Table des matières

Alexis Eveno, Mattéo Faye, Mylan Seux

1. MongoDB

Afin d’implémenter MongoDB, voici un BSON type :


```
{
  "competences": [
    {
      "_id": 1,
      "nom": "Attaque Puissante",
      "niveau": 5,
      "cooldown": 10,
      "effets": {
        "degats": 150,
        "zone": "proche",
        "type": "physique"
      },
      "cout": {
        "energie": 20,
        "mana": 0
      },
      "description": "Une attaque qui inflige de lourds dégâts aux ennemis proches."
    },
    {
      "_id": 2,
      "nom": "Bouclier Magique",
      "niveau": 3,
      "cooldown": 15,
      "effets": {
        "protection": 100,
        "duree": 8,
        "type": "magique"
      },
      "cout": {
        "energie": 0,
        "mana": 25
      },
      "description": "Un bouclier temporaire qui absorbe les dégâts magiques."
    }
}
```



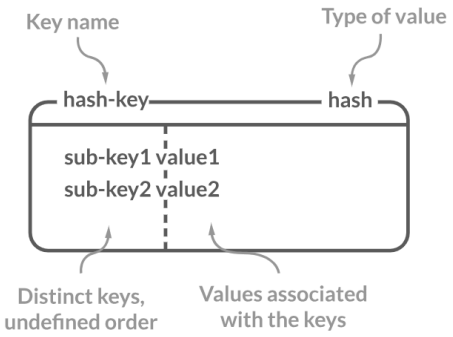
```
{
  "joueurs": [
    {
      "player_id": 1,
      "pseudo": "MaDaronne Uchiwa",
      "classe": "Guerrier",
      "niveau": 35,
```

```
      "competences": [
        { "_id": 1, "nom": "Coup Puissant" },
        { "_id": 7, "nom": "Cri de Guerre" }
      ],
      "inventaire": [
        { "_id": 1, "nom": "Épée du Brave", "quantite": 1 },
        { "_id": 2, "nom": "Bouclier de Fer", "quantite": 1 },
        { "_id": 14, "nom": "Anneau de Régénération", "quantite": 2 }
      ],
      "date_connexion": "2024-11-01"
    },
    {
      "player_id": 2,
      "pseudo": "MysticMage",
      "classe": "Mage",
      "niveau": 40,
      "competences": [
        { "_id": 4, "nom": "Flèche de Glace" },
        { "_id": 13, "nom": "Tempête d'Éclairs" }
      ],
      "inventaire": [
        { "_id": 4, "nom": "Potion de Mana", "quantite": 5 },
        { "_id": 13, "nom": "Bâton de Mage", "quantite": 1 },
        { "_id": 20, "nom": "Cape de Résistance au Feu", "quantite": 1 }
      ],
      "date_connexion": "2024-10-30"
    }
}
```

2. Partie Redis

Clé	Valeur	TTL
attack:<player_id>:<ennemy_id>	{dégâts": 150, "id_attaque": "attack_5678", "timestamp": "2024-11-06T14:25:30Z"}	15
move:<player_id>	{"coordonnees": { "x": 256, "y": 512 }, "timestamp": "2024-11-06T14:25:30Z"}	30
leaderboard:<statistic_type>	ZADD leaderboard:xp 1500 player123	24 heures

Les profils des joueurs sont stockés en utilisant des **hashes**[1], une structure clé-valeur optimisée pour contenir plusieurs champs.



Redis a été utilisé pour gérer les profils des joueurs via des hashes (clé `profile:<player_id>`) contenant des données comme le pseudo, la classe, et les compétences en JSON. Les interactions en temps réel, comme les combats (`combat:<player_id>:<enemy_id>`) et déplacements (`move:<player_id>`), utilisent des clés avec TTL[2] pour une expiration automatique. Les scores sont maintenus dans des sorted sets pour le classement des joueurs. Cette structure assure performance, simplicité et gestion dynamique des données.

---

[1] *La structure de Redis Hashes stocke un ensemble de paires champ-valeur. Il offre la possibilité d'ajouter, d'extraire ou de supprimer des éléments individuels et d'extraire l'intégralité du hachage ou encore d'utiliser un ou plusieurs champs du hachage, tel un compteur.*

[2] *Vous pouvez définir une date d'expiration pour les clés que vous synchronisez avec Redis en définissant une valeur TTL. Cette valeur correspond à la durée en secondes pendant laquelle les clés existeront dans Redis avant d'être automatiquement supprimées (Hightouch).*

3. Partie Cassandra:

Voici une modélisation de la base de données Cassandra.

player_id	attack	defense	xp	ts
1	80	30	150	2024-01-01 10:00:00
2		60	90	2024-01-02 11:30:00
3	60	60	120	2024-01-03 14:20:00
1			45	1728108199

Nous avons décidé de déployer notre base de données Cassandra dans un cluster composé de plusieurs nœuds pour assurer à la fois la **scalabilité** et **la tolérance aux pannes**. Dans Cassandra, un **keyspace** est un conteneur logique pour nos tables. Ce keyspace est déployé sur tous les nœuds du cluster, et c'est au niveau du keyspace que la stratégie de réplication est définie

Lorsque nous créons un keyspace, nous définissons la stratégie de réplication et le **facteur de réplication** (indiqué par `replication_factor`). Ce facteur détermine le nombre de copies de chaque donnée qui seront maintenues dans le cluster. Par exemple, avec le paramètre `replication_factor` **de 3**, les données seront répliquées sur trois nœuds distincts du cluster. Cela garantit une haute disponibilité et permet au système de rester opérationnel même en cas de panne de l'un des nœuds. Ce paramètre est modifiable à tout instant.

```
cassandra-node1:
  image: cassandra:latest
  container_name: cassandra-node1
  ports:
    - "9042:9042"
  networks:
    - backend
  environment:
    - CASSANDRA_CLUSTER_NAME=TestCluster
    - CASSANDRA_LISTEN_ADDRESS=cassandra-node1
    - CASSANDRA_RPC_ADDRESS=0.0.0.0
    - CASSANDRA_SEEDS=cassandra-node1
    - CASSANDRA_DC=DC1
    - CASSANDRA_RACK=Rack1
  volumes:
    - cassandra-data1:/var/lib/cassandra

cassandra-node2: <6 keys>

cassandra-node3: <6 keys>
```

CREATE KEYSPACE IF NOT EXISTS tp_keyspace WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 3};