**Database**

**Amazon Aurora**

Q: What is Amazon Aurora?

Amazon Aurora is a relational database engine that combines the speed and reliability of high-end commercial databases with the simplicity and cost-effectiveness of open source databases. Amazon Aurora MySQL delivers up to five times the performance of MySQL without requiring any changes to most MySQL applications; similarly, Amazon Aurora PostgreSQL delivers up to three times the performance of PostgreSQL. Amazon RDS manages your Amazon Aurora databases, handling time-consuming tasks such as provisioning, patching, backup, recovery, failure detection and repair. You pay a simple monthly charge for each Amazon Aurora database instance you use. There are no upfront costs or long-term commitments required.

Q: What does "five times the performance of MySQL" mean?

Amazon Aurora delivers significant increases over MySQL performance by tightly integrating the database engine with an SSD-based virtualized storage layer purpose-built for database workloads, reducing writes to the storage system, minimizing lock contention and eliminating delays created by database process threads. Our tests with SysBench on r3.8xlarge instances show that Amazon Aurora delivers over 500,000 SELECTs/sec and 100,000 UPDATEs/sec, five times higher than MySQL running the same benchmark on the same hardware.

Q: How do I optimize my database workload for Amazon Aurora MySQL/ PostgreSQL?

Amazon Aurora is designed to be compatible with MySQL/ PostgreSQL, so that existing MySQL/ PostgreSQL applications and tools can run without requiring modification. However, one area where Amazon Aurora improves upon MySQL/ PostgreSQL is with highly concurrent workloads. In order to maximize your workload’s throughput on Amazon Aurora, we recommend building your applications to drive a large number of concurrent queries and transactions.

Q: What are the minimum and maximum storage limits of an Amazon Aurora database?

The minimum storage is 10GB. Based on your database usage, your Amazon Aurora storage will automatically grow, up to 64 TB, in 10GB increments with no impact to database performance. There is no need to provision storage in advance.

Q: How do I enable backups for my DB Instance?

Automated backups are always enabled on Amazon Aurora DB Instances. Backups do not impact database performance.

Q: Can I take DB Snapshots and keep them around as long as I want?

Yes, and there is no performance impact when taking snapshots. Note that restoring data from DB Snapshots requires creating a new DB Instance.

Q: If my database fails, what is my recovery path?

Amazon Aurora automatically maintains 6 copies of your data across 3 Availability Zones and will automatically attempt to recover your database in a healthy AZ with no data loss. In the unlikely event your data is unavailable within Amazon Aurora storage, you can restore from a DB Snapshot or perform a point-in-time restore operation to a new instance. Note that the latest restorable time for a point-in-time restore operation can be up to 5 minutes in the past.

Q: What happens to my automated backups and DB Snapshots if I delete my DB Instance?

You can choose to create a final DB Snapshot when deleting your DB Instance. If you do, you can use this DB Snapshot to restore the deleted DB Instance at a later date. Amazon Aurora retains this final user-created DB Snapshot along with all other manually created DB Snapshots after the DB Instance is deleted. Only DB Snapshots are retained after the DB Instance is deleted (i.e., automated backups created for point-in-time restore are not kept).

Q: Can I share my snapshots with another AWS account?

Yes. Aurora gives you the ability to create snapshots of your databases, which you can use later to restore a database. You can share a snapshot with a different AWS account, and the owner of the recipient account can use your snapshot to restore a DB that contains your data. You can even choose to make your snapshots public – that is, anybody can restore a DB containing your (public) data. You can use this feature to share data between your various environments (production, dev/test, staging, etc.) that have different AWS accounts, as well as keep backups of all your data secure in a separate account in case your main AWS account is ever compromised.

Q: Does Amazon Aurora encrypt my data in transit and at rest?

Yes. Amazon Aurora uses SSL (AES-256) to secure the connection between the database instance and the application. Amazon Aurora allows you to encrypt your databases using keys you manage through AWS Key Management Service (KMS). On a database instance running with Amazon Aurora encryption, data stored at rest in the underlying storage is encrypted, as are its automated backups, snapshots, and replicas in the same cluster. Encryption and decryption are handled seamlessly.

Q: What is Amazon Aurora Serverless?

[Amazon Aurora Serverless](https://aws.amazon.com/rds/aurora/serverless/) is an on-demand, autoscaling configuration for the MySQL-compatible edition of Amazon Aurora. An Aurora Serverless DB cluster automatically starts up, shuts down, and scales capacity up or down based on your application's needs. Aurora Serverless provides a relatively simple, cost-effective option for infrequent, intermittent, or unpredictable workloads.

Q: What benefits does Parallel Query provide?

Faster performance: Parallel Query can speed up analytical queries by up to 2 orders of magnitude.

Operational simplicity and data freshness: you can issue a query directly over the current transactional data in your Aurora cluster.

Transactional and analytical workloads on the same database: Parallel Query allows Aurora to maintain high transaction throughput alongside concurrent analytical queries.

**Amazon RDS**

Q: What is Amazon RDS?

Amazon Relational Database Service (Amazon RDS) is a managed service that makes it easy to set up, operate, and scale a [relational database](https://aws.amazon.com/dms/) in [the cloud](https://aws.amazon.com/what-is-cloud-computing/). It provides cost-efficient and resizable capacity, while managing time-consuming database administration tasks, freeing you up to focus on your applications and business.

Amazon RDS gives you access to the capabilities of a familiar MySQL, MariaDB, Oracle, SQL Server, or PostgreSQL database. This means that the code, applications, and tools you already use today with your existing databases should work seamlessly with Amazon RDS. Amazon RDS can automatically back up your database and keep your database software up to date with the latest version. You benefit from the flexibility of being able to easily scale the compute resources or storage capacity associated with your relational database instance. In addition, Amazon RDS makes it easy to use replication to enhance database availability, improve data durability, or scale beyond the capacity constraints of a single database instance for read-heavy database workloads. As with all Amazon Web Services, there are no up-front investments required, and you pay only for the resources you use.

Q: Which relational database engines does Amazon RDS support?

Amazon RDS supports Amazon Aurora, MySQL, MariaDB, Oracle, SQL Server, and PostgreSQL database engines.

Q: What does Amazon RDS manage on my behalf?

Amazon RDS manages the work involved in setting up a relational database: from provisioning the infrastructure capacity you request to installing the database software. Once your database is up and running, Amazon RDS automates common administrative tasks such as performing backups and patching the software that powers your database. With optional [Multi-AZ deployments](https://aws.amazon.com/rds/faqs/#36), Amazon RDS also manages synchronous data replication across Availability Zones with automatic failover.

Since Amazon RDS provides native database access, you interact with the relational database software as you normally would. This means you're still responsible for managing the database settings that are specific to your application. You'll need to build the relational schema that best fits your use case and are responsible for any performance tuning to optimize your database for your application’s workflow.

Q: When would I use Amazon RDS vs. Amazon EC2 Relational Database AMIs?

