron-hadoop-04 / 10000	2018-12-04 18:17 by Administrator
MySQL-test	MySQL	metatron-web-03 / 3306	2018-12-04 13:51 by Administrator
MySQL-metatron-web-03-3306	MySQL	metatron-web-03 / 3306	2018-11-22 15:09 by Administrator

- **공개:** 데이터 커넥션을 공개 워크스페이스별로 선별하여 조회합니다.
- **생성한 사람:** 데이터 커넥션을 생성한 사람별로 선별하여 조회합니다.
- **DB 타입:** 데이터 커넥션을 DB 타입 (MySQL, PostgreSQL, Hive, Presto) 별로 선별하여 조회합니다.
- **보안:** 데이터 커넥션을 보안 유형 (항상 연결, 사용자 계정, 아이디와 비밀번호) 별로 선별하여 조회합니다.
- **생성한 시간:** 데이터 커넥션을 생성한 시간 (오늘, 지난 7일, 사용자설정 기간) 별로 선별하여 조회합니다.
- **검색:** 데이터 커넥션을 데이터 커넥션의 이름으로 검색하여 조회합니다.
- **데이터 커넥션 개수:** 현재 목록에 조회된 데이터 커넥션의 개수를 나타냅니다.
- **새로 만들기:** 클릭하면 새로운 데이터 커넥션을 생성할 수 있습니다.
- **삭제:** 데이터 커넥션에 마우스 오버 시 휴지통 아이콘이 나타납니다. 클릭하면 해당 데이터 커넥션을 삭제할 수 있습니다.

3.2.2 데이터 커넥션 만들기

데이터커넥션 생성하기 화면에서는 커넥션 생성에 필요한 정보를 입력하여 커넥션을 생성합니다.

데이터커넥션 생성하기
데이터 연결 설정을 입력해 주세요

DB 커넥션

MySQL ✓

PostgreSQL

Hive

Presto

Host	Port
Host	Port
<input type="checkbox"/> URL 만	
사용자이름	비밀번호
사용자이름을 입력해 주세요	비밀번호를 입력해 주세요
<p>보안</p> <input checked="" type="radio"/> 항상 연결 <input type="radio"/> 사용자의 계정으로 연결 <input type="radio"/> 아이디와 비밀번호로 연결	
<div style="border: 1px solid #ccc; padding: 2px 10px; display: inline-block;">유효성 체크</div>	
고급설정	

할당

1 워크스페이스 [선택](#)

모든 워크스페이스에서 이 데이터커넥션 사용을 허용

취소

마침

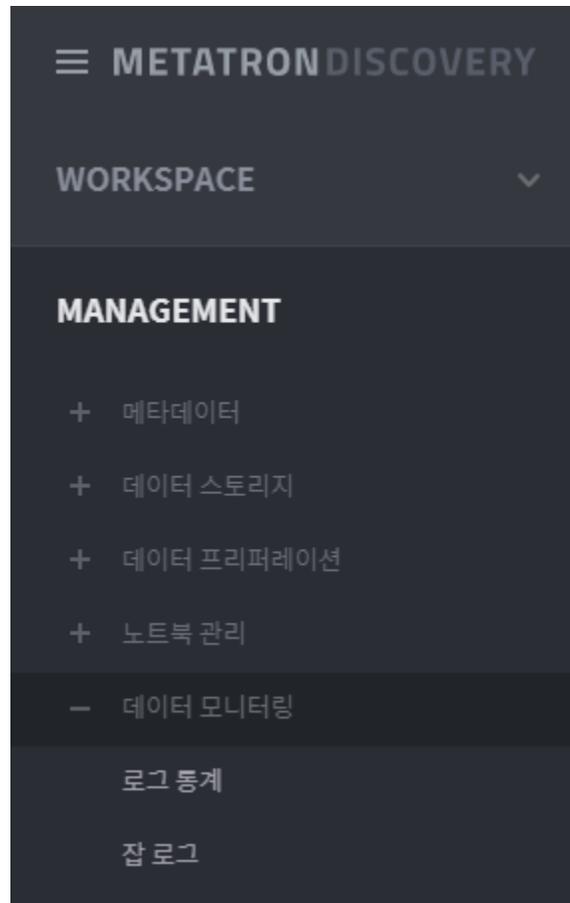
- **DB 타입:** 현재 총 4개 타입의 데이터베이스를 지원합니다. (MySQL, PostgreSQL Hive, Presto)
- **Host:** 연결할 호스트 값을 입력합니다.
- **Port:** 연결할 포트 번호를 입력합니다.
- **URL 만:** Host, Port 대신 DB URL을 입력합니다.
- **사용자이름:** 데이터베이스의 사용자 이름을 입력합니다.
- **비밀번호:** 데이터베이스의 비밀번호를 입력합니다.

- **보안:** 데이터 커넥션을 이용할 때 적용할 보안 방식을 설정합니다.
 - **항상 연결:** 데이터 커넥션 생성 시 사용자가 직접 입력한 정보를 사용하여 로그인합니다.
 - **사용자의 계정으로 연결:** Metatron Discovery에 등록되어 있는 사용자 계정 정보를 사용하여 로그인합니다.
 - **아이디와 비밀번호로 연결:** 데이터 커넥션을 사용할 때마다 계정 정보를 입력 받아서 로그인합니다.
- **유효성 체크:** 입력한 커넥션 정보가 유효한지 검사하며, 그 결과가 버튼 옆에 나타납니다. 정상적이라면 **유효한 커넥션**, 비정상적이라면 **잘못된 커넥션**이라는 문구가 나타납니다.
- **고급설정:** 옵션으로 커스텀 프로퍼티 키와 값을 추가할 수 있습니다.
- **할당:** 생성할 데이터 커넥션의 사용을 허용할 워크스페이스를 지정합니다.
 - **모든 워크스페이스에서 이 데이터 커넥션 사용을 허용:** 이 확인란에 체크하면 모든 워크스페이스에서 해당 데이터 커넥션을 사용할 수 있습니다.
 - **수정:** 해당 데이터 커넥션의 사용을 허용할 특정 워크스페이스를 지정할 때 사용합니다. 해당 데이터 커넥션을 오픈 데이터로 지정할 경우 이 버튼이 사라집니다.
 - **공유 워크스페이스 수:** 해당 데이터 커넥션을 사용할 수 있도록 허용된 워크스페이스의 수를 나타냅니다.

3.3 데이터 모니터링

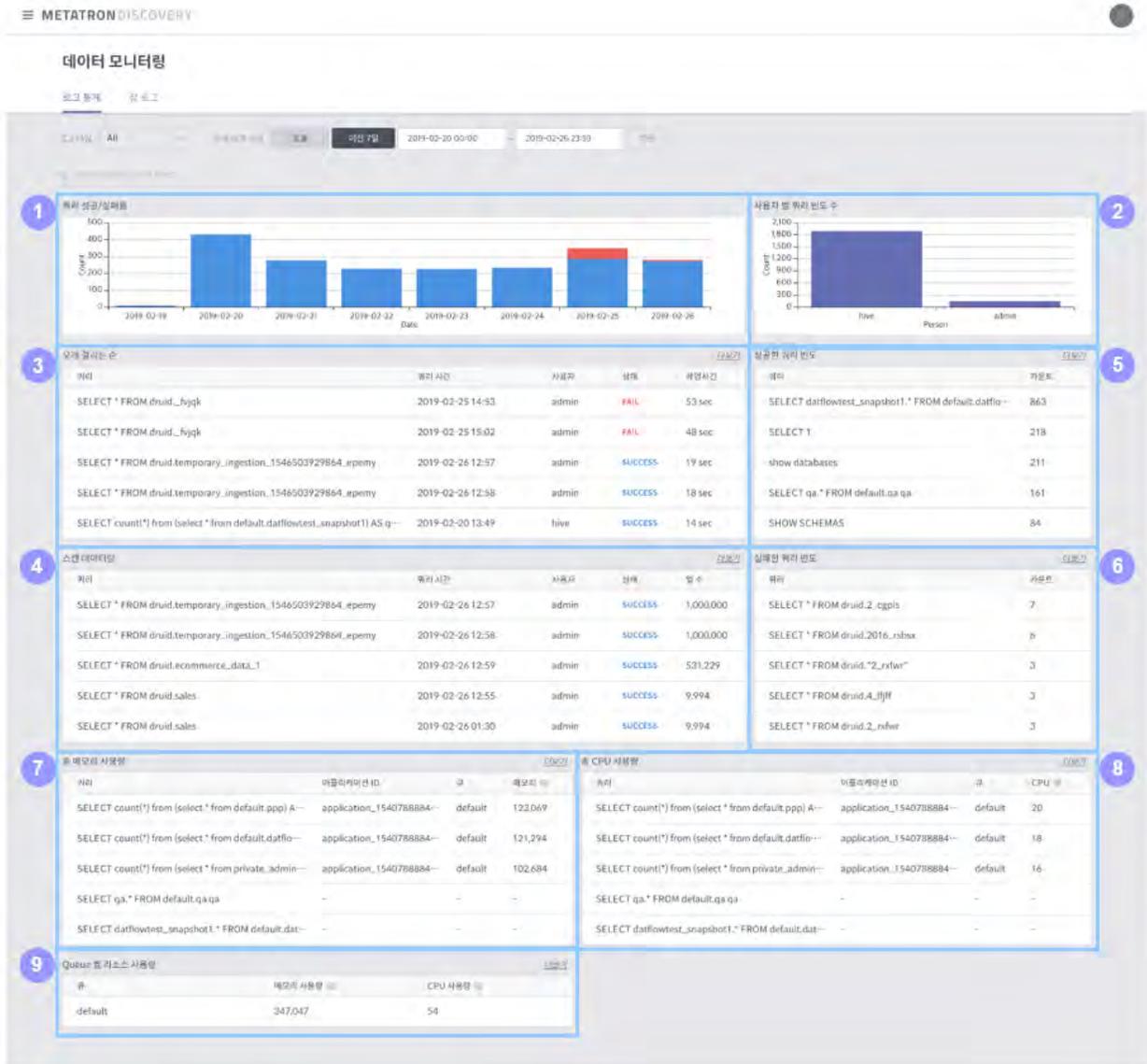
데이터 모니터링은 Metatron Workbench에서 유저가 Staging DB(내부 Hive DB) 및 Metatron과 연결된 외부 데이터베이스에 질의하는 모든 로그를 관측하는 기능입니다.

데이터 모니터링 메뉴는 메인 화면 좌측 패널에서 **MANAGEMENT > 데이터 스토리지 > 데이터 모니터링**을 통해 진입할 수 있습니다.



3.3.1 로그 통계

로그 통계에서는 Metatron Discovery 내 쿼리 수행과 관련된 각종 통계치를 모아 보여줍니다. 여기에서는 아래와 같이 총 9가지의 기본 통계를 조회할 수 있습니다.

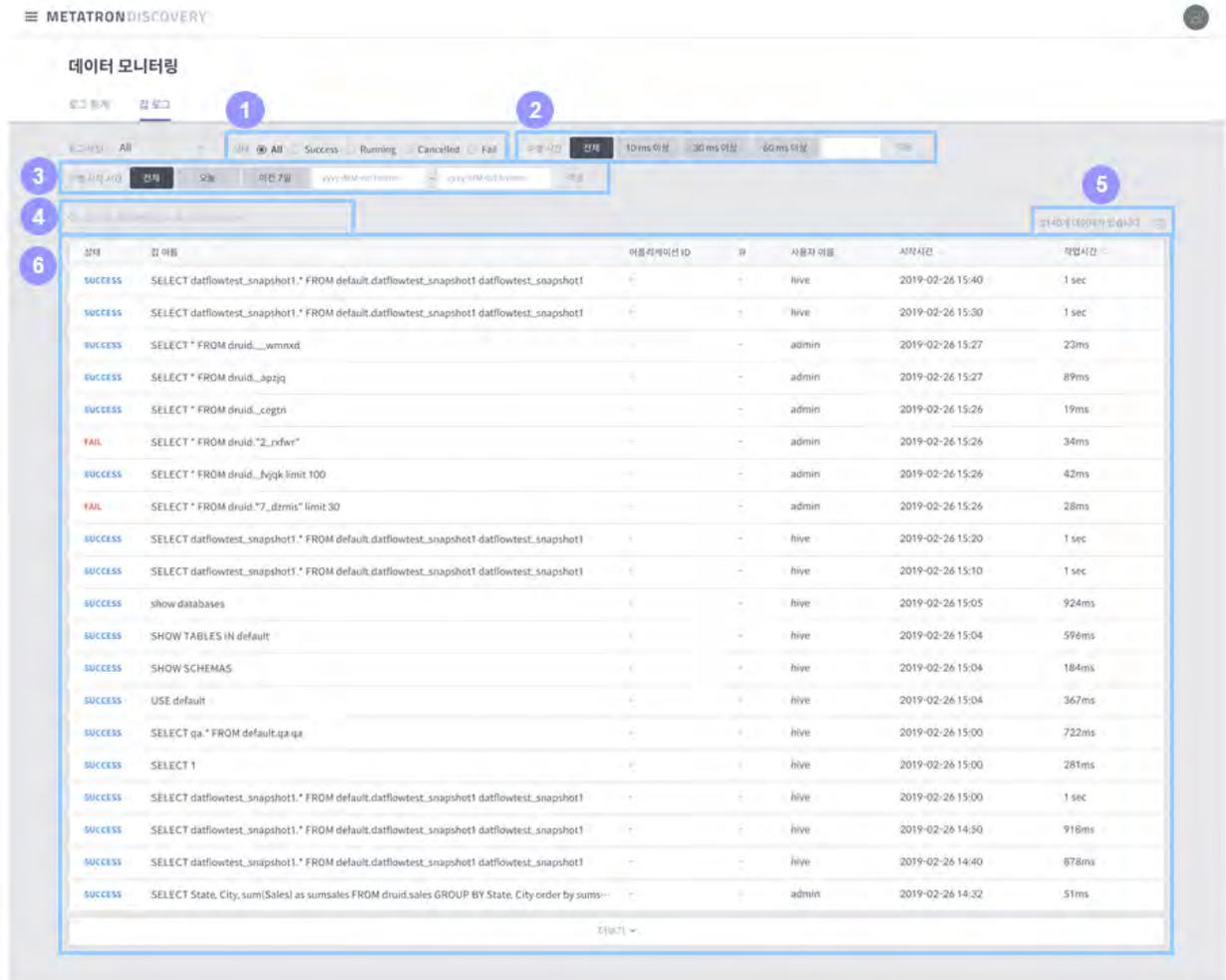


1. 쿼리 성공/실패율: Metatron에서 실행된 쿼리들의 성공률과 실패율이 나타납니다.
2. 사용자 별 쿼리 빈도 수: 쿼리를 수행한 사용자별 빈도수를 나타낸 그래프입니다. 표시된 막대 중 하나를 클릭하면 해당 사용자가 실행한 Job Log를 볼 수 있습니다.
3. 오래 걸리는 순: 수행한 쿼리들이 작업시간이 긴 순서대로 정렬되어 나타납니다.
4. 스캔 데이터량: 수행한 쿼리들이 데이터를 제일 많이 스캔한 순서대로 정렬되어 나타납니다.
5. 성공한 쿼리 빈도: 수행한 쿼리들이 성공한 빈도가 높은 순서대로 정렬되어 나타납니다.
6. 실패한 쿼리 빈도: 수행한 쿼리들이 실패한 빈도가 높은 순서대로 정렬되어 나타납니다.
7. 총 메모리 사용량: 수행한 쿼리들이 총 메모리 사용량이 큰 순서대로 정렬되어 나타납니다.

- 8. 총 CPU 사용량: 수행한 쿼리들이 총 CPU 사용량이 큰 순서대로 정렬되어 나타납니다.
- 9. Queue별 리소스 사용량: Hadoop 환경의 각 YARN queue에서 소모되는 리소스량을 보여줍니다.

3.3.2 잡 로그

본 메뉴에서는 Metatron에서 수행된 모든 쿼리의 내역을 조회할 수 있습니다. 쿼리 이력을 원하는 조건으로 검색하여 기존에 발생한 job 이력을 손쉽게 찾아볼 수 있습니다. 아래는 검색이 가능한 job 조건들입니다.



- 1. 상태: 수행된 쿼리들을 성패 기준으로 선별하여 조회합니다.
- 2. Limited elapsed time: 수행 시간이 오래 소요된 쿼리들을 선별하여 조회합니다. 기준 시간은 원하는 대로 설정 가능합니다.
- 3. Performed start Time: 쿼리 조회 시 적용되는 시간 기준입니다. 여기서의 시간은 각 쿼리가 수행을 시작하는 시간을 기준으로 합니다.

4. **Job 또는 어플리케이션으로 검색:** 현재 이력으로 남은 쿼리들을 쿼리문 또는 Application ID로 검색합니다.
5. **데이터 개수:** 현재 목록에 조회된 쿼리의 개수를 나타냅니다.
6. **Job 목록:** 설정한 선별 조건에 맞는 쿼리들을 보여줍니다. 이 중 하나를 클릭하면 상세 내역을 볼 수 있습니다.

쿼리 상세 정보

잡 로그 홈에 열거된 쿼리 중 하나를 클릭하면 해당 쿼리에 관한 다양한 정보와 이력을 확인할 수 있습니다. 상세 내역에서 조회 가능한 정보는 다음과 같습니다.

로그 정보

1	상태	SUCCESS
2	잡 이름	druid-hadoop
3	쿼리문	SELECT * FROM druid temporary_ingestion_1546503929864_epemv
4	시작시간	2019-02-26 12:57
5	작업시간	19 sec
6	사용자	admin

쿼리 정보

7	이름	DRUID
8	Host	metatron-hadoop-02
8	Port	8082
8	JDBC URL	jdbc:avatica:remoteurl=http://metatron-hadoop-02:8082/druid/v2/sql/avatica/

쿼리 이력

쿼리 시간	사용자	작업시간	상태
2019-02-26 15:27	Administrator	23 ms	SUCCESS
2019-02-26 15:27	Administrator	89 ms	SUCCESS
2019-02-26 15:26	Administrator	19 ms	SUCCESS
2019-02-26 15:26	Administrator	34 ms	FAIL
2019-02-26 15:26	Administrator	42 ms	SUCCESS

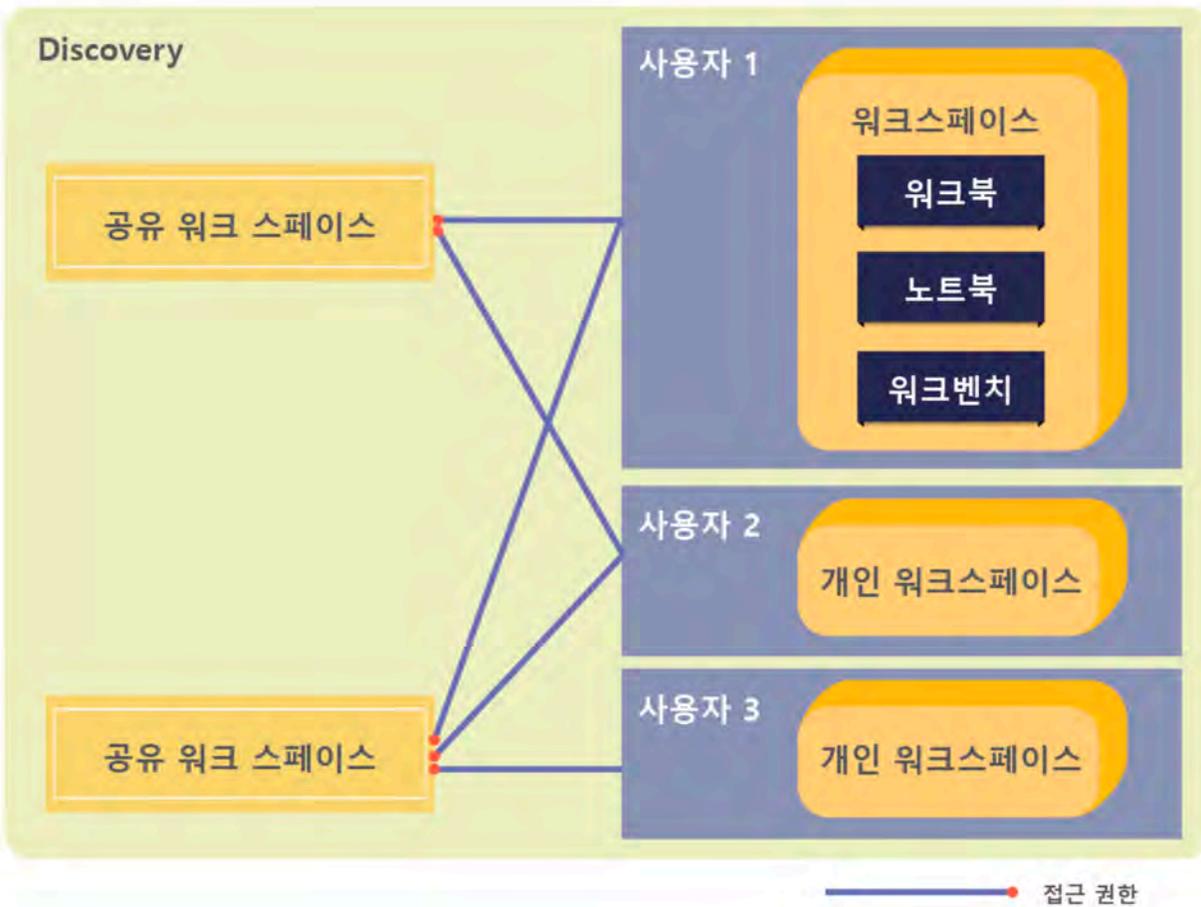
필터

1. **상태:** 해당 쿼리의 성공 여부를 나타냅니다.
2. **잡 이름:** 수행된 쿼리문입니다.
3. **시작시간:** 해당 쿼리가 수행되기 시작한 시간을 나타냅니다.
4. **작업시간:** 해당 쿼리가 수행되는 데 걸린 시간을 나타냅니다.
5. **사용자:** 해당 쿼리를 수행한 사용자 ID입니다.

6. **커넥션**: 워크벤치에서 실행된 쿼리일 경우, 대상 데이터 커넥션의 정보를 나타냅니다.
7. **동일 커넥션의 최근 사용 기록**: 워크벤치에서 실행된 쿼리일 경우, 해당 데이터베이스에서 수행된 최근 5건의 쿼리 내역과 그 결과가 나타납니다. Detail을 클릭하면 해당 쿼리문이 새 창에 출력됩니다.
8. **플랜**: 쿼리 수행 계획을 실행합니다.

CHAPTER 4

워크스페이스



워크스페이스는 Metatron Discovery의 분석 모듈인 워크북, 노트북, 워크벤치를 보관하는 작업 공간으로, 개인용 워크스페이스와 공유용 워크스페이스로 분류됩니다.

- **개인 워크스페이스:** Discovery 회원별로 하나씩 할당되는 개인용 워크스페이스입니다. 이 워크스페이스는 본인만 접근 가능합니다.
- **공유 워크스페이스:** 여러 사용자가 함께 사용할 수 있는 워크스페이스입니다. 분석 과정과 결과를 다른 사용자들과 공유하기 위한 공간입니다. 각 공유 워크스페이스의 소유자 또는 관리자는 Discovery 회원들에게 다양한 수준의 접근 권한을 부여할 수 있습니다.

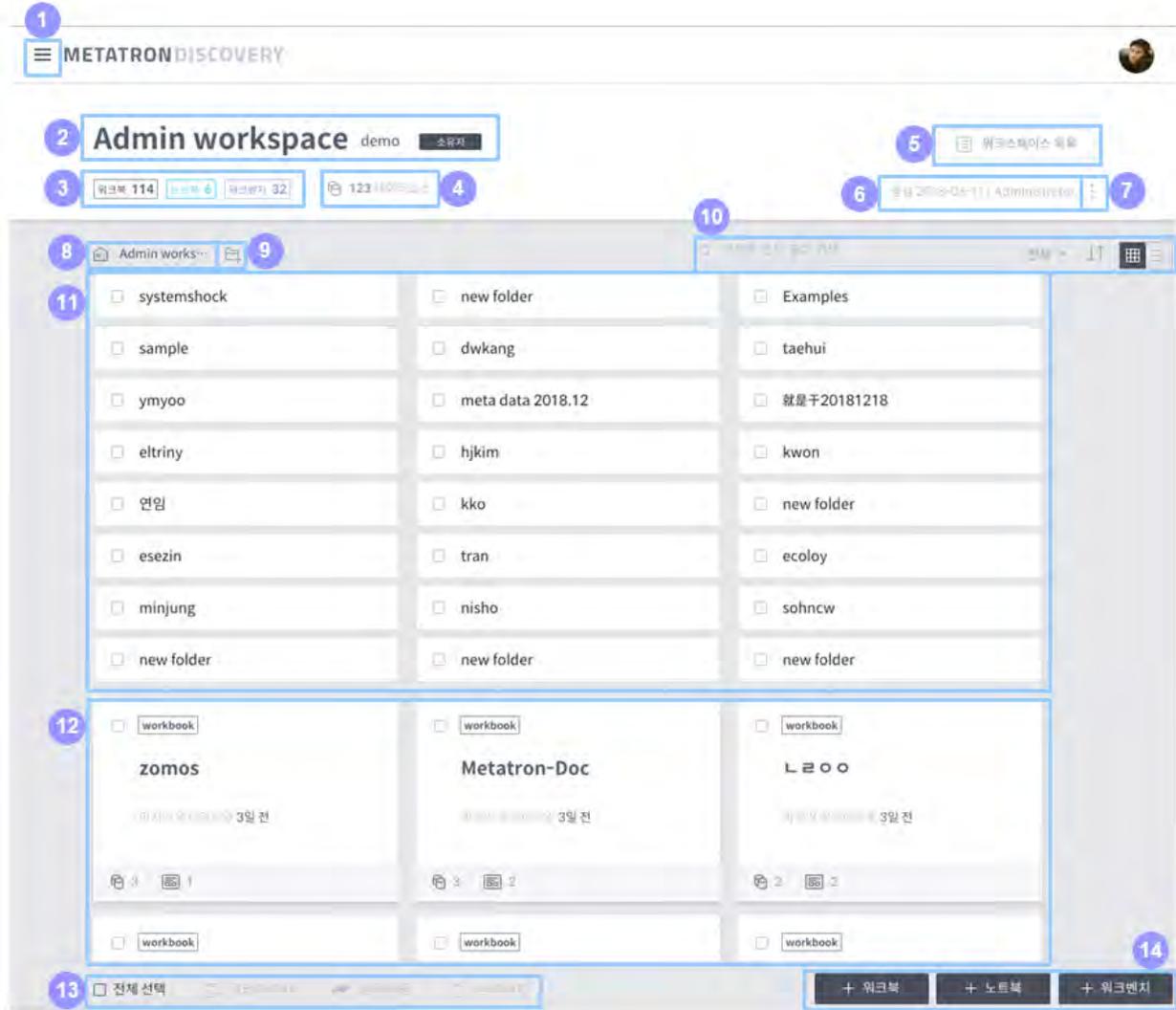
본 단원에서는 워크스페이스 홈 화면 구성과 UI, 그리고 공유 워크스페이스를 활용하는 방식에 대해 소개합니다.

4.1 워크스페이스 홈

워크스페이스 홈 화면에서는 Metatron Discovery의 분석 모듈 (워크북, 노트북, 워크벤치) 의 관리 기능을 수행할 수 있습니다.

4.1.1 워크스페이스 홈 화면 구성

아래는 워크스페이스 홈 화면의 전반적인 구성을 설명한 것입니다.



1. **메인 메뉴 버튼:** 해당 버튼을 클릭하면 다른 워크스페이스로 접근할 수 있는 패널이 열립니다.
2. **워크스페이스 정보:** 해당 워크스페이스의 이름과 설명을 보여줍니다. 로그인한 사용자가 소유하는 워크스페이스일 경우, 워크스페이스 이름 옆에 <소유자> 라는 아이콘이 보입니다.
3. **등록 개체 현황:** 워크스페이스 내에 등록된 개체 타입별 개수를 보여줍니다.
4. **데이터 소스:** 해당 워크스페이스에서 사용 중인 데이터 소스의 개수를 보여주며, 이 영역을 클릭하면 사용 중인 데이터 소스들의 목록이 나타납니다.
5. **워크스페이스 목록:** 이 버튼을 클릭하면 공유 워크스페이스의 목록이 나타납니다. (자세한 내용은 공유 워크스페이스 목록 참조)
6. **생성 정보:** 해당 워크스페이스의 생성일과 만든 사용자 이름을 보여줍니다.
7. **더 보기:** 해당 워크스페이스의 정보를 수정합니다.

- **이름 및 설명 수정:** 해당 워크스페이스의 이름과 설명을 수정합니다.
 - **공유 회원 및 그룹 설정:** 해당 워크스페이스에 접근할 수 있는 사용자와 그룹을 지정합니다. (자세한 사항은 [공유 워크스페이스 접근 권한 설정](#) 참조)
 - **노트북 서버 설정:** 노트북 모듈이 사용하는 외부 분석 도구 서버 접근 정보를 설정합니다.
 - **권한 스키마 설정:** 해당 워크스페이스에서의 사용자 역할별 접근 권한을 설정합니다. (자세한 사항은 [공유 워크스페이스 접근 권한 설정](#) 참조)
 - **소유자 변경:** 해당 워크스페이스의 소유자를 바꿉니다.
 - **워크스페이스 삭제:** 해당 워크스페이스를 삭제합니다.
8. **워크스페이스 경로:** 워크스페이스 내에서의 현재 위치를 확인합니다. 경로에 나열된 상위 폴더 중 하나를 클릭하면 해당 폴더로 이동합니다.
 9. **폴더 생성:** 클릭하면 현재 위치에 새 폴더를 생성합니다.
 10. **개체 목록 선별/정렬:**
 - **검색:** 해당 워크스페이스 내에서 개체 또는 폴더를 이름으로 검색합니다.
 - **개체 타입:** 워크북, 노트북, 워크벤치 중 원하는 개체 타입만을 선별해서 조회합니다.
 - **정렬 순서:** 폴더 및 개체를 이름 또는 업데이트 시간 순서로 정렬합니다.
 - **뷰 형식:** 워크스페이스 내 개체들의 열거 방식을 카드뷰와 리스트뷰 중에서 선택합니다.
 11. **폴더 목록:** 현재 위치에서 검색 조건에 부합하는 폴더들을 보여줍니다. 이중 하나를 클릭하면 해당 폴더 안으로 이동합니다. (개별 폴더 항목에 대한 자세한 설명은 [폴더 항목](#) 참조)
 12. **개체 목록:** 현재 위치에서 검색 또는 선별 조건에 부합하는 개체들을 보여줍니다. 이 중 하나를 클릭하면 해당 개체의 홈 화면으로 이동합니다. (개별 개체 항목에 대한 자세한 설명은 [개체 항목](#) 참조)
 13. **개체 선택/복사/이동/삭제:** 개체 전체 선택, 복사, 이동, 삭제를 합니다. (자세한 내용은 [폴더 및 개체 선택/복사/이동/삭제](#) 참조)
 14. **개체 생성:** 해당 워크스페이스에서 원하는 타입의 개체를 만드는 데 사용되는 버튼들입니다. (구체적인 절차는 각각 [워크북 만들기](#), [신규 노트북 생성하기](#), [워크벤치 만들기](#) 참조)

4.1.2 폴더 항목

각 폴더에 마우스를 올렸을 때 다음과 같이 항목들이 표시됩니다.



- **확인란:** 해당 폴더를 선택할 때 사용됩니다. 선택한 폴더는 복제, 이동, 삭제할 수 있습니다.
- **이름:** 해당 폴더의 이름입니다.
- **수정:** 클릭하면 폴더의 이름을 수정할 수 있습니다. 이 버튼은 해당 폴더 항목에 마우스를 오버할 때만 나타납니다.
- **삭제:** 클릭하면 해당 폴더가 삭제됩니다. 이 버튼은 해당 폴더 항목에 마우스를 오버할 때만 나타납니다.

4.1.3 개체 항목

각 개체에 마우스를 올렸을 때 다음과 같이 항목들이 표시됩니다.

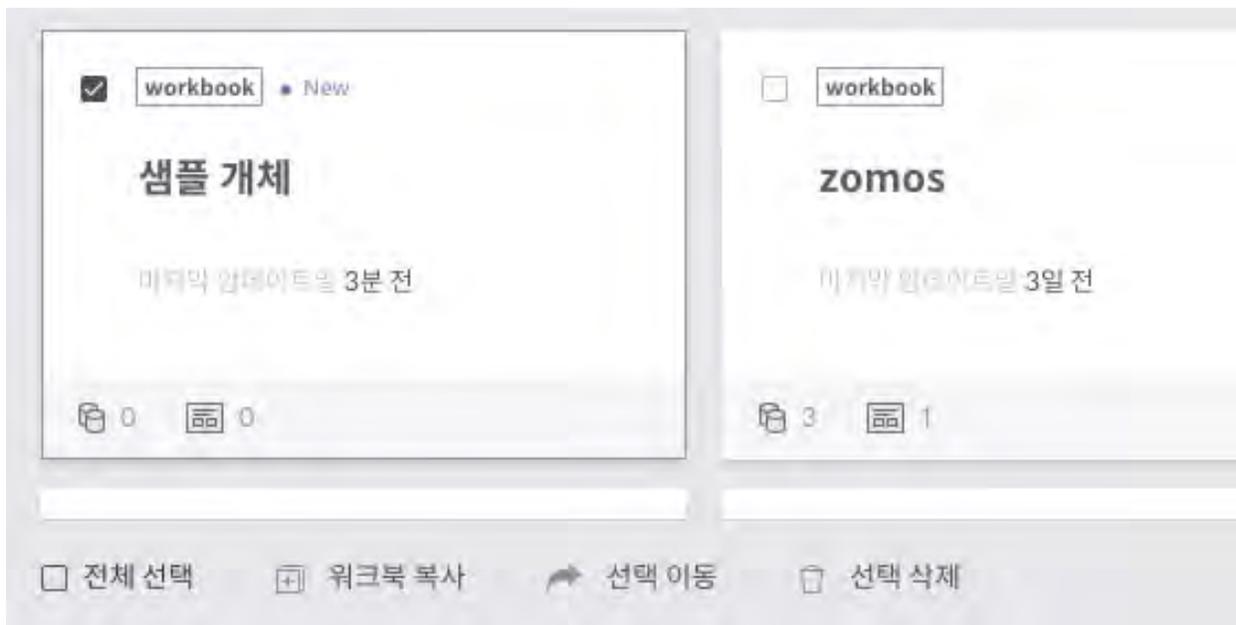


- **확인란:** 해당 개체를 선택할 때 사용됩니다. 선택한 개체는 복제, 이동, 삭제할 수 있습니다.
- **개체 타입:** 해당 개체가 워크북/노트북/워크벤치 중에서 어느 타입인지 보여줍니다.
- **삭제:** 클릭하면 해당 개체가 삭제됩니다. 이 버튼은 해당 개체 항목에 마우스를 오버할 때만 나타납니다.
- **이름:** 해당 개체의 이름입니다.
- **업데이트 시점:** 해당 개체가 마지막으로 업데이트된 시점을 보여줍니다.
- **데이터 소스/대시보드 개수:** 이 영역은 워크북 개체에만 고유합니다.

-  아이콘 옆 숫자는 해당 워크북에 연동된 데이터 소스의 개수를 가리킵니다.
-  아이콘 옆 숫자는 해당 워크북에 등록된 대시보드의 개수를 가리킵니다.

4.1.4 폴더 및 개체 선택/복사/이동/삭제

워크스페이스 내 폴더와 개체는 복사/이동/삭제가 가능합니다. 복사/이동/삭제를 원하는 폴더 또는 개체를 선택하면 워크스페이스 홈 화면 좌측 하단의 동작 버튼들이 활성화됩니다.



- **전체 선택:** 현재 폴더 목록과 개체 목록에 나타난 모든 항목을 선택합니다.
- **워크북 복사:** 워크북에만 유효한 기능입니다. 이 버튼을 클릭하면 선택한 워크북이 복제됩니다.
- **선택 이동:** 선택한 폴더 및 개체를 이동합니다. 워크북의 경우에는 다른 워크스페이스로 이동이 가능하고, 그 외의 항목들은 동일 워크스페이스 내 다른 폴더로 이동이 가능합니다. 워크북이 다른 개체와 함께 선택되어 있을 때는 이동을 실행할 수 없습니다.
- **선택 삭제:** 선택한 폴더 및 개체를 삭제합니다.

4.2 공유 워크스페이스

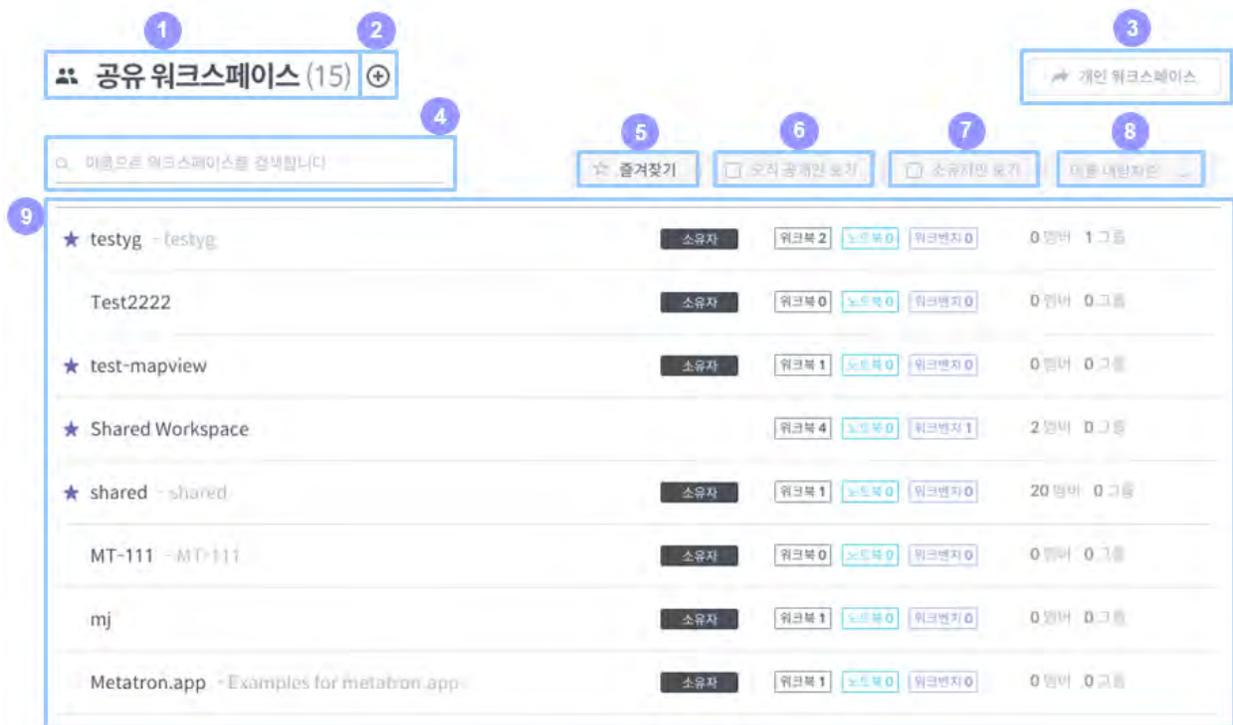
공유 워크스페이스는 여러 사용자가 함께 열람하고 사용하는 워크스페이스입니다. 아래 각 절에서는 공유 워크스페이스를 조회하고 생성하는 법을 설명하고, 공유 워크스페이스에 접근 가능한 사용자나 그룹을 설정할 수 있는 권한 스키마에 대해서 살펴봅니다.

4.2.1 공유 워크스페이스 목록

공유 워크스페이스 목록 화면에서는 로그인한 사용자가 접근할 수 있는 모든 공유 워크스페이스의 목록을 열람하고 원하는 워크스페이스로 이동할 수 있습니다. 이 화면은 다음과 같은 두 가지 방식으로 접근할 수 있습니다.

- Discovery 화면 좌측 상단에서 **버튼을 클릭하여 메인 패널을 연 후 워크스페이스 목록**을 클릭합니다.
- 워크스페이스 홈 화면 우측 상단의 **워크스페이스 목록** 버튼을 클릭합니다.

공유 워크스페이스 목록 화면은 다음과 같이 구성됩니다.



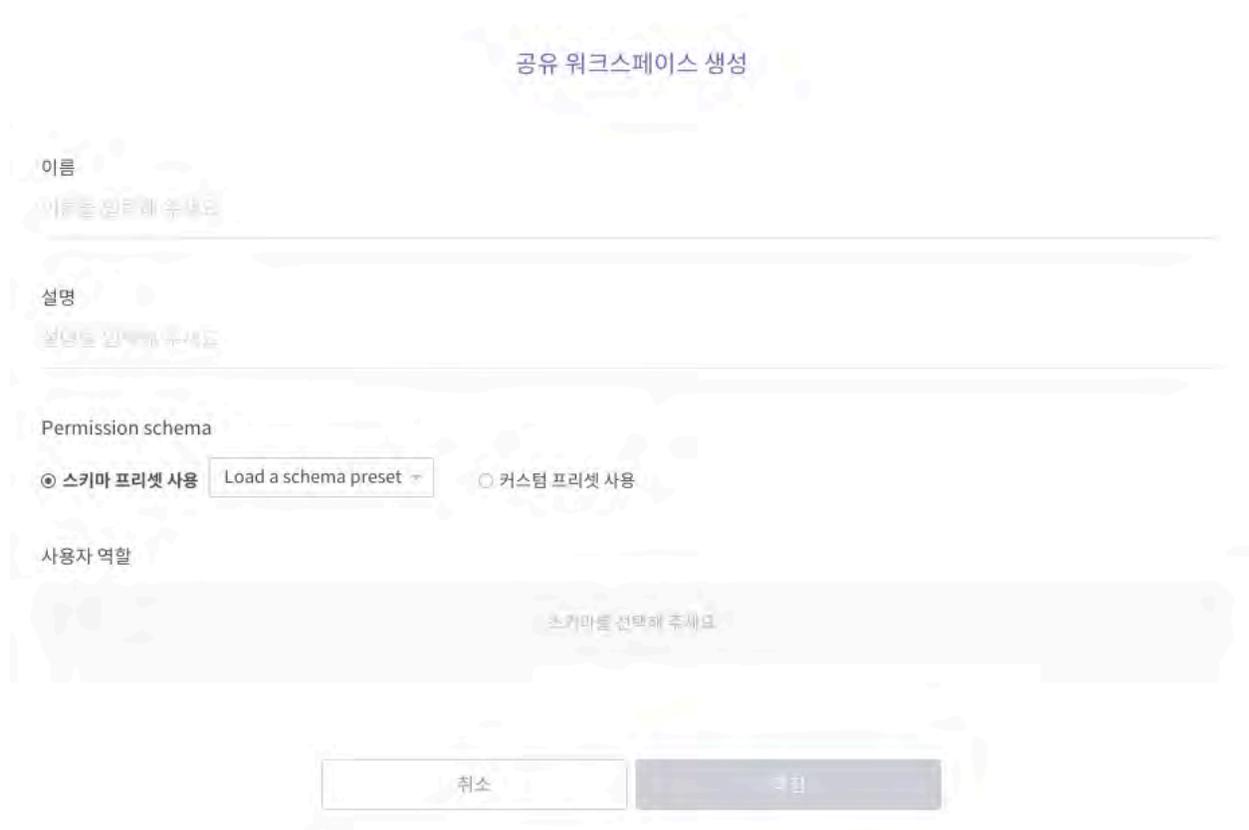
1. **공유 워크스페이스 개수**: 목록에 나열된 공유 워크스페이스의 개수가 나타납니다.
2. **공유 워크스페이스 추가**: 이 버튼을 클릭하면 공유 워크스페이스를 추가하는 화면으로 이동합니다.
(구체적인 절차는 [공유 워크스페이스 생성 참조](#))
3. **개인 워크스페이스**: 이 버튼을 클릭하면 로그인한 사용자의 개인 워크스페이스로 이동합니다.
4. **검색**: 공유 워크스페이스를 이름으로 검색합니다.
5. **즐거찾기**: 즐겨찾기로 지정된 워크스페이스만 선별합니다.
6. **오직 공개만 보기**: 전체 공개 설정된 워크스페이스만 선별합니다.
7. **소유자만 보기**: 로그인한 사용자가 관리자로 설정되어 있는 워크스페이스 목록이 나타납니다.

8. **이름 내림차순/오름차순:** 공유 워크스페이스 이름을 내림차순/오름차순으로 정렬합니다.
9. **워크스페이스 목록:** 설정한 선별 조건에 맞는 워크스페이스들을 보여줍니다. 이 중 하나를 클릭하면 해당 워크스페이스에 입장합니다.

4.2.2 공유 워크스페이스 생성

다음의 순서로 새 공유 워크스페이스를 생성할 수 있습니다.

1. 공유 워크스페이스 목록에서 **+** 버튼을 클릭하면 새 공유 워크스페이스를 생성하는 화면이 나타납니다.
2. **이름과 설명**을 입력한 후, 아래 설명을 참조하여 **Permission schema**를 설정합니다.



- **프리셋 스키마 사용:** 관리자가 기존에 정의해놓은 권한 스키마를 불러옵니다.
- **커스텀 스키마 사용:** 새 권한 스키마를 정의합니다. (새 권한 스키마를 정의하는 방식은 [공유 워크스페이스 접근 권한 설정 참조](#))

3. **마침** 버튼을 눌러 워크스페이스 생성을 완료합니다.

4.2.3 공유 워크스페이스 접근 권한 설정

공유 워크스페이스 접근 권한 설정은 기본적으로 다음과 같은 두 단계로 이루어집니다.

- 사용자 역할별 접근 권한 정의 (권한 스키마 설정 참조)
- 개별 사용자 또는 사용자 그룹 각각에게 적합한 사용자 역할 부여 (공유 멤버 및 그룹 설정 참조)

권한 스키마 설정

권한 스키마 조회하기

공유 워크스페이스 홈 화면 우측 상단에 있는 버튼을 클릭한 후, 권한 스키마 설정을 누르면 아래와 같이 현재 정의된 권한 스키마를 보여줍니다.

퍼미션 스키마 설정

User roles of shared

User role	Default role	Workbook			Notebook			Workbench			Workspace	
		View	Create	Edit any	View	Create	Edit any	View	Create	Edit any	Create folders	Set config.
Manager		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Editor		✓	✓	-	✓	✓	-	✓	✓	-	-	-
Watcher	●	✓	-	-	✓	-	-	✓	-	-	-	-
Guest		✓	-	-	-	-	-	-	-	-	-	-

Explanation

- Default role : Role to be granted when adding new members and groups
- View of (item) : Enable to access to item and to read contents
- Create of (item) : Enable to create, modify and delete items
- Edit any of (item) : Enable to create, modify and delete items which is created by other users
- Create folders : Enable to create, modify and delete folders
- Set config. : Enable to edit information and to set configuration of workspace

위 예시 그림에서는 사용자 역할 (User role) 로서 Manager, Editor, Watcher, Guest가 정의되어 있습니다. <권한 스키마>란, 이와 같이 각각 고유한 접근 권한이 정의된 사용자 역할들의 집합을 일컫는 말입니다.

각 사용자 역할에 대한 컬럼별 속성은 다음과 같습니다.

Default role

새롭게 추가되는 개별 사용자나 사용자 그룹에게는 Default role로 지정된 사용자 역할이 기본적으로 부여됨

워크북/노트북/워크벤치 개체 타입별 권한

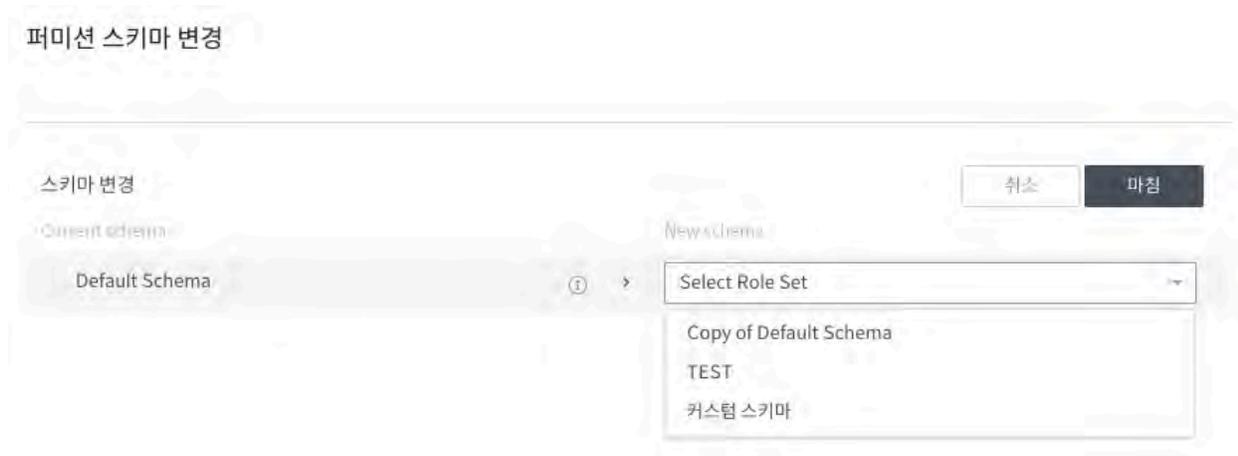
- **View:** 해당 타입의 개체에 접근하여 데이터를 열람할 수 있음
- **Create:** 해당 타입의 개체를 생성, 수정, 삭제할 수 있음
- **Edit any:** 다른 사용자가 생성한 해당 타입의 개체를 수정, 삭제할 수 있음

워크스페이스 권한

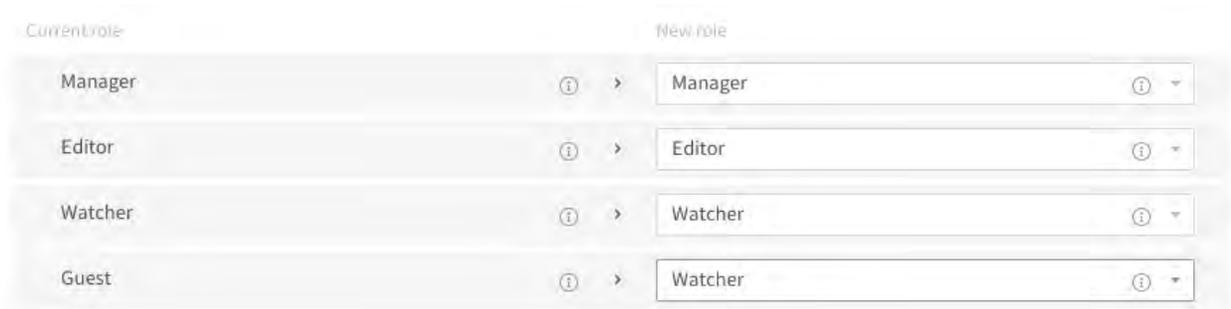
- Create folders: 워크스페이스 내 폴더를 생성, 수정, 삭제할 수 있음
- Set config.: 워크스페이스의 이름과 설명을 수정하고 워크스페이스 권한 스키마를 바꿀 수 있음

권한 스키마 설정 바꾸기

권한 스키마 조회 화면에서 **스키마 변경** 버튼을 클릭하면 아래와 같이 기존에 정의된 권한 스키마를 변경할 수 있는 화면이 나타납니다.



화면 우측에 있는 **Select Role Set** 콤보박스를 클릭하면 관리자가 정의해놓은 스키마들이 제시되고, 목록 맨 아래에는 새로운 사용자 역할을 정의할 수 있는 **커스텀 스키마** 항목이 있습니다. 이 중 하나를 선택하면 아래와 같은 화면이 나타납니다. (**커스텀 스키마** 항목을 선택했을 경우 사용자 역할별 권한부터 정의해야 합니다. New schema 우측의 버튼을 눌러 권한 설정 화면으로 이동한 후 **권한 스키마 조회하기** 항목의 설명을 참조하여 사용자 역할별로 권한을 설정하십시오.)



여기서는 현 권한 스키마의 각 사용자 역할을 새로운 권한 스키마에 정의된 사용자 역할로 치환하는 작업을 합니다. 각 사용자 역할 이름 옆에 있는 ⓘ 아이콘에 마우스를 오버하면 해당 사용자 역할에 할당된 권한이 나타납니다. **마침** 버튼을 누르면 권한 스키마 설정이 완료됩니다.

공유 멤버 및 그룹 설정

공유 워크스페이스 홈 화면 우측 상단에 있는 더보기 아이콘을 클릭한 후 **공유 멤버 및 그룹 설정**을 누르면 아래와 같이 공유 멤버 및 그룹 설정 화면이 나타납니다. 여기서는 권한 스키마에서 정의된 각 사용자 역할을 개별 사용자 또는 사용자 그룹에 할당하는 작업을 합니다. 아래 설명을 참조하여 사용자 역할을 할당한 후 **마침** 버튼을 누르면 워크스페이스 접근 권한 설정이 완료됩니다.

공유 멤버 및 그룹 설정

취소

마침

The screenshot shows the '공유 멤버 및 그룹 설정' (Shared Members and Groups Settings) interface. It features a search bar for users, a list of selected users, and a dropdown menu for assigning roles. The interface is annotated with numbered callouts:

- 1**: Member/Group tabs (멤버 66, 그룹 7)
- 2**: User Role dropdown (사용자 역할)
- 3**: User search list (사용자 이름으로 찾기)
- 4**: Role assignment dropdown (역할)

사용자 이름	이름	역할
j_test	권정은	Watcher ▾
chomtt...	chomtttest1	Watcher ▾
jhd1214	jhd	Watcher ▾
hongte...	홍태희	Watcher ▾
ehgud5	김도형	Watcher ▾
aurda	이명훈	Watcher ▾
guest	Guest	Watcher ▾
hiongun	김형근	Watcher ▾
jepark	박정은	Watcher ▾

1. 사용자 역할 할당 단위 선택

- **멤버 탭**: 사용자 역할을 개별 사용자 단위로 할당합니다.
- **그룹 탭**: 사용자 역할을 사용자 그룹 단위로 할당합니다. (사용자 그룹은 관리자 권한으로 지정할 수 있습니다.)

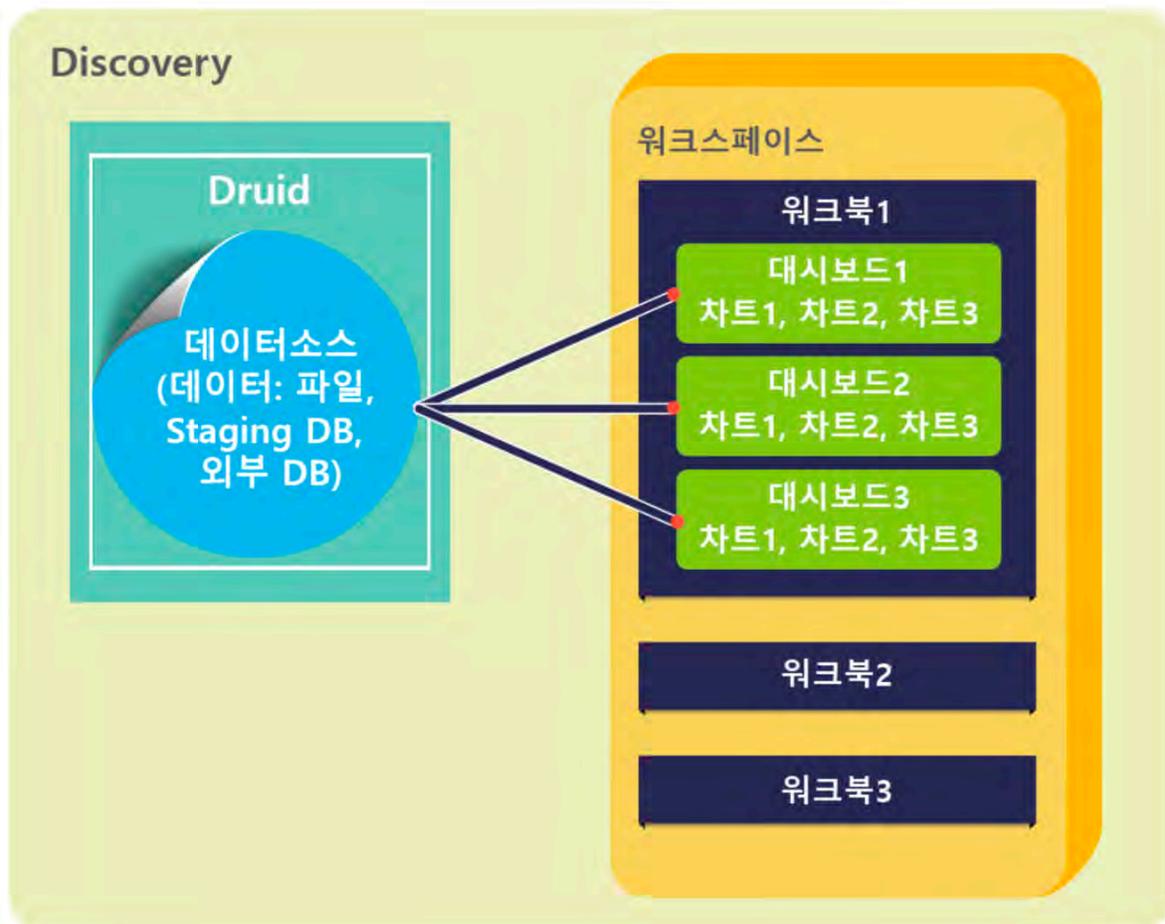
2. **사용자 역할**: 클릭하면 권한 스키마(사용자 역할별 권한 정의) 정보가 팝업 창으로 나타납니다.

3. **멤버/그룹 목록**: Discovery에 등록된 사용자들(그룹 탭에서는 그룹들)이 나열됩니다. 목록에 있는 사용자(그룹) 중 하나를 클릭하면 우측 역할 부여 영역에 추가됩니다. 이미 추가된 사용자(그룹)을 클릭하면 해당 사용자(그룹)가 우측 영역에서 제거됩니다.

4. **사용자 역할 할당**: 이 콤보박스를 클릭하면 현재 적용 중인 권한 스키마에 정의된 사용자 역할들이 나타납니다. 이 중에서 해당 사용자(그룹)에게 할당하고자 하는 역할을 선택하면 됩니다.

CHAPTER 5

워크북



워크북은 Metatron Discovery의 Druid 엔진을 기반으로 하는 비주얼 데이터 분석 모듈입니다. 위 그림과 같이 하나의 독립된 보고서 단위에 해당하는 각 워크북은 다양한 대시보드로 구성되며, 각 대시보드는 원천 데이터 분석 결과를 시각화해서 보여주는 각종 차트로 구성되어 있습니다.

워크북의 기본적인 특징은 다음과 같습니다.

- 시계열 기반의 다차원 데이터 소스를 이용하여 신속하고 유연한 데이터 분석 가능.
- 각 대시보드에서 각종 차트와 텍스트 등의 시각화 위젯을 배치함으로써 프레젠테이션 형식의 보고서로 활용 가능.
- 클러스터링, 예측선, 추세선 등의 자주 쓰는 알고리즘을 GUI(Graphical User Interface)로 구현 가능.

본 단원의 구성은 아래와 같습니다.

5.1 워크북 만들기

Metatron Discovery에서 워크북은 하나의 독립적인 데이터 분석 보고서로서 기능을 합니다. 하나의 워크북을 생성하면 그 안에 여러 대시보드 슬라이드를 담아서 적절한 순서대로 보여줄 수 있습니다.

워크북 생성 절차는 다음과 같습니다.

1. 워크스페이스 하단에 있는 + 워크북 버튼을 클릭하면 워크북을 생성할 수 있는 화면이 나타납니다.



2. 생성하고자 하는 워크북의 이름 (필수 사항) 과 설명을 입력하고 마침 버튼을 누릅니다. Continue to create a dashboard of a new workbook 박스에 체크하면 워크북 생성과 동시에 대시보드 생성하기 화면으로 넘어갑니다. 워크북은 그 안에 대시보드가 있어야 비로소 기능을 할 수 있기 때문에 이렇게 연결되는 것입니다.



3. 화면 가운데의 <+ 데이터 소스 추가> 버튼을 클릭 후, 데이터 소스를 선택하여 대시보드를 생성합니다. 대시보드를 생성하는 절차에 관한 자세한 내용은 [대시보드 만들기](#) 항목을 참조하십시오.



데이터소스를 선택해 주세요

취소

마침

🔍 데이터소스 이름 검색

오픈 데이터만 보기

타입

전체

No.	데이터소스	타입
<input type="checkbox"/>	89 estate_amt 오픈 데이터	수집형 ✓
<input type="checkbox"/>	88 yes月	수집형
<input type="checkbox"/>	87 yes 오픈 데이터	수집형
<input type="checkbox"/>	86 H analysis history 오픈 데이터	수집형
<input type="checkbox"/>	85 market-sales - Stream Data 오픈 데이터	수집형
<input type="checkbox"/>	84 s_history	수집형
<input type="checkbox"/>	83 yes1 오픈 데이터	수집형
<input type="checkbox"/>	82 tet - test	연결형
<input type="checkbox"/>	81 kkk	수집형
<input type="checkbox"/>	80 estate 오픈 데이터	수집형
<input type="checkbox"/>	79 temp-rollup - a	수집형
<input type="checkbox"/>	78 mysql_preset_engine_dialog_single_all	수집형
<input type="checkbox"/>	77 us 500 오픈 데이터	수집형
<input type="checkbox"/>	76 테스트연임	수집형
<input type="checkbox"/>	75 sales_geo - Sales data (2011-20... 오픈 데이터	수집형
<input type="checkbox"/>	74 ecommerce-data - from kaggle 오픈 데이터	수집형
<input type="checkbox"/>	73 test)hisotry	수집형
<input type="checkbox"/>	72 헤더행테스트	수집형
<input type="checkbox"/>	71 temp-test	수집형

estate_amt ☰ ✕

메타데이터 이름

설명

타입 수집형

공개설정 공개

생성일 2019-01-29

사이즈 828.21 KB

Rows 5,862

차원값 loc

차원값 ab idx

차원값 ab gu

축정값 ## py

축정값 ## amt

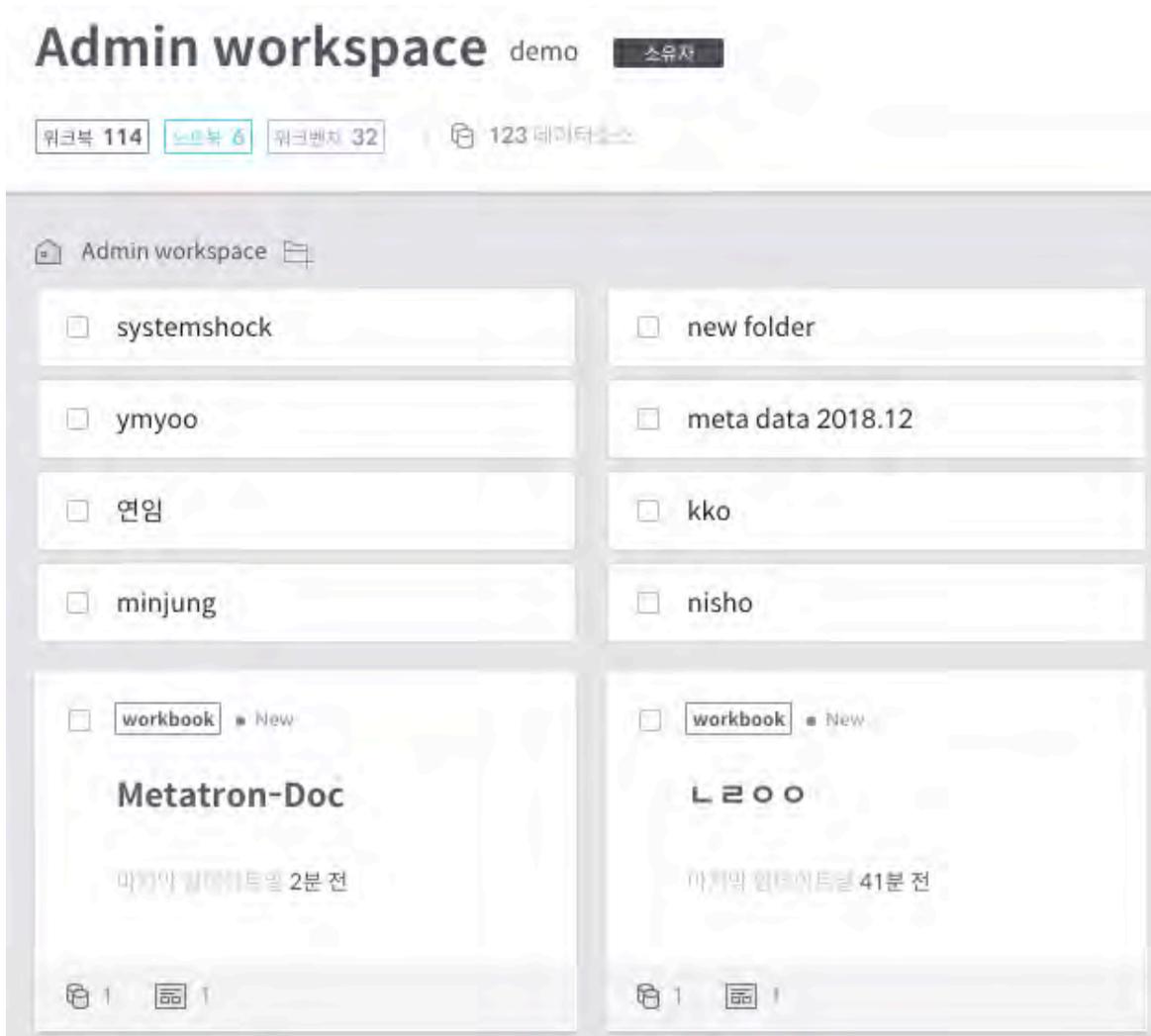
차원값 ab x

차원값 ab y

차원값 ab addr

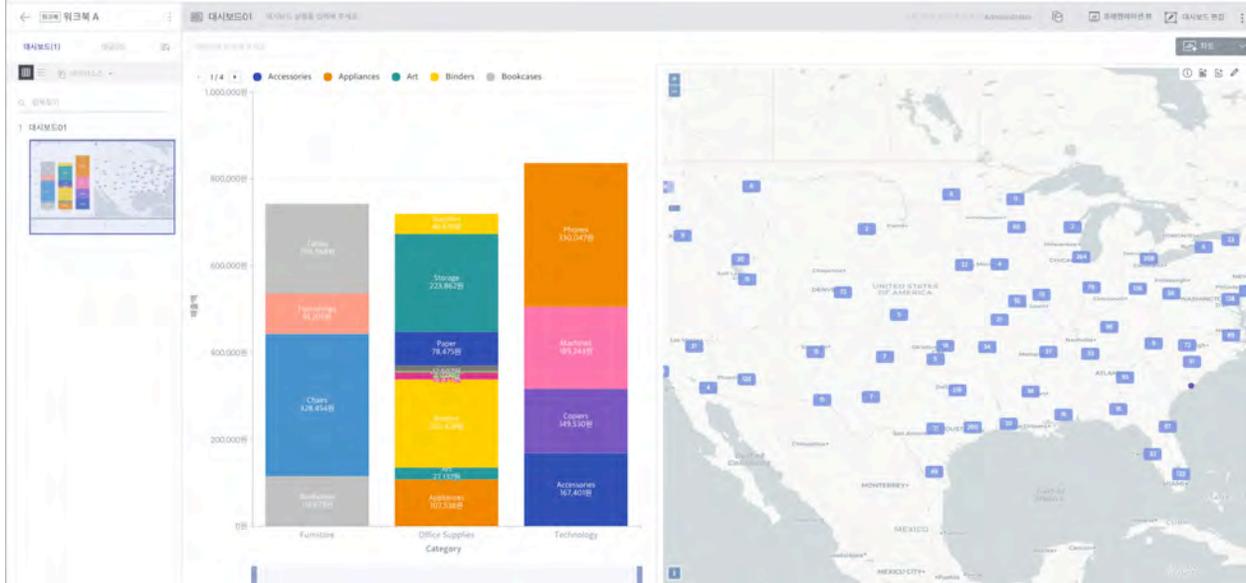
더보기 ▾

4. 새로 생성된 워크북은 아래와 같이 워크스페이스 화면에서 확인할 수 있습니다. 클릭하면 해당 워크북을 사용할 수 있는 화면이 나타납니다.



5.2 대시보드

대시보드는 워크북 안에 보관되며 특정한 데이터 소스를 사용자의 필요에 맞게 분석하고 시각화하는 기능을 제공합니다. 따라서 대시보드 생성 절차는 주로 데이터 소스를 연결하는 것부터 시작됩니다.



여러 데이터 소스를 기반으로 차트와 텍스트 기반의 시각화 분석에 더해, 피벗, 맵차트, 필터링 기능을 통해 분석 작업을 커스터마이징할 수 있습니다.

5.2.1 대시보드 만들기

대시보드 생성 절차는 다음과 같습니다.

1. 워크북 화면에서 + 데이터 소스 추가를 클릭합니다.



2. 해당 워크스페이스에 공개된 데이터 소스 목록에서 대시보드에 연결할 마스터 데이터 소스들을 선택합니다. 이후 단계에서는 여기서 선택하는 마스터 데이터 소스들에 join시킬 다른 데이터 소스를 추가적으로 선택할 수 있습니다.

데이터소스를 선택해 주세요

데이터소스 이름 검색 오픈 데이터만 보기 타입 전체

No.	데이터소스	타입
<input type="checkbox"/>	89 tet-test	연결형
<input type="checkbox"/>	88 kkk	수집형
<input type="checkbox"/>	87 estate 오픈 데이터	수집형
<input type="checkbox"/>	86 temp-rollup-a	수집형
<input type="checkbox"/>	85 mysql_preset_engine_dialog_single_all	수집형
<input type="checkbox"/>	84 us 500 오픈 데이터	수집형
<input type="checkbox"/>	83 테스트연임	수집형
<input type="checkbox"/>	82 sales_geo - Sales data (2011-2... 오픈 데이터	수집형
<input checked="" type="checkbox"/>	81 ecommerce-data - from kaggle 오픈 데이터	수집형
<input type="checkbox"/>	80 H analysis history 오픈 데이터	수집형
<input type="checkbox"/>	79 test)hisotry	수집형
<input type="checkbox"/>	78 s_history	수집형
<input checked="" type="checkbox"/>	77 헤더행테스트	수집형
<input type="checkbox"/>	76 temp-test	수집형

ecommerce-data

메타데이터 이름

설명 from kaggle

타입 수집형

공개설정 공개

생성일 2019-01-15

사이즈 16.26 MB

Rows 531,228

차원값	nb	InvoiceNo
차원값	nb	StockCode
차원값	nb	Description
측정값	#	Quantity
차원값	nb	InvoiceDate
측정값	##	UnitPrice
차원값	nb	CustomerID
차원값	nb	Country

- **데이터 소스 이름 검색:** 해당 워크스페이스에 허용된 데이터 소스를 이름으로 검색합니다.
- **오픈 데이터만 보기:** <오픈 데이터 소스>로 지정된 데이터 소스만 선별하여 볼 수 있습니다.
- **타입:** 데이터 소스를 연결형 또는 수집형으로 선별하여 볼 수 있습니다.
- **데이터 소스 목록:** 설정한 선별 조건에 맞는 데이터 소스들을 보여줍니다.
- **데이터 소스 정보:** 목록에서 선택한 데이터 소스의 정보를 간략하게 보여줍니다.

3. 둘 이상의 데이터 소스를 선택할 경우에는 한 데이터 소스를 다른 데이터 소스로 드래그함으로써 둘을 연결시킬 수 있습니다. 연결된 데이터 소스끼리는 상호간 필터링이 가능합니다. 데이터 소스 연결이 필요 없다면 **마침** 버튼을

클릭하십시오.



4. 한 데이터 소스를 다른 데이터 소스로 드래그하면 데이터 소스 연결을 설정할 수 있는 새 창이 열립니다. 양 테이블에서 상호간 필터링을 할 수 있는 연결 키로 사용할 컬럼을 하나씩 선택한 후 **마침**을 클릭하십시오.

연결 설정

취소

마침

Sales

State

Country	City	State	Postal Code
United States	Jonesb...	Arkansas	72401
United States	Jonesb...	Arkansas	72401
United States	Jonesb...	Arkansas	72401
United States	Jonesb...	Arkansas	72401
United States	Jonesb...	Arkansas	72401
United States	Jonesb...	Arkansas	72401
United States	Philadel...	Pennsylvania	19143
United States	Roswell	Georgia	30076
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Alexand...	Virginia	22304
United States	Mount P...	South Carolin	29464
United States	New Yor...	New York	10024
United States	Mission ...	California	92691
United States	San Diego	California	92037
United States	San Diego	California	92037
United States	San Diego	California	92037

filter test

State or Province

Country	Region	State or Province	City
United States	East	New York	New
United States	East	Massachusetts	Bost
United States	East	Massachusetts	Bost
United States	East	Massachusetts	Bost
United States	East	Massachusetts	Bost
United States	Central	Texas	Dalla
United States	West	Washington	Seat
United States	West	California	Los /
United States	West	California	Los /
United States	East	New York	New
United States	West	Washington	Seat
United States	West	California	Los /
United States	East	New York	New
United States	East	New York	New
United States	East	Massachusetts	Bost
United States	East	New York	New
United States	East	New York	New
United States	East	New York	New
United States	East	District of Columbia	Wasi
United States	West	Washington	Seat
United States	East	District of Columbia	Wasi
United States	East	District of Columbia	Wasi

5. 마스터 데이터 소스 간의 연결 관계를 설정하였으면 마침을 클릭합니다.

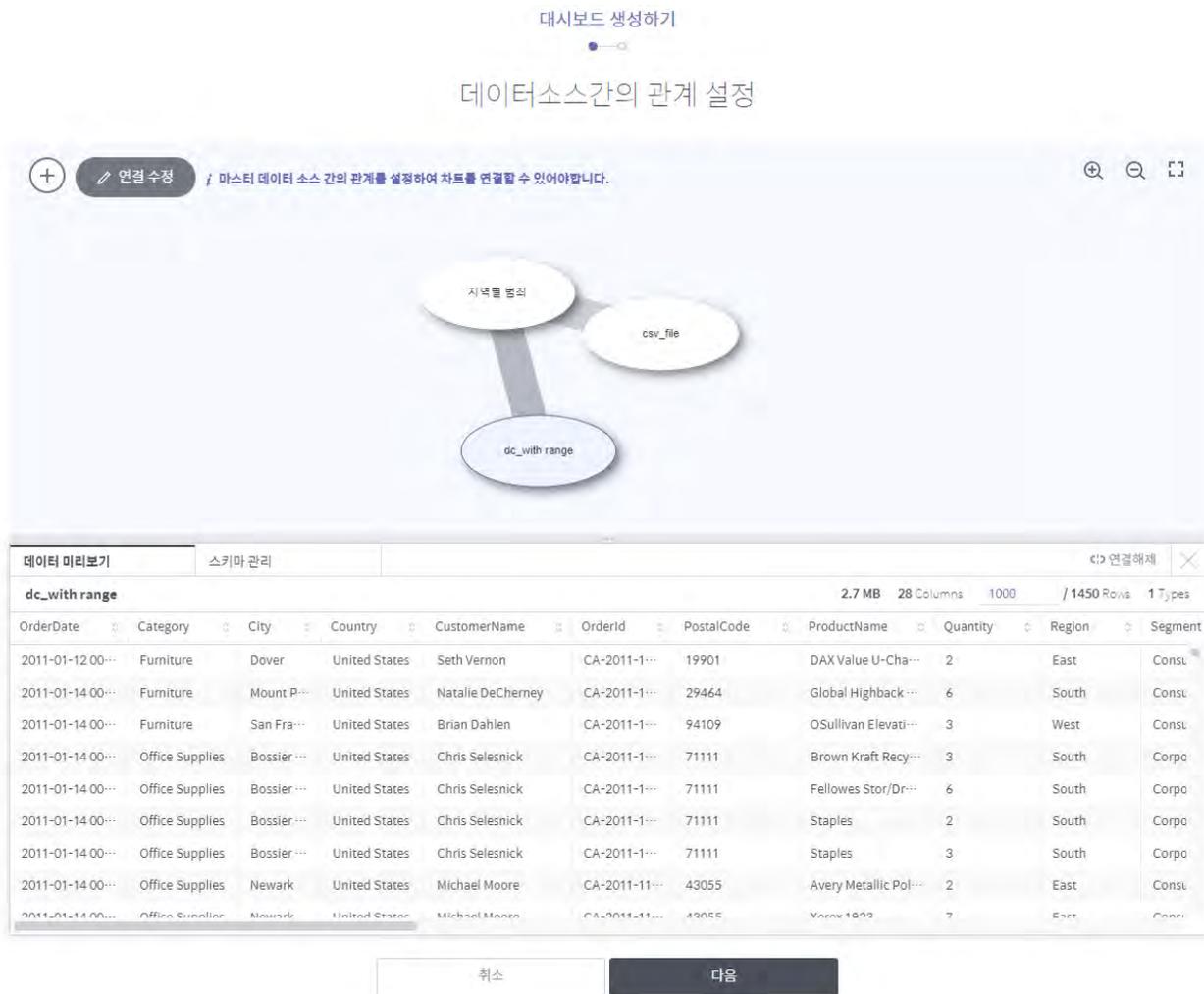
대시보드 생성하기



데이터소스간의 관계 설정



6. 아래 설명을 참조하여 마스터 데이터 소스 연결을 재설정하거나, 앞에서 선택한 최상위 데이터 소스에 join시킬 다른 데이터 소스들을 추가합니다.



마스터 데이터 소스 연결 관계 뷰

- : 클릭하면 새 마스터 데이터 소스를 추가할 수 있습니다.
- **연결 수정**: 클릭하면 데이터 소스 연결 관계를 수정할 수 있습니다.

개별 마스터 데이터 소스 설정 패널 (다이아그램에서 마스터 데이터 소스에 해당하는 타원 중 하나를 클릭 시 열림)

- **데이터 미리보기**: 데이터 소스 join에 따른 결과 테이블을 보여줍니다.
- **스키마 관리**: 선택한 데이터 소스의 join 관계를 관리할 수 있습니다 (자세한 절차는 다음 단계를 확인하십시오).
- **연결해제**: 클릭하면 선택한 데이터 소스가 제거됩니다.

-  : 클릭하면 패널이 닫힙니다.

7. 마스터 데이터 소스 중 하나를 다른 데이터 소스들과 join시키려면, 다이어그램에서 해당 타원 클릭 → 하단 패널에서 스키마 관리 탭 클릭 → + 조인을 위하여 데이터소스를 추가해 주세요 클릭 순으로 진행합니다.



8. 아래 설명을 참조하여 데이터 join 관계를 설정합니다.

조인 취소 **조인**

마스터 데이터소스

Sales

Row ID	Order ID	Order Date	Ship Date
1122	US-2014-14...	2014-01-01 00:...	2014-C
1123	US-2014-14...	2014-01-01 00:...	2014-C
1124	US-2014-14...	2014-01-01 00:...	2014-C
1125	US-2014-14...	2014-01-01 00:...	2014-C
1126	US-2014-14...	2014-01-01 00:...	2014-C
1127	US-2014-14...	2014-01-01 00:...	2014-C
1760	CA-2014-14...	2014-01-01 00:...	2014-C
1914	CA-2014-14...	2014-01-01 00:...	2014-C

Datasource to join

filter test

Shipping Cost	Customer ID	Customer Name
12.39	2189	Frank Cro
24.49	3011	Tammy Ri
19.99	3011	Tammy Ri
4.65	3011	Tammy Ri
3.98	1106	Maxine Cc
1.2	117	Linda Wei
7.44	553	Kristine C.
5.09	949	Ernest Ok

Category = Product Category **조인 키 추가**

조인 타입 1개 조인키

Inner

Left

Right

Full Outer

State = State or Province

결과 미리보기 47 Columns 1000 Rows

sales.Segment	sales.Sub-Category	filter_test.Order ID	sales.Order ID	filter_test.Product Container	sales.
Consumer	Furnishings	87676	CA-2014-124023	Medium Box	M
Consumer	Furnishings	89697	CA-2014-124023	Small Box	M
Consumer	Furnishings	89697	CA-2014-124023	Small Box	M
Consumer	Furnishings	86812	CA-2014-124023	Small Box	M

- **마스터 데이터 소스:** 새로 join시키고자 하는 데이터 소스의 마스터 데이터 소스에 관한 정보를 보여줍니다.
- **Datasource to join:** 마스터 데이터 소스에 조인할 데이터 소스를 선택합니다.
- **조인 키 추가:** <조인 키>는 마스터 데이터 소스와 조인할 데이터 소스 간의 컬럼별 조인 관계를 정의하기 위한 키입니다. 두 데이터 소스에서 서로 연결시킬 컬럼을 하나씩 선택한 후 이 버튼을 클릭하면 새로운 조인 키가 추가됩니다. 이때 각각의 데이터 소스의 컬럼에 정의된 데이터 타입이 일치해야 합니다.
- **조인 타입:** 데이터 소스를 어떻게 조인하여 변형할 것인지를 선택합니다. 다음과 같은 예시를

이용하여 각 조인 타입을 설명하겠습니다.

표 1: 마스터 데이터 소스

제품명 (조인 키)	가격
A	\$22.11
B	\$9.23
C	\$8.99
D	\$10.10

표 2: 조인할 데이터 소스

제품명 (조인 키)	판매량
B	100
D	200
E	50

- **Inner:** 조인 키 컬럼 내 데이터 값을 기준으로 마스터 데이터 소스와 조인할 데이터 소스에 공통적으로 해당하는 레코드만 결과 테이블에 반영합니다. (두 데이터 소스의 교집합)

제품명 (조인 키)	가격	판매량
B	\$9.23	100
D	\$10.10	200

- **Left:** 왼쪽 데이터 소스 (마스터 데이터 소스)의 조인 키 컬럼 내 데이터 값을 기준으로 오른쪽 데이터 소스 (조인할 데이터 소스)의 데이터를 가져와 조인한 뒤, 결과 테이블에 반영합니다. 오른쪽 데이터 소스의 레코드 중에서 왼쪽 데이터 소스에 없는 조인 키 컬럼 데이터 값을 가진 레코드는 버려집니다.

제품명 (조인 키)	가격	판매량
A	\$22.11	null
B	\$9.23	100
C	\$8.99	null
D	\$10.10	200

- **Right:** 오른쪽 데이터 소스 (조인할 데이터 소스)의 조인 키 컬럼 내 데이터 값을 기준으로 왼쪽 데이터 소스 (마스터 데이터 소스)의 데이터를 가져와 조인한 뒤, 결과 테이블에

반영합니다. 왼쪽 데이터 소스의 레코드 중에서 오른쪽 데이터 소스에 없는 조인 키 컬럼 데이터 값을 가진 레코드는 버려집니다.

제품명 (조인 키)	가격	판매량
B	\$9.23	100
D	\$10.10	200
E	\$null	50

- **Full Outer:** 조인 키 컬럼 내 데이터 값을 기준으로 양쪽 데이터 소스의 모든 데이터를 가져와 조인한 뒤, 결과 테이블에 반영합니다. (두 데이터 소스의 합집합)

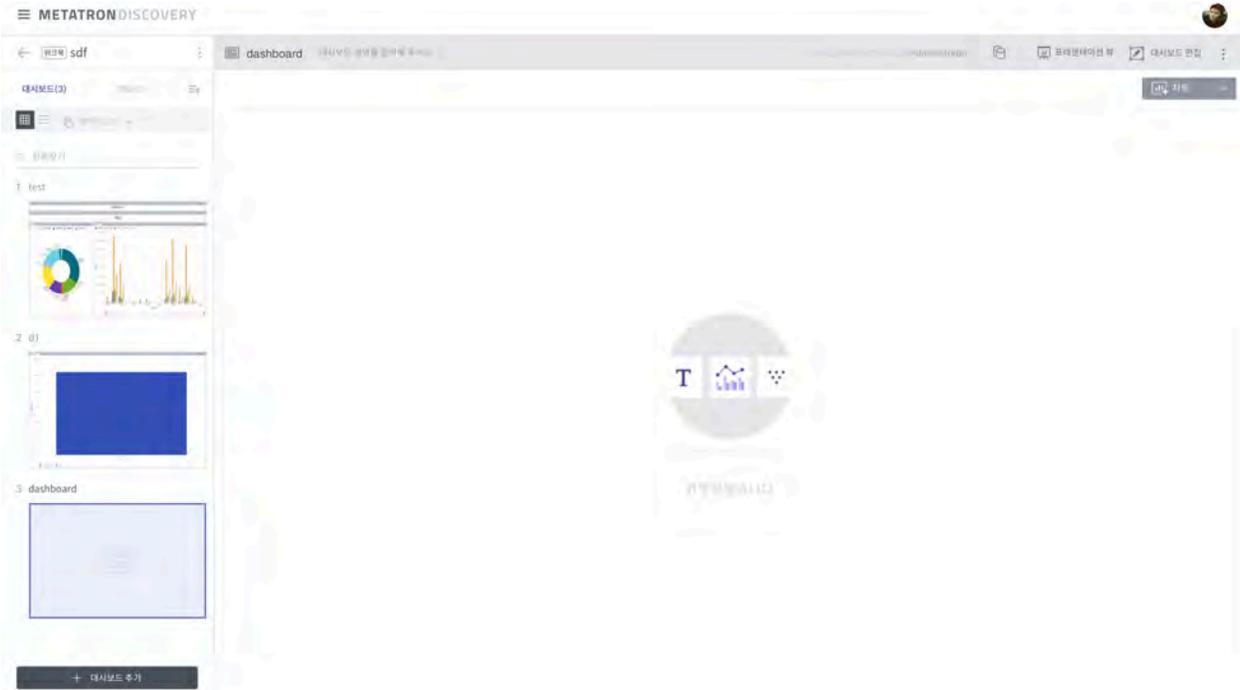
제품명 (조인 키)	가격	판매량
A	\$22.11	null
B	\$9.23	100
C	\$8.99	null
D	\$10.10	200
E	null	50

- **결과 미리보기:** 데이터 소스를 join한 결과값이 나타납니다.

