Deployment Options in Matillion ETL – Using Multiple Projects

This article describes the second of three commonly-used choices for how to manage and deploy your Matillion solution between multiple environments: for example development – test – production. Note that in this series we're looking exclusively at options which are available through Matillion ETL's web user interface. Additional options are available using Matillion's REST API.

WITH THIS METHOD, YOU CREATE MULTIPLE PROJECTS. FOR EXAMPLE:

ONE PROJECT FOR DEVELOPMENT
ONE PROJECT FOR QA
ONE PROJECT FOR PRODUCTION

YOU THEN PROMOTE CODE BETWEEN PROJECTS WHEN IT HAS BEEN DEVELOPED AND TESTED.

In order to describe the details, first a few definitions. What do we actually mean by "Matillion code"? And what's a Project as opposed to an "Environment"?

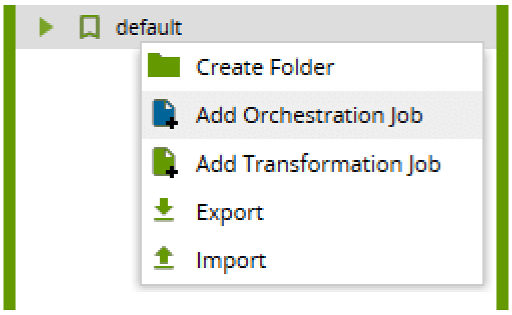
Orchestration and Transformation Jobs

Matillion is an ELT tool. It does two main things on your behalf:

1. Data ingestion – in other words loading data from external sources into the database
2. Data transformation – getting your data ready for visualization or analysis, for example, integration, reshaping, applying business rules, aggregation, and deriving calculated fields.

These two things are implemented by Orchestration Jobs and Transformation Jobs respectively.

Deployment Options in Matillion ETL – Using Multiple Projects

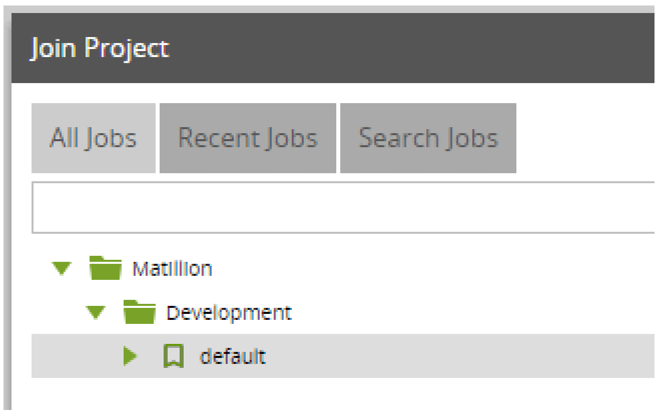


Orchestration jobs fulfill an additional command-and-control function: they define the overall sequence of events and can be scheduled. Orchestration jobs can call Transformation jobs as part of their work.

“Matillion Code” means the definition of the Orchestration and Transformation Jobs that you are using. Matillion Jobs always exist inside a Project.

Matillion Projects

A Project is simply a collection of metadata. When you’re working in Matillion you are always within the context of a Project. When you log in, it’s the first dialog that pops up after the login credentials.



Deployment Options in Matillion ETL – Using Multiple Projects

You'll need to select a Group, a Project and a Version. In the above screenshot, these are:

- Group: **Matillion**
- Project: **Development**
- Version: **default**

Every Group can contain multiple projects. You can define more than one Group although it's often simplest to stick with just one. A good choice for the Group name is your company name.

One Project can contain multiple Versions. You can create your own version snapshot at any time, but there's always a *current* one named "default".

Among other things, Projects contain:

- Orchestration and Transformation Job definitions
- Folder structure
- One or more Environments

