Cloud Transformation

First Website

CraigKeenan.Info: Lesson and Lab Guide

Lesson Primer

Amazon Web Services (AWS)

Amazon Global Infrastructure

AWS Global Infrastructure

2016 - 14 Regions & 38 Availability Zones
2017 - 4 more Regions & 11 More Availability Zones

A Region is a geographical area

Each region consists of 2 (or more) Availability Zones

An Availability Zone (AZ) is simply a Data Center

Edge Locations are CDN End Points for CloudFront

There are many more Edge Locations then Regions
Currently there are over 66 Edge Locations

Amazon Web Services Overview

Simple Storage Service (S3)

Amazon Simple Storage Service (Amazon S3) is storage for the Internet. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere on the web.

CloudFront

Amazon CloudFront is a web service that speeds up distribution of your static and dynamic web content, for example, .html, .css, .php, image, and media files, to end users. CloudFront delivers your content through a worldwide network of edge locations.

Elastic Compute Cloud (EC2)

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable computing capacity—literally, servers in Amazon's data centers—that you use to build and host your software systems.

Relational Database Service (RDS)

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.

Simple Storage Services (S3)

S3 Buckets

To upload your data (photos, videos, documents etc.), you first create a bucket in one of the AWS Regions. You can then upload any number of objects to the bucket. In terms of implementation, buckets and objects are resources.

Amazon S3 bucket names are globally unique, regardless of the AWS Region in which you create the bucket. You specify the name at the time you create the bucket.

S3 Objects

Amazon S3 is a simple key, value store designed to store as many objects as you want. You store these objects in one or more buckets

Note that your Amazon S3 resources (for example buckets and objects) are private by default. You will need to explicitly grant permission for others to access these resources.

CloudFront

CloudFront Distribution

When you want to use CloudFront to distribute your content, you create a distribution and specify configuration settings such as:

- Your origin, which is the Amazon S3 bucket or HTTP server from which CloudFront gets the files that it distributes.
- Whether you want the files to be available to everyone or you want to restrict access to selected users.
- Whether you want CloudFront to prevent users in selected countries from accessing your content.
- Whether you want CloudFront to create access logs.

Two Types of Distributions

Web Distributions

You can use web distributions to serve the following content over HTTP or HTTPS:

- Static and dynamic download content, for example, .html, .css, .php, and image files, using HTTP or HTTPS.
- Multimedia content on demand using progressive download and Apple HTTP Live Streaming (HLS).
- A live event, such as a meeting, conference, or concert, in real time.

RTMP Distributions

- RTMP distributions stream media files using Adobe Media Server and the Adobe Real-Time Messaging Protocol (RTMP). An RTMP distribution must use an Amazon S3 bucket as the origin.

Elastic Compute Cloud (EC2)

Amazon EC2 Features

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

Features of Amazon EC2

Amazon EC2 provides the following features:

- Virtual computing environments, known as instances
- Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types

Amazon EC2 Key Pairs

Amazon EC2 uses public–key cryptography to encrypt and decrypt login information. Public–key cryptography uses a public key to encrypt a piece of data, such as a password, then the recipient uses the private key to decrypt the data. The **public** and **private keys** are known as a **key pair**.

To log in to your instance, you must create a key pair, specify the name of the key pair when you launch the instance, and provide the private key when you connect to the instance. Linux instances have no password, and you use a key pair to log in using SSH. With Windows instances, you use a key pair to obtain the administrator password and then log in using RDP.

Connecting to Your Instances

Amazon EC2 instances created from most Amazon Machine Images (AMIs) enable you to connect using Remote Desktop for Windows and SSH client for Linux (i.e. Putty, or Mac OS Terminal). Windows Remote Desktop uses the Remote Desktop Protocol (RDP) and Linux uses SSH. Both enable you to connect to and use your instance in the same way you use a computer sitting in front of you.

Using Your Instances

You can use your instance just as you would any computer you manager. Account owners are expected to perform System Administration tasks (i.e. Operating System and Application Updates, Security Hardening, User Administration, etc.) You have Administrator or Root privileges for your EC2 Instances.

Stopping, Starting and Terminating Your EC2 Instances

You can stop and restart your instance if it has an Amazon EBS volume as its root device. The instance retains its instance ID, but can.

When you stop an instance, we shut it down. We don't charge hourly usage for a stopped instance, or data transfer fees, but we do charge for the storage for any Amazon EBS volumes.

While the instance is stopped, you can treat its root volume like any other volume, and modify it (for example, repair file system problems or update software). You just detach the volume from the stopped instance, attach it to a running instance, make your changes, detach it from the running instance, and then reattach it to the stopped instance..

If you decide that you no longer need an instance, you can terminate it. As soon as the state of an instance changes to `shutting-down` or `terminated`, we stop charging for that instance.

Relational Database Service (RDS)

DB Instances

The basic building block of Amazon RDS is the DB instance. A DB instance is an isolated database environment in the cloud. A DB instance can contain multiple user-created databases, and you can access it by using the same tools and applications that you use with a stand-alone database instance. You can create and modify a DB instance by using the Amazon AWS command line interface, the Amazon RDS API, or the AWS Management Console.

Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines. Each DB engine has its own supported features, and each version of a DB engine may include specific features.

The computation and memory capacity of a DB instance is determined by its DB instance class. You can select the DB instance that best meets your needs. If your needs change over time, you can change DB instances

Rebooting a DB Instance

In some cases, if you modify a DB instance, change the DB parameter group associated with the instance, or change a static DB parameter in a parameter group the instances uses, you must reboot the instance for the changes to take effect.

Rebooting a DB instance restarts the database engine service. A reboot also applies to the DB instance any modifications to the associated DB parameter group that were pending. Rebooting a DB instance results in a momentary outage of the instance, during which the DB instance status is set to rebooting.

Before you begin

Lesson Perquisites

You must have completed all previous lesson(s) before starting on this course.
Understand that Amazon Free Tier Pricing, so don't leave services running unnecessarily.

You only Pay for What You Use

With Amazon Web Services, you only pay for what you use. Since we will be using Free Tier eligible services for most parts of the lecture, our costs should be minimal. Each lesson is design to have you create, modify, and then delete any resources we create. As a matter of best practice, delete any resources you have created when you are done with the lesson. If you intend to pause, and then resume hours or even days later… STOP! Go back and delete any resources you created during this lab to avoid unnecessary charges.

Note:
We will eventually build out a 'Production' like environment with the intention of running our Website. When necessary, we will identify through the procedures what resources to leave running. If not specifically stated to leave the resource, it should be deleted upon completion of the lesson.

Gaining Practical Experience!

Experience is an accomplishment achieved mostly through practice. It is a process where one must engage in repetitive actions or activities to allow the knowledge to be absorbed at a fixed rate. At the same time, no one person can master every technology or skill. We will be explore a number of technology disciplines that in the past required many years, perhaps, even an entire career to claim mastery. Fortunately, AWS has simplified the learning curve and time required to be proficient. The remaining challenge is that there are many services to learn.

These labs are intended to provide structure and guidance to your Cloud Transformation. However, there are limits to its scope. We can NOT provide in depth knowledge transfer at a granular level for all topics. If you find some subject particularly interesting, then pause and go explore it on your own. How much exploring you do is entirely up to you. We hope you spend a lifetime learning new things.

The expectation is that you will be provided all the materials you need to complete the labs. Don't stop with what has been provided. Do the labs again and again. Do them differently. Explore what works, and what doesn't work as expected.

Lab Overview

Throughout all lessons, we will be acting as if we are an individual who is looking to get started with Amazon Web Services Cloud Computing. Fortunately, we have a friend who is a Cloud Guru, and this friend has agreed to guide us through the process. Our goal is to build a web site for our product/service. It can be anything we want it to be. For teaching purposes, we'll call our experiment OurAmazonWebServices (**OAWS**).

Note: When you see an example using **oaws** or **ouraws** in the name, you must replace with a name unique to your environment.

We ready to lay the foundations for our website, and will have many of the core components required built by the end of this lab. We need to familiarize ourselves with the core services like S3, EC2, CloudFront and RDS. The goal of the lesson is to deploy a S3 Bucket, deploy both Windows and Linux EC2 Instances, deploy a CloudFront Distribution, and finally deploy a MySQL RDS DB Instance.

Again, we are working with a minimal budget so everything we do at this time will be in the Free Tier. Pay attention and make sure you select the correct Free Tier services.

First Website - Lab Overview

Simple Storage Services (S3)

- ☐ Create an S3 Bucket
- ☐ Uploading Objects into Amazon S3
- ☐ Editing Object Permission – Make Object Public
- ☐ Hosting a Static Website on Amazon S3
- ☐ Deleting an Object in Amazon S3
- ☐ Empty a Bucket in Amazon S3
- ☐ Delete a Bucket in Amazon S3

CloudFront

- ☐ Create an CloudFront Web Distribution
- ☐ Deleting a Distribution

Elastic Compute Cloud (EC2)

- ☐ Launch EC2 Instance
- ☐ Creating a Key Pair Using Amazon EC2
- ☐ Connecting to Windows EC2 Instance
- ☐ Install the AWS Command Line on Microsoft Windows
- ☐ Connecting to Your Linux Instance using SSH
- ☐ Install the AWS Command Line Interface on Linux
- ☐ Stop and Start Your EC2 Instances
- ☐ Terminate Your EC2 Instance

Relational Database Service (RDS)

- ☐ Creating a DB Instance Running the MySQL Database Engine
- ☐ Modifying a DB Instance Running the MySQL Database Engine
- ☐ Rebooting a DB Instance
- ☐ Deleting a DB Instance

Cleanup!

- ☐ Stop/Terminate all unneeded resources
- ☐ Review Environment Configuration

Lab – First Website!

Simple Storage Service (S3)

Create an S3 Bucket

URL Link:
http://docs.aws.amazon.com/AmazonS3/latest/gsg/CreatingABucket.html

Now that you've signed up for Amazon S3, you're ready to create a bucket using the AWS Management Console. Every object in Amazon S3 is stored in a bucket.

Note
You are not charged for creating a bucket; you are charged only for storing objects in the bucket and for transferring objects in and out of the bucket.

To create a bucket

1. Sign into the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3
2. Click **Create Bucket**.
3. In the **Create a Bucket** dialog box, in the **Bucket Name** box, enter a bucket name.
   (Ex. www.ouraws.com)
   The bucket name you choose must be unique across all existing bucket names in Amazon S3.
   Note After you create a bucket, you cannot change its name. In addition, the bucket name is visible in the URL that points to the objects stored in the bucket. Ensure that the bucket name you choose is appropriate.
4. In the **Region** box, select a region. For this exercise, select **Oregon** from the drop-down list.
   You can choose a region to optimize latency, minimize costs, or address regulatory requirements. Objects stored in a region never leave that region unless you explicitly transfer them to another region.
5. Click **Create**. When Amazon S3 successfully creates your bucket, the console displays your empty bucket in the **Buckets** panel.

You've created a bucket in Amazon S3.

Uploading Objects into Amazon S3

URL Link: http://docs.aws.amazon.com/AmazonS3/latest/UG/UploadingObjectsintoAmazonS3.html

When you upload a folder, Amazon S3 uploads all the files and subfolders from the specified folder to your bucket. It then assigns a key value that is a combination of the uploaded file name and the folder name.

To upload files and folders into Amazon S3

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the buckets list, click the name of bucket where you want to upload an object and then click **Upload**.
3. Click **Add Files**.
4. In the dialog box that appears, click the file or files that you want to upload, and then click **Open**.
   Note: Create a file named 'index.html'. Using a text edit put the following in the file:
   '<html>HTML Sample</html>'.

