CSE177/EECS277 – DATABASE SYSTEMS IMPLEMENTATION
Project 3: Simple Relational Algebra Operators (Scan, Selection, Projection, WriteOut)
Due date: March 15 & 16 (in the lab)

This project requires the implementation of four relational algebra operators: Heap Scan, Selection, Projection, and WriteOut. With these operators, it is possible to run simple `SELECT-FROM-WHERE` SQL queries.

Heap Scan

A heap file stores the records of a table. The records are grouped into pages, which represent the I/O unit. Pages eliminate the requirement to access the file (disk) for every record. This improves the I/O bandwidth utilization. Records are stored in arbitrary order in a heap file. The only access path to a heap file is to read sequentially all the records, from beginning to end.

Class `DBFile` (`DBFile.h` and `DBFile.cc`) contains the interface of a heap file. It is straightforward:

- `Create` creates a new heap file. `FileType` has to be `Heap`. This is done only once, when a SQL table is created.
- `Open` gives access to the heap file. The name is taken from the catalog, for every table.
- `Close` closes the file.
- `MoveFirst` resets the file pointer to the beginning of the file, i.e., the first record.
- `GetNext` returns the next record in the file. The file pointer is moved to the following record.
- `AppendRecord` appends the record passed as parameter to the end of the file. This is the only method to add records to a heap file.
- `Load` extracts records with a known schema from a text file passed as parameters. Essentially, it converts the data from text to binary. Method `ExtractNextRecord` from class `Record` does all the work for a given schema.

You are required to implement all these methods. The good news is that we already provide you with all the functionality to access paged files on disk. Files `File.h` and `File.cc` contain two classes. Class `File` provides all the necessary functionality to access the pages from a file. Class `Page` stores records on pages. The methods from the heap file `DBFile` have only to invoke the methods of these two classes correctly.

The `Scan` relational operator is set-up by the query compiler. At runtime, method `GetNext` is invoked repeatedly. `Scan` reads the records from the heap file sequentially and passes them to the operator invoking `GetNext`. You have to invoke the methods of class `DBFile` for this.

Selection

Implement the `GetNext` method from the `Select` relational operator. Call `GetNext` for the producer operator. For every returned record, apply the selection predicate and, if the record satisfies the predicate, pass it to the invoking operator. `GetNext` returns when a record satisfying the predicate is found or no records exist anymore. Method `Run` from class `CNF` checks if the record satisfies the predicate, passed as a record of constants.

Projection

Implement the `GetNext` method from the `Project` relational operator. Call `GetNext` for the producer operator. For every returned record, apply the projection and return the trimmed record. Class `Record` has a `Project` method that does the job for you.

WriteOut

Implement the `GetNext` method from the `WriteOut` relational operator. Call `GetNext` for the producer operator and write the returned record in `outFile`. Class `Record` has a `print` method for this. `outFile` is set to an arbitrary file.

Requirements

- Generate TPC-H data:
  1. Download and extract the TPC-H data generator from `http://www.tpc.org/tpch/`.
  2. Read the documentation and the README file.
  3. Read the instructions in makefile.suite and generate a Makefile from it by defining the required variables:
     - CC is the compiler
     - DATABASE is the database
     - MACHINE is the operating system
     - WORKLOAD is TPCH
  4. Compile the code according to the instructions in the documentation.
  5. Generate a scale factor 1 TPCH database using DBGEN. At this point you will have a series of .tbl files, one for each table. Take a look over these files.
- Load the data into your database, i.e., create a heap file for every table. Write a simple program that takes as input parameters the name of the table you load data into, the location of the heap file corresponding to this table, and the location of the text file with the data. The program reads the schema corresponding to the table from the catalog. Then, it creates the heap file and opens it. Finally, it invokes method `Load` with the schema and the text file. `Load` invokes `ExtractNextRecord` for all the lines in the text file and appends the resulting binary records to the heap file. The catalog is updated with the name of the heap file corresponding to the table argument and the heap file is closed.
- Execute the simple queries we provide you in folder `Queries`. For this, you have to implement method `ExecuteQuery` from class `QueryExecutionTree`. The method simply calls `GetNext` for the root node until no more records are generated. Add the call to `ExecuteQuery` to the end of file `main.cc` provided in phase 2 of the project, compile, and execute with a query.

Resources

- `http://www.tpc.org/tpch/`