

Java-Success.com

Prepare to fast-track, choose & go places with 800+ Java & Big Data Q&As with lots of code & diagrams.

[Home](#) [Why? ▾](#) [300+ Java FAQs ▾](#) [300+ Big Data FAQs ▾](#) [Courses ▾](#)

[👤 Membership ▾](#) [Your Career ▾](#)

[Home](#) › [bigdata-success.com](#) › [Tutorials - Big Data](#) › [TUT - PySpark on Databricks](#) ›

01: Databricks getting started – Spark, Shell, SQL

01: Databricks getting started – Spark, Shell, SQL

 Posted on [April 10, 2020](#)

Step 1: Signup to Databricks community edition – <https://databricks.com/try-databricks>. Fill in the details and you can leave your mobile number blank. Select “COMMUNITY EDITION” ==> “GET STARTED”.

300+ Java Interview FAQs

300+ Java FAQs



16+ Java Key Areas Q&As



150+ Java Architect FAQs



80+ Java Code Quality Q&As



150+ Java Coding Q&As



300+ Big Data Interview FAQs

300+ Big Data FAQs



Tutorials - Big Data



TUT -  Starting Big Data



TUT - Starting Spark & Scala

DATABRICKS PLATFORM - FREE TRIAL

For businesses looking for a zero-management cloud platform built around Apache Spark

- Unlimited clusters that can scale to any size
- Job scheduler to execute jobs for production pipelines
- Fully interactive notebook with collaboration, dashboards, REST APIs
- Advanced security, role-based access controls, and audit logs
- Single Sign On support
- Integration with BI tools such as Tableau, Qlik, and Looker
- 14-day full feature trial (excludes cloud charges)

GET STARTED ON

 OR 

Please note that Azure Databricks is provided by Microsoft and is subject to Microsoft's terms.

By clicking on the "AWS" button to get started, you agree to the [Databricks Terms of Service](#).

COMMUNITY EDITION

For students and educational institutions just getting started with Apache Spark

- Single cluster limited to 6GB and no worker nodes
- Basic notebook without collaboration
- Limited to 3 max users
- Public environment to share your work

GET STARTED

By clicking "Get Started" for the Community Edition, you agree to the [Databricks Community Edition Terms of Service](#).

Databricks getting started

If you have a Cloud account then you can use it.

Step 2: Check your email and click the “link” in the email & reset your password.

Step 3: Login to Databricks notebook:

<https://community.cloud.databricks.com/login.html>.

The image shows the Databricks 'Getting started' dashboard. At the top, there's a navigation bar with the Databricks logo, a 'Home' button, and a 'Jobs' button. Below this, the main heading is 'Welcome to databricks'. The dashboard is divided into three main sections: 'Explore the Quickstart Tutorial', 'Import & Explore Data', and 'Create a Blank Notebook'. Each section has a brief description and a 'Get started' button. The 'Explore the Quickstart Tutorial' section mentions spinning up a cluster and running queries. The 'Import & Explore Data' section mentions quickly importing data and previewing its schema. The 'Create a Blank Notebook' section mentions creating a notebook to start querying, visualizing, and modeling data. On the left side, there's a sidebar with icons for 'Data', 'Clusters', 'Jobs', and 'Search'. At the bottom, there's a 'Common Tasks' section with links to 'New Notebook', 'Create Table', 'New Cluster', 'New Job', 'New MLflow Experiment', 'Import Library', and 'Read Documentation'. On the right side, there's a 'Recents' section showing 'Recent files appear here as you work.' and a 'What's new in v3.17' section with a link to 'View latest release notes'.

Step 4: Create a **CLUSTER** and it will take a few minutes to come up. This cluster will go down after 2 hours.

