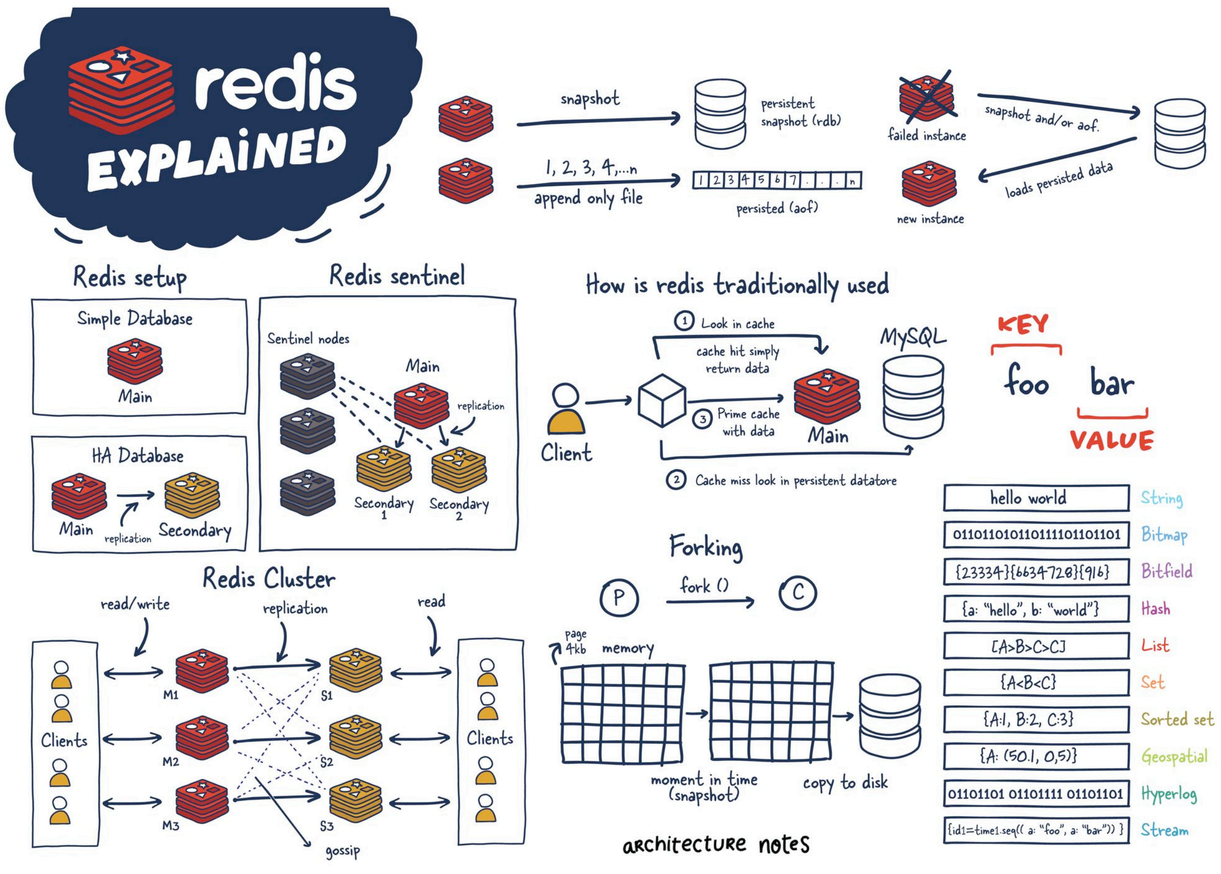
Redis Essentials

↗ courses	Redis Essential Training



What is Redis?

