BB-042

Bring Your SAS® and Python Worlds Together With SASPy!

Ted Conway is currently a Data Analyst working with Big Data in the financial sector.

He’s been a happy SAS camper for more than four decades, using SAS on IBM mainframes, PCs, Unix/Linux servers, and AWS, with a variety of programming languages and databases.

Ted studied Computer Science at the University of Illinois and DePaul University.

BB-042

Bring Your SAS and Python Worlds Together With SASPy!

Ted Conway, Chicago, IL

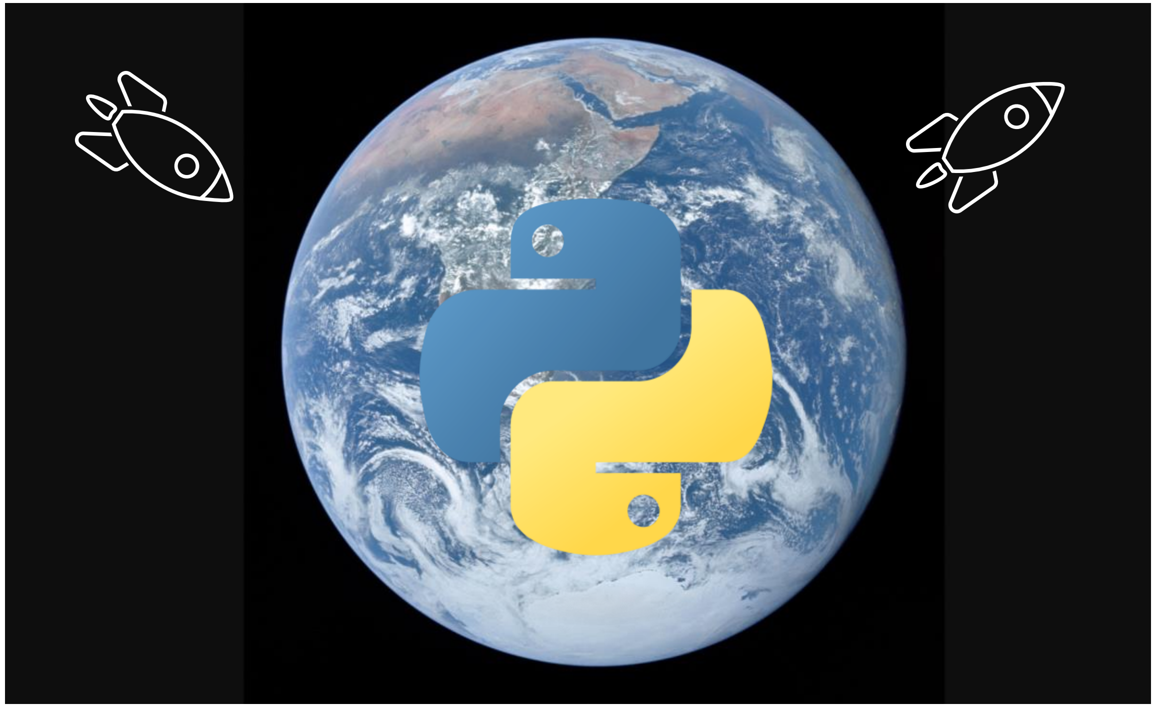
MWSUG 2024

WAUKESHA, WI

LEAD THE WAY AND PLUG INTO SAS!

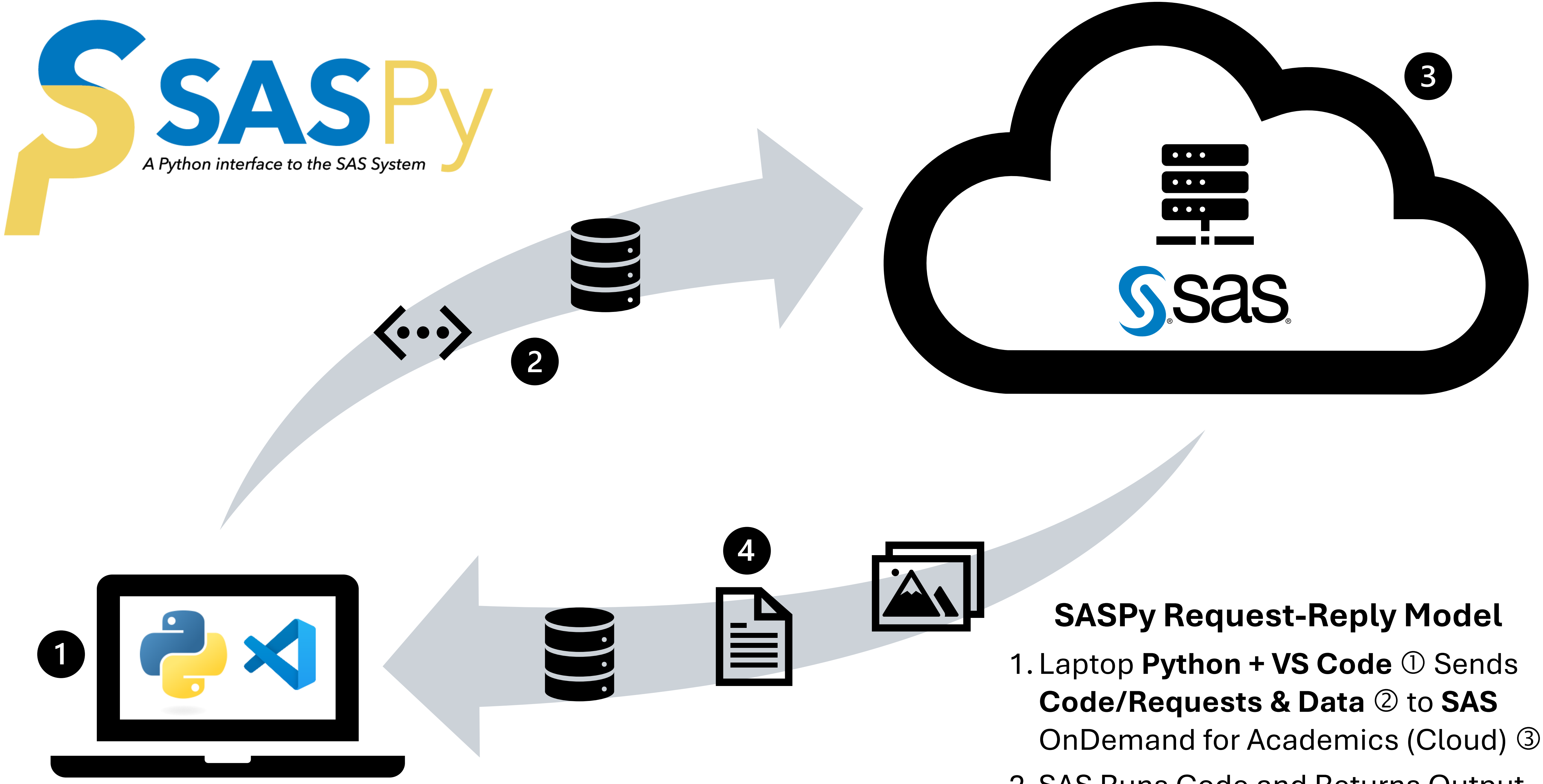
Bring Your SAS and Python Worlds Together With SASPy!

- What is SASPy and how might it help me?
- Got a few examples?
- Sounds good, how do I get started?



Why SASPy?

- Combine the power of SAS & Python!
- Tap into SAS code, procedures and data from directly from Python
- Allows Python and SAS to communicate across platforms & exchange data (SAS runs ‘headless’ in background)
- Brings your Python & SAS code and output together in the same notebooks (VS Code or Jupyter)
- Reduces context switching & friction involved in jumping between your Python and SAS worlds
- Choose Python and SAS features that are right for you and the task at hand
- SAS-supported Open Source



SASPy Request-Reply Model

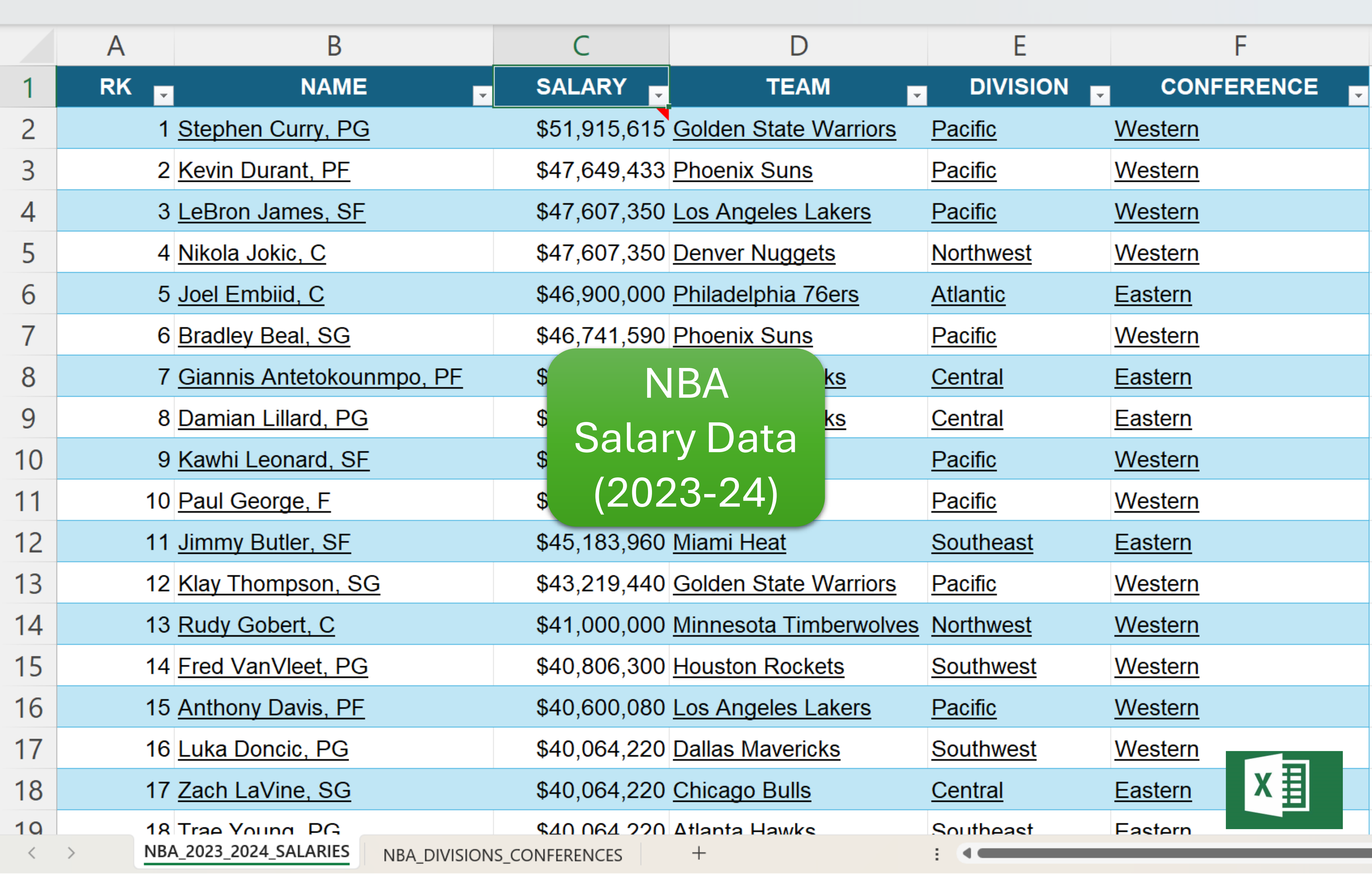
1. Laptop **Python + VS Code** ① Sends **Code/Requests & Data** ② to **SAS** OnDemand for Academics (Cloud) ③
2. SAS Runs Code and Returns Output (**Images, Reports, Data**) ④ to Python

SASPy Show-and-Tell

NBA Salary Data Visualization

- Create an NBA salary SAS dataset that will be accessed from Python
- Connect to Python from VS Code
- Submit SAS code directly from Python with the SASPy %%SAS magic
- Retrieve a SAS dataset into Pandas for use with Python (and vice versa!)
- Launch SAS tasks indirectly from Python using SASPy methods
- Create an interactive Sunburst Chart from the SAS NBA salary dataset using Python and Plotly!