Amazon Web Services provides a number of database alternatives for developers. Amazon RDS enables you to run a fully featured relational database while offloading database administration. Using one of our many relational database AMIs on [Amazon EC2](https://aws.amazon.com/ec2/)allows you to manage your own relational database in the cloud. There are important differences between these alternatives that may make one more appropriate for your use case.

Q: What is a database instance (DB instance)?

You can think of a DB instance as a database environment in the cloud with the compute and storage resources you specify. You can create and delete DB instances, define/refine infrastructure attributes of your DB instance(s), and control access and security via the [AWS Management Console](https://console.aws.amazon.com/), [Amazon RDS APIs](http://docs.aws.amazon.com/AmazonRDS/latest/APIReference/Welcome.html), and [AWS Command Line Interface](http://docs.aws.amazon.com/cli/latest/reference/rds/index.html). You can run one or more DB instances, and each DB instance can support one or more databases or database schemas, depending on engine type. You can specify the parameters for your DB instance including DB engine and version, license model, instance type, storage type and amount, and master user credentials.

You also have the ability to change your DB instance’s backup retention policy, preferred backup window, and scheduled maintenance window.

Q: What should I do if my queries seem to be running slowly?

For production databases we encourage you to enable [Enhanced Monitoring](https://aws.amazon.com/rds/faqs/#enhanced-monitoring), which provides access to over 50 CPU, memory, file system, and disk I/O metrics. You can enable these features on a per-instance basis and you can choose the granularity (all the way down to 1 second). High levels of CPU utilization can reduce query performance and in this case you may want to consider scaling your DB instance class.

Q: Can I test my DB instance with a new version before upgrading?

Yes. You can do so by creating a DB snapshot of your existing DB instance, restoring from the DB snapshot to create a new DB instance, and then initiating a version upgrade for the new DB instance. You can then experiment safely on the upgraded copy of your DB instance before deciding whether or not to upgrade your original DB instance.

Q: Will my DB instance remain available during scaling?

The storage capacity allocated to your DB Instance can be increased while maintaining DB Instance availability. However, when you decide to scale the compute resources available to your DB instance up or down, your database will be temporarily unavailable while the DB instance class is modified. This period of unavailability typically lasts only a few minutes, and will occur during the maintenance window for your DB Instance, unless you specify that the modification should be applied immediately.

Q: What is the difference between automated backups and DB Snapshots?

Amazon RDS provides two different methods for backing up and restoring your DB instance(s) automated backups and database snapshots (DB Snapshots).  
The automated backup feature of Amazon RDS enables point-in-time recovery of your DB instance. When automated backups are turned on for your DB Instance, Amazon RDS automatically performs a full daily snapshot of your data (during your preferred backup window) and captures transaction logs (as updates to your DB Instance are made). When you initiate a point-in-time recovery, transaction logs are applied to the most appropriate daily backup in order to restore your DB instance to the specific time you requested. Amazon RDS retains backups of a DB Instance for a limited, user-specified period of time called the retention period, which by default is 7 days but can be set to up to 35 days. You can initiate a point-in-time restore and specify any second during your retention period, up to the Latest Restorable Time. You can use the [DescribeDBInstances](http://docs.aws.amazon.com/AmazonRDS/latest/APIReference/API_DescribeDBInstances.html) API to return the latest restorable time for you DB instance, which is typically within the last five minutes. Alternatively, you can find the Latest Restorable Time for a DB instance by selecting it in the [AWS Management Console](https://console.aws.amazon.com/) and looking in the “Description” tab in the lower panel of the Console.  
DB Snapshots are user-initiated and enable you to back up your DB instance in a known state as frequently as you wish, and then restore to that specific state at any time. DB Snapshots can be created with the [AWS Management Console](https://console.aws.amazon.com/), [CreateDBSnapshot API](http://docs.aws.amazon.com/AmazonRDS/latest/APIReference/API_CreateDBSnapshot.html), or [create-db-snapshot command](http://docs.aws.amazon.com/cli/latest/reference/rds/create-db-snapshot.html) and are kept until you explicitly delete them.  
The snapshots which Amazon RDS performs for enabling automated backups are available to you for copying (using the AWS console or the [copy-db-snapshot command](http://docs.aws.amazon.com/cli/latest/reference/rds/copy-db-snapshot.html)) or for the snapshot restore functionality. You can identify them using the "automated" Snapshot Type. In addition, you can identify the time at which the snapshot has been taken by viewing the "Snapshot Created Time" field. Alternatively, the identifier of the "automated" snapshots also contains the time (in UTC) at which the snapshot has been taken.  
Please note: When you perform a restore operation to a point in time or from a DB Snapshot, a new DB Instance is created with a new endpoint (the old DB Instance can be deleted if so desired). This is done to enable you to create multiple DB Instances from a specific DB Snapshot or point in time.

Q: What happens to my backups and DB snapshots if I delete my DB instance?

When you delete a DB instance, you can create a final DB snapshot upon deletion; if you do, you can use this DB snapshot to restore the deleted DB instance at a later date. Amazon RDS retains this final user-created DB snapshot along with all other manually created DB snapshots after the DB instance is deleted. Refer to the [pricing page](https://aws.amazon.com/rds/pricing/) for details of backup storage costs.

Automated backups are deleted when the DB instance is deleted. Only manually created DB Snapshots are retained after the DB Instance is deleted.

Q: What is Amazon Virtual Private Cloud (VPC) and how does it work with Amazon RDS?

Amazon VPC lets you create a virtual networking environment in a private, isolated section of the AWS cloud, where you can exercise complete control over aspects such as private IP address ranges, subnets, routing tables and network gateways. With Amazon VPC, you can define a virtual network topology and customize the network configuration to closely resemble a traditional IP network that you might operate in your own datacenter.

One way that you can take advantage of VPC is when you want to run a public-facing web application while still maintaining non-publicly accessible backend servers in a private subnet. You can create a public-facing subnet for your webservers that has access to the Internet, and place your backend RDS DB Instances in a private-facing subnet with no Internet access.

Q: How is using Amazon RDS inside a VPC different from using it on the EC2-Classic platform (non-VPC)?

If your AWS account was created before 2013-12-04, you may be able to run Amazon RDS in an Amazon Elastic Compute Cloud (EC2)-Classic environment. The basic functionality of Amazon RDS is the same regardless of whether EC2-Classic or EC2-VPC is used. Amazon RDS manages backups, software patching, automatic failure detection, read replicas and recovery whether your DB Instances are deployed inside or outside a VPC.

Q: What is a DB Subnet Group and why do I need one?

A DB Subnet Group is a collection of subnets that you may want to designate for your RDS DB Instances in a VPC. Each DB Subnet Group should have at least one subnet for every Availability Zone in a given Region. When creating a DB Instance in VPC, you will need to select a DB Subnet Group. Amazon RDS then uses that DB Subnet Group and your preferred Availability Zone to select a subnet and an IP address within that subnet. Amazon RDS creates and associates an Elastic Network Interface to your DB Instance with that IP address.

