

```

import os
# Find the latest version of spark 3.0 from http://www.apache.org/dist/spark/ and enter as .th
# For example:
# spark_version = 'spark-3.0.3'
spark_version = 'spark-3.1.3'
os.environ['SPARK_VERSION'] = spark_version

# Install Spark and Java
!apt-get update
!apt-get install openjdk-11-jdk-headless -qq > /dev/null
!wget -q http://www.apache.org/dist/spark/\$SPARK\_VERSION/\$SPARK\_VERSION-bin-hadoop2.7.tgz
!tar -xvf $SPARK_VERSION-bin-hadoop2.7.tgz
!pip install -q findspark

# Set Environment Variables
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-11-openjdk-amd64"
os.environ["SPARK_HOME"] = "f/content/{spark_version}-bin-hadoop2.7"

# Start a SparkSession
import findspark
findspark.init()

```

```

Hit:1 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:3 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease [3,626 B]
Get:4 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [83.3 kB]
Hit:6 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease
Ign:7 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86\_64
Hit:8 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86\_64 InRelease
Hit:9 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86\_64
Hit:10 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Hit:11 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease
Hit:12 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [2,332 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [3,472 kB]
Get:16 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [3,040 kB]
Get:17 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [1,554 kB]
Fetched 10.7 MB in 3s (3,153 kB/s)
Reading package lists... Done

```

```

# Download the Postgres driver that will allow Spark to interact with Postgres.
!wget https://jdbc.postgresql.org/download/postgresql-42.2.16.jar

```

```

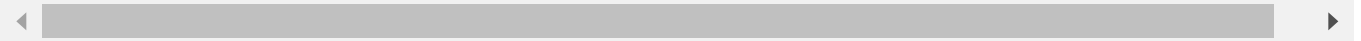
--2022-11-12 04:04:20-- https://jdbc.postgresql.org/download/postgresql-42.2.16.jar
Resolving jdbc.postgresql.org (jdbc.postgresql.org)... 72.32.157.228, 2001:4800:3e1:1::2
Connecting to jdbc.postgresql.org (jdbc.postgresql.org)|72.32.157.228|:443... connected
HTTP request sent, awaiting response... 200 OK
Length: 1002883 (979K) [application/java-archive]

```

Saving to: 'postgresql-42.2.16.jar'

postgresql-42.2.16. 100%[=====>] 979.38K 6.30MB/s in 0.2s

2022-11-12 04:04:20 (6.30 MB/s) - 'postgresql-42.2.16.jar' saved [1002883/1002883]



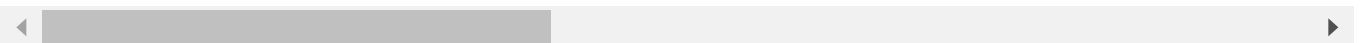
```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("M16-Amazon-Challenge").config("spark.driver.extraClassP
```

## ▼ Load Amazon Data into Spark DataFrame

```
from pyspark import SparkFiles
url = "https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Pet_Products_v1_00.t
spark.sparkContext.addFile(url)
df = spark.read.option("encoding", "UTF-8").csv(SparkFiles.get(""), sep="\t", header=True, in
df.show()
```

