Modular Architecture for Amazon Aurora PostgreSQL

Quick Start Reference Deployment

May 2019
Last update: June 2019 (revisions)

Tony Vattathil and Arabinda Pani
Amazon Web Services

Visit our GitHub repository for source files and to post feedback, report bugs, or submit feature ideas for this Quick Start.

Contents
Overview........................................................................................................................................2
Amazon Aurora .................................................................................................................................2
Cost and licenses .............................................................................................................................3
Architecture....................................................................................................................................4
Planning the deployment ....................................................................................................................5
Specialized knowledge .....................................................................................................................5
AWS account .................................................................................................................................5
Technical requirements ....................................................................................................................6
Deployment options ..........................................................................................................................7
Requirements for deploying Aurora into an existing VPC .............................................................7
Deployment steps .............................................................................................................................7
Step 1. Sign in to your AWS account .............................................................................................7
Step 2. Launch the Quick Start .......................................................................................................8
  Option 1: Parameters for deploying Aurora PostgreSQL into a new VPC ...............................9
  Option 2: Parameters for deploying Aurora PostgreSQL an existing VPC ..........................12
Step 3. Test the deployment ........................................................................................................ 15

Best practices for using Aurora ................................................................................................. 17

Use AWS CloudFormation for ongoing management .............................................................. 17

Security .................................................................................................................................... 18

FAQ .......................................................................................................................................... 18

Send us feedback .................................................................................................................. 19

Additional resources ................................................................................................................ 19

Document revisions ................................................................................................................ 19

This Quick Start was created by solutions architects and database specialists at Amazon
Web Services (AWS).

Quick Starts are automated reference deployments that use AWS CloudFormation
templates to deploy key technologies on AWS, following AWS best practices.

Overview

This Quick Start reference deployment guide provides step-by-step instructions for
deploying an Amazon Aurora PostgreSQL database on the AWS Cloud with high availability
and failover support.

This Quick Start is for users who are looking for a repeatable, customizable reference
deployment for Aurora PostgreSQL using AWS CloudFormation. You can also use the Quick
Start as a building block in your own automated deployments by using the existing-VPC
template that we’ve provided. This template will build only the Aurora
database components and is designed to plug into your existing AWS CloudFormation
templates as a drop-in PostgreSQL database replacement.

Amazon Aurora

Amazon Aurora is a fully managed relational database engine that’s compatible with
MySQL and PostgreSQL. MySQL and PostgreSQL combine the speed and reliability of
high-end commercial databases with the simplicity and cost-effectiveness of open-source
databases. The code, tools, and applications you use today with your existing MySQL and
PostgreSQL databases can be used with Aurora. With some workloads, Aurora can deliver
up to five times the throughput of MySQL and up to three times the throughput of PostgreSQL without requiring changes to most of your existing applications.

Aurora includes a high-performance storage subsystem. Its database engines are customized to take advantage of that fast distributed storage. The underlying storage grows automatically as needed, up to 64 terabytes. Aurora also automates and standardizes database clustering and replication, which are typically among the most challenging aspects of database configuration and administration.

Aurora is part of the managed database service Amazon Relational Database Service (Amazon RDS). Amazon RDS is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. If you are not already familiar with Amazon RDS, see the Amazon Relational Database Service User Guide.

You can choose Aurora as a DB engine option when setting up new database servers through Amazon RDS. Aurora provides the following features:

- Aurora takes advantage of the Amazon RDS features for management and administration. It uses the Amazon RDS AWS Management Console interface, AWS Command Line Interface (AWS CLI) commands, and API operations to handle routine database tasks such as provisioning, patching, backup, recovery, failure detection, and repair.

- Aurora management operations typically involve entire clusters of database servers that are synchronized through replication, instead of individual database instances. The automatic clustering, replication, and storage allocation make it simple and cost-effective to set up, operate, and scale your largest PostgreSQL deployments.

- You can bring data from Amazon RDS for PostgreSQL into Aurora by creating and restoring snapshots, or by setting up one-way replication. You can use push-button migration tools to convert your existing Amazon RDS for PostgreSQL applications to Aurora.

**Cost and licenses**

You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using the Quick Start.