Please note that, we strongly recommend you use the DNS Name to connect to your DB Instance as the underlying IP address can change (e.g., during failover).

For Multi-AZ deployments, defining a subnet for all Availability Zones in a Region will allow Amazon RDS to create a new standby in another Availability Zone should the need arise. You need to do this even for Single-AZ deployments, just in case you want to convert them to Multi-AZ deployments at some point.

Q: How do I connect to an RDS DB Instance in VPC?

DB Instances deployed within a VPC can be accessed by EC2 Instances deployed in the same VPC. If these EC2 Instances are deployed in a public subnet with associated Elastic IPs, you can access the EC2 Instances via the internet.

DB Instances deployed within a VPC can be accessed from the Internet or from EC2 Instances outside the VPC via VPN or bastion hosts that you can launch in your public subnet, or using Amazon RDS's Publicly Accessible option:

* To use a bastion host, you will need to set up a public subnet with an EC2 instance that acts as a SSH Bastion. This public subnet must have an internet gateway and routing rules that allow traffic to be directed via the SSH host, which must then forward requests to the private IP address of your RDS DB instance.
* To use public connectivity, simply create your DB Instances with the Publicly Accessible option set to yes. With Publicly Accessible active, your DB Instances within a VPC will be fully accessible outside your VPC by default. This means you do not need to configure a VPN or bastion host to allow access to your instances.

Q: Can I move my existing DB instances outside VPC into my VPC?

If your DB instance is not in a VPC, you can use the AWS Management Console to easily move your DB instance into a VPC. You can also take a snapshot of your DB Instance outside VPC and restore it to VPC by specifying the DB Subnet Group you want to use. Alternatively, you can perform a “Restore to Point in Time” operation as well.

Q: What precautions should I take to ensure that my DB Instances in VPC are accessible by my application?

You are responsible for modifying routing tables and networking ACLs in your VPC to ensure that your DB instance is reachable from your client instances in the VPC.

For Multi-AZ deployments, after failover, your client EC2 instance and RDS DB Instance may be in different Availability Zones. You should configure your networking ACLs to ensure that cross-AZ communication is possible.

Q: Can I encrypt connections between my application and my DB Instance using SSL/TLS?

Yes, this option is supported for all Amazon RDS engines.  
Amazon RDS generates an SSL/TLS certificate for each DB Instance. Once an encrypted connection is established, data transferred between the DB Instance and your application will be encrypted during transfer.  
While SSL offers security benefits, be aware that SSL/TLS encryption is a compute-intensive operation and will increase the latency of your database connection. SSL/TLS support within Amazon RDS is for encrypting the connection between your application and your DB Instance; it should not be relied on for authenticating the DB Instance itself.

Q: Can I encrypt data at rest on my Amazon RDS databases?

Amazon RDS supports encryption at rest for all database engines, using keys you manage using [AWS Key Management Service (KMS)](https://aws.amazon.com/kms/). On a database instance running with Amazon RDS encryption, data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas, and snapshots. Encryption and decryption are handled transparently.

You can also add encryption to a previously unencrypted DB instance or DB cluster by creating a DB snapshot and then creating a copy of that snapshot and specifying a KMS encryption key. You can then restore an encrypted DB instance or DB cluster from the encrypted snapshot.

Q: I wish to perform security analysis or operational troubleshooting on my RDS deployment. Can I get a history of all RDS API calls made on my account?

Yes. AWS CloudTrail is a web service that records AWS API calls for your account and delivers log files to you. The AWS API call history produced by CloudTrail enables security analysis, resource change tracking, and compliance auditing.

Q: What are DB Parameter groups? How are they helpful?

A database parameter group (DB Parameter Group) acts as a “container” for engine configuration values that can be applied to one or more DB Instances. If you create a DB Instance without specifying a DB Parameter Group, a default DB Parameter Group is used. This default group contains engine defaults and Amazon RDS system defaults optimized for the DB Instance you are running. However, if you want your DB Instance to run with your custom-specified engine configuration values, you can simply create a new DB Parameter Group, modify the desired parameters, and modify the DB Instance to use the new DB Parameter Group. Once associated, all DB Instances that use a particular DB Parameter Group get all the parameter updates to that DB Parameter Group.

Q: How can I monitor the configuration of my Amazon RDS resources?

You can use [AWS Config](https://aws.amazon.com/config/) to continuously record configurations changes to Amazon RDS DB Instances, DB Subnet Groups, DB Snapshots, DB Security Groups, and Event Subscriptions and receive notification of changes through [Amazon Simple Notification Service (SNS)](https://aws.amazon.com/sns/). You can also create AWS Config Rules to evaluate whether these RDS resources have the desired configurations.

Q: What does it mean to run a DB instance as a Multi-AZ deployment?

When you create or modify your DB instance to run as a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous “standby” replica in a different Availability Zone. Updates to your DB Instance are synchronously replicated across Availability Zones to the standby in order to keep both in sync and protect your latest database updates against DB instance failure. During certain types of planned maintenance, or in the unlikely event of DB instance failure or Availability Zone failure, Amazon RDS will automatically fail over to the standby so that you can resume database writes and reads as soon as the standby is promoted. Since the name record for your DB instance remains the same, your application can resume database operation without the need for manual administrative intervention.

Q: What do “primary” and “standby” mean in the context of a Multi-AZ deployment?

When you run a DB instance as a Multi-AZ deployment, the “primary” serves database writes and reads. In addition, Amazon RDS provisions and maintains a “standby” behind the scenes, which is an up-to-date replica of the primary. The standby is “promoted” in failover scenarios. After failover, the standby becomes the primary and accepts your database operations. You do not interact directly with the standby (e.g. for read operations) at any point prior to promotion.

Q: What are the benefits of a Multi-AZ deployment?

The chief benefits of running your DB instance as a Multi-AZ deployment are enhanced database durability and availability. The increased availability and fault tolerance offered by Multi-AZ deployments make them a natural fit for production environments.  
Running your DB instance as a Multi-AZ deployment safeguards your data in the unlikely event of a DB instance component failure or loss of availability in one Availability Zone. For example, if a storage volume on your primary fails, Amazon RDS automatically initiates a failover to the standby, where all of your database updates are intact. This provides additional data durability relative to standard deployments in a single AZ, where a user-initiated restore operation would be required and updates that occurred after the latest restorable time (typically within the last five minutes) would not be available.  
You also benefit from enhanced database availability when running your DB instance as a Multi-AZ deployment. If an Availability Zone failure or DB instance failure occurs, your availability impact is limited to the time automatic failover takes to complete. The availability benefits of Multi-AZ also extend to planned maintenance. For example, with automated backups, I/O activity is no longer suspended on your primary during your preferred backup window, since backups are taken from the standby. In the case of patching or DB instance class scaling, these operations occur first on the standby, prior to automatic fail over. As a result, your availability impact is limited to the time required for automatic failover to complete.  
Another implied benefit of running your DB instance as a Multi-AZ deployment is that DB instance failover is automatic and requires no administration.

