AWS Storage Options Whitepaper with RDS, DynamoDB & Database on EC2 Cont.

Provides a brief summary for the Ideal Use cases, Anti-Patterns and other factors for Amazon RDS, DynamoDB & Databases on EC2 storage options

Amazon RDS

* RDS is a web service that provides the capabilities of MySQL, Oracle, MariaDB, Postgres or Microsoft SQL Server relational database as a managed, cloud-based service
* RDS eliminates much of the administrative overhead associated with launching, managing, and scaling your own relational database on Amazon EC2 or in another computing environment.

Ideal Usage Patterns

* RDS is a great solution for cloud-based **fully-managed relational database**
* RDS is also optimal for new applications with **structured data that requires more sophisticated querying and joining capabilities** than that provided by Amazon’s NoSQL database offering, DynamoDB.
* RDS provides **full compatibility with the databases supported and direct access to native database engines, code and libraries** and is ideal for existing applications that rely on these databases

Anti-Patterns

* Index and query-focused data
  + If the applications **don’t require advanced features such as joins and complex transactions** and is more oriented toward indexing and querying data, **DynamoDB** would be more appropriate for this needs
* Numerous BLOBs
  + If the application **makes heavy use of files (audio files, videos, images, etc), it is a better choice to use S3 to store the objects** instead of database engines Blob feature and use RDS or DynamoDB only to save the metadata
* Automated scalability
  + RDS provides **pushbutton scaling and it only scales up and has limited scale out ability. If fully-automated scaling is needed**, DynamoDB may be a better choice.
* Complete control
  + RDS does not **provide admin access and does not enable the full feature set of the database engines**.
  + So if the application requires complete, OS-level control of the database server with full root or admin login privileges, a self-managed database on EC2 may be a better match.
* Other database platforms
  + RDS, at this time, **provides a MySQL, Oracle, MariaDB, PostgreSQL and SQL Server databases**.
  + If any other database platform (such as IBM DB2, Informix, or Sybase) is needed, it should be deployed on a self-managed database on an EC2 instance by using a relational database AMI, or by installing database software on an EC2 instance.

Performance

* RDS Provisioned IOPS, where the IOPS can be specified when the instance is launched and is guaranteed over the life of the instance, provides a high-performance storage option designed to deliver fast, predictable, and consistent performance for I/O intensive transactional database workload

Durability and Availability

* RDS leverages Amazon EBS volumes as its data store
* RDS provides database backups, for enhanced durability, which are replicated across multiple AZ’s
  + Automated backups
    - If enabled, RDS will automatically perform a full daily backup of your data during the specified backup window, and will also capture DB transaction logs
  + User initiated backups
    - User can initiate backups at time and they are not deleted unless deleted explicitly by the user
* RDS Multi AZ’s feature enhances both the durability and the availability of the database by synchronously replicating the data between a primary RDS DB instance and a standby instance in another Availability Zone, which prevents data loss,
* RDS provides a DNS endpoint and in case of an failure on the primary, it automatically fails over to the standby instance
* RDS also allows Read replicas for the supported databases, which are replicated asynchronously

Cost Model

* RDS offers a tiered pricing structure, based on the size of the database instance, the deployment type (Single-AZ/Multi-AZ), and the AWS region.
* Pricing for RDS is based on several factors: the DB instance hours (per hour), the amount of provisioned database storage (per GB-month and per million I/O requests), additional backup storage (per GB-month), and data transfer in/out (per GB per month)

Scalability and Elasticity

* RDS resources can be scaled elastically in several dimensions: database storage size, database storage IOPS rate, database instance compute capacity, and the number of read replicas
* RDS supports “pushbutton scaling” of both database storage and compute resources. Additional storage can either be added immediately or during the next maintenance cycle
* RDS for MySQL also enables you to scale out beyond the capacity of a single database deployment for read-heavy database workloads by creating one or more read replicas.
* Multiple RDS instances can also be configured to leverage database partitioning or sharding to spread the workload over multiple DB instances, achieving even greater database scalability and elasticity.

Interfaces

* RDS APIs and the AWS Management Console provide a management interface that allows you to create, delete, modify, and terminate RDS DB instances; to create DB snapshots; and to perform point-in-time restores
* There is no AWS data API for Amazon RDS.
* Once a database is created, RDS provides a DNS endpoint for the database which can be used to connect to the database.
* Endpoint does not change over the lifetime of the instance even during the failover in case of Multi-AZ configuration

Amazon DynamoDB

* Amazon DynamoDB is a fast, fully-managed NoSQL database service that makes it simple and cost-effective to store and retrieve any amount of data, and serve any level of request traffic.
* DynamoDB being a managed service helps offload the administrative burden of operating and scaling a highly-available distributed database cluster.
* DynamoDB helps meet the latency and throughput requirements of highly demanding applications by providing extremely fast and predictable performance with seamless throughput and storage scalability.
* DynamoDB provides both **eventually-consistent reads (by default), and strongly-consistent reads (optional)**, as well as implicit item-level transactions for item put, update, delete, conditional operations, and increment/decrement.
* Amazon DynamoDB handles the data as below :-
  + DynamoDB stores structured data in tables, indexed by primary key, and allows low-latency read and write access to items.
  + DynamoDB supports three data types: number, string, and binary, in both scalar and multi-valued sets.
  + Tables do not have a fixed schema, so each data item can have a different number of attributes.
  + Primary key can either be a single-attribute hash key or a composite hash-range key.
  + Local secondary indexes provide additional flexibility for querying against attributes other than the primary key.

Ideal Usage Patterns

* DynamoDB is ideal for existing or new applications that need a **flexible NoSQL database with low read and write latencies, and the ability to scale storage and throughput up or down as needed without code changes or downtime**.
* Use cases require a **highly available and scalable database because downtime or performance degradation has an immediate negative impact**on an organization’s business. *for e.g. mobile apps, gaming, digital ad serving, live voting and audience interaction for live events, sensor networks, log ingestion, access control for web-based content, metadata storage for S3 objects, e-commerce shopping carts, and web session management*

Anti-Patterns

* Structured data with Join and/or Complex Transactions
  + If the application **uses structured data and required joins, complex transactions or other relationship infrastructure provided by traditional database platforms**, it is better to use RDS or Database installed on an EC2 instance
* Large Blob data
  + If the application **uses large blob data *for e.g. media, files, videos etc.*, it is better to use S3 to store the objects and use DynamoDB to store metadata** *for e.g. name, size, content-type etc*
* Large Objects with Low I/O rate
  + DynamoDB uses SSD drives and is optimized for workloads with a high I/O rate per GB stored. If the **applications stores very large amounts of data that are infrequently accessed**, S3 might be a better choice
* Prewritten application with databases
  + For Porting an **existing application using databases**, RDS or database installed on the EC2 instance would be a better and seamless solution

Performance

* SSDs and limited indexing on attributes provides high throughput and low latency and drastically reduces the cost of read and write operations.
* Predictable performance can be achieved by defining the provisioned throughput capacity required for a given table.
* DynamoDB handles the provisioning of resources to achieve the requested throughput rate, taking away the burden to think about instances, hardware, memory, and other factors that can affect an application’s throughput rate.
* Provisioned throughput capacity reservations are elastic and can be increased or decreased on demand.

