MOZART STUDIO(KOR)

User Manual

VMS SOLUTIONS

본 문서는 고객사 전용으로 작성한 것으로 (주)브이엠에스 솔루션스의 사전 서면 동의 없이 고객사 외부에 회람, 인용, 사본 배포를 금합니다.

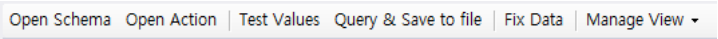
This material was prepared by VMS Solutions, solely for the use of our clients, and it is not be relied on by any third party without VMS Solutions written consent.

MOZART STUDIO (KOR)

Level 1	Level 2	Level 3
MOZART Client 설치		
기능 소개		
실행 구조	MOZART Model 개요	작업 유형 (Job Type)
프로젝트 파일 구조		Extended Arguments
사용자 권한	데이터다운로드/저장방법	
Studio 메뉴 구성	실행구조	
	프로젝트 저장하기	MOZART Model 개요
	Tool Option	
	프로젝트 실행	Grid View Preference
	모델데이터 정보 확인 출력	
Tool Option		
UI Views		
프로젝트 열기 / 저장하기		
View & Check Model	데이터 조회 View	
	Model 개요 및 생성 방법	**Arguments 관리**
		Data Source 관리
		Input Persist Configuration 정의
프로젝트 실행		**Output Persist Configuration 정의**
Open Item(Read Only)	필터,레이아웃 사용자지정	
데이터 조회 View	Grid View 사용법	
	데이터 Export/Import	
Grid View 사용법		
Pivot Grid View 사용법		
Input Dataset 기능		
데이터 Export/Import		
DataItem 정보 Import / Export	DataItem 생성 / 편집 방법	Schema 편집 방법
		DataAction 편집 방법
		DataItem 숨기는 방법
데이터다운로드/저장방법		
모델데이터 정보 확인 출력		
메뉴 등록		
Data Join 사용법		
데이터 Copy Tool 사용법		
Convert Tool 사용법		
Pivot Grid View 차트 기능		
데이터 Look Up 기능		

데이터 조회 View

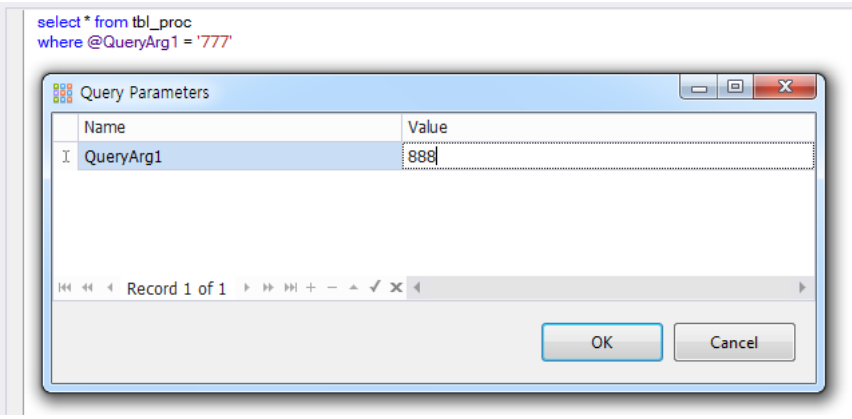
[Experiment Tab View] 에서 Input, Output 을 조회하는 경우에는 Input, Output 데이터가 조회됩니다. 데이터 조회 View 에서 기본적으로 제공하는 기능은 아래와 같습니다.



Open Schema : 해당 데이터의 Schema 정보를 조회하는 기능입니다. Schema View 는 **[Model Tab View]** 에서 Input, Output Schema 를 조회할 때와 동일합니다.

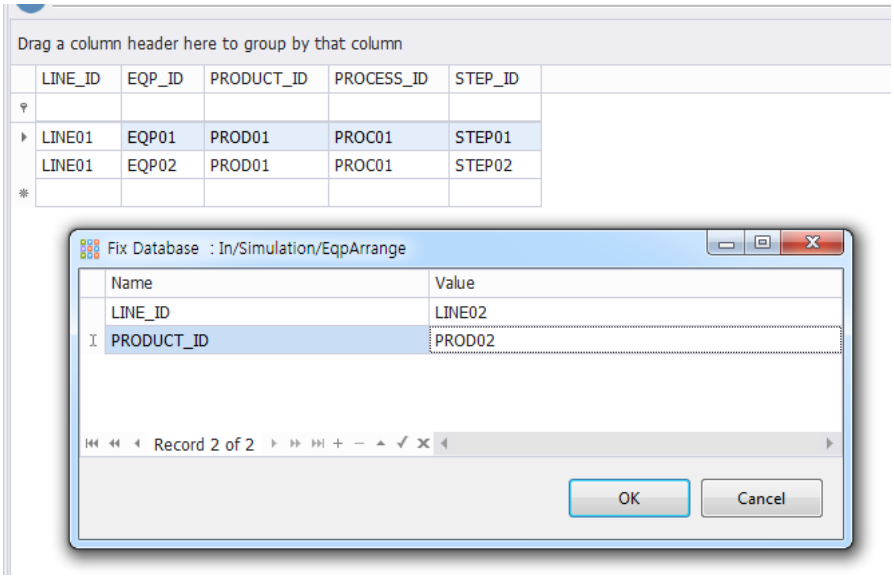
Open Action : 해당 데이터에 정의된 DataAction(쿼리)를 조회합니다. DataAction 이 여러개 등록되어 있는 경우 Active = true 인 DataAction 이 조회 됩니다. **[Model Tab View]**에서는 모든 DataAction 이 조회가능하며 Activate 설정을 할 수 있습니다.

Test Values : Data Item 에 설정된 DataAction 을 실행할 때 사용할 Query Argument 의 Test 값을 설정합니다. Query 문장에 설정된 Argument 가 표에 표시되며, 테스트를 위한 값을 입력합니다. Input data Item 에서만 사용가능합니다. 아래 그림에서와 같이 Query 문장에 "@", "$" 접두어를 사용한 변수가 사용된 경우에만 Query Parameters 창에 변수값을 설정 할 수 있도록 표시됩니다.



Query & Save to file : 활성화된 DataAction 을 실행하여 데이터를 Download 하고 파일을 업데이트합니다. Query 문장에 오류가 있는 경우에는 기존 데이터가 유지됩니다. Input data Item 에서만 사용가능합니다.

Fix Data : 데이터의 특정 컬럼의 값을 일괄적으로 변경하는 기능 입니다. Input/Output data Item 모두에서 사용 가능합니다. 아래와 같이 Fix Database 창에서 컬럼과 값을 입력한 경우 해당 컬럼의 값이 일괄적으로 변경됨을 알 수 있습니다.



Manage View : 그리드에 대한 사용자 설정 값을 저장/복원하는 기능과 그리드에서 설정한 Filter 를 사용자별로 저장하고 선택적으로 적용할 수 있는 기능입니다.

- Layout : 그리드 Layout 관련 설정 메뉴입니다.
  - Save Current Layout : 현재 Layout 을 저장합니다.
  - Load Default Layout : 시스템 기본 Layout을 적용합니다.
  - Load Custom Layout : 저장된 사용자 Layout 을 적용합니다.
- Filter : Filter 관련 메뉴입니다.
  - Save Current Condition : 현재 그리드에서 사용중인 Filter Set 을 저장합니다.
  - Load Filter Condition : 등록되어 있는 Filter 를 조회하고 특정 Filter 를 선택하여 설정할 수 있습니다. 해당 Dialog 에서 기존에 등록된 Filter 를 삭제할 수 있습니다.
  - Clear Current Condition : 현재 설정된 Filter 를 모두 삭제됩니다.
  - Edit Current Condition : 현재 설정된 Filter 를 편집할 수 있는 창이 열립니다.
  - Create New Condition : 현재 설정된 Filter 가 삭제되고, Filter Editor 가 활성화되어 Filter 를 편집할 수 있습니다.

MOZART Client 설치

개요

본 문서는 **MOZART Client 설치 파일을 다운받는 방법**과 **제품 설치/업데이트/삭제하는 방법**을 설명합니다. MOZART 제품을 설치하는 방법은 1) MozartClientExecute.exe 실행 파일을 통한 설치 방법과 2) Mozart Client Installer를 이용한 설치 방법이 있습니다. 두 방법

중 제품 설치 및 버전 업데이트의 편리성을 위해 **1) MozartClientExecute.exe 실행 파일을 통해 제품 설치**하는 것을 권장합니다. 그 외 수동으로 제품을 설치하고자 하는 경우 **2) Mozart Client Installer를 이용한 제품 설치 방법**을 확인하시기 바랍니다.

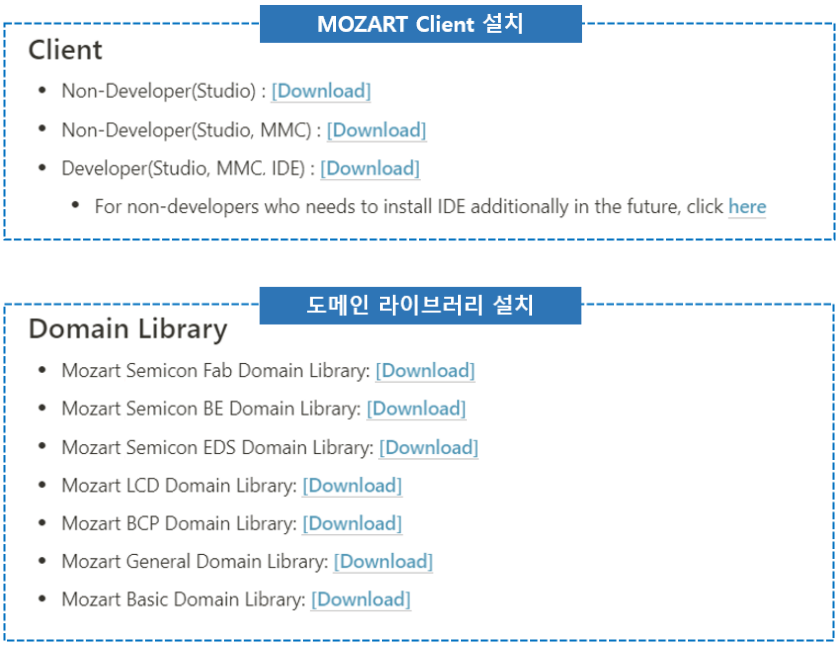
기능 사용에 필요한 스펙

- 적용버전: 2019.116.2.0 이상
- 적용대상: MOZART Client
- 적용 Domain: N/A

> 💡 **Note**
> 2019.115.1 버전 이하 사용자의 경우, 설치된 MOZART 제품을 모두 삭제 후 **Mozart Registry Cleaner**를 이용하여 완전 삭제 후 설치 및 업그레이드를 진행합니다.
> 자세한 내용은 **Mozart Registry Cleaner Gudie** 문서를 참고하시기 바랍니다.

Client 설치 파일 다운로드

MOZART Client는 💿 MOZART Download Archive 에서 일반 사용자(Non-Developer)와 개발자(Developer)에 따라 **Client 제품을 다운로드** 받을 수 있습니다. 또한 필요에 따라 **도메인 라이브러리를 추가 다운로드** 받습니다.



Mozart Download 페이지의 Client 파일과 Domain Library 파일

> 💡 2022.122.0 버전 부터 Product Download 페이지가 변경되었습니다.

MozartClientExecute.exe 사용 가이드

MozartClientExecute.exe 실행 파일을 통해 **Client** 제품을 자동 설치하는 방법을 안내합니다. 우선 아래 3가지 Client 파일의 구성을 확인하여 필요한 제품을 다운 받으시기 바랍니다.

Client 파일 구성

1. **Non-Developer(Studio)**

   MOZART Studio를 사용하는 **일반 사용자 대상** 설치 파일입니다.

   - 다운로드 파일 : **Studio.(version).zip**
   - 파일 구성 : Installer, ToBin, ToRoaming, Client.mdz, MozartClientExecute

2. **Non-Developer(Studio, MMC)**

   MOZART Studio, MMC(Mozart Management Console)를 사용하는 **일반 사용자 대상** 설치 파일입니다.

   - 다운로드 파일 : **Client.(version).zip**
   - 파일 구성 : Installer, ToBin, ToRoaming, Client.mdz, MMC.mdz, MozartClientExecute

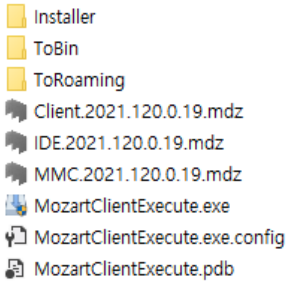
3. **Developer(Studio, MMC, IDE)**

   MOZART Studio, MMC(Mozart Management Console), IDE를 사용하는 **개발자 대상** 설치 파일입니다.

   - 다운로드 파일 : **Developer.(version).zip**
   - 파일 구성 : Installer, ToBin, ToRoaming, Client.mdz, MMC.mdz, MozartClientExecute

▼ 설치 파일에 대한 자세한 설명

아래는 MOZART Client 파일에 구성되어 있는 설치 파일의 설명입니다.



ex) Developer(Studio, MMC, IDE) 파일 구성

📌 - **Installer** : MOZART Client Installer를 설치하는 msi 파일이 있는 폴더입니다.

- **ToBin** : MOZART 설치 파일을 재패키지 시, Mozart Client가 설치되는 폴더(*C:\Program Files (x86)\VMS\Mozart\Client\Bin*))에 복사할 파일을 추가하는 폴더 입니다.

- **ToRoaming** : MOZART 설치 파일을 재패키지 시, *%appdata%\Mozart*에 복사할Studio/IDE/MMC 관련 config 및 라이선스 파일을 추가하는 폴더입니다.

- **Client.(Version).mdz** : MOZART Studio를 설치하는 파일입니다.

- **IDE.(Version).mdz** : MOZART IDE를 설치하는 파일입니다.

- **MMC.(Version).mdz** : MMC(MOZART Management Console)를 설치하는 파일입니다.

- **MozartClientExecute.exe** : 해당 폴더에 존재하는 mdz 및 dll, config 파일을 자동 설치하는 파일입니다.

제품 파일 구성

Aa 제품	파일 구성
Client.mdz	Mozart Studio, 공용 Mozart DLL, DevExpress DLL
IDE.mdz	Visual Studio 확장 도구 인스톨러(vsix), Basic.vfe

제품	파일 구성
MMC.mdz	MMC, MMC 관련 DLL
Domain Library.mdz	Domain DLL, Domain.vfe

2019.116.2 버전 이상 **MOZART Client** 설치 시 생성되는 파일의 경로는 아래와 같습니다.

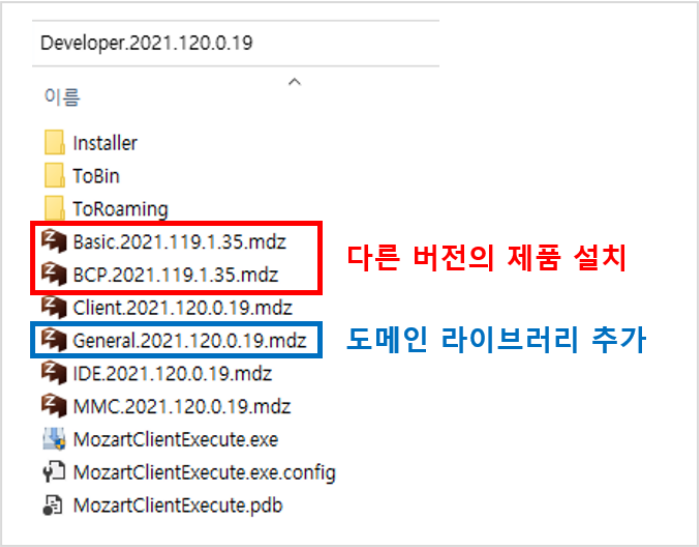
기본 설치 경로

MDZ	설치 경로
Studio	Program Files\VMS\Client\Bin **설정 파일** : %appdata%\Mozart\XXX_Studio\voptions.config
MMC	Program Files\VMS\Client\Bin **설정 파일** : %appdata%\Mozart\Mozart Management Console 2\ServerExplorer.vmgr
IDE	**VS 2017, 2019 :** %Program Files (x86)%\Microsoft Visual Studio\[2017 or 2019]\[Edition: Professional, Enterprise]\Common7\IDE\Extensions\[랜덤 이름] **VS 2015 :** %Program Files (x86)%\Microsoft Visual Studio 14.0\Common7\IDE\Extensions\[랜덤 이름] **VS 2013 :** %Program Files (x86)%\Microsoft Visual Studio 12.0\Common7\IDE\Extensions\[랜덤 이름]
Domain Library	**DLL :** Program Files\VMS\Client\Bin **vfe :** ProgramData\Mozart\FeatureExtensions **vfe 압축 해제 :** %appdata%\Mozart\IDE\[Domain Library.vfe]
LinqPad	%localappdata%\LINQPad\Drivers\DriverContext\4.6\Mozart.LinqPad.Driver

Note
MOZART Client 기본 설치 경로는 Mozart Client Installer의 설치 경로가 기준이 됩니다. 기본 설치 경로를 변경하려는 경우 Installer 설치 시 경로를 변경하시기 바랍니다.

MozartClientExecute.exe 로 설치/업데이트 방법

1. MOZART Download Archive 에서 다운받은 Client 파일을 압축 해제합니다. 해당 폴더에서 **도메인 라이브러리를 추가 설치**하거나 설치된 **제품의 버전 변경(업그레이드, 다운그레이드)**하고자 하는 경우 아래 예시처럼 **mdz 파일을 추가**합니다.
   - 변경하고자 하는 버전의 mdz 파일 추가 (설치된 제품의 업그레이드, 다운그레이드)
   - Domain Library mdz 파일 추가



Developer 폴더에 도메인과 다른 버전의 제품을 추가한 예제

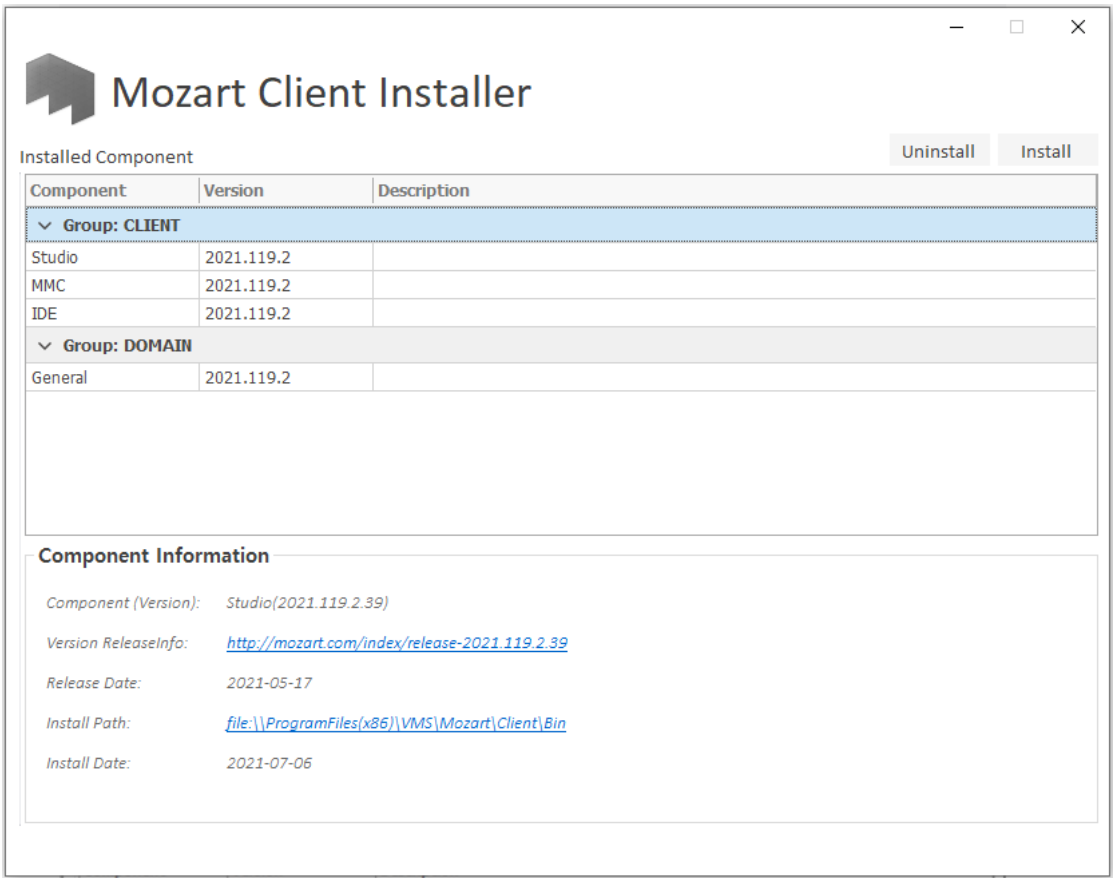
2. 추가적인 DLL, Config 파일도 함께 설치하고자 하는 경우, **ToBin**(*Client\Bin* 경로에 설치), **ToRoaming**(*%appdata%\Mozart* 경로에 설치) 폴더에 파일을 추가합니다. 자세한 방법은 **설치 파일 재배포 시나리오**를 참고하시기 바랍니다.
3. 설치 파일이 모두 준비되었으면 해당 폴더의 **MozartClientExecute.exe** 실행 파일을 관리자 권한으로 실행하여 설치를 진행합니다.
4. **IDE**를 설치하는 경우 **VSIX Installer**가 자동 실행됩니다. Visual Studio Extension을 설치하기 위해 **[Install]** 버튼을 누릅니다.
5. 설치가 완료되면 **Mozart Client Installer**를 실행하여 제품 설치가 완료된 것을 확인합니다.

> ⚠ **Note**
> IDE 제품은 MozartClientExecute.exe 실행 파일을 통한 버전 다운그레이드 설치는 불가합니다. IDE 제품은 Mozart Client Installer에서 삭제 후 낮은 버전을 설치하시기 바랍니다.

Mozart Client Installer 사용 가이드

Mozart Client Installer는 MOZART Client 및 Domain Library를 설치 및 삭제 할 수 있으며 설치된 제품의 빌드 버전 및 설치 경로와 같은 추가 정보를 확인 할 수 있습니다.

Mozart Client Installer 구성



Installed Component

Client 및 Domain Library 파일 별로 설치된 제품 컴포넌트를 버전과 함께 확인 할 수 있는 영역입니다. 우측 상단의 **[Uninstall], [Install]** 버튼을 통해 제품 설치 및 삭제가 가능합니다.

Component Information

Installed Component 에서 선택한 제품의 정보를 확인 할 수 있는 영역입니다.

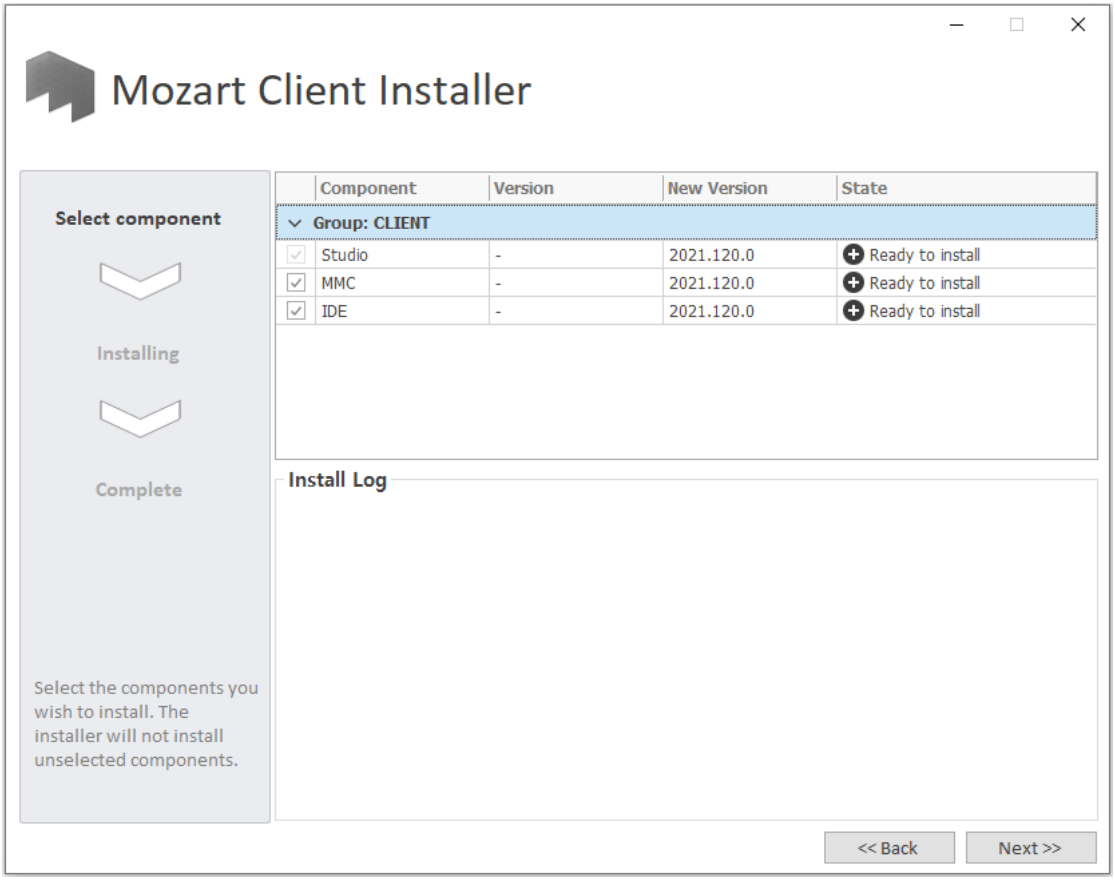
- **Component (Version)** : 선택한 제품 이름(설치 버전)

- **Version ReleaseInfo** : 릴리즈 목록을 확인할 수 있는 링크
- **Release Date** : 제품 배포일
- **Install Path** : 사용자 PC에 설치된 폴더 위치
- **Install Date** : 사용자 PC에 설치된 날짜

Mozart Client Installer 설치 및 MDZ 설치/업데이트 방법

Mozart Client Installer를 통해 제품을 수동으로 설치하는 것이 가능합니다. 그러나 Mozart Client Installer의 경우 설치된 제품보다 낮은 버전의 제품 설치는 불가능하여 버전 다운그레이드를 진행하려면 제품을 삭제한 후 낮은 버전을 설치해야 하는 제약이 있습니다. 제품의 버전 변경은 위의 MozartClientExecute.exe 실행 파일을 통한 설치를 참고하시기 바랍니다.

1. MOZART Download Archive 의 버전 페이지에서 필요한 제품을 확인하여 **Client 파일** 및 **도메인 라이브러리 파일**을 다운로드 합니다. (ex. *Developer.2021.120.0.19.zip, General.2021.120.0.19.mdz*)
2. 다운받은 Client 파일은 압축 해제 후 Installer 폴더 안의 ***MozartClientInstaller.Setup.msi*** 를 실행하여 **Mozart Client Installer** 를 설치한 후 실행합니다. (설치경로 : *C:\Program Files (x86)\VMS\Mozart\Client\Bin*)
3. **Mozart Client Installer**에서 우측상단 **[Install]** 버튼을 클릭합니다.
4. 2번의 파일에서 설치 또는 버전 업데이트 할 제품 mdz 파일을 선택하여 [열기]를 클릭합니다. 이때 제품 중 **Studio(Client.mdz)는 필수 선택 또는 선행 설치** 되어야 다른 제품 설치가 가능합니다.



5. 설치할 제품을 선택한 후 **[next]** 버튼을 눌러 설치를 시작합니다.
6. **IDE**를 설치하는 경우 **VSIX Installer**가 자동 실행이 됩니다. Visual Studio Extension을 설치하기 위해 **[Install]** 버튼을 누릅니다.
7. 설치가 완료되면 **[Done]** 버튼을 눌러 설치를 마칩니다.
8. 필요에 따라 위 방법대로 도메인 라이브러리를 추가 설치합니다. (ex. *General.2021.120.0.19.mdz*)

> ⚠️ **Warning**
> - **Studio(Client.mdz) 제품이 우선 설치**되어야 다른 제품 컴포넌트 설치가 가능합니다.
>
> - Mozart Client Installer에서 **Studio 버전보다 낮은 버전의 다른 제품 설치는 불가능**합니다. (ex. Studio 2021.120.0 설치 후, IDE 2021.119.2 설치)
> 해당 경우는 Studio 삭제를 진행하는 대신 **MozartClientExecute.exe 실행 파일을 통해 버전이 낮은 제품 설치가 가능**합니다. 위의 설명을 참고하여 Studio 버전보다 낮은 제품의 mdz 파일을 MozartClientExecute.exe 실행 파일과 동일한 경로에 복사하여 설치를 진행하시기 바랍니다.
>
> - 이미 설치된 **제품의 버전 업그레이드 설치는 가능**하나 설치된 제품보다 낮은 버전의 제품 설치는 불가능합니다. (ex. IDE 2021.120.0 설치 후, IDE 2021.119.2 설치)
> **제품의 버전을 낮추려면 설치된 제품을 삭제하여 낮은 버전을 설치**하시기 바랍니다.

> ⚠️ 2022.122.0 버전부터 ML 라이브러리는 설치된 도메인 라이브러리와 동일한 버전만 설치 할 수 있도록 변경되었습니다. ML 라이브러리 설치 시 함께 사용하는 도메인 라이브러리를 먼저 설치하여 주시고, 동일한 버전의 ML 라이브러리 파일을 설치하시기 바랍니다.

제품 및 Installer 삭제 방법

MOZART 제품을 삭제하기 위해서는 **Mozart Client Installer**를 통해서 삭제합니다.

1. 실행 중인 MOZART 제품을 모두 종료합니다.
2. **Mozart Client Installer**를 실행합니다.
3. **[Uninstall]** 버튼을 누릅니다.
4. 삭제하려는 제품을 선택하고 **[Next]** 버튼을 누릅니다.
5. **IDE**를 삭제하는 경우 **VSIX Installer**가 실행되어 Visual Studio Extension 삭제가 진행됩니다.
6. 선택한 제품의 삭제가 모두 완료되면 **[Done]** 버튼을 누릅니다.
7. **Mozart Client Installer**를 삭제하려면 위의 방법대로 모든 제품을 삭제한 후 `제어판 > 프로그램 추가 및 제거`에서 **Mozart Client Installer**를 제거합니다.

> 💡 **Note**
> - Studio의 경우 기본 설치 프로그램으로 Studio만 삭제 할 수 없습니다. Studio를 삭제할 경우 설치되어 있는 모든 제품이 함께 삭제됩니다.
> - Mozart Client Installer를 삭제하여도 설치한 제품을 **[Uninstall]** 하지 않을 시 제품은 삭제되지 않고 남아있습니다.

MOZART STUDIO 개요

기능 소개

MOZART STUDIO 는 다음의 세가지 유형의 기능을 지원합니다.

---

1. Model 검토

- Input, Output 데이터 Schema 검토
- Input, Output 의 저장, 조회를 위한 DataAction(Query) 검토
- System Input Argument 조회, Config Argument 조회
- Datasource 확인

---

2. Run experiment

- 시뮬레이션 옵션(Argument) 설정 및 실행 기능
- 개별 실행 별 결과 생성

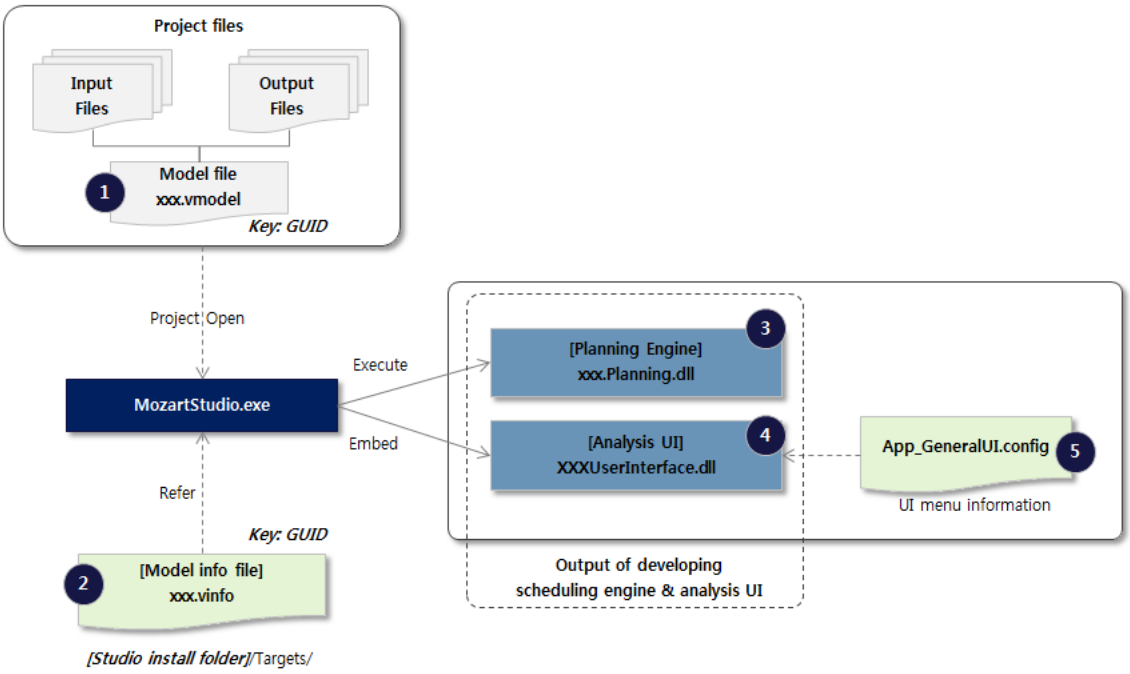
---

3. Verification & Validation

- Input, Output 데이터 및 외부 데이터소스를 활용하여 사용자가 데이터를 분석할 수 있는 UI 를 개발한 후 Studio 에 통합하는 기능
- Input 및 Output 데이터 분석을 통해 로직이 정상적으로 구현되었는지 검토할 수 있는 분석 UI
- Plan 결과가 실제 운영 시스템의 결과를 어느 정도 반영하는지 검토할 수 있는 분석 UI

실행 구조

MOZART STUDIO 는 제품 별 실행 파일로 실행됩니다. MOZART FP 라이선스를 사용하는 경우에는 MOZART IDE 를 통해 FP 모델만을 생성할 수 있으며, 이 경우에 MOZART STUDIO 는 FP_Studio 를 사용하는 방식입니다. Studio 로 특정 모델을 Open 하는 경우 Studio 모델의 실행을 위한 Dll 정보와 모델의 분석을 위한 UI Dll 정보를 참조해야 합니다. 기본적으로 이 정보는 IDE 에서 model 을 만들 때 model 정보에 포함됩니다. 그러나, 개발 환경과 배포 환경이 다른 경우 model 의 정보를 사용하지 않고 **“.vinfo”** 파일을 통해 외부에서 해당 정보를 제공 받아 필요시 사용하게 됩니다. 아래 그림을 참조합니다.



각 파일에 대한 설명은 아래의 표를 참조합니다.

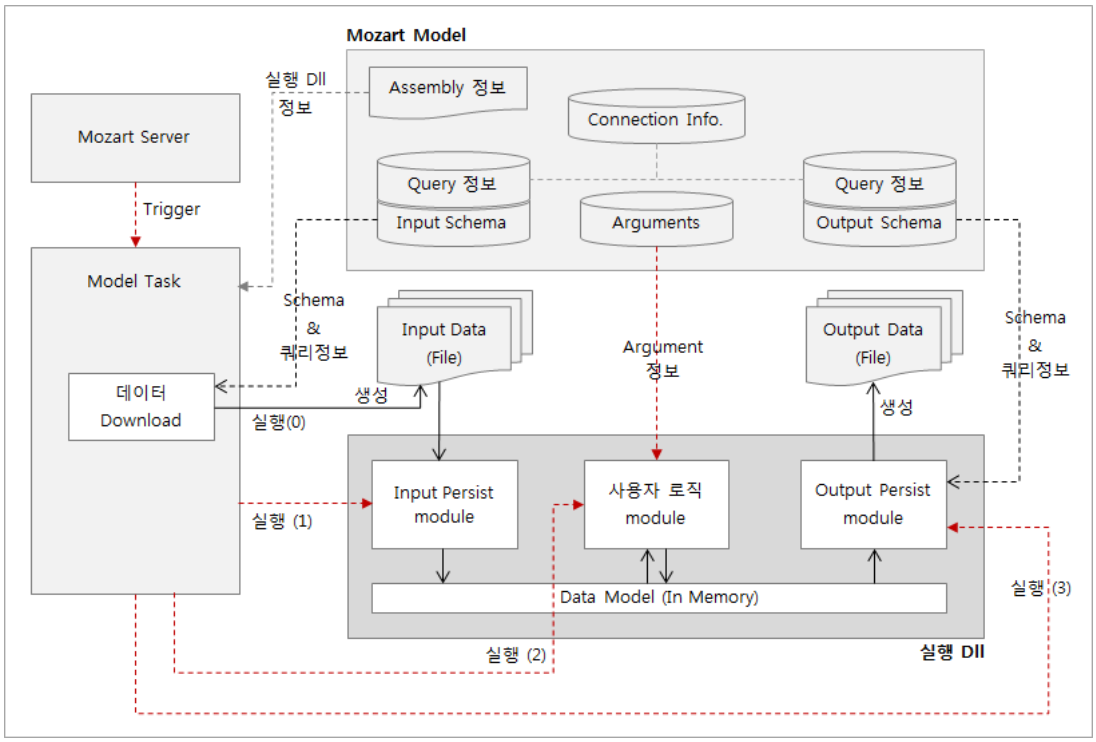
파일	설명
1) Model 파일 (확장자 : vmodel)	모델을 구성하는 Input/Output Data Schema, DataAction, Argument, DataSource 정보를 저장한 파일입니다. 모델의 실행한 결과가 있는 경우 결과 정보를 포함합니다. 한 모델은 여러 개의 Experiment 을 포함하며, 한 Experiment 에 여러 실행 결과를 포함할 수 있습니다.
2) Model 정보 파일 (확장자 : vinfo)	다음의 정보를 포함한 파일입니다. • 모델 실행을 위한 실행 엔진 Dll 정보 • 모델 수행 결과 분석을 위한 분석 UI Dll 정보 • 분석 UI Dll 에 구현된 UI 에 대한 메뉴 정보 본 파일은 MOZART STUDIO 가 설치된 “Bin” 폴더 하위의 “Targets” 폴더에 저장하도록 지정되어 있습니다. 파일이 없는 경우 MODEL 파일(.vmodel)에 포함된 정보를 참조합니다. Model 정보 파일(vinfo) 존재하면 파일에 설정된 내용이 Model 에 내장한 정보에 보다 우선 적용됩니다. 개발 환경에서 설정한 각 Dll 및 Config 파일의 경로와 운영 환경에서의 경로가 다른 경우에는 반드시 vinfo 파일을 작성하여 배포해야 합니다. vinfo 파일의 생성 방식은 Model 개요 를 참조합니다.
3) Planning Engine (XXXPlanning.dll)	모델 실행을 위한 Dll 입니다. MOZART IDE 를 통해 개발할 수 있습니다.
4) Analysis UI (XXXUserInterface.dll)	모델 수행 결과 분석을 위한 분석 UI Dll 입니다. Visual Studio 에서 MOZART STUDIO 에 확장 형태로 삽입될 수 있게 개발 해야 합니다.
5) Analysis UI Config 파일	분석 UI Dll 에 포함된 UI View 를 MOZART STUDIO 의 메뉴에 표시하고, 메뉴 동작시 실행시킬 View Assembly 를 지정하는 파일입니다.

MOZART MODEL 개요

MOZART Model은 MOZART 로 개발된 Logic 에서 사용되는 Input/Output Data Schema, Query, Data 연결 정보 및 Logic 의 처리 시에 활용되는 Arguments 에 대한 정의를 포함하는 Data 집합입니다. Logic 의 개발 시점에는 Schema 와 Arguments 가 사용되며 실행 시점에 Data 연결 정보와 Query 를 사용해 Data 를 받아오거나 저장하는 작업을 수행할 수 있습니다. 특히 실행을 위해서는 해당 Model

을 실행할 Logic 이 포함된 Assembly 정보와의 연결 정보가 필요하여 이 정보 또한 Model 에 포함되어 있습니다. 즉 MOZART Model 을 실행하기 위해서는 Model 파일과 Model 을 실행하기 위한 실행 파일이 필요합니다.

MOZART Framework 상에서 이러한 Model 을 실행하는 방법은 크게 두 가지 입니다. 첫 번째는 MOZART Studio 를 통해 실행하는 방법입니다. 보통 개발자가 개발 중 테스트를 위해 Model 을 실행하거나, 동일한 Model 에 대해 입력 정보를 변경하여 다양한 실험을 수행해보는 경우에 주로 사용하는 방법입니다. 두 번째는 MOZART Server에 의해 실행하는 방법입니다. Model 실행 결과를 사용 시나리오에 따라 운영에 적용하기 위해 사용하는 방법입니다. 아래 그림은 Model 의 구성과 실행 구조를 설명합니다.



Model Task 는 MOZART 에서 기본으로 제공하는 작업 유형 (Job Type)으로 실행 대상 Model 과 Model Task 에 의해 수행되는 (0)~(3)까지의 작업 수행 방식을 설정하기 위한 설정값(Extended Arguments) 을 입력하여 대상 Model 을 실행시켜주는 역할을 합니다. MOZART Studio 는 Model Task 와 유사하게 Model 을 실행시킬 수 있으나, 기본적으로 Inputs 이 생성된 상태의 Model 을 실행시키는 경우가 많으므로 Logic 실행 Input Data 를 Download 받는 작업은 필요 시 별도의 메뉴로 실행하며, Model 실행은 (1)~(3)까지 작업을 묶어 실행하게 됩니다.

작업 유형 (Job Type)

Job Type 은 Job Scheduler 에 의해 실행될 수 있는 작업 유형입니다. Job Type 은 실행을 위해 실행 방법을 결정하는 Arguments 를 가집니다. Job Type 은 Job 관리 화면의 **Edit Job Type** 메뉴에 의해 관리가 가능합니다. 또한 Job 을 정의할 때 선택하도록 표시됩니다.

다음은 MOZART Job Scheduler가 제공하는 기본 작업 유형에 대한 설명입니다.

전자 메일 보내기($sendmail)

해당 Job Type 으로 생성된 작업(Job) 이 실행되면 전자 메일을 보냅니다. Job Type 의 Arguments 로 보낸 사람, 받는 사람, 제목, 본문 및 첨부 파일을 지정해야 하며, 보내는 메일 Server (SMTP) 설정을 해주어야 합니다. 관리 목적으로 MMC 에 의해 실행되는 특정 Job 수행이 실패하는 등의 경우에 메일을 발송하도록 할 수 있습니다.

프로그램 실행($exec)

프로그램이나 스크립트를 시작합니다. **Program/script** 에서 지정된 프로그램을 실행하며 지정된 프로그램/스크립트에 명령줄 인수가 사용되는 경우 **Add arguments(optional)** 텍스트 상자에 이러한 인수를 제공할 수 있습니다. **Start in (optional)** 텍스트 상자에서 프로그램이나 스크립트를 실행하는 명령줄에 작업 디렉터리 (Working Directory) 를 지정할 수 있습니다. 이 디렉터리는 프로그램이나 스크립트 파일의 경로이거나 실행 파일에 사용되는 파일 경로여야 합니다. Windows 에서 기본적으로 제공하는 프로그램 및 사용자가 작성한 일반 실행 파일은 권한이 있는 경우 모두 실행 가능합니다.

Model Task ($model)

Model Task 는 MOZART IDE 를 통해 개발된 Model 기반 작업을 실행하기 위한 작업 유형입니다. Model Task 작업 유형은 Model 의 구동 방식(Mozart Model 개요 참조)을 설정하기 위한 기본 Arguments (Extended Arguments참조) 를 가지고 있습니다. 가장 기본이 되는 Arguments 는 Model Task 가 실행할 Model 정보로 Server 의 Working Folder 에 저장된 Model 을 지정합니다. 실행 대상이 되는 Model 은 수행을 위해 설정되어야 할 내용을 내부 Arguments 로 정의하고 있어 작업을 정의할 때 이 Arguments 들도 설정이 되어야 합니다.

Collaboration Task ($cola)

Collaboration Task 의 경우는 복수 개의 작업을 실행 시점에 작업 간의 통신을 통해 협응하여 실행할 수 있도록 만든 작업입니다. 기본이 되는 Model 을 선택하고 해당 Model 과 Collaboration 대상이 되는 Model 을 설정해 주어야 합니다. (Extended Arguments참조)

Extended Arguments

MOZART Project 로 생성된 Task(Dll 파일) 와 Model 은 실제로 실행될 경우 MOZART 실행 엔진의 Model Task 를 통해 수행됩니다. 이러한 Model Task 의 실행 옵션을 조정하기 위해 사전 정의된 Arguments 가 System Arguments 입니다. 개발자는 Model 의 Input Arguments 설정 시 사전 정의된 System Arguments 를 지정함으로써 사전 정의된 실행 옵션을 활용할 수 있습니다. 아래 표는 사전 정의된 System Arguments 에 대한 설명합니다.

Basic Arguments

Argument Name	Argument 설명	Data Type
#experiment	Model 실행 시 결과를 생성할 Experiment 명. 기본값은 "Experiment 1"	string
version-no	Model Version 명 (기본값은 {model-name}-{yyyyMMdd-HHmmss} 형식)	string
model-name	versionNo 없을 때 versionNo에 사용될 이름	string
start-time	Task 시작시간 (Simulation Clock)	DateTime
end-time	Task 종료시각 (Simulation Clock)	DateTime
period	Plan&Schedule 기간	float
period-unit	period 설정 단위(Days, Hours). 기본값은 일단위입니다(Days).	string
#start-time.AdjustMinutes	Job 실행 시작 시간에서 해당한 시간만큼 조정한 시간을 start-time으로 설정하도록 조정하는 입력변수	int
#model-file	실행 Model 명을 포함한 Full Path	string
#model-dll	Model 을 실행할 Dll 의 Full Path	string
#model-config	Model 실행 시 Config 파일의 Full Path	string

Data Download/Upload 옵션 Arguments

Argument Name	Argument 설명	Data Type
#overwrite_result	Result 덮어쓰기 여부	boolean
#use-database	Database 읽기 여부	boolean
#save-database	Output Data 를 Database 에 쓰기 여부	boolean

Argument Name	Argument 설명	Data Type
#db-to-file	시뮬레이션은 실행하지 않고 Database Sync 만 진행 여부 (기본값 = false)	boolean
#file-to-db	시뮬레이션은 실행하지 않고 Output에 대해 Database Sync만 진행 여부 (기본값 = false), True인 경우 Result가 생성되지 않음	boolean
#db-includes	Input Data Sync 대상 Table List 파일 이름, 대상 Table 들만 Sync 진행	string
#db-excludes	Input Data Sync 제외 Table List 파일 이름, 제외 Table 들을 제외한 모든 Table Sync 진행, #daction_includes 동일 값이 있는 경우, excludes 반영 안됨	string
#daction_excludes	DB 저장 단계에서 DataAction 실행을 제외하기 위한 Output DataItem의 체크박스 드롭다운 목록입니다. 선택한 DataItem은 활성화 조건과 관계없이 DataAction을 수행하지 않습니다. 이 옵션은 `#daction_includes`와 함께 사용할 수 없습니다.	string
#daction_includes	DB 저장 단계에서 DataAction을 실행하기 위한 OutputDataItem의 체크박스 드롭다운 목록입니다. 선택한 DataItem만 DataAction을 수행하고 선택되지 않은 DataItem은 활성화 조건에 관계없이 DataAction을 수행하지 않습니다. 이 옵션은 `#daction_excludes`과 함께 사용할 수 없습니다.	string
#daction_excludes/in	Preloading 또는 Persist-In 단계에서 DataAction 실행을 제외하기 위한 Input DataItem의 체크박스 드롭다운 목록입니다. 선택한 DataItem은 활성화 조건과 관계없이 DataAction을 수행하지 않습니다. 이 옵션은 `#daction_includes/in`와 함께 사용할 수 없습니다.	string
#daction_includes/in	Preloading 또는 Persist-In 단계에서 DataAction을 실행하기 위한 Input DataItem의 체크박스 드롭다운 목록입니다. 선택한 DataItem만 DataAction을 수행하고 선택되지 않은 DataItem은 활성화 조건에 관계없이 DataAction을 수행하지 않습니다. 이 옵션은 `#daction_excludes/in`과 함께 사용할 수 없습니다.	string
#dataSource-set-default	모델에서 기본으로 사용할 Connection String 설정. Key 는 DataSource의 이름이고, Value는 Connection String의 이름이며, 여러 Key/Value 입력 시 Semicolon(;)로 구분	Dictionary<string, string>
#datasource-set-default-exception	#dataSource-set-default의 Key/Value에 해당되는 Connection String을 못 찾 시 Exception 발생 여부	boolean
#commandTimeout	DataAction의 Command 별 Timeout을 설정. 시간 단위는 초 단위이며 기본값은 180초(3분). 자세한 내용은 여기를 확인 바랍니다.	integer

---

Logging/Performance Arguments

Argument Name	Argument 설명	Data Type
#log-dir	Log 출력 경로, Working Directory\Logs 기준의 상대경로	string
#log-level	로그 기록 수준 설정 Verbose~Fatal 설정 가능	string
#performance-profiling	Model 수행의 성능 집계 여부 (기본값 = false), false인 경우 Triggers,Monitoring에서 Trigger Execution Log 정보가 조회되지 않으며, Performance에서도 정보가 보이지 않음	boolean

---

Run Arguments

Argument Name	Argument 설명	Data Type
#more-config-[runindex]	반복 실행 회차 별 Arguments 값을 설정하는 변수. 지정되지 않은 경우 이전 회차의 Arguments 값을 그대로 사용하며, MMC 에서 자동으로 생성	string
#more-runs	Model 의 반복 실행 횟수	int
#run-index	반복 실행 회차 정보이며, MMC 에서 자동으로 생성	int
#exec-dependent-list	하나의 Trigger에 여러 Dependent Trigger가 연결되어 있는 경우 이를 선택적으로 실행 할 수 있는 Argument. 실행 할 Trigger 명을 입력하며 하나 이상 추가 쉼표(,)로 구분. #exec-dependent-list에 추가되는 Depdenet Trigger의 경우 Execution Type을 ReturnIfTrue로 설정해야 함. 자세한 내용은 여기를 참고	string

---

Temp Folder Run Arguments

Argument Name	Argument 설명	Data Type
#use-run-dir	Trigger 실행 시 임시실행 폴더 생성 여부. 자세한 내용은 여기를 참고 바랍니다.	boolean
#max-run-dir	임시실행 폴더 생성 개수 설정. Max 값을 초과하는 경우 가장 마지막에 생성된 폴더가 삭제	int
#use-parent-path	Dependent Trigger 실행 시 임시실행 폴더를 생성하지 않고 Reference Trigger의 가장 최근 임시실행 폴더를 사용할지 여부. 현재는 #file-to-db에서만 사용을 권함	boolean

Zip Model Arguments

Model 실행 후 압축 파일로 Model 을 압축 할 때 사용하는 규칙에 대한 설정 Arguments 입니다.

Zip Model Arguments

Argument Name	Argument 설명	Data Type
#create-csv-files	모델을 압축할 때 대상 파일을 `vdat` 포맷 외에 `csv` 포맷으로 추가 생성하여 저장하는 옵션입니다. 기본값은 false이며 이 옵션은 `#zip = true` 일 때 동작합니다. 옵션 설정시에는 압축파일 용량이 증가하니 이점 참고하여 설정하시기 바랍니다.	boolean
#zip	시뮬레이션 완료 후 Model 압축 여부	boolean
#zip.Use7z	모델 실행 결과를 7-zip으로 압축할지 여부입니다. 기본값은 `false` 이며 해당 옵션을 사용하기 위해서는 `#zip=true` 이어야 합니다. 압축파일 크기를 최대한 줄이기 위한 용도로, 실행 결과 데이터 양이 많은 경우 압축하는데 시간이 오래 걸려 Task 수행 시간이 증가됩니다.	boolean
#zip.Path	압축파일 생성 경로, 정하지 않으면 Model 파일이 있는 위치에 생성됨MOZART Server 의 Working Directory 기준으로 상대경로로 설정하며, 설정하지 않는 경우 Working Directory 에 생성됨	string
#zip.FileNamePostfix	압축 파일 이름 접미사	string
#zip.FileNameTemplate	압축 파일 이름 저장을 위한 Template기본 Template 은 “${model_name}_${zip_now}${zip_postfix}” 사용가능한 keyword 는 아래와 같습니다. ${model_name} : Model 명 ${now} : 압축실행 시점 시각(DateTime) ${zip_now} : 압축하는 시각문자열 (format : yyyyMMddHHmmss) ${zip_postfix} : 설정한 압축파일 이름 접미사 ${version-no} : Model 실행 Version 명 ${start_time} : Plan 시작 시간 (format : yyyyMMddHHmmss) ${end_time} : Plan 종료 시간 (format : yyyyMMddHHmmss)	string
#zip.UpdateToRecent	최근 압축파일 업데이트 여부. `true` 인 경우 가장 최근 압축 파일의 이름으로 압축파일을 생성하여 기존 파일을 덮어씀#zip.FileNameTemplate 이 yyyyMM 형식으로 시작할 경우 동일 년월의 최근 압축파일의 이름으로 압축파일 생성	boolean

Hosting Arguments

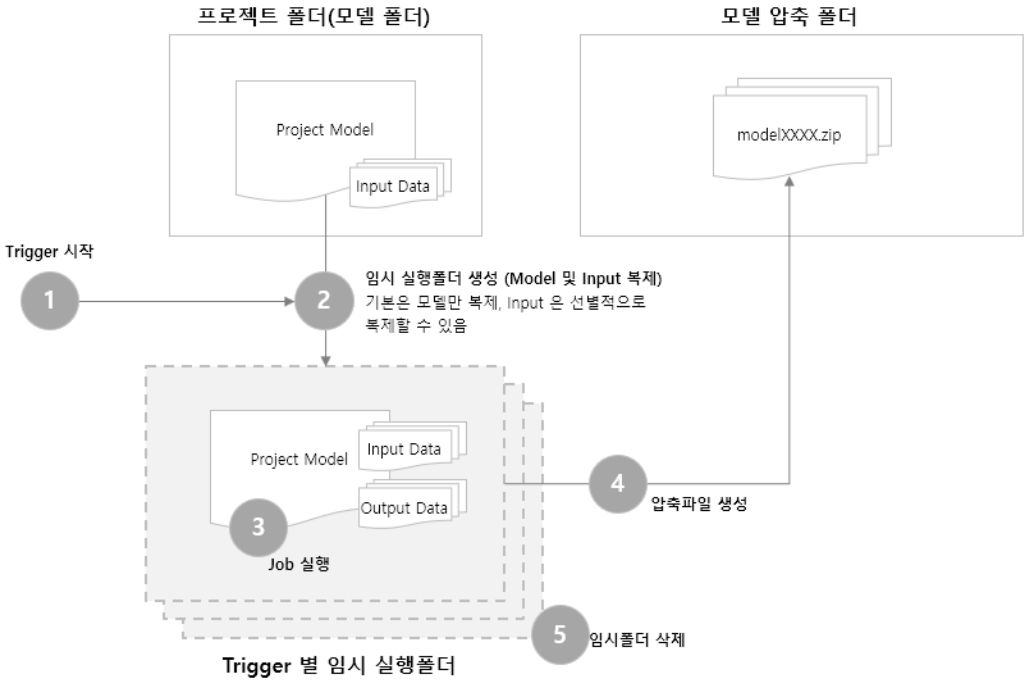
Job/Trigger 실행을 Hosting 할 서버 경로를 설정하는 Argument입니다.

Hosting Arguments

Argument Name	Argument 설명	Data Type
#host-dir	등록된 Job/Trigger의 Task DLL 및 Model이 Server 머신에 설치된 버전이 아닌 다른 버전을 통해 Hosting되어 수행이 필요 할 시 Hosting 할 Server DLL이 위치한 경로를 지정합니다. 해당 Argument 사용 시 Hosting 대상 Server DLL 파일들이 WorkingDirectory 내 존재해야 합니다. 자세한 내용은 여기를 보시기 바랍니다.	string
#host-version	등록된 Job/Trigger의 Task DLL 및 Model이 Server 머신에 설치된 버전이 아닌 다른 버전을 통해 Hosting되어 수행이 필요 할 시 Hosting 할 Server DLL의 버전을 지정합니다. 해당 Argument는 Mozart Server가 설치된 경로의 Execution 폴더 내 버전을 의미하며, 동작 방식은 #host-dir과 동일합니다. 만약 #host-dir과 #host-version이 동시에 설정이 되어 있으면, #host-version에 설정된 값이 우선순위 높습니다. 자세한 내용은 여기를 보시기 바랍니다.	string

Trigger 가 참조하는 모델 및 데이터를 임시폴더에 복사 후 실행 기능

서버에서 Trigger가 수행될 때 Trigger에 설정된 vModel 파일이 위치한 폴더(이하 모델 폴더)가 아닌 임시로 생성한 실행 폴더에서 Trigger를 수행하는 방법이 있습니다. 임시 실행 폴더를 이용 시 Trigger 구동 방식은 아래와 같습니다.



실행 폴더 사용을 위한 Argument

실행 폴더 사용 여부를 판단하기 위한 #use-run-dir 와 실행 폴더 생성 수를 제어 하기 위한 #max-run-dir Argument 가 추가되었습니다.

실행 폴더 사용을 위한 Argument

Argument	DataType	Description
#use-run-dir	boolean	실행 폴더를 사용하여 수행할지 여부(default : false)
#max-run-dir	int	실행 폴더 생성 갯수(default : 5)
#use-parent-path	boolean	Dependent Trigger 사용 시 Reference Trigger의 임시실행 폴더를 사용할지 여부 (default: false)

#use-run-dir = true인 경우, 실행 폴더가 Trigger 수행 시 생성되게 됩니다. 생성되는 실행 폴더의 위치는 다음과 같습니다.

실행 폴더 위치 : *WorkingDirectory\Execution\[Trigger 명]\Temp\[날짜-시간-임의명칭]*

max-run-dir 의 값에 따라 서버에서 유지할 실행 폴더 갯수가 정해집니다. (ex. 값이 2 이면, 가장 최근에 실행 된 2개의 폴더만 유지됩니다.) 만약 사용자가 실행 폴더를 열어 둔 상태에서 Trigger가 수행될 경우에는 사용 중인 폴더로 폴더가 삭제가 되지 않을 수 있습니다. 열어둔 폴더를 닫으면 다음에 실행 될 때 #max-run-dir 값에 따라 삭제됩니다.

사용 방법

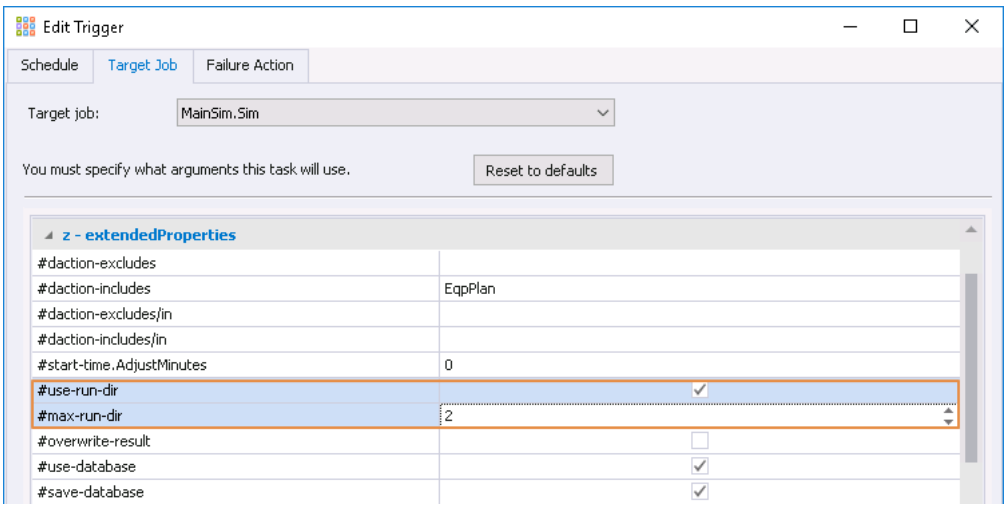
사용 중인 Trigger 를 임시 실행 폴더로 설정하기 전 주의사항

- 실행 폴더를 사용(#use-run-dir = true) 하여 수행 하기 위해서 Input, Output Data 에 대해 정리가 반드시 필요합니다. 배포(Project) 폴더에서 실행 폴더로 파일이 복사되기에 불필요한 파일을 복사하면서 시간 지연을 유발할 수 있기에 미리 삭제가 되어야 합니다. 기존에 사용하던 배포(Project) 폴더에서 Experiment 1 폴더를 반드시 삭제하기 바랍니다.
- 또한 고정적으로 사용되는 Input Data 가 있다면 이를 제외하고 모두 삭제해야 합니다. 실행 폴더로 복사 후 수행하기 때문에 사용하지 않는 Input Data 는 모델 폴더에서 정리되어 복사 시간이 오래 걸리지 않도록 정리하여 주시기 바랍니다. 만약, MMC 의 Project > Files 메뉴를 통해 Input Data 를 추가한 경우에는 반드시 MMC 의 Project > Files 메뉴에서 삭제하여 주시기 바랍니다.
- 116.2 버전부터 임시 실행 폴더에 (base) 폴더가 삭제되었습니다. 116.2 버전 이후 버전으로 업그레이드 시 먼저 (base) 폴더를 삭제 후 업그레이드 하기 바랍니다.

실행 폴더를 이용하여 동일한 Trigger를 병렬적으로 수행하기 위한 Job/Trigger Argument 설정 방법은 다음과 같습니다. Job/Trigger에서 설정방법은 동일합니다.

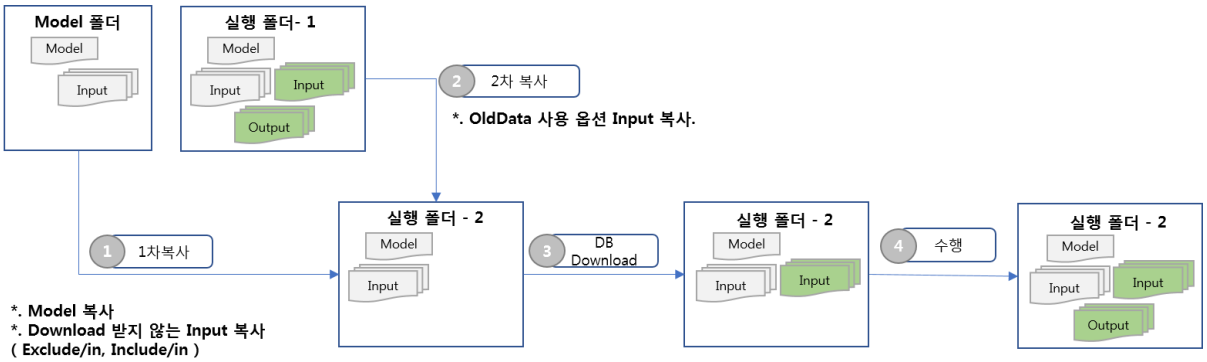
임시 실행 폴더 설정 방법

1. 병렬 수행을 설정할 Job 또는 Trigger을 선택하여 편집 창을 엽니다.
2. Edit Trigger 화면의 Target Job 탭의 #use-run-dir Argument를 체크하여 실행 폴더 사용하도록 합니다.
3. Edit Job 화면의 Target Job 탭의 #max-run-dir Argument에 허용할 실행 폴더의 수를 입력합니다.



실행 폴더 생성 및 수행 절차

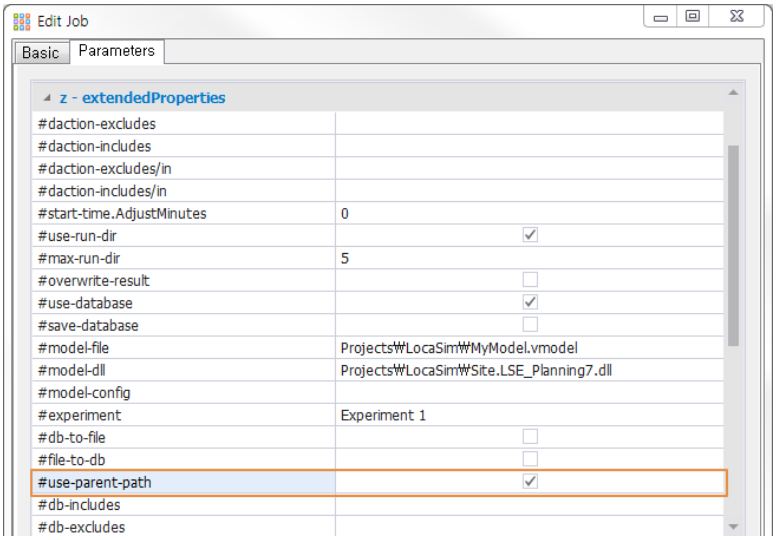
Job/Trigger Argument에서 #use-run-dir 값을 true로 설정한 경우에는 모델 폴더에서 실행 폴더로 model 파일과 Input 폴더가 모두 복사가 됩니다. 복사가 끝난 후 실행 폴더에서 엔진이 수행하게 되며, 실행 폴더 안에서 Job/Trigger Argument 에 설정된 값에 따라 Input/Output DataItem 의 Download 및 Save 에 대한 기능이 동작하게 됩니다.



> **Note**
> Input Data의 용량이 큰 경우에는 복사하는데 시간이 걸릴 수 있습니다. Input Data 중 사용하지 않는 Input Data 는 모델 폴더에서 정리하여 복사 시간이 오래 걸리지 않도록 관리하여 주시기 바랍니다.

Dependent Trigger에서 Reference Trigger의 임시 실행 폴더 사용 방법

Dependent Trigger는 경우에 따라 자신이 참조하는 선행 Trigger의 임시 실행 폴더를 사용할 필요가 있습니다. 예를 들어 선행 Trigger에서는 Backward/Forward Planning만 수행하고 후속 Trigger에서 그 결과를 DB에 저장만 한다면 후속 Trigger는 자신의 모델 폴더나 임시 실행 폴더가 아닌 선행 Trigger의 임시 실행 폴더를 사용하여 수행이 되어야 합니다. #use-parent-path Argument는 Dependent Trigger가 선행 Trigger의 임시 실행 폴더를 사용하여 실행할지 여부를 결정하는 Argument이며 MMC에서 설정 방법은 아래 그림과 같습니다.



앞서 기술한 것과 같이 #use-parent-path는 Dependent Trigger에서만 사용 가능한 Argument이며 주로 선행 Trigger의 결과를 DB에 저장만 하는 #file-to-db = true인 경우에서만 주로 사용됩니다. 다음은 #use-parent-path Argument의 사용 시나리오 예제입니다.

- Trigger A : Main Trigger
- Trigger B : Trigger A 수행 결과 중 중요 결과를 저장하기 위한 Dependent Trigger(Trigger A 종료 후 수행)
- Trigger C : Trigger A 수행 결과 중 모니터링 결과를 저장하기 위한 Dependent Trigger(Trigger B 종료 후 수행)
- Trigger A 에서 실행 폴더를 사용하도록 체크합니다.(#use-run-dir = true)
- Trigger B 는 Trigger A 가 종료 후 Trigger A 의 결과를 DB 에 저장하도록 설정하고(#file-to-db = true) Trigger A 의 위치를 참조할 수 있도록 설정합니다.(#use-parent-path = true, #use-run-dir = true)
- Trigger C 도 Trigger B 가 종료 후 Trigger A 의 결과를 DB 에 저장하도록 설정하고(#file-to-db = true) Trigger A 의 위치를 참조할 수 있도록 설정합니다.(#use-parent-path = true, #use-run-dir = true)
- Trigger A 종료 후 결과가 Trigger B, Trigger C 에서 저장되는지 확인합니다.

별첨

다음은 #use-run-dir, #max-run-dir, #use-parent-path Argument 사용 시 같이 사용 할 수 없거나 사용에 주의가 필요한 Argument들에 대한 목록입니다.

- **#db-to-file**: 임시 실행 폴더가 아닌 Model 폴더에서 수행됩니다.(기존과 동일함)
- **file to db job**: 임시 실행 폴더에서 수행 그러나 #use-parent-path = true인 경우 새로운 실행 폴더를 생성하지 않고 선행 Trigger의 실행 폴더를 사용하여 수행합니다.

- **overwrite job:** 운영시 #use-run-dir 과 함께 실행 할 수 없는 기능입니다. 함께 수행시 가장 마지막에 수행한 폴더에서 수행되며, 실행 폴더가 추가 생성되지 않습니다.

DB 쿼리 최대 수행 시간을 제어하는 방법

확장 아규먼트를 사용해서 DataAction의 타임 아웃 시간을 늘리는 방법에 대해서 설명합니다.

기능 사용에 필요한 스펙

- 적용버전: 2017.1.106.0 이상
- 적용대상: Mozart IDE, Mozart Studio

DB 에서 대량의 데이터를 조회하거나 저장할 때 쿼리 수행 시간이 길어질 수 있습니다. 이때 쿼리 수행 시간이 지정된 기본 설정된 시간(Mozart 기준: 180초)내에 완료되지 않으면 오류가 발생합니다. 이런 경우 확장 아규먼트인 `#commandTimeout` 를 이용하여 쿼리 수행 제한 시간을 사용자가 조절할 수 있습니다.

`#commandTimeout` 에 대한 기본적인 정보는 다음과 같습니다.

- 데이터 타입: integer
- 시간 단위: 초
- 기본값: 180초
- 동작 특징: 아규먼트의 값을 지정하는 경우 DB job retry count 설정은 무시합니다. 즉, 수행 시간이 설정한 시간을 초과하는 경우 오류 발생 후 다시 시도하지 않습니다.

사용 방법 및 예제

확장 아규먼트는 **IDE** 나 Mozart Studio 의 실행 설정 화면에 기본으로 표시되지 않습니다. 따라서 확장 아규먼트를 설정하는 방식은 두 가지로 코드로 `ModelContext.QueryArgs` 에 직접 설정하는 방법과 **Input Argument**로 등록하는 방법이 있습니다.

코드로 설정하는 방법

`#commandTimeout` 은 쿼리 수행 제한 시간을 조정하는 아규먼트이기 때문에 Data Item의 Data Action이 수행되기 이전 단계에서 값을 설정해야 합니다. 즉, **Main** 모듈의 Setup 단계(BeginSetup ~ EndSetup)에서 `#commandTimeout` 의 값을 설정해 주어야 하며, 그 중에서 쿼리 아규먼트를 설정 할 수 있는 `SetupQueryArgs` 함수에 구현하는 것을 권장 드립니다.

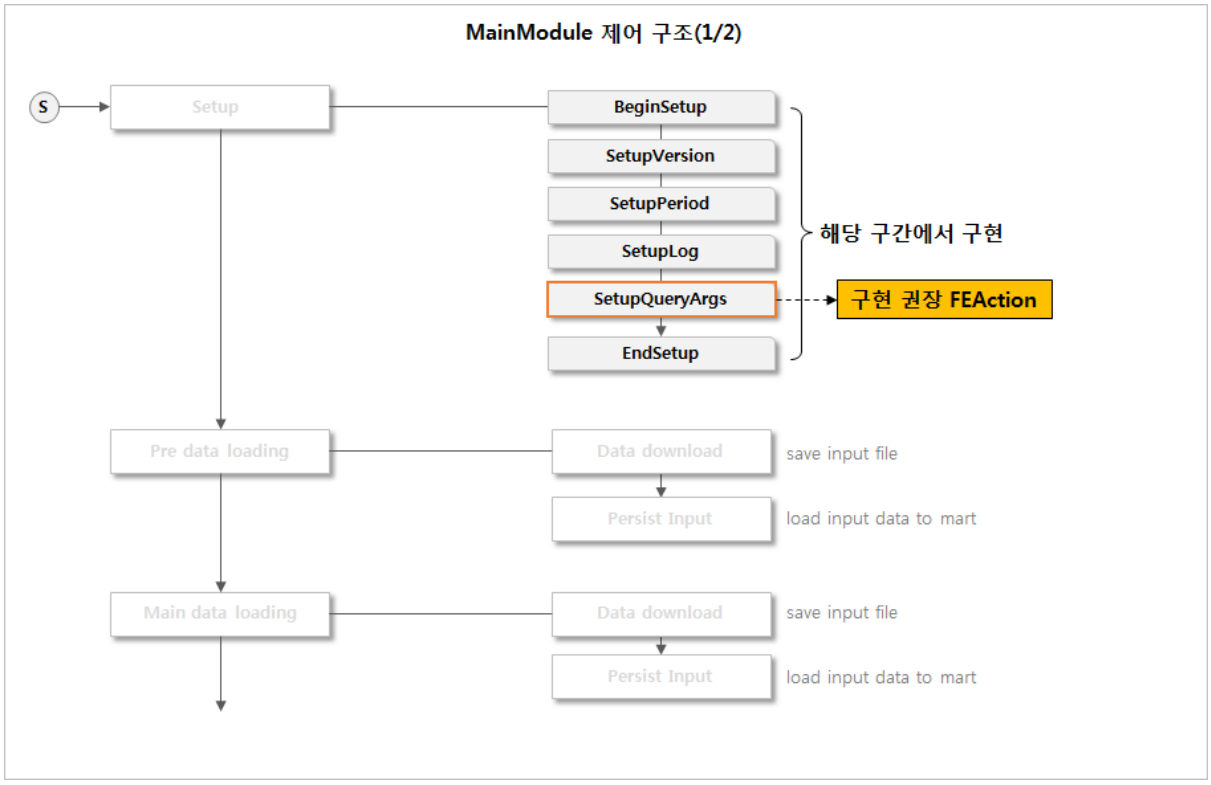


그림 1. Main Module 초기화 Phase 구조

예제

`#commandTimeout` 의 값을 코드로 설정하는 예제입니다. **Main** 모듈의 `SetupQueryArgs` FEAction에서 구현했습니다.

```
public void SETUP_QUERY_ARGS1(ModelTask task, ModelContext context, ref bool handled)
{
    //command timeout을 기본 180초에서 5분으로 증가 (초단위)
    context.QueryArgs["#commandTimeout"] = 3000;
}
```

`ModelContext.QueryArgs` 의 데이터 유형은 Dictionary이며 예제에서와 같이 `#commandTimeout` 문자열을 Key로 해서 시간을 설정하면 됩니다.

Input Arguments로 설정하는 방법

Main 모듈의 `SetupQueryArgs` FEAction의 Default Definition을 사용하는 경우 '$' 아규먼트를 활용하여 `#commandTimeout` 을 **Input Argument**에 등록하면 로직 구현 없이 사용이 가능합니다.

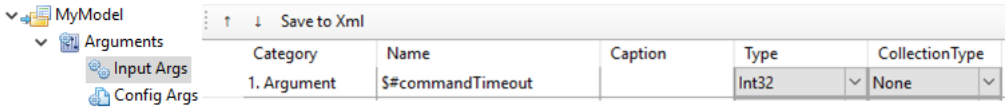


그림 2. Mozart IDE Input Arguments에 등록

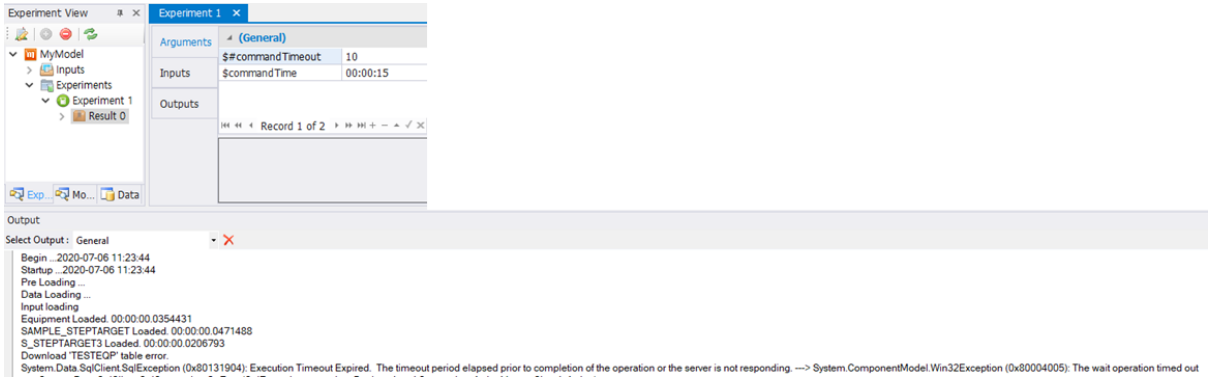
(참고)SetupQueryArgs Default Definition

```
public virtual void SETUP_QUERY_ARGS_DEF(ModelTask task, ModelContext context, ref bool handled)
{
    var args = new Dictionary<string, object>();

    foreach (var it in context.Arguments)
    {
        object value = it.Value;
        if (value == null || (value is string && value.ToString().Length == 0))
            continue;

        //Argument 앞에 '$'가 있는 경우 '$' 삭제 후 Argument로 설정
        if (it.Key[0] == '$')
```

```
        {
            args[it.Key.Substring(1)] = it.Value;
        }
    }

    context.QueryArgs = args;
}
```

결과 - 예시 변경 예정

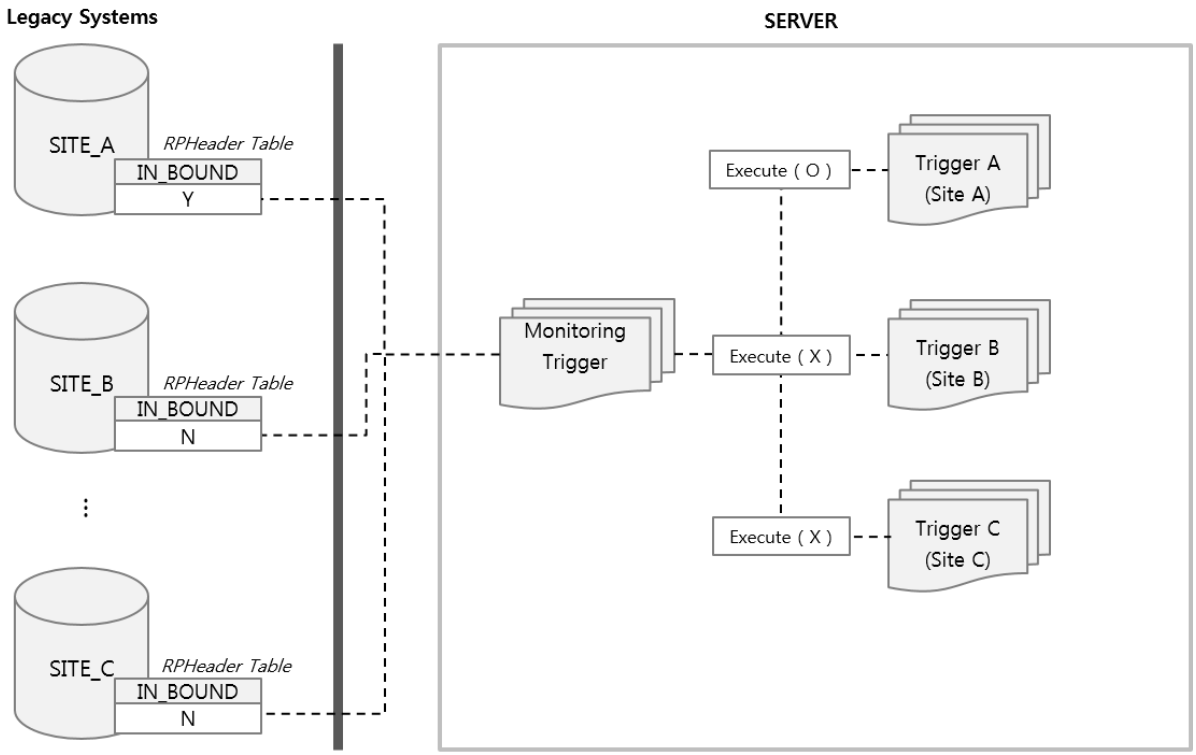
아래는 기본 설정된 시간과 #commandTimeout으로 수행 시간을 늘린 결과의 차이입니다.



선택적으로 Dependent Trigger 를 실행하는 방법

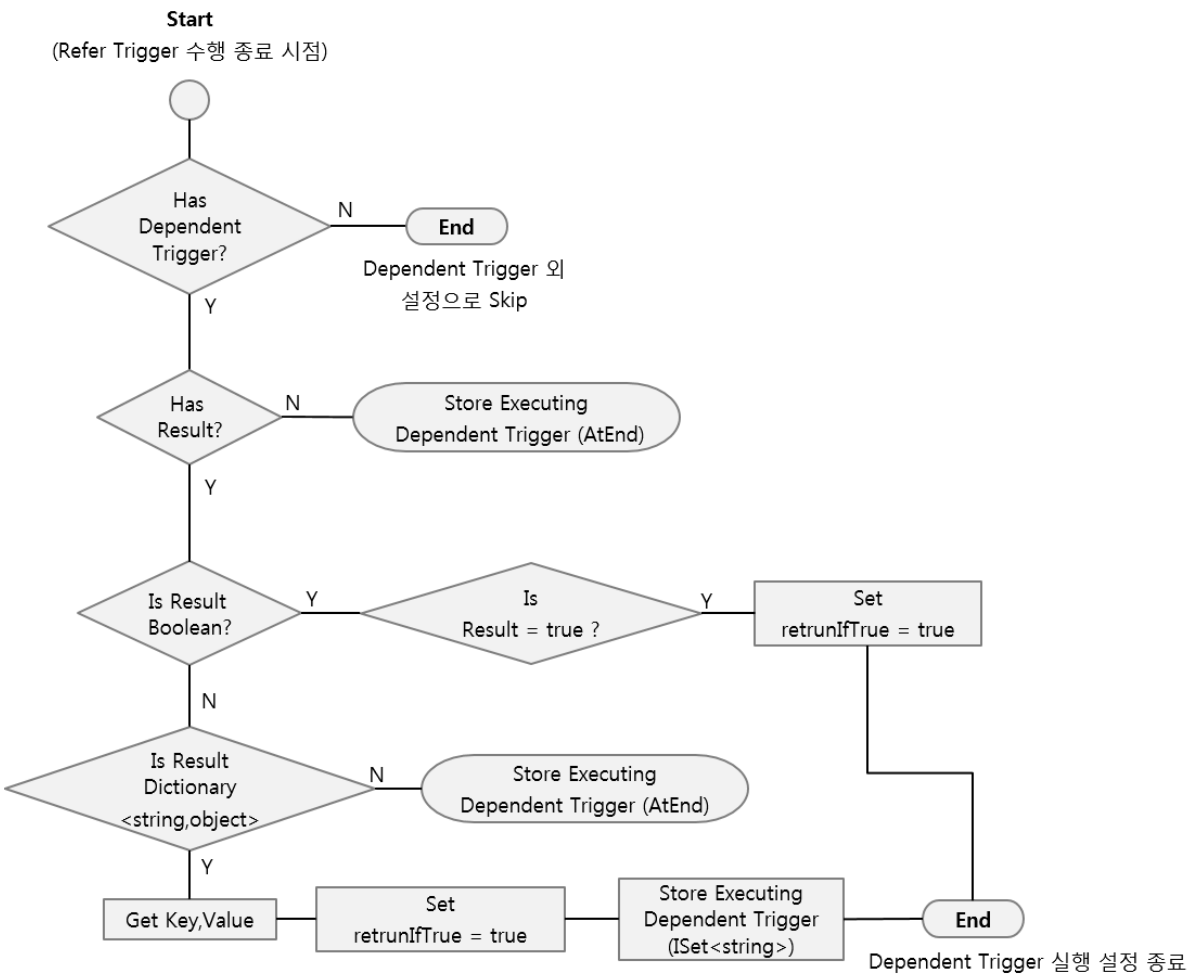
Selective Dependent Trigger 개요-시나리오

운영 중에 여러 Site를 동시에 관리해야 하는 경우가 있으며, 각 Site 별 Data들이 여러 DB로 분류되는 경우가 있습니다. 각 DB를 Monitoring 하기 위한 Task를 생성하여, Inbound 결과에 따라 Monitoring에 Dependent한 Trigger를 선택적으로 수행을 할 수 있습니다.



Selective Dependent Trigger 개요-동작방식

- Reference Trigger에서 **ModelContext.Result = True 를 반환하는 로직이 추가된 경우 해당 Trigger의 Dependent인 Trigger들 중 Execution Type이 ReturnIfTrue인 Trigger들은 다 실행**이 됩니다.
- **ModelContext.Result** 를 **Dictionary** 형식으로 반환 시 Dependent Trigger를 **Key(#exec-dependent-list)** 및 **Value(Trigger 이름)**에 설정된 Trigger만 실행합니다.
- 라이브러리 내부에서는 **Dictionary** 의 **ModelContext.Result**의 **Key(#exec-dependent-list)** 로 반환 시 **ReturnIfTrue = true 로 설정**합니다. 즉, 선택적으로 실행하기 위한 Trigger들의 **Execution Type**은 **ReturnIfTrue**로 설정을 해야 합니다.(AtEnd 및 AtStart는 선택적으로 실행이 불가합니다.)



#exec-dependent-list 사용 방법

사용자 구현 로직

```
[FeatureBind()]
public partial class Main
{
    /// <summary>
    /// </summary>
    /// <param name="context"/>
    /// <param name="handled"/>
    public void RUN1(ModelContext context, ref bool handled)
    {
        Dictionary<string, object> result = new Dictionary<string, object>();
        result.Add("#exec-dependent-list", "TriggerA,TriggerB");

        context.Result = result;
    }
}
```

Dictionary 생성

➔ 선택적으로 Dependent Trigger를 실행하고자 하는 경우 Main 함수에서 **ModelContext.Result**를 **Dictionary<string,object>** 형식으로 반환해야 합니다. 이때 사용하는 Key/Value는 다음과 같습니다.

- **Key** : #exec-dependent-list
- **Value** : Server에 등록된 Trigger의 이름, 구분자는 쉼표(',')를 사용합니다.

라이브러리 구현 내용

```
else if (result is IDictionary<string, object>)
{
    var dic = result as IDictionary<string, object>;
```

⋮

```
else if (dic.TryGetValue(KnownKeys.ExecDepenentList, out value))
{
    if (value is string)
    {
        returnIfTrue = true;

        exec = new HashSet<string>();
        foreach (var t in (value as string).Split(new char[] { ',' }, StringSplitOptions.RemoveEmptyEntries))
            exec.Add(t.Trim());
    }
}
```

#exec-dependent-list가 Key인 경우 라이브러리 내에서는 ModelContext.Result(ReturnIfTrue)를 True로 설정합니다. 이에 **Trigger Setting**에서 **Dependent Trigger**의 **Execution Type**은 반드시 **ReturnIfTrue**로 설정 해야합니다.

실행 할 Dependent Trigger를 **쉼표(',')**로 구분

#exec-dependent-list 사용 방법-Trigger Setting

Dependent Trigger를 선택적으로 실행하고자 하는 Trigger의 Execution Type은 ReturnIfTrue로 설정해야 합니다. 아래와 같이 설정을 하고 ExecDependent 가 실행 종료가 되면 Dependet로 걸려 있는 3개의 Trigger들 중 ServerTest1 및 ServerTest2만 실행이 됩니다.

Trigger Setting

Name	TriggerType	Category	JobName
ServerTest1	Dependent	DB Test	ServerTest - Oracle.ServerTest1
ServerTest2	Dependent	DB Test	
ServerTest3	Dependent	DB Test	

Referred Trigger: ExecDependent

Execution type: ReturnIfTrue

Execution delay: 00:00:03

ExecDependent Trigger 구현 내용

```
public void RUN1(ModelContext context, ref bool handled)
{
    Dictionary<string, object> result = new Dictionary<string, object>();
    result.Add("#exec-dependent-list", "ServerTest1,ServerTest2");

    context.Result = result;
}
```

Trigger 를 특정 버전 DLL 로 실행하는 방법

기능설명

Mozart Server 실행시 특정 버전으로 실행 할 수 있는 기능입니다. 즉, Mozart Server 버전 업그레이드 후 업그레이드 되기 전의 기존 모델들을 돌릴 수 있는 기능입니다.

기능 추가 배경

운영 환경은 안정성이 반드시 보장되어야 합니다. Mozart Server의 버전이 업그레이드가 되면 그 안에서 운영되고 있는 모델과 모델(엔진) DLL도 업그레이드된 버전의 Mozart Server와 호환이 가능하도록 다시 빌드 및 배포가 되어야 합니다. 만약, 예기치 못한 상황에서 새로 빌드된 모델에서 오류가 발생하면 문제 해결 전에는 이전에 안정적으로 운영된 버전으로 롤백을 해야합니다. 여기서 롤백은 모델 및 모델(엔진) DLL 뿐만 아니라, Mozart Server도 이전 버전으로 되돌리는 것을 의미합니다. 버전 롤백의 경우 많은 시간이 소모되며 그 과정에서 Trigger를 실행하는 스케쥴에도 영향을 끼칠 수 있습니다.

그래서 운영 안정성을 보장하면서 롤백에 의한 Trigger 실행 주기 영향 등의 문제를 예방하기 위해 MMC에서 Trigger 실행을 호스팅 할 버전을 지정 할 수 있는 기능들이 있습니다. 해당 기능을 통해 Mozart Server 버전 업그레이드 후 Trigger 실행에 실패하더라도 롤백 과정 없이 이전에 안정적으로 운영된 버전으로 Trigger를 실행 할 수 있습니다.

기능 구성

상이한 버전의 서버를 실행시키기 위한 Argument는 두 가지가 있습니다. 하나는 #host-dir 그리고 또 하나는 #host-version입니다. 두 Argument의 동작방식은 동일하지만 Trigger 실행이 참조하는 위치는 다릅니다. 차이는 다음과 같습니다.

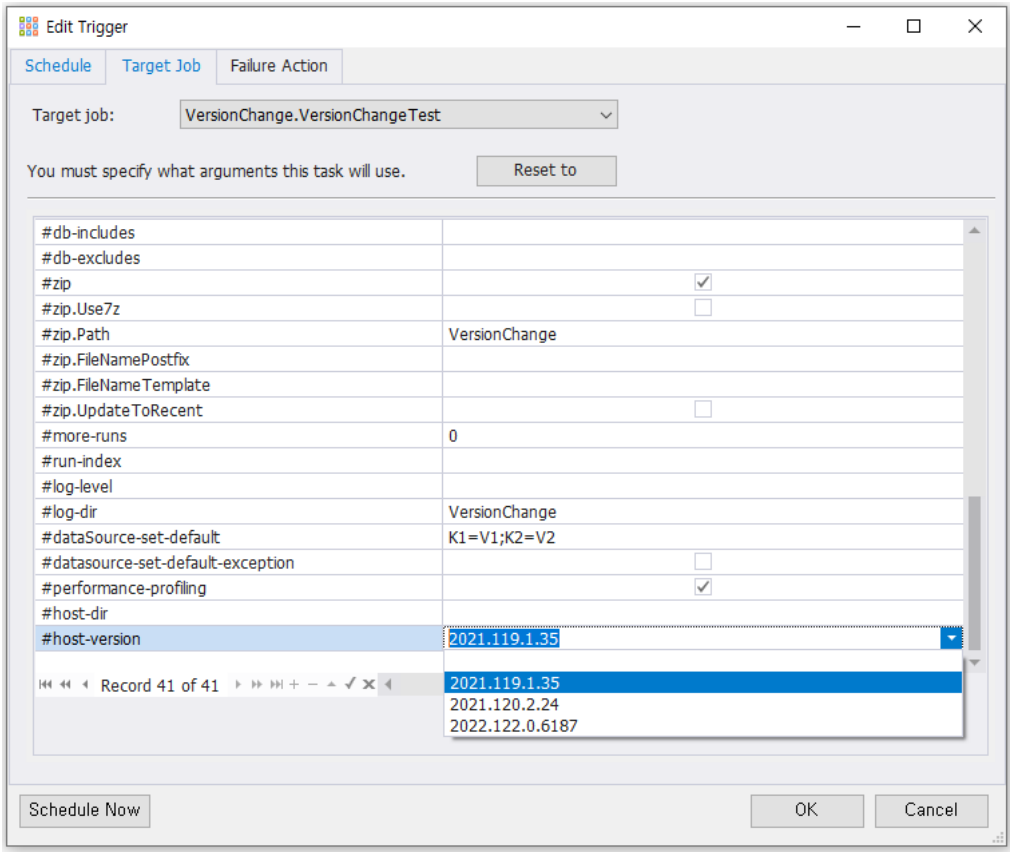
기능 구성

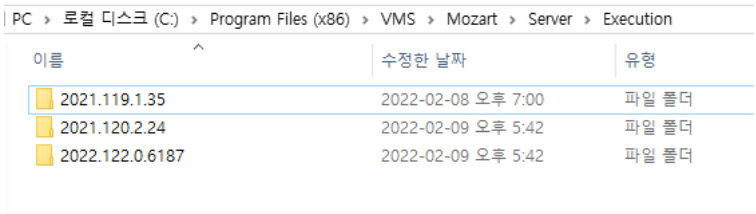
Argument	DataType	Description
#host-version	string	MOZART Configuration for Server에서 사용자가 지정한 Domain Library 설치 경로. 입력값은 제품의 버전 정보입니다.
#host-dir	string	상대경로: WorkingDir 내 서버 폴더 위치 절대경로: 사용자가 직접 입력한 경로

사용방법 및 예시

#host-version

1. **MMC (Mozart Management Console)**를 실행합니다. 실행 파일은 Mozart Client가 설치된 폴더에 있습니다. (i.e *C:\Program Files (x86)\VMS\Mozart\Client\Bin*)
2. Server Explorer에서 접속 할 서버를 선택, 마우스 우 클릭하여 **[Connect Server]**를 선택합니다.
3. 서버 접속에 필요한 정보(ID/PW) 입력 후 서버에 접속합니다. 다음, Server Explorer에서 Triggers를 더블 클릭 또는 마우스 우 클릭, **[Open]**을 선택하여 엽니다.
4. Triggers 탭에서 호스팅 버전을 변경하고자 하는 Trigger를 선택하여 Trigger 편집 다이얼로그를 엽니다.
5. **[Target Job]** 탭으로 이동하여 **[z - extendedProperties]** 카테고리의 목록 마지막까지 내려간 후 **#host-version** Argument에서 드롭다운을 통해 서버에 설치한 도메인 버전 중 호스팅 하려는 버전을 선택합니다.
   (서버의 도메인 설치 경로 : *C:\Program Files (x86)\VMS\Mozart\Server\Execution*)





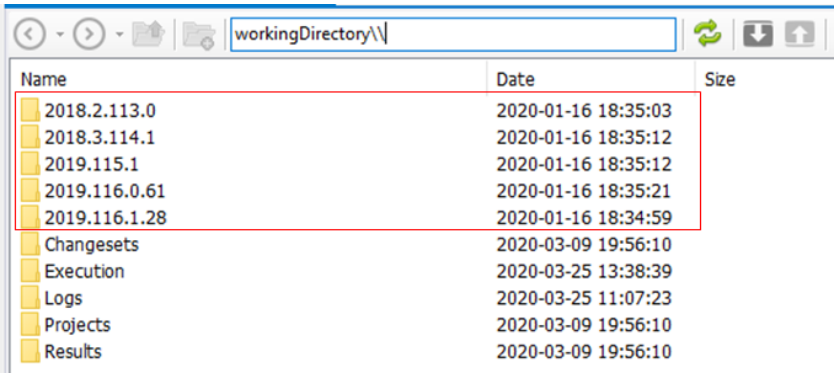
Note

- 도메인 라이브러리는 Mozart Server Configurator에서 설치하며, 서버의 *C:\Program Files (x86)\VMS\Mozart\Server\Execution* 경로에 버전 별로 폴더가 생성됩니다.
- Execution 내 버전 별 폴더는 해당 버전의 Domain Library mdz 패키지 파일을 설치하면 자동 생성됩니다.
- #host-version 설정이 없을 시, 서버에 설치된 도메인 중 가장 최신 버전으로 수행됩니다.

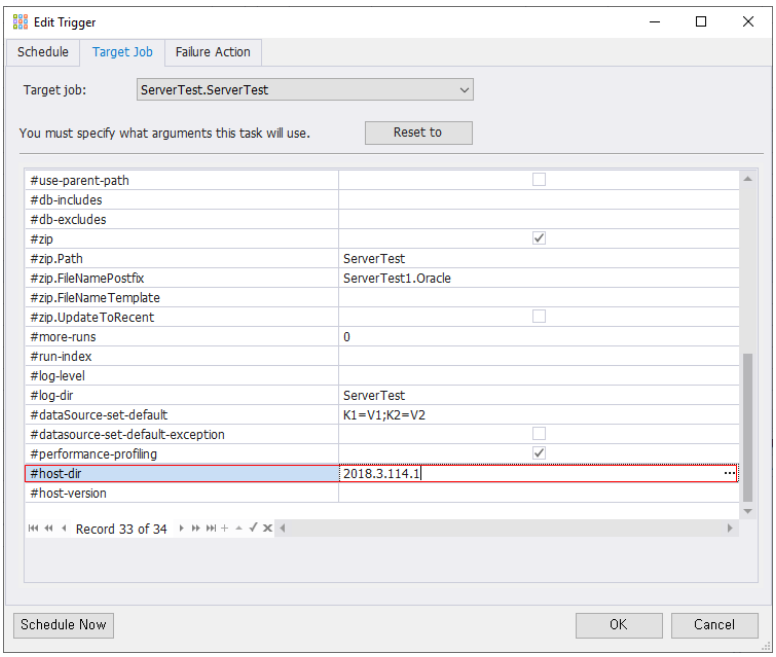
#host-dir

#host-dir 사용방법은 다음과 같으며 과정 중 사전 준비 작업이 포함되어 있습니다. 이 방법은 설치된 Mozart Server가 116.0 미만인 경우에만 사용하시기 바랍니다.

1. Mozart Server의 버전 업데이트를 하기 앞서, 현재 버전의 Mozart Server 폴더를 Working Directory 또는 임의 경로에 복사 합니다. Mozart Server 버전이 116.0 미만인 경우 보통 *C:\Program Files (x86)\VMS\Mozart\Server\Bin*에 설치되어 있습니다. 아래의 샘플은 Mozart Server를 Working Directory에 복사하면서 Bin 폴더의 이름을 버전명으로 변경했습니다.



2. Working Directory에 백업이 완료되었으면, Mozart Server의 버전을 업그레이드 합니다. #host-dir Argument는 Mozart Server 및 MMC 115.0 버전 이상부터 사용이 가능합니다.
3. Mozart Server 및 MMC의 업데이트가 완료되었으면 Mozart Client가 설치된 경로(i.e *C:\Program Files (x86)\VMS\Mozart\Client\Bin*)에서 **MozartManagementConsole2.exe**를 실행합니다.
4. Server Explorer에서 접속 할 서버를 선택, 마우스 우클릭하여 **[Connect Server]**를 선택합니다.
5. 서버 접속에 필요한 정보(ID/PW) 입력 후 서버에 접속합니다. 다음, Server Explorer에서 Triggers를 더블 클릭 또는 마우스 우클릭, **[Open]**을 선택하여 엽니다.
6. 활성화된 Triggers 탭에서 호스팅 서버를 변경하고자 하는 Trigger를 선택하여 Trigger 편집 다이얼로그를 엽니다.
7. **[Target Job]** 탭으로 이동하여 **[z - extendedProperties]** 카테고리의 목록 마지막까지 내려간 후 #host-dir Argument에 Working Directory에 백업해둔 Mozart Server의 폴더명을 입력합니다. 아래는 예제입니다.



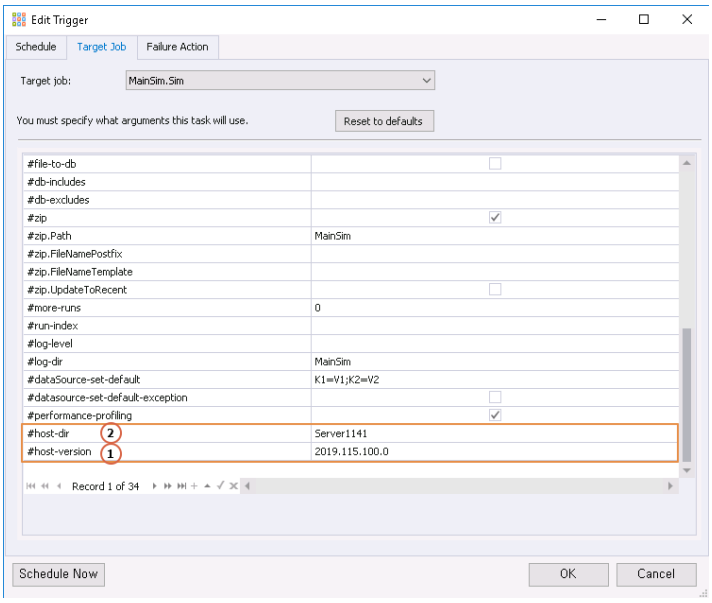
위와 같이 설정하면 해당 Trigger는 현재 설치된 Mozart Server 버전과 관계없이 #host-dir에 지정된 폴더 안에 있는 Mozart Server 및 Domain Library DLL를 참조해서 실행합니다. 단, #host-dir에 설정된 DLL를 참조해서 Trigger를 실행하기 위해서는 대상 모델 및 엔진 DLL 파일이 #host-dir의 버전과 항상 동기화되어 있어야 합니다. 위의 예제에서와 같이 #host-dir로 지정된 폴더의 DLL들이 2018.3.114.1이면, Trigger가 실행하는 모델 및 엔진 DLL 파일은 2018.3.114.1로 빌드되어 있어야 합니다.

실행 우선 순위

1. #host-version
2. #host-dir (1순위 사용자 지정 경로, 2순위 WorkingDirectory)

3. 1,2 값이 없을 경우 Mozart Server 설치 된 Execution 폴더 내 가장 높은 버전명을 가진 폴더(수동으로 폴더를 추가할 경우 버전명 규칙에 맞게 폴더명 생성이 필요합니다.)

   만약 아래 그림과 같이 #host-dir 과 #host-version 의 값이 각각 존재할 경우에는 2019.115.100.0 버전이 실행되게 됩니다.

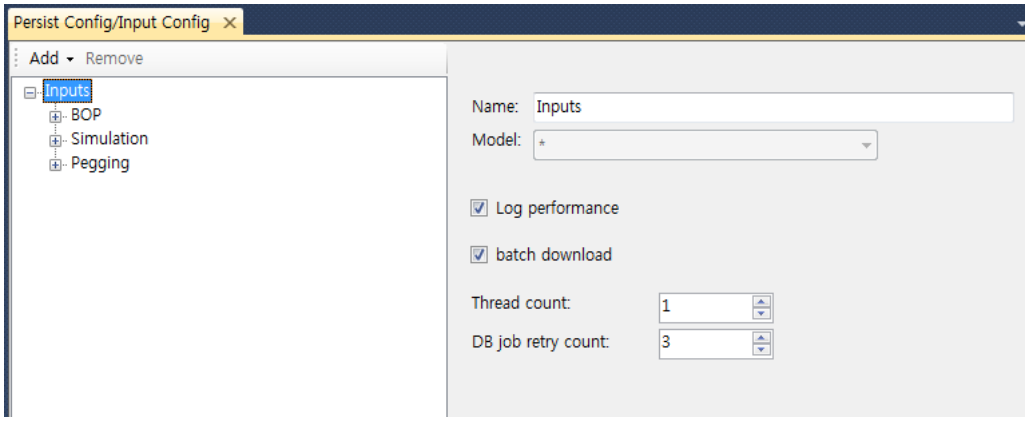


Input Persist Configuration 정의

Persist Config 에서는 Input DataItem 을 Memory 상의 Data 로 로딩하는 작업 절차와 방식을 정의합니다. Input Persist 의 구성요소는 Input Data Group, Input DataItem, Input Save Log 세 가지이며, 전체 Input Loading 시의 옵션은 Persist Config 의 Root Item 에 대한 입력을 통해 설정할 수 있습니다. 실행 메뉴는 MOZART Explorer 의 [Persist Config > Input Config] 입니다.

Input Data 전체 로딩 옵션 설정

최상위 Tree 노드를 선택하면 Data 로딩을 위한 옵션을 정의할 수 있는 메뉴가 활성화됩니다.

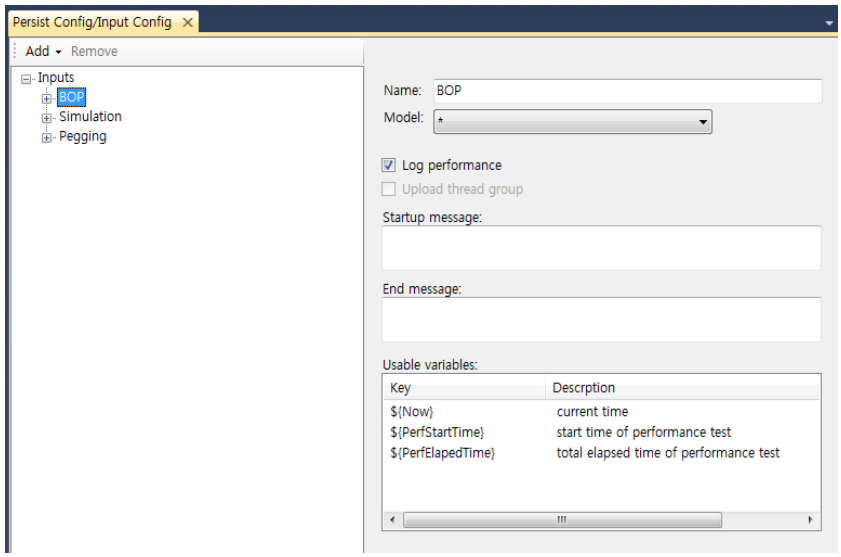


- **Name** : Inputs 전체에 대한 Log 명

- **Model** : 해당 Group 의 설정 내용을 적용할 Model 을 선택합니다. “*” 로 설정하는 경우 모든 Model 에 적용되며, 특정 Model 을 지정하는 경우 해당 그룹 내의 모든 DataItem 에 대한 로딩은 지정된 Model 에서만 Data 를 올리도록 처리합니다.
- **Log performance** : 전체 Data 로딩 소요시간을 기록할지 여부를 설정합니다.
- **Tread count : Batch Download** 를 수행하는 경우 사용할 Tread 수를 지정합니다.
- **DB job retry count : Batch Download** 시 DB 연결, 명령의 재실행 횟수를 지정합니다.