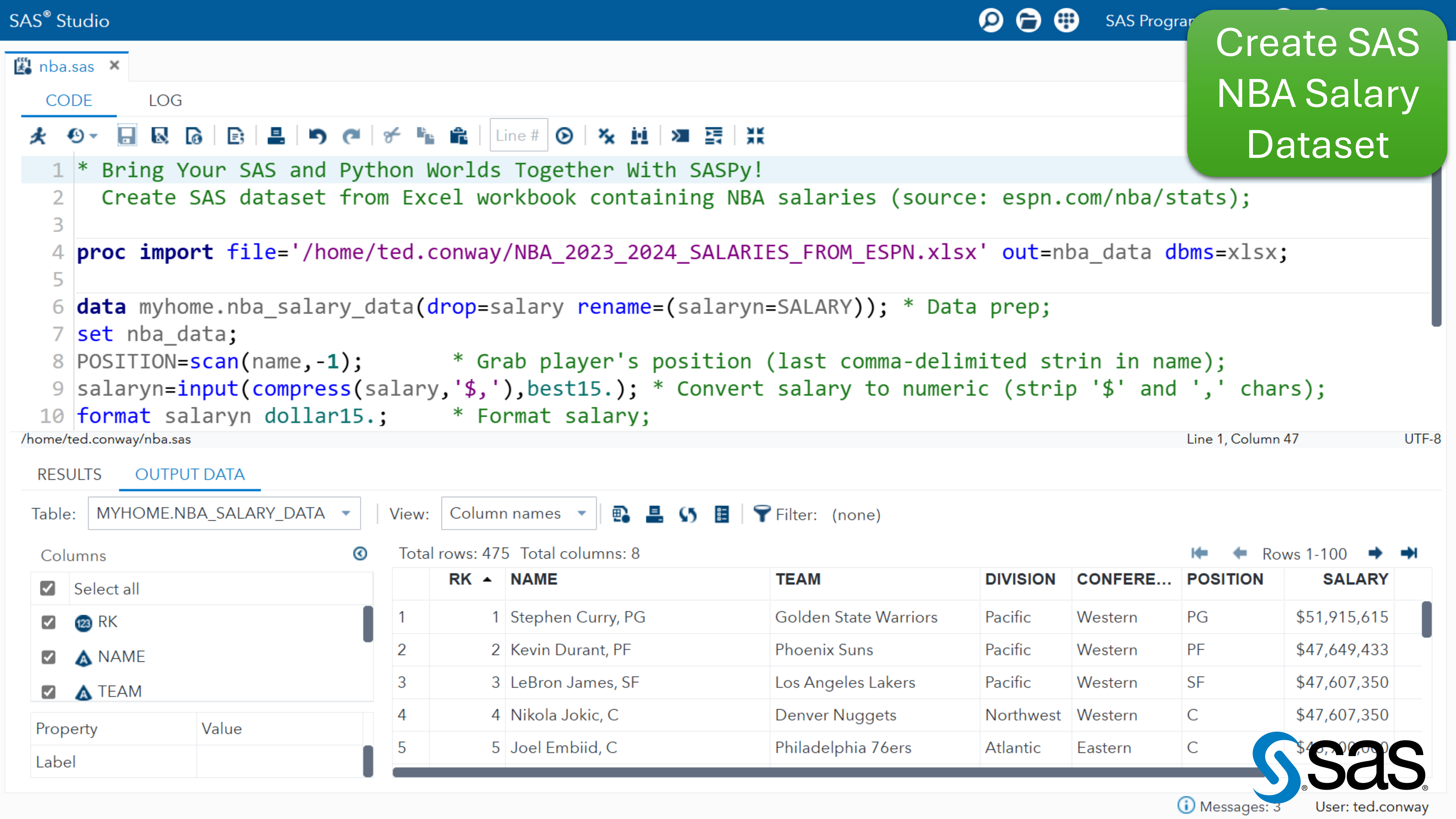
NBA SALARY DATA



RK	NAME	SALARY	TEAM	DIVISION	CONFERENCE
1	Stephen Curry, PG	$51,915,615	Golden State Warriors	Pacific	Western
2	Kevin Durant, PF	$47,649,433	Phoenix Suns	Pacific	Western
3	LeBron James, SF	$47,607,350	Los Angeles Lakers	Pacific	Western
4	Nikola Jokic, C	$47,607,350	Denver Nuggets	Northwest	Western
5	Joel Embiid, C	$46,900,000	Philadelphia 76ers	Atlantic	Eastern
6	Bradley Beal, SG	$46,741,590	Phoenix Suns	Pacific	Western
7	Giannis Antetokounmpo, PF	$	ks	Central	Eastern
8	Damian Lillard, PG	$	ks	Central	Eastern
9	Kawhi Leonard, SF	$		Pacific	Western
10	Paul George, F	$		Pacific	Western
11	Jimmy Butler, SF	$45,183,960	Miami Heat	Southeast	Eastern
12	Klay Thompson, SG	$43,219,440	Golden State Warriors	Pacific	Western
13	Rudy Gobert, C	$41,000,000	Minnesota Timberwolves	Northwest	Western
14	Fred VanVleet, PG	$40,806,300	Houston Rockets	Southwest	Western
15	Anthony Davis, PF	$40,600,080	Los Angeles Lakers	Pacific	Western
16	Luka Doncic, PG	$40,064,220	Dallas Mavericks	Southwest	Western
17	Zach LaVine, SG	$40,064,220	Chicago Bulls	Central	Eastern
18	Trae Young, PG	$40,064,220	Atlanta Hawks	Southeast	Eastern

NBA
Salary Data
(2023-24)

Create SAS NBA Salary Dataset



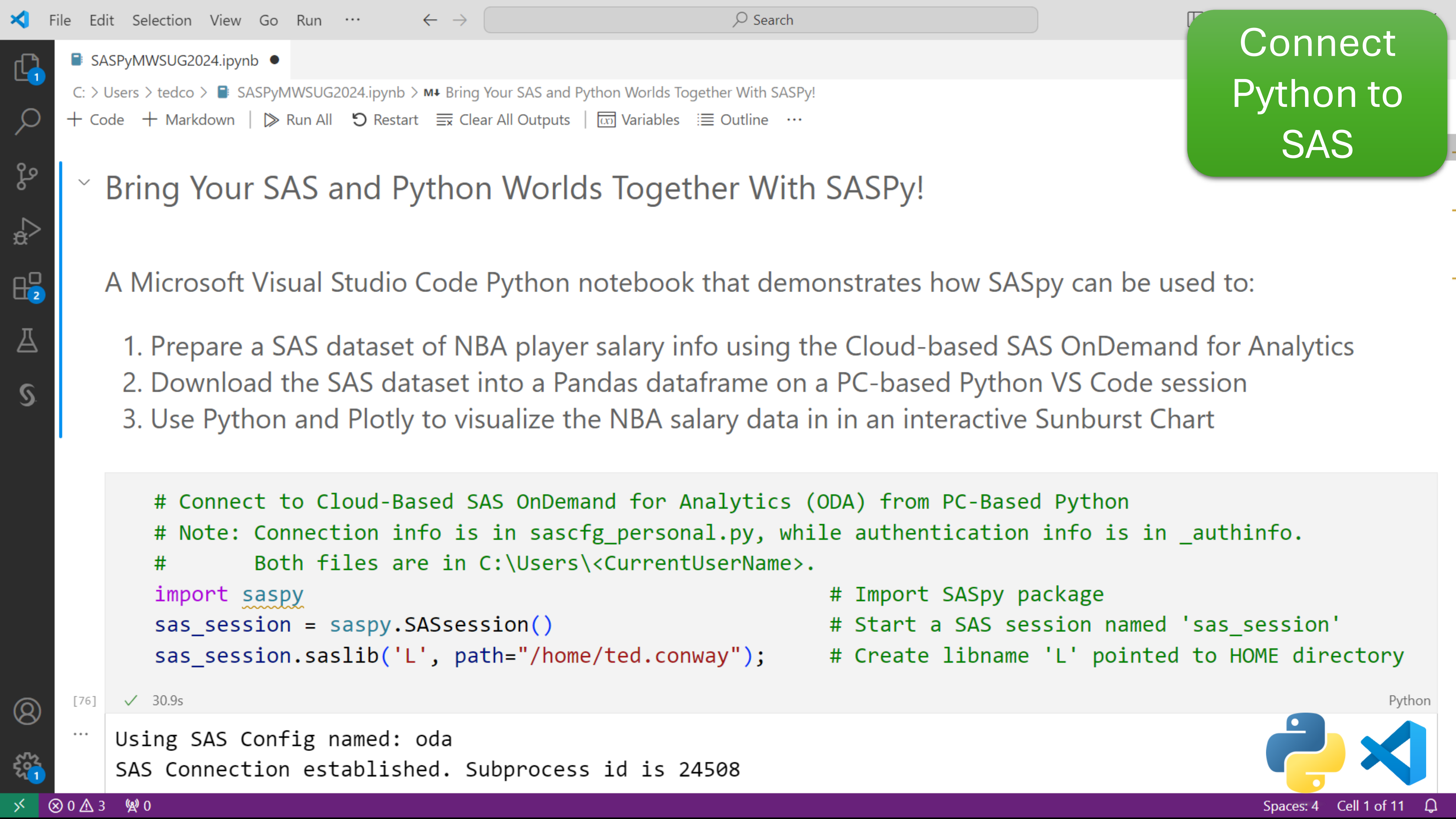
```
* Bring Your SAS and Python Worlds Together With SASPy!
  Create SAS dataset from Excel workbook containing NBA salaries (source: espn.com/nba/stats);

proc import file='/home/ted.conway/NBA_2023_2024_SALARIES_FROM_ESPN.xlsx' out=nba_data dbms=xlsx;

data myhome.nba_salary_data(drop=salary rename=(salaryn=SALARY)); * Data prep;
set nba_data;
POSITION=scan(name,-1);          * Grab player's position (last comma-delimited strin in name);
salaryn=input(compress(salary,'$,'),best15.); * Convert salary to numeric (strip '$' and ',' chars);
format salaryn dollar15.;        * Format salary;
```

Table: MYHOME.NBA_SALARY_DATA View: Column names Filter: (none)

Total rows: 475 Total columns: 8

	RK	NAME	TEAM	DIVISION	CONFERE...	POSITION	SALARY
1	1	Stephen Curry, PG	Golden State Warriors	Pacific	Western	PG	$51,915,615
2	2	Kevin Durant, PF	Phoenix Suns	Pacific	Western	PF	$47,649,433
3	3	LeBron James, SF	Los Angeles Lakers	Pacific	Western	SF	$47,607,350
4	4	Nikola Jokic, C	Denver Nuggets	Northwest	Western	C	$47,607,350
5	5	Joel Embiid, C	Philadelphia 76ers	Atlantic	Eastern	C	$4[illegible]00,000

Connect Python to SAS



Bring Your SAS and Python Worlds Together With SASPy!

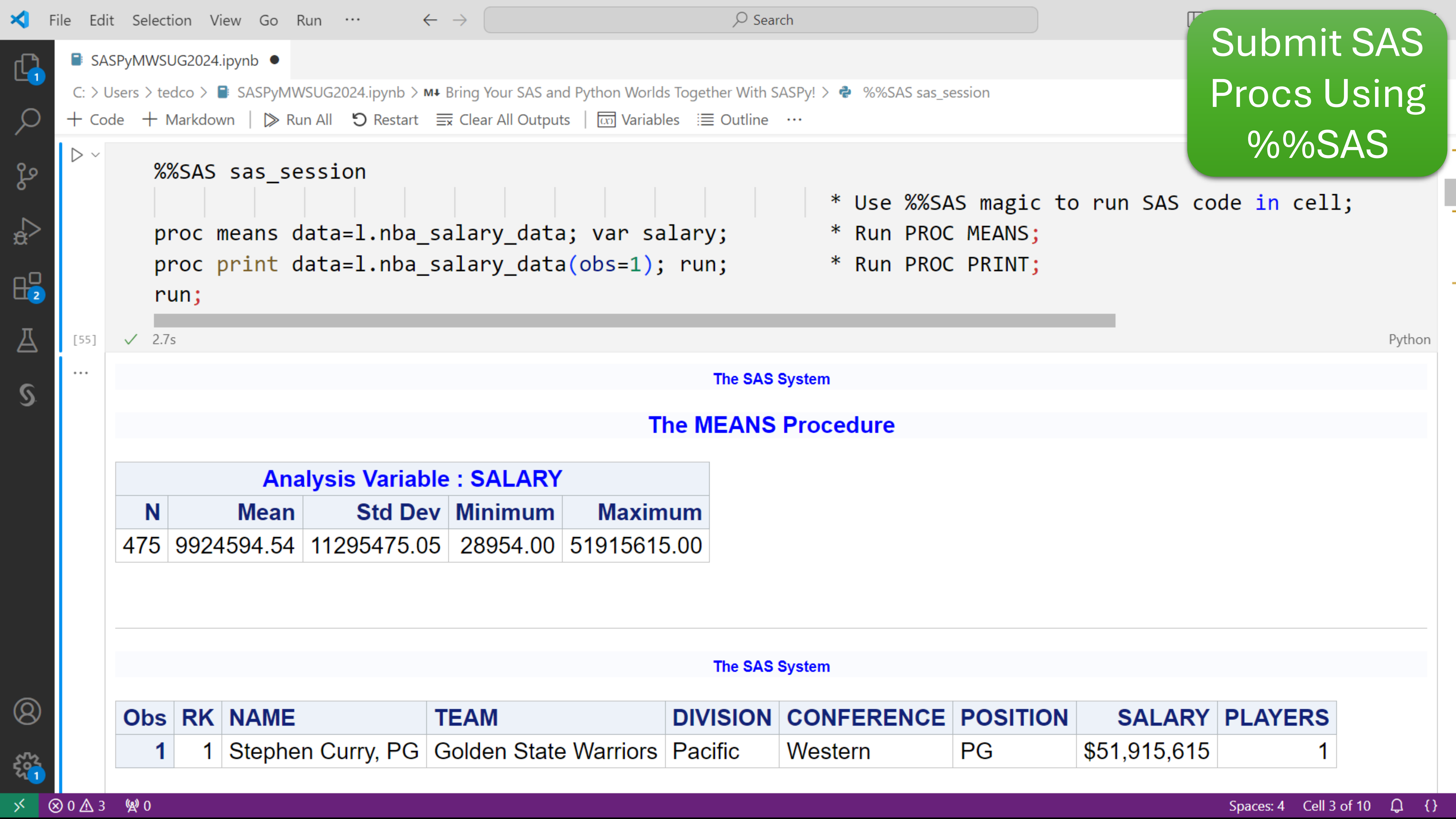
A Microsoft Visual Studio Code Python notebook that demonstrates how SASpy can be used to:

1. Prepare a SAS dataset of NBA player salary info using the Cloud-based SAS OnDemand for Analytics
2. Download the SAS dataset into a Pandas dataframe on a PC-based Python VS Code session
3. Use Python and Plotly to visualize the NBA salary data in in an interactive Sunburst Chart

```
# Connect to Cloud-Based SAS OnDemand for Analytics (ODA) from PC-Based Python
# Note: Connection info is in sascfg_personal.py, while authentication info is in _authinfo.
#       Both files are in C:\Users\<CurrentUserName>.
import saspy                                           # Import SASpy package
sas_session = saspy.SASsession()                       # Start a SAS session named 'sas_session'
sas_session.saslib('L', path="/home/ted.conway");      # Create libname 'L' pointed to HOME directory
```

```
Using SAS Config named: oda
SAS Connection established. Subprocess id is 24508
```

Submit SAS Procs Using %%SAS



```
%%SAS sas_session
                                                    * Use %%SAS magic to run SAS code in cell;
proc means data=l.nba_salary_data; var salary;      * Run PROC MEANS;
proc print data=l.nba_salary_data(obs=1); run;      * Run PROC PRINT;
run;
```