TUT - Starting with Python

TUT - Kafka

TUT - Pig

TUT - Apache Storm

TUT - Spark Scala on Zeppelin

TUT - Cloudera

TUT - Cloudera on Docker

TUT - File Formats

TUT - Spark on Docker

TUT - Flume

TUT - Hadoop (HDFS)

TUT - HBase (NoSQL)

TUT - Hive (SQL)

TUT - Hadoop & Spark

TUT - MapReduce


TUT - Spark and Scala

TUT - Spark & Java

TUT - PySpark on Databricks

TUT - Zookeeper


800+ Java Interview Q&As

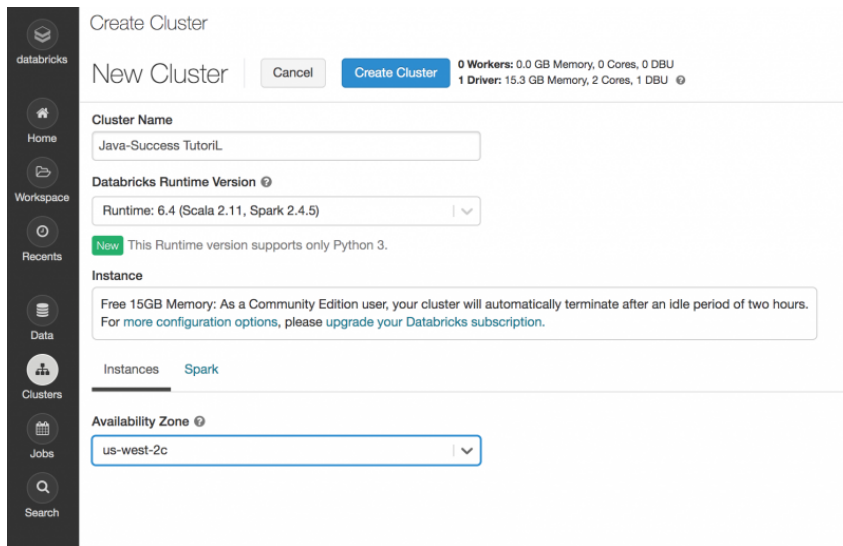
300+ Core Java Q&As 

300+ Enterprise Java Q&As 

150+ Java Frameworks Q&As 

120+ Companion
Tech Q&As

Tutorials -
Enterprise Java 

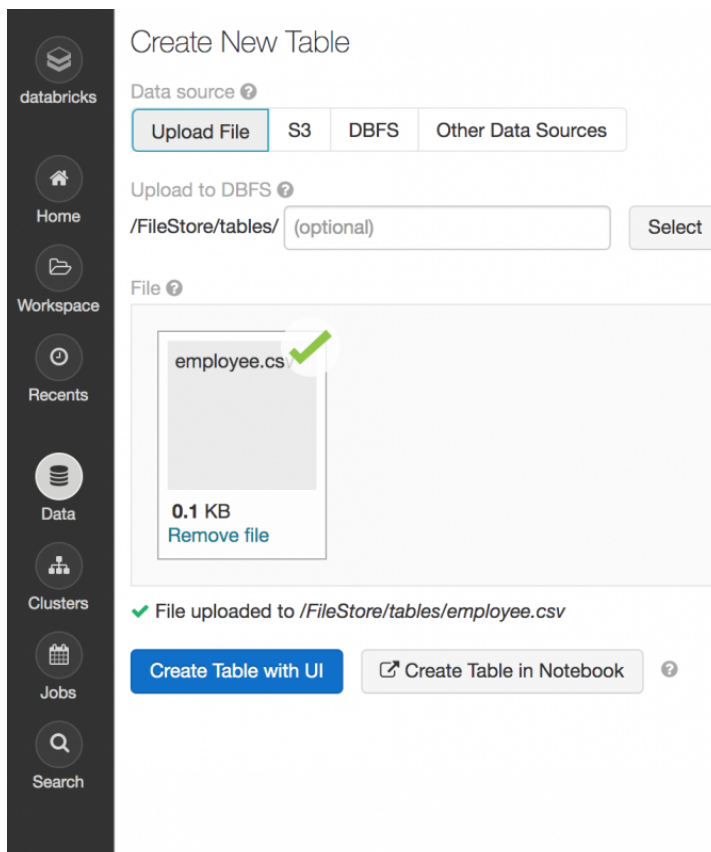


The screenshot shows the 'Create Cluster' page in Databricks. On the left is a sidebar with navigation icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area is titled 'Create Cluster' and 'New Cluster'. It includes a 'Cluster Name' field with 'Java-Success TutoriL', a 'Databricks Runtime Version' dropdown set to 'Runtime: 6.4 (Scala 2.11, Spark 2.4.5)', and an 'Instance' section with a warning about free memory. Below these is an 'Availability Zone' dropdown set to 'us-west-2c'. At the top right, it shows '0 Workers: 0.0 GB Memory, 0 Cores, 0 DBU' and '1 Driver: 15.3 GB Memory, 2 Cores, 1 DBU'.

