ElkDB: A NoSQL key-value document store

By Malik Naik Mohammed

Outline

Intro

ElkDB is a lightweight key-value document store where the value is the valid JSON document.

This software uses **Buckets**, and **Documents** which are similar to tables and records in relational model.

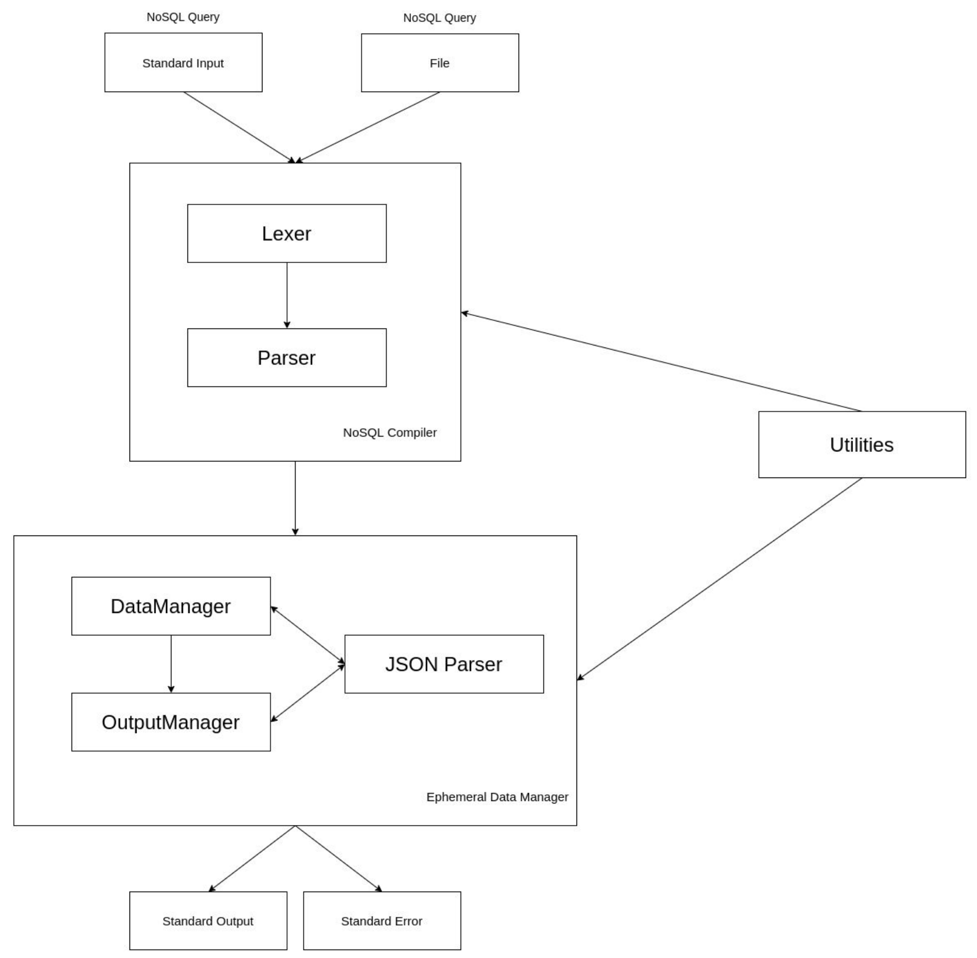
Features of ElkDB

- Small
- Fast
- Lightweight
- Ephemeral
- Key-value document store
- Supports SQL like query language

ElkDB Architecture

Core Components:

- Lexer
- Parser
- JSON Parser
- Data Manager
- Output Manager
- Utilities



Technologies

This project is implemented purely in C, and C++

Each component is written using the following libraries:

- Lexer/Scanner: Fast Lexical Analyzer (`flex`)
- Parser: GNU Bison (`bison/yacc`)
- Standard Input: GNU Readline `<readline/readline.h>` and GNU History `<readline/history.h>`
- JSON Parser: RapidJSON

