Platform 201

We will record today's presentation and provide the recording and slides after.

Introduce yourself in the Chat!

Name
Pronouns
Organization
What brought you here today?

Training Norms

1. Please keep yourself muted, help us reduce background noise.
2. Raise your hand or type questions into the chat.
3. Feel free to ask other participants questions in the chat!
4. We all have varying levels of Platform knowledge on this call. Ask the questions you have–no question is too simple or too advanced!

Session 2

Scripts - SQL & Python
Version Control
Workflows
Questions

Scripts - SQL & Python

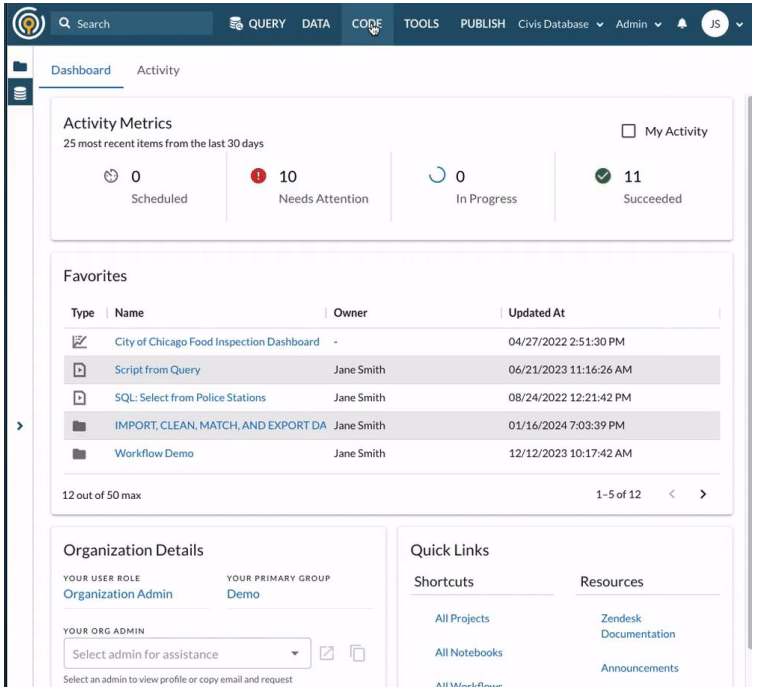
Creating Scripts
Parameters
Automation
Notifications

Scripts

Creating a new Script

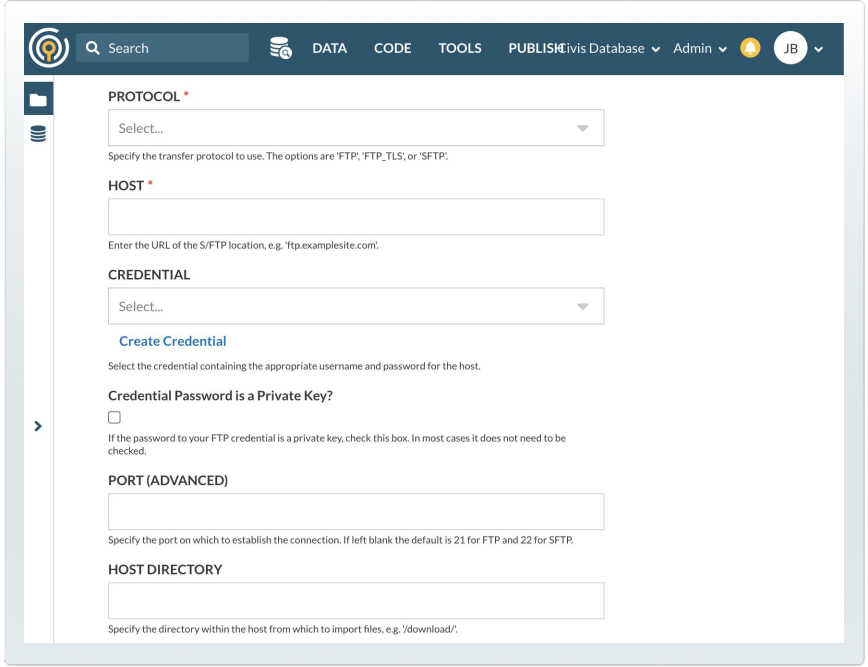
Use the Code tab to access your Scripts

- To create a new Script click Code from the top nav bar then select the language you want to use under Scripts
    - For SQL Scripts the current database selected from your dropdown will be used
- Enter your code into the editor
- Click Run
- SQL Scripts that end in a SELECT statement produce a CSV file of the results.



Scripts
Parameters

- Platform allows you to use custom parameters in scripts
- These parameters can be set and accessed in the UI
- This feature allows more technical team members to write the code while less technical team members edit and run the scripts

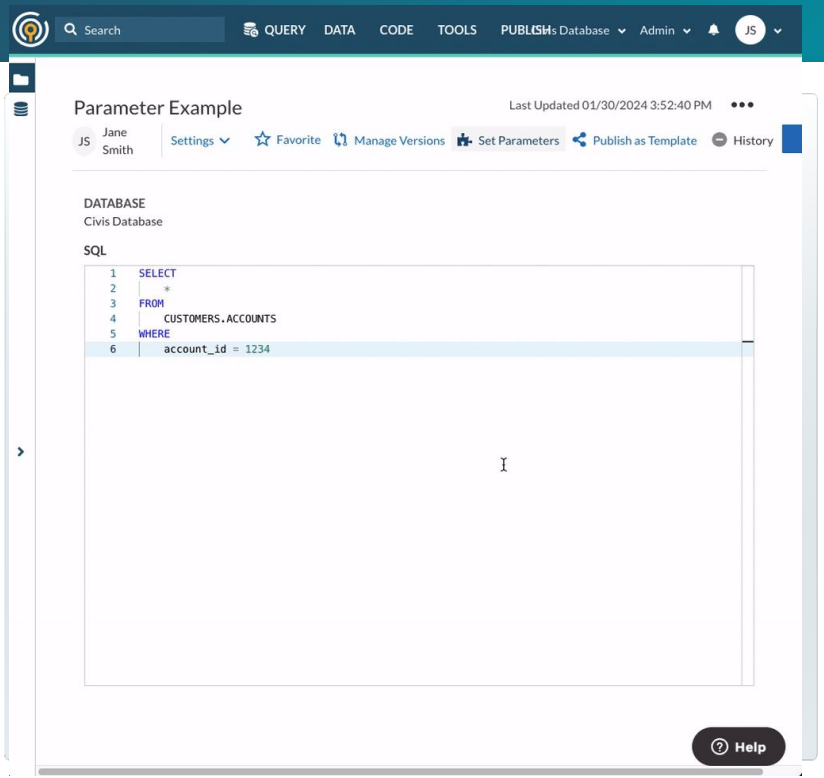


More information can be found in Civis Help Documentation: [Scripts: Parameters]