Databricks create a cluster

Step 5: Select “DATA”, and upload a file named “employee.csv”.

```
1 emp_id,emp_name,emp_city,emp_salary
2 1, John, Sydney, 35000.00
3 2, Peter, Melbourne, 45000.00
4 3, Sam, Sydney,55000.00
5
```



The screenshot shows the 'Create New Table' page in Databricks. The 'Data source' section has 'Upload File' selected. The 'Upload to DBFS' section shows the path '/FileStore/tables/' and a 'Select' button. The 'File' section shows a file named 'employee.csv' (0.1 KB) with a green checkmark and a 'Remove file' link. At the bottom, there are two buttons: 'Create Table with UI' and 'Create Table in Notebook'.

Databricks upload a .csv

Step 6: “Create Table With UI” as shown below:

Note: Please check the “First row is header” check box on the LHS so that column names appear from the file.

Create New Table

Select a Cluster to Preview the Table

Choose a cluster with which you will read and preview the data.

Cluster Java-Success TutorialL

[Preview Table](#)

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name employee_csv

Create in Database default

File Type CSV

Column Delimiter ,

☒ First row is header

☐ Infer schema

☐ Multi-line

[Create Table](#)

Table Preview

emp_id	emp_name	emp_city	emp_salary
1	John	Sydney	35000.00
2	Peter	Melbourne	45000.00
3	Sam	Sydney	55000.00

Databricks create a table with .csv

Click on “Create Table”.

Step 7: Click on the “databricks” icon on the LHS menu, and then “Create a Blank Notebook”.

Databricks blank notebook

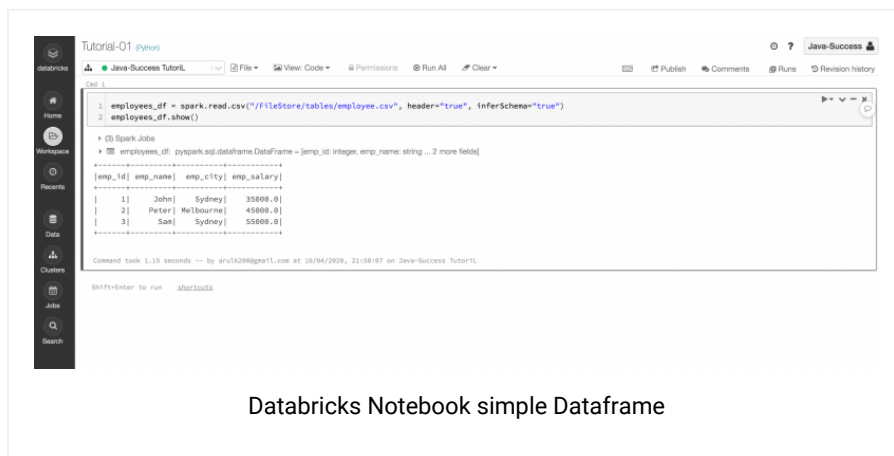
Spark in Python (i.e.PySpark)

Since we created the notebook as “python”, we don’t have to do “%python” as it is the default language. If you want to use “scala” then add “%scala” as the first line in a cell.

Step 8: Run the below PySpark code to display uploaded “/FileStore/tables/employee.csv” as a **Dataframe**.

```
1 employees_df = spark.read.csv("/FileStore/tables/employee.csv", header="true", inferSchema="true")
2 employees_df.show()
3
```

