ADMINISTRACIÓN DE BASE DE DATOS CON POSTGRESQL LABORATORIO 2. APLICACIÓN DE RESTRICCIONES

Luis Antonio Álvarez Oval
loval@unach.mx

Universidad Autónoma de Chiapas

Para citar este artículo:
Álvarez, L. (2015) Administración de base de datos con postgreSQL laboratorio 2. Aplicación de restricciones. *Espacio I+D Innovación más Desarrollo 4* (7) 196-220. Recuperado de http://espacioimasd.unach.mx/suplemento/espacioimasd_7_espanol.pdf

En esta segunda entrega de la serie de laboratorios de Administración de Base de Datos (ABD) se enseña la forma en que se aplican restricciones a los campos en una tabla de la base de datos. Los laboratorios se han diseñado para proporcionar los conceptos y la experiencia necesarios para conocer detalladamente el sistema, se aprovecha la función de "copiar y pegar" que nos ofrece el sistema operativo Windows para disminuir el esfuerzo del lector en la preparación del ambiente de trabajo y en la solución de los problemas. En la sección denominada "trabajo adicional" se requiere que el lector aplique la experiencia obtenida en la solución de problemas relacionados al tema central del laboratorio. La sección de conceptos básicos muestra la sintaxis de los comandos y da algunas explicaciones del uso de los mismos, este material ha sido tomado del manual de usuario del sistema PostgreSQL el cual está disponible en la página oficial de la herramienta, en algunos casos se ha tomado del sitio oficial en Español. Los conceptos básicos se aplican en torno a un proyecto que se denomina "Universidad ACME", el cual es producto de la imaginación del autor, así como la solución práctica de los problemas planteados. Los libros que se ofrecen en la sección de referencias, sirven como consulta para apoyar algunos de los conceptos que se aplican en la solución práctica de problemas de administración de base de datos.

Estos laboratorios se han preparado para procurar experiencia práctica a los estudiantes de la materia Administración de Base de Datos de la Licenciatura en Sistemas Computacionales que se ofrece en la Facultad de Contaduría Pública (FCP) del Campus IV de la Universidad Autónoma de Chiapas (UNACH). En la FCP se tienen al menos 14 años de experiencia en el uso de PostgreSQL en las aulas, proyectos de investigación y en sistemas que se han implementado para la automatización de las actividades cotidianas de la FCP. Como producto de esa experiencia académica e industrial se han obtenido estos laboratorios que se usan en las aulas para capacitar a nuestros estudiantes. Hemos

encontrado que los estudiantes se motivan al estudio cuando se concretan en estos ejercicios las ideas abstractas que se explican en las aulas, aunque ese será tema de otro artículo que será publicado en esta revista. También se tiene noticia de que son una fuente de consulta para egresados que laboran en el sector empresarial.

Como se ha mencionado previamente, la herramienta tiene características y lenguajes de programación estándar que ofrecen sistemas propietarios, por lo que los ejemplos fácilmente pueden ser aplicados en otros sistemas de bases de datos del mercado, o pueden ser referencia para aplicar los conceptos en proyectos industriales. Por lo que puedan servir como consulta a profesionales de las Ciencias de la Computación.

OBJETIVO

El lector aprenderá a restringir los campos de una Base de Datos usando los comandos que el SQL tiene para este fin en el sistema de administración de base de datos PostgreSQL.

PRERREQUISITOS

Se espera que el lector tenga experiencia previa en el uso y conversión de diagramas Entidad-Relación (E-R), los temas asociados al Diseño de Base de Datos no se cubren en este documento. También se espera que el usuario tenga conocimientos básicos del lenguaje de programación denominado SQL.

Es necesario instalar la base de datos PostgreSQL versión 9.3 sobre el sistema operativo Windows, verifique los requerimientos para instalación en la página oficial de la herramienta: www. postgresql.org. El sistema puede descargarse del sitio Web:

http://www.enterprisedb.com/products-services-training/pgdownload#windows

Si tiene alguna duda con respecto a PostgreSQL, le recomiendo visitar el sitio oficial con información publicada en idioma español: http://www.postgresql.org.es/primeros_pasos.

PARTES QUE COMPONEN ESTE LABORATORIO

1. Proyecto a desarrollar
2. Conceptos básicos
3. Preparación del ambiente de trabajo
4. Problemática a resolver
5. Trabajo adicional
6. Referencias

1. PROYECTO A DESARROLLAR

El ejercicio que se va a realizar consiste en un proyecto que describe el problema de una empresa dedicada a la prestación de servicios educativos: después de leer el texto se genera el diagrama E-R con la solución a éste problema, se continúa con la creación de las tablas y población de las tablas, para finalmente trabajar con los permisos de grupos y usuarios.

