Distributed Cache Service

Descripción general del servicio

Edición	01
Fecha	2025-01-23

HUAWEI TECHNOLOGIES CO., LTD.

Marcas y permisos


y otras marcas registradas de Huawei pertenecen a Huawei Technologies Co., Ltd.
Todas las demás marcas registradas y los otros nombres comerciales mencionados en este documento son propiedad de sus respectivos titulares.

Aviso

Las funciones, los productos y los servicios adquiridos están estipulados en el contrato celebrado entre Huawei y el cliente. Es posible que la totalidad o parte de los productos, las funciones y los servicios descritos en el presente documento no se encuentren dentro del alcance de compra o de uso. A menos que el contrato especifique lo contrario, ninguna de las afirmaciones, informaciones ni recomendaciones contenidas en este documento constituye garantía alguna, ni expresa ni implícita.

La información contenida en este documento se encuentra sujeta a cambios sin previo aviso. En la preparación de este documento se realizaron todos los esfuerzos para garantizar la precisión de sus contenidos. Sin embargo, ninguna declaración, información ni recomendación contenida en el presente constituye garantía alguna, ni expresa ni implícita.

Índice

1 Infografía para comparar DCS for Redis con Redis de código abierto

2 Qué es DCS

Huawei Cloud Distributed Cache Service (DCS) es un servicio de caché en memoria rápida, distribuida y en línea compatible con Redis. Es confiable, escalable, utilizable de inmediato y fácil de gestionar, lo que satisface sus requerimientos de alto rendimiento de lectura/escritura y rápido acceso a los datos.

- Uso inmediato

  DCS proporciona instancias de nodo único, principales/en espera, de separación de lectura/escritura, de Clúster Proxy y de Clúster Redis con especificaciones que van desde 128 MB hasta 2048 GB. Las instancias de DCS se pueden crear con pocos clics en la consola, sin la necesidad de preparar los servidores.

  Las instancias de DCS compatible con Redis 4.0, 5.0 y 6.0 están en contenedores y se pueden crear en segundos.

- Seguridad y confiabilidad

  El almacenamiento y el acceso a los datos de instancias están protegidos de forma segura a través de los servicios de gestión de seguridad de Huawei Cloud, incluidos Identity and Access Management (IAM), Virtual Private Cloud (VPC), Cloud Eye, and Cloud Trace Service (CTS).

  Las instancias principal/en standby y de clúster se pueden implementar dentro de una zona de disponibilidad (AZ) o a través de AZ.

- Escalamiento automático

  Las instancias DCS se pueden escalar hacia arriba o hacia abajo en línea, lo que le ayuda a controlar los costos en función de los requisitos de servicio.

- Gestión sencilla

  Se proporciona una consola basada en web para que realice varias operaciones, como reiniciar instancias, modificar parámetros de configuración y realizar copias de seguridad y restaurar datos. También se proporcionan interfaces de programación de aplicaciones (API) RESTful para la gestión automática de instancias.

- Migración en línea

  Puede crear una tarea de migración de datos en la consola para importar los archivos de copia de seguridad o migrar los datos en línea.

DCS for Redis

DCS for Redis está compatible con Redis 4.0/5.0/6.0.

NOTA

- DCS for Redis 3.0 ya no se proporciona. Puede utilizar DCS for Redis 4.0 o posterior en su lugar.
- Las instancias de DCS compatible con Redis 6.0 solo se admiten en algunas regiones.

Redis es un sistema de almacenamiento que admite varios tipos de estructuras de datos, incluidos los pares clave-valor. Se puede utilizar en los escenarios tales como el almacenamiento en caché de datos, la publicación/suscripción de eventos y la cola de alta velocidad, como se describe en **Escenarios de aplicación**. Redis se escribe en ANSI C, soporta lectura/escritura directa de **strings**, **hashes**, **lists**, **sets**, **sorted sets**, y **streams**. Redis trabaja con un conjunto de datos en memoria que puede persistir en el disco.

Las instancias de DCS Redis se pueden personalizar en función de sus requisitos.

- DCS for Redis 4.0/5.0/6.0 básico

Tabla 2-1 DCS for Redis 4.0/5.0/6.0 básico

Tipo de instancia	DCS for Redis proporciona los siguientes tipos de instancias para adaptarse a diferentes escenarios de servicio: ● Nodo único: Adecuado para almacenar en caché datos temporales en escenarios de baja confiabilidad. Las instancias de nodo único admiten operaciones de lectura/escritura altamente simultáneas, pero no admiten persistencia de datos. Los datos se eliminarán después de reiniciar las instancias. ● Principal/en espera: Cada instancia principal/en espera se ejecuta en dos nodos (un principal y otro en espera). El nodo en espera replica los datos en forma sincrónica desde el nodo principal. Si el nodo principal falla, el nodo es espera se convierte automáticamente en el nodo principal. Las operaciones de lectura y escritura se pueden dividir escribiendo en el nodo principal y leyendo desde el nodo en espera. Esto mejora el rendimiento general de lectura/escritura de caché. ● Clúster de proxy: Además del clúster de Redis nativo, una instancia de clúster de proxy tiene los proxy y balanceadores de carga. Los balanceadores de carga implementan el equilibrio de carga. Se distribuyen diferentes solicitudes a diferentes proxy para lograr una alta simultaneidad. Cada partición del clúster tiene un nodo principal y un nodo en espera. Si el nodo principal es defectuoso, el nodo en espera en la misma partición es promovida a la función del principal para asumir los servicios. ● Clúster Redis: Cada instancia de Clúster Redis consta de varias **particiones** y cada partición incluye un nodo principal y varias réplicas (o ninguna réplica). Las participaciones no son visibles para usted. Si el nodo principal falla, una réplica en la misma partición se hace cargo de los servicios. Puede dividir las operaciones de lectura y escritura escribiendo en el nodo principal y leyendo desde las réplicas. Esto mejora el rendimiento general de lectura/escritura de caché. ● Separación de lectura/escritura: Una instancia de separación de lectura/escritura tiene proxy y balanceadores de carga además de la arquitectura principal/en espera. Los balanceadores de carga implementan el equilibrio de carga, y diferentes solicitudes se distribuyen a diferentes proxy. Los proxy distinguen entre las solicitudes de lectura y escritura, y las envía a nodos principales o nodos en espera, respectivamente.
Especificaciones de instancias	DCS for Redis proporciona las instancias de diferentes especificaciones, que van desde 128 MB a 1024 GB.
Compatibilidad con software de código abierto:	Las instancias de DCS son compatibles con Redis 4.0/5.0/6.0 de código abierto.

Arquitectura subyacente	El despliegue en contenedores se realiza en máquinas físicas. 100,000 QPS en un nodo único para una instancia de Redis 4.0/5.0; 150,000 QPS en una instancia de nodo único de Redis 6.0 de edición básica y 170,000 QPS en una instancia principal/en espera de Redis 6.0 de edición básica.
Alta disponibilidad (HA) y recuperación ante desastres (DR)	Todas las instancias, excepto las de nodo único, se pueden desplegar en AZ dentro de una región con fuentes de energía y redes físicamente aisladas.

Para obtener más información acerca de Redis de código abierto, visite **https://redis.io/**.

- DCS for Redis 6.0 de edición profesional

  Huawei Cloud DCS edición profesional está completamente desarrollada por Huawei Cloud. Esta edición utiliza el modelo maestro-N*trabajador de subprocesos múltiples en lugar del modelo maestro-trabajador convencional, lo que mejora el rendimiento general n veces. DCS de edición profesional es totalmente compatible con motores, módulos y scripts de Redis en términos de scripts y atómica de eventos. Al utilizar el mismo hardware, esta edición duplica el QPS de Redis y reduce la latencia en aproximadamente un 60 %.

  En versiones anteriores a Redis 6.0, una consulta lenta a menudo causa que otras consultas se retrasen debido al modelo de subproceso único. Para abordar los problemas de rendimiento, la nueva edición ha realizado una serie de optimización basada en un modelo multiproceso. Se ha mejorado la simultaneidad de subprocesos múltiples para el procesamiento de eventos de E/S y backend; el acceso a los datos almacenados en caché se acelera aún más a través del bloqueo de giro justo; las claves caducadas se pueden eliminar dos veces más rápido gracias a algoritmos optimizados; La compatibilidad con las subclaves caduca también mejora el rendimiento de lectura/escritura de las claves grandes. Como resultado, la nueva edición es adecuada para escenarios que requieren un alto rendimiento de un solo nodo, como temas de tendencias y eventos de transmisión en vivo en Internet.

Tabla 2-2 DCS for Redis 6.0 de edición profesional

Tipo de instancia	DCS for Redis 6.0 de edición profesional incluye subediciones de rendimiento y almacenamiento. Solo están disponibles las instancias principales/en espera. La edición profesional (almacenamiento) utiliza discos de memoria y SSD. Cada instancia principal/en espera se ejecuta en dos nodos (uno principal y uno en espera). El nodo en espera replica los datos en forma sincrónica desde el nodo principal. Si el nodo principal falla, el nodo es espera se convierte automáticamente en el nodo principal. Las operaciones de lectura y escritura se pueden dividir escribiendo en el nodo principal y leyendo desde el nodo en espera. Esto mejora el rendimiento general de lectura/escritura de caché.

Especificaciones de instancias	8 GB, 16 GB, 32 GB, 64 GB
Compatibilidad con software de código abierto:	Totalmente compatible con Redis 6.
Arquitectura subyacente	Desplegado en VM. 400,000 QPS en un solo nodo.
HA y DR	Todas las instancias, excepto las de nodo único, se pueden desplegar en AZ dentro de una región con fuentes de energía y redes físicamente aisladas.

DCS for Memcached (descontinuado)

NOTA

DCS for Memcached ya no se proporciona. Puede usar instancias de DCS para Redis en su lugar.

Memcached es un sistema de almacenamiento en caché de clave-valor en memoria que admite la lectura/escritura de cadenas simples. A menudo se utiliza para almacenar en caché los datos de la base de datos back-end para aliviar la carga en estas bases de datos y acelerar las aplicaciones web. Para obtener más información sobre los escenarios de su aplicación, consulte **Escenarios de aplicación de Memcached (discontinuado)**.

Además de la compatibilidad total con Memcached, DCS for Memcached proporciona el modo de standby inmediata y la persistencia de datos.

Tabla 2-3 Configuración de instancia de DCS para Memcached

Tipo de instancia	DCS for Memcached proporciona los dos tipos de instancias siguientes para adaptarse a diferentes escenarios de servicio: Nodo único: Adecuado para almacenar en caché datos temporales en escenarios de baja confiabilidad. Las instancias de nodo único admiten operaciones de lectura/escritura altamente simultáneas, pero no admiten persistencia de datos. Los datos se eliminarán después de reiniciar las instancias. Principal/en espera: Cada instancia principal/en espera se ejecuta en dos nodos (un principal y otro en espera). El nodo en standby replica datos de forma sincrónica desde el nodo principal, pero no soporta las operaciones de lectura/escritura. Si el nodo principal falla, el nodo es espera se convierte automáticamente en el nodo principal.
Memoria	Especificación de instancias de DCS para Memcached de nodo único o principal/en espera: 2 GB, 4 GB, 8 GB, 16 GB, 32 GB y 64 GB.

HA y DR	Las instancias principal/en espera de DCS Memcached se pueden desplegar en AZs en la misma región con fuentes de energía y redes físicamente aisladas.

Para obtener más información acerca de Memcached de código abierto, visite **https://memcached.org/**.

3 Escenarios de aplicación

Escenarios de la aplicación de Redis

Muchos sitios web de comercio electrónico a gran escala y aplicaciones de transmisión de video y juegos requieren un acceso rápido a grandes cantidades de datos que tienen estructuras de datos simples y no necesitan consultas frecuentes de unión. En tales escenarios, puede usar Redis para lograr un acceso rápido pero económico a los datos. Redis le permite recuperar datos de almacenes de datos en memoria en lugar de depender completamente de bases de datos basadas en disco más lentas. Además, ya no es necesario realizar las tareas de gestión adicionales. Estas características hacen de Redis un complemento importante de las bases de datos tradicionales basadas en disco y un servicio básico esencial para las aplicaciones de Internet que reciben acceso de alta simultaneidad.

Los escenarios de aplicación típicos de DCS for Redis son los siguientes:

1. **Ventas flash de comercio electrónico**

   El catálogo de productos de comercio electrónico, las ofertas y los datos de ventas flash se pueden almacenar en caché en Redis.

   Por ejemplo, el acceso a datos de alta concurrencia en las ventas flash difícilmente puede ser manejado por las bases de datos relacionales tradicionales. Requiere que el hardware tenga una configuración más alta, como E/S de disco. Por el contrario, Redis soporta 100,000 QPS por nodo y le permite implementar el bloqueo usando comandos simples como **SET**, **GET**, **DEL** y **RPUSH** para manejar ventas flash.

   Para obtener más información sobre el bloqueo, véase **Implementación de bloqueos distribuidos con Redis** en *Prácticas recomendadas*.

2. **Comentarios de video en directo**

   En la transmisión en vivo, los datos de usuario en línea, clasificación de regalos y comentarios de viñetas se pueden almacenar como conjuntos ordenados en Redis.

   Por ejemplo, los comentarios de viñetas se pueden devolver mediante el comando **ZREVRANGEBYSCORE**. Los comandos **ZPOPMAX** y **ZPOPMIN** en Redis 5.0 pueden facilitar adicionalmente el procesamiento de mensajes.

3. **Tabla de clasificación del juego**

   En los juegos en línea, los jugadores de mayor clasificación se muestran y se actualizan en tiempo real. La clasificación de la tabla se puede almacenar como conjuntos ordenados, que son fáciles de usar con hasta 20 comandos.

   Para obtener más información, véase **Clasificación con Redis** en *Prácticas recomendadas*.

4. **Comentarios en redes sociales**

   En las aplicaciones web, las consultas de comentarios de publicaciones a menudo implican ordenar por tiempo en el orden descendente. A medida que los comentarios se acumulan, la clasificación se vuelve menos eficiente.

   Mediante el uso de listas en Redis, se puede devolver un número preestablecido de comentarios desde la caché, en lugar de desde el disco, facilitando la carga de la base de datos y acelerando las respuestas de la aplicación.

Escenarios de aplicación de Memcached (discontinuado)

Memcached es adecuado para almacenar datos simples clave-valor.

1. Páginas web

   El almacenamiento en caché de datos estáticos como páginas HTML, Cascading Style Sheets (CSS) e imágenes en instancias de DCS Memcached mejora el rendimiento de acceso de las páginas web.

2. Base de datos de frontend

   En sistemas dinámicos como redes sociales y sitios de blogs, las operaciones de escritura son mucho menos que las operaciones de lectura, como consultar a usuarios, amigos y artículos. Para facilitar la carga de la base de datos y mejorar el rendimiento, los siguientes datos se pueden almacenar en caché en Memcached:

   – Datos de acceso frecuente que no requieren actualizaciones en tiempo real y pueden caducar automáticamente

     Ejemplo: últimas listas de artículos y clasificaciones. Aunque los datos se generan constantemente, su impacto en la experiencia del usuario es limitado. Tales datos pueden almacenarse en caché durante un periodo de tiempo preestablecido y accederse desde la base de datos después de este periodo. Si los editores de páginas web desean ver la clasificación más reciente, se puede configurar una política de borrado o actualización de caché.

   – Datos de acceso frecuente que requieren actualizaciones en tiempo real

     Ejemplo: listas de amigos, listas de artículos y registros de lectura. Tales datos pueden almacenarse en caché en Memcached primero y, a continuación, actualizarse siempre que se produzcan cambios (agregar, modificar y eliminar datos).

3. Ventas flash

   Es difícil para las bases de datos tradicionales escribir una operación de colocación de pedidos durante las ventas flash en la base de datos, modificar los datos de inventario y garantizar la coherencia de las transacciones mientras se garantiza la experiencia del usuario ininterrumpida.

   Los comandos **incr** y **decr** de Memcached se pueden usar para almacenar información de inventario y completar la colocación de pedidos en memoria. Una vez que se envía un pedido, se genera un número de pedido. Entonces, el pedido puede ser pagado.

NOTA

Escenarios en los que Memcached no es adecuado:

- El tamaño de un único objeto de caché es superior a 1 MB.

  Memcached no puede almacenar en caché un objeto de más de 1 MB. En tales casos, utilice Redis.

- La clave contiene más de 250 caracteres.

  Para usar Memcached en tal escenario, puede generar un hash MD5 para la clave y almacenar en caché el hash en su lugar.

- Se requiere una alta fiabilidad de los datos.

  Memcached de código abierto no proporciona replicación, backup y migración de datos, por lo que la persistencia de datos no es compatible.

  Las instancias principal/en espera de DCS para Memcached soportan persistencia de datos. Para obtener más información, póngase en contacto con el soporte técnico.

- Se requieren estructuras y procesamiento de datos complejos.

  Memcached solo admite los pares simples de clave-valor, y no admite estructuras de datos complejas como listas y conjuntos, ni operaciones complejas como la ordenación.

4 Seguridad

4.1 Responsabilidades compartidas

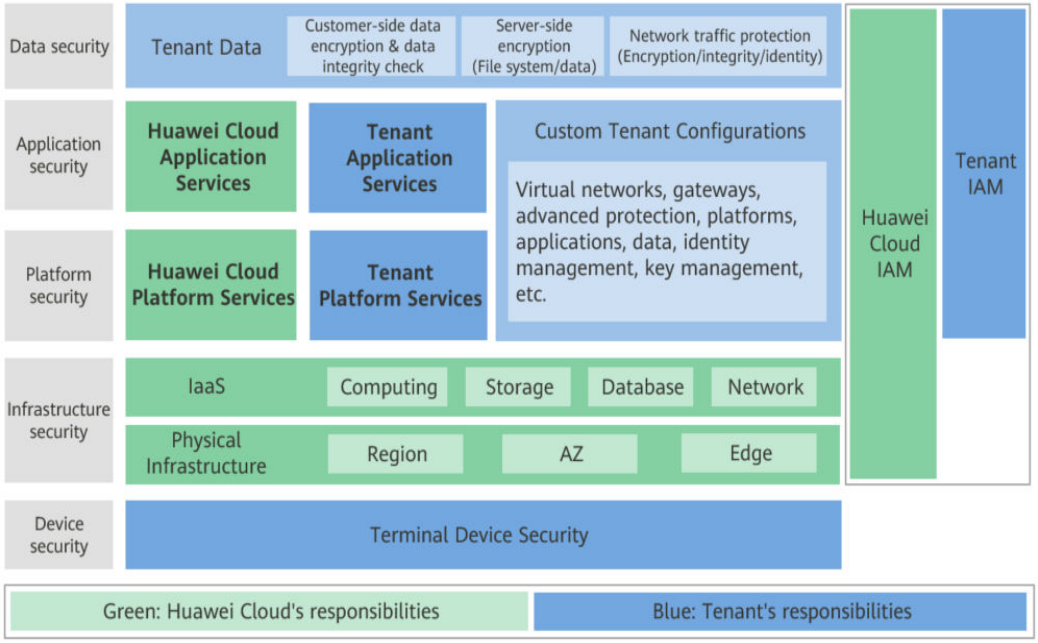
Huawei garantiza que su compromiso con la seguridad cibernética nunca se verá compensado por la consideración de intereses comerciales. Para hacer frente a los desafíos emergentes de seguridad en la nube y a las amenazas y ataques generalizados de seguridad en la nube, Huawei Cloud crea un sistema integral de garantía de seguridad de servicios en la nube para diferentes regiones e industrias basado en las ventajas únicas de software y hardware, las leyes, las regulaciones, los estándares de la industria y el ecosistema de seguridad de Huawei.

Figura 4-1 ilustra las responsabilidades compartidas por Huawei Cloud y los usuarios.

- **Huawei Cloud**: Garantizar la seguridad de los servicios en la nube y proporcionar nubes seguras. Las responsabilidades de seguridad de Huawei Cloud incluyen garantizar la seguridad de nuestros servicios IaaS, PaaS y SaaS, así como los entornos físicos de los centros de datos de Huawei Cloud donde nuestros IaaS, PaaS, y los servicios SaaS operan. Huawei Cloud es responsable no solo de las funciones de seguridad y el rendimiento de nuestra infraestructura, servicios en la nube y tecnologías, sino también de la seguridad general de la nube y, en el sentido más amplio, del cumplimiento de seguridad de nuestra infraestructura y servicios.
- **Tenant**: Utilizar la nube de forma segura. Los inquilinos de Huawei Cloud son responsables de la gestión segura y efectiva de las configuraciones personalizadas por el inquilino de los servicios en la nube, incluidos IaaS, PaaS y SaaS. Esto incluye, entre otros, redes virtuales, el sistema operativo de los hosts e invitados de máquinas virtuales, firewalls virtuales, API Gateway, servicios de seguridad avanzados, todo tipo de servicios en la nube, datos del inquilino, cuentas de identidad, y gestión de claves.

Libro blanco de seguridad de Huawei Cloud elabora las ideas y medidas para construir la seguridad en Huawei Cloud, incluidas las estrategias de seguridad en la nube, el modelo de responsabilidad compartida, el cumplimiento y la privacidad, las organizaciones y el personal de seguridad, la seguridad de la infraestructura, el servicio y la seguridad del inquilino, la seguridad de ingeniería, seguridad de O&M y seguridad del ecosistema.

Figura 4-1 Modelo de responsabilidad de seguridad compartida de Huawei Cloud



4.2 Autenticación de identidad y control de acceso

Autenticación de identidades

Los solicitantes de acceso deben presentar la credencial de identidad para la verificación de validez de identidad al acceder a DCS en la consola o al invocar a las API. DCS utiliza Identity and Access Management (IAM) para proporcionar tres modos de autenticación de identidad: **contraseñas**, **claves de acceso** y **claves de acceso temporales**. Además, DCS proporciona **protección de inicio de sesión** y **políticas de autenticación de inicio de sesión** para reforzar la seguridad de la autenticación de identidad.

Control de acceso

Puede asignar diferentes permisos para DCS a los empleados de la organización para una gestión detallada de los permisos. IAM proporciona autenticación de identidad, gestión de permisos y control de acceso, lo que le ayuda a proteger el acceso a sus recursos de Huawei Cloud. Para obtener más información, véase **Gestión de permisos**.

4.3 Protección de datos

DCS toma diferentes medidas para mantener los datos seguros y confiables.

Tabla 4-1 Métodos y funciones de protección de datos de DCS

Medida	Descripción	Referencia
DR y multiactivo	Para cumplir con los requisitos de confiabilidad de sus datos y servicios, puede desplegar una instancia de DCS en una sola AZ (sala de equipos única) o en todas las AZ (DR intraurbana).	**Recuperación ante desastres y solución multiactiva**
Replicación de datos	Los datos se pueden sincronizar incrementalmente entre réplicas para la consistencia de los datos de la caché. Cuando se produce una excepción de red o una falla de nodo, se realiza automáticamente una conmutación por error mediante las réplicas. Una vez rectificada la falla, se realiza una sincronización completa para garantizar la uniformidad de los datos.	**Replicación de datos**
Persistencia de los datos	Pueden ocurrir excepciones durante el funcionamiento diario del sistema de servicio. Algunos sistemas de servicio requieren alta confiabilidad, lo que incluye alta disponibilidad de instancias, seguridad de datos de caché, capacidad de recuperación e incluso almacenamiento permanente, para que los datos de copia de respaldo se puedan usar para restaurar instancias si ocurre una excepción, lo que garantiza la ejecución del servicio.	**Copia de respaldo y restauración de instancias**

4.4 Auditoría y logs

Cloud Trace Service (CTS)

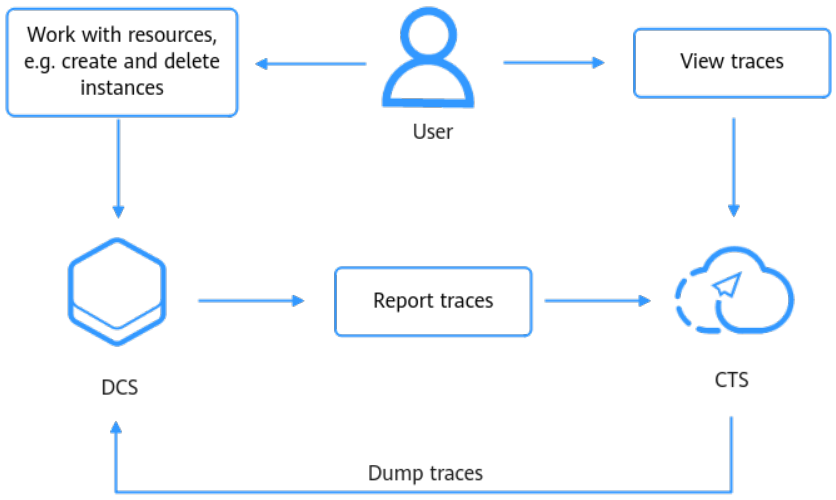
CTS registra operaciones en los recursos en la nube de su cuenta. Puede utilizar los registros generados por CTS para realizar análisis de seguridad, rastrear cambios en los recursos, auditar el cumplimiento y localizar fallos.

Después de habilitar CTS y configurar un rastreador, CTS puede registrar la gestión y las trazas de datos de DCS para su auditoría.

Para obtener detalles sobre cómo habilitar y configurar CTS, véase **Habilitación de CTS**.

Para obtener detalles sobre la gestión de DCS y las operaciones de datos que se pueden rastrear, véase **Operaciones registradas por CTS**.

Figura 4-2 CTS



4.5 Resiliencia

DCS proporciona una arquitectura de confiabilidad de tres niveles y utiliza DR entre AZ, DR de instancia dentro de AZ y replicación de datos de instancia para garantizar la durabilidad y confiabilidad del servicio.

Tabla 4-2 Arquitectura de confiabilidad de DCS

Solución de confiabilidad	Descripción
Recuperación ante desastres entre AZ	DCS proporciona instancias principales/en espera, de Clúster Redis y de Clúster Proxy que admiten DR entre AZ. Cuando una AZ es anormal, las instancias aún pueden proporcionar servicios.
Recuperación ante desastres de instancia dentro de la AZ	Una instancia de DCS principal/en espera, Clúster Redis o Clúster Proxy tiene múltiples nodos para DR a nivel de instancia dentro de una AZ. Si el nodo principal presenta fallas, los servicios se conmutan rápidamente a un nodo en espera para garantizar la continuidad de los servicios DCS.
Recuperación ante desastres de datos	La Recuperación ante desastres de datos se implementa con la replicación de datos.

4.6 Monitoreo de riesgos de seguridad

DCS utiliza Cloud Eye para ayudarlo a monitorear sus instancias de DCS y recibir alarmas y notificaciones en tiempo real. Puede obtener información clave sobre las instancias en tiempo

real, como solicitudes de servicio, uso de recursos, ancho de banda, cantidad de operaciones simultáneas y tiempos de control de flujos.

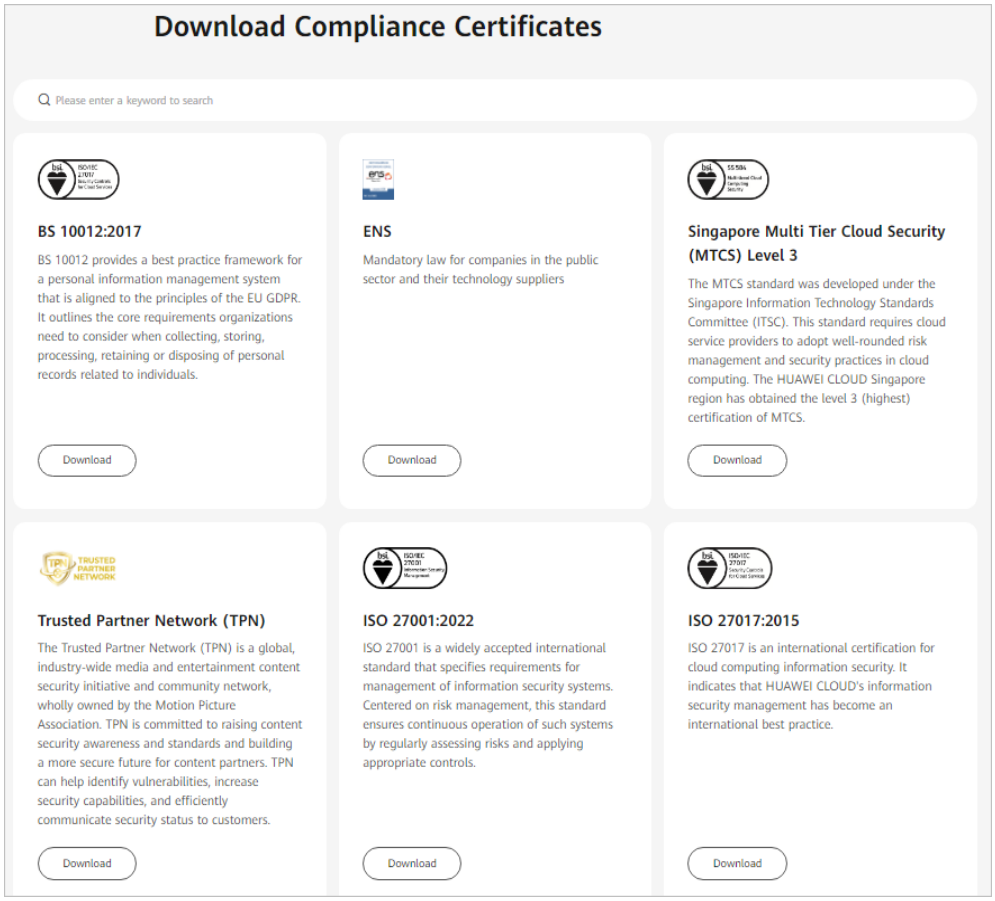
Para obtener detalles sobre las métricas de DCS y cómo crear reglas de alarma, véase **Metrics de DCS**.

4.7 Certificados

Certificados de Cumplimiento

Los servicios y plataformas de Huawei Cloud han obtenido diversas certificaciones de seguridad y cumplimiento de organizaciones autorizadas, como la Organización Internacional de Normalización (ISO). Puede **descargarlos** desde la consola.

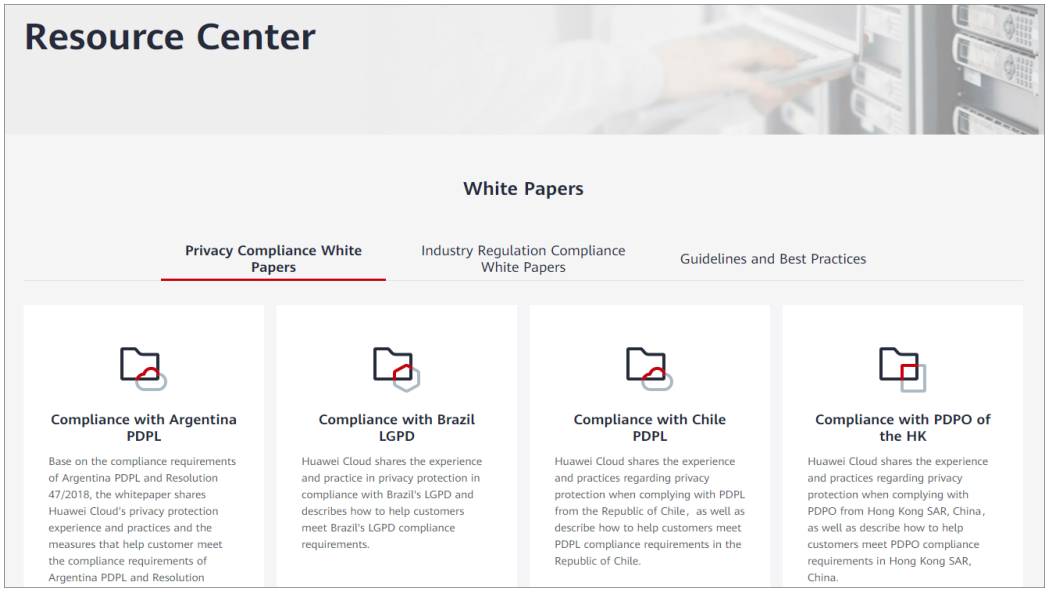
Figura 4-3 Descarga de certificados de cumplimiento



Centro de recursos

Huawei Cloud también proporciona los siguientes recursos para ayudar a los usuarios a cumplir con los requisitos de cumplimiento. Para obtener más información, consulte **Centro de recursos**.

Figura 4-4 Centro de recursos



4.8 Nota técnica sobre seguridad

Distributed Cache Service (DCS) es un servicio de base de datos en memoria seguro y confiable proporcionado por Huawei Cloud.

DCS cumple con las regulaciones de seguridad, se adhiere a los límites del servicio y nunca monetizará los datos del cliente. Le permite aprovisionar rápidamente diferentes tipos de instancias y admite el escalado automático de recursos de cómputo y almacenamiento según sea necesario. Para evitar la pérdida de datos, DCS proporciona funciones como copia de respaldo, instantáneas y restauraciones automatizadas. También le permite modificar los parámetros de configuración para el ajuste de instancias.

DCS ofrece muchas funciones para garantizar la confiabilidad y seguridad de los datos de la cuenta, como VPC, grupos de seguridad, listas blancas, encriptación de SSL para acceso público, copias de respaldo automatizadas, instantáneas de datos y despliegue entre AZ.

NOTA

Para obtener detalles sobre cómo DCS garantiza la seguridad de los datos, véase **Prácticas recomendadas de seguridad de DCS**.

Aislamiento de red

Puede configurar reglas de entrada de VPC para permitir que segmentos de direcciones IP específicos se conecten a sus instancias. Las instancias de DCS se ejecutan en una VPC independiente. Puede crear un grupo de subred entre AZ y desplegar instancias de alta disponibilidad en él. Después de crear una instancia, DCS asignará una dirección IP de subred a la instancia para la conexión. Después de desplegar las instancias de DCS en una VPC, puede usar una VPN para acceder a las instancias desde otras VPC. También puede crear un ECS en la VPC que aloja las instancias y conectar el ECS y las instancias con una dirección IP flotante. Las subredes y los grupos de seguridad se pueden utilizar juntos para aislar instancias de DCS y mejorar la seguridad.