Input Data Group 로딩 옵션 설정

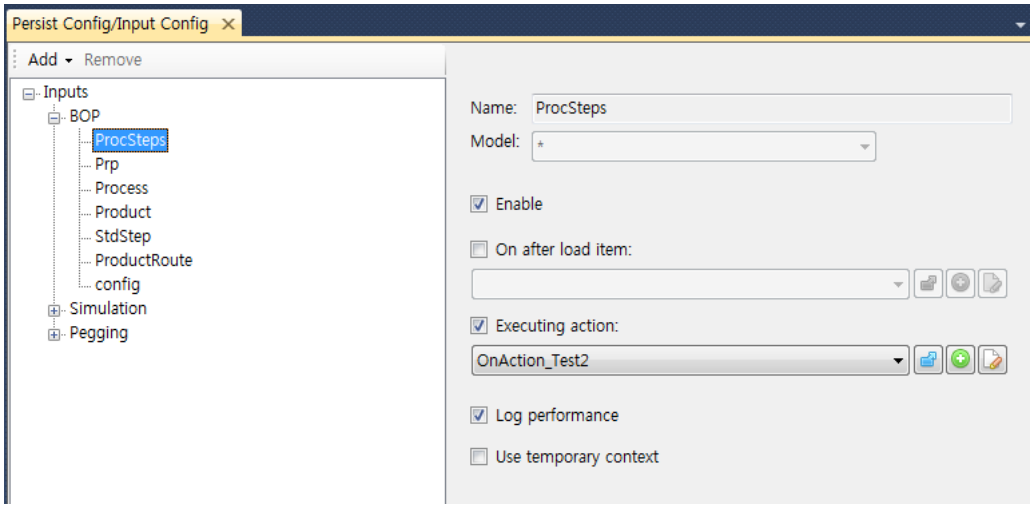
Data Group 별 로딩 옵션을 정의할 수 있습니다. Tree 메뉴에서 Data Group 을 선택하면 우측의 속성창을 통해 속성 설정이 가능합니다.



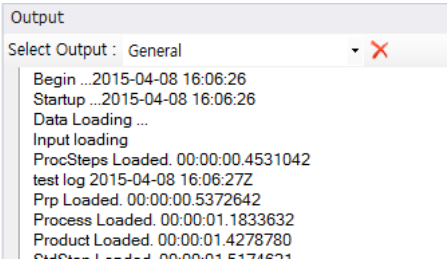
- **Name** : 로딩 시 사용할 Group 명입니다.
- **Log performance**: 전체 Data 로딩 소요시간을 기록할지 여부를 설정합니다.
- **Startup message**: Group 에 대한 로딩 시작 시점에 Log 에 남길 메시지를 편집합니다. 메시지 편집 시 하단의 Usable variables 를 사용할 수 있습니다.
  - **AtBegin** : 동일 시간 발생 Event 들을 처리하기 전에 가장 먼저 처리하는 옵션
  - **AtEnd** : 동일 시간 발생 Event 들을 모두 처리한 후 처리하는 옵션
- **End message**: Group 에 대한 로딩 완료 시점에 남길 메시지를 편집합니다. 메시지 편집 시 하단의 Usable variables 를 사용할 수 있습니다.

Input DataItem 별 로딩 옵션 설정

개별 DataItem 별로 로딩 시 옵션을 정의할 수 있습니다.



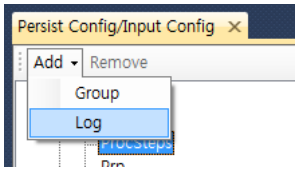
- **Name** : 로딩 시 사용할 DataItem 명으로 변경이 불가합니다.
- **Enable**: Data 로딩 여부를 설정합니다. 옵션을 체크하지 않은 경우 DataItem 이 자동으로 로드되지 않습니다.
- **On after load item** : DataItem 의 각 데이터 Row 별로 데이터의 변환이나 Validation 등을 사용자가 처리할 수 있는 Method 를 실행합니다. 본 항목이 체크된 경우 실행할 Method 는 바로 하단의 버튼을 이용하여 다음과 같은 작업을 수행할 수 있습니다. 함수의 리턴값은 해당 Data Row 의 로드 여부를 의미합니다. true 인 경우 해당 Row 가 로드된 Data 에 포함되며, false 인 경우 Filtering 됩니다. 메시지 편집 시 하단의 Usable variables 를 사용할 수 있습니다.
  - **Action 선택** : 사전에 등록된 Load Method 를 콤보박스에서 선택할 수 있습니다.
  - **Action 추가** : 해당 Data 를 가공하기 위한 함수명과 빈 함수의 소스를 생성합니다. [버튼이름] 버튼을 사용합니다. 함수가 생성되는 파일 명은 PersistInputs.cs 파일이며. Logic 폴더 하위에 생성됩니다.
  - **Action 편집** : 콤보박스에 선택된 함수를 편집하기 위해 소스로 이동합니다. [버튼이름] 버튼을 사용합니다.
  - **Action name 편집** : Action 의 함수 명을 편집하기 위해 사용합니다. [버튼이름] 버튼을 사용합니다.
- **Executing action**: DataItem 의 모든 Data 가 다 로딩 처리 된 후에 별도의 Load 처리를 추가적으로 하려고 하는 경우 Method 를 지정하고 이를 실행시키기 위한 옵션 항목입니다. On after load item 에서 Action 생성 관련 메뉴를 동일하게 사용할 수 있습니다.
- **Log performance**: DataItem 의 로딩 시간 기록여부를 설정합니다. Check 하면 아래그림과 같이 Log 정보에 DataItem 에 대한 로딩 시간이 기록됩니다.

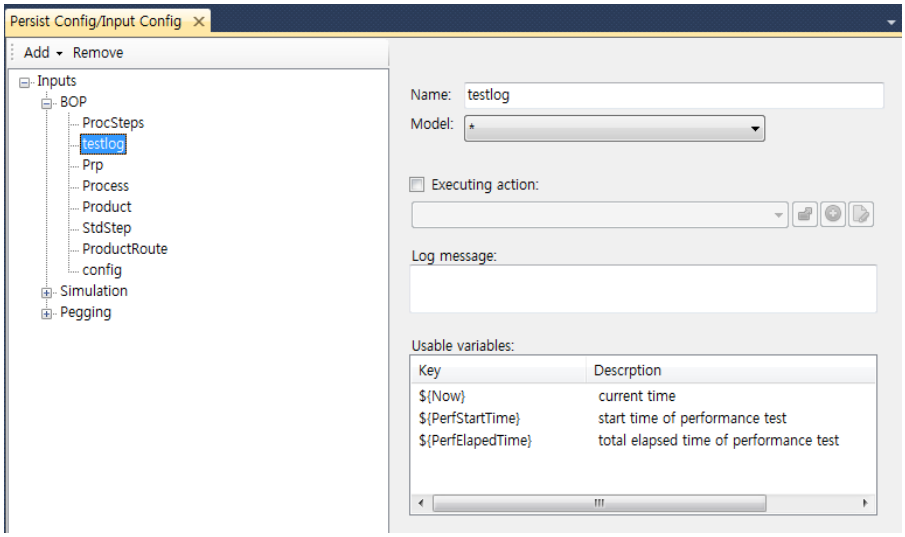


- **Use temporary context**: 일반적으로 DataItem 은 Model 의 실행 주기 내내 Memory 에 로드되어야 하는 Data 와 Data 로딩 시점에 임시적으로 로드했다가 Data 로드가 끝나는 시점에 Memory 에서 삭제해도 상관없는 경우가 있습니다. 본 옵션으로 이를 컨트롤 합니다. 본 옵션이 선택된 경우에는 Input Data Loading 이 완료되는 시점에 Data 가 Memory 에서 삭제됩니다. 생성된 클래스는 TempMart 를 통해 참조가 가능합니다.

일반 Log 생성 옵션 설정

Log Item 은 Data 로딩 시, DataItem 과 관계없이 특정 위치에 Log 를 남기거나 특정 Method 를 실행시키려고 할 때 사용할 수 있습니다. Log Item 은 Persist Config 의 [Add > Log] 메뉴를 통해 추가할 수 있습니다. Tree 특정 노드를 선택한 후 Log 를 추가하는 경우에는 해당 노드의 바로 아래 Log Item 이 생성되며, 속성 페이지를 통해 Log Item 의 옵션을 설정할 수 있습니다.

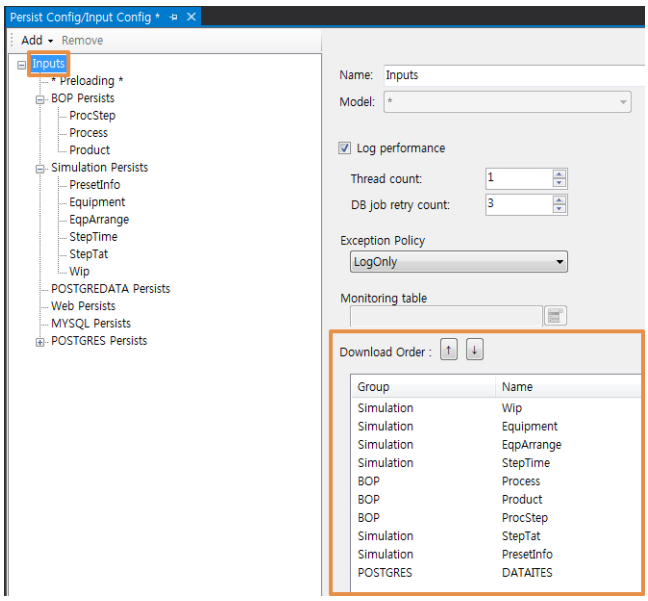




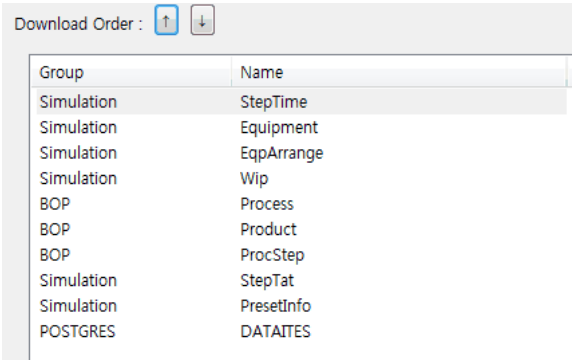
- **Name** : Log 명을 입력합니다.
- **Model** : 해당 항목을 사용할 Model 을 지정합니다.
- **Executing action** : Log 실행 순서에 실행할 Method 를 지정하고 실행 여부를 설정합니다. 본 옵션이 설정된 경우에는 하단에 등록된 Method 가 실행됩니다. 사전에 등록된 Method 를 찾아서 설정하거나, 신규 Method 를 생성 할 수 있습니다.
- **Log message** : Log Item 의 위치에 남길 Log 메시지를 설정합니다. Log 메시지 본문은 Usable variables 에 등록된 Key 를 사용하여 편집이 가능합니다.

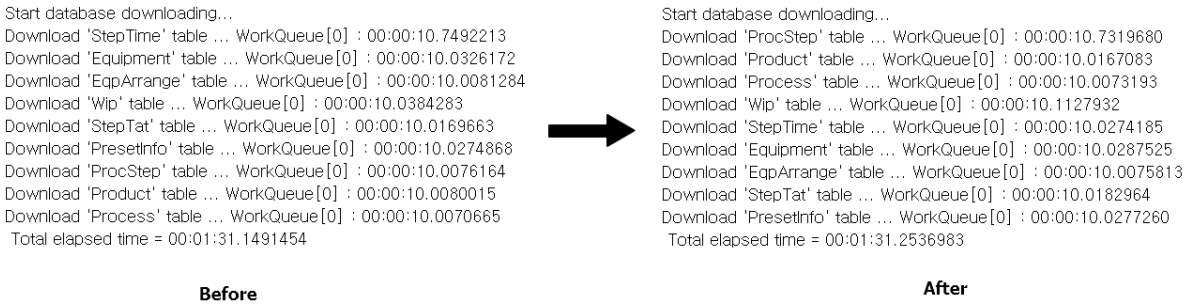
Input Data Download 순서 조절 기능

개별 DataItem간 Database로부터 데이터 다운로드하는 순서를 조정할 수 있습니다.



다운로드 순서는 [Download Order]에서 DataItem을 선택하여 또는 버튼을 누르면서 순서를 변경 할 수 있습니다.





DB Download 순서 변경 전후 모습

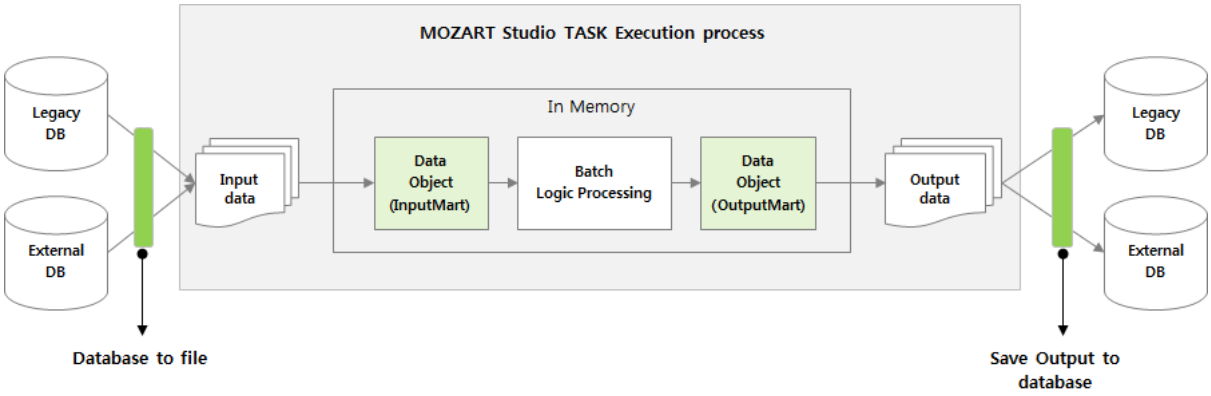
데이터 다운로드 순서와 Persist 단계에서 메모리에 데이터 적재하는 순서는 다르며, 데이터 적재 순서는 좌측 Tree 메뉴에서 순서를 조정합니다.

프로젝트 파일 구조

Project 파일은 Model 정보를 저장하는 .vmodel 파일과 Input 데이터, 결과데이터를 저장하는 폴더와 파일로 구성됩니다. 아래는 Project 를 구성하는 폴더/파일별 설명입니다.

Project 를 구성하는 폴더/파일별 설명

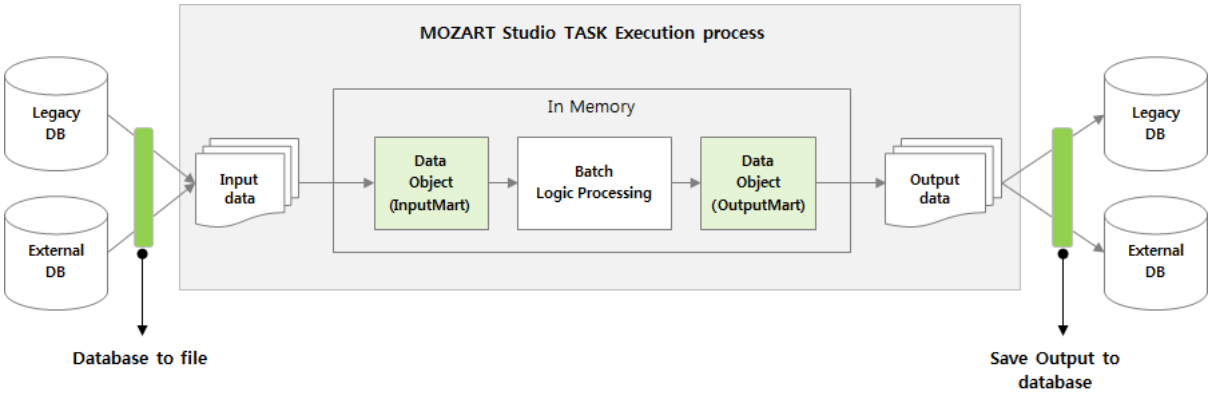
Aa 폴더/파일	설명
./XXX.vmodel	모델의 정보를 저장하고 있는 Binary 파일입니다. 모델 파일은 다음 정보를 포함합니다. - 모델을 구성하는 Input/Output Data Schema, DataAction, Argument, DataSource 정보 - 모델의 실행한 결과가 있는 경우 결과 정보를 포함합니다. 한 모델은 여러개의 Experiment 을 포함하며, 한 Experiment 에 여러 실행 결과를 포함할 수 있습니다.
./Data/[Input Schema name.vdat]	vmodel 파일이 저장되는 폴더에 “Data” 폴더 하위에 각 Input data item 명으로 데이터 파일이 저장됩니다. 각 데이터 파일에는 실행시 사용된 데이터가 저장됩니다. 데이터를 새로 다운로드 받을때마다 업데이트 되어 저장되며, 폴더의 파일을 교체하는 경우 교체된 데이터 셋을 사용하여 실행이 가능합니다.
./[Experiment name]/[Result name]/[Output Schema name.vdat]	모델 실행을 위해서는 Experiment 가 먼저 생성되어야 하며 Experiment 에서 입력 Argument 를 설정하고 실행하면 결과는 실행할때 마다 생성되고 결과 Set(Result) 의 이름을 지정할 수 있습니다. 실행된 결과는 Output 에서 정의한 Dataitem 별로 파일로 저장 됩니다. 한 모델에서 여러개의 Experiment 를 생성할 수 있으며, 한 Experiment 에서 여러 Result 를 생성할 수 있습니다.



상기 그림은 MOZART Studio 에서의 기본 실행 단계를 표시합니다. Studio 에서 실행 할 때는 Input Data 가 있다는 전제 하에 데이터를 메모리에 로딩(Persist)하여 실행하고 결과 파일을 생성하는 과정만 포함됩니다. 결과를 외부의 데이터베이스에 기록하는 것은 포함되지 않습니다. 서버 환경에서 실행될 때는 상기 “Database to file”, “Save Output to database” 두 과정을 일반적으로 자동 실행하지만 Studio 에서 실행할 때는 별도의 메뉴를 사용하여 실행 할 수 있습니다. MOZART Studio persist 기능 을 참조합니다.

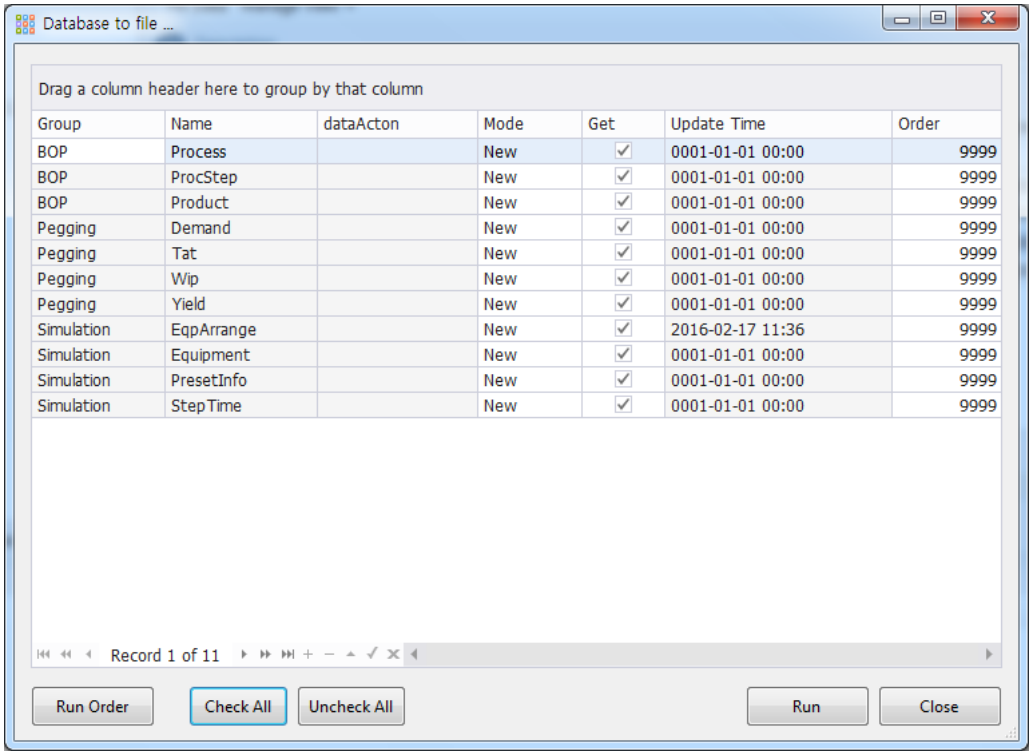
DB 데이터 다운로드 / 저장 방법

MOZART 에서 TASK 실행 구조는 아래 그림과 같습니다. 기본실행 환경에서 외부 데이터와의 Interface 가 발생하는 부분은 실행되지 않으며, 별도의 명령을 통해 실행할 수 있도록 되어 있습니다.



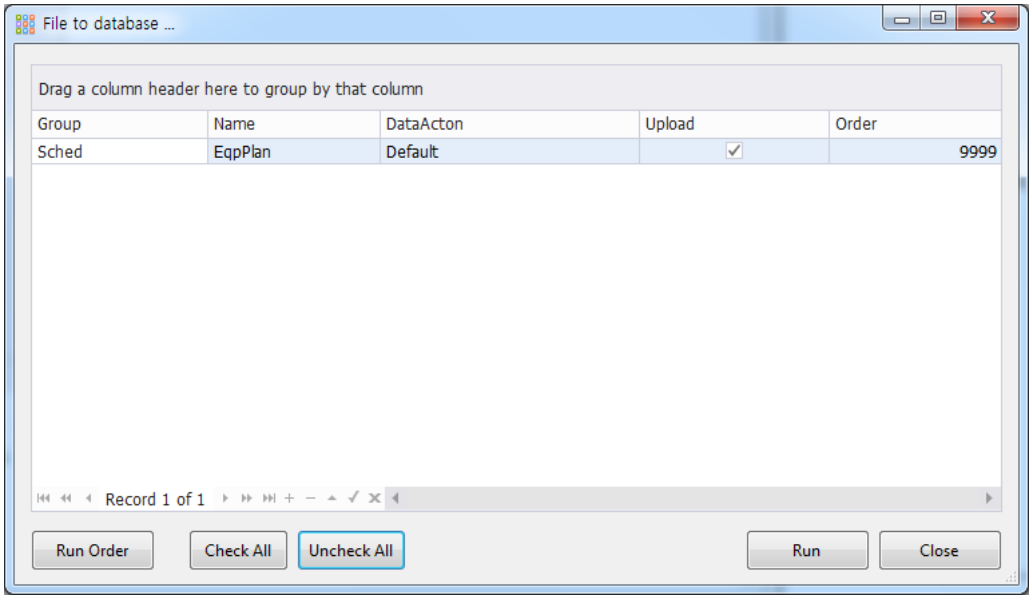
Database to file

외부 데이터베이스로부터 Input data schema 에 맞도록 정의된 쿼리를 실행하여 입력 파일을 생성하는 기능입니다. **[Project/Database to file]** 메뉴를 통해 실행할 수 있습니다. Input data 중 Download 할 데이터 항목을 선택하고 **[Run Order]**버튼을 사용하여 데이터 다운로드 순서를 정의 할 수 있습니다. 데이터 다운로드 시에는 시스템이 로드를 적절하게 분산하여 1~3 개의 Thread 를 사용하여 다운로드 합니다. 설정을 마치고 **[Run]** 버튼을 클릭하면 Download 를 수행합니다.



Save Output to database

생성된 Output 에 연결된 저장 쿼리를 실행하여 외부 데이터베이스에 정의된 Schema 로 데이터를 저장합니다. 먼저 저장이 필요한 Result 를 선택한 후 **[Project/Save Output to database]**메뉴를 통해 실행합니다. 메뉴 실행시 아래와 같은 설정 창이 활성화되며 저장 항목을 설정하여 해당 항목만 저장할 수 있습니다. Output dataitem 에 저장을 위한 DataAction 이 설정되어 있어야 하며 Activate=true 로 활성화되어 있어야 항목이 표시됩니다.



사용자 권한

MOZART Studio의 기능은 사용자 권한에 따라 사용 가능한 기능이 제한되어 있습니다. 기본 설치 파일에는 Download 권한으로 설치가 되며, 권한에 따라 MOZART Studio에서는 모델 수행 결과 조회 및 실행 외에 DataSource, DataItem, DataAction 등의 주요 정보도 조회 및 편집이 가능합니다. 하지만 이러한 정보들은 개발자 혹은 시스템 관리자가 아닌 이상 함부로 정보를 수정해서는 안됩니다. 이러한 부분을 미연의 방지하기 위해 사용자 권한에 따라 일부 기능들이 제한되어 있습니다. 사용자 권한 별 사용 가능한 기능들은 다음과 같습니다.

사용자 권한 별 사용 가능한 MOZART Studio 기능

번호	기능	세부 사항	ReadOnly	Download	Analyst	Power	Admin
1	모델(Project) 열기		O	O	O	O	O
2	모델 닫기		O	O	O	O	O
3	모델 저장			O	O	O	O
4	모델 다운로드		O	O	O	O	O
5	모델 편집					O	O
6	모델 정보 추출					O	O
7	모델 압축				O	O	O
8	Experiment 실행			O	O	O	O
9	Experiment 추가			O	O	O	O
10	Experiment 보기		O	O	O	O	O
11	Experiment 편집			O	O	O	O
12	Experiment 제거			O	O	O	O
13	Experiment 중지			O	O	O	O
14	Batch 추가			O	O	O	O
15	Batch 편집			O	O	O	O
16	Batch 제거			O	O	O	O
17	Result 편집			O	O	O	O
18	Result 제거			O	O	O	O
19	DB Download(Manual)	Database to file Query & Save to File				O	O
20	DB Download(Auto)	use-database	O	O	O	O	O
21	DB Upload(Manual)	Save output to database				O	O
22	DB Upload(Auto)	save-database	O	O	O	O	O
23	Input Argument 보기					O	O
24	Input Argument 편집					O	O
25	Input Argument 추출					O	O
26	Config Argument 보기					O	O
27	Config Argument 편집					O	O
28	Config Argument 추출					O	O
29	DataSource 보기					O	O
30	DataSource 편집						O

번호	기능	세부 사항	ReadOnly	Download	Analyst	Power	Admin
31	DataSource 추출					O	O
32	Schema 보기					O	O
33	Schema 편집					O	O
34	Schema 추출					O	O
35	Schema Report 작성					O	O
36	DataAction 보기					O	O
37	DataAction 편집					O	O
38	DataAction 추출					O	O
39	Input Data 보기		O	O	O	O	O
40	Input Data 편집			O	O	O	O
41	Input Data 추출			O	O	O	O
42	Output Data 보기		O	O	O	O	O
43	Output Data 편집					O	O
44	Output Data 추출			O	O	O	O
45	DB 연결					O	O
99	Studio 종료		O	O	O	O	O

USER GUIDE

Studio 메뉴 구성

Studio 의 기본 메뉴는 아래와 같이 구성됩니다. 추가적으로 실행 구조에서 볼 수 있듯이 분석 UI 를 추가하면 아래의 기본 메뉴에 분석 UI 에서 사용할 수 있는 메뉴들이 추가됩니다.

메뉴 구분	메뉴	메뉴 설명
File	-	프로젝트 파일의 Open, 저장 및 원격서버의 프로젝트 파일 다운로드 등과 관련된 메뉴 집합입니다.
File	Open Project	프로젝트 파일을 엽니다. 파일 선택창을 통해 .vmodel 파일을 지정하여 열 수 있습니다.
File	Close	열려있는 창을 닫습니다.
File	Close Project	열려있는 프로젝트를 종료합니다.
File	Save	선택된 창의 데이터를 저장합니다.
File	Save All	프로젝트 전체를 저장합니다.
File	Save As Project	프로젝트 전체를 다른 이름으로 저장합니다. 프로젝트 저장하기를 참조합니다.
File	Recent File List	최근 열었던 프로젝트 리스트를 표시하고 선택시 해당 프로젝트를 엽니다.
File	Download Data From Server…	MOZART 서버에 연결하여 프로젝트 압축파일을 다운로드하거나 다운로드 후 자동으로 프로젝트를 엽니다. “ **모델 다운로드 하여 열기** “를 참조합니다.
Edit	-	그리드의 복사/붙이기 및 검색 등 일반 메뉴에 대한 집합입니다.
Project	-	프로젝트의 모델 편집 및 실행과 관련된 메뉴 집합입니다.
Project	Refresh Item	Content
Project	Open Item(Read Only-Grid)	선택한 Data Item을 읽기 전용으로 확인 할 수 있도록 Grid 형태의로 창이 열립니다. 설명은 Open Item을 참조합니다.

메뉴 구분	메뉴	메뉴 설명
Project	Open Item(Read Only-Pivot)	선택한 Data Item을 읽기 전용으로 확인 할 수 있도록 Pivot Grid 형태의 창이 열립니다. 설명은 Open Item을 참조합니다.
Project	Delete Item	Experiment 및 결과 집합, 결과 데이터를 삭제 합니다. Model Tab View 에서는 Input, Output, Datasource 등의 Tree node item 에 대해 삭제할 수 있습니다.
Project	New Experiment	새로운 Experiment 를 생성합니다.
Project	New Batch	새로운 Batch Experiment 를 생성합니다. Batch Experiment 의 설명은 Batch Experiment 를 참조합니다.
Project	Run Experiment	등록된 Experiment 를 실행합니다.
Project	Run Experiment with Links	모델연결이 설정된 경우 연결된 모델을 함께 실행합니다.
Project	Terminate	모델 실행을 중단 합니다.
Project	Import	Input/Output data item 을 import 합니다. import 한 파일은 파일명이 같은 Input 이 있는 경우에는 자동으로 덮어쓰게 되며, 동일한 파일명의 input data 가 없는 경우 생성됩니다. 만일 해당 폴더가 없는 경우 폴더까지 자동 생성됩니다.
Project	Export	Input/Output data item 을 export 합니다. export 한 파일은 .vdif 파일로 저장됩니다
Project	Database to File	Input data item 에 연결된 DataAction 을 사용하여 데이터를 다운로드 하여 파일로 저장합니다. Download & Save functions 를 참조합니다.
Project	Save Output to database	Output data item 에 연결된 DataAction 을 사용하여 output 데이터를 target database 에 저장합니다. Download & Save functions를 참조합니다.
Project	Report schema	Input/Output schema 및 dataaction 을 XML 로 출력합니다. 모델 Document 출력 을 참조합니다.
Project	View Dataitem by Database	특정 Data Item 을 선택한 후 실행시키는 경우 해당 Data Item 에 사용된 Datasource 를 기준으로 동일한 Datasource 를 사용하는 Data Item 들을 그룹핑하여 Tree 에 표시합니다. Model Tab View 에서만 동작합니다. 모델 검토 기능을 참조합니다.
Project	Zip project files…	현행 Input, Output 및 모델 파일을 압축합니다. 메뉴를 실행하면 저장할 폴더/파일을 지정한 후 저장할 수 있습니다.
Project	Open model folder in window explorer	윈도우 탐색기를 통해 모델파일이 저장된 장소로 이동할 수 있습니다.
Project	Vdat File Information	Data Item 을 선택하면 메뉴가 활성화되며, 선택한 Data Item의 실제 데이터 파일(확장자 : vdat) 속성 창을 보여줍니다.
Data	Import	Input, Output 데이터를 외부 데이터로 부터 채우기 위한 기능입니다. Excel, Text file 두가지 옵션이 있습니다. 자세한 내용은 Data Export/Import 를 참조합니다.
Data	Export	Input, Output 데이터를 특정 데이터 형태로 Export 합니다. Data Export/Import 를 참조합니다.
Tool	-	MOZART Studio의 확장 기능을 관리하거나 실행 Option 을 설정합니다.
Tool	Run Add-In	등록된 확장 중 특정 기능을 실행합니다
Tool	Organize Add-Ins…	MOZART Studio 에 확장기능을 등록하거나 삭제하는 등의 관리를 수행합니다.
Tool	Options…	MOZART Studio 의 자동 업데이트 설정 및 모델 다운로드 서버 설정 등의 사용자 옵션을 정의합니다. Studio Option 설정을 참조합니다.
Tool	Copy Tool	Data Item에서 복사한 데이터의 텍스트 포맷을 설정합니다. 자세한 내용은 Grid View의 데이터 Copy Tool 사용법을 참조합니다.
Help	License Information	등록된 라이선스 정보 및 EULA 를 조회할 수 있으며, Mozart Product Activation Tool을 활성화 합니다. 라이선스 Activation 방법은 License Activation 을 참조합니다.

프로젝트 열기 / 저장하기

프로젝트 열기

MOZART Project를 열기 위해서는 두 가지 방식을 사용할 수 있습니다. Local 혹은 원격 폴더에 위치한 .vmodel 파일을 지정하여 여는 방식과 서버에서 수행된 시점 별 모델을 조회한 후 특정 모델을 다운로드 받아 여는 방식이 있습니다.

Open project from file

다음과 같은 방식으로 프로젝트를 열 수 있습니다.

1. **[File/Open Project]** 를 사용하여 .vmodel 파일을 지정하여 Project 를 엽니다.
2. vmodel 파일을 Drag 하여 Tree View 의 상단에 Drop 하여 Project 를 엽니다.
3. **[File/Recent File List]** 메뉴를 사용하여 최신 사용한 Project 를 조회하고 특정 Project 를 선택하여 파일을 엽니다.

Open project from Server

Downloads 서버를 등록한 경우에 서버에 운영 중인 모델을 조회하고 모델을 지정하여 Download 하거나 Download 한 후 모델을 자동으로 열 수 있습니다. Download 서버 설정은 Options 설정을 참조합니다.

1. **[File/Download data from server]** 메뉴를 선택하거나 Toolbar 의 아이콘을 선택합니다.
2. **[Download Data From Server]** dialog 에서 서버를 선택합니다.
3. 해당 서버에서 수행된 이력을 조회하고 Download 할 모델을 선택합니다.
4. Download 버튼을 클릭하며 모델을 다운로드 합니다. 이때, 모델 파일을 다운로드 받을 경로를 설정합니다. 기본 다운로드 경로는 **[Download Path]**에서 설정합니다.
   + **[Auto Classify]** 체크박스를 체크하면 설정된 Download Path 하위에 서버별 폴더를 생성하여 모델을 저장합니다.
5. 로컬 저장 경로에 서버에서 다운로드 하려고 하는 데이터와 동일한 이름의 폴더가 존재할 때 덮어쓰기 여부를 선택합니다.
   + **[Yes]** : 해당 폴더에 모델과 데이터들이 덮어쓰기 됩니다.
   + **[No]** : 이름이 변경되어 저장됩니다. 이름은 "원래 폴더 이름(1)"로 저장되며, 괄호안의 숫자가 존재할 경우에는 다음 숫자로 저장됩니다.
   + **[Cancel]** : 다운로드가 취소됩니다.
   + **[Apply to all items]** 체크박스를 체크하면 2개 이상의 모델을 다운로드 할 때 동일한 덮어쓰기 여부를 적용할 수 있습니다.

> 💡 **Note**
> 모델 파일 덮어쓰기 기능은 2021.118.3 버전부터 제공됩니다.

프로젝트 저장하기

프로젝트를 저장하기 위해서는 3가지 방식을 이용할 수 있습니다. MOZART Studio에서 활성화 된 창을 저장하는 방법(Save), 모든 열려진 창을 저장하는 방법(Save All), 모델파일과 모든 데이터(Input, Output)를 다른 이름으로 저장하는 방법(Save As Project)이 있습니다.

Save

현재 연결된 모델에 활성화 된 창에 연결된 정보만을 저장합니다. 비 활성화 된 창의 수정된 정보는 저장되지 않습니다.

Save All

현재 연결된 모델에 수정된 모든 정보를 저장하는 기능입니다. 활성화/비 활성화 구분 없이 모든 수정 사항이 저장됩니다.

Save As Project

Studio에서 연결한 모델 파일을 포함한 프로젝트(Data, Experiment 결과 폴더 포함)를 다른 이름으로 저장하기 위한 기능입니다.

1. Studio에서 모델을 엽니다.
2. [File/Save As Project] 를 눌러 Save As Project Dialog 창을 띄웁니다.

3. 저장이 필요한 폴더를 지정하고, 모델명을 지정하여 [저장]을 누릅니다.
4. 저장하는 폴더 위치에 동일한 이름의 모델(vmodel), 데이터(Input/Experiment 데이터)가 존재하는 경우, 덮어쓰기 여부를 결정하여 저장합니다.
5. 저장이 완료되면 새롭게 저장된 모델로 연결이 변경됩니다.

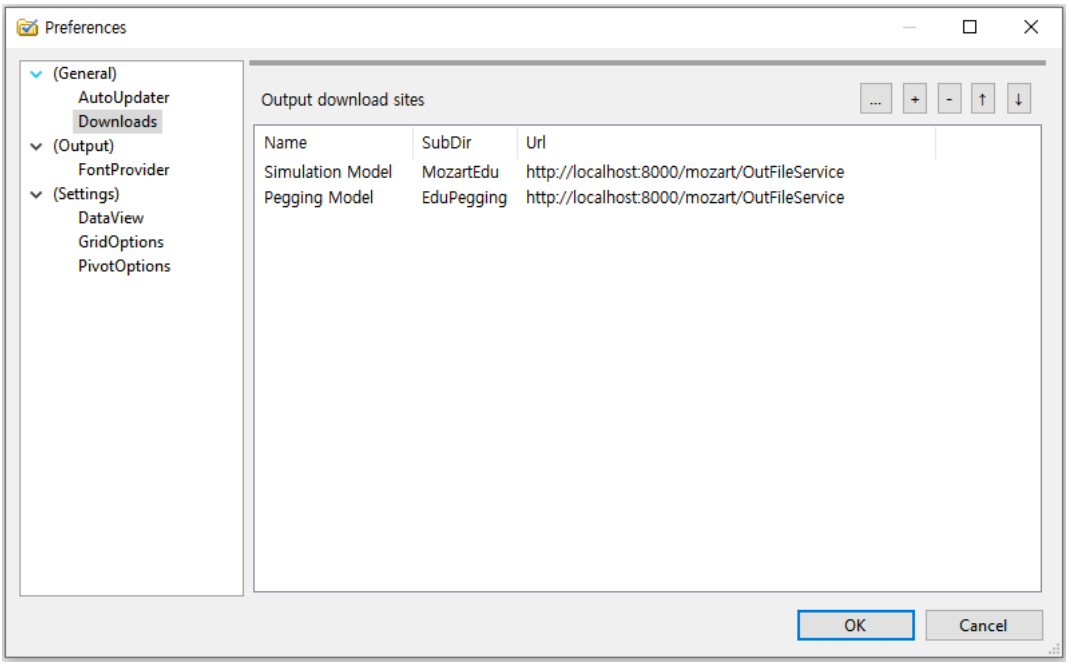
> **Note**
> **Save As Project** 메뉴는 2021.118.3 버전부터 제공됩니다.

Tool Option

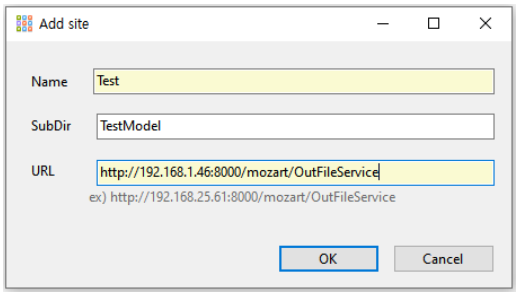
MOZART STUDIO 의 사용자 환경 설정은 **[Tool/Options ...]** 메뉴를 통해 실행 할 수 있습니다.

Model Download Setting

Client 에서는 Model Download Server 와 해당 Server 에서 Model 별 저장 폴더를 지정해주어야 합니다. 설정 방식은 각 사이트에서 구매한 **OOO_Studio** 를 실행하여 **[Tool > Options]** 메뉴를 통해 설정합니다.



- 왼쪽 Tree 에 Downloads 를 선택합니다.
- 오른쪽 리스트에 Download 사이트를 추가합니다. 추가할 때는 상단의 [+] 버튼을 이용합니다.



- **Name** : 추후 Model Download 시 콤보박스에 표시되는 사이트 명입니다.
- **SubDir** : Server 에 Model 이 저장되어있는 폴더명입니다. Server 에서 설정한 Model Download Base Folder에서부터 상대 경로로 지정됩니다.
- **URL** : Model 파일 Download 서비스를 제공하는 Server 의 서비스 URL 입니다. 형식은 위의 예시와 같은 방식으로 설정하며, 사용자 사이트의 운영팀을 통해 확인한 URL 을 입력합니다. 일반적으로 Port 설정과 IP 가 다를 수 있습니다. 이는 Server Setup 후 운영팀을 통해 확인합니다.
- 여러 개의 사이트를 등록할 수 있으며 상단의 화살표로 Download 창에서 표시 순서를 조정할 수 있습니다.
- 등록된 사이트의 내용을 수정하고자 할 때는 더블 클릭하거나, 상단의 **[...]** 버튼을 사용합니다.
- 추가한 대상을 삭제하고자 하는 경우에는 **[-]** 버튼을 사용합니다.

상기 예시와 같이 하나의 Server 에 여러 개의 폴더를 나누어서 등록하거나 여러 개의 Server 를 등록 할 수 있습니다. 위와 같이 설정한 후 Studio 에서 **[File > Download Data From Server]** 메뉴를 통해 Model 검색 창을 띄워 Model 을 Download 할 수 있습니다.

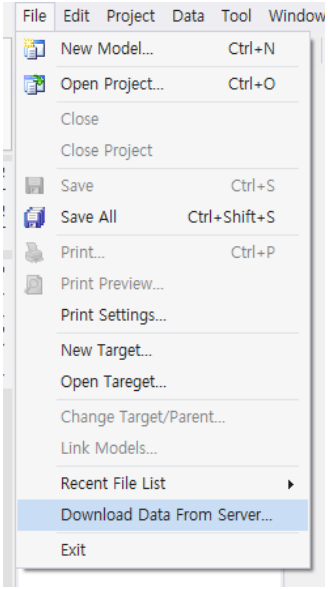
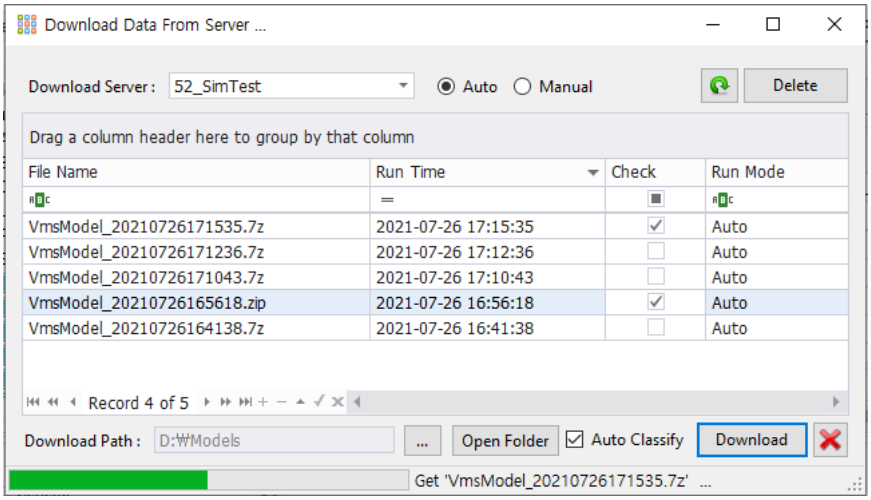


그림 3. Model Download를 위한 Download Data From Server 메뉴

메뉴를 실행하면 아래 그림과 같이 Download Server 콤보박스에 등록한 Name 별로 리스트가 뜨고 해당 리스트를 선택하는 경우 해당 폴더에서 Download 가능한 Model 의 리스트를 조회하여 Download 할 수 있습니다.

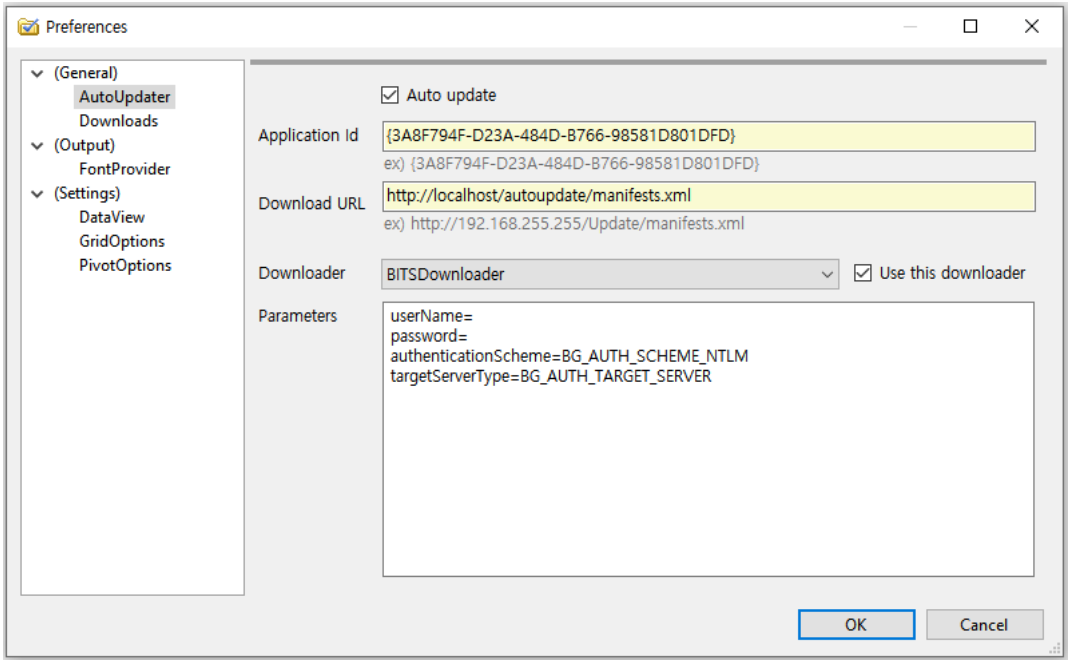


- **Download Server** : Download Server의 Site Name 입니다.
- **Auto, Manual** : Download Server 대상이 자동(Auto)/수동(Manual) 인지를 선택합니다.(기본값 : Auto)
- **Refresh** : 해당 서버의 파일들을 Refresh 합니다.
- **Delete** : 체크 선택한 Model 파일들을 삭제합니다.
- **Download Path** : 다운로드 파일을 저장 할 경로입니다.
- **Open Folder** : Download Path에 정의된 폴더를 탐색기에서 열어줍니다.
- **Auto Classify** : Download Path를 기준으로 Download Server 이름으로 모델 위치를 자동 구분 여부를 선택합니다.
- **Download** : 체크 선택한 Model 파일들을 Download 합니다.
- **X 버튼**(Download 옆) : **[Downlaod]** 버튼을 눌러 Download 진행 시 활성화 되며, Download 진행을 강제 중단합니다.
- **모델 다운로드 진행 상태바** : 모델 다운로드시 다운로드하는 모델의 다운로드 진행 정보 상태가 표시됩니다.

Auto Update Setting

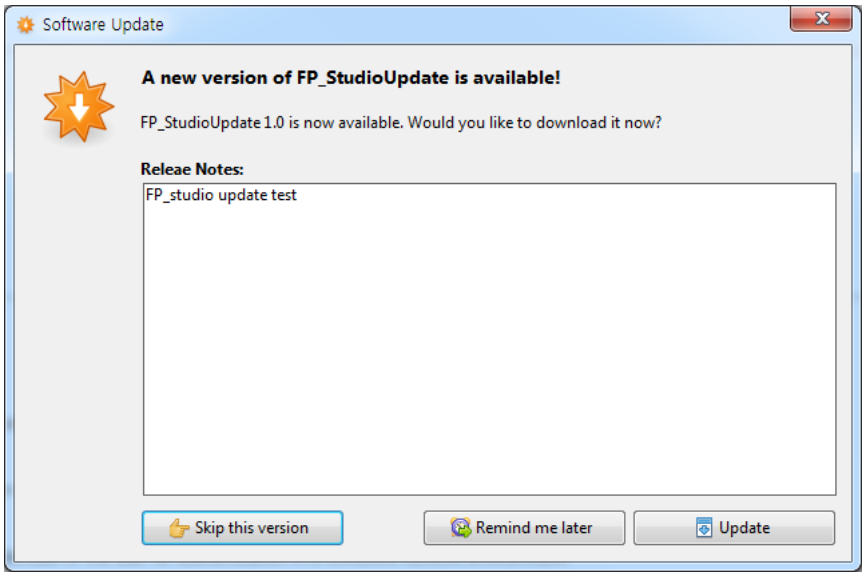
아래와 같은 순서로 MOZART Studio 에서 클라이언트 UI 및 관련 파일에 대한 자동 업데이트 설정을 할 수 있습니다.

1. Client 에서는 Studio 의 **[Tool > Options]** 메뉴를 사용하여 Auto Update 여부 및 업데이트 Server 를 설정합니다.
2. 설정항목은 아래와 같이 설정합니다.



- **Auto update** : 체크를 하면 하부에 입력된 정보를 기준으로 자동 업데이트가 되며, 체크해제 시 자동 업데이트를 하지 않습니다.
- **Application Id** : Studio 프로그램의 고유 아이디입니다. *사용자가 변경하지 않습니다*. Server 의 Manifest 파일과 비교할 때 대상 Application 에 대한 업데이트 정보만 비교합니다.
- **Download URL** : Update Server 의 URL 을 입력합니다. 형식은 다음과 같습니다.
- 형식 : http://[Server IP]/[Server 설정 시 응용 프로그램 별칭]/manifests.xml
- 변경부분은 [Server IP], [Server 설정 시 응용 프로그램 별칭] 입니다. Client UI 개발/운영조직에게 해당 부분을 확인하고 설정합니다.
- **Downloader** : Downloader 를 선택합니다. 기본은 BITDownloader 가 선택되어 있습니다.
- **Parameters** : Server 와의 인증에 사용되는 Parameters 로 수정하지 않습니다.

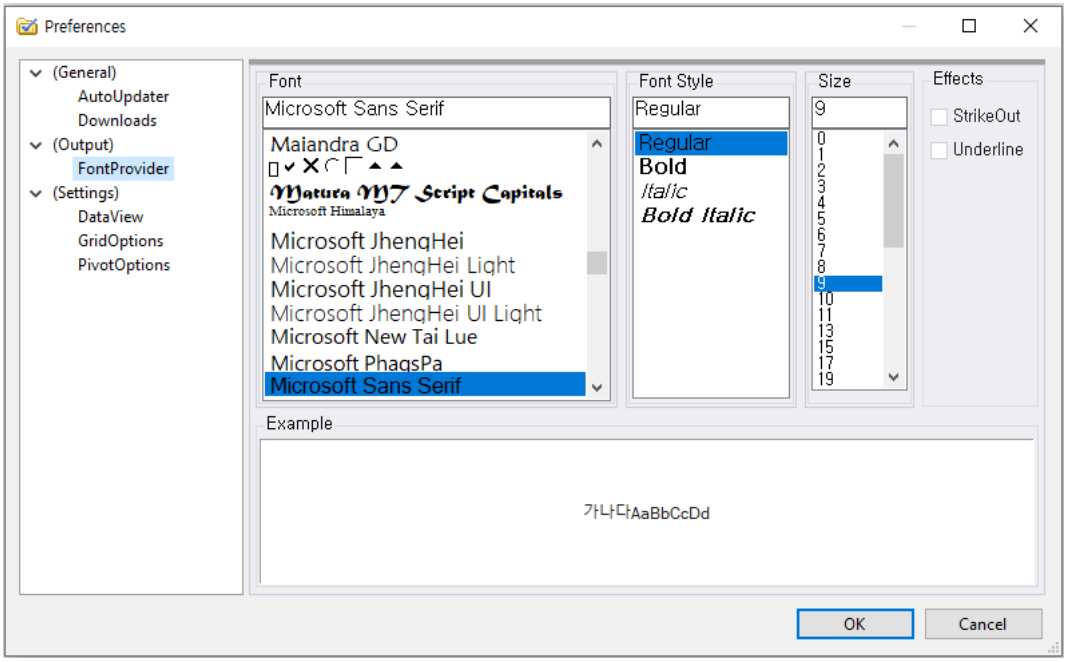
3. 항목 설정 후 Studio를 재시작하면 아래와 같은 다운로드 선택창이 활성화됩니다.

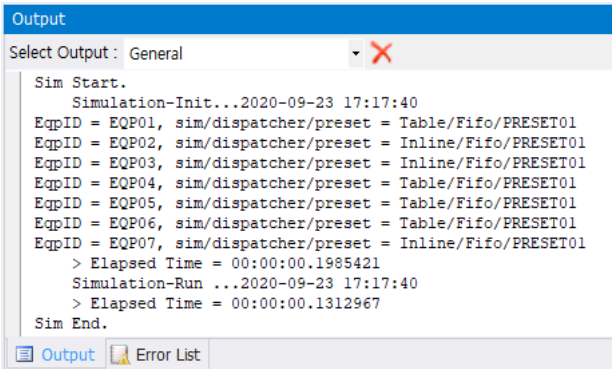


- **Skip this version** : 다음번 실행 시에도 해당 버전의 업데이트를 체크하지 않습니다.
- **Remind me later** : 다음번 실행 시에 해당 버전의 업데이트 수행 여부를 묻습니다.
- **Update** : 업데이트 내용을 다운로드 받고 Studio 를 재실행합니다.

Studio 출력창 폰트 설정

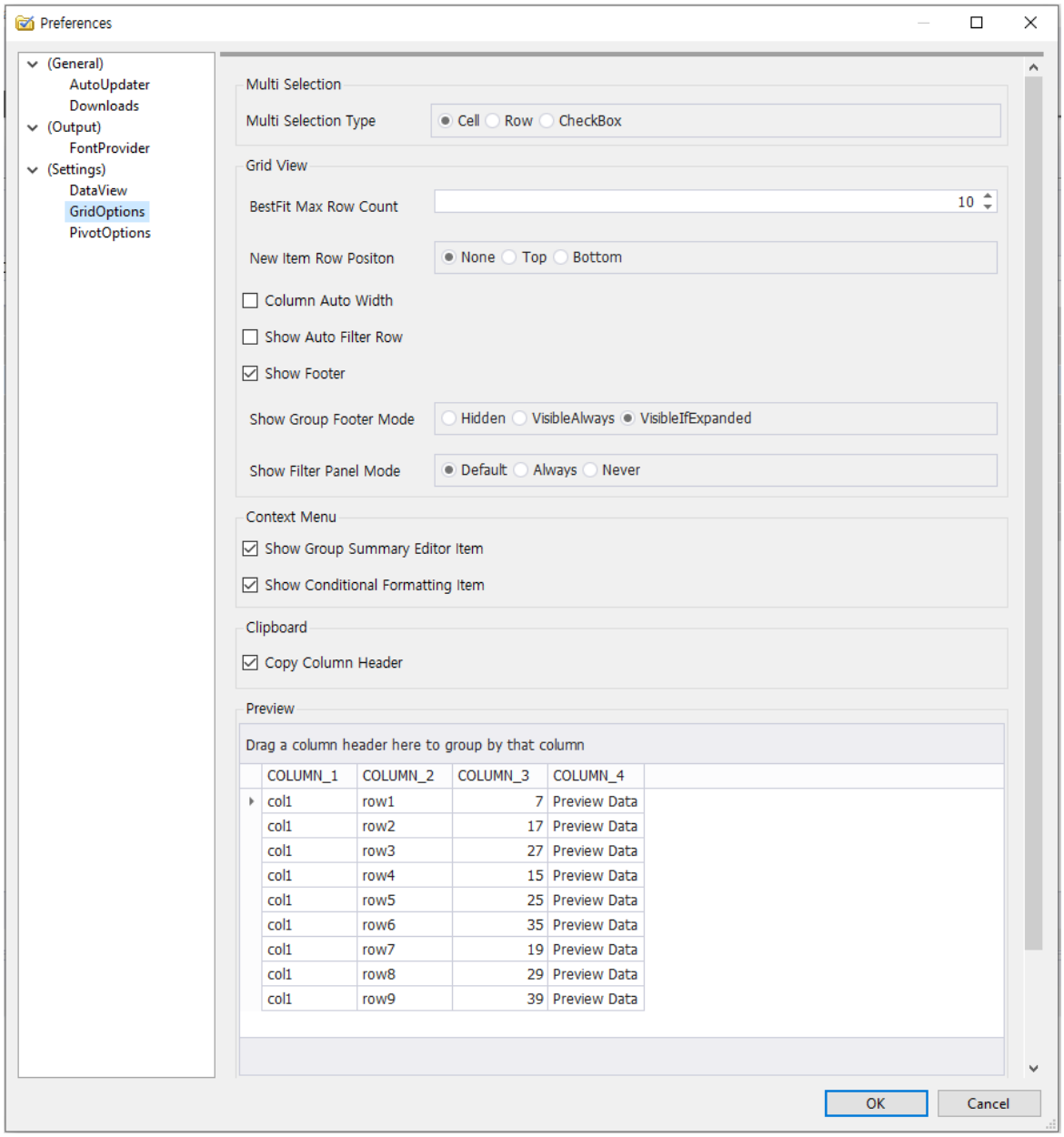
1. Studio를 실행하고 **[Tool > Options]** 메뉴로 실행합니다.
2. **[(Output)] > [FontProvider]** 메뉴에서 원하는 폰트, 스타일, 사이즈를 선택하여 **[OK]**를 통해 지정합니다.





GridOptions

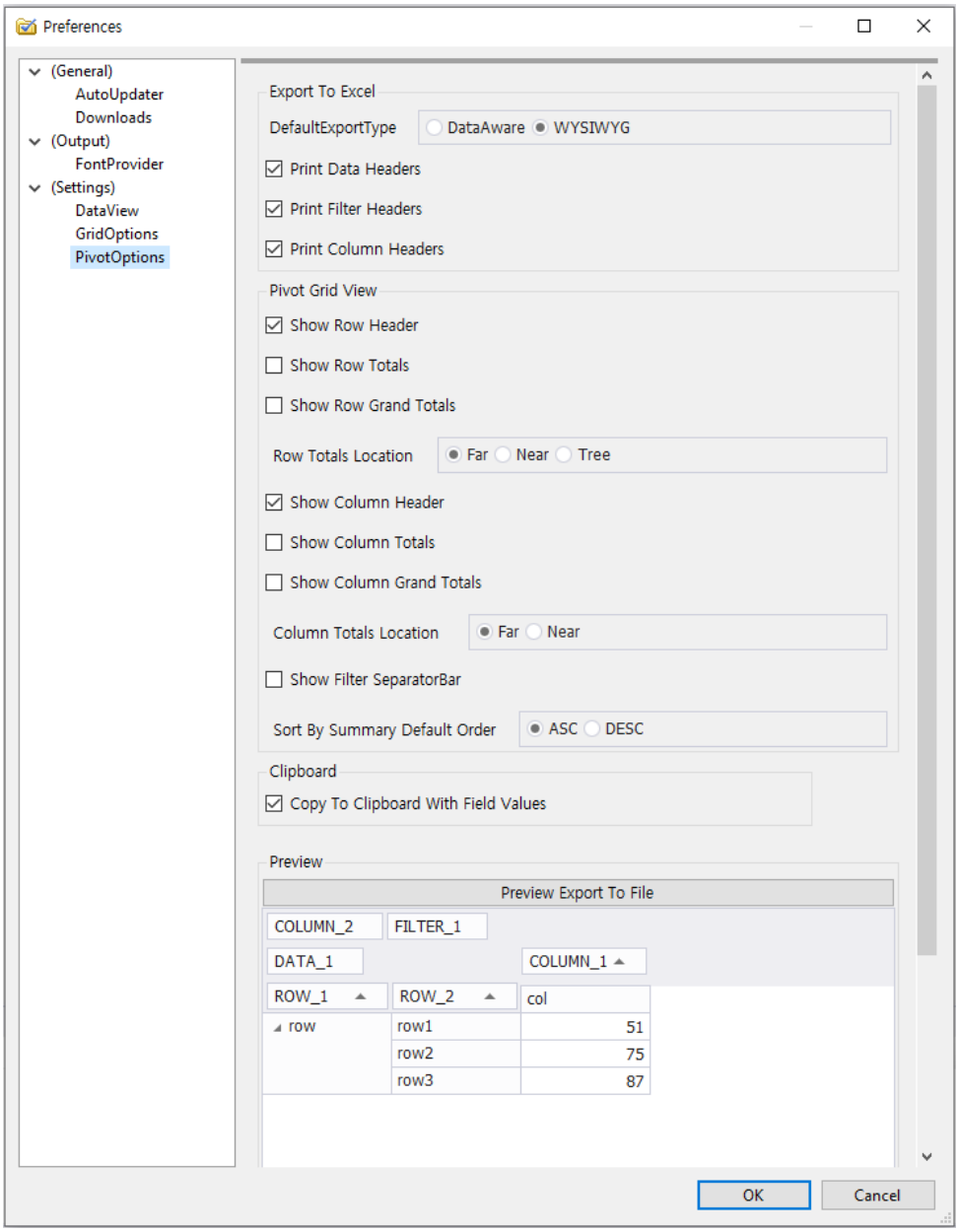
Studio에서는 제공하는 Data View에 대한 속성을 정의할 수 있습니다. 각 속성에 대한 설명은 [Grid View Preference 매뉴얼]에 상세 내용이 나와 있습니다. 각 속성의 값이 체크 또는 선택됨에 따라 변경되는 형태를 Preview를 통해 확인할 수 있습니다.



PivotOptions

Grid Option 과 마찬가지로 Pivot Grid View에 대한 속성을 정의하는 메뉴입니다. 각 속성의 기능은 **Pivot Grid Preference 매뉴얼**을 참고 할 수 있습니다.

각 속성의 값이 체크 또는 선택됨에 따라 변경되는 형태를 Preview를 통해 확인할 수 있습니다.

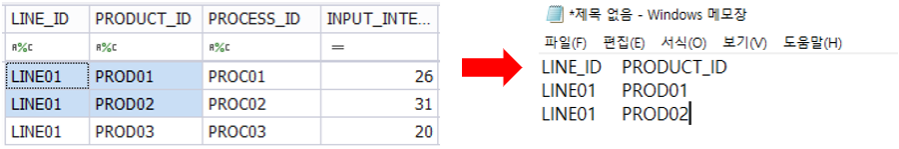


Grid View 사용법

[Experiment Tab View] 에서 Input, Output 을 조회하는 경우에는 Input, Output 데이터가 기본 그리드의 형태로 표시됩니다. 그리드의 경우 기본적인 검색 기능과 소팅 및 데이터 복사, Filter, Grouping 기능을 제공합니다. 아래는 그리드를 사용하여 데이터를 조회할 때 사용할 수 있는 기능에 대한 설명입니다.

복사 기능

그리드에서 데이터를 드래그하여 영역 지정 후 복사(Ctrl + C)하면 텍스트 형식으로 데이터 붙여넣기(Ctrl + V)가 가능합니다. 복사한 데이터의 기본 형식은 컬럼명을 포함한 데이터로 구성되며 Row는 줄바꿈(Enter), 컬럼은 탭(Tab)으로 구분되어 있습니다.



Copy Tool

Copy Tool은 데이터를 복사할 때 구분자 설정, 정렬, 중복 제거, 행/열 변경 등의 추가 **복사 설정이 가능한 보조 툴** 입니다. 해당 기능은 MOZART 2021.119.2 버전에서 지원하며 STUDIO의 상단 메뉴 Tool > Copy Tool을 실행하면 Copy Tool 팝업창이 나타납니다. 자세한 사용 방법은 데이터 Copy Tool 사용법을 참고하시기 바랍니다.

필터 기능

그리드의 컬럼 하단의 Text 입력 상자에 문자열을 입력하는 경우 해당 컬럼값을 기준으로 자동으로 필터가 생성됩니다. 기본 필터의 경우 String 형태인 경우 Contain 조건이며, 그 외 형태의 경우 Equal 조건을 필터를 생성합니다. 조건을 특정하기 위해서는 컬럼에 마우스 Over 시 생성되는 필터 아이콘을 선택하면 선택한 컬럼의 값을 체크하여 필터를 등록할 수 있습니다. 또한 그리드의 값을 컨텍스트 메뉴의 **[Add Filter]** 메뉴를 이용하여 지정한 값을 바로 필터에 등록할 수 있습니다.

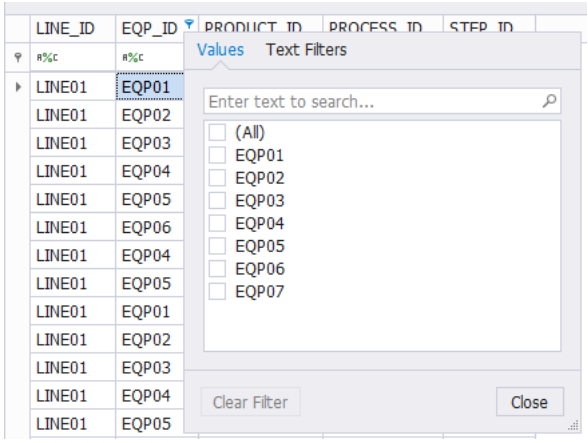


그림 1-1. Filter 를 입력하는 방법

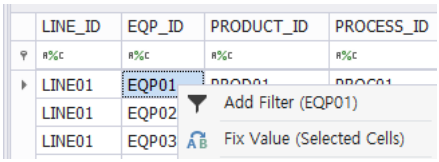


그림 1-2. Add Filter 메뉴를 이용한 Filter 입력

여러 컬럼에 동일한 방식으로 필터를 추가할 수 있으며, 그리드의 Multi Selection Type이 Cell 인 경우에는 여러 컬럼의 값을 다중 선택하여 필터에 등록할 수 있습니다. Control 버튼과 마우스를 클릭하고 컨텍스트 메뉴의 **[Add Filter] (Multi Select)** 를 통해 등록할 수 있습니다.

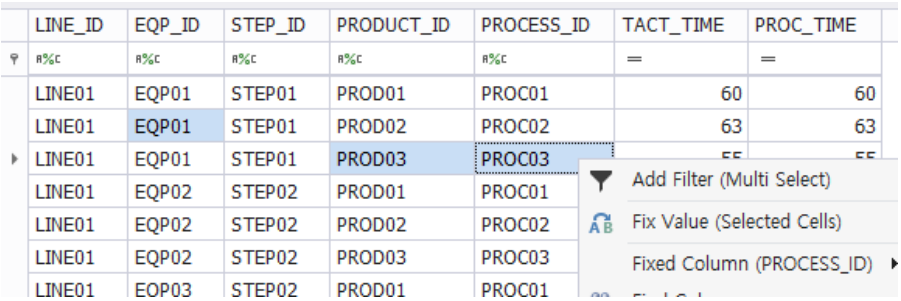


그림1-3. 여러 컬럼에 필터를 입력하는 방법

또한 직접 그리드 우하단의 **[Edit Filter]** 메뉴를 사용하여 필터를 지정할 수 있습니다.

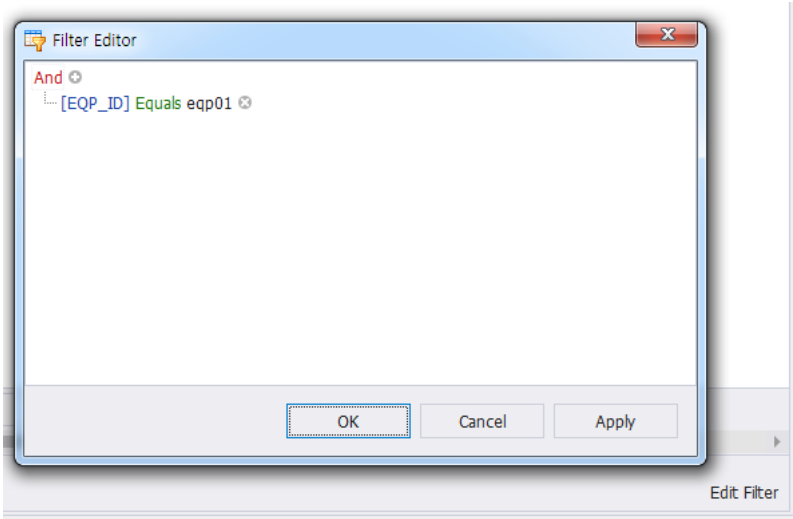


그림 2. Filter Editor

개별 View 에서 사용한 Filter 조건은 향후 적용을 위해 저장하고 로드하여 사용할 수 있습니다. 내용은 **Data Item View 사용법**의 **Manage View** 기능을 참조합니다.

소팅 기능

컬럼을 클릭하면 해당 컬럼으로 오름차순/내림차순이 변경되며 소팅을 할 수 있습니다. 복수 컬럼으로 소팅을 할 경우에는 소팅 순서에 따라 컬럼명에 오른쪽 마우스로 **[Sort Ascending][Sort Descending]** 메뉴를 순차적으로 적용합니다. 예를 들어 A 컬럼 > B 컬럼 순으로 소팅을 할 경우 A 컬럼에서 소팅을 실행한 후 B 컬럼을 선택하고 Pop up 메뉴를 사용하여 소팅을 하면 됩니다. 그렇지 않고 컬럼을 클릭하여 소팅을 할 경우에는 개별 컬럼 단위로만 소팅이 실행됩니다. 소팅 설정을 초기화 할때는 Popup 메뉴에서 **[Clear All Sorting] [Clear Sorting]** 기능을 사용합니다.

Grouping 및 Summary 기능

그리드의 특정 컬럼 값으로 Grouping 이 필요한 경우에는 해당 컬럼을 선택하여 그리드 상단의 “Drag a column header here to group by that column” 지시부분에 Drag&Drop 합니다. 혹은 그룹핑 대상 컬럼을 선택하고 Popup 메뉴에서 **[Group by this column]** 기능을 실행합니다. 이 경우 아래 그림과 같이 해당 컬럼으로 데이터를 그룹핑하여 볼 수 있습니다.

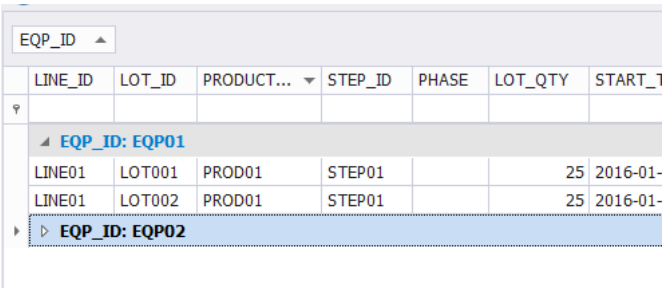


그림 3. 데이터 그룹핑 예시

데이터 그룹핑을 하면 전체 또는 그룹별로 데이터를 확장하거나 닫을 수 있습니다. 그룹핑된 전체 데이터에 대한 확인은 상단 그룹핑 필드의 Popup 메뉴에 있는 **[Full Expand]**, **[Full Collapse]** 기능을 사용하고, 그룹에 대한 데이터 확인은 선택한 그룹 필드의 Popup 메뉴에 있는 **[Expand Row]**, **[Collapse Row]** 기능을 사용하여 데이터를 볼 수 있습니다.

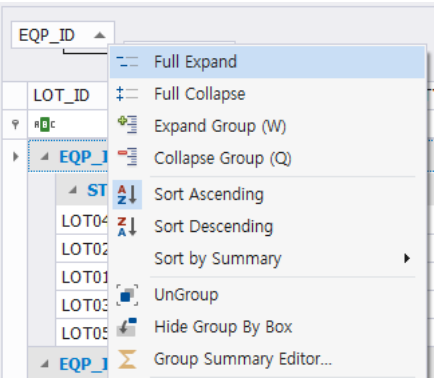


그림 3-1. 전체 데이터 그룹에 대한 Full Expand, Full Collapse

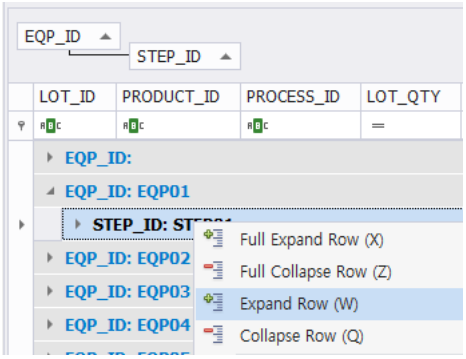


그림 3-2. 선택한 그룹에 대한 Expand, Collapse

데이터 그룹핑이 3개 이상의 그룹으로 사용할 경우 특정 그룹을 기준으로 모두 확장하거나 닫을 수도 있습니다. 선택한 그룹 필드의 Popup 메뉴에서 **[Expand Group]**, **[Collapse Group]** 기능을 이용하여 사용할 수 있습니다.

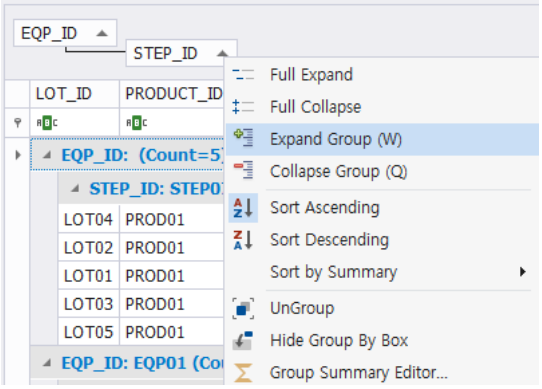


그림 3-3. 한 그룹에 대한 Expand Rows, Collapse Rows

Group Summary

그룹핑 된 데이터를 기준으로 Grouping Row에 표시 할 수 있습니다. 2가지 방법으로 표시할 수 있으며, 첫 번째 방법은 상단의 그룹핑 대상 필드에 오른쪽 마우스로 Popup 메뉴를 표시한 후 **[Group Summary Editor]** 를 사용하여 그룹핑된 행에 표시할 Summary 정보를 설정할 수 있습니다.

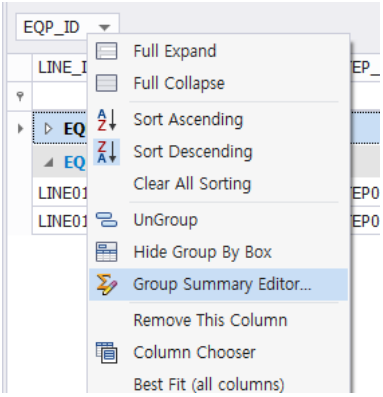


그림 4-1. Group Summary Editor Button

Group Summary Editor 에서는 아래와 같이 Summary 컬럼을 지정하고 Summary Type 을 선택하여 설정할 수 있습니다. Order 에서는 선택한 Summary 를 표시할 순서를 설정할 수 있습니다.

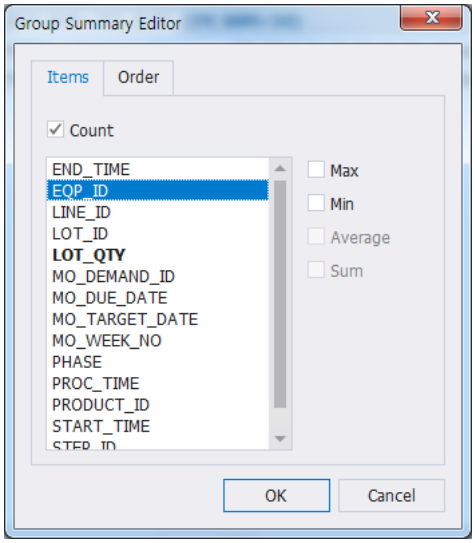


그림 5-1. Group Summary Editor의 Items Tab

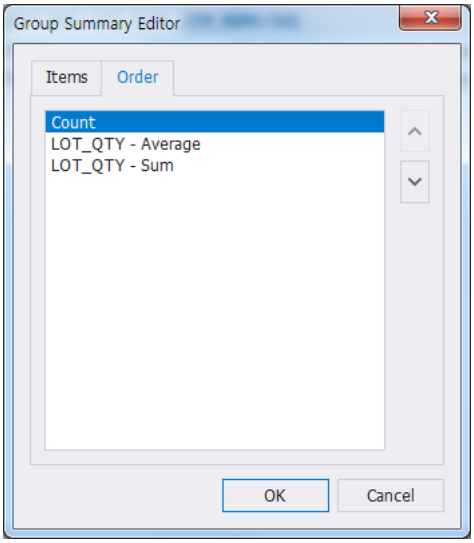
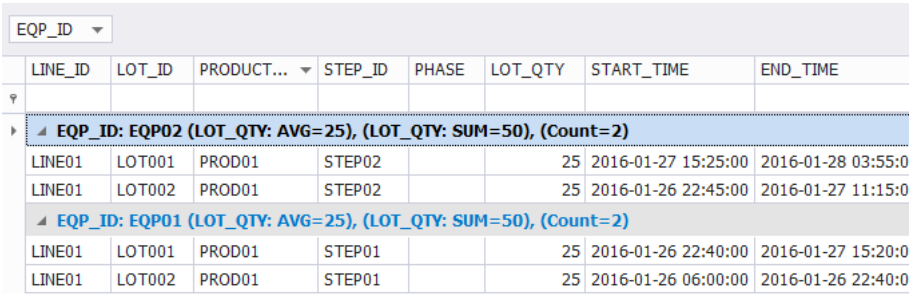


그림 5-2. Group Summary Editor의 Order Tab



EQP_ID

LINE_ID	LOT_ID	PRODUCT...	STEP_ID	PHASE	LOT_QTY	START_TIME	END_TIME
EQP_ID: EQP02 (LOT_QTY: AVG=25), (LOT_QTY: SUM=50), (Count=2)							
LINE01	LOT001	PROD01	STEP02		25	2016-01-27 15:25:00	2016-01-28 03:55:0
LINE01	LOT002	PROD01	STEP02		25	2016-01-26 22:45:00	2016-01-27 11:15:0
EQP_ID: EQP01 (LOT_QTY: AVG=25), (LOT_QTY: SUM=50), (Count=2)							
LINE01	LOT001	PROD01	STEP01		25	2016-01-26 22:40:00	2016-01-27 15:20:0
LINE01	LOT002	PROD01	STEP01		25	2016-01-26 06:00:00	2016-01-26 22:40:0

그림 5-3. Group Summary 결과

Group By This Column Template

두 번째는 상단의 그룹핑 대상 필드에 오른쪽 마우스로 Popup 메뉴를 표시한 후 **[Group By This Column Template]** 메뉴를 이용하여 만들 수 있습니다. 선택한 컬럼으로 그룹핑이 되며, 그룹핑시 해당 값에 대하여 5가지 타입으로 Summary 정보를 만들 수 있습니다. 단, Summary Type이 숫자가 아닌 경우에는 Count 외에는 표시가 되지 않습니다.

- Sum : 합계 정보
- Average : 평균 정보
- Count : 전체 갯수 정보
- Max : 최대값 정보
- Min : 최소값 정보

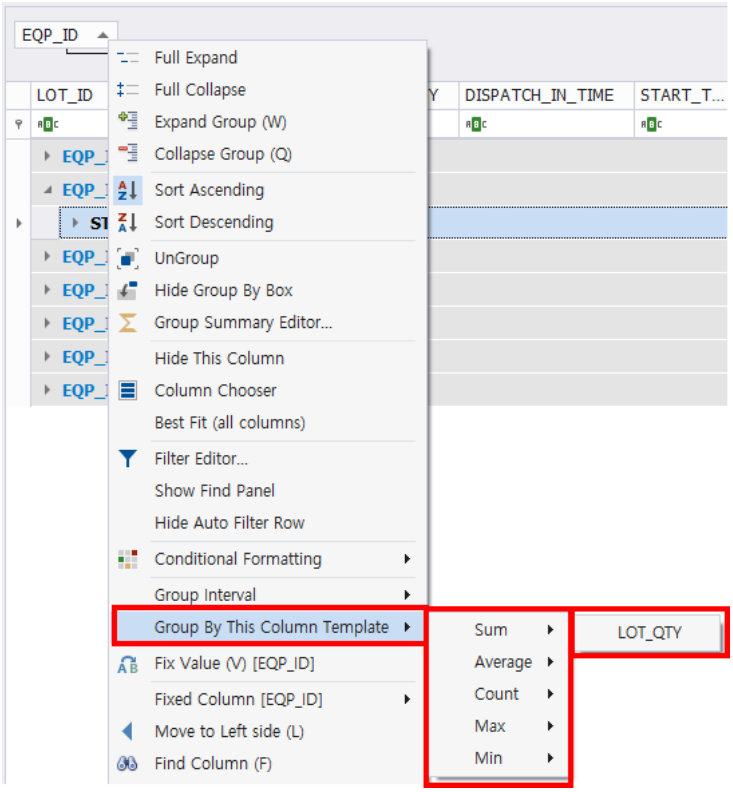


그림 6-1. Group By this Column Template 메뉴 화면

만약 **[Group By This Column Template]** 메뉴를 이용하여 LOT_QTY 컬럼의 값을 합한 값을 그룹핑 할 경우에는 **[그림 6-2]**과 같이 Footer에 표시가 됩니다. Footer에 표시할 때는 Preference 의 Show Group Footer Mode 에서 표시 방법을 설정할 수 있습니다.

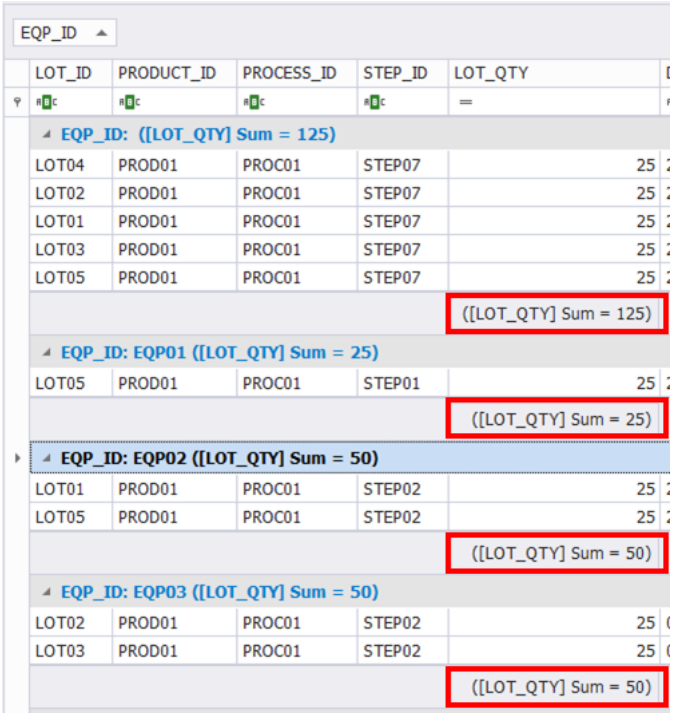


그림 6-2. Group By This Column Template를 이용하여 Sum > LOT_QTY 값이 Grouping Summary 및 Footer 에 표시된 화면

추가된 Footer를 제거하는 방법은 Footer 의 Context Menu 에서 None 을 선택하면 표시가 제거 됩니다.

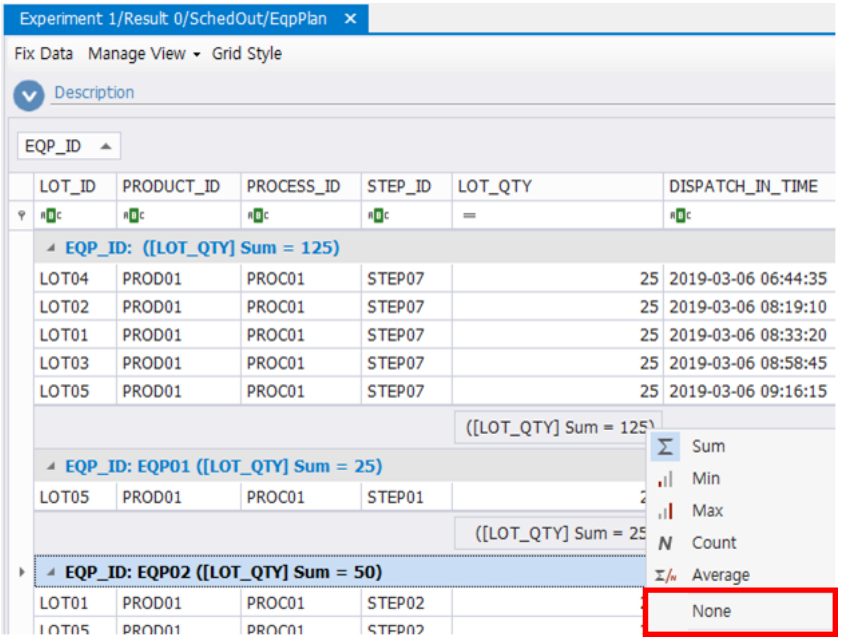


그림 6-3. 추가된 Footer를 제거하는 화면

Grouping을 해제하기 위해서는 그룹을 드래그하여 해제하거나, 그룹의 Context Menu 에서 **[UnGroup]** 을 이용하여 해제할 수 있습니다.

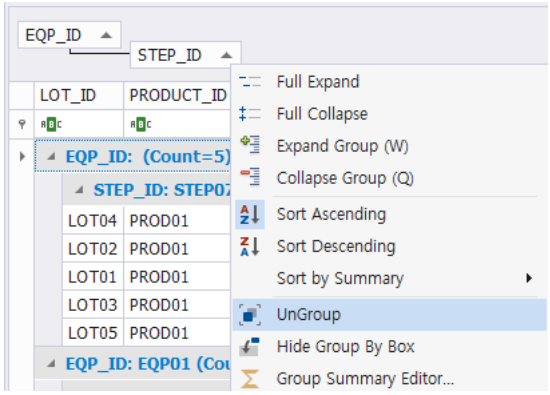


그림 7. 그룹 해제를 위한 UnGroup 메뉴 화면

Group Interval

DateTime 타입의 컬럼은 기본적으로 일 단위(Day) 기준으로 그룹핑이 됩니다. Group Interval을 통해 일 단위가 아닌 다른 기준으로 그룹핑을 변경할 수 있습니다.

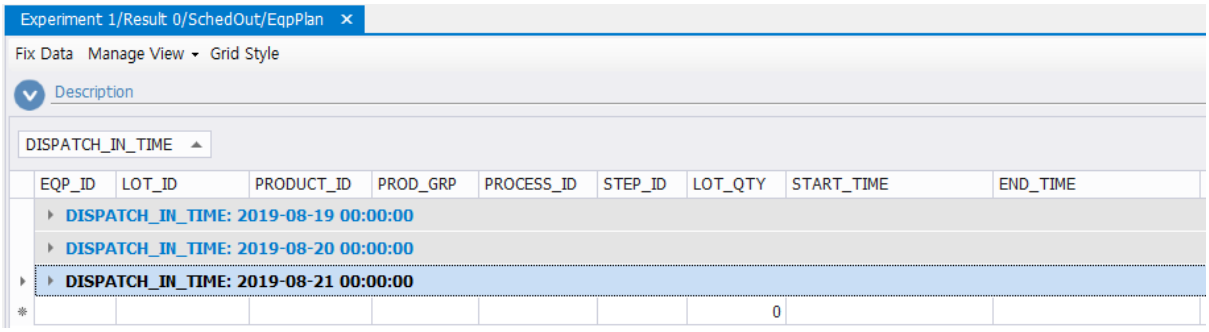


그림 8. DateTime 타입의 컬럼값을 그룹핑한 결과

기본값은 Default 이며, 그룹핑된 컬럼의 Context Menu를 통해 그룹핑 간격을 변경할 수 있습니다.

- Default : 기본값 (Day)
- Day : 일 기준 그룹핑
- Month : 월 기준 그룹핑
- Year : 년 기준 그룹핑
- DisplayText : 실제 날짜 값을 기준으로 그룹핑

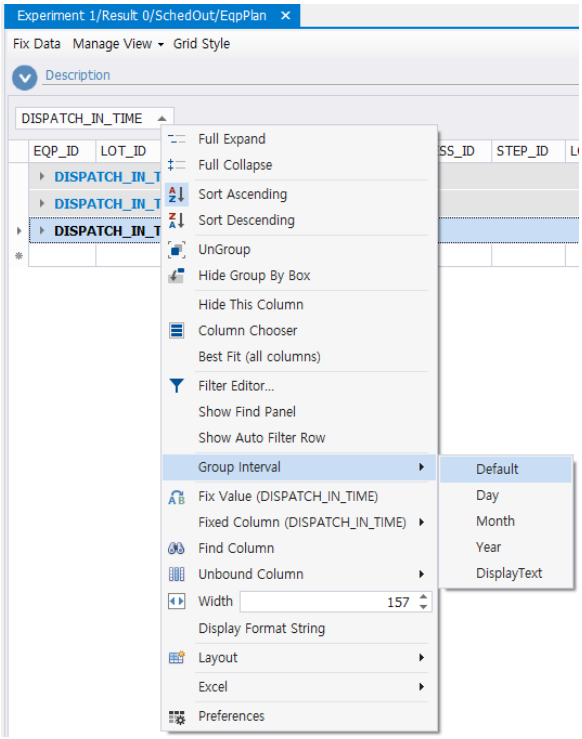


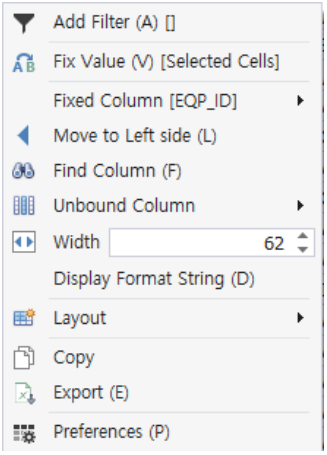
그림 9. Group Interval 메뉴

Grid View 보조 기능

MOZART 2020.118.0 버전부터 Grid View 보조 기능이 추가 되었습니다. 보조 기능은 Grid View에서 **Column Header 보조 메뉴**와 **Cell 보조 메뉴**로 나뉘어집니다.

단축키

주요 기능에 대하여 단축키 사용이 가능합니다. 각 메뉴에서 **( )** 괄호가 있는 메뉴는 단축키 사용이 가능한 메뉴입니다.



1. Expand Rows - **W** : Grouping 으로 사용할 때 확장할 때 사용
2. Collapse Rows - **Q** : **Grouping** 으로 사용 할 때 축소할 때 사용
3. Add Filter - **A** : **필터에 값을 추가**할 때 사용
4. Fix Value - **V** : **데이터 값을 수정**할 때 사용
5. Find Column - **F** : 컬럼을 찾을 때 사용
6. Display Format String- **D** : 사용자가 String Format Type을 지정할 때 사용
7. Preferences- **P** : **Grid 별 옵션을 지정할 때** 사용
8. Excel (View에 따라 Sub Menu가 생성되지 않는 경우도 있어서 추가)
    a. Import - **I** : Excel 에서 데이터를 가져올 때 사용
    b. Export - **E** : Excel 데이터로 추출할 때 사용

Column Header 보조 메뉴

MOZART Studio의 Main View에서 제공되는 기능들로써, 아래 [그림 10.]과 같이 Column Header의 우측 클릭을 통해 해당 기능을 확인 및 작동시킬 수 있습니다.

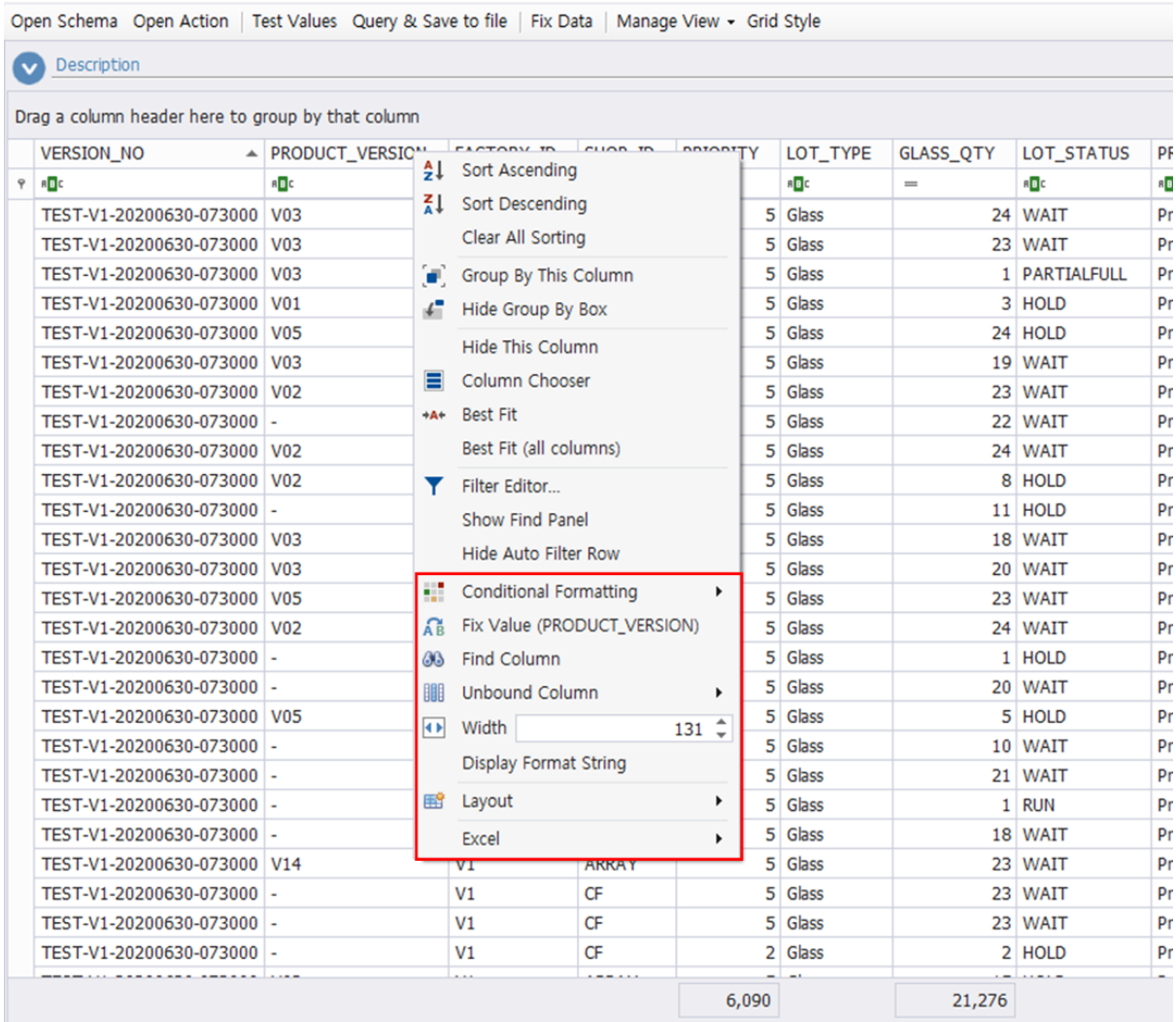


그림 10. Column Header 클릭시 나타나는 기능들

아래는 MOZART Studio에서 사용할 수 있는 기능 메뉴들을 나타냅니다.

Level 1	Level 2	Level 3	Description
Conditional Formatting	Highlight Cell Rules	Greater Than	사용자 입력 값 > Column 이면, Column들의 Format을 변경하 기능을 제공함
		Less Than	사용자 입력 값 < Column 이면, Column들의 Format을 변경하는 기능을 제공함
		Between	사용자 입력 값 < Column < 사용자 입력 값이면, Column들의 Format을 변경하는 기능을 제공함
		Equal To	사용자 입력 값 = Column 이면, Column들의 Format을 변경하는 기능을 제공함
		Text that Contains	사용자가 입력한 값이 Column의 값에 포함되면, Column들의 Format을 변경하는 기능을 제공함
		Custom Condition	사용자가 지정한 값이 Column에서 사용자가 지정한 조건에 따라 Format을 변경하는 기능을 제공함
	Unique /Duplicate Rules	Unique Values	Column에서 중복되지 않는 값들에 대한 Format을 변경하는 기능을 제공함
		Duplicate Values	Column에서 중복되는 값들에 대한 Format을 변경하는 기능을 제공함
	Manage Rules (string)		사용자에 의해 1개 이상의 Column의 Format을 변경하는 기능을 제공함

Level 1	Level 2	Level 3	Description
	Top/Bottom Rules (int)	Top 10 items	Column에서 가장 큰 값부터 사용자가 지정한 순번까지 Format을 변경하는 기능을 제공함
		Top 10 %	Column에서 가장 큰 값부터 사용자가 지정한 %까지 Format을 변경하는 기능을 제공함
		Bottom 10 items	Column에서 가장 작은 값부터 사용자가 지정한 순번까지 Format을 변경하는 기능을 제공함
		Bottom 10 %	Column에서 가장 작은 값부터 사용자가 지정한 %까지 Format을 변경하는 기능을 제공함
		Above Average	Column에서 평균 이상의 값 위로 Format을 변경하는 기능을 제공함
		Below Average	Column에서 평균 이하의 값 아래로 Format을 변경하는 기능을 제공함
	Data Update Rules		Column에서 데이터가 변경되었을 시 Format을 변경하는 기능을 제공
	Data Bars		Column의 데이터 값에 따라 Bar를 표시하는 기능을 제공함
	Color Scales		Column의 데이터 값에 따라 Color의 색상을 나타내는 기능을 제공함
	Icon Sets		Column의 데이터 값에 따라 Icon을 표시하는 기능을 제공함
	Clear Rules	Clear Rules from This Column	해당 Column의 Conditional Formatting기능을 제거함
		Clear Rules from All Column	모든 Column의 Conditional Formatting기능을 제거함
Data Join	Add Rule		Grid View 스키마에 하나의 Join을 추가하는 메뉴 입니다.
	Manage Rules		Grid View에서 추가한 Join 확인 및 수정하는 메뉴 입니다.
	Reset Rules		Grid View에서 추가한 Join Rules를 모두 삭제하는 초기화 메뉴 입니다.
Fix Value			선택된 Column에 속한 데이터를 한번에 수정할 수 있는 기능을 제공함
Fixed Column	Left	All Left By Column	선택된 Column을 포함한 왼쪽의 모든 컬럼을 그리드의 왼쪽에 고정하는 기능을 제공함
		Current Column Only	선택된 Column만 그리드의 왼쪽에 고정하는 기능을 제공함
	Right	All Right By Column	선택된 Column을 포함한 오른쪽의 모든 컬럼을 그리드의 오른쪽에 고정하는 기능을 제공함
		Current Column Only	선택된 Column만 그리드의 오른쪽에 고정하는 기능을 제공함
Find Column			많은 Column 중 사용자가 원하는 Column을 쉽게 찾을 수 있으며, 해당 Column을 맨 앞으로 위치하여 쉽게 사용할 수 있는 기능을 제공함
Unbound Column	Add Unbound Column		기존의 Column들을 조합하여 새로운 Column을 생성하기 위한 기능을 제공함
	Column Copy		기존의 Column을 복사하여 똑같은 Column을 생성하기 위한 기능을 제공함
Width			사용자가 Column 너비를 설정할 수 있는 기능을 제공함
Display Format String			사용자가 원하는 String Format Type을 지정할 수 있도록 Column의 Format값을 변경하는 기능을 제공함
Layout	Save Layout		Layout을 저장하는 기능을 제공함
	Load Layout		Default Layout을 불러오는 기능을 제공함
	Clear Layout		설정한 Layout이 없는 초기 UI로 돌아가는 기능을 제공함
	Manage Layout		저장된 Layout 정보를 확인하고 관리하는 기능을 제공함
Excel	Export		Mozart studio에서 출력된 데이터를 엑셀 파일로 내보내는 기능을 제공함

Level 1	Level 2	Level 3	Description
	Import		엑셀 파일로 저장된 데이터를 Mozart 데이터로 가져오는 기능을 제공함
Preference			Grid View 별 Option을 설정하는 기능을 제공

Conditional Formatting

본 기능은 사용자가 원하는 조건에 따라 Column의 Format을 변경할 수 있습니다. 해당 기능에서 세부 기능들로 나누어지며, 자세한 내용은 DevExpress > Conditional Formatting에서 확인 할 수 있습니다.

Highlight Cell Rules

사용자가 입력한 값에 따라 Column의 Cell Format을 변경하는 기능을 제공합니다.

Greater Than

Column에서 사용자가 입력한 값보다 크면 해당 값의 Format을 변경하는 기능을 제공합니다.

[그림 11.]와 같이 Greater Than 기능을 선택 후, Greater Than 창에서 Value값을 입력합니다. 이후, Value값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

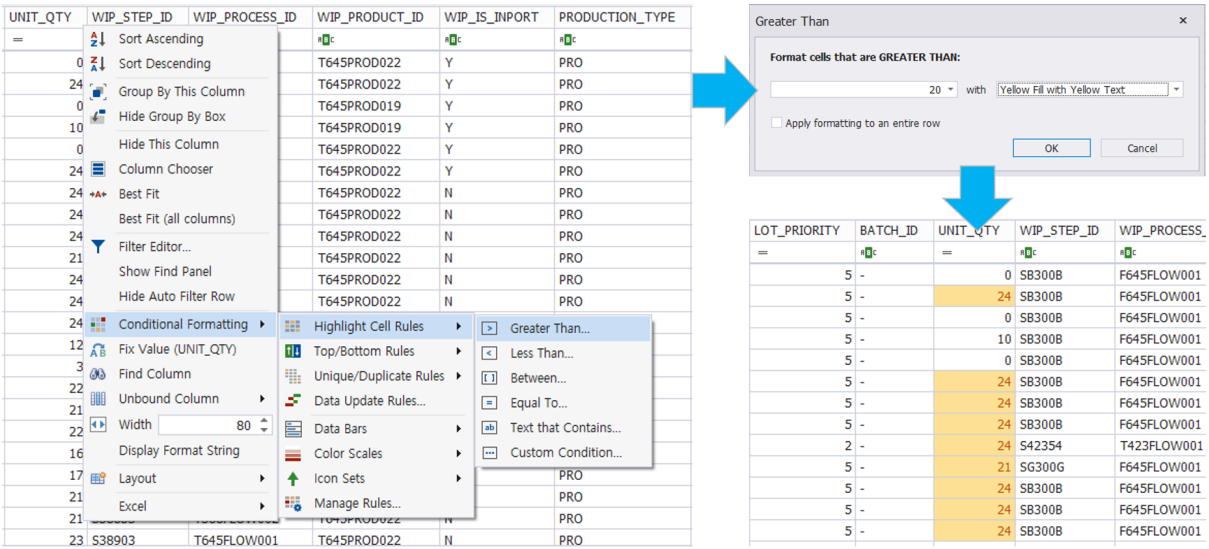


그림 11. Highlight Cell Rules 선택 후 Greater Than 기능 사용 예시

Less Than

Column에서 사용자가 입력한 값보다 작으면 해당 값의 Format을 변경하는 기능을 제공합니다.

[그림 12.]과 같이 Less Than 기능을 선택 후, Less Than 창에서 Value값을 입력합니다. 이후, Value값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

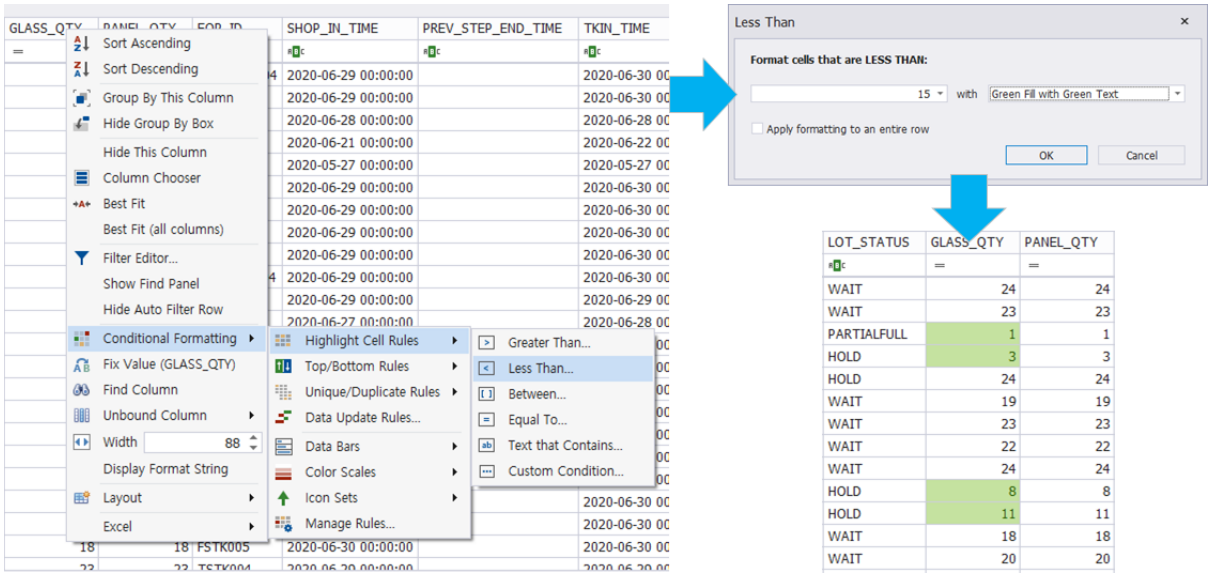


그림 12. Highlight Cell Rules 선택 후 Less Than 기능 사용 예시

Between

Column에서 사용자가 입력한 값 사이의 Format을 변경하는 기능을 제공합니다.

[그림 13.]과 같이 Between 기능을 선택 후, Between 창에서 2가지 value값을 선택합니다. 이후, 2가지 value 사이의 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

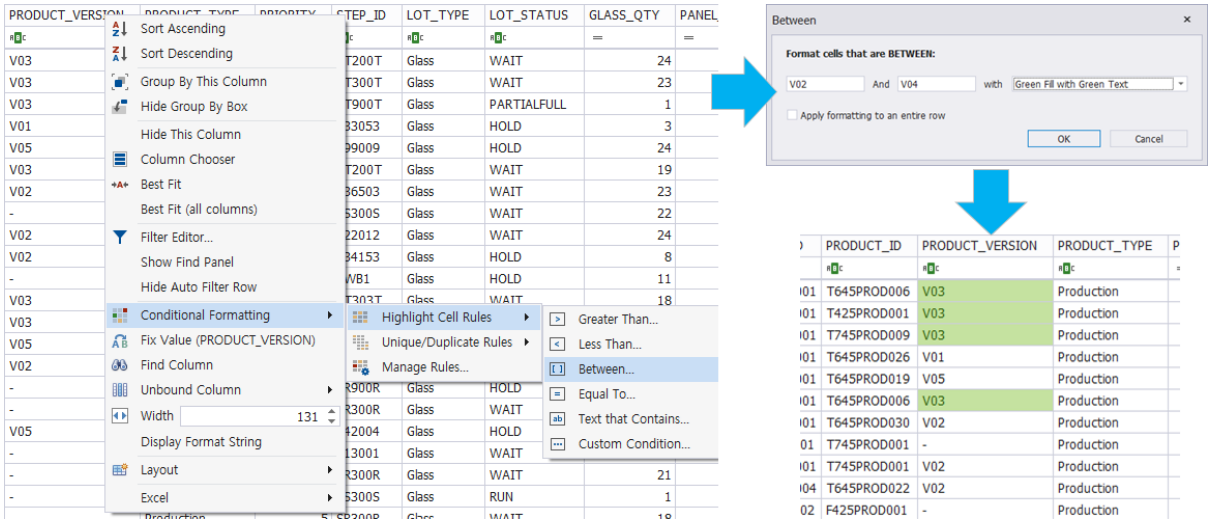


그림 13. Highlight Cell Rules 선택 후 Between 기능 사용 예시

Equal To

Column에서 사용자가 입력한 값과 똑같은 값이 존재하는 Cell의 Format을 변경하는 기능을 제공합니다.