Proyecto Universidad ACME

En UACME, se ofrecen dos tipos de cursos en el periodo especial de verano, en el cual se imparten cursos de verano y cursos extracurriculares. Los primeros son materias que un alumno regular que estudia una carrera cursa en este periodo, se le permite adelantar hasta dos materias; mientras que los segundos son

cursos especiales de capacitación que se ofrecen a alumnos regulares como estudiantes o profesionistas externos.

Los docentes de la UACME, son los únicos a los que se les permite impartir estos cursos, por los cuales recibe un pago adicional, se les paga de acuerdo a un tabulador que indica el costo de la hora de estos cursos de acuerdo al nivel académico del docente. El pago se genera a partir del alta del curso y solo se permite expedir un cheque por cada curso. Además los estudiantes deben acudir a pagar adicionalmente al costo del semestre por asistir a ellos.

UACME tiene dos departamentos que intervienen en la administración de los cursos:

A) Departamento de Administración (DA) y B) Departamento de Control Escolar (DCE). Corresponde al DA, efectuar el pago a los docentes y los cobros a los alumnos. El DA es dirigido por el C.P. Ávila y es auxiliado por el Sr. Cancino. Mientras que el DCE, es dirigido por el Lic. Barroso y auxiliado por los Sras. Tirado, Martínez, Aquino y Ramos y es en este donde se decide cuales cursos se imparten en el periodo, quién los imparte, y se aceptan las solicitudes de los alumnos. Un caso especial, es el de los Profesores, ya que el DA es quién les puede modificar el sueldo quincenal, mientras que el DCE ni siquiera puede visualizar éste. Lo curioso radica en que, es el DCE quién acepta los docentes y los registra en el sistema, pero es el DA donde se captura el sueldo. Importante es para la administración de la UACME que esta política se aplique al pie de la letra, y que sea implementado directamente sobre la DB. A continuación se describe detalladamente las tablas a las cuales tiene acceso el personal de cada

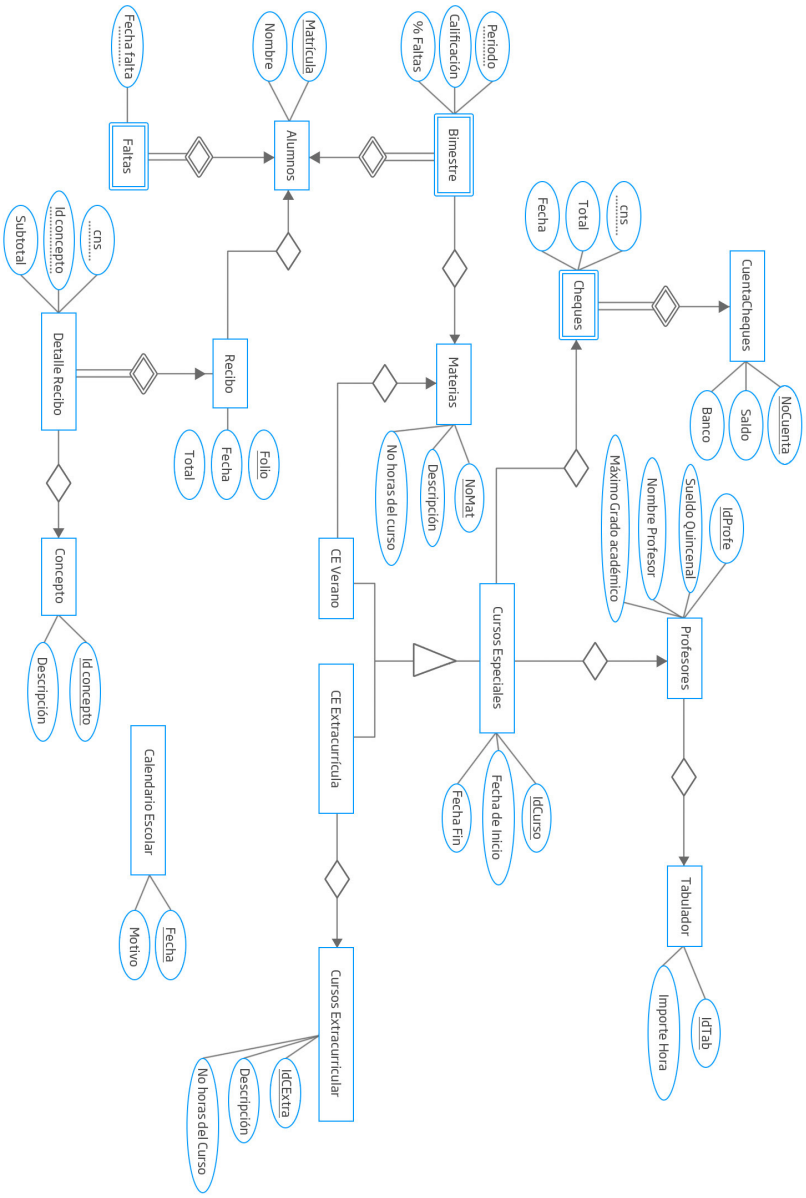
Tablas a las que se le permite el acceso al personal de la Secretaría Administrativa: CuentaCheques, Cheque, Tabulador, Profesores, Concepto, Recibo, y DetalleRecibo.