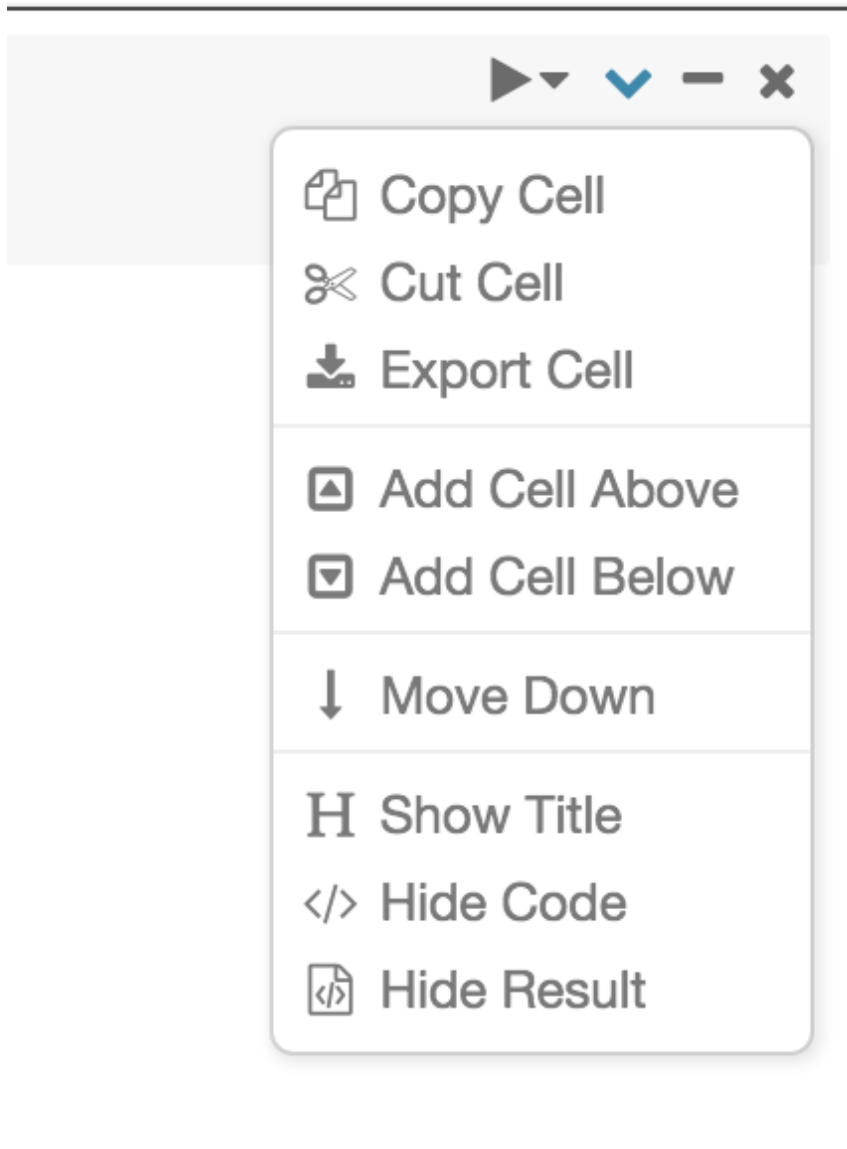
Click on “Run” in the top “RHS” menu.



The screenshot shows a Databricks notebook interface. The top bar includes the notebook name 'Tutorial-01 (python)', a file explorer, and various action buttons like 'Run All', 'Clear', 'Publish', 'Comments', 'Runs', and 'Revision history'. The main area displays a code cell with two lines of Python code: `employees_df = spark.read.csv("/FileStore/tables/employee.csv", header="true", inferSchema="true")` and `employees_df.show()`. Below the code, the output is shown as a table with columns `emp_id`, `emp_name`, `emp_city`, and `emp_salary`. The table contains three rows of data. At the bottom, a status bar indicates the command took 1.15 seconds to run.

emp_id	emp_name	emp_city	emp_salary
1	John	Sydney	35000.0
2	Peter	Melbourne	45000.0
3	Sam	Sydney	55000.0