[그림 14.] 같이 Equal To 기능을 선택 후, Equal To 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

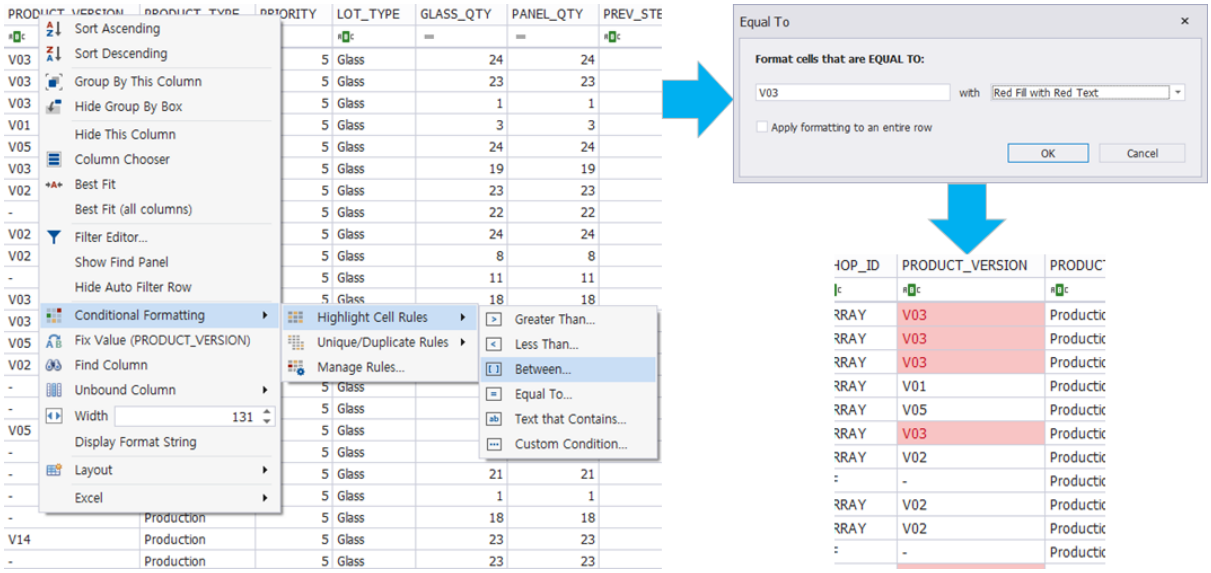


그림 14. Highlight Cell Rules 선택 후 Equal To 기능 사용 예시

Text that Contains

Column에서 사용자가 입력한 값이 포함된 Cell의 Format을 변경하는 기능을 제공합니다.

[그림 15.]과 같이 Text that Contains 기능을 선택 후, Text that Contains 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

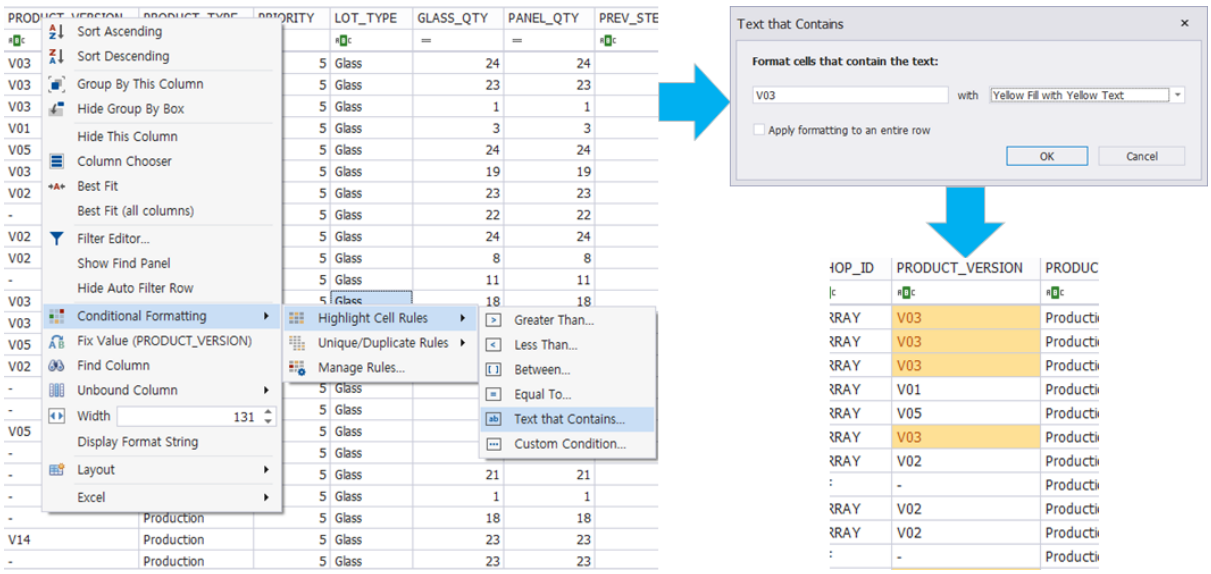


그림 15. Highlight Cell Rules 선택 후 Text that Contains 기능 사용 예시

Custom Condition

Column에서 사용자가 지정한 조건에 따라 Format을 변경하는 기능을 제공합니다.

[그림 16.] 같이 Custom Condition 기능을 선택 후, Custom Contains 창에서 해당된 Column에 대한 조건값을 선택합니다. 조건들은 values값에 대한 Operations (=, >, <, Equal, etc.)들을 나타내며, 조건을 선택 후 value(숫자 or 숫자+문자 or 문자) 입력합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

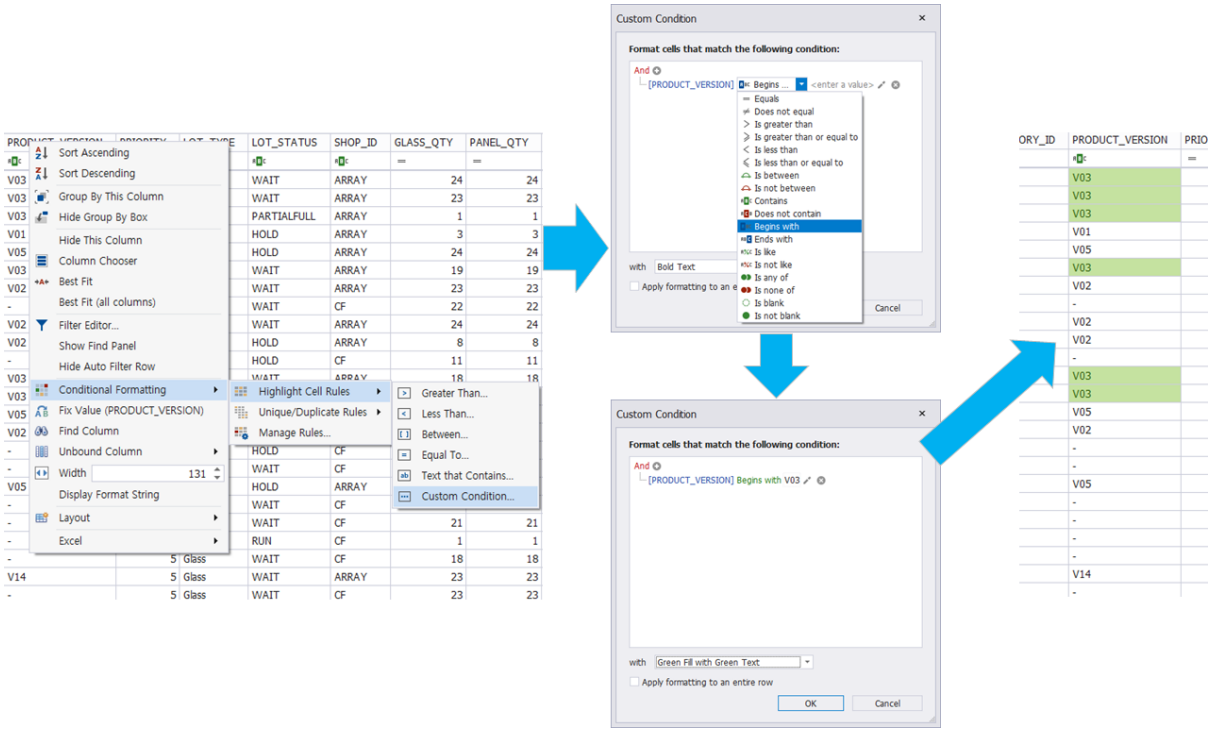


그림 16. Highlight Cell Rules 선택 후 Custom Condition 기능 사용 예시

Unique/Duplicate Rules

사용자가 입력한 값에 따라 Column의 Cell Format을 변경하는 기능을 제공합니다.

Unique Values

Column에서 중복되지 않는 값들에 대한 Format을 변경하는 기능을 제공합니다.

[그림 17.] 같이 Unique Values 기능을 선택 후, Unique Values 창에서 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

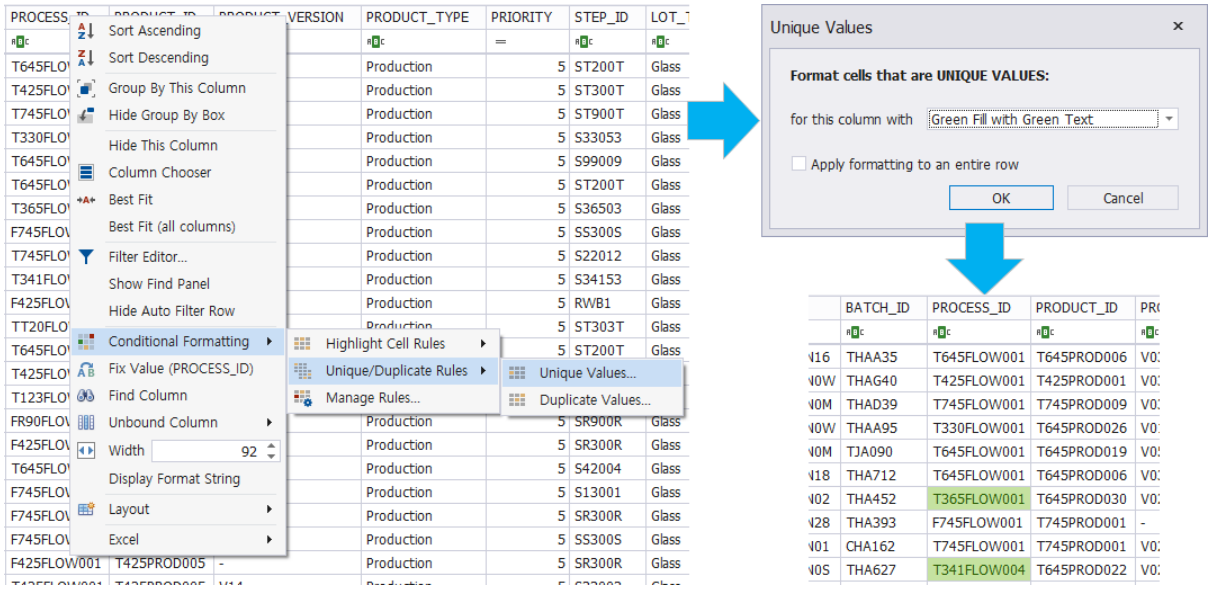


그림 17. Unique Values 기능 사용 예시

Duplicate Values

Column에서 중복되는 값들에 대한 Format을 변경하는 기능을 제공합니다.

[그림 18.]과 같이 Duplicate Values 기능을 선택 후, Duplicate Values 창에서 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

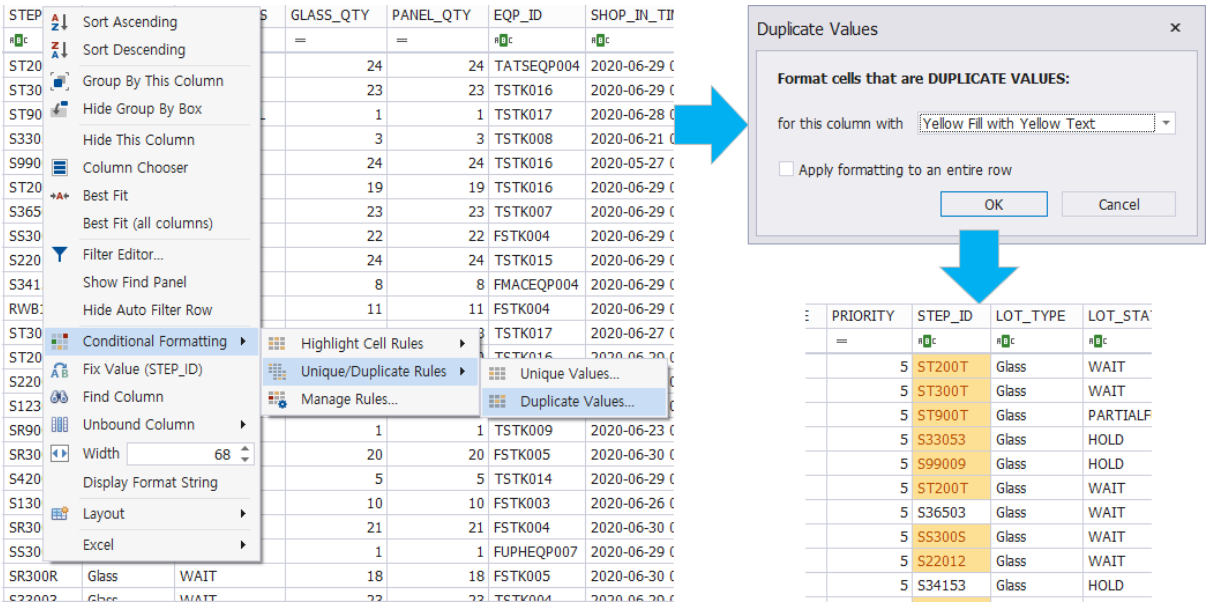


그림 18. Duplicate Values 기능 사용 예시

Manage Rules

사용자에 의해 string 값을 가진 1개 이상의 Column의 Format을 변경하는 기능을 제공합니다.

[그림 19.]과 같이 Manage Rules 기능을 선택 후 Conditional Formatting Rules Manager 창에서 New Rule을 선택하면, 해당 Column의 Cell에 대한 조건을 생성할 수 있습니다. Conditional Formatting Rules Manager창의 New Rule을 선택하면 Edit Formatting Rule 창이 나타나며, Select a Rule Type에서 원하는 조건을 선택합니다. 이후, Edit the Rule Description의 Format only cells with에서 Cell value과 조건들(Text 굵기, RED / GREEN / YELLOW 색 표현, 등) 그리고 value를 입력하게 됩니다. 아래의 Preview의 Format에서는 해당 cell의 색깔을 선택할 수 있습니다.

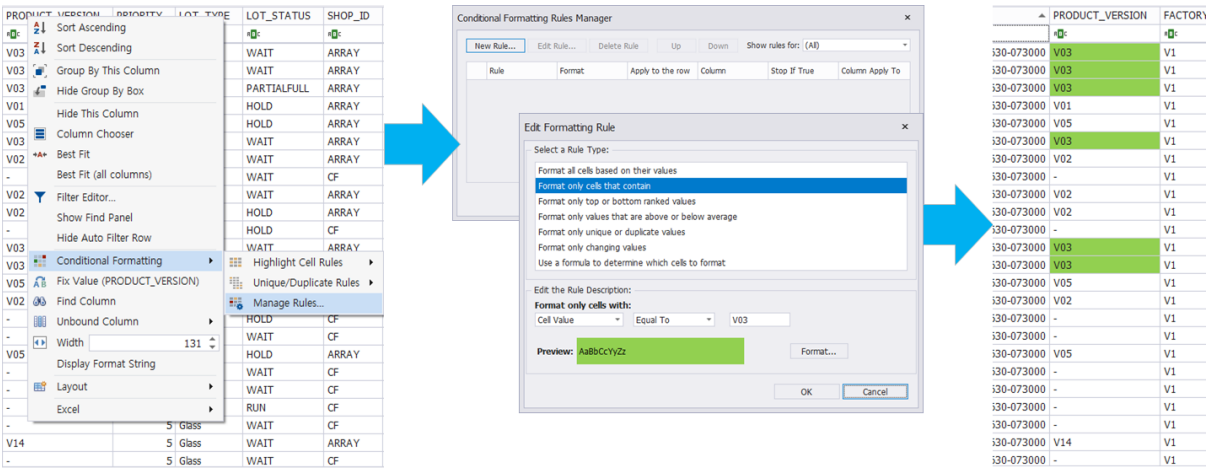


그림 19. Manage Rules 기능 사용 예시

Top/Bottom Rules (int)

사용자가 지정한 값에 따른 순번에 따라 Column의 Format을 변경하는 기능을 제공합니다.

Top 10 items

Column에서 가장 큰 값부터 사용자가 지정한 순번까지 Format을 변경하는 기능을 제공합니다.

[그림 20.]과 같이 Top 10 items 기능을 선택 후, Top 10 items 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

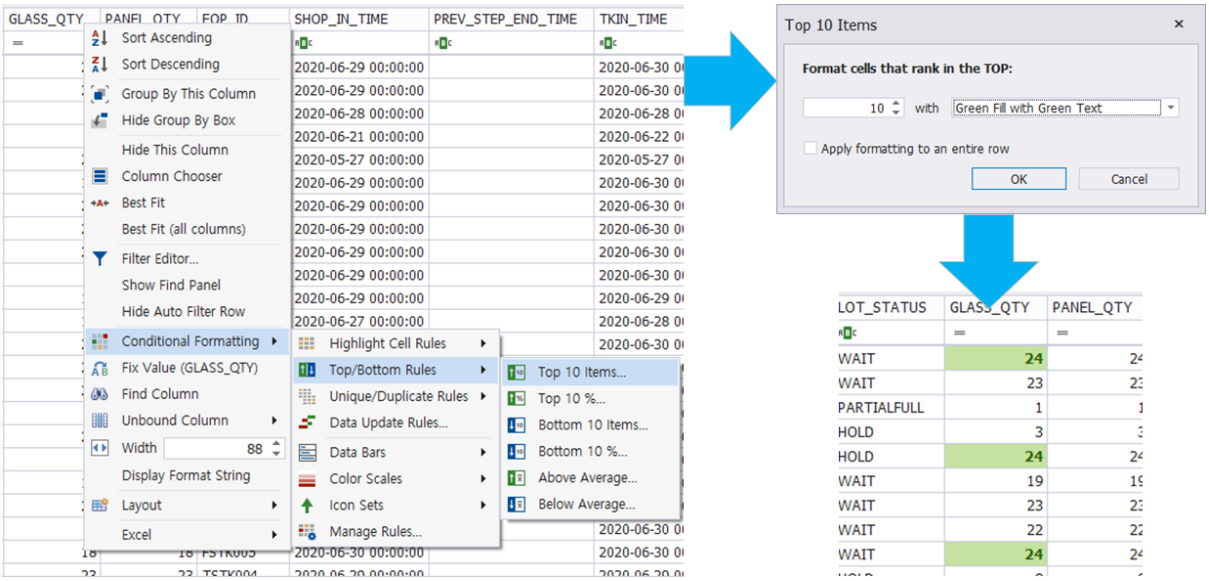


그림 20. Top 10 items 기능 사용 예시

중복 값이 존재하고 10 개의 최상위 값에 대해 기능 수행하는 경우에는 중복된 값도 상위 10개의 숫자에 포함되게 됩니다. 또한 중복값이 Top 10 기준보다 많은 경우에는 중복값 전체에 기능이 동작하게 됩니다. 예를 들어 {1,2,3,4,5,6,7,8,9,10,10,10,10,10,10,10,10,10,10,10} 값을 가지는 20개의 레코드가 존재할 때 상위 Top 10을 실행하면 10의 값을 가지는 11개의 레코드에 기능이 적용됩니다.

Top 10 %

Column에서 가장 큰 값부터 사용자가 지정한 %까지 Format을 변경하는 기능을 제공합니다.

[그림 21.] 같이 Top 10 % 기능을 선택 후, Top 10 % 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

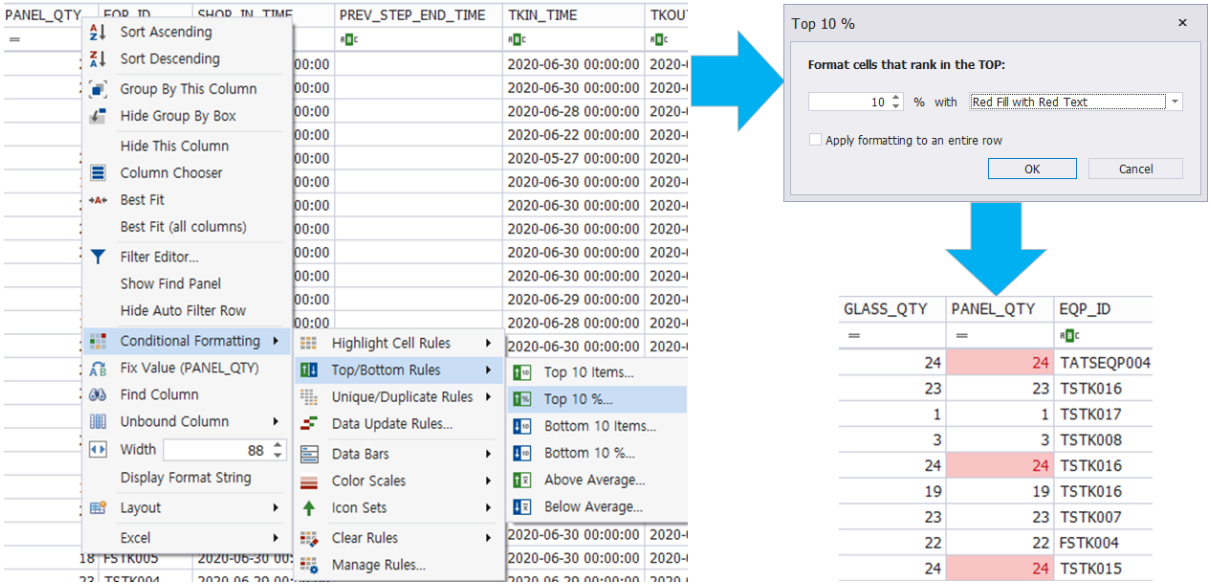


그림 21. Top 10 % 기능 사용 예시

Bottom 10 items

Column에서 가장 작은 값부터 사용자가 지정한 순번까지 Format을 변경하는 기능을 제공합니다.

[그림 22.]과 같이 Bottom 10 items 기능을 선택 후, Bottom 10 items 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

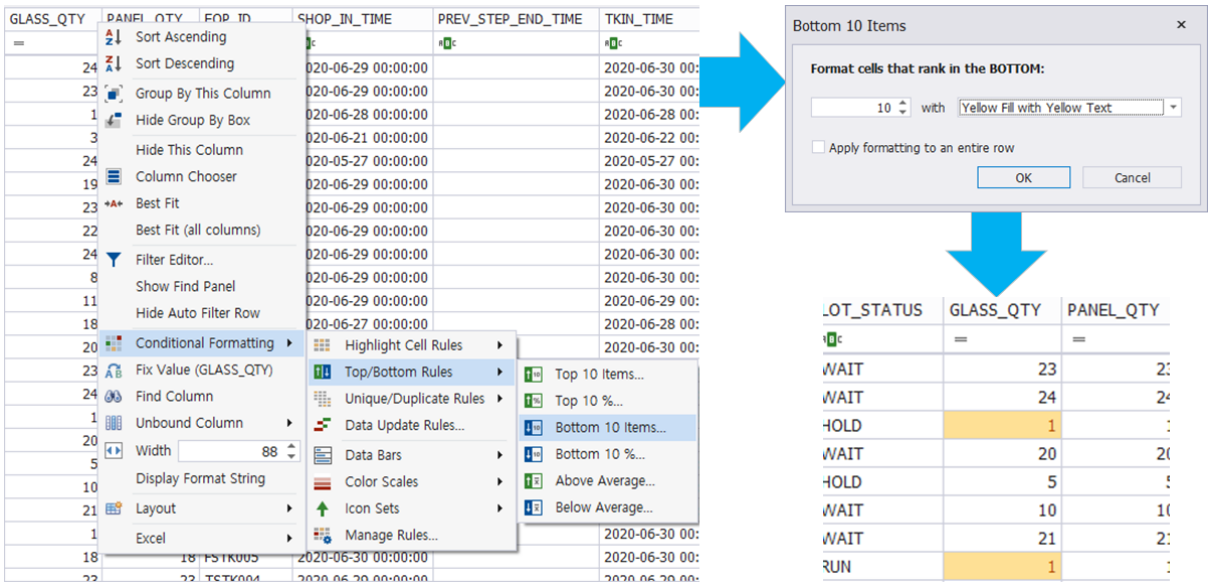


그림 22. Bottom 10 items 기능 사용 예시

Bottom 10도 Top 10과 동일하게 중복 값이 존재하고 10 개의 최하위 값에 대해 기능 수행하는 경우에는 중복된 값도 하위 10개의 숫자에 포함되게 됩니다. 또한 중복값이 Bottom 10 기준보다 많은 경우에는 중복값 전체에 기능이 동작하게 됩니다. 예를 들어 {1,1,1,1,1,1,1,1,1,1,3,3,3,10,10,10,10,10,10} 값을 가지는 20개의 레코드가 존재할 때 상위 Bottom 10을 실행하면 1의 값을 가지는 11개의 레코드에 기능이 적용됩니다.

Bottom 10 %

Column에서 가장 작은 값부터 사용자가 지정한 %까지 Format을 변경하는 기능을 제공합니다.

[그림 23.]과 같이 Bottom 10 % 기능을 선택 후, Bottom 10 % 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

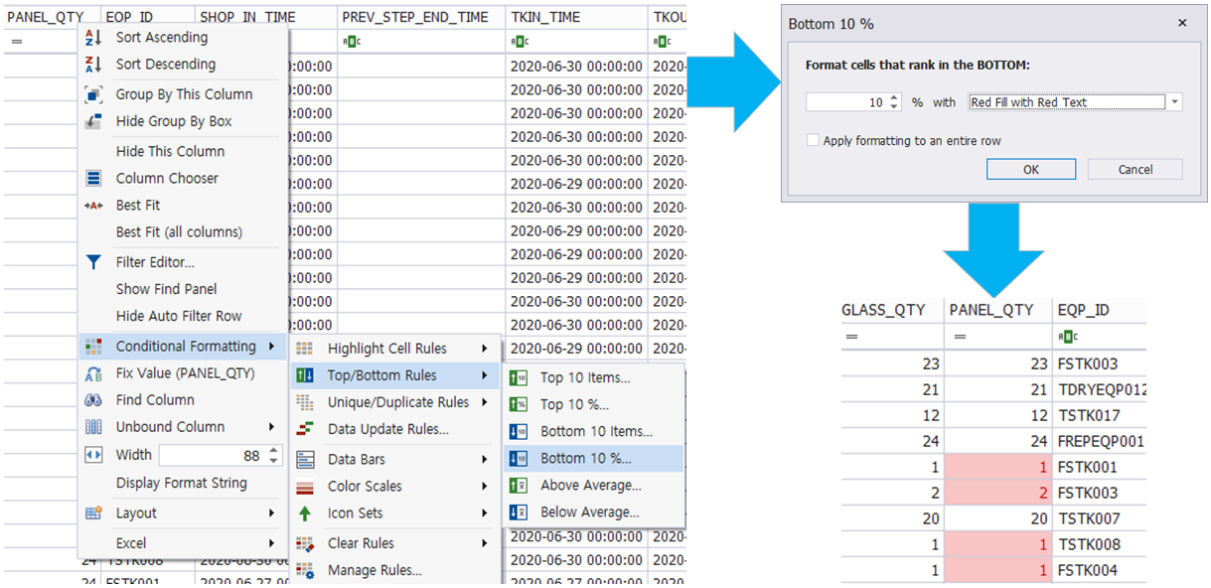


그림 23. Bottom 10 % 기능 사용 예시

Above Average

Column에서 평균 이상의 값 위로 Format을 변경하는 기능을 제공합니다.

[그림 24.] 같이 Above Average 기능을 선택 후, Above Average 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

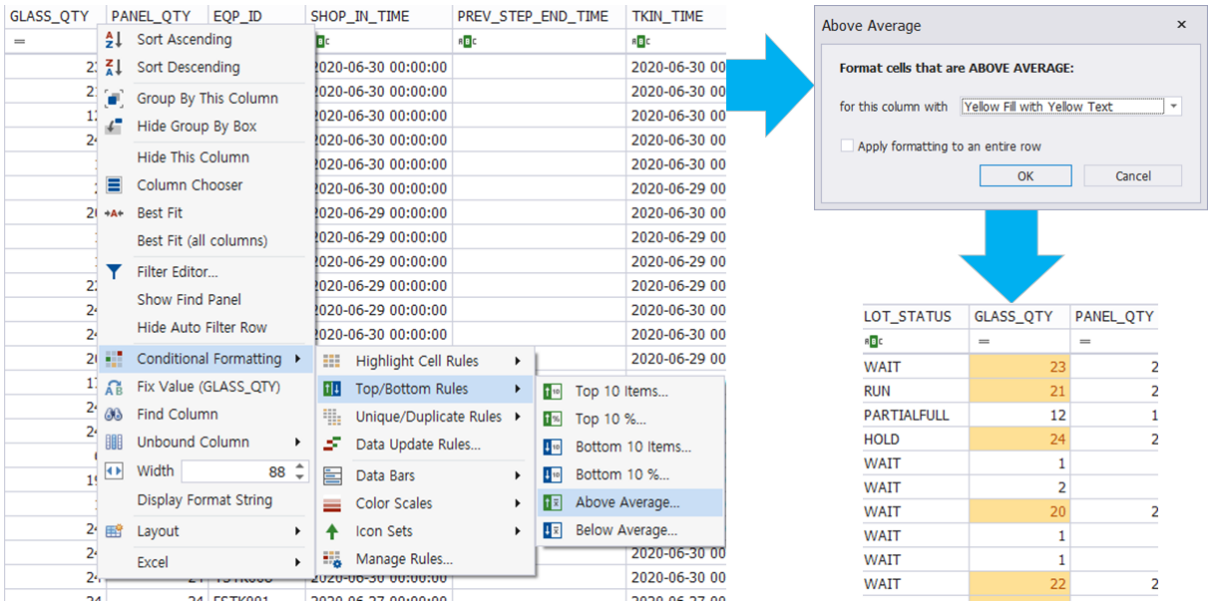


그림 24. Above Average 기능 사용 예시

Below Average

Column에서 평균 이하의 값 아래로 Format을 변경하는 기능을 제공합니다.

[그림 25.]과 같이 Below Average 기능을 선택 후, Below Average 창에서 value값을 선택합니다. 이후, value 값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

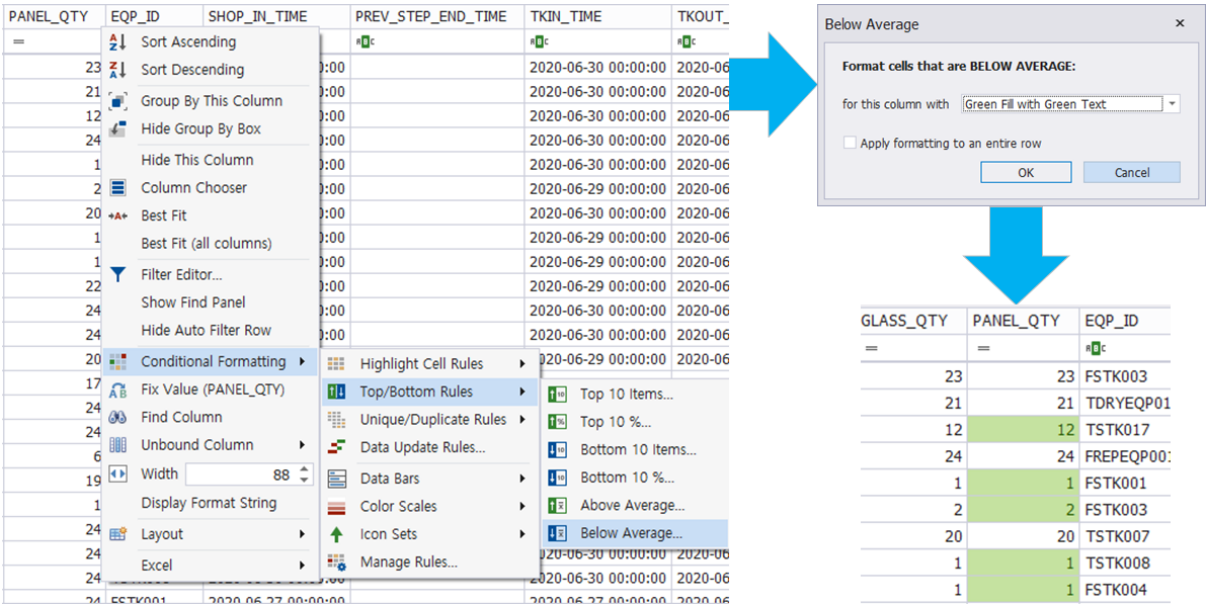


그림 25. Below Average 기능 사용 예시

Data Update Rules

Column의 데이터가 변경되었을 시(Changed, Increased, Decreased) Format을 변경하는 기능을 제공합니다.

[그림 26.] 같이 Data Update Rules 기능을 선택 후, Data Update Rules 창에서 value의 변경 조건값 (Changed, Increased, Decreased)을 선택합니다. 이후, 조건값에 대한 조건들(Text 굵기, RED/GREEN/YELLOW 색 표현, 등) 중 한 가지를 선택합니다. Apply formatting to an entire row선택 시, 해당 Cell의 row들 모두 조건을 반영할 수 있습니다.

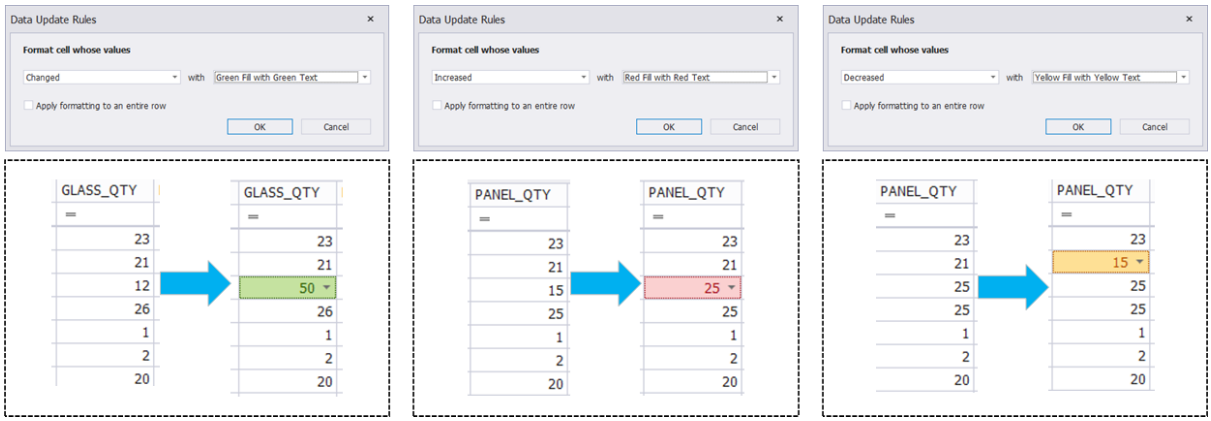


그림 26. Data Update Rules 기능 사용 예시 (왼쪽: Changed, Increased, Decreased)

Data Bars

Column의 데이터 값에 따라 Bar가 표시되는 기능을 제공합니다. Data Bar는 컬럼의 전체 범위 중 셀의 값이 차지하는 율에 따라 Bar 길이가 자동 생성됩니다. 최고값은 셀이 전체 채워지고, 최소값은 채워지지 않게 됩니다. Data Bars는 2가지 (Gradient Fill, Solid Fill)로 나누어지며 Gradient Fill은 음영처리된 색상을 나타내며, Solid Fill은 원색을 나타냅니다. 아래 [그림 27.]는 Data Bars의 2가지 형태 (Gradient Fill, Solid Fill)를 나타내며, Data Bars 기능을 선택 후 사용자가 원하는 Data Bars 형태를 선택하여 사용할 수 있습니다.

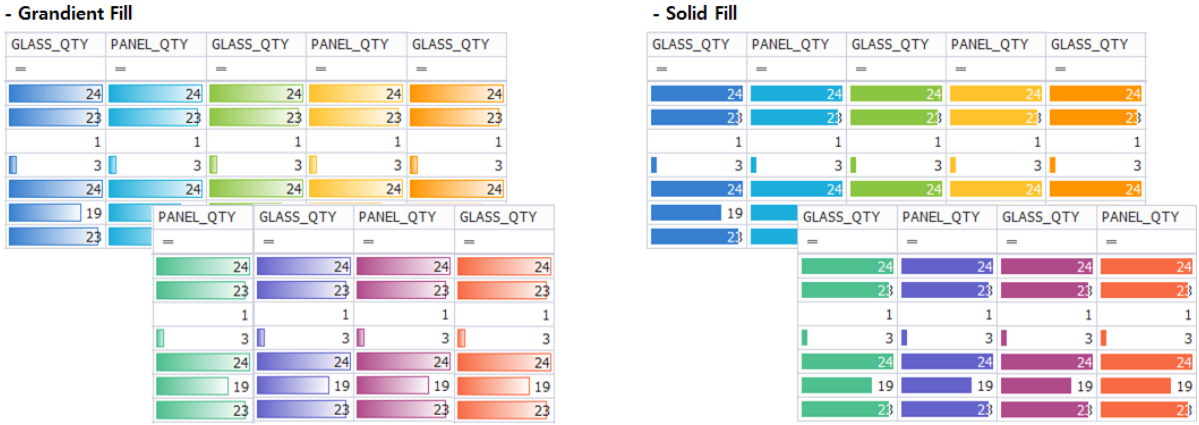


그림 27. Data Bars 기능 사용 예시 (왼쪽: Gradient Fill, 오른쪽: Solid Fill)

Color Scales

Column의 데이터 값에 따라 Color의 색상을 나타내는 기능을 제공하며, 아래 [그림 28.]은 Color Scales의 형태를 나타냅니다. 선택한 컬럼의 전체 값 중 최소값과 최대값 사이를 그라데이션 형태의 색상으로 표시됩니다. Color Scales 기능 선택 후 사용자가 원하는 Color Scales를 선택하여 사용할 수 있습니다.

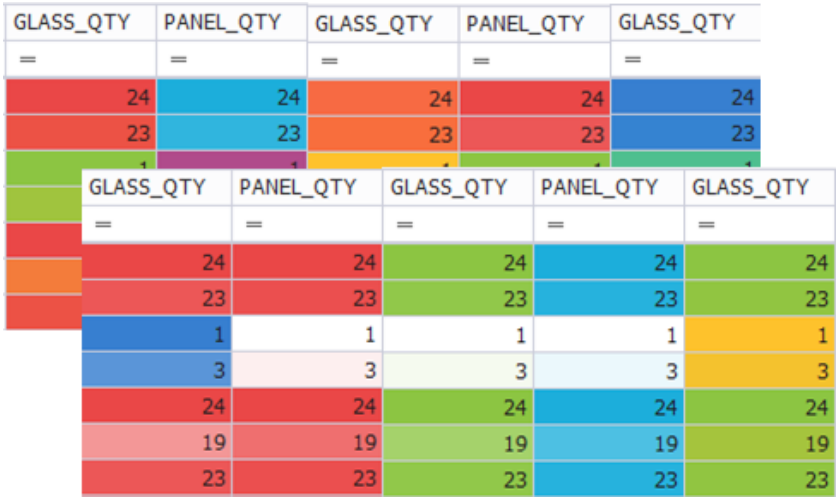


그림 28. Color Scales 기능 사용 예시

Icon Sets

Column의 데이터 값을 Icons(Ratings, Directional, Shapes, Symbols, Positive/Negative)으로 표시하는 기능을 제공합니다. 선택한 아이콘의 모양에 따라 전체 범위 중 차지하는 비율 기준으로 표시할 수 있습니다. 아래 [그림 29.]은 Icon Sets의 5가지 형태를 나타내며, Icon Sets 기능을 선택 후 사용자가 원하는 형태를 선택하여 사용할 수 있습니다.

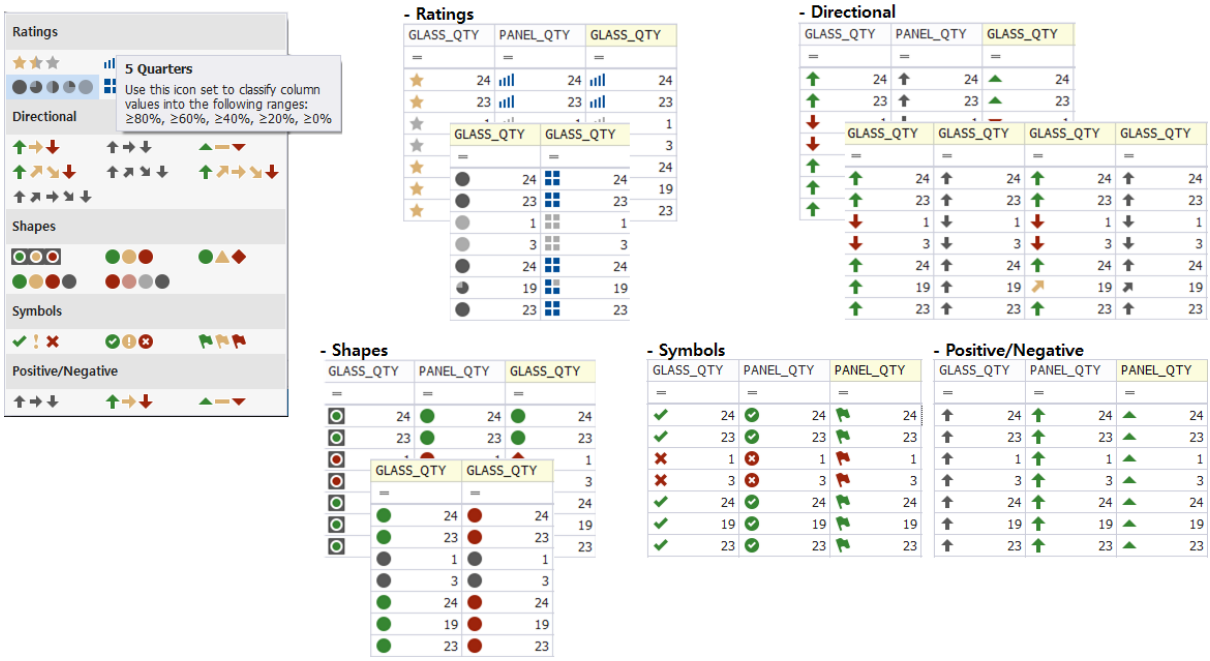


그림 29. Icon Sets 기능 사용 예시

Clear Rules

Clear Rules from This Column

Column에 Conditional Formatting기능을 사용한 뒤 해당 Column으로부터 기능을 제거합니다. 아래 [그림 30.]은 Clear Rules from This Column의 사용 순서를 나타냅니다.

Clear Rules from All Columns

Column에 Conditional Formatting기능을 사용한 뒤 모든 Columns으로부터 기능을 제거합니다.

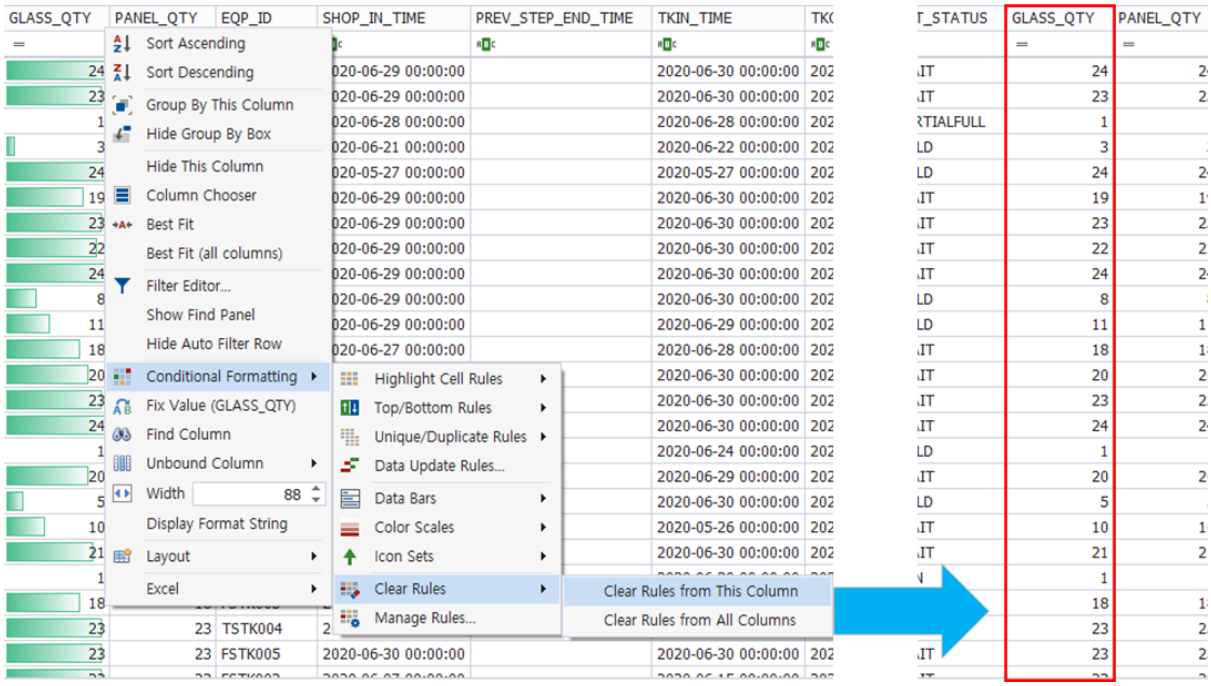


그림 30. Clear Rules 중 Clear Rules from This Column 기능 사용 예시

Data Join

Data Join 기능은 MOZART 2021.119.2 버전 이상에서 사용 가능합니다. 본 기능을 통해 사용자는 MOZART 모델의 데이터 아이템 간 조인 작업을 수행하고 결과 뷰를 저장하여 사용할 수 있습니다. Data Join 은 데이터베이스에서 지원하는 **조인 방식(Inner, Left, Right, Full)**을 모두 지원합니다. 본 기능은 MOZART 2020.118.0 버전에 추가된 Open Item(Read Only)기능에서 사용이 가능하며, Data Join 기능을 사용하는 자세한 방법은 Data Join 사용법을 참고하시기 바랍니다.

Fix Value

Fix Value는 선택된 해당 Column 또는 Cell에 속한 데이터를 한꺼번에 수정할 수 있는 기능을 제공합니다. 아래 [그림 31.]의 왼쪽은 Column 선택 후 Fix Value 기능을 통해 PRODUCT_TYPE의 Value 'Production'을 'Finish'로 변환된 모습을 보여주며, 오른쪽은 Cells 선택 후 Fix Value 기능을 통해 300 값으로 변환된 모습을 보여줍니다.

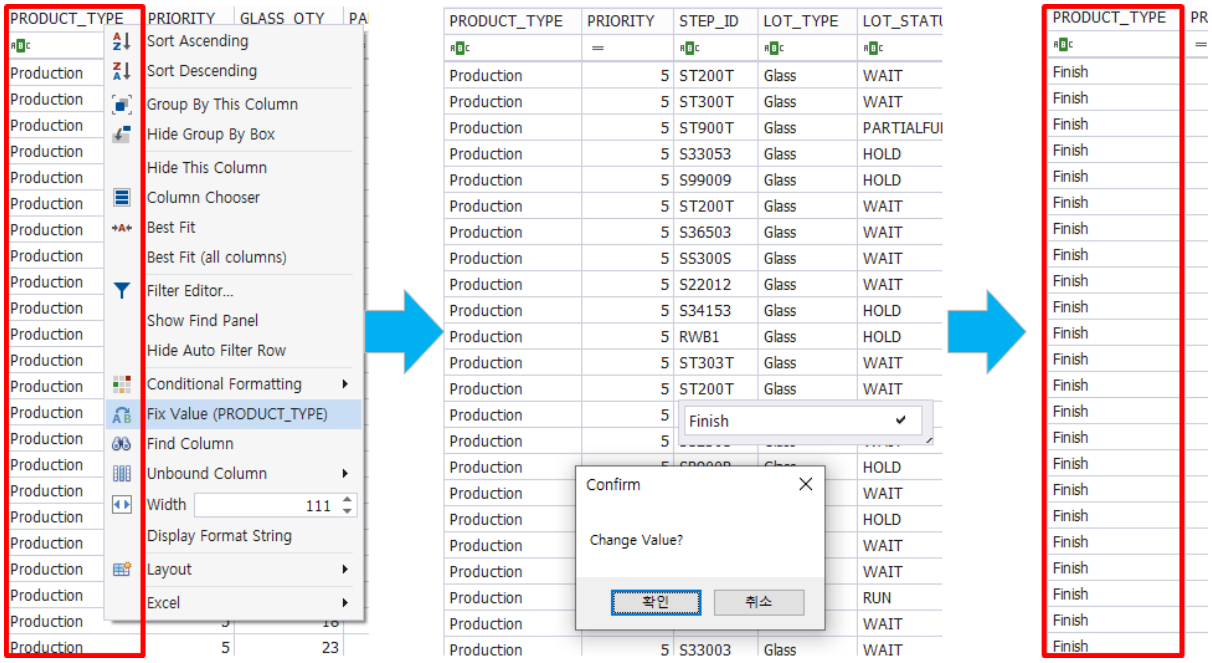


그림 31. Fix Value 예시 화면 (왼쪽: Column 선택 후 Fix Value 수행)

Fixed Column

Grid View에서 해당 Column을 왼쪽 또는 오른쪽으로 고정할 수 있는 기능입니다. 고정한 Column들은 Scroll에 영향을 받지 않고 항상 보여지게 됩니다.

열 고정하기

[그림 32.], [그림 31.]과 같이 고정을 원하는 Column 또는 Column의 Data를 우클릭하고 **[Fixed Column(Selected Column)]**을 누르면 Column을 고정하는 위치를 선택하기 위한 **[Left]**, **[Right]** 메뉴가 나타나고 각 항목의 하위에는 **[All Left(Right) by Column]**과 **[Current Column]** 메뉴가 나타납니다.

- **All Left(Right) by Column** : 해당 Column 왼쪽(오른쪽)에 존재하는 모든 Column이 왼쪽(오른쪽)으로 틀고정됩니다.
- **Current Column** : 해당 Column만 왼쪽(오른쪽)으로 틀고정됩니다.

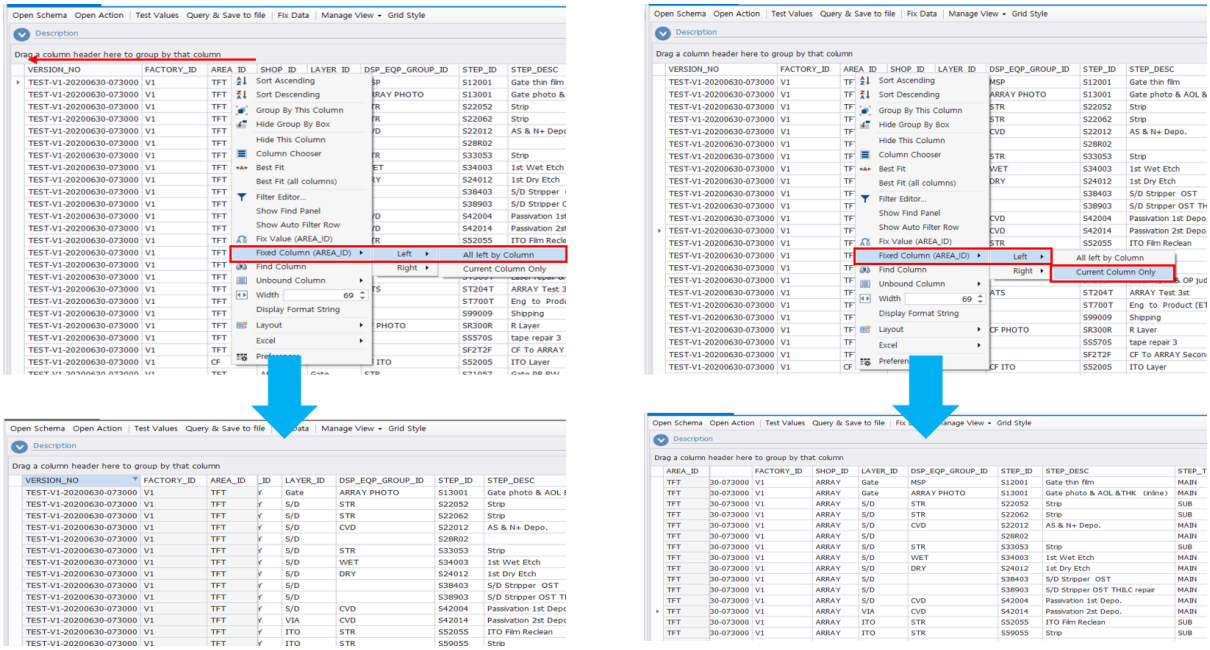


그림 32. Fixed Column-Left

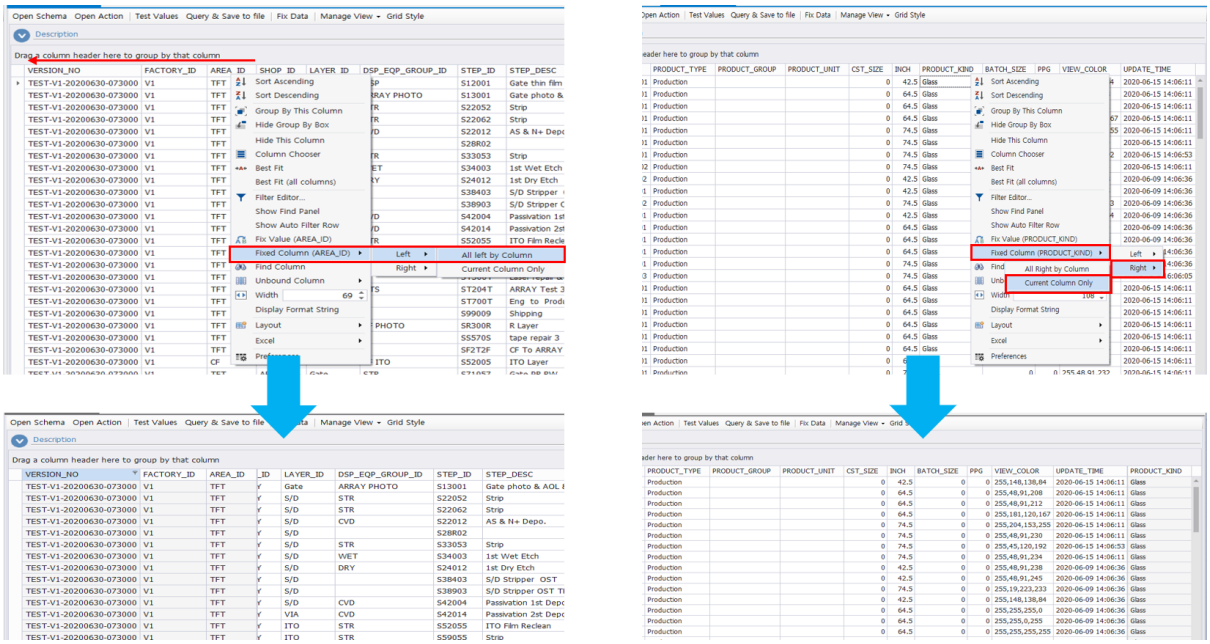


그림 33. Fixed Column-Right

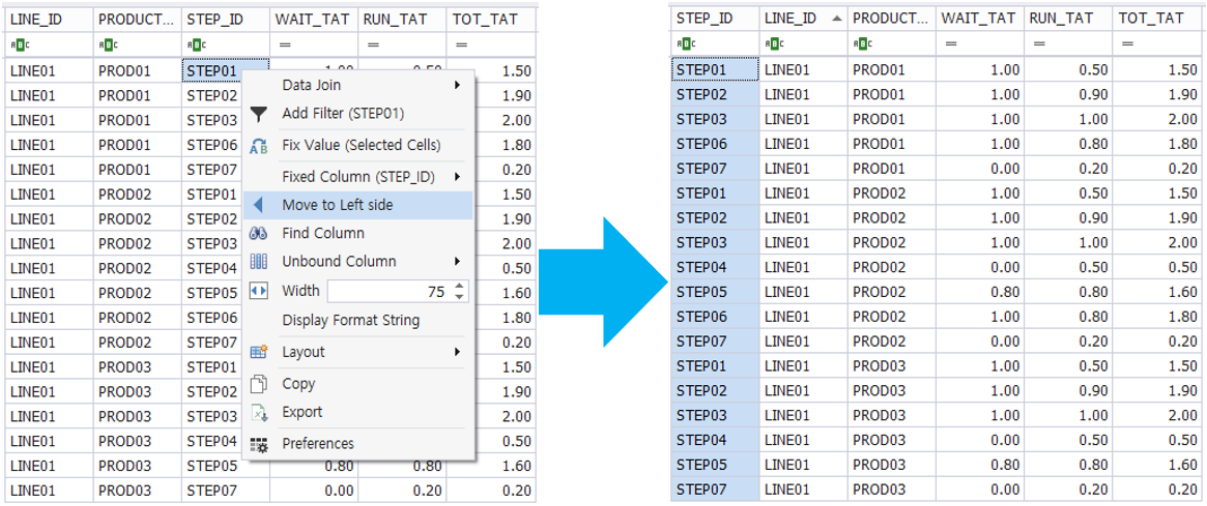
열 틀고정 해제하기

틀고정을 해제 하기 위해서는 틀고정된 Column 또는 Column의 Data를 우클릭하고 **Fixed Column(Selected Column)**을 누르면 현재 선택한 Column이 Fix된 위치의 메뉴가 활성화됩니다. 활성화 된 메뉴(Left or Right)를 선택하면 **All Left(Right) by Column**과 **Current Column** 메뉴가 나타납니다.

- **All Left(Right) by Column :** 해당 Column 왼쪽(오른쪽)에 존재하는 모든 Column이 순서대로 틀고정이 해제됩니다.
- **Current Column** : 해당 Column만 틀고정이 해제되며 원래 View의 기존 Column 위치로 돌아가게 됩니다.

Move to Left Side

GridView에서 선택한 Column을 가장 왼쪽으로 옮기는 기능이며 아래 **Find Column**의 [<] 왼쪽 버튼 기능과 동일합니다. Column을 우클릭하고 **Move to Left Side** 버튼을 눌러 Main Table의 가장 왼쪽 Column으로 이동합니다.



Move to Left Side를 통해 STEP_ID을 Column의 가장 좌측으로 이동한 예시

Find Column

Find Column은 많은 Column 중 사용자가 원하는 Column을 쉽게 찾을 수 있으며, 해당 Column을 맨 앞으로 위치하여 쉽게 사용할 수 있습니다. 아래 [그림 34.]은 Find Column 기능을 사용한 예시를 나타냅니다.

Find Column 기능을 선택 후, Find Column 창에서 Column을 선택합니다. 이후, 돋보기 버튼을 통해 선택된 Column을 찾으며, [<] 왼쪽 버튼을 누르면 Main Table의 가장 왼쪽으로 Column을 옮길 수 있습니다.

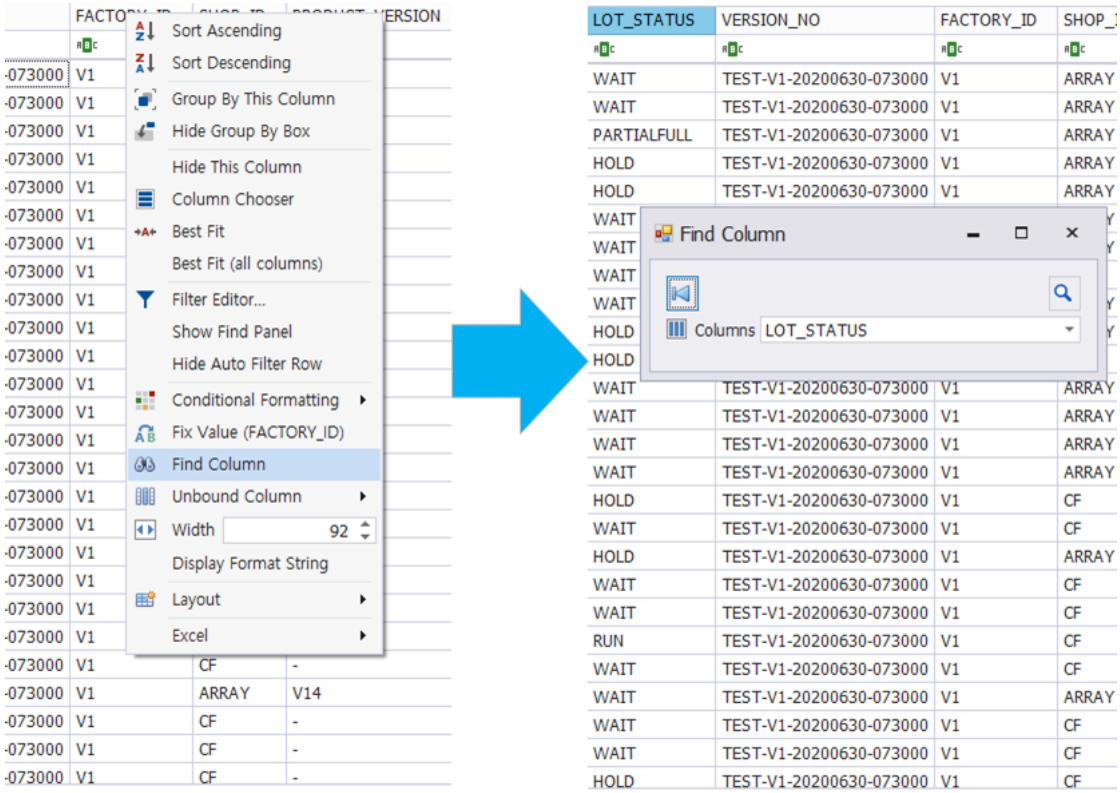


그림 34. Find Column을 통해 LOT_STATUS 찾은 후 Column 맨 앞으로 이동 예시

Unbound Column

Unbound Column은 추가적인 Column을 생성하기 위한 기능을 제공합니다. 해당 기능에서는 Add Unbound Column과 Copy Column 라는 2가지 세부 기능을 제공하며, 해당 내용은 아래 표와 같습니다. 아래 [그림 35.]은 Add Unbound Column에 대한 기능의 사용 순서를 나타냅니다.

Add Unbound Column 기능 사용 순서

Add Unbound Column	Copy Column
• Unbound Column 추가Expression	• 동일한 Column 생성
• Editor를 통해 Column 데이터 정의	• 값도 동일하게 나타남
• Column 색: Azur	• Column 색: LightYellow

Add Unbound Column

기존의 Column들을 조합하여 새로운 Column을 생성하기 위한 기능을 제공합니다. 해당 기능에서는 Columns을 선택한 뒤 Operations 및 Functions 이 제공되며, 기존의 Column과는 다른 새로운 Column을 생성할 수 있습니다.

[그림 35]과 같이 Add Unbound Column 기능을 선택 후, Unbound Column 창에서 Name과 Type (Integer, String, Decimal, DateTime, String, Boolean)를 선택합니다. 이후, Expression Editor 창에서 데이터에 필요한 Column들을 선택하며, 선택된 Column들의 조건들을 선택할 수 있습니다. 조건들은 Constrains (True, False, ?), Operators (+, - , *, /, %, etc.) 그리고 Functions (DateTime, Logical, Math, String, All 에 대한 기능) 선택할 수 있습니다.

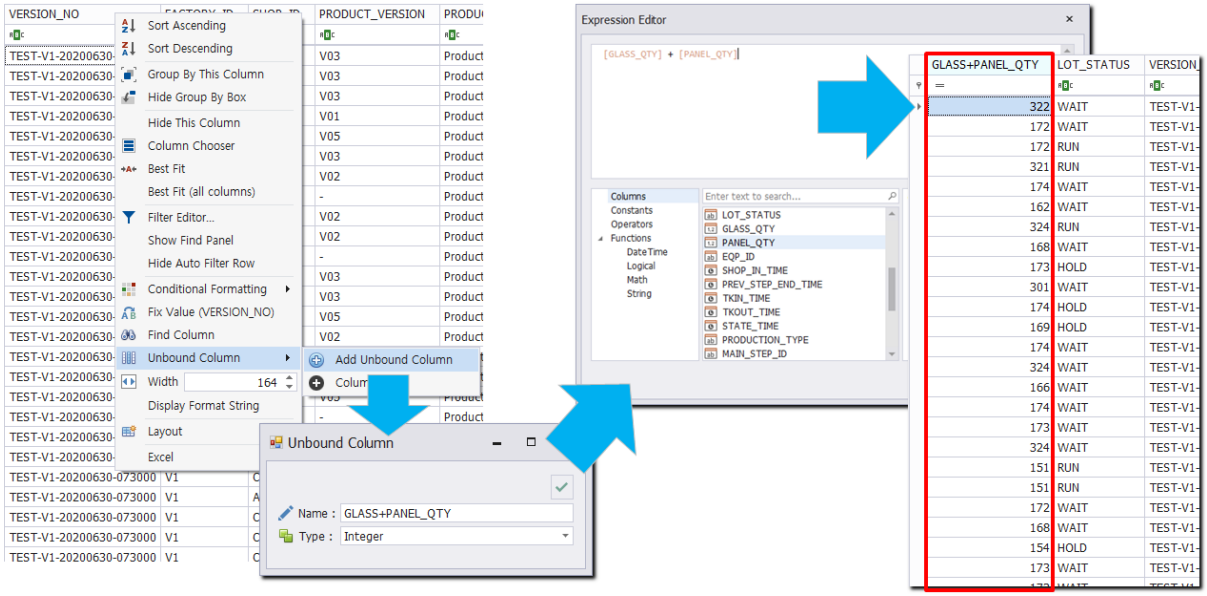


그림 35. Add Unbound Column 기능을 통한 새로운 Column 생성 예시

Column Copy

기존의 Column들을 복사하여 똑같은 Column을 생성하기 위한 기능을 제공합니다. Column Copy 기능을 선택하면, 해당 Column을 복사하여 Main Table의 맨 앞에서 생성됩니다. [그림 36.] Column Copy의 사용 순서를 나타냅니다.

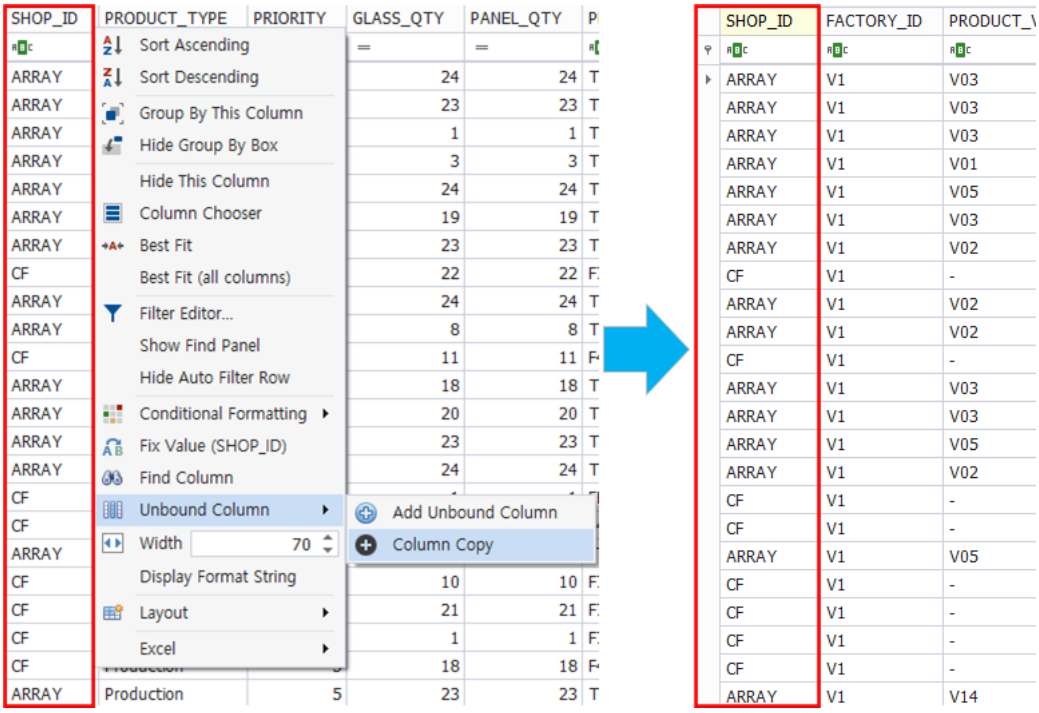


그림 36. Column Copy 기능을 통한 Column 생성 예시 (SHOP_ID)

Column Copy (Converted)

숫자형으로 된 컬럼의 값이 시간으로 사용될 때 시(Hour), 분(Min), 초(Sec) 의 단위로 변경된 값을 사용하는 새로운 컬럼으로 추가할 수 있습니다. 예(Min → Sec : 단위가 분(Min)인 기존 값을 초(Sec) 단위로 변환된 값이 추가되는 새로운 컬럼을 생성합니다.)

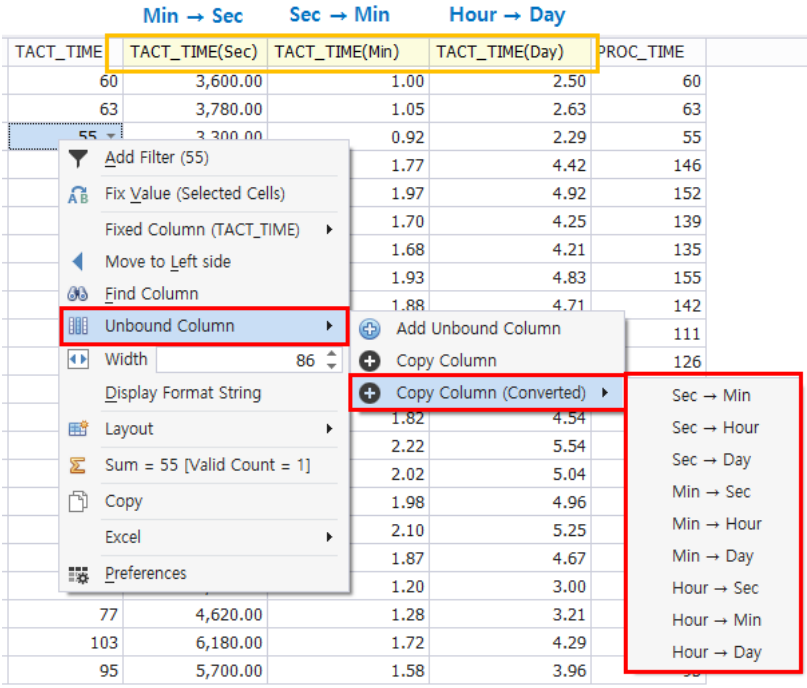


그림 37. Copy Column (Converted) 기능을 통한 시간 단위 변화 (TACT_TIME)

Width

본 기능은 사용자가 Column 너비를 설정할 수 있는 기능을 제공합니다. [그림 37.]은 Width의 사용 예시를 나타냅니다. 사용자가 Width값을 변경하지 않으면 default값으로 해당 column의 width만큼 값을 나타냅니다.

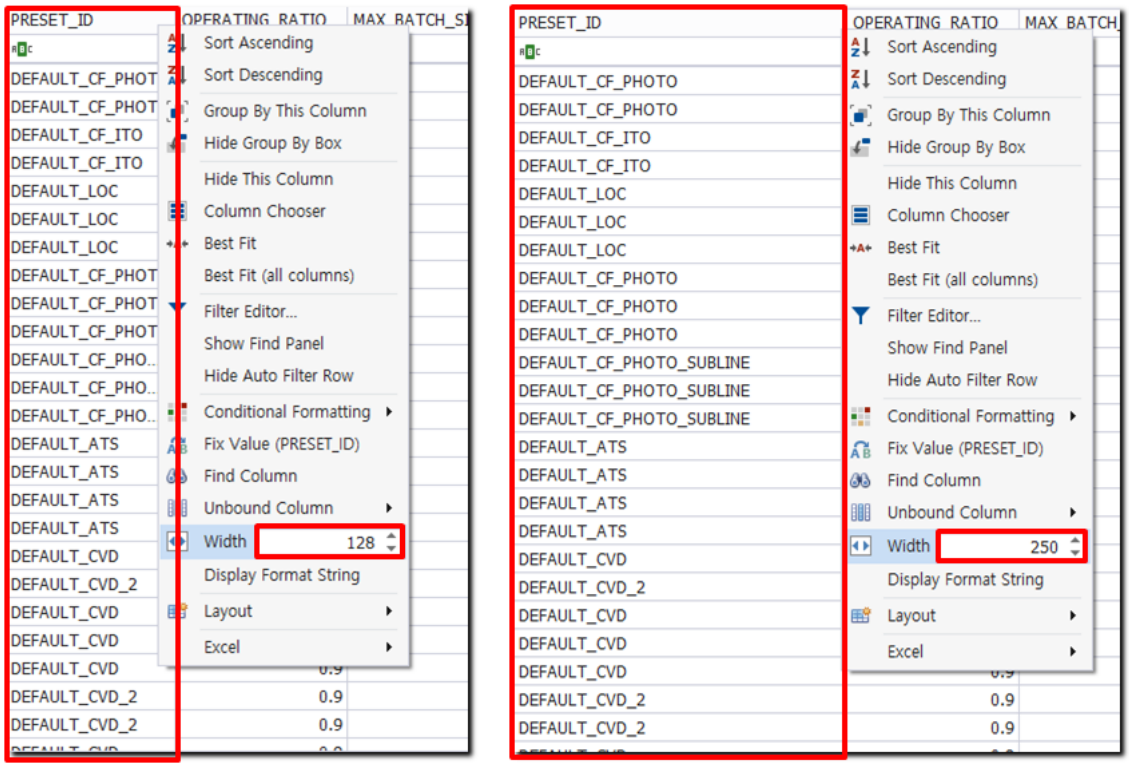


그림 37-1. Width기능을 통한 Column 너비 변경 예시 (왼쪽: Width 128, 오른쪽: Width 250)

MOZART 2021.120.0 버전 이상에서 여러 개의 Column을 일괄적으로 동일한 너비 변경이 가능합니다. 너비를 변경하고자 하는 Column의 일부 셀을 여러 개 선택하여 Width 값을 변경합니다.

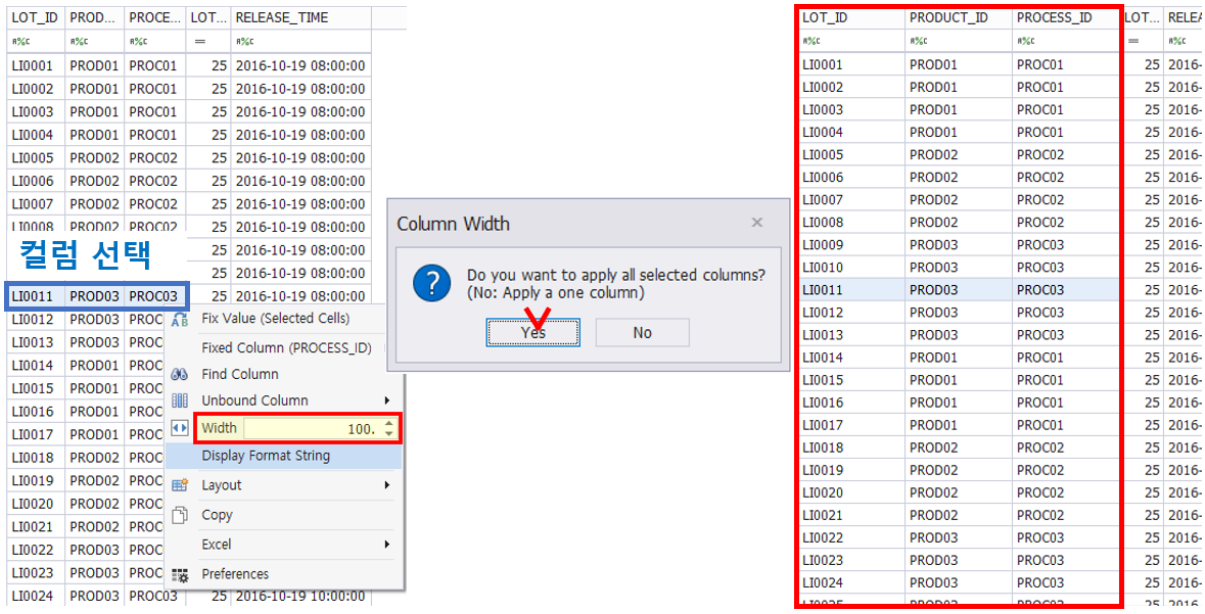


그림 37-2. Width기능을 통한 Column 너비 일괄 변경 예시 (Width 100으로 변경)

Display Format String

사용자가 원하는 String Format Type을 지정할 수 있도록 Column의 Format 값을 변경하는 기능을 제공합니다. 변경 가능한 Format Type은 4가지 (None, Numeric, DateTime, Custom)이 있으며, FormatString에서 해당 값을 변경할 수 있습니다.

[그림 38.] 같이 Display Format String 기능을 선택 후, DisplayFormatString 창에서 FormatType (None, Numeric, DateTime, Custom)들 중 한 가지와 FormatString에 Format값을 입력합니다. 이후, 입력된 Format 값에 따라 Column값이 변경됩니다.

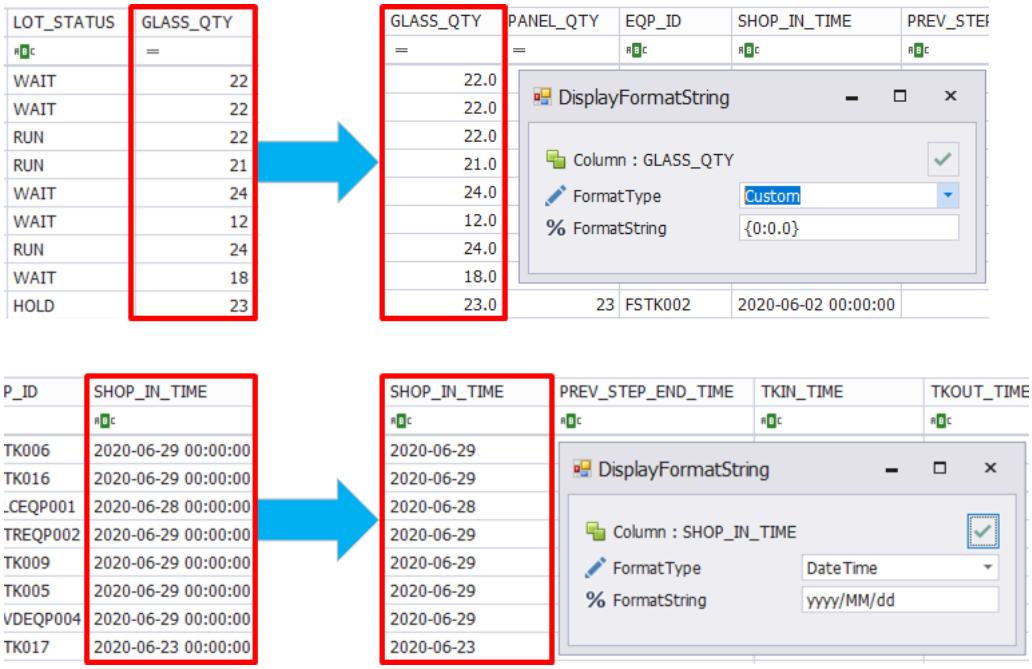


그림 38. Display Format String 기능 예시 (위: GLASS_QTY 1자리수 변경, 아래: SHOP_IN_TIME 시,분,초 제거)

Layout

Data Item에서 Layout은 컬럼 이동, 복사, 정렬, 그룹핑 등 다양한 형태로 View를 볼 수 있습니다. 다양한 형태로 설정한 View는 **Layout** 메뉴 기능을 통해 해당 Layout을 저장하고 관리 할 수 있습니다. Layout 정보는 일반 Data Item, Grid View, Pivot Grid View 별로 관리 되고, 설정 파일은 *%AppData%\mozart\XXX Studio\ViewInfo* 경로에 저장됩니다.

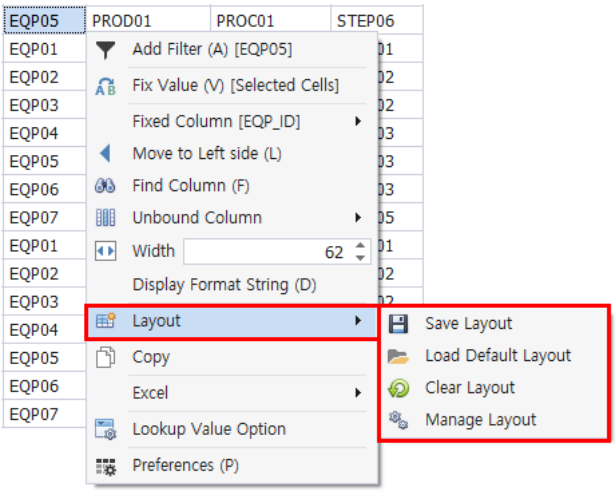


그림 39-1. Layout 메뉴

2022.122.0 버전 업데이트 이후
- 여러 개 Layout을 저장하고 관리 할 수 있습니다.
- 2022.122.0 이전 버전에서 해당 View에 대한 Layout 설정이 있었던 경우, **Manage Layout** 다이얼로그에 'Old Layout'으로 자동 저장 됩니다. (설정 파일 확장자 변경 : *.xml → .vgrid)*
- 한 View에 대한 Layout 정보 파일을 Export/Import 가 가능합니다. (참조)

Save Layout

현재 View에 설정된 Layout을 저장하는 기능입니다. **Save Layout** 다이얼로그에는 현재 View에 저장된 Layout 목록이 나타나고 아래에는 Layout 저장 설정 옵션이 있습니다. Save as 옵션을 통해 새로운 Layout을 저장하거나 기존 Layout을 선택(적색 Row 적용)하여 Layout을 덮어쓰기 합니다.

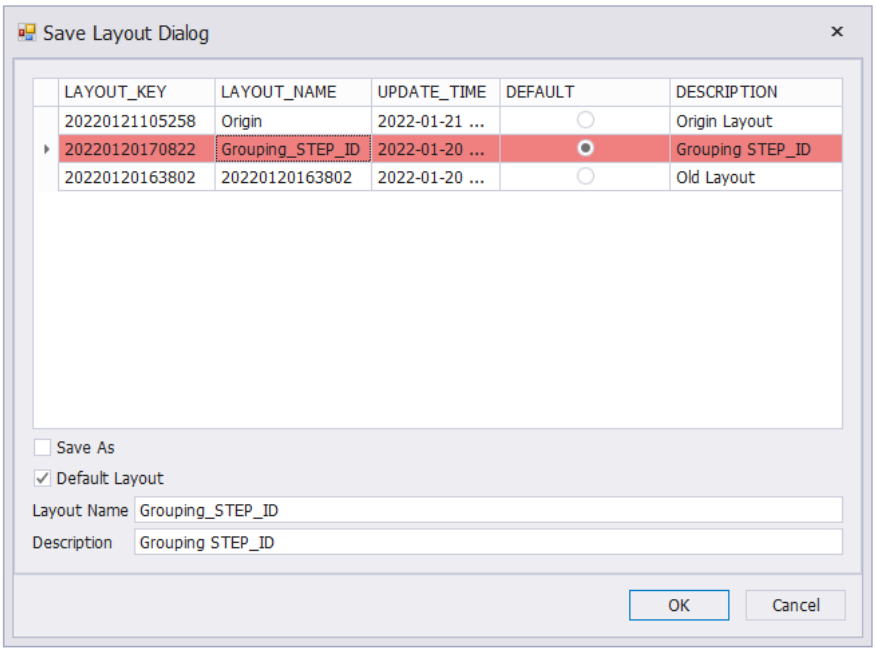


그림 39-2. Layout > Save Layout Dialog

- **Save as** : 새로운 Layout을 저장하는 옵션입니다. 체크 해제한 경우 선택한 Layout(적색 Row)을 덮어쓰기 저장합니다.
- **Default Layout** : 저장하는 Layout을 Default로 설정합니다.
- **Layout Name** : 저장하는 Layout의 이름입니다.
- **Description** : 저장하는 Layout의 설명입니다.

Load Default Layout

Default 설정된 Layout을 불러오는 기능을 제공합니다. **Manage Layout** 다이얼로그에서 Default Layout 확인 및 설정이 가능합니다.

Clear Layout

설정한 Layout이 없는 초기 상태의 Data View로 돌아갑니다. **Load Default Layout** 메뉴 또는 **Manage Layout** 메뉴의 **APPLY**를 통해 저장한 Layout을 다시 적용 할 수 있습니다.

Manage Layout

해당 View에 저장된 Layout 정보를 확인하고 관리하는 기능을 제공합니다. Layout 정보는 일반 Data Item View, Grid View, Pivot View 각 개별로 관리됩니다. Manage Layout 다이얼로그를 통해 Layout의 내용 수정, 삭제, 적용, Default 설정 등이 가능하며, 최근 적용된 Layout (녹색 Row) 확인이 가능합니다. 또한 Import/Export 기능을 통해 Layout 정보 파일을 저장하고 가져올 수 있습니다.

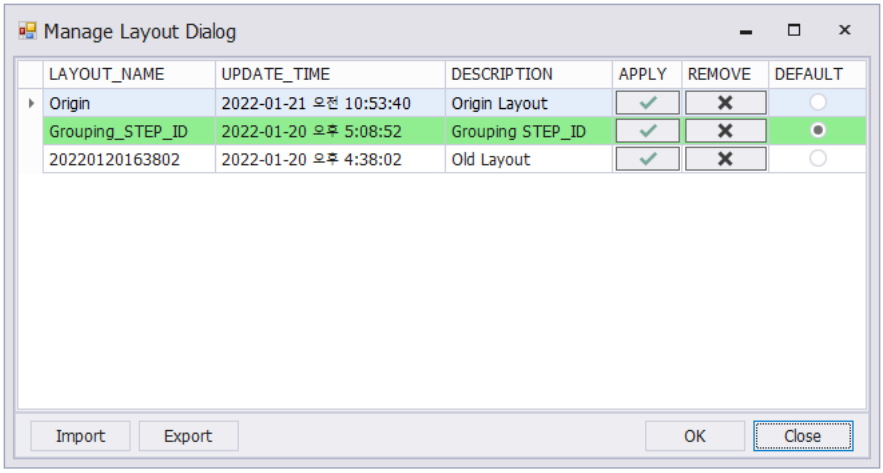


그림 39-3. Layout > Manage Layout Dialog

- **LAYOUT_NAME** : Layout의 이름입니다.
- **UPDATE_TIME** : Layout이 생성된 시간입니다.
- **DESCRIPTION** : Layout의 설명입니다.
- **APPLY** : Layout을 View에 적용합니다.
- **REMOVE** : Layout을 지웁니다.
- **DEFAULT** : Default Layout을 지정합니다. Default Layout은 View를 열거나 **Load Layout** 실행하면 나타나는 Layout 입니다.
- **Import / Export** : 해당 View에 저장된 Layout 정보를 *.grid* 파일로 추출하거나 가져오는 기능입니다. Import 시 동일한 Key (수정 시간 기준) 값을 가지는 Layout이 중복되는 경우 덮어쓰기 여부를 확인합니다.

Excel

본 기능은 Mozart Studio의 Column과 Cells에 입력된 데이터를 Excel 파일로 입/출력하는 기능 제공합니다. 해당 기능에서는 2가지 세부 기능인 Export To Excel과 Import From Excel 기능을 제공하며, 상세 내용은 아래와 같습니다.

Export

Mozart studio에서 출력된 데이터를 엑셀 파일로 내보내는 기능을 제공합니다. 아래 **[그림 40.]**은 Export의 사용 순서를 나타냅니다. Export 기능을 선택 후, Export to Microsoft Excel Document 창에서 데이터가 저장될 장소를 확인합니다. 이후, 저장된 정보가 Excel 정보로 Export되었음을 확인할 수 있습니다.

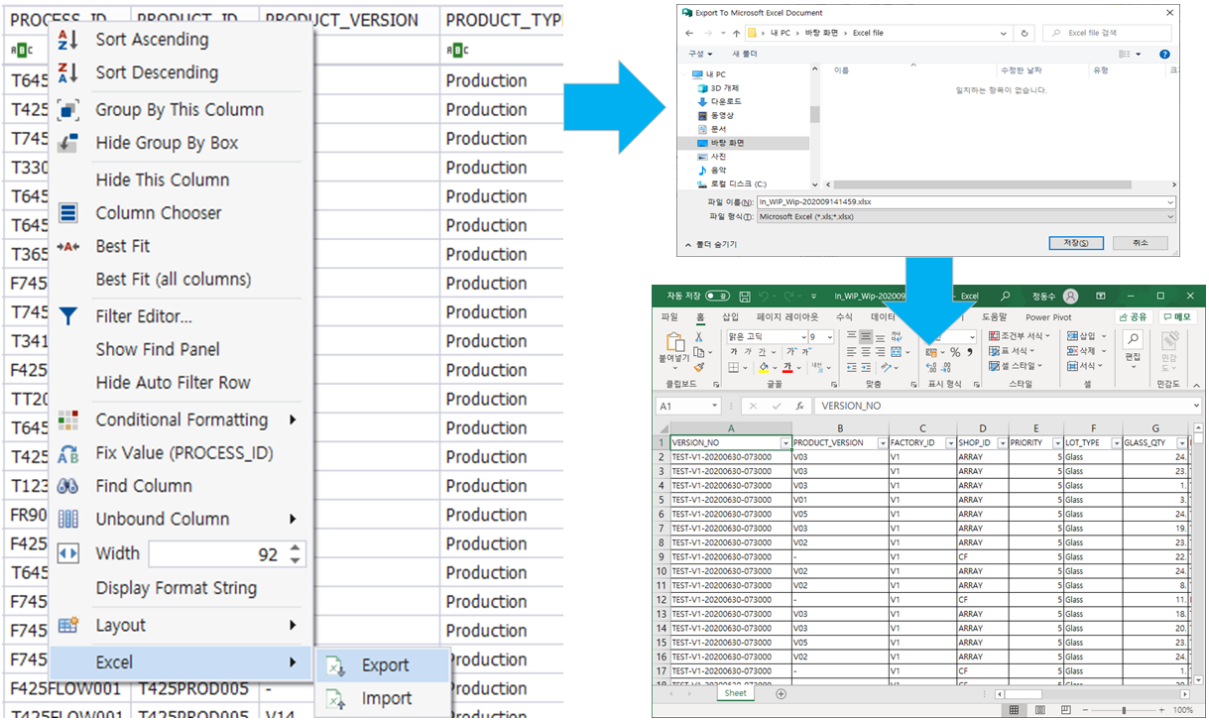


그림 40. Export 기능 사용 예시

Import

엑셀 파일로 저장된 데이터를 Mozart 데이터로 가져오는 기능을 제공합니다. 아래 **[그림 41.]**은 Import의 사용 순서를 나타냅니다. Import 기능을 선택 후 Import할 데이터를 선택한 뒤, 이를 Input Excel Data에 입력합니다. 이후, Append 버튼을 누르면 기존 Main Table에 정보가 추가로 더해지며, OK 버튼을 누르면 해당 정보만이 Main Table에 입력됩니다.

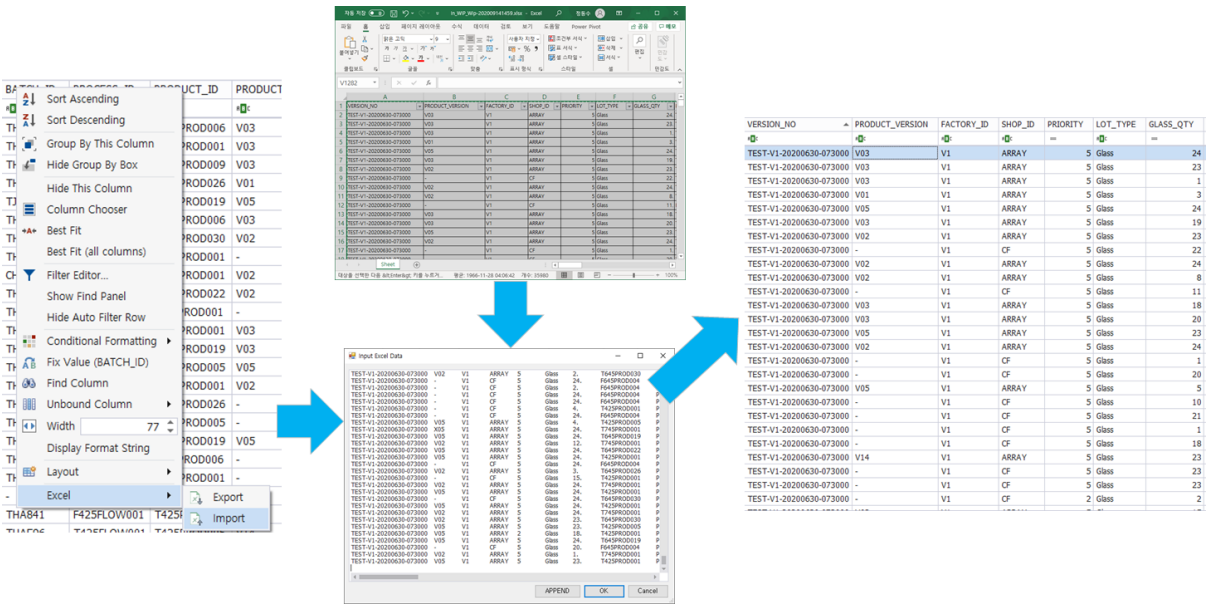


그림 41. Import 기능 사용 예시

Grid View Preference

각 Grid View별로 Option을 설정할 수 있는 기능입니다. 모델의 모든 Grid View들의 기본 Option은 **[Tool] > [Options] > [GridOptions]**을 따르게 되어 있으나 Grid View Preferences 기능을 통해 개별 Grid View의 Option을 설정할 수 있습니다.

[그림 42.]과 같이 해당 Grid View의 Columns or Cells을 우클릭한 후 메뉴에서 **Preferences**를 선택하면 Grid Setting 팝업 창이 나타납니다. 각 기능에 대한 설명은 아래와 같습니다.

Preivew를 통해 각 속성 값에 따른 변경사항을 확인 할 수 있습니다.

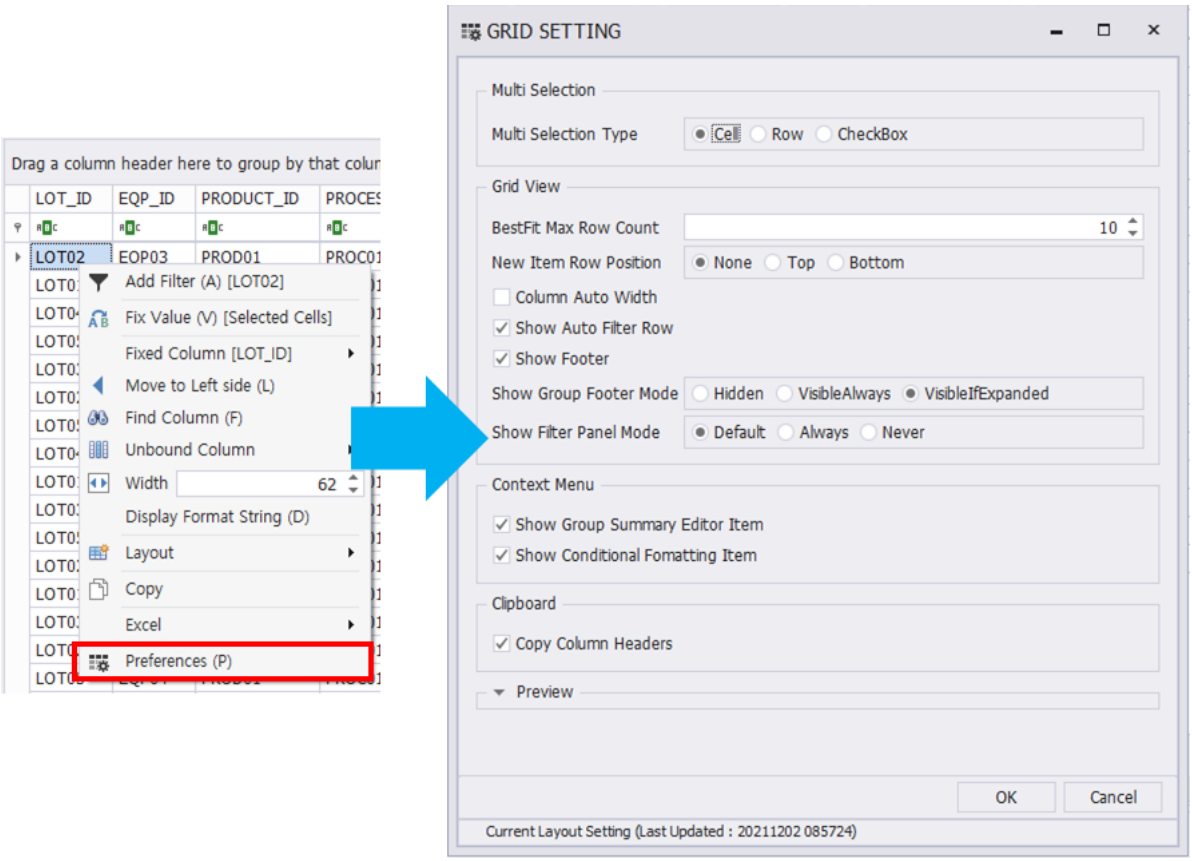


그림 42. Preferences > Grid Setting 화면

Multi Selection Type

View에서 Data를 클릭할 때 Selection Type에 대한 설정할 수 있습니다. Type에는 Cell, Row, CheckBox가 있으며, 각 Type에 대한 설명은 아래와 같습니다.

- Cell : Cell 별로 Data를 선택할 수 있습니다.
- Row : Row 별로 Data를 선택할 수 있으며, Cell을 따로 선택할 수는 없습니다.
- CheckBox : CheckBox를 Check 하여 Data를 선택할 수 있습니다.

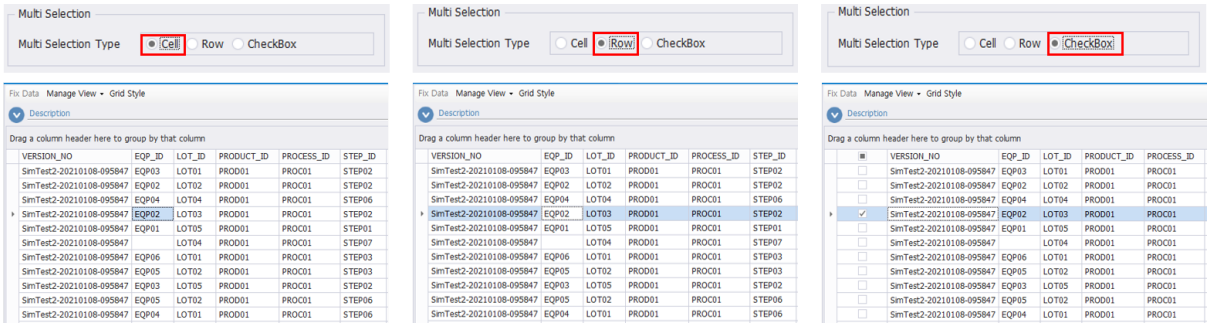


그림 43. Multi Selection Type 별 GridView

BestFit Max Row Count

BestFit Max Row Count에서 설정한 수만큼 Row Data Value Length를 고려하여 Column의 Width가 설정됩니다. Count Default는 -1이며 View 존재하는 모든 Row의 Data Value Length를 고려하여 설정됩니다.

[그림 44.]에서 좌측 그림과 같이 Row Count가 -1일 경우, View에 존재하는 모든 Row Data를 고려하여 Column(SHOP_ID)의 Width가 조절되며 우측 그림과 같이 Row Count가 3일 경우, 3번째 Row Data까지만 고려하여 Column(SHOP_ID)의 Width가 조절됩니다.

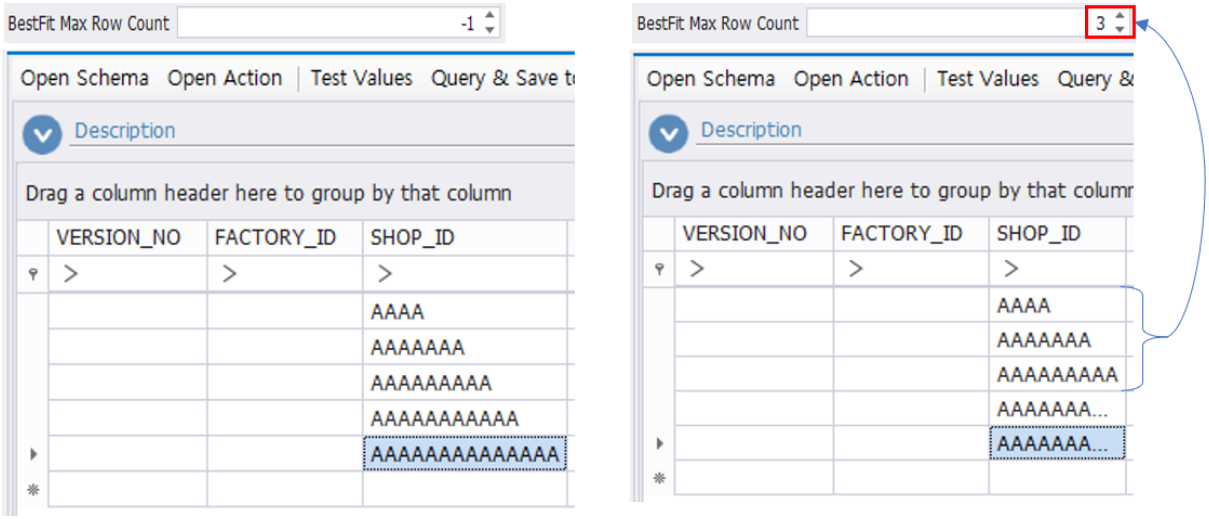


그림 44. BestFit Max Row Count

New Item Row Position

Grid View에서 빈 행이 표시되는 위치입니다. 사용자는 빈 행을 클릭하여 행을 Row에 추가할 수 있고 바로 Data를 입력하여 저장할 수 있습니다.

- None : 빈 행이 표시되지 않으며, View 하단의 Navigator에서 '+' 버튼을 통해 Row를 추가할 수 있습니다.
- Top : 빈 행이 View의 상단에 위치합니다. Row에 추가된 Data들은 상단에 추가되는 것이 아니라 하단에 위치하게 됩니다.
- Bottom : 빈 행이 View의 하단에 위치합니다. Row에 추가된 Data들은 하단에 위치하게 됩니다.

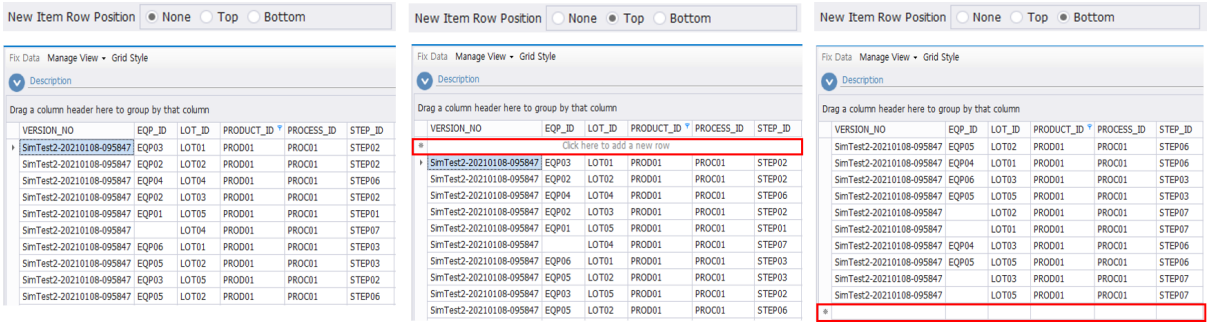


그림 45. New Item Row Position 비교

Column Auto Width

각 Column들의 Width를 자동으로 조절해주는 기능입니다. Column Auto Width 기능을 체크할 경우 Column들은 하단부의 Scroll 없이 한 화면에 해당 View의 모든 Column이 보여질 수 있도록 Width가 자동적으로 설정됩니다.

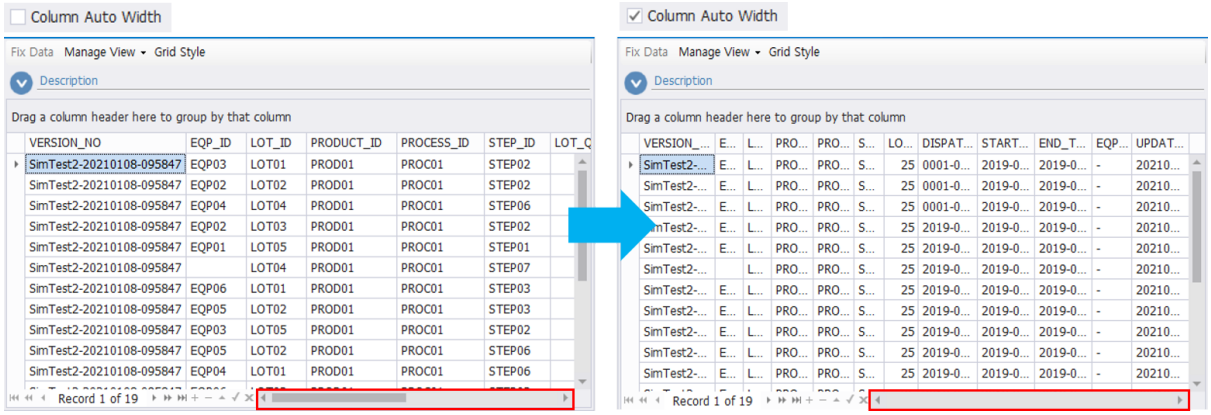


그림 46. Column Auto Width 비교

Show Footer

각 Column의 Sum, Min, Max 등을 보여주는 Footer를 보여주는 기능입니다. Show Footer을 Check하여 항상 하단에 Footer가 보이도록 설정할 수 있습니다. 또한 **[그림 48]**과 같이 Footer 영역에서 컨텍스트 메뉴의 Copy 기능을 이용하여 값을 복사하여 사용할 수 있습니다.