Control de acceso

Al crear una instancia de DCS, puede configurar grupos de seguridad (compatible con DCS for Redis 3.0, Redis 6.0 de edición profesional y Memcached) o listas blancas (compatibles con DCS for Redis 4.0, 5.0 y 6.0 de edición básica).

Puede establecer reglas de grupos de seguridad entrantes y salientes o configurar listas blancas para controlar el acceso a y desde las instancias de DCS dentro de una VPC.

No es necesario reiniciar las instancias al configurar grupos de seguridad o listas blancas.

Al crear una instancia de DCS, se recomienda habilitar la protección con contraseña y establecer una contraseña de acceso para la instancia para evitar que los clientes no autenticados accedan a la instancia por error.

Encriptación de almacenamiento y transmisión

RESP (REdis Serialization Protocol), el protocolo de comunicación de Reids, solo admite la transmisión de texto plano en versiones anteriores a Redis 6.0. Las instancias de DCS para Redis 6.0 de edición básica admiten el protocolo RESP3 y encriptación sobre SSL.

Para el acceso público a las instancias de DCS para Redis 3.0, puede habilitar encriptación TLS con Stunnel. Cuando DCS aprovisiona instancias, el Certificate Chain (CA) especificada generará un certificado de servicio único para cada instancia. Al conectarse a una instancia, los clientes pueden utilizar los certificados raíz de CA descargados de la consola de gestión para autenticar el servidor de instancia y cifrar datos durante la transmisión.

Si necesita cifrar los datos en tránsito cuando el acceso público no está habilitado, utilice una encriptación de algoritmo (como AES 256) para cifrar los datos antes del almacenamiento y mantener el acceso en el dominio de confianza. Los datos también se cifran antes de la persistencia en el disco.

Copia de respaldo automatizada y manual

Las instancias de DCS se pueden respaldar automática o manualmente. La función de copia de respaldo automatizada está deshabilitada por defecto. Los datos de copia de respaldo de una instancia se almacenan durante un máximo de 7 días. Una vez habilitado la copia de respaldo automatizada, puede restaurar los datos en la instancia. Durante la copia de respaldo automatizada, se respaldan todos los datos de una instancia y el rendimiento del nodo en espera se verá afectado. Las copias de respaldo manuales son copias de respaldo completas de instancias iniciadas por el usuario. Los datos de copia de respaldo se almacenan en buckets de OBS y se quitan al eliminar la instancia correspondiente.

Replicación de datos

Una instancia de DCS principal/en espera o de clúster se puede desplegar dentro de una AZ o en varias AZ para HA. Para el despliegue entre AZ, DCS inicia y mantiene la sincronización de datos. La alta disponibilidad se logra al hacer que un nodo en espera se haga cargo en caso de que ocurra una falla en el nodo principal. Cuando las operaciones son de lectura pesada, puede usar instancias de DCS para Redis 4.0 o posteriores que admitan la separación de lectura/escritura, o instancias de clúster que tengan varias réplicas. DCS mantiene la sincronización de datos entre el principal y las réplicas. Puede conectarse a diferentes direcciones de una instancia para aislar operaciones de lectura y escritura.

Eliminación de datos

Si elimina una instancia de DCS, se eliminarán todos los datos almacenados en la instancia. Nadie puede consultar ni restaurar los datos una vez eliminados.

5 Tipos de instancia de DCS

5.1 Redis de nodo único

Cada instancia de DCS para Redis de nodo único tiene solo un nodo y no admite la persistencia de datos. Son adecuados para servicios de caché que no requieren confiabilidad de datos.

NOTA

- No puede actualizar la versión de Redis para una instancia. Por ejemplo, una instancia de DCS para Redis 4.0 de nodo único no se puede actualizar a una instancia de DCS para Redis 5.0 de nodo único. Si el servicio requiere las características de versiones de Redis superiores, cree una instancia de DCS Redis de una versión superior y, a continuación, migre los datos de la instancia antigua a la nueva.
- Las instancias de nodo único no pueden garantizar la persistencia de los datos y no admiten el copia de respaldo de datos automático o manual. Tenga cuidado al usarlos.

Características

1. Baja sobrecarga del sistema y alto QPS

   Las instancias de nodo único no admiten sincronización de datos ni persistencia de datos, lo que reduce la sobrecarga del sistema y admite una mayor simultaneidad. El QPS de las instancias de DCS Redis de un solo nodo alcanza hasta 100,000.

2. Monitoreo de procesos y recuperación automática de fallas

   Con un mecanismo de monitoreo de HA, si una instancia de DCS de un nodo único resulta defectuosa, se inicia un nuevo proceso en 30 segundos para reanudar el aprovisionamiento del servicio.

3. Usabilidad lista para usar y sin persistencia de datos

   Las instancias de un nodo único de DCS se pueden usar de inmediato porque no implican la carga de datos. Si su servicio requiere un alto QPS, puede calentar los datos de antemano para evitar un fuerte impacto de simultaneidad en la base de datos backend.

4. Bajo costo y adecuado para desarrollo y pruebas

   Las instancias de un nodo único son un 40% más baratas que las instancias de DCS principal/en espera, y son adecuadas para configurar entornos de desarrollo o pruebas.

En resumen, las instancias DCS de nodo único admiten las operaciones de lectura/escritura altamente simultáneas, pero no admiten persistencia de datos. Los datos se eliminarán después

de reiniciar las instancias. Son adecuados para escenarios que no requieren persistencia de datos, como el almacenamiento en caché front-end de la base de datos, para acelerar el acceso y facilitar la carga de concurrencia fuera del back-end. Si los datos deseados no existen en la caché, las solicitudes irán a la base de datos. Al reiniciar el servicio o la instancia de DCS, puede pregenerar los datos de caché de la base de datos de disco para aliviar la presión sobre el backend durante el inicio.

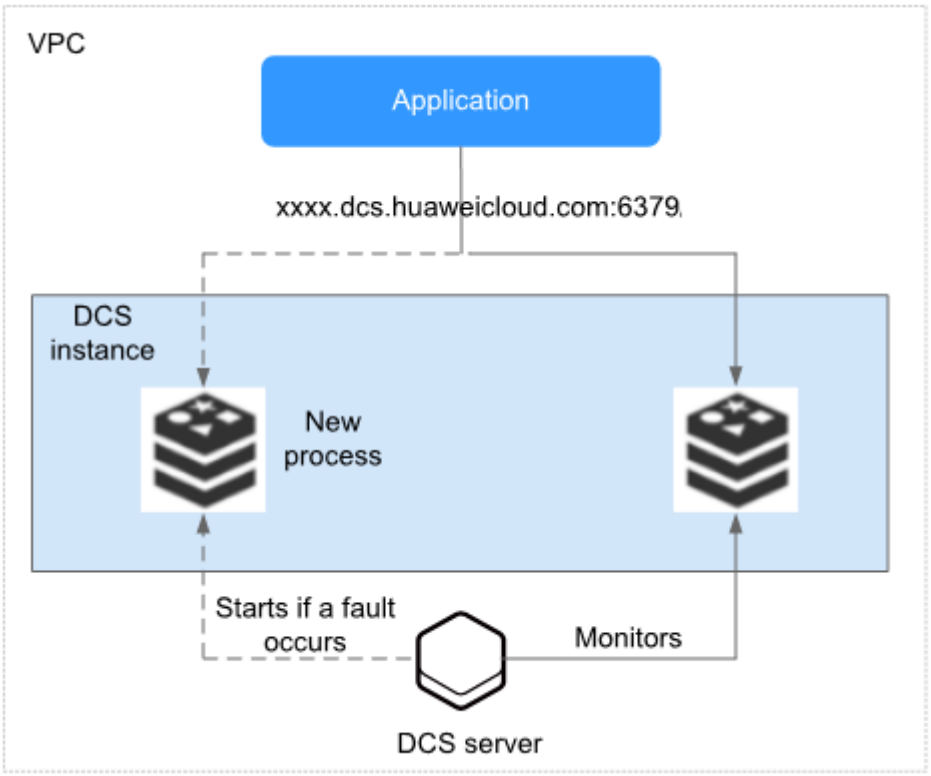
Arquitectura

Figura 5-1 muestra la arquitectura de una instancia de DCS para Redis de nodo único.

NOTA

Redis 3.0 y 6.0 professional no admiten la personalización de puertos y solo permiten el puerto 6379. Para Redis 4.0/5.0/6.0 básico, puede especificar un puerto o usar el puerto predeterminado 6379. En la siguiente arquitectura, se utiliza el puerto 6379. Si ha personalizado un puerto, reemplace **6379** por el puerto real.

Figura 5-1 Arquitectura de instancia de DCS para Redis de nodo único



Descripción de la arquitectura:

- **VPC**

  La VPC donde se ejecutan todos los nodos de la instancia.

- **Aplicación**

  El cliente de la instancia, que es la aplicación que se ejecuta en un Elastic Cloud Server (ECS).

  Las instancias de DCS para Redis y Memcached son respectivamente compatibles con los protocolos Redis y Memcached, y se puede acceder a través de clientes de código

abierto. Para obtener ejemplos de acceso a instancias de DCS con diferentes lenguajes de programación, véase las **instrucciones de acceso a instancias**.

- **Instancia de DCS**

  Una instancia de DCS de un nodo único, que tiene solo un nodo y un proceso de Redis.

  DCS monitorea la disponibilidad de la instancia en tiempo real. Si el proceso de Redis se vuelve defectuoso, DCS inicia un nuevo proceso en cuestión de segundos para reanudar el aprovisionamiento del servicio.

5.2 Redis principal/en espera

En esta sección se describen las instancias principal/en espera de DCS Redis.

NOTA

No puede actualizar la versión de Redis para una instancia. Por ejemplo, una instancia principal/en espera de DCS Redis 4.0 no se puede actualizar a una instancia principal/en espera de DCS Redis 5.0. Si el servicio requiere las características de versiones de Redis superiores, cree una instancia de DCS Redis de una versión superior y, a continuación, migre los datos de la instancia antigua a la nueva.

Características

Las instancias principal/en standby de DCS tienen mayor disponibilidad y confiabilidad que las instancias de un nodo único de DCS.

Las instancias principal/en standby de DCS tienen las siguientes características:

1. **Persistencia de los datos y alta confiabilidad**

   De forma predeterminada, la persistencia de datos está habilitada tanto por el nodo principal como por el en standby de una instancia principal/en standby de DCS para Redis.

   El nodo en standby de una instancia de Redis 3.0 es invisible para usted. Solo el nodo principal proporciona las operaciones de lectura/escritura de datos.

   El nodo en espera de una instancia de Redis 4.0/5.0/6.0 básico está visible para usted. Puede leer datos del nodo en standby conectándose con la dirección de solo lectura de la instancia.

2. **Sincronización de datos**

   Los datos en los nodos principal y en espera se mantienen consistentes a través de la sincronización incremental.

   NOTA

   Después de recuperarse de una excepción de red o un fallo de nodo, las instancias principal/en standby realizan una sincronización completa para garantizar la coherencia de los datos.

3. **Conmutación automática principal/en standby**

   Si el nodo principal se vuelve defectuoso, la instancia se desconecta y no está disponible durante varios segundos. El nodo es espera toma el control durante 15 a 30 segundos sin operaciones manuales para reanudar los servicios estables.

NOTA

- Durante la conmutación por error se producen desconexiones e indisponibilidades. El cliente del servicio debe poder volver a conectarse o reintentar.
- Una vez completada la conmutación por error principal/en espera, el nodo principal anterior defectuoso (entonces el nodo en espera) se recuperará más tarde. El acceso al servicio al nodo principal anterior fallará. En este caso, configure los SDK de Redis. Para obtener más información, véase **Conexión a Redis en un cliente**.

4. **Múltiples políticas de recuperación ante desastres**

   Cada instancia principal/en standby de DCS se puede implementar en AZs con fuentes de alimentación y redes físicamente aisladas. Las aplicaciones también se pueden desplegar entre las AZ para lograr HA tanto para datos como para aplicaciones.

5. **Separación de lecturas/escrituras**

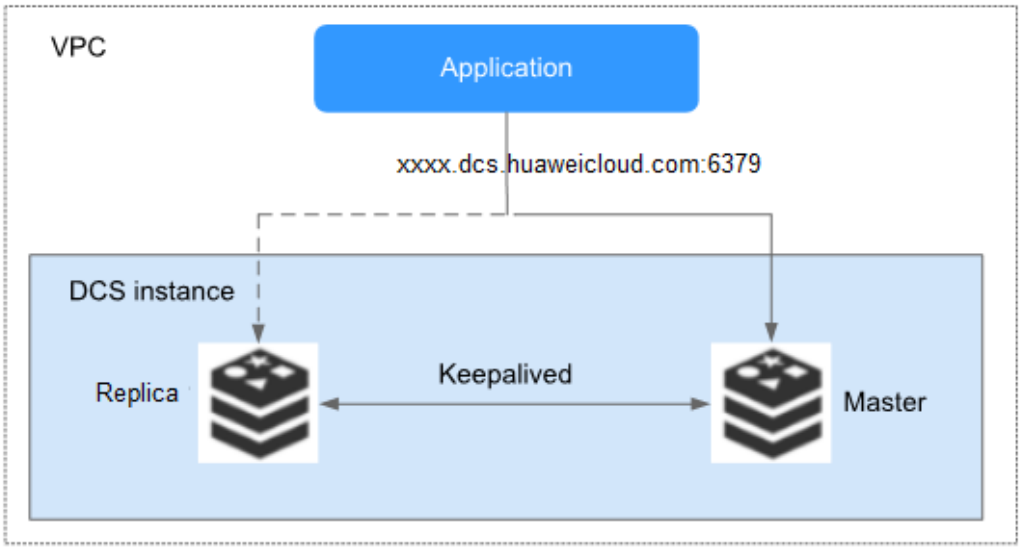
   Las instancias principales/en espera de DCS para Redis 4.0/5.0/6.0 básico admiten la separación de lectura/escritura del cliente. Al conectarse a una instancia de este tipo, puede utilizar la dirección de lectura/escritura para conectarse al nodo principal o utilizar la dirección de solo lectura para conectarse al nodo en standby.

   Si utiliza una instancia principal/en espera y necesita separación de lectura/escritura del lado del cliente, configure el cliente. Si se requiere la separación de lectura/escritura, se recomiendan las instancias de **separación de lectura/escritura**.

Arquitectura de instancias principal/en espera de DCS para Redis 3.0

Figura 5-2 muestra la arquitectura de una instancia principal/en espera de DCS compatible con Redis 3.0.

Figura 5-2 Arquitectura de una instancia principal/en espera de DCS para Redis 3.0



Descripción de la arquitectura:

- **VPC**

  La VPC donde se ejecutan todos los nodos de la instancia.

- **Aplicación**

El cliente Redis de la instancia, que es la aplicación que se ejecuta en el ECS.

Las instancias de DCS para Redis y Memcached son respectivamente compatibles con los protocolos Redis y Memcached, y se puede acceder a través de clientes de código abierto. Para obtener ejemplos de acceso a instancias de DCS con diferentes lenguajes de programación, consulte las **instrucciones de acceso a instancias**.

- **Instancia de DCS**

  Una instancia principal/en espera de DCS que tiene un nodo principal y un nodo de réplica. De forma predeterminada, la persistencia de datos está habilitada y los datos se sincronizan entre los dos nodos.

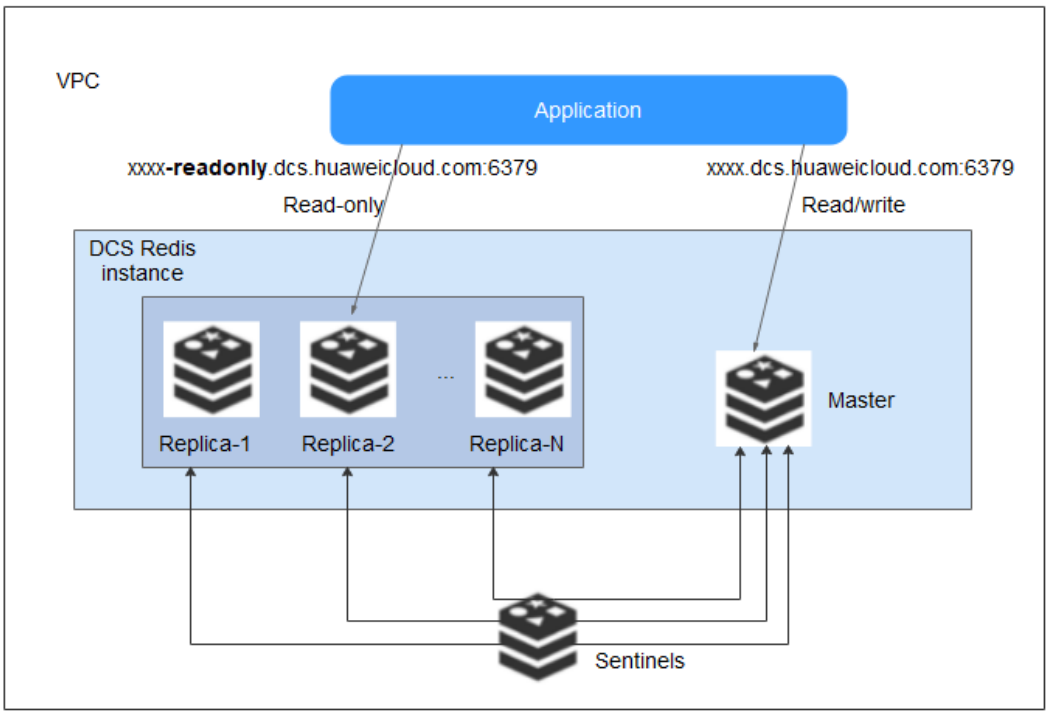
  DCS monitorea la disponibilidad de la instancia en tiempo real. Si el nodo principal se vuelve defectuoso, el nodo en espera se convierte en el nodo principal y reanuda el aprovisionamiento de servicio.

  El puerto Redis predeterminado es 6379.

Arquitectura de las instancias principal/en espera de DCS para Redis 4.0/5.0/6.0 básico

La siguiente figura muestra la arquitectura de una instancia principal/en espera de DCS para Redis 4.0/5.0/6.0 básico.

Figura 5-3 Arquitectura de las instancias principal/en espera de DCS para Redis 4.0/5.0/6.0 básico



Descripción de la arquitectura:

1. Cada instancia principal/en espera de DCS para Redis 4.0/5.0/6.0 básico tiene una dirección de nombre de dominio (para conectarse al nodo principal) para lectura y escritura y una dirección (para conectarse al nodo en espera) para solo lectura.

   Estas direcciones se pueden obtener en la página de detalles de la instancia en la consola de DCS.

2. Puede configurar Sentinel para una instancia principal/en espera para Redis 4.0/5.0/6.0 básico. Los Sentinel supervisan el estado de ejecución de los nodos principal y en espera. Si el nodo principal se vuelve defectuoso, se realizará una migración por falla.

   Sentinel son invisibles para usted y solo se utilizan en el servicio. Para obtener más información sobre Sentinel, véase **¿Qué es Sentinel?**

3. Un nodo de solo lectura tiene las mismas especificaciones que un nodo de lectura/escritura. Cuando se crea una instancia principal/en standby, se incluyen un par de nodos principal y en espera en la instancia de forma predeterminada.

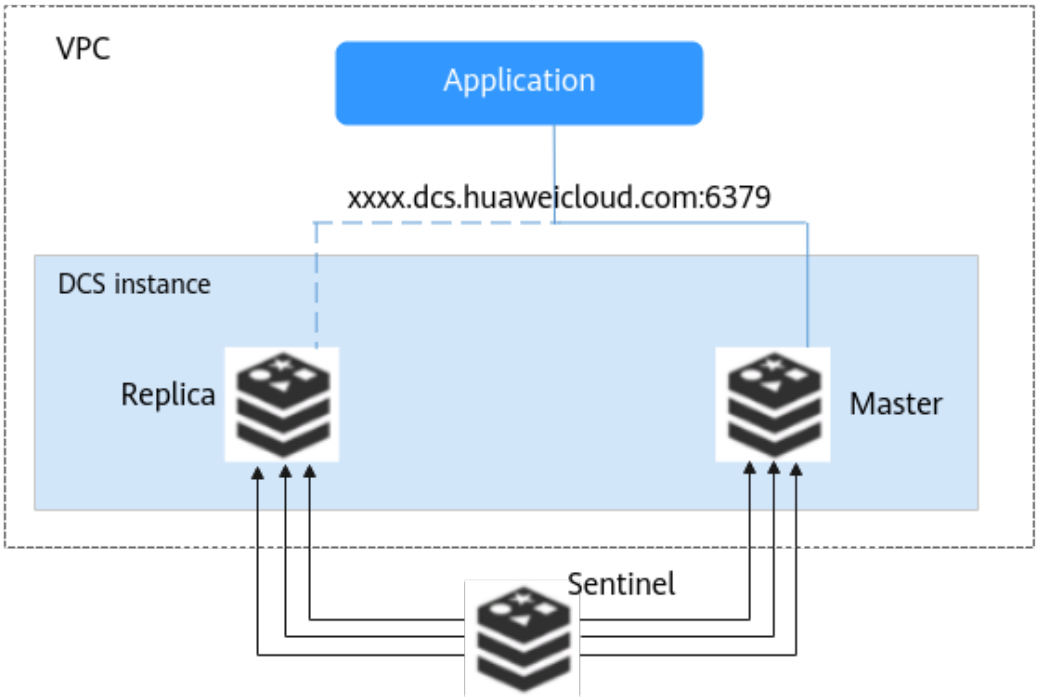
   NOTA

   - Para las instancias de DCS para Redis 4.0/5.0/6.0 básico, puede personalizar el puerto. Si no se especifica ningún puerto, se utilizará el puerto predeterminado 6379. En el diagrama de arquitectura, se utiliza el puerto 6379. Si ha personalizado un puerto, reemplace **6379** por el puerto real.
   - Nombres de dominio de solo lectura de instancias principal/en espera de DCS para Redis 4.0, 5.0 o 6.0 de edición básica no admiten el balanceo de carga. Para una alta confiabilidad y baja latencia, utilice instancias de separación de clúster o lectura/escritura.
   - Las solicitudes a la dirección de nombre de dominio pueden fallar si el par principal/en espera (DCS para Redis 4.0, 5.0 o 6.0 de edición básica) es defectuoso. Para lograr mayor confiabilidad y menor latencia, utilice instancias de separación de lecturas/escrituras.

Arquitectura de instancias principal/en espera de DCS para Redis 6.0 de edición profesional

Figura 5-4 muestra la arquitectura de una instancia principal/en espera de DCS para Redis 6.0 de edición profesional.

Figura 5-4 Arquitectura de instancia principal/en espera de DCS para Redis 6.0 de edición profesional



Descripción de la arquitectura:

- **VPC**

  La VPC donde se ejecutan todos los nodos de la instancia.

  NOTA

  Para el acceso intra-VPC, el cliente y la instancia deben estar en la misma VPC con configuraciones de reglas de grupo de seguridad específicas. Para obtener más información sobre cómo configurar las reglas, véase **¿Cómo configuro un grupo de seguridad?**.

- **Aplicación**

  El cliente Redis de la instancia, que es la aplicación que se ejecuta en el ECS.

  Las instancias de DCS para Redis son compatibles con Redis y se puede acceder a ellas con clientes de código abierto. Para obtener ejemplos de acceso a instancias de DCS con diferentes lenguajes de programación, véase **Conexión a Redis en un cliente**.

- **Instancia de DCS**

  Una instancia principal/en espera de DCS que tiene un nodo principal y un nodo de réplica.

  Las instancias principal/en espera de DCS para Redis 6.0 de la edición profesional admiten Sentinels. Los Sentinel supervisan el estado de ejecución de los nodos principal y en espera. Si el nodo principal se vuelve defectuoso, se realizará una migración por falla.

  Sentinel son invisibles para usted y solo se utilizan en el servicio. Para obtener más información sobre Sentinel, véase **¿Qué es Sentinel?**

  El puerto Redis predeterminado es 6379.

5.3 Clúster Proxy para Redis

DCS for Redis proporciona instancias de Clúster Proxy, que utilizan Linux Virtual Server (LVS) y proxies para lograr una alta disponibilidad. Las instancias de Clúster Proxy tienen las siguientes características:

- El cliente está desacoplado del servicio en la nube.
- Admiten millones de solicitudes simultáneas, lo que equivale a las instancias de Clúster Redis.
- Una amplia gama de especificaciones de memoria se adaptan a diferentes escenarios.

NOTA

- No puede actualizar la versión de Redis para una instancia. Por ejemplo, una instancia de Clúster Proxy de DCS para Redis 4.0 no se puede actualizar a una instancia de Clúster Proxy de DCS para Redis 5.0. Si el servicio requiere las características de versiones de Redis superiores, cree una instancia de DCS compatible con Redis de una versión superior y, a continuación, migre los datos de la instancia antigua a la nueva.
- Una instancia de Clúster Proxy puede conectarse de la misma manera que una instancia de nodo único o principal/en espera está conectada, sin ninguna configuración especial en el cliente. Puede utilizar la dirección IP o el nombre de dominio de la instancia, y no es necesario conocer o utilizar las direcciones de proxy o de partición.

Instancia de Clúster Proxy de DCS compatible con Redis 3.0

Las instancias de Clúster Proxy de DCS compatible con Redis 3.0 se basan en x86, son compatibles con **codis de código abierto** y vienen con especificaciones que van desde 64 GB

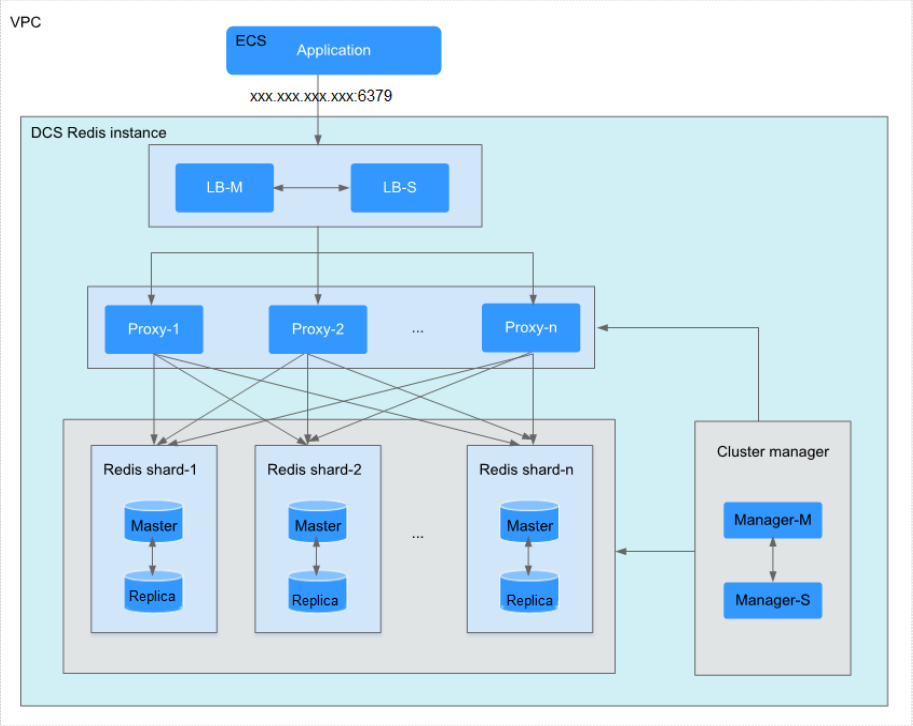
a 1024 GB, que cumplen con los requisitos para **millones de conexiones simultáneas** y **caché de datos masivos**. El almacenamiento y acceso de datos distribuidos es implementado por DCS, sin necesidad de desarrollo o mantenimiento.

Cada instancia de Clúster Proxy consta de balanceadores de carga, proxy, administradores de clústeres y **particiones**.

Tabla 5-1 Memoria total, proxies y particiones de instancias de Clúster Proxy de DCS compatible con Redis 3.0

Memoria total	Proxy	Particiones
64 GB	3	8
128 GB	6	16
256 GB	8	32
512 GB	16	64
1024 GB	32	128

Figura 5-5 Arquitectura de una instancia de Clúster Proxy de DCS compatible con Redis 3.0



Descripción de la arquitectura:

- **VPC**

  La VPC donde se ejecutan todos los nodos de la instancia.

NOTA

Si el acceso público no está habilitado para la instancia, asegúrese de que el cliente y la instancia están en la misma VPC y configure las reglas de grupo de seguridad para la VPC.

Si el acceso público está habilitado para la instancia, el cliente puede desplegarse fuera de la VPC para acceder a la instancia a través de la EIP enlazado a la instancia.

Para obtener más información, vea **Acceso público a una instancia de DCS compatible con Redis 3.0** y **¿Cómo configuro un grupo de seguridad?**

- **Aplicación**

  El cliente utilizado para acceder a la instancia.

  Se puede acceder a las instancias de DCS para Redis mediante clientes de código abierto. Para obtener ejemplos de acceso a instancias de DCS con diferentes lenguajes de programación, consulte **Acceso a una instancia de DCS compatible con Redis**.

- **LB-M/LB-S**

  Los balanceadores de carga, que se despliegan en modo HA principal/en standby. Las direcciones de conexión (**dirección IP:Puerto** y **nombre de dominio:Puerto**) de la instancia del clúster de DCS compatible con Redis son las direcciones de los balanceadores de carga.

- **Proxy**

  El servidor proxy utilizado para lograr una alta disponibilidad y procesar solicitudes de clientes de alta simultaneidad.

  Puede conectarse a una instancia Clúster Proxy en las direcciones IP de sus proxy.

- **Partición de Redis**

  Una partición del clúster.

  Cada partición consta de un par de nodos principal/de réplica. Si el nodo principal se vuelve defectuoso, el nodo de réplica se hace cargo automáticamente de los servicios del clúster.

  Si tanto el nodo principal como el de réplica de una partición son defectuosos, el clúster aún puede proporcionar servicios, pero los datos de la partición defectuosa son inaccesibles.

- **Administrador de clústeres**

  Los administradores de configuración del clúster, que almacenan las configuraciones y las políticas de particionamiento del clúster. No puede modificar la información acerca de los administradores de configuración.

Instancias de Clúster Proxy de DCS compatible con Redis 4.0/5.0/6.0 básico

NOTA

Las instancias de Clúster Proxy de DCS compatible con Redis 4.0 y posteriores se proporcionan solo en algunas regiones.

Las instancias de Clúster Proxy de DCS compatible con Redis 4.0/5.0/6.0 se crean en función de Redis 4.0/5.0/6.0 de código abierto y son compatibles con **codis de código abierto**. Proporcionan múltiples especificaciones de gran capacidad que van de 4 GB a 1024 GB.

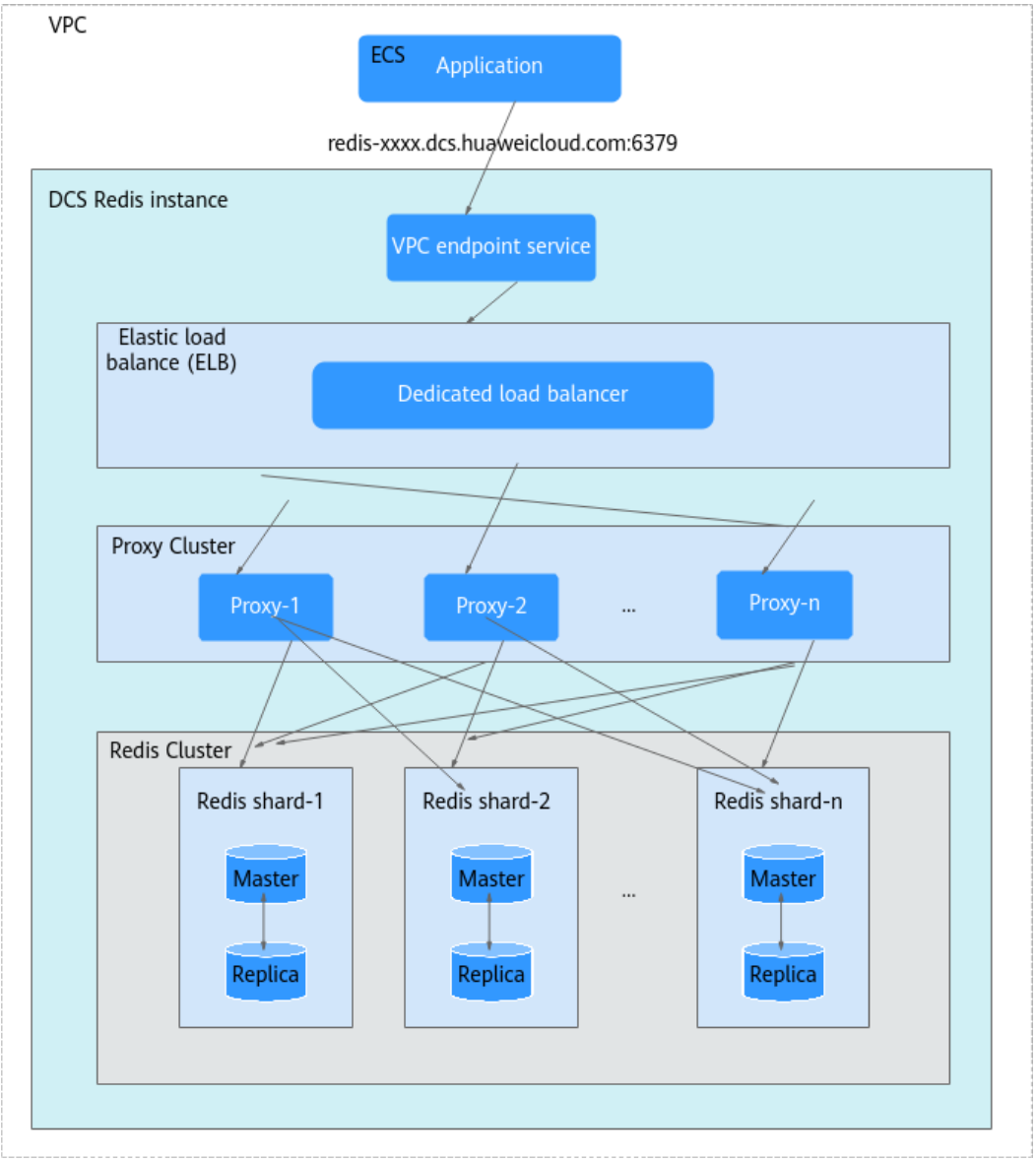
Tabla 5-2 enumera el número de particiones correspondientes a diferentes especificaciones. Puede personalizar el tamaño de la partición al crear una instancia. Actualmente, la cantidad de réplicas no se puede personalizar. Por defecto, cada partición tiene dos réplicas.