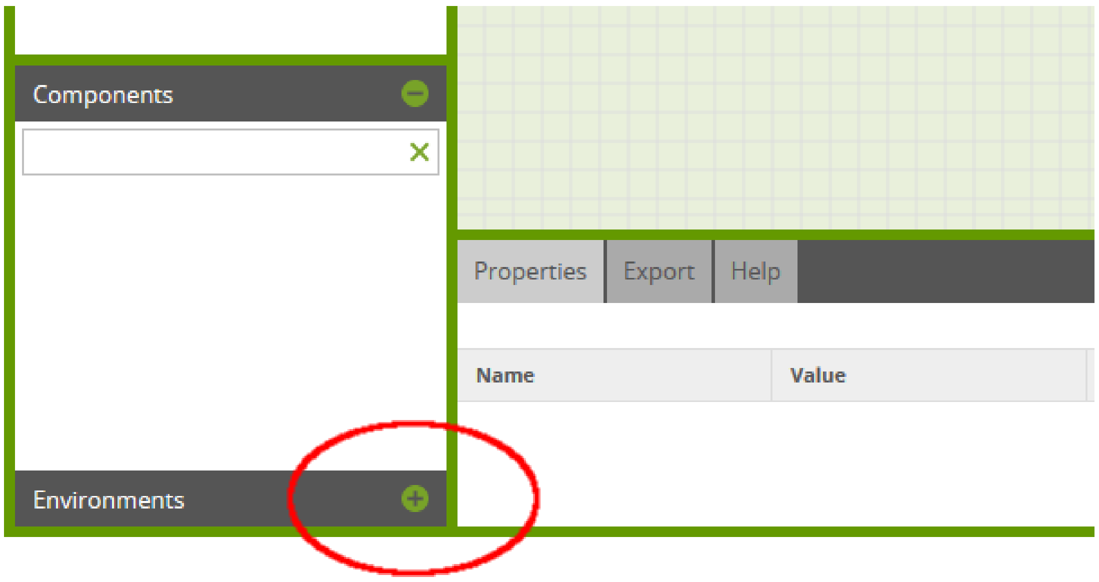
It's the "Environments" which define where the Matillion Jobs can be executed at runtime.

What's a Matillion Environment?

A Matillion Environment defines a target data warehouse.

To manage your Environments you'll need to expand the panel at the bottom left of the screen, which is minimized by default.

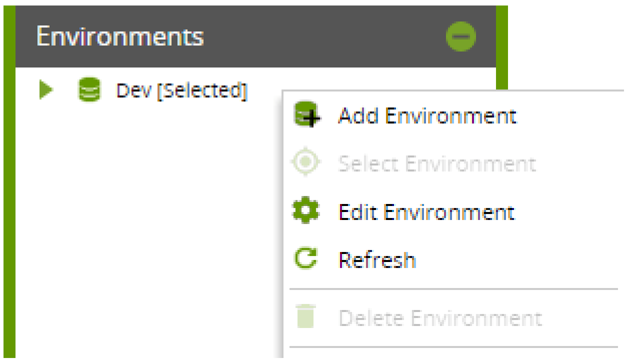
Deployment Options in Matillion ETL – Using Multiple Projects



When you first launched Matillion, you went through an initial configuration screen. This asked for details of the target data warehouse plus a couple of other configuration items. For this reason, you'll always have at least one Environment.

You can manage environments through the right-click context menu. The Edit Environment option will take you back to that initial configuration screen, where you can change the settings if necessary.

Deployment Options in Matillion ETL – Using Multiple Projects



You can add, edit and remove Environments using these options.

Note that:

1. Matillion has an overall cap on the total number of Environments that may exist within your installation. The cap depends upon the instance size. When you use multiple Projects then each Project will have at least one Environment, and they all count towards the cap. You'll need to switch between Projects to count the total number.
2. You are not permitted to delete the last Environment within a Project.

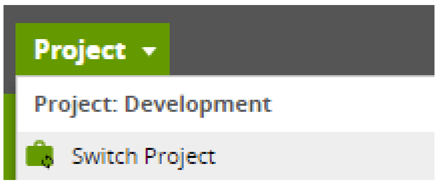
Solution Overview

With this code deployment option:

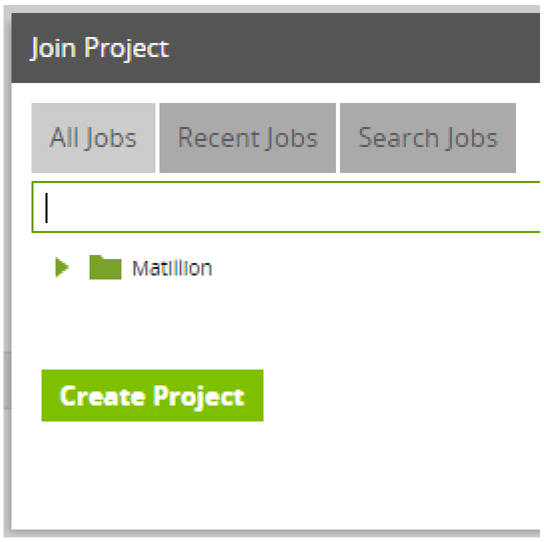
- You define multiple Projects (development, test, production)
- Every Project owns one corresponding Environment (development, test, production)
- You copy the code from one Project into another once it's ready to be promoted

If you started out with just one "Development" project, you might now add a "Production" project. Choose Switch Project from the Project menu.

