T5 - Group 3

By-
Shubhangi Singhal - ss100
Srishti Rawat - rawat4
Tanya Gupta - tanya3
Varad Deshpande - varadad2

Option A

For Option A, we first installed curl in our windows laptops and stored a part of the data stream in a csv file named ‘live.csv’. A trigger was created by us that took care of all the required constraints. The trigger we utilized is as follows:

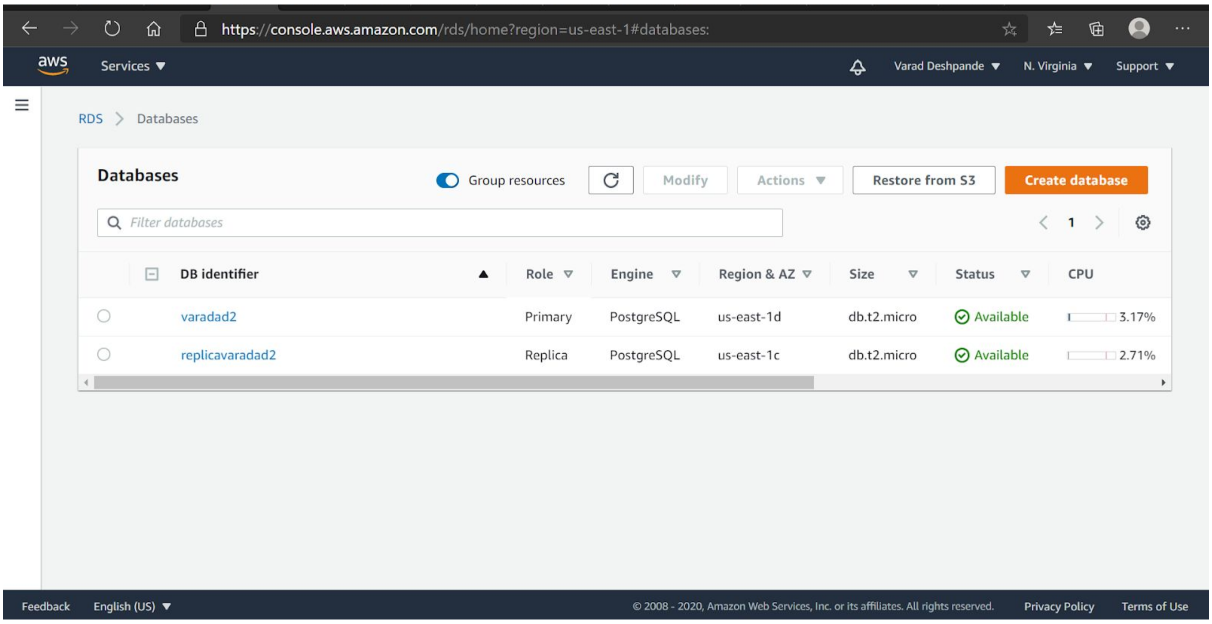
```
CREATE OR REPLACE FUNCTION insert_messages()
RETURNS trigger AS $BODY$

BEGIN
IF (new."Altitude" > 51000 or new."Altitude" is null) THEN
        new."Altitude" = (SELECT "Altitude" FROM "Messages" WHERE "AircraftHex" =
new."AircraftHex" and "Altitude" is not null and "Altitude" < 51000 ORDER BY "Altitude"
DESC LIMIT 1);
END IF;
IF (new."OnGround" = -1 or new."OnGround" is null) and (new."Altitude" > 1000 or
new."GroundSpeed" > 200 or new."Altitude" is null or new."GroundSpeed" is null) THEN
        new."OnGround" = 0;
END IF;
RETURN NEW;
END;
$BODY$
LANGUAGE 'plpgsql';

CREATE TRIGGER insert_valid BEFORE INSERT ON "Messages"
FOR EACH ROW EXECUTE PROCEDURE insert_messages();
```

Next, we tested this trigger by manually inserting values into the Messages table. While manually inserting the values, we came across some problems related to the use of single quotes, double quotes or no quotes for column names and column values. After an elaborate discussion and skimming through various different documentations, we could get the correct combination for manual insertion of records. After the tests passed, we just imported the live.csv file through pgadmin 4 to the Messages table. It took more time to execute than usual but the triggers ran successfully on the newly imported data. We ran a final check on the data to see if the trigger was applied on the specific rows that were supposed to change by the application of the trigger.