Memoria por partición=Especificación de instancia/Número de particiones. Por ejemplo, si una instancia de 48 GB tiene 6 particiones, el tamaño de cada partición es de 48 GB/6 = 8 GB.

Tabla 5-2 Memoria total, proxies y particiones de instancias de Clúster Proxy de DCS compatible con Redis 4.0/5.0/6.0 básico

Memoria total	Proxy	Particiones	Memoria por partición (GB)
4 GB	3	3	1.33
8 GB	3	3	2.67
16 GB	3	3	5.33
24 GB	3	3	8
32 GB	3	3	10.67
48 GB	6	6	8
64 GB	8	8	8
96 GB	12	12	8
128 GB	16	16	8
192 GB	24	24	8
256 GB	32	32	8
384 GB	48	48	8
512 GB	64	64	8
768 GB	96	96	8
1024 GB	128	128	8

Figura 5-6 Instancias de Clúster Proxy de DCS compatible con Redis 4.0/5.0/6.0 básico



Descripción de la arquitectura:

- **VPC**

  La VPC donde se ejecutan todos los nodos de la instancia.

  NOTA

  El cliente y la instancia del clúster deben estar en la misma VPC, y la lista blanca de la instancia debe permitir el acceso desde la dirección IP del cliente.

- **Aplicación**

  El cliente utilizado para acceder a la instancia.

  Se puede acceder a las instancias de DCS compatible con Redis con clientes de código abierto. Para obtener ejemplos de acceso a instancias de DCS con diferentes lenguajes de programación, véase **Conexión a Redis en un cliente**.

- **Servicio de punto de conexión de VPC**

Puede configurar su instancia de DCS compatible con Redis como un servicio de punto de conexión de VPC y acceder a la instancia en la dirección de servicio de punto de conexión de VPC.

La dirección IP o la dirección de nombre de dominio de la instancia Clúster Proxy de DCS compatible con Redis es la dirección del servicio de punto de conexión de VPC.

- **ELB**

  Los balanceadores de carga se despliegan en modo HA de clúster y soportan el despliegue multi-AZ.

- **Proxy**

  El servidor proxy utilizado para lograr una alta disponibilidad y procesar solicitudes de clientes de alta simultaneidad.

  No se puede conectar a una instancia de Clúster Proxy en las direcciones IP de sus proxies.

- **Clúster Redis**

  Una partición del clúster.

  Cada partición es una instancia de Redis de réplica dual principal/en espera. Cuando el nodo principal presenta fallas, el nodo en espera se conmutará al nodo principal después de 15 a 30 segundos. El acceso a la partición fallará hasta que se complete la conmutación.

  Si tanto el nodo principal como el de réplica de una partición son defectuosos, el clúster aún puede proporcionar servicios, pero los datos de la partición defectuosa son inaccesibles.

5.4 Clúster Redis

Las instancias de Clúster Redis de DCS utilizan la implementación distribuida nativa de Redis. Las instancias de Clúster Redis tienen las siguientes características:

- Son compatibles con los clústeres nativos de Redis.
- Heredan el diseño de cliente inteligente de Redis.
- Ofrecen un rendimiento muchas veces superior al de las instancias principales/en espera.

La separación de lectura/escritura se admite mediante la configuración del cliente para las instancias de Clúster Redis. **Leer más** sobre el soporte de DCS para la separación de lectura/escritura.

NOTA

- No puede actualizar la versión de Redis para una instancia. Por ejemplo, una instancia de Clúster Redis de DCS compatible con Redis 4.0 no se puede actualizar a una instancia de Clúster Redis de DCS para Redis 5.0. Si su servicio requiere las funciones de versiones de Redis posteriores, cree una instancia de Clúster Redis de una versión posterior y, a continuación, migre los datos de la instancia anterior a la nueva.
- El método de conexión de un cliente a una instancia de Clúster Redis es diferente al de conexión de un cliente a otros tipos de instancias. Para obtener más información, véase **Conexión a Redis en un cliente**.

Arquitectura

El tipo de instancia de Clúster Redis proporcionado por DCS es compatible con el **Clúster Redis nativo**, que utiliza clientes inteligentes y una arquitectura distribuida para realizar la partición.

Tabla 5-3 enumera las especificaciones de la partición para diferentes especificaciones de instancia.

Puede personalizar el tamaño de la partición al crear una instancia de Clúster Redis. Si el tamaño de la partición no está personalizado, se utiliza el tamaño predeterminado. **Tamaño de una partición = Especificación de instancia/Número de particiones**. Por ejemplo, si una instancia de 48 GB tiene 6 particiones, el tamaño de cada partición es de 48 GB/6 = 8 GB.

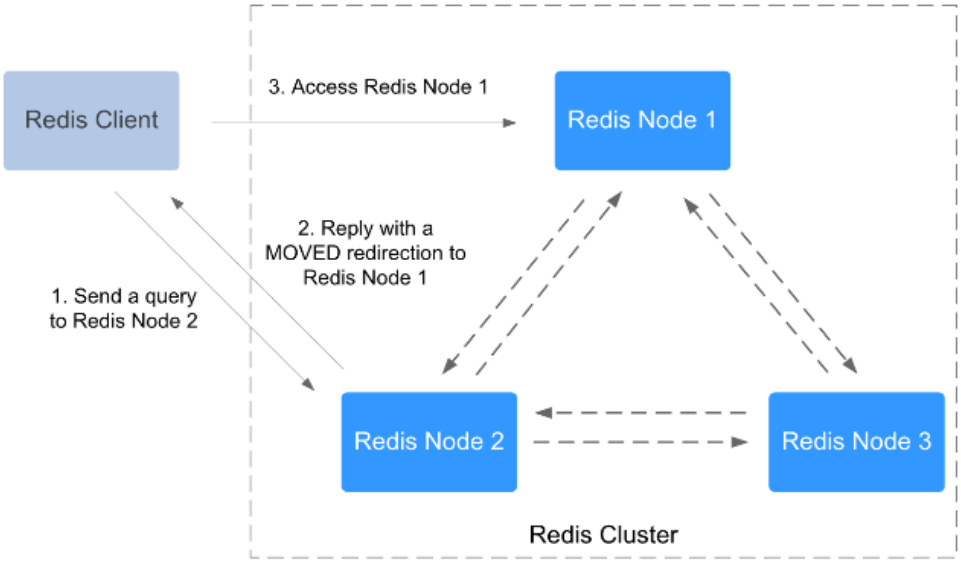
Tabla 5-3 Especificaciones de las instancias de Clúster Redis de DCS

Memoria total	Particiones
4 GB/8 GB/16 GB/24 GB/32 GB	3
48 GB	6
64 GB	8
96 GB	12
128 GB	16
192 GB	24
256 GB	32
384 GB	48
512 GB	64
768 GB	96
1024 GB	128
2048 GB	128

- Arquitectura distribuida

  Cualquier nodo de un Clúster Redis puede recibir solicitudes. Las solicitudes recibidas se redirigen luego al nodo correcto para su procesamiento. Cada nodo consta de un subconjunto de un principal y una (de forma predeterminada) o múltiples réplicas. Los roles principal o réplica se determinan a través de un algoritmo de elección.

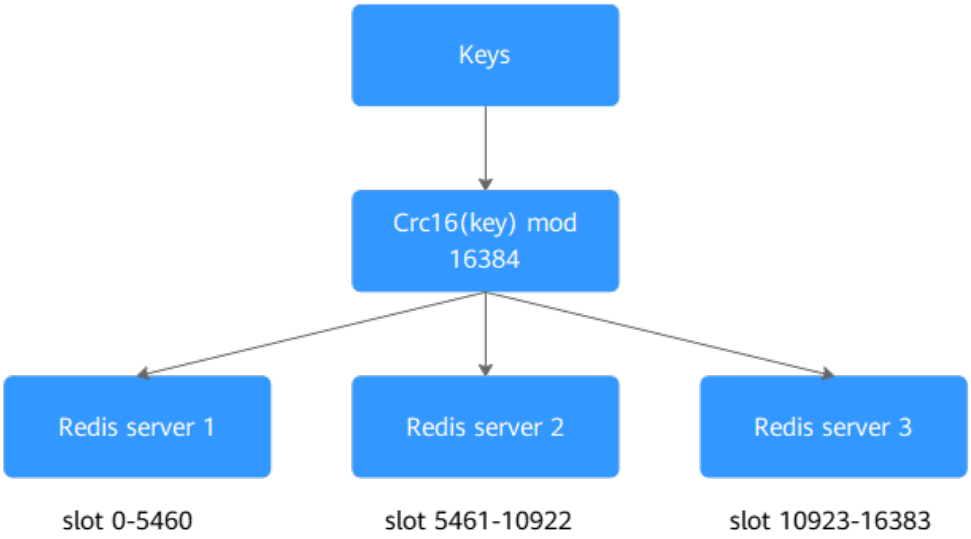
Figura 5-7 Arquitectura distribuida del Clúster Redis



- Partición previa

  En cada Clúster Redis hay 16,384 ranuras hash. La asignación entre las ranuras hash y los nodos de Redis se almacena en los servidores de Redis. Para calcular cuál es la ranura hash de la clave dada, simplemente tome el CRC16 del módulo de clave 16384.

Figura 5-8 Partición previa de Clúster Redis



NOTA

- Cada partición de un Clúster Redis es una instancia de Redis principal/en espera. Cuando el nodo principal de una partición presenta fallas, las conexiones de la partición se interrumpen en segundos y la partición deja de estar disponible. El nodo en espera se conmuta automáticamente dentro de los 15 a 30 segundos. La falla solo afecta a la partición en sí.
- Para las fallas del nodo principal en una sola partición, después de completar una conmutación por error principal/en espera, el nodo principal anterior defectuoso (entonces el nodo en espera) de la partición se recuperará más tarde. El acceso al servicio al nodo principal anterior de la partición fallará. En este caso, configure los SDK de Redis. Para obtener más información, véase **Conexión a Redis en un cliente**.

5.5 Redis de división de lectura/escritura

La separación de lectura/escritura es adecuada para escenarios con alta simultaneidad de lectura y pocas solicitudes de escritura, con el objetivo de mejorar el rendimiento de la alta simultaneidad y reducir los costos de operación.

NOTA

- El tipo de instancia de separación de lectura/escritura solo se admite en algunas regiones.
- Solo DCS Redis 4.0/5.0/6.0 de edición básica admite instancias de separación de lectura/escritura.

La separación de lectura/escritura se implementa en el lado del servidor por defecto. Los proxy distinguen entre las solicitudes de lectura y de escritura, y reenvían las solicitudes de escritura al nodo principal y las de lectura al nodo en espera. No es necesario realizar ninguna configuración en el cliente.

Cuando utilice instancias de separación de lectura/escritura, tenga en cuenta lo siguiente:

1. Las solicitudes de lectura se envían a réplicas. Hay un retraso cuando los datos se sincronizan desde el principal a las réplicas.

   Si sus servicios son sensibles al retraso, no utilice instancias de separación de lectura/escritura. En su lugar, utilice instancias de principal/en standby o de clúster.

2. La separación de lectura/escritura es adecuada cuando hay más solicitudes de lectura que solicitudes de escritura. Si hay muchas solicitudes de escritura, el principal y las réplicas pueden desconectarse, o la sincronización de datos entre ellas puede fallar después de la desconexión. Como resultado, el rendimiento de lectura se deteriora.

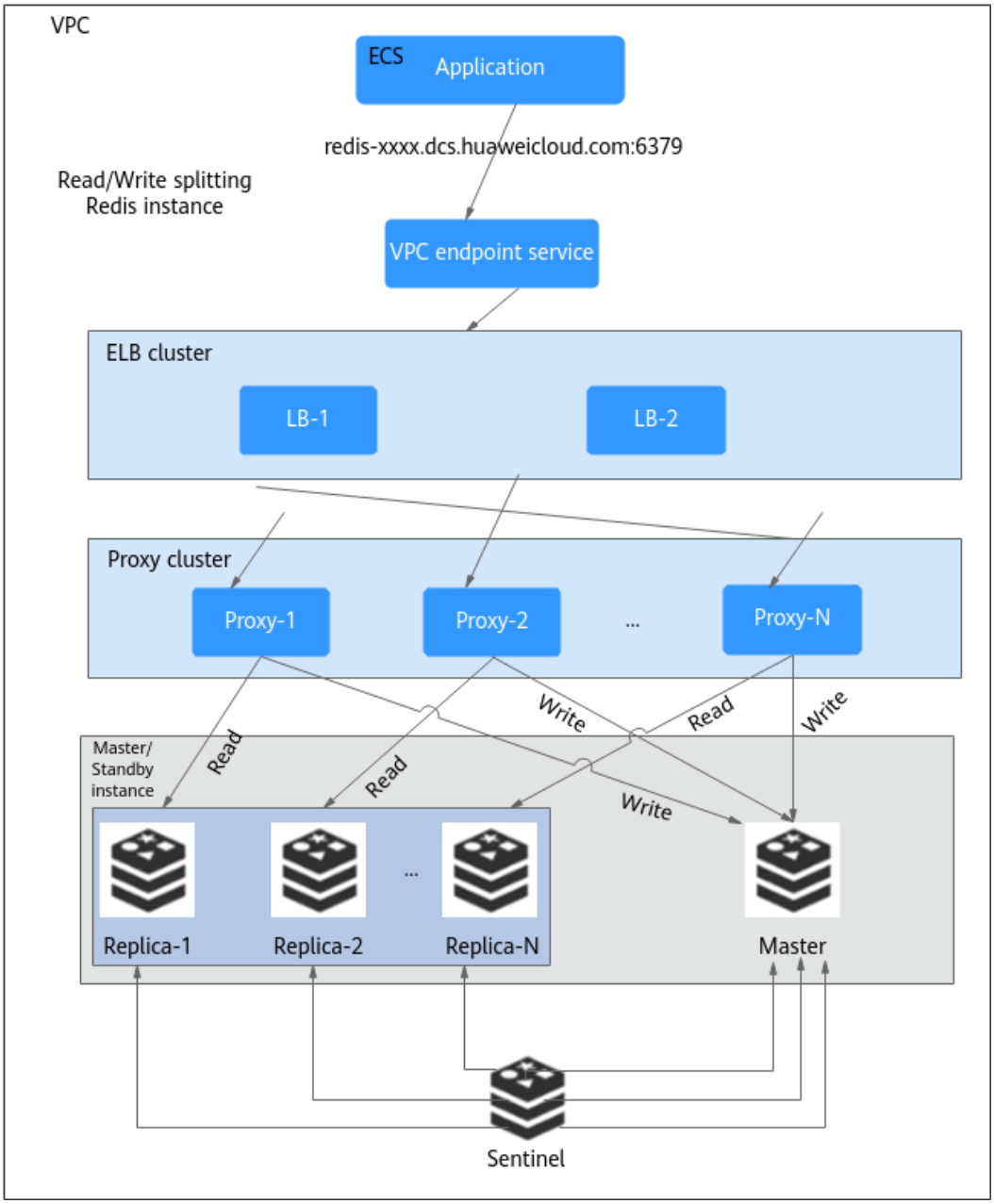
   Si sus servicios son pesados en escritura, use instancias principal/en espera o de clúster.

3. Si una réplica es defectuosa, toma algún tiempo sincronizar todos los datos del principal. Durante la sincronización, la réplica no proporciona servicios y el rendimiento de lectura de la instancia se deteriora.

   Para reducir el impacto de la interrupción, utilice una instancia con menos de 32 GB de memoria. Cuanto menor sea la memoria, menor será el tiempo para la sincronización completa de datos entre el principal y las réplicas, y menor será el impacto de la interrupción.

Arquitectura

Figura 5-9 Arquitectura de una instancia de separación de lectura/escritura



Descripción de la arquitectura:

- **Servicio de punto de conexión de VPC**

  Puede configurar su instancia de DCS compatible con Redis como un servicio de punto de conexión de VPC y acceder a la instancia en la dirección de servicio de punto de conexión de VPC.

  La dirección IP o el nombre de dominio de la instancia de separación de lectura/escritura de DCS compatible con Redis es la dirección del servicio de punto de conexión de VPC.

- **ELB**

Los balanceadores de carga se despliegan en modo HA de clúster y soportan el despliegue multi-AZ.

- **Proxy**

  Se usa un clúster proxy para distinguir entre las solicitudes de lectura y las de escritura, y reenviar las solicitudes de escritura al nodo principal y las de lectura al nodo de reserva. No es necesario configurar el cliente.

- **Clúster de Sentinel**

  Los Sentinel monitorean el estado del principal y de las réplicas. Si el nodo principal es defectuoso o anormal, se realiza una conmutación por error para garantizar que los servicios no se interrumpan.

- **Instancia principal/en espera**

  Una instancia de separación de lectura/escritura es esencialmente una instancia principal/en standby que consiste en un nodo principal y un nodo en espera. De forma predeterminada, la persistencia de datos está habilitada y los datos se sincronizan entre los dos nodos.

  Los nodos principal y en standby pueden desplegarse en diferentes AZ.

5.6 Comparación de tipos de instancias de DCS para Redis

Tabla 5-4 describe las diferencias entre los diferentes tipos de instancia de Redis en términos de características y comandos.

NOTA

DCS for Redis 3.0 ya no se proporciona. Puede utilizar DCS for Redis 4.0 o posterior.

Tabla 5-4 Diferencias entre los tipos de instancia de DCS

Concepto	Nodo único, separación de lectura/escritura, o principal/en espera	Clúster Proxy	Clúster Redis
Compatibilidad con versiones de Redis	Redis 3.0/4.0/5.0/6.0. Puede seleccionar una versión al crear una instancia. La edición profesional solo admite instancias principales/en espera.	Redis 3.0/4.0/5.0/6.0. Puede seleccionar una versión al crear una instancia.	Redis 4.0/5.0/6.0. Puede seleccionar una versión al crear una instancia.

Concepto	Nodo único, separación de lectura/ escritura, o principal/en espera	Clúster Proxy	Clúster Redis
Asistencia	● Notificaciones de espacio de claves ● Pipelining	● Pipelining, comando **MSET**, y comando **MGET** ● Comando **SCAN**, comando **KEYS** y Redis Slow Log ● Pub/Sub	● Notificaciones de espacio de claves ● Comandos **BRPOP**, **BLPOP** y **BRPOPLPUSH** ● Pub/Sub
Restricciones	Las instancias de nodo único no admiten la persistencia de datos, la copia de respaldo ni la restauración.	● El script LUA está restringido: Todas las claves deben estar en la misma ranura hash para evitar errores. **Etiquetas de hash** se recomiendan. ● Algunos comandos que contienen varias claves requieren que las claves estén en la misma ranura hash para evitar errores. **Etiquetas de hash** se recomiendan. Para obtener detalles sobre estos comandos de teclas múltiples, véase **Comandos de múltiples claves de instancias de Clúster Proxy**. ● No se admiten las notificaciones de espacio de claves.	● El script LUA está restringido: todas las claves deben estar en la misma ranura hash. **Etiquetas de hash** se recomiendan. ● El SDK del cliente debe ser compatible con Clúster Redis y ser capaz de procesar errores MOVED. ● Cuando se utiliza pipelining, el comando **MSET** o el comando **MGET**, todas las claves deben estar en el mismo intervalo hash para evitar errores. **Etiquetas de hash** se recomiendan. ● Cuando utilice las notificaciones de espacio de claves, establezca conexiones con cada servidor Redis y procese eventos en cada conexión. ● Cuando utilice un comando transversal o global como **SCAN** y **KEYS**, ejecute el comando en cada servidor de Redis.
Cliente	Cualquier cliente de Redis	Cualquier cliente Redis (sin necesidad de soportar el protocolo Clúster Redis)	Cualquier cliente que soporte el protocolo de Clúster Redis

Concepto	Nodo único, separación de lectura/escritura, o principal/en espera	Clúster Proxy	Clúster Redis
Comandos deshabilitados	**Compatibilidad de los comandos** enumera los comandos deshabilitados. **Restricciones de comandos** enumera el comando restringido para las instancias de separación de lectura/escritura.	**Compatibilidad de los comandos** enumera los comandos deshabilitados. **Restricciones de comandos** muestra el comando restringido para instancias de Clúster Proxy.	**Compatibilidad de los comandos** enumera los comandos deshabilitados. **Restricciones de comandos** muestra los comandos restringidos para instancias de Clúster Redis.

Concepto	Nodo único, separación de lectura/escritura, o principal/en espera	Clúster Proxy	Clúster Redis
Réplicas	Una instancia de nodo único solo tiene una réplica. Por defecto, una instancia principal/en espera o de separación de lectura/escritura tiene dos réplicas, una de ellas es la principal. Al crear una instancia de DCS para Redis principal/en espera o de separación de lectura/escritura, puede personalizar el número de réplicas, siendo una de ellas la principal. Actualmente, la cantidad de réplicas no se puede personalizar para instancias de DCS para Redis 3.0 y Redis 6.0 de edición profesional.	Cada partición de un clúster tiene y solo puede tener dos réplicas, siendo una de ellas la principal.	De forma predeterminada, cada partición de un clúster tiene dos réplicas. El número de réplicas de cada partición se puede personalizar, siendo una de ellas la principal. Al crear una instancia, puede establecer la cantidad de réplica en uno, lo que indica que la instancia solo tiene el nodo principal. En este caso, no se puede garantizar una alta fiabilidad de los datos.

Preguntas frecuentes relacionadas

- **¿DCS for Redis soporta múltiples bases de datos?**
- **¿Cuáles son las especificaciones de CPU de las instancias de DCS?**
- **¿DCS for Redis admite la división de lectura/escritura?**

5.7 Memcached de nodo único (descontinuado)

NOTA

DCS for Memcached ya no se proporciona. Puede usar instancias de DCS para Redis en su lugar.

En esta sección se describen las características y la arquitectura de las instancias de nodo único de DCS compatibles con Memcached.

Características

1. Baja sobrecarga del sistema y alto QPS

   Las instancias de nodo único no admiten sincronización de datos ni persistencia de datos, lo que reduce la sobrecarga del sistema y admite una mayor simultaneidad. El QPS de las instancias de nodo único de DCS para Memcached alcanza hasta 100,000.

2. Monitoreo de procesos y recuperación automática de fallas

   Con un mecanismo de monitoreo de HA, si una instancia de DCS de un nodo único resulta defectuosa, se inicia un nuevo proceso en 30 segundos para reanudar el aprovisionamiento del servicio.

3. Usabilidad lista para usar y sin persistencia de datos

   Las instancias de un nodo único de DCS se pueden usar de inmediato porque no implican la carga de datos. Si su servicio requiere un alto QPS, puede calentar los datos de antemano para evitar un fuerte impacto de simultaneidad en la base de datos backend.

4. Bajo costo y adecuado para desarrollo y pruebas

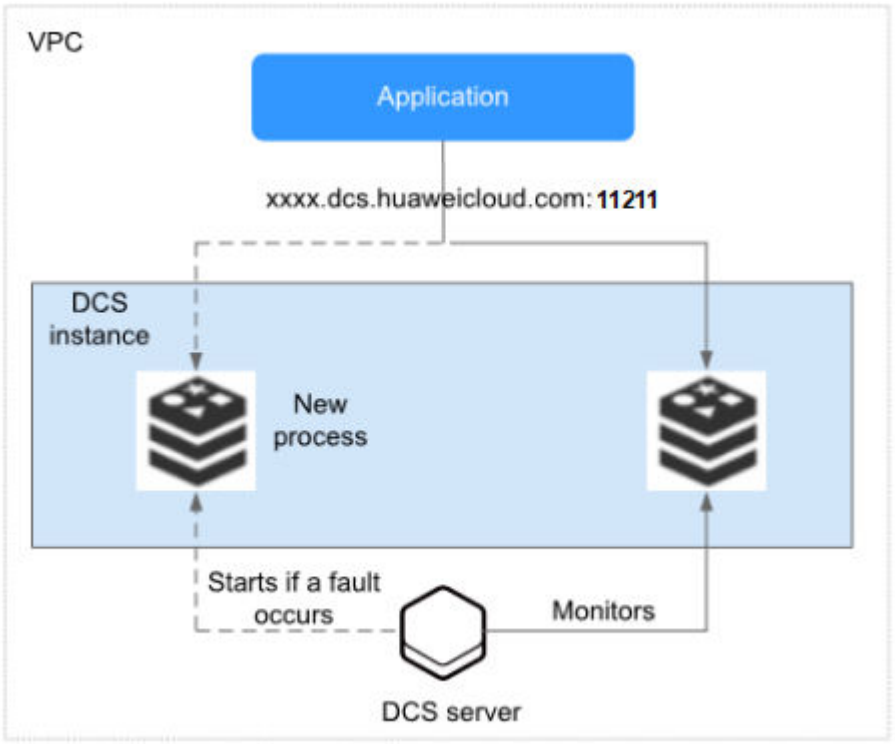
   Las instancias de un nodo único son un 40 % más baratas que las instancias de DCS principal/en espera, y son adecuadas para configurar entornos de desarrollo o pruebas.

En resumen, las instancias DCS de nodo único admiten las operaciones de lectura/escritura altamente simultáneas, pero no admiten persistencia de datos. Los datos se eliminarán después de reiniciar las instancias. Son adecuados para escenarios que no requieren persistencia de datos, como el almacenamiento en caché front-end de la base de datos, para acelerar el acceso y facilitar la carga de concurrencia fuera del back-end. Si los datos deseados no existen en la caché, las solicitudes irán a la base de datos. Al reiniciar el servicio o la instancia de DCS, puede pregenerar los datos de caché de la base de datos de disco para aliviar la presión sobre el backend durante el inicio.

Arquitectura

Figura 5-10 muestra la arquitectura de instancias de un solo nodo de DCS para Memcached.

Figura 5-10 Arquitectura de instancia de nodo único de DCS



Descripción de la arquitectura:

- **VPC**

  La VPC donde se ejecutan todos los nodos de la instancia.

  NOTA

  Las instancias de nodo único de DCS para Memcached no admiten el acceso público. El cliente y la instancia deben estar en la misma VPC con las configuraciones de reglas de grupo de seguridad.

  Para obtener más información, vea **¿Cómo configuro un grupo de seguridad?**

- **Aplicación**

  El cliente de la instancia, que es la aplicación que se ejecuta en un Elastic Cloud Server (ECS).

  Las instancias de DCS para Memcached son compatibles con el protocolo Memcached y se puede acceder a ellas con clientes de código abierto. Para obtener ejemplos de acceso a instancias de DCS con diferentes lenguajes de programación, consulte **Acceso a una instancia de DCS para Memcached**.

- **Instancia de DCS**

  Una instancia de nodo único de DCS, que tiene un solo nodo y un proceso de Memcached.

  DCS monitorea la disponibilidad de la instancia en tiempo real. Si el proceso de Memcached se vuelve defectuoso, DCS inicia un nuevo proceso para reanudar el aprovisionamiento del servicio.

  Utilice el puerto 11211 para acceder a una instancia de DCS compatible con Memcached.

5.8 Memcached principal/en espera (descontinuado)

NOTA

DCS for Memcached ya no se proporciona. Puede usar instancias de DCS para Redis en su lugar.

En esta sección se describen las instancias principal/en espera de DCS compatible con Memcached.

Características

Las instancias principal/en standby tienen mayor disponibilidad y confiabilidad que las instancias de un nodo único.

Las instancias principal/en standby de DCS compatible con Memcached tienen las siguientes características:

1. **Persistencia de los datos y alta confiabilidad**

   De forma predeterminada, la persistencia de datos está habilitada tanto por el nodo principal como por el en standby de una instancia principal/en standby de DCS para Memcached. Además, se admite la persistencia de datos para garantizar una alta fiabilidad de los datos.

   El nodo en standby de una instancia de DCS para Memcached es invisible para usted. Solo el nodo principal proporciona las operaciones de lectura/escritura de datos.

2. **Sincronización de datos**

   Los datos en los nodos principal y en espera se mantienen consistentes a través de la sincronización incremental.

   NOTA

   Después de recuperarse de una excepción de red o un fallo de nodo, las instancias principal/en standby realizan una sincronización completa para garantizar la coherencia de los datos.

3. **Conmutación automática principal/en standby**

   Si el nodo principal se vuelve defectuoso, el nodo de reserva se hace cargo en 30 segundos, sin requerir ninguna interrupción de servicio u operaciones manuales.

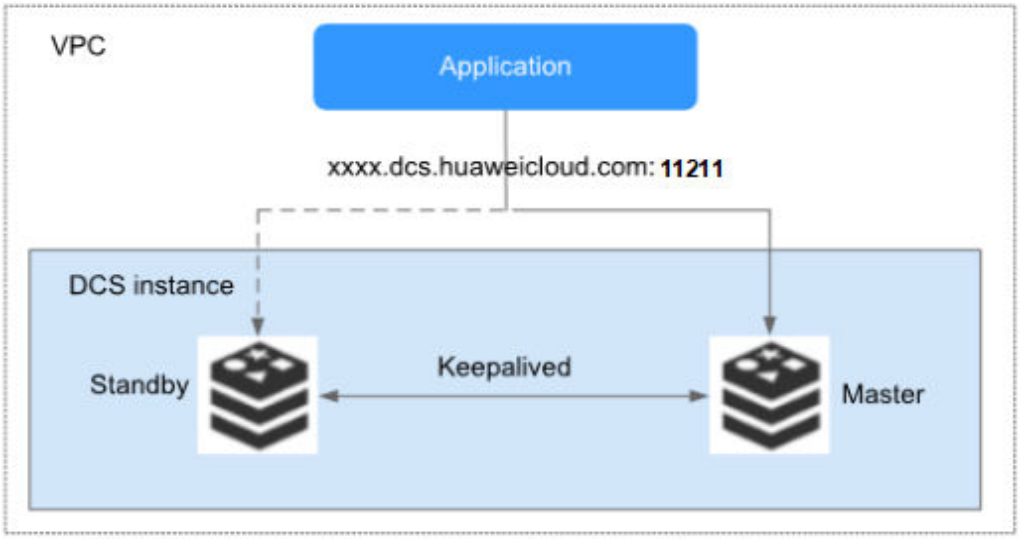
4. **Múltiples políticas de recuperación ante desastres**

   Cada instancia principal/en standby se puede desplegar en las AZ con fuentes de alimentación y redes físicamente aisladas. Las aplicaciones también se pueden desplegar entre las AZ para lograr HA tanto para datos como para aplicaciones.

Arquitectura de instancias principal/en standby de DCS compatible con Memcached

La **Figura 5-11** muestra la arquitectura de las instancias de DCS Memcached principales/en standby.

Figura 5-11 Arquitectura de instancia principal/en standby de DCS compatible con Memcached



Descripción de la arquitectura:

- **VPC**

  La VPC donde se ejecutan todos los nodos de la instancia.

  NOTA

  Las instancias principales/en standby de DCS Memcached no admiten el acceso público. El cliente y la instancia deben estar en la misma VPC con las configuraciones de reglas de grupo de seguridad.

  Para obtener más información, vea **¿Cómo configuro un grupo de seguridad?**

- **Aplicación**

  El cliente de Memcached de la instancia, que es la aplicación que se ejecuta en el ECS.

  Las instancias de DCS para Memcached son compatibles con el protocolo Memcached y se puede acceder a ellas con clientes de código abierto. Para obtener ejemplos de acceso a instancias de DCS con diferentes lenguajes de programación, consulte **Acceso a una instancia de DCS para Memcached**.

- **Instancia de DCS**

  Indica una instancia principal/en standby de DCS que tiene un nodo principal y un nodo en standby. De forma predeterminada, la persistencia de datos está habilitada y los datos se sincronizan entre los dos nodos.

  DCS monitorea la disponibilidad de la instancia en tiempo real. Si el nodo principal se vuelve defectuoso, el nodo en espera se convierte en el nodo principal y reanuda el aprovisionamiento de servicio.

  Utilice el puerto 11211 para acceder a una instancia de DCS compatible con Memcached.

6 Especificaciones de instancias de DCS

6.1 Especificaciones de las instancias de Redis 4.0 y 5.0

Esta sección describe las especificaciones de instancia de DCS para Redis 4.0 y 5.0, incluida la memoria total, la memoria disponible, el número máximo de conexiones permitidas, el ancho de banda máximo/asegurado y el rendimiento de referencia.

Las siguientes métricas están relacionadas con las especificaciones de instancia:

- Memoria usada: puede comprobar el uso de memoria de una instancia mediante la consulta de las métricas **Memory Usage** (Uso de memoria) y **Used Memory** (Memoria usada).
- Conexiones máximas: el número máximo de conexiones permitidas es el número máximo de clientes que se pueden conectar a una instancia. Para comprobar el número de conexiones a una instancia, consulte la métrica **Connected Clients** (Clientes conectados). Después de crear una instancia, esta métrica se puede cambiar modificando el parámetro **maxclients** en la página **Instance Configuration** > **Parameters** de la consola.
- QPS representa consultas por segundo, que es el número de comandos procesados por segundo. Para obtener más información sobre las pruebas de QPS, véase el **Libro blanco del rendimiento**.
- Ancho de banda: Puede ver la métrica **Flow Control Times** (Tiempos de control de flujo) para comprobar si el ancho de banda ha excedido el límite. También puede comprobar la métrica **Bandwidth Usage** (uso de ancho de banda). Esta métrica es solo para referencia, ya que puede ser superior al 100 %. Para obtener más información, véase **¿Por qué el uso de ancho de banda supera el 100 %?**

NOTA

- Las instancias de DCS compatibles con Redis 4.0 y 5.0 están disponibles en tipos de nodo único, principal/standby, Clúster Proxy, Clúster Redis y separación de lectura/escritura.
- Las instancias de DCS para Redis 4.0 y 5.0 admiten las arquitecturas x86 y Arm. Se recomienda x86. Arm no está disponible en algunas regiones.
- Las especificaciones disponibles en la consola varían según la región.

