# GameDay Test cases:

This document cover steps to be followed for simulating gameday scenarios of RDS PostgreSQL database.

**High level steps :**

1. Setup database, configure notification method and configure cloud watch alarms
2. Configure Client
3. Perform Data load
4. Run Tests
5. Capture result

**Detailed Steps :**

***Setup database, configure notification method and configure cloud watch alarms:*** Use below Cloud formation Template to setup database and related metric and alarms. This will create below resourced in AWS account

* 1. Database Parameter group (version 10 compatible)
  2. RDS PostgreSQL Database (10.6 version)
  3. SNS Notification Topic
  4. CloudWatch Alarms and Event Subscriptions:
     1. High CPU Utilization
     2. Low Free Memory
     3. Low Storage Space
     4. Event Subscription for database availability

AWSTemplateFormatVersion: '2010-09-09'  
Description: 'AWS CloudFormation Template to deploy RDS database'  
Parameters:  
  DBName:  
    Default: gamedaypgdb  
    Description: The database name  
    Type: String  
    MinLength: '1'  
    MaxLength: '64'  
    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]\*'  
    ConstraintDescription: must begin with a letter and contain only alphanumeric characters.  
  DBUser:  
    Description: The database admin account username  
    Type: String  
    MinLength: '1'  
    MaxLength: '16'  
  DBPassword:  
    NoEcho: 'true'  
    Description: The database admin account password  
    Type: String  
    MinLength: '8'  
    MaxLength: '41'  
  InstanceClass:  
    Description: The database istance class  
    Type: String  
    AllowedValues:   
      - db.t2.micro  
      - db.t2.small  
      - db.t2.large  
      - db.t3.large  
    MinLength: '8'  
    MaxLength: '41'  
  Email:  
    Type: String  
    Description: Email address to notify an alarm.  
Resources:  
  MyRdsDB:  
    Type: AWS::RDS::DBInstance  
    Properties:  
      DBInstanceIdentifier: !Ref 'DBName'  
      DBName: !Ref 'DBName'  
      DBInstanceClass: !Ref 'InstanceClass'  
      Engine: postgres  
      EngineVersion: 10.6  
      AllocatedStorage: '200'  
      StorageType: io1  
      Iops: 2000  
      MasterUsername: !Ref 'DBUser'  
      MasterUserPassword: !Ref 'DBPassword'  
      DBParameterGroupName: !Ref 'MyRDSParamGroup'  
      EnablePerformanceInsights: true  
  MyRDSParamGroup:  
    Type: AWS::RDS::DBParameterGroup  
    Properties:  
      Family: postgres10  
      Description: GameDay Database Parameter Group  
      Parameters:  
        shared\_preload\_libraries: 'auto\_explain, pgaudit, pglogical, pg\_stat\_statements, pg\_hint\_plan'  
        random\_page\_cost: '1'  
        seq\_page\_cost: '2'  
        max\_connections: 1000  
  AlarmNotificationTopic:  
    Type: 'AWS::SNS::Topic'  
    Properties:  
      DisplayName: "GameDayTopic"  
      Subscription:  
        - Endpoint: !Ref Email  
          Protocol: email  
  CPUAlarm:  
    Type: AWS::CloudWatch::Alarm  
    Properties:  
      AlarmName: HighCPUUtilization  
      AlarmDescription: 'High CPU Utilization'  
      AlarmActions:   
      - Ref: AlarmNotificationTopic  
      MetricName: CPUUtilization  
      Namespace: AWS/RDS  
      Statistic: Average  
      Period: '60'  
      EvaluationPeriods: '1'  
      Threshold: '80'  
      ComparisonOperator: GreaterThanThreshold  
      Dimensions:  
      - Name: DBInstanceIdentifier  #InstanceId  
        Value:  
          Ref: MyRdsDB  
  FreeableMemoryAlarm:  
    Type: AWS::CloudWatch::Alarm  
    Properties:  
      AlarmName: LowFreeMemory  
      AlarmDescription: FreeableMemoryAlarm  
      AlarmActions:  
      - Ref: AlarmNotificationTopic  
      MetricName: FreeableMemory  
      Namespace: AWS/RDS  
      Statistic: Average  
      Period: '300'  
      EvaluationPeriods: '1'  
      Threshold: '100000'  
      ComparisonOperator: LessThanOrEqualToThreshold  
      Dimensions:  
      - Name: DBInstanceIdentifier  
        Value:  
          Ref: MyRdsDB  
  FreeStorageSpaceAlarm:  
    Type: AWS::CloudWatch::Alarm  
    Properties:  
      AlarmName: LowFreeStoraeSpace  
      AlarmDescription: FreeStorageSpace alarm  
      AlarmActions:  
      - Ref: AlarmNotificationTopic  
      MetricName: FreeStorageSpace  
      Namespace: AWS/RDS  
      Statistic: Average  
      Period: '300'  
      EvaluationPeriods: '1'  
      Threshold: '10000'  
      ComparisonOperator: LessThanOrEqualToThreshold  
      Dimensions:  
      - Name: DBInstanceIdentifier  
        Value:  
          Ref: MyRdsDB  
  InstanceEventSubscription:  
    Type: AWS::RDS::EventSubscription  
    Properties:  
      EventCategories:  
      - failover  
      - availability  
      - failure  
      - deletion  
      SnsTopicArn:  
        Ref: AlarmNotificationTopic  
      SourceIds:  
      - Ref: MyRdsDB  
      SourceType: db-instance  
      Enabled: true  
Outputs:  
  ConnectionString:  
    Description: Database connection string  
    Value: !Join ['', ['psql -h ', !GetAtt [MyRdsDB, Endpoint.Address], ' -p ', !GetAtt [MyRdsDB, Endpoint.Port], ' -d ', !Ref 'DBName', ' -U ', !Ref 'DBUser']]

