



# **Amazon Web Services Data Engineering Immersion Day**

---

Database Migration Services Instructor Setup Instructions  
Jun 2019

## Table of Contents

<b><i>Limit Instruction:</i></b> .....	<b>2</b>
<b><i>Introduction</i></b> .....	<b>2</b>
<b><i>Create the Instructor Environment</i></b> .....	<b>3</b>
<b><i>Access Database from SQL Client (Optional)</i></b> .....	<b>5</b>
<b><i>Create the Change Data Capture Environment (Optional)</i></b> .....	<b>7</b>
<b><i>Generate the CDC Data (Optional)</i></b> .....	<b>12</b>
<b><i>Appendix: AWS CloudFormation Template</i></b> .....	<b>14</b>

## Limit Instruction:

This immersion day required each student to have their own account. If you are sharing single account with multiple students by creating a multiple IAM users, Account can hit following default service limit:

- VPC – VPCs per Region 5
- Glue - Number of crawlers per account 50
- Glue - Number of concurrent jobs runs per account 50
- Glue - Maximum DPUs used by a role at one time 300
- S3 – Number of buckets per account 100
- Athena - Number of DDL queries you can submit at the same time 20
- Athena - Number of DML queries you can submit at the same time 20
- RDS – Make sure you have enough disk space available in your RDS instance, if want to run DMS Change Data Capture (CDC) as generating large amount of data can exhaust RDS disk space.
- DMS - Make sure you have enough disk space available in your DMS replication instance, if want to run DMS Change Data Capture (CDC) as transferring large amount of CDC data can exhaust disk space.

## Introduction

**\*\*\*Make sure you select the us-east-1 (Virginia) region\*\*\***

The Database Migration Services (DMS) hands-on lab provide a scenario, where participant learns to hydrate Amazon S3 data lake with a relation database. To achieve that, participants need a source endpoint and this guide helps instructors set up a PostgreSQL database with public endpoint as the source database.

In this lab, you will complete the following tasks:

1. Create the source database environment.
2. Hydrate the source database environment.
3. Update the source database environment to demonstrate CDC replication within DMS.

Relevant information about this lab:

- Expected setup time: 45 minutes
- Source database name: sportstickets
- Source schema name: dms\_sample

Instructor will provide source database details to participants during main lab to configure source endpoint.

## Create the Instructor Environment

In this section, you are going to create a PostgreSQL RDS instance as data source for AWS Data Migration Service to consume by lab attendees for data migration to Amazon S3 data lake.

1. Sign in to the Console where you will host the source database environment.
2. Navigate to the AWS CloudFormation page.
3. Launch a new stack with the AWS CloudFormation template DMSLab\_instructor\_CFN.json provided with your lab package. Make sure to select us-east-1 (Virginia) region.

Alternatively, You can follow instruction in [Appendix : AWS CloudFormation Template](#) to create your AWS CloudFormation template for this lab.

4. Give stack name and Enter the Key Pair to use. Please make sure to create an Amazon EC2 Key pair if don't have one in select us-east-1 (Virginia) region. Follow User guide [Amazon EC2 key pairs](#) to create a key pair.

### Specify Details

Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. [Learn more.](#)

Stack name

### Parameters

KeyName   
Name of an existing EC2 KeyPair to enable SSH access to the instance

5. Enter a tag for the Name that identifies the resources as part of this lab.
6. Launch the stack. It may take 15 minutes for the stack to launch. This stack creates a new VPC, Subnets, Security groups, EC2 instance, Route table, Routes, and an RDS Postgres instance with sample data. You can see all resources listed below:

## Database Migration Services Instructor Environment for the Lab

Create Stack

Actions

Design template

Filter: Active

By Stack Name

Stack Name	Created Time	Status	Drift Status	Description
<input checked="" type="checkbox"/> dmslab-instructor	2019-01-08 15:58:56 UTC-0800	CREATE_COMPLETE	NOT_CHECKED	DMS Lab Instructor account

