Estructuras de Datos y Algoritmos

Entrega 1 – Sede Buceo - Obligatorio 2014

Se desea implementar un **sistema manejador de base de datos** que permita crear y mantener tablas, almacenar datos en ellas y realizar ciertas operaciones sobre los datos.

Una **base de datos** es un conjunto de tablas que poseen nombres únicos que permiten identificarlas. Una **tabla** es, desde el punto de vista lógico, un conjunto de **filas** y **columnas** (**campos**) donde se distribuye la información.

Ejemplo:

Tabla Personas

Nombre	**Apellido**	**CI**
Maria	Castaña	0000001
Juan	Perez	2256698
Daniel	Gonzalez	3333444
Laura	Perez	2123328

Cada columna tiene un nombre que debe ser único dentro de las columnas de la tabla y contiene datos de una misma naturaleza. Si tomamos como ejemplo la tabla de personas, cada columna representa un atributo de las personas. Luego, las filas contienen los datos de cada persona. Una **tupla** es la colección de datos presentes en una fila de la tabla, considerando el orden en que se presentan los datos. Esto nos lleva a que dos tablas con todas sus columnas iguales pero dispuestas en distinto orden se consideren tablas distintas. En otras palabras, una tabla es un conjunto de tuplas y una tupla es una lista de datos, debiendo cumplirse que para una misma tabla todas las tuplas tienen la misma cantidad de datos de los mismos tipos y en el mismo orden. No se permite que en una tabla existan dos tuplas idénticas.

Existen tres formas de **calificar** a una columna de una tabla: NOT EMPTY, PRIMARY KEY y ANY. NOT EMPTY significa que la columna no admite campos vacíos. Es decir no admite campos con valor EMPTY. El calificador PRIMARY KEY puede aplicarse a 0 o 1 columnas de una tabla. PRIMARY KEY indica que los valores de la columna no pueden ser vacíos y son únicos. Entonces, se puede identificar unívocamente a un tupla por el valor del campo correspondiente a la columna PRIMARY KEY. ANY se considera un calificador neutro, es decir, no restringe los valores de la columna. El valor EMPTY se acepta para cualquier columna, independientemente de su tipo de datos, siempre que la columna no esté calificada como NOT EMPTY ni como PRIMARY KEY. Notar que en el ejemplo previo de la tabla de personas, Apellido no podría ser un campo PRIMARY KEY, pero si el campo CI (de acuerdo a las tuplas del ejemplo).

Por motivos de simplicidad se considerarán sólo dos posibles tipos de datos almacenables en una tabla: STRING e INTEGER. Un STRING es una secuencia de caracteres donde se excluyen los siguientes caracteres: el mayor (>), el menor (<), el igual (=), el dos puntos (:) y el asterisco (*).

En resumen, el sistema deberá poder almacenar y administrar tablas que cumplan el siguiente esquema:
Un nombre que identifica a la tabla de manera única.
Las columnas, de las que se conoce su:
Nombre (que la identifica dentro de la tabla)
Tipo de datos (STRING, INTEGER)
Calificador que se aplica a esa columna (NOT_EMPTY, PRIMARY_KEY, ANY)

Consideraciones Generales

El sistema "**deberá**" implementar de forma eficiente las operaciones sobre la base de datos. En particular, teniendo en cuenta la posible existencia en las tablas de una clave primaria (columna PRIMARY KEY).
En caso de que alguna operación, de las que se describen más adelante, genere una tabla con tuplas repetidas, se deberán eliminar las tuplas repetidas, dejando tan sólo una de éstas. Note que de no hacerlo se estaría violando la definición de tabla.
Si la aplicación de alguna operación, de las que se describen más adelante, deja a las tablas de la base de datos en un estado inconsistente o la operación no está permitida por su especificación o por las reglas antes mencionadas se establecerá una situación de error en la cual:

- Permanecerá invariante el estado de la base de datos
- Se mostrará un mensaje de error adecuado y clarificador

A continuación se describen las operaciones del sistema.

Operaciones sobre la Base de Datos

(*) Crear Tabla

- ***createTable(nombreTabla)***

Descripción: Crea una nueva tabla vacía (sin columnas ni tuplas) en la base de datos con nombre: nombreTabla, siempre que no exista ya una tabla con dicho nombre. Esta operación deberá resolverse en $O(log_2(n))$ promedio, siendo n la cantidad de tablas de la base de datos.

Ejemplo: Crear 2 tablas llamadas Personas y Productos:

createTable ("Personas");
createTable ("Productos");

(*) Eliminar Tabla

- ***dropTable(nombreTabla)***

Descripción: Elimina la tabla de nombre nombreTabla de la base de datos, si éste existe, y las tuplas que la misma posee.

