SQL

(...la guía ***básica*** de ***supervivencia*** de dmi)

Universidad de los Andes
Demián Gutierrez
Enero 2009

Crear una Tabla (Esquema de Relación) (1) (Cascadas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# CREATE TABLE departamento (
             codigo INT PRIMARY KEY,
             nombre VARCHAR(50)
           );
NOTICE:  CREATE TABLE / PRIMARY KEY will create
  implicit index "departamento_pkey" for table "departamento"
CREATE TABLE
postgres=# CREATE TABLE profesor (
             cedula INT PRIMARY KEY,
             nombre VARCHAR(50),
             sueldo NUMERIC,
             codigodpto INT,
             FOREIGN KEY (codigodpto)
               REFERENCES departamento(codigo)
           );
NOTICE:  CREATE TABLE / PRIMARY KEY will create
  implicit index "profesor_pkey" for table "profesor"
CREATE TABLE
```

¿Recuerdan las cascadas con el “drop table”?

Crear una Tabla (Esquema de Relación) (2) (Cascadas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

DEPARTAMENTO

codigo	nombre
1	Computacion
2	Investigacion
3	Control

PROFESOR

cedula	nombre	sueldo	codigodpto
12489778	Pedro Perez	2000	1
13449543	Juan Rojas	2500	1
15669442	Jose Fuentes	2100	2
11639337	Miguel Redondo	2000	2
10274448	Andres Silva	2000	3
12539445	Luis Cardenas	3000	3

```
postgres=# DELETE FROM departamento WHERE codigo=1;
ERROR:  update or delete on table "departamento" violates foreign key
  constraint "profesor_codigodpto_fkey" on table "profesor"
DETAIL:  Key (codigo)=(1) is still referenced from table "profesor".
```

```
postgres=# UPDATE departamento SET codigo=20 WHERE codigo=2;
ERROR:  update or delete on table "departamento" violates foreign key
  constraint "profesor_codigodpto_fkey" on table "profesor"
DETAIL:  Key (codigo)=(1) is still referenced from table "profesor".
```

Crear una Tabla (Esquema de Relación) (3) (Cascadas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# CREATE TABLE departamento (
             codigo INT PRIMARY KEY,
             nombre VARCHAR(50)
           );
NOTICE:  CREATE TABLE / PRIMARY KEY will create
  implicit index "departamento_pkey" for table "departamento"
CREATE TABLE

postgres=# CREATE TABLE profesor (
             cedula INT PRIMARY KEY,
             nombre VARCHAR(50),
             sueldo NUMERIC,
             codigodpto INT,
             FOREIGN KEY (codigodpto)
               REFERENCES departamento(codigo)
               ON DELETE CASCADE
               ON UPDATE CASCADE
           );
NOTICE:  CREATE TABLE / PRIMARY KEY will create
  implicit index "profesor_pkey" for table "profesor"
CREATE TABLE
```

Crear una Tabla (Esquema de Relación) (4) (Cascadas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# DELETE FROM departamento WHERE codigo=1;
DELETE 1
postgres=# UPDATE departamento SET codigo=20 WHERE codigo=2;
UPDATE 1
```

DEPARTAMENTO

codigo	nombre
3	Control
20	Investigacion

PROFESOR

cedula	nombre	sueldo	codigodpto
10274448	Andres Silva	2000	3
12539445	Luis Cardenas	3000	3
15669442	Jose Fuentes	2100	20
11639337	Miguel Redondo	2000	20

RESTRICT	No permite la acción sobre la fila en cuestión (Pero no genera ningún error)
NO ACTION	No permite la acción sobre la fila en cuestión y genera un error. Esta es la opción por defecto
CASCADE	Realiza la eliminación o los cambios en cascada
SET NULL	Asigna el valor NULL a las filas que apunten a la fila modificada (Genera un error si esto no es posible o viola alguna restricción)
SET DEFAULT	Asigna el valor por defecto a las filas que apunten a la fila modificada (Genera un error si esto no es posible o viola alguna restricción)

Modificar Tablas (1)
(Añadir / Eliminar Columnas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# ALTER TABLE profesor ADD COLUMN seguro NUMERIC;
ALTER TABLE
```

