PharmaSUG 2025 - Paper HT-397

Trying Out Positron: New IDE for Statistical Programming

Phil Bowsher, Posit/RStudio PBC

Abstract

This paper will introduce a Pharmaverse workflow in Positron, an extensible, polyglot tool for writing code and exploring data, currently in Public Beta. The main Positron site for reference is: https://positron.posit.co/

Many pharmaceutical companies are shifting to an open-source backbone in Clinical Trials. Usually this effort includes creating a Next-Gen Statistical Computing Environments (SCE) where the focus is on open source languages for clinical reporting. A recent webinar by James Black discusses "The importance of the SCE in enabling our shift to open-source data science": https://pos.it/enable-oss

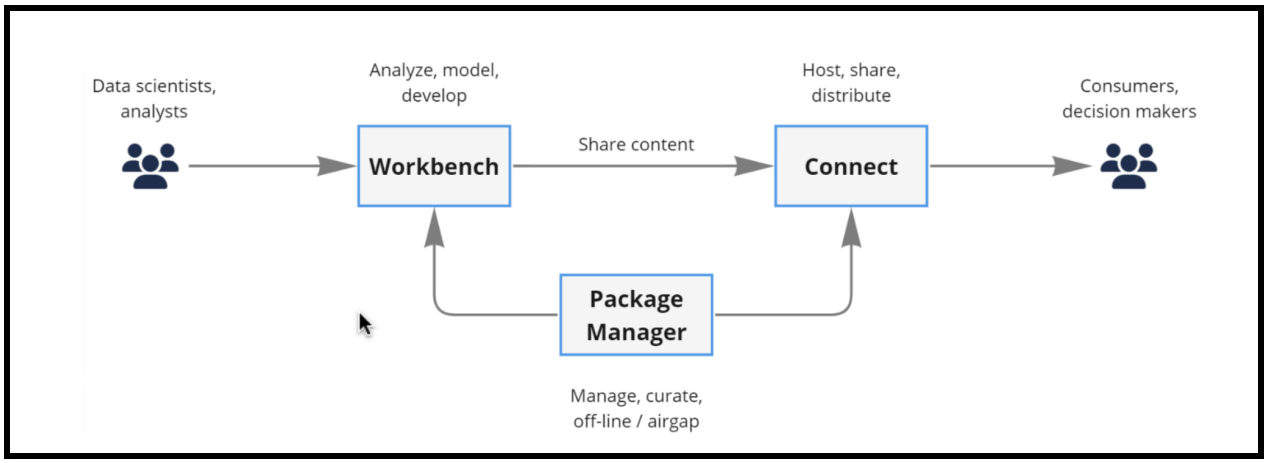
James highlights the use of R and Python programming in the SCEs to support new hires as well as use-cases like data science and Real-World Evidence. Moreover, many pharmaceutical companies are updating data environments to modern platforms like Databricks, Snowflake, etc., where workflows are often Python-based. This paper will highlight a Pharmaverse workflow in Positron within such an environment!

Introduction

You can find all of the examples in this paper here: https://github.com/philbowsher/Positron-Overview-for-Pharma

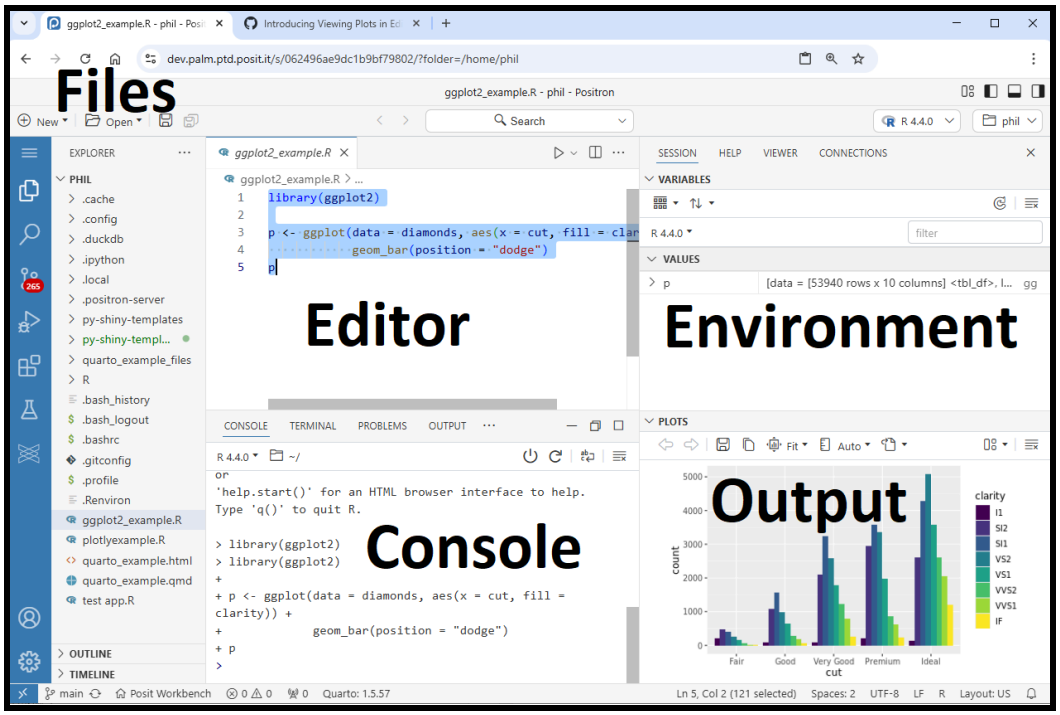
Positron is currently in Public Beta and is available for free on desktop here: https://positron.posit.co/download