Como casos especiales este departamento podrá acceder a consultar las tablas de Cursos Especiales, Cursos Especiales Verano, Cursos Especiales Extracurriculares, Cursos Extracurriculares

y Materias. Explícitamente no se les permite modificar ningún campo o registro.

Tablas a las que se le permite el acceso al personal de la Secretaría Escolar: CursosEspeciales, CursosExtracurricular, Materias, CEVerano, CEExtracurricula, Alumnos, Bimestre, Faltas, CalendarioEscolar.

Figura 1. Diagrama E/R que resuelve el problema anterior.



2. CONCEPTOS BÁSICOS

Aquí encontrará una versión modificada del manual de usuario de PostgreSQL que da una explicación del uso y la sintaxis de los comandos usados en el presente laboratorio. Para consultar el manual oficial en idioma inglés visite el sitio oficial de la misma en Internet: www.postgresql.org

Restricciones (CONSTRAINTS)

Éstas permiten usar restringir los datos, de tal modo que ayudan a prevenir que datos inválidos entren a la base de datos. El definir un tipo de datos para una columna es una restricción. Por ejemplo, una columna que tiene una restricción de tipo DATE, la columna valida las fechas.

USO DE NOT NULL
La restricción NOT NULL evita valores NULOS que aparecen en las columnas. La inserción de un valor NULL, o un INSERT que podría reunir una columna 2 con valor NULL, causarían que el INSERT falle.

UNIQUE
La restricción UNIQUE previene valores duplicados que se almacenan en la misma en la columna. CREATE TABLE despliega el nombre del índice único creado. Si una restricción UNIQUE es de múltiples campos, debe separar el UNIQUE con una línea para especificar las columnas que componen la restricción.

PRIMARY KEY (Llave Primaria)
Una llave primaria indica que una columna o grupo de columnas puede ser usado como un identificar único de renglones en la tabla. (Esto es una consecuencia directa de la definición de llave

primaria. Dese cuenta que una restricción única por sí misma no provee un identificador único porque no excluye valores nulos).

Al agregar una llave primaria automáticamente se crea un índice *btree* único sobre la columna o grupo de columnas usadas en la llave primaria.

Una tabla puede tener cuando mucho una llave primaria. (Puede haber cualquier cantidad de restricciones únicas y no-nulas, las cuales funcionalmente son lo mismo, pero solo una puede ser identificada como llave primaria). La teoría de las bases de datos relacionales dicta que cada tabla debe tener una llave primaria. Esta regla no es impuesta por Postgresql, pero es mejor seguirla.

Foreign Key/References (Llave foránea/Referencia)

Una restricción de llave foránea especifica que los valores en una columna (o un grupo de columnas) deben de coincidir con los valores que aparecen en alguna columna de otra columna. Decimos que esto mantiene la integridad referencial entre dos tablas relacionadas.

RESTRICT y CASCADING DELETES son dos de las opciones más comunes. RESTRICT evita el borrado de un renglón referenciado. NO ACTION significa que si todavía existen filas de referenciadas cuando se comprueba la restricción, se produce un error; este es el comportamiento por omisión si no se especifica nada. (La diferencia en esencia entre estas dos opciones es que NO ACTION permite que la validación sea diferida hasta más tarde en la transacción, mientras que RESTICT no). CASCADE especifica que cuando un renglón referenciado es borrado, los renglones referenciados deben ser borrados automáticamente también. Hay otras dos opciones: SET NULL y SET DEFAULT. Éstas causan que las columnas referenciadas en los renglones referenciados sean inicializados con nulos o sus valores por omisión, respectivamente, cuando el renglón referenciado es borrado. Note que éstas no lo excusan de observar cualquier restricción. Por ejemplo,

si una acción específica SET DEFAULT pero el valor por omisión no satisface la restricción de llave foránea, la operación fallará.