marketplace	customer_id	review_id	product_id	product_parent	product_title
US	28794885	REAKC26P07MDN	B00Q0K9604	510387886	(8-Pack) EZwhelp ...
US	11488901	R3NU70MZ4HQIEG	B00MBW509W	912374672	Warren Eckstein's...
US	43214993	R14QJW3XF8Q01P	B00840HUI0	902215727	Tyson's True Chew...
US	12835065	R2HB7AX0394ZGY	B001GS71K2	568880110	Soft Side Pet Cra...
US	26334022	RGKMPDQGSABR3	B004ABH1LG	692846826	EliteField 3-Door...
US	22283621	R1DJCVPQGCV66E	B00AX0LFM4	590674141	Carlson 68-Inch W...
US	14469895	R3V52EAWLPBFQ	B00DQFZGZ0	688538603	Dog Seat Cover Wi...
US	50896354	R3DK08J1J28QBI	B00DIRF9US	742358789	The Bird Catcher ...
US	18440567	R764DBXGRNECG	B00JRCBFUG	869798483	Cat Bed - Purrfec...
US	50502362	RW1853GAT0Z9F	B000L3XYZ4	501118658	PetSafe Drinkwell...
US	33930128	R33GITXNUF1AD4	B00BOEXWFG	454737777	Contech ZenDog Ca...
US	43534290	R1H7AVM81TAYRV	B001HBBQKY	420905252	Wellness Crunchy ...
US	45555864	R2ZOYAQZNNZZWV	B00701FHB0	302588963	Rx Vitamins Essen...
US	11147406	R2FN1H3CGW6J8H	B001P3NU30	525778264	Virbac C.E.T. Enz...
US	6495678	RJB41Q575XNG4	B00ZP6HS6S	414117299	Kitty Shack - 2 i...
US	2019416	R28W8BM1587CPF	B00IP05CUA	833937853	Wellness Kittles ...
US	40459386	R1II0M01NIG293	B001U8Y598	85343577	OmniPet Anti-Ant ...
US	23126800	RMB8N0DBRH340	B011AY4JWO	499241195	K9KONNECTION [New...
US	30238476	R24WB6A6WVPU6	B00DDSH5EA	409532388	SUNSEED COMPANY 3...
US	35113999	ROCJSH0P9YSRW	B00PJW5OR8	259271919	CXB1983(TM)Cute P...

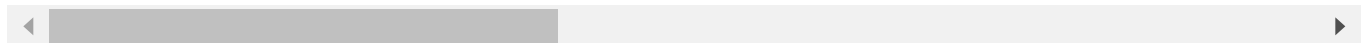
only showing top 20 rows



## ▼ Create DataFrames to match tables

```
from pyspark.sql.functions import to_date
# Read in the Review dataset as a DataFrame
df.show(5)
```

```
+-----+-----+-----+-----+-----+-----+
|marketplace|customer_id|review_id|product_id|product_parent|product_title|
+-----+-----+-----+-----+-----+-----+
|US|28794885|REAKC26P07MDN|B00Q0K9604|510387886|(8-Pack) EZwhelp ...|
|US|11488901|R3NU70MZ4HQIEG|B00MBW509W|912374672|Warren Eckstein's...|
|US|43214993|R14QJW3XF8Q01P|B00840HUIO|902215727|Tyson's True Chew...|
|US|12835065|R2HB7AX0394ZGY|B001GS71K2|568880110|Soft Side Pet Cra...|
|US|26334022|RGKMPDQGSABR3|B004ABH1LG|692846826|EliteField 3-Door...|
+-----+-----+-----+-----+-----+-----+
only showing top 5 rows
```



```
# Create the customers_table DataFrame
customers_df = df.groupby("customer_id").agg({"customer_id": "count"}).withColumnRenamed("count", "customer_count")
```

```
customers_df.show()
```

```
↳ +-----+-----+
|customer_id|customer_count|
+-----+-----+
|10270641|1|
|18365872|1|
|16711087|1|
|10742726|2|
|41169638|1|
|43622307|1|
|24540309|2|
|28258386|1|
|35329257|2|
|14552054|1|
|14529507|5|
|45392827|5|
|47282953|1|
|8201930|1|
|20109760|2|
|16405801|4|
|15056685|21|
|20840575|2|
|39048303|1|
|5596610|1|
+-----+-----+
only showing top 20 rows
```

```
# Create the products_table DataFrame and drop duplicates.
products_df = df.select(["product_id", "product_title"]).drop_duplicates()
```

```
products_df.show()
```

```
+-----+-----+
|product_id|      product_title|
+-----+-----+
|B00134HSYS|Special Edition P...|
|B00BS6NPBG|High Tech Pet 6-P...|
|B000F930FS|PetSafe ScoopFree...|
|B000FJ9QTW|Insight ActiviToy...|
|B000ALY00Q|SmartCat Bootsie'...|
|B00Q8ETIZ0|Dogloveit Rubber ...|
|B002VU2BA4|Kragonfly Interch...|
|B00QA3K3QM|Attmu Retractable...|
|B005DGHUC2|Zoo Med Laborator...|
|B000XY7C7C|All Four Paws, Th...|
|B004UUE260|Animal Planet PET...|
|B008APML2C|Chuckit Medium Ul...|
|B00ZJN7T8E|1 Half Portion Ja...|
|B00NRZC8LY|Hide-A-Toy Hallow...|
|B001P3NU4E|Virbac C.E.T. Enz...|
|B003TEQ2U6|Jolly Pets Jolly ...|
|B00DJSNF0M|Multipet Lenny th...|
|B00FXVFEQG|Bags on Board Dur...|
|B005ORDWYA|Just One Bite II ...|
|B003E770G4|  Petmate Sky Kennel|
+-----+-----+
```