9. 대시보드를 생성하기 위해 불러온 데이터 소스들에 관한 정보를 확인한 뒤, 이름과 설명을 입력하고 마침 버튼을 누르면 새로운 대시보드가 생성됩니다.



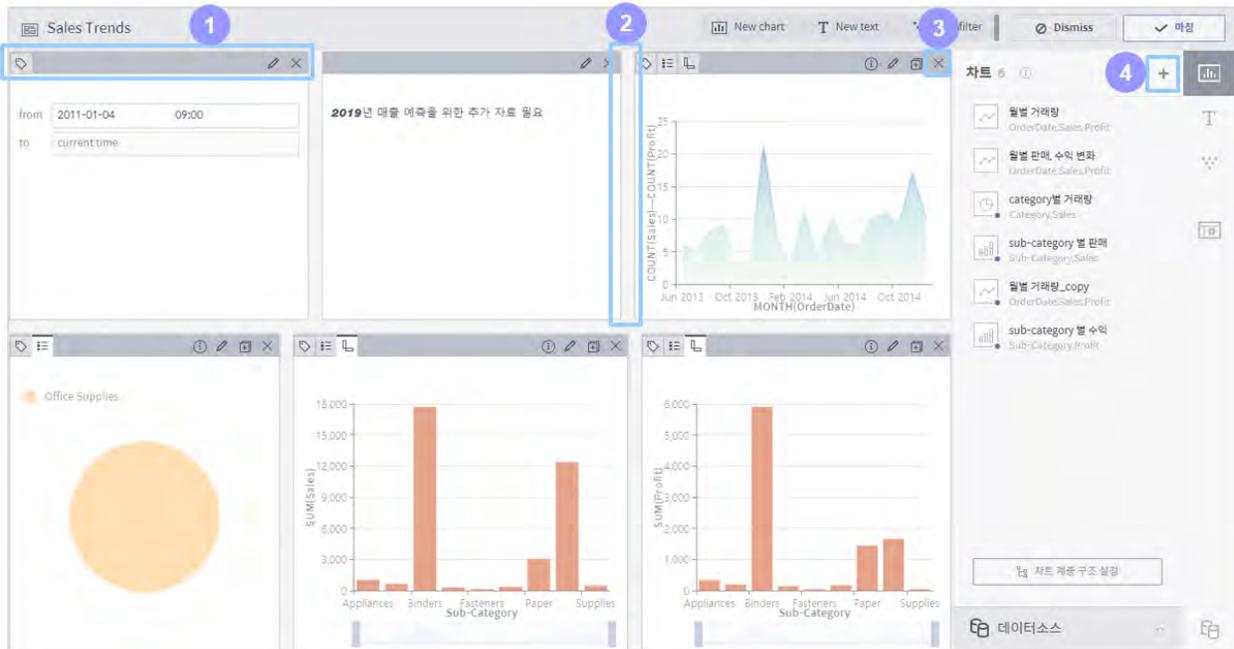
10. 새로 생성된 대시보드는 워크북 홈 화면에 추가됩니다. 클릭하면 해당 대시보드에 관한 화면이 표시됩니다.



5.2.2 대시보드 크기와 레이아웃 변경하기

대시보드 기본 화면에서 **대시보드 편집** 버튼을 클릭하면 해당 대시보드의 구성을 편집할 수 있는 화면으로 이동합니다. 여기서는 위젯 추가 기능을 포함하여 대시보드의 편집 및 계층 구조 설정, 레이아웃 변경 기능을 사용할 수 있습니다.

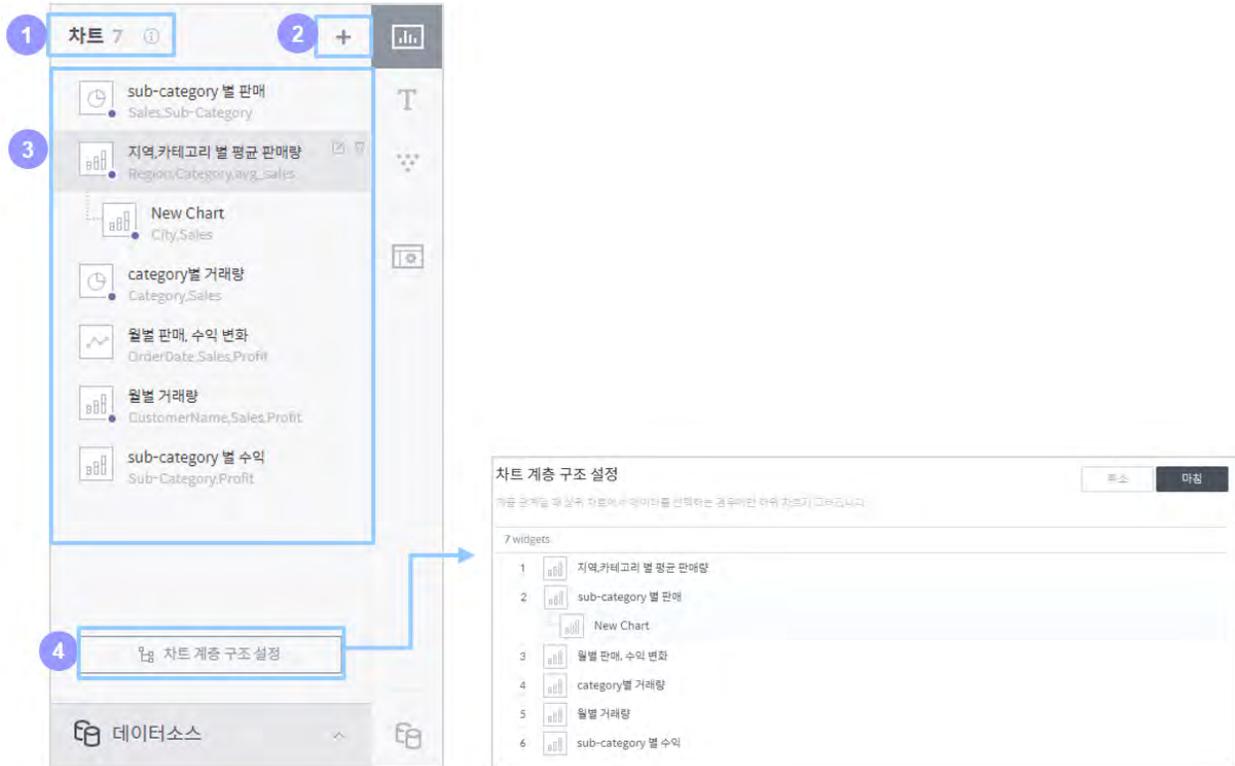
대시보드 위젯 배열 설정



1. **위젯 위치 변경:** 위젯 제목 부분을 드래그하여 위젯의 위치를 변경할 수 있습니다.
2. **위젯 너비 조정:** 위젯과 위젯의 사이를 움직여 위젯의 너비를 조정할 수 있습니다.
3. **화면에 위젯 추가:** 우측 패널에 있는 위젯 목록에서 원하는 위젯을 좌측 위젯 레이아웃 영역으로 드래그하면 해당 위젯이 레이아웃 영역에 추가됩니다.
4. **화면에서 위젯 삭제:** 위젯 레이아웃 영역에 표시된 각 위젯에서 X 버튼을 클릭하면 해당 위젯이 레이아웃 영역에서 제거됩니다.

차트 위젯 패널

차트 위젯 패널에서는 대시보드 내 차트의 추가, 수정, 삭제 등이 가능합니다.



1. **차트 위젯 수:** 현재 대시보드에 등록된 차트 위젯의 개수를 나타냅니다.
2. **차트 위젯 추가:** 대시보드 내에 새로운 차트 위젯을 생성할 수 있습니다.
3. **차트 위젯 목록:** 현재 대시보드에 등록된 차트 위젯들이 열거됩니다. 수정 또는 삭제를 원하는 위젯 항목에 마우스를 오버하면 이를 위한 아이콘이 나타납니다. 또한 위젯 항목을 위젯 레이아웃 영역으로 드래그하면 해당 위젯이 레이아웃 영역에 표시됩니다.
4. **차트 계층 구조 설정:** 대시보드 내 차트 간 상하 관계를 설정할 수 있습니다. 부모 차트에서 데이터 항목을 하나 선택하면 자식 차트가 그 항목을 기준으로 필터링됩니다. 계층 구조를 설정하려면 하위 관계로 설정할 차트를 드래그하여 원하는 상위 관계 차트 밑으로 옮기면 됩니다. 차트 계층 구조 설정이 완료되면 차트 메뉴 상에서도 구조가 변경된 것을 확인할 수 있습니다.

텍스트 위젯 패널

텍스트 위젯 패널에서는 대시보드 내 텍스트 위젯의 추가, 수정, 삭제 등이 가능합니다.

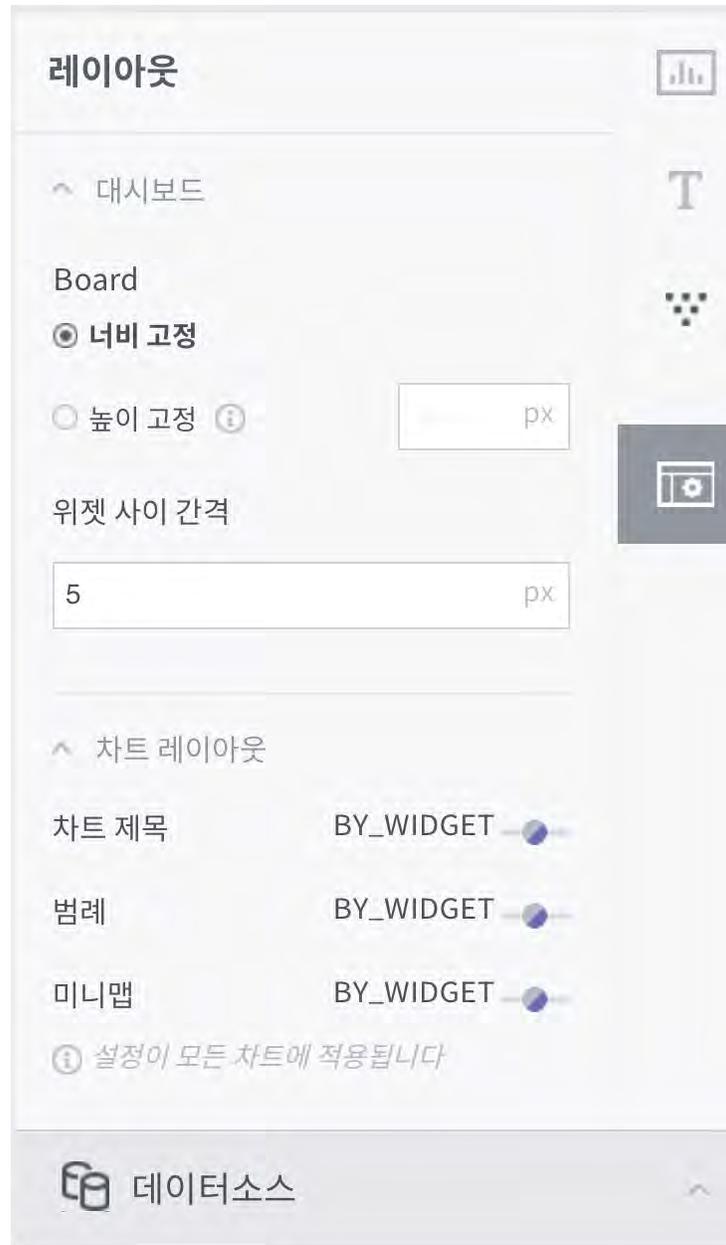


1. 텍스트 위젯 수: 현재 대시보드에 등록된 텍스트 위젯의 개수를 나타냅니다.
2. 텍스트 위젯 추가: 대시보드 내에 새로운 텍스트 위젯을 생성할 수 있습니다.

3. **텍스트 위젯 목록**: 현재 대시보드에 등록된 텍스트 위젯들이 열거됩니다. 수정 또는 삭제를 원하는 위젯 항목에 마우스를 오버하면 이를 위한 아이콘이 나타납니다. 또한 위젯 항목을 위젯 레이아웃 영역으로 드래그하면 해당 위젯이 레이아웃 영역에 표시됩니다.

레이아웃 패널

레이아웃 패널에서는 위젯 레이아웃 영역에서 위젯의 배열과 개별 위젯 표시 방법에 관하여 몇 가지 설정을 설정합니다.

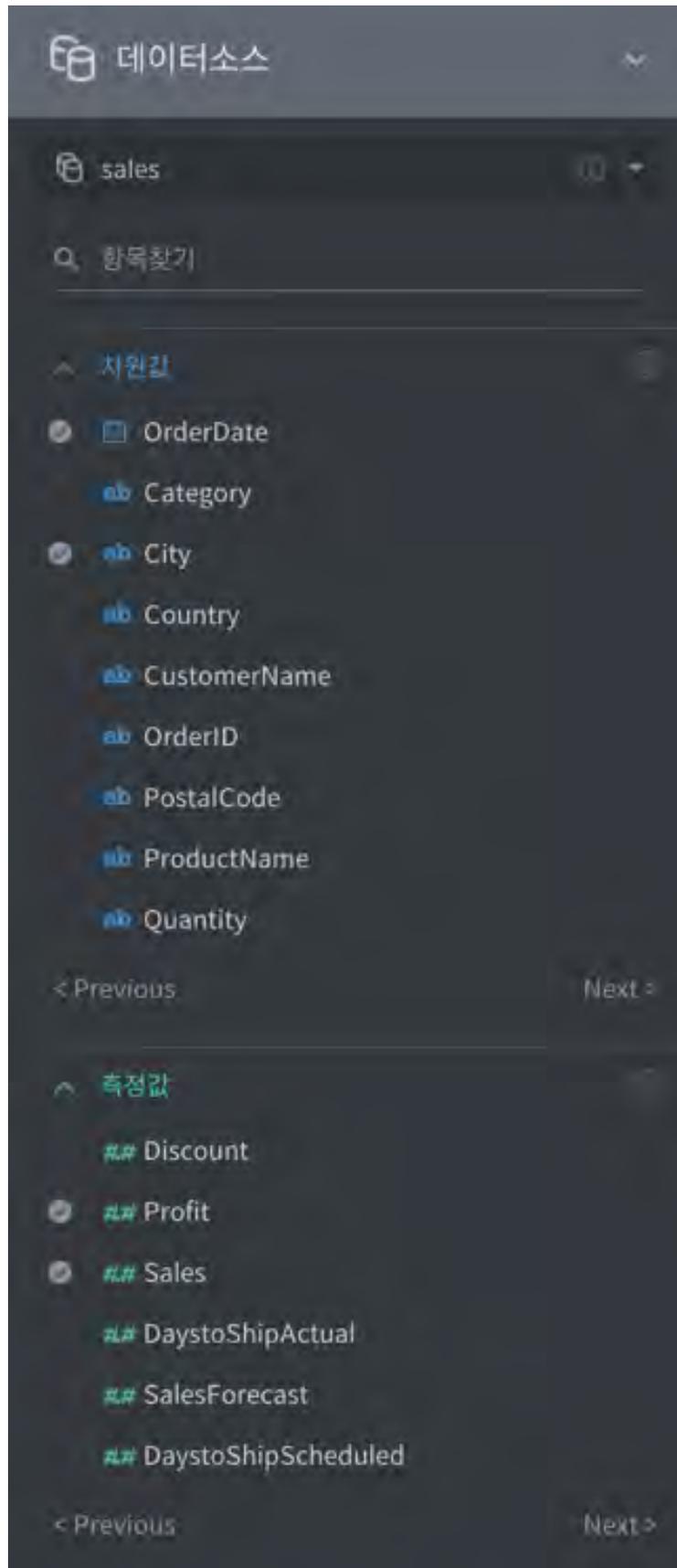


- 보드 높이 설정

- **너비 고정:** 보드의 높이를 화면에 맞춥니다.
- **높이 고정:** 보드의 높이를 고유한 픽셀 값으로 설정합니다.
- **위젯 사이 간격:** 위젯 레이아웃 영역에 표시되는 위젯 간의 간격을 지정합니다.
- **차트 제목:** 위젯 레이아웃 영역 내 차트 및 필터 위젯들의 제목 표시 여부를 일괄 설정합니다.
- **범례:** 위젯 레이아웃 영역 내 차트 위젯들의 범례 표시 여부를 일괄 설정합니다.
- **미니맵:** 위젯 레이아웃 영역 내 차트 위젯들의 미니맵 표시 여부를 일괄 설정합니다.

데이터 소스 패널

데이터 소스 패널에서는 연동된 데이터 소스의 정보를 열람·수정하고, 컬럼 필터를 간편하게 추가할 수 있습니다. 각 차원값 또는 측정값 우측의 필터 아이콘을 눌러서 필터를 추가해 보십시오.



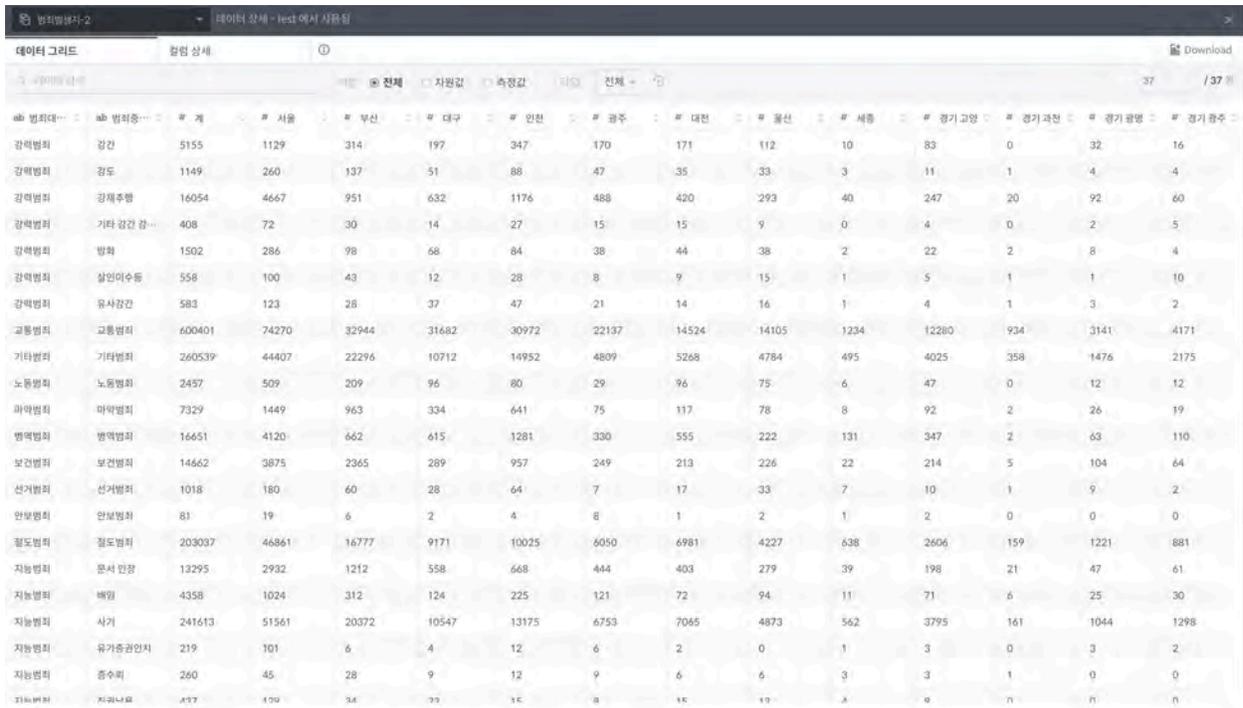
단, 여기서 지정/해제하는 필터는 대시보드 전체에 적용되는 글로벌 필터이고, 차트 에디터에서 지정/해제하는 필터는 차트 내 필터임을 유의하십시오.

5.2.3 대시보드 내에 데이터 소스 확인하기

대시보드 기본 화면에서  버튼을 클릭하면 해당 대시보드에서 사용하는 데이터 소스들의 정보를 보여주는 대화 상자가 나타납니다. 좌측 상단에서 확인하고자 하는 데이터 소스를 선택할 수 있습니다. 이 대화 상자는 크게 3가지 탭(데이터 그리드, 컬럼 상세 탭, 대시보드 데이터 정보)으로 구성됩니다.

데이터 그리드 탭

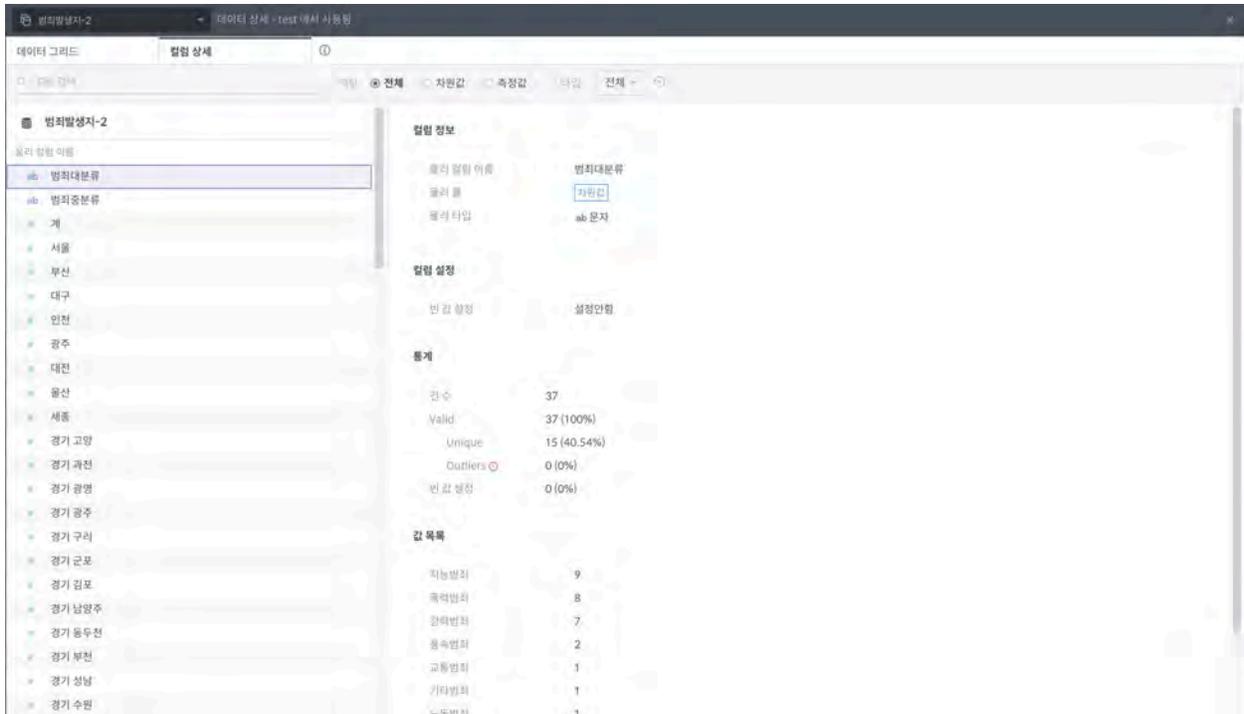
해당 데이터 소스의 레코드 값이 모두 표시됩니다.



메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론	메트론
강력범죄	강간	5155	1129	314	197	347	170	171	112	10	83	0	32	16			
강력범죄	강도	1149	260	137	51	88	47	35	33	3	11	1	4	4			
강력범죄	강제추행	16054	4667	951	632	1176	488	420	293	40	247	20	92	60			
강력범죄	기타 강간 강...	408	72	30	14	27	15	15	9	0	4	0	2	5			
강력범죄	강박	1502	286	98	68	84	38	44	38	2	22	2	8	4			
강력범죄	살인미수등	558	100	43	12	28	8	9	15	1	5	1	1	10			
강력범죄	유사강간	583	123	28	37	47	21	14	16	1	4	1	3	2			
교통범죄	교통범죄	600401	74270	32944	31682	30972	22137	14524	14105	1234	12280	934	3141	4171			
기타범죄	기타범죄	260539	44407	22296	10712	14952	4809	5268	4784	495	4025	358	1476	2175			
노동범죄	노동범죄	2457	509	209	96	80	29	96	75	6	47	0	12	12			
파악범죄	파악범죄	7329	1449	963	334	641	75	117	78	8	92	2	26	19			
범죄범죄	범죄범죄	16651	4120	662	615	1281	330	555	222	131	347	2	63	110			
보건범죄	보건범죄	14862	3875	2365	289	957	249	213	226	22	214	5	104	64			
선거범죄	선거범죄	1018	180	60	28	64	7	17	33	7	10	1	9	2			
안보범죄	안보범죄	81	19	6	2	4	8	1	2	1	2	0	0	0			
질도범죄	질도범죄	203037	46861	16777	9171	10025	6050	6981	4227	638	2606	159	1221	881			
지능범죄	문서 인장	13295	2932	1212	558	668	444	403	279	39	198	21	47	61			
지능범죄	배임	4358	1024	312	124	225	121	72	94	11	71	6	25	30			
지능범죄	사기	241613	51561	20372	10547	13175	6753	7065	4873	562	3795	161	1044	1298			
지능범죄	유가증권인자	219	101	6	4	12	6	2	0	1	3	0	3	2			
지능범죄	중수회	260	45	28	9	12	9	6	6	3	3	1	0	0			
지능범죄	지리정보	437	109	34	17	16	8	16	10	3	9	0	0	0			

컬럼 상세 탭

해당 데이터 소스를 구성하는 각 컬럼에 관한 정보를 상세히 보여줍니다.



대시보드 데이터 정보 탭

해당 데이터 소스 전체에 대한 요약 정보를 보여줍니다.



5.2.4 대시보드 프리젠테이션 하기

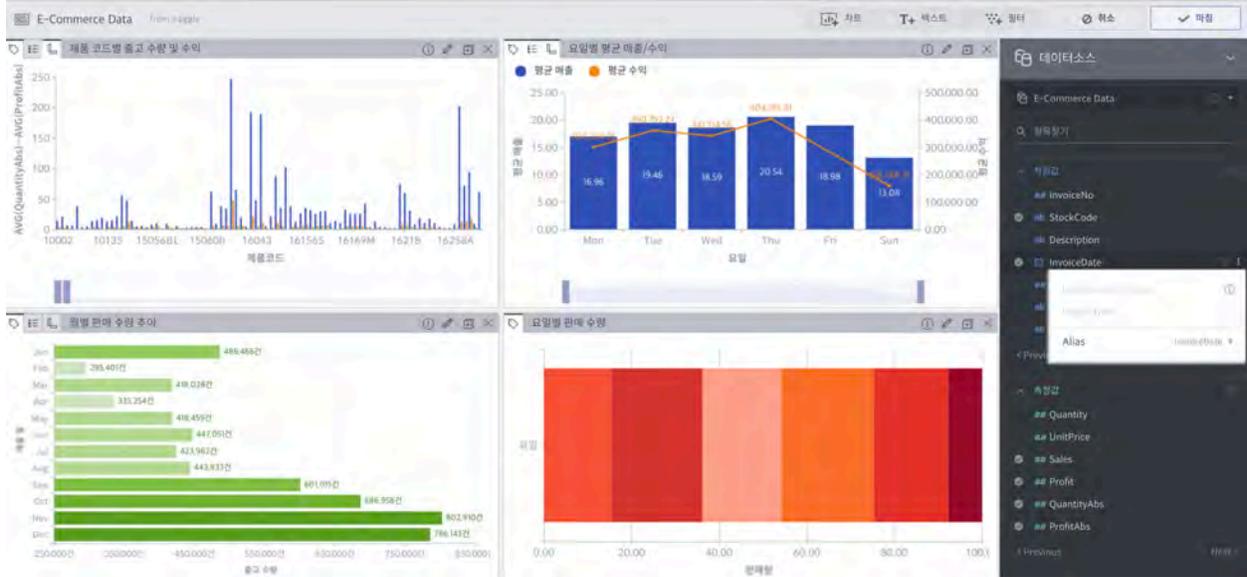
대시보드 기본 화면에서 **프리젠테이션 뷰** 버튼을 클릭하면 워크북의 대시보드들을 프리젠테이션에 적합한 UI로 열람할 수 있습니다. 이를 통해 사용자는 데이터 분석 결과를 쉽게 보고하고 공유할 수 있습니다.



1. 이름: 현재 표시 중인 대시보드의 이름입니다.
2. 슬라이드 네비게이션: 각 동그라미는 워크북 내 대시보드 각각을 가리킵니다. 예를 들어 네 번째 동그라미를 클릭하면 네 번째 대시보드 슬라이드로 이동하고 해당 동그라미가 강조됩니다.
3. 자동 슬라이드 쇼 설정: 시간 간격 선택 후 PLAY 버튼을 클릭하면, 자동 슬라이드 쇼가 시작되어 선택한 시간 간격을 주기로 슬라이드가 넘어갑니다.
4. 나가기: 프레젠테이션 뷰를 종료하고 워크북/대시보드 기본 화면으로 돌아갑니다.

5.2.5 컬럼에 새로운 이름 부여하기

대시보드 편집모드의 데이터 소스 패널에서 컬럼명에 마우스를 올린 후, 우측의 항목을 클릭하면 해당 컬럼의 Alias 값을 확인할 수 있습니다.



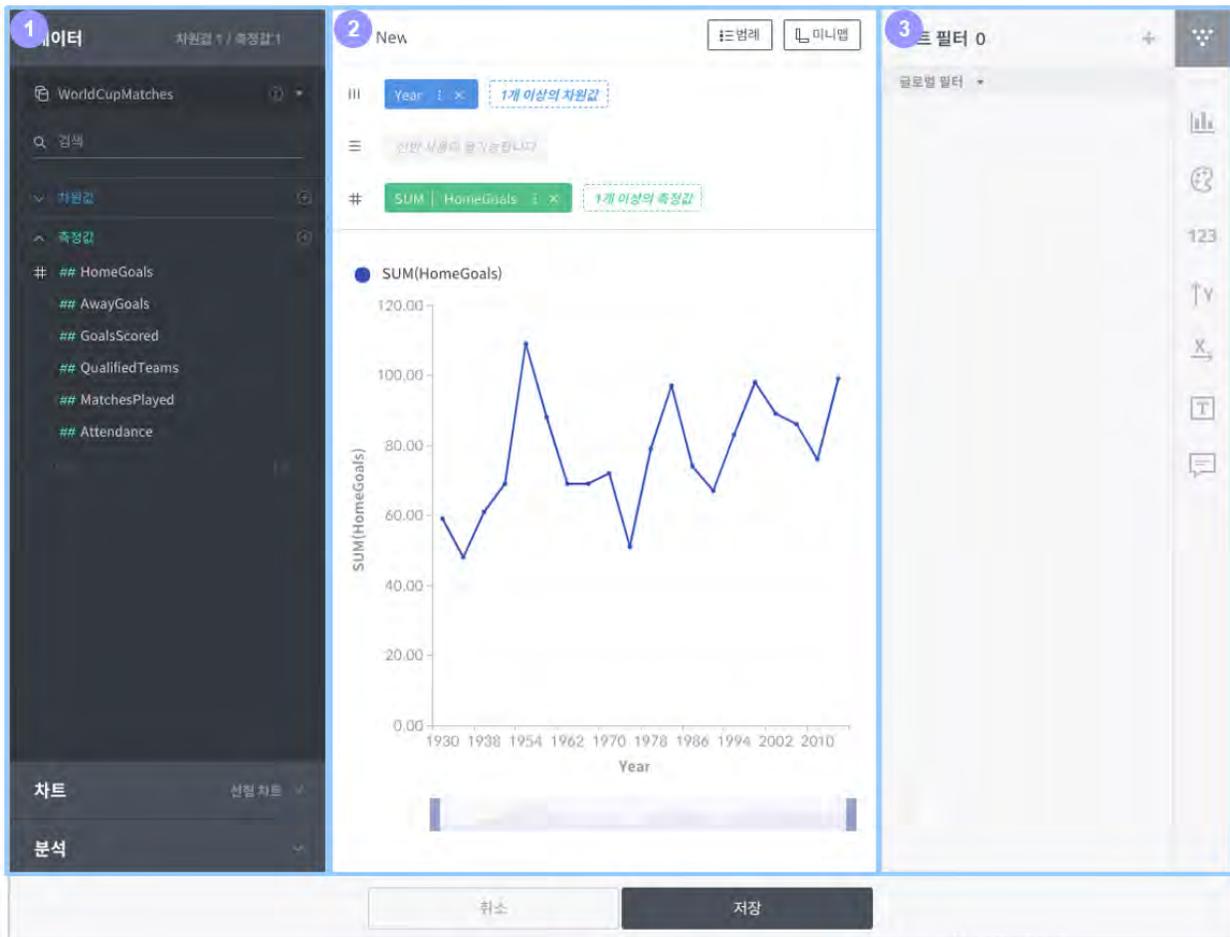
Alias 항목에 마우스를 올리면 새로운 컬럼명을 입력할 수 있는 창이 열리며, 입력 후 적용을 누르면 반영된 것을 확인할 수 있습니다.



5.3 차트

기본적으로 워크북 내 각 대시보드는 분석한 데이터를 시각화하여 보여주는 다양한 차트로 구성됩니다. 데이터 분석 목적에 맞게 차트를 만들기 위해 필수적으로 알아야 할 몇 가지 개념과 Discovery의 차트 구성 UI에 대해 설명합니다.

차트 홈 화면은 다음과 같이 세 영역으로 구분됩니다.



- 컬럼/차트 선택 영역:** 차트를 만들기 위해 필수적으로 행해야 하는 액션의 순서대로 UI가 구성되어 있습니다. Data(데이터 컬럼 리스트) 선택을 통해 차트를 피벗팅할 수 있으며, Chart(차트 종류 리스트)를 선택하여 데이터를 시각화할 수 있습니다. 또한 Analytics(분석)을 통해 원하는 분석 조건을 차트에 탑재할 수 있습니다.
- 시각화 영역:** 피벗팅할 수 있는 선반영역과 실제 차트가 그려지는 시각화 영역으로 구성됩니다. 컬럼/차트 선택 영역에서 차트를 그릴 수 있는 데이터와 차트가 선택되면 이 곳에서 차트가 나타납니다.
- 옵션 영역:** 차트를 보기 좋게 꾸미고, 차트가 보여지는 방식을 사용자의 기호에 맞게 선택할 수 있습니다. 옵션 영역은 필터, 파레트, 축, 숫자표현, 차트표현으로 이루어져 있습니다.

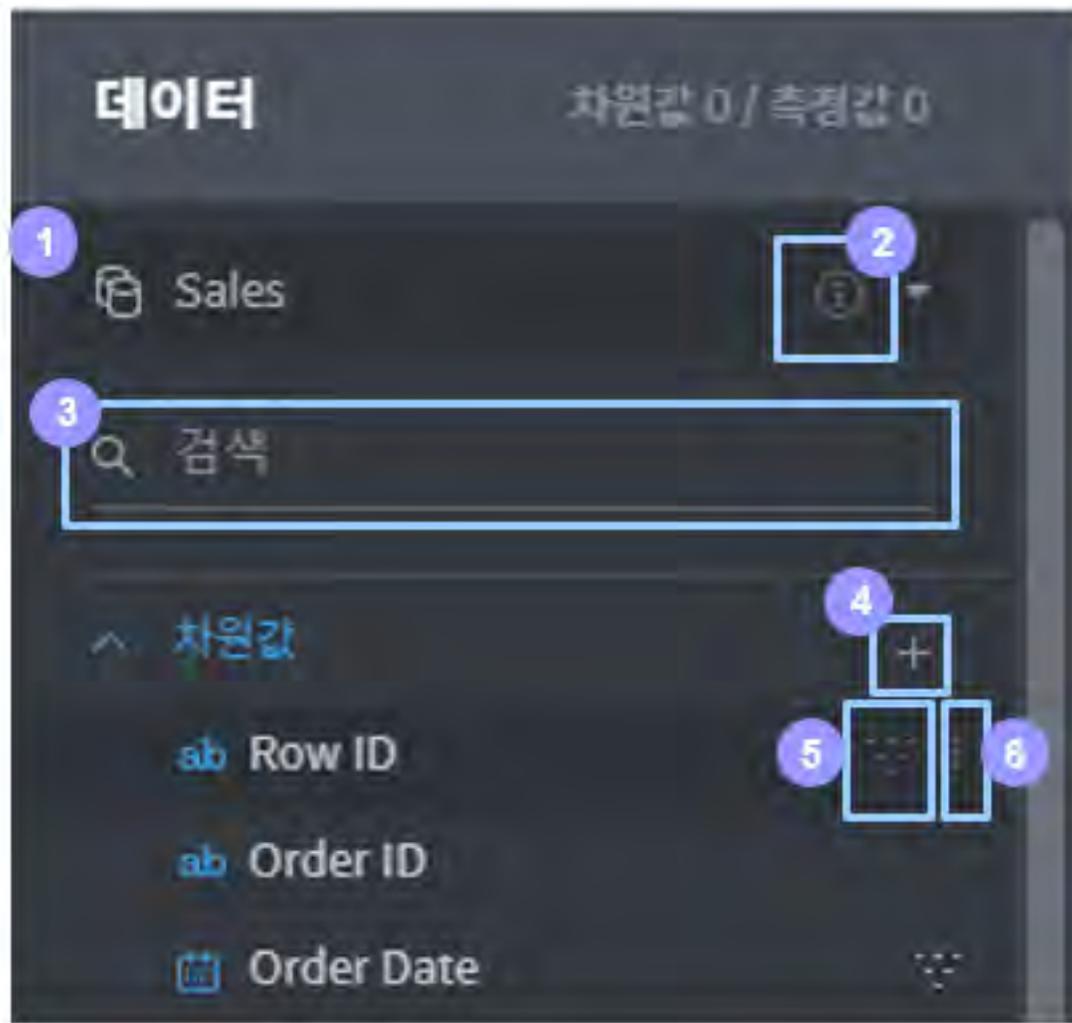
아래 각 절에서는 이러한 사용자 인터페이스를 이용하여 각종 차트를 만들고 관리하는 방식을 소개합니다.

5.3.1 데이터 컬럼 리스트

데이터 컬럼 리스트에 나열되는 컬럼들은 <차원값> 컬럼과 <측정값> 컬럼으로 분류됩니다. 차원값과 측정값의 개념에 대한 자세한 설명은 <차원값> 과 <측정값> 의 개념 항목을 참조하십시오.

데이터 컬럼 리스트 구성

데이터 컬럼 리스트에서는 연동된 데이터 소스들의 정보를 열람·수정하고, 컬럼 필터를 간편하게 추가하거나 제거할 수 있습니다.



1. 데이터 소스 선택/설정: 데이터 소스를 선택하거나 해당 데이터 소스의 연결/join 관계를 설정합니다.

2. **데이터 상세:** 클릭하면 새 대화 상자를 통해 선택된 데이터 소스에 관한 정보를 보여줍니다
3. **컬럼 이름으로 검색:** 데이터 소스 내 컬럼을 이름으로 검색합니다.
4. **사용자 컬럼 추가:** 클릭하면 데이터 소스에 있는 컬럼들을 조합·가공하여 새로운 컬럼을 만들 수 있는 대화 상자가 열립니다. 추가된 사용자 컬럼은 대시보드 전체에서 사용할 수 있습니다.
5. **필터 지정/해제:** 이 버튼은 해당 컬럼에 마우스를 오버하면 생기며, 클릭 시 해당 컬럼을 차트 필터로 지정하고 다시 한번 클릭하면 지정된 차트 필터가 해제됩니다. 필터로 지정된 컬럼 항목에는  아이콘이 마우스 오버와 상관 없이 표시됩니다.
6. **더 보기:** 이 버튼은 해당 컬럼에 마우스를 오버하면 생기며, 컬럼에 대한 추가적인 정보를 확인하고 별칭을 지정할 수 있습니다.
 -  : 클릭하면 새 대화 상자가 나타나면서 해당 컬럼의 요약 정보와 데이터 값들을 보여줍니다.
 - **논리 컬럼 이름:** 해당 컬럼의 논리적 컬럼명을 보여줍니다.
 - **타입:** 해당 컬럼의 논리적 데이터 타입을 보여줍니다.
 - **Alias:** 해당 컬럼에 대한 별칭을 지정할 수 있습니다. 정식 컬럼명은 영숫자와 몇 가지 특수문자로 제한되며 공백도 포함할 수 없기 때문에 보다 구분하기 편한 별칭을 등록하면 분석 시 편의를 도모할 수 있습니다. 지정된 별칭은 대시보드 전체에 적용됩니다.
 - **값 별칭:** 해당 컬럼에 포함된 각 데이터 값에 대해서도 별칭을 지정할 수 있습니다. 지정된 별칭은 대시보드 전체에 적용됩니다.

사용자 컬럼 추가

데이터 소스 컬럼 리스트에서 + 버튼을 클릭하면 사용자 컬럼 추가를 위한 대화 상자가 열립니다. 여기서는 데이터 소스에 있는 기존 컬럼들에 각종 공식을 적용하여 차트를 만드는 데 필요한 새로운 컬럼을 만들 수 있습니다.

사용자 컬럼

리셋 마침

1 컬럼명 DIMENSION_1

2 `CAST({CHARACTER_SET_NAME}, 'text')`

✓ 계산식에 이상이 없습니다 유효성 체크

추진

3 컬럼 추가 1/1

- ab CHARACTER_SET_NAME
- ab DEFAULT_COLLATE_NAME
- ab DESCRIPTION
- ## MAXLEN
- current_datetime

공식 추가

ALL

LOGICAL FUNCTION

IF

NVL

CASE

IN

TYPE_CONVERT FUNCTION

CAST

TIMESTAMP

UNIX_TIMESTAMP

TIME FUNCTION

DATEDIFF

4

CAST

TYPE_CONVERT FIELD

자정한 타입으로 값을 변환하여 반환합니다.

CAST(parameta,type)

- parameta: 문(는) 변환할 대상이 되는 문자열 혹은 숫자입니다.
- type: 문(는) 'DOUBLE', 'LONG', 'STRING', 'DATETIME' 중 하나로 변환될 타입입니다.

CAST('100.123', 'DOUBLE') => 100.123

CAST(TIMESTAMP('2016-01-01T12:00:00.000000')) => 2016-01-01 12:00:00.000000

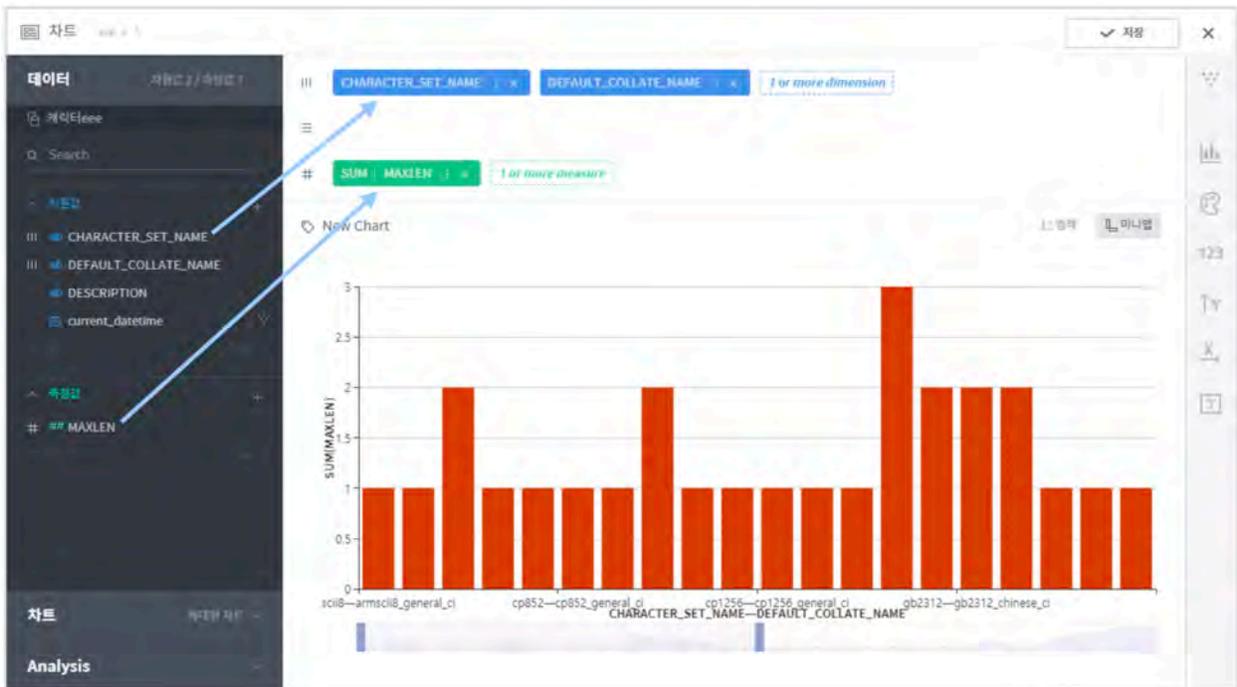
1. **컬럼 이름:** 사용자 컬럼의 이름을 적는 란입니다.
2. **코드 영역:** 사용자 컬럼을 만들기 위한 코드를 적는 란입니다. 아래의 컬럼 및 공식 목록에서 원하는 목록을 클릭하면 이 영역에 자동으로 타이핑 됩니다.
3. **컬럼 추가:** 데이터 소스에 주어진 기존 컬럼 목록입니다. 목록에 제시된 컬럼 항목 중 하나를 클릭하면 해당 컬럼이 코드 영역에 자동으로 타이핑됩니다.
4. **공식 추가:** Metatron에서 지원하는 공식 목록입니다. 목록에 제시된 공식 중 하나를 클릭하면 해당

공식이 코드 영역에 자동으로 타이핑되고 타이핑 커서가 파라미터를 입력하는 부분으로 자동 이동됩니다. 각 공식의 용도와 사용법, 예제에 관해서는 화면 우측의 도움말 상자를 참조하세요.

5.3.2 차트 그리기 (pivoting)

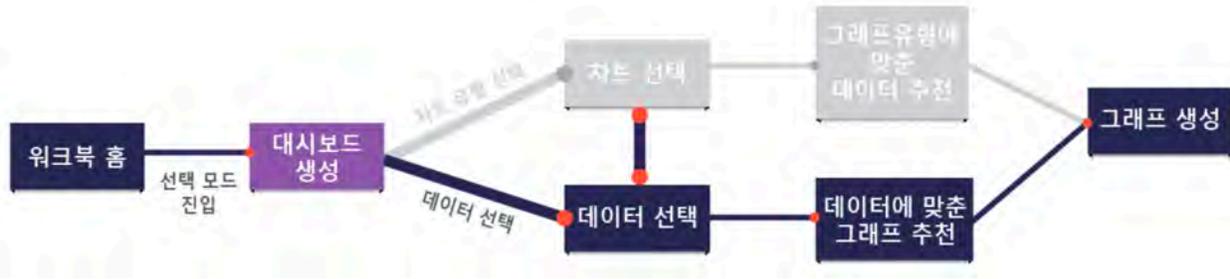
<Pivoting>이란

Pivoting이란 주어진 테이블을 특정 컬럼들을 기준으로 그룹화하는 과정을 의미하며, 이를 통해 분석가는 원천 데이터의 특정한 측면을 그래픽 또는 도표로 확인할 수 있습니다. 이러한 과정에는 의미 있는 데이터를 포함하는 컬럼들을 열/행/교차 선반에 배치하는 것을 포함합니다.



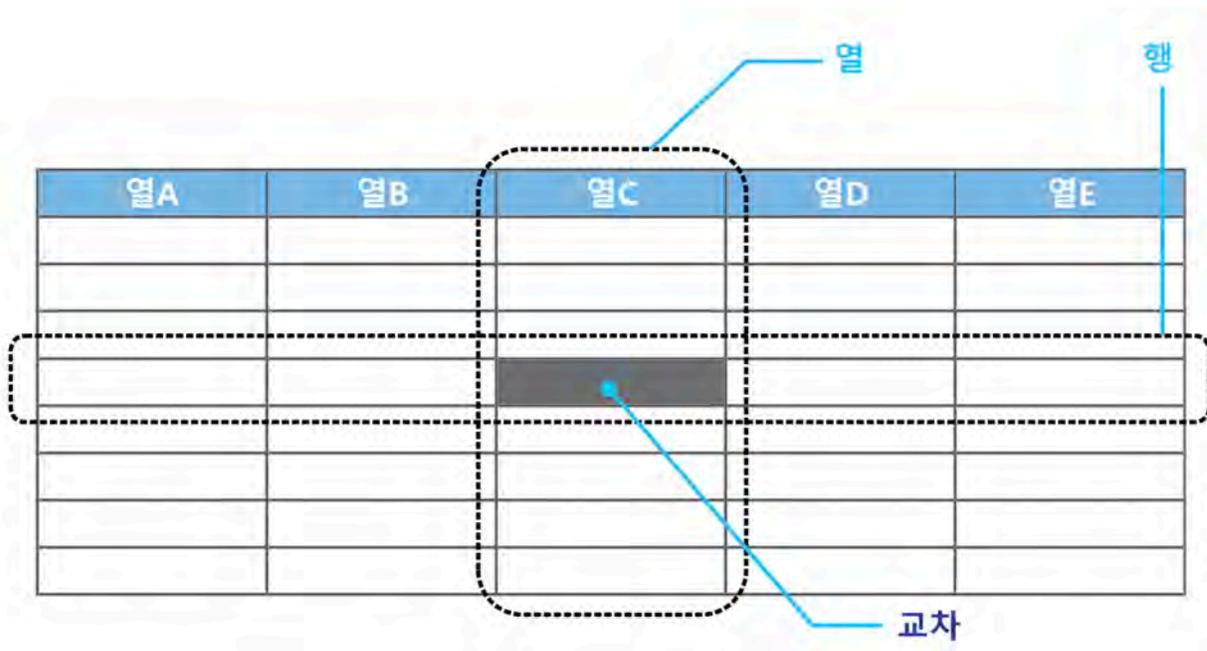
위 그림은 두 개의 차원값 컬럼을 열선반에 배치하고 하나의 측정값 컬럼을 교차선반에 배치 한 상태를 보여주고 있습니다. 차트에는 이렇게 선반에 올려놓은 컬럼들의 데이터가 표시됩니다.

차트 유형별로 선반별 필수/권장 컬럼 유형이 다르며, 컬럼들을 선반에 올려놓기 전에 먼저 차트 유형을 선택하면 선반에 필요한 컬럼 유형이 제시됩니다.

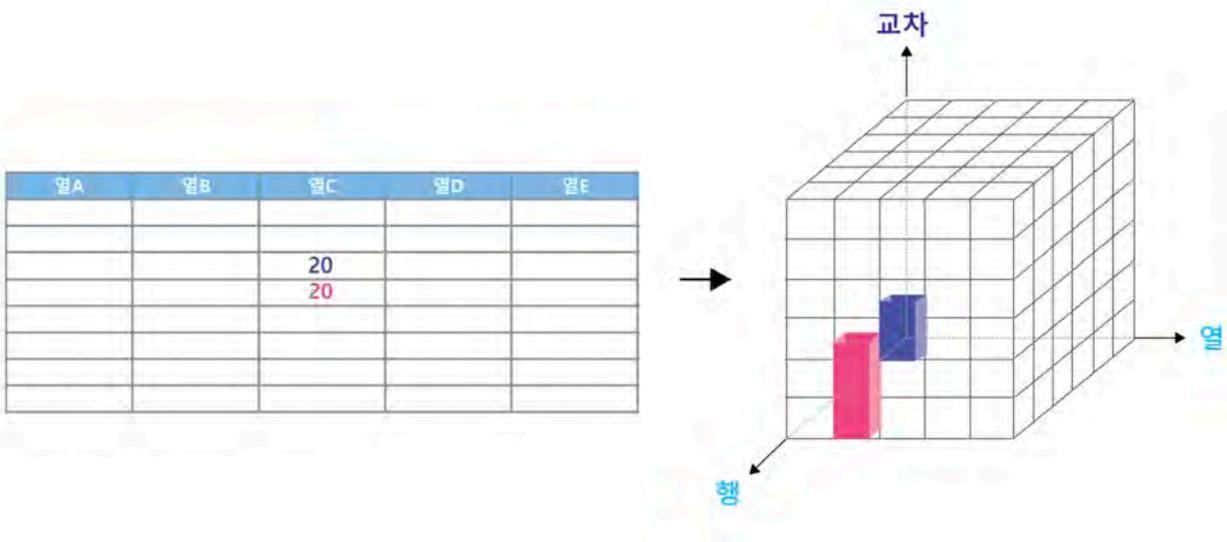


열/행/교차 선반의 개념

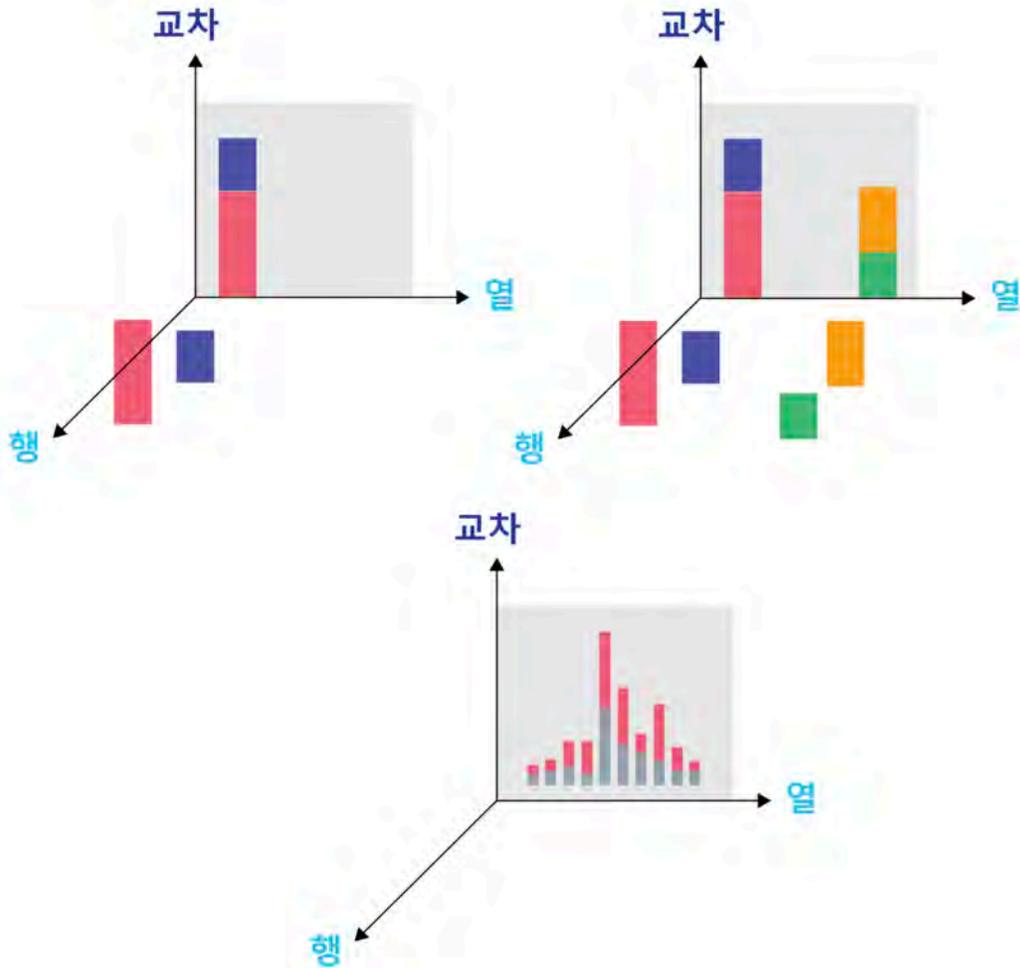
열/행/교차 선반의 개념은 Excel의 구조를 생각하면 쉽게 이해할 수 있습니다. 아래 그림과 같이 열/행은 블록을 정의하는 역할을 하고, 교차는 블록 안에 들어갈 값을 정하는 역할을 합니다.



Excel에서 데이터를 열/행/교차를 2차원 값인 그리드에 표현한다면, Metatron은 OLAP Data Discovery 도구로서, OLAP Cube를 통해 다차원에서 데이터를 조회합니다. 아래의 차트는 Metatron에서 3차원 큐브로 나타낸 열/행/교차 값의 축 그림입니다.



Excel 그리드의 값을 3차원 차트에서 표현한다면 그리드의 교차값이 막대바 형태로 여러 개 세워 지게 될 수 있습니다. Metatron에서는 2차원 단면으로 차트가 보여지기 때문에 열과 행 기준으로 막대 바를 쌓아 올려서 표현하게 됩니다. 결국 아래 그림의 회색 부분과 같은 2차원 형태의 차트로 나타납니다.

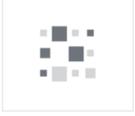


5.3.3 차트 유형 선택

Metatron Discovery는 20여 가지의 차트를 제공하고 있습니다. 차트를 선택하기 전에 먼저 컬럼들을 선반에 올려놓으면 그에 어울리는 차트들이 보라색으로 강조됩니다.



아래 표는 각 차트별 생성조건과 사용 속성, 사용 유형, 사용 예시에 대해 정리한 것입니다.

차트명/아이콘	생성 조건	사용 속성	사용 유형	사용 예시
 막대형 차트	열: 차원값 1개 이상 & 교차: 측정값 1개 이상	개별 항목의 값 비교	그룹들을 비교할 때 사용하거나 시간에 따른 변화 추이를 보고 싶을 때 사용합니다. 변화 추이가 클 때 사용하면 효과적입니다.	제품별 매출 및 수익 비교
 표	열 또는 행: 차원값 1개 이상 & 교차: 측정값 1개 이상	항목별 교차 데이터를 텍스트로 표시	특정 기준에 따른 측정 값을 보고 싶을 때 사용합니다. 시각화보다는 자세한 데이터와 정확한 값을 보려는 경우에 사용합니다.	연도별 매출 상세
 선형 차트	열: 차원값 1개 이상 & 교차: 측정값 1개 이상	시간의 흐름에 따른 데이터 변화	시간에 따른 변화 추이를 보고 싶을 때 사용합니다. 변화 추이가 작을 때는 막대형 차트보다 선형 차트를 사용하는 것이 효과적입니다.	월별 매출 추이
 분산형 차트	열: 측정값 1개 & 행: 측정값 1개 & 교차: 차원값 1개 이상	관련된 여러 항목의 연관관계 표시	두 변수 간의 관계를 정의하고 싶을 때 사용합니다.	제품의 매출과 수익의 관계
 히트맵	열 또는 행: 차원값 1개 이상 & 교차: 측정값 1개 이상	항목별 교차 데이터를 색상 분포 형태로 표시	색과 크기를 이용해 두 변수를 직관적으로 비교할 때 사용합니다. 표 차트에서 시각적 요소를 강조하기 위해 사용합니다.	지역별 각 제품 판매량
 원형 차트	교차: 차원값 1개 이상, 측정값 1개 이상	전체 대비 각 항목이 차지하는 비율	전체를 이루는 부분들을 비교할 때 사용합니다.	웹 브라우저의 마켓 점유 비교
				

5.3.4 차트 스타일 속성

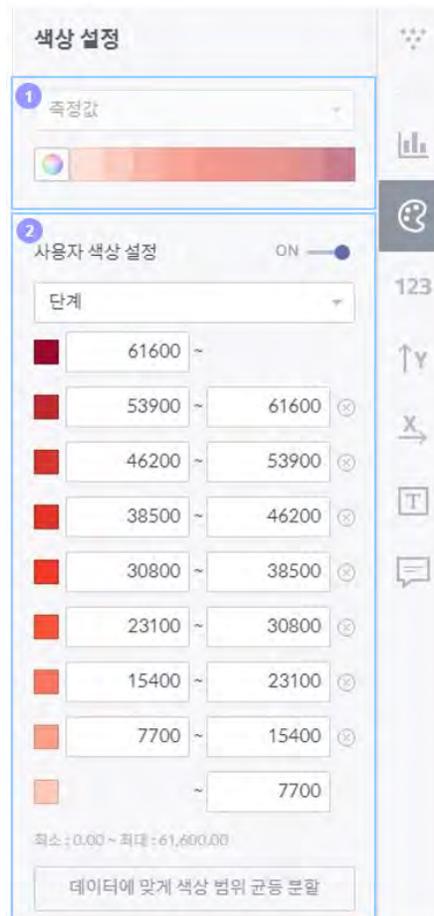
데이터를 피봇팅하고 나면 우측에 차트 스타일을 설정할 수 있는 옵션 메뉴가 나타납니다. 메뉴의 구성은 각 차트의 종류별로 다르게 나타납니다. 모든 차트 유형에 일반적으로 적용되는 설정 항목을 설명하고, 차트 유형별로 고유한 <공통 설정> 유형에 대해서 설명합니다.

차트 스타일 설정 메뉴

차트 스타일 설정 메뉴를 구성하는 각 항목별로 설정 방식을 설명합니다. 사용하는 차트 유형에 따라 아래 제시된 항목 중 일부가 사용되지 않을 수 있음을 유의하십시오.

색상 설정

차트에 들어가는 각종 색상을 정의합니다.

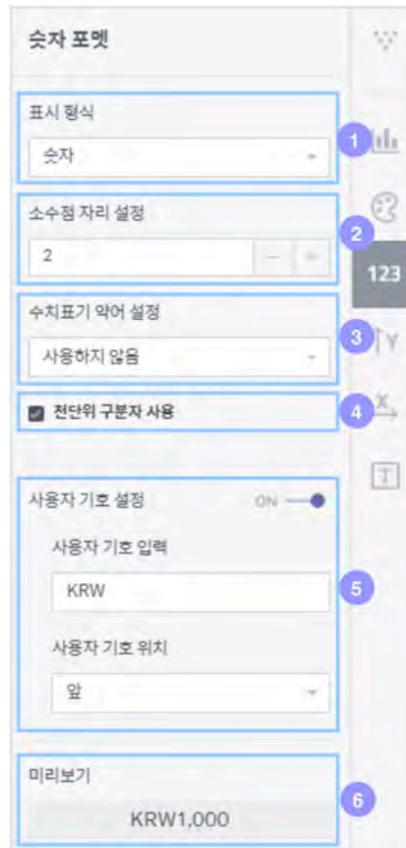


1. **그래프 색상 설정:** 그래프에서 데이터 표시 항목별 색상을 구분하는 기준을 정한 후 색상 테마를 선택합니다.

- **Series:** 측정값의 종류에 따라 색상을 구분합니다.
 - **Dimension:** 차원값의 종류에 따라 색상을 구분합니다.
 - **Measure:** 측정값의 크기에 따라 색상을 구분합니다.
2. **색상범위 설정:** 데이터 표시 색상 구분 기준을 **Measure**로 선택할 때 나오는 항목으로서, ON으로 설정하면 측정값의 범위에 따라 색상을 다르게 나타낼 수 있습니다. 색상범위는 최저 구간부터 시작해서 원하는 개수만큼 세분화할 수 있는데, 새로운 구간을 추가하려면 현재의 마지막 구간의 최댓값을 먼저 조정하고 다음 **새 범위 추가** 버튼을 클릭해야 합니다.

숫자 포맷

차트 내 그래프에 텍스트로 나타나는 데이터값의 표시 형식을 정의합니다. 이 기능을 사용하려면 데이터 레이블 설정 메뉴에서 레이블 표시 기능을 먼저 켜주세요.



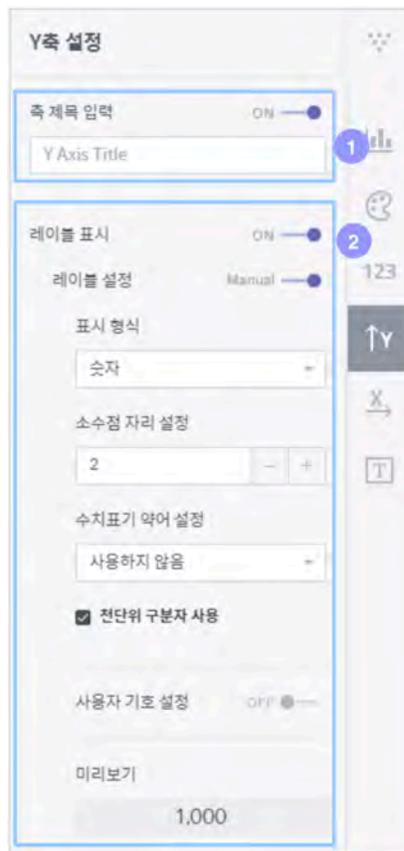
1. **표시 형식:** 데이터 값을 숫자, 통화, 퍼센트, 지수 중 어떤 형식으로 표시할지 선택합니다.
2. **소수점 자리 설정:** 데이터 값을 소수점 몇 번째 자리까지 표시할지 결정합니다.
3. **수치표시 약어 설정:** 데이터 값의 자릿수가 클 경우 천 단위 (K), 백만 단위 (M), 십억 단위 (B) 중 하나를 약어로 설정할 수 있습니다. **자동 조정**을 선택하면, 데이터 값들의 자릿수에 가장 적절한 단위가

자동으로 결정됩니다.

4. **천단위 구분자 사용**: 데이터 값들을 천단위 구분자를 사용하여 표시할 것인지 선택합니다.
5. **사용자 기호 설정**: 데이터 값들의 앞/뒤에 사용자가 원하는 텍스트를 삽입하여 표시할 수 있습니다.
6. **미리 보기**: 정의한 숫자 형식에 따른 결과를 예시로 보여줍니다.

Y축 설정 (차트 유형 세로형 기준)

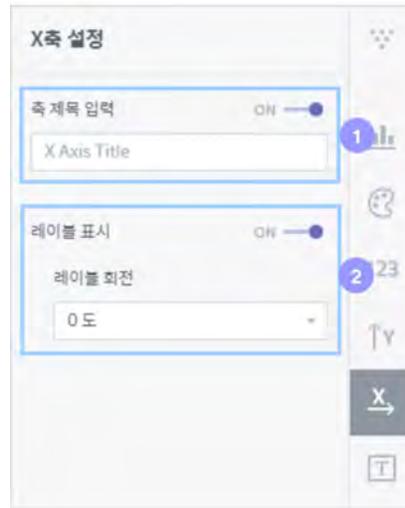
공통 설정 항목에서 차트 유형을 가로형으로 바꾸면 X축과 Y축 설정 항목이 뒤바뀝니다.



1. **축 제목 입력**: 차트의 Y축에 제목을 입력할 수 있습니다. 해당 기능을 사용하지 않으면 Y축에 제목이 나타나지 않게 됩니다.
2. **레이블 표시**: 차트의 Y축에 데이터 레이블을 표시할 것인지 선택합니다. 해당 기능을 사용하지 않으면 Y축에 데이터 레이블은 나타나지 않게 됩니다.
 - **레이블 설정**: Y축 데이터 레이블에 표시되는 숫자의 형식을 지정합니다. 자동으로 설정하면 **숫자 포맷** 항목의 설정값이 동일하게 반영되며, 수동으로 설정하면 Y축 데이터 레이블만의 고유한 형식을 지정할 수 있습니다.

X축 설정 (차트 유형 세로형 기준)

여기서는 차트의 X축 표시 방식을 정의합니다. 공통 설정 항목에서 차트 유형을 가로형으로 바꾸면 X축과 Y축 설정 항목이 뒤바뀝니다.



1. **축 제목 입력**: 차트의 X축에 제목을 입력할 수 있습니다. 해당 기능을 사용하지 않으면 X축에 제목이 나타나지 않게 됩니다.
2. **레이블 표시**: 차트의 X축에 데이터 레이블을 표시할 것인지 선택합니다. 해당 기능을 사용하지 않으면 X축에 데이터 레이블은 나타나지 않게 됩니다.
 - **레이블 회전**: 차트의 X축에 나타나는 데이터 레이블을 0도/45도/90도 중 어떤 각도로 표시할 것인지 선택합니다.

데이터 레이블 설정

차트 내 그래프에 데이터 값을 표시할 것인지 여부를 선택합니다.

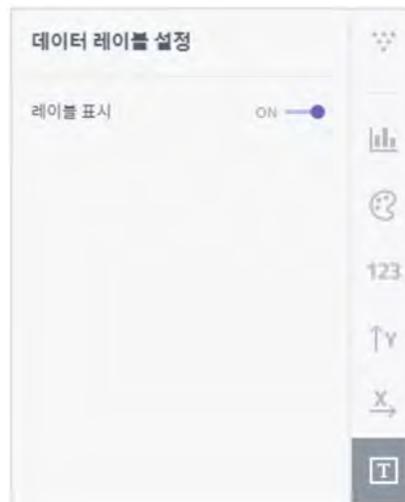


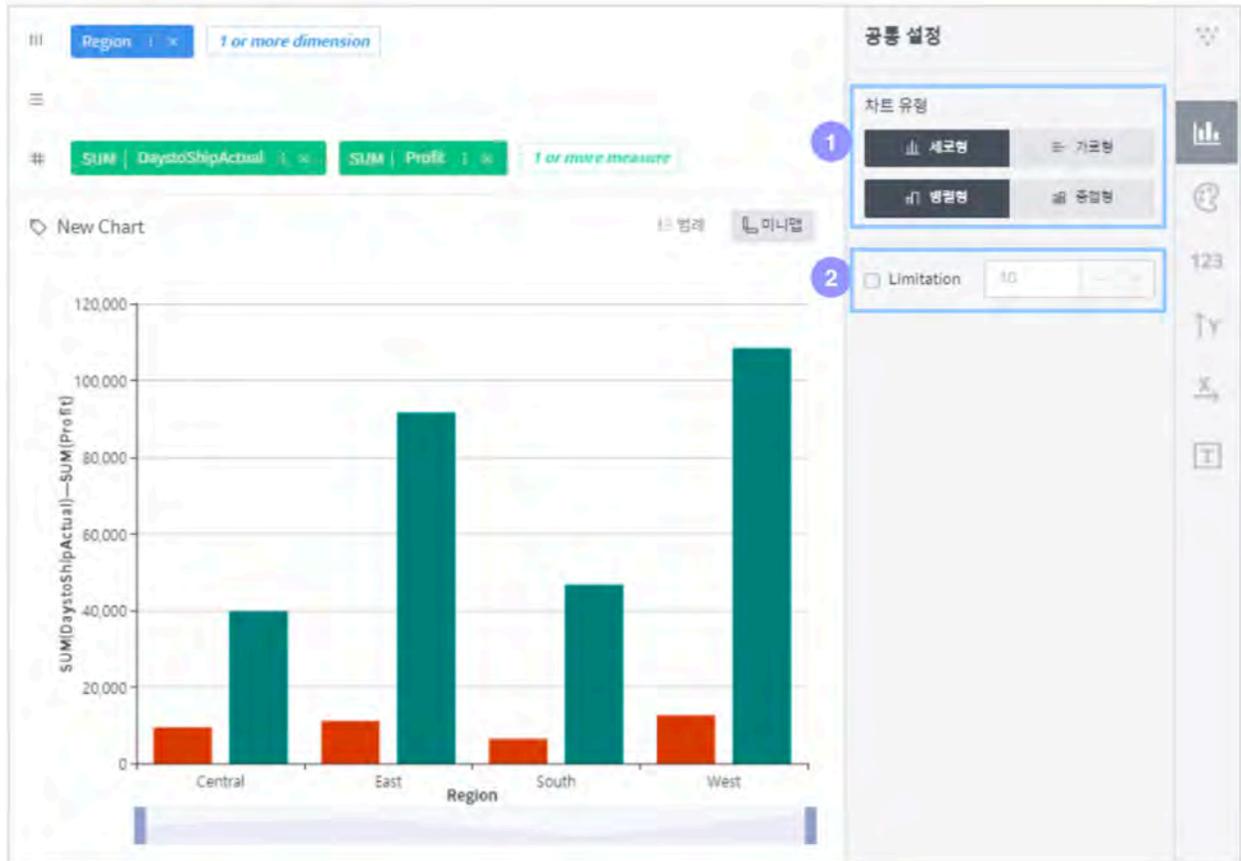
차트 유형별 공통 설정 항목

가장 보편적으로 사용되는 상위 6개 차트 (막대형 차트, 표, 선형 차트, 분산형 차트, 히트맵, 원형 차트)의 스타일 설정 방법에 대해 설명합니다.



막대형 차트

차원값 컬럼을 구성하는 각 범주 항목에 속한 데이터 값이 막대 모양으로 표시됩니다.



1. 차트 유형

- **세로형**: 차원값 축이 세로를 기준으로 하여 데이터 값 막대가 세로로 나타납니다.
- **가로형**: 차원값 축이 가로를 기준으로 하여 데이터 값 막대가 가로로 나타납니다.
- **병렬형**: 측정값을 2개 이상 선택했을 때 측정값별로 각각 다른 막대로 병렬하여 나타냅니다.
- **중첩형**: 측정값을 2개 이상 선택했을 때 모든 측정값을 한 막대에 중첩시켜 나타냅니다.

2. **Limitation**: 차트에 나타나는 컬럼의 개수를 결정합니다.

표

열/행 선반에 올려놓은 차원값 컬럼들의 범주 항목을 토대로 표 블록이 형성되며 그에 상응하는 측정값이 교차 영역에 텍스트로 표시됩니다.

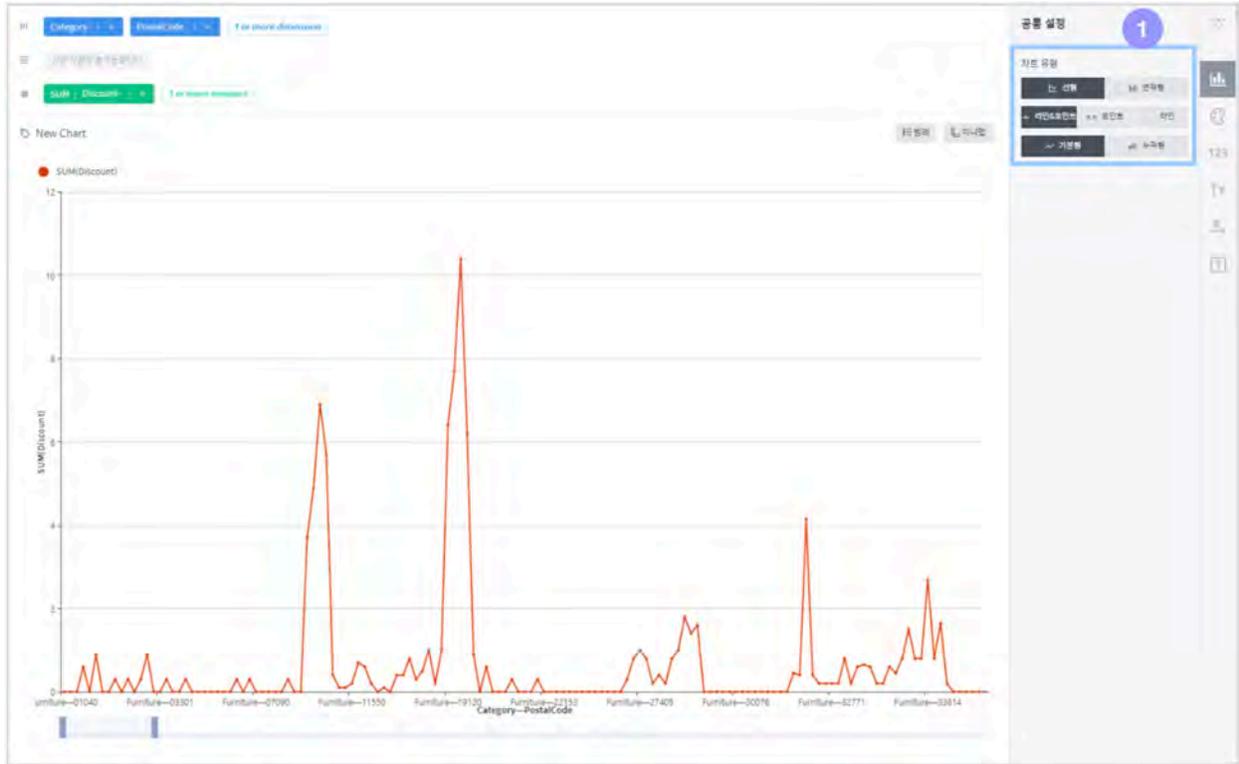
1. 차트 유형

- **피벗 데이터**: 분류 기준이 되는 차원값 범주가 동일한 측정값끼리는 한 셀에 집계 (SUM, MIN, MAX 등) 되는 방식으로 보여집니다.
- **원본 데이터**: 원본 측정값이 집계되지 않은 채로 특정한 차원값 컬럼을 기준으로 전부 출력됩니다.
- **세로보기**: 측정값의 데이터를 표에서 세로로 표시합니다. 원본 데이터 유형으로 표를 나타낼 때는 사용할 수 없습니다.
- **가로보기**: 피벗데이터 유형으로 표를 나타낼 경우 표를 가로보기 할 수 있습니다. 측정값의 데이터를 표에서 가로로 표시합니다.

2. **Show Head Colum**: 헤드 칼럼에 표시되는 텍스트 정렬 방식을 가로와 세로 별로 설정할 수 있습니다. 원본 데이터로 표시할 경우에는 헤드 칼럼은 필수로 표시됩니다. 피벗 데이터 유형일 경우에 헤드 칼럼을 사용하지 않을 수 있습니다.

선형 차트

차원값 컬럼을 구성하는 각 범주 항목에 속한 데이터 값이 점 모양으로 표시됩니다. 인접하는 범주 항목의 점끼리는 서로 연결되어 변화 추이를 확인할 수 있습니다.

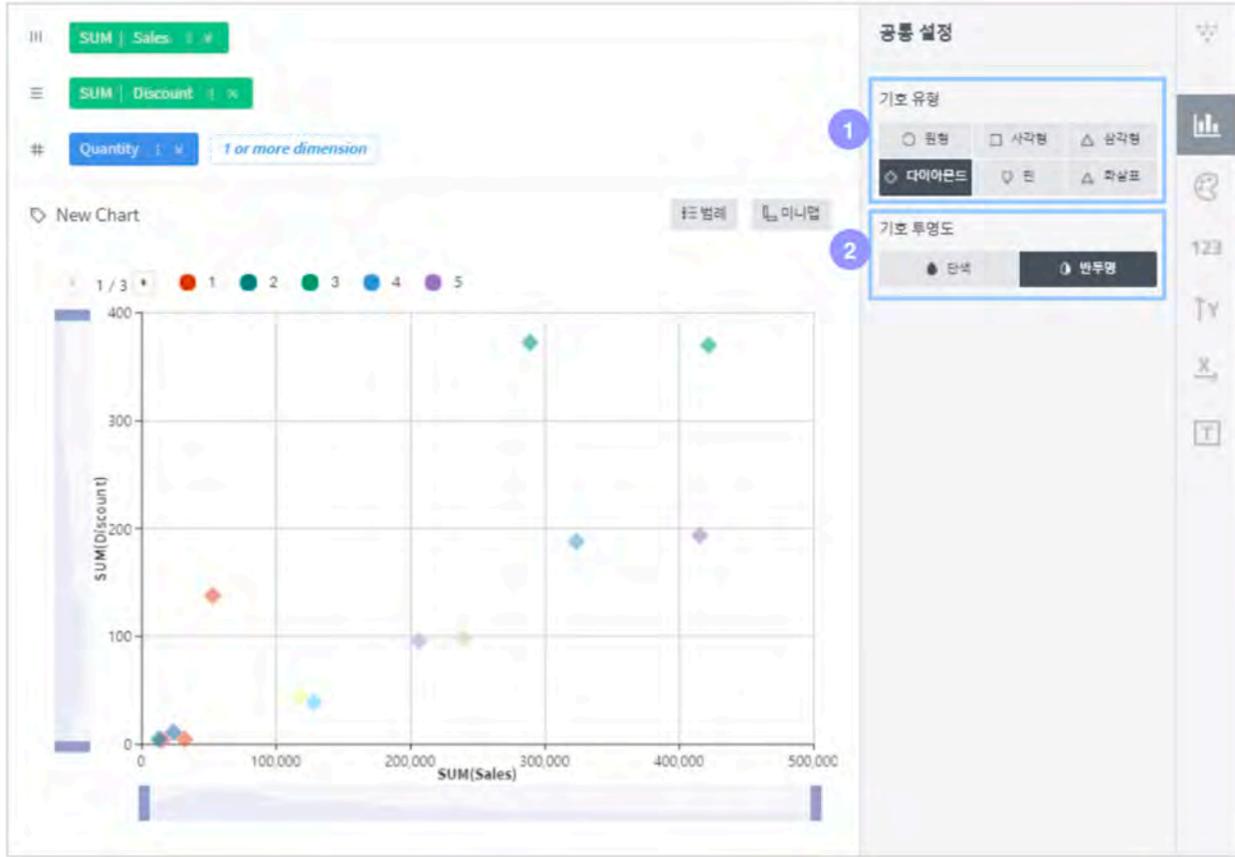


1. 차트 유형

- 선형: 측정값을 기준으로 선을 이어서 차트를 나타냅니다.
- 면적형: 선으로 이어진 면적에 색상을 입혀 차트를 나타냅니다.
- 라인&포인트: 측정값을 기준으로 한 점과 그 점을 연결한 선을 함께 나타냅니다.
- 포인트: 포인트는 측정값을 기준으로 한 점만 나타냅니다.
- 라인: 선의 연결만을 나타냅니다.
- 기본형: 측정값을 그대로 차트에 나타냅니다.
- 누적형: 측정값을 누적인 값을 차트에 나타냅니다.

분산형 차트

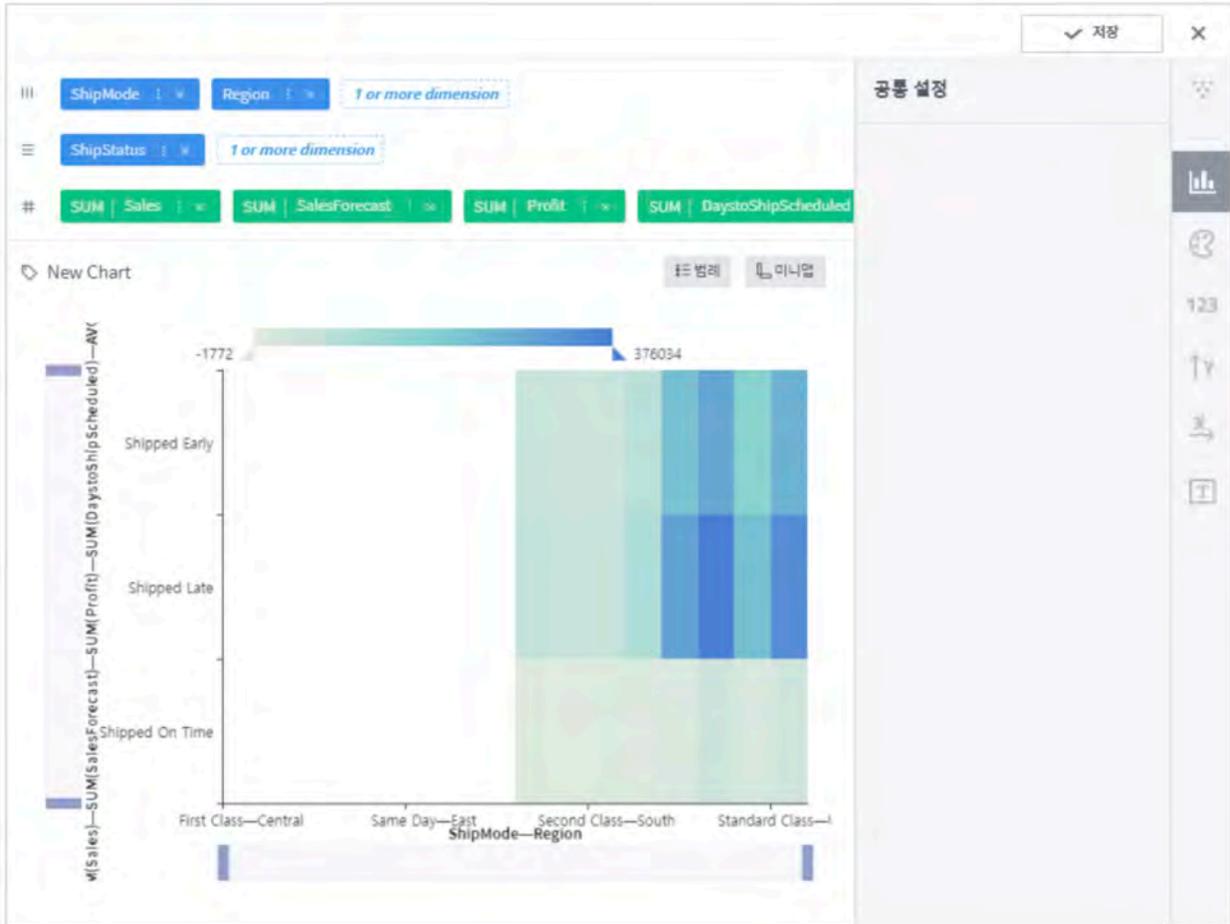
차원값 컬럼을 구성하는 각 범주 항목에 속한 데이터 값이 정의된 기호 모양으로 표시됩니다.



1. 기호 유형: 차트에 표시되는 기호의 모양을 설정합니다.
2. 기호 투명도: 차트에 표시되는 기호의 투명도를 설정합니다. 단색/반투명 중 선택하여 나타낼 수 있습니다.

히트맵

교차선반에 올려진 측정값 컬럼의 각 데이터 값이 색상으로 표시됩니다. 데이터 값이 클수록 색상 농도가 짙어집니다. 히트맵의 공통 설정 항목에는 설정할 사항이 없습니다.



원형 차트

차원값 컬럼의 각 범주 항목별로 차지하는 비중을 시각화하는 차트입니다.