Durability and Availability

* DynamoDB has **built-in fault tolerance that automatically and synchronously replicates data across three AZ’s in a region** for high availability and to help protect data against individual machine, or even facility failures.

Cost Model

* DynamoDB has three pricing components: provisioned throughput capacity (per hour), indexed data storage (per GB per month), data transfer in or out (per GB per month)

Scalability and Elasticity

* DynamoDB is both highly-scalable and elastic.
* DynamoDB provides unlimited storage capacity, and the service automatically allocates more storage as the demand increases
* Data is automatically partitioned and re-partitioned as needed, while the use of SSDs provides predictable low-latency response times at any scale.
* DynamoDB is also elastic, in that you can simply “dial-up” or “dial-down” the read and write capacity of a table as your needs change.

Interfaces

* DynamoDB provides a low-level REST API, as well as higher-level SDKs in different languages
* APIs provide both a management and data interface for Amazon DynamoDB, that enable table management (creating, listing, deleting, and obtaining metadata) and working with attributes (getting, writing, and deleting attributes; query using an index, and full scan).

Databases on EC2

* EC2 with EBS volumes allows hosting a self managed relational database
* Ready to use, prebuilt AMIs are also available from leading database solutions

Ideal Usage Patterns

* Self managed database on EC2 is an ideal scenario for users whose application **requires a specific traditional relational database not supported by Amazon RDS***for e.g. IBM DB2, Informix, or Sybase*
* **Users or applications that require a maximum level of administrative control and configurability**which is not provided by RDS

Anti-Patterns

* Index and query-focused data
  + If the applications **don’t require advanced features such as joins and complex transactions** and is more oriented toward indexing and querying data, DynamoDB would be more appropriate for this needs
* Numerous BLOBs
  + If the application **makes heavy use of files (audio files, videos, images, and so on)**, it is a better choice to use S3 to store the objects instead of database engines Blob feature and use RDS or DynamoDB only to save the metadata
* Automated scalability
  + Relational databases on EC2 leverages the scalability and elasticity of the underlying AWS platform, but this requires system administrators or DBAs to perform a manual or scripted task**. If you need pushbutton scaling or fully-automated scaling**, DynamoDB or RDS may be a better choice.
* RDS supported database platforms
  + If the **application using RDS supported database engine and all the features are available**, RDS would be a better choice instead of self managed relational database on EC2

Performance

* Performance depends on the size of the underlying EC2 instance, the number and configuration of the EBS volumes and the database itself
* Performance can be increased by scaling up memory and compute resources by choosing a larger Amazon EC2 instance size.
* For database storage, it is usually best to use EBS Provisioned IOPS volumes. To scale up I/O performance, the Provisioned IOPS can be increased, the number of EBS volumes changed, or use software RAID 0 (disk striping) across multiple EBS volumes, which will aggregate total IOPS and bandwidth.

Durability & Availability

* As the database on EC2 uses EBS as storage, it has the same durability and availability provided by EBS and can be further enhanced by using EBS snapshots or by using third-party database backup utilities (such as Oracle’s RMAN) to store database backups in Amazon S3

Cost Model

* Cost for running a database on EC2 instance is mainly determined by the size and the number of EC2 instance running, the size of the EBS volume used for database storage and any third party licensing cost for the database

Scalability & Elasticity

* Users of traditional relational database solutions on Amazon EC2 can take advantage of the scalability and elasticity of the underlying AWS platform by creating AMI and spawning multiple instances

# **Relation Database Service – RDS Overview**

* Amazon Relational Database Service (RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud.
* RDS provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.
* RDS features & benefits
  + CPU, memory, storage, and IOPS can be scaled independently.
  + manages backups, software patching, automatic failure detection, and recovery.
  + automated backups can be performed as needed, or manual backups can be triggered as well. Backups can be used to restore a database, and the RDS restore process works reliably and efficiently.
  + provides high availability with a primary instance and a synchronous secondary instance that can be failovered to seamlessly when a problem occurs.
  + provides elasticity & scalability by enabling MySQL, MariaDB, or PostgreSQL Read Replicas to increase read scaling.
  + supports MySQL, MariaDB, PostgreSQL, Oracle, Microsoft SQL Server, and the new, MySQL-compatible Amazon Aurora DB engine
  + in addition to the security in the database package, IAM users and permissions can help to control who has access to the RDS databases
  + databases can be further protected by putting them in a VPC, using SSL for data in transit and encryption for data in rest
  + However, **as it is a managed service, shell (root ssh) access to DB instances is not provided**, and this restricts access to certain system procedures and tables that require advanced privileges.

## RDS Components

* **DB Instance**
  + is a basic building block of RDS
  + is an isolated database environment in the cloud
  + each DB instance runs a DB engine. AWS currently supports MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server & Aurora DB engines
  + can be accessed from Amazon AWS command line tools, Amazon RDS  
    APIs, or the AWS Management RDS Console.
  + computation and memory capacity of an DB instance is determined by its DB instance class, which can be selected as per the needs
  + for each DB instance, 5 GB to 6 TB of associated storage capacity can be selected
  + storage comes in three types: Magnetic, General Purpose (SSD), and Provisioned IOPS (SSD), which differ in performance characteristics and price
  + each DB instance has a DB instance identifier, which is customer-supplied name and must be unique for that customer in an AWS region. It uniquely identifies the DB instance when interacting with the Amazon RDS API and AWS CLI commands.
  + each DB instance can host multiple databases, or a single Oracle database with multiple schemas.
  + can be hosted in an AWS VPC environment for better control
* **Regions and Availability Zones**
  + AWS resources are housed in highly available data center facilities in different areas of world, these data centers are called regions which further contain multiple distinct locations called Availability Zones
  + Each AZ is engineered to be isolated from failures in other AZs, and to provide inexpensive, low-latency network connectivity to other AZs in the same region
  + DB instances can be hosted in several AZs, an option called a Multi-AZ deployment.
    - Amazon automatically provisions and maintains a synchronous standby replica of the DB instance in a different AZ.
    - Primary DB instance is synchronously replicated across AZs to the standby replica
    - Provides data redundancy, failover support, eliminate I/O freezes, and minimize latency spikes during system backups.
* **Security Groups**
  + security group controls the access to a DB instance, by allowing access to the specified IP address ranges or EC2 instances
* **DB Parameter Groups**
  + A DB parameter group contains engine configuration values that can be applied to one or more DB instances of the same instance type
* **DB Option Groups**
  + Some DB engines offer tools that simplify managing the databases and making the best use of data.
  + Amazon RDS makes such tools available through option groups for e.g. Oracle Application Express (APEX), SQL Server Transparent Data Encryption, and MySQL memcached support.