Instancias de nodo único

Las instancias de nodo único de DCS compatibles con Redis 4.0 o 5.0 admiten las arquitecturas de CPU x86 y Arm. La siguiente tabla muestra las especificaciones.

Tabla 6-1 Especificaciones de instancias de nodo único de DCS compatibles con Redis 4.0 o 5.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
0.125	0.125	10,000/10,000	40/40	80,000	x86: redis.single.xu1.tiny.128 Arm: redis.single.au1.tiny.128
0.25	0.25	10,000/10,000	80/80	80,000	x86: redis.single.xu1.tiny.256 Arm: redis.single.au1.tiny.256
0.5	0.5	10,000/10,000	80/80	80,000	x86: redis.single.xu1.tiny.512 Arm: redis.single.au1.tiny.512
1	1	10,000/50,000	80/80	80,000	x86: redis.single.xu1.large.1 Arm: redis.single.au1.large.1
2	2	10,000/50,000	128/128	80,000	x86: redis.single.xu1.large.2 Arm: redis.single.au1.large.2

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
4	4	10,000/50,000	192/192	80,000	x86: redis.single.xu1.large.4 Arm: redis.single.au1.large.4
8	8	10,000/50,000	192/192	100,000	x86: redis.single.xu1.large.8 Arm: redis.single.au1.large.8
16	16	10,000/50,000	256/256	100,000	x86: redis.single.xu1.large.16 Arm: redis.single.au1.large.16
24	24	10,000/50,000	256/256	100,000	x86: redis.single.xu1.large.24 Arm: redis.single.au1.large.24
32	32	10,000/50,000	256/256	100,000	x86: redis.single.xu1.large.32 Arm: redis.single.au1.large.32
48	48	10,000/50,000	256/256	100,000	x86: redis.single.xu1.large.48 Arm: redis.single.au1.large.48

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
64	64	10,000/50,000	384/384	100,000	x86: redis.single.xu1.large.64 Arm: redis.single.au1.large.64

Instancia principal/en espera

Las instancias principal/en espera admiten las arquitecturas de CPU x86 y Arm. Una instancia puede tener de 2 a 5 réplicas. Por ejemplo, las especificaciones de una instancia basada en Arm pueden ser **Arm | principal/en standby | 2 réplicas** o **Arm | principal/en standby | 5 réplicas**. De forma predeterminada, una instancia principal/en standby tiene un nodo principal y dos réplicas (incluida la réplica principal).

Dado el mismo tamaño de memoria, las diferencias entre las instancias principal/en espera basadas en x86, las instancias principal/en espera basadas en Arm y las instancias principal/en espera con múltiples réplicas son las siguientes:

- La memoria disponible, el número máximo de conexiones, el ancho de banda asegurado/máximo y el QPS son los mismos.
- Código de especificación: en **Tabla 6-2** solo se muestran los nombres de especificación de las instancias basadas en x86 y Arm. Los nombres de las especificaciones reflejan el número de réplicas, por ejemplo, redis.ha.au1.large.**r2**.8 (principal/en standby | Arm | 2 réplicas | 8 GB) y redis.ha.au1.large.**r3**.8 (principal/en standby | Arm | 3 réplicas | 8 GB).
- Direcciones IP: Número de direcciones IP ocupadas = Número de nodos principales x Número de réplicas. Por ejemplo:

  2 réplicas: Número de direcciones IP ocupadas = 1 x 2 = 2

  3 réplicas: Número de direcciones IP ocupadas = 1 x 3 = 3

Tabla 6-2 Especificaciones de instancias principales/en espera de DCS para Redis 4.0 o 5.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
0.125	0.125	10,000/10,000	40/40	80,000	x86: redis.ha.xu1.tiny.r2.128 Arm: redis.ha.au1.tiny.r2.128
0.25	0.25	10,000/10,000	80/80	80,000	x86: redis.ha.xu1.tiny.r2.256 Arm: redis.ha.au1.tiny.r2.256
0.5	0.5	10,000/10,000	80/80	80,000	x86: redis.ha.xu1.tiny.r2.512 Arm: redis.ha.au1.tiny.r2.512
1	1	10,000/50,000	80/80	80,000	x86: redis.ha.xu1.large.r2.1 Arm: redis.ha.au1.large.r2.1
2	2	10,000/50,000	128/128	80,000	x86: redis.ha.xu1.large.r2.2 Arm: redis.ha.au1.large.r2.2

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
4	4	10,000/50,000	192/192	80,000	x86: redis.ha.xu1.large.r2.4 Arm: redis.ha.au1.large.r2.4
8	8	10,000/50,000	192/192	100,000	x86: redis.ha.xu1.large.r2.8 Arm: redis.ha.au1.large.r2.8
16	16	10,000/50,000	256/256	100,000	x86: redis.ha.xu1.large.r2.16 Arm: redis.ha.au1.large.r2.16
24	24	10,000/50,000	256/256	100,000	x86: redis.ha.xu1.large.r2.24 Arm: redis.ha.au1.large.r2.24
32	32	10,000/50,000	256/256	100,000	x86: redis.ha.xu1.large.r2.32 Arm: redis.ha.au1.large.r2.32
48	48	10,000/50,000	256/256	100,000	x86: redis.ha.xu1.large.r2.48 Arm: redis.ha.au1.large.r2.48

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
64	64	10,000/50,000	384/384	100,000	x86: redis.ha.xu1.large.r2.64 Arm: redis.ha.au1.large.r2.64

Instancias de Clúster Proxy

Las instancias de Clúster Proxy soportan las arquitecturas de CPU x86 y Arm. **Tabla 6-3** enumera las especificaciones.

Las instancias de Clúster Proxy no admiten la personalización del número de réplicas. Cada partición tiene dos réplicas por defecto. Para obtener más información sobre el número predeterminado de particiones, consulte **Tabla 5-2**. Al comprar una instancia, puede personalizar el tamaño de una sola partición.

NOTA

- En la siguiente tabla se muestran solo las especificaciones de instancia de Clúster Proxy con particiones predeterminadas. Si personaliza las particiones, véase el número máximo de conexiones, el ancho de banda asegurado/máximo y el código de especificación del producto (variante) en la tabla **Instance Specification** de la página **Buy DCS Instance** de la consola de DCS.
- Las conexiones máximas de un clúster son para toda la instancia y no para una sola partición. Conexiones máximas de una sola partición = Conexiones máximas de una instancia/Cantidad de particiones.
- El ancho de banda máximo y el ancho de banda asegurado de un clúster son para toda la instancia y no para una sola partición. La relación entre el ancho de banda de la instancia y el ancho de banda de una sola partición es la siguiente:
  - Ancho de banda de instancia = Ancho de banda de una sola partición x Número de particiones
  - Para una instancia de clúster, si la memoria de una sola partición es de 1 GB, el ancho de banda de una sola partición es de 384 Mbit/s. Si la memoria de una sola partición es superior a 1 GB, el ancho de banda de una sola partición es de 768 Mbit/s.
  - El límite superior del ancho de banda de una instancia de Clúster Proxy es de 10,000 Mbit/s. Es decir, incluso si el ancho de banda de una sola partición multiplicada por el número de particiones es superior a 10,000 Mbit/s, el ancho de banda de la instancia sigue siendo 10,000 Mbit/s.

Tabla 6-3 Especificaciones de instancias de Clúster Proxy de DCS compatibles con Redis 4.0 y 5.0

Especificación (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
4	4	3	20,000/20,000	2304/2304	240,000	x86: redis.proxy.xu1.large.4 Arm: redis.proxy.au1.large.4
8	8	3	30,000/30,000	2304/2304	240,000	x86: redis.proxy.xu1.large.8 Arm: redis.proxy.au1.large.8
16	16	3	30,000/30,000	2304/2304	240,000	x86: redis.proxy.xu1.large.16 Arm: redis.proxy.au1.large.16
24	24	3	30,000/30,000	2304/2304	240,000	x86: redis.proxy.xu1.large.24 Arm: redis.proxy.au1.large.24
32	32	3	30,000/30,000	2304/2304	240,000	x86: redis.proxy.xu1.large.32 Arm: redis.proxy.au1.large.32

Especificación (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
48	48	6	60,000/60,000	4608/4608	480,000	x86: redis.proxy.xu1.large.48 Arm: redis.proxy.au1.large.48
64	64	8	80,000/80,000	6144/6144	640,000	x86: redis.proxy.xu1.large.64 Arm: redis.proxy.au1.large.64
96	96	12	120,000/120,000	9216/9216	960,000	x86: redis.proxy.xu1.large.96 Arm: redis.proxy.au1.large.96
128	128	16	160,000/160,000	10,000/10,000	1,280,000	x86: redis.proxy.xu1.large.128 Arm: redis.proxy.au1.large.128
192	192	24	200,000/240,000	10,000/10,000	1,920,000	x86: redis.proxy.xu1.large.192 Arm: redis.proxy.au1.large.192
256	256	32	200,000/320,000	10,000/10,000	> 2,000,000	x86: redis.proxy.xu1.large.256 Arm: redis.proxy.au1.large.256

Especificación (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
384	384	48	200,000/480,000	10,000/10,000	> 2,000,000	x86: redis.proxy.xu1.large.384 Arm: redis.proxy.au1.large.384
512	512	64	200,000/500,000	10,000/10,000	> 2,000,000	x86: redis.proxy.xu1.large.512 Arm: redis.proxy.au1.large.512
768	768	96	200,000/500,000	10,000/10,000	> 2,000,000	x86: redis.proxy.xu1.large.768 Arm: redis.proxy.au1.large.768
1024	1024	128	200,000/500,000	10,000/10,000	> 2,000,000	x86: redis.proxy.xu1.large.1024 Arm: redis.proxy.au1.large.1024
2048	2048	128	200,000/500,000	10,000/10,000	> 2,000,000	x86: redis.proxy.xu1.large.2048 Arm: redis.proxy.au1.large.2048
4096	4096	128	200,000/500,000	10,000/10,000	> 2,000,000	x86: redis.proxy.xu1.large.2048 Arm: redis.proxy.au1.large.2048

Instancias de Clúster Redis

Las instancias de Clúster Redis soportan las arquitecturas de CPU x86 y Arm. Cada instancia puede tener de 1 a 5 réplicas. Por ejemplo, las especificaciones de instancia pueden ser **Clúster Redis | 1 réplica** o **Clúster Redis | 5 réplicas**. De forma predeterminada, una instancia de Clúster Redis tiene dos réplicas. Una instancia de Clúster Redis con solo 1 réplica indica que se ha reducido la cantidad de réplicas.

Dado el mismo tamaño de memoria, las diferencias entre las instancias Clúster Redis basadas en x86, las instancias Clúster Redis basadas en Arm y las instancias Clúster Redis con múltiples réplicas son las siguientes:

- La memoria disponible, la cantidad de partición (cantidad de nodo principal), el número máximo de conexiones, el ancho de banda asegurado/máximo y el QPS son los mismos.
- Nombre de especificación: **Tabla 6-4** solo muestra los nombres de especificación de instancias basadas en x86 y Arm con 2 réplicas. Los nombres de las especificaciones reflejan el número de réplicas, por ejemplo, redis.cluster.au1.large.**r2**.24 (Clúster Redis | Arm | 2 réplicas | 24 GB) y redis.cluster.au1.large.**r3**.24 (Clúster Redis | Arm | 3 réplicas | 24 GB).
- Direcciones IP: Número de direcciones IP ocupadas = Número de particiones x Número de réplicas. Por ejemplo:

  24 GB | Clúster Redis | 3 réplicas: Número de direcciones IP ocupadas = 3 x 3 = 9
- Memoria disponible por nodo = Memoria disponible de instancia/cantidad de nodo principal

  Por ejemplo, una instancia basada en x86 de 24 GB tiene 24 GB de memoria disponible y 3 nodos principales. La memoria disponible por nodo es de 24/3 = 8 GB.
- Límite máximo de conexiones por nodo = Límite máximo de conexiones/cantidad de nodo principal. Por ejemplo:

  Por ejemplo, una instancia basada en x86 de 24 GB tiene 3 nodos principales y el límite máximo de conexiones es de 150,000. El límite máximo de conexiones por nodo = 150,000/3 = 50,000.

NOTA

- La siguiente tabla muestra solo las especificaciones de instancia de Clúster Redis con particiones predeterminadas. Si personaliza las particiones, véase el número máximo de conexiones, el ancho de banda asegurado/máximo y el código de especificación del producto (variante) en la tabla **Instance Specification** de la página **Buy DCS Instance** de la consola DCS.
- Las conexiones máximas de un clúster son para toda la instancia y no para una sola partición. Conexiones máximas de una sola partición = Conexiones máximas de una instancia/Cantidad de particiones.
- El ancho de banda máximo y el ancho de banda asegurado de un clúster son para toda la instancia y no para una sola partición. La relación entre el ancho de banda de la instancia y el ancho de banda de una sola partición es la siguiente:
  - Ancho de banda de instancia = Ancho de banda de una sola partición x Número de particiones
  - Para una instancia de clúster, si la memoria de una sola partición es de 1 GB, el ancho de banda de una sola partición es de 384 Mbit/s. Si la memoria de una sola partición es superior a 1 GB, el ancho de banda de una sola partición es de 768 Mbit/s.

Tabla 6-4 Especificaciones de instancias de Clúster Redis de DCS compatible con Redis 4.0 y 5.0

Memoria total (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
4	4	3	30,000 / 150,000	2304/2304	240,000	x86: redis.cluster.xu1.large.r2.4 Arm: redis.cluster.au1.large.r2.4
8	8	3	30,000 / 150,000	2304/2304	240,000	x86: redis.cluster.xu1.large.r2.8 Arm: redis.cluster.au1.large.r2.8
16	16	3	30,000 / 150,000	2304/2304	240,000	x86: redis.cluster.xu1.large.r2.16 Arm: redis.cluster.au1.large.r2.16
24	24	3	30,000 / 150,000	2304/2304	300,000	x86: redis.cluster.xu1.large.r2.24 Arm: redis.cluster.au1.large.r2.24
32	32	3	30,000 / 150,000	2304/2304	300,000	x86: redis.cluster.xu1.large.r2.32 Arm: redis.cluster.au1.large.r2.32

Memoria total (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
48	48	6	60,000 / 300,000	4608/4608	> 300,000	x86: redis.cluster.xu1.large.r2.48 Arm: redis.cluster.au1.large.r2.48
64	64	8	80,000 / 400,000	6144/6144	500,000	x86: redis.cluster.xu1.large.r2.64 Arm: redis.cluster.au1.large.r2.64
96	96	12	120,000 / 600,000	9216/9216	> 500,000	x86: redis.cluster.xu1.large.r2.96 Arm: redis.cluster.au1.large.r2.96
128	128	16	160,000 / 800,000	12,288/12,288	1,000,000	x86: redis.cluster.xu1.large.r2.128 Arm: redis.cluster.au1.large.r2.128
192	192	24	240,000 / 1,200,000	18,432/18,432	> 1,000,000	x86: redis.cluster.xu1.large.r2.192 Arm: redis.cluster.au1.large.r2.192
256	256	32	320,000 / 1,600,000	24,576/24,576	> 2,000,000	x86: redis.cluster.xu1.large.r2.256 Arm: redis.cluster.au1.large.r2.256

Memoria total (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
384	384	48	480,000 / 2,400,000	36,864/36,864	> 2,000,000	x86: redis.cluster.xu1.large.r2.384 Arm: redis.cluster.au1.large.r2.384
512	512	64	640,000 / 3,200,000	49,152/49,152	> 2,000,000	x86: redis.cluster.xu1.large.r2.512 Arm: redis.cluster.au1.large.r2.512
768	768	96	960,000 / 4,800,000	73,728/73,728	> 2,000,000	x86: redis.cluster.xu1.large.r2.768 Arm: redis.cluster.au1.large.r2.768
1024	1024	128	1,280,000 / 6,400,000	98,304/98,304	> 2,000,000	x86: redis.cluster.xu1.large.r2.1024 Arm: redis.cluster.au1.large.r2.1024

Instancias de separación de lectura/escritura

- Actualmente, las instancias de separación de lectura/escritura solo admiten la arquitectura de CPU x86. **Tabla 6-5** enumera las especificaciones.
- Las conexiones máximas de una instancia de separación de lectura/escritura de Redis de DCS no se pueden modificar.
- Límite de ancho de banda por servidor Redis (MB/s) = Límite de ancho de banda total (MB/s)/Número de réplicas (incluidos los principales)
- Performance de referencia por nodo (QPS) = Performance de referencia (QPS)/Número de réplicas (incluidos los principales)

- Cuando utilice instancias de separación de lectura/escritura, tenga en cuenta lo siguiente:
  a. Las solicitudes de lectura se envían a réplicas. Hay un retraso cuando los datos se sincronizan desde el principal a las réplicas.

     Si sus servicios son sensibles al retraso, no utilice instancias de separación de lectura/escritura. En su lugar, puede usar instancias de principal/en standby o de clúster.
  b. La separación de lectura/escritura es adecuada cuando hay más solicitudes de lectura que solicitudes de escritura. Si hay muchas solicitudes de escritura, el principal y las réplicas pueden desconectarse, o la sincronización de datos entre ellas puede fallar después de la desconexión. Como resultado, el rendimiento de lectura se deteriora.

     Si sus servicios son pesados en escritura, use instancias principal/en espera o de clúster.
  c. Si una réplica es defectuosa, toma algún tiempo sincronizar todos los datos del principal. Durante la sincronización, la réplica no proporciona servicios y el rendimiento de lectura de la instancia se deteriora.

     Para reducir el impacto de la interrupción, utilice una instancia con menos de 32 GB de memoria. Cuanto menor sea la memoria, menor será el tiempo para la sincronización completa de datos entre el principal y las réplicas, y menor será el impacto de la interrupción.

Tabla 6-5 Especificaciones de las instancias de separación de lectura/escritura de DCS compatibles con Redis 4.0 o 5.0

Especificación	Memoria disponible (GB)	Réplicas (incluidos los principales)	Máximo de conexiones (predeterminado/límite)	Límite de ancho de banda (MB/s)	Límite de ancho de banda por servidor de Redis (MB/s)	Rendimiento de referencia (QPS)	Rendimiento de referencia por nodo (QPS)	Código de especificación (spec_code en la API)
8	8	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.8
8	8	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.8
8	8	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.8
8	8	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.8

Especificación	Memoria disponible (GB)	Réplicas (incluidos los principales)	Máximo de conexiones (predeterminado/límite)	Límite de ancho de banda (MB/s)	Límite de ancho de banda por servidor de Redis (MB/s)	Rendimiento de referencia (QPS)	Rendimiento de referencia por nodo (QPS)	Código de especificación (spec_code en la API)
8	8	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.8
16	16	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.16
16	16	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.16
16	16	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.16
16	16	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.16
16	16	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.16
32	32	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.32
32	32	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.32
32	32	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.32

Especificación	Memoria disponible (GB)	Réplicas (incluidos los principales)	Máximo de conexiones (predeterminado/límite)	Límite de ancho de banda (MB/s)	Límite de ancho de banda por servidor de Redis (MB/s)	Rendimiento de referencia (QPS)	Rendimiento de referencia por nodo (QPS)	Código de especificación (spec_code en la API)
32	32	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.32
32	32	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.32
64	64	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.64
64	64	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.64
64	64	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.64
64	64	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.64
64	64	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.64

6.2 Especificaciones de instancia de Redis 6.0

Esta sección describe las especificaciones de instancia de DCS para Redis 6.0, incluida la memoria total, la memoria disponible, el número máximo de conexiones permitidas, el ancho de banda máximo/asegurado y el rendimiento de referencia.

Las siguientes métricas están relacionadas con las especificaciones de instancia:

- Memoria usada: puede comprobar el uso de memoria de una instancia mediante la consulta de las métricas **Memory Usage** (Uso de memoria) y **Used Memory** (Memoria usada).
- Conexiones máximas: el número máximo de conexiones permitidas es el número máximo de clientes que se pueden conectar a una instancia. Para comprobar el número de conexiones a una instancia, consulte la métrica **Connected Clients** (Clientes conectados).
- QPS representa consultas por segundo, que es el número de comandos procesados por segundo. Para obtener más información sobre las pruebas de QPS, véase el **Libro blanco del rendimiento**.
- Ancho de banda: Puede ver la métrica **Flow Control Times** (Tiempos de control de flujo) para comprobar si el ancho de banda ha excedido el límite. También puede comprobar la métrica **Bandwidth Usage** (uso de ancho de banda). Esta métrica es solo para referencia, ya que puede ser superior al 100 %. Para obtener más información, véase **¿Por qué el uso de ancho de banda supera el 100 %?**

DCS for Redis 6.0 viene en ediciones básica, profesional (rendimiento) y profesional (almacenamiento). Actualmente solo están disponibles en algunas regiones, como CN North-Beijing4 y CN South-Guangzhou. Para obtener detalles sobre la arquitectura de las instancias de DCS para Redis 6.0, véase **Tipos de instancia de DCS**.

Las instancias profesionales de DCS para Redis 6.0 solo están disponibles en el tipo principal/en espera. Todos ellos están basados en la CPU x86.

NOTA

Las especificaciones disponibles en la consola varían según la región.

Edición básica, principal/en espera

Tabla 6-6 Especificaciones de instancias principales/en espera de la edición básica de DCS compatible con Redis 6.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
0.125	0.125	10,000/10,000	40/40	80,000	redis.ha.xu1.tiny.r2.128
0.25	0.25	10,000/10,000	80/80	80,000	redis.ha.xu1.tiny.r2.256
0.5	0.5	10,000/10,000	80/80	80,000	redis.ha.xu1.tiny.r2.512
1	1	10,000/50,000	80/80	80,000	redis.ha.xu1.large.r2.1

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
2	2	10,000/50,000	128/128	80,000	redis.ha.xu1.large.r2.2
4	4	10,000/50,000	192/192	80,000	redis.ha.xu1.large.r2.4
8	8	10,000/50,000	192/192	100,000	redis.ha.xu1.large.r2.8
16	16	10,000/50,000	256/256	100,000	redis.ha.xu1.large.r2.16
24	24	10,000/50,000	256/256	100,000	redis.ha.xu1.large.r2.24
32	32	10,000/50,000	256/256	100,000	redis.ha.xu1.large.r2.32
48	48	10,000/50,000	256/256	100,000	redis.ha.xu1.large.r2.48
64	64	10,000/50,000	384/384	100,000	redis.ha.xu1.large.r2.64

Edición básica, nodo único

Tabla 6-7 Especificaciones de instancias de nodo único de la edición básica de DCS compatible con Redis 6.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
0.125	0.125	10,000/10,000	40/40	80,000	redis.single.xu1.tiny.128
0.25	0.25	10,000/10,000	80/80	80,000	redis.single.xu1.tiny.256
0.5	0.5	10,000/10,000	80/80	80,000	redis.single.xu1.tiny.512

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
1	1	10,000/50,000	80/80	80,000	redis.single.xu1.large.1
2	2	10,000/50,000	128/128	80,000	redis.single.xu1.large.2
4	4	10,000/50,000	192/192	80,000	redis.single.xu1.large.4
8	8	10,000/50,000	192/192	100,000	redis.single.xu1.large.8
16	16	10,000/50,000	256/256	100,000	redis.single.xu1.large.16
24	24	10,000/50,000	256/256	100,000	redis.single.xu1.large.24
32	32	10,000/50,000	256/256	100,000	redis.single.xu1.large.32
48	48	10,000/50,000	256/256	100,000	redis.single.xu1.large.48
64	64	10,000/50,000	384/384	100,000	redis.single.xu1.large.64

Edición básica, Clúster Proxy

Tabla 6-8 enumera las especificaciones admitidas por las instancias de Clúster Proxy.

Las instancias de Clúster Proxy no admiten la personalización del número de réplicas. Cada partición tiene dos réplicas por defecto. Para obtener más información sobre el número predeterminado de particiones, consulte **Tabla 5-2**. Al comprar una instancia, puede personalizar el tamaño de una sola partición.

NOTA

- En la siguiente tabla se muestran solo las especificaciones de instancia de Clúster Proxy con particiones predeterminadas. Si personaliza las particiones, véase el número máximo de conexiones, el ancho de banda asegurado/máximo y el código de especificación del producto (variante) en la tabla **Instance Specification** de la página **Buy DCS Instance** de la consola de DCS.
- Las conexiones máximas de un clúster son para toda la instancia y no para una sola partición. Conexiones máximas de una sola partición = Conexiones máximas de una instancia/Cantidad de particiones.
- El ancho de banda máximo y el ancho de banda asegurado de un clúster son para toda la instancia y no para una sola partición. La relación entre el ancho de banda de la instancia y el ancho de banda de una sola partición es la siguiente:
  - Ancho de banda de instancia = Ancho de banda de una sola partición x Número de particiones
  - Para una instancia de clúster, si la memoria de una sola partición es de 1 GB, el ancho de banda de una sola partición es de 384 Mbit/s. Si la memoria de una sola partición es superior a 1 GB, el ancho de banda de una sola partición es de 768 Mbit/s.
  - El límite superior del ancho de banda de una instancia de Clúster Proxy es de 10,000 Mbit/s. Es decir, incluso si el ancho de banda de una sola partición multiplicada por el número de particiones es superior a 10,000 Mbit/s, el ancho de banda de la instancia sigue siendo 10,000 Mbit/s.

Tabla 6-8 Especificaciones de instancias de Clúster Proxy de DCS para Redis 6.0 de edición básica de

Memoria total (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
4	4	3	20,000/20,000	2,304/2,304	240,000	redis.proxy.xu1.large.4
8	8	3	30,000/30,000	2,304/2,304	240,000	redis.proxy.xu1.large.8
16	16	3	30,000/30,000	2,304/2,304	240,000	redis.proxy.xu1.large.16
24	24	3	30,000/30,000	2,304/2,304	240,000	redis.proxy.xu1.large.24
32	32	3	30,000/30,000	2,304/2,304	240,000	redis.proxy.xu1.large.32
48	48	6	60,000/60,000	4,608/4,608	480,000	redis.proxy.xu1.large.48
64	64	8	80,000/80,000	6,144/6,144	640,000	redis.proxy.xu1.large.64

Memoria total (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
96	96	12	120,000/120,000	9,216/9,216	960,000	redis.proxy.xu1.large.96
128	128	16	160,000/160,000	10,000/10,000	1,280,000	redis.proxy.xu1.large.128
192	192	24	200,000/240,000	10,000/10,000	1,920,000	redis.proxy.xu1.large.192
256	256	32	200,000/320,000	10,000/10,000	> 2,000,000	redis.proxy.xu1.large.256
384	384	48	200,000/480,000	10,000/10,000	> 2,000,000	redis.proxy.xu1.large.384
512	512	64	200,000/500,000	10,000/10,000	> 2,000,000	redis.proxy.xu1.large.512
768	768	96	200,000/500,000	10,000/10,000	> 2,000,000	redis.proxy.xu1.large.768
1024	1024	128	200,000/500,000	10,000/10,000	> 2,000,000	redis.proxy.xu1.large.1024
2048	2048	128	200,000/500,000	10,000/10,000	> 2,000,000	redis.proxy.xu1.large.2048

Edición básica, Clúster de Redis

- Código de especificación: en **Tabla 6-9** solo se muestran los códigos de especificación de las instancias con 2 réplicas (por defecto). Los nombres de especificación reflejan el número de réplicas, por ejemplo, redis.cluster.xu1.large.r**2**.4 (2 réplicas | 4 GB) y redis.cluster.xu1.large.r**1**.4 (1 réplica | 4 GB).
- Direcciones IP: Número de direcciones IP ocupadas = Número de particiones x Número de réplicas. Por ejemplo:

  24 GB | Clúster Redis | 2 réplicas: Número de direcciones IP ocupadas = 3 x 2 = 6
- Memoria disponible por nodo = Memoria disponible de instancia/Cantidad de nodo principal. Por ejemplo:

  Por ejemplo, una instancia de 24 GB tiene 24 GB de memoria disponible y 3 nodos principales. La memoria disponible por nodo es de 24/3 = 8 GB.
- Límite máximo de conexiones por nodo = Límite máximo de conexiones/Cantidad de nodos principales. Por ejemplo:

Por ejemplo, una instancia de 24 GB tiene 3 nodos principales y el límite máximo de conexiones es de 150,000. El límite máximo de conexiones por nodo = 150,000/3 = 50,000.

NOTA

- La siguiente tabla muestra solo las especificaciones de instancia de Clúster Redis con particiones predeterminadas. Si personaliza las particiones, véase el número máximo de conexiones, el ancho de banda asegurado/máximo y el código de especificación del producto (variante) en la tabla **Instance Specification** de la página **Buy DCS Instance** de la consola DCS.
- Las conexiones máximas de un clúster son para toda la instancia y no para una sola partición. Conexiones máximas de una sola partición = Conexiones máximas de una instancia/Cantidad de particiones.
- El ancho de banda máximo y el ancho de banda asegurado de un clúster son para toda la instancia y no para una sola partición. La relación entre el ancho de banda de la instancia y el ancho de banda de una sola partición es la siguiente:
  - Ancho de banda de instancia = Ancho de banda de una sola partición x Número de particiones
  - Para una instancia de clúster, si la memoria de una sola partición es de 1 GB, el ancho de banda de una sola partición es de 384 Mbit/s. Si la memoria de una sola partición es superior a 1 GB, el ancho de banda de una sola partición es de 768 Mbit/s.

Tabla 6-9 Especificaciones de instancias de Clúster Redis de DCS para Redis 6.0 de edición básica

Memoria total (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
4	4	3	30,000/150,000	2304/2304	240,000	redis.cluster.xu1.large.r2.4
8	8	3	30,000/150,000	2304/2304	240,000	redis.cluster.xu1.large.r2.8
16	16	3	30,000/150,000	2304/2304	240,000	redis.cluster.xu1.large.r2.16
24	24	3	30,000/150,000	2304/2304	300,000	redis.cluster.xu1.large.r2.24
32	32	3	30,000/150,000	2304/2304	300,000	redis.cluster.xu1.large.r2.32
48	48	6	60,000/300,000	4608/4608	> 300,000	redis.cluster.xu1.large.r2.48
64	64	8	80,000/400,000	6144/6144	500,000	redis.cluster.xu1.large.r2.64

Memoria total (GB)	Memoria disponible (GB)	Particiones (nodos principales)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
96	96	12	120,000/600,000	9216/9216	> 500,000	redis.cluster.xu1.large.r2.96
128	128	16	160,000/800,000	12,288/12,288	1,000,000	redis.cluster.xu1.large.r2.128
192	192	24	240,000/1,200,000	18,432/18,432	> 1,000,000	redis.cluster.xu1.large.r2.192
256	256	32	320,000/1,600,000	24,576/24,576	> 2,000,000	redis.cluster.xu1.large.r2.256
384	384	48	480,000/2,400,000	36,864/36,864	> 2,000,000	redis.cluster.xu1.large.r2.384
512	512	64	640,000/3,200,000	49,152/49,152	> 2,000,000	redis.cluster.xu1.large.r2.512
768	768	96	960,000/4,800,000	73,728/73,728	> 2,000,000	redis.cluster.xu1.large.r2.768
1024	1024	128	1,280,000/6,400,000	98,304/98,304	> 2,000,000	redis.cluster.xu1.large.r2.1024
2048	2048	128	2,560,000/12,800,000	98,304/98,304	> 2,000,000	redis.cluster.xu1.large.r2.2048

División de lectura/escritura de edición básica

- Para obtener detalles sobre las especificaciones de las instancias de separación de lectura/escritura, véase **Tabla 6-10**.
- Las conexiones máximas de una instancia de separación de lectura/escritura de Redis de DCS no se pueden modificar.
- Límite de ancho de banda por servidor Redis (MB/s) = Límite de ancho de banda total (MB/s)/Número de réplicas (incluidos los principales)
- Performance de referencia por nodo (QPS) = Performance de referencia (QPS)/Número de réplicas (incluidos los principales)
- Cuando utilice instancias de separación de lectura/escritura, tenga en cuenta lo siguiente:

  a. Las solicitudes de lectura se envían a réplicas. Hay un retraso cuando los datos se sincronizan desde el principal a las réplicas.

Si sus servicios son sensibles al retraso, no utilice instancias de separación de lectura/escritura. En su lugar, puede usar instancias de principal/en standby o de clúster.

b. La separación de lectura/escritura es adecuada cuando hay más solicitudes de lectura que solicitudes de escritura. Si hay muchas solicitudes de escritura, el principal y las réplicas pueden desconectarse, o la sincronización de datos entre ellas puede fallar después de la desconexión. Como resultado, el rendimiento de lectura se deteriora.

   Si sus servicios son pesados en escritura, use instancias principal/en espera o de clúster.

c. Si una réplica es defectuosa, toma algún tiempo sincronizar todos los datos del principal. Durante la sincronización, la réplica no proporciona servicios y el rendimiento de lectura de la instancia se deteriora.

   Para reducir el impacto de la interrupción, utilice una instancia con menos de 32 GB de memoria. Cuanto menor sea la memoria, menor será el tiempo para la sincronización completa de datos entre el principal y las réplicas, y menor será el impacto de la interrupción.

Tabla 6-10 Especificaciones de instancia de separación de lectura/escritura de DCS para Redis 6.0

Memoria total	Memoria disponible (GB)	Réplicas (incluidos los principales)	Máximo de conexiones (predeterminado/límite)	Límite de ancho de banda (MB/s)	Límite de ancho de banda por servidor de Redis (MB/s)	Rendimiento de referencia (QPS)	Rendimiento de referencia por nodo (QPS)	Código de especificación (spec_code en la API)
8	8	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.8
8	8	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.8
8	8	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.8
8	8	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.8
8	8	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.8

Memoria total	Memoria disponible (GB)	Réplicas (incluidos los principales)	Máximo de conexiones (predeterminado/límite)	Límite de ancho de banda (MB/s)	Límite de ancho de banda por servidor de Redis (MB/s)	Rendimiento de referencia (QPS)	Rendimiento de referencia por nodo (QPS)	Código de especificación (spec_code en la API)
16	16	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.16
16	16	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.16
16	16	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.16
16	16	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.16
16	16	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.16
32	32	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.32
32	32	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.32
32	32	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.32
32	32	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.32
32	32	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.32
64	64	2	20,000	192	96	160,000	80,000	redis.ha.xu1.large.p2.64

Memoria total	Memoria disponible (GB)	Réplicas (incluidos los principales)	Máximo de conexiones (predeterminado/límite)	Límite de ancho de banda (MB/s)	Límite de ancho de banda por servidor de Redis (MB/s)	Rendimiento de referencia (QPS)	Rendimiento de referencia por nodo (QPS)	Código de especificación (spec_code en la API)
64	64	3	30,000	288	96	240,000	80,000	redis.ha.xu1.large.p3.64
64	64	4	40,000	384	96	320,000	80,000	redis.ha.xu1.large.p4.64
64	64	5	50,000	480	96	400,000	80,000	redis.ha.xu1.large.p5.64
64	64	6	60,000	576	96	480,000	80,000	redis.ha.xu1.large.p6.64

Edición profesional (Rendimiento)

Actualmente, la edición profesional (rendimiento) de DCS for Redis 6.0 admite instancias principales/en standby basadas en CPU x86.