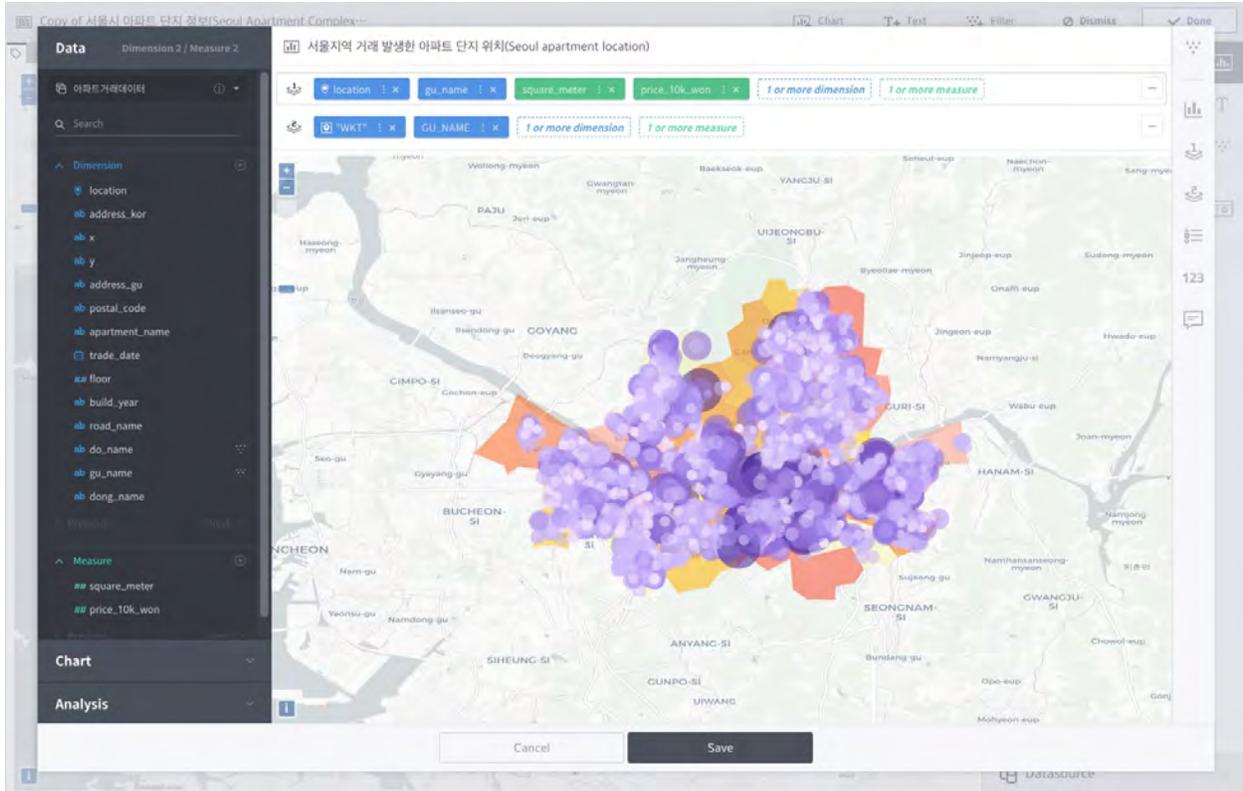


1. 차트 유형

- 부채꼴형: 차트가 원형으로 나타납니다.
- 도넛형: 차트가 도넛형으로 나타납니다.

5.3.5 맵뷰와 공간 연산

메타트론 디스커버리는 3.1.0 버전부터 위치 데이터를 시각화할 수 있는 맵뷰 기능을 제공하고 있습니다. 맵뷰는 기존 차트 유형들과는 다른 차트 생성 조건을 갖고 있습니다.



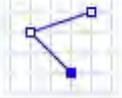
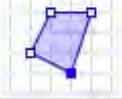
- 위치 속성 의 차원값이 1개 이상 필요합니다.
- 열/행/교차 선반이 아닌 **맵 레이어 선반** 에 데이터를 배치합니다.
- 레이어 별 스타일 속성 을 지정합니다.
- 공간 연산 이 가능합니다.

위치 속성의 차원값

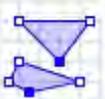
맵뷰를 사용하기 위해서는 Point, LineString, Polygon 과 같은 WKT Geometry 형식의 데이터로 이루어진 차원값 (dimension) 칼럼을 레이어 선반에 올려야 합니다. 위치값의 종류는 크게 세 가지 종류가 있습니다.

- **Point**: x, y로 이루어진 2D 좌표 지오메트리 타입입니다. GPS 데이터처럼 위도와 경도값이 있는 경우입니다.
- **Line**: 라인 좌표를 가진 지오메트리 타입입니다. WKT 형식으로 만들어진 LineString, MultiLineString 지오메트리를 지원합니다.
- **Polygon**: 도형 좌표를 가진 지오메트리 타입입니다. WKT 형식으로 만들어진 Polygon, MultiPolygon 지오메트리를 지원합니다.

Geometry primitives (2D)

Type	Examples	
Point		POINT (30 10)
LineString		LINESTRING (30 10, 10 30, 40 40)
Polygon		POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10))
		POLYGON ((35 10, 45 45, 15 40, 10 20, 35 10), (20 30, 35 35, 30 20, 20 30))

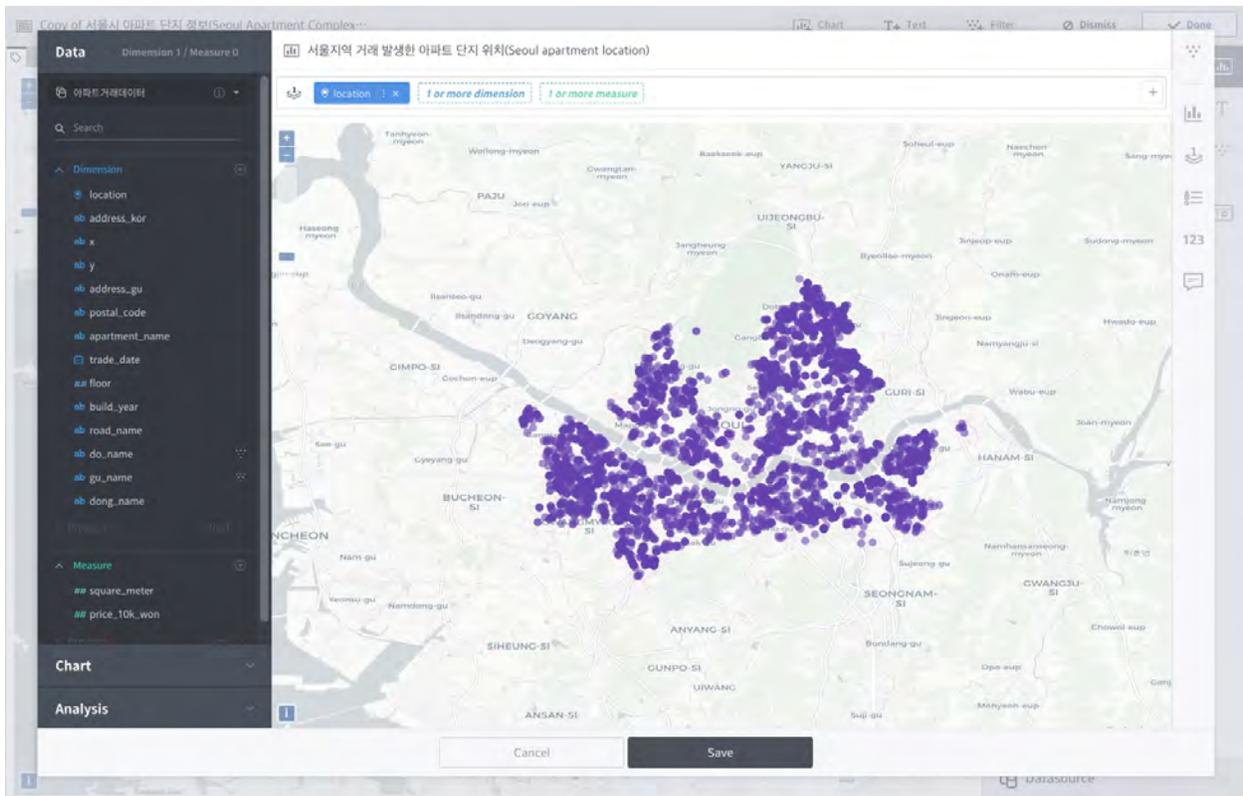
Multipart geometries (2D)

Type	Examples	
MultiPoint		MULTIPOINT ((10 40), (40 30), (20 20), (30 10))
		MULTIPOINT (10 40, 40 30, 20 20, 30 10)
MultiLineString		MULTILINESTRING ((10 10, 20 20, 10 40), (40 40, 30 30, 40 20, 30 10))
MultiPolygon		MULTIPOLYGON (((30 20, 45 40, 10 40, 30 20)), ((15 5, 40 10, 10 20, 5 10, 15 5)))
		MULTIPOLYGON (((40 40, 20 45, 45 30, 40 40)), ((20 35, 10 30, 10 10, 30 5, 45 20, 20 35), (30 20, 20 15, 20 25, 30 20)))
GeometryCollection		GEOMETRYCOLLECTION (POINT (40 10), LINESTRING (10 10, 20 20, 10 40), POLYGON ((40 40, 20 45, 45 30, 40 40)))

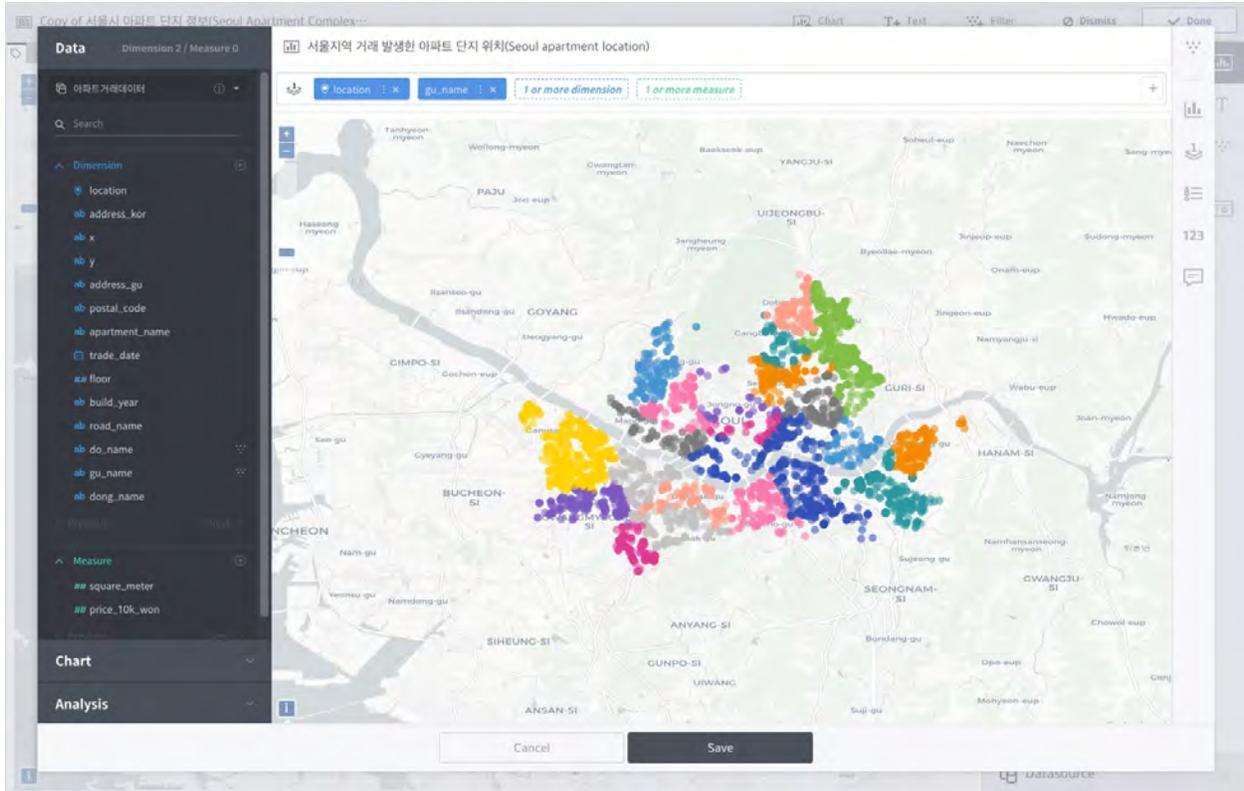
맵 레이어 선반



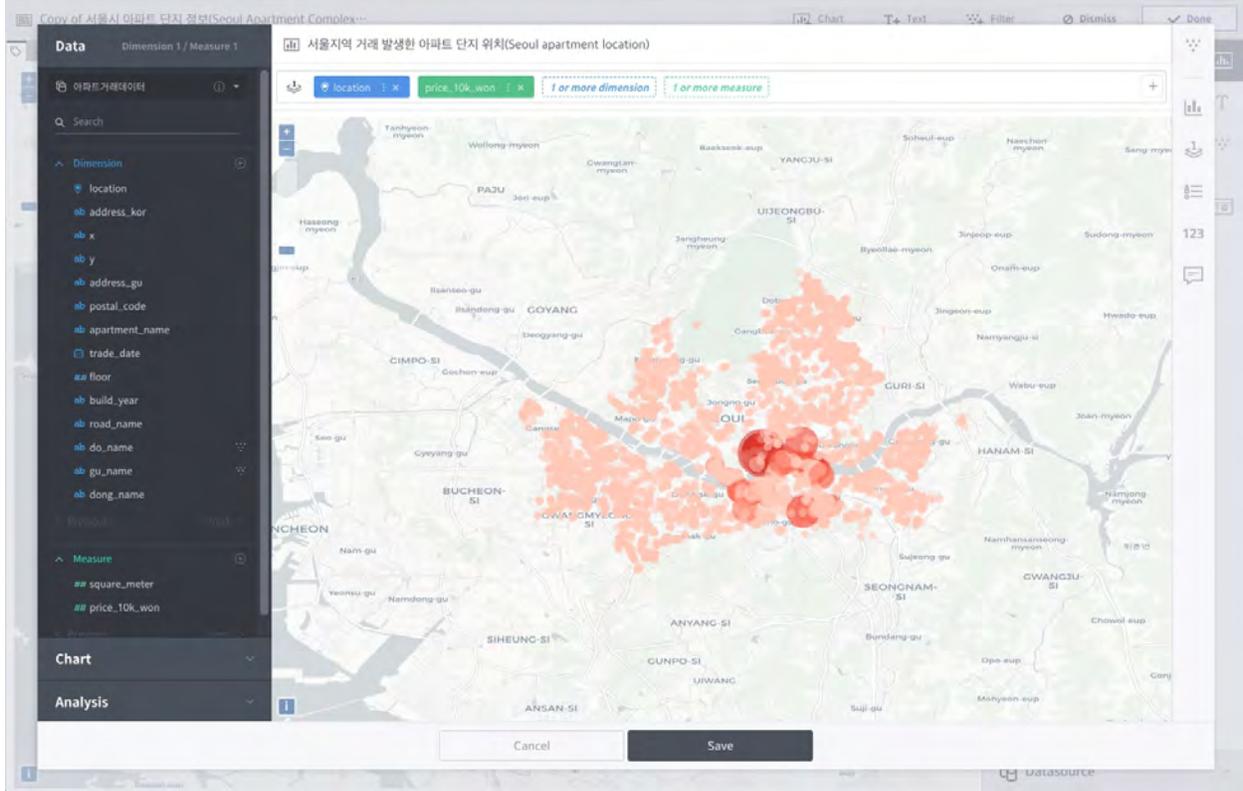
맵뷰에서는 다른 차트 유형과는 달리 열/행/교차 선반이 아닌 맵 레이어 선반을 갖고 있습니다. 맵 레이어 선반에는 반드시 위치 속성의 차원값을 1개 배치해야 합니다.



맵 레이어 선반에 문자 속성의 차원값을 배치하면 자동으로 해당 차원값으로 색상을 분류하여 표현하며 데이터 툴팁에 해당 차원값이 표기됩니다.

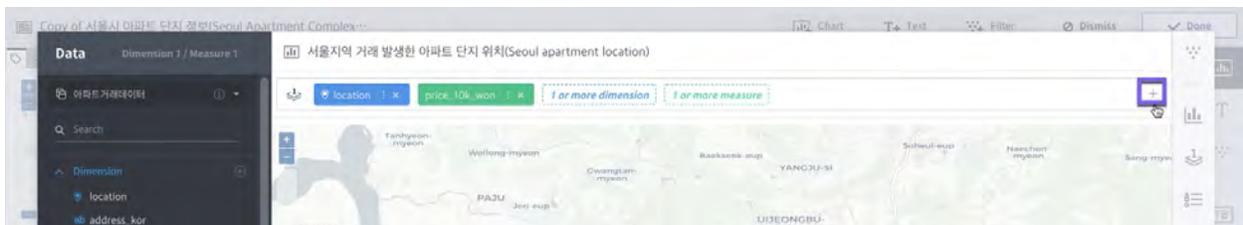


또한 측정값을 레이어 선반에 배치하면 측정값으로 색상을 분류하고 동시에 해당 측정값을 기준으로 포인트 크기를 다르게 표현합니다. 차원값과 마찬가지로 툴팁에 해당 측정값이 표기됩니다.



레이어 선반 추가

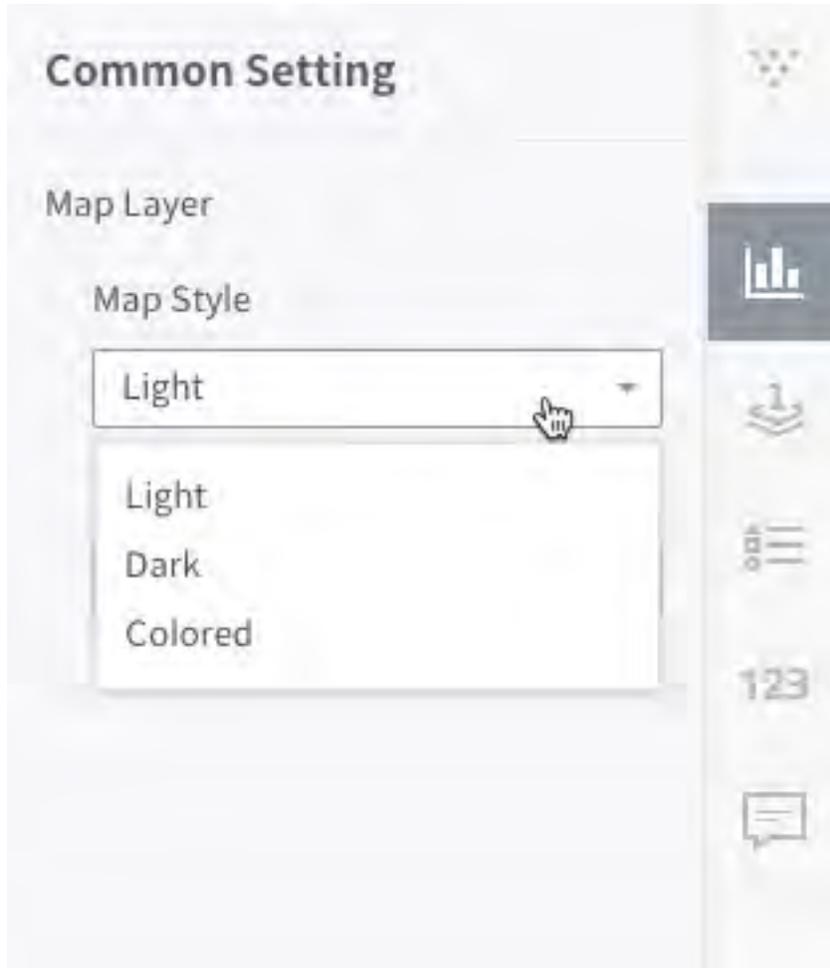
레이어 선반 우측의 + 버튼을 누르면 첫번째 레이어 위에 또 하나의 레이어를 추가할 수 있습니다. 각각의 레이어는 서로 다른 데이터소스를 사용해야 하며, 하나의 레이어에 두 개 이상의 데이터소스의 칼럼을 배치할 수 없습니다. 현재 최대 2 개의 레이어 선반을 지원합니다.



맵뷰 레이어 스타일 속성

공통 설정

지도 레이어에서 기본 지도를 표현하는 맵 스타일의 유형을 선택할 수 있습니다. OpenStreetMap을 활용하여 세 가지의 맵 스타일을 기본적으로 제공하고 있습니다.



- Open Street Map **Light** (기본값)
- Open Street Map **Dark**
- Open Street Map **Colored**

레이어 설정

레이어의 표현 방식을 설정합니다. 레이어 선반을 추가하면 1번 레이어와 2번 레이어에 대한 설정 메뉴가 각각 별도로 생성됩니다.

Point 타입 레이어 속성

Layer Setting

Layer Name
Layer1

Layer Type
Point Heatmap Hexagon Cluster

Point Type
 Circle
 Square
 Triangle

Color
 Color by
 ab gu_name
 Transparency
 50 %

Size
 Size by
 ## square_meter
 Radius Range
 5 px ~ 20 px

Outline
 OFF

1. **레이어 이름:** 맵뷰의 범례 및 툴팁 설정시 나타나는 레이어 명칭을 설정합니다.
2. **레이어 유형:** 데이터 포인트를 지도에 나타내는 유형을 포인트/히트맵/hexagon/클러스터 중 하나로 변경할 수 있습니다. 기본값은 포인트입니다.
3. **포인트 유형:** 레이어 유형이 포인트 인 경우 데이터 포인트의 모양을 변경할 수 있습니다. 원형/사각형/삼각형으로 표현 가능하며 기본값은 원형입니다. 클러스터 사용이 OFF 로 설정되어야 맵에 표현됩니다.
4. **색상:** 레이어 선반에 올린 문자 속성의 차원값이나 측정값으로 데이터 포인트의 색상을 구분하여 표현할 수 있습니다. 색상 기준이 없는 경우 팔레트에서 색상을 변경할 수 있습니다. 투명도를 %로 설정할 수 있습니다.
5. **크기:** 레이어 유형이 포인트 인 경우 레이어 선반에 올린 측정값을 기준으로 데이터 포인트의 크기를 구분하여 표현할 수 있습니다.
6. **아웃라인:** ON으로 설정시 각 데이터 포인트의 아웃라인을 그립니다. 기본값은 OFF이며 색상 및 굵기를 설정할 수 있습니다.
7. **클러스터 범위:** 레이어 유형이 클러스터 인 경우 클러스터링 범위를 %로 지정할 수 있습니다. 데이터 포인트가 많을수록 클러스터를 사용하는 것이 브라우저 성능에 유리합니다.
8. **흐림효과:** 레이어 유형이 히트맵 일 경우 히트맵의 흐림 효과를 조절할 수 있습니다. 기본값은 20%입니다.
9. **반경값:** 레이어 유형이 히트맵 또는 hexagon 일 경우 표시 반경을 1부터 100사이의 값으로 조절할 수 있습니다.

Line 타입 레이어 속성

Layer Setting

Layer Name
Layer1

Color

Color by
Dimension

ONEWAY

Transparency
10 36

Stroke

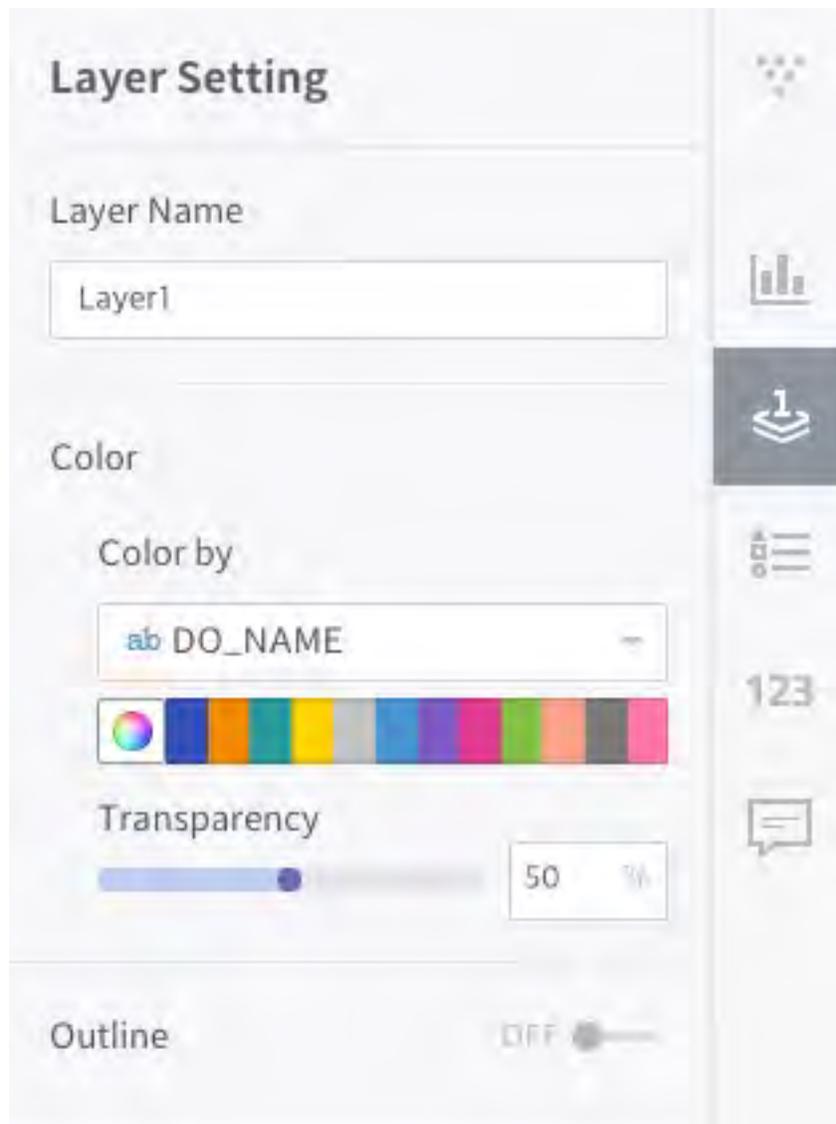
Stroke by
None

Max Stroke Width (pt) 2

LineType

1. **레이어 이름:** 맵뷰의 범례 및 툴팁 설정시 나타나는 레이어 명칭을 설정합니다.
2. **색상:** 레이어 선반에 올린 문자 속성의 차원값이나 측정값으로 데이터 포인트의 색상을 구분하여 표현할 수 있습니다. 색상 기준이 없는 경우 팔레트에서 색상을 변경할 수 있습니다. 투명도를 %로 설정할 수 있습니다.
3. **굵기:** 라인의 굵기를 설정할 수 있습니다.
4. **선 유형:** 실선/점선/파선 중 하나를 선택합니다. 기본값은 실선입니다.

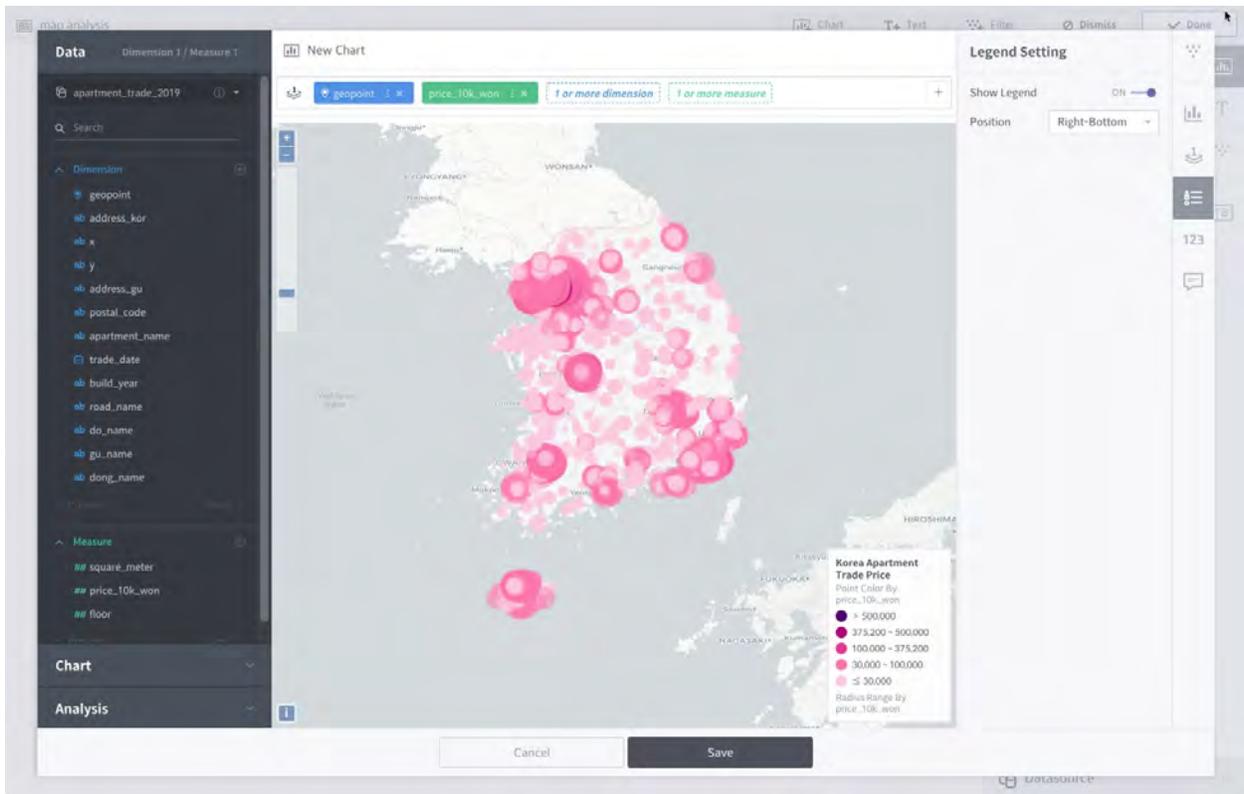
Polygon 타입 레이어 속성



1. **레이어 이름:** 맵뷰의 범례 및 톨팁 설정시 나타나는 레이어 명칭을 설정합니다.
2. **색상:** 레이어 선반에 올린 문자 속성의 차원값이나 측정값으로 데이터 포인트의 색상을 구분하여 표현할 수 있습니다. 색상 기준이 없는 경우 팔레트에서 색상을 변경할 수 있습니다. 투명도를 %로 설정할 수 있습니다.
3. **아웃라인:** ON으로 설정시 각 폴리곤의 아웃라인을 그립니다. 기본값은 OFF이며 색상 및 굵기를 설정할 수 있습니다.

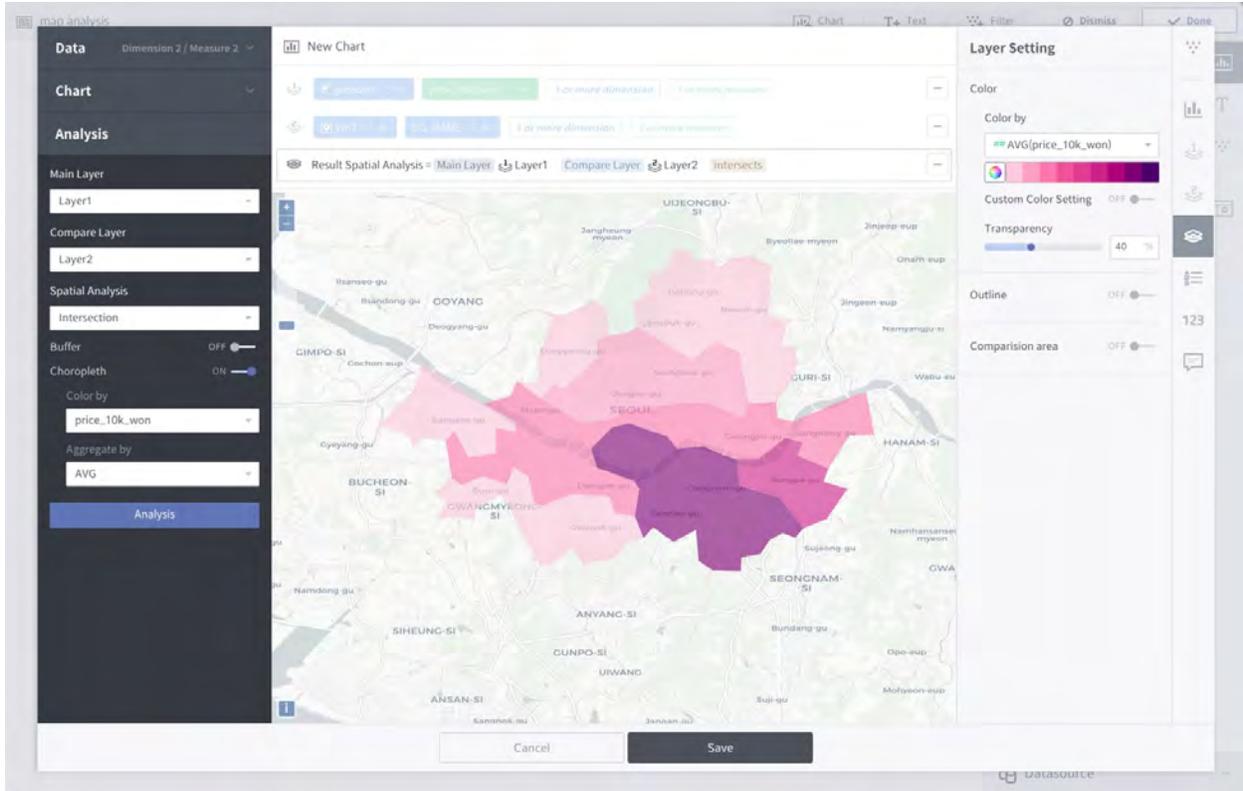
범례 설정

범례 표시 여부를 설정할 수 있습니다. 기본값은 OFF 이며, ON으로 변경 시 범례의 위치를 설정할 수 있습니다.



공간 분석

메타트론 디스커버리 맵뷰에서는 두 개의 레이어 간의 간단한 공간 분석 기능을 지원합니다. 왼쪽 분석 탭에서 연산식을 설정할 수 있으며, 현재 버전에서 공간 연산식은 두 종류를 지원합니다.



- **Within:** 기준 레이어의 요소와 비교 레이어의 요소 사이 거리를 지정하여 거리 내에 존재하는 값을 반환합니다.
- **Intersection:** 기준 레이어에서 비교 레이어와 겹치는 부분을 반환하는 방식입니다. Polygon > Line > Point의 순서로 더 큰 Geometry가 기준이 되는 경우, 결과 반환값이 달라질 수 있습니다.

각 연산식에서 추가적으로 설정할 수 있는 값은 아래와 같습니다.

- **근접 거리 입력 (Buffer):** 기준 레이어와 비교 레이어 간에 비교할 거리를 숫자로 입력하도록 설정할 수 있습니다. 거리 단위를 미터 또는 킬로미터로 변경할 수 있습니다.
- **단계구분도 보기 (Choropleth map):** 연산 결과 레이어를 단계구분도 형태로 표시할 수 있습니다. 단계구분도의 색상 기준을 선택할 수 있으며 겹치는 데이터의 수 (COUNT) 를 기본으로 색상을 나눕니다. 만약 기준 레이어에 측정값이 있을 경우 해당 측정값을 기준으로 색상을 변경할 수 있습니다.

5.4 필터

필터는 대시보드와 차트를 구성할 때 조건에 일치하는 데이터만 표출하도록 설정하는 기능입니다. 차트의 종류는 차트 필터, 글로벌 필터 두 가지로 구성됩니다. 차트 필터는 개별 차트에 적용되는 필터이고, 글로벌 필터는 하나의 대시보드 전체에 적용되는 필터입니다.

5.4.1 차트 필터

차트 필터는 각 컬럼별로 차트에 나타낼 데이터의 범위를 한정 짓는 기능을 합니다. 본 챕터에서는 차트 필터를 지정하고 활용하는 방법에 대해 설명합니다.

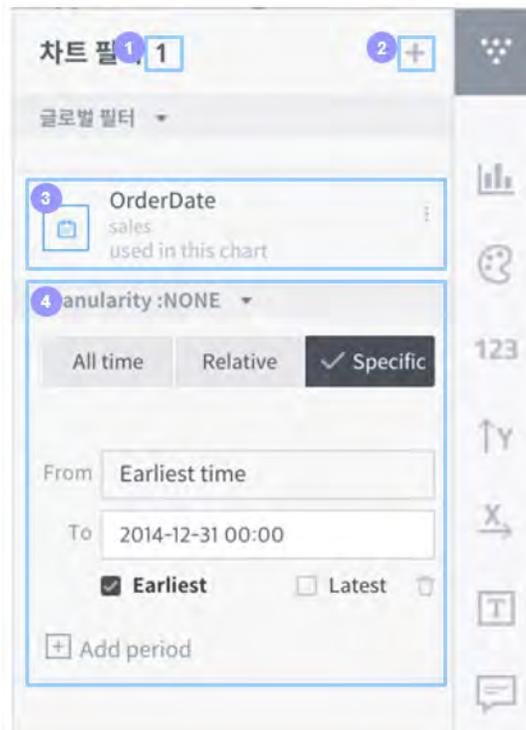
자동으로 포함되는 필터들

다음에 해당하는 컬럼들의 필터는 별도의 차트 필터를 추가하지 않아도 자동으로 포함됩니다.

- **타임스탬프 컬럼 필터**: Metatron 엔진의 시계열 특성 때문에 시간 조건 필터링이 필수적으로 사용됩니다.
- **추천 필터**: 데이터 소스 등록할 때 <추천 필터> 로 지정된 컬럼 필터들입니다.
- **글로벌 필터가 적용된 대시보드**: 해당 대시보드에 등록된 모든 차트에 공통적으로 적용되는 필터입니다.

차트 필터 패널

차트 홈 화면 우측에는 차트 필터 패널이 있습니다. 이 패널에서는 등록된 필터를 간단하게 조회·설정할 수 있습니다.



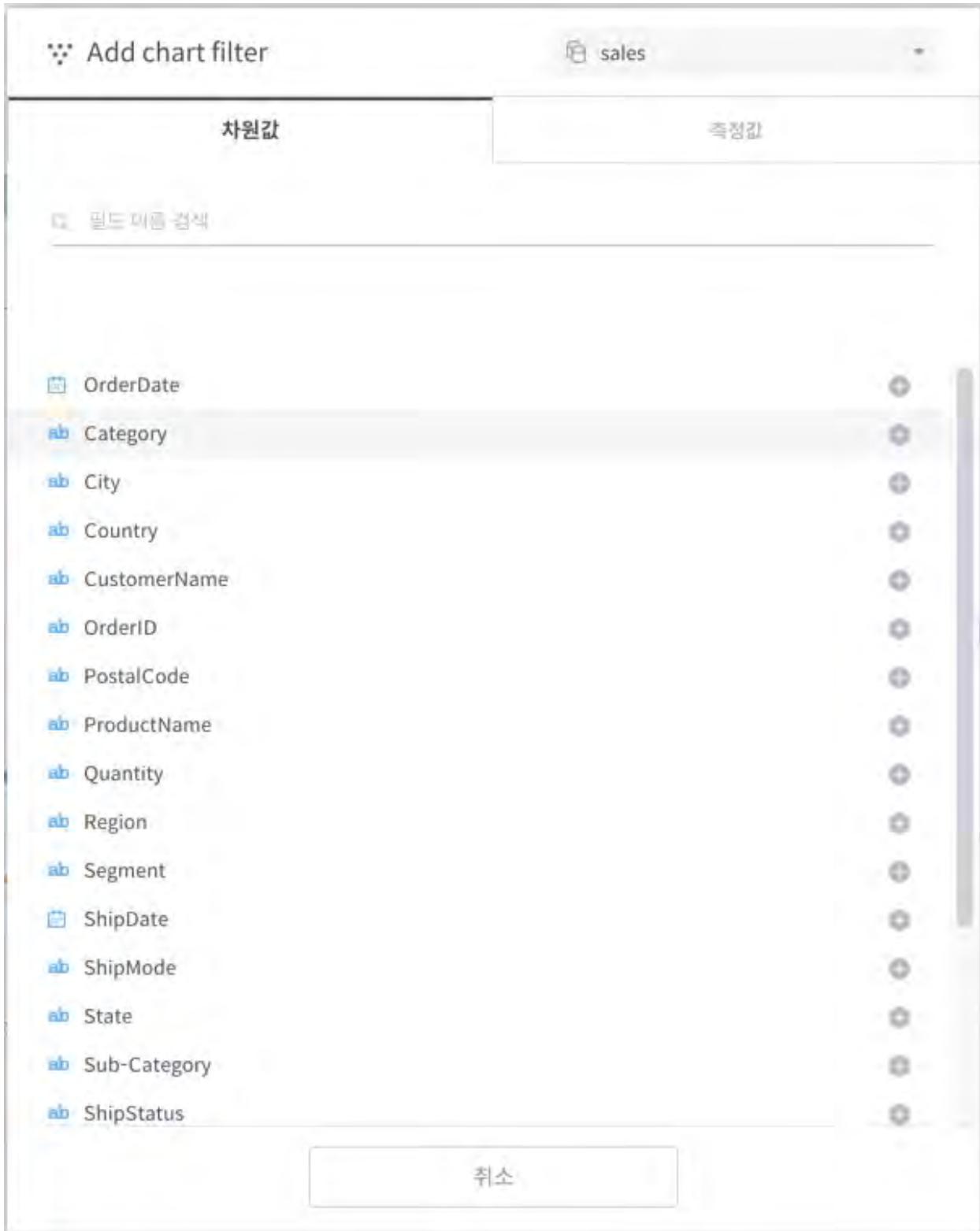
1. **필터 수**: 차트 패널 이름 옆에는 현재 적용된 차트가 몇 개인지 표시됩니다.

2. **필터 추가/변경:** 우측 상단의 (+) 버튼을 누르면 새로운 필터를 추가하거나 기존 필터를 세부 설정하는 팝업이 나타납니다.
3. **필터 대상 컬럼:** 개별 필터의 상단에는 필터가 적용되는 컬럼 정보를 보여줍니다.
4. **필터 상세 설정:** 개별 필터 우측 상단에 햄버거 메뉴를 누르면 필터를 초기화하거나 상세 설정할 수 있습니다.

차트 필터 대화 상자

차트 필터 패널 상단에서 버튼을 클릭하거나 각 필터 영역에서 버튼을 클릭하면 차트 필터 대화 상자를 열 수 있습니다. 이 대화 상자에서는 새로운 필터를 추가하거나 기존 필터를 세부적으로 설정할 수 있습니다.

차트 필터 대화 상자는 다음과 같이 차원값과 측정값 영역으로 구성됩니다.



차원값 필터링

해당 차트와 연동된 데이터 소스의 차원값을 필터로 지정할 수 있습니다.

The screenshot displays the 'Region' filter configuration page. At the top left, there is a back arrow, a logo with 'ab', and the title 'Region sales'. On the top right, there is a 'New Chart' button. Below the title, there are two tabs: '단건' (Single) and '다건' (Multiple), with '단건' selected. A search bar contains the text '아이템 이름으로 검색해 주세요'. To the right of the search bar are icons for filter type (funnel), sort (up/down arrows), and visibility (eye). Below the search bar is a list of filter options: '(모두)' (All), 'Central 2323', 'East 2848', 'South 1620', and 'West 3203'. To the right of this list is a 'Turn all on/off' toggle. At the bottom left, there is a checked 'All' checkbox. At the bottom, there is a 'Defined value' input field and a '추가' (Add) button. At the very bottom, there are two buttons: '취소' (Cancel) and '마침' (Finish).

- **범위 선택:** 선택한 필터의 컬럼에 들어있는 데이터 범주 중 필터링하여 차트에 표시할 범위를 선택합니다.
 - 단건: 하나의 데이터 범주만 선택하여 차트에서 표시할 수 있습니다.
 - 다건: 여러 개의 데이터 범주를 선택하여 차트에 표시할 수 있습니다.
- **검색:** 컬럼 속성 값이 너무 많은 경우, 원하는 결과만 볼 수 있도록 제한할 수 있습니다.
 - **이름으로 검색:** 컬럼의 속성값 이름으로 검색할 수 있습니다.
 - **속성 필터링:** 속성 값 이름을 정규식이나 와일드 카드로 매칭하거나, 측정값의 범위를 기준으로 조건을 걸어서 속성을 선별할 수 있습니다.

The screenshot shows a filter configuration window with the following sections:

- Matcher:** Two buttons, 'Wildcard' (checked) and 'Regular Expression'.
- Starting Value:** A text input containing 'c' and a dropdown menu labeled 'Starting Word'.
- Condition:** A text label 'Profit' followed by a minus sign, a dropdown menu '합계', a comparison operator '>=', and a text input '10'.
- Limitation:** A dropdown menu '상위', a text input '10', a dropdown menu '측정값 선택', and a dropdown menu '합계'.
- Buttons:** '초기화' (Reset) and '적용' (Apply).

- **Defined value:** 컬럼에 들어있지 않은 속성값을 필터 조건으로 추가하는 데 사용합니다. 현재 데이터 소스에는 없지만 추후에 들어올 수 있는 데이터를 미리 예측하여 필터를 생성하는 기능입니다.

타임스탬프 컬럼 필터 설정

시간 아이콘이 표시된 차원값은 타임스탬프 타입이며 해당 차원값은 타임스탬프 필터를 설정할 수 있습니다. 기본적으로 전체 시간 (All time) 으로 설정되어 있으며, 특정 기간의 데이터만 차트에 표출하고 싶은 경우 Relative 또는 Specific을

선택하여 설정합니다.

Relative는 현재 시점을 기준으로 상대적인 기간을 설정하여 해당 기간 동안의 데이터만 차트에 표출하도록 합니다.

The screenshot shows the configuration interface for a chart titled "OrderDate" under the "sales" category. At the top right, there are buttons for "New Chart" and "Timestamp". Below this, the "Granularity" is set to "NONE".

The main configuration area is divided into three sections: "Previous", "Current", and "Next".

- Relative Selection:** Three buttons are shown: "All time", "Relative" (which is selected with a checkmark), and "Specific". Below these buttons, the selected period is displayed as "2019-02 W08 ~ 2019-02 W09".
- Previous Section:** Contains buttons for "Yesterday", "Last Week", "Last Month", and "Last Year". Below these is a "Last" section with a dropdown menu showing "1" and "주" (week).
- Current Section:** Contains buttons for "Today", "This Week", "This Month", and "This Year".
- Next Section:** Contains buttons for "Tomorrow", "Next Week", "Next Month", and "Next Year". Below these is a "Next" section with a dropdown menu showing "1" and "주" (week).

At the bottom of the interface, there are two buttons: "취소" (Cancel) and "마침" (Finish/Apply).

Specific은 데이터의 특정 기간을 직접 설정하여 해당 기간 동안의 데이터만 차트에 표출하도록 합니다.

The screenshot shows the configuration interface for a chart titled "OrderDate" under the "sales" category. At the top right, there are buttons for "New Chart" and "Timestamp". Below this, the "Granularity" is set to "NONE". There are three tabs for time selection: "All time", "Relative", and "Specific", with "Specific" being the active tab. The "From" date is set to "2011-01-04 00:00" and the "To" date is set to "2014-12-31 00:00". There are checkboxes for "Earliest" and "Latest", both of which are currently unchecked. An "Add period" button is located below the date fields. At the bottom of the interface, there are two buttons: "취소" (Cancel) and "마침" (Finish), with "마침" being the active button.

측정값 필터링

해당 차트와 연동된 데이터 소스의 측정값을 필터로 지정할 수 있습니다.

❖ Add chart filter sales

차원값 측정값

🔍 필드 이름 검색

- Discount
- Profit
- Sales
- DaystoShipActual
- SalesForecast
- DaystoShipScheduled
- SalesperCustomer
- ProfitRatio

취소

필터로 설정할 측정값을 선택했으면 필터링할 값의 범위를 지정합니다.



← **Profit**
salés New Chart

Value range

Minimum Maximum

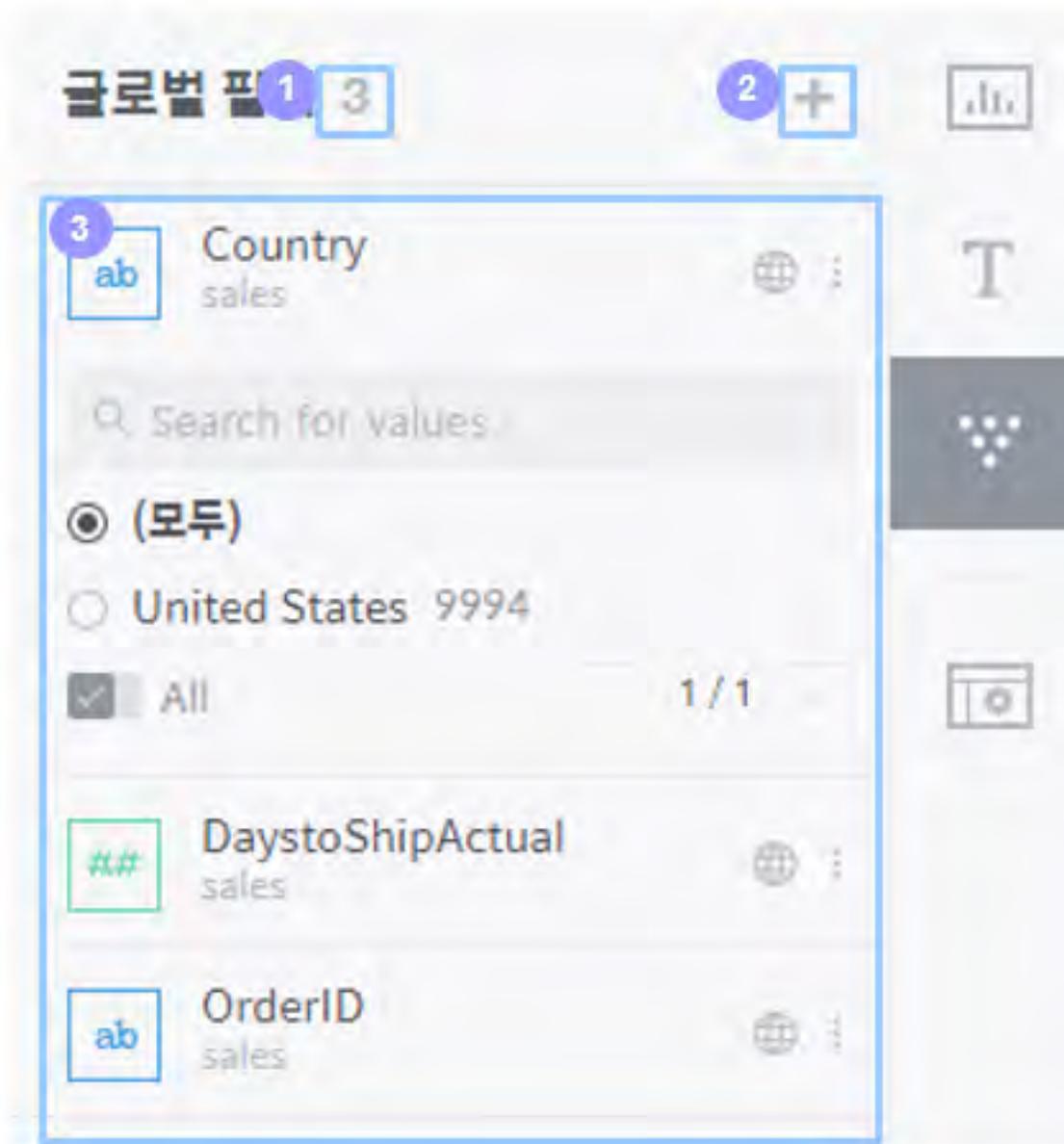
-6600 2984

Available input range: -6600 - 8400

취소 마침

5.4.2 글로벌 필터

글로벌 필터는 대시보드에 속한 모든 차트에 적용되는 데이터 표출 조건으로, 대시보드 편집 창의 필터 패널에서 추가, 수정, 삭제할 수 있습니다.



1. **필터 위젯 수:** 글로벌 필터 제목 옆에 현재 대시보드에 등록된 필터 위젯의 개수가 나타납니다.
2. **필터 위젯 추가:** 우측 상단의 (+)를 클릭하면 대시보드 내에 새로운 필터 위젯을 생성할 수 있습니다. 필터 생성 팝업 및 생성 방법은 직전 챕터의 차트 필터 생성 절차와 동일합니다.
3. **필터 위젯 목록:** 현재 대시보드에 등록된 필터 위젯들이 열거됩니다. 수정 또는 삭제를 원하는 위젯

항목에 마우스를 오버하면 이를 위한 아이콘이 나타납니다. 또한 위젯 항목을 위젯 레이아웃 영역으로 드래그하면 위젯이 위젯 레이아웃 영역에 표시됩니다.

전체 대시보드에 적용된 글로벌 필터는 차트 생성 시 개별 필터를 만들 때에 함께 조회됩니다. 또한 글로벌 필터 생성 시에도 개별 차트 필터가 있다면 어떤 칼럼에 생성되었는지 직관적으로 알려줍니다.

노트북



Metatron Discovery에서는 노트북 기능을 수행할 수 있습니다. 노트북이란, 라이브 코드, 등식, 시각화와 설명을 위한 텍스트 등을 포함한 문서를 만들고 공유할 수 있는 도구입니다. 주로 데이터 클리닝과 변형, 수치 시뮬레이션, 통계 모델링, 머신 러닝 등에 사용할 수 있습니다.

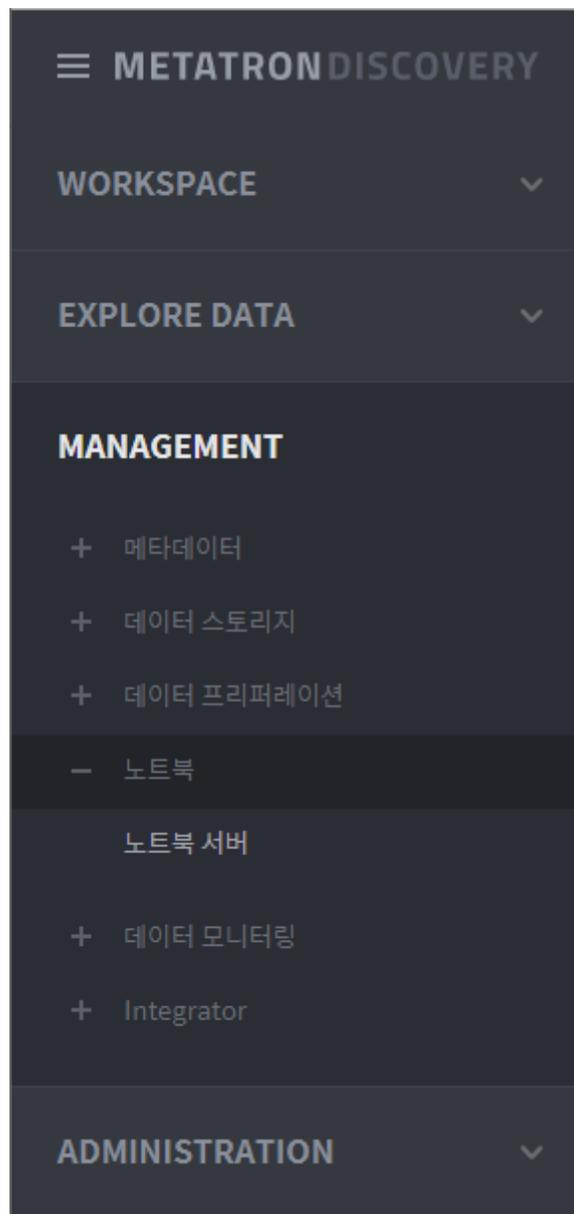
Metatron Discovery에서는 Jupyter와 Zeppelin을 등록하여 사용할 수 있도록 기능을 제공하고 있습니다. Jupyter는 데이터 과학 분야에서 주로 사용하는 Python, R 프로그래밍 언어를, Zeppelin은 Spark(Scala) 프로그래밍 언어를

활용하여 실시간으로 인터랙티브하게 데이터를 분석하고 시각화할 수 있도록 도와줍니다. 노트북을 수행하기 전, 노트북 서버 세팅이 되어있어야 합니다.

6.1 노트북 서버 관리

노트북 모듈의 사용을 허용하기 위해서는 먼저 관리자가 외부 분석 도구가 설치되어 있는 서버, 즉 <노트북 서버>와 연동을 해야 합니다.

메인 화면 좌측 패널에서 MANAGEMENT → 노트북 관리 → 노트북 서버 메뉴로 이동하면 노트북 서버를 새로 등록하거나 기존에 등록된 노트북 서버를 조회·수정할 수 있습니다.



6.1.1 노트북 서버 목록

이 화면에서는 노트북 서버들을 보여줍니다. 노트북 서버 목록은 서버 이름과 타입으로 필터링할 수 있으며, 목록에 나타난 서버 중 하나를 클릭하여 해당 서버의 정보를 열람·수정할 수 있습니다. 또한 서버 중 하나에 마우스를 오버하면 나타나는  버튼을 클릭하거나, 좌측에서 삭제하고자 하는 서버들의 체크란을 선택한 후 우측 상단에 있는 선택 삭제 버튼을 클릭하면 해당 노트북 서버가 삭제됩니다.

노트북

노트북 서버

타입 ALL

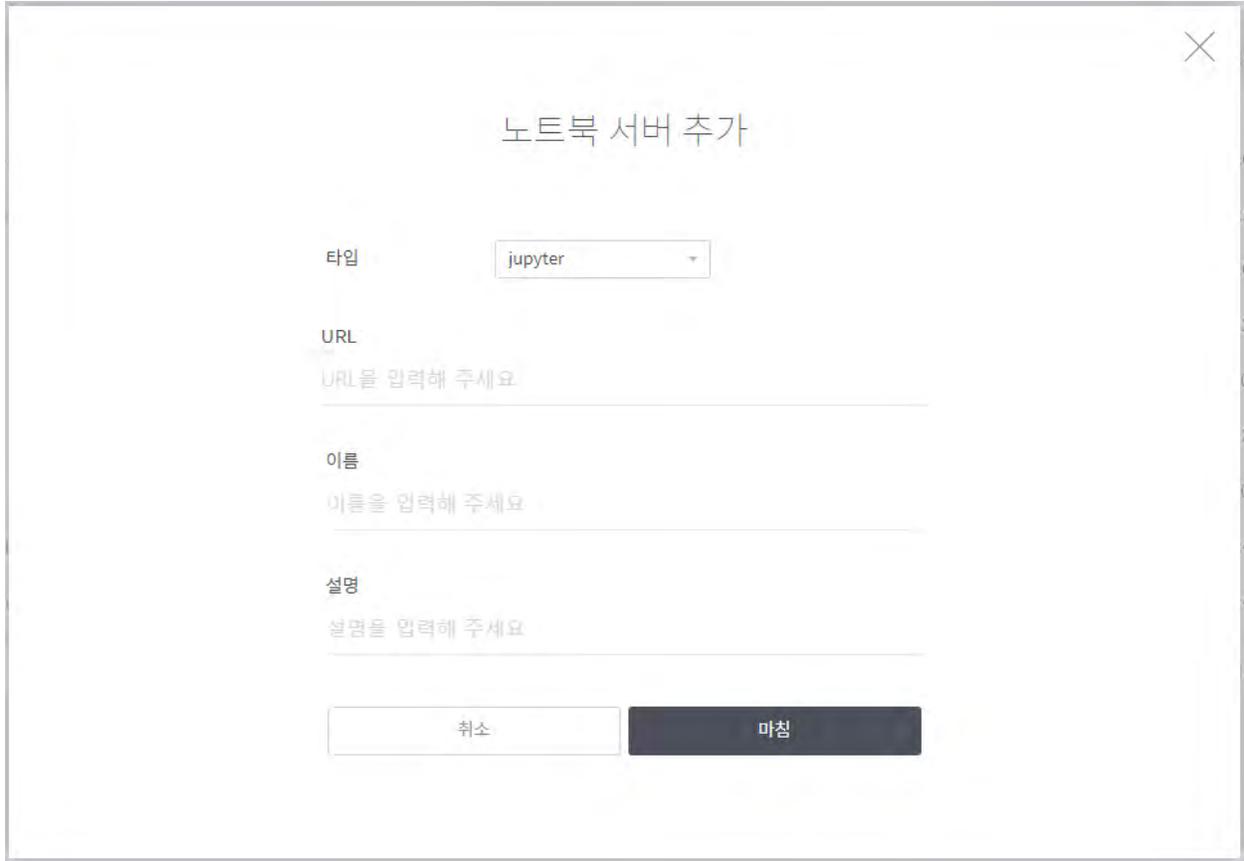
9개 데이터가 있습니다 [노트북 서버 추가](#) [시작 보기](#)

<input type="checkbox"/>	서버	타입	URL	수정일	생성일
<input type="checkbox"/>	QA_TEST2-test	zeppelin	http://jupyter.mcloud.sktelecom.com:80	2019-08-22 13:04 by admin	2019-08-22 13:01 by admin
<input type="checkbox"/>	QA_Test-test입니다 수정확인	jupyter	http://www.	2019-08-22 12:59 by admin	2019-08-22 12:58 by admin
<input type="checkbox"/>	jupyter-수정	jupyter	http://metatron-web-04:8888	2019-08-20 14:41 by admin	2019-07-02 19:03 by admin
<input type="checkbox"/>	asd-asd	zeppelin	https://zeppelin1.svc.stg.apm.cloud.metatr...	2019-07-22 15:06 by admin	2019-06-14 13:22 by admin
<input type="checkbox"/>	te1-테스트	zeppelin	http://52.231.201.148:8080	2019-05-20 09:50 by admin	2018-11-23 16:05 by admin
<input type="checkbox"/>	Zeppelin Dev-Metatron 개발서버에 구축된 Zeppelin	zeppelin	http://metatron-web-04:8080	2019-04-04 14:57 by admin	2019-03-21 14:35 by admin
<input type="checkbox"/>	te2	zeppelin	http://150.28.69.116:80	2018-11-23 16:06 by admin	2018-11-23 16:06 by admin
<input type="checkbox"/>	jupyter-default	jupyter	http://jupyter.mcloud.sktelecom.com:80	2018-08-24 15:49 by Polaris	2018-08-24 15:49 by Polaris
<input type="checkbox"/>	zeppelin-default	zeppelin	http://zeppelin.mcloud.sktelecom.com:80	2018-08-24 15:49 by Polaris	2018-08-24 15:49 by Polaris

1 / 20

6.1.2 노트북 서버 추가

노트북 관리 화면에서 노트북 서버 추가 버튼을 클릭하면 다음과 같은 노트북 서버 등록 화면이 팝업이 됩니다.



노트북 서버 추가

타입

URL
URL을 입력해 주세요

이름
이름을 입력해 주세요

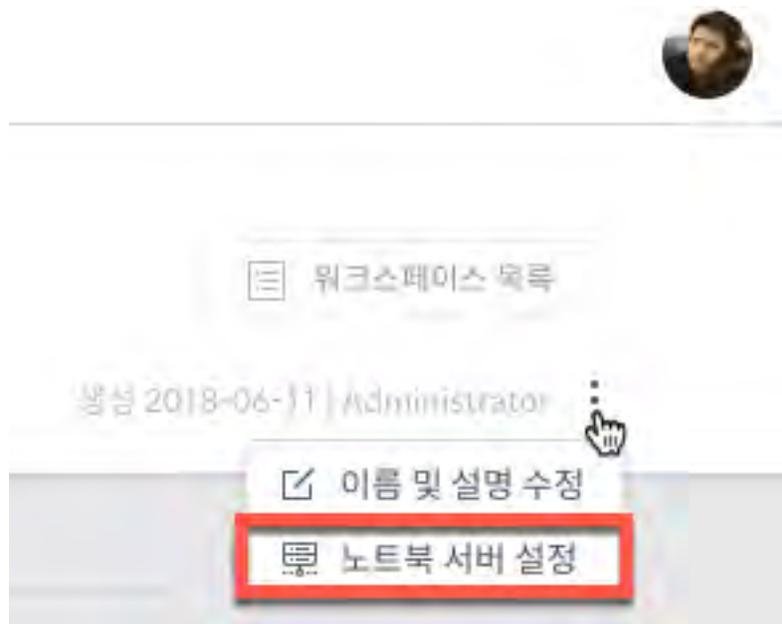
설명
설명을 입력해 주세요

- **타입:** 등록하려는 노트북 서버에 설치된 외부 분석 도구를 선택합니다. Jupyter와 zeppelin 중 선택이 가능합니다.
- **URL:** 등록하려는 노트북 서버의 URL을 입력합니다. http://와 https://를 지원합니다.
- **이름:** 등록하려는 노트북 서버의 이름을 입력합니다.
- **설명:** 등록하려는 노트북 서버에 대한 설명을 입력합니다.

6.2 노트북 서버 등록하기

워크스페이스에서 노트북 기능을 이용하여 데이터를 분석하기 위해서는 노트북 서버 초기 설정이 필요합니다. 노트북 서버 초기 설정 절차는 다음과 같습니다.

1. 워크스페이스의 우측 상단에 있는  버튼을 클릭한 후 노트북 서버 설정을 선택합니다.



2. 관리자가 사전에 등록해 둔 Jupyter, Zeppelin 서버 목록 중에서 본인 워크스페이스에서 연결해서 사용하고자 하는 노트북 서버를 선택 후 마침버튼을 클릭합니다.

- 아무 서버도 선택하지 않고자 한다면, **(없음)** 항목을 선택하십시오.

노트북 서버 설정

취소

마침

Jupyter
Zeppelin

연결된 서버 : jupyter

🔍 서버명을 검색해 주세요

서버	URL
<input type="radio"/> (없음)	
<input checked="" type="radio"/> jupyter ➡	http://metatron-web-04:8888
<input type="radio"/> jupyter-default	http://jupyter.mcloud.sktelecom.com:80
<input type="radio"/> QA_Test-test입니다 수정확인	http://www.

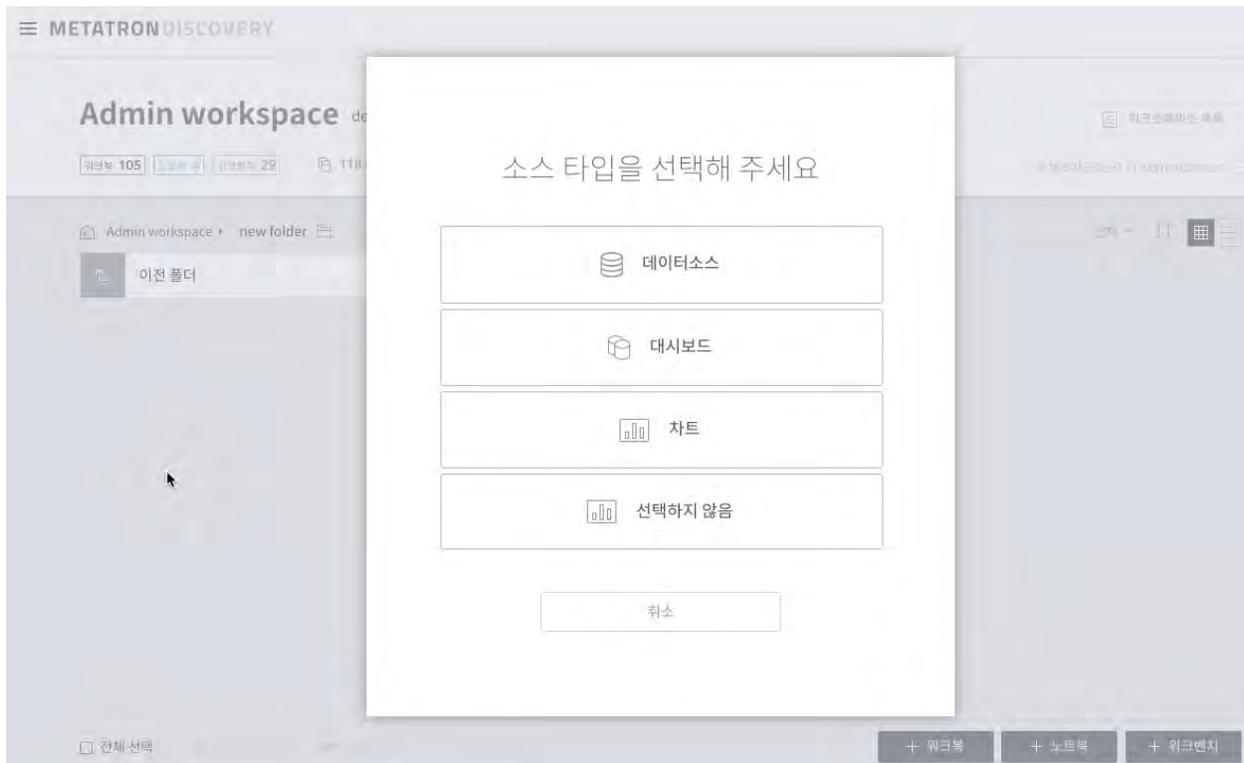
6.3 신규 노트북 생성하기

노트북 서버 설정이 완료되면, 노트북을 생성할 수 있습니다. 노트북 생성 절차는 다음과 같습니다.

1. 워크스페이스 하단에 있는 + 노트북 버튼을 클릭하면 노트북을 생성할 수 있는 화면이 나타납니다.



2. 노트북에서 분석하고자 하는 데이터셋 타입을 선택합니다. Metatron Discovery에서 사용하는 데이터 단위인 **데이터 소스**, **대시보드**, **차트** 그리고 **선택하지 않음** 중에서 선택할 수 있습니다. Zeppelin으로 분석하기를 원한다면, **선택하지 않음**을 선택하십시오.



3. **데이터 소스**, **대시보드**, **차트** 중 하나를 선택하면, 현재 Metatron Discovery에 등록된 데이터 목록을 조회할 수 있습니다. 분석 대상 데이터를 선택한 후 다음 버튼을 클릭합니다.


 노트북 생성하기


데이터소스를 선택해 주세요

sales ☐ 오픈 데이터원 보기 | 다됨 | 전체

No.	데이터소스	타입	사용처	수정일
6	market-sales - Stream Data 오픈 데이터	수집형	모든 노트북	2019-01-08
5	Sales 오픈 데이터	수집형	모든 노트북	2018-11-28
4	Sales_AA	수집형	사용하지 않음	2018-12-01
3	sales_csv_4r	수집형	사용하지 않음	2018-12-12
2	sales_geo - Salesdata (2011~2014) 오픈 데이터	수집형	모든 노트북	2019-01-16 ✓
1	snapshot_sales(ecoloy)	수집형	사용하지 않음	2018-12-04

sales_geo

메타데이터 이름

설명 Sales data (2011~2014)

타입 수집형

공개설정 공개

생성일 2018-10-29

사이즈 3.20 MB

Rows 9,994

- 자원값 OrderDate
- 자원값 ab Category
- 자원값 ab City
- 자원값 ab Country
- 자원값 ab CustomerName
- 측정값 ## Discount

취소
다음

4. 데이터를 분석하고자 하는 노트북 정보를 입력합니다. 초기 노트북 서버 설정에서 연결해 둔 노트북 서버에 한해서만 **서버 유형**을 선택할 수 있습니다. **Jupyter** 선택 시 <R> 또는 <Python> 언어를, **Zeppelin** 선택 시 <Spark> (scala) 언어를 선택하여 분석할 수 있게 됩니다.

다시 보기

노트북 생성하기

노트북 생성을 완료해 주세요

대미터 sales_geo

서버 타입 zeppelin

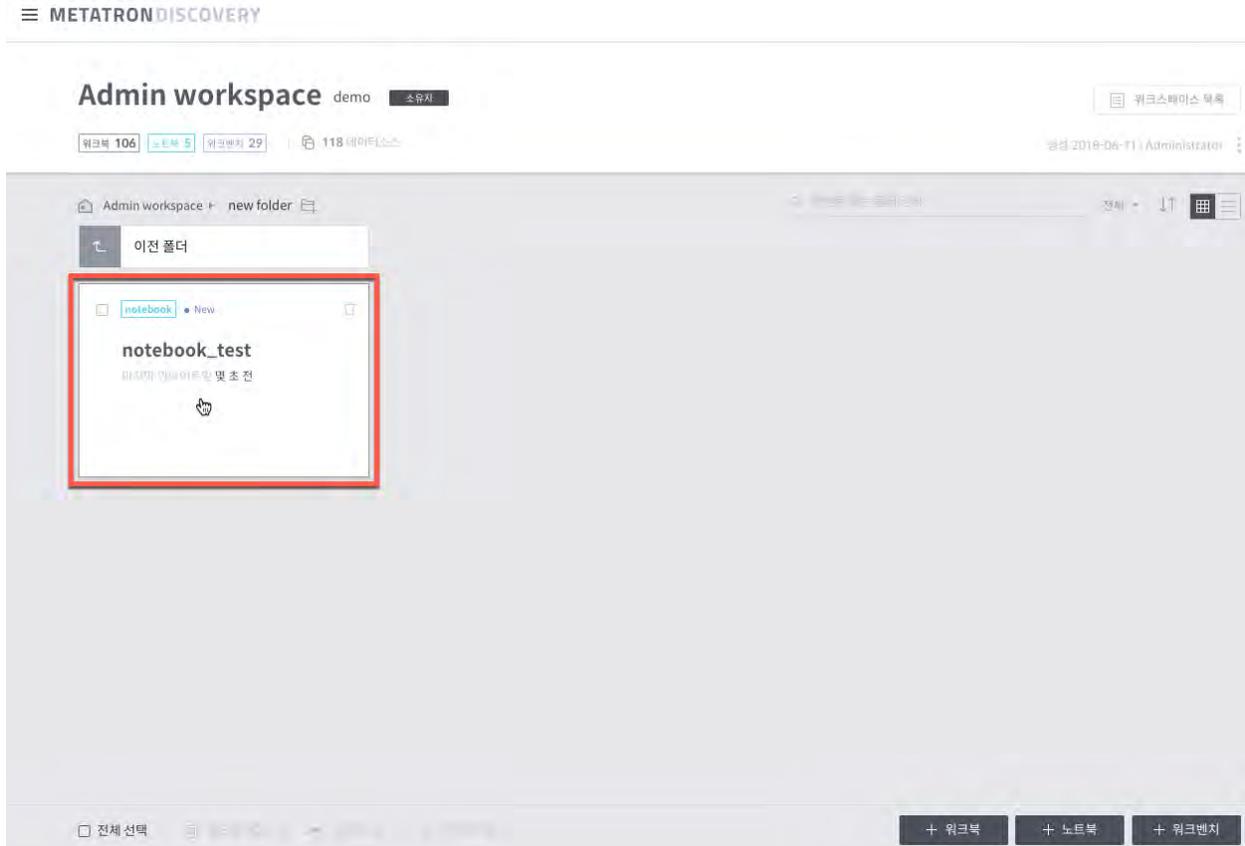
개발언어 SPARK

이름
notebook_test

설명
설명을 입력해 주세요

이전 마침

5. 노트북 생성이 완료되면, 해당 워크스페이스 화면에서 다음과 같이 확인할 수 있습니다.



6.4 노트북 활용하기

노트북을 생성하였으면, 스크립트를 작성하고 REST API를 통해 서비스할 수 있습니다. 노트북 활용 절차는 다음과 같습니다.

6.4.1 노트북 상세 조회

워크스페이스 화면에서 분석하고자 하는 노트북을 선택하면, 아래와 같은 상세 화면이 나타납니다. 노트북 생성 시 입력했던 데이터 타입과 데이터 소스 이름, 개발 언어, 코드를 조회할 수 있습니다.

The screenshot shows the METATRON DISCOVERY interface. At the top, there is a navigation bar with a hamburger menu icon and the text 'METATRON DISCOVERY'. Below this is a breadcrumb trail: '← 노트북 notebook_test' followed by the instruction '노트북 설명을 입력해 주세요.' Below the breadcrumb, there are four rows of configuration information:

- 소스 타입: DATASOURCE
- 데이터소스: sales_geo
- 개발 언어: SPARK
- 코드: 상세

Below this configuration section, there is a section for 'API' with the text 'API 정보가 없습니다.' and a button labeled 'API 생성'.

6.4.2 노트북 코딩

노트북 상세 조회 화면에서 코드란의 상세를 클릭하면, 노트북 페이지가 나타납니다. 노트북 페이지 상단에는 데이터셋을 로딩하는 코드가 삽입되어 있으며, 해당 셀을 실행하면 dataset 객체에 JSON 포맷의 데이터셋이 로딩됩니다.

The screenshot shows the Zeppelin Notebook interface. The top bar includes the Zeppelin logo, 'Notebook - Job', a search box 'Search your Notes', and a user indicator 'anonymous'. The notebook title is 'notebook_test'. The main area contains three code cells:

```
// 1. load dataset
import app.metatron.discovery.connector._;
val conf = new MetisClientSetting();
conf.setting("host", "metatron-web-01").setting("port", "8080");
val client = new MetisClient(conf);
val dataset = client.loadData(spark, "datasources", "ds-gis-37", "1000")
```

```
// 2. analyze
dataset.show()
```

Each code cell has a 'READY' status and control icons (run, refresh, stop) on the right side.

위 화면은 Zeppelin을 선택한 경우에 나타나며, 생성 시 선택한 데이터의 로딩을 위한 셀이 삽입되어 있습니다. 3번째 셀부터 프로그램 코딩 작업을 수행한 후 개발이 완료되면 **저장** 버튼을 클릭하십시오.

6.4.3 노트북 API 등록하기

작성한 노트북은 REST API 를 호출하여 결과값을 반환할 수 있습니다. 아래 설명을 참조하여 **리턴타입**을 선택한 후 **이름**과 **설명**을 기입하십시오.

- **HTML**: 노트북 스크립트 전체 실행 결과 화면을 HTML로 반환합니다.
- **JSON**: 노트북 스크립트에 작성된 사용자가 정의한 포맷의 JSON 객체를 결과로 반환합니다. 이 때 Metatron Discovery에서 제공하는 `response.write(...)` 함수를 사용하며, 예시 코드는 다음과 같습니다.
 - R 기반 노트북: `response.write(list(coefficient = 2, intercept = 0))`
 - Python 기반 노트북: `response.write({'coefficient' : 2.5, 'intercept' : 0})`
- **없음**: 노트북 스크립트를 실행하되 반환값은 제공하지 않습니다.

API 정보를 모두 입력한 후 **마침** 버튼을 클릭하면 API생성이 완료되고 아래와 같은 REST API URL을 확인할 수 있습니다. **Result** 버튼을 클릭하면 URL 실행 결과값을 팝업으로 조회할 수 있습니다.

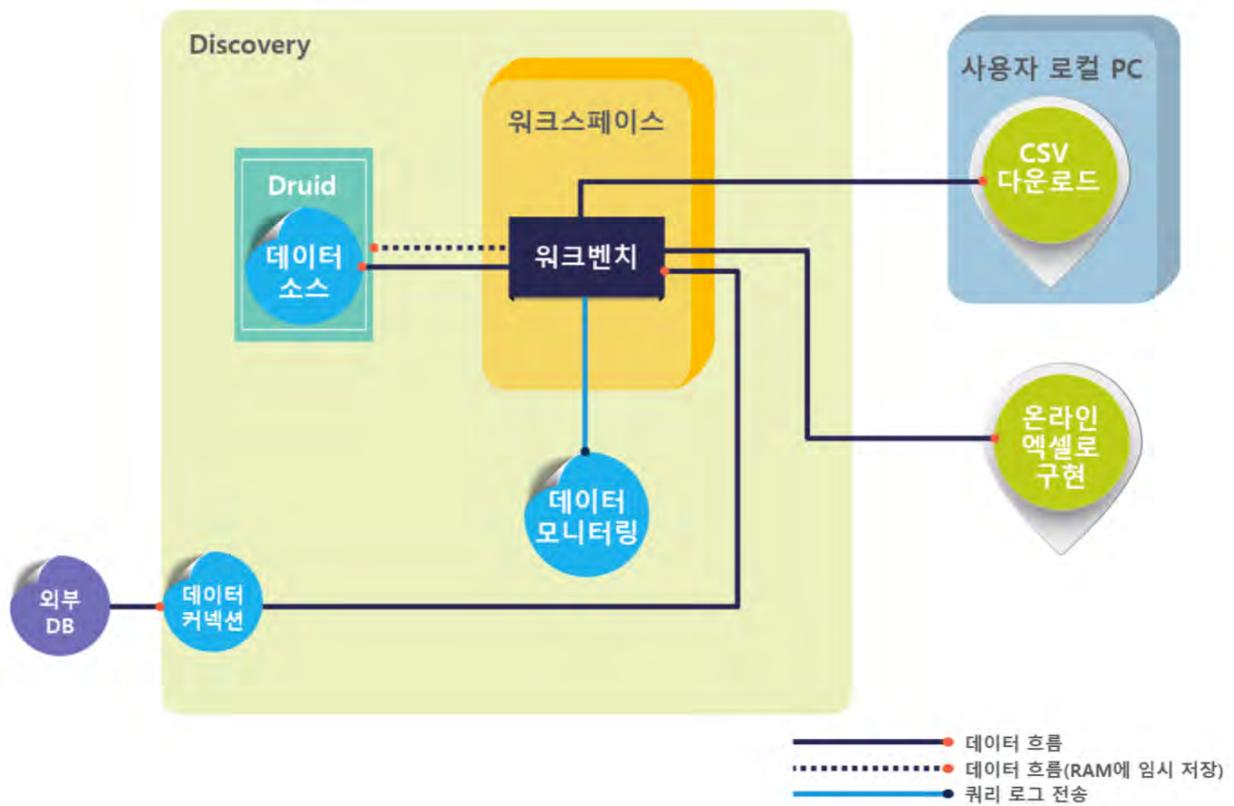
← 노트북 notebook_test 노트북 설명을 입력해 주세요

소스 타입	DATASOURCE
데이터소스	sales_geo
개발 언어	 SPARK
코드	상세 

이름	sample_api
설명	
URL	http://metatron-web-01:8080/api/notebooks/rest/dd44d0ba-2885-4ed6-bc09-14b9a9ab70ce
리턴타입	Void
API 결과	상태

[API 변경](#) [API 삭제](#)

워크벤치



Metatron Discovery 워크벤치는 SQL을 기반으로 하는 데이터 전처리 및 분석 환경을 제공합니다. 주요 기능은 다음과 같습니다.

- 다양한 외부 데이터베이스를 한꺼번에 작업공간에 조회 가능
- 연동된 데이터베이스의 테이블과 컬럼을 쉽게 조회/선택하며 상세 정보 열람 가능
- 쿼리 편집 도구가 내장되어 있으며 쿼리 결과를 실시간으로 확인하고 다양하게 활용 가능:
 - 쿼리결과를 로컬 파일로 다운로드 또는 온라인 엑셀로 출력
 - 쿼리 결과를 즉시 시각화함으로써 쿼리 결과로 출력된 데이터 형태를 쉽게 파악
 - 쿼리 결과를 데이터 소스로 저장하여 워크북이나 노트북에서 분석에 활용 가능

SQL 기반 분석 쿼리를 보관하는 각각의 문서를 <워크벤치>라고 부릅니다. 본 단원에서는 워크벤치를 생성하고 활용하는 절차를 소개합니다.

7.1 워크벤치 만들기

해당 워크스페이스에서 워크벤치를 사용하기 위해서는 워크벤치용 데이터 커넥션이 설정되어 있어야 합니다. 이에 관한 자세한 내용은 [데이터 커넥션](#) 항목을 참조하시기 바랍니다.

워크벤치 생성 절차는 다음과 같습니다.

1. 워크스페이스 하단에 있는 + 워크벤치 버튼을 클릭하면 워크벤치에서 데이터 분석에 사용할 데이터 커넥션을 연결할 수 있는 화면이 나타납니다.



2. 사용자가 연결하여 사용하고자하는 워크벤치용 데이터 커넥션을 선택한 후 다음 버튼을 클릭합니다.

워크벤치 생성하기

데이터 커넥션을 선택해 주세요

데이터 커넥션의 이름으로 검색해 주세요

DB 타입: 전체 | 계정 타입: All

No.	데이터 커넥션	타입	Host	Port	계정 타입	수정일
4	Hive-metatron-hadoop-01-10000	HIVE	metatron-hadoop-04	10000	항상 연결	2019-04-12
3	druid connection	DRUID	metatron-hadoop-02	8082	항상 연결	2019-04-08
2	MySQL-metatron-web-03-3306	MYSQL	metatron-web-03	3306	아이디와 비밀번호로 연결	2019-03-21
1	Presto-metatron-hadoop-01-8089	PRESTO	metatron-hadoop-01	8089	항상 연결	2019-03-19

더보기

취소 | 다음

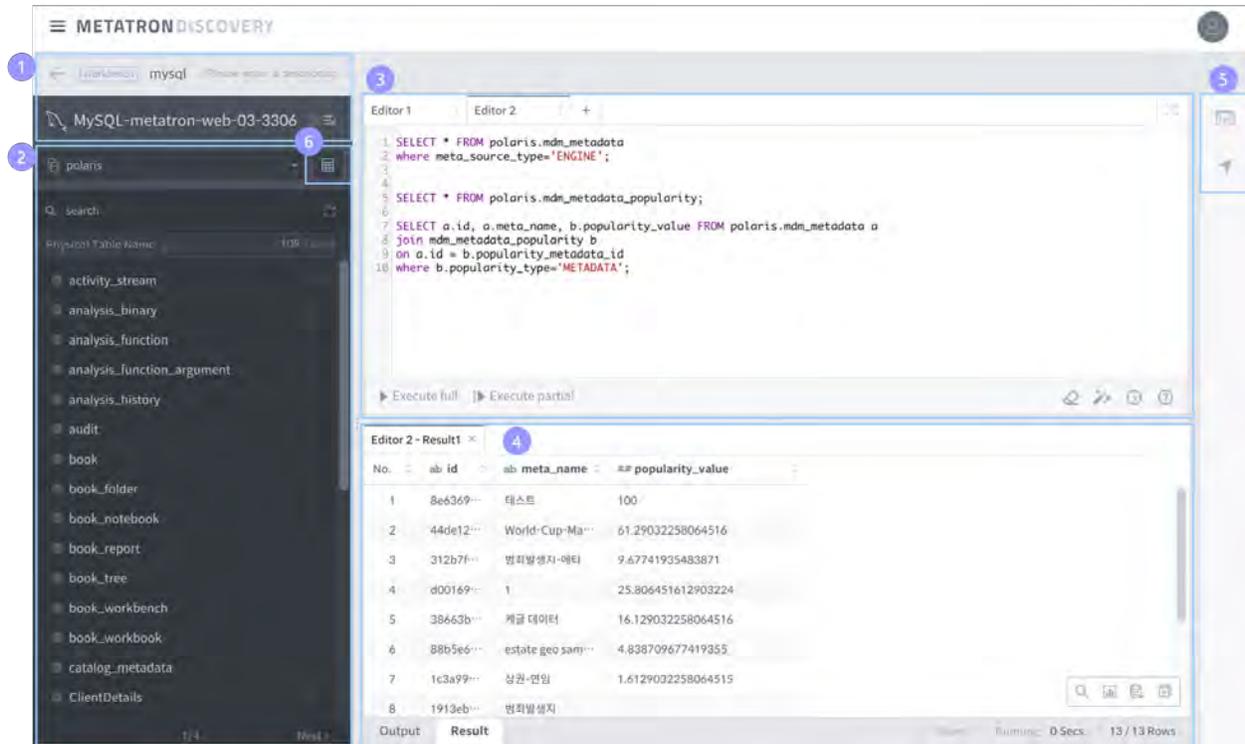
- **데이터 커넥션 이름으로 검색:** 해당 워크스페이스에 허용된 데이터 커넥션을 이름으로 검색합니다.
 - **DB Type:** 데이터 커넥션을 데이터베이스 타입 (Oracle/MySQL/Hive/presto/Tibero) 별로 선별하여 볼 수 있습니다. **모두**로 선택하면 모든 DB타입의 데이터 커넥션을 볼 수 있습니다.
 - **계정 타입:** 데이터 커넥션을 설정된 계정 타입 (관리자 직접 입력/사용자의 계정 사용/워크벤치 접속 시 직접 입력) 별로 선별하여 볼 수 있습니다. **All**로 선택하면 모든 계정 타입의 데이터 커넥션을 볼 수 있습니다.
 - **데이터 커넥션:** 설정한 선별 조건에 맞는 데이터 커넥션들을 보여줍니다.
3. 선택한 데이터 커넥션의 정보를 확인하고 이름과 설명을 입력하면 워크벤치가 생성됩니다.



