!! **delete** AWS RDS **password** and **endpoint** before making your notebook public

<Step0>: accounts

Zepl: [friends.limei@gmail.com](mailto:friends.limei@gmail.com) AWS: [limei\_hou@hotmail.com](mailto:limei_hou@hotmail.com) tongyi1@

AWS RDS “postgresqlforetl…” : root lmpostgre1! pgAdmin: lmpgAdmin1!

<Step1>: Data: picking two data sets from [review dataset](https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt) + Zepl Notebook 1) Extract the two datasets 2) Checking data values and data types

One dataset in one Zepl Notebook

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Software\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Video\_DVD\_v1\_00.tsv.gz

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.

%pyspark

from pyspark import SparkFiles

url = "https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_**Software**\_v1\_00.tsv.gz"

spark.sparkContext.addFile(url)

Software\_df = spark.read.option('header', 'true').csv(SparkFiles.get("amazon\_reviews\_us\_Software\_v1\_00.tsv.gz"), sep="\t", header=True, inferSchema=True)

Software\_df.show(1)

**|marketplace|customer\_id| review\_id |product\_id|product\_parent|product\_title |product\_category|**

| US | 42605767|R3EFW2STIYIY0I|B00MUTIDKI| 248732228|McAfee 2015 Inter...| Software|

**|star\_rating|helpful\_votes|total\_votes|vine|verified\_purchase| review\_headline | review\_body | review\_date|**

| 1| 2| 2| N| Y|I was very disapp...|I was very disapp...|2015-08-31 00:00:00|

Software\_df.count()

341931

Software\_df.printSchema()

root

|-- marketplace: string (nullable = true)

|-- customer\_id: integer (nullable = true)

|-- review\_id: string (nullable = true)

|-- product\_id: string (nullable = true)

|-- product\_parent: integer (nullable = true)

|-- product\_title: string (nullable = true)

|-- product\_category: string (nullable = true)

|-- star\_rating: integer (nullable = true)

|-- helpful\_votes: integer (nullable = true)

|-- total\_votes: integer (nullable = true)

|-- vine: string (nullable = true)

|-- verified\_purchase: string (nullable = true)

|-- review\_headline: string (nullable = true)

|-- review\_body: string (nullable = true)

|-- review\_date: timestamp (nullable = true)

Data types in pyspark?? Bool??

simple derivatives of the AtomicType class:

* Binary – Binary data.
* Boolean – Boolean values.
* Byte – A byte value.
* Date – A datetime value.
* Double – A floating-point double value.
* Integer – An integer value.
* Long – A long integer value.
* Null – A null value.
* Short – A short integer value.
* String – A text string.
* Timestamp – A timestamp value (typically in seconds from 1/1/1970).
* Unknown – A value of unidentified type.

Software\_df.cache()

DataFrame[marketplace: string, customer\_id: int, review\_id: string, product\_id: string, product\_parent: int, product\_title: string, product\_category: string, star\_rating: int, helpful\_votes: int, total\_votes: int, vine: string, verified\_purchase: string, review\_headline: string, review\_body: string, review\_date: timestamp]

Software\_df.select(["vine","verified\_purchase"]).describe().show()

+-------+------+-----------------+

|summary| vine |verified\_purchase|

+-------+------+-----------------+

| count |341931| 341931 |

| mean | null | null |

| stddev| null | null |

| min | N | N |

| max | Y | Y |

+-------+------+-----------------+

Software\_df.select("verified\_purchase").filter("verified\_purchase=='Y'").count()

<Step2>: from review datasets design DB table schema: <Products> <Customers> n:m => <Reviews> 1 review ⬄ n votes

Data types in PostgreSQL??

all these tables are dataset oriented?? All datasets has the same group of customers??

CREATE TABLE products (

product\_id TEXT PRIMARY KEY NOT NULL UNIQUE,

product\_title TEXT

product\_category TEXT [**no need** if this table is just for US ‘Software’ Review Dataset]

);

CREATE TABLE customers (

customer\_id INT PRIMARY KEY NOT NULL UNIQUE,

customer\_count INT [new **aggregation field** !!]

);

CREATE TABLE reviews (

review\_id TEXT PRIMARY KEY NOT NULL,

customer\_id INTEGER,

product\_id TEXT,

product\_parent INTEGER, [same as product\_id???]

[ product\_id is used across different languages, product\_parent is specific to a language.]

review\_date DATE [formate **yyyy-mm-dd** !!]

marketplace TEXT, [**no need** if this table is just for US Review Datasets]

star\_rating INTEGER, [put in vine table]

helpful\_votes INTEGER, [put in vine table]

total\_votes INTEGER, [put in vine table]

vine TEXT, [put in vine table] [Y,N values are TEXT, can be converted to BOOLEAN]

verified\_purchase TEXT, [put in vine table] [Y,N values are TEXT, can be converted to BOOLEAN]

review\_headline TEXT, [not number so not useful for analyzing]

review\_body TEXT [not number so not useful for analyzing]

);

CREATE TABLE vine (

review\_id TEXT PRIMARY KEY,

star\_rating INTEGER,

helpful\_votes INTEGER,

total\_votes INTEGER,

vine TEXT

);

<Step3>: create AWS RDS PostgreSQL DB instance

**<Step4>: locally in pgAdmin 1)create server connect to AWS RDS DB instance 2) create tables in AWS RDS DB instance**

CREATE TABLE products (

product\_id TEXT PRIMARY KEY NOT NULL UNIQUE,

product\_title TEXT

);

CREATE TABLE customers (

customer\_id INT PRIMARY KEY NOT NULL UNIQUE,

customer\_count INT

);

CREATE TABLE reviews (

review\_id TEXT PRIMARY KEY NOT NULL,

customer\_id INTEGER,

product\_id TEXT,

product\_parent INTEGER,

review\_date DATE

);

CREATE TABLE vine (

review\_id TEXT PRIMARY KEY,

star\_rating INTEGER,

helpful\_votes INTEGER,

total\_votes INTEGER,

vine TEXT

);

**<Step5>: Zepl notebook**

1) Extract the two datasets : done in <Step1>

2) Transform

products\_df = Software\_df.select(["product\_id", "product\_title"]).drop\_duplicates()

customers\_df = Software\_df.groupby("customer\_id").agg({"customer\_id": "count"}).withColumnRenamed("count(customer\_id)", "customer\_count")

from pyspark.sql.functions import to\_date

review\_df = Software\_df.select(["review\_id", "customer\_id", "product\_id", "product\_parent", to\_date("review\_date").alias("review\_date")])

vine\_df = Software\_df.select(["review\_id", "star\_rating", "helpful\_votes", "total\_votes", "vine"])

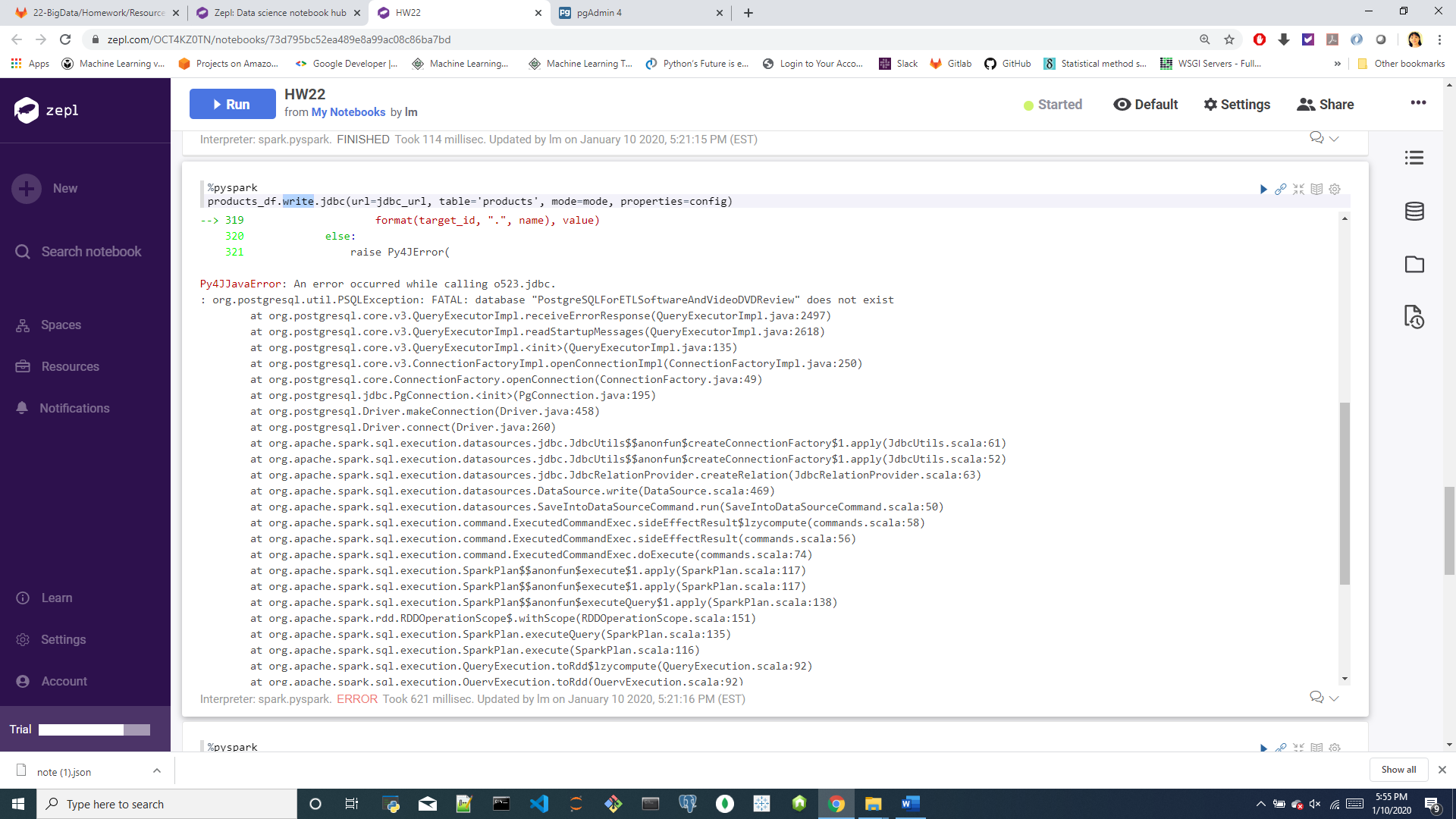
3) Load it into a SQL table on your RDS account (tables in AWS RDS DB instance)

%pyspark

mode = "append"

jdbc\_url="jdbc:postgresql://postgresqlforetlsoftwareandvideodvdreview.cgjfasmmonou.us-east-2.rds.amazonaws.com:5432/PostgreSQLForETLSoftwareAndVideoDVDReview"

config = {"user":"root", "password": "lmpostgre1!", "driver":"org.postgresql.Driver"}



**<Step6>: Analyze Amazon's Vine program(the vine column in the two datasets)**

Are Vine reviews truly trustworthy? Investigate whether Vine reviews are free of bias.

Use either PySpark or—for an extra challenge—SQL to analyze the data.

reduce noisy data, e.g., filtering for reviews that meet a certain number of helpful votes, total votes, or both.

### =========

### Level 1

* Use the furnished schemata to create tables in your RDS database.
* Create two separate ZEPL notebooks and **extract** any two datasets from the list at [review dataset](https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt), one into each notebook.

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

* Be sure to handle the header correctly. If you read the file without the header parameter, you may find that the column headers are included in the table rows.
* For each notebook (one dataset per notebook), complete the following:
  + Count the number of records (rows) in the dataset.
  + **Transform** the dataset to fit the tables in the [schema file](/CW-Coding-Bootcamp/CWRUCLE201907DATA4/blob/master/22-BigData/Homework/Resources/schema.sql). Be sure the DataFrames match in data type and in column name.
  + **Load** the DataFrames that correspond to tables into an RDS instance. **Note:** This process can take up to 10 minutes for each. Be sure that everything is correct before uploading.