Scripts

Parameters

- From any Platform job, click the puzzle icon “Set Parameter”. Within the Parameters menu:
  - Click “+ New Parameter”
  - Fill out the required fields, and optional fields as desired
  - Select “Add”
  - Select “Save” before
- Once you add a parameter, you should see it as an option in the script



Scripts

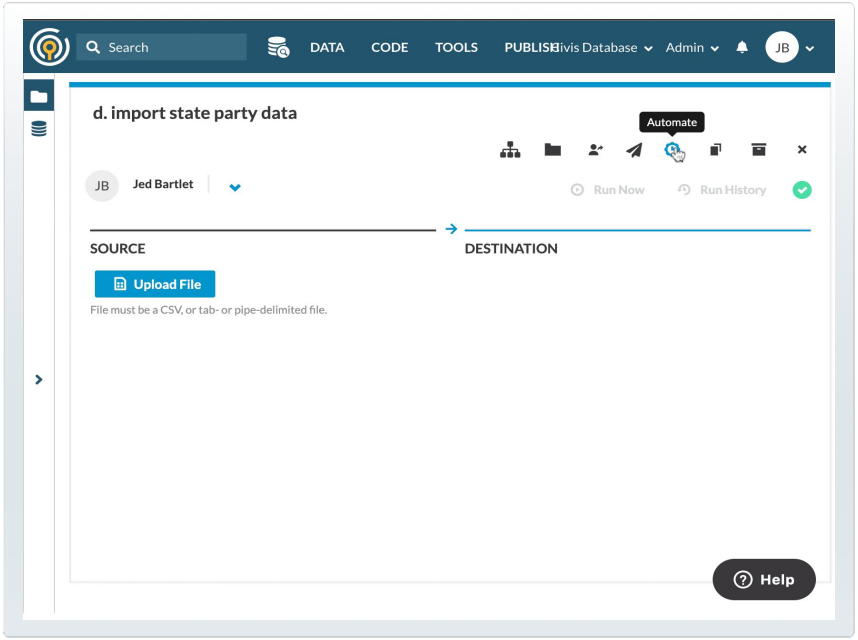
Parameters

Language	Parameter Format
SQL	Wrap parameters in double curly braces (e.g. {{`myParamName.literal`}})
Python	Read parameters as environment variables (e.g. `os.environ['myParamName']`)
R	Read parameters as environment variables (e.g. `Sys.getenv('myParamName')`)
Container	Read parameters as environment variables formatted in the script's coding language

Scripts
Automation

- From the Standard Action Menu, select the “Automate” icon
- Toggle “Automate” to on
- Scroll through the automation options and choose the schedule

Note: the automation options may look slightly different depending on the type of Platform Object you are automating. For example, a SQL script allows you to automate by day/month/time, whereas a csv import allows you to automate by day/time



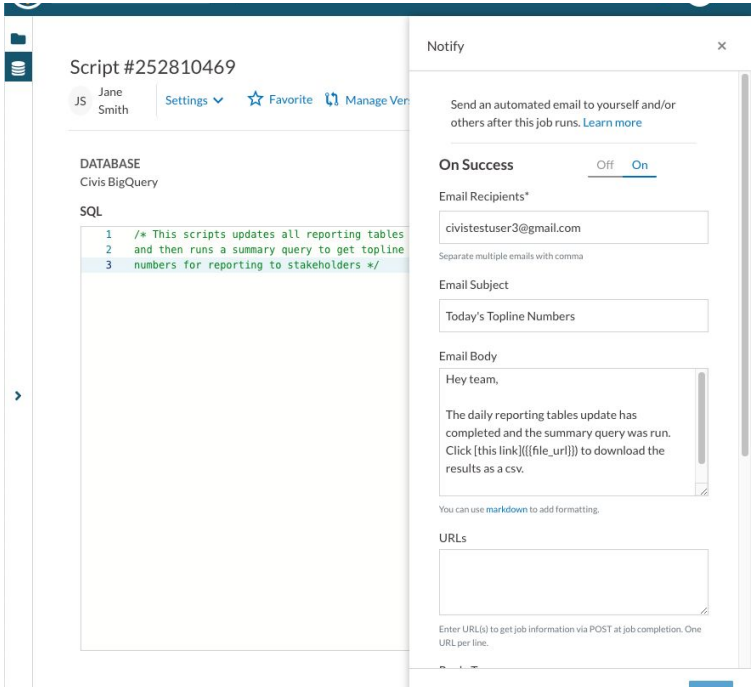
More information can be found in Civis Help Documentation: [Key Features: Automate]

Scripts
Notifications

Job outputs in Notification Emails

It is possible to add files, markdown tables or json formatted data produced by your Job to the success notifications.

- For SQL Scripts you can add a link to download the output CSV to your email notification by using [link text]({{file_url}})
- We have example code in Python and Javascript that will convert a SQL query into a markdown table and topline number that can be added to your email



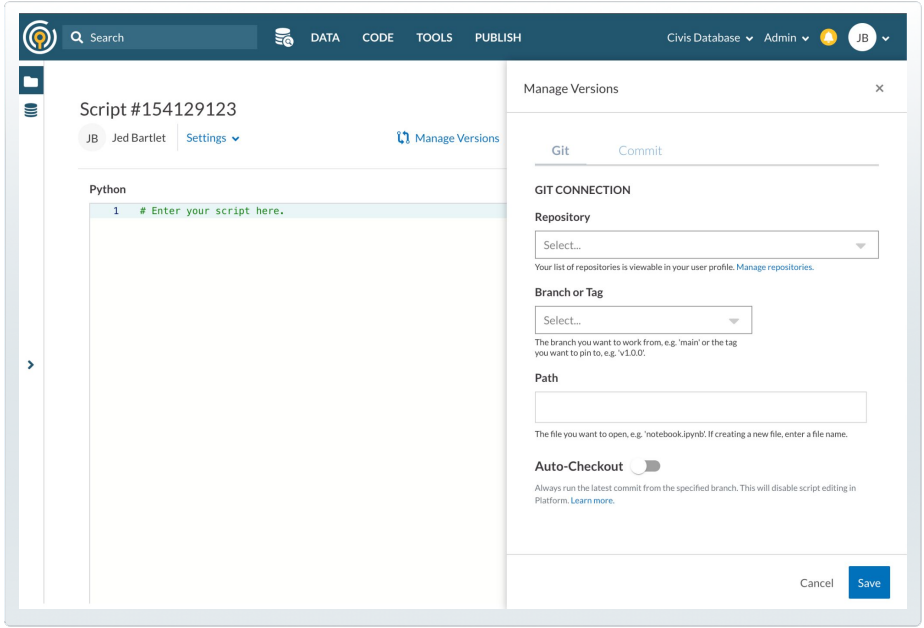
Version Control

Basics
Connecting Platform to GitHub
Setup
Commit, Push, and Checkout
Autocheckout

