

Piruz Alemi **Big Data Report:**

Subject: Amazon Web Services (AWS) + Apache Zeppeline (ZEPL) Data Science Analytic Platform: ETL, SQL, RDS

Date: April 3rd, 2020

Before I Began

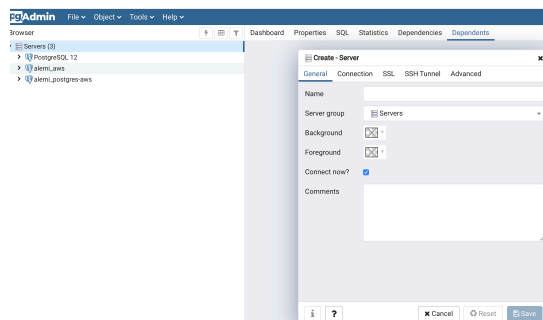
1. Created a new repository for this project called `big-data-challenge`.
2. Cloned the new repository to my computer.
3. Inside my local git repository, created a directory for the level of challenge I chose. I Used folder names corresponding to the challenges: **level-1** or **level-2**.
4. Added my converted ZEPL notebook to this folder. This became the main script to run for my analysis. I also added any SQL queries. I used a `.sql` file and added it to my repo.
5. Pushed the above changes to GitHub or GitLab.

Note:

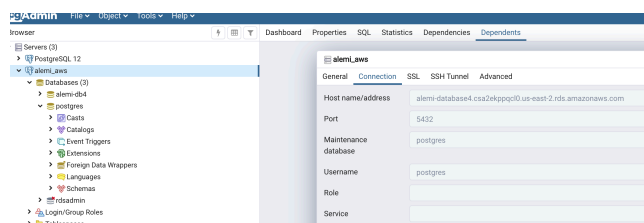
I used ETL processes in Big Data using PySpark and using AWS's Relational Databases.

I linked the AWS RDS to PG_Admin:

1. Created in AWS RDS, an end-point for the server pointing to the `alemi-db4`
2. On my local PG_Admin, created a new server – `alemi-db4`,



3. Copied and pasted the end-point from AWS – RDS to my newly created server on my local Pg-Admin



4. This connected my PG_Admin to AWS_RDS on the amazon shared public platform

5. In Google_Drive Collaborate Python Program, under SPARKS, established the connection and wrote a new table: Alemi_ETL_Info:

▼ Write DataFrame to RDS

```
# Configuration for RDS instance
mode="append"
jdbc_url = "jdbc:postgresql://alemi-database4.csa2ekppqcl0.us-east-2.rds.amazonaws.com:5432/alemi-db4"
config = {"user": "postgres",
          "password": "*****",
          "driver": "org.postgresql.Driver"}

[ ] # Write DataFrame to table
employee_personal_info.write.jdbc(url=jdbc_url, table='Alemi_ETL_info', mode=mode, properties=config)
```

- 6.

Background

In this assignment you will put my ETL skills to the test. Many of Amazon's shoppers depend on product reviews to make a purchase. Amazon makes these datasets publicly available. However, they are quite large and can exceed the capacity of local machines to handle. One dataset alone contains over 1.5 million rows; with over 40 datasets, this can be quite taxing on the average local computer.

1. My first goal for this assignment was to perform the ETL process completely in the cloud and upload a DataFrame to an RDS instance.

zepl

User Guide FAQ Support

Search docs

Notebooks

Sharing Notebooks

External Notebooks

Apache Zeppelin

GitHub

Amazon S3

Import Notebook

Export Notebook

Explore Public Notebooks

Notebook Versioning

Autocompletion

Interpreters

Apache Spark

Python

JDBC

o JDBC Interpreter

o Creating a New JDBC Interpreter

o Secure Database Connections

Config

Angular

Accessing Data

Data Sources

Data Source Integrations

JDBC Interpreter

Zepl supports a JDBC interpreter with drivers for popular databases. Drivers for the following databases are currently supported by Zepl:

- MySQL
- Oracle
- PostgreSQL
- Redshift

For additional database connections, please contact support@zepl.com.