5. If you are ready to upload the object immediately, without providing further details about the object, click **Start Upload**.
6. Click **Start Upload**.
   You can watch the progress of the upload from within the Transfers panel.

   Tip
   To hide the **Transfer** panel, click **None**. To open it again, click **Transfers**.

   When objects upload successfully to Amazon S3, they appear in the Objects and Folders list.

Editing Object Permissions – Make Object Public

URL Link: http://docs.aws.amazon.com/AmazonS3/latest/UG/EditingPermissionsonanObject.html

This section explains how to use the console to edit AWS account permissions for an object.
Bucket and object permissions are completely independent; an object does not inherit the permissions from its bucket.
Hint: Click on Object link (step 4) both before and after making public. Observe results.

To make an object accessible by everyone

1. Right-click the object that you want to make accessible, and then click **Make Public**.
   (Ex. index.html)
2. The console prompts you to confirm this change. Click **OK**. When the change is complete, click the Close button in the **Transfers** panel.
3. Click **Permissions**. The newly added grantee appears in the display.
4. Click the link for the object to share in the object properties pane

Hosting a Static Website on Amazon S3

URL Link: http://docs.aws.amazon.com/AmazonS3/latest/dev/HostingWebsiteOnS3Setup.html

You can host a static website on Amazon S3. On a static website, individual web pages include static content.

> **Note** Amazon Web Services (AWS) has resources for hosting dynamic websites. To learn more about website hosting on AWS, go to Websites and Website Hosting.

To create a bucket and configure it as a website

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Create a bucket.
3. Open the bucket **Properties** panel, click **Static Website Hosting**, and do the following:
   1. Select the **Enable website hosting**.
   2. In the **Index Document** box, add the name of your index document. This name is typically *index.html*.
   3. Click **Save** to save the website configuration.
   4. Note down the Endpoint.
      This is the Amazon S3-provided website endpoint for your bucket. You will use this endpoint in the following steps to test your website.
      Hint: Click on Website Endpoint link. Observe results

Test your website

Enter the following URL in the browser, replacing example-bucket with the name of your bucket and website-region with the name of the region where you deployed your bucket.

http://example-bucket.s3-website-region.amazonaws.com

If your browser displays your index.html page, the website was successfully deployed.

Deleting an Object in Amazon S3

URL Link: http://docs.aws.amazon.com/AmazonS3/latest/UG/DeletinganObject.html

Because all objects in your Amazon S3 bucket incur storage costs, you should delete objects that you no longer need.

To delete an object

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the **Objects and Folders** list, right-click the object that you want to delete, and then click **Delete**.
3. When a confirmation message appears, click **OK**.

Empty a Bucket in Amazon S3

URL Link: http://docs.aws.amazon.com/AmazonS3/latest/dev/delete-or-empty-bucket.html#empty-bucket-console

The Amazon S3 console supports emptying your bucket provided that the bucket contains less than 100,000 objects.

To empty a bucket

1. In the Amazon S3 console, open the context (right-click) menu on the bucket and choose **Empty Bucket**.

Delete a Bucket in Amazon S3

URL Link: http://docs.aws.amazon.com/AmazonS3/latest/dev/delete-or-empty-bucket.html#empty-bucket-console

The Amazon S3 console supports deleting a bucket that may or may not be empty.

To delete a bucket

In the Amazon S3 console, open the context (right-click) menu on the bucket and choose **Delete Bucket**.

CloudFront

Create a CloudFront Web Distribution

URL Link:
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/GettingStarted.html#GettingStartedCreateDistribution

You only need to perform a few basic steps to start delivering your content using CloudFront. The first step is signing up. After that, you create a CloudFront distribution, and then use the CloudFront domain name to reference content in your web pages or applications.

To create a CloudFront Web Distribution

Step 1: Sign up for Amazon Web Services

Step 2: Upload your content to Amazon S3 and grant object permissions

To upload your content to Amazon S3 and grant read permission to everyone

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the Amazon S3 console, choose **Create Bucket**.
3. In the **Create Bucket** dialog, enter a bucket name.
   (Ex. www.ouraws.com)
4. Select a region for your bucket. By default, Amazon S3 creates buckets in the US East (N. Virginia) region.
5. Choose **Create**.
6. Select your bucket in the **Buckets** pane, and choose **Upload**.
7. On the **Upload - Select Files** page, choose **Add Files**, and choose the files that you want to upload.
8. Enable public read privileges for each object that you upload to your Amazon S3 bucket.
   a. Choose Set Details.
   b. On the Set Details page, choose Set Permissions.
   c. On the Set Permissions page, choose Make everything public.
9. **Choose Start Upload**.
   After the upload completes, you can navigate to this item by its URL. Use your Amazon S3 URL to verify that your content is publicly accessible.

Step 3. Create a CloudFront Web Distribution

To create a CloudFront web distribution

1. Open the CloudFront console at https://console.aws.amazon.com/cloudfront/.
2. Choose **Create Distribution**.
3. On the **Select a delivery method for your content** page, in the **Web** section, choose **Get Started**.
4. On the **Create Distribution** page, under Origin Settings, choose the Amazon S3 bucket that you created earlier for **Origin Domain Name**. For **Origin ID**, **Origin Path**, **Restrict Bucket Access**, and **Origin Custom Headers**, accept the default values.
5. Under **Default Cache Behavior Settings**, accept the default values, and CloudFront will:
6. Under **Distribution Settings**, enter the applicable values:

Price Class
Select the price class that corresponds with the maximum price that you want to pay for CloudFront service.

AWS WAF Web ACL
If you want to use AWS WAF to allow or block HTTP and HTTPS requests based on criteria that you specify, choose the web ACL to associate with this distribution.

Alternate Domain Names (CNAMEs) (Optional)
Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution.

SSL Certificate
Accept the default value, **Default CloudFront Certificate**.

Default Root Object (Optional)
The object that you want CloudFront to request from your origin

Logging (Optional)
If you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket.

Cookie Logging
In this example, we're using Amazon S3 as the origin for your objects, and Amazon S3 doesn't process cookies, so we recommend that you select **Off** for the value of **Cookie Logging**.

Comment (Optional)
Enter any comments that you want to save with the distribution.

Distribution State

Select **Enabled** if you want CloudFront to begin processing requests as soon as the distribution is created, or select **Disabled** if you do not want CloudFront to begin processing requests after the distribution is created.

7. Choose **Create Distribution**.
8. After CloudFront has created your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. If you chose to enable the distribution, it will then be ready to process requests. This should take less than 15 minutes. The domain name that CloudFront assigns to your distribution appears in the list of distributions. (It also appears on the **General** tab for a selected distribution.)

Deleting a Distribution

URL Link:
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/HowToDeleteDistribution.html

If you no longer want to use a distribution, use the following procedure to delete it using the CloudFront console.

Note
CloudFront lets you create a combined total of up to 100 web and RTMP distributions for an AWS account.

To Delete a CloudFront Distribution Using the CloudFront Console

1. Sign in to the AWS Management Console and open the CloudFront console at https://console.aws.amazon.com/cloudfront/.
2. In the right pane of the CloudFront console, find the distribution that you want to delete.
3. If the value of the **State** column is **Disabled**, skip to Step 7.

   If the value of **State** is **Enabled** and the value of **Status** is **Deployed**, continue with Step 4 to disable the distribution before deleting it.

   If the value of **State** is **Enabled** and the value of **Status** is **InProgress**, wait until **Status** changes to **Deployed**. Then continue with Step 4 to disable the distribution before deleting it.
4. In the right pane of the CloudFront console, check the check box for the distribution that you want to delete.
5. Click **Disabled** to disable the distribution, and click **Yes, Disable** to confirm. Then click **Close**.
6. The value of the **State** column immediately changes to **Disabled**. Wait until the value of the **Status** column changes to **Deployed**.
7. Check the check box for the distribution that you want to delete.
8. Click **Delete**, and click **Yes, Delete** to confirm. Then click **Close**.

Elastic Compute Cloud (EC2)

Launch EC2 Instance

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/launching-instance.html

An instance is a virtual server in the AWS cloud. You launch an instance from an Amazon Machine Image (AMI). The AMI provides the operating system, application server, and applications for your instance.

When you sign up for AWS, you can get started with Amazon EC2 for free using the AWS Free Tier. You can either leverage the free tier to launch and use a micro instance for free for 12 months.

To launch an instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar at the top of the screen, the current region is displayed. Select the region for the instance. This choice is important because some Amazon EC2 resources can be shared between regions, while others can't. Select the region that meets your needs.
3. From the Amazon EC2 console dashboard, choose **Launch Instance**.
4. On the Choose an **Amazon Machine Image (AMI)** page, choose an AMI as follows:
    a. Select the type of AMI to use in the left pane from **Quick Start**
    b. Choose an AMI that meets your needs, and then choose **Select**.
    (Ex. **Amazon Linux AMI 2016.09.1 (HVM), SSD Volume Type** - ami-0b33d91d – Free Tier eligible)
5. On the **Choose an Instance Type** page, select the hardware configuration and size of the instance to launch. Larger instance types have more CPU and memory.
    To remain eligible for the free tier, choose the t2.micro instance type.
    Choose **Next: Configure Instance Details**.
6. On the **Configure Instance Details** page, change the following settings as necessary (expand Advanced Details to see all the settings), and then choose **Next: Add Storage**:
7. On the **Add Storage** page, you can specify volumes to attach to the instance besides the volumes specified by the AMI (such as the root device volume). You can change the following options, then choose **Next: Add Tags** when you have finished:
8. On the **Add Tags** page, specify tags for the instance by providing key and value combinations. Choose **Add another tag** to add more than one tag to your resource. Choose **Next: Configure Security Group** when you are done.
    (Ex. oawsEC2)
9. On the **Configure Security Group** page, use a security group to define firewall rules for your instance. These rules specify which incoming network traffic is delivered to your instance. All other traffic is ignored. Select or create a security group as follows, and then choose **Review and Launch**.
    To select an existing security group:
        Choose Select an existing security group. Your security groups are displayed.
        Select a security group from the list.
    To create a new security group:
        - Choose Create a new security group. The wizard automatically defines the launch-wizard-x security group.
        - You can edit the name and description of the security group.
            (Ex. oawsPublic-SG)
        - The wizard automatically defines an inbound rule to allow to you connect to your instance over SSH (port 22) for Linux or RDP (port 3389) for Windows. Caution This rule enables all IP addresses (0.0.0.0/0) to access your instance over the specified port
        - You can add rules to suit your needs.
            To add a rule, choose Add Rule, select the protocol to open to network traffic, and then specify the source. Choose My IP from the Source list to let the wizard add your computer's public IP address.
10. On the **Review Instance Launch** page, check the details of your instance, and make any necessary changes by choosing the appropriate **Edit** link.
    When you are ready, choose **Launch**.
11. In the **Select an existing key pair or create a new key pair** dialog box, you can choose an existing key pair, or create a new one. For example, choose **Choose an existing key pair**, then select the key pair you created when getting set up.
    (Ex. oawsKP)
    To launch your instance, select the acknowledgment check box, then choose **Launch Instances**.
    Important If you choose the Proceed without key pair option, you won't be able to connect to the instance unless you choose an AMI that is configured to allow users another way to log in.

Creating a Key Pair Using Amazon EC2

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html

You can create a key pair using the Amazon EC2 console or the command line. After you create a key pair, you can specify it when you launch your instance.

