



databricks

DATA+AI SUMMIT



Forward-looking Statement

This presentation has been prepared for informational purposes only. The information set forth herein does not purport to be complete or contain all relevant information. Statements contained herein are made as of the date of this presentation unless stated otherwise.

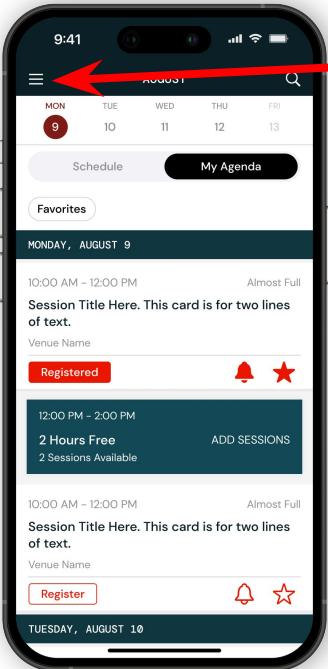
This presentation and the accompanying oral commentary may contain forward-looking statements. In some cases, forward-looking statements can be identified by terms such as "may", "will", "should", "expects", "plans", "anticipates", "could", "intends", "projects", "believes", "estimates", "predicts", or "continue", or the negative of these words or other similar terms or expressions that concern Databricks' expectations, strategy, plans, or intentions.

Forward-looking statements are based on information available at the time those statements are made and are inherently subject to risks and uncertainties that could cause actual results to differ materially from those expressed in or suggested by the forward-looking statements.

Forward-looking statements should not be read as a guarantee of future performance or outcomes. Except as required by law, Databricks does not undertake any obligation to publicly update or revise any forward-looking statement, whether as a result of new information, future developments or otherwise.

Complete Your Surveys

Your feedback has a direct impact on Data + AI Summit content



- You will receive a survey for each session attended
- Open the Databricks Events app and select “My Surveys” from the menu
- Surveys can also be submitted in the Attendee Portal

Next-Gen Data Science

How Posit and Databricks Are
Transforming Analytics at Scale

James Blair
Date



Agenda

Posit and Databricks
Better Together

Next-Gen Data Science
Introduction to Positron

GenAI Powered Applications
Natural language dashboarding

App Deployment
Share securely

Conclusion
Only the beginning

Posit and Databricks



Built to last

Proud to be a Certified B Corporation®.

Our mission is **free and open-source software** for data science, scientific research & technical communication for the public good.

Our corporate charter requires that company decisions **balance the interests** of community, customers, employees, & shareholders.

To ensure we can be a trustworthy partner and support our mission long-term, Posit **will remain independent and privately held**.



Certified

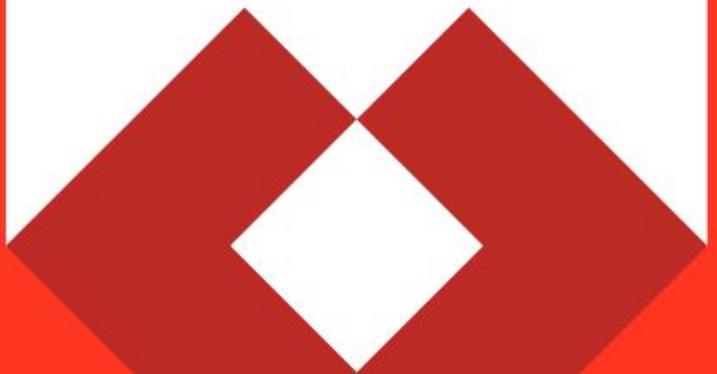


Corporation

Winner

2025 Databricks Partner Awards

**Developer Tools
Partner of the
Year**



posit®

ISV Partner Awards



tidytuesday/data/2020/2020-04-07/readme.md

Tour de France

The Tour de France is an annual men's multiple stage bicycle race primarily held in France, while also occasionally passing through nearby countries. Like the other Grand Tours (the Giro d'Italia and the Vuelta a España), it consists of 21 day-long stages over the course of 23 days. It has been described as "the world's most prestigious and most difficult bicycle race".

The data this week comes from [Alastair Rushworth's Data Package tdf](#) and [Kaggle](#).

Alastair has a very nice walkthrough of his data package at his [blog](#)!

I've added the Kaggle data which goes through 2017 for some additional stage-specific data not captured in his dataset. Please note that for the most part these datasets COULD be joined by year/edition.

Some other stats and records can be found on [Wikipedia](#).

Get the data here

```
# Get the Data  
  
tdf_winners <- read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/main/data/2020/2020-04-07/tdf.csv')  
  
# Or read in with tidyTuesdayR package (https://github.com/dslic-io/tidyTuesdayR)  
# PLEASE NOTE TO USE 2020 DATA YOU NEED TO USE tidyTuesdayR version ? from GitHub  
  
# Either ISO-8601 date or year/week works!  
  
# Install via pak::pak("dslic-io/tidyTuesdayR")  
  
tuesdata <- tidyTuesdayR::tt_load('2020-04-07')  
tuesdata <- tidyTuesdayR::tt_load(2020, week = 15)  
  
tdf_winners <- tuesdata$tdf_winners
```

Positron

A Next-Gen Data Science IDE



Posit Workbench x +

dev.databricks.posit.team/s/57ea13c286bd33c286bd3/workspaces/ ☆ F₂ ...

posit Workbench

James Blair Sign Out ?

Sessions

Positron Pro Session ACTIVE /home/james CREATED: 5/29/2021

New Session

Jupyter Notebook JupyterLab Positron Pro RStudio Pro VS Code

Session Name DAIS Demo

Session Credentials Edit Credentials

Azure Databricks Workspace

Cluster Options

When selecting a memory amount, use at least 4 GB

Resource Profile Large

CPU Memory (GB)

4 7.81

Image rstudio-workbench:ubuntu2204-2025.05.0 (default) Edit

Join session when ready Notify when ready Cancel Start Session

Posit Workbench 2025.05.0+496.pro5, "Mariposa Orchid" (29bcldf7) © 2009 - 2025 Posit Software, PBC

Welcome — Positron

dev.databricks.posit.team/s/e29766ae9dc1b109afa21/

Welcome — Positron

+ New Open ... Search SESSION CONNECTIONS HELP VIEWER ... X

P **Welcome** X

Positron
an IDE for data science

Start

New Notebook New File New Console

Open

Open File... Open Folder... New Folder... New Folder from Git...

Recent

CONSOLE TERMINAL PROBLEMS OUTPUT PORTS DEBUG CONSOLE

Python 3.12.6 (Global) started.
Python 3.12.6 (main, Apr 22 2025, 14:51:42) [GCC 11.4.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.12.3 -- An enhanced Interactive Python. Type '?' for help.
>>>

VARIABLES

Python 3.12.6 (Global) filter No variables have been created.

PLOTS

R 4.4.1 Python 3.12....

Layout: U.S.

eda.qmd — eda — Positron

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

eda.qmd — eda — Positron

CHAT

Let me check if the plot was created:

- ✓ Using "Get active plot"...

The histogram has been created successfully. It shows the distribution of elapsed times for the Tour de France stages. From the plot, we can observe that:

1. The distribution is heavily right-skewed
2. Most elapsed times are clustered in the lower range (close to 0)
3. There are some outliers with much longer elapsed times (extending out to around 500000)

The current scale might make it a bit hard to see the details in the main cluster of times. Would you like me to modify the plot to better visualize the distribution, perhaps by:

1. Using a log scale
2. Focusing on the main cluster of times
3. Converting the elapsed time to a more interpretable unit (like hours or minutes)

Just let me know if you'd like any of these adjustments to be made to the plot.

analyze the relationship between elapsed time and stage distance using a scatter plot

investigate outliers in elapsed time - what stages had unusually long times?

convert elapsed time to hours and create a more readable histogram

eda.qmd Current file

SESSION CONNECTIONS HELP VIEWER ...

VARIABLES

Python 3.11.12 (Uv: .venv)

DATA

elapsed_times pandas.Series [63913.0, 63968.0, 66012.0, ...]

