Tarea Corta II

Tecnológico de Costa Rica
Escuela de Ingeniería en Computación
Bases de Datos II (IC 4302)
Segundo Semestre 2023

1. **Objetivo General**
   - ➔ Implementar mecanismos de copias de respaldo en bases de datos SQL y NoSQL.
2. **Objetivos Específicos**
   - ➔ Automatizar una solución mediante el uso de Docker, Docker Compose, Kubernetes y Helm Charts.
   - ➔ Implementar mecanismos de respaldo en bases de datos en AWS Cloud.
   - ➔ Implementar mecanismos para restaurar bases de datos desde AWS Cloud.
   - ➔ Desarrollar habilidades en scripting utilizando Bash Shell.
   - ➔ Crear tareas programadas utilizando Cron Expressions.
3. **Datos Generales**
   - ➔ El valor de la tarea corta: 10%
   - ➔ La tarea debe ser implementada en grupos de máximo 5 personas.
   - ➔ La **fecha de entrega** es 17/10/2023 antes de las 10:00 pm.
   - ➔ Cualquier indicio de copia será calificado con una nota de 0 y será procesado de acuerdo con el reglamento. La copia incluye código/configuraciones que se puede encontrar en Internet y que sea utilizado parcial o totalmente sin el debido reconocimiento al autor.
   - ➔ La revisión es realizada por el profesor asignado al curso, él mismo se reserva el derecho de solicitar una revisión virtual con los miembros del grupo para evacuar cualquier duda sobre la implementación.
   - ➔ Se debe incluir documentación con instrucciones claras para ejecutar la tarea corta, en caso de no ser así se obtiene una nota de 0.
   - ➔ Se espera que todos y todas las integrantes del grupo entiendan la implementación suministrada.
   - ➔ Se deben seguir buenas prácticas de programación en el caso de que apliquen, entre ellas:
     - ◆ Documentación interna y externa.
     - ◆ Estándares de código.
     - ◆ Diagramas de arquitectura.
     - ◆ Diagramas de flujo
     - ◆ Pruebas unitarias.
   - ➔ Toda documentación debe ser implementada en Markdown.
   - ➔ Si la tarea no se encuentra automatizado obtendrá una nota de 0. Si la tarea no se entrega completa, la porción que sea entregada debe estar completamente automatizada, de lo contrario se obtendrá una nota de 0.
   - ➔ En caso de no entregar documentación se obtendrá una nota de 0.

- ➔ El email de entrega debe contener una copia de la tarea en formato tar.gz y un enlace al repositorio donde se encuentra almacenado, debe seguir los lineamientos en el programa de curso.
- ➔ Los grupos de trabajo se deben autogestionar, la persona facilitadora del curso no es responsable si un miembro del equipo no realiza su trabajo.
- ➔ En consulta o en mensajes de correo electrónico o Telegram, como mínimo se debe haber leído del tema y haber tratado de entender sobre el mismo, las consultas deben ser concretas y se debe mostrar que se intentó resolver el problema en cuestión.

4. **Descripción**

Para esta tarea corta implementaremos estrategias básicas de recuperación de fallas en bases de datos SQL y NoSQL, esto mediante la creación/restauración automática de copias de respaldo (backups/snapshots), para este ejercicio trabajaremos con los siguientes motores de base de datos:

- MariaDB
- PostgreSQL
- Elasticsearch
- MongoDB
- Neo4J
- CouchDB

Estos motores deben ser instalados mediante Kubernetes utilizando Helm Charts (al igual que ha sucedido en otras ocasiones), luego de esto se debe crear un Helm Chart llamado backups en el mismo se especifican las configuraciones necesarias para crear/restaurar los backups de cada una de las bases de datos, el mismo deberá tener al menos las siguientes opciones:

- namespace: es el namespace de Kubernetes donde se debe crear/restaurar la copia de seguridad.
- connectionString: la dirección de la base de datos donde se desea restaurar/crear la copia de seguridad.
- bucketName: nombre del bucket donde se debe guardar la copia de seguridad o desde donde se tiene que obtener el mismo.
- path: representa el folder donde se tiene que guardar/obtener la copia de seguridad.
- maxBackups: número máximo de backups a retener (funcionalidad opcional).
- awsSecret: nombre del secret de Kubernetes que contiene el key para poder subir/bajar copias de seguridad.
- name: nombre de la copia de seguridad que se debe descargar y restaurar.
- schedule: Expresión cron de Kubernetes.
- diskSize: tamaño de disco a utilizar para crear backups.

- storageClass: nombre de la clase de almacenamiento de Kubernetes que se utilizara para guardar temporalmente las copias de seguridad.
- provider: siempre AWS
- type: restore/backup.

Con excepción de Elasticsearch, todos los demás motores de búsqueda deben usar un Kubernetes CronJob y un Kubernetes Job para crear copias de seguridad (un ejemplo será proporcionado por el profesor) y un Kubernetes Job para restaurar las copias de seguridad.

La funcionalidad es que cada vez que se llama al CronJob o Job de creación de backups, se deberá crear una copia de seguridad y almacenarlo en AWS, opcionalmente si los grupos implementan una forma de mantener las ***n*** copias de seguridad más recientes y que esto sea configurable, se recibirá un 3% extra sobre el total de la tarea corta. Todas las copias de seguridad deben ser almacenadas con la fecha en que fueron creadas con formato YYYYmmDDHHMM (año, mes, día, hora y minuto).

En el caso de restaurar una copia de seguridad, cuando se ejecuta el Job, restaurar el backup especificado en la configuración.

Todo debe estar debidamente automatizado y documentado, de lo contrario se obtendrá una nota de 0.

Documentación

Se deberá entregar una documentación que al menos debe incluir:

- Instrucciones claras de como ejecutar su tarea corta.
- Pruebas realizadas, con pasos para reproducirlas.
- Recomendaciones y conclusiones (al menos 10 de cada una).
- La documentación debe cubrir todos los componentes implementados o instalados/configurados, en caso de que algún componente no se encuentre implementado, no se podrá documentar y tendrá un impacto en la completitud de la documentación.
- Referencias bibliográficas donde aplique.

Imaginen que la documentación está siendo creada para una persona con conocimientos básicos en el área de computación y ustedes esperan que esta persona pueda ejecutar su tarea corta y probarlo sin ningún problema, el uso de imágenes, diagramas, code snippets, videos, etc. son recursos que pueden ser útiles.

La documentación debe estar implementada en Markdown y compilada en un PDF.

5. **Recomendaciones**
   - Utilizar telnet/curl/postman para interactuar con bases de datos que exponen un API y verificar que los mismos están funcionando correctamente.
   - Utilizar Docker Desktop para instalar Kubernetes en Windows.
   - Realicen peer-review/checkpoints semanales y no esperen hasta el último momento para integrar

su aplicación.

- Crear un repositorio de GitHub e invitar al profesor.
- Realizar commits y push regulares.
- Para la revisión, se recomienda tener clientes de administración de las bases de datos para poder realizar consultas y observar su comportamiento.
- Se deben generar datasets de prueba para probar los backups.

6. **Entregables**

- Documentación.
- Docker files, Docker Compose files y Helm Charts junto con todos los archivos/scripts requeridos para ejecutar su tarea corta.

7. **Evaluación**

Funcionalidad / Requerimiento	Porcentaje
Documentación (*)	20%
Instalación de motores	20%
Creación/restauración de backups (10% por base de datos)	60%
	100

(*) La completitud de la documentación se encuentra acotado por la completitud de otros requerimientos.