Q: When running my DB instance as a Multi-AZ deployment, can I use the standby for read or write operations?

No, a Multi-AZ standby cannot serve read requests. Multi-AZ deployments are designed to provide enhanced database availability and durability, rather than read scaling benefits. As such, the feature uses synchronous replication between primary and standby. Our implementation makes sure the primary and the standby are constantly in sync, but precludes using the standby for read or write operations. If you are interested in a read scaling solution, [Read Replicas](https://aws.amazon.com/rds/faqs/#read-replicas) is an option.

Q: What happens when I convert my RDS instance from Single-AZ to Multi-AZ?

For the RDS for MySQL, MariaDB, PostgreSQL and Oracle database engines, when you elect to convert your RDS instance from Single-AZ to Multi-AZ, the following happens:

* A snapshot of your primary instance is taken
* A new standby instance is created in a different Availability Zone, from the snapshot
* Synchronous replication is configured between primary and standby instances

As such, there should be no downtime incurred when an instance is converted from Single-AZ to Multi-AZ. However, you may see increased latency while the data on the standby is caught up to match to the primary. You may observe elevated latencies relative to Single-AZ as a result of the synchronous data replication performed on your behalf.

Q: What happens during Multi-AZ failover and how long does it take?

Failover is automatically handled by Amazon RDS so that you can resume database operations as quickly as possible without administrative intervention. When failing over, Amazon RDS simply flips the canonical name record (CNAME) for your DB instance to point at the standby, which is in turn promoted to become the new primary. We encourage you to follow best practices and implement database connection retry at the application layer.

Failovers, as defined by the interval between the detection of the failure on the primary and the resumption of transactions on the standby, typically complete within one to two minutes. Failover time can also be affected by whether large uncommitted transactions must be recovered; the use of adequately large instance types is recommended with Multi-AZ for best results. AWS also recommends the use of Provisioned IOPS with Multi-AZ instances, for fast, predictable, and consistent throughput performance.

Q: After failover, my primary is now located in a different Availability Zone than my other AWS resources (e.g. EC2 instances). Should I be concerned about latency?

Availability Zones are engineered to provide low latency network connectivity to other Availability Zones in the same Region. In addition, you may want to consider architecting your application and other AWS resources with redundancy across multiple Availability Zones so your application will be resilient in the event of an Availability Zone failure.

Q: How do DB Snapshots and automated backups work with my Multi-AZ deployment?

You interact with automated backup and DB Snapshot functionality in the same way whether you are running a standard deployment in a Single-AZ or Multi-AZ deployment. If you are running a Multi-AZ deployment, automated backups and DB Snapshots are simply taken from the standby to avoid I/O suspension on the primary. Please note that you may experience increased I/O latency (typically lasting a few minutes) during backups for both Single-AZ and Multi-AZ deployments.

Initiating a restore operation (point-in-time restore or restore from DB Snapshot) also works the same with Multi-AZ deployments as standard, Single-AZ deployments. New DB instance deployments can be created with either the RestoreDBInstanceFromSnapshot or RestoreDBInstanceToPointInTime APIs. These new DB instance deployments can be either standard or Multi-AZ, regardless of whether the source backup was initiated on a standard or Multi-AZ deployment.

**Read replica**

Q: What does it mean to run a DB Instance as a read replica?

Read replicas make it easy to take advantage of supported engines' built-in replication functionality to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. You can create a read replica with a few clicks in the AWS Management Console or using the CreateDBInstanceReadReplica API. Once the read replica is created, database updates on the source DB instance will be replicated using a supported engine's native, asynchronous replication. You can create multiple read replicas for a given source DB Instance and distribute your application’s read traffic amongst them.

Since read replicas use supported engines' built-in replication, they are subject to its strengths and limitations. In particular, updates are applied to your read replica(s) after they occur on the source DB instance, and replication lag can vary significantly. Read replicas can be associated with Multi-AZ deployments to gain read scaling benefits in addition to the enhanced database write availability and data durability provided by [Multi-AZ deployments](https://aws.amazon.com/rds/faqs/#36).

Q: When would I want to consider using an Amazon RDS read replica?

There are a variety of scenarios where deploying one or more read replicas for a given source DB instance may make sense. Common reasons for deploying a read replica include:

Scaling beyond the compute or I/O capacity of a single DB instance for read-heavy database workloads. This excess read traffic can be directed to one or more read replicas.

* Serving read traffic while the source DB instance is unavailable. If your source DB Instance cannot take I/O requests (e.g. due to I/O suspension for backups or scheduled maintenance), you can direct read traffic to your read replica(s). For this use case, keep in mind that the data on the read replica may be “stale” since the source DB Instance is unavailable.
* Business reporting or data warehousing scenarios; you may want business reporting queries to run against a read replica, rather than your primary, production DB Instance.
* You may use a read replica for disaster recovery of the source DB instance, either in the same AWS Region or in another Region.

Q: Do I need to enable automatic backups on my DB instance before I can create read replicas?

Yes. Enable automatic backups on your source DB Instance before adding read replicas, by setting the backup retention period to a value other than 0. Backups must remain enabled for read replicas to work.

Q: Which versions of database engines support Amazon RDS read replicas?

*Amazon Aurora:* All DB clusters.

*Amazon RDS for MySQL:* All DB instances support creation of read replicas. Automatic backups must be and remain enabled on the source DB instance for read replica operations. Automatic backups on the replica are supported only for Amazon RDS read replicas running MySQL 5.6 and later, not 5.5.

*Amazon RDS for PostgreSQL:* DB instances with PostgreSQL version 9.3.5 or newer support creation of read replicas. Existing PostgreSQL instances prior to version 9.3.5 need to be upgraded to PostgreSQL version 9.3.5 to take advantage of Amazon RDS read replicas.

*Amazon RDS for MariaDB:* All DB instances support creation of read replicas. Automatic backups must be and remain enabled on the source DB Instance for read replica operations.

*Amazon RDS for Oracle:* Supported for Oracle version 12.1.0.2.v12 and higher and for all 12.2 versions using the Bring Your Own License model with Oracle Database Enterprise Edition and licensed for the Active Data Guard Option.

Q: How do I connect to my read replica(s)?

You can connect to a read replica just as you would connect to a standard DB instance, using the DescribeDBInstance API or AWS Management Console to retrieve the endpoint(s) for you read replica(s). If you have multiple read replicas, it is up to your application to determine how read traffic will be distributed amongst them.

Q: How many read replicas can I create for a given source DB instance?

Amazon RDS for MySQL, MariaDB, PostgreSQL, and Oracle allow you to create up to 5 read replicas for a given source DB instance.