### Level 2 (Optional)

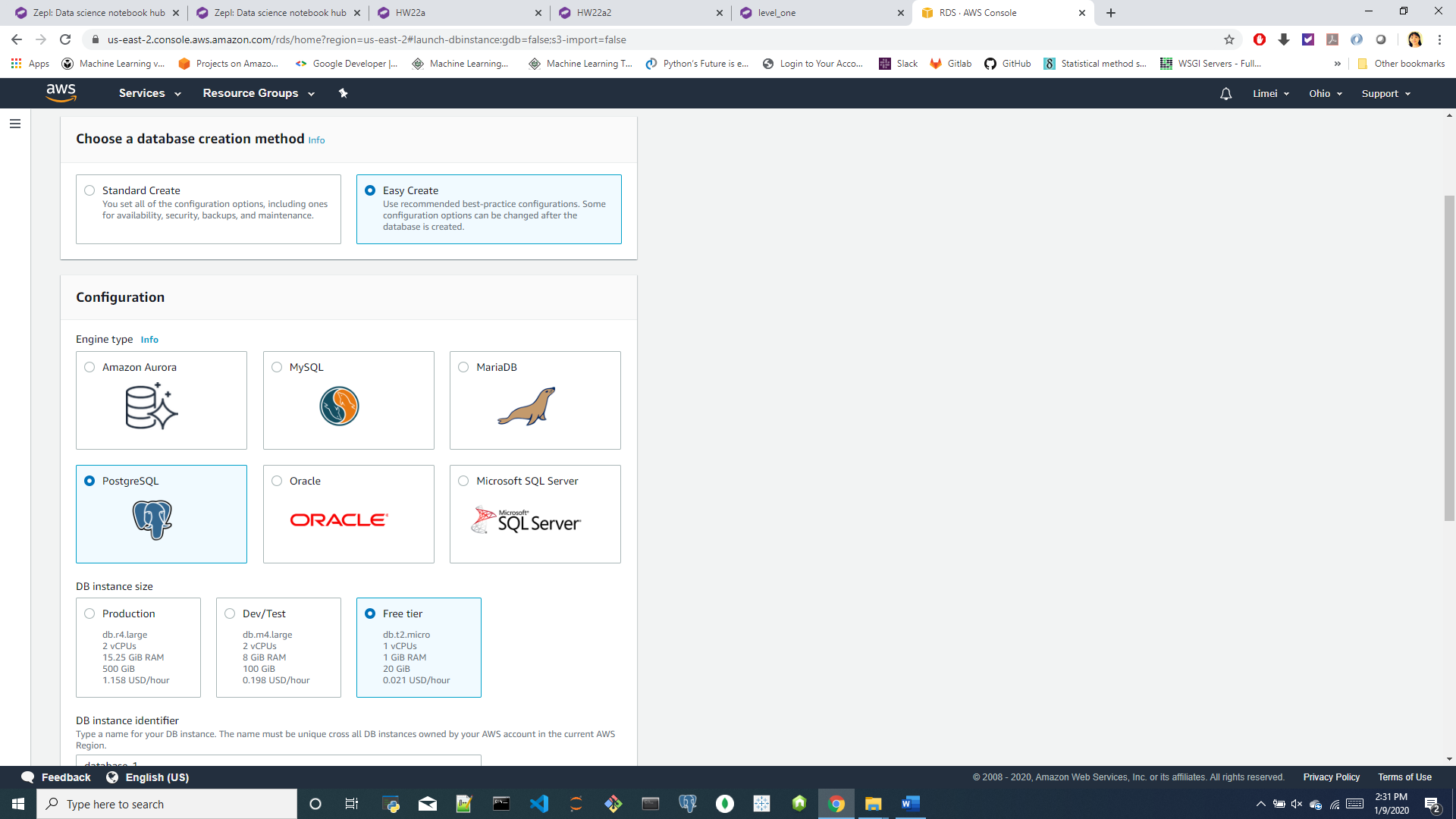
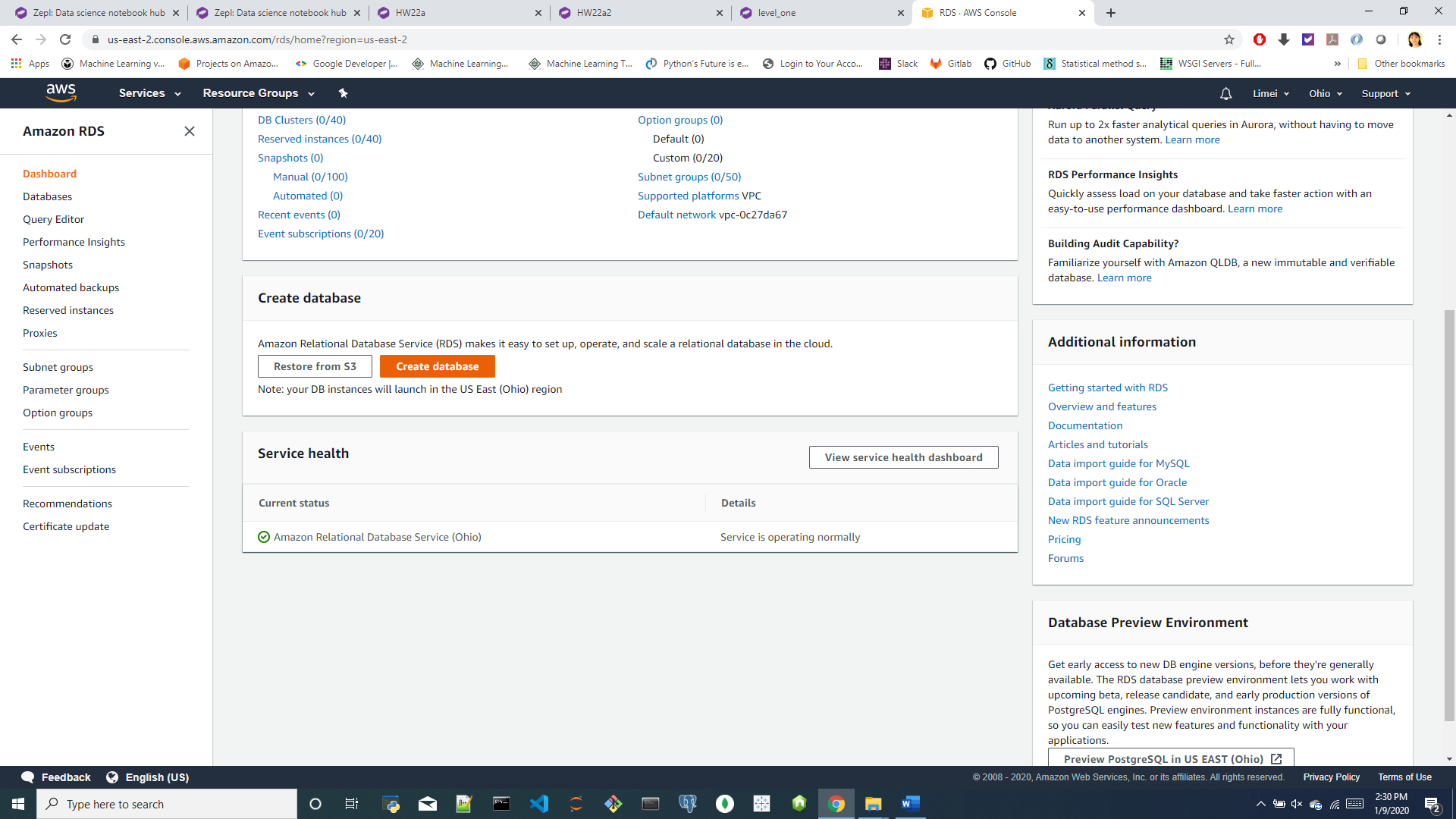
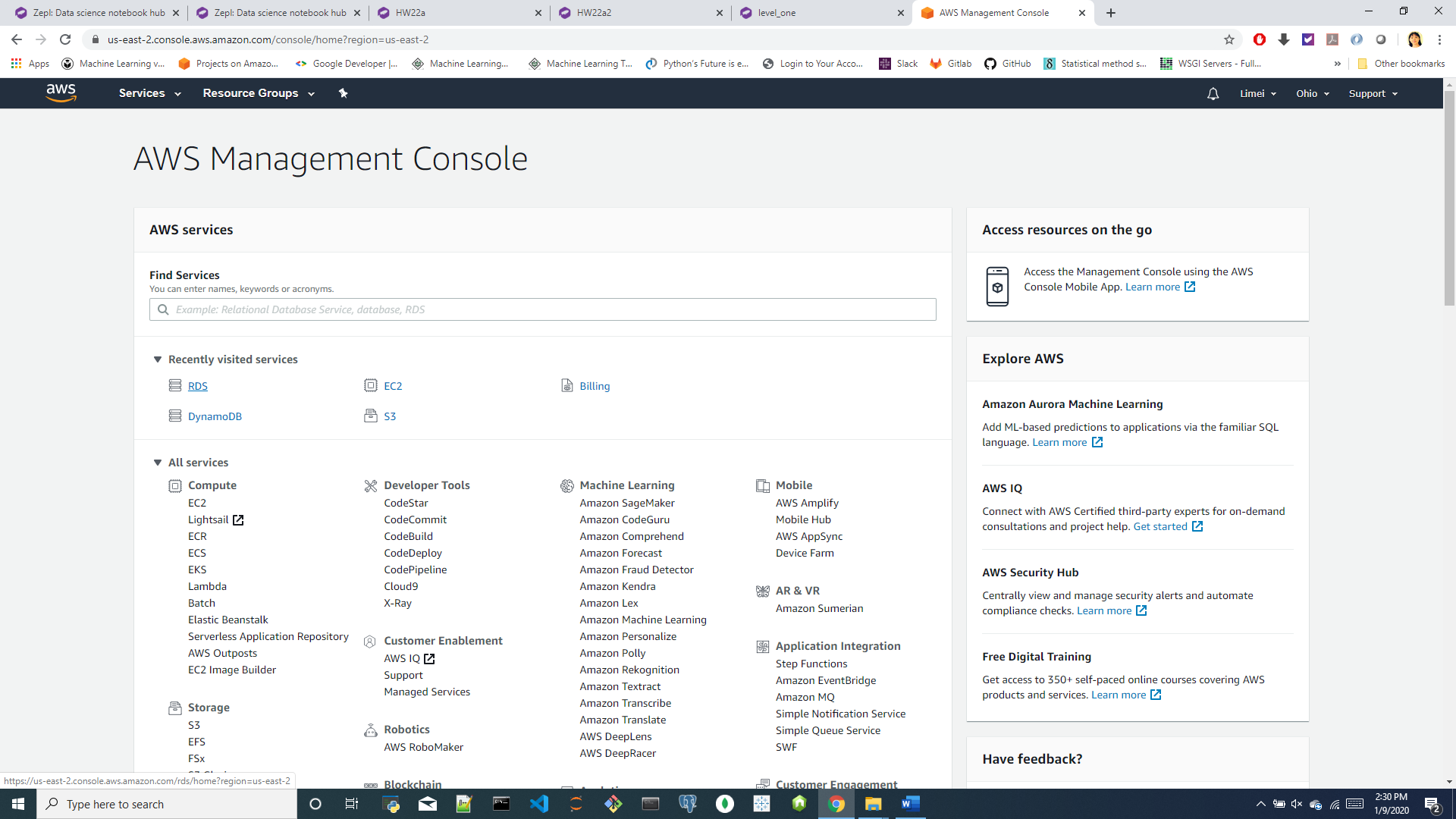
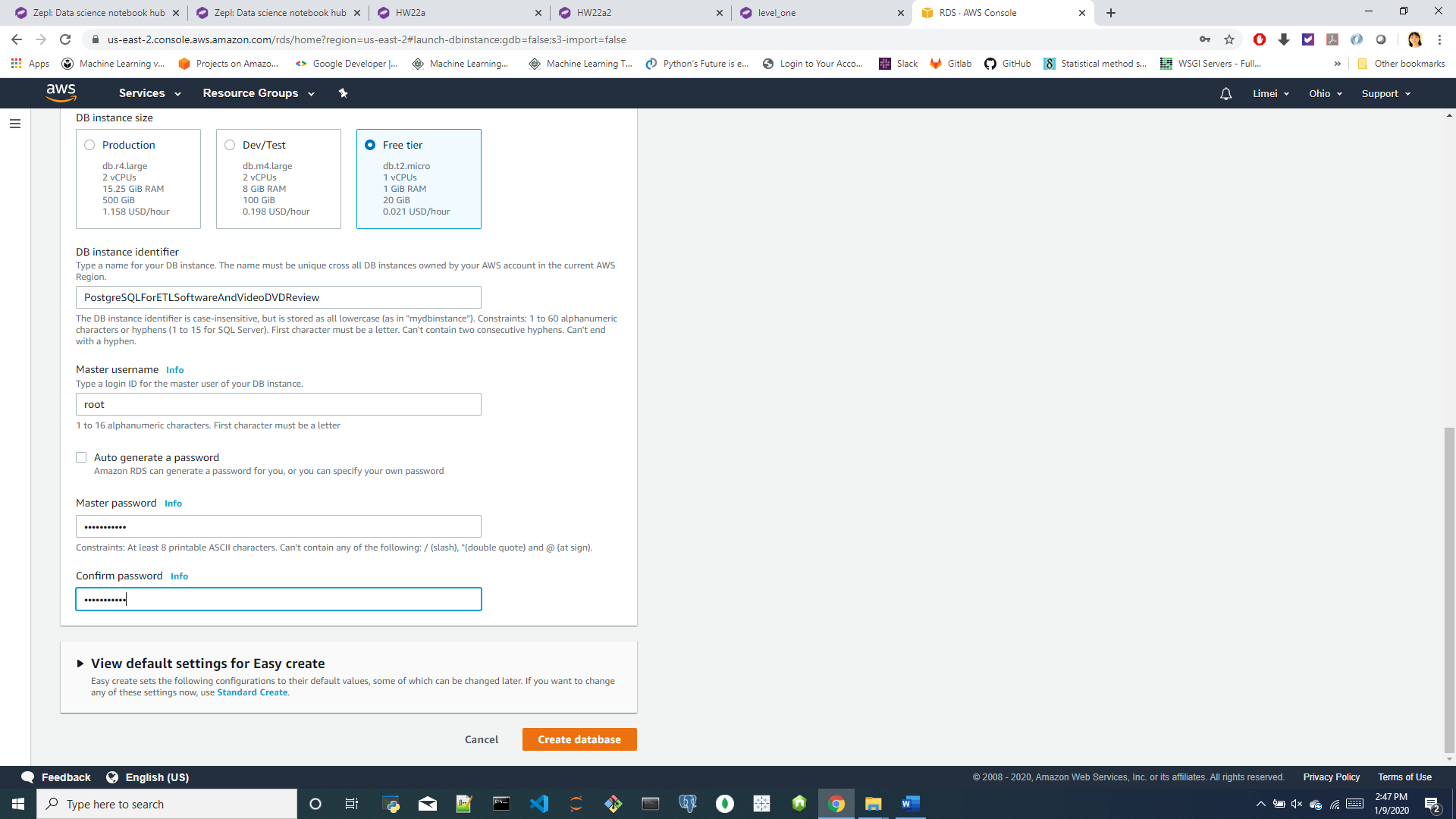
In Amazon's Vine program, reviewers receive free products in exchange for reviews.