De forma análoga a ON DELETE hay también ON UPDATE la cual es invocada cuando una columna referenciada es cambiada (actualizada). Las acciones posibles son las mismas. En este caso CASCADE significa que los valores actualizados de las columnas referenciadas deben ser copiados en los renglones referenciados.

Normalmente, un registro referenciado no necesita satisfacer la restricción de llave foránea si cualquiera de sus columnas referenciadas es nula. Si la coincidencia de nulos es agregada a la declaración de llave foránea, un renglón de referencia escapa satisfaciendo la restricción solo si todas las columnas referenciadas son nulas (por lo que una mezcla de valores nulos y no-nulos se garantiza que falle una restricción MATCH FULL). Si no quiere referencias renglones para poder evitar el satisfacer la restricción de llave foránea, declare las columnas referenciadas como no-nulas.

Una llave foránea de columnas referenciadas que ya sean una llave primaria o formen una restricción única. Esto significa que las columnas referenciadas siempre tienen un índice (la que subyace a la llave primaria o una restricción única); para así controlar si un renglón referenciado que tiene una coincidencia sea eficiente. Puesto que el borrado de un renglón de una tabla referenciada o la actualización de un renglón de una tabla referenciada requiere de la búsqueda en la tabla referenciada por renglones coincidentes con el viejo valor, también casi siempre es buena idea indexar las columnas referenciadas. Puesto que esto no siempre es necesario hay muchas opciones disponibles en como indexar, la declaración de una restricción de llave foránea no crea automáticamente un índice de las columnas referenciadas.

Modificación de la llave primaria

Si la restricción de una llave foránea referencia a una fila como su llave primaria, y la fila de la llave primaria es actualizada o eliminada, entonces la acción por omisión de la llave foránea es restringir la operación. Las opciones de llave foránea ON UPDATE y ON DELETE, sin embargo, permiten tomar una acción diferente.

Las opciones ON UPDATE y ON DELETE pueden tener las siguientes acciones:

- CASCADE UPDATE actualiza la llave primaria de todas las columnas referenciadas por ella.
- DELETE de la llave primaria causa la eliminación de todas las filas llaves primarias que son referencias por ella.
- SET NULL UPDATE y DELETE a la fila de la llave primaria causa que la llave primaria sea colocada a NULL.
- El NO ACTION es la acción por omisión.

La acción de llave foránea te ofrece una gran flexibilidad en cómo controlar los cambios de la llave primaria afectan las filas de las llaves foráneas.

Multi-columnas en Llaves Primarias y Llaves Foráneas

Para especificar una llave primaria multi-columna, es necesario escribir por separado el comando PRIMARY KEY en la sentencia CREATE TABLE. Una clave foránea multi-columna tiene los mismos requerimientos, por lo que es necesario escribir el comando FOREIGN KEY por separado.

Manejando valores nulos en las claves foráneas

Un valor NULL no puede referenciar a una llave primaria. Una única columna de llave foránea es un NULL o coincidir con una llave primaria. En una multi-columna de llave foránea, algunas veces solo una parte de la llave foránea puede ser NULL. El com-

portamiento por default permite que algunas columnas en una multi-columna Foreign Key sea NULL y otras no sean NOT NULL.

Usando un MATCH FULL en una multi-columna con restricción de foreign key requiere que todas las columnas de en la llave sean NULAS o que todas las columnas de no sean NULAS.

Primero, las tablas son usadas para mostrar que el default permite una columna de una foreign key debe ser colocada a NULL. A continuación, la tabla matchtest es creada con la opción de la restricción del foreign key MATCH FULL. MATCH FULL; permite que todas las llaves sean colocadas a NULL, pero requiere que establecer a NULL solamente algunos valores claves multi-columna.

Checando frecuentemente la clave foránea

Por default, las restricciones de foreign key son comprobadas al finalizar cada consulta de INSERT, UPDATE, y DELETE. Así, si realiza una modificación compleja, la restricción de la llave foránea debe quedar validada en todo momento.

CHECK

La restricción CHECK hace cumplir los valores de las columnas restringidas. Así la restricción puede restringir una columna, por ejemplo; para un conjunto de valores, solo números positivos o datos razonables. Por default, CHECK permite valores NULOS.

Regla de integridad referencial

Cuando se define una columna como clave foránea, las filas de la tabla pueden contener en esa columna o bien el valor nulo (ningún valor), o bien un valor que existe en la otra tabla, un error sería asignar a un habitante una población que no está en la tabla de poblaciones. Eso es lo que se denomina integridad referencial y consiste en que los datos que referencian otros (claves foráneas) deben ser correctos. La integridad referencial hace que el sistema gestor de la base de datos se asegure de que no haya en las claves

foráneas valores que no estén en la tabla principal. A continuación se muestra el enunciado:

> «Si una tupla de una tabla A posee atributos (a1...an) que hacen referencia a la clave primaria de otra tupla de una tabla B, dichos atributos poseen, o bien valores nulos, o bien valores (v1 ... vn) que se corresponden con la clave de una tupla concreta de B».