4. 워크벤치 생성이 완료되면, 생성된 워크벤치를 바로 확인할 수 있습니다.

7.2 워크벤치 사용하기

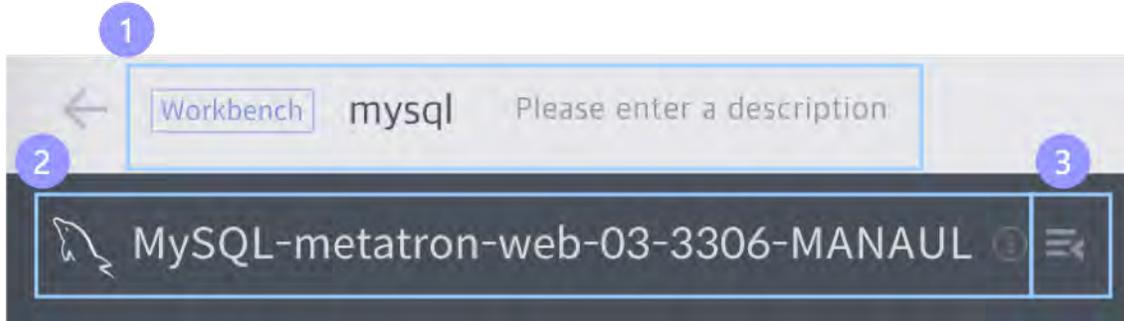
워크벤치에서는 SQL 데이터베이스 편집 및 관리가 용이하며 쿼리 결과를 다양한 형태로 시각화하여 저장할 수 있습니다. 워크벤치의 화면은 다음과 같은 5개의 영역으로 나뉘며, 추가적으로 스키마 브라우저를 열 수 있습니다.



1. 기본 정보 영역 (기본 정보 영역 참조)
2. 스키마 및 테이블 영역 (스키마 및 테이블 영역 참조)
3. 쿼리 에디터 영역 (쿼리 에디터 영역 참조)
4. 쿼리 결과 영역 (쿼리 결과 영역 참조)
5. 추가 도구 영역 (추가 도구 영역 참조)
6. 스키마 브라우저 (스키마 브라우저 참조)

7.2.1 기본 정보 영역

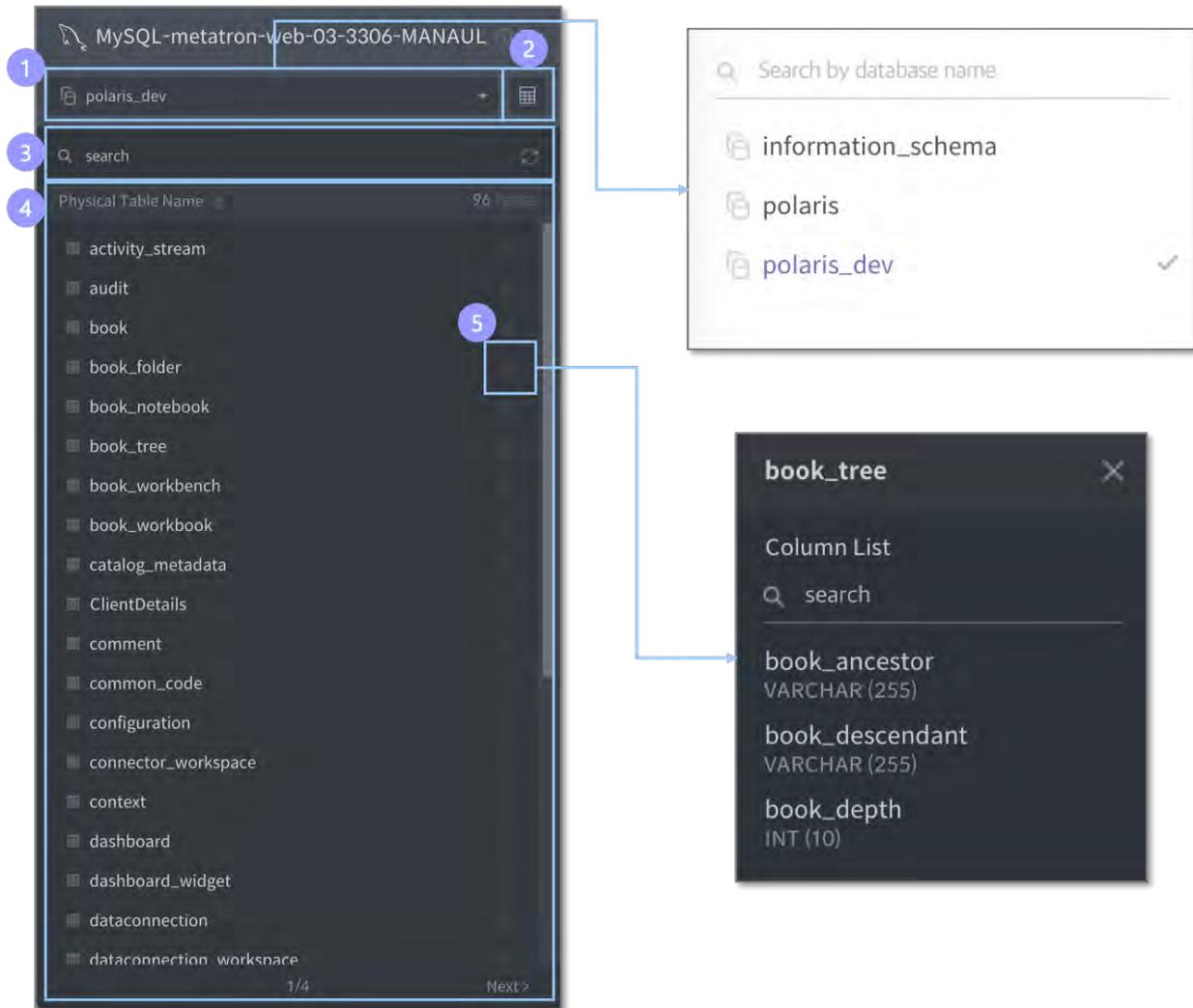
현재 작업하는 워크벤치에 관한 정보가 나타나는 영역입니다.



1. **이름**: 워크벤치의 이름입니다. 클릭하여 이름을 변경할 수 있습니다.
2. **데이터 커넥션**: 해당 워크벤치와 연결되어 있는 데이터 커넥션의 이름입니다.  아이콘을 클릭하면 자세한 정보가 나타납니다.
3.  : 패널을 접고 펼치는 UI 버튼입니다

7.2.2 스키마 및 테이블 영역

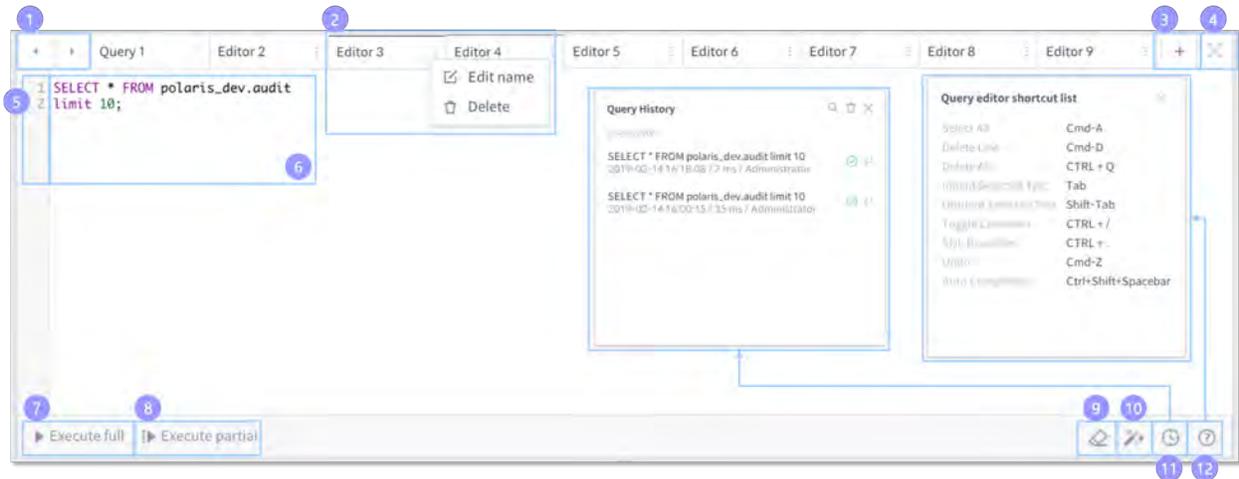
원하는 데이터베이스, 테이블, 컬럼을 손쉽게 쿼리 에디터에 입력할 수 있는 UI 기능입니다.



1. **데이터베이스 이름:** 현재 선택된 데이터베이스의 이름을 출력해줍니다. 해당 워크벤치에 등록된 데이터 커넥션의 첫 번째 데이터베이스를 기본적으로 선택됩니다. 클릭하면 데이터 커넥션에 포함된 모든 데이터베이스가 조회되며, 그 중에 하나를 선택하면 해당 데이터베이스로 변경됩니다.
2. **스키마 브라우저:** 선택된 데이터베이스의 테이블 목록과 각 테이블에 속한 모든 컬럼 및 레코드 정보를 확인할 수 있는 스키마 브라우저 화면이 팝업됩니다.
3. **테이블 검색:** 선택된 데이터베이스에 등록된 테이블을 이름으로 검색합니다.
4. **테이블 이름:** 필요한 데이터를 담은 테이블을 선택하면, 오른쪽 쿼리 에디터에 해당 테이블에 대한 `SELECT * FROM {table name}` 쿼리가 자동으로 입력됩니다.
5. **컬럼 목록:** 해당 테이블에 속한 모든 컬럼 이름과 각각의 데이터 타입이 나타납니다. 컬럼 이름을 클릭하면 쿼리 에디터에 자동으로 삽입됩니다.

7.2.3 쿼리 에디터 영역

쿼리를 작성하고 실행할 수 있는 에디터 화면입니다.

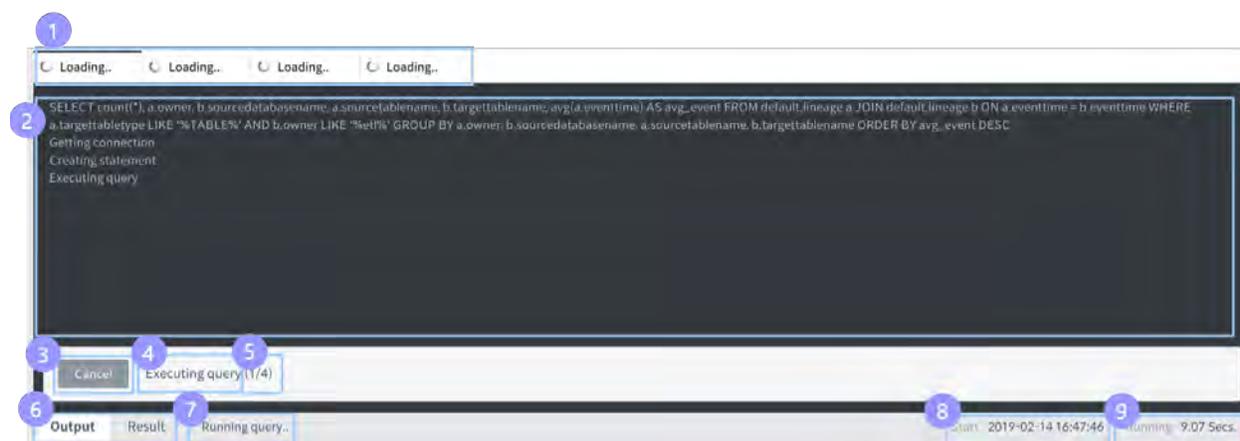


1.  : 탭이 너무 많을 경우 탭 영역을 좌우로 스크롤 할 수 있는 버튼입니다. 탭의 개수가 많지 않은 경우 나타나지 않습니다.
2. **탭**: 쿼리 관리를 위해 여러 개의 탭으로 나누어서 쿼리를 실행하거나 저장할 수 있습니다.  버튼을 클릭 시 탭 제목의 수정과 탭 삭제가 가능합니다.
3.  : 이 버튼을 클릭하면 새로운 탭이 추가됩니다.
4.  : 이 버튼을 클릭하면 쿼리 에디터 영역을 전체화면으로 확장하거나 축소 할 수 있습니다.
5. **쿼리 행**: 쿼리 코드의 행 번호를 보여줍니다.
6. **에디터 화면**: 이 곳에 쿼리 문을 작성합니다. 멀티 쿼리의 실행 및 단일 쿼리의 실행이 가능합니다.  을 쿼리 문이 끝날 때마다 삽입하면 쿼리를 나눠서 실행이 가능합니다. 자동완성 기능이 제공됩니다.
7. **쿼리 전체 실행**: 쿼리 에디터에 작성된 전체 쿼리를 실행합니다. (단축키: Ctrl + Enter)
8. **쿼리 부분 실행**: 쿼리 문 안에서 마우스가 위치한 특정 쿼리 또는 드래그하여 선택된 영역만 쿼리를 실행합니다. (단축키: Command + Enter)
9. **CLEAR SQL**: 쿼리 문을 모두 삭제 합니다.
10. **SQL BEAUTIFIER**: 이 버튼을 클릭하면 작성된 쿼리 문이 쿼리 문법 표준에 맞게 정렬됩니다.
11. **Query History**: 쿼리 에디터에서 수행한 과거 목록을 조회할수 있으며, 쿼리 선택 시 쿼리문이 쿼리 에디터에 추가됩니다.
12. **Query Editor 단축키**: 쿼리 에디터에서 사용가능한 단축키 목록입니다.

7.2.4 쿼리 결과 영역

쿼리가 실행되면 그 결과가 쿼리 결과 탭에 나타납니다. 모든 쿼리 결과는 계속 누적되지만, 원하는 결과 탭을 자유롭게 삭제할 수 있습니다. 쿼리 결과는 텍스트 그리드 형태로 제공되며, 차트 미리보기, 데이터 소스 저장, 다운로드 CSV 기능이 지원됩니다.

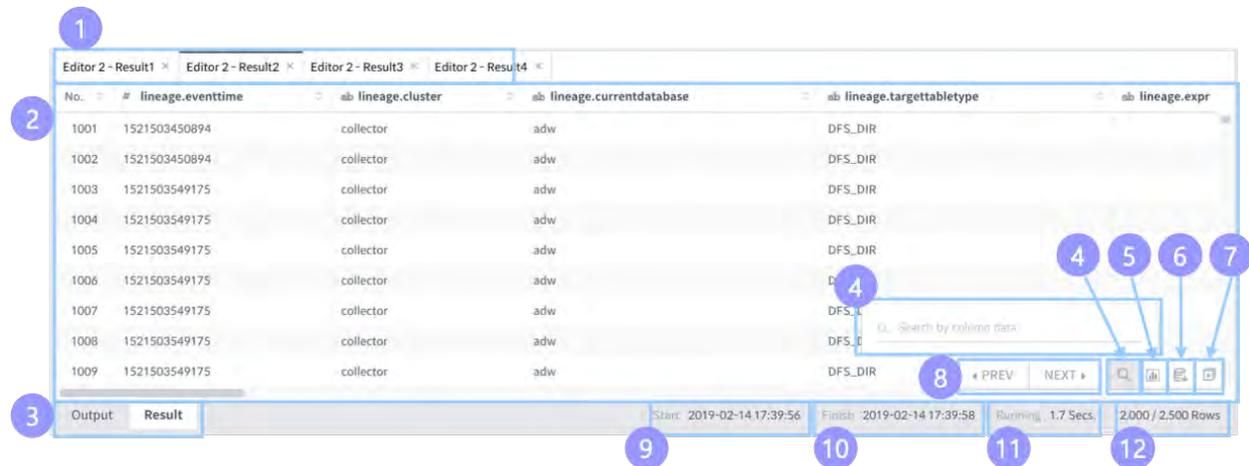
쿼리 수행 중



1. **쿼리 결과 탭:** 다중 쿼리 수행시 쿼리당 하나의 결과 탭이 생성되며 쿼리 수행 중인 탭일 경우 탭 제목에 <Loading> 메시지가 표시됩니다.
2. **쿼리 로그:** 쿼리 수행 로그를 보여주는 영역입니다. Hive 타입의 커넥션일 경우 Hive Job Log가 추가적으로 표시됩니다.
3. **쿼리 수행 취소:** 수행중인 쿼리를 취소합니다. DB 타입별 취소에 걸리는 시간이 다소 차이날 수 있습니다.
4. **쿼리 수행 단계:** 쿼리 수행의 총 5가지 단계 중 현재 단계를 표시합니다.
 - Getting connection
 - Creating statement
 - Executing query
 - Getting result set
 - Done!
5. **다중 쿼리 순서 표시:** 다중 쿼리 수행 시 현재 몇 번째 쿼리를 수행 중인지 표시합니다.
6. **표시 전환 탭:** 쿼리 수행 로그 탭과 쿼리 결과 탭을 전환하는 버튼입니다.

7. 쿼리 수행 상태: 3가지 쿼리 수행 상태를 표시합니다.
 - 쿼리 수행중
 - 쿼리 수행 실패
 - 쿼리 수행 취소
8. 쿼리 수행 시작 시각: 쿼리 수행 시작 시각을 표시합니다.
9. 쿼리 수행 경과 시간: 쿼리 수행 경과 시간을 표시합니다.

쿼리 수행 후



1. 쿼리 결과 탭: 다중 쿼리 수행시 쿼리당 하나의 결과 탭이 생성되며 쿼리 수행 중인 탭일 경우 탭 제목에 (Loading) 메시지가 표시됩니다.
2. 데이터 내역: 쿼리 실행에 의해 출력된 데이터 내역입니다. 출력된 데이터는 클립보드에 복사하여 활용할 수도 있습니다.
3. 표시 전환 탭: 쿼리 수행 로그 탭과 쿼리 결과 탭을 전환하는 버튼입니다.
4. 컬럼 데이터 검색: 결과 내 컬럼 및 값을 검색할 수 있습니다.
5. 차트 미리보기: 쿼리 결과를 이용하여 차트를 가상으로 그려볼 수 있습니다. 시각화를 위해 그려지는 것이고, 실제 워크스페이스 내용에 반영되지는 않습니다. (자세한 조작 방식은 차트 항목 참조)
6. 데이터 소스 저장: 쿼리 결과를 이용하여 워크스페이스 내 데이터 소스로 저장할 수 있습니다. 데이터 소스 생성 팝업이 나타나며, 데이터 커넥션 선택 및 테이블 선택 등의 과정은 워크벤치 결과 내용으로 대체됩니다. 따라서 스키마 정의 및 ingestion 주기 등의 과정이 곧바로 진행되게 됩니다. (자세한 절차는 데이터 소스 만들기 참조)

7. 다운로드 CSV: 쿼리 결과를 로컬 파일 (csv) 로 다운로드가 가능합니다.
8. 데이터 페이징: 1000건의 이상의 데이터일 경우 Prev, Next 버튼을 이용해 페이지 넘김이 가능합니다.
9. 쿼리 수행 시작 시각: 쿼리 수행 시작 시각을 표시합니다.
10. 쿼리 수행 종료 시각: 쿼리 수행 종료 시각을 표시합니다.
11. 쿼리 수행 경과 시간: 쿼리 수행 경과 시간을 표시합니다.
12. 쿼리 Row 정보: 쿼리 결과의 Row 숫자와 현재 페이지 정보를 표시합니다.

7.2.5 추가 도구 영역

추가 도구 영역은 워크벤치 활용에 도움을 주는 유용한 기능들로 구성되어 있습니다.

- 반복 사용 구문에 대한 글로벌 변수 설정 (글로벌 변수 (Global variable) 편집 기능 참조)
- 다른 워크벤치로 이동하기 위한 네비게이션 기능 (워크벤치 네비게이션 참조)

글로벌 변수 (Global variable) 편집 기능

어떤 구문이 반복적으로 사용되고 그 내용을 계속 바꿔가면서 쿼리를 실행해야 할 경우, 그 구문을 <글로벌 변수>로 지정해서 사용하면 편리합니다.

타입	이름	값
달력	변수 이름 입력	날짜 입력

부서코드	C60	
생성일	2019-02-01	

- 변수 타입: 글로벌 변수 타입으로는 달력과 텍스트가 제공됩니다.

- **새 변수 추가:** 원하는 변수 종류를 선택한 뒤 새 변수 추가 버튼을 누릅니다. 쿼리 에디터 영역에 해당 글로벌 변수가 추가됩니다.
- **이름:** 변수 이름을 입력합니다.
- **변수 값:** 달력은 날짜를 선택, 텍스트는 값을 입력하여 사용할 수 있습니다.

워크벤치 네비게이션

다른 워크벤치로 이동하는 기능을 제공합니다. 이동하기를 원하는 워크벤치를 클릭하면 해당 워크벤치로 이동합니다.



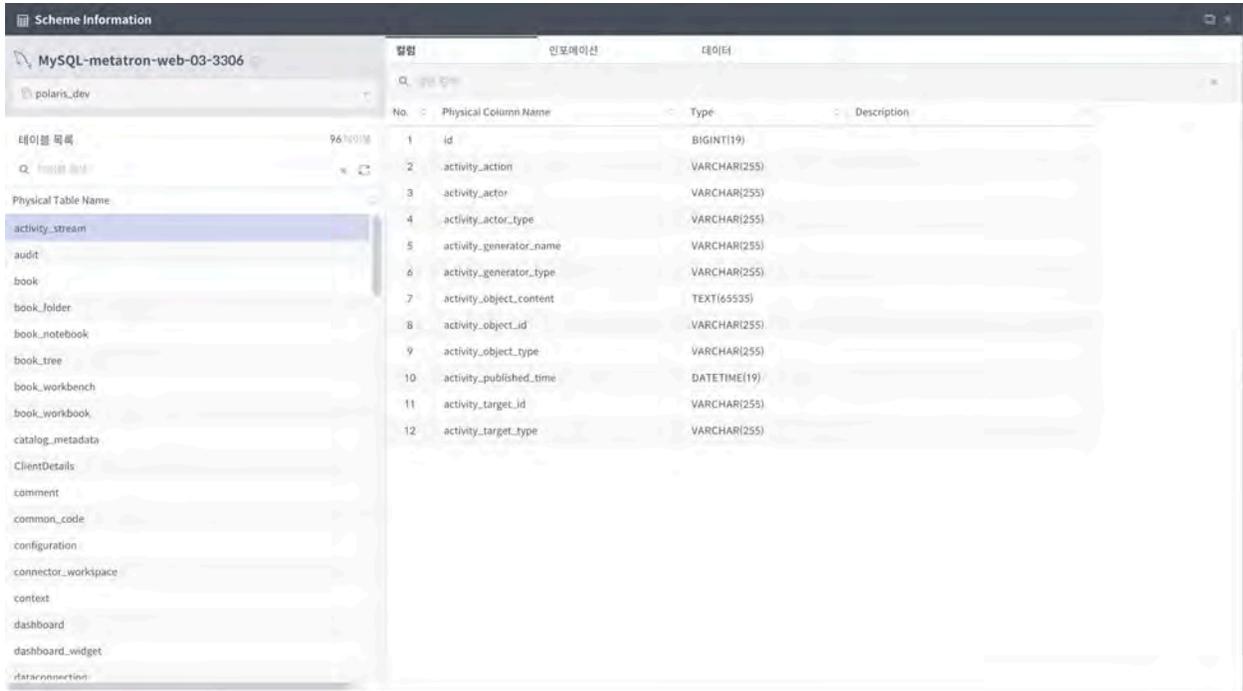
No.	워크벤치 이름	생성일
32	test sales	2019-02-07
31	asas	2019-01-30
30	ㅇㅇㅇㅇㅇ	2019-01-30
29	aaaaaa	2019-01-29
28	W	2019-01-24
27	Test	2019-01-22
26	s	2019-01-17
25	Test	2019-01-17
24	test	2019-01-16
23	test	2019-01-15
22	test_abc	2019-01-15
21	Test-Workbench-omelas	2019-01-14
20	Test2-mysql	2019-01-11
19	ㅇㅇ	2019-01-07
18	1234	2019-01-03
17	워크벤치	2018-12-26
16	sample workbench	2018-12-25
15	hive-trip-data	2018-12-21
14	man	2018-12-20
13	Test	2018-12-19

[+ 더보기](#)

- **워크벤치 검색:** 워크스페이스에 저장된 워크벤치를 검색합니다.
- **워크벤치 목록:** 워크스페이스에 저장된 모든 워크벤치를 보여줍니다. 나열된 워크벤치 중 하나를 클릭하면 해당 워크벤치로 이동됩니다.

7.2.6 스키마 브라우저

선택된 데이터베이스의 테이블 목록과 각 테이블에 속한 컬럼 및 레코드 정보를 확인할 수 있습니다.

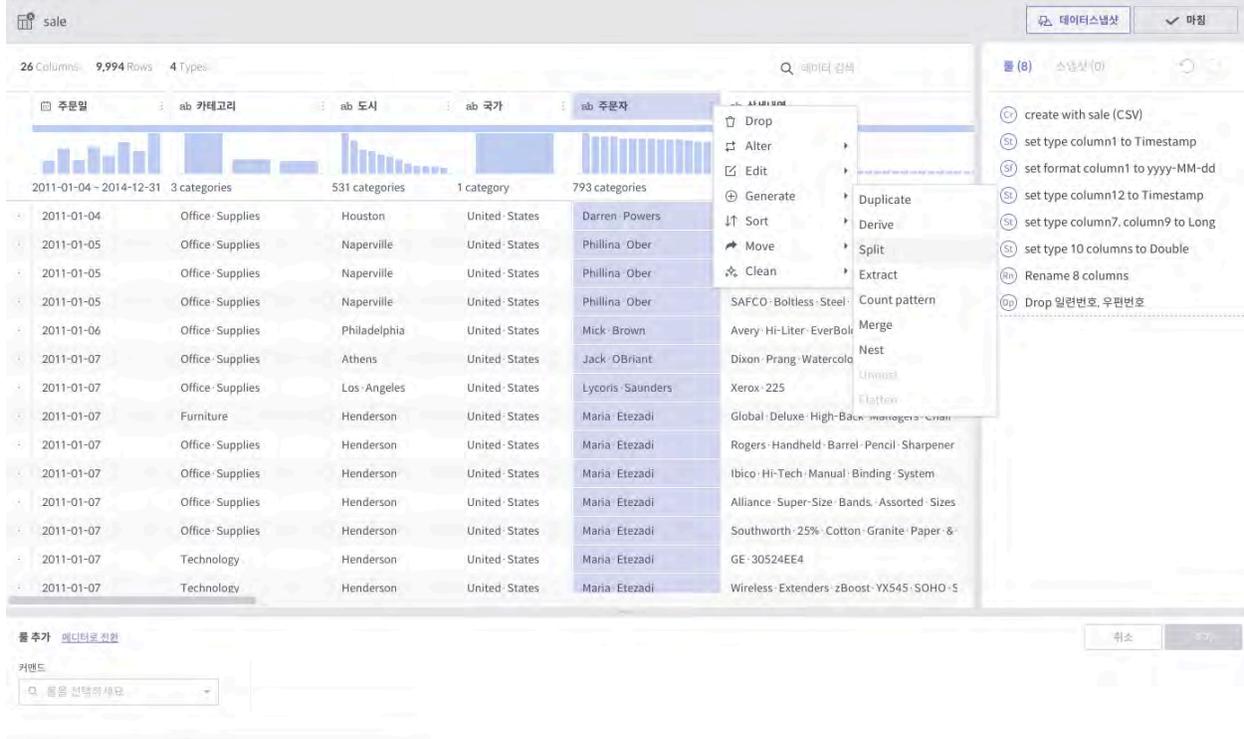


- **컬럼**: 선택한 테이블의 모든 컬럼의 이름과 속성을 보여줍니다.
- **인포메이션**: 선택한 테이블의 속성을 보여줍니다.
- **데이터**: 선택한 테이블의 데이터를 보여줍니다. 최대 50건의 데이터만 조회할 수 있습니다.

데이터 프리퍼레이션

데이터 프리퍼레이션은 파일, 테이블 등의 데이터셋을 분석에 용이한 형태로 정제하기 위한 변형 룰들을 생성하여, 그 결과를 HDFS, Hive 등으로 저장하는 툴입니다.

Metatron Discovery 데이터 프리퍼레이션의 장점

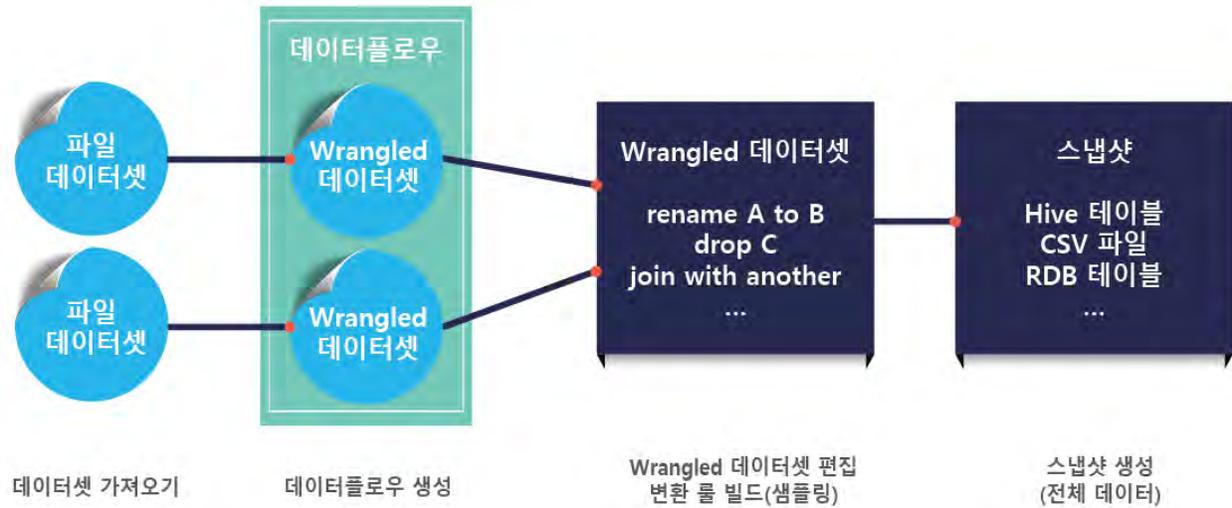


사용자는 위와 같이 GUI를 통해 Step-by-Step으로 변형 룰을 생성해낼 수 있습니다. 매 Step의 변형 결과가 데이터 분포도와 함께 메모리에 저장되기 때문에, 사용자는 이전 스텝 결과를 마우스 클릭만으로 쉽게 확인할 수 있을 뿐 아니라, 마치 텍스트 편집기를 쓰는 것처럼 **undo, redo** 등의 동작도 간단히 실행할 수 있습니다.

이러한 특징을 기반으로 데이터 프리퍼레이션 툴에서는 다음과 같은 장점을 활용할 수 있습니다.

- 프로그래밍이나 데이터 처리에 익숙하지 않아도 작업자가 원하는 형태의 결과를 얻을 수 있습니다.
- 보통 변형 룰 하나를 추가하려면 프로그래밍을 하거나 최소한 SQL문을 작성해야하지만, 데이터 프리퍼레이션 GUI를 통한 **탐색적 변형**을 활용하면 몇 번의 마우스 클릭이나 타이핑만으로 간편하게 변형 룰을 만들어내어 시간을 크게 절약할 수 있습니다.
- 기본적으로 수반되는 데이터 변형들은 자동으로 수행합니다. 예를 들어, 명백히 숫자로 보이는 컬럼에 대해 알아서 형변환 룰을 적용해줍니다. 이것은 언제나 **undo** 또는 룰 삭제가 용이하기 때문에 가능한 것입니다.
- 다양한 형태의 데이터를 결합하여 원하는 형태로 바꿔놓을 수 있습니다 (예: 기준 파일 + 팩트 테이블).
- 만들어 놓은 데이터 정제 결과를 다른 사람들과 공유함으로써, 물리적인 데이터를 주고 받는 부담을 줄여줍니다.
- 실제 데이터는 지우고 그것을 만드는 방법만 유지함으로써, 저장 공간을 아끼고 **ILM (Information Life Cycle)**을 줄일 수 있습니다. 다시 필요할 때 실제 데이터를 만들어내는 데에 부담이 줄어들기 때문입니다.

Metatron Discovery 데이터 프리퍼레이션의 구조



위 그림과 같이 데이터 프리퍼레이션은 정제할 대상 데이터를 참조하는 **데이터셋**, 지정된 데이터셋의 변형 룰들을 정의하는 **데이터플로우**, 그리고 그러한 룰들에 의해 변형된 결과물을 출력하는 **데이터 스냅샷**으로 구성됩니다.

8.1 상세 설치 가이드

Linux OS만 제공된 환경 (CentOS 7) 을 기준으로, 데이터 프리퍼레이션 기능을 모두 사용해볼 수 있도록 메타트론을 설치, 설정하는 것에 대한 가이드 문서입니다.

8.1.1 1. 필수 패키지 설치

루트로 다음 명령들을 실행합니다.

```
yum clean all && yum repolist && yum -y update
yum -y install tar unzip vi vim telnet apr apr-util apr-devel apr-util-devel net-tools curl openssl
↪elinks locate python-setuptools
yum -y install java-1.8.0-openjdk-devel.x86_64
export JAVA_HOME=/usr/lib/jvm/java
export PATH=$PATH:$JAVA_HOME/bin
```

8.1.2 2. Hadoop 설치

루트로 다음 명령들을 실행합니다. Hadoop 바이너리는 가까운 mirror를 통해서 다운로드 받는 것이 더 좋습니다.

```

yum -y install openssh-server openssh-clients rsync netstat wget
yum -y update libselinux

ssh-keygen -q -N "" -t dsa -f /etc/ssh/ssh_host_dsa_key
ssh-keygen -q -N "" -t rsa -f /etc/ssh/ssh_host_rsa_key
ssh-keygen -q -N "" -t rsa -f /root/.ssh/id_rsa
cp /root/.ssh/id_rsa.pub /root/.ssh/authorized_keys

wget http://archive.apache.org/dist/hadoop/common/hadoop-2.7.3/hadoop-2.7.3.tar.gz
tar -zxvf hadoop-2.7.3.tar.gz -C /opt
rm -f hadoop-2.7.3.tar.gz
ln -s /opt/hadoop-2.7.3 /opt/hadoop

export HADOOP_PREFIX=/opt/hadoop
export HADOOP_COMMON_HOME=$HADOOP_PREFIX
export HADOOP_HDFS_HOME=$HADOOP_PREFIX
export HADOOP_MAPRED_HOME=$HADOOP_PREFIX
export HADOOP_YARN_HOME=$HADOOP_PREFIX
export HADOOP_CONF_DIR=$HADOOP_PREFIX/etc/hadoop
export YARN_CONF_DIR=$HADOOP_PREFIX
export PATH=$PATH:$HADOOP_PREFIX/bin:$HADOOP_PREFIX/sbin

sed -i "/^export JAVA_HOME/ s:.*:export JAVA_HOME=$JAVA_HOME:" $HADOOP_CONF_DIR/hadoop-env.sh
sed -i "/^export HADOOP_CONF_DIR/ s:.*:export HADOOP_CONF_DIR=$HADOOP_CONF_DIR:" $HADOOP_CONF_DIR/
↪hadoop-env.sh

```

다음 파일들을 \$HADOOP_CONF_DIR에 넣어주세요.

```
core-site.xml hdfs-site.xml mapred-site.xml yarn-site.xml
```

계속해서 루트로 다음 명령들을 실행합니다.

```
$HADOOP_PREFIX/bin/hdfs namenode -format
```

다음 내용을 /root/.ssh/config에 다음 내용을 추가해주세요.

```

Host *
  UserKnownHostsFile /dev/null
  StrictHostKeyChecking no
  LogLevel quiet
  Port 2122

```

계속해서 루트로 다음 명령들을 실행합니다.

```
chmod 600 /root/.ssh/config
chown root:root /root/.ssh/config

chmod +x $HADOOP_CONF_DIR/*-env.sh

sed -i "/^[^#]*UsePAM/ s/.*#&/" /etc/ssh/sshd_config
echo "UsePAM no" >> /etc/ssh/sshd_config
echo "Port 2122" >> /etc/ssh/sshd_config
```

SSH 서버를 다시 시작합니다.

```
service sshd restart
```

HDFS 및 Yarn을 실행합니다.

```
start-dfs.sh
start-yarn.sh
```

Hadoop이 제대로 설치되었는지 테스트해봅니다.

```
hdfs dfs -mkdir -p /user/hadoop/input
hdfs dfs -put $HADOOP_PREFIX/LICENSE.txt /user/hadoop/input
hadoop jar $HADOOP_PREFIX/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar wordcount /user/
↳hadoop/input /user/hadoop/output
```

8.1.3 3. MySQL 설치

```
wget http://dev.mysql.com/get/mysql57-community-release-el7-7.noarch.rpm \
  && yum -y localinstall mysql57-community-release-el7-7.noarch.rpm \
  && yum repolist enabled | grep "mysql.*-community.*" \
  && yum -y install mysql-community-server mysql \
  && rm -f mysql57-community-release-el7-7.noarch.rpm
service mysqld start
```

다음의 명령을 통해 임시패스워드를 알아냅니다.

```
grep 'temporary password' /var/log/mysqld.log | awk {'print $11'}
Z&0+estx9vTt
```

위 패스워드를 이용해서 mysql_secure_installation을 실행합니다.

```
mysql_secure_installation
Enter password for user root: -> Z&0+estx9vTt
New password: -> Metatron123$
Re-enter new password: -> Metatron123$
Change the password for root ? ((Press y|Y for Yes, any other key for No) : y
New password: -> Metatron123$
Re-enter new password: -> Metatron123$
Do you wish to continue with the password provided? -> y
Remove anonymous users? -> enter
Disallow root login remotely? -> enter
Remove test database and access to it? -> enter
Reload privilege tables now? -> enter
```

MySQL에 접속해봅니다.

```
mysql -uroot -pMetatron123$
```

8.1.4 4. Hive 설치

```
wget http://mirror.navercorp.com/apache/hive/hive-2.3.6/apache-hive-2.3.6-bin.tar.gz \
  && tar -zxvf apache-hive-2.3.6-bin.tar.gz -C /opt \
  && rm -f apache-hive-2.3.6-bin.tar.gz \
  && ln -s /opt/apache-hive-2.3.6-bin /opt/hive
export HIVE_HOME=/opt/hive
export PATH=$PATH:$HIVE_HOME/bin:$HIVE_HOME/hcatalog/sbin
wget https://repo1.maven.org/maven2/mysql/mysql-connector-java/5.1.38/mysql-connector-java-5.1.38.jar
mv mysql-connector-java-5.1.38.jar $HIVE_HOME/lib/
```

다음 파일을 \$HIVE_HOME/conf에 넣어주세요.

```
hive-site.xml
```

Hive metastore를 초기화합니다.

```
mysql -uroot -pMetatron123$
create database hive_metastore;
create user 'hive'@'%' identified by 'Metatron123$';
grant all privileges on *.* to 'hive'@'%';
grant all privileges on hive_metastore.* to 'hive'@'%';
```

(다음 페이지에 계속)

(이전 페이지에서 계속)

```
create user 'hive'@'localhost' identified by 'Metatron123$';
grant all privileges on *.* to 'hive'@'localhost';
grant all privileges on hive_metastore.* to 'hive'@'localhost';
flush privileges;
quit
schematool -initSchema -dbType mysql
```

Hive를 시작합니다.

```
hdfs dfs -mkdir -p /user/hive/warehouse
mkdir -p $HIVE_HOME/hcatalog/var/log
hcat_server.sh start
hiveserver2 &
```

Hive에 접속해봅니다.

```
beeline -u jdbc:hive2://localhost:10000 "" ""
```

8.1.5 5. Druid 설치

```
wget https://sktmetatronkrsouthshared.blob.core.windows.net/metatron-public/discovery-dist/latest/druid-
↪0.9.1-latest-hadoop-2.7.3-bin.tar.gz
mkdir /servers
tar xzf druid-0.9.1-latest-hadoop-2.7.3-bin.tar.gz -C /servers
ln -s /servers/druid-* /servers/druid
export DRUID_HOME=/servers/druid
```

다음 파일들을 다운로드 받아서 지정된 위치로 넣어주세요.

Download URL	Target Location
jvm.config	\$DRUID_HOME/conf/druid/single/jvm.config
runtime.properties	\$DRUID_HOME/conf/druid/single/broker/runtime.properties
runtime.properties	\$DRUID_HOME/conf/druid/single/historical/runtime.properties
runtime.properties	\$DRUID_HOME/conf/druid/single/middleManager/runtime.properties

```
cd $DRUID_HOME
./start-single.sh
```

<http://localhost:8090/> 으로 접속이 된다면 성공한 것입니다.

8.1.6 6. Metatron 설치

```
wget https://sktmetatronkrsouthshared.blob.core.windows.net/metatron-public/discovery-dist/latest/  
↳metatron-discovery-latest-bin.tar.gz  
mkdir /servers  
tar xzf metatron-discovery-latest-bin.tar.gz -C /servers  
ln -s /servers/metatron-discovery-* /servers/metatron-discovery  
export METATRON_HOME=/servers/metatron-discovery
```

다음 파일들을 \$METATRON_HOME/conf에 넣어주세요.

```
application-config.yaml metatron-env.sh logback-console.xml
```

Metatron을 초기화합니다.

```
mysql -uroot -pMetatron123$  
create database polaris;  
create user 'polaris'@'%' identified by 'Metatron123$';  
grant all privileges on *.* to 'polaris'@'%';  
grant all privileges on hive_metastore.* to 'polaris'@'%';  
create user 'polaris'@'localhost' identified by 'Metatron123$';  
grant all privileges on *.* to 'polaris'@'localhost';  
grant all privileges on hive_metastore.* to 'polaris'@'localhost';  
flush privileges;  
quit  
cd $METATRON_HOME  
bin/metatron.sh --init start
```

진행상황을 보려면 log 파일을 tail 하세요.

```
tail -f logs/metatron-*.out
```

이제 <http://localhost:8180/> 으로 접속하면 됩니다.

8.1.7 7. Preptool 설치

```
yum -y install https://centos7.iuscommunity.org/ius-release.rpm \  
&& yum install -y python36u python36u-libs python36u-devel python36u-pip git \  
    (다음 페이지에 계속)
```

(이전 페이지에서 계속)

```

    && ln -s /bin/python3.6 /bin/python3 \
    && ln -s /bin/pip3.6 /bin/pip3 \
    && pip3 install requests
yum -y install git
git clone https://github.com/metatron-app/discovery-prep-tool.git
cd discovery-prep-tool

```

테스트용 파일을 다운로드 받습니다.

```
sales-data-sample.csv
```

```
python3 preptool -f sales-data-sample.csv
```

File dataset created라고 나오면 preptool이 제대로 동작하는 것입니다.

8.2 Docker Migration 가이드

Docker로 배포된 Metatron Discovery를 다른 docker image로 이관하는 작업에 대한 가이드 문서입니다.

<https://github.com/teamsprint/docker-metatron.git/> 을 사용한다고 가정합니다. 이에 대해선 <https://metatron.app/2020/01/21/deploying-metatron-with-the-fully-engineered-docker-image/> 를 참고하세요.

메타데이터 스토어로 MySQL을 사용한 경우에 대해서만 기술합니다.

8.2.1 1. 메타트론 서비스 중단

Docker 외부에서 다음 명령을 수행하면 docker instance 내부로 들어갑니다.

```

git clone https://github.com/teamsprint/docker-metatron.git/
cd docker-metatron
./attach.sh

```

Docker 내부에서 다음의 명령을 통해 메타트론 서비스를 중지시킵니다.

```

cd $METATRON_HOME
bin/metatron.sh stop

```

8.2.2 2. 메타데이터 스토어 백업

메타트론에서 사용하는 데이터셋, 데이터 플로우 등에 대한 정보를 먼저 백업해야 합니다. Docker 외부에서 다음 명령을 수행합니다. (Container의 이름이 metatron, 메타데이터 스토어로 사용한 데이터베이스의 이름이 polaris 경우)

```
sudo docker exec metatron /usr/bin/mysqldump -uroot -pMetatron123$ polaris > metadata_store_backup.sql
```

8.2.3 3. 설정 및 실행 스크립트 백업

```
sudo docker cp metatron:/servers/metatron-discovery/conf/application-config.yaml .
sudo docker cp metatron:/servers/metatron-discovery/conf/metatron-env.sh .
sudo docker cp metatron:/servers/metatron-discovery/conf/logback-console.sh .
sudo docker cp metatron:/servers/metatron-discovery/bin/metatron.sh .
sudo docker cp metatron:/servers/metatron-discovery/bin/common.sh .
```

8.2.4 4. 업로드된 파일 데이터셋, 데이터 스냅샷 백업

```
sudo docker cp metatron:/servers/metatron-discovery/dataprep/uploads .
sudo docker cp metatron:/servers/metatron-discovery/dataprep/snapshots .
```

일반적인 경우 데이터 스냅샷은 백업할 필요가 없습니다. 데이터 스냅샷이 작은 경우라면 금방 다시 만들어낼 수 있고, 다시 만들어내기 부담스러울 정도로 크다면 백업에 대한 부담도 크기 때문입니다.

Docker instance 내부의 데이터베이스에 저장한 데이터 스냅샷은 백업할 수 없습니다. Staging DB에 대한 설정을 따로 하지 않았다면 (docker instance의 초기 설정 그대로인 경우), staging DB 타입의 데이터 스냅샷도 백업할 수 없습니다.

8.2.5 5. 기존 docker instance 제거

다음의 명령을 수행하면 docker instance가 제거됩니다.

```
./destroy.sh
```

8.2.6 6. 새로운 docker instance 실행

Docker instance 외부에서 다음의 명령을 수행합니다.

```
./run.sh
```

새 바이너리로 패치하려면 run.sh을 편집하여 IMAGE_NAME을 수정해야 합니다.

Docker instance 내부에서 다음의 명령을 수행합니다.

```
./prepare-all-metatron.sh
```

보통 2~3분 내로 메타트론 서비스가 시작됩니다. 서비스 시작을 확인한 후 바로 서버를 내리고 복원을 시작합니다.

```
./stop-metatron.sh
```

8.2.7 7. 메타데이터 스토어 복원

Docker instance 외부에서 다음의 명령을 수행합니다.

```
cat metadata_store_backup.sql | sudo docker exec -i metatron /usr/bin/mysql -uroot -pMetatron123$  
↪polaris
```

8.2.8 8. 설정 및 실행 스크립트 복원

Docker instance 외부에서 다음의 명령을 수행합니다. 바이너리를 패치하는 경우, 설정 또는 실행 스크립트의 내용이 변한 경우 수정사항을 반영해주어야 합니다.

```
sudo docker cp application-config.yaml metatron:/servers/metatron-discovery/conf/  
sudo docker cp metatron-env.sh metatron:/servers/metatron-discovery/conf/  
sudo docker cp logback-console.sh metatron:/servers/metatron-discovery/conf/  
sudo docker cp metatron.sh metatron:/servers/metatron-discovery/bin/  
sudo docker cp common.sh metatron:/servers/metatron-discovery/bin/
```

8.2.9 9. 백업된 파일 데이터셋, 데이터 스냅샷 복원

Docker instance 외부에서 다음의 명령을 수행합니다.

```
sudo docker exec metatron mkdir -p /servers/metatron-discovery/dataprep  
sudo docker cp uploads metatron:/servers/metatron-discovery/dataprep/  
sudo docker cp snapshots metatron:/servers/metatron-discovery/dataprep/
```

8.2.10 10. 메타트론 서비스 시작

Docker instance 외부에서 다음의 명령을 수행합니다.

```
./attach.sh
```

Docker instance 내부에서 다음의 명령을 수행합니다.

```
./start-metatron.sh
```

보통 1~2분 내로 메타트론 서비스가 시작됩니다.

8.3 데이터셋 만들기

데이터셋은 데이터 프리퍼레이션의 가장 기본이 되는 단위로서, 데이터 연산의 대상이 되는 개체를 가리킵니다. **Imported Dataset**과 **Wrangled Dataset**의 두 가지 종류로 존재합니다.

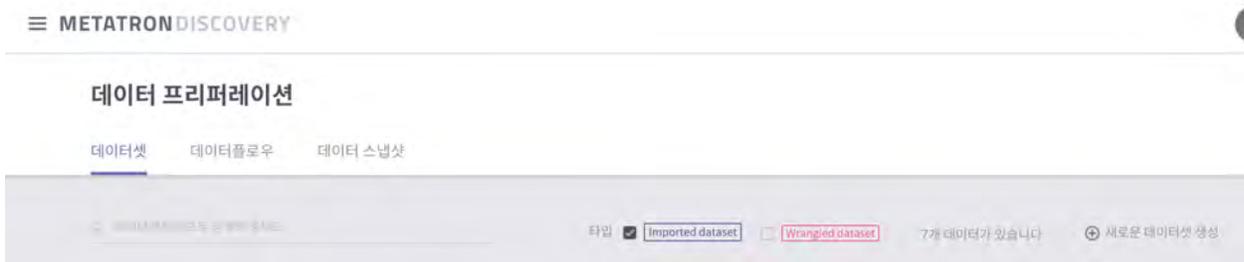
- **Imported Dataset:** 변형 규칙이 적용되기 전의 원천 데이터 개체
- **Wrangled Dataset:** 변형 규칙이 적용되어 분석 작업의 대상이 되는 데이터 개체

Wrangled Dataset은 변형 룰을 정의하는 **데이터플로우** 지정 과정에서 생성되는 것이며, 본 절차에서 생성되는 데이터 셋은 Imported Dataset입니다.

데이터셋 메뉴는 메인 화면 좌측 패널에서 **MANAGEMENT > 데이터 프리퍼레이션 > 데이터셋**을 통해 진입할 수 있습니다.



그런 다음 데이터셋 홈 화면 우측 상단에서 +새로운 데이터셋 생성 버튼을 클릭하면 새로운 데이터셋을 생성할 수 있습니다.



데이터셋 생성 화면에 들어가면 대상 원천 데이터의 타입을 선택해야 합니다.



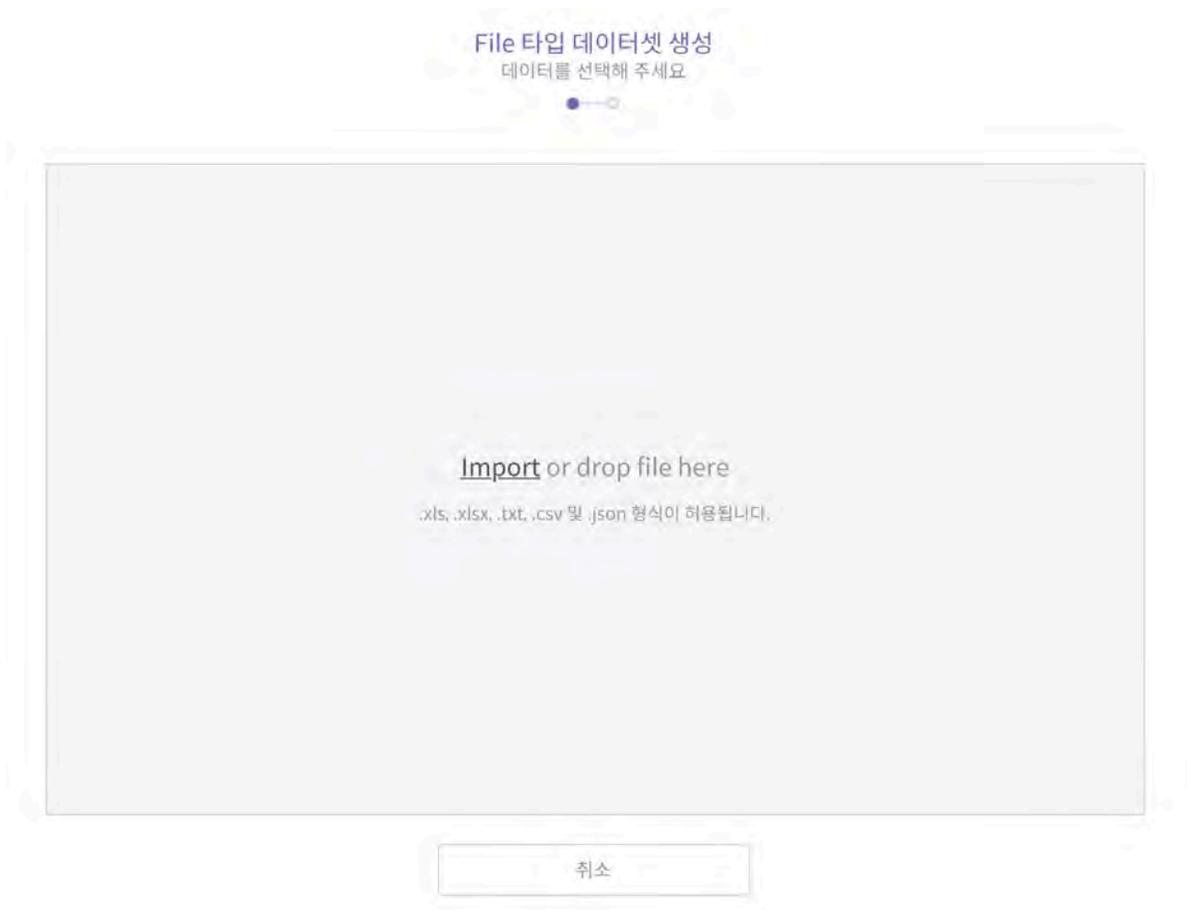
- **파일:** 사용자가 소유한 파일을 가져오거나 URI(곧 지원 예정) 를 이용하여 데이터셋을 생성합니다 (자세한 절차는 [파일 타입 데이터셋 생성하기 참조](#)).
- **Database:** 외부 데이터베이스의 접속 정보와 쿼리를 이용하여 데이터셋을 생성합니다 (자세한 절차는 [데이터베이스 타입 데이터셋 생성하기 참조](#)).
- **Staging DB:** Metatron 구동 시 설정된 Staging DB의 정보를 이용하여 데이터셋을 생성합니다 (자세한 절차는 [Staging DB 타입 데이터셋 생성하기 참조](#)).

참고: Staging DB는 ETL 프로세스에서 원활한 데이터 로딩을 위해 데이터를 임시 보관하는 클러스터 내 데이터베이스로서, 통상 Hive로 설정됩니다.

8.3.1 파일 타입 데이터셋 생성하기

사용자가 소유한 파일을 가져오거나 URI(곧 지원 예정) 를 이용하여 데이터셋을 생성합니다.

1. 데이터 타입 선택 화면에서 **파일**을 선택합니다.
2. 사용자 로컬 PC에서 데이터 소스로 사용할 파일을 가져옵니다. **Import** 버튼을 클릭하여 파일을 선택할 수도 있고 화면 상으로 파일을 끌어다 놓을 수도 있습니다. 파일을 가져왔으면 다음 버튼을 누릅니다.



3. 업로드 된 파일의 그리드 형태를 확인하고 컬럼 구분자를 지정합니다. 데이터가 바르게 출력되면 다음으로 넘어갑니다.



4. 생성할 데이터셋의 이름과 설명을 입력한 후 완료 버튼을 누릅니다.



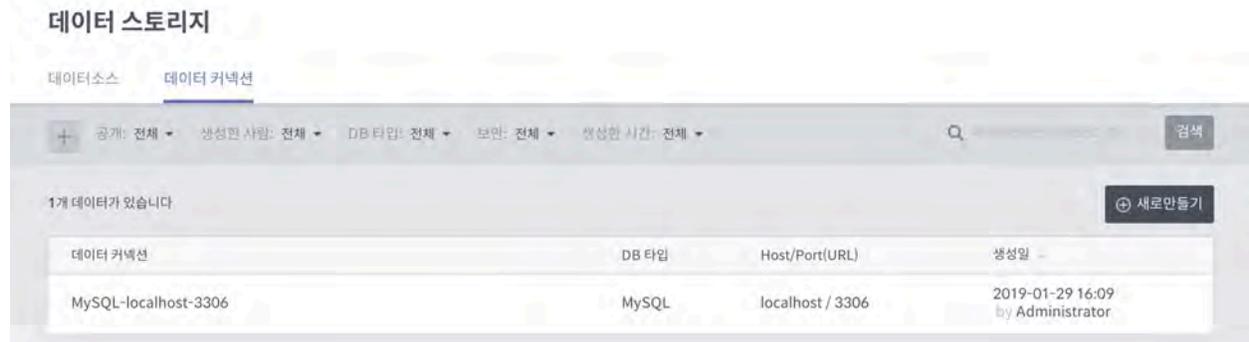
5. 데이터셋 생성이 완료되면 데이터셋 목록 화면으로 자동으로 이동합니다. 방금 생성한 데이터셋을 확인할 수 있습니다.



8.3.2 데이터베이스 타입 데이터셋 생성하기

외부 데이터베이스의 접속 정보와 쿼리를 이용하여 데이터셋을 생성합니다.

데이터베이스 타입의 데이터셋을 생성하기 위해서는 선행적으로 데이터 커넥션이 생성되어 있어야 합니다. 자세한 절차는 [데이터 커넥션 만들기 항목](#)을 참조하십시오.



해당 데이터 커넥션이 생성되었다면 다시 **MANAGEMENT > 데이터 프리퍼레이션 > 데이터셋 > +새로운 데이터셋 생성**을 통해 데이터 타입 선택 화면에 진입한 후, 다음의 절차를 진행합니다.

1. 데이터 타입 선택 화면에서 **Database**를 선택합니다.
2. 해당 데이터 커넥션을 선택하고 **테스트** 버튼을 눌러 유효한 커넥션임을 확인합니다.

DB 타입 데이터셋 생성
Data connection을 설정하십시오

DB 커넥션: MySQL-localhost-3...

MySQL-localhost-3306

MySQL ✓ Hive Presto Tiberio

Host	Port
localhost	3306

사용자 이름: polaris

패스워드:

보안

항상 연결

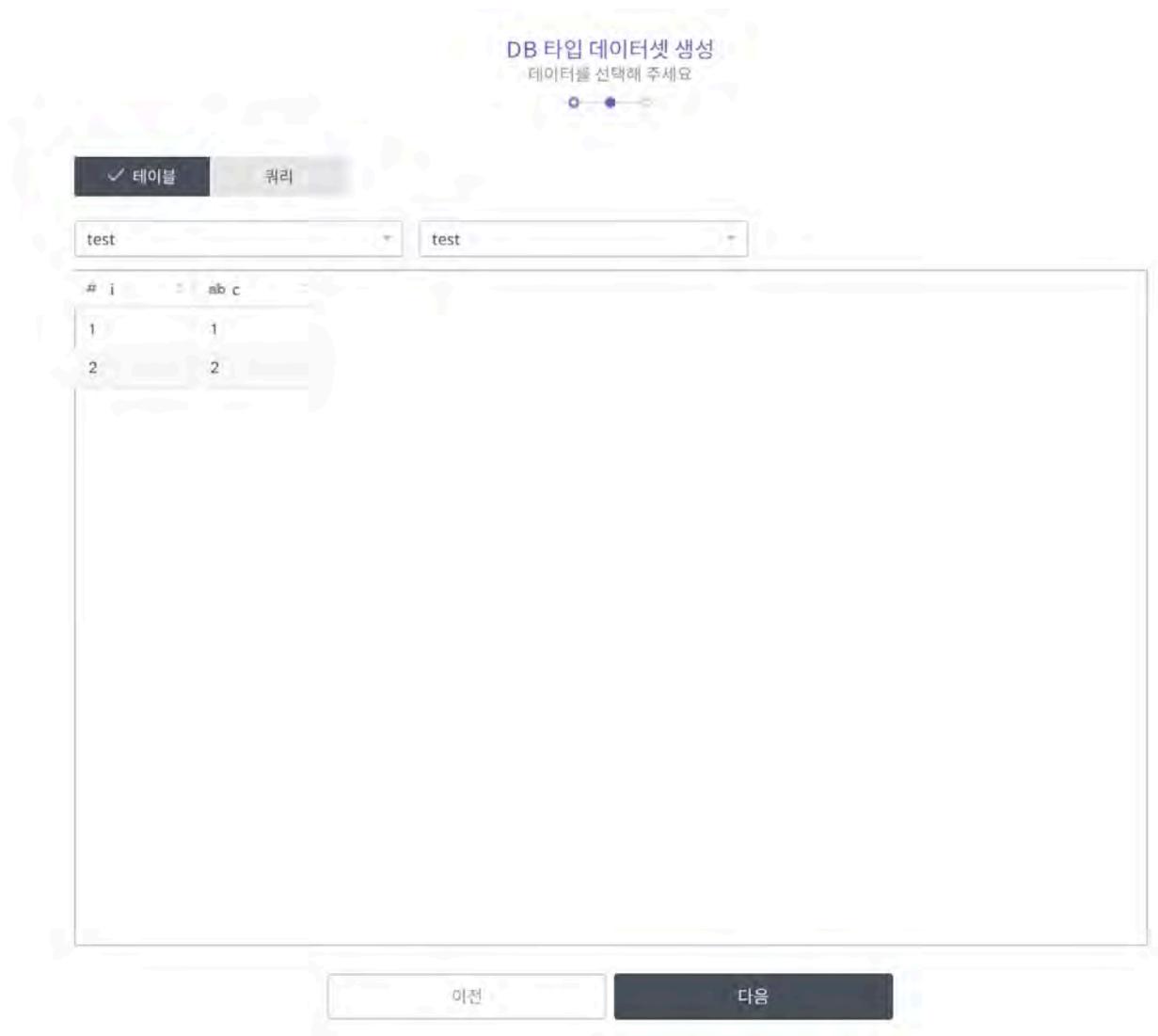
서버관리 권한으로 연결

이 데이터셋은 이항목으로 *Batch* 방식으로 적재 할 수 없습니다.

테스트 ✓ 유효한 커넥션

취소 다음

- 데이터를 선택합니다. 연결된 데이터베이스 계정에서 테이블을 선택할 수도 있고 쿼리문을 직접 작성할 수도 있습니다.



- **테이블**: 데이터베이스와 테이블명을 선택한 후 실제 저장될 데이터가 조회되면, 확인 후 **다음** 버튼을 누릅니다.
- **쿼리**: 원하는 데이터를 가져올 수 있는 쿼리문을 직접 작성하고 **실행** 버튼을 클릭하면 하단에 데이터가 보여집니다. 데이터를 확인한 후 **다음** 버튼을 누르십시오.

4. 생성할 데이터셋의 **이름**과 **설명**을 입력한 후 **완료** 버튼을 누릅니다.

DB 타입 데이터셋 생성
데이터셋 생성 완성하기

○ — ○ — ●

타입	MYSQL
데이터베이스	test
쿼리	select * from test
Host	localhost
Port	3306

이름
MySQL test dataset

설명
설명을 입력해 주세요

5. 데이터셋 생성이 완료되면 데이터셋 목록 화면으로 자동으로 이동합니다. 방금 생성한 데이터셋을 확인할 수 있습니다.

데이터 프리퍼레이션

데이터셋 데이터플로우 데이터 스냅샷

타입 Imported dataset Wrangled dataset 10개 데이터가 있습니다 + 새로운 데이터셋 생성

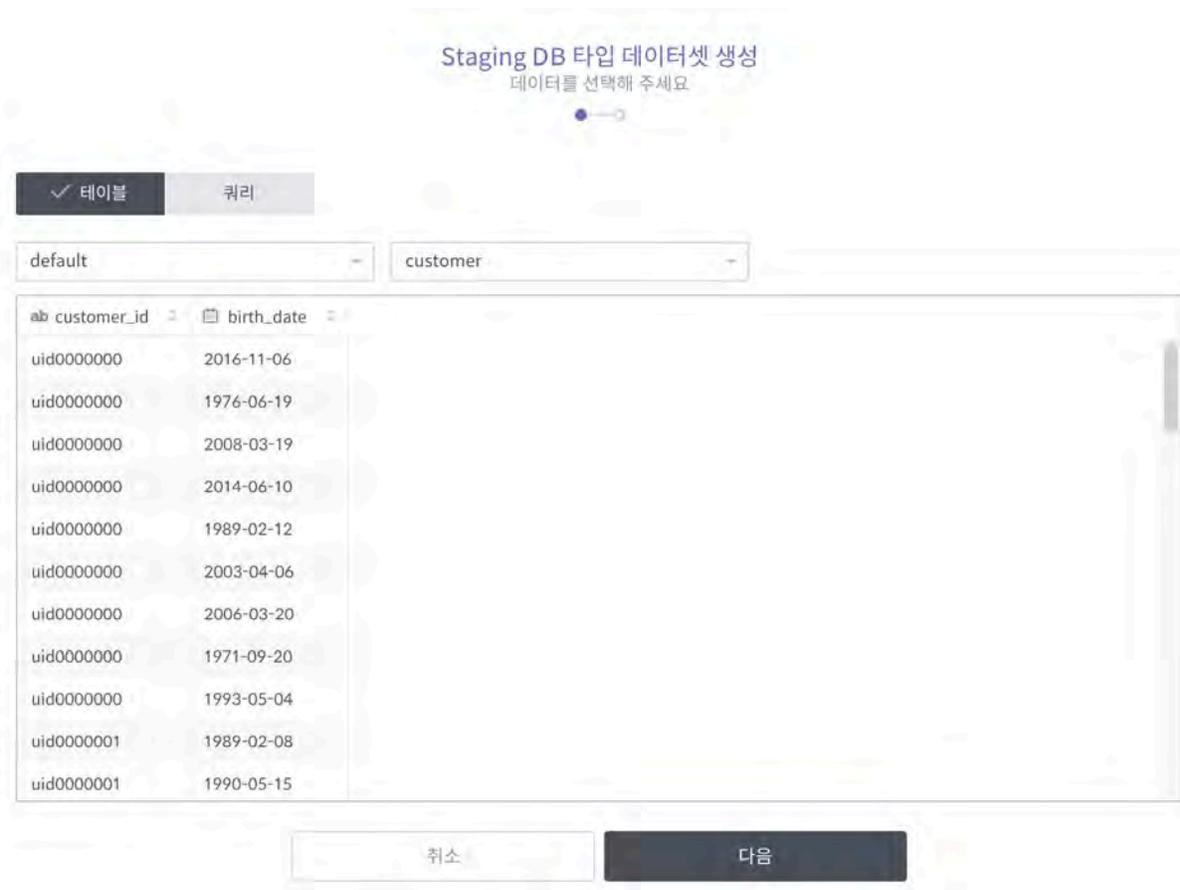
이름	사용처	소스	생성일
IMPORTED MySQL test dataset	0	Database	2019-02-01 14:58 by Administrator

8.3.3 Staging DB 타입 데이터셋 생성하기

Metatron 구동 시 설정된 Staging DB의 정보를 이용하여 데이터셋을 생성합니다.

Staging DB 타입의 데이터셋은 데이터 커넥션을 지정할 필요가 없다는 것을 제외하면 데이터베이스 타입의 데이터셋과 동일합니다.

1. 데이터 타입 선택 화면에서 **Staging DB**를 선택합니다.
2. 데이터를 선택합니다. 연결된 데이터베이스 계정에서 테이블을 선택할 수도 있고 쿼리문을 직접 작성할 수도 있습니다.



- **테이블**: 데이터베이스와 테이블명을 선택한 후 실제 저장될 데이터가 조회되면, 확인 후 **다음** 버튼을 누릅니다.
- **쿼리**: 원하는 데이터를 가져올 수 있는 쿼리문을 직접 작성하고 **실행** 버튼을 클릭하면 하단에 데이터가 보여집니다. 데이터를 확인한 후 **다음** 버튼을 누르십시오.

3. 생성할 데이터셋의 이름과 설명을 입력한 후 **완료** 버튼을 누릅니다.

Staging DB 타입 데이터셋 생성
데이터셋 생성 완성하기

타입	Staging DB
데이터베이스	default
테이블	customer

이름
customer (STAGING)

설명
설명을 입력해 주세요

이전 완료

4. 데이터셋 생성이 완료되면 데이터셋 목록 화면으로 자동으로 이동합니다. 방금 생성한 데이터셋을 확인할 수 있습니다.



8.4 데이터플로우 관리하기

데이터플로우는 **데이터셋**을 처리하는 단위입니다. 한 데이터플로우는 다수의 데이터셋들을 가져와 연관지어서 변형을 가할 수 있습니다. 다시 말해, 어떤 데이터셋이 변형 룰을 가지려면 반드시 한 데이터플로우에 속해야 하며, 그 안의 다른 데이터셋들과 join, union 등의 관계를 가질 수 있습니다.

아래와 같이 데이터플로우 상세 화면에서는 해당 데이터플로우에 속한 모든 데이터셋과 이들 간의 의존 관계, 그리고 각 데이터셋에 적용된 변형 룰들을 보여줍니다.

The screenshot displays the METATRON DISCOVERY web interface. At the top, the header shows 'METATRON DISCOVERY' and a user profile icon. Below the header, the breadcrumb path is 'test-20181114-1559 > 测试'. The main area shows a data flow diagram with nodes: 'sale (CSV)', 'sales (HIVE)', 'sale', 'feature1', and 'sales'. A sidebar on the right provides details for 'feature1'.

feature1 details:

- 설명: 설명을 입력해주세요.
- 데이터 프리뷰:

ab column2	ab column3
Office Supplies	Houston
Office Supplies	Naperville
Office Supplies	Naperville
- 타입: WRANGLED
- 규모: 9,994 row(s)
- 칼럼: 28 column(s)
- 사용처: 1 dataflow
- 플 리스트:
 - Cr create with sale (CSV)
 - St set type column1 to Timestamp
 - St set type column12 to Timestamp
 - St set type column7, column9 to Long
 - St set type 10 columns to Double
 - U union with sale

아래 각 하위 단위에서는 이러한 데이터플로우를 정의하기 위해 데이터셋을 추가하고, 변형 룰들을 편집하고, 변형 결과물을 데이터 스냅샷으로 출력하는 과정을 살펴봅니다.

데이터플로우 메뉴는 메인 화면 좌측 패널에서 **MANAGEMENT > 데이터 프리퍼레이션 > 데이터플로우**를 통해 진입할 수 있습니다.



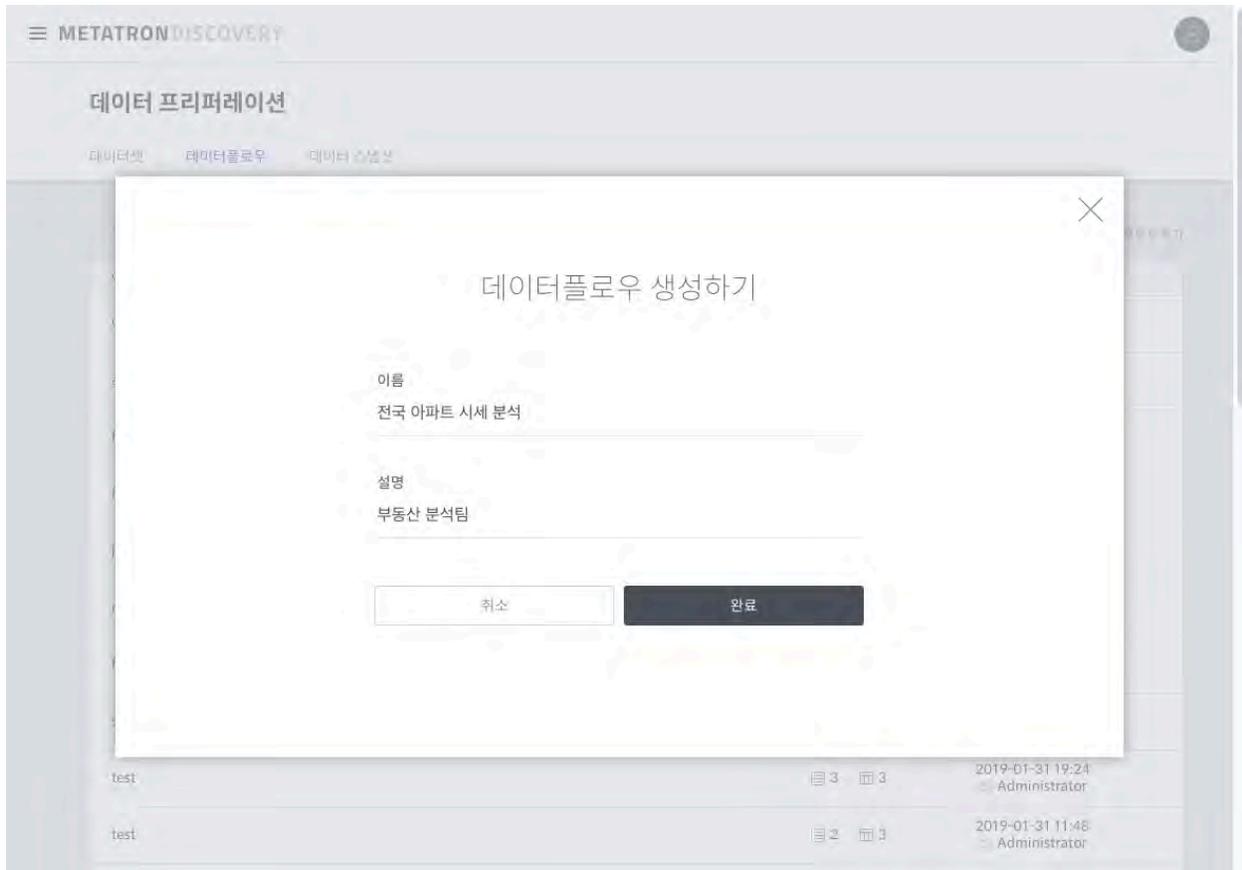
8.4.1 데이터셋 추가하기

데이터플로우 정의의 첫 단계는 데이터셋을 추가하는 것입니다. 이를 위해서는 아래와 같은 두 가지 방법이 있습니다.

- 빈 데이터플로우 생성 후 데이터셋 추가
- 데이터셋 상세 화면에서 바로 데이터플로우 생성

빈 데이터플로우 생성 후 데이터셋 추가

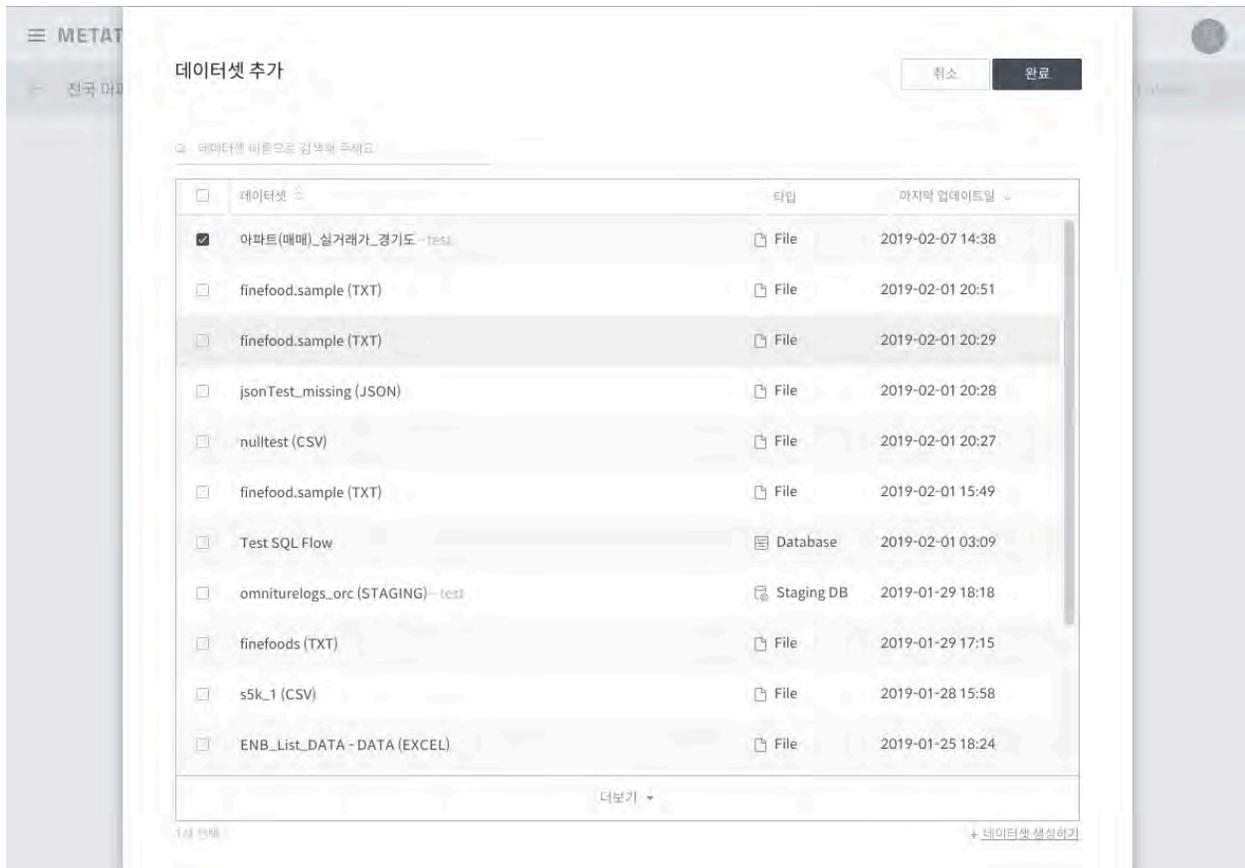
1. 데이터플로우 홈 화면 우측 상단에서 데이터플로우 추가를 클릭합니다.
2. 생성할 데이터플로우의 이름과 설명을 입력하고 완료를 누르면, 빈 데이터플로우가 생성됩니다.



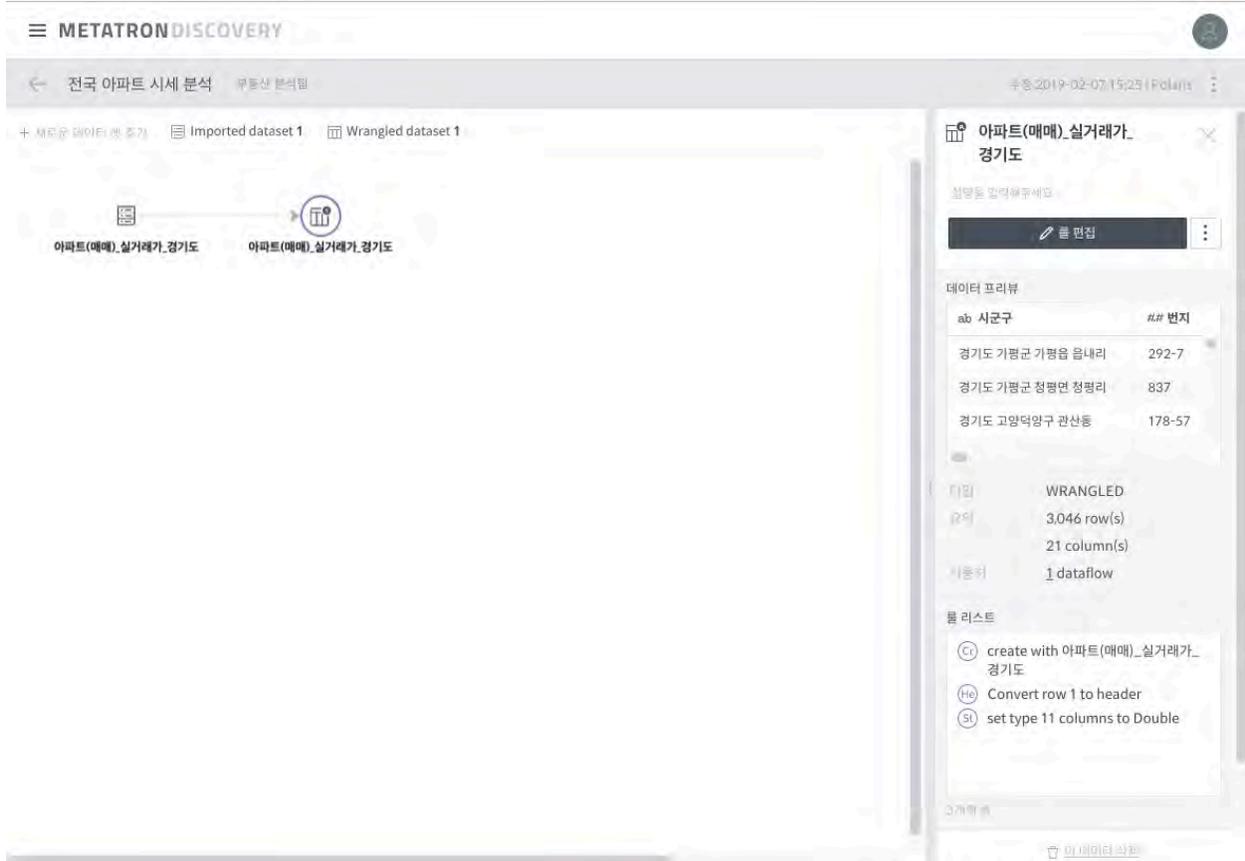
3. 화면 중앙에 위치한 이 데이터플로우에 데이터셋 추가 버튼을 누릅니다.



4. 추가할 데이터셋들을 선택합니다.



5. 선택한 Imported Dataset과 그에 상응하는 Wrangled Dataset이 생성되었으면, **룰 편집** 버튼을 눌러 룰 편집을 실시합니다 (자세한 절차는 [룰 편집 참조](#))



데이터셋 상세 화면에서 바로 데이터플로우 생성

데이터셋 상세 화면에서 이 데이터셋으로 새로운 데이터플로우 생성 버튼을 누르면, 자동으로 데이터플로우를 만들고 **롤 편집** 직전의 단계까지 진행합니다.

METATRON DISCOVERY

아파트(매매)_실거래가_경기도

수정 2019-02-07 14:38 | Polaris

정보

타입: FILE (EXCEL)

파일: 아파트(매매)_실거래가_경기도_위경도_20190207.xlsx (5).xlsx

시트: 아파트(매매)_실거래가_경기도_위경도_20190207.xl

시리: file:///data/metatron-discovery/dataprep/uploads/c7a33f...