VALUES

con <ibis.backends.databricks.BackendBackend

stages ibis.expr.types.relations.Table ibis.Expr

w WorkspaceClient(host='ht...') WorkspaceClient

FUNCTIONS

load_dotenv load_dotenv(dotenv_path: Union[str, Path])

CLASSES

WorkspaceClient <class 'databricks.sdk.WorkspaceClien... type

PLOTS

Distribution of Stage Elapsed Times

Keep Undo | 1 of 1 ↑ ↓

CONSOLE TERMINAL PROBLEMS 2 OUTPUT PORTS ...

```
Python 3.11.12 (Uv: .venv) ~ /data-and-ai-summit-2025/eda
4 elapsed_times = stages.select('elapsed').execute()
>>> import matplotlib.pyplot as plt
...
... # Get the elapsed time data
... elapsed_times = stages.elapsed.execute()
...
... # Create the histogram
... plt.figure(figsize=(10, 6))
... plt.hist(elapsed_times, bins=30, edgecolor='black')
... plt.title('Distribution of Stage Elapsed Times')
... plt.xlabel('Elapsed Time')
... plt.ylabel('Frequency')
... plt.grid(True, alpha=0.3)
>>>
```

Ln 46, Col 1 Spaces: 4 UTF-8 LF (Quarto Layout: U.S.

eda.qmd — eda — Positron

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

eda.qmd — eda — Positron

CHAT

Let me check if the plot was created:

✓ Using "Get active plot..."

The histogram has been created successfully. It shows the distribution of elapsed times for the Tour de France stages. From the plot, we can observe that:

1. The distribution is heavily right-skewed
2. Most elapsed times are clustered in the lower range (close to 0)
3. There are some outliers with much longer elapsed times (extending out to around 500000)

The current scale might make it a bit hard to see the details in the main cluster of times. Would you like me to modify the plot to better visualize the distribution, perhaps by:

1. Using a log scale
2. Focusing on the main cluster of times
3. Converting the elapsed time to a more interpretable unit (like hours or minutes)

Just let me know if you'd like any of these adjustments.

analyze the relationship between elapsed time and stage distance using a scatter plot

investigate outliers in elapsed time - what stages had unusually long times?

convert elapsed time to hours and create a more readable histogram

eda.qmd Current file

Positron Assistant

Context aware GenAI coding assistance for Data Science

```
4 elapsed_times = stages.select('elapsed').execute()
>>> import matplotlib.pyplot as plt
...
... # Get the elapsed time data
... elapsed_times = stages.elapsed.execute()
...
... # Create the histogram
... plt.figure(figsize=(10, 6))
... plt.hist(elapsed_times, bins=30, edgecolor='black')
... plt.title('Distribution of Stage Elapsed Times')
... plt.xlabel('Elapsed Time')
... plt.ylabel('Frequency')
... plt.grid(True, alpha=0.3)
>>>
```

SESSION CONNECTIONS HELP VIEWER ...

VARIABLES

Python 3.11.12 (Uv:.venv) filter

DATA

> elapsed_times pandas.Series [63913.0, 63968.0, 66012.0, ...]

VALUES

> con <ibis.backends.databricks.Backend... Backend

stages ibis.expr.types.relations.Table ibis.Expr

> w WorkspaceClient(host='http://...', type=WorkspaceClient)

FUNCTIONS

load_dotenv load_dotenv(dotenv_path: Union[st... function

CLASSES

> WorkspaceClient <class 'databricks.sdk.WorkspaceClien... type

PLOTS

Distribution of Stage Elapsed Times

Keep Undo ⌘ | 1 of 1 ↑ ↓ ⌘

CONSOLE TERMINAL PROBLEMS 2 OUTPUT PORTS ...

Python 3.11.12 (Uv:.venv) ~ /data-and-ai-summit-2025/eda

Ln 46, Col 1 Spaces: 4 UTF-8 LF () Quarto Layout: U.S. ⌘

eda.qmd — eda — Positron

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

eda.qmd — eda — Positron

CHAT

Let me check if the plot was created:

- ✓ Using "Get active plot..."

The histogram has been created successfully. It shows the distribution of elapsed times for the Tour de France stages. From the plot, we can observe that:

1. The distribution is heavily right-skewed
2. Most elapsed times are clustered in the lower range (close to 0)
3. There are some outliers with much longer elapsed times (extending out to around 500000)

The current scale might make it a bit hard to see the details in the main cluster of times. Would you like me to modify the plot to better visualize the distribution, perhaps by:

1. Using a log scale
2. Focusing on the main cluster of times
3. Converting the elapsed time to a more interpretable unit (like hours or minutes)

Just let me know if you'd like any of these adjustments.

analyze the relationship between elapsed time and stage distance using a scatter plot

investigate outliers in elapsed time - what stages had unusually long times?

convert elapsed time to hours and create a more readable histogram

eda.qmd Current file

SESSION CONNECTIONS HELP VIEWER ...

VARIABLES

Python 3.11.12 (Uv: .venv)

DATA

elapsed_times pandas.Series [63913.0, 63968.0, 66012...]

VALUES

con <ibis.backends.databricks.Backend... Backend

stages ibis.expr.types.relations.Table ibis.Expr

w WorkspaceClient(host='htt... WorkspaceClient

FUNCTIONS

load_dotenv load_dotenv(dotenv_path: Union[st... function

CLASSES

WorkspaceClient <class 'databricks.sdk.WorkspaceClien... type

PLOTS

Distribution of Stage Elapsed Times

The screenshot shows a Databricks workspace interface. On the left, there's a sidebar with various icons and a list of notebooks. The main area has a code editor window titled 'eda.qmd' containing Python code for generating a histogram. Below the code editor is a red-outlined box containing the text 'Code Editor' and 'Full featured editor'. To the right of the code editor is a histogram titled 'Distribution of Stage Elapsed Times' with 'Elapsed Time' on the x-axis and 'Frequency' on the y-axis. The histogram shows a highly right-skewed distribution with most bars between 0 and 100,000. A legend indicates that blue bars represent the main cluster of times, while smaller grey bars represent outliers. The bottom status bar shows the line and column numbers (Ln 46, Col 1), spaces used (Spaces: 4), and other workspace details.

```
eda.qmd > (code cell)
41 ``{python}
43 ...
44 ⌂ Run Cell | Run Above
45 ``{python}

46 import matplotlib.pyplot as plt
47
48 plt.figure(figsize=(10, 6))
49 plt.hist(stages.elapsed.execute(), bins=50)
50 plt.xlabel('Elapsed Time')
51 plt.ylabel('Frequency')
52 plt.title('Distribution of Stage Elapsed Times')
53 plt.show()
54 ...
55
56
```

Keep Undo ⌂ | 1 of 1 ↑ ↓ ⌂

CONSOLE TERMINAL PROBLEMS 2 OUTPUT PORTS ...

Python 3.11.12 (Uv: .venv)

```
>>> import ...
... # Get ...
... elapsed...
...
... # Create ...
... plt.f...
... plt.h...
... plt.t...
... plt.x...
... plt.ylabel('Frequency')
... plt.grid(True, alpha=0.3)
...
>>>
```

Frequency

Elapsed Time

Ln 46, Col 1 Spaces: 4 UTF-8 LF () Quarto ⌂ Layout: U.S. ⌂

eda.qmd — eda — Positron

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

eda.qmd — eda — Positron

CHAT

Let me check if the plot was created:

- ✓ Using "Get active plot"...

The histogram has been created successfully. It shows the distribution of elapsed times for the Tour de France stages. From the plot, we can observe that:

1. The distribution is heavily right-skewed
2. Most elapsed times are clustered in the lower range (close to 0)
3. There are some outliers with much longer elapsed times (extending out to around 500000)

The current scale might make it a bit hard to see the details in the main cluster of times. Would you like me to modify the plot to better visualize the distribution, perhaps by:

1. Using a log scale
2. Focusing on the main cluster of times
3. Converting the elapsed time to a more interpretable unit (like hours or minutes)

Just let me know if you'd like any of these adjustments.

analyze the relationship between elapsed time and stage distance using a scatter plot

investigate outliers in elapsed time - what stages had unusually long times?

convert elapsed time to hours and create a more readable histogram

eda.qmd Current file

Python 3.11.12 (Uv: .venv) | eda

Console

Native support for multiple concurrent Python and R consoles

```
CONSOLE TERMINAL PROBLEMS 2 OUTPUT PORTS ...
Python 3.11.12 (Uv: .venv) ~ /data-and-ai-summit-2025/eda
4 elapsed_times = stages.select('elapsed').execute()
>>> import matplotlib.pyplot as plt
...
... # Get the elapsed time data
... elapsed_times = stages.elapsed.execute()
...
... # Create the histogram
... plt.figure(figsize=(10, 6))
... plt.hist(elapsed_times, bins=30, edgecolor='black')
... plt.title('Distribution of Stage Elapsed Times')
... plt.xlabel('Elapsed Time')
... plt.ylabel('Frequency')
... plt.grid(True, alpha=0.3)
>>>
```

Distribution of Stage Elapsed Times

Frequency

Elapsed Time

SESSION CONNECTIONS HELP VIEWER ...

VARIABLES

Python 3.11.12 (Uv: .venv)

DATA

elapsed_times pandas.Series [63913.0, 63968.0, 66012.0, ...]