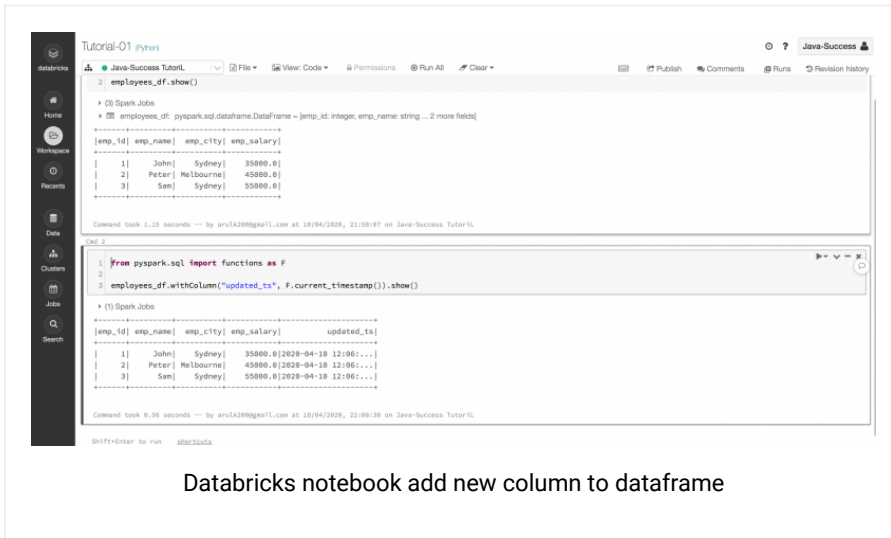
Databricks Notebook simple Dataframe



Use “Down Arrow” on in the RHS of a cell to create a new cell.

Step 9: Add a new column to the Dataframe in a separate cell and then **Run**.

```
1 from pyspark.sql import functions as F
2
3 employees_df.withColumn("updated_ts", F.current_timestamp())
4
```



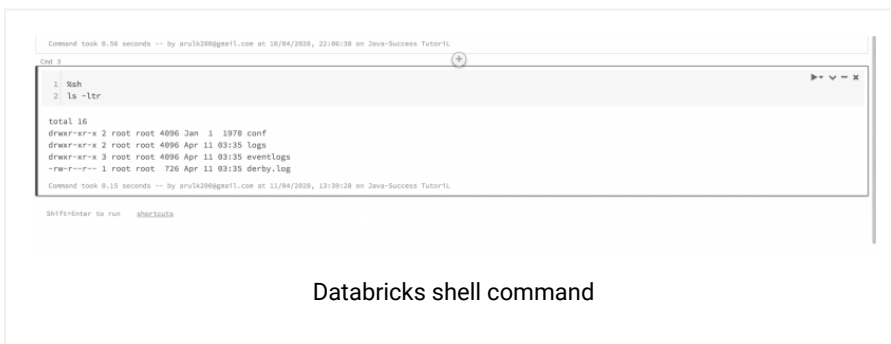
Databricks notebook add new column to dataframe

Run a shell command

You can run a shell command with “%sh” as shown below:

```
1 %sh
2
3 ls -ltr
4
```

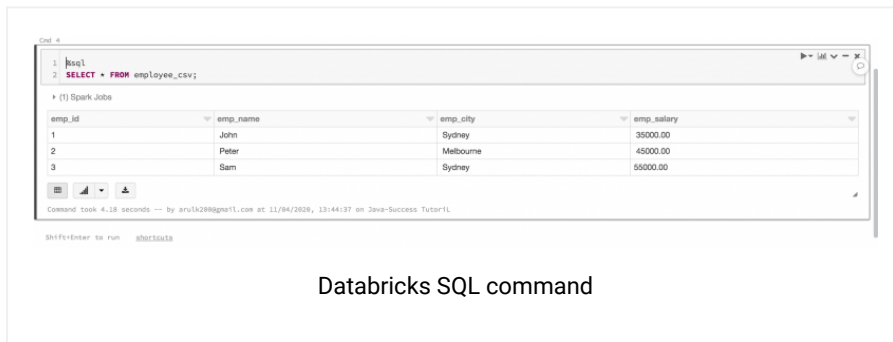
You can click on the “+” in the middle to add new cells.