only showing top 20 rows

```
# Create the review_id_table DataFrame.
```

```
# Convert the 'review_date' column to a date datatype with to_date("review_date", 'yyyy-MM-dd')
review_id_df = df.select(["review_id", "customer_id", "product_id", "product_parent", to_date(
```

```
review_id_df.show(5)
```

```
+-----+-----+-----+-----+-----+
|      review_id|customer_id|product_id|product_parent|review_date|
+-----+-----+-----+-----+-----+
| REAKC26P07MDN|  28794885|B00Q0K9604|    510387886| 2015-08-31|
|R3NUJ70MZ4HQIEG| 11488901|B00MBW509W|    912374672| 2015-08-31|
|R14QJW3XF8Q01P| 43214993|B00840HUI0|    902215727| 2015-08-31|
|R2HB7AX0394ZGY| 12835065|B001GS71K2|    568880110| 2015-08-31|
| RGKMPDQGSahr3| 26334022|B004ABH1LG|    692846826| 2015-08-31|
+-----+-----+-----+-----+-----+
```

only showing top 5 rows

```
# Create the vine_table. DataFrame
```

```
vine_df = df.select(["review_id", "star_rating", "helpful_votes", "total_votes", "vine", "verified",
```

```
vine_df.show(5)
```

review_id	star_rating	helpful_votes	total_votes	vine	verified_purchase
REAKC26P07MDN	5	0	0	N	Y
R3NU70MZ4HQIEG	2	0	1	N	Y
R14QJW3XF8Q01P	5	0	0	N	Y
R2HB7AX0394ZGY	5	0	0	N	Y
RGKMPDQGSAGR3	5	0	0	N	Y

only showing top 5 rows

## ▼ Connect to the AWS RDS instance and write each DataFrame to its table.

```
# Configure settings for RDS
mode = "append"
jdbc_url="jdbc:postgresql://challenge-amazon-database.cquzpliuux6z.us-west-2.rds.amazonaws.co
config = {"user":"postgres",
          "password": "challenge16",
          "driver":"org.postgresql.Driver"}

# Write review_id_df to table in RDS
review_id_df.write.jdbc(url=jdbc_url, table='review_id_table', mode=mode, properties=config)
```

```

111 get_return_value(answer, gateway_client, target_id, name)
    326         raise Py4JJavaError(
    327             "An error occurred while calling {0}{1}{2}.\n".
--> 328             format(target_id, ".", name), value)
    329     else:
    330         raise Py4JError(