Version Control

Basics

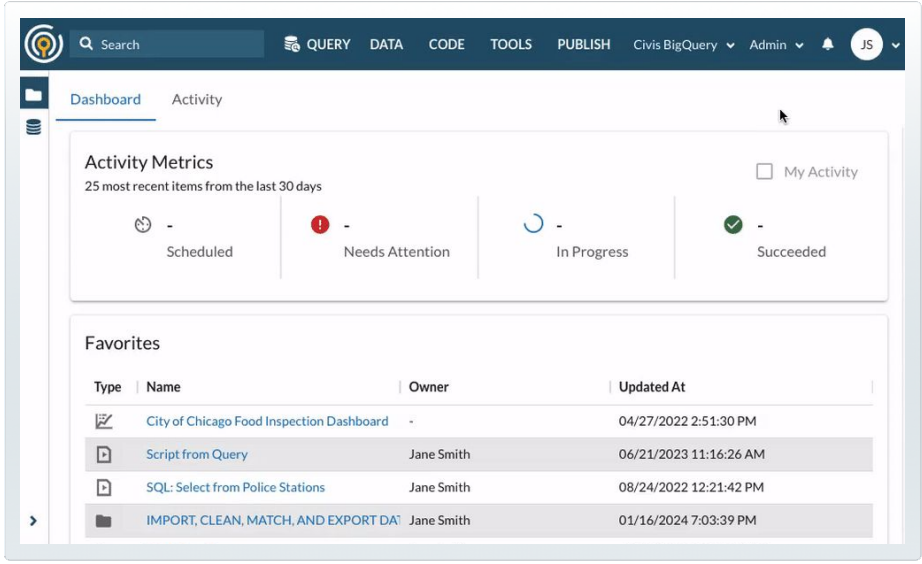
- Civis Platform supports version control for the following:
  - SQL, R, and Python Scripts
  - Workflows
  - HTML Reports
  - Notebooks
- Civis Platform supports version control via GitHub



Version Control

Connecting Platform to GitHub

- To connect to a GitHub:
  - Log into GitHub and have it open in another tab
  - Navigate to the Git Repos tab under My Profile
  - Click Add Repository
  - Click Connect to GitHub
  - If not automatically redirected to GitHub, click into your open GitHub Tab and then back to Platform
  - Search for and add a repository



Version Control

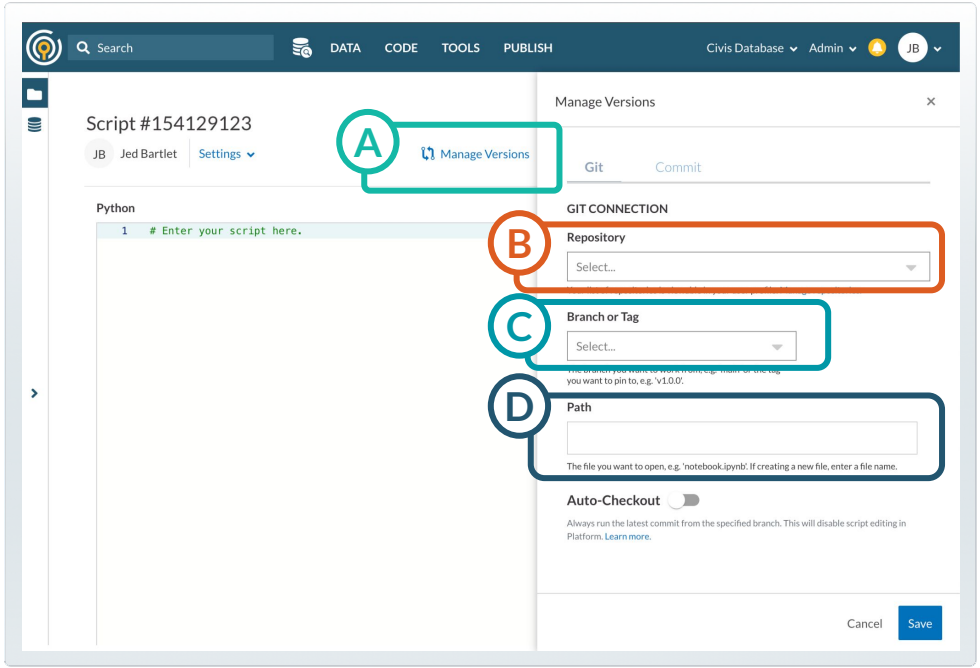
Setup

A. Manage Versions
to set up a Platform object for version control, click “Manage Versions”

B. Repository
a list of repositories your Git profile has access to

C. Branch
specific branch you would like to use. The branch must already exist in your repo

D. Path
If the file does not yet exist, Platform will create a new file for you in the repo.
Note: path must be a file, not a directory.

