Base de Datos I
Trabajo Práctico Especial
2do Cuatrimestre 2018

1. Objetivo

El objetivo de este Trabajo Práctico Especial es aplicar los conceptos de SQL Avanzado (PSM, Triggers) vistos a lo largo del curso, para implementar funcionalidades y restricciones no disponibles de forma estándar (que no pueden resolverse con Primary Keys, Foreign Keys, etc.).

2. Modalidad

El Trabajo Práctico estará disponible en el Campus a partir del 08/11/2018, indicándose allí mismo la fecha de entrega.

Se incluyen junto con el enunciado los archivos: **User.tsv, Badges.tsv** tomados del sitio StackExchange, https://archive.org/details/stackexchange, los cuales fueron transformados para adaptarlos a este trabajo.

El TP deberá realizarse en grupos de 4 alumnos y entregarse a través de la plataforma Campus ITBA hasta la fecha allí indicada.

3. Descripción del Trabajo

El sitio StackExchange, https://archive.org/details/stackexchange, ofrece datasets con los intercambios de los usuarios de StackOverflow en formato XML. Dichos datos están agrupados por temas, en archivos comprimidos.

Cada uno de esos archivos tiene cuatro documentos XML:

- **Users.xml**: contiene la información básica de algunos usuarios registrados en StackOverflow. Cada usuario está identificado con el atributo **id**
- **Badges.xml**: son las placas obtenidas por los usuarios. Los usuarios están identificados con el atributo **userId** y en atributo **name** se encuentra el nombre del badge
- **Posts.xml**: son las preguntas que publican los usuarios
- **Comments.xml**: son los comentarios que hacen los usuarios a algún post en particular

Para este trabajo, hemos transformado mediante plantillas XSLT los dos primeros archivos XML (Users y Badges), a archivos **tsv** (Tab Separated Values – similar al csv pero separados por Tabs en lugar de coma). Los nombres de las columnas se corresponden con los nombres de los atributos de los documentos XML.

La finalidad de este Trabajo Práctico Especial consiste en migrar los datos de los archivos tsv a una base de datos, producir un reporte y realizar algunas validaciones. Específicamente se debe hacer lo siguiente:

a) Crear las tablas apropiadas para importar los datos
b) Importar los datos
c) Crear un reporte
d) Recalcular automáticamente la reputación de los usuarios afectados por un cambio en su valor de votos positivos o negativos

a) Creación de las tablas.

Deben crearse dos tablas: **UserTP** y **Badge** que serán las receptoras de datos de los archivos **users.tsv** y **badges.tsv** respectivamente. Los campos y restricciones de las tablas deben crearse en base al análisis de los datos. Recordar que los archivos tsv son archivos de texto que pueden abrirse fácilmente con cualquier editor. Se recomienda que los nombres de los campos coincidan con los de las columnas de los tsv.

En base a los datos, se deben crear las claves apropiadas y las referencias necesarias.

b) Importación de los datos

Utilizando el comando COPY de PostgreSQL, se deben importar TODOS los datos de los archivos **tsv** en las tablas creadas en **a)**. Los datos de los archivos tsv provistos por la cátedra NO pueden ser modificados.

En el apéndice 1) IMPORTACION/EXPORTACION se encuentra un detalle de comandos de PostgreSQL que permiten realizar este proceso.

c) Reporte.

Luego de alguna interacción, los usuarios pueden recibir badges de StackOverflow. Hay 3 clases de badges: Gold, Silver y Bronze, y esta información se encuentra inicialmente en el archivo **badges.tsv**. La clase del Badge está en el atributo *class* y toma los valores 1, 2 o 3 los cuales se corresponden con Gold, Silver y Bronze respectivamente.

Se pide crear la función **ReporteBadge(usuarioDesde, usuarioHasta)** que recibe como parámetros dos *id* de usuario y, tomando los datos importados en **b)**, genere un reporte mostrando para cada usuario cuyo Id se encuentre en el intervalo [usuarioDesde,usuarioHasta], su *ID, nombre, reputación* y una *lista de sus badges* ***distintos*** junto con la cantidad de cada uno de ellos.