Q: Can I create a read replica in an AWS Region different from that of the source DB instance?

Yes, Amazon RDS supports cross-region read replicas. The amount of time between when data is written to the source DB instance and when it is available in the read replica will depend on the network latency between the two regions.

Q: Can I use a read replica to enhance database write availability or protect the data on my source DB instance against failure scenarios?

If you are looking to use replication to increase database write availability and protect recent database updates against various failure conditions, we recommend you run your DB instance as a Multi-AZ deployment. With Amazon RDS Read Replicas, which employ supported engines' native, asynchronous replication, database writes occur on a read replica after they have already occurred on the source DB instance, and this replication “lag” can vary significantly. In contrast, the replication used by Multi-AZ deployments is synchronous, meaning that all database writes are concurrent on the primary and standby. This protects your latest database updates, since they should be available on the standby in the event failover is required. In addition, with Multi-AZ deployments replication is fully managed. Amazon RDS automatically monitors for DB instance failure conditions or Availability Zone failure and initiates automatic failover to the standby

Q: Can I create a read replica with a Multi-AZ DB instance deployment as its source?

Yes. Since Multi-AZ DB instances address a different need than read replicas, it makes sense to use the two in conjunction for production deployments and to associate a read replica with a Multi-AZ DB Instance deployment. The “source” Multi AZ-DB instance provides you with enhanced write availability and data durability, and the associated read replica would improve read traffic scalability.

Q: If my read replica(s) use a Multi-AZ DB instance deployment as a source, what happens if Multi-AZ failover occurs?

In the event of Multi-AZ failover, any associated and available read replicas will automatically resume replication once failover has completed (acquiring updates from the newly promoted primary).

Q: Can I create a read replica of another read replica?

*Amazon Aurora, Amazon RDS for MySQL and MariaDB:* You can create a second-tier read replica from an existing first-tier read replica. By creating a second-tier read replica, you may be able to move some of the replication load from the master database instance to a first-tier Read Replica. Please note that a second-tier Read Replica may lag further behind the master because of additional replication latency introduced as transactions are replicated from the master to the first tier replica and then to the second-tier replica.

*Amazon RDS for PostgreSQL, Amazon RDS for Oracle:* Read Replicas of Read Replicas are not currently supported.

Q: Will my read replica be kept up-to-date with its source DB instance?

Updates to a source DB instance will automatically be replicated to any associated read replicas. However, with supported engines' asynchronous replication technology, a read replica can fall behind its source DB instance for a variety of reasons. Typical reasons include:

* Write I/O volume to the source DB instance exceeds the rate at which changes can be applied to the read replica (this problem is particularly likely to arise if the compute capacity of a read replica is less than the source DB Instance)
* Complex or long-running transactions to the source DB Instance hold up replication to the read replica
* Network partitions or latency between the source DB instance and a read replica

Read Replicas are subject to the strengths and weaknesses of supported engines' native replication. If you are using Read Replicas, you should be aware of the potential for lag between a Read Replica and its source DB Instance, or “inconsistency”.

Q: I scaled the compute and/or storage capacity of my source DB instance. Should I scale the resources for associated read replica(s) as well?

For replication to work effectively, we recommend that read replicas have as much or more compute and storage resources as their respective source DB instances. Otherwise replication lag is likely to increase or your read replica may run out of space to store replicated updates.

Q: What is Enhanced Monitoring for RDS?

Enhanced Monitoring for RDS gives you deeper visibility into the health of your RDS instances. Just turn on the “Enhanced Monitoring” option for your RDS DB Instance and set a granularity and Enhanced Monitoring will collect vital operating system metrics and process information, at the defined granularity.

For an even deeper level of diagnostics and visualization of your database load, and a longer data retention period, you can try [Performance Insights](https://aws.amazon.com/rds/performance-insights/).

Q: What information can I view on the RDS dashboard?

You can view all the system metrics and process information for your RDS DB Instances in a graphical format on the console. You can manage which metrics you want to monitor for each instance and customize the dashboard according to your requirements.

Q: When should I use CloudWatch instead of the RDS console dashboard?

You should use CloudWatch if you want to view historical data beyond what is available on the RDS console dashboard. You can monitor your RDS instances in CloudWatch to diagnose the health of your entire AWS stack in a single location. Currently, CloudWatch supports granularities of up to 1 minute and the values will be averaged out for granularities less than that.

**Amazon DynamoDB**

Q: What is Amazon DynamoDB?

DynamoDB is a fast and flexible nonrelational database service for any scale. DynamoDB enables customers to offload the administrative burdens of operating and scaling distributed databases to AWS so that they don’t have to worry about hardware provisioning, setup and configuration, throughput capacity planning, replication, software patching, or cluster scaling.

Q: What does DynamoDB manage on my behalf?

DynamoDB takes away one of the main stumbling blocks of scaling databases: the management of database software and the provisioning of the hardware needed to run it. You can deploy a nonrelational database in a matter of minutes. DynamoDB automatically scales throughput capacity to meet workload demands, and partitions and repartitions your data as your table size grows. Also, DynamoDB synchronously replicates data across three facilities in an AWS Region, giving you high availability and data durability.

Q: What is the consistency model of DynamoDB?

When reading data from DynamoDB, users can specify whether they want the read to be eventually consistent or strongly consistent:

* Eventually consistent reads (the default) – The eventual consistency option maximizes your read throughput. However, an eventually consistent read might not reflect the results of a recently completed write. All copies of data usually reach consistency within a second. Repeating a read after a short time should return the updated data.
* Strongly consistent reads — In addition to eventual consistency, DynamoDB also gives you the flexibility and control to request a strongly consistent read if your application, or an element of your application, requires it. A strongly consistent read returns a result that reflects all writes that received a successful response before the read.

Q: What kind of query functionality does DynamoDB support?

DynamoDB supports GET/PUT operations by using a user-defined primary key. The primary key is the only required attribute for items in a table. You specify the primary key when you create a table, and it uniquely identifies each item. DynamoDB also provides flexible querying by letting you query on nonprimary key attributes using [global secondary indexes](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GSI.html) and [local secondary indexes](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/LSI.html).

A primary key can be either a [single-attribute partition key or a composite partition-sort key](https://aws.amazon.com/blogs/database/choosing-the-right-dynamodb-partition-key/). A single-attribute partition key could be, for example, UserID. Such a single attribute partition key would allow you to quickly read and write data for an item associated with a given user ID.

DynamoDB indexes a composite partition-sort key as a partition key element and a sort key element. This multipart key maintains a hierarchy between the first and second element values. For example, a composite partition-sort key could be a combination of UserID (partition) and Timestamp (sort). Holding the partition key element constant, you can search across the sort key element to retrieve items. Such searching would allow you to use the [Query API](https://docs.aws.amazon.com/amazondynamodb/latest/APIReference/API_Query.html) to, for example, retrieve all items for a single UserID across a range of time stamps.

