Posrgresql

```
CREATE DATABASE "Colegios"
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    LC_COLLATE = 'Spanish_Spain.1252'
    LC_CTYPE = 'Spanish_Spain.1252'
    TABLESPACE = pg_default
    CONNECTION LIMIT = -1;
```

Creación de Tablas

```
create table Estudiante(
        est_doc varchar(15),
        est_nom varchar (30) not null,
        est_ape varchar (30) not null,
        est_sex varchar (1),
        est_grado int2 not null,
        est_fecnac date,
        est_colid varchar (15),
        primary key (est_doc),
        foreign key (est_colid) references Colegio(col_id) on update cascade on delete restrict
);

create table Colegio(
        col_id varchar(15),
        col_nom varchar(30) not null,
        col_dir varchar (20) not null,
        col_ancho float not null,
        col_largo float not null,
        col_terreno float,
        col_ciucodpos varchar(15),
        primary key (col_id),
        foreign key (col_ciucodpos) references Ciudad (ciu_codpos) on update cascade on delete
        restrict
);

create table Ciudad(
        ciu_codpos varchar(15),
        ciu_nom varchar(30) not null,
        ciu_tempromc int2,
        primary key (ciu_codpos)
);
```

Triggers

```
create or replace function tg_estudiante() returns trigger as
$$
	begin
		if (upper(new.est_sex) = 'F' or upper(new.est_sex) = 'M')then
			insert into "estudiante" values (new.est_doc, new.est_nom,
			new.est_ape, new.est_sex,new.est_grado,new.est_fecnac,new.est_colid);
		else
			raise exception 'Las datos para el campo de sexo de los estudiantes debe
			ser F o M, f o m';
		end if;
		return new;
	end;
$$
language 'plpgsql'

create trigger tr_estin before insert on estudiante
for each row
execute procedure tg_estudiante();


create or replace function tg_ciudad() returns trigger as
$$
	begin
		if (new.ciu_codpos = '' or upper(new.ciu_nom) = '')then
			raise exception 'Los campos del nombre o el código postal de la ciudad no
			deben estar vacíos';
		else
			if (exists (select * from ciudad where ciu_codpos=new.ciu_codpos)) then
				raise exception 'La ciudad ingresada ya existe';
			end if;
		end if;
		return new;
	end;
$$
language 'plpgsql'

create trigger tr_ciudad before insert on ciudad
for each row
execute procedure tg_ciudad();
```

```
create or replace function tg_valcol () returns trigger as
$$
	begin
		if (new.col_id = '' or upper(new.col_nom) = '')then
			raise exception 'Los campos del nombre o identificación del colegio no
			deben estar vacíos';
		else
			if (exists (select * from ciudad where col_id=new.col_id)) then
				raise exception ‘El Colegio ingresado  ya existe';
			end if;
		end if;
		return new;
	end;
$$
language 'plpgsql'

create trigger tr_valcol before insert on colegio
for each row
execute procedure tg_valcol ();


create or replace function tg_colegio() returns trigger as
$$
	begin
		if exists(select col_largo, col_ancho from colegio) then
			new.col_terreno = new.col_largo * new.col_ancho;
		end if;
		return new;
	end;
$$
language 'plpgsql'

create trigger tr_terreno before insert on colegio
for each row
execute procedure tg_colegio();
```

