

Atelier Redis – Documentation Complète

Plan de déroulement

1. Installation de Redis
2. Mise en place d'une répllication Redis
3. Développement d'une application web avec cache Redis
4. Vérifications et démonstrations

Partie 1 – Installation de Redis

Installation de Redis sur Ubuntu avec la commande suivante :

```
sudo apt install redis-server
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3-venv is already the newest version (3.12.3-0ubuntu2).
python3-pip is already the newest version (24.0+dfsg-1ubuntu1.1).
0 upgraded, 0 newly installed, 0 to remove and 49 not upgraded.
```

Lancement du serveur Redis maître :

```
redis-server --port 6379 --requirepass "motdepasse"
```

```
503:C 23 Jun 2025 09:18:34.874 # o00o000o000o Redis is starting o00o000o000o
503:C 23 Jun 2025 09:18:34.874 # Redis version=7.0.15, bits=64, commit=00000000, modified=0, pid=503, just started
503:C 23 Jun 2025 09:18:34.874 # Configuration loaded
503:M 23 Jun 2025 09:18:34.874 * monotonic clock: POSIX clock_gettime

Redis 7.0.15 (00000000/0) 64 bit

Running in standalone mode
Port: 6379
PID: 503

https://redis.io

503:M 23 Jun 2025 09:18:34.875 # Server initialized
503:M 23 Jun 2025 09:18:34.875 # WARNING Memory overcommit must be enabled! Without it, a background save or replication
may fail under low memory condition. Being disabled, it can also cause failures without low memory condition, see h
ttps://github.com/jemalloc/jemalloc/issues/1328. To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf an
d then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to take effect.
503:M 23 Jun 2025 09:18:34.876 * Ready to accept connections
```

Partie 2 – Architecture distribuée (réplication)

Lancement d'un serveur esclave Redis :

```
redis-server --port 6380 --slaveof 127.0.0.1 6379 --masterauth "motdepasse"
```

```
530:C 23 Jun 2025 09:22:37.285 # oOoOoOoOoOoOo Redis is starting oOoOoOoOoOoOo
530:C 23 Jun 2025 09:22:37.285 # Redis version=7.0.15, bits=64, commit=00000000, modified=0, pid=530, just started
530:C 23 Jun 2025 09:22:37.285 # Configuration loaded
530:S 23 Jun 2025 09:22:37.285 * monotonic clock: POSIX clock_gettime

Redis 7.0.15 (00000000/0) 64 bit

Running in standalone mode
Port: 6380
PID: 530

https://redis.io

530:S 23 Jun 2025 09:22:37.285 # Server initialized
530:S 23 Jun 2025 09:22:37.285 # WARNING Memory overcommit must be enabled! Without it, a background save or replication may fail under low memory condition
. Being disabled, it can also cause failures without low memory condition, see https://github.com/jemalloc/jemalloc/issues/1328. To fix this issue add '
vm.overcommit_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to take effect.
530:S 23 Jun 2025 09:22:37.286 * Ready to accept connections
530:S 23 Jun 2025 09:22:37.287 * Connecting to MASTER 127.0.0.1:6379
530:S 23 Jun 2025 09:22:37.287 * MASTER <-> REPLICATION sync started
530:S 23 Jun 2025 09:22:37.287 * Non blocking connect for SYNC fired the event.
530:S 23 Jun 2025 09:22:37.291 * Master replied to PING, replication can continue...
530:S 23 Jun 2025 09:22:37.291 * Partial resynchronization not possible (no cached master)
530:S 23 Jun 2025 09:22:42.948 * Full resync from master: 893cblae429c7760de3542c708ae3c7f3e3106d7:0
530:S 23 Jun 2025 09:22:42.949 * MASTER <-> REPLICATION sync: receiving streamed RDB from master with EOF to disk
530:S 23 Jun 2025 09:22:42.949 * MASTER <-> REPLICATION sync: Flushing old data
530:S 23 Jun 2025 09:22:42.949 * MASTER <-> REPLICATION sync: Loading DB in memory
530:S 23 Jun 2025 09:22:42.958 * Loading RDB produced by version 7.0.15
530:S 23 Jun 2025 09:22:42.958 * RDB age 0 seconds
530:S 23 Jun 2025 09:22:42.958 * RDB memory usage when created 0.91 Mb
530:S 23 Jun 2025 09:22:42.958 * Done loading RDB, keys loaded: 0, keys expired: 0.
```

Vérification de la réplication :

```
redis-cli -p 6380 info replication
```

[illegible]

Partie 3 – Application Web avec Cache Redis

Technologies utilisées : Python, Flask, redis-py.

Création d'une route /data/<key> avec une logique de cache :

```

GNU nano 7.2.1                                app.py *
from flask import Flask
import redis
import time

app = Flask(__name__)
cache = redis.Redis(host='localhost', port=6379, password='ton_mot_de_passe', decode_responses=True)

@app.route('/data/<key>')
def get_data(key):
    value = cache.get(key)
    if value:
        return f"[CACHE] {value}"
    else:
        time.sleep(2)
        value = f"data_pour_{key}"
        cache.setex(key, 60, value)
        return f"[SLOW] {value}"

if __name__ == '__main__':
    app.run(debug=True)
from flask import Flask, jsonify
import redis
import time
import logging

logging.basicConfig(level=logging.INFO)

app = Flask(__name__)

try:
    cache = redis.Redis(host='localhost', port=6379, password='ton_mot_de_passe', decode_responses=True)
    cache.ping()
    logging.info("Connexion à Redis réussie.")
except redis.RedisError as e:

```

```

except redis.RedisError as e:
    logging.error("Erreur de connexion à Redis.", exc_info=e)
    cache = None

def get_from_cache(key):
    if not cache:
        return None
    try:
        return cache.get(key)
    except redis.RedisError as e:
        logging.error("Erreur lors de la lecture du cache.", exc_info=e)
        return None

def set_to_cache(key, value, ttl=60):
    if not cache:
        return
    try:
        cache.setex(key, ttl, value)
    except redis.RedisError as e:
        logging.error("Erreur lors de l'écriture dans le cache.", exc_info=e)

def simulate_slow_database(key):
    time.sleep(2)
    fake_data = {
        "foo": "valeur_de_foo",
        "bar": "valeur_de_bar"
    }
    return fake_data.get(key, f"data_pour_{key}")

@app.route('/data/<key>')
def get_data(key):
    start_time = time.time()

    value = get_from_cache(key)
    if value:
        elapsed = time.time() - start_time
        logging.info(f"Cache hit pour la clé: {key}")

```

Lancement de l'application Flask.

Python app.py

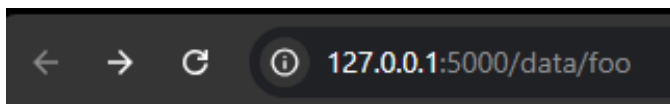
```

* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 282-458-232

```

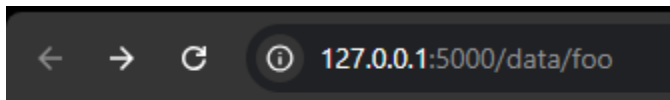
Partie 4 – Vérifications

Première requête (donnée absente du cache, réponse lente) :



[SLOW] data_pour_foo

Deuxième requête (donnée présente dans le cache, réponse rapide) :



[CACHE] data_pour_foo