Since most Statistical Programmers use Positron via a controlled server based environment, the focus of this paper will be on Positron via *Posit Workbench*. Posit Workbench is the programming developer platform part of Posit Team as seen below:



Posit Workbench installation is here: https://docs.posit.co/ide/server-pro/getting_started/installation/installation.html

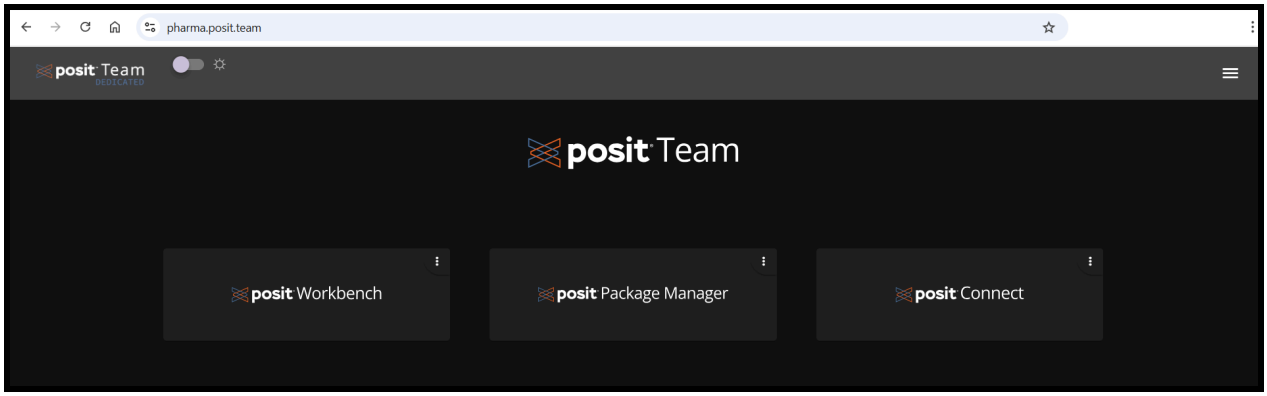
Many users are familiar with the 4-pane data science experience via the RStudio IDE. Positron has this layout as seen below:



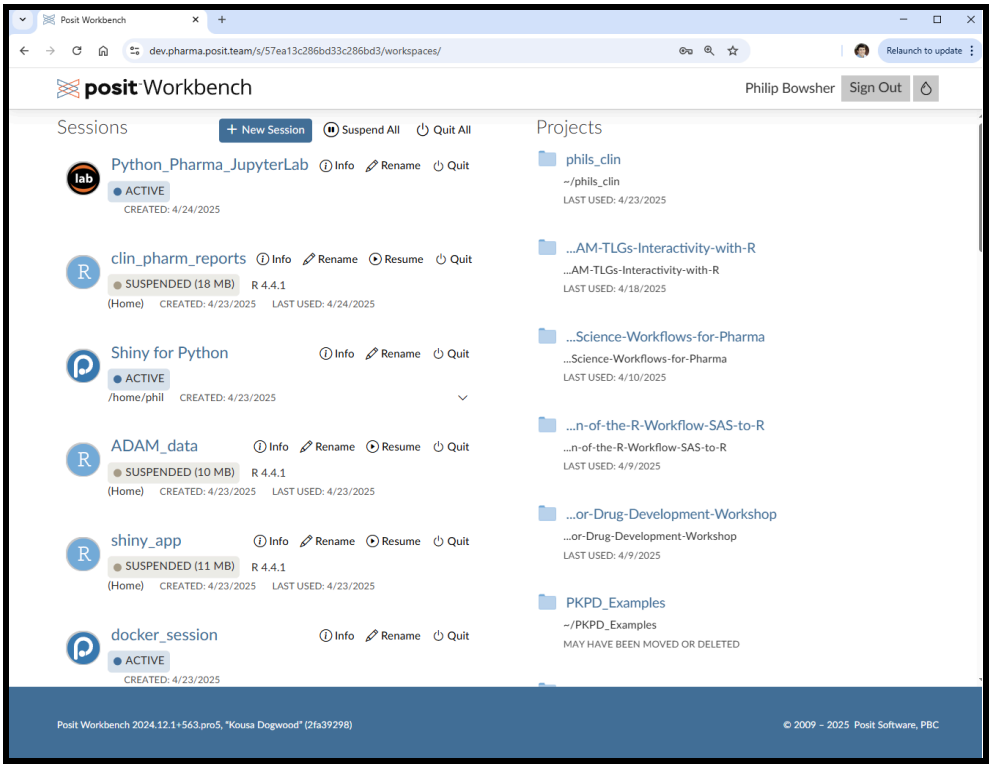
Let's get started!