```

**Py4JJavaError:** An error occurred while calling o74.jdbc.

: org.postgresql.util.PSQLException: FATAL: database "challenge-amazon-database" does not exist

```

    at
org.postgresql.core.v3.QueryExecutorImpl.receiveErrorResponse(QueryExecutorImpl.java:
    at
org.postgresql.core.v3.QueryExecutorImpl.readStartupMessages(QueryExecutorImpl.java:2
    at org.postgresql.core.v3.QueryExecutorImpl.<init>
(QueryExecutorImpl.java:147)
    at
org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryImpl
    at
org.postgresql.core.ConnectionFactory.openConnection(ConnectionFactory.java:51)
    at org.postgresql.jdbc.PgConnection.<init>(PgConnection.java:225)
    at org.postgresql.Driver.makeConnection(Driver.java:465)
    at org.postgresql.Driver.connect(Driver.java:264)
    at
org.apache.spark.sql.execution.datasources.jdbc.connection.BasicConnectionProvider.ge
    at
org.apache.spark.sql.execution.datasources.jdbc.connection.ConnectionProvider$.create
    at
org.apache.spark.sql.execution.datasources.jdbc.JdbcUtils$.anonfun$createConnectionF
    at
org.apache.spark.sql.execution.datasources.jdbc.JdbcRelationProvider.createRelation(J

```

# Write products\_df to table in RDS

# about 3 min

```
products_df.write.jdbc(url=jdbc_url, table='products_table', mode=mode, properties=config)
```

-----  
**Py4JJavaError** Traceback (most recent call last)

<ipython-input-40-f9bb330a0bb2> in <module>

1 # Write products\_df to table in RDS

2 # about 3 min

----> 3 products\_df.write.jdbc(url=jdbc\_url, table='products\_table', mode=mode,  
 properties=config)

3 frames

/content/spark-3.1.3-bin-hadoop2.7/python/lib/py4j-0.10.9-src.zip/py4j/protocol.py  
 in get\_return\_value(answer, gateway\_client, target\_id, name)

326 raise Py4JJavaError(  
 327 "An error occurred while calling {0}{1}{2}.\n".  
 --> 328 format(target\_id, ".", name), value)

329 else:  
 330 raise Py4JError(  
 331 "An error occurred while calling {0}{1}{2}.\n".  
 332 format(target\_id, ".", name), value)

**Py4JJavaError:** An error occurred while calling o171.jdbc.

: org.postgresql.util.PSQLException: The connection attempt failed.

at

org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryImpl

at

org.postgresql.core.ConnectionFactory.openConnection(ConnectionFactory.java:51)

at org.postgresql.jdbc.PgConnection.<init>(PgConnection.java:225)

at org.postgresql.Driver.makeConnection(Driver.java:465)

at org.postgresql.Driver.connect(Driver.java:264)

at

org.apache.spark.sql.execution.datasources.jdbc.connection.BasicConnectionProvider.ge

at

org.apache.spark.sql.execution.datasources.jdbc.connection.ConnectionProvider\$.create

at

org.apache.spark.sql.execution.datasources.jdbc.JdbcUtils\$.anonfun\$createConnectionF

at

org.apache.spark.sql.execution.datasources.jdbc.JdbcRelationProvider.createRelation(J

at

org.apache.spark.sql.execution.datasources.SaveIntoDataSourceCommand.run(SaveIntoData

at

org.apache.spark.sql.execution.command.ExecutedCommandExec.sideEffectResult\$lzycomput

at

org.apache.spark.sql.execution.command.ExecutedCommandExec.sideEffectResult(commands.

at

org.apache.spark.sql.execution.command.ExecutedCommandExec.doExecute(commands.scala:9

at

org.apache.spark.sql.execution.SparkPlan\$.anonfun\$execute\$1(SparkPlan.scala:180)

at

org.apache.spark.sql.execution.SparkPlan\$.anonfun\$executeQuery\$1(SparkPlan.scala:218)

at

```
# Write customers_df to table in RDS
```

```
# 5 min 14 s
```

```
customers_df.write.jdbc(url=jdbc_url, table='customers_table', mode=mode, properties=config)
```

at

```
# Write vine_df to table in RDS
```

```
# 11 minutes
```

```
vine_df.write.jdbc(url=jdbc_url, table='vine_table', mode=mode, properties=config)
```

at

at

[Colab paid products](#) - [Cancel contracts here](#)

❗ 1s completed at 9:06 PM

● ✕