The AWS CloudFormation template for this Quick Start includes configuration parameters that you can customize. Some of these settings, such as database instance type, will affect the cost of deployment. For cost estimates, see the pricing pages for each AWS service you will be using. Prices are subject to change.
**Tip** After you deploy the Quick Start, we recommend that you enable the AWS Cost and Usage Report to track costs associated with the Quick Start. This report delivers billing metrics to an S3 bucket in your account. It provides cost estimates based on usage throughout each month and finalizes the data at the end of the month. For more information about the report, see the AWS documentation.

**Architecture**

Deploying this Quick Start for a new virtual private cloud (VPC) with default parameters builds the following Aurora environment in the AWS Cloud.

![Architecture Diagram](image-url)

*Figure 1: Quick Start architecture for Aurora on AWS*
The Quick Start sets up the following:

- A highly available architecture that spans two Availability Zones.*
- A VPC configured with public and private subnets according to AWS best practices, to provide you with your own virtual network on AWS.*
- In the public subnets, managed network address translation (NAT) gateways to allow outbound internet access for resources in the private subnets.*
- In the public subnets, an optional Linux bastion host in an Auto Scaling group to allow inbound Secure Shell (SSH) access to Amazon Elastic Compute Cloud (Amazon EC2) instances in the private subnets.*
- In the private subnets, a **Aurora** DB cluster that includes two DB readers and one DB writer.
- An encryption key using AWS Key Management Service (AWS KMS). The Quick Start uses the key to enable encryption at rest for the Aurora DB cluster.
- An Amazon CloudWatch alarm to monitor the CPU on the bastion host and send an Amazon Simple Notification Service (Amazon SNS) notification when the alarm is triggered.

* The template that deploys the Quick Start into an existing VPC skips the components marked by asterisks and prompts you for your existing VPC configuration.

**Planning the deployment**

**Specialized knowledge**

This Quick Start assumes familiarity with database concepts and usage.

This deployment guide also requires a moderate level of familiarity with AWS services. If you’re new to AWS, visit the [Getting Started Resource Center](https://aws.amazon.com) and the [AWS Training and Certification website](https://aws.amazon.com/training/certification) for materials and programs that can help you develop the skills to design, deploy, and operate your infrastructure and applications on the AWS Cloud. For more information about AWS services used in this Quick Start, see the [Additional resources](#) section.

**AWS account**

If you don’t already have an AWS account, create one at [https://aws.amazon.com](https://aws.amazon.com) by following the on-screen instructions. Part of the sign-up process involves receiving a phone call and entering a PIN using the phone keypad.
Your AWS account is automatically signed up for all AWS services. You are charged only for the services you use.

**Technical requirements**

Before you launch the Quick Start, your account must be configured as specified in the following table. Otherwise, deployment might fail.

<table>
<thead>
<tr>
<th>Resources</th>
<th>If necessary, request service limit increases for the following resources. You might need to do this if you already have an existing deployment that uses these resources, and you think you might exceed the default limits with this deployment. For default limits, see the AWS documentation. AWS Trusted Advisor offers a service limits check that displays your usage and limits for some aspects of some services.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Resource</strong></td>
<td><strong>This deployment uses (default configuration)</strong></td>
</tr>
<tr>
<td>VPCs</td>
<td>1</td>
</tr>
<tr>
<td>VPC security groups</td>
<td>3</td>
</tr>
<tr>
<td>IAM roles</td>
<td>9</td>
</tr>
<tr>
<td>Auto Scaling groups</td>
<td>2</td>
</tr>
<tr>
<td>t2.medium instances</td>
<td>3</td>
</tr>
<tr>
<td>t2.micro instances</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Regions</th>
<th>Aurora with PostgreSQL compatibility isn’t currently supported in all AWS Regions. For a current list of supported regions, see AWS Regions and Endpoints in the AWS documentation.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Key pair</th>
<th>Make sure that at least one Amazon EC2 key pair exists in your AWS account in the region where you are planning to deploy the Quick Start. Make note of the key pair name. You’ll be prompted for this information during deployment. To create a key pair, follow the instructions in the AWS documentation. If you’re deploying the Quick Start for testing or proof-of-concept purposes, we recommend that you create a new key pair instead of specifying a key pair that’s already being used by a production instance.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Amazon S3 URLs</th>
<th>If you’re copying the templates to your own S3 bucket for deployment, make sure that you update the QSS3Bucket and QSS3Prefix parameters to reflect the location of the files in your bucket. Otherwise, deployment may fail or behave unexpectedly.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>IAM permissions</th>
<th>To deploy the Quick Start, you must log in to the AWS Management Console with IAM permissions for the resources and actions the templates will deploy. The AdministratorAccess managed policy within IAM provides sufficient permissions, although your organization may choose to use a custom policy with more restrictions.</th>
</tr>
</thead>
</table>
Deployment options

This Quick Start provides two deployment options:

- **Deploy Aurora PostgreSQL into a new VPC (end-to-end deployment).** This option builds a new AWS environment consisting of the VPC, subnets, NAT gateways, security groups, and an optional Linux bastion host, and then deploys Aurora into this new VPC.