The SAS System

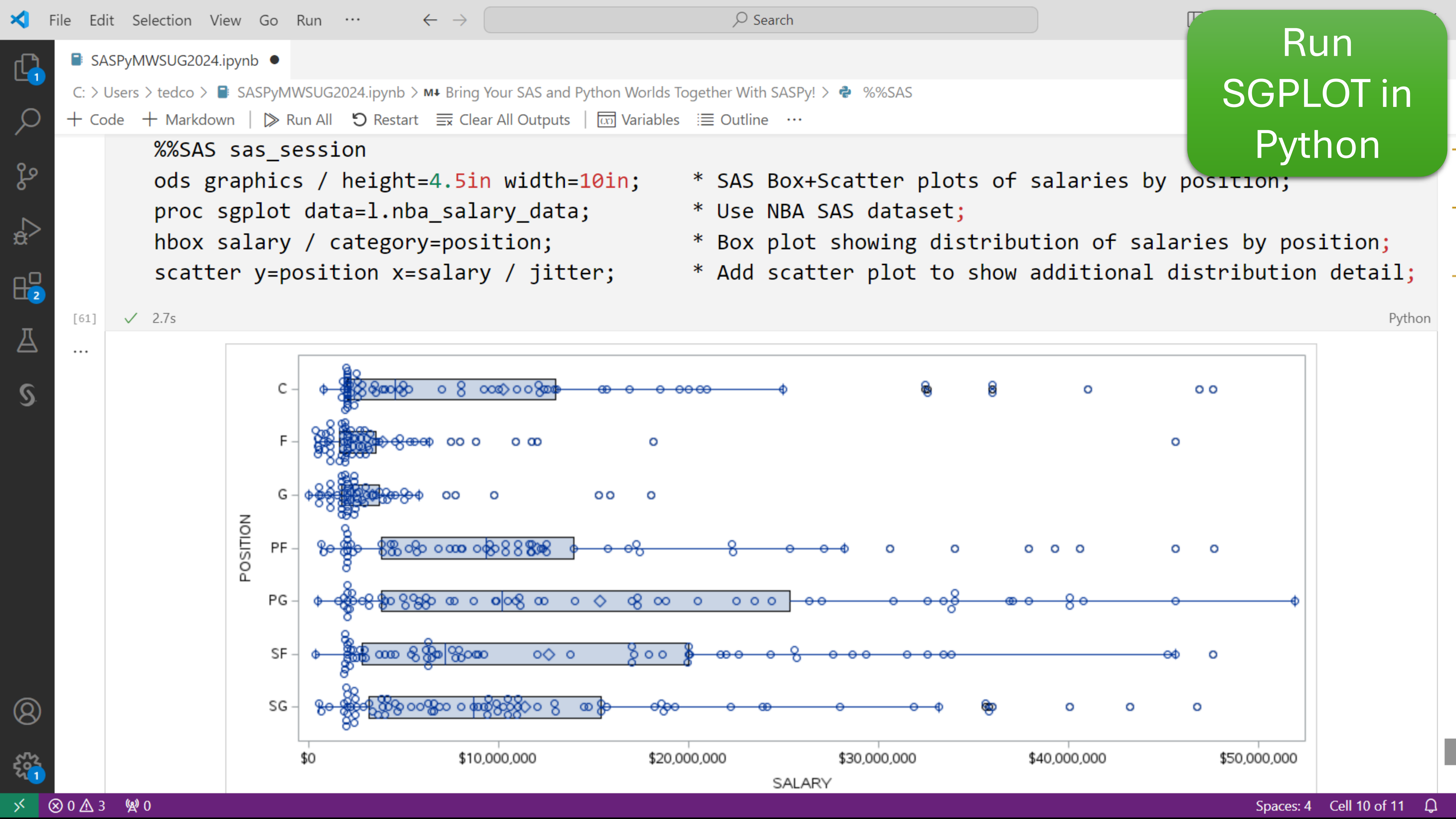
The MEANS Procedure

Analysis Variable : SALARY				
N	Mean	Std Dev	Minimum	Maximum
475	9924594.54	11295475.05	28954.00	51915615.00

The SAS System

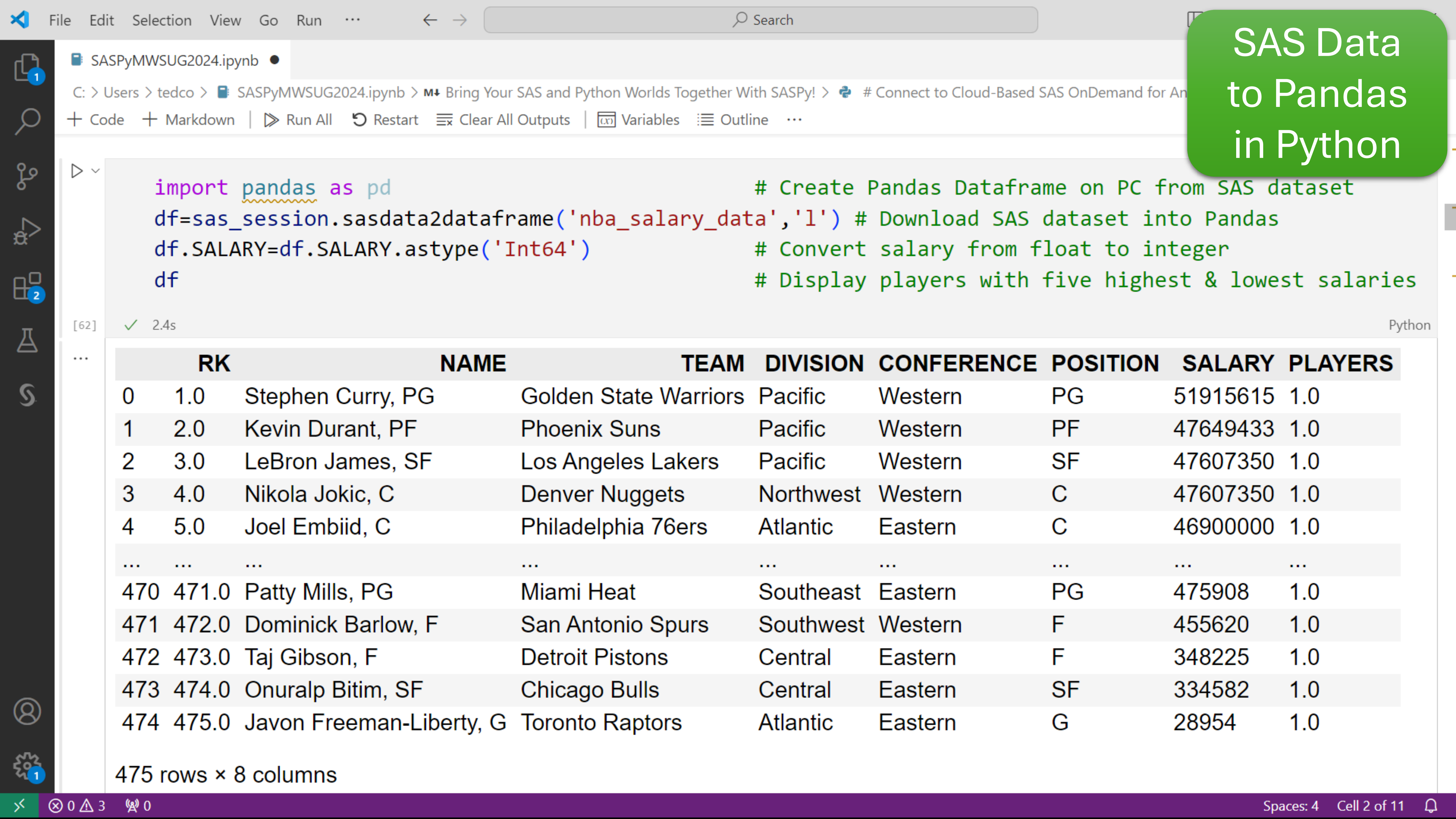
Obs	RK	NAME	TEAM	DIVISION	CONFERENCE	POSITION	SALARY	PLAYERS
1	1	Stephen Curry, PG	Golden State Warriors	Pacific	Western	PG	$51,915,615	1

Run SGPLOT in Python



```
%%SAS sas_session
ods graphics / height=4.5in width=10in;     * SAS Box+Scatter plots of salaries by position;
proc sgplot data=l.nba_salary_data;         * Use NBA SAS dataset;
hbox salary / category=position;            * Box plot showing distribution of salaries by position;
scatter y=position x=salary / jitter;       * Add scatter plot to show additional distribution detail;
```

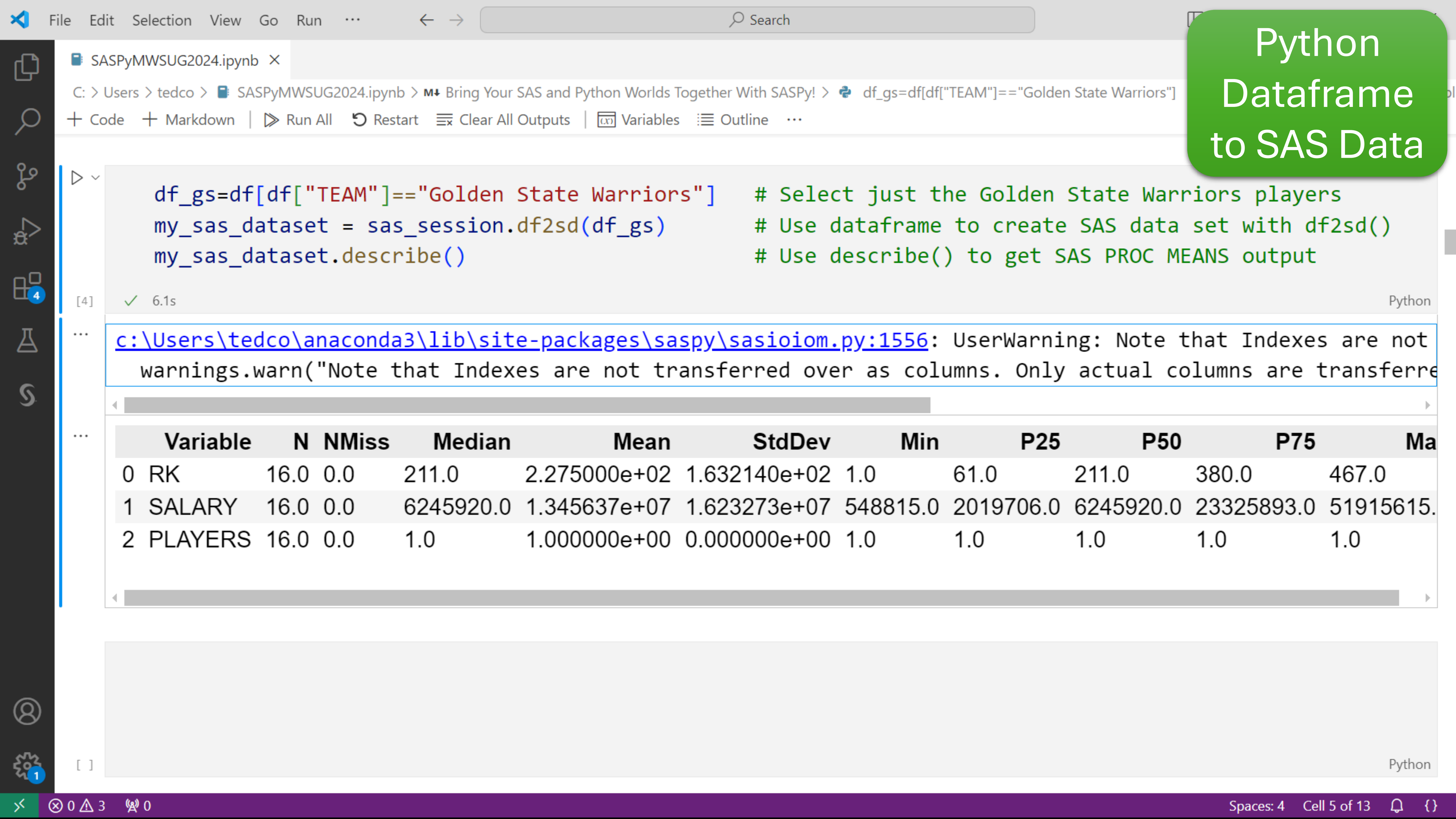
SAS Data to Pandas in Python


File Edit Selection View Go Run ···
Search
SASPyMWSUG2024.ipynb
C: > Users > tedco > SASPyMWSUG2024.ipynb > M↓ Bring Your SAS and Python Worlds Together With SASPy! > # Connect to Cloud-Based SAS OnDemand for An
+ Code + Markdown | Run All Restart Clear All Outputs | Variables Outline ···
[62] ✓ 2.4s
Python
⊗ 0 ⚠ 3 0
Spaces: 4 Cell 2 of 11

```
import pandas as pd                                  # Create Pandas Dataframe on PC from SAS dataset
df=sas_session.sasdata2dataframe('nba_salary_data','l') # Download SAS dataset into Pandas
df.SALARY=df.SALARY.astype('Int64')                  # Convert salary from float to integer
df                                                   # Display players with five highest & lowest salaries
```

```
     RK     NAME                        TEAM                    DIVISION   CONFERENCE  POSITION  SALARY    PLAYERS
0    1.0    Stephen Curry, PG           Golden State Warriors   Pacific    Western     PG        51915615  1.0
1    2.0    Kevin Durant, PF            Phoenix Suns            Pacific    Western     PF        47649433  1.0
2    3.0    LeBron James, SF            Los Angeles Lakers      Pacific    Western     SF        47607350  1.0
3    4.0    Nikola Jokic, C             Denver Nuggets          Northwest  Western     C         47607350  1.0
4    5.0    Joel Embiid, C              Philadelphia 76ers      Atlantic   Eastern     C         46900000  1.0
...  ...    ...                         ...                     ...        ...         ...       ...       ...
470  471.0  Patty Mills, PG             Miami Heat              Southeast  Eastern     PG        475908    1.0
471  472.0  Dominick Barlow, F          San Antonio Spurs       Southwest  Western     F         455620    1.0
472  473.0  Taj Gibson, F               Detroit Pistons         Central    Eastern     F         348225    1.0
473  474.0  Onuralp Bitim, SF           Chicago Bulls           Central    Eastern     SF        334582    1.0
474  475.0  Javon Freeman-Liberty, G    Toronto Raptors         Atlantic   Eastern     G         28954     1.0

475 rows × 8 columns
```