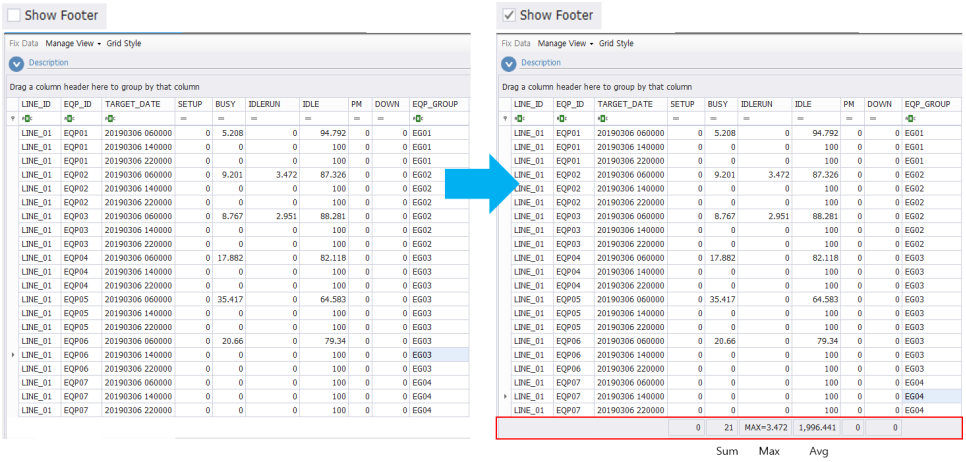


그림 47. Show Footer 비교

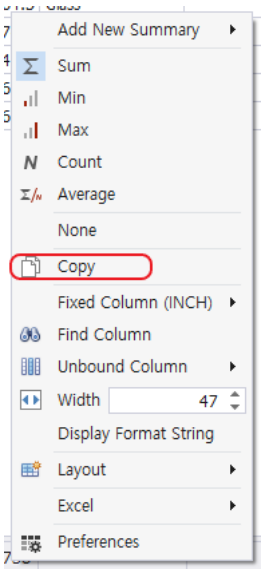


그림 48. Footer Copy

Show Auto Filter Row

각 열들에 Filter를 검색하듯이 적용할 수 있는 Filter Row를 보여주는 기능입니다. Filter Row의 기본 Filter 종류는 **[Tool]** > **[Options]** > **[DataView]**에서 Filter Condition을 통해 설정할 수 있으며, 개별 Grid View에서는 각 열의 Filter Icon을 클릭하여 변경할 수 있습니다.

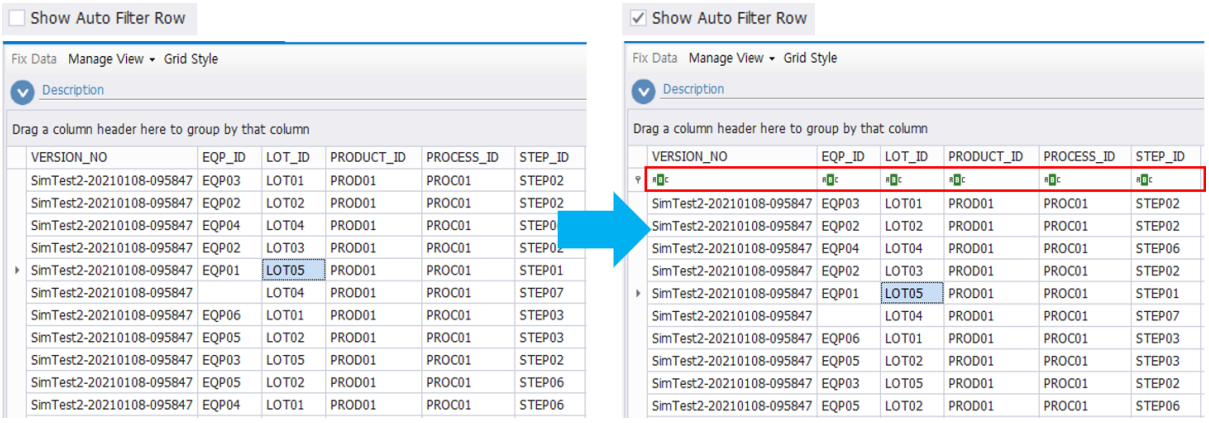


그림 46. Show Auto Filter Row 비교

Show Group Footer Mode

[Group Summary Editor](#) 를 사용할 때 Footer를 표시 방법을 설정할 수 있습니다. 기본값은 VisibleIfExplanded 입니다.

- Hidden : Footer가 표시되지 않도록 합니다.
- VisibleAlways : Group Summary에서 항상 Footer가 표시됩니다.
- VisibleIfExpanded : Group Summary 에서 Group 정보를 확장했을 때 Footer가 표시됩니다.

Show Filter Panel Mode

현재 Grid View에 적용된 Filter 정보를 볼 수 있는 Filter Panel의 Mode를 설정하는 기능입니다.

- Default : Filter를 설정할 경우에만 Panel이 나타납니다.
- Always : Filter를 설정하지 않아도 Panel이 하단에 항상 존재합니다.
- Never : Filter Panel이 나타나지 않습니다. Panel이 없어도 Filter는 적용됩니다.

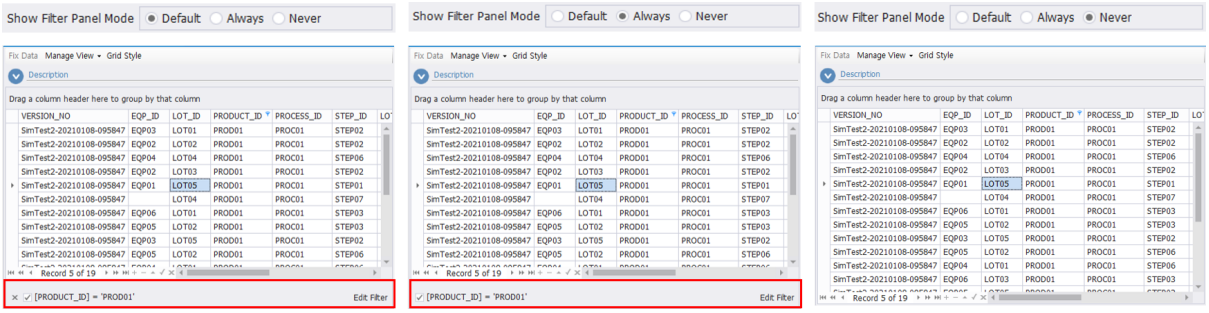


그림 48. Show Filter Panel Mode 비교

Show Group Summary Editor Item

Show Group Summary Editor Item을 Check하여 Grouping된 Column들은 우클릭했을 때 기능 목록에 Group Summary Editor 기능을 추가할 수 있습니다. Group Summary Editor의 자세한 기능은 [Grid View 사용법 > Grouping 및 Summary](#) 에서 확인하실 수 있습니다.

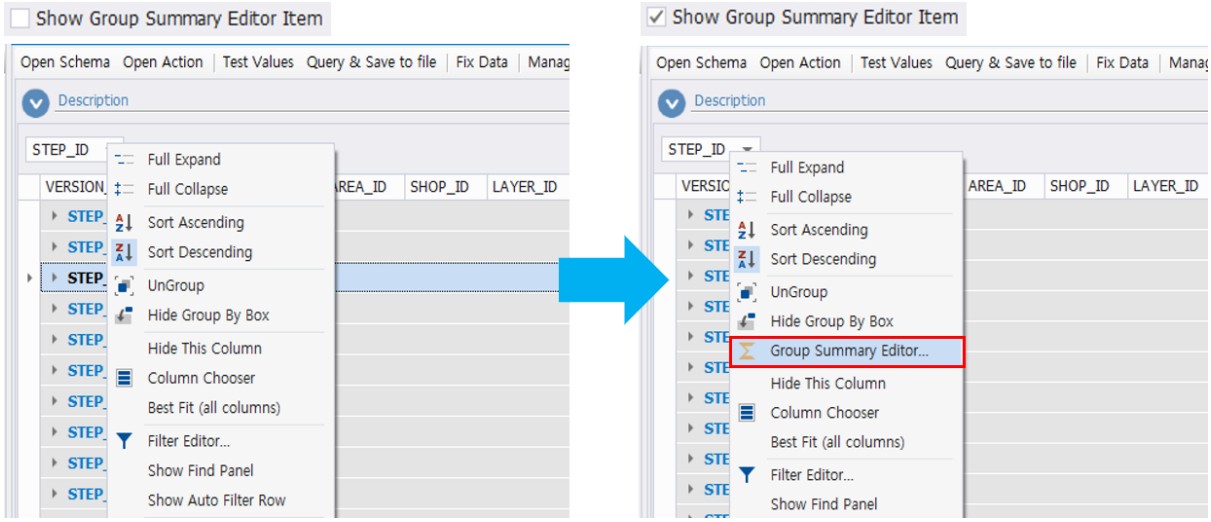


그림 49. Show Group Summary Editor Item

Show Conditional Formatting Item

Show Conditional Formatting item을 Check하여 Column을 우클릭 했을 때 나타나는 기능 목록에 Conditional Formatting 기능을 추가할 수 있습니다. Conditional Formatting 기능에 대한 자세한 설명 [Grid View 사용법 > Conditional Formatting]서 확인하실 수 있습니다.

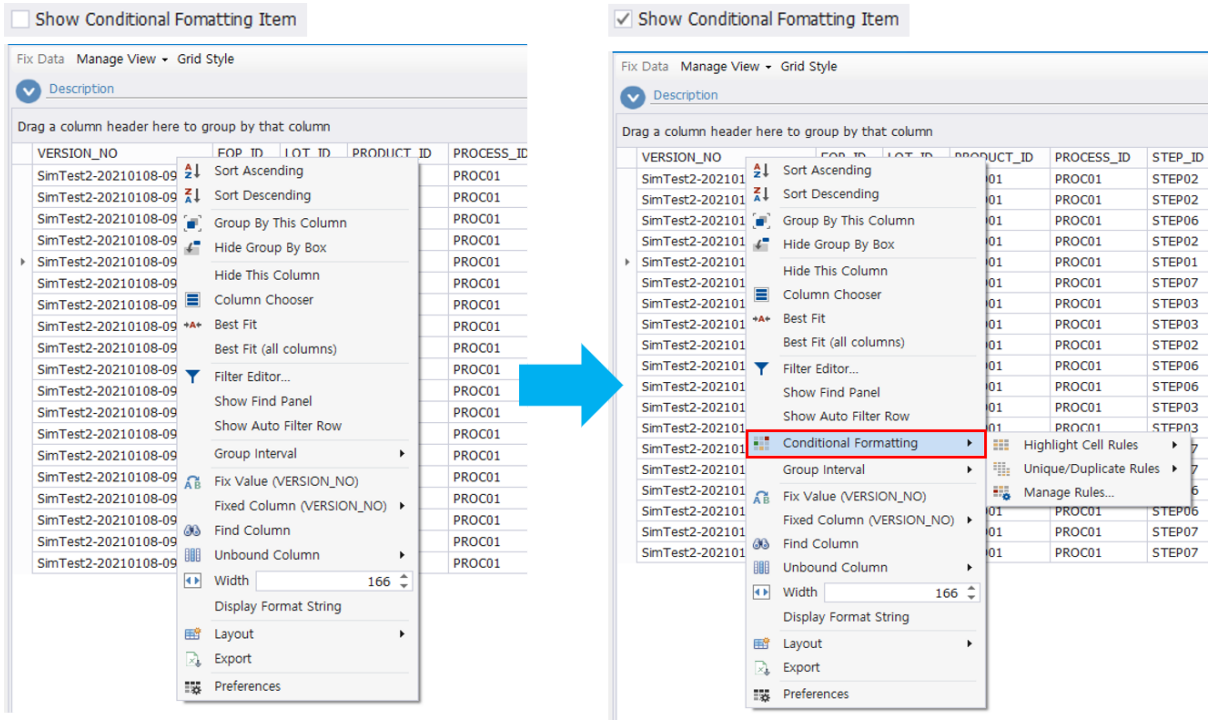


그림 50. Show Conditional Formatting Item

Copy Column Header

Grid View에서 Data를 복사할 때 Data가 속하는 Column Name의 복사 여부를 결정하는 기능입니다.

[그림 51.]과 같이 Copy Column Headers를 Check하면 해당 Data들의 Column Name이 함께 복사됩니다. Check하지 않으면 해당 Data들만 복사됩니다.

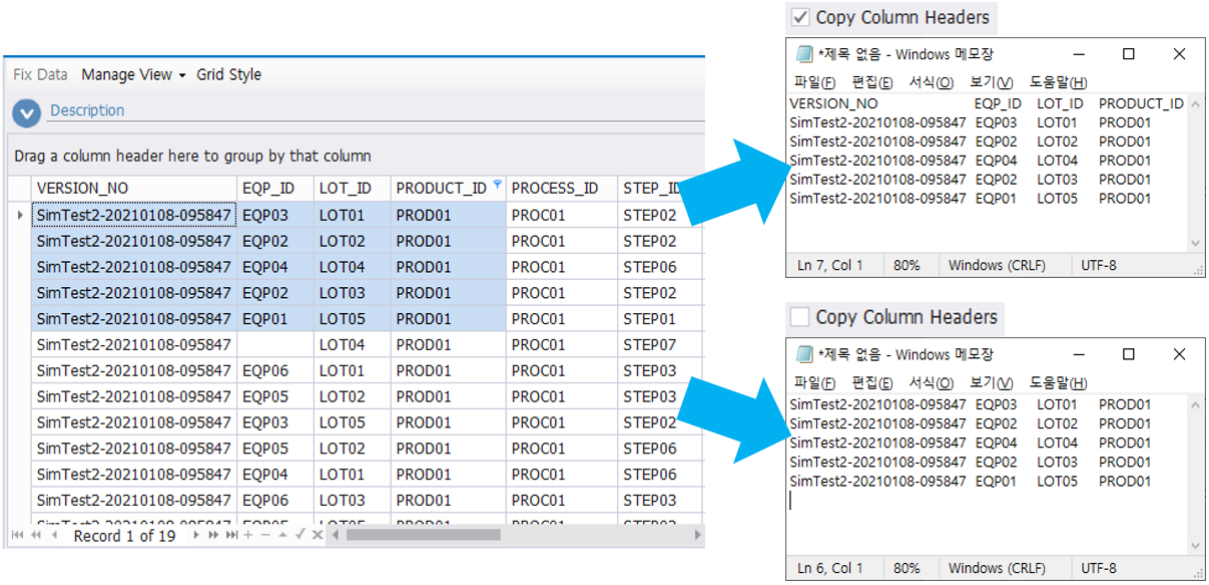


그림 51. Copy Column Header 비교

Cell 보조 메뉴

MOZART Studio의 Main View에서 제공되는 기능들로써, [그림 52.]과 같이 Cell을 우측 클릭하여 해당 기능을 확인 및 작동시킬 수 있습니다. Cell 보조메뉴에서 일부 기능은 Column Header에서 안내하는 기능과 동일합니다. Cell 전용 기능은 **[Tool] > [Options ...] > [Settings] > [DataView]**의 **Multi-Selection Type**이 Cell일 경우에만 사용할 수 있습니다.

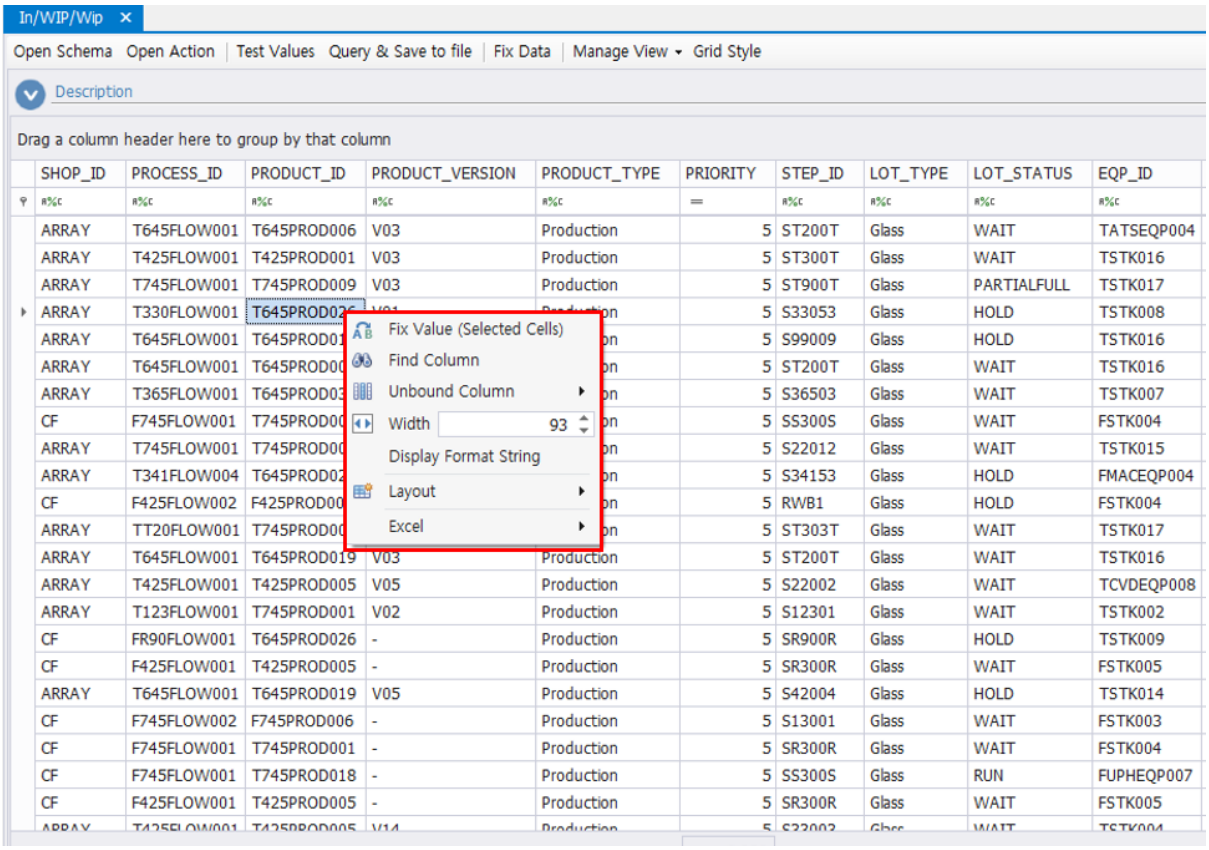


그림 52. Cell 클릭시 나타나는 기능들

아래는 UI 에서 사용할 수 있는 기능 메뉴들을 나타냅니다.

Level 1	Level 2	Level 3	Description
Fix Value			사용자 입력 값으로 해당 Cell을 변경할 수 있는 기능을 제공함
Find Column			Column을 찾을 수 있고 해당 Column을 맨 앞으로 이동시킬 수 있는 기능을 제공(Column Header의 기능과 동일)
Fixed Column	Left	All Left By Column	선택된 Column을 포함한 왼의 모든 컬럼을 그리드의 왼쪽에 고정하는 기능을 제공함
		Current Column Only	선택된 Column만 그리드의 왼쪽에 고정하는 기능을 제공함
	Right	All Right By Column	선택된 Column을 포함한 오른의 모든 컬럼을 그리드의 오른에 고정하는 기능을 제공함
		Current Column Only	선택된 Column만 그리드의 오른에 고정하는 기능을 제공함
Unbound Column		Add Unbound Column	새로운 Column을 추가할 수 있으며, 기존 Column의 데이터를 조합하여 데이터로 사용할 수 있는 기능을 제공(Column Header의 기능과 동일)
		Column Copy	사용자가 선택한 Column과 데이터가 동일한 Column을 생성할 수 있는 기능을 제공(Column Header의 기능과 동일)
Width			해당 Cell이 존재하는 Column의 Cell Width를 변경할 수 있는 기능을 제공(Column Header의 기능과 동일)
Display Format String			해당 Cell 존재하는 Column의 Format Type과 Format String을 사용자 설정값으로 변경할 수 있는 기능을 제공(Column Header의 기능과 동일)
Layout			사용자가 지정한 Layout을 저장/불러오기/삭제하는 기능을 제공(Column Header의 기능과 동일)
Sum (Calculated)		Sum	Drag한 Cell Data들의 합계를 제공
		Min	Drag한 Cell Data들 중 최솟값을 제공
		Max	Drag한 Cell Data들 중 최댓값을 제공
		Count	Drag한 Cell Data들의 개수를 제공
		Average	Drag한 Cell Data들의 평균을 제공
		Range(Max-Min)	Drag한 Cell Data들의 최댓값과 최솟값의 차이를 제공
Excel			해당 Cell이 포함된 Table을 Excel로 저장 또는 Excel 정보를 가져오기 할 수 있는 기능을 제공(Column Header의 기능과 동일)

Fix Value(Selected Cells)

Fix Value는 선택된 해당 Cell의 데이터를 수정할 수 있는 기능을 제공합니다.

[그림 53.] 같이 변경을 원하는 Cell을 우클릭한 후 Fix Value 기능을 선택하면 Value를 입력할 수 있는 Text Box가 나타납니다. Text Box에 사용자가 원하는 Value를 입력하고 확인 버튼을 클릭하면 Value가 변경됩니다.

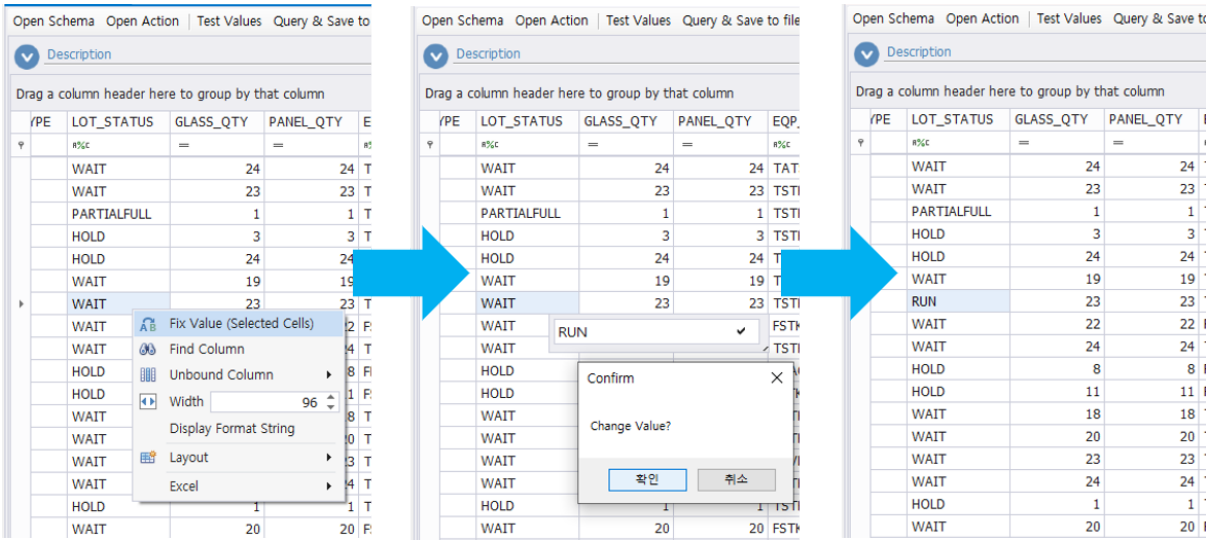


그림 53. Fix Value 사용 예시

Sum(Calculate)

해당 기능은 선택된 Cell 또는 Drag한 Cells에 대한 Calculation Value (Sum, Min, Max, Count and Average)를 나타내는 기능을 제공합니다. 해당 기능의 Default는 Sum이며, Calculation Value는 Column마다 다르게 설정이 가능하며 아래와 같이 해당 부분을 우클릭하여 변경할 수 있습니다.

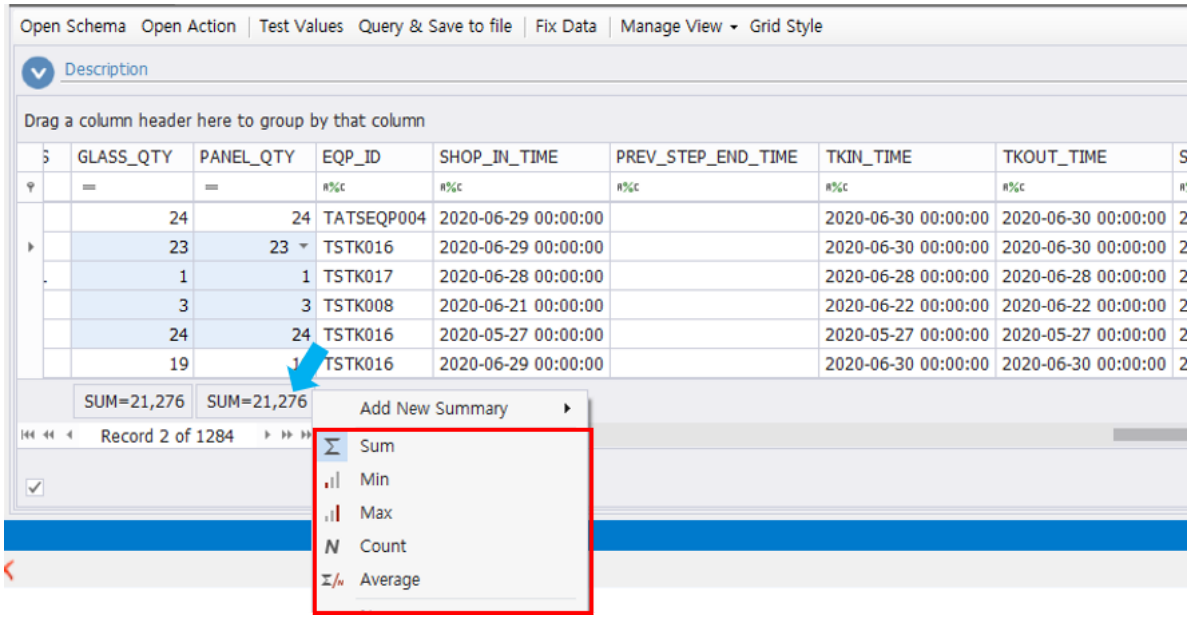


그림 54. Column의 Default Calculation Value 변경 방법

사용자는 Cell 또는 Cells을 클릭하거나 드래그하고 우클릭할 경우 Calculation Value에 해당하는 결과 값이 표현됩니다. [그림 55.] Calculation Value가 Sum, Average일 경우의 사용 예시입니다. 요약된 정보의 메뉴를 실행하면 Calculation Value 의 정보를 한눈에 볼 수 있는 **Summary Control** [그림 56.]이 나타나게 되고, 각 값들을 확인하고 복사하여 사용할 수 있습니다. Summary Control에는 Sum, Min, Max, Count, Average, Range(Max-Min) 정보를 제공합니다.

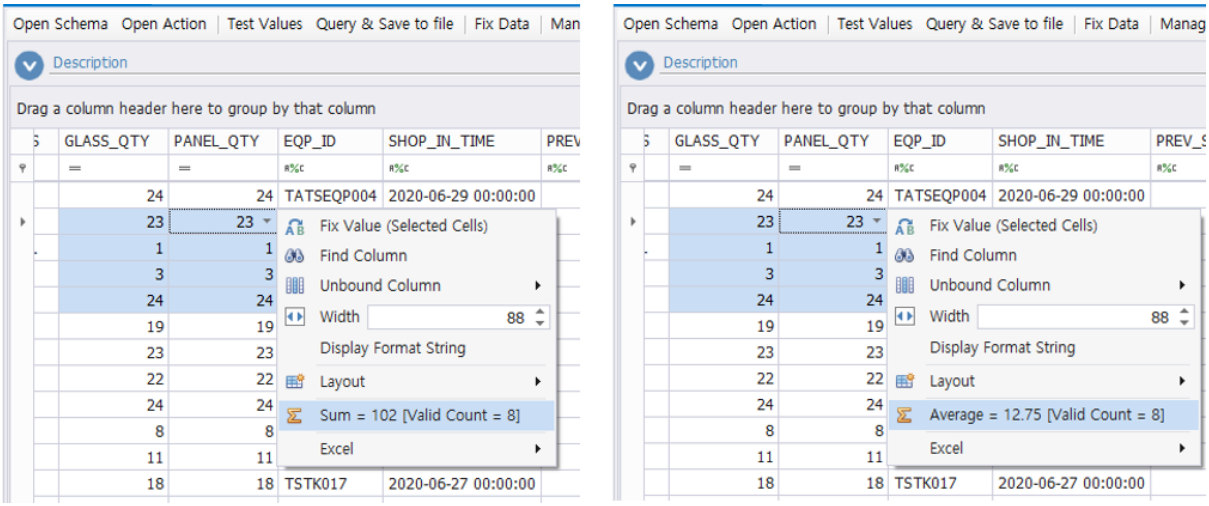


그림 55. Calculation Value가 Sum, Average일 경우의 사용 예시

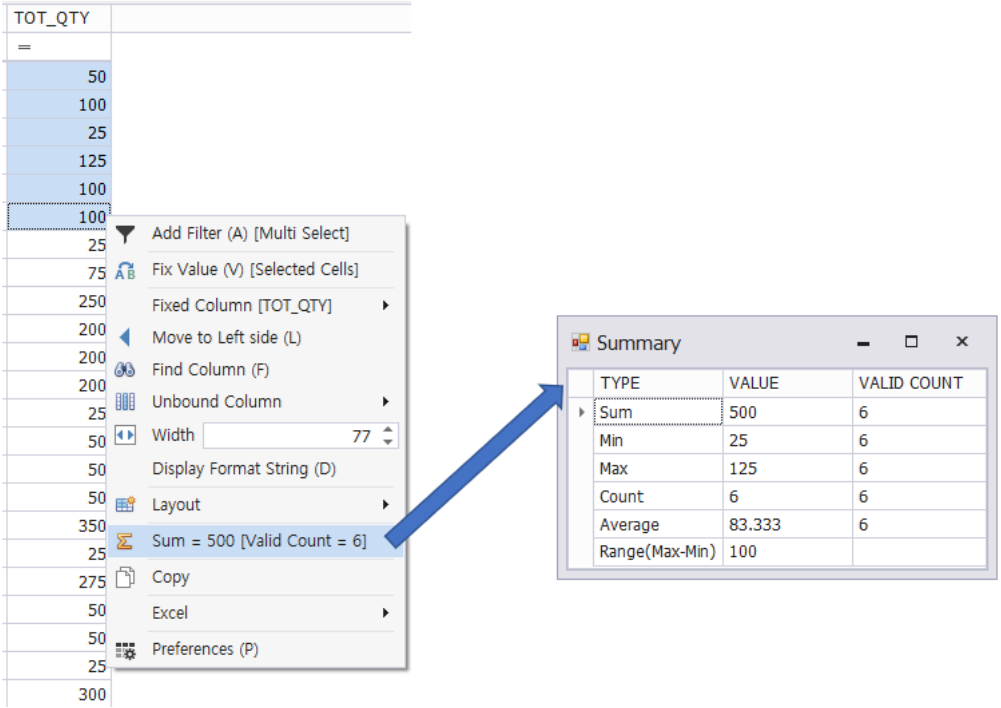


그림 56. Summary Control 사용 예시

Open Item(Read Only)

[Open Item(Read Only)] 메뉴는 **[Project]** 하위에 추가된 메뉴로써 모델 내 Inputs 또는 Outputs의 Data Item을 읽기 전용으로 보는 기능입니다.

Open Item 기능 목록

Menu Level 1	Menu Level 2	Description
Open Item	Open Item(Read Only-Grid)	Grid View로 해당 Data Item을 확인할 수 있는 기능을 제공
	Open Item(Read Only-Pivot)	Pivot Grid View로 해당 Data Item을 확인할 수 있는 기능을 제공

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/82f62502-8ac5-4cac-8ac9-fd90b410b475/MOZART_STUDIO_(_KOR_).pdf

Open Item

Open Item은 Main View에 해당 Data Item을 Grid 또는 Pivot으로 읽기 전용(Read Only) 창을 띄워 확인할 수 있는 기능을 제공합니다. 이 기능은 사용자가 Inputs/Outputs Data Item에 대한 분석을 할 때 사용될 수 있으며 창을 계속 추가하여 확인이 가능합니다.

Open Item(Read Only-Grid)

Main View에서 해당 Data Item을 Grid로 확인할 수 있는 기능을 제공합니다.

[그림 1.]과 같이 사용자가 원하는 Data Item을 우클릭 한 후 Open item(Read Only-Grid) 클릭하면 Main View에서 해당 Data Item을 Grid 형식으로 확인할 수 있습니다.

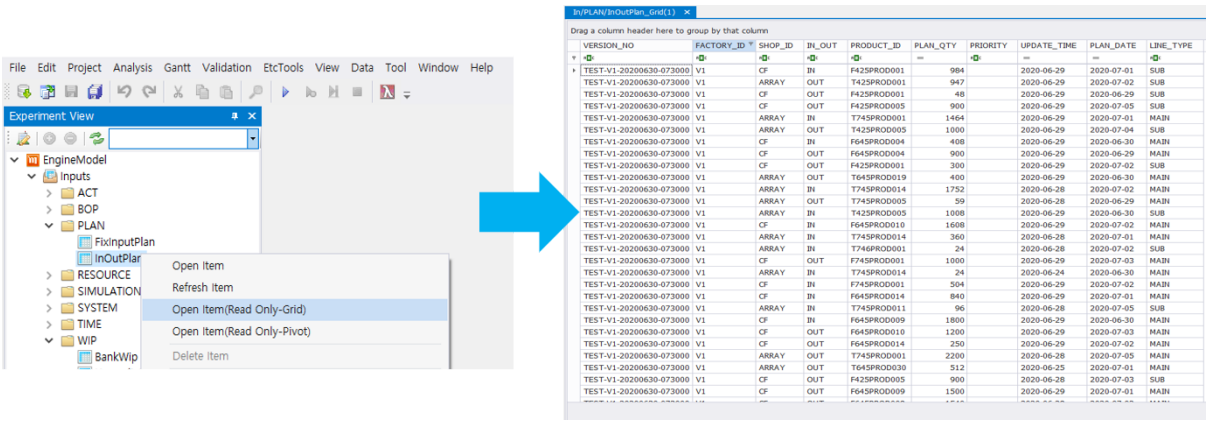


그림 1. Open Item(Read Only-Grid) 사용 예시

Open Item(Read Only-Pivot)

Main view에서 해당 Data Item을 Pivot Grid로 확인할 수 있는 기능을 제공합니다.

[그림 2.]와 같이 사용자가 원하는 Data Item을 우클릭 한 후 Open item(Read Only-Pivot)을 클릭하면 Main View에서 해당 Data item을 Pivot Grid 형식으로 확인할 수 있습니다. Pivot Grid의 경우, 사용자가 Data의 Area를 직접 설정하여 Data를 확인할 수 있습니다.

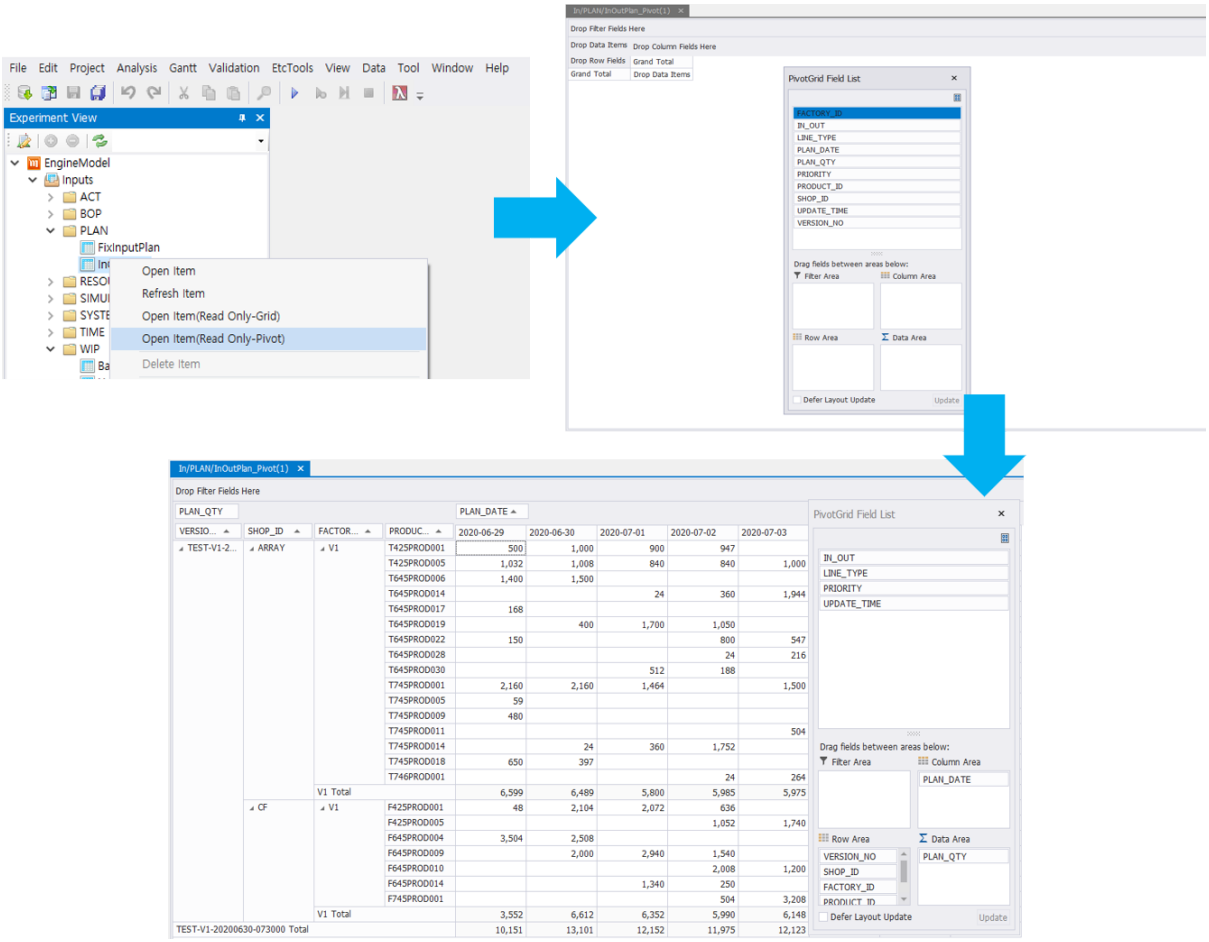


그림 2. Open Item(Read Only-Pivot) 사용 예시

Open Item 활용

동일한 DataItem에 대한 다수의 뷰 활용

기존에는 DataItem의 필터나 레이아웃을 적용하면서 데이터를 분석하였다면, 다수의 뷰를 생성하여 필터나 레이아웃을 다양하게 적용하여 동시에 참고하고 분석할 수 있습니다.

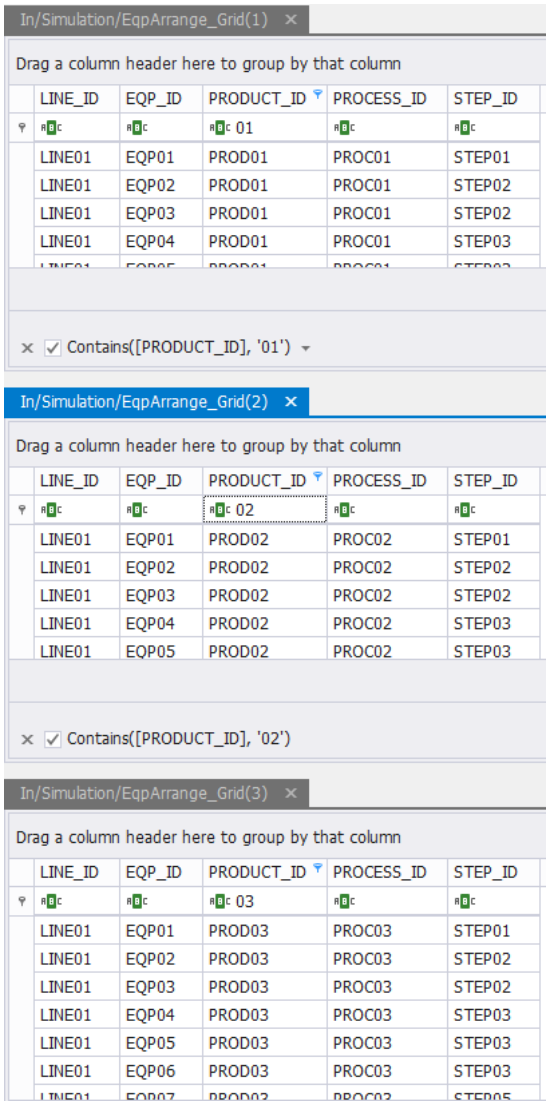


그림 3. 동일한 DataItem에 대해 다수의 뷰를 활용한 그림

Pivot Grid View 활용

Pivot Grid View를 지원함에 따라 다수의 뷰를 생성하여 Pivot Grid 분석을 할 수 있습니다. 따로 Pivot Grid UI를 개발하고 Pivot Grid View를 이용하여 분석할 수 있습니다.

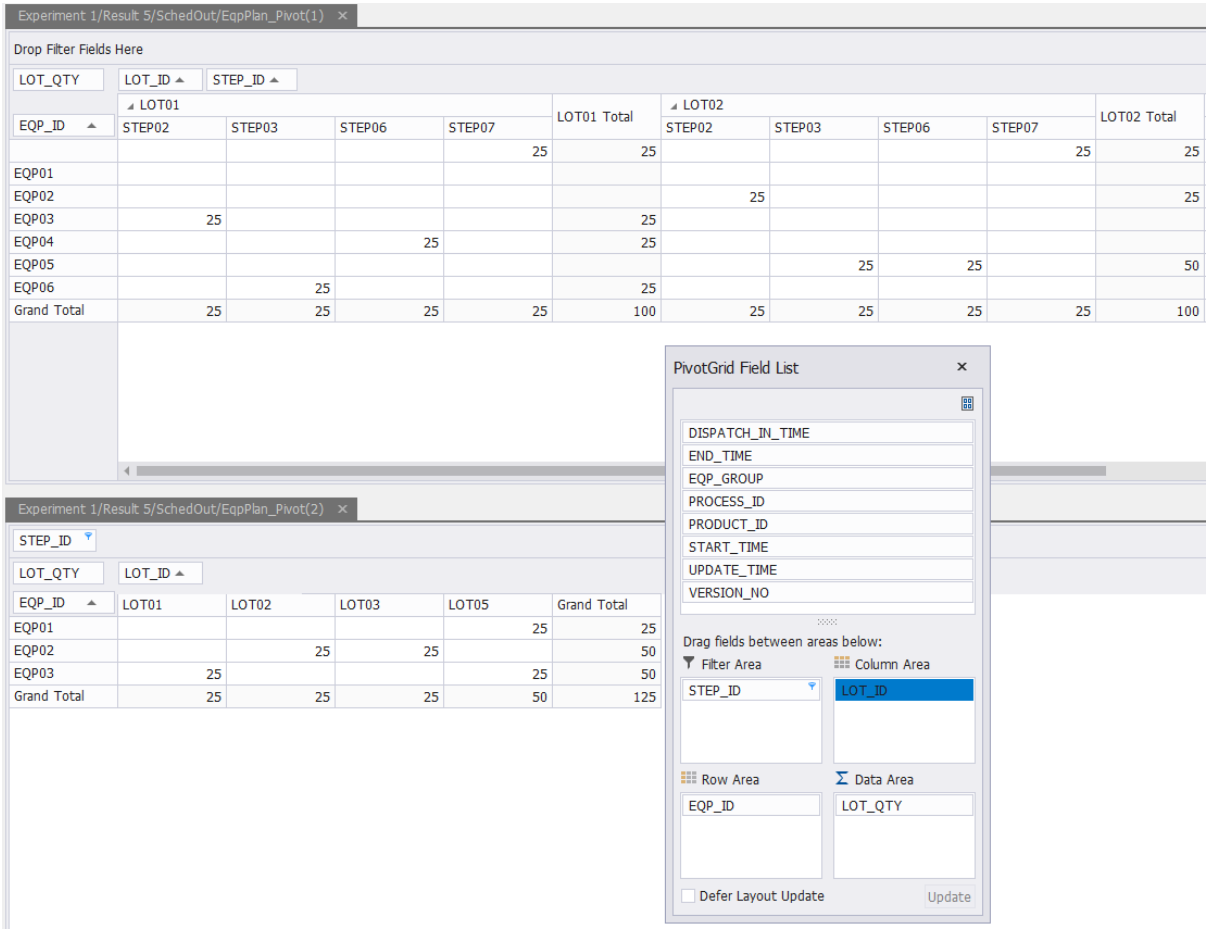


그림 4. Pivot Grid View를 이용한 데이터 분석 화면

사용자 뷰 정의 및 사용

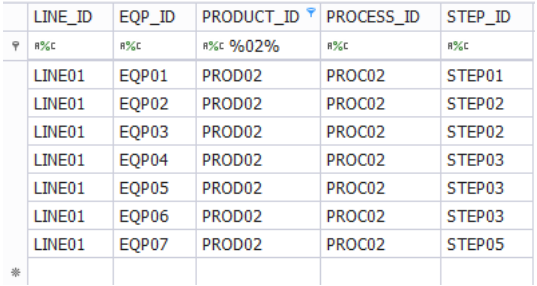
필터, 레이아웃 사용자 지정 및 공유 방법을 이용하여 사용자의 View를 생성 및 공유 할 수 있습니다.

필터, 레이아웃 사용자 지정 및 공유 방법

특정 DataItem의 사용자 Filter, Layout 지정

Filter 지정 방법

DataItem에서 Grid View의 Auto Filter Row에 사용자가 정의한 필터를 지정할 수 있습니다.



LINE_ID	EQP_ID	PRODUCT_ID	PROCESS_ID	STEP_ID
a%c	a%c	a%c %02%	a%c	a%c
LINE01	EQP01	PROD02	PROC02	STEP01
LINE01	EQP02	PROD02	PROC02	STEP02
LINE01	EQP03	PROD02	PROC02	STEP02
LINE01	EQP04	PROD02	PROC02	STEP03
LINE01	EQP05	PROD02	PROC02	STEP03
LINE01	EQP06	PROD02	PROC02	STEP03
LINE01	EQP07	PROD02	PROC02	STEP05

그림 1. Auto Filter Row에 필터를 추가한 그림

Auto Filter Row에서 Filter Condition에 대해 Filter 등록하고 사용자 PC에서 항상 동일하게 Filter가 반영되도록 저장할 수 있습니다. 데이터 조회 View 에서 Manage View의 Filter 메뉴에서 저장이 가능합니다.

Layout 지정 방법

Layout을 지정하는 방법은 3 가지가 있습니다. 첫 번째, 데이터 조회 View 에서 Manage View의 Layout 메뉴에서 저장이 가능합니다.

두 번째, Grid View, Pivot Grid View에서 Column Header에 우측 마우스로 Set Layout 메뉴를 이용하는 방법입니다. Grid View 사용법에서 **Column Header 보조 메뉴의 Layout 메뉴**를 참고하시기 바랍니다.

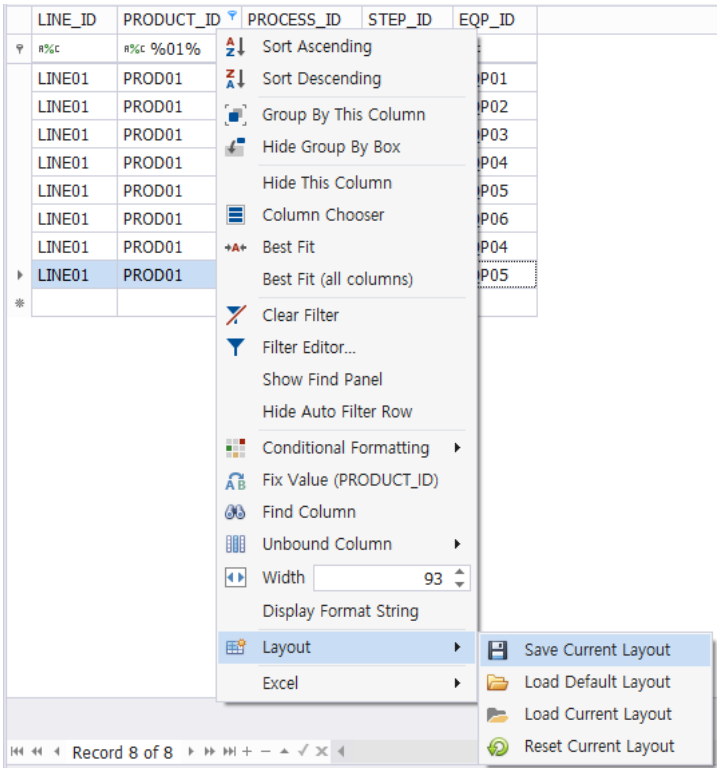


그림 4. Grid View에서 Column Header의 보조 메뉴 중 Layout 메뉴 화면

세 번째, Grid View, Pivot Grid View에서 Cell에 우측 마우스로 Set Layout 메뉴에서 Save Current Layout 을 눌러 저장할 수 있습니다. Grid View 사용법에서 **Cell 보조 메뉴의 Layout 메뉴**을 참고 하시기 바랍니다.

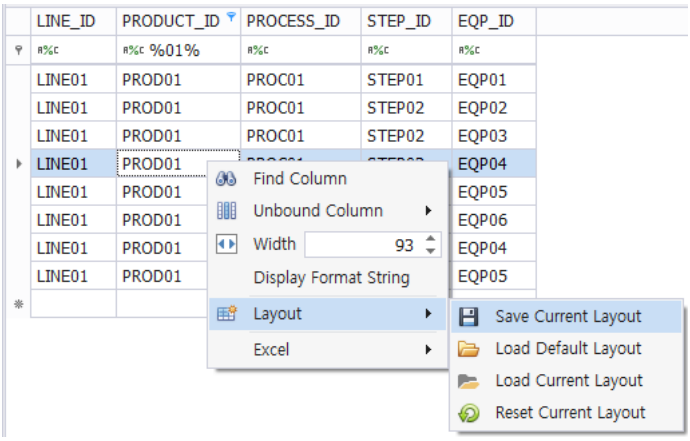


그림 5. Grid View에서 Cell의 보조 메뉴 중 Layout 메뉴 화면

사용자 정의 Filter, Layout 공유 방법

사용자 정의 정보 파일 저장 위치

사용자가 정의한 필터와 레이아웃 PC의 특정 위치에 저장이 됩니다. 필터는 vffs 확장자 형태로, 레이아웃은 xml 확장자 형태의 파일로 저장됩니다. 아래 위치에 저장되며, 저장 시 사용하는 Studio(LSE, APS, RTF, FP, CP, MP 등) 이름의 폴더 하위에 저장됩니다.

- **Filter** : *%appdata%\Mozart\(LSE Studio)\ViewInfo\Filter*
- **Layout** : *%appdata%\Mozart\(LSE Studio)\ViewInfo\Layout*

사용자 정의 파일 공유

필터와 레이아웃 정보가 저장된 파일을 다른 사용자의 PC에 복사하면 똑같은 필터와 레이아웃을 공유할 수 있습니다. 파일을 직접적으로 전달하거나, 오토 업데이트를 이용하여 다수의 사용자에게 제공할 수 있습니다.

사용자 정의 Layout 삭제/초기화 방법

Grid View에서 Manage View에서 Clear Current Condition(Filter), Load Default Layout(Layout)을 통해 필터와 레이아웃을 삭제/초기화 할 수 있습니다. 또한 저장 위치에 있는 파일을 삭제하면 필터와 레이아웃이 삭제 됩니다.

Data Join 사용법

Data Join 기능은 MOZART 2021.119.2 버전 이상에서 사용 가능합니다. 본 기능을 통해 사용자는 MOZART 모델의 데이터 아이템 간 조인 작업을 수행하고 결과 뷰를 저장하여 사용할 수 있습니다. Data Join 은 데이터베이스에서 지원하는 **조인 방식(Inner, Left, Right, Full)**을 모두 지원합니다. 본 기능은 MOZART 2020.118.0 버전에 추가된 Open Item (Read Only)기능에서 사용이 가능하며, 본 문서는 Grid View 메뉴를 기반으로 작성되었습니다.

> 💡 **Note**
> 1. 테이블의 그리드 뷰, 피봇 뷰에서 각각 하나의 Data Join 규칙을 가집니다.
> 2. 2개 이상의 테이블을 Join 하려면 먼저 두 테이블을 Join하는 규칙을 만들고 그 이후 **[Add Rule]**를 통해 Join 할 테이블을 추가 할 수 있습니다.
> 3. Data Join 규칙을 적용하면 기존 Layout을 덮어씁니다.

Data Join에 대한 자세한 사항은 여기에서 확인 바랍니다.

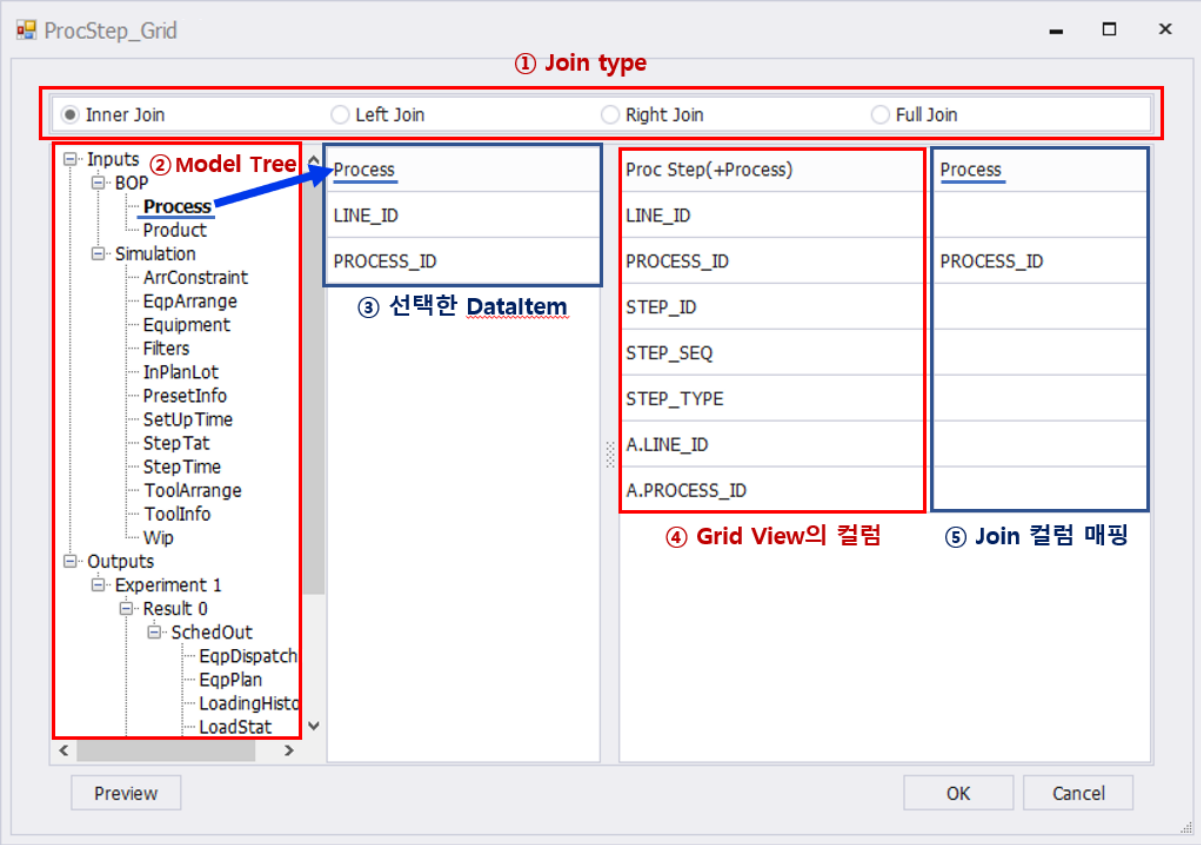
메뉴 구성

Data Join의 메뉴는 **[Add Rule], [Manage Rules], [Reset Rules]**으로 구성되어 있습니다.

> 💡 **Note**
> Join 조건이 없을 시 **[Manage Rules], [Reset Rules]** 메뉴는 비활성화 되어있습니다.

Add Rule

스키마 뷰에 하나의 Join을 추가하는 메뉴 입니다.

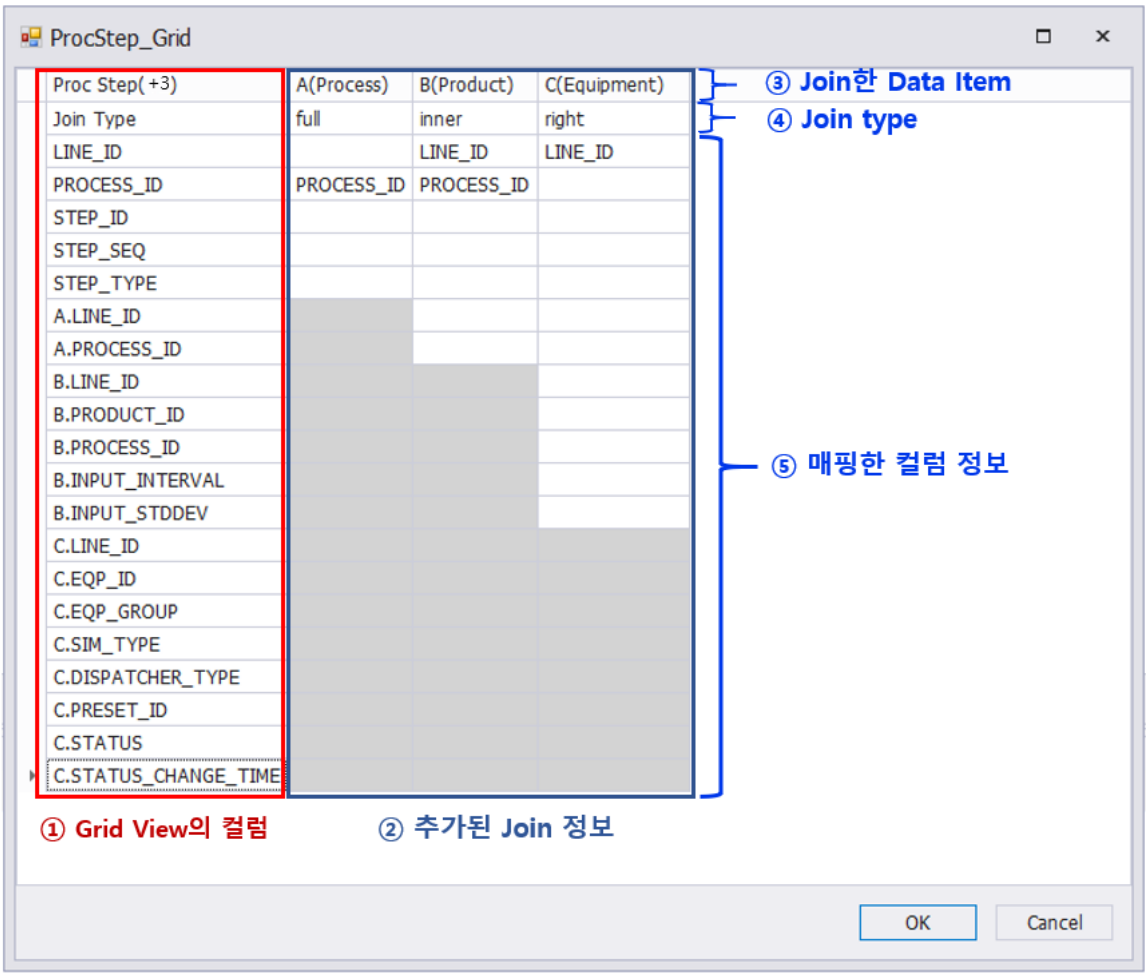


Add Rule 설정창

- ① **Join type :** 추가할 **Join type을 선택**하는 영역 입니다.
- ② **Model Tree :** 해당 스키마 뷰의 데이터 아이템을 제외한 모델의 데이터 아이템의 구성입니다. 해당 영역에서 스키마 뷰의 데이터와 **Join할 데이터 아이템을 선택**합니다.
- ③ **선택한 Data Item** : ②에서 선택한 데이터 아이템의 모든 컬럼을 나타냅니다. 해당 컬럼으로 Join 매핑을 설정합니다.
- ④ **Grid View의 컬럼** : 스키마 뷰의 모든 컬럼을 나타냅니다.
  - ④의 상단은 스키마 뷰의 데이터 아이템 명과 추가로 Join된 정보입니다.
    ex) Base_DataItem(+Join DataItem), Base_DataItem(+Join한 수)
  - "A. ", "B. ", "C. " 형식의 컬럼은 Join으로 추가된 데이터 아이템 별 컬럼 입니다.
- ⑤ **Join 컬럼 매핑** : ③의 컬럼과 뷰 컬럼의 **Join 매핑을 설정**하는 영역입니다.
- **[Preview] :** 설정한 Data Join의 결과를 나타냅니다.

Manage Rules

스키마 뷰에서 추가한 Join의 확인 및 수정하는 메뉴 입니다.



Manage Rules 설정창

Reset Rules

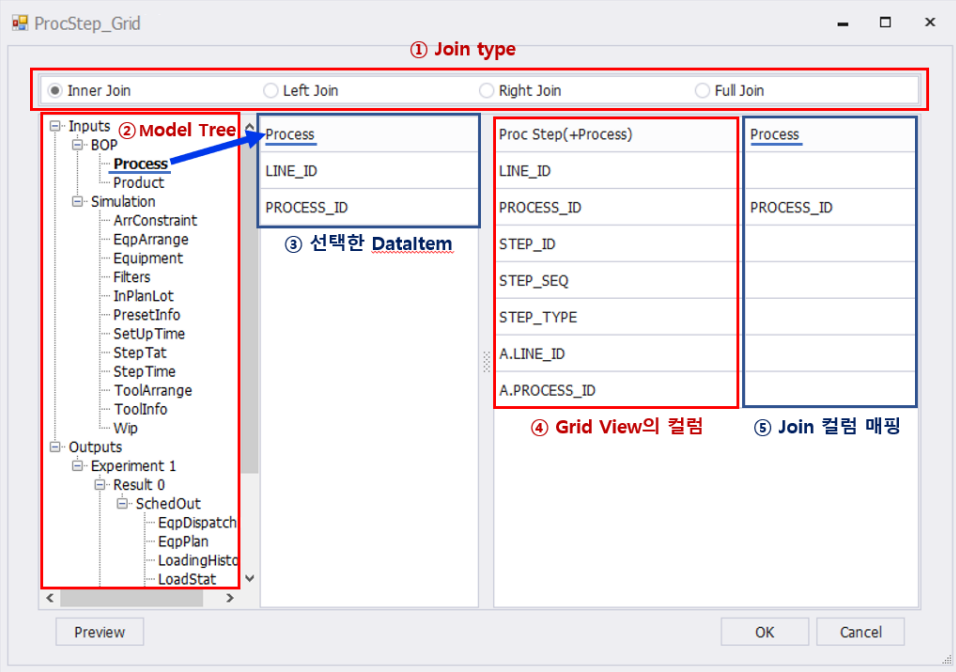
스키마 뷰에서 추가한 Join Rules를 모두 삭제하는 초기화 메뉴 입니다.

Data Join 추가 및 수정

아래는 스키마 뷰에서 **[Add Rule]** , **[Manage Rules]** 메뉴를 통해 Data Join을 추가하고 확인 및 수정, 삭제하는 방법을 설명합니다.

Data Join 추가

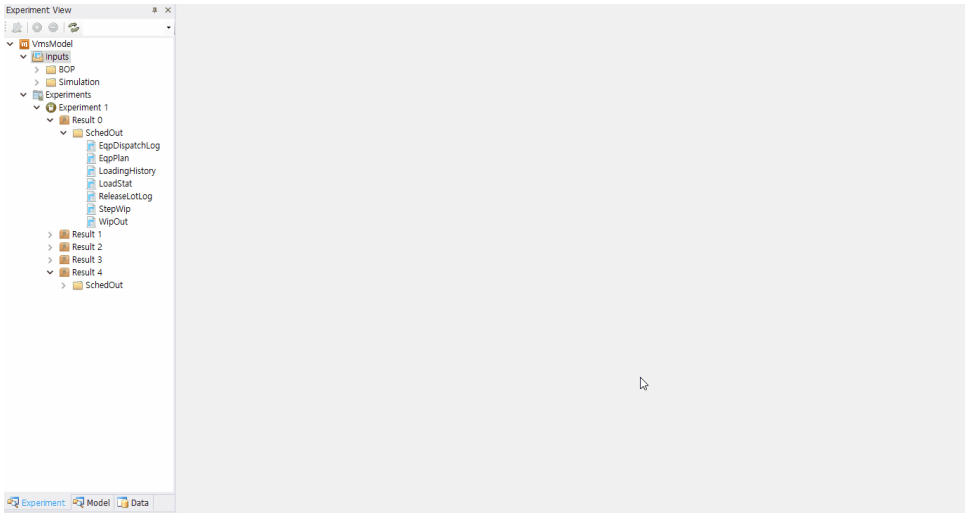
뷰 스키마에서 **[Data Join]** > **[Add Rule]** 메뉴를 실행하여 Join Rule을 추가할 수 있습니다.



Add Rule 설정창 - 영역 확인

1. 데이터 아이템 뷰에서 **[Add Rule]** 메뉴를 실행하여 Add Rule 설정창을 띄웁니다.
2. 뷰의 데이터와 Join할 데이터 아이템을 ②영역에서 선택하고 ①에서 Join Type을 선택합니다.
3. Join할 데이터 아이템의 컬럼을 뷰의 데이터와 매핑합니다.
   - ③에 있는 컬럼을 ⑤영역으로 드래그 합니다.
   - ③에 있는 컬럼을 더블 클릭하면 ⑤의 동일한 이름의 컬럼에 매핑이 됩니다.
   - ⑤에서 매핑된 컬럼을 우클릭, 더블클릭 하면 매핑이 사라집니다.
4. 우측 하단의 [OK]를 클릭하고 뷰에서 출력할 컬럼을 선택합니다.
5. 뷰에서 Join Rule이 적용된 데이터가 출력됩니다.

Inner Join 추가 예제



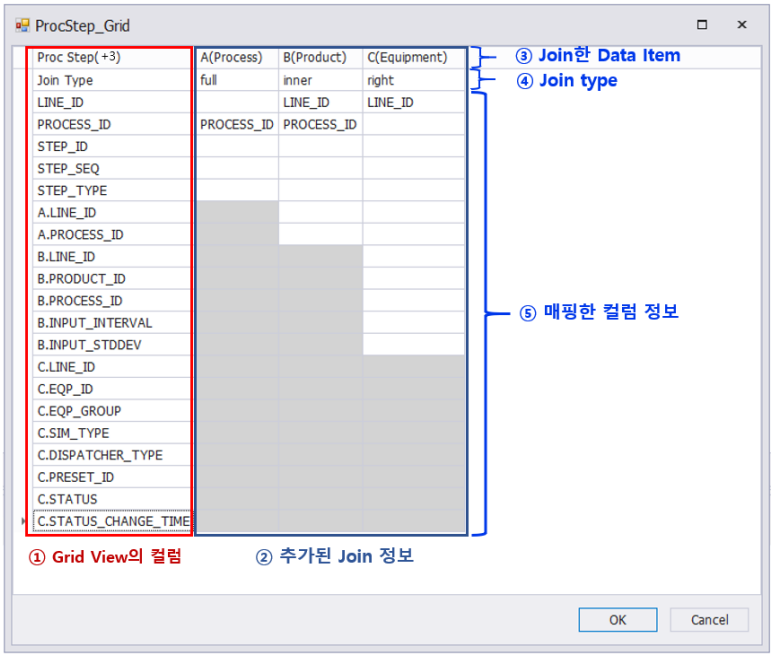
Inner Join 추가 예제

Data Join 확인, 수정, 삭제

스키마 뷰에 **[Data Join]** > **[Manage Rulse]** 메뉴를 통해 해당 뷰에서 추가한 Join Rules의 확인 및 수정, 삭제가 가능합니다.

확인

[Manage Rulse] 메뉴를 실행하여 Data Join Rules를 확인 할 수 있습니다.



Manage Rules 설정창 - 영역 참고

수정

Join type 변경

- **[Manage Rules]** 설정창의 ④영역에서 드롭박스를 통해 해당 데이터 아이템의 **Join type**을 변경합니다.

Join 매핑 컬럼 변경

- ⑤영역에서 드롭박스를 통해 **매핑 컬럼**을 추가하거나 다른 컬럼으로 변경합니다.
- ⑤영역에서 **매핑한 컬럼** 데이터를 지워서 해당 컬럼의 Join 매핑을 취소합니다.

삭제

[Manage Rules] 설정창에서 **Join type** 또는 **Join 매핑한 컬럼**을 빈 데이터로 설정하여 **Join**을 삭제합니다.

- ④영역에서 삭제하려는 Join 데이터 아이템의 **Join type**을 드롭박스에서 빈 데이터로 선택합니다.
- ⑤영역에서 삭제하려는 Join 데이터 아이템의 **모든 매핑 컬럼**을 지웁니다.

Data Join 초기화

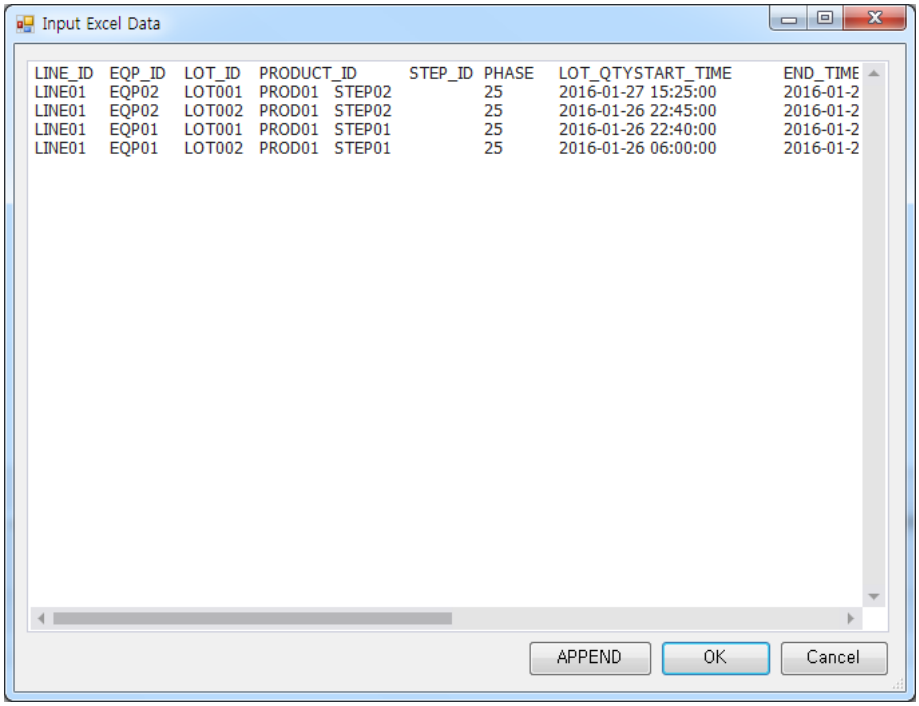
스키마 뷰에서 **[Data Join]** > **[Reset Rules]** 메뉴를 실행하여, 해당 뷰에 추가한 Join을 모두 초기화 합니다.

데이터 Export/Import

Import Data

Input Data 의 경우 기본적으로 실행 시작 시점에 외부 데이터 소스(DB)로부터 쿼리를 통해 다운로드하여 생성됨을 가정합니다. 그러나 테스트 목적으로 특정 시점의 Input Data 를 사용하거나 엑셀, Text 로부터 Input Data 를 만드는 경우에 Import 기능을 사용합니다.

- Excel : 엑셀 데이터를 복사하여 입력창에 붙여 넣은 후 **[Append][OK]** 버튼으로 데이터를 Import 합니다. **[Append]** 하는 경우 기존 데이터를 유지한 채로 데이터를 추가합니다. **[OK]** 버튼을 사용하는 경우 기존 데이터를 지운 후 추가 대상 데이터를 입력합니다. 아래 그림과 같이 첫 행은 컬럼명을 표시하고 두 번째 행부터 데이터를 입력합니다. 컬럼은 Tab 으로 구분합니다. 엑셀 데이터를 복사하여 붙여넣기 하는 방식으로 사용할 수 있습니다.



- Text file : Excel Import 기능과 유사하나 입력 파일을 지정하는 방식이 차이가 있습니다. **[Import/Text file]** 메뉴를 선택하면 입력 대상이 되는 Text 파일을 선택하도록 파일 선택 창이 활성화 됩니다. 상기 Excel 입력창에 입력한 형식과 동일한 형식으로 저장된 .txt 파일을 선택하면 기존 Input 데이터를 삭제한 후 선택한 파일의 데이터로 Replace 하는 방식으로 데이터를 Import 할 수 있습니다.

Export Data

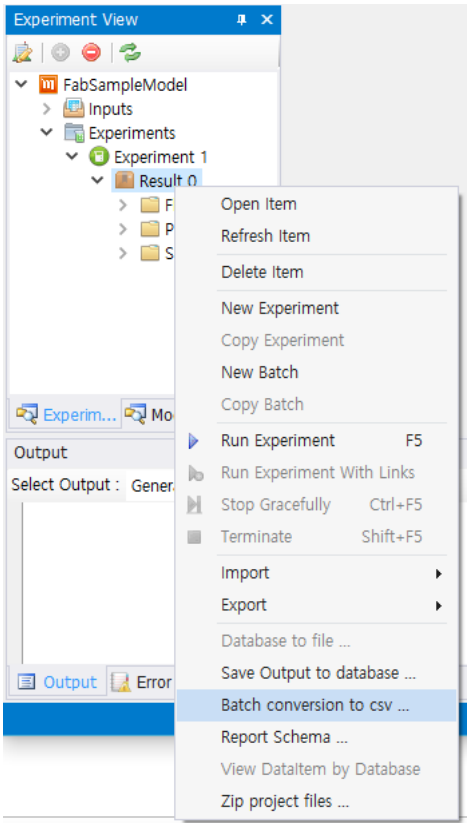
그리드의 데이터를 데이터 파일 형태로 출력하는 기능입니다. 지원되는 포맷은 아래 메뉴 이미지에서와 같이 Excel, Text file, HTML, XML, Rtf, Pdf, Mht 파일포맷을 지원합니다.

Batch Conversion to CSV

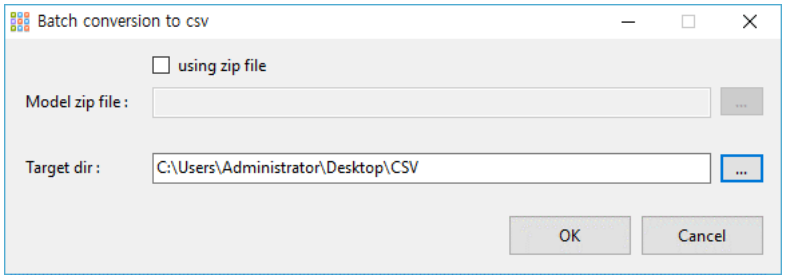
Import/Export Data의 경우는 선택한 DataItem에 한해서 추출이 가능합니다. 모델 전체의 Input/Output 데이터를 일괄적으로 Csv 파일 포맷으로 추출이 가능합니다. 해당 기능 사용을 위해서는 MOZART Studio 사용자 권한이 [Power] 이상이어야 하며 다음과 같이 두 가지 방법으로 추출이 가능합니다.

MOZART Studio에서 열려있는 모델에서 추출하는 방법

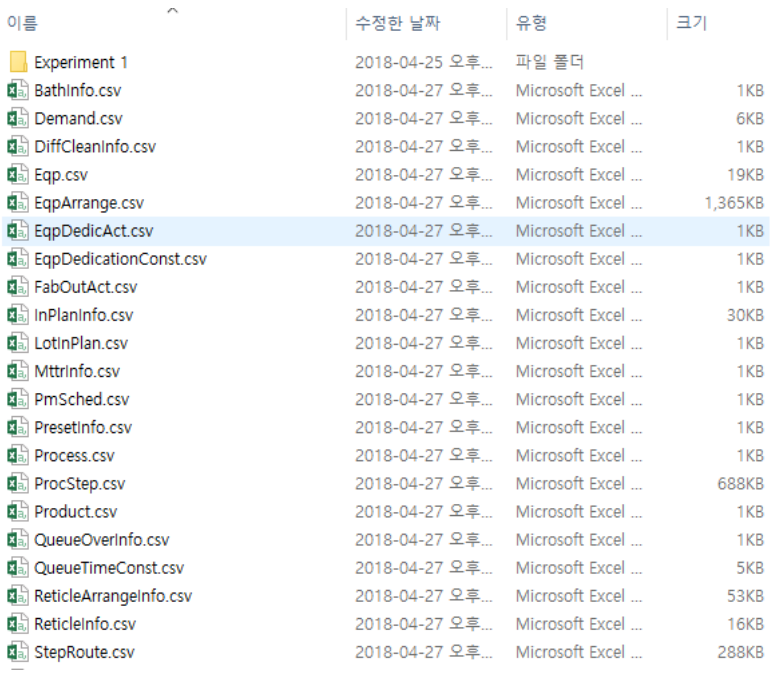
1. 모델 탐색기에서 마우스 우클릭, **[Batch conversion to csv...]**를 선택합니다.



2. Batch conversion to csv Dialog에서 **[Target dir :]**에 csv 파일들을 저장 할 위치를 지정하고 **[OK]** 버튼을 클릭합니다.



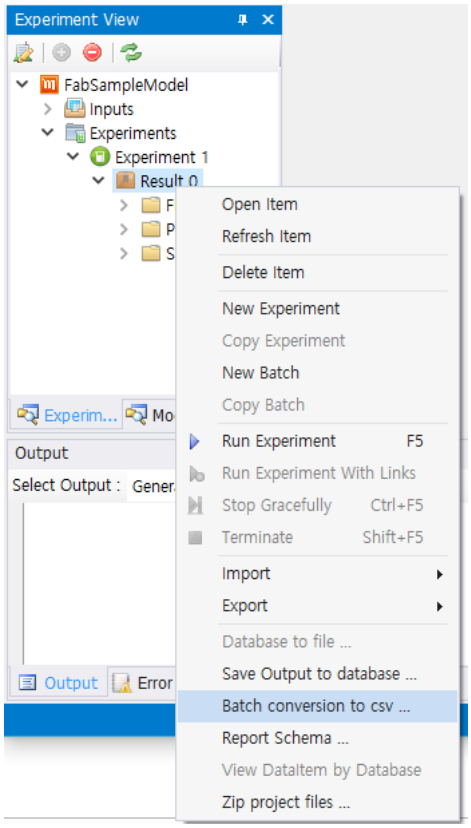
3. 성공적으로 추출이 되었으면 아래와 같이 저장이 됩니다.



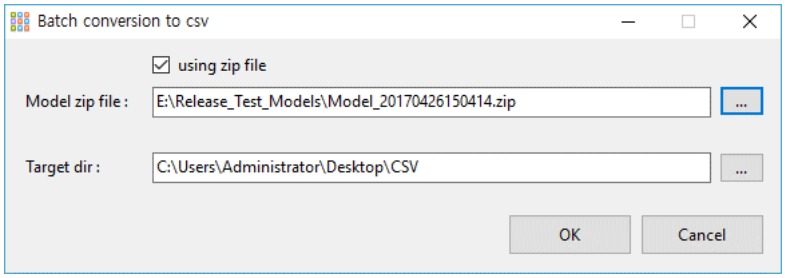
이름	수정한 날짜	유형	크기
Experiment 1	2018-04-25 오후...	파일 폴더	
BathInfo.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
Demand.csv	2018-04-27 오후...	Microsoft Excel ...	6KB
DiffCleanInfo.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
Eqp.csv	2018-04-27 오후...	Microsoft Excel ...	19KB
EqpArrange.csv	2018-04-27 오후...	Microsoft Excel ...	1,365KB
EqpDedicAct.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
EqpDedicationConst.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
FabOutAct.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
InPlanInfo.csv	2018-04-27 오후...	Microsoft Excel ...	30KB
LotInPlan.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
MttrInfo.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
PmSched.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
PresetInfo.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
Process.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
ProcStep.csv	2018-04-27 오후...	Microsoft Excel ...	688KB
Product.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
QueueOverInfo.csv	2018-04-27 오후...	Microsoft Excel ...	1KB
QueueTimeConst.csv	2018-04-27 오후...	Microsoft Excel ...	5KB
ReticleArrangeInfo.csv	2018-04-27 오후...	Microsoft Excel ...	53KB
ReticleInfo.csv	2018-04-27 오후...	Microsoft Excel ...	16KB
StepRoute.csv	2018-04-27 오후...	Microsoft Excel ...	288KB

Model 압축 파일에서 추출하는 방법

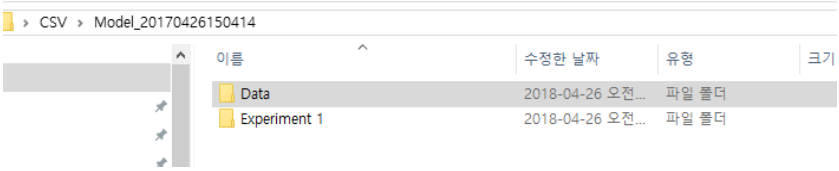
1. 모델 탐색기에서 마우스 우클릭, **[Batch conversion to csv..]**를 선택합니다.



2. Batch conversion to csv Dialog에서 **[using zip file]** 체크박스에 체크를 하고 **[Model zip file :]**에 Data를 추출할 Model 압축 파일을 **[Target Dir :]**에 CSV 파일들을 저장 할 위치를 선택합니다.



3. Model.zip에서 추출한 경우 아래와 같이 Data/Experiment 별로 저장이 됩니다.

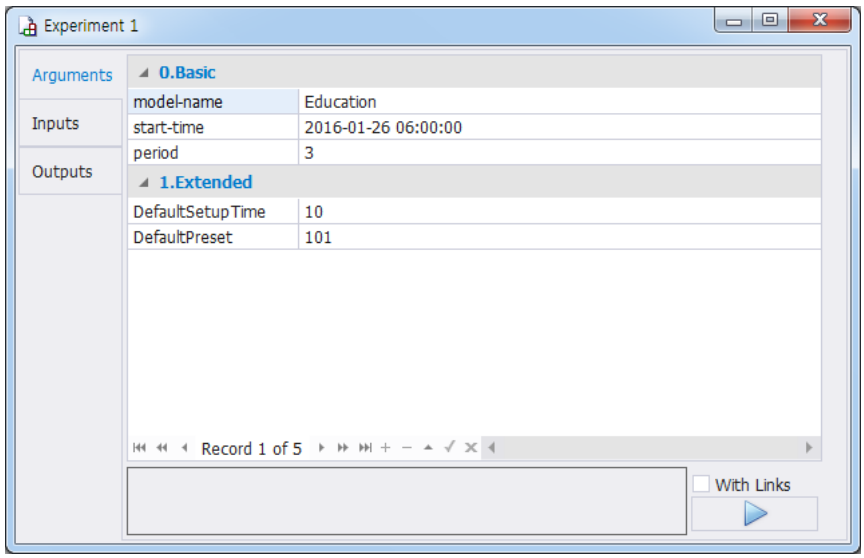


프로젝트 실행

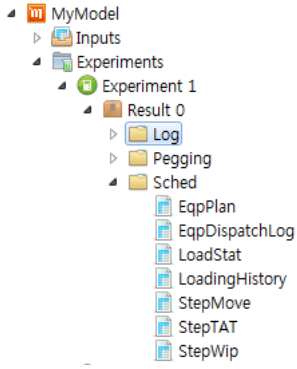
Model 실행을 위해서는 Experiment 를 정의하거나 Batch Experiment 를 먼저 생성해야 합니다. Experiment 를 정의하는 것은 실행 조건 (Input Argument) 을 정의하는 과정입니다. 설정된 실행 조건에 따라 Model 을 실행 할 수 있으며, 실행시마다 결과가 생성됨으로 Experiment : Result = 1 : N 의 구조를 갖습니다.

Experiment

실행 조건의 집합입니다. 한 모델에 여러 개의 **Experiment** 를 생성할 수 있습니다. Experiments 노드에서 오른쪽 마우스 버튼을 눌러 **[New Experiment]**메뉴를 사용합니다. Experiment 가 생성되면 상단 툴바의 실행 버튼 혹은 **"F5"** Key, Experiment 창의 실행 버튼을 사용하여 모델을 실행시킬 수 있습니다. Experiment 의 Arguments 창에서 설정한 Argument 에 따라 모델을 실행하고 결과가 한 개 생성됩니다.

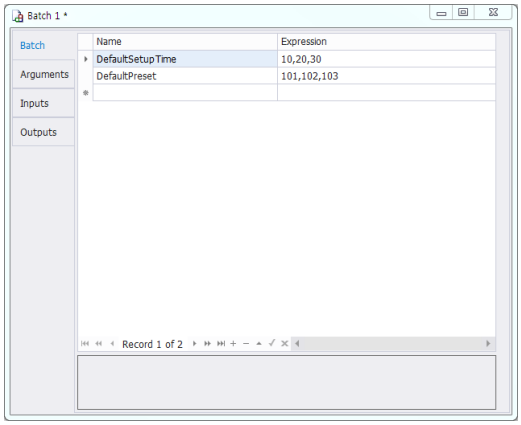


생성된 결과는 Experiment 노드 하위에 **[Result]** 의 형식으로 추가됩니다. 결과명은 사용자 Popup 메뉴를 사용하여 변경할 수 있습니다.

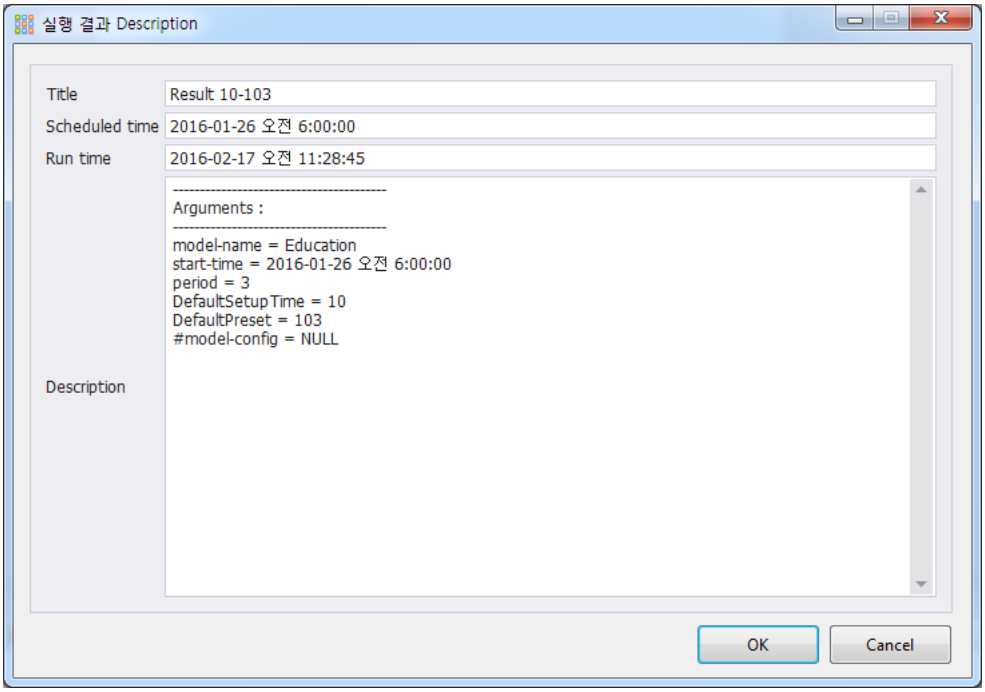


Batch

Experiment 의 경우 한번 실행 시 설정된 Argument Set 에 대해 하나의 결과를 생성합니다. 그러나 Batch Experiment 의 경우 설정된 Input Argument 의 조합에 따라 실행되며, 결과를 생성합니다. Batch Experiment 를 설정하기 위해서는 **Experiments** 노드에서 오른쪽 마우스 버튼을 눌러 **[New Batch]**메뉴를 사용합니다. Batch 설정 창은 기본적인 Argument 설정창과 함께 Batch Argument 를 설정할 수 있는 탭이 제공됩니다. Batch Tab 에서 Input Argument 중 값을 조정해서 실행하기 위한 Argument 를 설정하고 해당 Argument 의 변경값을 입력하여 Batch 실행을 계획 할 수 있습니다. 아래 입력 예시입니다. DefaultSetupTime 을 10,20,30분으로 변경하고 DefaultPreset 값을 101,102,103 순으로 변경하면서 변경 대상인 두 개의 Argument 의 값 조합에 따라 Model을 실행하고 결과를 생성합니다. Argument 값에 대한 구분자는 Comma 를 사용하며 Argument 의 값 타입이 Boolean 인 경우에는 "true/false" 문자열로 값을 입력합니다.



위와 같이 입력하는 경우 해당 Batch 에 결과는 총 3*3 = 9개가 생성됩니다. 결과는 기본적으로 아래와 같이 Batch Argument 의 값 조합을 기준으로 생성됩니다. 각 Result 노드를 더블 클릭하거나 선택한 상태에서 Enter 를 입력할 경우 해당 결과의 수행 설정값을 확인 할 수 있습니다.



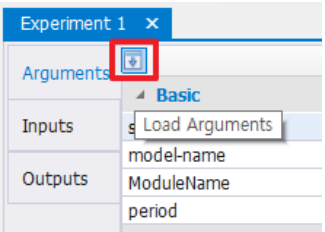
Load Arguments

Studio에서 Experiment를 실행하기 전 앞서 수행한 Result의 Argument 정보를 가져오는 기능입니다. Load Arguments는 다음의 2가지의 기능으로 사용할 수 있습니다.

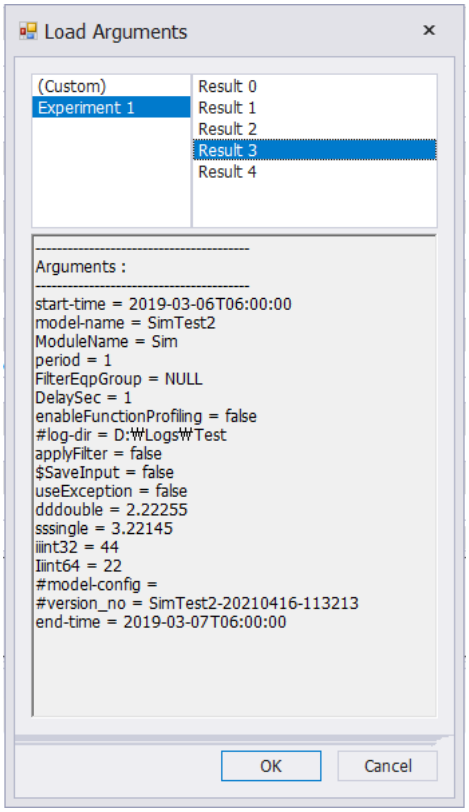
- 동일한 모델에서 이전에 수행한 Result의 Arguments 를 가져와서 사용하는 기능
- 다른 모델에서 수행한 Result의 Arguments 를 복사하여 가져와서 사용하는 기능

동일한 모델에서 사용하는 방법

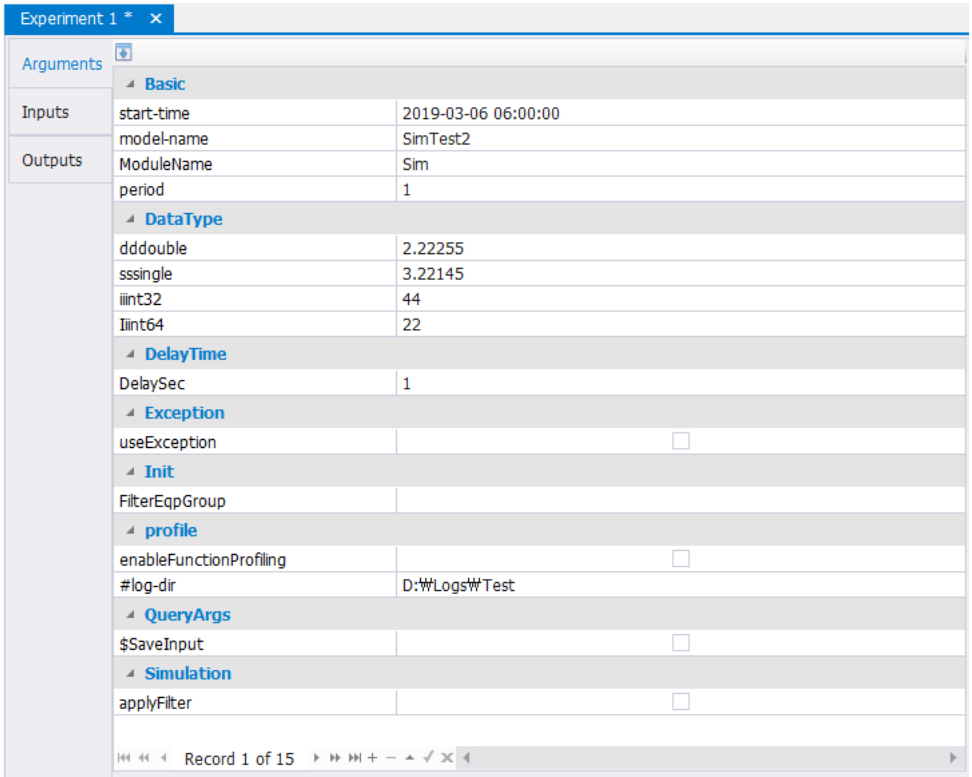
1. Experiment View의 Arguments 탭에서 좌측 상단의 Load Arguments 버튼을 클릭합니다.



2. 원하는 Experiment의 Result를 선택하여 **[OK]**를 누릅니다.

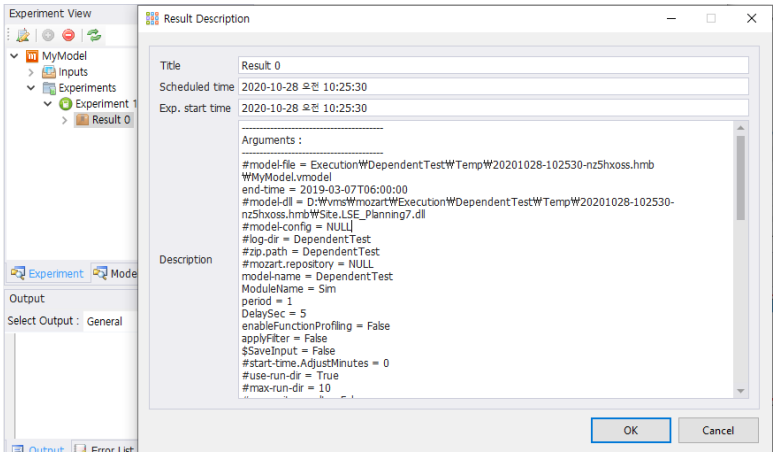


3. Result 선택 후 [OK]를 하면 해당 Result의 Arguments 설정값이 Experiment에 적용됩니다.

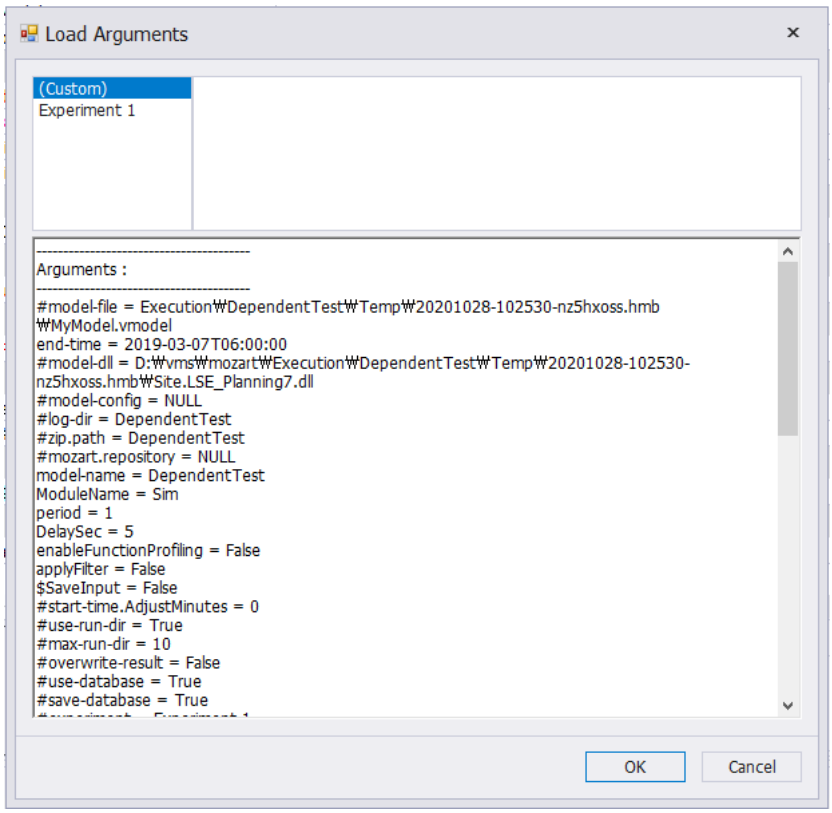


다른 모델에서 가져오는 방법

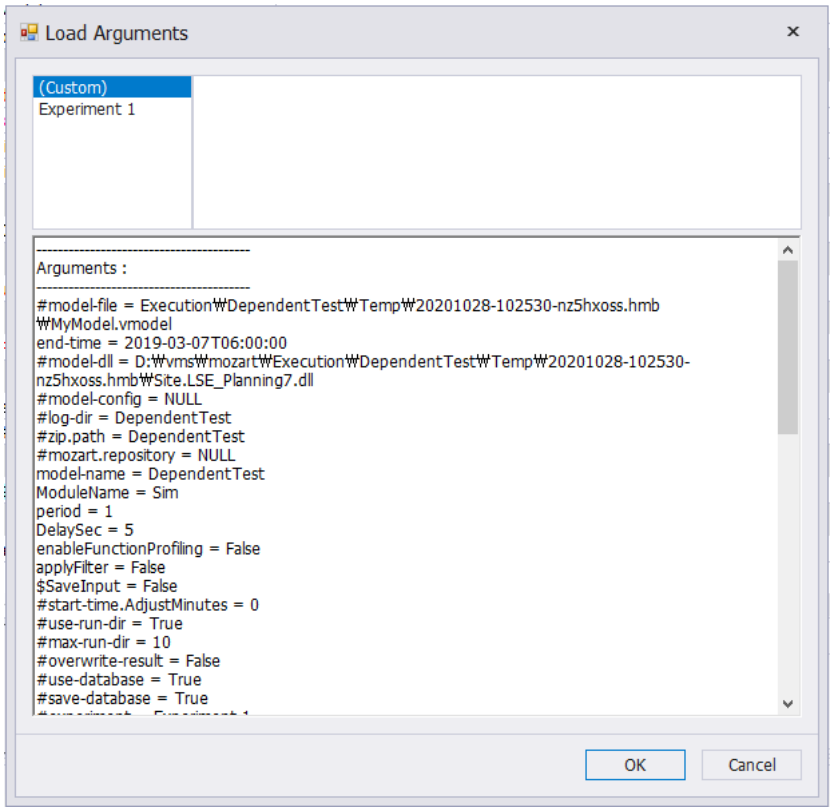
1. 다른 모델에서 수행된 Result 정보를 복사합니다. Result 를 엔터 또는 더블 클릭으로 Result Description 화면을 띄웁니다. Description 의 Argument 정보를 마우스로 드래그하여 복사합니다.



2. 복사한 Argument 정보를 반영하기 위해 Experiment 의 Load Arguments 를 실행합니다. **[(Custom)]** 메뉴를 선택하고 아래에 복사한 Arguments 값을 붙여넣기 합니다.



3. 일치하지 않는 Argument는 빨간색으로 표시 되며, 사용하고자 하는 값으로 변경하여 실험을 실행할 수 있습니다.



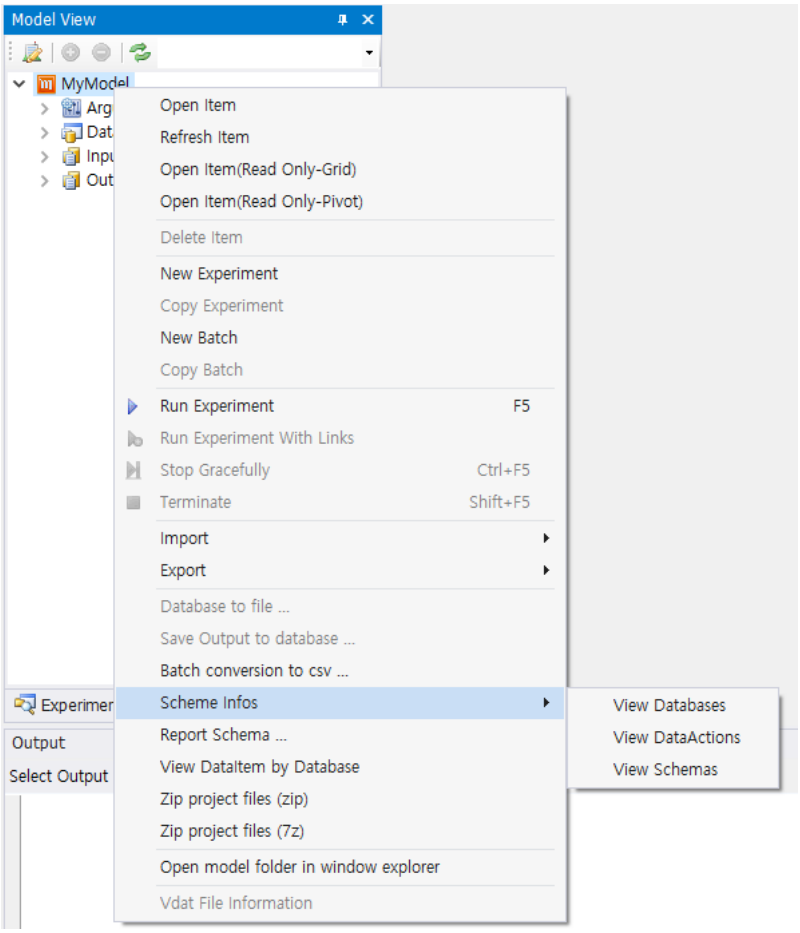
Note
Load Arguments 적용 후, Argument의 변수명이 빨간 글씨로 변경된 변수는 Load Arguments 기능을 통해 값이 반영되지 않은 변수입니다.
해당 변수 컬럼을 클릭하면 다시 기존 스타일로 변경 됩니다.

Model DataItem 정보 확인 및 출력 방법

Model에 등록된 DataItem과 DataSource 정보를 Studio를 통해 확인할 수 있습니다. **Schema Infos** 를 이용하여 확인하는 방법과 HTML 형태로 출력하는 Report Schema 기능을 이용하는 방법이 있습니다.

Schema Infos

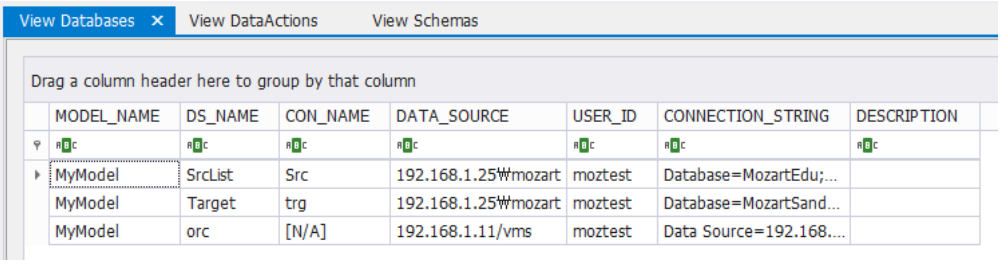
Studio에서 모델에 등록된 Databases, DataActions, Schemas 정보를 확인하는 기능입니다. Studio의 Model View 에서 **[Schema Infos]** 메뉴를 실행할 수 있습니다. Studio 사용 권한이 **Power**, **Admin** 인 경우에만 Grid로 출력하여 확인 할 수 있습니다.



View Databases

모델에 등록된 Data Source(Database) 정보를 확인 할 수 있습니다.

- MODEL_NAME : 모델 이름
- DS_NAME : Data Source 이름(Database 이름)
- CON_NAME : Data Source 내 연결 이름
- DATA_SOURCE : Data Source 연결 정보
- USER_ID : Data base 연결 계정(ID)
- CONNECTION_STRING : Data Source 연결 정보
- DESCRIPTION : 설명 정보



MODEL_NAME	DS_NAME	CON_NAME	DATA_SOURCE	USER_ID	CONNECTION_STRING	DESCRIPTION
MyModel	SrcList	Src	192.168.1.25₩mozart	moztest	Database=MozartEdu;...	
MyModel	Target	trg	192.168.1.25₩mozart	moztest	Database=MozartSand...	
MyModel	orc	[N/A]	192.168.1.11/vms	moztest	Data Source=192.168....	

View DataActions

모델에 등록된 DataAction 정보를 확인 할 수 있습니다.

- MODEL_NAME : 모델 이름

- CMD_TYPE : Command Type 정보
- CATEGORY : Schema Category 이름
- SCHEMA_NAME : Data Item 이름
- DATAACTION_NAME : DataAction 이름
- CMD_NAME : Data Source 내 Command 이름
- MAIN_DB : Data Source 내 Main DB 이름
- ALT_DS : MAIN_DB의 대체 DB 이름
- REAL_DS : 실제 연결되는 Data Source 이름
- CMD_TEXT : DataAction 내 Query 문

View Databases | View DataActions | View Schemas

Drag a column header here to group by that column

MODEL_NAME	CMD_TYPE	CATEGORY	SCHEMA_NAME	DATAACTION_NAME	CMD_NAME	MAIN_DB	ALT_DS	REAL_DS	CMD_TEXT
MyModel	Text	BOP	Process	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	BOP	Product	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	BOP	ProcStep	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	Simulation	Equipment	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	Simulation	Equipment	Default	Cmd2	SrcList		SrcList	--SELECT
MyModel	Text	Simulation	EqpArrange	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	Simulation	StepTime	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	Simulation	StepTat	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	Simulation	Wip	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	Simulation	PresetInfo	Default	Cmd1	SrcList		SrcList	SELECT
MyModel	Text	Simulation	PMSchedule	Default	Cmd1	SrcList		SrcList	
MyModel	Text	CONFIG	Config	Default	Cmd1	SrcList		SrcList	
MyModel	Text	LINQ	LinqInfo	Default	Cmd1	SrcList		SrcList	
MyModel	Text	ADAPTERTEST	Apater01	Default	Select	SrcList		SrcList	SELECT
MyModel	Text	ADAPTERTEST	Apater01	Default	Insert	SrcList		SrcList	INSERT INTO
MyModel	Text	ADAPTERTEST	Apater01	Default	Update	SrcList		SrcList	UPDATE
MyModel	Text	ADAPTERTEST	Apater01	Default	Delete	SrcList		SrcList	DELETE FR...

View Schemas

모델에 등록된 DataItem의 Schema 정보를 확인 할 수 있습니다.

