Module 6 Cheat Sheet: Monitoring and Tuning

Package/Method	Description	Code Example

agg()	Used to get the aggregate values like count, sum, avg, min, and max for each group.	`1. 1`
		`1. agg_df = df.groupBy("column_name").agg({"column_to_aggrega`
		Copied!

cache()	Apache Spark transformation that is often used on a DataFrame, data set, or RDD when you want to perform more than one action. cache() caches the specified DataFrame, data set, or RDD in the memory of your cluster's workers. Since cache() is a transformation, the caching operation takes place only when a Spark action (for example, count(), show(), take(), or write()) is also used on the same DataFrame, Dataset, or RDD in a single action.	`1. 1`
		`2. 2`
		`1. df = spark.read.csv("customer.csv")`
		`2. df.cache()`
		Copied!

cd	Used to move efficiently from the existing working	Basic syntax of the cd command:
		`1. 1`
		`1. cd [options]… [directory]`

Package/Method	Description	Code Example
	directory to different directories on your system.	Copied! Example 1: Change directory location to folder1. 1. 1 1. cd /usr/local/folder1 Copied! Example 2: Get back to the previous working directory. 1. 1 1. cd - Copied! Example 3: Move up one level from the present working directory tree. 1. 1 1. cd .. Copied!
def	Used to define a function. It is placed before a function name that is provided by the user to create a user-defined function.	1. 1 1. def greet(name): Copied! This function takes a name as a parameter and prints a greeting. 1. 1 1. print(f"Hello, {name}!") Copied! Calling the function: 1. 1 1. greet("John") Copied!
docker exec	Runs a new command in a running container. Only runs while the container's primary process is running, and it is not restarted if the container is restarted.	1. 1 2. 2 1. docker exec -it container_name command_to_run 2. docker exec -it my_container /bin/bash Copied!
docker rm	Used to remove one or	To remove a single container by name or ID: 1. 1

Package/Method	Description	Code Example
	more containers.	1. docker rm container_name_or_id Copied! To remove multiple containers by specifying their names or IDs: 1. 1 1. docker rm container1_name_or_id container2_name_or_id Copied! To remove all stopped containers: 1. 1 1. docker rm $(docker ps -aq) Copied!
docker run	It runs a command in a new container, getting the image and starting the container if needed.	1. 1 1. docker run [OPTIONS] IMAGE [COMMAND] [ARG...] Copied!
for	The *for* loop operates on lists of items. It repeats a set of commands for every item in a list.	1. 1 1. fruits = ["apple", "banana", "cherry"] Copied! Iterating through the list using a *for* loop for fruit in fruits: 1. 1 1. print(f"I like {fruit}s") Copied!
groupby()	Used to collect the identical data into groups on DataFrame and perform count, sum, avg, min, max functions on the grouped data.	1. 1 1. import pandas as pd Copied! Sample DataFrame: 1. 1 2. 2 3. 3 1. data = {'Category': ['A', 'B', 'A', 'B', 'A', 'B'], 2. 'Value': [10, 20, 15, 25, 30, 35]} 3. df = pd.DataFrame(data) Copied! Grouping by "Category" and performing aggregation operations: 1. 1 2. 2

Package/Method	Description	Code Example
		1. grouped = df.groupby('Category').agg({'Value': ['count', ' 2. print(grouped) Copied! Create a sample DataFrame: 1. 1 2. 2 3. 3 1. data = [("John", 25), ("Peter", 30), ("Julie", 35), ("Davi 2. columns = ["Name", "Age"] 3. df = spark.createDataFrame(data, columns) Copied! Show the current number of partitions. 1. 1 1. print("Number of partitions before repartitioning: ", df.r Copied! Repartition the DataFrame to 2 partitions. 1. 1 1. df_repartitioned = df.repartition(2) Copied! Show the number of partitions after repartitioning. 1. 1 1. print("Number of partitions after repartitioning: ", df_re Copied! Stop the SparkSession. 1. 1 1. spark.stop() Copied!
repartition()	Used to increase or decrease the RDD or DataFrame partitions by number of partitions or by a single column name or multiple column names.	
return	Used to end the execution of the function call and returns the result (value of the expression following the return keyword) to the caller.	1. 1 2. 2 3. 3 1. def add_numbers(a, b): 2. result = a + b 3. return result Copied! Calling the function and capturing the returned value: 1. 1 1. sum_result = add_numbers(5, 6) Copied!

Package/Method	Description	Code Example
		Printing the result. 1. 1 1. print("The sum is:", sum_result) Copied! Output. 1. 1 1. The sum is: 11 Copied!
show()	Spark DataFrame show() is used to display the contents of the DataFrame in a table row and column format. By default, it shows only 20 rows, and the column values are truncated at 20 characters.	1. 1 1. df.show() Copied!
spark.read.csv("path")	Using this, you can read a CSV file with fields delimited by pipe, comma, tab (and many more) into a Spark DataFrame.	1. 1 1. from pyspark.sql import SparkSession Copied! Create a SparkSession. 1. 1 1. spark = SparkSession.builder.appName("CSVReadExample").get Copied! Read a CSV file into a Spark DataFrame. 1. 1 1. df = spark.read.csv("path_to_csv_file.csv", header=True, i Copied! Show the first few rows of the DataFrame. 1. 1 1. df.show() Copied! Stop the SparkSession. 1. 1

Package/Method	Description	Code Example
		1. spark.stop() Copied! Basic syntax of the wget command; commonly used options are [-V], 1. 1 1. wget [options]… [URL]… Copied! Example 1: Specifies to download file.txt over HTTP website URL intc 1. 1 1. wget http://example.com/file.txt Copied! Example 2: Specifies to download the archive.zip over HTTP website U prompt in the interim. 1. 1 1. wget -b http://www.example.org/files/archive.zip Copied!
wget	Stands for web get. The wget is a free noninteractive file downloader command. Noninteractive means that it can work in the background when the user is not logged in.	
withColumn()	Transformation function of DataFrame which is used to change the value, convert the datatype of an existing column, create a new column, and many more.	Sample DataFrame: 1. 1 2. 2 3. 3 1. data = [("John", 25), ("Peter", 30), ("David", 35)] 2. columns = ["Name", "Age"] 3. df = spark.createDataFrame(data, columns) Copied! Using withColumn to create a new column and change values 1. 1 2. 2 3. 3 4. 4 5. 5 6. 6 7. 7 1. updated_df = df \ 2. .withColumn("DoubleAge", col("Age") * 2) # Create a n 3. updated_df = updated_df \ 4. .withColumn("AgeGroup", when(col("Age") <= 30, "Young" 5. .when((col("Age") > 30) & (col("Age") <= 4 6. .otherwise("Old")) # Create a new column 7. updated_df.show() Copied! Stop the SparkSession. 1. 1 1. spark.stop()

Package/Method	Description	Code Example

Copied!

Changelog

Date	Version	Changed by	Change Description
2023-09-20	2.0	Kunal Merchant	QC reviewed
2023-09-18	1.0	Sameeksha Saxena	Initial version created