Registro de auditoria

```
create table auditoria(
	id_auditoria serial not null,
	“nomtabla” varchar(45) not null,
	“Operación” char(1) not null,
	“valorAnterior” text,
	“NuevoValor” text,
```

```
        "Fecha" timestamp without time zone not null,
        "Usuario" varchar(45) not null,
        constraint id_auditoria primary key (id_auditoria)
) with (OIDS=FALSE);
alter table auditoria owner to postgres;


create or replace function aud_log() returns trigger as
$$
        begin
                if (TG_OP = 'DELETE') then
                        insert into auditoria ("nomtabla", "Operacion", "valorAnterior",
                        "NuevoValor", "Fecha","Usuario")
                                values  (TG_TABLE_NAME,'D', OLD, NULL, now(), USER);
                        return OLD;
                elseif (TG_OP = 'UPDATE') then
                        insert into auditoria ("nomtabla", "Operacion", "valorAnterior",
                        "NuevoValor","Fecha","Usuario")
                                        values  (TG_TABLE_NAME, 'U', OLD, NEW, now(), USER);
                        return NEW;
                elseif (TG_OP = 'INSERT') then
                        insert into auditoria ("nomtabla", "Operacion", "valorAnterior",
                        "NuevoValor", "Fecha","Usuario")
                                        values  (TG_TABLE_NAME, 'I', NULL, NEW, now(), USER);
                        return NEW;
                end if;
                return NULL;
        end;
$$
language 'plpgsql' VOLATILE COST 100;
ALTER FUNCTION aud_log() OWNER TO postgres;

create trigger tr_audit after insert or delete or update on ciudad
for each row
execute procedure aud_log();

create trigger tr_audit after insert or delete or update on colegio
for each row
execute procedure aud_log();

create trigger tr_audit after insert or delete or update on estudiante
for each row
execute procedure aud_log();
```

Anexos

```
insert into estudiante (est_doc,est_nom,est_ape,est_sex,est_grado,est_fecnac,est_colid)
        values  (103,'Juan','Pereza','P','8','19/03/2004',103);
```

Data Output | Explain | Messages | Notifications

```
ERROR:  Las datos para el campo de sexo de los estudiantes debe ser F o M, f o m
CONTEXT:  función PL/pgSQL tg_estudiante() en la línea 6 en RAISE
SQL state: P0001
```

```
insert into ciudad (ciu_codpos,ciu_nom,ciu_tempromc)
        values ('153','',25);
```

Data Output | Explain | Messages | Notifications

```
ERROR:  Los campos del nombre o el codigo postal de la ciudad no deben estar vacidos
CONTEXT:  función PL/pgSQL tg_ciudad() en la línea 4 en RAISE
SQL state: P0001
```

Data Output | Explain | Messages | Notifications

```
ERROR:  La ciudad ingresada ya existe
CONTEXT:  función PL/pgSQL tg_ciudad() en la línea 7 en RAISE
SQL state: P0001
```

```
insert into colegio (col_id, col_nom, col_dir, col_ancho, col_largo, col_ciucodpos)
        values ('496','Presentacion','Calle 9 # 2-38',146.3,156.9,'153');
```

```
select col_id, col_nom, col_largo, col_ancho, col_terreno from colegio
```

Data Output | Explain | Messages | Notifications

	col_id [PK] character varying (15)	col_nom character varying (30)	col_largo double precision	col_ancho double precision	col_terreno double precision
1	496	Presentacion	156.9	146.3	[null]

```
insert into ciudad (ciu_codpos,ciu_nom,ciu_tempromc)
        values ('198','Cucuta',32);
```