- **Deploy Aurora PostgreSQL into an existing VPC.** This option provisions Aurora in your existing AWS infrastructure. Use this option to plug Aurora PostgreSQL into your existing AWS CloudFormation templates as a drop-in PostgreSQL database replacement.

Requirements for deploying Aurora into an existing VPC

If you’re planning to deploy Aurora into an existing VPC, check the following:

- Your VPC must have two private subnets in different Availability Zones for the database instances. This Quick Start doesn’t support shared subnets.

- The private subnets require **NAT gateways** in their route tables, to allow the instances to download packages and software without exposing them to the internet.

- The domain name option must be configured in the DHCP options as explained in the Amazon VPC documentation.

- You’ll need a Linux bastion host that can be used to run the test procedure discussed in **step 3**. To install the bastion host, you can launch the **AWS CloudFormation template** we’ve provided into your existing subnet.

You’ll be prompted for your VPC settings when you launch the Quick Start.

If you don’t have an existing VPC that satisfies these requirements, we recommend that you use the **new VPC** deployment option, so the Quick Start will build this VPC for you in addition to deploying Aurora into the VPC.

Deployment steps

**Step 1. Sign in to your AWS account**

1. Sign in to your AWS account at [https://aws.amazon.com](https://aws.amazon.com) with an IAM user role that has the necessary permissions. For details, see **Planning the deployment** earlier in this guide.
2. Make sure that your AWS account is configured correctly, as discussed in the Technical requirements section.

3. Use the region selector in the navigation bar to choose the AWS Region where you want to deploy Aurora.

   **Note**  Aurora with PostgreSQL compatibility isn’t currently supported in all AWS Regions. For a current list of supported regions, see the AWS Regions and Endpoints webpage.

**Step 2. Launch the Quick Start**

   **Notes**  The instructions in this section reflect the older version of the AWS CloudFormation console. If you’re using the redesigned console, some of the user interface elements might be different.

   You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using this Quick Start. For full details, see the pricing pages for each AWS service you will be using in this Quick Start. Prices are subject to change.

1. Sign in to your AWS account, and choose one of the following options to launch the AWS CloudFormation template. For help choosing an option, see deployment options earlier in this guide.

   - **Deploy Aurora PostgreSQL into a new VPC on AWS**
   - **Deploy Aurora PostgreSQL into an existing VPC on AWS**

   Each deployment takes about 25 minutes to complete.

2. On the Select Template page, keep the default setting for the template URL, and then choose Next.

3. On the Specify Details page, change the stack name if needed. Review the parameters for the template. Provide values for the parameters that require input. For all other parameters, review the default settings and customize them as necessary.
In the following tables, parameters are listed by category and described separately for the two deployment options:

- **Parameters for deploying Aurora PostgreSQL into a new VPC**
- **Parameters for deploying Aurora PostgreSQL into an existing VPC**

When you finish reviewing and customizing the parameters, choose **Next**.