An key-value, in-memory, NoSQL database

Redis Installation (MacOS)

1. `brew install redis`
2. `redis-server`
3. (new terminal tab) `redis-cli`

Redis Features

1. Quick interactions with ephemeral data
2. Used to store simple information for quick access and improved performance of the overall system
3. Very rarely used as the main database, but has persistence feature

Redis Use Cases

1. Caching

   Redis is commonly used as a caching layer to store frequently accessed data, reducing database load and improving response times. It supports caching strategies like cache-aside and write-back, with features like time-to-live (TTL) for automatic eviction of stale data

2. Session Management

   Redis stores user session data efficiently, making it ideal for stateless applications. It ensures seamless user experiences by enabling session persistence, expiration, and high availability through replication

3. Real-time Analytics

   Redis powers real-time analytics by processing and aggregating data at high speed. This is useful for tracking website visits, user interactions, or product views in real time

4. Message Queues

   Redis implements message queues using lists and pub/sub mechanisms for background processing tasks such as sending emails or handling user requests. It ensures reliable job execution without delays

5. Pub/sub for notifications

   The publish/subscribe model in Redis enables real-time notifications and messaging systems, such as chat applications or live updates in collaborative platforms

6. Leaderboards

   Using sorted sets (ZSET), Redis efficiently creates leaderboards for gaming or fitness apps, providing real-time rankings based on scores or other metrics

7. API Rate Limiting

   Redis helps implement rate limiting to prevent API abuse by tracking the frequency of requests and enforcing limits in real time

8. Geospatial Applications

   Redis supports geospatial indexing to store and query location-based data, making it suitable for applications like delivery tracking or finding nearby services

9. Search and Query

   With modules like RedisSearch, Redis enables full-text search, secondary indexing, and complex queries for applications requiring fast search capabilities

10. Internet of Things (IoT)

    Redis is used in IoT systems for real-time data ingestion, processing, and analytics from sensors and devices due to its low latency and scalability

11. Microservices Coordination

    n microservices architectures, Redis facilitates inter-service communication, service discovery, and synchronization by acting as a shared data layer

Redis Data Structures

1. **Strings**

   Most basic data type
2. **Lists**

   Array of items
3. **Sets**

   Unique list items
4. **Sorted Sets**

   Sets that automatically sort by a specific value
5. **Hashes**

   Maps of key-value pairs
6. **Streams**

   Allow to define data that can be distributed to different clients
7. Bitmaps
8. HyperLogLogs
9. Geospatial Indexes

Redis Commands

1. SET
   a. Set a key and value pair
   b. Is atomic - can only succeed or fail but not partial save
   c. Use double quotes if the value has space
   d. Creates they key if it does not already exist
   e. E.g:

```
SET city “Cyberjaya, Malaysia”
```

2. GET
    a. Retrieve a value of a key or `nil` if the key does not exist
    b. Use double quotes if the key has spaces
    c. E.g:

    ```
    GET city //"Cyberjaya, Malaysia"
    ```

3. INCR/DECR/INCRBY/DECRBY
    a. Treats string values as numbers and is useful in counters
    b. E.g:

    ```
    SET counter 0
    GET counter //0
    INCR counter //1
    INCR counter //2
    DECR counter //1
    INCRBY counter 4 //5
    DESCRBY counter 2 //3
    GET counter //"3"
    ```

4. STRLEN
    a. Get the length of string value
    b. E.g:

    ```
    SET mykey "this is a string"
    STRLEN mykey //16
    ```

5. APPEND
    a. Append value to a string value

b. E.g:

```
SET myname “Evangelista”
APPEND myname " Grace"
GET myname //Evangelista Grace
```

6. GETRANGE
    a. Gets a portion of a string inside a key using zero-based indexes
    b. E.g:

```
GETRANGE myname 0 3 //"Evan"
```

7. HSET
    a. Set an object with variable attributes and values
    b. E.g:

```
HSET person:123 name "Grace" age "27" location "Cyberjaya"
HGET person:123 name // "Grace"
127.0.0.1:6379> HMGET person:123 name age
1) "grace"
2) "27"
```

8. HGET
    a. Get the value of an attribute of a hash object
    b. E.g:

```
HGET person:123 name // "Grace"
```

9. HMGET
    a. Get multiple attributes’ value from a hash object
    b. E.g:

```
127.0.0.1:6379> HMGET person:123 name age
1) "grace"
2) "27"
```