Python Dataframe to SAS Data

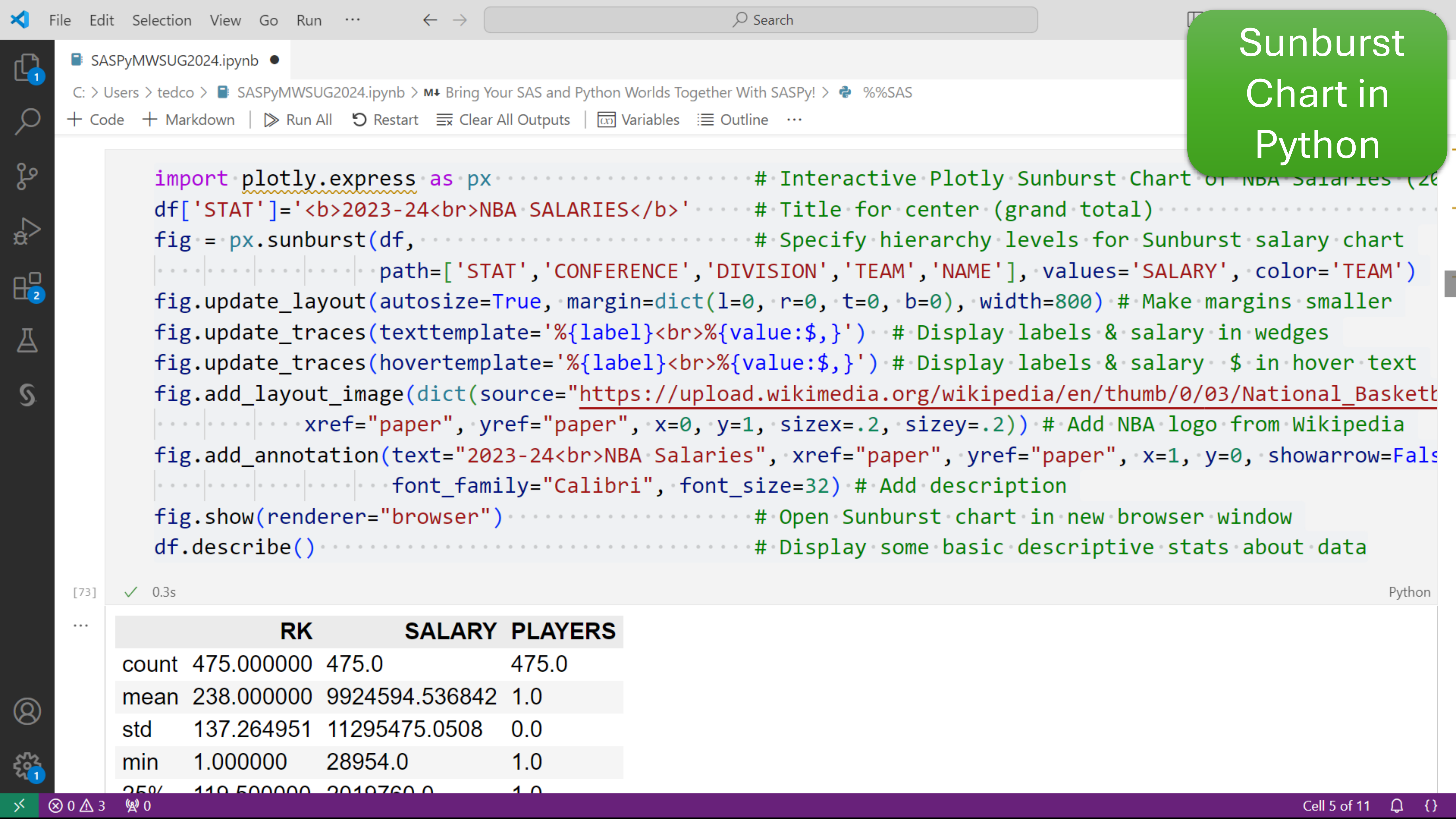


```python
df_gs=df[df["TEAM"]=="Golden State Warriors"]   # Select just the Golden State Warriors players
my_sas_dataset = sas_session.df2sd(df_gs)       # Use dataframe to create SAS data set with df2sd()
my_sas_dataset.describe()                       # Use describe() to get SAS PROC MEANS output
```

```
c:\Users\tedco\anaconda3\lib\site-packages\saspy\sasioiom.py:1556: UserWarning: Note that Indexes are not
  warnings.warn("Note that Indexes are not transferred over as columns. Only actual columns are transferre
```

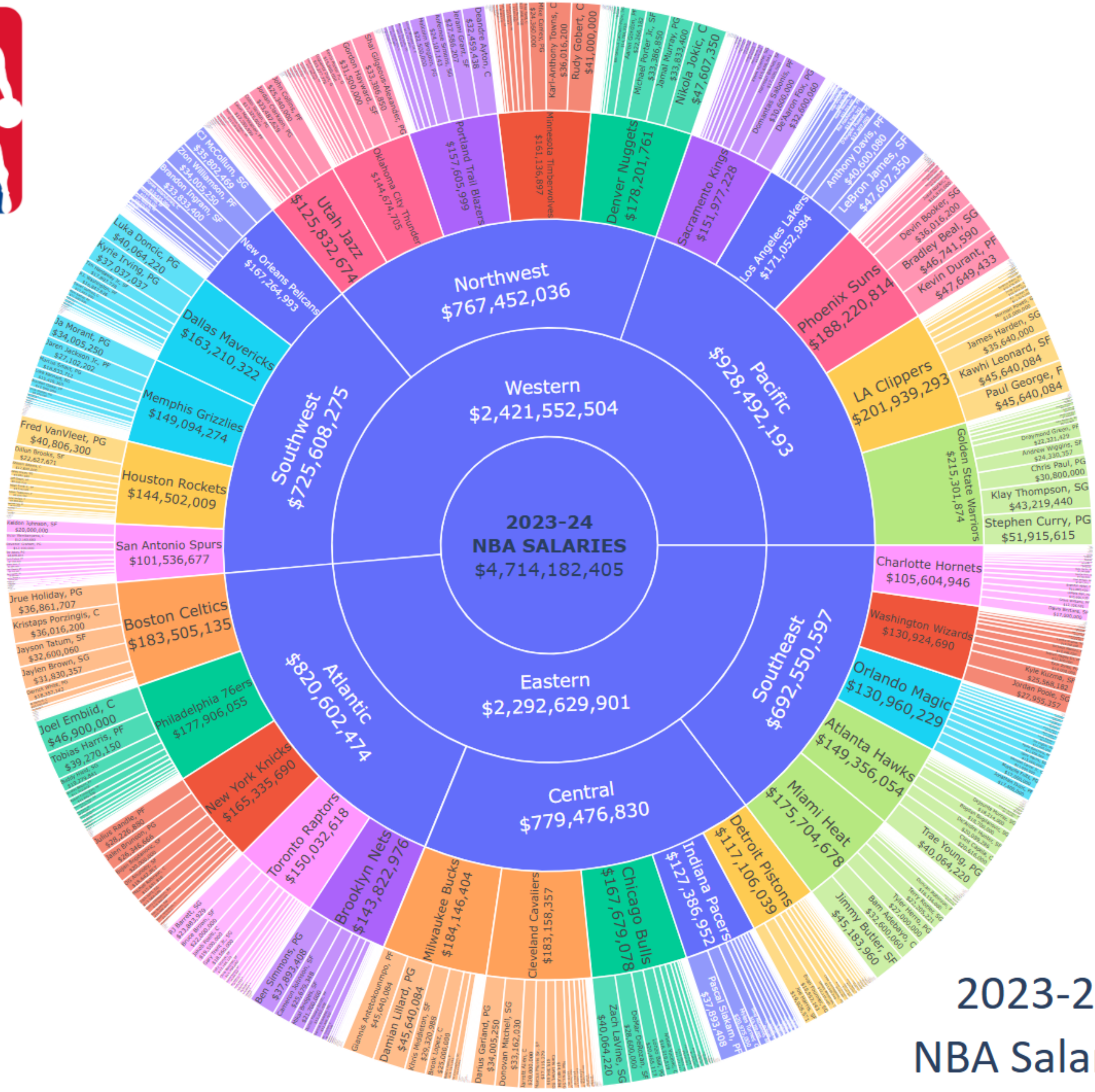
	Variable	N	NMiss	Median	Mean	StdDev	Min	P25	P50	P75	Ma
0	RK	16.0	0.0	211.0	2.275000e+02	1.632140e+02	1.0	61.0	211.0	380.0	467.0
1	SALARY	16.0	0.0	6245920.0	1.345637e+07	1.623273e+07	548815.0	2019706.0	6245920.0	23325893.0	51915615.
2	PLAYERS	16.0	0.0	1.0	1.000000e+00	0.000000e+00	1.0	1.0	1.0	1.0	1.0

Sunburst Chart in Python



```python
import plotly.express as px                    # Interactive Plotly Sunburst Chart of NBA Salaries (2
df['STAT']='<b>2023-24<br>NBA SALARIES</b>'    # Title for center (grand total)
fig = px.sunburst(df,                          # Specify hierarchy levels for Sunburst salary chart
                  path=['STAT','CONFERENCE','DIVISION','TEAM','NAME'], values='SALARY', color='TEAM')
fig.update_layout(autosize=True, margin=dict(l=0, r=0, t=0, b=0), width=800) # Make margins smaller
fig.update_traces(texttemplate='%{label}<br>%{value:$,}')  # Display labels & salary in wedges
fig.update_traces(hovertemplate='%{label}<br>%{value:$,}') # Display labels & salary  $ in hover text
fig.add_layout_image(dict(source="https://upload.wikimedia.org/wikipedia/en/thumb/0/03/National_Basketb
            xref="paper", yref="paper", x=0, y=1, sizex=.2, sizey=.2)) # Add NBA logo from Wikipedia
fig.add_annotation(text="2023-24<br>NBA Salaries", xref="paper", yref="paper", x=1, y=0, showarrow=Fals
                   font_family="Calibri", font_size=32) # Add description
fig.show(renderer="browser")                   # Open Sunburst chart in new browser window
df.describe()                                  # Display some basic descriptive stats about data
```

	RK	SALARY	PLAYERS
count	475.000000	475.0	475.0
mean	238.000000	9924594.536842	1.0
std	137.264951	11295475.0508	0.0
min	1.000000	28954.0	1.0
25%	119.500000	2019760.0	1.0

