Redis

Presentado por :
Luis Alberto Mosquera

Marzo 2023

CONTENIDO

1. **Historia**
   Origen Regis
2. **¿Qué es?**
   Descripción de Redis
3. **Características**
   Características de Redis
4. **Uso**
   Ejemplo de aplicación de Redis
5. **Comandos**
   Comandos de Redis

Marzo 2023

1

Historia
Origen de Redis

Marzo 2023

Historia de Redis

- ✓ Empezó en el año 2009 por Salvatore Sanfilipo.
- ✓ El objetivo era mejorar la escalabilidad de su empresa emergente italiana (Proyecto llamado LLOGG).
- ✓ En el año 2012, VMWare contrata a Salvatore y a otros integrantes para trabajar de lleno en Redis.

2

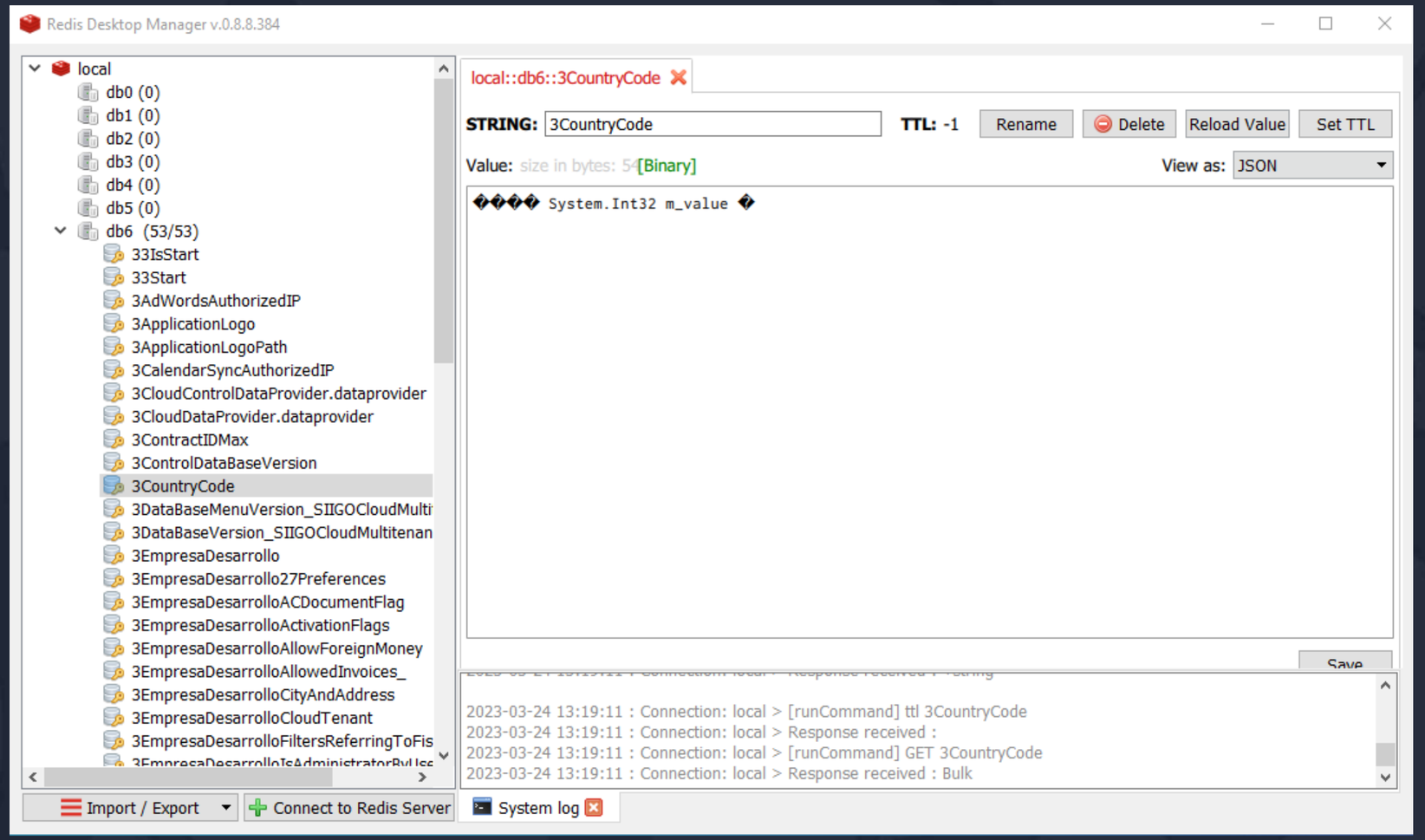
¿Qué es?
Descripción de Redis

Marzo 2023

¿Qué es de Redis?