Overview

Outputs

Resources

Events

Template

Parameters

Tags

Stack Policy

Change Sets

Rollback Triggers

Logical ID	Physical ID	Type	Drift Status	Status
EC2SubNet	subnet-0b52bedc2f53c4806	AWS::EC2::Subnet	NOT_CHECKED	CREATE_COMPLETE
RDSSubNet	subnet-0a4541c3d13362f6e	AWS::EC2::Subnet	NOT_CHECKED	CREATE_COMPLETE
RDSSubNet2	subnet-0b7b1101235fae8bd	AWS::EC2::Subnet	NOT_CHECKED	CREATE_COMPLETE
dbsgdefault	dmslab-instructor-dbsgdefault-93k8h7xcv9t	AWS::RDS::DBSecurityGroup	NOT_CHECKED	CREATE_COMPLETE
dbsubnetdefaultdmsinstructor...	dmslab-instructor-dbsubnetdefaultdmsinstructorvpc-14tq88sqfi65a	AWS::RDS::DBSubnetGroup	NOT_CHECKED	CREATE_COMPLETE
dchpassoc1	dmsla-dchpa-198OI556YFPL6	AWS::EC2::VPCHCOptionsAssociation	NOT_CHECKED	CREATE_COMPLETE
dmsinstructorvpc	vpc-01130773950fbc5a0	AWS::EC2::VPC	NOT_CHECKED	CREATE_COMPLETE
dopt1cc25278	dopt-0740d448dbbda82fa	AWS::EC2::DHCOOptions	NOT_CHECKED	CREATE_COMPLETE
egress1	dmslab-instructor-egress1-1JBKPM7WRZX	AWS::EC2::SecurityGroupEgress	NOT_CHECKED	CREATE_COMPLETE
egress2	dmslab-instructor-egress2-1JZ253WG1P2VW	AWS::EC2::SecurityGroupEgress	NOT_CHECKED	CREATE_COMPLETE
gw1	dmslab-gw1-19XRYSAATKUDQ	AWS::EC2::VPCGatewayAttachment	NOT_CHECKED	CREATE_COMPLETE
igw0887475a258f00277	igw-01d39e871fd8aa97a	AWS::EC2::InternetGateway	NOT_CHECKED	CREATE_COMPLETE
ingress1	ingress1	AWS::EC2::SecurityGroupIngress	NOT_CHECKED	CREATE_COMPLETE
ingress2	ingress2	AWS::EC2::SecurityGroupIngress	NOT_CHECKED	CREATE_COMPLETE
ingress3	ingress3	AWS::EC2::SecurityGroupIngress	NOT_CHECKED	CREATE_COMPLETE
ingress4	ingress4	AWS::EC2::SecurityGroupIngress	NOT_CHECKED	CREATE_COMPLETE
instance0f63b887480639040	i-0c8993117482a85ef	AWS::EC2::Instance	NOT_CHECKED	CREATE_COMPLETE
rdsdmslabdb	dmslabinstance	AWS::RDS::DBInstance	NOT_CHECKED	CREATE_COMPLETE
route1	dmsla-route-1RWYBCQ9W1EP	AWS::EC2::Route	NOT_CHECKED	CREATE_COMPLETE

- Once the stack is launched, navigate to the Amazon Relational Database Service (Amazon RDS) page and select **Instances** > **dmslabinstance** and Copy the instance Endpoint information as shown in below screenshot

[RDS](#) > [Databases](#) > dmslabinstance

### dmslabinstance

Modify

#### Summary

DB Name dmslabinstance	CPU <div>0.63%</div>	Info Available	Class db.t2.xlarge
Role Instance	Current activity <div>0 Connections</div>	Engine PostgreSQL	Region & AZ us-east-1c

Connectivity

Monitoring

Logs & events

Configuration

Maintenance & backups

Tags

#### Connectivity

##### Endpoint & port

Endpoint  
dmslabinstance.ccla1oozkrry.us-east-1.rds.amazonaws.com

Port  
5432

##### Networking

Availability zone  
us-east-1c

VPC  
DMSLabSourceDB (vpc-01130773950fbc5a0)

Subnet group  
dmslab-instructor-dbsubnetdefaultdmsinstructorvpc-14tq88sqfi65a

Subnets  
subnet-0b52bedc2f53c4806  
subnet-0b7b1101235fae8bd  
subnet-0a4541c3d13362f6e

##### Security

VPC security groups  
dmslab-instructor-sgrdsiaunchizard2-17GR8DN2692Q4 (sg-0e1160b1d20be6573) (active)

Public accessibility  
Yes

Certificate authority  
rds-ca-2015

Certificate authority date  
Mar 5th, 2020

4

The github repository for aws-database-migration-samples data is located here:  
<https://github.com/aws-samples/aws-database-migration-samples/tree/master/PostgreSQL/sampledb/v1>

You can read through the documentation to better understand the source database environment.

## Access Database from SQL Client (Optional)

You can follow below instruction to setup SQL Workbench to access your Postgre Database from SQL client:

<https://aws.amazon.com/getting-started/tutorials/create-connect-postgresql-db/>

In SQL Workbench:

Run following query to find out all Schema and table created.

```
SELECT * FROM pg_catalog.pg_tables;
```

