Artifact Description

Title:

**Benchmarking RDS Performance for MySQL and PostgreSQL Databases on AWS**

Authors:

Lam Ching Lung  
Ho Wai Sum  
Shih Wing Yan

Abstract

This artifact details the methods used to benchmark the performance of MySQL, Aurora MySQL, PostgreSQL, and Aurora PostgreSQL databases deployed on AWS. Using **sysbench** and **HammerDB**, we test these databases configured on **t3.medium** instances within the **us-east-1** region and **us-east-1a** availability zone. This document serves to aid reproducibility of our benchmark results by describing the setup and execution steps.

Description

**Type of artifact:** Experimental data, software

**Keywords:** AWS, RDS, MySQL, PostgreSQL, Aurora, sysbench, HammerDB, benchmark

**Metadata:**

**Hardware:**

* AWS EC2 instance (**t3.medium**)
* MySQL RDS instances (**t3.medium**)
* PostgreSQL RDS instances (**t3.medium**)
* AuroraMySQL RDS instances (**t3.medium**)
* AuroraPostgreSQL RDS instances (**t3.medium**)

**Software:**

* OS: Amazon Linux 2023,
* Tools:
  + sysbench 1.0.20
  + HammerDB 4.10
  + Docker

**Publicly available?:** Yes

Setup & Installation

**EC2 Configuration**

sudo yum -y install git

sudo yum -y install make automake libtool pkgconfig libaio-devel

sudo yum -y install mariadb105-devel postgresql-devel openssl-devel

cd /home/ec2-user

git clone https://github.com/akopytov/sysbench.git --branch 1.0.20

cd sysbench

./autogen.sh

./configure --with-pgsql

make -j

sudo make install

sysbench --version #(Verify the sysbench successfully installed)

**Database Setup**

Deploy RDS instances: MySQL, AuroraMySQL, PostgreSQL, AuroraPostgreSQL (all **t3.medium**).

Benchmarking Procedure

**Install docker**

sudo yum -y install docker

sudo systemctl start docker

sudo systemctl enable docker

sudo usermod -aG docker $USER

docker -v

**Sysbench Tests**

# The command I have used on setting up EC2 (to install sysbench):

sudo yum -y install git

sudo yum -y install make automake libtool pkgconfig libaio-devel

sudo yum -y install mariadb105-devel postgresql-devel openssl-devel

cd /home/ec2-user

git clone https://github.com/akopytov/sysbench.git --branch 1.0.20

cd sysbench

./autogen.sh

./configure --with-pgsql

make -j

sudo make install

sysbench --version #(Verify the sysbench successfully installed)

# To access your database using docker

# docker run -it --rm mysql mysql -h <mysql-host> -u <mysql-user> -p -P <mysql-port>

docker run -it --rm mysql mysql -h database-mysql.c7ec88qoassw.us-east-1.rds.amazonaws.com -u admin -p -P 3306

docker run -it --rm mysql mysql -h database-auroramysql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com -u admin -p -P 3306

docker run -it --rm postgres psql -h database-postgresql.c7ec88qoassw.us-east-1.rds.amazonaws.com -U postgres -d postgres

docker run -it --rm postgres psql -h database-aurorapostgresql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com -U postgres -d postgres

# enter password

# After accessing to your database, create table for sysbench and HammerDB benchmark test

CREATE DATABASE sbtest;

CREATE DATABASE hmtest;

# Running sysbench benchmark on MySQL

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--mysql-host=database-mysql.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--mysql-port=3306 \

--mysql-user=admin \

--mysql-password=Passw0rd123 \

--mysql-db=sbtest \

--db-driver=mysql \

--tables=10 \

--table\_size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

prepare

# Type the followings to show the benchmark result

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--mysql-host=database-mysql.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--mysql-port=3306 \

--mysql-user=admin \

--mysql-password=Passw0rd123 \

--mysql-db=sbtest \

--db-driver=mysql \

--tables=10 \

--table\_size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

run

# Running sysbench benchmark on Aurora MySQL

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--mysql-host=database-auroramysql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--mysql-port=3306 \

--mysql-user=admin \

--mysql-password=Passw0rd123 \

--mysql-db=sbtest \

--db-driver=mysql \

--tables=10 \

--table\_size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

prepare

# Type the followings to show the benchmark result

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--mysql-host=database-auroramysql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--mysql-port=3306 \

--mysql-user=admin \

--mysql-password=Passw0rd123 \

--mysql-db=sbtest \

--db-driver=mysql \

--tables=10 \

--table\_size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

run

# Running sysbench benchmark on PostgreSQL:

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--pgsql-host=database-postgresql.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--pgsql-port=5432 \

--pgsql-user=postgres \

--pgsql-password=Passw0rd123 \

--pgsql-db=sbtest \

--db-driver=pgsql \

--tables=10 \

--table-size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

prepare

# Type the followings to show the benchmark result

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--pgsql-host=database-postgresql.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--pgsql-port=5432 \