**OPTION 1: PARAMETERS FOR DEPLOYING AURORA POSTGRESQL INTO A NEW VPC**

**View template**

**Network configuration:**

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key name (KeyPairName)</td>
<td>Requires input</td>
<td>A public/private key pair, which allows you to connect securely to your instance after it launches. This is the key pair you created in your preferred region; see the Technical requirements section.</td>
</tr>
<tr>
<td>Availability Zones (Availability Zones)</td>
<td>Requires input</td>
<td>The list of Availability Zones to use for the subnets in the VPC. The Quick Start uses two Availability Zones from your list and preserves the logical order you specify.</td>
</tr>
<tr>
<td>VPC CIDR (VPCCIDR)</td>
<td>10.0.0.0/16</td>
<td>The CIDR block for the VPC.</td>
</tr>
<tr>
<td>Private subnet 1 CIDR (PrivateSubnet1CIDR)</td>
<td>10.0.0.0/19</td>
<td>The CIDR block for the private subnet located in Availability Zone 1.</td>
</tr>
<tr>
<td>Private subnet 2 CIDR (PrivateSubnet2CIDR)</td>
<td>10.0.32.0/19</td>
<td>The CIDR block for the private subnet located in Availability Zone 2.</td>
</tr>
<tr>
<td>Public subnet 1 CIDR (PublicSubnet1CIDR)</td>
<td>10.0.128.0/20</td>
<td>The CIDR block for the public subnet located in Availability Zone 1.</td>
</tr>
<tr>
<td>Public subnet 2 CIDR (PublicSubnet2CIDR)</td>
<td>10.0.144.0/20</td>
<td>The CIDR block for the public subnet located in Availability Zone 2.</td>
</tr>
</tbody>
</table>

**Linux bastion configuration:**

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create bastion stack (EnableBastion)</td>
<td>true</td>
<td>Set this parameter to <strong>false</strong> if you don’t want to provision a Linux bastion host instance. This entry point is typically used for debugging. If you don’t need SSH access, set this parameter to <strong>false</strong>.</td>
</tr>
<tr>
<td>Permitted IP range (RemoteAccessCIDR)</td>
<td>Requires input</td>
<td>The CIDR IP range that is permitted to access the Aurora database instances. We recommend that you set this value to a</td>
</tr>
</tbody>
</table>
trusted IP range. For example, you might want to grant only your corporate network access to the software. Note that the setting 0.0.0.0/0 will allow full public access to the database.

**Database configuration:**

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Database name</strong> (DBName)</td>
<td>AuroraPostgresDB</td>
<td>The name of the Aurora DB to provision. This is an alphanumeric string of 5-64 characters.</td>
</tr>
<tr>
<td><strong>Database auto minor version upgrade</strong> (DBAutoMinorVersion Upgrade)</td>
<td>false</td>
<td>Set this parameter to true if you want to enable your DB instances to receive minor DB engine version upgrades automatically when upgrades become available.</td>
</tr>
<tr>
<td><strong>Database backup retention period</strong> (DBBackupRetention Period)</td>
<td>35</td>
<td>The number of days to retain automatic database snapshots. To disable automatic backups, set this parameter to 0. For more information, see Working with Backups in the AWS documentation.</td>
</tr>
<tr>
<td><strong>Database Engine Version</strong> (DBEngineVersion)</td>
<td>9.6.9</td>
<td>The version of the database engine.</td>
</tr>
<tr>
<td><strong>Database instance class</strong> (DBInstanceClass)</td>
<td>db.r4.large</td>
<td>The DB (compute and memory capacity) class for the database instances. For more information, see Choosing the DB Instance Class in the AWS documentation.</td>
</tr>
<tr>
<td><strong>Database master username</strong> (DBMasterUsername)</td>
<td>pgadmin</td>
<td>The user name for the database administrator account. This is an alphanumeric string of 1-16 characters. The user name must start with an uppercase or lowercase letter (A-Z, a-z).</td>
</tr>
<tr>
<td><strong>Database master password</strong> (DBMasterUserPassword)</td>
<td><strong>Requires input</strong></td>
<td>The password for the database administrator account (8-64 character string).</td>
</tr>
<tr>
<td><strong>Database port</strong> (DBPort)</td>
<td>5432</td>
<td>The port that you want to access the database through. The DB instance will listen on this port for connections. This value must be in the range 1115-65535.</td>
</tr>
<tr>
<td><strong>Database encryption enabled</strong> (DBAllocatedStorage Encrypted)</td>
<td>true</td>
<td>Set this parameter to false if you don’t want to encrypt the database at rest. For more information, see Encrypting Amazon RDS Resources in the AWS documentation.</td>
</tr>
<tr>
<td><strong>Multi-AZ deployment</strong> (DBMultiAZ)</td>
<td>true</td>
<td>Set this parameter to false if you want to deploy the database instances in a single Availability Zone. You might want to disable Multi-AZ deployment if you’re testing or running continuous integration (CI) processes.</td>
</tr>
</tbody>
</table>
### Parameter label (name) | Default | Description
--- | --- | ---
SNS notification email (NotificationList) | db-ops @domain.com | The email that is used to configure an SNS topic for sending CloudWatch alarms and Amazon RDS event notifications. This must be a valid email address.