Se implementa en SQL usando el comando references, cuando se trata de una clave primaria de un solo atributo o Foreign Key cuando se trata de una clave primaria de más de un atributo.

3. PREPARACIÓN DEL AMBIENTE DE TRABAJO

Para poder aplicar los conceptos descritos en este laboratorio es necesario tener una base de datos en la cual aplicar las restricciones que requiere el proyecto de trabajo. Uno de los trabajos adicionales que el lector debe realizar es el de integrar los conceptos explicados en el Laboratorio 1 y el 2 en una sola base de datos, con el fin de obtener experiencia práctica. Por lo que en el este laboratorio usaremos una nueva base de datos, distinta a la usada en el “Laboratorio 1. Control de Usuarios”.

Creación de tablas

Las tablas que en esta sección encuentra se crearon aplicando las reglas de conversión del modelo E-R al relacional al diagrama E-R de la sección 1. Este laboratorio no intenta explicar esas reglas.

Esquemas para el diagrama E-R de la Universidad ACME: Los nombres de los campos en algunos casos fueron cambiados, con respecto del diagrama E-R, por motivos de tamaños del nombre, sin embargo los conceptos son los mismos.

CuentaCheques(ncuenta, saldo, banco)
Cheque(ncuenta, cns, total, fecha);
Tabulador(idtab, importehora);
Profesores(idprofe, idtab, nombre, maximo, sueldo);
CursosEspeciales(idcurso, idprofe,cns,fini,ffin);
CursosExtracurricular(idextra, decextra, nhorascurso);
Materias(nmat, des, horacurso);
CEVerano(idcurso, nmat);
CEExtracurricula(idcurso, idextra);
Alumnos(matricula, nombre);
Bimestre(matricula, periodo, nmat, calificacion, faltas);
Faltas(matricula, fecha);
Concepto (idconcepto, desconcepto);
Recibo (folio,matricula, fecharec, totalrec);
DetalleRecibo(folio, cns, idconcepto, subtotal);
CalendarioEscolar(fecha, motivo);

3.1 Políticas a implementar en las tablas para el diagrama E-R de la Universidad ACME

A continuación hacemos un resumen de las políticas que se van a implementar y las tablas que se verán afectadas, la finalidad de este resumen es la de tener una panorámica completa de las restricciones que se aplican a la base de datos.

Los siguientes comandos de creación de tablas e inserción de datos deben ser ejecutados usando el usuarios postgres (el usuario por omisión) y se debe de cambiar de usuario hasta que explícitamente se le indique. Note que a diferencia del primer laboratorio, en este la creación de tablas usando restricciones contiene más código, y se debe a que con el uso de restricciones la declaración de las tablas es más detallada. Los lineamientos para la creación de estas restricciones están desarrollados a partir de las políticas descritas por la empresa.

- Ningún atributo que sea por sí mismo llave primaria o sea parte de la llave primaria puede aceptar valores nulos.

- Cuando un registro del maestro de recibos sea borrado, automáticamente deben ser borrados todos los registros relacionados en la tabla detalle del recibo.

- Cuando una cuenta de cheques sea borrada, automáticamente deben ser borrados todos los registros relacionados en la tabla cheques.

- Cuando se elimine un registro de la tabla *CursosEspeciales*, automáticamente debe ser borrado cualquier registro relacionado en las tablas especializadas de *ceextracurricula* y *ceverano*.

- NO SE PERMITEN VALORES NULOS sobre los siguientes campos de cada tabla (las llaves primarias se restringen en el primer lineamiento):

  CuentaCheques(banco)
  Tabulador(importehora);
  Profesores(nombre, maximo);
  Materias(des, horacurso);
  Alumnos(nombre);
  Concepto(desconcepto);
  CalendarioEscolar(motivo);

- NO SE PERMITEN VALORES IGUALES O MENORES A CERO sobre los siguientes campos en las siguientes tablas:

  Cheque(total);
  Tabulador(importehora);

Profesores(sueldo);
Recibo(totalrec);
DetalleRecibo(subtotal);

- La fecha de emisión del cheque solo puede ser la de hoy o anterior pero nunca posterior a la fecha del sistema.

