How to Use the Apache Beam Notebook

Two Different Approaches for Week 5

You have **two options** for completing Week 5:

Option 1: Command-Line Script (`.py` file) ✅ Recommended for Submission

- Use `week5_beam.py` for running pipelines from the command line
- Better for production-style workflows
- Easier to submit as final deliverable
- Run with: `python3 week5_beam.py --input users.csv --output result.txt --task task1`

Option 2: Interactive Notebook (`.ipynb` file) ✅ Best for Learning

- Use the notebook version for interactive exploration
- Test each transformation step-by-step
- See immediate results
- Great for understanding how Beam works

Using the Notebook (`.ipynb`) - Step by Step

Step 1: Open Jupyter Notebook or Google Colab

For Google Colab:

1. Go to colab.research.google.com
2. Click "File" → "New notebook"
3. Copy cells from the code I provided into separate cells

For Local Jupyter:

```bash
pip install jupyter
jupyter notebook
```

Step 2: Structure Your Notebook

Create a new notebook with these cells:

```
📝 CELL 1: Setup & Installation
📝 CELL 2: Import Libraries
📝 CELL 3: Helper Functions
📝 CELL 4: Create/Upload Test Data
📝 CELL 5: Task 1 - Define Transform Class
📝 CELL 6: Task 1 - Test Interactively
📝 CELL 7: Task 1 - Run Full Pipeline
📝 CELL 8: Task 2.1 - Define Classes
📝 CELL 9: Task 2.1 - Test
📝 CELL 10: Task 2.1 - Run Pipeline
... etc
```

Step 3: Upload Your Data Files

In Google Colab:

```python
from google.colab import files
uploaded = files.upload()  # Select users.csv and orders.csv
```

In Local Jupyter:

- Just make sure `users.csv` and `orders.csv` are in the same directory

Step 4: Run Cells One by One

Execute each cell in order by:

- Pressing `Shift + Enter`
- Or clicking the "Play" button

Step 5: View Results Interactively

The notebook lets you see results immediately:

```python
# You'll see output like:
User;Gender;Age;Address;Date joined
Amy Sullivan;female;20;Westlake,OH,44145;2020-08-31
Paige Dixon;female;43;Hicksville,NY,11801;2020-03-22
```

Complete Notebook Example

Here's how to structure your actual `.ipynb` file:

Cell 1: Setup

```python
# Install Apache Beam
!pip install --quiet apache-beam
print("✓ Apache Beam installed")
```

Cell 2: Imports

```python
import apache_beam as beam
from apache_beam.io import ReadFromText, WriteToText
from datetime import datetime
```

Cell 3: Upload Data (Colab only)

```python
from google.colab import files
uploaded = files.upload()
```

Cell 4: Define Task 1 Transform

```python
class FormatUserData(beam.DoFn):
    def process(self, element):
        if element.startswith('User,'):
            yield 'User;Gender;Age;Address;Date joined'
            return

        parts = element.split(',')
        if len(parts) >= 5:
            user = parts[0].strip()
            gender = parts[1].strip().lower()
            age = parts[2].strip()
            address = parts[3].strip().replace('-', ',')
            date_joined = parts[4].strip().replace('/', '-')
            yield f"{user};{gender};{age};{address};{date_joined}"
```

Cell 5: Test Task 1 Interactively

```python
```

```
def myprint(x):
    print(x)
    return x

with beam.Pipeline() as p:
    (p
     | 'Read' >> ReadFromText('users.csv')
     | 'Format' >> beam.ParDo(FormatUserData())
     | 'Print' >> beam.Map(myprint)
    )
```

Cell 6: Run Task 1 Full Pipeline

```
python

with beam.Pipeline() as p:
    (p
     | 'Read' >> ReadFromText('users.csv')
     | 'Format' >> beam.ParDo(FormatUserData())
     | 'Write' >> WriteToText('outputs/marketing_format.txt')
    )
print("✓ Completed!")
```

Cell 7: View Results

```
python

!head outputs/marketing_format.txt-00000-of-00001
```

Repeat similar structure for Tasks 2.1, 2.2, and 2.3

Key Differences: Script vs Notebook

Feature	.py Script	.ipynb Notebook
Execution	Command line	Cell by cell
Testing	Run entire pipeline	Test each step
Visualization	Limited	Immediate output
Debugging	Harder	Easier
Submission	✅ Better	❌ Less common
Learning	Good	✅ Excellent

What to Submit for Week 5

You should submit the `.py` file, not the notebook!

Why?

1. The exercise asks for command-line execution
2. Scripts are standard for production pipelines
3. Easier for instructors to run and grade
4. Industry standard practice

But use the notebook to:

- ✅ Learn how transformations work
- ✅ Debug your code
- ✅ Test small examples
- ✅ Understand Beam concepts

Then transfer to the `.py` file for submission:

1. Develop and test in notebook
2. Copy working code to `week5_beam.py`
3. Test the script from command line
4. Submit the `.py` file

Converting Notebook Code to Script

If you develop in the notebook, here's how to convert to the script:

Notebook version (Cell):

```python
with beam.Pipeline() as p:
    (p
     | 'Read' >> ReadFromText('users.csv')
     | 'Format' >> beam.ParDo(FormatUserData())
     | 'Write' >> WriteToText('outputs/marketing_format.txt')
    )
```

Script version:

```python
```

```
def run_task1(input_file, output_file):
    pipeline_options = PipelineOptions()
    with beam.Pipeline(options=pipeline_options) as p:
        (p
         | 'Read' >> ReadFromText(input_file)
         | 'Format' >> beam.ParDo(FormatUserData())
         | 'Write' >> WriteToText(output_file))
```

Quick Start Commands

For Notebook (Colab/Jupyter):

```python
# Run this in the first cell
!pip install apache-beam
import apache_beam as beam
# Then copy remaining cells from my code
```

For Script (Command Line):

```bash
# Install
pip install apache-beam

# Run Task 1
python3 week5_beam.py --input users.csv --output outputs/marketing_format.txt --task task1

# Run Task 2.1
python3 week5_beam.py --input users.csv --output outputs/gender_totals.txt --task task2_gender

# Run Task 2.2
python3 week5_beam.py --input users.csv --output outputs/customer_totals.txt --task task2_dates

# Run Task 2.3
python3 week5_beam.py --input users.csv --output outputs/state_totals.txt --task task2_states
```

Recommendation

Best Workflow:

1. **Learn & Develop** → Use Notebook (`.ipynb`)
   - Copy my notebook code into Jupyter/Colab

- Run each cell to understand transformations
- Test with small sample data

2. **Finalize & Submit** → Use Script (`.py`)
   - Copy my script code into `week5_beam.py`
   - Test from command line
   - Submit the `.py` file

This gives you the best of both worlds! 🎯