Universidad Nacional

Escuela de Informática

Programa de Licenciatura

Curso: Sistemas Distribuidos

Segundo Proyecto

El objetivo de este proyecto es la creación de un conector Web hacia un servidor de bases de datos particionada. Una partición es una división de una base de datos lógica o sus elementos constitutivos en distintas partes independientes. Para ello se utilizará el protocolo TreeQL disponible en: https://treeql.org

Se deben crear tres programas, primero un conector que capturará todas las solicitudes HTTP y las pasará a un coordinador de procesos a través de ZMQ. Luego se debe programar el servidor de bases de datos que recibe solicitudes desde el coordinador y accede a una base de datos SQLite. Los datos serán manipulados usando el formato JSON, tal como se describe en TreeQL.

Protocolo TreeQL

TreeQL es un lenguaje de consulta para API y un tiempo de ejecución para completar esas consultas con sus datos existentes. TreeQL proporciona una descripción completa y comprensible de los datos en su API, brinda a los clientes el poder de pedir exactamente lo que necesitan y nada más, facilita la evolución de las API a lo largo del tiempo y habilita herramientas familiares para desarrolladores.

```
GET    /records/{table}      - list records
POST   /records/{table}      - create a record
GET    /records/{table}/{id} - read a record
PUT    /records/{table}/{id} - update a record's values
DELETE /records/{table}/{id} - delete a record
```

Se deben implementar únicamente las opciones básicas, no es necesario programar las opciones: include, join, filtros, ordenamientos, paginación o límite de registros.

Particionamiento de la base de datos

La base de datos se debe particionar en cuatro partes, para lo que se deben ejecutar cuatro procesos de acceso a bases de datos SQLite (aparte del proceso conector y el proceso coordinador). El coordinador de procesos que recibe las solicitudes desde el conector HTTP debe enviar cada consulta de datos a todos los servidores (patrón PUBLICAR/SUBSCRIBIR) para que alguno de ellos conteste la solicitud al coordinador y este a su vez responda al conector HTTP. Sin embargo, las operaciones de creación de registros se asignarán a un único servidor de datos (note que el patrón PUSH/PULL asigna las solicitudes en forma automática). Es importante resaltar que cada proceso accede a una base de datos SQLite diferente.

Pruebas de corridas

Se deben crear una base de datos de ejemplo con al menos 4 tablas y 10 registros por tabla (los registros se deben distribuir entre todas las copias de la base de datos). Se debe entregar un documento en donde se muestren al menos 10 pruebas de la ejecución de solicitudes al conector Web y la base de datos. Se debe mostrar tanto el formato de la solicitud al servidor como la respuesta que se recibe desde él. Se puede utilizar el programa CURL para realizar dichas consultas, o bien, el complemento de Firefox llamado RestClient.