Tabla 6-11 Especificaciones de instancias de edición profesional (rendimiento) de DCS compatible con Redis 6.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
8	8	10,000/50,000	1536/1536	400,000	redis.ha.xu1.large.enthp.8
16	16	10,000/50,000	1536/1536	400,000	redis.ha.xu1.large.enthp.16
32	32	10,000/50,000	1536/1536	400,000	redis.ha.xu1.large.enthp.32
64	64	10,000/50,000	1536/1536	400,000	redis.ha.xu1.large.enthp.64

Edición profesional (almacenamiento)

Actualmente, la edición profesional (almacenamiento) de DCS for Redis 6.0 admite instancias principales/en standby basadas en CPU x86.

Las instancias profesionales (de almacenamiento) utilizan memoria y SSD. Utilizan memoria para almacenar datos activos y SSD para almacenar todos los datos. "Memoria disponible" en la siguiente tabla es la capacidad del disco.

Tabla 6-12 Especificaciones de instancias de edición profesional (almacenamiento) de DCS compatible con Redis 6.0

Memoria total (GB)	Almacenamiento máximo (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
8	64	10,000/50,000	768/768	70,000	redis.ha.xu1.large.entst.8
16	128	10,000/50,000	768/768	70,000	redis.ha.xu1.large.entst.16
32	256	10,000/50,000	768/768	70,000	redis.ha.xu1.large.entst.32

6.3 Especificaciones de instancia de Redis 3.0 (descontinuado)

Esta sección describe las especificaciones de instancia de DCS para Redis 3.0, incluida la memoria total, la memoria disponible, el número máximo de conexiones permitidas, el ancho de banda máximo/asegurado y el rendimiento de referencia.

Las siguientes métricas están relacionadas con las especificaciones de instancia:

- Memoria usada: puede comprobar el uso de memoria de una instancia mediante la consulta de las métricas **Memory Usage** (Uso de memoria) y **Used Memory** (Memoria usada).
- Conexiones máximas: el número máximo de conexiones permitidas es el número máximo de clientes que se pueden conectar a una instancia. Para comprobar el número de conexiones a una instancia, consulte la métrica **Connected Clients** (Clientes conectados).
- QPS representa consultas por segundo, que es el número de comandos procesados por segundo.

NOTA

- Las instancias de DCS para Redis 3.0 están disponibles en los tipos de nodo único, principal/en standby y Clúster Proxy.
- DCS for Redis 3.0 ya no se proporciona. Puede utilizar DCS for Redis 4.0, 5.0 o 6.0 en su lugar.

Instancias de nodo único

Para cada instancia de nodo único de DCS para Redis, la memoria disponible es menor que la memoria total porque alguna memoria está reservada para sobrecargas del sistema, como se muestra en **Tabla 6-13**.

Tabla 6-13 Especificaciones de instancias de nodo único de DCS compatibles con Redis 3.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
2	1.5	5000/50,000	42/512	50,000	dcs.single_node
4	3.2	5000/50,000	64/1536	100,000	dcs.single_node
8	6.8	5000/50,000	64/1536	100,000	dcs.single_node
16	13.6	5000/50,000	85/3072	100,000	dcs.single_node
32	27.2	5000/50,000	85/3072	100,000	dcs.single_node
64	58.2	5000/60,000	128/5120	100,000	dcs.single_node

Instancia principal/en espera

Para cada instancia principal/en standby de DCS compatible con Redis, la memoria disponible es menor que la de una instancia de DCS compatible con Redis de nodo único porque alguna memoria está reservada para la persistencia de datos, como se muestra en **Tabla 6-14**. La memoria disponible de una instancia principal/en standby se puede ajustar para soportar tareas en segundo plano tales como persistencia de datos y sincronización principal/en standby.

Tabla 6-14 Especificaciones de instancias principal/en standby de DCS compatibles con Redis 3.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
2	1.5	5000/50,000	42/512	50,000	dcs.master_standby
4	3.2	5000/50,000	64/1536	100,000	dcs.master_standby
8	6.4	5000/50,000	64/1536	100,000	dcs.master_standby
16	12.8	5000/50,000	85/3072	100,000	dcs.master_standby
32	25.6	5000/50,000	85/3072	100,000	dcs.master_standby
64	51.2	5000/60,000	128/5120	100,000	dcs.master_standby

Instancias de Clúster Proxy

Además de una mayor memoria, las instancias de clúster cuentan con más conexiones permitidas, mayor ancho de banda permitido y más QPS que las instancias de nodo único y principal/en standby.

Tabla 6-15 Especificaciones de instancias de Clúster Proxy de DCS compatibles con Redis 3.0

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
64	64	90,000/90,000	600/5120	500,000	dcs.cluster
128	128	180,000/180,000	600/5120	500,000	dcs.cluster
256	256	240,000/240,000	600/5120	500,000	dcs.cluster
512	512	480,000/480,000	600/5120	500,000	dcs.cluster
1024	1024	960,000/960,000	600/5120	500,000	dcs.cluster

NOTA

- Las instancias de Clúster Proxy de DCS compatibles con Redis de pago por uso están disponibles en 64 GB, 128 GB y 256 GB.
- Las instancias de Clúster Proxy de DCS compatibles con Redis anuales/mensuales están disponibles en 64 GB, 128 GB, 256 GB, 512 GB y 1024 GB.

  Actualmente, solo la región CN-Hong Kong admite la facturación anual/mensual. Si necesita utilizar este modo de facturación en otras regiones, envíe un ticket de servicio en la consola para solicitar al personal técnico que active la función en el fondo.

6.4 Especificaciones de instancia de Memcached (descontinuado)

NOTA

DCS for Memcached ya no se proporciona. Puede usar instancias de DCS para Redis en su lugar.

Esta sección describe las especificaciones de instancia de DCS para Memcached, incluida la memoria total, la memoria disponible, el número máximo de conexiones permitidas, el ancho de banda máximo/asegurado y el rendimiento de referencia.

El número máximo de conexiones permitidas es el número máximo de clientes conectados a una instancia. Para comprobar el número de conexiones a una instancia, consulte la métrica **Connected Clients** (Clientes conectados).

QPS representa consultas por segundo, que es el número de comandos procesados por segundo.

NOTA

Las instancias de DCS Memcached están disponibles en tipos de nodo único y principal/en standby.

Instancias de nodo único

Para cada instancia de nodo único de DCS para Memcached, la memoria disponible es menor que la memoria total porque alguna memoria está reservada para sobrecargas del sistema, como se muestra en **Tabla 6-16**.

Tabla 6-16 Especificaciones de instancias de nodo único de DCS compatible con Memcached

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
2	1.5	5000/50,000	42/128	50,000	dcs.memcached.single_node
4	3.2	5000/50,000	64/192	100,000	dcs.memcached.single_node

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
8	6.8	5000/50,000	64/192	100,000	dcs.memcached.single_node
16	13.6	5000/50,000	85/256	100,000	dcs.memcached.single_node
32	27.2	5000/50,000	85/256	100,000	dcs.memcached.single_node
64	58.2	5000/50,000	128/384	100,000	dcs.memcached.single_node

Instancia principal/en espera

Para cada instancia principal/standby de DCS compatible con Memcached, la memoria disponible es menor que la memoria total porque cierta memoria está reservada para la persistencia de datos, como se muestra en **Tabla 6-17**. La memoria disponible de una instancia principal/en standby se puede ajustar para soportar tareas en segundo plano tales como persistencia de datos y sincronización principal/en standby.

Tabla 6-17 Especificaciones de instancias principal/en standby de DCS compatible con Memcached

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
2	1.5	5000/50,000	42/128	50,000	dcs.memcached.master_standby
4	3.2	5000/50,000	64/192	100,000	dcs.memcached.master_standby
8	6.8	5000/50,000	64/192	100,000	dcs.memcached.master_standby
16	13.6	5000/50,000	85/256	100,000	dcs.memcached.master_standby
32	27.2	5000/50,000	85/256	100,000	dcs.memcached.master_standby

Memoria total (GB)	Memoria disponible (GB)	Máximo de conexiones (predeterminado/límite) (Recuento)	Ancho de banda máximo/asegurado (Mbit/s)	Rendimiento de referencia (QPS)	Código de especificación (spec_code en la API)
64	58.2	5000/50,000	128/384	100,000	dcs.memcached.master_standby

7 Compatibilidad de los comandos

7.1 Comandos admitidos y deshabilitados por DCS for Redis 4.0

DCS for Redis 4.0 está desarrollado basado en Redis 4.0.14 y es compatible con los protocolos y comandos de código abierto. Esta sección describe la compatibilidad de DCS for Redis 4.0 con los comandos de Redis, incluidos los comandos compatibles y deshabilitados.

Las instancias de DCS para Redis admiten la mayoría de los comandos de Redis. Cualquier cliente compatible con el protocolo de Redis puede acceder a DCS.

- Por motivos de seguridad, algunos comandos de Redis están deshabilitados en DCS, como se indica en **Comandos deshabilitados por DCS for Redis 4.0**.
- Algunos comandos de Redis son compatibles con instancias DCS de clúster para operaciones de varias claves en la misma ranura. Para más detalles, véase **Restricciones de comandos**.
- Algunos comandos de Redis (como **KEYS**, **FLUSHDB** y **FLUSHALL**) tienen restricciones de uso, que se describen en **Otras restricciones del uso de comandos**.
- Se puede cambiar el nombre de algunos comandos de alto riesgo. Para más detalles, véase **Comandos que se pueden cambiar de nombre**.

Comandos compatibles con DCS for Redis 4.0

- **Tabla 7-1** y **Tabla 7-2** enumeran los comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 4.0.
- **Tabla 7-3** y **Tabla 7-4** enumeran los comandos Redis compatibles con las instancias de Clúster Proxy de DCS para Redis 4.0.
- **Tabla 7-5** y **Tabla 7-6** enumeran los comandos de Redis admitidos por la separación de lectura/escritura de instancias de DCS para Redis 4.0.

Para obtener más información sobre la sintaxis del comando, visite el **sitio web oficial de Redis**. Por ejemplo, para ver detalles sobre el comando SCAN, escriba SCAN en el cuadro de búsqueda de **esta página**.

NOTA

- Los comandos disponibles desde las versiones posteriores de Redis no son compatibles con las instancias de versiones anteriores. Ejecute un comando en redis-cli para comprobar si es compatible con DCS for Redis. Si se devuelve el mensaje "(error) ERR unknown command", el comando no es compatible.
- Para las instancias de DCS para Redis 4.0 en el modo Clúster Redis, asegúrese de que todos los comandos de una canalización se ejecuten en la misma partición.

Tabla 7-1 Comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 4.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	CLIENT KILL
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	CLIENT LIST
RANDOMKEY	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	CLIENT GETNAME
RENAME	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	CLIENT SETNAME
RENAMENX	INCR	HSET	LREM	SPOP	ZREVRANGE	CONFIG GET
RESTORE	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	MONITOR
SORT	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	SLOWLOG
TTL	MGET	HSCAN	RPOP	SUNION	ZSCORE	ROLE
TYPE	MSET	HSTRLEN	RPOPLPU	SUNIONSTORE	ZUNIONSTORE	SWAPDB

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
SCAN	MSETNX	HLEN	RPOPLPUSH	SSCAN	ZINTERSTORE	MEMORY
OBJECT	PSETEX	-	RPUSH	-	ZSCAN	CONFIG
PEXPIRE	SET	-	RPUSHX	-	ZRANGEBYLEX	COMMAND
PEXPIREAT	SETBIT	-	LPUSH	-	ZLEXCOUNT	-
KEYS	SETEX	-	-	-	ZREMRANGEBYSCORE	-
-	SETNX	-	-	-	ZREM	-
-	SETRANGE	-	-	-	-	-
-	STRLEN	-	-	-	-	-
-	BITFIELD	-	-	-	-	-

Tabla 7-2 Comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 4.0 (2)

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST
-	SUBSCRIBE	WATCH	SELECT (not supported by Redis Cluster instances)	SCRIPT KILL	GEORADIUS
-	UNSUBSCRIBE	-	-	SCRIPT LOAD	GEORADIUSBYMEMBER

Tabla 7-3 Comandos soportados por instancias de Clúster Proxy de DCS para Redis 4.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL (FLUSHALL SYNC not supported.)
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	ROLE
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	MEMORY
RENAME	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	COMMAND
RENAMENX	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	COMMAND COUNT
RESTORE	INCR	HSET	LREM	SPOP	ZREVRANGE	COMMAND GETKEYS
SORT	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	COMMAND INFO
TTL	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	CONFIG GET
TYPE	MGET	HSCAN	RPOP	SUNION	ZSCORE	CONFIG RESETSTAT

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
SCAN	MSET	HSTRLEN	RPOPLPUSH	SUNIONSTORE	ZUNIONSTORE	CONFIG REWRITE
OBJECT	MSETNX	HLEN	RPUSH	SSCAN	ZINTERSTORE	CONFIG SET
PEXPIRE	PSETEX	HKEYS	RPUSHX	-	ZSCAN	-
PEXPIREAT	SET	-	LPUSH	-	ZRANGEBYLEX	-
EXPIREAT	SETBIT	-	-	-	ZLEXCOUNT	-
KEYS	SETEX	-	-	-	ZREMRANGEBYSCORE	-
TOUCH	SETNX	-	-	-	ZREM	-
UNLINK	SETRANGE	-	-	-	ZREMRANGEBYLEX	-
RANDOMKEY	STRLEN	-	-	-	ZREVRANGEBYLEX	-
-	BITFIELD	-	-	-	-	-
-	GETBIT	-	-	-	-	-

Tabla 7-4 Comandos soportados por instancias de Clúster Proxy de DCS para Redis 4.0 (2)

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Cluster
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD	CLUSTER INFO
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH	CLUSTER NODES
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS	CLUSTER SLOTS
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST	CLUSTER ADDSLOTS

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Cluster
-	SUBSCRIBE	WATCH	SELECT	SCRIPT KILL	GEORADIUS	ASKING
-	UNSUBSCRIBE	-	CLIENT KILL	SCRIPT LOAD	GEORADIUSBYMEMBER	READONLY
-	-	-	CLIENT LIST	SCRIPT DEBUG YES| SYNC| NO	GEOSEARCH	READWRITE
-	-	-	CLIENT GETNAME	-	GEOSEARCHSTORE	-
-	-	-	CLIENT SETNAME	-	-	-

NOTA

Los comandos de clúster de la tabla anterior solo son compatibles con las instancias de Clúster Proxy creadas a partir del 1 de septiembre de 2022.

Tabla 7-5 Comandos soportados por la separación de lectura/escritura de instancias de DCS para Redis 4.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL (FLUSHALL SYNC not supported.)
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	MONITOR
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	SLOWLOG
RANDOMKEY	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	ROLE
RENAME	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	SWAPDB
RENAMENX	INCR	HSET	LREM	SPOP	ZREVRANGE	MEMORY
RESTORE	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	COMMAND
SORT	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	COMMAND COUNT
TTL	MGET	HSCAN	RPOP	SUNION	ZSCORE	COMMAND GETKEYS
TYPE	MSET	HSTRLEN	RPOPLPUSH	SUNIONSTORE	ZUNIONSTORE	COMMAND INFO
SCAN	MSETNX	HLEN	RPUSH	SSCAN	ZINTERSTORE	CONFIG GET
OBJECT	PSETEX	-	RPUSHX	-	ZSCAN	CONFIG RESETSTAT
PEXPIRE	SET	-	LPUSH	-	ZRANGEBYLEX	CONFIG REWRITE
PEXPIREAT	SETBIT	-	-	-	ZLEXCOUNT	CONFIG SET
EXPIREAT	SETEX	-	-	-	ZREMRANGEBYSCORE	-

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
KEYS	SETNX	-	-	-	ZREM	-
TOUCH	SETRANGE	-	-	-	ZREMRANGEBYLEX	-
UNLINK	STRLEN	-	-	-	ZREVRANGEBYLEX	-
-	BITFIELD	-	-	-	-	-
-	GETBIT	-	-	-	-	-

Tabla 7-6 Comandos soportados por la separación de lectura/escritura de instancias de DCS para Redis 4.0 (2)

HyperLog Log	Pub/Sub	Transactions	Connection	Scripting	Geo
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST
-	SUBSCRIBE	WATCH	SELECT	SCRIPT KILL	GEORADIUS
-	UNSUBSCRIBE	-	CLIENT KILL	SCRIPT LOAD	GEORADIUSBYMEMBER
-	-	-	CLIENT LIST	SCRIPT DEBUG YES|SYNC|NO	GEOSEARCH
-	-	-	CLIENT GETNAME	-	GEOSEARCHSTORE
-	-	-	CLIENT SETNAME	-	-

Comandos deshabilitados por DCS for Redis 4.0

A continuación se enumeran los comandos deshabilitados por DCS for Redis 4.0.

Tabla 7-7 Comandos de Redis desactivados en las instancias compatibles con Redis 4.0 de nodo único y principal/en espera

Generic (Key)	Server
MIGRATE	SLAVEOF
-	SHUTDOWN
-	LASTSAVE
-	DEBUG commands
-	SAVE
-	BGSAVE
-	BGREWRITEAOF
-	SYNC
-	PSYNC

Tabla 7-8 Comandos de Redis deshabilitados en instancias de Clúster Proxy de DCS compatibles con Redis 4.0

Generic (Key)	Server	Sorted Set
MIGRATE	BGREWRITEAOF	BZPOPMAX
MOVE	BGSAVE	BZPOPMIN
WAIT	CLIENT commands	ZPOPMAX
-	DEBUG OBJECT	ZPOPMIN
-	DEBUG SEGFAULT	-
-	LASTSAVE	-
-	PSYNC	-
-	SAVE	-
-	SHUTDOWN	-
-	SLAVEOF	-
-	LATENCY commands	-
-	MODULE commands	-
-	LOLWUT	-

Generic (Key)	Server	Sorted Set
-	SWAPDB	-
-	REPLICAOF	-
-	SYNC	-

Tabla 7-9 Comandos de Redis deshabilitados en instancias de Clúster Redis de DCS compatibles con Redis 4.0

Generic (Key)	Server	Cluster
MIGRATE	SLAVEOF	CLUSTER MEET
-	SHUTDOWN	CLUSTER FLUSHSLOTS
-	LASTSAVE	CLUSTER ADDSLOTS
-	DEBUG commands	CLUSTER DELSLOTS
-	SAVE	CLUSTER SETSLOT
-	BGSAVE	CLUSTER BUMPEPOCH
-	BGREWRITEAOF	CLUSTER SAVECONFIG
-	SYNC	CLUSTER FORGET
-	PSYNC	CLUSTER REPLICATE
-	-	CLUSTER COUNT-FAILURE-REPORTS
-	-	CLUSTER FAILOVER
-	-	CLUSTER SET-CONFIG-EPOCH
-	-	CLUSTER RESET

Tabla 7-10 Comandos de Redis deshabilitados en la separación de lectura/escritura de instancias de DCS para Redis 4.0

Generic	Server	Sorted Set
MIGRATE	BGREWRITEAOF	BZPOPMAX
WAIT	BGSAVE	BZPOPMIN
-	DEBUG OBJECT	ZPOPMAX
-	DEBUG SEGFAULT	ZPOPMIN
-	LASTSAVE	-

Generic	Server	Sorted Set
-	LOLWUT	-
-	MODULE LIST/LOAD/ UNLOAD	-
-	PSYNC	-
-	REPLICAOF	-
-	SAVE	-
-	SHUTDOWN [NOSAVE| SAVE]	-
-	SLAVEOF	-
-	SWAPDB	-
-	SYNC	-

Comandos que se pueden cambiar de nombre

Tabla 7-11 Comandos que se pueden cambiar de nombre

Comando	command, keys, flushdb, flushall, hgetall, scan, hscan, sscan y zscan Para las instancias de Clúster Proxy, también se puede cambiar el nombre de los comandos **dbsize** y **dbstats**.
Método	Véase **Renombrar comandos**.

7.2 Comandos admitidos y deshabilitados por DCS for Redis 5.0

DCS for Redis 5.0 está desarrollado basado en Redis 5.0.9 y es compatible con los protocolos y comandos de código abierto. Esta sección describe la compatibilidad de DCS for Redis 5.0 con los comandos de Redis, incluidos los comandos compatibles y deshabilitados.

Las instancias de DCS para Redis admiten la mayoría de los comandos de Redis. Cualquier cliente compatible con el protocolo de Redis puede acceder a DCS.

- Por motivos de seguridad, algunos comandos de Redis están deshabilitados en DCS, como se indica en **Comandos deshabilitados por DCS for Redis 5.0**.
- Algunos comandos de Redis son compatibles con instancias DCS de clúster para operaciones de varias claves en la misma ranura. Para más detalles, véase **Restricciones de comandos**.
- Algunos comandos de Redis (como **KEYS**, **FLUSHDB** y **FLUSHALL**) tienen restricciones de uso, que se describen en **Otras restricciones del uso de comandos**.

- Se puede cambiar el nombre de algunos comandos de alto riesgo. Para más detalles, véase **Comandos que se pueden cambiar de nombre**.

Comandos compatibles con DCS for Redis 5.0

- **Tabla 7-12** y **Tabla 7-13** enumeran los comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 5.0.
- **Tabla 7-14** y **Tabla 7-15** enumeran los comandos compatibles con instancias de Clúster Proxy de DCS para Redis 5.0.
- **Tabla 7-16** y **Tabla 7-17** enumeran los comandos de Redis admitidos por la separación de lectura/escritura de instancias de DCS para Redis 5.0.

Para obtener más información sobre la sintaxis del comando, visite el **sitio web oficial de Redis**. Por ejemplo, para ver detalles sobre el comando SCAN, escriba SCAN en el cuadro de búsqueda de **esta página**.

NOTA

- Los comandos disponibles desde las versiones posteriores de Redis no son compatibles con las instancias de versiones anteriores. Ejecute un comando en redis-cli para comprobar si es compatible con DCS for Redis. Si se devuelve el mensaje "(error) ERR unknown command", el comando no es compatible.
- Para las instancias de DCS para Redis 5.0 en el modo Clúster Redis, asegúrese de que todos los comandos de una canalización se ejecuten en la misma partición.

Tabla 7-12 Comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 5.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	KEYS
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	CLIENTKILL
RANDOMKEY	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	CLIENTLIST

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
RENAME	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	CLIENT GETNAME
RENAMENX	INCR	HSET	LREM	SPOP	ZREVRANGE	CLIENT SETNAME
RESTORE	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	CONFIG GET
SORT	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	MONITOR
TTL	MGET	HSCAN	RPOP	SUNION	ZSCORE	SLOWLOG
TYPE	MSET	HSTRLEN	RPOPLPU	SUNIONSTORE	ZUNIONSTORE	ROLE
SCAN	MSETNX	HLEN	RPOPLPUSH	SSCAN	ZINTERSTORE	SWAPDB
OBJECT	PSETEX	-	RPUSH	-	ZSCAN	MEMORY
PEXPIREAT	SET	-	RPUSHX	-	ZRANGEBYLEX	CONFIG
PEXPIRE	SETBIT	-	LPUSH	-	ZLEXCOUNT	COMMAND
KEYS	SETEX	-	-	-	ZPOPMIN	-
-	SETNX	-	-	-	ZPOPMAX	-
-	SETRANGE	-	-	-	ZREMRANGEBYSCORE	-
-	STRLEN	-	-	-	ZREM	-
-	BITFIELD	-	-	-	-	-

Tabla 7-13 Comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 5.0 (2)

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Stream
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD	XACK
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH	XADD

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Stream
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS	XCLAIM
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST	XDEL
-	SUBSCRIBE	WATCH	SELECT (not supported by Redis Cluster instances)	SCRIPT KILL	GEORADIUS	XGROUP
-	UNSUBSCRIBE	-	-	SCRIPT LOAD	GEORADIUSBYMEMBER	XINFO
-	-	-	-	-	-	XLEN
-	-	-	-	-	-	XPENDING
-	-	-	-	-	-	XRANGE
-	-	-	-	-	-	XREAD
-	-	-	-	-	-	XREADGROUP
-	-	-	-	-	-	XREVRANGE
-	-	-	-	-	-	XTRIM

Tabla 7-14 Comandos soportados por instancias de Clúster Proxy de DCS para Redis 5.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL (FLUSHALL SYNC not supported.)
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	ROLE
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	MEMORY
RENAME	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	COMMAND
RENAMENX	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	COMMAND COUNT
RESTORE	INCR	HSET	LREM	SPOP	ZREVRANGE	COMMAND GETKEYS
SORT	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	COMMAND INFO
TTL	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	CONFIG GET
TYPE	MGET	HSCAN	RPOP	SUNION	ZSCORE	CONFIG RESETSTAT
SCAN	MSET	HSTRLEN	RPOPLPUSH	SUNIONSTORE	ZUNIONSTORE	CONFIG REWRITE
OBJECT	MSETNX	HLEN	RPUSH	SSCAN	ZINTERSTORE	CONFIG SET
PEXPIRE	PSETEX	HKEYS	RPUSHX	-	ZSCAN	-
PEXPIREAT	SET	-	LPUSH	-	ZRANGEBYLEX	-
EXPIREAT	SETBIT	-	-	-	ZLEXCOUNT	-

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
KEYS	SETEX	-	-	-	ZREMRANGEBYSCORE	-
UNLINK	SETNX	-	-	-	ZREM	-
TOUCH	SETRANGE	-	-	-	ZREMRANGEBYLEX	-
RANDOMKEY	STRLEN	-	-	-	ZPOPMAX	-
-	BITFIELD	-	-	-	ZPOPMIN	-
-	GETBIT	-	-	-	BZPOPMAX	-
-	-	-	-	-	BZPOPMIN	-
-	-	-	-	-	ZREVRANGEBYLEX	-

Tabla 7-15 Comandos soportados por instancias de Clúster Proxy de DCS para Redis 5.0 (2)

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Cluster
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD	CLUSTER INFO
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH	CLUSTER NODES
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS	CLUSTER SLOTS
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST	CLUSTER ADDSLOTS
-	SUBSCRIBE	WATCH	SELECT	SCRIPT KILL	GEORADIUS	ASKING
-	UNSUBSCRIBE	-	CLIENT KILL	SCRIPT LOAD	GEORADIUSBYMEMBER	READONLY

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Cluster
-	-	-	CLIENT LIST	SCRIPT DEBUG YES|SYNC|NO	GEOSEARCH	READWRITE
-	-	-	CLIENT GETNAME	-	GEOSEARCHSTORE	-
-	-	-	CLIENT SETNAME	-	-	-

NOTA

Los comandos de clúster de la tabla anterior solo son compatibles con las instancias de Clúster Proxy creadas a partir del 1 de septiembre de 2022.

Tabla 7-16 Comandos soportados por la separación de lectura/escritura de instancias de DCS para Redis 5.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL (FLUSHALL SYNC not supported.)
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	MONITOR

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	SLOWLOG
RANDOMKEY	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	ROLE
RENAME	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	SWAPDB
RENAMENX	INCR	HSET	LREM	SPOP	ZREVRANGE	MEMORY
RESTORE	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	COMMAND
SORT	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	COMMAND COUNT
TTL	MGET	HSCAN	RPOP	SUNION	ZSCORE	COMMAND GETKEYS
TYPE	MSET	HSTRLEN	RPOPLPUSH	SUNIONSTORE	ZUNIONSTORE	COMMAND INFO
SCAN	MSETNX	HLEN	RPUSH	SSCAN	ZINTERSTORE	CONFIG GET
OBJECT	PSETEX	-	RPUSHX	-	ZSCAN	CONFIG RESETSTAT
PEXPIRE	SET	-	LPUSH	-	ZRANGEBYLEX	CONFIG REWRITE
PEXPIREAT	SETBIT	-	-	-	ZLEXCOUNT	CONFIG SET
EXPIREAT	SETEX	-	-	-	ZREMRANGEBYSCORE	-
KEYS	SETNX	-	-	-	ZREM	-
UNLINK	SETRANGE	-	-	-	ZREMRANGEBYLEX	-

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
TOUCH	STRLEN	-	-	-	BZPOPMAX	-
-	BITFIELD	-	-	-	BZPOPMIN	-
-	GETBIT	-	-	-	ZPOPMAX	-
-	-	-	-	-	ZPOPMIN	-
-	-	-	-	-	ZREVRANGEBYLEX	-

Tabla 7-17 Comandos soportados por la separación de lectura/escritura de instancias de DCS para Redis 5.0 (2)

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Stream
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD	XACK
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH	XADD
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS	XCLAIM
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST	XDEL
-	SUBSCRIBE	WATCH	SELECT	SCRIPT KILL	GEORADIUS	XGROUP
-	UNSUBSCRIBE	-	CLIENT KILL	SCRIPT LOAD	GEORADIUSBYMEMBER	XINFO
-	-	-	CLIENT LIST	SCRIPT DEBUG YES|SYNC|NO	GEOSEARCH	XLEN
-	-	-	CLIENT GETNAME	-	GEOSEARCHSTORE	XPENDING

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Stream
-	-	-	CLIENT SETNAME	-	-	XRANGE
-	-	-	-	-	-	XREAD
-	-	-	-	-	-	XREADGROUP
-	-	-	-	-	-	XREVRANGE
-	-	-	-	-	-	XTRIM

Comandos deshabilitados por DCS for Redis 5.0

A continuación se enumeran los comandos deshabilitados por DCS for Redis 5.0.

Tabla 7-18 Comandos de Redis desactivados en las instancias compatibles con Redis 5.0 de nodo único y principal/en espera

Generic (Key)	Server
MIGRATE	SLAVEOF
-	SHUTDOWN
-	LASTSAVE
-	DEBUG commands
-	SAVE
-	BGSAVE
-	BGREWRITEAOF
-	SYNC
-	PSYNC

Tabla 7-19 Comandos de Redis deshabilitados en instancias de Clúster Proxy de DCS compatibles con Redis 5.0

Generic (Key)	Server
MIGRATE	BGREWRITEAOF
MOVE	BGSAVE

Generic (Key)	Server
WAIT	CLIENT commands
-	DEBUG OBJECT
-	DEBUG SEGFAULT
-	LASTSAVE
-	PSYNC
-	SAVE
-	SHUTDOWN
-	SLAVEOF
-	LATENCY commands
-	MODULE commands
-	LOLWUT
-	SWAPDB
-	REPLICAOF
-	SYNC

Tabla 7-20 Comandos de Redis deshabilitados en instancias de Clúster Redis de DCS compatibles con Redis 5.0

Generic (Key)	Server	Cluster
MIGRATE	SLAVEOF	CLUSTER MEET
-	SHUTDOWN	CLUSTER FLUSHSLOTS
-	LASTSAVE	CLUSTER ADDSLOTS
-	DEBUG commands	CLUSTER DELSLOTS
-	SAVE	CLUSTER SETSLOT
-	BGSAVE	CLUSTER BUMPEPOCH
-	BGREWRITEAOF	CLUSTER SAVECONFIG
-	SYNC	CLUSTER FORGET
-	PSYNC	CLUSTER REPLICATE
-	-	CLUSTER COUNT-FAILURE-REPORTS
-	-	CLUSTER FAILOVER

Generic (Key)	Server	Cluster
-	-	CLUSTER SET-CONFIG-EPOCH
-	-	CLUSTER RESET

Tabla 7-21 Comandos de Redis deshabilitados en la separación de lectura/escritura de instancias de DCS para Redis 5.0

Generic	Server
MIGRATE	BGREWRITEAOF
WAIT	BGSAVE
-	DEBUG OBJECT
-	DEBUG SEGFAULT
-	LASTSAVE
-	LOLWUT
-	MODULE LIST/LOAD/UNLOAD
-	PSYNC
-	REPLICAOF
-	SAVE
-	SHUTDOWN [NOSAVE|SAVE]
-	SLAVEOF
-	SWAPDB
-	SYNC

Comandos que se pueden cambiar de nombre

Tabla 7-22 Comandos que se pueden cambiar de nombre

Comando	command, keys, flushdb, flushall, hgetall, scan, hscan, sscan y zscan Para las instancias de Clúster Proxy, también se puede cambiar el nombre de los comandos **dbsize** y **dbstats**.
Método	Véase **Renombrar comandos**.

7.3 Comandos admitidos y deshabilitados por DCS for Redis 6.0

Huawei Cloud DCS for Redis 6.0 es totalmente compatible con Redis 6 de código abierto.

Esta sección describe la compatibilidad de DCS for Redis 6.0 con los comandos de KeyDB, incluidos los comandos compatibles y deshabilitados.

Para obtener más información sobre la sintaxis de los comandos, visite el **sitio web oficial de Redis**.

Las instancias de DCS para Redis admiten la mayoría de los comandos de Redis. Cualquier cliente compatible con el protocolo de Redis puede acceder a DCS.

- Por motivos de seguridad, algunos comandos de Redis están deshabilitados en DCS, como se indica en **Comandos deshabilitados por DCS for Redis 6.0**.
- Algunos comandos de Redis son compatibles con instancias DCS de clúster para operaciones de varias claves en la misma ranura. Para más detalles, véase **Restricciones de comandos**.
- Algunos comandos de Redis (como **KEYS**, **FLUSHDB** y **FLUSHALL**) tienen restricciones de uso, que se describen en **Otras restricciones del uso de comandos**.
- Se puede cambiar el nombre de algunos comandos de alto riesgo. Para más detalles, véase **Comandos que se pueden cambiar de nombre**.

