

# RDS



Ya hemos visto que el almacenamiento en la nube ofrece un gran número de ventajas. Otro de los productos estrella de la computación en la nube es el uso de bases de datos, ya sean distribuidas o no.

La principal ventaja de utilizar un servicio de base de datos basado en la nube es que no requieren de la administración por parte del usuario. Éste sólo utiliza el servicio sin necesidad de tener conocimientos avanzados sobre su administración. Estos servicios se conocen como **administrados**, ya que la propia plataforma *cloud* se encarga de gestionar el escalado, las copias de seguridad automáticas, la tolerancia a errores y la alta disponibilidad, y por tanto, estos servicios forman parte de una solución PaaS.

Si nosotros creásemos una instancia EC2 e instalásemos cualquier sistema gestor de base de datos, como *MariaDB* o *PostgreSQL*, seríamos responsables de varias tareas administrativas, como el mantenimiento del servidor y la huella energética, el software, la instalación, la implementación de parches y las copias de seguridad de la base de datos, así como de garantizar su alta disponibilidad, de planificar la escalabilidad y la seguridad de los datos, y de instalar el sistema operativo e instalarle los respectivos parches.

## Datos relacionales - Amazon RDS

AWS ofrece *Amazon RDS* (<https://aws.amazon.com/es/rds/>) como servicio administrado que configura y opera una base de datos relacional en la nube, de manera que como desarrolladores sólo hemos de enfocar nuestros esfuerzos en los datos y optimizar nuestras aplicaciones.

### Instancias de bases de datos

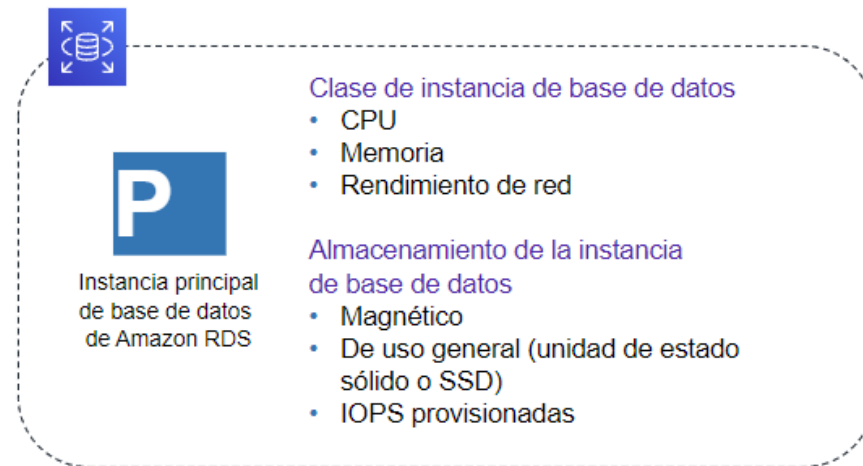
Una instancia de base de datos es un entorno de base de datos aislado que puede contener varias bases de datos creadas por el usuario. Se puede acceder a él utilizando las mismas herramientas y aplicaciones que utilizamos con una instancia de base de datos independiente.

Cuando vamos a crear una instancia de base de datos, primero hemos de indicar qué motor de base de datos ejecutar. Actualmente, RDS admite seis motores de bases de datos:

- [MariaDB](#), compatible con las versiones 10.3, 10.4, 10.5, 10.6 y 10.11.
- [MySQL](#), compatible con las versiones 5.7 y 8.0.
- [PostgreSQL](#), compatible con las versiones 11, 12, 13, 14, 15 y 16.
- [Amazon Aurora](#), con diferentes versiones compatibles con MySQL 5.7 y 8, o PostgreSQL 11, 12, 13, 14 y 15.
- [Microsoft SQL Server](#), que permite implementar varias versiones de SQL Server (2014, 2016, 2017 y 2019), incluidas las ediciones *Express*, *Web*, *Standard* y *Enterprise*.
- [IBM DB2](#), compatible con la versión 11, tanto en edición *Standard* como en la *Advanced*.
- y [Oracle](#), compatible con Oracle 19, con dos modelos de licencia diferentes: *Licencia incluida* y *Bring-Your-Own-License (BYOL)*.