- ✓ También es llamado servidor de diccioanrio remoto (remote diccionary server)
- ✓ Es una base de datos no relacional
- ✓ Almacena la información de forma clave-valor en memoria
- ✓ Es un proyecto de código libre

Imagen de Redis



3

Características de Redis

Marzo 2023

Características de Redis

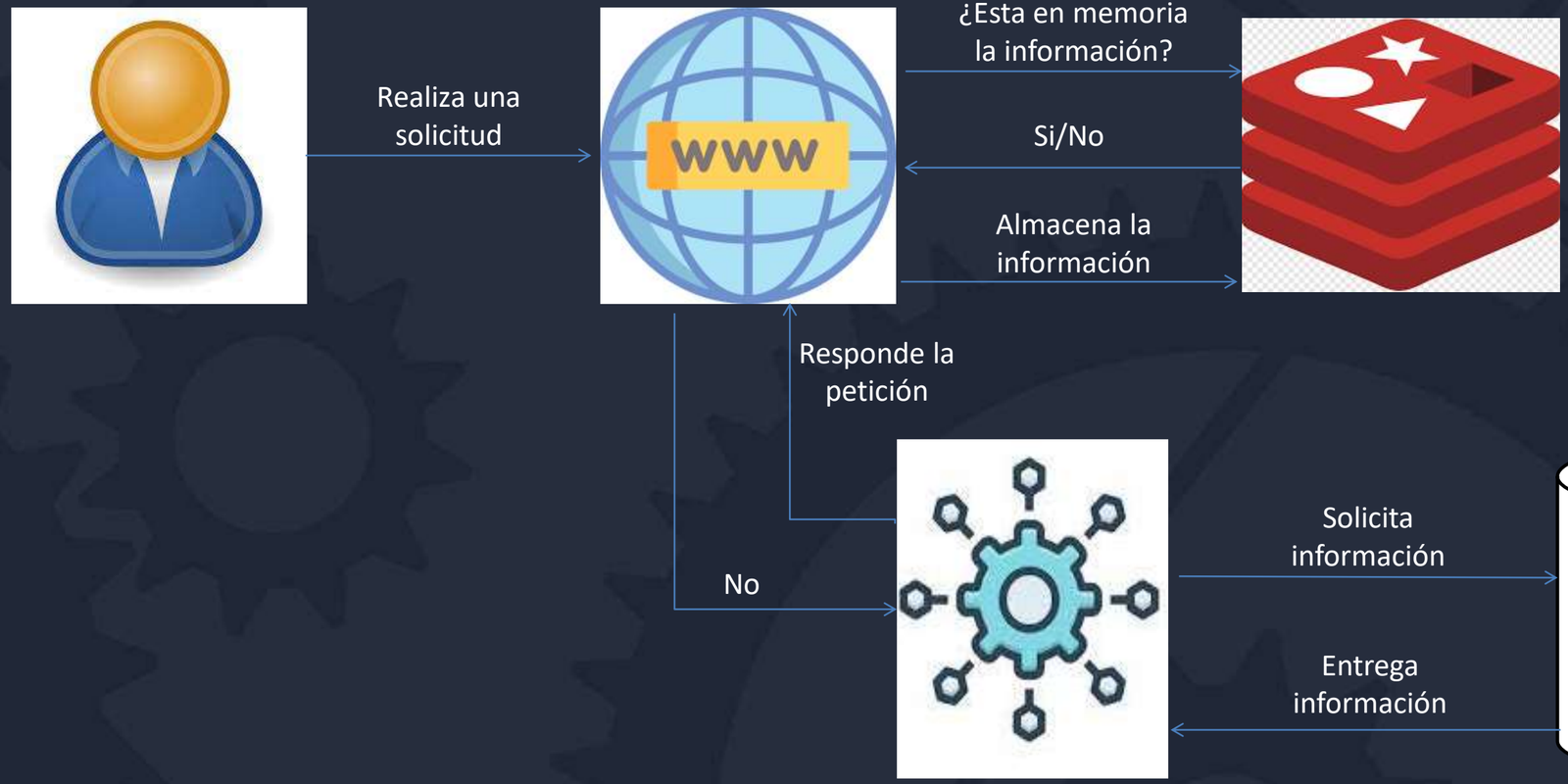
- ✓ Estructura de datos flexibles
- ✓ Replicación y persistencia
- ✓ Modelo abierto
- ✓ Facilidad de uso
- ✓ Disponibilidad y escalabilidad

