

Mordor DB

# Implementation

MordorDB is a console application written in C# .  Data is stored on disk using JSON.  The most recently accessed data will be stored in memory. In memory, objects are defined recursively. Each collection can contain any number of key-value pairs. The keys will be strings, and the values can either be primitive values (such as strings or integers) or sub-collections.

The log is a JSON object stored in a separate file. It is held in memory, and flushed to the disk every n operation, where n is configurable (these are the automatic checkpoints). The user can also force the log to be written to disk using a checkpoint keyword. The user can rollback to the last checkpoint using a rollback command.

Indexes are implemented using B+ trees. The user can create an index on any collection or sub-collection. If the user’s query only contains items that are in an index, that index will be used with referring back to the base collection.

# Query Language

MordorDB includes the capabilities to create, retrieve, update and delete via method called from command line. Results can be filtered with a where command.  In addition it includes aggregate functions for sum, max, min, average and count. Queries have specific commands followed by their node notation, for example **myCollection.myField.**

# User Manual

## Getting Started

In order to execute MordorDB, simply click on the exe file found in the debug directory of the bin folder. All files related to the database will be stored in the directory where the executable is located so creating a new folder for organization is recommended but not required.

## Commands

The following commands are available from the prompt:

**print** –

Displays a formatted view of the current data stored in memory.

**checkpoint** –

Creates a checkpoint and forces the log to write to disk

**rollback** –

Rolls the data back to its state at the last checkpoint

**help** –

Displays a list of commands that are available

**clear** –

Clears the terminal window

**exit** –

Writes the current data to file and ends the program