## RDS Interfaces

* RDS can be interacted with multiple interfaces
  + AWS RDS Management console
  + Command Line Interface
  + Programmatic Interfaces which include SDKs, libraries in different languages, and RDS API

## RDS Pricing

* Instance class
  + Pricing is based on the class (e.g., micro, small, large, xlarge) of the DB instance consumed.
* Running time
  + Billed by the instance-hour, which is equivalent to a single instance running for an hour for e.g., a single instance running for two hours = two instances running for one hour, both consume 2 instance-hours.
  + if a DB instance runs for only part of an hour, full instance-hour is charged
* Storage
  + Storage capacity provisioned for the DB instance is billed per GB per month.
  + If the provisioned storage capacity is scaled within the month, the bill will be pro-rated.
* I/O requests per month
  + Total number of storage I/O requests made in a billing cycle.
* Backup storage
  + Automated backups & any active database snapshots consume storage
  + Increasing backup retention period or taking additional database snapshots increases the backup storage consumed by the database.
  + RDS provides backup storage up to 100% of the provisioned database storage at no additional charge for e.g., if you have 10 GB-months of provisioned database storage, RDS provides up to 10 GB-months of backup storage at no additional charge.
  + Most databases require less raw storage for a backup than for the primary dataset, so if multiple backups are not maintained, you will never pay for backup storage.
  + Backup storage is free only for active DB instances.
* Data transfer
  + Internet data transfer in and out of your DB instance.
* Reserved Instance
  + In addition to regular RDS pricing, reserved DB instances can be purchased

# **RDS Multi-AZ & Read Replica Overview**

* DB instances replicas can be created in two ways Multi-AZ & Read Replica
* **Multi-AZ deployment**
  + Multi-AZ deployment provides high availability and failover support
  + RDS automatically provisions and manages a **synchronous** standby replica in a different AZ (independent infrastructure in a physically separate location)
  + RDS automatically fails over to the standby so that database operations can resume quickly without administrative intervention in case of
    - Planned database maintenance
    - Software patching
    - Rebooting of the Primary instance
    - Primary DB instance connectivity or host failure, or an
    - Availability Zone failure
* **Read Replica**
  + RDS uses the PostgreSQL, MySQL, and MariaDB DB engines’ built-in replication functionality to create a special type of DB instance called a Read Replica from a source DB instance.
  + Load on the source DB instance can be reduced by routing read queries from applications to the Read Replica.
  + Read Replicas allow elastic scaling beyond the capacity constraints of a single DB instance for read-heavy database workloads

## Multi-AZ deployment

* Multi-AZ deployments provides high availability and automatic failover support for DB instances
* Multi-AZ helps improve the durability and availability of a critical system, enhancing availability during planned system maintenance, DB instance failure and Availability Zone disruption.
* **Multi-AZ is a High Availability feature is not a scaling solution for read-only scenarios; standby replica can’t be used to serve read traffic. To service read-only traffic, use a Read Replica.**
* **Multi-AZ deployments for Oracle, PostgreSQL, MySQL, and MariaDB DB instances use Amazon technology, while SQL Server DB instances use SQL Server Mirroring.**
* In a Multi-AZ deployment,
  + RDS automatically provisions and maintains a **synchronous standby replica in a different Availability Zone**.
  + Copies of data are stored in different Availability Zones for greater levels of data durability.
  + Primary DB instance is **synchronously replicated** across Availability Zones to a standby replica to provide
    - data redundancy,
    - eliminate I/O freezes during snapshots and backups
    - and minimize latency spikes during system backups.
  + DB instances may have increased write and commit latency compared to a Single AZ deployment, due to the synchronous data replication
  + Transaction success is returned only if the commit is successful both on the primary and the standby DB
  + There might be a change in latency if the deployment fails over to the standby replica, although AWS is engineered with low-latency network connectivity between Availability Zones.
* When using the BYOL licensing model, a license for both the primary instance and the standby replica is required
* For production workloads, it is recommended to use Multi-AZ deployment with Provisioned IOPS and DB instance classes (m1.large and larger), optimized for Provisioned IOPS for fast, consistent performance.
* When Single-AZ deployment is modified to a Multi-AZ deployment (for engines other than SQL Server or Amazon Aurora)
  + RDS takes a snapshot of the primary DB instance from the deployment and restores the snapshot into another Availability Zone.
  + RDS then sets up synchronous replication between the primary DB instance and the new instance.
  + This avoids downtime when conversion from Single AZ to Multi-AZ

### RDS Multi-AZ Failover Process

* In the event of a planned or unplanned outage of the DB instance,
  + RDS automatically switches to a standby replica in another AZ, if enabled for Multi-AZ.
  + Time it takes for the failover to complete depends on the database activity and other conditions at the time the primary DB instance became unavailable.
  + Failover times are typically 60-120 secs. However, large transactions or a lengthy recovery process can increase failover time.
  + **Failover mechanism automatically changes the DNS record of the DB instance to point to the standby DB instance.**
  + Multi-AZ switch is seamless to the applications as there is no change in the endpoint URLs but just needs to re-establish any existing connections to the DB instance.
* RDS handles failover automatically so that database operations can be resumed as quickly as possible without administrative intervention.
* Primary DB instance switches over automatically to the standby replica if any of the following conditions occur:
  + An Availability Zone outage
  + Primary DB instance fails
  + DB instance’s server type is changed
  + Operating system of the DB instance is undergoing software patching
  + A manual failover of the DB instance was initiated using **Reboot with failover**(also referred to as **Forced Failover**)
* If the Multi-AZ DB instance has failed over, can be determined by
  + DB event subscriptions can be setup to notify you via email or SMS that a failover has been initiated.
  + DB events can be viewed via the Amazon RDS console or APIs.
  + Current state of your Multi-AZ deployment can be viewed via the RDS console and APIs.

## Read Replica

* Amazon RDS uses the **MySQL, MariaDB, and PostgreSQL** (version 9.3.5 and later) DB engines’ built-in replication functionality to create a Read Replica from a source DB instance.
* Updates made to the source DB instance are **asynchronously** copied to the Read Replica.
* Load on the source DB instance can be reduced by routing read queries from the applications to the Read Replica.
* Using Read Replicas allow DB to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads.
* Read Replica operates as a DB instance that allows read-only connections; applications can connect to a Read Replica the same way they would to any DB instance.