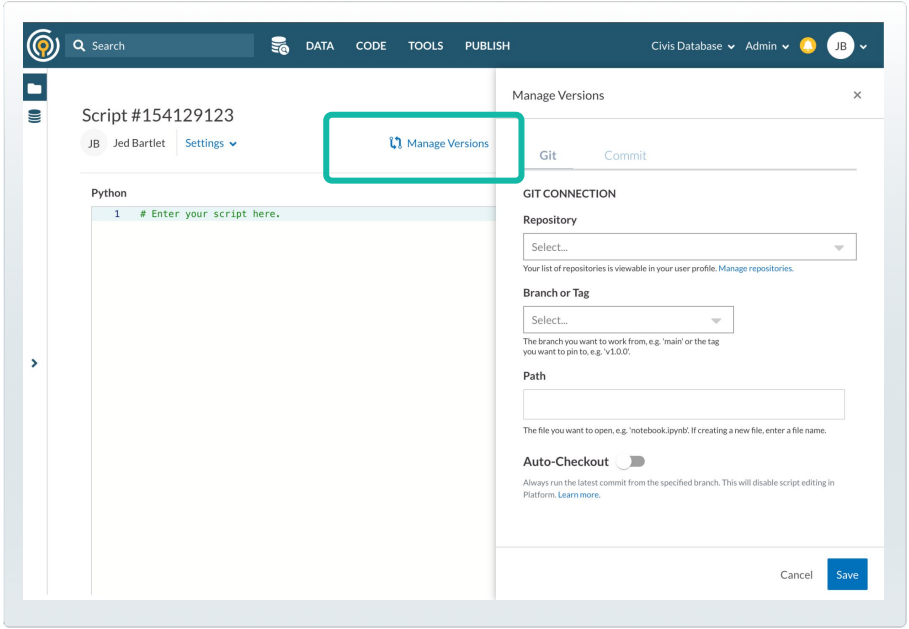


**Note:* *These steps apply to any existing or new SQL, R or Python script, workflow, HTML report, or stopped notebook.*

Version Control

Commit, Push, and Checkout

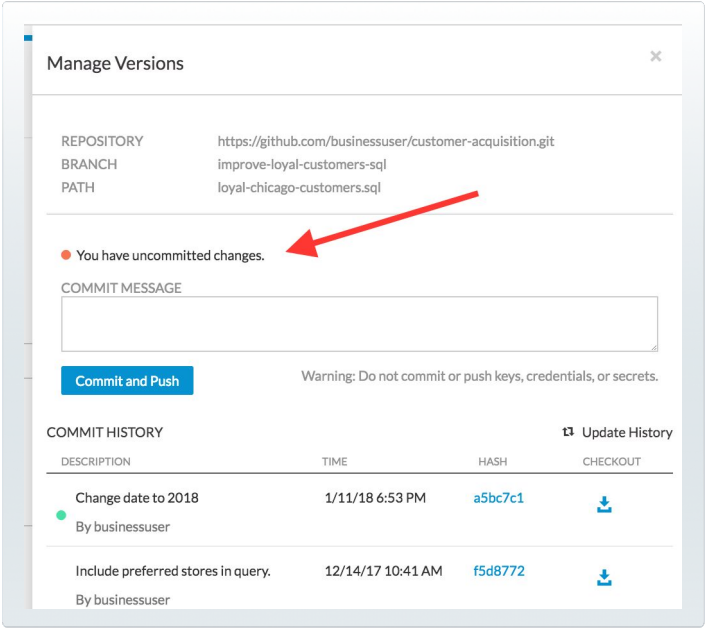
- After entering your repository's information, you can pull data from or push data to your repository through the Manage Versions pane
- While Platform automatically saves Platform objects, you will not see changes in Git until you manually commit and push these changes to Git repositories



Version Control

Commit, Push, and Checkout

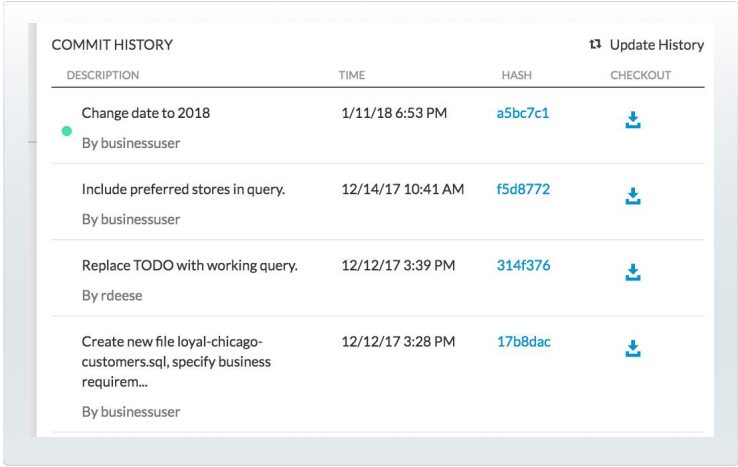
- As you make iterative edits to your script in Platform, you can commit changes to the specified path
- Once you make edits to the script, an icon will appear that prompts you to “Commit Changes”
- Navigate to the “commit” tab of the Manage Versions pane, add a commit message and press “commit and push”
- Platform will not automatically commit and push these changes



Version Control

Commit, Push, and Checkout

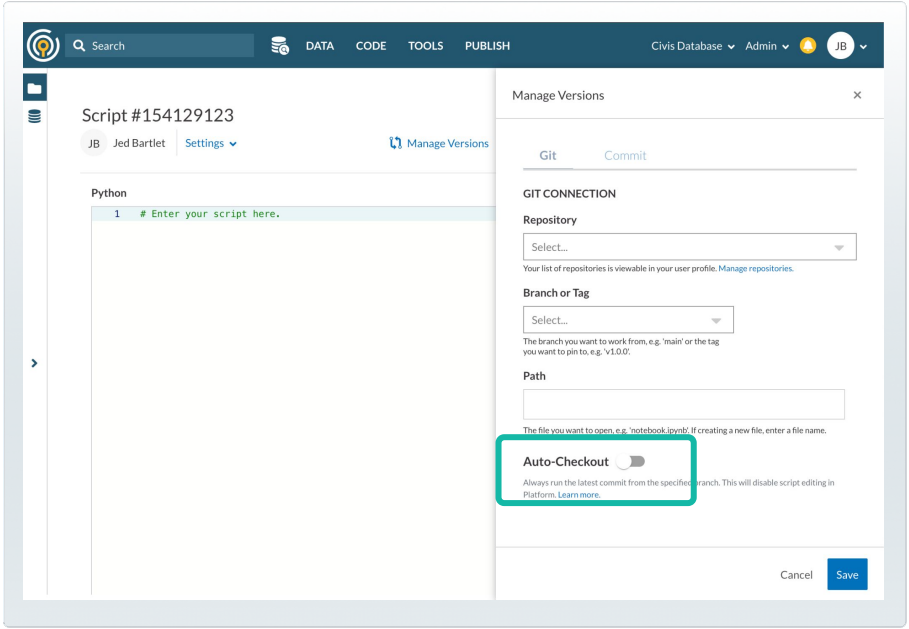
- Platform will automatically pull a log of commits where the connected file was changed in the specific repository and branch selected
- The green dot in the Git commit log indicates if the current code matches that commit
- You can checkout any commit shown in the log
    - Platform will warn you if checking out old code will overwrite any uncommitted changes



Version Control

Auto-checkout

- With Auto-Checkout Platform will always run the latest committed code from the specified branch of your repository.
- Editing script code in Platform will be disabled when Auto-Checkout is on to prevent conflicting changes
    - If you need to make changes to the script code, you can toggle Auto-Checkout off, make edits and commit the code.



