**How to Install and Run the Python cuDF library on the RCS Grid**

Harry Levinson, HBS IT, 4/19/2024

# Introduction

**cuDF** or cudf is a DataFrame library, an alternative to using panda operations or iterative code to process many rows of data. It is much faster than many standard panda operations.

These instructions assume you are using the RCS Grid where you may have access to a GPU compute node. This will download some packages managed by RAPIDS.ai. For more info see:

<https://docs.rapids.ai/>

# Pre-Requisites

* server or compute node with a GPU
* about 4 GB free disk space, wherever you store your conda environments
* access to internet to download conda packages
* some hands-on experience with **conda** and **mamba**, and Python 3.x

# Versions Installed & Tested

cudf 24.04

python 3.10

mamba 1.5.8

conda 24.3.0 (inside virtual environment)

# Installation

Here we install **mamba** in a new virtual environment, and use that where we can, to avoid any potentially older base conda environments on the server. Also, we use "less" to page the output, but you can use "more" instead or redirect to a file.

**1. Check that some version of conda is installed** – if it is not you will need to install it or find a machine or environment with a GPU that has it installed.

which conda

**2. Create and activate a new conda environment** – here we name it "cuda" but just substitute your own name starting here if you want to call it something else:

conda create -n cuda python

conda activate cuda

You should see your shell prompt change to reflect the name of your environment

**3. Install mamba inside the conda environment.** This will make installation and troubleshooting of package conflicts faster.

conda install -c conda-forge mamba

# Verify it is in your virtual environment

which mamba

mamba --version

**4. Install Python** in your virtual environment

conda install python=3.10

mamba list | grep python

**5. Install a recent version of cuDF** and supporting libraries, without specifying particular version numbers. This is less likely to cause problems with the installation solver.

conda config --env --add channels rapidsai

mamba install -c rapidsai -c nvidia -c conda-forge cudf

# Eyeball the results

mamba list | less

**6. Verify correct Python version** once more

which python

python -V # expect 3.10.x

**7. Quick smoke test** to see it working:

**(cuda) [jharvard@rhrcsnodxx ~ ] $ python**

Python 3.10.14 | packaged by conda-forge | (main, Mar 20 2024, 12:45:18) [GCC 12.3.0] on linux

Type "help", "copyright", "credits" or "license" for more information.

>>> import cudf

>>> print(cudf.\_\_version\_\_)

24.04.00

[CTRL+D to exit]

**8. Exit** your environment:

conda deactivate

# Sample Test Code (courtesy of Claude.ai)

Remember you need a GPU and a working environment for this to run!

Program output is shown in **bold red**

**$ conda activate cuda # use your own environment name here**

**(cuda) $ python**

import cudf

**# Create a sample DataFrame**

data = {

'name': ['John', 'Emma', 'Alex', 'Sophia'],

'age': [25, 30, 35, 28],

'city': ['New York', 'London', 'Paris', 'Tokyo']

}

df = cudf.DataFrame(data)

**# Display the DataFrame**

print("Original DataFrame:")

print(df)

**# EXPECTED OUTPUT**

**Original DataFrame:**

**name age city**

**0 John 25 New York**

**1 Emma 30 London**

**2 Alex 35 Paris**

**3 Sophia 28 Tokyo**

**# Perform a simple operation (e.g., filtering)**

filtered\_df = df[df['age'] > 30]

print("\nFiltered DataFrame:")

print(filtered\_df)

**# Perform an aggregation operation**

avg\_age = df['age'].mean()

print(f"\nAverage age: {avg\_age}")

**# EXPECTED OUTPUT**

**Filtered DataFrame:**

**name age city**

**2 Alex 35 Paris**

**Average age: 29.5**