===================

**Level 1**

<1> Use the furnished [schema file](file:///C:\CW-Coding-Bootcamp\CWRUCLE201907DATA4\blob\master\22-BigData\Homework\Resources\schema.sql) to create tables in your AWS RDS database.

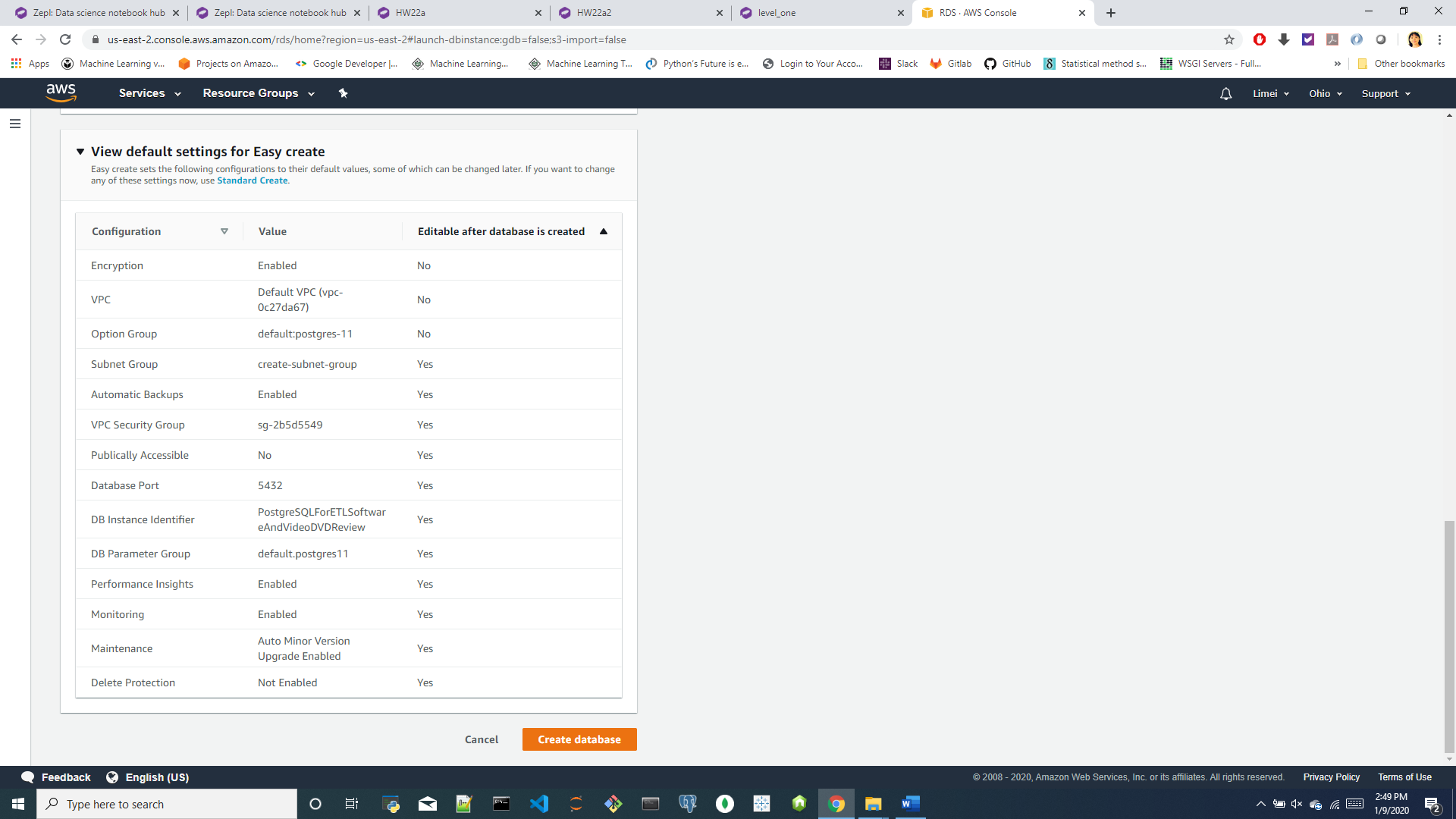
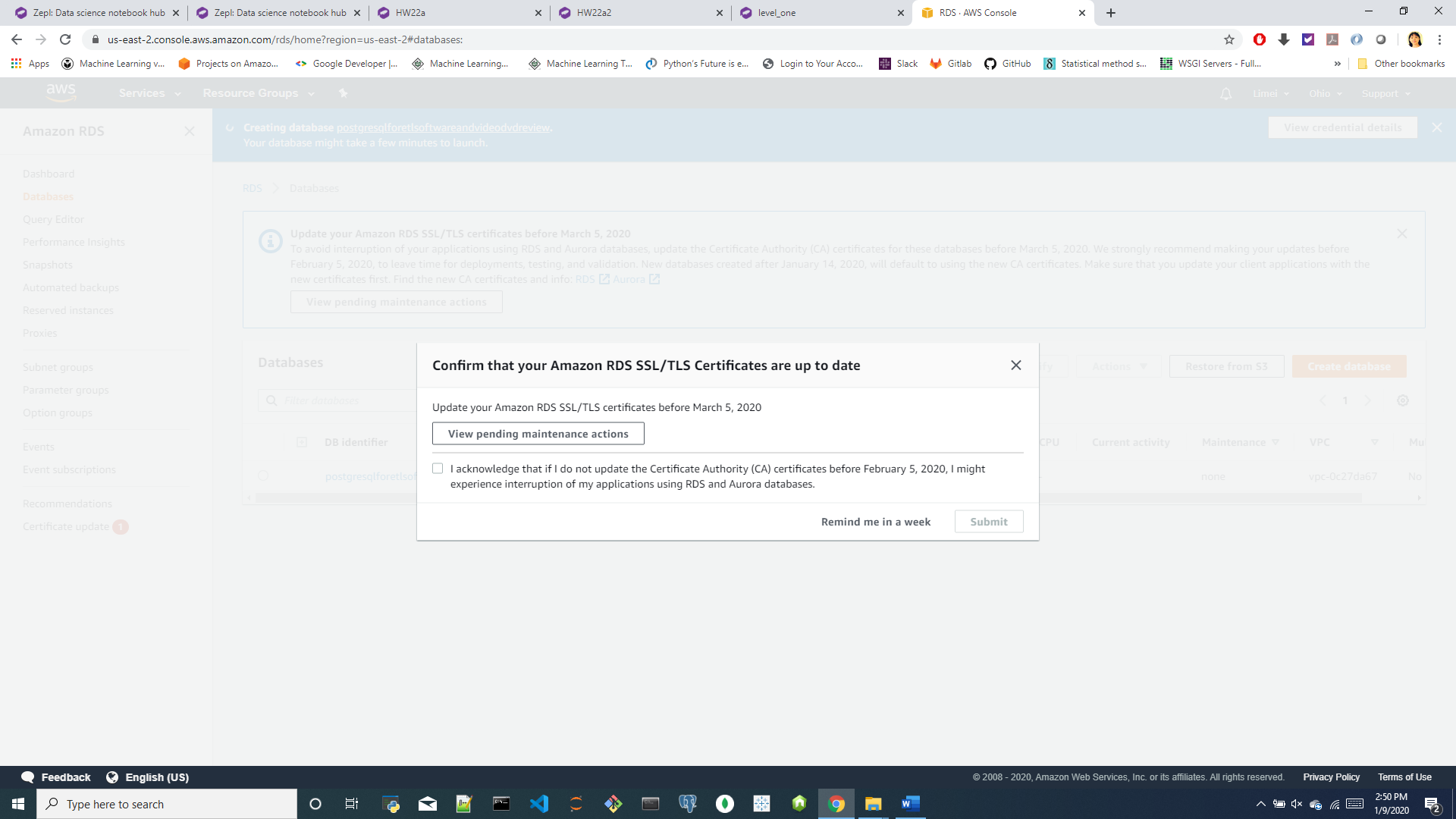
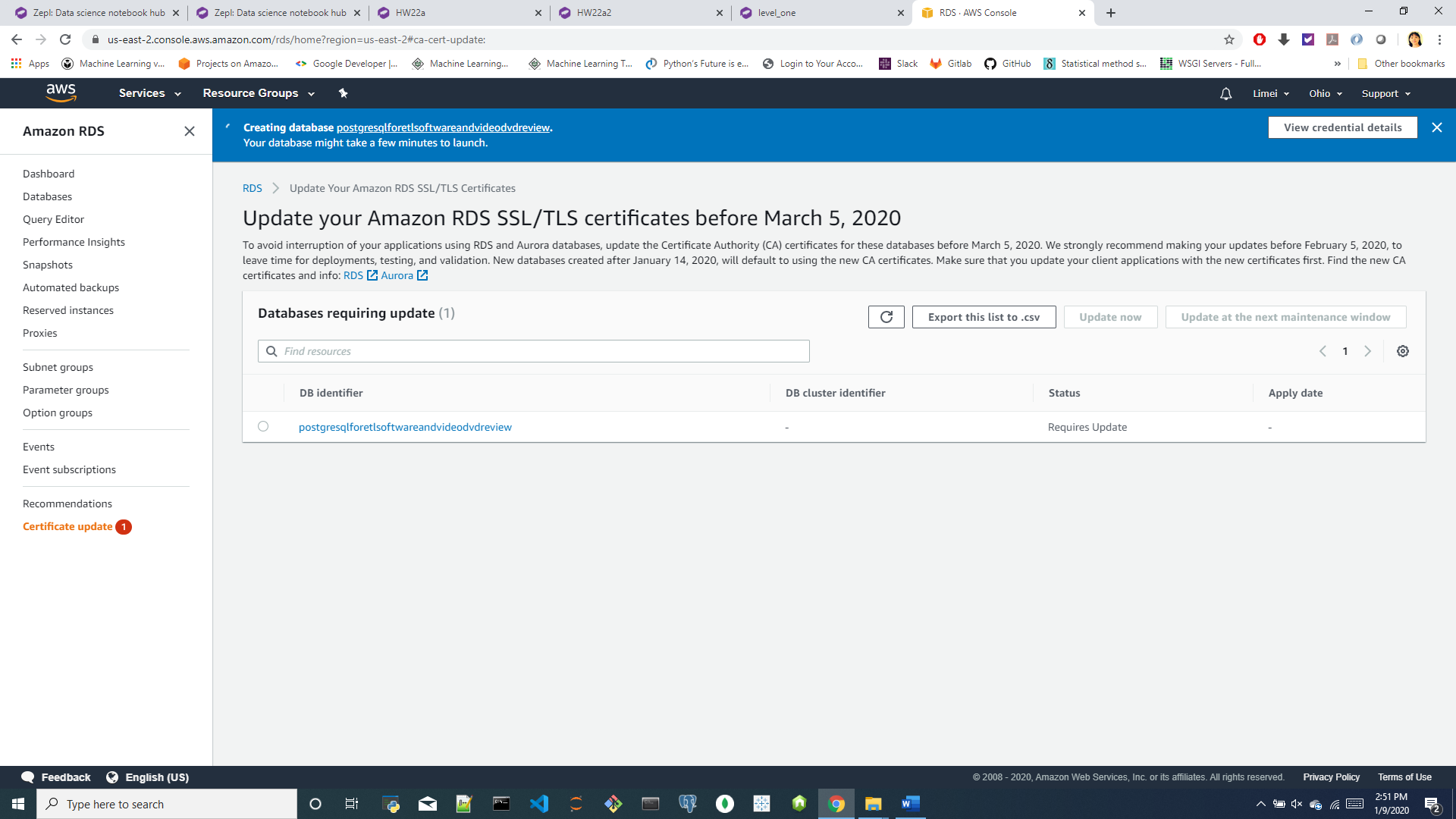
1)create AWS RDS PostgreSQL DB “PostgreSQLForETLSoftwareAndVideoDVDReview”

