Base De Données NoSQL

Réalisé par : Meziane Sarah
INFOA3-25/26

Guide d’installation et d’utilisation de Redis avec Docker

1. Pré-requis

Avant toute manipulation, il est nécessaire d’avoir **Docker installé et opérationnel** sur la machine hôte.
Docker permettra d’exécuter Redis dans un environnement isolé, sans installation directe sur le système.

---

2. Installation de Redis via Docker

2.1. Création du conteneur Redis

Redis peut être exécuté directement à partir de son image officielle Docker. La commande suivante permet de créer et lancer un conteneur Redis en arrière-plan :

docker run -d --name redis -p 6379:6379 redis:<version>

- --name redis : nom attribué au conteneur
- -p 6379:6379 : expose le port Redis standard
- redis:<version> : image Redis (ex. redis:latest)

Il est ensuite possible de vérifier que le conteneur est bien actif en exécutant :

docker ps

Un conteneur **redis** doit apparaître dans la liste des conteneurs en cours d’exécution.

2.2. Accès au client Redis à l’intérieur du conteneur

Pour exécuter des commandes Redis, il est nécessaire d'accéder au client interactif redis-cli fourni dans l’image.
Cela se fait via :

docker exec -it redis redis-cli

Cette commande ouvre une session interactive à l'intérieur du conteneur et permet d'utiliser Redis normalement.

Une fois connecté, il est possible d'exécuter toutes les commandes Redis, comme :

PING
SET key value
GET key

---

3. Présentation générale de Redis

3.1. Qu'est-ce que Redis ?

Redis est une **base de données NoSQL orientée clé-valeur** fonctionnant principalement **en mémoire RAM**, ce qui lui permet d'offrir des performances extrêmement élevées.
Son modèle simple mais flexible permet de stocker et manipuler des données très rapidement.

Redis est couramment utilisé comme :

- système de cache,
- gestionnaire de sessions,
- compteur en temps réel,
- file d'attente,
- ou encore moteur de messagerie (pub/sub).

3.2. Les opérations principales

Redis repose sur un modèle **clé → valeur**, ce qui permet d'exécuter très simplement des opérations CRUD :

- **C**reate : création d'une nouvelle clé

  SET clé valeur

- **R**ead : lecture d'une clé

GET clé

- **U**pdate : modification d'une valeur existante
  (Redis écrase automatiquement l'ancienne valeur lors d'un SET)

- **D**elete : suppression d'une clé

  DEL clé

Redis prend également en charge plusieurs structures de données avancées : listes, ensembles, ensembles ordonnés, hachages, etc.

Redis est largement employé dans des scénarios où la **vitesse** et la **faible latence** sont essentielles :

Compteurs en temps réel
Exemple : compteur de visites d'un site

INCR vues

Stockage temporaire avec expiration (sessions, jetons, OTP)

SET sessionID valeur EX 3600

4. Concepts:

4.1. Les lists (listes ordonnées)

Ajouter un élément en tête ou en queue :
LPUSH liste valeur
RPUSH liste valeur

Extraire un élément :
LPOP liste ; retirer en tête
RPOP liste ; retirer en queue

Consulter la liste :
LRANGE liste 0 -1 ; affiche toute la liste

4.2. Les sets (ensembles sans doublons)

Ajouter un élément :
SADD ensemble valeur

Voir tous les éléments :
SMEMBERS ensemble

Vérifier l'existence :
SISMEMBER ensemble valeur

Supprimer un élément :
SREM ensemble valeur

Opérations d'ensemble (niveau avancé)

Redis permet des opérations mathématiques sur des ensembles :

SUNION set1 set2 ; union
SINTER set1 set2 ; intersection
SDIFF set1 set2 ; différence

4.3. Le système Pub/Sub (messagerie instantanée)

Redis propose un système de publication/souscription léger permettant de diffuser des messages en temps réel.

Souscrire à un canal :

SUBSCRIBE canal

Publier un message sur un canal :

PUBLISH canal "message"

Très utilisé pour :

- des notifications internes,
- des mises à jour en temps réel,
- des architectures microservices.