Los recursos que se encuentran en una instancia de base de datos se definen en función de la clase de instancia de base de datos, y el tipo de almacenamiento se determina por el tipo de disco. Las instancias y el almacenamiento de base de datos difieren en cuanto a las características de rendimiento y al precio, lo que permite adaptar el coste y el rendimiento a las necesidades de nuestra base de datos.

## Amazon RDS









### Instancia de RDS

Por ejemplo, si seleccionamos el motor de *MariaDB*, podemos observar cómo mediante la creación sencilla nos ofrece tres propuestas de tamaño, dependiendo de si es para el entorno de producción, desarrollo y pruebas o el de la capa gratuita.

### Configuración

Tipo de motor [Información](#)

<input type="radio"/> Amazon Aurora 	<input type="radio"/> MySQL 	<input checked="" type="radio"/> MariaDB 
<input type="radio"/> PostgreSQL 	<input type="radio"/> Oracle 	<input type="radio"/> Microsoft SQL Server 

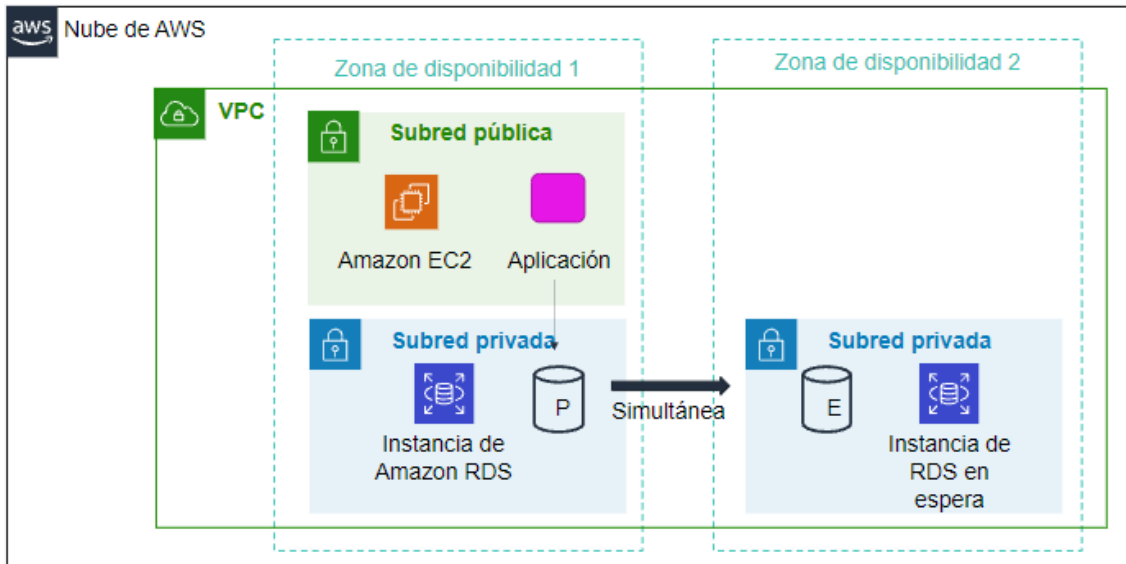
Tamaño de la instancia de base de datos

<input type="radio"/> Producción db.r6g.xlarge 4 vCPUs 32 GiB RAM 500 GiB 1.017 USD/hora	<input checked="" type="radio"/> Desarrollo y pruebas db.r6g.large 2 vCPUs 16 GiB RAM 100 GiB 0.231 USD/hora	<input type="radio"/> Capa gratuita db.t3.micro 2 vCPUs 1 GiB RAM 20 GiB 0.020 USD/hora
---	---	--

### Configuración de tamaño de la instancia con MariaDB

## Alta disponibilidad

Una de las características más importantes de *RDS* es la capacidad de configurar la instancia de base de datos para una alta disponibilidad con una [implementación Multi-AZ](#) (multizona). Al hacerlo, se genera de manera automática una **copia en espera** de la instancia de base de datos en otra zona de disponibilidad dentro de la misma VPC. Después de propagar la copia de la base de datos, las transacciones se replican de forma síncrona a la copia en espera.