Ejemplo: Eliminar la tabla Productos:

dropTable ("Productos");

Operaciones para modificar una tabla

(*) Agregar Columna

- ***addCol (nombreTabla, nombreCol, tipoCol, calificadorCol)***

Descripción: Agrega a la tabla de nombre nombreTabla, si éste existe, una nueva columna al final de nombre nombreCol, si éste no existe, tipo tipoCol y calificador calificadorCol. Si la tabla tiene tuplas, el nuevo campo tendrá el valor EMPTY en cada tupla. Por lo tanto, en el caso en que la tabla tenga tuplas no es válido que se agregue un calificador distinto de ANY. Tampoco es válido que calificadorCol sea PRIMARY KEY si existe ya una columna con dicho calificador en la tabla nombreTabla.

Ejemplo: Crear 3 columnas en la tabla Personas llamadas Nombre, Apellido y CI.

addCol (“Personas”, “Nombre”, STRING, NOT_EMPTY);
addCol (“Personas”, “Apellido”, STRING, NOT_EMPTY);
addCol (“Personas”, “CI”, INTEGER, NOT_EMPTY);

Tabla Personas

Nombre	**Apellido**	**CI**

Eliminar Columna

- ***dropCol (nombreTabla, nombreCol)***

Descripción: Elimina de la tabla de nombre nombreTabla, si éste existe, la columna de nombre nombreCol, si éste existe. Si la tabla tiene tuplas, entonces se eliminará de éstas el campo correspondiente a la columna eliminada. Si la tabla tenía una única columna de nombre nombreCol entonces quedará como tabla vacía.

Ejemplo: Eliminar la columna Apellido.

dropCol (“Personas”, “Apellido”);

Tabla Personas

Nombre	**CI**

Modificar Columna

- ***alterCol (nombreTab, nombreCol, tipoColNuevo, calificadorColNuevo, nombreColNuevo)***

Descripción: Modifica de la tabla de nombre nombreTabla, si éste existe, la columna de nombre nombreCol, si éste existe, quedando ésta columna con el nuevo tipo de datos tipoColNuevo, calificador calificadorColNuevo y nombre nombreColNuevo, si éste último no es el nombre de otra columna de la tabla. Si la tabla tiene tuplas, los valores de la columna modificada deberán satisfacer las nuevas características (tipo de dato y calificador). El tipo de datos sólo puede cambiar de integer a string y en este caso se deberá realizar la conversión de tipo de la columna especificada en todas las tuplas de la tabla.

Ejemplo: Convertir la columna CI en PRIMARY KEY.

alterCol (“Personas”, “CI”, INTEGER, PRIMARY_KEY, “CI”);

Operaciones para la Edición de Datos

(*) Insertar Tupla

- ***insertInto(nombreTabla, columnasTupla, valoresTupla)***

Descripción: Inserta en la tabla de nombre nombreTabla, si éste existe, una tupla con los valores dados en valoresTupla para las columnas indicadas en columnasTupla, si los nombres de las columnas existen, los valores son del tipo adecuado y satisfacen los calificadores correspondientes a cada columna. Si no se indican todas las columnas se inserta EMPTY en las otras. Por lo tanto, la operación se permite sólo si las columnas que no se indican tienen el calificador ANY. Los nombres de las columnas en columnasTupla y los valores de valoresTupla se separan con el uso del caracter dos puntos (:) y deben corresponderse uno a uno. Esto es, el nombre de columna *i* en columnasTupla con el valor en la posición *i* de valoresTupla. Si la tupla a insertar pertenece a la tabla, la operación no tendrá efecto.

Ejemplo: Insertar tuplas en la tabla Personas.

insertInto (“Personas”, “Nombre:CI”, “Telma:3333111”);
insertInto (“Personas”, “Nombre:CI”, “Jose:2566499”);
insertInto (“Personas”, “Nombre:CI”, “Juan:4232323”);
insertInto (“Personas”, “CI:Nombre”, “1555000:Pepe”);
insertInto (“Personas”, “CI:Nombre”, “2565000:Maria”);

Tabla Personas

Nombre	**CI**
Telma	3333111
Jose	2566499
Juan	4232323
Pepe	1111111
Maria	2565000

(*) Eliminar Tupla

- ***deleteFrom(nombreTabla, condicionEliminar)***

Descripción: Elimina de la tabla de nombre nombreTabla, si éste existe, todas las tuplas que cumplen la condición condiciónEliminar. La condición respeta el formato descrito para condiciones.