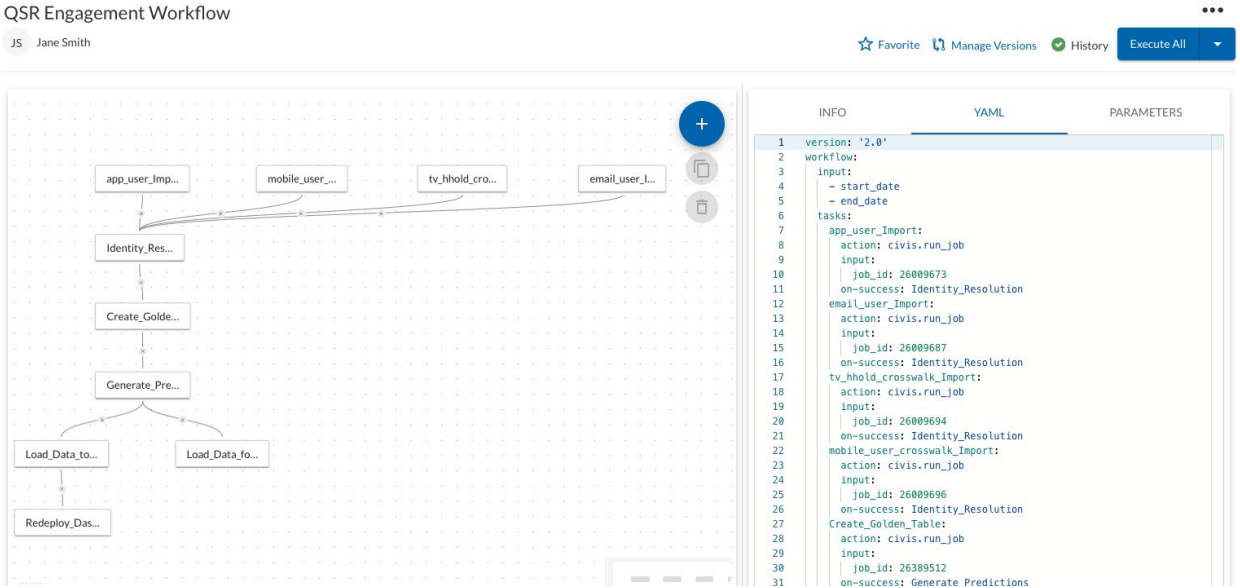
Workflows

Overview
Creating New Workflows
Parameters in Workflows
Executing Workflows

Workflows

Overview

Platform Workflows give users the ability to chain together jobs to accomplish a specific task.



More information can be found in Civis Help Documentation: [Workflow Basics]

Workflows

Overview

Graph and Info Tab

- Build Workflows using existing Jobs and Templates
- Create branching and basic task control flow
- Workflow Level Parameters for use with Templates

YAML

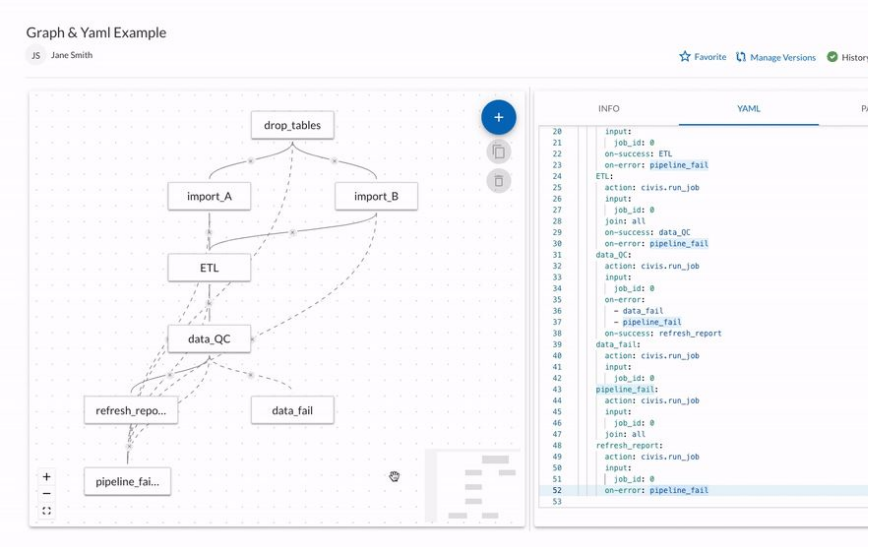
- Build Workflows using new or existing Jobs and Templates
- More advanced task control flow and conditional tasks
- Add retries to your Tasks
- Workflow level parameters can be used in new Jobs and with Templates
- Execute existing Workflow as a Workflow task
- And more

More information can be found in Civis Help Documentation: Advanced Workflows Features and Examples

Workflows

Overview

- The Graph is a visual representation of your workflow
- Changes made in the Graph UI or in the YAML will automatically update the other to reflect your changes.
- The workflow runs from top to bottom, traveling along the links between tasks
- Platform Workflows are built on the open source Mistral workflow engine
- You can create/edit workflows using the graph on the left and the INFO tab or via the Mistral YAML-based workflow language in the expandable YAML tab



Workflows

Overview

Workflow Control Flow

- On-success
  - Next task(s) will run after this task completes successfully
- On-complete
  - Next taks(s) will run after this task completes regardless of state
- On-error
  - Next task(s) will run after this task completes unsuccessfully
- Join
  - Determines if multiple parent tasks must complete before child task runs
  - Workflows created via UI defaults to a Join: All on all child tasks
  - Without Join: All the child task may run more than once