Option C

For option C, we created a replica RDS instance in one of the member’s AWS accounts. After creating the replica, endpoint information of that replica was shared with other team members. At the same time the server computer created a new table in the original DB instance and this got reflected in the connected/client servers on other’s pgadmin.

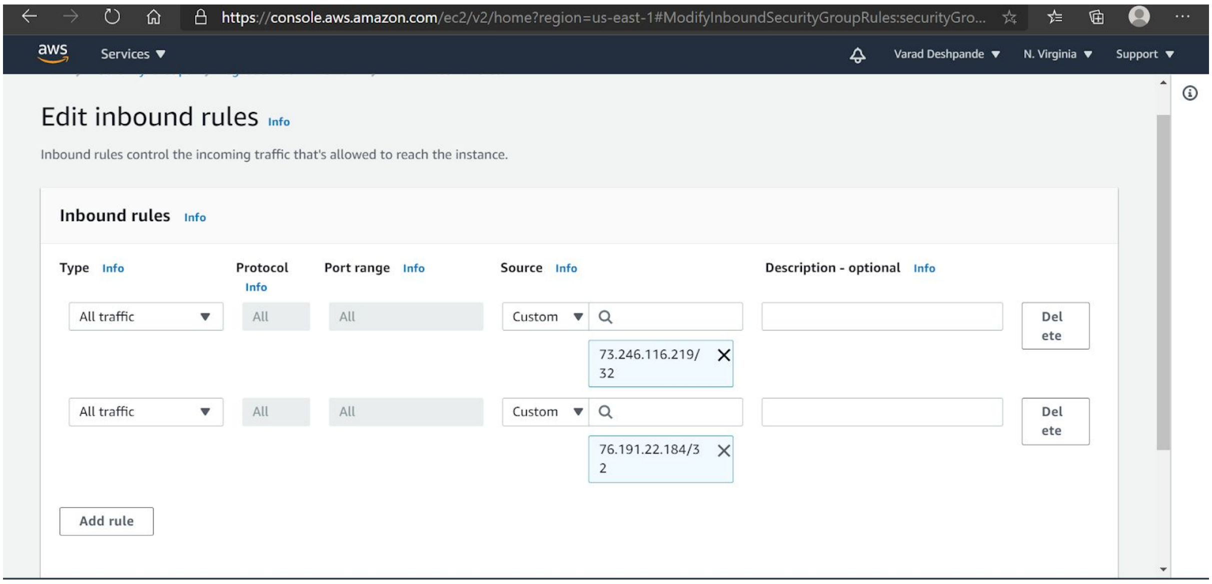
1) Creating a Replica of DB instance.



DB identifier	Role	Engine	Region & AZ	Size	Status	CPU
varadad2	Primary	PostgreSQL	us-east-1d	db.t2.micro	Available	3.17%
replicavaradad2	Replica	PostgreSQL	us-east-1c	db.t2.micro	Available	2.71%

Feedback
English (US) ▼
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

2) Adding public IP of teammates in Inbound rules for the replica



https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#ModifyInboundSecurityGroupRules:securityGro...
aws
Services ▼
Varad Deshpande ▼
N. Virginia ▼
Support ▼

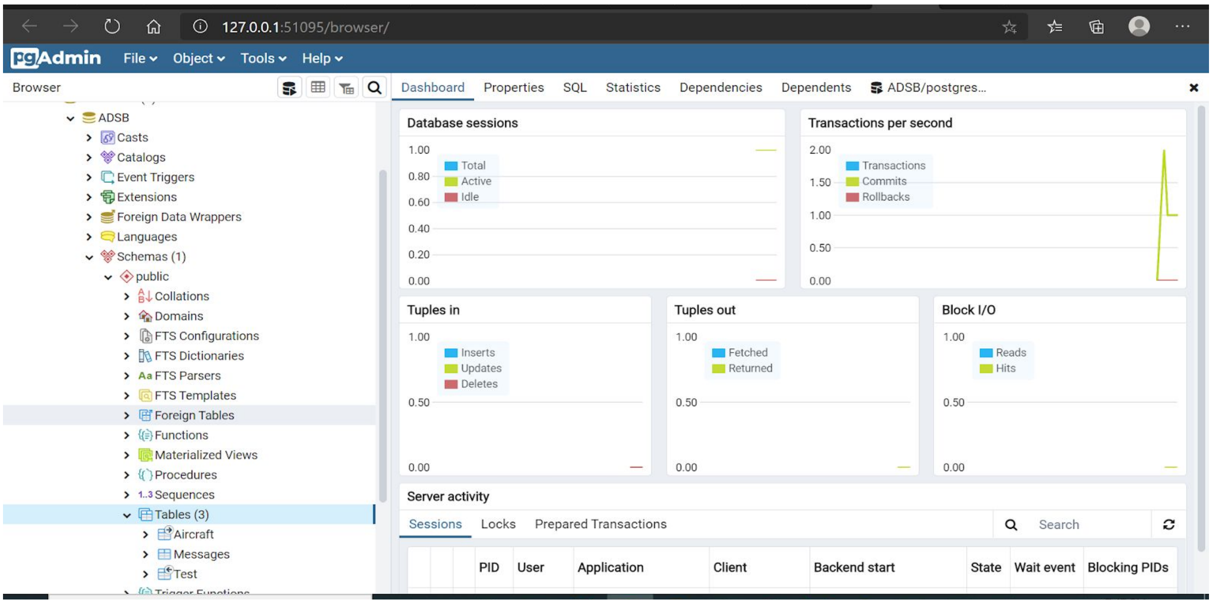
Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