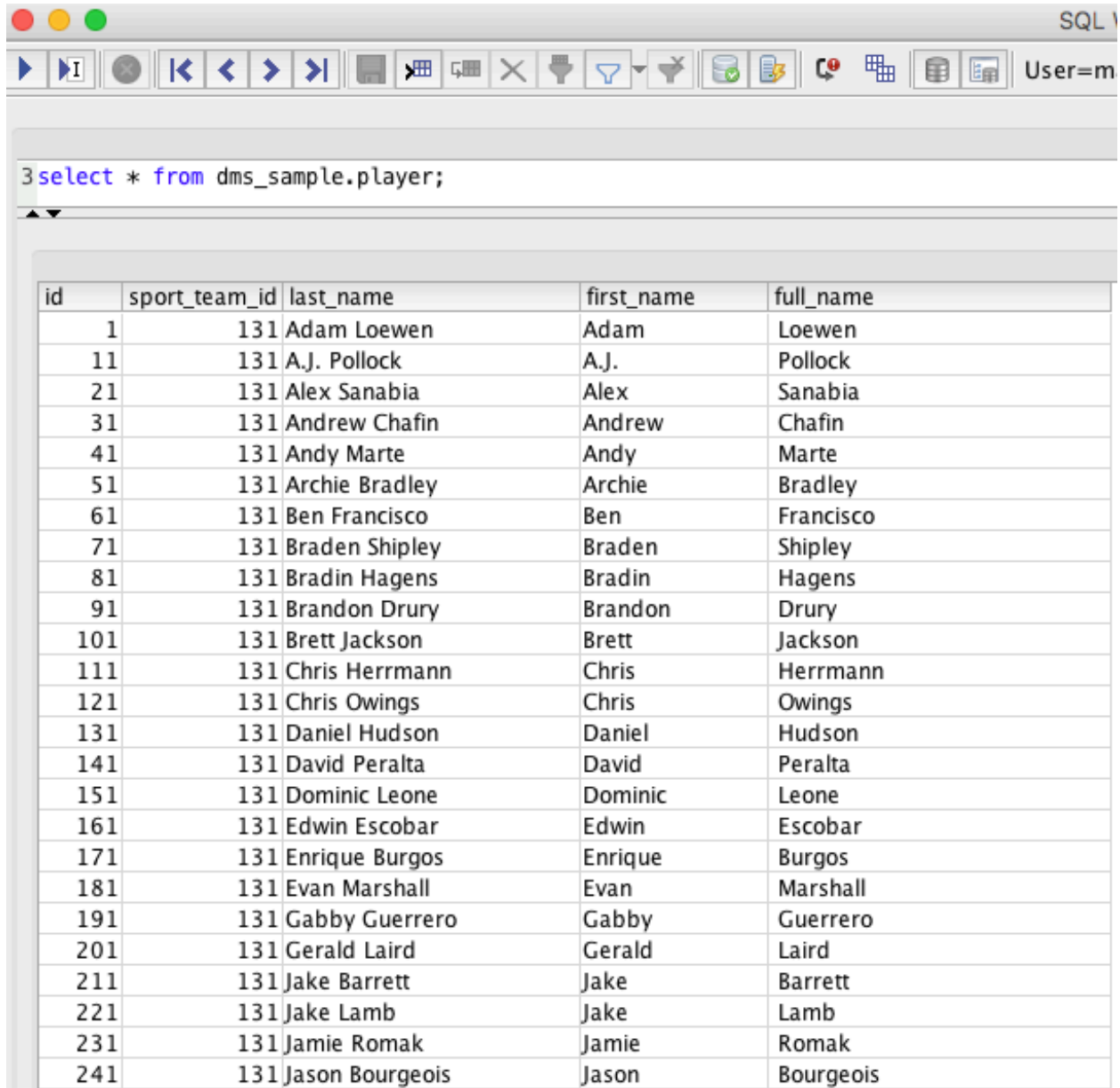
schemaname	tablename	tableowner	tablespace	hasindexes	hasrules	hastriggers	rowsecurity
dms_sample	player	master		true	false	true	false
dms_sample	seat_type	master		true	false	true	false
dms_sample	seat	master		true	false	true	false
dms_sample	sport_division	master		true	false	true	false
dms_sample	sport_league	master		true	false	true	false
pg_catalog	pg_statistic	rdsadmin		true	false	false	false
pg_catalog	pg_type	rdsadmin		true	false	false	false
pg_catalog	pg_policy	rdsadmin		true	false	false	false
pg_catalog	pg_authid	rdsadmin	pg_global	true	false	false	false
dms_sample	mlb_data	master		false	false	false	false
dms_sample	name_data	master		true	false	false	false
dms_sample	nfl_data	master		false	false	false	false
dms_sample	nfl_stadium_data	master		false	false	false	false
dms_sample	sport_type	master		true	false	true	false
dms_sample	person	master		true	false	true	false
dms_sample	sport_location	master		true	false	true	false
dms_sample	sport_team	master		true	false	true	false
dms_sample	sporting_event_ticket	master		true	false	true	false
dms_sample	sporting_event	master		true	false	true	false
dms_sample	ticket_purchase_hist	master		true	false	true	false
pg_catalog	pg_user_mapping	rdsadmin		true	false	false	false
pg_catalog	pg_subscription	rdsadmin	pg_global	true	false	false	false
pg_catalog	pg_attribute	rdsadmin		true	false	false	false
pg_catalog	pg_proc	rdsadmin		true	false	false	false
pg_catalog	pg_class	rdsadmin		true	false	false	false
pg_catalog	pg_attrdef	rdsadmin		true	false	false	false
pg_catalog	pg_constraint	rdsadmin		true	false	false	false
pg_catalog	pg_inherits	rdsadmin		true	false	false	false

Use following query to analyze a table

```
select * from schemaname.tablename;
```

For example:

```
select * from dms_sample.player;
```



The screenshot shows a SQL IDE window with a toolbar at the top containing various icons for navigation and execution. The query editor displays the command: `3 select * from dms_sample.player;`. Below the editor, the results are shown in a table format with 5 columns: `id`, `sport_team_id`, `last_name`, `first_name`, and `full_name`. The table contains 24 rows of player data.