### Read Replica creation

* Up to five Read Replicas can be created from one source DB instance.
* Creation process
  + Automatic backups must be enabled on the source DB instance by setting the backup retention period to a value other than 0
  + Existing DB instance needs to be specified as the source.
  + RDS takes a snapshot of the source instance and creates a read-only instance from the snapshot.
  + RDS then uses the **asynchronous replication** method for the DB engine to update the Read Replica for any changes to the source DB instance.
* RDS replicates all databases in the source DB instance.
* RDS sets up a secure communications channel between the source DB instance and the Read Replica, if that Read Replica is in a different AWS region from the DB instance.
* RDS establishes any AWS security configurations, such as adding security group entries, needed to enable the secure channel.
* During the Read Replica creation, a brief I/O suspension on the source DB instance can be experienced as the DB snapshot occurs.
* **I/O suspension typically lasts about one minute and can be avoided if the source DB instance is a Multi-AZ deployment (in the case of Multi-AZ deployments, DB snapshots are taken from the standby)**.
* Read Replica creation time can be slow if any long-running transactions are being executed and should wait for completion
* For multiple Read Replicas created in parallel from the same source DB instance, only one snapshot is taken at the start of the first create action.
* A Read Replica can be promoted to a new independent source DB, in which case the replication link is broken between the Read Replica and the source DB.  However, the replication continues for other replicas using the original source DB as the replication source

### Read Replica Deletion & DB Failover

* Read Replicas must be explicitly deleted, using the same mechanisms for deleting a DB instance.
* If the source DB instance is deleted without deleting the replicas, each replica is promoted to a stand-alone, single-AZ DB instance.
* If the source instance of a Multi-AZ deployment fails over to the standby, any associated Read Replicas are switched to use the secondary as their replication source.

### Read Replica Storage & Compute requirements

* A Read Replica, by default, is created with the same storage type as the source DB instance.
* For replication to operate effectively, each Read Replica should have the same amount of compute & storage resources as the source DB instance.
* Source DB instance, if scaled, Read Replicas should be scaled accordingly

### Read Replica Features & Limitations

* RDS does not support circular replication.
* DB instance cannot be configured to serve as a replication source for an existing DB instance; a new Read Replica can be created only from an existing DB instance for e.g., if MyDBInstance replicates to ReadReplica1, ReadReplica1 can’t be configured to replicate back to MyDBInstance.  From ReadReplica1, only a new Read Replica can be created, such as ReadRep2.
* Cross-Region Replication
  + **MySQL, PostgresSQL**(update from June 2016)**or MariaDB Read Replica can be created in a different region** than the source DB instance which helps to improve
    - disaster recovery capabilities (reduces RTO and RPO),
    - scale read operations into a region closer to end users,
    - migration from a data center in one region to another region
* Read Replica can be created from other Read replicas as well. However, the replica lag is higher for these instances and there cannot be more than four instances involved in a replication chain.

### Read Replica ComparisionRead Replica Use cases

* Read Replicas can be used in variety of use cases, including:
  + Scaling beyond the compute or I/O capacity of a single DB instance for read-heavy database workloads, directing excess read traffic to Read Replica(s)
  + Serving read traffic while the source DB instance is unavailable for e.g. If the source DB instance cannot take I/O requests due to backups I/O suspension or scheduled maintenance, the read traffic can be directed to the Read Replica(s). **However, the data might be stale.**
  + Business reporting or data warehousing scenarios where business reporting queries can be executed against a Read Replica, rather than the primary, production DB instance.

# **AWS RDS Storage**

* RDS storage uses Elastic Block Store (EBS) volumes for database and log storage.
* RDS automatically stripes across multiple EBS volumes to enhance IOPS performance, depending on the amount of storage requested

## RDS Storage Types

* RDS storage provides three storage types: Magnetic, General Purpose (SSD), and Provisioned IOPS (input/output operations per second).
* These storage types differ in performance characteristics and price, which allows tailoring of storage performance and cost to the database needs
* MySQL, MariaDB, PostgreSQL, and Oracle RDS DB instances can be created with up to 6TB of storage and SQL Server RDS DB instances with up to 4TB of storage when using the Provisioned IOPS and General Purpose (SSD)  
  storage types.
* Existing MySQL, PostgreSQL, and Oracle RDS database instances can be scaled to these new database storage limits without any downtime.

### Magnetic (Standard)

* Magnetic storage, also called standard storage, offers cost-effective storage that is ideal for applications with light or burst I/O requirements.
* They deliver approximately 100 IOPS on average, with burst capability of up to hundreds of IOPS, and they can range in size from 5 GB to 3 TB, depending on the DB instance engine.
* Magnetic storage is not reserved for a single DB instance, so performance can vary greatly depending on the demands placed on shared resources by other customers.

### General Purpose (SSD)

* General purpose, SSD-backed storage, also called gp2, can provide faster access than disk-based storage.
* They can deliver single-digit millisecond latencies, with a base performance of 3 IOPS per Gigabyte (GB) and the ability to burst to 3,000 IOPS for extended periods of time up to a maximum of 10,000 PIOPS.
* Gp2 volumes can range in size from 5 GB to 6 TB for MySQL, MariaDB, PostgreSQL, and Oracle DB instances, and from 20 GB to 4 TB for SQL Server DB instances.
* Gp2 is excellent for small to medium-sized databases.

### Provisioned IOPS

* Provisioned IOPS storage is designed to meet the needs of I/O-intensive workloads, particularly database workloads, that are sensitive to storage performance and consistency in random access I/O throughput.
* Provisioned IOPS storage is a storage type that delivers fast, predictable, and consistent throughput performance.
* For any production application that requires fast and consistent I/O performance, Amazon recommends Provisioned IOPS (input/output operations per second) storage.
* Provisioned IOPS storage is optimized for I/O intensive, online transaction processing (OLTP) workloads that have consistent performance requirements.
* Provisioned IOPS helps performance tuning.
* Provisioned IOPS volumes can range in size from 100 GB to 6 TB for MySQL, MariaDB, PostgreSQL, and Oracle DB engines. SQL Server Express and Web editions can range in size from 100 GB to 4 TB, while SQL Server Standard and Enterprise editions can range in size from 200 GB to 4 TB.
* Dedicated IOPS rate and storage space allocation is specified, when a DB instance is created. RDS provisions that IOPS rate and storage for the lifetime of the DB instance or until its changed.
* RDS delivers within 10 percent of the provisioned IOPS performance 99.9 percent of the time over a given year.

For detailed explanation on refer post @ [EBS volume Types](http://jayendrapatil.com/aws-ebs-volume-types/)

### Adding Storage and Changing Storage Type

* DB instance can be modified to use additional storage and converted to a different storage type.
* **However, storage allocated for a DB instance cannot be decreased**
* MySQL, MariaDB, PostgreSQL, and Oracle DB instances can be scaled up for storage, which helps improve I/O capacity.
* **Storage capacity nor the type of storage for a SQL Server DB instance can be changed due to extensibility limitations of striped storage attached to a Windows Server environment.**
* During the scaling process, the DB instance will be available for reads and writes, but may experience performance degradation
* Adding storage may take several hours; the duration of the process depends on several factors such as load, storage size, storage type, amount of IOPS provisioned (if any), and number of prior scale storage operations.
* While storage is being added, nightly backups are suspended and no other RDS operations can take place, including modify, reboot, delete, create Read Replica, and create DB Snapshot