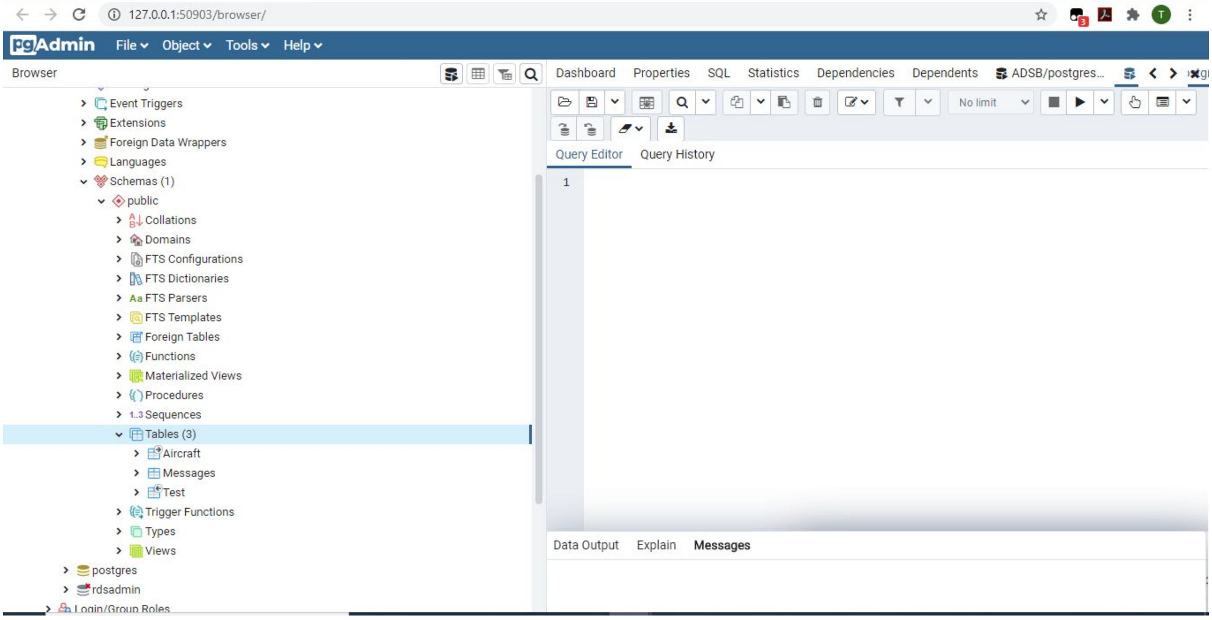
Inbound rules Info

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info	
All traffic	All	All	Custom 73.246.116.219/32		Delete
All traffic	All	All	Custom 76.191.22.184/32		Delete

3) Creating a new table in Varad's (Intake Server) DB Instance.



4) New Table getting reflected in Receiver's DB Instance


127.0.0.1:50903/browser/
pgAdmin
File
Object
Tools
Help
Browser
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Schemas (1)
public
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Procedures
Sequences
Tables (3)
Aircraft
Messages
Test
Trigger Functions
Types
Views
postgres
rdsadmin
Login/Group Roles
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
ADSB/postgres...
No limit
Query Editor
Query History
1
Data Output
Explain
Messages