To create your key pair using the Amazon EC2 console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **NETWORK & SECURITY**, choose **Key Pairs**.
   Tip The navigation pane is on the left side of the Amazon EC2 console.
3. Choose **Create Key Pair**.
4. Enter a name for the new key pair in the **Key pair name** field of the **Create Key Pair** dialog box, and then choose **Create**.
   (Ex. oawsKP)
5. The private key file is automatically downloaded by your browser. The base file name is the name you specified as the name of your key pair, and the file name extension is .pem. Save the private key file in a safe place.
   Important This is the only chance for you to save the private key file. You'll need to provide the name of your key pair when you launch an instance and the corresponding private key each time you connect to the instance.
6. If you will use an SSH client on a Mac or Linux computer to connect to your Linux instance, use the following command to set the permissions of your private key file so that only you can read it.
   $ chmod 400 oawsKP.pem

Connecting to your Windows EC2 Instance

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/connecting_to_windows_instance.html

Amazon EC2 instances created from most Windows Amazon Machine Images (AMIs) enable you to connect using Remote Desktop. Remote Desktop uses the Remote Desktop Protocol (RDP) and enables you to connect to and use your instance in the same way you use a computer sitting in front of you.

Prerequisites

- Launch a Windows EC2 Instance
  - (Ex. **Microsoft Windows Server 2012 R2 Base** - ami-abf616bd – Free Tier Eligible)
- Install an RDP client Your Windows computer includes an RDP client by default. You can check for an RDP client by typing mstsc at a Command Prompt window.
  Important
  Mac OS X users: If you are connecting to a Windows 2012 R2 instance, the Remote Desktop Connection client from the Microsoft website may not work. Use the Microsoft Remote Desktop app from the Apple App Store instead.
- Get the ID of the instance You can get the ID of your instance using the Amazon EC2 console (from the Instance ID column).
- Get the public DNS name of the instance You can get the public DNS for your instance using the Amazon EC2 console (check the Public DNS (IPv4) column; if this column is hidden, choose the Show/Hide icon and select Public DNS (IPv4)
- Get the public IP name of the instance You can get the public IP for your instance using the Amazon EC2 console (check the Public IP (IPv4) column; if this column is hidden, choose the Show/Hide icon and select Public IP (IPv4)

- Locate the private key You'll need the fully qualified path of the .pem file for the key pair that you specified when you launched the instance.
- Enable inbound RDP traffic from your IP address to your instance Ensure that the security group associated with your instance allows incoming RDP traffic from your IP address.
- For the best experience using Internet Explorer, run the latest version.

Connect to Your Windows Instance

To connect to a Windows instance, you must retrieve the initial administrator password and then specify this password when you connect to your instance using Remote Desktop.

The name of the administrator account depends on the language of the operating system. For example, for English, it's Administrator

The license for the Windows Server operating system (OS) allows two simultaneous remote connections for administrative purposes. The license for Windows Server is included in the price of your EC2 instance.

To connect to your Windows instance using an RDP client

1. In the Amazon EC2 console, select the instance, and then choose **Connect**.
2. In the **Connect To Your Instance** dialog box, choose **Get Password** (it will take a few minutes after the instance is launched before the password is available).
3. Choose **Browse** and navigate to the private key file you created when you launched the instance. Select the file and choose **Open** to copy the entire contents of the file into the **Contents** field.
4. Choose **Decrypt Password**. The console displays the default administrator password for the instance in the **Connect To Your Instance** dialog box, replacing the link to **Get Password** shown previously with the actual password.
5. Record the default administrator password, or copy it to the clipboard. You need this password to connect to the instance.
6. Choose **Download Remote Desktop File**. Your browser prompts you to either open or save the .rdp file. Either option is fine. When you have finished, you can choose **Close** to dismiss the **Connect To Your Instance** dialog box.
   - If you opened the .rdp file, you'll see the Remote Desktop Connection dialog box.
   - If you saved the .rdp file, navigate to your downloads directory, and open the .rdp file to display the dialog box.
     HINT If you are new to Remote Connectivity to a Windows Server, YouTube How to RDP a Windows Server. It's simple. Once you see the process you will be able to repeat.
     MAC OS X users, use Microsoft Remote Desktop app from the Apple App Store
7. You may get a warning that the publisher of the remote connection is unknown. If you are using **Remote Desktop Connection** from a Windows PC, choose **Connect** to connect to your instance. If you are using **Microsoft Remote Desktop** on a Mac, skip the next step.
8. When prompted, log in to the instance, using the administrator account for the operating system and the password that you recorded or copied previously. If your **Remote Desktop Connection** already has an administrator account set up, you might have to choose the **Use another account** option and enter the user name and password manually.

   Note

   Sometimes copying and pasting content can corrupt data. If you encounter a "Password Failed" error when you log in, try typing in the password manually.
9. Due to the nature of self-signed certificates, you may get a warning that the security certificate could not be authenticated. Simply choose Yes or Continue to continue if you trust the certificate.

After you connect, we recommend that you do the following:

- Change the administrator password from the default value. You change the password while logged on to the instance itself, just as you would on any other Windows Server.

- Create another user account with administrator privileges on the instance. Another account with administrator privileges is a safeguard if you forget the administrator password or have a problem with the administrator account. The user account must have permission to access the instance remotely. Open **System Properties**, choose **Remote**, and add the user to the **Remote Desktop Users** group.

Install the AWS Command Line Interface on Microsoft Windows

URL Link: http://docs.aws.amazon.com/cli/latest/userguide/awscli-install-windows.html

You can install the AWS CLI on Windows with a standalone installer.

MSI Installer
The AWS CLI is supported on Microsoft Windows XP or later. For Windows users, the MSI installation package offers a familiar and convenient way to install the AWS CLI without installing any other prerequisites.
When updates are released, you must repeat the installation process to get the latest version of the AWS CLI.

To install the AWS CLI using the MSI installer

1. Download the appropriate MSI installer.
   Download the AWS CLI MSI installer for Windows (64-bit)
   Download the AWS CLI MSI installer for Windows (32-bit)
   Note The MSI installer for the AWS CLI does not work with Windows Server 2008 (version 6.0.6002). Use pip to install with this version of Windows.
2. Run the downloaded MSI installer.
3. Follow the instructions that appear.

The CLI installs to C:\Program Files\Amazon\AWSCLI (64-bit) or C:\Program Files (x86)\Amazon\AWSCLI (32-bit) by default. To confirm the installation, use the aws --version command at a command prompt (open the START menu and search for "cmd" if you're not sure where the command prompt is installed).

```
aws --version
aws-cli/1.11.47 Python/2.7.9 Windows/2012Server botocore/1.5.10
```

If Windows is unable to find the executable, you may need to re-open the command prompt or add the installation directory to your PATH environment variable manually.

Updating an MSI Installation
The AWS CLI is updated regularly. Check out the Releases page on GitHub to see when the latest version was released. To update to the latest version, download and run the MSI installer again as detailed above.

Uninstalling
To uninstall the AWS CLI, open the Control Panel and select Programs and Features. Select the entry named AWS Command Line Interface and click Uninstall to launch the uninstaller. Confirm that you wish to uninstall the AWS CLI when prompted.
You can also launch the Programs and Features menu from the command line with the following command:
appwiz.cpl

Adding the AWS CLI Executable to your Command Line Path
After installing the AWS CLI, add the aws executable to your OS's PATH environment variable. With an MSI installation, this should happen automatically, but you may need to set it manually if the aws command is not working.
MSI installer (64-bit) – C:\Program Files\Amazon\AWSCLI
MSI installer (32-bit) – C:\Program Files (x86)\Amazon\AWSCLI

To modify your PATH variable (Windows)

1. Press the Windows key and type environment variables.
2. Choose Edit environment variables for your account.
3. Choose PATH and then choose Edit.
4. Add paths to the Variable value field, separated by semicolons. For example: C:\existing\path;C:\new\path
5. Choose OK twice to apply the new settings.
6. Close any running command prompts and re-open.

Connecting to Your Linux Instance Using SSH

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstancesLinux.html

After you launch your instance, you can connect to it and use it the way that you'd use a computer sitting in front of you.

Note

After you launch an instance, it can take a few minutes for the instance to be ready so that you can connect to it. Check that your instance has passed its status checks - you can view this information in the **Status Checks** column on the **Instances** page.

Prerequisites

- Launch a Windows EC2 Instance
  - (Ex. **Amazon Linux AMI 2016.09.1 (HVM), SSD Volume Type** - ami-0b33d91d – Free Tier eligible)
- Install an SSH client

  Your Linux computer most likely includes an SSH client by default. You can check for an SSH client by typing ssh at the command line.
- Install the AWS CLI Tools

  (Optional) If you're using a public AMI from a third party, you can use the command line tools to verify the fingerprint.
- Get the ID of the instance You can get the ID of your instance using the Amazon EC2 console (from the Instance ID column).
- Get the public DNS name of the instance You can get the public DNS for your instance using the Amazon EC2 console (check the Public DNS (IPv4) column; if this column is hidden, choose the Show/Hide icon and select Public DNS (IPv4)
- Get the public IP name of the instance You can get the public IP for your instance using the Amazon EC2 console (check the Public IP (IPv4) column; if this column is hidden, choose the Show/Hide icon and select Public IP (IPv4)
- Locate the private key You'll need the fully qualified path of the .pem file for the key pair that you specified when you launched the instance.
- Enable inbound RDP traffic from your IP address to your instance Ensure that the security group associated with your instance allows incoming RDP traffic from your IP address.

Connecting to Your Linux Instance

Use the following procedure to connect to your Linux instance using an SSH client.

To connect to your instance using SSH

1. (Optional) You can verify the RSA key fingerprint on your running instance by using one of the following commands on your local system (not on the instance). This is useful if you've launched your instance from a public AMI from a third party. Locate the SSH HOST KEY FINGERPRINTS section, and note the RSA fingerprint

(for example, 1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f) and compare it to the fingerprint of the instance.

get-console-output (AWS CLI)

aws ec2 get-console-output --instance-id instance_id

Note Ensure that the instance is in the running state, not the pending state. The SSH HOST KEY FINGERPRINTS section is only available after the first boot of the instance.

2. In a command-line shell, change directories to the location of the private key file that you created when you launched the instance.
3. Use the chmod command to make sure that your private key file isn't publicly viewable. Use the following command:

   chmod 400 /path/my-key-pair.pem
4. Use the ssh command to connect to the instance. You specify the private key (.pem) file and user_name@public_dns_name.

   For Amazon Linux, the user name is ec2-user.
   For RHEL, the user name is ec2-user or root.
   For Ubuntu, the user name is ubuntu or root.
   For Centos, the user name is centos.
   For Fedora, the user name is ec2-user.
   For SUSE, the user name is ec2-user or root. Otherwise, if ec2-user and root don't work, check with your AMI provider.

   ssh -i /path/my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-1.amazonaws.com

   HINT Use this command 'ssh ec2user@<IP> -i kp.pem'

   You see a response like the following.

```
The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com
(10.254.142.33)' can't be established.
RSA key fingerprint is
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
Are you sure you want to continue connecting (yes/no)?
```
5. (IPv6 only) Alternatively, you can connect to the instance using its IPv6 address. Specify the ssh command with the path to the private key (.pem) file and the appropriate user name.

   ssh -i /path/my-key-pair.pem ec2-user@2001:db8:1234:1a00:9691:9503:25ad:1761
6. (Optional) Verify that the fingerprint in the security alert matches the fingerprint that you obtained in step 1. If these fingerprints don't match, someone might be attempting a "man-in-the-middle" attack. If they match, continue to the next step.
7. Enter **yes**.

   You see a response like the following.