Formato de las condiciones

La condición se aplica a una serie de valores, lo que permite seleccionar una serie de tuplas de una tabla. Si la condición es vacía “”, se seleccionan todas las tuplas.
El formato de las condiciones es: *columna operador valor* (sin espacios en blanco intermedios). Los operadores a utilizar son: **=** “igual”, **<>** “Distinto”, **>** “Mayor”, < “Menor” y * “Igual Prefijo”.
Para comparar strings con el operador > o < se utilizará el orden lexicográfico habitual.
Por ejemplo, Sexo=Masculino, Edad<18, Código<>20, Apellido>Perez.

El operador * “Igual Prefijo” se puede utilizar solamente en columnas que sean PRIMARY KEY. Una condición que contiene el operador * resulta verdadera para una tupla si, y sólo si, el valor de la columna en dicha tupla empieza con el prefijo especificado.
Por ejemplo, CI*2256 (Todas las cédulas que comienzan con este número), Apellido*Per (Todos los apellidos que comienzan con Per; Ej: Perez, Peralta).

El valor EMPTY puede usarse en una condición. La condición c*olumna*=EMPTY resulta verdadera para una tupla si, y sólo si, el valor de la columna en dicha tupla es EMPTY; c*olumna*<>EMPTY resulta verdadera para una tupla si, y sólo si, el valor de la columna en dicha tupla es distinto de EMPTY. En cualquier otro caso, una condición que involucre el valor EMPTY resulta ser falsa. Asimismo, toda condición que no involucre al valor EMPTY resultará falsa al ser instanciada por una tupla que tenga el valor EMPTY en la columna de la condición.

Ejemplo: Eliminar todas las CI que comiencen en 256.

deleteFrom (“Personas”, “CI*256”);

Tabla Personas

Nombre	**CI**
Telma	3333111
Juan	4232323
Pepe	1111111

(*) Modificar Tupla

- ***update(nombreTabla, condicionModificar, columnaModificar, valorModificar)***

Descripción: Modifica en la tabla de nombre nombreTabla, si éste existe, el valor de las tuplas en la columna de nombre columnaModificar, si éste existe, que cumplen la condición condiciónModificar. En la columna especificada de las tuplas que cumplen la condición se asigna el valor valorModificar, siempre que este valor sea del tipo adecuado y satisfaga el calificador de la columna especificada. La condición respeta el formato descrito para condiciones.

Ejemplo: Modificar la CI de Pepe.

update (“Personas”, “Nombre=Pepe”, “CI”, “1555000”);

Tabla Personas

Nombre	**CI**
Telma	3333111
Juan	4232323
Pepe	1555000

Operaciones entre Tablas

(*) Selección

- ***selectWhere (nombreTabla1, condicion, nombreTabla2)***

Descripción: Dado un nombre de tabla nombreTabla1 y una condición, genera una nueva tabla en la base de datos de nombre nombreTabla2, si nombreTabla1 existe y nombreTabla2 no existe, con las tuplas de la tabla nombreTabla1 que cumplan la condición. La condición respeta el formato descrito para condiciones. La tabla resultado posee los mismos tipos de datos y calificadores para sus columnas que los correspondientes a la tabla parámetro.

Ejemplo: Copiar la tabla Personas a una nueva tabla llamada Personas2.

selectWhere (“Personas”, “”, “Personas2”);

Tabla Personas2

Nombre	**CI**
Telma	3333111
Juan	4232323
Pepe	1555000

(*) Proyección

- ***Select (nombreTabla1, columnas, nombreTabla2)***

Descripción: Dado un nombre de tabla nombreTabla1 y uno o más nombres de columnas de dicha tabla especificados en el parámetro columnas, genera una nueva tabla en la base de datos de nombre nombreTabla2, si nombreTabla1 existe, nombreTabla2 no existe y si todas las columnas especificadas pertenecen a tabla de nombre nombreTabla1. La tabla nombreTabla2 tendrá las tuplas de la tabla nombreTabla1 sólo con las columnas especificadas, manteniendo el orden establecido en el parámetro columnas. Los nombres de las columnas se separan con el uso del caracter dos puntos (:).

Ejemplo: Copiar la columa CI de la tabla Personas a una nueva tabla llamada Cedulas.

Select ("Personas", "CI", "Cedulas");

Tabla Cedulas

CI
3333111
4232323
1555000

(*) Join Natural

- ***Join (nombreTabla1, nombreTabla2, nombreTabla3)***

Descripción: Dadas dos tablas de nombres nombreTabla1 y nombreTabla2, tales que ambas tengan *exactamente una única* columna en común con el mismo nombre y tipo de datos, calificada además como PRIMARY KEY, genera una nueva tabla en la base de datos de nombre nombreTabla3, si nombreTabla1 y nombreTabla2 existen y nombreTabla3 no existe. La nueva tabla nombreTabla3 tendrá todas las columnas de la tabla nombreTabla1 y las columnas de nombreTabla2 excepto la que esté calificada como PRIMARY KEY (de modo de no repetirla), en ese orden. Asimismo, nombreTabla3 tendrá exclusivamente las tuplas que satisfagan la igualdad de las claves de ambas tablas.

