Memoria Redis

10/03/2021

Introducción

Esta es la memoria de la práctica de Redis para la asignatura Gestión/Almacenamiento de datos no estructurados. A continuación se explica el código adjunto en el fichero redis_twitter.ipynb, las distintas funciones que se han creado y las decisiones de diseño que se han tomado.

Para empezar, se ha decido crear la clase MicroBlog que contendrá todas las funciones que se piden en el ejercicio. Se ha elegido crear esta clase en vez de hacer las funciones por separado por dos motivos: En el enunciado se pide que las funciones nuevoFollower y nuevoFollowing no sean accesibles directamente, y la manera más sencilla de conseguir esto es que sean métodos privados de una clase.

El segundo motivo es que si esta base de datos fuera un proyecto real, sería conveniente impedir que los usuarios accedan directamente a la base de datos con la API de Redis-py. Encapsulando la base de datos en la clase se consigue que solo se pueda modificar la información almacenada con los propios métodos de la clase.

Ejercicio 1

Es esta sección se describen los métodos y estructuras pedidos en el apartado 3.1. Diseño de la base de datos.

Constructor

En el constructor de la clase MicroBlog se realiza la conexión con la base de datos Redis especificada poniendo la opción decode_responses=True. A continuación se ejecuta el comando FLUSHALL para borrar todas la información preexistente.

nuevoUsuario

Primero se obtiene un identificador para el nuevo usuario usando la función de Redis INCR. A continuación se crea una clave llamada 'user:id', donde id es el identificador del nuevo usuario, cuyo valor asociado es el nombre del usuario. Además, se introduce el nombre y el identificador del usuario en un hash cuya clave es 'users'.

'user:1' → 'Marta'

'user:2' → 'Pedro'

'user:3' → 'Ana'

'users' → {'Marta':1, 'Pedro':2,'Ana':3}

nuevoFollower

Primero obtiene los identificadores del usuario y del follower a partir del hash 'users' que se comentó en el método nuevoUsuario. Después introduce el identificador del follower y la fecha en formato timestamp en un hash de clave 'followers:id' donde id es el identificador del usuario que adquiere un follower.

'followers:1' → {'2' : '1536514789', '3' : '1536514902'}

'followers:2' → {'1' : '1536512478', '3' : '1536530147'}

'followers:3' → {'2' : '1536514789'}

nuevoFollowing

Muy parecido a nuevoFollower. Primero obtiene los identificadores del usuario y del following a partir del hash 'users' que se comentó en el método nuevoUsuario. Después introduce el identificador del following y la fecha en formato timestamp en un hash de clave 'followingd:id' donde id es el identificador del usuario que se convierte en follower.

'following:1' → {'2' : '1536514789', '3' : '1536514902'}

'following:2' → {'1' : '1536512478', '3' : '1536530147'}

'following:3' → {'2' : '1536514789'}

seguir

Primero llama al método privado 'existeUsuario' para los dos nombres de usuario introducidos. Si ambos usuarios existen, llama a los métodos 'nuevoFollower' y 'nuevoFollowing'.

nuevoPost

Primero comrpueba que el usuario que está publicando el post existe. Si el usuario existe, genera un nuevo identificador numérico para el post con la función INCR y obtiene el identificador del usuario que ha escrito el post. Después crea un hash de clave 'post:post_id' en el que almacena en mensaje, el identificador del usuario y la fecha en la que se publicó el mensaje.

A continuación añade el id del post a un set cuya clave es 'posts:user_id' donde user_id es el identificador del usuario que publica el mensaje. Por último, busca los ids de todos los followers del usuario y añade el id del post a los sets 'posts:follower_id', pero solo si la fecha de publicación es posterior a la fecha en la que se produjo en follow.

'post:1' → {'user_id':1, 'fecha':'1598741852', 'mensaje': 'mensaje de prueba'}

'post:2' → {'user_id':1, 'fecha':'1598741982', 'mensaje': 'Este es otro mensaje'}

'post:3' → {'user_id':2, 'fecha':'1598742002', 'mensaje': 'Mensaje de otro user'}

'posts:1' → ('1', '2')

'posts:2' → ('1', '2', '3')

Ejercicio 2

En esta sección se describe el proceso por el cual se introdujeron los datos de los ficheros relations.csv y twitter_sample.csv en la base de datos Redis.

Primero lee el fichero relations.csv con pandas. Una vez convertido en dataframe, modifica la columna Following_Time para que sea de tipo datetime y crea la columna Unix_Following_Time que contiene las fechas en formato unix timestamp. A continuación se usa un bucle en el que se llama al método 'nuevoUsuario' para introducir a todos los usuarios en la base de datos, y otro bucle en el que se llama al método 'seguir' para meter en la base de datos la información sobre los follows.

Luego, se lee el fichero twitter_sample.csv y se le aplican las mismas transformaciones de fechas que a relations.csv. Por último, se llama en un bucle al método 'nuevoPost' para meter todos los twits en la base de datos.

Ejercicio 3

Es esta sección se describen los métodos pedidos en el apartado 3.3. Pruebas.

obtenerFollowers

Primero se comprueba que el usuario existe y se obtiene su id. A partir de su id se obtiene el id de todos sus followers y las fechas en las que lo siguieron desde el hash 'followers:user_id'. De los ids de los followers se obtiene una lista con sus nombres, las fechas se convierten a formato datetime y se devuelve un array 2D con los nombres de los followers y la fecha en la que hicieron el follow.

obtenerFollowing

Primero se comprueba que el usuario existe y se obtiene su id. A partir de su id se obtiene el id de todos sus followings (gente a la que sigue) y las fechas en las que los siguió desde el hash 'followings:user_id'. De los ids de los followings se obtiene una lista con sus nombres, las fechas se convierten a formato datetime y se devuelve un array 2D con los nombres y la fecha.

obtenerTimeline

Primero se comprueba que el usuario existe y se obtiene su id. Depués se obtienen los ids de los posts de su timeline usando la función SORT y ordenando por fecha de los más recientes a los más antiguos. Si la variable tweets_propios es False, se quitan los ids de los posts del propio usuario. Por último se obtienen los mensajes de los posts a partir de los ids.