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com'
(RSA) to the list of known hosts.
```

Install the AWS Command Line Interface on Linux

URL Link: http://docs.aws.amazon.com/cli/latest/userguide/awscli-install-linux.html

You can install the AWS Command Line Interface and its dependencies on most Linux. The awscli package is available in repositories for other package managers such as APT and yum, but it is not guaranteed to be the latest version unless you get it from pip or use the bundled installer.

Amazon Linux

The AWS CLI comes pre-installed on the Amazon Linux AMI. Run **sudo yum update** after connecting to the instance to get the latest version of the package available via yum. If you need a more recent version of the AWS

CLI than what is available in the Amazon updates repository, uninstall the package (**sudo yum remove aws-cli**) and then install using pip.

Adding the AWS CLI Executable to your Command Line Path
After installing , add the aws executable to your OS's PATH environment variable.
Example AWS CLI install location - Linux with pip (user mode)
~/.local/bin

To modify your PATH variable (Linux, macOS, or Unix)

1. Find your shell's profile script in your user folder. If you are not sure which shell you have, run echo $SHELL.
   ```
   $ ls -a ~
   .  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
   ```
   - Bash – .bash_profile, .profile, or .bash_login.
   - Zsh – .zshrc
   - Tcsh – .tcshrc, .cshrc or .login.
2. Add an export command to profile script.
   ```
   export PATH=~/.local/bin:$PATH
   ```
   This command adds a path, ~/.local/bin in this example, to the current PATH variable.
3. Load the profile into your current session.
   ```
   $ source ~/.bash_profile
   ```
4. Verify that awscli is installed correctly.
   ```
   $ aws --version
   pip 8.1.2 from ~/.local/lib/python3.4/site-packages (python 3.4)
   ```
5. If not, use yum to install the AWS CLI.
   ```
   $ yum install aws-cli
   ```
6. Verify that the AWS CLI installed correctly.
   ```
   $ aws --version
   aws-cli/1.11.29 Python/2.7.12 Linux/4.4.41-36.55.amzn1.x86_64 botocore/1.4.86
   ```

To upgrade to the latest version, run the installation command again:
```
$ yum update aws-cli
```

Uninstalling
Amazon Linux instances manage their software using the yum package manager. The yum package manager can install, remove, and update software, as well as manage all of the dependencies for each package.

To remove Python, Pip and awscli

On Debian derivatives such as Ubuntu, use APT:
```
$ sudo apt-get remove aws-cli
```
On Red Hat and derivatives, use yum:
```
$ sudo yum remove aws-cli
```
On SUSE and derivatives, use zypper:
```
$ sudo zypper rm aws-cli
```

Stop and Start Your EC2 Instance

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Stop_Start.html

You can stop and restart your instance if it has an Amazon EBS volume as its root device. The instance retains its instance ID, but can change as described in the Overview section.

When you stop an instance, we shut it down. We don't charge hourly usage for a stopped instance, or data transfer fees, but we do charge for the storage for any Amazon EBS volumes.

Stopping and Starting Your Instances
You can start and stop your Amazon EBS-backed instance using the console or the command line.
By default, when you initiate a shutdown from an Amazon EBS-backed instance (using the shutdown, halt, or poweroff command), the instance stops. You can change this behavior so that it terminates instead.
To stop and start an Amazon EBS-backed instance using the console

1. In the navigation pane, choose **Instances**, and select the instance.
2. [EC2-Classic] If the instance has an associated Elastic IP address, write down the Elastic IP address and the instance ID shown in the details pane.
3. Choose **Actions**, select **Instance State**, and then choose **Stop**. If **Stop** is disabled, either the instance is already stopped or its root device is an instance store volume.

   Warning When you stop an instance, the data on any instance store volumes is erased. Therefore, if you have any data on instance store volumes that you want to keep, be sure to back it up to persistent storage.
4. In the confirmation dialog box, choose **Yes, Stop**. It can take a few minutes for the instance to stop. [EC2-Classic] When the instance state becomes stopped, the Elastic IP, Public DNS (IPv4), Private DNS, and Private IPs fields in the details pane are blank to indicate that the old values are no longer associated with the instance.
5. While your instance is stopped, you can modify certain instance attributes. For more information, see Modifying a Stopped Instance.
6. To restart the stopped instance, select the instance, choose **Actions**, select **Instance State**, and then choose **Start**.
7. In the confirmation dialog box, choose **Yes, Start**. It can take a few minutes for the instance to enter the `running state`.

Terminate Your EC2 Instance

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/terminating-instances.html

When you've decided that you no longer need an instance, you can terminate it. As soon as the state of an instance changes to `shutting-down` or `terminated`, you stop incurring charges for that instance.

You can't connect to or restart an instance after you've terminated it. However, you can launch additional instances using the same AMI. If you'd rather stop and restart your instance, see Stop and Start Your Instance. For more information, see Differences Between Reboot, Stop, and Terminate.

Instance Termination
After you terminate an instance, it remains visible in the console for a short while, and then the entry is automatically deleted.

Terminating an Instance
You can terminate an instance using the AWS Management Console or the command line.

To terminate an instance using the console

1. Before you terminate the instance, verify that you won't lose any data by checking that your Amazon EBS volumes won't be deleted on termination and that you've copied any data that you need from your instance store volumes to Amazon EBS or Amazon S3.
2. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
3. In the navigation pane, select **Instances**.

4. Select the instance, choose **Actions**, select **Instance State**, and then select **Terminate**.
5. Select **Yes, Terminate** when prompted for confirmation.

Relational Database Service (RDS)

HINT There is a lot of information for these steps. We have highlighted what you need to know for now, and will cover the details later. You can use the highlighting to speed up your lab, but be sure to go back and read the information to familiarize yourself with the key concepts. Don't worry about memorizing the information, just know what is possible.

Creating a DB Instance Running the MySQL Database Engine

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_CreateInstance.html

The easiest way to create a DB instance is to use the AWS Management Console. Once you have created the DB instance, you can use standard MySQL utilities such as MySQL Workbench to connect to a database on the DB instance.

The basic building block of Amazon RDS is the DB instance. The DB instance is where you create your MySQL databases.

AWS Management Console
To launch a MySQL DB instance

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top right corner of the AWS Management Console, select the region in which you want to create the DB instance.
3. In the navigation pane, click **Instances**.
4. Click **Launch DB Instance** to start the **Launch DB Instance Wizard**. The wizard opens on the **Select Engine** page
   (Ex. Choose MySQL which has free tier offering).
5. In the **Launch DB Instance Wizard** window, click the **Select** button for the MySQL DB engine.
6. The next step asks if you are planning to use the DB instance you are creating for production. If you are, select **Yes**. By selecting **Yes**, the failover option **Multi-AZ** and the **Provisioned IOPS** storage option will be preselected in the following step. Click **Next** when you are finished.
   (Ex. Select **Dev/Test MySQL**)
7. On the **Specify DB Details** page, specify your DB instance information. The following table shows settings for an example DB instance. Click **Next** when you are finished.
   (Ex.
   DB Instance Class: **db.t2.micro**
   Multi-AZ Deployment: **No**
   DB Instance Identifier: **oawsDB**
   Master Username: **dbadmin**
   Master Password: **dbadmin!**
   Confirm Password: **dbadmin!**
)

For this parameter...	...Do this:
License Model	MySQL has only one license model. Select the default, **`General-Public-License`**, to use the general license agreement for MySQL.

DB Engine Version	Select the version of MySQL that you want to work with. Note that Amazon RDS supports several versions of MySQL.
DB Instance Class	Select a DB instance class that defines the processing and memory requirements for the DB instance. For more information about all the DB instance class options, see DB Instance Class.
Multi-AZ Deployment	Determine if you want to create a standby replica of your DB instance in another Availability Zone for failover support. For more information about multiple Availability Zones, see Regions and Availability Zones.
Allocated Storage	Type a value to allocate storage for your database (in gigabytes). In some cases, allocating a higher amount of storage for your DB instance than the size of your database can improve I/O performance. For more information about storage allocation, see Amazon RDS Storage Types.
Storage Type	Select the storage type you want to use. For more information about storage, see Storage for Amazon RDS.
DB Instance Identifier	Type a name for the DB instance that is unique for your account in the region you selected. You may choose to add some intelligence to the name such as including the region and DB Engine you selected, for example **`mysql-instance1`**.
Master Username	Type a name using alphanumeric characters that you will use as the master user name to log on to your DB instance. The default privileges granted to the master user name account include: create, drop, references, event, alter, delete, index, insert, select, update, create temporary tables, lock tables, trigger, create view, show view, alter routine, create routine, execute, create user, process, show databases, grant option.
Master Password	Type a password that contains from 8 to 16 printable ASCII characters (excluding /,", and @) for your master user password.
Confirm Password	Re-type the Master Password for confirmation.

8. On the **Configure Advanced Settings** page, provide additional information that RDS needs to launch the MySQL DB instance. The table shows settings for an example DB instance. Specify your DB instance information, then click **Next Step**.
   (Ex. Database Name : **oawsDB)**

For this parameter...	...Do this:
VPC	Select the name of the Virtual Private Cloud (VPC) that will host your MySQL DB instance. If your DB instance will not be hosted in a VPC, select **Not in VPC**. For more information about VPC, see Virtual Private Clouds (VPCs) and Amazon RDS.
DB Subnet Group	This setting depends on the platform you are on. If you are a new customer to AWS, select **`default`**, which will be the default DB subnet group that was created for your account. If you are creating a DB instance on the previous E2-Classic platform and you want your DB instance in a specific VPC, select the DB subnet group you created for that VPC. For more information about VPC, see Virtual Private Clouds (VPCs) and Amazon RDS.
Publicly Accessible	Choose **`Yes`** to give the DB instance a public IP address, meaning that it will be accessible outside the VPC (the DB instance also needs to be in a public subnet in the VPC); otherwise, choose **`No`**, so the DB instance will only be accessible from inside the VPC. For more information about hiding DB instances from public access, see Hiding a DB Instance in a VPC from the Internet.

Availability Zone	Determine if you want to specify a particular Availability Zone. If you selected **Yes** for the Multi-AZ Deployment parameter on the previous page, you will not have any options here. For more information about Availability Zones, see Regions and Availability Zones.
DB Security Groups	Select the security group you want to use with this DB instance. For more information about security groups, see Working with DB Security Groups.
Database Name	Type a name for your default database of 1 to 64 alpha-numeric characters. If you do not provide a name, Amazon RDS will not create a database on the DB instance you are creating. To create additional databases, connect to the DB instance and use the SQL command CREATE DATABASE. For more information about connecting to the DB instance, see Connecting to a DB Instance Running the MySQL Database Engine.
Database Port	Specify the port that applications and utilities will use to access the database. MySQL installations default to port 3306. The firewalls at some companies block connections to the default MySQL port. If your company firewall blocks the default port, choose another port for the new DB instance.
DB Parameter Group	Select a **DB Parameter Group**, which is used to manage your DB engine configuration. Each MySQL version has a default parameter group you can use, or you can create your own parameter group. For DB engine configurations that you frequently use or to increase your database engine uptime, you can create your own DB parameter group. For more information about parameter groups, see Working with DB Parameter Groups.
Option Group	Select an **Option Group**, which is used to enable and configure DB engine features. Each MySQL version has a default option group you can use, or you can create your own option group. For more information about option groups, see Working with Option Groups.
Copy Tags To Snapshots	Choose this option to have any DB instance tags copied to a DB snapshot when you create a snapshot. For more information, see Tagging Amazon RDS Resources.
Enable Encryption	Select Yes to enable encryption at rest for this DB instance. For more information, see Encrypting Amazon RDS Resources.
Backup Retention Period	Select the number of days for Amazon RDS to automatically back up your DB instance. You can recover your database to any point in time during that retention period. For more information, see Working With Backups.
Backup Window	Specify the period of time during which your DB instance is backed up. During the backup window, storage I/O may be suspended while your data is being backed up and you may experience elevated latency. This I/O suspension typically lasts for the duration of the snapshot. This period of I/O suspension is shorter for Multi-AZ DB deployments, since the backup is taken from the standby, but latency can occur during the backup process. For more information, see Working With Backups.