#### Database tags (optional):

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
</table>
| Environment stage (EnvironmentStage) | none | The environment stage of the Aurora PostgreSQL DB cluster. The five options are:
- Development (dev)
- Testing (test)
- Pre-production (pre-prod)
- Production (prod)
- Not specified (none) |
| Application name (Application) | Optional | The business application that this deployment supports. |
| Application version (ApplicationVersion) | Optional | The version of the business application that this deployment supports. |
| Project cost center (ProjectCostCenter) | Optional | The cost center associated with the project. |
| Confidentiality classifier (Confidentiality) | Optional | The confidentiality classification of the data stored in the Aurora DB. The five options are:
- Public information (public)
- Private information (private)
- Confidential information (confidential)
- Personally identifiable information or protected health information (pii/phi)
- Not specified (none) |
| Compliance classifier (Compliance) | Optional | The compliance level for the data stored in the Aurora DB. The four options are:
- Health Insurance Portability and Accountability Act, or HIPAA (hipaa)
- Sarbanes-Oxley Act, or SOX (sox)
- Federal Information Processing Standards, or FIPS (fips)
- Other (other) |
**AWS Quick Start configuration:**

**Note** We recommend that you keep the default settings for the following two parameters, unless you are customizing the Quick Start templates for your own projects. Changing these settings will automatically update code references to point to a new Quick Start location. For details, see the [AWS Quick Start Contributor’s Guide](#).

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quick Start S3 bucket name</td>
<td>aws-quickstart</td>
<td>The S3 bucket you created for your copy of Quick Start assets, if you decide to customize or extend the Quick Start for your own use. The bucket name can include numbers, lowercase letters, uppercase letters, and hyphens, but should not start or end with a hyphen.</td>
</tr>
<tr>
<td>Quick Start S3 key prefix</td>
<td>quickstart-amazon-aurora/</td>
<td>The S3 key name prefix used to simulate a folder for your copy of Quick Start assets, if you decide to customize or extend the Quick Start for your own use. This prefix can include numbers, lowercase letters, uppercase letters, hyphens, and forward slashes.</td>
</tr>
</tbody>
</table>

**OPTION 2: PARAMETERS FOR DEPLOYING Aurora PostgreSQL AN EXISTING VPC**

*View template*

**Network configuration:**

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPC ID</td>
<td><em>Requires input</em></td>
<td>The ID of your existing VPC (e.g., vpc-0343606e) where you want to deploy the Aurora database.</td>
</tr>
<tr>
<td>Private subnet 1 ID</td>
<td><em>Requires input</em></td>
<td>The ID of the private subnet in Availability Zone 1 in your existing VPC (e.g., subnet-a0246dc0).</td>
</tr>
<tr>
<td>Private subnet 2 ID</td>
<td><em>Requires input</em></td>
<td>The ID of the private subnet in Availability Zone 2 in your existing VPC (e.g., subnet-b58c3d67).</td>
</tr>
<tr>
<td>Custom security group ID</td>
<td><em>Optional</em></td>
<td>The ID of the custom security group you want to use in your existing VPC (e.g., sg-7f16e910). If you leave this parameter blank, the Quick Start will create a security group.</td>
</tr>
</tbody>
</table>