PROFESOR

cedula	nombre	sueldo	codigodpto	seguro
1	Pedro Perez	2000	1	
2	Juan Rojas	2500	1	
3	Jose Fuentes	2100	2	

```
postgres=# ALTER TABLE profesor
               DROP COLUMN seguro;
ALTER TABLE
postgres=# ALTER TABLE profesor
               ADD COLUMN seguro NUMERIC DEFAULT 0.3;
ALTER TABLE
```

PROFESOR

cedula	nombre	sueldo	codigodpto	seguro
1	Pedro Perez	2000	1	0.3
2	Juan Rojas	2500	1	0.3
3	Jose Fuentes	2100	2	0.3

Modificar Tablas (2) (Añadir / Eliminar Restricciones)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# ALTER TABLE profesor ADD CHECK (nombre <> '');
ALTER TABLE
```

```
postgres=# \d profesor;
   Column   |          Type          |  Modifiers
------------+------------------------+-------------
 cedula     | integer                | not null
 ... (OTRAS COLUMNAS, ELIMINADAS POR RAZONES DE ESPACIO)
Indexes:
    "profesor_pkey" PRIMARY KEY, btree (cedula)
Check constraints:
    "profesor_nombre_check" CHECK (nombre::text <> ''::text)
Foreign-key constraints:
    "profesor_codigodpto_fkey" FOREIGN KEY (codigodpto)
    REFERENCES departamento(codigo)
```

```
postgres=# INSERT INTO profesor VALUES (7, '', 3000, 2);
ERROR:  new row for relation "profesor" violates
  check constraint "profesor_nombre_check"
```

```
postgres=# ALTER TABLE profesor
             DROP CONSTRAINT profesor_nombre_check;
ALTER TABLE
```

Modificar Tablas (3)
(Añadir / Eliminar Valores por Defecto)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# ALTER TABLE profesor
             ALTER COLUMN sueldo SET DEFAULT 2000;
ALTER TABLE

postgres=# \d profesor;
              Table "public.profesor"
   Column   |         Type          |  Modifiers
------------+-----------------------+-------------
 cedula     | integer               | not null
 nombre     | character varying(50) |
 sueldo     | numeric               | default 2000
 codigodpto | integer               |
Indexes:
    "profesor_pkey" PRIMARY KEY, btree (cedula)
Check constraints:
    "profesor_nombre_check" CHECK (nombre::text <> ''::text)
Foreign-key constraints:
    "profesor_codigodpto_fkey" FOREIGN KEY (codigodpto)
      REFERENCES departamento(codigo)

postgres=# ALTER TABLE profesor
             ALTER COLUMN sueldo DROP DEFAULT;
ALTER TABLE
```

Modificar Tablas (4)
(Cambiar el Tipo de Dato de una Columna)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# ALTER TABLE profesor ALTER COLUMN seguro TYPE INT;
ALTER TABLE

postgres=# \d profesor;
              Table "public.profesor"
   Column   |          Type          |  Modifiers
------------+------------------------+-------------
 cedula     | integer                | not null
 nombre     | character varying(50)  |
 sueldo     | integer                |
 codigodpto | integer                |
Indexes:
    "profesor_pkey" PRIMARY KEY, btree (cedula)
Check constraints:
    "profesor_nombre_check" CHECK (nombre::text <> ''::text)
Foreign-key constraints:
    "profesor_codigodpto_fkey" FOREIGN KEY (codigodpto)
      REFERENCES departamento(codigo)

postgres=# ALTER TABLE profesor
             ALTER COLUMN nombre TYPE VARCHAR(6);
ERROR:  value too long for type character varying(6)
```

Antes era numeric

Modificar Tablas (5)
(Renombrar Columnas / Tablas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# ALTER TABLE profesor RENAME COLUMN sueldo TO salario;
ALTER TABLE

postgres=# \d profesor
              Table "public.profesor"
   Column   |          Type          |  Modifiers
------------+------------------------+-------------
 cedula     | integer                | not null
 nombre     | character varying(50)  |
 salario    | integer                |
 codigodpto | integer                |
Indexes:
    "profesor_pkey" PRIMARY KEY, btree (cedula)
Check constraints:
    "profesor_nombre_check" CHECK (nombre::text <> ''::text)
Foreign-key constraints:
    "profesor_codigodpto_fkey" FOREIGN KEY (codigodpto)
      REFERENCES departamento(codigo)

postgres=# ALTER TABLE profesor RENAME TO empleado;
ALTER TABLE
```