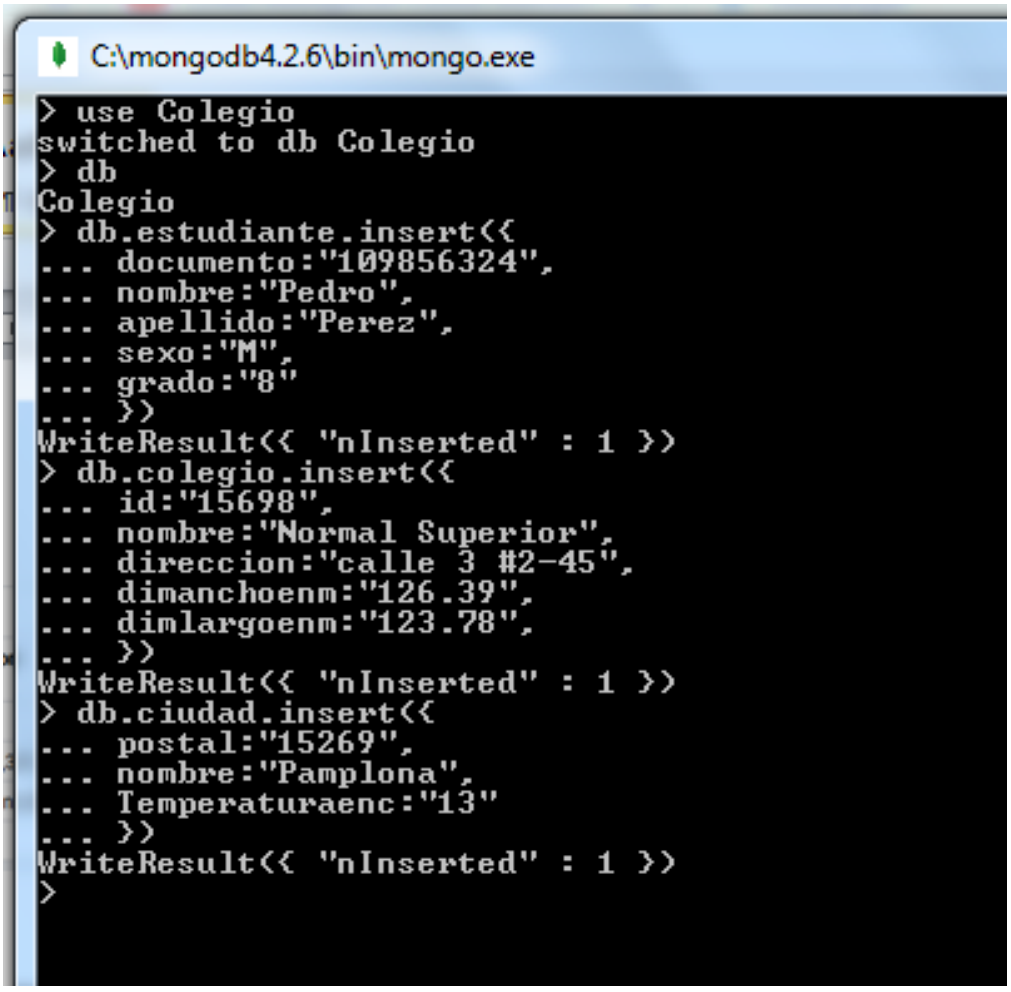
```
update ciudad set ciu_nom = 'Medellin' where ciu_codpos = '198';
```