Flex and Bison

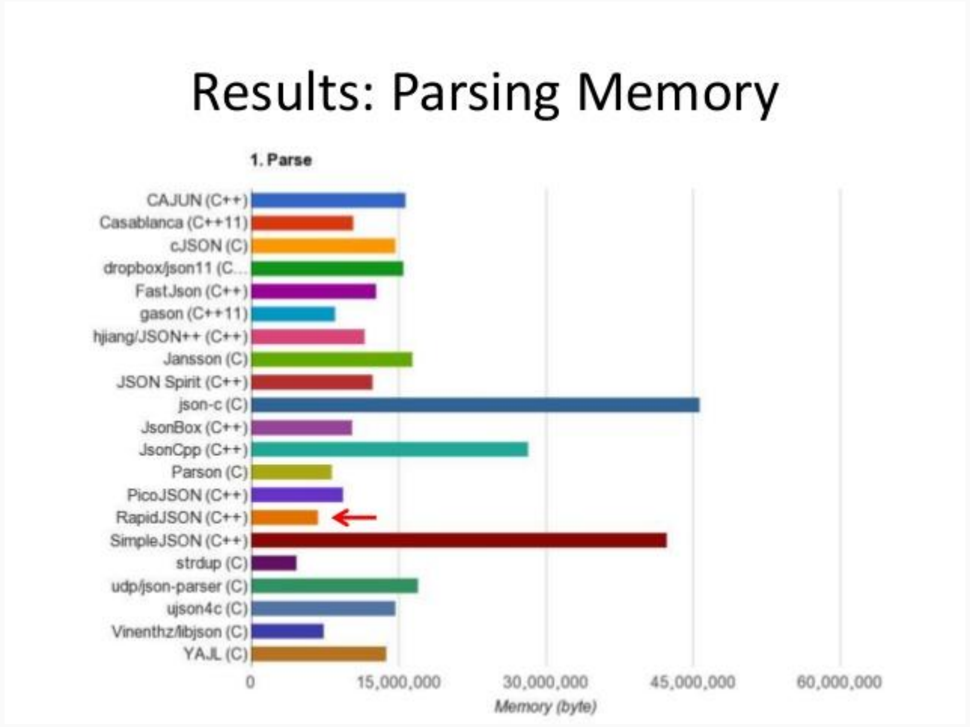
- Flex is a lexical analyzer/scanner that is used to tokenize the query.
- GNU Bison is a parser used to parse the SQL-like queries.

GNU Readline, History and RapidJSON

- GNU Readline provides the REPL (read–eval–print loop) to read the SQL-like queries in the CLI.
- GNU History provides bash/shell like history to track the queries.
- RapidJSON provides validation and pretty pring the JSON document to the CLI.

Why RapidJSON?

RapidJSON parses the JSON documents faster than other JSON Parsing libraries.

