CS-6360 Database Design

Project Milestone #1

Individual Submission

This is an individual assignment. Team members may collaborate with each other, but each student should submit their own separate program. Students are encouraged to experiment with different approaches to designing their program.

The objective is to create a program that opens a CSV format file (ASCII plain text) and read the records in it. You will then write the records to a new file whose format that is similar to how records are written to table files in a relational SQL database. Both the CSV file and this new file format are examples of a Storage Definition Language (SDL).

The records you may use for the development of your program are included in the file "`employee.csv`". The data file used for grading will be the same CSV schema, but a different employee data set.

Table File Description

Your program shall create a table file named `employee.tbl`. The file shall have a page size of 4096 bytes (i.e. 4kb) and be 10 pages long—numbered page 0 to page 9. Initially, the file will have every byte written with the byte value `0x00` byte. Therefore the file will be 40,960 bytes in size, both file size and "size on disk".

Your program shall read in a CSV file containing employee data in the format of Table 1. Each file record shall be written to the page corresponding to $H(k) = k \bmod 10$, based on the record's primary key of Ssn. Thus, a record with Ssn of 482-72-8472 should be written to page number 2 (the third page from the beginning of the file.

Each record is exactly 112 bytes long and shall be written to last 112 available bytes (the highest offset/ address) in its target page *in the order that it is processed from the import CSV file*. The first record in the CSV file shall be assigned the `rowid` 1, the next record 2, and so on. This is like auto_increment in SQL.

Each page in the table file has a sequence of header bytes at the beginning of the page (the lowest offset/ address). The first two bytes is a 16-bit two's complement integer (short int) with the high order byte first (i.e. "big endian"). This integer indicates the number of records on that page. The remainder of the page header is sequence of 2-byte integers, each of which indicates the offset (in byte) of each record on the page. The sequence of offsets should be ordered so that the `Ssn` values of their coresponding records is in Ssn order. This sequence of record page offsets is the ***index for the page***.

Important: The record content shall be written to the page in `rowid` order—the order that they were inserted—from the page's high address progressing to lowest address, regardless of the `Ssn`. However, the order of the index offsets should be in sorted order according to the `Ssn` of the each's coresponding record on the page.

Attribute Name	Data Type	Size (bytes)
Row ID	Integer	4
Ssn	String	9
First Name	String	20
Middle Initial	Byte	1
Last Name	String	20
Birth Date	String	10
Address	String	40
Sex	Byte	1
Salary	Integer	4
Department Number	Short Integer	2
Deletion Marker	Byte	1
Record Size Total		**112**

Table 1. Employee Table Schema