VALUES

con <ibis.backends.databricks.Backend... Backend

stages ibis.expr.types.relations.Table ibis.Expr

w WorkspaceClient(host='http://...', type=WorkspaceClient)

FUNCTIONS

load_dotenv load_dotenv(dotenv_path: Union[str, Path])

CLASSES

WorkspaceClient <class 'databricks.sdk.WorkspaceClien...' type=WorkspaceClient

PLOTS

eda.qmd — eda — Positron

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

eda.qmd — eda — Positron

CHAT

Let me check if the plot was created:

- ✓ Using "Get active plot"...

The histogram has been created successfully. It shows the distribution of elapsed times for the Tour de France stages. From the plot, we can observe that:

1. The distribution is heavily right-skewed
2. Most elapsed times are clustered in the lower range (close to 0)
3. There are some outliers with much longer elapsed times (extending out to around 500000)

The current scale might make it a bit hard to see the details in the main cluster of times. Would you like me to modify the plot to better visualize the distribution, perhaps by:

1. Using a log scale
2. Focusing on the main cluster of times
3. Converting the elapsed time to a more interpretable unit (like hours or minutes)

Just let me know if you'd like any of these adjustments.

analyze the relationship between elapsed time and stage distance using a scatter plot

investigate outliers in elapsed time - what stages had unusually long times?

convert elapsed time to hours and create a more readable histogram

eda.qmd Current file

Python 3.11.12 (Uv: .venv) ~ /data-and-ai-summit-2025/eda

```
4 elapsed_times = stages.select('elapsed').execute()
>>> import matplotlib.pyplot as plt
...
... # Get the elapsed time data
... elapsed_times = stages.elapsed.execute()
...
... # Create the histogram
... plt.figure(figsize=(10, 6))
... plt.hist(elapsed_times, bins=30, edgecolor='black')
... plt.title('Distribution of Stage Elapsed Times')
... plt.xlabel('Elapsed Time')
... plt.ylabel('Frequency')
... plt.grid(True, alpha=0.3)
>>>
```

SESSION CONNECTIONS HELP VIEWER ...

VARIABLES

Python 3.11.12 (Uv: .venv)

DATA

elapsed_times pandas.Series [63913.0, 63968.0, 66012.0, ...]

VALUES

con <ibis.backends.databricks.Backend... Backend

stages ibis.expr.types.relations.Table ibis.Expr

w WorkspaceClient(host='http://...', WorkspaceClient)

FUNCTIONS

load_dotenv load_dotenv(dotenv_path: Union[str, Path])

CLASSES

WorkspaceClient <class 'databricks.sdk.WorkspaceClien...' type

Plots

Plots render directly within the IDE

PLOTS

Distribution of Stage Elapsed Times

Frequency

Elapsed Time

LN 46, COL 1 SPACES: 4 UTF-8 LF (.) QUARTO LAYOUT: U.S.

eda.qmd — eda — Positron

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

eda.qmd — eda — Positron

CHAT

Let me check if the plot was created:

- ✓ Using "Get active plot"...

The histogram has been created successfully. It shows the distribution of elapsed times for the Tour de France stages. From the plot, we can observe that:

1. The distribution is heavily right-skewed
2. Most elapsed times are clustered in the lower range (close to 0)
3. There are some outliers with much longer elapsed times (extending out to around 500000)

The current scale might make it a bit hard to see the details in the main cluster of times. Would you like me to modify the plot to better visualize the distribution, perhaps by:

1. Using a log scale
2. Focusing on the main cluster of times
3. Converting the elapsed time to a more interpretable unit (like hours or minutes)

Just let me know if you'd like any of these adjustments.

analyze the relationship between elapsed time and stage distance using a scatter plot

investigate outliers in elapsed time - what stages had unusually long times?

convert elapsed time to hours and create a more readable histogram

eda.qmd Current file

CHAT

File

Open

Search

Session

Connections

Help

Viewer

Variables

Data

Values

Functions

Classes

PLOTS

Python 3.11.12 (Uv: .venv)

eda

Session

Connections

Help

Viewer

Environment

See environment details. The Viewer pane provides direct access to running applications.

4 elapsed_times = stages.select('elapsed').execute()
>>> import matplotlib.pyplot as plt
...
... # Get the elapsed time data
... elapsed_times = stages.elapsed.execute()
...
... # Create the histogram
... plt.figure(figsize=(10, 6))
... plt.hist(elapsed_times, bins=30, edgecolor='black')
... plt.title('Distribution of Stage Elapsed Times')
... plt.xlabel('Elapsed Time')
... plt.ylabel('Frequency')
... plt.grid(True, alpha=0.3)
>>>

Distribution of Stage Elapsed Times

Frequency

Elapsed Time

Ln 46, Col 1 Spaces: 4 UTF-8 LF () Quarto Layout: U.S.

eda.qmd — eda — Positron

eda.qmd

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

CONSOLE TERMINAL PROBLEMS 2 OUTPUT PORTS ...

Python 3.11.12 (Uv: .venv) ~ /data-and-ai-summit-2025/eda

eda.qmd

eda.qmd Current file

@ ...

databricks-apps* Posit Workbench 2 ▲ 0

Quarto: 1.6.42

eda.qmd — eda — Positron

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

eda.qmd — eda — Positron

CHAT

Let me check if the plot was created:

✓ Using "Get active plot"...

The histogram has been created successfully. It shows the distribution of elapsed times for the Tour de France stages. From the plot, we can observe that:

1. The distribution is heavily right-skewed
2. Most elapsed times are clustered in the lower range (close to 0)
3. There are some outliers with much longer elapsed times (extending out to around 500000)

The current scale might make it a bit hard to see the details in the main cluster of times. Would you like me to modify the plot to better visualize the distribution, perhaps by:

1. Using a log scale
2. Focusing on the main cluster of times
3. Converting the elapsed time to a more interpretable unit (like hours or minutes)

Just let me know if you'd like any of these adjustments made to the plot.

analyze the relationship between elapsed time and stage distance using a scatter plot

investigate outliers in elapsed time - what stages had unusually long times?

convert elapsed time to hours and create a more readable histogram

eda.qmd Current file

Python 3.11.12 (Uv: .venv) | filter

SESSION CONNECTIONS HELP VIEWER ...

VARIABLES

Python 3.11.12 (Uv: .venv)

DATA

elapsed_times pandas.Series [63913.0, 63968.0, 66012...]

VALUES

con <ibis.backends.databricks.Backend... Backend

stages ibis.expr.types.relations.Table ibis.Expr

w WorkspaceClient(host='https://...', port=443, token='...', type=WorkspaceClient)

FUNCTIONS

load_dotenv load_dotenv(dotenv_path: Union[str, Path])

CLASSES

WorkspaceClient <class 'databricks.sdk.WorkspaceClien...' type=WorkspaceClient>

PLOTS

Distribution of Stage Elapsed Times

Frequency

Elapsed Time

Ln 46, Col 1 | Spaces: 4 | UTF-8 | LF | Quarto | Layout: U.S.

GenAI Applications

Building intelligent dashboards



Data: collect(head(stages, 10))

dev.databricks.posit.team/s/e29766ae9dc1bd980d399/?folder=/home/james/data-and-ai-summit-2025/eda

Data: collect(head(stages, 1000)) — eda — Positron

SESSION CONNECTIONS HELP VIEWER ...

Clear Column Sorting

edition dbl year dbl start_date Date stage_results_id str rank dbl

	edition dbl	year dbl	start_date Date	stage_results_id str	rank dbl
1	1.00	1903.00	1903-07-01	stage-1	
2	1.00	1903.00	1903-07-01	stage-1	
3	1.00	1903.00	1903-07-01	stage-1	
4	1.00	1903.00	1903-07-01	stage-1	
5	1.00	1903.00	1903-07-01	stage-1	
6	1.00	1903.00	1903-07-01	stage-1	
7	1.00	1903.00	1903-07-01	stage-1	
8	1.00	1903.00	1903-07-01	stage-1	
9	1.00	1903.00	1903-07-01	stage-1	
10	1.00	1903.00	1903-07-01	stage-1	
11	1.00	1903.00	1903-07-01	stage-1	
12	1.00	1903.00	1903-07-01	stage-1	
13	1.00	1903.00	1903-07-01	stage-1	
14	1.00	1903.00	1903-07-01	stage-1	
15	1.00	1903.00	1903-07-01	stage-1	
16	1.00	1903.00	1903-07-01	stage-1	
17	1.00	1903.00	1903-07-01	stage-1	
18	1.00	1903.00	1903-07-01	stage-1	
19	1.00	1903.00	1903-07-01	stage-1	