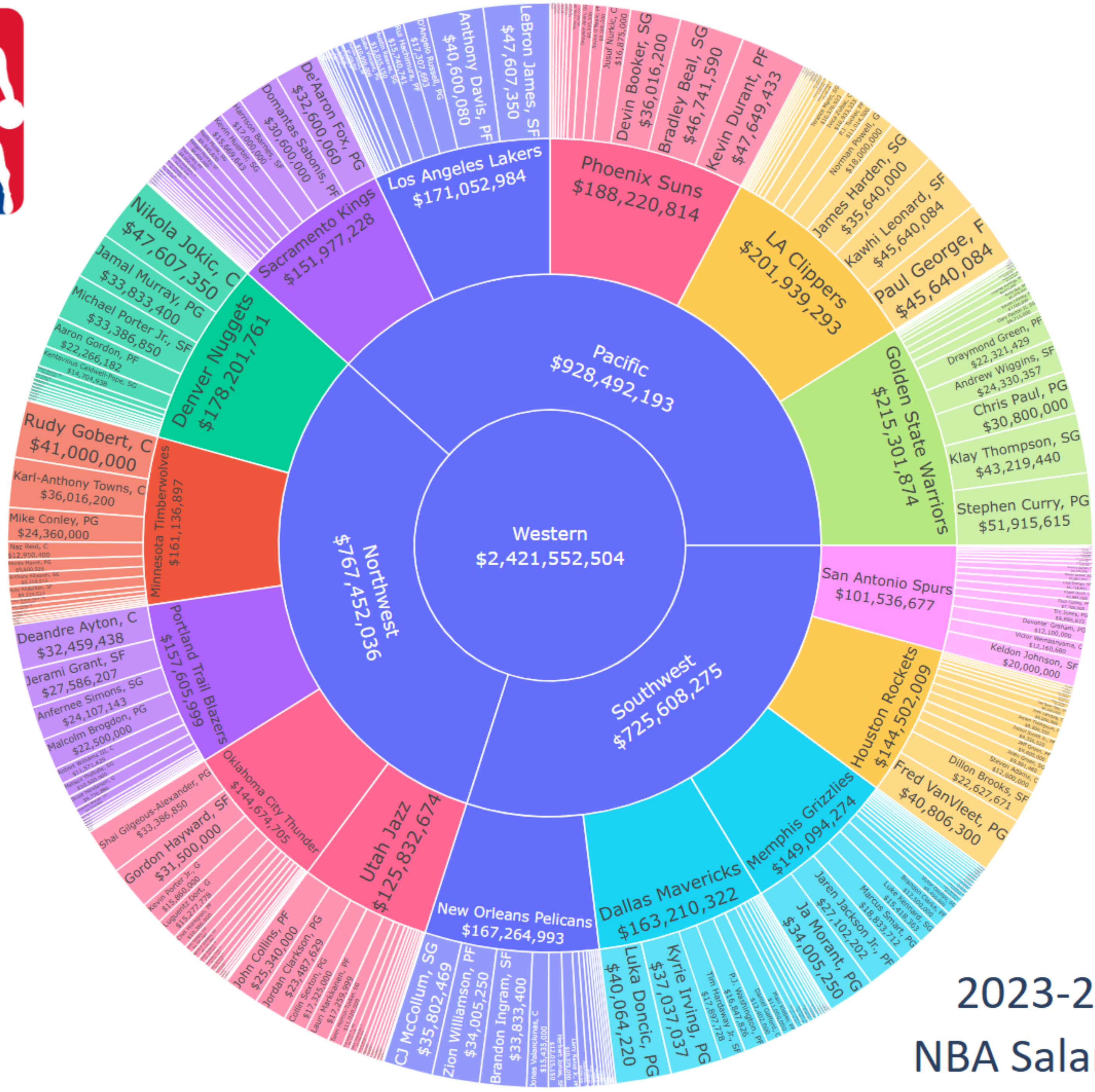


Zoom Level 1

ALL TEAMS

2023-24

NBA Salaries

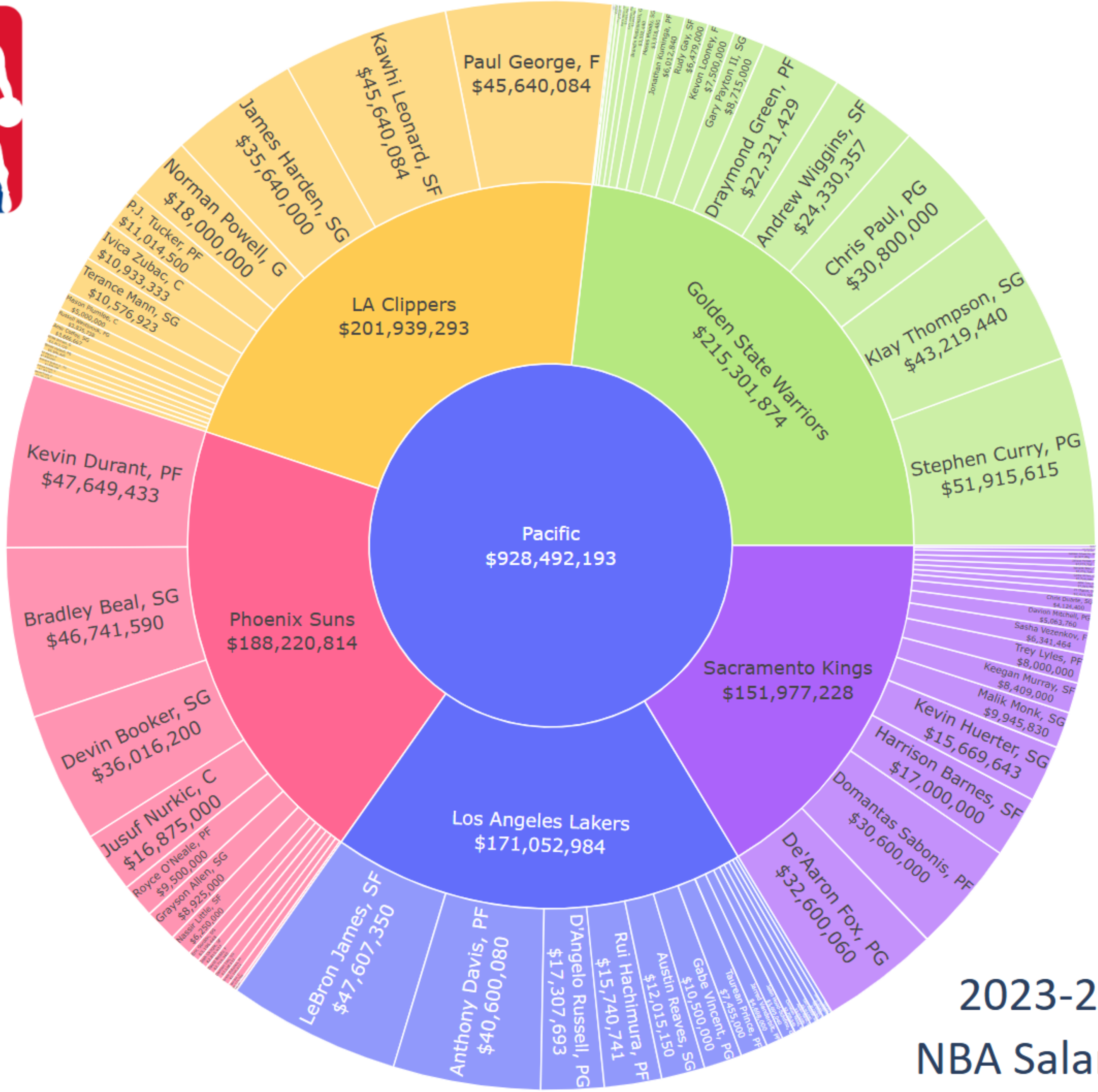


Zoom Level 2

WESTERN

2023-24

NBA Salaries



Zoom Level 3

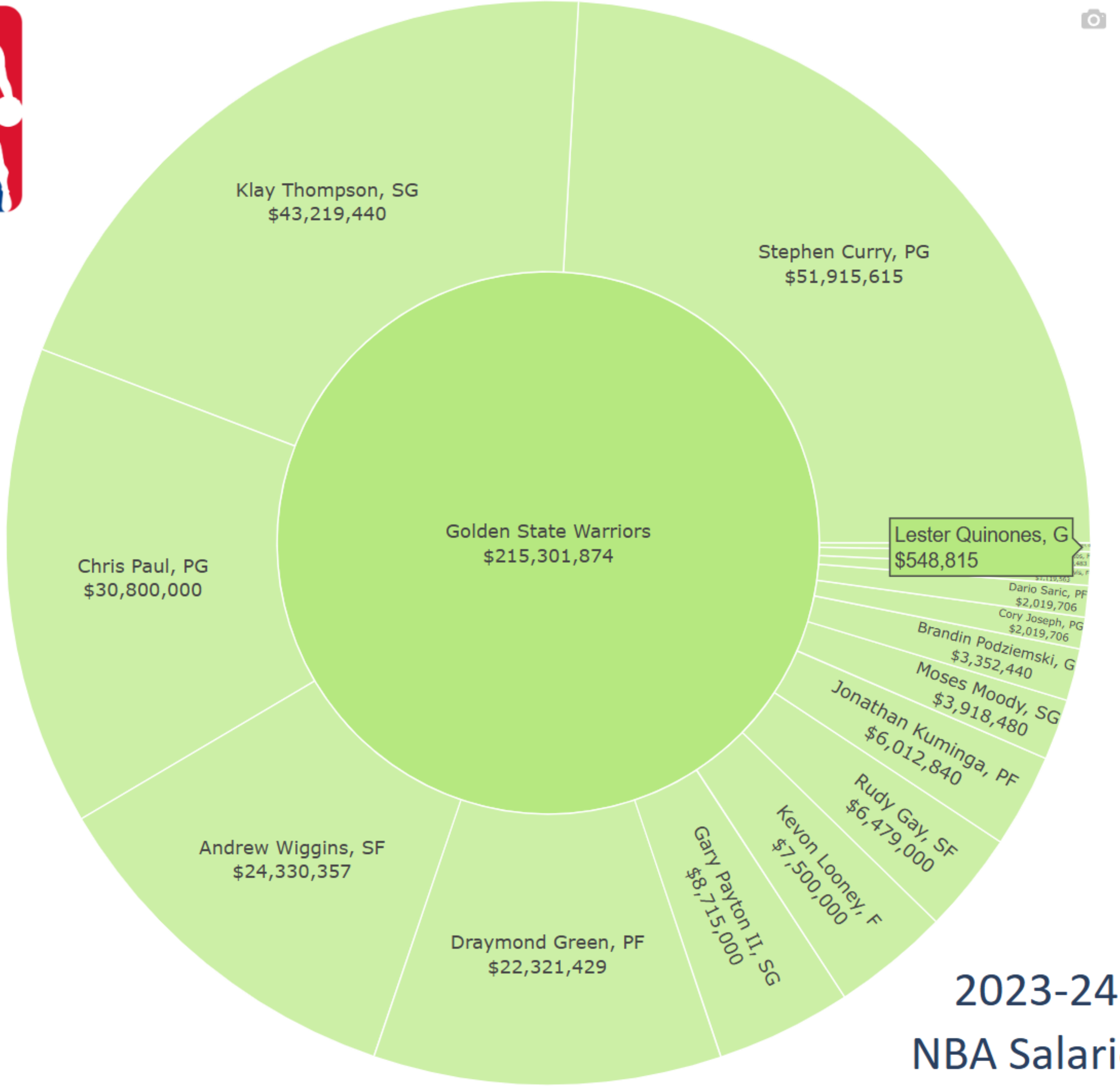
PACIFIC

2023-24

NBA Salaries

Zoom Level 4

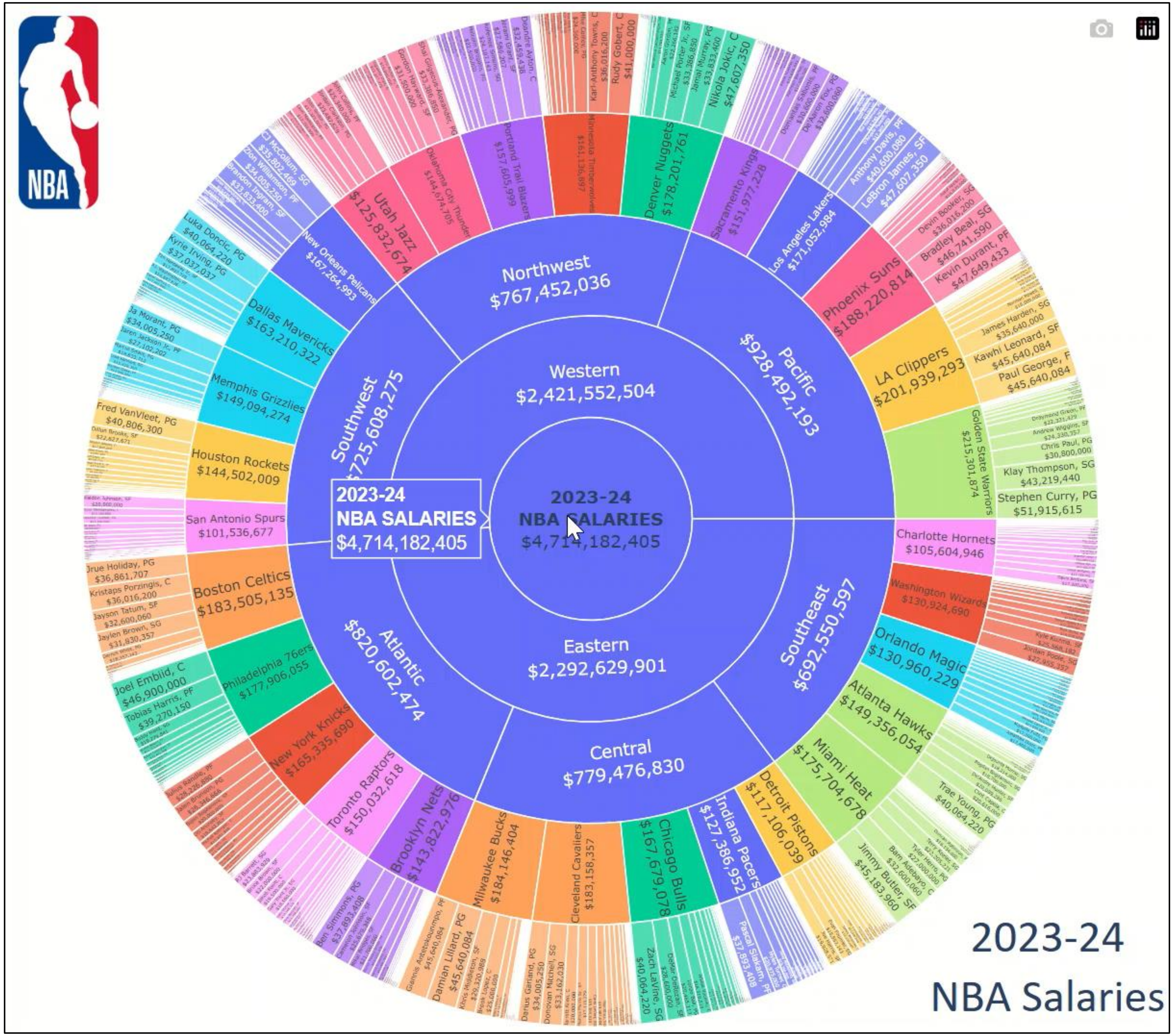
GOLDEN STATE WARRIORS



2023-24

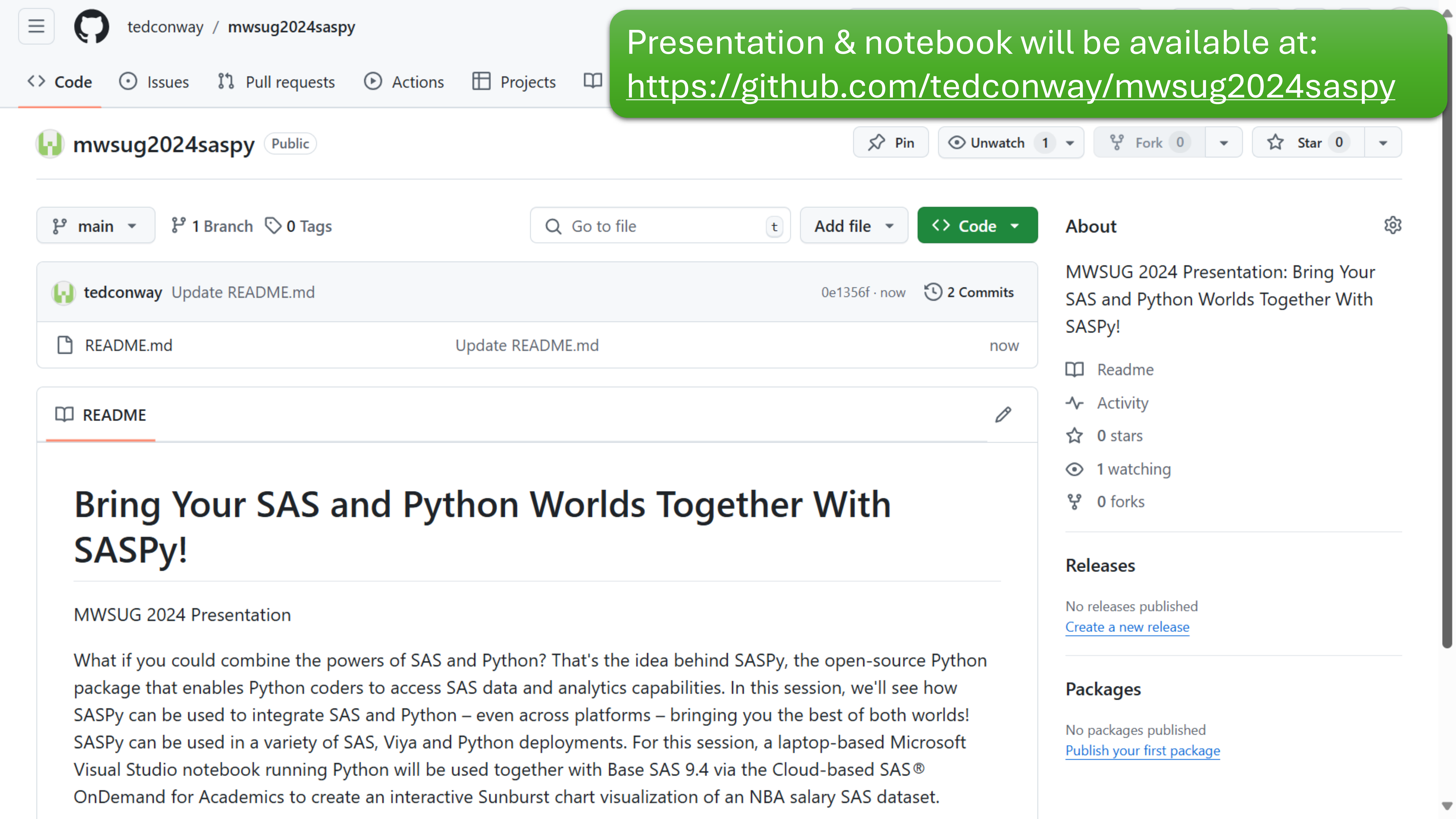
NBA Salaries

NBA_SALARY_DATA SAS Dataset
Interactive Plotly Viewer – Zoom Level 1-4



Movie!

