BoltDB

Шокарева Лилия

История создания

LMDB -> BoltDB

Задумка: простое быстрое key-value хранилище без дополнительных фичей

2017 - конец разработки

```go
var world = []byte("greeting")
func main() {
    //create database
    // if there is no bolt.db file, it will be created
    db, err := bolt.Open("C:/Users/kingn/go_test/bolt.db", 0644, nil)

    defer db.Close()
    key := []byte("hello")
    value := []byte("Hello World!")
    // store data
    err = db.Update(func(tx *bolt.Tx) error {
        //create bucket
        bucket, err := tx.CreateBucketIfNotExists(world)
        err = bucket.Put(key, value)
        return nil
    })

    //get data
    err = db.View(func(tx *bolt.Tx) error {
        // Assume bucket exists and has keys
        bucket := tx.Bucket(world)
        if bucket == nil {
            return fmt.Errorf("Bucket %s not found!", world)
        }
        val := bucket.Get(key)
        fmt.Println(string(val))
        return nil
    })
}
```

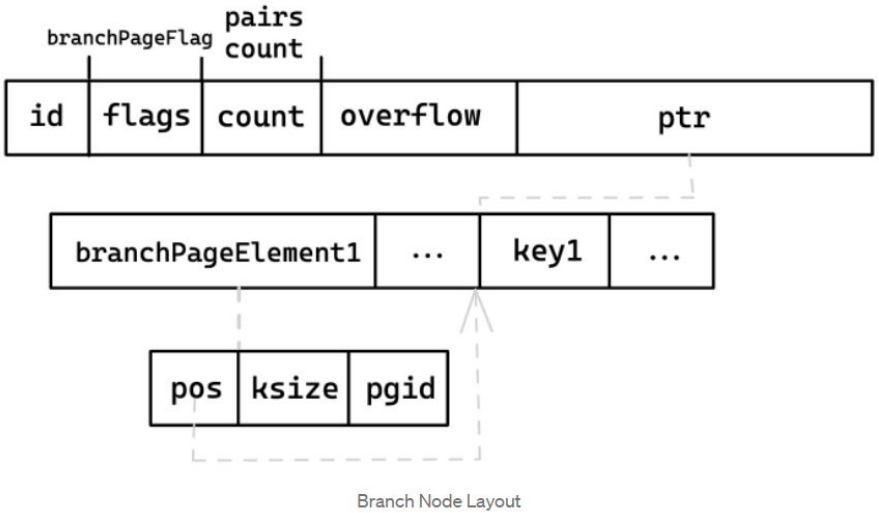
Начало работы и запросы

Вывод:

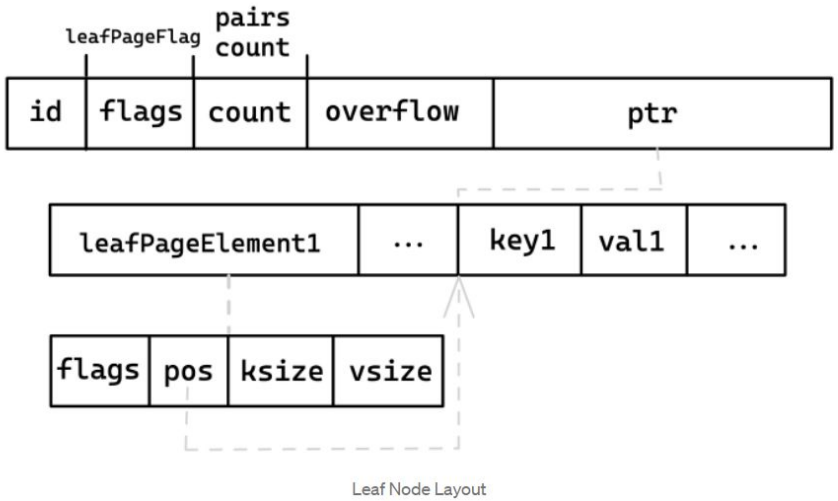
```
C:\Users\kingn\go_test>go run .
Hello World!
```

Внутреннее устройство

B+ дерево



узел не лист



узел лист

Про транзакции

- только одна read-write транзакция за раз
- сколько угодно read-only транзакций за раз
- отдельные транзакции не потокобезопасны
- создание транзакции потокобезопасно

Чего нет в BoltDB

- индексы
- патрицирование
- шардинг
- план запросов
- шифрование

Текущее развитие

- bbolt - fork основного репозитория, разработка ведется до сих пор
- strom - инструментарий для BoltDB, предоставляет индексы и дополнительные методы хранить и получать данные, а также улучшенная система запросов

Сравнение с другими СУБД

Postgres, MySQL, и другие реляционные бд

- структурирование данных *vs* доступ по ключу, без объединения
- отдельный сервер с данными *vs* импортированная библиотека

LevelDB, RocksDB

- LSM-деревья *vs* B-деревья

LMDB

- чистая производительность *vs* удобство использования

BoltDB

Шокарева Лилия