El reporte tendrá las siguientes características:

I. Encabezado con título "**`BADGES REPORT`**"

II. Encabezado de columnas:

```
"ID   Display Name    Reputation   Badge Name   Qtty"
```

III. Por cada usuario tiene que aparecer un renglón en el reporte, con el ID, el nombre y la reputación del usuario. Se debe mostrar para ese usuario la lista de badges distintos **ordenados alfabéticamente**. El primer badge y su cantidad de apariciones, deben estar en el **mismo** renglón que los datos del usuario y el resto encolumnados a continuación en los renglones subsiguientes

IV. Al finalizar los datos del usuario se debe indicar el total de badges de clase GOLD, el total de badges de clase SILVER y el total de badges de clase BRONZE obtenidos por dicho usuario

En caso de que no existieran datos para los parámetros ingresados, no se debe mostrar nada (ni siquiera el encabezado).

La función debe manejar los posibles errores.

Para saber cómo escribir el reporte ver el apéndice 2) Cómo escribir por pantalla.

Por ejemplo,

- si invocamos **ReporteBadge(4,5)** se debe obtener:

BADGES REPORT

ID	Display Name	Reputation	Badge Name	Qtty
4	adrian	1929	Autobiographer	1
			Precognitive	1
			Supporter	1
			GOLD Badges:	0
			SILVER Badges:	0
			BRONZE Badges:	3
5	Stefano Borini	1909	Autobiographer	1
			Beta	1
			Commentator	1
			Critic	1
			Custodian	1
			Editor	1
			Famous Question	1
			Good Answer	1
			Good Question	3
			Nice Answer	1
			Nice Question	8
			Notable Question	2
			Popular Question	5
			Precognitive	1
			Quorum	1
			Scholar	1
			Student	1
			Supporter	1
			Taxonomist	1
			Teacher	1
			Yearling	6
			GOLD Badges:	1
			SILVER Badges:	6
			BRONZE Badges:	14

- Si invocamos **ReporteBadge(1,1)** no se obtiene nada debido que el usuario con id=1 no existe en estos datos

d) Recálculo de *Reputation*

Tal como se explica en https://stackoverflow.com/help/whats-reputation, la reputación de los usuarios se calcula utilizando varios parámetros. En nuestro caso queremos recalcular la reputación de los usuarios cuando se produzca un cambio en los *UpVotes* o en los *DownVotes* del usuario sabiendo que 1(un) UpVote suma 5 puntos a la reputación y 1(un) DownVote le descuenta 2 puntos.

Hay que tener en cuenta además, que la reputación nunca puede valer menos que 1 y que el valor de UpVotes y DownVotes nunca puede ser menor que 0. En caso de que, luego de la modificación, la cantidad de UpVotes resultara menor que 0, para calcular el nuevo valor de Reputation se tomará el valor anterior de UpVotes (previo a la modificación), debido a que no se pueden descontar más que esos votos. Lo mismo ocurre con DownVotes

Se pide crear los triggers necesarios de modo tal que al modificarse el valor de *UpVotes* o *DownVotes* en la tabla que contiene los datos del usuario(**UserTP**), se

actualice el correspondiente valor de *Reputation* en base a lo explicado en el párrafo anterior.

Ejemplo

El usuario con *id* = 5, Stefano Borini, tiene un valor de *Reputation* = 1929, de *UpVotes*=20 y de *DownVotes* = 1.