Positron in Posit Workbench

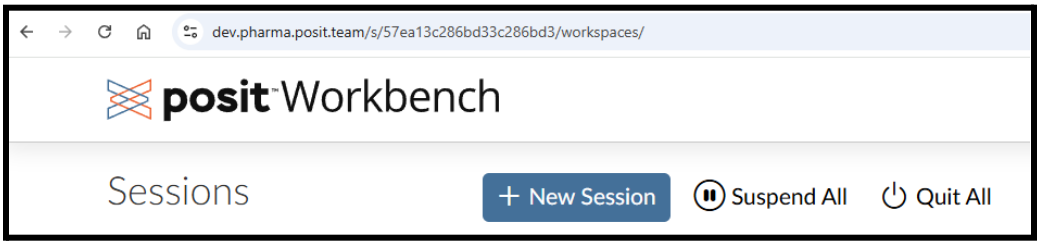
At your organization, they will likely give you access to Posit Team with Workbench, which looks like:



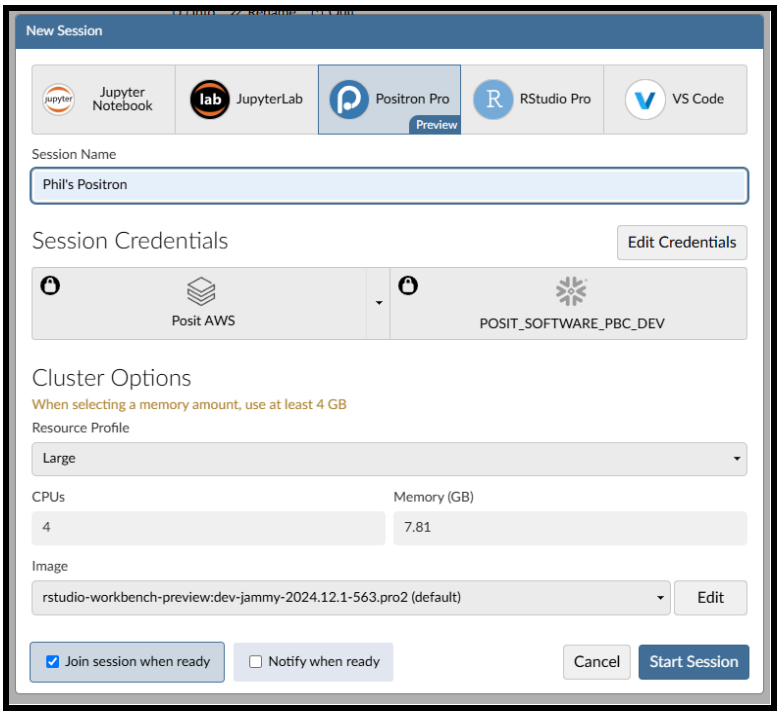
Positron is an Integrated development environment (IDE) within Posit Workbench. Posit Workbench provides many IDEs. From here, click Posit Workbench which will take you to the following page:



To open Positron, click “+ New Session” like:

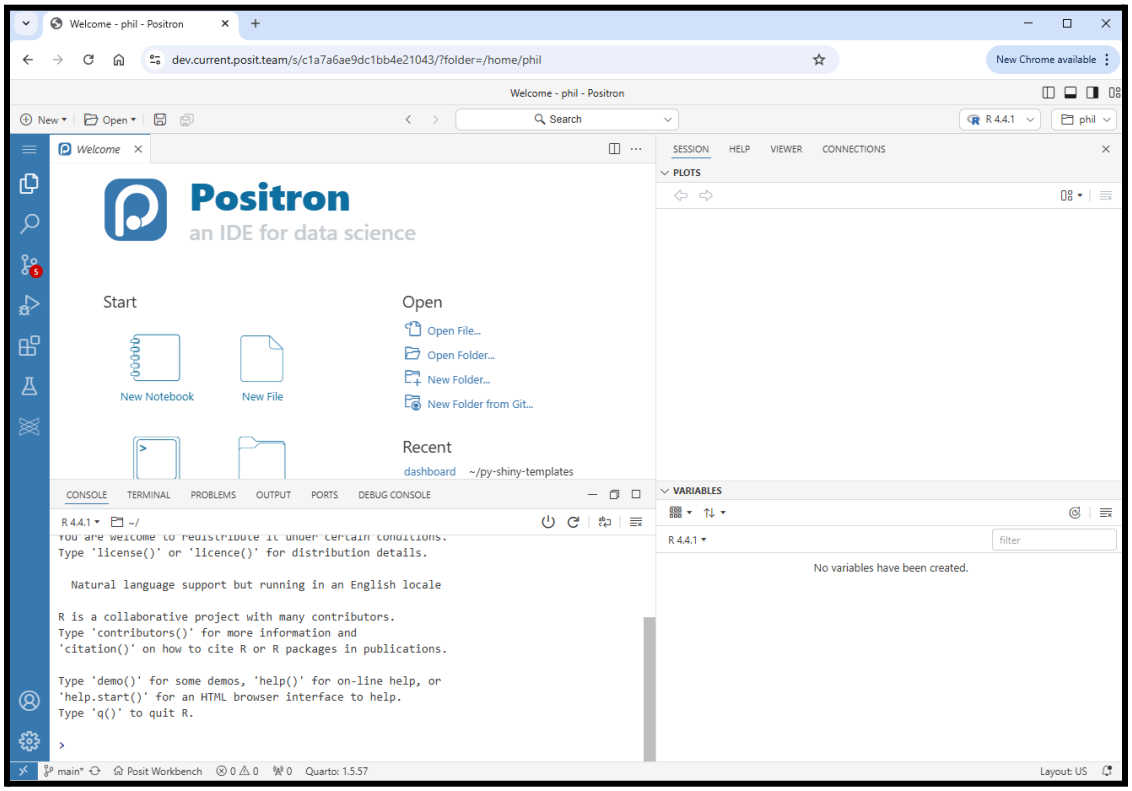


This will open the **New Session** dialog box:



Select the preferred resources. Please note that in this example, the Positron user IDE session will be generated from a Docker Image but not all Posit Workbench installations require or use Docker. Users can pick the preferred Docker Image as needed. There are three options for using Docker images as explained here: https://docs.posit.co/ide/server-pro/job_launcher/using_docker_images.html

