RocksDB - Compaction Filter

Compaction Filter trong RocksDB là một cơ chế cho phép bạn thực hiện các thay đổi tùy chỉnh trên dữ liệu khi RocksDB thực hiện quá trình Compaction. Quá trình Compaction là quá trình tối ưu hóa dữ liệu trong RocksDB bằng cách ghép các file SST thành file mới để tiết kiệm không gian lưu trữ và tăng hiệu suất truy vấn.

Với Compaction Filter, bạn có thể thay đổi giá trị của key hoặc loại bỏ key khỏi RocksDB trong quá trình Compaction. Điều này rất hữu ích khi bạn muốn áp dụng các quy tắc tùy chỉnh cho dữ liệu trong quá trình Compaction.

Dưới đây là một ví dụ về việc sử dụng Compaction Filter trong RocksDB bằng Java:

```
import org.rocksdb.*;

public class RocksDBCompactionFilterExample {

    public static class MyCompactionFilter extends AbstractCompactionFilter {
        @Override
        public boolean filter(
                int level, byte[] key, byte[] existingValue, byte[] value) {
            // Kiểm tra điều kiện tùy chỉnh và quyết định có loại bỏ key hay không
            // Trả về true nếu muốn loại bỏ key, ngược lại trả về false
            return false;
        }
    }

    public static void main(String[] args) {
        RocksDB.loadLibrary();
        try (final Options options = new Options().setCreateIfMissing(true);
             final RocksDB db = RocksDB.open(options, "/path/to/database")) {
            final ColumnFamilyHandle cfHandle = db.getDefaultColumnFamily();
            db.addCompactionFilter(cfHandle, new MyCompactionFilter());
        } catch (RocksDBException e) {
            // Xử lý lỗi khi thêm Compaction Filter không thành công
        }
    }
}
```

C++:

```
#include "rocksdb/db.h"

class MyCompactionFilter : public rocksdb::CompactionFilter {
public:
    bool Filter(int level,
                const rocksdb::Slice& key,
                const rocksdb::Slice& existing_value,
                std::string* new_value,
                bool* value_changed) const override {
        // Kiểm tra điều kiện tùy chỉnh và quyết định có loại bỏ key hay không
        // Trả về true nếu muốn loại bỏ key, ngược lại tr
```

Trong ví dụ trên, chúng ta tạo một class MyCompactionFilter kế thừa từ AbstractCompactionFilter và ghi đè phương thức filter(). Trong phương thức filter(), chúng ta có thể kiểm tra các điều kiện tùy chỉnh và quyết định xem có loại bỏ key hay không. Nếu muốn loại bỏ key, chúng ta trả về true, ngược lại trả về false.

Sau đó, chúng ta thêm Compaction Filter vào RocksDB bằng cách sử dụng phương thức addCompactionFilter(). Trong trường hợp này, chúng ta sử dụng getDefaultColumnFamily() để lấy handle của column family mặc định trong RocksDB.

Revision #1
Created 23 September 2023 09:34:32 by Laptrinh.vn
Updated 23 September 2023 10:56:18 by Laptrinh.vn