Si ejecutamos:

A) `UPDATE usertp SET upvotes=upvotes + 2 WHERE id=5;`

El usuario queda con los siguientes valores

id	displayname	reputation	upvotes	downvotes
5	Stefano Borini	1939	22	1

B) Si sobre el resultado de A) ejecutamos:
`UPDATE usertp SET upvotes=20 WHERE id=5;`

Obtenemos:

id	displayname	reputation	upvotes	downvotes
5	Stefano Borini	1929	20	1

C) Si sobre el resultado de B) ejecutamos:
`UPDATE usertp SET upvotes=upvotes - 200 WHERE id=5;`

Obtenemos:

id	displayname	reputation	upvotes	downvotes
5	Stefano Borini	1829	0	1

D) Si sobre el resultado de C) ejecutamos:
`UPDATE usertp SET downvotes=downvotes+6 WHERE id=5;` obtenemos

id	displayname	reputation	upvotes	downvotes
5	Stefano Borini	1817	0	7

E) Si consultamos UserTP de esta forma:

```
SELECT id, displayname, reputation, upvotes, downvotes
FROM usertp WHERE id>=10 AND id <15;
```

obtenemos:

id integer	displayname text	reputation integer	upvotes integer	downvotes integer
10	Lev Reyzin	4985	383	6
11	Shamim Hafiz	101	0	0
12	Lars Kotthoff	4291	9	3
13	Ben Webster	19840	321	70
14	jurassic	981	86	3

Al ejecutar:
`UPDATE usertp SET downvotes=downvotes-5 WHERE id>=10 AND id <15;`

obtenemos:

id	displayname	reputation	upvotes	downvotes
10	Lev Reyzin	4995	383	1
11	Shamim Hafiz	101	0	0
12	Lars Kotthoff	4297	9	0
13	Ben Webster	19850	321	65
14	jurassic	987	86	0

4. Entregables

Los alumnos deberán entregar los siguientes documentos:

- El script sql **funciones.sql** con el código necesario para crear las tablas, las funciones y los triggers

- Un informe que debe contener:
  - El rol de cada uno de los participantes del grupo. Si bien en el TP deben estar involucrados todos los integrantes, se debe asignar un rol de supervisión de cada una de las tareas. Mínimamente los roles son: encargado del informe, encargado de las funciones, encargado del trigger, encargado del funcionamiento global del proyecto y encargado de investigación. Pueden asignarse más roles en caso de requerirse
  - Todo lo investigado para realizar el TP
  - Las dificultades encontradas y cómo se resolvieron
  - También se debe detallar aquí el proceso de importación de los datos realizado
  - El informe debe tener como máximo 3 páginas

5. Evaluación

La evaluación del trabajo se llevará a cabo utilizando los parámetros establecidos en la rúbrica asociada a la actividad en el Campus.

Se tendrá en cuenta que las consultas, más allá del funcionamiento (lo cual es fundamental), sean genéricas.

Los docentes ejecutarán el proceso usando los conjuntos de datos entregados pero podrán también hacer pruebas con otros conjuntos de datos de similares características para evaluar el funcionamiento en distintos escenarios.

El informe deberá estar completo y sin faltas de ortografía.

En caso de que el trabajo no cumpliera los requisitos básicos para ser aprobado, los alumnos serán citados en la fecha de recuperatorio para defenderlo y corregir los errores detectados.

Apéndices

1) IMPORTACION / EXPORTACION

Para importar datos a nuestras tablas a partir de archivos de texto o csv (comma value separated), PostgreSQL utiliza el comando COPY...FROM.

Existe también el comando COPY…TO que, de manera análoga, exporta el contenido de una tabla a un archivo estándar del file system.