When ready, click Start Session. This will spawn a Session in the organization's cloud environment (AWS, Azure etc.). Below is an image of Positron accessed in Posit Workbench via a browser:



Pharmaverse Workflow in Positron

Below we will complete a simple Pharmaverse workflow in Positron using the Pharmaverse Examples here:

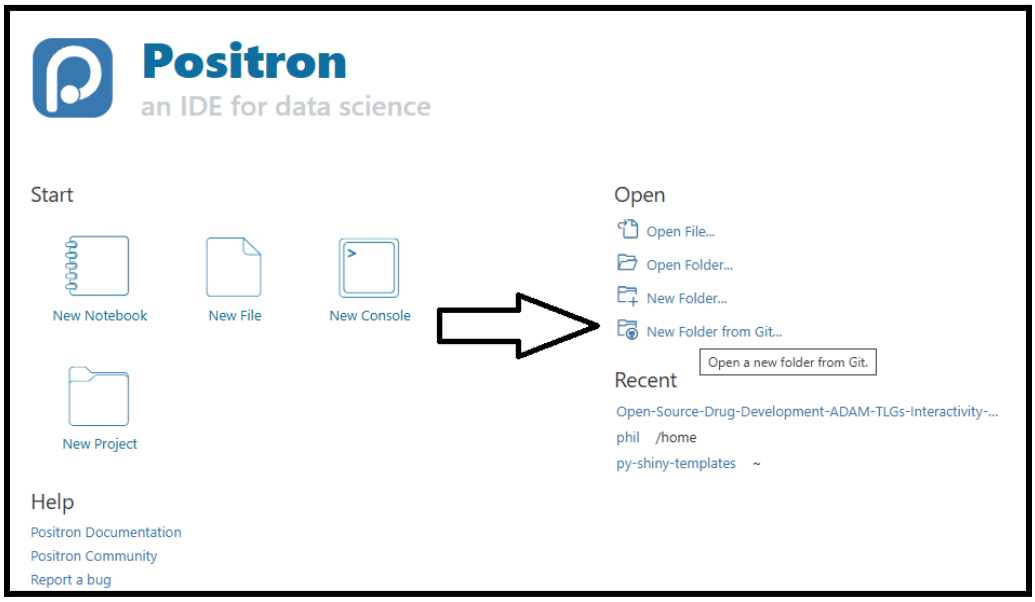
https://pharmaverse.github.io/examples/

In this example, we will use *Admiral* to create the ADaM Subject-level Analysis (**adsl)** and Adverse Events (**adae)** datasets. We will follow the steps here:

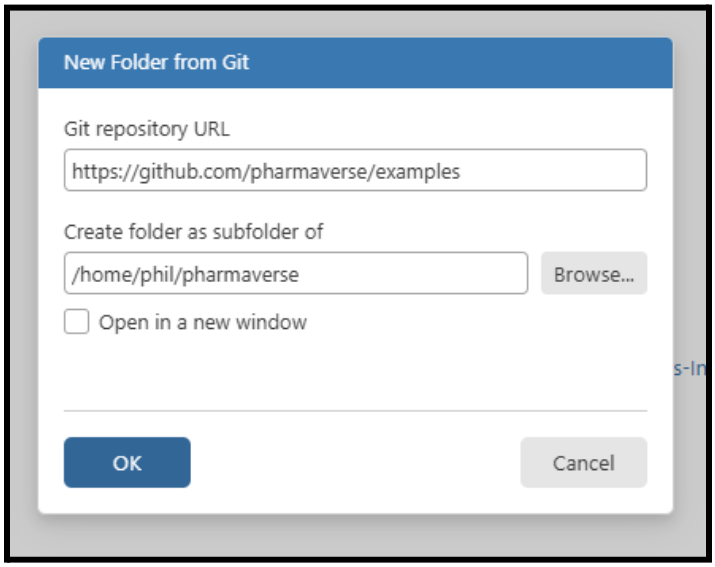
https://pharmaverse.github.io/examples/adam/adsl.html

https://pharmaverse.github.io/examples/adam/adae.html

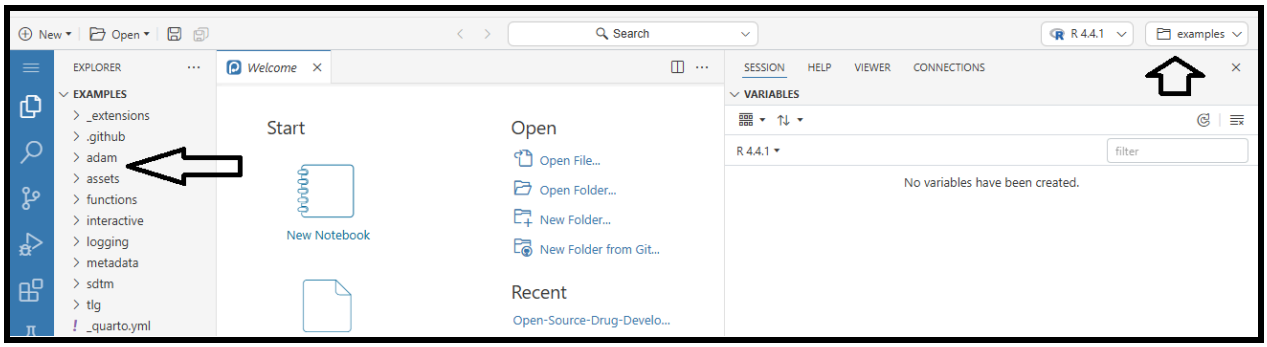
To start, clone the repository from Github in Positron using the following:



The following dialog box will open:

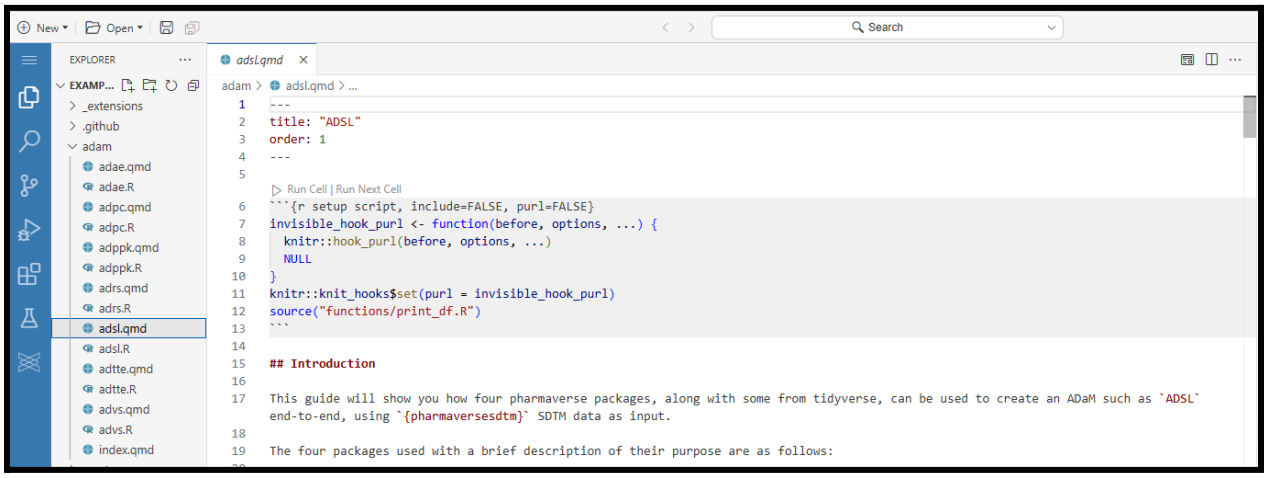


Positron will refresh and now the IDE will be operating within the "examples" folder. Open the “adam” folder as seen here:

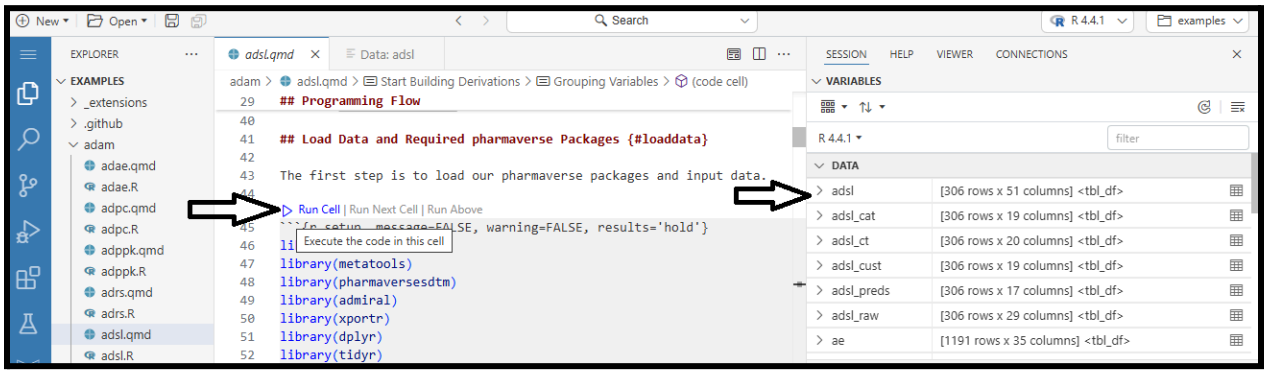


We will run a *Quarto* file. *Quarto* is a next-generation version of *R Markdown*. Positron provides world class support for *Quarto* and users can read more at: https://quarto.org/