Workflows

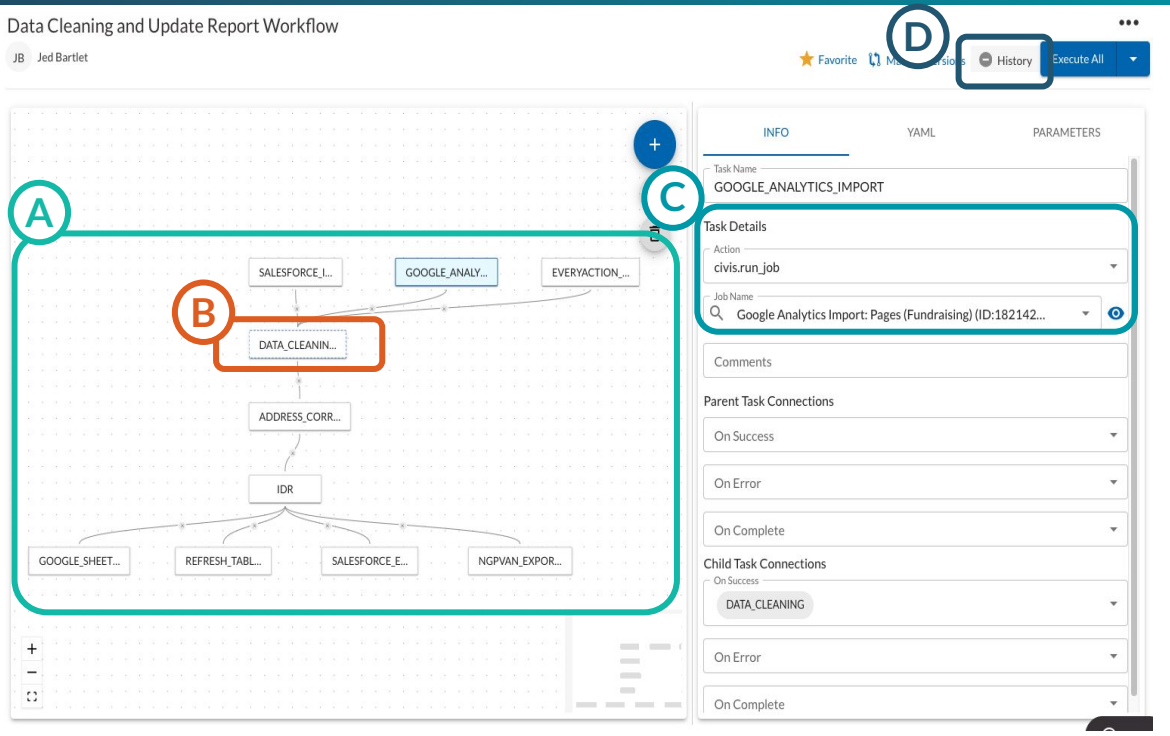
Graph View Terms to Know

A. Workflow
a collection of related jobs intended to be run with specific dependencies on each other.

B. Task
a unit of work inside of a workflow.

C. Action
the job that will be run at a particular moment during a workflow.

D. Execution
a single run of a workflow.



Workflows

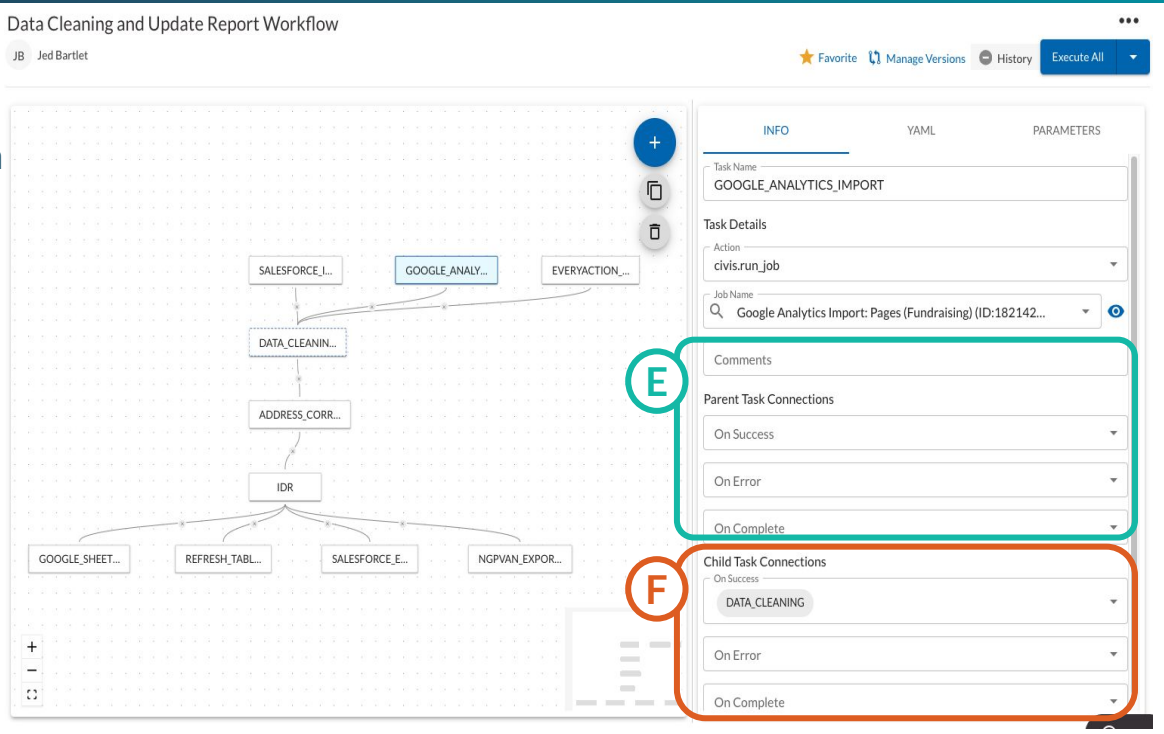
Graph View Terms to Know

E. Parent Task(s)

The task(s) that the currently selected task will run after when the run condition is met

F. Child Task(s)

The task(s) that will run after the currently selected task when the run condition is met