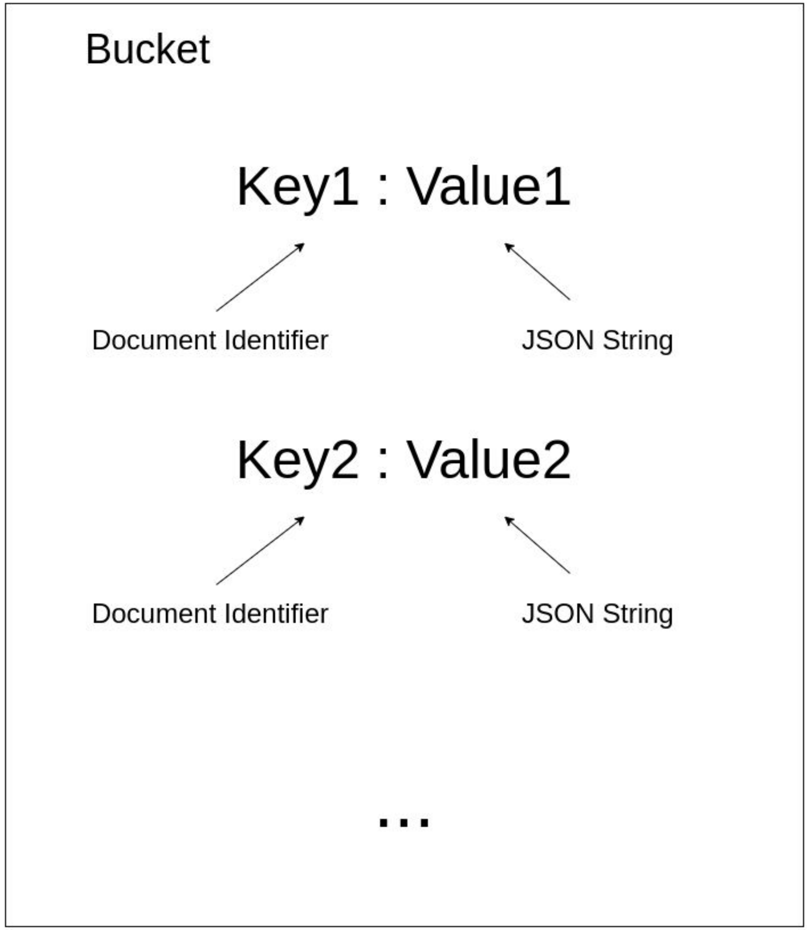


Production Rules (in GNU Bison)

```
_start:
      special_commands
    | query
    ;
special_commands:
      '\' SHOW_BUCKETS { bucket_manager->listBuckets(); }
    ...
    ;

query:
      SELECT * FROM name  { bucket_manager->selectBucket($2, $4); }
    |
      CREATE BUCKET name { bucket_manager->createBucket($3); }
    ...
    ;
```

Bucket Architecture

The bucket is a collection of documents where each document is identified by the Document Identifier i.e; key and value of the document is a JSON string.



Ephemeral Buckets

ElkDB by default supports ephemeral buckets.

Ephemeral buckets does not persist data to disk.

The main advantage of Ephemeral buckets is to omit disk reads and writes and utilize only the main memory.

Couchbase a NoSQL database softwares supports this bucket.

NoSQL Query in ElkDB

```
-- Creates a bucket.
CREATE BUCKET <BUCKET_NAME>;

-- Inserts document into <BUCKET_NAME> if <DOCUMENT_ID> does not exists.
INSERT INTO <BUCKET_NAME> VALUES (‘<DOCUMENT_ID>', '<JSON_DOCUMENT_DATA>');

-- Updates the document if already exists, otherwise inserts a new document.
UPSERT INTO <BUCKET_NAME> VALUES (‘<DOCUMENT_ID>', '<JSON_DOCUMENT_DATA>');

-- Deletes the document with <DOCUMENT_ID> from the <BUCKET_NAME>
DELETE FROM <BUCKET_NAME> WHERE _ID='<DOCUMENT_ID>';

-- Returns all the customers as JSON
SELECT * FROM <BUCKET_NAME>;
```