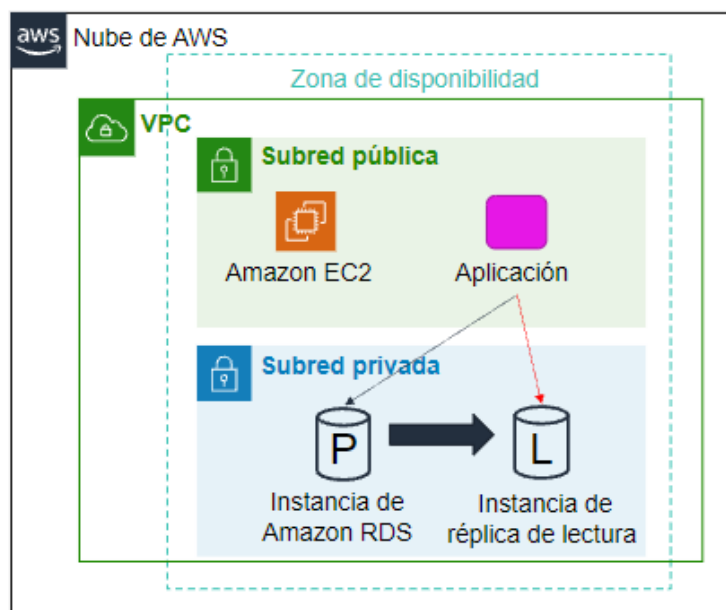


*Alta disponibilidad en Multi-AZ*

Por lo tanto, si la instancia de base de datos principal falla en una implementación Multi-AZ, RDS activa automáticamente la instancia de base de datos en espera como la nueva instancia principal.

### **Réplica de lectura**

RDS también admite la creación de [réplicas de lectura](#) para *MySQL*, *MariaDB*, *PostgreSQL* y *Amazon Aurora*.



*Réplica de lectura*

Las actualizaciones que se realizan en la instancia principal se copian de manera asíncrona en la instancia de réplica de lectura, de manera que direccionando las consultas a esta nueva réplica reduciremos la carga de la instancia principal.

Las réplicas de lectura también pueden convertirse en la instancia de base de datos principal, pero, debido a la replicación asíncrona, este proceso debe hacerse de forma manual.

Las réplicas de lectura pueden crearse en una región diferente a la utilizada por la base de datos principal, lo que puede mejorar la recuperación de desastres y/o disminuir la latencia al dirigir las lecturas a una réplica de lectura lo más cercana al usuario.

## Casos de uso

*RDS* es ideal para las aplicaciones web y móviles que necesitan una base de datos con alto rendimiento, enorme escalabilidad en el almacenamiento y alta disponibilidad.

Se recomienda *RDS* cuando nuestra aplicación necesite:

- Transacciones o consultas complejas.
- Tasa de consulta o escritura media a alta: hasta 30.000 IOPS (15.000 lecturas + 15.000 escrituras).
- No más de una única partición o nodo de trabajo.
- Alta durabilidad.

En cambio, no se recomienda cuando:

- Tasas de lectura o escritura muy grandes (por ejemplo, 150.000 escrituras por segundo).
- Fragmentación causada por el gran tamaño de los datos o las altas demandas de rendimiento.
- Solicitudes y consultas GET o PUT simples que una base de datos NoSQL puede manejar.
- Personalización del sistema de administración de bases de datos relacionales (en este caso, es mejor instalar por nuestra cuenta el SGBD que necesitemos en una instancia EC2).

## Costes

Claramente, el coste de *RDS* va a ser superior al coste de la infraestructura que utiliza. Aun así, hemos de tener en cuenta las ventajas de ahorrarnos el mantenimiento y las tareas asociadas que normalmente realiza un administrador de bases de datos (DBA).

Su [coste](#) se calcula en base al tiempo de ejecución (calculado en horas) así como las características de la base de datos, como pueden ser el motor, el [tipo de instancia](#) y su cantidad, así como la clase de memoria de la base de datos.

Otros gastos asociados son:

- almacenamiento aprovisionado: el almacenamiento para copias de seguridad de hasta el 100% del almacenamiento de nuestra base de datos activa es gratuito. Una vez que se termina la instancia de base de datos, el almacenamiento para copias de seguridad se factura por GB por mes.
- cantidad de solicitudes de entrada y de salida.