Comandos admitidos por DCS for Redis 6.0 de edición básica

- **Tabla 7-23** y **Tabla 7-24** enumeran los comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 6.0.
- **Tabla 7-25** y **Tabla 7-26** enumeran los comandos compatibles con instancias de Clúster Proxy de DCS para Redis 6.0.
- **Tabla 7-27** y **Tabla 7-28** enumeran los comandos de Redis admitidos por las instancias de separación de lectura/escritura de DCS para Redis 6.0.

Para obtener más información sobre la sintaxis del comando, visite el **sitio web oficial de Redis**. Por ejemplo, para ver detalles sobre el comando SCAN, escriba SCAN en el cuadro de búsqueda de **esta página**.

Tabla 7-23 Comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 6.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	CONFIG GET
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	MONITOR
RANDOMKEY	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	SLOWLOG
RENAME	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	ROLE
RENAMENX	INCR	HSET	LREM	SPOP	ZREVRANGE	SWAPDB
RESTORE	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	MEMORY
SORT	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	CONFIG
TTL	MGET	HSCAN	RPOP	SUNION	ZSCORE	ACL
TYPE	MSET	HSTRLEN	RPOPLPU	SUNIONSTORE	ZUNIONSTORE	COMMAND
SCAN	MSETNX	HLEN	RPOPLPUSH	SSCAN	ZINTERSTORE	-
OBJECT	PSETEX	-	RPUSH	SMISMEMBER	ZSCAN	-
PEXPIREAT	SET	-	RPUSHX	-	ZRANGEBYLEX	-
PEXPIRE	SETBIT	-	LPUSH	-	ZLEXCOUNT	-
KEYS	SETEX	-	BLMOVE	-	ZPOPMIN	-
COPY	SETNX	-	LMOVE	-	ZPOPMAX	-
-	SETRANGE	-	LPOS	-	ZREMRANGEBYSCORE	-
-	STRLEN	-	-	-	ZREM	-

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
-	BITFIELD	-	-	-	ZDIFF	-
-	BITFIELD_RO	-	-	-	ZDIFFSTORE	-
-	GETDEL	-	-	-	ZINTER	-
-	GETEX	-	-	-	ZMSCORE	-
-	-	-	-	-	ZRANDMEMBER	-
-	-	-	-	-	ZRANGESTORE	-
-	-	-	-	-	ZUNION	-

Tabla 7-24 Comandos soportados por instancias de nodo único, principal/en espera y de Clúster Redis de DCS para Redis 6.0 (2)

HyperLoglog	Pub/Sub	Transactions	Connection	Scripting	Geo	Stream
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD	XACK
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH	XADD
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS	XCLAIM
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST	XDEL
-	SUBSCRIBE	WATCH	SELECT (not supported by Redis Cluster instances)	SCRIPT KILL	GEORADIUS	XGROUP
-	UNSUBSCRIBE	-	CLIENT CACHING	SCRIPT LOAD	GEORADIUSBYMEMBER	XINFO

HyperLoglog	Pub/Sub	Transactions	Connection	Scripting	Geo	Stream
-	-	-	CLIENT GETREDIR	-	-	XLEN
-	-	-	CLIENT INFO	-	-	XPENDING
-	-	-	CLIENT TRACKING	-	-	XRANGE
-	-	-	CLIENT TRACKINGINFO	-	-	XREAD
-	-	-	CLIENT UNPAUSE	-	-	XREADGROUP
-	-	-	CLIENT KILL	-	-	XREVRANGE
-	-	-	CLIENT LIST	-	-	XTRIM
-	-	-	CLIENT GETNAME	-	-	XAUTOCLAIM
-	-	-	CLIENT SETNAME	-	-	XGROUP CREATECONSUMER
-	-	-	HELLO	-	-	-
-	-	-	RESET	-	-	-

Tabla 7-25 Comandos soportados por instancias de Clúster Proxy de DCS para Redis 6.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL (FLUSHALL SYNC not supported.)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	ROLE
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	MEMORY
RENAME	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	COMMAND
RENAMENX	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	COMMAND COUNT
RESTORE	INCR	HSET	LREM	SPOP	ZREVRANGE	COMMAND GETKEYS
SORT	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	COMMAND INFO
TTL	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	CONFIG GET
TYPE	MGET	HSCAN	RPOP	SUNION	ZSCORE	CONFIG RESETSTAT
SCAN	MSET	HSTRLEN	RPOPLPUSH	SUNIONSTORE	ZUNIONSTORE	CONFIG REWRITE
OBJECT	MSETNX	HLEN	RPUSH	SSCAN	ZINTERSTORE	CONFIG SET
PEXPIRE	PSETEX	HKEYS	RPUSHX	SMISMEMBER	ZSCAN	-

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
PEXPIREAT	SET	HRANDFIELD	LPUSH	-	ZRANGEBYLEX	-
EXPIREAT	SETBIT	-	BLMOVE	-	ZLEXCOUNT	-
KEYS	SETEX	-	LMOVE	-	ZREMRANGEBYSCORE	-
UNLINK	SETNX	-	LPOS	-	ZREM	-
TOUCH	SETRANGE	-	-	-	ZREMRANGEBYLEX	-
RANDOMKEY	STRLEN	-	-	-	ZPOPMAX	-
COPY	BITFIELD	-	-	-	ZPOPMIN	-
-	GETBIT	-	-	-	BZPOPMAX	-
-	BITFIELD_RO	-	-	-	BZPOPMIN	-
-	GETDEL	-	-	-	ZREVRANGEBYLEX	-
-	GETEX	-	-	-	ZDIFF	-
-	-	-	-	-	ZDIFFSTORE	-
-	-	-	-	-	ZINTER	-
-	-	-	-	-	ZMSCORE	-
-	-	-	-	-	ZRANDMEMBER	-
-	-	-	-	-	ZRANGESTORE	-
-	-	-	-	-	ZUNION	-

Tabla 7-26 Comandos soportados por instancias de Clúster Proxy de DCS para Redis 6.0 (2)

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo	Cluster
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD	CLUSTER INFO
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH	CLUSTER NODES
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS	CLUSTER SLOTS
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST	CLUSTER ADDSLOTS
-	SUBSCRIBE	WATCH	SELECT	SCRIPT KILL	GEORADIUS	ASKING
-	UNSUBSCRIBE	-	CLIENT KILL	SCRIPT LOAD	GEORADIUSBYMEMBER	READONLY
-	-	-	CLIENT LIST	SCRIPT DEBUG YES|SYNC|NO	GEOSEARCH	READWRITE
-	-	-	CLIENT GETNAME	-	GEOSEARCHSTORE	-
-	-	-	CLIENT SETNAME	-	-	-
-	-	-	HELLO	-	-	-

Tabla 7-27 Comandos admitidos por instancias de separación de lectura/escritura de DCS para Redis 6.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL (FLUSHALL SYNC not supported.)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	MONITOR
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	SLOWLOG
RANDOMKEY	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	ROLE
RENAME	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	SWAPDB
RENAMENX	INCR	HSET	LREM	SPOP	ZREVRANGE	MEMORY
RESTORE	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	COMMAND
SORT	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	COMMAND COUNT
TTL	MGET	HSCAN	RPOP	SUNION	ZSCORE	COMMAND GETKEYS
TYPE	MSET	HSTRLEN	RPOPLPUSH	SUNIONSTORE	ZUNIONSTORE	COMMAND INFO
SCAN	MSETNX	HLEN	RPUSH	SSCAN	ZINTERSTORE	CONFIG GET
OBJECT	PSETEX	HRANDFIELD	RPUSHX	SMISMEMBER	ZSCAN	CONFIG RESETSTAT

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
PEXPIRE	SET	-	LPUSH	-	ZRANGEBYLEX	CONFIG REWRITE
PEXPIREAT	SETBIT	-	BLMOVE	-	ZLEXCOUNT	CONFIG SET
EXPIREAT	SETEX	-	LMOVE	-	ZREMRANGEBYSCORE	-
KEYS	SETNX	-	LPOS	-	ZREM	-
UNLINK	SETRANGE	-	-	-	ZREMRANGEBYLEX	-
TOUCH	STRLEN	-	-	-	BZPOPMAX	-
COPY	BITFIELD	-	-	-	BZPOPMIN	-
-	GETBIT	-	-	-	ZPOPMAX	-
-	BITFIELD_RO	-	-	-	ZPOPMIN	-
-	GETDEL	-	-	-	ZREVRANGEBYLEX	-
-	GETEX	-	-	-	ZDIFF	-
-	-	-	-	-	ZDIFFSTORE	-
-	-	-	-	-	ZINTER	-
-	-	-	-	-	ZMSCORE	-
-	-	-	-	-	ZRANDMEMBER	-
-	-	-	-	-	ZRANGESTORE	-
-	-	-	-	-	ZUNION	-

Tabla 7-28 Comandos admitidos por instancias de separación de lectura/escritura de DCS para Redis 6.0 (2)

HyperLoglog	Pub/Sub	Transactions	Connection	Scripting	Geo	Stream
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD	XAUTOCLAIM
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH	XGROUP CREATE CONSUMER
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS	XACK
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST	XADD
-	SUBSCRIBE	WATCH	SELECT	SCRIPT KILL	GEORADIUS	XCLAIM
-	UNSUBSCRIBE	-	CLIENT KILL	SCRIPT LOAD	GEORADIUSBYMEMBER	XDEL
-	-	-	CLIENT LIST	SCRIPT DEBUG YES|SYNC|NO	GEOSEARCH	XGROUP
-	-	-	CLIENT GETNAME	-	GEOSEARCHSTORE	XINFO
-	-	-	CLIENT SETNAME	-	-	XLEN
-	-	-	HELLO	-	-	XPENDING
-	-	-	-	-	-	XRANGE
-	-	-	-	-	-	XREAD
-	-	-	-	-	-	XREADGROUP
-	-	-	-	-	-	XREVRANGE
-	-	-	-	-	-	XTRIM

Comandos admitidos por DCS for Redis 6.0 de edición profesional

A continuación se enumeran los comandos compatibles con DCS for Redis 6.0 de edición profesional.

Tabla 7-29 Comandos compatibles con DCS for Redis 6.0 de edición profesional (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
COPY	APPEND	HDEL	BLMOVE	SADD	BZPOPMAX	FLUSHALL
DEL	BITCOUNT	HEXISTS	LINDEX	SCARD	BZPOPMIN	FLUSHDB
DUMP	BITOP	HGET	LINSERT	SDIFF	ZADD	DBSIZE
EXISTS	BITPOS	HGETALL	LLEN	SDIFFSTORE	ZCARD	TIME
EXPIRE	BITFIELD	HINCRBY	LPOP	SINTER	ZCOUNT	INFO
MOVE	DECR	HINCRBYFLOAT	LPUSHX	SINTERSTORE	ZDIFF	CLIENT KILL
PERSIST	DECRBY	HKEYS	LRANGE	SISMEMBER	ZDIFFSTORE	CLIENT LIST
PTTL	GET	HMGET	LREM	SMEMBERS	ZINCRBY	CLIENT GETNAME
RANDOMKEY	GETRANGE	HMSET	LSET	SMOVE	ZINTER	CLIENT SETNAME
RENAME	GETSET	HSET	LTRIM	SPOP	ZINTERSTORE	CONFIG GET
RENAMENX	GETDEL	HSETNX	RPOP	SRANDMEMBER	ZLEXCOUNT	MONITOR
SORT	GETEX	HVALS	LMOVE	SREM	ZMSCORE	SLOWLOG
TTL	INCR	HSCAN	RPOPLPUSH	SUNION	ZPOPMAX	ROLE
TYPE	INCRBY	HSTRLEN	RPUSH	SUNIONSTORE	ZPOPMIN	SWAPDB
SCAN	INCRBYFLOAT	HLEN	RPUSHX	SSCAN	ZRANDMEMBER	MEMORY
PEXPIREAT	MGET	HRANDFIELD	LPUSH	SMISMEMBER	ZRANGE	LASTSAVE

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
PEXPIRE	MSET	-	BLPOP	-	ZRANGEBYLEX	REPLCONF
OBJECT ENCODING	MSETNX	-	BRPOP	-	ZRANGEBYSCORE	LASTSAVE
OBJECT FREQ	PSETEX	-	BRPOPLPUSH	-	ZRANGESTORE	COMMAND
OBJECT IDLETIME	SET	-	LPOS	-	ZRANK	COMMAND COUNT
OBJECT REFCOUNT	SETBIT	-	-	-	ZREM	COMMAND GETKEYS
RESTORE	SETEX	-	-	-	ZREMRANGEBYLEX	COMMAND INFO
TOUCH	SETNX	-	-	-	ZREMRANGEBYRANK	CONFIG
UNLINK	SETRANGE	-	-	-	ZREMRANGEBYSCORE	-
EXPIREAT	STRLEN	-	-	-	ZREVRANGE	-
KEYS	SUBSTR	-	-	-	ZREVRANGEBYLEX	-
WAIT	-	-	-	-	ZREVRANGEBYSCORE	-
-	-	-	-	-	ZREVRANK	-
-	-	-	-	-	ZSCAN	-
-	-	-	-	-	ZSCORE	-
-	-	-	-	-	ZUNION	-
-	-	-	-	-	ZUNIONSTORE	-

Tabla 7-30 Comandos compatibles con DCS for Redis 6.0 de edición profesional (2)

HyperLoglog	Pub/Sub	Connection	Scripting	Geo	Stream	Bitmaps
PFADD	PSUBSCRIBE	AUTH	EVAL	GEOADD	XACK	BITCOUNT
PFCOUNT	PUBLISH	CLIENT CACHING	EVALSHA	GEODIST	XADD	BITFIELD
PFMERGE	PUBSUB	CLIENT GETNAME	SCRIPT DEBUG	GEOHASH	XAUTOCLAIM	BITFIELD_RO
PFSELFTEST	PUNSUBSCRIBE	CLIENT GETREDIR	SCRIPT EXISTS	GEOPOS	XCLAIM	BITOP
-	SUBSCRIBE	CLIENT ID	SCRIPT FLUSH	GEORADIUS	XDEL	BITPOS
-	UNSUBSCRIBE	CLIENT INFO	SCRIPT KILL	GEORADIUSBYMEMBER	XGROUP	GETBIT
-	-	CLIENT KILL	SCRIPT LOAD	GEORADIUSBYMEMBER_RO	XINFO	SETBIT
-	-	CLIENT LIST	-	GEORADIUS_RO	XLEN	-
-	-	CLIENT PAUSE	-	GEOSEARCH	XPENDING	-
-	-	CLIENT REPLY	-	GEOSEARCHSTORE	XRANGE	-
-	-	CLIENT SETNAME	-	-	XREAD	-
-	-	CLIENT TRACKING	-	-	XREADGROUP	-
-	-	CLIENT TRACKINGINFO	-	-	XREVRANGE	-
-	-	CLIENT UNBLOCK	-	-	XSETID	-

HyperLoglog	Pub/Sub	Connection	Scripting	Geo	Stream	Bitmaps
-	-	CLIENT UNPAUSE	-	-	XTRIM	-
-	-	ECHO	-	-	-	-
-	-	HELLO	-	-	-	-
-	-	PING	-	-	-	-
-	-	QUIT	-	-	-	-
-	-	RESET	-	-	-	-
-	-	SELECT	-	-	-	-

Comandos deshabilitados por DCS for Redis 6.0

Tabla 7-31 Comandos deshabilitados en instancias de nodo único, principal/en espera y Clúster Redis de DCS para Redis 6.0

Generic (Key)	Server	Cluster
MIGRATE	SLAVEOF	CLUSTER MEET
-	SHUTDOWN	CLUSTER FLUSHSLOTS
-	LASTSAVE	CLUSTER ADDSLOTS
-	DEBUG commands	CLUSTER DELSLOTS
-	SAVE	CLUSTER SETSLOT
-	BGSAVE	CLUSTER BUMPEPOCH
-	BGREWRITEAOF	CLUSTER SAVECONFIG
-	SYNC	CLUSTER FORGET
-	PSYNC	CLUSTER REPLICATE
-	-	CLUSTER COUNT-FAILURE-REPORTS
-	-	CLUSTER FAILOVER
-	-	CLUSTER SET-CONFIG-EPOCH
-	-	CLUSTER RESET

Tabla 7-32 Comandos de Redis deshabilitados en instancias de Clúster Proxy de DCS compatibles con Redis 6.0

Generic (Key)	Server	Connection
MIGRATE	BGREWRITEAOF	CLIENT CACHING
MOVE	BGSAVE	CLIENT GETREDIR
WAIT	CLIENT commands	CLIENT INFO
-	DEBUG OBJECT	CLIENT TRACKING
-	DEBUG SEGFAULT	CLIENT TRACKINGINFO
-	LASTSAVE	CLIENT UNPAUSE
-	PSYNC	RESET
-	SAVE	-
-	SHUTDOWN	-
-	SLAVEOF	-
-	LATENCY commands	-
-	MODULE commands	-
-	LOLWUT	-
-	SWAPDB	-
-	REPLICAOF	-
-	SYNC	-
-	ACL	-
-	FAILOVER	-

Tabla 7-33 Comandos de Redis deshabilitados en la separación de lectura/escritura de instancias de DCS para Redis 6.0

Generic	Server	Connection
MIGRATE	BGREWRITEAOF	CLIENT CACHING
WAIT	BGSAVE	CLIENT GETREDIR
-	DEBUG OBJECT	CLIENT INFO
-	DEBUG SEGFAULT	CLIENT TRACKING
-	LASTSAVE	CLIENT TRACKINGINFO
-	LOLWUT	CLIENT UNPAUSE

Generic	Server	Connection
-	MODULE LIST/LOAD/ UNLOAD	RESET
-	PSYNC	-
-	REPLICAOF	-
-	SAVE	-
-	SHUTDOWN [NOSAVE| SAVE]	-
-	SLAVEOF	-
-	SWAPDB	-
-	SYNC	-
-	ACL	-
-	FAILOVER	-

Tabla 7-34 Comandos de Redis desactivados en instancias de DCS para Redis 6.0 de edición profesional

Generic (Key)	Server	HyperLoglog
MIGRATE	SLAVEOF	PFDEBUG
-	SHUTDOWN	-
-	SAVE	-
-	BGSAVE	-
-	BGREWRITEAOF	-
-	SYNC	-
-	PSYNC	-
-	REPLICAOF	-

Comandos que se pueden cambiar de nombre

Tabla 7-35 Comandos que se pueden cambiar de nombre

Comando	command, keys, flushdb, flushall, hgetall, scan, hscan, sscan y zscan Para las instancias de Clúster Proxy, también se puede cambiar el nombre de los comandos **dbsize** y **dbstats**.

Método	Véase **Renombrar comandos**.

7.4 Comandos soportados y deshabilitados en Web CLI

Web CLI es una herramienta de línea de comandos proporcionada en la consola de DCS. Esta sección describe la compatibilidad de la CLI web con los comandos de Redis, incluidos los comandos compatibles y deshabilitados. Actualmente, solo DCS for Redis 6.0/5.0/4.0 soporta Web CLI.

NOTA

- Las claves y los valores no pueden contener espacios.
- Si el valor está vacío, se devuelve **nil** después de ejecutar el comando **GET**.

Comandos admitidos en la CLI web

A continuación se enumeran los comandos admitidos cuando se usa la CLI web. Para obtener más información sobre la sintaxis del comando, visite el **sitio web oficial de Redis**. Por ejemplo, para ver detalles sobre el comando SCAN, escriba SCAN en el cuadro de búsqueda de **esta página**.

Tabla 7-36 Comandos soportados por CLI Web (1)

Generic (Key)	String	List	Set	Sorted Set	Server
DEL	APPEND	RPUSH	SADD	ZADD	FLUSHALL
OBJECT	BITCOUNT	RPUSHX	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE
EXPIRE	BITPOS	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	LLEN	SINTERSTORE	ZRANGEBYSCORE	CLIENT KILL
PTTL	GET	LPOP	SISMEMBER	ZRANK	CLIENT LIST
RANDOMKEY	GETRANGE	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	CLIENT GETNAME
RENAME	GETSET	LRANGE	SMOVE	ZREMRANGEBYCORE	CLIENT SETNAME
RENAMENX	INCR	LREM	SPOP	ZREVRANGE	CONFIG GET

Generic (Key)	String	List	Set	Sorted Set	Server
SCAN	INCRBY	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	SLOWLOG
SORT	INCRBYFLOAT	LTRIM	SREM	ZREVRANK	ROLE
TTL	MGET	RPOP	SUNION	ZSCORE	SWAPDB
TYPE	MSET	RPOPLPU	SUNIONSTORE	ZUNIONSTORE	MEMORY
-	MSETNX	RPOPLPUSH	SSCAN	ZINTERSTORE	-
-	PSETEX	-	-	ZSCAN	-
-	SET	-	-	ZRANGEBYLEX	-
-	SETBIT	-	-	ZLEXCOUNT	-
-	SETEX	-	-	-	-
-	SETNX	-	-	-	-
-	SETRANGE	-	-	-	-
-	STRLEN	-	-	-	-
-	BITFIELD	-	-	-	-

Tabla 7-37 Comandos soportados por CLI Web (2)

Hash	HyperLoglog	Connection	Scripting	Geo	Pub/Sub
HDEL	PFADD	AUTH	EVAL	GEOADD	UNSUBSCRIBE
HEXISTS	PFCOUNT	ECHO	EVALSHA	GEOHASH	PUBLISH
HGET	PFMERGE	PING	SCRIPT EXISTS	GEOPOS	PUBSUB
HGETALL	-	QUIT	SCRIPT FLUSH	GEODIST	PUNSUBSCRIBE
HINCRBY	-	-	SCRIPT KILL	GEORADIUS	-

Hash	HyperLoglog	Connection	Scripting	Geo	Pub/Sub
HINCRBYFLOAT	-	-	SCRIPT LOAD	GEORADIUSBYMEMBER	-
HKEYS	-	-	-	-	-
HMGET	-	-	-	-	-
HMSET	-	-	-	-	-
HSET	-	-	-	-	-
HSETNX	-	-	-	-	-
HVALS	-	-	-	-	-
HSCAN	-	-	-	-	-
HSTRLEN	-	-	-	-	-

Comandos deshabilitados en la CLI web

A continuación se enumeran los comandos deshabilitados cuando se usa la CLI web.

Tabla 7-38 Comandos deshabilitados en Web CLI (1)

Generic (Key)	Server	Transactions	Cluster
MIGRATE	SLAVEOF	UNWATCH	CLUSTER MEET
WAIT	SHUTDOWN	REPLICAOF	CLUSTER FLUSHSLOTS
DUMP	DEBUG commands	DISCARD	CLUSTER ADDSLOTS
RESTORE	CONFIG SET	EXEC	CLUSTER DELSLOTS
-	CONFIG REWRITE	MULTI	CLUSTER SETSLOT
-	CONFIG RESETSTAT	WATCH	CLUSTER BUMPEPOCH
-	SAVE	-	CLUSTER SAVECONFIG
-	BGSAVE	-	CLUSTER FORGET
-	BGREWRITEAOF	-	CLUSTER REPLICATE
-	COMMAND	-	CLUSTER COUNT-FAILURE-REPORTS
-	KEYS	-	CLUSTER FAILOVER

Generic (Key)	Server	Transactions	Cluster
-	MONITOR	-	CLUSTER SET-CONFIG-EPOCH
-	SYNC	-	CLUSTER RESET
-	PSYNC	-	-
-	ACL	-	-
-	MODULE	-	-

Tabla 7-39 Comandos deshabilitados en Web CLI (2)

List	Connection	Sorted Set	Pub/Sub
BLPOP	SELECT	BZPOPMAX	PSUBSCRIBE
BRPOP	-	BZPOPMIN	SUBSCRIBE
BLMOVE	-	BZMPOP	-
BRPOPLPUSH	-	-	-
BLMPOP	-	-	-

7.5 Restricciones de comandos

Las instancias de Clúster Redis admiten algunos comandos de Redis para operaciones de varias claves en la misma ranura. En **Tabla 7-40** se enumeran los comandos restringidos.

Algunos comandos admiten varias teclas pero no el acceso entre ranuras. Para más detalles, véase **Tabla 7-42**. En **Tabla 7-41** se enumeran los comandos restringidos.

Tabla 7-43 enumera los comandos restringidos para instancias de separación de lectura/escritura.

NOTA

Mientras se ejecutan comandos que tardan mucho en ejecutarse, como **FLUSHALL**, las instancias DCS pueden no responder a otros comandos y pueden cambiar al estado defectuoso. Después de que el comando termine de ejecutarse, la instancia volverá a la normalidad.

Comandos restringidos de Redis en las instancias de Clúster Redis de DCS

Tabla 7-40 Comandos restringidos de Redis en las instancias de Clúster Redis de DCS

Categoría	Descripción
Set (Conjunto)	

Categoría	Descripción
SINTER	Devuelve los miembros del conjunto resultante de la intersección de todos los conjuntos dados.
SINTERSTORE	Igual a **SINTER**, pero en lugar de devolver el conjunto de resultados, se almacena en *destination*.
SUNION	Devuelve los miembros del conjunto resultante de la unión de todos los conjuntos dados.
SUNIONSTORE	Igual a **SUNION**, pero en lugar de devolver el conjunto de resultados, se almacena en *destination*.
SDIFF	Devuelve los miembros del conjunto resultante de la diferencia entre el primer conjunto y todos los conjuntos sucesivos.
SDIFFSTORE	Igual a **SDIFF**, pero en lugar de devolver el conjunto de resultados, se almacena en *destination*.
SMOVE	Mueve **member** del conjunto en **source** al conjunto en *destination*.
Sorted Set (Conjunto ordenado)	
ZUNIONSTORE	Calcula la unión de conjuntos ordenados de *numkeys* dadas por las claves especificadas.
ZINTERSTORE	Calcula la intersección de conjuntos ordenados por *numkeys* dadas por las claves especificadas.
HyperLogLog	
PFCOUNT	Devuelve la cardinalidad aproximada calculada por la estructura de datos HyperLogLog almacenada en la variable especificada.
PFMERGE	Combina varios valores HyperLogLog en un valor único.
Key	
RENAME	Cambia el nombre de *key* a *newkey*.
RENAMENX	Cambia el nombre de *key* a *newkey* si *newkey* aún no existe.
BITOP	Realiza una operación a nivel de bits entre varias claves (que contienen valores de cadena) y almacena el resultado en la clave de destino.
RPOPLPUSH	Devuelve y elimina el último elemento (tail) de la lista almacenada en el origen y lo empuja al primer elemento (head) de la lista almacenada en *destination*.
String	
MSETNX	Establece las claves dadas en sus respectivos valores.

Comandos de Redis restringidos en instancias de Clúster Proxy de DCS

Tabla 7-41 Comandos de Redis restringidos en instancias de Clúster Proxy de DCS

Categoría	Comando	Restricción
Sets	SMOVE	Para una instancia de Clúster Proxy, las claves de origen y destino deben estar en la misma ranura.
Sorted sets	BZPOPMAX	Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura. No está compatible con instancias de Clúster Proxy de DCS para Redis 4.0.
	BZPOPMIN	
Geo	GEORADIUS	• Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura. • Para una instancia de Clúster Proxy con múltiples bases de datos, la opción **STORE** no es compatible.
	GEORADIUSBYMEMBER	
	GEOSEARCHSTORE	
Connection	CLIENT KILL	• Solo se admiten los dos formatos siguientes: – CLIENT KILL ip:port – CLIENT KILL ADDR ip:port • El campo **id** tiene un valor aleatorio y no cumple con el requisito idc1<idc2→Tc1<Tc2.
	CLIENT LIST	• Solo se admiten los dos formatos siguientes: – CLIENT LIST – CLIENT LIST [TYPE normal|master|replica|pubsub] • El campo **id** tiene un valor aleatorio y no cumple con el requisito idc1<idc2→Tc1<Tc2.
	SELECT index	Multi-DB de instancias de Clúster Proxy se puede implementar cambiando las claves. No se recomienda esta solución. Para obtener más información sobre las restricciones de varias bases de datos en instancias de clúster proxy, véase **¿Cuáles son las limitaciones de la implementación de varias bases de datos en una instancia de Clúster Proxy?**

Categoría	Comando	Restricción
HyperLogLog	PFCOUNT	Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura.
	PFMERGE	
Keys	RENAME	Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura.
	RENAMENX	

	SCAN	● Las instancias de Clúster Proxy no soportan el comando **SCAN** en canalizaciones.
		● Para una instancia de Clúster Proxy, puede ejecutar el comando **SCAN** a una partición específica agregando *ip*:*port*. (Las direcciones IP y los puertos de partición se pueden consultar a través del comando **icluster nodes**)
		Si el comando **SCAN** ejecutado en la partición especificada no devuelve la respuesta deseada, sus proxies pueden ser de una versión anterior. En este caso, póngase en contacto con el servicio de asistencia técnica.
		`icluster nodes`
		`xxx 192.168.00.00:1111@xxx xxx connected 10923-16383`
		`xxx 192.168.00.01:2222@xxx xxx connected 0-5460`
		`xxx 192.168.00.02:3333@xxx xxx connected 5461-10922`
		`SCAN 0 match * COUNT 5`
		`192.168.00.02:3333`
		`1) "0"`
		`2) 1) "key1"`
		`2) "key2"`
		`3) "key3"`
		`4) "key4"`
		`5) "key5"`

Lists	BLPOP	Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura.
	BRPOP	
	BRPOPLPUSH	
Pub/Sub	PSUBSCRIBE	Las instancias de Clúster Proxy no admiten la suscripción a eventos de espacio de claves, por lo que no habría un fallo en la suscripción a eventos de espacio de claves.

Categoría	Comando	Restricción
Scripting	EVAL	● Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura. ● Cuando la función multi-DB está habilitada para una instancia Clúster Proxy, se modificará el parámetro **KEYS**. Preste atención al parámetro **KEYS** usado en el script Lua.
	EVALSHA	

<table>
<tr><th>Categoría</th><th>Comando</th><th>Restricción</th></tr>
<tr><td rowspan="8">Server</td><td>MEMORY DOCTOR</td><td rowspan="8">Para una instancia de Clúster Proxy, agregue el ip:port del nodo al final del comando.
Haga lo siguiente para obtener la dirección IP y el número de puerto de un nodo (se toma MEMORY USAGE como ejemplo):
1. Ejecute el comando cluster keyslot key para consultar el número de ranura de una clave.
2. Ejecute el comando icluster nodes para consultar la dirección IP y el número de puerto correspondientes a la ranura donde se encuentra la clave. Si no se devuelve la información requerida después de ejecutar el comando icluster nodes, su instancia de Clúster Proxy puede ser de una versión anterior. En este caso, ejecute el comando cluster nodes.
3. Ejecute el comando MEMORY USAGE key ip:port.
Si multi-DB está habilitada para la instancia de Clúster Proxy, ejecute el comando MEMORY USAGE xxx:As {key} ip:port, donde xxx indica la BD donde está el valor de la clave. Por ejemplo, DB0, DB1 y DB255 corresponden a 000, 001 y 255, respectivamente.
A continuación se muestra un ejemplo de una instancia de Clúster Proxy de base de datos única:
<pre>set key1 value1
OK
get key1
value1
cluster keyslot key1
9189
icluster nodes
xxx 192.168.00.00:1111@xxx xxx
connected 10923-16383
xxx 192.168.00.01:2222@xxx xxx
connected 0-5460
xxx 192.168.00.02:3333@xxx xxx
connected 5461-10922
MEMORY USAGE key1
192.168.00.02:3333
54</pre></td></tr>
<tr><td>MEMORY HELP</td></tr>
<tr><td>MEMORY MALLOC-STATS</td></tr>
<tr><td>MEMORY PURGE</td></tr>
<tr><td>MEMORY STATS</td></tr>
<tr><td>MEMORY USAGE</td></tr>
<tr><td>MONITOR</td></tr>
<tr><td rowspan="2">Strings</td><td>BITOP</td><td rowspan="2">Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura.</td></tr>
<tr><td>MSETNX</td></tr>
</table>

Categoría	Comando	Restricción
Transactions	WATCH	Para una instancia de Clúster Proxy, todas las claves transferidas deben estar en la misma ranura.
	MULTI	El orden de los comandos entre ranuras en una transacción no está garantizado. Los siguientes comandos no se pueden utilizar en transacciones:**WATCH**, **MONITOR**, **RANDOMKEY**, **KEYS**, **SCAN**, **SUBSCRIBE**, **UNSUBSCRIBE**, **PSUBSCRIBE**, **PUNSUBSCRIBE**, **SCRIPT**, **EVAL**, **EVALSHA**, **DBSIZE**, **AUTH**, **FLUSHDB**, **FLUSHALL**, **CLIENT**, **MEMORY**
	EXEC	
Streams	XACK	Actualmente, las instancias Clúster Proxy no soportan Streams.
	XADD	
	XCLAIM	
	XDEL	
	XGROUP	
	XINFO	
	XLEN	
	XPENDING	
	XRANGE	
	XTRIM	
	XREVRANGE	
	XREAD	
	XREADGROUP GROUP	
	XAUTOCLAIM	

Comandos de múltiples claves de instancias de Clúster Proxy

Tabla 7-42 Comandos de múltiples claves de instancias de Clúster Proxy

Categoría	Comando
Comandos de varias teclas que admiten el acceso entre ranuras	DEL, MGET, MSET, EXISTS, SUNION, SINTER, SDIFF, SUNIONSTORE, SINTERSTORE, SDIFFSTORE, ZUNIONSTORE, ZINTERSTORE

Categoría	Comando
Comandos de varias claves que no admiten el acceso entre ranuras	SMOVE, SORT, BITOP, MSETNX, RENAME, RENAMENX, BLPOP, BRPOP, RPOPLPUSH, BRPOPLPUSH, PFMERGE, PFCOUNT, BLMOVE, COPY, GEOSEARCHSTORE, LMOVE, ZRANGESTORE

Comandos de Redis restringidos para instancias de división de lectura/escritura

Tabla 7-43 Comandos de Redis restringidos para instancias de división de lectura/escritura

Categoría	Comando	Restricción
Connection	CLIENT KILL	● Solo se admiten los dos formatos siguientes: – CLIENT KILL ip:port – CLIENT KILL ADDR ip:port ● El campo **id** tiene un valor aleatorio y no cumple con el requisito idc1<idc2→Tc1<Tc2.
	CLIENT LIST	● Solo se admiten los dos formatos siguientes: – CLIENT LIST – CLIENT LIST [TYPE normal|master|replica|pubsub] ● El campo **id** tiene un valor aleatorio y no cumple con el requisito idc1<idc2→Tc1<Tc2.

