# Module 3 Cheat Sheet: Apache Spark

| Package/Method | Description | Code Example |
|---|---|---|
| appName() | A name for your job to display on the cluster web UI. | `1. 1`<br>`2. 2`<br><br>`1. from pyspark.sql import SparkSession`<br>`2. spark = SparkSession.builder.appName("MyApp").getOrCreate()`<br>Copied! |
| cache() | An Apache Spark transformation often used on a DataFrame, data set, or RDD when you want to perform multiple actions. 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, data set, or RDD in a single action. | `1. 1`<br>`2. 2`<br><br>`1. df = spark.read.csv("customer.csv")`<br>`2. df.cache()`<br>Copied! |
| count() | Returns the number of elements with | `1. 1`<br>`2. 2`<br><br>`1. count = df.count()` |

| | | |
|---|---|---|
| | the specified value. | 2. print(count)<br>Copied! |
| createTempView() | Creates a temporary view that can later be used to query the data. The only required parameter is the name of the view. | 1. 1<br>1. df.createOrReplaceTempView("cust_tbl")<br>Copied! |
| filter() | Returns an iterator where the items are filtered through a function to test if the item is accepted or not. | 1. 1<br>1. filtered_df = df.filter(df['age'] > 30)<br>Copied! |
| getOrCreate() | Get or instantiate a SparkContext and register it as a singleton object. | 1. 1<br>1. spark = SparkSession.builder.getOrCreate()<br>Copied! |
| import | Used to make code from one module accessible in another. Python imports are crucial for a successful code structure. You may reuse code and keep your projects manageable by using imports effectively, which can increase your productivity. | 1. 1<br>1. from pyspark.sql import SparkSession<br>Copied! |
| len() | Returns the number of items in an object. When the object is a string, the len() function | 1. 1<br>2. 2<br>1. row_count = len(df.collect())<br>2. print(row_count)<br>Copied! |

| | |
|---|---|
| | returns the number of characters in the string. |
| map() | Returns a map object (an iterator) of the results after applying the given function to each item of a given iterable (list, tuple, etc.)<br>1. 1<br>2. 2<br>1. rdd = df.rdd.map(lambda row: (row['name'],<br>2. row['age']))<br>Copied! |
| pip | To ensure that requests will function, the pip program searches for the package in the Python Package Index (PyPI), resolves any dependencies, and installs everything in your current Python environment.<br>1. 1<br>1. pip list<br>Copied! |
| pip install | The pip install <package> command looks for the latest version of the package and installs it.<br>1. 1<br>1. pip install pyspark<br>Copied! |
| print() | Prints the specified message to the screen or other standard output device. The message can be a string or any other object; the object will be converted into a string before being written to the screen.<br>1. 1<br>1. print("Hello, PySpark!")<br>Copied! |
| printSchema() | Used to print or display the<br>1. 1<br>1. df.printSchema() |

| | | |
|---|---|---|
| | schema of the DataFrame or data set in tree format along with the column name and data type. If you have a DataFrame or data set with a nested structure, it displays the schema in a nested tree format. | Copied! |
| sc.parallelize() | Creates a parallelized collection. Distributes a local Python collection to form an RDD. Using range is recommended if the input represents a range for performance. | 1. 1<br>1. rdd = sc.parallelize([1, 2, 3, 4, 5])<br>Copied! |
| select() | Used to select one or multiple columns, nested columns, column by index, all columns from the list, by regular expression from a DataFrame. select() is a transformation function in Spark and returns a new DataFrame with the selected columns. | 1. 1<br>1. selected_df = df.select('name', 'age')<br>Copied! |
| show() | Spark DataFrame show() is used to display the | 1. 1<br>1. df.show() |

| | |
|---|---|
| | contents of the DataFrame in a table row and column format . By default, it shows only twenty rows, and the column values are truncated at twenty characters. |
| spark.read.json | Spark SQL can automatically infer the schema of a JSON data set and load it as a DataFrame. The read.json() function loads data from a directory of JSON files where each line of the files is a JSON object. Note that the file offered as a JSON file is not a typical JSON file. |
| spark.sql() | To issue any SQL query, use the sql() method on the SparkSession instance . All spark.sql queries executed in this manner return a DataFrame on which you may perform further Spark operations if required. |
| time() | Returns the current time in the number of seconds since |

Copied!

```
1
```

```
json_df = spark.read.json("customer.json")
```

Copied!

```
1
2
```

```
result = spark.sql("SELECT name, age FROM cust_tbl WHERE age > 30")
result.show()
```

Copied!

```
1
2
3
```

```
from pyspark.sql.functions import current_timestamp
current_time = df.select(current_timestamp().alias("current_time"))
```

the Unix Epoch.

```
3. current_time.show()
```

Copied!

# Changelog

| Date | Version | Changed by | Change Description |
|---|---|---|---|
| 2023-09-06 | 1.0 | Sameeksha Saxena | Initial version created |