Федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
Факультет информатики, математики и компьютерных наук

ЛАБОРАТОРНАЯ РАБОТА №2

Файловая база данных

Выполнил:
Студент группы 23КНТ-5
Камаева Е.С.
Ф.И.О.

Нижний Новгород 2024

СОДЕРЖАНИЕ

1. Описание выбранной предметной области

Завод занимается производством деталей различных наименований. Контроль качества и управление информацией о произведенных деталях играют ключевую роль в работе предприятия. Система должна обеспечивать хранение, структуризацию и анализ данных, связанных с производством и реализацией изделий.

Итак, ключевой элемент – деталь. Требуется хранить информацию о множестве деталей в табличном представлении.

У каждого изделия есть следующие характеристики:

- 6-значный серийный номер изделия, уникальный для каждой детали;
- 6-значное кодовое название (наименование);
- дата производства;
- индекс соответствия эталону (значение варьируется от 0.01 до 1.00 – эта характеристика определяется с помощью измерительных приборов на станке);
- метка «Продано» (“+” или “-“).

2. Цель работы

Определим **цель работы**:

Реализовать файловую базу данных для работы с информацией по выбранной предметной области.

3. Краткое описание реализации

Работа с базой данных основана на использовании хеш-таблиц. Этот подход позволяет добиться оптимальной сложности алгоритмов (поиска, удаления, вставки).

Файл БД: .csv; файлы хеш-таблиц: .csv; файл removed: .txt.

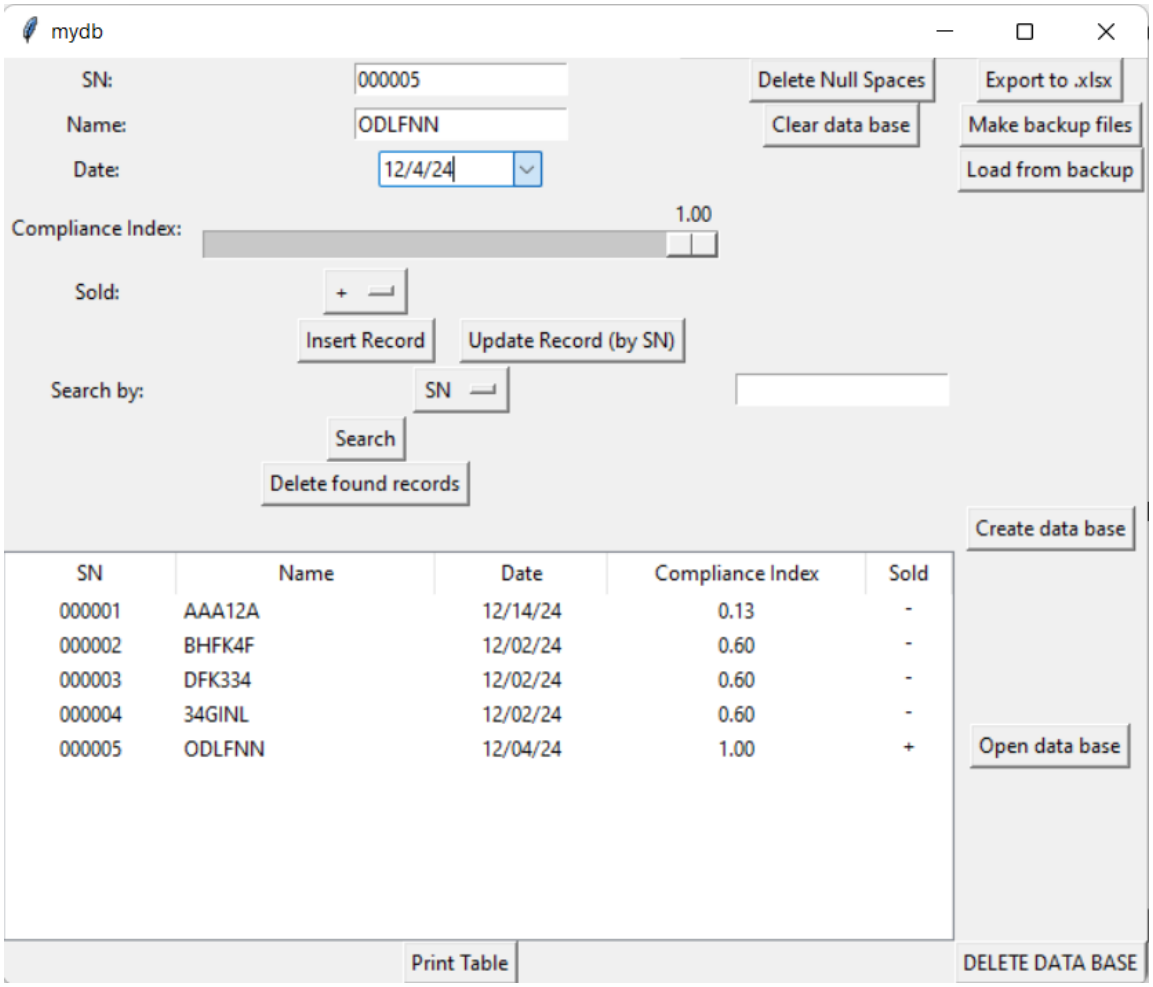
Для каждого поля создаётся отдельная хеш-таблица, в которой пара ключ-значение выглядит следующим образом: ключ – значение самого поля, значение – набор сдвигов в байтах относительно начала файла (т.е. это список), где поле принимает значение, соответствующее ключу. При начале работы программы хеш-таблицы не создаются на основе таблицы БД, а считываются из соответствующих файлов. Такой подход будет обоснован дальше.