La sintaxis es la siguiente;

```
COPY table_name [ ( column_name [, ...] ) ]
    FROM { 'filename' | PROGRAM 'command' | STDIN }
    [ [ WITH ] ( option [, ...] ) ]

COPY { table_name [ ( column_name [, ...] ) ] | ( query ) }
    TO { 'filename' | PROGRAM 'command' | STDOUT }
    [ [ WITH ] ( option [, ...] ) ]

donde option puede ser:

    FORMAT format_name
    OIDS [ boolean ]
    DELIMITER 'delimiter_character'
    NULL 'null_string'
    HEADER [ boolean ]
    QUOTE 'quote_character'
    ESCAPE 'escape_character'
    FORCE_QUOTE { ( column_name [, ...] ) | * }
    FORCE_NOT_NULL ( column_name [, ...] )
    FORCE_NULL ( column_name [, ...] )
    ENCODING 'encoding_name'
```

Alguna de las opciones más comunes:

table_name
Nombre de una tabla existente (opcionalmente prefijada con el nombre del esquema).

column_name
Lista opcional de columnas a copiar. Si no se especifican, se copian todas las columnas de la tabla.

query
Una consulta (SELECT o VALUES) cuyo resultado será copiado. Esta consulta debe estar encerrada entre paréntesis.

filename
El nombre (incluyendo el path) del archivo de entrada o salida. El path del archivo de entrada puede ser absoluto o relativo. El path del archivo de salida, debe ser absoluto.

`FORMAT`
Selecciona el formato de los datos a ser leídos o escritos: `text`, `csv` (Comma Separated Values), o `binary`. El default es `text`.

`DELIMITER`

Determina el caracter que separa las columnas dentro de cada fila. En los archivos de texto, el default es el tab, en los CSVs, la coma. Tiene que ser un único de character de un byte. Esta opción no se permite para el format `binary`.

`HEADER`

Especifica que el archivo contiene encabezado con los nombres de las columnas. En modo salida, la primera línea contiene los nombres de las columnas de la tabla y en la entrada, la primera línea sólo se ignora. Esta opción sirve únicamente para el formato `CSV`.

`QUOTE`

Especifica qué carácter se usa como comillas cuando el valor del dato está encomillado. El default es la comilla doble. Debe ser un carácter de un byte. Esta opción sirve únicamente para el formato `CSV`.

IMPORTANTE: como para usar COPY hay que tener privilegios de Superuser, vamos a utilizar la opción \COPY por consola, en la cual se pueden utilizar los parámetros antes mencionados

Por ejemplo

\COPY –u username nombretabla FROM arch.csv csv header delimiter ';'

Por ejemplo, si quisiéramos importar el archivo proveedores.csv a la tabla del mismo nombre deberíamos ejecutar:

```
\copy proveedor to proveedor.csv header delimiter ';' csv;
```

Estamos aquí indicando que los campos están separados por punto y coma y que la primera línea corresponde a los nombres de las columnas.

Este comando debe estar incluido en un script sql (import.sql por ejemplo) que ejecutamos desde consola:

```
psql –h bd1.it.itba.edu.ar -U nombreusuario –f import.sql PROOF
```

2) Cómo escribir en pantalla

En el servidor bd1.it.itba.edu.ar tenemos instalada la extensión **dbms_output**. Este paquete de Oracle agrega funcionalidad a PostgreSQL permitiéndole escribir por pantalla. Las funciones disponibles son:

Funciones

dbms_output.enable([buffer_size int4])	Start dbms_output support, elective buffer_size set up maximum buffer storage size
dbms_output.disable()	Deactivate dbms_output support, put, put_line, new_line commands have no effect
dbms_output.serveroutput(bool)	Start client output display demand, start dbms_output at the same time
dbms_output.put(text)	Insert text in output buffer
dbms_output.put_line(text)	Insert line (text with end of line symbol)
dbms_output.new_line()	Insert end of line symbol

Por ejemplo, para imprimir por pantalla 'Hello World!', se debe ejecutar:

```
DO $$
BEGIN
        PERFORM DBMS_OUTPUT.DISABLE();
        PERFORM DBMS_OUTPUT.ENABLE();
        PERFORM DBMS_OUTPUT.SERVEROUTPUT ('t');
        PERFORM DBMS_OUTPUT.PUT_LINE ('Hello World!');
END; $$
```

Se recomienda hacer este trabajo en el servidor y no localmente dado que esta extensión no se encuentra disponible por default.