Deployment Options in Matillion ETL – Using Multiple Projects

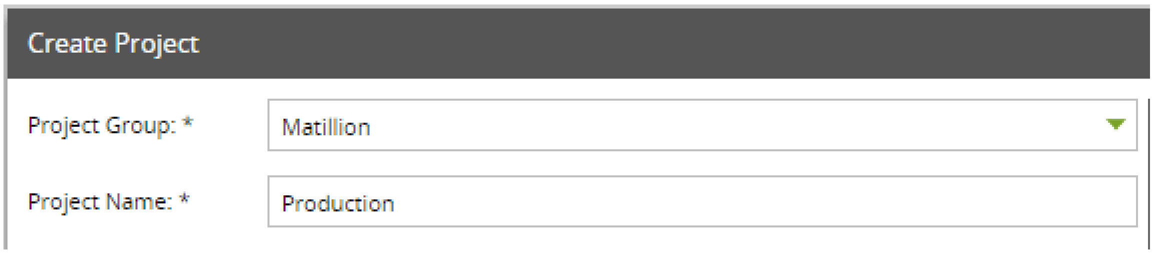


Then press the **Create Project** button.



Be sure to choose the same Group name (your company name) from the **Project Group** dropdown list, and name your new **Project Name** "Production":

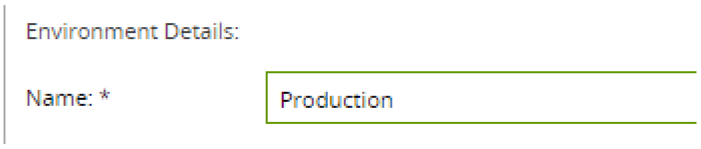
Deployment Options in Matillion ETL – Using Multiple Projects



Create Project

Project Group: *	Matillion
Project Name: *	Production

You're also creating a new Environment at this point (remember, every Project must always have at least one Environment), and it's a good idea to name this "Production" too:



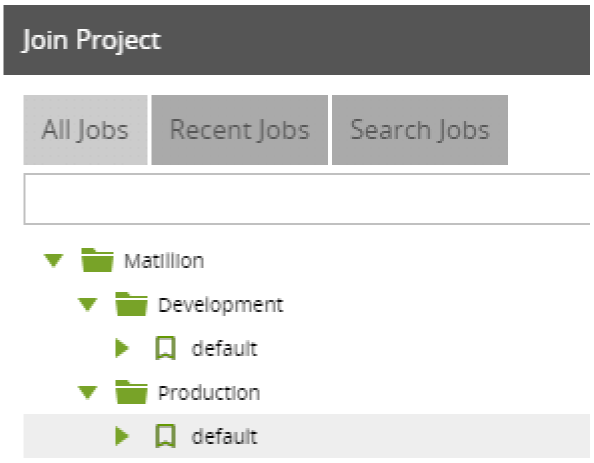
Environment Details:

Name: *	Production

Now when you follow Project / Switch Project you should find two Projects:

- A "Development" project, which owns an Environment named "Dev", pointing to the Development target data warehouse.
- A "Production" project, which owns an Environment named "Production", pointing to the Production target data warehouse.

Deployment Options in Matillion ETL – Using Multiple Projects



Developers would log onto the Development project to do their work. When finished, they would export their work ready for a DBA or DevOps person to log onto the Production project to import and schedule the jobs.

You may extrapolate this scenario to fit any combination of environments that you need. Some examples are:

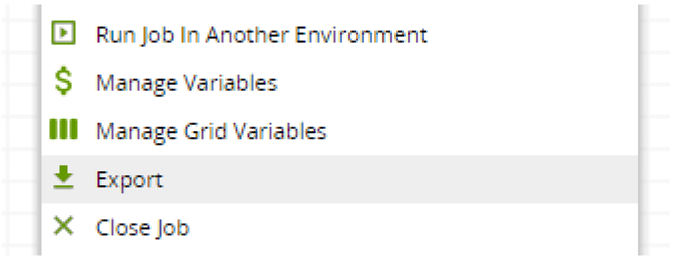
- Dev → Production
- Dev → System Test → Production
- Dev → System Test → UAT → Production

Exporting and Importing Jobs

Matillion's mechanism for copying job metadata from one Project into another is via the Export/Import feature.

While editing a Job, you can find the Export option from the right-click context menu:

Deployment Options in Matillion ETL – Using Multiple Projects



You can also get to the same dialog by following the Project / Export menu. The dialog allows you to select one or more Jobs you want to save and download as a JSON file. You can put this file into external version control.

From the Project / Import menu, you can import a Matillion-generated JSON file, and choose which jobs to import into the target Project.