Antes se llamaba sueldo

Vistas

```
postgres=# CREATE VIEW profesores_departamentos AS
  SELECT profesor.cedula, profesor.nombre AS nombreProf,
         departamento.codigo, departamento.nombre AS nombreDpto
    FROM profesor, departamento
    WHERE profesor.codigodpto = departamento.codigo;
CREATE VIEW
```

```
postgres=# SELECT * FROM profesores_departamentos;
  cedula  |   nombreprof    | codigo |  nombredpto
----------+-----------------+--------+---------------
 12489778 | Pedro Perez     |      1 | Computaion
 13449543 | Juan Rojas      |      1 | Computaion
 15669442 | Jose Fuentes    |      2 | Control
 11639337 | Miguel Redondo  |      2 | Control
 10274448 | Andres Silva    |      3 | Investigacion
 12539445 | Luis Cardenas   |      3 | Investigacion
 14556333 | Hector Mora     |      1 | Computaion
 15889332 | Felipe Redondo  |      3 | Investigacion
(8 rows)
```

Del manual de postgres: Currently, views are read only: the system will not allow an insert, update, or delete on a view.

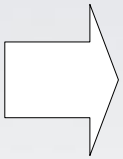
Índices

```
postgres=# CREATE INDEX idx_sueldo_profesor
             ON profesor(sueldo);
CREATE INDEX

postgres=# CREATE INDEX idx_codigo_nombre
             ON departamento(codigo, nombre);
CREATE INDEX

postgres=# \di
                        List of relations
 Schema |         Name          | Type  |  Owner   |    Table
--------+-----------------------+-------+----------+--------------
 public | departamento_pkey     | index | postgres | departamento
 public | idx_codigo_nombre     | index | postgres | departamento
 public | idx_sueldo_profesor   | index | postgres | profesor
 public | profesor_pkey         | index | postgres | profesor
(4 rows)

postgres=# DROP INDEX idx_sueldo_profesor;
DROP INDEX

postgres=# DROP INDEX idx_codigo_nombre;
DROP INDEX
```

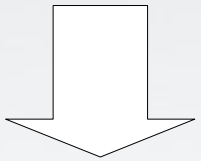
Identificadores Abstractos Autoincrementales

DEPARTAMENTO

```
codigo|   nombre
------+-------------
     3|Control
    20|Investigacion
```



```
postgres=# SELECT MAX(codigo)+1
             FROM departamento;
 max
-----
  21
```



```
postgres=# INSERT INTO departamento
             VALUES (21, 'Computacion');
INSERT 0 1
```

O bien...

```
postgres=# INSERT INTO departamento
             VALUES (
               (SELECT MAX(codigo)+1 FROM departamento),
               'Sist. Interpretativa'
             );
INSERT 0 1
```

¿Ventajas / Desventajas?

Secuencias (1) (Sólo PostgreSQL)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# CREATE SEQUENCE seq_prof_id;
CREATE SEQUENCE

postgres=# SELECT * FROM seq_profesor_id;
   seq_name|lst_val|inc_by|max_val|min_val|cache_value|log_cnt|isCyc|isCall
-----------+-------+------+-------+-------+-----------+-------+-----+------
seq_prof_id|      1|     1| GRANDE|      1|          1|      1|f    |f
(1 row)

postgres=# SELECT nextval('seq_profesor_id');
 nextval
---------
       1
postgres=# SELECT nextval('seq_profesor_id');
 nextval
---------
       2

