Database Optimization

```
#include <iostream>
#include <vector>
#include <omp.h>

using namespace std;

// define a struct for database entry
struct DatabaseEntry {
    int id;
    string name;
    int age;
};

// define a vector to hold database entries
vector<DatabaseEntry> database;

// function to add an entry to the database
void addEntry(DatabaseEntry entry) {
    #pragma omp critical
    {
        database.push_back(entry);
    }
}

// function to delete an entry from the database
void deleteEntry(int id) {
    #pragma omp parallel for
    for(int i=0; i<database.size(); i++) {
        if(database[i].id == id) {
            #pragma omp critical
            {
                database.erase(database.begin() + i);
            }
        }
    }
}

// function to update an entry in the database
void updateEntry(int id, string name, int age) {
    #pragma omp parallel for
    for(int i=0; i<database.size(); i++) {
        if(database[i].id == id) {
```

```
            #pragma omp critical
            {
                database[i].name = name;
                database[i].age = age;
            }
        }
    }
}

// function to retrieve an entry from the database
DatabaseEntry getEntry(int id) {
    DatabaseEntry result;
    #pragma omp parallel for
    for(int i=0; i<database.size(); i++) {
        if(database[i].id == id) {
            #pragma omp critical
            {
                result = database[i];
            }
        }
    }
    return result;
}

int main() {
    // get number of entries from user
    int numEntries;
    cout << "Enter number of database entries: ";
    cin >> numEntries;

    // get database entries from user
    for(int i=0; i<numEntries; i++) {
        int id, age;
        string name;
        cout << "Enter database entry #" << i+1 << ":" << endl;
        cout << "ID: ";
        cin >> id;
        cout << "Name: ";
        cin >> name;
        cout << "Age: ";
        cin >> age;
        addEntry({id, name, age});
    }
```

```
    // delete an entry from the database
    int deleteId;
    cout << "Enter ID of entry to delete: ";
    cin >> deleteId;
    deleteEntry(deleteId);

    // update an entry in the database
    int updateId, updateAge;
    string updateName;
    cout << "Enter ID of entry to update: ";
    cin >> updateId;
    cout << "Enter updated name: ";
    cin >> updateName;
    cout << "Enter updated age: ";
    cin >> updateAge;
    updateEntry(updateId, updateName, updateAge);

    // retrieve an entry from the database
    int getId;
    cout << "Enter ID of entry to retrieve: ";
    cin >> getId;
    DatabaseEntry entry = getEntry(getId);
    cout << "Name: " << entry.name << ", Age: " << entry.age << endl;

    return 0;
}
```