Aunque se recomienda utilizar la [calculadora de costes](#) para afinar en el presupuesto, por ejemplo, una base de datos con *MariaDB* con una instancia db.m4.large con 2 procesadores y 8GB de RAM, en una única AZ, con un porcentaje de utilización del 100% y 30GB para almacenar los datos, cuesta alrededor de 131\$ mensuales. En cambio si la cambiamos por dos instancias más potentes, como puede ser la db.m4.4xlarge, con 16 procesadores y 64 GB de RAM, en multi-AZ ya sube a unos 4.100\$ al mes.

Es importante recordar que si reservamos las instancias estos costes se reducirían en proporción a 2350\$ (reserva de un año) o 1526\$ (reserva de tres años).

## Hola RDS

A continuación realizaremos un ejemplo sencillo donde vamos a crear una base de datos. Para ello, crearemos una instancia de *MariaDB* y nos conectaremos desde *DBeaver*:

[RDS](#) > Crear base de datos

### Crear base de datos


**Elegir un método de creación de base de datos** [Información](#)


☒ **Creación estándar**  
Puede definir todas las opciones de configuración, incluidas las de disponibilidad, seguridad, copias de seguridad y mantenimiento.


☐ **Creación sencilla**  
Utilice las configuraciones recomendadas. Algunas opciones de configuración se pueden cambiar después de crear la base de datos.


**Opciones del motor**


Tipo de motor [Información](#)


☐ Aurora (MySQL Compatible)  



☐ Aurora (PostgreSQL Compatible)  



☐ MySQL  


☒ MariaDB  


☐ PostgreSQL  


☐ Oracle  


☐ Microsoft SQL Server  


☐ IBM Db2  


▼ Ocultar filtros

☐ **Mostrar versiones compatibles con las escrituras optimizadas de Amazon RDS** [Información](#)  
Las escrituras optimizadas de Amazon RDS mejoran el rendimiento de escritura hasta 2 veces sin costo adicional.

Versión del motor

MariaDB 10.6.14 ▼

### Creación de la BD en RDS

Para empezar, desde la consola de AWS, crearemos nuestra base de datos a la que llamaremos ***instituto***.

En nuestro caso hemos seguido la creación estándar con una plantilla de la capa gratuita (utiliza una instancia *db.t4g.micro*). Una vez configurado el usuario *admin* y la contraseña (al menos debe tener ocho caracteres), debemos configurar la conectividad.

### Configuración

**Identificador de instancias de bases de datos** [Información](#)

Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos pertenecientes a su cuenta de AWS en la región de AWS actual.

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en minúsculas (como en "miinstanciadebd"). Restricciones: de 1 a 60 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos. No puede terminar con un guion.

**Configuración de credenciales**

**Nombre de usuario maestro** [Información](#)

Escriba un ID de inicio de sesión para el usuario maestro de la instancia de base de datos.

De 1 a 16 caracteres alfanuméricos. El primer carácter debe ser una letra.

☐ Generación automática de contraseña

Amazon RDS puede generar una contraseña en su nombre, o bien puede especificar su propia contraseña.

**Contraseña maestra** [Información](#)

Restricciones: debe tener al menos 8 caracteres ASCII imprimibles. No puede contener ninguno de los siguientes caracteres: / (barra diagonal), ' (comillas simples), " (dobles comillas) y @ (signo de arroba).

**Confirmar contraseña** [Información](#)

**Configuración de la instancia**

Las opciones de configuración de la instancia de base de datos que aparecen a continuación están limitadas a las que admite el motor que ha seleccionado anteriormente.

**Clase de instancia de base de datos** [Información](#)

- ☐ Clases estándar (incluye clases m)
- ☐ Clases optimizadas para memoria (incluye clases r y x)
- ☒ Clases con ráfagas (incluye clases t)

2 vCPUs
1 GiB RAM
Red: 2085 Mbps

☐ Incluir clases de generación anterior

## Configuración de la conectividad en RDS

Usar el modelo de credenciales *Autoadministrado*

### Configuración de credenciales

#### Nombre de usuario maestro [Información](#)

Escriba un ID de inicio de sesión para el usuario maestro de la instancia de base de datos.

1 a 16 caracteres alfanuméricos. El primer carácter debe ser una letra.

#### Administración de credenciales

Puede usar AWS Secrets Manager o administrar sus credenciales de usuario maestro.

☐ **Administrado en AWS Secrets Manager - *más seguro***

RDS genera una contraseña y la administra durante todo su ciclo de vida mediante AWS Secrets Manager.