Presentation & notebook will be available at:
https://github.com/tedconway/mwsug2024saspy



About

MWSUG 2024 Presentation: Bring Your SAS and Python Worlds Together With SASPy!

Bring Your SAS and Python Worlds Together With SASPy!

MWSUG 2024 Presentation

What if you could combine the powers of SAS and Python? That's the idea behind SASPy, the open-source Python package that enables Python coders to access SAS data and analytics capabilities. In this session, we'll see how SASPy can be used to integrate SAS and Python – even across platforms – bringing you the best of both worlds! SASPy can be used in a variety of SAS, Viya and Python deployments. For this session, a laptop-based Microsoft Visual Studio notebook running Python will be used together with Base SAS 9.4 via the Cloud-based SAS® OnDemand for Academics to create an interactive Sunburst chart visualization of an NBA salary SAS dataset.

Thank You!

- **Contact Info**
- Ted Conway
- ted.j.conway@gmail.com
- @vivasasvegas (Twitter)

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

APPENDIX

Getting Started With SASPy

Resources/Links

♪♫ Let's start at the very beginning /
A very good place to start... ♪♫

SASPy

Approximately 60 minutes • Makes 1 installation

TIP If desired, use Jupyter instead of VS Code.

INGREDIENTS

- SAS 9.4
- Python, Packages (e.g., Anaconda distribution)
- SASPy
- Plotly
- Microsoft VS Code IDE
- Java
- Configuration & authentication files
- Data
- Python Code
- SAS Code

Calories approx. 0; total fat 0g; saturated fat 0g; cholesterol 0mg; sodium 0mg; total carbohydrate 0g; dietary fiber 0g; protein 0g

1 Install and/or get access to required software from various websites – SAS 9.4, Python & packages (SASPy, Plotly, Anaconda Python + packages, VS Code IDE, Java.

2 Configure SAS connection (IOM, SSH, etc.) and any required authentication.

3 Identify desired data sources – .csv files, SAS datasets, database tables, etc.

4 Add Python and SAS code, run, and serve!

Software Learn Support Partners About Us

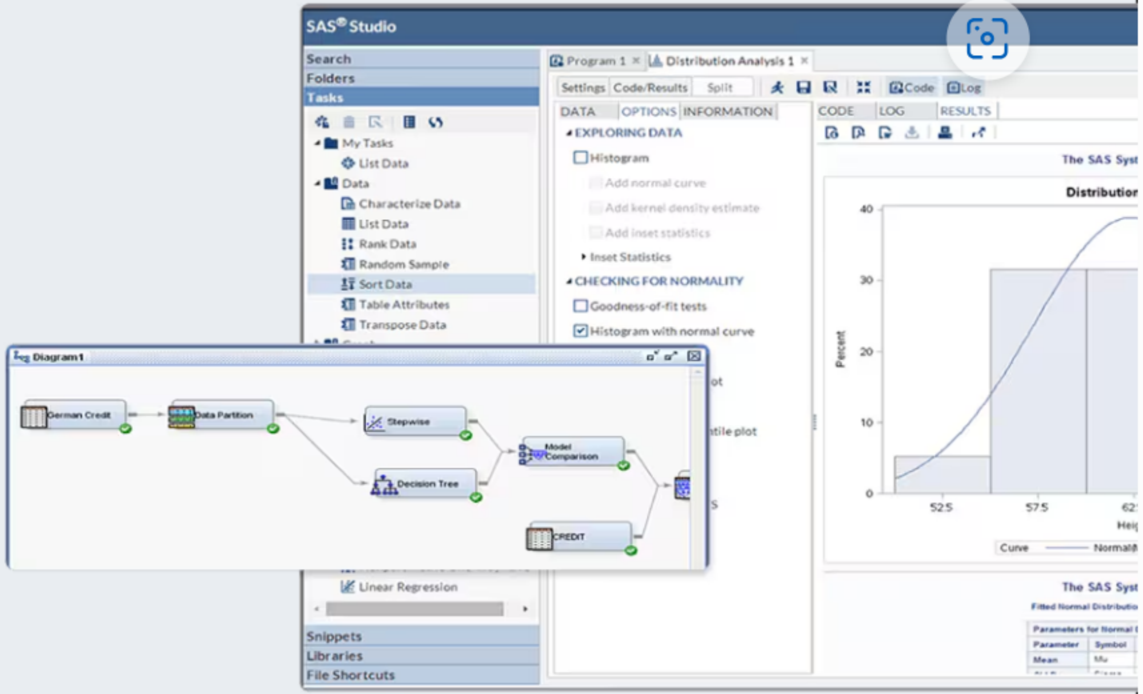
VISIT

SAS® OnDemand for Academics

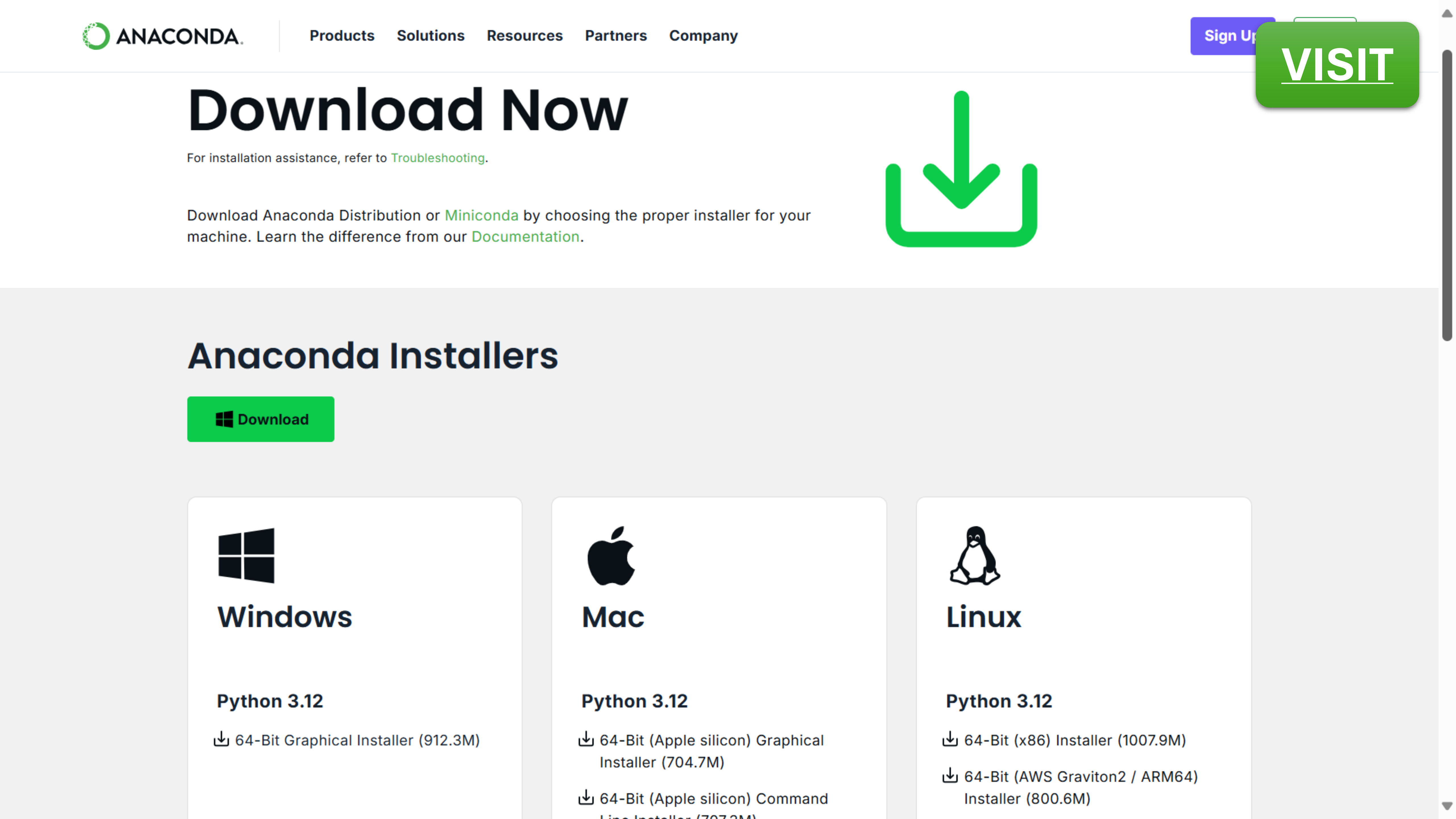
Overview Features List Learn & Support

Access SAS software in the cloud – for free

Access now



SAS OnDemand for Academics



Download Now

For installation assistance, refer to Troubleshooting.

Download Anaconda Distribution or Miniconda by choosing the proper installer for your machine. Learn the difference from our Documentation.

Anaconda Installers

Windows

Python 3.12

64-Bit Graphical Installer (912.3M)

Mac

Python 3.12

64-Bit (Apple silicon) Graphical Installer (704.7M)

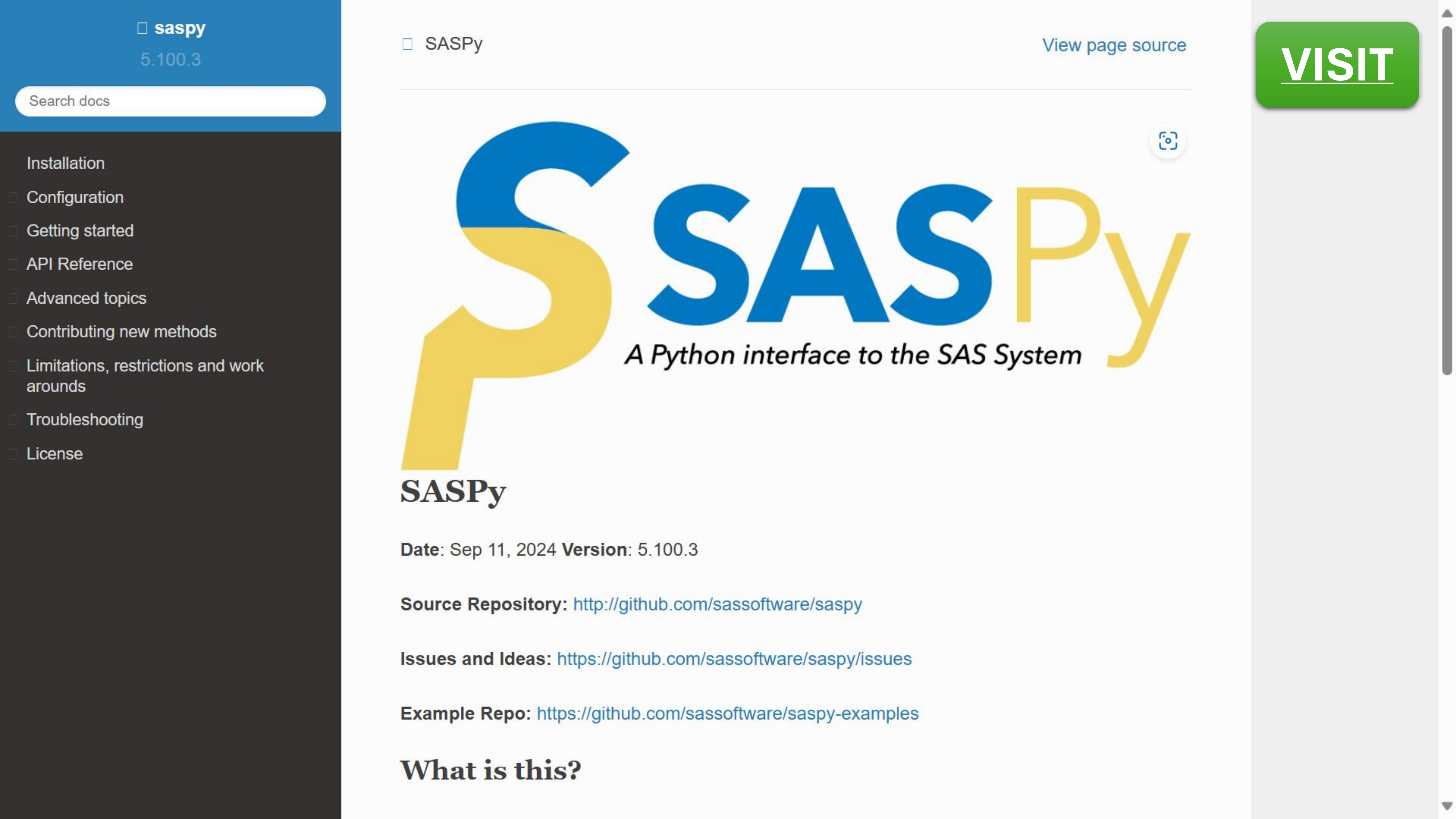
64-Bit (Apple silicon) Command Line Installer (707.3M)