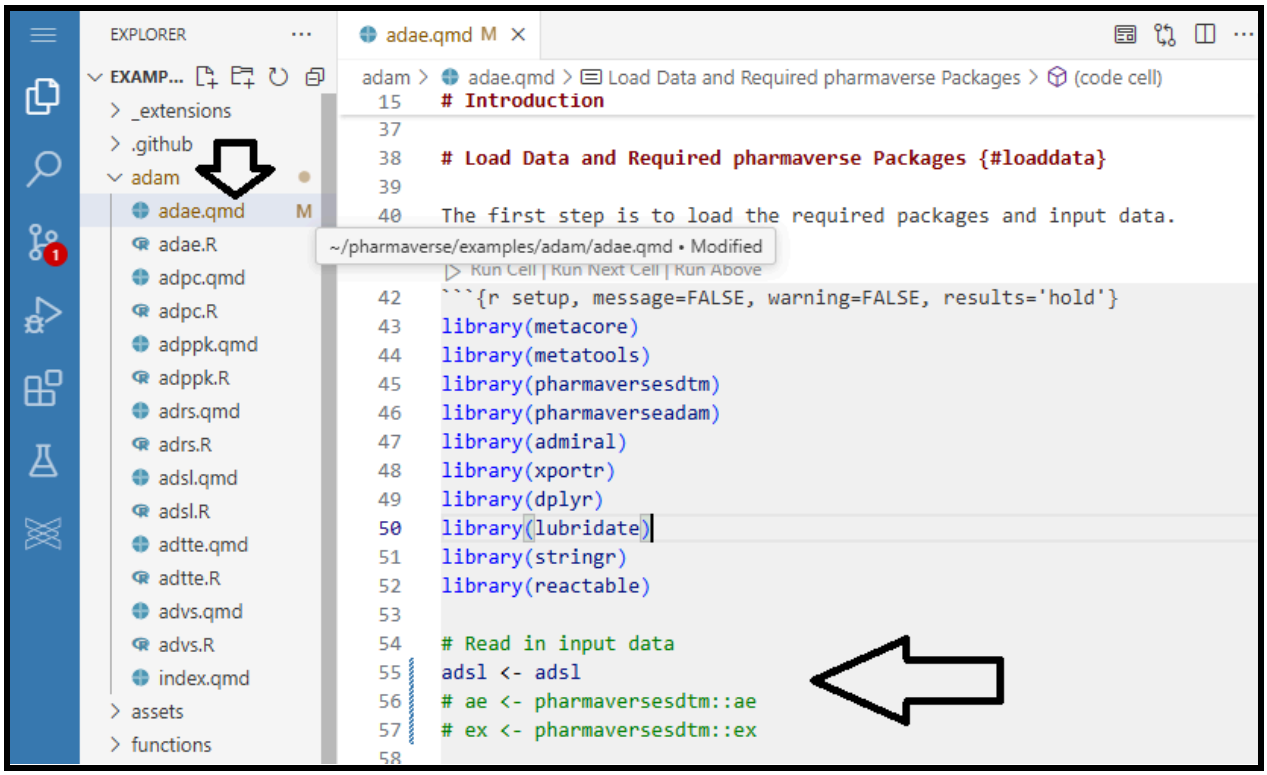
Open the *adsl.qmd* file as below:



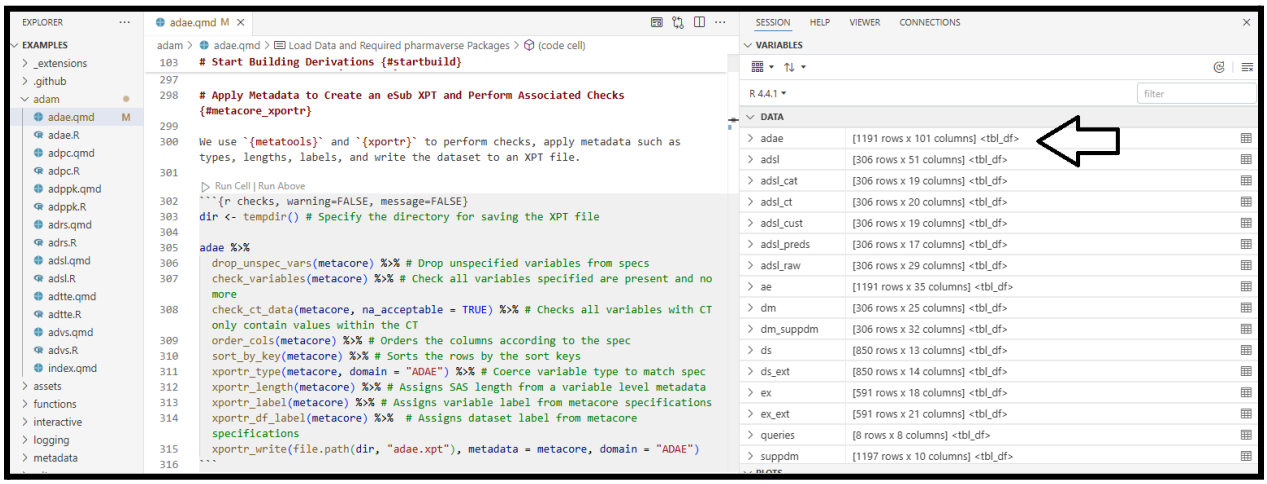
Now we will run the *Quarto* code chunks using the “Run/Play” button. Continue running the chunks all the way through the *Quarto* document to generate the **adsl** dataset.



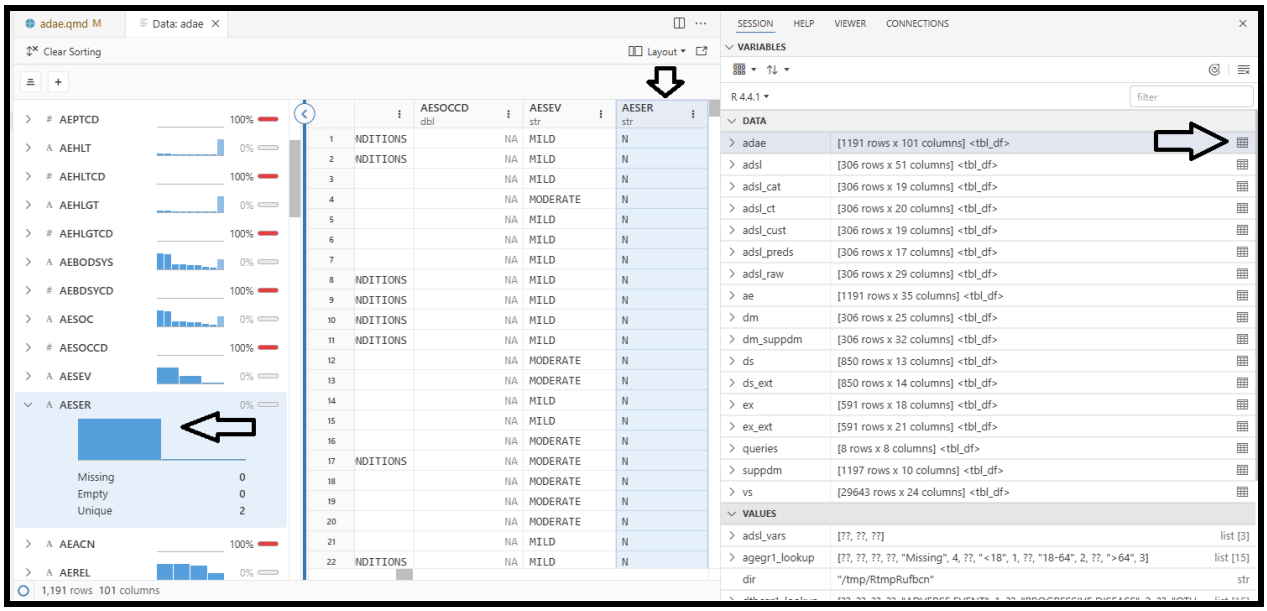
Now that **adsl** has been made with the *Admiral* package, we will run the adae.qmd file. Open the adae.qmd file by clicking on it. If preferred, users can use the **adsl**, **ae**, and **ex** datasets from the *pharmaverseadam* R package. Since we made those in the steps above, we can comment that out and use the created files as seen here:



As before, run the chunks in the *Quarto* document to create our final **adae** dataset:



We will review the **adae** dataset by clicking on the the dataset icon on the right side of name as in:



Positron provides an advanced Data Explorer (https://positron.posit.co/data-explorer.html) for users to do code-first exploration of data. R users can run the familiar View() to see data which now has included features for exploring and filtering data. The Data Explorer helps to review the columns as well as produce some visual analytics.

Using GenAI in Positron

Now that we have created the **adae** dataset, we will use generative artificial intelligence (GenAI) to create an app with *Shiny for R*.

Below we will create the Shiny app with the *ellmer* R package. In addition to ellmer, the Positron team is building a native *Positron Assistant* to provide LLM integration within Positron, both for chat and for inline completions. It fulfills a role similar to Github Copilot. A video of the Positron Assistant can be seen here: https://posit.co/use-cases/ai/.

Positron Assistant is not yet available for users, but major progress is already underway, and announcements for this integration will be available here: https://positron.posit.co/

Below we will use *ellmer*: https://github.com/tidyverse/ellmer

A similar tool for Python users is chatlas: https://posit-dev.github.io/chatlas/

In addition to *ellmer* and the Positron Assistant, Positron is also compatible with GenAI extensions on OpenVSX such as Google Gemini, AWS Q, and Continue.dev.

Let's get started creating a Shiny app in Positron with GenAI!

Using GenAI in Positron to Create a Shiny App

First, we will specify the provider/model choice in Positron. Users will need to obtain an API key from the model organization and then store that securely in the environment. Posit PBC recommends saving an environment variable rather than using the API key directly in users code. In R, users can use the R package *usethis*, to set up the API keys in Positron. Using *usethis*, add your API key(s) to your .Renviron file. You can open your .Renviron for editing with **usethis::edit_r_environ()**. Add the API key(s) to your .Renviron, for example:

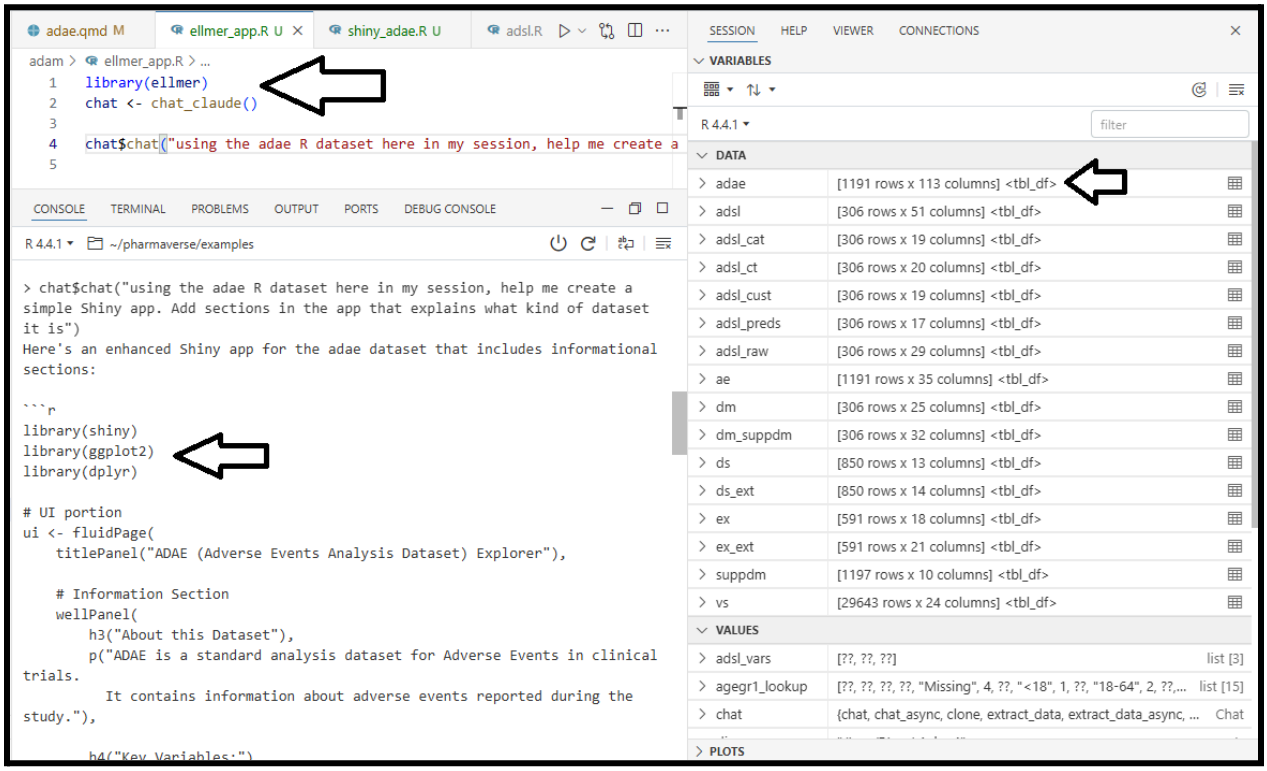
OPENAI_API_KEY=my-api-key-openai-xyz
ANTHROPIC_API_KEY=api-key-anthropic-xyz
GOOGLE_API_KEY=api-key-google-xyz