## Performance Metrics

* Amazon RDS provides several metrics that can be used to determine how the DB instance is performing.
  + **IOPS**
    - the number of I/O operations completed per second.
    - it is reported as the average IOPS for a given time interval.
    - RDS reports read and write IOPS separately on one minute intervals.
    - Total IOPS is the sum of the read and write IOPS.
    - Typical values for IOPS range from zero to tens of thousands per second.
  + **Latency**
    - the elapsed time between the submission of an I/O request and its completion
    - it is reported as the average latency for a given time interval.
    - RDS reports read and write latency separately on one minute intervals in units of seconds.
    - Typical values for latency are in the millisecond (ms)
  + **Throughput**
    - the number of bytes per second transferred to or from disk
    - it is reported as the average throughput for a given time interval.
    - RDS reports read and write throughput separately on one minute intervals using units of megabytes per second (MB/s).
    - Typical values for throughput range from zero to the I/O channel’s maximum bandwidth.
  + **Queue Depth**
    - the number of I/O requests in the queue waiting to be serviced.
    - these are I/O requests that have been submitted by the application but have not been sent to the device because the device is busy servicing other I/O requests.
    - it is reported as the average queue depth for a given time interval.
    - RDS reports queue depth in one minute intervals. Typical values for queue depth range from zero to several hundred.
    - Time spent waiting in the queue is a component of Latency and  
      Service Time (not available as a metric).

## Amazon RDS Storage Facts

* First time a DB instance is started and accesses an area of disk for the first time, the process can take longer than all subsequent accesses to the same disk area. This is known as the “**first touch penalty**”. Once an area of disk has incurred the first touch penalty, that area of disk does not incur the penalty again for the life of the instance, even if the DB instance is rebooted, restarted, or the DB instance class changes. Note that a DB instance created from a snapshot, a point-in-time restore, or a read replica is a new instance and does incur this first touch penalty.
* RDS manages the DB instance and it reserves overhead space on the instance. While the amount of reserved storage varies by DB instance class and other factors, this reserved space can be as much as one or two percent of the total storage
* Provisioned IOPS provides a way to reserve I/O capacity by specifying IOPS. Like any other system capacity attribute, maximum throughput under load will be constrained by the resource that is consumed first, which could be IOPS, channel bandwidth, CPU, memory, or database internal resources.
* Current maximum channel bandwidth available is 4000 megabits per second (Mbps) full duplex. In terms of the read and write throughput metrics, this equates to about 210 megabytes per second (MB/s) in each direction. A perfectly balanced workload of 50% reads and 50% writes may attain a maximum combined throughput of 420 MB/s, which includes protocol overhead, so the actual data throughput may be less.
* Provisioned IOPS works with an I/O request size of 32 KB. Provisioned IOPS consumption is a linear function of I/O request size above 32 KB. An I/O request smaller than 32 KB is handled as one I/O; for e.g. 1000 16 KB I/O requests are treated the same as 1000 32 KB requests. I/O requests larger than 32 KB consume more than one I/O request; while, a 48 KB I/O request consumes 1.5 I/O requests of storage capacity; a 64 KB I/O request consumes 2 I/O requests

## Factors That Impact Storage Performance

* Several factors can affect the performance of a DB instance, such as instance configuration, I/O characteristics, and workload demand.
* System related activities also consume I/O capacity and may reduce database instance performance while in progress:
  + DB snapshot creation
  + Nightly backups
  + Multi-AZ peer creation
  + Read replica creation
  + Scaling storage
* System resources can constrain the throughput of a DB instance, but there can be other reasons for a bottleneck. Database could be the issue if :-
  + Channel throughput limit is not reached
  + Queue depths are consistently low
  + CPU utilization is under 80%
  + Free memory available
  + No swap activity
  + Plenty of free disk space
  + Application has dozens of threads all submitting transactions as fast as the database will take them, but there is clearly unused I/O capacity.

AWS RDS Security

* AWS provides multiple features to provide RDS security
  + DB instance can be hosted in a VPC for the greatest possible network access control
  + IAM policies can be used to assign permissions that determine who is allowed to manage RDS resources
  + Security groups allow to control what IP addresses or EC2 instances can connect to the databases on a DB instance
  + Secure Socket Layer (SSL) connections with DB instances
  + RDS encryption to secure RDS instances and snapshots at rest.
  + Network encryption and transparent data encryption (TDE) with Oracle DB instances

RDS Authentication and Access Control

* IAM can be used to control which RDS operations each individual user has permission to call

Encrypting RDS Resources

* RDS encrypted instances use the industry standard AES-256 encryption algorithm to encrypt data on the server that hosts the RDS instance
* RDS then handles authentication of access and decryption of this data with a minimal impact on performance, and with no need to modify your database client applications
* Data at Rest Encryption
  + can be enabled on RDS instances to encrypt the underlying storage
  + encryption keys are managed by KMS
  + can be enabled only during instance creation
  + once enabled, the encryption keys cannot be changed
  + if the key is lost, the DB can only be restored from the backup
* Once encryption is enabled for an RDS instance,
  + logs are encrypted
  + snapshots are encrypted
  + automated backups are encrypted
  + read replicas are encrypted
* Cross region replicas and snapshots copy does not work since the key is only available in a single region
* RDS DB Snapshot considerations
  + DB snapshot encrypted using an KMS encryption key can be copied
  + Copying an encrypted DB snapshot, results in an encrypted copy of the DB snapshot
  + When copying, DB snapshot can either be encrypted with the same KMS encryption key as the original DB snapshot, or a different KMS encryption key to encrypt the copy of the DB snapshot.
  + An unencrypted DB snapshot can be copied to an encrypted snapshot, a quick way to add encryption to a previously encrypted DB instance.
  + Encrypted snapshot can be restored only to an encrypted DB instance
  + If a KMS encryption key is specified when restoring from an unencrypted DB cluster snapshot, the restored DB cluster is encrypted using the specified KMS encryption key
  + Copying an encrypted snapshot shared from another AWS account, requires access to the KMS encryption key used to encrypt the DB snapshot.
  + Because KMS encryption keys are specific to the region that they are created in, encrypted snapshot cannot be copied to another region
* Transparent Data Encryption (TDE)
  + Automatically encrypts the data before it is written to the underlying storage device and decrypts when it is read  from the storage device
  + is supported by Oracle and SQL Server
    - Oracle requires key storage outside of the KMS and integrates with CloudHSM for this
    - SQL Server requires a key but is managed by RDS

SSL to Encrypt a Connection to a DB Instance

* Encrypt connections using SSL for data in transit between the applications and the DB instance
* Amazon RDS creates an SSL certificate and installs the certificate on the DB instance when RDS provisions the instance.
* SSL certificates are signed by a certificate authority. SSL certificate includes the DB instance endpoint as the Common Name (CN) for the SSL certificate to guard against spoofing attacks
* While SSL offers security benefits, be aware that SSL encryption is a compute-intensive operation and will increase the latency of the database connection.

RDS Security Groups

* Security groups control the access that traffic has in and out of a DB instance
* VPC security groups act like a firewall controlling network access to your DB instance.
* VPC security groups can be configured and associated with the DB instance to allow access from an IP address range, port, or EC2 security group
* Database security groups default to a “deny all” access mode and customers must specifically authorize network ingress.