Workflows

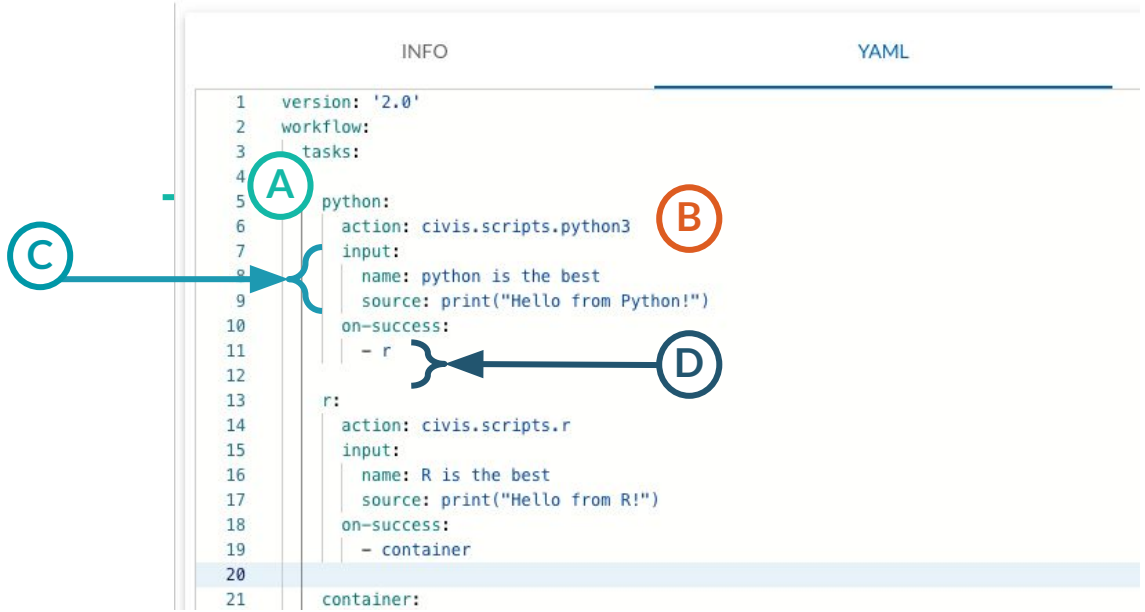
YAML Terms to Know

A. Task Name
The name of the task,
for example, “Python”

B. Task Action
API Call Endpoint

C. Task Input
Information that are passed into the API endpoint called

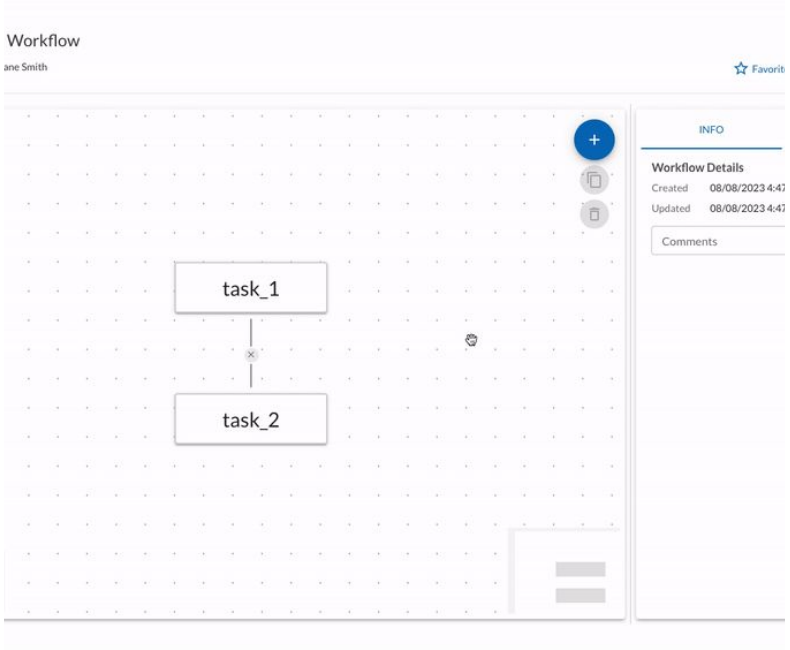
D. Child Task(s)
Names of the tasks to run once current task completed, if the run condition is met



Workflows

Creating a New Workflow: INFO Tab

- Click the + sign in the upper right, if you have a task selected already the new task will become a child
- Find existing Jobs using the Job Name Field
- Delete a task by clicking the trash can icon
- Click and drag one of the blue dots from one task to another task to chain the two together
- To unchain two jobs, click the X on the line connecting the task



Workflows

Creating a New Workflow: YAML Tab

- The YAML tab shows the workflow definition in YAML form
- The workflow contains a list of tasks, each with a unique name
- Each task specifies an action (e.g. running a job), an input list (e.g. the ID of the job to run), and an optional list of other tasks that should be run immediately following the success, completion, or failure of this task (on-success, on-complete, on-error)

Workflows

Creating New Workflow: YAML Tab

- Users can also refer to existing Platform jobs and workflows in the Mistral workflow language
  - Existing jobs use the `civis.run_job task action`
  - This action takes a single input: `job_id`.
- Existing workflows can be executed within another workflow
  - Use the `civis.workflow.execute` task action to execute
  - This action takes in a single input: `workflow_id`.

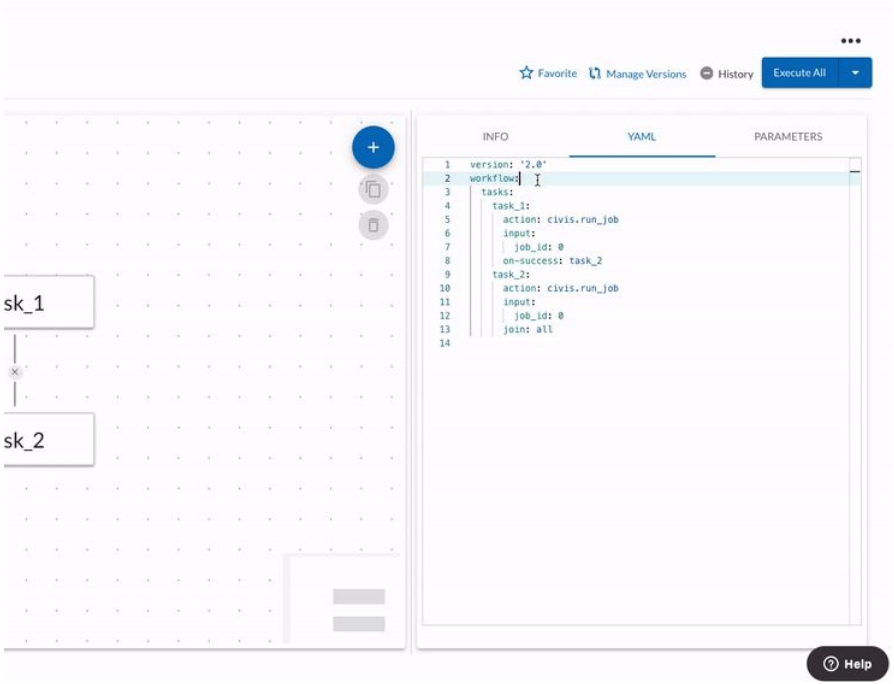
```
tasks:
 task_a:
  action: civis.run_job
  input:
    job_id: 34
```

```
tasks:
 execute_subworkflow:
   action: civis.workflows.execute
    input:
     workflow_id: 56
```

Workflows

Parameters in Workflows

- Use the Parameters tab, to create new parameters or see parameters declared in YAML
- In the input fields that appear, enter values for your parameters
- You can supply default value in the Parameter tab or YAML, it will appear pre-populated in the input field
- Each parameter input will be interpreted as a string
- Parameters are accessed via YAQL like this `<% $.parameter_name %>`