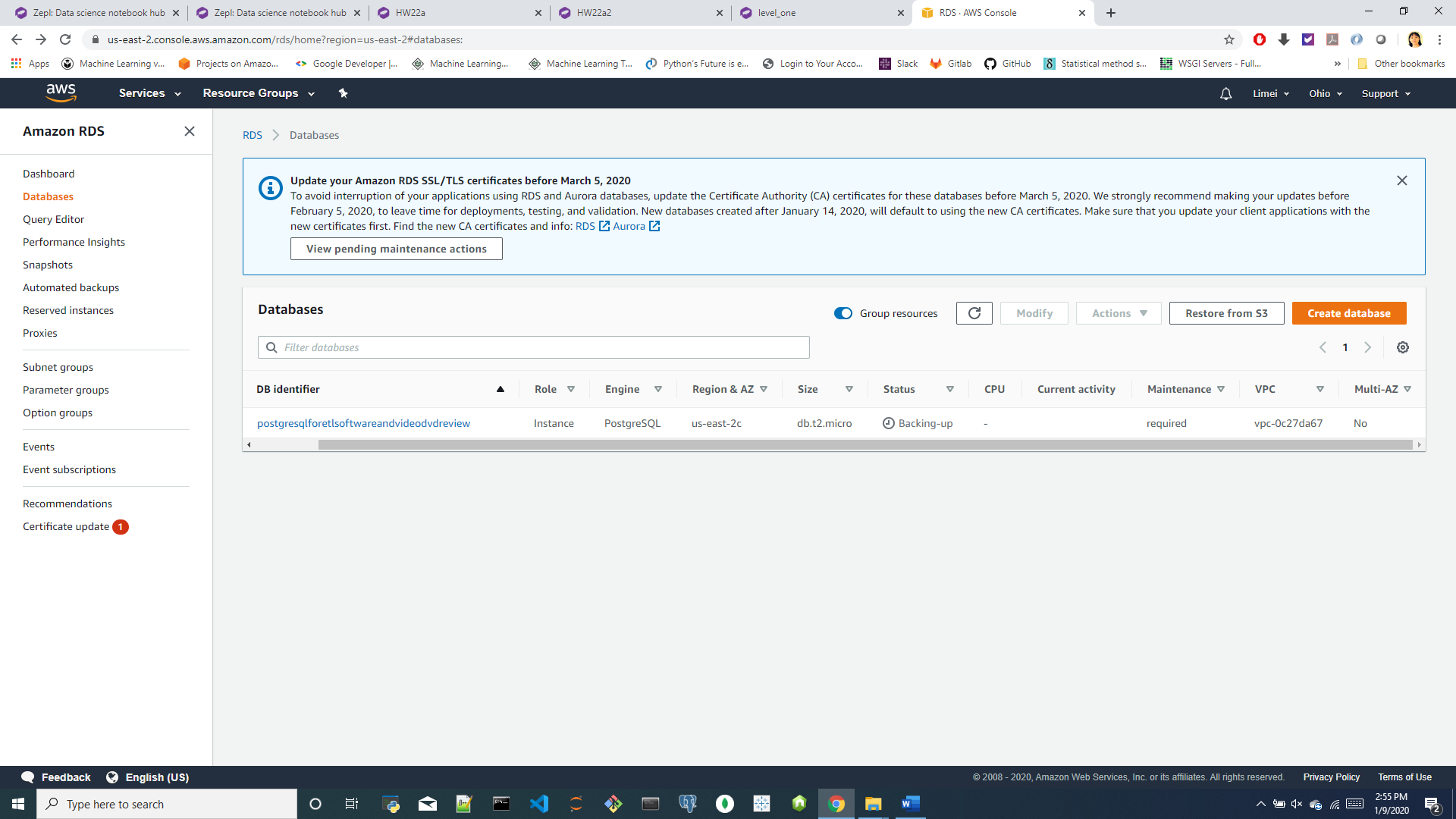
 

DB instance identifier: [PostgreSQLForETLSoftwareAndVideoDVDReview]

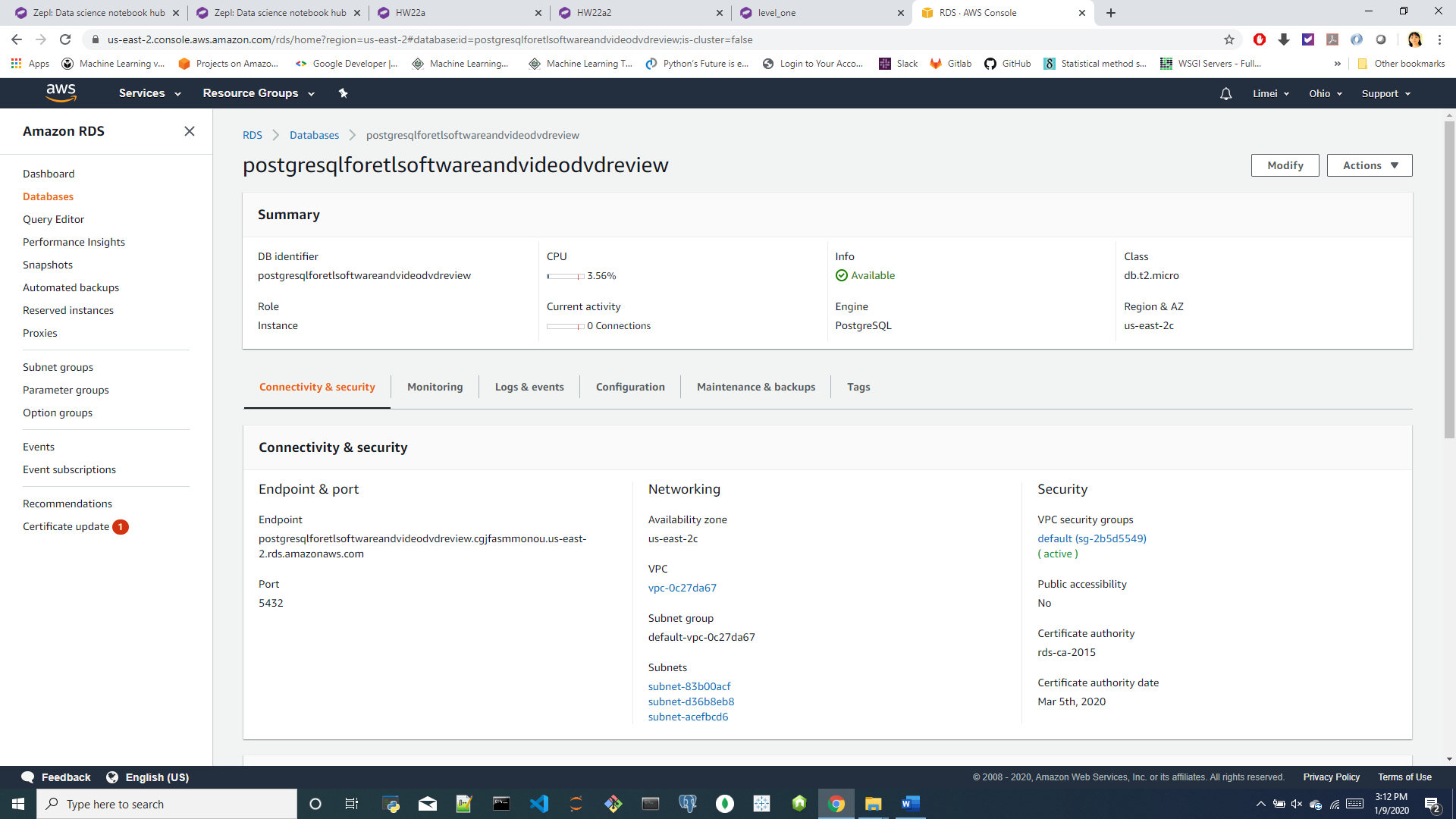
Master username: [root]

Master password: [lmpostgre1!]