사이즈: 777.9 KB

교차: 3,047 row(s)

21 column(s)

데이터

ab 시군구	## 번지	## 본번	## 부번	ab 단지명
경기도 가평군 가평읍 읍내리	292-7	292	7	메덴
경기도 가평군 청평면 청평리	837	837	0	청평삼성세르빌
경기도 고양덕양구 관산동	178-57	178	57	새서울
경기도 고양덕양구 관산동	178-57	178	57	새서울
경기도 고양덕양구 관산동	178-57	178	57	새서울
경기도 고양덕양구 관산동	178-57	178	57	새서울
경기도 고양덕양구 도내동	983	983	0	엘에이치원풍도래울마을2단지
경기도 고양덕양구 도내동	983	983	0	엘에이치원풍도래울마을2단지
경기도 고양덕양구 도내동	983	983	0	엘에이치원풍도래울마을2단지

사용처

+ 기존 데이터플로우에 추가 이 데이터셋으로 새로운 데이터플로우 생성

아파트(매매)_실거래가_경기도_0207_1438 수정 2019-02-07 14:38 | polaris

전국 아파트 시세 분석 - 부동산 분석팀 수정 2019-02-07 15:25 | polaris

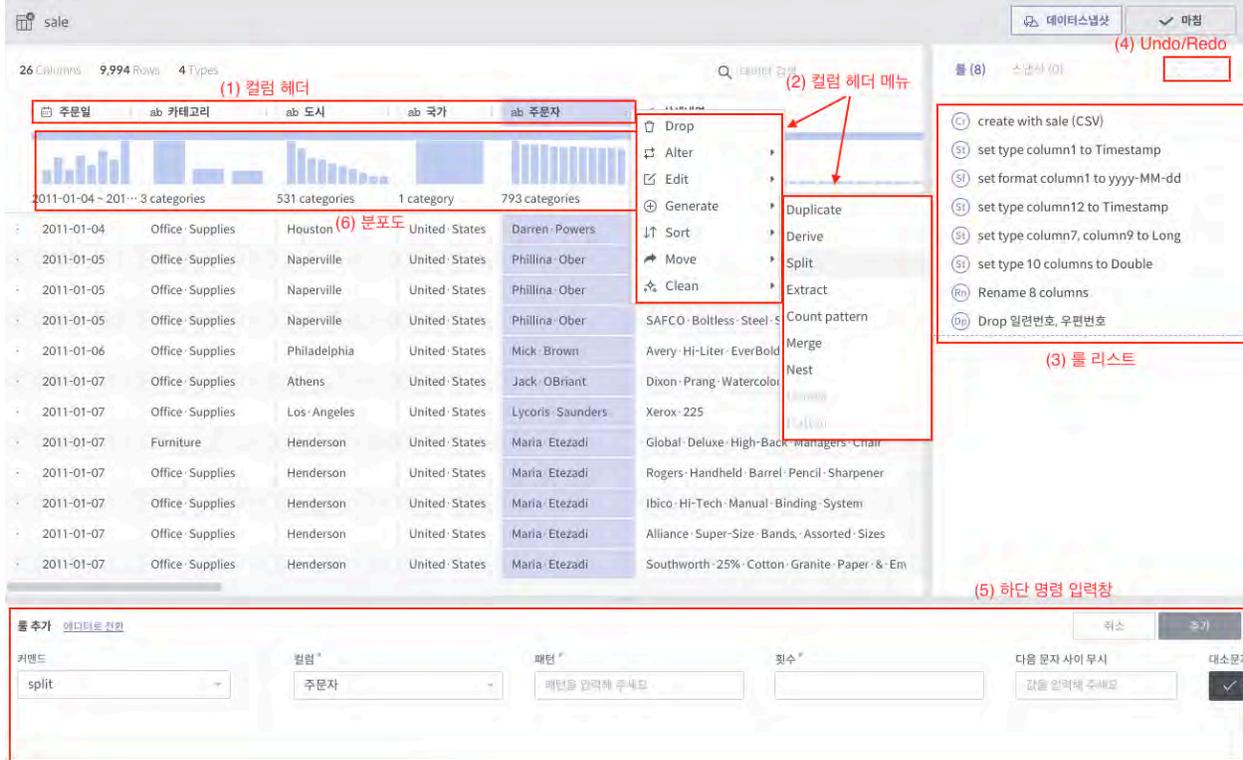
참고: 데이터플로우의 이름은 데이터셋 이름을 기반으로 자동으로 지어집니다.

8.4.2 룰 편집

데이터 프리퍼레이션에서 가장 핵심적인 작업은 데이터를 변형 (주로 정제) 하는 룰 (rule) 을 만들어내는 것입니다. 이 변형 룰과 입출력 명세를 합쳐서 우리는 실제 데이터에 적용하거나, 또 비슷한 다른 데이터에 적용하거나, 이런 작업들을 스케줄링합니다.

이제 룰을 만들고, 결과를 확인하고, 룰을 다시 변경하거나 삭제하는 일에 대한 설명을 하겠습니다.

먼저 룰 편집 화면의 구성은 다음과 같습니다.



1. 컬럼 타입, 이름, 메뉴 버튼
2. 룰 간편 생성을 위한 메뉴
3. 적용된 룰 리스트 및 중간 삽입 버튼 (룰 사이에 마우스를 갖다대면 나옴)
4. Undo, redo가 가능한 경우 활성화
5. 룰 세부 내용 입력 패널
6. 컬럼 값 분포 및 distinct count, type mismatch 또는 null value 관련 정보 등

룰 생성하기

컬럼 헤더 메뉴를 이용하는 방법

1. 컬럼 헤더를 클릭해서 대상 컬럼을 고릅니다.
 - 기능 키를 이용해서 다수의 컬럼을 고를 수 있습니다.
 - OS에 따라 ^ 또는 키를 누른 채 클릭을 하면 해당 컬럼이 선택/해제됩니다. (토글)
 - Shift 키를 누른 채 클릭을 하면 범위가 선택됩니다.

samsung_ship

937 Columns 5,052 Rows 3 Types Q 데이터 검색

ab column1	ab column2	## column3	## column4	# column5	# column6	# colum
85 categories	85 categories	33.81 ~ 34.10	128.84 ~ 128.96	162959 ~ 175...	2019 ~ 2019	1 ~ 1
· 2019-01-26 16:30	2019-01-26 16:30	34.10035	128.95722	162959	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10035	128.95722	162959	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10032	128.9572	163000	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10025	128.95717	163002	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10023	128.95715	163003	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10023	128.95715	163003	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.1002	128.95713	163004	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10017	128.95712	163005	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10012	128.95707	163007	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10012	128.95707	163007	2019	1
· 2019-01-26 16:30	2019-01-26 16:30	34.10008	128.95705	163008	2019	1

2. 선택된 컬럼 중 하나의 헤더에서 아이콘을 클릭해서 헤더 메뉴를 연 후, 변환 명령을 선택합니다.

- 이 중에서 **drop**, **settype** 등은 즉시 실행됩니다.

937 Columns 5,052 Rows 3 Types

데이터 검색

ab column1 ab column2 ## column3 ## column4 # column5 # column6 # column

85 categories 85 categories 33.81 ~ 34.10

2019-01-26 16:30 2019-01-26 16:30 34,10035

2019-01-26 16:30 2019-01-26 16:30 34,10035

2019-01-26 16:30 2019-01-26 16:30 34,10032

2019-01-26 16:30 2019-01-26 16:30 34,10025 128.95717 163002 2019 1

2019-01-26 16:30 2019-01-26 16:30 34,10023 128.95715 163003 2019 1

2019-01-26 16:30 2019-01-26 16:30 34,10023 128.95715 163003 2019 1

2019-01-26 16:30 2019-01-26 16:30 34,1002 128.95713 163004 2019 1

2019-01-26 16:30 2019-01-26 16:30 34,10017 128.95712 163005 2019 1

2019-01-26 16:30 2019-01-26 16:30 34,10012 128.95707 163007 2019 1

2019-01-26 16:30 2019-01-26 16:30 34,10012 128.95707 163007 2019 1

2019-01-26 16:30 2019-01-26 16:30 34,10008 128.95705 163008 2019 1

3. 추가 입력이 필요한 경우, 하단 명령 입력 패널을 통해 내용을 입력한 후 추가 버튼을 누릅니다.

콜 추가 [에디트로 전환](#)

취소 **추가**

커맨드 필름* 새로운 필름 이름* 전체 필름 변경

4. 몇몇 룰의 경우에는 분포도 막대를 선택해서 명령을 수행할 수도 있습니다.

- 분포도의 막대를 클릭하면 해당 범위를 조건으로 필터링 등을 실시할 수 있습니다 (토글).
- Type mismatch, null value 그래프를 클릭해서 해당 값들에 대해 조건을 걸 수도 있습니다.

ab column1 ab column2 ab column3 ab column4 ab column5 ab column6 ab column7

85 categories 85 categories 4265 categories 4254 categories 4177 categories 1 category 1 category

2019-01-26 16:39 2019-01-26 16:39 34.08185 128.94647 163946 2019 1

2019-01-26 16:39 2019-01-26 16:39 34.08182 128.94645 163948 2019 1

2019-01-26 16:39 2019-01-26 16:39 34.08178 128.94643 163949 2019 1

2019-01-26 16:39 2019-01-26 16:39 34.08178 128.94643 163950 2019 1

하단 명령 입력 패널을 이용하는 방법

1. 화면 하단 명령 입력 패널에서 변환 룰 (커맨드) 을 선택합니다.

The screenshot shows a data table with columns for date, values, and year. A dropdown menu is open over the table, listing various transformation commands with their descriptions:

- header**: 지정된 행의 값을 필드 이름으로 지정합니다.
- keep**: 조건을 만족하는 행만 유지합니다.
- replace**: 특정 열에서 패턴을 만족하는 값을 새로운 값으로 대체합니다.
- rename**: 새로운 필드 이름을 입력합니다.
- set**: 특정 열의 값을 수식의 결과치로 대체합니다.
- settype**: (description partially obscured)

At the bottom of the dropdown, there is a '커맨드' (Command) dropdown menu. To the right of the table, there are '취소' (Cancel) and '추가' (Add) buttons.

2. 추가 입력이 필요한 경우 내용을 더 입력한 후 추가 버튼을 누릅니다.

- 대상 컬럼을 고르는 입력 패널이 있지만, 이 경우에도 컬럼 헤더를 클릭해서 컬럼을 지정할 수도 있습니다.

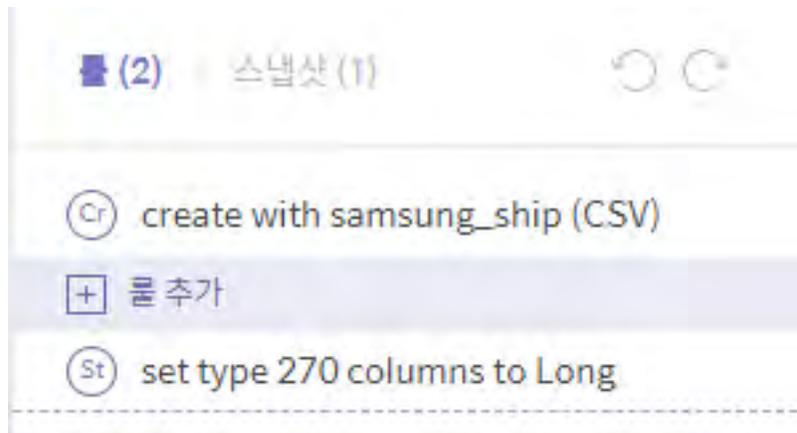
The screenshot shows a dialog box titled '룰 추가 에디터로 전환' (Switch to Rule Addition Editor). It contains the following fields:

- 커맨드** (Command): A dropdown menu with 'rename' selected.
- 컬럼 *** (Column): A dropdown menu with 'column5' selected.
- 새로운 컬럼 이름 *** (New Column Name): A text input field containing 'new_column'.

There are '취소' (Cancel) and '추가' (Add) buttons at the top right, and a '전체 컬럼 변경' (Change All Columns) button at the bottom right.

룰 리스트 중간에 삽입하는 방법

1. 화면 우측의 변환 룰 리스트에서 새 룰을 삽입하고자 하는 룰 경계에 마우스를 갖다대면 + 룰 추가 버튼이 생깁니다. 이 버튼을 누르십시오.



2. 화면 하단 명령 입력 패널에서 변환 룰 (커맨드) 를 선택하고 추가 내용을 입력한 후, **추가** 버튼을 누릅니다.

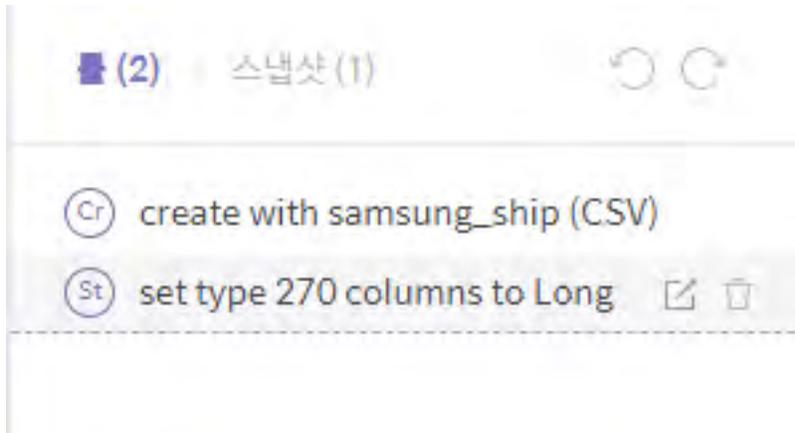
- 이렇게 룰을 중간에 삽입하면, 삽입된 위치 이후의 모든 룰이 영향을 받습니다.
- 이 때 정상적으로 수행될 수 없는 룰이 생기면, 빨간 색으로 표시되고 해당 스텝은 이전 스텝의 결과를 그대로 갖게 됩니다.



생성된 룰 편집하기

룰 수정

1. 화면 우측의 변환 룰 리스트에서 수정하고자 하는 룰 위에 마우스를 갖다대면  버튼이 생깁니다. 이 버튼을 누르십시오.



2. 화면 하단 명령 입력 패널에서 변환 룰 내용을 수정한 후, **완료** 버튼을 누릅니다.

- 룰이 수정되면 그 이후의 모든 룰이 영향을 받습니다.

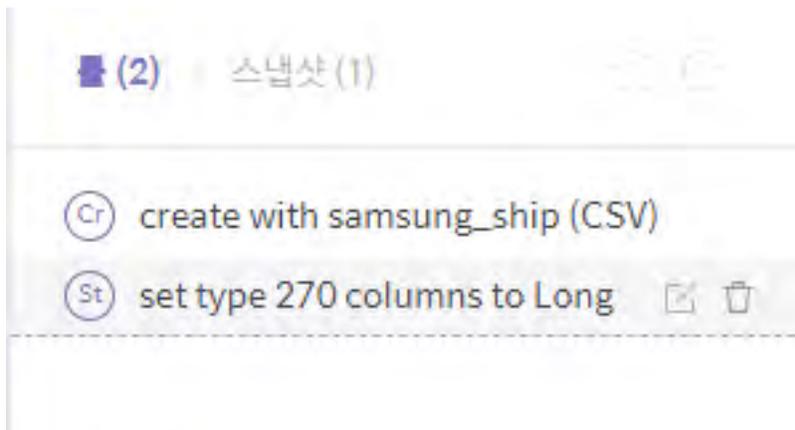
룰 수정 [에디터로 전환](#) 취소 **완료**

커맨드 | 필명* | 새로운 타입*

룰 삭제

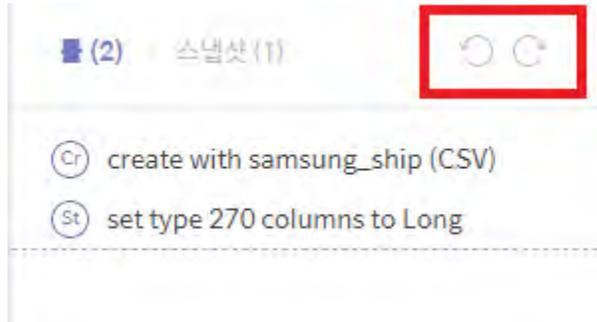
화면 우측의 변환 룰 리스트에서 삭제하고자 하는 룰 위에 마우스를 갖다대면  버튼이 생깁니다. 이 버튼을 누르십시오.

- 선택한 룰이 제거되면 그 이후 모든 룰이 모두 영향을 받습니다.



undo 및 redo

화면 우측의 변환 룰 리스트 상단에는 **undo** 및 **redo** 동작을 수행할 수 있는 아이콘이 있습니다.



어떤 명령 수행 후, 직전의 상태로 되돌리고자 할 때엔  버튼을 클릭하십시오.

- 데이터셋에 대한 변형 (룰 생성, 수정, 삭제 모두 포함) 이 직전의 상태로 되돌아갑니다.
- 해당 변형의 영향을 받은 다른 모든 룰도 직전의 상태가 됩니다.

다시 그 명령을 수행하고자 할 때엔  버튼을 클릭하십시오.

- 해당 명령을 그대로 다시 수행하는 것보다는  를 이용하는 것이 더 빠릅니다. 그 결과가 메모리에 저장되어있기 때문입니다.

8.4.3 룰의 종류

본 절에서는 각 룰을 다음과 같은 항목으로 구분하여 설명합니다.

- 룰 이름
- 필수 인자
- 선택 인자
- 상세 설명
- 주의사항

현재 데이터 프리퍼레이션에서 지원하는 룰 종류는 다음과 같습니다.

- `drop`
- `header`

- settype
- setformat
- rename
- keep
- delete
- replace
- set
- derive
- split
- merge
- extract
- countpattern
- nest
- unnest
- flatten
- aggregate
- pivot
- unpivot
- join
- union
- window

이러한 룰과 더불어 각종 수식 함수를 제공함으로써, 데이터 프리퍼레이션은 일반적인 데이터 정제에 필요한 대부분의 기능들을 지원하고 있습니다.

drop

필수 인자

- 컬럼: 대상 컬럼 리스트

상세 설명

- 선택된 컬럼들을 삭제합니다.

header

필수 인자: 컬럼명을 담고 있는 행 번호 (1-base)

상세 설명

- 지정된 행의 내용을 컬럼명으로 설정합니다.
- 첫 행에 컬럼명이 있는 CSV 파일을 읽어들이는 때에 유용합니다.
- 특별한 설정이 없는 한, 데이터 프리퍼레이션은 자동으로 header를 수행합니다. 자동 적용된 header의 결과를 원치 않는 경우 해당 룰을 삭제하면 되지만, 보통 그런 일은 흔치 않습니다.

settype

필수 인자

- 컬럼: 대상 컬럼 리스트
- 새로운 타입: Long, Double, String, Boolean, Timestamp 중 택 1

선택 인자

- 포맷 지정: Timestamp의 경우 format string (Joda time)

상세 설명

- 선택된 컬럼들의 타입을 바꿉니다.
- Type mismatch가 발생해도 룰은 성공하며, type mismatch는 이후 따로 해결해주어야 합니다.

setformat

필수 인자

- 컬럼: 대상 컬럼 리스트
- 포맷 지정: Jodatime의 포맷 스트링

상세 설명

- Timestamp 컬럼의 화면 표시 형식을 바꿉니다.
- 대상 컬럼이 반드시 Timestamp 타입이어야 합니다.

주의사항

- 포맷 지정 입력창은 아래처럼 입력에 따라 제시되는 리스트가 변합니다. 원하는 포맷을 앞에서부터 치다보면 리스트에 나오는 후보가 점점 좁혀집니다.



rename

필수 인자

- 컬럼: 대상 컬럼 (1개)
- 새로운 컬럼 이름: 새로운 이름

상세 설명

- 선택된 컬럼의 이름을 변경합니다.
- 2개 이상의 컬럼들에 대해 한번에 rename을 수행하고자 할 때엔 하단 명령 입력창에 있는 전체 컬럼 변경 버튼을 클릭하면 다음과 같은 팝업창이 뜹니다.

Rename

취소

완료

sale

26 column(s)

전	후
주문일	주문일
카테고리	카테고리
도시	도시
국가	국가
주문자	주문자
상세내역	상세내역
column9	column9
column10	column10
column11	column11
column12	column12
column13	column13

주문일	ab 카테고리	ab 도시	ab 국가	ab 주문자
2011-01-04T00:00:00.000Z	Office Supplies	Houston	United States	Darre
2011-01-05T00:00:00.000Z	Office Supplies	Naperville	United States	Philli
2011-01-05T00:00:00.000Z	Office Supplies	Naperville	United States	Philli
2011-01-05T00:00:00.000Z	Office Supplies	Naperville	United States	Philli
2011-01-06T00:00:00.000Z	Office Supplies	Philadelphia	United States	Mick
2011-01-07T00:00:00.000Z	Office Supplies	Athens	United States	Jack
2011-01-07T00:00:00.000Z	Office Supplies	Los Angeles	United States	Lycor
2011-01-07T00:00:00.000Z	Furniture	Henderson	United States	Marie
2011-01-07T00:00:00.000Z	Office Supplies	Henderson	United States	Marie
2011-01-07T00:00:00.000Z	Office Supplies	Henderson	United States	Marie
2011-01-07T00:00:00.000Z	Office Supplies	Henderson	United States	Marie
2011-01-07T00:00:00.000Z	Office Supplies	Henderson	United States	Marie
2011-01-07T00:00:00.000Z	Technology	Henderson	United States	Marie

keep

필수 인자

- 조건: Boolean이 결과로 나오는 조건식

상세 설명

- 조건식을 참으로 하는 행만 남기고 나머지 행을 지웁니다.

The screenshot displays the Metatron Data Catalog interface for a table named 'sale'. The table has 26 columns and 9,994 rows. The columns are: 주문일, ab 카테고리, ab 도시, ab 국가, and ab 주문자. The table is filtered to show only rows where the '카테고리' column is 'Furniture'. The interface includes a search bar, a filter panel on the right, and a 'keep' filter applied to the '카테고리' column. The filter panel shows a list of operations such as 'create with sale (CSV)', 'set type column1 to Timestamp', and 'set format column1 to yyyy-MM-dd'.

주문일	ab 카테고리	ab 도시	ab 국가	ab 주문자
01-04-11	Office-Supplies	Houston	United States	Darren Powers
01-05-11	Office-Supplies	Naperville	United States	Phillina Ober
01-05-11	Office-Supplies	Naperville	United States	Phillina Ober
01-05-11	Office-Supplies	Naperville	United States	Phillina Ober
01-06-11	Office-Supplies	Philadelphia	United States	Mick Brown
01-07-11	Office-Supplies	Athens	United States	Jack OBriant
01-07-11	Office-Supplies	Los Angeles	United States	Lycoris Saunder
01-07-11	Furniture	Henderson	United States	Maria Etezadi
01-07-11	Office-Supplies	Henderson	United States	Maria Etezadi
01-07-11	Office-Supplies	Henderson	United States	Maria Etezadi
01-07-11	Office-Supplies	Henderson	United States	Maria Etezadi

delete

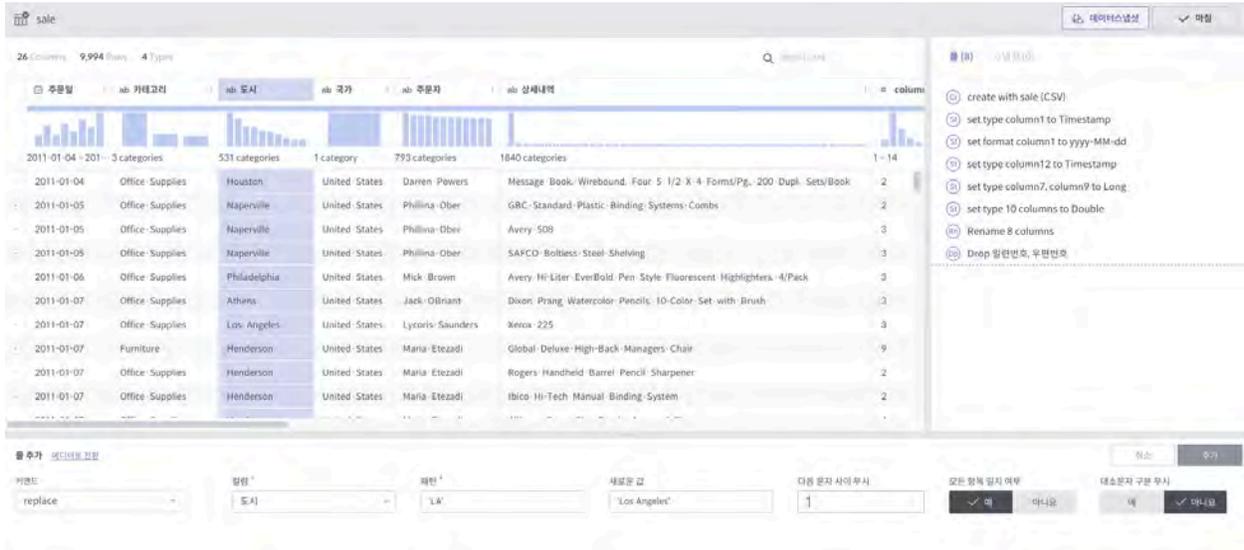
필수 인자

- 조건: Boolean이 결과로 나오는 조건식

상세 설명

- 조건식을 참으로 하는 행을 모두 지웁니다. `keep`과 정반대로 동작합니다.

replace



필수 인자

- 컬럼: 대상 컬럼 리스트
- 패턴: 갈아치울 대상 문자열 패턴
 - 상수 문자열인 경우: '로 감싸져 있는 경우 ('seoul', '서울', '서울 특별시' 등)
 - 정규식인 경우: /로 감싸져 있는 경우 (/[,_]+/, /\s+\$/ 등)
- 새로운 값: 새롭게 놓일 문자열 수식
 - 상수 문자열
 - 정규식의 그룹을 이용한 문자열 수식: \$1_\$2_\$3 등

선택 인자

- 다음 문자 사이 무시: 이 안에 입력된 문자 사이 내용에 대해서는 치환을 하지 않습니다.
- 모든 항목 일치 여부: 단어의 모든 문자가 일치해야하는지 여부
- 대소문자 구분 무시: 대소문자를 동일하게 취급할지 여부

상세 설명

- 선택된 컬럼들에 대하여, 문자열 변환을 수행합니다.

주의사항

- 새로운 값에는 '이나 /를 사용하지 않습니다.
- 새로운 값에 다른 컬럼의 값을 이용할 수 없습니다. replace는 순수히 해당 컬럼 내용안에서의 문자열 변환입니다. (cf. set 룰)

set

The screenshot displays the Metatron Data Platform interface for a table named 'sale'. The table has 26 columns and 741 rows. The columns are: 주문일, 카테고리, 도시, 국가, 주문자, and 상세내역. The interface shows a data grid with columns for 주문일, 카테고리, 도시, 국가, 주문자, and 상세내역. The 'set' rule is selected in the sidebar, and its configuration is shown in the main pane. The rule configuration includes a '조건' (Condition) field with 'set' selected, and a '수식' (Expression) field with a complex SQL-like expression: 'Keep rows where time_between(주문일, '2013-07-01T00:00:00.000Z', '2013-10-01T00:00:00.000Z')'.

필수 인자

- 컬럼: 대상 컬럼 리스트
- 수식: 대상 컬럼의 값이 될 수식. 다른 컬럼값을 참조할 수 있습니다. (cf. replace 룰)
 - 여러 컬럼을 대상으로 하는 경우, \$col이라고 쓰면 각 컬럼에 대해 변환을 할 때, 그 대상 컬럼을 가리키도록 치환됩니다.
 - 즉, column1, column2에 대해 set 명령을 사용하는 경우, column1에 대해 변환을 할 때엔 \$col이 column1 이 되고, column2에 대해 변환을 할 때엔 \$col이 column2가 됩니다.

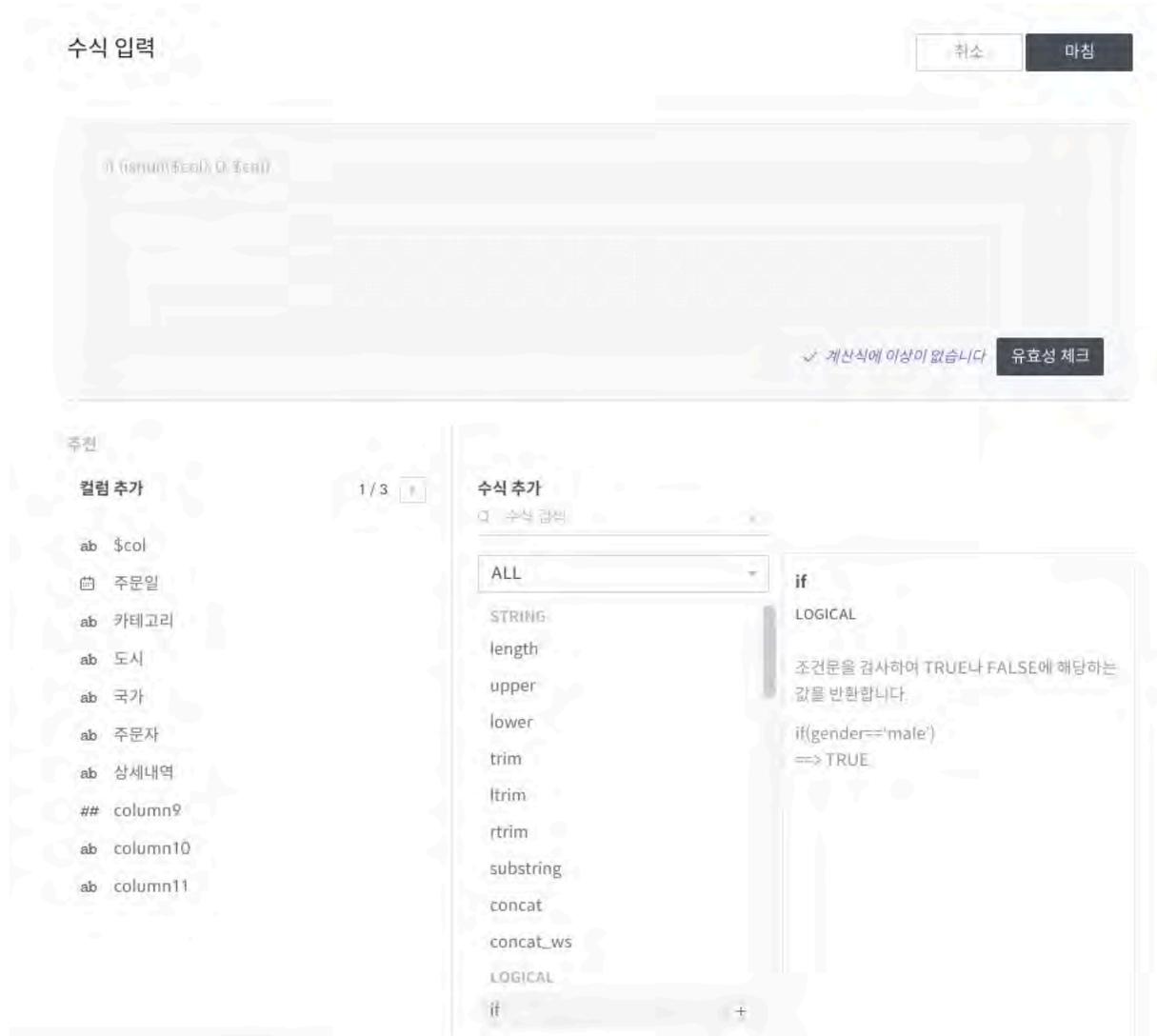
선택 인자

- 다음 조건에서만 수행
 - 이 조건을 만족하는 행에 대해서만 set 룰을 적용합니다.

- SQL문에서 WHERE과 같다고 생각하면 됩니다.

상세 설명

- 해당 컬럼의 값을 주어진 수식의 결과값으로 대체합니다.
- 복잡한 수식을 사용하는 경우 고급 편집기를 클릭하면 다음과 같은 팝업창이 뜹니다.



고급 편집기를 활용하면 컬럼의 리스트와 함수의 리스트 및 각 설명, 예시를 보면서 넓은 창에서 수식을 편집할 수 있고, 실제로 쿼리를 실행하기 전에 수식의 유효성을 체크해볼 수 있습니다.

derive

필수 인자

- 수식: 새로운 컬럼의 값이 될 수식. `set` 룰과 마찬가지로 다른 컬럼값을 참조할 수 있습니다.
- 새로운 컬럼 이름

상세 설명

- `set` 룰과 비슷하지만, 어떤 컬럼의 값을 대체하는 것이 아니라 새로운 컬럼을 만들어냅니다.

주의사항

- 수식에 등장하는 컬럼 중에 제일 마지막 컬럼 뒤로 삽입됩니다.

split

필수 인자

- 컬럼: 대상 컬럼 리스트
- 패턴: `split`의 기준이 되는 문자열 수식. `replace` 룰과 같이 정규식을 허용합니다.
- 횡수: 몇 개의 컬럼으로 나눌 것인지 여부입니다.

상세 설명

- 각 행에 대해 주어진 횡수 - 1 만큼 `split`을 합니다.
- 컬럼 내용에 패턴이 더 이상 없을 경우 `null` 값을 가진 컬럼을 만들어냅니다.

주의사항

- 횡수에 해당하는 개수의 컬럼이 생기게 된다는 것에 유의하세요.

merge

필수 인자

- 컬럼: 대상 컬럼 리스트
- 구분자: 컬럼들을 이을 상수 문자열
- 새로운 컬럼 이름

상세 설명

- 대상 컬럼들을 구분자로 이어서 새로운 컬럼을 만듭니다.

주의사항

- `replace` 룰도 마찬가지로, ' 로 감싸는 것은 생략할 수 있습니다. 즉, / 로도, ' 로도 감싸지지 않는 문자열이 입력되었을 시, 알아서 ' 로 감싸서 전달되게 되어있습니다.

extract

필수 인자

- 컬럼: 대상 컬럼 리스트
- 패턴: 추출할 문자열 패턴. `replace` 룰과 마찬가지로 정규식을 허용합니다.
- 횟수: 추출할 횟수

선택 인자

- 다음 문자 사이 무시: 이 안에 입력된 문자 사이 내용에 대해서는 치환을 하지 않습니다.
- 대소문자 구분 무시: 대소문자를 동일하게 취급할지 여부

상세 설명

- 패턴에 매치되는 내용으로 새로운 컬럼을 만듭니다.

주의사항

- 여러 개의 대상 컬럼이 있는 경우, 추출의 결과는 각 컬럼의 뒤로 붙습니다.

countpattern

필수 인자

- 컬럼: 대상 컬럼 리스트
- 패턴: 찾아낼 문자열 패턴. `replace` 룰과 마찬가지로 정규식을 허용합니다.

선택 인자

- 다음 문자 사이 무시: 이 안에 입력된 문자 사이 내용에 대해서는 치환을 하지 않습니다.
- 대소문자 구분 무시: 대소문자를 동일하게 취급할지 여부

상세 설명

- 패턴에 매치되는 내용이 몇 군데에 있는지 세어서, 그 숫자로 새 컬럼을 만듭니다.
- `extract`와 상당히 비슷합니다. 내용을 추출하는 것이 아니라, 그 숫자를 세는 것만 다를 뿐입니다.

주의사항

- 여러 개의 대상 컬럼이 있는 경우, 추출의 결과는 각각 컬럼의 뒤로 붙습니다.

nest

필수 인자

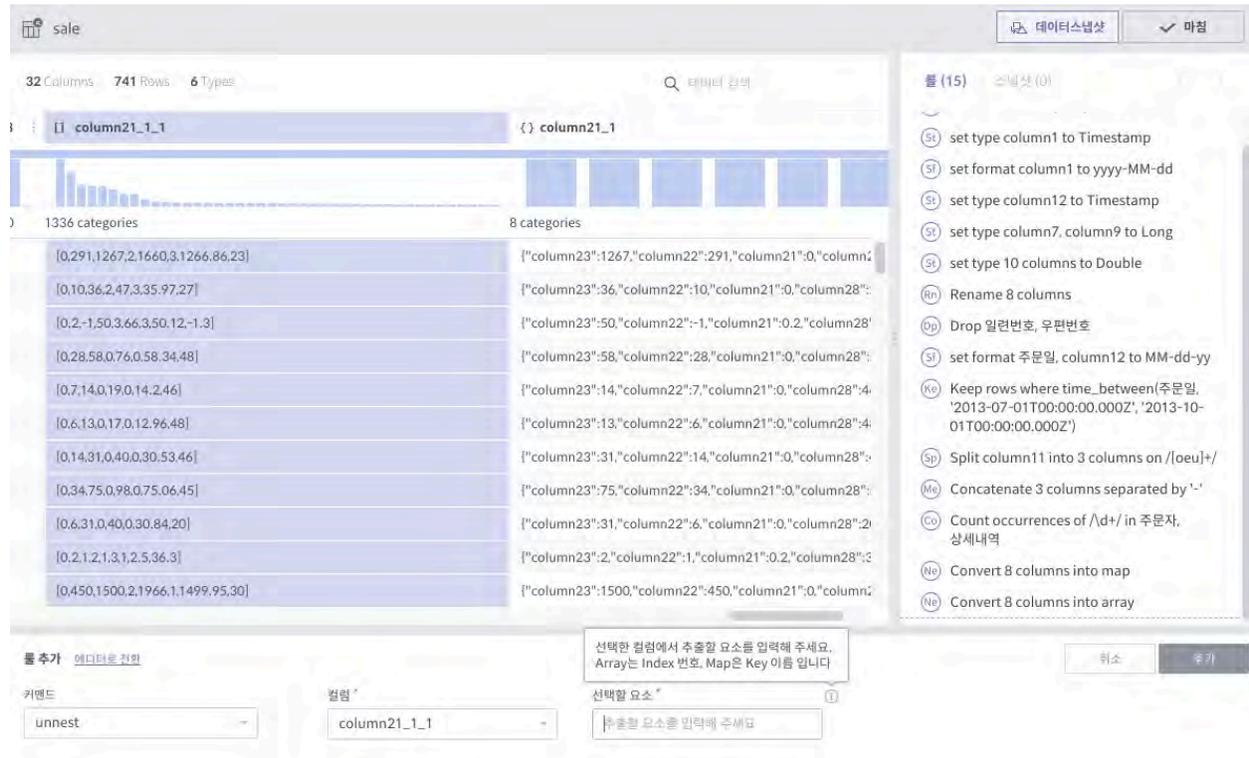
- 컬럼: 대상 컬럼 리스트
- 타입: Map 또는 Array
- 새로운 컬럼 이름

상세 설명

- 대상 컬럼을 주어진 타입으로 묶습니다.
- 다음은 각각 Array, Map으로 묶여진 경우입니다.

The screenshot displays the Metatron Data Catalog interface for a table named 'sale'. The table has 32 columns and 741 rows. The selected columns are 'column28' and 'column21_1'. The data is shown in a grid format with various values, including arrays and JSON objects. A right-hand sidebar shows a list of operations for the selected columns, such as 'set type column1 to Timestamp', 'set format column1 to yyyy-MM-dd', 'Rename 8 columns', and 'Convert 8 columns into array'.

unnest



필수 인자

- 컬럼: 대상 컬럼 (1개)
- 선택할 요소: Array의 경우 0-base index, Map의 경우 key값

상세 설명

- Array 또는 Map에서 지정된 요소를 빼서 새 컬럼으로 만듭니다.

주의사항

- 대상 컬럼은 반드시 Array 또는 Map 타입이어야 합니다.

flatten

필수 인자

- 컬럼: 대상 컬럼 (1개)

상세 설명

- Array의 각각 원소를 해당 컬럼의 값으로 삼는 행을 만들어냅니다.

주의사항

- 대상 컬럼은 반드시 Array 타입이어야 합니다.

The screenshot shows the Metatron Data Studio interface. The main view displays a table with 32 columns and 741 rows. The selected column is 'column21_1_1', which contains array values. The right sidebar shows a list of 16 operations, with 'flatten' selected. The bottom panel shows the 'flatten' operation configuration with the target column set to 'column21_1_1'.

Product Name	Array Value
OSullivan · 2-Door · Barrister · Bookcase · in · Odessa · Pine	[1267,2,1660,3]
Bulldog · Vacuum · Base · Pencil · Sharpener	[36,2,47,3]
Kingston · Digital · DataTraveler · 16GB · USB · 2.0	[50,3,66,3]
Binding · Machine · Supplies	[58,0,76,0]
Wilson · Jones · Hanging · View · Binder · White · 1	[14,0,19,0]
Xerox · 210	[13,0,17,0]
lpright · Avery · 477	[31,0,40,0]
lpright · Southworth · 25% · Cotton · Antique · Laid · Paper · & · Envelopes	[75,0,98,0]
lpright · KeyTronic · KT400U2 · Keyboard · Black	[31,0,40,0]
ky · Zipper · Ring · Binder · Pockets	[2,1,3,1]
en · Hewlett · Packard · 310 · Color · Digital · Copier	[1500,2,1966,1]
jk · Global · Airflow · Leather · Mesh · Back · Chair · Black	[528,1,692,1]

위와 같이 Array 컬럼에 4개의 원소가 있는 경우, 각 원소의 값에 대해 1개씩 행이 생깁니다. 이 때 대상 Array 컬럼을 제외한 모든 컬럼들의 값은 동일하게 됩니다.

The screenshot displays the Metatron Data Editor interface. At the top, it shows 'sale' as the data source, with 32 columns, 2,964 rows, and 5 types. A search bar contains '데이터 검색'. Below this, a table of data is visible with columns for '문자', 'ab 상세내역', and 'column21_1_1'. The table lists various product categories like 'Fox' and 'Kingston' with their respective values. On the right side, a transformation menu is open, showing 17 items. The first item is 'set type column1 to Timestamp'. Other items include 'set format column1 to yyyy-MM-dd', 'set type column12 to Timestamp', 'set type column7, column9 to Long', 'set type 10 columns to Double', 'Rename 8 columns', 'Drop 일련번호, 우편번호', 'set format 주문일, column12 to MM-dd-yy', 'Keep rows where time_between(주문일, '2013-07-01T00:00:00.000Z', '2013-10-01T00:00:00.000Z')', 'Split column11 into 3 columns on /[oeu]+/', 'Concatenate 3 columns separated by '-'', 'Count occurrences of /[d+]/ in 주문자, 상세내역', 'Convert 8 columns into map', 'Convert 4 columns into array', and 'Move column21_1_1 after 상세내역'. At the bottom, there are buttons for '클 추가', '메타데이터 전환', '취소', and '완료'.

aggregate

sale - aggregate

6 Columns 9,994 Rows 3 Types

ab 주문월 ab 권역 배송일 ab 배송등급 ab 주 # 배송기간

48 categories 4 categories 2011-01-08 ~ 2015-01-06 4 categories 49 categories 0 - 7

2011-01	Central	11-Jan-08	Standard Class	Texas	4
2011-01	Central	11-Jan-09	Standard Class	Illinois	4
2011-01	Central	11-Jan-09	Standard Class	Illinois	4
2011-01	Central	11-Jan-09	Standard Class	Illinois	4
2011-01	East	11-Jan-13	Standard Class	Pennsylvania	7
2011-01	South	11-Jan-08	First Class	Georgia	1
2011-01	West	11-Jan-09	Second Class	California	2
2011-01	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	South	11-Jan-11	Standard Class	Kentucky	4

플 추가 [에디터로 전환](#)

커맨드 aggregate

수식 * avg('배송기간')

그룹화 기준 * 주문일, 권역, 배송등급

취소 추가

필수 인자

- 수식: Aggregation 함수 리스트
- 그룹화 기준: Group By에 쓰일 컬럼 리스트

상세 설명

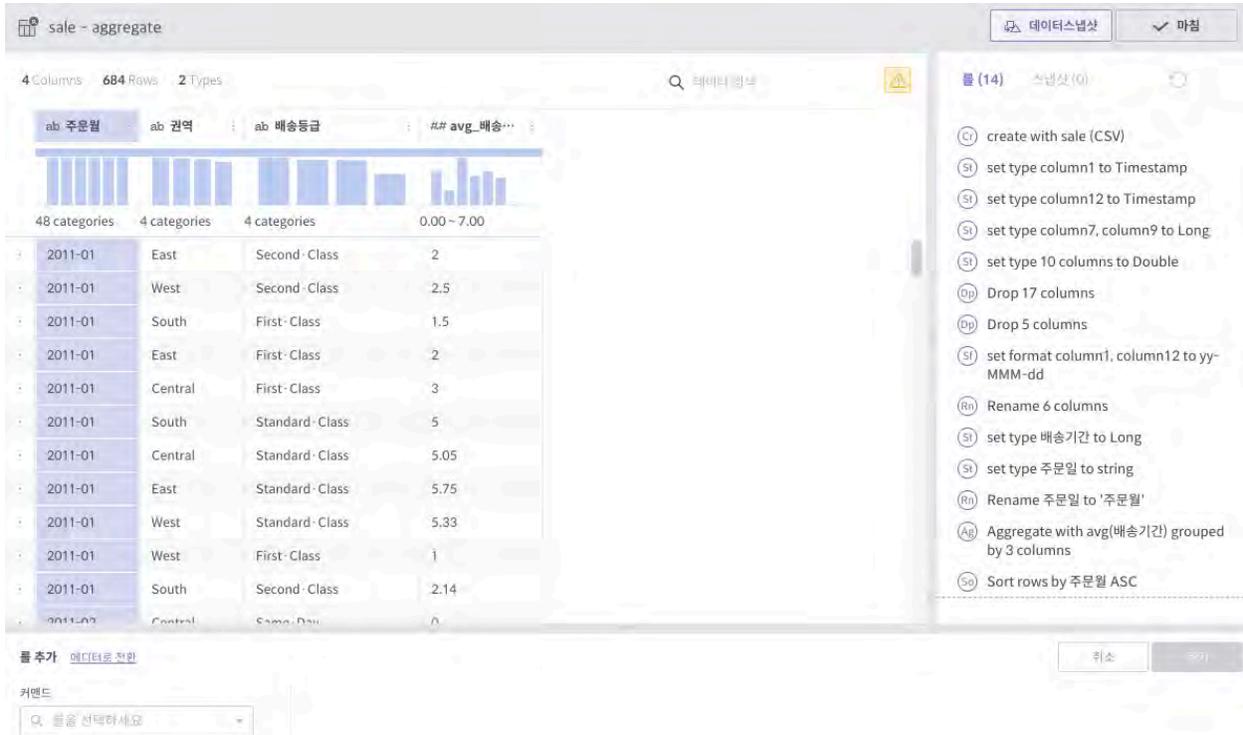
- 그룹화 기준 컬럼들 각 조합에 대해 Group By 연산을 수행한 결과를 새로운 컬럼으로 추가합니다.
- 각 수식 당 한 컬럼씩 생깁니다. 예를 들어, 평균값과 카운트를 수식으로 지정하였을 경우, 2개의 컬럼이 생깁니다.
- 현재 지원하는 Aggregation 함수는 다음과 같습니다.

- count()
- sum(colname)
- avg(colname)
- min(colname)

- max(colname)

주의사항

- 샘플링된 결과에 대해서만 연산을 수행합니다. 때문에 전체 데이터에 대한 결과, 즉 스냅샷은 달라질 수 있습니다.
- count함수 사용시 () 를 꼭 붙여야 하는 것에 유의하십시오.
- count(colname) 은 현재 지원하지 않습니다.



pivot

필수 인자

- 컬럼: 피벗 대상 컬럼 리스트
- 수식: 컬럼의 값이 될 수식 리스트 (Aggregation 함수만 가능)
- 그룹화 기준: Group By에 쓰일 컬럼 리스트

상세 설명

- 대상 컬럼들과 그룹화 기준 컬럼들 각 조합에 대해 Group By 연산을 수행하고, 그 결과를 새로운 컬럼값으로 하는 데이터셋을 만듭니다.
- 각 수식에 대해 컬럼 세트들이 생깁니다. 예를 들어, 평균값과 카운트를 수식으로 지정하였을 경우, 피벗 대상 컬럼들의 값이 결국 10개의 그룹으로 나뉠 경우, 20개의 컬럼이 생겨나게 됩니다.

주의사항

- 최소 2개의 컬럼에 대한 복합 Group By를 할 때에 사용됩니다. (피벗 대상 1개, 그룹화 기준 1개)
- 일반적으로 컬럼명이 길어지기 때문에, 뒤이어 전체 rename을 필요로 하는 경우가 많습니다.

sale
데이터스냅샷
마침

17 Columns 48 Rows 2 Types
데이터필드

ab 주문월
중부 당일...
중부 특급...
중부 무등...
중부 일반...
동부 당일...
동부 특급...
동부 무등...

48 categories

Year	0.00 ~ 0.33	0.00 ~ 3.00	0.00 ~ 5.00	4.06 ~ 5.80	0.00 ~ 0.00	0.00 ~ 3.00	0.00 ~ 4.25
2012-05	0	2.5	3.47	4.76	0	2.47	2.83
2012-06	0	3	3	4.06	0	3	4
2012-07	0	2	4.93	5.5	0	1.94	2.5
2012-08	0	1.33	4	4.96	0	2.92	2.55
2012-09	0	2.67	3	4.83	0	2.33	3.76
2012-10	0	3	3.78	5.04	0	1.88	3.13
2012-11	0	1	2.78	4.83	0	2.18	2.77
2012-12	0.13	2.33	3.23	5.05	0	2.22	4.07
2013-01	0	0	2	4.46	0	2	2.89
2013-02	0	2	3	4.88	0	1.43	3.33
2013-03	0	2.63	3.8	5.09	0	2.8	2.13

플 추가 [에디터로 전환](#)

커멘드

취소

▶

플 (18) 소련식 (0)

- ⌵ Drop 17 columns
- ⌵ Drop 5 columns
- SF set format column1, column12 to yy-MMM-dd
- Rn Rename 6 columns
- St set type 배송기간 to Long
- St set type 주문일 to string
- Rn Rename 주문일 to '주문월'
- Pv Pivot 권역, 배송등급 and compute avg(배송기간) grouped by 주문월
- Rn Rename 16 columns
- Mv Move 중부 당일배송 before 중부 특급배송
- Mv Move 동부 당일배송 before 동부 특급배송
- Mv Move 남부 당일배송 before 남부 특급배송
- Mv Move 서부 당일배송 before 서부 특급배송

unpivot

The screenshot shows the Metatron Data Editor interface for a pivot table named 'sale - pivot'. The table has 17 columns and 48 rows. The columns are grouped into categories: 남부 당일, 남부 특급, 남부 우등, 남부 일반, 서부 당일, 서부 특급, 서부 우등, and 서부 일반. The values are numerical. The interface includes a toolbar with '데이터스냅샷' and '마침' buttons, a search bar, and a list of 18 actions on the right. The main area shows a pivot table with columns for categories and values. Below the table, the 'unpivot' operation is configured with '컬럼' (columns) and '그룹 수' (group count) fields.

필수 인자

- 컬럼: 컬럼값으로 내릴 대상 컬럼들 리스트
- 그룹 수: 결과 컬럼 숫자 (기본적으로 1)

상세 설명

- 선택된 컬럼들에 대해 컬럼 이름과 컬럼의 값을 내용으로 하는 컬럼 2개를 만듭니다. (그룹 수가 1인 경우)
- 그룹 수가 선택된 컬럼 숫자와 같은 경우, 각 컬럼 이름과 값에 해당하는 컬럼들을 만듭니다. 즉, 10개 컬럼에 대해 그룹 수 10으로 unpivot을 하면, 총 20개 컬럼이 생깁니다.

주의사항

- 그룹 수가 대상 컬럼 수의 약수인 경우는 곧 지원할 예정입니다.

〈그룹 수가 1인 경우〉

The screenshot shows the Metatron Data Catalog interface for a dataset named 'sale - pivot'. The main view displays a pivot table with 3 columns and 768 rows. The columns are '주문월' (Order Month), 'key1' (Key 1), and 'value1' (Value 1). The pivot table shows data for the year 2011, grouped by month and region (중부, 동부, 남부). A bar chart above the table shows the distribution of values across categories.

On the right side, a list of operations is displayed, including:

- Drop 17 columns
- Drop 5 columns
- set format column1, column12 to yy-MMM-dd
- Rename 6 columns
- set type 배송기간 to Long
- set type 주문월 to string
- Rename 주문월 to '주문월'
- Pivot 권역, 배송등급 and compute avg(배송기간) grouped by 주문월
- Rename 16 columns
- Move 중부 당일배송 before 중부 특급배송
- Move 동부 당일배송 before 동부 특급배송
- Move 남부 당일배송 before 남부 특급배송
- Move 서부 당일배송 before 서부 특급배송
- Convert 16 columns into rows

At the bottom, there is a '플 추가' (Add Filter) button and a '커맨드' (Command) input field with a search icon and the text '결과를 선택하세요' (Select results).

<그룹 수가 컬럼 수와 같은 경우>

sale - pivot

33 Columns 48 Rows 2 Types

ab 주문월 ab key1 ## value1 ab key2 ## value2 ab key3 ## value3 ab key4 ## value4

48 categories 1 category 0.00 ~ 0.33 1 category 0.00 ~ 3.00 1 category 0.00 ~ 5.00 1 category 4.06 ~ 5.80

2011-01	중부·달일배송	0	중부·특급배송	3	중부·우등배송	0	중부·일반배송	5.05
2011-02	중부·달일배송	0	중부·특급배송	2	중부·우등배송	4.13	중부·일반배송	5.22
2011-03	중부·달일배송	0	중부·특급배송	2.4	중부·우등배송	0	중부·일반배송	5.1
2011-04	중부·달일배송	0	중부·특급배송	2	중부·우등배송	2.7	중부·일반배송	4.33
2011-05	중부·달일배송	0	중부·특급배송	2.17	중부·우등배송	2.22	중부·일반배송	4.67
2011-06	중부·달일배송	0	중부·특급배송	2.5	중부·우등배송	4.31	중부·일반배송	5.09
2011-07	중부·달일배송	0	중부·특급배송	3	중부·우등배송	3	중부·일반배송	4.42
2011-08	중부·달일배송	0	중부·특급배송	3	중부·우등배송	5	중부·일반배송	4.92
2011-09	중부·달일배송	0	중부·특급배송	2.36	중부·우등배송	4.27	중부·일반배송	5.47
2011-10	중부·달일배송	0	중부·특급배송	2.5	중부·우등배송	3.09	중부·일반배송	4.63

플 (19) | 스크린샷 (0)

- set type 10 columns to Double
- Drop 17 columns
- Drop 5 columns
- set format column1, column12 to yy-MMM-dd
- Rename 6 columns
- set type 배송기간 to Long
- set type 주문일 to string
- Rename 주문일 to '주문일'
- Pivot 권역, 배송등급 and compute avg(배송기간) grouped by 주문일
- Rename 16 columns
- Mv Move 중부 달일배송 before 중부 특급배송
- Mv Move 동부 달일배송 before 동부 특급배송
- Mv Move 남부 달일배송 before 남부 특급배송

플 추가 | 데이터로 전환

키보드

Q, 플을 선택하세요

취소

join



join은 다른 룰들과는 달리, 별도의 팝업창을 갖습니다.

필수 인자 (팝업에서 선택하거나 입력)

- join 대상 데이터셋: 같은 데이터플로우 내의 Wrangled 데이터셋
- join 결과로 나올 컬럼들 (토큰)
- join 키: 여러 개 입력 가능
- join 타입: 현재 내부조인만 지원

상세 설명

- 대상 데이터셋과 연결해서 컬럼들을 만들어 냅니다.
- 기본적으로 관계형 데이터베이스의 join과 같습니다.
- **결과보기** 버튼으로 실제 룰적용 전에 join 결과를 볼 수 있습니다.

주의사항

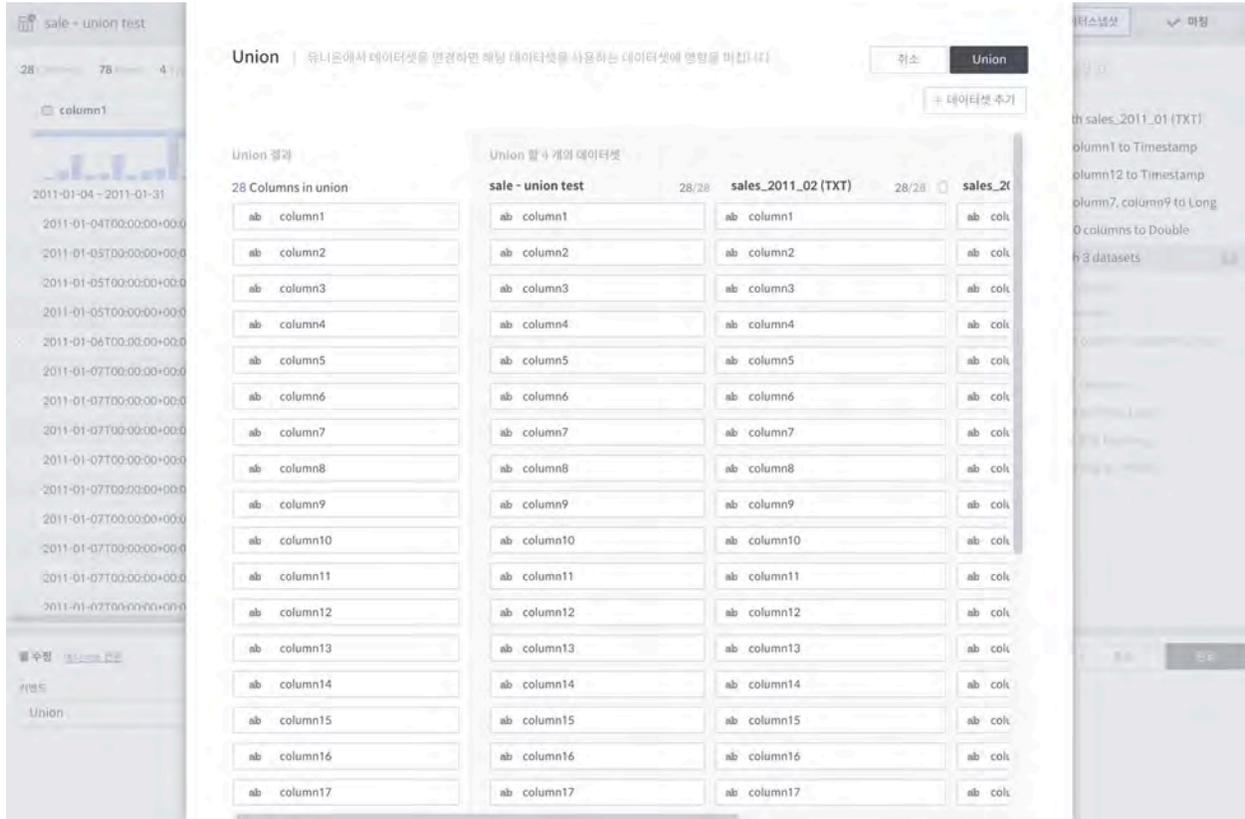
- 결과로 나올 컬럼에 join 키가 꼭 포함되어있어야 합니다.

The screenshot shows the Metatron Data Studio interface for a query named 'sale - pivot'. The main view displays a table with 5 columns: 주문월 (Order Month), 권역 (Region), 배송유형 (Delivery Type), 평균기간 (Average Period), and 책임자 (Responsible Person). The table contains 16 rows of data for the year 2011, grouped by region and delivery type. A bar chart above the table shows the distribution of data across categories.

The right-hand side of the interface shows a query editor with 23 steps. The final step is a join operation: 'Join with sale_representative'. The query editor also includes a search bar and a '시작' (Start) button.

주문월	권역	배송유형	평균기간	책임자
2011-01	중부	당일배송	0	James-Lee
2011-01	중부	특급배송	3	James-Lee
2011-01	중부	우등배송	0	James-Lee
2011-01	중부	일반배송	5.05	James-Lee
2011-01	동부	당일배송	0	Charles-Blake
2011-01	동부	특급배송	2	Charles-Blake
2011-01	동부	우등배송	2	Charles-Blake
2011-01	동부	일반배송	5.75	Charles-Blake
2011-01	남부	당일배송	0	Jim-Halbert
2011-01	남부	특급배송	1.5	Jim-Halbert
2011-01	남부	우등배송	2.14	Jim-Halbert
2011-01	남부	일반배송	5	Jim-Halbert
2011-01	서부	당일배송	0	Marry-Main
2011-01	서부	특급배송	1	Marry-Main

union



union 역시 join처럼 별도의 팝업창을 갖습니다.

필수 인자 (팝업에서 선택)

- union 대상 데이터셋: 다수 선택 가능

상세 설명

- 지정된 데이터셋의 내용도 함께 처리합니다.
- 기본적으로 관계형 데이터베이스의 union all과 같습니다.

주의사항

- 대상 데이터셋은 union을 수행하는 데이터셋과 컬럼명과 타입, 그리고 컬럼 개수가 일치해야 합니다.

Union

Union을 하기 위한 데이터셋 추가 닫기 + selection 추가

데이터셋 이름으로 검색해 주세요

<input type="checkbox"/>	데이터셋	타입	마지막 업데이트일
<input type="checkbox"/>	sale - union test	WRANGLD	2019-02-15 18:14
<input type="checkbox"/>	sale - pivot	WRANGLD	2019-02-15 17:56
<input type="checkbox"/>	sale_representative	WRANGLD	2019-02-15 17:55
<input checked="" type="checkbox"/>	sales_2011_02 (TXT)	WRANGLD	2019-02-12 19:34
<input checked="" type="checkbox"/>	sales_2011_03 (TXT)	WRANGLD	2019-02-12 19:34
<input checked="" type="checkbox"/>	sales_2011_04	WRANGLD	2019-02-12 19:34
<input type="checkbox"/>	sale - aggregate	WRANGLD	2019-02-08 22:30
<input type="checkbox"/>	sale	WRANGLD	2019-02-08 21:17
<input type="checkbox"/>	sale	WRANGLD	2019-02-08 21:12

이제 선택

window

The screenshot displays the Metatron Data Catalog interface for a dataset named 'sale - union test'. The main area shows a table with columns: 주문일, 카테고리, 권역, 배송일, 배송등급, 주, and 배송기간. The table contains 17 rows of data. On the right side, a list of 13 filters is applied to the data view. At the bottom, there are controls for windowing and aggregation functions.

주문일	카테고리	권역	배송일	배송등급	주	배송기간
2011-01	Office Supplies	Central	11-Jan-08	Standard Class	Texas	4
2011-01	Office Supplies	Central	11-Jan-09	Standard Class	Illinois	4
2011-01	Office Supplies	Central	11-Jan-09	Standard Class	Illinois	4
2011-01	Office Supplies	Central	11-Jan-09	Standard Class	Illinois	4
2011-01	Office Supplies	East	11-Jan-13	Standard Class	Pennsylvania	7
2011-01	Office Supplies	South	11-Jan-08	First Class	Georgia	1
2011-01	Office Supplies	West	11-Jan-09	Second Class	California	2
2011-01	Furniture	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	Office Supplies	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	Office Supplies	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	Office Supplies	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	Office Supplies	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	Office Supplies	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	Technology	South	11-Jan-11	Standard Class	Kentucky	4
2011-01	Technology	South	11-Jan-11	Standard Class	Kentucky	4

Applied Filters (13):

- create with sales_2011_01 (TXT)
- set type column1 to Timestamp
- set type column12 to Timestamp
- set type column7, column9 to Long
- set type 10 columns to Double
- union with 3 datasets
- Drop 17 columns
- Drop 4 columns
- set format column1, column12 to yyyy-MM-dd
- Rename 7 columns
- set type 배송기간 to Long
- set type 주문일 to string
- Rename 주문일 to '주문일'

Windowing and Aggregation Settings:

- 커널: window
- 수식: rolling_avg(배송기간, 3, 3)
- 그룹화 기준: 주
- 주요 필터: 주문일

The screenshot displays the Metatron Data Catalog interface. At the top, it shows 'sale - union test' with 8 columns, 417 rows, and 4 types. Below this is a data table with columns: 주문일, 카테고리, 권역, 배송일, 배송등급, 주, 배송기간, and 주별 평균 배송기간. The table contains 20 rows of data. To the right, a list of transformations is shown, including 'create with sales_2011_01 (TXT)', 'set type column1 to Timestamp', 'union with 3 datasets', and 'Rename 주문일 to '주문일''. At the bottom, there is a '필 추가' (Required) section with a search box and a '필 추가' button.

필수 인자

- 수식: Window 함수 리스트
- 그룹화 기준: 이 그룹안에서 행의 순서가 만들어짐. 없으면 그냥 전체적으로 정렬 기준 적용
- 정렬 기준: 이 컬럼의 순서로 전후 관계가 만들어짐. 없으면 그냥 데이터가 입력되는 순서

상세 설명

- 앞의 행, 뒤의 행의 내용을 토대로 수식을 계산해서 컬럼값을 생성합니다.
- 그룹화 기준내에서 정렬 기준으로 순서를 정합니다.
 - 예를 들어, 위의 예시에서는 주 (state) 별로 앞뒤 3개씩의 행을 포함해서 평균값을 계산합니다.
 - 화면상에서는 바로 앞에 보인다고 해도, 주가 같지 않으면 더 앞의 행을 보게됩니다.
- 현재 지원하는 Window 함수는 다음과 같습니다.
 - row_number()
 - lead(colname, int)
 - lag(colname, int)

- `rolling_sum(colname, int, int)`
- `rolling_avg(colname, int, int)`
- Window 함수와 더불어 Aggregation 함수도 사용할 수 있습니다.

주의사항

- Window 함수 사용시, 인자 수가 부족하거나 한 상황에 대해 적절한 에러메시지가 제공되지 않습니다. 유의하시기 바랍니다.

8.4.4 함수 목록

틀은 주요 기능을 이용하기 위해서 함수를 사용할 수 있습니다.

본 절에서는 각 함수를 다음과 같은 항목으로 구분하여 설명합니다.

- 카테고리
- 설명
- 함수 형식
- 입력값
- 출력
- 예시
- 기타 사항

현재 데이터 프리퍼레이션에서 지원하는 함수 종류는 다음과 같습니다.

- `length`
- `if`
- `isnull`
- `isnan`
- `upper`
- `lower`
- `trim`
- `ltrim`

- rtrim
- substring
- concat
- concat_ws
- year
- month
- day
- hour
- minute
- second
- millisecond
- now
- add_time
- sum
- avg
- max
- min
- count
- math.abs
- math.acos
- math.asin
- math.atan
- math.cbrt
- math.ceil
- math.cos
- math.cosh

- `math.exp`
- `math.expm1`
- `math.getExponent`
- `math.round`
- `math.signum`
- `math.sin`
- `math.sinh`
- `math.sqrt`
- `math.tan`
- `math.tanh`
- `time_diff`
- `timestamp`
- `row_number`
- `rolling_sum`
- `rolling_avg`
- `lag`
- `lead`
- `ismismatched`
- `contains`
- `startswith`
- `endswith`

함수는 지속적으로 추가 보완될 수 있습니다.

length

카테고리

- String Function

설명

- 입력된 문자열의 길이를 반환합니다.

함수 형식

- length(string_value)

입력값

- string_value: 길이를 구하고자 하는 문자열.

출력

- Integer

예시

- length(first_name)

if

카테고리

- Logical Function

설명

- 조건문을 검사하여 TRUE나 FALSE에 해당하는 값을 반환합니다.

함수 형식

- if(condition)
- if(condition, true_value, false_value)

입력값

- condition: 참/거짓 여부를 검사하고자 하는 조건.
- true_value: 조건문이 참일 경우 반환되는 값.
- false_value: 조건문이 거짓일 경우 반환되는 값.

출력

- Any

예시

- `if(gender=='male') : TRUE`
- `if(age<18, 'kid', 'adult') : 'adult'`

기타 사항

- `true_value/false_value`가 없는 경우에는 Boolean type의 결과 TRUE 혹은 FALSE를 반환합니다.
- `true_value`와 `false_value`의 데이터 타입은 동일해야 합니다.

isnull

카테고리

- Logical Function

설명

- 입력된 컬럼의 값이 null 인지 판단합니다. null이면 TRUE, 아니면 FALSE를 반환합니다.

함수 형식

- `isnull(condition)`

입력값

- `condition`: null 여부를 판단하고자 하는 컬럼.

출력

- Boolean

예시

- `isnull(telephone) : FALSE`

isnan

카테고리

- Logical Function

설명

- 입력된 값이 NaN(Not-a-Number) 인지 판단합니다. NaN이면 TRUE, 아니면 FALSE를 반환합니다.

함수 형식

- isnan(condition)

입력값

- condition: NaN 여부를 판단하고자 하는 컬럼이나 수식.

출력

- Boolean

예시

- isnan(1000/ratio)

기타 사항

- condition의 결과는 Double Value이어야 합니다.

upper

카테고리

- String Function

설명

- 입력된 문자열 내의 알파벳을 모두 대문자로 치환하여 반환합니다.

함수 형식

- upper(string_value)

입력값

- string_value: 대문자로 치환하고자 하는 문자열.

출력

- String

예시

- upper(last_name)
- upper('Hello world') : 'HELLO WORLD'

lower

카테고리

- String Function

설명

- 입력된 문자열 내의 알파벳을 모두 소문자로 치환하여 반환합니다.

함수 형식

- lower(string_value)

입력값

- string_value: 소문자로 치환하고자 하는 문자열.

출력

- String

예시

- lower(last_name)
- lower('Hello WORLD') : 'hello world'

trim

카테고리

- String Function

설명

- 입력된 문자열의 앞/뒤에 있는 공백을 제거하여 반환합니다.

함수 형식

- trim(string_value)

입력값

- string_value: 공백을 제거하고자 하는 문자열.

출력

- String

예시

- `trim(comment)`
- `trim(' . Hi! '): ' . Hi!'`

ltrim

카테고리

- String Function

설명

- 입력된 문자열의 앞(왼쪽)에 있는 공백을 제거하여 반환합니다.

함수 형식

- `ltrim(string_value)`

입력값

- `string_value`: 공백을 제거하고자 하는 문자열.

출력

- String

예시

- `ltrim(comment)`
- `ltrim(' . Hi! '): ' . Hi! '`

rtrim

카테고리

- String Function

설명

- 입력된 문자열의 뒤(오른쪽)에 있는 공백을 제거하여 반환합니다.

함수 형식

- `rtrim(string_value)`

입력값

- string_value: 공백을 제거하고자 하는 문자열.

출력

- String

예시

- rtrim(comment)
- rtrim(' . Hi! '): ' . Hi!'

substring

카테고리

- String Function

설명

- 입력된 문자열의 일부를 반환합니다.

함수 형식

- substring(string_value, begin_index, offset)
- substring(string_value, begin_index)

입력값

- string_value: 편집하고자 하는 문자열.
- begin_index: 대상 문자열에서 추출하고자 하는 부분의 시작 index. 문자열의 처음은 0. 음수로 입력하면 문자열의 마지막 글자부터 거슬러 올라간다.
- offset: 대상 문자열에서 추출하고자 하는 문자열의 길이. 입력하지 않으면 begin_index부터 문자열의 마지막 까지 추출한다.

출력

- String

예시

- substring(user_id, 0, 5)
- substring('hello world', 1, 7) : 'ello w'

- `substring(<metatron>, -2) : <on>`

concat

카테고리

- String Function

설명

- 입력된 복수의 문자열을 연결하여 반환합니다.

함수 형식

- `concat(string_value1, string_value2, string_value3)`

입력값

- `string_value(X)`: 연결하고자 하는 문자열. n개를 복수로 입력 가능.

출력

- String

예시

- `concat(first_name, '-', last_name) : 'Jane-Doe'`
- `concat('1980', '02') : '198002'`

concat_ws

카테고리

- String Function

설명

- 입력된 복수의 문자열을 연결하면서 문자열 사이에 Separator(구분자)를 넣어 반환합니다.

함수 형식

- `concat(separator, string_value1, string_value2)`

입력값

- `separator`: 연결할 문자열들 사이에 들어갈 구분자.

- string_value(X): 연결하고자 하는 문자열. n개를 복수로 입력 가능.

출력

- String

예시

- concat_ws(',', first_name, last_name) : 'Jane, Doe'
- concat_ws('-', '010', '1234', '5678') : '010-1234-5678'

year

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에서 연도에 해당하는 값을 반환합니다.

함수 형식

- year(timestamp_value)

입력값

- timestamp_value: 연도를 추출하고자 하는 timestamp

출력

- Integer

예시

- year(birthday)

month

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에서 월에 해당하는 값을 반환합니다.

함수 형식

- month(timestamp_value)

입력값

- timestamp_value: 월을 추출하고자 하는 timestamp

출력

- Integer

예시

- month(birthday)

day

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에서 일에 해당하는 값을 반환합니다.

함수 형식

- day(timestamp_value)

입력값

- timestamp_value: 일을 추출하고자 하는 timestamp

출력

- Integer

예시

- day(birthday)

hour

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에서 시간에 해당하는 값을 반환합니다.

함수 형식

- hour(timestamp_value)

입력값

- timestamp_value: 시간을 추출하고자 하는 timestamp

출력

- Integer

예시

- hour(last_login)

minute

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에서 분에 해당하는 값을 반환합니다.

함수 형식

- minute(timestamp_value)

입력값

- timestamp_value: 분을 추출하고자 하는 timestamp

출력

- Integer

예시

- minute(last_login)

second

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에서 초에 해당하는 값을 반환합니다.

함수 형식

- `second(timestamp_value)`

입력값

- `timestamp_value`: 초를 추출하고자 하는 timestamp

출력

- Integer

예시

- `second(last_login)`

millisecond

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에서 밀리초 (1/1000 초)에 해당하는 값을 반환합니다.

함수 형식

- `millisecond(timestamp_value)`

입력값

- `timestamp_value`: 밀리초를 추출하고자 하는 timestamp

출력

- Integer

예시

- millisecond(last_login)

now

카테고리

- Timestamp Function

설명

- 입력된 Timezone 기준의 현재 시간을 반환합니다.

함수 형식

- now()
- now(timezone)

입력값

- timzone: 현재시간을 구하고자 하는 Timezone의 full-name.

출력

- Integer

예시

- now()
- now('Asia/Seoul')

기타 사항

- Timezone 값을 입력하지 않을 시 UTC 기준의 시간 반환.

add_time

카테고리

- Timestamp Function

설명

- 입력된 Timestamp 값에 일정 Time unit 값을 더하거나 뺀 값을 반환합니다.

함수 형식

- `add_time(timestamp, delta, time_unit)`

입력값

- `timestamp`: 대상이되는 원본 timestamp 값
- `delta`: 더하거나 빼고자 하는 날짜/시간 값
- `time_unit`: 더하거나 빼고자 하는 날짜/시간의 단위 (문자열로 입력). `year`, `month`, `day`, `hour`, `minute`, `second`, `millisecond`.

출력

- Integer

예시

- `add_time(end_date, 10, 'day')`
- `add_time(end_date, -1, 'month')`

sum

카테고리

- Aggregation Function

설명

- 대상 값들의 합을 반환합니다.

함수 형식

- `sum(target_col)`

입력값

- `target_col`: 합을 구하고자 하는 대상 컬럼

출력

- Double

예시

- `sum(profit)`

기타 사항

- Aggregation과 Window 룰에서만 사용 가능.

avg

카테고리

- Aggregation Function

설명

- 대상 값들의 평균을 반환합니다.

함수 형식

- avg(target_col)

입력값

- target_col: 평균을 구하고자 하는 대상 컬럼

출력

- Double

예시

- avg(profit)

기타 사항

- Aggregation과 Window 룰에서만 사용 가능.

max

카테고리

- Aggregation Function

설명

- 대상 값들 중 가장 큰 값을 반환합니다.

함수 형식

- max(target_col)

입력값

- target_col: 최대값을 구하고자 하는 대상 컬럼

출력

- Double

예시

- max(profit)

기타 사항

- Aggregation과 Window 룰에서만 사용 가능.

min

카테고리

- Aggregation Function

설명

- 대상 값들 중 가장 작은 값을 반환합니다.

함수 형식

- min(target_col)

입력값

- target_col: 최소값을 구하고자 하는 대상 컬럼

출력

- Double

예시

- min(profit)

기타 사항

- Aggregation과 Window 룰에서만 사용 가능.

count

카테고리

- Aggregation Function

설명

- 대상의 줄 (row) 수를 반환합니다.

함수 형식

- count()

출력

- Double

예시

- count()

기타 사항

- Aggregation과 Window 룰에서만 사용 가능.

math.abs

카테고리

- Math Function

설명

- 입력된 값의 절대값을 반환합니다.

함수 형식

- math.abs(value)

입력값

- value: 절대값을 구하고자 하는 숫자.

출력

- Double

예시

- math.abs(-10) : 10

math.acos

카테고리

- Math Function

설명

- 입력된 값의 아크코사인 값을 반환합니다.

함수 형식

- `math.acos(value)`

입력값

- `value`: 아크코사인 값을 구하고자 하는 코사인값으로 -1에서 1 사이의 값.

출력

- Double

예시

- `math.acos(-1) : 3.141592653589793`

math.asin

카테고리

- Math Function

설명

- 입력된 값의 아크사인 값을 반환합니다.

함수 형식

- `math.asin(value)`

입력값

- `value`: 아크사인 값을 구하고자 하는 사인값으로 -1에서 1 사이의 값.

출력

- Double

예시

- `math.asin(-1)` : -1.5707963267948966

math.atan

카테고리

- Math Function

설명

- 입력된 값의 아크사인 값을 반환합니다.

함수 형식

- `math.atan(value)`

입력값

- `value`: 아크사인 값을 구하고자 하는 사인값으로 -1에서 1사이의 값.

출력

- Double

예시

- `math.asin(-1)` : -1.5707963267948966

math.cbrt

카테고리

- Math Function

설명

- 입력된 값의 세제곱근 값을 반환합니다.

함수 형식

- `math.cbrt(value)`

입력값

- `value`: 세제곱근 값을 구하고자 하는 숫자.

출력

- Double

예시

- `math.cbrt(5)` : 1.709975946676697

`math.ceil`

카테고리

- Math Function

설명

- 입력된 값을 일의 배수가 되도록 올림한 값을 반환합니다.

함수 형식

- `math.ceil(value)`

입력값

- `value`: 일의 자리로 올림 하고자 하는 숫자.

출력

- Double

예시

- `math.ceil(15.142)` : 16

`math.cos`

카테고리

- Math Function

설명

- 입력된 값의 코사인 값을 반환합니다.

함수 형식

- `math.cos(value)`

입력값

- value: 코사인 값을 구하고자 하는 라디안 각도

출력

- Double

예시

- `math.cos(45)` : 0.5253219888177297

math.cosh

카테고리

- Math Function

설명

- 입력된 값의 하이퍼볼릭 코사인 값을 반환합니다.

함수 형식

- `math.cosh(value)`

입력값

- value: 하이퍼볼릭 코사인 값을 구하고자 하는 숫자.

출력

- Double

예시

- `math.cosh(9)` : `COSH(9)` => 4051.5420254925943

math.exp

카테고리

- Math Function

설명

- 자연 로그값 e를 입력된 값만큼 거듭제곱한 값을 반환합니다.

함수 형식

- `math.exp(value)`

입력값

- `value`: 자연 로그값 e 를 거듭제곱 하고자 하는 횟수.

출력

- Double

예시

- `math.exp(4)` : 54.598150033144236

`math.expm1`

카테고리

- Math Function

설명

- 자연 로그값 e 를 입력된 값만큼 거듭제곱한 값에서 1을 뺀 값을 반환합니다.

함수 형식

- `math.expm1 (value)`

입력값

- `value`: 자연 로그값 e 를 거듭제곱 하고자 하는 횟수.

출력

- Double

예시

- `math.expm1 (4)` : 53.598150033144236

`math.getExponent`

카테고리

- Math Function

설명

- 입력된 값 N에 대하여 $2^{\text{exp}} \leq N$ 을 만족하는 exp 값 중 가장 큰 값을 반환합니다.

함수 형식

- `math.getExponent(value)`

입력값

- value: $2^{\text{exp}} \leq N$ 를 만족하는 exp 값을 찾을 때 N에 해당하는 숫자.

출력

- Double

예시

- `math.getExponent(9) : 3`

math.round

카테고리

- Math Function

설명

- 입력된 값을 일의 자리로 반올림 한 값을 반환합니다.

함수 형식

- `math.round(value)`

입력값

- value: 일의 자리로 반올림 하고자 하는 숫자

출력

- Double

예시

- `math.round(14.2) : 14`

math.signum

카테고리

- Math Function

설명

- 입력된 값의 부호를 반환합니다.

함수 형식

- `math.signum(value)`

입력값

- `value`: 부호를 추출하고자 하는 숫자

출력

- Double

예시

- `math.signum(-24) : -1`

기타 사항

- 입력된 숫자가 양수인 경우 1, 0인 경우 0, 음수인 경우 -1을 반환한다.

math.sin

카테고리

- Math Function

설명

- 입력된 값의 사인 값을 반환합니다.

함수 형식

- `math.sin(value)`

입력값

- `value`: 사인 값을 구하고자 하는 라디안 각도

출력

- Double

예시

- `math.sin(90)` : 0.8939966636005579

math.sinh

카테고리

- Math Function

설명

- 입력된 값의 하이퍼볼릭 사인 값을 반환합니다.

함수 형식

- `math.sinh(value)`

입력값

- `value`: 하이퍼볼릭 사인 값을 구하고자 하는 숫자

출력

- Double

예시

- `math.sinh(1)` : 1.1752011936438014

math.sqrt

카테고리

- Math Function

설명

- 입력된 값의 제곱근을 반환합니다.

함수 형식

- `math.sqrt(value)`

입력값

- `value`: 제곱근 값을 구하고자 하는 숫자

출력

- Double

예시

- `math.sqrt(4) : 2`

`math.tan`

카테고리

- Math Function

설명

- 입력된 값의 탄젠트 값을 반환합니다.

함수 형식

- `math.tan(value)`

입력값

- value: 탄젠트 값을 구하고자 하는 라디안 각도

출력

- Double

예시

- `math.tan(10) : 0.6483608274590866`

`math.tanh`

카테고리

- Math Function

설명

- 입력된 값의 하이퍼볼릭 탄젠트 값을 반환합니다.

함수 형식

- `math.tanh(value)`

입력값

- value: 하이퍼볼릭 탄젠트 값을 구하고자 하는 각도.

출력

- Double

예시

- `math.tanh(4)` : 0.999329299739067

time_diff

카테고리

- Timestamp Function

설명

- 입력된 두 Timestamp 값의 차를 millisecond 단위로 계산하여 반환합니다.

함수 형식

- `time_diff(timestamp1, timestamp2)`

입력값

- timestamp1: $C = B - A$ 에서 A에 해당하는 시간 값.
- timestamp2: $C = B - A$ 에서 B에 해당하는 시간 값.

출력

- Double

예시

- `time_diff(order_date, shipped_date)`

기타 사항

- 결과 값 = `timestamp2 - timestamp1`

timestamp

카테고리

- Timestamp Function

설명

- 새로운 Timestamp 값을 생성합니다.

함수 형식

- timestamp(value, format)

입력값

- value: timestamp 값으로 생성하고자 하는 날짜/시간 값.
- format: value 값의 시간 형식.

출력

- Timestamp

예시

- timestamp('2011-01-01', 'yyyy-MM-dd') : 2011-01-01T00:00:00.000Z

row_number

카테고리

- Window Function

설명

- Partition 내에서 Order 순으로 정렬한 Row의 일련번호를 생성합니다.

함수 형식

- row_number()

출력

- Long

예시

- row_number()

기타 사항

- Window Rule에서만 사용 가능.

rolling_sum

카테고리

- Window Function

설명

- Partition 내에서 앞/뒤의 지정한 수의 Row의 값들의 합을 반환합니다.

함수 형식

- `rolling_sum(target_col, before, after)`

입력값

- `target_col`: 합을 구하고자 하는 대상 컬럼 명.
- `before`: 합산하고자 하는 선행 row의 수.
- `after`: 합산하고자 하는 후행 row의 수.

출력

- Long/Double

예시

- `rolling_sum(profit, 3, 3)` : 같은 partition 내의 앞뒤 3개 row를 포함해 총 7개 row의 profit을 합산.

기타 사항

- Window Rule에서만 사용 가능.

rolling_avg

카테고리

- Window Function

설명

- Partition 내에서 앞/뒤의 지정한 수의 Row의 값들의 평균값을 반환합니다.

함수 형식

- `rolling_avg(target_col, before, after)`

입력값

- target_col: 평균을 구하고자 하는 대상 컬럼 명.
- before: 평균을 구하고자 하는 선행 row의 수.
- after: 평균을 구하고자 하는 후행 row의 수.

출력

- Long/Double

예시

- rolling_avg(profit, 3, 3) : 같은 partition 내의 앞뒤 3개 row를 포함해 총 7개 row의 profit의 평균.

기타 사항

- Window Rule에서만 사용 가능.

lag

카테고리

- Window Function

설명

- Partition 내에서 지정한 수 만큼 앞선 Row의 값을 반환합니다.

함수 형식

- lag(target_col, before)

입력값

- target_col: 대상 컬럼 명.
- before: 현재 row에서 얼마큼 앞선 row를 반환할지 지정하는 수.

출력

- Long/Double

예시

- lag(profit, 2) : 같은 partition 내 2 줄 위의 row의 profit 값을 반환합니다. 2 줄 위의 값이 없다면 null을 반환합니다.

기타 사항

- Window Rule에서만 사용 가능.

lead

카테고리

- Window Function

설명

- Partition 내에서 지정한 수 만큼 뒤에 있는 Row의 값을 반환합니다.

함수 형식

- `lead(target_col, after)`

입력값

- `target_col`: 대상 컬럼 명.
- `after`: 현재 row에서 얼마큼 뒤에 있는 row를 반환할지 지정하는 수.

출력

- Long/Double

예시

- `lead(profit, 2)` : 같은 partition 내 2 줄 아래의 row의 profit 값을 반환합니다. 2 줄 아래의 값이 없다면 null을 반환합니다.

기타 사항

- Window Rule에서만 사용 가능.

ismismatched

카테고리

- Logical Function

설명

- 지정한 컬럼의 Value가 특정 Column Type과 일치하는지 여부를 반환합니다.

함수 형식

- `ismismatched(target_col, column_type)`

입력값

- `target_col`: type을 검사하고자 하는 컬럼 명.
- `column_type`: 일치 여부를 확인하고자 하는 Type. (문자열로 입력) String, Boolean, Timestamp, Long, Double

출력

- Boolean

예시

- `ismismatched(birth_date, timestamp)` : 해당 row의 값이 timestamp인 경우엔 false, 아닌 경우엔 true.

contains

카테고리

- String Function

설명

- 지정한 컬럼의 Value가 특정 문자열을 포함하는지 여부를 반환합니다.

함수 형식

- `contains(target_col, search_word)`

입력값

- `target_col`: 문자열을 검색하고자 하는 컬럼 명.
- `search_word`: 컬럼에서 찾고자 하는 문자열.

출력

- Boolean

예시

- `contains(name, <son>)` : name에 son이 들어가는 경우 True. <Micheal Jackson>, <Son Heung Min> 등

startswith

카테고리

- String Function

설명

- 지정한 컬럼의 Value가 특정 문자열로 시작하는지 여부를 반환합니다.

함수 형식

- startswith(target_col, search_word)

입력값

- target_col: 문자열을 검색하고자 하는 컬럼 명.
- search_word: 컬럼에서 찾고자 하는 문자열.

출력

- Boolean

예시

- startswith(name, <김>) : name이 <김>으로 시작하는 경우 True. <김철수>, <김수지> 등

endswith

카테고리

- String Function

설명

- 지정한 컬럼의 Value가 특정 문자열을 끝나는지 여부를 반환합니다.

함수 형식

- endswith(target_col, search_word)

입력값

- target_col: 문자열을 검색하고자 하는 컬럼 명.
- search_word: 컬럼에서 찾고자 하는 문자열.