***Configure Client :*** This is optional step. If you have psql and pgbench installed and database connectivity exists you may ignore this step. Otherwise run below CloudFormation Template to create a EC2 instance in same region with psql and pgbench configured.

Prerequisites : Create a EC2 key pair

  AWSTemplateFormatVersion: "2010-09-09"  
  Description: "EC2 Client for Gameday"  
  Parameters:  
    EC2KeyPair:  
      Description: EC2 Key Pair  
      Type: "AWS::EC2::KeyPair::KeyName"  
    Ec2InstanceClass:  
      Description: EC2 istance class  
      Type: String  
      AllowedValues:   
        - t2.micro  
        - t2.small  
        - t2.large  
        - t3.large  
    Ec2az:    
      Description: EC2 Availability Zone  
      Type: "AWS::EC2::AvailabilityZone::Name"        
  Resources:   
    "GamdgayEC2" :  
      Type: AWS::EC2::Instance  
      Properties:         
        AvailabilityZone: !Ref Ec2az  
        EbsOptimized: false  
        ImageId: ami-011b3ccf1bd6db744  
        InstanceType: !Ref Ec2InstanceClass  
        KeyName: !Ref EC2KeyPair  
        Monitoring: true  
        UserData:  
          'Fn::Base64':  
            !Sub |  
             #!/bin/bash  
             yum -y update  
             yum -y install wget  
             yum -y --enablerepo=extras install epel-release  
             yum -y install python-pip  
             pip install pystache argparse python-daemon requests  
             yum -y install http://download-ib01.fedoraproject.org/pub/epel/7/x86\_64/Packages/l/llvm5.0-libs-5.0.1-7.el7.x86\_64.rpm  
             yum -y install http://download-ib01.fedoraproject.org/pub/epel/7/x86\_64/Packages/l/llvm5.0-5.0.1-7.el7.x86\_64.rpm  
             yum -y install https://download.postgresql.org/pub/repos/yum/11/redhat/rhel-7-x86\_64/pgdg-redhat11-11-2.noarch.rpm  
             yum -y install postgresql11\* -y  
             cd /opt  
             curl -O https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.tar.gz  
             curl -O https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.amzn1.noarch.rpm  
             tar -xvpf aws-cfn-bootstrap-latest.tar.gz  
             cd aws-cfn-bootstrap-1.4/  
             python setup.py build  
             python setup.py install  
             ln -s /usr/init/redhat/cfn-hup /etc/init.d/cfn-hup  
             chmod 775 /usr/init/redhat/cfn-hup  
             mkdir /opt/aws  
             mkdir /opt/aws/bin  
             ln -s /usr/bin/cfn-hup /opt/aws/bin/cfn-hup  
             set -o errexit  
             /usr/bin/cfn-init -v --stack ${AWS::StackName} --resource EC2Instance --region ${AWS::Region}  
             /usr/bin/cfn-signal --exit-code $? --stack ${AWS::StackName} --resource GamdgayEC2 --region ${AWS::Region}

***Perform Data Load:*** Run below command from your client system. This may take few hours to complete

/usr/pgsql-11/bin/pgbench \  
-h <<Database Endpoint>> -p <<Database Port>> -d <<DB Name>> -U <<DB User>> \  
-i -s 5000 --quiet --no-vacuum --fillfactor=70 --foreign-keys

***Run Test cases :*** Use below command to generate load in the database any

/usr/pgsql-11/bin/pgbench \  
-h <<Database Endpoint>> -p <<Database Port>> -d <<DB Name>> -U <<DB User>> \  
-b tpcb-like --client=1000 -j 10 --connect --protocol=simple --time=600

**Some example of alarm generated :**

