AWS Whitepaper

Determining IOPS Needs for Oracle Database on AWS

Determining IOPS Needs for Oracle Database on AWS: AWS Whitepaper

Table of Contents

This whitepaper is for historical reference only. Some content might be outdated and some links might not be available.

Determining IOPS Needs for Oracle Database on AWS

Publication date: **November 17, 2021** (*Document history*)

Amazon Web Services (AWS) provides a comprehensive set of services and tools for deploying Oracle Database on the AWS Cloud infrastructure, one of the most reliable and secure cloud computing services available today. Many businesses of all sizes use Oracle Database to handle their data needs. Oracle Database performance relies heavily on the performance of the storage subsystem, but storage performance always comes at a price. This whitepaper includes information to help you determine the input/output operations per second (IOPS) necessary for your database storage system to have the best performance at optimal cost.

Introduction

AWS offers customers the flexibility to run Oracle Database on either Amazon Relational Database Service (Amazon RDS), which is a managed database service in the cloud, or on Amazon Elastic Compute Cloud (Amazon EC2). Many customers prefer to use Amazon RDS for Oracle Database because it provides an easy, managed option to run Oracle Database on AWS without having to think about infrastructure provisioning, or installing and maintaining database software. You can also run Oracle Database directly on Amazon EC2, which allows you full control over setup of the entire infrastructure and database environment.

To get the best performance from your database, you must configure the storage tier to provide the IOPS and throughput that the database needs. This is a requirement for both Oracle Database on Amazon RDS and Oracle Database on Amazon EC2. If the storage system does not provide enough IOPS to support the database workload, you will have sluggish database performance and transaction backlog. However, if you provision much higher IOPS than your database actually needs, you will have unused capacity.

The elastic nature of the AWS infrastructure allows you to increase or decrease the total IOPS available for Oracle Database on Amazon EC2, but doing this could have a performance impact on the database, requires extra effort, and might require database downtime.

Storage options for Oracle Database

For Oracle Database storage on AWS, you must use Amazon Elastic Block Store (Amazon EBS) volumes, which offer the consistent, low-latency performance required to run your Oracle Database. Each Amazon EBS volume is automatically replicated within its Availability Zone to protect you from component failure, which provides high availability and durability. Amazon EBS provides these volume types:

- General Purpose solid state drive (SSD) (gp2)
- General Purpose SSD (gp3)
- Provisioned IOPS SSD (io1)
- Provisioned IOPS SSD (io2)
- Throughput Optimized hard disk drive (HDD) (st1)
- Cold HHD (sc1)

Volume types differ in performance characteristics and cost. For the high and consistent IOPS required for Oracle Database, Amazon EBS General Purpose SSD or Amazon EBS Provisioned IOPS SSD volumes are the best fit.

For gp2 volumes, IOPS performance is directly related to the provisioned capacity. gp2 volumes can deliver a consistent baseline of 3 IOPS/GB up to a maximum of 16,000 IOPS (based on 16 KB/IO) for a 16 TB volume. Input/output (I/O) is included in the price of gp2 volumes, so you pay only for each gigabyte of storage that you provision. gp2 volumes also have the ability to burst to 3,000 IOPS per volume, independent of volume size, to meet the periodic spike in performance that most applications need. This is a useful database feature, for which you can predict normal IOPS needs well, but you might still experience an occasional higher spike based on specific workloads.

Currently, gp3 is available for Oracle databases running on Amazon EC2. It has the same qualities of gp2 but also increases the throughput from 250 MiB/s to 1,000 MiB/s.

gp2 volumes are sufficient for most Oracle Database workloads. If you need more IOPS and throughput than gp2 can provide, Provisioned IOPS (PIOPS) is the best choice. io2 volumes can provide up to 64,000 IOPS per volume for AWS Nitro-based instances and 32,000 IOPS per volume for other instance families.

Throughput optimized HDD volumes (st1) offer low-cost HDD volume designed for intensive workloads that require less IOPS but high throughput. Oracle databases used for data warehouses

and data analytics purposes can use st1 volumes. Any log processing or data staging areas that require high throughput, such as Oracle external tables or external BLOB storage, can use `st1` volumes. `st1` volumes can handle a maximum of 500 IOPS per volume.

Cold HDD volumes (`sc1`) are suitable for legacy systems that you retain for occasional reference or archive purposes. These systems are accessed less frequently and only a few scans are performed each day on the volume.

You can create striped volumes (areas of free space on multiple volumes) for more IOPS and larger capacity. The maximum IOPS an EC2 instance can support across all EBS volumes is 260,000. The maximum IOPS an RDS instance can support is 256,000. Use only Amazon EBS-optimized instances with gp2 and PIOPS. You can use multiple EBS volumes individually for different data files, but striped volumes allow better throughput, balancing, scalability, and burstable performance (for gp2).

