# **Aula Prática de Laboratório – Redis**

## **Objetivos da Aula**

· Compreender os conceitos fundamentais do Redis

· Instalar e configurar o Redis localmente

· Executar operações básicas e avançadas

· Implementar casos de uso práticos

· Explorar estruturas de dados do Redis

## **Parte 1: Introdução e Instalação**

### **O que é Redis?**

Redis (Remote Dictionary Server) é um banco de dados NoSQL em memória que funciona como estrutura de dados. É conhecido por sua alta performance e versatilidade.

### **Principais características:**

· Armazenamento em memória (RAM)

· Estruturas de dados avançadas

· Persistência opcional

· Replicação e clustering

· Pub/Sub messaging

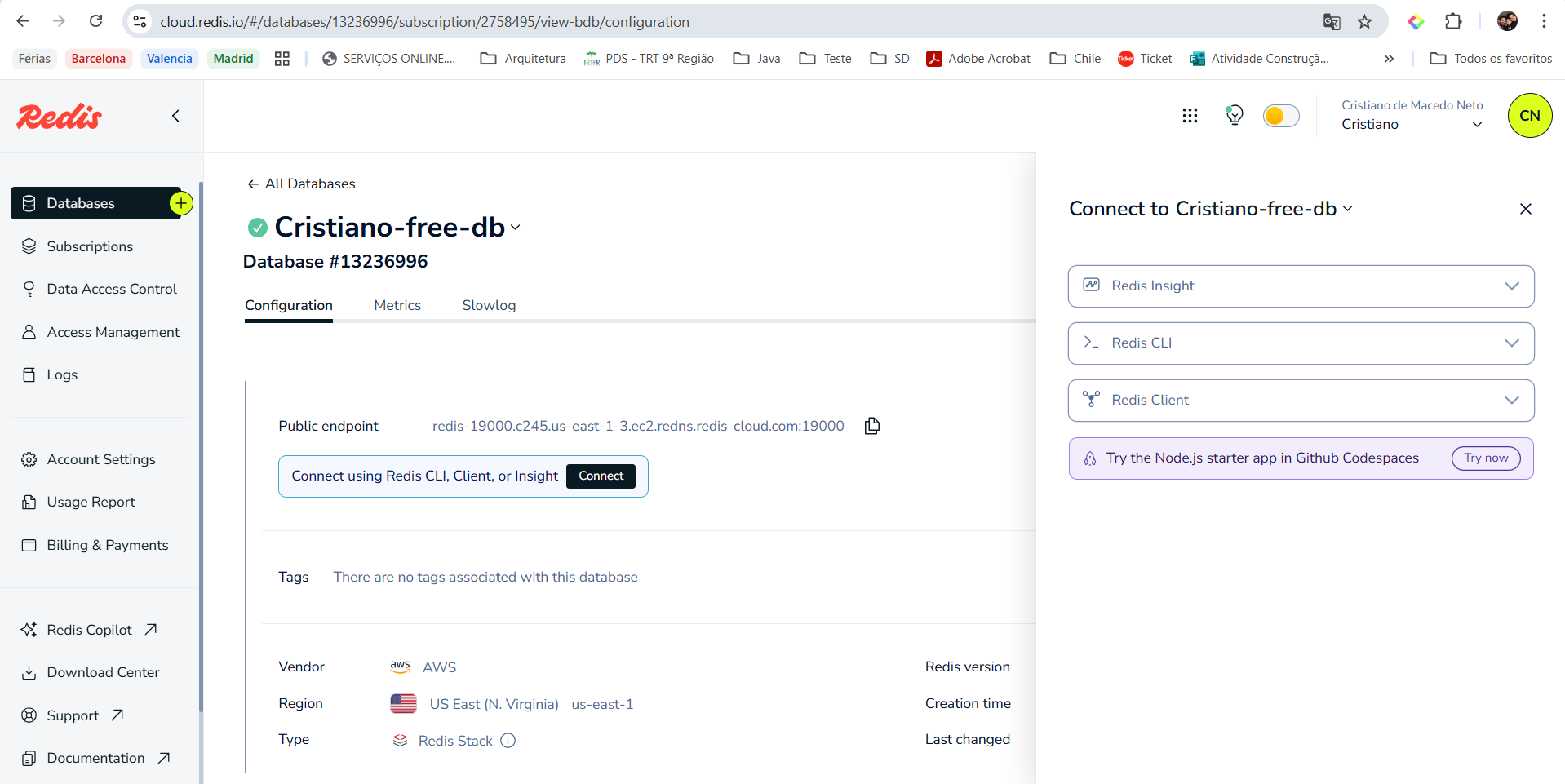
### 

### 

### 

### **Instalação do Redis ou Cloud**

Sugestão cloud: <https://cloud.redis.io/>



#### **No Ubuntu/Debian:**

| sudo apt update sudo apt install redis-server |
| --- |

#### **No macOS:**

| brew install redis |
| --- |

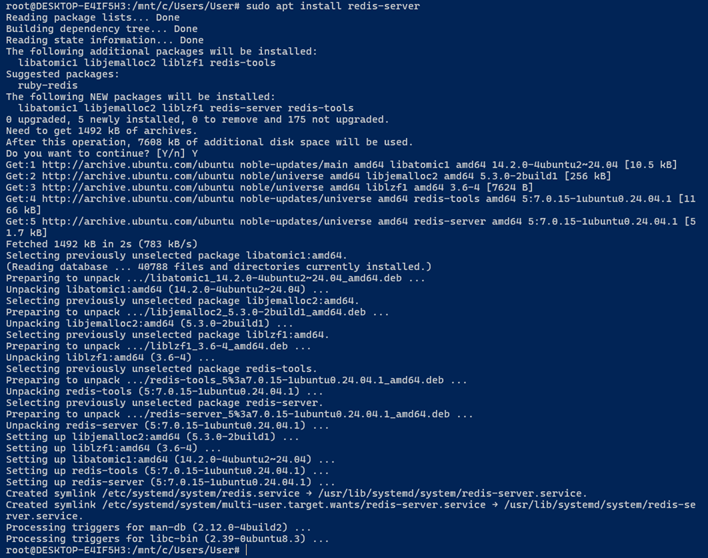
#### **No Windows:**

##### **# Via WSL**

| wsl sudo apt update sudo apt install redis-server |
| --- |

##### **# Redis Stack no Docker**

| docker run -d --name redis-stack -p 6379:6379 redis/redis-stack:latest |
| --- |



#### 

#### **Verificando a Instalação**

| redis-server -version |
| --- |



| redis-cli ping |
| --- |



## Parte 2: Comandos Básicos

### Iniciando o Redis

| # Terminal 1 - Servidor redis-server |
| --- |

| Caso ocorra o erro apresentado na figura a seguir, é porque não foi feito a configuração do server. |
| --- |

| **Configurações Importantes**  *# Arquivo redis.conf*  maxmemory 256mb  maxmemory-policy allkeys-lru  timeout 300 |
| --- |

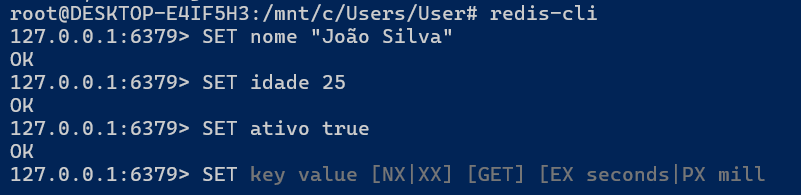
| # Terminal 2 - Cliente redis-cli |
| --- |

### **Operações Fundamentais**

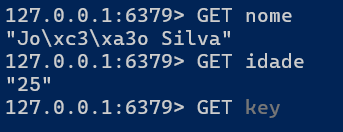
#### **Strings (Chave-Valor Básico)**

redis

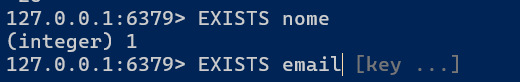
| # Definir uma chave SET nome "João Silva" SET idade 25 SET ativo true |
| --- |



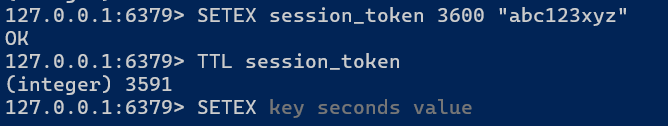
| # Recuperar valores GET nome GET idade |
| --- |



| # Verificar se chave existe EXISTS nome EXISTS email |
| --- |



| # Definir com expiração (TTL) SETEX session\_token 3600 "abc123xyz" TTL session\_token |
| --- |



| # Incrementar/Decrementar SET contador 0 INCR contador INCRBY contador 5 DECR contador |
| --- |

#### 

## Exercício Prático 1

| # Crie um sistema de contador de visitas SET visitas\_site 0 INCR visitas\_site INCR visitas\_site GET visitas\_site |
| --- |

| # Crie uma sessão de usuário que expira em 30 segundos SETEX user\_session\_123 30 "usuario\_logado" TTL user\_session\_123 # Aguarde alguns segundos e verifique novamente TTL user\_session\_123 |
| --- |

## Parte 3: Estruturas de Dados Avançadas

### Listas

| # Adicionar elementos LPUSH tarefas "Estudar Redis" LPUSH tarefas "Fazer exercícios" RPUSH tarefas "Revisar conteúdo" |
| --- |

| # Visualizar lista LRANGE tarefas 0 -1 |
| --- |

| # Remover elementos LPOP tarefas RPOP tarefas |
| --- |

| # Tamanho da lista LLEN tarefas |
| --- |

### **Sets (Conjuntos)**

| # Adicionar elementos únicos SADD tecnologias "Python" SADD tecnologias "JavaScript" SADD tecnologias "Redis" SADD tecnologias "Python" # Não será duplicado |
| --- |

| # Visualizar set SMEMBERS tecnologias |
| --- |

| # Verificar se elemento existe SISMEMBER tecnologias "Python" |
| --- |

| # Operações entre sets SADD frontend "JavaScript" "HTML" "CSS" SADD backend "Python" "Node.js" "Redis" SINTER frontend backend # Interseção SUNION frontend backend # União |
| --- |

### **Hashes (Objetos)**

| # Criar um usuário HSET usuario:1 nome "Maria Santos" HSET usuario:1 email "maria@email.com" HSET usuario:1 idade 28 |
| --- |

| # Definir múltiplos campos HMSET usuario:2 nome "Pedro Lima" email "pedro@email.com" idade 32 |
| --- |

| # Recuperar dados HGET usuario:1 nome HGETALL usuario:1 |
| --- |

| # Incrementar campo numérico HINCRBY usuario:1 idade 1 |
| --- |

### **Sorted Sets (Conjuntos Ordenados)**

| # Ranking de pontuação ZADD ranking 100 "jogador1" ZADD ranking 250 "jogador2" ZADD ranking 180 "jogador3" |
| --- |

| # Ver ranking ordenado ZRANGE ranking 0 -1 WITHSCORES ZREVRANGE ranking 0 -1 WITHSCORES # Ordem decrescente |
| --- |

| # Buscar por score ZRANGEBYSCORE ranking 150 300 |
| --- |

### **Exercício Prático 2**

| # Sistema de E-commerce Simples  # 1. Produto (Hash) HMSET produto:1 nome "Notebook Dell" preco 2500.00 categoria "informatica" estoque 10  # 2. Carrinho de compras (Lista) LPUSH carrinho:user123 "produto:1" LPUSH carrinho:user123 "produto:2"  # 3. Categorias (Set) SADD categoria:informatica "produto:1" "produto:3" SADD categoria:eletronicos "produto:2" "produto:4"  # 4. Ranking de produtos mais vendidos (Sorted Set) ZADD produtos\_vendidos 15 "produto:1" ZADD produtos\_vendidos 8 "produto:2" ZADD produtos\_vendidos 23 "produto:3" |
| --- |

## **Parte 4: Casos de Uso Práticos**

### **Cache de Aplicação**

| # Cache de consulta ao banco de dados SET cache:usuario:123 '{"nome":"João","email":"joao@email.com"}' EX 300 |
| --- |

| # Cache de página web SET cache:pagina:/produtos '<!DOCTYPE html>...' EX 600 |
| --- |

### **Sistema de Sessões**

| # Criar sessão HMSET sessao:abc123 usuario\_id 456 ip "192.168.1.1" login\_time "2024-01-15 10:30:00" EXPIRE sessao:abc123 1800 # 30 minutos |
| --- |

| # Verificar sessão ativa EXISTS sessao:abc123 HGETALL sessao:abc123 |
| --- |

### **Pub/Sub (Mensageria)**

| # Terminal 1 - Subscriber SUBSCRIBE noticias SUBSCRIBE canal\_geral |
| --- |

| # Terminal 2 - Publisher PUBLISH noticias "Nova versão do Redis lançada!" PUBLISH canal\_geral "Manutenção programada às 22h" |
| --- |

### **Rate Limiting**

| # Limitar 5 requisições por minuto por IP SET rate\_limit:192.168.1.1 1 EX 60 INCR rate\_limit:192.168.1.1 # Se > 5, bloquear usuário |
| --- |

## **Parte 5: Monitoramento e Otimização**

### **Comandos de Monitoramento**

| # Informações do servidor INFO  # Estatísticas específicas INFO memory INFO stats INFO replication  # Monitorar comandos em tempo real MONITOR  # Chaves que estão ocupando mais memória MEMORY USAGE usuario:1 |
| --- |

### **Comandos de Manutenção**

| # Listar todas as chaves (cuidado em produção!) KEYS \*  # Buscar chaves por padrão KEYS usuario:\* KEYS cache:\*  # Limpar database atual FLUSHDB  # Limpar todas as databases FLUSHALL  # Persistir dados no disco SAVE BGSAVE |
| --- |

## **Parte 6: Exercício Final**

### **Sistema de Blog com Redis**

| # 1. Estrutura de Posts HMSET post:1 titulo "Introdução ao Redis" autor "João Dev" conteudo "Redis é um banco..." views 0 likes 0 HMSET post:2 titulo "NoSQL vs SQL" autor "Maria Tech" conteudo "Comparação entre..." views 0 likes 0  # 2. Índice de posts por autor SADD posts:autor:joao "post:1" "post:3" SADD posts:autor:maria "post:2" "post:4"  # 3. Tags dos posts SADD tags:redis "post:1" "post:5" SADD tags:nosql "post:1" "post:2"  # 4. Ranking de posts mais visualizados ZADD posts\_populares 0 "post:1" ZADD posts\_populares 0 "post:2"  # 5. Comentários (Lista) LPUSH comentarios:post:1 '{"autor":"leitor1","texto":"Ótimo post!","data":"2024-01-15"}'  # 6. Sistema de curtidas (Set) SADD likes:post:1 "user123" "user456"  # 7. Cache de página inicial SET cache:home:posts '[{"id":1,"titulo":"Intro Redis"},{"id":2,"titulo":"NoSQL vs SQL"}]' EX 300  # Simular visualização de post HINCRBY post:1 views 1 ZINCRBY posts\_populares 1 "post:1" |
| --- |

### **Implementação em Python**

| import redis import json  # Conectar ao Redis r = redis.Redis([conforme exemplo https://cloud.redis.io/])  class BlogRedis:  def criar\_post(self, post\_id, titulo, autor, conteudo):  post = {  "titulo": titulo,  "autor": autor,  "conteudo": conteudo,  "views": 0,  "likes": 0  }    r.hset(f"post:{post\_id}", mapping=post)  r.sadd(f"posts:autor:{autor}", f"post:{post\_id}")    def visualizar\_post(self, post\_id):  r.hincrby(f"post:{post\_id}", "views", 1)  r.zincrby("posts\_populares", 1, f"post:{post\_id}")  return r.hgetall(f"post:{post\_id}")    def curtir\_post(self, post\_id, user\_id):  if r.sadd(f"likes:post:{post\_id}", user\_id):  r.hincrby(f"post:{post\_id}", "likes", 1)  return True  return False  # Exemplo de uso blog = BlogRedis() blog.criar\_post(1, "Meu primeiro post", "joao", "Conteúdo do post...") print(blog.visualizar\_post(1)) |
| --- |

## 