postgres=# SELECT * FROM seq_profesor_id;
   seq_name|lst_val|inc_by|max_val|min_val|cache_value|log_cnt|isCyc|isCall
-----------+-------+------+-------+-------+-----------+-------+-----+------
seq_prof_id|      2|     1| GRANDE|      1|         32|      1|f    |t
(1 row)
```

Resuelve el problema de la concurrencia *(porque garantiza la atomicidad de la operación de consultar e incrementar la secuencia*

Secuencias (2) (Sólo PostgreSQL)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
postgres=# CREATE TABLE profesor (
             cedula INT PRIMARY KEY
               DEFAULT nextval('seq_profesor_id'),
             nombre VARCHAR(50),
             sueldo NUMERIC,
             codigoDpto INT REFERENCES departamento(codigo)
           );

postgres=# INSERT INTO profesor VALUES
(DEFAULT, 'Pedro Perz', 2000, 1),
(DEFAULT, 'Juan Rojas', 2000, 1);
INSERT 0 2

postgres=# SELECT * FROM profesor;
 cedula |   nombre    | sueldo | codigodpto
--------+-------------+--------+------------
      1 | Pedro Perz |   2000 |          1
      2 | Juan Rojas |   2000 |          1
(2 rows)

postgres=# DROP SEQUENCE seq_profesor_id;
DROP SEQUENCE
```

Procedimientos Almacenados (1)

```
postgres=# CREATE LANGUAGE plpgsql;
CREATE LANGUAGE
```

```
postgres=# CREATE FUNCTION insertDepartamento(nombre VARCHAR(50))
  RETURNS INT AS $$
DECLARE
    maxid RECORD;
BEGIN
    SELECT MAX(codigo) + 1 AS x INTO maxid FROM departamento;
    INSERT INTO departamento VALUES (maxid.x, nombre);
    RETURN maxid.x;
END
$$ LANGUAGE plpgsql;
CREATE FUNCTION
```

```
postgres=# SELECT insertDepartamento('Informatica');
 insertdepartamento
--------------------
                  24
(1 row)
```

```
postgres=# DROP FUNCTION insertdepartamento(VARCHAR(50));
DROP FUNCTION
```

Gatillos (Triggers) (1)

UNIVERSIDAD DE LOS ANDES VENEZUELA

- Primero se define un procedimiento almacenado a ejecutar cuando el gatillo se dispara.
- Luego se define el gatillo.
- El gatillo incluye el o los eventos que lo disparan, cuando se dispara, si es a nivel de sentencia o de fila y el procedimiento almacenado que se ejecuta al dispararse.
- En base al valor de retorno del procedimiento almacenado asociado al gatillo se pueden conseguir diferentes efectos.

Gatillos (Triggers) (2)

Evento al que está asociado un gatillo:

ON INSERT: Se dispara el gatillo al ejecutarse una instrucción INSERT sobre la tabla a la cual está asociado

ON UPDATE: Se dispara el gatillo al ejecutarse una instrucción UPDATE sobre la tabla a la cual está asociado

ON DELETE: Se dispara el gatillo al ejecutarse una instrucción DELETE sobre la tabla a la cual está asociado

Pero, además...

Gatillos (Triggers) (3)

El alcance del gatillo:

POR FILA: El gatillo se ejecuta para cada una de las filas afectadas por la sentencia por la que fue disparado

POR SENTENCIA: El gatillo se ejecuta una sola vez para la sentencia que lo disparó (sin importar cuantas filas sean afectadas

Pero, además...

Gatillos (Triggers) (4)

El momento en que se dispara el gatillo:

ANTES: Los gatillos de este tipo se ejecutan antes de que la sentencia se ejecute o de que esta afecte una fila en particular (dependiendo de si es por sentencia o por fila)

DESPUÉS: Los gatillos de este tipo se ejecutan después de que la sentencia se haya ejecutado o de que haya afectado una fila en particular (dependiendo si es por sentencia o por fila)

Pero, además...

Gatillos (Triggers) (5)

El valor de retorno del procedimiento almacenado define:

NULL: No se ejecuta la operación que disparó el gatillo para la fila para la cual el procedimiento retornó NULL. En caso de ser un gatillo por sentencia el procedimiento siempre debe retornar NULL.

UNA FILA: En el caso de los gatillos asociados a sentencias INSERT o UPDATE se puede retornar la fila que se desea sea resultado de la operación, es decir, se pueden alterar los valores a ser actualizados o insertados para una fila en cuestión

Gatillos (Triggers) (7)

MINMAX_SUELDO_PROFESOR

min_sueldo	max_sueldo
1900	3500

Gatillos (Triggers) (6)

```
postgres=# CREATE FUNCTION onInsertProfesor() RETURNS TRIGGER AS $$
             DECLARE
               min_max_sueldo RECORD;
               avg_sueldo RECORD;
             BEGIN
               SELECT * INTO min_max_sueldo
                 FROM minmax_sueldo_profesor;

               IF NEW.sueldo < min_max_sueldo.min_sueldo
                 OR NEW.sueldo > min_max_sueldo.max_sueldo THEN
                 SELECT AVG(sueldo) AS x INTO avg_sueldo
                   FROM profesor;
                 NEW.sueldo = avg_sueldo.x;
               END IF;
               RETURN NEW;
             END
           $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

Si el sueldo de un profesor es menor que el mínimo sueldo definido, o mayor que el máximo entonces le asigna el sueldo promedio

```
postgres=# CREATE TRIGGER verificarSueldo BEFORE INSERT ON profesor
             FOR EACH ROW EXECUTE PROCEDURE onInsertProfesor();
CREATE TRIGGER
```

Gatillos (Triggers) (7)

MINMAX_SUELDO_PROFESOR

```
 min_sueldo | max_sueldo
------------+------------
       1900 |       3500
```

```
postgres=# INSERT INTO profesor
             VALUES (14556333, 'Hector Mora', 1000, 1);
INSERT 0 1
postgres=# INSERT INTO profesor
             VALUES (15889332, 'Felipe Redondo', 5000, 3);
INSERT 0 1
```

PROFESOR

cedula	nombre	sueldo	codigodpto
12489778	Pedro Perez	2000	1
13449543	Juan Rojas	2500	1
15669442	Jose Fuentes	2100	2
11639337	Miguel Redondo	2000	2
10274448	Andres Silva	2000	3
12539445	Luis Cardenas	3000	3
14556333	Hector Mora	2266.6666666666666667	1
15889332	Felipe Redondo	2266.6666666666666667	3

¿Cómo insertar Registros resultantes de una consulta en otra tabla?

UNIVERSIDAD DE LOS ANDES VENEZUELA

A petición del público...

```
postgres=# CREATE TABLE autor (
             cedula INT PRIMARY KEY,
             nombre VARCHAR(50),
             estado VARCHAR(50)
           );
NOTICE:  CREATE TABLE / PRIMARY KEY will create
  implicit index "autore_pkey" for table "autore"
CREATE TABLE