Q: What is In-Memory Acceleration with DAX?

Amazon DynamoDB is designed for scale and performance. In most cases, the DynamoDB response times can be measured in single-digit milliseconds. However, there are certain use cases that require response times in microseconds. For these use cases, *DynamoDB Accelerator (DAX)* delivers fast response times for accessing eventually consistent data.

DAX is a DynamoDB-compatible caching service that enables you to benefit from fast in-memory performance for demanding applications. DAX addresses three core scenarios:

1. As an in-memory cache, DAX reduces the response times of eventually-consistent read workloads by an order of magnitude, from single-digit milliseconds to microseconds.
2. DAX reduces operational and application complexity by providing a managed service that is API-compatible with Amazon DynamoDB, and thus requires only minimal functional changes to use with an existing application.
3. For read-heavy or bursty workloads, DAX provides increased throughput and potential operational cost savings by reducing the need to over-provision read capacity units. This is especially beneficial for applications that require repeated reads for individual keys.

Q: What are ideal use-cases for DAX?

DAX provides access to eventually consistent data from DynamoDB tables, with microsecond latency. A multi-AZ DAX cluster can serve millions of requests per second.

DAX is ideal for:

* Applications that require the fastest possible response time for reads. Some examples include real-time bidding, social gaming, and trading applications. DAX delivers fast, in-memory read performance for these use cases.
* Applications that read a small number of items more frequently than others. For example, consider an e-commerce system that has a one-day sale on a popular product. During the sale, demand for that product (and its data in DynamoDB) would sharply increase, compared to all of the other products. To mitigate the impacts of a "hot" key and a non-uniform data distribution, you could offload the read activity to a DAX cache until the one-day sale is over.
* Applications that are read-intensive, but are also cost-sensitive. With DynamoDB, you provision the number of reads per second that your application requires. If read activity increases, you can increase your tables' provisioned read throughput (at an additional cost). Alternatively, you can offload the activity from your application to a DAX cluster, and reduce the amount of read capacity units you'd need to purchase otherwise.
* Applications that require repeated reads against a large set of data. Such an application could potentially divert database resources from other applications. For example, a long-running analysis of regional weather data could temporarily consume all of the read capacity in a DynamoDB table, which would negatively impact other applications that need to access the same data. With DAX, the weather analysis could be performed against cached data instead.

DAX is *not* ideal for:

* Applications that require strongly consistent reads (or cannot tolerate eventually consistent reads).
* Applications that do not require microsecond response times for reads, or that do not need to offload repeated read activity from underlying tables.
* Applications that are write-intensive, or that do not perform much read activity.
* Applications that are already using a different caching solution with DynamoDB, and are using their own client-side logic for working with that caching solution.

**Amazon ElastiCache**

Q: What is Amazon ElastiCache?

Amazon ElastiCache is a web service that makes it easy to deploy and run Memcached or Redis protocol-compliant server nodes in [the cloud](https://aws.amazon.com/what-is-cloud-computing/). Amazon ElastiCache improves the performance of web applications by allowing you to retrieve information from a fast, managed, in-memory system, instead of relying entirely on slower disk-based databases. The service simplifies and offloads the management, monitoring and operation of in-memory environments, enabling your engineering resources to focus on developing applications. Using Amazon ElastiCache, you can not only improve load and response times to user actions and queries, but also reduce the cost associated with scaling web applications.

Amazon ElastiCache automates common administrative tasks required to operate a distributed in-memory key-value environment. Using Amazon ElastiCache, you can add a caching or in-memory layer to your application architecture in a matter of minutes via a few clicks of the AWS Management Console. Once a cluster is provisioned, Amazon ElastiCache automatically detects and replaces failed nodes, providing a resilient system that mitigates the risk of overloaded databases, which slow website and application load times. Through integration with Amazon CloudWatch monitoring, Amazon ElastiCache provides enhanced visibility into key performance metrics associated with your nodes. Amazon ElastiCache is protocol-compliant with Memcached and Redis, so code, applications, and popular tools that you use today with your existing Memcached or Redis environments will work seamlessly with the service. With the support for clustered configuration in Amazon ElastiCache, you get the benefits of fast, scalable and easy to use managed service that can meet the needs of your most demanding applications. As with all Amazon Web Services, there are no up-front investments required, and you pay only for the resources you use.

Q: What is in-memory caching and how does it help my applications?

The in-memory caching provided by Amazon ElastiCache can be used to significantly improve latency and throughput for many read-heavy application workloads (such as social networking, gaming, media sharing and Q&A portals) or compute-intensive workloads (such as a recommendation engine). In-memory caching improves application performance by storing critical pieces of data in memory for low-latency access. Cached information may include the results of I/O-intensive database queries or the results of computationally-intensive calculations.

Q: Can I use Amazon ElastiCache for use cases other than caching?

A: Yes. ElastiCache for Redis can be used as a primary in-memory key-value data store, providing fast, sub millisecond data performance, high availability and scalability up to 250 nodes and 250 shards, giving you up to 170.6 TB of in-memory data. Other use cases, such as leaderboards, rate limiting, queues, and chat. Amazon ElastiCache for Redis offers purpose-built in-memory data structures and operators to manage real-time geospatial data at scale and speed. Amazon ElastiCache for Redis gives you a fast in-memory data store to build and deploy machine learning models quickly. Amazon ElastiCache for Redis offers a fast, in-memory data store to power live streaming use cases. ElastiCache for Redis can be used to store metadata for user profile and viewing history, authentication information/tokens for millions of users, and manifest files to enable CDNs to stream videos to millions of mobile and desktop users at a time. Amazon ElastiCache for Redis offers List data structure making it easy to implement a lightweight, persistent queue. Lists offer atomic operations as well as blocking capabilities, making them suitable for a variety of applications that require a reliable message broker or a circular list. Use Amazon ElastiCache for Redis with streaming solutions such as Apache Kafka and Amazon Kinesis as an in-memory data store to ingest, process, and analyze real-time data with sub-millisecond latency. ElastiCache is an ideal choice for real-time analytics uses cases such as social media, ad targeting, personalization, [IoT and time-series data analytics](https://aws.amazon.com/blogs/database/managing-iot-and-time-series-data-with-amazon-elasticache-for-redis/). Amazon ElastiCache for Redis is highly suited as a session store to manage session information such as user authentication tokens, session state, and more. Simply use ElastiCache for Redis as a fast key-value store with appropriate TTL on session keys to manage your session information. Session management is commonly required for online applications, including games, e-commerce websites, and social media platforms.

Q: Can I use Amazon ElastiCache through AWS CloudFormation?

AWS CloudFormation simplifies provisioning and management by providing AWS CloudFormation templates for quick and reliable provisioning of the services or applications. AWS CloudFormation provides comprehensive support for Amazon ElastiCache by providing templates to create cluster (both MemCached and Redis) and Replication Groups.

Q: How do I control access to Amazon ElastiCache?