1,000 rows 13 columns

CONSOLE TERMINAL PROBLEMS 18 OUTPUT PORTS DEBUG CONSOLE

R 4.4.1 ~ /data-and-ai-summit-2025/eda

+ View()
> |

jb-demos - Spark SQL R

- > connect_workflow_demo
- > hive_metastore
- jb-demos
- > titanic
- stages
 - edition : DOUBLE
 - year : DOUBLE
 - start_date : DATE
 - stage_results_id : STRING
 - rank : DOUBLE
 - time : DOUBLE
 - rider : STRING
 - age : DOUBLE
 - team : STRING
 - points : DOUBLE
 - elapsed : DOUBLE
 - bib_number : DOUBLE
 - stage_status : STRING
- main
- samples
- > sol-eng-demo-table
- > sol_eng_baseball
- > sol_eng_databricks_hack
- > sol_eng_demo_nickp
- > sol_eng_demo_thederlof

Layout: U.S.

app.py — python-app — Positron

dev.databricks.posit.team/s/e29766ae9dc1b109afa21/?folder=/home/james/data-and-ai-summit-2025/python-app

app.py — python-app — Positron

Python 3.11.12 (Uv: .venv)

python-app

EXPLORER

PYTHON-APP

- > .databricks
- > .posit
- > .venv
- .env
- .python-version
- app.py
- ! app.yaml
- data_description.md
- greeting.md
- instructions.md
- pyproject.toml
- requirements.txt
- uv.lock

app.py > () pd

```
1 import pandas as pd
2 import numpy as np
3 import plotly.express as px
4 import plotly.graph_objects as go
5 from pathlib import Path
6 import os
7 from shiny import App, ui, reactive
8 from shinywidgets import output_widget, render_widget
9 import matplotlib.pyplot as plt
10 import chatlas
11 import querychat
12 import re
13 from scipy import stats
14 from databricks.sdk import WorkspaceClient
15 from posit.connect.external.databricks import (
16     ConnectStrategy,
17     databricks_config
```

SESSION CONNECTIONS HELP VIEWER ...

VARIABLES

Run Shiny App
Debug Shiny App
Run Python File in Console ⌘⌘Enter
Run Python File in Terminal
Run Selection in Console ⌘⌘Enter
Debug Python File in Terminal

Python Debugger: Debug Python File
Python Debugger: Debug using launch.json
Run Script in a Workbench Job

CONSOLE TERMINAL PROBLEMS OUTPUT PORTS DEBUG CONSOLE

~/data-and-ai-summit-2025/python-app

Python 3.11.12 (Uv: .venv) started.
Python 3.11.12 (main, May 17 2025, 13:48:36) [Clang 20.1.4]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.12.3 -- An enhanced Interactive Python. Type '?' for help.

>>>

Ln 1, Col 1 Spaces: 4 UTF-8 LF {} Python Layout: U.S.

app.py — python-app — Positron

SESSION CONNECTIONS HELP VIEWER COPILOT EDITS

https://dev.databricks.posit.team/s/e29766ae9dc1bfce58217/p/7683b831/

Tour de France Analysis

- Filter to show only riders who have won a stage

Data Analysis Questions

- What was the average rider age for each year?
- Which rider has won the most stages?

How many total cyclists are in the data?

I'll find the total number of unique cyclists in the dataset for you.

```
SELECT COUNT(DISTINCT rider) AS total_cyclists
```

total_cyclists
5162

There are 5,162 unique cyclists in the Tour de France dataset.

Enter a message...

Overview

Most Stage Wins

Rider	Stage Wins
Vlaminck Eddy	35
endish Mark	30
ault Bernard	28
Grevès René	25
igade André	24

Most Stages Completed

Rider	Stages Completed
temelk Joop	387
anel Sylvain	370
mpe Lucien	366
apie George	353
or Raymond	335

Stage Time Distribution

CONSOLE TERMINAL PROBLEMS OUTPUT ... + v - x bash Shiny

```
data-and-ai-summit-2025:james:python-app$ /home/james/databricks-app/venv/bin/python -m shiny r --port 45983 --reload --autoreload-port 45045 /home/james/databricks-app/app.py
```

325628256639
0055.15.azur
edatabricks.
net/serving-
endpoints/ch
at/completio
ns "HTTP/1.1"
200 "OK"
chatlas -
Provider
'Databricks'
generated a
response of 20
tokens from an
input of 3276
tokens.

[05/30/25 16:18:57] INFO

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python Layout: U.S.

Query History - Databricks

adb-3256282566390055.15.azuredatabricks.net/sql/history?o=3256282566390055&queryId=01f03d71-c871-1640-871b-9450b0243ea8&userId=3...

Microsoft Azure | databricks

Search data, notebooks, recents, and more...

Finished New Query Profile: ON ▾

James Blair · jb-demo-warehouse · Run on DBSQL

ID: 01f03d71-c871-1640-871b-9450b0243ea8

SELECT COUNT(DISTINCT rider) AS total_cyclists
...2 more lines

Bytes read 5.80 MB ▾0% Bytes written 0

Rows read 255,752 Rows written 0

Files read 1 ▾0% Files written 0

See query profile >

See longest operations for this query

Query wall-clock duration

Total wall-clock duration 419 ms

> Scheduling 7% 28 ms

Optimizing query & pruning files 54% 228 ms

Executing 39% 163 ms

Start time May 30, 2025, 10:18:55 AM GMT-06:00

End time May 30, 2025, 10:18:55 AM GMT-06:00

Result fetching by client 10 ms

Query Source

Sqlalchemy +

Query History

User: Me (james@posit.co)

Last 7 days

Compute Duration Status

Query Started at Duration Source

SELECT COUNT(DISTINCT rider) AS total_cyclists FROM stages

Expand query text

> SELECT * FROM stages

> SELECT DISTINCT stage_status FROM stages WHERE stage_st...

> SELECT COUNT(DISTINCT stage_status) FROM stages

> SELECT MIN(bib_number), MAX(bib_number) FROM stages

> SELECT MIN(elapsed), MAX(elapsed) FROM stages

> SELECT MIN(points), MAX(points) FROM stages

> SELECT COUNT(DISTINCT team) FROM stages

> SELECT MIN(age), MAX(age) FROM stages

> SELECT COUNT(DISTINCT rider) FROM stages

> SELECT MIN(time), MAX(time) FROM stages

> SELECT MIN(rank), MAX(rank) FROM stages

Load more

See query profile

New

Workspace Recents Catalog Workflows Compute Marketplace

SQL

SQL Editor Queries Dashboards Genie Alerts

Query History

SQL Warehouses

Data Engineering

Job Runs Data Ingestion Pipelines

Machine Learning

Playground Experiments Features Models Services

posit-dev/querychat

github.com/posit-dev/querychat

README MIT license

querychat: Chat with your data in any language

querychat is a multilingual package that allows you to chat with your data using natural language queries. It's available for:

- [R - Shiny](#)
- [Python - Shiny for Python](#)

Overview

Imagine typing questions like these directly into your dashboard, and seeing the results in realtime:

- "Show only penguins that are not species Gentoo and have a bill length greater than 50mm."
- "Show only blue states with an incidence rate greater than 100 per 100,000 people."
- "What is the average mpg of cars with 6 cylinders?"

querychat is a drop-in component for Shiny that allows users to query a data frame using natural language. The results are available as a reactive data frame, so they can be easily used from Shiny outputs, reactive expressions, downloads, etc.

No packages published

Contributors 4

jcheng5 Joe Cheng
cpsievert Carson Sievert
chendaniely Daniel Chen
schloerke Barret Schloerke

Languages

Language	Percentage
Python	46.7%
R	41.2%
CSS	7.2%
Makefile	4.9%

app.py — python-app — Positron

dev.databricks.posit.team/s/e29766ae9dc1bfce58217?folder=/home/james/data-and-ai-summit-2025/python-app

SESSION CONNECTIONS HELP VIEWER COPILOT EDITS

https://dev.databricks.posit.team/s/e29766ae9dc1bfce58217/p/7683b831/

Tour de France Analysis

- Filter to show only riders who have won a stage

Data Analysis Questions

- What was the average rider age for each year?
- Which rider has won the most stages?

How many total cyclists are in the data?

I'll find the total number of unique cyclists in the dataset for you.

```
SELECT COUNT(DISTINCT rider) AS total_cyclists
```

total_cyclists
5162

There are 5,162 unique cyclists in the Tour de France dataset.