Ejemplo: Considere la tabla Personales y la tabla Profesiones que sigue (con CI su clave primaria). El join natural de ambas tablas, presentadas abajo, genera la tabla Pers_Prof que se muestra a continuación:

Join ("Personas", "Profesiones", "Pers_Prof");

Tabla Personas

Nombre	**CI**
Telma	3333111
Juan	8232323
Pepe	1555000

Tabla Profesiones

CI	**Cargo**
3333111	Dentista
7777777	Escribano
8232323	Ingeniero

CI es clave (PRIMARY KEY) en las tres tablas.

Tabla Pers_Prof

Nombre	**CI**	**Cargo**
Telma	3333111	Dentista
Juan	8232323	Ingeniero

Operaciones para la Impresión de Información

(*) Listar Tablas

- ***printTables()***

Descripción: Imprime los nombres de las tablas de la base de datos del sistema, ordenados alfabéticamente de menor a mayor.

Ejemplo: Imprimir las tablas.

printTables ();

Cedulas
Personas
Personas2
Profesiones
Prof_Pers

(si éstas fueran las únicas 5 tablas de la base de datos)

(*) Listar Esquema

- ***printMetadata(nombreTabla)***

Descripción: Imprime el esquema de la tabla de nombre nombreTabla, si éste existe. Es decir, imprime el nombre de la Tabla, los nombres de sus columnas en el orden correspondiente, indicando para cada columna su tipo de datos y calificador.

Ejemplo: Mostrar el esquema de la tabla Personas.

printMetadata ("Personas");

Personas

Nombre - STRING - NOT_EMPTY
CI - INTEGER - PRIMARY KEY

(*) Listar Tabla

- ***printDataTable (nombreTabla, ordenadaPor)***

Descripción: Imprime las tuplas de la tabla de nombre nombreTabla, si éste existe, ordenados de acuerdo a las columnas especificadas en el parámetro ordenadaPor. Los nombres de las columnas se expresan en el formato $columna_{i1}$:$columna_{i2}$:…:$columna_{ik}$. Las tuplas se muestran ordenadas por $columna_1$ (de menor a mayor) y si dos tuplas coinciden en el valor de la columna $columna_{i1}$, entonces éstas se ordenan (de menor a mayor) por el campo $columna_{i2}$, y así sucesivamente según los campos especificados, de izquierda a derecha, en ordenadaPor (que pueden ser un subconjunto de los campos de la tabla). Considerar que en la ordenación para la impresión, de menor a mayor, de los valores de una columna, los valores EMPTY deben aparecer al final, luego de los valores no vacíos.
Si el parámetro ordenadaPor es "", imprime las tuplas en cualquier orden, salvo que la tabla tenga un campo PRIMARY KEY, en cuyo caso las tuplas se imprimen ordenadas (de menor a mayor) según dicho campo.

Ejemplo: Mostrar las tuplas, de la tabla Productos que sigue, ordenadas por Cantidad y luego por Descripción.

Tabla Productos

ID	Descripcion	Cantidad
1	Marcador	50
2	Engrampadora	99
3	Boligrafo	50

printDataTable (“Productos”, “Cantidad:Descripcion”);

Productos

ID:Descripcion:Cantidad

3:Boligrafo:50
1:Marcador:50
2:Engrampadora:99

Operaciones Adicionales del Sistema

Recuperar Tabla

- ***undelete()***

Descripción: Permite recuperar la última tabla eliminada. La cantidad máxima de tablas que se podrán recuperar será un valor constante a determinar (RECUPERAR_MAX). Esto es, el sistema guarda en cada instante a lo sumo las RECUPERAR_MAX últimas tablas eliminadas. Las tablas eliminadas anteriores a éstas no podrán ser recuperados nunca más. Si al recuperar una tabla existe otra con el mismo nombre, el comando quedará sin efecto y la tabla se eliminará definitivamente.

Tabla Modificadas

- ***recent(k)***

Descripción: Imprime los nombres de, a lo sumo, las últimas k tablas *diferentes* modificadas a través de las operaciones de edición de datos (únicamente). Se lista primero la tabla más recientemente modificada, luego la segunda más recientemente modificada y así sucesivamente.

Las operaciones marcadas con (*) son obligatorias

El plazo de la entrega es hasta el viernes 31 de octubre del 2014 a las 23 59.