☒ **Autoadministrado**

Cree su propia contraseña o pida a RDS que cree una contraseña para que pueda administrarla.

## Instancias permitidas en AWS Academy

Si queréis crear bases de datos con máquinas más potentes, podéis utilizar instancias hasta nivel *large*, y a ser posible a ráfagas (instancias *t*). Dentro de la *Configuración adicional*, es importante deshabilitar la monitorización mejorada (no tenemos permiso para su uso en *AWS Academy*).

Como vamos a querer acceder a nuestro servidor de MariaDB desde fuera de una VPC de EC2, necesitamos configurar el acceso público. Al hacerlo, no quiere decir que ya sea accesible desde fuera de internet, ya que necesitamos configurar su grupo de seguridad (recordad que funciona a modo

de *firewall*). Así pues, es recomendable crear un nuevo grupo de seguridad para que permitamos las conexiones del puerto 3306 a nuestra IP.

**Conectividad** Información

**Recurso informático**  
Elija si desea configurar una conexión a un recurso informático para esta base de datos. La configuración de una conexión cambiará automáticamente la configuración de conectividad para que el recurso informático pueda conectarse a esta base de datos.

☒ **No se conecte a un recurso informático EC2**  
No configure una conexión a un recurso informático para esta base de datos. Puede configurar manualmente una conexión a un recurso informático más adelante.

☐ **Conectarse a un recurso informático de EC2**  
Configure una conexión a un recurso informático EC2 para esta base de datos.

**Virtual Private Cloud (VPC)** Información  
Elija la VPC. La VPC define el entorno de red virtual para esta instancia de DB.

Default VPC (vpc-0ea93980693e11db5)

Solo se muestran las VPC con grupos de subredes de base de datos correspondientes.

Después de crear una base de datos, no puede cambiar su VPC.

**Grupo de subred de DB** Información  
Elija el grupo de subred de DB. El grupo de subred de DB define las subredes e intervalos de IP que puede usar la instancia de DB en la VPC seleccionada.

default-vpc-0ea93980693e11db5

**Acceso público** Información

☒ **Sí**  
RDS asigna una dirección IP pública a la base de datos. Las instancias de Amazon EC2 y otros recursos fuera de la VPC pueden conectarse a la base de datos. Los recursos de la VPC también pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

☐ **No**  
RDS no asigna una dirección IP pública a la base de datos. Solo las instancias de Amazon EC2 y otros recursos dentro de la VPC pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

**Grupo de seguridad de VPC (firewall)** Información  
Elija uno o varios grupos de seguridad de VPC para permitir el acceso a su base de datos. Asegúrese de que las reglas del grupo de seguridad permitan el tráfico entrante adecuado.

☐ **Elegir existente**  
Elegir grupos de seguridad de VPC existentes

☒ **Crear nuevo**  
Crear un grupo de seguridad nuevo de VPC

Nuevo nombre del grupo de seguridad de VPC  
RDSAbiertaSG

**Zona de disponibilidad** Información

Sin preferencia

► Configuración adicional

### Configuración de la conectividad en RDS

Así pues, una vez creada (lo cual tarda unos minutos), podremos seleccionar la instancia y ver su panel de información:

RDS > Databases > instituto

**instituto** Modificar Acciones ▼

---

**Resumen**

Identificador de base de datos instituto	CPU -	Estado 🟢 Disponible	Clase db.t4g.micro
Rol Instancia	Actividad actual 0 Conexiones	Motor MariaDB	Región y AZ us-east-1f

---

**Conectividad y seguridad** | Supervisión | Registros y eventos | Configuración | Mantenimiento y copias de seguridad | Etiquetas

---

**Conectividad y seguridad**

<b>Punto de enlace y puerto</b>  Punto de enlace instituto.cdexqeikfdkr.us-east-1.rds.amazonaws.com  Puerto 3306	<b>Redes</b>  Zona de disponibilidad us-east-1f  VPC vpc-0ea93980693e11db5  Grupo de subredes default-vpc-0ea93980693e11db5  Subredes subnet-0619846117c326e1a subnet-09370aa213e78b349 subnet-0cbac317195b354ea subnet-04cb1c7c14857f95a subnet-04ecab278e0271c57 subnet-02867d9cd78449079  Tipo de red IPv4	<b>Seguridad</b>  Grupos de seguridad de la VPC RDSAbiertaSG (sg-082a7687117e4c107) 🟢 Activo  Accesible públicamente Sí  Entidad de certificación rds-ca-2019  Fecha de la entidad de certificación August 22, 2024, 19:08 (UTC+02:00)
--	---	---