BY DEFAULT, THE JOBS WILL BE ADDED INTO THE SAME FO
STRUCTURE AS THEY WERE AT THE SOURCE.

YOU'LL FIND IT EASIEST TO MAINTAIN EXACTLY THE SA
FOLDER STRUCTURE IN DEVELOPMENT AS IN PRODUCTIO

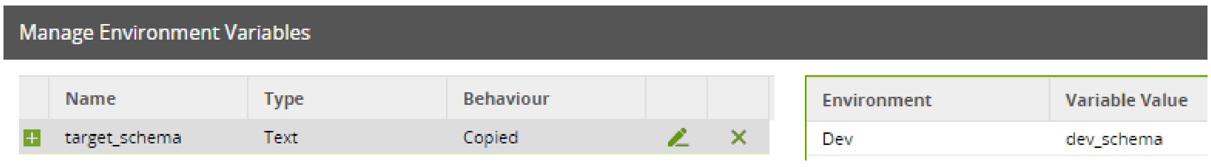
Matillion also has various REST API endpoints for exporting and importing Jobs. These are used for automation, although be aware that API-generated metadata formats are not compatible with the UI-based export and import.

A note on Environment Variables

Environment variables have a very useful feature in that you can set a different default per Environment.

Deployment Options in Matillion ETL – Using Multiple Projects

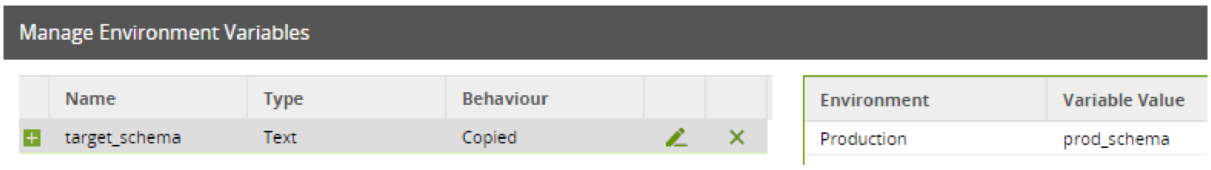
While logged into the Development Project, you might have a variable named ***target_schema***. Its default value (for the Dev environment) is set to ***dev_schema***.


Manage Environment Variables

Name	Type	Behaviour
target_schema	Text	Copied

Environment	Variable Value
Dev	dev_schema

Similarly, while logged into the Production Project, the same variable can exist, but with a default value (for the Production environment this time) set to ***prod_schema***.

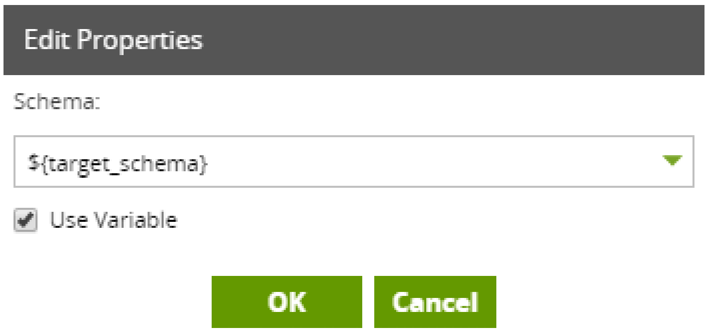

Manage Environment Variables

Name	Type	Behaviour
target_schema	Text	Copied

Environment	Variable Value
Production	prod_schema

So, instead of hardcoding the schema name of a Table Input component, for example, you could set it to ${target_schema}

Deployment Options in Matillion ETL – Using Multiple Projects



Schema:

${target_schema}

✓ Use Variable

Then, in Development the Table Input would automatically read from the ***dev_schema***, and once deployed into the Production Project it would automatically start to read from the ***prod_schema*** with no code change required.

Incidentally, this is why when you export an Environment Variable, the default value is not among the properties that are saved into the JSON export file.

Summary

When you use multiple Projects to manage code deployment, you set up multiple parallel copies of the codebase in a different Project, and each copy pointing to a different target data warehouse (dev, test or production).

You control exactly when and what code you promote to the different environments. Also, you can take advantage of the Environment Variable feature which allows you to automatically customize job behavior according to where it is running.

Backup mechanisms work as normal and are still recommended.

Be aware of Matillion ETL's cap on the total number of Environments that you can create because this applies across all projects simultaneously.

Deployment Options in Matillion ETL – Using Multiple Projects

Some metadata items, including user logins, OAuth credentials, and the Password Manager are not Project-specific. These are server-wide and automatically apply to every Project. More on this subject in the third article in this series.

Reference : https://www.matillion.com/blog/deployment-options-in-matillion-etl-using-multiple-projects