Enable Enhanced Monitoring	Choose **Yes** to enable gathering metrics in real time for the operating system that your DB instance runs on. For more information, see Enhanced Monitoring.
Granularity	Only available if **Enable Enhanced Monitoring** is set to **Yes**. Set the interval, in seconds, between when metrics are collected for your DB instance.
Auto Minor Version Upgrade	Select **Yes** if you want to enable your DB instance to receive minor DB Engine version upgrades automatically when they become available.
Maintenance Window	Select the weekly time range during which system maintenance can occur. For more information about the maintenance window, see Adjusting the Preferred DB Instance Maintenance Window.

In addition, Federated Storage Engine is currently not supported by Amazon RDS for MySQL.

9. Click **Launch DB Instance** to create your MySQL DB instance.
10. On the final page of the wizard, click **Close**.

11. On the Amazon RDS console, the new DB instance appears in the list of DB instances. The DB instance will have a status of **creating** until the DB instance is created and ready for use. When the state changes to **available**, you can connect to the DB instance. Depending on the DB instance class and store allocated, it could take several minutes for the new instance to be available.

Modifying a DB Instance Running the MySQL Database Engine

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ModifyInstance.MySQL.html

You can change the settings of a DB instance to accomplish tasks such as adding additional storage or changing the DB instance class.

You can have the changes apply immediately or have them applied during the DB instance's next maintenance window. Applying changes immediately can cause an outage in some cases; for more information on the impact of the Apply Immediately option when modifying a DB instance, see Modifying an Amazon RDS DB Instance and Using the Apply Immediately Parameter.

To modify a MySQL DB instance

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the navigation pane, click **Instances**.
3. Select the check box for the DB instance that you want to change, click **Instance Actions** and then click **Modify**.
4. In the **Modify DB** Instance dialog box, change any of the following settings that you want:
   (Ex. **Backup Retention Period** = 1 Day)

Setting	Description
Instance Specifications	
DB Engine Version	In the list provided, click the version of the MySQL database engine that you want to use.
DB Instance Class	In the list provided, click the DB instance class that you want to use. For information about instance classes, see DB Instance Class.
Multi-AZ Deployment	If you want to deploy your DB instance in multiple Availability Zones, click **Yes**; otherwise, click **No** .
Storage Type	Select the storage type you want to use. For more information about storage, see Storage for Amazon RDS. The following changes cause an outage to occur: From **General Purpose (SSD)** to **Magnetic**. From **General Purpose (SSD)** to **Provisioned IOPS (SSD)**, if you are using a custom parameter group. From **Magnetic** to **General Purpose (SSD)**. From **Magnetic** to **Provisioned IOPS (SSD)**. From **Provisioned IOPS (SSD)** to **Magnetic**. From **Provisioned IOPS (SSD)** to **General Purpose (SSD)**, if you are using a custom parameter group.
Allocated Storage	Specify how much storage, in gigabytes, to allocate for your DB instance. The minimum allowable value is 5 GB; the maximum is 6 TB. Note that you can only increase the amount of storage when modifying a DB instance, you cannot reduce the amount of storage allocated.
Settings	
DB Instance Identifier	You can rename the DB instance by typing a new name. When you change the DB instance identifier, an instance reboot will occur immediately if you set `Apply Immediately` to **true**, or will occur during the next maintenance window if you set `Apply Immediately` to **false**. This value is stored as a lowercase string.

New Master Password	Type a password for your master user. The password must contain from 8 to 41 alphanumeric characters. By resetting the master password, you also reset permissions for the DB instance. For more information, see Resetting the DB Instance Owner Role Password.
Network and Security	
Subnet Group	Choose the subnet group for the DB instance. You can use this setting to move your DB instance to a different VPC. If your DB instance is not in a VPC, you can use this setting to move your DB instance into a VPC. For more information, see Moving a DB Instance Not in a VPC into a VPC.
Security Group	Select the security group you want associated with the DB instance. For more information about security groups, see Working with DB Security Groups.
Certificate Authority	Select the certificate you want to use.
Publicly Accessible	Choose **Yes** to give the DB instance a public IP address, meaning that it will be accessible outside the VPC (the DB instance also needs to be in a public subnet in the VPC); otherwise, choose **No**, so the DB instance will only be accessible from inside the VPC. For more information about hiding DB instances from public access, see Hiding a DB Instance in a VPC from the Internet.
Database Options	
Parameter Group	Select the parameter group you want associated with the DB instance. Changing this setting does not result in an outage. The parameter group name itself is changed immediately, but the actual parameter changes are not applied until you reboot the instance without failover. The DB instance will NOT be rebooted automatically and the parameter changes will NOT be applied during the next maintenance window. For more information about parameter groups, see Working with DB Parameter Groups.
Option Group	Select the option group you want associated with the DB instance. For more information about option groups, see Working with Option Groups.
Copy Tags to Snapshots	Select this option to have any DB instance tags copied to a DB snapshot when you create a snapshot.
Database Port	Specify a new port you want to use to access the database. The port value must not match any of the port values specified for options in the option group for the DB instance. Your database will restart when you change the database port regardless of whether **Apply Immediately** is checked.
Backup	
Backup Retention Period	Specify the number of days that automatic backups will be retained. To disable automatic backups, set this value to 0. **Note** An immediate outage will occur if you change the backup retention period from 0 to a non-zero value or from a non-zero value to 0.
Backup Window	Set the time range during which automated backups of your databases will occur. Specify a start time in Universal Coordinated Time (UTC) and a duration in hours.
Enable Enhanced Monitoring	Choose **Yes** to enable gathering metrics in real time for the operating system that your DB instance runs on. For more information, see Enhanced Monitoring.
Granularity	Only available if **Enable Enhanced Monitoring** is set to **Yes**. Set the interval, in seconds, between when metrics are collected for your DB instance.
Maintenance	
Auto Minor Version Upgrade	If you want your DB instance to receive minor engine version upgrades automatically when they become available, click **Yes**. Upgrades are installed only during your scheduled maintenance window.
Maintenance Window	Set the time range during which system maintenance, including upgrades, will occur. Specify a start time in UTC and a duration in hours.

5. To apply the changes immediately, select the **Apply Immediately** check box. Selecting this option can cause an outage in some cases; for more information on the impact of the **Apply Immediately** option, see Modifying an Amazon RDS DB Instance and Using the Apply Immediately Parameter.

6. When all the changes are as you want them, choose **Continue**.
7. On the confirmation page, review your changes. If they are correct, choose **Modify DB Instance** to save your changes.
   Alternatively, choose **Back** to edit your changes, or choose **Cancel** to cancel your changes.

Rebooting a DB Instance

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RebootInstance.html

In some cases, if you modify a DB instance, change the DB parameter group associated with the instance, or change a static DB parameter in a parameter group the instances uses, you must reboot the instance for the changes to take effect.

Rebooting a DB instance restarts the database engine service. A reboot also applies to the DB instance any modifications to the associated DB parameter group that were pending. Rebooting a DB instance results in a momentary outage of the instance, during which the DB instance status is set to rebooting

The time required to reboot is a function of the specific database engine's crash recovery process. To improve the reboot time, we recommend that you reduce database activities as much as possible during the reboot process to reduce rollback activity for in-transit transactions.

In the console, the **Reboot** option may be disabled if the DB instance is not in the "Available" state. This can be due to several reasons, such as an in-progress backup or a customer-requested modification or a maintenance-window action.

AWS Management Console
To reboot a DB instance

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the navigation pane, click **Instances**.
3. Select the check box of the DB instance that you want to reboot.
4. Select **Instance Actions** and then select **Reboot** from the drop down menu.
5. To force a failover from one AZ to another, select the **Reboot with failover**? check box in the **Reboot DB Instance** dialog box.
6. Click **Yes, Reboot**. To cancel the reboot instead, click **Cancel**.

Deleting a DB Instance

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_DeleteInstance.html

You can delete a DB instance in any state and at any time. To delete a DB instance, you must specify the name of the instance and specify if you want to have a final DB snapshot taken of the instance.

Important
If you choose not to create a final DB snapshot, you will not be able to later restore the DB instance to its final state. When you delete a DB instance, all automated backups are deleted and cannot be recovered. Manual DB snapshots of the instance are not deleted.

If the DB instance you want to delete has a Read Replica, you should either promote the Read Replica or delete it.

Deleting a DB Instance with No Final Snapshot

You can skip creating a final DB snapshot if you want to quickly delete a DB instance. Note that when you delete a DB instance, all automated backups are deleted and cannot be recovered. Manual snapshots are not deleted.

AWS Management Console

To delete a DB instance with no final DB snapshot

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the **DB Instances** list, select the check box next to the DB instance you wish to delete.
3. Click **Instance Actions**, and then select **Delete** from the context menu.
4. Select **No** in the **Create final Snapshot**? drop-down list box.
5. Click **Yes, Delete**.

Deleting a DB Instance with a Final Snapshot

You can create a final DB snapshot if you want to be able to restore a deleted DB instance at a later time. All automated backups will also be deleted and cannot be recovered. Manual snapshots are not deleted.

AWS Management Console

To delete a DB instance with a final DB snapshot

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the **DB Instances** list, select the check box next to the DB Instance you wish to delete.
3. Click **Instance Actions**, and then select **Delete** from the context menu.
4. Select **Yes** in the **Create final Snapshot**? drop-down list box.
5. Type the name of your final DB snapshot into the **Final Snapshot name** text box.
6. Click **Yes, Delete**.

Cleanup!

Stop/Terminate All Unneeded Resources

Delete all resources that you built during your lab, but do not intend to keep.

- Delete any unneeded RDS DB Instance
- Delete any unneeded EC2 Instances
- Delete any unneeded CloudFront Distributions
- Delete any unneeded S3 Buckets

We recommend, if you have time, repeat the labs several times. This is pretty basic stuff, but it will get advanced quickly.

Look around the AWS Management Console. There are dozens of services we have not touched on. Don't be preoccupied. We will get through it in a fun and easy way.

Environment Configuration

At the end of this lab, your environment should be configured as follows:

Lesson 2

Simple Storage Services (S3)

1. S3 Bucket Created (1 bucket).
2. Object, a index.html file, uploaded and made public in the S3 Bucket

CloudFront

3. One CloudFront Web Distribution for our S3 Bucket (1 Distribution)

Elastic Compute Cloud (EC2)

4. EC2 Key Pair Created (1 Key Pair)

Knowledge Base

Amazon Web Services

AWS Global Infrastructure

URL Link https://aws.amazon.com/about-aws/global-infrastructure/

The AWS Cloud operates 42 Availability Zones within 16 geographic Regions around the world, with five more Availability Zones and two more Regions coming online throughout the next year.

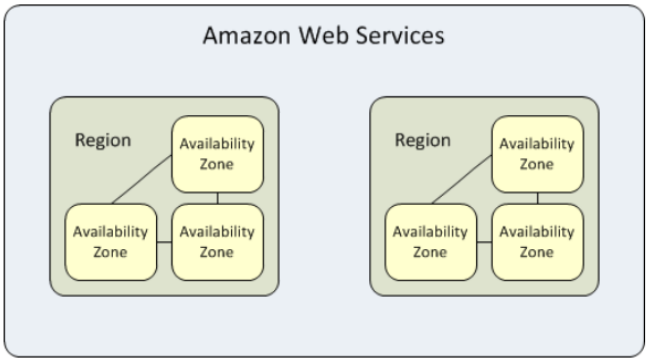
AWS Regions and Availability Zones
The AWS Cloud infrastructure is built around Regions and Availability Zones ("AZs"). A Region is a physical location in the world where we have multiple Availability Zones. Availability Zones consist of one or more discrete data centers, each with redundant power, networking and connectivity, housed in separate facilities. These Availability Zones offer you the ability to operate production applications and databases which are more highly available, fault tolerant and scalable than would be possible from a single data center. The AWS Cloud operates 42 Availability Zones within 16 geographic Regions around the world.

Regions and Availability Zones

URL Link http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html

Amazon EC2 is hosted in multiple locations world-wide. These locations are composed of regions and Availability Zones. Each region is a separate geographic area. Each region has multiple, isolated locations known as Availability Zones. Amazon EC2 provides you the ability to place resources, such as instances, and data in multiple locations. Resources aren't replicated across regions unless you do so specifically.
Amazon operates state-of-the-art, highly-available data centers. Although rare, failures can occur that affect the availability of instances that are in the same location. If you host all your instances in a single location that is affected by such a failure, none of your instances would be available.

Region and Availability Zone Concepts
Each region is completely independent. Each Availability Zone is isolated, but the Availability Zones in a region are connected through low-latency links. The following diagram illustrates the relationship between regions and Availability Zones.



Amazon EC2 resources are either global, tied to a region, or tied to an Availability Zone. For more information, see Resource Locations.

Regions
Each Amazon EC2 region is designed to be completely isolated from the other Amazon EC2 regions. This achieves the greatest possible fault tolerance and stability.

When you view your resources, you'll only see the resources tied to the region you've specified. This is because regions are isolated from each other, and we don't replicate resources across regions automatically.

When you launch an instance, you must select an AMI that's in the same region.

Availability Zones
When you launch an instance, you can select an Availability Zone or let us choose one for you. If you distribute your instances across multiple Availability Zones and one instance fails, you can design your application so that an instance in another Availability Zone can handle requests.

An Availability Zone is represented by a region code followed by a letter identifier; for example, us-east-1a. To ensure that resources are distributed across the Availability Zones for a region, we independently map Availability Zones to identifiers for each account. For example, your Availability Zone us-east-1a might not be the same location as us-east-1a for another account. There's no way for you to coordinate Availability Zones between accounts.

The following table lists the regions provided by an AWS account. You can't describe or access additional regions from an AWS account, such as AWS GovCloud (US) or China (Beijing).

Code	Name
us-east-1	US East (N. Virginia)
us-east-2	US East (Ohio)
us-west-1	US West (N. California)
us-west-2	US West (Oregon)
ca-central-1	Canada (Central)
eu-west-1	EU (Ireland)
eu-central-1	EU (Frankfurt)
eu-west-2	EU (London)
ap-northeast-1	Asia Pacific (Tokyo)
ap-northeast-2	Asia Pacific (Seoul)
ap-southeast-1	Asia Pacific (Singapore)
ap-southeast-2	Asia Pacific (Sydney)
ap-south-1	Asia Pacific (Mumbai)
sa-east-1	South America (São Paulo)

For more information, see AWS Global Infrastructure.

The number and mapping of Availability Zones per region may vary between AWS accounts. To get a list of the Availability Zones that are available to your account, you can use the Amazon EC2 console or the command line interface. For more information, see Describing Your Regions and Availability Zones.

Regions and Endpoints
When you work with an instance using the command line interface or API actions, you must specify its regional endpoint. For more information about the regions and endpoints for Amazon EC2, see Regions and Endpoints in the Amazon Web Services General Reference.

Simple Storage Services (S3)

What Is Amazon S3?

URL Link http://docs.aws.amazon.com/AmazonS3/latest/dev/Welcome.html

Amazon Simple Storage Service is storage for the Internet. It is designed to make web-scale computing easier for developers.

Amazon S3 has a simple web services interface that you can use to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, fast, inexpensive data storage infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers.

This guide explains the core concepts of Amazon S3, such as buckets and objects, and how to work with these resources using the Amazon S3 application programming interface (API).

Working with Amazon S3 Buckets

URL Link http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingBucket.html

Amazon S3 is cloud storage for the Internet. To upload your data (photos, videos, documents etc.), you first create a bucket in one of the AWS Regions. You can then upload any number of objects to the bucket.
In terms of implementation, buckets and objects are resources.

Amazon S3 bucket names are globally unique, regardless of the AWS Region in which you create the bucket. You specify the name at the time you create the bucket.

Amazon S3 creates buckets in a region you specify. You can choose any AWS Region that is geographically close to you to optimize latency, minimize costs, or address regulatory requirements. For example, if you reside in Europe, you might find it advantageous to create buckets in the EU (Ireland) or EU (Frankfurt) regions.

> **Note**
> Objects belonging to a bucket that you create in a specific AWS Region never leave that region, unless you explicitly transfer them to another region. For example, objects stored in the EU (Ireland) region never leave it.

Creating a Bucket
By default, you can create up to 100 buckets in each of your AWS accounts. If you need additional buckets, you can increase your bucket limit by submitting a service limit increase.

When you create a bucket, you provide a name and AWS Region where you want the bucket created.

Within each bucket, you can store any number of objects. You can create a bucket using any of the following methods:

- Create the bucket using the console.
- Create the bucket programmatically using the AWS SDKs.

About Permissions
You can use your AWS account root credentials to create a bucket and perform any other Amazon S3 operation. However, AWS recommends not using the root credentials of your AWS account to make requests such as create a bucket. Instead, create an IAM user, and grant that user full access (users by default have no

permissions). We refer to these users as administrator users. You can use the administrator user credentials, instead of the root credentials of your account, to interact with AWS and perform tasks, such as create a bucket, create users, and grant them permissions.

The AWS account that creates a resource owns that resource. For example, if you create an IAM user in your AWS account and grant the user permission to create a bucket, the user can create a bucket. But the user does not own the bucket; the AWS account to which the user belongs owns the bucket. The user will need additional permission from the resource owner to perform any other bucket operations.

Accessing a Bucket
You can access your bucket using the Amazon S3 console. Using the console UI, you can perform almost all bucket operations without having to write any code.

Amazon S3 supports both virtual-hosted–style and path-style URLs to access a bucket.

- In a virtual-hosted–style URL, the bucket name is part of the domain name in the URL. For example:
  - `http://bucket.s3.amazonaws.com`
  - `http://bucket.s3-aws-region.amazonaws.com`.

  In a virtual-hosted–style URL, you can use either of these endpoints.
- In a path-style URL, the bucket name is not part of the domain (unless you use a region-specific endpoint). For example:
  - US East (N. Virginia) region endpoint, `http://s3.amazonaws.com/bucket`
  - Region-specific endpoint, `http://s3-aws-region.amazonaws.com/bucket`

  In a path-style URL, the endpoint you use must match the region in which the bucket resides.

Important
Because buckets can be accessed using path-style and virtual-hosted–style URLs, we recommend you create buckets with DNS-compliant bucket names.

Working with Amazon S3 Objects

URL Link http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingObjects.html

Amazon S3 is a simple key, value store designed to store as many objects as you want. You store these objects in one or more buckets. An object consists of the following:

- **Key** – The name that you assign to an object. You use the object key to retrieve the object.
- **Value** – The content that you are storing. An object value can be any sequence of bytes. Objects can range in size from zero to 5 TB.
- **Metadata** – A set of name-value pairs with which you can store information regarding the object. You can assign metadata, referred to as user-defined metadata, to your objects in Amazon S3. Amazon S3 also assigns system-metadata to these objects, which it uses for managing objects.
- **Access Control Information** – You can control access to the objects you store in Amazon S3. Amazon S3 supports both the resource-based access control, such as an Access Control List (ACL) and bucket policies, and user-based access control.

For more information about working with objects, see the following sections. Note that your Amazon S3 resources (for example buckets and objects) are private by default. You will need to explicitly grant permission for others to access these resources. For example, you might want to share a video or a photo stored in your Amazon S3 bucket on your website. That will work only if you either make the object public or use a presigned URL on your website.

CloudFront

URL Link http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html

What Is Amazon CloudFront?

Amazon CloudFront is a web service that speeds up distribution of your static and dynamic web content, such as .html, .css, .php, and image files, to your users. CloudFront delivers your content through a worldwide network of data centers called **edge locations**. When a user requests content that you're serving with CloudFront, the user is routed to the edge location that provides the lowest latency (time delay), so that content is delivered with the best possible performance. If the content is already in the edge location with the lowest latency, CloudFront delivers it immediately. If the content is not in that edge location, CloudFront retrieves it from an Amazon S3 bucket or an HTTP server (for example, a web server) that you have identified as the source for the definitive version of your content.

Suppose you're serving the following image from a traditional web server, not from CloudFront.
(The image is owned by NASA and comes from the Visible Earth website at http://visibleearth.nasa.gov/.)

You're serving the image using the URL `http://example.com/globe_west_540.png`. Your users can easily navigate to this URL and see the image, but they probably don't know that their request was routed from one network to another—through the complex collection of interconnected networks that comprise the Internet—until the image was found.

Further suppose that the web server that you're serving the image from is in Seattle, Washington, USA, and that a user in Austin, Texas, USA requests the image. The following traceroute list (courtesy of www.WatchMouse.com) shows one way that this request could be routed.

In this example, the request was routed 10 times within the United States before the image was retrieved, which is not an unusually large number of hops (the network paths between the source and the final destination). If your user were in Europe, the request would be routed through even more networks to reach your server in Seattle. The number of networks and the distance that the request and the image must travel have a significant impact on the performance, reliability, and availability of the image.

CloudFront speeds up the distribution of your content by routing each user request to the edge location that can best serve your content. Typically, this is the CloudFront edge location that provides the lowest latency. This dramatically reduces the number of networks that your users' requests must pass through, which improves performance. Users get lower latency—the time it takes to load the first byte of the file—and higher data transfer rates. You also get increased reliability and availability because copies of your files (also known as objects) are now held in multiple edge locations around the world.

Overview of Web and RTMP Distributions

URL Linkhttp://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/distribution-overview.html

When you want to use CloudFront to distribute your content, you create a distribution and specify configuration settings such as:

- Your origin, which is the Amazon S3 bucket or HTTP server from which CloudFront gets the files that it distributes. You can specify any combination of up to 10 Amazon S3 buckets and/or HTTP servers as your origins.
- Whether you want the files to be available to everyone or you want to restrict access to selected users.
- Whether you want CloudFront to require users to use HTTPS to access your content.
- Whether you want CloudFront to forward cookies and/or query strings to your origin.

- Whether you want CloudFront to prevent users in selected countries from accessing your content.
- Whether you want CloudFront to create access logs.

The number of files that you can serve per distribution is unlimited.

Web Distributions
You can use web distributions to serve the following content over HTTP or HTTPS:

- Static and dynamic download content, for example, .html, .css, .php, and image files, using HTTP or HTTPS.
- Multimedia content on demand using progressive download and Apple HTTP Live Streaming (HLS). For more information, see the applicable topic in Working with Web Distributions. You can't serve Adobe Flash multimedia content over HTTP or HTTPS, but you can serve it using a CloudFront RTMP distribution. See RTMP Distributions below.
- A live event, such as a meeting, conference, or concert, in real time. For live streaming, you create the distribution automatically by using an AWS CloudFormation stack. For more information, see the applicable live-streaming tutorial in CloudFront Streaming Tutorials.

For web distributions, your origin can be either an Amazon S3 bucket or an HTTP server, for example, a web server. For more information about how web distributions work, including the values that you specify when you create a web distribution, see Working with Web Distributions. For information about creating a web distribution, see Task List for Creating a Web Distribution.

RTMP Distributions
RTMP distributions stream media files using Adobe Media Server and the Adobe Real-Time Messaging Protocol (RTMP). An RTMP distribution must use an Amazon S3 bucket as the origin.

For information about the values you specify when you create an RTMP distribution, see Working with RTMP Distributions. For information about creating an RTMP distribution, see Task List for Streaming Media Files Using RTMP.

Amazon Elastic Compute Cloud (EC2)

What Is Amazon EC2?

URL Link http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

Features of Amazon EC2

Amazon EC2 provides the following features:

- Virtual computing environments, known as instances
- Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types
- Secure login information for your instances using key pairs (AWS stores the public key, and you store the private key in a secure place)
- Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as instance store volumes
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as regions and Availability Zones
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups
- Static IPv4 addresses for dynamic cloud computing, known as Elastic IP addresses
- Metadata, known as tags, that you can create and assign to your Amazon EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as virtual private clouds (VPCs)

How to Get Started with Amazon EC2

The first thing you need to do is get set up to use Amazon EC2. After you are set up, you are ready to complete the Getting Started tutorial for Amazon EC2. Whenever you need more information about a feature of Amazon EC2, you can read the technical documentation.

Get Up and Running

- Setting Up with Amazon EC2
- Getting Started with Amazon EC2 Linux Instances

Basics

- Instances and AMIs
- Regions and Availability Zones
- Instance Types
- Tags

Networking and Security

- Amazon EC2 Key Pairs
- Security Groups
- Elastic IP Addresses

- Amazon EC2 and Amazon VPC

Storage

- Amazon EBS
- Instance Store

Related Services

You can provision Amazon EC2 resources, such as instances and volumes, directly using Amazon EC2. You can also provision Amazon EC2 resources using other services in AWS.

To automatically distribute incoming application traffic across multiple instances, use Elastic Load Balancing.

To monitor basic statistics for your instances and Amazon EBS volumes, use Amazon CloudWatch.

To monitor the calls made to the Amazon EC2 API for your account, including calls made by the AWS Management Console, command line tools, and other services, use AWS CloudTrail.

To get a managed relational database in the cloud, use Amazon Relational Database Service (Amazon RDS) to launch a database instance. Although you can set up a database on an EC2 instance, Amazon RDS offers the advantage of handling your database management tasks, such as patching the software, backing up, and storing the backups.

To import virtual machine (VM) images from your local environment into AWS and convert them into ready-to-use AMIs or instances, use VM Import/Export.

Accessing Amazon EC2

Amazon EC2 provides a web-based user interface, the Amazon EC2 console. If you've signed up for an AWS account, you can access the Amazon EC2 console by signing into the AWS Management Console and selecting EC2 from the console home page.

If you prefer to use a command line interface, you have the following options:

- AWS Command Line Interface (CLI)
  Provides commands for a broad set of AWS products, and is supported on Windows, Mac, and Linux. To get started, see AWS Command Line Interface User Guide.
- AWS Tools for Windows PowerShell
  Provides commands for a broad set of AWS products for those who script in the PowerShell environment. To get started, see the AWS Tools for Windows PowerShell User Guide.

Pricing for Amazon EC2

When you sign up for AWS, you can get started with Amazon EC2 for free using the AWS Free Tier.
Amazon EC2 provides the following purchasing options for instances:

On-Demand instances
: Pay for the instances that you use by the hour, with no long-term commitments or up-front payments.

Reserved Instances
: Make a low, one-time, up-front payment for an instance, reserve it for a one- or three-year term, and pay a significantly lower hourly rate for these instances.

Spot instances
: Specify the maximum hourly price that you are willing to pay to run a particular instance type. The Spot price fluctuates based on supply and demand, but you never pay more than the maximum price you

specified. If the Spot price moves higher than your maximum price, Amazon EC2 shuts down your Spot instances.

For a complete list of charges and specific prices for Amazon EC2, see Amazon EC2 Pricing.

To calculate the cost of a sample provisioned environment, see AWS Economics Center.

To see your bill, go to your AWS Account Activity page. Your bill contains links to usage reports that provide details about your bill. To learn more about AWS account billing, see AWS Account Billing.

If you have questions concerning AWS billing, accounts, and events, contact AWS Support.

For an overview of Trusted Advisor, a service that helps you optimize the costs, security, and performance of your AWS environment, see AWS Trusted Advisor.

Amazon EC2 Key Pairs

Amazon EC2 uses public–key cryptography to encrypt and decrypt login information. Public–key cryptography uses a public key to encrypt a piece of data, such as a password, then the recipient uses the private key to decrypt the data. The public and private keys are known as a key pair.

To log in to your instance, you must create a key pair, specify the name of the key pair when you launch the instance, and provide the private key when you connect to the instance. Linux instances have no password, and you use a key pair to log in using SSH. With Windows instances, you use a key pair to obtain the administrator password and then log in using RDP.

Creating a Key Pair

You can use Amazon EC2 to create your key pair.

Alternatively, you could use a third-party tool and then import the public key to Amazon EC2.

Each key pair requires a name. Be sure to choose a name that is easy to remember. Amazon EC2 associates the public key with the name that you specify as the key name.

Amazon EC2 stores the public key only, and you store the private key. Anyone who possesses your private key can decrypt your login information, so it's important that you store your private keys in a secure place.
The keys that Amazon EC2 uses are 2048-bit SSH-2 RSA keys. You can have up to five thousand key pairs per region.

Launching and Connecting to Your Instance

When you launch an instance, you should specify the name of the key pair you plan to use to connect to the instance. If you don't specify the name of an existing key pair when you launch an instance, you won't be able to connect to the instance. When you connect to the instance, you must specify the private key that corresponds to the key pair you specified when you launched the instance.

Note

Amazon EC2 doesn't keep a copy of your private key; therefore, if you lose a private key, there is no way to recover it. If you lose the private key for an instance store-backed instance, you can't access the instance; you should terminate the instance and launch another instance using a new key pair. If you lose the private key for an EBS-backed Linux instance, you can regain access to your instance.

Key Pairs for Multiple Users
If you have several users that require access to a single instance, you can add user accounts to your instance. For more information, see Managing User Accounts on Your Linux Instance. You can create a key pair for each user, and add the public key information from each key pair to the .ssh/authorized_keys file for each user on your instance. You can then distribute the private key files to your users. That way, you do not have to distribute the same private key file that's used for the root account to multiple users.

Stop and Start Your EC2 Instance

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Stop_Start.html

You can stop and restart your instance if it has an Amazon EBS volume as its root device. The instance retains its instance ID, but can change as described in the Overview section.

When you stop an instance, we shut it down. We don't charge hourly usage for a stopped instance, or data transfer fees, but we do charge for the storage for any Amazon EBS volumes. Each time you start a stopped instance we charge a full instance hour, even if you make this transition multiple times within a single hour.

While the instance is stopped, you can treat its root volume like any other volume, and modify it (for example, repair file system problems or update software). You just detach the volume from the stopped instance, attach it to a running instance, make your changes, detach it from the running instance, and then reattach it to the stopped instance. Make sure that you reattach it using the storage device name that's specified as the root device in the block device mapping for the instance.

If you decide that you no longer need an instance, you can terminate it. As soon as the state of an instance changes to `shutting-down` or `terminated`, we stop charging for that instance. For more information, see Terminate Your Instance.

Contents

Overview
You can only stop an Amazon EBS-backed instance. To verify the root device type of your instance, describe the instance and check whether the device type of its root volume is `ebs` (Amazon EBS-backed instance) or `instance store` (instance store-backed instance).

When you stop a running instance, the following happens:

- The instance performs a normal shutdown and stops running; its status changes to stopping and then stopped.
- Any Amazon EBS volumes remain attached to the instance, and their data persists.
- Any data stored in the RAM of the host computer or the instance store volumes of the host computer is gone.
- In most cases, the instance is migrated to a new underlying host computer when it's started.
- When you stop and start a Windows instance, the EC2Config service performs tasks on the instance such as changing the drive letters for any attached Amazon EBS volumes.

For more information, see Differences Between Reboot, Stop, and Terminate.

You can modify the following attributes of an instance only when it is stopped:

- Instance type
- User data
- Kernel
- RAM disk

If you try to modify these attributes while the instance is running, Amazon EC2 returns the `IncorrectInstanceState` error.

Stopping and Starting Your Instances
You can start and stop your Amazon EBS-backed instance using the console or the command line.
By default, when you initiate a shutdown from an Amazon EBS-backed instance (using the shutdown, halt, or poweroff command), the instance stops. You can change this behavior so that it terminates instead.

Modifying a Stopped Instance
You can change the instance type, user data, and EBS-optimization attributes of a stopped instance using the AWS Management Console or the command line interface. You can't use the AWS Management Console to modify the `DeleteOnTermination`, kernel, or RAM disk attributes.
To modify an instance attribute

- To change the instance type, see Resizing Your Instance.
- To change the user data for your instance, see Configuring Instances with User Data.
- To enable or disable EBS–optimization for your instance, see Modifying EBS–Optimization.
- To change the `DeleteOnTermination` attribute of the root volume for your instance, see Updating the Block Device Mapping of a Running Instance

Troubleshooting
If you have stopped your Amazon EBS-backed instance and it appears "stuck" in the `stopping` state, you can forcibly stop it. For more information, see Troubleshooting Stopping Your Instance.

Terminate Your EC2 Instance

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/terminating-instances.html

When you've decided that you no longer need an instance, you can terminate it. As soon as the state of an instance changes to `shutting-down` or `terminated`, you stop incurring charges for that instance.

You can't connect to or restart an instance after you've terminated it. However, you can launch additional instances using the same AMI. If you'd rather stop and restart your instance, see Stop and Start Your Instance. For more information, see Differences Between Reboot, Stop, and Terminate.

Instance Termination
After you terminate an instance, it remains visible in the console for a short while, and then the entry is automatically deleted. You cannot delete the terminated instance entry yourself. After an instance is terminated, resources such as tags and volumes are gradually disassociated from the instance, therefore may no longer be visible on the terminated instance after a short while.

When an instance terminates, the data on any instance store volumes associated with that instance is deleted. By default, Amazon EBS root device volumes are automatically deleted when the instance terminates. However, by default, any additional EBS volumes that you attach at launch, or any EBS volumes that you attach to an

existing instance persist even after the instance terminates. This behavior is controlled by the volume's `DeleteOnTermination` attribute, which you can modify.

You can prevent an instance from being terminated accidentally by someone using the AWS Management Console, the CLI, and the API. This feature is available for both Amazon EC2 instance store-backed and Amazon EBS-backed instances. Each instance has a `DisableApiTermination` attribute with the default value of `false` (the instance can be terminated through Amazon EC2). You can modify this instance attribute while the instance is running or stopped (in the case of Amazon EBS-backed instances).

You can control whether an instance should stop or terminate when shutdown is initiated from the instance using an operating system command for system shutdown.

If you run a script on instance termination, your instance might have an abnormal termination, because we have no way to ensure that shutdown scripts run. Amazon EC2 attempts to shut an instance down cleanly and run any system shutdown scripts; however, certain events (such as hardware failure) may prevent these system shutdown scripts from running.

Relational Database Service (RDS)

What Is Amazon Relational Database Service (Amazon RDS)?

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizeable capacity for an industry-standard relational database and manages common database administration tasks.

Topics

Why would you want a managed relational database service? Because Amazon RDS takes over many of the difficult or tedious management tasks of a relational database.

- When you buy a server, you get CPU, memory, storage, and IOPS, all bundled together. With Amazon RDS, these are split apart so that you can scale them independently. So, for example, if you need more CPU, less IOPS, or more storage, you can easily allocate them.
- Amazon RDS manages backups, software patching, automatic failure detection, and recovery.
- In order to deliver a managed service experience, Amazon RDS does not provide shell access to DB instances, and it restricts access to certain system procedures and tables that require advanced privileges.
- You can have automated backups performed when you need them, or create your own backup snapshot. These backups can be used to restore a database, and the Amazon RDS restore process works reliably and efficiently.