When not using [VPC](http://docs.amazonwebservices.com/AmazonVPC/latest/UserGuide/VPC_Introduction.html), Amazon ElastiCache allows you to control access to your clusters through Cache Security Groups. A Security Group acts like a firewall, controlling network access to your cluster. By default, network access is turned off to your clusters. If you want your applications to access your cluster, you must explicitly enable access from hosts in specific [EC2 security groups](http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/using-network-security.html). This process is called ingress.

To allow network access to your cluster, create a Security Group and link the desired EC2 security groups (which in turn specify the EC2 instances allowed) to it. The Security Group can be associated with your cluster at the time of creation, or using the "Modify" option on the [AWS Management Console](https://console.aws.amazon.com/elasticache/).

Q: Can programs running on servers in my own data center access Amazon ElastiCache?

Yes. You can access an Amazon ElastiCache cluster from an application running in your data center providing there is connectivity between your VPC and the data center either through VPN or Direct Connect.

Q: Can programs running on EC2 instances in a VPC access Amazon ElastiCache?

Yes, EC2 instances in a VPC can access Amazon ElastiCache if the ElastiCache cluster was created within the VPC.

Q: What is Amazon Virtual Private Cloud (VPC) and why may I want to use with Amazon ElastiCache?

Amazon VPC lets you create a virtual networking environment in a private, isolated section of the Amazon Web Services (AWS) cloud, where you can exercise complete control over aspects such as private IP address ranges, subnets, routing tables and network gateways. With Amazon VPC, you can define a virtual network topology and customize the network configuration to closely resemble a traditional IP network that you might operate in your own datacenter.

One of the scenarios where you may want to use Amazon ElastiCache in a VPC is if you want to run a public-facing web application, while still maintaining non-publicly accessible backend servers in a private subnet. You can create a public-facing subnet for your webservers that has access to the Internet, and place your backend infrastructure in a private-facing subnet with no Internet access. Your backend infrastructure could include RDS DB Instances and an Amazon ElastiCache Cluster providing the in-memory layer.

**Memcached**

Q: What can I cache using Amazon ElastiCache for Memcached?

You can cache a variety of objects using the service, from the content in persistent data stores (such as Amazon RDS, DynamoDB, or self-managed databases hosted on EC2) to dynamically generated web pages (with Nginx for example), or transient session data that may not require a persistent backing store. You can also use it to implement high-frequency counters to deploy admission control in high volume web applications.

Q: Can I use Amazon ElastiCache for Memcached with an AWS persistent data store such as Amazon RDS or Amazon DynamoDB?

Yes, Amazon ElastiCache is an ideal front-end for data stores like Amazon RDS or Amazon DynamoDB, providing a high-performance middle tier for applications with extremely high request rates and/or low latency requirements.

Q: How does Amazon ElastiCache interact with other Amazon Web Services?

Amazon ElastiCache is ideally suited as a front-end for Amazon Web Services like Amazon RDS and Amazon DynamoDB, providing extremely low latency for high performance applications and offloading some of the request volume while these services provide long lasting data durability. The service can also be used to improve application performance in conjunction with Amazon EC2 and EMR.

**Redis**

Q: What is Amazon ElastiCache for Redis?

Amazon ElastiCache for Redis is a web service that makes it easy to deploy and run Redis protocol-compliant server nodes in the cloud. The service enables the management, monitoring and operation of a Redis node; creation, deletion and modification of the node can be carried out through the ElastiCache console, the command line interface or the web service APIs. Amazon ElastiCache for Redis supports Redis Master / Slave replication.

**Amazon Redshift**

Q: What is Amazon Redshift?

Amazon Redshift is a fast, fully managed data warehouse that makes it simple and cost-effective to analyze all your data using standard SQL and your existing Business Intelligence (BI) tools. It allows you to run complex analytic queries against petabytes of structured data, using sophisticated query optimization, columnar storage on high-performance local disks, and massively parallel query execution. Most results come back in seconds. With Redshift, you can start small for just $0.25 per hour with no commitments and scale out to petabytes of data for $1,000 per terabyte per year, less than a tenth the cost of traditional solutions. Amazon Redshift also includes Amazon Redshift Spectrum, allowing you to directly run SQL queries against exabytes of unstructured data in Amazon S3. No loading or transformation is required, and you can use open data formats, including Avro, CSV, Grok, Ion, JSON, ORC, Parquet, RCFile, RegexSerDe, SequenceFile, TextFile, and TSV. Redshift Spectrum automatically scales query compute capacity based on the data being retrieved, so queries against Amazon S3 run fast, regardless of data set size.

Traditional data warehouses require significant time and resource to administer, especially for large datasets. In addition, the financial cost associated with building, maintaining, and growing self-managed, on-premise data warehouses is very high. As your data grows, you have to constantly trade-off what data to load into your data warehouse and what data to archive in storage so you can manage costs, keep ETL complexity low, and deliver good performance. Amazon Redshift not only significantly lowers the cost and operational overhead of a data warehouse, but with Redshift Spectrum, also makes it easy to analyze large amounts of data in its native format without requiring you to load the data.

Amazon Redshift gives you fast querying capabilities over structured data using familiar SQL-based clients and business intelligence (BI) tools using standard ODBC and JDBC connections. Queries are distributed and parallelized across multiple physical resources. You can easily scale an Amazon Redshift data warehouse up or down with a few clicks in the [AWS Management Console](https://aws.amazon.com/console/) or with a single [API call](http://docs.aws.amazon.com/redshift/latest/APIReference/Welcome.html). Amazon Redshift automatically patches and backs up your data warehouse, storing the backups for a user-defined retention period. Amazon Redshift uses replication and continuous backups to enhance availability and improve data durability and can automatically recover from component and node failures. In addition, Amazon Redshift supports Amazon Virtual Private Cloud (Amazon VPC), SSL, AES-256 encryption and Hardware Security Modules (HSMs) to protect your data in transit and at rest.

Q: How does the performance of Amazon Redshift compare to most traditional databases for data warehousing and analytics?