**Database configuration:**

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database name</td>
<td>AuroraPostgresDB</td>
<td>The name of the Aurora DB to provision. This is an alphanumeric string of 5-64 characters.</td>
</tr>
<tr>
<td>Parameter label (name)</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Database auto minor version upgrade</td>
<td>false</td>
<td>Set this parameter to <strong>true</strong> if you want to enable your DB instances to receive minor DB engine version upgrades automatically when upgrades become available.</td>
</tr>
<tr>
<td>Database backup retention period</td>
<td>35</td>
<td>The number of days to retain automatic database snapshots. To disable automatic backups, set this parameter to 0. For more information, see <a href="#">Working with Backups</a> in the AWS documentation.</td>
</tr>
<tr>
<td>Database Engine Version</td>
<td>9.6.9</td>
<td>The version of the database engine.</td>
</tr>
<tr>
<td>Database instance class</td>
<td>db.r4.large</td>
<td>The DB (compute and memory capacity) class for the database instances. For more information, see <a href="#">Choosing the DB Instance Class</a> in the AWS documentation.</td>
</tr>
<tr>
<td>Database master username</td>
<td>pgadmin</td>
<td>The user name for the database administrator account. This is an alphanumeric string of 1-16 characters. The user name must start with an uppercase or lowercase letter (A-Z, a-z).</td>
</tr>
<tr>
<td>Database master password</td>
<td><strong>Requires input</strong></td>
<td>The password for the database administrator account (8-64 character string).</td>
</tr>
<tr>
<td>Database port</td>
<td>5432</td>
<td>The port that you want to access the database through. The DB instance will listen on this port for connections. This value must be in the range 1115-65535.</td>
</tr>
<tr>
<td>Database connection CIDR</td>
<td>10.0.0.0/16</td>
<td>The CIDR block allowed for external access. We recommend that you specify your VPC CIDR.</td>
</tr>
<tr>
<td>Multi-AZ deployment</td>
<td>true</td>
<td>Set this parameter to <strong>false</strong> if you want to deploy the database instances in a single Availability Zone. You might want to disable Multi-AZ deployment if you’re testing or running continuous integration (CI) processes.</td>
</tr>
<tr>
<td>Database encryption enabled</td>
<td>true</td>
<td>Set this parameter to <strong>false</strong> if you don’t want to encrypt the database at rest. For more information, see <a href="#">Encrypting Amazon RDS Resources</a> in the AWS documentation.</td>
</tr>
<tr>
<td>SNS notification email</td>
<td>db-ops</td>
<td>The email that is used to configure an SNS topic for sending CloudWatch alarms and Amazon RDS event notifications. This must be a valid email address.</td>
</tr>
</tbody>
</table>
Database tags (optional):

<table>
<thead>
<tr>
<th>Parameter label (name)</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
</table>
| Environment stage (EnvironmentStage) | none | The environment stage of the Aurora PostgreSQL DB cluster. The five options are:  
- Development (dev)  
- Testing (test)  
- Pre-production (pre-prod)  
- Production (prod)  
- Not specified (none) |
| Application name (Application) | Optional | The business application that this deployment supports. |
| Application version (ApplicationVersion) | Optional | The version of the business application that this deployment supports. |
| Project cost center (ProjectCostCenter) | Optional | The cost center associated with the project. |
| Confidentiality classifier (Confidentiality) | Optional | The confidentiality classification of the data stored in the Aurora DB. The five options are:  
- Public information (public)  
- Private information (private)  
- Confidential information (confidential)  
- Personally identifiable information or protected health information (pii/phi)  
- Not specified (none) |
| Compliance classifier (Compliance) | Optional | The compliance level for the data stored in the Aurora DB. The four options are:  
- Health Insurance Portability and Accountability Act, or HIPAA (hipaa)  
- Sarbanes-Oxley Act, or SOX (sox)  
- Federal Information Processing Standards, or FIPS (fips)  
- Other (other) |

4. On the Options page, you can specify tags (key-value pairs) for resources in your stack and set advanced options. When you’re done, choose Next.

5. On the Review page, review and confirm the template settings. Under Capabilities, select the two check boxes to acknowledge that the template will create IAM resources and that it might require the capability to auto-expand macros.

6. Choose Create to deploy the stack.

7. Monitor the status of the stack. When the status is CREATE_COMPLETE, the Aurora DB cluster is ready.
8. Use the URLs displayed in the **Resources** tab for the stack to view the resources that were created.

![Figure 2: Aurora resources after successful deployment](image)

**Step 3. Test the deployment**

**Note** This testing procedure assumes that you are using the new-VPC template (**deployment option 1**) with the **Create bastion stack** (**EnableBastion**) parameter enabled. If you use the existing-VPC template (**deployment option 2**), you also need to launch the [AWS CloudFormation template](#) for deploying a Linux bastion host into your existing subnet before proceeding.