- You can get high availability with a primary instance and a synchronous secondary instance that you can failover to when problems occur. You can also use MySQL, MariaDB, or PostgreSQL Read Replicas to increase read scaling.
- You can use the database products you are already familiar with: MySQL, MariaDB, PostgreSQL, Oracle, Microsoft SQL Server, and the new, MySQL-compatible Amazon Aurora DB engine.
- In addition to the security in your database package, you can help control who can access your RDS databases by using AWS IAM to define users and permissions. You can also help protect your databases by putting them in a virtual private cloud.
-

To begin learning more:

- If you are new to RDS but you are familiar with other Amazon Web Services, start with an introduction to the Amazon RDS Components. This section discusses the key components of Amazon RDS and how they map to those that you currently work with on your local network.
- For an overview of all AWS products, see What is Cloud Computing?
- Amazon Web Services provides a number of database services. For guidance on which service is best for your environment, see Running Databases on AWS

Amazon RDS Components

Security Groups

DB Instances

The basic building block of Amazon RDS is the DB instance. A DB instance is an isolated database environment in the cloud. A DB instance can contain multiple user-created databases, and you can access it by using the same tools and applications that you use with a stand-alone database instance. You can create and modify a DB instance by using the Amazon AWS command line interface, the Amazon RDS API, or the AWS Management Console.

Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines. Each DB engine has its own supported features, and each version of a DB engine may include specific features. Additionally, each DB engine has a set of parameters in a DB parameter group that control the behavior of the databases that it manages.

The computation and memory capacity of a DB instance is determined by its DB instance class. You can select the DB instance that best meets your needs. If your needs change over time, you can change DB instances. For information about DB instance classes, see DB Instance Class. For pricing information on DB instance classes, go to the Pricing section of the Amazon Relational Database Service (Amazon RDS) product page.

For each DB instance, you can select from 5 GB to 6 TB of associated storage capacity. Each DB instance class has minimum and maximum storage requirements for the DB instances that are created from it. It's important to have sufficient storage so that your databases have room to grow and that features for the DB engine have room to write content or log entries.

DB instance storage comes in three types: Magnetic, General Purpose (SSD), and Provisioned IOPS (SSD). They differ in performance characteristics and price, allowing you to tailor your storage performance and cost to the needs of your database. For a complete discussion of the different volume types, see the topic Amazon EBS Volume Types.

You can run a DB instance on a virtual private cloud using Amazon's Virtual Private Cloud (VPC) service. When you use a virtual private cloud, you have control over your virtual networking environment: you can select your own IP address range, create subnets, and configure routing and access control lists. The basic functionality of Amazon RDS is the same whether it is running in a VPC or not; Amazon RDS manages backups, software patching, automatic failure detection, and recovery. There is no additional cost to run your DB instance in a VPC. For more information on VPC and RDS, see Virtual Private Clouds (VPCs) and Amazon RDS.

Regions and Availability Zones

Amazon cloud computing resources are housed in highly available data center facilities in different areas of the world (for example, North America, Europe, or Asia). Each data center location is called a region.
Each region contains multiple distinct locations called Availability Zones, or AZs. Each Availability Zone is engineered to be isolated from failures in other Availability Zones, and to provide inexpensive, low-latency network connectivity to other Availability Zones in the same region. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location. For a list of regions and Availability Zones, see Regions and Availability Zones.

You can run your DB instance in several Availability Zones, an option called a Multi-AZ deployment. When you select this option, Amazon automatically provisions and maintains a synchronous standby replica of your DB instance in a different Availability Zone. The primary DB instance is synchronously replicated across Availability Zones to the standby replica to provide data redundancy, failover support, eliminate I/O freezes, and minimize latency spikes during system backups.

Security Groups
A security group controls the access to a DB instance. It does so by allowing access to IP address ranges or Amazon EC2 instances that you specify.
Amazon RDS uses DB security groups, VPC security groups, and EC2 security groups. In simple terms, a DB security group controls access to a DB instance that is not in a VPC, a VPC security group controls access to a DB instance inside a VPC, and an Amazon EC2 security group controls access to an EC2 instance and can be used with a DB instance.

Available RDS Interfaces
Topics
Amazon RDS Console
Command Line Interface
Programmatic Interfaces

There are several ways that you can interact with Amazon RDS.

Amazon RDS Console
The Amazon RDS console is a simple web-based user interface. From the console, you can perform almost all tasks you need to do from the RDS console with no programming required. To access the Amazon RDS console, sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.

Command Line Interface
Amazon AWS provides a command line interface that gives you access to much of the functionality that is available in the Amazon RDS API. For more information, see the AWS Command Line Interface Documentation and AWS Command Line Reference for Amazon RDS.

Programmatic Interfaces
The following table lists the resources that you can use to access Amazon RDS programmatically.

Resource	Description
AWS SDKs	The AWS SDKs include sample code, libraries, tools, documentation, and templates. To download the AWS SDKs, go to AWS Software Development Kits (SDKs).
Libraries	AWS provides libraries, sample code, tutorials, and other resources for software developers who prefer to build applications using language-specific APIs instead of Amazon Relational Database Service's SOAP and Query APIs. These libraries provide basic functions (not included in Amazon Relational Database Service's SOAP and Query APIs), such as request authentication, request retries, and error handling so you can get started more easily. Libraries and resources are available for the following languages: Java PHP Python Ruby Windows and .NET For libraries and sample code in all languages, see Sample Code & Libraries.

Amazon RDS API	If you prefer, you can code directly to the Amazon RDS API. For more information, see Amazon RDS Application Programming Interface (API), and see the Amazon Relational Database Service API Reference.

How You Are Charged for Amazon RDS

When you use Amazon RDS, you pay only for what you use, and there are no minimum or setup fees. You are billed according to the following criteria.

- Instance class – Pricing is based on the class (e.g., micro, small, large, xlarge) of the DB instance consumed.
- Running time – You are billed by the instance-hour, which is equivalent to a single instance running for an hour. For example, both a single instance running for two hours and two instances running for one hour consume 2 instance-hours. If a DB instance runs for only part of an hour, you are charged for a full instance-hour.
- Storage – The storage capacity that you have provisioned to your DB instance is billed per GB per month. If you scale your provisioned storage capacity within the month, your bill will be pro-rated.
- I/O requests per month – Total number of storage I/O requests that you have made in a billing cycle.
- Backup storage – Backup storage is the storage that is associated with automated database backups and any active database snapshots that you have taken. Increasing your backup retention period or taking additional database snapshots increases the backup storage consumed by your database. Amazon RDS provides backup storage up to 100% of your provisioned database storage at no additional charge. For example, if you have 10 GB-months of provisioned database storage, we will provide 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 you don't keep multiple backups, 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.

In addition to regular RDS pricing, you can purchase reserved DB instances. Reserved DB instances let you make a one-time up-front payment for a DB instance and reserve the DB instance for a one- or three-year term at significantly lower rates.

For Amazon RDS pricing information, see the Amazon RDS product page.

Monitoring an Amazon RDS DB Instance

There are several ways that you can track the performance and health of a DB instance. You can use the free Amazon CloudWatch service to monitor the performance and health of a DB instance; performance charts are shown in the Amazon RDS console. You can subscribe to Amazon RDS events to be notified when changes occur with a DB instance, DB Snapshot, DB parameter group, or DB security group.

What's Next?

This section introduced you to the basic infrastructure components that RDS offers. What should you do next?

Getting Started

Create a DB instance using instructions in the Getting Started with Amazon RDS section.

Database Engine Specific Topics

You can review information specific to a particular DB engine in the following sections:

Oracle on Amazon RDS
MySQL on Amazon RDS
Microsoft SQL Server on Amazon RDS
PostgreSQL on Amazon RDS

Aurora on Amazon RDS
MariaDB on Amazon RDS

Rebooting a DB Instance

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RebootInstance.html

In some cases, if you modify a DB instance, change the DB parameter group associated with the instance, or change a static DB parameter in a parameter group the instances uses, you must reboot the instance for the changes to take effect.

Rebooting a DB instance restarts the database engine service. A reboot also applies to the DB instance any modifications to the associated DB parameter group that were pending. Rebooting a DB instance results in a momentary outage of the instance, during which the DB instance status is set to rebooting. If the Amazon RDS instance is configured for MultiAZ, it is possible that the reboot will be conducted through a failover. An Amazon RDS event is created when the reboot is completed.

If your DB instance is a Multi-AZ deployment, you can force a failover from one availability zone to another when you select the **Reboot** option. When you force a failover of your DB instance, Amazon RDS automatically switches to a standby replica in another Availability Zone and updates the DNS record for the DB instance to point to the standby DB instance. As a result, you will need to clean up and re-establish any existing connections to your DB instance. **Reboot with failover** is beneficial when you want to simulate a failure of a DB instance for testing, or restore operations to the original AZ after a failover occurs. For more information, see High Availability (Multi-AZ).

The time required to reboot is a function of the specific database engine's crash recovery process. To improve the reboot time, we recommend that you reduce database activities as much as possible during the reboot process to reduce rollback activity for in-transit transactions.

In the console, the **Reboot** option may be disabled if the DB instance is not in the "Available" state. This can be due to several reasons, such as an in-progress backup or a customer-requested modification or a maintenance-window action.

This page intentionally left blank.

Lesson Summary

Amazon Global Infrastructure

AWS Global Infrastructure
- 2016 - 14 Regions & 38 Availability Zones
- 2017 - 4 more Regions & 11 More Availability Zones

A Region is a geographical area
- Each region consists of 2 (or more) Availability Zones

An Availability Zone (AZ) is simply a Data Center

Edge Locations are CDN End Points for CloudFront
- There are many more Edge Locations then Regions
- Currently there are over 66 Edge Locations

Amazon Web Services Overview

Know Following Services:
- S3
- CloudFront
- EC2
- RDS

Simple Storage Service (S3)

S3 Buckets

Object based storage only (for files)

Buckets are created in Regions.
Bucket Names must be unique across all of AWS
Bucket Names have naming rules. (lowercase only, etc.)

Bucket and Objects have permissions. (Ex. make object public)

Buckets can Host Static Websites
There are URL links for both S3 Bucket Object and S3 Bucket Static Website

S3 Core Fundamentals
- Key (name)
- Value (data)

Understand Differences between a link to an S3 bucket, and a link to a website hosted on S3
Different Names
- S3 Bucket: https://s3.amazonaws.com/<Bucket-Name>/
- S3 Website: http://<Bucket-Name>.s3-website-us-east-1.amazonaws.com

Bucket uses https, S3 Website uses http
- However, it can be achieved for website using cloud front CDN and SSL

CloudFront

CloudFront Distribution Network (CDN)

Amazon CloudFront is a web service that speeds up distribution of your static and dynamic web content to end users. CloudFront delivers your content through a worldwide network of edge locations.

Elastic Compute Cloud (EC2)

EC2 Instances

An instance is a virtual server in the AWS cloud
EC2 Instances are created by Region.

You launch an instance from an **Amazon Machine Image (AMI)**. The AMI provides the operating system, application server, and applications for your instance.

Instances come in different **Instance Type** (Famliy, Size, vCPUs, Memory, etc)

Instances Details can be modified (Number of Instances, Network, Subnet, etc)
Instance Storage can be modified (Additional volumes added)

Instances can be Tagged with identifying data
Instance use Security Groups to control traffic to/from an EC2 Instance

We require a Key Pair to access the Instances

Instances can be stopped, restarted, and terminated

Relational Database Service (RDS)

RDS Instances

RDS Instance Types
- SQL Server
- Oracle
- MySQL Server
- PostgreSQL
- Aurora
- MariaDB

We used MySQL because it is free tier

DB Instances can be created, modified, rebooted, and deleted.

When deleting a DB Instance, you will be prompted to save a final snapshot.
Otherwise, your data will be deleted.