IOPS basics

IOPS is the standard measure of I/O operations per second on a storage device. It includes both read and write operations. The amount of I/O used by Oracle Database can vary greatly in a time period, based on the server load and the specific queries running. If you are migrating an existing Oracle Database to AWS, to ensure that you get the best performance regardless of load, you must determine the peak IOPS used by your database and provision Amazon EBS volumes on AWS accordingly. If you choose an IOPS number based on the average IOPS used by your existing database, you should have sufficient IOPS for the database in most cases, but database performance will suffer at peak load. You can mitigate this issue to some extent by using Amazon EBS gp2 volumes, which have the ability to burst to higher IOPS for small periods of time.

Customers sometimes assume that they need much more IOPS than they actually do. This assumption occurs if customers confuse storage system IOPS with database IOPS. Most enterprises use storage area network (SAN) systems that can provide 100,000–200,000 or more IOPS for storage. The same SAN storage is usually shared by multiple databases and file systems, which means the total IOPS provided by the storage system is used by many more applications than a single database.

Most Oracle Database production systems in domains such as enterprise resource planning (ERP) and customer relationship management (CRM) are in the range of 3,000–30,000 IOPS. Your individual application might have different IOPS requirements. A performance test environment's IOPS needs are generally identical to those of production environments, but for other test and development environments, the range is usually 200–2,000 IOPS.

Some online transaction processing (OLTP) systems use up to 60,000 IOPS. There are Oracle databases that use more than 60,000 IOPS, but that is unusual. If your environment shows numbers outside these parameters, you should complete further analysis to confirm your numbers.

Estimating IOPS for an existing database

The best way to estimate the actual IOPS that is necessary for your database is to query the system tables over a period of time and find the peak IOPS usage of your existing database. To do this, you measure IOPS over a period of time and select the highest value. You can get this information from the GV$SYSSTAT *dynamic performance view*, which is a special view in Oracle Database that provides database performance information. This view is continuously updated while the database is open and in use. Oracle Enterprise Manager and Automatic Workload Repository (AWR) reports also use these views to gather data. There is a GV$ view for almost all V$ views. GV$ views contain data for all nodes in a Real Application Cluster (RAC) identified by an instance ID. You can also use GV$ views for non-RAC systems, which have only one row for each performance criterion.

To determine IOPS, you can modify the following sample Oracle PL/SQL script for your needs and run the script during peak database load in your environment. For better accuracy, run this during the same peak period for a few days and then choose the highest value as the peak IOPS.

Because the sample script captures data and stores the PEAK_IOPS_MEASUREMENT table, you must first create the table with this code:

```
CREATE TABLE peak_iops_measurement (capture_timestamp date,
total_read_io number, total_write_io number, total_io number,
total_read_bytes number, total_write_bytes number, total_bytes
number);
```

The following script runs for an hour (run_duration := 3600) and captures data every five seconds (capture_gap := 5). It then calculates the average I/O and throughput per second for those 5 seconds, and stores this information in the table. To best fit your needs, you can modify the run_duration and capture_gap values to change the number of seconds that the script runs and the frequency in seconds that data is captured.

```
DECLARE
run_duration number := 60;
capture_gap number := 5;
loop_count number :=run_duration/capture_gap;
rdio number;
wtio number;
prev_rdio number :=0;
prev_wtio number :=0;
```

```
rdbt number;
wtbt number;
prev_rdbt number;
prev_wtbt number;

BEGIN
  FOR i in 1..loop_count LOOP
  SELECT SUM(value) INTO rdio from gv$sysstat
  WHERE name ='physical read total IO requests';
  SELECT SUM(value) INTO wtio from gv$sysstat
  WHERE name ='physical write total IO requests';
  SELECT SUM(value)* 0.000008 INTO rdbt from gv$sysstat
  WHERE name ='physical read total bytes';
SELECT SUM(value* 0.000008) INTO wtbt from gv$sysstat
WHERE name ='physical write total bytes';


IF i > 1  THEN
INSERT INTO peak_iops_measurement (capture_timestamp,total_read_io,
total_write_io, total_io, total_read_bytes,total_write_bytes, total_bytes)
VALUES (sysdate,(rdio-prev_rdio)/5,(wtio-prev_wtio)/5,((rdio-prev_rdio)/5)+
((wtio-prev_wtio))/5,(rdbt-prev_rdbt)/5,(wtbt-prev_wtbt)/5,((rdbt-prev_rdbt)/5)+
((wtbt-prev_wtbt))/5);

END IF;

prev_rdio := rdio;
prev_wtio := wtio;

prev_rdbt := rdbt;
prev_wtbt := wtbt;

DBMS_LOCK.SLEEP(capture_gap);

END LOOP;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
      ROLLBACK;
END;
/
```

The important values are `total_io` and `total_bytes`. The script captures the split of time spent in read and write operations that you can use for comparison later.