Linux

Python 3.12

64-Bit (x86) Installer (1007.9M)

64-Bit (AWS Graviton2 / ARM64) Installer (800.6M)



VISIT

SASPy

A Python interface to the SAS System

SASPy

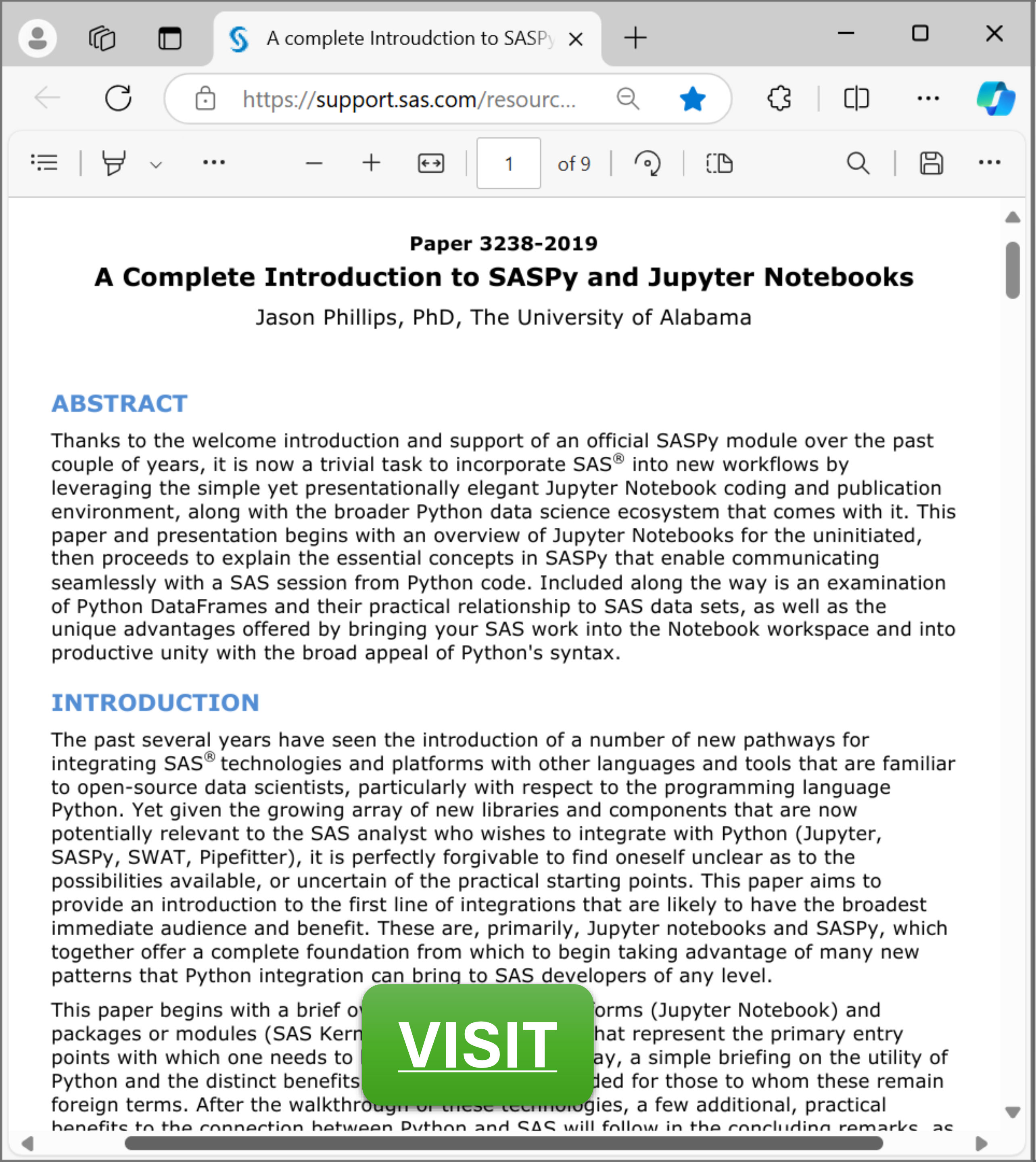
Date: Sep 11, 2024 **Version**: 5.100.3

Source Repository: http://github.com/sassoftware/saspy

Issues and Ideas: https://github.com/sassoftware/saspy/issues

Example Repo: https://github.com/sassoftware/saspy-examples

What is this?


A complete Introudction to SASPy
https://support.sas.com/resourc...
1
of 9
VISIT

Paper 3238-2019

A Complete Introduction to SASPy and Jupyter Notebooks

Jason Phillips, PhD, The University of Alabama

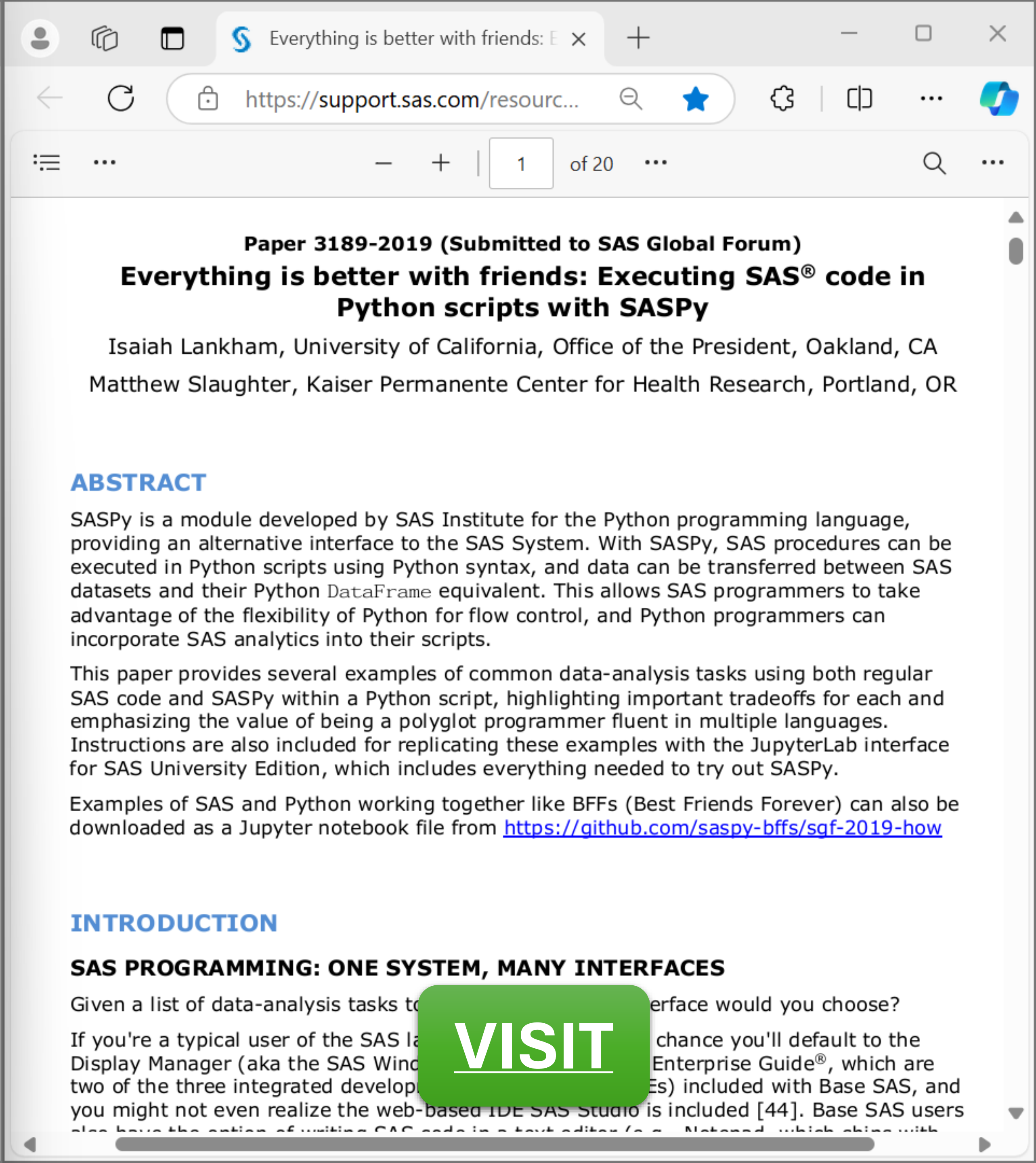
ABSTRACT

Thanks to the welcome introduction and support of an official SASPy module over the past couple of years, it is now a trivial task to incorporate SAS® into new workflows by leveraging the simple yet presentationally elegant Jupyter Notebook coding and publication environment, along with the broader Python data science ecosystem that comes with it. This paper and presentation begins with an overview of Jupyter Notebooks for the uninitiated, then proceeds to explain the essential concepts in SASPy that enable communicating seamlessly with a SAS session from Python code. Included along the way is an examination of Python DataFrames and their practical relationship to SAS data sets, as well as the unique advantages offered by bringing your SAS work into the Notebook workspace and into productive unity with the broad appeal of Python's syntax.

INTRODUCTION

The past several years have seen the introduction of a number of new pathways for integrating SAS® technologies and platforms with other languages and tools that are familiar to open-source data scientists, particularly with respect to the programming language Python. Yet given the growing array of new libraries and components that are now potentially relevant to the SAS analyst who wishes to integrate with Python (Jupyter, SASPy, SWAT, Pipefitter), it is perfectly forgivable to find oneself unclear as to the possibilities available, or uncertain of the practical starting points. This paper aims to provide an introduction to the first line of integrations that are likely to have the broadest immediate audience and benefit. These are, primarily, Jupyter notebooks and SASPy, which together offer a complete foundation from which to begin taking advantage of many new patterns that Python integration can bring to SAS developers of any level.

This paper begins with a brief o orms (Jupyter Notebook) and
packages or modules (SAS Kern hat represent the primary entry
points with which one needs to ay, a simple briefing on the utility of
Python and the distinct benefits ded for those to whom these remain
foreign terms. After the walkthrough of these technologies, a few additional, practical
benefits to the connection between Python and SAS will follow in the concluding remarks, as


Everything is better with friends: E
https://support.sas.com/resourc...
1
of 20
VISIT

Paper 3189-2019 (Submitted to SAS Global Forum)

Everything is better with friends: Executing SAS® code in Python scripts with SASPy

Isaiah Lankham, University of California, Office of the President, Oakland, CA

Matthew Slaughter, Kaiser Permanente Center for Health Research, Portland, OR

ABSTRACT

SASPy is a module developed by SAS Institute for the Python programming language, providing an alternative interface to the SAS System. With SASPy, SAS procedures can be executed in Python scripts using Python syntax, and data can be transferred between SAS datasets and their Python `DataFrame` equivalent. This allows SAS programmers to take advantage of the flexibility of Python for flow control, and Python programmers can incorporate SAS analytics into their scripts.

This paper provides several examples of common data-analysis tasks using both regular SAS code and SASPy within a Python script, highlighting important tradeoffs for each and emphasizing the value of being a polyglot programmer fluent in multiple languages. Instructions are also included for replicating these examples with the JupyterLab interface for SAS University Edition, which includes everything needed to try out SASPy.

Examples of SAS and Python working together like BFFs (Best Friends Forever) can also be downloaded as a Jupyter notebook file from https://github.com/saspy-bffs/sgf-2019-how

INTRODUCTION

SAS PROGRAMMING: ONE SYSTEM, MANY INTERFACES

Given a list of data-analysis tasks to erface would you choose?

If you're a typical user of the SAS la chance you'll default to the
Display Manager (aka the SAS Wind Enterprise Guide®, which are
two of the three integrated develop Es) included with Base SAS, and
you might not even realize the web-based IDE SAS Studio is included [44]. Base SAS users
also have the option of writing SAS code in a text editor (e.g., Notepad, which ships with

SASPy Connection Configuration Files

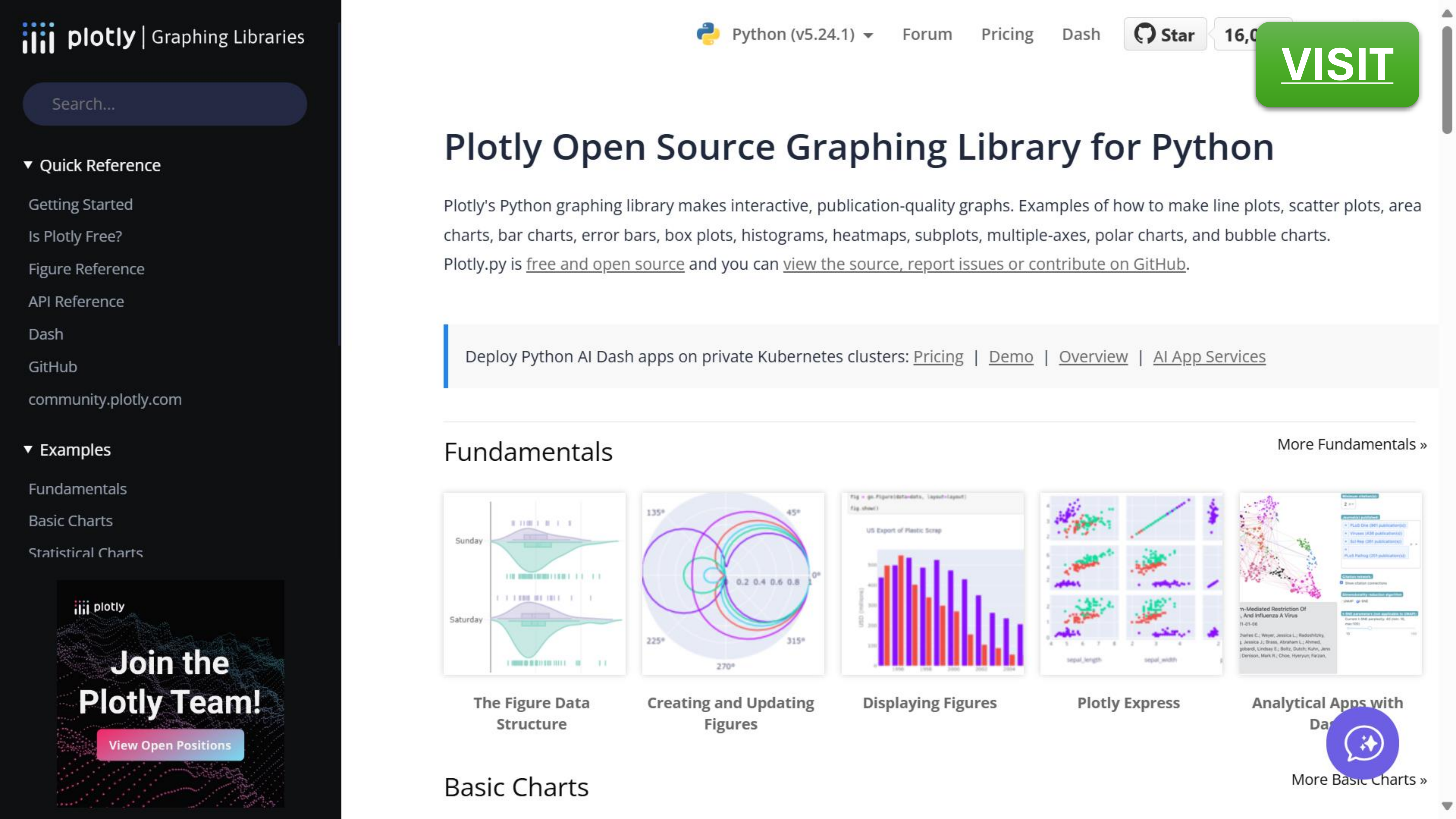
Sample sascfg_personal.py File

```
SAS_config_names=['oda']

oda = {'java' : 'C:\\Program Files (x86)\\Common Files\\Oracle\\Java\\javapath\\java.exe',

'iomhost' : ['odamid-usw2.oda.sas.com','odaws02-usw2.oda.sas.com','odaws03-usw2.oda.sas.com','odaws04-usw2.oda.sas.com'],

'iomport' : 8591,

'authkey' : 'oda',

'encoding' : 'utf-8'

}
```

Sample _authinfo File

```
oda user YourSasID password YourSasPW
```

Note: Connects Windows 11 laptop to SAS OnDemand for Academics (Cloud)



Plotly Open Source Graphing Library for Python

Plotly's Python graphing library makes interactive, publication-quality graphs. Examples of how to make line plots, scatter plots, area charts, bar charts, error bars, box plots, histograms, heatmaps, subplots, multiple-axes, polar charts, and bubble charts.

Plotly.py is free and open source and you can view the source, report issues or contribute on GitHub.

> Deploy Python AI Dash apps on private Kubernetes clusters: Pricing | Demo | Overview | AI App Services

Fundamentals

More Fundamentals »

The Figure Data Structure

Creating and Updating Figures

Displaying Figures

Plotly Express

Analytical Apps with Da

Basic Charts

More Basic Charts »

Visual Studio Code Docs Updates Blog API Extensions FAQ GitHub Copilot Search Docs

VISIT

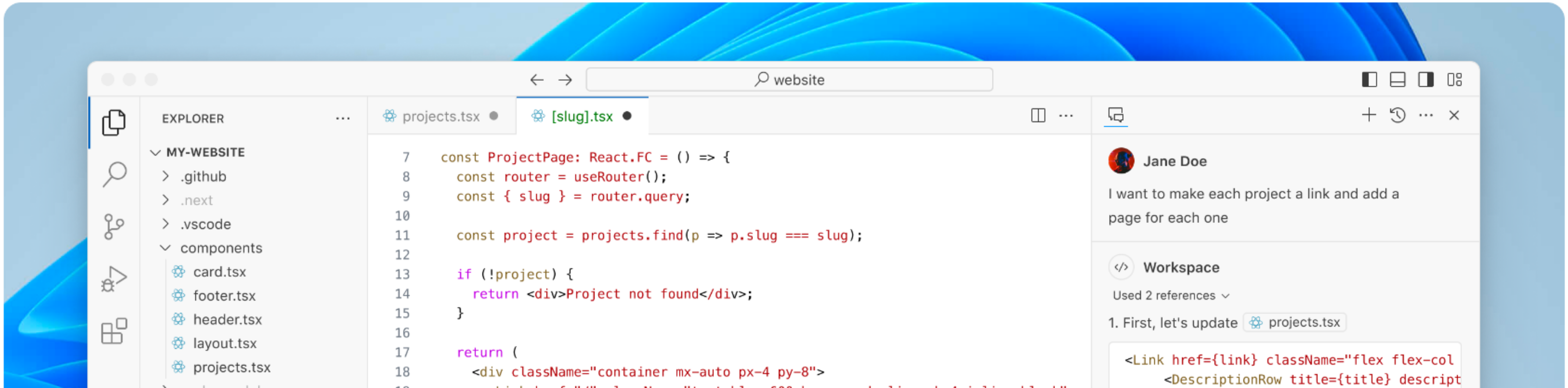
Version 1.94 is now available! Read about the new features and fixes from September.

Code faster with AI

Visual Studio Code with GitHub Copilot supercharges your code with AI-powered suggestions, right in your editor.

Download for Windows

Try GitHub Copilot



```
const ProjectPage: React.FC = () => {
  const router = useRouter();
  const { slug } = router.query;

  const project = projects.find(p => p.slug === slug);

  if (!project) {
    return <div>Project not found</div>;
  }

  return (
    <div className="container mx-auto px-4 py-8">
```

Jane Doe

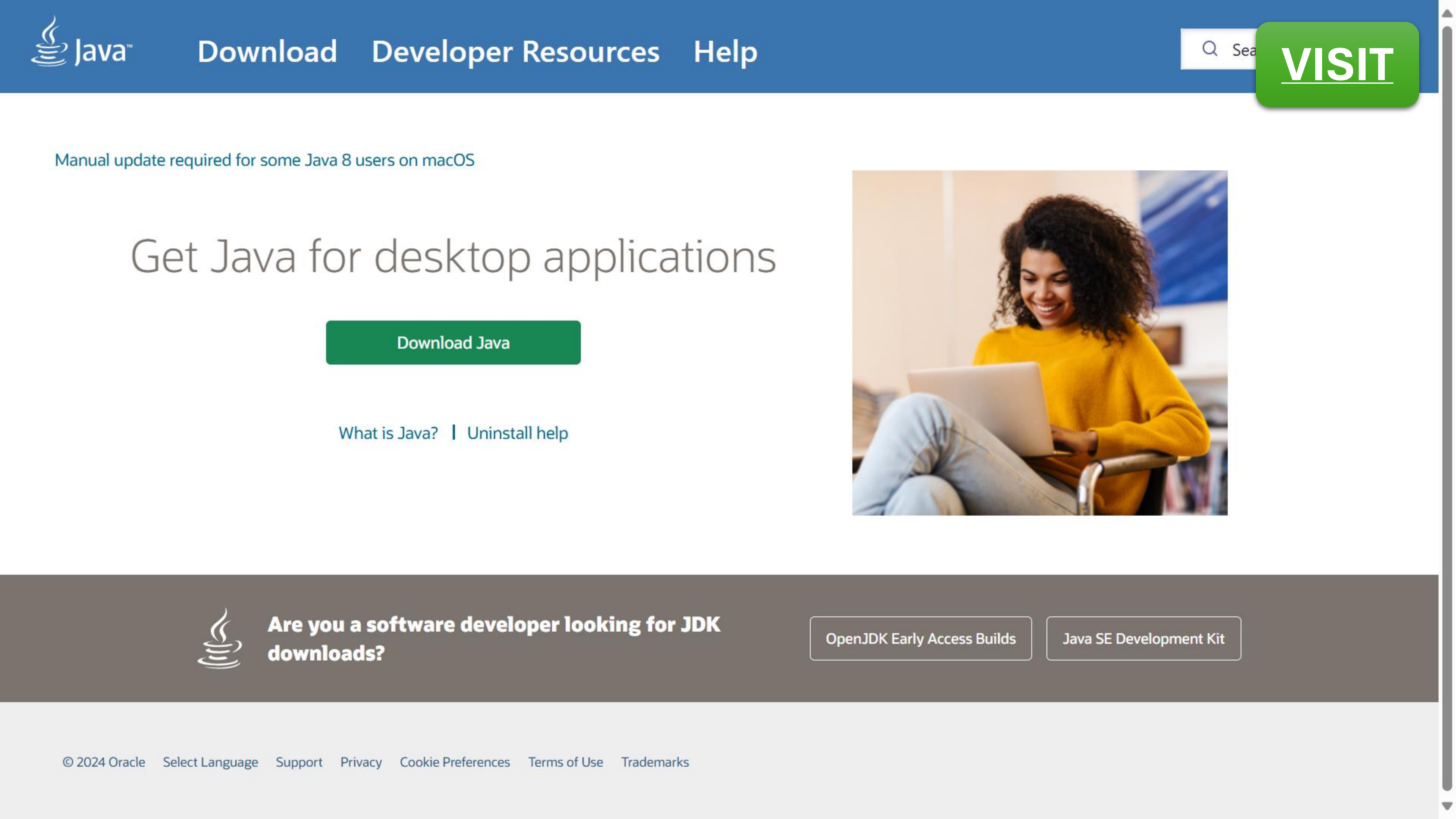
I want to make each project a link and add a page for each one

Workspace

Used 2 references

1. First, let's update projects.tsx

```
<Link href={link} className="flex flex-col
    <DescriptionRow title={title} descript
```



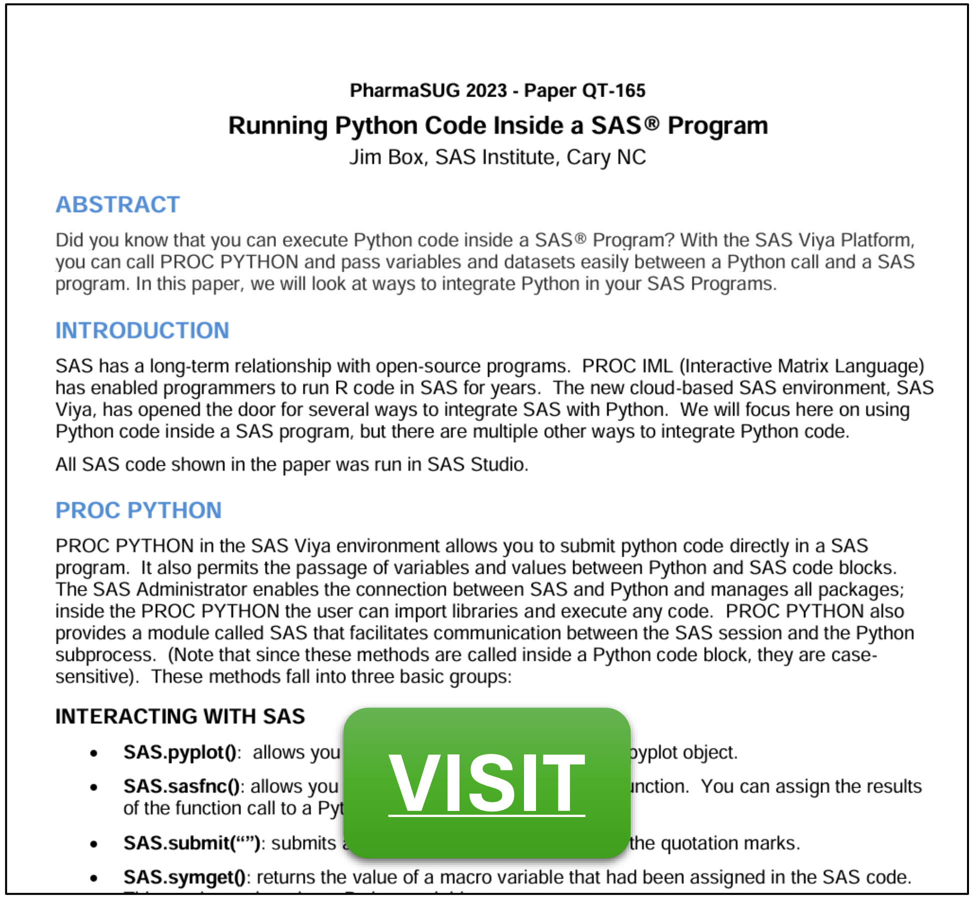
Manual update required for some Java 8 users on macOS

Get Java for desktop applications

Are you a software developer looking for JDK downloads?

© 2024 Oracle Select Language Support Privacy Cookie Preferences Terms of Use Trademarks

Off-Topic But Related: Run Python Code from SAS?



PharmaSUG 2023 - Paper QT-165

Running Python Code Inside a SAS® Program

Jim Box, SAS Institute, Cary NC

ABSTRACT

Did you know that you can execute Python code inside a SAS® Program? With the SAS Viya Platform, you can call PROC PYTHON and pass variables and datasets easily between a Python call and a SAS program. In this paper, we will look at ways to integrate Python in your SAS Programs.

INTRODUCTION

SAS has a long-term relationship with open-source programs. PROC IML (Interactive Matrix Language) has enabled programmers to run R code in SAS for years. The new cloud-based SAS environment, SAS Viya, has opened the door for several ways to integrate SAS with Python. We will focus here on using Python code inside a SAS program, but there are multiple other ways to integrate Python code.

All SAS code shown in the paper was run in SAS Studio.

PROC PYTHON

PROC PYTHON in the SAS Viya environment allows you to submit python code directly in a SAS program. It also permits the passage of variables and values between Python and SAS code blocks. The SAS Administrator enables the connection between SAS and Python and manages all packages; inside the PROC PYTHON the user can import libraries and execute any code. PROC PYTHON also provides a module called SAS that facilitates communication between the SAS session and the Python subprocess. (Note that since these methods are called inside a Python code block, they are case-sensitive). These methods fall into three basic groups:

INTERACTING WITH SAS

- **SAS.pyplot()**: allows you pyplot object.
- **SAS.sasfnc()**: allows you unction. You can assign the results of the function call to a Pyt
- **SAS.submit("")**: submits the quotation marks.
- **SAS.symget()**: returns the value of a macro variable that had been assigned in the SAS code.

VISIT