Master User Account Privileges

* When you create a new DB instance, the default master user that used gets certain privileges for that DB instance
* Subsequently, other users with permissions can be created

Event Notification

* Event notifications can be configured for important events that occur on the DB instance
* Notifications of a variety of important events that can occur on the RDS instance, such as whether the instance was shut down, a backup was started, a failover occurred, the security group was changed, or your storage space is low can be received.

RDS DB Instance Maintenance and Upgrades

Changes to a DB instance can occur when a DB instance is manually modified *for e.g. DB engine version is upgraded, or when Amazon RDS performs maintenance on an instance*

Amazon RDS Maintenance

* Periodically, Amazon RDS performs maintenance on Amazon RDS resources, such as DB instances and most often involves updates to the DB instance’s operating system (OS).
* Maintenance items can either
  + be applied manually on a DB instance at ones convenience
  + or wait for the automatic maintenance process initiated by Amazon RDS during the defined weekly maintenance window.
* Maintenance window only determines when pending operations start, but does not limit the total execution time of these operations. Maintenance operations are not guaranteed to finish before the maintenance window ends, and can continue beyond the specified end time.
* Maintenance update availability can be checked both on the RDS console and by using the RDS API. And if an update is available, one can
  + Defer the maintenance items.
  + Apply the maintenance items immediately.
  + Schedule them to start during the next defined maintenance window
* Maintenance items marked as
  + **Required** cannot be deferred indefinitely, if deferred AWS will send a notify the time when the update will be performed next
  + **Available** and can be deferred indefinitely and the update will not be applied to the DB instance.
* **Required patching is automatically scheduled only for patches that are related to security and instance reliability**. Such patching occurs infrequently (typically once every few months) and seldom requires more than a fraction of your maintenance window.
* Maintenance items require that RDS take your DB instance offline for a short time. Maintenance that requires DB instance to be offline include scale compute operations, which generally take only a few minutes from start to finish, and required operating system or database patching.
* **Multi-AZ deployment for the DB instance reduces the impact of a maintenance event by following these steps:**
  + **Perform maintenance on the standby.**
  + **Promote the standby to primary.**
  + **Perform maintenance on the old primary, which becomes the new standby.**
* When database engine for the DB instance is modified in a Multi-AZ deployment, RDS upgrades both the primary and secondary DB instances at the same time. In this case, the database engine for the entire Multi-AZ deployment is shut down during the upgrade.

Operating System Updates

* Upgrades to the operating system are most often for security issues and should be done as soon as possible.
* OS updates on a DB instance can be applied at ones convenience or can wait for the maintenance process initiated by RDS to apply the update during the defined maintenance window
* DB instance is not automatically backed up when an OS update is applied, and should be backup up before the update is applied

Database Engine Version Upgrade

* DB instance engine version can be upgraded when a new DB engine version is supported by RDS.
* Database version upgrades consist of major and minor version upgrades.
  + Major database version upgrades
    - can contain changes that are not backward-compatible
    - RDS doesn’t apply major version upgrades automatically
    - DB instance should be manually modified and thoroughly tested before applying it to the production instances.
  + Minor version upgrades
    - Each DB engine handles minor version upgrade slightly differently  
      *for e.g. RDS automatically apply minor version upgrades to a DB instance running PostgreSQL, but must be manually applied to a DB instance running Oracle.*
* Amazon posts an announcement to the forums announcement page and sends a customer e-mail notification before upgrading an DB instance
* Amazon schedule the upgrades at specific times through the year, to help plan around them, because downtime is required to upgrade a DB engine version, even for Multi-AZ instances.
* RDS takes two DB snapshots during the upgrade process.
  + First DB snapshot is of the DB instance before any upgrade changes have been made. If the upgrade fails, it can be restored from the snapshot to create a DB instance running the old version.
  + Second DB snapshot is taken when the upgrade completes. After the upgrade is complete, database engine can’t be reverted to the previous version. For returning to the previous version, restore the first DB snapshot taken to create a new DB instance.
* If the DB instance is using read replication, all of the Read Replicas must be upgraded before upgrading the source instance.
* If the DB instance is in a Multi-AZ deployment, both the primary and standby replicas are upgraded at the same time and would result in an outage. The time for the outage varies based on your database engine, version, and the size of your DB instance.

RDS Maintenance Window

* Every DB instance has a weekly maintenance window defined during which any system changes are applied.
* Maintenance window is an opportunity to control when DB instance modifications and software patching occur, in the event either are requested or required.
* If a maintenance event is scheduled for a given week, it will be initiated during the 30 minute maintenance window as defined
* Maintenance events mostly complete during the 30 minute maintenance window, although larger maintenance events may take more time
* 30-minute maintenance window is selected at random from an 8-hour block of time per region. If you don’t specify a preferred maintenance window when you create the DB instance, Amazon RDS assigns a 30-minute maintenance window on a randomly selected day of the week.
* RDS will consume some of the resources on the DB instance while maintenance is being applied, minimally effecting performance.
* For some maintenance events, a Multi-AZ failover may be required for a maintenance update to complete.

AWS RDS Monitoring & Notification

* RDS integrates with CloudWatch and provides metrics for monitoring
* CloudWatch alarms can be created over a single metric that sends an SNS message when the alarm changes state
* RDS also provides SNS notification whenever any RDS event occurs

CloudWatch RDS Monitoring

* RDS DB instance can be monitored using CloudWatch, which collects and processes raw data from RDS into readable, near real-time metrics.
* The statistics are recorded for a period of two weeks, so that you can access historical information and gain a better perspective on how the service is performing.
* By default, RDS metric data is automatically sent to Amazon CloudWatch in 1-minute periods
* CloudWatch RDS Metrics
  + BinLogDiskUsage – Amount of disk space occupied by binary logs on the master. Applies to MySQL read replicas.
  + CPUUtilization – Percentage of CPU utilization.
  + DatabaseConnections – Number of database connections in use.
  + DiskQueueDepth – The number of outstanding IOs (read/write requests) waiting to access the disk.
  + FreeableMemory – Amount of available random access memory.
  + FreeStorageSpace – Amount of available storage space.
  + **ReplicaLag** – Amount of time a Read Replica DB instance lags behind the source DB instance. Applies to MySQL, MariaDB, and PostgreSQL Read Replicas.
  + SwapUsage – Amount of swap space used on the DB instance.
  + ReadIOPS – Average number of disk I/O operations per second.
  + WriteIOPS – Average number of disk I/O operations per second.
  + ReadLatency – Average amount of time taken per disk I/O operation.
  + WriteLatency – Average amount of time taken per disk I/O operation.
  + ReadThroughput – Average number of bytes read from disk per second.
  + WriteThroughput – Average number of bytes written to disk per second.
  + NetworkReceiveThroughput – Incoming (Receive) network traffic on the DB instance, including both customer database traffic and Amazon RDS traffic used for monitoring and replication.
  + NetworkTransmitThroughput – Outgoing (Transmit) network traffic on the DB instance, including both customer database traffic and Amazon RDS traffic used for monitoring and replication.