After you have collected data for a sufficient amount of time, you can find the peak IOPS used by your database by running the following query, which takes the highest value from the column `total_io`.

```
SELECT MAX(total_io) PeakIOPS FROM peak_iops_measurement;
```

To prepare for any unforeseen performance spikes, we recommend that you add an additional 10 percent to this peak IOPS number to account for the actual IOPS that your database needs. This actual IOPS is the total number of IOPS you should provision for your Amazon EBS volume (gp2 or io1).

Estimating IOPS for a new database

If you are setting up a database for the first time on AWS and you don't have any existing statistics, you can use an IOPS number based on the expected number of application transactions per second. Though the IOPS necessary per transaction can vary widely—based on the amount of data involved, the number of queries in a transaction, and the query complexity—generally, 30 IOPS per transaction is a good number to consider.

For example, if you are expecting 100 transactions per second, you can start with 3,000 IOPS Amazon EBS volumes. Because the amount of data in a new database is usually small, changing the IOPS associated with Amazon EBS will be relatively simple whether your database is on Amazon RDS or Amazon EC2.

Considering throughput

In addition to determining the right IOPS, it is also important to make sure your instance configuration can handle the throughput needs of your database. *Throughput* is the measure of the transfer of bits across the network between the EC2 instance running your database and the Amazon EBS volumes that store the data. The amount of available throughput relates directly to the network bandwidth that is available to the EC2 instance and the capability of Amazon EBS to receive data.

Amazon EBS–optimized instances consistently achieve the given level of performance. For more information, refer to Instance Types that Support EBS Optimization in the *Amazon EC2 User Guide for Linux Instances*. You can find more about Amazon EC2–Amazon EBS configuration in the Amazon EC2 User Guide. In addition to bandwidth availability, there are other considerations that affect which EC2 instance you should choose for your Oracle Database. These considerations include your database license, virtual CPUs available, and memory size.

Verifying your configuration

After you configure your environment based on the IOPS and throughput numbers necessary for your environment, you can verify your configuration before you install the database with the Oracle Orion tool, which is available from Oracle. Oracle Orion simulates Oracle Database I/O workloads using the same I/O software stack as Oracle Database, which provides a measurement of IOPS and throughput that simulates what your database will experience. For more details about this tool and to download it, refer to the Oracle website.

Conclusion

AWS provides the option to run Oracle in Amazon RDS or Amazon EC2. Choose RDS for a fully managed service, or EC2 if you prefer full control.

AWS offers various storage services that allow the workload to optimized for cost or performance. As workloads and requirements change, the solution can scale up or down elastically.

Contributors

The following individuals and organizations contributed to this document:

- Jayaraman Vellore Sampathkumar, Amazon Web Services
- Abdul Sathar Sait, Amazon Web Services
- Jinyoung Jung, Amazon Web Services
- Jason Massie, Amazon Web Services

Further reading

For additional information about using Oracle Database with AWS services, refer to the following resources.

Oracle Database on AWS

- *Advanced Architectures for Oracle Database on Amazon EC2* whitepaper
- *Strategies for Migrating Oracle Database to AWS* whitepaper
- *Choosing the Operating System for Oracle Workloads on Amazon EC2* whitepaper
- *Best Practices for Running Oracle Database on AWS* whitepaper

Oracle on AWS

- Oracle and Amazon Web Services
- Amazon RDS for Oracle Database
- Oracle in the Amazon Web Services Cloud - FAQ

Oracle Reference Architecture

- Oracle quick start on AWS

Oracle licensing on AWS

- Licensing Oracle Software in the Cloud Computing Environment

Getting started with Oracle RMAN backups and Amazon S3

- Getting Started: Backup Oracle databases directly to AWS with Oracle RMAN

AWS service details and pricing

- AWS Cloud Products
- AWS Architecture Center
- AWS Documentation

- AWS Whitepapers
- AWS Pricing
- AWS Pricing Calculator

Document history

To be notified about updates to this whitepaper, subscribe to the RSS feed.

Change	Description	Date
Whitepaper updated	Updates for technical accuracy.	November 17, 2021
Initial publication	Whitepaper first published.	December 1, 2018

Note

To subscribe to RSS updates, you must have an RSS plug-in enabled for the browser that you are using.

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.