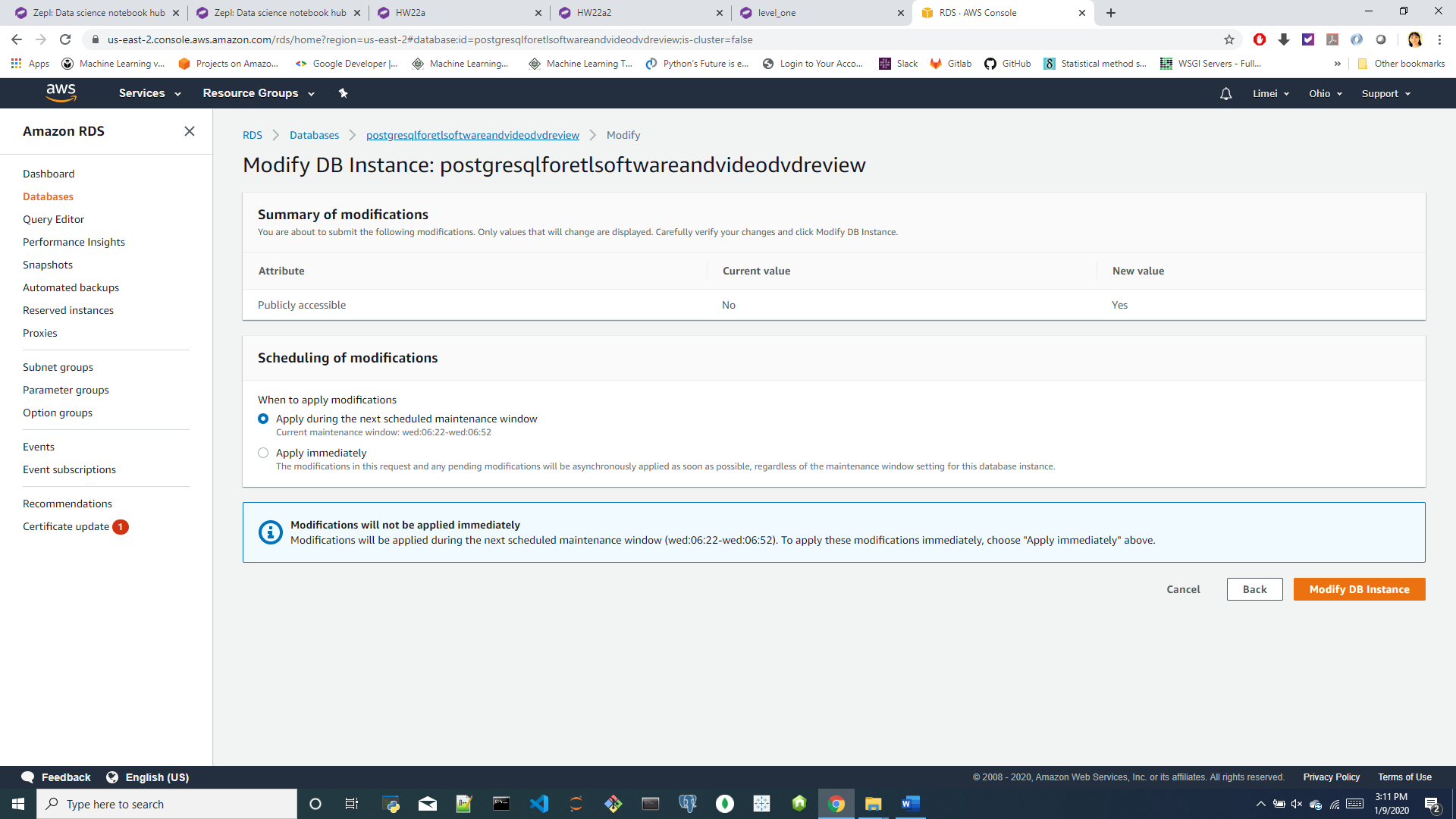
  

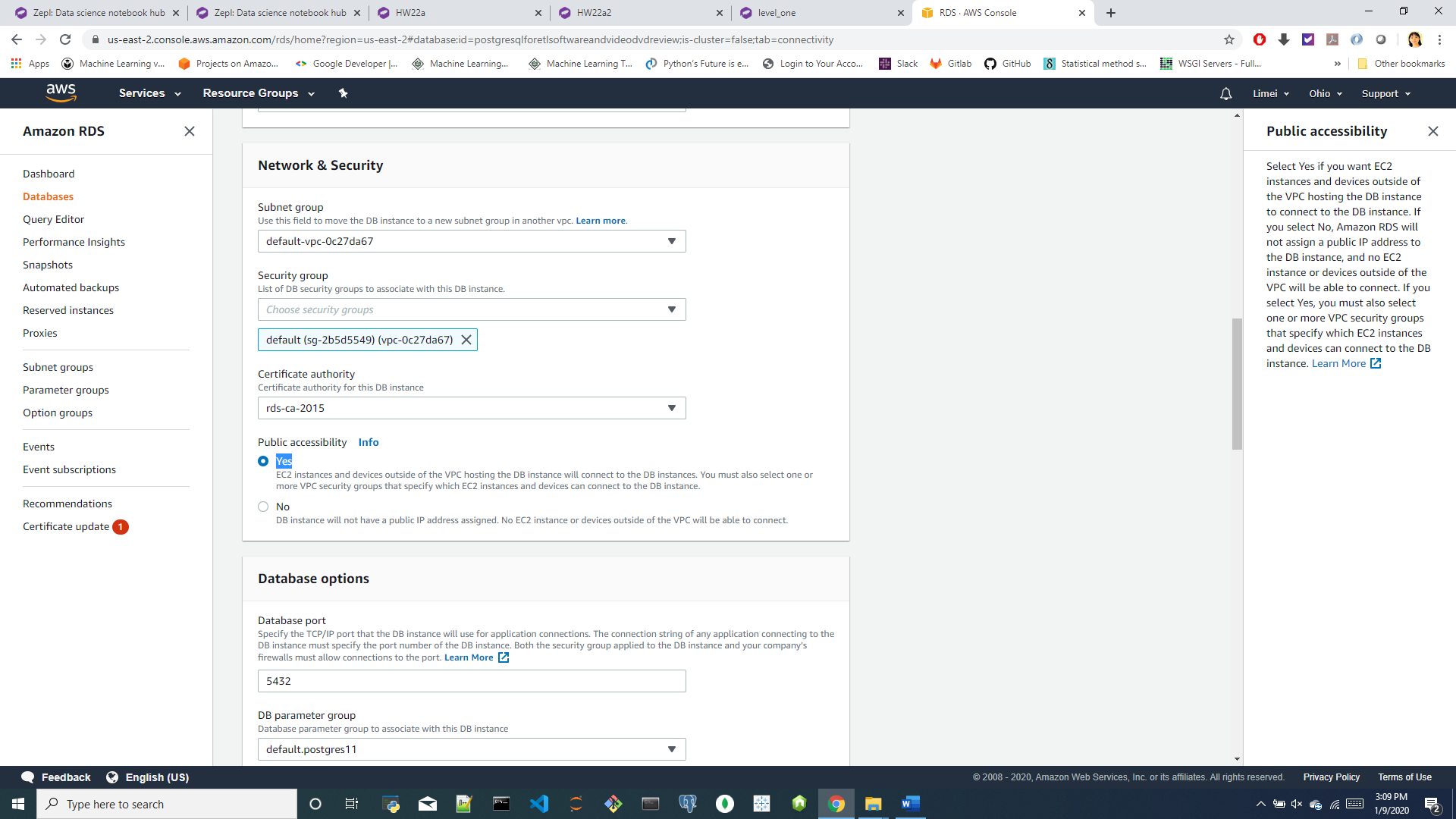


VPC: vpc-0c27da67



Endpoint:

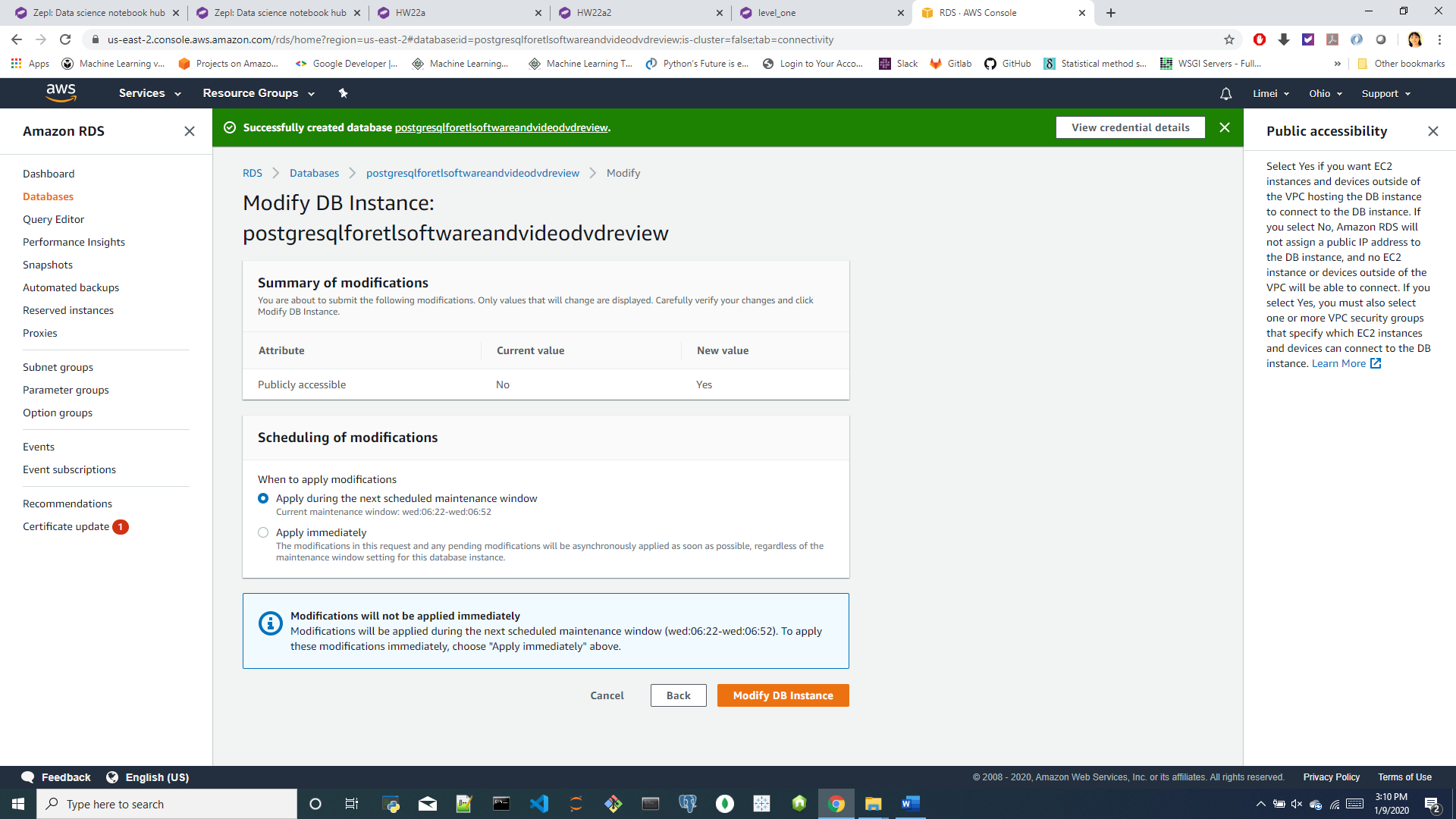




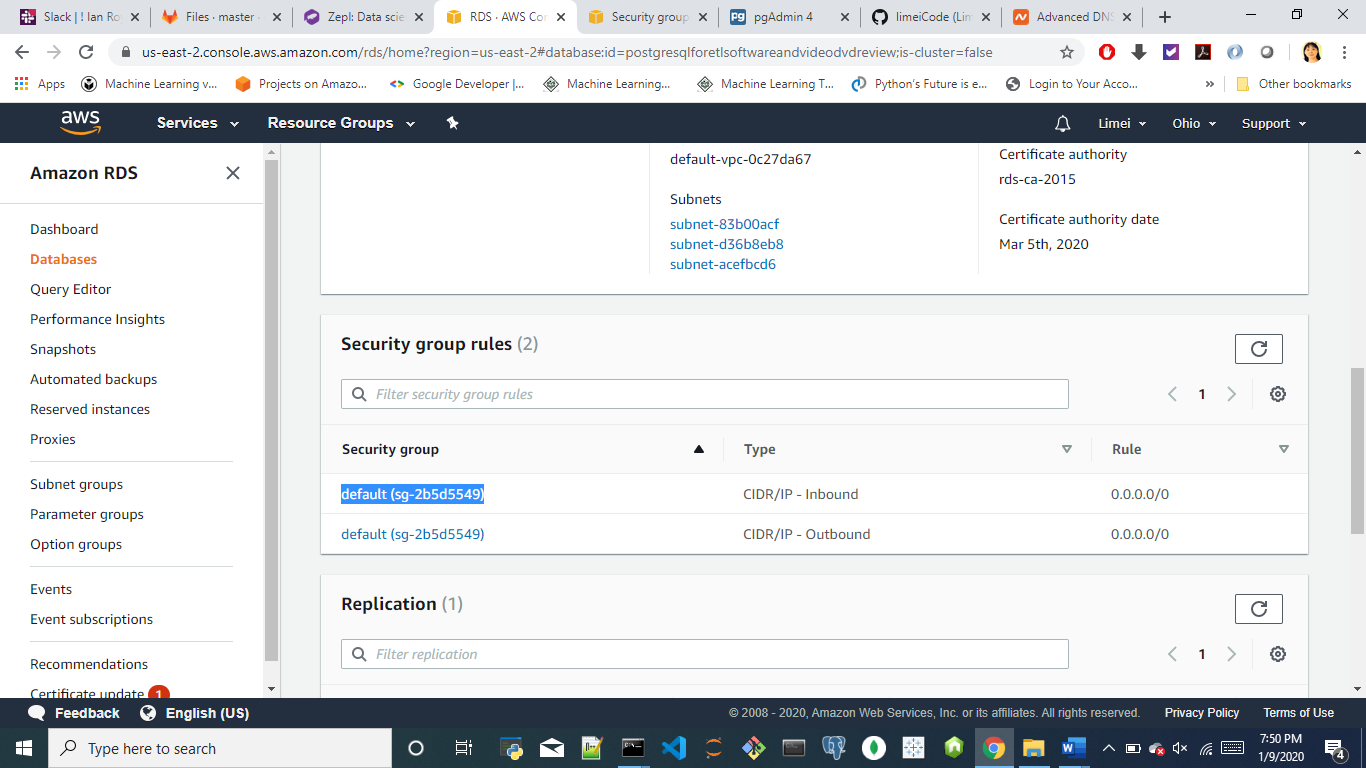
<a>Change “Public accessibility” from No to Yes: this is for development need, not for production!!

