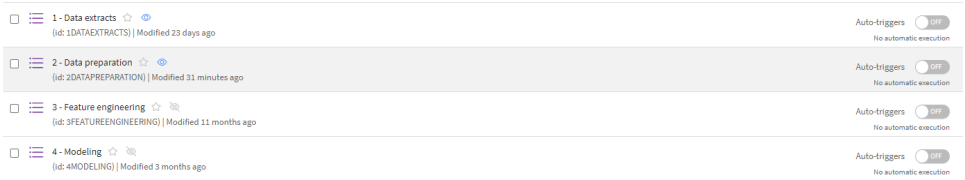
Scenarios

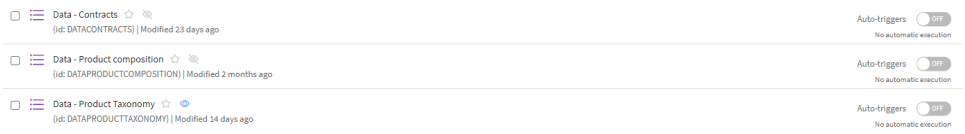
Scenarios architecture

In the project, we use several layers of scenarios.

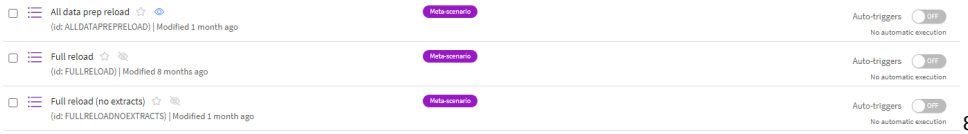
- Numbered scenarios: they represent each big step of the project methodology and are launched in order.



- Data scenarios: all prefixed by "Data -", they are called by the "All data prep reload" scenario and can differ from one GBU to another.



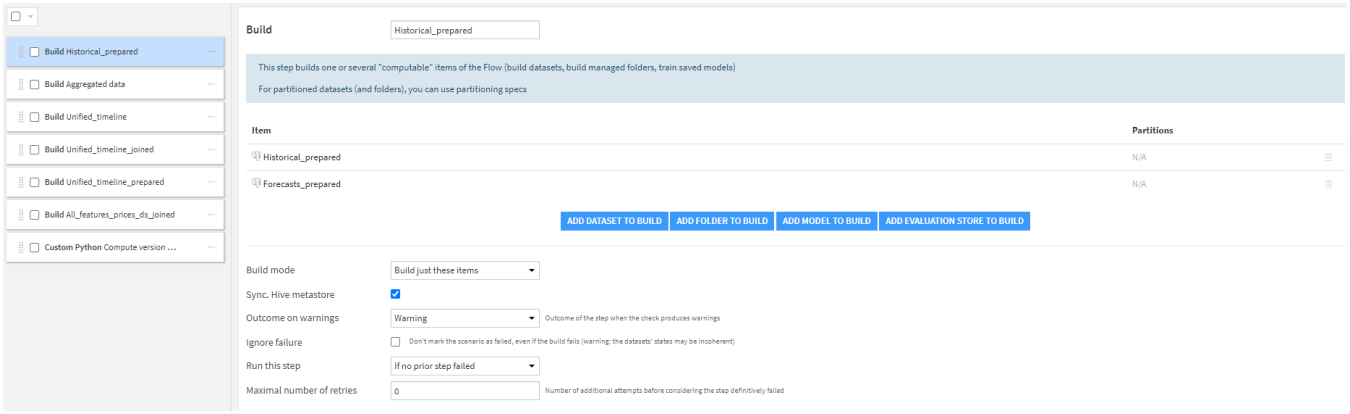
- Meta-scenarios: These scenarios are calling other scenarios in the right order to run big parts if not the whole Dataiku Flow. They are tagged to be retrieved more easily.



805

Good practices

There are a few things to know when using scenarios:



- When several datasets/folders are built inside the same step (as Historical_prepared and Forecasts_prepared in the screenshot above), they will be built **at the same time.** Therefore, if there are dependencies between them, several steps must be used instead.
- By default, the build mode is set as "Build dependecies then these items". This causes two issues:

- Dataiku might **fail** at building all dependencies on big projects like we have.
- It makes reloading **only parts** of the Flow harder.
  Therefore, it is necessary to switch to **"Build just these items"** mode.

List of scenarios

For clarity, only scenarios with specific behavior will be listed below.

Prod reload

The only automated scenario used in the production environment. It will launch the other scenarios iteratively from the extraction of input data to the output data storage for front-ends.

Trigger

This scenario should have an active trigger on both the MASTER project in design and the automation project (refer to the technical architecture here if these terms are not familiar).

The trigger will be SQL based but the condition of reload will be different from one GBU to another (based on historical date change for SpP and forecast date change for Novecare).

Custom step

The first custom step is used to update the version_name of the project automatically, this allows us to make sure the MASTER and prod project versions are always correct and different from the versions used in dev environments.

2 - Data preparation

Custom step

Contains a custom python step at the end to update the "version_date" variable based on the latest data available (either in forecasts or historical also, depending on the GBU).

This is essential to have the proper "version_data_date" displayed in the final outputs and to make sure we only keep the latest run of each month for a version in case we have several (reruns in prod due to errors or bad data for example).

Run Hyperparams Opt

Once in a while (recommended every 6 months or so), the model will have to be re-optimized for each product family. This is the purpose of this scenario.

It contains a first custom Python step to enable the *run_cross_val* variable which is normally set to False for the monthly runs. Then, the Optimized_hyperparams dataset is executed and another custom Python step sets back the *run_cross_val* variable to False.