3.2 Solución a los problemas planteados

Ahora pasamos de las políticas a los comandos necesarios en PostgreSQL que dan solución a los problemas planteados, esta propuesta requiere que la base de datos sea reconfigurada y para visualizar con detalle estos cambios, se sugiere que compare las que se crearon en el Laboratorio 1 contra aquellas que se modifican en este laboratorio. Los cambios que usted descubre, son las restricciones que se están aplicando en la base de datos.

Para cumplir con las políticas de la empresa, se reconstruyen las tablas en una nueva base de datos a la que llamamos UACMEREST. Copie y pegue las tablas que se muestran a continuación en la nueva base de datos, recuerde usar al usuario postgres en todos los casos. Finalmente, inserte los datos que se proporcionan para poder verificar que las restricciones planteadas son cubiertas por la nueva definición de las tablas.

```
-- Creando la base de datos UACMEREST
create database uacmerest;
-- Cambiarse de la BD por omisión a la ACME (en
PSQL)
\c uacmerest

--Creación de las tablas
-- la llave primaria no puede aceptar valores
nulos
CREATE TABLE cuentacheques (
ncuenta integer NOT NULL,
```

```
saldo numeric(7,2),
banco varchar NOT NULL,
CONSTRAINT cuentacheques_pkey PRIMARY KEY
(ncuenta)
);


-- la llave primaria no puede aceptar valores
nulos
-- El campo total debe ser mayor o igual a cero
-- Cuando se elimina el registro de la Cuenta de
Cheques …
-- todos los cheques de esa cuenta se borran
CREATE TABLE cheque (
ncuenta integer NOT NULL,
cns integer NOT NULL,
total numeric(10,2) CONSTRAINT importe_invalido
CHECK ( total > 0 ),
fecha date CONSTRAINT fecha_invalida CHECK (fecha
< now() ),
CONSTRAINT cheque_pkey PRIMARY KEY (ncuenta,
cns),
CONSTRAINT cheque_ncuenta_fkey FOREIGN KEY
(ncuenta)
REFERENCES cuentacheques (ncuenta) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE CASCADE
);

-- la llave primaria no puede aceptar valores
nulos
-- El campo importehora debe ser mayor o igual a
cero
CREATE TABLE tabulador (
idtab integer NOT NULL,
importehora varchar NOT NULL CONSTRAINT
importehora_invalido CHECK ( importehora > 0 ),
CONSTRAINT tabulador_pkey PRIMARY KEY (idtab)
);

-- la llave primaria no puede aceptar valores
nulos
```

```
-- El campo sueldo debe ser mayor o igual a cero
-- Cuando se borra un registro de la tabla
tabulador se borran los registros asociados en
-- la tabla profesores
CREATE TABLE profesores (
idprofe integer NOT NULL,
idtab integer,
nombre varchar NOT NULL,
maximo varchar NOT NULL,
sueldo double precision CONSTRAINT sueldo_
invalido CHECK ( sueldo > 0 ),
CONSTRAINT profesores_pkey PRIMARY KEY (idprofe),
CONSTRAINT profesores_idtab_fkey FOREIGN KEY
(idtab)
REFERENCES tabulador (idtab) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE NO ACTION
);


-- la llave primaria no puede aceptar valores
nulos
CREATE TABLE cursosespeciales (
idcurso integer NOT NULL,
idprofe integer,
fini varchar,
ffin varchar,
ncuenta integer,
cns integer,
CONSTRAINT cursosespeciales_pkey PRIMARY KEY
(idcurso),
CONSTRAINT cursosespeciales_ncuenta_fkey FOREIGN
KEY (ncuenta, cns)
REFERENCES cheque (ncuenta, cns) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE NO ACTION
);


-- la llave primaria no puede aceptar valores
nulos
```

```
CREATE TABLE cursosextracurricular (
idextra integer NOT NULL,
decextra text,
nhorascurso integer,
CONSTRAINT cursosextracurricular_pkey PRIMARY KEY
(idextra)
);

CREATE TABLE materias (
nmat integer NOT NULL,
des varchar NOT NULL,
horacurso integer NOT NULL,
CONSTRAINT materias_pkey PRIMARY KEY (nmat)
);

CREATE TABLE ceverano (
idcurso integer NOT NULL,
nmat integer,
CONSTRAINT ceverano_pkey PRIMARY KEY (idcurso),
CONSTRAINT ceverano_nmat_fkey FOREIGN KEY (nmat)
REFERENCES materias (nmat) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE CASCADE
);

CREATE TABLE ceextracurricula (
idcurso integer NOT NULL,
idextra integer,
CONSTRAINT ceextracurricula_pkey PRIMARY KEY
(idcurso),
CONSTRAINT ceextracurricula_idextra_fkey FOREIGN
KEY (idextra)
REFERENCES cursosextracurricular (idextra) MATCH
SIMPLE
ON UPDATE NO ACTION ON DELETE CASCADE
);

CREATE TABLE alumnos (
matricula integer NOT NULL,
nombre varchar NOT NULL,
CONSTRAINT alumnos_pkey PRIMARY KEY (matricula)
);
```

```
CREATE TABLE bimestre (
periodo integer NOT NULL,
matricula integer NOT NULL,
nmat integer,
calificacion integer,
faltas double precision,
CONSTRAINT bimestre_pkey PRIMARY KEY (matricula,
periodo),
CONSTRAINT bimestre_matricula_fkey FOREIGN KEY
(matricula)
REFERENCES materias (nmat) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE NO ACTION,
CONSTRAINT bimestre_matricula_fkey1 FOREIGN KEY
(matricula)
REFERENCES alumnos (matricula) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE NO ACTION
);

CREATE TABLE faltas (
fecha varchar NOT NULL,
matricula integer,
CONSTRAINT faltas_pkey PRIMARY KEY (fecha),
CONSTRAINT faltas_matricula_fkey FOREIGN KEY
(matricula)
REFERENCES alumnos (matricula) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE NO ACTION
);

CREATE TABLE concepto (
idconcepto integer NOT NULL,
desconcepto varchar NOT NULL,
CONSTRAINT concepto_pkey PRIMARY KEY (idconcepto)
);

CREATE TABLE recibo (
folio integer NOT NULL,
matricula integer,
fecharec varchar,
totalrec double precision CONSTRAINT totalrec_
invalido CHECK ( totalrec > 0 ),
CONSTRAINT recibo_pkey PRIMARY KEY (folio),
```

```
CONSTRAINT recibo_matricula_fkey FOREIGN KEY
(matricula)
REFERENCES alumnos (matricula) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE NO ACTION
);

CREATE TABLE detallerecibo (
cns integer NOT NULL,
idconcepto integer,
folio integer NOT NULL,
subtotal double precision CONSTRAINT subtotal_
invalido CHECK ( subtotal > 0 ),
CONSTRAINT detallerecibo_pkey PRIMARY KEY (folio,
cns),
CONSTRAINT detallerecibo_folio_fkey FOREIGN KEY
(folio)
REFERENCES recibo (folio) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE NO ACTION,
CONSTRAINT detallerecibo_idconcepto_fkey FOREIGN
KEY (idconcepto)
REFERENCES concepto (idconcepto) MATCH SIMPLE
ON UPDATE NO ACTION ON DELETE CASCADE
);

CREATE TABLE calendarioescolar (
fecha varchar NOT NULL,
motivo varchar NOT NULL,
CONSTRAINT calendarioescolar_pkey PRIMARY KEY
(fecha)
);
```