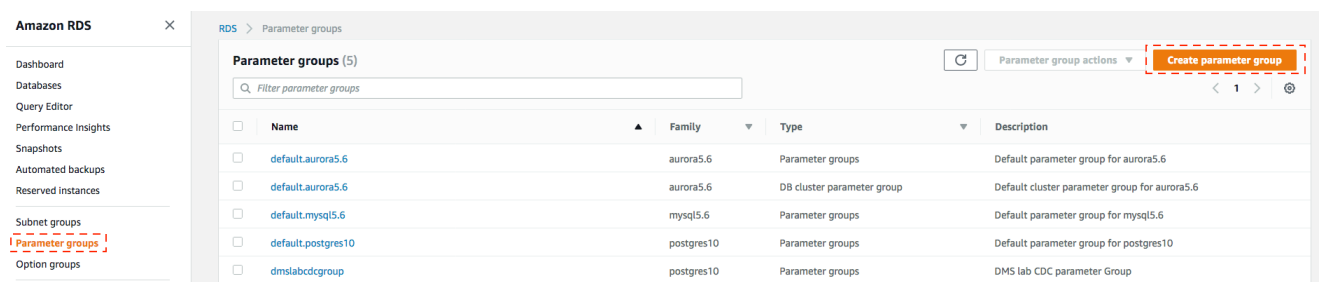
id	sport_team_id	last_name	first_name	full_name
1	131	Adam Loewen	Adam	Loewen
11	131	A.J. Pollock	A.J.	Pollock
21	131	Alex Sanabia	Alex	Sanabia
31	131	Andrew Chafin	Andrew	Chafin
41	131	Andy Marte	Andy	Marte
51	131	Archie Bradley	Archie	Bradley
61	131	Ben Francisco	Ben	Francisco
71	131	Braden Shipley	Braden	Shipley
81	131	Bradin Hagens	Bradin	Hagens
91	131	Brandon Drury	Brandon	Drury
101	131	Brett Jackson	Brett	Jackson
111	131	Chris Herrmann	Chris	Herrmann
121	131	Chris Owings	Chris	Owings
131	131	Daniel Hudson	Daniel	Hudson
141	131	David Peralta	David	Peralta
151	131	Dominic Leone	Dominic	Leone
161	131	Edwin Escobar	Edwin	Escobar
171	131	Enrique Burgos	Enrique	Burgos
181	131	Evan Marshall	Evan	Marshall
191	131	Gabby Guerrero	Gabby	Guerrero
201	131	Gerald Laird	Gerald	Laird
211	131	Jake Barrett	Jake	Barrett
221	131	Jake Lamb	Jake	Lamb
231	131	Jamie Romak	Jamie	Romak
241	131	Jason Bourgeois	Jason	Bourgeois

Following sections are optional you only need to execute, if you want to show change data capture replication with DMS.

## Create the Change Data Capture Environment (Optional)

If you are planning to show ongoing CDC capability you should also set the following attributes:

1. Create a custom DB parameter group in RDS console for postgres10. Go to Amazon RDS Parameter groups and click on Create Parameter group button as shown below:



The screenshot shows the 'Create parameter group' form in the Amazon RDS console. The breadcrumb navigation is 'RDS > Parameter groups > Create parameter group'. The form title is 'Create parameter group'. Below the title is a section 'Parameter group details' with the instruction: 'To create a parameter group, choose a parameter group family, then name and describe your parameter group'. The form contains three main sections:

- Parameter group family:** A dropdown menu labeled 'DB family that this DB parameter group will apply to' with 'postgres10' selected.
- Group name:** A text input field labeled 'Identifier for the DB parameter group' containing 'dataeng-dms-group'.
- Description:** A text input field labeled 'Description for the DB parameter group' containing 'Parameter for Ticket Database'.

At the bottom right of the form are two buttons: 'Cancel' and 'Create'.

2. In your custom parameter group, you should:



## Database Migration Services Instructor Environment for the Lab

- Set `rds.logical_replication` to 1. This is a static parameter that requires a reboot of the DB instance for the parameter to take effect .
- Set the `wal_sender_timeout` parameter to 0. Setting this parameter to 0 prevents PostgreSQL from terminating replication connections that are inactive longer than the specified timeout.
- Increase `max_wal_senders` parameter from 10 to 20 to accommodate for Data Migration Service.

RDS > Parameter groups > dataeng-dms-group > Preview changes

### Preview changes

Parameter	Current Value	New Value
max_wal_senders	10	20
rds.logical_replication	0	1
wal_sender_timeout	30000	0

[Close](#) [Save changes](#)

- Go to your RDS Databases and click on modify.

Amazon RDS

- Dashboard
- Databases**
- Query Editor
- Performance Insights
- Snapshots
- Automated backups
- Reserved instances
- Subnet groups

RDS > Databases > dmslabinstance

dmslabinstance

[Modify](#) [Actions](#)

#### Summary

DB Identifier dmslabinstance	CPU 0.63%	Info Available	Class db.t2.xlarge
Role Instance	Current activity 0 Connections	Engine PostgreSQL	Region & AZ us-east-1c

- Scroll down to **Database options** section, Modify the RDS instance we created, and associate the custom parameter group with the RDS instead of the default parameter group, and click continue at bottom of screen.

### Database options

**Database port**

Specify the TCP/IP port that the DB instance will use for application connections. The connection string of any application connecting to the DB instance must specify the port number of the DB instance. Both the security group applied to the DB instance and your company's firewalls must allow connections to the port. [Learn More](#)

**DB parameter group**

Database parameter group to associate with this DB instance

**IAM DB authentication** [Info](#)

☐ Enable IAM DB authentication  
 Manage your database user credentials through AWS IAM users and roles.