출력

- Boolean

예시

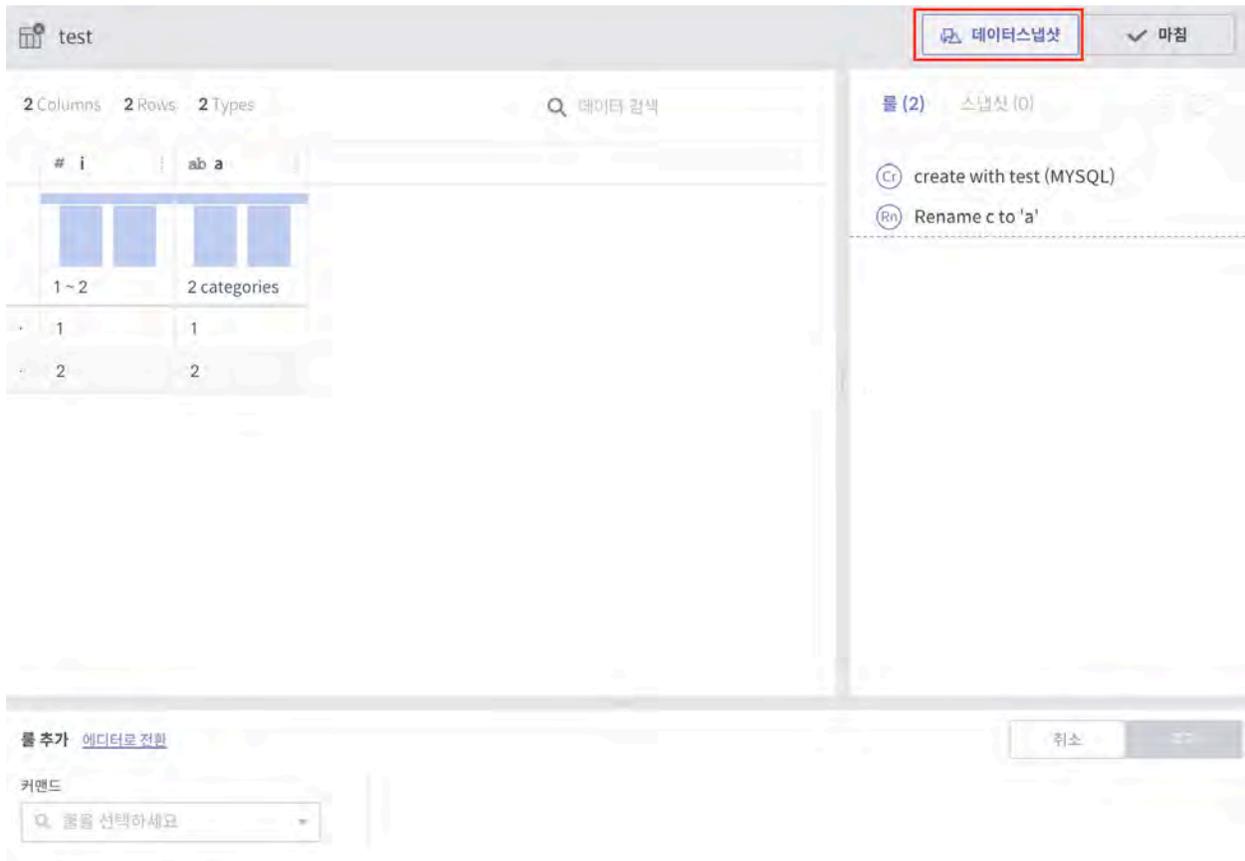
- `endswith(customer_code, <M>)` : `customer_code`가 M으로 끝나는 경우 True. <1340M>, <0020M> 등

8.4.5 데이터 스냅샷 만들기

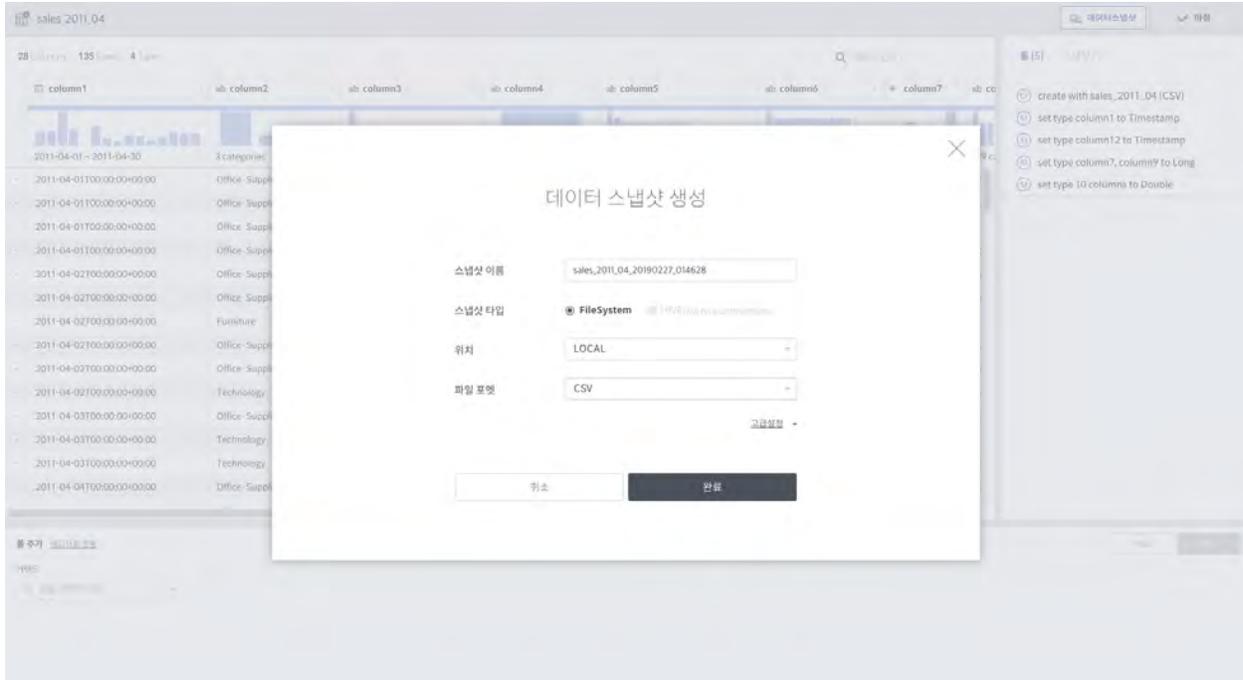
데이터플로우에서 룰 편집이 완료된 데이터셋은 <데이터 스냅샷>을 찍어서 로컬 PC에 파일로 다운로드 받거나, Metatron 엔진으로 즉시 적재할 수 있습니다. 데이터 스냅샷을 실행하면 10,000개 행 이하의 샘플 데이터에만 적용된 룰을 전체 데이터에 대해 적용하게 됩니다.

스냅샷을 만드는 방법은 다음과 같습니다.

1. 룰 편집 창 우측 상단에 위치한 **데이터 스냅샷** 버튼을 클릭합니다.



2. 스냅샷 생성을 위한 팝업이 나타나면, 스냅샷 생성 위치를 FILE과 HIVE (STAGING_DB) 중에서 지정합니다.



- FILE 경로로 스냅샷의 위치를 지정하는 경우, CSV 또는 JSON의 포맷으로 생성할 수 있습니다.



- HIVE 옵션은 STAGING_DB가 설정된 경우에만 활성화됩니다. 스키마명과 테이블명을 지정하면 해당 테이블에 스냅샷이 생성됩니다.

데이터 스냅샷 생성

스냅샷 이름: jsonTest_null_20190227_091201

스냅샷 타입: FileSystem HIVE

DB 이름: cazen_lee

테이블 이름: snapshot1

고급설정 ▾

취소 완료

3. 스냅샷을 생성하면 동일한 창에서 스냅샷 생성 상태와 정보를 조회할 수 있습니다.

 데이터스냅샷  마침

물 (5) : 스냅샷 (2)

Success

  sales_2011_04_20190227_0146_28 

2019-02-27 10:47:09

Success

  sales_2011_04_20190218_0759_21 

2019-02-18 16:59:26

 스냅샷 목록으로 이동

8.5 데이터 스냅샷 결과 이용하기

데이터플로우를 통해 생성된 데이터 스냅샷은 다음과 같이 활용할 수 있습니다.

- 데이터 스냅샷 결과 확인하기
- Metatron 엔진으로 적재하기
- CSV파일로 다운로드 받기

8.5.1 데이터 스냅샷 결과 확인하기

스냅샷의 생성 단계는 다음과 같은 3가지 경우로 분류됩니다.

- 성공 = SUCCEEDED
- 실패 = FAILED
- 처리중 = INITIALIZING, RUNNING, WRITING, TABLE_CREATING, CANCELING

스냅샷의 상세한 처리 결과는 다음 2가지 경로를 통해 확인할 수 있습니다.

- MANAGEMENT > 데이터 프리퍼레이션 > 데이터 스냅샷의 스냅샷 목록을 통해서 확인

≡ METATRONDISCOVERY

데이터 프리퍼레이션

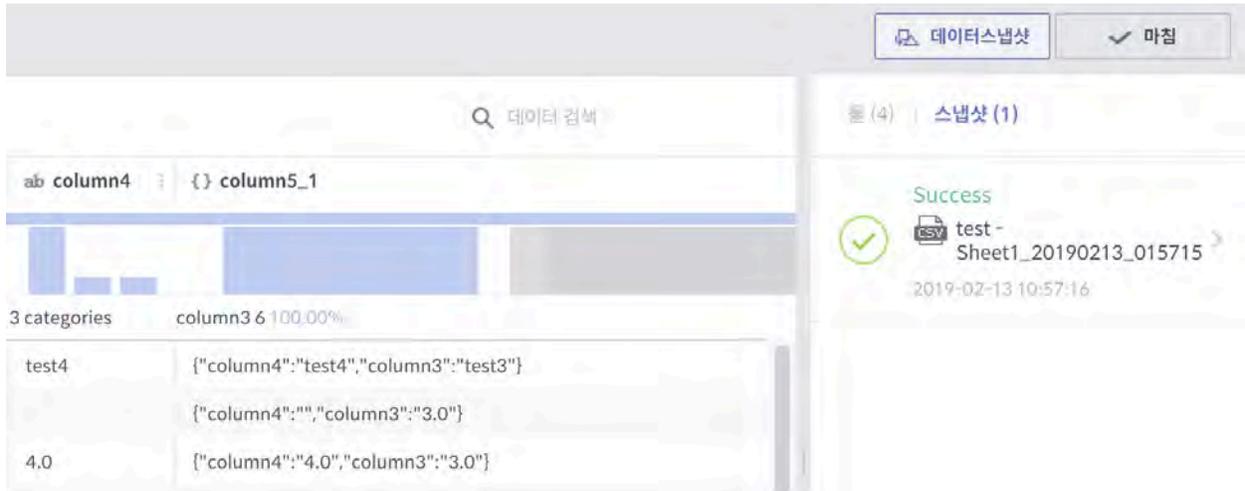
데이터셋 데이터플로우 **데이터 스냅샷**

스냅샷 타입 All 상태 전체 성공 실패 처리중

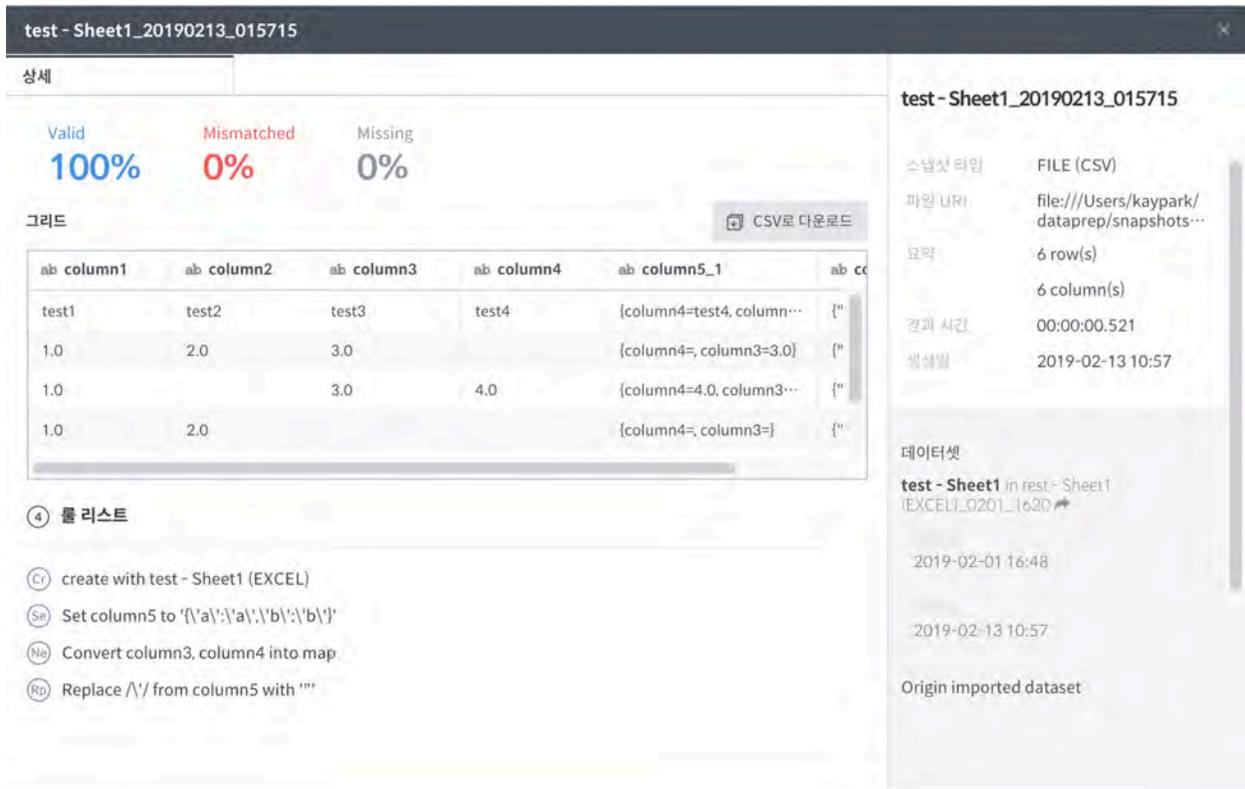
8개 데이터가 있습니다

이름	데이터플로우 데이터셋	스냅샷 타입	상태	경과 시간	생성일
test - Sheet1_20190213_015715	test - Sheet1 (EXCEL)_0201_1620 t...	FILE (CSV)	✓	00:00:00.521	2019-02-13 10...
test - Sheet1_20190201_073721	test - Sheet1 (EXCEL)_0201_1620 t...	FILE (CSV)	✗	00:00:00.136	2019-02-01 16...
test - Sheet1_20190201_073712	test - Sheet1 (EXCEL)_0201_1620 t...	FILE (JSON)	✗	00:00:00.204	2019-02-01 16...
crunchbase_monthly_e Round...	crunchbase_monthly_e Rounds (EX...	FILE (CSV)	✓	00:00:02.159	2019-01-29 16...

- 데이터플로우의 룰 편집 화면 우측에서 스냅샷 (#) 탭을 클릭



생성에 성공한 스냅샷의 상세 보기 화면으로 들어가면, 데이터의 유효성 비율, 생성된 스냅샷의 그리드 등을 확인할 수 있고, 스냅샷 결과를 CSV 파일로 다운로드할 수 있습니다.



유효한 데이터가 생성되지 못한 스냅샷의 상세 보기 화면으로 들어가면, 실패를 발생시킨 예외의 로그가 표시됩니다.

The screenshot displays the Metatron interface for a failed data snapshot. The main window is titled "test - Sheet1_20190201_073721".

에러 로그 (Error Log):

```

app.metatron.discovery.domain.dataprep.teddy.TeddyExecutor.createUriSnapshot(TeddyExecutor.java:330)
app.metatron.discovery.domain.dataprep.teddy.TeddyExecutor.run(TeddyExecutor.java:162)
app.metatron.discovery.domain.dataprep.teddy.TeddyExecutor$$FastClassBySpringCGLIB$$8a9fff2b.invoke()
org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:204)
org.springframework.aop.framework.CglibAopProxy$$CglibMethodInvocation.invokeJoinpoint(CglibAopProxy.java:738)
)
org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:157)
org.springframework.aop.interceptor.AsyncExecutionInterceptor$1.call(AsyncExecutionInterceptor.java:115)
java.util.concurrent.FutureTask.run(FutureTask.java:266)
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
java.lang.Thread.run(Thread.java:745)

```

물 리스트 (List):

- create with test - Sheet1 (EXCEL)
- Set column5 to '[a:\'a\',b:\'b\']'

test - Sheet1_20190201_073721 (Metadata):

- 스냅샷 타입: FILE (CSV)
- 파일 URI: file:///Users/kaypark/dataprep/snapshots...
- 결과 시간: 00:00:00.136
- 생성일: 2019-02-01 16:37

데이터셋 (Dataset):

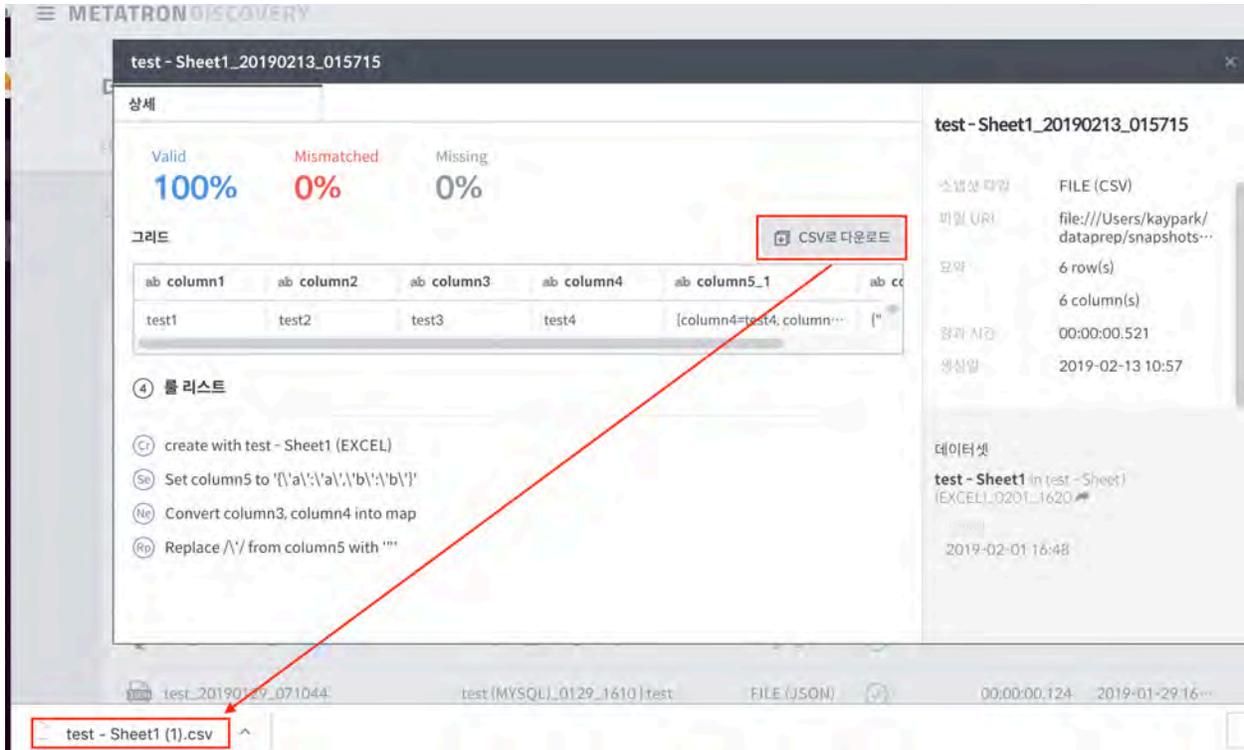
- test - Sheet1 in test - Sheet1 (EXCEL)_0201_1620
- 2019-02-01 16:20
- 2019-02-01 16:21
- Origin imported dataset
- test - Sheet1 (EXCEL)

8.5.2 Metatron 엔진으로 적재하기

(개발예정)

8.5.3 CSV파일로 다운로드 받기

생성에 성공한 스냅샷의 상세 보기에서는 CSV로 다운로드가 가능합니다.



다운로드한 파일은 표준 CSV 형식으로, <comma>로 구분되고 <new line>으로 개행합니다.

시트 1

test - Sheet1 (1)

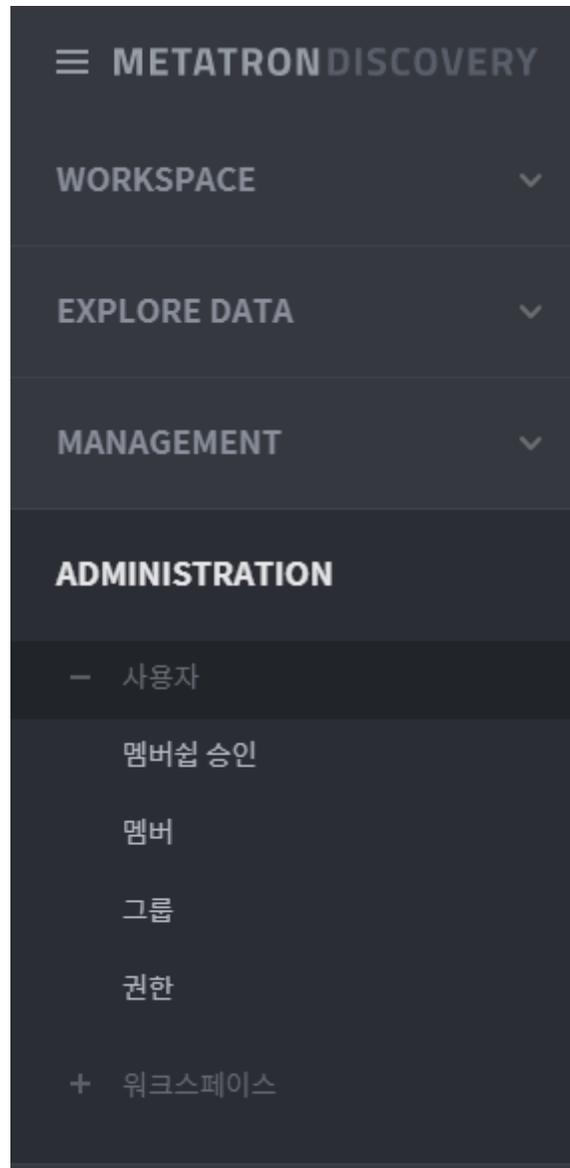
column1	column2	column3	column4	column5_1	column5
test1	test2	test3	test4	{column4=test4, column3=test3}	{"a":"a","b":"b"}
1	2	3		{column4=, column3=3.0}	{"a":"a","b":"b"}
1		3	4	{column4=4.0, column3=3.0}	{"a":"a","b":"b"}
1	2			{column4=, column3=}	{"a":"a","b":"b"}
		3		{column4=, column3=3.0}	{"a":"a","b":"b"}

CHAPTER 9

계정 관리

관리자는 Metatron Discovery 사용자들의 멤버십과 사용 권한을 설정·관리할 수 있으며, 그룹 기능을 이용하여 이러한 관리를 더욱 효율화할 수 있습니다.

사용자 관리를 위해서는 메인 화면 좌측 패널에서 ADMINISTRATION → 사용자 클릭 후 원하는 하위 메뉴를 선택하면 됩니다.



9.1 멤버십 승인

이 메뉴에서는 사용자들의 가입 신청 내역을 보여줍니다. 아래 화면에서 보이는 것과 같이 가입이 거절되었거나, 승인을 대기 중인 신청 건들이 목록에 나타납니다. 승인을 마친 사용자는 이 메뉴가 아닌 **멤버** 메뉴에서 확인할 수 있습니다.

사용자

[멤버십 승인](#)
[멤버](#)
[그룹](#)
[권한](#)

상태 전체 대기 거부 🔄 초기화

요청 날짜 ~

🔍 사용자 이름으로 검색해 주세요 16개 데이터가 있습니다

사용자 이름	이름	이메일	요청 날짜	상태
applicant	EE	applicant24@gmail.com	2019-08-21 22:39	<input checked="" type="button" value="승인"/> <input type="button" value="거절"/>
tester_00	tester_00	skt.metatron@gmail.com	2019-05-14 13:08	거부됨 <input type="button" value="🔍"/>
tester_00	tester_00	skt.metatron@gmail.com	2019-05-14 12:53	거부됨 <input type="button" value="🔍"/>
tester_00	tester_00	skt.metatron@gmail.com	2019-05-14 11:37	거부됨 <input type="button" value="🔍"/>
sehwa.lee	sehwa.lee	sehwa.lee@sk.com	2019-05-09 18:43	거부됨 <input type="button" value="🔍"/>
admin_test	aaa	kyungtaak@gamil.com	2019-04-29 10:13	거부됨 <input type="button" value="🔍"/>
asd	ASD	asd@asd.com	2018-12-10 13:12	거부됨 <input type="button" value="🔍"/>
sbparks	sbparks	sbparks@sbparks.sbparks	2018-12-06 13:23	거부됨 <input type="button" value="🔍"/>
tester	Tester	test@test.com	2018-11-23 13:11	거부됨 <input type="button" value="🔍"/>
pp333	pp333	ppeee@test.com	2018-11-22 14:51	거부됨 <input type="button" value="🔍"/>
pp333	pp333	pp333@pp333.pp333	2018-11-15 15:52	거부됨 <input type="button" value="🔍"/>
pp222	pp222	pp222@pp222.pp222	2018-11-15 15:51	거부됨 <input type="button" value="🔍"/>
pp111	pp111	pp111@pp111.pp111	2018-11-15 15:51	거부됨 <input type="button" value="🔍"/>
pp000	pp000	pp000@pp000.pp000	2018-11-15 15:51	거부됨 <input type="button" value="🔍"/>
p888	p888	p888@p888.p888	2018-11-15 15:50	거부됨 <input type="button" value="🔍"/>
ppp3333	ppp3333	ppp3333@cc.com	2018-11-15 15:02	거부됨 <input type="button" value="🔍"/>

한 페이지당 목록 20

9.2 멤버

이 메뉴에서는 등록을 마친 사용자들을 열람·관리할 수 있습니다.

Metatron Discovery 가입은 다음 두 방법 중 하나로 이루어집니다.

- 사용자가 회원 가입 신청 후 관리자가 승인 ([멤버십 승인 참조](#))
- 관리자가 직접 등록 ([사용자 등록 참조](#))

9.2.1 멤버 홈 화면

멤버 홈 화면에서는 Metatron Discovery 시스템에 가입된 사용자들을 열거하여 보여줍니다. 사용자 목록은 여러 기준으로 필터링할 수 있으며, 목록에 나타난 사용자 중 하나를 클릭하여 해당 사용자의 정보를 열람·수정할 수 있습니다.

사용자

멤버십 승인 **멤버** 그룹 권한

상태 전체 활성화 비활성화 초기화

가입 날짜 **전체** 오늘 이전 7일 yyyy-MM-dd hh:mm ~ yyyy-MM-dd hh:mm 적용

🔍 사용자 이름으로 검색해 주세요 41개 데이터가 있습니다 사용자 생성

멤버 (이름)	사용자 이름	이메일	가입 날짜	상태
admin	admin	metatron.app@gmail.com	2018-08-24 15:49	활성화
Guest	guest	guest@metatron.com	2018-08-24 15:49	활성화
Polaris	polaris	polaris@metatron.com	2018-08-24 15:49	활성화
Metatron	metatron	metatron@metatron.com	2018-08-24 15:49	활성화
skt_geo_demo	skt_geo_demo	june.woo.lee@sk.com	2018-10-10 14:52	활성화
dskim	qatester	qa@test.er	2018-11-15 09:16	활성화
p333	p333	p333@p333.p333	2018-11-15 15:48	활성화
p444	p444	p444@p444.p444	2018-11-15 15:48	활성화
p666	p666	p666@p666.p666	2018-11-15 15:49	활성화
p777	p777	p777@p777.p777	2018-11-15 15:49	활성화
p999	p999	p999@p999.p999	2018-11-15 15:50	활성화
p000	p000	p000@p000.p000	2018-11-15 15:50	활성화
a111	a111	a111@a111.a111	2018-11-19 13:52	활성화
a222	a222	a222@a222.a222	2018-11-19 13:52	활성화
Tester	tester	test@test.com	2018-11-23 13:13	활성화
p1234	p1234	p1234@p1234.p1234	2018-12-06 11:33	활성화
12dectestcho	12dectestcho	12dectestcho@magenta.works	2018-12-06 14:25	활성화
김연임	deidera	deidera@magenta.works	2018-12-07 13:58	활성화

9.2.2 사용자 정보 열람 및 수정

홈 화면 목록에 나타난 사용자 중 하나를 클릭하면, 아래와 같이 해당 사용자의 정보를 보여주는 화면으로 이동합니다.

admin

가입 날짜 2018-08-24 15:49

상태 **활성화**

정보

	이름	admin
	사용자 이름	admin
	이메일	metatron.app@gmail.com
	권한	시스템 관리, 데이터 관리 권한과 모니터링, 공유 워크스페이스 사용, 개인 워크스페이스 사용, 워크스페이스 커스텀 스킴 관리
	전화번호	0000000000

그룹 (1)

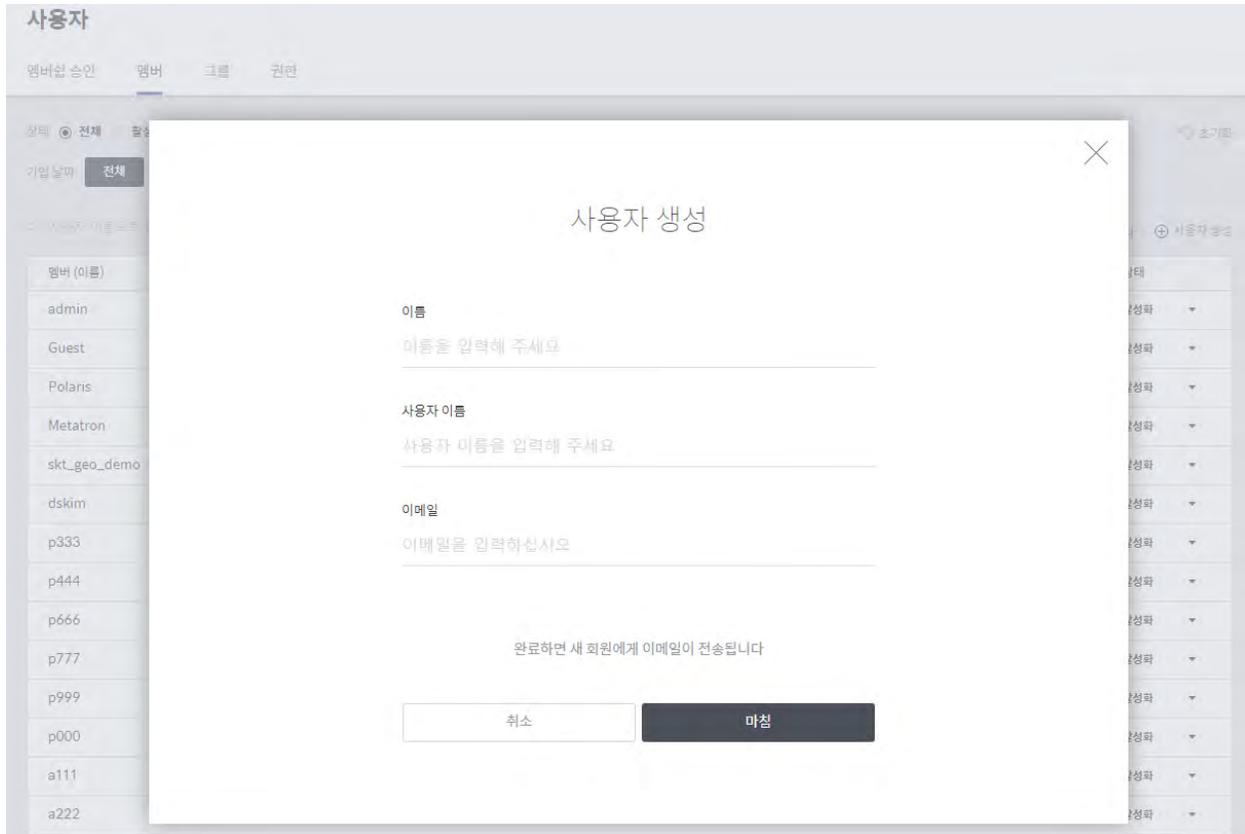
System-Admin	시스템 관리, 데이터 관리 권한과 모니터링, 공유 워크스페이스 사용, 개인 워크스페이스 사용, 워크스페이스 커스텀 스킴 관리
--------------	---

이 화면에서는 기본 정보 열람과 더불어 몇 가지 설정이 가능합니다.

- **상태 설정 (활성화/비활성화):** 해당 사용자를 비활성화시키면 시스템에 로그인할 수 없게 됩니다.
- **패스워드 초기화:** 해당 사용자가 비밀번호를 잊어버린 경우, 비밀번호를 초기화할 수 있는 메일을 보내줍니다.
- **그룹 설정:**  아이콘을 클릭하면 해당 사용자가 속한 그룹을 추가·삭제할 수 있습니다. 그룹과 관련된 설명은 [그룹 항목](#)을 참조하십시오.

9.2.3 사용자 등록

홈 화면 우측 상단에서 사용자 생성 버튼을 클릭하면 아래와 같은 사용자 생성 팝업이 나타납니다.



사용자의 실명과 ID, 이메일을 입력하면 사용자 등록이 완료되며, 해당 메일 주소로 가입 내역이 전달됩니다.

9.3 그룹

Metatron Discovery 사용자들을 그룹으로 지정하면 다음과 같은 편의 기능을 사용할 수 있습니다.

- 그룹에 속한 사용자들의 권한 일괄 설정
- 그룹에 속한 사용자들에게 일괄 메일 전송

9.3.1 그룹 홈 화면

그룹 홈 화면에는 현재 Metatron Discovery에 등록된 사용자 그룹들을 보여줍니다. 그룹 목록은 여러 기준으로 필터링할 수 있으며, 목록에 나타난 그룹 중 하나를 클릭하여 해당 그룹의 정보를 열람·수정할 수 있습니다.

사용자

[멤버십 승인](#)
[멤버](#)
[그룹](#)
[권한](#)

생성 날짜 **전체** **오늘** **이전 7일** ~
초기화

사용자 이름으로 검색하세요 30개 데이터가 있습니다

그룹 #	설명	멤버 #	생성 날짜	
Data-Manager		12	2018-08-24 15:49	
General-User		60	2018-08-24 15:49	
System-Admin		9	2018-08-24 15:49	
#1425		0	2019-03-07 10:42	<input type="button" value="삭제"/>
11222	1122222	0	2018-11-15 15:46	<input type="button" value="삭제"/>
14	14	0	2018-11-15 15:46	<input type="button" value="삭제"/>
1414.14142		0	2019-03-07 10:46	<input type="button" value="삭제"/>
15	15	0	2018-11-15 15:46	<input type="button" value="삭제"/>
16	16	0	2018-11-15 15:46	<input type="button" value="삭제"/>
17	17	0	2018-11-15 15:46	<input type="button" value="삭제"/>
18	18	0	2018-11-15 15:46	<input type="button" value="삭제"/>
19	19	0	2018-11-15 15:47	<input type="button" value="삭제"/>
2	2	0	2018-11-15 15:44	<input type="button" value="삭제"/>
20	20	1	2018-11-15 15:47	<input type="button" value="삭제"/>
21	21	0	2018-11-15 15:47	<input type="button" value="삭제"/>
22	22	0	2018-11-15 15:47	<input type="button" value="삭제"/>
3	3	0	2018-11-15 15:46	<input type="button" value="삭제"/>
4	4	0	2018-11-15 15:46	<input type="button" value="삭제"/>

9.3.2 그룹 정보 열람 및 수정

홈 화면 목록에 나타난 그룹 중 하나를 클릭하면, 아래와 같이 해당 그룹의 정보를 보여주는 화면으로 이동합니다.

← Data-Manager

생성 날짜 2018-08-24 15:49 by **admin**
 마지막 수정 날짜 2019-06-12 17:04 by **admin**

정보

이름	Data-Manager
설명	
권한	데이터 관리 권한과 모니터링, 공유 워크스페이스 사용, 개인 워크스페이스 사용

멤버(10)
 모든 멤버에게 이메일 보내기

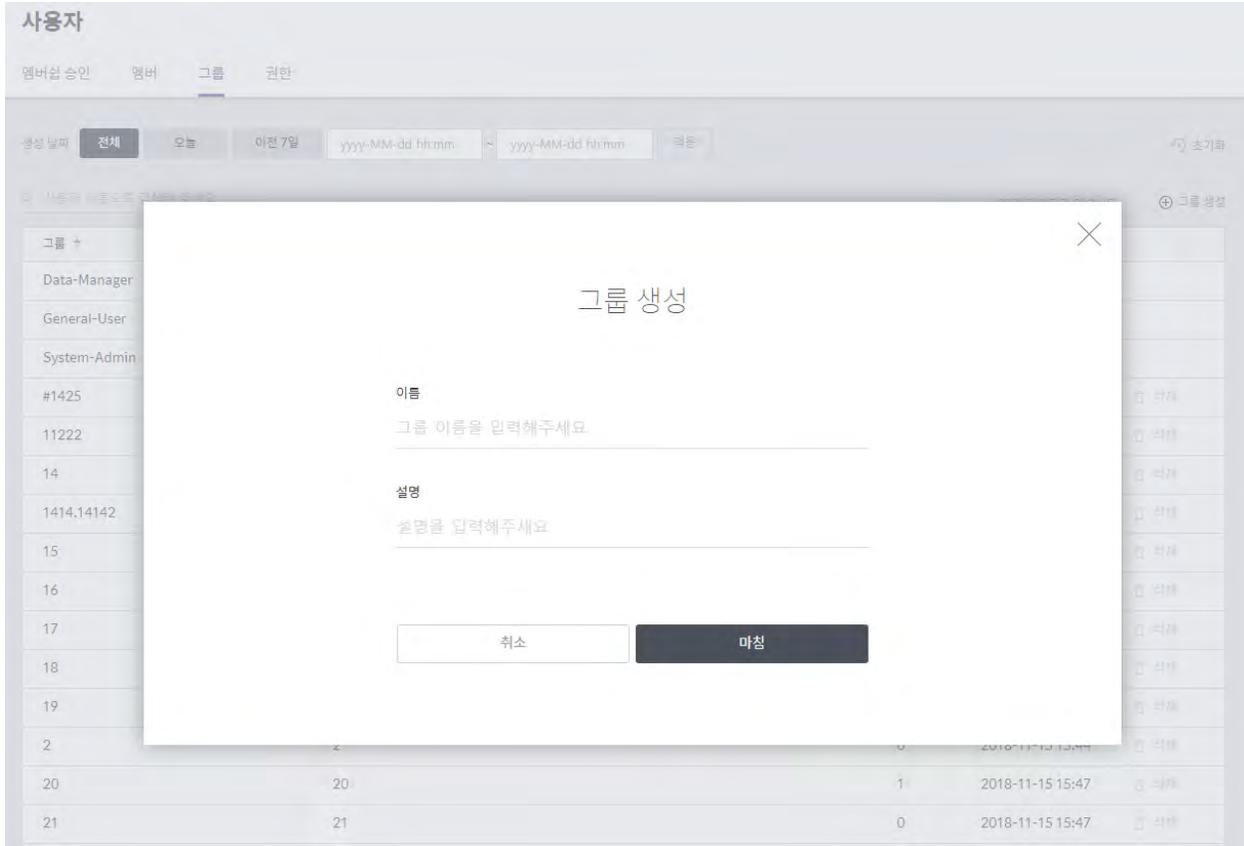
polaris
qatester
deidera
demo
heesoo
jungil.park
choong
sting
kyungtaak
SKH

이 화면에서 제공되는 기능은 다음과 같습니다.

- 해당 그룹의 기본 정보와 부여된 권한, 그리고 소속된 사용자들을 확인할 수 있습니다.
- 아이콘을 클릭하면 해당 그룹의 소속 멤버를 추가·삭제할 수 있습니다.
- 모든 멤버에게 이메일 보내기 버튼을 클릭하면 해당 그룹에 소속된 모든 멤버에게 이메일을 보낼 수 있습니다.

9.3.3 그룹 등록

홈 화면 우측 상단에서 그룹 생성 버튼을 클릭하면 아래와 같은 그룹 생성 팝업이 나타납니다.



이름과 설명을 입력한 후 마침 버튼을 클릭하면 새로운 그룹이 생성됩니다.

9.4 사용자 권한

Metatron Discovery는 아래 화면과 같이 총 4종류의 권한을 지원하여 사용자 계정별로 접근 권한에 차등을 둘 수 있습니다. 본 메뉴에서는 개별 사용자 또는 그룹에게 부여하는 권한을 설정할 수 있습니다.

사용자

멤버십 승인 멤버 그룹 권한

4개 데이터가 있습니다

권한	설명	멤버	그룹
데이터 관리 권한과 모니...	데이터 관리 메뉴로 접근할 수 있으며, 데이터생성하고 관리할 수 있습니다. 또한, 데이터의 사용을 모니터링할 수 있습니다	0	2
워크스페이스 커스텀 스...	사용자가 소유하고 있는 워크스페이스에 커스텀 스키마를 생성하고 관리할 수 있습니다	0	1
개인 워크스페이스 사용	자신만 접근할 수 있는 개인 워크스페이스가 있고 해당 워크스페이스의 관리 권한을 갖습니다	0	3
공유 워크스페이스 사용	새로운 공유 워크스페이스를 생성할 수 있으며, 본인이 소속된 공유 워크스페이스에 접근할 수 있습니다	1	3

홈 화면에 제시된 4개의 권한 중 하나를 클릭하면 아래와 같이 해당 권한이 부여된 개인 사용자와 그룹이 표시됩니다.

≡ METATRON DISCOVERY

← 데이터 관리 권한와 모니터링 데이터 관리 메뉴로 접근할 수 있으며, 데이터 생성하고 관리할 수 있습니다. 또한, 데이터의 사용을 모니터링할 수 ...

정보

이름	데이터 관리 권한와 모니터링
설명	데이터 관리 메뉴로 접근할 수 있으며, 데이터 생성하고 관리할 수 있습니다. 또한, 데이터의 사용을 모니터링할 수 있습니다

사용자

멤버 (0)	멤버가 없습니다 ⚙
그룹 (2)	▼ Data-Manager and 1 more groups. ⚙

멤버 또는 그룹 영역에서 ⚙ 아이콘을 클릭하면 아래와 같은 설정 팝업이 나타나서 해당 권한을 부여받는 멤버/그룹을 설정할 수 있습니다.

공유 멤버 및 그룹 설정

취소

마침

멤버 2

그룹 2

Q 사용자 이름으로 검색

- 전체 (15/41)
- #error (test)
- 12dectestcho (12dectestcho)**
- a111 (a111)
- a222 (a222)**
- admin (admin)
- al.lee (al.lee)
- choong (choong)
- DD (member)
- delete_user2 (delete_user2)
- delete_user3 (delete_user3)
- Demo (demo)
- dskim (qatester)
- eeee (eeee)
- Guest (guest)
- hive (hive)

2개 선택

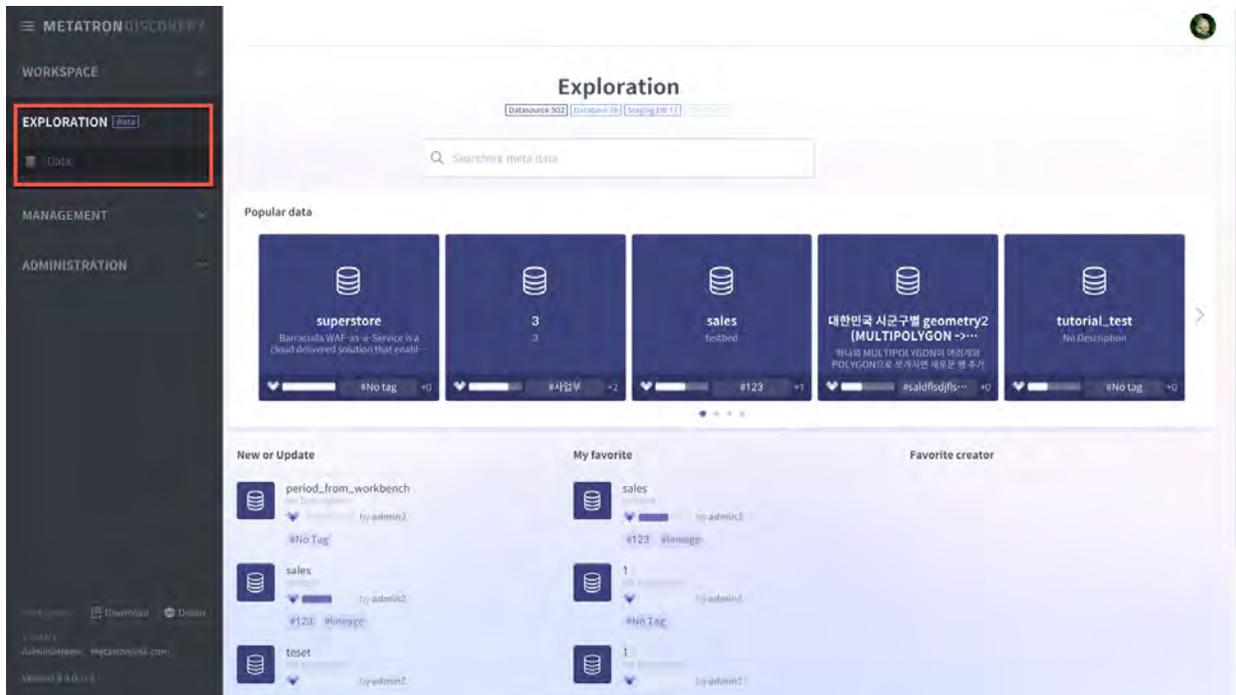
이름	사용자 이름	X
12dectestcho	12dectestcho	X
a222	a222	X

관리자는 Metatron Discovery 사용자들의 멤버십과 사용 권한을 설정·관리할 수 있으며, 그룹 기능을 이용하여 이러한 관리를 더욱 효율화할 수 있습니다.

데이터 탐색을 위해서는 메인 화면 좌측 패널에서 Exploration 클릭 후 원하는 하위 메뉴를 선택하면 됩니다. 또한 사용자의 원활한 데이터 탐색을 위해서 Admin은 Metadata를 관리하여야 합니다. Management > Exploration 클릭 후 원하는 하위 메뉴를 선택하면 됩니다.

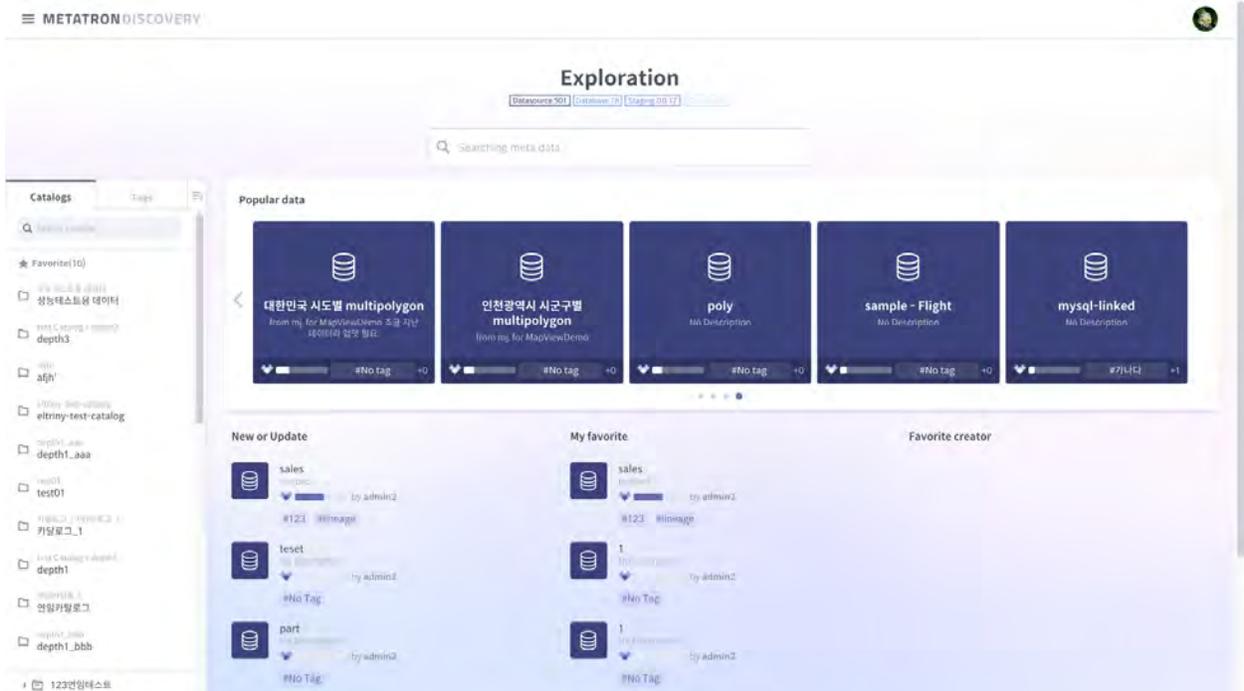
10.1 데이터 탐색하기

Metatron Discovery에서 제공하는 데이터 탐색의 목적은, 데이터가 어디 있던 쉽게 찾을 수 있고, 찾은 데이터로 시각화할 수 있는 기능을 지원하고 있습니다.



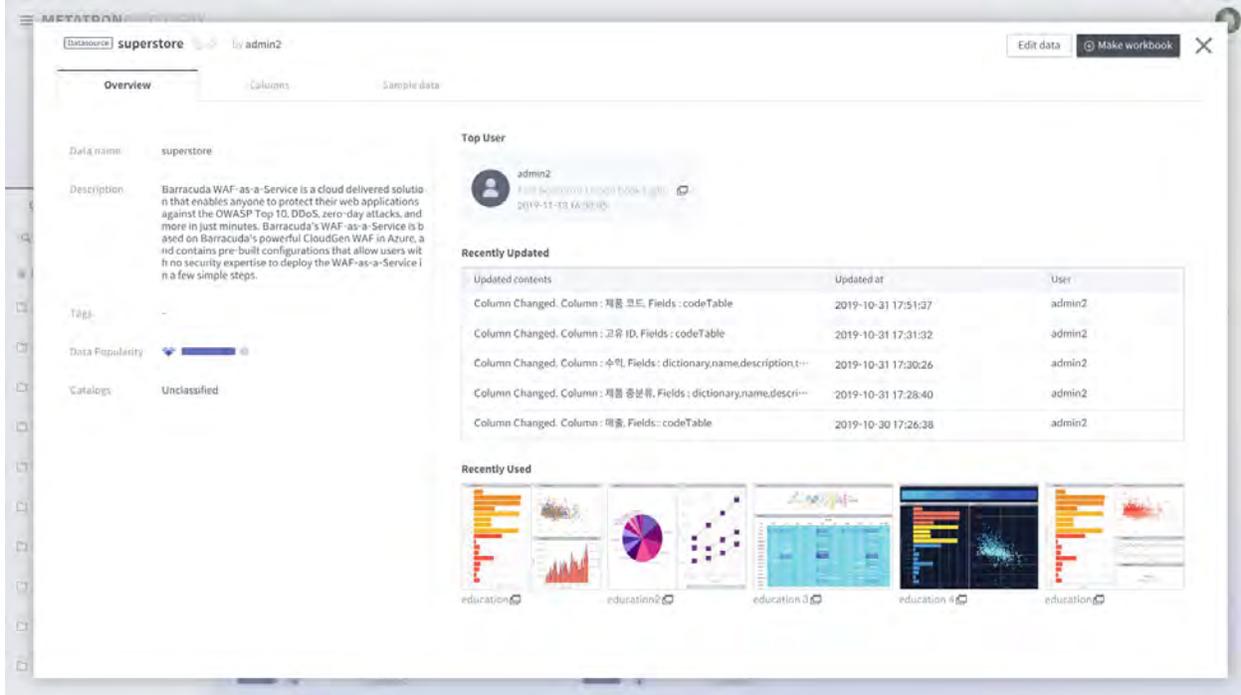
10.1.1 데이터 탐색 오버뷰 화면

현재 사용하고 계신 원천 DB 내 데이터, 그리고 Metatron Discovery에서 제공하는 StagingDB(Slave DB) 및 Engine(Druid) 내 데이터를 관리할 수 있습니다.

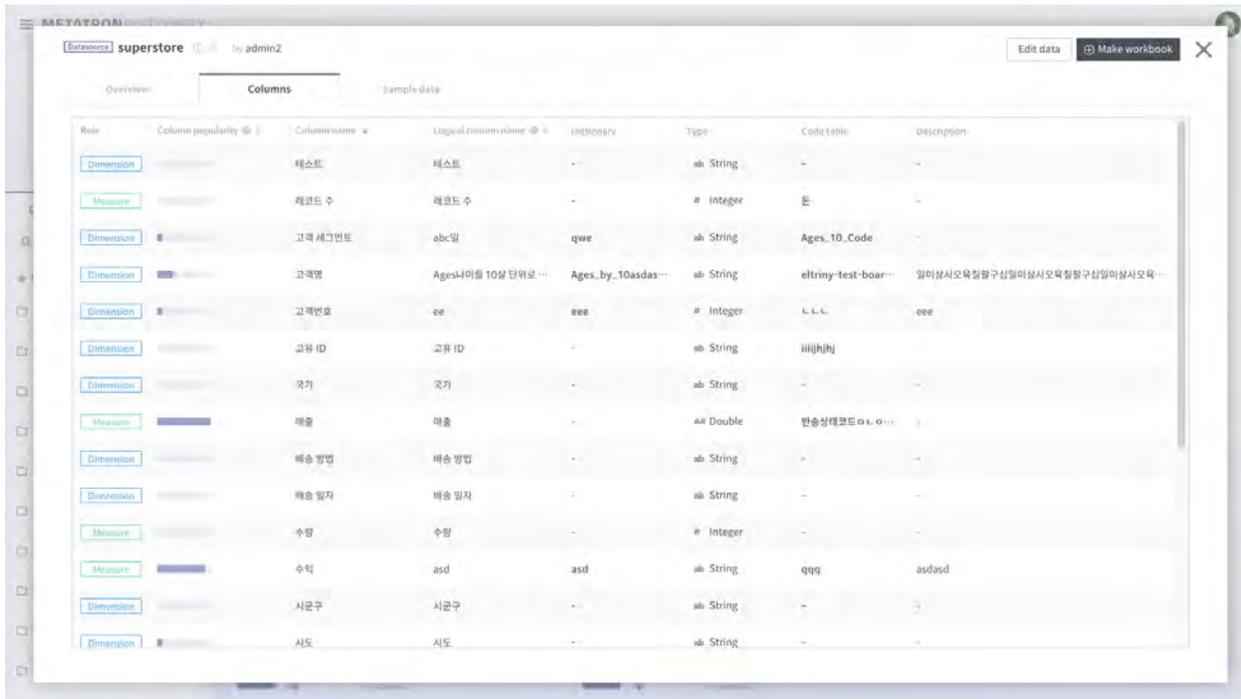


10.1.2 데이터 탐색 상세 화면

데이터 탐색을 통해 원하는 데이터를 빠르게 찾을 수 있도록 기능을 제공하고 있습니다.



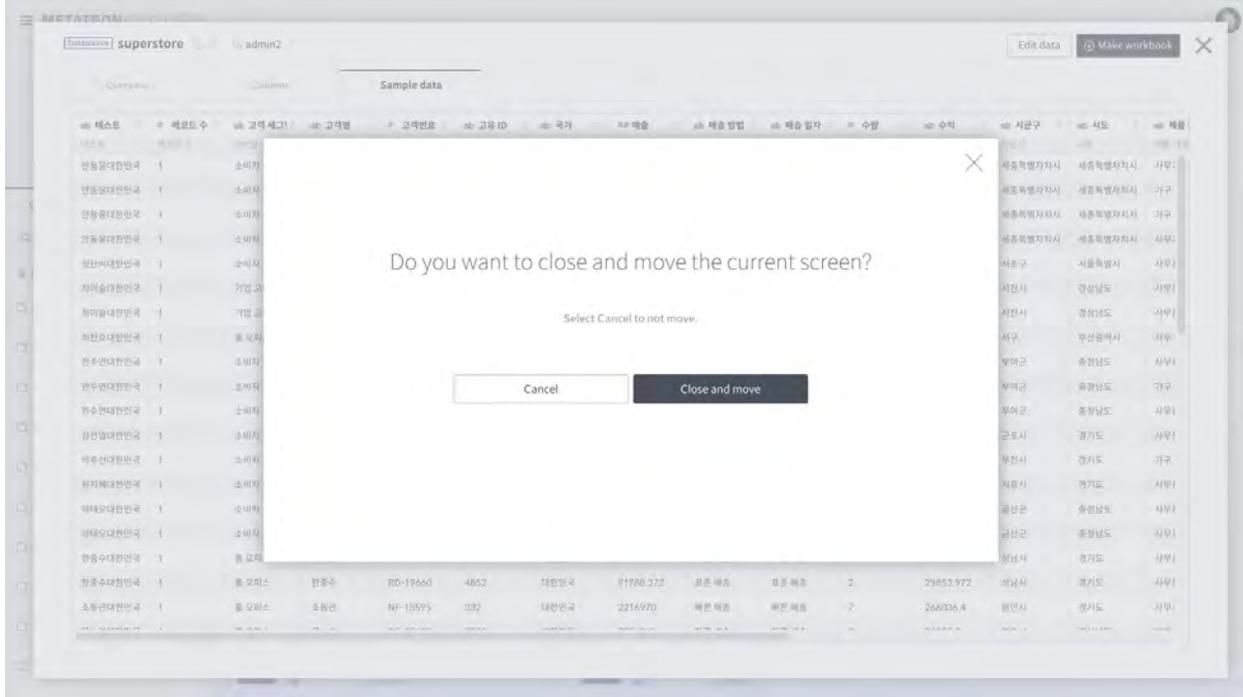
데이터는 크게 3부분으로 정보를 제공합니다. Overview, Column Scheme, Sample Data 입니다. 각 데이터 타입에 따라, 워크북 (Datasource 타입인 경우), 워크벤치 (DB 타입인 경우) 를 만들 수 있는 액션버튼도 제공합니다.



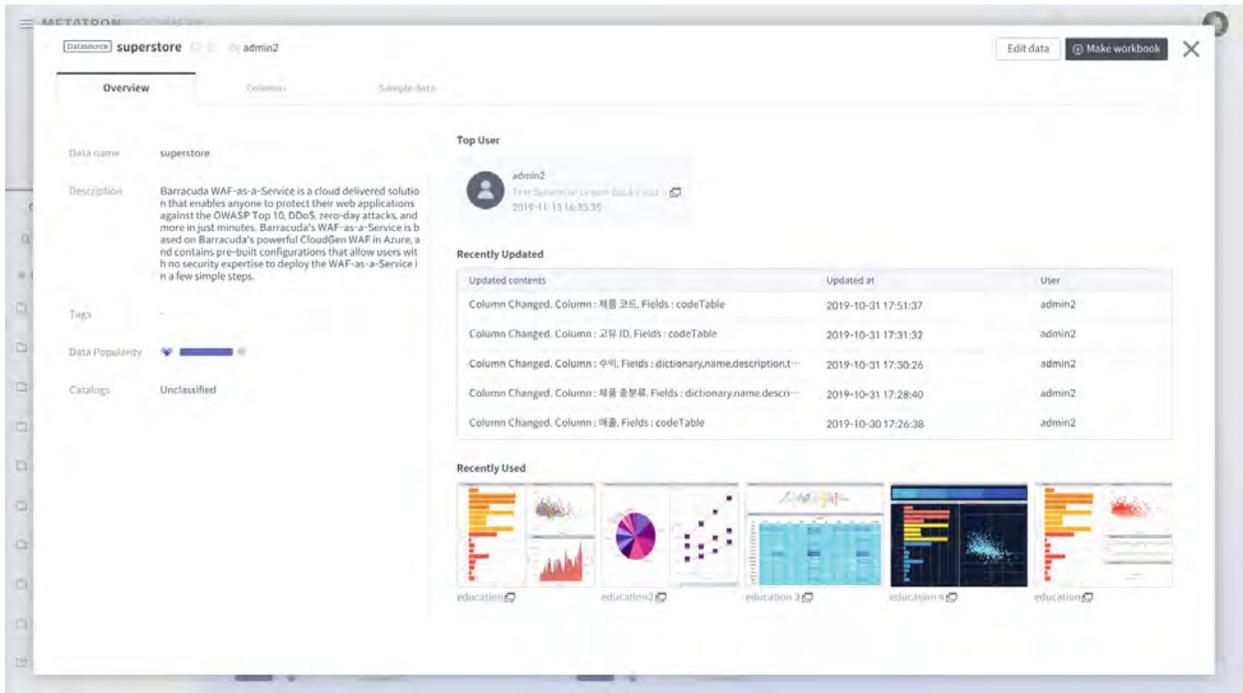
샘플 데이터의 경우 현재 100건까지만 보여줍니다. 권한이 있는 경우 <Management> Exploration'을 통해 더 많은 데이터를 확인하고 다운로드 할 수 있습니다. 권한의 여부는 해당 화면 상단에 <Edit data'라는 버튼의 유무로 확인할 수 있으며, 해당 버튼을 클릭하면 <Management> Exploration'로 이동이 가능합니다.

store	product	category	product name	product ID	country	sales	sales type	sales channel	quantity	price	region	city	sales status
영동출대한국	1	소비자	안동출	KL-16645	10156	대한민국	930705.12	표준 배송	표준 배송	4	-325853.28	세종특별자치시	세종특별자치시 사무
영동출대한국	1	소비자	안동출	KL-16645	10157	대한민국	162632.38	표준 배송	표준 배송	2	-97589.52	세종특별자치시	세종특별자치시 가구
영동출대한국	1	소비자	안동출	KL-16645	10158	대한민국	70245.36	표준 배송	표준 배송	2	-26940.24	세종특별자치시	세종특별자치시 가구
영동출대한국	1	소비자	안동출	KL-16645	10159	대한민국	70686	표준 배송	표준 배송	2	3488.4	세종특별자치시	세종특별자치시 사무
성남비대한국	1	소비자	황탄비	CM-12235	3697	대한민국	137820.564	표준 배송	표준 배송	2	-46819.836	서울시	서울특별시 사무
최아출대한국	1	기업 고객	최아출	KN-16450	8	대한민국	36600.66	당일 배송	당일 배송	3	4837.86	서산시	경상남도 사무
최아출대한국	1	기업 고객	최아출	KN-16450	9	대한민국	281624.04	당일 배송	당일 배송	1	112620.24	서산시	경상남도 사무
최한출대한국	1	홈 오피스	최한출	DP-13390	5081	대한민국	196727.4	표준 배송	표준 배송	1	51132.6	서구	부산광역시 사무
함수출대한국	1	소비자	함수출	LB-16735	6066	대한민국	6126.12	빠른 배송	빠른 배송	1	556.92	부여군	충청남도 사무
함수출대한국	1	소비자	함수출	LB-16735	6067	대한민국	215404.11	빠른 배송	빠른 배송	1	-71807.49	부여군	충청남도 가구
함수출대한국	1	소비자	함수출	LB-16735	6068	대한민국	344392.29	빠른 배송	빠른 배송	3	-125256.51	부여군	충청남도 사무
상산출대한국	1	소비자	상산출	BD-11500	5693	대한민국	27907.2	표준 배송	표준 배송	3	13953.6	군포시	경기도 사무
박주출대한국	1	소비자	박주출	SW-20455	8832	대한민국	49755.6	표준 배송	표준 배송	1	23378.4	부천시	경기도 가구
완지출대한국	1	소비자	완지출	KM-16660	3778	대한민국	31150.8	표준 배송	표준 배송	1	13372.2	시흥시	경기도 사무
하태출대한국	1	소비자	하태출	BH-11710	3748	대한민국	51202.368	표준 배송	표준 배송	3	6128.568	공산군	충청남도 사무
하태출대한국	1	소비자	하태출	BH-11710	3749	대한민국	41550.516	표준 배송	표준 배송	7	-27636.084	공산군	충청남도 사무
함중출대한국	1	홈 오피스	함중출	RD-19660	4851	대한민국	39544.686	표준 배송	표준 배송	3	11621.086	성남시	경기도 사무
함중출대한국	1	홈 오피스	함중출	RD-19660	4852	대한민국	91788.372	표준 배송	표준 배송	2	29853.972	성남시	경기도 사무
조동출대한국	1	홈 오피스	조동출	NF-18595	332	대한민국	2216970	빠른 배송	빠른 배송	7	266036.4	용인시	경기도 사무

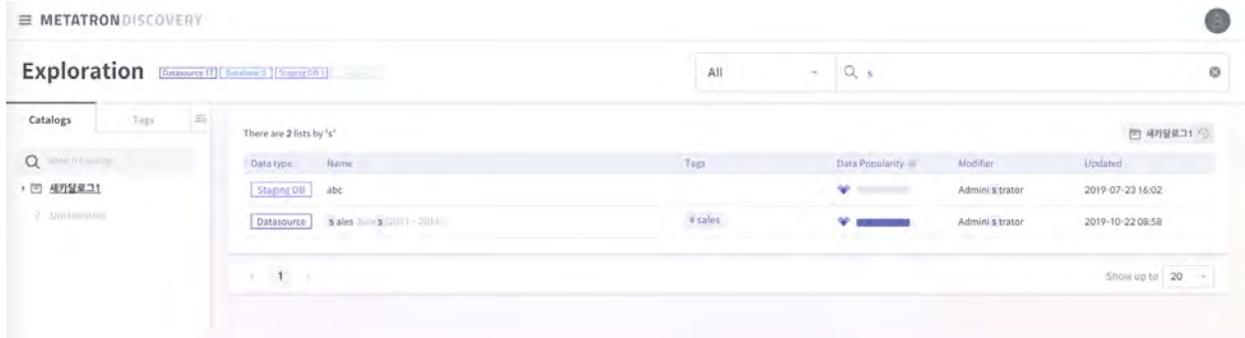
다른 메뉴로 점핑 시 다음과 같은 경고창이 보여집니다.



아래 화면은 <Management> 'Exploration' 로 이동하였을때 화면입니다. 해당 공간에서, 관리자로서의 좀 더 많은 메타 정보를 상세히 볼 수 있고 관리할 수 있습니다.



Metatron Discovery에서는 데이터를 카타로그로 관리할 수 있습니다. 카타로그을 그룹 등의 분류 군에 따라 구별해 두고 빠르게 데이터를 검색 하는 용도로 사용할 수 있습니다.



10.1.3 Favorite Data 화면

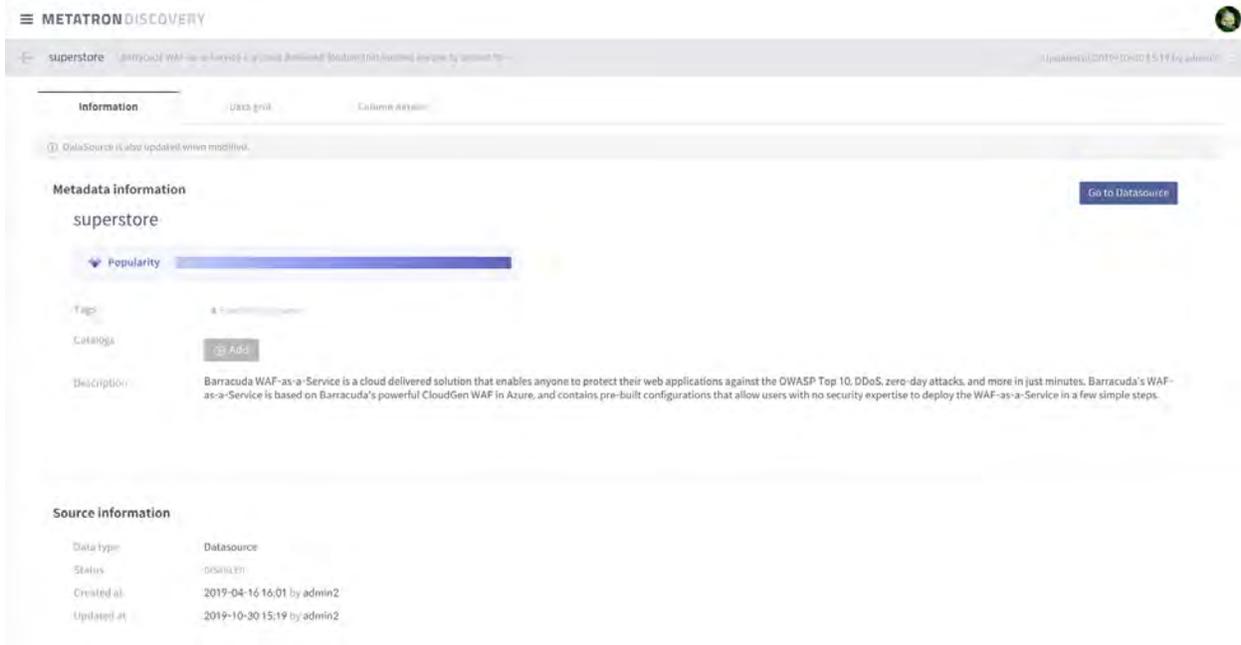
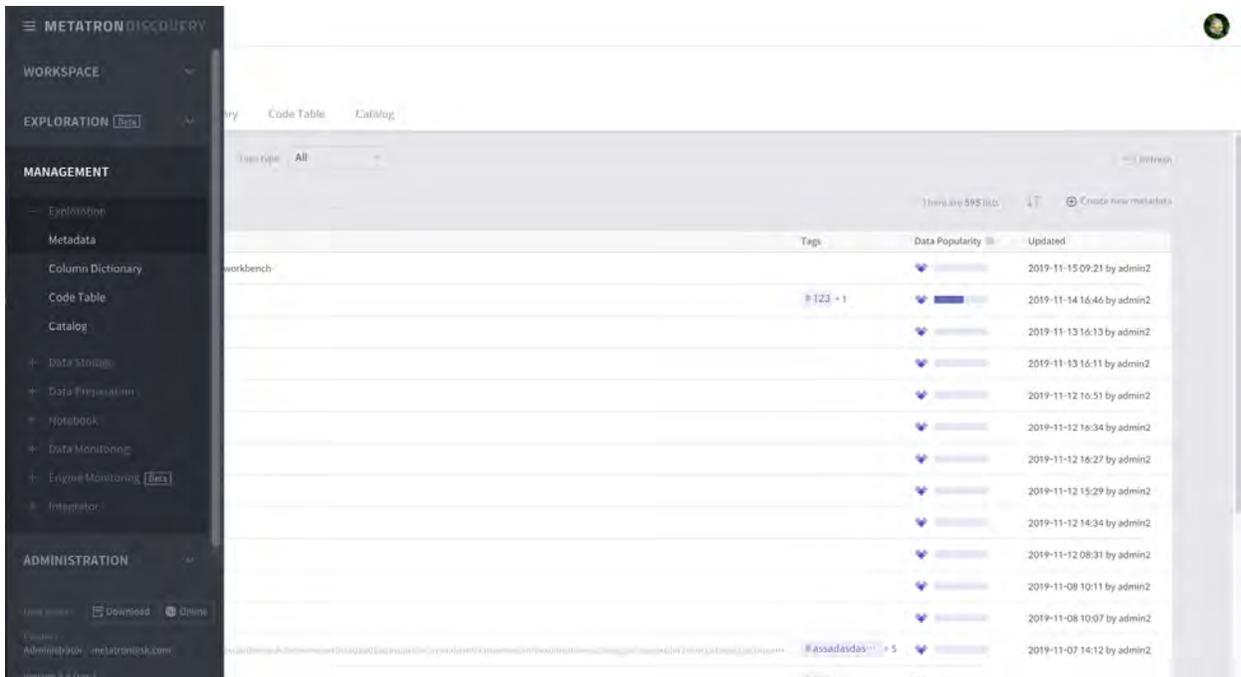
해당 기능은 준비 중입니다.

10.1.4 Data Creator 화면

해당 기능은 준비 중입니다.

10.2 메타데이터 관리하기

메타데이터는 Exploration에서 보여지는 각 데이터를 관리하고 더 자세하게 분석하기 위한 용도로 만들어졌습니다.



10.3 Column Dictionary

Exploration

Metadata Column Dictionary Code Table Catalog

Updated Data: All Today Last 7 days Filter Time: 10:00

Column Name	Type	Updated
abc일	STRING	2019-09-09 10:50 by admin2
Ages나이를 10살 단위로 표현나이를 10살 단위로 표현나이를 10살 단위로 표현 알	STRING	2019-11-07 16:17 by admin2
asd	STRING	2019-08-28 17:37 by admin2
ee	INTEGER	2018-11-12 10:45 by admin2
etrixy-hide-2	STRING	2019-04-29 11:04 by admin2
integer_test	TIMESTAMP	2019-08-21 17:18 by admin2
page	TIMESTAMP	2019-04-18 14:41 by admin2
ship_date	TIMESTAMP	2019-10-21 14:03 by admin2
string_test_c	STRING	2019-07-01 16:15 by admin2
test	TIMESTAMP	2019-08-27 16:31 by admin2
test123123132	STRING	2019-08-28 14:05 by admin2
testtttt	TIMESTAMP	2019-07-02 17:45 by admin2
test_time	TIMESTAMP	2019-07-02 17:39 by admin2
time_format_with_ms	TIMESTAMP	2019-06-17 15:19 by admin2

시도코드

Created on: 2019-02-21 15:08 by admin2 Last updated on: 2019-02-21 15:08 by admin2 [Delete this Column Dictionary](#)

Dictionary Information

Recommended Column Name: 시도코드, 시도명

Recommended Short Name: 시도코드

Description: 시도코드를 시도명으로

Code table: 시도코드to시도명

Format Information

Type: String

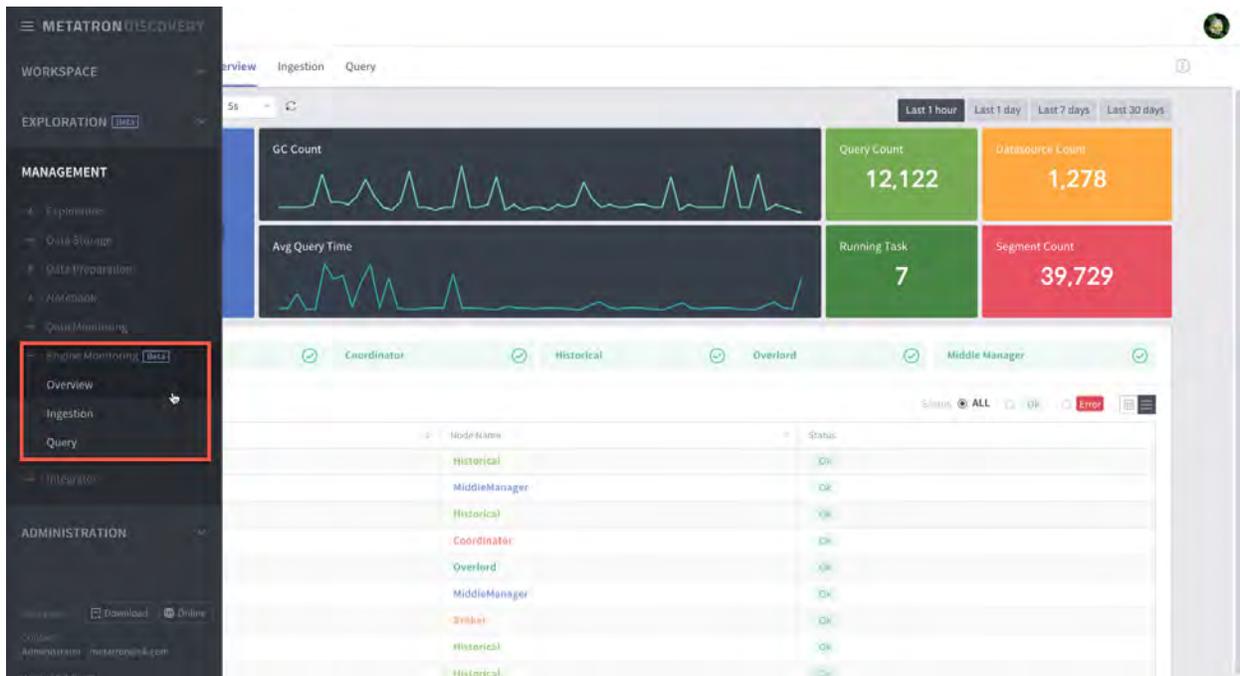
Used in Metadata (1)

Metadata name	Logical column name	Logical type	Format
전국상권	시도 코드	STRING	

CHAPTER 11

엔진 모니터링

엔진 모니터링은 Metatron Engine의 모니터링을 의미합니다. Metatron Engine은 Druid를 사용한 시계열 기반의 엔진입니다. 엔진 모니터링은 Ingestion, Query에 대한 상태 모니터링과 로그 정보를 보여줍니다.

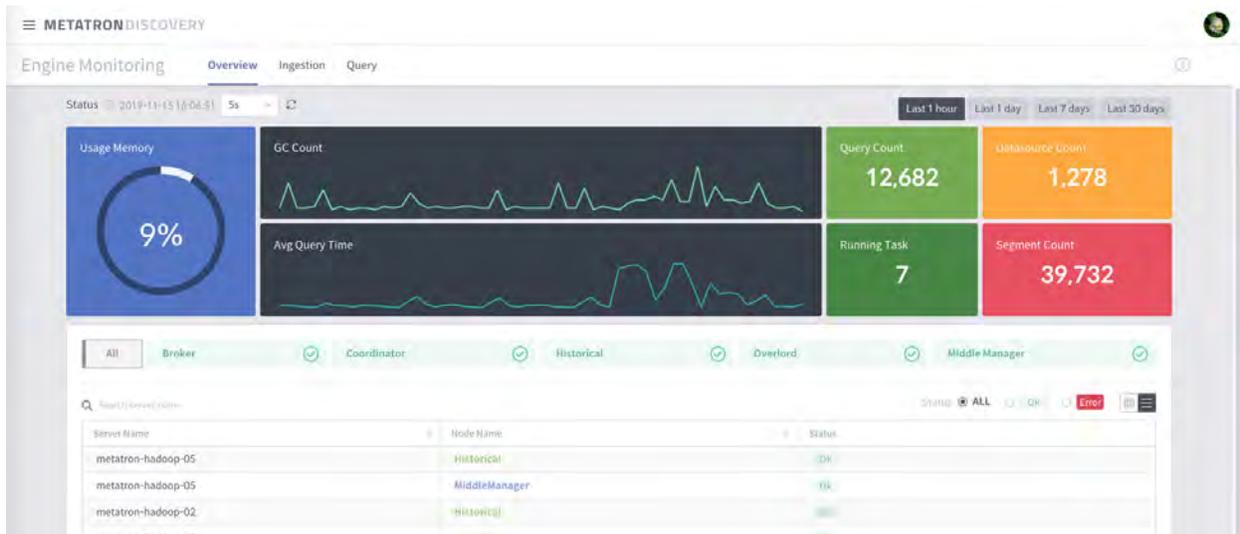


해당 기능은 Metatron Discovery 3.4.0 이후 버전부터 지원합니다.

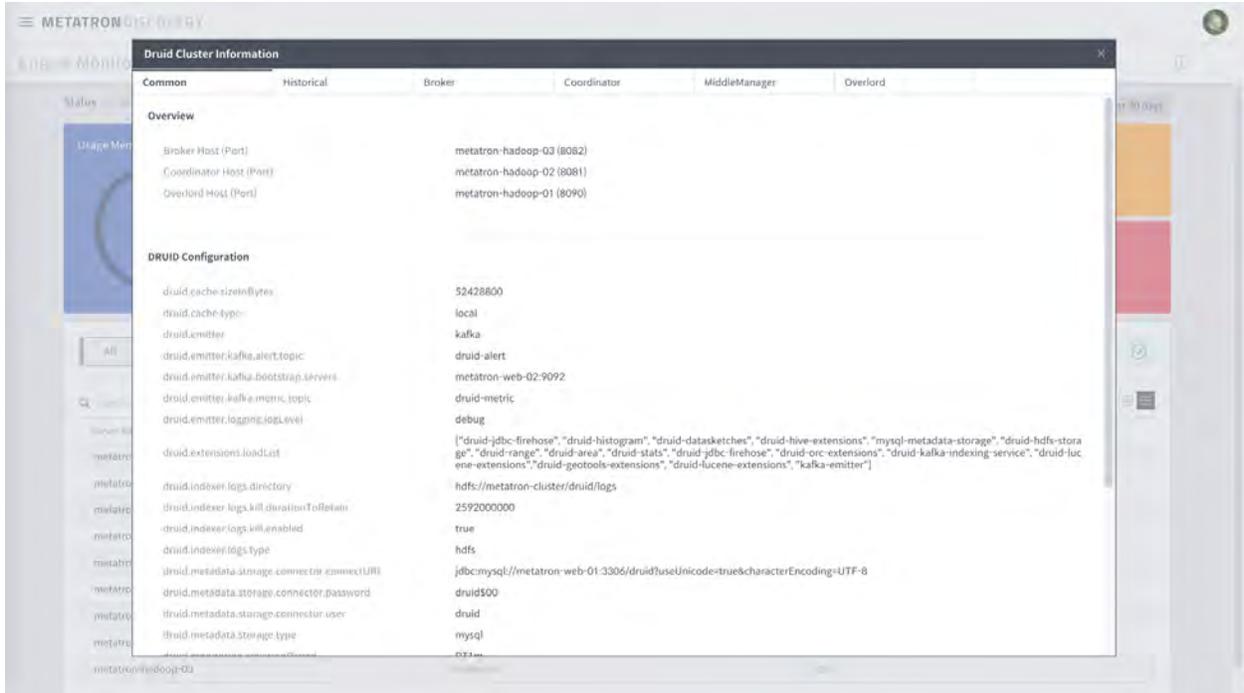
11.1 Overview

11.1.1 Druid Setting Configuration

Druid 설치 정보를 알 수 있습니다. 우측 상단을 보면 information버튼이 있습니다. 해당 버튼을 클릭하면, 설치된 정보를 확인 가능합니다.



Druid 설치 설정을 Common, Historical Node, Broker Node, Coordinator Node, Middle Manager Node, Overlord Node로 각각 확인 가능합니다.



11.1.2 Historical 사용량

각 historical node 의 사용량 표시합니다. Coordinator의 servers 리스트에서 개별 서버 항목 추출하여 만들어집니다.

11.1.3 Cluster 전체 사용량

Druid historical 모니터링기능을 제공합니다.

Cluster의 사용량 정보는 아래와 같습니다.

- cluster 전체 사용량 확인 가능
- 개별 historical 사용량 확인 가능

Coordinator의 servers 리스트를 이용하여 만들어진 KPI 입니다.

Field	Description	Example
Node Count	historical node 수	
MaxSize		
currSize		
Used		
FreeSize		

11.1.4 Historical 사용량

각 historical node 의 사용량 표시합니다. Coordinator의 servers 리스트에서 개별 서버 항목 추출하여 만들어집니다.

11.2 Ingestion

Druid Indexing Service 모니터링입니다. 해당 페이지에서는 Index task의 실행 상태 및 task 관련 정보를 제공합니다.

아래와 같은 정보를 제공해줍니다.

- **MiddleManager 상태 확인 가능**
 - worker 별 용량, 현재 사용량 현황
- **Supervisor 상태 확인 가능**
 - supervisor 별 상태
 - terminate (suspend, reset) 기능 제공
- **Task 상태 확인 가능**
 - runningTasks, pendingTasks, waitingTasks, completedTasks
 - log, kill 기능 제공
- **Lockbox 상태 확인 가능**

Ingestion 에는 supervisor와 middle manager에 대한 정보도 같이 확인할 수 있다.

11.2.1 Tasks

Task는 다음과 같이 4개로 분류할 수 있습니다.

- pending task: worker 할당을 기다리는 task
- running task: 실행중인 task
- waiting task: lock을 기다리는 task
- completed task: 완료된 task로 SUCCESS, FAIL의 두 상태로 나뉜다.

Task 상세 정보 및 메뉴는 아래와 같습니다.

Field	Description	Example
id	taskId	
type		
dataSource		
createdTime		
queueInsertionTime		
status		
runnerStatusCode		
duration		
locationhost		
locationport		
payload		
status	상태	
log		
log last 8k		
kill		
ingestion		

이와 같습니다.