Inserción de datos para algunas tablas recién construidas

```
insert into CuentaCheques values(1,700,'HSBC');
insert into CuentaCheques values(2,9000,'HSBC');
insert into CuentaCheques values(3,60,'HSBC');
insert into CuentaCheques values(4,10,'HSBC');
insert into CuentaCheques values(5,1000,'HSBC');
insert into CuentaCheques values(6,200,'HSBC');
```

```
insert into Cheque values(1,10,200,'2008-02-01');
insert into Cheque
values(2,10,575.20,'2008-02-01');
insert into Cheque values(2,20,20,'2008-02-01');
insert into Cheque values(3,10,600,'2007-02-01');
insert into Cheque values(4,10,800,'2007-02-01');
insert into Cheque values(5,10,100,'2007-02-01');
insert into Cheque values(6,10,300,'2007-02-01');
insert into Tabulador values(10,100);
insert into Tabulador values(20,200);
insert into Tabulador values(30,300);
insert into Tabulador values(40,400);
insert into Tabulador values(50,500);
insert into Tabulador values(60,600);
insert into Tabulador values(70,700);
insert into Profesores values(1,40,'Roberto',
'Maestria',15000);
insert into Profesores values(2,70,'Carlos',
'Doctorado',25000);
insert into Profesores values(3,20,'Luis',
'Licenciatura',6000);
insert into Profesores values(4,30,'Yunuan',
'Maestria',12000);
insert into Profesores values(5,10,'Julio',
'Licenciatura',4500);
insert into Profesores values(6,20,'Samuel',
'Licenciatura',5500);
insert into CursosEspeciales
values(1,1,1,20070204,20050204);
insert into CursosEspeciales
values(2,2,2,20070204,20050204);
insert into CursosEspeciales
values(3,3,3,20070204,20050204);
insert into CursosEspeciales
values(4,4,4,20070204,20050204);
insert into CursosEspeciales
values(5,5,5,20070204,20050204);
insert into CursosExtracurricular
values(1,'admin',204);
insert into CursosExtracurricular
values(2,'diseño',204);
```

```
insert into CursosExtracurricular
values(3,'bdd',204);
insert into CursosExtracurricular
values(4,'java',204);
insert into Materias values(1,'admin bdd',204);
insert into Materias values(2,'redes',204);
insert into Materias values(3,'redes 2',204);
insert into Materias values(4,'admin bdd',204);
```