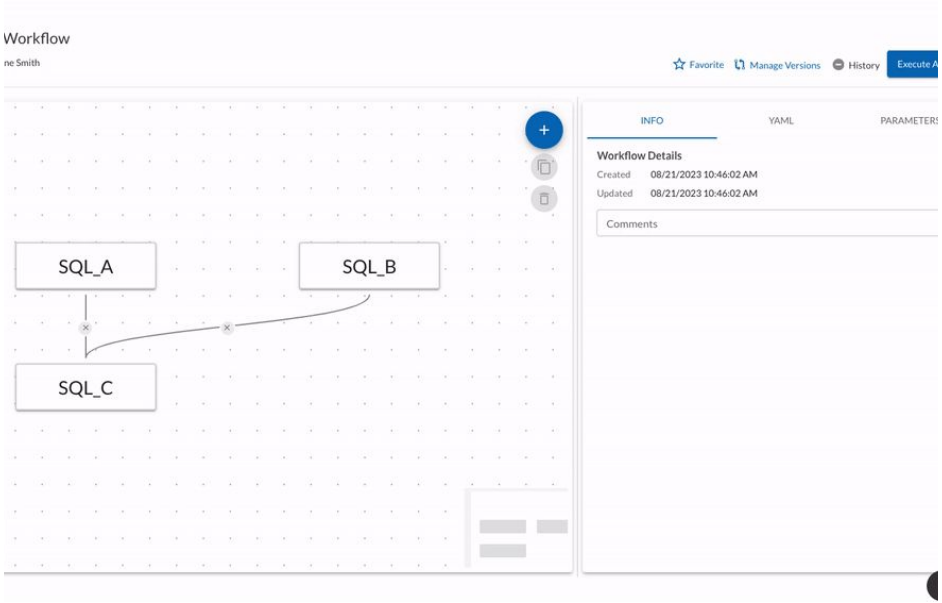


More information can be found in Civis Help Documentation: [Advanced Workflows Features and Examples]

Workflows

Executing Workflows

- To run a workflow, click “Execute (All Tasks)”
  - If Execute is grayed out, there is a problem with the workflow definition
- Once running, the workflow graph will display the execution and highlight when tasks have run successfully or failed
- You will be able to see the Job logs of a selected task in the INFO tab

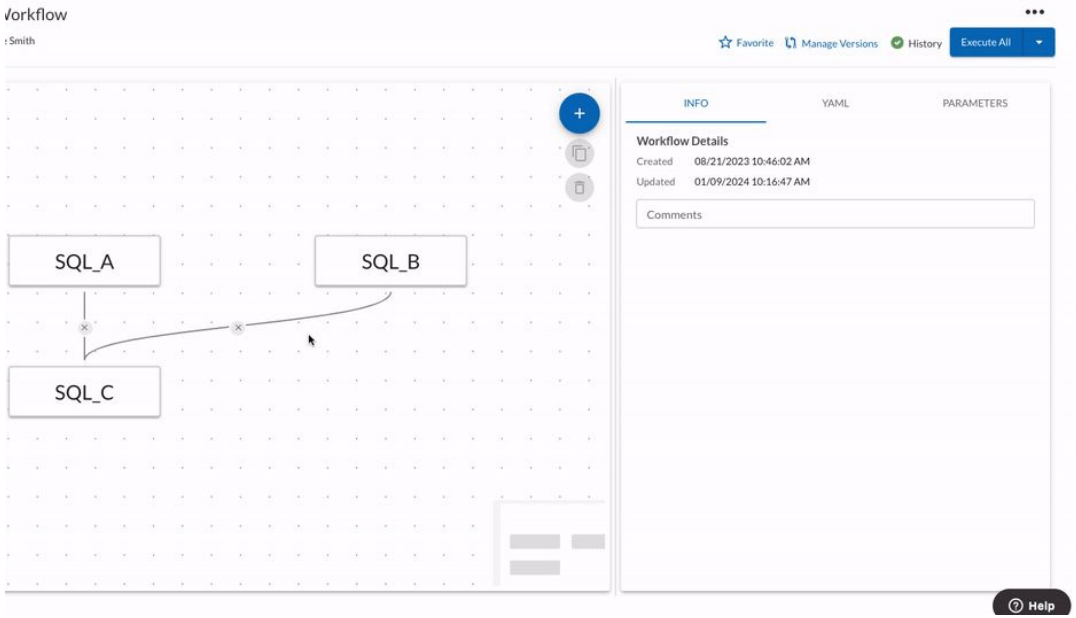


More information can be found in Civis Help Documentation: [Executing, Automating, and Debugging Workflows]

Workflows

Executing Workflows: Partial Executions

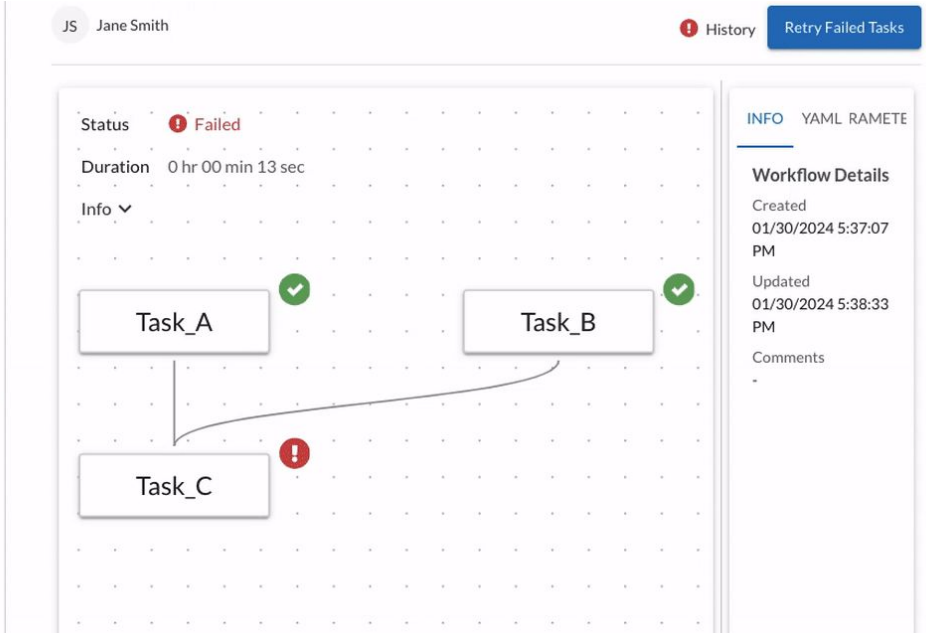
- To execute only some tasks click a task and then hover over other tasks so that the check bock to their right appears and select that box to include that task
- When you have the tasks you wish to execute selected click the “Execute Selected Tasks” button
- When the execution loads tasks not included will be greyed out



Workflows

Executing Workflows

- If your Workflow fails you can use the re-try failed tasks button. Your workflow will retry any failed tasks and resume execution
- Use “History” to see the status of a Workflow’s previous executions
- Clicking on any of these executions displays a more detailed view in the main window, where you can see the graph of tasks and the Workflow definition that the execution was run with



Thank you!

Reach out to **Support@CivisAnalytics.com** with any questions!

Our next webinar will be with our Product Team on new features and updates to Platform. Watch for the sign up form!