- MODEL_NAME : 모델 이름
- DATA_LAYER : Inputs/Outputs 구분
- CATEGORY : DataItem의 DataGroup 이름
- SCHEMA_NAME : DataItem 이름
- COLUMN_NAME : 컬럼 이름
- DATA_TYPE : 컬럼의 Data Type
- DEFAULT_VALUE : 기본 값 정보
- ALLOW_NULL : Null 값 허용 여부(허용(Y), 미허용(N))
- IS_PRIMARY_KEY : Primary Key 여부
- HIDDEN : Hidden Column 여부
- ~~CAPTION : Caption 정보~~
- DESCRIPTION : 설명 정보

View Databases | View DataActions | View Schemas

Drag a column header here to group by that column

MODEL_NAME	DATA_LAYER	CATEGORY	SCHEMA_NAME	COLUMN_NAME	DATA_TYPE	DEFAULT_VALUE	ALLOW_NULL	IS_PRIMARY_KEY	HIDDEN	CAPTION	DESCRIPTION
MyModel	Inputs	BOP	Process	LINE_ID	String						
MyModel	Inputs	BOP	Process	PROCESS_ID	String						
MyModel	Inputs	BOP	Product	LINE_ID	String						
MyModel	Inputs	BOP	Product	PRODUCT_ID	String						
MyModel	Inputs	BOP	Product	PROCESS_ID	String		Y				
MyModel	Inputs	BOP	Product	INPUT_INTERVAL	Decimal		Y				
MyModel	Inputs	BOP	Product	INPUT_STDDEV	Decimal		Y				
MyModel	Inputs	BOP	ProcStep	LINE_ID	String						
MyModel	Inputs	BOP	ProcStep	PROCESS_ID	String						
MyModel	Inputs	BOP	ProcStep	STEP_ID	String						
MyModel	Inputs	BOP	ProcStep	STEP_SEQ	Int32		Y				
MyModel	Inputs	BOP	ProcStep	STEP_TYPE	String		Y				
MyModel	Inputs	Simulation	Equipment	LINE_ID	String						
MyModel	Inputs	Simulation	Equipment	EQP_ID	String						
MyModel	Inputs	Simulation	Equipment	EQP_GROUP	String		Y				
MyModel	Inputs	Simulation	Equipment	SIM_TYPE	String		Y				
MyModel	Inputs	Simulation	Equipment	DISPATCHER_TYPE	String		Y				

Report Schema 기능

모델 정보는 Input, Output data부터 실행 옵션, 쿼리 및 DB연결 정보 등을 포함합니다. 이는 주요한 설계 정보입니다. 설계 내용의 가장 최신 정보를 모델을 통해 확인 할 수 있으며, HTML 형태로 문서를 출력하여 공유 할 수 있습니다. **[Project/Report Schema]** 메뉴를 사용합니다.

메뉴를 통해 생성된 문서는 아래 예제와 같이 모델 내의 모든 데이터를 포괄합니다.

FabSampleModel

DataSource

Name	Disabled	Hidden	LastUpdate	LastAccess	AccessCount	Description
FabDB	False	False			0	

DataSourceItem

Path	Index	Name	ProviderName	ConnectionString	Description
FabDB	0	SQL_SERVER	SqlServer	Initial Catalog=FabSampleDB;Data Source=192.168.1.11;User ID=sa;Password=qmdldpa2000;	

DataItem

Location	Name	Category	Disabled	Hidden	LastUpdate	LastAccess	AccessCount	Description
Inputs	Product	BOP	False	False	2020-08-21 15:23:23		0	
Inputs	Process	BOP	False	False	2020-08-21 15:23:23		0	
Inputs	ProcStep	BOP	False	False	2020-08-21 15:23:23		0	
Inputs	Eqp	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	EqpArrange	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	StepTime	Simulation	False	False	2020-08-21 15:23:24		0	
Inputs	InPlanInfo	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	PresetInfo	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	Wip	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	StepRoute	BOP	False	False	2020-08-21 15:23:23		0	
Inputs	ReticleInfo	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	ReticleArrangeInfo	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	LotInPlan	Simulation	False	False	2020-08-21 15:23:23		0	
Inputs	Demand	Pegging	False	False	2020-08-21 15:23:23		0	

DataSchema

Location	Path	Index	Name	DataType	DefaultValue	AllowNulls	IsPrimaryKey	Caption	Editor	Description
Inputs	Product	0	LINE_ID	string		True	False			
Inputs	Product	1	PRODUCT_ID	string		True	False			
Inputs	Product	2	STD_PRODUCT_ID	string		True	False			
Inputs	Product	3	PROCESS_ID	string		True	False			
Inputs	Process	0	LINE_ID	string	={A,B}	False	True		ComboBox(...)	
Inputs	Process	1	PROCESS_ID	string		False	True			
Inputs	ProcStep	0	LINE_ID	string		False	True			production line id for process
Inputs	ProcStep	1	PROCESS_ID	string		False	True			process id
Inputs	ProcStep	2	STEP_ID	string		False	True			step id
Inputs	ProcStep	3	STEP_DESC	string		True	False			step description
Inputs	ProcStep	4	STEP_SEQ	int		True	False			step sequence
Inputs	ProcStep	5	STD_STEP_ID	string		True	False			standard step id
Inputs	ProcStep	6	STEP_TYPE	string		True	False			step type (MAIN/SUB)
Inputs	ProcStep	7	STEP_GROUP	string		True	False			step group (PHOTO/ETCH/IMP/DIFF...)
Inputs	ProcStep	8	PROCESS_GROUP	string		True	False			eqp group for step processing
Inputs	Eqp	0	LINE_ID	string		True	False			
Inputs	Eqp	1	EQP_ID	string		True	False			
Inputs	Eqp	2	PARENT_EQP_ID	string		True	False			
Inputs	Eqp	3	PROCESS_GROUP	string		True	False			
Inputs	Eqp	4	STEP_GROUP	string		True	False			

DataAction

Location	Path	Index	Name	DataSource	Active	LastUpdate	LastAccess	AccessCount	Description
Inputs	Product	0	Default	FabDB	True			0	
Inputs	Process	0	Default	FabDB	True			0	
Inputs	ProcStep	0	Default	FabDB	True			0	
Inputs	Eqp	0	Default	FabDB	True			0	
Inputs	EqpArrange	0	Default	FabDB	True			0	
Inputs	StepTime	0	Default	FabDB	True			0	
Inputs	InPlanInfo	0	Default	FabDB	True			0	
Inputs	PresetInfo	0	Default	FabDB	True			0	
Inputs	Wip	0	Default	FabDB	True			0	
Inputs	StepRoute	0	Oracle	FabDB	False			0	
Inputs	StepRoute	1	MSSQL	FabDB	True			0	
Inputs	ReticleInfo	0	Default	FabDB	True			0	
Inputs	ReticleArrangeInfo	0	Default	FabDB	True			0	
Inputs	LotInPlan	0	Default	FabDB	True			0	
Inputs	Demand	0	Default	FabDB	True			0	
Inputs	Yield	0	Default	FabDB	True			0	
Inputs	StepTatInfo	0	Default	FabDB	True			0	
Inputs	PmSched	0	Default	FabDB	True			0	
Inputs	MttrInfo	0	Default	FabDB	True			0	
Inputs	BathInfo	0	Default	FabDB	True			0	
Inputs	QueueTimeConst	0	Default	FabDB	True			0	
Inputs	DiffCleanInfo	0	Default	FabDB	True			0	
Inputs	EqpDedicAct	0	Default	FabDB	True			0	

DataCommand

Location	Path	Index	Name	CommandText
Inputs	Product/Default	0	Cmd1	SELECT A.LINE_ID, A.PRODUCT_ID, A.STD_PRODUCT_ID, A.PROCESS_ID FROM GEN_PRODUCT A
Inputs	Process/Default	0	Cmd1	SELECT A.LINE_ID, A.PROCESS_ID FROM GEN_PROCESS A
Inputs	ProcStep/Default	0	Cmd1	SELECT A.LINE_ID, A.PROCESS_ID, A.STEP_ID, A.STEP_DESC, A.STEP_SEQ, A.STD_STEP_ID, A.STEP_TYPE, A.STEP_GROUP, A.PROCESS_GROUP FROM GEN_PROCSTEP A
Inputs	Eqp/Default	0	Cmd1	SELECT A.LINE_ID, A.EQP_ID, A.PARENT_EQP_ID, A.PROCESS_GROUP, A.STEP_GROUP, 1 IS_ACTIVE, A.SIM_TYPE, A.DISPATCHER_TYPE, A.PRESET_ID, 1.0 OPERATING_RATIO, A.CHAMBER_COUNT, A.MIN_BATCH_SIZE, A.MAX_BATCH_SIZE, CASE WHEN A.SIM_TYPE = 'LotBatch' OR A.SIM_TYPE = 'BatchInline' then 1200 ELSE 0 END BATCH_WAIT_TIME, A.STATUS, A.STATUS_CHANGE_TIME FROM GEN EQUIPMENT A

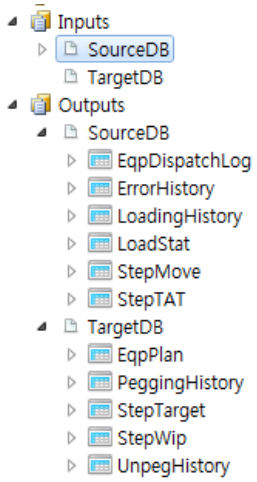
View & Check Model

MOZART Model 은 실행에 필요한 Input data, Input Argument 에 대한 정보, 실행 결과를 저장하기 위한 Output data 에 대한 정의 정보를 포함합니다. 또한 Input Data 의 추출과 Output 데이터 저장을 위한 DataSource 정보를 저장하고 있습니다. 해당 모델은 모두 MOZART IDE 의 모델 편집 기능을 통해 작성되며, MOZART STUDIO 의 **[Tree View/Model Tab]**을 통해 동일한 View 로 조회가 가능합니다.

1. Arguments : Input Argument/Config Argument 를 확인 합니다.
   * Input Argument : Argument 값이 Model 실행 시점에 고정적으로 결정 가능한 경우 사용되는 Argument 입니다.
   * Config Argument : Argument 값 자체를 실행 Context 에 따라 변경된 값을 사용해야 하는 경우에 사용됩니다. 예를 들어, 실행하는 라인에 따라 동일 Argument 의 사용값이 변경되어야 하는 경우 Config Argument 를 사용할 수 있습니다.
2. Database : Model 전체에서 사용할 수 있는 Data Source 를 확인 할 수 있습니다.
3. Inputs : 모델 실행을 위해 필요한 Input 데이터 Schema 및 데이터를 다운로드하기 위해 필요한 쿼리(DataAction)을 확인 할 수 있습니다. Data View 에서 Schema 확인 및 쿼리 수정 방식은 **Data Item View 사용방식**을 참조합니다.
4. Outputs : 모델 실행 결과가 저장되기 위한 데이터 Schema 및 DB 저장에 필요한 쿼리(DataAction)을 확인 할 수 있습니다. Data View 에서 Schema 확인 및 쿼리 수정 방식은 **Data Item View 사용방식**을 참조합니다.

각 기능별 정의방식은 MOZART IDE 의 **MODEL 편집 기능**을 참조합니다.

추가적으로 **[Model Tab View]**에서 **[Project/View dataitem by Database]** 메뉴를 사용하면 Input, Output 데이터의 경우 대상 데이터를 다운로드 혹은 저장을 위해 연결된 데이터 소스별로 Data Item 을 그룹핑하여 표시 할 수 있습니다. 아래 그림은 기능이 적용된 TreeView 를 표시합니다.



DataItem 정보 Import/Export 하는 방법

Mozart IDE 또는 Mozart Studio에서는 DataItem 정보를 Import 또는 Export 할 수 있습니다. 스키마가 추가되거나 변경되었을 때 해당 모델을 운영에 적용하기 전 현재 운영 중인 모델에 스키마를 추가하여 테스트할 때 사용하거나, 신규 프로젝트를 진행할 때 비슷한 스키마가 다른 프로젝트에 있는 경우 새로 정의하지 않고 가져오는 용도로 사용이 가능합니다.

DataItem Import/Export 주요 사항

파일 확장자

Mozart IDE 또는 Mozart Studio에서 Export된 DataItem의 확장자는 다음과 같습니다.

- 개별 파일: vditem
- Input 또는 Output 파일 집합: vditems

> **Note**
> Mozart IDE와 Mozart Studio의 DataItem 정보 파일의 확장자가 통일된 것은 2021.119.1 버전부터 입니다. 이전 버전에서 Mozart IDE에서 Export한 vditem을 Mozart Studio에서 Import 하려면 Export 후 파일 확장자를 vdinfo로 변경하시면 됩니다.

메뉴 설명

Mozart IDE에서 DataItem을 선택하여 마우스 우클릭으로 Context Menu를 열면 Import, Export 메뉴를 사용할 수 있습니다. Import, Export 단위에 따라 사용되는 명령은 다음과 같습니다.

- **Import, Export > [DataItem(N)...]**: 선택한 DataItem 1개 또는 복수의 DataITem을 Import 또는 Export할 수 있습니다. (vditem)
- **Import, Export > [DataItems...]**: 선택한 DataGroup이 속한 Input 또는 Output 전체를 Import, Export합니다. (vditems)

Import/Export되는 정보들

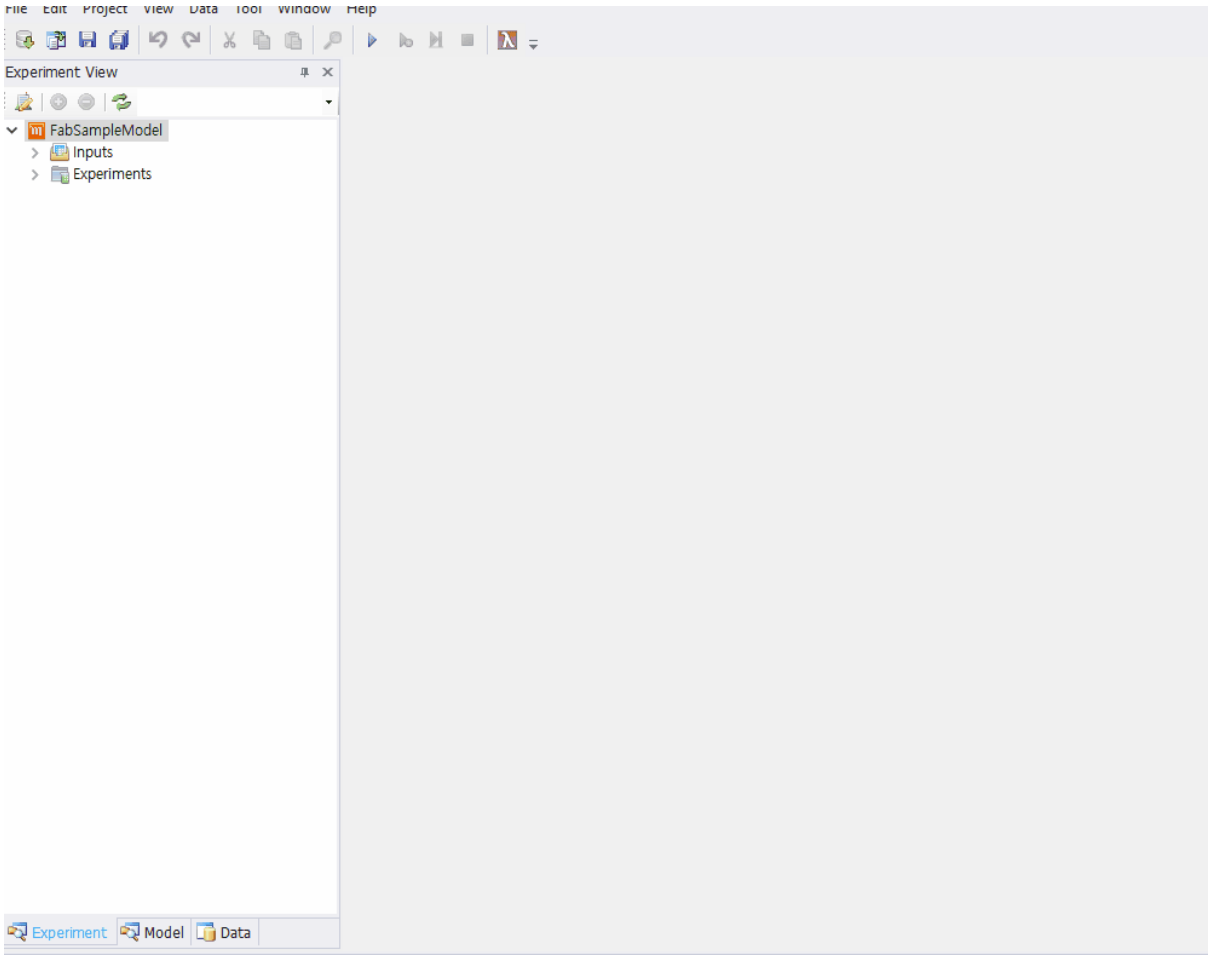
DataItem을 Import, Export하면 다음 정보들이 포함됩니다.

- DataItem 스키마 정보
- DataAction 구현 및 설정 내용

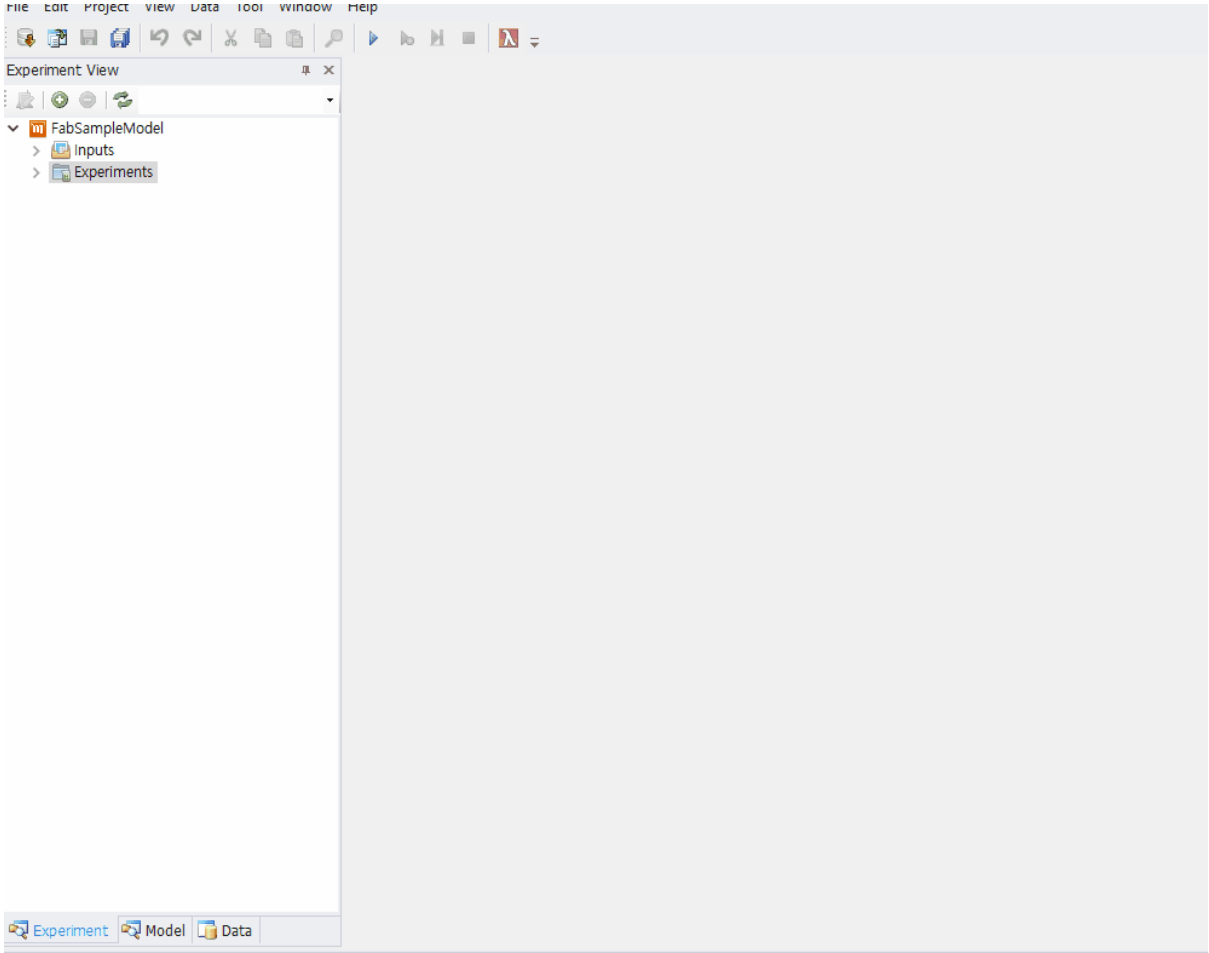
Connection String이 포함된 DataSource의 경우 DataSource를 별도로 Import/Export 해야합니다.

DataItem Export 예시

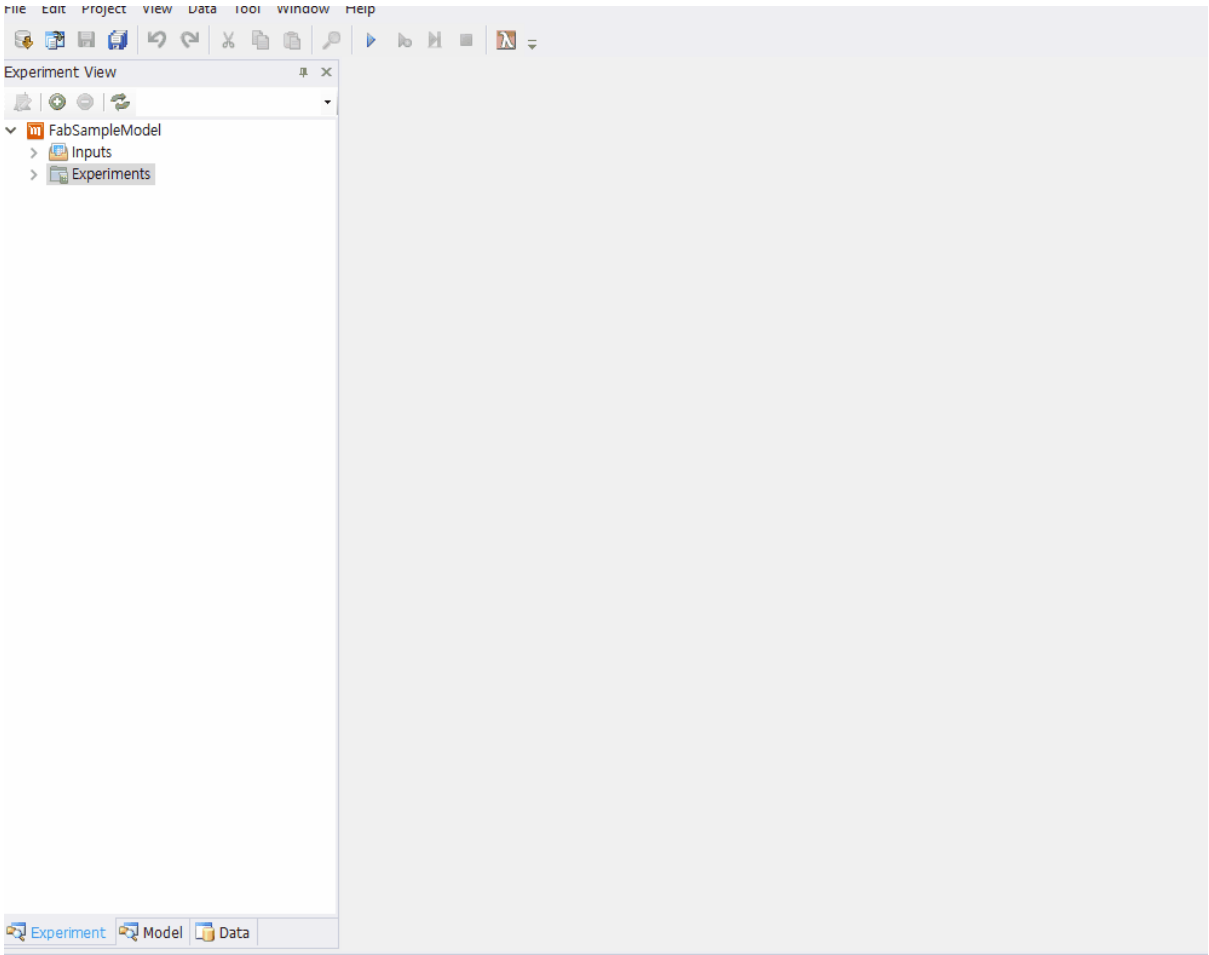
단일 DataItem Export하기



다중 DataItem Export하기

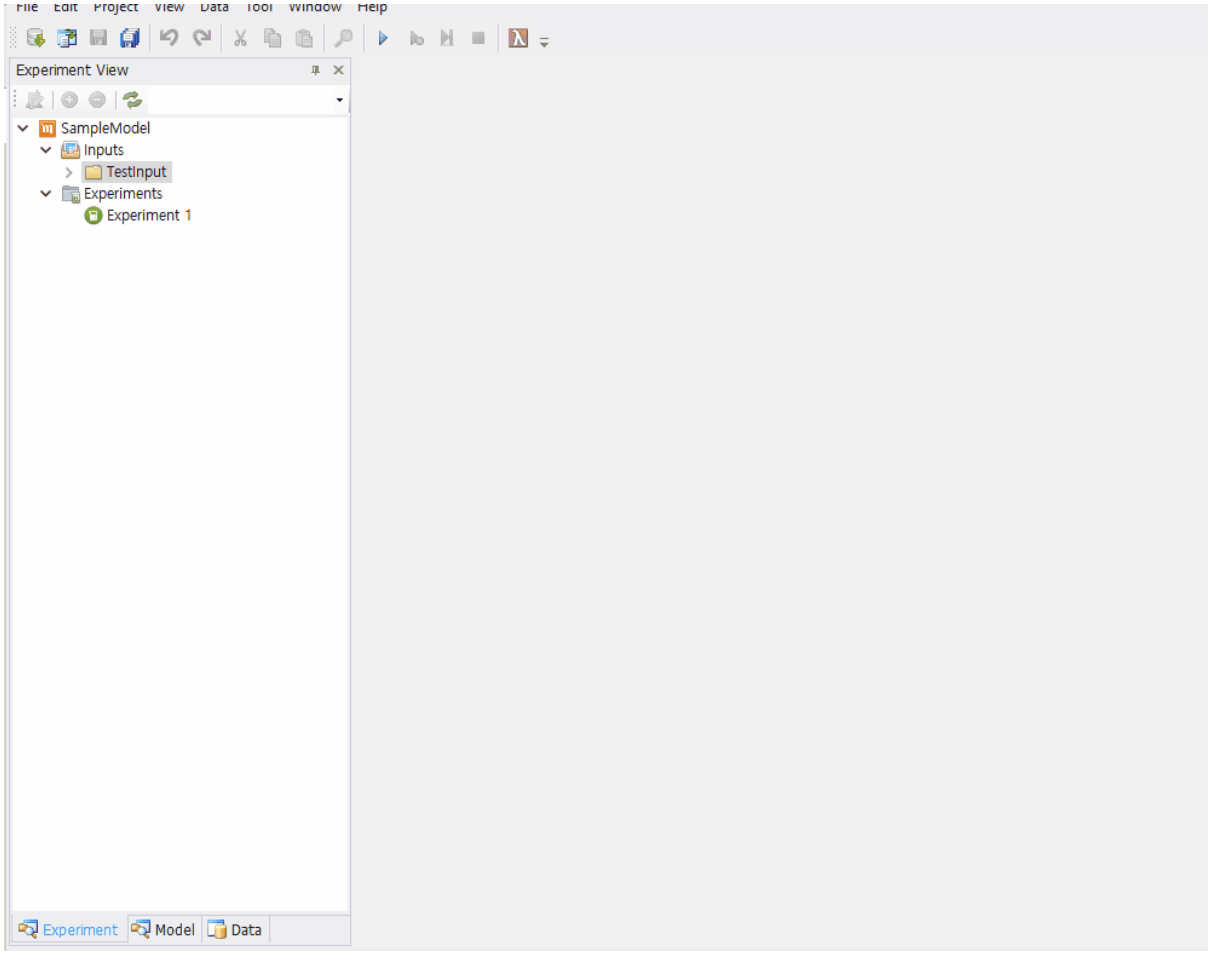


스키마 전체 Export하기

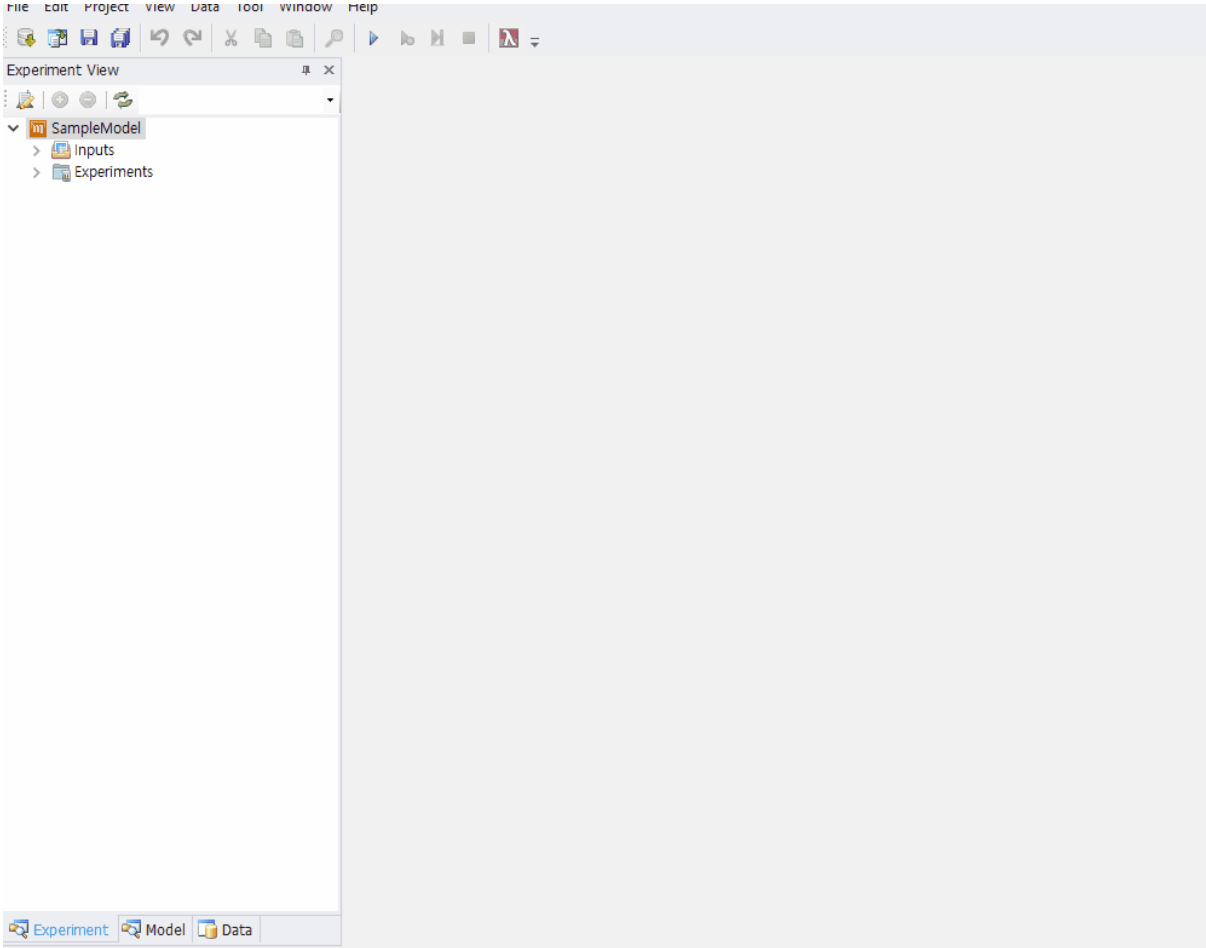


DataItem Import 예시

단일/다중 DataItem Import하기



스키마 전체 Import하기



Mozart Studio에서 Export한 DataItem을 Mozart IDE에서 Import 하는 방법은 DataItme 생성 / 편집 방법를 참고 바랍니다.

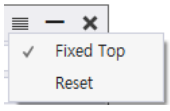
데이터 Copy Tool 사용법

사용 가능 버전 : **MOZART 2021.119.2 이상**

Copy Tool 기능을 통해 데이터 아이템 또는 그리드 뷰, 피봇 뷰(Open Item (Read Only))에서 복사한(Ctrl + C) 데이터의 텍스트 포맷을 변경할 수 있습니다. Copy Tool은 데이터를 복사할 때 구분자 설정, 정렬, 중복 제거, 행/열 변경 등의 추가 설정이 가능한 보조툴입니다. 복사 설정은 **Copy Tool 팝업창이 활성화**된 상태에서 **데이터를 복사**할 때 적용됩니다. Copy Tool 팝업창이 비활성화가 되면 기본 복사 기능이 적용됩니다.

Copy Tool

Studio의 상단 메뉴 Tool > Copy Tool을 실행하면 Copy Tool 팝업창이 나타납니다. Copy Tool 을 처음 실행하면 기본 Copy 포맷이 설정되어 있습니다. 기본 Copy 포맷은 행 구분자가 엔터(\r\n)이고 컬럼 구분자가 탭(\t) 입니다.



Copy Tool의 ≡ 메뉴

Copy 설정

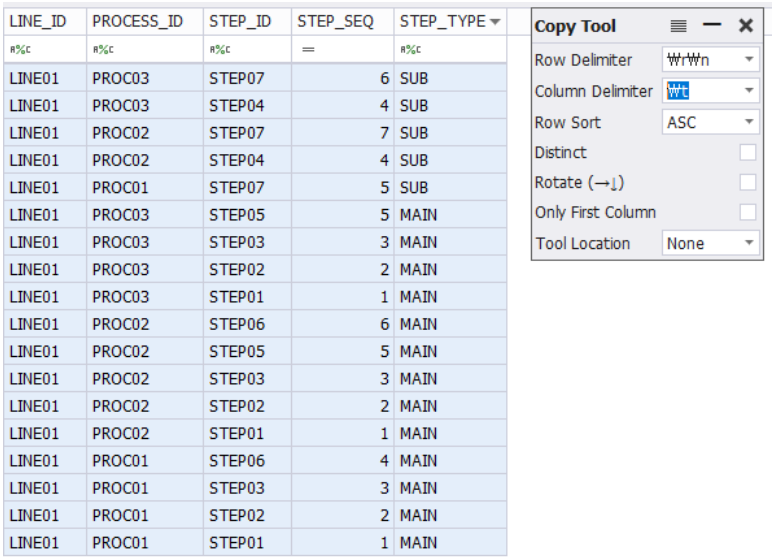
Copy 설정은 팝업창을 닫아도 자동 저장됩니다.

- **Fixed Top** : 체크 시 Copy Tool 팝업창을 항상 위로 고정합니다.
- **Reset** : Copy Tool 포맷 설정을 기본 설정으로 변경합니다.
- **Row Delimiter** : 데이터 행 간 구분자를 선택하거나 입력하여 설정합니다.
  - 구분자 선택 : `(None)`, `\t(Tab)` , `\r\n(Enter)`, `;`, `:`, `,`, `.`, `/`, `-`
- **Column Delimiter** : 데이터 컬럼 간 구분자를 선택하거나 입력하여 설정합니다.
  - 구분자 선택 : `(None)`, `\t(Tab)` , `\r\n(Enter)`, `;`, `:`, `,`, `.`, `/`, `-`
- **Row Sort** : 행 기준으로 정렬합니다. (`ASC` : 오름차순, `DESC` : 내림차순)
- **Distinct** : 중복되는 행을 제외하여 복사합니다.
- **Rotate** : 데이터 행, 열을 변경합니다. (기본 : `→↓`, 체크 시 : `↓→`)
- **Only First Column** : 복사한 데이터 중 첫 컬럼 데이터만 복사합니다.
- **Tool Location** : Copy Tool 팝업창의 위치를 선택합니다.(ex. `Top`, `Center`, `Bottom`, ...) 위치를 지정하지 않고 드래그를 통해 팝업창 위치를 정할 수 있습니다.

> 💡 그리드 뷰, 피봇 뷰에서 **컬럼 헤더를 제외하고 데이터 복사**가 가능합니다.
> - Tool > Options ... > GridOptions > Copy Column Headers 체크 해제
> - Tool > Options ... > PivotOptions > Copy To Clipboard With Field Values 체크 해제

> ✔️ **Copy Tool 설정 변경 후 데이터 복사(Ctrl + C)**를 진행해야 합니다.

Copy Tool 예제



LINE_ID	PROCESS_ID	STEP_ID	STEP_SEQ	STEP_TYPE
LINE01	PROC03	STEP07	6	SUB
LINE01	PROC03	STEP04	4	SUB
LINE01	PROC02	STEP07	7	SUB
LINE01	PROC02	STEP04	4	SUB
LINE01	PROC01	STEP07	5	SUB
LINE01	PROC03	STEP05	5	MAIN
LINE01	PROC03	STEP03	3	MAIN
LINE01	PROC03	STEP02	2	MAIN
LINE01	PROC03	STEP01	1	MAIN
LINE01	PROC02	STEP06	6	MAIN
LINE01	PROC02	STEP05	5	MAIN
LINE01	PROC02	STEP03	3	MAIN
LINE01	PROC02	STEP02	2	MAIN
LINE01	PROC02	STEP01	1	MAIN
LINE01	PROC01	STEP06	4	MAIN
LINE01	PROC01	STEP03	3	MAIN
LINE01	PROC01	STEP02	2	MAIN
LINE01	PROC01	STEP01	1	MAIN

Copy Tool 예제

LINE_ID	PROCESS_ID	STEP_ID	STEP_SEQ	STEP_TYPE
LINE01	PROC01	STEP01	1	MAIN
LINE01	PROC01	STEP02	2	MAIN
LINE01	PROC01	STEP03	3	MAIN
LINE01	PROC01	STEP06	4	MAIN
LINE01	PROC01	STEP07	5	SUB
LINE01	PROC02	STEP01	1	MAIN
LINE01	PROC02	STEP02	2	MAIN
LINE01	PROC02	STEP03	3	MAIN
LINE01	PROC02	STEP04	4	SUB
LINE01	PROC02	STEP05	5	MAIN
LINE01	PROC02	STEP06	6	MAIN
LINE01	PROC02	STEP07	7	SUB
LINE01	PROC03	STEP01	1	MAIN
LINE01	PROC03	STEP02	2	MAIN
LINE01	PROC03	STEP03	3	MAIN
LINE01	PROC03	STEP04	4	SUB
LINE01	PROC03	STEP05	5	MAIN
LINE01	PROC03	STEP07	6	SUB

메모장에 데이터 붙여넣기 결과

라이선스 인증 방법

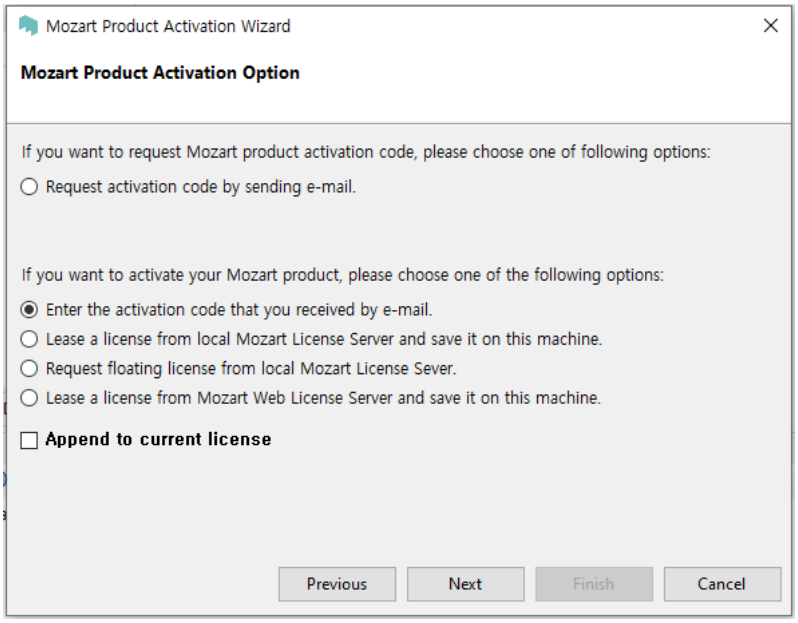
개요

MOZART Studio 나 IDE 에서 라이선스가 인증되지 않은 상태에서는 **Mozart Product Activation Wizard** 창이 활성화되어 라이선스 인증 절차를 수행하도록 지원합니다. 현재 MOZART 에서 지원하는 라이선스 활성화 방식은 아래의 세 가지 방식입니다.

1. **Activation 코드 입력을 통한 라이선스 활성화** : 라이선스 키 요청을 통해 발급 받은 Activation 코드를 Activation Wizard 에 입력하여 활성화 하는 방식 입니다. 라이선스 발급 요청이 필요할 시에는 MOZART 라이선스 키 요청 방법을 참고하시기 바랍니다.
2. **Local License Service 를 통한 라이선스 활성화** : 사용자 조직 내부에 MOZART SERVER 를 설치한 경우 MOZART IDE, STUDIO 에 대한 라이선스를 자동 발행 할 수 있습니다. MOZART SERVER 는 Activation code 발급을 통해서만 활성화가 가능합니다.
3. **Floating License를 통한 라이선스 활성화**: 사용자 조직 내부에 MOZART SERVER를 설치한 경우 MOZART STUDIO에 대한 라이선스를 MOZART STUDIO를 매번 실행 할 때마다 서버를 통해 인증 받는 방식입니다. 서버와의 접속이 끊기거나 허용된 네트워크를 벗어나면 MOZART STUDIO 사용이 자동 중단 됩니다.

Activation code 입력을 통한 Activation 방법

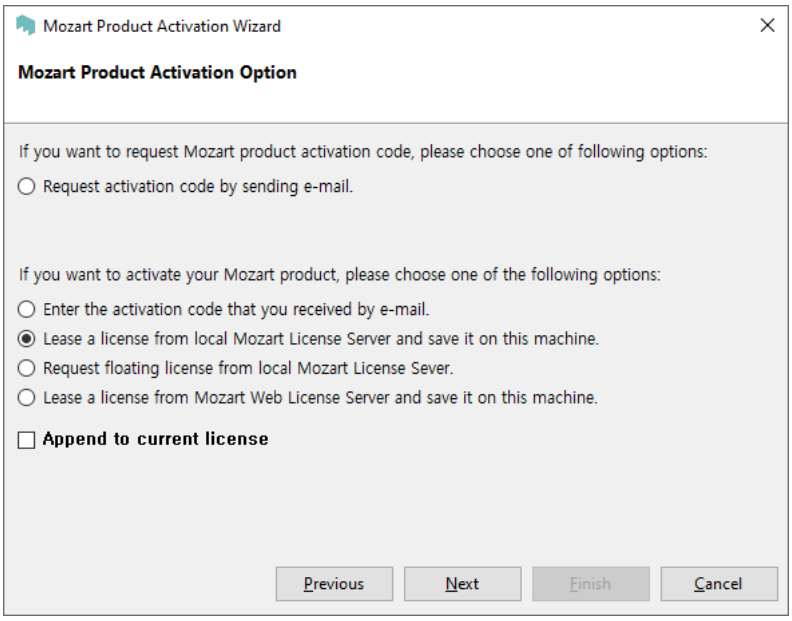
1. Mozart product activation option 선택 단계에서 "Enter the activation code that you received by e-mail" 을 선택한 후 **[Next]** 버튼을 클릭합니다.



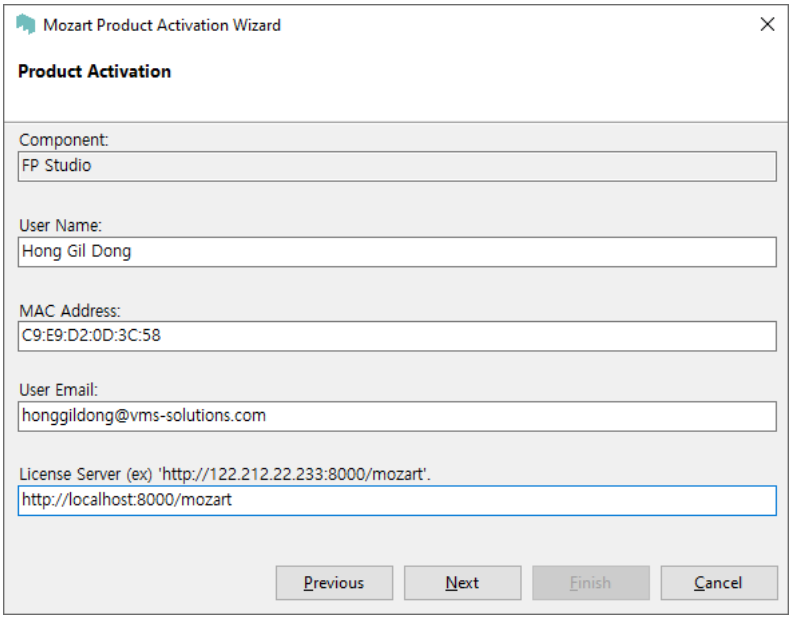
2. 수령한 Activation Code 를 복사하여 창에 붙여넣은 후 **[Next]** 버튼을 클릭합니다.
3. 정상적으로 활성화되면 활성화 정보가 표시됩니다.
4. **[Finish]** 버튼을 눌러 활성화를 마칩니다.

Local License Service 를 통한 Activation 방법

1. Mozart product activation option 선택 단계에서 하단의 “Lease a license from Mozart License Server” 옵션을 선택하고 **[Next]** 버튼을 클릭합니다.



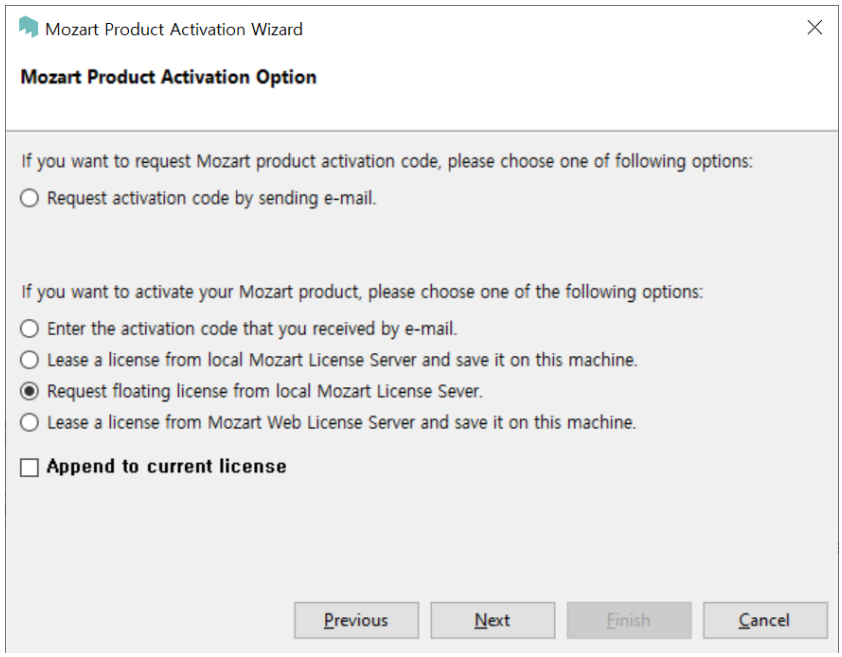
2. User Name, User Email 그리고 라이선스를 인증 받을 서버 주소를 입력합니다. 서버 주소 입력 시에는 Mozart Server 서비스의 포트 번호와 Application Name을 포함합니다.
(예: http://1.1.1.1:8000/mozart)



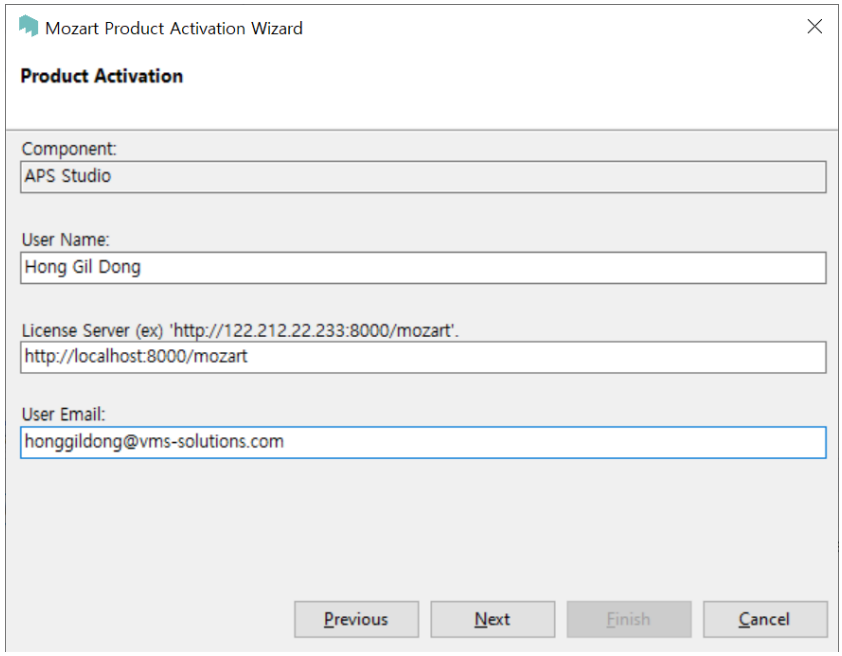
3. 정보를 다 입력하였다면 **[Next]** 버튼을 클릭합니다. 라이선스 인증이 완료되면 **[Finish]** 버튼을 클릭하여 **Mozart Product Activation Tool**를 종료합니다.

Floating License를 통한 Activation 방법

1. Mozart product activation option 선택 단계에서 하단의 “Request floating license from Mozart License Server” 옵션을 선택하고 **[Next]** 버튼을 클릭합니다.



2. User Name, User Email 그리고 라이선스를 인증 받을 서버 주소를 입력합니다. 서버 주소 입력 시에는 Mozart Server 서비스의 포트 번호와 Application Name을 포함합니다.
   (예: http://1.1.1.1:8000/mozart)



3. 정보를 다 입력하였다면 **[Next]** 버튼을 클릭합니다. 라이선스 인증이 완료되면 **[Finish]** 버튼을 클릭하여 **Mozart Product Activation Tool**를 종료합니다.

Model 개요 및 생성 방법

Model 의 개념

Model 은 MOZART 개발 Project 의 Logic 수행에 필요한 Input, Output Data (Schema) 및 실행 시점에 Data 를 Logic 에서 사용할 수 있는 객체 형태로 전환하거나 Logic 의 수행 결과를 Data File, Database 에 저장하기 위한 Query 와 설정정보를 저장하고 있는 정보파일입니다. 또한 Project 수행 방식이나 옵션을 위해 입력 받을 수 있는 Argument 정보도 포함합니다. MOZART IDE 를 통해 개발되는 Project 의 경우 대부분 다양한 Input Data 를 사용하고 여러 Data Source 를 사용하게 되는 경우가 많으며 Model 을 통해 이러한 정보를 일관되게 관리하고 모델링할 수 있습니다.

Model 은 하나의 MOZART Project 에 복수 개를 생성할 수 있습니다. 최초 생성된 Model 의 내용을 상속 받아 Child Model 을 생성하고 수정하여 사용할 수 있습니다. 다중 Model 생성 및 사용에 대해서 **Multi Model Project** 를 참조합니다

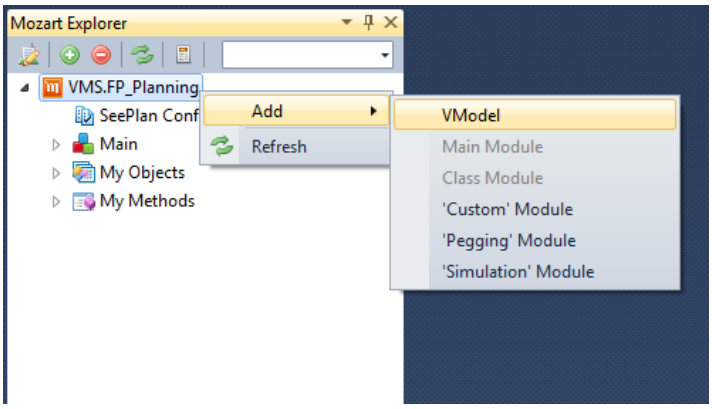
Model 의 구성요소

- Arguments : 실행 시 사용되는 Logic 실행의 옵션 및 설정값을 정의하는 기능입니다. Arguments 의 종류에는 Input Arguments 와 Config Arguments 가 있습니다.
- **Database** : Project 에서 사용되는 모든 Data 연결문자열을 관리하기 위한 기능입니다. 등록된 Data Source 는 Schema 편집 및 DataAction 편집 시 Data Source 속성에 설정값으로 사용이 가능합니다.
- **Inputs** : Task 의 Input Data Schema 와 DataAction 을 정의하고, 여러 Input Data 의 로드 순서 및 방법을 정의하기 위한 기능입니다.
- **Outputs**: Task 의 Output Data Schema 와 저장순서, 방법을 정의하기 위한 기능입니다. 기본 Outputs 의 경우 파일로 저장되며, Outputs 정의 시 옵션 및 Query 지정을 통해 추가적으로 Database 저장이 가능합니다.

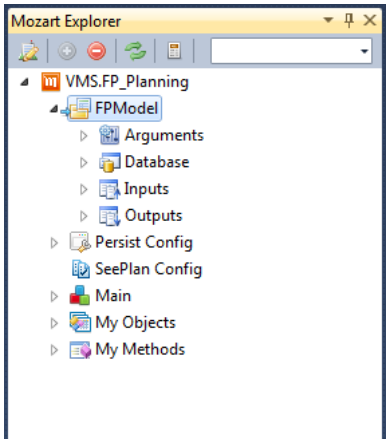
Project에 Model 추가하기

신규로 생성된 Project 에 Model 을 추가하는 방법은 아래와 같습니다.

1. MOZART Explorer 의 최상위 노드를 선택합니다.
2. 오른쪽 마우스 버튼으로 팝업메뉴를 활성화합니다.
3. **[Add > VModel]** 메뉴를 선택합니다.



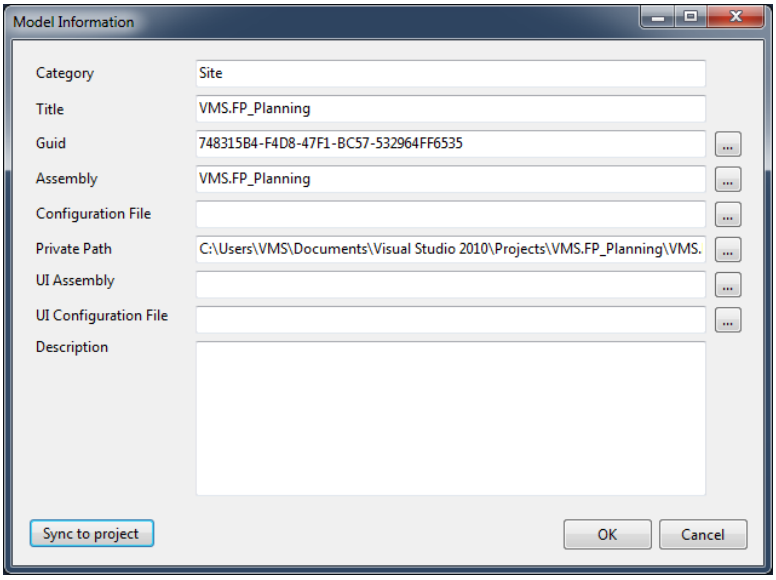
4. Model 명을 입력하는 다이얼로그가 활성화되면, 생성하려는 Model 명을 입력합니다.
5. MOZART Explorer 에서 입력된 Model 명의 Model 이 생성되었음을 확인합니다.



Model 정보 편집

Model 정보는 .vmodel 이라는 확장자로 저장이 되며 Model 의 구성요소에 대한 정보와 Model 의 속성정보를 포함합니다. MOZART IDE 를 통해 개발된 Task (Dll 파일) 는 반드시 Model 과 연계되어 수행할 수 있으므로 Model 에는 Model 을 실행할 dll 과 연관 관계 정보, Model 실행을 위한 환경 설정 정보가 포함됩니다. 이 정보의 조회와 편집 방법은 아래 절차를 따릅니다.

1. Model 노드를 선택하고 팝업메뉴로 Model 정보 편집메뉴 **[Edit Vinfo]** 를 실행하면 아래와 같은 Model 정보 편집창이 열립니다.



2. 필요한 항목에 대한 편집을 수행하고 저장합니다. 항목에 대한 설명은 아래와 같습니다.
   - Base Library : 소스 프로젝트와 연결된 도메인 라이브러리 이름입니다. 'SeePlan.General' 과 같이 SeePlan. 뒤에 표시되는 이름이 도메인 라이브러리입니다.
   - Version : 소스 프로젝트와 연결되는 도메인 라이브러리의 버전을 선택해야 합니다. 선택 대상은 개발 PC에 설치된 Base Library의 버전을 선택할 수 있습니다.
   - Category : Model 의 유형을 의미하며, 기본값은 Project 생성 시의 Site Prefix 로 설정됩니다.
   - Title : Model 의 명칭입니다.
   - Guid : Model 에 대한 GUID 입니다. Model 을 기반으로 만들어진 Task (Job Type) 를 구분하기 위한 ID 로 사용됩니다. 오른쪽 버튼을 사용해서 재생성할 수 있습니다.
   - Assembly : Model 을 실행하기 위한 Dll 명입니다. 오른쪽 버튼을 사용해서 지정할 수 있습니다.
   - Configuration File : Model 실행을 위한 설정 파일을 지정합니다. 오른쪽 버튼을 사용해서 파일을 선택하거나 직접 입력할 수 있습니다.
   - Private Path : 실행 대상 Assembly 파일(Dll 파일)을 위치시키는 폴더명입니다. 오른쪽 버튼을 이용하여 사용자가 직접 지정할 수 있으며, 하단의 Sync to project 명령을 사용하면 Model 이 포함된 MOZART Project 의 "bin/Debug" 로 자동 설정이 됩니다.
   - Description : Model 에 대한 설명 입니다.
3. 본 설정 내용은 Model 파일(.vmodel) 에 저장되며 필요시 해당 정보만 파일로 Export 하는 것이 가능합니다. 팝업 메뉴의 **[Export VInfo]** 를 실행합니다.
4. 파일이 저장될 폴더를 지정하면 "Model 명.vinfo" 파일이 생성됩니다.
5. VInfo 파일에는 상기 Model 정보가 포함되며, 향후 MOZART Studio 에 사용자정의 User Interface 를 추가할 때 필요합니다. 생성된 파일의 내용은 아래 그림과 같습니다.

VInfo 파일 생성내용

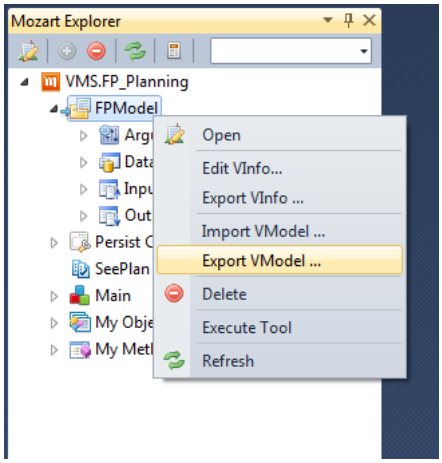
```
<target guid="7db113e2-1e7d-493a-8b2b-be22dd1b13ae" category="Site" title="VMS.FP_Planning">
  <path configurationFile="" privatePath="D:\Mozart\VMS.FP_Planning\VMS.FP_Planning\bin\Debug" />
  <type assemblyName="VMS.FP_Planning" typeName="" />
  <description />
</target>
```

Model Export, Import하기

유사한 Proejct 를 수행하게 될 경우 Model 을 복사하여 신규 Project 를 수행해야 하는 요구가 있을 수 있습니다. 이때에는 기존의 Source Model 을 Export 하여 신규 Model 에 Import 하여 사용할 수 있습니다.

Export 하기

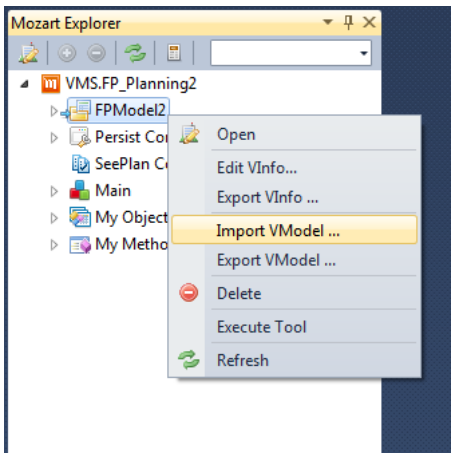
1. Model 노드에서 팝업메뉴의 [**Export VModel**] 메뉴를 실행합니다.



2. Model 저장 폴더를 지정한 후 저장합니다.

Import 하기

1. 신규 Project 에서 우선 빈 Model 을 추가합니다.
2. 추가된 Model 노드에서 팝업 메뉴의 [**Import VModel**] 메뉴를 실행합니다.



3. 기존에 Export 했던 .vmodel 파일을 선택합니다.
4. Model 정보를 확인해보면 신규 Project 에 맞도록 정보가 변경되었음을 확인할 수 있으며, 기존 Model 의 모든 Data 가 신규 Model 정보에 생성됨을 확인할 수 있습니다.

DataItem 생성 / 편집 방법

DataItem 은 Data Schema 에 Data 를 채우거나 Schema의 Data 를 외부 Database 에 저장하기 위한 Query 집합인 DataAction으로 구성됩니다. MOZART 에서는 Query 관리를 위해 DataAction이라는 개념을 사용합니다.

Inputs, Outputs 모두 동일한 구조의 DataItem 을 사용합니다. Schema 와 DataAction 은 1:N 의 관계를 가집니다. 즉, 하나의 DataItem 은 한 개의 Schema 와 1개 이상의 DataAction 으로 구성될 수 있습니다. Outputs 의 경우 Database 에 저장이 필요 없을 때에는 DataAction 없이 Schema 만 생성할 수 있습니다. DataAction 또한 하나의 Query 만으로 구성되지 않으며, 내부적으로 복수 개의 Query(Command) 를 포함하며 실행할 때는 등록된 Command 순서에 따라 순차적으로 실행합니다. 따라서 특정Table 에 Data 를 입력하기 전에 해당 Table 의 Data 를 지워야 하는 경우, 하나의 DataAction 에 Delete Command 와 Insert Command 를 순차적으로 실행할 수 있도록 구성되어 있습니다.

MOZART 라이선스 키 요청 방법

소개

MOZART 제품키 발급 절차가 변경됩니다. 2022.4.6 일부터는 MOZART 홈페이지에 가입된 사용자가 라이선스 발급 기능을 통해 자동으로 라이선스를 받을 수 있습니다. 자세한 사용 방법은 본 가이드를 참고해주시기 바랍니다.

기존과 달라진 점과 특징

라이선스 키 신청이 홈페이지로 이관되면서 기존 방식과 달라진 점은 다음과 같습니다.

- 키 자동 발급으로 인해 대기 시간 최대 5분으로 단축
- 라이선스 키 신청 현황을 실시간으로 확인 가능
- 요청서 작성 시 반복적인 정보 입력 최소화

사용자 구분

홈페이지에 가입된 회원에 한해서 라이선스 키 신청을 할 수 있습니다..가입된 회원 중 일반 사용자와 사이트 관리자로 사용자 역할이 분류됩니다.

- **사이트 관리자**: 가입 회원 중 일부 임직원에게만 부여되는 역할로 본인 외 근무하는 사이트의 고객 또는 고객 자산의 서버의 라이선스 키를 대리 신청 할 수 있는 사용자입니다. 사이트 관리자는 본사 측에서 메일을 통해 대상자 지정을 요청합니다.
- **일반 사용자**: 사이트 관리자 외는 모두 일반 사용자입니다. 일반 사용자는 본인 외 같은 기업 소속 사용자의 라이선스 키를 대리 신청 할 수 있는 사용자입니다.

라이선스 키 자동 발급 여부 및 기간

라이선스 유형 그리고 제품 Component에 따라 라이선스 키 신청 시 자동 발급 여부와 기간이 결정됩니다.

				신청권한		
라이선스 유형	컴포넌트	사용기간(발급일로부터)	자동 발급 여부	사이트관리자	자사직원	고객사용자
구매 - 고객용	Mozart Studio	1년	Y	✓	✗	✓
	Mozart IDE	1년	구매수량 이내: Y	✓	✗	✓
	Mozart Server	영구				
	Mozart WISE	1년	N	✓	✗	✓
대여 - 임직원용	Mozart Studio	1년	Y	✓	✓	✗
	Mozart IDE					
	Mozart WISE					
	Mozart Server	1년	N	✓	✓	✗
평가	Mozart Studio	7일	Y	✓	✓	✓
	Mozart IDE					
	Mozart Server					
	Mozart WISE					

라이선스 유형

- **구매**: MOZART 제품을 구매하여 라이선스 계약이 체결된 **고객 전용** 라이선스 유형입니다. 홈페이지에 가입된 고객(일반 사용자) 및 사이트 관리자는 해당 유형으로 라이선스 키 신청이 가능합니다. 프로젝트 수행 용도로만 키 신청이 가능합니다.
- **대여**: **임직원 전용**이며 일반 사용자 및 사이트 관리자 모두 신청이 가능한 유형입니다. 프로젝트 수행 용도로만 키 신청이 가능하며 근무하는 사이트와 관계없이 임직원이 사용하는 경우 해당 유형으로 키 신청을 해야 합니다. 고객 자산의 PC에 라이선스 키를 신청하는 경우에는 사이트 관리자가 Company를 상주하는 고객사 변경 후 **[구매]** 유형에서 신청을 해야 합니다.
- **평가**: 제품 평가, 교육 수행 그리고 긴급 사용 용도를 위한 임시 라이선스 키를 발급 받을 수 있는 유형입니다. 홈페이지에 가입된 회원 **누구나** 신청이 가능합니다.

제품 라이선스 키 신청 방법

인터넷 익스플로러(IE)에서는 제품 라이선스 키 신청 페이지가 열리지 않습니다. 크롬이나 엣지 브라우저를 통해 접속하기 바랍니다.

1. 로그인 후 My Page 이동

라이선스 키 신청은 로그인 후 My Page에서 신청이 가능합니다. 사용자 정보 우측에 사용자가 속한 고객사와 라이선스 키를 발급 받을 수 있는 제품 목록으로 표시됩니다. 임직원 중 사이트 관리자는 기업을 변경하여 제품 키를 신청 할 수 있습니다.

> 💡 **도움말:**
> **[Company]** 목록에 고객 기업 또는 본인이 소속된 기업이 표시되지 않은 경우, 그리고 기업을 선택하였는데 제품이 표시되지 않은 경우 support@vms-solutions.com로 문의 주시기 바랍니다. 다음과 같은 사유로 표시되지 않을 수 있습니다.
>
> **[목록에 기업이 없는 경우]**
> 1. 사이트 관리자가 라이선스 키를 대리 신청 가능한 기업이 아님
> 2. 대상 기업에서 제품 구매를 하지 않음
>
> **[제품이 없는 경우]**
> 1. 기업은 등록되었으나 아직 제품 구매를 하지 않았거나 구매 정보가 등록되지 않음

2. 라이선스 키 요청서 작성

My Page에서 **[Get Key]**를 선택하면 대상 제품에 대한 라이선스 키 요청 페이지가 열립니다. 아래는 요청서 화면 구성입니다.

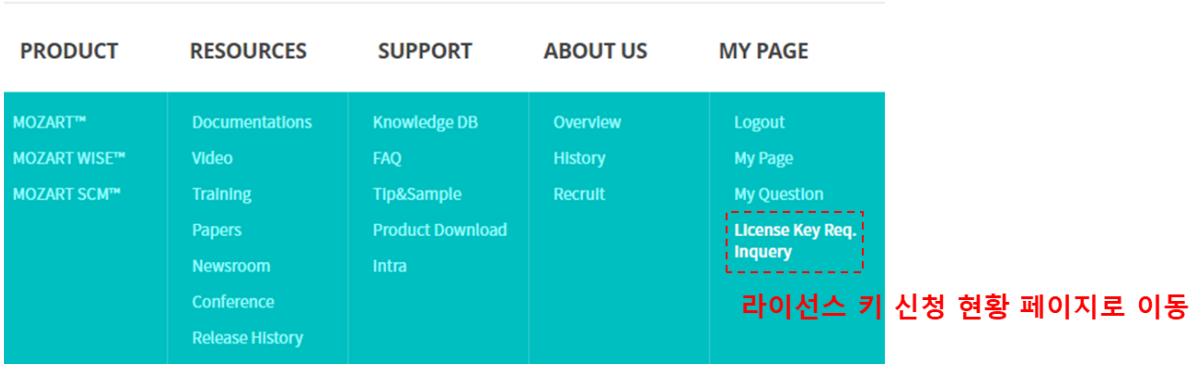
- **키 요청자 기본 정보**: 요청자가 선택한 제품과 라이선스 유형에 대한 정보를 표시합니다. 해당 영역은 수정이 불가하며 키를 요청하기 전 대상을 올바르게 선택하였는지 확인하기 위한 용도입니다.
- **키 요청서**: 키를 요청하기 위해 작성이 필요한 내용입니다.
- **라이선스 키 요청 목록**: 요청서 작성 후 **[추가]**를 선택하였을 때 목록에 추가되는 키 요청 내용입니다.
- **요청 삭제 버튼**: 요청서 추가 후 목록에 있는 요청서를 제거하기 위한 버튼입니다.
- **도움말**: MAC 주소 또는 컴퓨터 이름 확인 방법에 대해 설명하는 도움말과, 자동 발급 남은 키 수량에 대한 도움말을 표시합니다.

라이선스 키 요청은 다음과 같이 진행합니다.

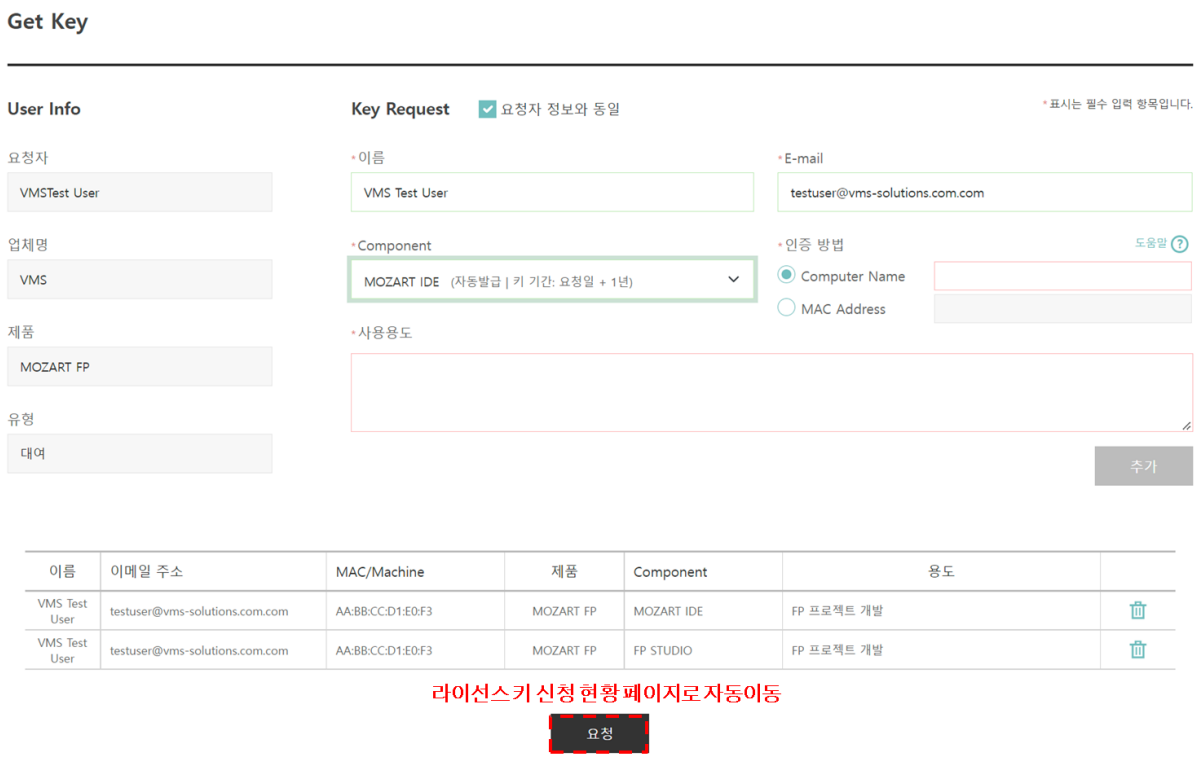
1. 키를 발급 받는 대상, 이메일 주소, 컴포넌트, 사용 용도, MAC 주소(**확인 방법**) 또는 컴퓨터 이름(확인 방법)에 대한 내용을 모두 작성합니다. 모든 항목에 대한 내용이 채워지면 **[추가]** 버튼이 활성화 됩니다.
2. 요청서 작성이 끝나면 **[추가]** 버튼을 눌러 요청 목록에 추가합니다. 다른 사용자 또는 컴포넌트에 대해 추가적으로 키 요청이 필요하면 1번 과정을 다시 진행하여 **[추가]** 버튼을 눌러 요청을 추가합니다.
3. 모든 요청이 추가 되었으면 **[요청]** 버튼을 눌러 라이선스 키를 신청합니다.

3. 라이선스 키 신청 현황

라이선스 키 신청 현황은 홈페이지 헤더 메뉴의 **My Page > License Key Req. Inquery**에서 확인 할 수 있습니다. 또한, 라이선스 키 요청서 페이지에서 키를 요청하게 되면 자동으로 라이선스 신청 현황 페이지로 이동합니다.



홈페이지 헤더 메뉴를 통해 이동하는 방법



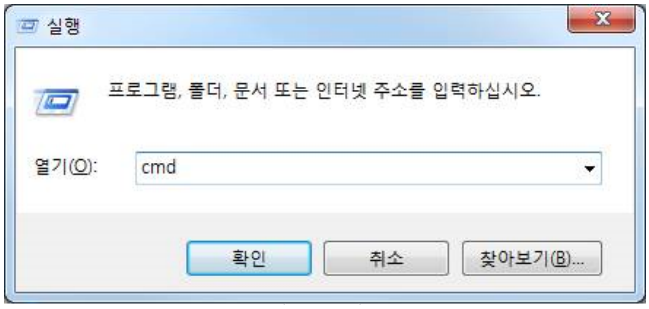
라이선스 키 신청 현황 페이지에서는 로그인한 사용자가 요청한 키의 전반적인 정보와 발급 상태를 조회 할 수 있습니다. 페이지 상단에서 발급 상태 그리고 신청 기간 조건을 설정하여 신청 현황을 조회 할 수 있습니다. 페이지를 처음 열 때는 최근 1주일에 대한 신청 현황을 조회 할 수 있습니다.

문의 및 문제 발생 시 연락

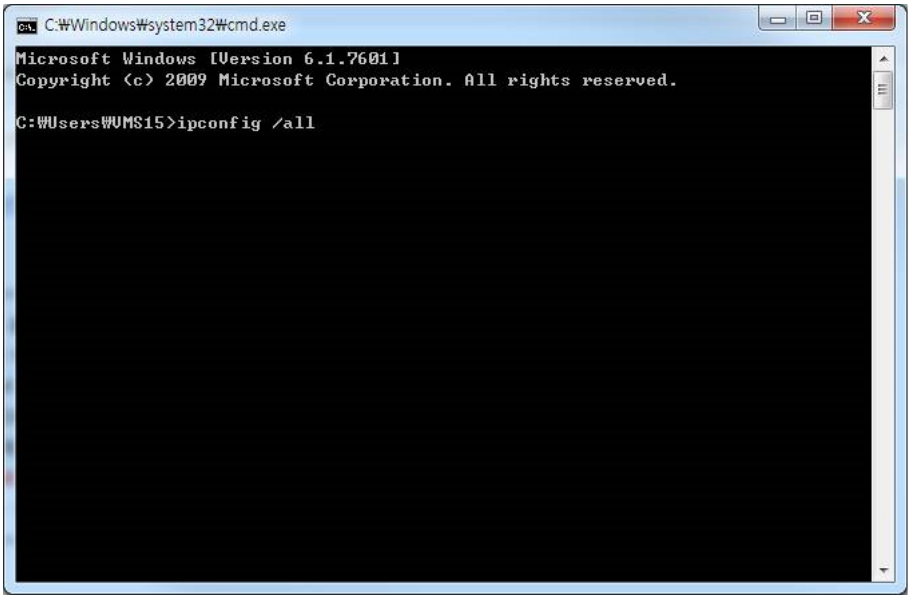
라이선스 키 신청에 대한 문의 사항 또는 문제 발생 시 support@vms-solutions.com으로 메일 보내주시기 바랍니다.

MAC 주소 확인 방법

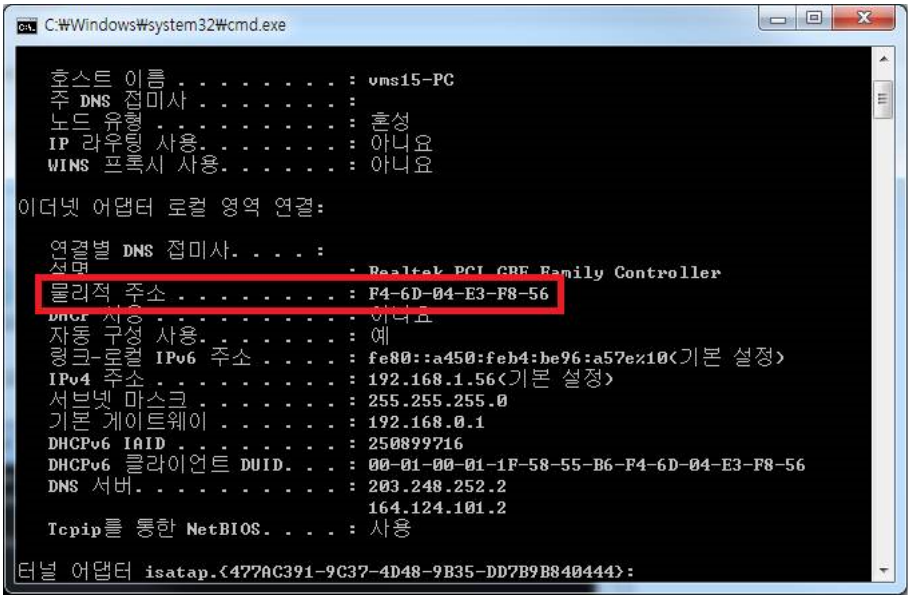
1. 키보드의 **윈도우 키 + r** 를 눌러 실행창에 "**cmd**" 입력 후 콘솔 창을 실행합니다.



2. 커맨드 창에 "**ipconfig /all**" 입력 후 Enter 키를 누릅니다.



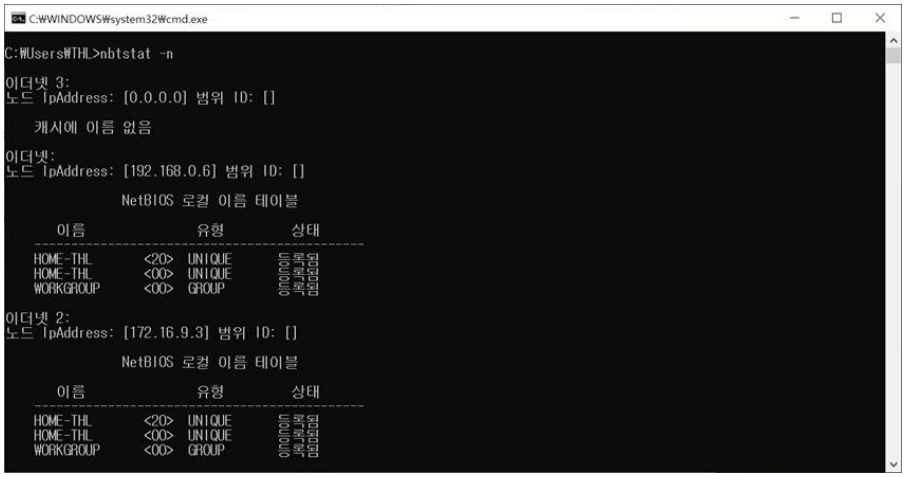
3. 명령을 실행 후 아래 그림과 같이 “물리적 주소”가 MAC 주소에 해당이 됩니다. LAN 카드가 두 개 있는 경우 네트워크에 물려 있는 LAN 카드의 MAC 주소를 보내주시면 됩니다.



컴퓨터 이름 확인 방법

라이선스 키 신청에 필요한 컴퓨터 이름은 아래와 같이 진행하면 확인이 가능합니다.

1. Windows + R 키를 눌러 실행창에 "**cmd"**라고 입력하여 명령 프롬프트를 실행합니다.
2. 명령창에 “**nbtstat -n**”을 입력 후 Enter 키를 누릅니다.
3. **[유형]**이 **UNIQUE**이고 **<00>** 접미사가 붙은 **[이름]**이 라이선스 인증 요청 시 전달되어야하는 컴퓨터 이름입니다. (예: HOME-THL)



[그림 1. nbtstat 실행 결과]

컴퓨터 이름에 관하여

컴퓨터 이름은 윈도우 운영체제를 설치하는 과정에서 지정 할 수 있으며, 설치 후에도 이름을 변경 할 수 있습니다. 여기서 컴퓨터 이름은 UNIQUE 유형에 해당됩니다. 컴퓨터 이름은 최대 16바이트까지 지정할 수 있으며 15번째 바이트까지는 컴퓨터 이름에 해당되며 마지막 바이트는 제공되는 서비스의 접미사입니다. 즉, 컴퓨터 이름은 1자~15자까지만 작성할 수 있습니다. **컴퓨터 이름이 15자를 넘어가면 Mozart 제품 라이선스가 인증 되지 않으니 꼭 유의 하시기 바랍니다**.

[그림 1.]과 같이 <00>,<20>이 접미사이며 <00>은 컴퓨터 이름, <20>은 서버 서비스의 이름을 뜻합니다. 해당 PC에서 파일과 프린터 공유를 활성화 한 경우 <20> 접미사가 붙습니다.

컴퓨터 이름은 각 컴퓨터에 고유하게 사용되어야 하며 동일 네트워크 상의 다른 컴퓨터의 이름과 중복되면 충돌이 발생합니다. 만약 충돌 발생 시에는 윈도우 제어판 또는 윈도우 설정을 통해 컴퓨터 이름을 변경해야 합니다.

참고자료:

컴퓨터, 도메인, 사이트 및 OUS 이름 지정 - Windows Server

이 문서에서는 Active Directory 디렉터리 서비스에 정의된 Windows, NetBIOS 도메인 이름, DNS 도메인 이름, Active Directory 사이트 및 US(조직 구성 단위)의 컴퓨터 계정에 대한 명명 규칙에 대해 설명합니다. 다음에 적용됩니다. Windows Server 2012 R2 원래 KB 번호: 909264 이 문서에서는 다음 항목에 대해 논의합니다.

https://docs.microsoft.com/ko-kr/troubleshoot/windows-server/identity/naming-conventions-for-computer-domain-site-ou

제어판에서 컴퓨터 이름 변경 방법

1. **[제어판]** → **[시스템]**으로 이동합니다.
2. **[컴퓨터 이름, 도메인 및 작업그룹 설정]**에서 **[설정 변경]**을 선택합니다.
3. **[컴퓨터 이름]** 탭에서 **[변경]**을 선택하여 **[컴퓨터 이름/도메인 변경]** 팝업창에서 **[컴퓨터 이름:]**에 변경할 이름을 입력합니다.
4. **[OK]**를 눌러 변경사항을 저장하고, 컴퓨터를 재시작합니다.

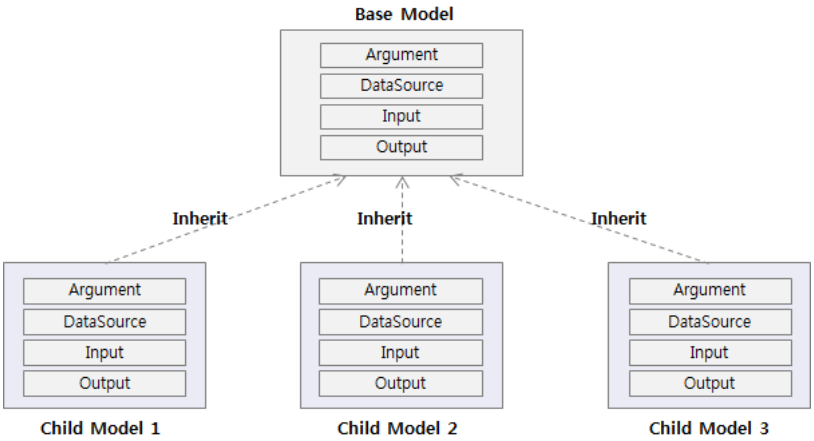
윈도우 설정에서 컴퓨터 이름 변경 방법

이 방법은 윈도우 10, 윈도우 서버 2016 이상부터 사용 가능합니다.

1. Windows + X를 눌러 **[시스템(Y)]** 메뉴를 선택합니다.
2. **[디바이스 사양]** 항목에서 **[이 PC의 이름 바꾸기]**를 선택합니다.
3. 변경할 컴퓨터 이름을 입력하고 **[다음]**을 선택합니다.
4. 컴퓨터를 재시작합니다.

다중 Model 프로젝트

MOZART Project 를 구성하는 요소 중 Model 은 복수 개를 생성할 수 있습니다. 단 첫 번째로 추가된 Model 이 Base Model 이 되며, 이후 추가되는 Model 은 Base Model 을 상속 받아 생성됩니다. 아래 그림은 Base Model 과 Child Model 의 관계를 도식화 합니다.



Scheduling 시스템을 라인별로 구축하게 되는 경우 기본적인 Inputs, Outputs 에 대한 구조가 거의 유사하고 Data Source 나 Arguments 등 일부만 차이가 나게 됩니다. 이런 경우에 초기 구축 라인의 Model 은 Base Model 로 하고 이후 Model 을 상속을 통해 만들 수 있습니다. 상속을 받은 Child Model 의 경우 Base Model 과의 관계를 유지하기 위해 편집에 제약이 발생합니다. 다음은 Model 을 구성하는 항목 별 편집 제약에 대해 설명합니다.

Child Model의 편집 제약사항

제약에 대한 대상은 모두 Base Model 의 항목을 대상으로 합니다. Base Model 의 항목에 대한 삭제는 기본적으로 수행이 불가하며, 항목 별 신규 Item 의 추가는 가능합니다. 다만 수정이 필요한 경우, 항목에 따라 허용하는 범위가 다르게 설정되어 있습니다. 항목 별 상세 내용은 아래와 같습니다.

Arguments : Input Arguments 와 Config Arguments 에 대한 편집 제약입니다.

- Arguments 추가 : O
- Arguments 수정 : X
- Arguments 삭제 : X
- Config Arguments 의 값 Mapping Data : X

Database

- Database 추가 : O
- Database 명 변경 : X
- Database 연결정보 변경 : O
- Database 연결정보 추가 및 순서 조정: O
- Database 삭제 : X

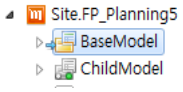
Inputs, Outputs

- Schema 추가 : O
- Schema 수정 : X (속성추가, 변경)
- Schema 삭제 : X
- DataAction 추가 : O
- DataAction 수정 : O (DataAction 이름 변경 외 Query 등 모든 부분 수정 가능)

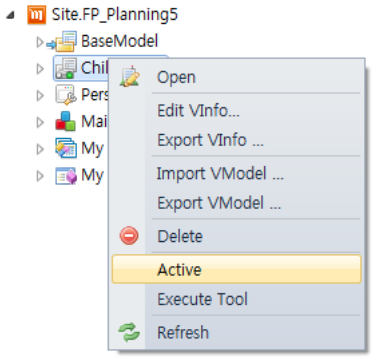
- DataAction 이름변경 : X
- DataAction 삭제 : X

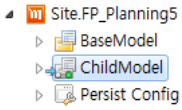
Child Model의 추가 및 활성화

Child Model 은 Model 추가 방식과 동일하게 MOZART Project 의 최상위 노드에서 **[Add > VModel]** 메뉴를 실행하여 추가합니다. Model 명칭은 Base Model 과 달라야 하며, 아래 그림과 같이 Base Model 과 Child Model 의 아이콘이 다릅니다.

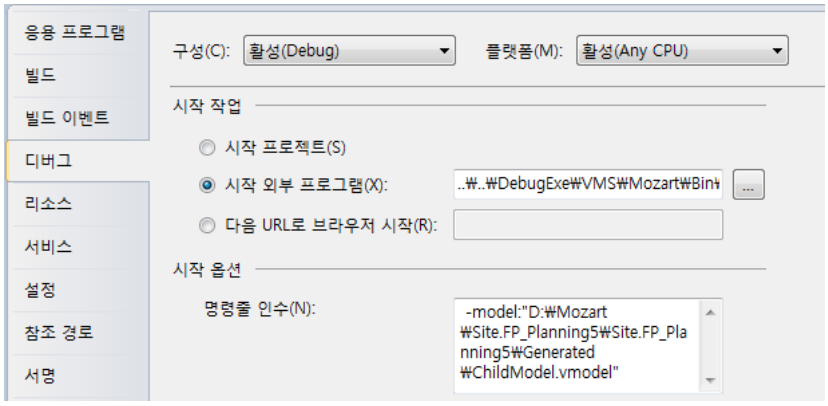


상기 Base Model 표시 아이콘에 파란색 화살표는 현재 활성화된 Model 을 의미합니다. 활성화된 Model 은 Debugging 시에 Project 에서 사용하는 Model 입니다. 사용자가 Debugging 시점에 사용할 Model 을 설정하기 위해서는 활성화 대상 Model 을 선택한 후 팝업메뉴에서 **[Active]** 메뉴를 실행합니다. 실행하면 대상 Model 의 아이콘에 파란색 화살표로 활성화된 상태를 표시합니다.

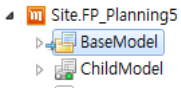




활성화된 Model 정보는 Project 속성 정보에 실행 Model 로 설정됩니다. 아래 그림의 명령줄 인수에서 Model 이 활성화된 Model 로 자동 설정됩니다.



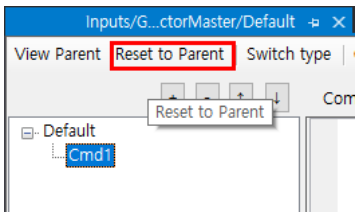
초록색 점의 의미



초록색 점

Base Model 과 Child Model 이 있을 때 Child Model 은 Base Model 을 참조하여 생성되어 있습니다. 만약 Child Model 이 수정되면 재정의(Override) 되었다는 표시로 초록색 점이 생성되게 됩니다.

DataAction 의 경우 아래 **Reset to Parent** 기능을 통해 다시 Base Model 의 값으로 복원이 가능합니다.

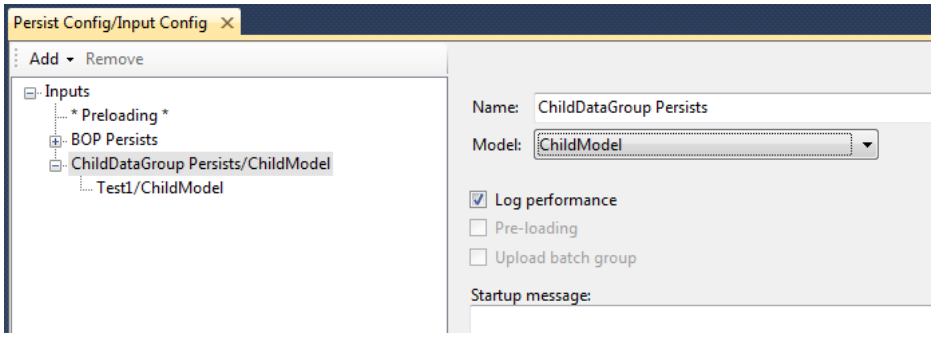


Model 별 Logic 작성

시뮬레이션 Logic 개발 시에 Model 별로 다른 Logic 을 적용해야 하는 경우 혹은 특정 Model 의 Arguments 를 사용하여 Logic 을 개발할 경우에 Model 을 참조하거나 설정하는 방식을 설명합니다.

Persist Config 에서 Model 별 Logic 구성 방법

Persist Group 별로 Model 을 지정할 수 있습니다. 아래 그림과 같이 Model 설정 콤보박스에서 Model 을 선택하면 해당 Model 에서만 Data 를 로딩하게 됩니다. 개별 Data 를 로드할 때 Model 별로 처리하기 위해서는 하단의 Logic 구현 시 Model 확인 방법을 통해 사용자가 Model 을 구분해서 구현합니다.



Logic 구현시 Model 을 확인하는 방법

Model Context 객체의 Vmodel 명 속성을 사용하여 현재 실행 중인 Model 을 확인합니다. VModel 명은 Tree 에 표시된 Model 표시명과 같습니다. 아래는 사용 예제입니다.

```
if (ModelContext.Current.VModelName == "ChildModel")
{
    // Child Model 에 대한 처리 Logic 구현
    // 예: 특정 Model 에 대해서만 Data 를 로딩 혹은 로딩 제외
}
```

Project 에 추가된 모든 Model 파일은 **[Project Root Folder /Generated]** 폴더에 Model 명.vmodel 파일로 생성됩니다. 배포 시에는 Base Model 과 Child Model 의 배포 방식에 차이가 없습니다.

Arguments 관리

Arguments 는 실행 시 사용되는 Logic 실행의 옵션 및 설정값을 정의하는 기능입니다.

MOZART 의 Arguments 는 로직 구현 시점에 사용하기 위해 정의할 수 있는 입력 파라미터입니다. 용도와 사용 방식에 따라 다음 두 가지 방식으로 정의하여 사용할 수 있습니다.

- Input Arguments : 실행 패널에서 조정할 수 있는 입력 파라미터입니다. 실행 전에 실행 조건 설정 화면에서 사용자가 설정합니다.
- Config Arguments : 입력 데이터와 바인딩하여 사용할 수 있는 입력 파라미터입니다. 실행 시점에 입력 데이터를 통해 파라미터 값이 설정됩니다.

Input Arguments

Task 실행 시 입력 Parameters 로 받아오는 Arguments 로 Module 실행 전체에서 사용되는 Context 와 상관없이 동일한 Parameters 값으로 사용되는 Arguments 를 정의합니다. 주로 Logic 의 실행 옵션에 대한 Parameters 로 사용합니다. 예를 들어 “장비 PM Schedule 반영여부”, “수율적용여부” 와 같은 옵션 설정 시 주로 사용합니다. Input Arguments 의 경우 개발 시 MOZART Studio 의 Experiment 설정 시 설정하거나 Server 의 Job, Trigger 등록 시 Arguments 로 설정할 수 있습니다.

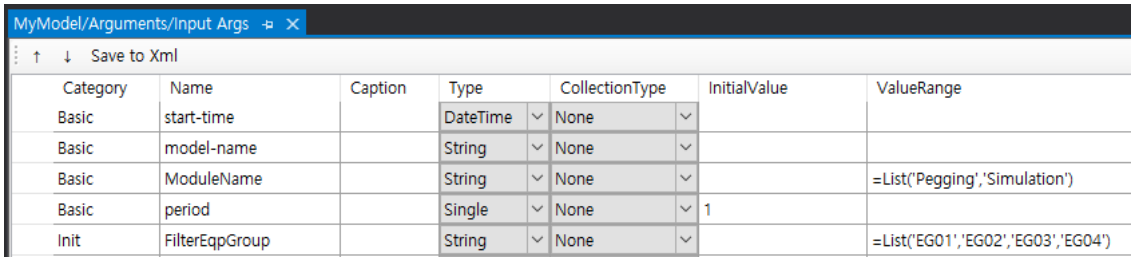
Input Arguments 의 경우 시스템에서 사전 정의된 Arguments 를 등록하여 Model 구동방식을 제어할 수 있습니다. (참조: **System Argument**)

Arguments 기본 설정방법

Arguments 는 아래와 같은 절차로 설정합니다.

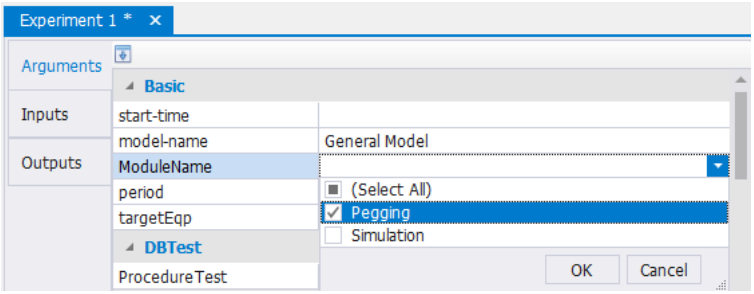
1. MOZART Explorer 의 등록된 Model 을 선택합니다.
2. 하위의 Arguments 노드를 펼치면 Input Args, Config Args 두 개의 노드가 있으며, 두 가지 중 등록이 필요한 Arguments 를 선택하여 더블 클릭합니다.
3. Arguments 를 등록할 수 있는 창이 열리면, 그리드의 Cell 별로 등록할 값을 입력합니다.
   - **Category** : Argument Group 으로 MOZART Studio 나 MMC 의 Job, Trigger 등록화면에서 Category 단위로 Argument 를 그룹핑하여 표시하게 됩니다.
   - **Name** : Argument 명
   - **Caption** : MOZART Studio 에서의 Argument 표시 명
   - **Type** : Argument Data 타입
   - **Collection Type** : Argument 를 집합변수로 생성할 수 있습니다. 사용가능한 형식은 List, Set, Dictionary 가 있으며, Dictionary 의 경우 Key 는 string 형식으로 정의됩니다.
   - **InitialValue** : Argument 의 기본 값
   - **ValueRange** : Argument 의 설정 값이 특정 범위의 값으로 제약될 때 해당 값의 범위를 입력합니다. 입력된 값의 Range 는 MOZART Studio 나 MMC 의 Parameter 입력 시, 콤보박스 형태로 표시하여 입력 받을 수 있습니다. Data 입력을 아래 그림과 같이 한 경우, MOZART Studio 의 [Experiment / Arguments] 입력창에서 해당 Argument 에 대해 리스트 에서 선택하여 입력할 수 있습니다. 체크 박스에서 여러 개의 값이 체크된 경우 하나의 string 값으로 들어옵니다.
   - **Description** : Argument 설명

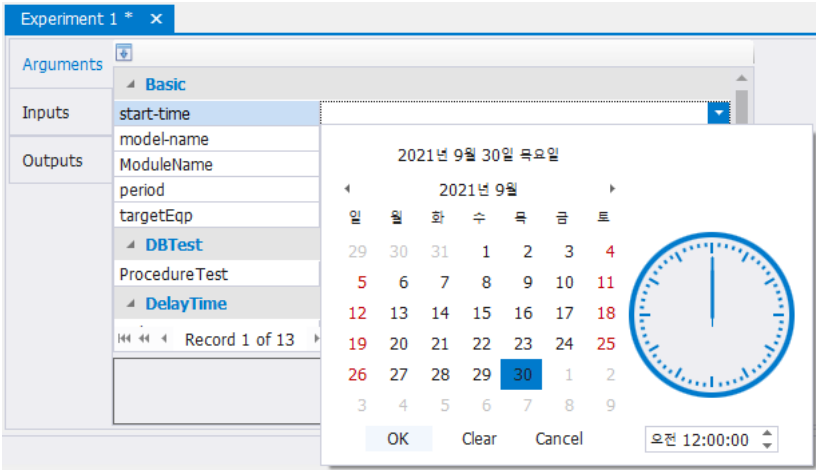
   [입력 예]



Category	Name	Caption	Type	CollectionType	InitialValue	ValueRange
Basic	start-time		DateTime	None		
Basic	model-name		String	None		
Basic	ModuleName		String	None		=List('Pegging','Simulation')
Basic	period		Single	None	1	
Init	FilterEqpGroup		String	None		=List('EG01','EG02','EG03','EG04')

[Experiment / Arguments 입력화면 예]





4. 입력된 내용을 저장합니다. (**Ctrl + S** 혹은 상단 메뉴바의 [저장] 버튼 사용)
5. Argument 가 입력/수정되는 경우 MOZART Explorer 의 Arguments 왼쪽에 수정 여부가 표시됩니다.
6. MOZART Explorer 에 Focus 가 있는 상태에서 저장을 하면 Model 이 저장됩니다.
7. Model 이 정상적으로 저장되면 InputMart 를 통해 설정한 Argument 를 사용할 수 있습니다.

Input Arguments 참조방법

입력된 Arguments 의 값은 InputMart 를 통해 사용할 수 있습니다. Input Arguments 의 경우, InputMart 의 GlobalParameters 를 통해 참조할 수 있으며, 참조하는 방법은 아래의 예제 소스를 참고하시기 바랍니다.

Input Arguments 값 참조

```
// Input Arguments 로 SourceFolder, DestinationFolder 를 정의한 경우
// 아래와 같은 방식으로 참조 가능
string source = InputMart.Instance.GlobalParameters.SourceFolder;
string destination = InputMart.Instance.GlobalParameters.DestinationFolder;
```

Config Arguments

Input Arguments 와 유사하나 Config Arguments 는 Module 실행 시 Context 에 따라 달라지는 경우에 주로 사용합니다. 예를 들어 기본 공정이동 시간이나, 기본 Setup 시간 같은 경우 별도의 Input 에 의해 값이 처리될 수 있으나 Config 로 처리할 경우 라인별 공정 이동 시간을 달리하거나, 장비 그룹 별 Setup 시간을 달리 설정해야 하는 경우 Config Arguments 를 사용합니다. Config 의 값은 Input Data 를 만들고 이를 연결하여 사용하는 방식입니다. Config Arguments 설정방법에 대한 자세한 사항은 **Config Arguments 의 사용방법**을 참조하시기 바랍니다.

Config Arguments 참조방법

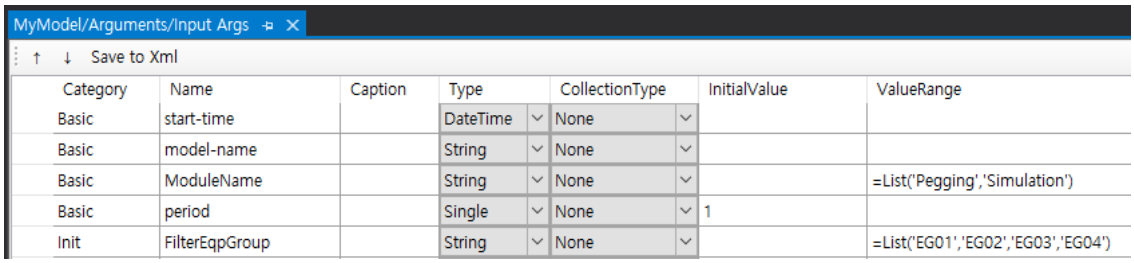
입력된 Arguments 의 값은 InputMart 를 통해 사용할 수 있습니다. Config Arguments 의 경우, InputMart 의 GetConfigParameters 를 통해 참조할 수 있으며, 참조하는 방법은 아래의 예제 소스를 참고하시기 바랍니다.

Config Arguments 값 참조

```
// Config Arguments 로 McGrpSetupTime 을 정의하고
// 해당 Config 값이 "EQPGRP1,60 ; EQPGRP2,45" 와 같을 때
ConfigParameters conf = InputMart.Instance.GetConfigParameters(string.Empty);
string[] grps = conf.mcGrpSetTimes.Split(';');
```

System Arguments 종류

Config Arguments 설정방법



Category	Name	Caption	Type	CollectionType	InitialValue	ValueRange
Basic	start-time		DateTime	None		
Basic	model-name		String	None		
Basic	ModuleName		String	None		=List('Pegging','Simulation')
Basic	period		Single	None	1	
Init	FilterEqpGroup		String	None		=List('EG01','EG02','EG03','EG04')

Data Source 관리

MOZART Project 에서 사용할 수 있는 Data Source 를 등록하고 관리하기 위한 기능입니다. 한 번 등록된 Data Source 는 Inputs/Outputs Data 를 정의하고 DataAction 을 생성하기 위한 Data Source 로 사용됩니다. MOZART 의 Data Source 는 복수 개의 연결 문자열 정보를 포함할 수 있도록 구성되며, 첫 번째 Data 연결 문자열이 동작하는 방식을 사용합니다. 필요 시 코드 내부에서 Data Source 의 연결문자열 명으로 Data Source 를 지정할 수 있습니다.

Data Source 관리방법

1. MOZART Explorer 의 Database 노드를 선택합니다.
2. Database 노드에서 오른쪽 마우스 버튼을 클릭하여 **[New DataSource…]** 팝업메뉴를 실행합니다.
3. Edit Data Source 다이얼로그에서 Data Source 명을 입력한 후, **[Add]** 버튼을 사용하여 연결 문자열을 등록합니다.
4. Data Source 다이얼로그에서 Data 연결 문자열 명을 입력한 후, Data Provider 콤보박스에서 대상이 되는 Data Provider 를 선택합니다.

   > **Note**
   > 지원대상이 되는 Provider 는 Microsoft Access, Microsoft SQL Server, Oracle, DB2, SQLite, MySql, ProgreSQL 일곱 가지입니다.

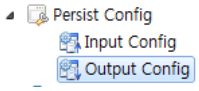
5. Data 연결 문자열을 입력합니다. 문자열은 하단의 텍스트박스에 직접 입력하거나, […] 버튼을 이용하여 Data Source 별 입력창을 통해 입력할 수 있습니다.

- **[…]** 버튼을 클릭하면 Select database 다이얼로그가 열립니다. "**Select a database**" 콤보박스에서 기본에 등록된 Database 를 선택할 수 있으며, 신규로 추가할 경우에는 <**New Database Connection**>을 선택합니다.
- 신규로 추가하는 경우 하단의 Select a database type 콤보박스에서 database type 을 선택한 후 **[Create…]** 버튼을 클릭합니다.
- Database Type 에 따른 입력창을 통해 접속 정보를 입력하면 Data Source 가 추가됩니다.

6. 연결 문자열이 등록되면 Data Source 다이얼로그 하단 리스트에 등록된 모든 연결 문자열이 표시됩니다.
7. 연결 문자열을 삭제할 때에는 삭제 대상인 문자열을 리스트에서 선택하고 **[Remove]** 버튼을 사용하여 삭제할 수 있습니다.
8. 연결 문자열 편집 시에는 해당 문자열을 리스트에서 더블 클릭하거나 선택 후 **[Edit]** 버튼을 사용합니다.
9. Data Source 에 여러 문자열이 있는 경우, 특별한 설정을 하지 않으면 첫 번째 문자열이 적용됩니다. 적용되는 문자열을 변경하고자 하는 경우 **[Up][Dn]** 버튼을 사용합니다.