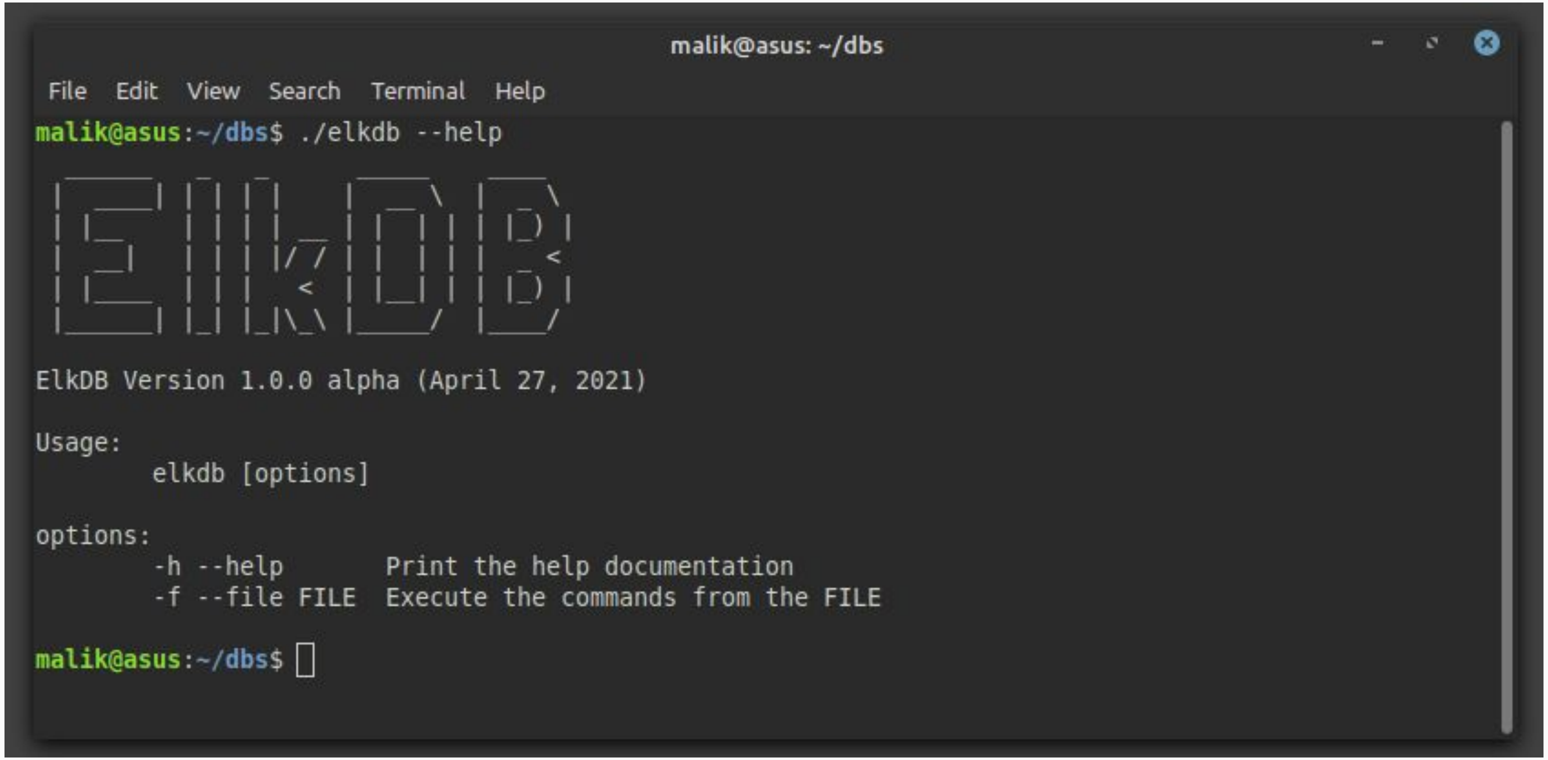
JSON Schema

This customer schema contains two sample customers data (C20, and C21).

```
{
    "C20": {
        "name": "John Doe",
        "address": {
            "street": "4010 W Housing Dr.",
            "city": "Fort Wayne, IN",
            "zipcode": "46815"
        }
    },
    "C21": {
        "name": "Anna Roberts",
        "address": {
            "street": "455 North Cityfront Plaza",
            "city": "Chicago, IL",
            "zipcode": "60611"
        },
        "rating": 565
    }
}
```