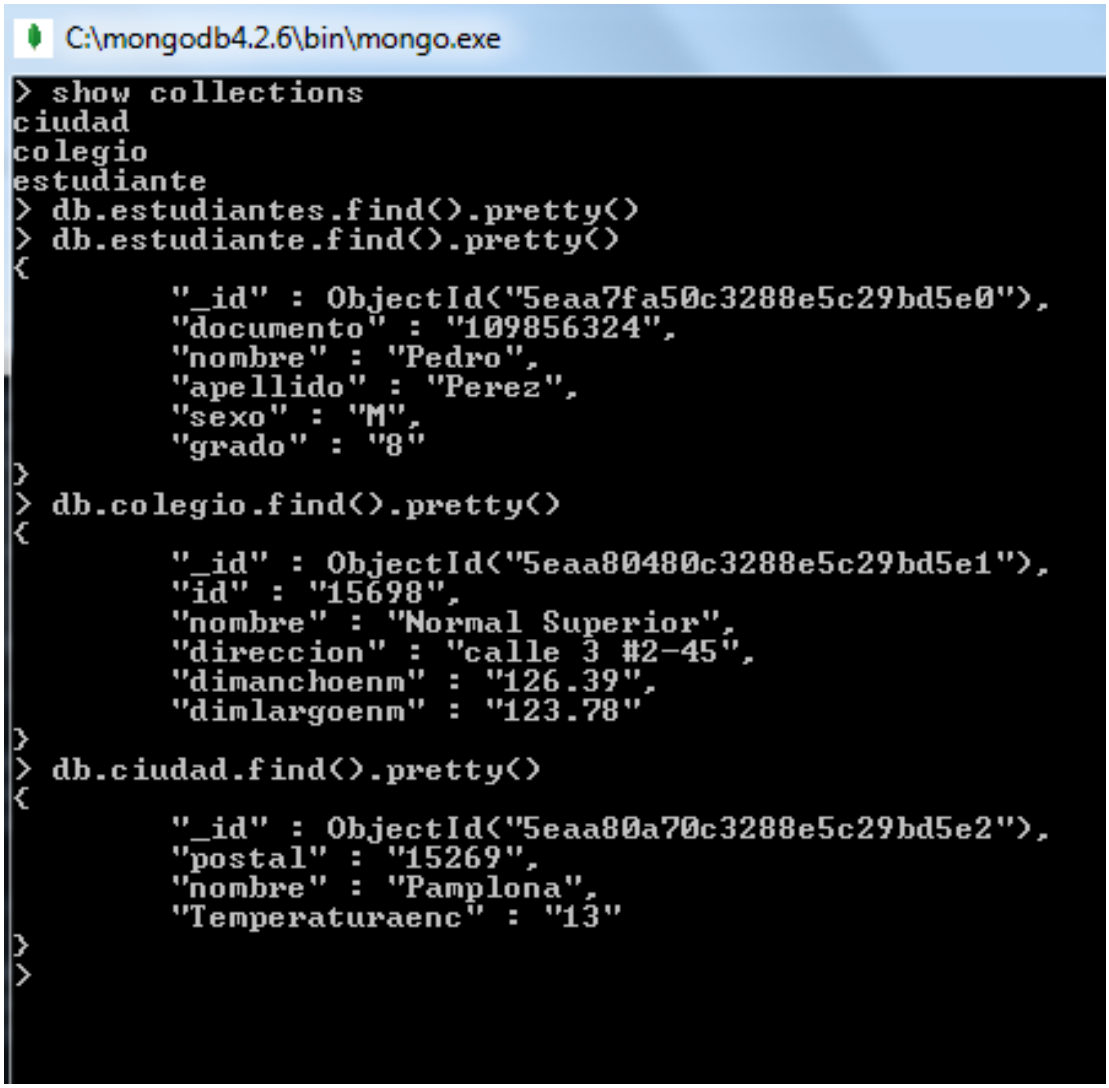
```
delete from ciudad where ciu_codpos = '198';

select * from auditoria
```

Data Output | Explain | Messages | Notifications

	id_auditoria [PK] integer	nomtabla character varying (45)	Operacion character (1)	valorAnterior text	NuevoValor text	Fecha timestamp without time zone	Usuario character varying (45)
1	5	ciudad	I	[null]	(198,Cucuta,32)	2020-04-30 01:07:04.00276	postgres
2	6	ciudad	U	(198,Cucuta,32)	(198,Medellin,32)	2020-04-30 01:11:20.673441	postgres
3	7	ciudad	D	(198,Medellin,32)	[null]	2020-04-30 01:12:12.099382	postgres

MongoDB




C:\mongodb4.2.6\bin\mongo.exe
> show collections
ciudad
colegio
estudiante
> db.estudiantes.find().pretty()
> db.estudiante.find().pretty()
{
"_id" : ObjectId("5eaa7fa50c3288e5c29bd5e0"),
"documento" : "109856324",
"nombre" : "Pedro",
"apellido" : "Perez",
"sexo" : "M",
"grado" : "8"
}
> db.colegio.find().pretty()
{
"_id" : ObjectId("5eaa80480c3288e5c29bd5e1"),
"id" : "15698",
"nombre" : "Normal Superior",
"direccion" : "calle 3 #2-45",
"dimanchoenm" : "126.39",
"dimlargoenm" : "123.78"
}
> db.ciudad.find().pretty()
{
"_id" : ObjectId("5eaa80a70c3288e5c29bd5e2"),
"postal" : "15269",
"nombre" : "Pamplona",
"Temperaturaenc" : "13"
}
>