Task ID	Status	Created time	Duration	Datasource	Type
index_kafka_dacoe_flink_geo_f13596c212c226d_bnamidan	RUNNING	2019-11-15 13:36:10.897	00:00:00	dacoe_flink_geo	kafka
index_kafka_dacoe_flink_1_1_0651d87a6709f50_ldoamibh	RUNNING	2019-11-15 13:36:10.799	00:00:00	dacoe_flink_1_1	kafka
index_kafka_systemshockrealtime20190827_12_0420d52b114173_idfbimik	RUNNING	2019-11-15 13:36:09.555	00:00:00	systemshockrealtime20190827_12	kafka
index_kafka_realtime_server_load_json_01_aad601ac12bb553_npllnhbd	RUNNING	2019-11-15 12:54:57.041	00:00:00	realtime_server_load_json_01	kafka
index_kafka_stream_test_3_20583fba514c5b_cmfjmij	RUNNING	2019-11-15 12:54:56.977	00:00:00	stream_test_3	kafka
index_kafka_systemshockrealtime20190827_12_0420d52b114173_nnnjcpm	RUNNING	2019-11-15 12:36:02.378	00:00:00	systemshockrealtime20190827_12	kafka
index_kafka_dacoe_flink_geo_f13596c212c226d_hjhbnibf	RUNNING	2019-11-15 12:36:02.378	00:00:00	dacoe_flink_geo	kafka
index_kafka_dacoe_flink_1_1_0651d87a6709f50_gbelcfl	RUNNING	2019-11-15 12:36:02.378	00:00:00	dacoe_flink_1_1	kafka
index_kafka_druid-metric_63bf128627d3806_alkamshf	RUNNING	2019-11-15 05:16:47.928	00:00:00	druid-metric	kafka
index_kafka_druid-metric-topic_d70ef2b0f6d8d77_gooahaln	RUNNING	2019-11-14 17:47:02.250	00:00:00	druid-metric-topic	kafka
index_oivws_2019-11-15T04:30:05.096Z	SUCCESS	2019-11-15 13:30:05.096	00:01:26	_oivws	index
index_batch_test_2019-11-15T04:30:04.118Z	SUCCESS	2019-11-15 13:30:04.118	00:00:08	batch_test	index
index_oivws_2019-11-15T04:20:04.839Z	SUCCESS	2019-11-15 13:20:04.839	00:01:50	_oivws	index
index_batch_test_2019-11-15T04:20:04.056Z	SUCCESS	2019-11-15 13:20:04.056	00:00:08	batch_test	index
index_oivws_2019-11-15T04:10:06.084Z	SUCCESS	2019-11-15 13:10:06.084	00:01:23	_oivws	index

상세 화면을 살펴보면 다음과 같습니다. (아래는 Kafka를 사용한 경우입니다)

Information

- Queue (creation time): 2019-11-15T04:55:04.937Z
- Created Time: 2019-11-15T04:55:04.924Z
- Host: metatron-hadoop-05
- Location: metatron-hadoop-05:9105
- Datasource: stream_test_3
- Type: kafka
- Precedence: 0
- UnitResource: 0
- Throughput: 0

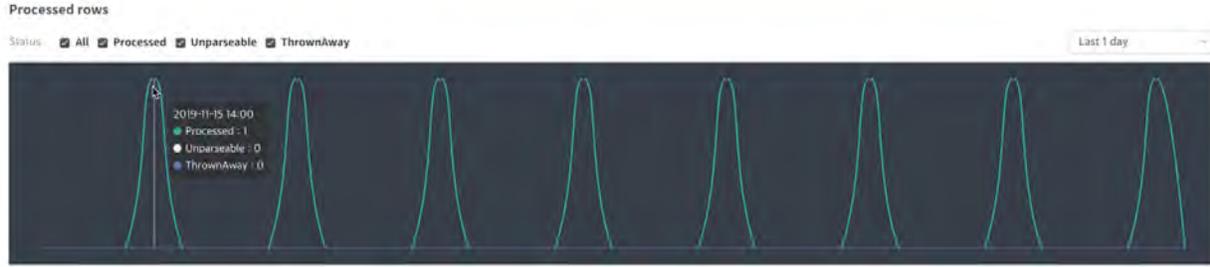
Status (Log BK)

RUNNING

```

2019-11-15T04:55:13.201 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.server.PriorityResource to GuiceInstantiatedComponentProvider
2019-11-15T04:55:13.228 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.server.Http.SecurityStateResourceFilter to GuiceInstantiatedComponentProvider
2019-11-15T04:55:13.238 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.server.Http.SegmentListenerResource to GuiceInstantiatedComponentProvider
2019-11-15T04:55:13.237 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.segment realtime.Increase.ChallengerResource to GuiceInstantiatedComponentProvider
2019-11-15T04:55:13.242 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.query.Lookups.LookupsListenerResource to GuiceInstantiatedComponentProvider
2019-11-15T04:55:13.245 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.query.Lookups.LookupsInspectorResource to GuiceInstantiatedComponentProvider
2019-11-15T04:55:13.248 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.server.http.security.ConfigResourceFilter to GuiceInstantiatedComponentProvider
2019-11-15T04:55:13.251 INFO [main] com.sun.jersey.guice.spi.container.GuiceComponentProviderFactory - Binding to druid.server.StatusResource to GuiceManagedComponentProvider with the scope "Undefined"
2019-11-15T04:55:13.280 WARN [main] com.sun.jersey.spi.inject.Errors - The following warnings have been detected with resource and/or provider classes:
WARNING: A HTTP GET method public void io.druid.server.Http.SegmentListenerResource.getSegments(long,long,java.servlet.http.HttpServletRequest) throws java.io.IOException, MUST return a non-void type.
2019-11-15T04:55:13.296 INFO [main] org.eclipse.jetty.server.handler.ContextHandler - Started o.e.j.s.ServletContextHandler@5ee40f5f{/,http://0.0.0.0:9105}
2019-11-15T04:55:13.316 INFO [main] org.eclipse.jetty.server.AbstractConnector - Started ServerConnector@131c0b4f{HTTP/1.1,http://0.0.0.0:9105}
2019-11-15T04:55:13.317 INFO [main] com.metamx.common.lifecycle.Lifecycle$AnnotationBasedHandler - Invoking start method on public void io.druid.query.lookups.LookupsReferencesManager.start() on object io.druid.query.Lookups.LookupsReferencesManager@666b8e27
2019-11-15T04:55:13.317 INFO [main] io.druid.query.lookups.LookupsReferencesManager - Started lookup factory references manager
2019-11-15T04:55:13.318 INFO [main] com.metamx.common.lifecycle.Lifecycle$AnnotationBasedHandler - Invoking start method on public void io.druid.server.listener.announcer.ListenerResourceAnnouncer.start() on object io.druid.query.Lookups.ListenerResourceAnnouncer@666b8e27
2019-11-15T04:55:13.324 INFO [main] io.druid.server.listener.announcer.ListenerResourceAnnouncer - Announcing start time on [/druid/listeners/lookups/_default/metatron-hadoop-05:9105]
    
```

Ingestion **RUNNING**



아래는 Kafka가 아닌 일반 Task의 경우의 모습입니다.

METATRON DISCOVERY

Index_olwvs_2019-11-15T05:30:06.027Z

Information

Queue Ingestion Time	1970-01-01T00:00:00.000Z
Created Time	2019-11-15T05:30:06.027Z
Host	
Location	
DataSource	_olwvs
Type	index

Status (Log 8K)

SUCCESS

```

at io.druid.emitter.kafka.KafkaEmitter.sendToKafka(KafkaEmitter.java:178) [kafka-emitter-0.9.1-SNAPSHOT.jar:0.9.1-SNAPSHOT]
at io.druid.emitter.kafka.KafkaEmitter.sendMetricsToKafka(KafkaEmitter.java:165) [kafka-emitter-0.9.1-SNAPSHOT.jar:0.9.1-SNAPSHOT]
at io.druid.emitter.kafka.KafkaEmitter.access$500(KafkaEmitter.java:51) [kafka-emitter-0.9.1-SNAPSHOT.jar:0.9.1-SNAPSHOT]
at io.druid.emitter.kafka.KafkaEmitter$2.run(KafkaEmitter.java:136) [kafka-emitter-0.9.1-SNAPSHOT.jar:0.9.1-SNAPSHOT]
at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511) [1.8.0_171]
at java.util.concurrent.FutureTask.runAndReset(FutureTask.java:308) [1.8.0_171]
at java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask.access$301(ScheduledThreadPoolExecutor.java:180) [1.8.0_171]
at java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask.run(ScheduledThreadPoolExecutor.java:274) [1.8.0_171]
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149) [1.8.0_171]
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624) [1.8.0_171]
at java.lang.Thread.run(Thread.java:748) [1.8.0_171]
2019-11-15T05:31:33.017 INFO [main] com.metamx.common.lifecycle.Lifecycle$AnnotationBasedHandler - Invoking stop method/public void io.druid.emitter.kafka.KafkaEmitter.close() on object[io.druid.emitter.kafka.KafkaEmitter@1162714a].
2019-11-15T05:31:33.017 INFO [main] org.apache.kafka.clients.producer.KafkaProducer - Closing the Kafka producer with timeoutMillis = 9223372036854775807 ms.
2019-11-15T05:31:33.017 INFO [main] io.druid.cli.CHProcess - Finished peon task
Heap
garbage-first heap: total 1032192K, used 521215K, (0x0000000700000000, 0x0000000700101f00, 0x00000007c0000000)
region size 1024K, 510 young (522240K), 32 survivors (32768K)
Metaspace: used 62549K, capacity 63942K, committed 64128K, reserved 1105920K
class space: used 2715K, capacity 8059K, committed 8064K, reserved 1048576K

```

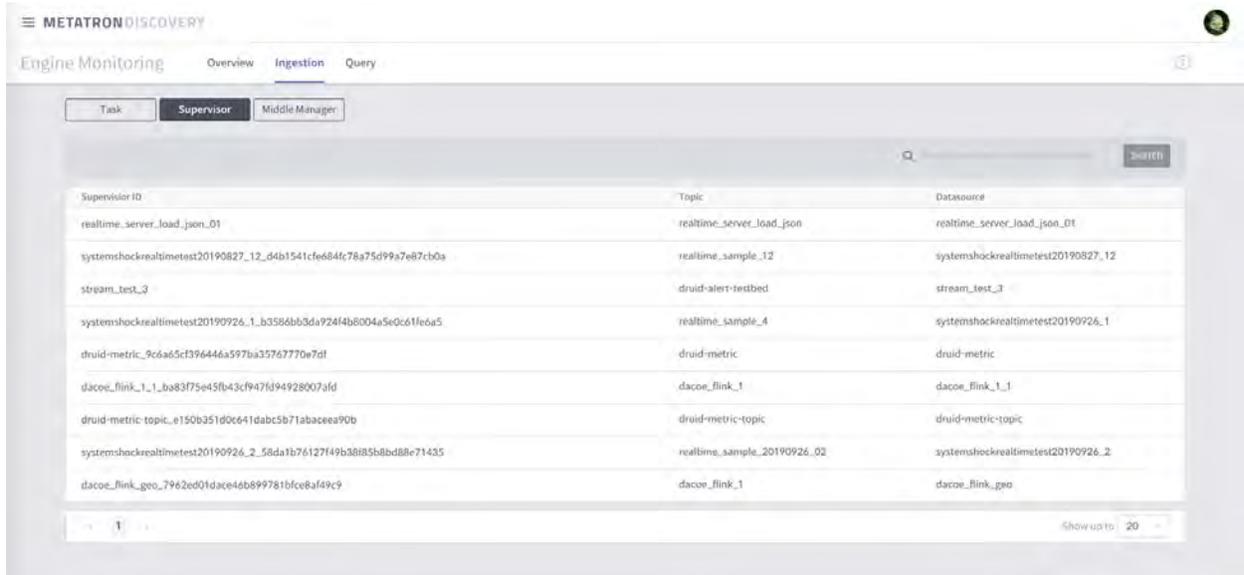
Ingestion SUCCESS

11.2.2 Supervisors

실행중인 Supervisors의 모니터링을 할 수 있다. 확인 가능한 Supervisor 상세 정보 및 menu는 다음과 같다.

Field	Description	Example
Status	get supervisorIDs로 제공되는 supervisor는 모두 running 상태임	
Datasource		
Detailed Status	status API 로 제공되는 정보	
Lag	kafka의 lag 정보. emitter 사용	
Spec		
Shutdown	Terminate supervisor. 관련된 task도 함께 kill 됨	

이와 같습니다.



11.2.3 MiddleManagers

worker 리스트를 의미합니다.

The screenshot shows the METATRON DISCOVERY Engine Monitoring interface. The 'Middle Manager' tab is selected. A table lists the following data:

Worker Host	Worker IP	Version	CapacityUsed/Total	Availability Groups	Running Tasks	Completed Time
metatron-hadoop-04:8091	localhost	0	4/10	4	4	2019-11-15 13:56:05.217
metatron-hadoop-05:8091	localhost	0	6/10	6	6	2019-11-15 13:51:55.992

The screenshot shows the METATRON DISCOVERY interface for a specific worker, 'metatron-hadoop-04:8091'. The 'information' section displays the following details:

- Host: metatron-hadoop-04:8091
- IP: localhost
- Capacity: 1 / 4/10
- Version: 0
- Availability Groups: 4
 - index_kafka_realtime_server_load_json_01_ba4ad60c3804abb
 - index_kafka_dacoe_flink_1_1_0651d87a6709f50
 - index_kafka_stream_test_3_0b5459136c28e81
 - index_kafka_druid-metric_63bf28627d3806
- Running Tasks: 4
 - index_kafka_realtime_server_load_json_01_ba4ad60c3804abb_bdmldbkk
 - index_kafka_druid-metric_63bf28627d3806_alkamjh
 - index_kafka_stream_test_3_0b5459136c28e81_gfflqjh
 - index_kafka_dacoe_flink_1_1_0651d87a6709f50_gmckohjp
- Last Completed Task Time: 2019-11-15T07:01:32.648Z

11.3 Query

METATRON DISCOVERY

Engine Monitoring Overview Ingestion Query

Result: ALL Service: ALL Type: ALL Start/End Time: ALL

ID	Result	Service	Host	Type	Datasource	Started Time	Duration (ms)
43a0b41f-a226-491c-8e3a-c5cd7397262f	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	ettest	2019-11-15 14:31:59.360	1
ec81be45-d6e2-4094-9135-9d6750186003	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	stcaz	2019-11-15 14:31:59.142	1
13d75e7b-5965-41e6-845f-5769dc54308a	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	JustATestName	2019-11-15 14:31:58.928	0
bdcdbe5d-3541-4040-a2f2-c66259169f4	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	dd11	2019-11-15 14:31:58.721	0
94d9b85a-3250-4c95-89cf-a01d2c92e177	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	hive__preset_engine_di...	2019-11-15 14:31:58.504	3
88185af6-788d-49f7-90fc-6f2beb795936	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	stage_part_test2	2019-11-15 14:31:58.274	0
394da03a-b9e6-4aed-81d8-598b52d01d78	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	dss_parete1	2019-11-15 14:31:58.060	0
77004272-dad1-4106-b0cc-1898ecd8929a	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	order	2019-11-15 14:31:57.828	0
32f9b133-bb91-45ad-9283-45c87b34c0f4	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	ggrrgg	2019-11-15 14:31:57.606	0
49df052b-b4ef-42ad-9d14-8624e1d42bd3	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	대한민국 시군구별 multip...	2019-11-15 14:31:57.389	0
33b63a6f-3849-47d5-abac-26b4054858a8	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	lhhk_vhyad	2019-11-15 14:31:57.168	0
f000db3-96db-4c55-ac89-8799e1853b8a	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	Query Parsed(eventtime)	2019-11-15 14:31:56.952	1
68280d2d-2461-4192-acca-5c7426f1a990	Success	druid/prod/broker	metatron-hadoop-03.8082	segmentMetadata	correlation matrix for m...	2019-11-15 14:31:56.732	0

METATRON DISCOVERY

Query Information

Query ID	43a0b41f-a226-491c-8e3a-c5cd7397262f
Result	Success
Service	druid/prod/broker
Host	metatron-hadoop-03.8082
Type	segmentMetadata
Datasource	ettest
Started Time	2019-11-15 14:31:59.360
Duration	1 ms

68280d2d-2461-4192-acca-5c7426f1a990 Success druid/prod/broker metatron-hadoop-03.8082 segmentMetadata correlation matrix for m... 2019-11-15 14:31:56.732 0

Part II

EX-pack for Workflow Integrator

Integrator 확장팩 소개

Integrator 확장팩은 Hadoop 워크플로우 관리 시스템인 Apache Oozie를 조작하기 쉬운 GUI로 구현하고, 워크플로우를 거쳐 적재된 데이터를 Metatron Discovery에 바로 사용할 수 있도록 지원하는 모듈입니다. 이를 통해 사용자는 반복적으로 수행해야 할 Hadoop 작업 루틴을 간편하게 설계·구축하고 실행 주기를 설정하여, Metatron Discovery 작업에 필요한 데이터를 주기적으로 확보할 수 있습니다.

Integrator 확장팩의 주요 특징은 다음과 같습니다.

워크플로우 편집과 예약을 동시에

직관적인 차트 에디터를 이용하여 워크플로우를 손쉽게 생성하고 실행을 예약할 수 있습니다.

여러 클러스터를 한번에 관리

워크플로우 내 작업 노드별로 원천 데이터의 출처와 가공 후 적재할 테이블을 자유롭게 지정할 수 있어, 여러 클러스터를 한번에 관리할 수 있습니다.

워크플로우 공유

구축된 워크플로우는 조직 내 여러 사람이 함께 공유하고 관리할 수 있습니다.

알람 및 보고서

예약된 워크플로우 실행 결과는 SMS, 이메일, 메신저 등 다양한 채널을 통해 보고받을 수 있습니다.

CHAPTER 13

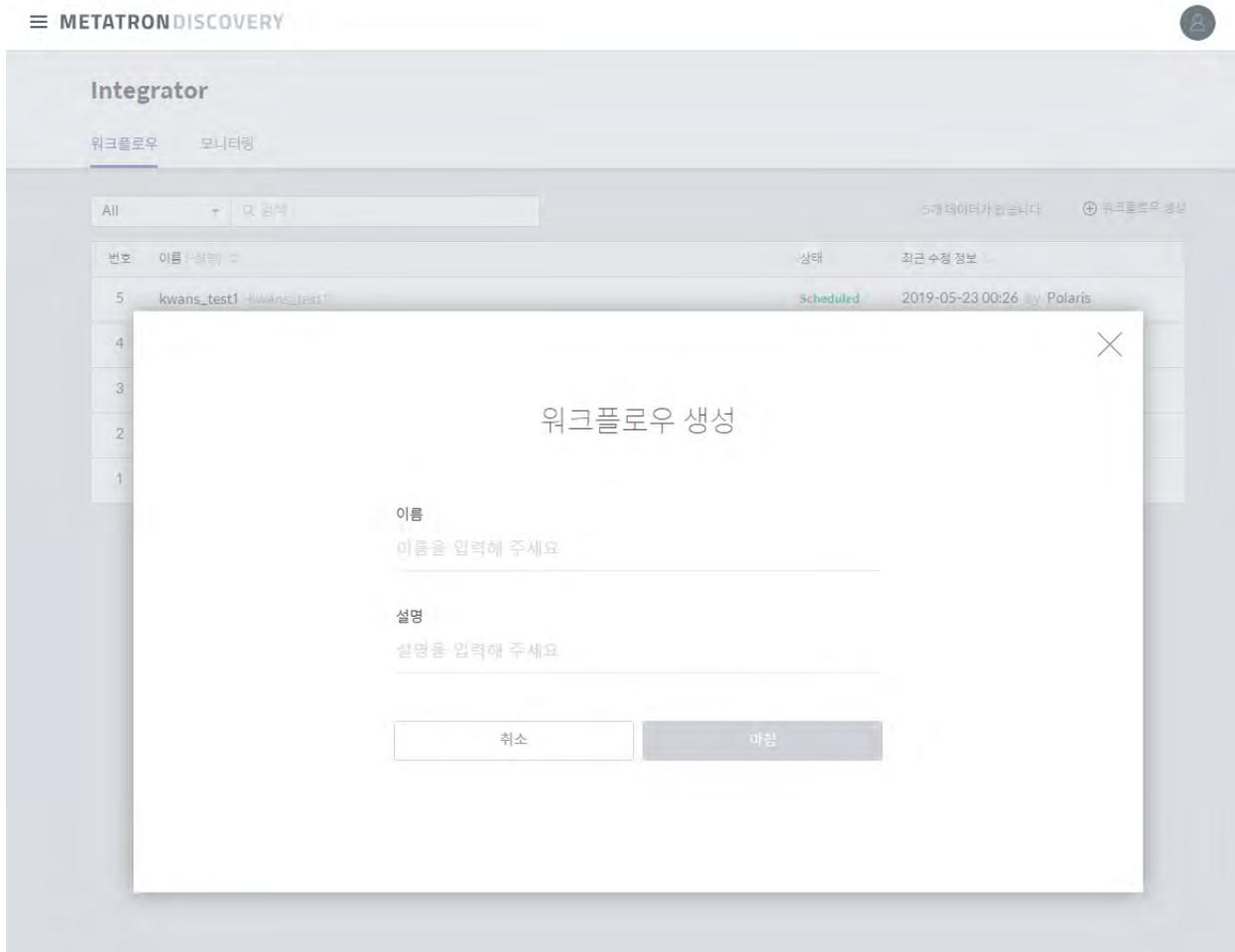
워크플로우 리스트

Integrator 메인 홈에서 워크플로우 탭으로 들어가면, 아래와 같이 현재 등록된 워크플로우들을 열거하여 보여줍니다. 상태 컬럼에서는 각 워크플로우의 진행 현황을 간략하게 보여줍니다.

번호	이름 (-설명)	상태	최근 수정 정보
5	kwans_test1 -kwans_test1	Scheduled	2019-05-23 00:26 by Polaris
4	test	Draft saved	2019-05-23 00:01 by Polaris
3	test2	Created	2019-05-21 16:05 by Polaris
2	ibk-approval-request-test	Scheduled	2019-05-17 03:12 by Administrator
1	collect apartment trade data -극동우 아파트 실거래가 데이터	Created	2019-03-28 13:44 by Polaris

리스트에 나열된 워크플로우 중 하나를 클릭하면 해당 워크플로우의 에디터 화면으로 이동합니다. 워크플로우 에디터에 대한 자세한 설명은 워크플로우 에디터 항목을 참조하십시오.

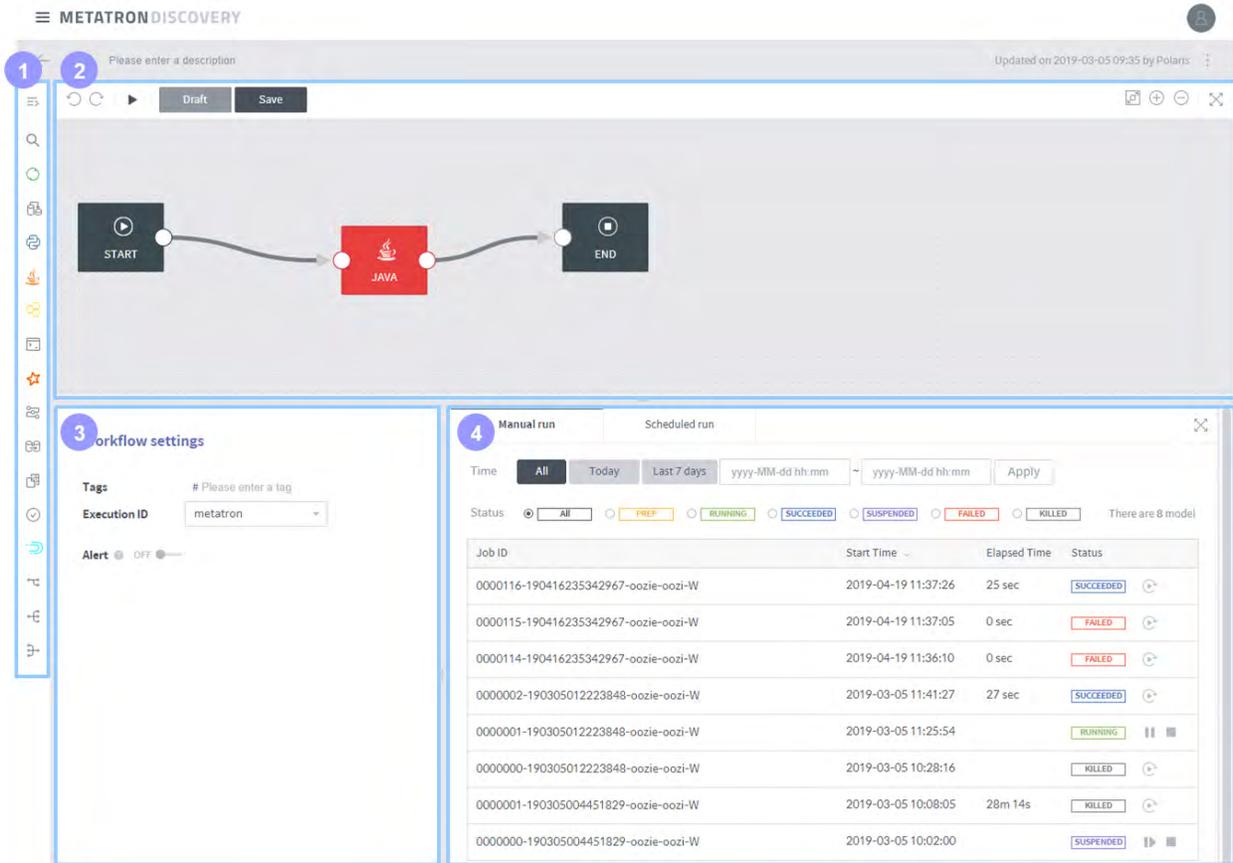
화면 우측 상단의 + 워크플로우 생성을 클릭하면 새로운 워크플로우를 생성할 수 있는 대화 상자가 열립니다. 만들고자하는 워크플로우의 이름과 설명을 입력한 후 마침 버튼을 클릭하면 새로운 워크플로우가 생성됩니다.



CHAPTER 14

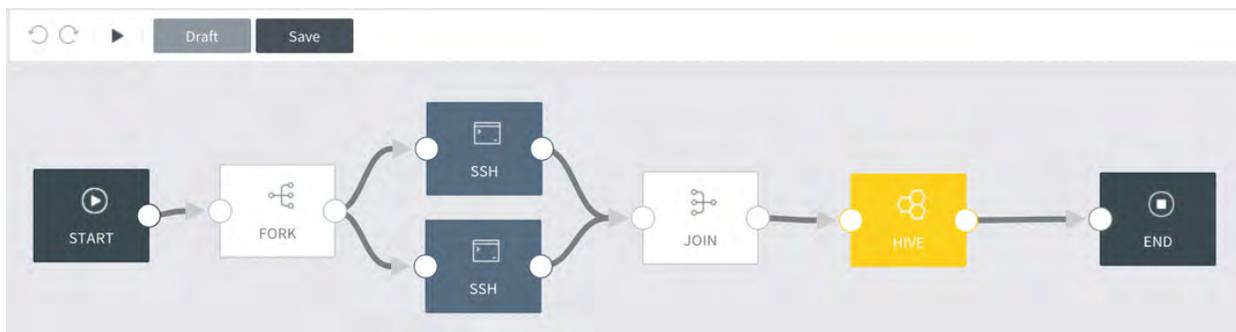
워크플로우 에디터

워크플로우 에디터는 선택한 Hadoop 워크플로우를 손쉽게 편집하고 실행을 스케줄링할 수 있는 GUI를 제공합니다. 워크플로우리스트에 열거된 워크플로우 중 하나를 클릭하면 워크플로우 에디터로 이동하며, 화면 구성은 아래와 같습니다.



- 워크플로우 노드 선택 영역:** 워크플로우에 추가할 노드들을 선택하는 영역입니다. ≡ 버튼을 클릭하면 패널이 확장되어 각 노드의 명칭을 확인할 수 있습니다. 다음과 같은 두 종류의 노드로 구분됩니다.

 - 액션 노드 (에디터에서 <Task' 로 분류):** 원천 데이터를 Hadoop 클러스터에서 수집·가공·적재하기 위한 각각의 연산처리 작업을 정의합니다. 자세한 내용은 **액션 노드** 항목을 참조하십시오.
 - 제어 흐름 노드 (에디터에서 <General' 로 분류):** 워크플로우의 시작과 끝을 정의하고, 액션 노드들의 흐름 경로를 결정하는 역할을 합니다. 자세한 내용은 **제어 흐름 노드** 항목을 참조하십시오.
- 워크플로우 차트 캔버스:** 추가한 노드들 간의 시퀀스를 정의하는 영역입니다. 아래 그림과 같이 노드 선택 영역에서 원하는 노드들을 캔버스로 드래그한 후, 원하는 시퀀스에 맞춰서 노드끼리 연결을 하면 워크플로우 차트가 간단하게 완성됩니다.



영역 상단에 있는 버튼을 사용하여 undo와 redo가 가능하며, 버튼을 클릭하면 현재 정의된 워크플로우가 실행됩니다. 또한 **Draft** 버튼을 클릭하면 현재까지 작업한 워크플로우가 저장되고, **Save** 버튼을 클릭하면 실제 워크플로우로서 반영이 됩니다.

3. **워크플로우 노드 설정 영역**: 워크플로우 차트 캔버스에서 선택한 각 개별 노드의 상세 작업 내역을 설정하는 영역입니다. 자세한 설정 방식은 [액션 노드](#) 및 [제어 흐름 노드](#)에서 해당 노드 항목을 참조하십시오.
4. **워크플로우 실행 내역 표시 영역**: 정의된 워크플로우의 실행 내역을 보여주는 영역입니다.
 - **Manual run 탭**: 에디터 좌측 상단에 있는 버튼을 클릭하여 수동으로 워크플로우를 실행한 내역을 보여줍니다.
 - **Scheduled run 탭**: 정해진 시간에 따라 워크플로우 실행을 예약하는 UI를 제공하고 예약된 내역을 보여줍니다. 자세한 내용은 [워크플로우 실행 예약하기](#) 항목을 참조하십시오.

워크플로우 에디터 사용을 위해 보다 상세한 설명이 필요한 부분에 관해서는 아래와 같이 정리하였습니다.

14.1 액션 노드

Integrator의 액션 노드들 (action nodes) 은 원천 데이터를 Hadoop 클러스터에서 수집·가공·적재하기 위한 각각의 연산처리 작업을 정의합니다. 아래와 같이 여러 가지 Hadoop 작업과 몇몇 추가적인 개별 시스템 작업 (Java, Shell 등) 을 지원합니다.

- Sqoop
- MR
- EXEC
- Java
- HIVE Query
- SSH

- Spark
- Sub-Workflow
- DistCp
- HDFS
- Done
- Druid

14.1.1 Sqoop

RDB 상의 데이터를 가져오거나 간단한 쿼리를 실행할 수 있는 task 입니다.

14.1.2 MR

Local에 있는 jar를 실행하는 데 사용합니다.

14.1.3 EXEC

Python, shell 등 로컬에 있는 파일을 실행하는데 사용합니다.

14.1.4 Java

Java Task는 Java Class를 실행하고자 할 때 사용합니다. (단, main 함수가 구현되어 있어야 합니다.)

14.1.5 HIVE Query

Hive 쿼리를 실행할 때 사용합니다.

14.1.6 SSH

원격지 (remote) 에 있는 명령어를 실행할 때 사용합니다. 다만, remote 서버는 SSH password-less login 설정이 되어 있어야 합니다.

14.1.7 Spark

SPARK를 실행하는데 사용합니다.

14.1.8 Sub-Workflow

기존 만들어진 Workflow와 연계 시 사용됩니다. 여러개의 Workflow 를 묶어서 실행하고자 할 때 각각의 Workflow 를 Task 로 정의합니다.

14.1.9 DistCp

Source Hadoop Cluster 에서 Target Hadoop Cluster에 파일 복사시 사용합니다.

14.1.10 HDFS

Hadoop File 관리시 사용합니다.

14.1.11 Done

완료시 Done 파일을 생성합니다.

14.1.12 Druid

Druid 엔진에 데이터 증분적재하기 위해 사용합니다.

14.2 제어 흐름 노드

Integrator의 제어 흐름 노드들 (control-flow nodes) 은 워크플로우의 시작과 끝을 정의하고, 액션 노드들의 흐름 경로를 결정하는 역할을 합니다. 지원하는 노드는 다음과 같습니다.

- Start
- End
- Decision

- Fork
- Join

14.2.1 Start

모든 워크플로우의 시작점입니다. 워크플로우를 실행하기 위해서는 필수로 들어가야 합니다.

14.2.2 End

모든 워크플로우의 종료점입니다. 워크플로우를 종료하기 위해서는 필수로 들어가야 합니다.

14.2.3 Decision

조건에 따라 분기할 수 있게 하는 노드입니다. 가지수 만큼의 Switch case 문이 발생합니다.

14.2.4 Fork

조건 없이 분기하여 무조건 실행하는 동시실행 분기함수 (parallel execution) 입니다.

14.2.5 Join

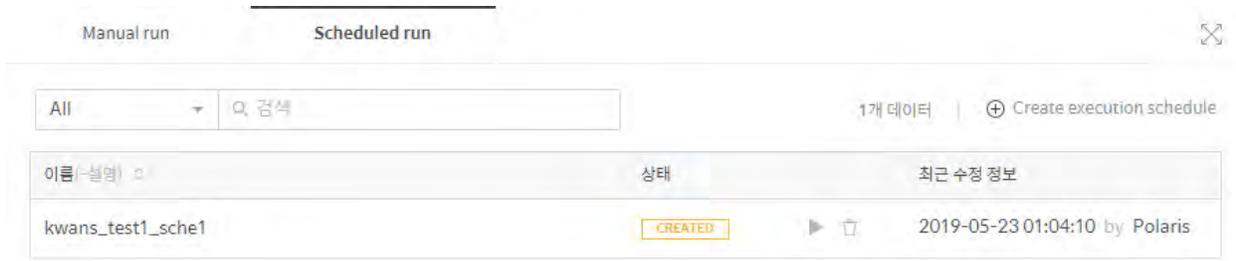
여러 노드를 합쳐주는 역할을 합니다.

14.3 워크플로우 실행 예약하기

워크플로우를 정해진 주기에 따라 반복적으로 실행해야할 경우 이러한 실행을 예약하고, 그 결과를 SMS, 메신저, 이메일 등으로 보고받을 수 있습니다.

14.3.1 예약 실행 리스트

워크플로우 에디터 우측 하단의 실행 내역 표시 영역에서 **Scheduled run** 탭을 클릭하면, 아래와 같이 해당 워크플로우에 대해 등록된 예약 실행 리스트가 나타납니다. 이 리스트에는 각 예약 실행 항목의 실행 현황이 표시되며, ▶ 버튼을 누르면 해당 예약 건이 실행되고 🗑 버튼을 누르면 삭제됩니다.



14.3.2 예약 실행 추가하기

Scheduled run 영역에서 + Create execution schedule을 클릭하면 다음과 같이 새 예약 실행을 생성하는 대화 상자가 열립니다. 아래 설명을 참조하여 각 필드 입력 후 생성 버튼을 클릭하십시오.

Create a New Execution Schedule

name

Description

Tags # 태그명을 입력하세요

Workflow

Period From To

Frequency

Concurrency

Timeout(min)

Datasets

Configuration

Variables

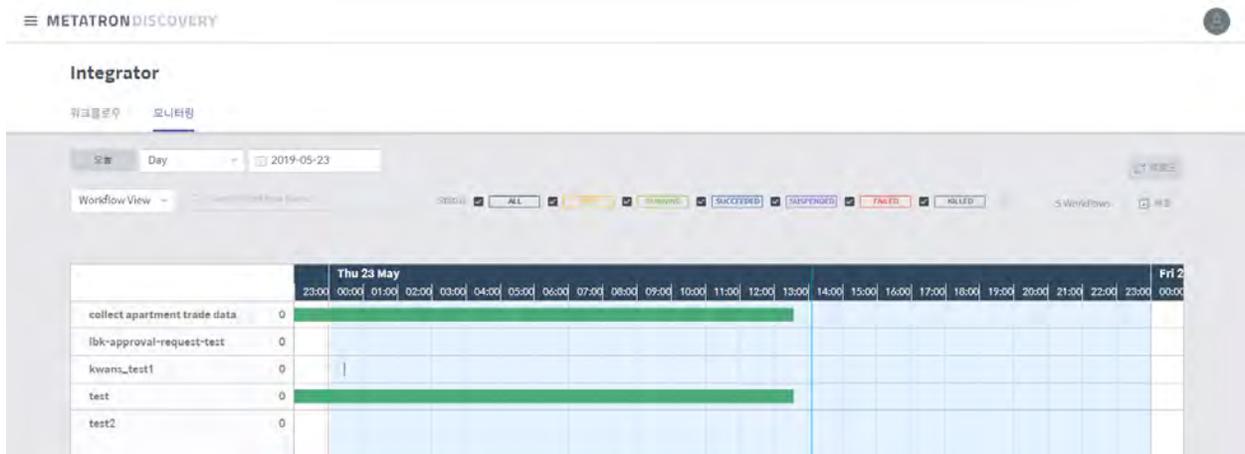
Alert OFF

- **Name:** 예약 실행의 이름을 입력합니다.
- **Description:** 예약 실행에 대한 설명을 입력합니다.
- **Tags:**

- **Workflow:** 예약 실행할 워크플로우를 선택합니다.
- **Period:** 예약 실행될 기간의 시작과 끝을 설정합니다.
- **Frequency:** 예약실행되는 기간 중 반복 주기를 설정합니다.
- **Concurrency:**
- **Timeout(min):**
- **Datasets:**
- **Configuration:**
- **Variables:**
- **Alert:**

모니터링

Integrator 메인 홈에서 **모니터링** 탭으로 들어가면 등록된 각 워크플로우의 시간대별 실행 현황과 예약 정보를 그래프 형식으로 보여줍니다.

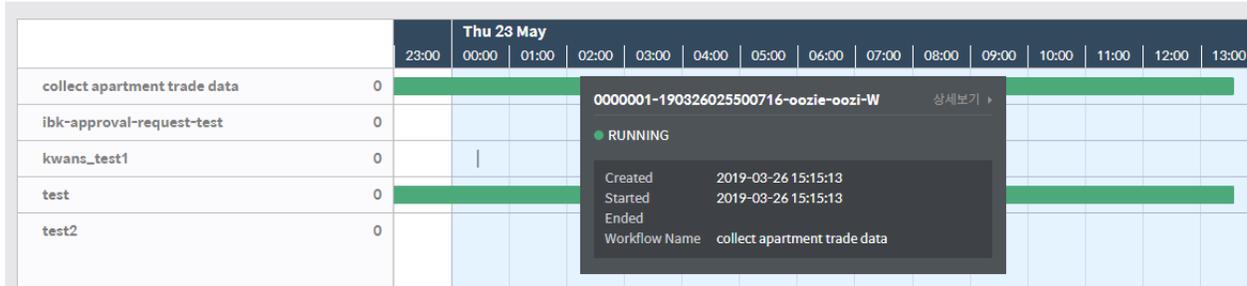


그래프에 표시된 상태 바 각각은 예약 또는 직접 명령한 워크플로우 실행 건을 가리키며, 다음과 같은 방식으로 실행 정보를 보여줍니다.

- 위치와 길이: 해당 건이 실행된 기간에 해당하는 타임라인 구간에 표시됩니다.

- 색상: 화면 상단에 범례로 표시된 **Status** 항목의 표시 색상과 동일한 색상으로 표시됩니다. 예를 들어 녹색으로 표시된 상태 바는 해당 실행 건이 running 상태임을 의미합니다.

상태 바 위에 마우스 커서를 오버하면 아래와 같은 실행 내역이 나타나며, 대화 상자 우측 상단의 **상세보기**를 클릭하면 자세한 정보를 확인할 수 있습니다.



16.1 데이터 소스 적재 위임

- 대용량의 데이터 적재 시 발생하는 시스템 부하를 방지하기 위해 background 처리

16.2 워크벤치와 연계

- 특정 쿼리의 반복 실행
- 오래 걸리는 쿼리의 위임

16.3 데이터 프리퍼레이션과 연계

- Wrangled dataset 반복 사용

Part III

EX-pack for Anomaly Detection

Metatron Anomaly 소개

이상 탐지 확장팩 Anomaly는 Machine Learning 예측 모델을 기반으로 데이터 흐름의 비정상적인 상황을 감지하여 사용자가 즉각적으로 확인할 수 있도록 도와주는 도구입니다.

17.1 기본 원리

아래 그림과 같이 Anomaly는 대상 데이터 소스의 집계값을 실시간으로 예측하고 실제 값을 모니터링합니다.



여기서 **Predict**로 표시된 값은 머신러닝 기반으로 예측한 데이터 집계값이고, **Actual**로 표시된 값은 실제로 모니터링한 결과 값입니다. 아래 그림과 같이 두 값 간의 격차가 커질수록 **total abnormal score**가 증가하게 됩니다. 즉, 실제치가 예상치와 다르면 데이터 집계값이 그만큼 정상 범위를 벗어났다고 간주하는 것입니다.



이 예시에서는 abnormal score가 20점에 도달하면 Low 레벨의 알람을 발생시키고, 40점을 넘으면 Moderate, 60점을 넘으면 Major, 80점을 넘으면 Critical 알람을 발생시키도록 설정되어 있습니다. training data에 따르면 사용자들은 4월 6일 15시에 Critical 등급의 알람을 받았을 것이라고 예측할 수 있습니다.

이렇게 설정된 알람 룰은 데이터가 갱신되면서 알람을 발생시키고 다양한 채널로 사용자에게 통보됩니다. 따라서 관련 시스템 운영자 및 사용자들은 데이터 이상 상황에 즉각 대처할 수 있습니다.

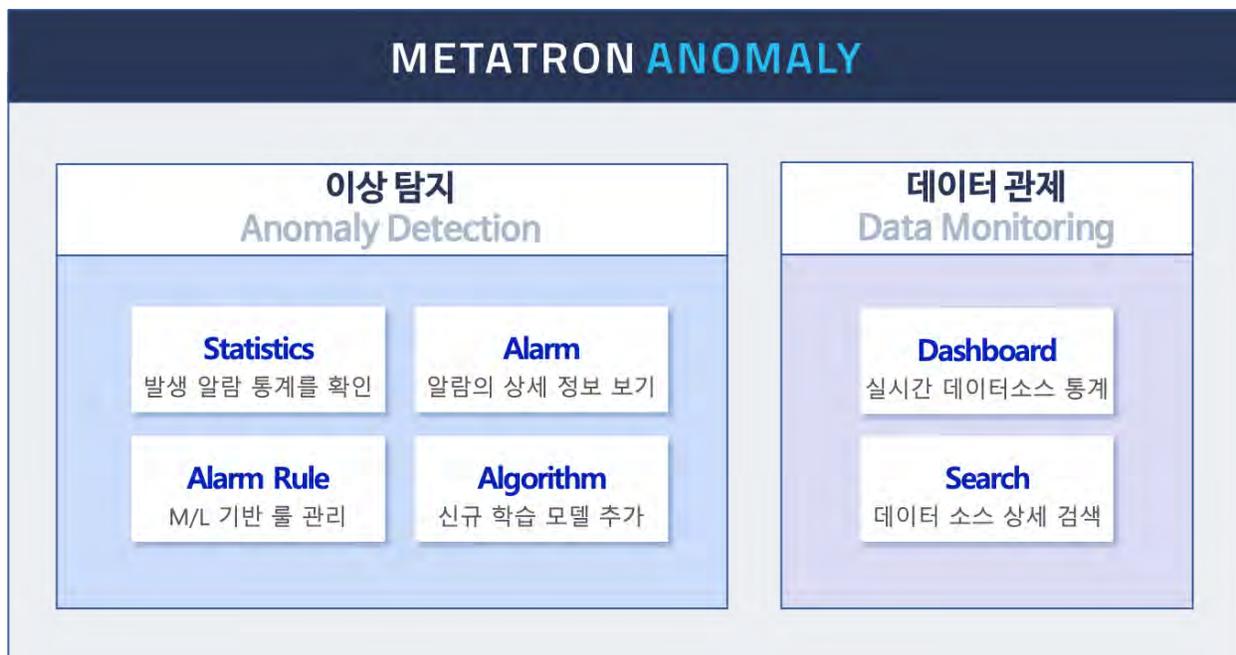
17.2 주요 기능

Anomaly의 주요 기능은 다음과 같습니다.

- **Auto Machine Learning** : 머신러닝 기반 예측 모델을 자동으로 추천하여 사용자 편의성 증진
- **Alarm & Report** : 비정상적인 상황 발생 시 즉각 알람 발생 및 보고서 생성 지원
- **Analyze** : 데이터 원천을 분석할 수 있는 실시간 대시보드 및 실시간 검색 기능 지원
- **Link with Learning System** : 신규 알고리즘 모델을 적용할 수 있도록 3rd-party 시스템 연계를 지원

17.3 구조

Anomaly의 메뉴 구성은 다음과 같이 크게 이상 탐지와 데이터 관제 두 개의 카테고리로 나누어 집니다.



Anomaly Detection 메뉴에서는 전반적인 이상 탐지 알람 통계 (Statistics), 발생한 알람들의 정보 (Alarm), 알람이 발생하는 규칙 설정 (Alarm Rule), 신규 알고리즘 추가 (Algorithm) 를 지원합니다.

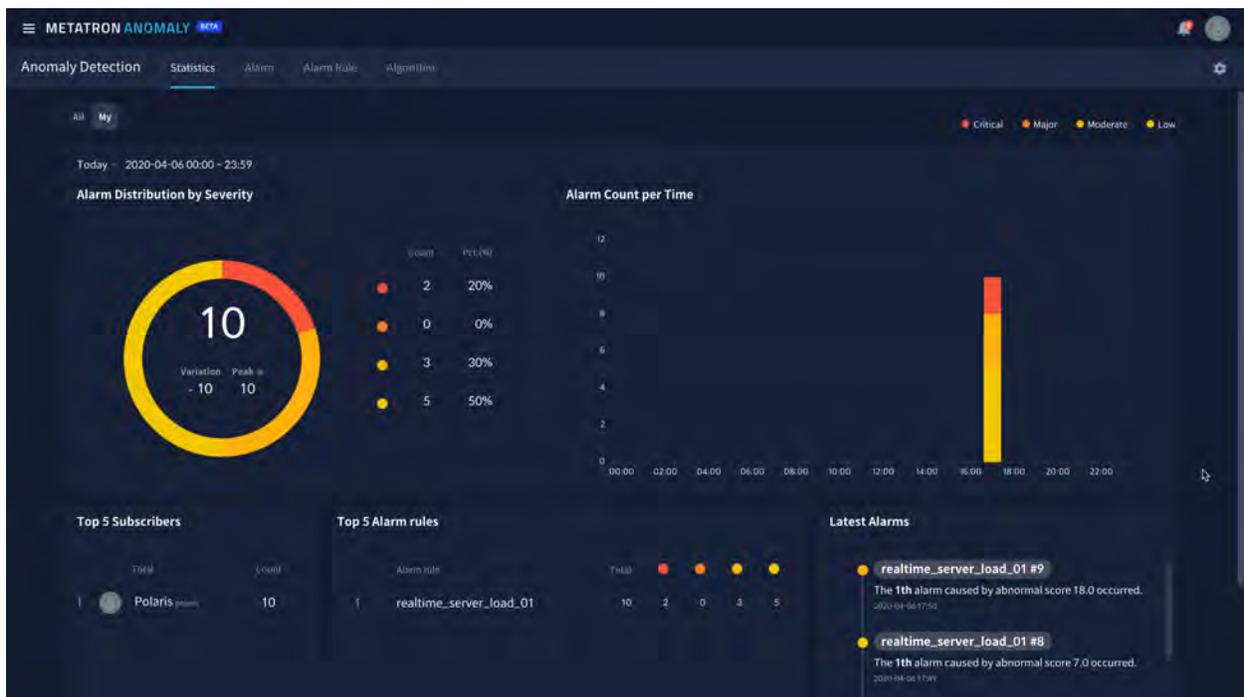
Data Monitoring 메뉴에서는 알람이 발생할 경우 데이터 소스 원천에 대해서 분석할 수 있도록 실시간 대시보드 (Dashboard) 와 원천을 쿼리할 수 있도록 하는 실시간 검색 (Search) 기능을 제공합니다.

주요 메뉴 간 이동이 간편하고 세부 항목 간 참조 기능이 구축되어 있어 발생한 알람 내역 및 설정된 알람 룰, 그리고 전반적인 알람 현황 간의 유기적인 파악이 용이합니다. 또한 알람 발생 시 동일한 시스템 내에서 원천을 탐색할 수 있는 기능도 있어서 원인 파악에 더욱 용이합니다.

통계

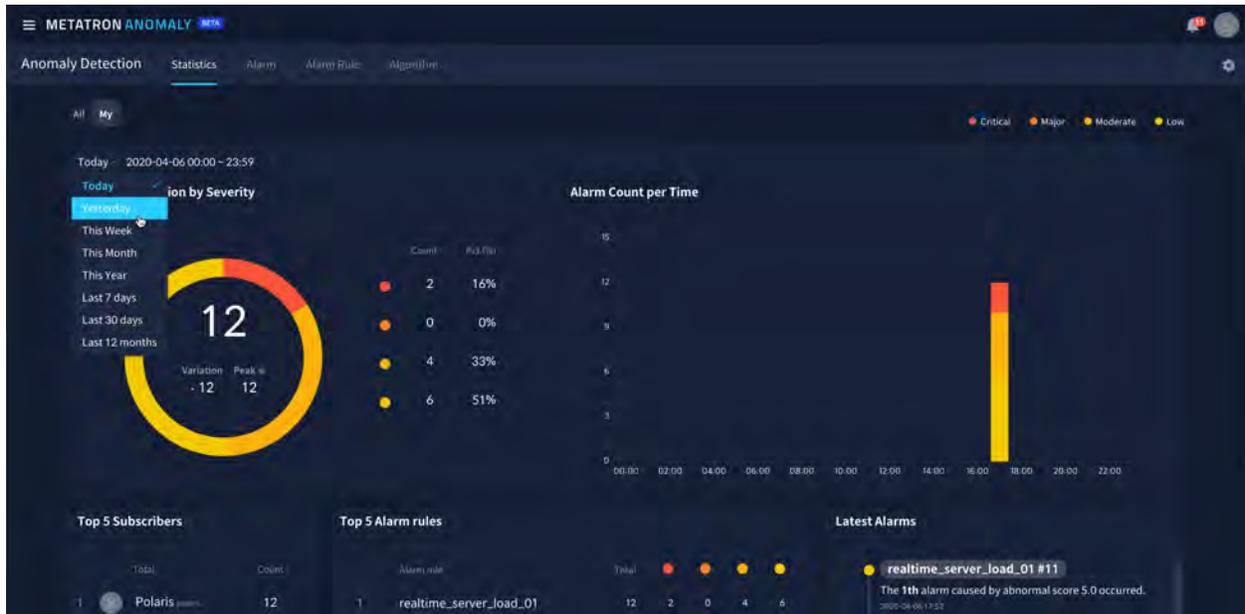
Statistics 탭 메뉴에서는 발생한 알람의 전반적인 통계를 보여줍니다. 이 페이지에서는 사용자가 지금까지 발생한 알람의 현황을 다각도로 파악할 수 있도록 중요도, 알람 발생 시기, 알람 룰 등의 다양한 기준으로 통계를 산출하여 제시합니다.

페이지 기본 구성은 다음과 같습니다.



- **Alarm Distribution by Severity:** 심각도별 알람 발생 비중을 보여줍니다.
- **Alarm Count per Time:** 시간대별 알람 빈도를 보여줍니다.
- **Top 5 Subscribers:** 가장 많은 알람을 통보받은 사용자 5명을 보여줍니다.
- **Top 5 Alarm rules:** 가장 많은 알람을 일으킨 알람 룰 5개를 보여줍니다.
- **Latest Alarms:** 가장 최근에 발생한 알람들을 보여줍니다.

페이지 상단의 기간 설정 메뉴를 이용하면 통계를 산출하는 기준 기간을 변경할 수 있습니다.



알람 내역 열람하기

Alarm 탭 메뉴에서는 지금까지 발생한 알람 내역을 확인할 수 있습니다. 알람의 전체적인 현황을 보여주는 통계 페이지와는 다르게 이 메뉴에서는 보다 개별적인 알람들을 열람하고 탐색하는 데 최적화된 UI를 제공합니다.

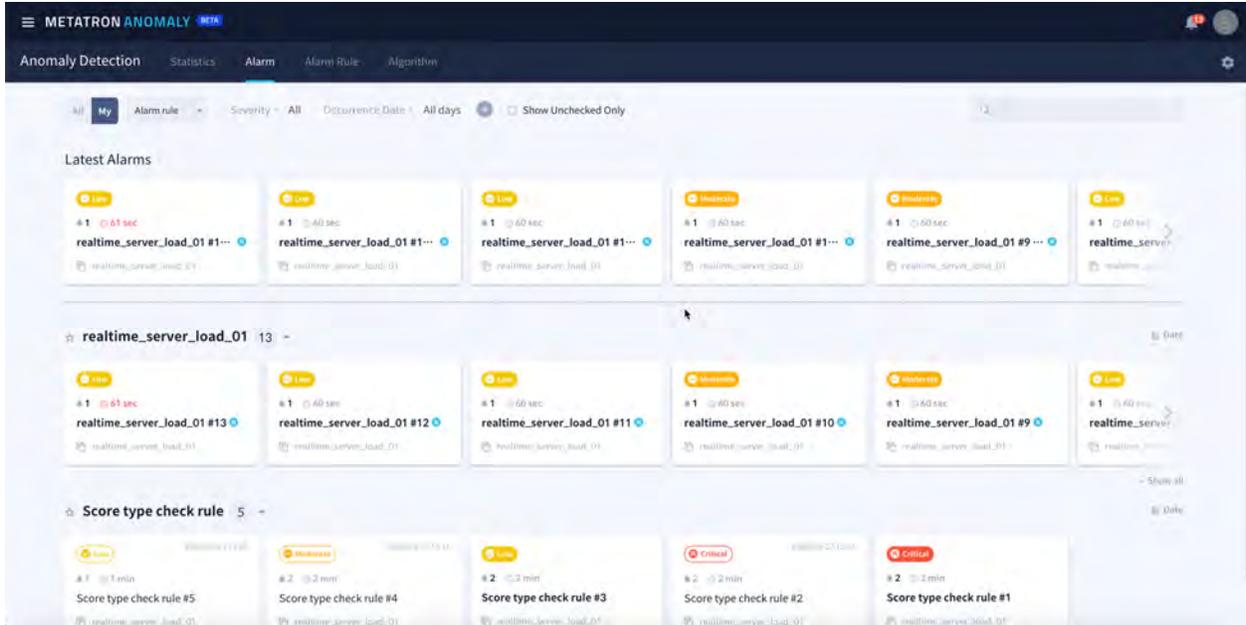
이 메뉴는 다음의 두 가지 페이지로 구성되어 있습니다.

- 알람 리스트
- 알람 상세

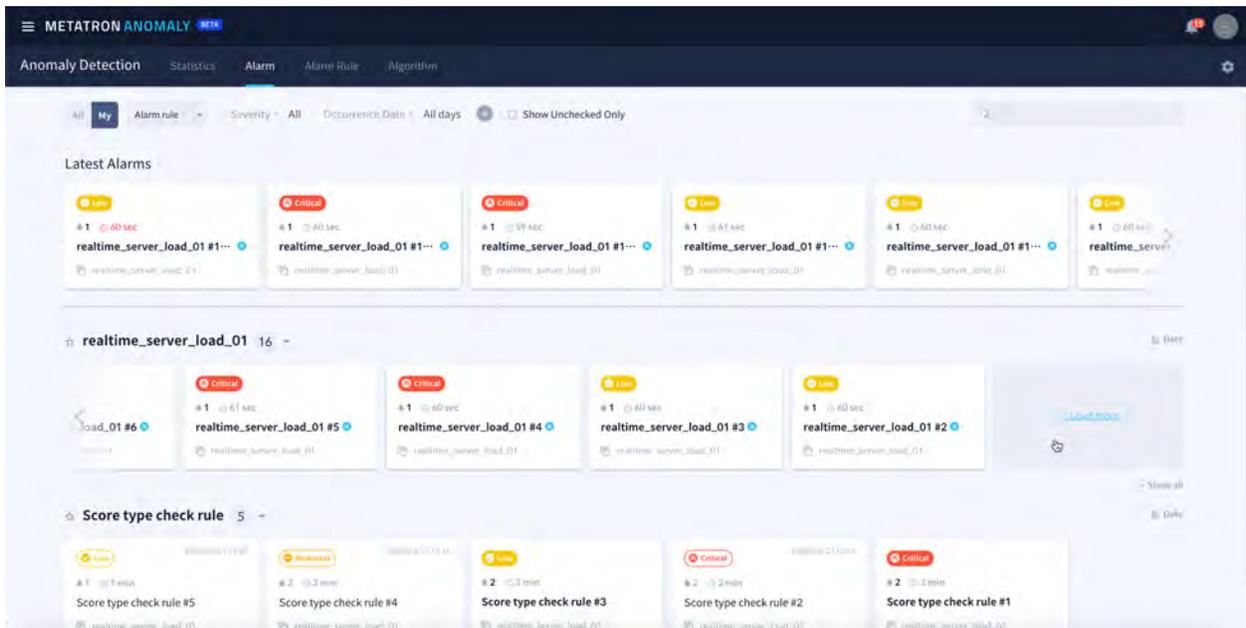
19.1 알람 리스트

Alarm 탭으로 들어가면 현재까지 발생한 알람들을 열거하여 보여줍니다. 화면 상단에 있는 Alarm rule / Timeline 선택 박스를 이용하여, 알람 리스트를 알람 룰 기준으로 정렬할 수도 있고, 발생한 시간 기준으로 정렬할 수도 있습니다.

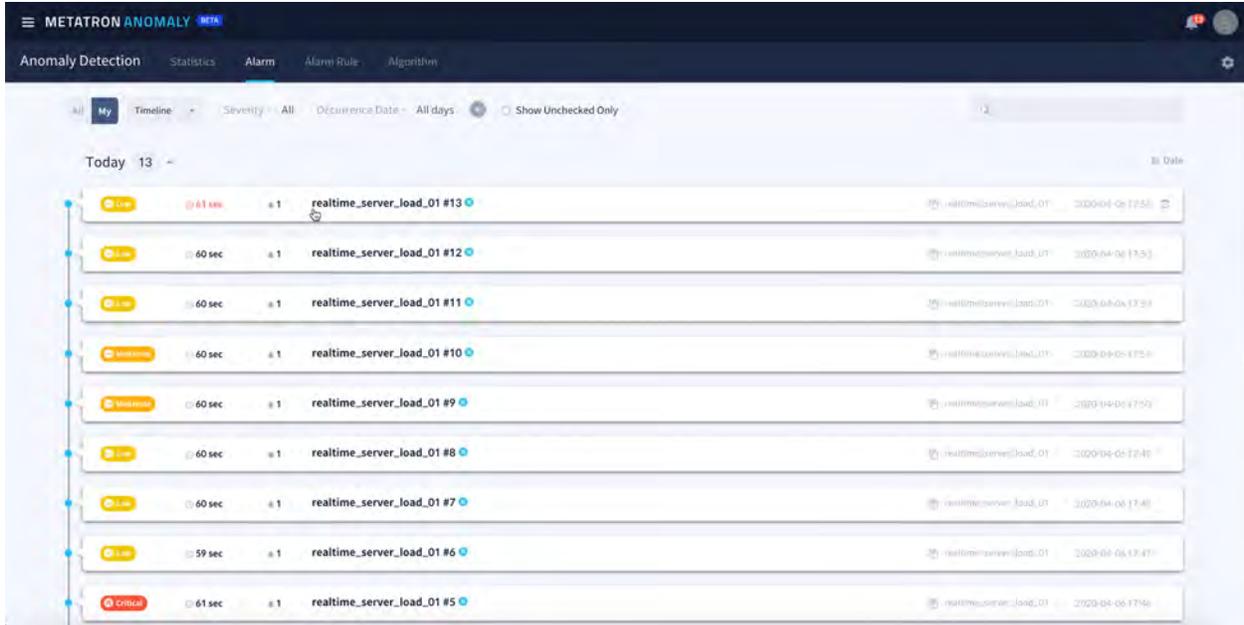
- Alarm rule (알람 룰 기준으로 정렬)



- Timeline (발생 시간 기준으로 정렬)



카테고리 맨 끝에 있는 + Load more를 클릭하면 해당 카테고리 내 더 많은 알람 항목을 보여줍니다.



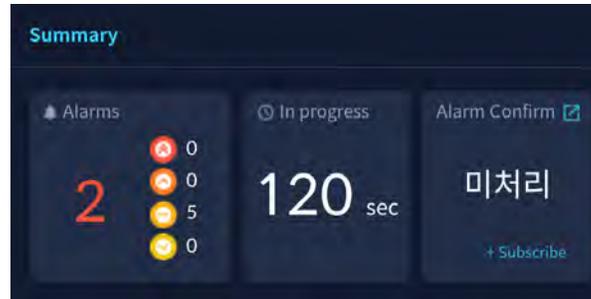
19.2 알람 상세

알람 리스트에 열거된 항목 중 하나를 선택하면 해당 알람에 대한 상세 정보를 열람할 수 있습니다. 아래는 알람 상세 페이지의 각 영역별 설명입니다.

19.2.1 Summary

이 영역에서는 해당 알람의 발생 현황을 보여줍니다. 정해진 주기에 따라 알람이 연속적으로 발생하면 1개의 알람 항목으로 계속 유지되며, 알람의 심각도 (severity) 기준을 넘은 데이터 포인트 수가 함께 표기됩니다. 또한 알람을 확인한 후 처리 결과를 기록할 수 있도록 링크를 제공합니다. 내가 생성한 알람 룰이 아닌 경우 **Subscribe**를 눌러 해당 알람 룰로 추후 발생한 알람들에 대해 알림을 받을 수 있습니다.

아래 그림 예시에서는 알람이 2번 연속적으로 발생하였고 (Alarms), 알람 확인 주기가 1분이기 때문에 2건의 알람이 총 120초 동안 지속된 것을 보여주고 있습니다. (Elapsed Time).



19.2.2 Alarm History 영역

이 영역에서는 해당 알람에 적용된 알람 룰에 의해 발생한 알람의 이력을 보여줍니다. 우측 링크 아이콘을 누르면 해당 알람으로 이동합니다.

Severity 모든타입 There are 15 items

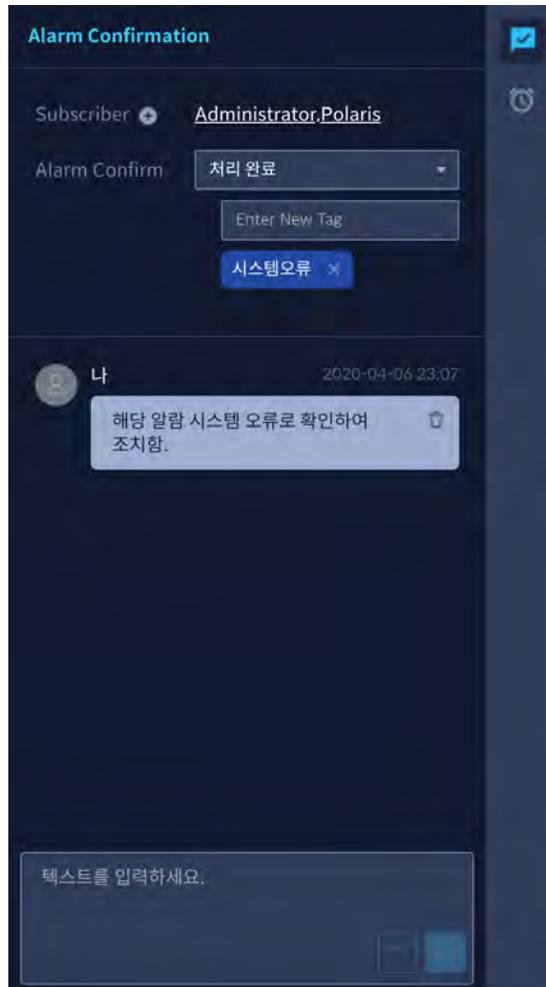
NO	Occurrence time	Alarm interval	Alarm	
211	2020-04-06 23:26 60 sec	1	1	🔗
210	2020-04-06 23:25 60 sec	1	1	🔗
209	2020-04-06 23:24 60 sec	1	1	🔗
208	2020-04-06 23:23 120 sec	1	2	🔗
207	2020-04-06 23:21 60 sec	1	1	🔗
206	2020-04-06 23:20 60 sec	1	1	🔗
205	2020-04-06 23:19 60 sec	1	1	🔗

19.2.3 Alarm Confirmation 탭

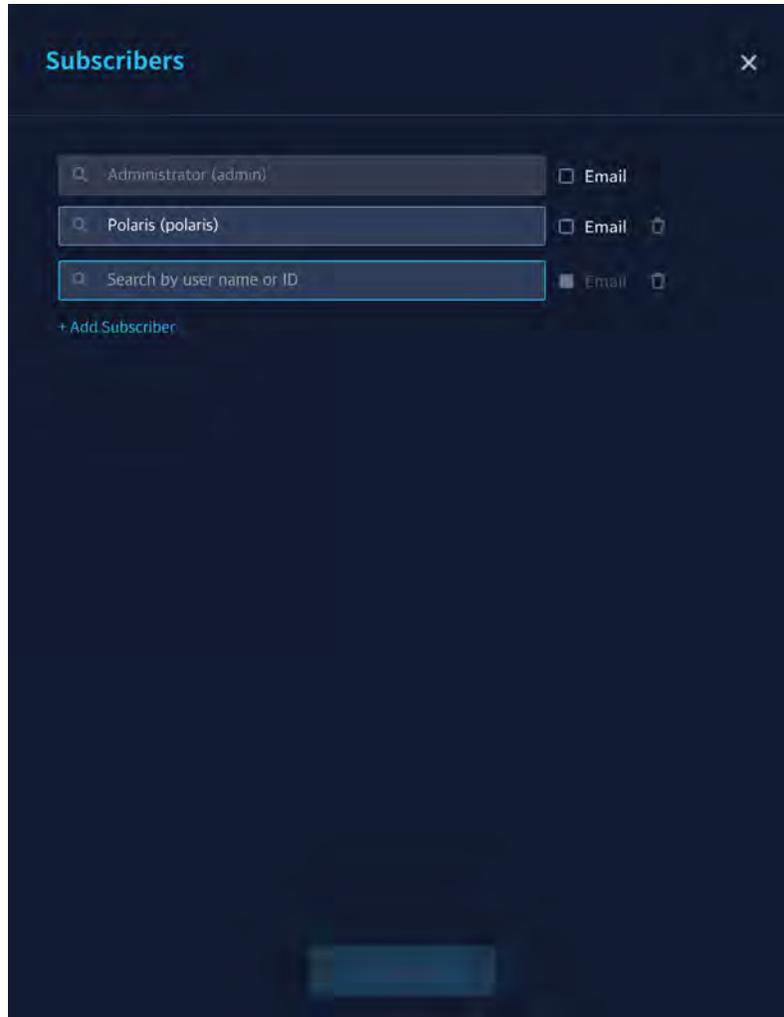
우측 탭 첫번째 메뉴에서는 알람 확인 후 해당 알람 구독자 리스트를 확인하고 (Subscriber) 알람을 확인하여 상태를 기록하고 (Alarm Confirm) 작업자가 기록을 남길 수 있는 커뮤니케이션 기능을 제공합니다.

알람 확인 항목은 4가지로 구분됩니다.

- **미처리:** 알람 최초 발생 시 기본값. 해당 알람에 대해 어떠한 조치도 취하지 않은 상태
- **알림 중지:** 해당 알람을 사용자가 확인하여 더이상 알림 (notification) 을 받지 않는 상태
- **처리 완료:** 해당 알람을 확인하고 조치를 취한 상태로, 해당 알람에 관련된 tag 기록 가능
- **오탐:** 이상 상태가 아닌데 발생한 알람



구독자 (Subscriber) 는 해당 알람에 관계된 유저를 아이디로 검색하여 추가할 수 있으며, E-mail에 체크하면 해당 유저 정보에 기록된 이메일로 알람을 발송합니다.

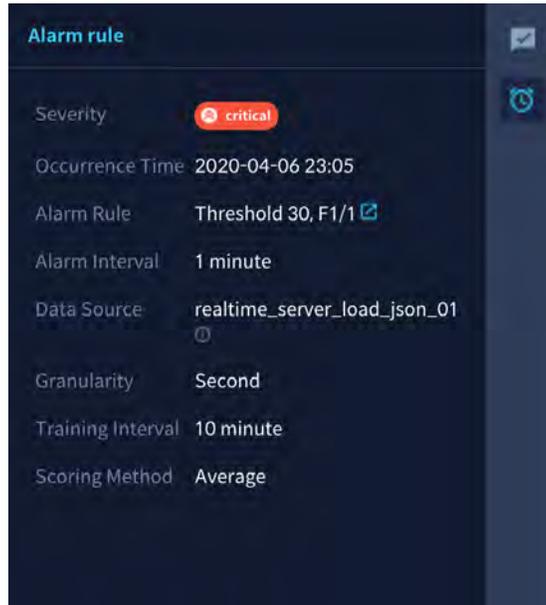


19.2.4 Alarm Rule 탭

이 영역에서는 해당 알람의 심각도와 알람 발생 시각, 그리고 이 알람을 발생시킨 룰과 데이터 소스에 관련된 정보를 보여줍니다.

- **Severity:** 현재 발생한 알람의 심각도
- **Occurrence Time:** 알람 발생 시각
- **Alarm Rule:** 알람을 발생시킨 임계치와 임계치 초과 건수/알람 발생 검사 주기. 우측 링크 버튼 클릭 시 해당 알람 룰로 이동
- **Alarm Interval:** 알람 발생 검사 주기. 1분일 경우 1분마다 Abnormal score가 임계치를 넘었는지 검사

- **Data Source:** 데이터 소스 정보
- **Granularity:** 데이터 소스가 적재되는 시간 단위
- **Training Interval:** 모델 학습을 위해 사용한 데이터 기간
- **Scoring Method:** 여러 개의 측정값 (Measure) 을 사용할 경우 Abnormal Score를 계산하는 방식



19.2.5 View by Chart 탭

이 탭 영역에서는 해당 알람 구간에서 모니터링한 데이터의 Abnormal Score 를 그래프로 보여줍니다. 각 조건별 점수 임계치 (Threshold) 에 상응하는 알람 (Critical, Major, Moderate, Low) 별로 발생된 알람의 건수를 확인할 수 있습니다. 차트 산출 방식에 관해서는 [기본 원리](#) 항목을 참조하십시오.



- **Total abnormal score:** 알람 룰에 포함된 모든 측정값 컬럼에 대한 Abnormal Score를 보여줍니다.
- **Chart by measures:** 알람 룰에 포함된 각 개별 측정값 컬럼 데이터의 예측치와 실제치의 추이를 보여줍니다.

19.2.6 View by Table 탭

이 탭 영역에서는 각 알람 발생 건별로 데이터 실제치와 예측치, 그리고 Abnormal Score를 표 형식으로 나열합니다.

The screenshot shows the 'View by Table' tab of the dashboard. It contains a table with the following data:

	Occurrence time	Total Abnormal Score	SUM_cpu (Weight Value: 73%)			SUM_memory (Weight Value: 132%)		
			Actual	Predict	Score	Actual	Predict	Score
1	2020-04-06 23:05:00	35	2,163	1,156	64	1,981	2,239	6
2	2020-04-06 23:05:53	5	2,124	2,160	2	4,049	4,425	8
3	2020-04-06 23:05:57	6	2,159	2,149	1	2,691	3,196	11

Below the table, there is a '+ Load More' link.

Metatron Anomaly는 사용자가 직접 데이터에서 이상을 탐지하는 규칙을 쉽게 생성하고 관리할 수 있도록 지원합니다. Anomaly의 알람 룰은 다음과 같은 특징이 있습니다.

- 오류 이력 없는 모든 실시간 데이터에 대한 비지도 학습 기반의 머신러닝 지원
- 3 step으로 이루어진 쉽고 빠른 알람 룰 생성
- 통계 기반의 7종 예측 모델 기본 제공
- 모델 자동 학습 및 최적의 모델 추천
- 적용된 모델 정확도 하락 시 모델 재학습 (re-learning) 지원

본 단원의 구성은 아래와 같습니다.

20.1 알람 룰 만들기

Anomaly는 다음의 절차를 순차적으로 수행하도록 안내하여 사용자가 원하는 알람 룰을 쉽게 생성할 수 있도록 지원합니다.

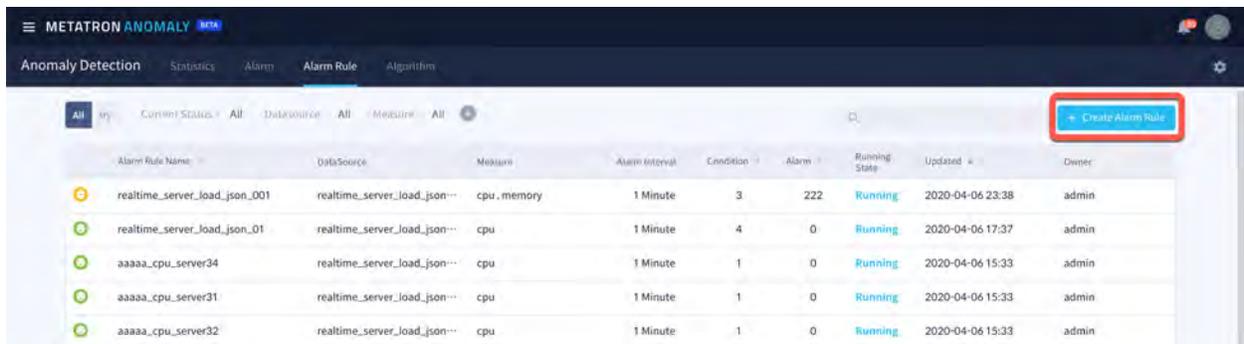
- 데이터 소스 선정
- 모니터링할 지표 선택하기

- 트레이닝 기간 설정하기
- 모델 선택하기
- 알람 룰 조건 설정하기
- 알람 룰 완성하기

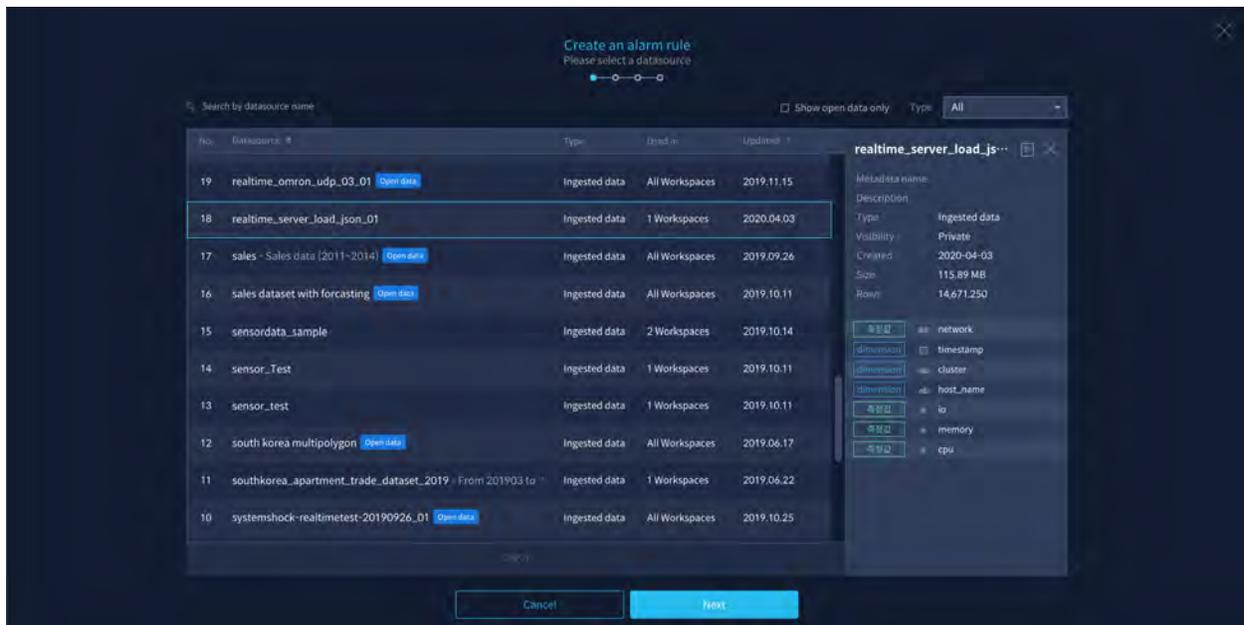
20.1.1 데이터 소스 선정

알람 룰을 생성하기 위해서는 가장 먼저 모니터링할 데이터 소스를 선정해야 합니다.

1. Alarm Rule 우측 상단에 있는 **Create Alarm Rule** 버튼을 클릭합니다.



2. 모니터링하고자 하는 데이터 소스를 선택합니다.



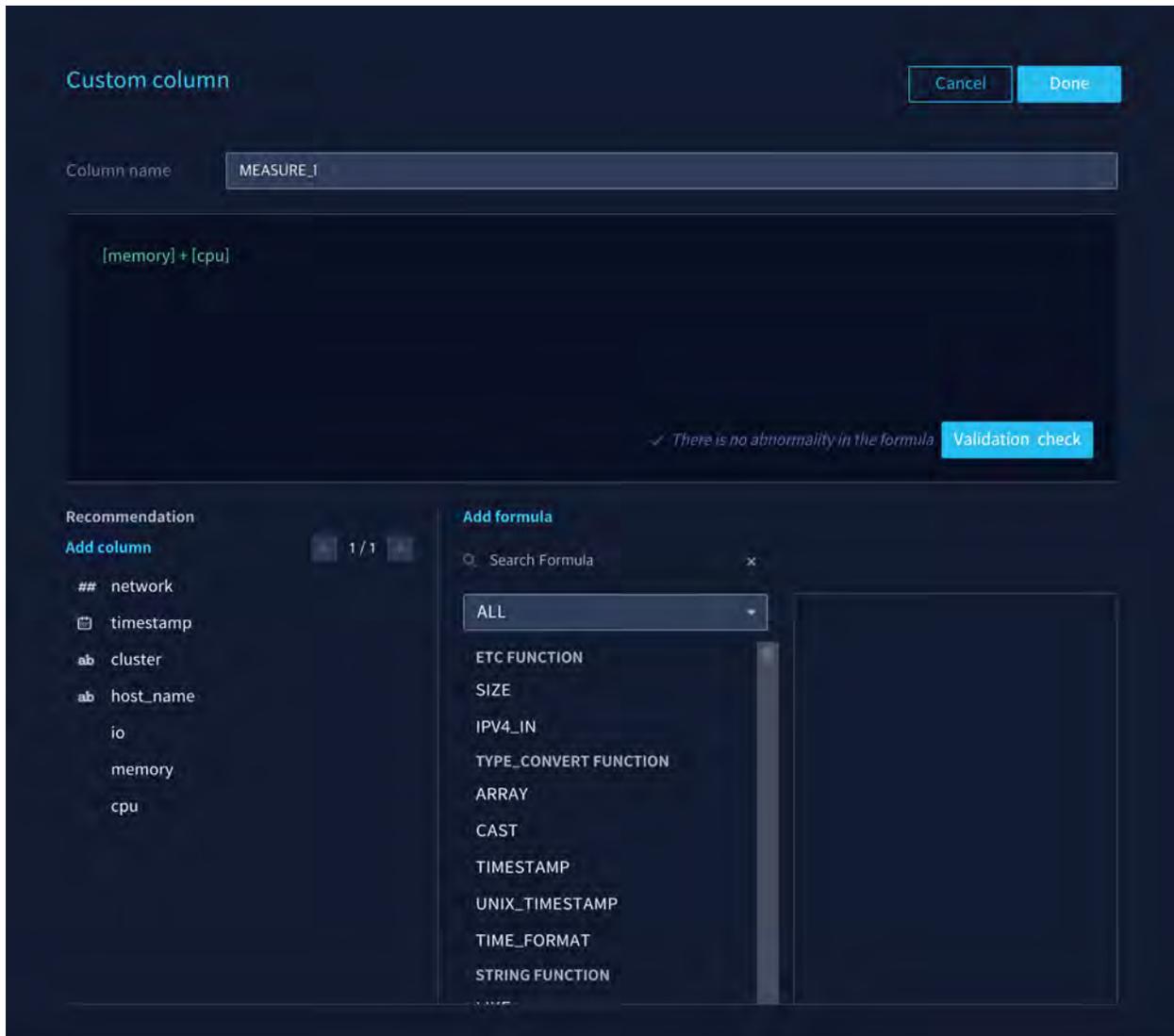
20.1.2 모니터링할 지표 선택하기

데이터 소스를 선택하면 다음 화면으로 넘어가면서 좌측에 **Data** 패널이 열립니다. 이 패널에서 아래와 같이 모니터링할 지표를 선택할 수 있습니다.

1. **Measure 선택:** Measure 영역에서 알람을 설정하고자 하는 측정값 컬럼을 선택합니다. 클릭한 측정값 컬럼은 Aggregate 선반에 자동으로 옮겨집니다.



2. **사용자 컬럼 추가:** 필요한 경우 기존 컬럼에 수식을 적용하여 새로운 컬럼을 만들 수도 있습니다. Measure 영역의 우측 상단에서  버튼을 클릭하여 대화 상자를 열고 직접 새로운 컬럼을 만들어 보세요.



3. **Measure 집계 방식 변경:** Aggregate 선반에 올려진 각 컬럼을 클릭하여 원하는 집계 방식을 선택합니다. 기본값은 SUM으로 지정되어 있습니다.

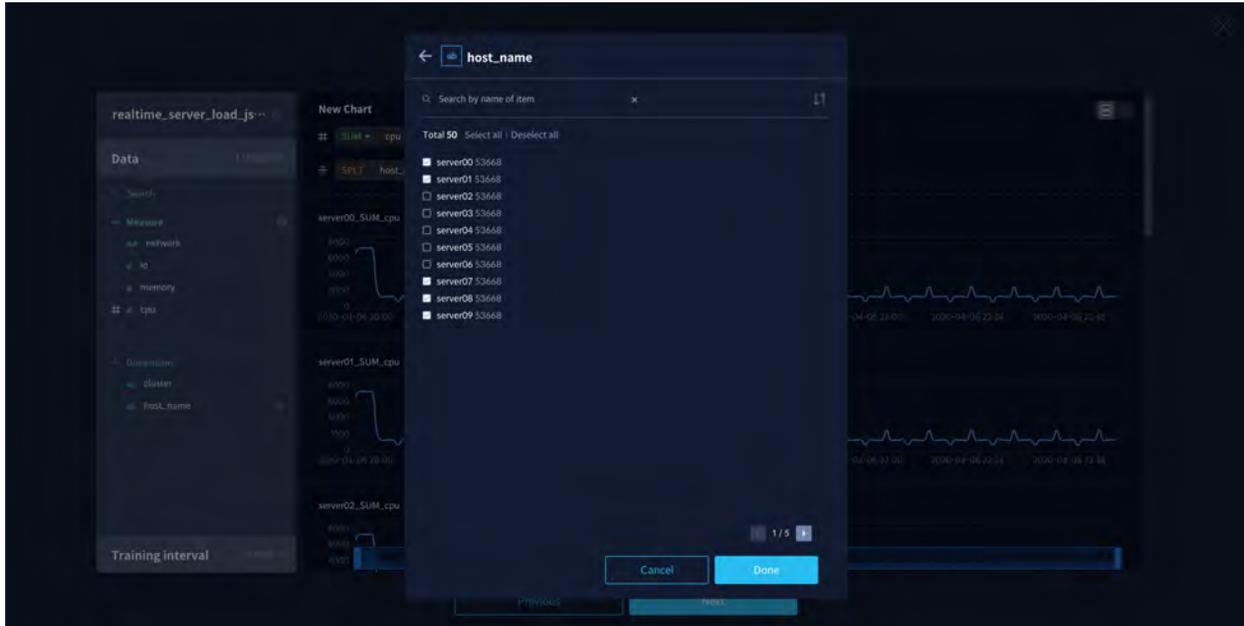


4. **Split:** 필요할 경우 차원값 컬럼을 기준으로 aggregate 데이터를 분할할 수 있습니다. **Dimension** 영역에서 분할의 기준으로 삼을 측정값 컬럼에 마우스 커서를 오버한 후  버튼을 클릭하세요. split 가능한 개수는 최대 10 개이며, dimension 값이 10개 이상이라면 임의의 값 10개가 선택됩니다.



5. **Dimension 값으로 필터링:** 필요할 경우 차원값 (Dimension) 컬럼을 기준으로 데이터를 필터링할 수 있습니다.

Dimension 영역에서 필터를 설정할 측정값 컬럼에 마우스 커서를 오버한 후  버튼을 클릭하세요. 그 후 아래 화면처럼 모니터링이 필요한 특정 범주를 선택하세요.



20.1.3 트레이닝 기간 설정하기

모니터링할 지표 선택을 마쳤으면 **Training interval** 패널에서 예측 모델 트레이닝에 사용할 데이터 범위를 지정할 수 있습니다.

1. 모델 학습에 사용할 데이터의 기본 시간 단위를 **Granularity** 메뉴에서 결정할 수 있습니다. 그래프를 보면서 데이터의 패턴을 가장 잘 보여주는 형태의 단위를 선택합니다.



2. 모델 학습에 사용할 데이터의 범위를 설정합니다. 위에 설정한 기본 Granularity보다 같거나 큰 단위로 학습할 데이터 범위를 입력할 수 있습니다.

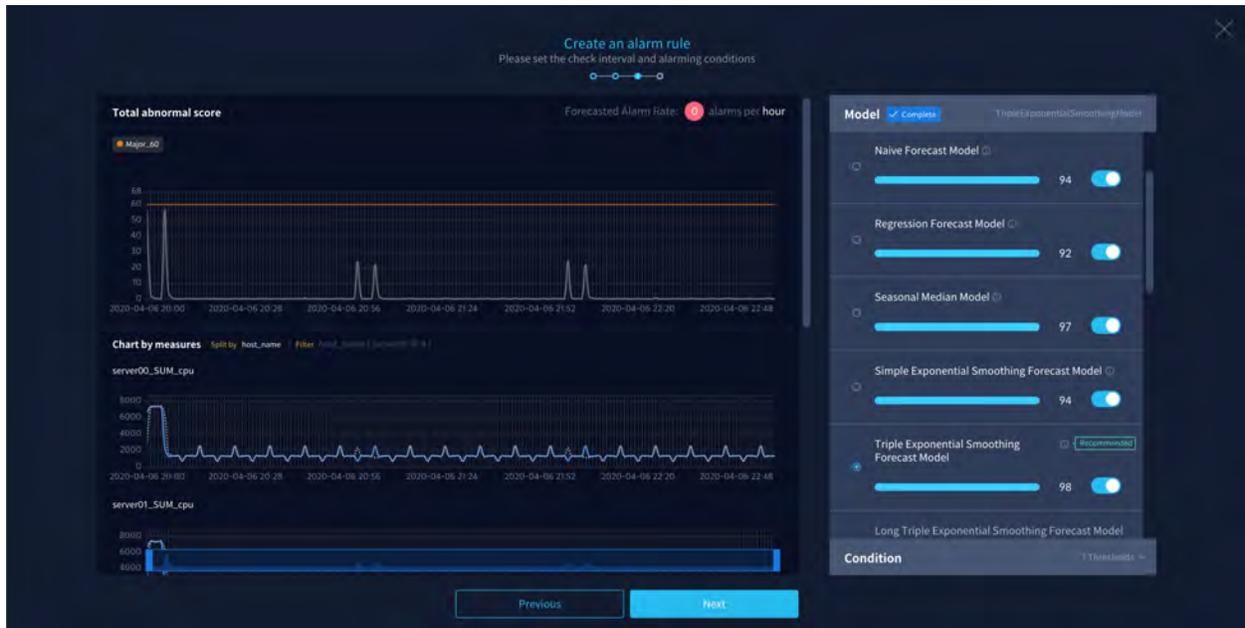


3. 모든 설정을 마쳤으면 **Next**를 클릭합니다.