Demo

Help

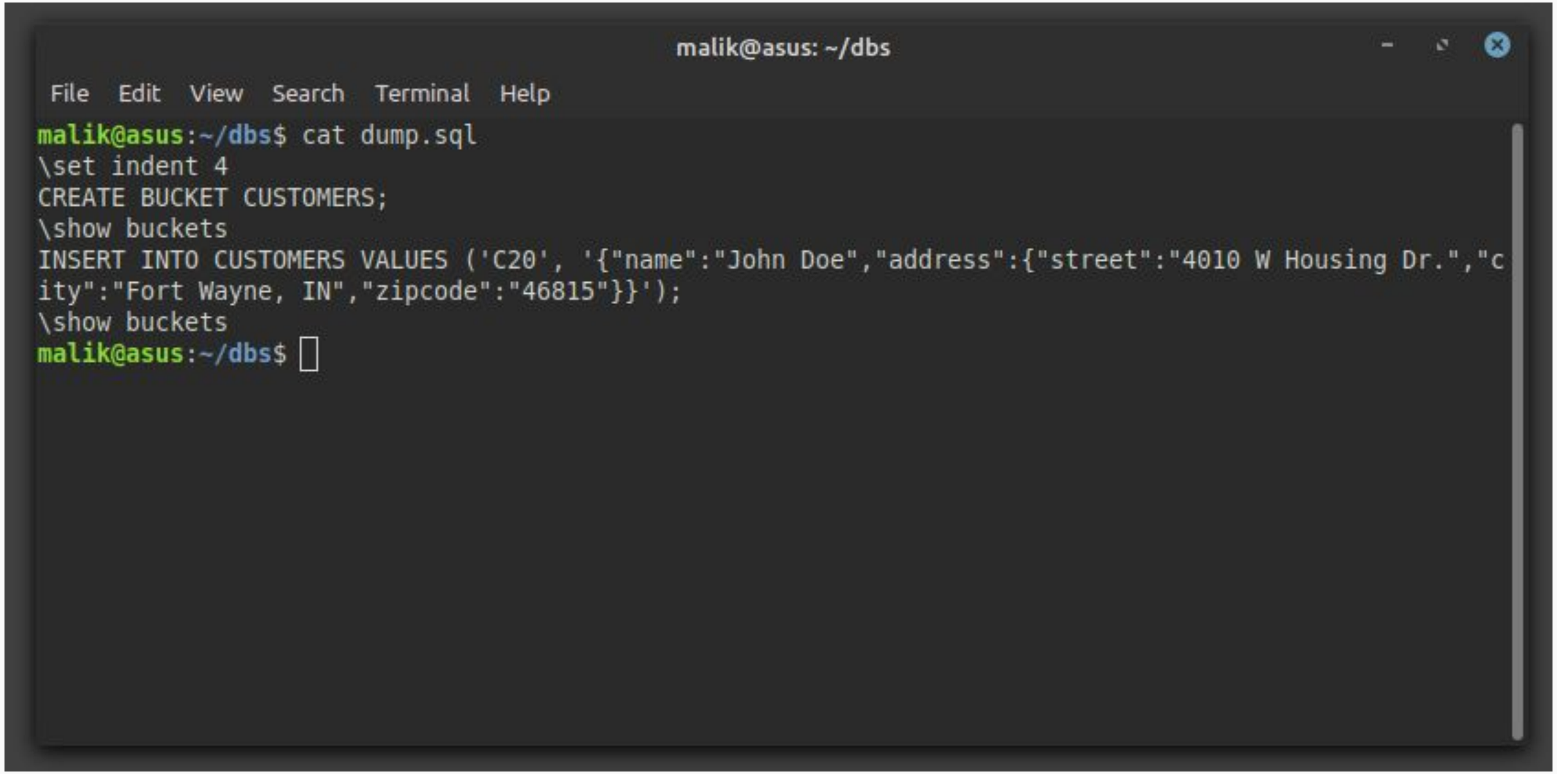


```
malik@asus:~/dbs$ ./elkdb --help
 ______  _   _      _____   ____
|  ____|| | | |    |  __ \ |  _ \
| |__   | | | | __ | |  | || |_) |
|  __|  | | | |/ / | |  | ||  _ <
| |____ | | |   <  | |__| || |_) |
|______||_| |_|\_\ |_____/ |____/

ElkDB Version 1.0.0 alpha (April 27, 2021)

Usage:
        elkdb [options]

options:
        -h --help       Print the help documentation
        -f --file FILE  Execute the commands from the FILE

malik@asus:~/dbs$
```