Now we are ready to use ellmer to create a Shiny app. Below we will use the Anthropic Claude model.

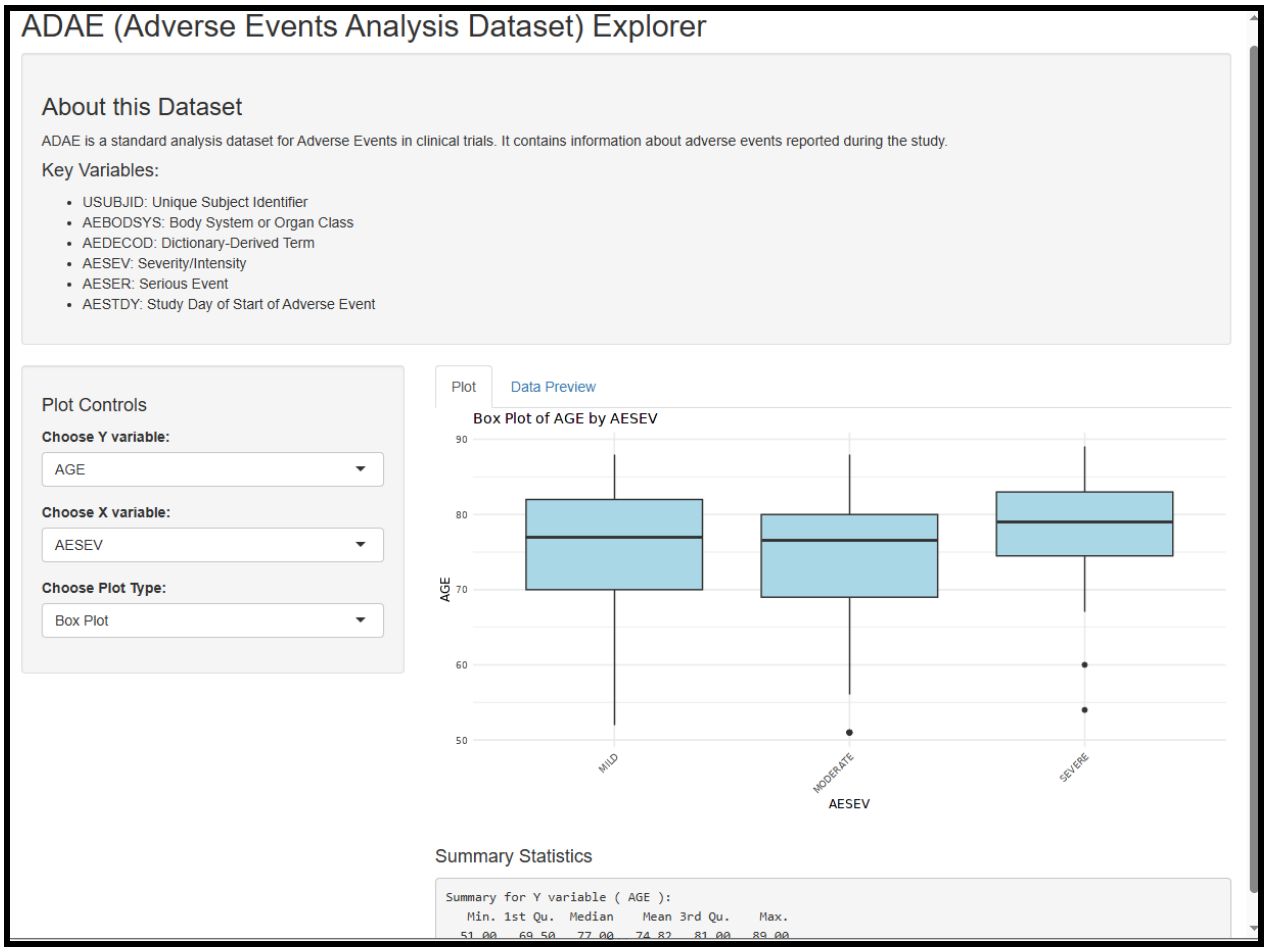
After securely connecting to the Anthropic Claude model, each connection will start by creating a new chat object like as below in R:

```
library(ellmer)

chat <- chat_claude()

chat$chat("using the adae R dataset here in my session, help me create a simple Shiny app. Add sections in the app that explains what kind of dataset it is")
```

This code will be executed in Positron as below



This will generate the following Shiny app!



Conclusion & Contact

The information above highlights the exciting new IDE, Positron and an example workflow using Pharmaverse and GenAI to create a Shiny app on **adae** data. Many organizations are moving to Open source languages like R and Python for clinical reporting and Positron can help modernize clinical processes with new polyglot features and GenAI capabilities.

Phil Bowsher

phil@posit.co

https://github.com/philbowsher