☒ Disable

5. Review changes, select **Apply immediately**.

RDS > Databases > Modify

## Modify DB Instance: dmslabinstance

**Summary of modifications**

You are about to submit the following modifications. Only values that will change are displayed. Carefully verify your changes and click Modify DB Instance.

Attribute	Current value	New value
DB parameter group	default.postgres10	dataeng-dms-group
Master password		*****

**Scheduling of modifications**

When to apply modifications

☐ Apply during the next scheduled maintenance window  
 Current maintenance window: sun:05:20-sun:05:50

☒ **Apply immediately**  
 The modifications in this request and any pending modifications will be asynchronously applied as soon as possible, regardless of the maintenance window setting for this database instance.

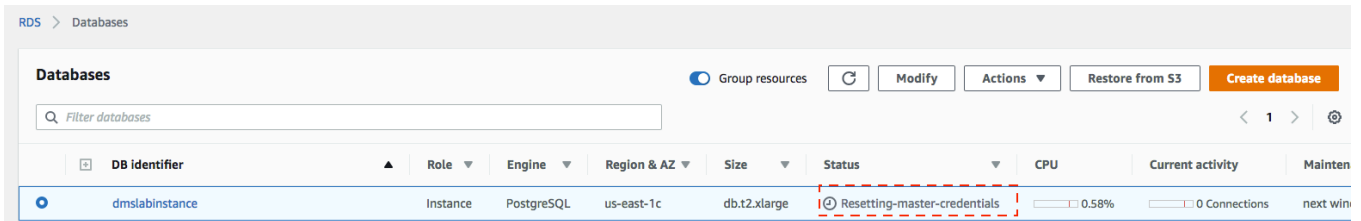
**Potential unexpected downtime**

If you choose to apply changes immediately, please note that any changes in the pending modifications queue are also applied. If any of the pending modifications require downtime, choosing this option can cause unexpected downtime.

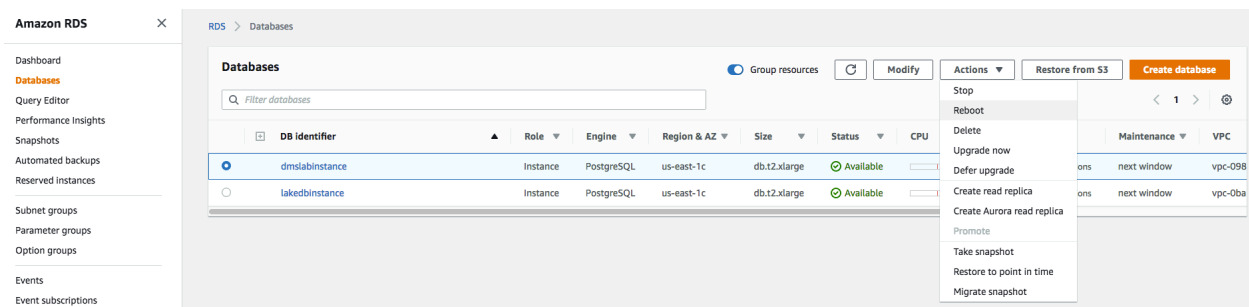
Cancel

6. click on **Modify DB Instance** button.

7. Wait until status change to available.



8. Click on the Actions drop down at reboot the db instances as shown below:



Wait until status change to available.

9. Now SSH to your ec2 instance and run following:

```
psql --host=<instanceaddress> --port=5432 --dbname=sportstickets --
username=master
```

For example : `nohup psql --host=dmslabinstance.ccla1oozkrry.us-east-1.rds.amazonaws.com --port=5432 --dbname=sportstickets --username=master`

enter the password "master123" when prompted, then you can run the following SQL script to create the wrappers needed for DMS CDC replication:

```
BEGIN;
CREATE SCHEMA IF NOT EXISTS fnRenames;
CREATE OR REPLACE FUNCTION fnRenames.pg_switch_xlog()
RETURNS pg_lsn AS $$
    SELECT pg_switch_wal(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION fnRenames.pg_xlog_replay_pause()
RETURNS VOID AS $$
    SELECT pg_wal_replay_pause(); $$ LANGUAGE SQL;
```

## Database Migration Services Instructor Environment for the Lab

```
CREATE OR REPLACE FUNCTION
fnRenames.pg_xlog_replay_resume() RETURNS VOID AS $$
    SELECT pg_wal_replay_resume(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_current_xlog_location() RETURNS pg_lsn AS $$
    SELECT pg_current_wal_lsn(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_is_xlog_replay_paused() RETURNS boolean AS $$
    SELECT pg_is_wal_replay_paused(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION fnRenames.pg_xlogfile_name(lsn
pg_lsn) RETURNS TEXT AS $$
    SELECT pg_walfile_name(lsn); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_last_xlog_replay_location() RETURNS pg_lsn AS $$
    SELECT pg_last_wal_replay_lsn(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_last_xlog_receive_location() RETURNS pg_lsn AS $$
    SELECT pg_last_wal_receive_lsn(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_current_xlog_flush_location() RETURNS pg_lsn AS
$$
    SELECT pg_current_wal_flush_lsn(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_current_xlog_insert_location() RETURNS pg_lsn AS
$$
    SELECT pg_current_wal_insert_lsn(); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_xlog_location_diff(lsn1 pg_lsn, lsn2 pg_lsn)
RETURNS NUMERIC AS $$
    SELECT pg_wal_lsn_diff(lsn1, lsn2); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_xlogfile_name_offset(lsn pg_lsn, OUT TEXT, OUT
INTEGER) AS $$
    SELECT pg_walfile_name_offset(lsn); $$ LANGUAGE SQL;
CREATE OR REPLACE FUNCTION
fnRenames.pg_create_logical_replication_slot(slot_name name,
plugin name,
    temporary BOOLEAN DEFAULT FALSE, OUT slot_name name, OUT
xlog_position pg_lsn) RETURNS RECORD AS $$
    SELECT slot_name::NAME, lsn::pg_lsn FROM
pg_catalog.pg_create_logical_replication_slot(slot_name,
plugin,
    temporary); $$ LANGUAGE SQL;

ALTER user master SET search_path to fnRenames, pg_catalog,
"$user", public;
COMMIT;
```