dump.sql

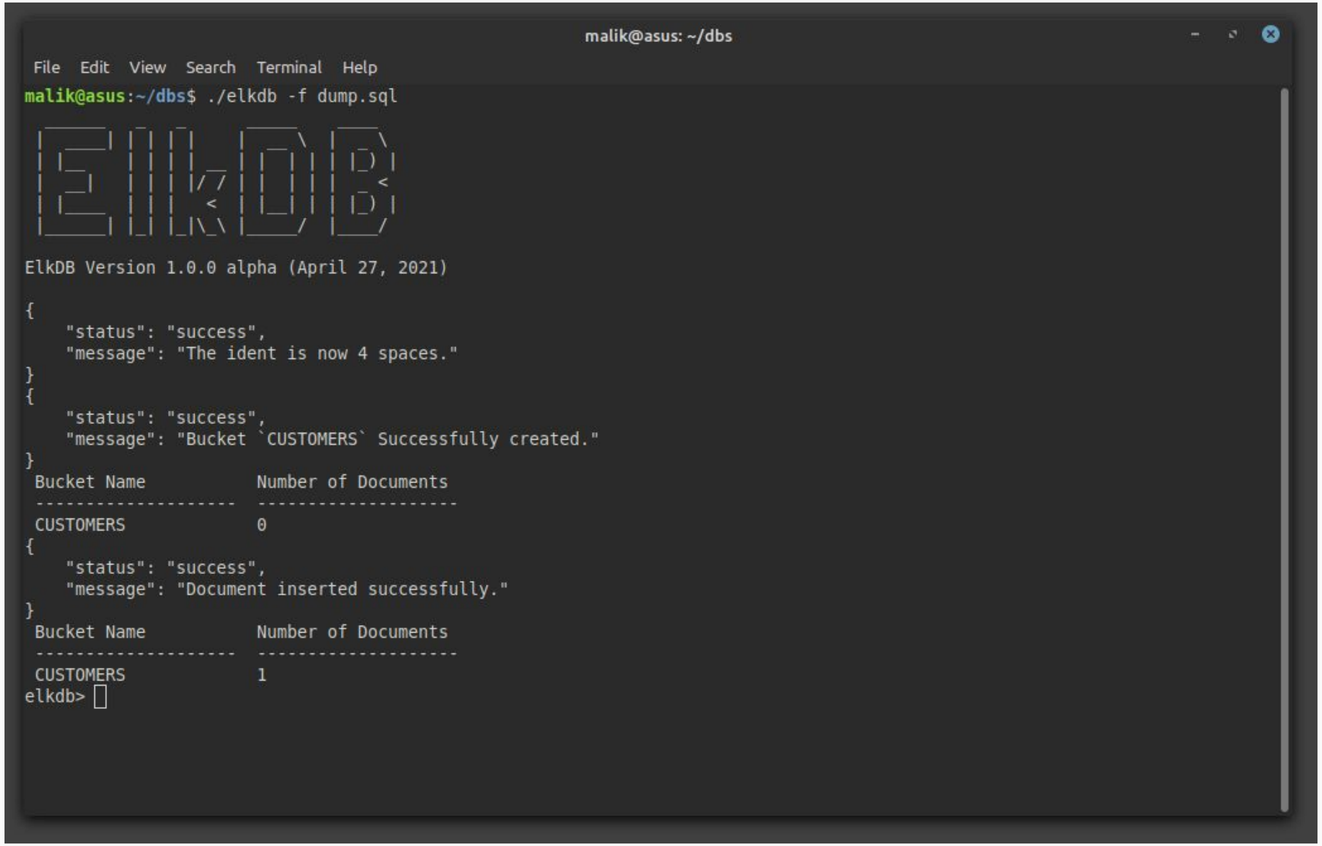
```
\set indent 4

CREATE BUCKET CUSTOMERS;

\show buckets

INSERT INTO CUSTOMERS VALUES ('C20', '{"name":"John
Doe","address":{"street":"4010 W Housing Dr.","city":"Fort
Wayne, IN","zipcode":"46815"}}');

\show buckets
```

dump.sql



