Université de Lille

TP 5.5 REDIS - CRUD

R5.A.10 - NoSQL
BUT3 - Info
Philippe Mathieu
2024–2025

Objectifs

Redis est une base de données NoSQL open-source de type `clé-valeur` à haute disponibilité. Redis peut gérer jusqu'à 16 databases avec 2^{32} clés dans chacune. Ce TP a pour objectif de faire un tour rapide de ses différentes fonctionnalités. Manuel Redis

Exercice 1 : Installation du serveur Redis

Il existe différentes manières d'installer Redis. Via votre gestionnaire de paquetage (`apt-get install`) , via le tarball fourni sur le site officiel https://redis.io/download/ ou via un service docker (voir Annexe). Pour des raisons de simplicité nous utiliserons docker.

Q1. créez une fois pour toutes le service docker avec la dernière version de Redis

```
docker run --name monredis -v mes_data_redis:/data -p 6379:6379 -d
↪  redis:latest
```

Q2. Connectez vous à ce service via le client standard `redis-cli`

```
docker exec -it monredis redis-cli
ou
docker exec -it monredis bash
```

Exercice 2 : Bien démarrer

Q1. Tapez la commande `PING`
Si tout s'est bien passé le serveur vous renvoie un `PONG`

Q2. Tapez la commande `DBSIZE`
Cette commande fournit le nombre de clés de la base courante. A priori 0

Q3. Tapez `SELECT 5`
Cette commande change de base et passe à la base 5. Cela se voit dans le prompt. Quand rien n'est indiqué c'est que l'on est sur la base 0.

Q4. Tapez la commande `CONFIG GET *`
Cette commande permet de voir l'ensemble des paramètres du serveur actuel. Redis fournit toujours ses réponses avec les valeurs écrites sous les clés. L'ensemble des lignes de sorties est numéroté. Vous devriez avoir 404 lignes, ce qui correspond à 202 clés.

Q5. On peut bien sûr rechercher une clé spécifique. Tapez la commande `CONFIG GET dbfilename`
Vous obtenez alors une clé-valeur spécifique.

Q6. Un joker peut être utilisé dans les recherches. Tapez la commande `CONFIG GET active*`
Des jokers peuvent être utilisés pour référencer des groupes de clés. Il y en a ici 9 (active, tls, repl,...).

Q7. Tapez la commande `QUIT`
Vous venez de quitter votre première session avec REDIS.

Tour d'horizon des différents types de données

Pour les exercices suivants, reconnectez vous à la base 0

Gradez sous le coude la liste des commandes : https://redis.io/commands/

Exercice 3 : Les chaines

Les chaines de caractères sont les données de bases. Il n'y a pas d'entier en REDIS, tout est représenté sous forme de chaines. Attention : les données sont *case-sensitive*.

Q1. Affectez la clé `nom` à la valeur `paul`

Q2. Affichez la valeur de cette clé

Q3. Affectez la clé `compteur` à la valeur `2`

Q4. incrémentez `compteur` de 1

Q5. Affichez la valeur de cette clé

Q6. incrémentez `compteur` de 10

Q7. Affichez la valeur de cette clé

Q8. Affichez toutes les clés

Exercice 4 : Le TTL

Il est possible de définir le temps d'expiration d'une clé à l'aide de la commande `EXPIRE` ou de l'argument `EX`. -1 indique qu'il n'y a pas d'expiration. Quand la valeur arrive à -2 la clé n'est plus accessible.

Q1. Affectez la clé `compteur` à la valeur `20`

Q2. Donnez à cette clé une validité de 10 sec

Q3. Affichez plusieurs fois sa validité jusqu'à arriver à -2

Q4. Tentez de lire la valeur de `compteur`

Q5. Recommencez la même démarche mais en indiquant une date de validité à l'affectation (et pas une durée comme précédemment).
Redis veut de l'EPOCH unix , pas des dates ISO. Pour convertir une date ISO en EPOCH soit on passe par un site https://www.epochconverter.com/ soit on utilise la commande unix `Date`

Q6. Affichez les clés de la base

Exercice 5 : Les listes

Les listes permettent de sauvegarder une liste d'éléments (l'ordre d'empilement est toujours conservé)

Q1. Créez une liste `chats` avec plusieurs races de chats (birman mainecoon ragdol somali abyssin bengal ...)

Q2. Affichez la longueur de la liste `chats`

Q3. Affichez les 3 premiers de la liste

Q4. Rajoutez 3 races en fin de liste (burmese siamois persan)

Q5. Affichez la nouvelle longueur

Q6. Affichez toute la liste

Exercice 6 : Les sets

Les sets permettent de sauvegarder des informations de manière non ordonnées mais avec unicité. Contrairement aux liste un set ne peut contenir plusieurs fois le même élément.