Overview

Most Stage Wins	Most Stages Completed
Vierckx Eddy 35	temelk Joop 387
endish Mark 30	anel Sylvain 370
ault Bernard 28	mpe Lucien 366
Grevès René 25	apie George 353
igade André 24	or Raymond 335

Stage Time Distribution

CONSOLE TERMINAL PROBLEMS OUTPUT ... + - Enter a message... ↴

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python Layout: U.S.

```
# app.py > {} pd
87 def server(input, output, session):
88     w = WorkspaceClient(
89         config = cfg
90     )
91
92     def databricks_claude(system_prompt: str) -> chatlas.Chat:
93         return chatlas.ChatDatabricks(
94             model="databricks-claude-3-7-sonnet",
95             system_prompt=system_prompt,
96             workspace_client = w
97         )
98
99     # Add Databricks SQLAlchemy connection
100    access_token      = w.tokens.create().token_value
101    server_hostname  = w.config.hostname
102    http_path        = os.getenv("DATABRICKS_HTTP_PATH")
103    catalog          = os.getenv("DATABRICKS_CATALOG")
104    schema           = os.getenv("DATABRICKS_SCHEMA")
105
106    querychat_config = querychat.init(
107        SQLAlchemySource(
108            create_engine(
109                url=f".databricks://token:{access_token}@{server_host}{http_path}&catalog={catalog}&schema={schema}"
110            ),
111            "stages",
112            greeting=greeting,
113            data_description=data_desc,
114            create_chat_callback=databricks_claude
115        )
116    )
117
118    # Initialize querychat server object
119    chat = querychat.server("chat", querychat_config)
```