Before connecting to your database, it's important to check the following:

- the database is currently up and running
- the database is accessible from the public internet
- you have the proper credentials to access the database

Creating a New JDBC Interpreter

First you'll need to create an interpreter to provide the database connection information as follows:

1. Click the **Create interpreter** button from the Interpreter settings page
2. Select **jdbc** from the **Interpreter type** group
3. Provide a name for your interpreter (note that this name will be used in the notebook to call this JDBC interpreter)
4. Provide the JDBC connection URL, username and password
5. Select a JDBC driver from the dropdown menu

Use the **Test connection** button to test the connection.

Using Your JDBC Database Interpreter

Once you have created the JDBC interpreter, you can use it in your notebook by providing the `%Interpreter: name` directive. For example, if you have created your JDBC interpreter with name "psql", you can use `%psql` in the notebook as follows:

```

# Configuration for RDS instance
mode="append"
jdbc_url = "jdbc:postgresql://alemi-database4.csa2ekppqcl9.us-east-2.rds.amazonaws.com:5432/alemi-db4"
config = {"user": "postgres",
          "password": "*****",
          "driver": "org.postgresql.Driver"}

# Write DataFrame to table
df.write.jdbc(url=jdbc_url, table='Aleml_ETL_Info', mode=mode, properties=config)

# Create a new DataFrame for Market Star Rating

# Write DataFrame to table
df3.write.jdbc(url=jdbc_url, table='market_star_rating', mode=mode, properties=config)

```

2. My second goal was to use PySpark or SQL to perform a statistical analysis of selected data.

There were two levels to this homework assignment. I completed both levels.

1. Create DataFrames to match production-ready tables from two big Amazon customer review datasets.
2. Analyzed whether reviews from Amazon's Vine program are trustworthy.

Steps

Level 1

- Used the furnished schemata to create tables in my RDS database.
- Created two separate ZEPL notebooks and **extracted** any two datasets from the list at [review dataset](#), one into each notebook.

```

SAMPLE CONTENT:
https://s3.amazonaws.com/amazon-reviews-pds/tsv/sample_us.tsv
https://s3.amazonaws.com/amazon-reviews-pds/tsv/sample_fr.tsv

DATA COLUMNS:
marketplace - 2 letter country code of the marketplace where the review was written.
customer_id - Random identifier that can be used to aggregate reviews written by a single author.
review_id - The unique ID of the review.
product_id - The unique Product ID the review pertains to. In the multilingual dataset the reviews for the same product in different countries can be grouped by the same product_id.
product_parent - Random identifier that can be used to aggregate reviews for the same product.
product_title - Title of the product.
product_category - Broad product category that can be used to group reviews (also used to group the dataset into coherent parts).
star_rating - The 1-5 star rating of the review.
helpful_votes - Number of helpful votes.
total_votes - Number of total votes the review received.
vine - Review was written as part of the Vine program.
verified_purchase - The review is on a verified purchase.
review_headline - The title of the review.
review_body - The review text.
review_date - The date the review was written.

DATA FORMAT
Tab ('\t') separated text file, without quote or escape characters.
First line in each file is header; 1 line corresponds to 1 record.

US REVIEWS DATASET:
https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Wireless_v1_00.tsv.gz
https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Watches_v1_00.tsv.gz
https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Video_Games_v1_00.tsv.gz
https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Video_DVD_v1_00.tsv.gz
https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Video_v1_00.tsv.gz

```

Note: It is possible to ETL both data sources in a single notebook, but due to the large data sizes, it was easier to work with these S3 data sources in two separate ZEPL notebooks.

Made sure to handle the header correctly. I noted that the column headers are included in the table rows, which made loading into PG_Admin straightforward. I just needed to create the 1. Server, establish the connection based on the end-point created in AWS

– RDS and 2. Create the DB. 3. Loading the table, with TSV file was a straightforward write from data frame command as in:

```
df.write.jdbc(url=jdbc_url, table='Alemi_ETL_info', mode=mode, properties=config)
```

- For each notebook (one dataset per notebook), complete the following:
 - **Counted** the number of records (rows) in the dataset.
 - **Transformed** the dataset to fit the tables in the [schema file](#).
 - Made sure the DataFrames match in data type and in column name.
 - **Loaded** the DataFrames that correspond to tables into an RDS instance.
Note: This process took up to 10 minutes for each! I made sure that everything was correct before uploading.

Level 2 (Optional)

In Amazon's Vine program, reviewers received free products in exchange for reviews.

Amazon has several policies to reduce the bias of its Vine reviews: <https://www.amazon.com/gp/vine/help?ie=UTF8>.