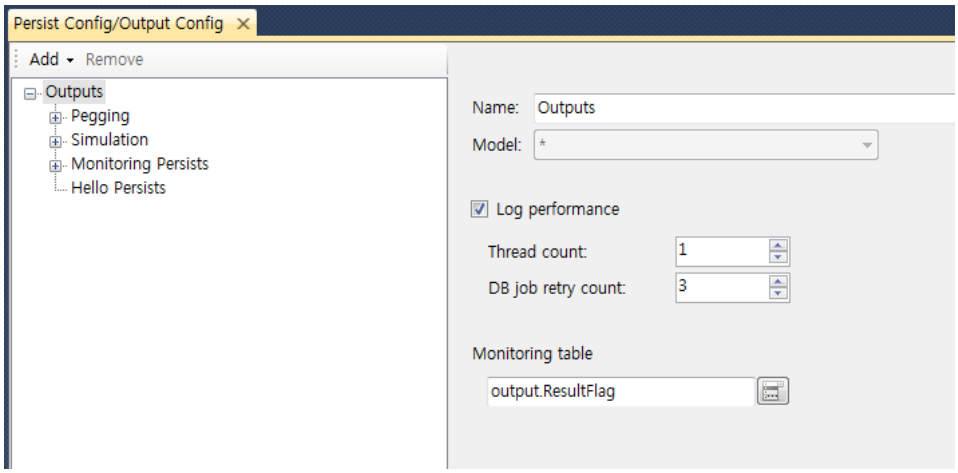
Output Persist Configuration 정의

output persist configuration 에서는 Memory 상의 Output DataItem 을 File, DB 에 저장하는 작업절차와 방식을 정의합니다. 일련의 과정을 진행하기 위해 Output Persist 를 정의하는 구성요소는 Output Data Group, Output DataItem, Output Save Log 의 세 가지 유형으로 나눌 수 있으며, 전체 Output 을 처리하기 위한 옵션도 Persist Config 를 통해 설정할 수 있습니다. 실행 메뉴는 [Persist Config > Output Config] 입니다.



Output Data 전체 저장 옵션 설정

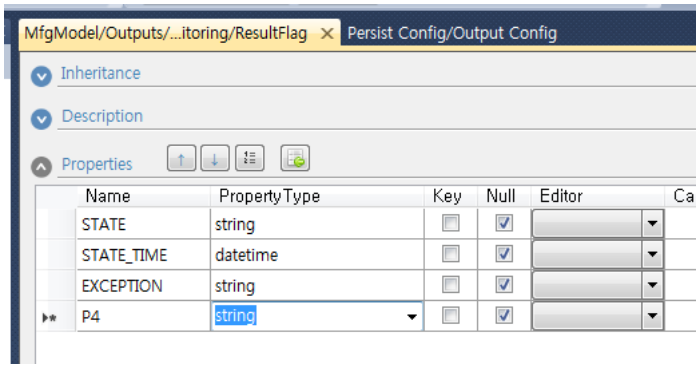
최상위 Tree 노드를 선택하면 Data 저장을 위한 옵션을 정의할 수 있는 메뉴가 활성화됩니다.



- **Name** : Outputs 전체에 대한 Log
- **Log performance**: 전체 Data 저장 소요시간을 기록할지 여부를 설정합니다.
- **Thread count: Batch Upload** 를 수행하는 경우 사용할 Thread 수를 지정합니다.
- **DB job retry count**: DB 연결, 명령의 재실행 횟수를 지정합니다.
- **Monitoring table** : 전체 Task 의 수행 종료 시점에 Task 수행 성공/실패 여부를 기록하기 위한 Output 을 정의할 수 있습니다. Output 을 설정한 후 Main Control 의 ShutDown Action 을 사용하여 결과를 기록할 수 있습니다.

Monitoring 사용 예시

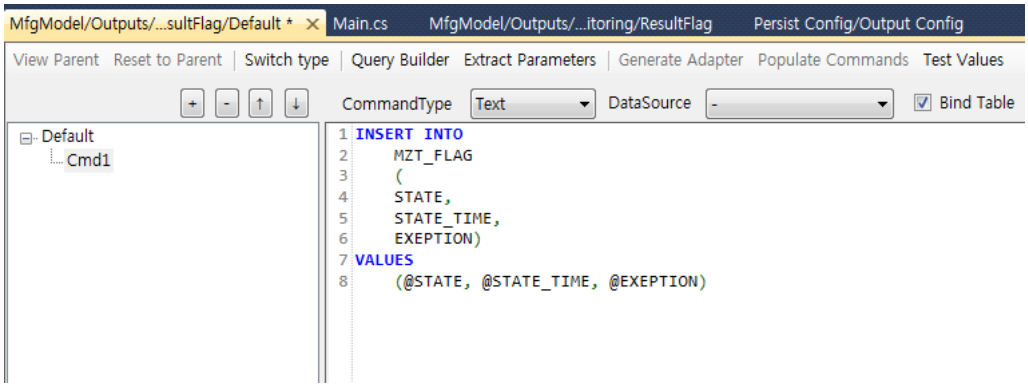
1. Output 결과 기록을 위한 Output DataItem 을 정의합니다.



2. output persist config 에 Monitoring Table 속성에 결과 기록을 위한 Output DataItem ('1'에서 정의한 Item) 을 지정합니다.
3. Main Control 의 ShutDown Action 을 구현합니다.

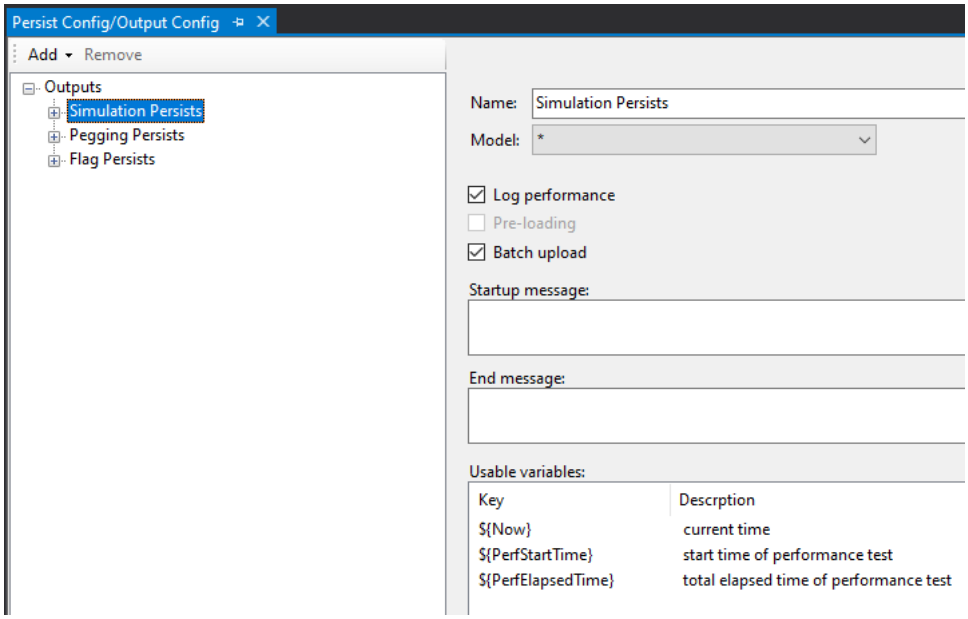
```
public void SHUTDOWN_0(ModelTask task, ref bool handled)
{
    ResultFlag result = new ResultFlag();
    result.STATE_TIME = DateTime.Now;
    result.STATE = task.HasErrors ? "FAIL" : "SUCCESS";
    result.EXCEPTION = task.Exception.Message;

    OutputMart.Instance.ResultFlag.Add(result);
}
```

4. 결과 Output 에 대한 DataAction 설정을 통해 Monitoring 용 DB 에 결과를 기록합니다.



Output Data Group 저장 옵션 설정

Data Group 별 저장 옵션을 정의할 수 있습니다. Tree 메뉴에서 Data Group 을 선택하면 우측의 속성창을 통해 속성 설정이 가능합니다.



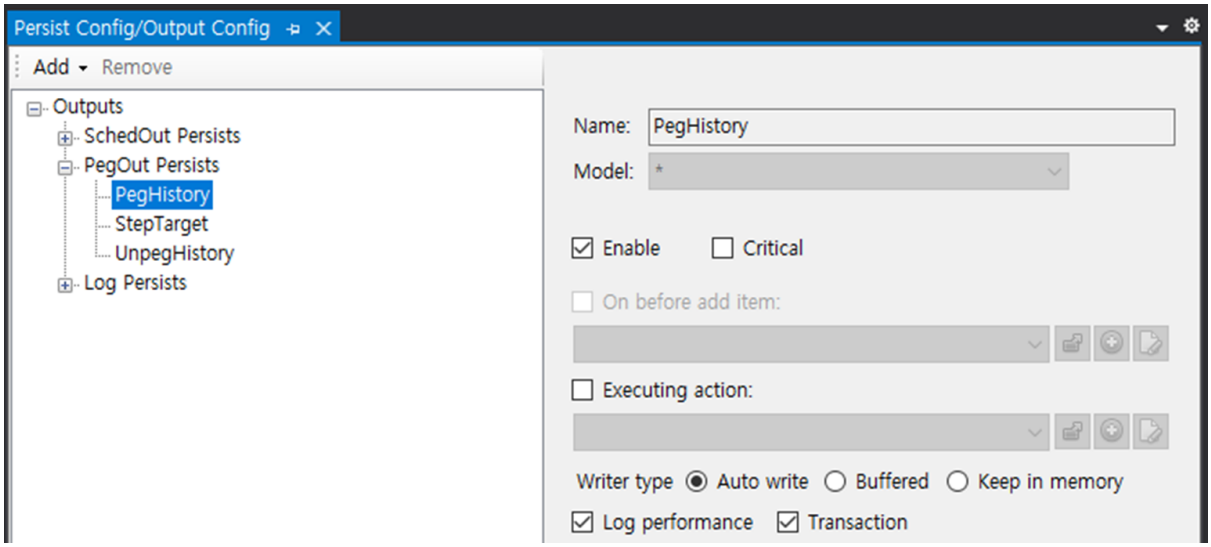
- **Name** : 로딩 시 사용할 Group 명입니다.
- **Log performance** : 전체 Data 저장 소요시간을 기록할지 여부를 설정합니다.
- **Batch upload**: Thread를 할당해서 다른 Group의 테이블과 동시에 DB Save를 수행할지 여부입니다. 예를 들어 Data Group A, B 에 Batch upload를 체크하고 Thread Count가 2면, DB Save를 수행할 때 A Group의 테이블 하나와 B Group의 테이블 하나가 동시에 DB에 Write 합니다.
- **Startup message**: Group 에 대한 저장 시작 시점에 Log 에 남길 메시지를 편집합니다. 메시지 편집 시 하단의 Usable variables 를 사용할 수 있습니다.
- **End message**: Group 에 대한 저장 완료 시점에 남길 메시지를 편집합니다. 메시지 편집 시 하단의 Usable variables 를 사용할 수 있습니다.

Usable variable 사용예시

- **Log Message 입력** : This is test log start message at ${Now}
- **결과 출력** : This is test log start message at 2014-09-22 06:05:11

Output DataItem 저장 옵션 설정

개별 DataItem 별로 저장 옵션을 정의할 수 있습니다.



- **Name** : DataItem 명으로 변경이 불가합니다.
- **Enable**: Data 저장 여부를 설정합니다. 옵션을 체크하지 않은 경우 DataItem 이 파일로 저장되지 않습니다.
- **Executing action**: DataItem 의 Data 가 저장 된 후, 추가적인 Data 후처리를 하는 경우 사용됩니다. 해당 Output Data Table 의 Data 를 가공하여 별도의 Outputs 을 추가하는 등의 사용자 코드를 작성할 수 있습니다. 편집 방법은 Input Persist Config의 Action 메뉴 사용방법 과 같습니다.
- **Writer Type** : Outputs 의 저장방식을 설정합니다. 아래는 각 옵션별 저장 형태입니다.
  - **Auto write** : 기본 옵션, OutputMart 의 Data 가 1만 건 마다 Output File 로 Data 를 쓰고, 파일로 저장된 Data 는 Memory 상에서 삭제합니다.

Output Persist Write type	Output DataItem	Action	Output Mart	vdat Files	Output File 생성 후 Output Mart 상태
Auto write	Output A	Save	Output A	Output A.vdat Every 10,000 rows saves to file & delete 10,000 rows in Output Mart	Output A(Empty)

  - **Buffered** : Output 정의 시 Buffer 저장소를 하나 더 만듭니다. OutputMart 에 데이터가 저장 된 후에는 속성 값을 변경 할 수 없기 때문에, 버퍼에 결과 정보를 넣은 후, 결과의 속성 값을 변경하여 최종 OutputMart 에 저장하기 위한 용도입니다. Output Data 의 속성이 모두 확정되지 않은 경우 임시로 Data 를 Buffer 에 저장해 두었다가 속성이 확정되는 시점에 Output 으로 옮기고 Buffer 에서 삭제합니다. 시뮬레이션 종료시점에 Buffer 에 남아있는 Data 는 Output Data 로 저장되기 때문에 Buffer 에 대한 관리코드는 사용자가 직접 작성 해야 합니다. Output 으로 저장된 Data 는 Auto write 와 같이 1만개 단위로 Output file 에 저장한 후 Memory 상에서 삭제됩니다. 아래 Buffered 로 설정된 Output 에 Data 를 기록하는 예제코드를 참조합니다.

Output Persist Write type	Output DataItem	Action	Output Mart	vdat Files	Output File 생성 후 Output Mart 상태
Buffered	Output B	Validate & Verification Output Save	Output B Output B Buffer	Output B.vdat Use Buffer, must use RemoveBuffer	Output B(Empty) Output B Buffer(Empty)

  - **Keep in memory** : 전체 작업의 시작부터 종료시점까지 Memory 상에 유지되며, 중간에 Output File로 기록하지 않습니다.

Output Persist Write type	Output DataItem	Action	Output Mart	vdat Files	Output File 생성 후 Output Mart 상태
Keep in memory	Output C	Save	Output C	Output C.vdat	Output C

- **Transaction** : DataItem에 등록된 DataAction에서 등록된 Command 수행 시 Commit 시점을 지정하기 위한 속성입니다.
  - 예제. 하나의 DataAction에 Cmd1, Cmd2, Cmd3 이 존재할 때
    - True : Cmd1 → Cmd2 → Cmd3 → Commit 수행
    - False : Cmd1 → Commit 수행 → Cmd2 → Commit 수행 → Cmd3 → Commit 수행

Output DataItem 저장 옵션 설정(Buffered)

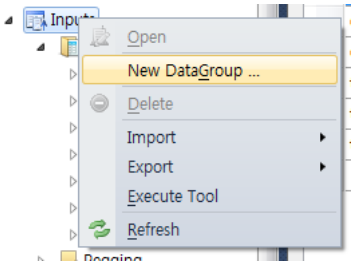
```
private void WriteEqpLoadingPlan(ILot lot, IHandlingBatch hb, AoEquipment eqp)
{
    SiteLot mLot = lot as SiteLot;
    string eqpID = eqp.EqpID;

    // Buffer 에 저장된 작업 시작 정보를 찾음
    // FindBuffer() 사용할 때 Output DataItem의 Schema의 key 값으로 Buffer를 찾음
    var plan = OutputMart.Instance.EqpLoadingPlan.FindBuffer(eqpID, mLot.LotID, mLot.CurrentStep.StepID);

    if (plan ==null)
    {
        plan = new EqpLoadingPlan();
        plan.EQP_ID = eqpID;
        plan.LOT_ID = mLot.LotID;
        plan.STEP_ID = mLot.CurrentStep.StepID;
        plan.START_TIME = mLot.CurrentPlan.StartTime;
        plan.PRODUCT_ID = mLot.Product.ProductID;
        plan.PROCESS_ID = mLot.Process.ProcessID;

        // 작업 시작정보가 없는 경우 Buffer 에 추가
        OutputMart.Instance.EqpLoadingPlan.AddBuffer(plan);
    }
    else
    {
        // 시작 정보가 있는 경우 종료 정보를 쓰고 최종 Output 에 추가함
        // Buffer 의 Data 를 삭제
        OutputMart.Instance.EqpLoadingPlan.RemoveBuffer(plan);
        plan.END_TIME = eqp.NowDT;
        plan.QTY = mLot.UnitQty;

        // Output Data 에 추가
        OutputMart.Instance.EqpLoadingPlan.Add(plan);
    }
}
```

Schema 편집 방법

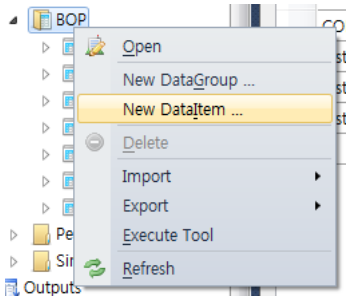
시스템에서 사용될 Input, Output Data 에 Schema 및 Schema 를 구성하는 Column 에 대한 속성을 설정하는 방법을 설명합니다. Input Data 의 경우 기본 Schema 편집 외에 추가적으로 InputMart 에 해당 Data 를 참조하기 위해 View 를 정의 할 수 있습니다. 아래는 두 가지 처리 방법을 설명합니다.

Schema 편집방법

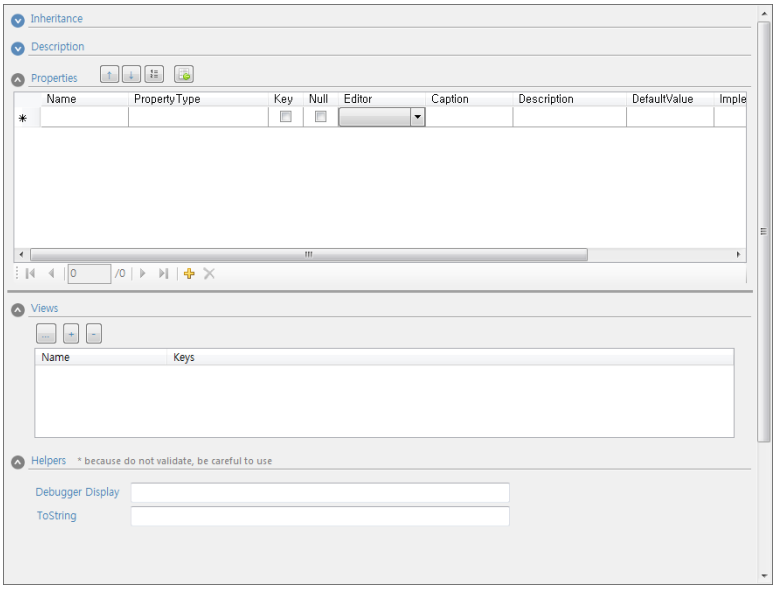
1. MOZART Explorer 의 **[Model > Inputs]** 를 선택하고 팝업메뉴에서 **[New DataGroup]**메뉴를 선택합니다.



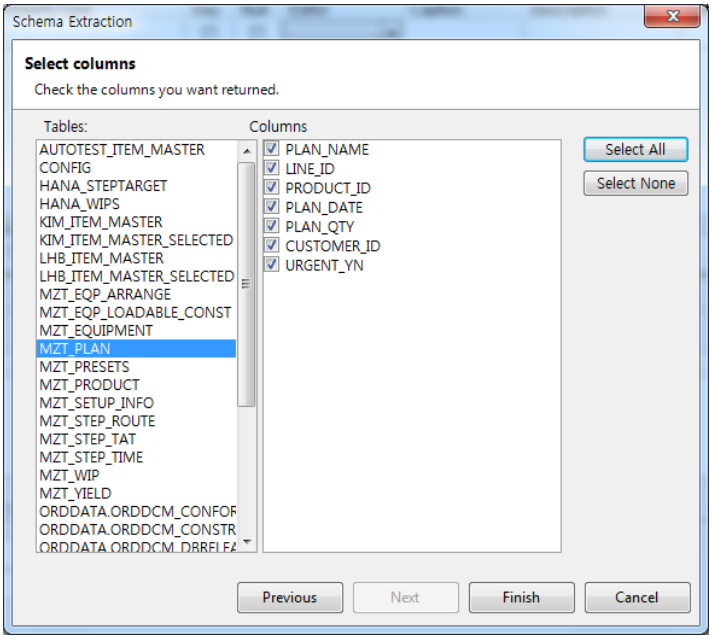
2. 새로 추가된 Data Group 이나 기존 Data Group 을 선택하고 팝업메뉴에서 [New DataItem]을 선택합니다.



3. DataItem 명을 입력하면 Tree 에 입력한 DataItem 이 생성되며 Schema 편집 화면이 열립니다.

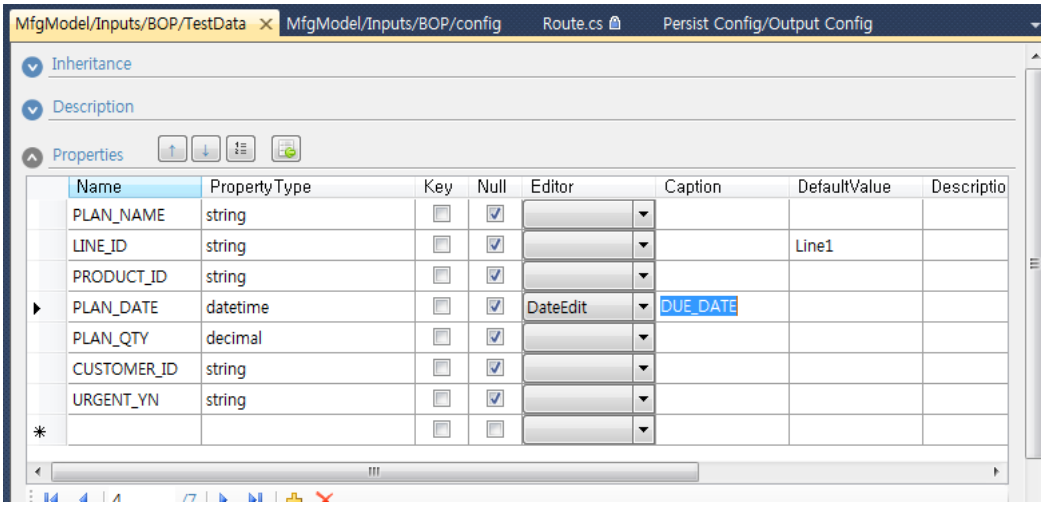


4. Data Column 을 입력하는데 Column 별로 수동 입력을 할 수도 있고, 특정 Data Source 의 Table 로부터 Column 을 가져와 Schema 를 생성할 수도 있습니다.
   - DB Table 로부터 생성하기 위해서는 상단의 [버튼이름] 버튼을 클릭합니다.
   - Schema Extraction Dialog 가 활성화되고 “Select a datababe” 콤보박스에서 기존에 연결정보 중 대상 Database 를 선택합니다.
   - 만일 대상 Database 가 없는 경우 신규로 연결정보를 입력합니다. (Datasource 입력 방식 참조)
   - Data 선택된 경우 **[Next]** 버튼으로 Table 및 Schema 에 사용될 Column 을 선택합니다.

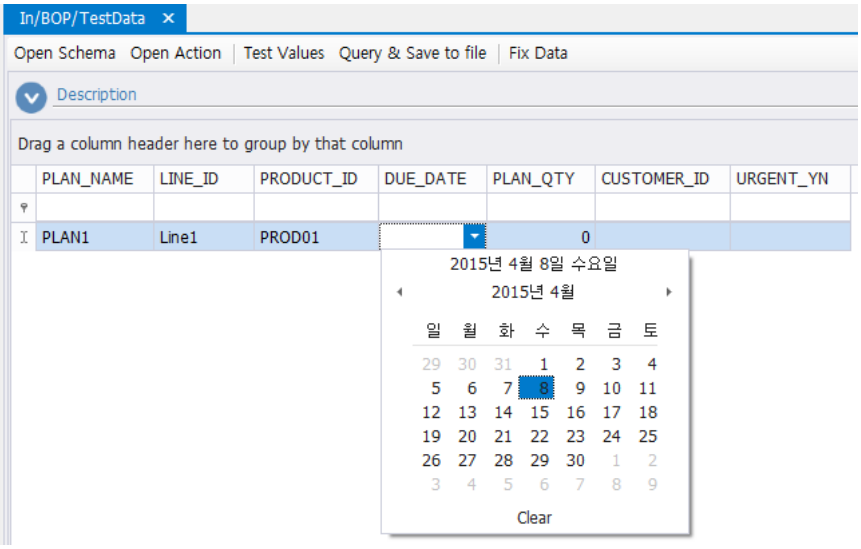


- Column 을 선택한 후 **[Finish]** 버튼을 클릭한 후 생성된 Schema 를 확인합니다.

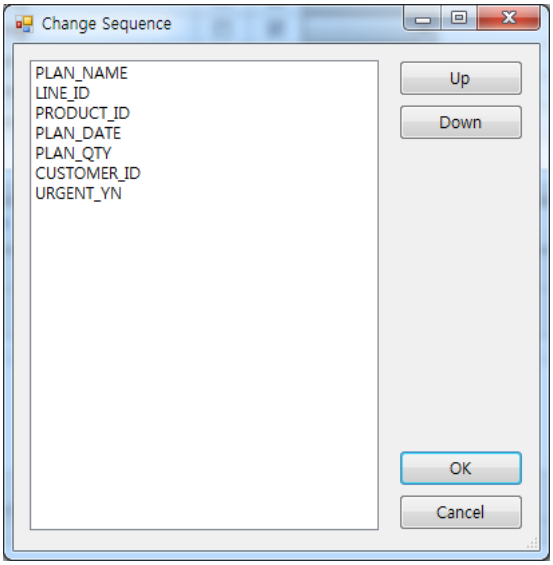
5. Data Column 의 추가적인 속성을 입력합니다.



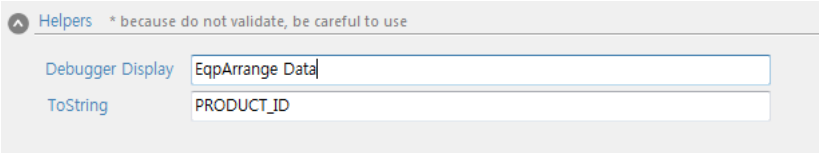
- **이름** : 열 명
- **PropertyType** : Column Data 타입
- **Caption** : MOZART Studio 상에서의 Column 표시 명입니다. 위 그림에서 PLAN_DATE 의 Caption 을 DUE_DATE 으로 설정한 경우 아래 그림과 같이 Studio 에서 그리드의 Column 명이 DUE_DATE 로 표시됩니다.
- **Description** : Column 설명
- **Key** : Column 의 Primary Key 여부입니다. 입력된 Data 에 대한 유효성을 체크합니다.
- **Null** : Column 의 Null 값 허용 여부입니다. 입력된 Data 에 대한 유효성을 체크합니다.
- **Editor** : MOZART Studio 의 Experiment 창에서 Data 그리드의 Column 값 입력시 입력 컨트롤로 유형은 {“TimeEdit”,”DateEdit”,”MemoEdit”,”ComboBox”}의 네 가지 유형이 있습니다. Editor 를 선택한 경우 Studio 에서 Data 를 입력할 때 선택한 Editor 가 활성화 됩니다. PLAN_DATE 에 대해 “DateEdit” 타입으로 선택한 경우 아래 그림과 같이 입력 시 날짜 선택 Dialog 가 활성화됩니다.
- **DefaultValue** : 기본값입니다. 상기 그림에서 LINE_ID 속성의 기본값을 “Line1” 으로 설정한 경우 아래와 같이 Data 입력 시 기본값으로 자동 입력됩니다.



6. 입력한 Column 의 순서조정이 필요한 경우, 속성 그리드의 위치 조정 대상 속성을 선택한 후 ⬆️ ⬇️ 버튼으로 위치를 조정하거나, [버튼이름] 버튼으로 순서조정 다이얼로그박스를 사용하여 조정할 수 있습니다. 아래와 같은 다이얼로그를 통해 리스트박스에서 순서조정이 필요한 Column 을 선택한 후, 우측 상단의 **[Up][Down]** 버튼을 사용하여 위치를 조정하고 완료합니다.



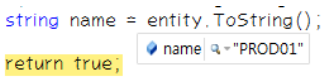
7. **Helper 설정** : Debugging 및 구현 시점에 Data 클래스에 대한 표시명 및 기본 문자열을 설정합니다.



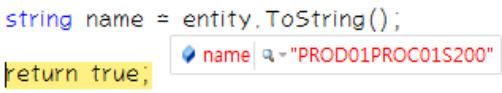
- **Debugger Display** : Debugging 시 변수에 대한 정보 표시 명입니다. 상기 이미지와 같이 설정하는 경우 Debugging 시 해당 객체는 아래 그림과 같이 표시됩니다.

> **Note**
> Schema 에 정의된 속성값을 표시하기 위해서는 {LINE_ID}{PRODUCT_ID} 와 같은 방식으로 속성값을 대괄호"{}" 로 감싸서 입력합니다. 그외에 입력된 문자는 위의 예와 같이 입력값이 문자열로 그대로 출력됩니다.

- **ToString** : Debugging 시점 및 로직 작성 시점에 객체의 .ToString() 함수를 사용할 때의 결과로 사용할 속성을 설정합니다. Data 의 속성명을 설정하거나 간단한 Expression 을 사용할 수 있습니다. 상기 그림과 같이 설정한 경우 아래 왼쪽과 같이 결과가 출력되며, ToString 속성을 "PRODUCT_ID+PROCESS_ID+STEP_ID" 와 같이 설정하는 경우 아래 오른쪽 그림과 같이 Expression 결과가 출력됩니다.



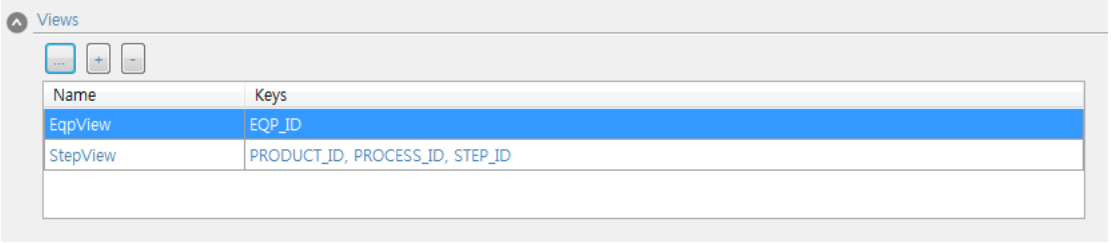
"PRODE_ID"로 설정한 경우



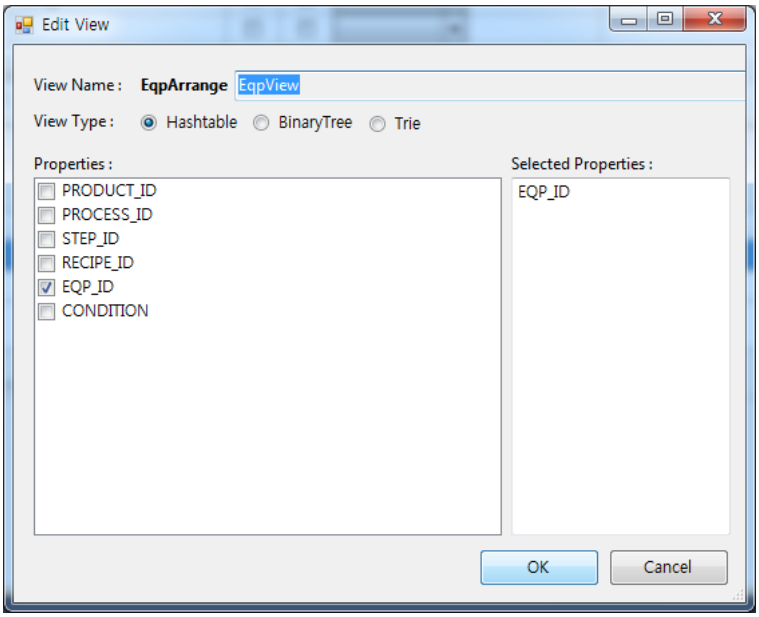
"PRODUCT_ID+PROCESS_ID+STEP_ID"로 설정한 경우

DataView 편집방법

Schema 로 정의된 Data 는 기본적으로 InputMart 에 EntityTable 형태로 자동으로 생성됩니다. Persist config 에서 "Use temporary context" 옵션을 설정한 경우 TempMart 에 생성됩니다. 시뮬레이션 수행 시 생성된 EntityTable 의 Data 를 사용할 때 검색 속도를 향상시키기 위해 참조되는 Key 에 따라 View 를 설정할 수 있습니다. 다음은 Data 의 View 를 설정하는 방법을 설명합니다.



1. DataItem 편집 화면 하단의 View 섹션에서 [...] 버튼을 클릭하면 View 설정창이 열립니다.



2. View 설정창에서 View 이름을 설정하고, View Type 을 선택한 후 해당 View 의 Key 항목을 선택한 후 [OK] 버튼을 눌러 저장합니다. Data 의 양이나 검색방식에 따라 View Type 을 선택합니다. Sorting 이 필요없이 검색만 수행하는 경우에는 HashTable type 을

주로 사용하며, Sorting 을 많이 해야 하는 경우에는 Binary Tree 를 사용하는 편이 보다 효율적입니다.

3. Lsit 에 입력한 View 가 생성됨을 확인합니다.
4. 해당 View 는 아래의 예제와 같이 InputMart 를 통해 참조할 수 있으며, EntityView 의 설정된 Key 로 빠른 검색이 가능합니다. 아래는 생성된 View 의 참조 및 활용에 대한 예제입니다.

View 사용 예제

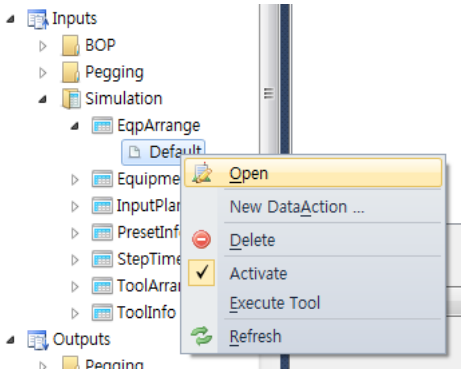
```
public IList<string> GETLOADABLEEQPLIST(DispatchingAgent da, IHandlingBatch hb, ref bool handled, IList<string> prevReturnValue)
     {
     SiteLot lot = hb.Sample as SimpleMfgSemiconLot;
     var arrs = InputMart.Instance.EqpArrangeStepView.FindRows(lot.Product.ProductID, lot.Process.ProcessID, lot.CurrentStepID);
     HashSet<string> loadables = new HashSet<string>();
     foreach (EqpArrange arr in arrs)
     {
         loadables.Add(arr.EQP_ID);
     }

 return loadables.ToList<string>();
 }
```

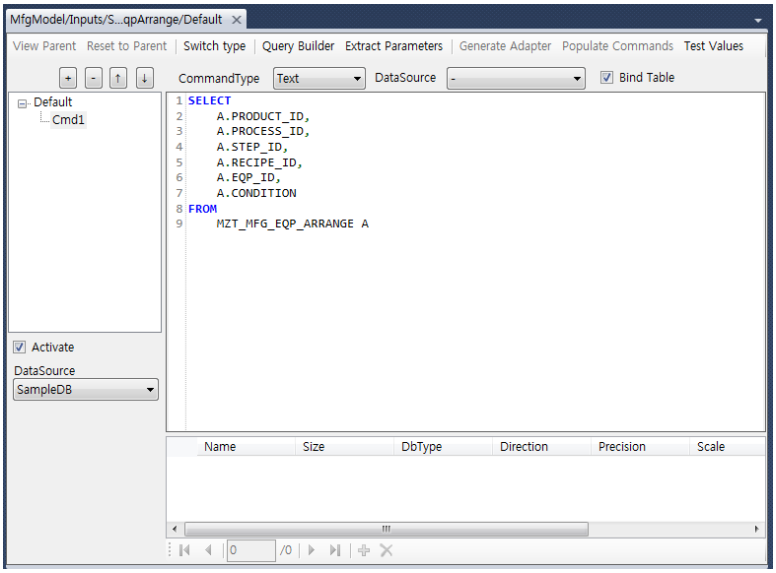
DataAction 편집 방법

DataAction 은 복수 개의 Query 로 구성될 수 있는 Query 관리 단위입니다. Input DataItem 에 Binding 되는 DataAction 의 경우, Input Schema 에 Data 를 Download 하는 역할을 하는 Query 이며, Output DataItem에 Binding 되는 DataAction 의 경우 해당 Output Data 를 Database 에 삽입하기 위한 Query 입니다. DataAction 은 다음의 절차로 추가, 수정, 삭제할 수 있습니다.

DataAction 편집방법

1. 대상이 되는 DataItem 을 선택하고 Tree 를 펼치면 “Default” DataAction 이 표시되며, 해당 노드를 더블 클릭하거나 팝업의 **[Open]** 메뉴를 클릭하면 DataAction 편집창이 열립니다.

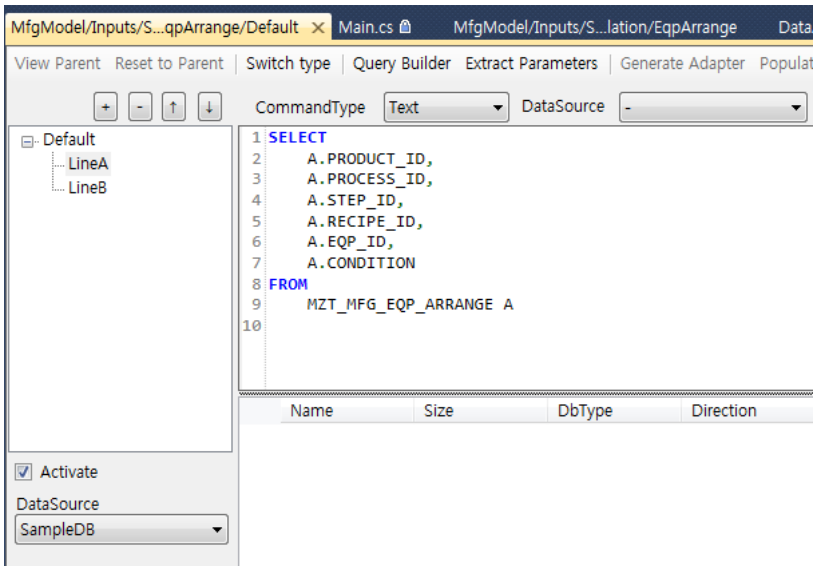




2. 만일에 기존의 DataAction 외에 신규 DataAction 을 추가하는 경우에는 해당 DataItem 을 선택하고 팝업메뉴의 **[New DataAction]** 을 사용합니다. 한 DataItem 에 DataAction 이 복수 개 등록된 경우에는 Activate 항목이 체크된 DataAction 이 실행됩니다.
3. DataAction 의 왼쪽 하단에 Data Source 콤보박스에서 DataAction 의 Query 에서 사용할 Data Source 를 선택합니다.
4. 한 DataAction 은 여러 개의 Query 문장을 저장할 수 있습니다. 편집창 왼쪽 Tree 에 표시된 최하위 노드가 개별 Query 입니다. 상단의 ➕ ➖ 버튼을 통해 Command 를 추가할 수 있으며, ⬆️ ⬇️버튼을 통해 실행 순서를 조정할 수 있습니다. Tree 에서의 순서대로 실행됩니다.

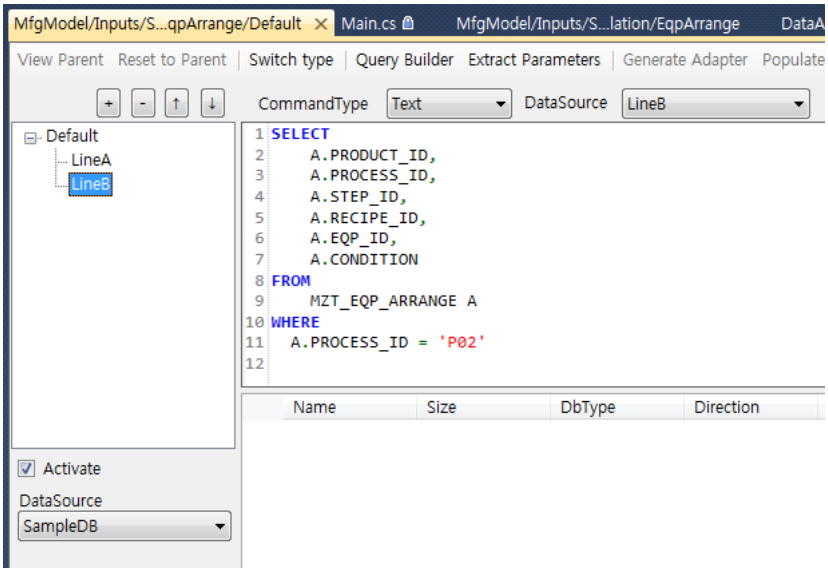
복수 Command 등록 예제

첫 번째 Command 는 SampleDB 에서 LineA 에 대한 Data 를 Query 하고, 두 번째 Command 는 LineB DB 에서 LineB 에 대한 Data 를 Query 하여 두 결과의 합집합으로 Input Data 를 처리하는 예제입니다.
[첫 번째 Command 등록]



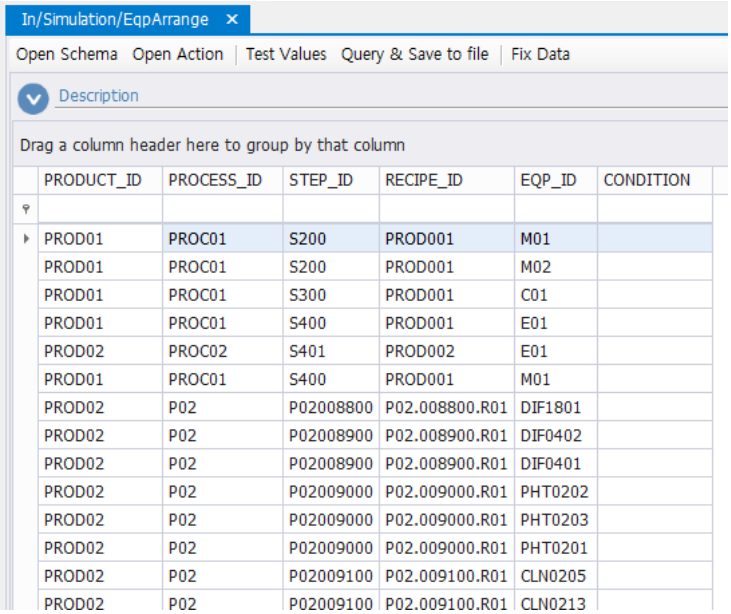
[두 번째 Command 등록]

Command 별 Data Source 를 두 번째 DB 에 대한 소스로 설정하고, Query 결과 Column 은 첫 번째 Command 와 동일하게 구성합니다.



[Query 결과]

[Query & Save to file] 메뉴로 실행하면 아래와 같이 두 Data 가 병합된 형태의 결과를 확인 할 수 있습니다



PRODUCT_ID	PROCESS_ID	STEP_ID	RECIPE_ID	EQP_ID	CONDITION
PROD01	PROC01	S200	PROD001	M01	
PROD01	PROC01	S200	PROD001	M02	
PROD01	PROC01	S300	PROD001	C01	
PROD01	PROC01	S400	PROD001	E01	
PROD02	PROC02	S401	PROD002	E01	
PROD01	PROC01	S400	PROD001	M01	
PROD02	P02	P02008800	P02.008800.R01	DIF1801	
PROD02	P02	P02008900	P02.008900.R01	DIF0402	
PROD02	P02	P02008900	P02.008900.R01	DIF0401	
PROD02	P02	P02009000	P02.009000.R01	PHT0202	
PROD02	P02	P02009000	P02.009000.R01	PHT0203	
PROD02	P02	P02009000	P02.009000.R01	PHT0201	
PROD02	P02	P02009100	P02.009100.R01	CLN0205	
PROD02	P02	P02009100	P02.009100.R01	CLN0213	

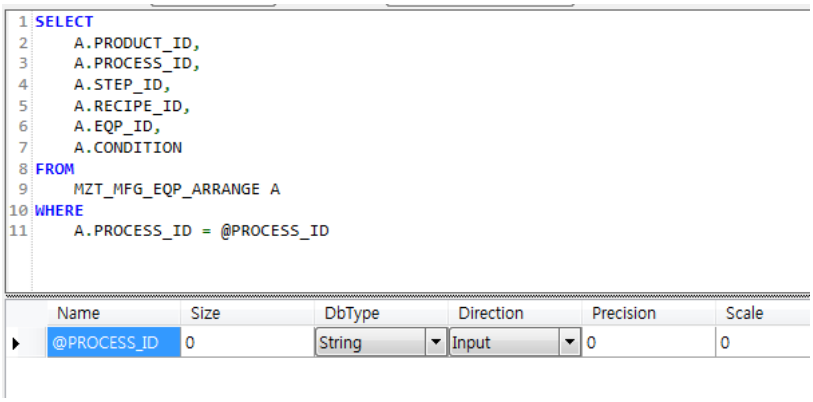
5. 오른쪽 편집창에 Query 를 입력합니다.

> **Note**
> 편집창 상단의 콤보박스 및 체크박스 옵션은 다음과 같은 설정을 지원합니다.

- **CommandType** : Query 편집창의 Query 종류 선택
  - **Text** : 일반 Query 문장을 입력합니다.
  - Stored Procedure : 저장프로시저 실행을 위한 프로시저 이름을 입력합니다.
  - **TableDirect** : Data Source 의 특정 Table Data 와 직접 연결
  - **DataSource** : Data Action 의 기본 Data Source는 왼쪽 TreeView 하단의 Data Source 가 되며, 만일 Tree 의 특정 Command 에서 사용할 Data Source 를 별도로 지정하고 싶은 경우 설정합니다. 해당 Command(Query) 에서는 지정된 Data Source 에서 Query 를 실행합니다. 예를 들어 두 개의 다른 Data Source(A, B DB) 에서 동일한 Schema 의 Data 를 Query 하여 Data 의 합집합을 사용하는 경우 사용할 수 있는 방법입니다.

- **Bind Table** : 일반적으로 Output DataItem 에서 사용하는 옵션입니다. 해당 옵션이 선택된 경우 대상 Data Row 단위로 해당 Query 가 실행됩니다. Select Query 에는 반영되지 않습니다.

6. Query 작성 시, Arguments 를 사용하고자 하는 경우에는 “@” 문자를 접두어로 붙여 변수로 사용할 수 있습니다. Query 작성 시, 변수를 사용하여 작성한 후 상단 메뉴 바의 **[Extract Parameters]** 버튼을 클릭하면 하단의 그림과 같이 Parameters 가 자동 등록됩니다



위 그림에서 처럼 “@PROCESS_ID” 변수를 사용하여 입력된 PROCESS_ID 에 대한 ARRANGE 만 가져올 수 있도록 Query 를 작성할 수 있습니다. 설정된 Parameters 는 Main Control 의 **[Main > SetQueryArgs]** 함수에서 값을 설정할 수 있습니다. 또한 Studio 에서 사용자가 Query 를 테스트할 때는 Studio 의 **[Test Values]** 메뉴를 사용하여 Parameters 값을 설정할 수 있습니다. 아래는 두 가지 방식에 대한 사용 예제입니다.

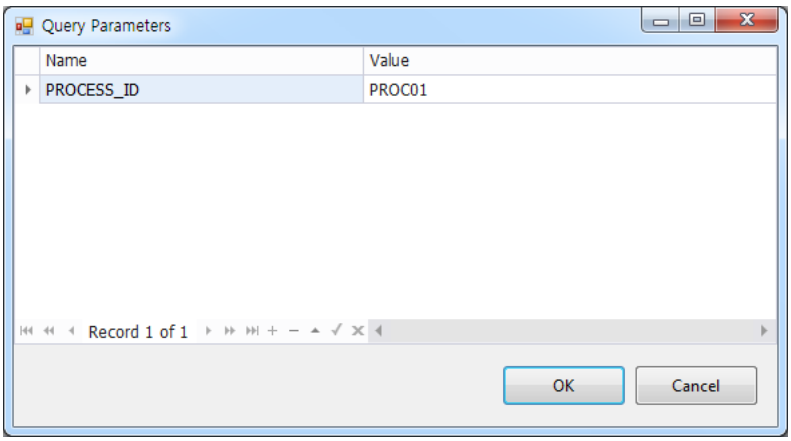
Query Arguments 의 설정 및 실행 결과

```
// Main>Main>SetQueryArgs 구현 내용입니다.
// 아래와 같이 변수를 입력하면, 입력 Data 의 Query 수행 시 Argument 를 반영하여 Query 합니다.
 public void SETUPQUERYARGS_0(ModelTask task, ModelContext context, ref bool handled)
 {
     // ModelContext 의 QueryArgs 에 Query 작성 시 사용한 변수에 대한 값을 입력합니다.
     context.QueryArgs.Add("PROCESS_ID", "PROC01");
 }
```

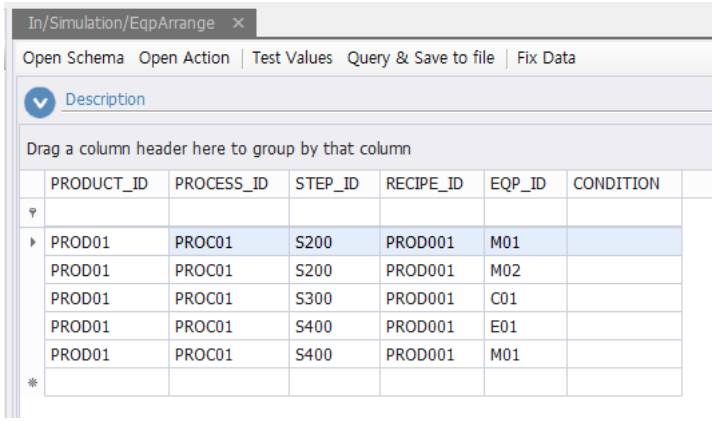
Studio 에서 테스트하기

Studio 상에서 해당 Query 를 테스트하기 위해서는 먼저 Input Data 를 조회한 후 상단의 **[Test Values]** 메뉴를 사용해서 Query 에 사용한 변수에 대한 값을 입력합니다. 그 후 **[Query & Save to file]** 메뉴를 사용하여 Query 를 수행하면 변수가 반영된 결과를 얻을 수 있습니다.

[Test Values 실행 및 변수값 입력]



[Query & Save to file 실행 결과]



PRODUCT_ID	PROCESS_ID	STEP_ID	RECIPE_ID	EQP_ID	CONDITION
PROD01	PROC01	S200	PROD001	M01	
PROD01	PROC01	S200	PROD001	M02	
PROD01	PROC01	S300	PROD001	C01	
PROD01	PROC01	S400	PROD001	E01	
PROD01	PROC01	S400	PROD001	M01	

7. 왼쪽 Query Tree View 하단의 Activate 체크 박스를 체크하고 저장하면 해당 Schema 에 대해 설정한 DataAction 이 활성화됩니다.
8. 상단 메뉴 바의 경우, dataaction 의 Command 관리 기능과 Query 를 지원하는 기능으로 구성됩니다.
   - **Query Builder** : Query 작성을 지원합니다. Data Source 에 연결한 후 Table Schema 를 이용한 Select, Insert, Update, Delete Query 생성을 지원합니다.
   - **Extract Parameter** : Query 에 삽입된 Parameters 를 추출하여 Query 편집창 하단의 Parameters List 에 자동으로 추가합니다. 추출대상이 되는 변수는 접두어 "@" 를 붙여 정의합니다.
     - 예: @LINE_ID, @PROCESS_ID 등
   - **Test Values** : Query 작성 시 Parameters 를 설정한 경우에 Parameters 의 테스트 값을 입력하기 위해 사용합니다. 입력된 값은 Studio 에서 **"Query & Save to file"** 메뉴 실행 시 기본적으로 사용됩니다. Studio 의 **"Test Values"** 메뉴를 실행하면 DataAction 에서 설정한 값이 기본으로 설정된 것을 확인할 수 있습니다.

참고. Stored Procedure 호출하는 방법

[CommandType] 중 `StoredProcedure` 타입의 경우 기능 오류가 있어 `Text` 타입으로 사용하는 것을 권장합니다.

데이터베이스별 사용 방법

MS SQL

MS SQL 에서는 실행 명령(`execute`, `exec`)을 통하여 실행할 수 있습니다. 실행 명령 뒤에 **데이터베이스이름.스키마이름.저장프로시저이름** 형태로 쿼리 작성이 되어야 합니다.

```
//매개변수 없을 때
exec Database_Name.Schema_Name.Procedure_Name

//매개변수 있을 때
exec Database_Name.Schema_Name.Procedure_Name @Param1, @Param2, ...
```

Oracle / MySql / DB2 / PostgreSQL

Oracle, MySql, DB2, PostgreSQL의 경우는 `Call` 명령을 통하여 실행할 수 있습니다. 매개 변수와 관련하여 괄호를 저장 프로시저 이름 옆에 추가하여 쿼리 작성이 되어야 합니다.

```
//매개변수 없을 때
call Procedure_Name()

//매개변수 있을 때
call Procedure_Name(@Param1, @Param2, ...)
```

> ⚠️ Oracle의 경우 Call 명령은 Oracle 9i 버전부터 사용 가능합니다.
> 9i 이전 버전의 경우 `Exec(Execute)` 명령을 사용하기 바라며, `Exec(Execute)` 명령 사용시 매개변수가 없을 때는 저장 프로시저 이름 옆에 괄호를 사용하지 않습니다.

주의사항

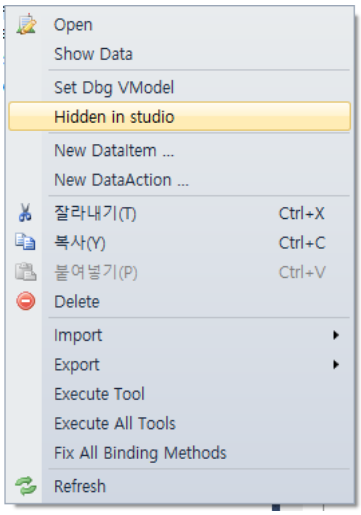
Output DataAction에서 저장 프로시저를 호출하는 경우

- Bind Table의 옵션이 True인 경우 Data Row 갯수 만큼 쿼리가 수행되며, 의도에 맞게 수행되는지 설정 확인을 하시기 바랍니다.
- Output Data Row 값을 매개변수로 전달하여 프로시저로 저장하는 경우에는 저장 속도가 빠른 **BulkCopy 사용**을 권장합니다.

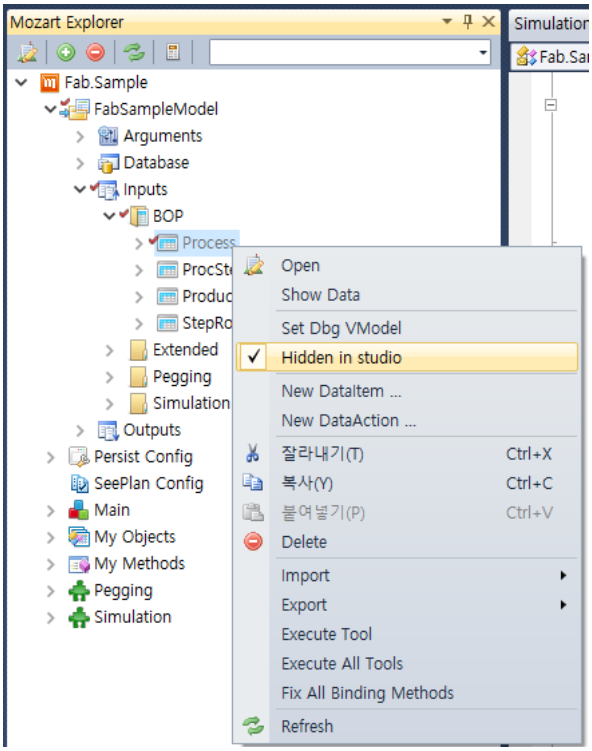
DataItem 숨기는 방법

MOZART IDE를 통해 특정 DataItem을 숨기면서 MOZART Studio에 최종적으로 노출시킬 DataItem을 설정 할 수 있습니다. 기본적으로는 별도의 설정이 없으면 IDE를 통해 추가한 DataItem은 MOZART Studio에서 노출이 됩니다. 다음은 MOZART IDE상에서 DataItem을 숨기는 방법입니다.

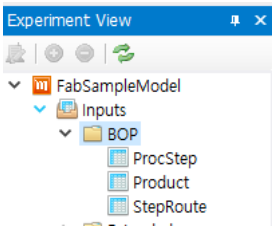
1. MOZART Explorer의 **[Model > Input/Output > DataGroup >DataItem]** 선택하고 마우스 우 클릭 메뉴에서 **[Hidden in studio]** 메뉴를 선택합니다.



2. 아래와 같이 **[Hidden in studio]**가 체크되어있고 해당 DataItem 명의 색상이 변경된 것을 확인하였으면 변경 사항을 저장합니다.

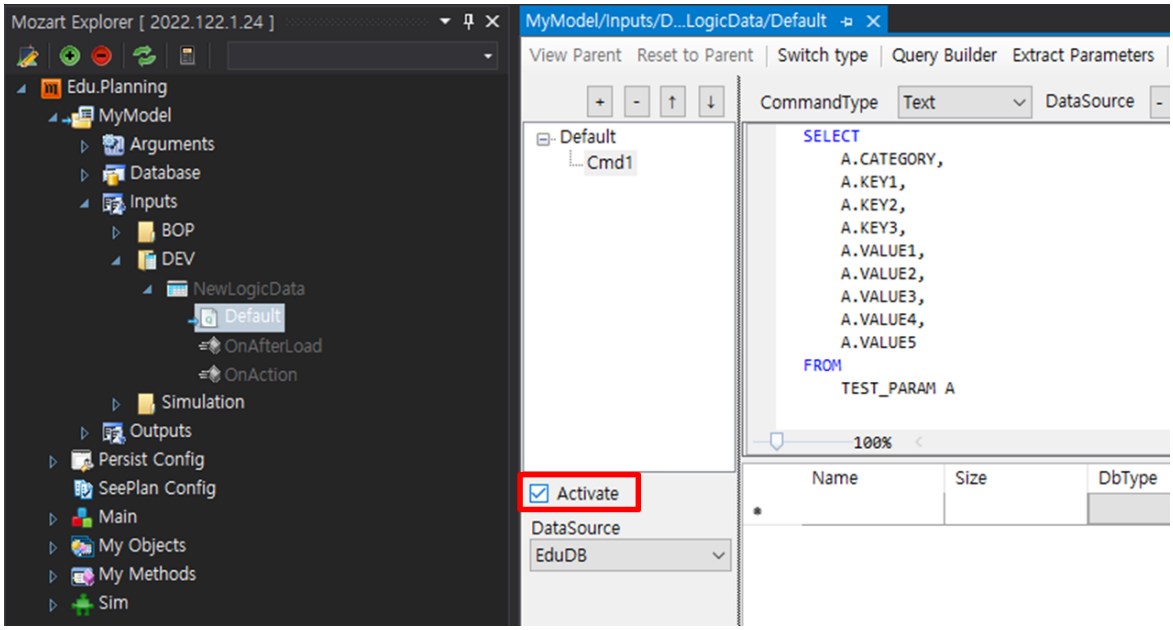


3. 프로젝트를 빌드하여 MOZART Studio를 실행합니다. DataItem을 숨긴 **[Model > Input/Output > DataGroup]**을 선택하여 숨겨진 DataItem이 안 보이는지 확인합니다.



주의 사항

MOZART Server 에서 모델의 DataItem이 **[Hidden In Studio]**를 통해 숨겨져도 DataAction이 연결되어 있으면 수행이 됩니다. DataAction 이 수행되지 않기를 원할 경우에는 **Activate**를 체크 해제하기 바랍니다.



System Arguments 종류

MOZART Project 로 생성된 Task(Dll 파일) 와 Model 은 실제로 실행될 경우 MOZART 실행엔진의 Model Task 를 통해 수행됩니다. 이러한 Model Task 의 실행 옵션을 조정하기 위해 사전 정의된 Arguments 가 System Arguments 입니다. 개발자는 Model 의 Input Arguments 설정 시 사전 정의된 System Arguments 를 지정함으로써 사전 정의된 실행 옵션을 활용할 수 있습니다. 아래 표는 사전 정의된 System Arguments 에 대한 설명합니다.

Basic Arguments

Argument Name	Argument 설명	Data Type
model	Model 파일 명(실행하는 위치의 경로가 포함되는 Full name)	string
experiment	Model 실행시 결과를 생성할 Experiment 명. 기본값은 “Experiment 1”	string
versionNo	Model Version 명 (기본값은 {model-name}-{yyyyMMdd-HHmmss} 형식)	string
model-name	versionNo 없을 때 versionNo에 사용될 이름	string
start-time	Task 시작시각 (Simulation Clock)	DateTime
end-time	Task 종료시각 (Simulation Clock)	DateTime
period	Plan&Schedule 기간	float
period-unit	period 설정 단위(기본은 일단위)	string
start-time.AdjustMinutes	Job 실행 시작 시간에서 해당한 시간만큼 조정한 시간을 start-time 으로 설정하도록 조정하는 입력변수	int
model-file	실행 Model 명을 포함한 Full Path	string
model-dll	Model 을 실행할 Dll 의 Full Path	string
model-config	Model 실행시 Config 파일의 Full Path	string

Data Download/Upload 옵션 Arguments

Argument Name	Argument 설명	Data Type
overwrite_result	Result 덮어쓰기 여부	bool
useDatabase	Database 읽기 여부	bool

Argument Name	Argument 설명	Data Type
saveDatabase	Output Data 를 Database 에 쓰기 여부	bool
db-To-file	시뮬레이션은 실행하지 않고 Database Sync 만 진행 여부 (기본값 = false)	bool
db-includes	Input Data Sync 대상 Table List 파일 이름, 대상 Table 들만 Sync 진행	string
db-excludes	Input Data Sync 제외 Table List 파일 이름, 제외 Table 들을 제외한 모든 Table Sync 진행,daction_includes 동일 값이 있는 경우, excludes 반영 안됨	string
daction_excludes	시뮬레이션 완료 후 DataAction 실행하지 않을 Table List, Comma 로 구분	string
daction_includes	시뮬레이션 완료 후 DataAction 실행할 Table List, Comma 로 구분	string
daction_excludes/in	시뮬레이션 시작 시 Input Schema 의 DataAction 중 실행에서 제외할 Action List, Comma 로 구분	string
daction_includes/in	시뮬레이션 시작 시 Input Schema 의 DataAction 중 실행할 Action List, Comma 로 구분	string

Run Arguments

Argument Name	Argument 설명	Data Type
more-runs	Model 의 반복 실행 횟수	int
more-config-[runindex]	반복 실행 회차 별 Arguments 설정 값이 저장된 xml 파일, config 파일에서 설정되지 않은 Arguments 는 이전 회차에서 사용한 Arguments 반영반복 실행 시 이전 회차에서 Loading 한 Data 들이 Memory 에서 지워지지 않아 다시 Loading 하면서 Error 가 발생할 수 있으므로 more-runs 사용 시 Result 기록 후 DataMart 를 Reset 해야 함	string

Zip Model Arguments

실행 후 압축파일로 Model 을 압축할 때 사용하는 규칙에 대한 설정 Arguments 입니다.

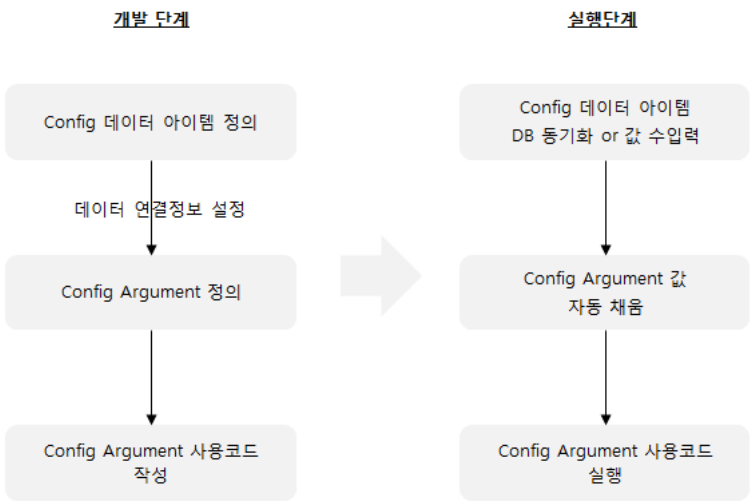
Zip Model Arguments

Argument Name	Argument 설명	Data Type
zip	시뮬레이션 완료 후 Model 압축 여부, 우선 반영	bool
zip.FileNameTemplate	압축파일 이름 저장을 위한 Template 기본 Template 은 “${model_name}_${zip_now}${zip_postfix}”사용가능한 keyword 는 아래와 같습니다. * ${model_name} : Model 명* ${now} : 압축실행 시점 시각(DateTime)* ${zip_now} : 압축하는 시각문자열(format : yyyyMMddHHmmss)* ${zip_postfix} : 설정한 압축파일 이름 접미사* ${version-no} : Model 실행 Version 명	string

Config Arguments 설정방법

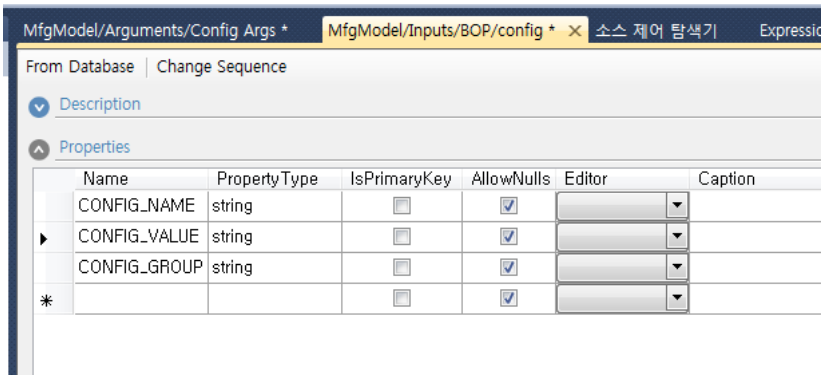
Config Argument 의 설정 방법을 설명합니다.

Config Arguments 의 Arguments 정의 방식은 Input Arguments 입력 방식과 동일합니다. 단, Config 의 경우 사용자 조직에서 기존에 관리하던 기준 정보를 참조해야 하는 경우가 많으므로, 이를 위해 Input DataItem 으로 실제 Arguments 값을 관리하고, 실행 시점에 해당 DataItem 값을 자동으로 Config Arguments 에 할당하는 방식을 사용할 수 있도록 기능이 구성되어 있습니다. 아래 그림과 같습니다.

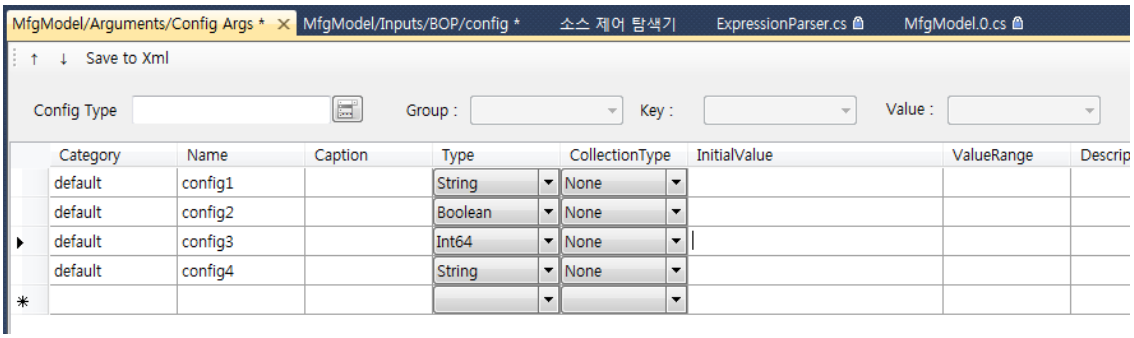


개발단계에서의 Config Arguments 정의&사용방식

1. Config Data 를 저장할 DataItem 을 Inputs 에 생성합니다. (아래 그림)



2. 시스템에서 사용하게 될 Config Arguments 를 정의합니다. (아래 그림)

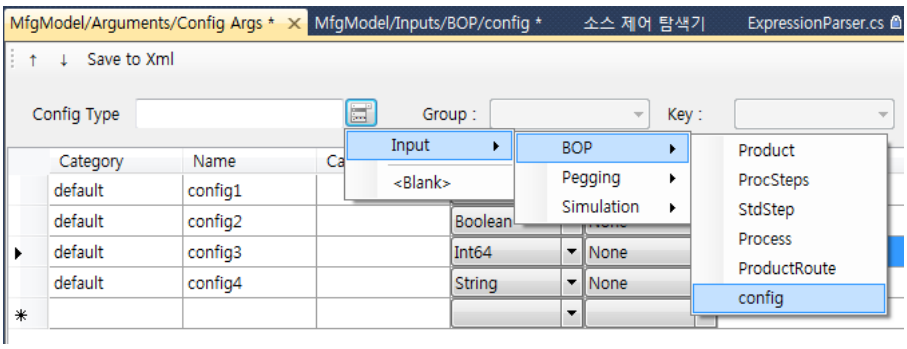


3. Config Arguments 를 자동으로 할당할 때 사용할 DataItem 정보 및 Column 매핑 정보를 설정합니다.

   실행 시점에 Config Arguments 의 값을 채울 수 있는 Config Data Table 값과 Mapping Field 에 대한 설정을 추가적으로 수행합니다.

   - **Config Type** : Config Arguments 의 Data Table 을 Input DataItem 중 선택하여 입력합니다.

- **Group** : Input DataItem Data 를 Grouping 하여 관리하고자 할 때 사용하며, DataItem 의 값을 설정할 때 해당 Column 에 Group 을 설정합니다. 선택한 Config Type 의 Column 중 선택할 수 있으며 필수항목은 아닙니다.
- **Key** : Config Arguments 의 Name 속성과 Mapping 할 Column 을 지정합니다. 선택한 Config Type 의 Column 중 선택할 수 있습니다.
- **Value** : Config Arguments 의 값을 Mapping 할 Column 을 지정합니다. 선택한 Config Type 의 Column 중 선택할 수 있습니다.



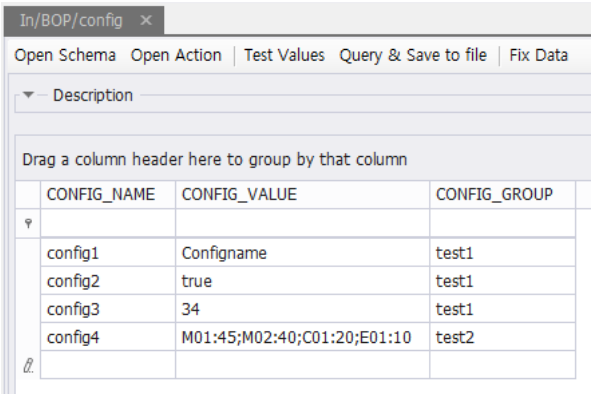
4. 설정된 Config Arguments 를 사용하여 Logic 을 구현합니다. 아래는 예시적인 코드를 보여줍니다.

Config Arguments 사용 예

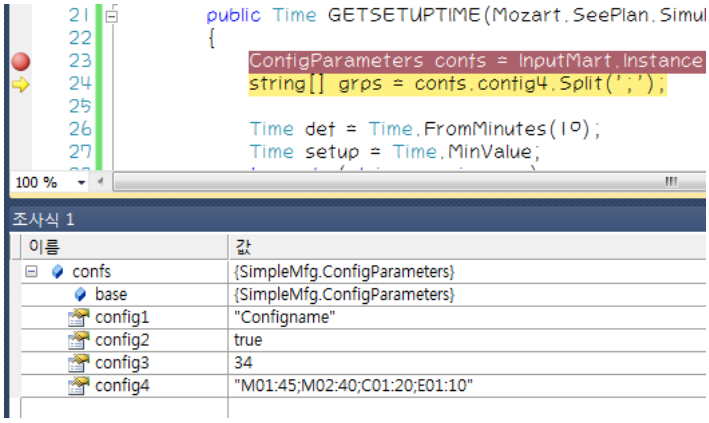
```
// Setup 시간을 확인할때 장비 Group 에 따라 다르게 적용하기 위한 코딩 예
 // config4 를 장비별 Setup 시간을 정의한 Config Arguments 로 설정했을 때
public Time GETSETUPTIME(Mozart.SePlan.Simulation.AoEquipment aeqp, IHandlingBatch hb, ref bool handled, Time prevReturnValue)
{
   ConfigParameters confs = InputMart.Instance.GetConfigParameters(string.Empty);
   string[] grps = confs.config4.Split(';');

   Time def = Time.FromMinutes(10);
   Time setup = Time.MinValue;
   foreach (string grp in grps)
   {
       string[] setupTime = grp.Split(':');
       if (aeqp.EqpID == setupTime[0])
       {
           setup = Time.FromMinutes(Convert.ToInt32(setupTime[1]));
           break;
       }
   }

   if (setup == Time.MinValue)
   setup = def;

   return setup;
}
```

실행단계에서 확인

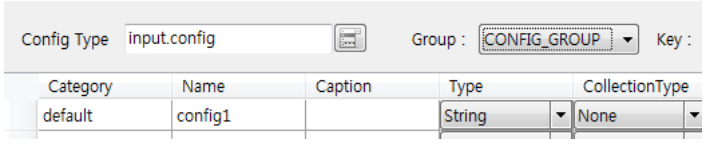
1. MOZART Studio 에서 Config 정보로 사용할 Data 를 확인합니다. 만일 DB 로부터 Data 를 Sync 하도록 DataAction 을 추가한 상태라면 [Query & Save to file] 메뉴를 사용하여 값을 가져옵니다.



2. 위와 같이 설정한 경우 Config Parameters 의 Config 명 변수에 값이 자동으로 할당되고 상기 로직 구현 예제와 같이 직접 참조가 가능하게 됩니다. 아래 그림은 Debugging 시점에 해당 confs 변수의 값 할당 현황을 보여줍니다.



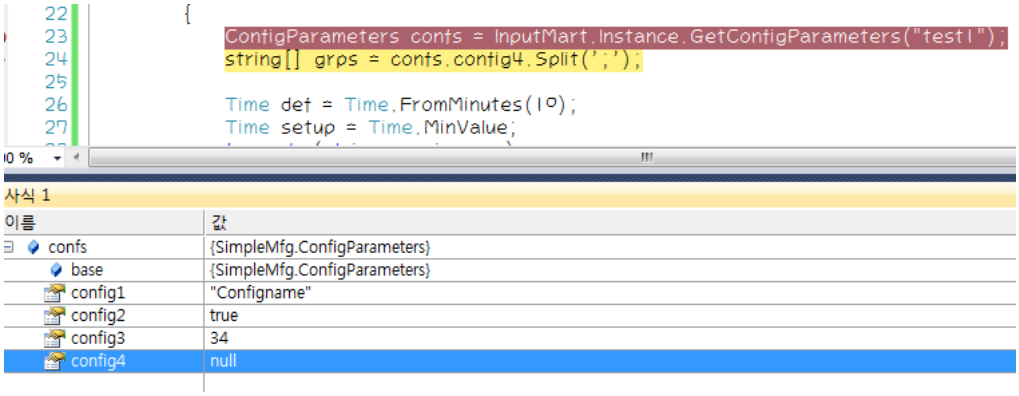
3. 라인별로 혹은 특정 Group 에 대한 Config 를 별도로 관리하고자 할 경우에는 Config Arguments 설정 시 Group Mapping Column을 선택한 후 아래와 같은 방식으로 Config 변수를 가져올 때 특정 Group 을 지정하는 방식을 사용할 수 있습니다.



Group별 Config Arguments 받아오기

```
// 전체를 받아오는 경우
 ConfigParameters confs = InputMart.Instance.GetConfigParameters(string.Empty);
 // 특정 Group 을 받아오는 경우, 아래와 같이 설정하면 config Table 의 CONFIG_GROUP = "test1" 인 세 개의 설정정보만 Mapping 값이 설정되어 생성됩니다.
 ConfigParameters confs = InputMart.Instance.GetConfigParameters("test1");
```

4. 위와 같이 특정 Group 에 대한 설정 시 결과 값은 아래와 같이 변경되어 “config4”에 대한 값이 null 로 setting 됩니다.



BulkCopy 사용 방법

DB Save 할 때 대량의 레코드를 배치 처리하여 빠르게 INSERT 하는 방식을 MOZART IDE에서 설정하는 방법에 대한 문서입니다.

개요

시스템 운영 중 설비 계획, 공정 별 생산 목표 등 대량의 레코드를 생성해야 할 때가 있습니다. Mozart에서 **Text** 형식으로 INSERT 쿼리를 작성했을 때는 1개의 레코드에 1개의 INSERT 쿼리가 실행되는 구조입니다. 레코드 수가 많은 경우 이러한 방법은 매우 큰 부하 작업이므로 DB에 데이터를 기록하는데 시간이 많이 소모됩니다.

MOZART에서는 대량의 레코드를 DB에 기록하기 위해 배열 변수(Batch)에 여러 개의 레코드를 설정하고 하나의 쿼리로 INSERT 할 수 있는 **BulkCopy** 기능이 있습니다. **BulkCopy**를 이용하면 쿼리 실행 횟수가 줄어 그만큼 부하를 줄일 수 있으며 DB에 데이터를 기록하는 시간도 보통의 INSERT 방식보다 빠릅니다. 또한 MOZART의 **BulkCopy** 기능은 MSSQL, 오라클 등의 여러 Data Provider에서 사용이 가능합니다.

본 문서는 MOZART를 통해 **BulkCopy**를 설정하는 방법을 설명합니다.

사용 방법

BulkCopy는 MOZART IDE에서 설정이 가능합니다. DataAction 편집창에서 **[CommandType]**을 **BulkCopy**로 설정하면 아래와 같이 값을 입력 할 수 있는 속성들이 제공됩니다.

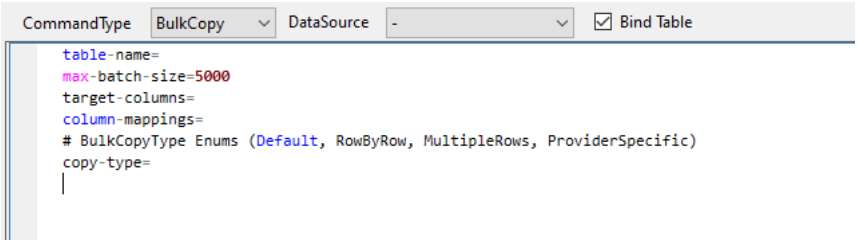


그림 1. BulkCopy 설정 화면

각 속성에 대한 정의는 다음과 같으며, **모든 속성은 필수 입력입니다**.

- ***table-name*** : 데이터를 기록할 DB의 테이블 이름입니다.
- ***max-batch size*** : 일괄적으로 INSERT를 실행할 배치의 크기를 설정합니다. (기본값: 5,000)
- ***target-columns*** : 쓰기 대상 DataItem의 컬럼 명. 다수의 컬럼을 입력하는 경우 구분자는 쉼표(,)입니다.
- ***coulumn-mappings*** : DataItem의 컬럼과 DB 컬럼의 간의 매핑 정보를 설정합니다. (사용 예: DataItem 컬럼명 = DB Table 컬럼명) DataItem의 컬럼명과 DB 테이블의 컬럼명이 일치하는 경우 생략이 가능합니다.

- ***copy-type*** : INSERT 방식을 설정합니다.
  - **Default** : DB Provider에서 제공하는 방식으로 INSERT를 진행합니다. 예를 들어 DB Provider가 Oracle인 경우 여러 레코드를 일괄적으로 기록하는 `MultipleRow` 방식을 사용합니다. 전체 데이터에 대한 Transaction-Commit을 수행하는 관계로 삽입 중 오류 발생 시 이전 삽입 데이들도 Rollback 됩니다.
  - **RowByRow** : 주어진 데이터 컬렉션을 반복하면서 레코드 단위로 별도의 INSERT 문을 실행하여 DB에 삽입하는 방식입니다. 일부 DB Provider는 `RowByRow` 만 제공 할 수 있습니다. 전체 데이터에 대한 Transaction-Commit을 수행하는 관계로 삽입 중 오류 발생 시 이전 삽입 데이터들도 Rollback 됩니다.
  - **MultipleRow** : INSERT ALL 방식으로 레코드 단위가 아닌 배치 단위로 DB에 삽입하는 방식입니다. 전체 데이터에 대한 Transaction-Commit을 수행하는 관계로 삽입 중 오류 발생 시 이전 삽입 데이터들도 Rollback 됩니다. 다만, DB Provider가 해당 방식을 지원하지 않으면 자동으로 `RowByRow` 방식으로 DB에 데이터를 삽입합니다.
  - **ProviderSpecific** : 실질적인 BulkCopy 방식으로 배치 단위로 Transaction-Commit을 수행합니다. 데이터 삽입 중 오류 발생 시 해당 배치에 대해서만 Rollback 되며 이전에 삽입된 배치 단위들은 DB에 기록됩니다. DB Provider에서 해당 기능을 제공하지 않으면 `MultipleRow` 방식으로 데이터를 삽입합니다. (주의 사항 참고)

예제

```
table-name= VMS_SAMPLE_TABLE //INSERT 대상 DB 테이블명
max-batch-size=5000 //일괄적으로 INSERT 할 레코드 수
target-columns=VERSION_NO,LINE_ID,PRODUCT_ID //INSERT 대상 컬럼
//DB 컬럼 매핑 정보
column-mappings=VERSION_NO=VERSION_NO,LINE_ID=LINEID,PRODUCT_ID=PRODUCTID
# BulkCopyType Enums (Default, RowByRow, MultipleRows, ProviderSpecific)
copy-type= ProviderSpecific
```

위와 같이 **BulkCopy**를 설정하면 INSERT가 실행 될때마다 5,000의 레코드가 DB에 삽입됩니다. DataItem의 컬럼명과 DB의 컬럼명이 다른 경우 **column-mappings**에 매핑 정보를 설정하면 되며 **Text** 방식에서 사용하는 DataAction 편집창 하단의 파라미터는 따로 설정하지 않아도 됩니다.

Name	Size	DbType	Direction	Precision	Scale	SourceVersion	SourceColumn
@step_target_id	255	AnsiString	Input	0	0	Current	
@version_no	255	AnsiString	Input	0	0	Current	
@product_id	255	AnsiString	Input	0	0	Current	
@step_id	255	AnsiString	Input	0	0	Current	
@in_qty	0	Decimal	Input	10	3	Current	
@out_qty	0	Decimal	Input	10	3	Current	

BulkCopy 사용 시 주의점

target-columns / column-mappings 설정 주의 사항

BulkCopy 에서 **target-columns** 그리고 **column-mappings**를 설정할 때 줄 바꿈이 있는 경우 오류가 발생합니다. INSERT 대상 컬럼이 많은 경우 가독성이 매우 떨어지는 문제가 있으며 컬럼간 줄 바꿈이 있어도 오류가 발생하지 않도록 기능 개선 예정입니다. 기능 개선 전까지는 줄 바꿈 없이 컬럼명을 기재 바랍니다.

```
table-name= VMS_SAMPLE_TABLE
max-batch-size=5000
target-columns= //오류 발생
VERSION_NO,
LINE_ID,
PRODUCT_ID
column-mappings= //오류 발생
VERSION_NO=VERSION_NO,
LINE_ID=LINEID,
PRODUCT_ID=PRODUCTID
# BulkCopyType Enums (Default, RowByRow, MultipleRows, ProviderSpecific)
copy-type= Default
```

Batch 단위 Commit 옵션

Batch 단위로 Transaction-Commit하는 기능은 2021.118.3 버전부터 사용이 가능합니다. 이전 버전에서는 `copy-type=ProviderSpecific` 으로 설정해도 오류 발생 시 전체 Rollback되는 점 참고 바랍니다.

BulkCopy 속도가 INSERT와 유사하거나 느린 경우

BulkCopy를 수행했는데 INSERT 보다 느린 경우 다음 사항을 체크 하시기 바랍니다.

copy-type 설정 확인

현재 `copy-type=ProviderSpecific` 외에는 DB에 데이터 저장하는 속도가 INSERT문이랑 유사합니다. `copy-type` 값을 확인하고 변경하기 바랍니다.

컬럼에 Index나 설정되어 있는지 확인

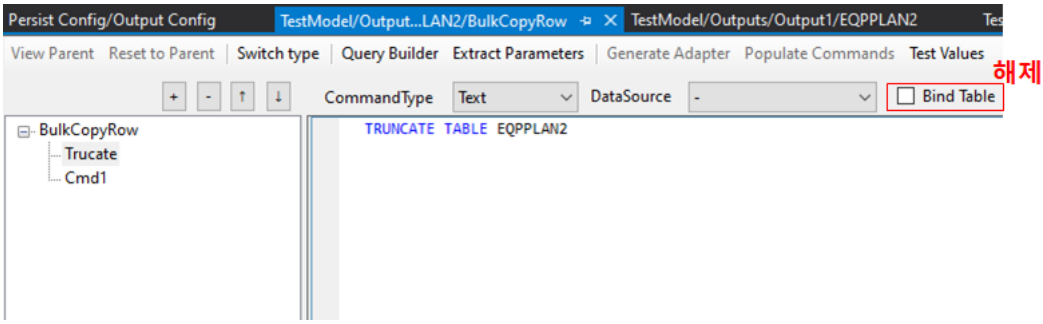
Non-clustered index가 있는 테이블 그리고 해당 테이블이 비어 있는 경우 BulkCopy 속도가 현저히 떨어질 수 있습니다. 이 경우에는 BulkCopy를 수행하기 전 Index를 모두 Drop한 다음 BulkCopy로 데이터를 적재하고나서 다시 해당 테이블에 Index를 추가하면 속도가 개선됩니다.

max-batch-size 값 조정하기

BulkCopy 시 `max-batch-size` 값이 크다고 해서 항상 속도가 빠른 것은 아닙니다. BulkCopy를 수행하는 PC의 메모리양과 데이터를 받는 서버 PC의 메모리양을 그리고 네트워크 환경에 따라 batch 크기에 따른 속도 차이가 발생합니다. 권장하는 batch 크기는 500~10,000 사이이며 현재 환경에서 최적의 크기를 찾기 위해서는 기준 batch 크기를 설정한 다음 1,000 단위 혹은 2배 증가시키면서 시간을 측정하여 가장 속도가 빨랐던 batch 크기로 `max-batch-size` 값을 설정하면 됩니다.

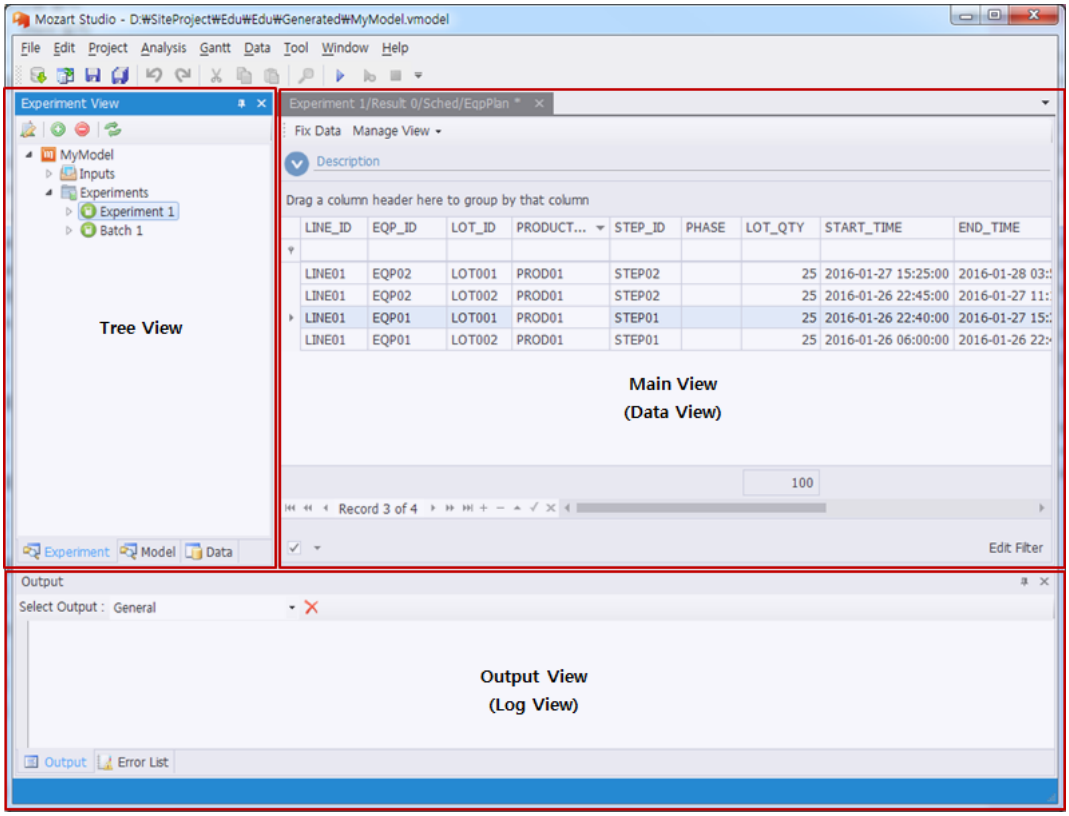
DataAction의 command 확인하기

BulkCopy를 수행하는 DataAction의 command 목록 중 Delete나 Truncate Table를 수행하는 command가 있는지 확인 바랍니다. 만약 있다면 Delete 또는 Truncate Table command의 **[Bind Data]** 옵션을 해제하시기 바랍니다. **[Bind Data]** 옵션이 활성화된 경우 Row 갯수만큼 Delete와 Truncate Table이 수행되기 때문에 Row 수가 많으면 시간이 오래 걸립니다. 즉, BulkCopy가 아니더라도 다른 Insert나 Update command 전에 Delete 또는 Truncate Table를 수행하는 command가 있으면 반드시 **[Bind Data]** 옵션은 해제 바랍니다.



UI Views

MOZART STUDIO 의 기본 UI 는 세가지 영역으로 구분됩니다.



Tree View

Docking Window 로 주로 화면의 좌측에 기본 위치합니다. 해당 Tree View 에서는 3가지 Tab 을 통해 실행결과, 모델, 데이터소스를 조회 가능하도록 기능을 제공합니다.

- Experiment Tab View : Model 의 실행시 사용된 Input과 Experiment 별 실행결과 데이터를 Tree 형태로 제공합니다.
- Model Tab View : Model 에 정의된 Input/Output data item 에 대한 schema, data action 을 조회 하거나 정의된 Argument, Datasource 정의를 조회 할 수 있습니다. Local Model 의 수정은 가능하지만 실행 Dll 과 연결시 오류를 유발 할 수 있음으로 수정하여 재 배포하는 작업은 할 수 없습니다.
- Data Tab View : Model 에 설정된 Data source 별 데이터 테이블, Stored Procedure 를 조회 할 수 있습니다. 필요시 추가적인 데이터 소스를 등록하여 조회 가능합니다.

Main View

Tree View를 통해 선택되거나 Main Menu bar 를 통해 활성화된 창들이 열리는 View 입니다. 주로 Input/Output Data 를 조회하거나 분석/설정 UI 창을 표시합니다.

Output View

Docking Window로 주로 화면의 하단에 기본 위치합니다. Model 실행시 로그를 조회 할 수 있으며, 모델 실행 종료시 엔진에서 발생한 Error, Warning, Information 을 Level 에 따라 조회 할 수 있습니다.

Pivot Grid View 사용법

Pivot Grid View는 기본적으로 Grid View에서 제공되는 기능과 동일하게 동작합니다. 기본 기능은 **Grid View 사용법**을 참고 할 수 있습니다. 그 외 Pivot Grid View 특성에 맞게 동작하는 기능에 대해 설명합니다.

PivotGrid Field List

Pivot Grid에서 각 필드 목록을 보여주는 창입니다. 각각의 필드를 아래 영역(Area)에 드래그하여 피봇 그리드를 생성할 수 있습니다. 기본적으로 필드를 영역으로 드래그 할 때 Pivot Grid에 자동으로 반영됩니다. 만약 각각의 필드를 영역에 드래그 후 한꺼번에 반영 여부를 결정하는 옵션(**Defer Layout Update**)을 체크하면 **Update** 버튼을 눌러 Pivot Grid에 반영할 수 있습니다.

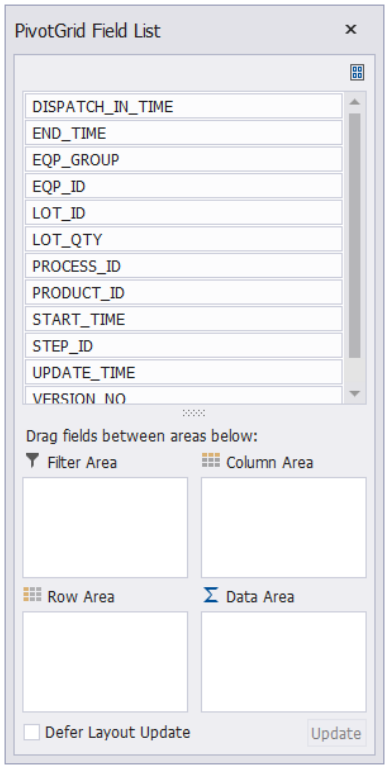
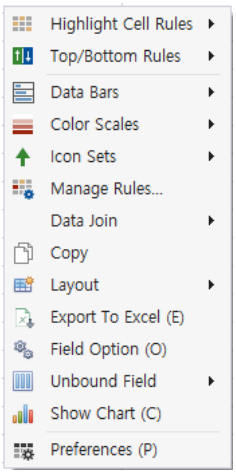


그림 1. Pivot Grid Field List Enter a caption for this image (optional)

단축키

주요 기능에 대하여 단축키 사용이 가능합니다. 각 메뉴 옆 괄호() 안의 글자가 있는 메뉴는 단축키 사용이 가능한 메뉴입니다.



1. Export To Excel - **E** : Excel 파일로 추출할 때 사용
2. FieldOption - **O** : Pivot Grid 의 Cell, Value Format 을 설정할 때 사용

3. Show Chart - **C** : Pivot Grid 데이터를 차트로 표현할 때 사용
4. Preferences - **P** : Pivot Grid Option을 설정할 때 사용

Excel Export

Pivot Grid View 의 화면과 데이터를 액셀 파일로 내보내는 기능입니다.

ExcelExport Type 설정

Pivot Grid View에서 Export To Excel 기능을 사용할 때 ExportType을 설정할 수 있습니다. 기본값은 DataWare 입니다. **[Tool] > [Options...]**를 선택하면 Preference Popups 창이 나타나며 해당 창의 **[Setting] > [PivotOptions]**에서 DefaultExportType의 **DataWare** 또는 **WYSIWYG**를 선택하고 확인 클릭 시 설정이 적용됩니다. [그림 2.] ExcelExport Type 설정 방법 순서입니다.

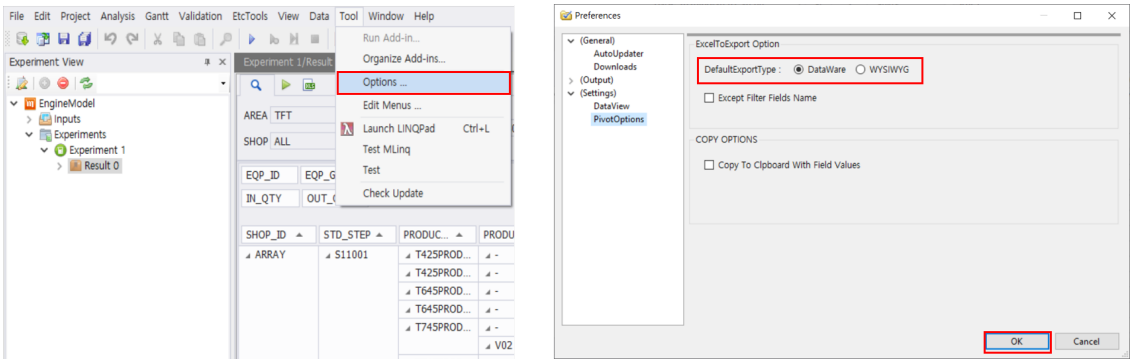


그림 2. PivotOptions에서 ExportType 설정 방법

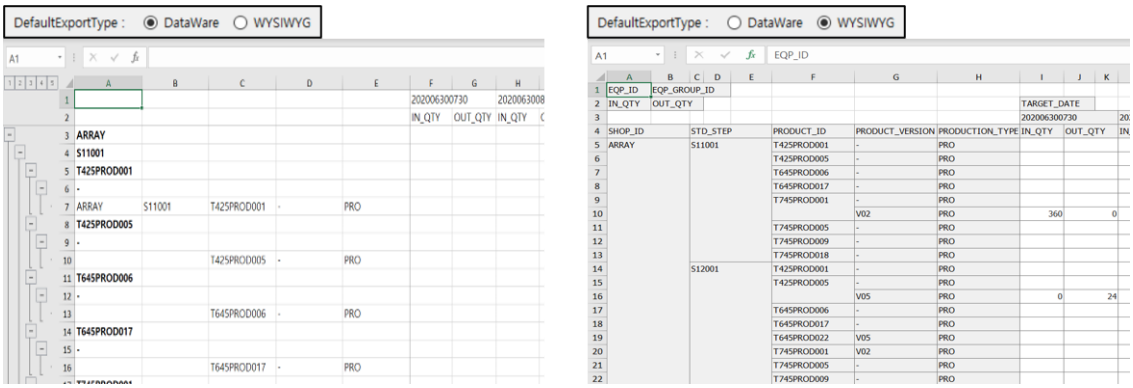


그림 3. ExportType에 따른 Excel 결과 예시

ExportType 설정 후 사용자는 [그림 4.] 같이 Cell 또는 Column Header를 우클릭한 후 **Export To Excel** 기능을 클릭하고 Open Dialog에서 File name과 Directory를 설정하여 Excel File을 저장할 수 있습니다.

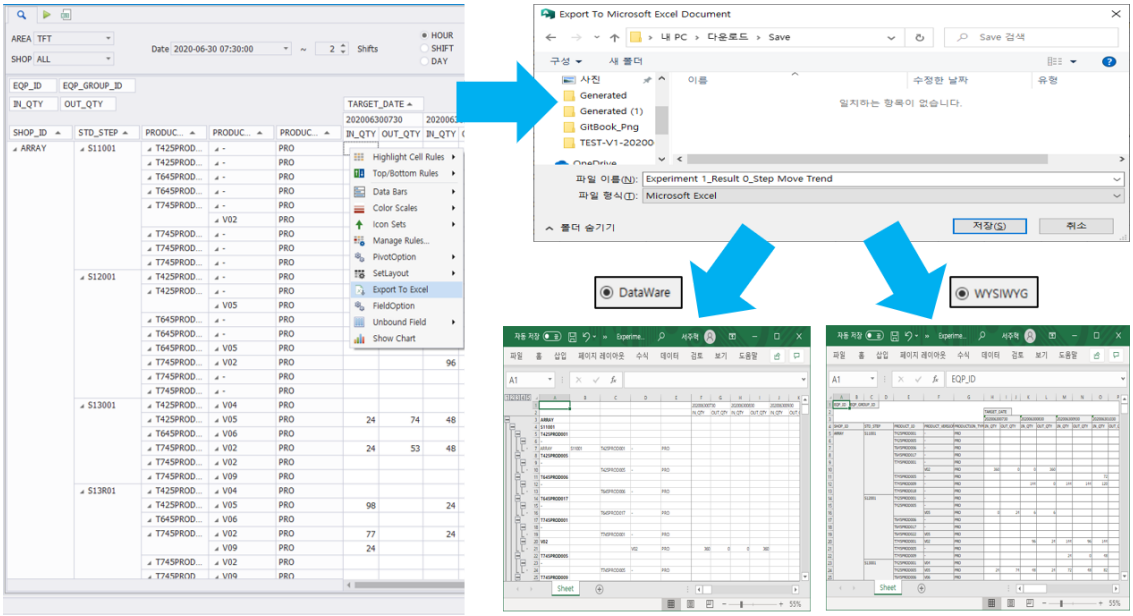


그림 4. Export To Excel 기능 사용 예시

ExcelExport Hidden Column 제외 설정

Export Type이 WYSIWYG인 경우 Excel Export 시 Filter Field Area 부분의 Column Names가 같이 추출 됩니다. 기본값은 False(UnCheck) 이며, Filter Fields Name이 함께 추출됩니다. 사용자는 **[Tool] > [Options...]** 선택 후 **Preferences Popup** 창에서 **[Setting] > [PivotOptions]**의 **Except Filter Fields Name**의 Check button을 클릭하여 해당 Column Names를 제외하고 Table을 추출 할 수 있습니다. [그림 5.] Hidden Column 설정 방법 순서입니다.

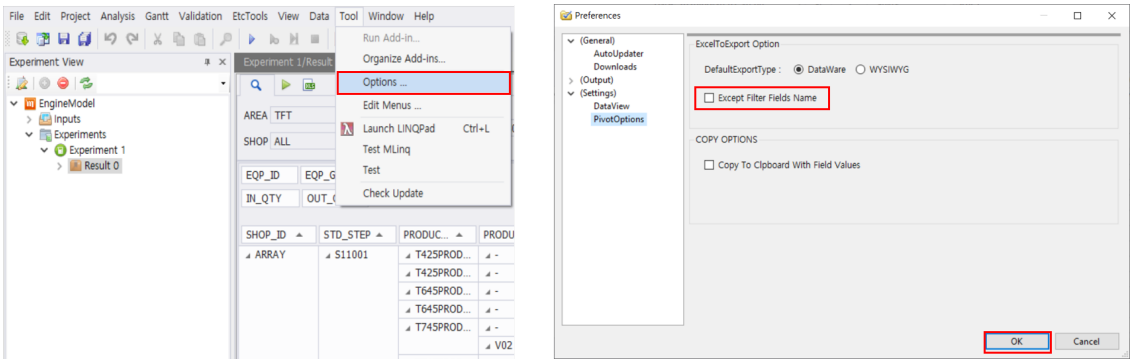


그림 5. PivotOptions에서 Hidden Column 제외 설정 방법

[그림 6.]과 같이 **[PivotOptions]**에서 **Except Filter Fields Name**를 UnCheck 시 아래의 좌측 그림처럼 Filter Field Area에 존재하는 Column Names가 Export 되며, Check 시 아래의 우측 그림처럼 Column names 이 Export 되지 않습니다. (단, 열려 있던 View에는 적용되지 않기 때문에 View를 재실행해야 합니다.)

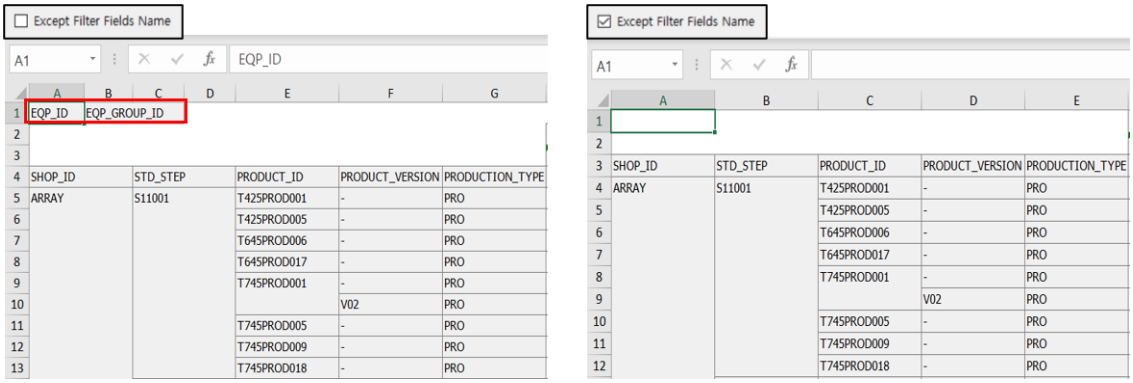


그림 6. Except Filter Fields Name 적용 결과 예시

Sum(Calculate)

해당 기능은 드래그 한 Cells에 대해 Calculation Value (Sum, Max, Min, Count, Average, Max-Min) 결과를 제공합니다. Grid View의 경우, 하나의 Cell만 선택하여도 해당 기능을 사용할 수 있지만 Pivot View의 경우, 둘 이상의 Cells를 선택하여야 해당 기능을 사용할 수 있습니다.

[그림 7.]과 같이 사용자는 Numeric Type Column의 2개 이상의 Data를 드래그 한 후 우 클릭하고 **Sum(Valid Count: *)**을 클릭합니다. 클릭 후 **Summary Control**이 나타나며, Calculation Value의 결과값을 확인하실 수 있습니다.

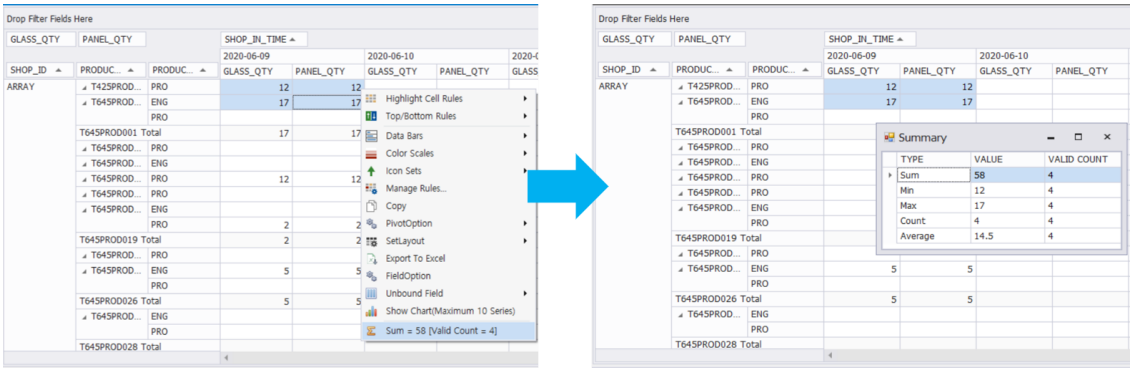


그림 7. Summary Control 사용 예시

Pivot Grid Preference

각 Pivot Grid View별로 Option을 설정할 수 있는 기능입니다. 모델의 모든 Pivot Grid View들의 기본 Option은 **[Tool]** > **[Options]** > **[PivotOptions]** 따르게 되어 있으나 Pivot Grid View Preferences 기능을 통해 개별 View의 Option을 설정할 수 있습니다.

[그림 8.]과 같이 해당 Pivot Grid View의 Columns or Cells을 우클릭한 후 메뉴에서 Preferences를 선택하면 Pivot Grid Setting 팝업 창이 나타납니다. 각 기능에 대한 설명은 아래와 같습니다.