7.6 Otras restricciones del uso de comandos

En esta sección se describen las restricciones de algunos comandos de Redis.

Comando de KEYS

En el caso de una gran cantidad de datos almacenados en caché, la ejecución del comando **KEYS** puede bloquear la ejecución de otros comandos durante mucho tiempo u ocupar una memoria excepcionalmente grande. Por lo tanto, al ejecutar el comando **KEYS**, describa el patrón exacto y no utilice **keys *** difusos. No utilice el comando **KEYS** en el entorno de producción. De lo contrario, el servicio que se ejecuta se verá afectado.

Comandos en el grupo de Server

- Mientras se ejecutan comandos que tardan mucho en ejecutarse, como **FLUSHALL**, las instancias DCS pueden no responder a otros comandos y pueden cambiar al estado defectuoso. Después de que el comando termine de ejecutarse, la instancia volverá a la normalidad.

- Cuando se ejecuta el comando **FLUSHDB** o **FLUSHALL**, la ejecución de otros comandos de servicio puede bloquearse durante mucho tiempo en el caso de una gran cantidad de datos almacenados en caché.

Comandos EVAL y EVALSHA

- Cuando se ejecutan los comandos **EVAL** o **EVALSHA**, el parámetro del comando debe contener al menos una clave. De lo contrario, el mensaje de error, se muestra "ERR eval/evalsha numkeys must be bigger than zero in redis cluster mode" ("ERR eval/evalsha numkeys debe ser mayor que cero en el modo redis cluster").
- Cuando se ejecuta el comando **EVAL** o **EVALSHA**, una instancia de clúster de DCS para Redis utiliza la primera clave para calcular las ranuras. Asegúrese de que las claves que se van a utilizar en su código estén en la misma ranura. Para obtener más información, visite el **sitio web oficial de Redis**.
- Para el comando **EVAL**:
  - Aprenda las funciones de script de Lua de Redis antes de ejecutar el comando **EVAL**. Para obtener más información, visite el **sitio web oficial de Redis**.
  - El tiempo de ejecución de un script de Lua es de 5 segundos. Deben evitarse las declaraciones que consumen mucho tiempo, como las declaraciones de sueño durante mucho tiempo y las declaraciones de bucle grande.
  - Al invocar a un script Lua, no utilice funciones aleatorias para especificar claves. De lo contrario, los resultados de la ejecución son inconsistentes en los nodos principal y en espera.

Depuración de scripts de Lua

Cuando se depuran scripts Lua para las instancias de Clúster Proxy y de separación de lectura/escritura, solo se admite el modo de no bloqueo asíncrono **--ldb**. No se soporta el modo de bloqueo sincrónico **--ldb-sync-mode**. Por defecto, la simultaneidad máxima en cada proxy es de **2**. Esta restricción no se aplica a otros tipos de instancias.

Otras restricciones

- El límite de tiempo para ejecutar un comando Redis es de 15 segundos. Para evitar que otros servicios fallen, se activará un switchover principal/replicado una vez que se agote el tiempo de ejecución del comando.
- Las instancias de clúster de DCS para Redis creadas antes del 10 de julio de 2018 deben actualizarse para que admitan los siguientes comandos:

  SINTER, SDIFF, SUNION, PFCOUNT, PFMERGE, SINTERSTORE, SUNIONSTORE, SDIFFSTORE, SMOVE, ZUNIONSTORE, ZINTERSTORE, EVAL, EVALSHA, BITOP, RENAME, RENAMENX, RPOPLPUSH, MSETNX, SCRIPT LOAD, SCRIPT KILL, SCRIPT EXISTS y SCRIPT FLUSH

7.7 Comandos admitidos y deshabilitados por DCS for Redis 3.0 (descontinuados)

DCS for Redis 3.0 está desarrollado basado en Redis 3.0.7 y es compatible con los protocolos y comandos de código abierto. Esta sección describe la compatibilidad de DCS for Redis 3.0 con los comandos de Redis, incluidos los comandos compatibles, los comandos

deshabilitados, las secuencias de comandos y los comandos no compatibles de versiones posteriores de Redis y las restricciones en el uso de los comandos.

NOTA

DCS for Redis 3.0 ya no se proporciona. Puede utilizar DCS for Redis 4.0, 5.0 o 6.0 en su lugar.

Las instancias de DCS para Redis admiten la mayoría de los comandos de Redis. Cualquier cliente compatible con el protocolo de Redis puede acceder a DCS.

- Por motivos de seguridad, algunos comandos de Redis están deshabilitados en DCS, como se indica en **Comandos deshabilitados por DCS for Redis 3.0**.
- Algunos comandos de Redis son compatibles con instancias DCS de clúster para operaciones de varias claves en la misma ranura. Para más detalles, véase **Restricciones de comandos**.
- Algunos comandos de Redis (como **KEYS**, **FLUSHDB** y **FLUSHALL**) tienen restricciones de uso, que se describen en **Otras restricciones del uso de comandos**.

Comandos compatibles con DCS for Redis 3.0

A continuación se enumeran los comandos compatibles con DCS for Redis 3.0. Para obtener más información sobre la sintaxis del comando, visite el **sitio web oficial de Redis**. Por ejemplo, para ver detalles sobre el comando SCAN, escriba SCAN en el cuadro de búsqueda de **esta página**.

NOTA

- Los comandos disponibles desde las versiones posteriores de Redis no son compatibles con las instancias de versiones anteriores. Ejecute un comando en redis-cli para comprobar si es compatible con DCS for Redis. Si se devuelve el mensaje "(error) ERR unknown command", el comando no es compatible.
- Las instancias de Clúster Proxy no admiten los siguientes comandos enumerados en las tablas:
  - Grupo de **List**: **BLPOP**, **BRPOP**, y **BRPOPLRUSH**
  - Los comandos de **CLIENT** en el grupo de **Server**: **CLIENT KILL**, **CLIENT GETNAME**, **CLIENT LIST**, **CLIENT SETNAME**, **CLIENT PAUSE**, y **CLIENT REPLY**.
  - Grupo de **Server**: **MONITOR**
  - Grupo de **Transactions**: **UNWATCH** y **WATCH**
  - Grupo de **Key**: **RANDOMKEY** (para las instancias antiguas)

Tabla 7-44 Comandos soportados por instancias de DCS para Redis 3.0 (1)

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
DEL	APPEND	HDEL	BLPOP	SADD	ZADD	FLUSHALL
DUMP	BITCOUNT	HEXISTS	BRPOP	SCARD	ZCARD	FLUSHDB
EXISTS	BITOP	HGET	BRPOPLRUSH	SDIFF	ZCOUNT	DBSIZE

Generic (Key)	String	Hash	List	Set	Sorted Set	Server
EXPIRE	BITPOS	HGETALL	LINDEX	SDIFFSTORE	ZINCRBY	TIME
MOVE	DECR	HINCRBY	LINSERT	SINTER	ZRANGE	INFO
PERSIST	DECRBY	HINCRBYFLOAT	LLEN	SINTERSTORE	ZRANGEBYSCORE	CLIENT KILL
PTTL	GET	HKEYS	LPOP	SISMEMBER	ZRANK	CLIENT LIST
RANDOMKEY	GETRANGE	HMGET	LPUSHX	SMEMBERS	ZREMRANGEBYRANK	CLIENT GETNAME
RENAME	GETSET	HMSET	LRANGE	SMOVE	ZREMRANGEBYCORE	CLIENT SETNAME
RENAMENX	INCR	HSET	LREM	SPOP	ZREVRANGE	CONFIG GET
RESTORE	INCRBY	HSETNX	LSET	SRANDMEMBER	ZREVRANGEBYSCORE	MONITOR
SORT	INCRBYFLOAT	HVALS	LTRIM	SREM	ZREVRANK	SLOWLOG
TTL	MGET	HSCAN	RPOP	SUNION	ZSCORE	ROLE
TYPE	MSET	-	RPOPLPU	SUNIONSTORE	ZUNIONSTORE	-
SCAN	MSETNX	-	RPOPLPUSH	SSCAN	ZINTERSTORE	-
OBJECT	PSETEX	-	RPUSH	-	ZSCAN	-
KEYS	SET	-	RPUSHX	-	ZRANGEBYLEX	-
-	SETBIT	-	-	-	-	-
-	SETEX	-	-	-	-	-
-	SETNX	-	-	-	-	-
-	SETRANGE	-	-	-	-	-
-	STRLEN	-	-	-	-	-

Tabla 7-45 Comandos soportados por instancias de DCS para Redis 3.0 (2)

HyperLogLog	Pub/Sub	Transactions	Connection	Scripting	Geo
PFADD	PSUBSCRIBE	DISCARD	AUTH	EVAL	GEOADD
PFCOUNT	PUBLISH	EXEC	ECHO	EVALSHA	GEOHASH
PFMERGE	PUBSUB	MULTI	PING	SCRIPT EXISTS	GEOPOS
-	PUNSUBSCRIBE	UNWATCH	QUIT	SCRIPT FLUSH	GEODIST
-	SUBSCRIBE	WATCH	SELECT	SCRIPT KILL	GEORADIUS
-	UNSUBSCRIBE	-	-	SCRIPT LOAD	GEORADIUSBYMEMBER

Comandos deshabilitados por DCS for Redis 3.0

A continuación se enumeran los comandos deshabilitados por DCS for Redis 3.0.

Tabla 7-46 Comandos de Redis desactivados en las instancias de DCS compatibles con Redis 3.0 de nodo único y principal/en espera

Generic (Key)	Server
MIGRATE	SLAVEOF
-	SHUTDOWN
-	LASTSAVE
-	DEBUG commands
-	COMMAND
-	SAVE
-	BGSAVE
-	BGREWRITEAOF

Tabla 7-47 Comandos de Redis deshabilitados en instancias de Clúster Proxy de DCS compatibles con Redis 3.0

Generic (Key)	Server	List	Transactions	Connection	Cluster	codis
MIGRATE	SLAVEOF	BLPOP	DISCARD	SELECT	CLUSTER	TIME
MOVE	SHUTDOWN	BRPOP	EXEC	-	-	SLOTSINFO
-	LASTSAVE	BRPOPLPUSH	MULTI	-	-	SLOTSDEL
-	DEBUG commands	-	UNWATCH	-	-	SLOTSMGRTSLOT
-	COMMAND	-	WATCH	-	-	SLOTSMGRTONE
-	SAVE	-	-	-	-	SLOTSCHECK
-	BGSAVE	-	-	-	-	SLOTSMGRTTAGSLOT
-	BGREWRITEAOF	-	-	-	-	SLOTSMGRTTAGONE
-	SYNC	-	-	-	-	-
-	PSYNC	-	-	-	-	-
-	MONITOR	-	-	-	-	-
-	CLIENT commands	-	-	-	-	-
-	OBJECT	-	-	-	-	-
-	ROLE	-	-	-	-	-

7.8 Comandos admitidos y desactivados por DCS for Memcached (descontinuados)

NOTA

DCS for Memcached ya no se proporciona. Puede usar instancias de DCS para Redis en su lugar.

Memcached soporta el protocolo de texto basado en TCP y el protocolo binario. Cualquier cliente compatible con un protocolo de Memcached puede acceder a las instancias de DCS.

Protocolo de texto de Memcached

El protocolo de texto Memcached utiliza texto ASCII para transferir comandos, lo que le ayuda a compilar clientes y problemas de depuración. Las instancias de DCS para Memcached pueden incluso conectarse directamente mediante Telnet.

En comparación con el protocolo binario de Memcached, el protocolo de texto de Memcached es compatible con más clientes de código abierto, pero el protocolo de texto no admite autenticación.

NOTA

Los clientes pueden usar el protocolo de texto Memcached para acceder a las instancias de Memcached de DCS solo si está habilitado el acceso sin contraseña. El acceso sin contraseña significa que el acceso a las instancias de Memcached de DCS no estará protegido por nombre de usuario y contraseña, y cualquier cliente de Memcached que satisfaga las reglas del grupo de seguridad en la misma VPC puede acceder a las instancias. Habilitar el acceso sin contraseña plantea los riesgos de seguridad. Tenga cuidado al habilitar el acceso sin contraseña.

Tabla 7-48 enumera los comandos admitidos por el protocolo de texto Memcached y describe si estos comandos son compatibles con las instancias de DCS Memcached.

Tabla 7-48 Comandos compatibles con el protocolo de texto de Memcached

Comando	Descripción	Soportado por DCS
add	Agrega datos.	Sí
set	Establece datos, incluida la adición o modificación de datos.	Sí
replace	Reemplaza los datos.	Sí
append	Agrega datos después del valor de la clave especificada.	Sí
prepend	Agrega datos antes del valor de una clave especificada.	Sí
cas	Comprueba y establece los datos.	Sí
get	Consulta datos.	Sí
gets	Consulta detalles de datos.	Sí
delete	Elimina los datos.	Sí
incr	Agrega la cantidad especificada al contador solicitado.	Sí
decr	Quita la cantidad especificada en el contador solicitado.	Sí
touch	Actualiza el tiempo de caducidad de los datos existentes.	Sí
quit	Cierra la conexión.	Sí

Comando	Descripción	Soportado por DCS
flush_all	Invalida todos los datos existentes. **NOTA** El valor de la opción delay (si existe) debe ser **0**.	Sí
version	Consulta información de la versión de Memcached.	Sí
stats	Gestiona las estadísticas de operación. **NOTA** Actualmente, solo se pueden consultar estadísticas básicas. No se pueden consultar los comandos de los parámetros opcionales.	Sí
cache_memlimit	Ajusta el límite de memoria caché.	No
slabs	Consulta el uso de estructuras de almacenamiento internas.	No
lru	Gestiona las políticas de eliminación de datos caducados.	No
lru_crawler	Gestiona los hilos de eliminación de datos caducados.	No
verbosity	Establece el nivel de detalle de la salida del registro.	No
watch	Inspecciona lo que está pasando internamente.	No

Protocolo binario de Memcached

El protocolo binario Memcached codifica comandos y operaciones en estructuras específicas antes de enviarlos. Los comandos se representan mediante cadenas de caracteres predefinidas.

El protocolo binario de Memcached proporciona más características pero menos clientes que el protocolo de texto de Memcached. El protocolo binario de Memcached es más seguro que el protocolo de texto de Memcached, ya que además admite la autenticación simple y autenticación de capa de seguridad (SASL).

Tabla 7-49 enumera los comandos admitidos por el protocolo de binario de Memcached y describe si estos comandos son compatibles con las instancias de DCS Memcached.

Tabla 7-49 Comandos soportados por el protocolo binario de Memcached

Código de Comando	Comando	Descripción	Soportado por DCS
0x00	GET	Consulta datos.	Sí
0x01	SET	Establece datos, incluida la adición o modificación de datos.	Sí
0x02	ADD	Agrega datos.	Sí
0x03	REPLACE	Reemplaza los datos.	Sí
0x04	DELETE	Elimina los datos.	Sí
0x05	INCREMENT	Agrega la cantidad especificada al contador solicitado.	Sí
0x06	DECREMENT	Quita la cantidad especificada en el contador solicitado.	Sí
0x07	QUIT	Cierra la conexión.	Sí
0x08	FLUSH	Invalida todos los datos existentes. **NOTA** El valor de la opción delay (si existe) debe ser **0**.	Sí
0x09	GETQ	Queries data. El cliente no recibirá ninguna respuesta en caso de falla.	Sí
0x0a	NOOP	Instrucción de no operación, equivalente a ping.	Sí
0x0b	VERSION	Consulta información de la versión de Memcached.	Sí
0x0c	GETK	Consulta datos y agrega una clave al paquete de respuesta.	Sí
0x0d	GETKQ	Consulta datos y devuelve una clave. El cliente no recibirá ninguna respuesta en caso de falla.	Sí
0x0e	APPEND	Agrega datos después del valor de la clave especificada.	Sí
0x0f	PREPEND	Agrega datos antes del valor de una clave especificada.	Sí

Código de Comando	Comando	Descripción	Soportado por DCS
0x10	STAT	Consulta estadísticas de instancias de DCS Memcached. **NOTA** Actualmente, solo se pueden consultar estadísticas básicas. No se pueden consultar los comandos de los parámetros opcionales.	Sí
0x11	SETQ	Establece datos, incluida la adición o modificación de datos. El comando **SETQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí
0x12	ADDQ	Agrega datos. A diferencia del comando **ADD**, el comando **ADDQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí
0x13	REPLACEQ	Reemplaza los datos. A diferencia del comando **REPLACE**, el comando **REPLACEQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí
0x14	DELETEQ	Elimina los datos. A diferencia del comando **DELETE**, el comando **DELETEQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí
0x15	INCREMENTQ	Agrega la cantidad especificada al contador solicitado. A diferencia del comando **INCREMENT** , el comando **INCREMENTQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí

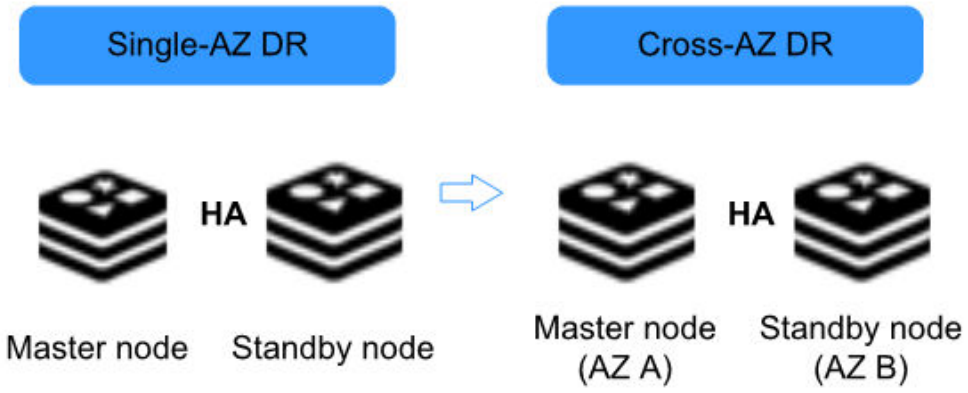
Código de Comando	Comando	Descripción	Soportado por DCS
0x16	DECREMENTQ	Quita la cantidad especificada en el contador solicitado. A diferencia del comando **DECREMENT**, el comando **DECREMENTQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí
0x17	QUITQ	Cierra la conexión.	Sí
0x18	FLUSHQ	Borra datos y no devuelve información. **NOTA** El valor de la opción delay (si existe) debe ser **0**.	Sí
0x19	APPENDQ	Agrega datos después del valor de la clave especificada. A diferencia del comando **APPEND** , el comando **APPENDQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí
0x1a	PREPENDQ	Agrega datos antes del valor de una clave especificada. A diferencia del comando **PREPEND**, el comando **PREPENDQ** solo devuelve una respuesta en caso de errores. El cliente no recibirá ninguna respuesta en caso de éxito.	Sí
0x1c	TOUCH	Actualiza el tiempo de caducidad de los datos existentes.	Sí
0x1d	GAT	Consulta los datos y actualiza el tiempo de caducidad de los datos existentes.	Sí
0x1e	GATQ	Consulta datos y devuelve una clave. El cliente no recibirá ninguna respuesta en caso de falla.	Sí

Código de Comando	Comando	Descripción	Soportado por DCS
0x23	GATK	Consulta datos, agrega una clave al paquete de respuesta y actualiza el tiempo de caducidad de los datos existentes.	Sí
0x24	GATKQ	Consulta datos, devuelve una clave y actualiza el tiempo de caducidad de los datos existentes. El cliente no recibirá ninguna respuesta en caso de falla.	Sí
0x20	SASL_LIST_MECHS	Pregunta al servidor qué mecanismos de autenticación SASL admite.	Sí
0x21	SASL_AUTH	Inicia la autenticación SASL.	Sí
0x22	SASL_STEP	Se requieren pasos de autenticación adicionales.	Sí

8 Recuperación ante desastres (DR) y solución multiactiva

Ya sea que utilice DCS como caché frontend o almacenamiento de datos backend, DCS siempre está listo para garantizar la confiabilidad de los datos y la disponibilidad del servicio. La siguiente figura muestra la evolución de las arquitecturas DR de DCS.

Figura 8-1 Evolución de la arquitectura DR de DCS



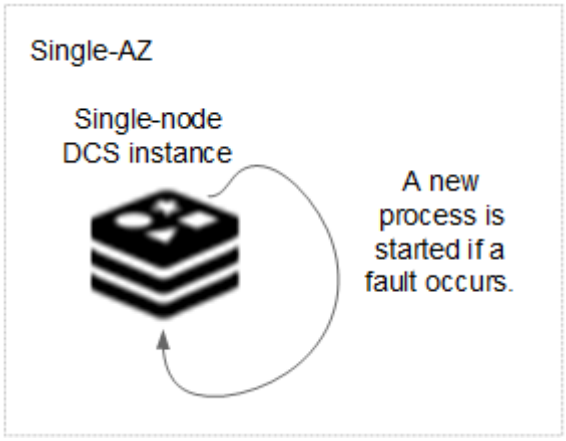
Para cumplir con los requisitos de confiabilidad de sus datos y servicios, puede optar por desplegar su instancia de DCS dentro de una única AZ o entre AZ.

HA de la AZ única dentro de una región

Despliegue de la AZ única significa desplegar una instancia dentro de una sala de equipos físicos. DCS proporciona HA de proceso/servicio, persistencia de datos y políticas de DR en standby activa para diferentes tipos de instancias de DCS.

Instancia de nodo único de DCS: cuando DCS detecta una falla de proceso, se inicia un nuevo proceso para garantizar el HA del servicio.

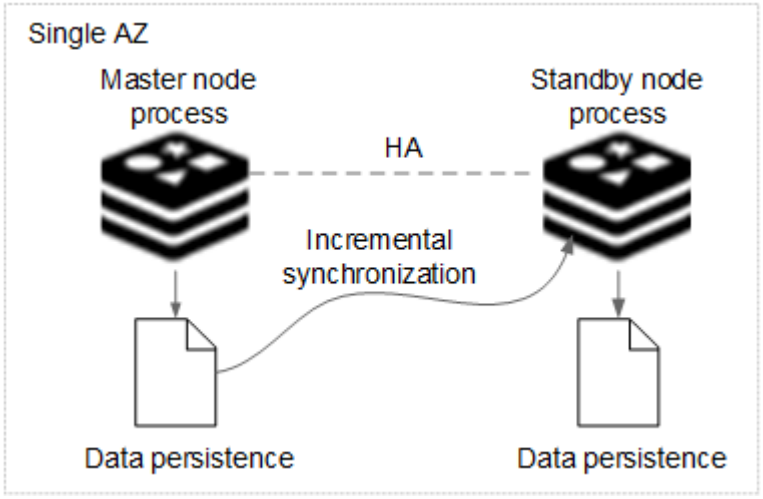
Figura 8-2 HA para una instancia de DCS de nodo único implementada dentro de una AZ



Principal/en espera, separación de lectura/escritura o instancia de DCS de clúster: De forma predeterminada, los datos se mantienen en el disco en el nodo principal y se sincronizan y persisten en el nodo en espera de manera incremental, lo que permite lograr la persistencia de datos y la espera activa.

La siguiente figura muestra la sincronización de datos y la persistencia de los procesos de los nodos principales y en espera, incluidos los procesos de las instancias de separación principales/en espera y de lectura/escritura y cada partición de instancias de clúster.

Figura 8-3 HA entre nodos principales y en espera dentro de una AZ



DR entre las AZ dentro de una región

Los nodos principales y en espera de una instancia de DCS (no se incluye el tipo de nodo único) se pueden desplegar en las AZ (en diferentes salas de equipos). Las fuentes de alimentación y las redes de diferentes AZ están aisladas físicamente. Cuando se produce una falla en la AZ donde se despliega el nodo principal, el nodo en espera se conecta al cliente y se hace cargo de las operaciones de lectura y escritura de datos.

Figura 8-4 Despliegue entre las AZ de una instancia principal/en espera de DCS

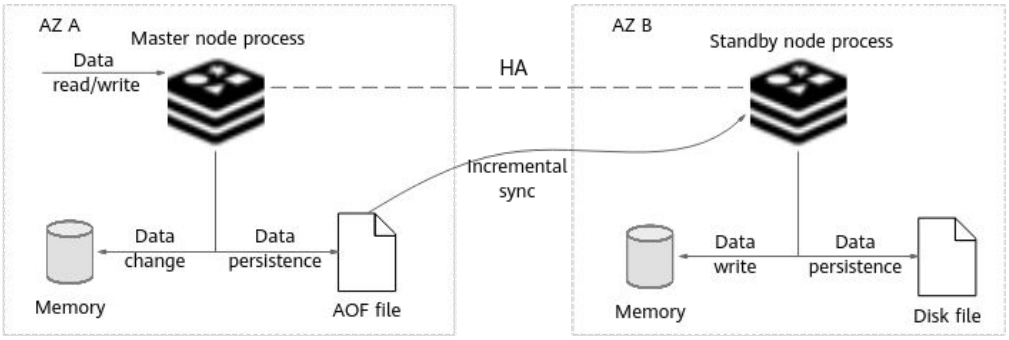


Figura 8-5 Despliegue entre las AZ de una instancia de DCS de separación de lectura/escritura

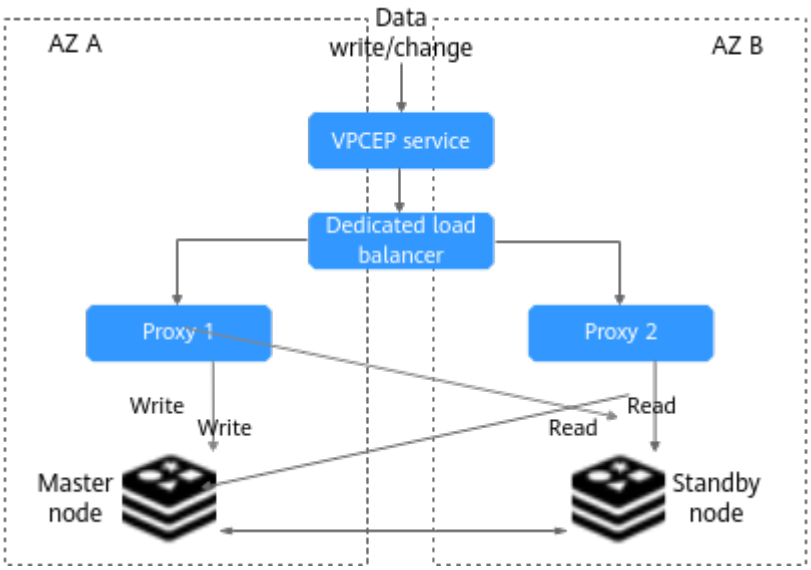


Figura 8-6 Despliegue entre las AZ de una instancia de Clúster Proxy de DCS

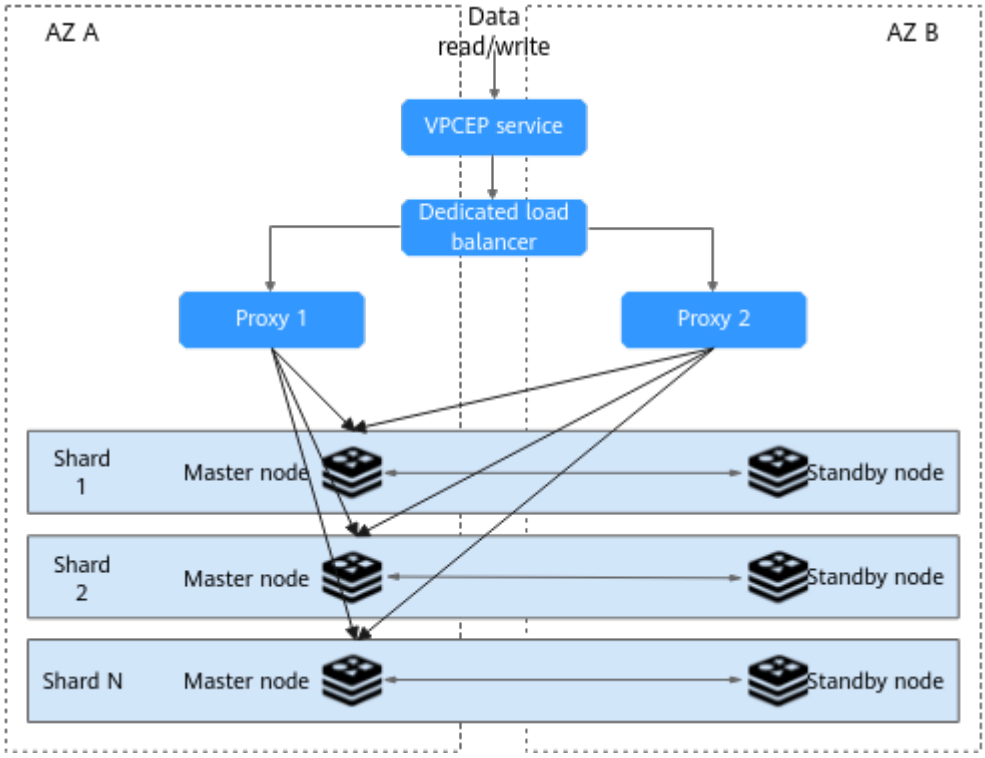
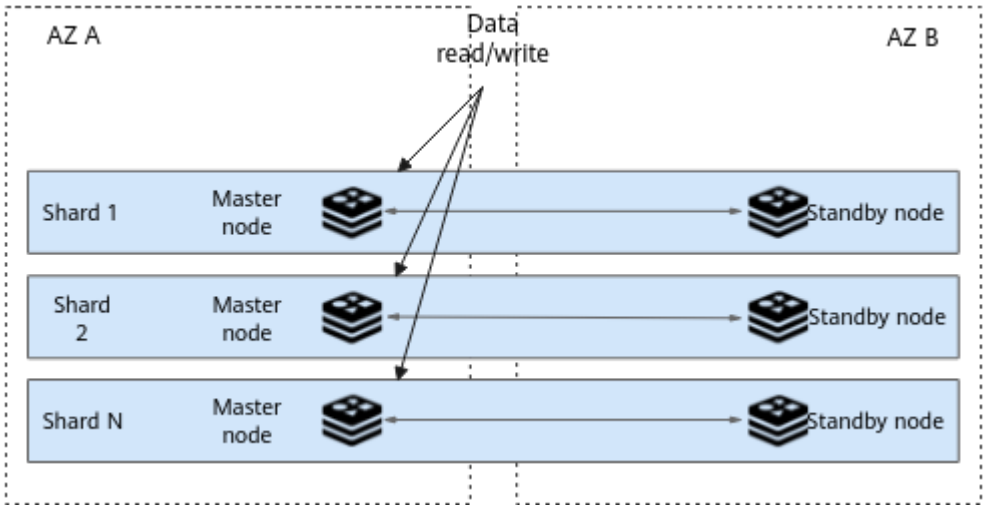
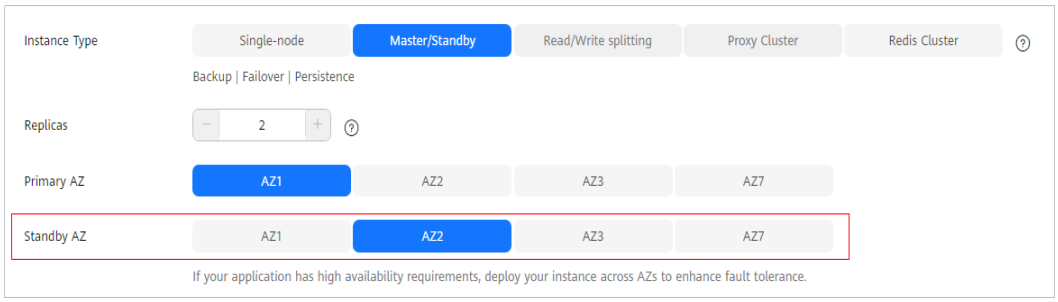


Figura 8-7 Despliegue entre las AZ de una instancia de Clúster Redis de DCS



Al crear una instancia de DCS principal/en espera, de clúster o de separación de lectura/escritura, seleccione una AZ en espera que sea diferente de la AZ principal como se muestra a continuación.

Figura 8-8 Selección de diferentes AZ



NOTA

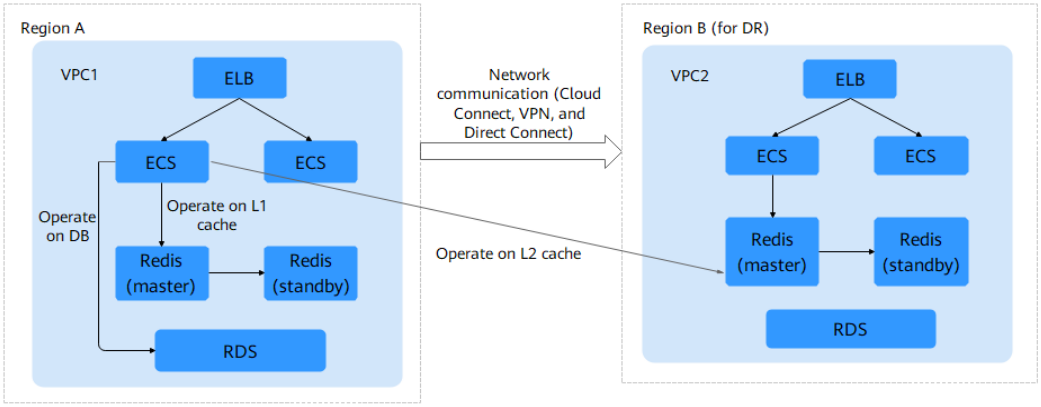
También puede desplegar su aplicación en todas las AZ para garantizar la fiabilidad de los datos y la disponibilidad del servicio en caso de interrupciones en el suministro eléctrico o en la red.

