Trabajo práctico 3: **Álgebra relacional**

Normativa

Límite de entrega: Coordinar con su corrector/a para hacer la entrega alguno de los siguientes miércoles: 13, 20, 27 de noviembre. Enviar el zip al mail: `algo2.dc+TP3@gmail.com`

Normas de entrega: Ver "Información sobre la cursada" en el sitio Web de la materia. (`http://campus.exactas.uba.ar`)

Versión: 1.2 del 21 de noviembre de 2019

1. Enunciado

El objetivo de este TP es implementar en C++ todos los módulos correspondientes a la base de datos diseñada en el TP2.

El código debería respetar el diseño propuesto en el TP 2 de la manera más fiel posible. Obviamente se permite y se espera que corrijan todos los potenciales *bugs* que puedan llegar a encontrar en el diseño así como adaptaciones requeridas durante la corrección del TP 2.

1.1. Módulos básicos

Pueden utilizar las siguientes clases de la STL de C++ para los respectivos módulos básicos:

Módulo	**Clase**
Lista Enlazada	`std::list`
Pila	`std::stack`
Cola	`std::queue`
Vector	`std::vector`
Diccionario Lineal	`std::map`
Conjunto Lineal	`std::set`

Además, se proveen los siguientes módulos:

1. Un archivo auxiliar `src/Tipos.h` que define algunos renombres de tipos.
2. Una implementación completa del módulo **Consulta**, en los archivos `src/Consulta.h` y `src/Consulta.cpp`. Pueden utilizar este módulo sin hacerle ninguna modificación. Sin embargo, no se proveen implementaciones del operador de asignación (`operator=`) ni del constructor por copia para este módulo. *Si los necesitan, deben implementarlos.*
3. Una implementación *incompleta* del módulo **Registro**, en los archivos `src/Registro.h` y `src/Registro.cpp`. Deberán completar la representación privada e implementación de los métodos.
4. Una clase *incompleta* **Driver**, en los archivos `src/Driver.h` y `src/Driver.cpp`, cuyo funcionamiento se detalla en la sección Testing, más adelante. Deberán completar la representación privada e implementación de los métodos para poder validar su implementación con respecto a los tests provistos por la cátedra.
5. En el directorio `src/modulos_basicos` se encuentran implementaciones de Conjunto Lineal y Diccionario Lineal usando los archivos header `linear_set.h` y `linear_map.h`, respectivamente.

1.2. Detalles de implementación

Como es usual en los trabajos del laboratorio, el proyecto debe estar escrito utilizando alguna versión del lenguaje `C++` que se encuentre soportada por el compilador `g++` (por ejemplo C++11 o C++14). En el directorio `src/` se deben ubicar todos los `.h`, `.cpp` y `.hpp`. Los distintos tipos de archivo deben respetar su finalidad esperada.

A excepción de la clases `Consulta` y `Registro`, deberán implementar una nueva clase para cada módulo definido en el diseño presentado durante el TP2 (e.j.: `BaseDeDatos` o `Tabla`). Para el módulo `Consulta` deberán usar la provista por la cátedra. Para el módulo `Registro` deberán completar la implementación en `Registro.h` y `Registro.cpp`, manteniendo la interfaz ya presente. Podrán agregar cualquier otra operación a su interfaz que requieran para el funcionamiento de su diseño.

2. Testing

Cada módulo debe venir acompañado de un conjunto de tests en un archivo separado que compruebe su correcta funcionalidad. Por ejemplo, si cuentan con un módulo `Tabla`, deben incluir un conjunto de tests en el archivo `tests/test_tabla.cpp`. No se pide que los tests sean exhaustivos, pero sí razonablemente completos y representativos.

Además, el TP debe pasar el conjunto de tests diseñados por la cátedra, que se encuentran en el archivo `tests/driver_test.cpp`. Puesto que cada grupo puede utilizar una interfaz diferente para la clase `BaseDeDatos`, dependiendo del diseño que cada grupo haya elegido en el TP 2, los tests interactúan con la base de datos exclusivamente a través de la clase `Driver`.

La clase `Driver` permite cargar tablas desde archivos de texto (en formato CSV) y ejecutar consultas sobre dichas tablas. Para utilizar correctamente el driver de test:

- **No deben** modificar los tests en `tests/test_driver.cpp`.
- **Deben** completar la parte privada y la implementación de los métodos de la clase `Driver` en `src/Driver.h` y `src/Driver.cpp`.

Los métodos que deben completar de la clase `Driver` son:

1. `void Driver::crearTabla(NombreTabla t, vector<NombreCampo> cs, NombreCampo k)` — Crea una tabla con nombre `t`, con una lista de campos `cs` de tal modo que `k` es el campo clave. *Precondición:* `k` debe ser uno de los elementos del vector `cs`.
2. `void Driver::insertarRegistro(NombreTabla t, Registro r)` Inserta un registro `r` en la tabla `t`. *Precondición:* `t` debe ser una tabla existente y el conjunto de campos del registro `r` debe coincidir con el conjunto de campos de la tabla.
3. `Respuesta consultar(const Consulta& q)` — Devuelve el resultado de ejecutar la consulta `q`. Notar que el tipo `Respuesta` es un renombre de `vector<Registro>`.

Además, la clase `Driver` cuenta con un método `void Driver::leerDataset(string dataset)` ya implementado por la cátedra que tiene el siguiente efecto:

- Crea una tabla con el nombre indicado en `dataset`.
- La llena con datos leídos desde un archivo ubicado en `datos/<dataset>.txt`

Por ejemplo:

```
Driver d;
d.leerDataset("trenes"); // Carga la tabla trenes desde datos/trenes.txt
Respuesta r = d.consultar(Consulta("select(from(trenes), linea, 'Roca')"));
```

3. Entrega

La entrega consistirá de un archivo en formato `zip` que deberá incluir:

- El documento digital en formato `pdf` con el diseño completo de todos los módulos correspondiente al TP2.
- El código fuente en `C++` que complete la implementación del programa.

La entrega se enviará por mail a la dirección `algo2.dc+tp3@gmail.com`. Además, cada grupo deberá coordinar con su corrector/a una **instancia de visado** del TP. El objetivo de la instancia de visado es analizar conjuntamente el código y comunicar personalmente las decisiones tomadas y dificultades encontradas durante el desarrollo.

El mail deberá tener como **Asunto** los números de libreta separados por punto y coma (;). Por ejemplo:

To: algo2.dc+tp3@gmail.com
From: alumno-algo2@dc.uba.ar
Subject: 123/45; 67/8; 910/11; 12/13
Adjunto: tp3.zip