--pgsql-user=postgres \

--pgsql-password=Passw0rd123 \

--pgsql-db=sbtest \

--db-driver=pgsql \

--tables=10 \

--table-size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

run

# Running sysbench benchmark on Aurora PostgreSQL:

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--pgsql-host=database-aurorapostgresql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--pgsql-port=5432 \

--pgsql-user=postgres \

--pgsql-password=Passw0rd123 \

--pgsql-db=sbtest \

--db-driver=pgsql \

--tables=10 \

--table-size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

prepare

# Type the followings to show the benchmark result

sysbench /usr/local/share/sysbench/oltp\_read\_write.lua \

--pgsql-host=database-aurorapostgresql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com \

--pgsql-port=5432 \

--pgsql-user=postgres \

--pgsql-password=Passw0rd123 \

--pgsql-db=sbtest \

--db-driver=pgsql \

--tables=10 \

--table-size=10000 \

--threads=8 \

--time=120 \

--report-interval=10 \

run

**HammerDB Tests**

# This are to obtain the libmysqlclient.so.21 environment variable so that the HammerDB can run mysql

sudo wget https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm

sudo dnf install mysql80-community-release-el9-1.noarch.rpm -y

sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2023

sudo yum install mysql-server

sudo yum install mysql-devel

# Install HammerDB:

wget https://github.com/TPC-Council/HammerDB/releases/download/v4.10/HammerDB-4.10-Linux.tar.gz

tar -xzvf HammerDB-4.10-Linux.tar.gz

cd HammerDB-4.4

#launch HammerDB cli

./hammerdbcli

# Launch HammerDB WebService (GUI for viewing benchmark result)

cd /home/ec2-user/HammerDB-4.10 # the path where the HammerDB is installed

./hammerdbcliws # launch hammerdb webservice

# To access the hammerdb webservice, type "[ec2-instance:ip]:8080" in the browser

# You will then able to view the job by ID,

# click "result", you are able to see the chart showing the NOPM and TPM

# click "result data" for actual NOPM and TPM number

# Steps for running HammerDB TPROC-C benchmark test on MySQL:

dbset db mysql

diset connection mysql\_host database-mysql.c7ec88qoassw.us-east-1.rds.amazonaws.com

diset connection mysql\_port 3306

diset tpcc mysql\_user admin

diset tpcc mysql\_pass Passw0rd123

diset tpcc mysql\_dbase hmtest

diset tpcc mysql\_count\_ware 10

diset tpcc mysql\_num\_vu 8

diset tpcc mysql\_rampup 2

diset tpcc mysql\_duration 2

diset tpcc mysql\_total\_iterations 1000000

buildschema

vustatus

vurun

vudestroy

# Steps for running HammerDB TPROC-C benchmark test on AuroraMySQL:

dbset db mysql

diset connection mysql\_host database-auroramysql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com

diset connection mysql\_port 3306

diset tpcc mysql\_user admin

diset tpcc mysql\_pass Passw0rd123

diset tpcc mysql\_dbase hmtest

diset tpcc mysql\_count\_ware 10

diset tpcc mysql\_num\_vu 8

diset tpcc mysql\_rampup 2

diset tpcc mysql\_duration 2

diset tpcc mysql\_total\_iterations 1000000

buildschema

vustatus

vurun

vudestroy

# Steps for running HammerDB TPROC-C benchmark test on PostgreSQL:

dbset db pg

diset connection pg\_host database-postgresql.c7ec88qoassw.us-east-1.rds.amazonaws.com

diset connection pg\_port 5432

diset tpcc pg\_user postgres

diset tpcc pg\_superuserpass Passw0rd123

diset tpcc pg\_pass Passw0rd123

diset tpcc pg\_dbase hmtest

diset tpcc pg\_count\_ware 10

diset tpcc pg\_num\_vu 8

diset tpcc pg\_rampup 2

diset tpcc pg\_duration 2

diset tpcc pg\_total\_iterations 1000000

buildschema

vustatus

vurun

vudestroy

# Steps for running HammerDB TPROC-C benchmark test on AuroraPostgreSQL:

dbset db pg

diset connection pg\_host database-aurorapostgresql-instance-1.c7ec88qoassw.us-east-1.rds.amazonaws.com

diset connection pg\_port 5432

diset tpcc pg\_user postgres

diset tpcc pg\_superuserpass Passw0rd123

diset tpcc pg\_pass Passw0rd123

diset tpcc pg\_dbase hmtest

diset tpcc pg\_count\_ware 10

diset tpcc pg\_num\_vu 8

diset tpcc pg\_rampup 2

diset tpcc pg\_duration 2

diset tpcc pg\_total\_iterations 1000000

buildschema

vustatus

vurun

vudestroy

**Performance Metrics:** Transactions per Minute (TPM), New Orders per Minute (NOPM).

**Graphs and Charts:** Benchmark results visualized in HammerDB WebService.

How to Run the Benchmark

Detailed step-by-step commands are provided in the Setup & Installation and Benchmarking Procedure sections.

Evaluation and Expected Outcome

Comparison of performance across different database configurations.

Analysis of scalability and efficiency in database handling under load.

**Steps to set up databases in AWS RDS Management Console:**

MySQL:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

PostgresSQL:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated