# Hello!

I am Esmaeil Kazemi
I'm interested in learning how are you?
You can find me at @eskazemi











### NOSQL

VS

SQL



eskazemi



**Redis stands for Remote Dictionary Server** 









- 1- Features
- 2- application
- 3- data type
- 4- Message Queue
- **5-** Transactions
- 6- Pipelining
- 7- Lua Scripts

- 8- Persistence
- 9- Benchmarks
- 10- configuration
- 11- ACLs
- 12- Redis Cluster
- 13- Redis vs Memcached
- 14- Redis vs Hazelcast
- 15- Redis vs RDBMS











#### in-memory وجود دارد تعداد زیادی دیتابیس های

- **Redis**
- **►** Memcached
  - **CouchDB**
  - **→** Hazelcast





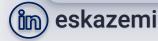




از آنجایی که هر دوی این ها برای موارد مشترکی مانند cache می توانند استفاده شود همواره مقایسه ای بین آنها انجام می شود و در پارامتر های زیادی از جنبه های مختلف می توان این دو رو بررسی کرد اما با توجه به تغییرات Redis در ورژن های بالاتر تقریبا انتخاب Redis امری بدیهی هست.

 ✓ data type نوع داده هایی هست که نگه می دارند. memcached فقط توانایی ذخیره نوع ساده رشته را به عنوان مقدار دارد در صورتی که Redis دارای ۵ نوع Data Type اصلی هست از آنجایی که Redis دارای Data Type های بیشتری هست استفاده از آن می تواند باعث سهولت در نگه داری ساختار های داده ای پیچیده تر و کاربرد های متفاوت تری بشود.

Redis : Memory usage ✓ عملکرد بهتری داد به خصوص در بحث آزاد سازی سریع فضای RAM استفاده شده بعد از flush کردن داده ها.

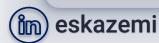




✓ نگهداری داده: توی Redis شما می تونین دیتاتون رو persistence کنین، یعنی اجازه بدین دیتاتون ریخته بشه توی دیسک و بعدش هر زمانی که بهش نیازش داشتین از دیسک دوباره انتقال داده میشه به مموری. در حالی که memcached این فیچر رو ساپورت نمی کنه و اگه سرورتون پایین بیاد یا ری استارت بشه دیتای ذخیره شدتون نابود میشه. البته در صورتی که Memcached نیاز به ابزار های استارت بشه دیتای ذخیره شدتون نابود میشه. البته در صورتی که third-party source برای dumpگرفتن داده ها دارد. البته ناگفته نماند که در بعضی موارد هم مانند سرعت خواندن و نوشتن و یا Scaling این دو بسیار به هم نزدیک هستند و برتری خاصی حس نمی شود.

√ کلاسترینگ: Redis به طور داخلی دارای فیچر رپلیکیشن و پارتیشنینگ هستش و همین فیچر برای hash کردن مناسبش می کنه. توی memcashed ما این فیچر ها رو نداریم در memcashed ولی چون کردن مناسبه، یعنی می memcached خیلی مناسبه، یعنی می تونیم کلید هامون رو توی چندین memcached ذخیره کنیم.

✓ همزمانی: memcached از ساختار multithreade استفاده می کنه که بهش اجازه میده چندین کانکشن رو
 به طور همزمان با استفاده از ترد های جداگانه هندل کنه. در حالی که Redis برای هندل کردن درخواست ها
 از یک ترد استفاده می کنه که این موضوع در memcached می تونه باعث بالا رفتن پرفورمنس با استفاده از
 همزمانی بشه.



66

#### جمع بندي

اگه تنها هدفتون کش کردن دیتاس، اگه به یه سیستم کش ساده علاقه دارین، اگه نمی خواین از دیتا تایپ های پیشرفته ای استفاده کنین، اگه دیتای زیادی دارین و می خواین با استفاده از چندین core از پرفورمنس Multithread استفاده کنین، اگه می خواین سیستم کش کردنتون رو همون جوری که رشد می کنین به صورت horizontally گسترش بدین، در همه این موارد از memcached استفاده کنین.













#### **Architecture**

**Redis** is an in-memory data store that supports data replication and persistence for durability.

**Memcashed** is a high-performance, distributed memory object caching system

Replication

Data replication is not supported.

Data replication is supported.

concurrency

It is a single-threaded architecture.

It is a multi-threaded architecture.









**Advanced Data Structures** (e.g., lists, sets, hashes)

Data Model

uses a simple key-value store, which is great for caching results

**Transactions** 

Transactions is not supported.

Transactions is supported.

**Data eviction** 

LRU (Least Recently Used).

LRU, LFU (Least Frequently Used), Random, TTL (Time-to-Live)

Persistence

Optional persistence (e.g., snapshots, eskazemi append-only files)

No built-in persistence





#### **Scalability**

offers primary-replica replication, allowing it to scale reads. With Redis Cluster, it can also scale writes.

can scale horizontally, but it does not support replication or partitioning out of the box

#### Performance

Faster than Redis for small value sizes, but slower for larger values Generally slower than Memcached for small value sizes, but faster for larger values

#### **Popular Use Cases**

Caching, session management, message broker, realtime data processing, job queue

is used by platforms like YouTube, Reddit, and Facebook to speed up dynamic web applications by reducing database load. 13

## Thanks!

Any questions?

You can find me at:

- @eskazemi
- m.esmaeilkazemi@gmail.com







