NDBI040: PRACTICAL CLASS 3

REDIS

Based on NDBI040 practical class materials created by Martin Svoboda; Tutor: Pavel Čontoš; November 4th 2020

(RECOMMENDED) REQUIREMENTS

- macOS / Linux command line or PuTTy / WinSCP on Windows
- TextEdit, nano, Notepad or any other simple text editor

SERVER ACCESS

CONNECT TO NOSQL SERVER

- `ssh` on macOS / Linux
- `PuTTy` on Windows

- nosql.ms.mff.cuni.cz:42222
- Login and password send by e-mail
- Change your initial password (if not yet changed) by `passwd`

TRANSFER FILES

- `scp` on macOS / Linux
- `WinSCP` on Windows

REDIS

- In-memory data structure store
- Open source
- Master-slave replication architecture
- Sharding
- High availability
- Various persistence levels

- http://redis.io/
- Developed by Redis Labs
- Implemented in C

REDIS

FUNCTIONALITY

- Standard key-value store
- Support for structured values
  - E.g. lists, sets, hashes, sorted sets
- Time-to-live
- Transactions

REAL-WORLD USERS

- Twitter, GitHub, Pinterest, StackOverflow, ...

DATA MODEL

- `Instance` → `databases` → `objects`

DATABASE

- Collection of objects
- Databases do not have names but integer identifiers

OBJECT

- Key-value pair
- Key is a string (i.e. any binary data)
- Values can be atomic (i.e. string) or structured (i.e. list, set, sorted set, hash)

DATA TYPES

STRING

- The only atomic data type
- May contain any binary data (e.g. string, integer counter, PNG image, ...)
- Maximal allowed size is 512 MB

LIST

- Ordered collection of strings
- Elements should preferably be read / written at the head / tail

DATA TYPES

SET

- Unordered collection of strings
- Duplicate values are not allowed

SORTED SET

- Ordered collection of strings
- The order is given by a score (floating number value) associated with each element (from the smallest to the greatest score)

HASH

- Associative map between string fields and string values
- Field names have to be mutually distinct

INTERFACE

CLI - BASIC MODE

- Commands are passed as standard command line arguments
  - `redis-cli PING`
  - `redis-cli -n 16 DBSIZE`
- Batch processing is possible as well
  - `cat script.txt | redis-cli`

CLI - INTERACTIVE MODE

- Users type database commands at the prompt
  - `redis-cli`

- REST (Redis Serialization Protocol)

FIRST STEPS

CHECK REDIS STATUS

- `redis-cli PING`

OPEN REDIS CLIENT

- `redis-cli`

SELECT YOUR DATABASE

- `SELECT number`
- Your database number: sent by e-mail at the beginning of semester

BASIC COMMANDS

- `HELP command`
  - Provides basic information about Redis commands
- `CLEAR`
  - Clears the terminal screen
- `FLUSHDB`
  - Deletes all the keys in the currently selected database
- `BGSAVE`
  - Saves the current dataset (asynchronously, on background)
  - I.e. stores the database snapshot to the hard drive
- `QUIT`
  - Closes the connection

STRINGS

BASIC COMMANDS

- `SET key value` inserts / replaces a given string
- `GET key` returns a given string

STRING OPERATIONS

- `STRLEN key` returns a string length
- `APPEND key value` appends a value at the end of a string
- `GETRANGE key start end` returns a substring
  - Both the boundaries are considered to be inclusive
  - Position starts at 0
  - Negative offsets for positions starting at the end
- `SETRANGE key offset value` replaces a substring
  - Binary are padded when the original string is not long enough

STRINGS

COUNTER OPERATIONS

- `INCR key` increments a value by 1
- `DECR key` decrements a value by 1
- `INCRBY key increment` increments a value by a given amount
- `DECRBY key decrement` decrements a value by a given amount

OBJECTS

OBJECT QUERYING

- `EXISTS key` determines whether a key exists
- `KEYS pattern` finds all the keys matching a pattern (*, ?, ...)
  - E.g. `KEYS *`