Los datos insertados solo sirven para demostrar el funcionamiento de los privilegios de acceso, queda del usuario insertar datos en el resto de las tablas para demostrar que las reglas de acceso son funcionales para cada usuario.

Borrado de las tablas

En caso de necesitar borrar las tablas este es el orden en que deben ser borradas, ya que la aplicación estricta de la integridad referencial puede generar problemas en el proceso de borrado.

```
drop table CalendarioEscolar;
drop table DetalleRecibo;
drop table Recibo;
drop table Concepto;
drop table Faltas;
drop table Bimestre;
drop table Alumnos;
drop table CEExtracurricula;
drop table CEVerano;
drop table Materias;
drop table CursosExtracurricular;
drop table CursosEspeciales;
drop table Profesores;
drop table Tabulador;
drop table Cheque;
drop table CuentaCheques;
```

Verificando las tablas con restricciones

Después de construir las tablas, hay que validar que nuestras restricciones funcionen adecuadamente. Puesto que algunas implican fechas, y se desconoce el momento en que el lector practique con este laboratorio, cuando sea necesario se le indica que fecha debe capturar.

- Intente insertar el siguiente registro en la tabla cheques.

  ```
  insert into Cheque values(6,10,0,'2014-02-01');
  ```

  ¿Se le permitió ejecutar la operación? Explique ¿Qué pasó?

- Elimine la cuenta de cheques no. 6

  ```
  Delete from CuentaCheques where ncuenta = 6;
  ```

  ¿Funcionó? Ahora verifique que paso con el cheque 10 que inserto en el paso anterior. Explique

- Inserte el siguiente registro en la tabla Concepto:

  ```
  Insert into concepto values(1);
  ```

  ¿Funcionó? Explique detalladamente.

- Intente insertar el siguiente registro en la tabla cheques

  ```
  insert into Cheque values(6,10,10, <fecha pos-
  terior al día en que se ejecuta>);
  ```

  Ejemplo si hoy es 19 de Febrero del 2008 use como fecha posterior el 21 de Febrero.

  ¿Funcionó? Explique detalladamente.

5. Trabajo adicional

Los siguientes problemas no están resueltos, por lo que es necesario aplicar su experiencia adquirida para resolver estos.

1. 2. Usando a los distintos usuarios verifique que se le permitan efectuar los movimientos acordes al privilegio de acceso asignado sobre cada una de las tablas.
2. Construya la regla faltante para el grupo escolar sobre la vista VistaProfesoresEscolar, construya la regla para cuando el usuario barroso desea eliminar el registro del docente Luis.
3. Agregue restricciones adicionales. Use la siguiente política: El encargado de capturar los cursos especiales será el departamento administrativo (cualquier usuario), pero quién asignará el docente será el departamento escolar. Construya la vista y la regla que va a reglamentar esta inserción de datos.
    - La inserción de datos a la tabla CursosEspeciales y las tablas especializadas la efectuará el DA, con el campo profesor en nulos o referenciando a un registro especial (por ejemplo, un profesor no válido) de la tabla Profesores.
    - La modificación de las tablas CursosEspeciales para asignar al Profesor la efectuará el DCE.
4. Explique las causas por las que las instrucciones para los usuarios: barroso y ávila, marcan errores o funcionan adecuadamente.
5. Integre los conceptos explicados en los Laboratorios 1 y 2 de esta serie en una sola base de datos. Pistas: Destruya la base de datos UACME, aplique los conceptos del Laboratorio 2 y luego aplique los privilegios de acceso de los usuarios.

REFERENCIAS

PostgreSQL Introduction and Concepts by Bruce Momjiam

PostgreSQL by Susan Douglas and Korry Douglas.

Manual de postgreSQL: **www.postgresql.org**

Fundamentos de Base de Datos. 5a. Edición de Abraham Silberschatz. Ed. McGraw-Hill

Elmasri y Navathe: Fundamentos de Sistemas de Bases de Datos - 3ª edición, 2002.

Garcia-Molina, Ullman y Widom: *Database systems: the complete book*. Prentice-Hall.