postgres=# CREATE TABLE autor_merida (
             cedula INT PRIMARY KEY,
             nombre VARCHAR(50)
           );
NOTICE:  CREATE TABLE / PRIMARY KEY will create
  implicit index "autor_merida_pkey" for table "autor_merida"
CREATE TABLE
```

¿Cómo insertar Registros resultantes de una consulta en otra tabla?

UNIVERSIDAD DE LOS ANDES VENEZUELA

AUTOR

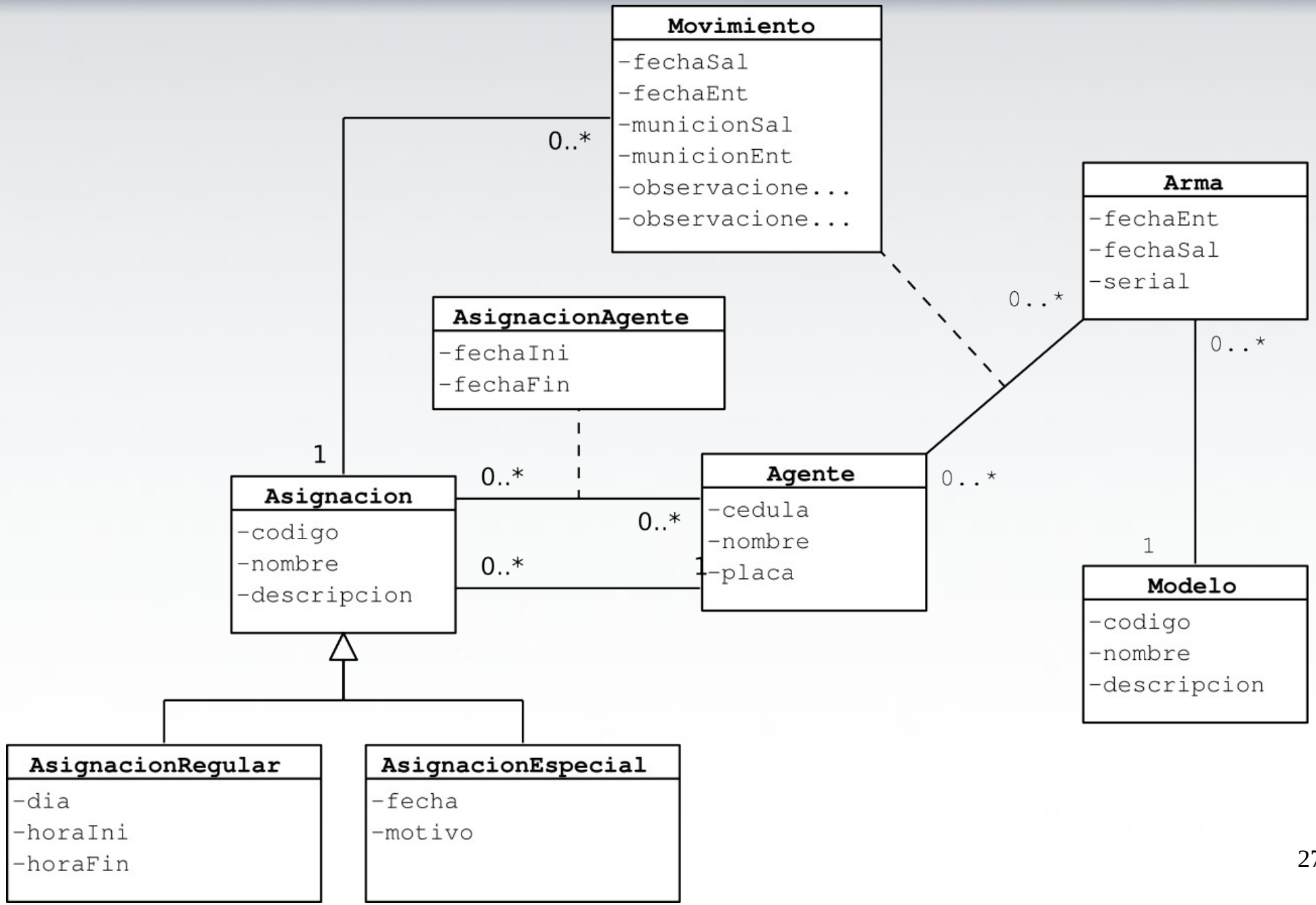
cedula	nombre	estado
13449009	Pedro Perez	Merida
12334019	Jose Garcia	Merida
11325768	Miguel Quintero	Zulia
10896529	Luis Silva	Zulia
9627166	Andres Torres	Miranda

```
postgres=# INSERT INTO autor_merida (cedula, nombre)
             SELECT cedula, nombre
               FROM autor
               WHERE estado='Merida';
INSERT 0 2
```

AUTOR_MERIDA

cedula	nombre
13449009	Pedro Perez
12334019	Jose Garcia

Ejemplo:


Movimiento
-fechaSal
-fechaEnt
-municionSal
-municionEnt
-observacione...
-observacione...
Arma
-fechaEnt
-fechaSal
-serial
AsignacionAgente
-fechaIni
-fechaFin
Asignacion
-codigo
-nombre
-descripcion
Agente
-cedula
-nombre
-placa
Modelo
-codigo
-nombre
-descripcion
AsignacionRegular
-dia
-horaIni
-horaFin
AsignacionEspecial
-fecha
-motivo
0..*
1
0..*
0..*
0..*
0..*
0..*
1
1

Ejemplo:

Movimiento
(Cedula, Serial, fechaSal, fechaEnt, munSal, munEnt, obsSal, obsEnt, *codigoAsig*)

Arma
(Serial, fechaEnt, FechaSal, *codModelo*)

Agente
(Cedula, Nombre, Placa)

AsignacionAgente
(*Codigo*, *Cedula*, nombre, descripcion)

Modelo
(Codigo, nombre, descripción)

Asignación
(Codigo, nombre, descripcion, *cedulaResp*)

AsignacionRegular
(*Codigo*, dia, horaIni, horaFin)

AsignacionEspecial
(*Codigo*, fecha, motivo)

UNIVERSIDAD DE LOS ANDES VENEZUELA

¡Gracias!