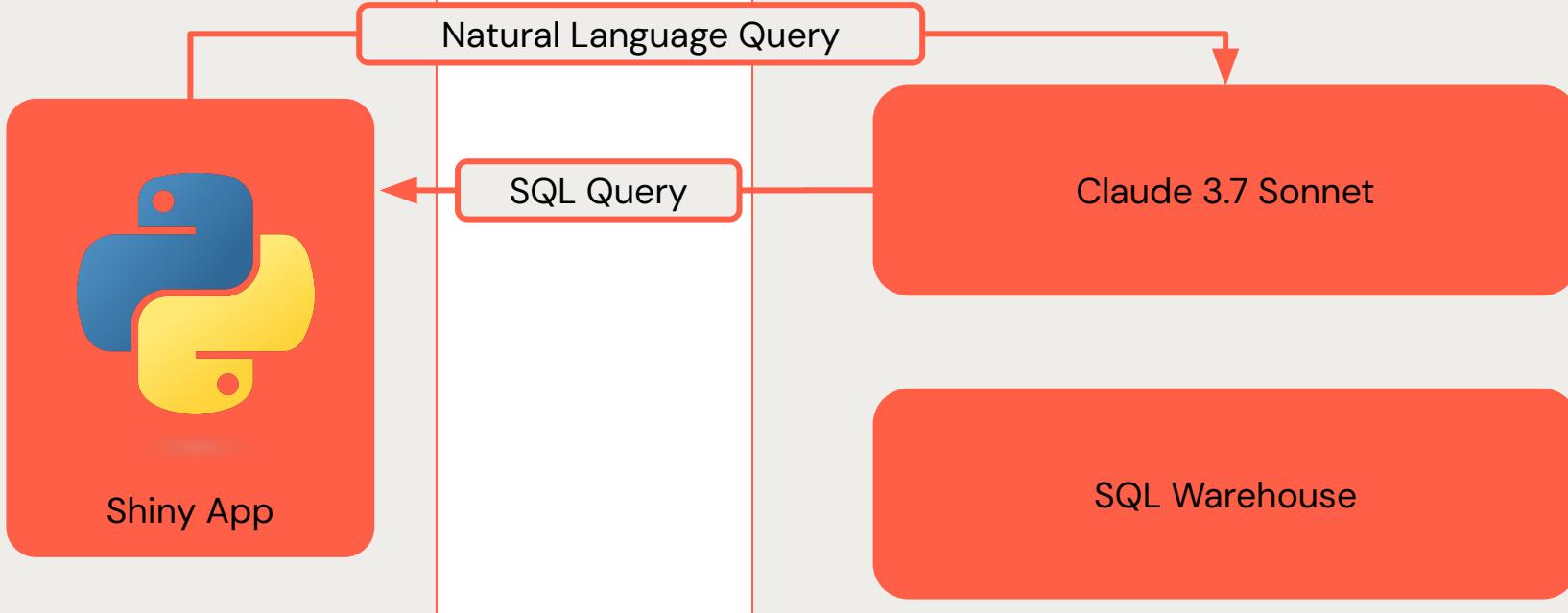


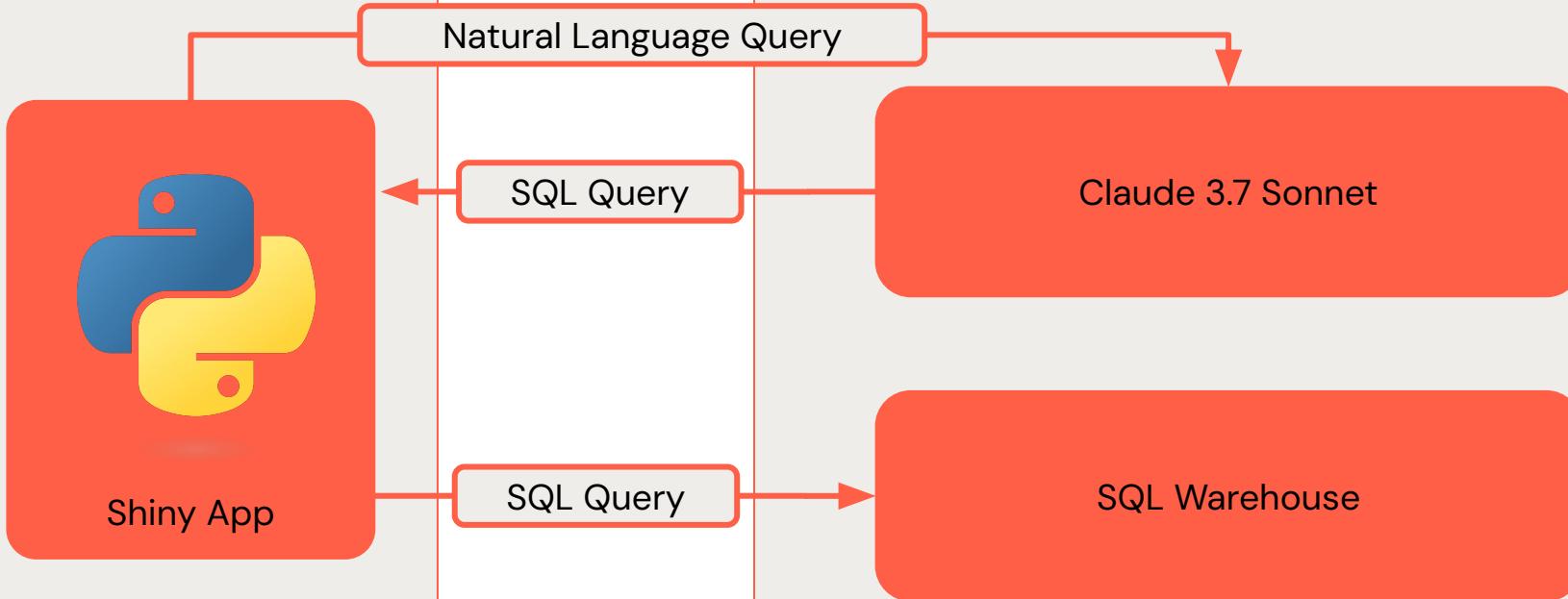
Natural Language Query

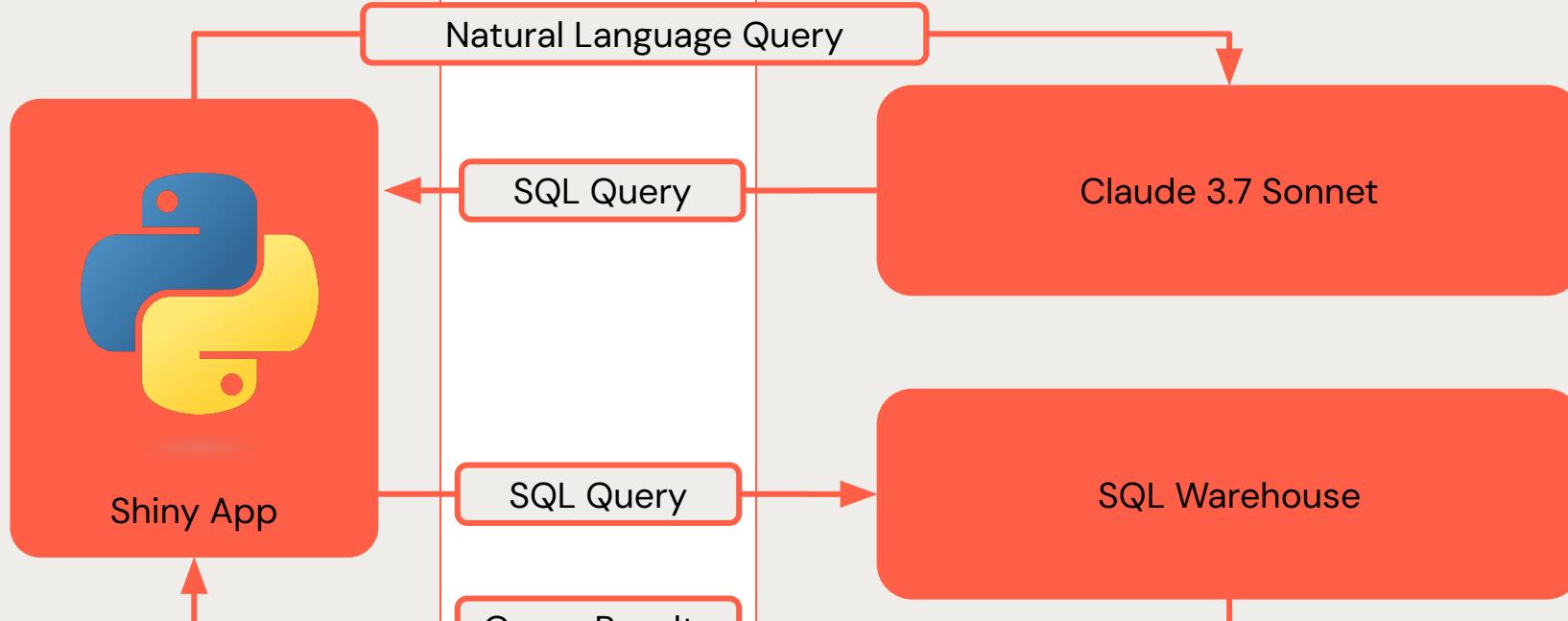
Claude 3.7 Sonnet

SQL Warehouse









App Deployment

Share securely



app.py — python-app — Posit

dev.databricks.posit.team/s/e29766ae9dc1bfce58217/?folder=/home/james/data-and-ai-summit-2025/python-app

Create a New Deployment

Please select where to publish...

POSIT PUBLISHER: HOME

DEPLOYMENT

Select... (new or existing)

CREDENTIALS

HELP AND FEEDBACK

CONSOLE TERMINAL PROBLEMS OUTPUT PORTS DEBUG CONSOLE

Shiny

```
2
3 ui.input_slider("n", "N", 0, 100, 20)
4
5
6 @render.code
7 def txt():
8     return f"n*2 is {input.n() * 2}"
```

simple-app-publish james@session-e29766ae9dc1bfce58217-james---publishing-example-4gn4hs:~/simple-app-publish\$./simple-app-publish/.venv/bin/python -m shiny run --reload /home/james/simple-app-publish/simple-app.py

INFO: 172.16.243.0:0 - "GET /lib/ionrangeslider-2.3.1/js/ion.rangeSlider.min.js HTTP/1.1" 200 OK

INFO: 172.16.8.192:0 - "GET /lib/bootstrap-components-0.9.0.9000/components.min.js HTTP/1.1" 200 OK

INFO: 172.16.218.0:0 - "GET /lib/bootstrap-5.3.1/bootstrap.min.css HTTP/1.1" 200 OK

INFO: 172.16.218.0:0 - "GET /lib/requirejs-2.3.6/require.min.js HTTP/1.1" 200 OK

INFO: 172.16.139.192:0 - "GET /lib/selectize-0.12.6/js/selectize.min.js HTTP/1.1" 200 OK

INFO: 172.16.54.0:0 - "GET /lib/shiny-busy-indicators-1.4.0/busy-indicators.css HTTP/1.1" 200 OK

INFO: 172.16.243.0:0 - "GET /lib/selectize-0.12.6/accessibility/js/selectize-plugin-alloy.min.js HTTP/1.1" 200 OK

INFO: 172.16.100.128:0 - "GET /lib/bootstrap-5.3.1/font.css HTTP/1.1" 200 OK

INFO: 172.16.102.128:0 - "GET /lib/bootstrap-5.3.1/fonts/H1_SiyksKILxRpg3hIP6sJ7fM7PqlPevW.woff2 HTTP/1.1" 200 OK

INFO: 172.16.24.64:0 - "GET /lib/bootstrap-5.3.1/fonts/menvYaGs126MiZpBA-UvNbX2vNxB0b0j20VTS-muw.woff2 HTTP/1.1" 200 OK

INFO: 10.37.151.237:0 - "GET /lib/shiny-busy-indicators-1.4.0/spinners/ring.svg HTTP/1.1" 200 OK

INFO: 172.16.146.64:0 - "GET /lib/bootstrap-5.3.1/fonts/menvYaGs126MiZpBA-UFUicVXSCEKx2cmqvXLwqhuU6F.woff2 HTTP/1.1" 200 OK

('172.16.139.192', 0) - "WebSocket /websocket/" [accepted]

INFO: connection open

INFO: connection closed

Ln 8, Col 37 Spaces: 4 UTF-8 LF {} Python Layout: U.S.

Databricks

adb-3256282566390055.15.azure.databricks.net/apps/tdf-dashboard?o=3256282566390055

Microsoft Azure | databricks

Search data, notebooks, recents, and more... ⌘ + P

posit-databricks-eastus2 ▾

New

Workspace

Recents

Catalog

Workflows

Compute

Marketplace

SQL

SQL Editor

Queries

Dashboards

Genie

Alerts

Query History

SQL Warehouses

Data Engineering

Job Runs

Data Ingestion

Pipelines

Machine Learning

Playground

Experiments

Features

Models

Services

Compute > Apps >

tdf-dashboard

Send feedback ⌂ Edit Permissions Deploy

Overview Authorization Deployments Logs Environment

Running https://tdf-dashboard-3256282566390055.15.azure.databricksapps.com

About this app

Creator james@posit.co

Created on 8 hours ago

Last update 8 hours ago

App ID 8a63158d-51fd-40a0-a42...

Description querychat example app

Compute

Up to 2 vCPUs, 6 GB memory, 0.5 DBU/hour

Deployment

Source: /Workspace/Users/james@posit.co/tdf-dashboard

Last deploy completed 8 hours ago

Inspect deployed code

May 29, 2025, 11:43:17 PM MDT ✓ Stopped app successfully

May 29, 2025, 11:43:18 PM MDT ✓ Source code downloaded successfully

May 29, 2025, 11:43:18 PM MDT ✓ App spec validated successfully

May 29, 2025, 11:43:20 PM MDT ✓ App started successfully

App resources

Key	Type	Details	Permissions
sql_warehouse	SQL warehouse	jb-demo-warehouse	Can use
serving_endpoint	Serving endpoint	databricks-claude-3-7-sonnet	Can query

Edit in your IDE

Prerequisites: Set up your Python environment and the Databricks CLI.

Sync the files

Databricks Tour de France Analysis tdf-dashboard-3256282566390055.15.azure.databricksapps.com

Tour de France Analysis

- What was the average rider age for each year?
- Which rider has won the most stages?

Filter to the most recent decade

I'll filter the dashboard to show data from the most recent decade in the dataset.

```
SELECT MAX(year) FROM stages
```

max(year)
2019.0

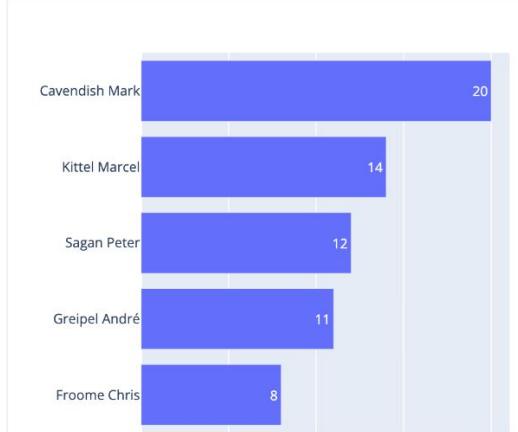
Based on the data, the most recent year in the dataset is 2019. I'll filter to show the last decade (2010-2019).

```
SELECT * FROM stages WHERE year
```

I've filtered the dashboard to show only the Tour de France data from the most recent decade (2010-2019).