Databricks shell command

Run a SQL command

```
1 %sql
2
3 SELECT * FROM employee_csv;
4
```



```

1 | %sql
2 | SELECT * FROM employee_csv;

```

emp_id	emp_name	emp_city	emp_salary
1	John	Sydney	35000.00
2	Peter	Melbourne	45000.00
3	Sam	Sydney	55000.00

Command took 4.18 seconds -- by arul2000gn@l.com at 11/04/2020, 13:44:37 on Java-Success Tutorial

Databricks SQL command

dbutils

```

1 | %python
2 |
3 | dbutils.fs.rm("/tmp/my-delta-table", recurse=True)
4 |

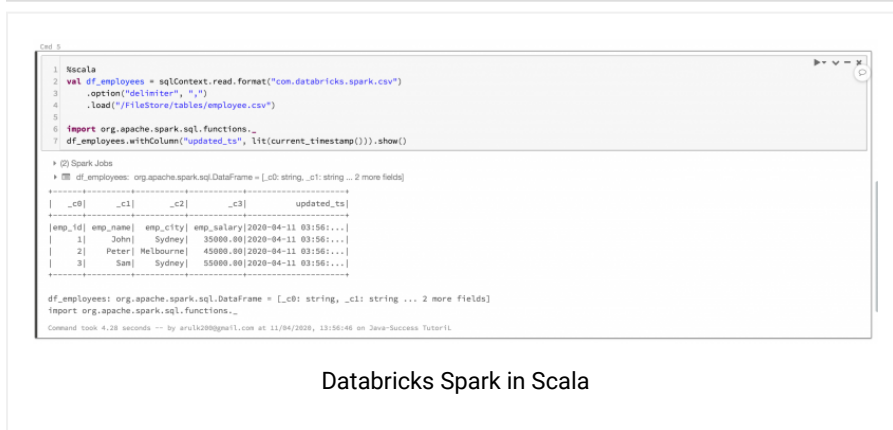
```

Spark in Scala

```

1 | %scala
2 |
3 | val df_employees = sqlContext.read.format("com.databricks.spark.csv")
4 |   .option("delimiter", ",")
5 |   .load("/FileStore/tables/employee.csv")
6 |
7 | import org.apache.spark.sql.functions._
8 | df_employees.withColumn("updated_ts", lit(current_timestamp()))
9 |

```



```

1 | %scala
2 | val df_employees = sqlContext.read.format("com.databricks.spark.csv")
3 |   .option("delimiter", ",")
4 |   .load("/FileStore/tables/employee.csv")
5 |
6 | import org.apache.spark.sql.functions._
7 | df_employees.withColumn("updated_ts", lit(current_timestamp()))
8 |

```

emp_id	emp_name	emp_city	emp_salary	updated_ts
1	John	Sydney	35000.00	2020-04-11 03:56:...
2	Peter	Melbourne	45000.00	2020-04-11 03:56:...
3	Sam	Sydney	55000.00	2020-04-11 03:56:...

df_employees: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 2 more fields]
 import org.apache.spark.sql.functions._
 Command took 4.28 seconds -- by arul2000gn@l.com at 11/04/2020, 13:56:46 on Java-Success Tutorial

Databricks Spark in Scala

Important: PySpark API

Have the PySpark API [PySpark modules](#) handy to code. You can click on “Dataframe” to see what

functions are available. For example, **withColumn** function in the Dataframe module

withColumn(colName, col)

[\[source\]](#)

Returns a new **DataFrame** by adding a column or replacing the existing column that has the same name.

The column expression must be an expression over this **DataFrame**; attempting to add a column from some other **DataFrame** will raise an error.

Parameters:

- **colName** – string, name of the new column.
- **col** – a **Column** expression for the new column.

```
>>> df.withColumn('age2', df.age + 2).collect()
[Row(age=2, name='Alice', age2=4), Row(age=5, name='Bob', age2=7)]
```

New in version 1.3.

Databricks PySpark Modules & API

Where are my notebooks saved?

Your notebooks will be saved under “workspace” ==>
“users” ==> “[your username]”

What if you want to practice in Scala?

You can try the examples from [Tutorials – Spark Scala on Zeppelin](#) with some minor changes.

◀ AWS Web Application Security Q&As

02: Databricks – Spark schemas, casting & PySpark API ▶

Disclaimer

The contents in this Java-Success are copyrighted and from EmpoweringTech pty ltd. The EmpoweringTech pty ltd has the right to correct or enhance the current content without any prior notice. These are general advice only, and one needs to take his/her own circumstances into consideration. The EmpoweringTech pty ltd will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. Links to external sites do not imply endorsement of the linked-to sites. [Privacy Policy](#)