1. Connect to the bastion host by using SSH with the key pair that you specified during deployment and the IP address that is displayed on the **Resources** tab of the AWS CloudFormation stack, as shown in Figure 3.

```
ssh ec2-user@3.216.187.227
```
2. To test the database installation, you need to install a PostgreSQL RPM package that provides the `pg_isready` command:

```
sudo yum install postgresql96 -y
```

3. From the AWS CloudFormation console, **Outputs** tab for `AuroraStack`, note the values of `DBName`, `RDSEndpointAddress`, and `RDSEndpointPort`, as shown in Figure 4.
4. To test that the PostgreSQL database is accepting connections, use the `pg_isready` command:

   ```
   pg_isready -d <DNName> -h <RDSEndpointAddress> -p <RDSEndpointPort>
   ```

5. If the database is ready, the command output will display **accepting connections**, as shown in Figure 5.

   ```
   [ec2-user@ip-10-0-157-129 ~]$ pg_isready -d AuroraPostgresDB -h tcat-tag-postgres-46bb5ba3-aurora-auroradbcluster-71rp5dercgws.cluster-ca3arvdgq9qr.us-east-1.rds.amazonaws.com -p 3306 | grep accepting
tcat-tag-postgres-46bb5ba3-aurora-auroradbcluster-71rp5dercgws.cluster-ca3arvdgq9qr.us-east-1.rds.amazonaws.com:3306 - accepting connections
[ec2-user@ip-10-0-157-129 ~]$
   ```

**Figure 5: Confirmation that the Aurora database is accepting connections**

**Best practices for using Aurora**

For detailed information about working with Aurora PostgreSQL, see the [Amazon Aurora User Guide](https://aws.amazonaws.com/).

**Use AWS CloudFormation for ongoing management**

We recommend using the AWS CloudFormation console at [https://console.aws.amazon.com/cloudformation/](https://console.aws.amazon.com/cloudformation/) to manage updates and deletions for the resources that are created by this Quick Start. If you use the Amazon EC2 console, the AWS Command Line Interface (AWS CLI), or API to change or delete resources created by this
Quick Start, future AWS CloudFormation operations on the stack might behave unexpectedly.

**Security**

The workload template in this reference deployment provides the ability to integrate custom security groups for added flexibility and ease of deployment into existing environments.

**FAQ**

**Q.** I encountered a CREATE_FAILED error when I launched the Quick Start.

**A.** If AWS CloudFormation fails to create the stack, we recommend that you relaunch the template with **Rollback on failure** set to **No**. (This setting is under **Advanced** in the AWS CloudFormation console, **Options** page.) With this setting, the stack’s state will be retained and the instance will be left running, so you can troubleshoot the issue.

| Important | When you set **Rollback on failure** to **No**, you will continue to incur AWS charges for this stack. Please make sure to delete the stack when you finish troubleshooting. |

For additional information, see [Troubleshooting AWS CloudFormation](https://aws.amazon.com) on the AWS website.

The following table lists specific CREATE_FAILED error messages you might encounter.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Possible cause</th>
<th>What to do</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance ID did not stabilize</td>
<td>You have exceeded your IOPS for the region.</td>
<td>Request a limit increase by completing the request form in the AWS Support Center.</td>
</tr>
</tbody>
</table>

**Q.** I encountered a size limitation error when I deployed the AWS CloudFormation templates.

**A.** We recommend that you launch the Quick Start templates from the links in this guide or from another S3 bucket. If you deploy the templates from a local copy on your computer or from a non-S3 location, you might encounter template size limitations when you create the stack. For more information about AWS CloudFormation limits, see the [AWS documentation](https://aws.amazon.com).
Send us feedback

To post feedback, submit feature ideas, or report bugs, use the Issues section of the GitHub repository for this Quick Start. If you’d like to submit code, please review the Quick Start Contributor’s Guide.

Additional resources

AWS resources

- [Getting Started Resource Center](#)
- [AWS General Reference](#)
- [AWS Glossary](#)

AWS services

- [Amazon Aurora](#)
- [AWS CloudFormation](#)
- [Amazon CloudWatch](#)
- [Amazon EC2](#)
- [AWS KMS](#)
- [Amazon SNS](#)
- [Amazon VPC](#)

Other Quick Start reference deployments

- [AWS Quick Start home page](#)

Document revisions

<table>
<thead>
<tr>
<th>Date</th>
<th>Change</th>
<th>In sections</th>
</tr>
</thead>
<tbody>
<tr>
<td>June 2019</td>
<td>Added a parameter DBEngineVersion to Database configuration tables</td>
<td>Launch the Quick Start: <a href="#">Option 1</a> and <a href="#">Option 2</a></td>
</tr>
<tr>
<td>May 2019</td>
<td>Initial publication</td>
<td>—</td>
</tr>
</tbody>
</table>