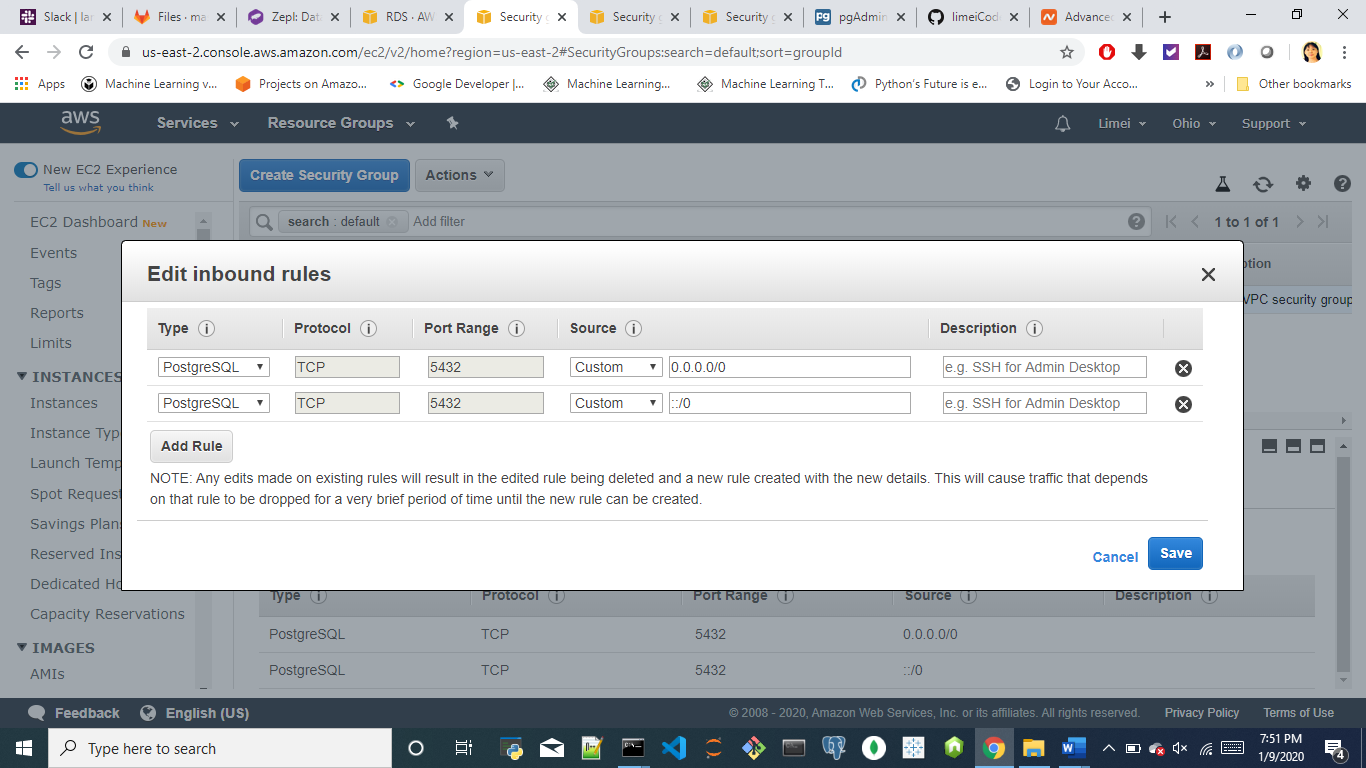
Public accessibility: EC2 instances and devices outside of the VPC hosting the DB instance will connect to the DB instances. You must also select one or more VPC security groups that specify which EC2 instances and devices can connect to the DB instance.

Select Yes if you want EC2 instances and devices outside of the VPC hosting the DB instance to connect to the DB instance. If you select No, Amazon RDS will not assign a public IP address to the DB instance, and no EC2 instance or devices outside of the VPC will be able to connect. If you select Yes, you must also select one or more VPC security groups that specify which EC2 instances and devices can connect to the DB instance.

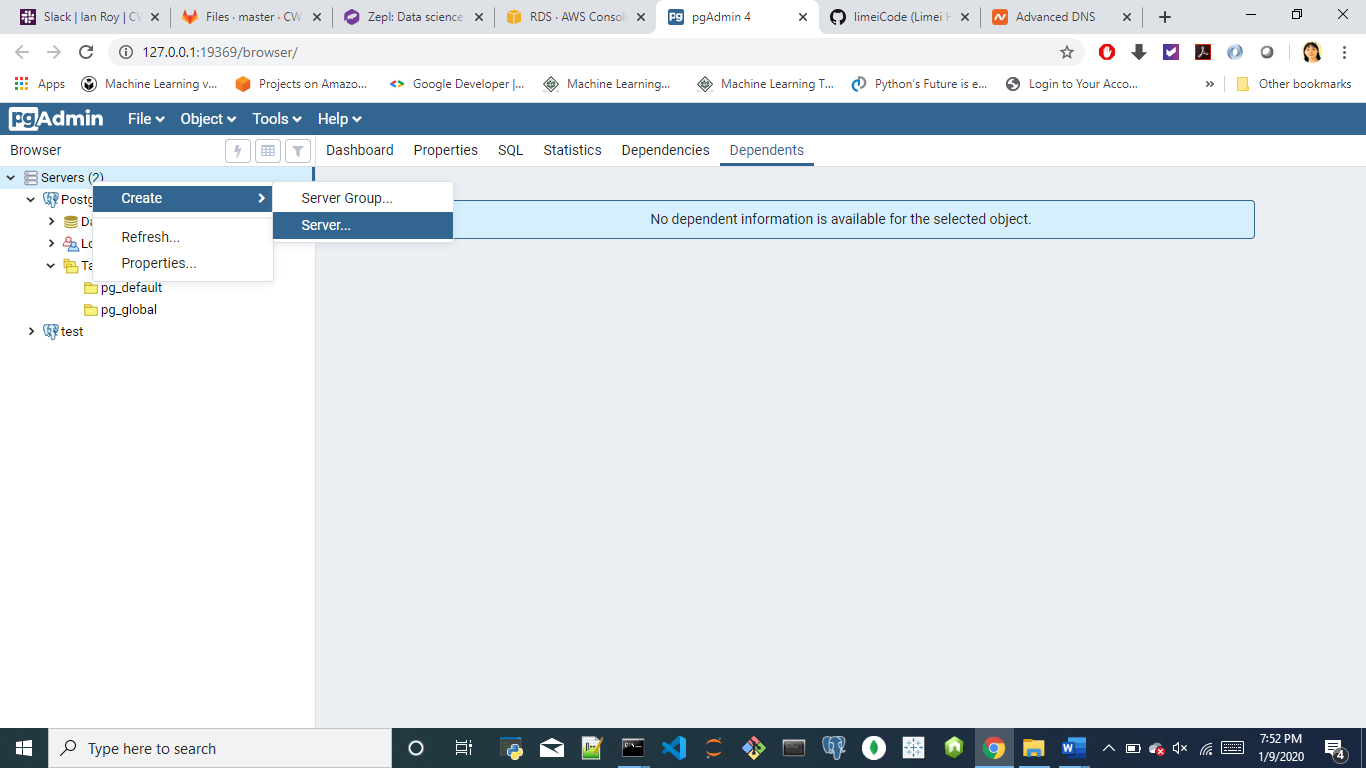


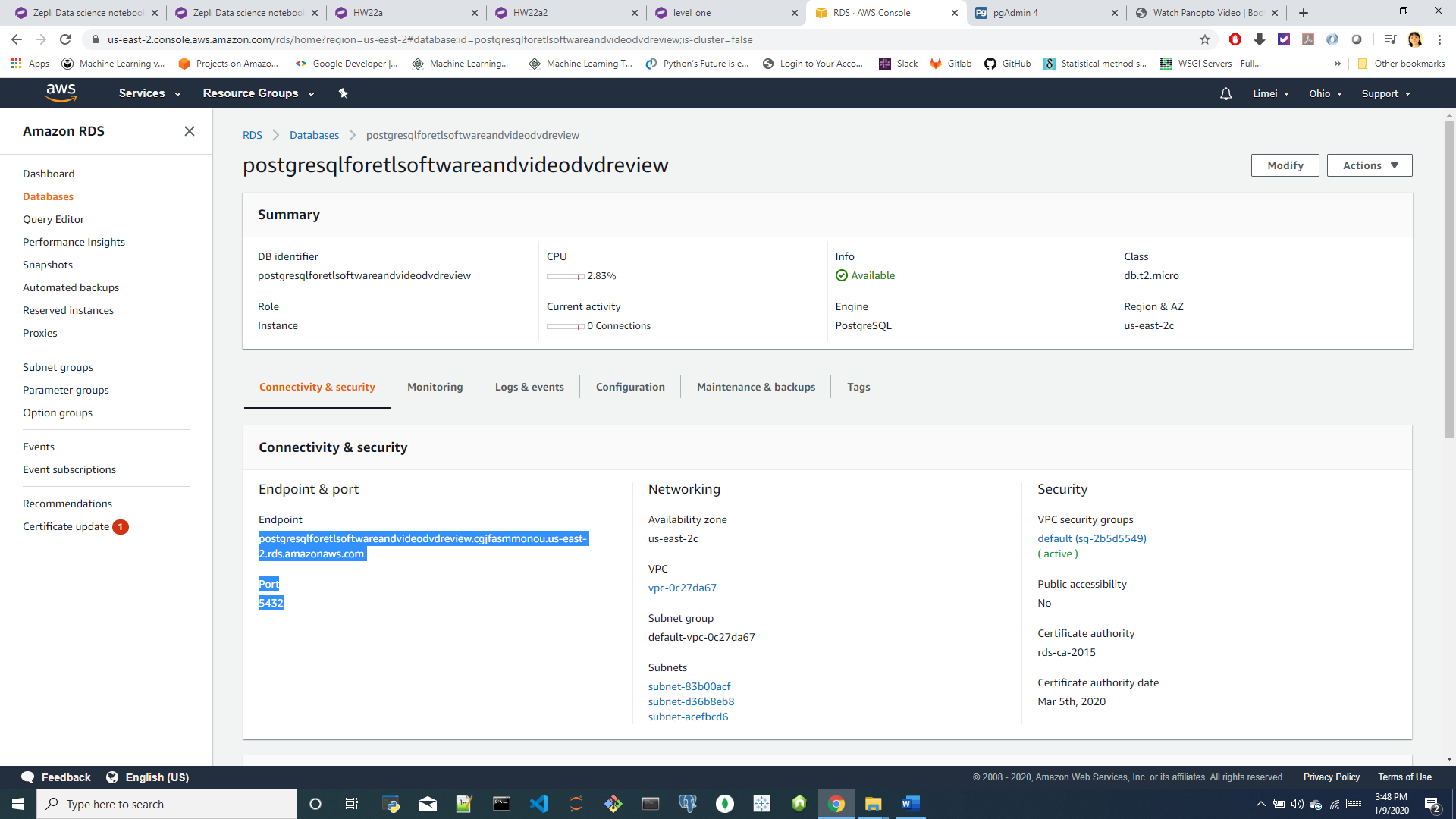
<b> add inbound rule





2) Use the furnished [schema file](file:///C:\CW-Coding-Bootcamp\CWRUCLE201907DATA4\blob\master\22-BigData\Homework\Resources\schema.sql) to create tables in pgAdmin locally





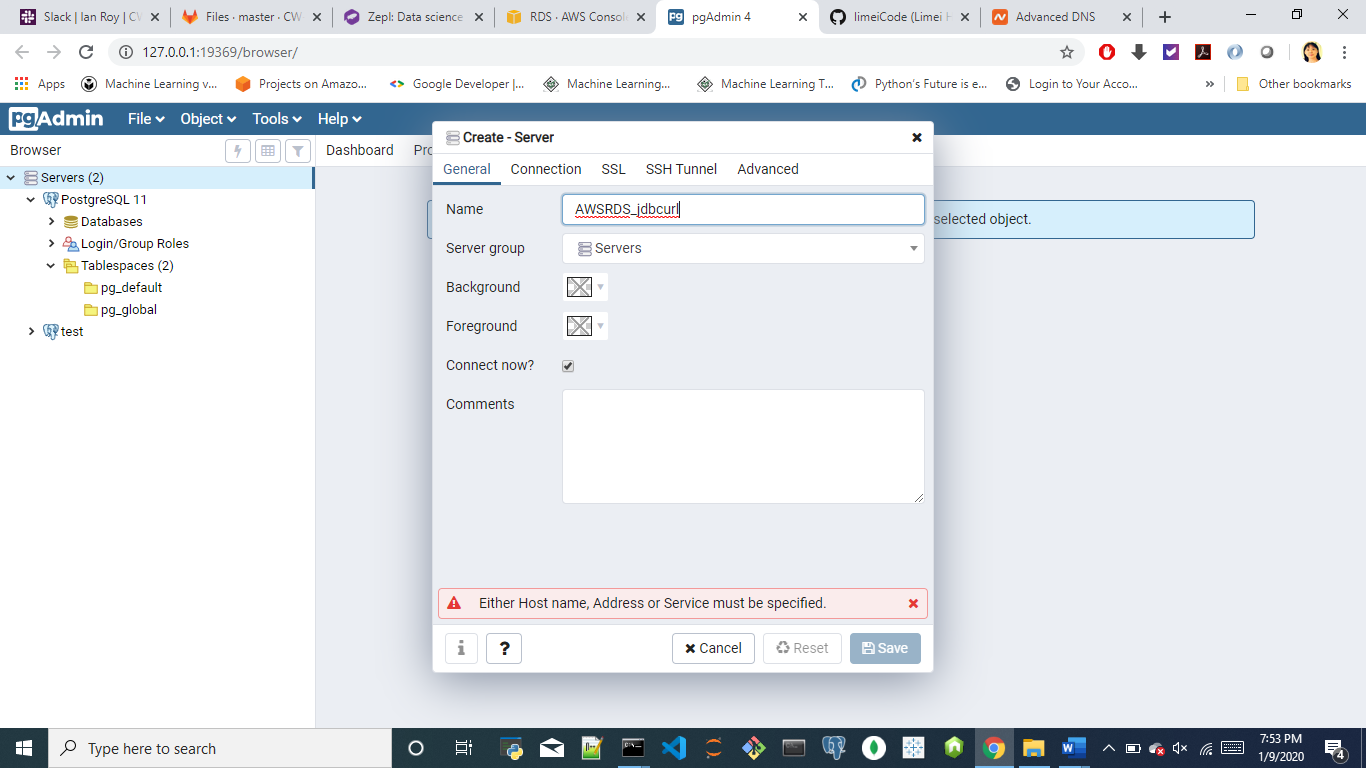
Endpoint: [ postgresqlforetlsoftwareandvideodvdreview.cgjfasmmonou.us-east-2.rds.amazonaws.com ]

Port: [ 5432 ]

DB instance identifier: [ PostgreSQLForETLSoftwareAndVideoDVDReview ] [postgresqlforetlsoftwareandvideodvdreview]

Master username: [ root ]

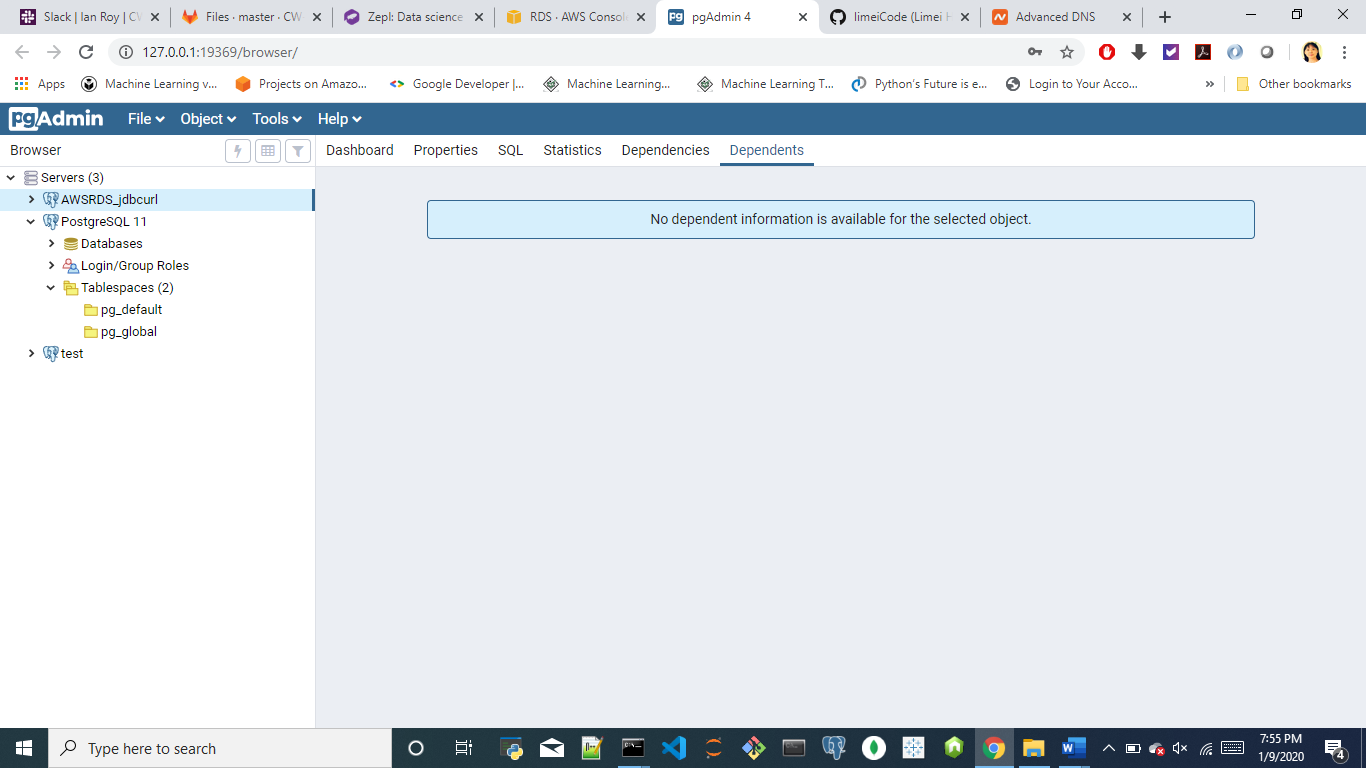
Master password: [ lmpostgre1! ]