Multiactividades entre las regiones

Actualmente, Huawei Cloud DCS no admite multiactividad entre regiones porque Redis no tiene una solución madura de actividad-actividad. **Actividad-actividad es diferente de recuperación ante desastres o HA principal/en espera.**

No se puede lograr Redis activo-activo entre las nubes o las regiones porque los protocolos de serialización de REdis (RESP) personalizados no están unificados. Si se requiere activo-activo, se puede implementar mediante **escritura dual en el extremo de la aplicación**.

Figura 8-9 Escritura dual en el lado de la aplicación para lograr multi-actividad



Notas:

1. La solución de escritura dual **no puede garantizar la coherencia de la caché** (debido a problemas de red). Las **aplicaciones** deben **tolerar la incoherencia de la caché** (estableciendo el tiempo de vida para lograr una eventual consistencia). Si las aplicaciones requieren una fuerte coherencia de la caché, esta solución no es adecuada. Actualmente, no existe una solución en la industria para garantizar una sólida consistencia de la caché entre regiones.
2. Se recomienda realizar operaciones en la caché L2 entre regiones en modo asincrónico.

9 Diferencias del motor de caché

9.1 Comparación entre las versiones de Redis

Al crear una instancia de DCS compatible con Redis, puede seleccionar la versión del motor de caché y el tipo de instancia.

NOTA

- DCS for Redis 3.0 ya no se proporciona. Puede utilizar DCS for Redis 4.0, 5.0 o 6.0 en su lugar.
- Las arquitecturas subyacentes varían según la versión de Redis. Una vez que se elige una versión de Redis, no se puede cambiar. Por ejemplo, no puede actualizar una instancia de DCS Redis 4.0 a Redis 5.0 o 6.0. Si necesita una versión de Redis superior, cree una nueva instancia que cumpla con sus requisitos y, a continuación, migre los datos de la instancia antigua a la nueva.

- **Versión**

  DCS soporta Redis 3.0/4.0/5.0/6.0. **Tabla 9-1** describe las diferencias entre estas versiones. Para obtener más información, véase **Nuevas funciones de DCS for Redis 4.0**, **Nuevas funciones de DCS for Redis 5.0** y **Nuevas funciones de DCS for Redis 6.0**.

  Tabla 9-1 Diferencias entre las versiones de Redis

Característica	Redis 3.0 (descontinuado)	Redis 4.0 y Redis 5.0	Redis 6.0
Compatibilidad con software de código abierto:	Redis 3.0.7	Redis 4.0.14 y 5.0.9, respectivamente	Edición básica: Redis 6.2.7 Edición profesional: Redis 6

Característica	Redis 3.0 (descontinuado)	Redis 4.0 y Redis 5.0	Redis 6.0
Modo de despliegue de instancia	Basado en las máquinas virtuales	Containerizado basado en los servidores físicos	Edición básica: en contenedor basado en servidores físicos Edición profesional: basada en VM
Tiempo necesario para crear una instancia	3–15 minutos Instancia de clúster: 10–30 minutos	8 segundos	Edición básica: 8 s Edición profesional: de 5 a 15 minutos
QPS	100,000 QPS por nodo Para obtener QPS de cada especificación de instancia, véase **Especificaciones de instancia de Redis 3.0 (descontinuado)**.	100,000 QPS por nodo Para obtener QPS de cada especificación de instancia, véase **Especificaciones de las instancias de Redis 4.0 y 5.0**.	Edición básica: 100,000 QPS por nodo Edición profesional: 400,000 QPS por nodo Para obtener QPS de cada especificación de instancia, véase **Especificaciones de instancia de Redis 6.0**.
Acceso a la red pública	Se admite	No se admite	No se admite
Conexión de nombre de dominio	Compatible con VPC	Compatible con VPC	Compatible con VPC
Gestión de datos visualizados	No se admite	Proporciona CLI web para el acceso a Redis y la gestión de datos.	Proporciona CLI web para el acceso a Redis y la gestión de datos.

Característica	Redis 3.0 (descontinuado)	Redis 4.0 y Redis 5.0	Redis 6.0
Tipo de instancia	Nodo único, principal/en espera y Clúster Proxy	Nodo único, principal/en espera, separación de lectura/escritura, Clúster Proxy y Clúster Redis	Edición básica: nodo único, principal/en espera, separación de lectura/escritura, Clúster Proxy y Clúster Redis Edición profesional: principal/en espera
Ampliación/reducción de la capacidad	Ampliación y reducción de la capacidad en línea	Ampliación y reducción de la capacidad en línea	Ampliación y reducción de la capacidad en línea
Copia de respaldo y restauración	Compatible con las instancias principal/en standby y de Clúster Proxy	Compatible con las instancias de separación de lectura/escritura, principal/standby, de Clúster Proxy y de Clúster Redis	Compatible con las instancias de separación de lectura/escritura, principal/standby, de Clúster Proxy y de Clúster Redis

- **Tipo de instancia**

  DCS proporciona los tipos de instancia de nodo único, principal/en standby, de Clúster Proxy, de Clúster Redis y de separación de lectura/escritura. Para obtener más información sobre sus arquitecturas y escenarios de aplicación, consulte **Tipos de instancia de DCS**.

9.2 Comparación entre las ediciones profesionales y básicas

Huawei Cloud DCS de la edición profesional está completamente desarrollada por Huawei Cloud y es compatible con el software de Redis de código abierto. Esta edición utiliza el modelo de master-N*worker de subprocesos múltiples en lugar del modelo master-worker convencional. Cada subproceso de trabajo puede escuchar en los puertos para supervisar las solicitudes de establecimiento de conexiones de red, aceptar las solicitudes y establecer conexiones de red, leer y escribir datos (en conexiones de red, por ejemplo, sockets) y analizar y procesar comandos de Redis. Como resultado, el rendimiento mejora N veces.

Tabla 9-2 Comparación entre las ediciones profesionales y básicas

Concepto	Edición básica	Ediciones profesionales
Compatibilidad con software de código abierto:	Redis 4.0/5.0/6.0 de código abierto, subproceso único	Compatible con Redis 6.0 de código abierto, subprocesos múltiples
Rendimiento	100,000 QPS por partición, hasta 1 ms de latencia	Profesional (rendimiento): 400,000 QPS por partición Profesional (almacenamiento): 70,000 QPS por partición Hasta 1 ms de latencia
Especificaciones de instancias	Tipos de instancias: nodo único, principal/en espera, clúster y separación de lectura/escritura ● Nodo único y principal/en espera: de 128 MB a 64 GB ● División de lectura/escritura: 8–32 GB ● Clúster: hasta 2048 GB	Actualmente, solo está disponible el tipo de instancia principal/en espera. Hay dos ediciones: ● Profesional (rendimiento): de 8 a 64 GB ● Profesional (almacenamiento): memoria de 8 a 32 GB y hasta 256 GB con almacenamiento en SSD Para obtener detalles sobre las diferencias entre las ediciones profesionales (rendimiento) y profesionales (almacenamiento), véase **Tabla 9-3**.
Seguridad de datos	● Autorización detallada y listas blancas de direcciones IP ● Persistencia y copia de respaldo de datos (excepto para el tipo de nodo único) ● Recuperación ante desastres entre AZ ● Conmutación por error automática ● Expansión de capacidad y cambio de tipo de instancia con unos pocos clics	● Autorización detallada y grupos de seguridad ● Persistencia y copia de respaldo ● Recuperación ante desastres entre AZ ● Conmutación por error automática ● Expansión de capacidad con unos pocos clics

Tabla 9-3 Diferencias entre las ediciones profesionales (rendimiento) y profesionales (almacenamiento)

Concepto	Edición profesional (Rendimiento)	Edición profesional (almacenamiento)
QPS	400,000 QPS por partición, mayor que la edición de almacenamiento	70,000 QPS por partición
Almacenamiento	No se admite el almacenamiento de SSD. Los datos se almacenan completamente en la memoria. Los datos se conservan usando archivos AOF.	Memoria + SSD La memoria almacena los datos activos y los SSD almacenan todos los datos.
Formato del archivo de copia de respaldo	AOF y RDB	RDB

10 Comparación entre servicios DCS y servicios de caché de código abierto

DCS admite las instancias de nodo único, principal/en standby y de clúster, lo que garantiza un alto rendimiento de lectura/escritura y un acceso rápido a los datos. También soporta varias operaciones de gestión de instancias para facilitar su O&M. Con DCS, solo necesita centrarse en la lógica del servicio, sin preocuparse por los problemas de despliegue, monitoreo, escalado, seguridad y recuperación de fallas.

DCS es compatible con Redis y Memcached de código abierto, y se puede personalizar según sus requisitos. Esto hace que DCS características únicas además de las ventajas de las bases de datos de caché de código abierto.

DCS for Redis vs. Redis de código abierto

Tabla 10-1 Diferencias entre DCS for Redis y Redis de código abierto

Característica	Redis de código abierto	DCS for Redis
Despliegue del servicio	Requiere de 0.5 a 2 días para preparar servidores.	● Una instancia profesional de DCS para Redis 3.0 o 6.0 se despliega en VM y se puede crear en 5 a 15 minutos. ● Las instancias básicas de DCS para Redis 4.0/5.0/6.0 se almacenan en contenedores y se pueden crear en 8 segundos.
Versión	-	Profundamente comprometido con la comunidad de código abierto y admite la última versión de Redis. Actualmente, se soportan Redis 3.0/4.0/5.0/6.0.
Seguridad	La seguridad de la red y del servidor es responsabilidad del usuario.	● La seguridad de la red se garantiza mediante las VPC y los grupos de seguridad de Huawei Cloud. ● La confiabilidad de los datos está garantizada por la replicación de datos y el backup programado.
Rendimiento	-	100,000 QPS por nodo. DCS for Redis 6.0 puede alcanzar QPS de 400,000 por nodo.

Característica	Redis de código abierto	DCS for Redis
Monitoreo	Proporciona solo estadísticas básicas.	Proporciona más de 30 métricas de monitoreo y umbral de alarma personalizable y políticas. ● Diversas métricas: – Las métricas externas incluyen el número de comandos, las operaciones simultáneas, las conexiones, los clientes y las conexiones denegadas. – Las métricas de uso de recursos incluyen uso de CPU, uso de memoria física, rendimiento de entrada de red y rendimiento de salida de red. – Las métricas internas incluyen el uso de la capacidad de instancia, así como el número de claves, claves caducadas, canales de PubSub y patrones de PubSub y aciertos de espacio de claves. ● Umbrales de alarma personalizados y políticas para diferentes métricas para ayudar a identificar fallas de servicio.
Copia de respaldo y restauración	Se admite	● Soporta copias de seguridad programadas y manuales. Los archivos de copia de seguridad se pueden descargar. ● Los datos de copia de seguridad se pueden restaurar en la consola.
Gestión de parámetros	Sin gestión de parámetros visualizados	● La gestión de parámetros visualizados es compatible con la consola. ● Los parámetros de configuración se pueden modificar en línea. ● Se puede acceder a los datos y modificarlos en la consola.
Escalamiento vertical	Interrumpe los servicios e implica un procedimiento complejo, desde modificar la memoria RAM del servidor hasta modificar la memoria Redis y reiniciar el SO y los servicios.	● Soporta escalamiento en línea y escalamiento en línea sin interrumpir los servicios. ● Las especificaciones se pueden escalar hacia arriba o hacia abajo dentro de la gama disponible en función de los requisitos de servicio.
O&M	O&M manual	Servicios O&M de extremo a extremo las 24 horas de los 7 días

DCS for Memcached vs. Memcached de código abierto

Tabla 10-2 Diferencias entre DCS for Memcached y Memcached de código abierto

Característica	Memcached de código abierto	DCS for Memcached
Despliegue del servicio	Requiere de 0.5 a 2 días para preparar servidores.	Crea una instancia en 5 a 15 minutos.
Seguridad	La seguridad de la red y del servidor es responsabilidad del usuario.	● La seguridad de la red se garantiza mediante las VPC y los grupos de seguridad de Huawei Cloud. ● La confiabilidad de los datos está garantizada por la replicación de datos y el backup programado.
Rendimiento	-	100,000 QPS por nodo
Monitoreo	Proporciona solo estadísticas básicas.	Proporciona más de 30 métricas de monitoreo y umbral de alarma personalizable y políticas. ● Diversas métricas: – Las métricas externas incluyen el número de comandos, las operaciones simultáneas, las conexiones, los clientes y las conexiones denegadas. – Las métricas de uso de recursos incluyen uso de CPU, uso de memoria física, rendimiento de entrada de red y rendimiento de salida de red. – Las métricas internas incluyen el uso de la capacidad de instancia, así como el número de claves, claves caducadas, canales de PubSub y patrones de PubSub y aciertos de espacio de claves. ● Umbrales de alarma personalizados y políticas para diferentes métricas para ayudar a identificar fallas de servicio.
Copia de respaldo y restauración	No admitido	● Soporta copias de seguridad programadas y manuales. ● Los datos de copia de seguridad se pueden restaurar en la consola.
Mantenimiento visualizado	Sin gestión de parámetros visualizados	● La gestión de parámetros visualizados es compatible con la consola. ● Los parámetros de configuración se pueden modificar en línea.

Característica	Memcached de código abierto	DCS for Memcached
Escalamiento vertical	Interrumpe los servicios e implica un procedimiento complejo, desde modificar la memoria RAM del servidor hasta modificar la memoria Redis y reiniciar el SO y los servicios.	● Soporta escalamiento en línea sin interrumpir los servicios. ● Las especificaciones se pueden escalar hacia arriba o hacia abajo dentro de la gama disponible en función de los requisitos de servicio.
O&M	O&M manual	Servicios O&M de extremo a extremo las 24 horas de los 7 días
Persistencia de los datos	No admitido	Compatible con las instancias principal/en espera

11 Notas y restricciones

Instancia de Redis

Tabla 11-1 Notas y restricciones

Concepto	Notas y restricciones
Versión de instancia	Actualmente, DCS admite Redis 3.0 (descontinuado), 4.0, 5.0 y 6.0. Las instancias de Redis no se pueden actualizar, pero los datos de una instancia anterior se pueden migrar a una posterior.
Seguridad de datos	● Para controlar el acceso a las instancias de la edición profesional de DCS para Redis 3.0 y Redis 6.0, puede utilizar grupos de seguridad. Las listas blancas no son compatibles. ● Para controlar el acceso a instancias de edición básica de DCS para Redis 4.0 o Redis 5.0 y 6.0, puede utilizar listas blancas. No se admiten grupos de seguridad. ● La edición básica de DCS para Redis 6.0 admite la encriptación de SSL.
Persistencia de los datos	● No disponible para instancias de nodo único. ● Instancias principal/en espera, separación de lectura/escritura y clúster (excepto clúster de réplica única): la persistencia de datos se admite de forma predeterminada.
Separación de lecturas/escrituras	● Instancias de separación de lectura/escritura: la separación de lectura/escritura se implementa en el servidor por defecto. ● Clúster Redis e instancias principales/en espera: la separación de lectura/escritura se implementa en el cliente, lo que requiere configuraciones manuales. ● La separación de lectura/escritura no se admite en otras instancias.

Concepto	Notas y restricciones
Backup de datos	La consola admite el copia de respaldo de datos automático o manual para instancias que no sean de nodo único.
VPC y subred	Se fija una vez creada la instancia.

Cambio de instancia

Tabla 11-2 Notas y restricciones

Concepto	Notas y restricciones
Cambio de especificaciones o tipos de instancias de Redis	● Se recomienda cambiar las instancias durante las horas no pico. De lo contrario, el cambio puede fallar. ● Cambie la cantidad y la capacidad de la réplica por separado. ● Solo se puede eliminar una réplica por operación. ● Para obtener más información, véase **Modificación de las especificaciones**.
Ajuste del ancho de banda de la instancia de DCS	● Esta función no está disponible para instancias de DCS Redis de edición profesional. ● Esta función solo está disponible para instancias en estado **Running**. ● El rango de ajuste de ancho de banda va desde el ancho de banda asegurado de la instancia hasta su ancho de banda máximo. Por lo general, el ancho de banda máximo por shard es de 2048 Mbit/s cuando la máquina física del nodo de instancia tiene suficientes recursos.
Cambiar las instancias de clúster para que estén en todas las AZ	Disponible solo para instancias de clúster de una sola AZ con dos o más réplicas. ● Para actualizar AZ para instancias de clúster de proxy: – El funcionamiento del Service puede fluctuar durante el cambio. Realice esta operación durante las horas de menor actividad. – Si su aplicación no puede reconectarse ni recuperarse de las excepciones, intente reiniciarla después del cambio. ● Para actualizar las AZ para instancias de clúster de Redis: – El cambio de las AZ no interrumpirá los servicios ni el nodo principal, pero afectará ligeramente el rendimiento. Realice esta operación durante las horas de menor actividad. – El cambio de AZ interrumpe las conexiones a algunas réplicas. Asegúrese de que su aplicación pueda recuperarse automáticamente de excepciones y volver a conectarse a Redis.

Migración de datos

Tabla 11-3 Notas y restricciones

Concepto	Notas y restricciones
Actualización de versión	Para migrar una instancia, la versión de la instancia de destino debe ser posterior a la de origen. La migración de una instancia posterior a una anterior puede fallar.
Migración en línea	● Para migrar instancias de Redis en línea en la consola de DCS, la red entre el origen y el destino debe estar conectada y los comandos **SYNC** y **PSYNC** deben estar permitidos en el origen. ● No puede utilizar redes públicas para la migración en línea. ● El origen debe ser Redis 3.0 o posterior. ● Se recomienda realizar la migración en línea durante las horas no pico. De lo contrario, el uso de la CPU de la instancia de origen puede aumentar y la latencia puede aumentar.
Switch IP	● Las direcciones IP y los nombres de dominio de las instancias de DCS Redis de origen y destino se pueden conmutar en la consola después de que la instancia de origen se haya migrado completa e incrementalmente. ● No disponible para instancias de edición profesional. ● No disponible para instancias de clúster Redis.

12 Facturación

DCS soporta el modo de pago por uso y el modo de facturación anual/mensual. Para obtener más información, véase **Detalles de precios de productos**.

NOTA

Los modos de facturación disponibles en la consola varían según la región. Algunas regiones no soportan el modo de facturación anual/mensual.

Conceptos de facturación

El uso de DCS se factura según la especificación de instancia de DCS.

Concepto de facturación	Descripción
Instancia de DCS	Facturación basada en las especificaciones de instancia de DCS.

Nota: Los cargos de DCS de Huawei Cloud se basan en las especificaciones de instancia de DCS seleccionadas en lugar de la capacidad de caché real.

Modos de facturación

DCS ofrece dos modos de facturación: pago por uso y anual/mensual. Se recomienda pagar por uso si no está seguro de sus necesidades futuras de servicio y desea evitar pagar por los recursos no utilizados. Sin embargo, si está seguro de sus necesidades, anual/mensual será menos costoso.

- Anual/Mensual: Ofrece un descuento mayor que el modo de pago por uso y se recomienda para usuarios a largo plazo.
- Pago por uso (por hora): puede iniciar y detener las instancias de DCS según sea necesario y se le facturará en función de la duración de su uso de las instancias DCS. La facturación se inicia cuando se crea una instancia de DCS y finaliza cuando se elimina la instancia de DCS. La unidad de tiempo mínima es de un segundo.
- Puede cambiar entre los modos anual/mensual y de pago por uso.

Modificaciones en la configuración

Puede cambiar las especificaciones de una instancia de DCS para Redis o Memcached, es decir, escalar o reducir una instancia y cambiar el tipo de instancia. Después de cambiar correctamente las especificaciones, la instancia se factura en función de las nuevas especificaciones. Para obtener más detalles, véase **Modificación de especificaciones de instancia de DCS**.

Renovación

Cuando se caduca un paquete de recursos, puede renovarlo o establecer reglas de renovación automática para ello. Para obtener más información acerca de cómo renovar paquetes de recursos, consulte **Gestión de renovación**.

Preguntas frecuentes

Para obtener más información sobre la facturación de DCS, véase **Preguntas frecuentes de compras y facturación**.

13 Gestión de permisos

Si necesita asignar diferentes permisos a los empleados de su empresa para acceder a sus recursos de DCS, IAM es una buena opción para la gestión de permisos detallada. IAM proporciona autenticación de identidad, gestión de permisos y control de acceso, lo que le ayuda a proteger el acceso a sus recursos de Huawei Cloud.

Con IAM, puede usar su cuenta de Huawei Cloud para crear usuarios de IAM para sus empleados y asignar permisos a los usuarios para controlar su acceso a tipos de recursos específicos. Por ejemplo, algunos desarrolladores de software de su empresa necesitan usar recursos de DCS, pero no deben poder eliminar instancias de DCS ni realizar otras operaciones de alto riesgo. En este escenario, puede crear usuarios de IAM para los desarrolladores de software y concederles solo los permisos necesarios para usar los recursos de DCS.

Si su cuenta de Huawei Cloud no requiere usuarios individuales de IAM para la administración de permisos, omita esta sección.

IAM se puede utilizar de forma gratuita. Solo paga por los recursos de su cuenta. Para obtener más información acerca de IAM, consulte **Descripción de servicio de IAM**.

Permisos de DCS

De forma predeterminada, los nuevos usuarios de IAM no tienen permisos asignados. Debe agregar un usuario a uno o más grupos y adjuntar políticas o funciones de permisos a estos grupos. Los usuarios heredan permisos de los grupos a los que se agregan y pueden realizar operaciones específicas a servicios en la nube según los permisos.

DCS es un servicio a nivel de proyecto implementado y accedido en regiones físicas específicas. Para asignar permisos de DCS a un grupo de usuarios, especifique el ámbito como proyectos específicos de la región y seleccione proyectos para que los permisos surtan efecto. Si se selecciona **All projects**, los permisos surtirán efecto para el grupo de usuarios en todos los proyectos específicos de la región. Al acceder a DCS, los usuarios deben cambiar a una región en la que se les haya autorizado a usar este servicio.

Puede conceder permisos a los usuarios mediante roles y políticas.

- Roles: Tipo de mecanismo de autorización de grano grueso que define permisos relacionados con las responsabilidades del usuario. Este mecanismo proporciona solo un número limitado de roles de nivel de servicio para la autorización. Al usar roles para conceder permisos, también debe asignar otros roles de los que dependen los permisos para que surtan efecto. Sin embargo, los roles no son una opción ideal para la autorización detallada y el control de acceso seguro.

- Políticas: Un tipo de mecanismo de autorización detallado que define los permisos necesarios para realizar operaciones en recursos de nube específicos bajo ciertas condiciones. Este mecanismo permite una autorización más flexible basada en políticas, cumpliendo los requisitos para un control de acceso seguro. Por ejemplo, puede conceder a los usuarios de DCS solo los permisos para las instancias de DCS operativas. La mayoría de las políticas definen permisos basados en API. Para conocer las acciones de API admitidas por DCS, véase **Políticas de permisos y acciones admitidas**.

En la **tabla 1** se enumeran todas las funciones y políticas definidas por el sistema compatibles con DCS.

Tabla 13-1 Roles definidos por el sistema y políticas admitidas por DCS

Nombre de rol/ política	Descripción	Tipo	Dependencia
DCS FullAccess	Todos los permisos para DCS. Los usuarios con estos permisos pueden operar y usar todas las instancias de DCS.	Política definida por el sistema	Acciones necesarias para comprar instancias profesionales: iam:permissions:listRolesForAgencyOnProject iam:agencies:listAgenciesiam:roles:listRoles iam:permissions:grantRoleToAgencyOnProject iam:agencies:createAgency iam:agencies:deleteAgency
DCS UserAccess	Permisos de usuario comunes para DCS, excluidos los permisos para crear, modificar, eliminar instancias de DCS y modificar especificaciones de instancia.	Política definida por el sistema	Ninguna
DCS ReadOnlyAccess	Permisos de solo lectura para DCS. Los usuarios a los que se han concedido estos permisos solo pueden ver los datos de instancia de DCS.	Política definida por el sistema	Ninguna

Nombre de rol/ política	Descripción	Tipo	Dependencia
DCS Administrator	Permisos de administrador para DCS. Los usuarios con estos permisos pueden operar y usar todas las instancias de DCS.	Rol definido por el sistema	Los roles **Server Administrator** y **Tenant Guest** deben asignarse en el mismo proyecto.
DCS AgencyAccess	Permisos para asignar a las agencias de DCS. Estos permisos son utilizados por un tenant para delegar DCS para realizar las siguientes operaciones en los recursos del tenant cuando sea necesario. Son irrelevantes para las operaciones realizadas por los usuarios autorizados. ● Consulta de una subred ● Consulta de la lista de subredes ● Consulta de un puerto ● Consulta de la lista de puertos ● Actualización de un puerto ● Creación de un puerto	Política definida por el sistema	Ninguna

NOTA

La política de **DCS UserAccess** es diferente de la política de **DCS FullAccess**. Si configura ambas instancias, no podrá crear, modificar, eliminar ni escalar instancias de DCS porque las instrucciones de denegación tendrán prioridad sobre las instrucciones permitidas.

La **tabla 2** enumera las operaciones comunes soportadas por cada política de sistema de DCS. Elija las políticas de sistema adecuadas de acuerdo con esta tabla.

Tabla 13-2 Operaciones comunes apoyadas por cada política de sistema

Operación	DCS FullAccess	DCS UserAccess	DCS ReadOnlyAccess	DCS Administrator
Modificación de parámetros de configuración de instancia	√	√	×	√
Eliminación de tareas en segundo plano	√	√	×	√
Acceso a instancias mediante Web CLI	√	√	×	√
Modificación del estado de ejecución de instancia	√	√	×	√
Ampliación de la capacidad de instancia	√	×	×	√
Cambio de contraseñas de instancia	√	√	×	√
Modificación de instancias de DCS	√	×	×	√
Realización de una conmutación principal/en espera	√	√	×	√
Copia de seguridad de datos de instancia	√	√	×	√

Operación	DCS FullAccess	DCS UserAccess	DCS ReadOnlyAccess	DCS Administrator
Análisis de claves grandes o claves de acceso rápido	√	√	×	√
Creación de instancias de DCS	√	×	×	√
Eliminación de archivos de copia de seguridad de instancia	√	√	×	√
Restauración de datos de instancia	√	√	×	√
Reajuste de la contraseña de instancia	√	√	×	√
Migración de datos de instancias	√	√	×	√
Descarga de datos de copia de seguridad de instancia	√	√	×	√
Eliminación de instancias de DCS	√	×	×	√
Consulta de los parámetros de configuración de instancia	√	√	√	√

Operación	DCS FullAccess	DCS UserAccess	DCS ReadOnlyAccess	DCS Administrator
Consulta de registros de restauración de instancia	√	√	√	√
Consulta de registros de copia de seguridad de instancia	√	√	√	√
Consulta de instancias de DCS	√	√	√	√
Consulta de tareas en segundo plano de instancias	√	√	√	√
Consulta de todas las instancias	√	√	√	√
Funcionamiento de consultas lentas	√	√	√	√

Enlaces útiles

- **Descripción de servicio de IAM**
- **Creación de un usuario y concesión de permisos de DCS**
- **Políticas de permisos y acciones admitidas**

14 Conceptos básicos

Instancia de DCS

Una instancia es la unidad de recurso mínima proporcionada por DCS.

Puede seleccionar el motor de caché de Redis o de Memcached. Los tipos de instancia pueden ser de nodo único, principal/en espera o de clúster. Para cada tipo de instancia, hay varias especificaciones disponibles.

Para más detalles, véase **Especificaciones de instancias de DCS** y **Tipos de instancia de DCS**.

Acceso a la red pública

Se puede acceder a una instancia de Redis 3.0 con una EIP (Elastic IP) en un cliente mediante la EIP. Se puede acceder a una instancia de Redis 4.0/5.0/6.0 en público usando Elastic Load Balance (ELB).

Stunnel se utiliza para cifrar el contenido de comunicación en el acceso a la red pública. El retardo de la red es ligeramente mayor que el de la VPC, por lo que el acceso a la red pública es adecuado para la puesta en marcha local en la fase de desarrollo.

Para obtener más información, consulte las **Instrucciones de acceso público**.

Acceso sin contraseña

Se puede acceder a las instancias de DCS en una VPC sin contraseñas. La latencia es menor porque no hay autenticación de contraseña.

Puede habilitar el acceso sin contraseña para instancias que no tienen datos confidenciales. Para garantizar la seguridad de los datos, no se le permite habilitar el acceso sin contraseña para instancias habilitadas con acceso a red pública.

Para obtener más información, véase **Configuración de la contraseña de Redis (Modificación del modo de acceso a la instancia de Redis)**.

Ventana de tiempo de mantenimiento

La ventana de tiempo de mantenimiento es el período en el que el equipo de servicio de DCS actualiza y mantiene la instancia.

El mantenimiento de la instancia de DCS tiene lugar solo una vez cada trimestre y no interrumpe los servicios. Aun así, se recomienda seleccionar un período de tiempo cuando la demanda de servicio es baja.

Al crear una instancia, debe especificar una ventana de tiempo de mantenimiento, que se puede modificar después de crear la instancia.

Para obtener más información, véase **Consulta de detalles de instancia**.

Despliegue entre AZ

Las instancias de separación principal/en espera, de clúster y de lectura/escritura se despliegan en diferentes AZ con fuentes de alimentación y redes físicamente aisladas. Las aplicaciones también se pueden desplegar en las AZ para lograr HA tanto para datos como para aplicaciones.

Al crear una instancia, puede seleccionar una AZ principal y una AZ en espera.

Partición

Una **partición** es una unidad de gestión de una instancia de clúster de DCS para Redis. Cada partición corresponde a un proceso de redis-servidor. Un clúster consta de varias particiones. Cada partición tiene varias ranuras. Los datos se distribuyen a las ranuras. El uso de particiones aumenta la capacidad de caché y las conexiones simultáneas.

Cada instancia de clúster consta de varias particiones. De forma predeterminada, cada partición es una instancia principal/en espera con dos réplicas. El número de particiones es igual al número de nodos principales en una instancia de clúster.

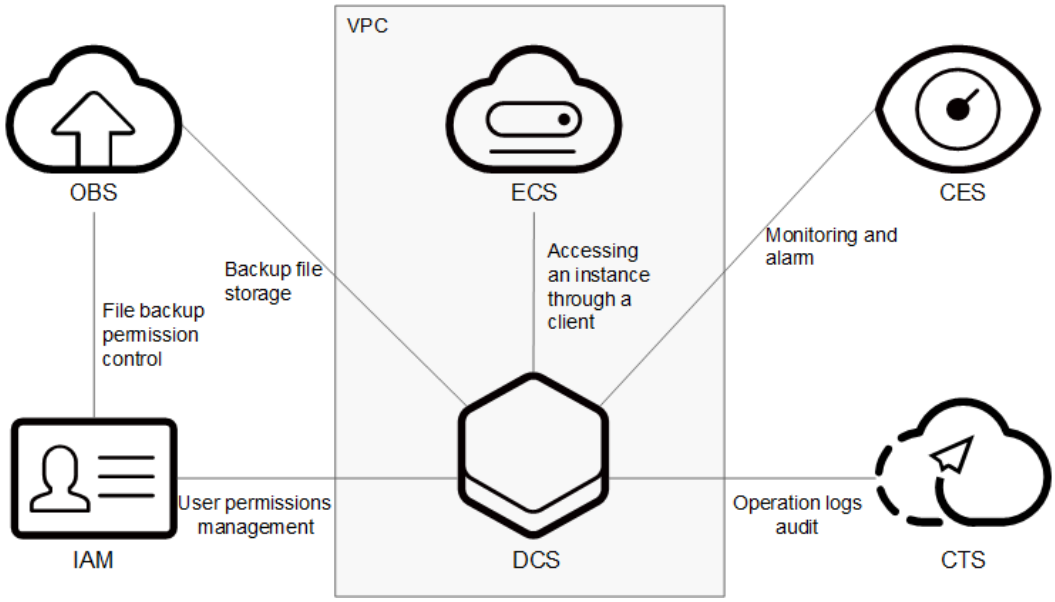
Réplica

Una réplica es un **nodo** de una instancia de DCS. Una instancia de réplica única no tiene nodo en espera. Una instancia de dos réplicas tiene un nodo principal y un nodo en espera. Por defecto, cada instancia principal/en espera tiene dos réplicas. Si el número de réplicas se establece en tres para una instancia principal/en espera, la instancia tiene un nodo principal y dos nodos en espera. Una instancia de nodo único tiene solo un nodo.

15 Servicios relacionados

DCS se utiliza junto con otros servicios en Huawei Cloud, incluidos VPC, ECS, IAM, Cloud Eye, CTS y Object Storage Service (OBS).

Figura 15-1 Relaciones entre DCS y otros servicios



VPC

Una VPC es un entorno de red virtual aislado en Huawei Cloud. Puede configurar intervalos de direcciones IP, subredes y grupos de seguridad, asignar EIP y asignar ancho de banda en una VPC.

DCS se ejecuta en las VPC. El servicio de VPC gestiona los EIP y ancho de banda, y proporciona grupos de seguridad. Puede configurar reglas de acceso para grupos de seguridad para proteger el acceso a DCS.

ECS

Elastic Cloud Server (ECS) es un servidor en la nube que ofrece recursos de cómputo bajo demanda y escalables para aplicaciones seguras, flexibles y eficientes.

Puede acceder y gestionar sus instancias de DCS mediante un ECS.

IAM

IAM proporciona la autenticación de identidad, la gestión de permisos y el control de acceso.

Con IAM, puede controlar el acceso a DCS.

Cloud Eye

Cloud Eye es un servicio de monitoreo seguro, escalable e integrado. Con Cloud Eye, puede supervisar su servicio DCS y configurar reglas de alarma y notificaciones.

Cloud Trace Service (CTS)

CTS le proporciona un historial de operaciones realizadas en recursos de servicios en la nube. Con CTS, puede consultar, auditar y realizar operaciones de retroceso. Las trazas incluyen las solicitudes de operación enviadas mediante la consola de gestión o las API abiertas y los resultados de estas solicitudes.

OBS

OBS proporciona un servicio de almacenamiento seguro y rentable utilizando objetos como unidades de almacenamiento. Con OBS, puede almacenar y gestionar el ciclo de vida de grandes cantidades de datos.

Puede almacenar archivos de copia de seguridad de instancia de DCS en OBS.