Details on the above script can be found here , You can also copy from below docs and change user name :

[https://docs.aws.amazon.com/dms/latest/userguide/CHAP\\_Source.PostgreSQL.html#CHAP\\_Source.PostgreSQL.v10](https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Source.PostgreSQL.html#CHAP_Source.PostgreSQL.v10)

## Generate the CDC Data (Optional)

When you want to generate transactions to demonstrate DMS CDC (Change Data Capture) functionality you can execute the following commands:

```
psql --host=<instanceaddress> --port=5432 --dbname=sportstickets --username=master
```

enter the password "master123" when prompted, then you can execute the following within the psql command prompt (sportstickets=>)

The following will generate 1000 ticket sales in batches of 1-6 tickets to randomly selected people for a random price (within a range.) A record of each transaction is recorded in the ticket\_purchase\_hist table:

```
select dms_sample.generateticketactivity(1000);
```

```
[ec2-user@ip-10-0-0-40 ~]$ psql --host=dmslabinstance.ciny3gywsvdz.us-east-1.rds.amazonaws.com --port=5432 --dbname=sportstickets --username=master
Password for user master:
psql (9.2.24, server 10.4)
WARNING: psql version 9.2, server version 10.0.
         Some psql features might not work.
SSL connection (cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256)
Type "help" for help.

sportstickets=> select dms_sample.generateticketactivity(1000);
| generateticketactivity
|-----
(1 row)

sportstickets=> █
```

Run following query in SQL Workbench to see changes in table:

```
select * from dms_sample.ticket_purchase_hist order by transaction_date_time desc
limit 100;
```

The screenshot shows the SQL Workbench/J interface. The title bar reads "SQL Workbench/J Postgre DB - Default.wksp". The toolbar includes various icons for navigation and execution. The "Statement 1" tab is active, displaying the following SQL query:

```
4
5
6 select * from dms_sample.ticket_purchase_hist order by transaction_date_time desc limit 100;
```

The "Result 1" tab is active, showing a table with 5 columns: `sporting_event_ticket_id`, `purchased_by_id`, `transaction_date_time`, `transferred_from_id`, and `purchase_price`. The table contains 20 rows of data, all with a `purchase_price` of 40.29.

sporting_event_ticket_id	purchased_by_id	transaction_date_time	transferred_from_id	purchase_price
15304431	2839100	2019-05-29 22:03:28		40.29
15304321	2839100	2019-05-29 22:03:28		40.29
15304541	2839100	2019-05-29 22:03:28		40.29
15304451	2839100	2019-05-29 22:03:28		40.29
15304361	2839100	2019-05-29 22:03:28		40.29
15305671	2839100	2019-05-29 22:03:28		40.29
15304621	2839100	2019-05-29 22:03:28		40.29
15304531	2839100	2019-05-29 22:03:28		40.29
15316071	2839100	2019-05-29 22:03:28		40.29
15316001	2839100	2019-05-29 22:03:28		40.29
15304391	2839100	2019-05-29 22:03:28		40.29
15305701	2839100	2019-05-29 22:03:28		40.29
15305691	2839100	2019-05-29 22:03:28		40.29
15305661	2839100	2019-05-29 22:03:28		40.29
15304641	2839100	2019-05-29 22:03:28		40.29
15304611	2839100	2019-05-29 22:03:28		40.29
15305921	2839100	2019-05-29 22:03:28		40.29
15305881	2839100	2019-05-29 22:03:28		40.29
15305861	2839100	2019-05-29 22:03:28		40.29
15305841	2839100	2019-05-29 22:03:28		40.29
15305811	2839100	2019-05-29 22:03:28		40.29

Once you've sold some tickets you can run the `generateTransferActivity` procedure. The following will transfer tickets from the owner to another person. The whole "batch" of tickets purchased is transferred 80% of the time and 20% of the time an individual ticket is transferred.

```
select dms_sample.generatetransferactivity(100);
```

### Note:

When enabling CDC functionality in DMS, only one DMS instance/task should activate "Ongoing replication" to avoid conflicts.

When replicating to multiple targets, the processing to fan out the updates should begin with the Amazon S3 bucket, that is the target of the DMS task responsible for Ongoing replication. The process should not begin with the source database, as only one CDC process should be tracking and setting the last committed transaction that was replicated.

## Appendix: AWS CloudFormation Template

The AWS CloudFormation template is below. This template only works in the us-east-1 region.