### Resumen de instancia en RDS

A continuación, si copiamos la información del punto de enlace y creamos una conexión en *DBeaver*, veremos que nos conectamos correctamente (si no hemos creado un nuevo grupo de seguridad, deberemos editar el grupo de seguridad por defecto, y añadir una regla de entrada para el protocolo TCP para el puerto 3306, y por ejemplo para todo internet - 0.0.0.0/0).

Conectar a base de datos

**Connection Settings**  
MariaDB ajustes de conexión

General | Driver properties | SSH | Proxy | SSL

Server

Server Host: instituto.cdexqeikfdkr.us-east-1.rds.amazonaws.com Port: 3306

Database:

Authentication (Database Native)

Nombre de usuario: admin

Contraseña: •••••••• ☒ Save password locally

Advanced

Server Time Zone: Auto-detect

Local Client: MySQL Binaries

📘 You can use variables in connection parameters. Connection details (name, type, ...)

Driver name: MariaDB Driver Settings

Probar conexión ... ← Anterior Siguiente → Finalizar Cancelar

### Configuración en DBeaver



Una vez conectado, ya procedemos de la misma manera que si tuviéramos la base de datos en un entorno local.

## AWS CLI

También es posible crear una instancia de RDS directamente desde AWS CLI. Para ello, el primer paso es crear el grupo de seguridad mediante **ec2 create-security-group**, al cual le ponemos un nombre para luego poder referenciarlo:

```
aws ec2 create-security-group \  
    --group-name rds-public \  
    --description "Acceso abierto para RDS MariaDB"
```

Tras su creación, obtendremos su identificador:

```
{  
    "GroupId": "sg-03b19c935d87f01a0",  
    "SecurityGroupArn": "arn:aws:ec2:us-east-1:533267238455:security-group/sg-03b19c935d87f01a0"  
}
```

A continuación, configuramos el grupo de seguridad y le añadimos una regla de entrada mediante **ec2 authorize-security-group-ingress** para permitir las conexiones al puerto de *MariaDB* desde cualquier dirección:

```
aws ec2 authorize-security-group-ingress \  
    --group-name rds-public \  
    --protocol tcp \  
    --port 3306 \  
    --cidr 0.0.0.0/0
```

Tras su ejecución, obtendremos de nuevo su identificador:

```
{  
    "Return": true,  
    "SecurityGroupRules": [  
        {  
            "SecurityGroupRuleId": "sgr-0d90f0e63a9ba67d2",  
            "GroupId": "sg-03b19c935d87f01a0",  
            "GroupOwnerId": "533267238455",  
            "IsEgress": false,  
            "IpProtocol": "tcp",  
            "FromPort": 3306,  
            "ToPort": 3306,  
            "CidrIpv4": "0.0.0.0/0",
```

```

        "SecurityGroupRuleArn": "arn:aws:ec2:us-east-1:533267238455:security-
group-rule/sg-0d90f0e63a9ba67d2"
    }
}
}

```

Finalmente, ya tenemos todo preparado para crear la instancia mediante **rds create-db-instance**, indicando la clase de la instancia, el usuario y contraseña, así como las diferentes características que hemos configurado de forma visual en el apartado anterior (recuerda modificar el identificador del grupo de seguridad por el identificador que hayas obtenido en los pasos previos):

```

aws rds create-db-instance \
    --db-instance-identifier iabd-rds \
    --db-instance-class db.t3.micro \
    --engine mariadb \
    --master-username admin \
    --master-user-password adminadmin \
    --publicly-accessible \
    --allocated-storage 20 \
    --vpc-security-group-ids sg-03b19c935d87f01a0 \
    --db-name MiBaseDeDatos \
    --port 3306 \
    --region us-east-1

```

## Uso de instantáneas

Por defecto, al crear una instancia en RDS se crean copias de seguridad diarias que se retienen automáticamente hasta 7 días.