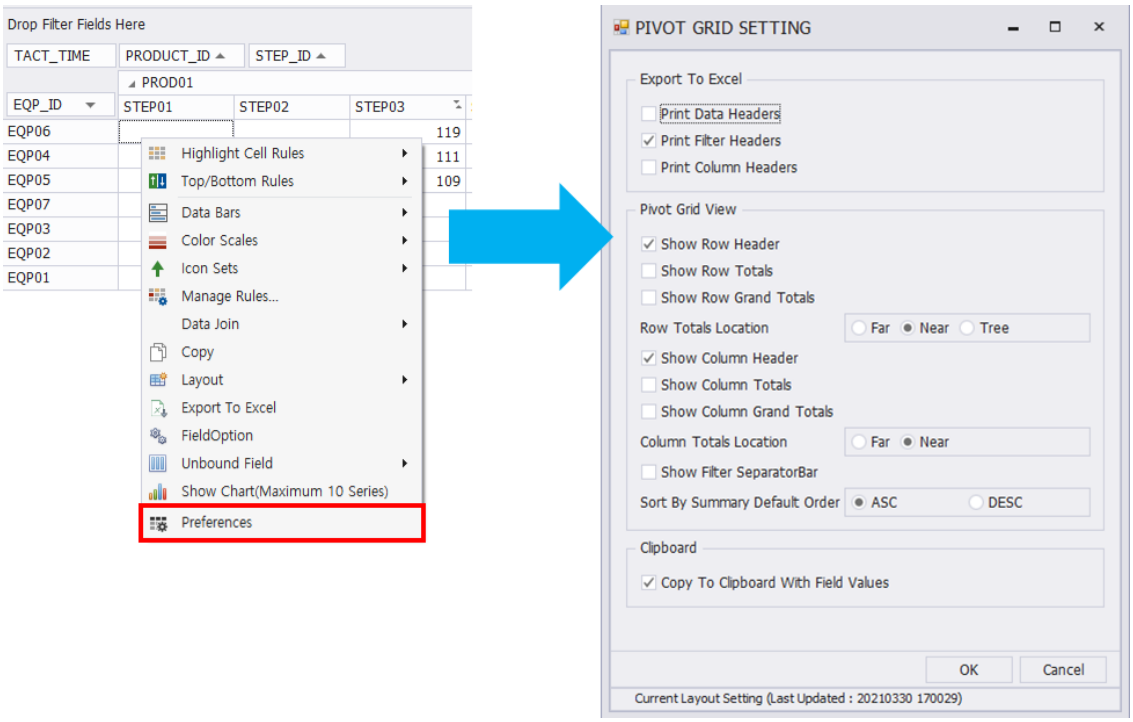


그림 8. Pivot Grid Setting 화면

Print Data Headers

Excel Export 기능 사용 시 View의 Data Headers를 같이 Export 할 지 선택할 수 있는 기능입니다.

[그림 9.]과 같이 Print Data Headers를 Check 하면 View의 Data Headers가 같이 출력됩니다.

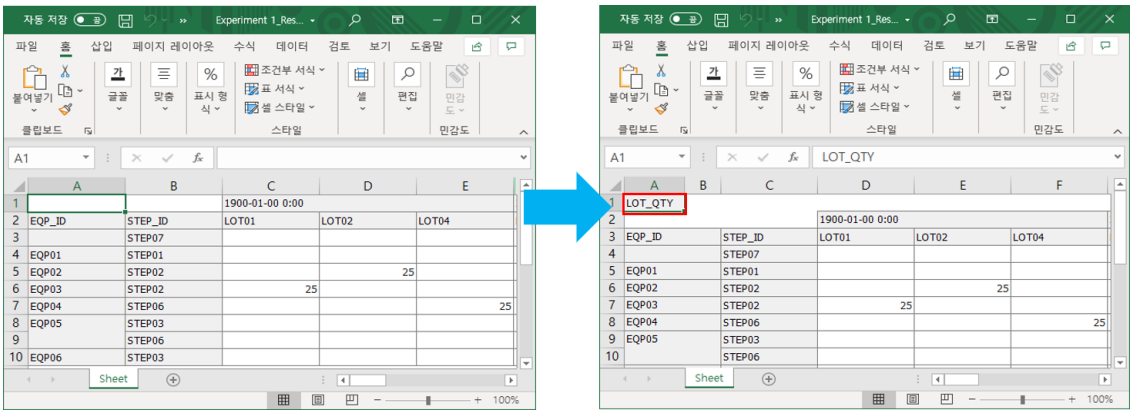


그림 9. Print Data Headers 옵션 비교

Print Filter Headers

Excel Export 기능 사용 시 View의 Filter Headers를 같이 Export 할 지 선택할 수 있는 기능입니다.

[그림 10.]과 같이 Print Filter Headers를 Check 하면 View의 Filter Headers가 같이 출력됩니다.

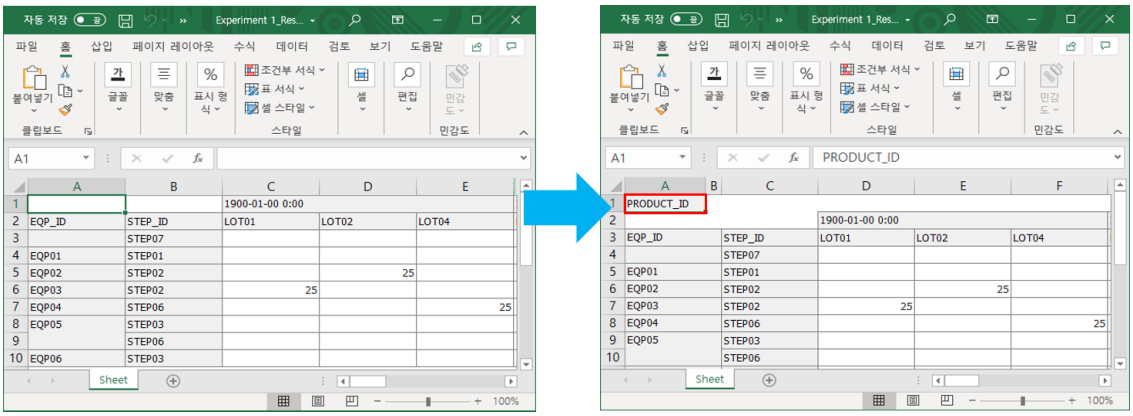


그림 10. Print Filter Headers 옵션 비교

Print Column Headers

Excel Export 기능 사용 시 View의 Column Headers를 같이 Export 할 지 선택할 수 있는 기능입니다.

[그림 11.]과 같이 Print Column Headers를 Check 하면 View의 Column Headers가 같이 출력됩니다.

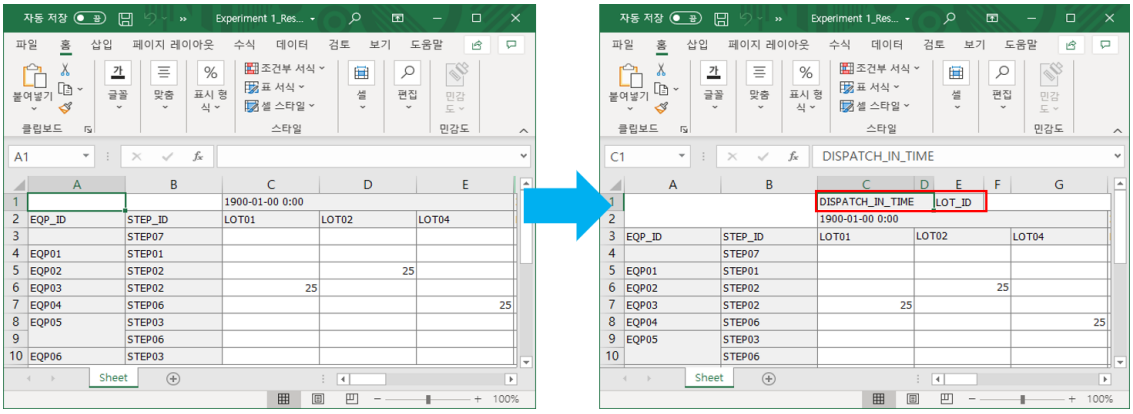


그림 11. Print Column Headers 옵션 비교

Show Row Header

Pivot Grid View에서 Row Header 부분에 속하는 Column Name을 보여주는 기능입니다. Show Row Header의 Check를 해제하고 Export to Excel을 실시하면 Row Header가 출력되지 않습니다.

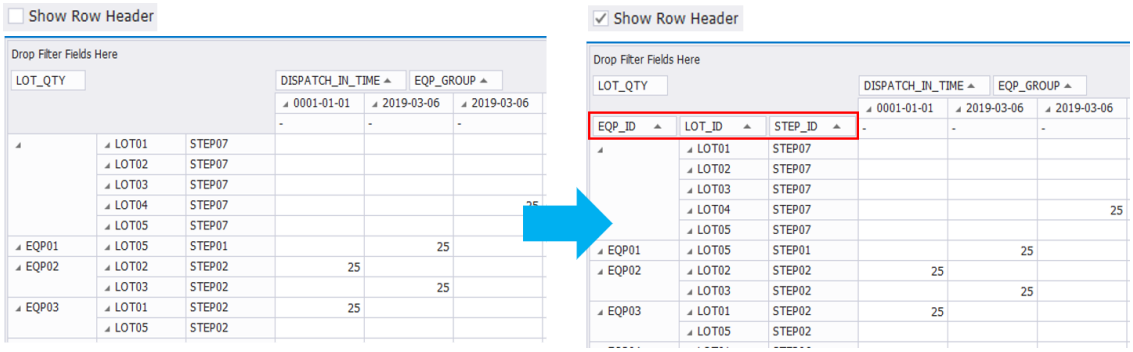


그림 12. Show Row Header 옵션 비교

Show Row Totals

View에서 각 Row의 Total을 보여주는 기능입니다.

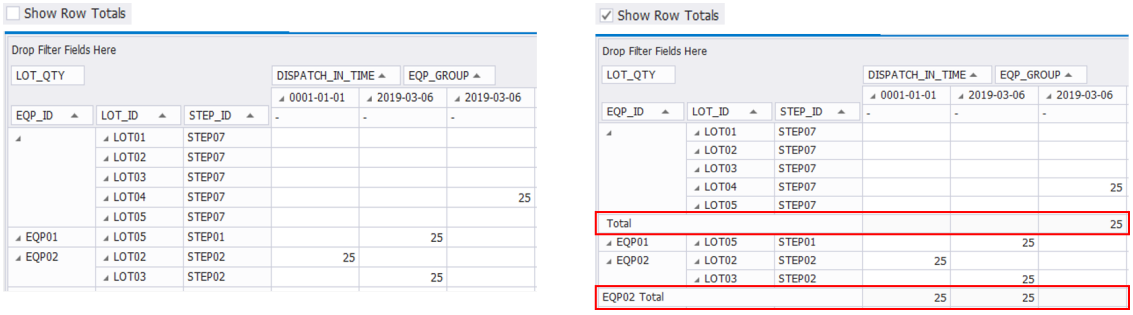


그림 13. Show Row Totals 옵션 비교

Show Row Grand Totals

View에서 Row의 Grand Total을 보여주는 기능입니다.

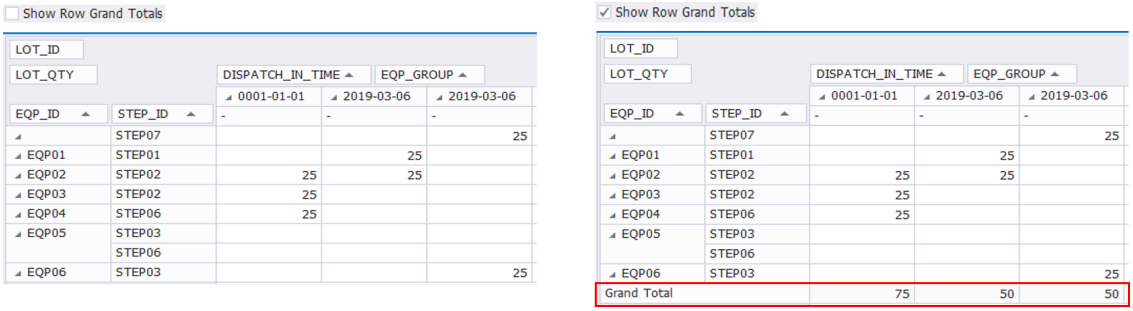


그림 14. Show Row Grand Totals 옵션 비교

Row Totals Location

View에서 Row의 Total or Grand Total의 위치를 지정할 수 있는 기능입니다.

- Far : Total의 위치가 Row의 하단에 위치하게 됩니다.
- Near : Total의 위치가 Row의 상단에 위치하게 됩니다.
- Tree : Total이 Tree 형식으로 위치하게 됩니다.

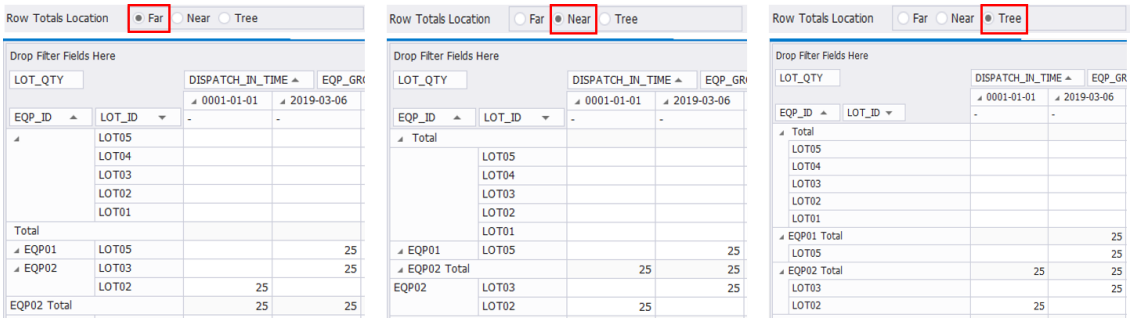


그림 15. Row Totals Location 옵션 비교

Show Column Header

Pivot Grid View에서 Column Header 부분에 속하는 Column Name을 보여주는 기능입니다.

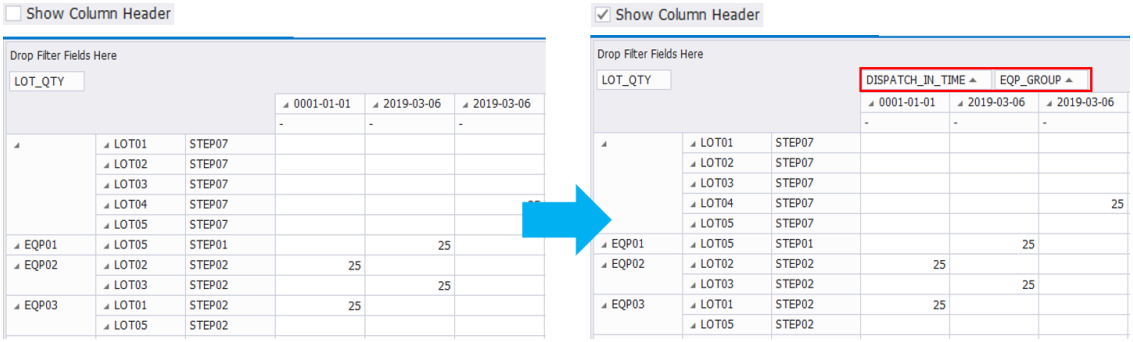


그림 16. Show Column Header 옵션 비교

Show Column Totals

View에서 Column의 Total을 보여주는 기능입니다.

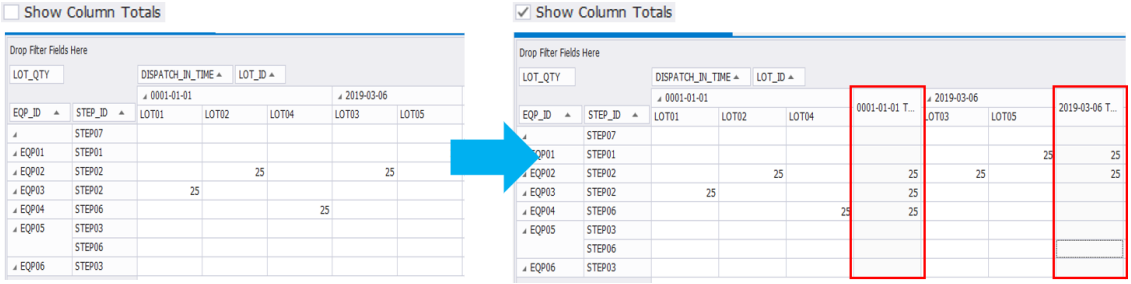


그림 17. Show Column Totals 옵션 비교

Show Column Grand Totals

View에서 Column의 Grand Total을 보여주는 기능입니다.

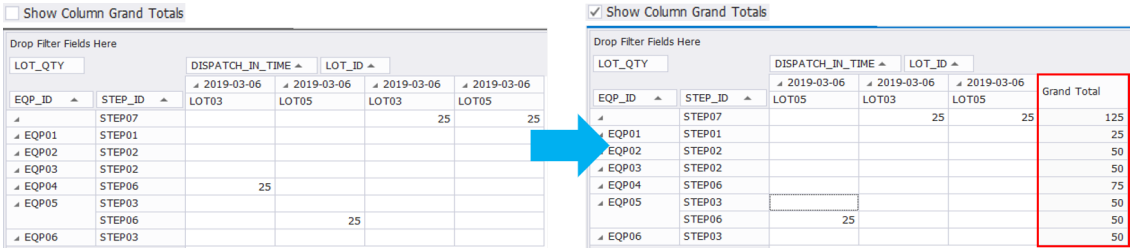


그림 18. Show Column Grand Totals 옵션 비교

Column Totals Location

View에서 Column의 Total or Grand Total의 위치를 지정할 수 있는 기능입니다.

- Far : Total의 위치가 Column의 우측에 위치하게 됩니다.
- Near : Total의 위치가 Column의 상단에 위치하게 됩니다.

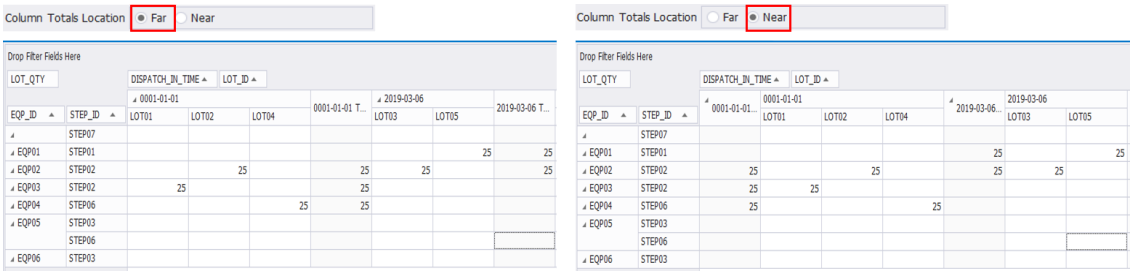


그림 19. Column totals Location 옵션 비교

Show Filter SeparatorBar

Filter Fields Header 부분과 Data, Column, Row 등의 Header 부분을 분리하는 Separator Bar를 나오도록 설정할 수 있는 기능입니다. **[그림 20.]**과 같이 Show Filter SeparatorBar Check 시 Filter Fields 부분 밑에 선이 나타나게 됩니다.

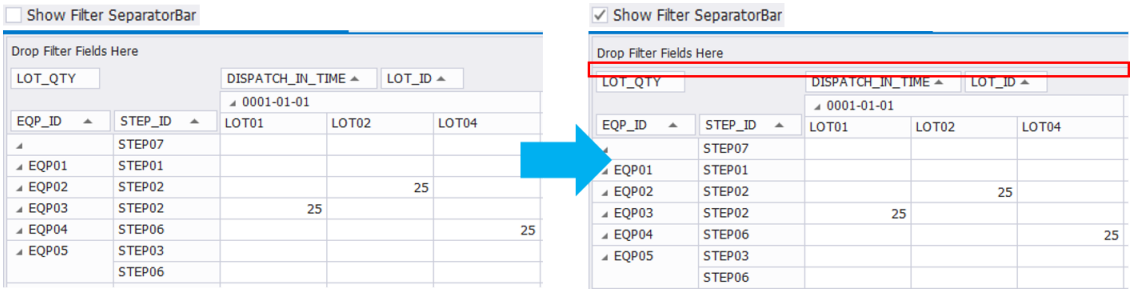


그림 20. Show Filter SeparatorBar 옵션 비교

Sort By Summary Default Order

Pivot Grid View에서 Column Area와 Row Area에 있는 데이터 값에 대해 오름차순(ASC)/내림차순(DESC) 정렬에 대한 기본 방식을 지정하는 기능입니다. 기본 값은 오름차순(ASC)이며, 정렬을 원하는 기준을 변경해서 사용할 수 있습니다.

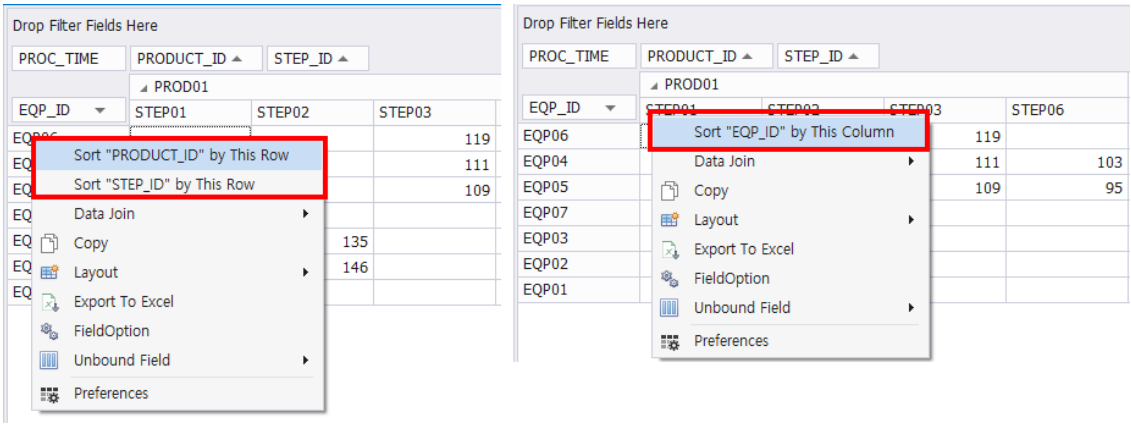


그림 21. Sort By Summary Default Order

Copy To Clipboard With Field Values

Pivot Grid View에서 Data를 복사할 때 Data가 속하는 Row Header 부분의 Column Name과 Column Header부분의 Column Name을 함께 복사할 지의 여부를 결정하는 기능입니다.

[그림 22.]과 같이 Copy To Clipboard With field Values를 Check하면 해당 Data들의 Row Header, Column Header의 Column Name들이 함께 복사됩니다. Check하지 않으면 해당 Data들만 복사됩니다.

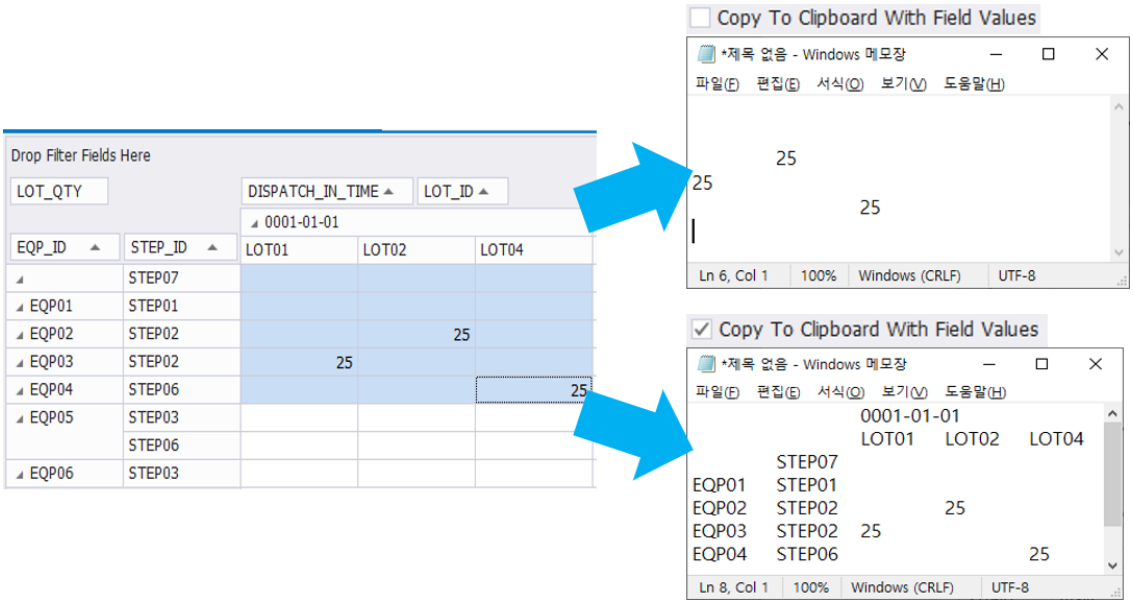


그림 22. Copy To Clipboard With Field Values 옵션 비교

그리드의 데이터 복사에 대한 자세한 내용은 [Grid View 사용법의 복사 기능](#)을 참고하시기 바랍니다.

Pivot Grid View 차트 기능

기능 개요

Open Item(Read Only-Pivot)을 통해 만든 Pivot Grid 데이터는 **[Show Chart]** 기능을 이용하여 차트로 표현 할 수 있습니다. 그리드 형태의 데이터를 분석하기 어려울 때 별도의 데이터 가공과 UI 개발 없이 간단하게 차트로 표현하고자 할 때 사용하기 유용한 기능입니다.

사용 조건

[Show Chart] 기능을 사용하기 위한 조건은 다음과 같습니다.

- Mozart Studio 버전이 2021.119.2 이상
- Open Item(Read Only-Pivot)으로 Pivot 그리드 생성

메뉴 설명

[Show Chart] 를 실행하면 상단 툴바에 다음 메뉴들이 노출되며 각각 메뉴의 정의와 구성은 다음과 같습니다.

Chart Type 메뉴

[Show Chart]에서 기본으로 제공하는 차트 유형입니다. 차트 유형으로는 Bar, Line, Point, Area가 있으며 Line 차트가 기본 유형입니다.

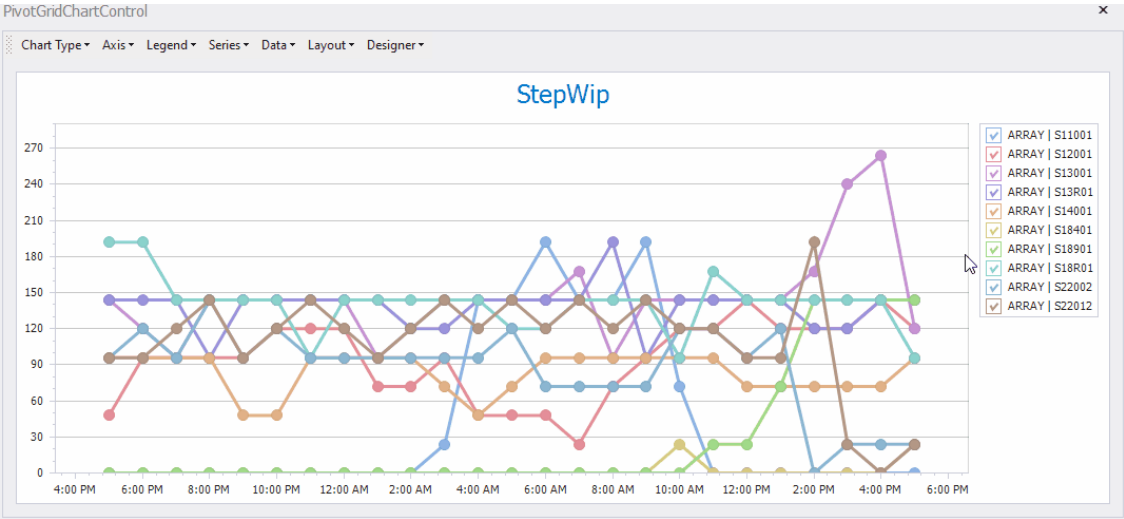
가로 축은 Pivot Grid Field의 Column Area/Data Area 항목을 기준으로 하고 있으며, Row Area 기준으로 항목을 묶어서 세 Data Area의 값에 세로 축 그래프가 그려집니다.

- **Bar**: 세로 막대형 차트를 그립니다.
- **Line**: 표식이 있는 꺾은선형을 그립니다.
- **Point**: 분산형 차트를 그립니다.
- **Area**: 영역형 차트를 그립니다.
- **Rotate**: 현재 차트의 x와 y 축의 위치를 변경합니다. 즉 세로 차트의 경우, 가로 차트로 그려집니다.

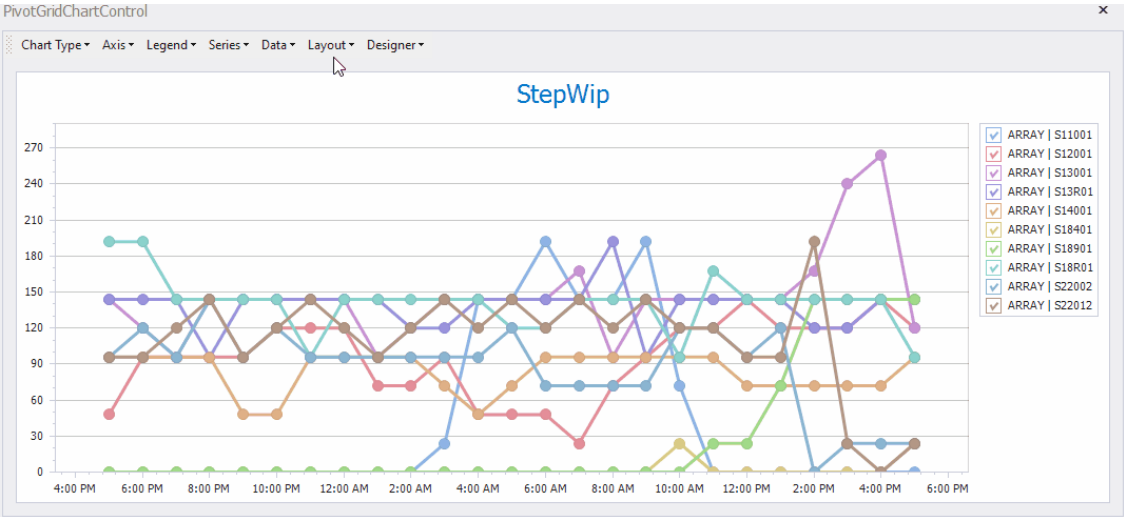
Axis 메뉴

차트의 x,y 축 설정 관련 메뉴입니다. 해당 메뉴는 x,y축이 있는 차트 한해서 적용됩니다.

- **Show Axis-X**: x축 표시 여부입니다. 기본은 true입니다.
- **Show Axis-Y**: y축 표시 여부입니다. 기본은 true입니다.
- **Show Axis-Both**: x/y축 표시 여부입니다. 만약 **[Show Axis-X]**와 **[Show Axis-Y]** 둘 다 체크되어 있으면 **[Show Axis-Both]**는 자동으로 활성화 되며 그 반대의 경우 마찬가지입니다.
- **Rotate Axis-X(45°)**: x축 레이블을 시계방향으로 45도 회전합니다.
- **Enable Axis-X Zoom**: x 축을 기준으로 차트를 확대/축소 하는 옵션입니다. Enable Axis-Y Zoom과 동시 사용이 가능합니다.



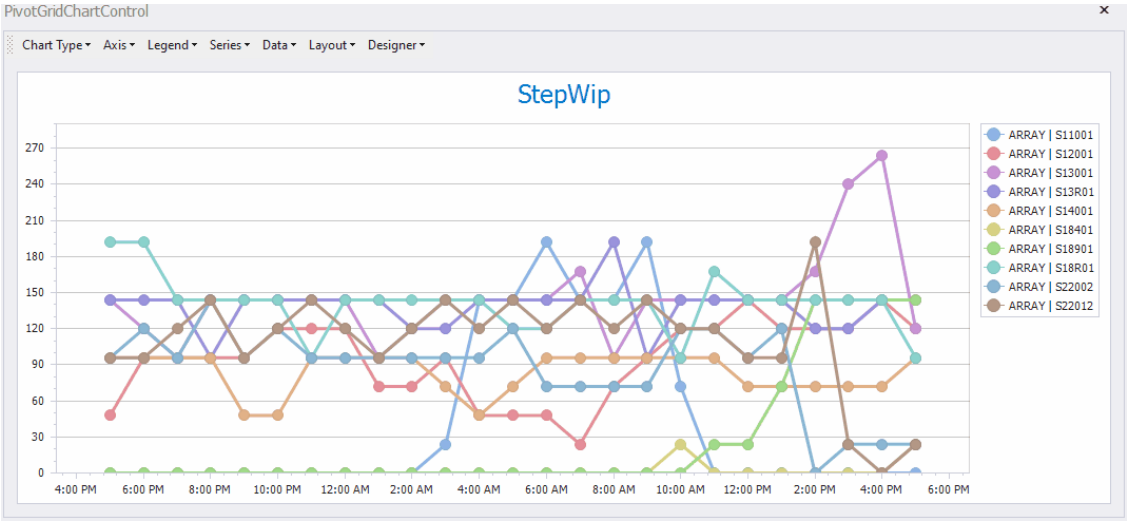
- **Enable Axis-Y Zoom**: y 축을 기준으로 차트를 확대/축소 하는 옵션입니다. Enable Axis-X Zoom과 동시 사용이 가능합니다.



Legend 메뉴

차트 범례를 설정하는 메뉴입니다.

- **Show Legend**: 범례 표시 여부입니다. 기본적으로 활성화 되어있습니다.
- **Only Marker**: 범례 표식 표시 여부입니다. **[Only Maker]**가 활성화되면 **[Only Check]**는 해제됩니다.
- **Only Check**: 차트에 표시할 범례 선택 체크박스 사용 여부입니다. **[Only Check]**가 활성화 되면 범례 표식 대신 체크박스가 노출되며, 체크되어 있는 범례만 차트가 그려집니다. 처음 차트 그릴 때는 **[Only Check]**가 활성화되어 있습니다.
- **Check Marker**: 체크박스/범례 표식 둘다 활성화되며 체크박스/범례 표식 순서로 표시됩니다.
- **Marker Check**: **[Check Marker]**와 동일하면 표시 순서만 범례 표식/체크박스 순으로 변경됩니다.

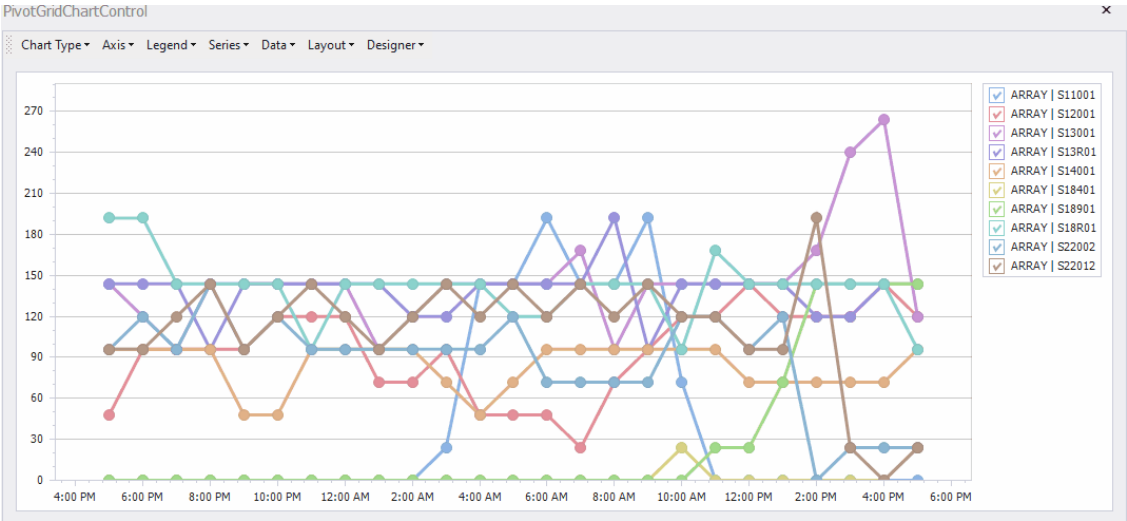


Legend 메뉴 사용 예시

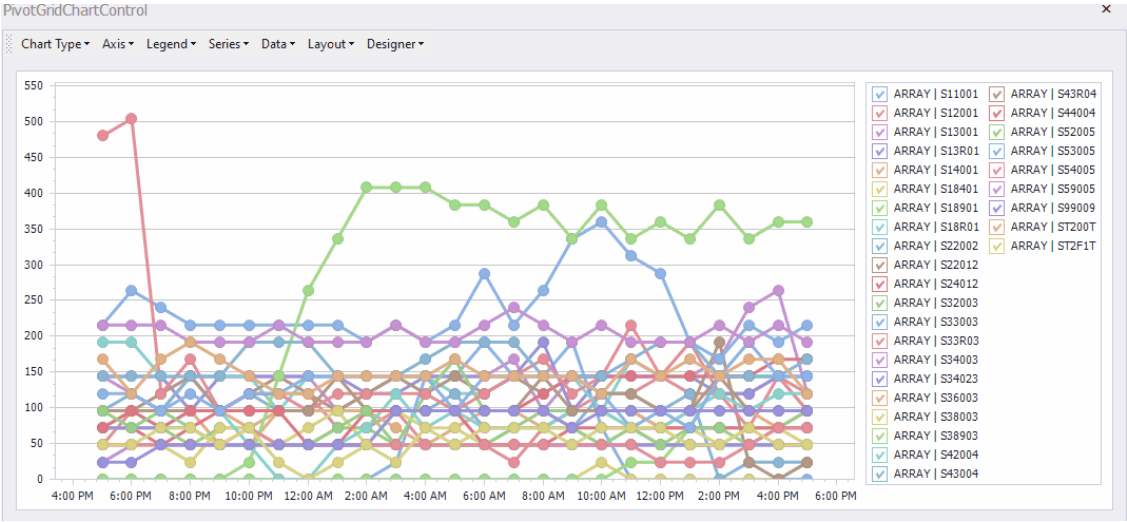
Series 메뉴

Pivot Grid에서 선택한 데이터의 최대 표현 개수 또는 표현 대상을 선택하는 메뉴입니다.

- **Max Allowed Series**: 차트에 표현할 최대 데이터 계열 개수입니다.
  - **Only n개 items**: Pivot Grid에서 선택한 데이터 계열 중 최대 n개만 차트에 그립니다. 선택 가능한 옵션은 10개, 30개 50개입니다.
  - **All items**: Pivot Grid에서 선택한 모든 데이터 계열을 차트에 그립니다.
  - **Custom**: Pivot Grid에서 선택한 데이터 계열 중 사용자가 지정한 개수만큼 차트에 그립니다.



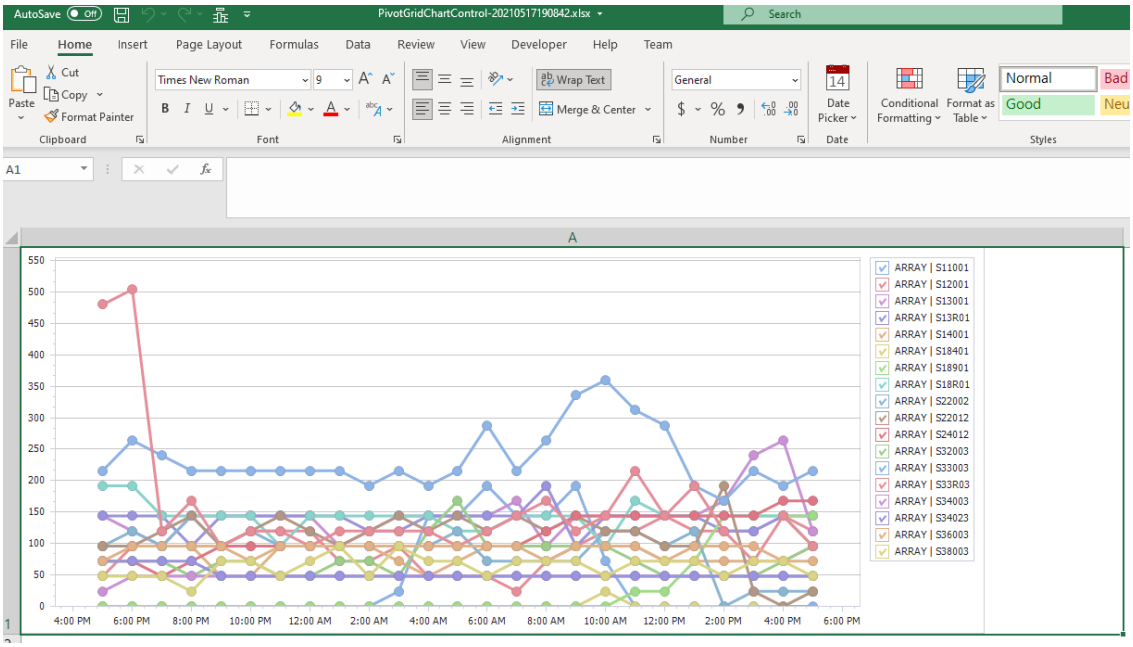
- **Show Series**: Pivot Grid에서 선택한 데이터 그리드 목록이며 사용자가 차트에 그릴 데이터 계열을 선택할 수 있습니다.



Data 메뉴

현재 그려진 차트를 추출할 형식을 설정하는 메뉴입니다. 다음과 같은 피일 형식으로 추출이 가능합니다.

- Excel
- Png
- HTML
- Rtf
- Pdf
- Mht



Excel 파일로 추출한 예

Layout 메뉴

차트의 레이아웃을 저장/불러오기/초기화하는 메뉴입니다. Pivot Grid 레이아웃과는 별개로 독립적인 레이아웃으로 저장되며, 레이아웃 파일은 *%appdata%\Mozart\XXXStudio\ViewInfo\Layout* 폴더에 ***_Chart.xml** 형식으로 저장됩니다.

- **Save Layout**: 현재 차트의 레이아웃을 저장합니다.
- **Load Layout**: 이전에 저장한 차트의 레이아웃을 로딩합니다.

- **Reset Layout**: 기본 레이아웃으로 또는 저장된 레이아웃이 있는 경우 저장된 레이아웃으로 초기화합니다.

Designer 메뉴

기본 제공되는 차트의 속성 설정 또는 기본 제공되는 차트 외 Pivot Grid에서 선택한 데이터를 표현할 차트 종류와 속성을 설정할 수 있는 디자이너 도구입니다. 차트 종류에 따라 설정 가능한 속성들이 다양하게 있습니다. 본 도구는 DevExpress의 [Chart Wizard 컨트롤](#)을 그대로 사용하고 있습니다.

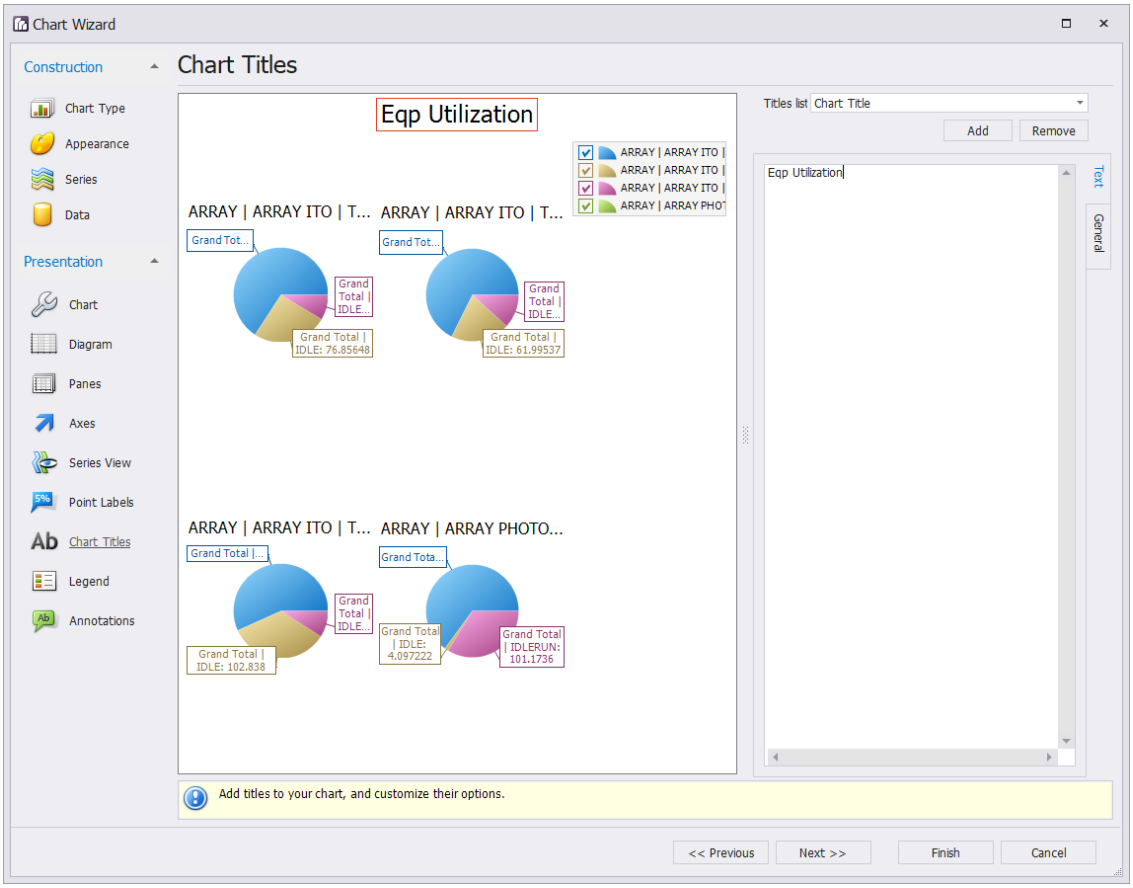


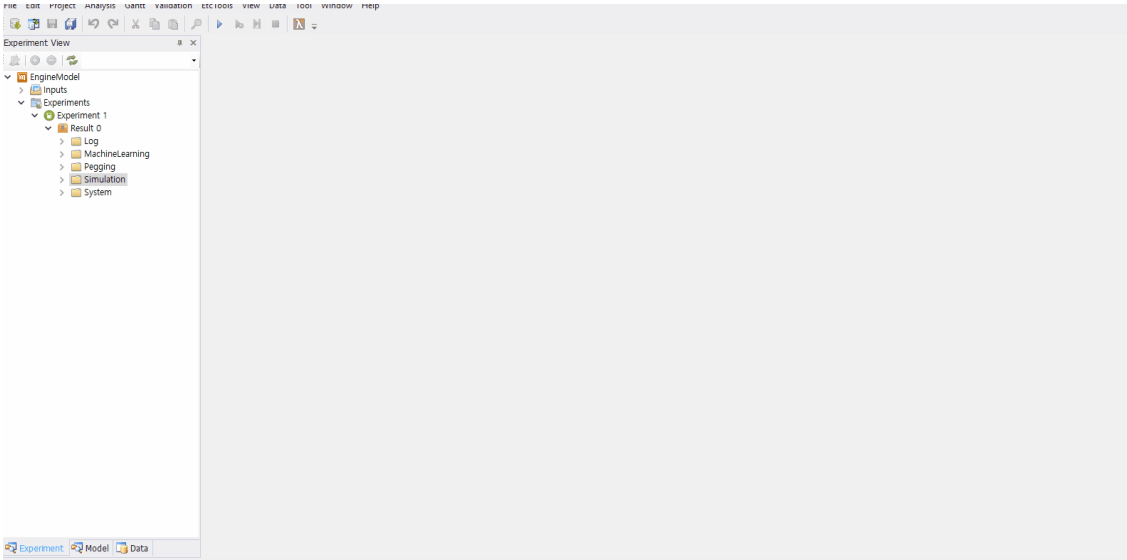
차트 종류 선택 및 속성 설정 예

> **Note:**
> 모든 메뉴는 상단 툴바 외에 차트에서 마우스 우측 클릭 시 컨텍스 메뉴에서도 선택이 가능합니다.

사용 방법 및 예제

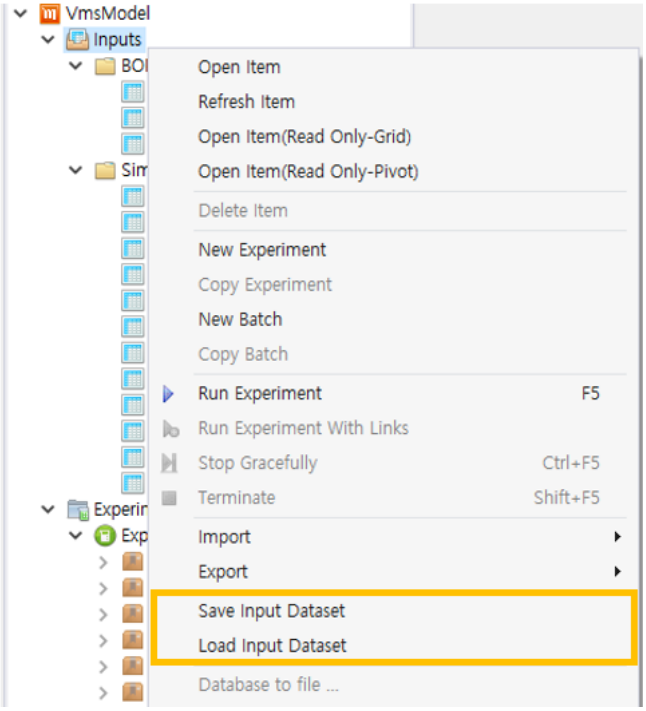
다음은 Pivot Grid의 데이터를 차트로 표현하기 위한 방법입니다. 본 문서에서는 선형 차트를 그리는 예제를 보여드립니다.

1. 차트로 표현할 DataItem을 [Model View]에서 선택하여 Pivot 뷰를 만들기 위해 컨텍스트 메뉴에서 **[Open Item(Read-Only Pivot)]**을 선택합니다.
2. **[PivotGrid Field List]**에서 차트를 그리기 위해 필요한 컬럼들을 Filter Area, Column Area, Row Area, Data Area에 설정합니다.
3. 생성된 Pivot 뷰에서 차트로 표현할 데이터를 마우스로 드래그하여 선택합니다.
4. 선택이 완료되면 마우스 우 클릭으로 컨텍스트 메뉴를 열어서 **[Show Chart]**를 실행합니다.



Input Dataset 기능

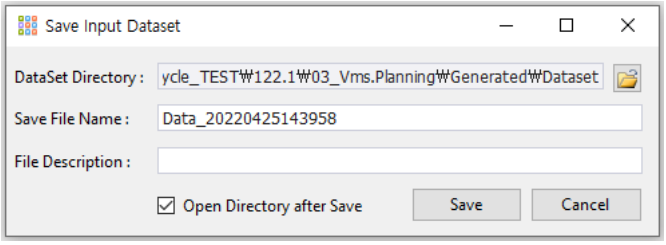
모델의 Inputs 컨텍스트 메뉴 **[Save Input Dataset]**, **[Load Input Dataset]**를 통해, 한 모델에 대한 기준정보 데이터(Input Data)를 여러 버전으로 저장하고 불러올 수 있습니다. 해당 기능은 MOZART 2022.122.1 버전부터 사용 가능합니다.



Save Input Dataset

모델이 현재 사용하고 있는 Input Data (모델 파일과 동일한 위치에 있는 *Data* 폴더 내 ***.vdata** 파일)를 압축하고 저장하는 기능입니다. 데이터를 압축하고 저장하는 과정에서 압축파일을 저장하기 위한 *Dataset* 폴더 그리고 압축파일의 정보를 가지고 있는

DataInfos.xml 파일이 생성됩니다.



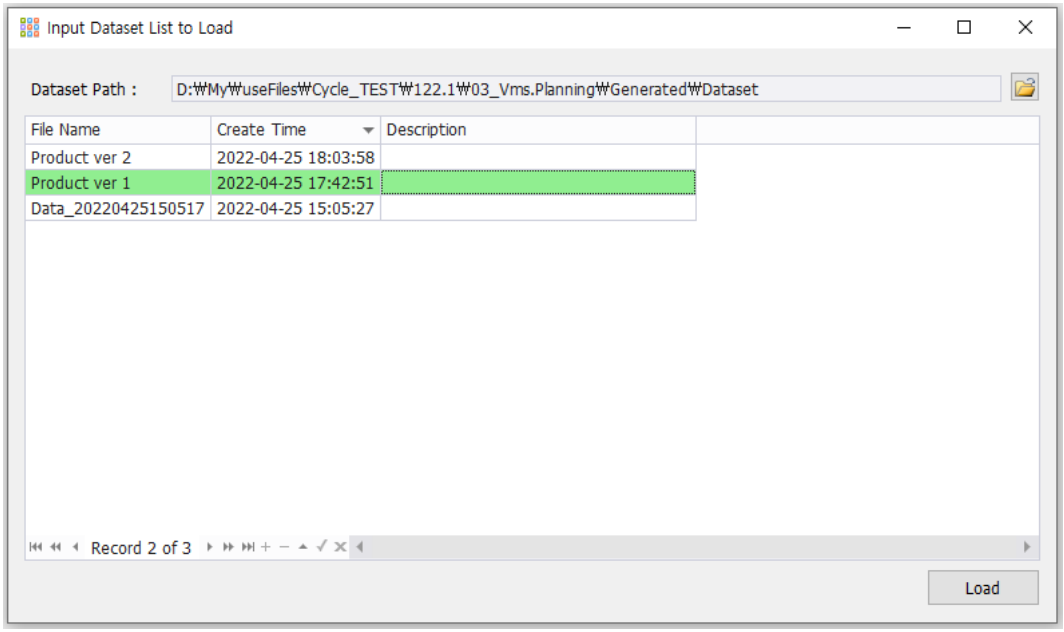
- **Dataset Directory** : Dataset 파일의 저장 경로입니다. 저장 경로는 고정이며 현재 열려있는 모델 파일의 위치에 Dataset 폴더가 생성됩니다. 좌측의 📂 버튼을 누르면 **[Dataset Path]**를 파일탐색기로 엽니다.
- **Save File Name** : 저장할 파일 명입니다. (기본형식 : Data_생성시간)
- **File Description** : 저장하는 Dataset에 대한 설명입니다.
- **Open Directory after Save** : 저장 후 *Dataset* 폴더를 파일 탐색기로 여는 여부입니다.

Dataset 생성/저장하는 방법

1. 모델의 **Inputs** 노드를 우클릭하여 컨텍스트 메뉴에서 **[Save Input Dataset]**를 선택합니다.
2. **[Save File Name]**에서 Dataset 파일 명을 지정합니다. (기본형식 : Data_생성시간)
3. (선택) Dataset에 대한 설명을 **[File Description]** 항목에 입력합니다.
4. (선택)저장 후 Dataset 폴더로 바로 이동하려면 **[Open Directory after Save]**에 체크합니다**.**
5. **[Save]**를 선택하여 Dataset 압축 파일을 저장합니다.

Load Input Dataset

모델의 Dataset 목록에서 선택된 Input Dataset을 로딩 합니다. 녹색으로 표시된 행은 모델이 현재 사용하고 있는 Dataset이며, 모델 실행 후 생성되는 Result 정보에서 Dataset 정보 확인이 가능합니다.



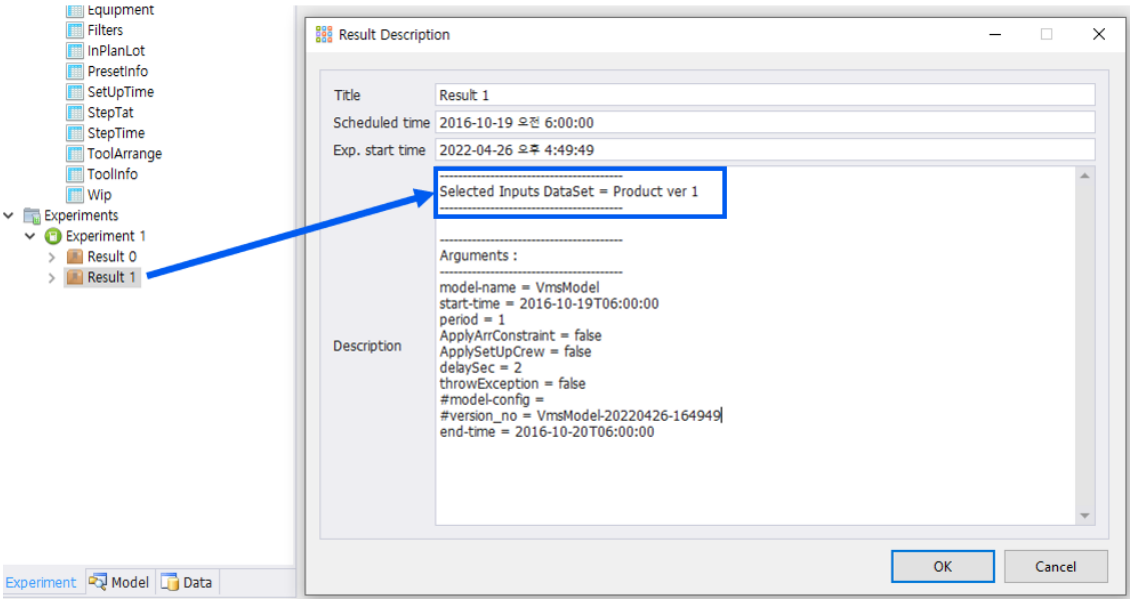
- **Dataset Path** : Dataset 파일이 위치한 경로입니다. 좌측의 📂 버튼을 누르면 **[Dataset Path]**를 파일탐색기로 엽니다.
- **File Name** : Dataset 파일명입니다.
- **Create Time** : Dataset이 생성된 시간입니다.
- **Description** : Dataset에 대한 설명입니다.
- **Load** : 목록에서 선택한 Dataset을 로딩합니다.

Dataset에 있는 Input Data 로딩하는 방법

1. **Inputs** 노드를 우클릭하여 컨텍스트 메뉴에서 **[Load Input Dataset]**을 선택하여 Dataset 목록을 불러옵니다.
2. Dataset 목록에서 로딩할 Dataset의 행을 마우스 좌측 버튼 더블 클릭 또는 행을 선택한 다음 **[Load]** 버튼을 눌러 Dataset을 로딩합니다. 로딩된 Input Dataset의 행은 Dataset 목록에서 녹색으로 표시됩니다.

Result 별 적용된 Dataset 확인

각 Result 별로 상세 내용을 조회(Result N 노드 마우스 왼쪽 버튼 더블 클릭)하면 `Selected Inputs Dataset` 항목에 모델에서 사용한 Dataset 이름이 표시됩니다.



주의 사항

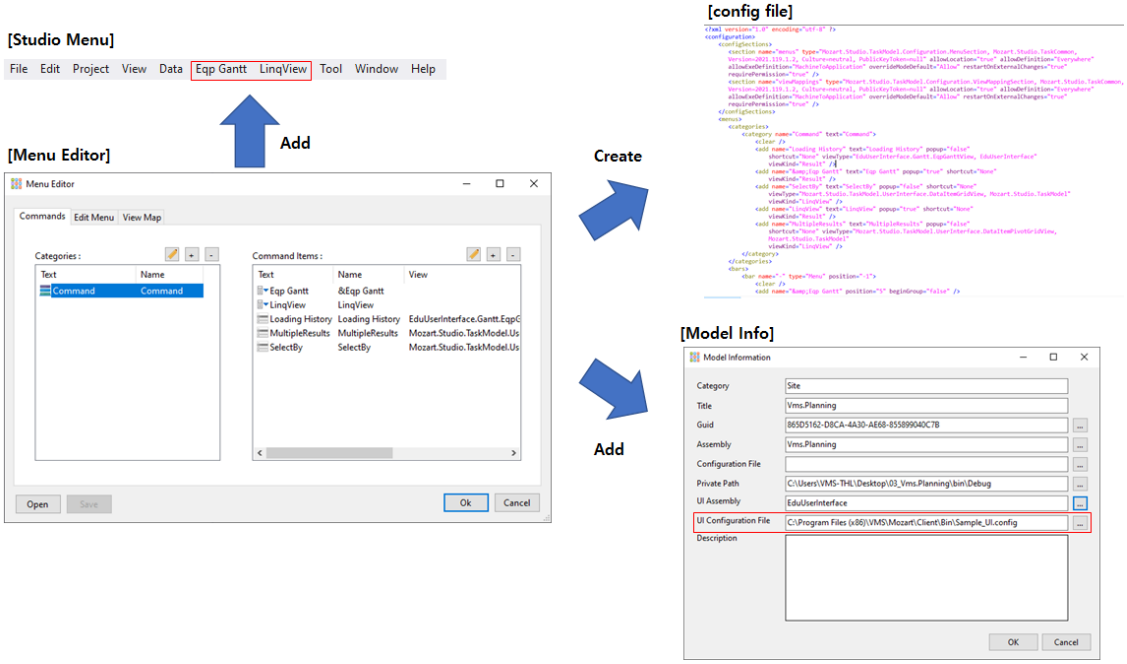
1. Mozart Studio에서 직접 추가한 DataItem은 Dataset 저장/로딩 대상이 안됩니다.
2. Dataset 파일의 수정 또는 삭제하기 위해서는 Dataset 폴더에서 파일을 직접 수정하거나 삭제해야합니다. 수정 또는 삭제 된 파일 정보는 **DatasetInfos.xml**에 자동으로 업데이트 됩니다.

메뉴 등록

개요

Mozart Studio에는 Menu Editor라는 도구를 이용하여 메뉴를 등록할 수 있습니다. 예를 들어 사용자가 개발한 UI, LINQPad로 구현한 Mlinq UI, LINQ View를 메뉴로 등록할 수 있습니다. 또한 **Menu Editor**는 모델 데이터의 기본 View를 Grid가 아닌 사용자가 개발한 UI로 변경할 수 있는 기능도 제공합니다.

Menu Editor로 메뉴를 등록하면 모델에서 참조하는 config 파일이 생성이 되고 모델 정보(**vinfo**)에 등록됩니다. 이 config 파일은 다른 모델에서 등록하고 사용하거나 혹은 메뉴 편집 때 필요한 파일입니다. 아래 그림은 **Menu Editor**를 통해 메뉴를 등록했을 때 나타나는 결과입니다.



기능 설명

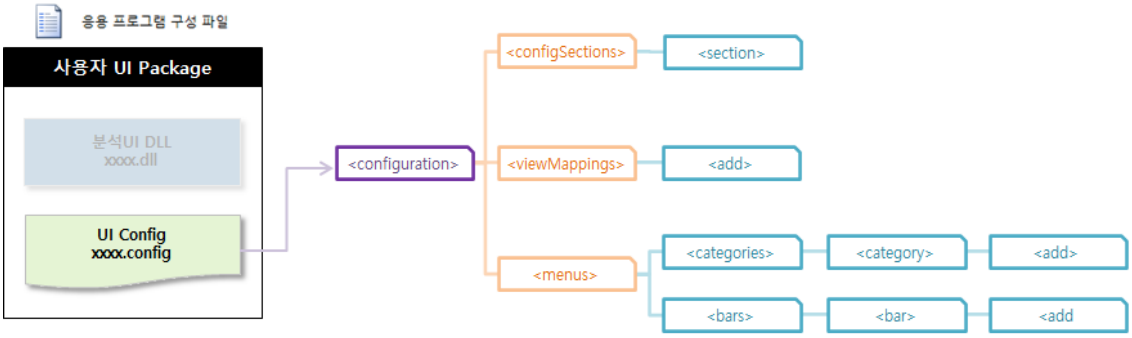
Menu Editor는 메뉴를 정의하는 **[Commands]**, Studio의 메뉴를 구성하고 관리하는 **[Edit Menu]**, 그리고 DataItem의 기본 View를 다른 UI로 설정하는 **[View Map]** 이렇게 3개의 영역으로 분류됩니다.

Menu Editor를 통해 생성되는 Config 파일은 xml 문서입니다. 이 파일은 사용자가 직접 만들어도 되고 아니면 **Menu Editor**를 통해 생성해도 됩니다.

본 문서에서는 Menu Editor의 기능, Menu Editor의 각 영역과 Config 파일과의 관계, 그리고 Config 파일의 속성에 대해 설명합니다.

Config 파일 구성 요소

사용자가 개발한 UI DLL, MLinq, Linq View를 Mozart Studio와 연결시켜주는 역할을 하며 기본 구조는 다음과 같습니다.



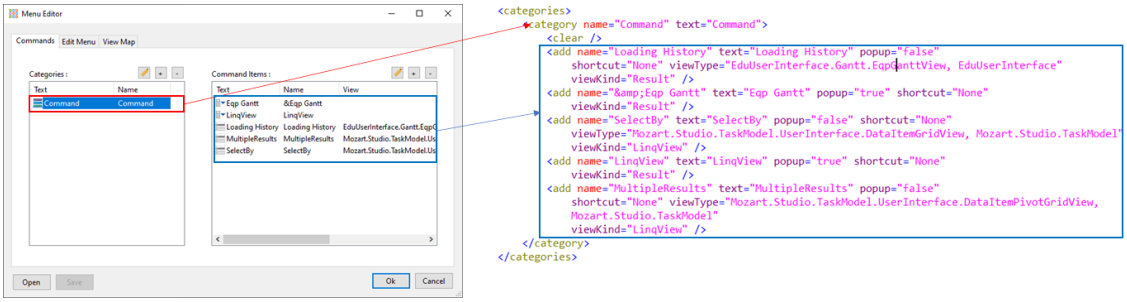
Config 파일은 .NET Framework에서 제공하는 응용 프로그램 구성 파일로 처음 생성 시 `<configuration>` 요소 하나만 있으며 여기서 Menu Editor를 통해 나머지 요소들을 추가합니다. 각 구성 요소 및 속성에 대한 정의는 아래 페이지를 참고 바랍니다.

Menu Editor 구성 요소

Menu Editor의 각 영역별 기능 및 Config 파일과의 관계를 설명합니다.

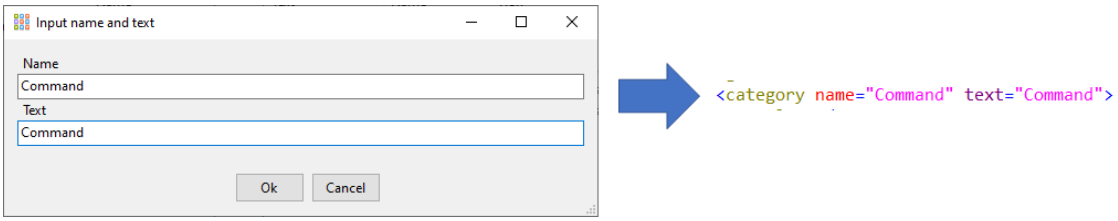
Commands

Commands 영역은 Mozart Studio에 등록할 메뉴를 정의하는 영역입니다. Config 파일의 `<categories>`, `<category>`를 정의합니다.



Categories 메뉴

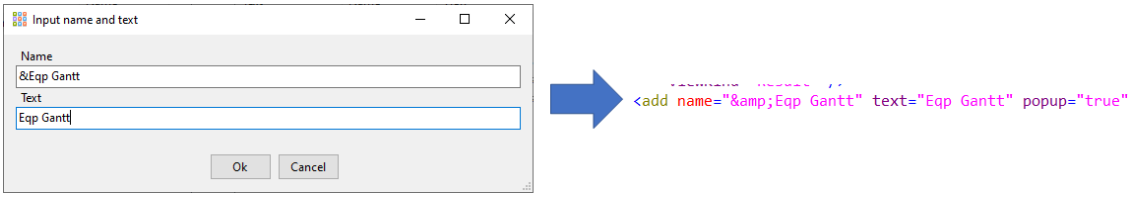
`<category>`의 `name`과 `text`를 정의하는 메뉴입니다.



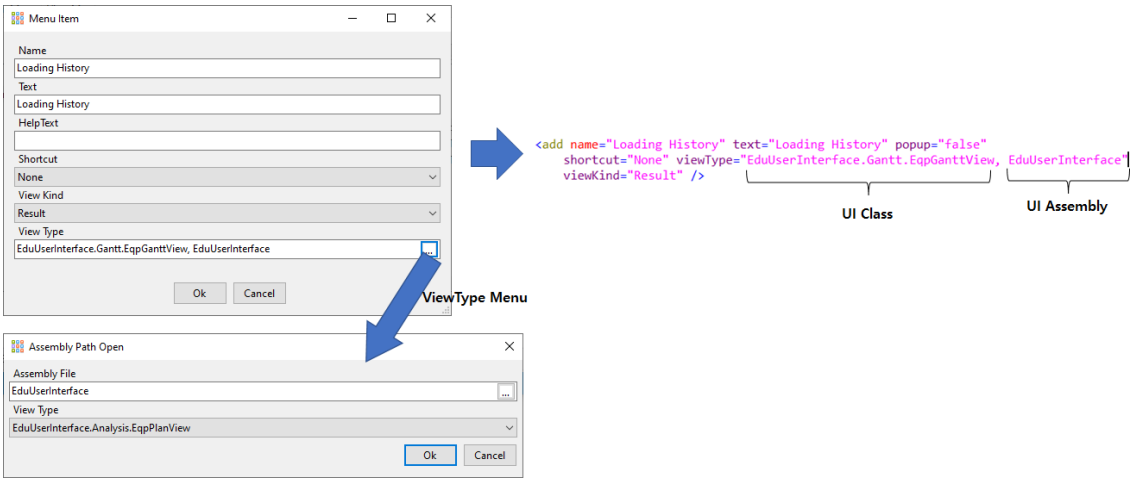
Command Items 메뉴

`<category>`에 추가할(`<add>`) 메뉴를 정의하는 영역이며, 팝업 메뉴를 추가하는 **[Add Popup]**과 메뉴를 추가하는 **[Add Menu]**로 구성되어 있습니다. **[Add Popup]**으로 추가되는 메뉴는 `popup` 속성이 `true`로 설정됩니다.

[Add Popup]

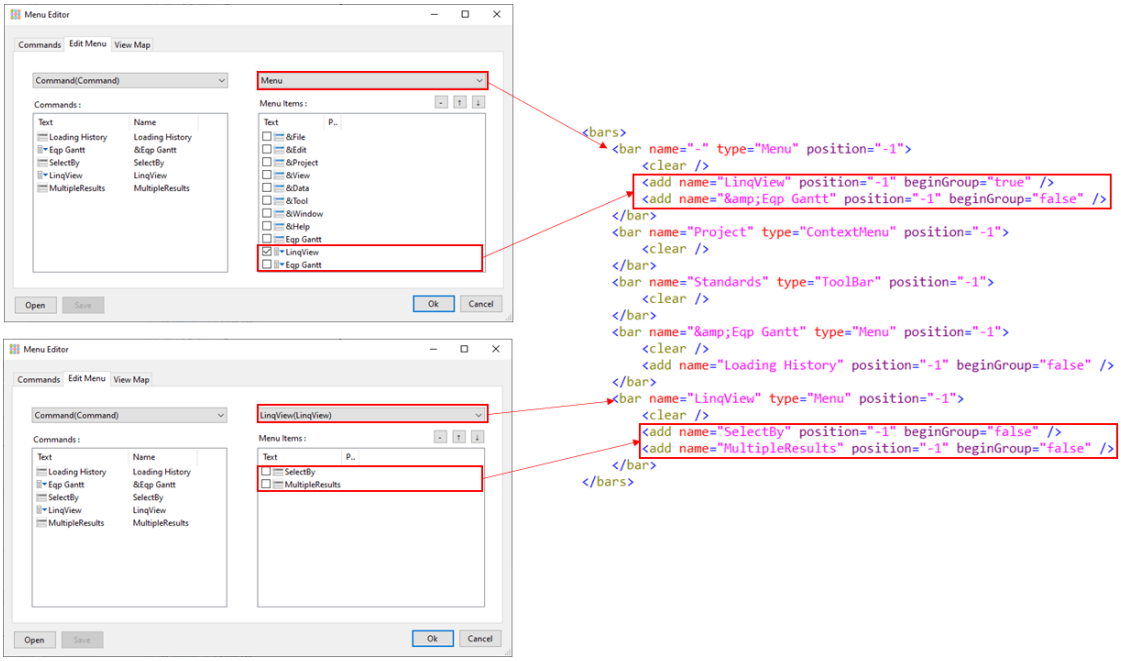


[Add Menu]

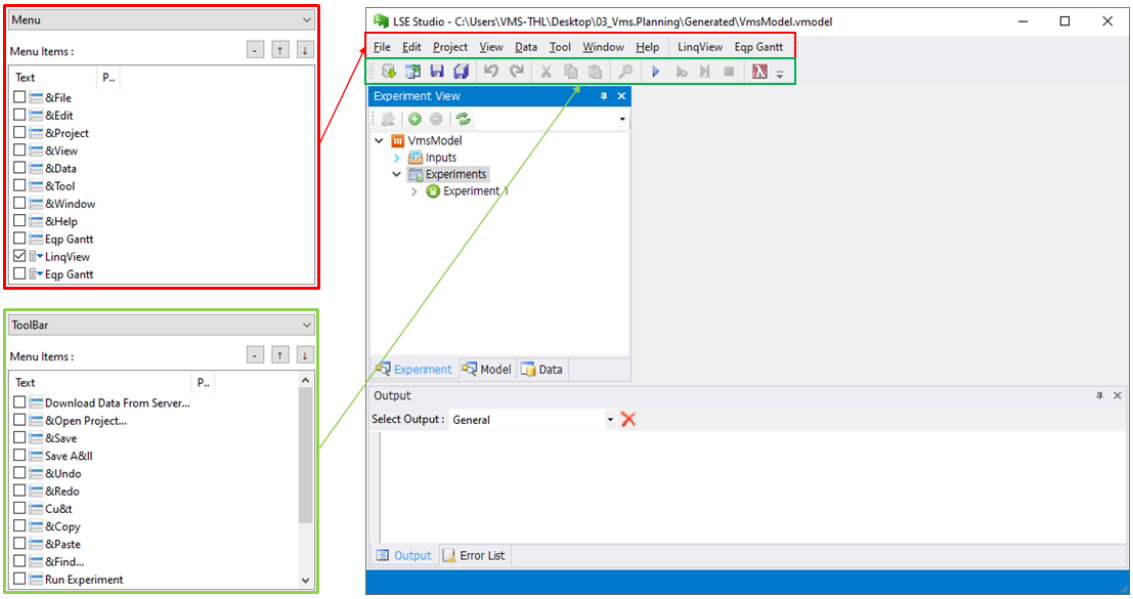


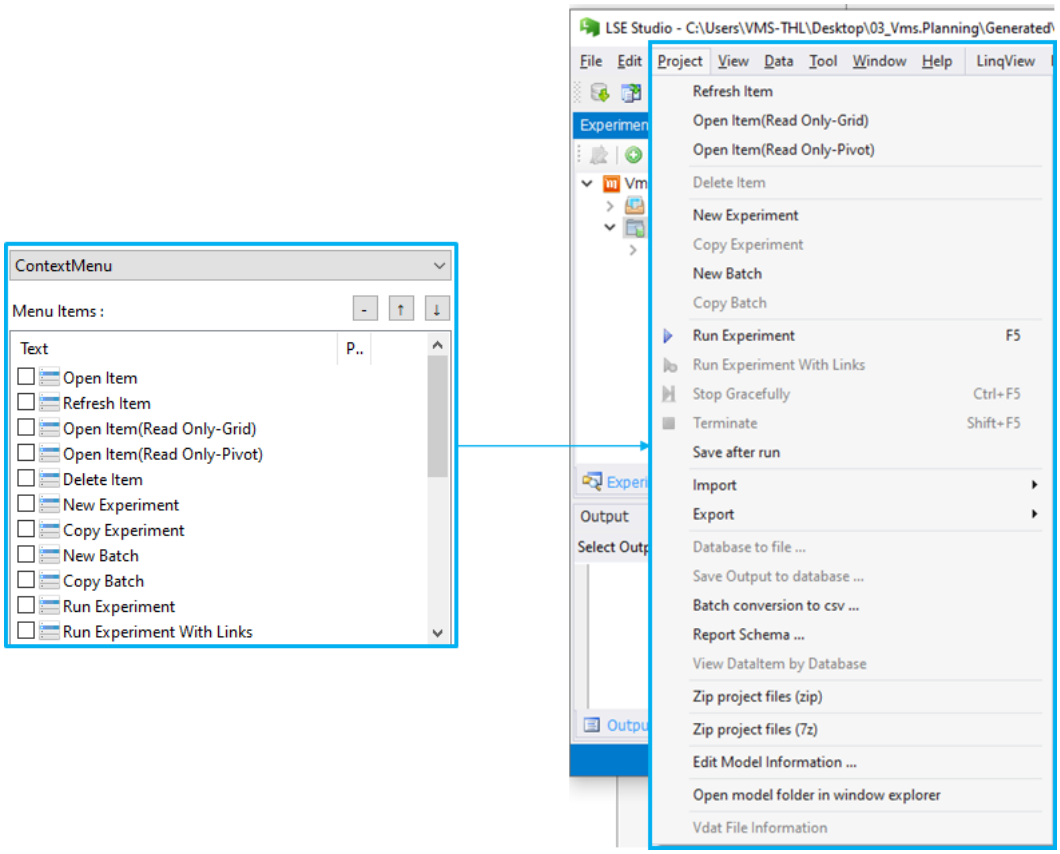
Edit Menu

[Edit Menu]는 **[Command]**에 정의한 메뉴를 Mozart Studio 메뉴에 등록하는 영역입니다. 여기서 메뉴 그룹핑, 순서, 등록 위치 등을 설정할 수 있습니다. Config 파일에서는 `<bars>`, `<bar>` 요소에 해당됩니다.



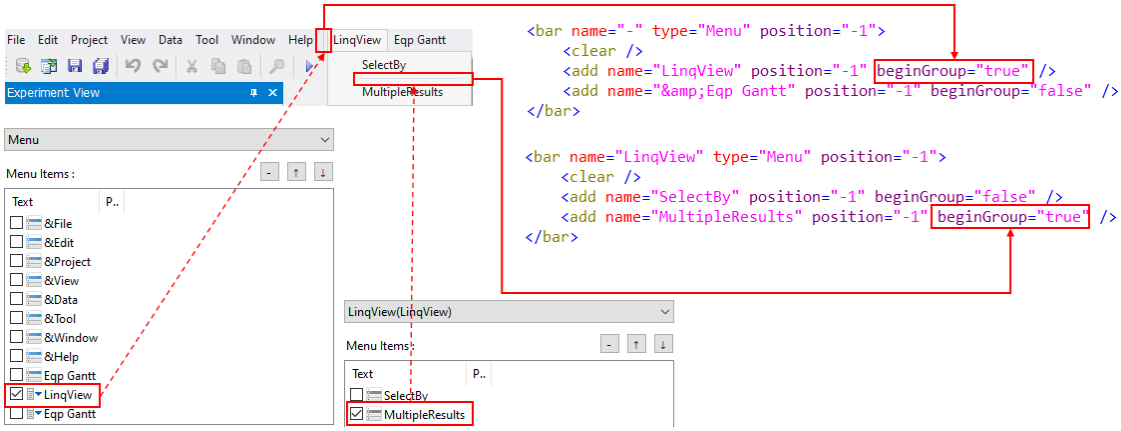
사용자가 정의한 메뉴는 Mozart Studio의 **[Menu]**, **[Context Menu]**, 그리고 **[Tool Bar]**에 등록할 수 있습니다. **[Menu], [Tool Bar], [Context Menu]**의 Mozart Studio에서의 위치는 다음과 같습니다.





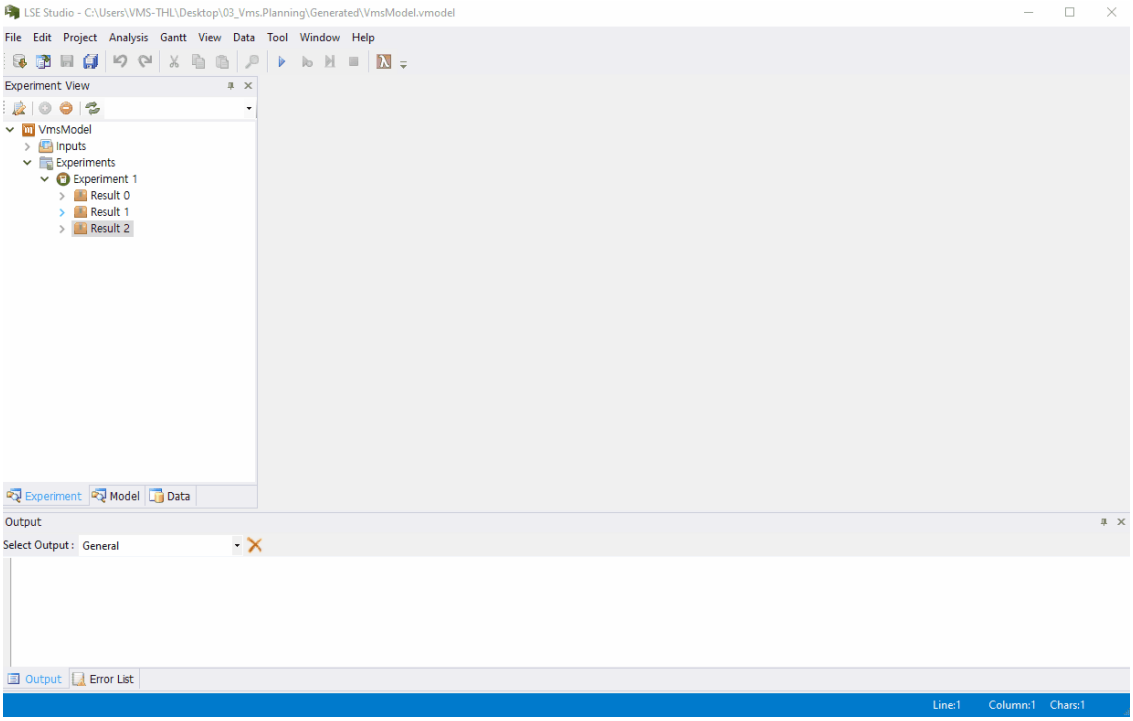
beginGroup 속성

`beginGroup` 은 메뉴간 경계선 추가 여부를 설정하는 속성입니다. 예를 들어 Mozart Studio와 사용자가 추가한 메뉴간 구분을 위해 경계선을 추가할 수 있으며, 팝업메뉴 하위의 메뉴들 중 유사한 용도로 구분하기 위해서 추가할 수 있습니다. 이때 경계선의 시작점이 되는 메뉴의 `beginGroup=true` 로 설정하면 해당 메뉴 앞에 경계선이 추가됩니다. **Menu Editor**에서는 체크 박스를 체크하면 `beginGroup=true` 로 설정됩니다.



View Map

[View Map]은 메뉴 등록과는 별개로, 특정 데이터 테이블을 조회 할 때 Grid가 아닌 사용자 UI, Mlinq, 또는 Linq View로 기본 View를 변경하고자 할 때 설정하는 영역입니다. 아래는 **[View Map]**을설정했을 때 결과 화면입니다.



즉, 메뉴로 등록한 UI를 실행하지 않고 항상 지정된 UI로 데이터 테이블을 조회하고 할 시에는 **[View Map]**에 등록하면 됩니다. **[View Map]**에 추가된 데이터 테이블은 아래와 같이 Config 파일의 `<viewMappings>` 요소에 내용이 추가됩니다.

사용 방법 및 예제

메뉴 등록 방법을 설명합니다. 본 문서에서 사용한 예제는 아래 링크를 통해 받을 수 있습니다.

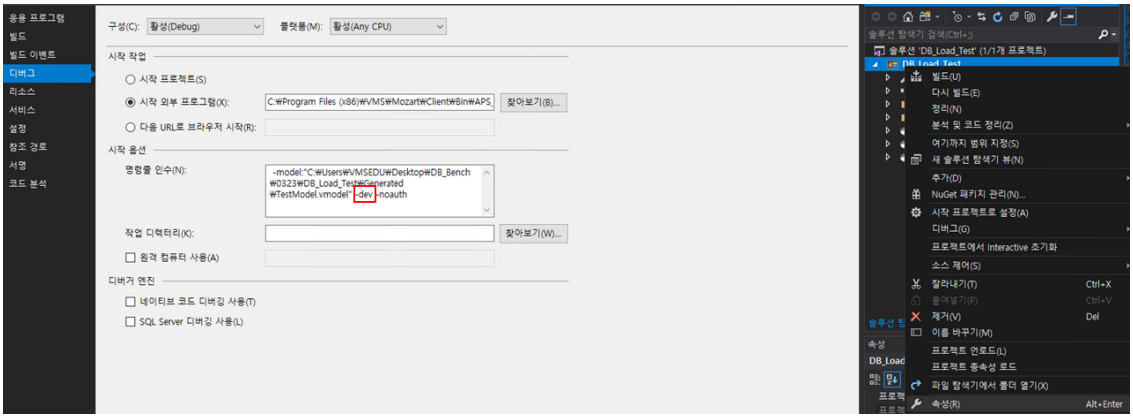
[샘플 다운로드]

메뉴 등록 준비 사항

[Menu Editor]를 사용하기 위해서는 Mozart Studio를 **"개발자"** 옵션으로 실행해야합니다. 개발자 옵션으로 Mozart Studio를 실행하는 방법은 다음과 같습니다.

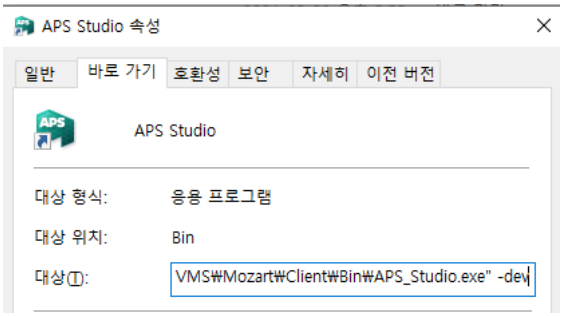
[Mozart IDE에서 설정 방법]

1. 솔루션 탐색기에서 프로젝트에 마우스 우 클릭으로 콘텍스트 메뉴를 실행하여 [속성] 메뉴를 선택합니다.
2. **[디버그]** 탭으로 이동 후, **[시작 옵션]**의 **[명령줄 인수(N):]**에서 맨 뒤에 **-dev**를 추가합니다.



[Mozart Studio 바로가기에서 설정 방법]

1. 윈도우 시작 메뉴(*C:\ProgramData\Microsoft\Windows\Start Menu\Programs\VMS\Mozart Client*)에서 Mozart Studio 바로가기의 속성을 엽니다.
2. **[바로 가기]** 탭에서 **[대상(T):]**의 맨 끝에 **-dev**를 추가합니다.



[UI DLL 파일, Mlinq 파일 저장 위치]

사용자 개발 UI나 MLinq를 메뉴로 등록하려면 관련 파일들이 사전에 준비되어 있어야 합니다. 일반적으로 이 파일들은 Mozart Studio가 실행되는 위치와 동일한 경로에 저장해 두는 것이 좋습니다. (*i.e. C:\Program Files (x86)\VMS\Mozart\Client\Bin*)

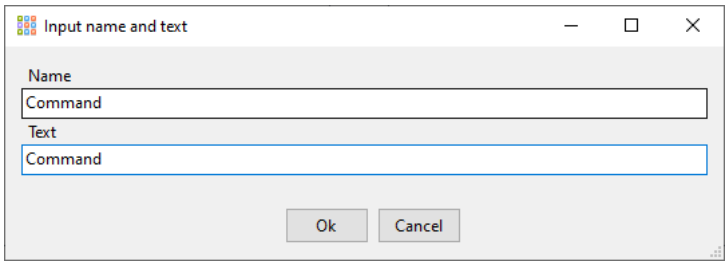
- 사용자 개발 UI에 필요한 파일: 개발한 UI DLL 파일, 모델 스키마.DLL 파일
- Mlinq: Mlinq 파일

메뉴 정의 (Commands 탭)

가장 처음에는 **Menu Editor**를 통해 메뉴 카테고리와 메뉴를 정의합니다. **Menu Editor**는 Mozart Studio를 개발자 옵션으로 실행하여 **[Tool] > [Edit Menus..]**를 선택하면 실행됩니다.

Category 추가

1. **[Categories:]**에서 (**+**) 버튼을 눌러 메뉴 카테고리를 생성합니다.
2. **[Input name and text]** 다이얼로그에서 **Name**과 **Text**에 값을 입력 후 **[OK]** 버튼을 눌러 카테고리를 추가합니다.



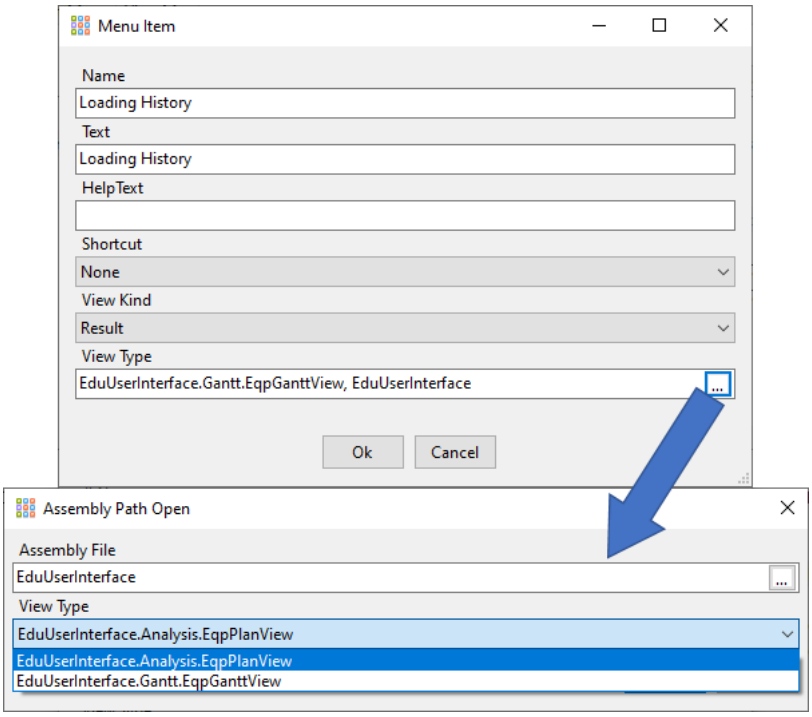
Command Item 추가

1. **[Categories:]**에서 추가한 카테고리 선택, 그 다음 **[Command Items :]**에서 (**+**) 버튼을 눌러 메뉴를 추가합니다. 팝업 메뉴를 추가는 **[Add Popup]**, 일반 메뉴는 **[Add Menu]**를 선택하면 됩니다.
2. Mozart Studio 메뉴에 노출할 최상위 메뉴를 **[Add Popup]**으로 추가합니다.
3. **[Input name and text]** 다이얼로그에서 **Name**과 **Text**에 값을 입력 후 **[OK]** 버튼을 눌러 카테고리를 추가합니다.
4. 다음은, 팝업 메뉴 하위에 추가할 메뉴를 **[Add Menu]**로 추가합니다. 각 항목에 대한 정의는 구성 요소의 `<category>` 항목을 확인 바랍니다.

[Add Menu]로 메뉴를 추가할 때는 [View Kind]에 따라 메뉴 설정 방법이 다릅니다. **[View Kind]**에 따라 설정하는 방법은 다음과 같습니다.

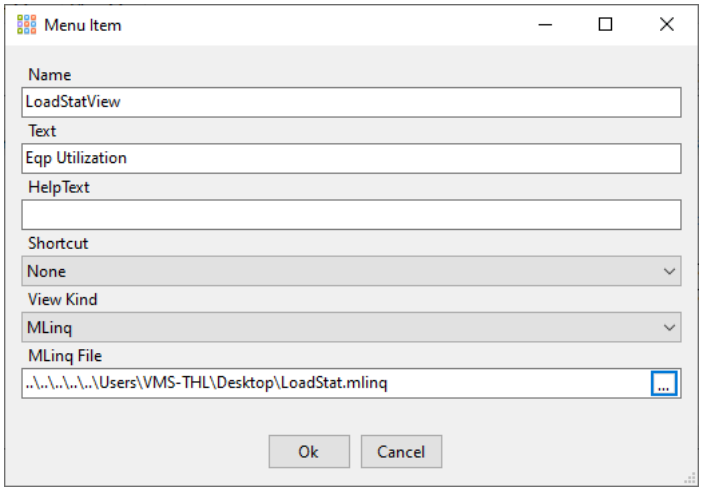
사용자 UI 추가하는 방법

1. Mozart 라이브러리를 이용해서 개발한 UI를 메뉴로 추가할 때는 `View Kind = Result`로 설정합니다.
2. **[View Type]**에는 개발한 UI 프로젝트의 DLL를 선택합니다. **[...]**을 누르면 **[Assembly Path Open]** 다이얼로그가 열리며 이때 **[Assembly File]**에는 UI DLL 파일을 선택하고 **[View Type]**에는 실행 할 UI를 선택합니다.



Mlinq UI 추가하는 방법

1. Mlinq UI를 메뉴로 추가하려면 `ViewKind=Mlinq`로 설정합니다.
2. **[Mlinq File]**에서는 실행할 Mlinq 파일을 선택합니다.



Linq View를 추가하는 방법

> 💡 Linq View는 ViewType을 설정하지 못하는 버그가 있어 본 문서에서는 이를 우회하는 방법으로 설명합니다.

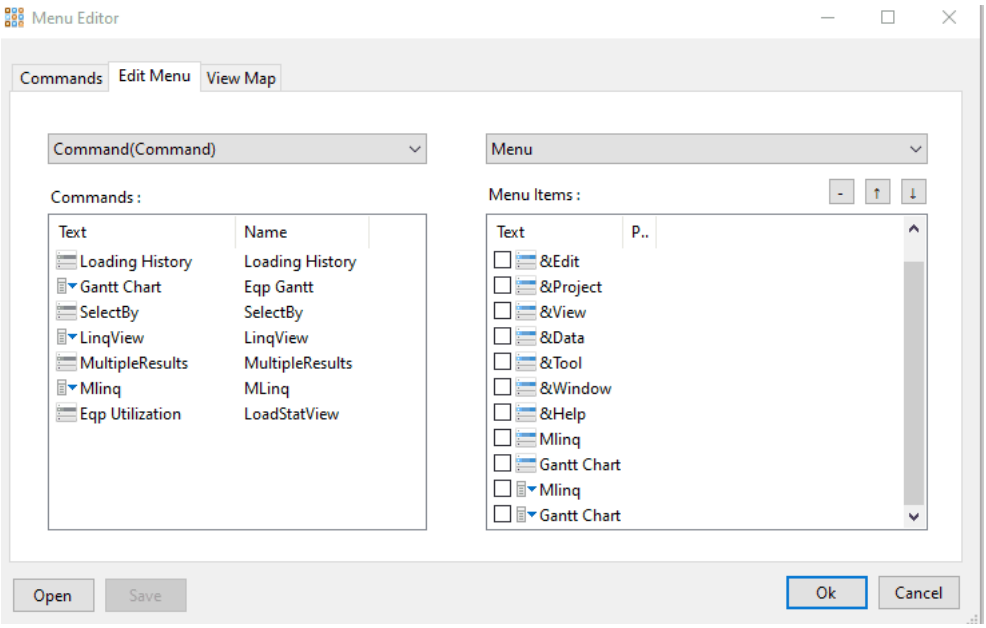
1. **Name**에는 **LinqInfo** 테이블에 등록한 `NAME` 컬럼의 값을 입력합니다. Linq View 사용 방법은 여기를 통해 확인 바랍니다.
2. 그 다음 `ViewKind=Result`로 설정합니다.

3. **[View Type]**에서 **[...]**을 눌러 **[Assembly Path Open]** 다이얼로그에서 다음과 같이 설정합니다.
   - **Assembly File**: **Mozart.Studio.TaskModel.DLL** (파일 위치: *C:\Program Files (x86)\VMS\Mozart\Client\Bin*)
   - **View Type**: LINQ 결과를 Grid로 조회하려면 `Mozart.Studio.TaskModel.UserInterface.DataItemGridView`를 선택, Pivot Grid로 조회하려면 `Mozart.Studio.TaskModel.UserInterface.DataItemPivotGridView`를 선택합니다.
4. **[View Type]** 설정이 완료 되었으면 `View Kind=LinqView`로 설정합니다.

메뉴 등록 (Edit Menu)

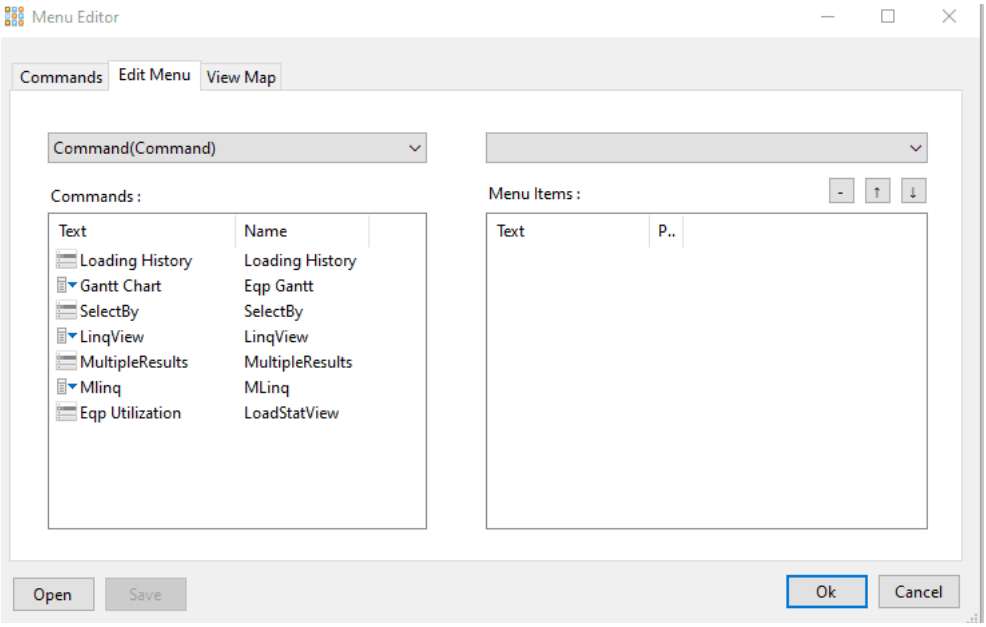
메뉴를 정의하였으면 그 다음에는 메뉴를 구성하고, 등록 그리고 위치를 선정하는 작업을 진행합니다.

1. **[Edit Menu]** 탭으로 이동하여 좌측 드롭다운 리스트에서 등록한 카테고리를 선택합니다.
2. 처음에는 팝업메뉴를 등록합니다. 우측 드롭다운 리스트에서 **[Menu]**, **[ToolBar]**, **[Context Menu]** 중 하나를 선택합니다.
3. **[Commands:]**에 있는 팝업메뉴를 선택하여 우측 **[Menu Items:]**로 Drag & Drop 합니다. 메뉴의 순서는 ↑ 또는 ↓로 조절합니다. 그리고 메뉴 사이 경계선을 추가하고자 하는 경우 기준이 되는 메뉴 체크 박스에 체크합니다.



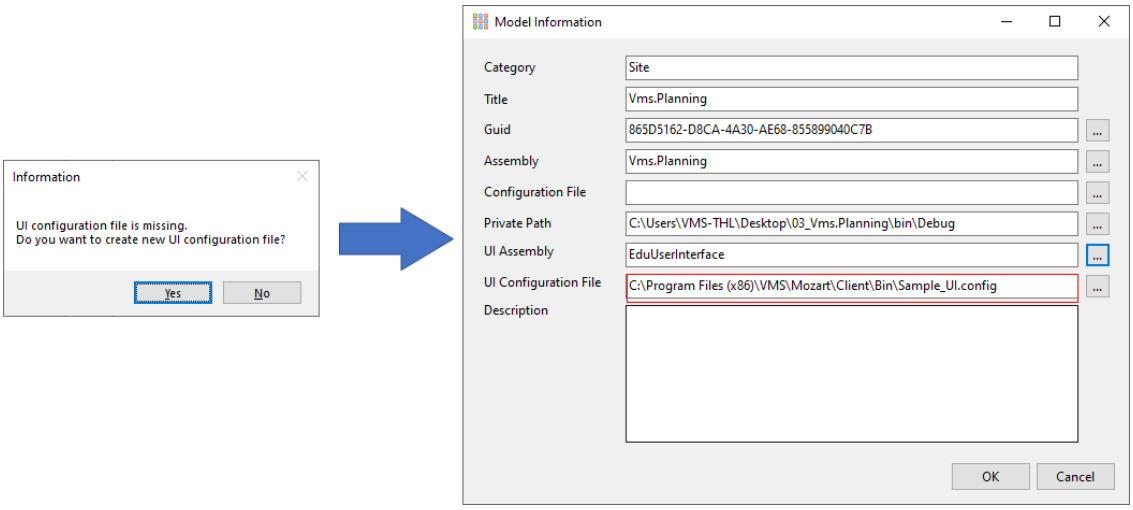
4. 팝업 메뉴를 추가하였으면 우측 드롭다운 리스트에서 일반 메뉴를 추가할 팝업 메뉴를 선택합니다.

5. 3번 과정과 같이 팝업 메뉴에 추가할 메뉴를 Drag & Drop 합니다. 메뉴의 순서는 ↑ 또는 ↓로 조절합니다. 그리고 메뉴 사이 경계선을 추가하고자 하는 경우 기준이 되는 메뉴 체크 박스에 체크합니다.



설정 저장하기(Save)

모든 설정이 완료되었으면 하단의 **[Save]** 버튼을 눌러 저장합니다. Config 파일이 처음 생성되는 경우 아래와 같이 Config 파일을 생성 여부에 대한 팝업창이 출력됩니다. 이때 **[Yes]**를 눌러 파일을 저장하면 Config 파일이 생성되고 모델에 자동으로 UI Config 파일 위치 정보가 등록됩니다.



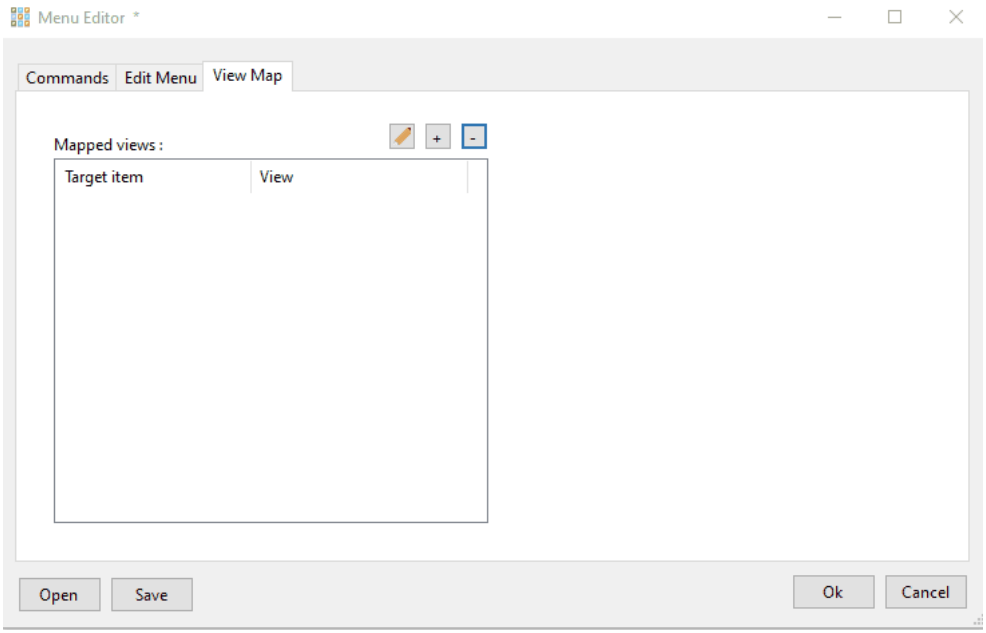
모델의 기본 View 변경 (View Map)

> 💡 View Map은 사용자가 개발한 UI (View Kind ="Result")만 설정 가능합니다.

메뉴 등록을 할 때 **[Commands]**와 **[Edit Menu]**는 반드시 설정해야 하지만, **[View Map]**은 아닙니다. 앞서 설명한 것과 같이 모델의 특정 데이터 테이블을 조회했을 때 View를 기본 Grid에서 다른 View로 변경하고자 할 때 설정합니다. 기본적인 설정 방법은 **[Add Menu]**와 유사합니다.

1. **[View Map]** 탭에서 (+)를 **[Input/Output mapped view]** 다이얼로그를 실행합니다.
2. **[Target Item]**에서 View를 변경할 Input 또는 Output 테이블을 선택합니다.

3. **[View Type]**에서 변경할 View를 선택합니다. 설정 방법은 Command Item 추가에서 하는 방법과 동일합니다. 모든 설정이 끝났으면 **[Save]** 버튼을 눌러 저장합니다. **[OK]** 버튼을 누르면 저장되지 않고 설정 내용이 지워지니 꼭 주의 바랍니다.



View Map 등록 시 주의 사항

UI를 개발할 때 초기화(Document Load) 구현부에서 보통 아래와 같이 구현을 합니다.

```
private void LoadInit()
{
    if(this.Document != null)
       IExperimentResultItem result = this.Document.GetResultItem();
}
```

메뉴에서 UI를 실행할 때는 `Document` 가 `IMenuItem` 형식으로 로딩되어 `GetResultem` 함수를 호출하면 모델의 Result에 해당되는 `IExperimentResultItem` 를 생성할 수 있습니다. 그러나 **View Map**으로 UI를 실행할 때는 `Document` 가 `ResultDocItem` 형식으로 로딩 되어, `GetResultItem` 함수를 호출하면 위의 `result` 에 `null` 값이 들어와 UI를 실행 할 수 없습니다.

아래와 같이 구현하면 **View Map**과 메뉴 중 어느 곳에서 UI를 실행해도 `result` 를 불러올 수 있습니다.

```
private void LoadInit()
{
    if (this.Document != null)
    {
        if (this.Document.ProjectItem is IResultDocItem)
            this.result = (this.Document.ProjectItem as IResultDocItem).Parent; //View Map
        else
            this.result = this.Document.GetResultItem(); //Menu
      }
}
```

Mozart Studio에서 개발 없이 Grid/Pivot UI 만드는 방법

개요

Mozart Studio의 Grid와 Pivot Grid 기능은 모델에서 이미 생성된 데이터를 바인딩하여 화면에 로딩합니다. 그래서 그리드의 레이아웃을 편집 할수 있지만 테이블간 데이터를 조인하는 등의 2차 가공은 할 수 없습니다. 그리고 만약 그런 기능이 필요할 시에는 전용 UI를 따로 개발해야합니다.

이런 부분을 보완하기 위해 모델의 데이터를 LINQ 스크립트로 가공하고 이를 View로 설정하여 Studio에서 열어보고 메뉴로 등록 할 수 있는 **Linq View** 기능이 2021.118.3 버전에 추가되었습니다.

Linq View 기능은 다음과 같은 작업을 할때 유용합니다.

- 초기 데이터 검증 시 모델에 생성된 각각의 Result를 하나의 View에서 비교하고자 하는 경우
- 모델의 각 테이블 별 데이터를 조합하여 새로운 데이터소스를 만들어서 이를 추가 UI 개발 없이 Studio에서 보고자 하는 경우

즉, 모델 수행 중에는 생성될 필요가 없는 데이터이지만, 모델 수행이 종료되고 나서 그 결과를 분석하기 위해 즉석에서 데이터를 가공하고 바로 보는 목적으로 매우 유용합니다.

본 문서는 Linq View에 대한 기본적인 기능 설명과 사용 방법을 안내합니다.

기능 설명

LINQ View 기능은 Mozart Studio의 모든 사용자 권한에서 사용할 수 있습니다. 기능 사용을 위해서는 Studio의 **[Tools]** 메뉴에 **[Test SimpleLinq]**라는 메뉴를 실행하면 됩니다. **[Test SimpleLinq]** 메뉴는 **LinqInfo** 테이블이 모델에 있어야지만 **[Tools]** 메뉴에서 보입니다.