Copy and paste this template into an instructor\_dmslab.json file on your computer and save it. Select that file in AWS CloudFormation for Step 3.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "KeyName": {
      "Description": "Name of an existing EC2 KeyPair to enable SSH access to the instance",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription": "must be the name of an existing EC2 KeyPair in us-east-1 only."
    }
  },
  "Resources": {
    "dmsinstructorvpc": {
      "Type": "AWS::EC2::VPC",
      "Properties": {
        "CidrBlock": "10.0.0.0/24",
        "InstanceTenancy": "default",
        "EnableDnsSupport": "true",
        "EnableDnsHostnames": "true",
        "Tags": [
          {
            "Key": "Name",
            "Value": "DMSLabSourceDB"
          }
        ]
      }
    },
    "RDSSubNet": {
      "Type": "AWS::EC2::Subnet",
      "Properties": {
        "CidrBlock": "10.0.0.0/28",
        "AvailabilityZone": "us-east-1d",
        "VpcId": {
          "Ref": "dmsinstructorvpc"
        }
      }
    }
  }
}
```

```

    "Tags": [
      {
        "Key": "Name",
        "Value": "DMSLabRDS1"
      }
    ]
  },
  "EC2SubNet": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
      "CidrBlock": "10.0.0.32/28",
      "AvailabilityZone": "us-east-1c",
      "VpcId": {
        "Ref": "dmsinstructorvpc"
      }
    },
    "Tags": [
      {
        "Key": "Name",
        "Value": "DMSLabEC2"
      }
    ]
  },
  "RDSSubNet2": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
      "CidrBlock": "10.0.0.16/28",
      "AvailabilityZone": "us-east-1b",
      "VpcId": {
        "Ref": "dmsinstructorvpc"
      }
    },
    "Tags": [
      {
        "Key": "Name",
        "Value": "DMSLabRDS2"
      }
    ]
  },
  "igw0887475a258f00277": {
    "Type": "AWS::EC2::InternetGateway",
    "Properties": {
      "Tags": [
        {

```



```

        "Key": "Name",
        "Value": "DMSLabIGW"
    }
  ]
}
},
"dopt1cc25278": {
  "Type": "AWS::EC2::DHCPOptions",
  "Properties": {
    "DomainName": "ec2.internal",
    "DomainNameServers": [
      "AmazonProvidedDNS"
    ]
  }
},
"rtb0c3fae104a7b64456": {
  "Type": "AWS::EC2::RouteTable",
  "Properties": {
    "VpcId": {
      "Ref": "dmsinstructorvpc"
    },
    "Tags": [
      {
        "Key": "Name",
        "Value": "DMSLabRT"
      }
    ]
  }
},
"instancei0f63b887480639040": {
  "Type": "AWS::EC2::Instance",
  "Properties": {
    "DisableApiTermination": "false",
    "InstanceInitiatedShutdownBehavior": "stop",
    "EbsOptimized": "true",
    "ImageId": "ami-04681a1dbd79675a5",
    "InstanceType": "t3.2xlarge",
    "KeyName": { "Ref" : "KeyName" },
    "UserData" : { "Fn::Base64" : { "Fn::Join" : [ "", [
      "#!/bin/bash -xe\n",
      "yum install -y postgresql\n",
      "yum install -y git\n",
      "yum update -y\n",
      "cd /home/ec2-user\n",

```

```

"git clone https://github.com/aws-samples/aws-database-migration-
samples.git\n"
    ]]],
    "Monitoring": "false",
    "Tags": [
      {
        "Key": "Name",
        "Value": "DMSLabEC2"
      }
    ],
    "NetworkInterfaces": [
      {
        "DeleteOnTermination": "true",
        "Description": "Primary network interface",
        "DeviceIndex": 0,
        "SubnetId": {
          "Ref": "EC2SubNet"
        },
        "PrivateIpAddresses": [
          {
            "PrivateIpAddress": "10.0.0.40",
            "Primary": "true"
          }
        ],
        "GroupSet": [
          {
            "Ref": "sgDMSLabSG"
          }
        ],
        "AssociatePublicIpAddress": "true"
      }
    ]
  },
  "rdsdmslabdb": {
    "Type": "AWS::RDS::DBInstance",
    "Properties": {
      "AllocatedStorage": "20",
      "AllowMajorVersionUpgrade": "false",
      "AutoMinorVersionUpgrade": "true",
      "DBInstanceClass": "db.t2.xlarge",
      "DBInstanceIdentifier": "dmslabinstance",
      "Port": "5432",
      "PubliclyAccessible": "true",
      "StorageType": "gp2",

```

```

"BackupRetentionPeriod": "7",
"MasterUsername": "master",
"MasterUserPassword": "master123",
"PreferredBackupWindow": "04:00-04:30",
"PreferredMaintenanceWindow": "sun:05:20-sun:05:50",
"DBName": "sportstickets",
"Engine": "postgres",
"EngineVersion": "10.4",
"LicenseModel": "postgresql-license",
"DBSubnetGroupName": {
  "Ref": "dbsubnetdefaultdmsinstructorvpc"
},
"VPCSecurityGroups": [
  {
    "Ref": "sgrdslaunchwizard2"
  }
],
"Tags": [
  {
    "Key": "workload-type",
    "Value": "other"
  }
]
}
},
"dbsubnetdefaultdmsinstructorvpc": {
  "Type": "AWS::RDS::DBSubnetGroup",
  "Properties": {
    "DBSubnetGroupDescription": "Created from the RDS Management Console",
    "SubnetIds": [
      {
        "Ref": "RDSSubNet"
      },
      {
        "Ref": "EC2SubNet"
      },
      {
        "Ref": "RDSSubNet2"
      }
    ]
  }
},
"sgDMSLabSG": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {

```