Si editamos la instancia, en la **Configuración adicional** podemos configurar la **Copias de seguridad** para cambiar cuantos días debe retener las copias de seguridad realizadas, así como la franja horaria en la que se realizará la copia cada día.

Estas copias, para incrementar la tolerancia a fallos, la podemos persistir además en otra región:

### Copia de seguridad

☒ **Habilitar las copias de seguridad automatizadas.**  
Crea una instantánea de un momento dado de su base de datos

Periodo de retención de copia de seguridad [Información](#)  
El número de días (1 a 35) durante los que se conservan las copias de seguridad automatizadas.

7 días

Periodo de copia de seguridad [Información](#)  
El intervalo de tiempo diario (en UTC) durante el cual RDS realiza copias de seguridad automatizadas.

☒ Elegir una ventana

☐ Sin preferencia

Hora de inicio

07 : 58 UTC

Duración

0.5 horas

☒ Copiar las etiquetas en las instantáneas

Replicación de copias de seguridad [Información](#)

☐ Habilitar la replicación en otra región de AWS

Cuando se habilita la replicación, se crean de forma automática copias de seguridad de la instancia de base de datos para la recuperación de desastres en la región seleccionada, además de la región actual.

### Configuración de las copias de seguridad

El proceso de crear una instantánea de los datos requiere congelar toda la actividad en disco, lo que puede provocar que algunas peticiones a la base de datos se retrasen o incluso falle. Debido a esto, es

recomendable crear las instantáneas cuando menos se emplee la aplicación (normalmente, por las noches)

## Creación manual

Además, podemos crear una instantánea de forma manual en cualquier momento (por ejemplo, antes de añadir una nueva funcionalidad que suponga modificar la estructura de la base de datos). Para ello, podemos pulsar sobre el botón de *Acciones* y seleccionamos **Realizar instantánea**:

RDS > Instantáneas de > Realizar instantánea

### Tomar instantánea de la base de datos

**Preferencias**  
Para tomar una instantánea de base de datos, elija una instancia de base de datos y asigne un nombre a la instantánea de base de datos.

**Instancia de base de datos**  
Identificador de instancia de base de datos. Es la clave única que identifica una Instancia de base de datos.

instituto ▼

**Nombre de la instantánea**  
Identificador de la instantánea de la base de datos.

snapshot-manual

El identificador de la instantánea no distingue entre mayúsculas y minúsculas, pero se almacena todo en minúsculas, como en "mysnapshot". No puede ser nulo, vacío o estar en blanco. Debe contener de 1 a 255 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede terminar con un guión ni contener dos guiones consecutivos.

Cancelar Realizar instantánea

### *Creación de una instantánea de forma manual*

Si queremos crearla mediante AWS CLI, usaremos el comando [rds create-db-snapshot](#):

```
aws rds create-db-snapshot --db-snapshot-identifier snapshot-manual-cli --db-instance-identifier instituto
```

## Restaurando

En cualquier momento podemos crear una nueva instancia de RDS a partir de una instantánea. Para ello, seleccionamos la instantánea que nos interesa, y seleccionamos **Acciones y Restaurar instantánea**.

Nos pedirá confirmar el motor del gestor de la base de datos (en nuestro caso *MariaDB*), asignarle un nuevo nombre y vamos a poder cambiar el tipo de la máquina así como la cantidad de almacenamiento y demás opciones que tenemos al crear una nueva instancia.

RDS > Instantáneas de > Restaurar instantánea

## Restaurar instantánea

Está creando una nueva instancia de base de datos o clúster de base de datos a partir de una instantánea. Se ha seleccionado el grupo de seguridad de la VPC y el grupo de parámetros predeterminados para la nueva instancia o clúster de base de datos, pero puede cambiar esta configuración.

### Configuración de la instancia de base de datos

Motor de base de datos

MariaDB

### Configuración

ID de la instantánea de base de datos

El identificador de la instantánea de base de datos.

[snapshot-manual](#)

Identificador de instancias de bases de datos [Información](#)

Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos pertenecientes a su cuenta de AWS en la región de AWS actual.

instituto-restored

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en minúsculas (como en "miinstanciadebd"). Restricciones: de 1 a 60 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos. No puede terminar con un guion.

### Configuración de la instancia

Las opciones de configuración de la instancia de base de datos que aparecen a continuación están limitadas a las que admite el motor que ha seleccionado anteriormente.