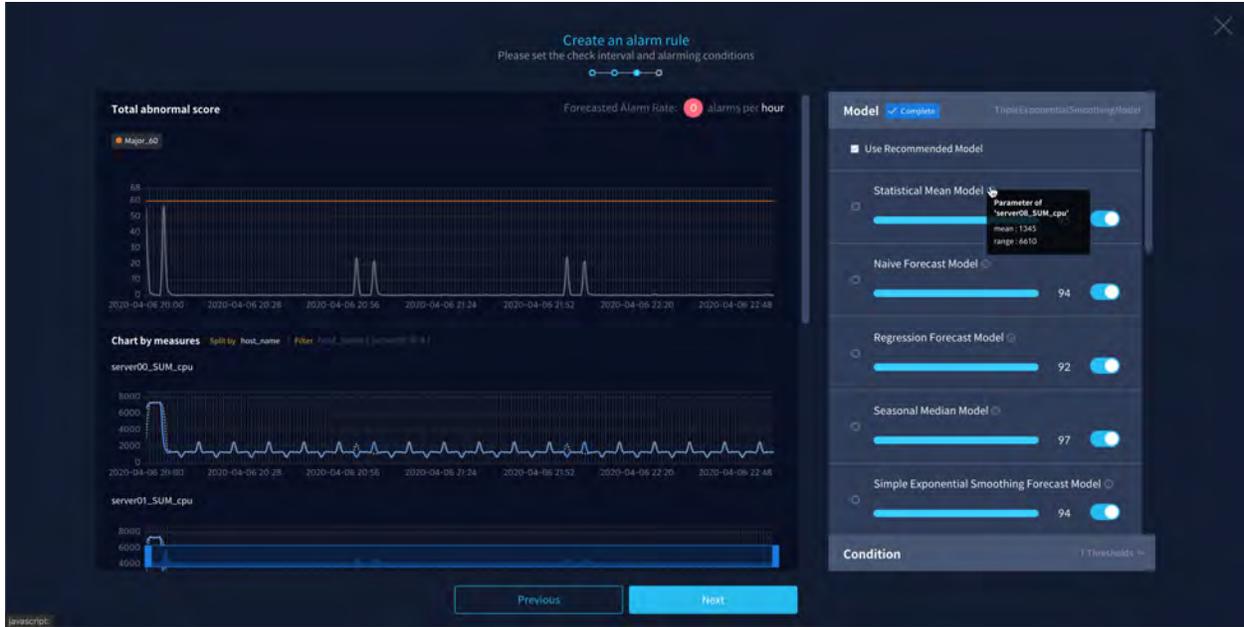
20.1.4 모델 선택하기

이제 **Model** 패널로 넘어가서 어떤 예측 모델을 사용할지 선택합니다. Anomaly는 앞서 설정한 학습 데이터 기간으로 각각의 모델을 트레이닝시킨 후 정확도를 계산합니다. 아래 두 방법 중 하나를 통해 적합한 예측 모델을 선택할 수 있습니다.

- **추천 모델 사용:** 기본적으로 우측에 표시되는 정확도 점수 (100점 만점) 가 가장 높은 모델이 **Recommend** 태그와 함께 자동 선택됩니다.



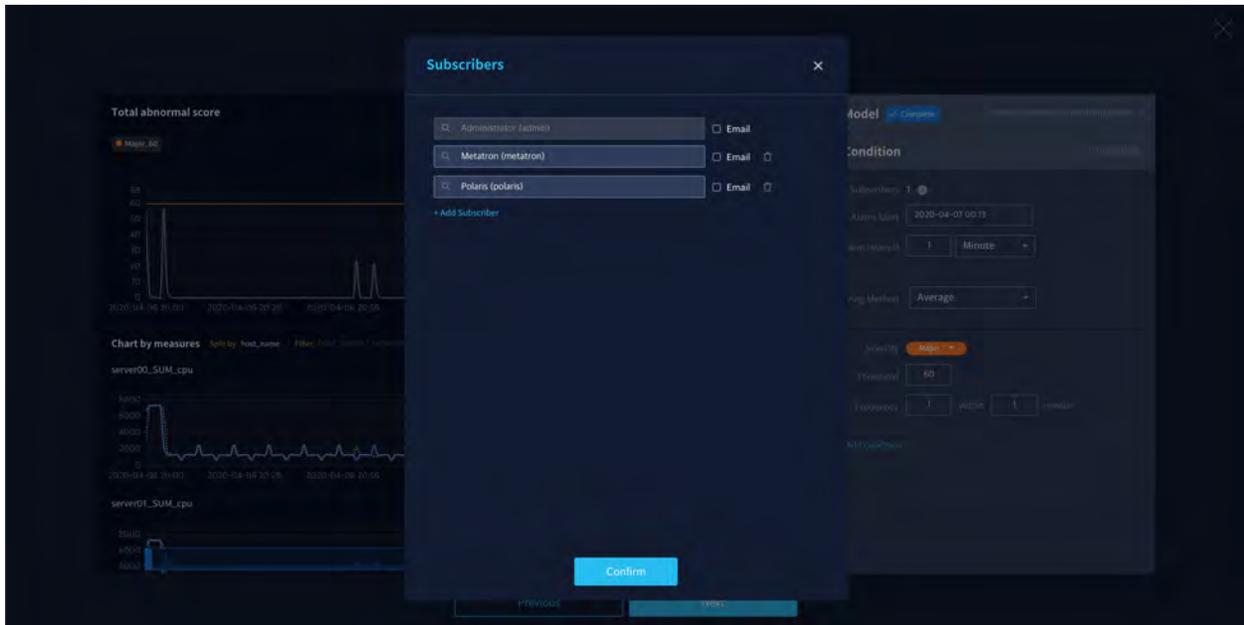
- **비교 후 직접 선택:** 각 모델을 선택하면 예측값과 Abnormal Score를 그래프에서 볼 수 있습니다. 가장 적합하다고 생각되는 모델을 직접 선택할 수 있습니다. 모델명 우측 아이콘에 마우스를 오버하면 해당 모델이 학습된 상세 값을 볼 수 있습니다.



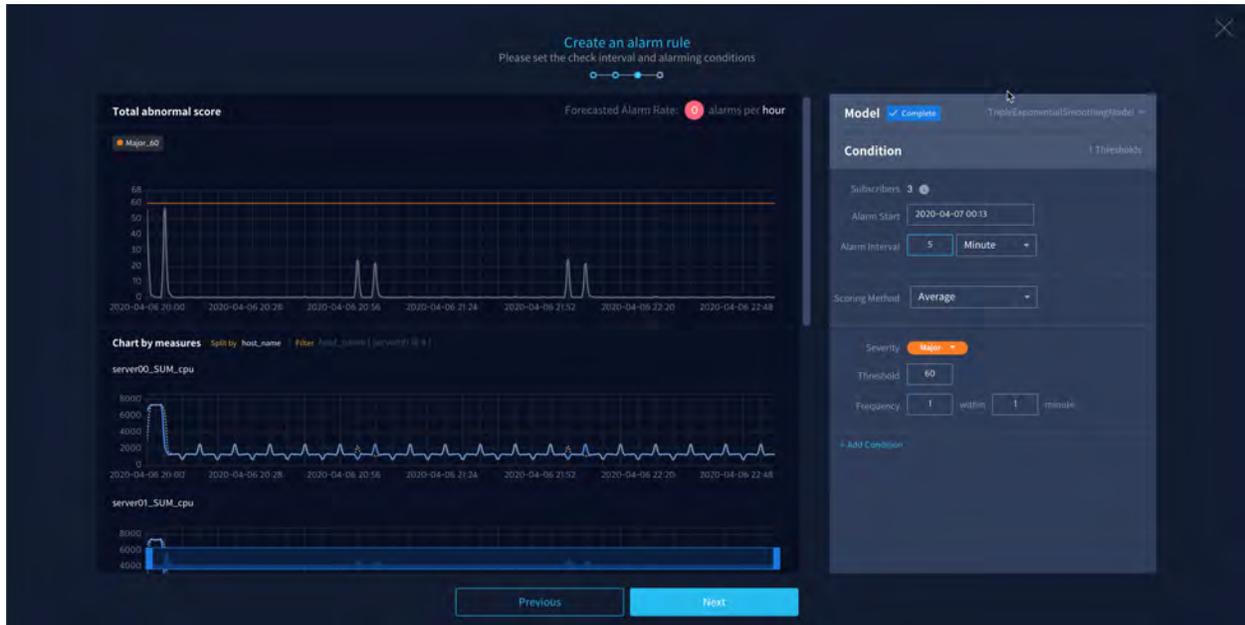
20.1.5 알람 룰 조건 설정하기

사용할 예측 모델을 선택하였으면, **Condition** 패널에서 알람이 발생하는 조건을 설정할 수 있습니다.

1. **Subscribers** 항목의 우측에 있는 ⓘ 버튼을 클릭하여 대화 상자를 연 후, 알람 발생 시 통보 받는 대상과 통보 방식을 설정합니다.

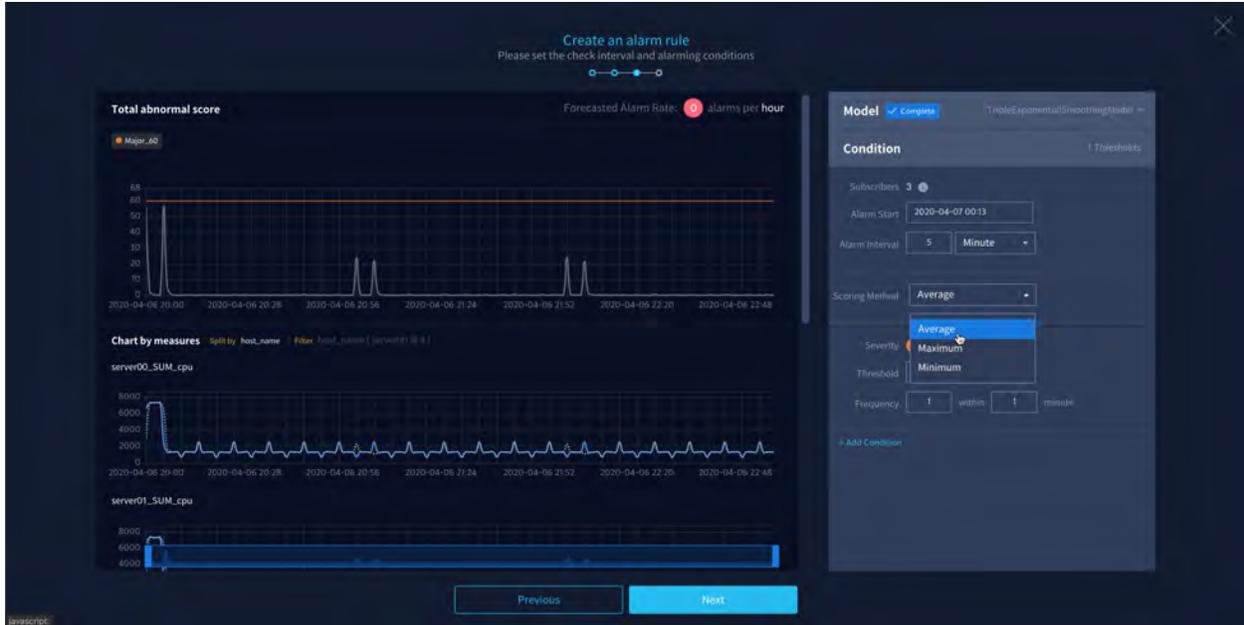


2. 아래 각 항목의 설명을 참고하여 알람 발생 조건을 설정합니다.



- **Alarm Start:** 알람 조건 검사를 시작할 순간을 설정합니다. 이 설정값에 해당하는 시간 이후부터 알람 발생을 검사합니다.
- **Alarm Interval:** 알람 발생 조건을 검사하는 주기를 설정합니다.

3. **Scoring Method**는 여러 개의 측정값 (Measure) 에서 Abnormal Score를 계산하는 방식을 결정합니다. 기본 값은 각 측정값에서 계산된 Abnormal Score들의 평균값 (Average) 으로 계산하며, 최대값 (Maximum) 또는 최소값 (Minimum) 으로 변경할 수 있습니다.



4. 아래 각 항목의 설명을 참고하여 모니터링 대상 데이터의 abnormal score에 따른 알람 발동 조건을 설정합니다. 기본적으로 하나의 조건이 주어지며, + Add Condition 버튼을 클릭하면 조건을 추가할 수 있습니다.



- **Severity:** 주어진 조건에 해당하는 알람의 심각도를 설정합니다.
- **Threshold:** Abnormal Score가 이 설정값을 초과하면 데이터 이상 상태로 간주됩니다.
- **Frequency:** Abnormal Score가 한계값을 초과하는 빈도가 어떠한 때 알람을 발생시킬지 결정

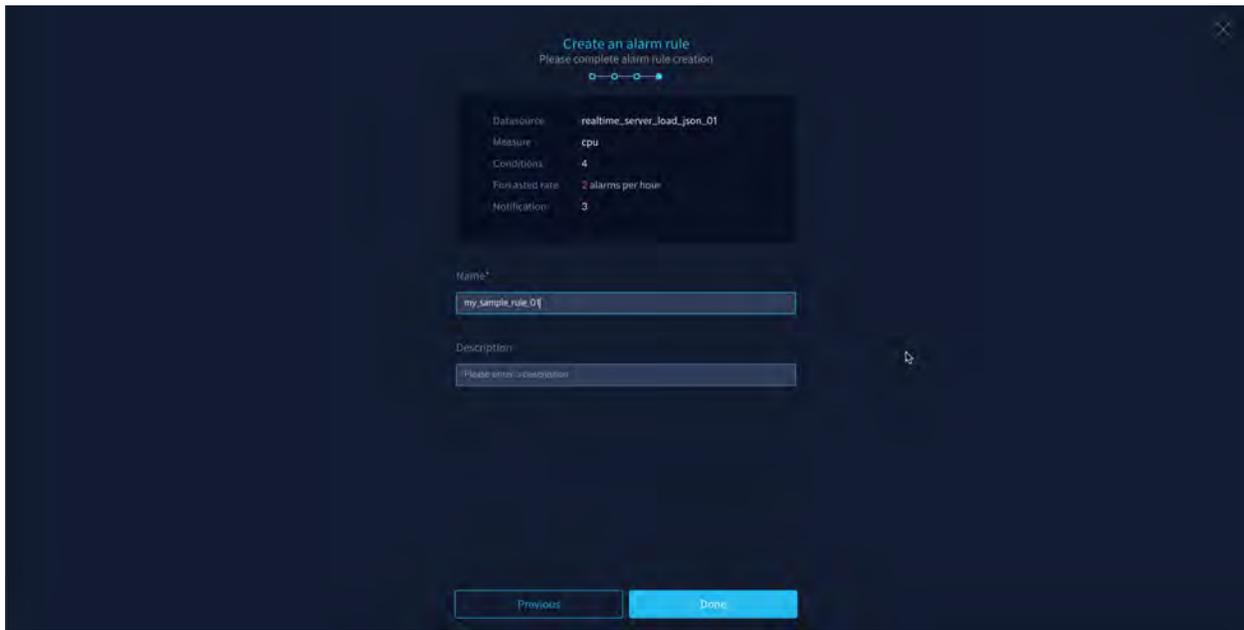
합니다. 예를 들어, 《3 within 5 minute》로 설정한 경우에는 abnormal score가 5분 안에 3회 이상 한계값을 초과하면 알람이 발생합니다.

5. 모든 설정을 마쳤으면 **Next**를 클릭합니다.

20.1.6 알람 룰 완성하기

알람 룰 설정이 끝났으면 아래와 같이 알람 룰 만들기 절차를 마무리합니다.

1. 알람 룰의 이름과 설명을 기입한 후 **Done** 버튼을 클릭합니다.



2. 생성된 알람 룰은 알람 룰 리스트의 최상단에 노출되며 실행 중인 상태 (Running) 로 나타납니다.



20.2 알람 룰 내역 열람 · 수정하기

Alarm Rule 탭 메뉴에서는 등록된 알람 룰을 열람 · 수정할 수 있습니다. 또한 이 메뉴에서는 선택한 예측 모델에 따라 산출되고 있는 데이터 abnormal score 현황도 쉽게 파악할 수 있습니다.

알람 룰 메뉴는 다음의 두 가지 페이지로 구성되어 있습니다.

- 알람 룰 리스트
- 알람 룰 상세

20.2.1 알람 룰 리스트

Alarm Rule 탭으로 들어가면 현재 등록된 알람 룰들을 열거하여 보여줍니다.

Alarm Rule Name	DataSource	Measure	Alarm Interval	Condition	Alarm	Running State	Updated	Owner
my_sample_rule_01	realtime_server_load_json...	cpu	5 Minute	4	0	Running	2020-04-07 00:28	admin
realtime_server_load_json_001	realtime_server_load_json...	cpu.memory	1 Minute	3	272	Running	2020-04-07 00:28	admin
realtime_server_load_json_01	realtime_server_load_json...	cpu	1 Minute	4	0	Running	2020-04-06 17:37	admin

리스트에 표시되는 정보는 아래와 같으며, 이를 기준으로 열거할 룰을 필터링하거나 검색할 수 있습니다.

- **Current Status:** 해당 룰에 따른 모니터링 결과 상태
- **Alarm Rule Name:** 해당 룰의 이름
- **DataSource:** 모니터링 대상 데이터 소스
- **Measure:** 모니터링 대상 측정값 컬럼
- **Alarm Interval:** 알람 발생 주기
- **Condition:** 해당 룰에 적용된 알람 발생 조건의 개수
- **Alarm:** 해당 룰에 의해 발생한 알람의 수
- **Running:** 해당 룰의 모니터링 활성화 여부
- **Updated:** 해당 룰을 마지막으로 업데이트한 시간과 사용자

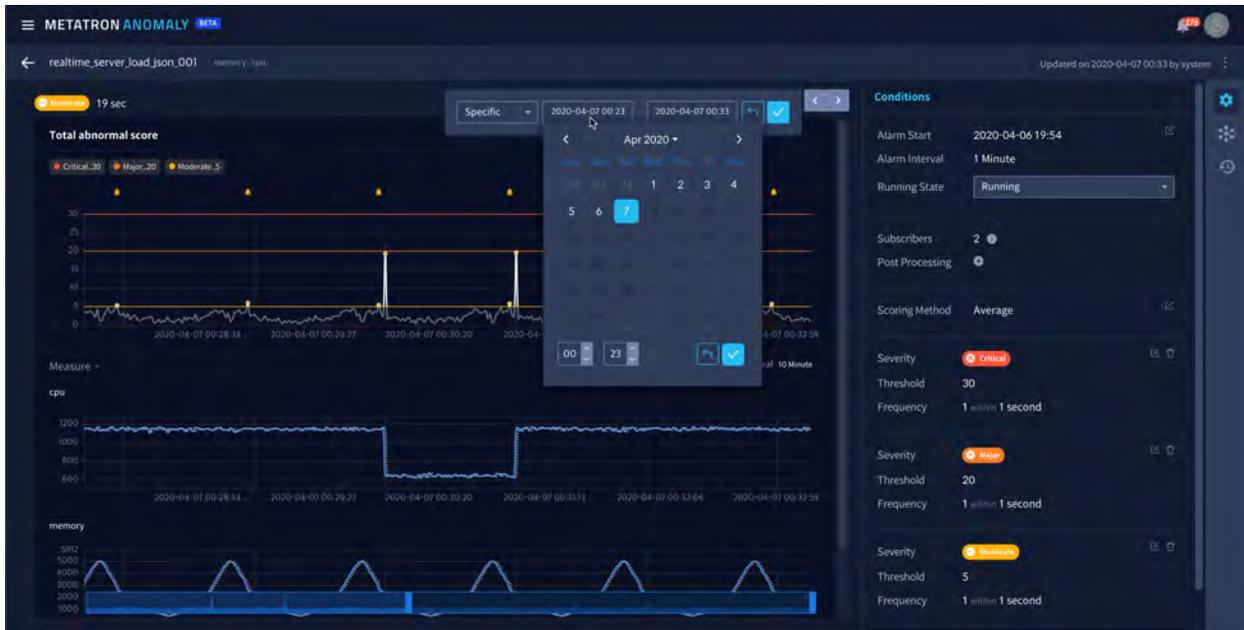
- Owner: 해당 룰을 생성한 사용자

20.2.2 알람 룰 상세

알람 룰 리스트에 열거된 항목 중 하나를 선택하면 해당 알람 룰에 대한 상세 정보를 열람하고 설정을 수정할 수 있습니다. 화면 좌측에서는 모니터링 현황을 시각화하여 보여주고, 우측에는 알람 룰 조건 설정값이 표시됩니다.

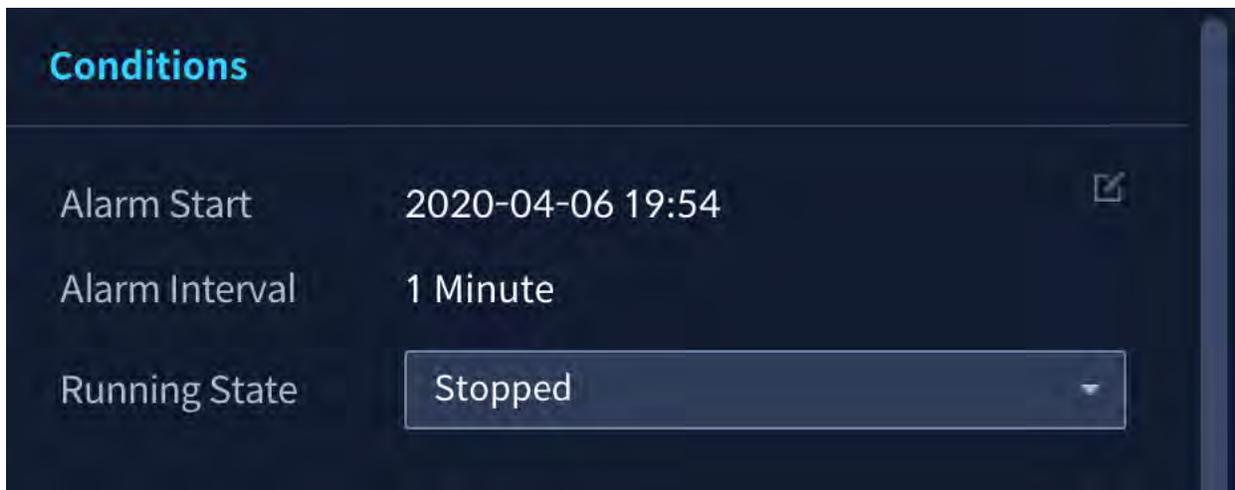


모니터링 현황 영역 상단에는 화면에 보여주는 모니터링 기간 설정값이 표시되어 있습니다.  아이콘을 클릭하면 기간 설정값을 변경할 수 있습니다.

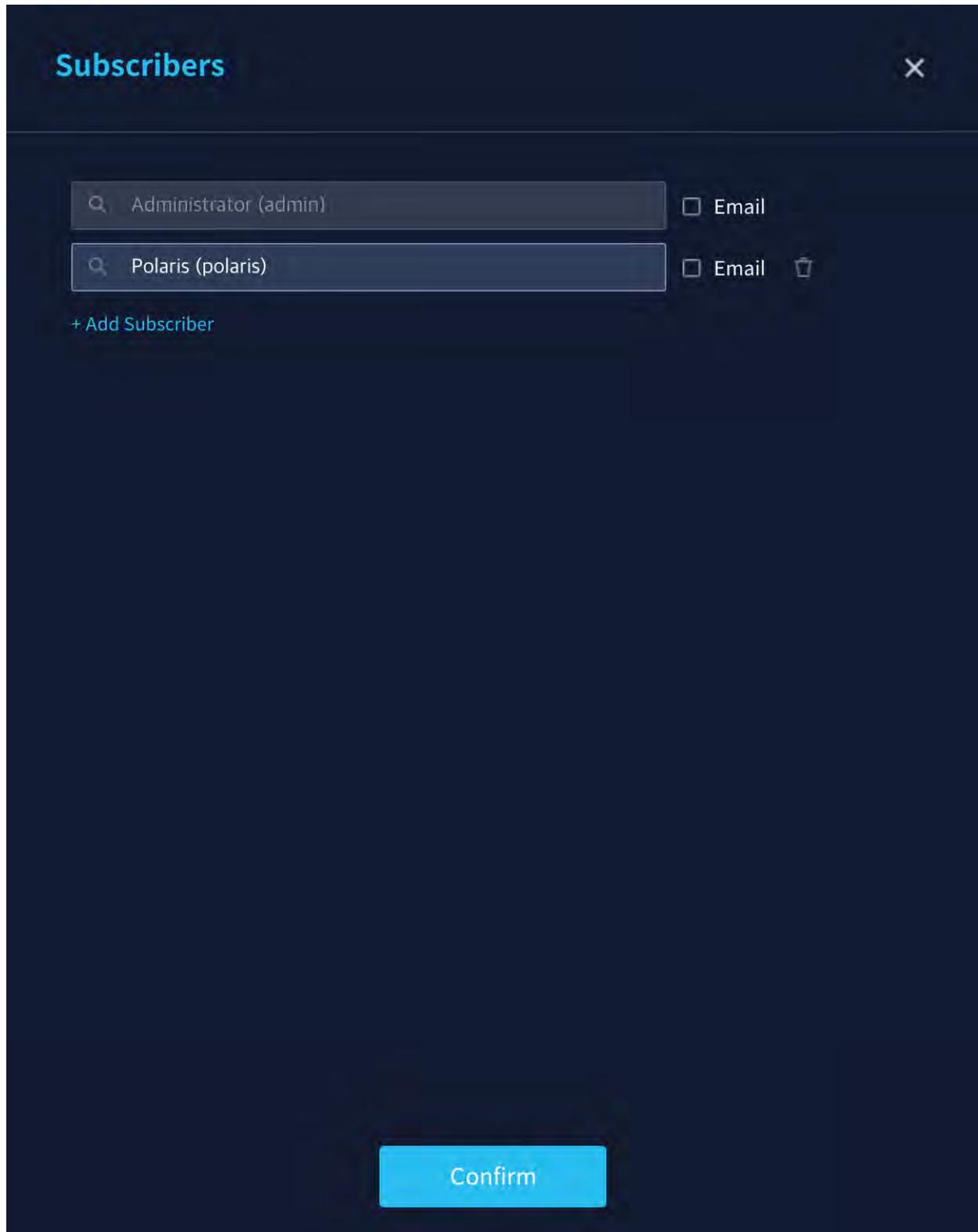


우측의 조건 영역은 해당 룰의 전반적인 설정을 조정할 수 있습니다.

- **Alarm Start:** 알람이 발생하는지 검사하기 시작한 시간
- **Alarm Interval:** 시스템이 알람 발생 조건을 검사하는 주기
- **Running State:** 해당 알람 룰 조건을 검사하는 중인지 (Running) 안 하는 중인지 (Stopped) 에 관한 여부



Subscribers 우측 아이콘을 누르면 해당 알람 룰의 구독자를 추가/변경할 수 있습니다.



해당 룰로 인해 알람이 발생 할 때 추가적인 작업을 하도록 설정할 수 있는 Post Processing을 제공합니다. Post Processing은 현재 두 가지 기능을 제공하고 있습니다.

- **Script Execution:** 쉘 스크립트를 등록하여 실행
- **Additional Chart:** 알람 상세에 표 차트를 노출

Post Processing

Additional Chart

- Script Execution
- Additional Chart

Description

Severity

Data Column

Dimension

Measure

또한 기존에 설정된 알람 발생 기준값을 수정할 수 있습니다. 자세한 내용은 알람 룰 조건 설정하기 항목을 참조하십시오.



우측 끝단에서  버튼을 누르면 Conditions 패널이 Alarm History 패널로 전환되어 지금까지 발생한 알람 이력을 보여줍니다 (다시  버튼을 누르면 Conditions 패널로 되돌아옵니다).



20.3 모델 매니저

시계열 데이터에 머신러닝 모델을 적용했을 때, 일반적으로 시간이 지나면 데이터 패턴이 변하고 모델의 정확도가 점차 하락하는 문제점이 발생합니다. 이 경우 데이터 과학자들은 데이터 관리자에게 요청하여 신규 데이터를 가져온 다음 직접 모델을 재학습시켜 일정 수준 이상의 정확도를 확보하면 시스템에 재배포하는 과정을 거쳐야 합니다. 이 과정은 때로는 수 개월까지 소요될 수 있습니다.

Metatron Anomaly는 데이터 과학자, 데이터 관리자가 아닌 일반 사용자들도 쉽게 모델을 재학습할 수 있는 **모델 매니저** 기능을 지원합니다.

모델 매니저는 다음의 기능들로 이루어집니다.

- 모델 정확도 변동 추이
- 모델 재학습 및 학습 이력
- 모델 비교 및 신규 모델 적용

생성된 알람 룰 상세 페이지 우측 메뉴의  아이콘을 누르면 모델 매니저로 진입합니다.



20.3.1 모델 정확도 변동 추이

상단에서는 모델 정확도가 가장 최근 학습했을 때보다 얼마나 증가하거나 떨어졌는지 보여주며, 그래프에 마우스 오버하면 기간에 따라서 정확도 점수가 변화하는 것을 수치로 보여줍니다. 하단에는 현재 적용된 모델의 정보 및 적용 시점이

표기됩니다.



20.3.2 모델 재학습 및 학습 이력

만약 정확도가 원하는 수치보다 떨어졌을 경우 우측 상단의 Train 버튼을 눌러 재학습할 수 있습니다. 재학습 대상이 될 학습 데이터 범위와 알고리즘 종류를 선택하고 Train을 누르면 재학습을 시작합니다.

Model training [X]

Training interval

Granularity Minute

Last 3 Hour

From 2020-04-08 10:35

To 2020-04-08 13:34

Algorithm

- Statistical Mean Model
- Naive Forecast Model
- Regression Forecast Model
- Seasonal Median Model
- Simple Exponential Smoothing Forecast Model
- Triple Exponential Smoothing Forecast Model
- Long Triple Exponential Smoothing Forecast Model

Train

재학습이 시작되면 Training History에서 지금 시간으로 기록된 메뉴에서 학습 현황을 볼 수 있습니다. 또한 리스트에서 과거에 학습한 내역 또한 확인할 수 있습니다.



20.3.3 모델 비교 및 신규 모델 적용

신규 모델 우측의  아이콘을 누르면 기존에 적용된 모델과 신규 학습한 모델을 비교할 수 있습니다. 기 적용된 모델은 파란색, 신규로 선택한 모델은 분홍색으로 표시되어 그래프 상에서 두 모델의 예측값과 Abnormal Score 값을 비교할 수 있습니다.



새롭게 학습시킨 모델을 룰에 적용하려면 우측의 **Apply this training model** 메뉴에서 **Apply this training model** 을 클릭합니다. 적용된 모델에는 **Applied** 태그가 표시됩니다.



알고리즘

Metatron Anomaly는 머신러닝 알고리즘을 활용하여 시계열 데이터의 비정상 수치에 대해 알람을 발생시킵니다. 이러한 이상치 탐지 알고리즘은 학습 시 비정상 sample을 활용하는지 여부에 따라 두 가지로 나누어 집니다.

- **Supervised Anomaly Detection:** 정상/비정상 여부가 존재하는 학습 데이터 셋을 사용해서 이상치를 검출하는 지도학습 알고리즘. 정확도가 높으나 비정상 sample 취득에 시간과 비용 소모.
- **Unsupervised Anomaly Detection:** 대부분의 데이터가 정상 sample이라는 가정으로, 데이터 셋에 비정상 여부가 존재하지 않더라도 이상치를 검출할 수 있는 비지도학습 알고리즘.

Metatron Anomaly는 일반적인 정상/비정상 데이터 label이 없는 모든 시계열 데이터에서도 이상 탐지가 가능하도록 Unsupervised 알고리즘 학습을 기본으로 제공합니다.

Metatron Anomaly에서는 이러한 알고리즘들을 관리하고 신규 알고리즘을 추가할 수 있는 **알고리즘 매니저**기능을 제공합니다. 알고리즘 매니저는 다음 세 페이지로 이루어집니다.

- [알고리즘 리스트](#)
- [신규 알고리즘 생성](#)
- [알고리즘 상세](#)

21.1 알고리즘 리스트

Anomaly Detection 하위 메뉴 중 **Algorithm** 탭으로 들어가면 모델 학습에 사용 가능한 알고리즘을 리스트에서 확인할 수 있습니다.

No	구분	Algorithm Name	Description	Notebook	Language	Status	Date	Owner
1	기본	Statistical Mean Model	Statistical Mean Model	-	-	Available	2020-02-28	system
2	기본	Naive Forecast Model	Naive Forecast Model	-	-	Available	2020-02-28	system
3	기본	Regression Forecast Model	Regression Forecast Model	-	-	Available	2020-02-28	system
4	기본	Seasonal Median Model	Seasonal Median Model	-	-	Available	2020-02-28	system
5	기본	Simple Exponential Smoothing Forecast Model	Simple Exponential Smoothing Forecast Model	-	-	Available	2020-02-28	system
6	기본	Triple Exponential Smoothing Forecast Model	Triple Exponential Smoothing Forecast Model	-	-	Available	2020-02-28	system
7	기본	Long Triple Exponential Smoothing Forecast Model	Long Triple Exponential Smoothing Forecast Model	-	-	Available	2020-02-28	system

기본적으로 Metatron Anomaly는 아래의 7가지 통계 알고리즘들을 시스템에 내장하고 있습니다.

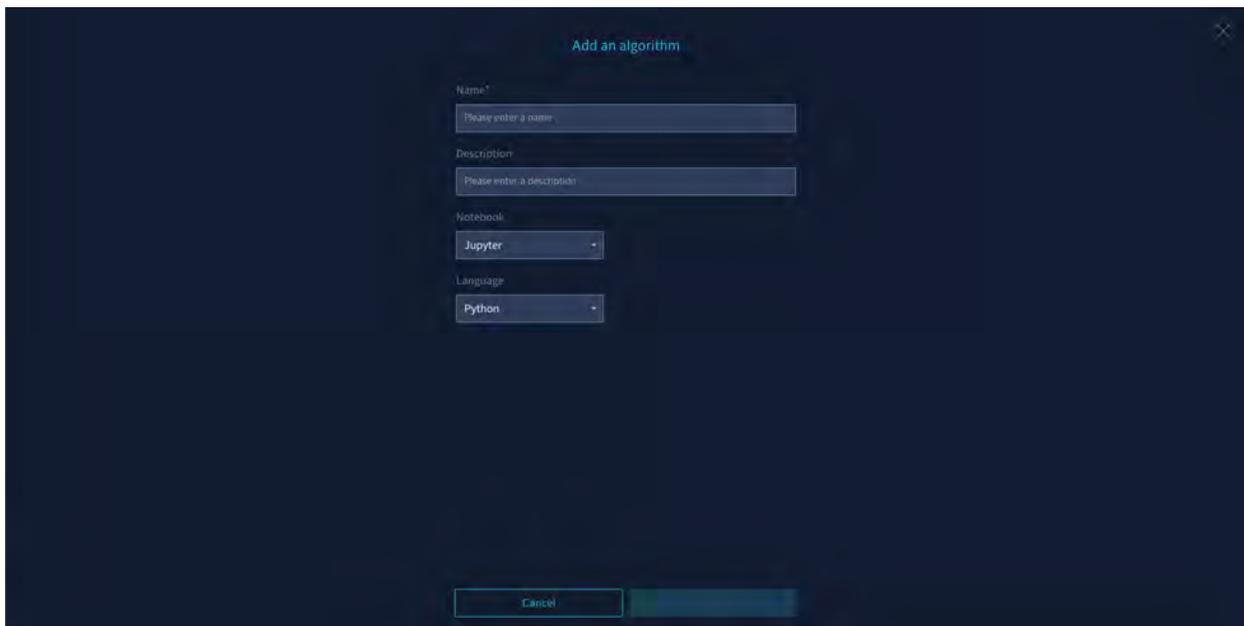
- Seasonal Median Model
- Statistical Mean Model
- Regression Forecast Model
- Naive Forecast Model
- Simple Exponential Smoothing Forecast Model
- Triple Exponential Smoothing Forecast Model
- Long Triple Exponential Smoothing Forecast Model

21.2 신규 알고리즘 생성

알고리즘 페이지 우측 상단의 **+ Algorithm** 버튼을 클릭하면 신규 알고리즘을 추가할 수 있습니다.



신규로 생성할 알고리즘의 이름, 설명을 입력합니다. 알고리즘 생성에 기본적으로 사용 가능한 작업 환경은 Jupyter Notebook이며, Python 언어를 사용할 수 있습니다.



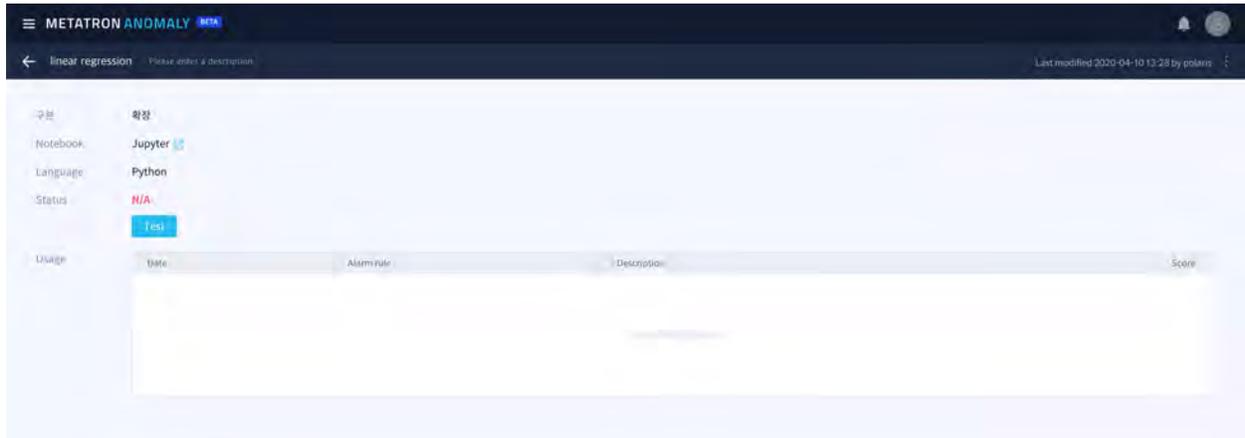
21.3 알고리즘 상세

신규 알고리즘을 생성하면 상세 페이지로 이동합니다. 구분에는 사용자가 직접 생성한 알고리즘이면 확장, 시스템에 기본 구현된 알고리즘이면 기본으로 표시됩니다.

Notebook 옆의  를 누르면 신규 알고리즘을 구현할 수 있는 Jupyter Notebook 환경으로 이동합니다. 기본 템플릿으로 linear regression 알고리즘이 구현되어 있으며, 사용자가 적절히 변형하여 신규 알고리즘을 구현할 수 있습니다.

구현된 알고리즘을 테스트하여 시스템에 적합한지 확인하여야 합니다. 하단의 Test 버튼을 누르면 구현된 알고리즘이 시스템에 적합하지 내부적으로 테스트를 진행합니다. **Status**에 그 결과가 나타납니다. 테스트 결과는 한번도 테스트한

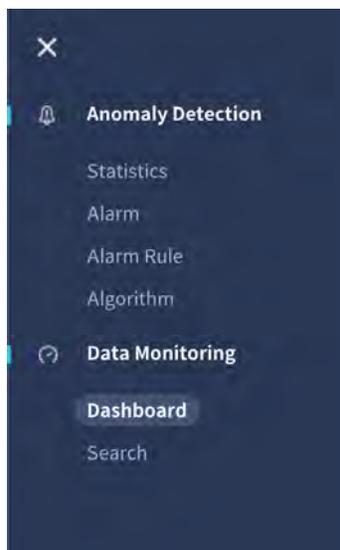
적이 없으면 **N/A**, 실패하면 **Fail**, 성공하면 **Available**로 기록됩니다.



대시보드

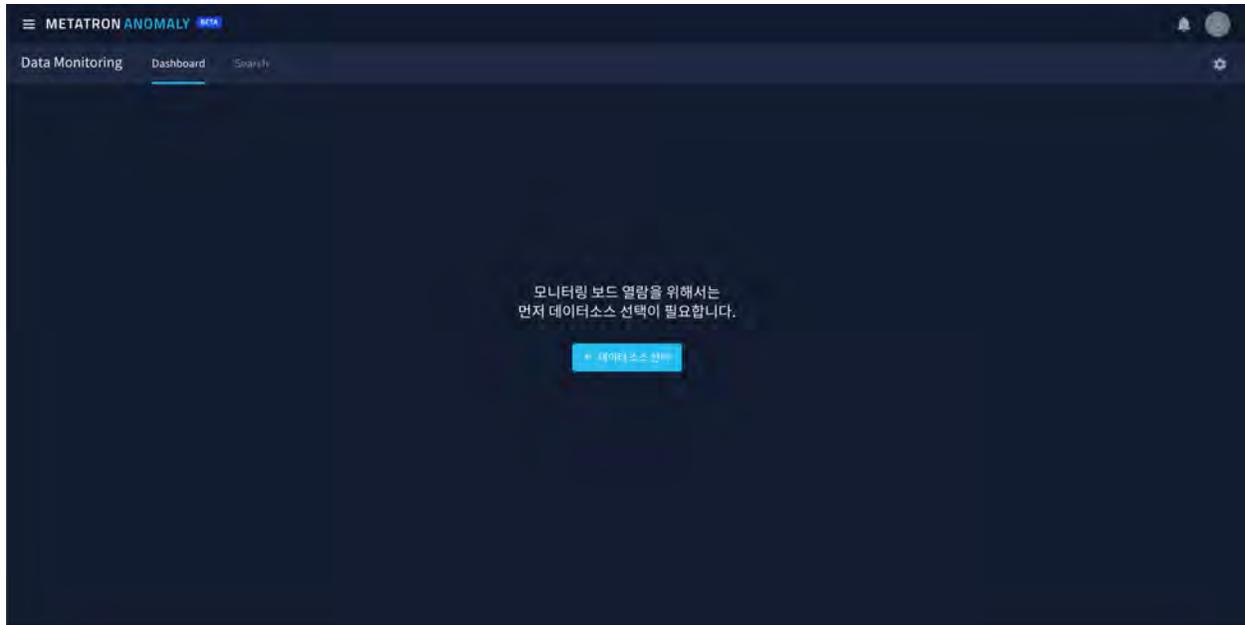
Metatron Anomaly는 머신러닝 모델을 활용한 이상 탐지 기능과 별도로 데이터 소스 자체에 대한 모니터링 기능을 제공합니다. 알람이 발생한 후에 원인을 찾기 위해서 활용하거나, 어떤 측정값 (measure) 에 어떤 차원값 (dimension) 을 대상으로 알람 룰을 생성할지 확인하기 위해서도 활용할 수 있습니다.

그 중 대시보드는 Data Monitoring의 하위 메뉴로, 정해진 차트 몇 가지로 빠르게 데이터 소스에 대한 현황을 파악할 수 있도록 만들어진 기능입니다.

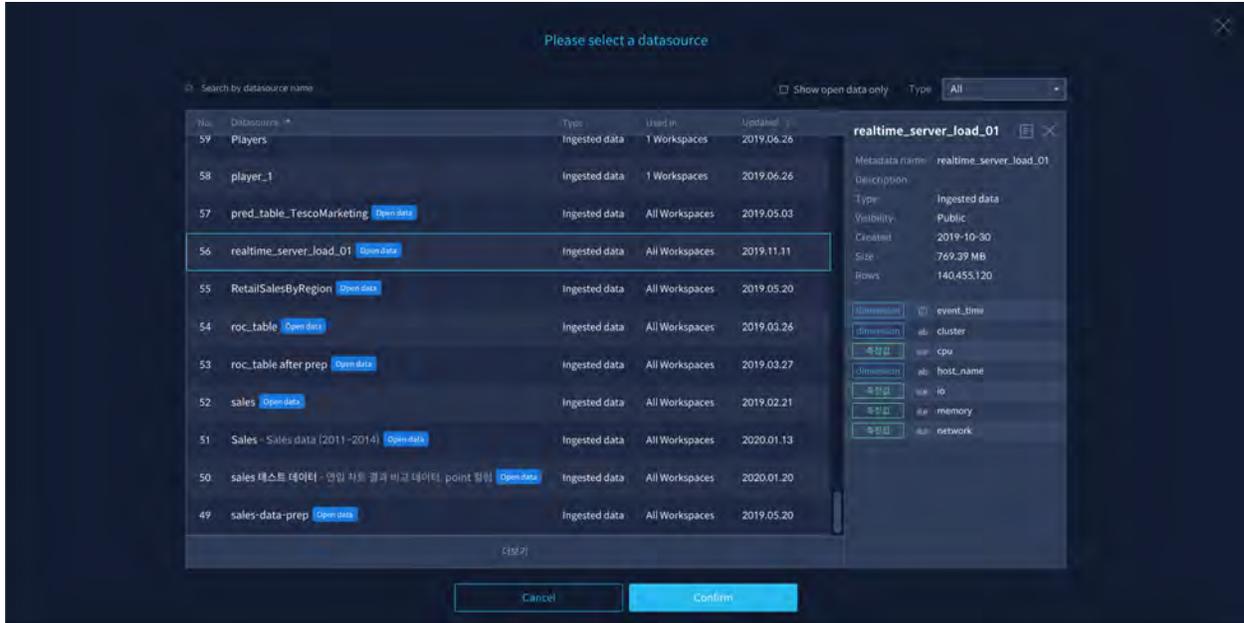


22.1 데이터 소스 선정

가장 먼저 모니터링할 데이터 소스를 선택해야 합니다. 대시보드에 처음으로 진입하면 데이터소스를 선택하는 버튼이 다음과 같이 나타납니다.



버튼을 클릭 후 모니터링할 데이터 소스를 선택합니다.



22.2 실시간 대시보드

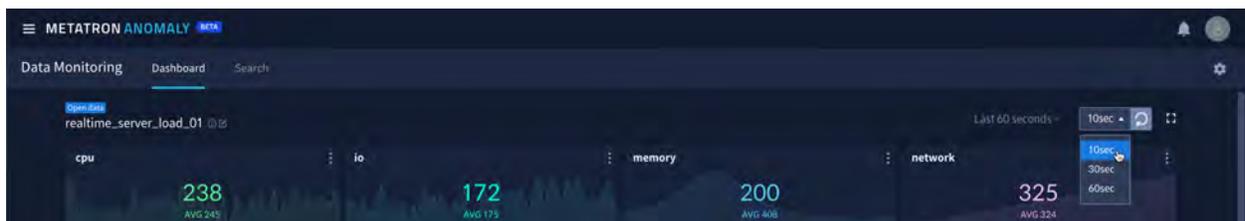
데이터 소스를 선택하면 즉시 주요 측정 값 4개에 대한 차트들로 이루어진 대시보드가 만들어집니다. 이 대시보드는 사용자가 다른 메뉴로 이동 후에 다시 돌아와도 그대로 유지됩니다.



1. 대시보드의 상단에는 데이터소스의 정보를 보여줍니다. 모니터링 하고자 하는 데이터 소스를 변경하려면 데이터 소스 이름 우측의  을 누릅니다.
2. 만약 데이터 소스 선택 후 어떤 차트도 그려지지 않는다면 우측 drop-down 메뉴에서 모니터링 대상 기간을 확인하세요. 이 대시보드는 지속적으로 업데이트 되는 데이터 소스를 모니터링 하는 것을 권장합니다. 정적인 데이터 소스에 대한 대시보드는 Metatron Discovery를 활용해보세요.



3. 상단의  를 누르면 대시보드를 정해진 시간마다 갱신합니다. 기본적으로는 10초마다 업데이트하며 갱신 주기는 3초, 20초 또는 30초로 변경할 수 있습니다.



4.  을 누르면 전체화면 모드로 전환됩니다. 전체화면 모드에서 다시  을 누르면 본 화면으로 돌아옵니다.



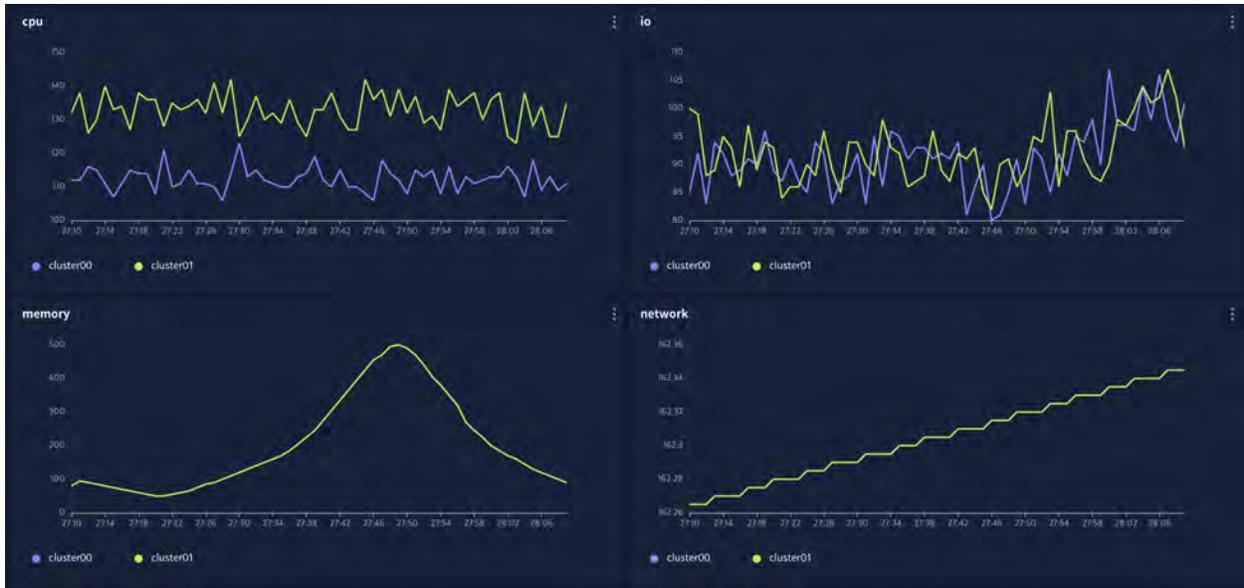
22.3 차트

대시보드에는 선정된 데이터 소스에서 임의의 측정값 (measure) 4개에 대한 11개의 차트가 자동으로 그려집니다.

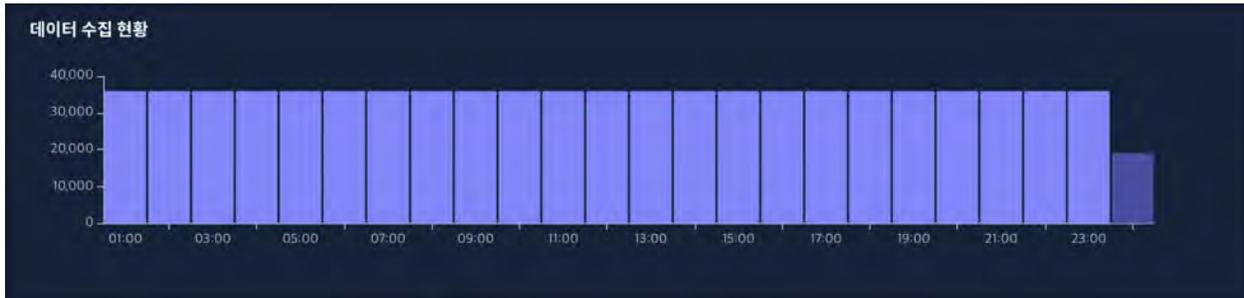
- 측정값 (measure) KPI 차트 : 개별 측정값 4개에 대해 현재값 및 평균 값에 대한 KPI 차트입니다.



- 차원값 (dimension) 별 측정값 라인 차트 : 임의로 선택된 차원값 1개에 대해 개별 측정값 4개에 대한 라인 차트를 그립니다.



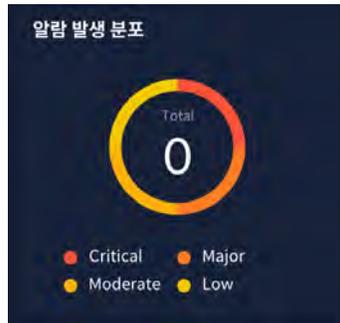
- **데이터 수집 현황 :** 24시간 동안 몇 개의 데이터 레코드가 수집되었는지 기록한 막대 차트입니다.



- **데이터 수집 지연 시간 :** 가장 최근 데이터가 수집된 시간과 현재 시간과의 차이로 수집 지연 시간을 나타낸 차트입니다.



- **알람 발생 분포** : 해당 데이터 소스로 발생한 알람들을 심각도 별로 나누어 개수를 나타내는 파이 차트입니다.

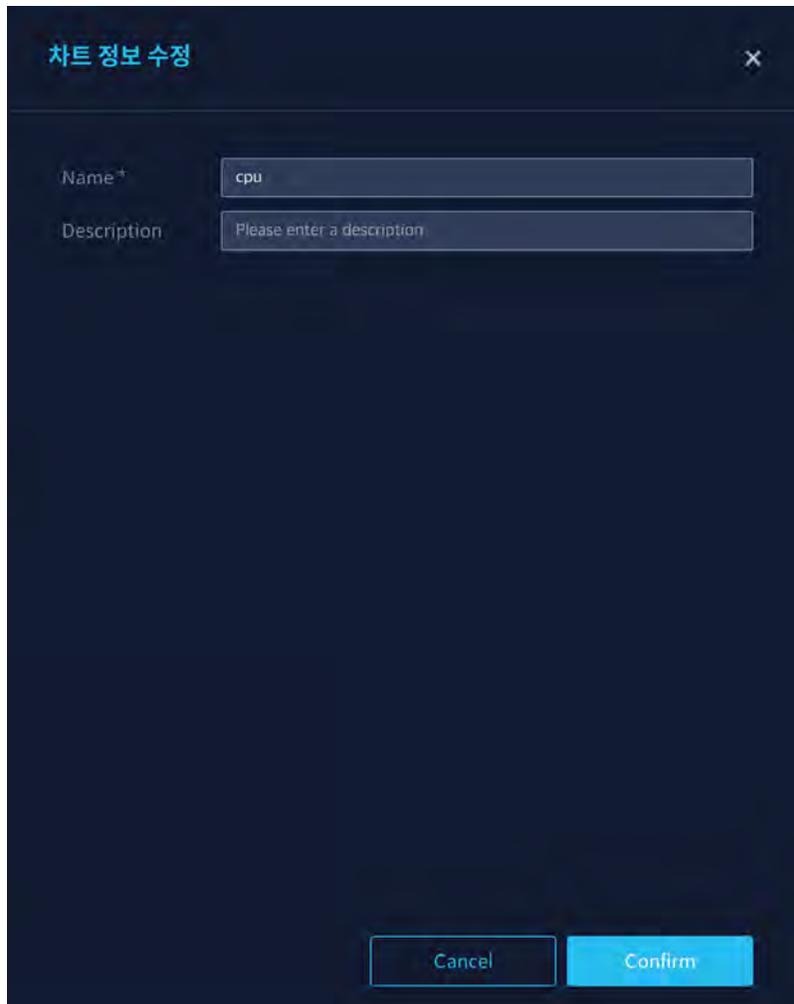


22.3.1 차트 변경

각 차트들은 우측의  버튼을 누르면 정보를 변경할 수 있습니다.

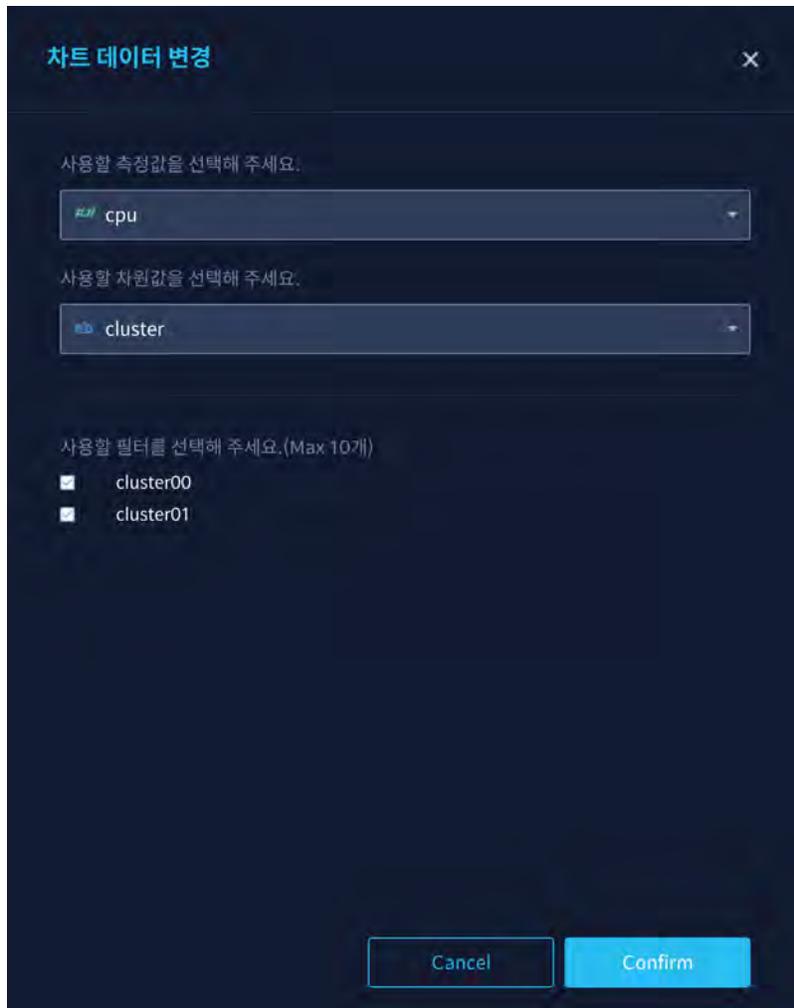


1. **차트 정보 수정** : 차트의 이름을 변경하거나 설명을 추가할 수 있습니다.



The image shows a dark-themed modal dialog box titled "차트 정보 수정" (Edit Chart Information) with a close button (X) in the top right corner. The dialog contains two input fields: "Name" with the value "cpu" and "Description" with the placeholder text "Please enter a description". At the bottom of the dialog, there are two buttons: "Cancel" and "Confirm".

2. **차트 데이터 변경** : 차트에 그려지는 측정값 또는 차원값을 변경할 수 있습니다.

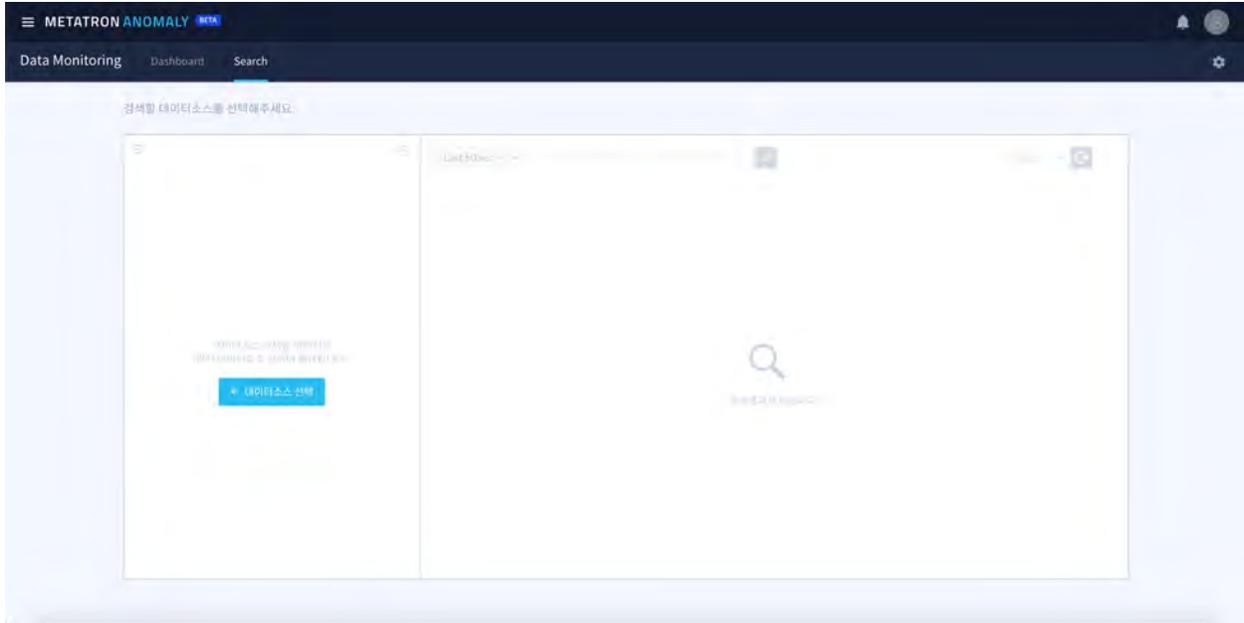


3. **차트 새로고침**: 해당 차트에 대해 최신 데이터로 업데이트 합니다.
4. **차트 초기화**: 최초 설정된 측정값 및 차원값으로 그려진 차트로 초기화합니다.

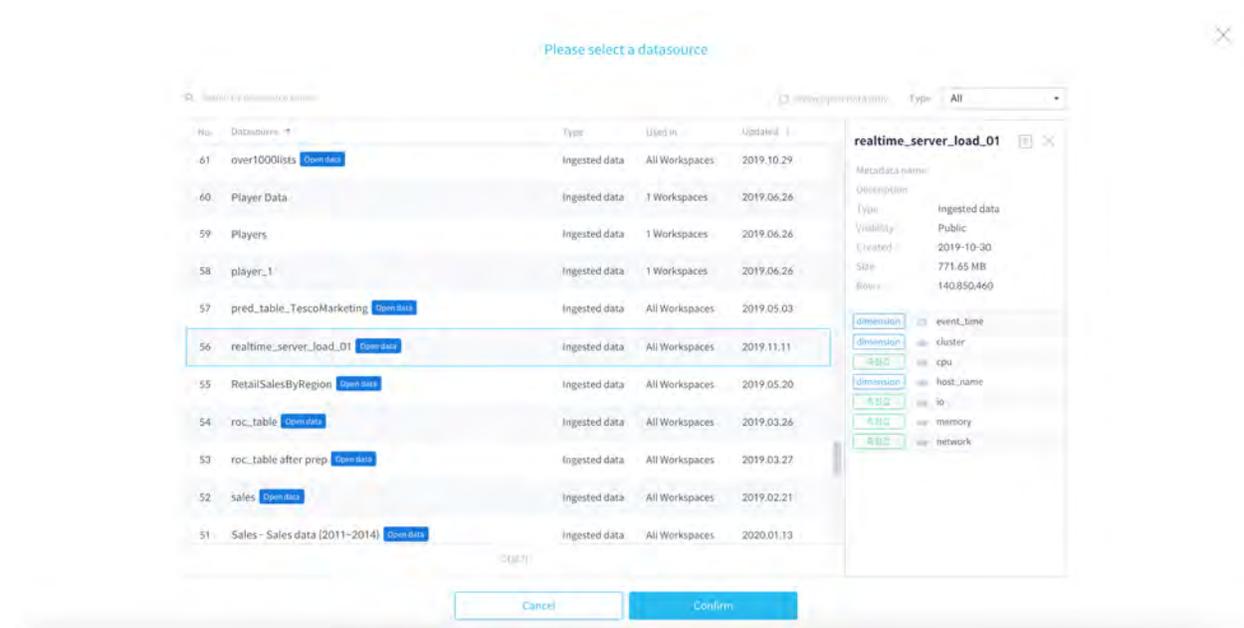
일반적으로는 이상치를 검출하는 시스템과 데이터를 조회하는 시스템이 별도로 존재하여 이상 탐지 직후 원인을 찾기 위해 데이터를 조회하려면 또 다른 시스템에 접근해야만 합니다. Metatron Anomaly는 이상 탐지 알람을 받은 직후 동일한 시스템 내에서 사용자가 선택한 데이터 소스에 대해 다양한 조건으로 조회할 수 있는 기능을 제공합니다.

23.1 데이터 소스 선정

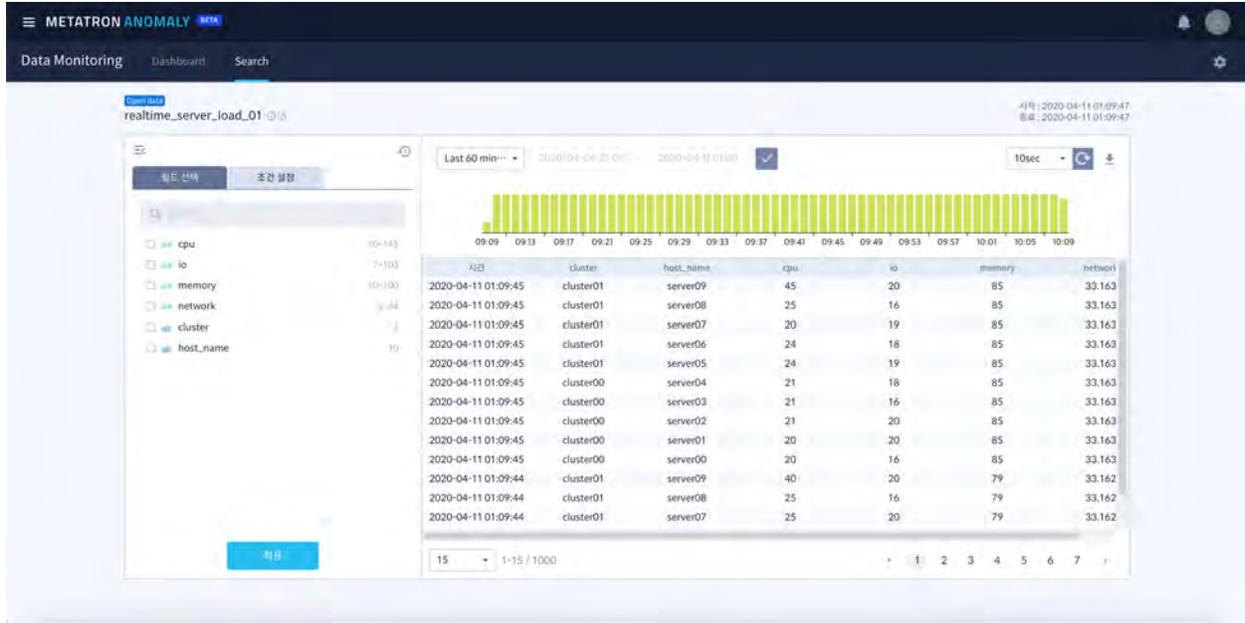
검색 화면에 초기 접근 시 가장 먼저 데이터 소스를 선택해야 합니다. 선택한 데이터 소스는 다른 데이터 소스로 변경하기 전에는 다른 화면으로 이동해도 계속 유지됩니다.



하단의 데이터소스 선택 버튼을 누르면 데이터 소스 선택 창으로 이동합니다.



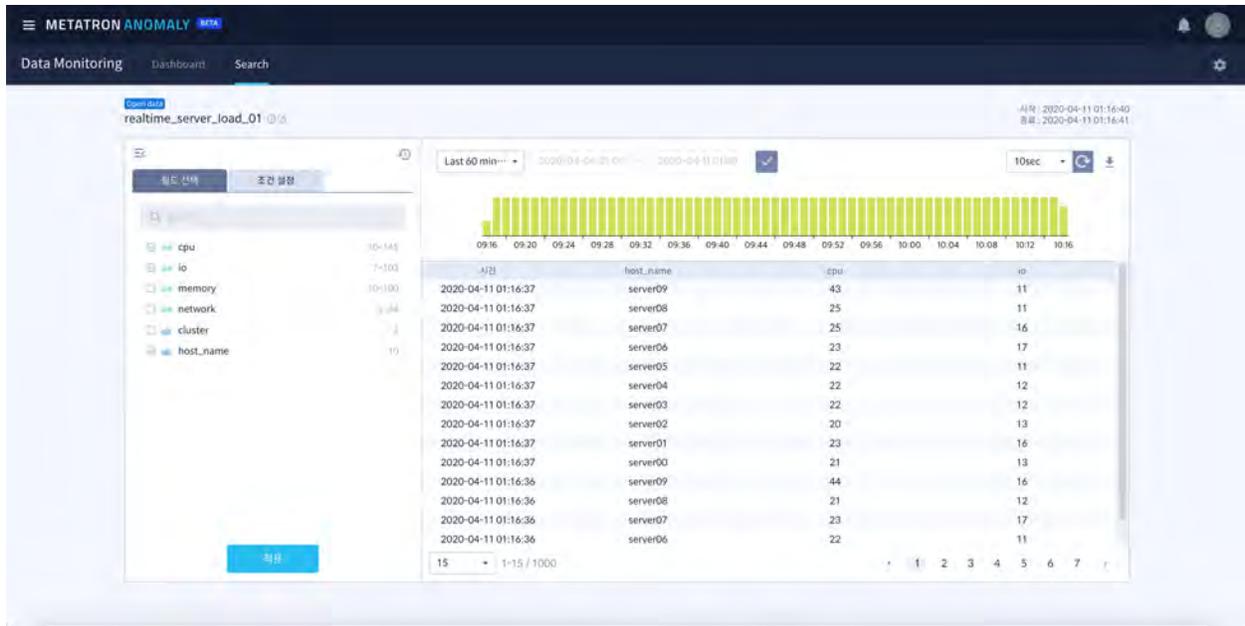
데이터 소스가 선택되면 기본적으로 전체 필드에 대한 결과 값이 조회됩니다. 조회 기간은 데이터 소스 수집 시간 단위에 따라 기본값이 다르게 설정되어 있습니다.



23.2 필드 선택 및 조건 설정

23.2.1 필드 선택

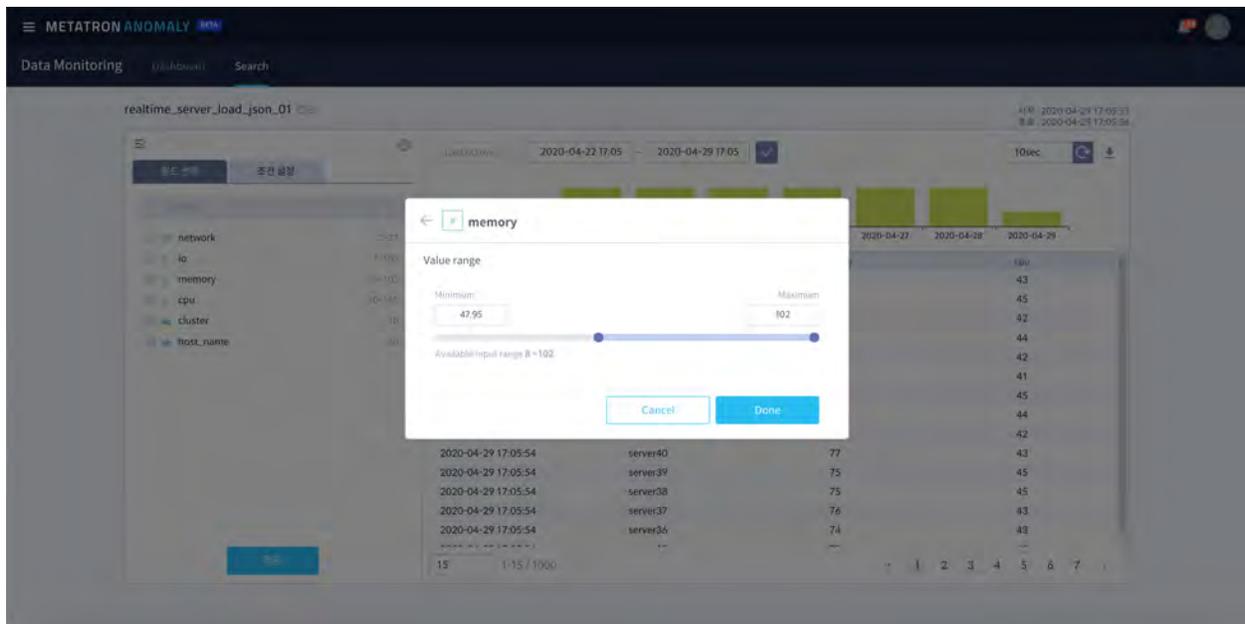
각 필드 값 우측에는 측정값 (measure) 의 경우 값의 범위를, 차원값 (dimension) 의 경우 값의 개수를 보여줍니다. 검색을 원하는 필드를 선택하여 적용을 누르면 해당하는 필드 값에 대해서만 조회할 수 있습니다.



필드명 우측의 숫자는 각 컬럼 값의 범위 또는 고유값의 수를 의미합니다.

- 컬럼이 Measure 라면 우측의 숫자는 해당 컬럼의 최소값부터 최대값의 범위를 의미합니다.
- 컬럼이 Dimension 이면 우측의 숫자는 해당 컬럼이 가지는 고유값의 수를 의미합니다.

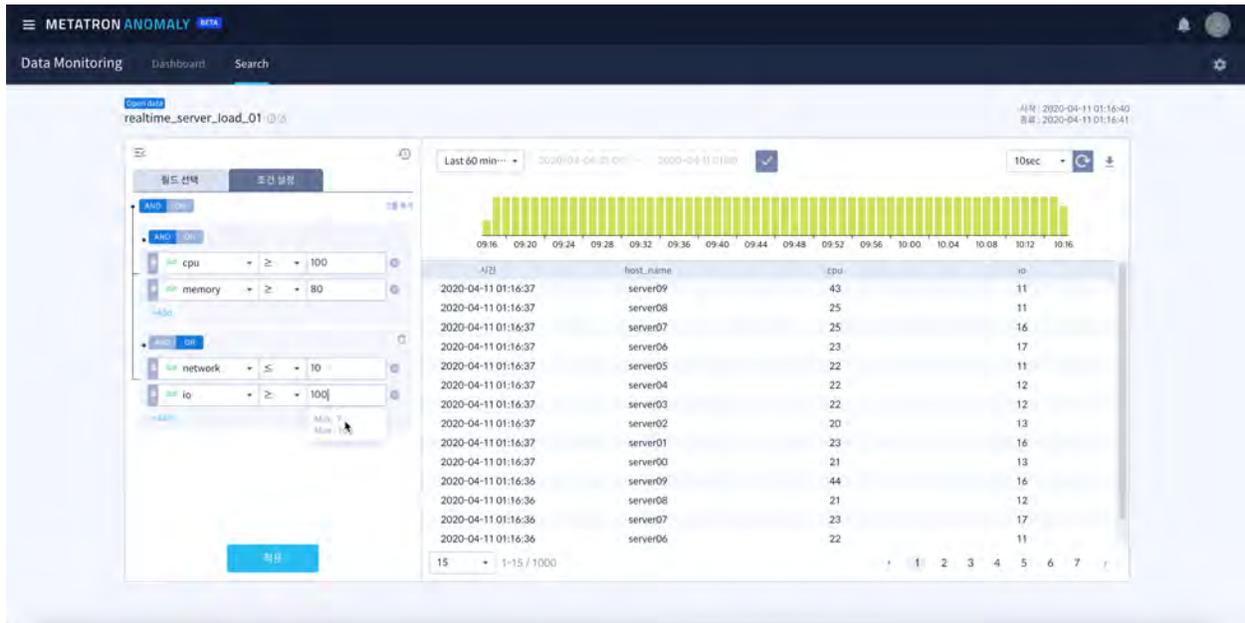
필드명 우측의 숫자를 클릭하면 해당 컬럼에 대해 빠르게 필터링할 수 있으며, 추가된 필터는 조건설정 탭에서 확인할 수 있습니다.



23.2.2 조건 설정

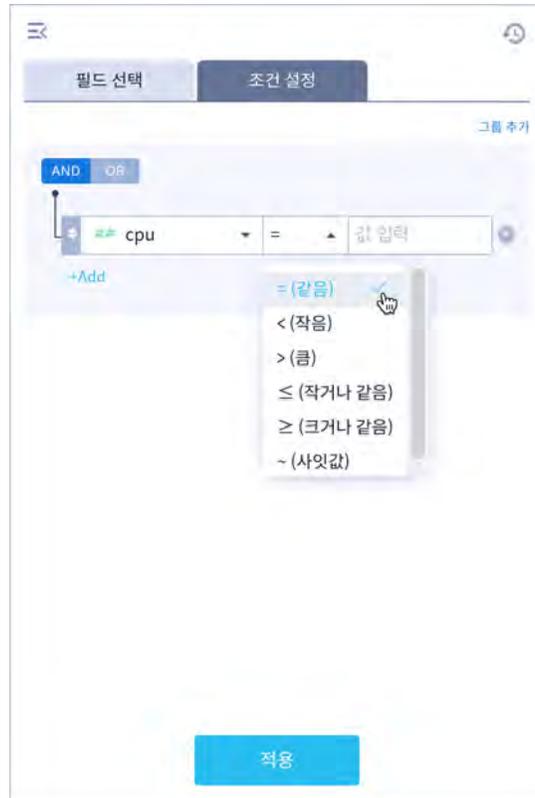
조건 설정 탭으로 이동하면 각 필드에 대한 검색 조건을 상세히 설정할 수 있습니다. 데이터 조회 질의문을 작성하지 못하는 시스템 운영자 및 현업 사용자들도 손쉽게 복잡한 조건을 설정하여 데이터를 조회할 수 있도록 UI를 제공하고 있습니다.

1. 개별 필드값에 대한 조건식들은 서로 and / or 관계를 설정할 수 있으며, 상단의 그룹 추가를 통해 그룹 간에도 and / or 관계를 설정할 수 있습니다.



2. 측정값 필드 조건식에 제공되는 비교 연산자는 다음 6가지입니다.

- =(같음)
- < (작음)
- ≤ (작거나 같음)
- ≥ (크거나 같음)
- ~ (사잇값)

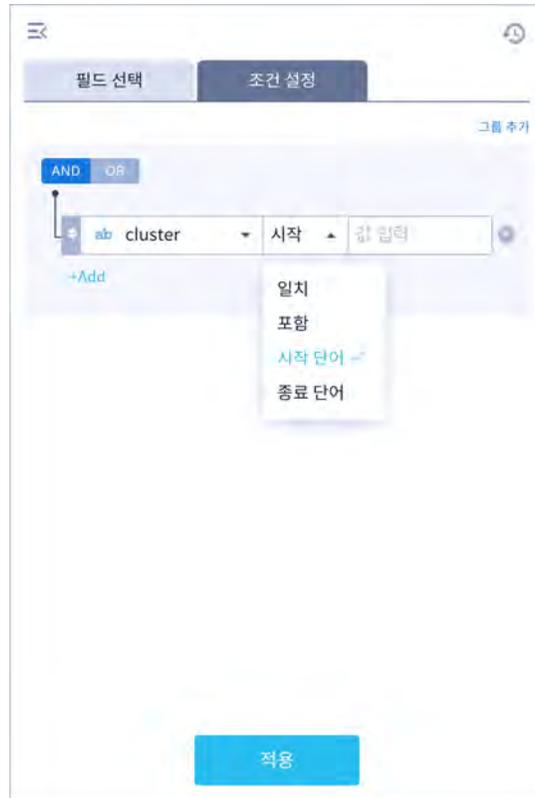


측정값에 대한 조건값 입력창에 커서가 입력되면 최소값과 최대값이 툴팁으로 조회됩니다. 해당 값을 참조하여 조건식을 입력할 수 있습니다.

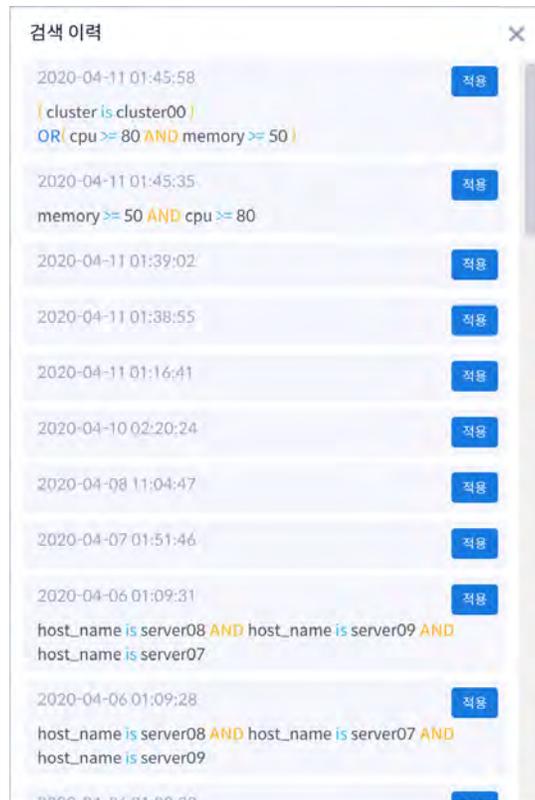


3. 차원값 필드 조건식에 제공되는 연산자는 다음 4가지입니다.

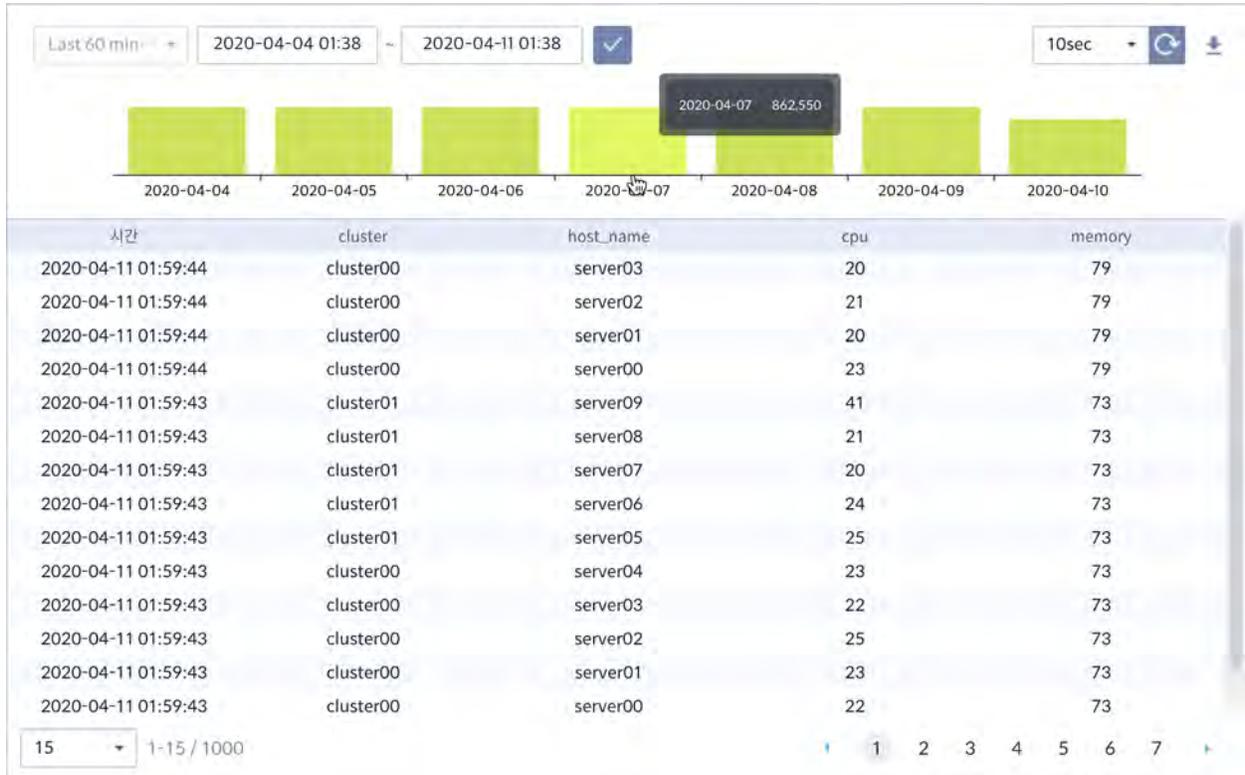
- 시작
- 포함
- 시작 단어
- 종료 단어



4. 모든 검색 이력들은  을 눌러 해당 데이터 소스에 대한 조회할 수 있습니다. 각 검색 이력에 대해 검색 시간, 조건 등을 조회할 수 있으며 우측의 적용을 누르면 동일한 조건으로 데이터를 조회합니다.



23.3 검색 결과



- 조회 기간 설정:** 검색 결과 창 상단에는 조회 대상이 되는 데이터 기간을 설정할 수 있습니다. 현재 시간 대비 상대적인 기간을 drop-down 메뉴에서 선택하거나, 특정 시간 범위를 지정하여 조회할 수 있습니다.
- 검색 결과 실시간 갱신:** 우측의  버튼을 누르면 10초 간격으로 검색 결과를 업데이트하여 신규로 들어오는 데이터에 대해 조회를 지원합니다. 갱신 주기는 30초 또는 60초로 변경 가능하며, 다시 `icon_reset`을 누르면 갱신을 중지할 수 있습니다.
- 엑셀파일 다운로드:**  를 누르면 현재 조회 결과를 엑셀파일 (.xls) 로 다운로드 합니다.
- 히스토그램:** 막대 차트는 데이터가 저장된 시간 단위 별로 데이터 개수를 count한 히스토그램 (histogram) 입니다.
- 리스트 항목 노출 개수 변경:** 한번에 조회되는 데이터 레코드 수는 최대 1000개이며, 한 화면에 보여줄 레코드 수를 하단의 drop-down 메뉴에서 15개, 30개, 또는 50개로 변경할 수 있습니다.