Поиск осуществляется с помощью хеш-таблиц. Удаление, в свою очередь, тесно связано с поиском (ведь нужно найти все записи, совпадающие по значению какого-либо поля). В данной реализации удаление не происходит физически; строка помечается, как удаленная, заменой SN на строку вида «------« - это позволяет быстро и просто идентифицировать удаленную запись при выводе таблицы на экран.

Нужно отметить, что вместе с самой БД и файлами хеш-таблиц хранится файл removed, а также список removed в памяти с записанными позициями удалённых записей. Это нужно для того, чтобы оптимизировать использование памяти, вставляя новые записи на место удалённых. Если же удалённых записей нет, то новые строки добавляются в конец файла.

Теперь вернемся к необходимости создания отдельных файлов для хеш-таблиц. Из-за особенностей реализации алгоритма удаления не исключена ситуация, когда бóльшее количество строк – пустые. В этом случае при создании хеш-таблиц из файла БД прохождение по файлу будет «непродуктивным»: записей много, и лишь немногие нужно занести в хеш-таблицы. Считывание с отдельных файлов, в которых хранятся только уникальные значения полей, эффективнее использует ресурсы.

4. Основные функции GUI



Работа с базой данных осуществляется через графический интерфейс. Пользователю доступны следующие функции:

- Create data base – создать новую базу данных;
- Open data base – открыть существующую базу данных;
- DELETE DATA BASE – удалить базу данных и все её файлы;
- Insert – вставка новой записи в БД. Чтобы выполнить операцию, необходимо заполнить все поля: SN, Name, Date, Compliance Index и Sold;
- Update Record (by SN) – обновление существующей записи по SN. Пользователю нужно ввести SN записи, которую он хочет изменить, а также новые значения полей;

- Search – поиск записей в БД по заданному полю (выбирается в выпадающем списке). На экран выводятся все совпавшие по значению записи;
- Delete found records – удаление найденных с помощью Search записей;
- Delete Null Spaces – удаление пустых строк в таблице. Функция нужна, чтобы время от времени очищать БД от ненужных (удаленных) строк;
- Clear data base – очистка базы данных без её удаления;
- Export to .xlsx – экспорт таблицы в файл формата .xlsx;
- Make backup files – создание бэкап-файлов;
- Load from backup – загрузка таблицы из бэкап-файлов.

5. Сложность алгоритмов

Для временной статистики были написаны функции для генерации n записей в таблице (доступны в исходных файлах в bd.py). В результате измерений были получены следующие результаты:

n	*Вставка n записей*	*Вставка 1 записи, сек*	*Поиск, сек*	*Удаление, сек*
10	0,0036	0,0004	0,0145	0,0000
50	0,0165	0,0003	0,0145	0,0010
100	0,2790	0,0028	0,0140	0,0006
200	0,0532	0,0003	0,0163	0,0010
400	0,1119	0,0003	0,0166	0,0005
600	0,1503	0,0003	0,0170	0,0006
800	0,7684	0,0010	0,0141	0,0000
1000	0,4543	0,0005	0,0180	0,0000
1500	0,8986	0,0006	0,0199	0,0000
3000	1,1870	0,0004	0,0181	0,0000
5000	1,8373	0,0004	0,0211	0,0010
10000	3,0435	0,0003	0,0272	0,0010


Время вставки
0,0500
0,0400
0,0300
0,0200
0,0100
0,0000
Время, с
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000
Кол-во записей в таблице, n


Время поиска
0,3000
0,2500
0,2000
0,1500
0,1000
0,0500
0,0000
Время, с
0
2000
4000
6000
8000
10000
Кол-во записей в таблице, n


Время удаления
0,0500
0,0400
0,0300
0,0200
0,0100
0,0000
Время, с
0
2000
4000
6000
8000
10000
Кол-во записей в таблице, n

Как и ожидалось, реализованные через хеш-таблицы алгоритмы показали сложность ~ O(1) – константная. В среднем обращение к элементу хеш-таблицы по ключу (а именно это и используется в алгоритмах поиска/удаления), далее возвращается список смещений и за O(1) в файле находятся эти значения (в файле можно индексироваться по байтам, как в массиве, за константное время). Небольшое увеличение времени работы алгоритма поиска со значительным увеличением числа записей может быть связано с возникновением коллизий, однако в среднем сложность остаётся O(1). При использовании для поиска значений, которые могут повторяться, сложность не увеличится, т.к. все смещения для соответствующего значения мы находим в хеш-таблице O(1) по ключу (а потом либо выводим записи на экран, либо помечаем как удалённые).

6. Заключение

В ходе лабораторной работы было дано описание предметной области «Производство деталей на заводе». Была создана файловая база данных. По результатам лабораторной работы сформирован отчёт с анализом используемых алгоритмов.