Clase de instancia de base de datos [Información](#)

▼ Ocultar filtros

☒ Incluir clases de generación anterior

☐ Clases estándar (incluye clases m)

☐ Clases optimizadas para memoria (incluye clases r y x)

☒ Clases con ráfagas (incluye clases t)

db.t4g.small

2 vCPUs 2 GiB RAM Red: 2085 Mbps

### Almacenamiento

Tipo de almacenamiento [Información](#)

SSD de uso general (gp2)

Rendimiento de referencia determinado por el tamaño del volumen

Almacenamiento asignado [Información](#)

20 GiB

El valor mínimo es 20 GiB y el valor máximo es 16.384 GiB

### Restaurando una instantánea

Si la queremos restaurar mediante AWS CLI, usaremos el comando [rds restore-db-instance-from-db-snapshot](#):

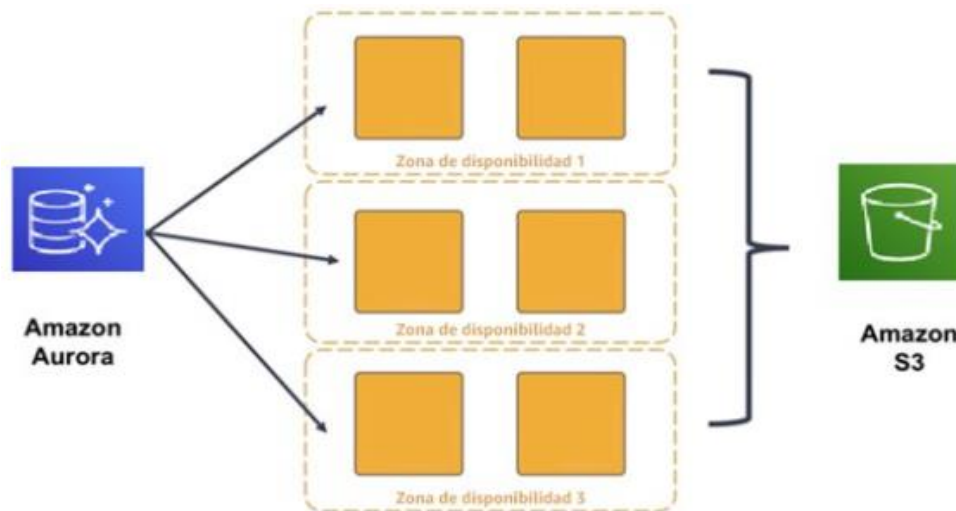
```
aws rds restore-db-instance-from-db-snapshot --db-instance-identifier instituto-new --db-snapshot-identifier snapshot-manual-cli
```

Si no le indicamos más parámetros, creará una instancia con las mismas características que la instancia original.

## Amazon Aurora

*Amazon Aurora* es una base de datos relacional compatible con *MySQL* y *PostgreSQL* optimizada para la nube. Combina el rendimiento y la disponibilidad de las bases de datos comerciales de alta gama con la simplicidad y la rentabilidad de las bases de datos de código abierto. Ofrece dos modelos, el clásico basado en instancias y un [modelo serverless](#) en el cual se contratan unidades de computación (ACU).

Al estar desarrollado de forma nativa por Amazon se adapta mejor a su infraestructura en coste, rendimiento y alta disponibilidad. Está pensado como un subsistema de almacenamiento distribuido de alto rendimiento, ofreciendo automatización de las tareas que requieren mucho tiempo, como el aprovisionamiento, la implementación de parches, las copias de seguridad, la recuperación, la detección de errores y su reparación.



*Alta disponibles con Aurora*

*Aurora* replica varias copias de los datos en múltiples zonas de disponibilidad y realiza copias de seguridad continuas de los datos en S3.

Respecto a la seguridad, hay varios niveles disponibles, incluidos el aislamiento de la red con *VPC*, el cifrado en reposo por medio de claves creadas y controladas con *AWS KMS* y el cifrado de los datos en tránsito mediante *SSL*.

Respecto al coste, si cogemos el mismo ejemplo anterior de una instancia de Aurora compatible con *MySQL* con dos procesadores y 8GB de RAM, en este caso, la *db.t4g.large*, el precio se queda en 106\$ mensuales.

## Referencias

- [Guía de usuario de Amazon RDS](#)