But are Vine reviews truly trustworthy? My task was to investigate whether Vine reviews are free of bias.

I Used both PySpark or—for an extra challenge—SQL to analyze the data.

- Before I used SQL, first I used Spark on ZEPL to extract and transform the data and load it into a SQL table on my RDS account. Performed my analysis with SQL queries on RDS.
- While there were no hard requirements for the analysis, I considered steps to reduce noisy data, e.g., filtering for reviews that meet a certain number of helpful votes, total votes, or both.
- Submitted a summary of my findings and analysis.

Resources I used were:

[customer review datasets](#)

Hints and Considerations

- Consulted the troubleshooting guide for handling issues with ZEPL.
 - Made sure that every cell begins with `%pyspark` in ZEPL. This specifies the interpreter, which I had to have one for each cell.
 - If you import a Jupyter notebook in ZEPL, made sure to delete `%python`, which is automatically added to each cell.
-

Submission

- Copied my ZEPL notebooks into Jupyter Notebooks and upload those to GitHub.
- Copied my SQL queries into `.sql` files and uploaded to GitHub.
- **Important:** I Did not upload notebooks that contain my RDS password and endpoint. I made sure to delete them before making your notebook public!

alemi ETL-RDS.ipynb

```
[42] # Configuration for RDS instance
mode="append"
jdbc_url = "jdbc:postgresql://alemi-database4.csa2ekpgqo10.us-east-2.rds.amazonaws.com:5432/alemi-db4"
config = {"user": "postgres",
          "password": "*****",
          "driver": "org.postgresql.Driver"}

# Write DataFrame to table
df.write.jdbc(url=jdbc_url, table='Aleml_ETL_info', mode=mode, properties=config)

# Create a new DataFrame for Market Star Rating

# Write DataFrame to RDS

[47] # Write DataFrame to table

df3.write.jdbc(url=jdbc_url, table='market_star_rating', mode=mode, properties=config)

[ ]
```

The loaded data in PG_Admin is noted below:

pgAdmin

Query Editor

```
select * from aleml_etl_info
```

marketplace	customer_id	review_id	product_id	product_parent	product_title	product_category	star_rating	help
1	US	44311596	R1UE3BP8GOC	B00ZLHA74D	81812633	Super Jumbo Pla...	Toys	2
2	US	18403006	R1HCUJSG5A	B00L71H8F4	69230292	Barkology Prince...	Toys	2
3	US	41137194	R09G5J99W	B0040781Y1	36230375	Fun Express Inte...	Toys	5
4	US	1662075	R1Y32G31B	B008P9X9X8	21013375	Zuko 2-4004 A.C.	Toys	5
5	US	45601416	R30SLU70C	B009FEMC8	89531627	Intex River Run I...	Toys	5
6	US	24768699	R36EDU38E	B0007JF0PC	95306346	56 Pieces of Woo...	Toys	5
7	US	12918717	RZX17JYV8R	B00KQJNNZ9	64636846	New Age Scare H...	Toys	5
8	US	27225899	R48377CCD	B001RA3H8J	22343727	Franklin Sports M...	Toys	3
9	US	47781802	R0AGD9WDL	B00GNDV8D	43865679	Prismm - Opere...	Toys	5
10	US	47566726	R3NFZ2CJ8R	B008W18P8Q	38710728	Pappa Pig 7 Woo...	Toys	3
11	US	41168357	R2773FQZ9	B00CAEDCD	72805974	Steel Pets Car Se...	Toys	5
12	US	43173284	R1Y045302	B002G54DAA	57447684	BCW - Deluxe Car...	Toys	5
13	US	23310293	R248788C6A	B00ARPLGVY	261944918	Bartle Doll and F...	Toys	5
14	US	15688946	R248788C6A	B001CTC0AD	276247652	Star Wars Clow...	Toys	5
15	US	12364189	R3BPE1J2J2	B008B7P8CA	875828700	Melissa & Doug ...	Toys	5
16	US	32866903	R300K9N9V1	B000TTLAZA	148264874	Baby Einstein Oct...	Toys	5
17	US	1297934	R1C878318D	B00130V99D	289360126	Climb Climber Gd...	Toys	1

Thank you, all my supporters, for an outstanding teamwork!

Respectfully,

Piruz Alemi
April 5th, 2020