4

Uso
Ejemplo de aplicación de Redis

Marzo 2023

Uso de Redis


Realiza una solicitud
www
¿Esta en memoria la información?
Si/No
Almacena la información
Responde la petición
No
Solicita información
Entrega información
Base datos

5 Comandos de Redis

Marzo 2023

Comandos de Redis

- Set : Permite crear la llave (clave-valor).
  - ✓ Ejemplo: SET name “John Wick”
- Get: Busca la información por la clave.
  - ✓ Ejemplo: GET name
- Del nombre de la clave: Borra la información por su clave
  - ✓ Ejemplo: Del name
- Exists nombre de la clave: Determinar si una clave existe
  - ✓ Ejemplo: EXISTS name
- ✓ KEYS *: Devuelve todas la claves que existen
- ✓ Flushall : Borra todas las claves que existen.

Comandos de Redis (imagenes)

Comandos basicos

```
127.0.0.1:6379> SET name "John Wick"
OK
127.0.0.1:6379> GET name
"John Wick"
127.0.0.1:6379> SET age 32
OK
127.0.0.1:6379> get age
"32"
127.0.0.1:6379> DEL age
(integer) 1
127.0.0.1:6379> get age
(nil)
127.0.0.1:6379> EXISTS age
(integer) 0
127.0.0.1:6379> EXISTS name
(integer) 1
127.0.0.1:6379> KEYS *
1) "name"
127.0.0.1:6379> SET age 32
OK
127.0.0.1:6379> KEYS *
1) "name"
2) "age"
127.0.0.1:6379> flushall
OK
127.0.0.1:6379> KEYS *
(empty list or set)
127.0.0.1:6379>
```

Tiempo de almacenamiento

```
127.0.0.1:6379> SET name "John"
OK
127.0.0.1:6379> ttl name
(integer) -1
127.0.0.1:6379> expire name 10
(integer) 1
127.0.0.1:6379> ttl name
(integer) 7
127.0.0.1:6379> ttl name
(integer) 6
127.0.0.1:6379> ttl name
(integer) 5
127.0.0.1:6379> ttl name
(integer) 4
127.0.0.1:6379> ttl name
(integer) 3
127.0.0.1:6379> ttl name
(integer) 3
127.0.0.1:6379> ttl name
(integer) 1
127.0.0.1:6379> ttl name
(integer) -2
127.0.0.1:6379> GET name
(nil)
127.0.0.1:6379>
```

Manejo de listas

```
127.0.0.1:6379> lpush pets dog
(integer) 1
127.0.0.1:6379> get pets
(error) WRONGTYPE Operation against a key holding the wrong kind of value
127.0.0.1:6379> lrange pets 0 -1
1) "dog"
127.0.0.1:6379> lpush pets cat
(integer) 2
127.0.0.1:6379> lrange pets 0 -1
1) "cat"
2) "dog"
127.0.0.1:6379> rpush pets fish
(integer) 3
127.0.0.1:6379> lrange pets 0 -1
1) "cat"
2) "dog"
3) "fish"
127.0.0.1:6379> lpop pets
"cat"
127.0.0.1:6379> lrange pets 0 -1
1) "dog"
2) "fish"
127.0.0.1:6379> rpop pets
"fish"
127.0.0.1:6379> lrange pets 0 -1
1) "dog"
127.0.0.1:6379>
```

GRACIAS !

Marzo 2023