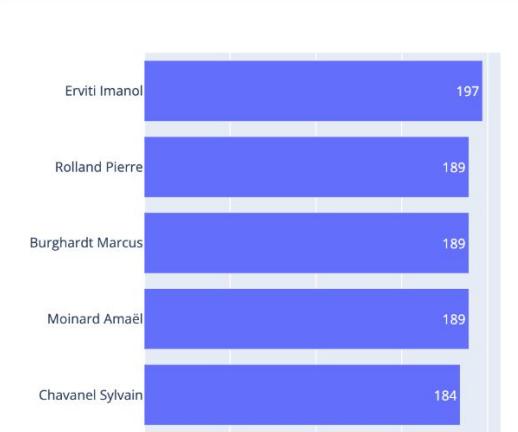
Enter a message... 

Most Stage Wins



Rider	Stage Wins
Cavendish Mark	20
Kittel Marcel	14
Sagan Peter	12
Greipel André	11
Froome Chris	8

Most Stages Completed



Rider	Stages Completed
Erviti Imanol	197
Rolland Pierre	189
Burghardt Marcus	189
Moinard Amaël	189
Chavanel Sylvain	184

Stage Time Distribution



tdf-dashboard - Posit Connect

pub.databricks.posit.team/connect/#/apps/08b53606-f454-451c-8532-591b52275621/access

Content / tdf-dashboard

Tour de France Analysis

Hello! Welcome to the Tour de France dashboard! I'm here to help you explore the Tour de France dataset. Here are some things you can do:

Filtering and Sorting

- Show only the most recent 10 years of data
- Filter to show only riders who have won a stage

Data Analysis Questions

- What was the average rider age for each year?
- Which rider has won the most stages?

Enter a message...

Overview

Most Stage Wins

Rider	Stage Wins
Merckx Eddy	35
Cavendish Mark	30
Hinault Bernard	28
Le Grevès René	25
Darrigade André	24

Most Stages Completed

Rider	Stages Completed
Zoetemelk Joop	387
Chavanel Sylvain	370
Van Impe Lucien	366
Hincapie George	353
Poulidor Raymond	335

Stage Time Distribution

A horizontal timeline visualization showing the distribution of stage times across various stages.

JB james@posit.co

Info Access Runtime Schedule Tags Vars

SHARING

Anyone - no login required
 All users - login required
 Specific users or groups

Who can view or change this application

JB James Blair james@posit.co
DD Demo Databricks Us... demo_databricks_us...
NP Nick Pelikan nick.pelikan@posit...

INTEGRATIONS

A Azure Databricks - Viewer
The Azure Databricks integration

CONTENT URL



Doped Cyclist Filter - Databricks

adb-3256282566390055.15.azuredatabricks.net/editor/queries/3605713726373844?o=3256282566390055

Microsoft Azure | databricks

Search data, notebooks, recents, and more...

posit-databricks-eastus2

New

Workspace

Recents

Catalog

Workflows

Compute

Marketplace

SQL

SQL Editor

Queries

Dashboards

Genie

Alerts

Query History

SQL Warehouses

Data Engineering

Job Runs

Data Ingestion

Pipelines

Machine Learning

Playground

Experiments

Features

Models

Services

Doped Cyclist Filter

New SQL editor: ON

Run all (1000)

jb-demo-warehouse S Schedule Share Save

```
CREATE OR REPLACE FUNCTION doping(rider STRING)
  RETURN IF(current_user() != 'demo_databricks_user@posit.co', true, rider NOT IN (
    -- This list was generated by asking Claude for a list of known doping cyclists
    "Armstrong Lance",
    "Landis Floyd",
    "Hamilton Tyler",
    "Ullrich Jan",
    "Pantani Marco",
    "Contador Alberto",
    "Valverde Alejandro",
    "Vinokourov Alexandre",
    "Basso Ivan",
    "Rasmussen Michael",
    "Schleck Frank",
    "Kreuziger Roman",
    "Rebellin Davide",
    "Di Luca Danilo",
    "Ricco Riccardo",
    "Petacchi Alessandro",
    "Scarponi Michele",
    "Sanchez Samuel",
    "Cobo Juan Jose",
    "Hornier Chris",
    "Mosquera Ezequiel",
    "Piepoli Leonardo",
    "Kohl Bernhard",
    "Sellal Emanuele",
    "Jaksche Jorg",
    "Zabel Erik"
  ))
```

Add parameter

Output

tdf-dashboard - Posit Connect

pub.databricks.posit.team/connect/#/apps/08b53606-f454-451c-8532-591b52275621/access

Content / tdf-dashboard

Tour de France Analysis

Filter to show only riders who have won a stage

Data Analysis Questions

- What was the average rider age for each year?
- Which rider has won the most stages?

How many stages did Lance Armstrong win?

I'll find out how many stages Lance Armstrong won in the Tour de France.

```
SELECT COUNT(*) AS stages_won
FROM stages
WHERE rider LIKE '%Armstrong%Lai'
```

stages_won
24

Lance Armstrong won 24 stages in the Tour de France according to this dataset.

Enter a message... ↑

Overview

Most Stage Wins

Merckx Eddy	35
Cavendish Mark	30
Hinault Bernard	28
Le Grevès René	25
Darrigade André	24

Most Stages Completed

Zoetemelk Joop	387
Chavanel Sylvain	370
Van Impe Lucien	366
Hincapie George	353
Poulidor Raymond	335

Stage Time Distribution

SHARING

Anyone - no login required

All users - login required

Specific users or groups

Who can view or change this application

JB James Blair james@posit.co

DD Demo Databricks Us... demo_databricks_us...

NP Nick Pelikan nick.pelikan@posit...

Remove All Groups and Users

INTEGRATIONS

Azure Databricks - Viewer

The Azure Databricks integration

Log out

CONTENT URL

tdf-dashboard - Posit Connect

pub.databricks.posit.team/connect/#/apps/08b53606-f454-451c-8532-591b52275621

Content / tdf-dashboard

Tour de France Analysis

How many stages did Lance Armstrong win?

I'll check how many stages Lance Armstrong won in the Tour de France.

```
SELECT COUNT(*) AS stage_wins
```

stage_wins
0

Let me check if there might be a different way his name is recorded in the database:

```
SELECT DISTINCT rider FROM stag
```

rider
Lance Pascal

I need to check if there's another way Lance Armstrong might be listed:

```
SELECT DISTINCT rider FROM stag
```

Enter a message...

Overview

Most Stage Wins

Rider	Stage Wins
Merckx Eddy	35
Cavendish Mark	30
Hinault Bernard	28
Le Grevès René	25
Darrigade André	24

Most Stages Completed

Rider	Stages Completed
Zoetemelk Joop	387
Chavanel Sylvain	370
Van Impe Lucien	366
Hincapie George	353
Poulidor Raymond	335

Stage Time Distribution

Enter a message...

tdf-dashboard - Posit Connect

pub.databricks.posit.team/connect/#/apps/08b53606-f454-451c-8532-591b52275621

Content / tdf-dashboard

Tour de France Analysis

riderstage_wins

rider	stage_wins
Merckx Eddy	35
Cavendish Mark	30
Hinault Bernard	28
Le Grevès René	25
Darrigade André	24

(Showing only the first 5 rows out of 10.)

Based on my search of the database, Lance Armstrong doesn't appear to be listed among the riders in this dataset. This could be because:

1. His name might be recorded differently than expected
2. His results may have been removed from the dataset due to his disqualification from the Tour de France (his 7 Tour de France titles from 1999-2005 were stripped due to doping violations)

According to official Tour de France records, Lance Armstrong had 22 stage wins before his disqualification, but these appear to have been removed from this particular dataset.

Overview

Most Stage Wins

rider	stage_wins
Merckx Eddy	35
Cavendish Mark	30
Hinault Bernard	28
Le Grevès René	25
Darrigade André	24

Most Stages Completed

rider	stages_completed
Zoetemelk Joop	387
Chavanel Sylvain	370
Van Impe Lucien	366
Hincapie George	353
Poulidor Raymond	335

Stage Time Distribution

Enter a message...