MODIFICATION OF OBJECTS

- `DEL key ...` removes a given object / objects
- `RENAME key newkey` changes key of a given object

TYPE INFORMATION

- `TYPE key` determines the type of a given object
  - Types: string, list, set, zset, hash

VOLATILE OBJECTS

- Keys with limited time to live
  - When a specified timeout elapses, a given object is removed
  - Works with any data type

COMMANDS

- `EXPIRE key seconds`
  - Sets a timeout for a given object, i.e. makes the object volatile
  - Can be called repeatedly to change the timeout
- `TTL key`
  - Returns the remaining time to live for a key that has a timeout
- `PERSIST key`
  - Removes the existing timeout, i.e. makes the object persistent

LISTS

INSERTION OF NEW ELEMENTS

- `LPUSH key value` adds a new element to the head
- `RPUSH key value` adds a new element to the tail
- `LINSERT key BEFORE|AFTER pivot value` inserts an element before / after another value

RETRIEVAL OF ELEMENTS

- `LPOP key` removes and returns the first element
- `RPOP key` removes and returns the last element
- `LINDEX key index` gets an element by its index
  - The first item is at position 0
  - Negative positions are allowed as well
- `LRANGE key start stop` gets a range of elements

LISTS

REMOVAL OF ELEMENTS

- `LREM key count value`
  - Removes a given number of matching elements from a list
  - Positive / negative = moving from head to tail / tail to head
  - 0 = all the items are removed

OTHER OPERATIONS

- `LLEN key` gets the length of a list

SETS

BASIC OPERATIONS

- `SADD key value ...`
  - Adds an element / elements into a set
- `SREM key value ...`
  - Removes an element / elements from a set

DATA QUERYING

- `SISMEMBER key value`
  - Determines whether a set contains a given element
- `SMEMBERS key`
  - Gets all the elements of a set

SETS

OTHER OPERATIONS

- `SCARD key` gets the number of elements in a set

SET OPERATIONS

- `SUNION key ...`
- `SINTER key ...`
- `SDIFF key ...`
  - Calculates and returns a set union / intersection / difference of two or more sets

HASHES

BASIC OPERATIONS

- `HSET key field value` sets the value of a hash field
- `HGET key field` gets the value of a hash field

BATCH ALTERNATIVES

- `HMSET key field value ... ...`
  - Sets values of multiple fields of a given hash
- `HMGET key field ... ...`
  - Gets values of multiple fields of a given hash

HASHES

FIELD RETRIEVAL OPERATIONS

- `HEXISTS key field` determines whether a field exists
- `HGETALL key` gets all the fields and values
  - Individual fields and values are interleaved
- `HKEYS key` gets all the fields in a given hash
- `HVALS key` gets all the values in a given hash

OTHER OPERATIONS

- `HDEL key field ...`
  - Removes a given field / fields from a hash
- `HLEN key` returns the number of fields in a given hash

SORTED SETS

BASIC OPERATIONS

- `ZADD key score value ... ...`
  - Inserts one element / multiple elements into a sorted set
- `ZREM key value ...`
  - Removes one element / multiple elements from a sorted set

WORKING WITH SCORE

- `ZSCORE key value`
  - Gets the score associated with a given element
- `ZINCRBY key increment value`
  - Increments the score of a given element

SORTED SETS

RETRIEVAL OF ELEMENTS

- `ZRANGE key start stop`
  - Returns all the elements within a given range based on positions
- `ZRANGEBYSCORE key min max`
  - Returns all the elements within a given range based on scores

OTHER OPERATIONS

- `ZCARD key`
  - Gets the overall number of elements
- `ZCOUNT key min max`
  - Counts all the elements within a given range based on score

REFERENCES

- Commands
  - https://redis.io/commands
- Documentation
  - https://redis.io/documentation
- Data types
  - https://redis.io/topics/data-types