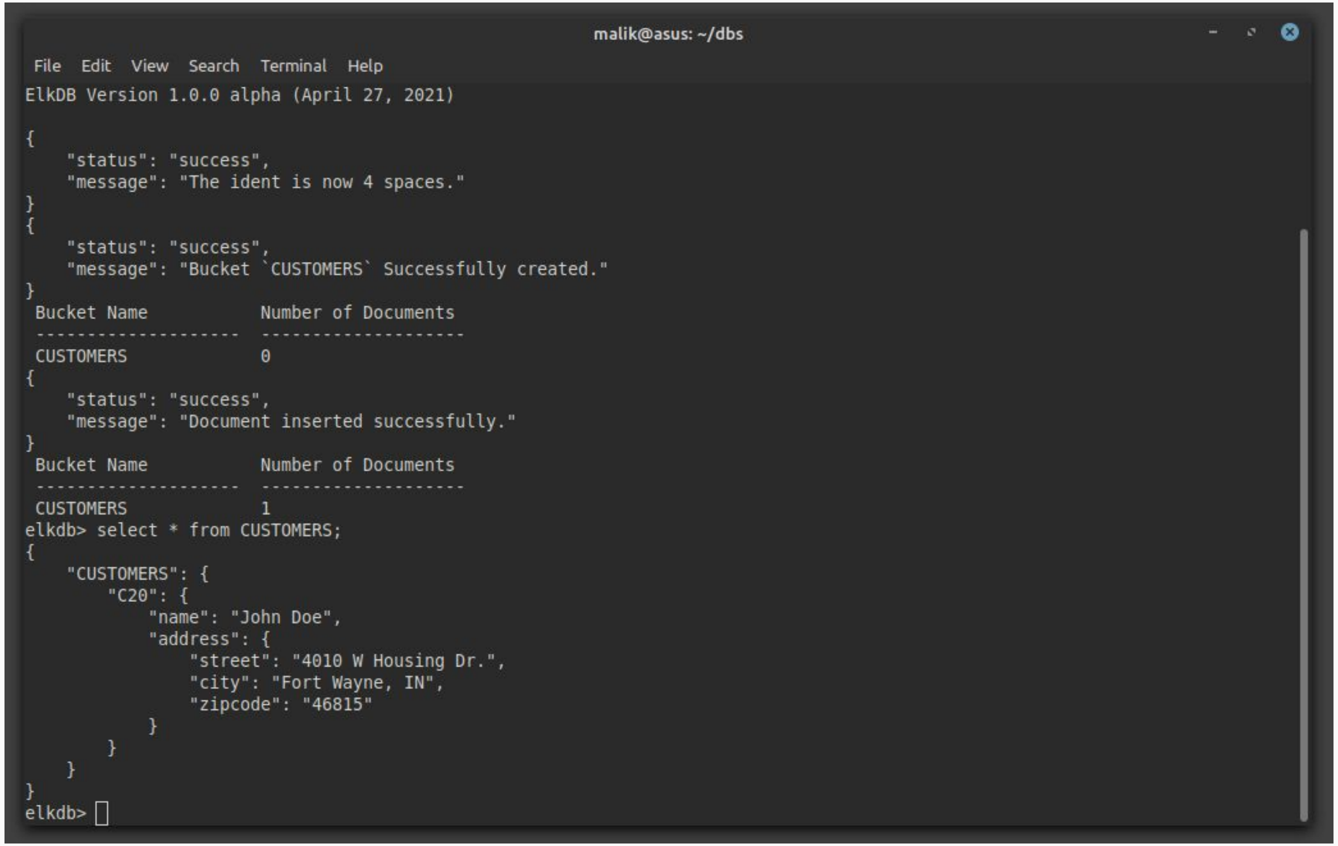
```
malik@asus:~/dbs$ cat dump.sql
\set indent 4
CREATE BUCKET CUSTOMERS;
\show buckets
INSERT INTO CUSTOMERS VALUES ('C20', '{"name":"John Doe","address":{"street":"4010 W Housing Dr.","c
ity":"Fort Wayne, IN","zipcode":"46815"}}');
\show buckets
malik@asus:~/dbs$ 
```