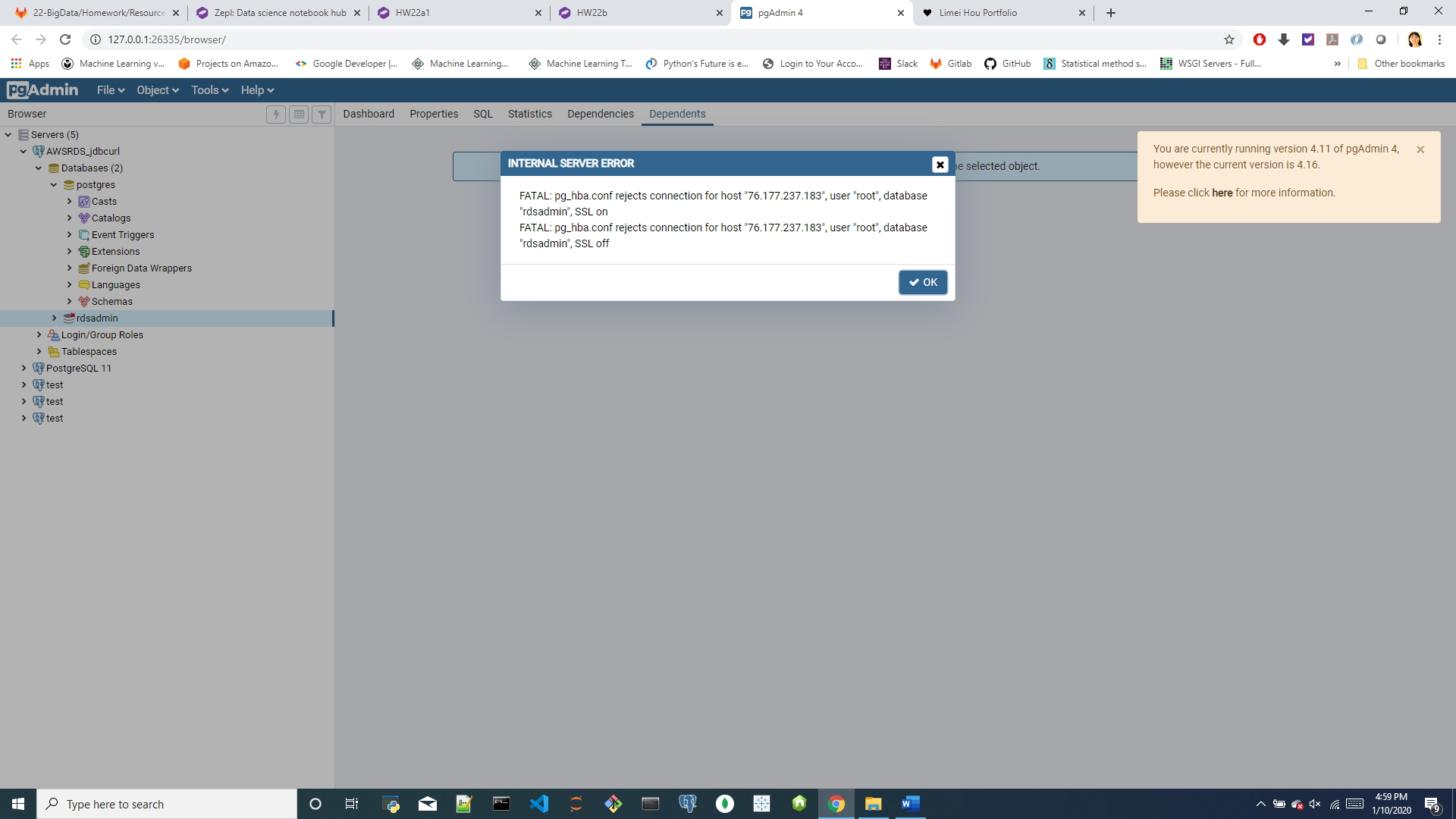


Name: any

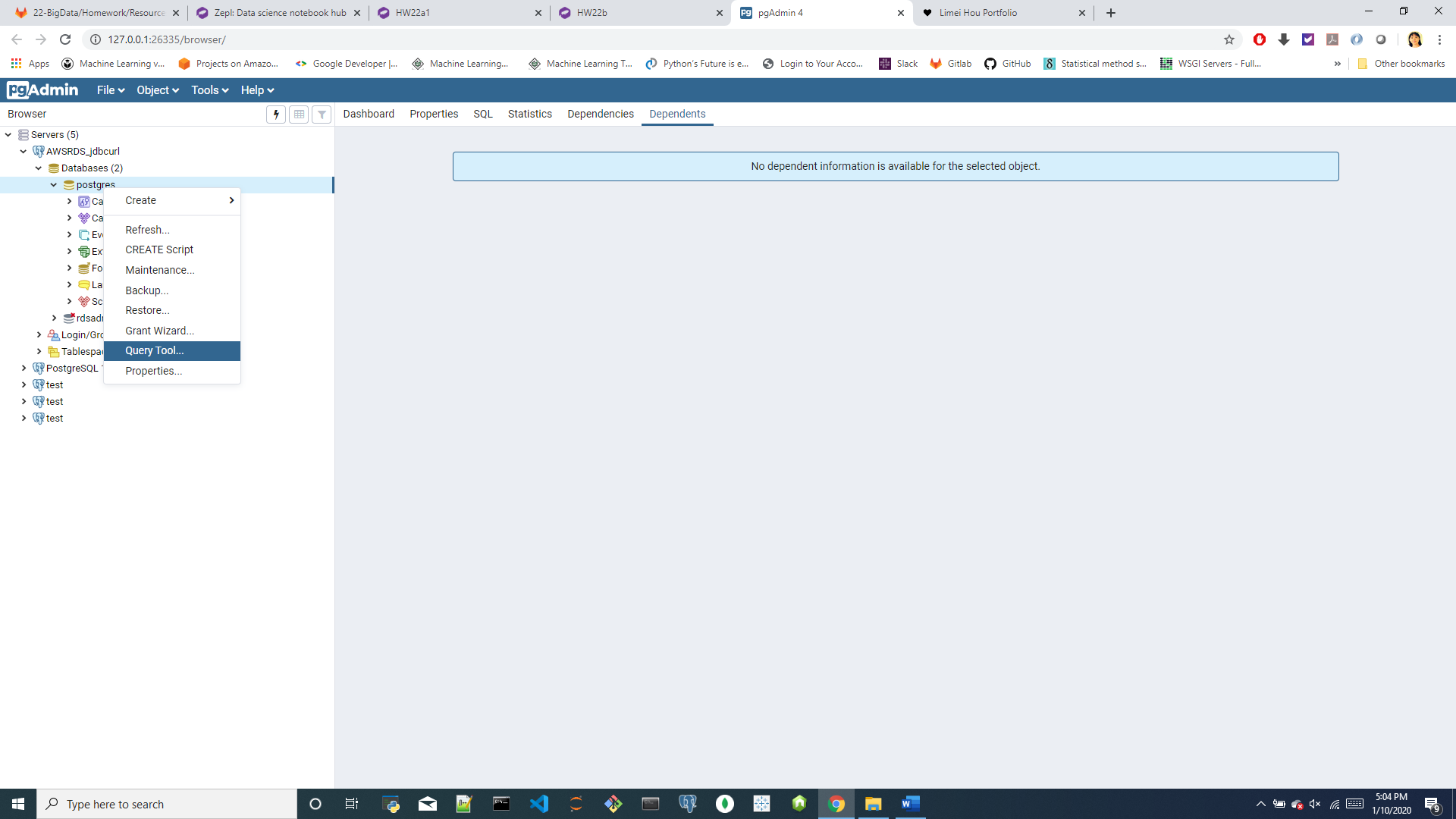


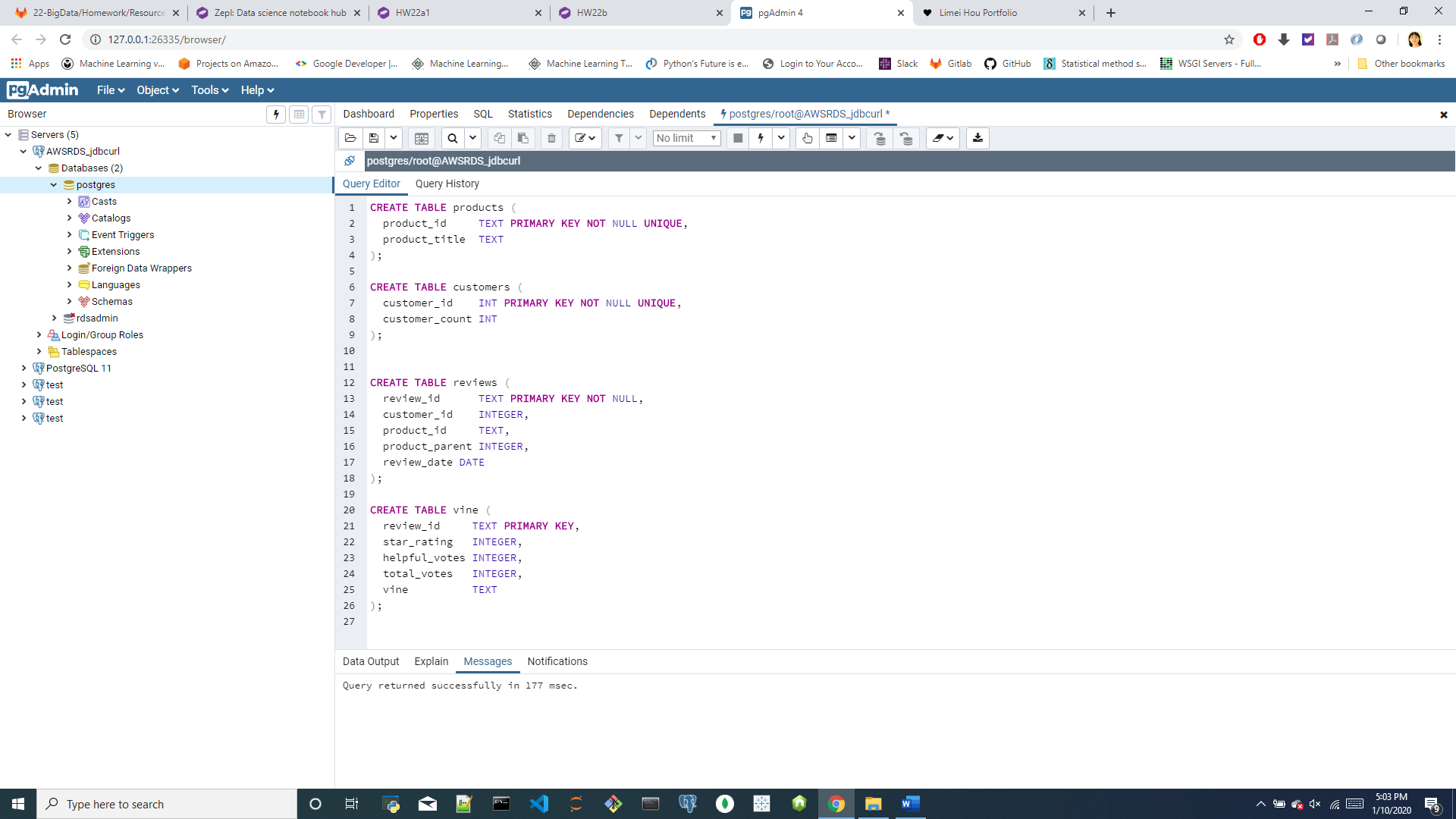
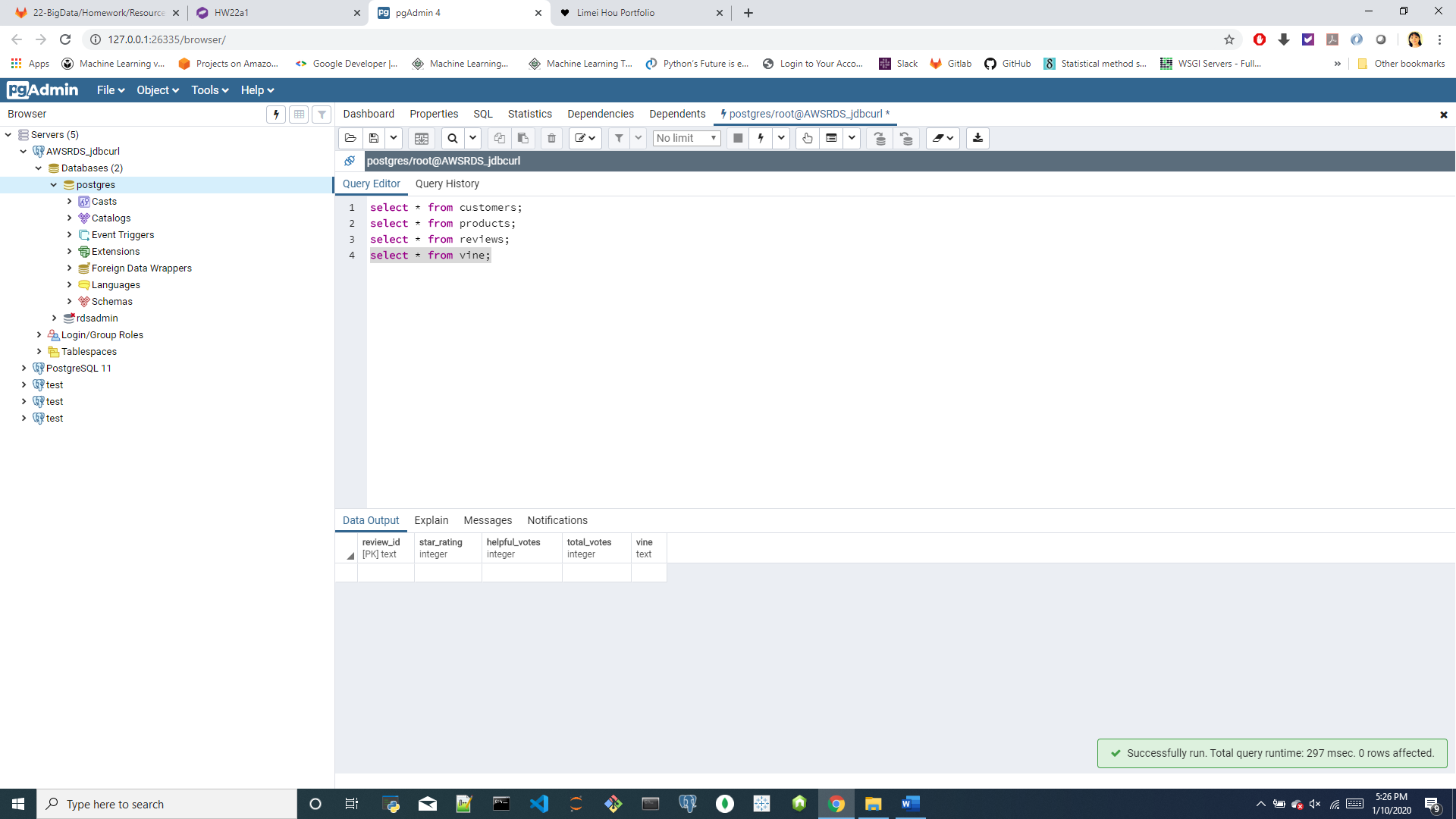
Maintenance database: must be ‘postgres’, it’s the default DB name.





Two Databases: postgres, rdsadmin



<2> Create two separate ZEPL notebooks and **extract** any two datasets from the list at [review dataset](https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt), one into each notebook.

Note1 It is possible to ETL both data sources in a single notebook, but due to the large data sizes, it will be easier to work with these S3 data sources in two separate ZEPL notebooks.

Note2 Be sure to handle the header correctly. If you read the file without the header parameter, you may find that the column headers are included in the table rows.

Note3 TSV: stands for Tab Separated Values.

For each notebook (one dataset per notebook), complete the following: <<ETL\_Software.json>> <<ETL\_Video\_DVD.json>>

1)**Read** . .tsv.gz : [spark.read.csv(SparkFiles.get(" .tsv.gz"), sep="**\t**", header=**True**, inferSchema=True)]

2)**Count** the number of records (rows) in the dataset. [df.count()]

**3)Transform** the dataset to fit the tables in the [schema file](/CW-Coding-Bootcamp/CWRUCLE201907DATA4/blob/master/22-BigData/Homework/Resources/schema.sql). Be sure the DataFrames match in data type and in column name.

**4)Load** the DataFrames that correspond to tables into an RDS instance.

This process can take up to 10 minutes for each. Be sure that everything is correct before uploading.

=<>= data sets: <https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt>

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

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Toys\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Tools\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Sports\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_**Software**\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Shoes\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Pet\_Products\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Personal\_Care\_Appliances\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_PC\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Outdoors\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Office\_Products\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Musical\_Instruments\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Music\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Mobile\_Electronics\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Mobile\_Apps\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Major\_Appliances\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Luggage\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Lawn\_and\_Garden\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Kitchen\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Jewelry\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Home\_Improvement\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Home\_Entertainment\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Home\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Health\_Personal\_Care\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Grocery\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Gift\_Card\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Furniture\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Electronics\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Digital\_Video\_Games\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Digital\_Video\_Download\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Digital\_Software\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Digital\_Music\_Purchase\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Digital\_Ebook\_Purchase\_v1\_01.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Digital\_Ebook\_Purchase\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Camera\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Books\_v1\_02.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Books\_v1\_01.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Books\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Beauty\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Baby\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Automotive\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Apparel\_v1\_00.tsv.gz

MULTILINGUAL REVIEWS DATASET:

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_multilingual\_US\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_multilingual\_UK\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_multilingual\_JP\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_multilingual\_FR\_v1\_00.tsv.gz

https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_multilingual\_DE\_v1\_00.tsv.gz

