Actividad 2 - Conceptos y comandos básicos de la replicación en bases de datos NoSQL

Docente:

Ing. JORGE CASTAÑEDA

Presentado por:
WILIAN DEIBY CARRERA ANDRADE
ID: 100157745

Base de datos avanzadas

Programa de Ingeniería de Software
Universidad Iberoamericana

Noviembre - 2024

Documento de Requerimientos No Funcionales: Redundancia y Disponibilidad 24x7

1. Introducción

Este documento define los requerimientos no funcionales en cuanto a la redundancia y disponibilidad 24x7 del sistema de gestión de la Copa América 2021. Dado que el sistema debe ser capaz de manejar grandes cantidades de datos en tiempo real, como los resultados de partidos, estadísticas de jugadores y decisiones de árbitros, es crítico que el sistema sea altamente disponible y tolerante a fallos.

2. Requerimientos de Redundancia y Alta Disponibilidad

El sistema debe ser capaz de operar sin interrupciones en todo momento, incluso si uno o más de sus nodos fallan. Esto se logrará mediante la implementación de una replicación en múltiples nodos y la configuración adecuada de estrategias de conmutación por error.

2.1. Requerimientos de Redundancia

La base de datos debe estar replicada en mínimo tres nodos para garantizar la redundancia y protección de los datos. Esto asegurará que, en caso de fallos en un nodo o un grupo de nodos, los datos continúen siendo accesibles desde otros nodos.

Requerimientos específicos:

- Tres nodos deben ser suficientes para garantizar la disponibilidad en caso de que uno de los nodos falle.

- Los datos deben estar replicados en cada nodo en tiempo real para asegurar que las actualizaciones en el nodo primario sean reflejadas rápidamente en los nodos secundarios.

- La replicación debe ser asincrónica para reducir la latencia de las operaciones de escritura.

2.2. Requerimientos de Disponibilidad 24x7

El sistema debe estar disponible 24 horas al día, 7 días a la semana, sin interrupciones de servicio. Para lograr esto, se utilizará un mecanismo de conmutación por error automática y balanceo de carga en el sistema de bases de datos.

Requerimientos específicos:

- Los nodos secundarios deben ser elegibles para ser promovidos a primarios en caso de que el nodo primario falle, asegurando que la base de datos siga operando sin tiempos de inactividad significativos.

- El tiempo de inactividad por fallos debe ser inferior a 5 segundos.

- El sistema debe permitir actualizaciones de software y mantenimiento sin que los usuarios finales experimenten interrupciones.

2.3. Estrategia de Replicación

El sistema debe usar la replicación en conjuntos de réplicas (Replica Sets) de MongoDB para proporcionar redundancia. Esta estrategia garantiza que los datos estén replicados en múltiples nodos y que, en caso de que el nodo primario falle, un nodo secundario pueda ser promovido automáticamente como nuevo primario.

- Replicación síncrona/asincrónica: La replicación se debe configurar en modo asincrónico para permitir una alta disponibilidad de las operaciones de escritura y reducir la latencia.

- Conmutación por error: La conmutación por error debe ser automática para garantizar que, en caso de un fallo del nodo primario, un nodo secundario sea promovido inmediatamente.

- Manejo de fallos: El sistema debe ser capaz de manejar múltiples fallos de nodos sin perder datos.

3. Requerimientos de Escalabilidad

El sistema debe ser escalable, permitiendo agregar nuevos nodos o servidores en el futuro sin interrumpir el servicio. Se recomienda usar una infraestructura que permita aumentar la capacidad de manera flexible (escalabilidad horizontal).

4. Seguridad y Control de Acceso

El sistema debe contar con políticas de seguridad para proteger los datos y garantizar que solo los usuarios autorizados puedan acceder a la información sensible. Las comunicaciones entre los nodos deben ser cifradas.

Consultas para Replicar las Bases de Datos en Mínimo 3 Nodos

Estrategia de Replicación:

Para la réplica de bases de datos en MongoDB, utilizaremos un Replica Set. Los pasos para crear un entorno de replicación que cumpla los requerimientos mencionados son los siguientes:

1. Definir los nodos:

- Nodo primario: hostname1:27017

- Nodo secundario 1: hostname2:27017

- Nodo secundario 2: hostname3:27017

2. Crear el Replica Set:

En cada nodo, ejecutar el siguiente comando para iniciar el proceso de replicación:

```
mongo --port 27017
```

En el shell de MongoDB, ejecutar el siguiente comando en el nodo primario para iniciar el Replica Set:

```
rs.initiate({
  _id: "copaAmericaReplicaSet",
  members: [
    { _id: 0, host: "hostname1:27017" },
    { _id: 1, host: "hostname2:27017" },
    { _id: 2, host: "hostname3:27017" }
  ]
```

})

Esto iniciará el Replica Set y añadirá tres nodos a él.

3. Verificar el estado del Replica Set:

Para verificar que los nodos han sido correctamente añadidos al Replica Set, ejecutar el siguiente comando:

rs.status()

4. Agregar un nodo secundario manualmente:

Si se requiere agregar un nodo secundario adicional a un Replica Set ya iniciado, se puede usar el siguiente comando:

rs.add("hostname4:27017")

Esto agregará el nodo secundario hostname4 al Replica Set.

Comandos adicionales para la configuración y la conmutación por error:

1. Habilitar la conmutación por error (en caso de que el nodo primario falle):

rs.stepDown()

Esto promoverá un nodo secundario a primario, si el nodo primario falla.

2. Habilitar la monitorización de los nodos:

rs.printReplicationInfo()

Conclusiones

Con esta configuración, el sistema estará preparado para manejar grandes volúmenes de datos, asegurar la alta disponibilidad y proveer redundancia en caso de fallos. La implementación de un Replica Set en MongoDB asegura que los datos estén siempre disponibles, incluso si uno de los nodos falla, garantizando que el sistema de gestión de la Copa América 2021 esté disponible **24x7**.