Dumping data to ElkDB



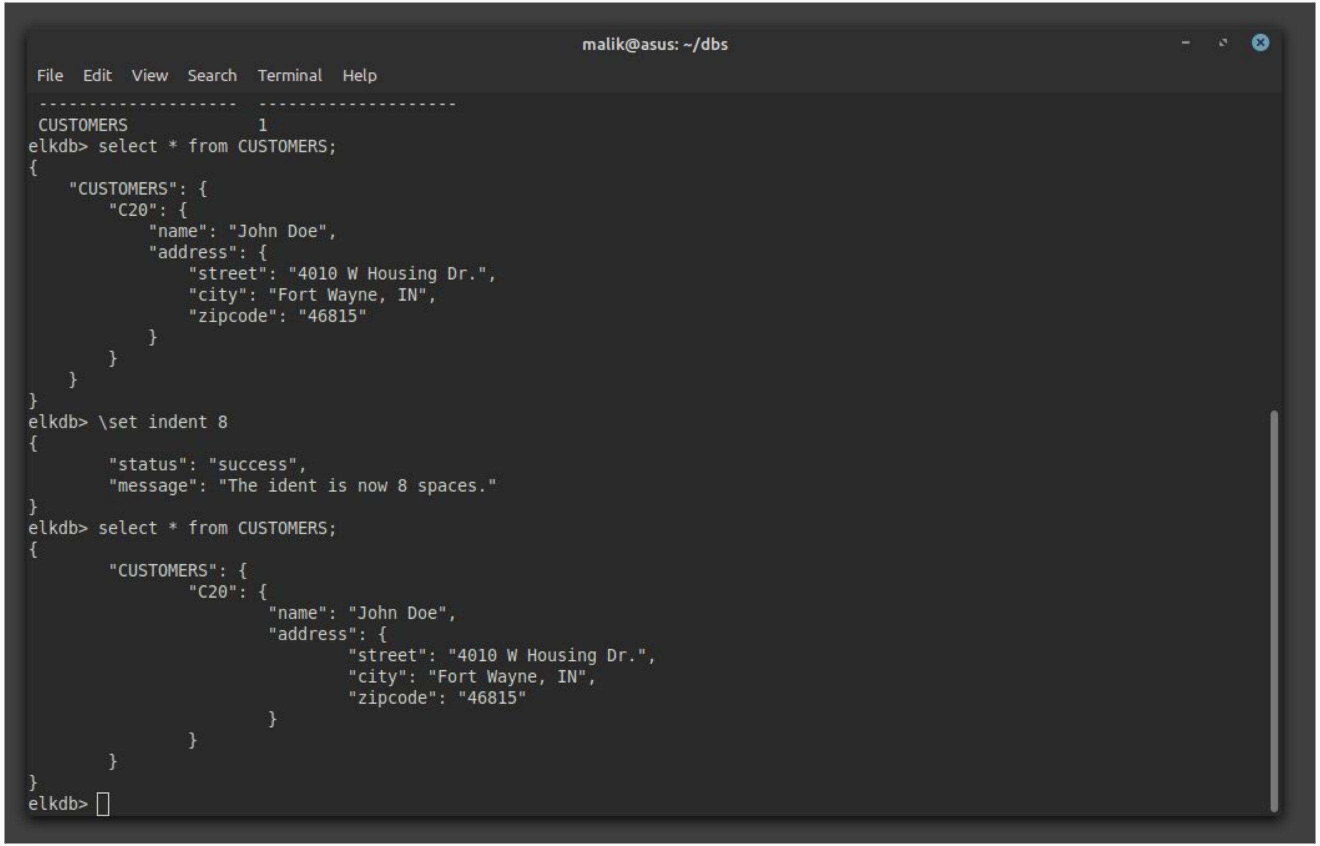
```
malik@asus:~/dbs$ ./elkdb -f dump.sql

  ______   _   _      _____    ____
 |  ____| | | | |    |  __ \  |  _ \
 | |__    | | | | __ | |  | | | |_) |
 |  __|   | | | |/ / | |  | | |  _ <
 | |____  | | |   <  | |__| | | |_) |
 |______| |_| |_|\_\ |_____/  |____/

ElkDB Version 1.0.0 alpha (April 27, 2021)

{
    "status": "success",
    "message": "The ident is now 4 spaces."
}
{
    "status": "success",
    "message": "Bucket `CUSTOMERS` Successfully created."
}
 Bucket Name            Number of Documents
 --------------------  --------------------
 CUSTOMERS              0
{
    "status": "success",
    "message": "Document inserted successfully."
}
 Bucket Name            Number of Documents
 --------------------  --------------------
 CUSTOMERS              1
elkdb> 
```

Selecting from dumped data



```
ElkDB Version 1.0.0 alpha (April 27, 2021)

{
    "status": "success",
    "message": "The ident is now 4 spaces."
}
{
    "status": "success",
    "message": "Bucket `CUSTOMERS` Successfully created."
}
 Bucket Name           Number of Documents
 --------------------  --------------------
 CUSTOMERS             0
{
    "status": "success",
    "message": "Document inserted successfully."
}
 Bucket Name           Number of Documents
 --------------------  --------------------
 CUSTOMERS             1
elkdb> select * from CUSTOMERS;
{
    "CUSTOMERS": {
        "C20": {
            "name": "John Doe",
            "address": {
                "street": "4010 W Housing Dr.",
                "city": "Fort Wayne, IN",
                "zipcode": "46815"
            }
        }
    }
}
elkdb> 
```

Setting indentation



```
 --------------------  --------------------
 CUSTOMERS             1
elkdb> select * from CUSTOMERS;
{
    "CUSTOMERS": {
        "C20": {
            "name": "John Doe",
            "address": {
                "street": "4010 W Housing Dr.",
                "city": "Fort Wayne, IN",
                "zipcode": "46815"
            }
        }
    }
}
elkdb> \set indent 8
{
        "status": "success",
        "message": "The ident is now 8 spaces."
}
elkdb> select * from CUSTOMERS;
{
        "CUSTOMERS": {
                "C20": {
                        "name": "John Doe",
                        "address": {
                                "street": "4010 W Housing Dr.",
                                "city": "Fort Wayne, IN",
                                "zipcode": "46815"
                        }
                }
        }
}
elkdb> 
```

Limitations

This database software right now doesn’t support persistence to disk.

It is available only for Linux based operating system.

Conclusion

ElkDB can be used in the IoT based devices where the big database software cannot be installed.

Mozilla’s Web of Things (WoT) uses JSON as the file exchange format to communicate between the IoT devices.

It’s going to be open-source:

https://github.com/ElkDB