```

    "GroupDescription": "launch-wizard-6 created 2018-08-29T15:10:01.302-
04:00",
    "VpcId": {
      "Ref": "dmsinstructorvpc"
    }
  },
  "sgrdslaunchwizard2": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Created from the RDS Management Console:
2018/08/29 18:14:15",
      "VpcId": {
        "Ref": "dmsinstructorvpc"
      },
      "Tags": [
        {
          "Key": "Name",
          "Value": "DMSLabRDS-SG"
        }
      ]
    }
  },
  "dbsgdefault": {
    "Type": "AWS::RDS::DBSecurityGroup",
    "Properties": {
      "GroupDescription": "default"
    }
  },
  "gw1": {
    "Type": "AWS::EC2::VPCEGatewayAttachment",
    "Properties": {
      "VpcId": {
        "Ref": "dmsinstructorvpc"
      },
      "InternetGatewayId": {
        "Ref": "igw0887475a258f00277"
      }
    }
  },
  "subnetroute1": {
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Properties": {
      "RouteTableId": {
        "Ref": "rtb0c3fae104a7b64456"
      }
    }
  }
}

```

```

    },
    "SubnetId": {
      "Ref": "RDSSubNet2"
    }
  },
  "subnetroute2": {
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Properties": {
      "RouteTableId": {
        "Ref": "rtb0c3fae104a7b64456"
      },
      "SubnetId": {
        "Ref": "RDSSubNet"
      }
    }
  },
  "subnetroute3": {
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Properties": {
      "RouteTableId": {
        "Ref": "rtb0c3fae104a7b64456"
      },
      "SubnetId": {
        "Ref": "EC2SubNet"
      }
    }
  },
  "route1": {
    "Type": "AWS::EC2::Route",
    "Properties": {
      "DestinationCidrBlock": "0.0.0.0/0",
      "RouteTableId": {
        "Ref": "rtb0c3fae104a7b64456"
      },
      "GatewayId": {
        "Ref": "igw0887475a258f00277"
      }
    }
  },
  "DependsOn": "gw1"
},
"dchpassoc1": {
  "Type": "AWS::EC2::VPCDHCPOptionsAssociation",
  "Properties": {
    "VpcId": {

```

```

        "Ref": "dmsinstructorvpc"
    },
    "DhcpOptionsId": {
        "Ref": "dopt1cc25278"
    }
},
"ingress1": {
    "Type": "AWS::EC2::SecurityGroupIngress",
    "Properties": {
        "GroupId": {
            "Ref": "sgDMSLabSG"
        },
        "IpProtocol": "tcp",
        "FromPort": "22",
        "ToPort": "22",
        "CidrIp": "0.0.0.0/0"
    }
},
"ingress2": {
    "Type": "AWS::EC2::SecurityGroupIngress",
    "Properties": {
        "GroupId": {
            "Ref": "sgrdslaunchwizard2"
        },
        "IpProtocol": "tcp",
        "FromPort": "5432",
        "ToPort": "5432",
        "SourceSecurityGroupId": {
            "Ref": "sgDMSLabSG"
        },
        "SourceSecurityGroupOwnerId": "649225637812"
    }
},
"ingress3": {
    "Type": "AWS::EC2::SecurityGroupIngress",
    "Properties": {
        "GroupId": {
            "Ref": "sgrdslaunchwizard2"
        },
        "IpProtocol": "tcp",
        "FromPort": "5432",
        "ToPort": "5432",
        "CidrIp": "72.21.196.67/32"
    }
}

```

```

},
"ingress4": {
  "Type": "AWS::EC2::SecurityGroupIngress",
  "Properties": {
    "GroupId": {
      "Ref": "sgrdslaunchwizard2"
    },
    "IpProtocol": "tcp",
    "FromPort": "5432",
    "ToPort": "5432",
    "CidrIp": "0.0.0.0/0"
  }
},
"egress1": {
  "Type": "AWS::EC2::SecurityGroupEgress",
  "Properties": {
    "GroupId": {
      "Ref": "sgDMSLabSG"
    },
    "IpProtocol": "-1",
    "CidrIp": "0.0.0.0/0"
  }
},
"egress2": {
  "Type": "AWS::EC2::SecurityGroupEgress",
  "Properties": {
    "GroupId": {
      "Ref": "sgrdslaunchwizard2"
    },
    "IpProtocol": "-1",
    "CidrIp": "0.0.0.0/0"
  }
}
},
"Description": "DMS Lab Instructor account",
"Metadata": {
  "AWS::CloudFormation::Designer": {
    "a79fb943-c167-4e59-8eda-911d4acc331f": {
      "size": {
        "width": 60,
        "height": 60
      },
      "position": {
        "x": 810,
        "y": 390
      }
    }
  }
}

```

```
    },  
    "z": 1,  
    "embeds": []  
  }  
}  
}
```