Conclusion

And a look towards the future



In Review

- Positron is a next-gen IDE purpose built for data science and is **best in class** for developing and deploying interactive data apps
- Databricks OAuth support in Posit Workbench **simplifies Databricks access** for Python and R developers
- **querychat** provides a simple interface for building intuitive, **GenAI powered app interfaces** that work directly with Databricks
- Publish and deploy applications to Posit Connect or Databricks Apps for **secure, scalable access** across your organization
- Unity Catalog governance and access controls can be used to **limit who has access to what data**



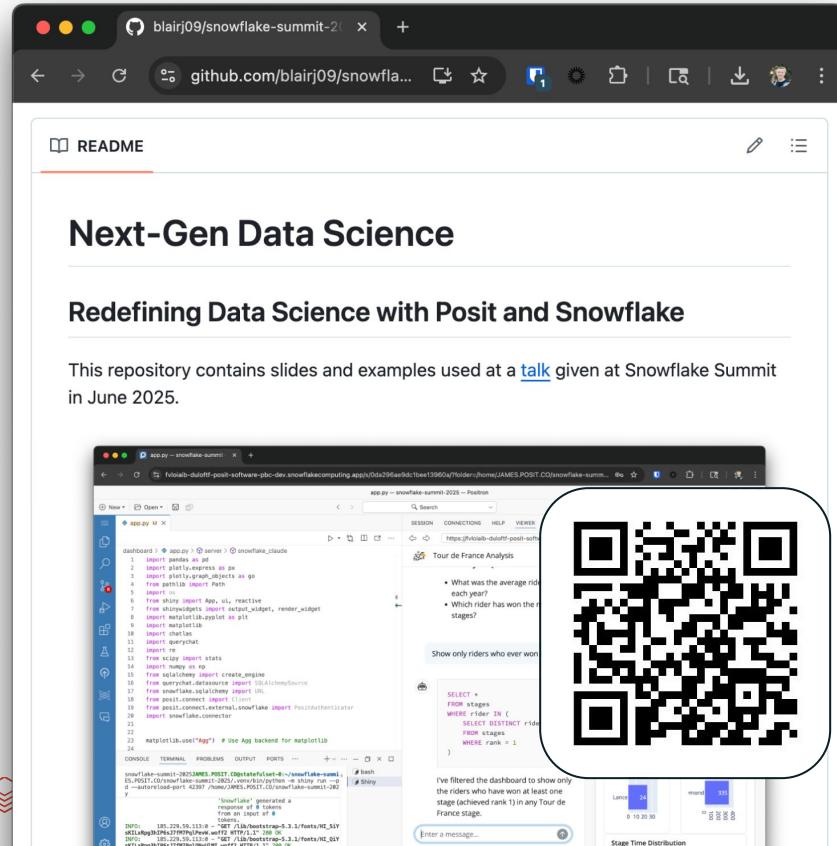
What's Next

- One-click publishing for Databricks Apps
- Posit tools available within Databricks
- Databricks-hosted LLM support for Positron Assistant
- Improved SQL support within Positron



*These are not promises but rather aspirations

Learn More



The screenshot shows a GitHub repository page for 'blairj09/snowflake-summit-2025'. The README file contains the following text:

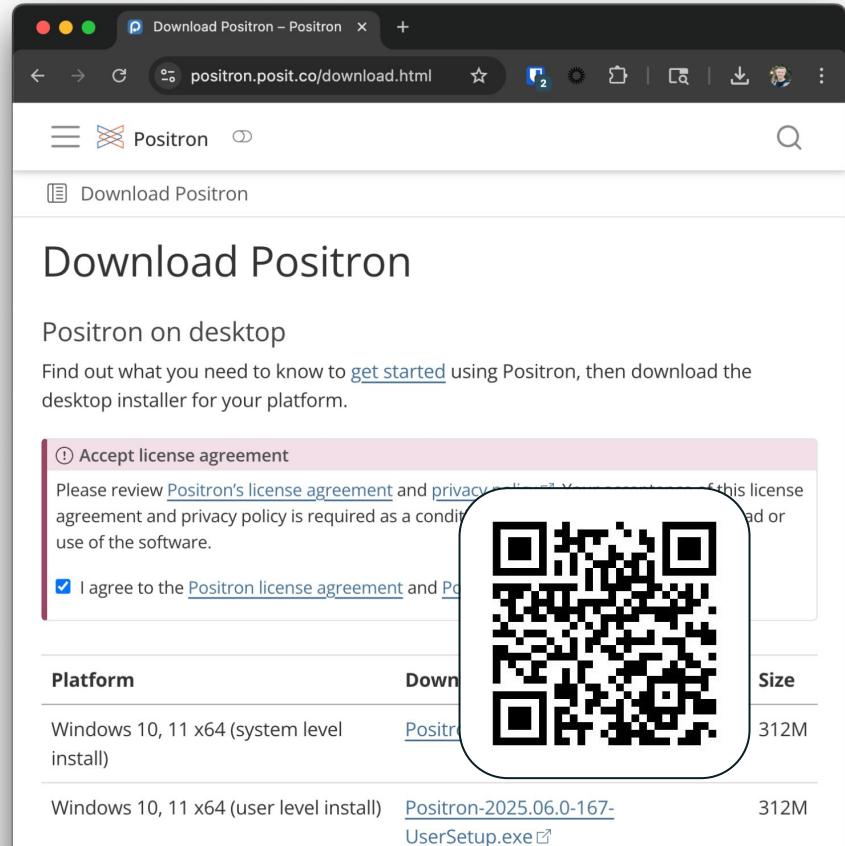
Next-Gen Data Science

Redefining Data Science with Posit and Snowflake

This repository contains slides and examples used at a [talk](#) given at Snowflake Summit in June 2025.

A screenshot of a POSIT dashboard titled 'Tour de France Analysis' is displayed. The dashboard includes a code editor with Python and R scripts, a data viewer showing a table of rider statistics, and a chart titled 'Stage Time Distribution'.

A QR code is overlaid on the bottom right of the dashboard screenshot.



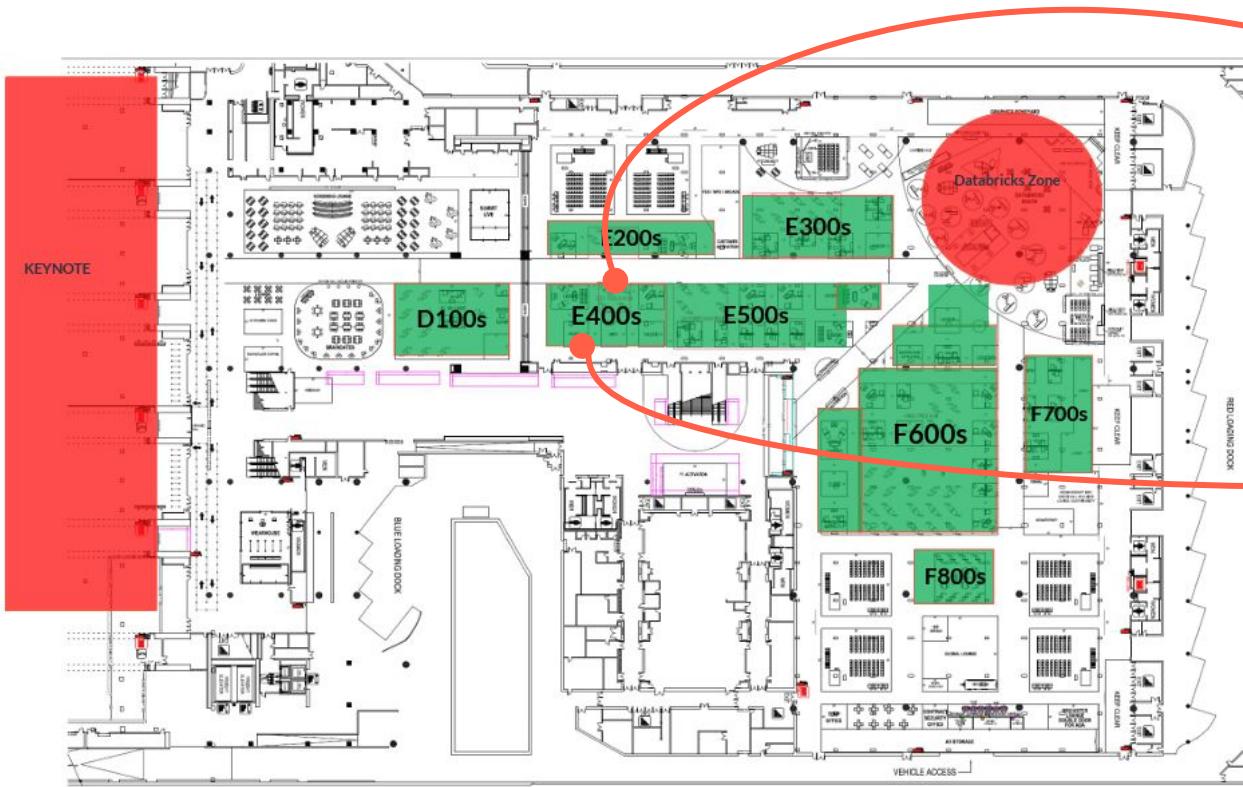
The screenshot shows the 'Download Positron' page on positron.posit.co. The current step is 'Accept license agreement'. The text reads:

Please review [Positron's license agreement](#) and [privacy policy](#). Accepting this license agreement and privacy policy is required as a condition of use of the software.

I agree to the [Positron license agreement](#) and [Privacy Policy](#).

A QR code is overlaid on the bottom right of the page.

Visit Us



**Posit Booth
E405**

**Posit Lounge
402**





databricks