기능 사용에 앞서 **LinqInfo** 테이블과 **SimpleLinq** 메뉴의 구성에 대해 먼저 살펴보도록 하겠습니다.

LinqInfo 테이블

LINQ View를 사용하기 위해서는 LINQ 명칭, LINQ 구문, 언어 유형, Argument와 같은 정보를 기준정보로 가지고 있어야 합니다.

LinqInfo 테이블 정의

Aa 컬럼명	Data Type	정의
QUERY	string	LINQ 구문
NAME	string	정의한 LINQ 스크립트(QUERY)에 이름입니다.
TYPE	string	LINQPad에서 사용하는 언어 유형입니다. 현재는 C# Expression, C# Statements, C# Program만 사용 가능합니다.
ARGUMENTS	string	LINQ 구문에서 정의한 '$' Argument와 기본값입니다. 형식은 Argument= Default Value입니다. Argument는 필수 입력 사항입니다. 여러개의 Argument를 입력하는 경우 구분자는 쉼표(,)입니다.
Description	string	LINQ에 대한 설명입니다.

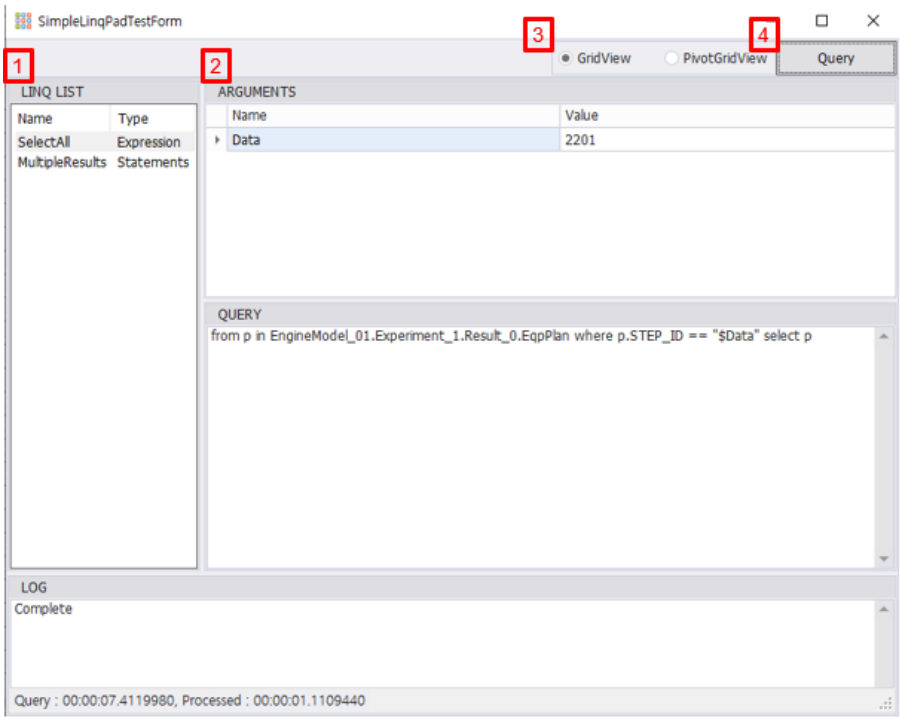
TYPE 값

LinqInfo 테이블의 컬럼 중 `TYPE` 은 사전에 정의된 키워드 값을 입력해야 합니다. `TYPE` 에서 사용할 수 있는 값과 정의는 다음과 같습니다.

- **Expression**: LINQPad의 C# Expression에 해당되며 단순 표현식으로 한 명령에 대해 테스트 하는 용도로 사용됩니다. 여러 줄의 실행 명령을 사용할 수 없습니다.
- **Statements**: LINQPad의 C# Statements에 해당되며 여려 표현식의 묶음인 Statement(s)를 실행 할 수 있습니다.
- **Program**: LINQPad의 C# Program에 해당되며 C# 프로그램처럼 Main과 Class 등 모든 내용을 구현할 수 있습니다.

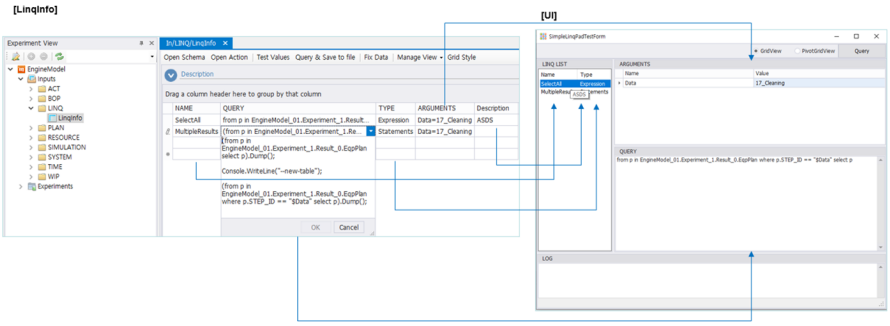
SimpleLinq

SimpleLinq는 **LinqInfo** 테이블의 정의한 LINQ를 실행하기 위한 도구입니다. 아래 그림은 **SimpleLinq**를 실행했을 때 모습이며 메뉴 설명은 다음과 같습니다.



1. **LINQ LIST**: LinqInfo 테이블에 정의한 LINQ 목록입니다. 여기서 실행할 LINQ를 선택합니다.
2. **ARGUMENTS**: LINQ 구문에 정의한 "$" Argument와 값을 설정하는 영역입니다. LinqInfo 테이블의 ARGUMENTS 컬럼의 내용을 표시합니다.
3. **GridView/PivotGridView**: LINQ 결과를 보고싶은 Control에 따라 GridView 또는 PivotGridView를 선택하는 영역입니다.
4. **Query**: LINQ를 실행하는 버튼입니다.

아래 그림은 **LinqInfo** 테이블과 **SimpleInfo** 메뉴간의 관계입니다. 그림을 눌러 확대해서 보시기 바랍니다.



사용 방법 및 예제

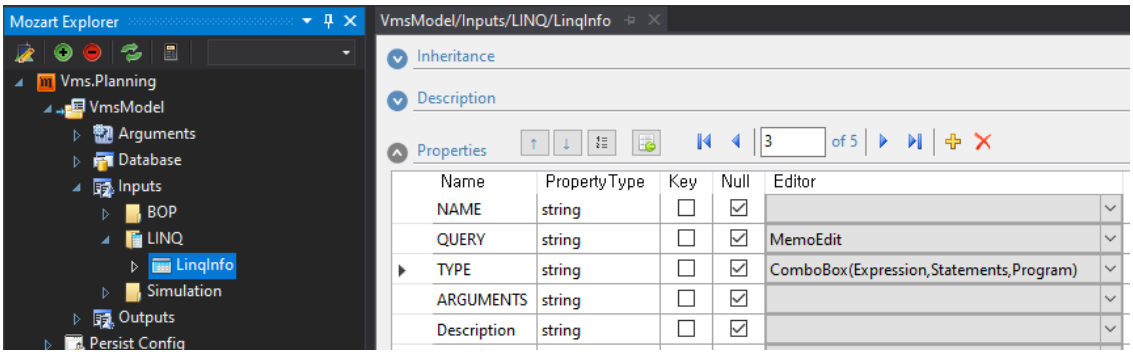
LINQ View를 사용하기 위해서는 기본적으로 LINQPad를 어느정도 사용할 줄 알아야 합니다. 만약 LINQPad를 아직 사용해보지 못했다면 아래 링크에 교육 영상 및 자료가 있으니 꼭 보고 오시기 바랍니다.

예제에서 사용한 LinqInfo 스키마 정보는 아래 링크를 통해 다운 받을 수 있습니다. Mozart IDE에서 DataItem 스키마 정보를 Import하는 방법은 DataItem 정보 Import/Export 하는 방법를 참고 바랍니다.

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/1ca3e27f-71fa-46c4-bfbf-4a73ea7db9c2/LinqInfo.vditem

스텝 1. LinqInfo 테이블 생성

가장 먼저 LINQ View를 사용할 모델의 **[Input]**에 테이블을 추가합니다. 테이블 이름은 반드시 **LinqInfo**라고 명명해야 하며 LinqInfo 테이블 정의에 명시된 컬럼들을 추가합니다. 그 외에 컬럼들은 추가해도 **SimpleLinq**에서 인식을 못합니다.

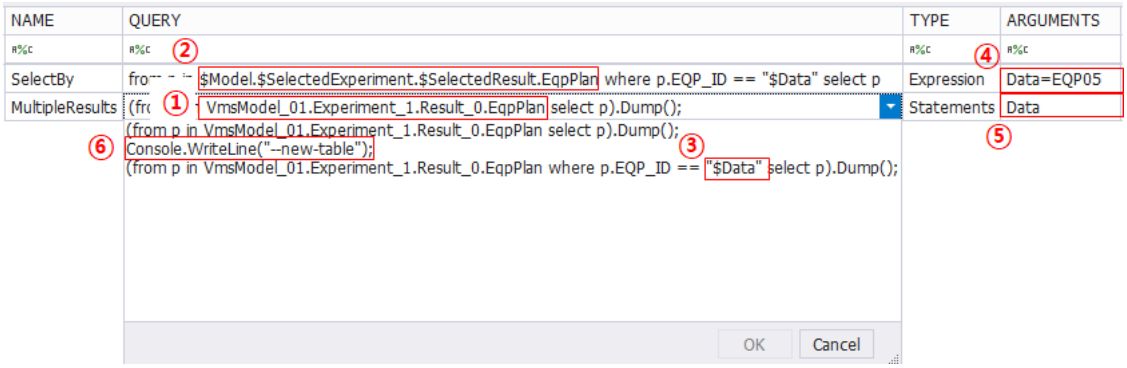


`QUERY` 컬럼에는 LINQ 구문과 같은 긴 문자열이 들어가기 때문에 **[Editor]** 속성을 **MemoEdit**으로 설정하는 것이 좋습니다. **MemoEdit**으로 설정된 컬럼은 Mozart Studio에서 컬럼을 편집할 때 텍스트 에디터처럼 사용 할 수 있습니다.

그리고 `TYPE` 와 같이 값이 정해진 컬럼은 **[Editor]** 속성을 **ComboBox**로 설정하고 **ComboBox** 안에 값을 설정하면 Mozart Studio에서 컬럼을 편집할 때 사전에 등록된 값을 선택할 수 있습니다.

스텝 2. LINQ 구문 작성

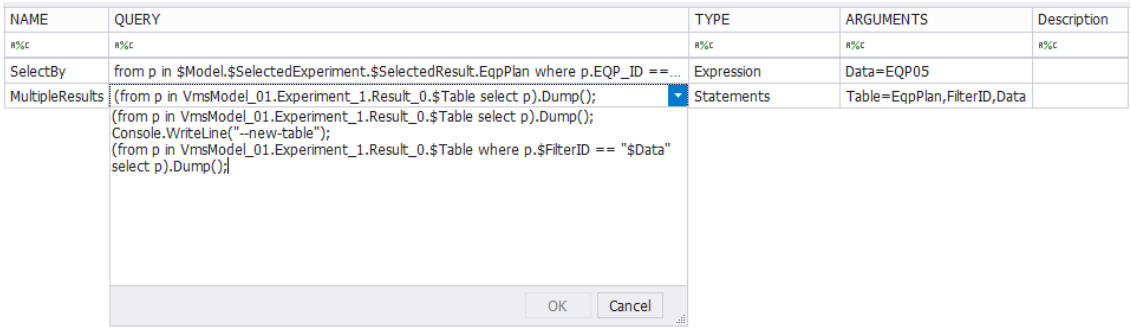
LinqInfo 스키마를 정의했으면 다음에는 Mozart Studio를 실행하여 **LinqInfo** 테이블에 값을 입력합니다. 아래는 예제입니다.



1. 모델에서 데이터를 읽기 위해서는 기본적으로 "*모델명_모델인덱스.Experiment_접미사.Result_접미사.테이블명*" 형식으로 LINQ 구문을 선언합니다.
2. 따로 지정하지 않고 **[Experiment View]**에서 선택한 **[모델] > [Experiment] > [Result]**의 데이터를 로딩하려면 사전 정의된 Argument(예: *$Model.$SelectedExperiment.$SelectedResult.테이블명*)를 사용하면 됩니다.
   - **$Model**: 현재 선택된 모델
   - **$SelectedExperiment**: 현재 선택된 Experiment
   - **$SelectedResult**: 현재 선택된 Result

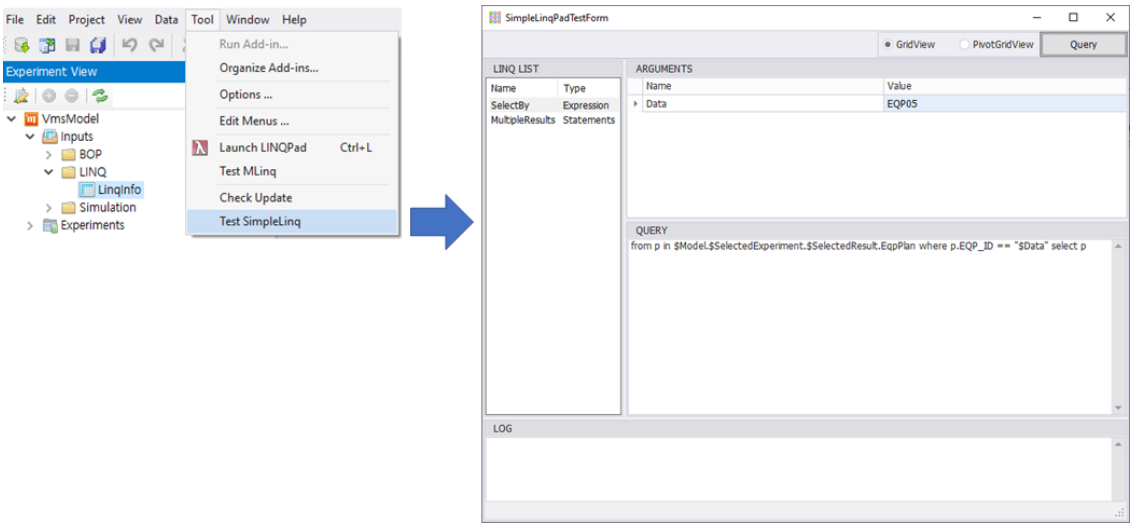
3. LINQ 구문에서 Argument를 선언할 때는 Argument 앞에 $를 붙입니다.
4. LINQ 구문에 선언한 Argument는 **LinqInfo** 테이블 `ARGUMENTS` 컬럼에 `Argument = Default Value` 형식으로 입력합니다.
5. 기본값을 주고 싶지 않을 때는 Argument만 입력합니다.
6. 출력 결과(`Dump`)마다 새 탭으로 열어서 보려면 LINQ 구문에 `Dump` 함수 다음에 `Console.WriteLine("--new-table")` 를 추가하면 됩니다.

LINQ 구문에 Argument를 여러개 선언한 경우에는 쉼표(,)로 구분하여 `ARGUMENTS` 컬럼에 등록 할 수 있습니다.



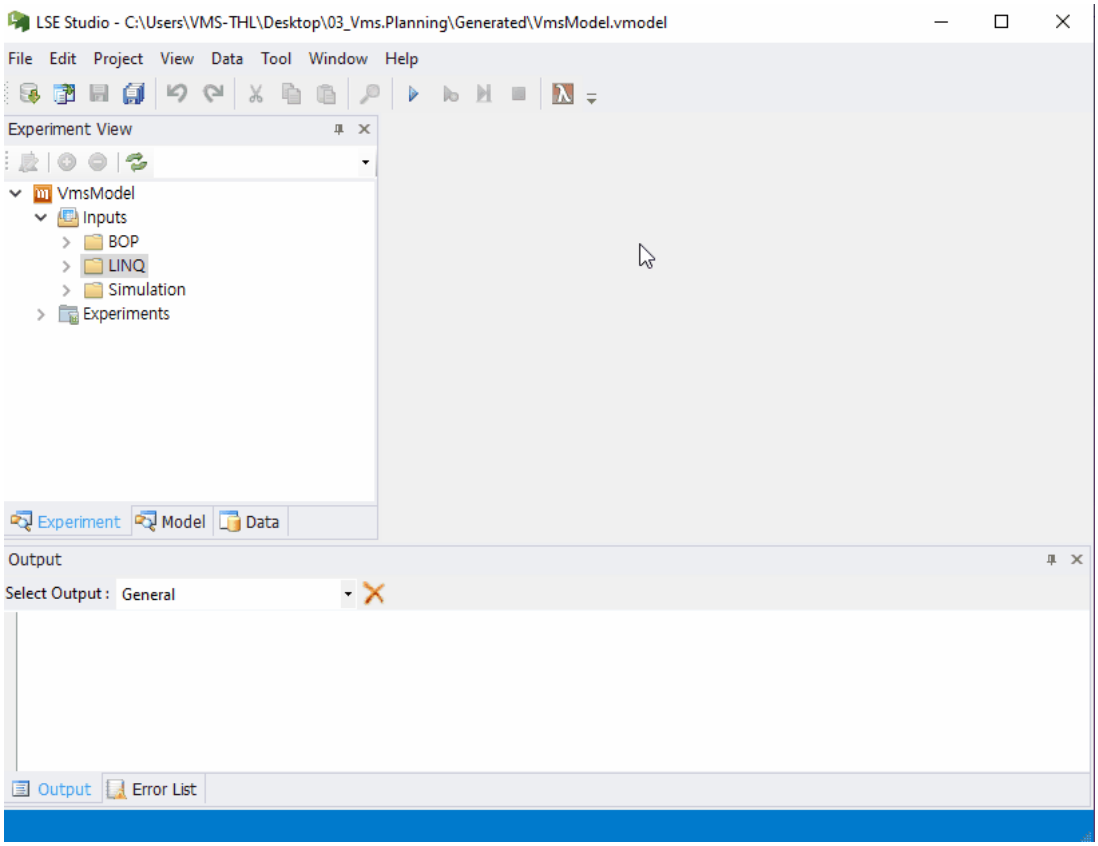
스텝 3. SimpleLinq 실행

LinqInfo 테이블에 필요한 정보를 모두 입력했다면 **SimpleLinq** 도구를 열어 스크립트를 실행합니다. **SimpleLinq**는 Mozart Studio 툴바의 **[Tool] > [Test SimpleLinq]**로 실행합니다.



처음에 **[LINQ LIST]**에서 실행 할 LINQ를 선택합니다. 그 다음 **[ARGUMENTS]** 영역에서 Argument 값을 지정하고 필요에 따라 최종적으로 **[QUERY]** 영역에서 LINQ문을 수정할 수 있습니다. 이때 수정된 내용은 **LinqInfo** 테이블에 저장되지 않습니다. 모든 설정이 끝났으면 GridView, PivotGridView 중 데이터를 표시할 컨트롤을 선택하여 **[Query]** 버튼으로 선택한 LINQ를 실행합니다. 이때 **[Open-Item(Read-Only)]**이 실행이 되어 결과를 출력합니다.

[예제]



스텝 4. Studio 메뉴에 등록하기

Mozart Studio에 메뉴 등록하는 방법은 메뉴 등록을 참고 바랍니다.

LINQ View를 Studio 메뉴에 등록하면 매번 **[Tools] > [Test SimpleLinq]** 실행 없이 바로 결과를 볼 수 있습니다. 다만 이때는 반드시 **LinqInfo** 테이블의 `ARGUMENTS` 컬럼의 Argument에 값이 있어야 합니다.

Studio 메뉴에 LINQ View 기능을 등록하기 위해서는 모델이 사용하고 있는 UI config 파일을 수정해야 합니다. 만약 UI config 파일이 없으면 새로 만들어야 합니다.

① LinqInfo Table NAME Column Value

② Pivot Grid

```
    <add name="SelectBy" text="SelectBy" popup="false"
shortcut="None" viewType="Mozart.Studio.TaskModel.UserInterface.DataItemPivotGridView, Mozart.Studio.TaskModel"
viewKind="LinqView"/>
    <add name="MultipleResults" text="MultipleResults" popup="false"
shortcut="None" viewType="Mozart.Studio.TaskModel.UserInterface.DataItemGridView, Mozart.Studio.TaskModel"
viewKind="LinqView"/>
```

② Grid

③

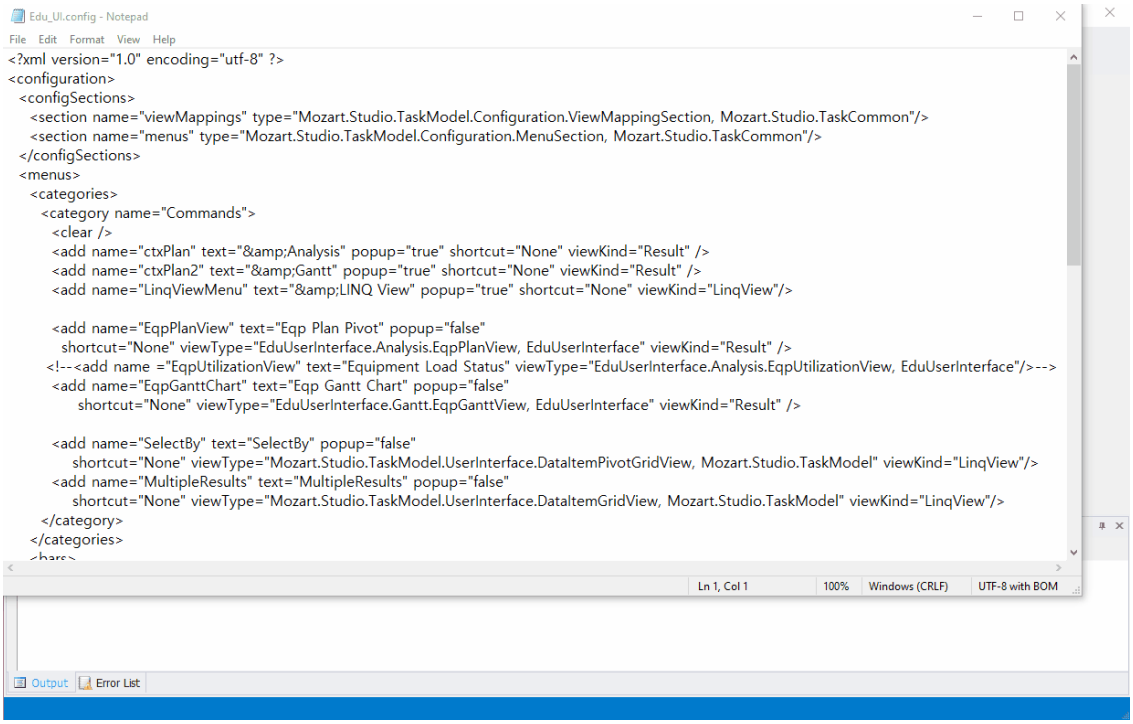
UI config 파일에서 추가하고자 하는 LINQ View를 `<category>` 요소에 내용을 추가합니다. 이때 주요 속성에 대한 값은 아래와 같습니다.

1. **name**: **LinqInfo** 테이블의 `NAME` 컬럼의 값을 입력합니다.
2. **viewType**: 여기에는 LINQ 결과를 Grid 또는 Pivot Grid로 볼 것인지 여부를 입력합니다. 사용하고자 하는 유형에 따라 `viewType` 을 아래의 값으로 설정합니다.
   - ***Grid***: Mozart.Studio.TaskModel.UserInterface.DataItemGridView, Mozart.Studio.TaskModel

- ***Pivot Grid***: Mozart.Studio.TaskModel.UserInterface.DataItemPivodGridView, Mozart.Studio.TaskModel

3. **viewKind**: `"LinqView"`라고 입력합니다.

[예제]



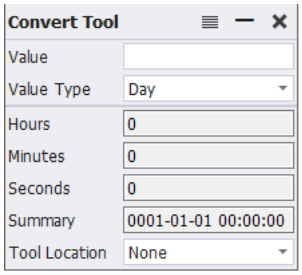
Convert Tool 사용법

사용 가능 버전 : **MOZART Client 2021.121.0 이상**

Convert Tool 은 자주 사용되는 시간 단위 변환값을 참조할 수 있는 보조 도구입니다.

Convert Tool

Studio의 상단 메뉴 **Tool > Convert Tool**을 실행하면 Convert Tool 팝업창이 나타납니다. 기본 단위는 Day로 되어 있으며, Value에 값을 넣으면 아래에 시(Hours), 분(Minutes), 초(Seconds), TimsSpan(Summary) 형태로 변환하여 보여줍니다.



Convert Tool 속성

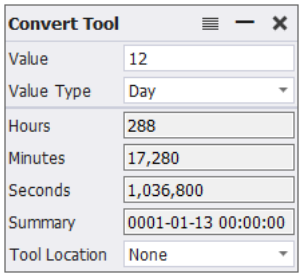
Convert Tool 의 값은 팝업창을 닫아도 자동 저장됩니다.

- **Value** : 체크 시 Copy Tool 팝업창을 항상 위로 고정합니다.
- **Value Type** : Copy Tool 포맷 설정을 기본 설정으로 변경합니다.
- **Hours, Minutes, Seconds, Summary** : 각 값은 읽기 전용으로 Value와 Value Type 에 따라 자동 표시됩니다.
- **Tool Location** : Copy Tool 팝업창의 위치를 선택합니다.(ex. `Top`, `Center`, `Bottom`, ...) 위치를 지정하지 않고 드래그를 통해 팝업창 위치를 정할 수 있습니다.

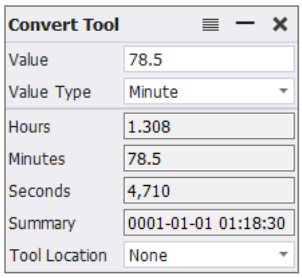
Convert Tool 사용 예시

12 일(Day)을 입력했을 때(좌측)와 78.5 분(Minute)을 입력했을 때(우측) 화면입니다.

각 단위 별 값을 복사하여 활용할 수 있습니다.



12 일(Day)를 입력한 화면



78.5 분(Minute)을 입력한 화면

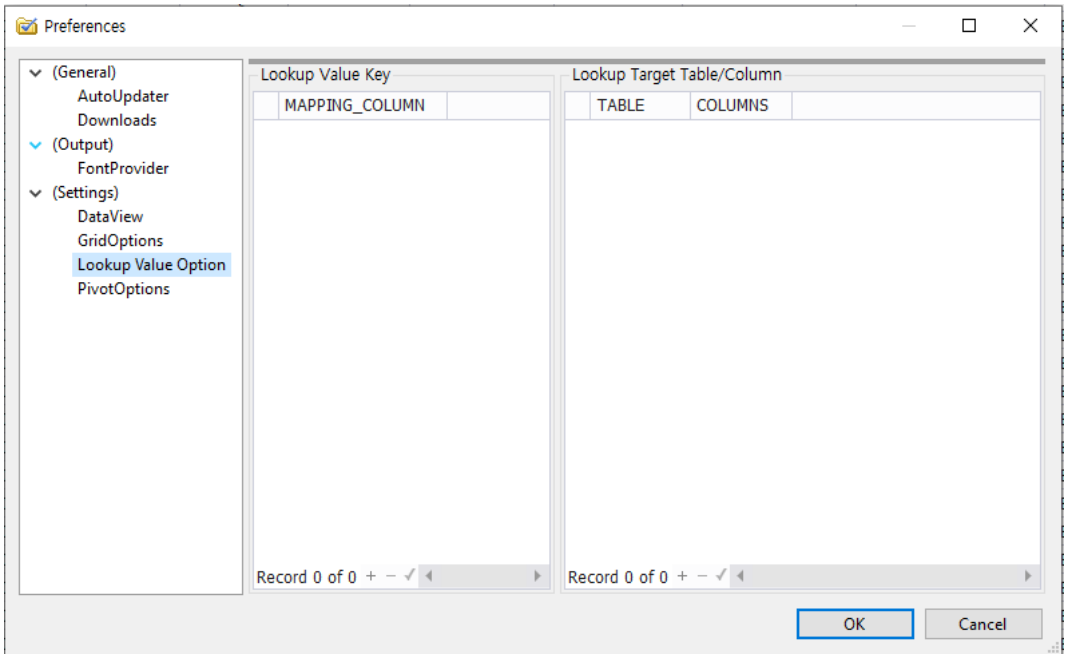
데이터 Look Up 기능

사용 가능 버전 : **MOZART 2022.122.0 이상**

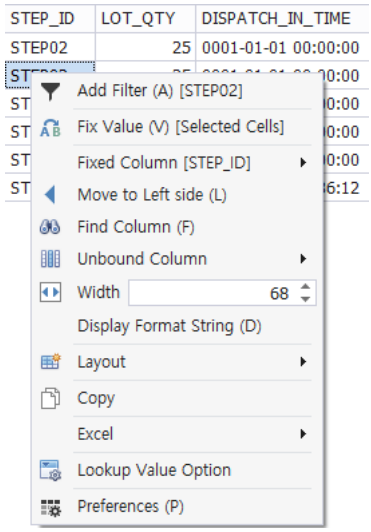
모델에서 결과 데이터에서 영향을 주는 주요 기준정보가 존재할 경우, 이 정보를 일괄 조회 할 수 있습니다. 데이터의 연관성을 설정 한 후, 결과 데이터 분석 시 여러 기준 정보 테이블을 이동하면서 찾을 필요 없이 하나의 화면에서 일괄 확인이 가능합니다.

Look Up 설정

Look Up 기능을 사용하기 위해서는 우선, Look Up 키를 사용 할 컬럼과 Look Up 대상 테이블과 컬럼을 설정해 주어야 합니다. 설정 화면은 Studio의 상단 메뉴 **[Tool > Option]** 을 실행하여 Preferences 창에서 좌측의 **[(Settings) > Lookup Value Option]** 메뉴 화면에서 설정 할 수 있습니다.



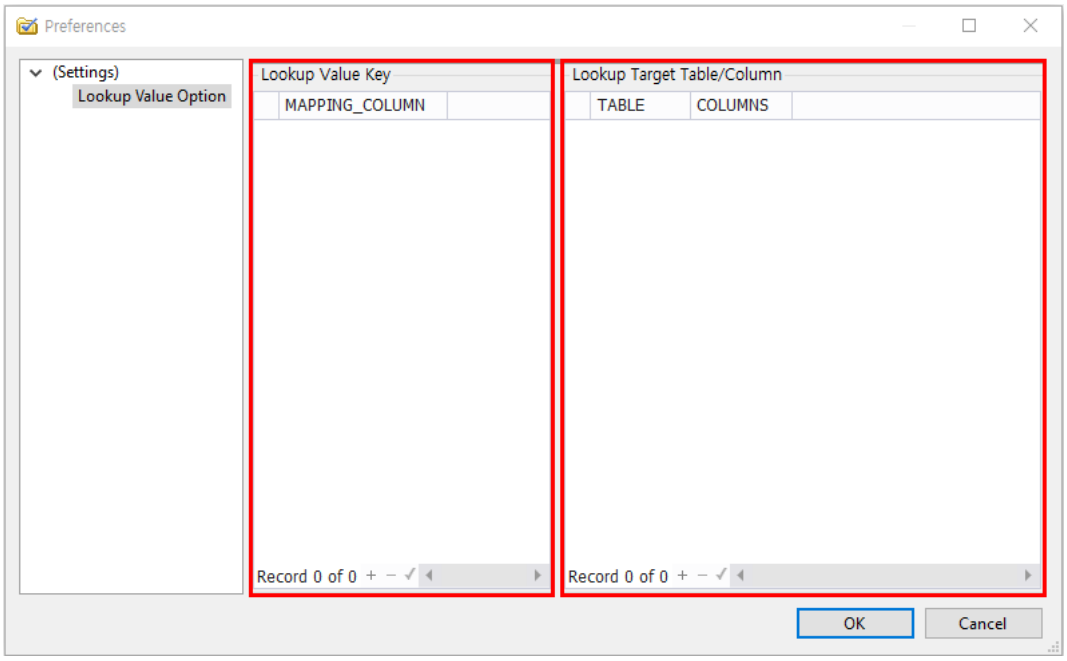
또한 Studio 상단 메뉴가 아닌, 데이터 그리드 뷰에서 컨텍스트 메뉴 중 **[Lookup Value Option]** 을 이용하여 설정화면으로 이동 할 수 있습니다.



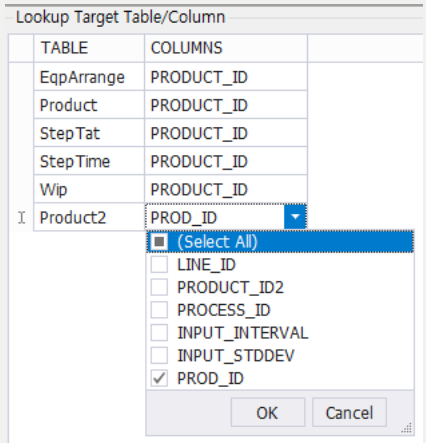
Look Up Value Option 설정

크게 2개의 영역으로 나뉘어지며, 각각의 역할은 다음과 같습니다.

- **Lookup Value Key** : Lookup 에 사용할 컬럼을 입력하는 영역입니다. 아래 +(추가), -(삭제) 버튼을 이용하여 `MAPPING COLUMN` 을 추가 할 수 있습니다. `MAPPING COLUMN` 은 1 개의 Key 기준으로 여러 개의 Lookup Target Table/Column 을 가지도록 되어 있습니다.
- **Lookup Target Table/Column** : **Lookup Value Key** 에 등록된 모델 내 데이터 테이블과 컬럼을 지정하는 영역입니다. Lookup Value Key 에서 입력 된 컬럼이름을 모델의 Input 데이터에서 찾아서 자동으로 보여줍니다. `MAPPING_COLUMN` 의 값을 조회 할 때 Input 데이터 아이템을 기준으로 조회가 됩니다.

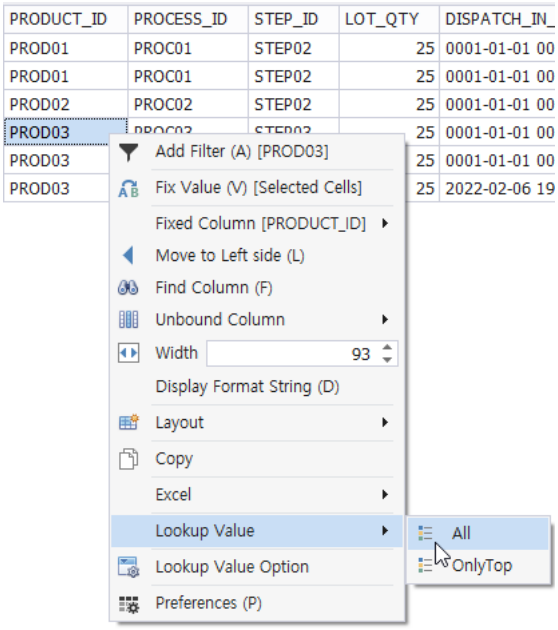


아래 예제와 같이 Look Up Key 가 PRODUCT_ID 인 경우 PROD_ID 처럼 다른 컬럼 이름으로 되어 있는 경우에는 자동 추가가 되지 않습니다. 이러한 경우에는 직접 아래 추가(+) 버튼을 눌러 Target Table/Column에 추가하여 Look Up 기능 실행시 포함되어 나오도록 할 수 있습니다.

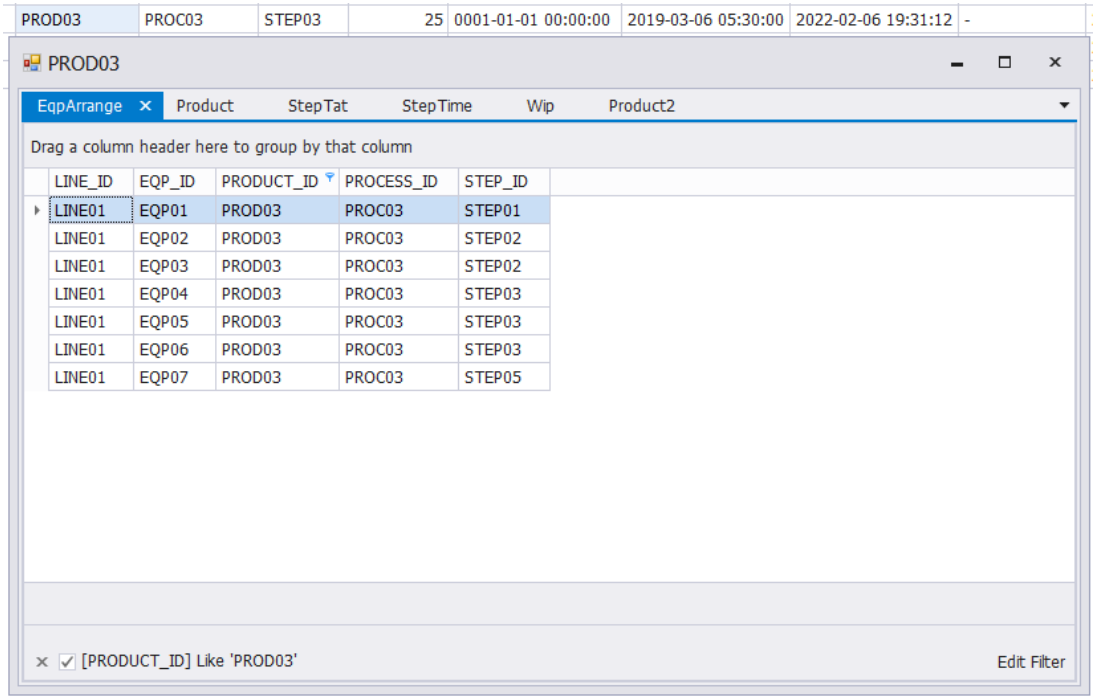


Look Up Value 실행

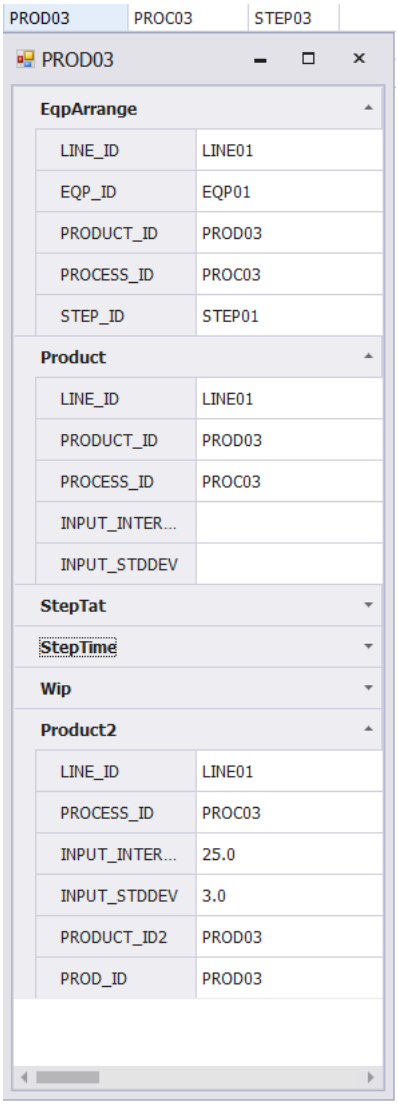
Look Up Value Option 설정이 되어 있다면, 해당 컬럼에 컨텍스트 메뉴를 누르면 Lookup Value 메뉴가 활성화되어 선택된 값을 기준으로 설정된 테이블의 Column 값을 보여줍니다. 데이터 셀에서 우측 마우스를 클릭하여 컨텍스트 메뉴인 Lookup Value > All / OnlyTop 을 선택하여 실행 할 수 있습니다.



- **All** : 등록된 Table/Column 의 모든 테이블에서 선택된 값의 정보를 멀티 탭 형태로 모두 보여주는 메뉴입니다.



- **OnlyTop** : 등록된 Table/Column 의 모든 테이블에서 상위 1개의 레코드 데이터만 보여주는 메뉴입니다.

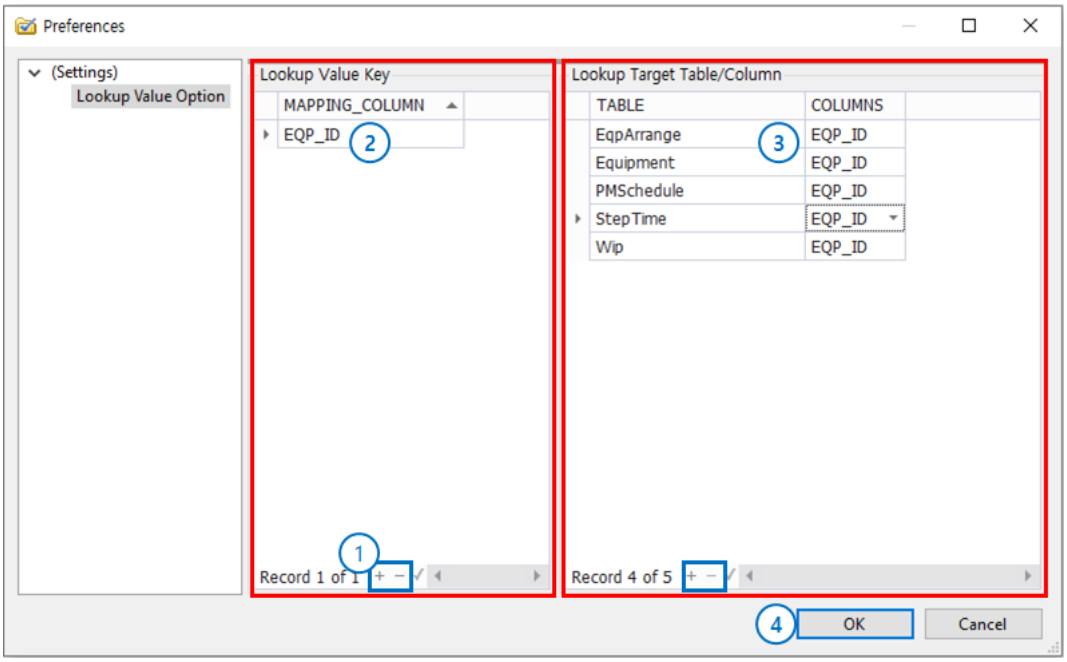


데이터 테이블이 표시되는 순서는 LookUp Target Table/Column 에 등록된 데이터 테이블의 순서대로 표시됩니다.

예제

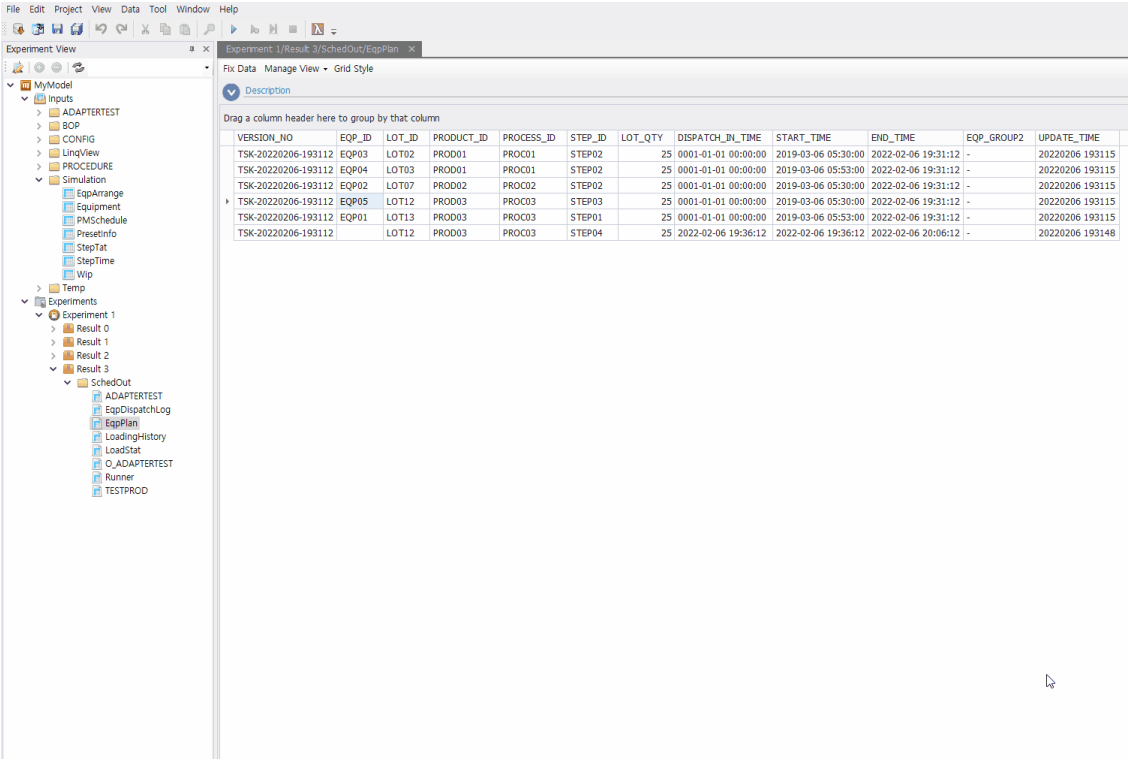
아래 예제는 Lookup Value Option 화면에서 Lookup 컬럼으로 EQP_ID를 등록하는 과정입니다.

1. Lookup Value Option 화면에서 Lookup Value Key 영역의 레코드 **추가(+)버튼**(①)을 눌러서 추가합니다.
2. MAPPING_COLUMN(②)에 Lookup 대상 Column을 입력합니다.
3. MAPPING_COLUMN이 입력이 되면 모델에 등록된 Input 모델의 데이터 아이템에서 컬럼의 이름이 동일한 정보를 찾아서 Lookup Target Table/Column(③) 에 자동으로 표시가 됩니다.
4. Lookup Target Table/Column 영역에서 불필요한 TABLE/COLUMNS 정보가 있다면 레코드 **삭제(-) 버튼**을 눌러서 삭제 할 수 있습니다.
5. **OK 버튼**(④)을 눌러 Lookup Value Option 을 종료합니다.



6. Input/Output 데이터 테이블에서 EQP_ID 컬럼에서 컨텍스트 메뉴를 누르면 **[Lookup Value]** 메뉴가 활성화 되어 조회할 수 있습니다.

사용 영상



Mozart Registry Cleaner Guide

Registry Cleaner 를 사용 해야 하는 이유

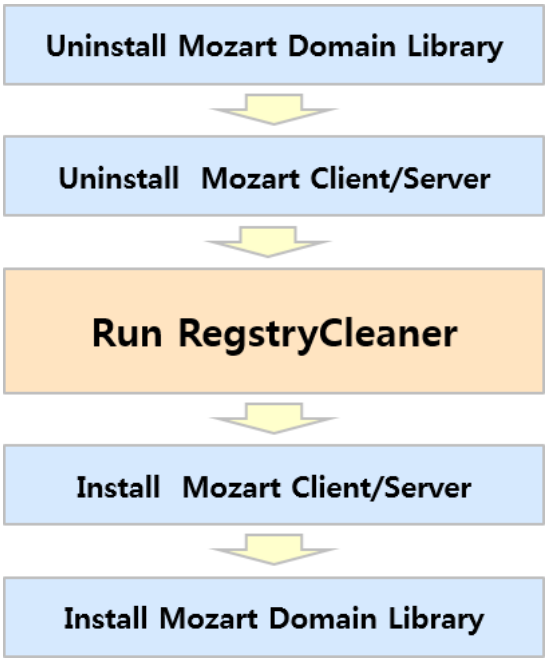
- Mozart 제품의 Release가 이루어지면서 반복적인 설치와 삭제로 인해 특정 파일 들이 삭제 되지 않는 현상이 발견 되었습니다.
- 이러한 현상을 바로 잡고 정상적인 설치 환경으로 만들기 위해 Mozart Registry Cleaner 개발하여 배포하게 되었습니다.

Mozart Registry Cleaner 사용 대상

- Mozart 제품을 `모두 삭제` 후 아래와 같이 Mozart 가 설치 된 곳에 다음과 같은 파일들이 남아 있다면 반드시 사용하여야 합니다.
  - **Mozart Client 사용자**
    - `Mozart 가 설치 되었던 폴더에 Mozart 로 시작 하는 파일이 존재 하는 경우`
    - ex) Mozart.RuleFlow.dll/pdb, ...
- **Mozart Server 사용자**
  - `Mozart 가 설치 되었던 폴더에 Mozart 로 시작 하는 파일이 존재 하는 경우`
  - ex) Mozart.RuleFlow.dll/pdb, ...

Mozart Registry Cleaner 사용 방법

- Registry Cleaner는 Mozart 제품을 `모두 삭제 한 후 실행` 되어야 합니다.

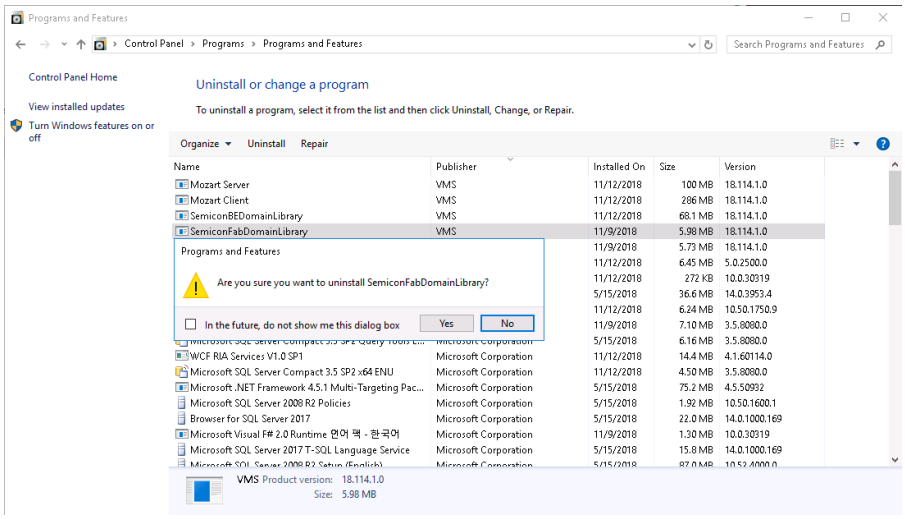


Registry Cleaner 는 실행 파일과 설정 파일 총 2개의 파일로 구성 되어 있습니다.

- Mozart.RegstryCleaner.exe
- Mozart.RegstryCleaner.exe.config

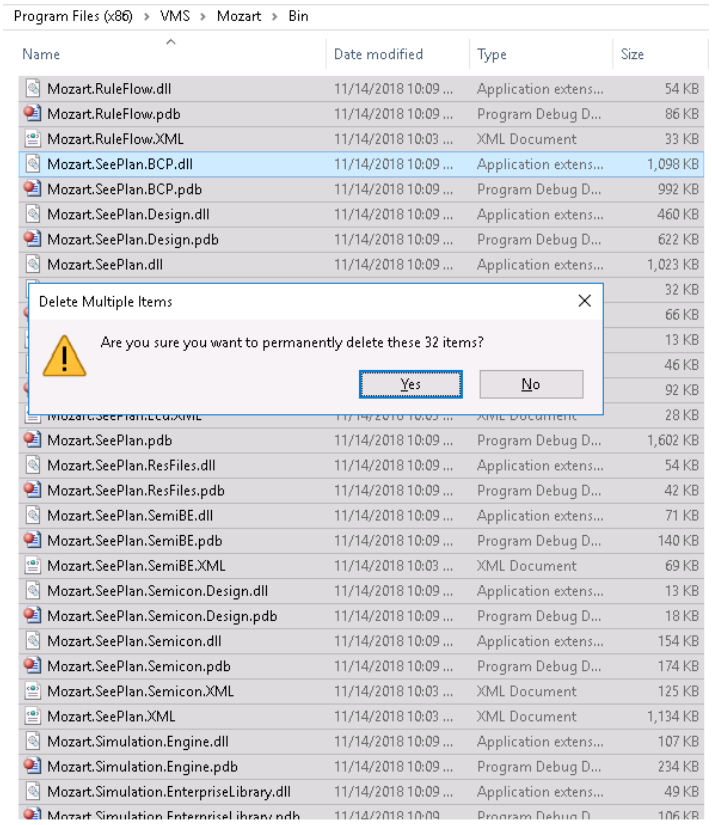
1. 1.Mozart 제품을 모두 삭제 합니다.

   i. 삭제 순서는 Domain Library 삭제 후 Mozart Client/Server 순서입니다.



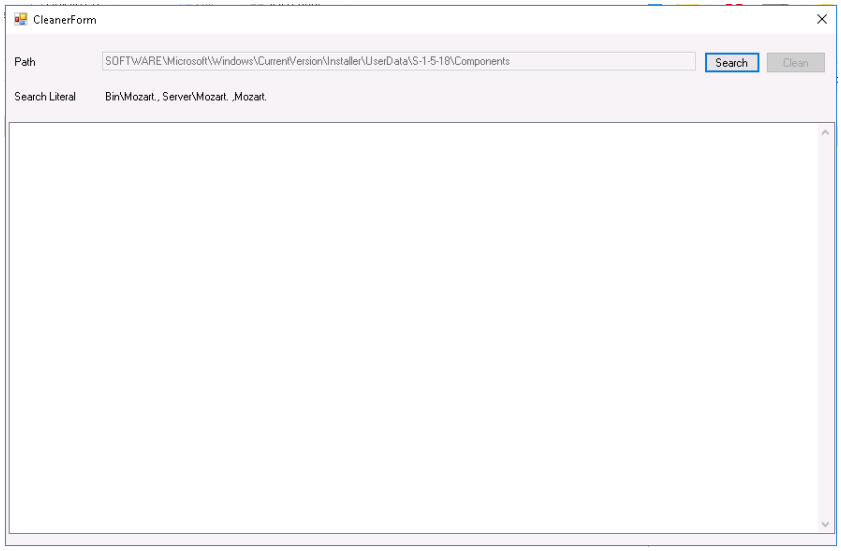
2. `Mozart 를 모두 삭제 후` Mozart가 설치 되었던 폴더 내 Mozart로 시작하는 파일이 존재한다면 `수동 삭제`를 하십시오.

   i. Mozart 로 시작하는 파일은 모두 수동 삭제하여야 합니다.



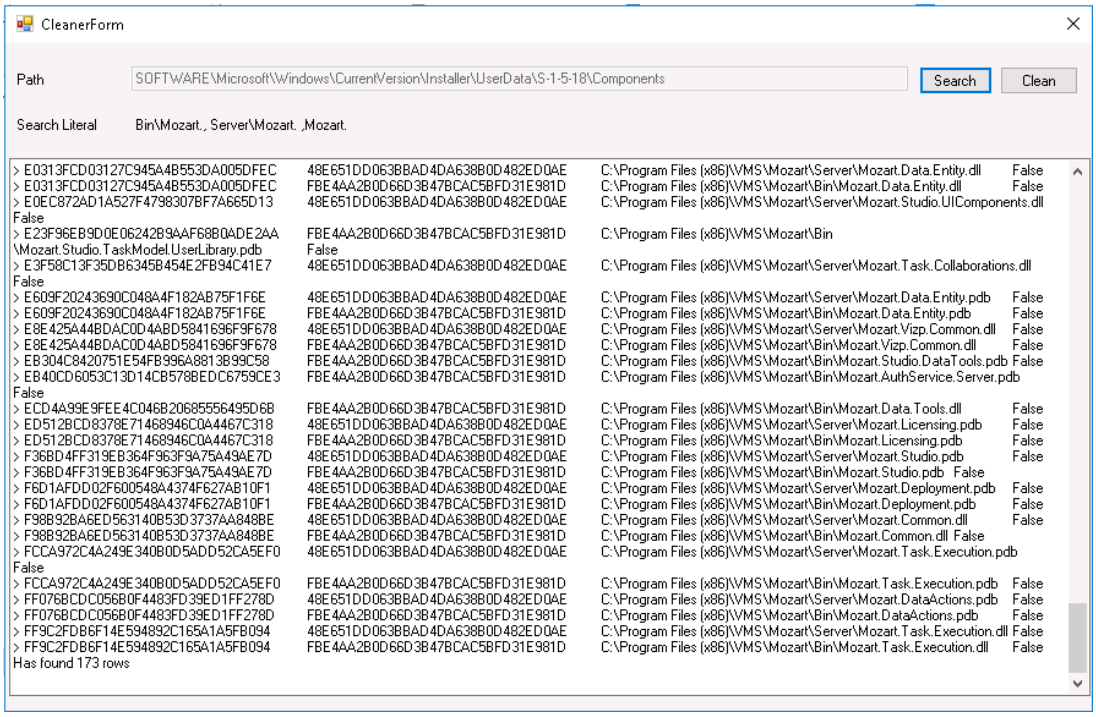
3. Mozart 제품을 모두 삭제 후, Mozart.RegstryCleaner.exe 를 실행합니다.

   i. Path와 Search Literal은 고정되어 있습니다.

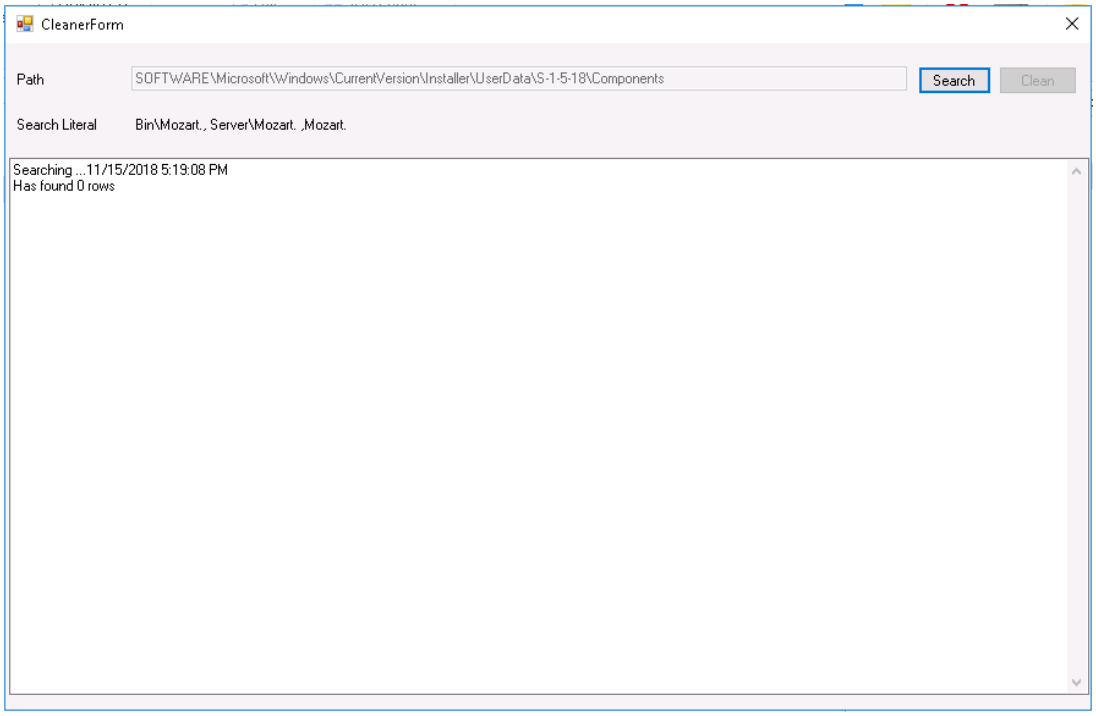


4. Search 를 눌러 Registry 검색을 합니다.

   i. Path에서 Literal 정보를 검색하여 보여줍니다.

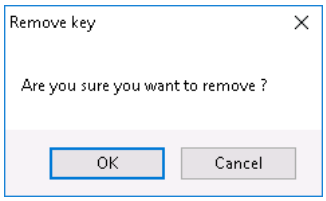


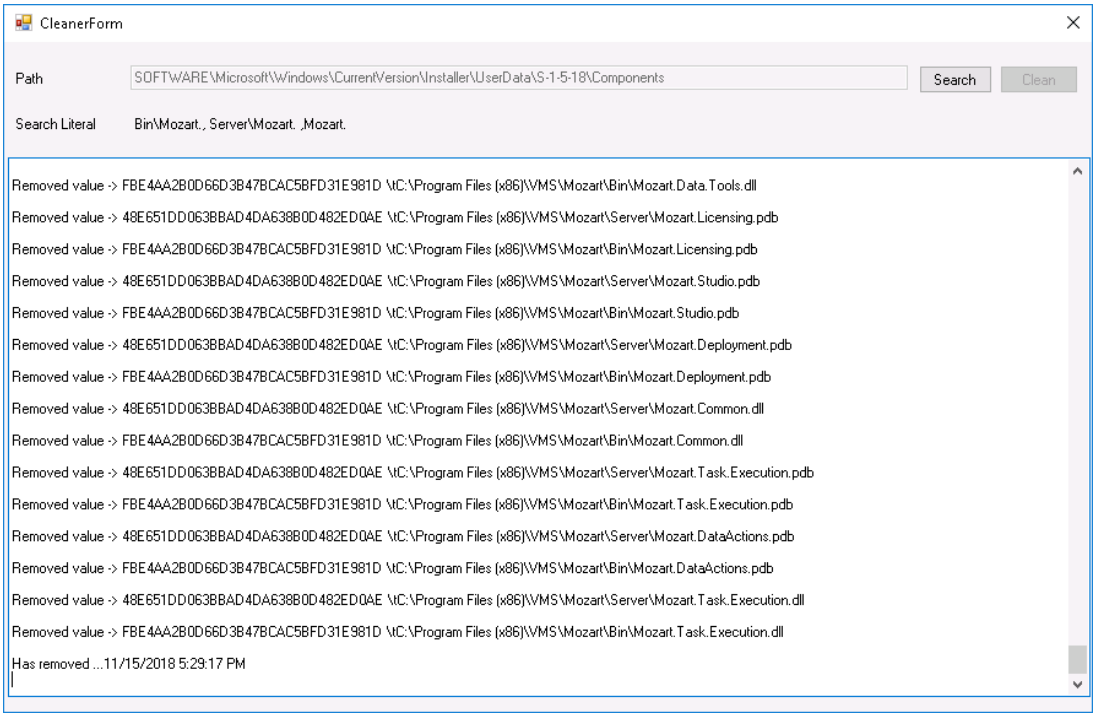
   ii. 아래와 같이 검색되는 것이 없다면, 이상 현상이 없기 때문에 Registry Cleaner를 중단하여 주십시오.



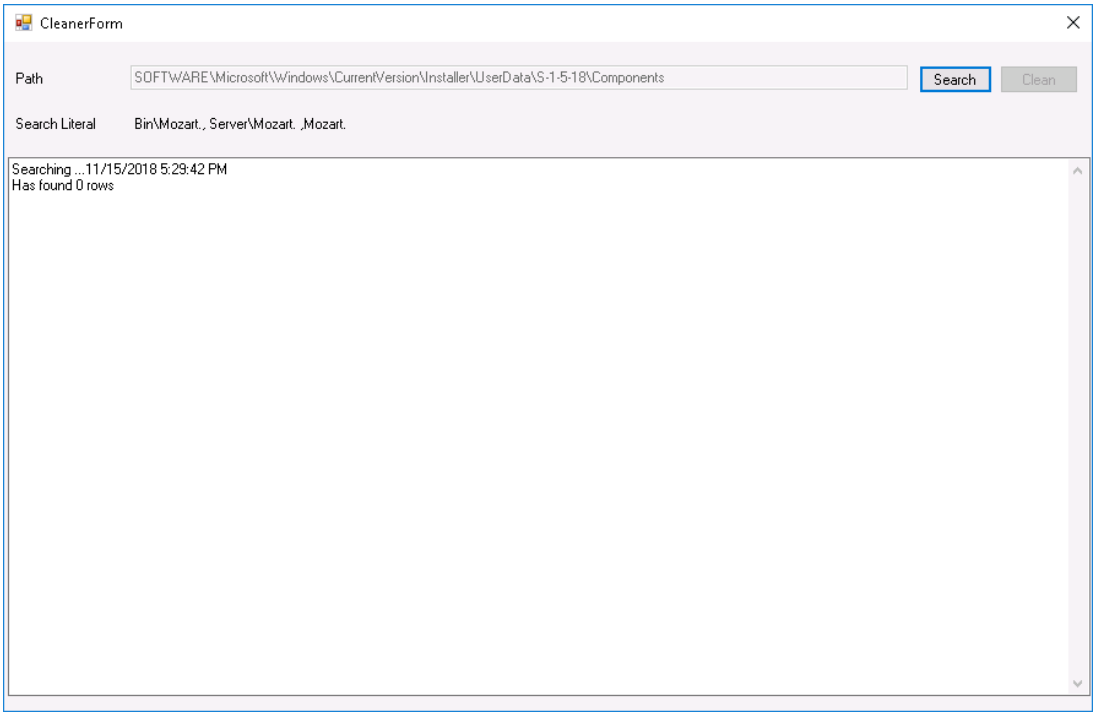
5. Search 결과가 있다면 Clean 을 눌러 Registry 정리를 실시합니다.

   i. 이 삭제를 통해 Mozart 제품과 관련이 있는 Registry가 정리가 됩니다.





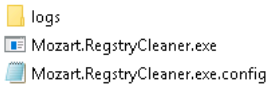
6. 삭제하였다면 다시 Search를 통해 삭제 여부를 확인하면 됩니다.



i. 삭제가 잘 되었다면 검색이 되지 않아야 합니다.

7. Mozart.RegstryCleaner 의 검색 및 삭제의 이력은 실행 파일과 동일한 위치에 Log 폴더가 생성되어 Log 기록이 되며, 확인이 가능합니다.

   i. Search 이력과 삭제 이력 모두 기록됩니다.



8. Mozart와 관련하여 Regstry 정리가 완료되었으며, 이제 Mozart 제품을 설치하여 사용하시면 됩니다.

설치 파일 재배포 시나리오

Mozart 설치파일을 End-User에게 배포하는 방법에 대한 가이드 문서입니다.

재패키지 하여 배포하는 방법

사이트 내 초기 설치 대상자들에게 Mozart 설치파일 배포 전 사이트 특성에 맞게 재패키지하는 방법에 대해 안내 드립니다.

개요

본사에서 배포된 Studio.zip 압축을 해제하면 그 안에 `ToBin`, `ToRoaming`, `Installer` 총 3개의 폴더가 존재합니다. 이 중 `ToBin`, `ToRoaming` 은 사이트 내 초기 설치 대상자들이 설치 후 Mozart에 대한 추가적인 설정을 안 해도 되게, 배포자가 그러한 설정 파일들을 사전에 추가하기 위한 폴더들입니다.

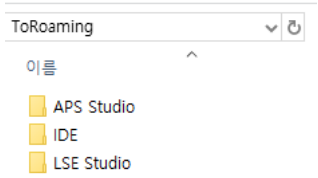
기본 절차는 초기 설치 대상자들이 필요한 각종 파일 및 폴더들을 추가하여 이를 다시 압축한 다음 재배포를 합니다. 이때, 초기 설치 대상자들은 재배포한 압축 파일을 받아 압축 해제 후 그 안에 있는 `MozartClientExecute.exe`를 실행하면 기본 Mozart 설치와 더불어 배포자가 추가한 파일들도 같이 설치됩니다.

주요 파일/폴더

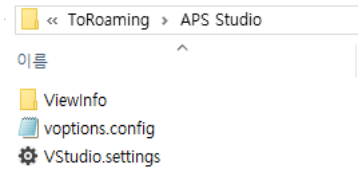
다음은 재배포에 있어 주요 역할을 하는 파일 및 폴더에 대한 설명입니다.

- ***ToBin*** : Mozart Client가 설치되는 폴더(i.e *C:\Program Files (x86)\VMS\Mozart\Client\Bin*) 에 복사할 파일 및 폴더를 추가하는 위치입니다. 다음과 같은 파일들이 주로 ToBin에 들어갑니다.
  - Studio 사용자 권한 DLL 파일 (VStudio.AdmSample.dll/pdb 또는 커스텀 제작 DLL)
  - 사이트 로직,UI DLL 파일
  - UI config 파일
  - Studio config 파일 (XXX_Studio.config: Studio 권한 파일 사용 여부 설정 파일)
  - Targets 폴더 (Targets*.vinfo)

- ***ToRoaming*** : Studio/IDE/MMC 관련 config 및 라이선스 파일이 저장되는 %APPDATA%\Mozart에 복사할 폴더입니다. ToRomaing 폴더에는 복사할 파일에 대해 반드시 폴더 단위로 추가해야하며, 폴더 명은 Roaming 폴더에 있는 실제 폴더명으로 해야합니다. 다음과 같은 정보들이 주로 ToRoaming에 들어갑니다.
  - ***Studio 폴더***
    - AutoUpdate 및 Model Download 정보 파일 : 각 Studio 폴더 안에 있는 `voptions.config` 파일. 해당 파일이 복사되면 사용자는 Model Download 서버나, AutoUpdate 사용 여부 및 서버 정보를 별도로 설정 안해도 됩니다.
    - Studio Layout 파일 : 각 Studio 폴더 안에 있는 ViewInfo\Layout*.xml 파일. 사용자가 특정 DataItem에 대해 보고 싶은 컬럼 순서 혹은 원하지 않은 컬럼등을 숨기는 등의 설정이 저장된 파일입니다.
    - Studio Filtering 정보 : 각 Studio 폴더 안에 있는 ViewInfo\Filter*.vffs 파일. 사용자가 특정 DataItem에 대해 필터링 정보가 설정된 파일입니다.
    - Floating License 파일: Studio 및 IDE 폴더에 있는 `licenseServer.inf` 파일, 해당 파일에 사용자명, 이메일 주소, 서버 주소를 입력하면 해당 사용자는 라이선스 인증을 별도로 진행하지 않아도 됩니다. 단, 사이트 내 Mozart Server를 통해 인증 가능하다는 조건에서 사용이 가능한 파일입니다.
  - ***MMC 폴더*** : Mozart Management Console 2 폴더에 있는 `ServerExplorer.vmgr` 파일
  - ***IDE 폴더*** : Floating License를 사용하는 경우 해당 파일이 추가된 IDE 폴더입니다.
    ToRoaming 폴더에 추가 후 모습



ToRoaming\APS Studio 내 파일 추가 후 모습



- ***MozartClientExecute.exe.config*** : MozartClientExecute.exe 를 실행할 때 사용되는 설정값이 저장된 파일입니다. 설치되는 Mozart Studio 실행 파일의 바로가기 파일을 시작 메뉴와 바탕화면에 등록 할 수 있습니다.
  - `GenerateShortcut` 의 `source-files` 에 있는 각각의 Studio 의 value 값이 `true` 이면 `MozartClientExecute.exe` 로 설치할 때 시작 메뉴에 등록 됩니다. 실제 Studio 실행 파일은 모두 Studio 가 설치된 폴더에 설치가 되며, 실행파일의 바로가기

파일이 시작 메뉴에 등록됩니다.

```
// APS Studio 만 시작메뉴에 등록하는 설정
  <ShortCut name="GenerateShortcut" cfg-name="">
    <source-files>
      <add name="AMS_Studio" value="false" />
      <add name="APS_Studio" value="true" />
      <add name="CP_Studio" value="false" />
      <add name="FP_Studio" value="false" />
      <add name="LSE_Studio" value="false" />
      <add name="MP_Studio" value="false" />
      <add name="RTF_Studio" value="false" />
    </source-files>
  </ShortCut>
```

- `#desktop-shortcut` 의 value 값을 `true` 로 설정하면 `GenerateShortcut` 에 `source-files` 에서 `true` 로 설정된 Studio 실행파일이 `MozartClientExecute.exe` 로 설치할 때 바탕화면에 추가됩니다.

```
// Studio 실행 파일의 바로 가기 파일을 바탕화면에 추가하는 설정
<add key="#desktop-shortcut" value="true"/>
```

사용 방법

다음은 재배포를 준비하는 절차입니다. 준비 시 아래 순서대로 진행 바랍니다. Studio.zip의 압축이 풀려 있다는 가정하의 절차인 점 참고 바랍니다.

배포

1. ToBin, ToRoaming 각각 위치에 복사가 필요한 파일 및 폴더를 추가합니다. 이때, `ReadMe.txt` 는 더미 파일이니 삭제해도 무방합니다.
2. `MozartClientExecute.exe.config` 파일을 수정하여 설치할 Studio 의 시작 메뉴 및 바탕화면 추가 여부를 설정합니다.
3. 모든 준비가 끝났으면 배포 할 수 있는 형태로 압축합니다.

설치

1. 배포자한테 받은 파일을 받아 압축을 해제합니다.
2. Mozart Client 설치를 위해 `MozartClientExecute.exe` 를 실행합니다.

AutoUpdate 활용 방법

개요

End-User가 초기 설치 대상자가 아니고 Installer를 통해 업데이트 진행을 원치 않거나 UI 또는 로직 DLL 등의 업데이트만 필요한 경우 AutoUpdate를 이용하여 배포하면 됩니다.

AutoUpdate 사용을 위한 Client 및 Server는 사전에 준비되어 있어야 합니다. AutoUpdate 설정 및 준비는 다음 링크들을 참고 바랍니다.

- **AutoUpdate 설정 (Client & Server)**
- **ManifestEditor를 이용한 manifest.xml 생성/편집 방법**

AutoUpdate를 이용 시 다음과 같은 업데이트를 주로 진행 할 수 있습니다.

- Mozart Studio 및 Domain Library 버전 업데이트
- Targets 정보 갱신
- 사이트 로직 및 UI DLL, config 파일 갱신
- Model Download 목록 갱신

- 그 외 필요한 파일 갱신

AutoUpdate를 통해 배포 할 파일들은 대부분 Bin 아니면 Roaming 폴더에 배포를 하며 배포 목적에 따라 manifest.xml 정보를 갱신하게 됩니다. 다음은 배포 목적에 따른 manifest.xml 설정 방법에 대한 가이드입니다.

Bin 폴더로 배포

Mozart Client 설치 시 기본으로 제공되는 `manifests.xml`에는 `Application Deploy`라는 Process가 정의되어 있습니다. 해당 Process는 `manifests.xml` 내 <location> 요소에 정의된 위치에 파일을 다운 받고 업데이트를 진행합니다. 만약, <location>에 지정된 위치가 없으면 기본은 Studio가 실행된 경로에다가 파일을 업데이트합니다.

Studio 실행 파일은 Bin 폴더에 위치하고 있으니 Bin에 대한 업데이트 진행 시 일반적으로 <location>은 지정하지 않습니다.

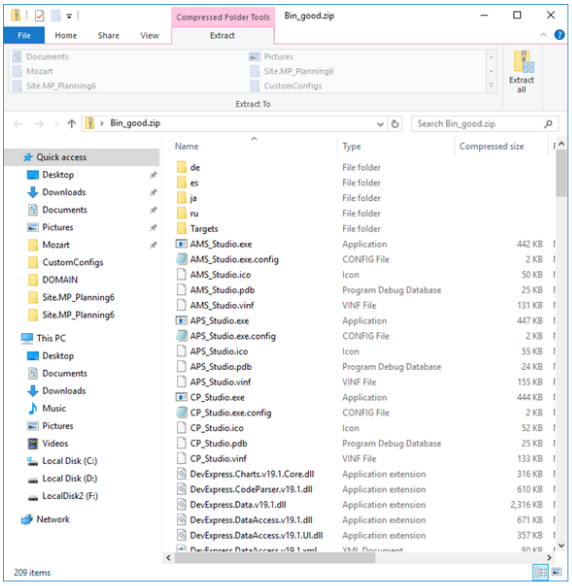
즉, Targets 폴더, 사이트 로직 및 UI DLL, config 파일, 일반 사용자에 대한 Mozart Client 및 Domain Library 업데이트는 기본 제공되는 manifests.xml를 이용하면 됩니다.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" manifestId="{4
CC6D7E2-C32C-4168-9378-4DD885FA73ED}" mandatory="False" xmlns="urn:schemas-microsoft-com:PAG:updater-application-block:v2:m
anifest">
  <title>APS Studio</title>
  <version>2019.116.2.35</version>
  <description>Sample Release</description>
  <application applicationId="{3A8F794F-D23A-484D-B766-98581D801DFD}">
    <entryPoint file="APS_Studio.exe" parameters="" />
    <location>.</location>
  </application>
  <files base="http://192.168.1.54/Mozart.Update" hashComparison="No">
    <file source="update-files.zip" transient="No" />
  </files>
  <activation>
    <tasks>
      <task type="Mozart.AutoUpdater.ActivationProcessors.WaitForApplicationExitProcessor, Mozart.AutoUpdater" name="WaitFo
rApplicationExitProcessor" />
      <task type="Mozart.AutoUpdater.ActivationProcessors.ApplicationDeployProcessor, Mozart.AutoUpdater" name="Application
DeployProcessor" />
      <task type="Mozart.AutoUpdater.ActivationProcessors.StartApplicationProcessor, Mozart.AutoUpdater" name="RestartProce
ssor" />
    </tasks>
  </activation>
</manifest>
```

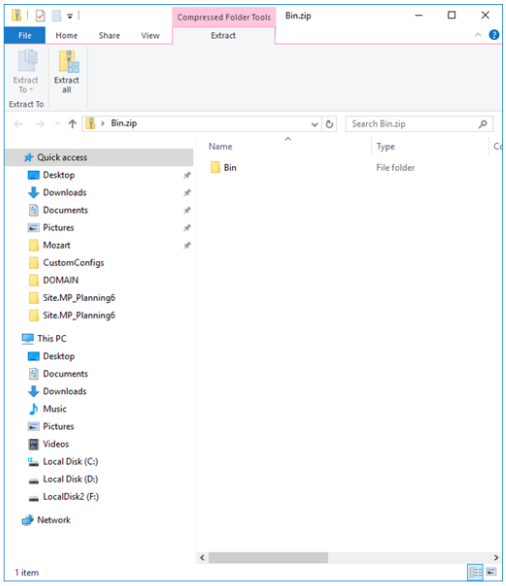
사용 예제

다음은 Mozart Client 버전 업데이트를 예제로 Bin 폴더 배포 방법에 대해 안내 드립니다.

1. 본인 PC에 최신 버전의 Mozart Client 및 Domain Library를 설치합니다.
2. 설치된 경로의 Bin 폴더 안의 파일들을 압축합니다. 압축 파일 내부는 아래와 같이 YES 형태로 되어 있어야 합니다.



Yes (O)



No (X)

3. `manifests.xml` 파일을 열어서 다음 내용을 수정합니다. 기본 제공되는 `manifests.xml` 파일은 Mozart Client 설치 경로의 Updater\Server에 있습니다. (i.e *C:\Program Files (x86)\VMS\Mozart\Client\Updater\Server*)
   a. <manifest> 요소의 manifestId 속성에 GUID 값을 갱신합니다. manifestId는 `ManifestEditor.exe`를 실행하여 `Manifest Properties` 탭의 **[Generate]** 버튼을 클릭하면 manifestId가 생성됩니다. 생성된 값을 manifestId에 복사합니다.
   b. <file> 요소의 source 속성에 압축 파일명을 기입합니다.
4. `manifests.xml` 파일의 변경 내용을 저장하고 AutoUpdate를 진행하는 서버 폴더에 manifests.xml 파일 및 배포 할 압축 파일을 업로드합니다.

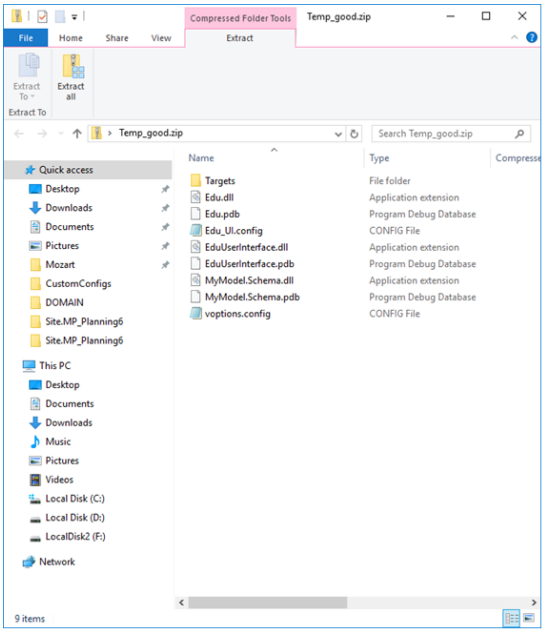
Roaming 폴더로 배포

Roaming 폴더로의 배포는 다양한 방법이 있습니다. 기본 제공되는 manifests.xml 파일의 Process만 이용하는 경우 <location>에 배포 할 Roaming 폴더 경로를 지정하면 됩니다. 그러나 두 경로 이상 동시에 배포가 필요한 경우에는 `Application Deploy` Process만으로 처리가 어렵습니다. 이 경우에는 `Application Deploy` 외 파일/폴더 복사 Process를 함께 활용해야합니다.

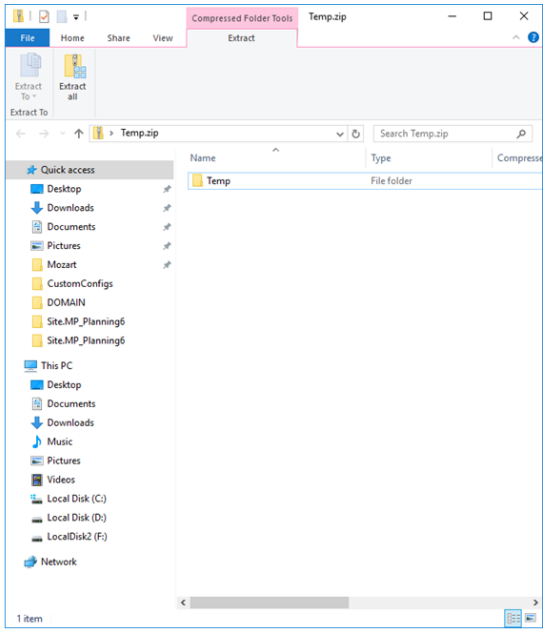
사용 방법

이번 예제에서는 Bin 폴더에 Targets 폴더, UI DLL 및 config, 사이트 로직 DLL 파일 그리고 Roaming에는 APS Studio의 Model 다운로드 경로 정보가 있는 `voptions.config` 파일을 동시에 배포하는 방법에 대한 예제입니다. 본 예제에서는 기본 제공되는 manifests.xml 파일을 이용합니다.

1. 한 곳에 Targets 폴더, UI DLL 및 config, 사이트 로직 DLL 그리고 `voptions.config` 파일을 모아둡니다.
2. 모은 파일들을 압축합니다. 압축 파일 내부는 아래와 같이 YES 형태로 되어 있어야 합니다.



YES(O)



NO(X)

3. 압축한 파일을 manifests.xml 파일이 있는 위치로 복사 합니다. 이때 복사하는 압축 파일의 이름이 manifests.xml의 <file source>에 명시된 이름과 동일해야합니다.
4. `ManifestEditor.exe`를 실행하여 **[Open]** 버튼을 클릭, 편집 할 manifests.xml 파일을 엽니다.
5. `Manifests Properties` 탭에서 **[Generate]** 버튼을 클릭하여 manifestId를 갱신합니다.
6. `Title`, `Version`, `Release Notes` 등의 내용을 필요하면 수정합니다.
7. `Activation Process`로 이동하여 다음과 같이 `File Copy` Process를 하나 추가합니다.
   a. Processor Name에 Process의 이름을 기입합니다. (예: CopyToRoaming)
   b. Processor Type에는 `File Copy`를 선택, [Add] 버튼을 클릭합니다.
   c. Source에는 `Application Deploy` Process가 파일을 다운 받는 위치를 지정합니다. 여기서는 voptions.config 파일을 복사하려고 하니 `%mozart%Bin\voptions.config`라고 기입합니다.
   d. Destination에는 파일이 복사될 위치를 지정합니다. 이번 예제에서는 `voptions.config`를 Roaming\APS Studio 폴더에 복사를 하니 `%appdata%\Mozart\APS Studio`라고 기입합니다.
   e. 파일 덮어쓰기가 필요하면 `Overwrite` 체크 박스에 체크 합니다.
   f. 모든 설정이 완료 되었으면 **[OK]** 버튼을 클릭하여 Process 추가합니다.
8. 파일 복사 후 Bin 폴더에서 해당 파일 삭제가 필요하니 다음과 같이 `File Deleate` Process를 하나 추가합니다.
   a. Processor Name에 Process의 이름을 기입합니다. (예: DeleteFromBin)
   b. Processor Type에는 `File Delete`를 선택, **[Add]** 버튼을 클릭합니다.
   c. Path에는 삭제 할 파일의 위치를 지정합니다. 이번 예제에서는 Bin 폴더 안에 있는 `voptions.config` 파일을 지워야 하니 `%mozart%Bin\voptions.config`라고 기입합니다.
   d. 모든 설정이 완료 되었으면 **[OK]** 버튼을 클릭하여 Process 추가합니다.
9. Processors List에서 Process 실행 순서를 변경합니다. 공교롭게도 `ManifestEditor`에서 순서 변경 기능이 없는 관계로 실행 순서에 맞게 Process를 지우고 다시 추가하는 방법으로 해야합니다. 이 부분에 대해서는 추후 기능 개선을 하도록 하겠습니다. 예제에서는 Studio를 재시작하는 Process를 마지막에 실행되도록 설정하면 되는 관계로 다음과 같이 `Start Application` Process를 지우고 다시 추가합니다.
   a. Processors List에서 `RestartProcessor` 이름을 갖는 Process 선택 후 **[Remove]** 버튼을 클릭하여 제거합니다.
   b. Processor Name에 제거한 `RestartProcessor`를 입력합니다.
   c. Processor Type에는 `Start Application`을 선택하고 **[Add]** 버튼을 클릭하여 Process를 추가합니다.

10. **[Validate]** 버튼을 클릭하여 편집한 manifests.xml 파일에 유효성 검사를 합니다. "The manifest is valid"라고 메시지가 출력되면 정상입니다.
11. Process 추가가 완료되었으면 **[Save]** 버튼을 클릭하여 변경 내용을 저장하고 **[Close]** 버튼을 클릭하여 MaifestEditor를 종료합니다.
12. AutoUpdate를 진행하는 서버 폴더에 `manifests.xml` 파일 및 배포 할 압축 파일을 업로드합니다.

위와 같이 설정하면 `Application Deploy`를 통해 압축파일을 다운 받아 Studio가 실행된 위치인 Bin 폴더에 압축을 해제하고, File Copy/Delete Process가 voptions 파일을 Romaing/APS Studio 복사 후 Bin 폴더에서 해당 파일을 삭제합니다. 즉, Bin 폴더에는 Targets, UI DLL 및 config, 사이트 로직 DLL 파일을 업데이트하고, `voptions.config` 파일만 Roaming 폴더에 업데이트하게 됩니다.

Folder 단위로 업데이트를 할 경우 Folder Copy/Delete Process를 이용하면 되며, 하나 이상의 파일을 다른 위치로 복사를 해야 한다면 파일 한 개당 File Copy/Delete Process를 생성해야합니다.

#create-csv-files

Argument 설명

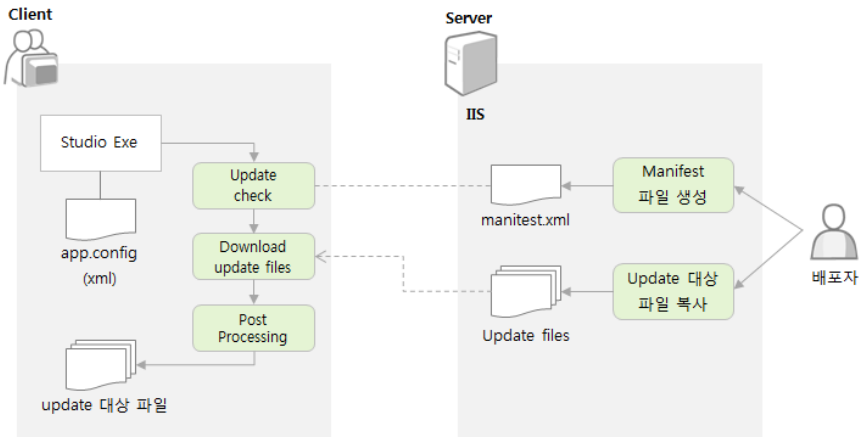
모델을 압축할 때 대상 파일을 `vdat` 포맷 외에 `csv` 포맷으로 추가 생성하여 저장하는 옵션입니다. 기본값은 false이며 이 옵션은 `#zip = true` 일 때 동작합니다. 옵션 설정시에는 압축파일 용량이 증가하니 이점 참고하여 설정하시기 바랍니다.

BucketHistory.csv	6/14/2022 3:02 PM	Microsoft Excel C...	55 KB
BucketHistory.vdat	6/14/2022 3:02 PM	VDAT File	61 KB
EdsOut.csv	6/14/2022 3:02 PM	Microsoft Excel C...	2,070 KB
EdsOut.vdat	6/14/2022 3:02 PM	VDAT File	1,409 KB
EngineRunTime.csv	6/14/2022 3:02 PM	Microsoft Excel C...	1 KB
EngineRunTime.vdat	6/14/2022 3:02 PM	VDAT File	7 KB
EqpPlan.csv	6/14/2022 3:02 PM	Microsoft Excel C...	29,438 KB
EqpPlan.vdat	6/14/2022 3:02 PM	VDAT File	19,423 KB
ErrorHistory.csv	6/14/2022 3:02 PM	Microsoft Excel C...	8,223 KB
ErrorHistory.vdat	6/14/2022 3:02 PM	VDAT File	7,548 KB
ExeFpRunFlag.csv	6/14/2022 3:02 PM	Microsoft Excel C...	1 KB
ExeFpRunFlag.vdat	6/14/2022 3:00 PM	VDAT File	3 KB
InPlan.csv	6/14/2022 3:02 PM	Microsoft Excel C...	2,818 KB
InPlan.vdat	6/14/2022 3:02 PM	VDAT File	1,971 KB

#create-csv-files = true일 때 모델의 결과 데이터

AutoUpdate 설정

Auto Update 는 사용자 사이트에서 직접 개발한 UI Component 외에 Config 파일과 같은 설정파일을 업데이트하는 경우에 사용됩니다. 즉, 사용자 조직의 특정 Server 에 업데이트된 파일을 압축된 형태로 올려놓으면, 사용자 PC 에서 초기에 설정한 Server 로 연결하여 업데이트 파일이 있는 경우 이를 자동으로 업데이트할 수 있습니다. 구조는 아래 그림과 같습니다.



이러한 설정을 하기 위해서는 IIS 가 설치된 회사 내 Server 를 Update Server 로 사용하거나 Auto Update 용도로 별도로 Server 설정을 해주어야 합니다. 아래는 Server 설정 방법입니다.

Server 의 사전 준비사항

.NET Framework 4.0 이상 IIS (Internet Information Server) version 6 이상

Server 설정 방법

1. 업데이트 대상 파일을 저장해둘 폴더를 정합니다.
2. 'IIS (인터넷 정보 서비스) 관리자'를 실행합니다.
3. **[응용 프로그램 풀 > 응용 프로그램 풀 추가]** 메뉴로 응용 프로그램 풀을 추가합니다. 이때 .NET 버전을 4.0으로 설정합니다. (응용 프로그램 명은 임의로 정의합니다. 예: MozartUpdateServer)
4. **[사이트 > Default Web Site > 응용 프로그램 추가]** 메뉴를 사용하여 응용 프로그램을 추가합니다. 아래와 같이 입력값을 설정합니다.
   - **별칭** : 등록할 응용 프로그램의 별칭을 입력합니다. 별칭은 **Client 에서 Server URI 입력 시 필요한 명칭**입니다. Client 에서 **Download URL 값 설정 시 "host 주소/[별칭]/manifests.xml"** 과 같이 입력해야 합니다.
   - **응용 프로그램 풀** : [**선택**] 버튼으로 2번에서 등록한 응용 프로그램 풀을 선택하여 입력합니다.
   - **실제 경로** : 1번에서 설정한 업데이트 대상 파일을 저장해놓을 폴더를 지정합니다.
5. MOZART Client 배포 시 설치된 Auto Updater 폴더에 ManifestEditor 를 사용하여 Manifest 파일을 편집합니다. 자세한 내용은 Manifest Editor 사용법 을 참고하시기 바랍니다.
6. 신규 생성 혹은 작성한 Manifest.xml 파일을 업데이트 대상 폴더에 복사하면 Server 설정은 완료됩니다.

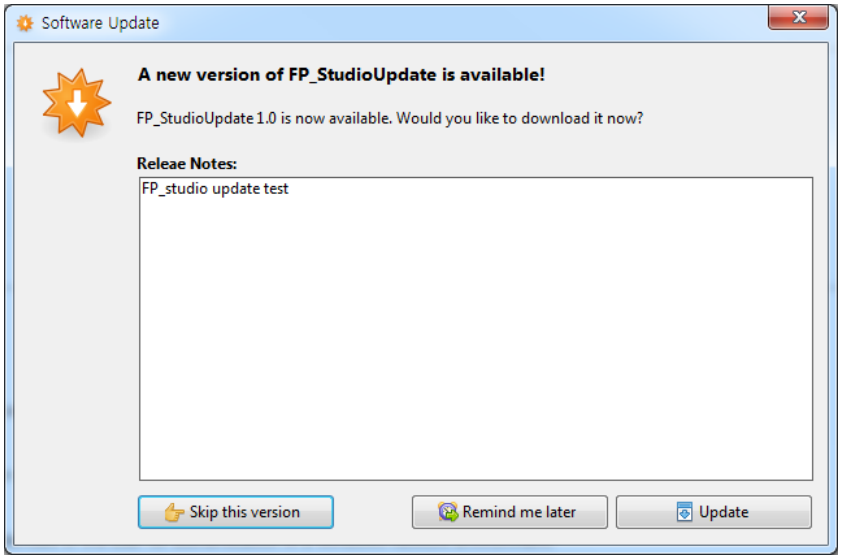
> **Note**
> Default Web Site 에서 사용하는 Port 가 열려있어야 합니다. 기본적으로 사용하는 80 Port 의 경우에도 Server 에서 막아 놓은 경우 오류가 발생할 수 있으니 Server 설정 시 확인하시기 바랍니다.

Client 설정 방법

아래와 같은 순서로 MOZART Studio 에서 클라이언트 UI 및 관련 파일에 대한 자동 업데이트 설정을 할 수 있습니다.

1. Client 에서는 Studio 의 [Tool > Options] 메뉴를 사용하여 Auto Update 여부 및 업데이트 Server 를 설정합니다.

2. 각 설정항목은 아래와 같이 설정합니다.
   - Auto update : 체크를 하면 하부에 입력된 정보를 기준으로 자동 업데이트가 되며, 체크해제 시 자동 업데이트를 하지 않습니다.
   - Application Id : Studio 프로그램의 고유 아이디입니다. 사용자가 변경하지 않습니다. Server 의 Manifest 파일과 비교할 때 대상 Application 에 대한 업데이트 정보만 비교합니다.
   - Download URL : Update Server 의 URL 을 입력합니다. 형식은 다음과 같습니다.
     + 형식 : http://[Server IP]/[Server 설정 시 응용 프로그램 별칭]/manifests.xml
     + 변경부분은 [Server IP], [Server 설정 시 응용 프로그램 별칭] 입니다. Client UI 개발/운영조직에게 해당 부분을 확인하고 설정합니다.
   - Downloader : Downloader 를 선택합니다. 기본은 BITDownloader 가 선택되어 있습니다.
   - Parameters : Server 와의 인증에 사용되는 Parameters 로 수정하지 않습니다.
3. 항목 설정 후 Studio를 재시작하면 아래와 같은 다운로드 선택창이 활성화됩니다.

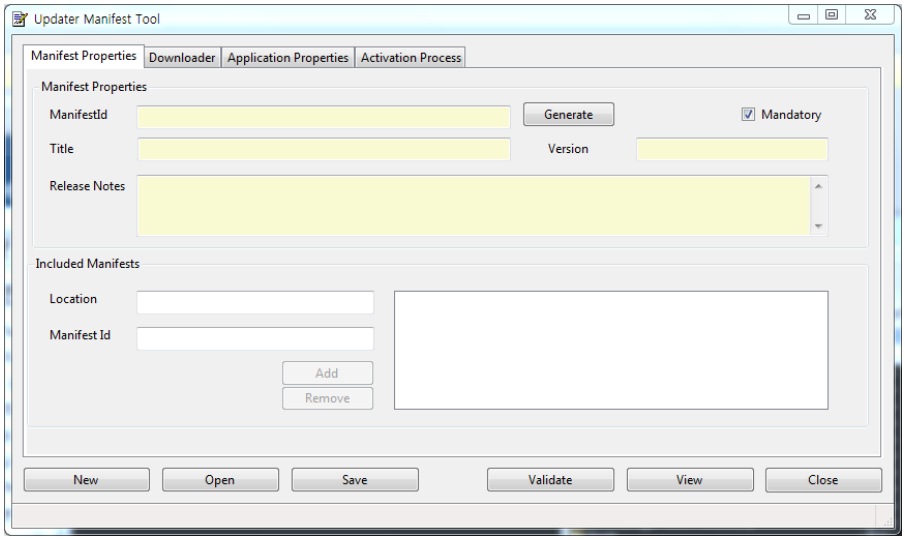


   - Skip this version : 다음번 실행 시에도 해당 버전의 업데이트를 체크하지 않습니다.
   - Remind me later : 다음번 실행 시에 해당 버전의 업데이트 수행 여부를 묻습니다.
   - Update : 업데이트 내용을 다운로드 받고 Studio 를 재실행합니다.

Manifest Editor 사용법

Manifest Editor 는 Manifest 파일을 생성/편집하기 위한 편집기입니다. MOZART Client 설치 시 MOZART 설치경로 아래의 Updater 폴더에 ManifestEditor 가 함께 설치됩니다.

1. Server 실행파일 폴더의 ManifestEditor 를 실행합니다. 총 4개의 Tab 으로 구성된 다이얼로그가 실행됩니다.



2. Manifest Properties Tab 의 내용을 입력합니다.

   - **ManifestId** : 신규 업데이트 파일이 배포될 때마다 변경되는 배포 ID 와 같습니다. Client 는 해당 ManifestId 의 값을 비교하여 Auto Update 여부를 판별합니다. 우측의 [Generate] 버튼을 통해 신규 GUID 를 생성할 수 있습니다. 배포 시마다 변경되어야 합니다. (※xml 문서 Key = manifestId)
   - **Title** : 해당 Manifest 의 이름입니다. (※xml 문서 Key = title)
   - **Version** : 배포되는 제품의 버전을 기록합니다. 배포 시 규칙에 따라 업데이트합니다. (※xml 문서 Key = version)
   - **Release Note** : 배포되는 버전의 수정사항에 대해 간략히 기록합니다. 배포 내용에 따라 업데이트합니다. (※xml 문서 Key = description)

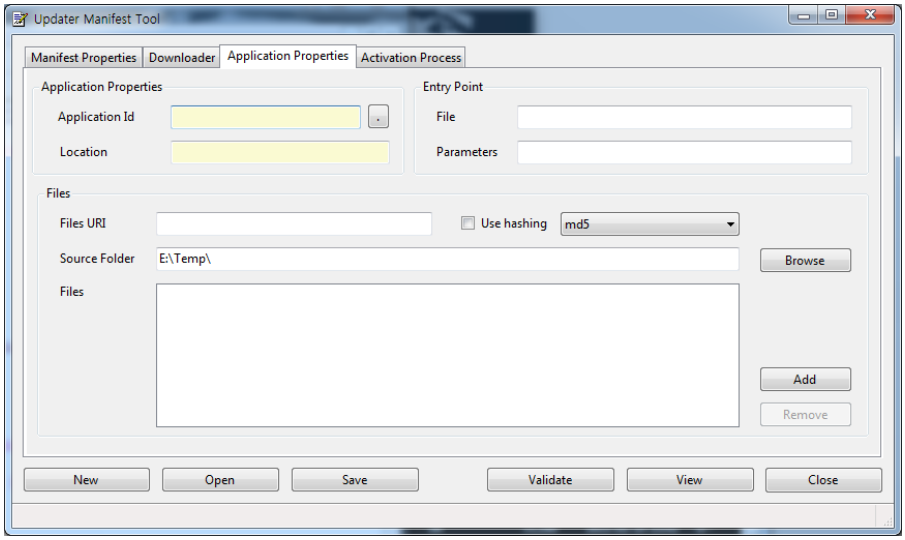
   Manifest File 예시

```
  <?xml version="1.0" encoding="utf-8"?>
  <manifest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  manifestId="{AF7A3BD5-10A1-4155-BBF8-631906D86DAE}" mandatory="False" xmlns="urn:schemas-microsoft-com:PAG:updat
er-application-block:v2:manifest">
    <title>FP Studio</title>
    <version>1.0.3</version>
    <description>first release</description>
    <application applicationId="{3A8F794F-D23A-484D-B766-98581D801DFD}">
      <entryPoint file="FP_Studio.exe" parameters="" />
      <location>.</location>
    </application>
    <files base="http://xxx.xxx.xxx.xxx/MozartUpdate" hashComparison="No">
      <file source="update-files.zip" transient="No" />
    </files>
    <activation>
      <tasks>
        <task type="Mozart.AutoUpdater.ActivationProcessors.WaitForApplicationExitProcessor, Mozart.AutoUpdater" n
ame="WaitForApplicationExitProcessor" />
        <task type="Mozart.AutoUpdater.ActivationProcessors.ApplicationDeployProcessor, Mozart.AutoUpdater" name
="ApplicationDeployProcessor" />
      </tasks>
    </activation>
  </manifest>
```

   상기 예시 파일은 **[Mozart Client 설치 폴더 > Updater > Server]** 에 **manifests.xml** 파일로 포함되어 있습니다.

3. Application Properties Tab 의 내용을 입력합니다.

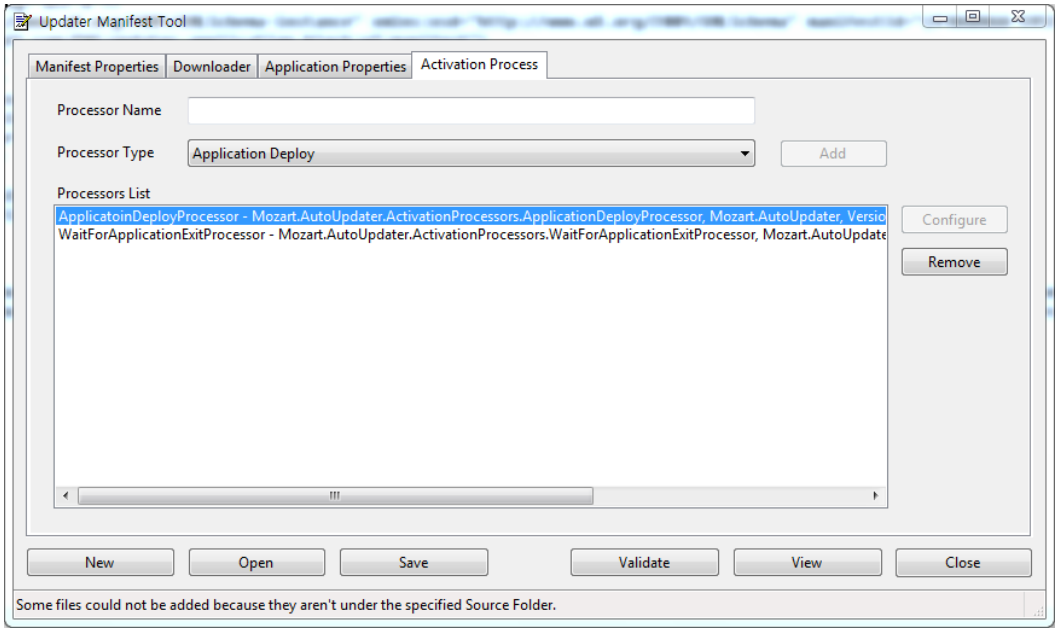
   업데이트 대상이 되는 Application 의 정보를 설정합니다. 필수 설정 항목은 아래와 같습니다.



- **Application Properties**
  - **ApplicationId** : 업데이트 주 대상 프로그램의 GUID 로 대상 파일의 App.config 파일에 있는 GUID 를 사용합니다. 두 개의 GUID 를 항상 같도록 설정하면 됩니다. (※xml 문서 Key = application/applicationId)
  - **Location** : Download 되는 파일 저장 위치입니다. (※xml 문서 Key = application/location)
- **Entry Point**
  - **File** : 실행 파일 명입니다. FP_Studio 에 대한 업데이트를 설정하는 경우 'FP_Studio.exe' 로 설정합니다. (※xml 문서 Key = application/entrypoint file)
  - **Parameters** : 실행 시 Parameters 를 설정합니다. Studio 업데이트 시 별도 Parameters가 필요하지 않습니다.
- **Files**
  - **Files URI** : Download 받을 파일이 위치한 URI 를 설정합니다. 일반적으로 Server 에서는 LocalHost 에 설정한 경로를 입력합니다. (※xml 문서 Key = files/base)
  - **Source Folder** : 파일이 위치한 로컬 폴더를 선택합니다.
  - **Files** : 선택한 로컬 폴더의 파일 중 Download 받을 파일 명을 입력합니다. 일반적으로 업데이트할 파일을 압축한 파일로 선택합니다. (※xml 문서 Key = files/source)

4. Activation Process Tab 의 내용을 입력합니다.

   Update 시 실행할 Processor 를 선택합니다. 복수로 선택 가능합니다.



Processor Name 을 입력하고 Processor Type 선택 후 [**Add**] 버튼으로 Processor 를 추가합니다. Processor Type 의 유형은 아래와 같습니다. 여러 Process 를 함께 등록할 수 있습니다. MOZART Studio 의 자동 업데이트를 위해 꼭 필요한 항목은 Application Deploy, Wait For Exit 두 가지입니다. Server 설치 폴더에 함께 배포된 manifests.xml 파일을 참고하시기 바랍니다. **(※xml 문서 Key = activation/tasks/task)**

- **Application Deplo**y : 다운 받은 파일을 업데이트 대상 폴더로 복사, 파일이 압축 파일이면 압축 해제합니다.
- **File Copy** : config 파일에 설정된 대로 특정 파일을 특정 위치로 복사합니다.
- **File Delete** : config 파일에 설정된 대로 특정 경로의 특정 형식의 파일들을 삭제합니다.
- **Folder Copy** : config 파일에 설정된 대로 특정 폴더를 특정 위치로 복사합니다.
- **Folder Delete** : config 파일에 설정된 대로 특정 폴더를 삭제합니다.
- **GAC Util** : GACUtil을 사용해서 config 파일에 설정된 대로 GAC 관리합니다. (등록/삭제/...)
- **Hash Validation** : 다운 받은 파일 중 config에 설정된 특정 파일과 Server 의 원본 파일 간 Hash 값 비교합니다.
- **Install Util** : InstallUtil을 사용해서 Service 를 관리합니다.
- **MSI** : config 파일에 설정된 패키지를 설치/제거/패치합니다.
- **Start Application** : 파일 다운로드 및 업데이트 완료후 프로그램을 다시 시작합니다.
- **Uncompress** : config 파일에 설정된 대로 특정 압축 파일을 특정 경로에 압축 해제합니다.
- **Wait For Exit** : 업데이트를 위해 Client 를 종료하고, 업데이트가 완료되기를 기다립니다.