Q1. Créez un set `mult3` contenant tous les multiples de 3 jusque 20

Q2. Créez un set `mult5` contenant deux fois tous les multiples de 5 jusque 20

Q3. Affichez toutes les valeurs de `mult5`

Q4. Combien de valeurs avez vous rentré ? combien y-en a t-il ? Pourquoi ?

Q5. Affichez l'intersection de ces 2 sets

Q6. Rangez dans `res` la différence `mult3 - mult5`

Exercice 7 : Les sets avec score

Il est aussi possible de créer des sets ordonnés en introduisant une notion de score.

Q1. Ajoutez dans une clé `classement` le nom `paul` avec un score de 10

Q2. Ajoutez dans une clé `classement` le nom `marie` avec un score de 15

Q3. Ajoutez dans une clé `classement` le nom `louis` avec un score de 3

Q4. Affichez tout le contenu de `classement`

Q5. Affichez le rang de `paul`

Q6. Effacez `louis`

Q7. Affichez à nouveau le rang de `paul`

Exercice 8 : Les hashes

Les hashs permettent de sauvegarder des informations plus complexes qui ont des sous propriétés, des sortes d'objets.

Q1. Créez un hashes `user:1` avec la propriété `nom` à `paul`

Q2. Ajoutez la propriété `age` à 23

Q3. Ajoutez la propriété `email` à `paul@gmail.com`

Q4. Affichez tous les éléments

Q5. Créez en une seule instruction un hashes `user:2` avec les propriétés nom,prenom,age aux valeurs virginie, 25, virginie@gmail.com

Q6. Affichez toutes les valeurs de `user:2`

Q7. Affichez uniquement la valeur de `nom` de `user:2`

Utilisation standard de Redis

Exercice 9 : Pub/Sub

Ouvrir trois `redis-cli` sur le même serveur. Le premier servira à diffuser les messages, et chacun des deux autres sera abonné à un canal spécifique.

Q1. Pour s'y retrouver facilement, Redis permet d'affecter un nom à chaque client avec la commande `client setname <lenom>`. Affectez `Diffuseur`, `Paul` et `Virginie` à chacun de ces clients.

Q2. La commande `client list` permet d'avoir les informations sur l'ensemble des clients connectés. Quel est l'age de chaque client ? Quelle est leur durée d'inactivité ?

Q3. Dans le client `Paul` tapez : `SUBSCRIBE sports`

Q4. Dans le client `virginie` tapez : `SUBSCRIBE news`
Vous constatez que les clients sont maintenant bloqués et en attente.

Q5. Dans le client `Diffuseur` tapez :

```
PUBLISH sports "PSG va à nouveau rencontrer Marseille"
PUBLISH news "Les élections présidentielles auront lieu le 10 avril"
```

Le mode `SUBSCRIBE` peut-être arrêté avec un `Ctrl+C` dans la console du client.

Exercice 10 : Utiliser Redis dans un langage de programmation

L'une des utilisations fréquentes de Redis, consiste à l'utiliser comme système de cache : plutôt que de refaire des calculs, on les range dans REDIS et à chaque recherche, on vérifie si ça n'a pas déjà été calculé.

Q1. Ecrire un programme java `TP55RestSansCache.java` avec comme paramètre un numéro de `user`, et qui exécute une requête REST de type GET sur le site jsonplaceholder et affiche les données du `user` correspondant.
Un programme équivalent à `curl -X GET https://jsonplaceholder.typicode.com/users/8`

```
....
URL url = new URL("https://jsonplaceholder.typicode.com/users/5");
// + Integer.parseInt(args[0]));
HttpURLConnection con = (HttpURLConnection) url.openConnection();
con.setRequestMethod("GET");
int responseCode = con.getResponseCode();
System.out.println("CODE: "+responseCode); // A priori 200
.....
// lire les lignes de la réponse en parcourant con.getInputStream()
....
con.disconnect();
```

Si vous lancez plusieurs fois ce programme, il se reconnecte à chaque fois à `jsonplaceholder`, et ré-exécute la requête. Il prendra donc à chaque exécution un temps constant.

Q2. Modifiez ce programme pour qu'il affiche aussi son temps d'exécution en millisecondes

Q3. Téléchargez la librairie Jedis qui permet d'interagir avec Redis en Java

Q4. Ecrire maintenant un programme java `TP55RestAvecCache.java` qui reprend le précédent mais qui, avant d'exécuter la requête, regarde dans REDIS si la réponse y est déjà. Dans le cas contraire, la requête est alors effectuée et la réponse est rangée dans REDIS (une clé par utilisateur).

Si vous lancez plusieurs fois ce programme sur le même utilisateur, il se connecte plus qu'une seule fois (la première). Par ailleurs les données sont maintenant persistantes.

Q5. Comparez les temps d'exécution.

Q6. Vérifiez dans la base REDIS que les données récupérées sont bien persistantes.