RDS Event Notification

* RDS uses the SNS to provide notification when an RDS event occurs
* RDS groups the events into categories, which can be subscribed so that a notification is sent when an event in that category occurs.
* Event category for a DB instance, DB cluster, DB snapshot, DB cluster snapshot, DB security group or for a DB parameter group can be subscribed
* Event notifications are sent to the email addresses provided during subscription creation
* Subscription can be easily turn off notification without deleting a subscription by setting the Enabled radio button to No in the RDS console or by setting the Enabled parameter to false using the CLI or RDS API.

AWS RDS Best Practices

AWS recommends RDS best practices in terms of Monitoring, Performance and security

Amazon RDS Basic Operational Guidelines

* **Monitoring**
  + Memory, CPU, and storage usage should be monitored.
  + CloudWatch can be setup for notifications when usage patterns change or when the capacity of deployment is approached, so that system performance and availability can be maintained
* **Scaling**
  + Scale up the DB instance when approaching storage capacity limits.
  + There should be some buffer in storage and memory to accommodate unforeseen increases in demand from the applications.
* **Backups**
  + Enable Automatic Backups and set the backup window to occur during the daily low in WriteIOPS.
* On a MySQL DB instance,
  + Do not create more than 10,000 tables using Provisioned IOPS or 1000 tables using standard storage. Large numbers of tables will significantly increase database recovery time after a failover or database crash. If you need to create more tables than recommended, set the innodb\_file\_per\_table parameter to 0.
  + Avoid tables in the database growing too large. Provisioned storage limits restrict the maximum size of a MySQL table file to 6 TB. Instead, partition the large tables so that file sizes are well under the 6 TB limit. This can also improve performance and recovery time.
* **Performance**
  + If the database workload requires more I/O than provisioned, recovery after a failover or database failure will be slow.
  + To increase the I/O capacity of a DB instance,
    - Migrate to a DB instance class with High I/O capacity.
    - Convert from standard storage to Provisioned IOPS storage, and use a DB instance class that is optimized for Provisioned IOPS.
    - if using Provisioned IOPS storage, provision additional throughput capacity.
* **Multi-AZ & Failover**
  + Deploy applications in all Availability Zones, so if an AZ goes down, applications in other AZs will still be available.
  + Use Amazon RDS DB events to monitor failovers.
  + Set a TTL of less than 30 seconds, if the client application is caching the DNS data of the DB instances. As the underlying IP address of a DB instance can change after a failover, caching the DNS data for an extended time can lead to connection failures if the application tries to connect to an IP address that no longer is in service.
  + Multi-AZ requires transaction logging feature to be enabled. Do not use features like Simple recover mode, offline mode or Read-only mode which turn of transaction logging.
  + To shorten failover time
    - Ensure that sufficient Provisioned IOPS allocated for your workload. Inadequate I/O can lengthen failover times. Database recovery requires I/O.
    - Use smaller transactions. Database recovery relies on transactions, so break up large transactions into multiple smaller transactions to shorten failover time
  + Test failover for your DB instance to understand how long the process takes for your use case and to ensure that the application that accesses your DB instance can automatically connect to the new DB instance after failover.

DB Instance RAM Recommendations

* An Amazon RDS performance best practice is to allocate enough RAM so that the working set resides almost completely in memory.
* Value of ReadIOPS should be small and stable.
* ReadIOPS metric can be checked, using AWS CloudWatch while the DB instance is under load, to tell if the working set is almost all in memory
* If scaling up the DB instance class with more RAM, results in a dramatic drop in ReadIOPS, the working set was not almost completely in memory.
* Continue to scale up until ReadIOPS no longer drops dramatically after a scaling operation, or ReadIOPS is reduced to a very small amount.

Amazon RDS Security Best Practices

* Do not use AWS root credentials to manage Amazon RDS resources; and IAM users should be created for everyone,
* Grant each user the minimum set of permissions required to perform his or her duties.
* Use IAM groups to effectively manage permissions for multiple users.
* Rotate your IAM credentials regularly.

Using Enhanced Monitoring to Identify Operating System Issues

* Amazon RDS provides metrics in real time for the operating system (OS) that your DB instance runs on.
* Enhanced monitoring is available for all DB instance classes except for db.t1.micro and db.m1.small.

Using Metrics to Identify Performance Issues

* To identify performance issues caused by insufficient resources and other common bottlenecks, you can monitor the metrics available for your Amazon RDS DB instance
* Performance metrics should be monitored on a regular basis to benchmark  the average, maximum, and minimum values for a variety of time ranges. to help identify performance degradation.
* Amazon CloudWatch alarms can be set for particular metric thresholds to be alerted when they are reached or breached
* A DB instance has a number of different categories of metrics which includes CPU, memory, disk space, IOPS, db connections and network traffic, and how to determine acceptable values depends on the metric.
* One of the best ways to improve DB instance performance is to tune the most commonly used and most resource-intensive queries to make them less expensive to run.

Recovery

* **MySQL**
  + InnoDB is the recommended and supported storage engine for MySQL DB instances on Amazon RDS.
  + However, MyISAM performs better than InnoDB if you require intense, full-text search capability.
  + Point-In-Time Restore and snapshot restore features of Amazon RDS for MySQL require a crash-recoverable storage engine and are supported for the InnoDB storage engine only.
  + Although MySQL supports multiple storage engines with varying capabilities, not all of them are optimized for crash recovery and data durability.
  + MyISAM storage engine does not support reliable crash recovery and might prevent a Point-In-Time Restore or snapshot restore from working as intended which might result in lost or corrupt data when MySQL is restarted after a crash.
* **MariaDB**
  + XtraDB is the recommended and supported storage engine for MariaDB DB instances on Amazon RDS.
  + Point-In-Time Restore and snapshot restore features of Amazon RDS for MariaDB require a crash-recoverable storage engine and are supported for the XtraDB storage engine only.
  + Although MariaDB supports multiple storage engines with varying capabilities, not all of them are optimized for crash recovery  
    and data durability.
  + *For e.g although Aria is a crash-safe replacement for MyISAM, it might still prevent a Point-In-Time Restore or snapshot restore from working as intended. This might result in lost or corrupt data when MariaDB is restarted after a crash.*

# **RDS Back Up, Restore and Snapshots**

* RDS creates a **storage volume snapshot** of the DB instance, backing up the entire DB instance and not just individual databases.
* RDS provides two different methods Automated and Manual for backing up your DB instances:

## ****Automated backups****

* Backups of the DB instance are automatically created and retained
* Automated backups are enabled by default for a new DB instance.
* Automated backup occurs during a daily user-configurable period of time, known as **preferred backup window.**
  + If a preferred backup window is not specified when an DB instance is created, RDS assigns a default 30-minute backup window which is selected at random from an 8-hour block of time per region.
  + Changes to the backup window take effect immediately.
  + Backup window cannot overlap with the weekly maintenance window for the DB instance.
* Backups created during the backup window are retained for a user-configurable number of days , known as **backup retention period**
  + If the backup retention period is not set, RDS defaults the period retention period to one day one day if created using RDS API or the AWS CLI, or seven days if created AWS Console
  + Backup retention period can be modified with valid values are 0 (for no backup retention) to a maximum of 35 days.
* Manual snapshot limits (50 per region) do not apply to automated backups
* If the backup requires more time than allotted to the backup window, the backup will continue to completion.
* An immediate outage occurs if the backup retention period is changed
  + from 0 to a non-zero value as the first backup occurs immediately or
  + from non-zero value to 0 as it turns off automatic backups, and deletes all existing automated backups for the instance.
* **RDS uses the periodic data backups in conjunction with the transaction logs to enable restoration of the DB Instance to any second during the retention period, up to the LatestRestorableTime (typically up to the last few minutes).**
* During the backup window,
  + for Single AZ instance, storage I/O may be briefly suspended while the backup process initializes (typically under a few seconds) and a brief period of elevated latency might be experienced.
  + **for Multi-AZ DB deployments, there is No I/O suspension since the backup is taken from the standby instance**
* Automated DB snapshots are deleted when
  + the retention period expires
  + the automated DB snapshots for a DB instance is disabled
  + the DB instance is deleted
* When a DB instance is deleted,
  + a final DB snapshot can be created upon deletion; which can be used to restore the deleted DB instance at a later date.
  + RDS retains the final user-created DB snapshot along with all other manually created DB snapshots
  + **all automated backups are deleted and cannot be recovered**

### ****Point-In-Time Recovery****

* In addition to the daily automated backup, RDS archives database change logs. This enables recovery of the database to any point in time during the backup retention period, up to the last five minutes of database usage.
* Disabling automated backups also disables point-in-time recovery
* RDS stores multiple copies of your data, but for Single-AZ DB instances these copies are stored in a single availability zone.
* If for any reason a Single-AZ DB instance becomes unusable, point-in-time recovery can be used to launch a new DB instance with the latest restorable data

## DB Snapshots (User Initiated)

* Manual DB snapshots are user-initiated backups that enables to back up a  DB instance to a known state, and restore to that specific state at any time.
* **RDS keeps all manual DB snapshots until explicitly deleted**

### DB Snapshots Creation

* DB snapshot is a user-initiated storage volume snapshot of DB instance, backing up the entire DB instance and not just individual databases.
* DB snapshots enable backing up of the DB instance in a known state as needed, and can then be restored to that specific state at any time.
* DB snapshots are kept until explicitly deleted
* Creating DB snapshot on a Single-AZ DB instance results in a brief I/O suspension that typically lasting no more than a few minutes.
* Multi-AZ DB instances are not affected by this I/O suspension since the backup is taken on the standby instance

### DB Snapshot Restore

* DB instance can be restored to any specific time during this retention period, creating a new DB instance.
* New DB instance with a different endpoint is created by restoring from a DB snapshot
* RDS uses the periodic data backups in conjunction with the transaction logs to enable restoration of the DB Instance to any second during the retention period, up to the LatestRestorableTime (typically up to the last few minutes).
* Option group associated with the DB snapshot is associated with the restored DB instance once it is created. However, option group is associated with the VPC so would apply only when the instance is restored in the same VPC as the DB snapshot
* However, **the default DB parameter and security groups are associated with the restored instance**. After the restoration is complete, any custom DB parameter or security groups used by the instance restored from should be associated explicitly
* A DB instance can be restored with a different storage type than the source DB snapshot. In this case the restoration process will be slower because of the additional work required to migrate the data to the new storage type for e.g. from GP2 to Provisioned IOPS
* A DB instance can be restored with a different edition of the DB engine only if the DB snapshot has the required storage allocated for the new edition for e.g., to change from SQL Server Web Edition to SQL Server Standard Edition, the DB snapshot must have been created from a SQL Server DB instance that had at least 200 GB of allocated storage, which is the minimum allocated storage for SQL Server Standard edition

### DB Snapshot Copy

* Amazon RDS supports two types of DB snapshot copying.
  + Copy an automated DB snapshot to create a manual DB snapshot in the same AWS region. Manual DB snapshot are not deleted automatically and can be kept indefinitely.
  + Copy either an automated or manual DB snapshot from one region to another region. By copying the DB snapshot to another region, a manual DB snapshot is created that is retained in that region
* Manual DB snapshots can be shared with other AWS accounts and copy DB snapshots shared to you by other AWS accounts
* Snapshot Copy Encryption
  + DB snapshot that has been encrypted using an AWS Key Management System (AWS KMS) encryption key can be copied
  + Copying an encrypted DB snapshot, results in an encrypted copy of the DB snapshot
  + When copying, DB snapshot can either be encrypted with the same KMS encryption key as the original DB snapshot, or a different KMS encryption key to encrypt the copy of the DB snapshot.
  + An unencrypted DB snapshot can be copied to an encrypted snapshot, a quick way to add encryption to a previously encrypted DB instance.
  + Encrypted snapshot can be restored only to an encrypted DB instance
  + If a KMS encryption key is specified when restoring from an unencrypted DB cluster snapshot, the restored DB cluster is encrypted using the specified KMS encryption key
  + Copying an encrypted snapshot shared from another AWS account, requires access to the KMS encryption key that was used to encrypt the DB snapshot.
  + **NOTE** – AWS now allows copying encrypted DB snapshots between accounts and across multiple regions as seamlessly as unencrypted snapshots. Refer [blog post](https://aws.amazon.com/about-aws/whats-new/2016/12/amazon-rds-now-supports-copying-encrypted-snapshots-of-encrypted-db-instances-across-regions/)

### DB Snapshot Sharing

* Manual DB snapshot or DB cluster snapshot can be shared with up to 20 other AWS accounts.
* Manual snapshot shared with other AWS accounts can copy the snapshot, or restore a DB instance or DB cluster from that snapshot.
* Manual snapshot can also be shared as public, which makes the snapshot available to all AWS accounts. Care should be taken when sharing a snapshot as public so that none of the private information is included
* Shared snapshot can be copied to another region.
* However, following limitations apply when sharing manual snapshots with other AWS accounts:
  + When a DB instance or DB cluster is restored from a shared snapshot using the AWS CLI or RDS API, the Amazon Resource Name (ARN) of the shared snapshot as the snapshot identifier should be specified
  + DB snapshot that uses an option group with permanent or persistent options cannot be shared
  + A permanent option cannot be removed from an option group. Option groups with persistent options cannot be removed from a DB instance once the option group has been assigned to the DB instance.
* DB snapshots that have been encrypted “at rest” using the AES-256 encryption algorithm can be shared
* Users can only copy encrypted DB snapshots if they have access to the AWS Key Management Service (AWS KMS) encryption key that was used to encrypt the DB snapshot.
* AWS KMS encryption keys can be shared with another AWS account by adding the other account to the KMS key policy.
* However, KMS key policy must first be updated by adding any accounts to share the snapshot with, before sharing an encrypted DB snapshot