Amazon Redshift uses a variety of innovations to achieve up to ten times higher performance than traditional databases for data warehousing and analytics workloads:

* *Columnar Data Storage:* Instead of storing data as a series of rows, Amazon Redshift organizes the data by column. Unlike row-based systems, which are ideal for transaction processing, column-based systems are ideal for data warehousing and analytics, where queries often involve aggregates performed over large data sets. Since only the columns involved in the queries are processed and columnar data is stored sequentially on the storage media, column-based systems require far fewer I/Os, greatly improving query performance.
* *Advanced Compression:* Columnar data stores can be compressed much more than row-based data stores because similar data is stored sequentially on disk. Amazon Redshift employs multiple compression techniques and can often achieve significant compression relative to traditional relational data stores. In addition, Amazon Redshift doesn't require indexes or materialized views and so uses less space than traditional relational database systems. When loading data into an empty table, Amazon Redshift automatically samples your data and selects the most appropriate compression scheme.
* *Massively Parallel Processing (MPP):* Amazon Redshift automatically distributes data and query load across all nodes. Amazon Redshift makes it easy to add nodes to your data warehouse and enables you to maintain fast query performance as your data warehouse grows.
* *Redshift Spectrum:* Redshift Spectrum enables you to run queries against exabytes of data in Amazon S3. There is no loading or ETL required. Even if you don’t store any of your data in Amazon Redshift, you can still use Redshift Spectrum to query datasets as large as an exabyte in Amazon S3. When you issue a query, it goes to the Amazon Redshift SQL endpoint, which generates the query plan. Amazon Redshift determines what data is local and what is in Amazon S3, generates a plan to minimize the amount of Amazon S3 data that needs to be read, requests Redshift Spectrum workers out of a shared resource pool to read and process data from Amazon S3, and pulls results back into your Amazon Redshift cluster for any remaining processing.

Q: When would I use Amazon Redshift vs. Amazon RDS?

Both Amazon Redshift and [Amazon RDS](https://aws.amazon.com/rds/) enable you to run traditional relational databases in the cloud while offloading database administration. Customers use Amazon RDS databases both for online-transaction processing (OLTP) and for reporting and analysis. Amazon Redshift harnesses the scale and resources of multiple nodes and uses a variety of optimizations to provide order of magnitude improvements over traditional databases for analytic and reporting workloads against very large data sets. Amazon Redshift provides an excellent scale-out option as your data and query complexity grows or if you want to prevent your reporting and analytic processing from interfering with the performance of your OLTP workload.

Q: When should I use Amazon Athena vs. Redshift Spectrum?

[Amazon Athena](https://aws.amazon.com/athena/) is the simplest way to give any employee the ability to run ad-hoc queries on data in Amazon S3. Athena is serverless, so there is no infrastructure to setup or manage, and you can start analyzing your data immediately.

If you have frequently accessed data, that needs to be stored in a consistent, highly structured format, then you should use a data warehouse like Amazon Redshift. This gives you the flexibility to store your structured, frequently accessed data in Amazon Redshift, and use Redshift Spectrum to extend your Amazon Redshift queries out to the entire universe of data in your Amazon S3 data lake. This gives you the freedom to store your data where you want, in the format you want, and have it available for processing when you need.

**Amazon Elastic Transcoder**

Q: What is Amazon Elastic Transcoder?

Amazon Elastic Transcoder is a highly scalable, easy to use and cost effective way for developers and businesses to convert (or “transcode”) video and audio files from their source format into versions that will playback on devices like smartphones, tablets and PCs.

Q: What can I do with Amazon Elastic Transcoder?

You can use Amazon Elastic Transcoder to convert video and audio files into supported output formats optimized for playback on desktops, mobile devices, tablets, and televisions. In addition to supporting a wide range of input and output formats, resolutions, bitrates, and frame rates, Amazon Elastic Transcoder also offers features for automatic video bit rate optimization, generation of thumbnails, overlay of visual watermarks, caption support, DRM packaging, progressive downloads, encryption and more.

Q: Why should I use Amazon Elastic Transcoder?

Amazon Elastic Transcoder manages all the complexity of running media transcoding in the AWS cloud. Amazon Elastic Transcoder enables you to focus on your content, such as the devices you want to support and the quality levels you want to provide, rather than managing the infrastructure and software needed for conversion. Amazon Elastic Transcoder scales to handle the largest encoding jobs. As with all Amazon Web Services, there are no up-front investments required, and you pay only for the resources that you use.

**Amazon Kinesis Video Streams**

Q: What is Amazon Kinesis Video Streams?

Amazon Kinesis Video Streams makes it easy to securely stream video from connected devices to AWS for analytics, machine learning (ML), and other processing. Kinesis Video Streams automatically provisions and elastically scales all the infrastructure needed to ingest streaming video data from millions of devices. It also durably stores, encrypts, and indexes video data in your streams, and allows you to access your data through easy-to-use APIs. Kinesis Video Streams enables you to quickly build computer vision and ML applications through integration with Amazon Rekognition Video and libraries for ML frameworks such as Apache MxNet, TensorFlow, and OpenCV.

Q: What is time-encoded data?

Time-encoded data is any data in which the records are in a time series, and each record is related to its previous and next records. Video is an example of time-encoded data, where each frame is related to the previous and next frames through spatial transformations. Other examples of time-encoded data include audio, RADAR, and LIDAR signals. Amazon Kinesis Video Streams is designed specifically for cost-effective, efficient ingestion, and storage of all kinds of time-encoded data for analytics and ML use cases.

Q: What are common use cases for Kinesis Video Streams?

Kinesis Video Streams is ideal for building computer vision-enabled ML applications that are becoming prevalent in a wide range of use cases such as the following:

*Smart Home*

With Kinesis Video Streams, you can easily stream video and audio from camera-equipped home devices such as baby monitors, webcams, and home surveillance systems to AWS. You can then use the streams to build a variety of smart home applications ranging from simple video playback to intelligent lighting, climate control systems, and security solutions.

*Smart City*

Many cities have installed large numbers of cameras at traffic lights, parking lots, shopping malls, and just about every public venue, capturing video 24/7. You can use Kinesis Video Streams to securely and cost-effectively ingest, store, and analyze this massive volume of video data to help solve traffic problems, help prevent crime, dispatch emergency responders, and much more.

*Industrial Automation*

You can use Kinesis Video Streams to collect a variety of time-encoded data such as RADAR and LIDAR signals, temperature profiles, and depth data from industrial equipment. You can then analyze the data using your favorite machine learning framework including Apache MxNet, TensorFlow, and OpenCV for industrial automation use cases like predictive maintenance. For example, you can predict the lifetime of a gasket or valve and schedule part replacement in advance, reducing downtime and defects in a manufacturing line.

Q: What does Amazon Kinesis Video Streams manage on my behalf?

Amazon Kinesis Video Streams is a fully managed video ingestion and storage service. It enables you to securely ingest, process, and store video at any scale for applications that power robots, smart cities, industrial automation, security monitoring, machine learning (ML), and more. Kinesis Video Streams also ingests other kinds of time-encoded data like audio, RADAR, and LIDAR signals. Kinesis Video Streams provides you SDKs to install on your devices to make it easy to securely stream video to AWS. Kinesis Video Streams automatically provisions and elastically scales all the infrastructure needed to ingest video streams from millions of devices. It also durably stores, encrypts, and indexes the video streams and provides easy-to-use APIs so that applications can access and retrieve indexed video fragments based on tags and timestamps. Kinesis Video Streams provides a library to integrate ML frameworks such as Apache MxNet, TensorFlow, and OpenCV with video streams to build machine learning applications. Kinesis Video Streams is integrated with Amazon Rekognition Video, enabling you to build computer vision applications that detect objects, events, and people.